Aplicación móvil para la gestión de residuos y reducción de la huella de carbono

Pedro Arriola, Oscar López, Yongbum Park, Santiago Taracena y Gabriel Vicente

UVG UNIVERSIDAD DEL VALLE DE GUATEMALA

UNIVERSIDAD DEL VALLE DE GUATEMALA
Facultad de Ingeniería

Aplicación móvil para la gestión de residuos y reducción de la huella de carbono

Trabajo de graduación en modalidad de Megaproyecto Tecnológico
presentado por
Pedro Arriola, Oscar López, Yongbum Park, Santiago Taracena y Gabriel Vicente
para optar al grado académico de Licenciado en Ingeniería en Ciencia de la Computación y Tecnologías de la Información

Guatemala, 2024

UNIVERSIDAD DEL VALLE DE GUATEMALA
Facultad de Ingeniería

Aplicación móvil para la gestión de residuos y reducción de la huella de carbono

Trabajo de graduación en modalidad de Megaproyecto Tecnológico
presentado por
Pedro Arriola, Oscar López, Yongbum Park, Santiago Taracena y Gabriel Vicente
para optar al grado académico de Licenciado en Ingeniería en Ciencia de la Computación y Tecnologías de la Información

Guatemala, 2024

Vo.Bo.:

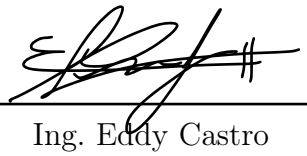
(f)

Ing. Erick Marroquín

Tribunal Examinador:

(f)

Ing. Erick Marroquín

(f) 

Ing. Eddy Castro

(f)

MSc. Douglas Barrios

Fecha de aprobación: Guatemala, 29 de noviembre 2024.

Prefacio

Este proyecto de graduación surge como una respuesta a la creciente necesidad de abordar problemáticas relacionadas con la gestión de residuos y la reducción de la huella de carbono, temas que cobran cada vez mayor relevancia en el contexto de la crisis ambiental global. La iniciativa combina avances tecnológicos y sostenibles para desarrollar soluciones innovadoras que permitan a los usuarios clasificar, gestionar y reducir el impacto ambiental de sus actividades diarias, al tiempo que fomenta la concienciación y la educación ambiental.

El trabajo presentado abarca un enfoque integral que incluye el diseño de una infraestructura en la nube escalable y segura, el desarrollo de modelos avanzados de inteligencia artificial para la clasificación de residuos y la implementación de herramientas de análisis de datos. Estas tecnologías no solo permiten procesar grandes volúmenes de información de manera eficiente, sino que también ofrecen a los usuarios una experiencia práctica y enriquecedora para comprender y mejorar su impacto ambiental. Además, el proyecto incorpora un componente educativo, con el desarrollo de un chatbot interactivo que facilita el acceso a información sobre reciclaje y sostenibilidad.

El desarrollo de esta propuesta ha requerido una colaboración constante entre el equipo de trabajo y actores clave como instituciones locales y expertos en el tema. A lo largo del proceso, se han enfrentado desafíos técnicos y logísticos que fueron superados mediante un aprendizaje continuo y un esfuerzo conjunto. Las soluciones planteadas integran tecnologías de vanguardia, como modelos de aprendizaje profundo y arquitecturas cloud, aplicadas a un contexto específico que busca generar un cambio positivo en la comunidad.

Este megaproyecto no solo representa un logro académico y técnico, sino también un compromiso con el desarrollo sostenible y la responsabilidad ambiental. Los resultados obtenidos reflejan el esfuerzo por crear herramientas tecnológicas innovadoras y accesibles, orientadas a la construcción de un futuro más consciente y sostenible. Confío en que el trabajo aquí presentado servirá como una base sólida para investigaciones y desarrollos futuros, contribuyendo al avance de tecnologías responsables y al fortalecimiento de la conciencia ambiental en nuestra sociedad.

Agradecimientos

Queremos expresar nuestro más sincero agradecimiento a la Universidad, que no solo nos brindó las herramientas y el conocimiento necesario para afrontar este desafío, sino también el entorno propicio para desarrollarnos como profesionales comprometidos con la innovación y el impacto positivo en nuestra sociedad. Su apoyo institucional y académico ha sido fundamental para alcanzar los objetivos de este megaproyecto.

Extendemos nuestra gratitud a nuestros asesores, cuyas orientaciones, experiencia y paciencia fueron pilares indispensables en cada etapa de este proyecto. Su guía constante no solo enriqueció nuestro trabajo, sino también nuestra perspectiva sobre la importancia del rigor técnico, la colaboración y el compromiso profesional.

Agradecemos profundamente a la Municipalidad de Guatemala, especialmente a la Unidad de Reciclaje y al Centro de Educación Ambiental, por su colaboración y respaldo en la realización de este proyecto. Su compromiso con la sostenibilidad y su apertura para compartir información y recursos esenciales han sido clave para el desarrollo de esta solución.

Finalmente, queremos dedicar un agradecimiento especial a nuestras familias y amistades, quienes con su apoyo incondicional, palabras de ánimo y comprensión, han sido un motor constante durante este proceso. Su presencia ha sido fundamental para superar los desafíos y alcanzar las metas trazadas. Este proyecto es un reflejo no solo de nuestro esfuerzo, sino también del amor, la confianza y el respaldo que siempre nos han brindado.

A todos, muchas gracias.

Índice

Prefacio III

Agradecimientos IV

Lista de Figuras XVI

Lista de Cuadros XVIII

Resumen XVIII

1. Introducción 1

2. Objetivos 2
2.1. Objetivo general . . . 2
2.2. Objetivos específicos . . . 2

3. Justificación 4

4. Marco teórico 5
4.1. Marco teórico general . . . 5
4.1.1. Problemática ambiental global y local . . . 5
4.1.2. Gestión de residuos sólidos . . . 7
4.1.3. Reducción de la huella de carbono . . . 9
4.1.4. Tecnología y medio ambiente . . . 10
4.1.5. Aspectos legales y ambientales en Guatemala . . . 12
4.2. Marco teórico sobre experiencia de usuario . . . 14
4.2.1. Diseño centrado en el usuario . . . 14
4.2.2. Experiencia de usuario . . . 17
4.2.3. Técnicas de recopilación de información y muestreo . . . 18
4.2.4. Prototipado y herramientas . . . 22
4.2.5. Desarrollo de aplicaciones móviles . . . 24
4.2.6. Evaluación y métricas de impacto . . . 26
4.3. Marco teórico sobre *backend*, gestión de base de datos y análisis de datos . . . 28
4.3.1. *Backend* . . . 28
4.3.2. Tecnologías para el desarrollo de aplicaciones escalables . . . 28
4.3.3. Bases de datos en la gestión de residuos y clasificación de desechos . . . 31
4.3.4. Desarrollo de una *API* . . . 34
4.3.5. Análisis de datos generados por los usuarios . . . 35

4.3.6. Escalabilidad y rendimiento en el sistema de gestión de residuos 36
4.3.7. *CI/CD*: integración y entrega continua 36
4.3.8. Tecnologías utilizadas en el proyecto 37
4.4. Marco teórico sobre inteligencia artificial 39
4.4.1. Tecnología: *machine learning*, inteligencia artificial y *LLM* 39
4.4.2. Leyes, regulaciones y ética en el uso de la inteligencia artificial 41
4.4.3. Impacto socioeconómico de la IA en la gestión de residuos 43
4.4.4. Futuro y tendencias en IA para la gestión de residuos 43
4.4.5. Desafíos técnicos en la implementación de IA y ML 43
4.4.6. Evaluación de impacto y sostenibilidad 44
4.4.7. Interdisciplinariedad en el desarrollo de IA 44
4.4.8. Aspectos técnicos de *machine learning* 45
4.4.9. Aspectos técnicos de la *inteligencia artificial* y *deep learning* 50
4.4.10. *Explainable AI (XAI)* 62
4.4.11. Modelos de *machine learning* e *inteligencia artificial* en la gestión de residuos 65
4.4.12. Visión por computadora para clasificación de imágenes 67
4.4.13. Modelos de lenguaje de gran escala (*large language models, LLM*) y *chatbots* inteligentes 69
4.4.14. Tecnologías para la implementación de modelos de *inteligencia artificial* . . . 71
4.4.15. *TensorFlow* y visión por computadora con redes neuronales convolucionales (CNN) 74
4.4.16. *Hugging Face* y su rol en la construcción de *chatbots* inteligentes 76
4.4.17. *OpenAI* y su *API* para modelos de lenguaje 78
4.4.18. *FastAPI* y *Jinja2*: herramientas para el desarrollo de aplicaciones web 80
4.4.19. *MLOps* y *MLFlow* 81
4.5. Marco teórico sobre infraestructura en la nube 84
4.5.1. Manejo de desechos y disminución de la huella de carbono 84
4.5.2. Regulaciones y directrices en la gestión de residuos en Guatemala 85
4.5.3. Arquitectura de nube 86
4.5.4. Principales proveedores de infraestructura 89
4.5.5. Avances tecnológicos 90
4.5.6. Costos en la nube y licencias de los proveedores principales 95
4.5.7. Tipos de servicios en la nube 96
4.5.8. Beneficios de desarrollar en la nube vs *On-Premise* 99
4.5.9. Impacto ecológico de desarrollar en la nube 100
4.6. Marco teórico sobre seguridad 100
4.6.1. Seguridad cibernética en el desarrollo de aplicaciones 100
4.6.2. Contexto local: Guatemala 104
4.6.3. Metodologías de desarrollo seguro 108
4.6.4. *OWASP SAMM (Software Assurance Maturity Model)* 111
4.6.5. Herramientas de análisis de seguridad 114
4.6.6. Reglamento General de Protección de Datos (GDPR) 119

5. Metodología **121**
5.1. Metodología de experiencia de usuario 121
5.2. Enfoque metodológico 122
5.3. Diseño del estudio 122
5.4. Asesoría con la municipalidad de Guatemala 123
5.4.1. Asesoría para la definición del alcance de la aplicación 123
5.5. Definición de usuarios 123
5.6. Población y muestra 126
5.7. Investigación de usuarios 127
5.7.1. Encuesta de conocimiento y comportamiento actual sobre gestión de residuos 128
5.7.2. Encuesta de necesidades y expectativas sobre la aplicación 128

5.8. Prototipado de la solución . . . 129
5.8.1. Encuestas sobre el prototipo de la solución . . . 129
5.8.2. Resumen de los cambios realizados en cada iteración . . . 130
5.9. Desarrollo de la interfaz gráfica de la aplicación . . . 132
5.9.1. Priorización de funcionalidades . . . 133
5.10. Evaluación final a la aplicación funcional . . . 133
5.11. Metodología de *backend*, gestión de base de datos y análisis de datos . . . 135
5.12. Elección de tecnologías . . . 135
5.13. Primera iteración . . . 136
5.13.1. Base de datos: diseño versión inicial . . . 136
5.13.2. *API*: definición de las funciones . . . 137
5.14. Segunda iteración . . . 138
5.14.1. Base de datos: diseño de tablas para ubicación y medición . . . 138
5.14.2. *API*: creación de funciones *CRUD* para ubicaciones y mediciones . . . 138
5.15. Tercera iteración . . . 139
5.15.1. Base de datos: diseño de tabla para la zona de la ciudad . . . 139
5.15.2. *API*: creación de modelos y uso de ORM . . . 140
5.16. Cuarta iteración . . . 141
5.16.1. Base de datos: cambios en mediciones de desechos . . . 141
5.16.2. *API*: ampliación de controladores . . . 142
5.16.3. Análisis de datos: creación de métricas iniciales . . . 143
5.17. Quinta iteración . . . 144
5.17.1. Base de datos: creación de tabla para tipos de desecho . . . 144
5.17.2. *API*: implementación de autenticación con *JWT* . . . 145
5.17.3. Análisis de datos: diseño de funciones principales . . . 146
5.18. Sexta iteración . . . 147
5.18.1. Base de datos: creación de tabla de bitácora de auditoría . . . 147
5.18.2. *API*: implementación de registros en bitácora de auditoría . . . 148
5.18.3. Análisis de datos: diseño de funciones de análisis . . . 149
5.19. Séptima iteración . . . 150
5.19.1. Base de datos: optimización y mejora de la estructura . . . 150
5.19.2. *API*: optimización y mejora del *API* . . . 151
5.19.3. Análisis de datos: creación de las rutas de análisis en *API* . . . 152
5.20. Octava iteración . . . 153
5.20.1. Análisis de datos: creación de análisis de datos global de los usuarios . . . 153
5.21. Implementación de *CI/CD* . . . 154
5.21.1. Pruebas automatizadas . . . 155
5.21.2. Manejo y monitoreo de errores con *Sentry* . . . 159
5.21.3. Implementación y automatización con *GitHub Actions* . . . 161
5.22. Metodología de infraestructura en la nube . . . 162
5.23. Identificación de los recursos necesarios . . . 163
5.23.1. Evaluación de las pantallas . . . 163
5.23.2. Requerimientos derivados del prototipo . . . 165
5.24. Comparación de proveedores de nube . . . 166
5.24.1. *Templates*, paquetes y *all in one* . . . 176
5.25. Implementación y configuración de recursos . . . 178
5.25.1. Comparativa entre desarrollos . . . 182
5.25.2. Configuración de inicio de sesión con *Google* y *Facebook* . . . 183
5.25.3. Despliegue de un modelo de IA . . . 187
5.25.4. Análisis de costos y mantenimiento . . . 189
5.26. Estimación de costos y mantenimiento para diferentes escenarios de afluencia . . . 189
5.26.1. Interfaz de programación de aplicaciones desarrollada . . . 189
5.26.2. Base de datos relacional . . . 190
5.26.3. Modelo de inteligencia artificial . . . 190

5.26.4. Servicios de autenticación . . . 191
5.26.5. Almacenamiento de datos . . . 191
5.26.6. Resumen de costos mensuales estimados . . . 191
5.26.7. Consideraciones de mantenimiento . . . 192
5.27. Metodología de seguridad . . . 192
5.28. Fase 1: selección de tecnologías . . . 192
5.28.1. Herramientas de análisis estático de código . . . 192
5.28.2. Herramientas de análisis dinámico de código para aplicaciones móviles . . . 193
5.29. Fase 2: preparación e instalación de herramientas . . . 195
5.29.1. Reuniones semanales de seguimiento y planificación . . . 198
5.30. Fase 3: definición de la estrategia de seguridad . . . 198
5.30.1. Gobernanza: educación y orientación . . . 198
5.30.2. Diseño: arquitectura segura . . . 198
5.30.3. Implementación: gestión de defectos . . . 198
5.30.4. Verificación: pruebas de seguridad . . . 199
5.31. Fase 4: evaluaciones de seguridad y mejora continua . . . 199
5.31.1. Capacitación del equipo en *clean code* y *OWASP Top* 10 . . . 199
5.31.2. Reuniones semanales de seguimiento y planificación . . . 200
5.31.3. Prueba de *phishing* . . . 200
5.31.4. Análisis estático del código . . . 201
5.31.5. Migración a *SonarCloud* y resultados de cobertura . . . 202
5.31.6. Configuración de *CI/CD* en *SonarCloud* . . . 206
5.31.7. Revisión de seguridad: detección y mitigación de vulnerabilidades . . . 207
5.31.8. Evaluación de seguridad en modelos de *IA* y *chatbot* . . . 208
5.32. Fase 5: Escaneo y resultados de la aplicación móvil . . . 210
5.32.1. Escaneo estático con *MobSF* . . . 210
5.32.2. Escaneo dinámico con *AppSweep* . . . 212
5.33. Fase 6: revisión y optimización de permisos en *AWS IAM Identity Center* . . . 213
5.33.1. Revisión de grupos y permisos . . . 213
5.33.2. Proceso de validación y ajuste de permisos . . . 214
5.34. Metodología de inteligencia artificial . . . 215
5.35. Selección de tecnologías . . . 215
5.35.1. Plataforma de entrenamiento y desarrollo: *Google Colab* . . . 215
5.35.2. *Framework* de redes neuronales: *TensorFlow* . . . 216
5.35.3. Manipulación y preprocesamiento de imágenes: *PILLOW* . . . 216
5.35.4. Modelo de chat inteligente: *OpenAI API* con *GPT*-3.5 Turbo . . . 216
5.35.5. *Framework* para *endpoints* de chat: *FastAPI* . . . 217
5.35.6. Resumen de ventajas de las tecnologías seleccionadas . . . 217
5.36. Descripción del *dataset* Utilizado . . . 217
5.36.1. Características principales del *dataset* . . . 217
5.36.2. Relevancia del *dataset* en el contexto Guatemalteco . . . 218
5.37. Ejemplos visuales de imágenes en el *dataset* . . . 218
5.38. Justificación del uso de Redes Neuronales Convolucionales (*CNN*) y *Transfer Learning* 219
5.38.1. Ventajas del uso de *CNNs* en clasificación de imágenes . . . 219
5.38.2. *Transfer Learning* para mejorar la precisión del modelo . . . 220
5.39. Justificación del Uso de Redes Neuronales Convolucionales (*CNN*), *Transfer Learning* y *Fine-Tuning* . . . 220
5.39.1. Ventajas del Uso de *CNNs* en Clasificación de Imágenes . . . 220
5.39.2. *Transfer Learning* y *Fine-Tuning* para Mejorar la Precisión del Modelo . . . 221
5.40. Inspiración en estudios anteriores . . . 221
5.40.1. Estudios relacionados y modelos similares . . . 221
5.40.2. Aplicación de las metodologías inspiradas . . . 221
5.41. Metodología para la selección de modelos y preparación de datos . . . 222
5.41.1. Preparación del *dataset*: *undersampling* y preprocesamiento . . . 222

5.41.2. Preprocesamiento específico de imágenes para cada modelo . . . 222
5.41.3. Implementación y código para la preparación de datos de entrenamiento . . . 223
5.41.4. Visualización de imágenes preprocesadas para cada modelo . . . 224
5.42. Definición y Configuración de la Arquitectura de los Modelos de *Transfer Learning* . 226
5.42.1. Arquitectura base y capas congeladas . . . 226
5.42.2. Capas superiores (*Top Model*) . . . 226
5.42.3. Implementación y configuración de cada modelo . . . 227
5.42.4. Justificación de la arquitectura y las capas superiores . . . 229
5.43. Entrenamientos preliminares para la selección del modelo definitivo . . . 229
5.43.1. Hiperparámetros de entrenamiento . . . 229
5.43.2. Optimizador y tasa de aprendizaje . . . 229
5.43.3. Uso de *GPU* para acelerar el entrenamiento . . . 230
5.43.4. *Callbacks* utilizados . . . 230
5.43.5. Evaluación de rendimiento y selección del modelo definitivo . . . 230
5.43.6. Resumen de hiperparámetros y estrategias . . . 231
5.44. Análisis de métricas de rendimiento y selección del modelo final . . . 231
5.44.1. Evaluación de métricas de rendimiento . . . 231
5.44.2. Determinación de la métrica clave . . . 231
5.44.3. Selección de *Xception* como modelo final . . . 231
5.44.4. Conclusiones de la validación y selección del modelo final . . . 232
5.45. Entrenamiento final con el modelo *Xception* . . . 232
5.45.1. Evaluación visual y análisis del modelo *Xception* . . . 234
5.45.2. Estrategias de interpretabilidad . . . 235
5.45.3. Conclusión de la metodología . . . 236
5.46. Metodología de implementación y evaluación del *chatbot* inteligente . . . 237
5.46.1. Implementación del *chatbot* inteligente . . . 237
5.46.2. Evaluación del rendimiento del *chatbot* . . . 238
5.46.3. Interfaz gráfica para pruebas externas . . . 239
5.46.4. Conclusión de la metodología de evaluación . . . 241

6. Resultados **242**
6.1. Resultados de experiencia de usuario . . . 242
6.1.1. Investigación inicial de usuarios . . . 242
6.1.2. Desarrollo iterativo del prototipo . . . 252
6.1.3. Evaluación final a la aplicación funcional . . . 255
6.2. Resultados de *backend*, gestión de base de datos y análisis de datos . . . 256
6.2.1. Diseño y estructura de la base de datos . . . 257
6.2.2. *API* . . . 258
6.2.3. Análisis de datos . . . 273
6.2.4. Implementación de *CI/CD* . . . 276
6.3. Resultados de inteligencia artificial . . . 282
6.3.1. Resultados de *Xception* . . . 282
6.3.2. Resultados del *chatbot* inteligente . . . 301
6.4. Resultados de seguridad . . . 306
6.4.1. Educación y orientación . . . 306
6.4.2. Calidad del código y mejora continua . . . 307
6.4.3. Escaneo de vulnerabilidades en aplicación móvil . . . 314
6.4.4. Análisis del código de los modelos de IA . . . 315
6.4.5. Análisis de permisos en *AWS* . . . 316
6.5. Resultados de infraestructura en la nube . . . 320
6.5.1. Recursos utilizados . . . 320
6.5.2. Resultados de funcionalidad . . . 321
6.5.3. Diagrama de infraestructura final . . . 326
6.5.4. Pruebas de rendimiento . . . 326

6.5.5. Plan de mantenimiento y costos . . . 329
6.5.6. Consideraciones de mantenimiento . . . 331

7. Análisis de resultados 332
7.1. Análisis de resultados de experiencia de usuario . . . 332
7.1.1. Usuarios identificados . . . 332
7.1.2. Prototipado iterativo . . . 334
7.1.3. Evaluación final a la aplicación . . . 336
7.2. Análisis de resultados de *backend*, gestión de base de datos y análisis de datos . . . 336
7.2.1. Elección de las tecnologías . . . 336
7.2.2. Diseño y estructura de la base de datos . . . 337
7.2.3. *API* . . . 337
7.2.4. Análisis de datos . . . 338
7.2.5. Implementación de *CI/CD* . . . 338
7.3. Análisis de resultados de inteligencia artificial . . . 339
7.3.1. Análisis de resultados del modelo *Xception* . . . 339
7.3.2. Análisis de resultados del *chatbot* inteligente . . . 349
7.3.3. Implicaciones de los agentes inteligentes en la Ciudad de Guatemala . . . 352
7.4. Resultados de seguridad . . . 354
7.4.1. Cultura de seguridad y capacitación en el equipo . . . 354
7.4.2. Reducción de vulnerabilidades mediante *SonarQube* y *SonarCloud* . . . 354
7.4.3. Exclusiones de cobertura y configuración de calidad de código . . . 355
7.4.4. *Code smells* y resolución . . . 355
7.4.5. Código duplicado y principio *DRY* . . . 356
7.4.6. Complejidad ciclomática . . . 356
7.4.7. Densidad de comentarios . . . 356
7.4.8. Integración de seguridad en componentes de inteligencia artificial y *chatbot* . . . 357
7.4.9. Análisis de vulnerabilidades en la aplicación móvil con *AppSweep* y *MobSF* . . . 357
7.4.10. Configuraciones de permisos en *AWS* . . . 358
7.5. Resultados de infraestructura en la nube . . . 359
7.5.1. *AWS Elastic Beanstalk* y *Amazon EC2* . . . 359

8. Conclusiones 363
8.1. Conclusiones de experiencia de usuario . . . 363
8.2. Conclusiones de *backend*, gestión de base de datos y análisis de datos . . . 364
8.3. Conclusiones de inteligencia artificial . . . 365
8.4. Conclusiones de seguridad . . . 365
8.5. Conclusiones de infraestructura en la nube . . . 366

9. Recomendaciones 368
9.1. Recomendaciones de experiencia de usuario . . . 368
9.2. Recomendaciones de *backend*, gestión de base de datos y análisis de datos . . . 368
9.3. Recomendaciones de inteligencia artificial . . . 369
9.4. Recomendaciones de seguridad . . . 370
9.5. Recomendaciones de infraestructura en la nube . . . 371

Bibliografía 395

Lista de Figuras

4.0. Composición general de los residuos y desechos sólidos comunes en Guatemala [1] . . 6
4.1. Tipos y clasificación de residuos [2] 7
4.2. Diagrama sobre el concepto de huella de carbono y su relevancia [3] 10
4.3. Diagrama de las fases del diseño centrado en el usuario 15
4.4. Diagrama sobre la metodología y flujo de trabajo de *design thinking* [4] 16
4.5. Ejemplo de prototipado de baja fidelidad en formato de *wireframe* 23
4.6. Ejemplo de prototipado de alta fidelidad desarrollado en *Figma* 23
4.7. Arquitectura cliente-servidor: comunicación entre *frontend* y *backend* [5] 29
4.8. Diagrama de comunicación *REST API*. Fuente: [6] 30
4.9. Diagrama de la base de datos. Elaboración propia 32
4.10. Representación de la Segunda Forma Normal (*Second Normal Form*). Fuente: [7] . . 33
4.11. Matriz de confusión que ilustra la precisión del clasificador a través de verdaderos positivos, falsos positivos, verdaderos negativos y falsos negativos. 47
4.12. Curva *ROC* que muestra el desempeño de un clasificador binario. 47
4.13. Diagrama de árbol de decisión que representa las decisiones y sus posibles consecuencias. 48
4.14. Arquitectura del modelo *MobileNetV2*. 56
4.15. Arquitectura del modelo *InceptionV3*. 57
4.16. Arquitectura del modelo *DenseNet201*. 59
4.17. Arquitectura del modelo *VGG19*. 60
4.18. Arquitectura del modelo *Xception*. 61
4.19. Ejemplo de un Árbol de Decisión utilizado para la Interpretabilidad. 63
4.20. Ejemplo de *Grad-CAM* mostrando las áreas de atención del modelo. 63
4.21. Ejemplo de *LIME* explicando las decisiones de un modelo de clasificación. 64
4.22. Ejemplo de *SHAP* mostrando la contribución de cada variable en una predicción. . . 64
4.23. Ejemplo de *SmoothGrad* reduciendo el ruido en los mapas de saliencia. 64
4.24. Ejemplo de un mapa de calor que muestra las áreas de una imagen que más contribuyen a la clasificación. 69
4.25. Ejemplo de un mapa de atención en un LLM, mostrando la relevancia de diferentes palabras en un contexto específico. 71

5.1. Diagrama de resumen de los pasos del proyecto 122
5.2. Asesorías con la Municipalidad de manera remota 124
5.3. Distribución poblacional en Guatemala el año 2020 [8] 124
5.4. Mapa de empatía de un estudiante respecto al tema de reciclaje y gestión de residuos 126
5.5. Agregación de una pantalla de configuración de direcciones durante la primera iteración del prototipo 131

5.6. Mejoras en la pantalla de configuración de información de los usuarios 131
5.7. Cambios en la paleta de colores para indicar acciones importantes 131
5.8. Pantallas de la aplicación en desarrollo 134
5.9. Diagrama entidad-relación de la primera iteración 137
5.10. Diagrama entidad-relación de la segunda iteración 138
5.11. Diagrama entidad-relación de la tercera iteración 140
5.12. Diagrama entidad-relación de la cuarta iteración 142
5.13. Reunión para Estructurar el Análisis de Datos 143
5.14. Diagrama entidad-relación de la quinta iteración 145
5.15. Diagrama entidad-relación de la sexta iteración 148
5.16. Diagrama entidad-relación de la séptima iteración 151
5.17. Ejemplo de solicitudes en origines de conexión 154
5.18. Tablas resultantes de la conexión del *API* 154
5.19. *Pipeline* de *CI/CD* [9] 155
5.20. Tablero del proyecto en *Sentry* 160
5.21. Pantallas *login* - *signin* 163
5.22. Opciones usuario 164
5.23. Pantalla estadísticas 164
5.24. Formulario de ingesta 165
5.25. Pantallas Inteligencia artificial 165
5.26. Resultado de *scan* del 5 de agosto. 195
5.27. Página de inicio de *AppSweep*. 196
5.28. Página de carga de archivos en *MobSF*. 197
5.29. Resultados de phishing del equipo. 201
5.30. Resultado de *scan* del 14 de agosto. 202
5.31. Primer escaneo en *SonarCloud*. 203
5.32. Configuración de exclusiones en *SonarCloud*. 204
5.33. Configuración de calidad en *SonarCloud*. 205
5.34. Cobertura del código en *SonarCloud*. 206
5.35. Grafica de cobertura en *sonarCloude*. 206
5.36. Configuración de *SonarCloud*. 207
5.37. Gráfica de vulnerabilidades detectadas en *SonarQube*. 207
5.38. Gráfica de vulnerabilidades detectadas en *SonarCloud*. 208
5.39. Resultado del escaneo del 20 de octubre para el segundo modelo. 208
5.40. Primer escaneo del *chatbot* en *SonarCloud*. 209
5.41. Último escaneo del *chatbot* en *SonarCloud*. 210
5.42. Repositorio de *MobSF*. 211
5.43. *MobSF* ejecutándose localmente. 211
5.44. Interfaz gráfica de *MobSF* 212
5.45. Resultado del escaneo inicial en *MobSF*. 212
5.46. Resultado del escaneo en *AppSweep*. 213
5.47. *AWS IAM Identity Center*: cuentas y grupos predeterminados. 214
5.48. *AWS IAM Identity Center*: permisos de usuario revisados. 214
5.49. Ejemplos de imágenes de cada clase en el *dataset*: plástico, papel, orgánico, metal, vidrio y baterías 219
5.50. Imágenes preprocesadas para *MobileNetV2* 224
5.51. Imágenes preprocesadas para *InceptionV3* 225
5.52. Imágenes preprocesadas para *DenseNet201* 225
5.53. Imágenes preprocesadas para *VGG19* 225
5.54. Imágenes preprocesadas para *Xception* 226
5.55. Imágenes reales utilizadas para probar el modelo Xception en un contexto cotidiano 235
5.56. *Saliency maps* generados con *SmoothGRAD* para interpretar las decisiones del modelo *Xception* en distintas clases de residuos 236
5.57. Interfaz gráfica desarrollada para pruebas del *chatbot* 240

6.1. Resultados a la pregunta: ¿Cuál es tu rango de edad? . . . 243
6.2. Resultados a la pregunta: ¿En qué municipio vives? . . . 243
6.3. Resultados a la pregunta: ¿Cuál es tu nivel educativo? . . . 243
6.4. Resultados a la pregunta: ¿Cuánto sabes sobre la clasificación de residuos? . . . 244
6.5. Resultados a la pregunta: ¿Sabes identificar los diferentes tipos de residuos (orgánicos, reciclables, peligrosos, etc.)? . . . 244
6.6. Resultados a la pregunta: ¿Estás familiarizado/a con los símbolos de reciclaje en los productos? . . . 244
6.7. Resultados a la pregunta: ¿Conoces las políticas locales de gestión de residuos en tu comunidad? . . . 245
6.8. Resultados a la pregunta: ¿Sabes a dónde va la basura que produces? . . . 245
6.9. Resultados a la pregunta: ¿Con qué frecuencia reciclas? . . . 245
6.10. Resultados a la pregunta: ¿Tienes un sistema de clasificación de residuos en tu hogar? 246
6.11. Resultados a la pregunta: ¿Qué tipo de residuos clasificas regularmente? . . . 246
6.12. Resultados a la pregunta: ¿Utilizas alguna aplicación o recurso digital para ayudarte en la gestión de residuos? . . . 246
6.13. Resultados a la pregunta: ¿Qué es lo más desafiante al momento de gestionar tus recursos? . . . 247
6.14. Resultados a la pregunta: ¿Has recibido alguna educación formal o informal sobre la gestión de residuos? . . . 247
6.15. Resultados a la pregunta: ¿Qué tan importante crees que es reciclar y gestionar los residuos correctamente? . . . 247
6.16. Resultados a la pregunta: ¿Crees que gestionar bien los residuos puede tener un impacto positivo en el medio ambiente? . . . 248
6.17. Resultados a la pregunta: ¿Estarías dispuesto/a a cambiar tus hábitos de consumo si supieras que reduce tu huella de carbono? . . . 248
6.18. Resultados a la pregunta: ¿Qué funcionalidades te gustaría que tuviera una aplicación de gestión de residuos? . . . 249
6.19. Resultados a la pregunta: ¿Qué tipo de información te gustaría recibir de la aplicación? 249
6.20. Resultados a la pregunta: ¿Qué te motivaría a usar una aplicación de gestión de residuos regularmente? . . . 249
6.21. Resultados a la pregunta: ¿Qué tan importante es para ti que la aplicación sea fácil de usar? . . . 250
6.22. Resultados a la pregunta: ¿Cómo preferirías que la aplicación te notificara sobre la gestión de tus residuos? . . . 250
6.23. Resultados a la pregunta: ¿Qué tan importante es la personalización en la experiencia de la aplicación? . . . 250
6.24. Resultados a la pregunta: ¿Qué te parecería más útil en una aplicación móvil? . . . 251
6.25. Resultados a la pregunta: ¿Qué tan importante es para ti que la aplicación tenga un diseño visualmente atractivo? . . . 251
6.26. Resultados a la pregunta: ¿Qué dispositivo usarías principalmente para acceder a la aplicación? . . . 251
6.27. Resultados de las calificaciones de la primera iteración del prototipo . . . 252
6.28. Resultados de las calificaciones de la primera iteración del prototipo . . . 253
6.29. Resultados de las calificaciones de la primera iteración del prototipo . . . 254
6.30. Resultados del *eye tracking* en pantallas como la principal y la de configuración de perfil . . . 256
6.31. Diagrama entidad-relación final . . . 258
6.32. Registro de usuarios . . . 259
6.33. *Login* de usuarios . . . 260
6.34. Registro en la bitácora de auditoría . . . 260
6.35. Intento de acceso no autorizado . . . 260
6.36. Consulta de análisis para usuarios autorizados . . . 261
6.37. *Tests* realizados al *API* . . . 262

6.38. Resultado de los *tests* acertados . . . 263
6.39. Resultados prueba de carga del 20 % . . . 264
6.40. Resultados prueba de carga del 20 % . . . 264
6.41. Resultados prueba de carga del 20 % . . . 265
6.42. Resultados prueba de carga del 20 % . . . 266
6.43. Resultados prueba de carga del 50 % . . . 267
6.44. Resultados prueba de carga del 50 % . . . 267
6.45. Resultados prueba de carga del 50 % . . . 268
6.46. Resultados prueba de carga del 50 % . . . 269
6.47. Resultados de la prueba de carga para el 100 % de la población . . . 270
6.48. Resultados de la prueba de carga para el 100 % de la población . . . 270
6.49. Resultados de la prueba de carga para el 100 % de la población . . . 271
6.50. Resumen de resultados de la prueba de carga para el 100 % . . . 272
6.51. Resultado top 5 desechos usuario . . . 273
6.52. Resultado porcentaje comparación usuario . . . 274
6.53. Resultado porcentaje comparación usuario . . . 274
6.54. Resultado top 5 desechos generales . . . 275
6.55. Resultado desechos reciclables generales . . . 275
6.56. Resultado CO2 y Agua generales . . . 276
6.57. Resultado de los *tests* acertados . . . 277
6.58. Resultado del *action* de los *tests* acertados . . . 277
6.59. Resultado de los *tests* fallados . . . 278
6.60. Resultado del *action* de los *tests* fallados . . . 278
6.61. Resultado de los *tests* fallados en el *mail* . . . 279
6.62. Resultado de los errores de *Sentry* . . . 279
6.63. Resultado de la descripcion de un error en *Sentry* . . . 280
6.64. Resultado una notificacion de error . . . 280
6.65. Resultado del apartado de *performance* de *Sentry* . . . 280
6.66. Resultado del detalle de descripción de *performance* de *Sentry* . . . 281
6.67. Resultado de los traces de *Sentry* . . . 281
6.68. Resultado de *GitHub Actions* . . . 282
6.69. Resultado del *deploy* . . . 282
6.70. Evolución de la precisión del modelo durante el entrenamiento y validación (*Xception* Final) . . . 283
6.71. Evolución de la pérdida del modelo durante el entrenamiento y validación (*Xception* Final) . . . 283
6.72. Matriz de confusión del modelo *Xception* en el conjunto de prueba . . . 284
6.73. Interfaz de *MLFlow* mostrando los experimentos registrados. Aquí se observan las ejecuciones del modelo *Xception* y sus correspondientes métricas y parámetros . . . 285
6.74. Detalles del experimento *Xception Model Training* en *MLFlow*. Se observan las métricas y los parámetros registrados durante el entrenamiento del modelo . . . 285
6.75. Detalles del experimento Entrenamiento con *Xception* y *GradCAM* en *MLFlow*. Se destacan las métricas de precisión y pérdida, junto con los parámetros utilizados . . . 286
6.76. Ejemplos combinados de mapas de atención generados con *GradCAM* para las diferentes clases de residuos. Se observan las áreas de atención del modelo, que se enfocan en las características clave de cada residuo, como bordes, texturas y patrones específicos, esenciales para la correcta clasificación . . . 287
6.77. Explicación generada con *LIME* para la clase *Battery*. Las áreas resaltadas muestran las características que más influyeron en la predicción, indicando bordes y regiones específicas . . . 288
6.78. Explicación generada con *LIME* para la clase *Glass*. Las áreas resaltadas indican las texturas y bordes que el modelo consideró importantes para clasificar correctamente el residuo . . . 289

6.79. Explicación generada con *LIME* para la clase *Metal*. Las áreas destacadas muestran las regiones clave que ayudaron al modelo a identificar las propiedades del metal, como el brillo y las texturas metálicas 290
6.80. Explicación generada con *LIME* para la clase *Organic*. Las áreas resaltadas indican las texturas y patrones orgánicos que fueron más relevantes para la predicción 291
6.81. Explicación generada con *LIME* para la clase *Paper*. Las áreas importantes se centran en las texturas y bordes del papel que influenciaron la decisión del modelo 292
6.82. Explicación generada con *LIME* para la clase *Plastic*. Las áreas resaltadas muestran las regiones específicas que el modelo utilizó para identificar el plástico, como los contornos y las variaciones en la textura 293
6.83. *Saliency Map* generado con *SmoothGrad* para la clase *Battery* 293
6.84. *Saliency Map* generado con *SmoothGrad* para la clase *Glass*. 294
6.85. *Saliency Map* generado con *SmoothGrad* para la clase *Metal* 294
6.86. *Saliency Map* generado con *SmoothGrad* para la clase *Organic* 294
6.87. *Saliency Map* generado con *SmoothGrad* para la clase *Paper* 295
6.88. *Saliency Map* generado con *SmoothGrad* para la clase *Plastic* 295
6.89. Clasificación de un residuo real perteneciente a la clase *Papel* 295
6.90. Clasificación de un residuo real perteneciente a la clase *Metal* 296
6.91. Clasificación de un residuo real perteneciente a la clase *Plastic* 296
6.92. Clasificación de un residuo real perteneciente a la clase *Plastic* 296
6.93. Clasificación de un residuo real perteneciente a la clase *Organic* 297
6.94. Clasificación de un residuo real perteneciente a la clase *Glass* 297
6.95. Clasificación de un residuo real perteneciente a la clase *Battery* 297
6.96. Clasificación de un residuo real perteneciente a la clase *Paper* 298
6.97. Clasificación de un residuo real perteneciente a la clase *Metal*. 298
6.98. Clasificación de un residuo real perteneciente a la clase *Organic* 298
6.99. Clasificación de un residuo real perteneciente a la clase *Metal* 299
6.100Clasificación de un residuo real perteneciente a la clase *Plastic* 299
6.101Clasificación de un residuo real perteneciente a la clase *Glass* 299
6.102Clasificación de un residuo real perteneciente a la clase *Plastic* 300
6.103Clasificación de un residuo real perteneciente a la clase *Plastic*, pero que fue clasificado como *Paper* 300
6.104Clasificación de un residuo real perteneciente a la clase *Organic* 300
6.105Evaluación de la precisión de las respuestas del *chatbot* 302
6.106Evaluación de la relevancia temática de las respuestas 302
6.107Evaluación de la capacidad del *chatbot* para rechazar preguntas fuera de tema 303
6.108Evaluación de la calidad de las recomendaciones proporcionadas 303
6.109Evaluación de la consistencia en el estilo de las respuestas 304
6.110Tiempo promedio de respuesta del *chatbot*. 305
6.111Gráfica de vulnerabilidades detectadas en *SonarQube* 308
6.112Gráfica de vulnerabilidades detectadas en *SonarCloud* 309
6.113Cobertura del código en *SonarCloud* 309
6.114Gráfica de cobertura en *SonarCloud* 310
6.115Gráfica de *code smell* en *SonarCloud* 311
6.116Gráfica de líneas de código duplicadas en *SonarCloud* 312
6.117Gráfica de complejidad en *SonarCloud* 313
6.118Gráfica de densidad de comentarios en *SonarCloud* 314
6.119Vulnerabilidades detectadas en *AppSweep* 315
6.120Permisos del rol *DatabaseAdministrator* asignados a Oscar López 317
6.121Permisos del rol *SystemAdministrator* asignados a Pedro Pablo 318
6.122Permisos del rol *ReadOnlyAccess* asignados a Santiago Taracena y Yongbum Park . 319
6.123Usuarios *IAM Identity Center* 321
6.124Administración de cuenta única por múltiples usuarios 321
6.125Configuración de políticas para cada usuario 322

6.126Entorno de creación de *Elastic Beanstalk* 322
6.127Estado de producción de interfaz de programación de aplicación 322
6.128EC2 que aloja *API Express* 323
6.129Base de datos disponibles para *Elastic Beanstalk* 323
6.130Conexión para base de datos de producción 324
6.131Configuracion de perfil para *Sage Maker* 324
6.132*Deployment* de modelo h5 para producción 325
6.133*Endpoints* disponibles para consultar el modelo 325
6.134Diagrama de infraestructura en la nube 326
6.135Prueba de rendimiento, 70 solicitudes por segundo 327
6.136Prueba de rendimiento, 35 solicitudes por segundo 328
6.137Resultados sobre las pruebas de rendimiento 329

7.1. Diagrama de roles y responsabilidades del equipo 358

Lista de Cuadros

4.0. Resumen sobre métodos y prácticas actuales de reciclaje y gestión de residuos 8
4.1. Estrategias para la reducción de la huella de carbono 11
4.2. Resumen de conceptos y principios del diseño centrado en el usuario 15
4.3. Técnicas de recopilación de información y muestreo 19
4.4. Resumen de conceptos en el desarrollo de aplicaciones móviles 25
4.5. Comparación entre bases de datos relacionales y no relacionales 30
4.6. Etapas del SDLC y medidas de seguridad. 110

5.1. Perfiles de Usuarios 125
5.2. Comparativa de herramientas de diseño para prototipos 130
5.3. Comparativa de tecnologías para desarrollo móvil 133
5.4. Comparación de Servicios *IAM* entre *AWS*, *Azure*, *Google Cloud* y *AWS IAM Identity Center* 167
5.5. Comparación de Servicios *CIAM* entre *AWS, Azure* y *Google Cloud* 168
5.6. Comparación de Servicios *SQL* entre *AWS, Azure* y *Google Cloud* 171
5.7. Comparación de servicios de *API* entre *AWS*, *Azure* y *Google Cloud Platform* 173
5.8. Comparación de servicios de inteligencia artificial entre *AWS, Azure* y *Google Cloud Platform* 175
5.9. Comparación de Soluciones Integrales de Despliegue de Aplicaciones entre *AWS, Azure* y *Google Cloud Platform* 177
5.10. Métricas de rendimiento de cada modelo evaluado en el conjunto de validación . . . 231
5.11. Métricas de evaluación detalladas para el modelo *Xception* 232
5.12. Distribución de datos de entrenamiento y validación 233

6.1. Comentarios y retroalimentación sobre la primera iteración del prototipo 253
6.2. Comentarios y retroalimentación sobre la segunda iteración del prototipo 254
6.3. Comentarios y retroalimentación sobre la tercera iteración del prototipo 255
6.4. Resultados de las pruebas de usabilidad 255
6.5. Resultados de pruebas de carga a diferentes tasas de solicitudes 272
6.6. Comparación de rendimiento en tres escenarios de carga en API de producción . . . 273
6.7. Reporte de clasificación del modelo *Xception* en el conjunto de prueba 284
6.8. Puntaje total y porcentaje de precisión de cada participante en la evaluación del *chatbot* 304
6.9. Tiempos de respuesta del *chatbot* en cada prueba realizada 305
6.10. Pruebas de interacción con el *chatbot* y sus respuestas 306
6.11. Resultados de los *tests* de *phishing* 307

7.1. Ejemplos de interacciones con el *chatbot* y su análisis 351

Resumen

Este proyecto de graduación presenta una solución tecnológica integral orientada a abordar los desafíos actuales relacionados con la gestión de residuos y la reducción de la huella de carbono. La propuesta combina herramientas de última generación en inteligencia artificial, infraestructura en la nube y análisis de datos, con el objetivo de ofrecer una plataforma accesible y escalable que facilite el manejo responsable de desechos y promueva prácticas sostenibles entre sus usuarios.

La solución desarrollada incluye una aplicación móvil que permite a los usuarios registrar y clasificar residuos, calcular su impacto ambiental y acceder a recomendaciones personalizadas para reducir su huella de carbono. Este sistema se apoya en una arquitectura *cloud* diseñada para garantizar la seguridad, disponibilidad y eficiencia en el procesamiento de grandes volúmenes de datos, así como en modelos avanzados de aprendizaje profundo y un *chatbot* interactivo que promueve la educación ambiental y la concienciación ciudadana.

El proyecto aborda tanto los aspectos técnicos, como el diseño e implementación de los componentes de *software* y *hardware* necesarios, como también los desafíos logísticos y sociales asociados a la gestión de residuos en un contexto urbano. Además, se incluyen pruebas de rendimiento, análisis de costos y estrategias de mejora continua que aseguran la viabilidad y sostenibilidad a largo plazo de la solución propuesta.

En conjunto, este trabajo representa un esfuerzo por integrar tecnología, sostenibilidad y educación en una herramienta práctica que contribuya al bienestar de la sociedad y al cuidado del medio ambiente, sentando las bases para futuras investigaciones y aplicaciones en este ámbito crítico.

CAPÍTULO 1

Introducción

La gestión de residuos y la reducción de la huella de carbono se han convertido en temas prioritarios en la agenda global, especialmente ante los desafíos ambientales derivados del crecimiento urbano, el aumento en la generación de desechos y el impacto del cambio climático. En este contexto, las tecnologías emergentes ofrecen una oportunidad única para desarrollar soluciones innovadoras que contribuyan a la sostenibilidad y promuevan la conciencia ambiental.

Este proyecto tiene como propósito diseñar e implementar una solución tecnológica que combine herramientas avanzadas de inteligencia artificial, análisis de datos e infraestructura en la nube para abordar los desafíos asociados a la clasificación, manejo y reducción de residuos. A través de una aplicación móvil accesible y escalable, se busca empoderar a los usuarios con herramientas para gestionar de forma responsable sus desechos, evaluar el impacto ambiental de sus actividades y adoptar prácticas más sostenibles.

La propuesta presentada en este informe no solo aborda los aspectos técnicos de diseño e implementación, sino también los desafíos sociales y culturales que influyen en la adopción de prácticas sostenibles. El trabajo incluye un análisis detallado de las necesidades del usuario, el desarrollo de una arquitectura tecnológica robusta y la integración de funcionalidades como modelos de aprendizaje profundo y *chatbots* interactivos.

En las siguientes secciones, se expone de manera integral el proceso de desarrollo del proyecto, desde la conceptualización y diseño, hasta la implementación y evaluación de la solución final. Este informe refleja el compromiso por contribuir al cuidado del medio ambiente mediante una herramienta práctica e innovadora, cuyo objetivo final es fomentar un cambio positivo en la gestión de residuos y la concienciación ambiental.

CAPÍTULO 2

Objetivos

2.1. Objetivo general

Desarrollar una aplicación móvil integral que promueva prácticas sostenibles de reciclaje en Guatemala, abordando la problemática de la producción desmedida de basura y su incorrecto procesamiento mediante la implementación de módulos especializados. Estos incluyen: un *backend* robusto con gestión eficiente de bases de datos y análisis de datos, una arquitectura en la nube escalable, altos estándares de seguridad en todo el ciclo de vida del *software*, y herramientas de inteligencia artificial para la clasificación de residuos y asistencia a los usuarios a través de un chat inteligente.

2.2. Objetivos específicos

- Diseñar una interfaz de usuario intuitiva y atractiva para la aplicación móvil, basada en las mejores prácticas de diseño centrado en el usuario y adaptada a las necesidades y preferencias de la audiencia objetivo investigada.
- Desarrollar la interfaz gráfica de la aplicación móvil, priorizando aquellas funcionalidades que tengan un impacto directo en la reducción de residuos y la promoción de prácticas sostenibles.
- Evaluar la efectividad y la usabilidad de la aplicación mediante pruebas piloto con un grupo seleccionado de usuarios, así como a través de encuestas de satisfacción y retroalimentación cualitativa.
- Investigar y seleccionar las tecnologías más adecuadas para el desarrollo de la aplicación, considerando escalabilidad y compatibilidad con dispositivos móviles.
- Crear una estructura sólida para almacenar datos sobre los tipos de residuos y la frecuencia de desecho de los usuarios.
- Desarrollar un sistema de base de datos que permita almacenar y gestionar la información sobre los desechos clasificados por los usuarios y los datos relacionados con la gestión de residuos.
- Desarrollar una *API* que permita la comunicación entre el *frontend* y la base de datos, garantizando la integridad de los datos transmitidos.

- Procesar y analizar los datos generados por los usuarios para obtener métricas y tendencias útiles que los usuarios puedan utilizar para mejorar sus prácticas de gestión de residuos.
- Realizar un análisis de necesidades y especificaciones técnicas para determinar los recursos necesarios en la nube.
- Crear un diseño detallado de la arquitectura que incluya todos los componentes necesarios, como servidores, bases de datos, servicios de red y almacenamiento.
- Llevar a cabo pruebas de rendimiento de la infraestructura en la nube diseñada para determinar mantenimiento y costos.
- Realizar una evaluación exhaustiva de posibles vulnerabilidades en el diseño y código de la aplicación utilizando herramientas como *SonarQube* y *AppSweep*, complementadas con pruebas manuales, siguiendo el *OWASP Top 10*, para generar un informe detallado con vulnerabilidades detectadas y recomendaciones de mitigación.
- Auditar y apoyar la implementación de prácticas de desarrollo seguro en todas las etapas del ciclo de vida del *software* integrando políticas de seguridad desde la planificación hasta el despliegue, siguiendo una metodología *SAMM*, asegurando que todas las versiones cumplan con los estándares de seguridad establecidos y reduzcan vulnerabilidades en producción.
- Auditar y apoyar la implementación de las mejores prácticas de autenticación, autorización y protección de datos mediante las implementaciones como el de autenticación multifactor (*MFA*), controles de acceso basados en roles (*RBAC*), y cifrado avanzado para datos en tránsito y en reposo, siguiendo las guías de *OWASP*, mejorando la protección de la integridad, confidencialidad y disponibilidad de los datos de los usuarios.
- Colaborar estrechamente con el equipo para mitigar riesgos y resolver problemas de seguridad durante el desarrollo mediante reuniones semanales de revisión de seguridad, fomentando una cultura de seguridad proactiva para reducir riesgos y mejorar la capacidad de respuesta del equipo.
- Auditar y apoyar la realización de análisis de seguridad exhaustivos para identificar y mitigar vulnerabilidades en la aplicación móvil de la organización. Esto incluirá la realización de diversos tipos de evaluaciones de seguridad, como análisis de vulnerabilidades y revisiones de configuración, para garantizar la protección de los datos de los usuarios, asegurando así la integridad, confidencialidad y disponibilidad de la información.
- Desarrollar un modelo de *deep learning* para la identificación y clasificación precisa de diferentes tipos de residuos, utilizando únicamente imágenes.
- Implementar un sistema de chat inteligente basado en un modelo de lenguaje de aprendizaje profundo (LLM) para facilitar la interacción informativa con los usuarios sobre la gestión de residuos.
- Evaluar la precisión y efectividad del modelo de clasificación de residuos mediante métricas de desempeño específicas para modelos de *deep learning*.
- Evaluar la efectividad del chat inteligente en términos de precisión y tiempos de respuesta al proporcionar información y consejos sobre la gestión de residuos, utilizando métricas de desempeño de *deep learning*.

CAPÍTULO 3

Justificación

La problemática ambiental generada por el manejo inadecuado de residuos y la creciente huella de carbono es un desafío que afecta no solo al medio ambiente, sino también a la calidad de vida de las personas, especialmente en entornos urbanos. En ciudades como Guatemala, la falta de sistemas eficientes de gestión de residuos y la limitada conciencia sobre el impacto ambiental de las actividades diarias dificultan la adopción de prácticas sostenibles. Ante esta situación, es imperativo desarrollar soluciones tecnológicas que no solo aborden estos desafíos, sino que también fomenten la educación y el cambio de comportamiento en la población.

Este proyecto se justifica por su enfoque en la integración de tecnologías emergentes para ofrecer una herramienta práctica que permita a los usuarios gestionar de manera responsable sus desechos, calcular su impacto ambiental y tomar decisiones informadas para reducir su huella de carbono. A través de una aplicación móvil accesible y una infraestructura en la nube eficiente y escalable, se busca democratizar el acceso a la información y a las herramientas necesarias para promover prácticas sostenibles.

Además, el desarrollo de este proyecto responde a la necesidad de encontrar soluciones aplicables y replicables en contextos similares, abordando tanto los aspectos técnicos como los desafíos sociales y culturales asociados a la gestión de residuos. La implementación de modelos de inteligencia artificial y herramientas de análisis de datos no solo facilita la funcionalidad del sistema, sino que también genera un impacto educativo significativo al promover la concienciación ambiental.

Por último, este trabajo no solo busca contribuir al cuidado del medio ambiente, sino también posicionarse como un modelo de innovación tecnológica orientada al desarrollo sostenible. Su relevancia radica en el potencial de generar un cambio positivo en la forma en que la sociedad gestiona sus residuos, fomentando una cultura de responsabilidad ambiental y sentando las bases para futuras iniciativas en este ámbito crítico.

CAPÍTULO 4

Marco teórico

4.1. Marco teórico general

4.1.1. Problemática ambiental global y local

La problemática ambiental es una cuestión crítica que afecta a nivel global y local en Guatemala, con implicaciones en la sostenibilidad del planeta y la calidad de vida de la población. A nivel global, el cambio climático, la pérdida de biodiversidad y la contaminación de los océanos son algunos de los principales desafíos. Estos problemas han aumentado por actividades como la deforestación, la quema de combustibles y la sobreexplotación de los recursos naturales, lo que ha resultado en un aumento de las emisiones de gases de efecto invernadero y en la degradación de los ecosistemas. Según la Organización de las Naciones Unidas, el calentamiento global podría aumentar 1.5°C entre 2030 y 2052 si no se toman medidas para reducir las emisiones, lo que desencadenaría eventos climáticos extremos y afectaría gravemente la biodiversidad y las economías globales [10]. En el ámbito local, las problemáticas ambientales están vinculadas a las condiciones socioeconómicas de cada región. En muchas comunidades, la contaminación, la deforestación y la mala gestión de residuos son desafíos persistentes. Estos problemas se ven agravados por la urbanización descontrolada y la falta de infraestructura adecuada. En América Latina, por ejemplo, la deforestación en la Amazonía continúa siendo una preocupación central, ya que no solo afecta al clima global, sino que también tiene impactos directos en las comunidades indígenas y en la biodiversidad local [11].

Contextualización del problema de la gestión de residuos a nivel mundial y en Guatemala

La gestión de residuos es un desafío ambiental global, intensificado por el crecimiento poblacional, la urbanización y el aumento en la generación de desechos. A nivel mundial, se estima que se generan 2.01 mil millones de toneladas de residuos sólidos urbanos al año, de los cuales solo el 19 % es reciclado o compostado, mientras que el resto termina en vertederos o es mal gestionado, lo que tiene impactos graves en el medio ambiente y la salud pública [12]. En Guatemala, la situación de la gestión de residuos refleja tanto los desafíos globales como problemáticas locales específicas. El país enfrenta un crecimiento sostenido en la generación de residuos, con una estimación de 2.1 millones de toneladas de residuos sólidos urbanos al año, de las cuales más del 50 % son orgánicos y solo una fracción

mínima es reciclada [13].

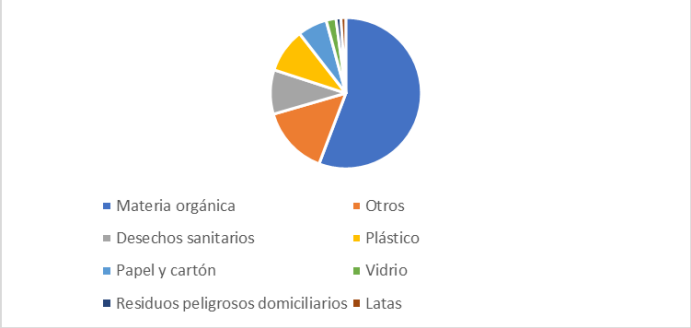


Figura 4.0. Composición general de los residuos y desechos sólidos comunes en Guatemala [1]

Impacto ambiental de la producción desmedida de basura y el incorrecto procesamiento de desechos

La producción desmedida de basura y el incorrecto procesamiento de desechos tienen efectos devastadores en el medio ambiente. La acumulación de residuos en vertederos y su incineración ineficiente liberan grandes cantidades de metano y dióxido de carbono, contribuyendo significativamente al cambio climático [14]. Además, los desechos plásticos, que representan una gran parte de los residuos mal gestionados, terminan en los océanos, donde afectan a la fauna marina, contaminan el agua y entran en la cadena alimentaria, lo que tiene repercusiones tanto ecológicas como para la salud humana [15].

En el contexto urbano, la mala gestión de los residuos sólidos puede llevar a la contaminación del suelo y del agua subterránea, lo que a su vez afecta la salud pública. Los lixiviados de vertederos mal gestionados contienen sustancias tóxicas que pueden filtrarse en el suelo y contaminar fuentes de agua, afectando no solo a los ecosistemas, sino también a las comunidades humanas que dependen de estas fuentes para consumo [16].

Efecto invernadero y sus consecuencias sobre la salud humana, la flora y la fauna

El efecto invernadero es un fenómeno natural esencial para mantener la temperatura de la Tierra en niveles habitables, pero su intensificación debido a las actividades humanas, como la quema de combustibles fósiles y la deforestación, ha provocado un aumento significativo en las concentraciones de gases de efecto invernadero, como el dióxido de carbono (CO_2) y el metano (CH_4). Este incremento está directamente relacionado con el calentamiento global, que tiene graves consecuencias para la salud humana, la flora y la fauna [17]. El cambio climático derivado del efecto invernadero intensificado ha aumentado la frecuencia e intensidad de fenómenos meteorológicos extremos, como olas de calor y tormentas, que afectan la salud humana al aumentar el riesgo de enfermedades relacionadas con el calor, problemas respiratorios y enfermedades transmitidas por vectores [18].

Además, el cambio climático influenciado por el efecto invernadero está alterando los hábitats de muchas especies, lo que lleva a la migración, el cambio en los patrones de reproducción y, en algunos casos, la extinción de especies que no pueden adaptarse a las nuevas condiciones [19]. Las plantas y los animales están experimentando cambios en sus ciclos de vida, como la floración temprana y la migración anticipada, lo que puede desestabilizar los ecosistemas y afectar la biodiversidad global.

4.1.2. Gestión de residuos sólidos

La gestión de residuos sólidos es un reto bastante complejo que abarca aspectos técnicos, económicos y sociales, y busca que los materiales generados por la población sean procesados adecuadamente tras su uso. A nivel global, la producción de residuos ha crecido considerablemente debido al aumento poblacional, la urbanización y el desarrollo económico, lo cual sobrecarga los sistemas de manejo y gestión de residuos sólidos existentes actualmente [20]. Si bien enfoques tradicionales como el uso de vertederos e incineración de los residuos producidos siguen presentes, hoy en día se opta por alternativas sostenibles como la reducción en la fuente, el reciclaje, el compostaje y la valorización energética, que reducen el impacto ambiental y promueven el aprovechamiento de recursos [21]. En países en desarrollo, lamentablemente, la problemática se ve agravada y los esfuerzos de gestión de residuos se ven limitados por la falta de fondos, infraestructura y cumplimiento de normativas, y la informalidad en el manejo de residuos lleva a problemas como vertederos no controlados y mayores riesgos para la salud pública y el medio ambiente [22].

Tipos de residuos y su clasificación

Los residuos generados por la población pueden clasificarse de varias maneras según múltiples factores y características de los mismos, dependiendo de su origen, composición y peligrosidad. Según la Organización Mundial de la Salud, los residuos se dividen comúnmente en residuos municipales, industriales, biomédicos, electrónicos y peligrosos, cada uno con características y requerimientos de manejo específicos al momento de pasar a ser residuos en lugar de materiales utilizables [23].

Los residuos municipales provienen de hogares, comercios e instituciones e incluyen tanto materiales orgánicos, como restos de alimentos, como inorgánicos, tales como plásticos y metales. Los residuos industriales, generados en procesos manufactureros, suelen contener sustancias químicas y requieren manejo especializado. En el ámbito de la salud, los residuos biomédicos producidos en hospitales y laboratorios contienen agentes patógenos con potencial riesgo sanitario. Los residuos electrónicos, cuya relevancia ha crecido con el auge tecnológico, abarcan dispositivos como móviles y electrodomésticos, conteniendo metales pesados y toxinas que exigen una disposición adecuada [24]. Finalmente, los residuos peligrosos, incluyendo materiales radiactivos y tóxicos, representan serios riesgos ambientales y para la salud, exigiendo un manejo específico [25].

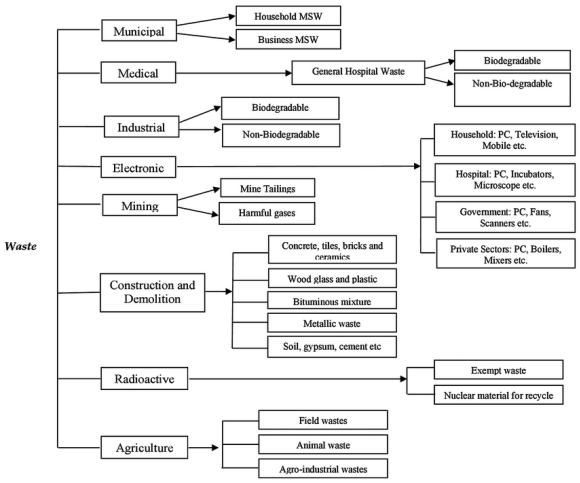


Figura 4.1. Tipos y clasificación de residuos [2]

Métodos y prácticas actuales de reciclaje y gestión de residuos

Los métodos y prácticas actuales de reciclaje y gestión de residuos han avanzado considerablemente en las últimas décadas, con un enfoque creciente en la sostenibilidad y la promoción de la economía circular. Entre las principales prácticas de reciclaje se destacan la separación en la fuente, el reciclaje mecánico y el reciclaje químico. La separación en la fuente es crucial para mejorar la eficiencia del reciclaje, ya que permite la recolección de materiales más limpios y menos contaminados, lo que optimiza la calidad del material reciclado y reduce los costos de procesamiento. El reciclaje mecánico, ampliamente utilizado para materiales como plásticos, vidrio y metales, transforma los residuos en nuevos productos, aunque enfrenta limitaciones debido a la degradación de la calidad del material tras varios ciclos de reciclaje. Por otro lado, el reciclaje químico emerge como una solución innovadora, descomponiendo los polímeros a nivel molecular para generar materiales de calidad similar al original, lo que permite superar las restricciones del reciclaje mecánico [26, 27].

En términos de gestión de residuos, se han adoptado enfoques integrados que combinan diversas estrategias para reducir el impacto ambiental. Estas incluyen la reducción en la fuente, el reciclaje, el compostaje y la valorización energética mediante la incineración controlada. Las plantas de incineración modernas, equipadas con tecnologías avanzadas de control de emisiones, permiten la generación de energía a partir de residuos con un impacto ambiental relativamente bajo. Asimismo, el compostaje ha cobrado importancia como una práctica sostenible para gestionar residuos orgánicos, transformando restos de alimentos y residuos de jardín en compost, un producto valioso para la mejora de suelos agrícolas [28]. Estas estrategias integradas buscan no solo minimizar la cantidad de residuos que llegan a vertederos, sino también aprovecharlos como recursos, contribuyendo así a una gestión más sostenible y eficiente.

Método/Práctica	**Descripción**
Separación en la fuente	Consiste en clasificar los residuos en su origen, facilitando la recolección de materiales más limpios y menos contaminados, lo que mejora la calidad del material reciclado y optimiza los procesos de reciclaje.
Reciclaje mecánico	Transformación física de materiales reciclables en nuevos productos. Es comúnmente utilizado para materiales como plásticos, vidrio y metales. Uno de sus desafíos es la degradación del material tras múltiples ciclos de reciclaje.
Reciclaje químico	Descomposición de polímeros a nivel molecular para reconstruirlos, generando materiales de calidad similar al original. Este método evita la degradación del material que ocurre con el reciclaje mecánico.
Reducción en la fuente	Estrategia de gestión de residuos que busca minimizar la generación de residuos desde el punto de origen, disminuyendo así el volumen de residuos que requiere ser gestionado posteriormente.
Compostaje	Conversión de residuos orgánicos (restos de alimentos, residuos de jardín) en compost, una enmienda valiosa para suelos agrícolas, promoviendo una gestión sostenible de los residuos orgánicos.
Valorización energética	Generación de energía mediante la incineración controlada de residuos. Las plantas de incineración modernas emplean tecnologías avanzadas de control de emisiones, lo que reduce el impacto ambiental asociado a la gestión de residuos.

Cuadro 4.0. Resumen sobre métodos y prácticas actuales de reciclaje y gestión de residuos

4.1.3. Reducción de la huella de carbono

La reducción de la huella de carbono se refiere al conjunto de acciones y estrategias implementadas para disminuir la cantidad de gases de efecto invernadero, principalmente dióxido de carbono, que una persona, empresa o actividad emite a la atmósfera. La huella de carbono incluye tanto las emisiones directas, como las provenientes del uso de combustibles fósiles en el transporte o la energía, como las emisiones indirectas, que resultan de los productos y servicios que consumimos, desde la producción de alimentos hasta la manufactura industrial [29].

Existen diversas formas de reducir la huella de carbono, que van desde el uso de energías renovables, la implementación de tecnologías más eficientes, hasta cambios en el comportamiento individual, como reducir el consumo de carne o utilizar medios de transporte más sostenibles. El concepto de huella de carbono se ha convertido en un criterio clave en políticas de sostenibilidad y en la responsabilidad social corporativa de muchas empresas [30]. A nivel individual, se promueve la adopción de hábitos como la reducción del consumo de energía, el reciclaje y la disminución del uso de plásticos de un solo uso [31]. Asimismo, la tecnología ha jugado un papel importante, con el desarrollo de aplicaciones móviles que permiten a los usuarios monitorear y controlar su impacto ambiental [32]. En el contexto global, los esfuerzos para reducir la huella de carbono están alineados con los Objetivos de Desarrollo Sostenible de las Naciones Unidas, particularmente el ODS 13, que busca tomar medidas urgentes para combatir el cambio climático y sus impactos [33].

Concepto de huella de carbono y su relevancia en la crisis climática

La huella de carbono es una medida que calcula la cantidad total de gases de efecto invernadero (GEI) emitidos directa o indirectamente por una actividad, individuo, organización o país, expresada generalmente en toneladas de CO_2 equivalente. Este concepto se ha vuelto central en la discusión sobre la crisis climática, ya que permite cuantificar el impacto de diversas actividades humanas en el calentamiento global, facilitando así la identificación de las áreas en las que es posible reducir las emisiones de GEI [34]. La relevancia de la huella de carbono radica en su capacidad para sensibilizar tanto a individuos como a organizaciones acerca de la necesidad de adoptar prácticas más sostenibles y reducir su contribución al cambio climático.

En un contexto global donde la temperatura media del planeta continúa aumentando, la huella de carbono se ha convertido en un indicador clave para la formulación de políticas ambientales y la evaluación de la sostenibilidad de productos y servicios. Además, muchas empresas y gobiernos han comenzado a utilizar este concepto para establecer objetivos de reducción de emisiones, fomentar la transparencia en sus operaciones, y desarrollar estrategias que mitiguen su impacto ambiental [35]. La huella de carbono no solo destaca la relación directa entre actividades humanas y la crisis climática, sino que también ofrece un marco práctico para medir, gestionar y reducir las emisiones de carbono.

Estrategias para la reducción de la huella de carbono

La reducción de la huella de carbono es clave en la mitigación del cambio climático, abarcando desde la adopción de energías renovables hasta mejoras en la eficiencia energética y la economía circular. La transición hacia fuentes como la energía solar y eólica es una de las medidas más efectivas, ya que reduce las emisiones sin generar dióxido de carbono y apoya la independencia energética y la creación de empleos verdes [36]. Por otro lado, la mejora de la eficiencia energética en sectores como el transporte, la industria y los edificios, mediante el uso de tecnologías como la iluminación *LED*, electrodomésticos eficientes y el fomento de vehículos eléctricos, permite una reducción significativa en el consumo de energía y las emisiones [37].

A nivel corporativo, la compensación de carbono a través de proyectos de reforestación o captura

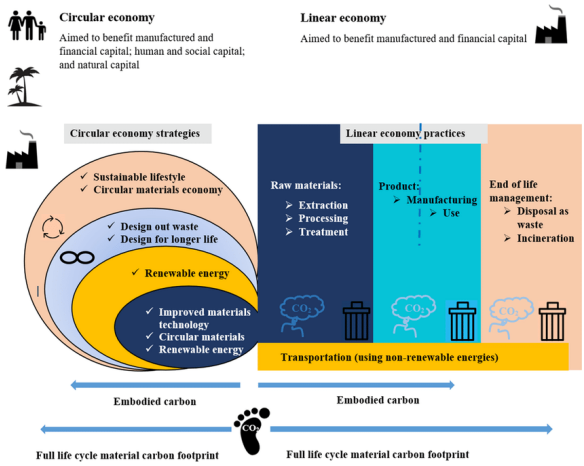


Figura 4.2. Diagrama sobre el concepto de huella de carbono y su relevancia [3]

de CO_2 permite neutralizar las emisiones inevitables y responde a las demandas de sostenibilidad de los consumidores [38]. La economía circular también juega un papel importante al minimizar la extracción de nuevos recursos mediante la reducción, reutilización y reciclaje de materiales, lo que reduce las emisiones asociadas a la producción y el descarte de productos [39]. En cuanto a las acciones individuales, cambios como reducir el consumo de carne, optar por productos locales y limitar el uso de plásticos desechables pueden tener un impacto significativo, reforzado por la educación y sensibilización para promover prácticas sostenibles [40].

Relación entre la gestión de residuos y la reducción de la huella de carbono

La relación entre la gestión de residuos y la reducción de la huella de carbono es fundamental en la estrategia global para mitigar el cambio climático. La gestión eficaz de residuos no solo previene la acumulación de desechos que podrían contribuir al calentamiento global, sino que también facilita la reducción de las emisiones de gases de efecto invernadero asociadas con la producción y disposición de residuos. En particular, la reducción, reutilización y reciclaje de materiales contribuyen a disminuir la demanda de nuevos recursos y a reducir las emisiones de CO_2 provenientes de la extracción, producción y transporte de estos materiales [41]. Por ejemplo, el reciclaje de aluminio ahorra aproximadamente un 95 % de la energía necesaria para producir aluminio a partir de bauxita, lo que resulta en una significativa reducción de las emisiones de CO_2 [42].

Además, la gestión de residuos orgánicos, a través del compostaje, ayuda a capturar carbono en el suelo y a reducir las emisiones de metano que se generan en los vertederos cuando los residuos orgánicos se descomponen anaeróbicamente. El compostaje no solo disminuye la huella de carbono de los residuos orgánicos, sino que también mejora la salud del suelo y su capacidad para almacenar carbono, contribuyendo a una reducción neta de CO_2 en la atmósfera [43]. La integración de prácticas de gestión de residuos en una estrategia de reducción de huella de carbono es, por lo tanto, esencial para alcanzar los objetivos climáticos globales y promover un futuro sostenible.

4.1.4. Tecnología y medio ambiente

La interacción entre tecnología y medio ambiente es crucial para enfrentar los desafíos de sostenibilidad actuales. Las tecnologías emergentes, como la captura y almacenamiento de carbono, están diseñadas para reducir las emisiones de CO_2 en sectores industriales, almacenando el gas en formaciones geológicas y previniendo su liberación a la atmósfera [44]. Además, el Internet de las

Estrategia	Descripción
Transición a energías renovables	Consiste en la adopción de fuentes de energía como la solar y la eólica, que no emiten gases de efecto invernadero. Esta transición contribuye a reducir la huella de carbono, además de promover la independencia energética y la creación de empleos verdes.
Mejora de la eficiencia energética	Implementación de medidas para optimizar el uso de energía en edificios, transporte e industria, como el uso de electrodomésticos eficientes, iluminación *LED* y mejoras en el aislamiento térmico, lo que reduce el consumo energético y las emisiones de carbono.
Movilidad sostenible	Fomento del uso de vehículos eléctricos y la creación de infraestructuras como redes de transporte público y carriles para bicicletas, con el objetivo de disminuir la dependencia de combustibles fósiles y reducir las emisiones del sector transporte.
Compensación de carbono	Estrategia empresarial que consiste en invertir en proyectos que absorben o evitan emisiones de CO_2, como la reforestación o la captura y almacenamiento de carbono, ayudando a neutralizar las emisiones inevitables.
Economía circular	Se basa en la reducción, reutilización y reciclaje de materiales, disminuyendo la necesidad de extraer nuevos recursos y reduciendo las emisiones asociadas con la producción y el descarte de productos.
Cambios en el estilo de vida	A nivel individual, la reducción del consumo de carne, la elección de productos locales y de temporada, y la disminución del uso de plásticos desechables son medidas que ayudan a reducir la huella de carbono.
Educación y sensibilización	La educación y la concientización son clave para empoderar a las personas a tomar decisiones informadas que contribuyan a la mitigación del cambio climático mediante acciones sostenibles.

Cuadro 4.1. Estrategias para la reducción de la huella de carbono

Cosas y la inteligencia artificial están revolucionando la gestión ambiental mediante el monitoreo en tiempo real de la calidad del aire y el agua, optimización de recursos y predicción de patrones ambientales [45].

En la agricultura, las tecnologías de precisión, como drones y sensores avanzados, contribuyen a una gestión más eficiente de los cultivos y el agua, promoviendo prácticas agrícolas sostenibles [46]. Sin embargo, el avance tecnológico también plantea retos, como el aumento de residuos electrónicos, lo que hace necesario desarrollar métodos de reciclaje para minimizar el impacto ambiental de la tecnología [47].

El papel de las tecnologías de la información en la promoción de prácticas sostenibles

Las tecnologías de la información juegan un papel crucial en la promoción de prácticas sostenibles al permitir una gestión más eficiente de recursos y facilitando la toma de decisiones informadas. La implementación de tecnologías de la información puede mejorar la sostenibilidad en diversas áreas, desde la gestión de energía hasta la optimización de cadenas de suministro. Por ejemplo, los sistemas de gestión de energía basados en TI permiten monitorear y controlar el consumo de energía en tiempo real, lo que ayuda a reducir el desperdicio y a mejorar la eficiencia energética [48].

Además, las tecnologías de la información facilitan la adopción de prácticas de economía circular, permitiendo una mejor gestión del ciclo de vida de los productos. Las plataformas digitales y las herramientas de análisis de datos permiten a las empresas rastrear el origen y el destino de los materiales, optimizar el reciclaje y la reutilización, y reducir el desperdicio [49]. Por ejemplo, los sistemas de trazabilidad basados en *blockchain* pueden proporcionar transparencia en las cadenas de suministro, garantizando que los materiales reciclados sean realmente reciclados y no terminen en vertederos [50].

Uso de aplicaciones móviles para la educación y concientización ambiental

Las aplicaciones móviles han emergido como herramientas poderosas para la educación y concientización ambiental, permitiendo a los usuarios aprender sobre sostenibilidad y prácticas ecológicas de manera interactiva y accesible. Estas aplicaciones proporcionan información en tiempo real, permiten la participación en iniciativas comunitarias y fomentan cambios de comportamiento a través de la gamificación y el seguimiento de actividades. Además, las aplicaciones móviles pueden facilitar la educación ambiental en contextos formales e informales, integrando juegos y desafíos que enseñan a los usuarios sobre reciclaje, conservación de recursos y biodiversidad [51].

4.1.5. Aspectos legales y ambientales en Guatemala

Legislación sobre manejo de residuos y reciclaje en Guatemala

En Guatemala, el marco legal para la gestión de residuos ha sido fundamental para abordar los desafíos ambientales y promover la sostenibilidad. Las principales normativas incluyen leyes que regulan desde la generación hasta la disposición final de los residuos, con un énfasis creciente en el reciclaje y la reducción de desechos.

Ley de protección y mejoramiento del medio ambiente (decreto 68-86)

Esta ley, promulgada en 1986, es un pilar de la legislación ambiental en Guatemala. Su objetivo es proteger y mejorar el medio ambiente a través de la regulación del uso de los recursos naturales y la prevención de la contaminación. El decreto 68-86 establece la obligación de gestionar adecuadamente los residuos sólidos para minimizar su impacto negativo en la salud pública y el entorno [52].

Reglamento para la gestión integral de los residuos y desechos sólidos comunes (acuerdo gubernativo 164-2021)

Aprobado en 2021, este reglamento es una herramienta clave para la gestión de residuos sólidos en Guatemala. Establece directrices claras para la separación, recolección, transporte y disposición final de los residuos, promoviendo prácticas de reciclaje y la reducción de la cantidad de desechos que terminan en vertederos. El reglamento también contempla sanciones para quienes no cumplan con las normativas, subrayando la importancia de un cumplimiento estricto para proteger el medio ambiente [53, 54].

Política nacional de residuos sólidos (MARN, 2010)

El ministerio de ambiente y recursos naturales (MARN) desarrolló esta política como una respuesta a la creciente generación de residuos y su impacto ambiental. La política se centra en la

reducción de residuos, la promoción del reciclaje y la valorización de los desechos. También fomenta la participación activa de la ciudadanía y la implementación de programas educativos para concienciar sobre la importancia de la gestión adecuada de residuos [55].

Impacto ambiental del manejo de residuos en Guatemala

El manejo inadecuado de residuos representa uno de los mayores desafíos ambientales en Guatemala. Los vertederos, muchos de los cuales están llegando a su capacidad máxima, son una fuente significativa de contaminación del suelo y del agua. Además, la descomposición de residuos orgánicos en estos sitios genera emisiones de metano, un potente gas de efecto invernadero que contribuye al cambio climático [56].

En las áreas rurales, donde la infraestructura para la gestión de residuos es limitada, la quema de basura es una práctica común. Esta forma de eliminación de residuos no solo contamina el aire, sino que también representa un riesgo para la salud humana, provocando enfermedades respiratorias y otras afecciones. Un informe de la FAO subraya la necesidad de mejorar la infraestructura y las prácticas de manejo de residuos en áreas rurales para mitigar estos impactos negativos [57].

Estrategias internacionales y su aplicación en Guatemala

Guatemala ha tomado medidas para alinearse con las mejores prácticas internacionales en la gestión de residuos, inspirándose en enfoques como la economía circular. La Fundación Ellen MacArthur, por ejemplo, promueve un modelo en el que los materiales se mantienen en uso el mayor tiempo posible, reduciendo así la necesidad de nuevos recursos y minimizando los desechos [58]. A pesar de estos avances conceptuales, la implementación efectiva de estas estrategias en Guatemala enfrenta obstáculos significativos, incluidos limitados recursos financieros y una infraestructura inadecuada.

La Estrategia Europea para Plásticos en una Economía Circular, lanzada en 2018, es otro modelo relevante que Guatemala podría emular. Esta estrategia tiene como objetivo reducir los plásticos de un solo uso y fomentar el reciclaje a gran escala, lo cual es crucial en un contexto donde los plásticos representan una gran parte de los residuos no biodegradables [59].

Crisis de desechos en Guatemala

El informe titulado **Análisis de la generación de desechos sólidos en Guatemala**, publicado por el Instituto de Problemas Nacionales de la Universidad de San Carlos de Guatemala (IPNUSAC) en 2023, proporciona un análisis exhaustivo sobre la creciente crisis de desechos en el país. Este estudio revela que la generación de desechos sólidos ha aumentado considerablemente en las últimas décadas, superando la capacidad de gestión de los municipios guatemaltecos [60].

El estudio destaca varias deficiencias clave en la infraestructura para el manejo de residuos, particularmente en áreas rurales y urbanas marginales. La falta de rellenos sanitarios adecuados ha llevado a un uso generalizado de vertederos a cielo abierto, que no solo contribuyen a la contaminación ambiental, sino que también representan un riesgo significativo para la salud pública. Además, la contaminación de cuerpos de agua debido a la disposición inadecuada de residuos es un problema crítico identificado en el informe [60].

Este informe subraya la necesidad urgente de implementar políticas más efectivas y de invertir en infraestructura adecuada para mitigar los impactos ambientales y sociales asociados con la mala gestión de residuos en Guatemala.

Desafíos socioeconómicos en la gestión de residuos

En Guatemala, la gestión de residuos está marcada por una fuerte disparidad entre las áreas urbanas y rurales. Mientras que las ciudades más grandes como la Ciudad de Guatemala cuentan con servicios de recolección de basura, las comunidades rurales y las áreas urbanas marginales a menudo carecen de dichos servicios, lo que agrava los problemas de salud pública y contaminación ambiental. Además, un gran porcentaje del reciclaje es llevado a cabo por trabajadores informales, conocidos como "guajeros", quienes operan en vertederos a cielo abierto, enfrentando condiciones laborales peligrosas y sin acceso a protección social [54].

Educación y sensibilización ambiental

La concientización pública es clave para mejorar la gestión de residuos en Guatemala. Aunque existen iniciativas locales de educación ambiental, los esfuerzos a gran escala han sido insuficientes para cambiar los hábitos de la población. Es necesario incorporar programas de educación ambiental desde una edad temprana en las escuelas, así como campañas de sensibilización pública dirigidas a adultos [57].

Impacto del cambio climático y la gestión de residuos

El manejo de residuos está estrechamente vinculado al cambio climático, especialmente debido a las emisiones de metano provenientes de los vertederos. Mejorar la gestión de residuos y promover el reciclaje no solo reduce la contaminación local, sino que también contribuye a mitigar el cambio climático global [56].

4.2. Marco teórico sobre experiencia de usuario

4.2.1. Diseño centrado en el usuario

El Diseño Centrado en el Usuario (DCU) es una filosofía de diseño que pone al usuario en el centro de todo el proceso de desarrollo de productos y servicios. Esta aproximación asegura que las necesidades, deseos y limitaciones de los usuarios finales se consideren en todas las etapas del proceso de diseño, desde la investigación inicial hasta el desarrollo y las pruebas de usabilidad. El objetivo principal del DCU es crear productos que sean no solo funcionales, sino también fáciles de usar y accesibles, mejorando la experiencia general del usuario [61].

En el contexto moderno, el DCU se ha vuelto esencial en el desarrollo de *software* y aplicaciones móviles. Las metodologías ágiles y las pruebas iterativas son componentes clave de este enfoque, permitiendo la incorporación constante de retroalimentación del usuario en el ciclo de desarrollo. Esto no solo mejora la satisfacción del usuario, sino que también puede reducir costos a largo plazo al minimizar la necesidad de rediseños o revisiones extensas [62].

La importancia del DCU también se refleja en un diseño que busca que los productos sean accesibles para la mayor cantidad de personas posible. Esto incluye consideraciones como el diseño para personas con discapacidades visuales, auditivas o motoras, lo que amplía el alcance de los productos tecnológicos en una sociedad cada vez más diversa [63]. Además, el DCU se ha adaptado a las tecnologías emergentes como la inteligencia artificial y la realidad aumentada, donde el desafío no solo es crear interfaces usables, sino también asegurar que la interacción con estos sistemas sea intuitiva y confiable para una amplia gama de usuarios [64].

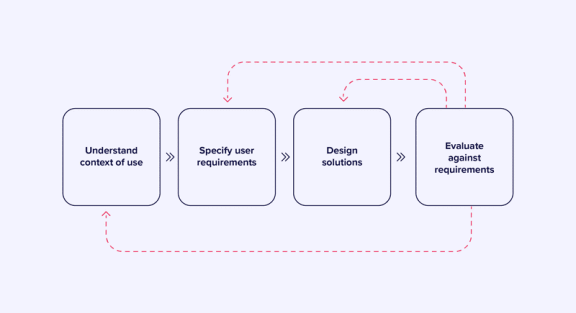


Figura 4.3. Diagrama de las fases del diseño centrado en el usuario

Conceptos y principios del diseño centrado en el usuario

El DCU se basa en principios clave que guían la creación de productos orientados a satisfacer las necesidades de los usuarios. Entre estos, destaca la comprensión profunda de los usuarios a través de investigaciones sobre su comportamiento y contexto, lo que permite desarrollar soluciones alineadas con sus expectativas [65].

La creación de prototipos facilita la prueba de ideas y la identificación temprana de problemas, mientras que la iteración continua asegura mejoras constantes en la funcionalidad y usabilidad del producto [66].

La accesibilidad es también un principio esencial del DCU, promoviendo el diseño de interfaces compatibles con tecnologías de asistencia y siguiendo directrices como las WCAG para hacer productos accesibles a personas con diversas habilidades [67]. La empatía se destaca como un componente fundamental, ya que permite a los diseñadores comprender las experiencias de los usuarios y desarrollar productos inclusivos y satisfactorios [68].

Concepto/Principio	**Descripción**
Comprensión de los usuarios	Involucra realizar investigaciones exhaustivas sobre los comportamientos, preferencias y el contexto de uso de los usuarios, permitiendo diseñar soluciones que se ajusten a sus necesidades reales.
Creación de prototipos	Permite visualizar y probar ideas planteadas para un proyecto rápidamente antes de la implementación final. Ayuda a identificar problemas potenciales en una etapa temprana, reduciendo riesgos y costos.
Iteración constante	Se refiere a la repetición de ciclos de diseño, pruebas y refinamiento para asegurar que el producto final sea funcional, intuitivo y agradable de usar.
Accesibilidad	Promueve el diseño de productos que puedan ser utilizados por el mayor número de personas, independientemente de sus habilidades o discapacidades, siguiendo directrices como las WCAG.
Empatía	Los diseñadores deben ponerse en el lugar de los usuarios para comprender sus experiencias y problemas, lo que contribuye a la creación de productos más inclusivos y satisfactorios.

Cuadro 4.2. Resumen de conceptos y principios del diseño centrado en el usuario

Importancia de la experiencia de usuario (UX) en el éxito de aplicaciones móviles

La experiencia de usuario (UX) es crucial para el éxito de las aplicaciones móviles, ya que impacta en la satisfacción y retención de los usuarios en un mercado competitivo. Una UX bien diseñada mejora la facilidad de uso y proporciona un valor añadido, lo que aumenta la lealtad del usuario y reduce las tasas de abandono [69]. Aspectos esenciales de la UX incluyen una navegación intuitiva y tiempos de carga rápidos, siendo determinante el rendimiento para evitar que los usuarios abandonen la aplicación. *Google* destaca que más del 50 % de los usuarios abandonan aplicaciones que tardan más de tres segundos en cargar [70].

Además, la personalización es clave, ya que adapta la experiencia a las preferencias individuales del usuario, lo cual mejora la satisfacción y fomenta el uso continuo [71]. El diseño visual y la interacción emocional también son importantes, ya que las aplicaciones visualmente atractivas y emocionalmente positivas tienden a tener mayor éxito, al ofrecer tanto funcionalidad como una experiencia agradable para el usuario [72].

Metodologías de diseño de experiencia de usuario aplicadas en proyectos tecnológicos

Las metodologías de diseño de UX son fundamentales en la creación de productos tecnológicos que combinan funcionalidad con una experiencia positiva. El DCU es una de las principales metodologías, enfocándose en comprender las necesidades del usuario mediante investigación, prototipos y pruebas iterativas para alinear el producto con sus expectativas [73]. Otra metodología clave es el *design thinking*, que aborda la resolución creativa de problemas en cinco etapas (empatizar, definir, idear, prototipar y testear) y fomenta la colaboración entre equipos para soluciones innovadoras y deseables [74].

Lean UX, por su parte, es especialmente relevante en entornos ágiles y se centra en la creación rápida de prototipos y en la validación constante de ideas con usuarios, permitiendo una adaptación rápida a los cambios. Es ideal para *startups* o proyectos con recursos limitados, donde la velocidad y la eficiencia son esenciales [66].

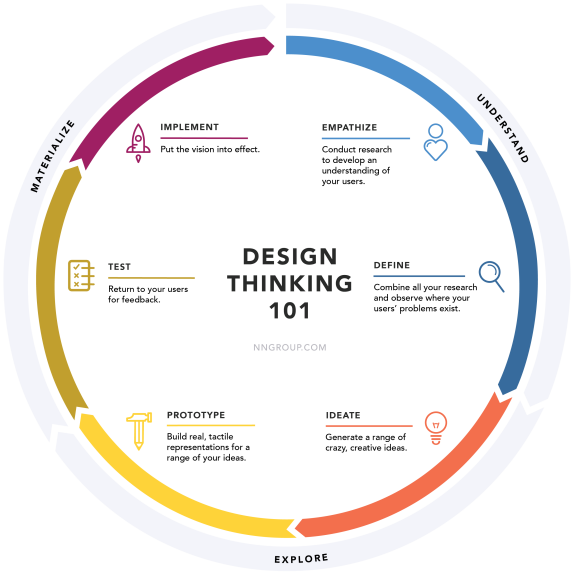


Figura 4.4. Diagrama sobre la metodología y flujo de trabajo de *design thinking* [4]

4.2.2. Experiencia de usuario

La experiencia de usuario (UX) es fundamental en el diseño de productos interactivos, especialmente aplicaciones móviles, y abarca la percepción y respuesta del usuario al interactuar con el producto. Definida por la ISO como la percepción resultante del uso o la anticipación del uso de un sistema, la UX integra aspectos emocionales, psicológicos y funcionales para ofrecer una interacción positiva y fluida [75]. En aplicaciones móviles, la UX va más allá de la interfaz y considera eficiencia, accesibilidad, estética y satisfacción, buscando que el usuario experimente una interacción intuitiva y sin esfuerzo [76].

El DCU es clave para garantizar una buena UX, ya que adapta el producto a las necesidades y expectativas del usuario a través de iteraciones basadas en retroalimentación constante [77]. La accesibilidad es también crucial en UX, asegurando que las aplicaciones sean utilizables por personas con distintas capacidades, lo que amplía el público y el impacto social de la aplicación [78]. La satisfacción final del usuario, influenciada tanto por la funcionalidad como por la estética, es el objetivo central de la UX, contribuyendo a la retención y al éxito del producto en el mercado [79].

Usabilidad en aplicaciones móviles

La usabilidad es esencial en el diseño de aplicaciones móviles, ya que determina la facilidad con que los usuarios pueden interactuar efectivamente con un sistema. Incluye la simplicidad de la interfaz, la claridad en la navegación y la velocidad de respuesta, factores que minimizan la curva de aprendizaje, reducen el tiempo para completar tareas y disminuyen errores, incrementando así la satisfacción del usuario [80]. Dado que las pantallas pequeñas y la interacción táctil plantean desafíos específicos, es fundamental asegurar que las funcionalidades clave sean accesibles y comprensibles [81]. Las pruebas de usabilidad durante el desarrollo permiten identificar y corregir problemas en escenarios reales, evaluando indicadores de eficiencia y satisfacción para mejorar continuamente la experiencia de usuario [82].

Diseño de interfaces móviles

El diseño de interfaces móviles se enfoca en la creación de elementos visuales y táctiles que permiten a los usuarios interactuar con las aplicaciones de manera intuitiva y eficiente. Dado el limitado espacio en las pantallas móviles, el diseño debe priorizar la simplicidad, claridad y consistencia en todos los elementos gráficos [83]. Aspectos como la disposición de los botones, la tipografía y el color juegan un papel crucial en la usabilidad, ya que influyen en la percepción del usuario sobre la facilidad de uso. Un buen diseño se caracteriza por minimizar la complejidad visual [63].

La consistencia en el diseño de interfaces gráficas también es esencial para ofrecer una UX satisfactoria, ya que facilita la comprensión y memorización de los patrones de interacción [62]. Mantener un diseño coherente entre las diferentes pantallas de la aplicación y entre aplicaciones del mismo ecosistema (como las plataformas *iOS* y *Android*) refuerza la familiaridad y reduce la curva de aprendizaje. Asimismo, es fundamental garantizar que la interfaz sea adaptativa y responsive, ajustándose a distintas resoluciones y tamaños de pantalla, lo que permite que los usuarios tengan una experiencia óptima en cualquier dispositivo. El uso de guías de estilo y bibliotecas de componentes visuales específicos para móviles, como *Material Design* de *Google* o las *Human Interface Guidelines* de aplicaciónle, facilita la creación de interfaces que respetan tanto las limitaciones físicas como las expectativas de los usuarios [84].

Feedback del usuario

El *feedback* del usuario es fundamental en el diseño y mejora de aplicaciones móviles, ya que proporciona información directa sobre la percepción y experiencia de quienes utilizan la aplicación. Este *feedback*, obtenido a través de encuestas, reseñas, análisis de comportamiento y pruebas de usabilidad, permite identificar problemas en la interfaz, barreras de usabilidad y áreas que requieren optimización [62]. Incorporar la retroalimentación en el ciclo de desarrollo ajusta la aplicación a las necesidades reales de los usuarios y permite corregir errores de forma continua. Además, en el entorno móvil, donde la tecnología y las expectativas cambian rápidamente, el uso de herramientas de análisis ayuda a rastrear el comportamiento del usuario, detectar patrones de uso y zonas de abandono. Este enfoque no solo mejora la experiencia del usuario, sino que también fomenta su lealtad, demostrando que sus opiniones son valoradas y los problemas resueltos oportunamente [78].

Evaluación de la experiencia del usuario (UX)

La evaluación de la experiencia de usuario (UX) es esencial para entender y mejorar cómo los usuarios perciben e interactúan con una aplicación, recopilando datos cualitativos y cuantitativos sobre aspectos como facilidad de uso, satisfacción, eficiencia y efectividad de la interfaz [85]. A través de técnicas como pruebas de usabilidad, encuestas, entrevistas y análisis de comportamiento en tiempo real, los diseñadores identifican áreas problemáticas y optimizan la aplicación en función de métricas de rendimiento. Este proceso asegura que el diseño cumpla tanto con los requisitos técnicos como con las expectativas del usuario final.

La evaluación de UX no es solo una etapa final, sino que debe integrarse iterativamente a lo largo del ciclo de vida del producto. Métodos como el análisis heurístico, en el que expertos evalúan la interfaz según principios de usabilidad, y la evaluación contextual, que observa a los usuarios en su entorno natural, ofrecen *insights* profundos sobre la experiencia real [80]. Este enfoque no solo mejora la funcionalidad, sino que incrementa el valor emocional y la satisfacción general, fomentando la lealtad y el compromiso del usuario con la aplicación [86].

4.2.3. Técnicas de recopilación de información y muestreo

En el desarrollo de cualquier investigación científica, la adecuada selección de las técnicas de recopilación de información y muestreo es crucial para asegurar la validez y confiabilidad de los datos obtenidos. En el contexto de este proyecto, enfocado en la creación de una aplicación móvil para la gestión de residuos y la reducción de la huella de carbono, es fundamental utilizar metodologías que permitan entender a fondo las prácticas de los usuarios, sus necesidades y percepciones respecto a la sostenibilidad.

Técnicas de recopilación de información

Las técnicas de recopilación de información permiten obtener datos cualitativos y cuantitativos. Una de las más empleadas es la entrevista, que puede ser estructurada, semiestructurada o no estructurada. Este método proporciona información detallada sobre las opiniones y experiencias de los usuarios con respecto al manejo de residuos. Las entrevistas semiestructuradas son especialmente útiles en esta investigación, ya que permiten explorar con flexibilidad temas relacionados con los hábitos de reciclaje y la conciencia ambiental de los usuarios, adaptándose a las respuestas dadas por los participantes [87, 88].

Otra técnica relevante es la encuesta o cuestionario, que es apropiada para obtener datos cuantitativos de una muestra representativa. Las encuestas permiten medir de manera precisa las actitudes

y comportamientos de los usuarios en relación con la reducción de la huella de carbono, ofreciendo una visión más amplia y generalizable [89]. Además, los grupos focales resultan útiles para la exploración cualitativa de las percepciones de los usuarios. A través de la interacción grupal, se pueden identificar dinámicas y opiniones comunes o divergentes entre los participantes [90].

En cuanto a la observación directa, esta técnica permite analizar el comportamiento de los usuarios en su entorno natural sin intervención directa, obteniendo datos sobre sus interacciones cotidianas con el reciclaje y el manejo de residuos. Esto es esencial para identificar posibles mejoras en la usabilidad de la aplicación móvil [91]. Finalmente, la revisión de literatura y datos secundarios proporciona un marco de referencia para comparar los hallazgos con investigaciones previas sobre gestión de residuos y sostenibilidad, además de fortalecer la fundamentación teórica del proyecto [92].

Técnica de recopilación	**Descripción**
Entrevistas	Proporcionan información detallada sobre las opiniones y experiencias de los usuarios en torno al manejo de residuos. Las entrevistas semiestructuradas permiten flexibilidad en la exploración de temas relacionados con hábitos de reciclaje y conciencia ambiental.
Encuestas	Permiten obtener datos cuantitativos de una muestra representativa para medir actitudes y comportamientos sobre la reducción de la huella de carbono.
Grupos focales	Útiles para explorar cualitativamente las percepciones de los usuarios mediante la interacción grupal, identificando dinámicas y opiniones comunes o divergentes.
Observación directa	Analiza el comportamiento de los usuarios en su entorno natural sin intervención directa, proporcionando datos sobre interacciones cotidianas con el reciclaje.
Revisión de literatura y datos secundarios	Proporciona un marco de referencia para comparar hallazgos con investigaciones previas sobre gestión de residuos y sostenibilidad.

Cuadro 4.3. Técnicas de recopilación de información y muestreo

Técnicas de muestreo

Dentro de las técnicas de muestreo más utilizadas, se encuentran las de muestreo probabilístico. Por ejemplo, el muestreo aleatorio simple otorga a cada individuo de la población una probabilidad igual de ser seleccionado, lo que contribuye a la imparcialidad en la recolección de datos [93]. Asimismo, el muestreo estratificado permite dividir a la población en subgrupos o estratos homogéneos, como diferentes rangos de edad o niveles de conciencia ambiental, asegurando que cada uno de estos grupos esté representado adecuadamente en la muestra [94].

Por otro lado, el muestreo no probabilístico puede ser útil cuando se requiere acceso a grupos específicos de usuarios. El muestreo intencional o de conveniencia se basa en seleccionar a los individuos que cumplan con ciertos criterios, como aquellos que ya practican reciclaje de manera activa [95]. El muestreo por bola de nieve es adecuado para estudiar comunidades o redes en las que los participantes se recomiendan mutuamente, facilitando el acceso a personas con interés en la sostenibilidad y gestión ambiental [96].

Validez y confiabilidad

La validez y confiabilidad de los datos son criterios fundamentales en la investigación. La validez indica la precisión con la que los instrumentos de medición capturan el fenómeno de interés, como

las percepciones sobre reciclaje y huella de carbono [97]. Para evaluar la validez, se puede utilizar la correlación de *Pearson*, que establece una relación cuantitativa entre los resultados de un instrumento y un criterio de referencia. La ecuación para calcular este coeficiente es:

$$r = \frac{n(\sum xy) - (\sum x)(\sum y)}{\sqrt{[n\sum x^2 - (\sum x)^2][n\sum y^2 - (\sum y)^2]}}$$

donde n representa el número de pares de datos, x es la variable medida, y y es el criterio de referencia [98].

Por otro lado, la confiabilidad implica que los resultados sean consistentes a lo largo del tiempo y replicables en contextos similares. Un método común para medir la confiabilidad es el coeficiente alfa de *Cronbach*, que evalúa la consistencia interna de un conjunto de *ítems* en un *test*. La ecuación para calcular α es:

$$\alpha = \frac{k}{k-1}\left(1 - \frac{\sum \sigma_i^2}{\sigma_t^2}\right)$$

En esta ecuación, k es el número de ítems en el *test*, σ_i^2 es la varianza de cada ítem, y σ_t^2 es la varianza total del *test* [99].

Muestreo probabilístico

El muestreo probabilístico se caracteriza por otorgar a todos los miembros de la población una probabilidad conocida y no nula de ser seleccionados para formar parte de la muestra. Esto garantiza que los resultados obtenidos sean representativos y que puedan generalizarse a toda la población objetivo [93]. Dentro de este tipo de muestreo, el muestreo aleatorio simple es una de las técnicas más utilizadas.

La probabilidad de que un individuo sea seleccionado en un muestreo aleatorio simple se puede expresar mediante la siguiente ecuación:

$$P(i) = \frac{1}{N}$$

donde $P(i)$ es la probabilidad de selección del individuo i y N es el tamaño total de la población.

Otra técnica relevante dentro del muestreo probabilístico es el muestreo estratificado, que se utiliza cuando la población se puede dividir en subgrupos homogéneos o estratos que comparten características específicas, como la edad, el género o el nivel de conciencia ambiental. Este tipo de muestreo es particularmente útil para asegurar que cada subgrupo esté representado adecuadamente en la muestra.

La ecuación para determinar el tamaño de la muestra en muestreo estratificado se expresa como:

$$n = \sum_{h=1}^{L} \frac{N_h}{N} n$$

donde n es el tamaño total de la muestra, N_h es el tamaño de cada estrato, N es el tamaño total de la población y L es el número de estratos.

En este caso, se seleccionan muestras de cada estrato de manera proporcional o igualitaria, dependiendo de la investigación [94].

Muestreo no probabilístico

El muestreo no probabilístico se emplea cuando no es posible garantizar que todos los individuos de la población tengan una probabilidad conocida de ser seleccionados. A pesar de no ser representativo en el sentido estadístico, este tipo de muestreo es muy útil en situaciones en las que se requiere un enfoque en grupos específicos, y sobre todo en ocasiones en las que características que puedan resultar importantes en ciertas poblaciones a muestrear no sean tan claras como en un escenario ideal [88].

Entre los métodos de muestreo no probabilístico, se destacan:

- **Muestreo por conveniencia**: selección de participantes disponibles o dispuestos a participar, facilitando la recolección rápida de datos.
- **Muestreo intencional o dirigido**: búsqueda deliberada de individuos con características específicas. Este enfoque permite obtener información detallada sobre las necesidades y expectativas de los usuarios más activos en temas medioambientales [94].
- **Muestreo por bola de nieve**: técnica útil para acceder a poblaciones difíciles de alcanzar. Los participantes iniciales refieren a otros individuos con características similares, facilitando la creación de una red de contactos que comparten intereses comunes, como la preocupación por el medio ambiente y el reciclaje [100].

El muestreo intencional puede expresarse en términos de una proporción esperada de individuos con ciertas características p, lo que puede ayudar a estimar el número necesario de participantes con esas características. La ecuación para calcular el tamaño de la muestra necesaria en este contexto se puede expresar como:

$$n = \frac{Z^2 \cdot p(1-p)}{E^2}$$

donde n es el tamaño de la muestra, Z es el valor crítico de Z para el nivel de confianza deseado (por ejemplo, 1.96 para un 95 % de confianza), p es la proporción esperada de la característica en la población y E es el margen de error tolerado.

La dinámica del muestreo por bola de nieve puede representarse matemáticamente en un modelo de red, donde cada participante P_i tiene la capacidad de referir a k nuevos participantes:

$$P_{n+1} = P_n + k \cdot P_n$$

donde P_n es el número de participantes en la etapa n y k es el número promedio de nuevos participantes referidos por cada participante.

Este enfoque puede facilitar el crecimiento exponencial de la muestra y, al mismo tiempo, mejorar la calidad de la información recopilada sobre las prácticas de reciclaje y sostenibilidad dentro de la comunidad.

Tamaño de la muestra

El tamaño de la muestra es crucial para garantizar la precisión y validez en investigaciones con poblaciones diversas, como los usuarios de una aplicación de gestión de residuos. Un tamaño de muestra pequeño podría no captar adecuadamente las distintas prácticas y niveles de conciencia ambiental, mientras que uno excesivamente grande podría ser costoso sin aportar precisión significativa [93]. Además, reducir el margen de error es esencial para reflejar fielmente las necesidades de la población, considerando la inclusión de subgrupos diversos como jóvenes y personas con o sin experiencia en reciclaje [89].

Validez y confiabilidad

La validez y confiabilidad son esenciales para asegurar la calidad de los datos en cualquier investigación. La validez indica si un instrumento mide con precisión el fenómeno de interés; para esto, es necesario que los instrumentos estén alineados con los objetivos del estudio. En el caso de una aplicación para la gestión de residuos, es clave que las encuestas reflejen comportamientos y actitudes hacia el reciclaje y la sostenibilidad [88]. Existen varios tipos de validez: la de contenido, que requiere que los ítems del instrumento reflejen adecuadamente el constructo y se validen con expertos [91]; y la de criterio, que evalúa si el instrumento puede predecir o correlacionar con otros criterios, como datos municipales de reciclaje [97].

La confiabilidad se refiere a la consistencia de los resultados. Un instrumento confiable produce resultados similares en condiciones similares, lo cual puede verificarse mediante pruebas piloto para ajustar posibles inconsistencias [88]. Además, el coeficiente alfa de *Cronbach* evalúa la consistencia interna del instrumento, asegurando que los ítems midan el mismo constructo [94]. La técnica de prueba y re-prueba también puede usarse, aplicando el instrumento en dos momentos distintos para evaluar la estabilidad de los resultados [95].

4.2.4. Prototipado y herramientas

El prototipado es una etapa esencial en el desarrollo de productos digitales, especialmente en aplicaciones móviles, ya que permite crear representaciones preliminares para visualizar, probar y validar ideas antes de su implementación [101]. Este proceso ayuda a los equipos de diseño y desarrollo a simular el funcionamiento del producto, identificar problemas y obtener retroalimentación temprana de los usuarios y las partes interesadas. Los prototipos se clasifican comúnmente en baja y alta fidelidad: los de baja fidelidad, como *wireframes* y bocetos, son rápidos de elaborar y permiten explorar múltiples ideas de diseño sin profundizar en detalles [102].

Por su parte, los prototipos de alta fidelidad son más detallados y se acercan a la versión final del producto. Estos prototipos se utilizan en etapas avanzadas para evaluar la usabilidad del producto en condiciones más realistas [103]. Además, el prototipado es fundamental para validar suposiciones sobre el comportamiento de los usuarios, lo que reduce el riesgo de invertir recursos en soluciones que podrían no cumplir con las expectativas del usuario [104]. La validación mediante prototipos facilita la identificación de problemas de usabilidad difíciles de prever en la fase conceptual, permitiendo ajustes tempranos [82].

Asimismo, el prototipado mejora la UX al proporcionar una representación que permite realizar pruebas con usuarios reales, identificando barreras y oportunidades de mejora en la estructura, interacción e interfaz gráfica del producto [105]. Finalmente, el prototipado fomenta la iteración continua, permitiendo que el diseño evolucione a medida que se recaba información sobre las necesidades del usuario y las limitaciones técnicas del proyecto [106].

Tipos de prototipado

El proceso de prototipado se puede dividir en varias categorías según el nivel de detalle y funcionalidad que se requiera para validar las ideas de diseño. Entre los tipos más comunes están el prototipado de baja fidelidad, el prototipado de alta fidelidad y el prototipado interactivo. Cada uno de estos enfoques tiene aplicaciones específicas y aporta ventajas en diferentes etapas del ciclo de desarrollo de un producto digital.

Prototipado de baja fidelidad

El prototipado de baja fidelidad se caracteriza por su simplicidad y rapidez, utilizando bocetos, *wireframes* y diagramas de flujo para definir la estructura básica del diseño sin enfocarse en detalles visuales o funcionales. Con herramientas sencillas como lápiz y papel o *software* asequible, los diseñadores pueden explorar diferentes ideas y flujos de usuario sin gran inversión de tiempo o recursos, facilitando la discusión del flujo de interacción dentro del equipo [102]. Herramientas digitales como *Balsamiq* y *Sketch* permiten crear y ajustar rápidamente estos prototipos, siendo especialmente útiles en las primeras fases del desarrollo para validar conceptos y estructuras [80].

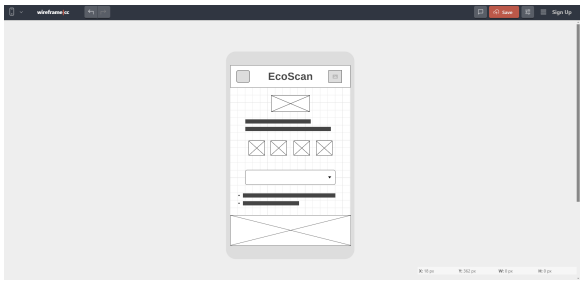


Figura 4.5. Ejemplo de prototipado de baja fidelidad en formato de *wireframe*

Prototipado de alta fidelidad

A medida que avanza el desarrollo, los prototipos de alta fidelidad se vuelven esenciales, ya que representan casi versiones finales del producto en diseño visual y experiencia de usuario. A diferencia de los prototipos de baja fidelidad, estos incluyen tipografía, colores, imágenes y elementos interactivos que permiten a los usuarios navegar y experimentar el producto de manera cercana a la versión terminada [103]. Herramientas como *Figma*, *Adobe XD* y *Sketch* son clave para crear estos prototipos, ya que facilitan el diseño detallado y la inclusión de transiciones y animaciones interactivas [106].

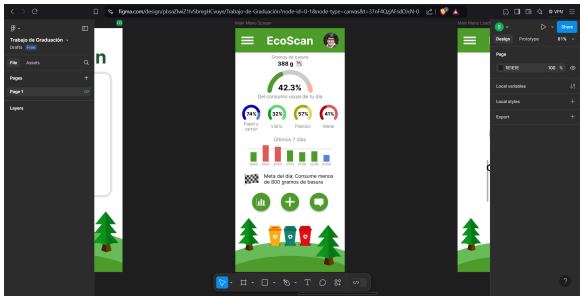


Figura 4.6. Ejemplo de prototipado de alta fidelidad desarrollado en *Figma*

Prototipado interactivo

El prototipado interactivo permite simular tanto la apariencia como las interacciones del usuario con el producto, probando flujos de trabajo, transiciones, animaciones y respuestas a las acciones del usuario. Este enfoque es útil para evaluar la intuitividad de la interfaz y realizar ajustes que optimicen la experiencia [82]. Herramientas como *InVision*, *Axure* y *Figma* son populares para este tipo de prototipado, ya que permiten crear flujos de trabajo complejos y vincular pantallas, facilitando la navegación como si fuera el producto final. El prototipado interactivo es clave para identificar problemas de usabilidad y garantizar una navegación fluida en la aplicación [107].

Proceso de iteración con herramientas

La iteración en el diseño es crucial para el éxito de un producto, ya que permite ajustes continuos basados en el *feedback* de usuarios y *stakeholders*. Este proceso facilita la creación rápida de prototipos que pueden modificarse de manera ágil y compartirse con los equipos para recibir retroalimentación inmediata, agilizando el proceso de revisión sin la necesidad de múltiples reuniones o correos electrónicos [108]. Además, los prototipos interactivos pueden ser distribuidos para que los *stakeholders* y *testers* añadan comentarios directamente, lo que hace que el ciclo de retroalimentación sea eficiente y rápido [107].

Un aspecto clave de la iteración es la capacidad de ajustar versiones del diseño mediante componentes reutilizables. Los cambios realizados en un componente específico se reflejan automáticamente en todo el prototipo, lo cual ahorra tiempo y asegura consistencia visual y funcional en cada versión [106]. Esto permite realizar ajustes rápidos basados en el *feedback*, sin modificar manualmente cada instancia del diseño.

4.2.5. Desarrollo de aplicaciones móviles

Las aplicaciones móviles son programas creados para ejecutarse en dispositivos portátiles, como teléfonos inteligentes y tabletas, y han transformado el entorno digital al ofrecer acceso constante a una amplia gama de servicios y funcionalidades. Diseñadas para pantallas táctiles, estas aplicaciones optimizan la experiencia del usuario (UX) al aprovechar características nativas del dispositivo, como la cámara y el GPS, lo cual las hace más robustas que los sitios web convencionales [109]. Existen tres tipos principales de aplicaciones móviles: las nativas, diseñadas específicamente para un sistema operativo, lo que permite un rendimiento y una integración óptimos con el *hardware* [110]; las híbridas, que operan en varias plataformas con una base de código común, aunque con ciertas limitaciones de rendimiento [111]; y las aplicaciones web, accesibles desde un navegador, que ofrecen una funcionalidad reducida pero no requieren instalación [112].

El proceso de desarrollo de una aplicación móvil se estructura en varias fases. La fase inicial de análisis define los requisitos, objetivos y expectativas, donde el estudio de mercado y la competencia son fundamentales para crear una aplicación competitiva [113]. En la fase de diseño, se desarrollan maquetas y prototipos centrados en la experiencia e interfaz de usuario, asegurando una interacción fluida e intuitiva [114]. Durante la implementación, se crea tanto la interfaz visual como la lógica interna de la aplicación, integrando bases de datos y servicios externos cuando es necesario [115]. La fase de pruebas garantiza el correcto funcionamiento en diferentes dispositivos, y el mantenimiento final incluye la resolución de errores y la actualización continua para asegurar la usabilidad y rendimiento de la aplicación en el tiempo [116].

Concepto	Descripción
Aplicaciones móviles	Programas diseñados para ejecutarse en dispositivos portátiles como teléfonos inteligentes y tabletas, optimizados para pantallas más pequeñas y capacidades táctiles.
Funciones nativas	Capacidad de las aplicaciones móviles para aprovechar funciones del dispositivo, como cámara, GPS y sensores de movimiento, ofreciendo funcionalidades más robustas que las versiones *web*.
Tipos de aplicaciones	Se clasifican en tres tipos: *Nativas*: Diseñadas para un sistema operativo específico, con acceso completo al *hardware*. *Híbridas*: Desarrolladas con tecnologías *web*, se ejecutan en múltiples plataformas con una sola base de código. *Web*: Sitios *web* adaptados para uso móvil, accesibles a través de un navegador, con menor funcionalidad que las aplicaciones nativas o híbridas.
Ciclo de desarrollo	Proceso estructurado que incluye: *Análisis*: Definición de requisitos y análisis de mercado. *Diseño*: Creación de maquetas y prototipos enfocados en UX y UI. *Implementación*: Codificación de la aplicación (*frontend* y *backend*). *Pruebas*: Validación a través de pruebas unitarias, de integración y de aceptación.

Cuadro 4.4. Resumen de conceptos en el desarrollo de aplicaciones móviles

Lenguajes de programación

Un lenguaje de programación es un conjunto de instrucciones que permite a los programadores comunicarse con un sistema informático para realizar tareas específicas. Estos lenguajes proporcionan reglas gramaticales y semánticas que los desarrolladores usan para escribir código, el cual luego es traducido y ejecutado por el *hardware* [117]. Los lenguajes de programación son esenciales en el desarrollo de *software*, ya que permiten definir el comportamiento de un sistema, desde la interfaz de usuario hasta la gestión de datos y la integración de servicios externos [118].

Frameworks y herramientas de desarrollo

En el desarrollo de aplicaciones móviles, los *frameworks* y herramientas son esenciales, ya que simplifican el proceso de creación de *software* al proporcionar código, estilos y componentes reutilizables y funcionalidades predefinidas que ahorran una enorme cantidad de trabajo al momento de desarrollar. Al ofrecer una estructura base y facilitar el acceso a funciones nativas del dispositivo, los *frameworks* permiten a los desarrolladores enfocarse en la lógica de negocio y la experiencia de usuario, optimizando también el desarrollo multiplataforma.

La relación entre un *framework* y un lenguaje de programación es clave. El *framework* actúa como una capa de soporte sobre el lenguaje, extendiendo sus capacidades y organizando el código de manera eficiente. Esta integración permite aprovechar al máximo el lenguaje subyacente, facilitando el desarrollo de aplicaciones robustas y reduciendo los tiempos de desarrollo [119].

Desarrollo *frontend* y *backend*

En el desarrollo de aplicaciones móviles, es fundamental comprender la distinción entre el *frontend* y el *backend*, ya que ambos componentes desempeñan roles esenciales en la estructura y funcionamiento de una aplicación. El *frontend* es la parte visible e interactiva que el usuario experimenta, mientras que el *backend* maneja la lógica de negocio, la gestión de datos y la interacción con los servidores. Uno de los lenguajes más utilizados en el desarrollo *frontend* es *JavaScript*, especialmente cuando se emplean *frameworks* como *React Native* para la creación de aplicaciones móviles.

Application Programming Interface (*API*)

Las *API* son esenciales en la interacción entre el *frontend* y el *backend*, ya que definen un conjunto de reglas para que diferentes sistemas se comuniquen entre sí, facilitando el intercambio de datos y funcionalidades [120]. En una aplicación móvil, el *frontend* usa llamadas *API* para solicitar o enviar información al *backend*, como al consultar un catálogo de productos o procesar pagos [121].

Las *API* permiten desacoplar el *frontend* del *backend*, lo que facilita el mantenimiento y la escalabilidad de la aplicación, ya que los cambios en una capa no afectan directamente a la otra [122]. Además, ofrecen integración con servicios externos, ampliando las capacidades de la aplicación sin tener que desarrollar todas las funciones internamente.

Pruebas unitarias

En el desarrollo de aplicaciones móviles, las pruebas son una etapa fundamental para garantizar la calidad, fiabilidad y funcionamiento correcto de la aplicación antes de ser lanzada al mercado. Existen diversos métodos de prueba que los desarrolladores utilizan para verificar el comportamiento del *software* y corregir posibles errores. Entre los más importantes se encuentran las pruebas unitarias y las pruebas de interfaz de usuario.

Las pruebas unitarias son un tipo de prueba que se centra en la validación de pequeñas unidades de código de manera individual. Estas unidades suelen ser funciones o métodos específicos dentro de la aplicación, y el objetivo es asegurarse de que cada una de ellas funcione de manera independiente y como se espera [123]. Las pruebas unitarias son esenciales para detectar errores en etapas tempranas del desarrollo y garantizar que el código base sea confiable [124]. Además, la automatización de pruebas unitarias facilita la integración continua y el mantenimiento del código a lo largo del tiempo.

4.2.6. Evaluación y métricas de impacto

La evaluación y las métricas de impacto son esenciales para medir la efectividad de un proyecto, permitiendo analizar el cumplimiento de objetivos mediante datos cuantitativos y cualitativos. La evaluación implica un proceso sistemático de recopilación de información para determinar el valor de un proyecto y garantizar el uso eficiente de los recursos [125]. Las métricas de impacto, en cambio, son indicadores específicos que evalúan el grado de éxito de una intervención y sus efectos a largo plazo, seleccionándose según la naturaleza del proyecto y los resultados deseados [126].

Este proceso incluye el análisis de los resultados comparados con un estado base, a través de evaluaciones formativas, que permiten ajustes en tiempo real, y sumativas, que miden los resultados finales [127]. En conjunto, la evaluación y las métricas de impacto facilitan decisiones informadas, aseguran la sostenibilidad y el éxito del proyecto, y ayudan a justificar su continuidad a largo plazo [128].

Indicadores de rendimiento clave

Los indicadores de rendimiento clave (*KPI*) son métricas esenciales para evaluar el éxito de una aplicación en relación con sus objetivos estratégicos, especialmente en contextos como la gestión de residuos y la reducción de la huella de carbono. Seleccionar los *KPI* adecuados permite a los desarrolladores y responsables evaluar el desempeño de la aplicación y guiar mejoras continuas [129].

Entre los *KPI* destacados están la tasa de adopción, que mide cuántos usuarios descargan y utilizan la aplicación, y la retención de usuarios, que evalúa su compromiso a lo largo del tiempo [130]. Para los objetivos ambientales, se pueden medir la reducción de residuos y la disminución de la huella de carbono, reflejando el impacto sostenible de la aplicación [131]. Los *KPI* deben ser *SMART* (específicos, medibles, alcanzables, relevantes y limitados en el tiempo) para asegurar que los datos obtenidos sean útiles y orienten decisiones efectivas [132].

Métricas de usabilidad

Las métricas de usabilidad son esenciales para evaluar la calidad de la experiencia de usuario en una aplicación móvil, proporcionando información sobre cómo interactúan los usuarios y ayudando a identificar áreas de mejora [80]. En el desarrollo de una aplicación para la gestión de residuos, estas métricas garantizan que los usuarios puedan cumplir sus objetivos de manera eficiente.

Entre las métricas clave están la satisfacción del usuario, que se mide mediante encuestas y refleja el cumplimiento de sus expectativas [133]; el tiempo de tarea, que indica la eficiencia al medir cuánto tiempo toma completar una tarea específica [134]; y la tasa de error, que identifica problemas de diseño al contar los errores cometidos durante el uso [85]. La facilidad de uso, medida a través de pruebas de usabilidad, evalúa la simplicidad y la curva de aprendizaje de la aplicación, asegurando una experiencia fluida para los usuarios [135].

Métodos de validación

La validación de la efectividad de una aplicación es fundamental para asegurar el cumplimiento de sus objetivos y el impacto proyectado. Entre los métodos comunes de validación se encuentran las pruebas de campo, las entrevistas y los estudios de caso [136]. Las pruebas de campo permiten evaluar la aplicación en un entorno real, recopilando datos cuantitativos sobre su uso y su impacto, como la reducción de residuos y la disminución de la huella de carbono, además de identificar posibles problemas técnicos [137].

Las entrevistas ofrecen una perspectiva cualitativa al profundizar en la experiencia del usuario, recopilando opiniones y sugerencias para mejorar la aplicación, y validando el impacto social y emocional sobre la sostenibilidad [138]. Los estudios de caso, por su parte, permiten un análisis detallado de la experiencia de un grupo específico de usuarios, evaluando tanto el impacto cuantitativo como los cambios en actitudes y comportamientos sostenibles [139].

4.3. Marco teórico sobre *backend*, gestión de base de datos y análisis de datos

4.3.1. *Backend*

¿Qué es un *backend*?

El *backend* se refiere a la parte de una aplicación o sistema que se encarga de la lógica de negocio, el procesamiento de datos y la interacción con la base de datos, es decir, el lado del servidor. A diferencia del *frontend*, que es la interfaz visible para el usuario, el *backend* maneja las solicitudes y respuestas que posibilitan las funcionalidades internas de la aplicación [140]. En términos generales, el *backend* actúa como el puente que conecta el usuario final con la base de datos y otros servicios de la aplicación [141].

Componentes de un *backend*

El *backend* de una aplicación se compone de varios elementos fundamentales: el servidor, la base de datos y el propio código de la aplicación. El servidor es la infraestructura donde se ejecutan las solicitudes de los clientes; las bases de datos almacenan y gestionan los datos de la aplicación; y el código de *backend* define la lógica de negocio y los procesos de la aplicación [142]. Además, los *backend* modernos suelen incluir *API* que facilitan la comunicación entre el *frontend* y otros servicios externos [143].

Backend más comunes

Existen diferentes tecnologías y *frameworks* utilizados comúnmente para desarrollar *backend*, entre ellos *Node.js*, que destaca por su naturaleza asíncrona y capacidad para manejar múltiples solicitudes simultáneamente [144], *Django*, un *framework* basado en *Python* conocido por su robustez y seguridad [145], y *Ruby on Rails*, que simplifica el desarrollo al proporcionar una estructura organizada y preconfigurada para aplicaciones web [146]. Estos *frameworks* han ganado popularidad debido a su facilidad de uso, eficiencia y adaptabilidad a diferentes tipos de aplicaciones.

4.3.2. Tecnologías para el desarrollo de aplicaciones escalables

Introducción a la escalabilidad en aplicaciones móviles

La escalabilidad en el desarrollo de aplicaciones móviles es la capacidad de un sistema para manejar un número creciente de usuarios, datos y transacciones sin comprometer el rendimiento. Esto implica que el sistema puede crecer y adaptarse a demandas cada vez mayores sin necesidad de realizar cambios estructurales significativos. En el contexto de aplicaciones para la gestión de residuos, la escalabilidad es esencial porque se espera que el volumen de datos crezca significativamente a medida que más usuarios adopten la plataforma. Según [144], la escalabilidad no solo se refiere al *hardware*, sino también a cómo se diseñan las aplicaciones para garantizar que puedan distribuir la carga de manera eficiente a lo largo del tiempo. Esto incluye aspectos como el diseño de la base de datos y la capacidad de la *API* para manejar múltiples solicitudes simultáneas.

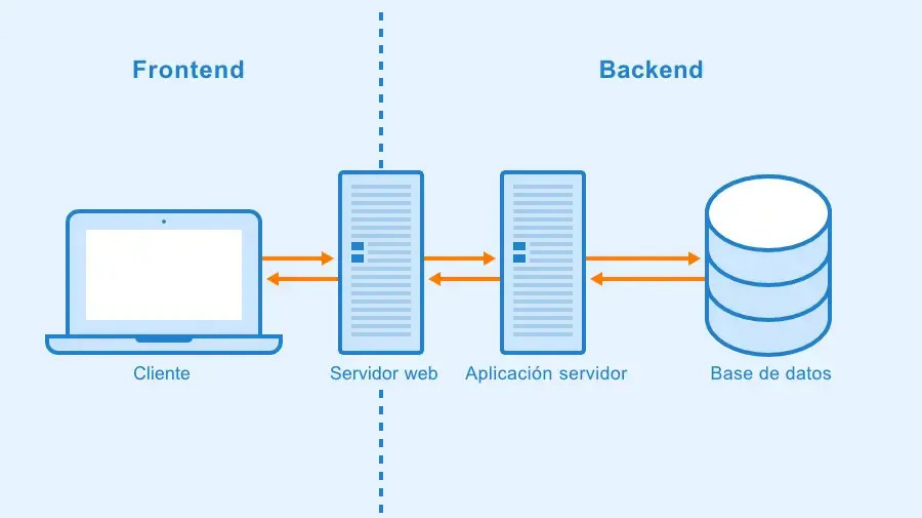


Figura 4.7. Arquitectura cliente-servidor: comunicación entre *frontend* y *backend* [5]

Tecnologías actuales para el desarrollo de aplicaciones escalables

Existen diversas tecnologías que permiten que las aplicaciones móviles sean escalables. Estas tecnologías incluyen tipos de bases de datos y el uso de *framworks* que permiten la construcción de *API RESTful* para la comunicación entre sistemas. La elección de la tecnología adecuada depende del tamaño de la aplicación, el tipo de datos que se manejará y los requisitos de rendimiento.

Plataformas para creación de *API* (*Node.js, Django, Flask*, etc.)

Las plataformas para la creación de *API* son fundamentales para el desarrollo de aplicaciones escalables. *Node.js* es una opción ampliamente utilizada debido a su naturaleza asíncrona y su capacidad para manejar múltiples solicitudes de manera eficiente. Esto la convierte en una herramienta ideal para construir *API* en tiempo real, como las de gestión de residuos [144]. Por otro lado, *Django*, un *framework* basado en *Python*, es conocido por su robustez y capacidad para gestionar aplicaciones complejas de manera eficiente. *Django* incluye un *ORM (Object-Relational Mapping)* que facilita la interacción con bases de datos relacionales, lo que lo hace adecuado para API que requieren seguridad y escalabilidad [145]. Finalmente, *Flask*, también basado en *Python*, es una alternativa más ligera que se puede utilizar cuando se necesita una mayor flexibilidad y personalización en el desarrollo de la *API*.

Bases de datos relacionales vs no relacionales (*SQL* vs *NoSQL*)

La elección de la base de datos es crítica para la escalabilidad de una aplicación. Las bases de datos *SQL*, como *PostgreSQL* y *MySQL*, son bases de datos relacionales que permiten una estructura clara y bien definida de los datos. Estas bases de datos son adecuadas para aplicaciones que requieren transacciones complejas y consistencia de los datos [147]. En cambio, las bases de datos *NoSQL*, como *MongoDB* o *Cassandra*, son más adecuadas cuando se manejan grandes volúmenes de datos no estructurados o cuando la disponibilidad es prioritaria sobre la consistencia [148]. Las bases de datos *NoSQL* permiten la escalabilidad horizontal, lo que significa que es posible distribuir los datos a través de múltiples servidores, lo que es ideal para aplicaciones que crecen rápidamente.

Características	**Bases de datos relacionales**	**Bases de datos no relacionales**
Modelo de datos	Basado en tablas con filas y columnas, usa relaciones (*joins*).	Basado en documentos, grafos, pares clave-valor, o modelos de columnas.
Estructura	Esquema fijo y predefinido (estructurado).	Esquema flexible (puede ser no estructurado).
Integridad de los datos	Alta integridad mediante claves primarias y foráneas.	Baja o variable integridad; depende de la aplicación.
Escalabilidad	Vertical (aumentar recursos del servidor).	Horizontal (añadir más servidores/nodos).
Consultas	Lenguaje *SQL*, ideal para datos estructurados y relaciones complejas.	Varía (*NoSQL*), más eficiente para grandes volúmenes y consultas rápidas.
Transacciones	Soporte completo para ACID (Atomicidad, Consistencia, Aislamiento, Durabilidad).	Puede no soportar ACID completamente, prioriza la disponibilidad y escalabilidad (BASE).
Ejemplos	*MySQL*, *PostgreSQL*, *Oracle*.	*MongoDB*, *Cassandra*, *Redis*.

Cuadro 4.5. Comparación entre bases de datos relacionales y no relacionales

API restful y protocolos de comunicación

Las *API RESTful* son la arquitectura preferida para el desarrollo de aplicaciones escalables debido a su simplicidad y eficiencia. *REST* (*Representational State Transfer*) utiliza los métodos HTTP estándar como *GET*, *POST*, *PUT* y *DELETE* para interactuar con los recursos del sistema [149]. Una *API* bien diseñada garantiza que la comunicación entre el *frontend* y la base de datos sea rápida y segura. Además, *REST* permite que las aplicaciones escalen al distribuir las solicitudes de manera eficiente entre múltiples servidores, lo que es crucial en una aplicación de gestión de residuos donde los usuarios envían datos de manera continua.

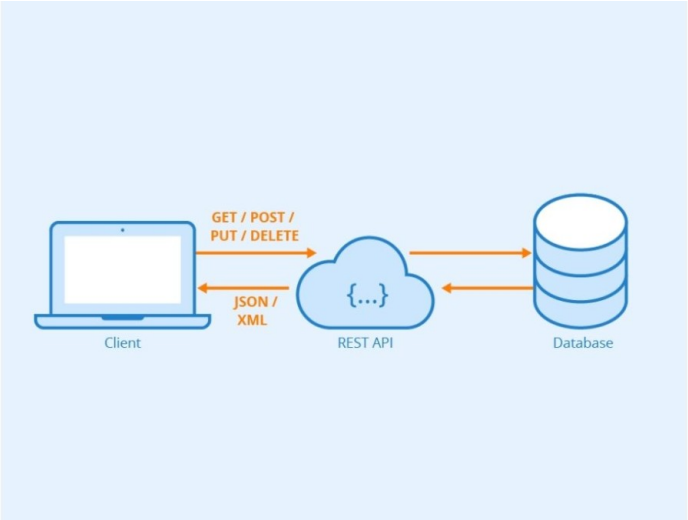


Figura 4.8. Diagrama de comunicación *REST API*. Fuente: [6]

Factores para la selección de tecnologías adecuadas en proyectos móviles

Seleccionar las tecnologías adecuadas para el desarrollo de aplicaciones móviles implica considerar varios factores, como la escalabilidad, el rendimiento, la facilidad de uso y los costos de mantenimiento. Según [148], las plataformas y las bases de datos deben seleccionarse con base en su capacidad para manejar el volumen esperado de datos y usuarios, así como en su capacidad para adaptarse a futuras necesidades de la aplicación. También es importante considerar la comunidad de soporte y la documentación disponible para cada tecnología, lo que facilita su implementación y resolución de problemas a lo largo del desarrollo del proyecto.

4.3.3. Bases de datos en la gestión de residuos y clasificación de desechos

Diseño de bases de datos para aplicaciones de gestión de residuos

El diseño de la base de datos es fundamental para cualquier sistema de gestión de residuos, ya que los datos deben almacenarse de manera eficiente para garantizar que la aplicación funcione correctamente. Una base de datos bien estructurada permite que la información de los usuarios, los residuos y las transacciones se organice de manera que sea fácilmente accesible y procesable. El diseño de una base de datos para este tipo de aplicaciones implica la creación de un modelo de datos que refleje la realidad del problema que se está abordando, como la clasificación de residuos y la frecuencia con que los usuarios generan estos desechos.

Modelos entidad-relación (E-R) aplicados a la gestión de residuos

El *modelo Entidad-Relación (E-R)* es una herramienta útil para el diseño de bases de datos, ya que permite identificar las entidades principales del sistema, como los usuarios, los tipos de residuos y las transacciones, y las relaciones entre estas entidades [150]. En una aplicación de gestión de residuos, las relaciones entre los usuarios y los tipos de residuos son clave para garantizar que los datos se organicen de manera lógica y eficiente. Además, el modelo E-R facilita la identificación de atributos clave para cada entidad, como la frecuencia de desecho de cada tipo de residuo por parte de los usuarios, lo que es esencial para generar métricas útiles.

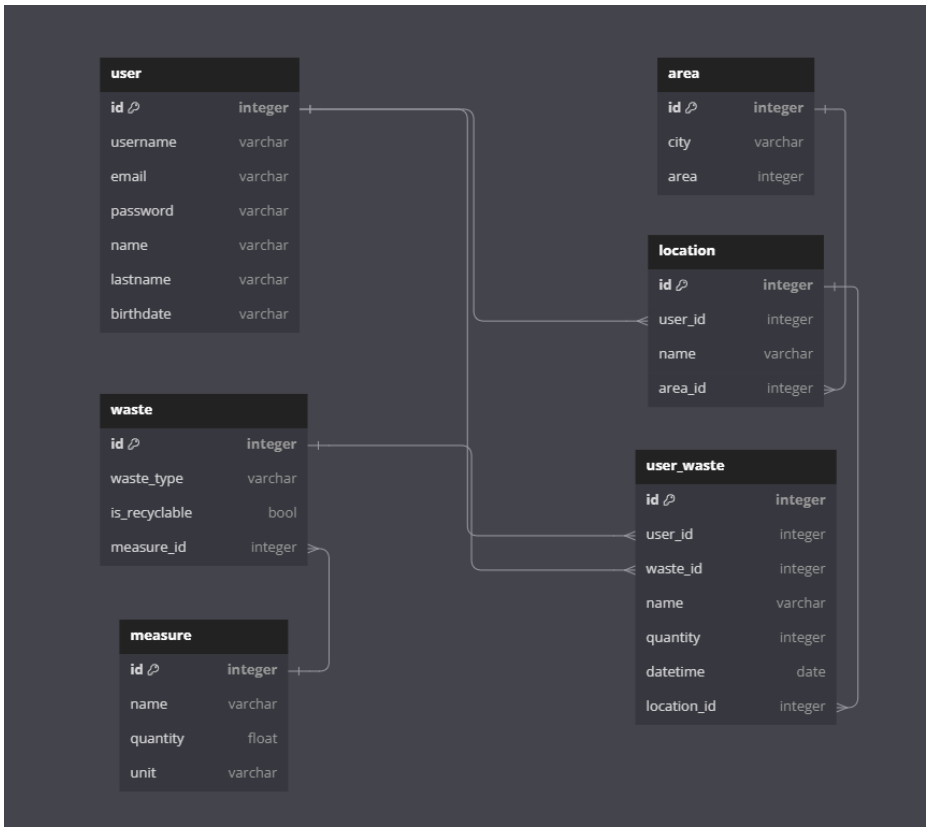


Figura 4.9. Diagrama de la base de datos. Elaboración propia

Estructuración de datos: tipos de residuos y frecuencia de desecho

La estructuración de datos es un aspecto crítico en cualquier base de datos. En el caso de la gestión de residuos, es importante almacenar información detallada sobre los tipos de residuos que se generan y la frecuencia con que los usuarios realizan el desecho de estos. Esta información permite no solo el seguimiento de las prácticas de los usuarios, sino también la generación de informes que pueden ayudar a mejorar la eficiencia del sistema. De acuerdo con [151], una base de datos bien estructurada no solo facilita el almacenamiento de los datos, sino que también mejora la capacidad del sistema para generar consultas eficientes y precisas.

Normalización de bases de datos: asegurando la eficiencia y consistencia

La *normalización* de bases de datos es un proceso que tiene como objetivo eliminar redundancias y asegurar que los datos estén organizados de manera eficiente. Esto se logra dividiendo las tablas de la base de datos en entidades más pequeñas y asegurando que cada entidad tenga un único propósito. Según [151], la normalización se lleva a cabo en diferentes etapas, conocidas como formas normales.

Primeras, segundas y terceras formas normales

La primera forma normal (1NF) asegura que los datos en cada tabla estén en formato tabular, es decir, que no existan grupos repetidos de datos. La segunda forma normal (2NF) elimina la dependencia parcial de las claves primarias, mientras que la tercera forma normal (3NF) elimina

dependencias transitivas [150]. En un sistema de gestión de residuos, la normalización asegura que cada tipo de residuo y cada usuario se gestionen de manera independiente, lo que facilita el análisis posterior de los datos.

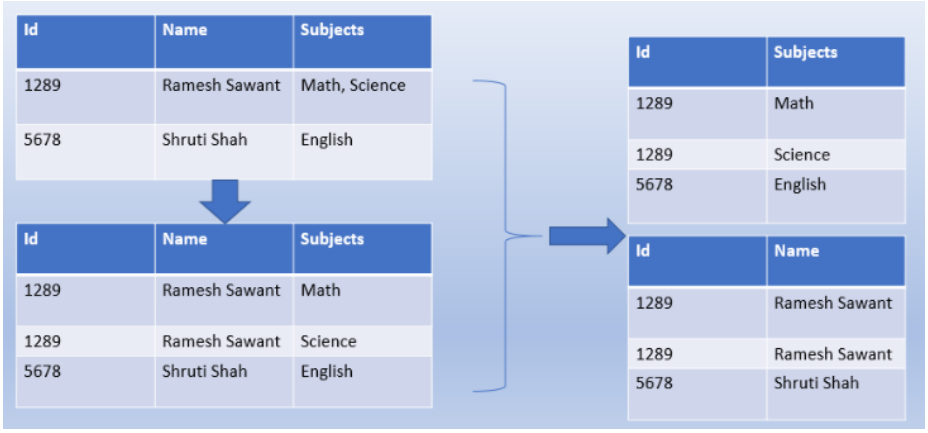


Figura 4.10. Representación de la Segunda Forma Normal (*Second Normal Form*). Fuente: [7]

Ventajas y desventajas de la normalización en sistemas escalables

La normalización tiene varias ventajas, incluida la eliminación de redundancias, lo que reduce el tamaño de la base de datos y mejora su eficiencia. Sin embargo, también tiene algunas desventajas, como el aumento en la complejidad de las consultas, especialmente en sistemas grandes y escalables. Según [150], en algunos casos puede ser beneficioso desnormalizar partes de la base de datos para mejorar el rendimiento, aunque esto debe hacerse con precaución para evitar inconsistencias.

Gestión de transacciones: principios ACID y su aplicación

El manejo de transacciones en bases de datos es esencial para garantizar que las operaciones se realicen de manera segura y consistente. Los principios ACID (Atomicidad, Consistencia, Aislamiento y Durabilidad) aseguran que las transacciones en la base de datos sean correctas y que los datos permanezcan consistentes, incluso en situaciones de fallo [152]. En el caso de una aplicación de gestión de residuos, estos principios son fundamentales para garantizar que la información sobre los residuos clasificados por los usuarios se almacene correctamente y que los datos puedan recuperarse sin errores.

Atomicidad, consistencia, aislamiento y durabilidad

La atomicidad asegura que todas las operaciones dentro de una transacción se completen correctamente, o no se realicen en absoluto. La consistencia garantiza que la base de datos pase de un estado válido a otro, mientras que el aislamiento asegura que las transacciones concurrentes no interfieran entre sí. Finalmente, la durabilidad garantiza que, una vez que una transacción se completa, los datos no se perderán incluso si el sistema falla [152]. Estos principios son críticos para asegurar que los datos de los usuarios sobre la clasificación de residuos y otros comportamientos se almacenen y procesen correctamente.

4.3.4. Desarrollo de una *API*

Introducción a la arquitectura cliente-servidor

El diseño de una *API (Application Programming Interface)* es crucial para la comunicación entre el *frontend* (la interfaz de usuario) y la base de datos de una aplicación móvil. En el caso de una aplicación de gestión de residuos, la *API* actúa como un intermediario que permite que los datos del usuario se envíen al servidor y se almacenen en la base de datos, y que las respuestas se envíen de vuelta al usuario en tiempo real. Según [153], una *API* bien diseñada permite que la aplicación se comunique de manera eficiente y segura, facilitando la escalabilidad del sistema.

Principios de diseño de una *API restful* para aplicaciones móviles

El diseño de una *API RESTful* implica el uso de los principios de *REST* (*Representational State Transfer*), que utiliza los métodos *HTTP* (*GET*, *POST*, *PUT*, *DELETE*) para interactuar con los recursos del sistema [149]. Las *API RESTful* son ligeras y fáciles de implementar, lo que las convierte en una opción popular para aplicaciones móviles que requieren comunicación en tiempo real. Además, *REST* es escalable y flexible, permitiendo que los desarrolladores añadan nuevas funcionalidades sin tener que reescribir grandes partes del código existente.

Endpoints, métodos *HTTP* (*GET, POST, PUT, DELETE*)

En una *API RESTful*, los *endpoints* son los puntos de acceso a los recursos del sistema. Cada *endpoint* representa una entidad específica, como los datos del usuario o la información sobre los tipos de residuos. Los métodos *HTTP (GET, POST, PUT, DELETE)* permiten realizar operaciones sobre estos recursos. Por ejemplo, *GET* se utiliza para recuperar información, mientras que *POST* se utiliza para crear nuevos registros. [153] destacan que una *API* bien estructurada debe seguir los principios *REST* para asegurar que la comunicación entre el *frontend* y la base de datos sea eficiente y fácil de escalar.

Seguridad en la transmisión de datos (autenticación, encriptación)

La seguridad es un aspecto fundamental en el diseño de una *API*, especialmente cuando se manejan datos sensibles, como el comportamiento de los usuarios y su huella de carbono. Para garantizar la seguridad en la transmisión de datos, se utilizan técnicas como la *autenticación* de usuarios, que asegura que solo usuarios autorizados puedan acceder a los recursos. Además, la encriptación de los datos mediante HTTPS es esencial para evitar que la información sea interceptada durante la transmisión [154]. Esto garantiza que los datos de los usuarios estén protegidos contra accesos no autorizados y ataques de intermediarios.

Integridad de los datos transmitidos: mecanismos de validación y seguridad

La integridad de los datos es crucial para asegurar que la información que se transmite entre el *frontend* y la base de datos sea precisa y esté completa. Los mecanismos de validación de datos aseguran que los datos enviados por el usuario sean correctos antes de que se almacenen en la base de datos. Esto incluye la validación de los tipos de datos, la longitud de los campos y la estructura del contenido [155]. Además, es importante implementar técnicas de seguridad como el uso de *tokens* de autenticación (por ejemplo, JWT) para proteger los datos contra accesos no autorizados [154].

4.3.5. Análisis de datos generados por los usuarios

Introducción al análisis de datos en la gestión de residuos

El análisis de datos es fundamental para extraer valor de la información generada por los usuarios en una aplicación de gestión de residuos. Los datos, como la clasificación de residuos y la frecuencia de desecho, pueden analizarse para identificar patrones y proporcionar recomendaciones a los usuarios. Según [156], el análisis de datos permite descubrir tendencias en el comportamiento de los usuarios que pueden ayudar a mejorar la eficiencia del sistema y fomentar prácticas más sostenibles entre los usuarios.

Técnicas para procesar y analizar datos

Existen varias técnicas de análisis de datos que son útiles en el contexto de la gestión de residuos. Entre las más comunes se encuentran la minería de datos, que permite extraer patrones a partir de grandes volúmenes de datos, y el análisis predictivo, que puede anticipar el comportamiento futuro de los usuarios basado en datos históricos. De acuerdo con [156], estas técnicas son útiles para identificar qué tipos de residuos se generan con mayor frecuencia y cómo pueden mejorarse los hábitos de desecho de los usuarios.

Minería de datos y análisis de tendencias

La minería de datos es un proceso que utiliza algoritmos avanzados para analizar grandes volúmenes de datos y descubrir patrones ocultos. En una aplicación de gestión de residuos, la minería de datos puede ayudar a identificar tendencias en el comportamiento de los usuarios, como la frecuencia con la que clasifican ciertos tipos de residuos o los días de la semana en que generan más desechos. Estas tendencias pueden ser utilizadas para proporcionar recomendaciones personalizadas a los usuarios y para mejorar el diseño de la aplicación [156].

Herramientas de análisis (*Python, R, SQL*)

Las herramientas de análisis de datos, como *Python, R y SQL*, son ampliamente utilizadas para procesar y analizar grandes volúmenes de datos. *Python* es particularmente útil debido a sus bibliotecas de análisis de datos, como *Pandas* y *NumPy*, que permiten manipular y analizar datos de manera eficiente [157]. *R*, por su parte, es popular en el análisis estadístico y visualización de datos, mientras que *SQL* se utiliza para consultar y extraer datos directamente desde bases de datos relacionales. Estas herramientas facilitan el procesamiento y análisis de los datos generados por los usuarios en tiempo real.

Métricas para la reducción de la huella de carbono

Una de las funciones clave de la aplicación es la capacidad de generar métricas relacionadas con la huella de carbono de los usuarios. Estas métricas pueden incluir información sobre la cantidad de residuos reciclados por cada usuario, la frecuencia con la que desechan ciertos tipos de residuos y el impacto ambiental de sus acciones. Según [157], las herramientas de análisis de datos permiten generar estas métricas de manera automática, proporcionando retroalimentación útil para los usuarios y ayudándolos a reducir su huella de carbono.

4.3.6. Escalabilidad y rendimiento en el sistema de gestión de residuos

En el contexto de la gestión de residuos, la escalabilidad y el rendimiento de la base de datos y el *API* son fundamentales, ya que un sistema mal diseñado puede fallar a medida que la cantidad de usuarios y transacciones crece. La escalabilidad se refiere a la capacidad del sistema para aumentar su rendimiento y recursos conforme crece la demanda, mientras que el rendimiento se relaciona con la rapidez y eficiencia del sistema en la ejecución de sus operaciones.

Optimización de consultas y operaciones CRUD

Las operaciones CRUD (*Create, Read, Update, Delete*) son las operaciones básicas que se realizan sobre una base de datos. Optimizar estas operaciones es esencial para mejorar el rendimiento general del sistema y asegurar que las consultas se ejecuten de manera eficiente, especialmente cuando se trabaja con grandes volúmenes de datos.

Optimización de *queries* y consultas *SQL*

La optimización de *queries SQL* es otra técnica crucial para mejorar el rendimiento de las consultas en una base de datos. Esto implica escribir consultas eficientes que utilicen los índices correctamente y eviten operaciones costosas, como los *joins* innecesarios o las subconsultas complejas. Según [150], una buena práctica es analizar las consultas utilizando herramientas de análisis de rendimiento (como el *EXPLAIN* en *PostgreSQL*) para identificar cuellos de botella y optimizar las partes que consumen más tiempo. Las consultas optimizadas reducen la carga sobre la base de datos y permiten que el sistema maneje un mayor número de transacciones sin sacrificar el rendimiento.

Pruebas de carga y rendimiento en el *backend*

Las pruebas de carga y rendimiento son esenciales para garantizar que el *backend* de la aplicación pueda manejar la demanda esperada de usuarios y transacciones sin degradar el rendimiento.

Las pruebas de rendimiento se utilizan para evaluar cómo el sistema responde bajo diferentes condiciones de carga. Estas pruebas simulan un número creciente de usuarios o transacciones para identificar cuándo el sistema comienza a tener problemas de rendimiento. Según [158], las pruebas de rendimiento deben realizarse en condiciones lo más cercanas posibles a las del entorno de producción para garantizar que los resultados sean representativos. Además, es importante realizar pruebas de estrés, que someten al sistema a una carga superior a la esperada para ver cómo responde ante situaciones extremas.

4.3.7. *CI/CD*: integración y entrega continua

El desarrollo de *software* moderno hace uso de *CI/CD* (Integración Continua/Entrega Continua o Despliegue Continuo) para mejorar la eficiencia, calidad y consistencia de las aplicaciones. *CI/CD* permite la integración frecuente de código y despliegues automáticos a entornos de producción, reduciendo errores y asegurando que las aplicaciones estén listas para ser lanzadas con mayor rapidez [159–161]. Este proceso se compone de varios elementos fundamentales: pruebas, monitoreo y automatización.

Pruebas

Las pruebas en *CI/CD* son esenciales para identificar y solucionar errores en etapas tempranas del desarrollo, lo cual es especialmente importante en el contexto de *API RESTful*, donde se espera que cada cambio en el código mantenga la consistencia de los puntos de conexión y sus respuestas. A través de pruebas unitarias, de integración y de rendimiento, los equipos pueden asegurar que la *API* funcione de acuerdo con sus especificaciones sin afectar negativamente la experiencia del usuario final [162, 163]. Las herramientas para pruebas automáticas incluyen *Mocha* para pruebas unitarias, *Postman* para pruebas de *endpoints* de la *API* y *Artillery* para pruebas de carga y rendimiento en entornos de producción [164, 165]. La integración de estas pruebas en *pipelines* de CI/CD permite validaciones consistentes y rigurosas, contribuyendo a la robustez de la *API* [166].

Monitoreo

El monitoreo continuo es crucial para asegurar que la *API RESTful* mantenga un rendimiento óptimo y esté libre de errores una vez desplegada en producción. Las herramientas de monitoreo permiten rastrear métricas como tiempos de respuesta, uso de recursos y posibles errores en tiempo real, elementos esenciales para aplicaciones con un alto volumen de solicitudes. Herramientas como *Prometheus* y *Grafana* son comúnmente utilizadas para monitoreo de métricas y visualización, mientras que *Sentry* se utiliza para monitoreo de errores, ofreciendo un seguimiento detallado de excepciones en el código [167–169]. La combinación de estas herramientas permite una gestión proactiva de la *API*, ayudando a identificar y corregir problemas antes de que afecten a los usuarios finales [170].

Automatización

La automatización en *CI/CD* facilita la integración y despliegue continuo mediante la eliminación de tareas manuales repetitivas, permitiendo a los desarrolladores concentrarse en mejorar el código en lugar de en procesos de integración y despliegue. Los *scripts* de automatización configuran la ejecución de pruebas, la compilación y el despliegue de la *API* de manera eficiente, garantizando una consistencia y velocidad adecuadas al proceso de desarrollo [171, 172]. Herramientas como *Jenkins*, *CircleCI*, *GitLab CI/CD* y *GitHub Actions* permiten implementar pipelines de *CI/CD* completamente automatizados, los cuales aseguran una integración continua desde la codificación hasta la entrega en producción [173,174]. *GitHub Actions*, en particular, facilita la configuración de *workflows* de *CI/CD* directamente desde el repositorio de código, integrándose de manera nativa con *GitHub* y permitiendo a los desarrolladores automatizar tareas como pruebas, construcción y despliegue cada vez que se realiza un cambio en el código [175]. Estas herramientas permiten a los desarrolladores mantener un flujo de trabajo constante y eficiente, minimizando errores y reduciendo el tiempo de entrega [176].

4.3.8. Tecnologías utilizadas en el proyecto

El desarrollo de una aplicación para la gestión de residuos y reducción de la huella de carbono requiere el uso de diversas tecnologías que permitan no solo el manejo eficiente de los datos, sino también asegurar la escalabilidad, seguridad y facilidad de uso de la aplicación. A continuación, se presentan las principales tecnologías utilizadas en el desarrollo del proyecto, junto con las justificaciones para su selección.

Elección de tecnologías para el *backend*

Para la elección de las tecnologías utilizadas en el desarrollo del *backend* y la gestión de datos en el proyecto, se priorizó la escalabilidad, el rendimiento y la facilidad de uso. Las tecnologías seleccionadas fueron **Express.js** para el *API* y ***PostgreSQL*** como motor de base de datos. Estas tecnologías proporcionan un marco robusto que permite manejar grandes volúmenes de datos y usuarios, asegurando que la aplicación se mantenga eficiente conforme crezca el número de usuarios.

Express.js para el *API*

Express.js es un *framework* ligero y flexible basado en ***Node.js*** que permite la creación rápida y eficiente de *API RESTful*. Este *framework* fue elegido debido a su naturaleza asíncrona y su capacidad para manejar múltiples solicitudes de manera concurrente, lo que lo convierte en una opción ideal para sistemas que requieren escalabilidad y alto rendimiento [144]. Además, *Express.js* ofrece una gran modularidad, lo que facilita la adición de *middleware* y la configuración de rutas para las operaciones *CRUD* en la base de datos [177].

Express.js destaca en aplicaciones donde la baja latencia es crucial, como en la comunicación entre el *frontend* y el *API* en una aplicación de gestión de residuos. Gracias a su arquitectura asíncrona, se pueden procesar múltiples solicitudes sin bloquear el servidor, lo que garantiza tiempos de respuesta rápidos y escalabilidad sin necesidad de aumentar excesivamente los recursos del servidor.

PostgreSQL como motor de base de datos

La base de datos seleccionada fue ***PostgreSQL***, un sistema de gestión de bases de datos relacional de código abierto que soporta grandes volúmenes de datos y consultas complejas. *PostgreSQL* es compatible con las propiedades ***ACID (Atomicidad, Consistencia, Aislamiento y Durabilidad)***, lo que garantiza la seguridad y consistencia de los datos incluso en situaciones de fallos o concurrencia [147]. Además, *PostgreSQL* permite una integración eficiente con aplicaciones desarrolladas en *Express.js*, lo que facilita la gestión de operaciones *CRUD* y análisis avanzados directamente desde la base de datos [178].

PostgreSQL también soporta la escalabilidad horizontal y vertical, lo que lo convierte en una opción ideal para aplicaciones que manejan un creciente número de usuarios y transacciones [151]. Las capacidades avanzadas de *PostgreSQL* para indexación, consultas de ventana y funciones analíticas permiten que el sistema realice análisis de datos en tiempo real sin la necesidad de herramientas externas.

En conjunto, estas dos tecnologías proporcionan una base sólida para el desarrollo del sistema, permitiendo una escalabilidad eficiente y un rendimiento alto a medida que la plataforma crece.

Integración entre *Express.js* y *PostgreSQL*

La integración entre ***Express.js*** y ***PostgreSQL*** es crucial para el correcto funcionamiento de la aplicación. A través de la *API RESTful* construida con *Express.js*, las operaciones ***CRUD*** **(Crear, Leer, Actualizar, Borrar)** se realizan sobre la base de datos *PostgreSQL*, asegurando que los datos ingresados por los usuarios se almacenen y gestionen de manera eficiente [144].

El uso combinado de estas tecnologías permite que la aplicación pueda escalar sin problemas, ya que *Express.js* maneja las solicitudes de los usuarios mientras que *PostgreSQL* gestiona el almacenamiento y recuperación de datos. Además, la conexión entre ambas herramientas garantiza la integridad de las transacciones y la consistencia de los datos almacenados.

Herramientas de análisis de datos

El análisis de los datos generados por los usuarios se realiza completamente dentro de ***PostgreSQL***, aprovechando sus capacidades avanzadas para el procesamiento y análisis de datos. *PostgreSQL* no solo gestiona el almacenamiento de datos estructurados, sino que también facilita consultas complejas y operaciones analíticas que son fundamentales para extraer *insights* útiles y generar métricas sobre la clasificación de residuos y la frecuencia de desecho de los usuarios.

Gracias a funciones como agregaciones, consultas de ventana, y capacidades de *join* complejas, *PostgreSQL* permite realizar análisis detallados que ayudan a entender mejor los patrones de comportamiento de los usuarios y a mejorar la eficiencia del sistema de gestión de residuos. Además, las capacidades de indexación y optimización de consultas de *PostgreSQL* aseguran que el análisis de datos se realice de manera eficiente, incluso con grandes volúmenes de datos, garantizando tiempos de respuesta rápidos y reduciendo la carga en el sistema [178].

4.4. Marco teórico sobre inteligencia artificial

4.4.1. Tecnología: *machine learning*, inteligencia artificial y *LLM*

La inteligencia artificial (IA) y el *machine learning* (ML) han transformado diversos sectores, incluyendo la gestión de residuos. Estas tecnologías permiten el análisis de grandes volúmenes de datos, optimización de procesos y mejora en la toma de decisiones. En la gestión de residuos, IA y ML están siendo utilizados para mejorar la eficiencia en la recolección, clasificación y reciclaje, así como para desarrollar modelos predictivos que ayuden a anticipar la generación de residuos y a optimizar las rutas de recolección.

Conceptos clave de *machine learning* e inteligencia artificial

Inteligencia artificial (IA) se refiere a la capacidad de las máquinas para realizar tareas que requieren inteligencia humana, como el reconocimiento de voz, la visión por computadora y la toma de decisiones. La IA engloba varias subdisciplinas, incluyendo el *Machine Learning*, que es una de las más relevantes para la gestión de residuos.

***Machine learning* (ML)** es una rama de la IA que se enfoca en la creación de sistemas que pueden aprender y mejorar automáticamente a partir de la experiencia sin ser explícitamente programados. El ML utiliza algoritmos y modelos estadísticos para identificar patrones en los datos, lo que permite a las máquinas hacer predicciones o tomar decisiones basadas en datos históricos [179].

Redes neuronales artificiales (ANN) son un conjunto de algoritmos inspirados en la estructura y función del cerebro humano. Las ANNs se utilizan ampliamente en aplicaciones de ML para tareas como la clasificación de imágenes y la predicción de series temporales.

Modelos de lenguaje de gran escala (LLM)

Los Modelos de lenguaje de gran escala (LLM) son un tipo de IA que ha ganado gran popularidad debido a su capacidad para procesar y generar lenguaje natural de manera coherente y contextual. Estos modelos, como GPT-3 y GPT-4, están entrenados en vastos conjuntos de datos textuales, lo que les permite realizar tareas complejas como la traducción, el resumen de textos, la generación de código, y más. En el contexto de la gestión de residuos y el reciclaje, los LLM pueden ser utilizados para:

- **Análisis y resumen de documentación:** los LLM pueden analizar grandes volúmenes de documentos legislativos, informes técnicos y estudios científicos, proporcionando resúmenes y análisis que facilitan la toma de decisiones en políticas de gestión de residuos.
- **Asistencia en la comunicación:** pueden ser implementados para mejorar la comunicación con el público sobre prácticas de reciclaje, generando mensajes personalizados o respuestas a preguntas frecuentes sobre gestión de residuos.
- **Apoyo en investigación y desarrollo:** los LLM pueden ayudar en la investigación científica, facilitando la revisión de literatura y la generación de hipótesis basadas en análisis de datos textuales.

Los LLM, sin embargo, presentan desafíos, como la necesidad de vastos recursos computacionales y la preocupación por el sesgo inherente en los datos de entrenamiento. Es crucial que su implementación en el campo de la gestión de residuos sea acompañada de una supervisión ética y técnica adecuada [180].

Aplicaciones de *machine learning* e inteligencia artificial en la gestión de residuos

El uso de ML e IA en la gestión de residuos ha mostrado un gran potencial para mejorar la eficiencia y reducir costos. Algunas aplicaciones clave incluyen:

Clasificación de residuos

La clasificación automática de residuos utilizando visión por computadora es una de las aplicaciones más prometedoras. Mediante el uso de redes neuronales convolucionales (CNN), las máquinas pueden identificar y clasificar diferentes tipos de residuos, como plásticos, metales y orgánicos, en tiempo real. Esto mejora la precisión del reciclaje y reduce la necesidad de intervención manual [181].

Optimización de rutas de recolección

Los modelos predictivos basados en ML pueden analizar patrones históricos de generación de residuos y utilizar esa información para optimizar las rutas de recolección. Esto no solo reduce el tiempo y los costos operativos, sino que también disminuye las emisiones de carbono asociadas con los camiones de recolección [182].

Predicción de la generación de residuos

El ML también se utiliza para predecir la cantidad y el tipo de residuos que se generarán en determinadas áreas o períodos de tiempo. Esto permite a las municipalidades y empresas de gestión de residuos planificar con mayor precisión y asegurar que los recursos estén disponibles cuando se necesiten [183].

Estudios y ejemplos relevantes

Diversos estudios han demostrado la efectividad del uso de IA y ML en la gestión de residuos:

- Un estudio realizado por *Truong* et al. (2022) desarrolló un sistema de clasificación de residuos basado en redes neuronales profundas, que alcanzó una precisión del 95 % en la identificación de materiales reciclables [182].
- *Marin* et al. (2021) implementaron un modelo predictivo para anticipar la generación de residuos en áreas urbanas, lo que permitió optimizar la recolección y reducir costos operativos en un 20 % [183].
- Un proyecto piloto en Europa utilizó IA para mejorar la clasificación de residuos electrónicos, logrando un aumento del 30 % en la recuperación de materiales valiosos [181].

Estos estudios subrayan el papel crucial que ML e IA pueden desempeñar en la creación de sistemas de gestión de residuos más eficientes y sostenibles.

Retos y futuro del uso de IA, ML y LLM en la gestión de residuos

Aunque las aplicaciones de IA y ML en la gestión de residuos están avanzando rápidamente, existen desafíos que deben superarse. Estos incluyen la necesidad de grandes volúmenes de datos de alta calidad, la integración de estas tecnologías en sistemas de gestión de residuos existentes y la aceptación por parte de los operadores humanos. Los *LLM* también plantean desafíos éticos y técnicos, como la gestión del sesgo y la transparencia de los modelos. A medida que estas tecnologías continúan desarrollándose, se espera que desempeñen un papel cada vez más importante en la promoción de prácticas sostenibles y en la reducción del impacto ambiental de la gestión de residuos [179, 180].

4.4.2. Leyes, regulaciones y ética en el uso de la inteligencia artificial

El uso de la *Inteligencia Artificial* (IA) está en auge a nivel global, y con ello, ha surgido la necesidad de establecer marcos legales y regulatorios que aseguren su uso ético y seguro. En Guatemala, la legislación específica sobre IA aún está en desarrollo, pero existen marcos regulatorios generales que pueden aplicarse al uso de esta tecnología.

Leyes y regulaciones sobre el uso de la inteligencia artificial

Legislación en Guatemala

En Guatemala, la Ley de Protección de Datos Personales (Decreto 25-2012) es una de las normativas más relevantes que puede impactar el uso de IA, especialmente en aplicaciones que involucran el procesamiento de datos personales. Esta ley establece las bases para la recolección, almacenamiento y uso de datos, exigiendo el consentimiento de los individuos y la implementación de medidas de seguridad para proteger su privacidad [184].

Aunque no existe una regulación específica para IA en Guatemala, se han dado pasos importantes en la región para fomentar el desarrollo ético y responsable de la IA. Un ejemplo reciente es la participación de Guatemala en el Primer Foro de Ética de la Inteligencia Artificial, donde se discutieron las necesidades y desafíos regionales para implementar una IA que respete los derechos humanos y promueva el bienestar social [185].

Regulaciones internacionales

A nivel internacional, la Unión Europea ha sido pionera en el establecimiento de un marco regulatorio para la IA con la propuesta del *Reglamento de IA* (**Artificial Intelligence Act**). Este reglamento clasifica los sistemas de IA en función de su riesgo, estableciendo obligaciones legales para desarrolladores y usuarios de IA en función del nivel de riesgo que sus aplicaciones representan [186].

En los Estados Unidos, aunque no existe una regulación federal específica sobre IA, se han emitido directrices a nivel estatal y se están desarrollando normativas sectoriales para regular aplicaciones específicas de IA, como el uso de algoritmos en la toma de decisiones de crédito y empleo [187].

Ética en el uso de la inteligencia artificial

La ética en el uso de la IA es un tema crucial, dado que esta tecnología puede tener un impacto significativo en la sociedad. Los principales desafíos éticos incluyen la transparencia de los algoritmos, el sesgo en la toma de decisiones, la privacidad, y la responsabilidad ante los errores o decisiones automatizadas que perjudican a los individuos.

Transparencia y explicabilidad

Uno de los principios fundamentales en la ética de la IA es la transparencia. Los sistemas de IA deben ser capaces de explicar cómo y por qué se toman ciertas decisiones. Esto es particularmente importante en aplicaciones sensibles como la gestión de residuos, donde las decisiones automatizadas pueden afectar a comunidades enteras. La falta de transparencia puede generar desconfianza en la tecnología y dificultar su adopción [179].

Sesgo y discriminación

El sesgo en los algoritmos de IA es un problema ético ampliamente discutido. Si los datos utilizados para entrenar los modelos de IA contienen sesgos, estos pueden perpetuar y amplificar la discriminación. Por ejemplo, en la gestión de residuos, un algoritmo sesgado podría priorizar áreas urbanas ricas para la recolección en detrimento de comunidades rurales pobres. Es crucial que los desarrolladores de IA trabajen para identificar y mitigar estos sesgos [188].

Privacidad y seguridad de los datos

La privacidad es otro tema ético clave. Los sistemas de IA que recopilan y procesan grandes cantidades de datos personales deben hacerlo de manera que se proteja la privacidad de los individuos. Esto es particularmente relevante en aplicaciones de IA que monitorean o registran datos sensibles, como la recolección y gestión de residuos [184].

Responsabilidad y gobernanza de la IA

El concepto de responsabilidad en la IA se refiere a quién o qué es responsable cuando un sistema de IA falla o toma una decisión que resulta en daños. En Guatemala, la falta de regulación específica para IA plantea preguntas sobre quién es responsable en estos casos: el desarrollador del sistema, el usuario o la entidad que implementa la IA. A nivel global, se están desarrollando marcos de gobernanza para abordar estos desafíos, asegurando que la IA se utilice de manera ética y responsable [186].

4.4.3. Impacto socioeconómico de la IA en la gestión de residuos

La implementación de la *Inteligencia Artificial* (IA) en la gestión de residuos tiene un impacto significativo tanto en la economía como en la estructura social. Desde la creación de nuevos empleos hasta la optimización de recursos, la IA está transformando la forma en que las comunidades gestionan sus desechos.

Creación de empleo y economía circular

La introducción de IA en la gestión de residuos está creando nuevas oportunidades de empleo en áreas como el desarrollo de algoritmos, el análisis de datos y la operación de sistemas automatizados. A medida que las empresas adoptan tecnologías avanzadas, surge la necesidad de trabajadores con habilidades en programación, ciencia de datos y mantenimiento de *hardware*. Además, la IA está impulsando la economía circular al optimizar la reutilización de materiales y reducir el desperdicio [189].

Desigualdades digitales

A pesar de los beneficios económicos, la implementación de IA también puede exacerbar las desigualdades digitales. Las comunidades con acceso limitado a la tecnología pueden quedar rezagadas, lo que genera disparidades en la gestión de residuos y en el acceso a empleos tecnológicos. Es crucial que las políticas públicas aborden estas desigualdades y promuevan un acceso equitativo a las oportunidades creadas por la IA [188].

4.4.4. Futuro y tendencias en IA para la gestión de residuos

El futuro de la IA en la gestión de residuos promete avances significativos, impulsados por tecnologías emergentes y un enfoque creciente en la sostenibilidad.

Desarrollos emergentes

Entre las tendencias más notables se encuentran el uso de **blockchain** para la trazabilidad de residuos y la integración de *Internet de las Cosas* (IoT) con IA para la monitorización en tiempo real de la generación y gestión de residuos. Estas tecnologías permiten una mayor transparencia y eficiencia en todo el proceso de gestión, desde la recolección hasta el reciclaje [182].

Automatización completa de procesos

A largo plazo, se espera que la automatización completa de los procesos de reciclaje sea una realidad. Con la combinación de IA, robots autónomos y sensores avanzados, será posible manejar grandes volúmenes de residuos con mínima intervención humana, aumentando así la eficiencia y reduciendo los errores [180].

4.4.5. Desafíos técnicos en la implementación de IA y ML

A pesar de los avances, la implementación de IA y *Machine Learning* (ML) en la gestión de residuos enfrenta varios desafíos técnicos.

Limitaciones tecnológicas

El desarrollo de modelos de IA requiere grandes volúmenes de datos de alta calidad, lo cual puede ser un desafío en sectores como la gestión de residuos, donde la recolección de datos consistentemente etiquetados es complicada. Además, la complejidad de los modelos de **Deep Learning** dificulta su interpretación, lo que puede limitar su adopción en sectores donde la transparencia es crucial [179].

Seguridad y resiliencia de los sistemas de IA

La seguridad de los sistemas de IA es una preocupación crítica, especialmente en aplicaciones que involucran infraestructura esencial como la gestión de residuos. La resiliencia de estos sistemas frente a ataques cibernéticos y fallos técnicos es esencial para garantizar su fiabilidad y sostenibilidad a largo plazo [188].

4.4.6. Evaluación de impacto y sostenibilidad

La integración de IA en la gestión de residuos también debe ser evaluada desde una perspectiva de sostenibilidad, considerando tanto los impactos ambientales como los económicos.

Evaluación de ciclo de vida (LCA) aplicada a sistemas de IA

La Evaluación de Ciclo de Vida (LCA) es una herramienta clave para medir el impacto ambiental de los sistemas de IA, considerando todo el ciclo de vida del sistema, desde la extracción de recursos hasta la disposición final. Implementar LCA en la gestión de residuos permite identificar áreas donde se pueden reducir los impactos negativos y mejorar la sostenibilidad [**?**].

Sostenibilidad y eficiencia energética

El desarrollo y ejecución de modelos de IA pueden tener un impacto considerable en términos de consumo energético. Es crucial adoptar estrategias que minimicen este consumo, como la optimización de algoritmos y el uso de infraestructura más eficiente energéticamente [189].

4.4.7. Interdisciplinariedad en el desarrollo de IA

El desarrollo y aplicación de IA en la gestión de residuos no pueden realizarse de manera aislada; requieren un enfoque interdisciplinario que combine conocimientos técnicos con una comprensión profunda de los contextos sociales, ambientales y éticos.

Colaboración multidisciplinaria

La colaboración entre ingenieros, científicos de datos, expertos en medio ambiente y sociólogos es fundamental para el desarrollo de soluciones integrales que sean técnicamente sólidas y socialmente responsables. Esta colaboración garantiza que los sistemas de IA diseñados para la gestión de residuos no solo sean eficientes, sino que también respeten los principios de equidad y sostenibilidad [180].

Educación y formación en IA y sostenibilidad

Formar a las futuras generaciones en la intersección de la IA y la sostenibilidad es crucial para asegurar un desarrollo responsable. Esto implica la creación de programas educativos que integren la enseñanza de técnicas avanzadas de IA con una comprensión profunda de los desafíos ambientales y sociales [179].

4.4.8. Aspectos técnicos de *machine learning*

La implementación de *machine learning* (ML) e *inteligencia artificial* (IA) en la gestión de residuos requiere una comprensión profunda de los modelos, algoritmos, hiperparámetros y métricas de desempeño. En esta sección, se abordan los aspectos técnicos esenciales que facilitan la aplicación exitosa de estas tecnologías en el campo de la sostenibilidad.

Modelos de *machine learning* aplicables

Existen varios modelos de *machine learning* que son particularmente útiles en la gestión de residuos, cada uno con sus características y ventajas específicas:

- **Árboles de decisión:** son modelos de clasificación que dividen los datos en ramas basadas en decisiones binarias. Son fáciles de interpretar y útiles para la clasificación de residuos.
- **Redes neuronales convolucionales (CNN):** utilizadas para la clasificación de imágenes, como en la identificación de tipos de residuos en una cinta transportadora. Las CNN son eficaces para procesar datos visuales complejos.
- **Máquinas de soporte vectorial (*SVMs*):** útiles para la clasificación de residuos cuando las clases son separables con un margen claro. Las *SVMs* maximizan este margen, logrando alta precisión en la clasificación.
- **Redes neuronales recurrentes (*RNNs*):** se aplican en series temporales y predicción de la generación de residuos, ya que pueden manejar datos secuenciales y temporales.

Hiperparámetros en *machine learning*

Los hiperparámetros son parámetros que definen la estructura del modelo y cómo se entrena. A diferencia de los parámetros aprendidos, los hiperparámetros son fijados antes del proceso de entrenamiento. A continuación se describen varios hiperparámetros clave:

- **Tasa de aprendizaje (η):** controla el tamaño de los pasos que da el algoritmo durante la optimización. Si es demasiado alta, el modelo puede oscilar; si es demasiado baja, el modelo puede tardar mucho en converger.
- **Número de épocas:** define cuántas veces el modelo verá el conjunto completo de datos de entrenamiento. Demasiadas épocas pueden llevar al sobreajuste (*overfitting*), mientras que pocas pueden resultar en un subajuste (*underfitting*).
- **Número de neuronas y capas en redes neuronales:** afecta la capacidad del modelo para capturar relaciones complejas. Un número excesivo de neuronas puede llevar al sobreajuste.

- **Tamaño del *mini-batch*:** controla el número de muestras que el modelo procesa antes de actualizar los parámetros. Un tamaño de *mini-batch* pequeño puede hacer que el entrenamiento sea ruidoso pero rápido, mientras que un tamaño grande puede estabilizar las actualizaciones pero requiere más memoria.
- **Regularización (L2, L1):** ayuda a prevenir el sobreajuste aplicando una penalización a los parámetros del modelo. La regularización L2 es común en redes neuronales, mientras que la regularización L1 es útil para generar modelos esparsos.
- **Tasa de abandono (*Dropout*):** en redes neuronales, se refiere a la probabilidad de que una neurona sea *apagada* durante una iteración del entrenamiento para prevenir el sobreajuste.
- **Momento (*Momentum*):** es un hiperparámetro que ayuda a acelerar el gradiente descendente hacia la dirección más relevante y a evitar oscilaciones en los mínimos locales.

Ecuaciones matemáticas clave

Las siguientes ecuaciones son fundamentales en el desarrollo y entrenamiento de modelos de ML:

Función de costo (costo o pérdida)

La función de costo mide la diferencia entre las predicciones del modelo y los valores reales. En el caso de la regresión, se utiliza comúnmente el *Error Cuadrático Medio* (MSE):

$$J(\theta) = \frac{1}{2m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})^2$$

Donde:

- m es el número de ejemplos de entrenamiento.
- $h_\theta(x^{(i)})$ es la predicción del modelo.
- $y^{(i)}$ es el valor real.

Gradiente descendente

El algoritmo de gradiente descendente se utiliza para minimizar la función de costo ajustando los parámetros del modelo:

$$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta)$$

Donde:

- θ_j son los parámetros del modelo.
- α es la tasa de aprendizaje.
- $\frac{\partial}{\partial \theta_j} J(\theta)$ es el gradiente de la función de costo.

Métricas de desempeño en *machine learning*

Las métricas de desempeño son cruciales para evaluar la efectividad de los modelos de ML. Algunas de las métricas más comunes incluyen:

- **Precisión:** proporción de predicciones correctas sobre el total de predicciones realizadas.
- **Exactitud (*Accuracy*):** medida global de desempeño para un modelo de clasificación.
- ***ROC-AUC*:** area bajo la curva *ROC*, que mide la capacidad de un modelo para distinguir entre clases.
- ***F1 score*:** medida ponderada que combina precisión y *recall*, útil en escenarios con clases desbalanceadas.

Visualización de modelos y resultados

Las visualizaciones son herramientas poderosas para interpretar y comunicar los resultados de los modelos de ML. Las matrices de confusión, gráficas *ROC* y diagramas de árbol de decisión son ejemplos de visualizaciones que ayudan a entender cómo funciona un modelo.

n=192	**Predicted: 0**	**Predicted: 1**	
Actual: 0	TN = 118	FP = 12	130
Actual: 1	FN = 47	TP = 15	62
	165	27	

Figura 4.11. Matriz de confusión que ilustra la precisión del clasificador a través de verdaderos positivos, falsos positivos, verdaderos negativos y falsos negativos.

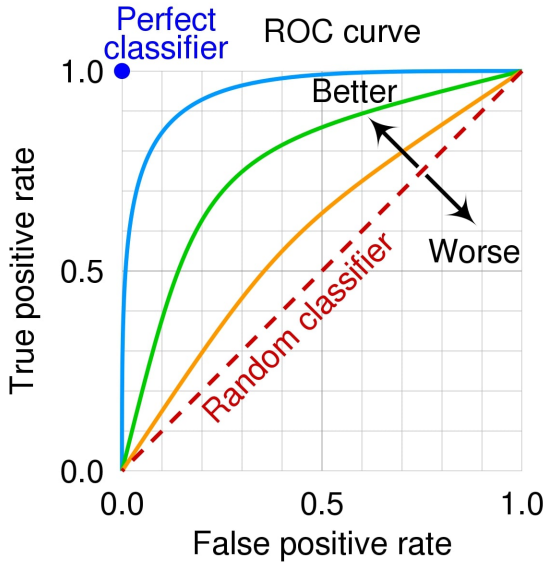


Figura 4.12. Curva *ROC* que muestra el desempeño de un clasificador binario.

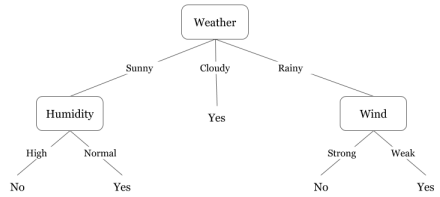


Figura 4.13. Diagrama de árbol de decisión que representa las decisiones y sus posibles consecuencias.

Regularización y penalización en modelos

La regularización es crucial para prevenir el sobreajuste en modelos complejos, especialmente cuando se trabaja con datos de alta dimensionalidad. A continuación, se describen las dos formas más comunes de regularización:

Regularización L1 y L2

La regularización L1 agrega una penalización proporcional a la suma de los valores absolutos de los coeficientes, mientras que la regularización L2 penaliza la suma de los cuadrados de los coeficientes [190].

- **L1 (*Lasso*):**

$$J(\theta) = \frac{1}{m}\sum_{i=1}^{m}\left(h_\theta(x^{(i)}) - y^{(i)}\right)^2 + \lambda\sum_{j=1}^{n}|\theta_j|$$

- **L2 (*Ridge*):**

$$J(\theta) = \frac{1}{m}\sum_{i=1}^{m}\left(h_\theta(x^{(i)}) - y^{(i)}\right)^2 + \lambda\sum_{j=1}^{n}\theta_j^2$$

Donde λ es el factor de regularización que controla la fuerza de la penalización.

Algoritmo de optimización *Adam*

El optimizador *Adam* combina las ventajas del método de *Momentum* y *Adaptive Gradient Algorithm (AdaGrad)*. Es ampliamente utilizado debido a su capacidad para ajustarse dinámicamente a diferentes características de los datos [191].

$$m_t = \beta_1 m_{t-1} + (1-\beta_1)\nabla_\theta J(\theta)$$
$$v_t = \beta_2 v_{t-1} + (1-\beta_2)(\nabla_\theta J(\theta))^2$$
$$\hat{m}_t = \frac{m_t}{1-\beta_1^t}, \quad \hat{v}_t = \frac{v_t}{1-\beta_2^t}$$

$$\theta_{t+1} = \theta_t - \frac{\alpha \hat{m}_t}{\sqrt{\hat{v}_t} + \epsilon}$$

Donde:

- α es la tasa de aprendizaje.
- β_1 y β_2 son los coeficientes de decaimiento exponencial para las estimaciones de momentos.

Regularización mediante *Dropout*

El *Dropout* es una técnica de regularización que consiste en apagar (ignorar) de manera aleatoria ciertas neuronas durante el entrenamiento. Esto previene que las neuronas se vuelvan demasiado dependientes entre sí, ayudando a evitar el sobreajuste.

Funciones de activación

Las funciones de activación son componentes críticos de las redes neuronales, ya que introducen no linealidad en el modelo, permitiéndole aprender y modelar relaciones complejas.

- **Función sigmoide:**
$$\sigma(x) = \frac{1}{1 + e^{-x}}$$
se utiliza principalmente en la última capa de una red neuronal para problemas de clasificación binaria [192].
- **Función ReLU (*Rectified Linear Unit*):**
$$f(x) = \text{máx}(0, x)$$
ReLU es la función de activación más utilizada en redes profundas debido a su simplicidad y efectividad al mitigar el problema del gradiente desvanecido [193].

Evaluación del sesgo y la varianza

El análisis de sesgo y varianza es fundamental para entender la capacidad de generalización de un modelo de *Machine Learning*.

- **Sesgo:** se refiere a la capacidad de un modelo para capturar las relaciones subyacentes entre las variables de entrada y salida.
- **Varianza:** refleja la sensibilidad del modelo a los cambios en los datos de entrenamiento. Un modelo con alta varianza puede sobreajustarse a los datos de entrenamiento [194].

Métodos de validación y evaluación

Para evaluar el rendimiento de un modelo de *Machine Learning*, es crucial usar métodos como la validación cruzada.

Validación cruzada *K-Fold*

En la validación cruzada *K-Fold*, el conjunto de datos se divide en K subconjuntos. El modelo se entrena K veces, cada vez utilizando K-1 subconjuntos para el entrenamiento y el restante para la prueba [191].

La métrica final es el promedio de los resultados obtenidos en cada iteración, lo que proporciona una estimación robusta del desempeño del modelo.

4.4.9. Aspectos técnicos de la *inteligencia artificial* y *deep learning*

Redes neuronales profundas (*Deep Neural Networks*)

Las redes neuronales profundas (DNNs) son el núcleo de muchas aplicaciones modernas de *Deep Learning*. Están compuestas por múltiples capas de neuronas, cada una de las cuales realiza transformaciones lineales y no lineales sobre los datos de entrada. Estas redes son capaces de modelar relaciones complejas en datos de alta dimensionalidad, lo que las hace ideales para tareas como la clasificación de residuos.

Modelo matemático de una neurona

El funcionamiento básico de una neurona en una DNN se describe mediante la siguiente ecuación:

$$z = \sum_{i=1}^{n} w_i x_i + b$$

Donde:

- z es la entrada ponderada a la función de activación.
- w_i son los pesos de las conexiones sinápticas.
- x_i son las entradas.
- b es el sesgo (bias).

La salida de la neurona se obtiene aplicando una función de activación $\sigma(z)$:

$$a = \sigma(z)$$

Redes neuronales convolucionales (*convolutional neural networks, CNN*)

Las redes neuronales convolucionales (CNN) son un tipo de red neuronal profunda optimizada para el procesamiento de datos estructurados espacialmente, como imágenes. En la gestión de residuos, las CNN se utilizan ampliamente para la clasificación automática de residuos a partir de imágenes.

Operación convolucional

La operación convolucional es el núcleo de una CNN. Matemáticamente, se define como:

$$S(i,j) = (I * K)(i,j) = \sum_m \sum_n I(i-m, j-n) \cdot K(m,n)$$

Donde:

- I es la matriz de la imagen de entrada.
- K es el filtro (*kernel*) aplicado.
- $S(i,j)$ es la salida de la convolución en la posición (i,j).

Redes neuronales recurrentes (*recurrent neural networks, RNNs*)

Las redes neuronales recurrentes (RNNs) son adecuadas para procesar secuencias de datos, como la predicción de generación de residuos a lo largo del tiempo. En una RNN, la salida de cada paso temporal se retroalimenta como entrada para el siguiente paso, lo que se modela mediante:

$$h_t = \sigma(W_{hh} h_{t-1} + W_{xh} x_t + b_h)$$

Donde:

- h_t es el estado oculto en el tiempo t.
- W_{hh} es la matriz de pesos de la recurrencia.
- W_{xh} es la matriz de pesos de la entrada.
- x_t es la entrada en el tiempo t.
- b_h es el sesgo.

Funciones de activación en *deep learning*

Las funciones de activación introducen no linealidad en los modelos de *Deep Learning*, permitiendo que las redes neuronales aprendan relaciones complejas en los datos.

Función sigmoide

La función sigmoide es una de las funciones de activación más antiguas:

$$\sigma(z) = \frac{1}{1 + e^{-z}}$$

Es especialmente útil en la clasificación binaria, ya que convierte cualquier valor en una probabilidad entre 0 y 1.

Función *ReLU (Rectified Linear Unit)*

La función *ReLU* se define como:

$$f(z) = \text{máx}(0, z)$$

Es la función de activación más utilizada en redes neuronales profundas debido a su simplicidad y efectividad, especialmente en la mitigación del problema del gradiente desvanecido.

Optimización y entrenamiento en *deep learning*

El proceso de entrenamiento de modelos de *Deep Learning* implica la minimización de una función de pérdida $J(\theta)$, que generalmente se optimiza mediante el método de gradiente descendente y sus variantes.

Gradiente descendente

El gradiente descendente actualiza los parámetros del modelo siguiendo la dirección opuesta al gradiente de la función de pérdida:

$$\theta := \theta - \alpha \nabla_\theta J(\theta)$$

Donde:

- α es la tasa de aprendizaje.
- $\nabla_\theta J(\theta)$ es el gradiente de la función de pérdida respecto a los parámetros θ.

Algoritmo de optimización *Adam*

El optimizador *Adam* ajusta los pesos de la red utilizando momentos de primer y segundo orden:

$$\begin{aligned}
m_t &= \beta_1 m_{t-1} + (1 - \beta_1) \nabla_\theta J(\theta) \\
v_t &= \beta_2 v_{t-1} + (1 - \beta_2)(\nabla_\theta J(\theta))^2 \\
\hat{m}_t &= \frac{m_t}{1 - \beta_1^t}, \quad \hat{v}_t = \frac{v_t}{1 - \beta_2^t} \\
\theta_{t+1} &= \theta_t - \frac{\alpha \hat{m}_t}{\sqrt{\hat{v}_t} + \epsilon}
\end{aligned}$$

Regularización en redes profundas

La regularización es esencial para evitar el sobreajuste en modelos complejos de *Deep Learning*. Técnicas comunes incluyen:

- ***Dropout*:** durante el entrenamiento, algunas neuronas se **apagan** aleatoriamente para evitar la co-adaptación excesiva. Matemáticamente, se describe mediante:

$$a^{(l)} = Dropout(h^{(l-1)}, p)$$

  Donde p es la probabilidad de que una neurona se mantenga activa.

- **Regularización L2:** agrega una penalización a la función de pérdida, proporcional al cuadrado de los pesos del modelo:

$$J(\theta) = J(\theta) + \lambda \sum_{j=1}^{n} \theta_j^2$$

Transferencia de aprendizaje (*transfer learning*)

La *Transferencia de Aprendizaje* es una técnica fundamental en *deep learning* que permite la reutilización del conocimiento adquirido por un modelo en una tarea previamente aprendida para aplicarlo en una nueva tarea relacionada. Esta técnica se basa en la premisa de que un modelo que ha sido entrenado en un conjunto de datos amplio y diverso puede captar características generales y representaciones que son útiles en una variedad de aplicaciones, incluso en dominios con datos específicos y limitados. Un ejemplo prominente de esta técnica es el uso de modelos preentrenados en conjuntos de datos extensivos, como *ImageNet*, el cual contiene millones de imágenes de categorías variadas [195].

En términos prácticos, en lugar de construir un modelo desde cero para cada tarea, el *Transfer Learning* permite aprovechar un modelo preentrenado como base, ajustando algunas de sus capas para adaptarlo a una nueva tarea específica. Este proceso de ajuste, conocido como *fine-tuning*, generalmente implica conservar y *congelar* las primeras capas de la red (las cuales capturan características generales de bajo nivel) y entrenar o modificar las capas finales para que el modelo se especialice en la nueva tarea [196]. En el ámbito de la clasificación de imágenes, esta técnica es especialmente útil porque permite reducir el tiempo de entrenamiento y mejorar la precisión al aprovechar las representaciones aprendidas por redes previamente entrenadas.

Aplicación de transferencia de aprendizaje en clasificación de imágenes

La transferencia de aprendizaje ha demostrado ser particularmente ventajosa en tareas de clasificación de imágenes donde los conjuntos de datos específicos suelen ser limitados o complejos. En la clasificación de residuos, por ejemplo, los objetos pueden variar en términos de color, forma, textura y contexto, lo cual hace que sea difícil capturar todas las variaciones mediante un conjunto de datos pequeño. Al aplicar *Transfer Learning*, se puede reutilizar la capacidad de reconocimiento de patrones aprendida por un modelo en un conjunto de datos grande y aplicarla a un contexto específico, como la clasificación de distintos tipos de residuos.

Ventajas del *transfer learning* en tareas de clasificación

La adopción del *Transfer Learning* ofrece diversas ventajas en la clasificación de imágenes, como las siguientes:

- **Reducción del tiempo de entrenamiento**: dado que el modelo ya ha aprendido características básicas y generales, el tiempo necesario para adaptarlo a una nueva tarea es significativamente menor que al entrenar un modelo desde cero [196].

- **Mejor adaptabilidad a la tarea específica**: al realizar *fine-tuning* en las capas finales de un modelo preentrenado, el modelo puede captar los patrones específicos del nuevo conjunto de datos sin perder las características generales previamente aprendidas [197].
- **Mayor precisión con conjuntos de datos limitados**: debido a que el modelo preentrenado ya cuenta con una representación compleja de características, la cantidad de datos adicionales necesarios para obtener buenos resultados en una nueva tarea es menor. Esto es ideal en situaciones donde el acceso a grandes volúmenes de datos es limitado.

La técnica de *Transfer Learning* se ha convertido en una práctica estándar en *deep learning* para tareas de clasificación, no solo por su eficiencia en recursos, sino también por su capacidad para adaptar modelos a tareas especializadas de manera rápida y efectiva. En el contexto de este proyecto, se emplean modelos preentrenados para optimizar el reconocimiento y clasificación de distintos tipos de residuos, lo cual permite aprovechar las ventajas de precisión y eficiencia computacional inherentes a esta técnica [196, 197].

Modelos de *transfer learning* en clasificación de imágenes

En el campo de la visión por computadora, el uso de modelos de *Transfer Learning* se ha vuelto predominante debido a su capacidad para aprovechar características previamente aprendidas y aplicarlas en nuevas tareas, lo que permite reducir el tiempo de entrenamiento y mejorar la precisión en entornos con datos limitados. Este proyecto emplea varios de estos modelos reconocidos en la literatura, cada uno con arquitecturas optimizadas para maximizar el rendimiento en clasificación de imágenes. A continuación, se explican brevemente los modelos *Xception, MobileNetV2, InceptionV3, VGG19* y *DenseNet201*, los cuales son ampliamente utilizados en el ámbito de la transferencia de aprendizaje y han sido ajustados para la clasificación de residuos en este estudio.

- ***Xception***: desarrollado por François Chollet, *Xception* es un modelo basado en convoluciones separables en profundidad, lo cual permite realizar cálculos eficientes y reducir el número de parámetros sin perder precisión [198]. Su arquitectura modular, inspirada en la arquitectura Inception, ha demostrado ser efectiva para tareas de clasificación de gran escala y fue ajustada en este proyecto para identificar diferentes tipos de residuos.
- ***MobileNetV2***: propuesto por Sandler et al., *MobileNetV2* se diseñó con un enfoque en dispositivos móviles y sistemas con recursos limitados. Este modelo utiliza bloques residuales invertidos y capas de embotellamiento lineal, logrando una estructura que es tanto precisa como computacionalmente eficiente [199]. Estas características lo hacen ideal para aplicaciones de clasificación en tiempo real, como las utilizadas en este proyecto.
- ***InceptionV3***: este modelo, desarrollado por Szegedy et al., utiliza bloques de convolución denominados "*Inception*" que capturan características visuales de múltiples escalas. *InceptionV3* ha sido ampliamente adoptado para clasificación de imágenes debido a su capacidad de procesamiento paralelo de patrones de diferentes tamaños, lo cual permite una extracción robusta de características [200]. En este proyecto, *InceptionV3* se aplica para capturar patrones visuales complejos en la clasificación de materiales reciclables y no reciclables.
- ***VGG19***: creado por Simonyan y Zisserman, *VGG19* es conocido por su arquitectura profunda y simple, compuesta por bloques de convolución seguidos de capas completamente conectadas. Su diseño, aunque más pesado computacionalmente, permite una extracción detallada de características visuales, lo cual es ventajoso en tareas que requieren precisión en detalles finos [201]. *VGG19* se ajusta particularmente bien para distinguir texturas y formas en los diferentes tipos de residuos.

- ***DenseNet201***: propuesto por Gao Huang et al., *DenseNet201* se caracteriza por su conectividad densa entre capas, en la cual cada capa recibe como entrada la salida de todas las capas anteriores. Esto promueve la reutilización de características y facilita la retropropagación del gradiente, mejorando la precisión del modelo con un menor número de parámetros [202]. En este proyecto, *DenseNet201* permite captar patrones recurrentes en las imágenes de residuos, aprovechando la transferencia de aprendizaje para optimizar el desempeño.

Estos modelos representan algunas de las arquitecturas de *Transfer Learning* más avanzadas, optimizadas para la clasificación de imágenes en diferentes dominios. Su aplicación en la clasificación de residuos aprovecha su capacidad para reconocer patrones complejos y diversos, necesarios para la correcta identificación de materiales y su posterior tratamiento en los flujos de reciclaje.

MobileNetV2

MobileNetV2 es una arquitectura de red neuronal convolucional diseñada para optimizar el rendimiento en dispositivos móviles y sistemas con limitaciones computacionales. Desarrollado por Mark Sandler y colaboradores, este modelo introduce innovaciones arquitectónicas significativas que mejoran su eficiencia sin comprometer su precisión en tareas de clasificación de imágenes [199].

Aspectos técnicos y arquitectura de *MobileNetV2* *MobileNetV2* se basa en dos conceptos principales: los *bloques residuales invertidos* y las *capas de embotellamiento lineal*. Estos componentes permiten reducir la cantidad de parámetros del modelo, manteniendo su capacidad para captar características visuales complejas.

- **Bloques residuales invertidos**: en la arquitectura de *MobileNetV2*, los bloques residuales invertidos son una modificación de los bloques residuales tradicionales. Mientras que los bloques residuales comunes suelen expandir el espacio de características al inicio del bloque y luego lo reducen al final, en *MobileNetV2* este proceso se invierte. La arquitectura comienza con una capa de expansión que aumenta las dimensiones de los canales, permitiendo que la convolución *depthwise* capte las características con mayor precisión. Luego, una capa de proyección reduce el número de canales, lo que ayuda a mantener la eficiencia del modelo sin perder información importante.
- **Convoluciones *depthwise* separable**: *MobileNetV2* emplea convoluciones *depthwise separable*, donde la convolución se divide en dos operaciones: una convolución *depthwise*, que aplica un filtro separado a cada canal, y una convolución *pointwise*, que combina los canales. Este método reduce considerablemente el número de cálculos en comparación con una convolución estándar, mejorando así la eficiencia sin comprometer la capacidad de aprendizaje del modelo.
- **Capa de embotellamiento lineal**: para minimizar la pérdida de información en los pasos de compresión de características, *MobileNetV2* utiliza una capa lineal de embotellamiento al final de cada bloque residual. Esta capa mantiene la representación lineal de las características y evita la distorsión de información causada por funciones de activación como ReLU en espacios de baja dimensionalidad, ya que esta podría eliminar componentes críticos de las características aprendidas.

Estructura general de la red *MobileNetV2* utiliza una estructura repetitiva de bloques de convolución *depthwise separable* y bloques residuales invertidos, intercalados con capas de embotellamiento lineal, lo que da como resultado una red eficiente y compacta. Esta arquitectura permite que el modelo sea altamente optimizado para dispositivos móviles o aplicaciones en tiempo real, ya que mantiene un equilibrio entre precisión y costo computacional.

Aplicación y rendimiento La arquitectura de *MobileNetV2* ha demostrado un rendimiento sobresaliente en tareas de clasificación y detección de objetos, especialmente en contextos donde el uso de recursos computacionales es limitado. En este proyecto, se emplea *MobileNetV2* para la clasificación de residuos debido a su eficiencia y capacidad para extraer características relevantes en imágenes con una estructura de bajo costo computacional, lo que permite reducir el tiempo de inferencia al clasificar diferentes tipos de materiales reciclables y no reciclables [199].

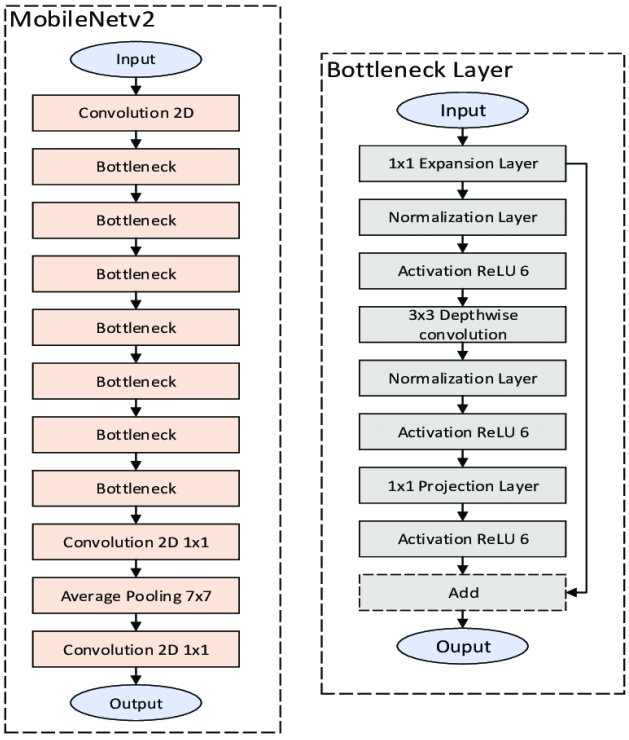


Figura 4.14. Arquitectura del modelo *MobileNetV2*.

InceptionV3

InceptionV3 es una arquitectura de red neuronal convolucional desarrollada por Christian Szegedy y su equipo en *Google*. Forma parte de la familia de arquitecturas *Inception*, conocidas por su capacidad de extraer características a diferentes escalas de manera eficiente. *InceptionV3* fue diseñado para mejorar la precisión en tareas de clasificación de imágenes manteniendo un uso eficiente de los recursos computacionales, lo que lo convierte en una opción ideal para tareas de clasificación de imágenes a gran escala [200].

Aspectos técnicos y arquitectura de *InceptionV3* *InceptionV3* introduce varias mejoras sobre sus predecesores, incluyendo módulos de convolución de diferentes tamaños y técnicas avanzadas de regularización, que optimizan su desempeño y eficiencia. Los componentes principales de *InceptionV3* son los siguientes:

- **Módulos *Inception***: la arquitectura de *InceptionV3* está organizada en módulos llamados "*Inception*" que combinan convoluciones de diferentes tamaños (por ejemplo, 1x1, 3x3 y 5x5) dentro de un mismo módulo. Esta estructura permite que la red capture características a

distintas escalas, lo que mejora la capacidad del modelo para reconocer patrones variados en las imágenes. Los módulos *Inception* fusionan múltiples dimensiones de las características extraídas en una sola representación optimizada.

- **Factorización de convoluciones**: para reducir la complejidad computacional, *InceptionV3* emplea la factorización de convoluciones. En lugar de realizar una convolución de tamaño 5x5, el modelo utiliza una secuencia de dos convoluciones 3x3, que requieren menos operaciones y mejoran la eficiencia del modelo sin comprometer su capacidad de captura de patrones complejos.
- **Convoluciones asimétricas (1x7 y 7x1)**: *InceptionV3* también emplea convoluciones asimétricas, como 1x7 y 7x1, para capturar patrones de características en una dirección específica y luego combinarlos. Esto reduce el costo computacional en comparación con las convoluciones cuadradas (por ejemplo, 7x7) y permite un análisis más fino de características dentro de la imagen.
- **Reducción del tamaño de la red mediante *pooling***: los módulos de *pooling* en *InceptionV3* se utilizan para reducir las dimensiones de las características intermedias, limitando el tamaño del modelo y mejorando su eficiencia sin perder precisión. Este enfoque permite que el modelo se ajuste a recursos computacionales limitados y sea aplicable a tareas de clasificación que requieren rapidez y precisión.

Técnicas de regularización y optimización *InceptionV3* incorpora varias técnicas de regularización para evitar el sobreajuste y mejorar la generalización del modelo:

- ***Dropout***: durante el entrenamiento, *InceptionV3* emplea la técnica de *dropout* en sus capas totalmente conectadas, lo que evita la co-adaptación de neuronas específicas y mejora la generalización del modelo al introducir aleatoriedad en el entrenamiento.
- ***Batch* normalization**: la normalización por lotes se utiliza en los módulos *Inception* para estabilizar el proceso de entrenamiento y permitir que el modelo aprenda más rápido. Esta técnica reduce la sensibilidad a la inicialización de pesos y ayuda a prevenir problemas de estancamiento en redes profundas.

Aplicación y rendimiento Gracias a su capacidad para capturar características a múltiples escalas y su eficiencia computacional, *InceptionV3* es ideal para aplicaciones de clasificación de imágenes con un alto grado de variabilidad. En el contexto de este proyecto, *InceptionV3* fue empleado para identificar patrones visuales complejos en imágenes de residuos, permitiendo una clasificación precisa y eficiente de los materiales reciclables y no reciclables [200]. Su capacidad para manejar patrones a múltiples escalas le permite ser particularmente útil en la clasificación de objetos de tamaños y formas variadas, comunes en el contexto de residuos sólidos.

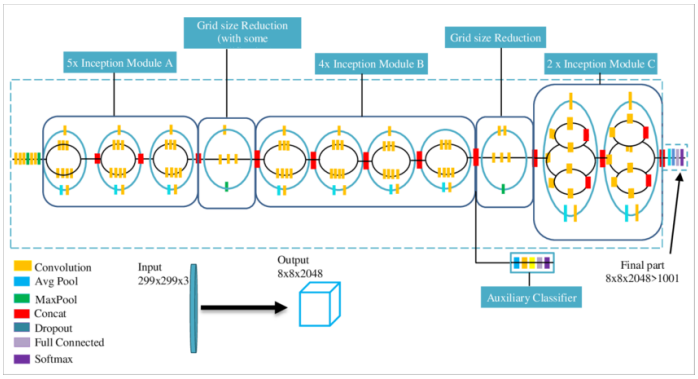


Figura 4.15. Arquitectura del modelo *InceptionV3*.

DenseNet201

DenseNet201 es una variante avanzada de las redes densamente conectadas (*DenseNets*), desarrollada por Gao Huang y colaboradores. Esta arquitectura introduce un esquema de conexión densa que conecta cada capa de la red con todas las capas anteriores, promoviendo un flujo eficiente de información y gradientes a través de la red [202]. Este diseño es especialmente útil en redes profundas, ya que ayuda a preservar la información y a evitar problemas de gradiente.

Aspectos técnicos y arquitectura de *DenseNet201* *DenseNet201* se caracteriza por sus conexiones densas y su organización en bloques convolucionales, lo cual optimiza tanto la precisión como la eficiencia en el uso de parámetros. Los elementos principales de su arquitectura son los siguientes:

- **Conexiones densas**: a diferencia de las redes tradicionales, donde cada capa transmite su salida solo a la capa siguiente, en *DenseNet201* cada capa se conecta directamente con todas las capas anteriores. Esto significa que cada capa recibe como entrada las características de todas las capas previas, permitiendo un flujo de información continuo y evitando la pérdida de detalles importantes. Estas conexiones densas mejoran el flujo de gradientes y reducen la probabilidad de problemas de desaparición del gradiente en redes profundas.
- **Bloques densos y capas de transición**: *DenseNet201* organiza sus capas en bloques densos, compuestos por varias capas convolucionales, y en capas de transición, que reducen las dimensiones espaciales mediante *pooling*. Las capas de transición también incluyen convoluciones de 1x1, que limitan el número de canales y permiten al modelo mantener una estructura compacta y eficiente.
- **Control de crecimiento de características (*growth rate*)**: para controlar la cantidad de información adicional que cada capa introduce, *DenseNet201* utiliza un parámetro denominado *growth rate*, el cual determina la cantidad de canales añadidos por cada nueva capa. Esto evita que el número de parámetros crezca de forma excesiva, manteniendo el modelo en un tamaño manejable sin sacrificar la capacidad de aprendizaje.

Ventajas del diseño de *DenseNet201* *DenseNet201* ofrece varias ventajas en comparación con arquitecturas de redes convencionales:

- **Reutilización de características aprendidas**: las conexiones densas permiten que cada capa acceda a las características de todas las capas previas, facilitando la reutilización de información y promoviendo un aprendizaje más eficiente.
- **Optimización del flujo de gradientes**: la estructura densa facilita la retropropagación de gradientes hacia capas anteriores, optimizando el aprendizaje y mitigando problemas de pérdida de gradiente en redes profundas.
- **Reducción de parámetros**: debido a las conexiones densas y al control de crecimiento, *DenseNet201* requiere menos parámetros que modelos equivalentes, reduciendo así el riesgo de sobreajuste y optimizando el uso de recursos computacionales.

DenseNet201 se destaca en tareas de clasificación de imágenes y otros problemas de visión por computadora, donde la preservación de detalles en las características es crucial. Su diseño permite

captar tanto características de bajo nivel en las primeras capas como patrones más complejos en las capas posteriores, logrando un balance entre profundidad y eficiencia [202].

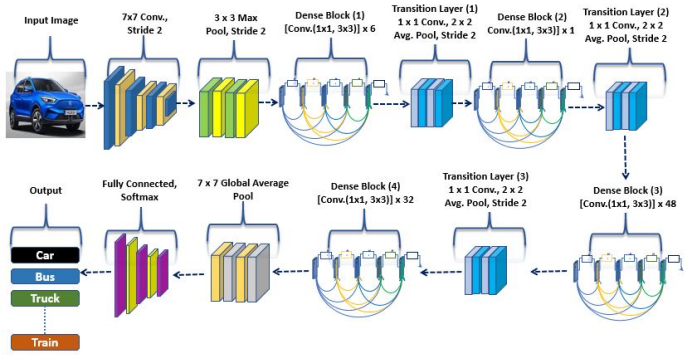


Figura 4.16. Arquitectura del modelo *DenseNet201*.

VGG19

VGG19 es una red neuronal convolucional desarrollada por Karen Simonyan y Andrew Zisserman en el Visual *Geometry Group* de la Universidad de *Oxford*. Esta arquitectura se caracteriza por su simplicidad y profundidad, utilizando un diseño modular y profundo que ha demostrado ser altamente efectivo en la clasificación de imágenes de gran escala [201]. *VGG19* emplea una secuencia de capas convolucionales y de *pooling* organizadas en bloques, lo que facilita el aprendizaje de características visuales detalladas.

Aspectos técnicos y arquitectura de *VGG19* *VGG19* es una variante de la familia *VGG*, con 19 capas de aprendizaje organizadas en una estructura simple pero poderosa. Los elementos clave de su diseño incluyen:

- **Capas convolucionales de 3x3**: uno de los elementos distintivos de *VGG19* es el uso de convoluciones de tamaño reducido (3x3) repetidas varias veces en cada bloque. Este tamaño de filtro permite capturar características finas y detalladas en las imágenes, y al repetirlas en profundidad, *VGG19* puede captar patrones de mayor complejidad. A diferencia de los filtros más grandes, el tamaño 3x3 optimiza el uso de recursos computacionales al mismo tiempo que incrementa la profundidad de la red.
- **Bloques de convolución y *pooling***: la arquitectura de *VGG19* se organiza en bloques, donde se agrupan dos o tres capas convolucionales seguidas de una capa de *max pooling*. Este diseño reduce las dimensiones espaciales progresivamente y aumenta el número de canales, permitiendo que el modelo aprenda tanto características locales de bajo nivel como representaciones abstractas de alto nivel.
- **Capas completamente conectadas al final de la red**: en las etapas finales, *VGG19* emplea tres capas totalmente conectadas, seguidas de una función de activación *ReLU* y una capa de *softmax* para la salida de clasificación. Estas capas completamente conectadas consolidan las características aprendidas en una representación final, asignando una probabilidad a cada clase objetivo.

Ventajas del diseño de *VGG19* *VGG19* es conocida por su estructura modular y su capacidad para captar patrones detallados en imágenes, lo que le da ventajas importantes en clasificación de imágenes:

- **Profundidad y simplicidad**: la estructura de *VGG19*, aunque más profunda que las redes anteriores, mantiene un diseño sencillo que facilita su implementación y ajuste.
- **Extracción de características detalladas**: la repetición de filtros 3x3 permite capturar patrones complejos en imágenes, ideal para aplicaciones donde la precisión en detalles es crítica.
- **Versatilidad en transferencia de aprendizaje**: gracias a su arquitectura profunda y su capacidad para aprender representaciones detalladas, *VGG19* es frecuentemente empleada en tareas de *transfer learning*, ya que sus capas convolucionales pueden adaptarse eficazmente a nuevos contextos de clasificación.

VGG19 ha demostrado ser particularmente útil en aplicaciones de clasificación y detección de objetos, proporcionando una estructura robusta para tareas de visión por computadora que requieren una extracción detallada de características [201].

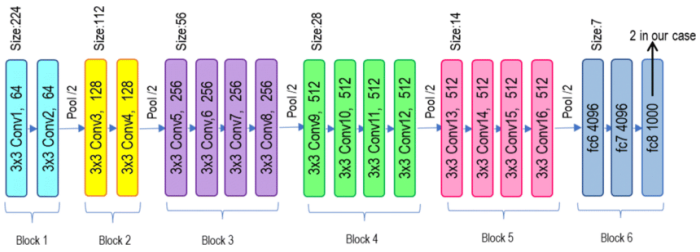


Figura 4.17. Arquitectura del modelo *VGG19*.

Xception

Xception, introducido por François Chollet, es una arquitectura de red neuronal convolucional avanzada que mejora los principios de la arquitectura Inception utilizando convoluciones separables en profundidad. *Xception* es una abreviatura de "*Extreme Inception*" y se basa en la hipótesis de que la correlación espacial y la correlación entre canales de una imagen pueden modelarse de manera más efectiva mediante convoluciones separadas. Este enfoque permite que el modelo sea más eficiente en el uso de parámetros sin comprometer la capacidad de aprendizaje [198].

Aspectos técnicos y arquitectura de *Xception* *Xception* emplea un diseño basado en convoluciones separables en profundidad y bloques residuales que ayudan a optimizar el flujo de información en redes profundas. Los aspectos clave de su arquitectura incluyen:

- **Convoluciones separables en profundidad**: la arquitectura *Xception* sustituye las convoluciones estándar por convoluciones separables en profundidad en todos sus bloques. Una convolución separable en profundidad consiste en dos pasos: una convolución *depthwise*, que aplica un filtro específico a cada canal de entrada, y una convolución *pointwise* de 1x1, que combina los canales. Esta separación permite reducir significativamente el número de parámetros y operaciones, logrando una mayor eficiencia computacional.
- **Bloques residuales y conexiones directas**: en *Xception*, los bloques de convolución están organizados en bloques residuales, con conexiones directas que permiten un flujo eficiente

de gradientes durante el entrenamiento. Cada bloque residual contiene múltiples capas de convoluciones separables en profundidad, y la conexión directa ayuda a mitigar problemas de desaparición de gradiente en redes profundas.

- **Separación completa de correlaciones espaciales y de canal**: a diferencia de modelos previos como *Inception*, donde las convoluciones combinan correlaciones espaciales y de canal en una única operación, *Xception* maneja estos dos tipos de correlaciones por separado. Este enfoque mejora la capacidad del modelo para aprender patrones complejos sin aumentar el costo computacional. La convolución *depthwise* se encarga de captar las correlaciones espaciales dentro de cada canal, mientras que la convolución *pointwise* permite la interacción entre canales.

Ventajas de la arquitectura *Xception* *Xception* presenta varias ventajas que la hacen particularmente eficiente para tareas de clasificación y reconocimiento de imágenes:

- **Reducción en el número de parámetros**: gracias al uso de convoluciones separables en profundidad, *Xception* logra una reducción significativa en el número de parámetros, lo cual facilita su entrenamiento en entornos de recursos limitados.
- **Eficiencia en el flujo de gradientes**: la inclusión de bloques residuales permite un flujo constante de gradientes, evitando la desaparición de gradiente y mejorando la capacidad de aprendizaje en redes profundas.
- **Flexibilidad en transferencia de aprendizaje**: al separar correlaciones espaciales y de canal, *Xception* es capaz de aprender características detalladas que pueden adaptarse eficientemente a nuevas tareas mediante *transfer learning*.

Aplicaciones y rendimiento *Xception* ha mostrado un rendimiento competitivo en varias tareas de visión por computadora, desde clasificación de imágenes hasta detección de objetos. Su estructura modular, basada en convoluciones separables y bloques residuales, permite un balance entre profundidad y eficiencia computacional, haciéndolo adecuado para implementaciones en las que se requiere alta precisión sin un costo computacional elevado [198]. Al mejorar el diseño de *Inception*, *Xception* logra una precisión comparable o superior en varios conjuntos de datos de referencia con un uso de recursos más eficiente.

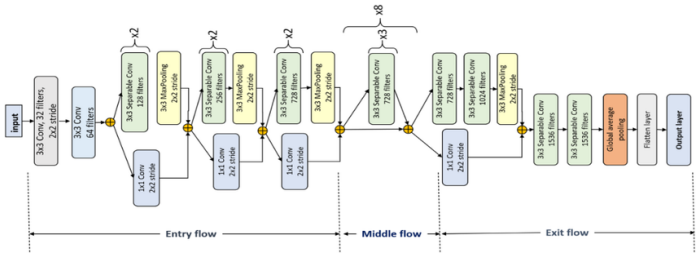


Figura 4.18. Arquitectura del modelo *Xception*.

Evaluación y métricas de desempeño

Las métricas de desempeño son fundamentales para evaluar la efectividad de los modelos de *Deep Learning*. Entre las más comunes se incluyen:

- **Precisión (*accuracy*):** proporción de predicciones correctas sobre el total.

- **Exactitud (*precision*) y exhaustividad (*recall*):** miden la capacidad del modelo para identificar correctamente las clases relevantes.
- ***F1 score*:** media armónica de la precisión y el *recall*, útil en escenarios con clases desbalanceadas:

$$F1 = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall}$$

4.4.10. *Explainable AI (XAI)*

En los últimos años, el desarrollo y la implementación de sistemas de inteligencia artificial (IA) han crecido exponencialmente, abarcando múltiples sectores como la medicina, la finanza, y la gestión de residuos. Sin embargo, la complejidad de estos modelos, en especial los basados en redes neuronales profundas, ha generado una preocupación importante sobre su falta de interpretabilidad. Es en este contexto donde surge el campo de *Explainable AI (XAI)*.

Definición y objetivo de *Explainable AI*

Explainable AI (XAI) se refiere a un conjunto de métodos y técnicas que permiten a los humanos entender y confiar en las decisiones o predicciones realizadas por los sistemas de IA. La principal motivación detrás de XAI es mitigar el efecto de las "cajas negras" en los modelos de aprendizaje automático, proporcionando explicaciones claras y comprensibles sobre cómo y por qué un modelo ha tomado una decisión específica. Según el estudio de Adadi y Berrada (2018), XAI no solo mejora la interpretabilidad sino que también incrementa la transparencia y la confianza en los sistemas de IA [203].

Importancia de la interpretabilidad

La necesidad de interpretabilidad es particularmente crítica en aplicaciones sensibles, como diagnósticos médicos, sistemas de crédito, y control de vehículos autónomos. Un modelo interpretable permite:

- Detectar posibles sesgos en el modelo.
- Entender el proceso de toma de decisiones del modelo.
- Mejorar la confianza de los usuarios y desarrolladores en la IA.
- Facilitar la depuración y mejora continua del sistema.

Métodos de *Explainable AI*

Los métodos de XAI se dividen en dos categorías principales: *Métodos Post-Hoc* y *Métodos Intrínsecos*. A continuación, se describen algunos de los métodos más populares en cada categoría.

Métodos intrínsecos

Estos métodos incorporan interpretabilidad dentro de la estructura del modelo desde el inicio. Algunos ejemplos incluyen:

- **Modelos lineales y arboles de decisión**: estos son modelos que, por su naturaleza, son fácilmente interpretables. Por ejemplo, un árbol de decisión proporciona reglas claras y explícitas sobre cómo se hacen las predicciones.
- **Modelos de regresión**: la regresión lineal, por ejemplo, es un modelo simple donde los coeficientes pueden interpretarse como la influencia de cada variable de entrada en la predicción.
- **Redes neuronales interpretables**: estas son arquitecturas diseñadas para ser más comprensibles, como las redes neuronales sumativas o modelos simplificados.

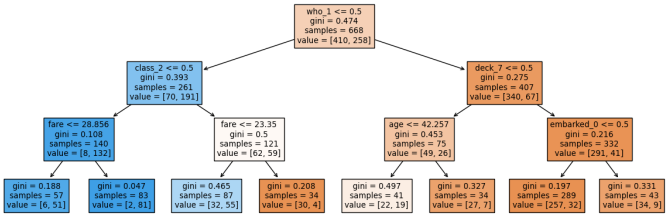


Figura 4.19. Ejemplo de un Árbol de Decisión utilizado para la Interpretabilidad.

Métodos post-hoc

Los métodos *post-hoc* se aplican a modelos complejos ya entrenados para proporcionar interpretabilidad. Algunos de los métodos más reconocidos son:

- ***Grad-CAM (Gradient-weighted Class Activation Mapping)***: *Grad-CAM* es un método visual que utiliza los gradientes de las predicciones a lo largo de las capas de convolución para identificar las regiones de una imagen que contribuyen a la clasificación. Esto se usa comúnmente en modelos de visión por computadora [204].

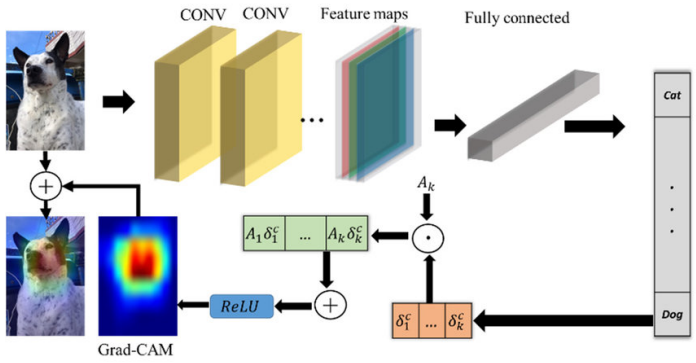


Figura 4.20. Ejemplo de *Grad-CAM* mostrando las áreas de atención del modelo.

- ***LIME (Local Interpretable Model-agnostic Explanations)***: *LIME* funciona creando un modelo interpretable, como una regresión lineal, que aproxima el comportamiento del modelo original en una vecindad local alrededor de una predicción específica. Esto ayuda a entender cómo las variables de entrada afectan una predicción particular [205].

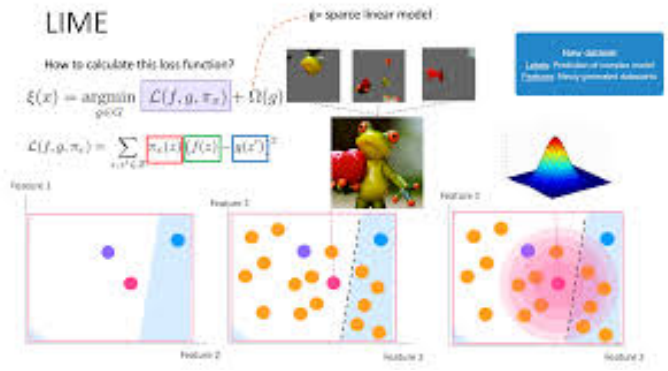


Figura 4.21. Ejemplo de *LIME* explicando las decisiones de un modelo de clasificación.

- ***SHAP (SHapley Additive exPlanations)***: basado en la teoría de juegos, *SHAP* calcula la contribución de cada característica a la predicción final de un modelo, asignando un valor de *Shapley* a cada variable. Este método es ampliamente usado debido a su capacidad para proporcionar explicaciones consistentes y justas [206].

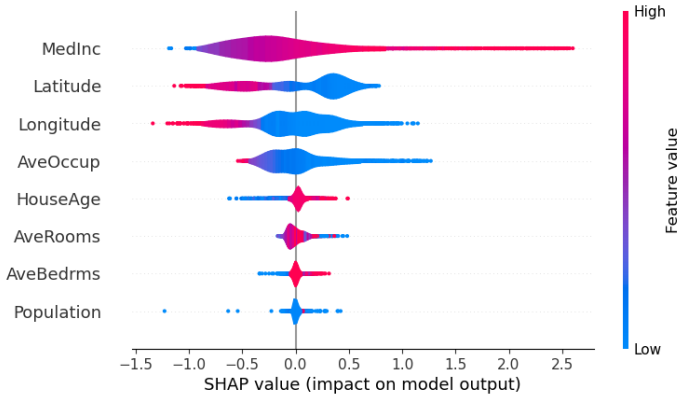


Figura 4.22. Ejemplo de *SHAP* mostrando la contribución de cada variable en una predicción.

- ***SmoothGrad***: este método suaviza los mapas de saliencia generando múltiples versiones de una imagen con ruido y promediando los resultados. De esta manera, se reduce el ruido y se obtiene una visualización más clara de las áreas de importancia [207].

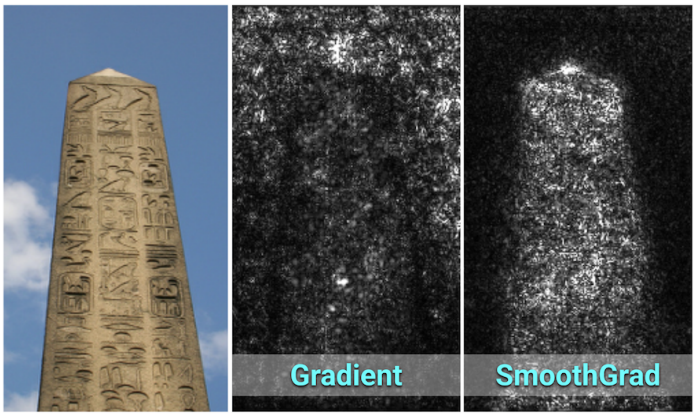


Figura 4.23. Ejemplo de *SmoothGrad* reduciendo el ruido en los mapas de saliencia.

- **Interpretación basada en contrafactuales**: esta técnica se centra en generar ejemplos que muestren cómo los cambios en las variables de entrada afectarían el resultado de una predicción. Esto es útil para entender qué factores son críticos para una decisión dada.

Desafíos en *Explainable AI*

Aunque XAI ha avanzado significativamente, todavía enfrenta varios desafíos. Uno de los principales es el equilibrio entre interpretabilidad y precisión: a menudo, los modelos más interpretables, como los árboles de decisión, no son tan precisos como las redes neuronales profundas. Además, la subjetividad en lo que constituye una "buena" explicación varía según el usuario y el contexto [208].

Otro problema es la escalabilidad, ya que algunas técnicas de XAI, como *LIME*, pueden ser computacionalmente costosas. Además, los sesgos inherentes a los datos de entrenamiento pueden ser difíciles de detectar, incluso con modelos explicables.

Aplicaciones de *Explainable AI*

XAI se está aplicando en numerosos campos, como:

- **Medicina**: para justificar las decisiones de diagnóstico y tratamiento.
- **Finanzas**: en la explicación de aprobaciones o rechazos de crédito.
- **Seguridad**: en la interpretación de decisiones tomadas por sistemas de vigilancia y análisis de comportamiento.

Estas aplicaciones son fundamentales para garantizar que los sistemas de IA sean transparentes, éticos y responsables.

4.4.11. Modelos de *machine learning* e *inteligencia artificial* en la gestión de residuos

Diversos modelos de *Machine Learning* e *Inteligencia Artificial* han sido desarrollados para abordar los desafíos en la gestión de residuos, desde la clasificación automática hasta la optimización de la recolección y reciclaje. A continuación, se describen algunos de los casos más destacados.

RecycleNet: clasificación automática de residuos

RecycleNet es un modelo basado en redes neuronales convolucionales (CNN) diseñado para la clasificación automática de residuos en tiempo real. Este modelo utiliza imágenes capturadas en cintas transportadoras para identificar y separar materiales reciclables como papel, plástico, metal y vidrio. Su implementación ha permitido mejorar significativamente la eficiencia en plantas de reciclaje, reduciendo la necesidad de intervención humana [209].

Sistema de gestión inteligente de residuos de *SAS*

Un modelo analítico creado por Agustín Marín, de la empresa *SAS*, aplica técnicas avanzadas de *Machine Learning* para optimizar la recolección de residuos en entornos urbanos. Este sistema

predice la generación de residuos en diferentes áreas, lo que permite planificar las rutas de recolección de manera más eficiente, reduciendo costos y emisiones de carbono [210].

Project recicla AI

Este proyecto, desarrollado por *DoradoReciclaAI*, utiliza dos fuentes de datos: **TrashNet/* y **Waste Classification Data**, para entrenar un modelo robusto que mejora la precisión en la clasificación de residuos. El modelo se enfoca en la identificación de materiales difíciles de clasificar, como plásticos degradables y residuos electrónicos, y se integra en aplicaciones móviles para facilitar el reciclaje en comunidades locales [211].

Aplicación web para la clasificación de residuos

Desarrollada como parte de un trabajo de investigación en la Universidad EAN, esta aplicación web utiliza un modelo de *Machine Learning* para clasificar residuos sólidos de manera automática. El modelo está diseñado para ayudar a los usuarios a identificar correctamente los residuos que pueden reciclarse, contribuyendo a una mejor gestión desde el origen [209].

WasteNet: modelo de clasificación de residuos en *smart cities*

WasteNet es un modelo diseñado para clasificar residuos en entornos urbanos inteligentes (*Smart Cities*). Utiliza una combinación de sensores IoT y redes neuronales convolucionales (CNN) para identificar y clasificar residuos en tiempo real, optimizando así el proceso de reciclaje y reduciendo la cantidad de residuos enviados a vertederos. *WasteNet* ha sido implementado en varias ciudades como parte de iniciativas para mejorar la sostenibilidad urbana [212].

EcoSort: optimización de rutas de recolección de residuos

EcoSort es un sistema basado en *Machine Learning* que optimiza las rutas de recolección de residuos utilizando modelos predictivos. Este sistema analiza datos históricos y en tiempo real sobre la generación de residuos para planificar rutas más eficientes, reduciendo así el consumo de combustible y las emisiones de CO_2. *EcoSort* ha sido implementado en ciudades de Europa como parte de programas de gestión de residuos sostenibles [213].

AI-Waste: clasificación avanzada de residuos electrónicos

AI-Waste es un modelo que utiliza técnicas de *Deep Learning* para clasificar componentes de residuos electrónicos (*e-waste*). Este modelo ha sido desarrollado para identificar y separar metales valiosos como el oro y el cobre en dispositivos electrónicos desechados, facilitando el reciclaje de materiales escasos y la reducción de la minería extractiva [214].

Consideraciones finales

La implementación efectiva de ML e IA en la gestión de residuos requiere una comprensión sólida de los modelos, hiperparámetros y métricas de desempeño. Además, la capacidad de ajustar y optimizar estos elementos es clave para desarrollar sistemas robustos y precisos que contribuyan a la sostenibilidad y eficiencia en la gestión de residuos.

4.4.12. Visión por computadora para clasificación de imágenes

La visión por computadora es una rama de la inteligencia artificial que permite a las máquinas interpretar y comprender el mundo visual a partir de imágenes o videos. En el contexto de la gestión de residuos, la visión por computadora es clave para desarrollar clasificadores automáticos que puedan identificar y categorizar diferentes tipos de residuos a partir de imágenes.

Conceptos clave en visión por computadora

Los conceptos fundamentales en visión por computadora incluyen el preprocesamiento de imágenes, la extracción de características y la clasificación. A continuación, se describen algunos de los aspectos más importantes:

Preprocesamiento de imágenes

El preprocesamiento de imágenes es una etapa crítica en la cual las imágenes se ajustan y preparan para ser utilizadas por los modelos de *Machine Learning* o *Deep Learning*. Las técnicas comunes incluyen:

- **Redimensionamiento:** ajuste del tamaño de la imagen a las dimensiones requeridas por el modelo. Generalmente, las imágenes se redimensionan para que tengan el mismo tamaño, lo cual simplifica el procesamiento.
- **Normalización:** escala los valores de los píxeles para que estén dentro de un rango estándar, típicamente entre 0 y 1, o entre -1 y 1. Esto ayuda a estabilizar el entrenamiento del modelo [215].
- **Aumento de datos (*data augmentation*):** técnicas que generan nuevas imágenes a partir de las existentes mediante transformaciones como rotación, traslación, reflejo y ajuste de brillo. Esto aumenta la diversidad del conjunto de datos y reduce el sobreajuste [216].

Extracción de características

La extracción de características consiste en identificar y extraer información relevante de las imágenes que permita al modelo distinguir entre diferentes clases de residuos. En las redes neuronales convolucionales (CNN), esto se realiza automáticamente mediante capas convolucionales que aplican filtros sobre la imagen para detectar bordes, texturas y patrones.

Operación convolucional

Matemáticamente, la operación convolucional en una imagen se define como:

$$S(i,j) = (I * K)(i,j) = \sum_m \sum_n I(i-m, j-n) \cdot K(m,n)$$

Donde:

- I es la matriz de la imagen de entrada.

- K es el filtro (*kernel*) aplicado.
- $S(i, j)$ es la salida de la convolución en la posición (i, j).

Clasificación de imágenes

La clasificación de imágenes implica asignar una etiqueta o categoría a una imagen dada. Los modelos de *Deep Learning*, como las CNN, han demostrado ser altamente eficaces para esta tarea. El proceso de clasificación generalmente incluye:

- **Redes completamente conectadas:** al final de las capas convolucionales, la imagen es aplanada y pasa a través de una o más capas completamente conectadas que realizan la clasificación final [217].
- **Función de activación *Softmax*:** la última capa suele utilizar la función *Softmax* para convertir los valores de salida en probabilidades:

$$Softmax(z_i) = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}}$$

  Donde K es el número de clases [218].

Relación con *machine learning* e *inteligencia artificial*

La visión por computadora es un campo interdisciplinario que se apoya en métodos de *Machine Learning* e *Inteligencia Artificial* para resolver problemas complejos. En el contexto de la clasificación de residuos, las CNN son la elección preferida debido a su capacidad para aprender representaciones jerárquicas de las imágenes [219].

Optimización y entrenamiento de modelos de visión por computadora

El entrenamiento de modelos de visión por computadora implica minimizar una función de pérdida que mide la discrepancia entre las predicciones del modelo y las etiquetas verdaderas. El método de optimización más común es el gradiente descendente estocástico (SGD), con variantes como el algoritmo *Adam*.

Función de pérdida categórica

Para tareas de clasificación, la función de pérdida categórica entropía cruzada se utiliza ampliamente:

$$L(y, \hat{y}) = -\sum_{i=1}^{K} y_i \log(\hat{y}_i)$$

Donde:

- y_i es el vector de etiquetas verdaderas.
- $\hat{y}_i$ es la predicción del modelo.

Visualización de características y mapas de calor

Una herramienta útil para entender cómo un modelo de visión por computadora toma decisiones es la visualización de mapas de calor. Los mapas de activación visualizan las áreas de la imagen que más influyen en la decisión del modelo, proporcionando información valiosa sobre el proceso de clasificación.

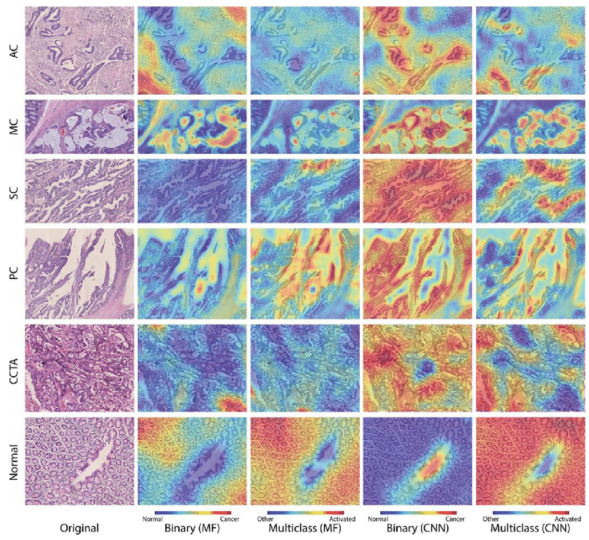


Figura 4.24. Ejemplo de un mapa de calor que muestra las áreas de una imagen que más contribuyen a la clasificación.

Consideraciones finales

La visión por computadora, en combinación con técnicas de *Machine Learning* e *Inteligencia Artificial*, proporciona una base poderosa para la automatización en la clasificación de residuos. La implementación adecuada de preprocesamiento, extracción de características y optimización del modelo es esencial para lograr un sistema preciso y robusto.

4.4.13. Modelos de lenguaje de gran escala (*large language models, LLM*) y *chatbots* inteligentes

Los *Large Language Models* (LLM) son modelos de inteligencia artificial diseñados para procesar y generar lenguaje natural. Estos modelos han alcanzado un rendimiento sobresaliente en tareas de procesamiento de lenguaje natural (NLP), como traducción automática, generación de texto, resumen, y respuestas a preguntas. Los LLM han revolucionado la forma en que se desarrollan los *chatbots*, permitiendo interacciones más naturales y precisas con los usuarios.

Conceptos clave en LLM

Los LLM se basan en arquitecturas de redes neuronales profundas, principalmente las transformadores (*Transformers*), que son capaces de manejar secuencias largas de texto de manera eficiente.

Transformers

El transformador es la arquitectura base detrás de los LLM modernos. Se compone de un mecanismo de atención, que permite al modelo enfocarse en diferentes partes del input dependiendo del contexto. La ecuación del mecanismo de atención es:

$$Attention(Q, K, V) = softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

Donde:

- Q es la matriz de consultas (*queries*).
- K es la matriz de claves (*keys*).
- V es la matriz de valores (*values*).
- d_k es la dimensión de las claves.

Implementación de *chatbots* inteligentes con LLM

Los *chatbots* basados en LLM como GPT-3 o BERT pueden comprender y generar respuestas coherentes en un amplio rango de contextos. La capacidad de estos modelos para captar matices del lenguaje humano les permite interactuar de manera más efectiva con los usuarios, brindando asistencia personalizada y relevante en tiempo real.

Función de costo en LLM

Durante el entrenamiento, los LLM se optimizan para minimizar una función de costo que mide la discrepancia entre las predicciones del modelo y las palabras reales en la secuencia de entrenamiento. Generalmente, la función de costo utilizada es la entropía cruzada:

$$L = -\sum_{t=1}^{T} \log P(y_t | y_1, \ldots, y_{t-1}, \theta)$$

Donde:

- y_t es la palabra en el tiempo t.
- θ son los parámetros del modelo.

Ejemplo de arquitectura de un *chatbot* inteligente

Un *chatbot* inteligente basado en LLM generalmente incluye las siguientes etapas:

- **Preprocesamiento del texto:** limpieza y tokenización del texto de entrada.
- **Generación de respuesta:** el LLM genera una respuesta basada en el contexto y el historial de la conversación.
- **Postprocesamiento:** ajustes finales a la respuesta para mejorar la naturalidad.

Ejemplos de LLM poderosos para *chatbots*

- **GPT-3:** un modelo con 175 mil millones de parámetros, desarrollado por *OpenAI*, utilizado en aplicaciones de *chatbots*, asistentes virtuales, y generación de contenido.
- **BERT:** modelo de *Google* basado en transformadores bidireccionales, optimizado para tareas como la clasificación de texto, análisis de sentimiento, y respuesta a preguntas.
- **T5:** un modelo de *Google* que aborda todas las tareas de NLP como problemas de traducción, logrando resultados impresionantes en diversas aplicaciones.

Implicaciones y consideraciones éticas en el uso de LLM

Dado el poder de los LLM, existen preocupaciones éticas relacionadas con su uso, como la generación de desinformación, sesgo en los resultados, y la privacidad de los datos utilizados para entrenarlos. Es crucial implementar salvaguardas y políticas para mitigar estos riesgos [220], [221].

Visualización de la atención en LLM

La visualización de la atención en modelos como *BERT* permite entender qué palabras o frases son más relevantes para una tarea específica. Esta visualización puede proporcionar información valiosa sobre cómo los modelos toman decisiones.

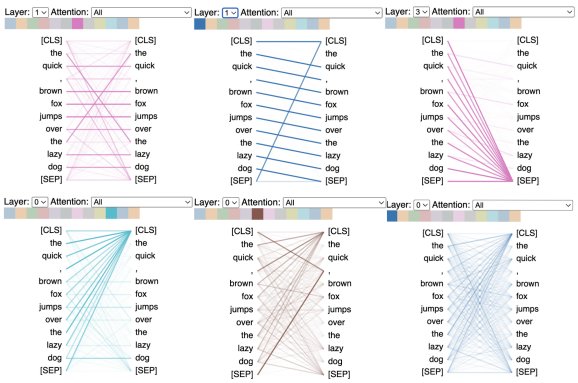


Figura 4.25. Ejemplo de un mapa de atención en un LLM, mostrando la relevancia de diferentes palabras en un contexto específico.

4.4.14. Tecnologías para la implementación de modelos de *inteligencia artificial*

El desarrollo e implementación de modelos de *Machine Learning* e *Inteligencia Artificial* requieren el uso de herramientas y plataformas avanzadas que faciliten la programación, entrenamiento, y despliegue de dichos modelos. En esta sección, se abordarán las principales tecnologías utilizadas en este campo.

Python

Python es el lenguaje de programación por excelencia para *Machine Learning* e *Inteligencia Artificial*. Su sintaxis simple y su vasta colección de librerías especializadas lo hacen ideal para el

desarrollo de modelos. Librerías como *NumPy, Pandas*, y *Scikit-learn* son fundamentales para tareas de manipulación de datos, estadística, y aprendizaje automático [220].

Kaggle

Kaggle es una plataforma que permite a los investigadores y desarrolladores compartir *datasets*, colaborar en proyectos, y competir en desafíos de *Machine Learning*. Además, *Kaggle* también es un repositorio importante de *datasets*, ofreciendo una vasta colección de conjuntos de datos públicos que pueden ser utilizados para entrenar y probar modelos [222].

Jupyter Notebooks

Jupyter Notebooks es una herramienta de código abierto que permite combinar código, texto, y visualizaciones en un solo documento. Es ampliamente utilizado en el desarrollo de modelos debido a su flexibilidad y capacidad para facilitar la experimentación rápida [223].

Google Colab Pro

Google Colab Pro es una versión mejorada de *Google Colab* que ofrece acceso a recursos adicionales como *GPU* y *TPU*, tiempos de ejecución más largos, y mayores límites de uso. Es una herramienta esencial para el entrenamiento de modelos de *Deep Learning* de gran escala.

Recursos y características de *Google Colab Pro*

Google Colab Pro proporciona acceso a *GPU* de alta gama como *Tesla* P100 y V100, lo que permite entrenar modelos complejos de manera eficiente. También ofrece integraciones directas con bibliotecas como *TensorFlow, PyTorch, y Hugging Face*, facilitando la implementación de modelos avanzados [224].

Integraciones y facilidad de uso

Colab se integra con *Google Drive*, permitiendo a los usuarios almacenar y acceder fácilmente a *datasets*, modelos, y resultados. Además, su compatibilidad con bibliotecas de *Python* como *TensorFlow* y *PyTorch* lo convierte en una plataforma ideal para el desarrollo de proyectos de IA [225].

Librerías de *Python*

Python cuenta con una gran variedad de librerías especializadas en *Machine Learning* y *Deep Learning*, entre las cuales destacan:

- ***TensorFlow*:** desarrollada por *Google, TensorFlow* es una de las bibliotecas más populares para *Deep Learning*. Proporciona herramientas para construir y entrenar redes neuronales profundas y modelos de ML a gran escala [226].
- ***PyTorch*:** creada por *Facebook, PyTorch* es otra biblioteca popular para *Deep Learning*, conocida por su flexibilidad y facilidad de uso. *PyTorch* es ampliamente utilizado en investigación y producción [227].

- ***Hugging Face*:** una plataforma que ofrece herramientas para la implementación de modelos de procesamiento de lenguaje natural (NLP), como BERT y GPT-3. *Hugging Face* facilita el uso de LLM preentrenados y su ajuste fino [225].
- ***OpenCV*:** una librería de visión por computadora fundamental para el procesamiento y análisis de imágenes. Se utiliza para tareas como la detección de objetos, la transformación de imágenes y la extracción de características visuales [228].
- ***Keras*:** una *API* de alto nivel construida sobre *TensorFlow*, que facilita la creación de modelos de *Deep Learning* mediante una interfaz más sencilla y amigable [229].
- ***Scikit-Image*:** librería basada en *NumPy* para el procesamiento de imágenes, útil para tareas como la segmentación, la clasificación, y la manipulación de imágenes [230].
- ***Pillow*:** derivada de la biblioteca *PIL (Python Imaging Library), Pillow* es una librería para el procesamiento y manipulación de imágenes en *Python*. Permite realizar una amplia gama de operaciones de edición de imágenes, como el cambio de tamaño, el recorte, la rotación y la aplicación de filtros. Además, es compatible con numerosos formatos de imagen, como *JPEG, PNG y GIF. Pillow* resulta útil en tareas de preprocesamiento de imágenes para aplicaciones de *Machine Learning* y *Deep Learning*, y permite realizar transformaciones geométricas y manipulaciones a nivel de píxel, que son esenciales para el aumento de datos (*data augmentation*) y la preparación de imágenes de entrada [231].

CUDA y el uso de tarjetas gráficas

CUDA es una plataforma de computación paralela desarrollada por *NVIDIA* que permite a los desarrolladores utilizar *GPU* para acelerar aplicaciones de cómputo intensivo como el entrenamiento de modelos de *Deep Learning*. El uso de *GPU* es fundamental para reducir los tiempos de entrenamiento y manejar modelos con millones de parámetros.

Optimización con *CUDA*

Las tarjetas gráficas como *NVIDIA Tesla, Quadro* y *RTX* son compatibles con *CUDA*, lo que permite ejecutar cálculos en paralelo de manera más rápida que con *CPUs* tradicionales. La optimización del entrenamiento con *CUDA* es un factor clave para la eficiencia en proyectos de IA a gran escala [232].

Uso de *datasets* para entrenamiento

El uso de *datasets* de alta calidad es crucial para entrenar modelos de *Machine Learning* efectivos. Los *datasets* deben estar bien etiquetados, ser diversos y representar adecuadamente las clases de interés.

Características de un buen *dataset*

Un buen *dataset* debe cumplir con las siguientes características:

- **Relevancia:** los datos deben estar directamente relacionados con el problema que se busca resolver.
- **Calidad:** los datos deben estar limpios, sin valores atípicos o inconsistencias significativas.

- **Diversidad:** el *dataset* debe cubrir un amplio rango de variaciones posibles para evitar sesgos en el modelo.
- **Tamaño:** un mayor número de ejemplos ayuda al modelo a generalizar mejor.

Ejemplos de *datasets* para clasificación de residuos

Existen varios *datasets* que se han utilizado para la clasificación de residuos:

- ***TrashNet*:** un conjunto de datos compuesto por imágenes de diferentes tipos de residuos como vidrio, papel, metal, plástico y basura orgánica [233].
- ***Waste Classification Data*:** *dataset* que contiene imágenes de residuos clasificadas en varias categorías, útil para entrenar modelos de clasificación automática [234].
- ***TACO*:** *dataset* de imágenes para la clasificación y detección de residuos, utilizado en competiciones de *Kaggle* [235].

Relevancia de *Kaggle* como repositorio de *datasets*

Kaggle no solo es una plataforma de competencias, sino también uno de los repositorios más grandes de *datasets* públicos. Los usuarios pueden acceder a una amplia gama de *datasets* de alta calidad para tareas como la clasificación de imágenes, el análisis de texto, y más [222].

4.4.15. *TensorFlow* y visión por computadora con redes neuronales convolucionales (CNN)

TensorFlow es una de las plataformas más utilizadas para el desarrollo de modelos de *Deep Learning*, especialmente en el ámbito de la visión por computadora. Proporciona herramientas robustas para construir y entrenar redes neuronales convolucionales (CNN), que son el estándar de facto para tareas de clasificación de imágenes, detección de objetos, y segmentación.

Redes neuronales convolucionales (CNN)

Las redes neuronales convolucionales (CNN) son un tipo de red neuronal especialmente diseñada para procesar datos con una estructura de cuadrícula, como imágenes. Se componen de capas convolucionales que aplican filtros sobre la entrada para extraer características relevantes, seguidas de capas de agrupación (*pooling*) que reducen la dimensionalidad.

Estructura de una CNN

Una CNN típica consta de:

- **Capas convolucionales:** aplican convoluciones sobre la entrada para detectar características como bordes y texturas. Matemáticamente, la operación convolucional se define como:

$$S(i,j) = (I * K)(i,j) = \sum_m \sum_n I(i-m, j-n) \cdot K(m,n)$$

donde I es la matriz de la imagen y K es el *kernel* o filtro aplicado.

- **Capas de agrupación (*pooling*):** reducen la dimensionalidad de la salida de las capas convolucionales, conservando la información más relevante.
- **Capas completamente conectadas:** al final de la red, estas capas realizan la clasificación en función de las características extraídas.

Implementación de CNN en *TensorFlow*

TensorFlow facilita la construcción e implementación de CNN mediante su *API* de alto nivel, *Keras*. A continuación se presenta un ejemplo básico de cómo implementar una CNN para la clasificación de imágenes:

```
import tensorflow as tf
from tensorflow.keras import layers, models

# Definición del modelo
model = models.Sequential([
    layers.Conv2D(32, (3, 3), activation='relu', input_shape=(64, 64, 3)),
    layers.MaxPooling2D((2, 2)),
    layers.Conv2D(64, (3, 3), activation='relu'),
    layers.MaxPooling2D((2, 2)),
    layers.Conv2D(64, (3, 3), activation='relu'),
    layers.Flatten(),
    layers.Dense(64, activation='relu'),
    layers.Dense(10, activation='softmax')
])

# Compilación del modelo
model.compile(optimizer='adam',
              loss='sparse_categorical_crossentropy',
              metrics=['accuracy'])

# Entrenamiento del modelo
model.fit(train_images, train_labels, epochs=10, validation_data=(test_images, test_labels))
```

Ventajas de usar *TensorFlow* en visión por computadora

TensorFlow ofrece varias ventajas para la implementación de modelos de visión por computadora:

- **Escalabilidad:** *TensorFlow* permite entrenar modelos en múltiples *GPU* y *TPU*, lo que es crucial para manejar grandes volúmenes de datos.
- **Compatibilidad con *Keras*:** la integración de *Keras* facilita el desarrollo rápido de modelos, con una sintaxis clara y simplificada.
- **Biblioteca de modelos preentrenados:** *TensorFlow Hub* proporciona acceso a modelos preentrenados que pueden ser ajustados para tareas específicas de visión por computadora.

Casos de uso en visión por computadora

TensorFlow ha sido utilizado en numerosos proyectos de visión por computadora, que incluyen:

- **Clasificación de residuos:** implementaciones de CNN entrenadas para clasificar imágenes de residuos en diferentes categorías (vidrio, papel, plástico, etc.).
- **Detección de objetos:** utilización de modelos como *YOLO* y *SSD* para detectar y localizar objetos en imágenes en tiempo real.
- **Segmentación semántica:** aplicación de redes como *U-Net* para segmentar imágenes en diferentes clases, útil en tareas médicas y de medio ambiente.

Optimización y despliegue de modelos en producción

TensorFlow también facilita la optimización y despliegue de modelos en producción a través de *TensorFlow Lite* y *TensorFlow Serving*, permitiendo que los modelos se ejecuten en dispositivos móviles y servidores con baja latencia.

Conclusiones

La combinación de *TensorFlow* y CNN proporciona una plataforma poderosa y flexible para resolver problemas complejos de visión por computadora. La capacidad de escalar, junto con la facilidad de uso y la extensa comunidad de usuarios, hacen de *TensorFlow* una opción ideal para proyectos de clasificación de imágenes y más.

4.4.16. *Hugging Face* y su rol en la construcción de *chatbots* inteligentes

Hugging Face es una plataforma y comunidad líder en el ámbito del procesamiento del lenguaje natural (NLP). Ofrece una amplia colección de modelos preentrenados, incluyendo *Large Language Models* (LLM) como BERT, GPT-3, y T5, que son fundamentales para la construcción de *chatbots* inteligentes. *Hugging Face* facilita el acceso a estos modelos a través de su *Transformers library*, permitiendo a los desarrolladores implementar soluciones avanzadas de NLP con un mínimo esfuerzo.

Acceso a modelos preentrenados

Uno de los mayores beneficios de *Hugging Face* es su extensa biblioteca de modelos preentrenados. Los desarrolladores pueden acceder a estos modelos y utilizarlos directamente en sus aplicaciones sin necesidad de entrenar desde cero. Esto ahorra tiempo y recursos significativos. Ejemplos de modelos disponibles incluyen:

- **GPT-2 y GPT-3:** modelos de generación de texto que pueden ser utilizados para construir *chatbots* que mantienen conversaciones fluidas y naturales.
- **BERT:** modelo bidireccional optimizado para tareas de comprensión del lenguaje, como respuesta a preguntas y clasificación de texto.
- **T5:** un modelo versátil que aborda diversas tareas de NLP como problemas de traducción, incluyendo la generación de respuestas en *chatbots*.

Implementación de *chatbots* con *Hugging Face*

Hugging Face facilita la implementación de *chatbots* a través de su *API* y su librería *Transformers*. Los pasos típicos para construir un *chatbot* incluyen:

- **Carga del modelo:** utilizar la función *from_ pretrained* para cargar un modelo preentrenado relevante para la tarea del *chatbot*.
- **Ajuste fino (*Fine-Tuning*):** ajustar el modelo a un conjunto de datos específico para personalizar las respuestas del *chatbot* según el dominio.
- **Despliegue:** integrar el modelo ajustado en una aplicación web o móvil, o incluso en plataformas de mensajería como *Slack* o *Telegram*.

Ejemplo de código para cargar un modelo preentrenado y generar respuestas:

```
from transformers import AutoModelForCausalLM, AutoTokenizer

tokenizer = AutoTokenizer.from_pretrained("gpt2")
model = AutoModelForCausalLM.from_pretrained("gpt2")

inputs = tokenizer("Hola, ¿cómo estás?", return_tensors="pt")
outputs = model.generate(inputs.input_ids, max_length=50, do_sample=True)

print(tokenizer.decode(outputs[0], skip_special_tokens=True))
```

Facilidades adicionales de *Hugging Face*

Además de los modelos preentrenados, *Hugging Face* ofrece herramientas como ***Model Hub*** y ***Datasets***, que proporcionan acceso a miles de modelos y conjuntos de datos públicos, facilitando la experimentación y el desarrollo de soluciones NLP.

Integraciones con *Google Colab* y *AWS*

Hugging Face se integra fácilmente con *Google Colab*, lo que permite aprovechar los recursos de *GPU* para entrenar y ajustar modelos de manera eficiente. También se integra con *Amazon Web Services (AWS)* a través de *SageMaker*, facilitando el despliegue de modelos en producción.

Consideraciones éticas en el uso de LLM

Dado que los modelos preentrenados de *Hugging Face* pueden tener sesgos inherentes derivados de los datos con los que fueron entrenados, es crucial considerar las implicaciones éticas de su uso, especialmente en aplicaciones sensibles como *chatbots* que interactúan con usuarios vulnerables. Implementar mecanismos de revisión y ajuste es fundamental para mitigar estos riesgos.

Relevancia para la construcción de *chatbots* inteligentes

La plataforma *Hugging Face* democratiza el acceso a modelos avanzados de NLP, permitiendo que tanto desarrolladores experimentados como novatos puedan construir *chatbots* potentes y eficientes. La facilidad para cargar modelos, ajustarlos, y desplegarlos convierte a *Hugging Face* en una herramienta esencial para cualquier proyecto que implique procesamiento de lenguaje natural.

4.4.17. *OpenAI* y su *API* para modelos de lenguaje

OpenAI es una organización dedicada a la investigación y desarrollo de inteligencia artificial avanzada. Uno de sus logros más conocidos es la creación de modelos de lenguaje natural basados en arquitecturas de redes neuronales profundas, como la serie de modelos GPT (*Generative Pre-trained Transformer*). Los modelos GPT son herramientas versátiles capaces de realizar tareas de generación y comprensión de lenguaje, tales como resumen, traducción, respuesta a preguntas y generación de texto. En particular, el modelo GPT-3.5-turbo se distingue por su eficiencia y alta capacidad de comprensión contextual, características que lo hacen adecuado para aplicaciones de asistente virtual [180].

API de *OpenAI*

La *API* de *OpenAI* permite el acceso a los modelos de lenguaje mediante un enfoque basado en la nube. Con esta *API*, los desarrolladores pueden integrar modelos de procesamiento de lenguaje natural (PLN) en aplicaciones sin necesidad de contar con infraestructura computacional especializada. La *API* acepta entradas de texto en lenguaje natural, que el modelo procesa para generar una respuesta adecuada basada en su entrenamiento.

Características principales de la *API*

- **Interfaz sencilla y flexible**: la *API* de *OpenAI* se basa en peticiones HTTP estándar, utilizando el método `POST` para enviar datos. Los desarrolladores solo necesitan una clave de *API* para acceder al servicio, y la documentación proporciona parámetros configurables como el modelo específico a utilizar (e.g., GPT-3.5-turbo), la longitud de respuesta en *tokens*, y el nivel de creatividad o precisión de las respuestas (`temperature`).
- **Control de tokens**: los modelos de *OpenAI* operan en base a *tokens*, una unidad que representa palabras, fragmentos de palabras o caracteres especiales en el texto. El control de *tokens* permite a los desarrolladores limitar la longitud de las respuestas y gestionar el costo de uso de la *API*, ya que *OpenAI* cobra según la cantidad de *tokens* procesados.
- **Versatilidad en la configuración**: la *API* permite configurar parámetros como `temperature`, que ajusta el grado de aleatoriedad en las respuestas, y `max_tokens`, que define el límite de *tokens* en la respuesta generada. Estos ajustes permiten adaptar el comportamiento del modelo según la tarea específica.

GPT-3.5-turbo: características técnicas y arquitectura

GPT-3.5-turbo es una variante optimizada del modelo GPT-3, diseñada para ofrecer mayor eficiencia en términos de costos de procesamiento y memoria, sin perder precisión en la generación de texto. Basado en la arquitectura *Transformer*, GPT-3.5-turbo emplea mecanismos de autoatención que le permiten capturar dependencias contextuales a través de múltiples capas, proporcionando respuestas coherentes y contextualmente precisas.

Arquitectura de *Transformer* GPT-3.5-turbo sigue la arquitectura *Transformer*, introducida por *Vaswani* et al. en 2017 [236]. Esta arquitectura es altamente eficiente en modelos de lenguaje debido a su capacidad para aprender relaciones contextuales mediante un mecanismo de autoatención. El *Transformer* utiliza capas de autoatención y mecanismos de *feed-forward* en paralelo, lo cual

permite al modelo manejar relaciones a largo plazo en el texto de entrada. En el caso de GPT-3.5-turbo, el modelo está preentrenado en un vasto corpus de texto y luego ajustado para una amplia gama de tareas de PLN.

***Tokens* y manejo de información** GPT-3.5-turbo divide el texto en unidades llamadas *tokens*, que pueden ser palabras completas, fragmentos de palabras o caracteres especiales. Este modelo maneja los *tokens* en dos fases principales:

- **Tokenización**: al recibir un texto, GPT-3.5-turbo lo descompone en *tokens* mediante un proceso de tokenización. Esto permite que el modelo entienda y procese el texto en fragmentos que representan palabras individuales o combinaciones de caracteres.
- **Límite de *tokens* y costos**: cada solicitud a la *API* incluye una restricción de *tokens*, especificada en el parámetro `max_tokens`. Dado que *OpenAI* cobra por el uso de *tokens*, controlar este parámetro ayuda a optimizar el costo y evitar respuestas excesivamente largas. Además, la longitud de contexto del modelo también está limitada por la capacidad de *tokens*, lo cual implica que GPT-3.5-turbo puede procesar hasta un número específico de *tokens* (generalmente 4096 o 8192 *tokens* en total, dependiendo de la versión de la *API*) antes de que la información anterior empiece a desaparecer.

Generación de respuestas y parámetros de control La generación de texto en GPT-3.5-turbo está regulada por varios parámetros:

- ***Temperature***: este parámetro ajusta la creatividad del modelo. Valores de `temperature` más altos, como 0.8, producen respuestas más diversas y creativas, mientras que valores bajos, como 0.2, generan respuestas más coherentes y conservadoras.
- ***Top-p* (Núcleo de muestreo)**: en lugar de elegir *tokens* basándose en la probabilidad más alta, el parámetro `top-p` permite al modelo seleccionar *tokens* de un subconjunto que cumpla un umbral de probabilidad. Este método permite generar respuestas de alta calidad y reducir respuestas inusuales.
- **Frecuencia y penalización de presencia**: estos parámetros ajustan la probabilidad de que el modelo repita o utilice ciertas palabras. La penalización de frecuencia reduce la probabilidad de repetición de palabras ya usadas en la conversación, mientras que la penalización de presencia desalienta al modelo de mencionar palabras específicas según las preferencias del usuario.

Ventajas de GPT-3.5-turbo en procesamiento de lenguaje natural

GPT-3.5-turbo es ampliamente utilizado en aplicaciones que requieren una comprensión profunda del lenguaje natural debido a su capacidad para generar texto de alta calidad y su flexibilidad. Las principales ventajas de este modelo incluyen:

- **Capacidad de comprensión contextual**: gracias a su arquitectura *Transformer* y preentrenamiento extenso, GPT-3.5-turbo puede captar contextos complejos, manteniendo coherencia en respuestas largas y encadenadas.
- **Adaptabilidad a múltiples tareas**: desde asistencia al usuario hasta generación de contenido y traducción, GPT-3.5-turbo se adapta fácilmente a una amplia gama de aplicaciones de PLN mediante ajustes en los parámetros de la *API*.

- **Eficiencia en el uso de recursos**: la optimización en la cantidad de parámetros y la eficiencia en el manejo de *tokens* permiten que GPT-3.5-turbo ofrezca un rendimiento de alta calidad a un menor costo computacional.

Este modelo ha transformado el campo del PLN al ofrecer capacidades de lenguaje que permiten su aplicación en sistemas de asistencia, *chatbots*, creación de contenido y análisis de texto, facilitando el desarrollo de interfaces más naturales y accesibles para los usuarios [180].

4.4.18. *FastAPI* y *Jinja2*: herramientas para el desarrollo de aplicaciones web

FastAPI y *Jinja2* son dos herramientas populares en el desarrollo de aplicaciones web modernas. *FastAPI* es un marco de desarrollo web de alto rendimiento basado en *Python*, especialmente diseñado para la construcción de *API* rápidas y escalables. *Jinja2*, por su parte, es un motor de plantillas que permite generar contenido *HTML* dinámico, facilitando la creación de interfaces web interactivas. En conjunto, estas herramientas ofrecen una solución robusta para el desarrollo de aplicaciones web rápidas, eficientes y mantenibles.

FastAPI: desarrollo de *API* rápidas y escalables

FastAPI es un marco de trabajo basado en *Python* para la creación de *API* de alto rendimiento. Diseñado por Sebastián Ramírez, *FastAPI* se basa en el estándar *ASGI (Asynchronous Server Gateway Interface),* lo que permite la ejecución de peticiones de manera asíncrona y mejora significativamente el rendimiento de las aplicaciones web [237].

Características Principales de *FastAPI*

- **Soporte para operaciones asíncronas**: *FastAPI* permite el uso de funciones asíncronas (`async`) para gestionar múltiples solicitudes de forma simultánea, lo que optimiza el tiempo de respuesta en aplicaciones con alta demanda. Esta capacidad lo hace especialmente útil en escenarios donde es necesario procesar peticiones concurrentes, como en *chatbots* o sistemas de recomendación.
- **Validación automática de datos**: basado en las anotaciones de tipo de *Python (type hints), FastAPI* valida automáticamente los datos de entrada, simplificando la gestión de errores y asegurando que las solicitudes cumplen con los requisitos esperados. Esta funcionalidad se logra a través de la integración con la biblioteca *Pydantic*, que permite la validación y conversión de datos en tiempo de ejecución.
- **Documentación automática**: *FastAPI* genera documentación automática de las *API* mediante las especificaciones *OpenAPI* y *Swagger UI*. Esto facilita la colaboración entre desarrolladores y permite a los usuarios de la *API* comprender de forma clara las rutas, parámetros y respuestas que ofrece el servicio.
- **Alto rendimiento**: gracias a su compatibilidad con *ASGI* y su arquitectura asíncrona, *FastAPI* ofrece un rendimiento superior en comparación con otros marcos web tradicionales de *Python*, como *Flask* o *Django*. La combinación de la ejecución asíncrona y la optimización interna de *FastAPI* lo convierte en una opción ideal para aplicaciones que requieren alta eficiencia.

Jinja2: motor de plantillas para contenido *HTML* dinámico

Jinja2 es un motor de plantillas en *Python* que permite la generación dinámica de contenido *HTML*. Desarrollado por Armin Ronacher, *Jinja2* es ampliamente utilizado en aplicaciones web para separar la lógica de negocio de la presentación de la interfaz, facilitando la creación de interfaces web que son tanto dinámicas como mantenibles [238].

Características principales de *Jinja2*

- **Sintaxis similar a *Python***: la sintaxis de *Jinja2* es intuitiva y similar a la de *Python*, lo que facilita su adopción por parte de desarrolladores familiarizados con el lenguaje. Esto permite realizar operaciones lógicas y de control de flujo directamente en las plantillas, como bucles, condiciones y filtros.
- **Separación de lógica y presentación**: *Jinja2* permite separar la lógica de presentación de la lógica de negocio de una aplicación, albergando el código *HTML* y el contenido dinámico en archivos de plantilla. Esto facilita el mantenimiento del código y promueve la reutilización de componentes de la interfaz.
- **Filtros y extensiones**: *Jinja2* ofrece una amplia variedad de filtros y extensiones que permiten manipular datos de manera eficiente en las plantillas. Los filtros permiten transformar valores (como formatear fechas o cambiar el formato de texto), mientras que las extensiones añaden funcionalidades avanzadas, como la inclusión de plantillas y el uso de macros.
- **Seguridad integrada**: *Jinja2* incluye funcionalidades de escapado automático, lo cual ayuda a proteger las aplicaciones web contra ataques de inyección de código *HTML* (*Cross-Site Scripting* o XSS). Esta característica hace de *Jinja2* una herramienta segura y confiable para la generación de contenido dinámico en aplicaciones web.

Integración de *FastAPI* y *Jinja2* en aplicaciones web

La combinación de *FastAPI* y *Jinja2* permite construir aplicaciones web eficientes y dinámicas. *FastAPI* proporciona las rutas y gestiona las solicitudes de la *API*, mientras que *Jinja2* facilita la creación de plantillas *HTML* personalizables que responden de manera dinámica a las peticiones de los usuarios. Esta integración es especialmente útil en el desarrollo de aplicaciones que requieren una interfaz web, como *dashboards* de administración, sistemas de recomendación o asistentes virtuales. En conjunto, *FastAPI* y *Jinja2* ofrecen una solución completa para desarrollar aplicaciones web con capacidades avanzadas y rendimiento optimizado.

4.4.19. *MLOps* y *MLFlow*

El desarrollo y la implementación de modelos de *Machine Learning (ML)* a escala requieren prácticas estandarizadas para gestionar eficazmente el ciclo de vida de los modelos. Este enfoque se conoce como *MLOps*, que combina principios de *DevOps* con técnicas de ingeniería de datos y ML [239]. *MLOps* es esencial para proyectos que buscan un despliegue continuo y un mantenimiento adecuado de los modelos de ML en producción.

Concepto de *MLOps*

MLOps, o *Machine Learning Operations*, se refiere a un conjunto de prácticas diseñadas para automatizar y optimizar el ciclo de vida completo de los modelos de ML, desde la preparación

de datos y el entrenamiento del modelo hasta su despliegue y monitoreo [240]. La idea central es garantizar que los modelos sean reproducibles, confiables y escalables.

Componentes clave de *MLOps*

- **Preparación y preprocesamiento de datos**: la gestión de datos es fundamental en *MLOps*, ya que los modelos dependen de la calidad de los datos. Se implementan técnicas para limpiar y procesar datos de manera consistente, asegurando su validez y relevancia [241].
- **Entrenamiento del modelo**: incluye la selección de algoritmos, ajuste de hiperparámetros y validación. *MLOps* exige que los experimentos sean rastreables y reproducibles para optimizar y comparar modelos de manera efectiva [242].
- **Versionado de modelos**: los modelos evolucionan con el tiempo, por lo que el versionado es crucial. Permite rastrear los cambios en datos, arquitecturas y configuraciones, facilitando la selección de modelos óptimos para su despliegue [243].
- **Despliegue y escalado**: la automatización del despliegue en entornos de producción asegura actualizaciones rápidas y seguras, garantizando que los modelos puedan manejar diferentes cargas de trabajo [240].
- **Monitoreo y mantenimiento**: se requiere un monitoreo en tiempo real para detectar desviaciones en el rendimiento del modelo, especialmente cuando hay cambios en la distribución de los datos, conocido como *drift* de datos [242].
- **Automatización de pruebas**: asegura la calidad y estabilidad del modelo, verificando que las predicciones sean consistentes y fiables antes de desplegar el modelo [241].
- **Gestión de la infraestructura**: se encarga de administrar los recursos para entrenamiento y despliegue, garantizando eficiencia y seguridad [243].

Beneficios de *MLOps*

- **Escalabilidad**: permite a las organizaciones escalar modelos de manera eficiente, ajustándose a las demandas cambiantes [239].
- **Automatización y eficiencia**: la automatización de tareas reduce el riesgo de errores manuales y ahorra tiempo en el proceso de desarrollo [240].
- **Reproducibilidad**: garantiza que los experimentos puedan replicarse con precisión, lo cual es vital para validar los resultados y mantener la coherencia [242].
- **Mejor monitoreo y mantenimiento**: permite un seguimiento continuo del rendimiento, facilitando la detección de problemas y ajustes necesarios [241].

MLFlow: herramienta de *MLOps*

MLFlow es una plataforma de código abierto que gestiona el ciclo de vida de los modelos de ML. Proporciona un marco integral para rastrear experimentos, versionar modelos, manejar flujos de trabajo y desplegar modelos [240]. Fue desarrollada por *Databricks* y ha ganado gran aceptación en la comunidad de ciencia de datos.

Componentes de *MLFlow*

MLFlow se compone de cuatro módulos principales:

- ***MLFlow tracking***: permite registrar y consultar experimentos, almacenando parámetros, métricas y artefactos del modelo. Esto facilita el análisis comparativo y la reproducibilidad [244].
- ***MLFlow projects***: define un formato estándar para empaquetar proyectos, incluyendo dependencias y entornos, lo cual simplifica la ejecución reproducible en distintos entornos [244].
- ***MLFlow models***: ofrece un estándar para empaquetar modelos que permite una fácil integración con herramientas de inferencia y despliegue [240].
- ***MLFlow model registry***: un sistema centralizado para gestionar las versiones de los modelos y promoverlos a producción, asegurando una gestión efectiva del ciclo de vida [244].

Ventajas de utilizar *MLFlow*

- **Rastreo integral de experimentos**: facilita el seguimiento detallado de los experimentos, promoviendo una gestión eficaz y análisis exhaustivo [244].
- **Gestión eficiente de modelos**: el versionado estructurado ayuda a mantener un control efectivo sobre los modelos, facilitando colaboraciones y mejoras [240].
- **Promueve la colaboración**: *MLFlow* centraliza el manejo de experimentos y modelos, permitiendo a los equipos trabajar de manera más coordinada [242].
- **Integración con *CI/CD***: se puede integrar con sistemas de integración y entrega continua, optimizando el proceso de despliegue y actualización en producción [244].

Casos de uso de *MLFlow*

MLFlow se aplica en varios contextos, tales como:

- **Investigación y desarrollo**: ayuda a gestionar y rastrear experimentos, agilizando el proceso de investigación científica [243].
- **Despliegue de modelos a producción**: facilita la transición de los modelos desde el entorno de desarrollo hasta la producción, asegurando un rendimiento robusto y continuo [239].
- **Monitoreo y actualización**: proporciona herramientas que permiten rastrear el rendimiento y actualizar los modelos según sea necesario, asegurando que sigan siendo efectivos a lo largo del tiempo [242].

La integración de *MLFlow* y *MLOps* en proyectos de ML es esencial para garantizar que los modelos no solo se desarrollen eficazmente, sino que también se desplieguen y mantengan adecuadamente en entornos complejos.

4.5. Marco teórico sobre infraestructura en la nube

4.5.1. Manejo de desechos y disminución de la huella de carbono

Manejo de desechos

La basura es todo lo que se descarta después de haber cumplido su función y necesita ser eliminado. Si no se maneja adecuadamente, tiene efectos negativos sobre el medio ambiente y la salud humana. En la actualidad, las personas generan grandes cantidades de basura. Todas las actividades producen residuos, por lo que estamos obligados a convivir con ellos y buscar maneras de aprovecharlos. Nuestro objetivo es minimizar estos impactos. Debemos cuidar el medio ambiente porque nos proporciona todos los recursos que nosotros y nuestros hijos necesitamos para vivir. De él obtenemos alimentos, aire, agua y materiales esenciales para nuestra supervivencia. Para mantener la calidad de vida y la salud, es fundamental cuidar del medio ambiente y sus recursos naturales [245]

La gestión de residuos comienza con la clasificación adecuada de los desechos, lo que permite un procesamiento más eficiente y efectivo. Este principio es fundamental en toda estrategia de manejo de residuos, ya que su principal objetivo es optimizar la gestión de los desechos y, por lo tanto, reducir su impacto ambiental [245].

La correcta segregación de residuos en categorías como orgánicos, reciclables, y no reciclables, facilita el reciclaje y la reutilización de materiales. Esto no solo disminuye la cantidad de basura que llega a los vertederos, sino que también reduce la contaminación del suelo, agua y aire. Al separar los residuos desde su origen, se mejora la eficiencia de las plantas de reciclaje y se promueve una economía circular, donde los materiales se reintegran al ciclo productivo [246].

Además, una buena gestión de residuos contribuye a la conservación de recursos naturales y la reducción de emisiones de gases de efecto invernadero. La implementación de programas de educación y concienciación sobre la importancia de la clasificación de desechos es esencial para lograr una participación activa de la comunidad [247].

En resumen, el manejo de desechos basado en una clasificación adecuada es una práctica esencial para mejorar la calidad de la gestión de residuos y mitigar su impacto ambiental. Al adoptar estas prácticas, podemos avanzar hacia un futuro más sostenible y saludable para todos. Y es una de las definiciones principales que inicia este proyecto y su módulo correspondiente de infraestructura [248].

Mitigación de emisiones de carbono

Asociado al manejo de residuos y su correcta implementación, surge un impacto directo en las emisiones de carbono, lo que conlleva una reducción significativa de su huella de carbono. Al implementar prácticas eficientes de gestión de residuos, se reduce la cantidad de residuos que se envían a los vertederos, donde la descomposición de la materia orgánica produce metano, un gas de efecto invernadero mucho más potente que el dióxido de carbono [249].

La separación y reciclaje de materiales como papel, plástico, vidrio y metales no solo evitan que estos materiales se acumulen en los vertederos, sino que también reducen la necesidad de producir nuevos materiales desde cero [247]. La producción de materiales reciclados generalmente consume menos energía y recursos en comparación con la producción de nuevos materiales, lo que se traduce en menores emisiones de gases de efecto invernadero [250].

Asimismo, la correcta gestión de residuos orgánicos a través del compostaje evita la liberación de metano y genera compost que puede utilizarse para mejorar la calidad del suelo, cerrando así el ciclo de nutrientes y reduciendo la dependencia de fertilizantes químicos. Además, el uso de

tecnologías avanzadas de tratamiento de residuos, como la digestión anaeróbica y la incineración con recuperación de energía, permite convertir los residuos en fuentes de energía renovable, disminuyendo la dependencia de combustibles fósiles [249].

Un manejo eficiente de residuos también implica la reducción de residuos en su origen, mediante políticas de producción más limpia y consumo responsable. Esto incluye el diseño de productos más duraderos, reparables y reciclables, y la promoción de estilos de vida que minimicen el desperdicio [251].

4.5.2. Regulaciones y directrices en la gestión de residuos en Guatemala

Luego de tener estas definiciones claras, lo que se trata de alcanzar respecto a este contexto es extender la situación actual de Guatemala y el porqué este módulo es imperativo para poder obtener el proyecto funcional. Adentrándonos más en el contexto de Guatemala podemos aclarar los siguientes puntos.

Principales regulaciones

En el acuerdo gubernativo número 73-2021 del reglamento orgánico interno del ministerio de ambiente y recursos naturales se crea la dirección para el manejo de residuos y desechos sólidos, como el órgano responsable de elaborar los planes, programas, proyectos, guías y la normativa nacional para el fomento de la política nacional para la gestión integral de residuos y desechos sólidos le corresponden las funciones siguientes [252]:

1. Elaborar el plan de inspecciones para verificar el cumplimiento de los compromisos acordados para el manejo de los residuos y de los desechos sólidos en el país;
2. Coordinar con la dirección de cumplimiento legal, las inspecciones de oficio o por denuncia con relación a vertederos o botaderos ilegales;
3. Emitir opiniones técnicas sobre planes, programas y proyectos vinculados a la gestión integral de los residuos y los desechos sólidos que sean requeridas;
4. Desarrollar las acciones necesarias para la modernización de la gestión integral de los residuos y los desechos sólidos;
5. Realizar en coordinación con la dirección de formación y participación social, los procesos de capacitación sobre el tema de gestión integral de los residuos y los desechos sólidos, a nivel nacional;
6. Proporcionar asistencia técnica al sector público o privado, para el manejo de residuos y desechos sólidos, cuando así lo requieran; y,
7. Otras funciones que le sean asignadas por el despacho superior, de conformidad con la Ley.

Podemos apreciar que la entidad principal que maneja los desechos sólidos en Guatemala corresponde a la dirección para el manejo de residuos y desechos sólidos. Sin embargo, existen otras regulaciones cómo el ministerio de ambiente y recursos naturales (MARN) que emitió el acuerdo gubernativo 164-2021: reglamento para la gestión integral de los residuos y desechos sólidos comunes. Que es solo una de las iniciativas que promueve, para impactar, de manera positiva, a la sociedad guatemalteca, lo que repercutirá en un mejor futuro para todos [247].

Situaciones críticas

Los decretos gubernamentales en Guatemala han intentado regular la gestión de desechos sólidos. Sin embargo, la falta de seriedad y protocolos evidencia un desafío persistente en este ámbito. Esta situación subraya la necesidad de iniciativas continuas y propuestas innovadoras. Guatemala se enfrenta a un reto considerable en la gestión adecuada de los desechos sólidos. Aunque se han promulgado regulaciones, su implementación enfrenta obstáculos diversos, desde la insuficiencia de recursos hasta problemas de corrupción.

La aprobación del reglamento para la gestión integral de los residuos y desechos sólidos comunes es un paso crucial. Su objetivo es gestionar estos materiales desde su origen para proteger la salud pública y prevenir la contaminación ambiental. Sin embargo, la prórroga de la segunda fase de este reglamento hasta febrero de 2025 indica retrasos significativos en su implementación [248].

Es fundamental reconocer que la responsabilidad no recae únicamente en el gobierno. Cada individuo desempeña un papel crucial en esta causa. Medidas simples, como reducir el consumo de plásticos y practicar el reciclaje en el hogar, pueden contribuir significativamente al cuidado del medio ambiente y la salud pública [248]

El impacto negativo de una gestión inadecuada de los desechos sólidos en el medio ambiente y la salud pública es evidente. La contaminación del suelo, el agua y el aire son consecuencias directas de esta problemática. Además, la exposición a desechos tóxicos representa un riesgo significativo para la salud de la población.

Para abordar estos desafíos, Guatemala podría tomar inspiración de iniciativas exitosas en otros lugares del mundo. Modelos de gestión de desechos sólidos eficientes, implementados en países o ciudades con condiciones similares, podrían adaptarse y aplicarse en el contexto guatemalteco. Estas iniciativas podrían incluir programas de reciclaje innovadores, incentivos para la reducción de residuos y campañas de sensibilización ambiental.

4.5.3. Arquitectura de nube

Al entender plenamente el contexto actual de la problemática que aborda el proyecto, así como la función específica que este módulo busca cumplir dentro de ese contexto, podemos ahora concentrarnos en definir las metas y objetivos relacionados con su desarrollo. El objetivo principal de este módulo es diseñar e implementar una arquitectura en la nube que provea todos los recursos necesarios para el desarrollo de una aplicación móvil dedicada a fomentar prácticas sostenibles de reciclaje en Guatemala.

Para esclarecer este propósito, es importante comenzar definiendo qué entendemos por arquitectura en la nube. La arquitectura en la nube constituye un componente esencial en el entorno de la computación en la nube. Se trata del diseño que engloba y conecta todos los elementos y tecnologías requeridos para la operación eficiente de aplicaciones y servicios en la nube [253]. En este sentido, la arquitectura en la nube no solo facilita el despliegue y funcionamiento de sistemas, sino que también proporciona la flexibilidad y escalabilidad necesarias para adaptarse a las demandas cambiantes del entorno digital actual.

Principios de diseño

Basándonos en la referencia del articulo [251] el marco de buena arquitectura establece una serie de principios generales de diseño que facilitan la creación de arquitecturas sólidas en entornos de nube. En primer lugar, se subraya la importancia de evitar conjeturas sobre la capacidad necesaria al implementar cargas de trabajo, dado que la nube ofrece la capacidad de escalar automáticamente

hacia arriba o hacia abajo según la demanda, minimizando así el riesgo de recursos ociosos costosos o limitaciones de rendimiento. Asimismo, se destaca la necesidad de probar sistemas a escala de producción, aprovechando la flexibilidad de la nube para crear entornos de prueba bajo demanda, lo que permite realizar pruebas completas y retirar recursos de manera eficiente y rentable.

Otro principio fundamental es la automatización con miras a la experimentación arquitectónica. La automatización permite crear y replicar cargas de trabajo de forma económica y evitar los gastos asociados con los esfuerzos manuales [254]. Esta práctica no solo agiliza los procesos, sino que también permite realizar un seguimiento de los cambios, auditar su impacto y revertir si es necesario [255]. Además, se recomienda considerar el uso de arquitecturas evolutivas, que permiten que los sistemas evolucionen con el tiempo para adaptarse a los cambios en los requisitos empresariales, aprovechando la capacidad de la nube para automatizar y probar bajo demanda, lo que reduce el riesgo de impacto ante cambios en el diseño inicial [256].

Otro punto importante es impulsar las decisiones arquitectónicas mediante el uso de datos. En la nube, es posible recopilar datos sobre cómo las decisiones arquitectónicas afectan el comportamiento de la carga de trabajo, lo que permite tomar decisiones fundamentadas para mejorar continuamente la arquitectura [257]. Al tratarse de infraestructura como código, estos datos pueden utilizarse para informar y mejorar las elecciones de arquitectura a lo largo del tiempo [258]. Por último, se sugiere mejorar mediante "días de juego", programando periódicamente simulaciones de eventos de producción para probar el rendimiento de la arquitectura y los procesos, lo que ayuda a identificar áreas de mejora y desarrollar la experiencia organizacional en la gestión de eventos [259].

Principios de desarrollo sostenible

La tecnología de la nube ha surgido como un elemento crucial en la búsqueda de la sostenibilidad en el ámbito de las organizaciones de tecnología de la información (TI). Constatamos que estas entidades representan una parte significativa de la huella de carbono de las empresas a nivel global [260]. A pesar de ello, se ha observado un crecimiento en la conciencia sobre la importancia de la sostenibilidad en este sector, con más del 40 % de las compañías de TI reconociendo la necesidad de adoptar prácticas más sostenibles y aumentando sus inversiones en programas que promuevan este objetivo.

En este contexto, la tecnología de la nube se destaca como una herramienta fundamental para abordar los desafíos de sostenibilidad. La nube ofrece una serie de recursos y soluciones que pueden contribuir significativamente a la reducción de la huella de carbono de las organizaciones [261]. Por ejemplo, proporciona calculadoras de sostenibilidad que permiten medir y evaluar el impacto ambiental de las operaciones en la nube [262]. Además, la inteligencia artificial (IA) y la analítica ofrecen la capacidad de optimizar los datos y los procesos para mejorar la eficiencia en toda la cadena de valor [263].

Otro aspecto relevante es la gestión de costes (*FinOps*), que permite integrar la sostenibilidad en los procesos de control de costes sin comprometer los objetivos ambientales [264]. Asimismo, la innovación del ecosistema y los nuevos modelos operativos de TI fomentan el desarrollo de prácticas más sostenibles y eficientes [265]. En este sentido, la nube puede impulsar la integración de la sostenibilidad en todas las áreas de la organización, desde la infraestructura hasta el desarrollo de aplicaciones [266].

Se propone un modelo de madurez para la sostenibilidad en la nube que se centra en tres niveles: la optimización del centro de datos y la infraestructura, la gestión de datos y plataformas, y el desarrollo de aplicaciones [250]. Se espera que la sostenibilidad se convierta en uno de los criterios principales de compra de servicios en la nube para 2025, lo que impulsará la adopción de prácticas más sostenibles en toda la industria.

En resumen, la tecnología de la nube ofrece un amplio abanico de oportunidades para promover la sostenibilidad en las organizaciones de TI. Sin embargo, es importante que las empresas establezcan objetivos claros y desarrollen una estrategia integral para aprovechar al máximo el potencial de la nube en este sentido. Esto incluye la evaluación del progreso, la definición de una hoja de ruta a largo plazo y la integración de la sostenibilidad en la cultura organizacional. Con un enfoque adecuado, la nube puede ser un motor clave para la transformación hacia modelos más eficientes y sostenibles.

Infraestructura en la nube multiplataforma vs en un solo proveedor

La elección entre utilizar una infraestructura en la nube de un solo proveedor o una configuración multiplataforma es una decisión estratégica que impacta tanto la flexibilidad como el costo y la complejidad de la administración [267].

- **Infraestructura de un solo proveedor**: al optar por un único proveedor de nube, las empresas pueden beneficiarse de una integración más profunda y simplificada entre los servicios de dicho proveedor, lo que facilita la administración y reduce la latencia [268]. Además, los proveedores suelen ofrecer descuentos por el volumen de uso y soporte específico al cliente [269]. Sin embargo, esta estrategia puede conllevar una dependencia considerable del proveedor (*vendor lock-in*), lo que podría limitar la flexibilidad y la capacidad de migrar a otros servicios en el futuro [270].
- **Infraestructura multiplataforma**: la implementación de una infraestructura en múltiples plataformas permite a las organizaciones aprovechar las mejores capacidades de cada proveedor, evitando la dependencia de un solo servicio y reduciendo el riesgo de fallas de disponibilidad [271]. Esta estrategia también permite una mayor flexibilidad y resiliencia, ya que las cargas de trabajo se pueden distribuir o mover entre plataformas [272]. No obstante, la gestión de una infraestructura multiplataforma requiere una mayor complejidad operativa, conocimientos de integración adicionales y posibles costos de transferencia de datos entre nubes [273].

Tanto una infraestructura en un solo proveedor como una configuración multiplataforma tienen ventajas y desventajas, y la elección dependerá de las necesidades específicas de flexibilidad, presupuesto, resiliencia y capacidades técnicas de la organización [274].

Costos e implementación

En el entorno de desarrollo actual, los precios del almacenamiento en la nube juegan un papel crucial en las decisiones financieras y operativas de las organizaciones. En contraste con el modelo de costos tradicional del almacenamiento local, los precios en la nube pueden resultar confusos y desafiantes de entender. Es por ello que se ha vuelto imperativo para las empresas comprender a fondo estas estructuras de precios para poder maximizar el valor y minimizar los costos [275].

Uno de los puntos clave es la variedad de variables que influyen en los costos del almacenamiento en la nube empresarial. Estas variables incluyen no solo el costo del almacenamiento en sí, expresado generalmente como un costo por *gigabyte* por mes, sino también las operaciones de almacenamiento, la transferencia de datos y el ancho de banda de egreso [276]. Este último aspecto es especialmente relevante ya que puede generar costos adicionales significativos, sobre todo en entornos con altos volúmenes de tráfico de datos [277].

Los niveles de rendimiento, diseñados para datos activos de uso frecuente y transaccionales, los niveles de bajo contacto, destinados a datos que requieren acceso ocasional pero rápido, y los niveles de archivado, ideales para almacenamiento a largo plazo con tiempos de recuperación más lentos pero costos significativamente más bajos [278].

Un aspecto relevante es la importancia de optimizar los costos de almacenamiento en la nube. Estrategias como la migración de datos a niveles de almacenamiento más rentables pueden ser clave para lograr este objetivo [279]. Es por eso que se opta por utilizar máquinas y recursos que se cobran por consumo, lo que les permite pagar solo por los recursos que realmente utilizan en lugar de incurrir en costos fijos por capacidades que podrían no necesitar en su totalidad [280].

En resumen, al comprender las variables involucradas y las estrategias para optimizar los costos, las organizaciones pueden tomar decisiones más informadas y eficientes en cuanto a su infraestructura de almacenamiento en la nube.

4.5.4. Principales proveedores de infraestructura

Diferenciadores marcados

Amazon Web Services (*AWS*) se consolida como el principal proveedor de servicios en la nube a nivel mundial, exhibiendo una cuota de mercado impresionante del 32 % y generando ingresos anuales que rondan los $80 mil millones. Esta prominencia se refleja no solo en su participación en el mercado, sino también en su extensa base de clientes, que supera los 1.45 millones de empresas [281]. Detrás de *AWS*, *Microsoft Azure* emerge como un competidor formidable, con una participación del 22 %, seguido de cerca por *Google Cloud Platform (GCP)* con el 11 % de la cuota de mercado [282]. Estas cifras, respaldadas por el análisis de Filip Stojanović (2023), evidencian la sólida presencia de estos tres gigantes en el ámbito de la computación en la nube, donde ofrecen una gama diversa de servicios y soluciones que abarcan desde el almacenamiento hasta la inteligencia artificial.

En el caso de los proveedores de nube privada, como *VMware* y *OpenStack*, se enfocan en proporcionar servicios de computación en la nube exclusivamente a una sola organización. Esta modalidad brinda a las empresas un mayor nivel de seguridad y control, ya que todo el entorno de la nube está aislado para su uso exclusivo [283]. Este modelo es especialmente atractivo para organizaciones que manejan datos sensibles o que requieren un alto grado de personalización en sus servicios en la nube.

Por otro lado, los proveedores de nube pública, como *Amazon Web Services (AWS), Microsoft Azure y Google Cloud Platform (GCP),* ofrecen recursos informáticos a varias empresas o individuos. Estos proveedores operan a gran escala y proporcionan una amplia gama de servicios que van desde el almacenamiento hasta la computación de alto rendimiento [284]. Aunque estas plataformas están compartidas entre múltiples usuarios, ofrecen niveles sólidos de seguridad y cumplimiento normativo [285].

Además de estos dos tipos de proveedores, también existe la opción de la nube híbrida, que combina características tanto de la nube privada como de la pública. Este enfoque flexible permite a las organizaciones almacenar datos confidenciales en una nube privada mientras aprovechan la escalabilidad y la economía de la nube pública [286]. Ejemplos de proveedores de nube híbrida incluyen *Microsoft Azure, Nube Oracle* y Nube *IBM*.

En resumen, mientras que los proveedores de nube privada priorizan la exclusividad y el control, los proveedores de nube pública ofrecen escalabilidad y una amplia gama de servicios compartidos. La elección entre estos modelos dependerá de las necesidades específicas de cada organización en términos de seguridad, control y escalabilidad.

Estilos de interfaz, tipos de trabajo y casos de uso

Los Servicios *Web de Amazon (AWS)* lideran el mercado y sus servicios principales incluyen *Amazon EC2* para servidores virtuales, S3 para almacenamiento escalable y RDS para bases de datos gestionadas [287]. *AWS* se destaca por su red global de centros de datos y su soporte para la

nube privada, híbrida y *multicloud*, siendo reconocible por su escalabilidad en la nube pública. Se pueden realizar despliegues de aplicaciones, ejecución de cargas de trabajo y almacenamiento seguro de datos, pagando solo por los recursos consumidos [288].

Microsoft Azure, con una cuota de mercado del 22 %, ofrece soluciones *IaaS, PaaS y SaaS*, centrándose en aplicaciones de computación en la nube híbrida [289]. Su infraestructura local integrada y su amplia gama de funciones lo hacen ideal para empresas que utilizan tanto nubes privadas como públicas.

La Plataforma en la nube de *Google* (GCP), con una cuota de mercado del 11 %, se distingue por su enfoque en inteligencia artificial (IA) y aprendizaje automático, gestionando grandes cantidades de datos en diversos formatos [290]. Sus herramientas de aprendizaje automático hacen que la interpretación de datos sea más comprensible, ideal para empresas centradas en los datos.

La Nube *Oracle*, con una cuota de mercado del 2 %, ofrece soluciones de rendimiento empresarial y una variedad de herramientas para gestionar operaciones clave en la nube. Orientado principalmente a aplicaciones empresariales, destaca por su robusta arquitectura, copias de seguridad automáticas y sofisticadas funciones de red [291].

Por último, la nube de *IBM*, con una cuota de mercado del 3 %, se destaca por sus capacidades de seguridad. Ofrece soluciones *full-stack* de computación en la nube con un enfoque en la seguridad, incluyendo herramientas como *IBM Cloud Pak, IBM Security Guardium e IBM QRadar*, ideales para gestionar riesgos empresariales y evitar amenazas activas al sistema [292].

Como resultado, los principales proveedores de servicios en la nube, como *AWS, Microsoft Azure, Google Cloud Platform, Oracle Cloud* y *IBM Cloud*, ofrecen una variedad de soluciones para satisfacer las necesidades de las empresas en un mundo cada vez más digitalizado. Desde la escalabilidad y flexibilidad de *AWS* hasta el enfoque en inteligencia artificial de GCP y las sólidas capacidades de seguridad de *IBM Cloud*, cada proveedor tiene sus propias fortalezas y características distintivas. La elección del proveedor adecuado dependerá de las necesidades específicas de cada empresa en cuanto a rendimiento, seguridad, integración y costo. Sin embargo, en conjunto, estos servicios en la nube están transformando la manera en que las organizaciones gestionan sus operaciones. [293].

4.5.5. Avances tecnológicos

A pesar de que se está planteando el uso de tecnologías de proveedores de servicios en la nube públicos, esto no implica que no se puedan realizar implementaciones tecnológicas nuevas para que la arquitectura diseñada presente la mayor cantidad de características destacables y sostenibles.

Migración de infraestructura

Ampliando el concepto de diseño e implementación variada de una arquitectura en la nube, podemos explorar cómo este enfoque busca maximizar la flexibilidad y la eficiencia al permitir que una misma estructura de diseño sea adaptable y aplicable en una variedad de servicios y componentes en la nube.

En esencia, esta estrategia reconoce que la arquitectura en la nube es altamente dinámica y está en constante evolución. Por lo tanto, en lugar de desarrollar diseños específicos para cada servicio o componente, se busca crear una arquitectura general que pueda ser fácilmente adaptada y desplegada en diferentes entornos de nube, ya sean públicos, privados o híbridos [294].

Sin embargo, este enfoque presenta varios desafíos significativos desde el principio. Uno de los principales desafíos es la diversidad de proveedores de servicios en la nube y las diferencias en sus características, *API*, configuraciones y protocolos. Cada proveedor de servicios de nube tiene su

propia infraestructura, conjuntos de herramientas y políticas que deben ser consideradas al diseñar una arquitectura en la nube que sea compatible con múltiples plataformas [260].

Por ejemplo, un servicio que funciona de manera óptima en una plataforma de nube puede no ser compatible o puede requerir modificaciones significativas para funcionar correctamente en otra plataforma. Esto puede incluir diferencias en la forma en que se gestionan los recursos, se implementan las políticas de seguridad, se escalan las aplicaciones, se realizan las copias de seguridad y se integran con otros servicios [295]. Además, las actualizaciones y cambios en los servicios en la nube pueden afectar la interoperabilidad de la arquitectura, lo que requiere una gestión proactiva y una rápida adaptación para garantizar que los sistemas continúen funcionando de manera eficiente y segura [296].

En resumen, el diseño e implementación variada de una arquitectura en la nube es un desafío complejo que requiere un enfoque cuidadoso y una comprensión profunda de las diferentes plataformas de nube y sus características. Sin embargo, al superar estos desafíos, las organizaciones pueden beneficiarse de una mayor flexibilidad, agilidad y eficiencia en sus operaciones en la nube [286].

Identificación de componentes

Para garantizar la integralidad del proyecto, es esencial llevar a cabo una exhaustiva identificación de los servicios necesarios [297]. Este proceso se inicia con un análisis minucioso de las pantallas previstas para la aplicación, lo que permite comprender a fondo las funcionalidades requeridas y los datos que se manipularán en la plataforma. Cada pantalla o interfaz de usuario proporciona pistas valiosas sobre los servicios que serán imprescindibles para su funcionamiento, ya sea en términos de autenticación de usuarios, gestión de sesiones, interacción con bases de datos u otros aspectos.

Por otro lado, el modelo de inteligencia artificial constituye un componente central de la aplicación en muchos casos. Por lo tanto, identificar los servicios necesarios para ejecutar este modelo es de suma importancia. Esto abarca desde los servicios de procesamiento de datos hasta el almacenamiento de modelos y la infraestructura de cómputo necesaria para el entrenamiento y la inferencia. Además, se deben considerar los servicios de integración que permitan conectar el modelo de IA con otras partes de la aplicación, asegurando así un funcionamiento fluido y eficiente [297].

En paralelo, la recolección de información es un aspecto crucial, especialmente si el modelo de inteligencia artificia (IA) se basa en el aprendizaje supervisado o requiere grandes conjuntos de datos para su entrenamiento. Por lo tanto, es esencial identificar los servicios necesarios para recopilar, almacenar y procesar los datos que alimentarán el modelo. Esto podría incluir servicios de bases de datos para el almacenamiento seguro de datos de usuarios, servicios de interfaz de programación de aplicaciones (API) para la interacción con otras plataformas o sistemas externos, así como herramientas de análisis de datos para extraer información relevante y preparar los datos para su uso en el modelo de IA [298]. En resumen, la identificación de estos servicios sienta las bases para una implementación exitosa y completa del proyecto [299].

Nuevas tendencias en tecnologías verdes

Con el creciente enfoque en la sostenibilidad, las tecnologías verdes están revolucionando la forma en que se gestionan los recursos y los desechos. Las nuevas innovaciones están diseñadas para reducir el impacto ambiental y promover el uso eficiente de los recursos [300].

Tecnologías limpias y su impacto

Las tecnologías limpias, también conocidas como *cleantech*, abarcan una gama de soluciones que buscan mitigar los efectos del cambio climático. Entre estas se incluyen la energía solar, eólica, biomasa y tecnologías de reciclaje avanzadas. Su impacto se refleja en la disminución de la dependencia de combustibles fósiles, la reducción de emisiones de gases de efecto invernadero y el aprovechamiento de materiales reciclables para crear productos sostenibles [301].

Innovación en el uso de recursos renovables

El uso de recursos renovables, como la energía solar y eólica, ha crecido significativamente en los últimos años. Estas innovaciones permiten a las empresas no solo cumplir con las normativas medioambientales, sino también reducir costos operativos a largo plazo. Además, los avances en tecnologías de almacenamiento de energía y eficiencia energética están impulsando la adopción masiva de soluciones renovables en industrias clave [302].

Sistemas de gestión ambiental en la industria

Los sistemas de gestión ambiental (SGA) son marcos estructurados utilizados por organizaciones para minimizar su impacto ambiental. Implementar estos sistemas implica establecer políticas, procedimientos y prácticas que permiten a las empresas cumplir con la legislación ambiental y mejorar su desempeño en sostenibilidad [303].

Modelos de gestión ambiental

Existen varios modelos de gestión ambiental que han sido adoptados por empresas de todo el mundo, siendo el más conocido la norma ISO 14001. Este estándar proporciona una estructura clara para planificar, implementar, supervisar y mejorar las políticas ambientales dentro de una organización, facilitando la gestión eficiente de recursos y la minimización de residuos [303].

Certificaciones y estándares internacionales

Las certificaciones como ISO 14001, *EMAS* (Sistema Comunitario de Gestión y Auditoría Ambientales) y LEED (Liderazgo en Energía y Diseño Ambiental) permiten a las empresas demostrar su compromiso con la sostenibilidad. Estos estándares no solo se centran en la reducción de emisiones y residuos, sino también en mejorar la eficiencia energética y promover la responsabilidad social corporativa [304, 305].

Impacto de la digitalización en la gestión de residuos

La digitalización está cambiando el enfoque hacia la gestión de residuos, ofreciendo nuevas formas de optimizar procesos, reducir costes y aumentar la eficiencia operativa. Mediante la integración de herramientas digitales, las organizaciones pueden gestionar de manera más eficaz sus residuos, logrando una trazabilidad completa de su ciclo de vida [306].

Herramientas digitales para la reducción de desechos

Las plataformas de gestión de residuos digitales permiten a las empresas controlar el flujo de residuos en tiempo real. Estas herramientas pueden incluir *software* de gestión de residuos, aplicaciones móviles para la separación de residuos en origen y tecnologías de mapeo para la optimización de rutas de recolección. El uso de estos sistemas reduce significativamente la cantidad de residuos enviados a vertederos y mejora el reciclaje [307].

Big data y la optimización de procesos

El uso de *Big Data* y análisis predictivos está transformando la gestión de residuos al proporcionar información clave sobre patrones de consumo y producción. A través del análisis de datos, las empresas pueden identificar oportunidades para reducir desperdicios, mejorar la eficiencia del reciclaje y ajustar sus procesos productivos para ser más sostenibles [308].

Educación y conciencia ambiental

La educación y la concientización son factores fundamentales para fomentar la sostenibilidad y el manejo adecuado de los recursos. A través de campañas educativas y de sensibilización, se busca promover hábitos de consumo responsable y prácticas ecológicas tanto a nivel individual como corporativo [309].

Programas educativos sobre gestión de residuos

Diversas organizaciones y gobiernos han implementado programas educativos destinados a la gestión de residuos, con el fin de instruir a la población sobre la importancia del reciclaje y la reducción de desechos. Estos programas incluyen desde charlas en escuelas hasta programas corporativos de sostenibilidad para empleados [310, 311].

Campañas de concientización y su eficiencia

Las campañas de concientización ambiental, como las dirigidas a reducir el uso de plásticos de un solo uso o el ahorro de energía, han demostrado ser altamente efectivas. Al involucrar a la comunidad y crear incentivos para comportamientos sostenibles, estas campañas logran cambios significativos en los patrones de consumo y manejo de residuos [312].

Economía circular y su relación con la tecnología

La economía circular se basa en el principio de mantener los recursos en uso durante el mayor tiempo posible, maximizando su valor y minimizando la generación de desechos. La tecnología juega un papel clave en la implementación de modelos de economía circular, facilitando la reutilización y reciclaje de productos [58].

Conceptos clave de economía circular

En contraste con el modelo económico lineal de "tomar, hacer, desechar", la economía circular busca cerrar el ciclo de vida de los productos a través de la reutilización de materiales, el reciclaje

de componentes y la recuperación de recursos. Este enfoque reduce la demanda de materias primas vírgenes y minimiza el impacto ambiental [313].

Aplicaciones tecnológicas en el ciclo de vida de los productos

La tecnología está permitiendo el desarrollo de soluciones innovadoras para facilitar la transición hacia una economía circular. Ejemplos incluyen aplicaciones de trazabilidad de productos, plataformas para el intercambio de materiales reciclados y herramientas de simulación que ayudan a las empresas a diseñar productos con menor impacto ambiental [314].

Integración de IoT en la gestión de residuos

El Internet de las Cosas (IoT) está transformando la gestión de residuos al permitir una recolección y procesamiento de datos en tiempo real. Mediante sensores conectados, las empresas pueden monitorear la generación de residuos y optimizar su manejo, contribuyendo a la reducción de la huella de carbono [315].

Sensores y monitoreo en tiempo real

Los sensores IoT permiten la supervisión en tiempo real de contenedores de residuos, informando cuándo están llenos y optimizando las rutas de recolección. Esto no solo reduce costos logísticos, sino que también disminuye el consumo de combustibles y las emisiones asociadas al transporte [315,316].

Automatización de procesos y reducción de residuos

La automatización, a través de robots de reciclaje y sistemas de clasificación inteligentes, está mejorando la eficiencia en el manejo de residuos. Estas tecnologías permiten separar materiales reciclables de forma más rápida y precisa, aumentando las tasas de reciclaje y reduciendo la cantidad de residuos que terminan en vertederos [317, 318].

Casos de éxito en la reducción de huella de carbono

En todo el mundo, muchas organizaciones y gobiernos han liderado esfuerzos para reducir la huella de carbono, implementando medidas innovadoras que combinan tecnología y sostenibilidad. Estos casos de éxito sirven como modelos para otras entidades que buscan reducir su impacto ambiental [319].

Iniciativas globales

Ejemplos como el Acuerdo de París, las políticas de cero emisiones netas de empresas como *Google* y *Amazon*, y los esfuerzos de ciudades como *Copenhague* y *Ámsterdam* en su lucha contra el cambio climático destacan por su enfoque en la reducción de emisiones de carbono. Estas iniciativas promueven la energía renovable, el transporte sostenible y la eficiencia energética [320, 321].

Empresas que han liderado el cambio

Compañías como *Tesla, Unilever y IKEA* han adoptado modelos de negocio basados en la sostenibilidad. *Tesla*, por ejemplo, ha transformado la industria automotriz con sus vehículos eléctricos, mientras que *Unilever* ha implementado prácticas de producción sostenibles a lo largo de su cadena de suministro [322, 323].

4.5.6. Costos en la nube y licencias de los proveedores principales

La administración de costos en la nube y las licencias son factores cruciales a la hora de seleccionar un proveedor de infraestructura en la nube. Los principales proveedores como *Amazon Web Services (AWS), Microsoft Azure* y *Google Cloud Platform (GCP)* ofrecen diferentes modelos de precios y licencias, diseñados para ajustarse a las necesidades específicas de cada organización [324].

Amazon Web Services (AWS)

AWS utiliza un modelo de precios basado en el uso, conocido como *pay-as-you-go*, donde los clientes pagan solo por los recursos que utilizan. Los costos varían según la cantidad de almacenamiento, procesamiento, y ancho de banda consumido, entre otros factores [288]. AWS también ofrece descuentos por uso prolongado a través de instancias reservadas y *Savings Plans*, lo que puede resultar beneficioso para cargas de trabajo predecibles.

En cuanto a las licencias, *AWS* permite a los clientes traer sus propias licencias de *software* (BYOL, *Bring Your Own License*) o adquirir licencias bajo un modelo de suscripción, dependiendo de las necesidades de la aplicación [325].

Microsoft Azure

Al igual que *AWS*, *Azure* utiliza un modelo de facturación por consumo (*pay-as-you-go*), lo que permite a los usuarios pagar solo por los recursos utilizados. Además, *Azure* ofrece opciones como el *Azure Hybrid Benefit*, que permite a las empresas utilizar sus licencias existentes de *Windows Server* o *SQL Server*, reduciendo así los costos [326].

Azure también proporciona sus licencias bajo suscripción y permite la opción de *BYOL* para algunos productos de *software*. *Azure* tiene un fuerte enfoque en la integración con productos *Microsoft*, lo que puede ofrecer ventajas a las organizaciones que ya son usuarios de sus servicios [327].

Google Cloud Platform (GCP)

GCP sigue un modelo similar al de *AWS* y *Azure*, cobrando a los usuarios según el uso de sus servicios. *Google* ofrece descuentos automáticos por uso sostenido (*Sustained Use Discounts*) y planes de precios por compromiso a largo plazo, lo que puede ayudar a reducir los costos para las empresas con necesidades constantes de recursos [328].

En términos de licencias, GCP también permite a los usuarios aprovechar sus licencias existentes a través de *BYOL* o adquirir licencias en sus servicios gestionados. *Google* es conocido por su enfoque en soluciones de código abierto, lo que puede ser una ventaja para ciertas industrias [329].

Comparación de costos y licencias

Cada uno de estos proveedores ofrece estructuras de precios y licencias flexibles, lo que facilita a las empresas optimizar sus costos según el uso y las necesidades. Sin embargo, la elección del proveedor adecuado dependerá de factores como la integración con sistemas existentes, la naturaleza de las aplicaciones y el volumen de recursos requeridos [330].

- ***AWS***: flexible con varias opciones de ahorro (instancias reservadas, *Savings Plans*) [288].
- ***Azure***: ventajas significativas para usuarios de productos *Microsoft* con *Azure Hybrid Benefit* [326].
- ***GCP***: descuentos automáticos por uso sostenido y fuerte enfoque en soluciones de código abierto [328].

Opciones de licenciamiento en la nube

Además de las licencias tradicionales, muchos proveedores de nube están impulsando modelos de *Software as a Service* (*SaaS*), donde los usuarios pueden acceder a *software* y aplicaciones en la nube bajo un modelo de suscripción mensual o anual. Este enfoque elimina la necesidad de grandes inversiones iniciales en licencias de *software* y facilita la escalabilidad según el crecimiento del negocio [331].

Las licencias basadas en *SaaS* suelen incluir actualizaciones automáticas y soporte, lo que simplifica la administración para los equipos de TI y asegura que las empresas siempre utilicen las versiones más recientes de los productos [332].

4.5.7. Tipos de servicios en la nube

Los servicios en la nube se clasifican en varios modelos que proporcionan diferentes niveles de control y responsabilidad para los usuarios. Estos modelos cubren desde la gestión de infraestructura básica hasta servicios especializados como la gestión de identidades. Entre los más comunes se encuentran:

Infraestructura como servicio (*IaaS*)

La Infraestructura como Servicio (*IaaS*) ofrece recursos de computación fundamentales, como máquinas virtuales, almacenamiento y redes a través de Internet. Los usuarios tienen control sobre el sistema operativo, las aplicaciones y los recursos de red, pero el proveedor gestiona el *hardware* subyacente. Ejemplos populares de *IaaS incluyen Amazon Web Services (AWS), Microsoft Azure* y *Google Compute Engine (GCE)* [284].

Plataforma como servicio (*PaaS*)

La Plataforma como Servicio (*PaaS*) proporciona un entorno completo para el desarrollo de aplicaciones en la nube. Los desarrolladores pueden crear, probar y desplegar aplicaciones sin preocuparse por la infraestructura subyacente. Este enfoque permite centrarse en el desarrollo de la lógica de negocio, mientras el proveedor gestiona la infraestructura y las actualizaciones. Ejemplos de *PaaS* incluyen *Google App Engine, Microsoft Azure App Services y Heroku* [333].

Software como servicio (SaaS)

El *Software* como Servicio (*SaaS*) permite a los usuarios acceder a aplicaciones completas a través de Internet sin necesidad de instalarlas o gestionarlas localmente. El proveedor de *SaaS* se encarga de todas las actualizaciones, mantenimiento y seguridad. *SaaS* es ideal para aplicaciones que requieren accesibilidad inmediata desde cualquier lugar. Ejemplos de *SaaS* incluyen *Google Workspace, Salesforce y Microsoft 365* [334].

Gestión de identidades y accesos para clientes (CIAM)

Customer Identity and Access Management (CIAM) es un servicio en la nube que gestiona las identidades y accesos de los usuarios externos, como clientes o consumidores. CIAM ofrece autenticación segura, autorización, gestión de perfiles de usuario y cumplimiento de normativas de privacidad, lo que permite a las empresas ofrecer experiencias de acceso personalizadas sin comprometer la seguridad. Ejemplos de CIAM incluyen *Azure AD B2C, Okta Customer Identity y Auth0* [335].

Seguridad como servicio (SECaaS)

La Seguridad como Servicio (SECaaS) ofrece soluciones de seguridad basadas en la nube, incluyendo protección contra amenazas, monitoreo de seguridad y gestión de identidades. Este tipo de servicio permite a las organizaciones asegurar sus aplicaciones y datos sin tener que implementar soluciones de seguridad complejas en sus instalaciones. Proveedores como *Cloudflare, McAfee y Zscaler* ofrecen soluciones de SECaaS [336].

Función como servicio (FaaS)

La Función como Servicio (FaaS), también conocida como computación sin servidor, permite a los desarrolladores ejecutar funciones en respuesta a eventos sin gestionar la infraestructura subyacente. FaaS se usa para crear aplicaciones que responden a eventos como cambios en la base de datos o solicitudes HTTP, pagando solo por el tiempo de ejecución de la función. Ejemplos incluyen *AWS Lambda, Google Cloud Functions* y *Azure Functions* [337].

Identidad como servicio (IDaaS)

El modelo de Identidad como Servicio (IDaaS) ofrece soluciones de gestión de identidades basadas en la nube para empleados y usuarios externos. IDaaS incluye características como autenticación, *Single Sign-On* (SSO) y autenticación multifactor (MFA). Es utilizado para gestionar el acceso seguro a aplicaciones en la nube y garantizar la seguridad de los datos. Ejemplos de IDaaS incluyen *Okta, Ping Identity* y *OneLogin* [338].

Comunicación como servicio (CaaS)

La Comunicación como Servicio (CaaS) proporciona servicios de comunicación en la nube, como videollamadas, mensajería y conferencias. Las empresas pueden integrar estos servicios en sus flujos de trabajo sin necesidad de administrar *hardware* o *software* de comunicación. Ejemplos de CaaS incluyen plataformas como *Zoom, Microsoft Teams* y *Twilio* [339].

Almacenamiento como servicio (SaaS)

El Almacenamiento como Servicio permite a las organizaciones almacenar y gestionar grandes volúmenes de datos en la nube. Este servicio es escalable, accesible y elimina la necesidad de gestionar servidores locales de almacenamiento. Ejemplos de almacenamiento en la nube incluyen *Amazon S3, Google Cloud Storage* y *Microsoft Azure Blob Storage* [340].

Cada uno de estos servicios en la nube ofrece diferentes niveles de control y flexibilidad, lo que permite a las organizaciones seleccionar el modelo que mejor se adapte a sus necesidades técnicas, operativas y de negocio.

Consideraciones para servicios de proveedores principales

Al seleccionar servicios de proveedores de nube como *AWS, Google Cloud* y *Azure*, es fundamental tener en cuenta ciertos aspectos específicos que impactan la gestión, eficiencia y seguridad de la infraestructura. A continuación, se presentan algunas consideraciones clave para los principales proveedores de servicios en la nube [341]:

- ***AWS***: *AWS* ofrece una gran variedad de servicios y capacidades que pueden ser personalizados para distintas cargas de trabajo. Es recomendable evaluar la configuración de escalado automático para optimizar costos en aplicaciones con variación de demanda. Además, es importante utilizar servicios como *AWS CloudTrail* y *AWS Config* para el monitoreo y cumplimiento, ya que ayudan a mantener la visibilidad de cambios en la infraestructura y auditorías de seguridad [342].
- ***Google Cloud***: *Google Cloud* destaca por su integración de inteligencia artificial y aprendizaje automático. Es recomendable aprovechar herramientas como *BigQuery* para análisis de datos en tiempo real y servicios de ML para aplicaciones que necesiten capacidades avanzadas de procesamiento de datos. Para la seguridad, se sugiere emplear *Google Cloud Armor* para la protección contra amenazas externas, especialmente en aplicaciones expuestas a internet [343].
- ***Azure***: *Azure* proporciona un amplio soporte para aplicaciones híbridas y locales, por lo que es ideal para empresas con necesidades de infraestructura mixta. Al utilizar *Azure*, es beneficioso aprovechar *Azure* Monitor para la observabilidad y gestión de rendimiento de aplicaciones, así como *Azure Policy* para la implementación de políticas de cumplimiento y seguridad en múltiples recursos. También se recomienda configurar la recuperación ante desastres mediante *Azure Site Recovery* para minimizar el tiempo de inactividad [344].

IAM (*Identity and Access Management*) en la nube

El IAM (*Identity and Access Management*) es un sistema que permite gestionar identidades y accesos de manera centralizada. En el contexto de la nube, IAM facilita la creación, asignación y control de permisos y roles a los diferentes usuarios y servicios. Existen varios estilos de IAM en la nube, tales como [345]:

- **IAM basado en roles (RBAC)**: permite asignar permisos basados en roles predefinidos, ideal para organizaciones con jerarquías de permisos claras [346].
- **IAM basado en atributos (ABAC)**: otorga permisos según atributos específicos de los usuarios o recursos, permitiendo políticas de acceso más dinámicas y detalladas [347].

- **IAM basado en políticas (PBAC)**: las políticas de permisos se definen mediante condiciones específicas que deben cumplirse, adecuadas para entornos con políticas de acceso complejas [348].
- **IAM descentralizado y federado**: permite la integración con otros sistemas de identidad externos, facilitando la gestión de acceso en aplicaciones y servicios *multi-cloud* [349].

Tipos de IAM en proveedores de nube

AWS:

- ***AWS Identity and Access Management* (IAM)**: proporciona control de acceso basado en roles (RBAC), basado en atributos (ABAC) y basado en políticas (PBAC), permitiendo políticas detalladas que controlan el acceso a recursos específicos [350].
- ***AWS Organizations***: permite administrar cuentas y políticas en una organización, adecuado para gestionar múltiples cuentas dentro de una estructura corporativa [351].
- ***AWS SSO* (Single Sign-On)**: proporciona acceso centralizado a múltiples aplicaciones y cuentas de *AWS*, integrándose con proveedores de identidad externos [352].

Google Cloud:

- ***Identity and Access Management* (IAM)**: implementa *RBAC* y *PBAC*, permitiendo crear políticas de acceso detalladas para usuarios, grupos y cuentas de servicio [353].
- ***BeyondCorp Enterprise***: basado en el modelo de *Zero Trust*, donde el acceso se controla sin necesidad de *VPN*, enfocándose en la seguridad de acceso [354].
- ***Identity-Aware Proxy* (IAP)**: controla el acceso a aplicaciones protegidas mediante políticas de acceso y autenticación de usuario [355].

Azure:

- ***Azure Active Directory* (AAD)**: ofrece control de acceso y autenticación con *RBAC* y *ABAC*, integrando usuarios y grupos en un único directorio [356].
- ***Role-Based Access Control* (RBAC)**: permite asignar roles específicos a usuarios y recursos dentro de *Azure* [357].
- ***Conditional Access Policies***: basado en *ABAC*, permite configurar políticas de acceso dinámico usando condiciones específicas, como ubicación y estado del dispositivo [358].

4.5.8. Beneficios de desarrollar en la nube vs *On-Premise*

Desarrollar en la nube ofrece múltiples beneficios en comparación con el desarrollo *on-premise*. Algunos de los beneficios más destacados incluyen [359]:

- **Escalabilidad y elasticidad**: la nube permite escalar recursos de forma dinámica según las necesidades del proyecto, lo cual es más complejo y costoso de lograr en entornos *on-premise*.

- **Reducción de costos operativos**: al eliminar la necesidad de infraestructura física y de mantenimiento, los desarrolladores pueden enfocarse en el desarrollo y despliegue sin preocuparse por el *hardware*.
- **Acceso global y colaboración**: la nube permite que los equipos trabajen desde cualquier lugar, facilitando la colaboración global y el acceso a los recursos de manera segura [360].
- **Integración con herramientas modernas y *CI/CD***: las plataformas en la nube ofrecen integración nativa con herramientas de *CI/CD*, permitiendo despliegues continuos y un desarrollo ágil [361].

4.5.9. Impacto ecológico de desarrollar en la nube

El desarrollo en la nube tiene un impacto ecológico positivo al reducir el consumo de energía y el uso de *hardware* físico. Algunos de los efectos ecológicos de desarrollar en la nube incluyen [362]:

- **Optimización de recursos energéticos**: los centros de datos en la nube emplean técnicas avanzadas de enfriamiento y aprovechamiento de energías renovables, reduciendo su huella de carbono en comparación con instalaciones *on-premise*.
- **Eficiencia en el uso de infraestructura**: los proveedores de nube consolidan recursos, maximizando el uso de los servidores y minimizando el desperdicio de *hardware*.
- **Disminución de desechos electrónicos**: la infraestructura de nube reduce la necesidad de actualizaciones de *hardware* y evita que las empresas deban reemplazar equipos frecuentemente, disminuyendo los desechos electrónicos [363].

4.6. Marco teórico sobre seguridad

4.6.1. Seguridad cibernética en el desarrollo de aplicaciones

Importancia de la seguridad cibernética

La seguridad cibernética es el conjunto de procesos y herramientas utilizados para proteger de forma anticipada y defender dispositivos, sistemas electrónicos, servidores, redes y programas frente a posibles ciberataques. Con el mundo cada vez más digitalizado, la ciberseguridad se ha vuelto indispensable para la protección de la información personal y empresarial ante amenazas cibernéticas. La creciente interconexión de dispositivos y sistemas incrementa la vulnerabilidad frente a ataques como el robo de datos, *hacking* y *ransomware* [364].

Existen principalmente tres tipos de seguridad informática:

1. **Seguridad de *hardware*:** se refiere a la protección de los dispositivos que se utilizan para proteger sistemas, redes, aplicaciones y programas. Entre los métodos más empleados se encuentran los sistemas de alimentación ininterrumpida (**SAI**), servidores *proxy*, cortafuegos (***firewalls***), módulos de seguridad de *hardware* (**HSM**) y los sistemas de prevención de pérdida de datos (**DLP**). Esta protección no solo incluye el entorno virtual, sino también la seguridad física de los equipos frente a posibles daños.
2. **Seguridad de *software*:** está enfocada en defender los sistemas contra ataques cibernéticos que puedan aprovechar vulnerabilidades en los programas o configuraciones incorrectas de los mismos.

3. **Seguridad de red:** su objetivo es proteger los datos accesibles a través de la red, evitando que sean modificados o robados. Entre las principales amenazas se encuentran los virus, troyanos y programas espía [365].

En la seguridad cibernética, existen tres pilares fundamentales:

- **Confidencialidad:** garantiza que únicamente los usuarios autorizados puedan acceder a los recursos, datos e información.
- **Integridad:** se enfoca en asegurar que los datos no sean alterados sin autorización y que solo los usuarios con permisos puedan realizar dichas modificaciones.
- **Disponibilidad:** asegura que los datos siempre estén disponibles para su uso cuando se necesiten [365].

Los ciberdelincuentes emplean diversos métodos de ataque, entre los cuales destacan los siguientes:

***Malware*:** este es el tipo de ataque más común y hace referencia a un *software* malicioso (*malicious software*), que suele provenir de archivos descargados. El *malware* permite a los delincuentes acceder a redes y dañar, bloquear o robar información de los equipos de los usuarios.

***Phishing*:** es un ataque donde se suplanta la identidad de una persona o entidad. Consiste en crear copias de sitios *web* o enviar correos electrónicos que solicitan información confidencial, como contraseñas o datos de tarjetas de crédito, haciéndose pasar por instituciones legítimas.

***Man in the middle*:** en este ataque, el delincuente intercepta la comunicación entre dos personas y roba la información que están intercambiando.

Ataques de denegación de servicio distribuido (*DDoS*): este tipo de ataque busca saturar o colapsar un servidor o página *web* mediante el envío masivo de falsas solicitudes, lo que impide su funcionamiento normal.

Amenaza interna: este tipo de ataque suele ser llevado a cabo por ex empleados que aún tienen acceso a los sistemas de la empresa y utilizan esa ventaja para obtener datos confidenciales o infectarlos con *malware* [364].

Para mitigar estos riesgos, las empresas pueden seguir las siguientes recomendaciones:

- **Educar a los empleados sobre los riesgos cibernéticos:** la formación del personal es fundamental para que los empleados conozcan los riesgos a los que están expuestos y puedan identificar posibles amenazas, como correos electrónicos de *phishing* o enlaces sospechosos.
- **Implementar políticas de seguridad adecuadas:** es necesario establecer políticas claras que definan cómo debe gestionarse la seguridad en la empresa, desde el uso de contraseñas seguras hasta la correcta manipulación de la información confidencial.
- **Utilizar *software* de seguridad actualizado:** mantener el *software* de seguridad actualizado es crucial para protegerse contra las amenazas más recientes. Los sistemas obsoletos pueden tener vulnerabilidades que los ciberdelincuentes pueden aprovechar.
- **Cifrar los datos sensibles:** el cifrado de datos asegura que, aunque la información sea interceptada por terceros, no pueda ser leída sin la clave adecuada. Esto es especialmente importante para proteger datos sensibles como información financiera o personal.

- **Implementar un control de acceso y autenticación multifactor:** limitar el acceso a los sistemas y datos solo a los usuarios autorizados es esencial. Además, el uso de autenticación multifactor agrega una capa adicional de seguridad al requerir más de un método de verificación.
- **Realizar auditorías y evaluaciones de riesgos periódicas:** es importante llevar a cabo revisiones regulares para identificar posibles vulnerabilidades en los sistemas y evaluar si las medidas de seguridad actuales son adecuadas o necesitan ser mejoradas.
- **Tener un plan de respuesta ante incidentes:** contar con un plan de acción en caso de un ataque cibernético es fundamental para minimizar el impacto. Esto incluye procedimientos para la recuperación de datos y la contención del incidente.
- **Mantener copias de seguridad de los datos:** las copias de seguridad periódicas garantizan que, en caso de pérdida o robo de información, los datos puedan ser recuperados sin mayores problemas.
- **Cumplir con las normativas de seguridad:** las empresas deben seguir las regulaciones locales e internacionales en materia de protección de datos y ciberseguridad para evitar sanciones legales y garantizar la seguridad de la información.
- **Colaborar con expertos en ciberseguridad:** trabajar con profesionales del sector permite a las empresas tener acceso a las mejores prácticas y soluciones especializadas para enfrentar las amenazas más sofisticadas.
- **Monitorear continuamente los sistemas:** la supervisión constante de los sistemas permite detectar actividades sospechosas en tiempo real y tomar medidas inmediatas para evitar daños [364].

OWASP Top 10

El desarrollo de aplicaciones, en un entorno tecnológico cada vez más complejo y dinámico, enfrenta una serie de amenazas y vulnerabilidades que pueden comprometer la seguridad, integridad y disponibilidad de las aplicaciones y los datos que gestionan. Estas amenazas afectan tanto a las aplicaciones *web* como a las aplicaciones móviles, y pueden ser identificadas mediante el *OWASP Top 10*, que proporciona un listado de las vulnerabilidades más críticas en cada caso [366].

Según ***OWASP***, las 10 amenazas más comunes en aplicaciones web son las siguientes:

1. **Control de acceso interrumpido:** la falta de control adecuado puede provocar la divulgación no autorizada de información, la modificación o destrucción de datos, o la ejecución de funciones comerciales sin los permisos correspondientes.
2. **Fallos criptográficos:** la falta de cifrado adecuado en los datos sensibles, como contraseñas, números de tarjetas de crédito o registros médicos, puede exponer información confidencial y violar normativas como el Reglamento General de Protección de Datos (***GDPR***) o el Estándar de Seguridad de Datos *PCI* (***PCI DSS***).
3. **Inyección:** ocurre cuando no se validan adecuadamente los datos de entrada, permitiendo a los atacantes ejecutar comandos no autorizados y obtener acceso a información confidencial.
4. **Diseño inseguro:** se refiere a la falta de un diseño seguro desde el inicio del desarrollo de la aplicación, lo que provoca la ausencia de controles de seguridad necesarios para proteger frente a ataques específicos.

5. **Configuración incorrecta de seguridad:** errores en la configuración de la aplicación, el servidor o la infraestructura pueden dejar vulnerabilidades abiertas que los atacantes pueden aprovechar para comprometer la seguridad.
6. **Componentes vulnerables y obsoletos:** utilizar componentes desactualizados o con vulnerabilidades conocidas permite a los atacantes explotar esas debilidades.
7. **Errores de identificación y autenticación:** fallos en la identificación y autenticación de usuarios permiten a los atacantes suplantar la identidad de otros usuarios y acceder a funciones restringidas.
8. **Fallas en la integridad de datos y *software*:** la falta de validación en las actualizaciones de *software* puede permitir a los atacantes inyectar código malicioso o alterar datos críticos.
9. **Errores de registro y monitoreo de seguridad:** la falta de implementación adecuada de mecanismos de registro y monitoreo impide detectar y responder a amenazas de manera efectiva.
10. **Falsificación de solicitudes del lado del servidor (SSRF):** los atacantes pueden manipular el servidor para realizar solicitudes a otros servicios sin autorización, accediendo a datos sensibles sin necesidad de autenticación [366].

Además, *OWASP* también ha publicado un *Top 10* de riesgos específicos para aplicaciones móviles:

1. **Uso inadecuado de credenciales:** esta vulnerabilidad se refiere a la explotación de credenciales codificadas de manera inadecuada, lo que facilita el robo de datos.
2. **Seguridad inadecuada de la cadena de suministro:** los atacantes pueden aprovechar las vulnerabilidades en la cadena de suministro para manipular funcionalidades de la aplicación, acceder a aplicaciones móviles de terceros o al *backend* de los servidores, e incluso ejecutar ataques de denegación de servicio (**DoS**).
3. **Autenticación y autorización inseguras:** los atacantes pueden enviar solicitudes directamente al servidor *backend*, sin interactuar con la aplicación, obteniendo información o actuando con privilegios excesivos.
4. **Validación insuficiente de la entrada y salida de datos:** la falta de validación adecuada puede permitir inyecciones *SQL*, lo que resulta en el robo o manipulación de datos, o en la ejecución de código que interrumpa el funcionamiento de la aplicación.
5. **Comunicación insegura:** las fallas en los protocolos criptográficos permiten a los atacantes interceptar datos, suplantar identidades o robar información confidencial.
6. **Controles de privacidad inadecuados:** la gestión incorrecta de la información de los usuarios puede permitir a los atacantes realizar extorsiones, fraudes financieros o dañar la reputación de las víctimas.
7. **Protecciones insuficientes de los binarios:** los binarios pueden contener claves de acceso a *API* comerciales o ser utilizados para eludir controles de seguridad de la aplicación.
8. **Mala configuración de seguridad:** las configuraciones incorrectas de seguridad y permisos pueden ser explotadas por los atacantes para acceder a información confidencial o realizar acciones maliciosas.
9. **Almacenamiento de datos inseguro:** cuando los desarrolladores no almacenan correctamente los datos, los atacantes pueden comprometer cuentas, manipular información o acceder a archivos de configuración y claves criptográficas.

10. **Criptografía insuficiente:** el uso de cifrado débil o la falta total de cifrado puede resultar en la exfiltración de información confidencial, pérdidas financieras o problemas legales por incumplimiento de normativas [367].

Buenas prácticas de seguridad en el desarrollo de *software*

El término "buenas prácticas de seguridad"se refiere al conjunto de técnicas y metodologías que deben implementarse para que el desarrollo de *software* sea más eficiente, rápido y seguro. Estas prácticas ayudan a minimizar las vulnerabilidades desde el inicio del ciclo de vida del *software*, garantizando su calidad y protección frente a posibles ataques.

Existen varias formas de implementar buenas prácticas de seguridad, entre las que destacan:

COBIT (Control Objectives for Information and Related Technology*):** *COBIT* es un marco que permite evaluar y controlar todos los aspectos de la tecnología de la información (TI***) dentro de una organización, abarcando desde los sistemas de información hasta el personal que los gestiona. Su objetivo principal es proporcionar a las empresas una herramienta que facilite el cumplimiento de los objetivos del negocio a través de la automatización y optimización de los recursos *TI*.

***ITIL (Information Technology Infrastructure Library*):** *ITIL* es un conjunto de buenas prácticas para la gestión de servicios y recursos *TI*, cuyo objetivo es garantizar la calidad y eficiencia de los procesos productivos de una organización. *ITIL* divide el ciclo de vida de los servicios en cinco etapas:

- **Estrategia del servicio:** identifica los activos estratégicos de la empresa que generan valor o la diferencian en el mercado.
- **Diseño del servicio:** explora el portafolio de servicios para identificar debilidades y fortalezas, y establecer políticas y estándares que mejoren la calidad.
- **Transición del servicio:** cubre el proceso de cambio entre la mejora de servicios existentes o la creación de nuevos servicios, identificados durante la fase de diseño.
- **Operación del servicio:** gestiona y controla los procesos necesarios para la puesta en marcha del servicio en producción.
- **Mejora continua del servicio:** optimiza la calidad del servicio mediante la retroalimentación constante de las fases anteriores, facilitando la adaptación al entorno cambiante [368].

***Clean code*:** "*Clean code*"se refiere a código que es fácil de leer, entender y mantener. Este concepto promueve principios como el uso de nombres claros, evitar la duplicación de código, escribir funciones cortas y utilizar comentarios solo cuando sean realmente útiles. El objetivo es que el código no solo funcione bien, sino que sea comprensible para otros desarrolladores, lo que facilita su mantenimiento y mejora la calidad del *software* a largo plazo [369].

4.6.2. Contexto local: Guatemala

Situación actual de la gestión de residuos en Guatemala

En los últimos años, la población de Guatemala ha experimentado un crecimiento sostenido. Según datos del Banco Mundial, entre 1960 y 2021, el 20 % de la población rural migró hacia las zonas urbanas. En Guatemala, actualmente solo existen dos sitios principales para la gestión de

residuos sólidos: el relleno sanitario de la zona 3 y el vertedero de AMSA en Villa Nueva. Estos son los encargados de procesar los residuos generados por toda la población urbana [370].

En 2021, el Ministerio de Ambiente y Recursos Naturales (MARN) emitió el Acuerdo Gubernativo 164-2021: **Reglamento para la gestión integral de los residuos y desechos sólidos comunes**. Este reglamento establece normas sanitarias y ambientales que buscan prevenir el deterioro ambiental, reducir la contaminación y mejorar la salud de los guatemaltecos. Durante los primeros dos años desde la implementación del reglamento, es obligatoria la clasificación primaria de los desechos en orgánicos e inorgánicos. Además, se contempla una clasificación secundaria, que incluye categorías como reciclables (papel, cartón, vidrio, plástico, metal, multicapa) y no reciclables, cuyo cumplimiento será obligatorio a partir del tercer año de vigencia del acuerdo. Sin embargo, un cambio reciente, realizado el 9 de agosto de 2024, postergó la obligatoriedad de esta clasificación secundaria hasta el 11 de febrero de 2025, permitiendo por ahora el cumplimiento únicamente de la clasificación primaria [371].

Otra modificación significativa del reglamento se refiere al Artículo 17, que regula la recolección y transporte de residuos. Anteriormente, se exigía que los vehículos utilizados en estas tareas fueran construidos con materiales sólidos y resistentes a la corrosión, pero ahora el requisito se ha simplificado, solicitando únicamente protección contra la corrosión. Además, se exige que los vehículos sean herméticos para evitar el derrame de líquidos contaminantes durante el transporte [372].

El Artículo 28 del reglamento establece que las tecnologías empleadas para el tratamiento de los desechos sólidos deben seleccionarse en función de la naturaleza de los residuos y los objetivos de gestión. No es obligatorio aplicar un tratamiento previo antes de la disposición final si no es necesario. Por otro lado, el Artículo 29 especifica que el tratamiento de residuos puede incluir procesos como la incineración o la reducción mecánica de su tamaño mediante técnicas de fragmentación, molienda o trituración, lo que facilita su disposición final [371].

Por último, el Artículo 54 estipula que se impondrán multas a las entidades que, tras una verificación, no cumplan con los compromisos ambientales establecidos. Las sanciones pueden variar entre uno (1) y cuarenta (40) salarios mínimos mensuales por cada incumplimiento, dependiendo de la gravedad de la infracción, la cual será determinada por el MARN [371].

Es alentador que Guatemala esté tomando medidas concretas para abordar el tema de la gestión de residuos. A lo largo de los últimos años, se han implementado y mejorado reglamentos clave que buscan reducir la contaminación, proteger la salud pública y garantizar la sostenibilidad ambiental. Con la aplicación gradual de estos reglamentos, se espera que el país logre un manejo más eficiente y responsable de sus desechos, promoviendo una cultura de reciclaje y reducción de residuos. La postergación de ciertas medidas muestra flexibilidad en la adaptación de las normativas, pero también resalta la necesidad de compromiso por parte de la ciudadanía y las instituciones para lograr una gestión integral y efectiva en el largo plazo.

Políticas y regulaciones ambientales

En Guatemala, diversas entidades administrativas a nivel nacional son responsables de la gestión ambiental en diferentes áreas. La Comisión Nacional del Medio Ambiente (CONAMA) se encarga de asesorar, coordinar y aplicar la política nacional ambiental. Su estructura organizacional y funciones están reguladas por la Ley de Protección y Mejoramiento del Medio Ambiente.

El Artículo 97 de la Constitución de Guatemala establece que el estado, las municipalidades y los habitantes del territorio nacional están obligados a propiciar el desarrollo social, económico y tecnológico que prevenga la contaminación del ambiente y mantenga el equilibrio ecológico. Se dictarán todas las normas necesarias para garantizar que la utilización y el aprovechamiento de la fauna, de la flora, de la tierra y del agua se realicen racionalmente, evitando su depredación [373].

Por su parte, el Artículo 64 establece que "Se declara de interés nacional la conservación, protección y mejoramiento del patrimonio natural de la Nación. El Estado fomentará la creación de parques nacionales, reservas y refugios naturales, los cuales son inalienables. Una ley garantizará su protección y la de la fauna y la flora que en ellos exista" [373].

En cuanto a las políticas nacionales de protección del medio ambiente y desarrollo sostenible, a pesar de los esfuerzos realizados en sectores como el agropecuario y forestal, Guatemala aún carece de una política nacional del medio ambiente sancionada por las más altas instancias del Ejecutivo, tal como lo exige el Decreto 68-86. En el sector agropecuario, se han definido políticas orientadas al desarrollo sostenible con la participación de los actores involucrados. En el ámbito forestal, se busca maximizar los beneficios socioeconómicos de los recursos mediante la ampliación de la cobertura forestal y el manejo sostenible de los bosques. En cuanto a la protección de la diversidad biológica, aunque no existe una política explícita, se han desarrollado estrategias institucionales para la conservación y el uso sostenible de los recursos genéticos [373].

En cuanto a las regulaciones más recientes, el Acuerdo Gubernativo 189-2017 establece que las políticas públicas de Guatemala están orientadas a la protección, conservación y mejoramiento de los recursos naturales. La Política Nacional de Desarrollo Rural Integral (PNDRI) impulsa políticas sectoriales dirigidas al desarrollo rural, buscando mejorar la calidad de vida de las poblaciones en situación de pobreza y exclusión. Por su parte, la Política Marco de Gestión Ambiental se enfoca en mejorar la calidad ambiental y conservar el patrimonio natural, garantizando el equilibrio ecológico necesario para la vida y promoviendo el bienestar social, económico y cultural de las generaciones actuales y futuras [374].

Otra política importante es la Política Nacional de Cambio Climático, que tiene como objetivo reducir las vulnerabilidades, promover la adaptación al cambio climático y mitigar las emisiones de gases de efecto invernadero. Esta política también integra la educación sobre cambio climático en los sistemas educativos formales e informales [374].

Asimismo, la Política Nacional para el Manejo Integral de los Residuos y Desechos Sólidos establece lineamientos claros para garantizar la salud pública a través del manejo adecuado de los residuos, fomentando prácticas de producción más limpia. De igual manera, la Política Nacional para la Gestión Ambientalmente Racional de Productos Químicos y Desechos Peligrosos busca reducir los riesgos ambientales y de salud derivados de la gestión inadecuada de productos químicos [374].

Por otro lado, la Política para el Manejo Integral de las Zonas Marino Costeras se centra en la mejora de la calidad de vida de las comunidades costeras, mientras que la Política Ambiental de Género y su Plan de Acción promueven la equidad de género y la inclusión de hombres y mujeres en la protección de los bienes y servicios ambientales [374].

Desafíos y oportunidades en la seguridad cibernética en Guatemala

El aumento significativo de los ciberataques en Guatemala ha puesto de relieve la necesidad de implementar soluciones integrales de seguridad cibernética, especialmente en un contexto de rápida transformación digital. Con un promedio de 1,727 ataques por semana, las organizaciones guatemaltecas, particularmente en el sector financiero, se han convertido en objetivos frecuentes de estos ataques. Según Eli Faskha, *CEO* de Soluciones Seguras, la digitalización no gestionada adecuadamente puede aumentar significativamente los riesgos de ciberseguridad. En este contexto, es crucial que las empresas identifiquen sus vulnerabilidades y aprendan a detectar, protegerse y responder a las amenazas cibernéticas para garantizar la continuidad de sus operaciones [375].

En el marco legal, Guatemala cuenta con disposiciones específicas para sancionar delitos informáticos a través de su Código Penal. Entre los artículos más relevantes se encuentran:

- **Artículo 274 A:** sanciona con prisión de seis meses a cuatro años y una multa de doscientos a dos mil quetzales a quien destruya, borre o inutilice registros informáticos.
- **Artículo 274 B:** aplica la misma pena a quien altere, borre o inutilice instrucciones o programas utilizados por computadoras.
- **Artículo 274 C:** impone prisión de seis meses a cuatro años y una multa de quinientos a dos mil quinientos quetzales a quien, sin autorización, copie o reproduzca instrucciones o programas de computación.
- **Artículo 274 D:** establece una pena de seis meses a cuatro años de prisión y una multa de doscientos a mil quetzales para quien cree un banco de datos o registro informático que afecte la intimidad de las personas.
- **Artículo 274 E:** sanciona con prisión de uno a cinco años y una multa de quinientos a tres mil quetzales a quien utilice registros informáticos para ocultar, alterar o distorsionar información en actividades comerciales.
- **Artículo 274 F:** impone prisión de seis meses a dos años y una multa de doscientos a mil quetzales a quien acceda sin permiso a bancos de datos o archivos electrónicos ajenos [376].

A pesar de las sanciones contempladas en el Código Penal, estas leyes resultan insuficientes frente al creciente número de ataques cibernéticos. La digitalización en el país ha evidenciado la necesidad urgente de actualizar y fortalecer las normativas legales en esta área. Aunque existen penalizaciones, no disuaden de manera efectiva a los ciberdelincuentes ni protegen adecuadamente a las empresas y usuarios guatemaltecos. El desarrollo de leyes más robustas y modernas es fundamental para hacer frente a las amenazas cibernéticas actuales y emergentes.

En 2022, se presentó el Decreto 39-2022, que proponía la Ley de prevención y protección contra la ciberdelincuencia. Sin embargo, esta normativa fue finalmente archivada debido a la controversia que generaron algunos de sus artículos. En particular, el Artículo 9, que penalizaba el acceso no autorizado a información confidencial, fue criticado por contravenir el Artículo 35 de la Constitución, que garantiza el acceso a fuentes de información. Asimismo, el Artículo 19, que sancionaba el ciberacoso y la divulgación de información, fue considerado ambiguo, lo que planteaba el riesgo de que pudiera utilizarse para criminalizar críticas a funcionarios públicos o restringir la libertad de expresión [377].

Estas disposiciones planteaban el peligro de que la ley fuera utilizada para silenciar a los medios de comunicación, periodistas y ciudadanos que ejercen su derecho a la libre expresión. La posibilidad de que investigaciones periodísticas o críticas en redes sociales, como memes o sátira política, fueran tratadas como delitos, despertó preocupación sobre su impacto en el ejercicio periodístico y la participación cívica. A pesar de que la ley buscaba mejorar la ciberseguridad en Guatemala, fue percibida como una amenaza a los derechos fundamentales, lo que finalmente llevó a su archivo el 24 de agosto de 2022 [377].

Es evidente que el fortalecimiento de la ciberseguridad en Guatemala presenta desafíos significativos, no solo en términos tecnológicos, sino también en el marco legal y regulatorio. Sin embargo, esta situación también representa una oportunidad para que el país desarrolle normativas más sólidas y protectoras, que equilibren la seguridad con la preservación de los derechos fundamentales de sus ciudadanos.

4.6.3. Metodologías de desarrollo seguro

Introducción al desarrollo seguro de *software*

El desarrollo seguro de *software* coloca la seguridad como un requisito en todas las fases del Ciclo de Vida del Desarrollo de *Software* (***SDLC***, por sus siglas en inglés). Desde la fase de diseño hasta la implementación y el mantenimiento, el objetivo principal es identificar, mitigar y prevenir vulnerabilidades que puedan afectar la integridad del *software* y la seguridad de los usuarios.

Uno de los aspectos más críticos de un modelo de desarrollo seguro es que, según *IBM Security*, el 60 % de los ataques cibernéticos provienen de vulnerabilidades presentes en el código fuente [378]. La falta de un enfoque de seguridad desde el inicio del proyecto puede resultar en una integración deficiente de las medidas de protección durante el proceso de desarrollo. Esto no solo hace que sea más complejo y costoso aplicar correcciones de seguridad al final de las fases, sino que también puede generar pérdidas económicas significativas y la pérdida de confianza por parte de los clientes debido a incidentes de seguridad.

Existen varias metodologías y enfoques que complementan el Desarrollo Seguro de *Software* (**SSD**), cada una de ellas con un enfoque específico para abordar la seguridad en diferentes momentos del ciclo de desarrollo:

- **Ciclo de vida de desarrollo de seguridad (*SecDevOps*):** esta metodología incorpora herramientas y prácticas de seguridad en el ciclo de desarrollo continuo. *SecDevOps* incluye la automatización de pruebas de seguridad y proporciona retroalimentación rápida para identificar y corregir vulnerabilidades a lo largo del proceso de desarrollo.
- **Desarrollo seguro por diseño (*SSDLC*):** el *SSDLC* se enfoca en integrar la seguridad desde la etapa de diseño, incorporando controles de seguridad en los requisitos iniciales y en la arquitectura del *software*, lo que ayuda a prevenir vulnerabilidades desde las primeras fases del proyecto.
- **Modelo de madurez de seguridad (*SMM*):** esta metodología evalúa y mejora la madurez de la seguridad en el entorno de desarrollo. Proporciona un marco para identificar el estado actual de seguridad, analizar los puntos débiles y establecer metas claras de mejora en la seguridad del *software*.
- **Metodologías ágiles y seguridad (*agile security*):** a través de revisiones continuas y en cada iteración del ciclo de desarrollo ágil, se incorporan análisis de seguridad de código, pruebas de penetración y ajustes necesarios para abordar nuevas amenazas de manera proactiva.
- **Análisis de amenazas y modelado (*TAM*):** el *TAM* identifica amenazas y riesgos desde las etapas iniciales del desarrollo mediante el análisis de riesgos y el modelado de amenazas, lo que permite a los equipos de desarrollo anticipar posibles vulnerabilidades y diseñar estrategias de mitigación tempranas [379].

Cada una de estas metodologías tiene como objetivo garantizar que el *software* no solo sea funcional, sino que también esté protegido frente a amenazas y vulnerabilidades. Al implementar estas prácticas desde las primeras fases del *SDLC*, se logra un entorno de desarrollo más seguro, con menos errores críticos que podrían afectar la continuidad y seguridad de las operaciones.

Ciclo de vida del desarrollo de *software* seguro (*SDLC*)

El Ciclo de Vida del Desarrollo de *Software* (***SDLC***, por sus siglas en inglés) es un proceso estructurado que permite a los equipos de desarrollo diseñar y crear *software* de alta calidad de

manera rentable y eficiente. El objetivo principal del *SDLC* es reducir los riesgos del proyecto a través de una planificación anticipada, garantizando que el *software* cumpla con las expectativas del cliente durante su fase de producción y a lo largo de su vida útil. Este enfoque divide el desarrollo de *software* en una serie de fases bien definidas, que pueden ser asignadas, completadas y medidas de manera efectiva [380].

Generalmente, el *SDLC* se compone de seis a ocho fases:

- **Planificación:** en esta fase se recopilan los requisitos de expertos, directivos y clientes para crear un documento que especifique los objetivos del *software*. También se realizan análisis de costos, estimación de recursos y beneficios, y se asignan roles dentro del equipo de desarrollo.
- **Análisis de los requisitos:** se definen las funciones que debe ejecutar el *software*, asegurando que estas se alineen con las necesidades del cliente.
- **Diseño:** se analizan las especificaciones técnicas del *software* y se define la mejor forma de integrarlo en la infraestructura de tecnología de la información (***TI***) existente en la empresa.
- **Desarrollo:** esta fase incluye la creación y codificación del *software* conforme a los lineamientos establecidos en las fases previas.
- **Documentación:** se produce toda la información necesaria para que los usuarios y las partes interesadas puedan utilizar el sistema de manera efectiva.
- **Pruebas:** se llevan a cabo pruebas automáticas y manuales para verificar que el *software* cumple con los requisitos establecidos y no presenta fallos de seguridad. A menudo, las pruebas se realizan simultáneamente con el desarrollo.
- **Implementación:** el *software* es desplegado en el entorno de producción, haciéndolo accesible a los usuarios finales.
- **Mantenimiento:** después de la implementación, se ofrece soporte continuo para resolver problemas, supervisar el rendimiento, mejorar la experiencia del usuario y reforzar la ciberseguridad [381].

Para garantizar la seguridad del *software* a lo largo de todas las fases del *SDLC*, se debe seguir el enfoque del Ciclo de Vida de Desarrollo de *Software* Seguro (***SSDLC***), que incorpora medidas de seguridad específicas en cada etapa del proceso. Estas medidas ayudan a identificar y mitigar vulnerabilidades antes de que afecten el rendimiento o la seguridad del *software*.

La integración de estas medidas de seguridad a lo largo del *SDLC* asegura que el *software* no solo cumpla con sus requisitos funcionales, sino que también esté protegido contra amenazas emergentes y riesgos cibernéticos [382].

Principios de diseño seguro

Los principios de diseño seguro son esenciales para asegurar que las aplicaciones y los sistemas sean resistentes frente a vulnerabilidades y amenazas, protegiendo la confidencialidad, integridad y disponibilidad de los datos. Estos principios ofrecen un marco claro para que los desarrolladores construyan aplicaciones más seguras y confiables, alineando las mejores prácticas de seguridad desde el diseño inicial hasta la implementación final.

Uno de los principios más relevantes es el de menor privilegio, que establece que un usuario o proceso solo debe tener los permisos mínimos necesarios para realizar sus tareas, lo que minimiza el riesgo de uso indebido de esos privilegios. La reducción de permisos disminuye la exposición a

Etapa del SDLC	**Medidas de seguridad**
Planificación	Evaluar los riesgos y el panorama de amenazas a la seguridad. Evaluar el impacto potencial de los incidentes de seguridad, como los riesgos para la reputación de la empresa.
Análisis de los requisitos	Incluir requisitos de seguridad. Asegurarse de cumplir con las normativas aplicables.
Diseño	Elaborar modelos de amenazas. Incorporar consideraciones de seguridad en el plan de arquitectura. Evaluar cómo las decisiones de diseño afectan la seguridad, como la elección de la plataforma y la interfaz de usuario.
Desarrollo	Capacitar a los desarrolladores en prácticas de codificación seguras. Utilizar herramientas de pruebas de seguridad a lo largo del proceso de desarrollo. Evaluar las dependencias del *software* y mitigar los riesgos de seguridad asociados.
Documentación	Documentar los procesos y controles de seguridad. Prepararse para auditorías, controles de cumplimiento normativo y revisiones de seguridad.
Pruebas	Implementar revisiones de código. Realizar pruebas estáticas y dinámicas de seguridad de las aplicaciones.
Implementación	Evaluar la seguridad del entorno de producción. Revisar las configuraciones de seguridad.
Mantenimiento	Supervisar el sistema en busca de amenazas. Estar preparado para corregir vulnerabilidades y gestionar intrusiones.

Cuadro 4.6. Etapas del SDLC y medidas de seguridad.

posibles ataques y garantiza que, en caso de comprometerse un usuario o sistema, el impacto sea limitado [383].

Otro principio clave es la economía y simplificación de los mecanismos de seguridad. Este principio sugiere que los mecanismos de seguridad deben ser lo más simples posible para reducir la posibilidad de errores y, a su vez, garantizar una mayor eficacia. Al reducir la complejidad del sistema, se mejora la capacidad de detectar y corregir errores de seguridad. Además, las configuraciones por defecto seguras garantizan que un sistema esté protegido desde el principio, asegurando que, si un acceso no está explícitamente permitido, esté prohibido por defecto. Esto refuerza la idea de limitar accesos no autorizados [383].

El principio de diseño abierto es igualmente fundamental. Establece que la seguridad de un sistema no debe depender del secreto de su diseño, sino de la solidez de su implementación. Esto promueve la transparencia y permite que el diseño de seguridad sea revisado por la comunidad, lo que ayuda a identificar debilidades que podrían ser pasadas por alto en un entorno cerrado. La seguridad no debe basarse en el secretismo, sino en la robustez del diseño [384].

La mediación completa es otro principio clave que implica que todos los accesos a recursos deben ser verificados, independientemente de los permisos previamente concedidos o almacenados en caché. Esto asegura que cualquier cambio en los privilegios o en el contexto del usuario sea aplicado de manera inmediata y consistente, evitando accesos indebidos.

El enfoque de *MICROSOFT WELL-ARCHITECTED* se alinea con estos principios, promoviendo la filosofía de confianza cero (***zero trust***), en la que cada interacción dentro del sistema debe ser verificada explícitamente, sin suposiciones sobre la confiabilidad de los usuarios o dispositivos. Este enfoque ayuda a limitar el impacto de un ataque al restringir los permisos y minimizando el radio de explosión en caso de una violación de seguridad. Además, el enfoque de *Microsoft* refuerza que la seguridad no es un proceso único, sino continuo, y debe ser revisado y mejorado a lo largo del

tiempo, adaptándose a nuevos desafíos y ataques [385].

Finalmente, la aceptación psicológica también es crucial para el diseño seguro. Los mecanismos de seguridad deben ser fáciles de usar y no interferir de manera significativa con la usabilidad del sistema. Esto asegura que los usuarios no busquen eludir las medidas de seguridad, sino que las adopten de forma natural. Los sistemas deben ser tan accesibles con medidas de seguridad como lo serían sin ellas, promoviendo así su aceptación entre los usuarios finales [384].

4.6.4. *OWASP SAMM (Software Assurance Maturity Model)*

Introducción a *OWASP SAMM*

El Modelo de Madurez de Aseguramiento de *Software* (*OWASP SAMM*, por sus siglas en inglés) es un marco desarrollado por *OWASP* diseñado para ayudar a las organizaciones a evaluar, formular e implementar una estrategia de seguridad en el desarrollo de *software* que pueda integrarse de manera efectiva en su ciclo de vida del desarrollo de *software* (***SDLC***). Este modelo es flexible y se adapta a diversos contextos organizacionales, independientemente de si el *software* es desarrollado internamente, externalizado o adquirido. Además, *OWASP SAMM* es aplicable a distintas metodologías, ya sean tradicionales, ágiles o basadas en *DevOps*.

El principal objetivo de *OWASP SAMM* es proporcionar una guía clara que permita a las organizaciones evaluar su postura actual en cuanto a seguridad del *software*, definir una estrategia de mejora y establecer una hoja de ruta para la implementación de prácticas seguras. Esto facilita que las organizaciones puedan progresar de forma iterativa en la adopción de mejores prácticas, logrando un enfoque más maduro y robusto para el aseguramiento del *software*.

Una de las características más destacadas de *OWASP SAMM* es su flexibilidad, lo que permite que organizaciones de cualquier tamaño personalicen y adopten el modelo según sus necesidades específicas. *OWASP SAMM* organiza sus principios en 15 prácticas de seguridad, distribuidas en cinco funciones empresariales clave. Estas funciones cubren desde la gobernanza hasta la implementación y operación de *software*. Cada una de las prácticas de seguridad está estructurada en tres niveles de madurez, que representan un progreso iterativo y constante hacia metas más sofisticadas y métricas de éxito más estrictas.

A medida que las organizaciones avanzan en los niveles de madurez, pueden medir su progreso y ajustar sus estrategias de acuerdo con sus capacidades y recursos. Es importante destacar que *OWASP SAMM* no exige alcanzar el nivel máximo en todas las categorías, sino que permite a las organizaciones establecer sus propios objetivos de acuerdo con sus necesidades y prioridades. Esto asegura que cada entidad pueda avanzar en su proceso de aseguramiento de *software* de una manera controlada y adaptada a su contexto operativo [386].

Objetivos de *OWASP SAMM*

El *OWASP Software Assurance Maturity Model (SAMM)* es un marco desarrollado por *OWASP* que permite a las organizaciones mejorar la seguridad de su *software* de manera estructurada y controlada. A través de *OWASP SAMM*, las empresas pueden evaluar sus prácticas actuales, identificar puntos débiles en sus procesos de desarrollo de *software* y establecer estrategias para avanzar en su madurez de seguridad. Este modelo integra la seguridad en cada fase del ciclo de vida del desarrollo de *software* (***SDLC***), lo que es esencial en un entorno donde las amenazas cibernéticas evolucionan constantemente y donde la ciberseguridad debe ser una prioridad tanto para el desarrollo de *software* como para la adquisición de soluciones de terceros [387].

OWASP SAMM tiene varios objetivos clave, entre los que destacan:

- **Evaluar la postura de seguridad actual:** mediante la evaluación de las prácticas de seguridad vigentes, las organizaciones pueden obtener una visión clara de su nivel de madurez en cuanto a seguridad del *software*. Esto les permite identificar en qué áreas deben mejorar para fortalecer su seguridad.
- **Definir una estrategia de seguridad:** el modelo proporciona una guía detallada para que las empresas puedan establecer una estrategia de seguridad alineada con sus objetivos y necesidades específicas. Esta estrategia considera los recursos disponibles, el nivel de riesgo de la organización y los requisitos normativos que deben cumplir.
- **Establecer un plan de acción y una hoja de ruta:** *OWASP SAMM* facilita la creación de una hoja de ruta con acciones específicas, divididas en fases y plazos, para implementar mejoras en las prácticas de seguridad. Estas actividades se estructuran en tres niveles de madurez, lo que permite a las organizaciones avanzar de manera progresiva y medible en la seguridad de su *software*.

Una de las principales ventajas de *OWASP SAMM* es su flexibilidad. El modelo está diseñado para adaptarse a cualquier tipo de organización, ya sea pequeña, mediana o grande, y puede implementarse independientemente del enfoque de desarrollo utilizado, ya sea tradicional, ágil o *DevOps*. Además, *OWASP SAMM* permite la personalización, lo que facilita que cada organización defina su propio nivel de madurez deseado para cada práctica de seguridad. Esta capacidad de ajuste convierte a *OWASP SAMM* en una herramienta poderosa para la optimización continua de la seguridad del *software*, especialmente en un entorno en el que los ciberataques son cada vez más sofisticados y perjudiciales [387].

Componentes y prácticas de *SAMM*

El *OWASP Software Assurance Maturity Model (SAMM)* está estructurado en torno a cinco funciones de negocio clave, cada una de las cuales abarca un conjunto de prácticas de seguridad. Estas funciones y prácticas forman el núcleo del modelo y están diseñadas para guiar a las organizaciones en la implementación de medidas de seguridad a lo largo del ciclo de vida del *software* (***SDLC***). Cada función tiene asociadas tres prácticas de seguridad específicas, que a su vez están organizadas en diferentes niveles de madurez. Este enfoque modular permite a las organizaciones avanzar progresivamente en su estrategia de seguridad, adaptándose a sus necesidades y capacidades [387].

Funciones de negocio en *SAMM*

Las funciones de negocio de *SAMM* cubren las principales áreas de seguridad que deben ser consideradas en cualquier organización:

- **Gobernanza (G):** se enfoca en la gestión de la seguridad a nivel organizativo, incluyendo la estrategia de seguridad, el cumplimiento normativo y la formación. Esta función asegura que la seguridad esté integrada en los procesos empresariales desde el nivel más alto.
- **Diseño (D):** aborda la seguridad durante el diseño del *software*, desde los requisitos hasta la arquitectura. El objetivo es asegurar que el *software* se construya con medidas de seguridad adecuadas desde sus cimientos.
- **Implementación (I):** se enfoca en la construcción y despliegue del *software*, asegurando que el código sea seguro y que se minimicen los defectos antes de su lanzamiento.
- **Verificación (V):** cubre las pruebas de seguridad a lo largo del ciclo de vida del *software*, incluyendo análisis de seguridad y pruebas basadas en requisitos para detectar vulnerabilidades antes de que lleguen a producción.

- **Operaciones (O):** asegura que la seguridad esté garantizada durante la operación del *software* en producción, preservando la confidencialidad, integridad y disponibilidad de los datos.

Prácticas de seguridad en *SAMM*

Cada función de negocio contiene tres prácticas de seguridad específicas, lo que da un total de 15 prácticas de seguridad dentro del marco *SAMM*. A continuación, se mencionan algunos ejemplos clave:

- **Gobernanza (G):** estrategia y métricas; Política y cumplimiento; Formación y orientación.
- **Diseño (D):** evaluación de amenazas; Requisitos de seguridad; Arquitectura de seguridad.
- **Implementación (I):** construcción segura; Implantación segura; Gestión de fallos.
- **Verificación (V):** pruebas de seguridad; Revisión de código; Evaluación de vulnerabilidades.
- **Operaciones (O):** gestión de incidentes; Monitoreo de seguridad; Respuesta a incidentes.

Flujos y niveles de madurez

Cada práctica de seguridad dentro de *SAMM* se divide en dos flujos que representan diferentes aspectos de la práctica. Por ejemplo, en la práctica de implantación segura, uno de los flujos es la gestión de secretos, que incluye medidas para proteger las credenciales y otros secretos críticos.

Además, las actividades dentro de cada flujo están estructuradas en tres niveles de madurez, lo que permite a las organizaciones avanzar de manera progresiva en la implementación de medidas de seguridad:

- **Nivel 1:** actividades más básicas y fáciles de implementar, como la formalización de procesos.
- **Nivel 2:** introducción de automatizaciones y pruebas de seguridad más avanzadas.
- **Nivel 3:** verificación automatizada de la seguridad en todos los componentes, tanto internos como externos.

Actividades y progresión

OWASP SAMM propone hasta 90 actividades distribuidas a lo largo de las prácticas y niveles de madurez, lo que permite a las organizaciones avanzar de manera iterativa en su seguridad. Las actividades están diseñadas para mejorar la seguridad de forma estructurada y controlada. Por ejemplo, en el flujo de implantación segura, el Nivel 1 incluye la formalización de procesos de despliegue, el Nivel 2 automatiza las verificaciones de seguridad durante el despliegue y el Nivel 3 asegura la integridad total del *software* desplegado, independientemente de su origen (interno o externo).

Con estos componentes y prácticas, *OWASP SAMM* proporciona una estructura completa que permite a las organizaciones evaluar, planificar e implementar medidas de seguridad ajustadas a sus necesidades y nivel de madurez. La modularidad del modelo asegura que cada empresa pueda enfocarse en las prácticas más relevantes para su contexto y mejorar gradualmente su postura de seguridad [387].

4.6.5. Herramientas de análisis de seguridad

SonarQube

Funcionalidades principales

¿Qué es *SonarQube*?

SonarQube es una herramienta de análisis estático de código diseñada para evaluar y mejorar la calidad del código fuente de un proyecto. Su principal objetivo es identificar problemas comunes que puedan afectar la eficiencia y la seguridad del código, tales como *bugs*, vulnerabilidades, malas prácticas de programación y *code smells*. Gracias a su capacidad para realizar análisis exhaustivos, *SonarQube* se ha convertido en una herramienta esencial para garantizar que el código fuente sea seguro y de alta calidad [388].

Características de *SonarQube*

SonarQube cuenta con una serie de características que lo convierten en una herramienta indispensable para los desarrolladores de *software*. A continuación, se detallan las más importantes:

- **Análisis de calidad del código:** *SonarQube* realiza un análisis profundo del código fuente y proporciona métricas claras sobre su calidad. Permite detectar problemas relacionados con la complejidad ciclomática, la duplicación de código y el incumplimiento de los estándares de codificación. Esto ayuda a los desarrolladores a comprender mejor las debilidades de su código y a mejorar su estructura.
- **Detección de *bugs* y vulnerabilidades:** *SonarQube* identifica posibles errores y vulnerabilidades que podrían afectar el rendimiento y la seguridad de la aplicación, tales como riesgos del *OWASP Top 10*, incluyendo inyecciones y problemas de autenticación. Esta capacidad es fundamental para asegurar que el *software* esté libre de problemas graves antes de su implementación, contribuyendo a la estabilidad y seguridad del proyecto [389].
- ***Security hotspot detection*:** permite detectar áreas del código que requieren revisión adicional sin ser vulnerabilidades inmediatas, tales como el manejo de permisos y el uso de datos sensibles. *SonarQube* destaca estas secciones para una revisión manual, garantizando que se sigan las mejores prácticas de seguridad [389].
- **Detección de secretos codificados:** *SonarQube* escanea el código en busca de información sensible, como claves *API*, contraseñas o certificados, y proporciona sugerencias para su almacenamiento seguro, evitando exposiciones accidentales de información [389].
- **Detección de *cross-site scripting (XSS)*:** detecta y previene vulnerabilidades *XSS* en aplicaciones *web*, escaneando entradas no validadas y sugiriendo prácticas de codificación segura para mitigar riesgos [389].
- **Prevención de *SQL injection*:** detecta y marca áreas donde las consultas *SQL* pueden ser vulnerables a inyecciones, recomendando consultas parametrizadas o el uso de *ORM* para reducir riesgos [389].
- **Detección de duplicación de código:** identifica y destaca secciones duplicadas en la base de código, promoviendo la refactorización para mejorar la consistencia y reducir las vulnerabilidades asociadas [389].
- **Detección de vulnerabilidades en librerías de código abierto:** *SonarQube* analiza componentes de código abierto y dependencias en el proyecto, cruzando las versiones usadas con vulnerabilidades conocidas y sugiriendo actualizaciones o parches [389].

- **Seguridad de *API*:** analiza los *endpoints de API* para detectar problemas de autenticación, entradas no validadas y exposición de datos sensibles, generando informes centrados en vulnerabilidades de *API* [389].
- **Protección contra *buffer overflow*:** detecta operaciones de manejo de memoria inseguras, como límites de arreglo no verificados o copias excesivas de datos, proporcionando recomendaciones para evitar *exploits* de desbordamiento de búfer [389].
- **Prevención de ejecución remota de código (RCE):** escanea el código para detectar deserialización insegura y problemas de validación de entrada que puedan llevar a la ejecución de código remoto, destacando el código que requiere una revisión cuidadosa [389].
- **Cobertura de código para pruebas de seguridad:** permite hacer un seguimiento de métricas de cobertura de código para pruebas de seguridad, resaltando las secciones no cubiertas que pueden representar riesgos y proporcionando información para mejorar la cobertura de las pruebas [389].
- **Detección de *code smells*:** *SonarQube* identifica y recomienda la refactorización de código que presenta *code smells* o prácticas deficientes que pueden llevar a problemas de seguridad y mantenimiento a largo plazo [389].

Estas características no solo permiten que *SonarQube* detecte problemas, sino que también proporcionan soluciones y recomendaciones claras para mejorar el código y asegurar que cumpla con los estándares más altos de calidad y seguridad [390].

¿Por qué usar *SonarQube*?

El uso de *SonarQube* ofrece una serie de beneficios clave para los desarrolladores:

- **Mejorar la calidad del código:** *SonarQube* proporciona informes detallados sobre la calidad del código, lo que permite a los desarrolladores identificar y corregir problemas de calidad y seguridad antes de que lleguen a producción.
- **Ahorro de tiempo y esfuerzo:** al automatizar el proceso de análisis del código, *SonarQube* ayuda a ahorrar tiempo y esfuerzo en la identificación de problemas de calidad. Los desarrolladores pueden centrarse en la corrección de errores y la mejora continua sin necesidad de realizar análisis manuales exhaustivos.
- **Garantizar la seguridad y estabilidad de la aplicación:** *SonarQube* es fundamental para asegurar que la aplicación sea segura y estable, lo que a su vez garantiza la satisfacción del usuario final y la protección de la reputación de la empresa que desarrolla el *software*.

Cómo implementar *SonarQube* en tu proyecto

Implementar *SonarQube* en un proyecto es un proceso sencillo que puede integrarse de manera efectiva en los flujos de trabajo existentes. A continuación, se describen los pasos básicos para configurar *SonarQube*:

1. **Instalación:** descarga e instala la versión más reciente de *SonarQube*. Se recomienda revisar los requisitos de sistema para asegurarse de que la infraestructura del proyecto pueda soportar la herramienta.
2. **Integración en *CI/CD*:** configura la integración de *SonarQube* con tu plataforma de *CI/CD* para asegurar que el análisis de calidad del código se realice automáticamente en cada *commit* o despliegue.

3. **Análisis de código:** ejecuta los análisis de código y revisa los informes generados. Estos informes identificarán problemas clave y proporcionarán recomendaciones específicas para mejorar el código.
4. **Corrección de problemas:** usa los resultados del análisis para guiar el proceso de corrección de errores y mejoras en la calidad del código.

La implementación de *SonarQube* desde las primeras fases del proyecto asegura que el código cumpla con los más altos estándares de calidad y seguridad desde el inicio. Esto, a su vez, permite a los desarrolladores detectar y corregir problemas de manera temprana, evitando posibles complicaciones en etapas avanzadas del desarrollo [389, 390].

Beneficios y limitaciones

Beneficios

El uso de *SonarQube* en este proyecto ofrece varios beneficios clave:

- **Identificación temprana de problemas:** la capacidad de detectar *bugs*, vulnerabilidades y malos olores en el código (*code smells*) en las primeras etapas del desarrollo permite a los desarrolladores realizar correcciones antes de que estos problemas afecten el proyecto en fases más avanzadas.
- **Mejora continua del código:** gracias a los informes detallados y las recomendaciones proporcionadas por *SonarQube*, se fomenta una cultura de mejora continua dentro del equipo, lo que resulta en un código más limpio y eficiente.
- **Ahorro de tiempo y recursos:** al automatizar los análisis de código, *SonarQube* ahorra tiempo en las revisiones manuales y reduce la posibilidad de errores humanos, lo que a su vez mejora la eficiencia del equipo de desarrollo.

Limitaciones

Sin embargo, el uso de *SonarQube* también presenta algunas limitaciones:

- **Configuración inicial compleja:** configurar *SonarQube* y *Sonar Scanner* puede ser un proceso complicado, especialmente para aquellos que no están familiarizados con la herramienta. Esto puede requerir tiempo adicional para aprender a usarlo correctamente.
- **Limitaciones en la personalización:** aunque *SonarQube* ofrece muchas reglas y configuraciones predeterminadas, personalizar reglas específicas para proyectos particulares puede ser limitado sin la edición de pago de la herramienta.
- **Dependencia del análisis estático:** *SonarQube* se basa principalmente en el análisis estático de código, lo que significa que no detecta problemas de seguridad que solo son visibles durante la ejecución del *software* [389, 390].

Appsweep

Funcionalidades principales

¿Qué es *AppSweep*?

AppSweep es una herramienta gratuita de análisis de seguridad para aplicaciones móviles desarrollada por *Guardsquare*. Está diseñada específicamente para analizar aplicaciones *Android*, evaluando posibles vulnerabilidades y ofreciendo recomendaciones para corregirlas. Su principal objetivo es identificar riesgos de seguridad que podrían comprometer la integridad de la aplicación o la información del usuario, proporcionando a los desarrolladores un informe detallado sobre los problemas encontrados y las soluciones recomendadas.

Características de *AppSweep*

AppSweep ofrece una serie de características que lo convierten en una herramienta eficaz para la detección de vulnerabilidades en aplicaciones móviles:

- **Análisis de seguridad detallado:** *AppSweep* realiza un análisis exhaustivo de las aplicaciones *Android*, detectando vulnerabilidades que podrían poner en riesgo la seguridad del usuario, como fugas de datos, configuraciones inseguras y posibles accesos no autorizados.
- **Detección de permisos innecesarios:** la herramienta identifica si la aplicación está solicitando permisos innecesarios que puedan aumentar el riesgo de exposición a ataques o a fugas de datos.
- **Compatibilidad con archivos *APK*:** *AppSweep* permite cargar directamente el archivo *APK* de la aplicación para analizar su seguridad. Esto facilita la integración en el flujo de trabajo de desarrollo, ya que el análisis puede realizarse antes de la publicación de la aplicación en tiendas como *Google Play*.
- **Recomendaciones específicas:** *AppSweep* no solo detecta vulnerabilidades, sino que también proporciona recomendaciones claras y específicas para solucionarlas, ayudando a los desarrolladores a mitigar los riesgos de seguridad de manera eficiente.
- **Interfaz *web* y local:** *AppSweep* puede ser utilizado en línea mediante su plataforma *web* o de manera local, descargando la herramienta, lo que ofrece flexibilidad en el modo de implementación dependiendo de las necesidades del proyecto.

Estas características convierten a *AppSweep* en una solución eficiente para garantizar la seguridad en aplicaciones *Android* durante todo el ciclo de desarrollo, asegurando que el producto final esté libre de vulnerabilidades críticas.

Configuración y uso en el proyecto

En este proyecto, *AppSweep* se utilizará para analizar la seguridad de la aplicación móvil que estamos desarrollando. Para ello, el primer paso será descargar *AppSweep* o utilizar su versión en línea, según se considere más adecuado para el flujo de trabajo del equipo.

El proceso de configuración y uso se llevará a cabo de la siguiente manera:

1. **Descarga e instalación:** se descargará *AppSweep* en el entorno local o se utilizará la versión *web* si se prefiere no instalar la herramienta localmente. En cualquier caso, la herramienta permitirá analizar archivos ***APK*** de la aplicación *Android*.
2. **Cargar el archivo *APK*:** una vez que se haya desarrollado la versión de la aplicación móvil, se generará el archivo ***APK*** correspondiente, que será cargado en *AppSweep* para su análisis.
3. **Análisis de vulnerabilidades:** *AppSweep* analizará el ***APK*** para identificar posibles vulnerabilidades, fugas de datos o configuraciones inseguras. El análisis incluirá una revisión de los permisos solicitados por la aplicación, verificando si son adecuados o si exponen a la aplicación a riesgos innecesarios.

4. **Revisión del informe:** al finalizar el análisis, *AppSweep* generará un informe detallado con todas las vulnerabilidades encontradas. Este informe será revisado por el equipo de desarrollo para corregir los problemas y asegurar que la aplicación cumpla con los estándares de seguridad requeridos antes de su lanzamiento.
5. **Corrección de problemas:** los desarrolladores realizarán los ajustes y correcciones basados en las recomendaciones proporcionadas por *AppSweep*, asegurando que la aplicación sea lo más segura posible antes de su despliegue.

Este ciclo de análisis y corrección se repetirá de manera iterativa para asegurar que cada nueva versión de la aplicación esté libre de vulnerabilidades y cumpla con los más altos estándares de seguridad.

Beneficios y limitaciones

Beneficios

El uso de *AppSweep* ofrece varias ventajas clave para la seguridad del desarrollo de aplicaciones móviles en este proyecto:

- **Detección exhaustiva de vulnerabilidades:** *AppSweep* realiza un análisis exhaustivo de los archivos ***APK***, detectando una amplia gama de vulnerabilidades que podrían poner en riesgo la seguridad de la aplicación y la privacidad de los usuarios.
- **Facilidad de uso:** la herramienta es fácil de usar, con una interfaz intuitiva tanto en su versión *web* como en la local. Esto permite que los desarrolladores puedan identificar problemas de seguridad rápidamente y sin complicaciones.
- **Recomendaciones detalladas:** *AppSweep* no solo detecta problemas, sino que también proporciona soluciones específicas y claras para que los desarrolladores puedan corregir las vulnerabilidades encontradas de manera eficiente.
- **Versatilidad:** la posibilidad de usar *AppSweep* tanto en línea como de manera local ofrece flexibilidad según las necesidades del proyecto.

Limitaciones

Sin embargo, *AppSweep* también presenta algunas limitaciones que deben tenerse en cuenta:

- **Análisis limitado a aplicaciones *Android*:** *AppSweep* está diseñado específicamente para analizar aplicaciones *Android*, lo que significa que no es útil para proyectos que incluyan aplicaciones *iOS* u otras plataformas.
- **Limitaciones en la versión gratuita:** aunque *AppSweep* es gratuito, algunas funciones avanzadas de análisis podrían estar disponibles solo en versiones pagas o complementarias, lo que puede limitar su alcance para algunos proyectos.
- **Dependencia del análisis estático:** al igual que otras herramientas de análisis estático, *AppSweep* puede no detectar todos los problemas de seguridad que solo se manifiestan durante la ejecución, lo que puede requerir pruebas adicionales con herramientas de análisis dinámico.

4.6.6. Reglamento General de Protección de Datos (GDPR)

El Reglamento General de Protección de Datos (***GDPR***), implementado desde mayo de 2018, se refiere a una serie de reglas creadas por el Parlamento Europeo, el Consejo de la Unión Europea y la Comisión Europea para regular el tratamiento de datos personales de los ciudadanos de la Unión Europea (UE). Este reglamento tiene como objetivo principal proteger la privacidad de los ciudadanos y asegurar que las organizaciones que recopilan, almacenan y procesan estos datos lo hagan de manera transparente, legal y justa. Las disposiciones del ***GDPR*** cubren tanto los datos personales, como las fotos, direcciones de correo electrónico, y direcciones *IP*, hasta información más sensible, como los datos financieros o de salud.

Uno de los principios fundamentales establecidos en el artículo 5 del ***GDPR*** es la licitud, equidad y transparencia en el tratamiento de los datos personales [391]. Esto significa que los datos deben ser recolectados solo con el consentimiento explícito del usuario, para fines legítimos, y las organizaciones deben garantizar la transparencia al explicar cómo usarán los datos personales de las personas afectadas. Además, el ***GDPR*** establece el principio de minimización de datos, lo que implica que solo se deben recopilar los datos estrictamente necesarios para el propósito específico para el cual se solicitan. De igual manera, los datos recolectados deben ser exactos y actualizados, y el controlador debe asegurarse de que cualquier dato incorrecto sea corregido o eliminado.

Otro aspecto importante es la limitación de almacenamiento, que exige que los datos no sean conservados más allá del tiempo necesario para cumplir con los fines originales de su recolección. Además, el ***GDPR*** establece que las empresas deben implementar medidas técnicas y organizativas adecuadas para garantizar la seguridad, confidencialidad e integridad de los datos personales, protegiéndolos contra accesos no autorizados, pérdidas accidentales o destrucción.

El artículo 6 del ***GDPR*** destaca los casos en los que el tratamiento de datos es lícito [392]. Para que el procesamiento sea legal, al menos uno de los siguientes puntos debe cumplirse: el consentimiento del usuario, el cumplimiento de un contrato, la necesidad de proteger intereses vitales, el cumplimiento de una obligación legal, la realización de tareas de interés público o la búsqueda de un interés legítimo. En todos los casos, las organizaciones deben demostrar su responsabilidad y capacidad para cumplir con estos requisitos, conforme lo establece el artículo 5.

GDPR y Guatemala

Aunque el Reglamento General de Protección de Datos (***GDPR***) se aplica principalmente dentro de la Unión Europea, su alcance ha tenido efectos más allá de las fronteras europeas, afectando también a empresas y organizaciones fuera de la UE que procesan datos de ciudadanos europeos. Para las organizaciones que tienen clientes o relaciones comerciales con residentes de la UE, cumplir con el ***GDPR*** es obligatorio, independientemente de la ubicación geográfica de la organización.

En Guatemala, aunque no existe una legislación específica que regule la protección de datos de manera tan detallada como el ***GDPR***, se utiliza la Ley de Libre Acceso a la Información Pública [393], que regula el manejo de los datos personales y los datos sensibles. Esta ley establece ciertos mecanismos de protección y tipifica delitos relacionados con la violación de la privacidad de datos. Sin embargo, su aplicación no es tan estricta ni extensa como el ***GDPR***, lo que deja un vacío legal en la protección de datos en comparación con otros países de la región. Esto representa un reto para las organizaciones que buscan operar en Guatemala o en otros países de América latina con normativas menos estrictas, pero que interactúan con datos de ciudadanos europeos.

En Latinoamérica, varios países han avanzado en la adopción de leyes de protección de datos. Chile fue pionero en la región al promulgar su ley en 1999, seguido por Argentina en 2000, y otros países como Uruguay (2008), México (2010), y Brasil (2018). A pesar de la ausencia de un marco regulatorio uniforme en toda la región, el impacto del ***GDPR*** ha incentivado el desarrollo de

legislaciones locales enfocadas en la protección de datos personales, ya que muchas empresas latinoamericanas realizan transacciones o tienen clientes en Europa, lo que las obliga a adaptarse a los estándares internacionales.

CAPÍTULO 5

Metodología

5.1. Metodología de experiencia de usuario

La metodología en la que consiste el desarrollo de una aplicación móvil como la que ha sido planteada y mencionada con anterioridad se conforma de varios pasos claves para garantizar el complimento de los objetivos planteados, y al mismo tiempo desarrollar una experiencia de usuario cómoda, sencilla y que motive a sus usuarios a hacer uso de las funcionalidades de la misma con el objetivo de mantener un *tracking* de basura consumida por los mismos usuarios y, de esta manera, lograr apoyar en la reducción de sus huellas de carbono y en una clasificación correcta de los desechos que son generados en el día a día.

Estos pasos claves pueden resumirse en los siguientes:

1. Investigar la problemática a solucionar a fondo
2. Definir a los usuarios clave y público objetivo del proyecto
3. Recopilar información del público objetivo del proyecto que nos indique las funcionalidades clave a desarrollar
4. Prototipar de forma iterativa evaluando constantemente el proyecto con una base de usuarios de prueba
5. Desarrollar la solución prototipada que se obtenga como resultado del proceso iterativo
6. Evaluar y validar la solución desarrollada
7. Analizar y discutir sobre los resultados obtenidos

Con la realización de los pasos anteriormente mencionados es posible cumplir con los objetivos planteados inicialmente en la fase de definición del proyecto y redacción del protocolo de trabajo de graduación. Con la investigación, definición de usuarios y prototipado de la solución a la problemática investigada se diseñará una interfaz de usuario intuitiva y atractiva para la aplicación móvil planteada. Con el desarrollo de esta solución prototipada se obtendrá como producto final la interfaz gráfica funcional de la aplicación móvil. Con la evaluación y validación de la aplicación y el análisis y

discusión de resultados se evaluará la efectividad y la usabilidad de la aplicación móvil. Finalmente, con estos objetivos cumplidos se habrá abordado la problemática de la producción desmedida de basura y su incorrecto procesamiento de manera efectiva, proporcionando una solución que motive a sus usuarios a reducir su consumo de basura y huella de carbono.

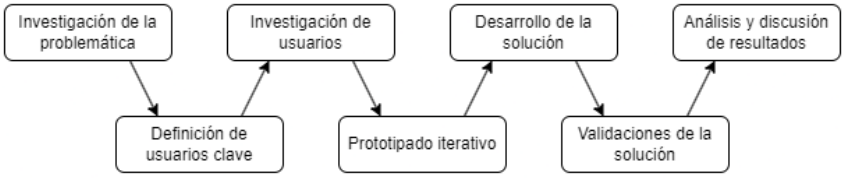


Figura 5.1. Diagrama de resumen de los pasos del proyecto

5.2. Enfoque metodológico

El enfoque metodológico del proyecto se basa en el modelo de *design thinking*, un enfoque que promueve la innovación y la resolución creativa de problemas. Este método se caracteriza por ser iterativo y por incluir cinco etapas fundamentales: empatizar, definir, idear, prototipar y testear. El enfoque radica en su capacidad para adaptarse a las necesidades cambiantes de los usuarios y para fomentar un entendimiento de los desafíos que enfrentan en su interacción con la gestión de residuos.

En el contexto de la aplicación móvil que se ha mencionado anteriormente, el *design thinking* permite abordar la problemática desde una perspectiva holística. A través de la fase de empatía se busca comprender las motivaciones, comportamientos y necesidades de los usuarios en relación con el reciclaje, la gestión de sus desechos y la reducción de la huella de carbono. Este procedimiento permite identificar las funcionalidades clave que la aplicación debe ofrecer para ser realmente útil y atractiva.

La fase de definición permite establecer claramente el problema que estamos abordando y asegurar que la solución a desarrollar se alinee con las expectativas de los usuarios. Durante la fase de ideación, el plan es generar una variedad de ideas y conceptos para la aplicación, fomentando creatividad al momento de prototipar y desarrollar. Posteriormente, en la etapa de prototipado, se crearán versiones iniciales de la aplicación que serán evaluadas con usuarios reales, lo que proporcionará retroalimentación valiosa.

Finalmente, en la fase de *testing*, se realizarán pruebas de usabilidad y satisfacción para garantizar que la solución final cumpla con las expectativas y necesidades de los usuarios. Este enfoque iterativo no solo facilita la incorporación de mejoras continuas, sino que también asegura que la aplicación esté diseñada de manera efectiva para promover prácticas sostenibles en la gestión de residuos.

5.3. Diseño del estudio

El diseño de este estudio se basa en un enfoque descriptivo y exploratorio, centrado en la identificación y comprensión de las necesidades de los usuarios en relación con la gestión de residuos a través de una aplicación móvil. Este enfoque es coherente con el método de *design thinking*, ya que permite un análisis profundo de la problemática antes de desarrollar soluciones concretas.

La investigación se llevará a cabo mediante la recopilación de datos cualitativos y cuantitativos. Se realizan entrevistas y encuestas al público objetivo para identificar las funcionalidades más relevantes y las expectativas de los usuarios. Esta información será crucial para guiar el proceso

de diseño y desarrollo de la aplicación, asegurando que responda efectivamente a las necesidades y comportamientos de los usuarios en su esfuerzo por gestionar y reducir sus residuos.

Este diseño permite una iteración constante, donde los hallazgos de la investigación inicial informarán cada fase del proceso de desarrollo, desde la ideación hasta la evaluación de la solución final, todo con el objetivo del diseño de una solución que cumpla los objetivos planteados.

5.4. Asesoría con la municipalidad de Guatemala

Durante la realización del proyecto se contó con el apoyo de la Municipalidad de Guatemala para múltiples asesorías, consejos, documentación y más información importante que, al final del día, cimentó muchas bases y límites para el alcance del proyecto. La Municipalidad compartió conocimientos prácticos y ejemplos de proyectos similares en el área, lo cual ayudó a definir un marco claro para el alcance y los objetivos de nuestro proyecto, asegurando que cada etapa estuviera alineada con los estándares y necesidades de la comunidad.

Un papel clave en este proceso fue desempeñado por el Ingeniero Alejandro Rodríguez, miembro del equipo de la Unidad de Reciclaje de la Dirección de Medio Ambiente de la Municipalidad, quien actuó como asesor directo para todos los miembros de nuestro equipo de trabajo del megaproyecto tecnológico. Su experiencia en temas de reciclaje y gestión de residuos permitió a nuestro equipo comprender los desafíos específicos de la región, desde las barreras logísticas hasta los aspectos culturales y sociales de la gestión de residuos. La orientación del Ingeniero Rodríguez no solo nos ayudó a diseñar soluciones viables y efectivas, sino que también facilitó una mayor cohesión en el equipo, alinear esfuerzos y reforzar el propósito de nuestra propuesta de forma precisa.

5.4.1. Asesoría para la definición del alcance de la aplicación

El acompañamiento del Ingeniero Rodríguez fue muy importante para delinear el alcance de la aplicación móvil en cuestión a las personas a quienes debía satisfacer la experiencia de usuario a diseñar, permitiéndome enfocar sus funcionalidades en las necesidades y expectativas de los habitantes del departamento de Guatemala. Sus observaciones y conocimientos me ayudaron a adaptar la aplicación para ofrecer una solución accesible y relevante para la comunidad, buscando no solo cumplir con el objetivo del proyecto, sino también contribuir al desarrollo de prácticas sostenibles y al bienestar ambiental del área metropolitana. Por esta razón, múltiples secciones a continuación hacen referencia a la población del departamento de Guatemala y hacen mención a la misma como el público objetivo de la aplicación.

5.5. Definición de usuarios

En la sección de alcance se comentó un poco sobre cómo el proyecto busca poder alcanzar a toda la población del departamento de Guatemala. La ciudad tiene una distribución demográfica relativamente bien definida, en el que según el censo llevado a cabo en 2020, la gran mayoría de la población de la misma es joven, ubicada entre la población de niños recién nacidos hasta los 29 años. Tomando en cuenta que la mayor parte de la población del área metropolitana se encuentra en este rango de edad, podemos comenzar a identificar perfiles generalizados de los usuarios que harán uso de la aplicación.

Los usuarios potenciales de la aplicación móvil que se ha plantado hasta el momento abarcan una variedad relativamente amplia de perfiles basándonos en la población del área metropolitana de Guatemala, lo que incluye desde estudiantes y profesionales jóvenes hasta personas responsables del

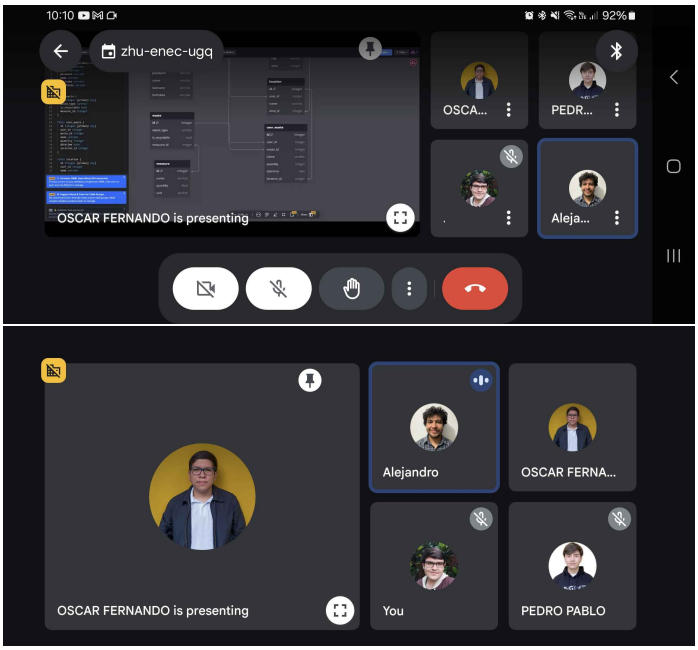


Figura 5.2. Asesorías con la Municipalidad de manera remota

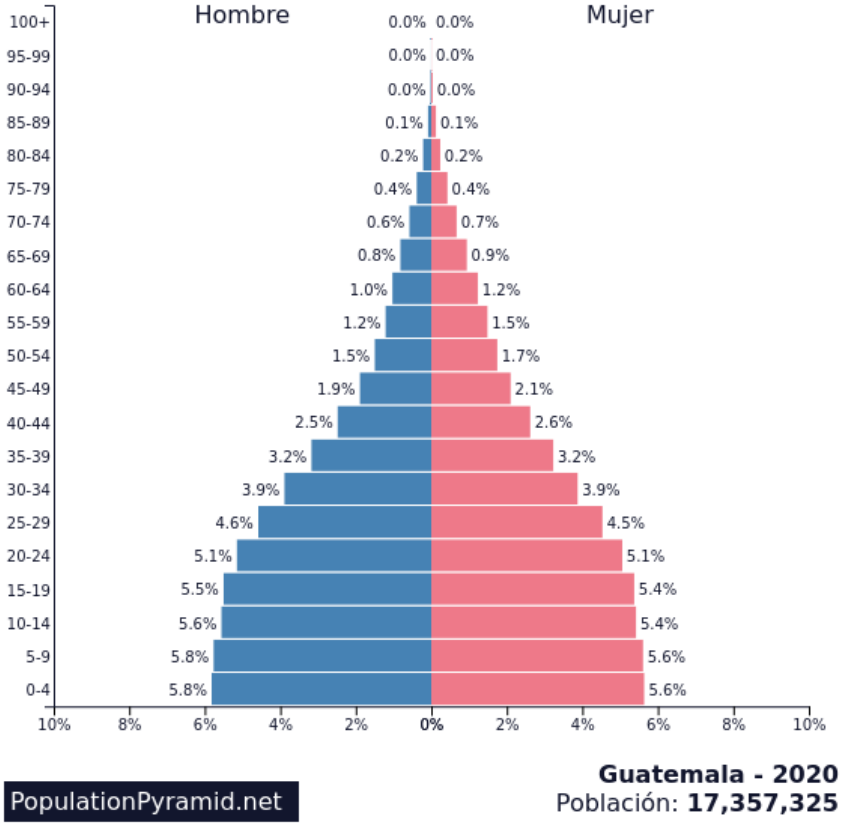


Figura 5.3. Distribución poblacional en Guatemala el año 2020 [8]

hogar y adultos mayores. Este amplio espectro de usuarios presenta diferentes niveles de conocimiento sobre reciclaje y sostenibilidad, así como diversas motivaciones para participar en la gestión de sus desechos. También vale la pena mencionar que este espectro de usuarios también presenta diferentes niveles de comprensión de soluciones tecnológicas como lo son una aplicación móvil como la que se planea en el proyecto. Algunos de los perfiles más importantes que se identifican son los siguientes:

1. **Estudiantes**: este grupo incluye a jóvenes de diferentes niveles educativos. Buscan herramientas que les ayuden a comprender mejor el reciclaje y su impacto ambiental. Son propensos a adoptar nuevas tecnologías y valoran las aplicaciones que incorporan elementos de gamificación y educación.
2. **Profesionales y trabajadores**: adultos que pasan gran parte de su día en entornos laborales. Están interesados en soluciones prácticas que les permitan gestionar sus residuos de manera eficiente en el hogar y en el trabajo. Buscan funcionalidades que faciliten el registro de su consumo de basura y proporcionen recomendaciones sobre reciclaje.
3. **Responsables del hogar**: este grupo se centra en quienes se encargan de la gestión del hogar. Necesitan orientación clara sobre cómo clasificar correctamente los desechos y cómo reducir la producción de basura. Buscan herramientas que les ayuden a involucrar a toda la familia en prácticas sostenibles.
4. **Comerciantes y empresarios**: dueños de pequeñas empresas que buscan maneras de ser más sostenibles en sus operaciones. Están interesados en funcionalidad que les ayude a gestionar residuos generados por su negocio y a implementar prácticas responsables que también atraigan a consumidores conscientes.
5. **Adultos mayores**: personas con experiencia de vida que desean contribuir al medio ambiente. Aunque pueden tener menos familiaridad con la tecnología, su interés en el reciclaje y la sostenibilidad es fuerte. Es crucial diseñar una interfaz intuitiva y fácil de usar para este grupo.

Con los perfiles principales ubicados podemos tabular un cuadro de los mismos perfiles, resaltando características que proporcionan información importante para pensar en características que aíslan a los perfiles de los demás, motivaciones que tienen para poder utilizar la aplicación a desarrollar, y necesidades que la misma aplicación debe poder satisfacer basadas en las motivaciones anteriormente mencionadas.

Grupo de Usuario	**Características**	**Motivaciones**	**Necesidades**
Estudiantes	Jóvenes de diferentes niveles educativos, *tech-savvy*	Aprender sobre reciclaje	Herramientas educativas y gamificación
Profesionales y trabajadores	Adultos en entornos laborales	Gestionar residuos eficientemente	Funcionalidades para el registro de basura
Responsables del hogar	Encargados de la gestión del hogar	Reducir la producción de basura	Orientación sobre clasificación de desechos
Comerciantes y empresarios	Dueños de pequeñas empresas que buscan ser más sostenibles	Implementar prácticas responsables	Herramientas para gestionar residuos generados en sus negocios
Adultos mayores	Personas con experiencia de vida	Contribuir al medio ambiente	Interfaz intuitiva y fácil de usar

Cuadro 5.1. Perfiles de Usuarios

Este cuadro de perfiles de usuarios proporciona una comprensión integral de los distintos grupos que interactuarán con la aplicación móvil, destacando sus características, motivaciones y necesidades

específicas. Esta información es fundamental para el desarrollo de un mapa de empatía, ya que permite profundizar en la experiencia de cada grupo, en este caso, los estudiantes. Al crear el mapa de empatía, se busca explorar no solo qué piensan y sienten los usuarios, sino también cómo se comportan, qué dicen y qué acciones realizan en su vida diaria respecto al reciclaje y la sostenibilidad. Esta conexión entre los perfiles de usuarios y el mapa de empatía garantiza que el diseño de la aplicación se base en una comprensión holística y matizada de los usuarios, facilitando la creación de soluciones efectivas y relevantes que respondan a sus necesidades y aspiraciones.

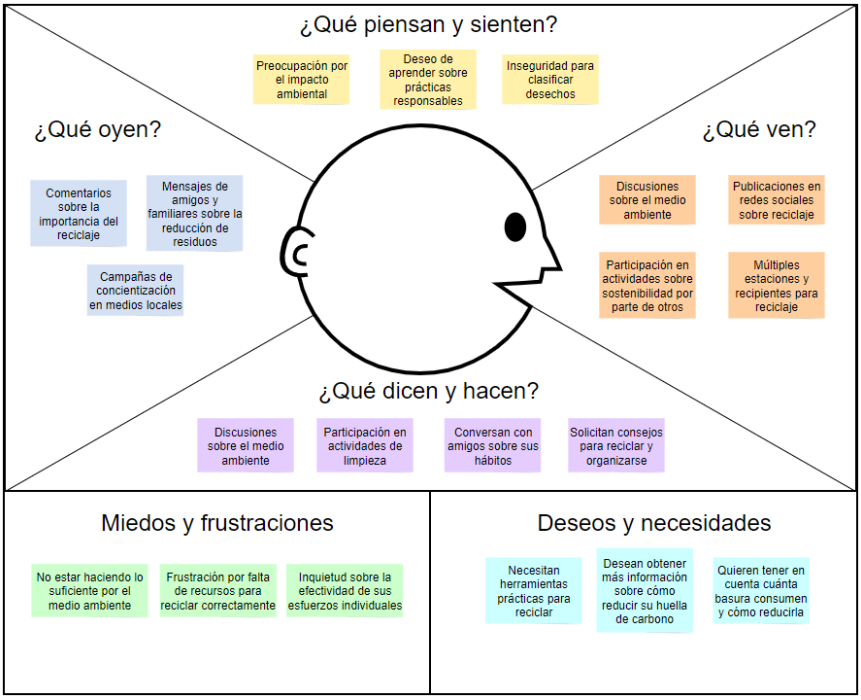


Figura 5.4. Mapa de empatía de un estudiante respecto al tema de reciclaje y gestión de residuos

5.6. Población y muestra

Con los perfiles de los usuarios identificados, lo siguiente consiste en obtener una cantidad de personas que podamos tomar en cuenta como muestra de la población del área metropolitana de Guatemala, ya que el proyecto tiene como objetivo satisfacer las necesidades de gestión de residuos y reducción de la huella de carbono de la misma área metropolitana. Tomando en cuenta que la población del área metropolitana, consistente en los 17 municipios de Guatemala, consistía en 3,599,257 personas [394], podemos realizar los cálculos para la obtención de una cantidad de personas que nos podría dar información significativa para cumplir los objetivos del proyecto.

Para calcular el tamaño de la muestra necesario para realizar las encuestas, se utilizaron las siguientes ecuaciones. Primero, se determinó el tamaño de muestra para una población infinita con un margen de error del 5 % y un nivel de confianza del 95 %. La ecuación es:

$$n = \frac{Z^2 \cdot p \cdot (1-p)}{e^2}$$

Donde:

- n es el tamaño de muestra.
- Z es el valor Z correspondiente al nivel de confianza (1.96 para un 95 % de confianza).
- p es la proporción esperada (asumimos $p = 0.5$ para obtener máxima variabilidad).
- e es el margen de error, en este caso 0.05 (5 %).

Sustituyendo los valores en la ecuación:

$$n = \frac{1.96^2 \cdot 0.5 \cdot (1 - 0.5)}{0.05^2} = 384.16$$

Dado que la población objetivo no es infinita, sino que tiene aproximadamente 3,600,000 habitantes, el tamaño de muestra debe ajustarse para una población finita. La ecuación ajustada es:

$$n_{\text{ajustado}} = \frac{n}{1 + \frac{n-1}{N}}$$

Donde:

- n_{ajustado} es el tamaño de muestra ajustado.
- N es el tamaño de la población, en este caso $N = 3,599,257$.

Sustituyendo los valores:

$$n_{\text{ajustado}} = \frac{384.16}{1 + \frac{384.16-1}{3,599,257}} \approx 384$$

Por lo tanto, el tamaño de muestra ajustado para una población finita es aproximadamente 384 personas.

El margen de error del 5 % y el nivel de confianza del 95 % se seleccionaron debido a su amplio uso en estudios sociales y de mercado que buscan obtener resultados representativos de una población sin incurrir en costos excesivos o tiempos de recopilación de datos demasiado prolongados. Un margen de error del 5 % indica que existe un rango de ±5 puntos porcentuales alrededor de los resultados obtenidos en la muestra, lo cual es aceptable para investigaciones que no requieren un nivel de precisión extremadamente alto pero aún garantizan conclusiones confiables.

Por otro lado, el nivel de confianza del 95 % asegura que si se repitiera el estudio múltiples veces, en el 95 % de los casos, los resultados estarían dentro del margen de error especificado. Este nivel es ampliamente reconocido y utilizado en investigaciones de opinión pública, encuestas de mercado y estudios académicos porque proporciona un buen balance entre precisión y eficiencia en la recolección de datos. Niveles más altos de confianza (como el 99 %) requerirían un aumento significativo en el tamaño de la muestra, lo que no siempre es factible o necesario, mientras que niveles más bajos podrían comprometer la fiabilidad de los resultados.

5.7. Investigación de usuarios

Con el objetivo de conocer mejor a los potenciales usuarios de la aplicación a nivel demográfico y a nivel de conocimientos sobre el tema y necesidades se realizaron múltiples encuestas que nos permitían adquirir información sobre cómo se encuentra el nivel educativo de la muestra de potenciales

usuarios respecto a conocimientos sobre clasificación de basura y reciclaje, y también información sobre las necesidades y expectativas que tienen sobre un proyecto como el que se ha planteado hasta el momento.

Para la obtención de estos resultados se realizaron dos encuestas utilizando *Google Forms*. La primera de ellas busca obtener información demográfica y sobre conocimientos de gestión de residuos y reciclaje de manera general por parte de las personas encuestadas. La segunda de las encuestas busca obtener información sobre las expectativas de los posibles usuarios respecto a las funcionalidades de la aplicación a desarrollar.

5.7.1. Encuesta de conocimiento y comportamiento actual sobre gestión de residuos

La primera de estas dos encuestas desarrolladas en *Google Forms* se tituló "Conocimiento y comportamiento actual sobre la gestión de residuos en Guatemala". La misma tenía cuatro secciones muy importantes, que recopilaban información de las siguientes áreas:

- Información demográfica
- Conocimientos sobre la gestión de residuos
- Comportamiento actual en la gestión de residuos
- Actitudes y percepciones

Con el apoyo de las respuestas de la muestra a evaluar se buscó responder algunas de las siguientes inquietudes respecto a los usuarios potenciales de la aplicación:

- ¿Cómo afecta la demografía de los usuarios potenciales a sus expectativas de la aplicación?
- ¿Qué tantos conocimientos tienen los usuarios potenciales de la aplicación en lo que respecta la gestión de residuos?
- ¿Qué percepción actual tienen los usuarios potenciales respecto al reciclaje?
- ¿Cuáles son las dificultades que tienen los usuarios potenciales para gestionar sus residuos?
- ¿Qué dolores debe solventar la aplicación para que a los usuarios potenciales se les facilite gestionar sus residuos y reducir sus huellas de carbono?
- ¿Realmente están dispuestos los usuarios potenciales a generar un cambio en sus hábitos dadas las herramientas necesarias?

5.7.2. Encuesta de necesidades y expectativas sobre la aplicación

Por otro lado, la segunda de estas dos encuestas desarrolladas se tituló "Necesidades y expectativas sobre una aplicación de gestión de residuos en Guatemala". Esta tenía tres secciones importantes que recopilaban información sobre las siguientes áreas:

- Necesidades de los usuarios
- Expectativas sobre la usabilidad

- Accesibilidad y diseño

Al igual que la encuesta anterior, las respuestas por parte de la muestra a evaluar buscaron responder inquietudes como las siguientes:

- ¿Qué funcionalidades de la aplicación se deben priorizar?
- ¿Qué preferencias tienen los usuarios por ciertas funcionalidades?
- ¿Qué ideas les parecen más útiles a los usuarios?

5.8. Prototipado de la solución

El desarrollo del prototipo inicial se llevó a cabo utilizando *Figma*, una herramienta de diseño colaborativo que permite crear prototipos interactivos de alta fidelidad. Al principio se contaba con varias posibles herramientas para la realización del mismo prototipo. Aparte de *Figma*, las otras opciones que se tomaron en cuenta consistieron en *Adobe XD* y Adobe *Illustrator*. Se eligió *Figma* debido a su accesibilidad en línea, lo que facilita el acceso desde cualquier dispositivo sin requerir instalaciones adicionales, optimizando el flujo de trabajo y la velocidad en las iteraciones de diseño [395]. A diferencia de Adobe *Illustrator*, que está más orientado a la creación de gráficos estáticos, *Figma* permite crear interacciones de usuario de manera más eficiente, simulando una navegación fluida y proporcionando una experiencia más realista del producto final [396].

Ya con una tecnología elegida con el objetivo de desarrollar el prototipo, el proceso se estructuró en tres iteraciones, cada una de las cuales se centró en recoger retroalimentación y realizar mejoras significativas en el diseño de la aplicación. A continuación se describen las formas en las que se obtuvo retroalimentación por parte de potenciales usuarios para cambios a realizar durante el proceso iterativo, y los principales cambios realizados por parte de las tres iteraciones al prototipo, con algunos ejemplos puntuales de cambios.

5.8.1. Encuestas sobre el prototipo de la solución

Al finalizar cada una de las iteraciones al prototipo, para complementar el proceso de realizar pruebas con usuarios puntuales, también se realizó una encuesta en la que se presentaban varias de las pantallas diseñadas más importantes de la aplicación con el objetivo de recibir una mayor cantidad de opiniones que permitieran tomar en cuenta aspectos de la aplicación a rediseñar de una forma más sencilla y fácil de esparcir hacia la mayor cantidad de población y usuarios potenciales posibles. Estas encuestas consistían en una estructura bastante sencilla:

1. Presentar una captura con la pantalla a evaluar
2. Solicitar una calificación de 1 a 10 de la pantalla a nivel estético
3. Solicitar comentarios opcionales sobre el por qué de la calificación

El objetivo de estas encuestas no sólo es obtener una valoración numérica que indique las pantallas que tienen un mayor margen de mejora, sino también obtener comentarios sobre estas mismas posibles mejoras para implementarlas y evaluar los cambios en las siguientes iteraciones. La sección en la que se solicitan comentarios opcionales sobre el por qué de la calificación es, valga la redundancia, opcional, para que las personas encuestadas no inviertan demasiado tiempo en responder y,

Tecnología	Pros	Contras
Figma	Accesible desde cualquier dispositivo con conexión a Internet, permitiendo trabajar de manera ágil sin necesidad de instalaciones adicionales. También ofrece una experiencia de prototipado interactivo de alta fidelidad, permitiendo una navegación fluida para simular la experiencia de usuario final.	Requiere una conexión constante a Internet, lo cual puede ser una limitación en entornos de trabajo sin acceso confiable. Además, algunas de sus funciones avanzadas pueden ser limitadas en comparación con *software* especializado en diseño gráfico.
Adobe XD	Incluye capacidades de prototipado interactivo y se integra bien con otras aplicaciones de *Adobe*, lo cual facilita la importación y exportación de archivos en el ecosistema *Adobe*.	Necesita ser instalado en el dispositivo y tiende a requerir mayor capacidad de *hardware*. Las funciones de colaboración en tiempo real son menos avanzadas en comparación con *Figma*, lo cual puede limitar el flujo de trabajo colaborativo.
Adobe Illustrator	Es ideal para el diseño de gráficos y elementos visuales detallados, proporcionando alta precisión en el diseño de componentes individuales. Su capacidad para crear elementos visuales complejos lo hace una opción preferida para gráficos de alta calidad.	No está optimizado para el prototipado interactivo, lo que limita su uso para simular experiencias de usuario. Además, tiene una curva de aprendizaje mayor para la creación de interfaces, siendo menos intuitivo para el diseño de experiencia de usuario en comparación con *Figma* y *Adobe XD*.

Cuadro 5.2. Comparativa de herramientas de diseño para prototipos

de esta manera, recibir una mayor cantidad de respuestas. Aunque los comentarios no sean tantos, debido a la simplicidad de los diseños de las pantallas, es muy probable que los potenciales usuarios resalten las mismas posibles mejoras.

5.8.2. Resumen de los cambios realizados en cada iteración

- **Primera iteración**: durante la primera iteración se presentó un prototipo inicial que permitió identificar la falta de ciertas pantallas necesarias para llevar a cabo funcionalidades clave. Esta etapa fue fundamental, ya que la retroalimentación obtenida de los usuarios de prueba destacó áreas que requerían más atención y desarrollo. Un ejemplo puntual consistió en la agregación de una pantalla específica para configurar direcciones de los usuarios.
- **Segunda iteración**: la segunda iteración del prototipo se centró en mejorar aspectos específicos de la interfaz de usuario. Basado en los comentarios recibidos, se realizaron ajustes en el diseño visual, buscando una estética más atractiva en algunas pantallas y una experiencia de usuario más fluida. Un ejemplo puntual consistió en la mejora significativa de la pantalla de configuración de usuarios que inicialmente no agradó a las personas que probaron la aplicación.
- **Tercera iteración**: durante la tercera se desarrolló el prototipo final. Este prototipo incorporó todas las mejoras sugeridas y se alineó con las expectativas de los usuarios. Se realizaron pruebas de usabilidad para evaluar la efectividad del diseño y asegurarnos de que cumpliera

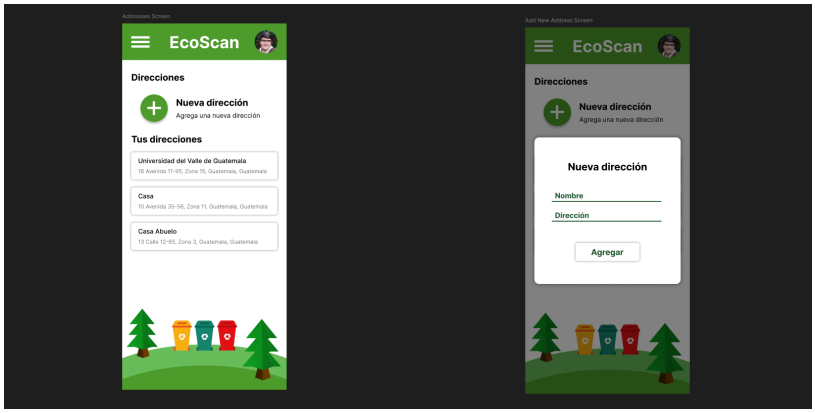


Figura 5.5. Agregación de una pantalla de configuración de direcciones durante la primera iteración del prototipo

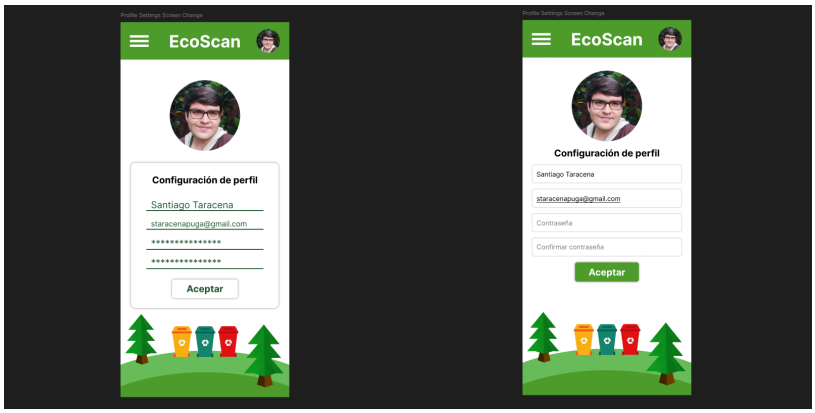


Figura 5.6. Mejoras en la pantalla de configuración de información de los usuarios

con los objetivos planteados al inicio del proyecto. Este enfoque iterativo no solo garantizó un producto de calidad, sino que también fomentó un diseño centrado en el usuario, maximizando la relevancia y funcionalidad de la aplicación. Un ejemplo puntual de un cambio realizado esta última iteración consistió en cambiar parte de la paleta de colores para adecuar mejor los colores de botones de acciones como cerrar sesión o eliminar una cuenta.

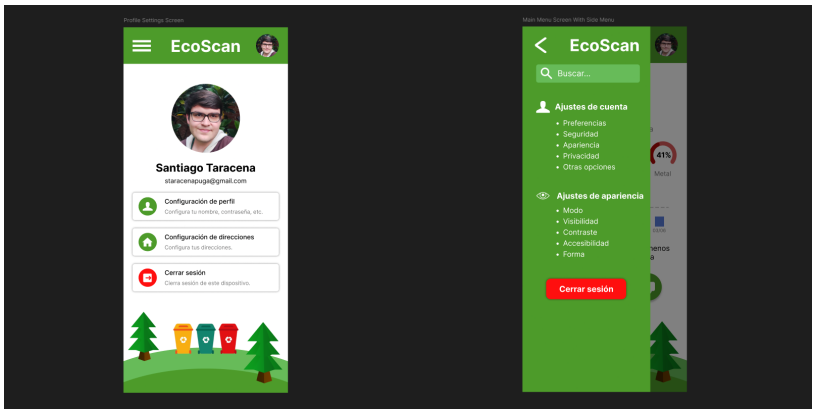


Figura 5.7. Cambios en la paleta de colores para indicar acciones importantes

A través de este proceso iterativo, se logró no solo un prototipo visualmente atractivo, sino también una aplicación funcional que responde a las necesidades y expectativas de los usuarios. La implementación de la retroalimentación en cada fase permitió identificar y abordar áreas críticas, garantizando que el diseño final no solo cumpliera con los requisitos técnicos, sino que también proporcionara una experiencia de usuario fluida y satisfactoria. Este enfoque centrado en el usuario ha sido esencial para el desarrollo de una solución efectiva, reafirmando la importancia de la iteración y la colaboración en el proceso de diseño. Con el prototipo final terminado, el cual fue producto de las iteraciones realizadas, se procedió a la etapa de desarrollo de la interfaz gráfica de la solución planteada.

5.9. Desarrollo de la interfaz gráfica de la aplicación

El desarrollo de la interfaz gráfica de la aplicación se llevó a cabo utilizando tecnologías modernas y robustas, específicamente *JavaScript, React Native* y *Node.js*. La elección de estas tecnologías fue estratégica y se fundamentó en su capacidad para proporcionar una experiencia de usuario fluida y un desarrollo eficiente para poder ser cumplido en los aproximadamente cinco meses de desarrollo del trabajo de graduación en cuestión. *JavaScript* es un lenguaje de programación versátil que permite tanto el desarrollo del *frontend* como del *backend*, lo que facilita la comunicación y la integración entre ambas partes de la aplicación. Por su parte, *Node.js* se utilizó para construir el *backend* de la aplicación, lo que permite manejar solicitudes de manera eficiente y escalar la aplicación según sea necesario. La elección de estas dos tecnologías principales anteriormente mencionadas va especialmente de la mano con la elección de *React Native*, la cual es una librería que permite desarrollar interfaces gráficas para aplicaciones móviles en plataformas como *iOS* y *Android*, lo cual se consideró necesario dado que según las estadísticas proporcionadas por *StatCounter* [397], el 77 % de los usuarios de dispositivos móviles en el país utilizan *Android*, mientras que el 23 % restante prefieren *iOS*. Esta distribución significativa de usuarios entre ambas plataformas indica la importancia de desarrollar una aplicación móvil que pueda satisfacer las necesidades de ambos grupos de usuarios de manera eficiente y efectiva.

El desarrollo de la aplicación comenzó con la creación de las pantallas iniciales de inicio de sesión y registro. Estas pantallas son fundamentales, ya que establecen la primera interacción del usuario con la aplicación. Se diseñaron con un enfoque en la usabilidad, garantizando que fueran intuitivas y accesibles. La pantalla de registro incluye campos obligatorios que permiten a los usuarios ingresar información relevante, como su correo electrónico y contraseña. A medida que el desarrollo avanzaba, se crearon tanto la pantalla principal de la aplicación como las pantallas de configuración de datos del usuario. Estas pantallas permiten a los usuarios personalizar su experiencia en la aplicación, ingresando información relevante sobre sus hábitos de consumo y preferencias de reciclaje. La interfaz se diseñó para ser clara y fácil de navegar, facilitando el ingreso y la edición de datos. Se incluyeron opciones para agregar direcciones para tomar en cuenta el consumo de basura dada cierta dirección y localización geográfica.

Además, se desarrollaron pantallas para el análisis de la información recabada por la aplicación. Estas pantallas son cruciales, ya que ofrecen a los usuarios una visión clara de sus hábitos de reciclaje y consumo, permitiéndoles identificar áreas de mejora. Se implementaron gráficos interactivos y resúmenes visuales que muestran el impacto de las acciones del usuario a lo largo del tiempo. Se garantizó que la información se presentara de manera atractiva y fácil de interpretar, lo que motiva a los usuarios a continuar utilizando la aplicación y adoptar prácticas más sostenibles.

Finalmente, durante el proceso de desarrollo, se realizaron pruebas continuas para garantizar la funcionalidad y la estabilidad de la aplicación. Esto incluyó pruebas de usabilidad con un grupo diverso de usuarios, lo que permitió identificar áreas de mejora y optimizar la experiencia general. La combinación de estas tecnologías y un enfoque centrado en el usuario ha sido clave para crear una aplicación que no solo es funcional, sino que también se alinea con las necesidades y expectativas de

Tecnología	Pros	Contras
React Native	Código reutilizable para *Android* e *iOS*. Gran comunidad y soporte. Acceso a componentes nativos.	Dependencia de bibliotecas externas. Rendimiento inferior en comparación con aplicaciones nativas. Curva de aprendizaje para nuevos desarrolladores.
Flutter	Alta velocidad de desarrollo. Rendimiento cercano al nativo. Personalización de la interfaz de usuario.	Tamaño de la aplicación más grande. Comunidad más pequeña en comparación con *React Native*. Algunos *widgets* pueden requerir ajustes para la plataforma.
Xamarin	Integración con *Visual Studio*. Código compartido con *.NET*. Acceso a *API* nativas.	Curva de aprendizaje para desarrolladores no familiarizados con *C#*. Tamaño de la aplicación más grande. Dependencia de la plataforma para el rendimiento.
Ionic	Basado en tecnologías *web (HTML, CSS, JavaScript)*. Fácil de aprender para desarrolladores *web*. Gran cantidad de *plugins* disponibles.	Rendimiento inferior en comparación con aplicaciones nativas. Limitaciones en la funcionalidad nativa. Dependencia de *WebView* para renderizar la interfaz.
Android Studio	Herramienta oficial para desarrollo en *Android*. Acceso completo a *API* y bibliotecas nativas. Amplias opciones de depuración y pruebas.	Solo para *Android*, no multiplataforma. Requiere conocimientos de *Java* o *Kotlin*. Puede ser pesado y consumir muchos recursos.
Swift	Alto rendimiento para aplicaciones *iOS*. Código limpio y fácil de leer. Integración completa con el ecosistema de *Apple*.	Solo para desarrollo en *iOS*. Curva de aprendizaje para nuevos desarrolladores. Necesidad de mantener el código separado para *iOS* y *Android*.

Cuadro 5.3. Comparativa de tecnologías para desarrollo móvil

los usuarios en su camino hacia un consumo más sostenible.

5.9.1. Priorización de funcionalidades

Para asegurar un desarrollo eficiente y enfocado en los objetivos centrales del proyecto, se priorizaron algunas de las funcionalidades sobre otras. Las características de la aplicación fueron clasificadas en función de su impacto en la experiencia de usuario y su relevancia para la resolución de la problemática planteada. Las funcionalidades de mayor prioridad son aquellas que promuevan directamente la reducción de residuos y el reciclaje, mientras que otras características más complementarias, como la gamificación, fueron implementadas en etapas posteriores del desarrollo.

5.10. Evaluación final a la aplicación funcional

Se llevaron a cabo pruebas de usabilidad para evaluar la intuitividad de la aplicación móvil, centradas en diversas acciones clave que los usuarios realizarían en su día a día usando la solución

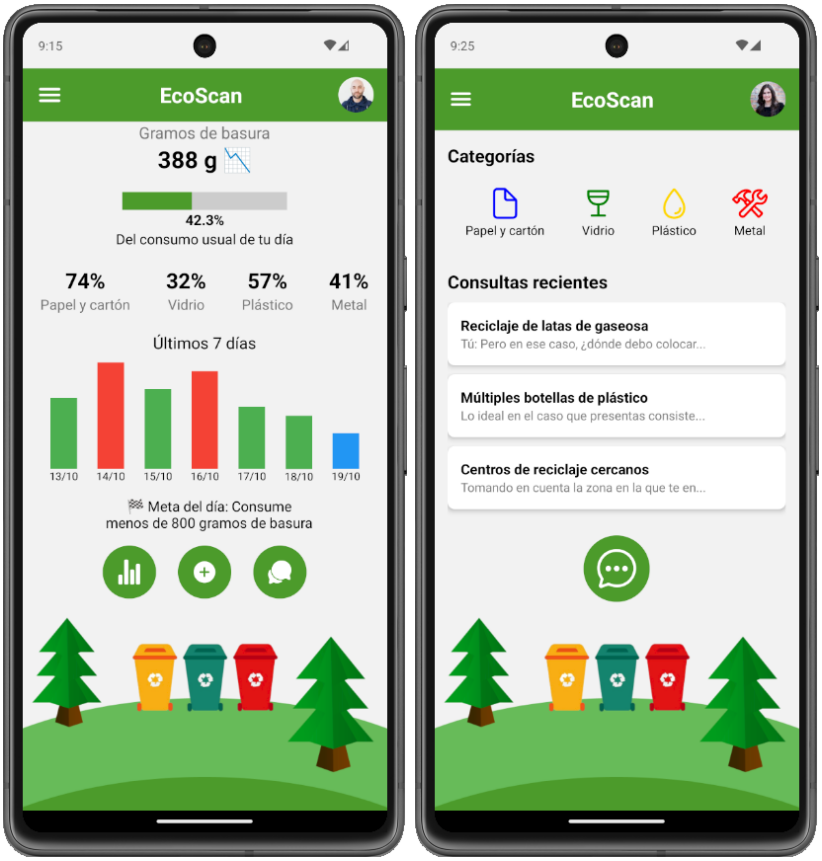


Figura 5.8. Pantallas de la aplicación en desarrollo

desarrollada. Durante estas pruebas, se midió el tiempo que los participantes tardaban en completar tareas específicas, entre las cuales se encuentran las siguientes:

- Crear una cuenta
- Iniciar sesión
- Registrar un consumo de basura
- Formular una pregunta al *chatbot*
- Cambiar foto de perfil
- Agregar una dirección
- Revisar analíticas diarias

Este enfoque permitió identificar no solo la eficiencia de la aplicación, sino también cualquier posible obstáculo que los usuarios pudieran encontrar durante su interacción.

Para obtener una comprensión más profunda del comportamiento del usuario, se formaron dos grupos. El primer grupo realizó las pruebas de usabilidad convencionales, donde se tomó nota del tiempo de respuesta y se recopilaron observaciones cualitativas sobre su experiencia. Por otro lado, el segundo grupo utilizó tecnología de *eye tracking* durante las mismas pruebas mediante el uso de un aparato para esta técnica llamado Tobii *Eye Tracker* 5. Esta técnica proporciona resultados más

visuales y detallados sobre cómo los usuarios navegan por la interfaz de la aplicación. Al rastrear el movimiento de los ojos, es posible identificar los elementos que captan más atención, así como aquellos que pueden causar distracción o confusión.

Los resultados de ambas pruebas se compararán para obtener una visión integral sobre la usabilidad de la aplicación. El análisis de los resultados generados tanto por estas últimas evaluaciones como las anteriores proporcionarán material para verificar el cumplimiento de los objetivos planteados inicialmente durante la concepción del proyecto.

5.11. Metodología de *backend*, gestión de base de datos y análisis de datos

El desarrollo de la aplicación se realizó en diversas etapas clave, cada una orientada a cumplir con los objetivos específicos del proyecto, como la creación del *API*, la gestión de la base de datos y el análisis de datos para una aplicación de gestión de residuos y reducción de huella de carbono. Durante este proceso, se tomaron decisiones estratégicas en cuanto a la selección de tecnologías, el diseño estructural de la base de datos, la implementación de las funcionalidades en el *API*, y el desarrollo de análisis de los datos de los usuarios.

Asimismo, a lo largo de la ejecución del proyecto, se tomaron decisiones basadas en buenas prácticas de desarrollo de *software* para optimizar la arquitectura del *API* principal del proyecto. Con base en un análisis riguroso, se implementaron procesos de *CI/CD* fundamentados en la metodología *DevOps*, con el objetivo de garantizar una entrega continua y de calidad.

Cada iteración representó un paso hacia un sistema más eficiente y adaptable, alineado con las necesidades de los usuarios, los objetivos de sostenibilidad del proyecto y la visión de establecer una base sólida para su futura expansión.

5.12. Elección de tecnologías

La selección de tecnologías fue una de las decisiones más importantes en el desarrollo del proyecto, ya que influiría en la escalabilidad, flexibilidad y sostenibilidad de la aplicación a largo plazo. Se evaluaron diferentes opciones basadas en criterios de rendimiento, facilidad de uso y soporte continuo.

El motor de base de datos elegido fue *PostgreSQL*, una base de datos de código abierto reconocida por su robustez y cumplimiento de las propiedades ACID (Atomicidad, Consistencia, Aislamiento y Durabilidad), lo cual la convierte en una excelente opción para gestionar los datos de una aplicación que requiere transacciones seguras y consistentes. *PostgreSQL* es altamente escalable, permitiendo manejar grandes volúmenes de datos y ejecutar consultas complejas de manera eficiente. Otra ventaja clave es su capacidad para realizar análisis avanzados directamente en la base de datos, eliminando la necesidad de exportar datos a herramientas externas y simplificando la arquitectura del sistema.

Para el desarrollo del *API* se eligió *Express.js*, un *framework* ligero de *Node.js*. *Express.js* permite la construcción rápida y flexible de *API* RESTful, fundamental para una comunicación fluida entre el *frontend* y la base de datos. Su naturaleza asíncrona facilita el manejo de múltiples solicitudes concurrentes, haciéndolo ideal para una aplicación que necesita un rendimiento ágil y alta disponibilidad. Esta combinación de tecnologías garantiza un sistema eficiente, escalable y fácil de mantener, capaz de adaptarse a las crecientes demandas de la aplicación.

Una vez seleccionadas las tecnologías principales para el *API* y la estructura de la base de datos, se optó por herramientas complementarias que aseguren la calidad del sistema. Se eligió la

herramienta de pruebas de carga *Artillery*, que permite una conexión eficaz con aplicaciones basadas en *Express.js*, facilitando la simulación de escenarios de tráfico y evaluando el comportamiento del *API* bajo diferentes cargas.

Además, *Artillery* se integra fácilmente con los procesos de *CI/CD* de *GitHub Actions*, lo cual permite automatizar estas pruebas y monitorizar el rendimiento del *API* a lo largo del tiempo. Esto asegura un control continuo del rendimiento y una preparación adecuada para las necesidades futuras de la aplicación.

Para proporcionar un monitoreo continuo de la *API*, se seleccionó una herramienta de monitoreo de errores: *Sentry*. Esta decisión se basó en la facilidad de implementación que *Sentry* ofrece para aplicaciones basadas en *Express.js*, además de ser una herramienta robusta que proporciona la información necesaria para gestionar y resolver errores de manera eficiente.

5.13. Primera iteración

5.13.1. Base de datos: diseño versión inicial

En la **primera iteración**, se diseñó la estructura inicial de la base de datos, la cual se centró en tres tablas principales: ***user***, ***waste*** y **user_waste**. Estas tablas establecieron las bases del sistema, permitiendo gestionar la relación entre los usuarios y los residuos que generan.

- ***user***: esta tabla almacena la información básica de los usuarios, incluyendo campos como `id`, nombre de usuario (`username`), correo electrónico, contraseña, nombre, apellido y fecha de nacimiento. Esta estructura es clave para identificar de manera única a cada usuario en el sistema.
- ***waste***: define los diferentes tipos de residuos, con un `id` único, el tipo de residuo (`waste_type`), si es reciclable o no (`is_recyclable`), y el tipo de medida asociada (`measure_type`), lo que proporciona flexibilidad para manejar residuos de distintas categorías.
- ***user_waste***: relaciona a los usuarios con los residuos que generan. Contiene los campos `id` del usuario, `id` del residuo, nombre del residuo, la cantidad generada, la fecha en que se generó el residuo, y el lugar donde ocurrió la generación de residuos. Esta tabla almacena todos los registros que los usuarios ingresan.

A continuación se presenta el diagrama entidad-relación de la primera iteración de la base de datos:

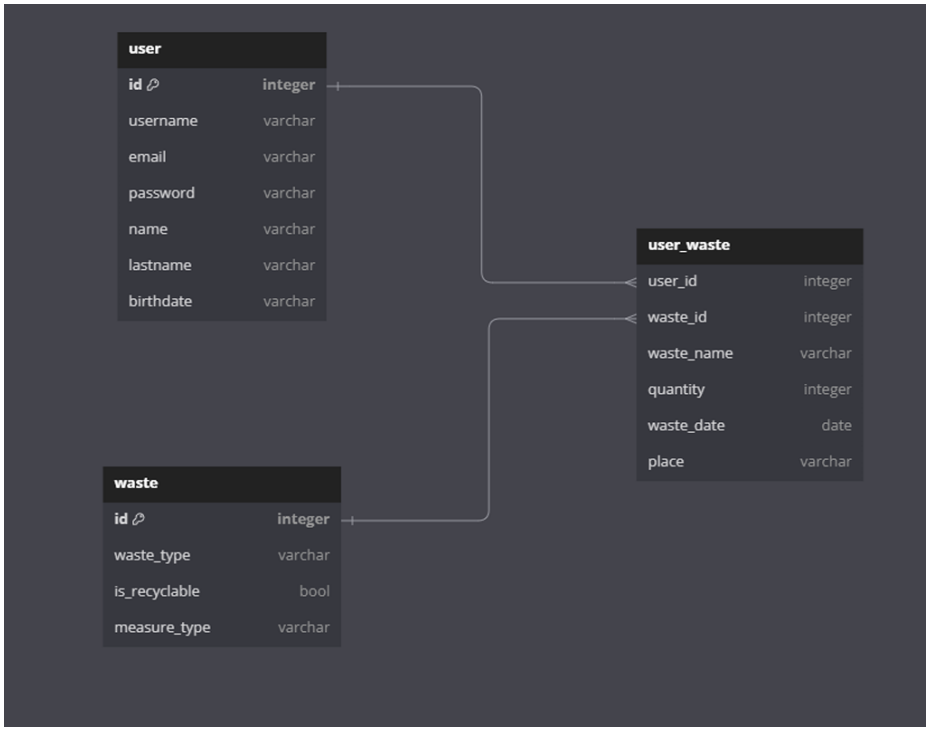


Figura 5.9. Diagrama entidad-relación de la primera iteración

5.13.2. *API*: definición de las funciones

Durante la **primera iteración**, se desarrollaron las funciones esenciales que permitieron a los usuarios registrarse en el sistema, acceder a sus cuentas y registrar los residuos generados. Estas funciones establecieron una base sencilla pero funcional para la interacción entre los usuarios y la base de datos, proporcionando un punto de partida confiable para futuras ampliaciones y mejoras del sistema.

En esta fase, se diseñó y configuró la estructura inicial de la base de datos. Las funciones de acceso a la base de datos se implementaron de forma manual, sin el uso de un ORM, permitiendo realizar las operaciones *CRUD* (Crear, Leer, Actualizar y Eliminar) correspondientes para cada entidad en el sistema. Este enfoque inicial facilitó una implementación básica de las operaciones de gestión de datos y sentó las bases para manejar la estructura principal de las tablas y sus datos.

A continuación se presenta un *query* de ejemplo de creación de Usuarios:

Listing 5.1. Archivo `createUser.sql`

```
INSERT INTO user (username, email, password, name, lastname, birthdate)
VALUES ('nombre_de_usuario', 'correo_electronico', 'contrasena', 'nombre',
'apellido', 'fecha_de_nacimiento');
```

5.14. Segunda iteración

5.14.1. Base de datos: diseño de tablas para ubicación y medición

En la **segunda iteración**, se realizó una revisión del diseño inicial para mejorar la precisión y el detalle de los datos almacenados. Se añadieron las tablas ***location*** y ***measure***, lo que permitió incluir más información sobre la ubicación de los residuos generados y las medidas utilizadas para cuantificarlos.

- ***location***: almacena las ubicaciones relacionadas con los registros de residuos, permitiendo identificar el lugar donde se generan. Incluye los campos `id`, nombre del lugar, y su geoposición (coordenadas GPS).
- ***measure***: define las unidades de medida utilizadas para cuantificar los desechos, como kilogramos, litros, entre otros. Contiene los campos `id`, nombre de la medida, cantidad y unidad, lo que facilita la estandarización de las cantidades registradas.

El siguiente diagrama muestra la estructura mejorada de la base de datos en la segunda iteración:

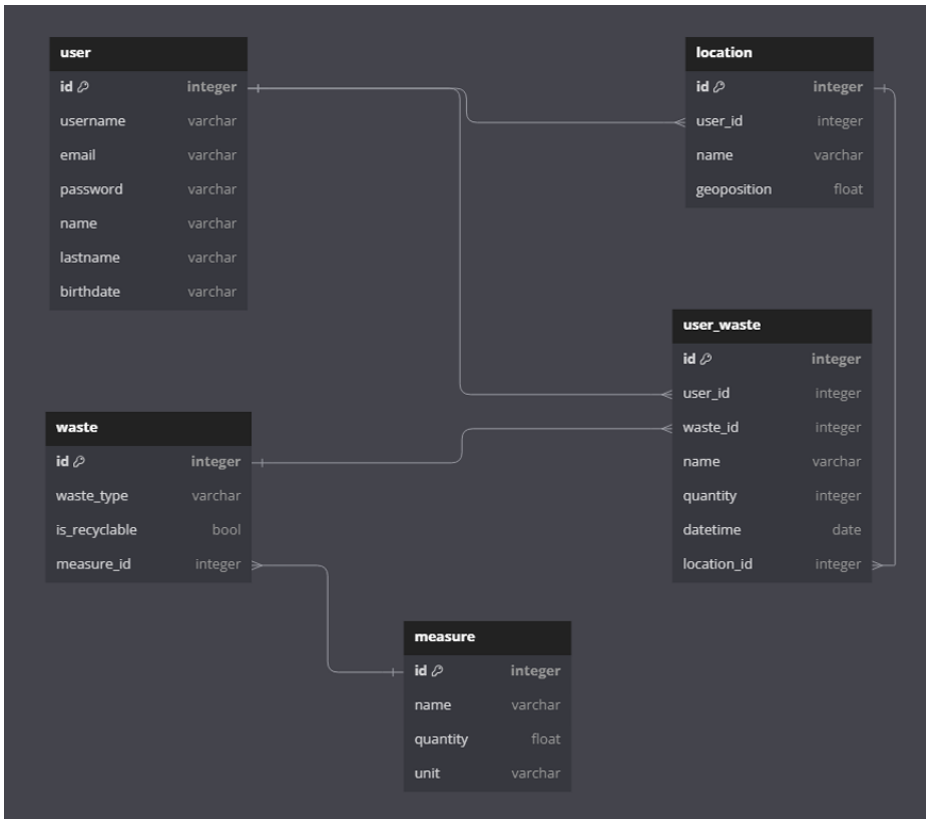


Figura 5.10. Diagrama entidad-relación de la segunda iteración

5.14.2. *API*: creación de funciones *CRUD* para ubicaciones y mediciones

En la **segunda iteración**, se ampliaron las funcionalidades del *API* para gestionar las nuevas tablas ***location*** y ***measure***, permitiendo registrar de manera más detallada los residuos, junto con

la ubicación de donde provienen y las medidas asociadas.

Durante esta iteración, se implementaron los modelos y las relaciones entre todos los modelos existentes en el sistema principal. Para las tablas recién añadidas, se desarrollaron las funciones *CRUD* (Crear, Leer, Actualizar y Eliminar) de manera manual, facilitando un control directo sobre el acceso a la base de datos. Asimismo, se implementaron las primeras validaciones de datos para asegurar que los valores ingresados en estas tablas cumplieran con los requisitos establecidos, contribuyendo a la integridad y coherencia de la información en el sistema.

A continuación se presenta un ejemplo de función de validación:

Listing 5.2. Archivo `validateUser.js`

```
// middleware/validateUser.js
const { body, validationResult } = require('express-validator');

// Validacion para el registro de usuarios
exports.validateRegister = [
    body('email').isEmail().withMessage('Invalid email format'),
    body('password').isLength({ min: 6 }).withMessage('Password must be at
        least 6 characters long'),
    (req, res, next) => {
        const errors = validationResult(req);
        if (!errors.isEmpty()) {
            return res.status(400).json({ success: false, errors: errors.
                array() });
        }
        next();
    },
];
```

5.15. Tercera iteración

5.15.1. Base de datos: diseño de tabla para la zona de la ciudad

En la **tercera iteración**, se introdujo una mejora significativa en la gestión de las ubicaciones de los registros de residuos. En lugar de utilizar coordenadas geográficas, se optó por un sistema basado en zonas dentro de la Ciudad de Guatemala, facilitando así la clasificación y administración de los desechos por zonas.

Para soportar esta funcionalidad, se añadió la tabla ***area***, la cual está vinculada a ***location*** para que cada ubicación esté asociada a un área específica de la ciudad.

- ***area***: almacena información sobre las áreas geográficas dentro de la ciudad, incluyendo campos como `id`, el nombre de la ciudad, y el área específica, lo que facilita la identificación de zonas dentro del sistema.

A continuación, se presenta el diagrama actualizado de la estructura de la base de datos en la tercera iteración:

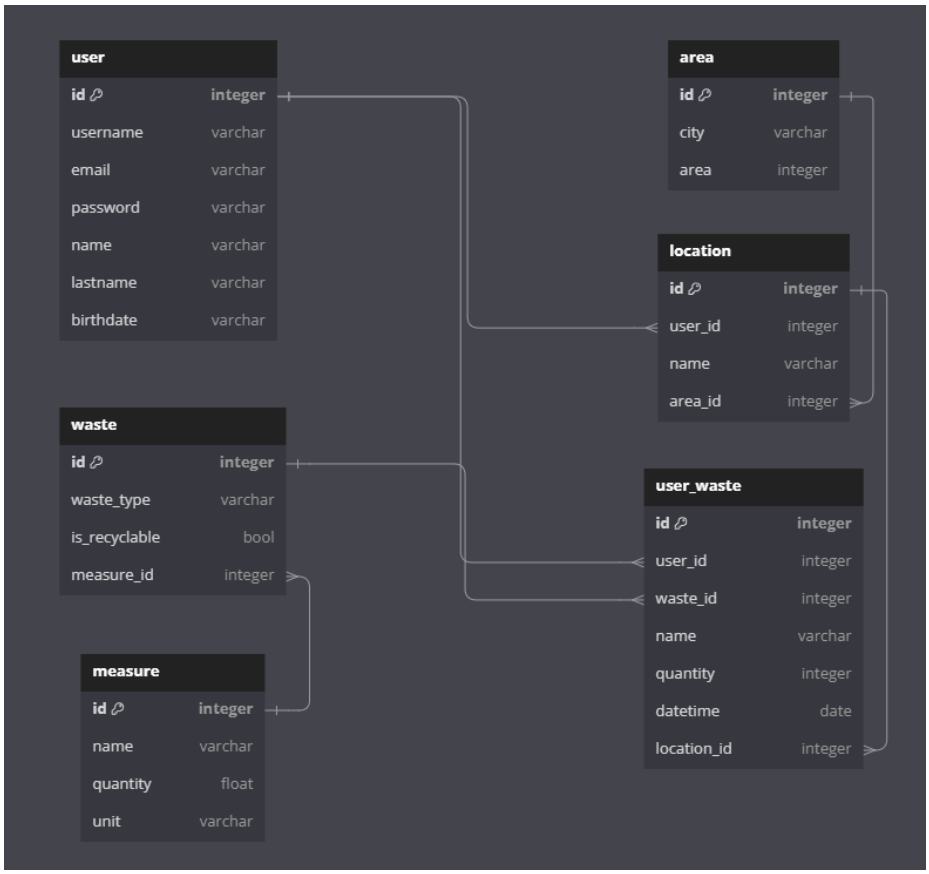


Figura 5.11. Diagrama entidad-relación de la tercera iteración

5.15.2. *API*: creación de modelos y uso de ORM

Durante la **tercera iteración**, se realizó un cambio fundamental en la arquitectura del *API*, utilizando ***Express.js*** como el *framework* para el servidor y ***Sequelize*** como el ORM para la interacción con la base de datos *PostgreSQL*. Esto permitió mejorar la escalabilidad y eficiencia del sistema.

El código se organizó en carpetas específicas para garantizar una estructura modular:

- En la carpeta `config`, se creó el archivo `database.js`, que se encarga de la conexión a *PostgreSQL* y de la sincronización de los modelos con la base de datos.
- Los modelos de la base de datos se organizaron en la carpeta `models`, definiendo las propiedades de las tablas y las relaciones entre ellas, como en los archivos `User.js`, `Waste.js`, `Area.js`, `Location.js`, `Measure.js` y `UserWaste.js`.
- Los controladores para gestionar las operaciones CRUD sobre cada entidad se implementaron en la carpeta `controllers`
- Las rutas *RESTful* se definieron en la carpeta `routes`, permitiendo a los usuarios interactuar con la *API*.

A continuación se presenta un ejemplo de un modelo con *Sequelize*:

Listing 5.3. Archivo `Area.js`

```
// models/Area.js
const { DataTypes } = require('sequelize');
const sequelize = require('../config/database');

const Area = sequelize.define('Area', {
    id: {
        type: DataTypes.INTEGER,
        autoIncrement: true,
        primaryKey: true,
    },
    city: {
        type: DataTypes.STRING,
        allowNull: false,
    },
    area: {
        type: DataTypes.INTEGER,
        allowNull: false,
    },
}, {
    tableName: 'area',
    freezeTableName: true
});

module.exports = Area;
```

5.16. Cuarta iteración

5.16.1. Base de datos: cambios en mediciones de desechos

En la **cuarta iteración**, se realizaron cambios significativos en la estructura de la base de datos, reflejando un enfoque renovado para el ingreso y manejo de los datos de residuos por parte de los usuarios. Estos cambios se implementaron con el objetivo de proporcionar análisis y métricas más precisas y detalladas, permitiendo a los usuarios acceder a información más completa sobre sus hábitos de gestión de residuos.

Las modificaciones introducidas en esta iteración respondieron a la necesidad de simplificar el modelo de datos y ofrecer un análisis preciso de los registros de desechos proporcionados por el usuario.

La integración de los campos `measure_type` y `measure_unit` directamente en el cuadro `waste` en la cuarta iteración simplifica el proceso de ingreso de datos y facilita el acceso a la información sin la necesidad de relaciones adicionales. Este cambio permite un acceso más rápido a las unidades de medida y tipos de residuos, lo que es crucial para el análisis en tiempo real y la presentación de métricas detalladas a los usuarios.

Además, la introducción del campo `average_weight` proporciona una solución más flexible para gestionar los residuos que se cuantifican en unidades, ofreciendo una mayor precisión en el cálculo de los residuos totales. Finalmente, la adición del campo `has_waste_collection` en el cuadro `location` permite recopilar datos valiosos sobre la disponibilidad de servicios de recolección de basura, lo que enriquece los análisis sobre la accesibilidad y la gestión de residuos en diferentes ubicaciones.

A continuación se presenta el diagrama actualizado de la estructura de la base de datos en la cuarta iteración:

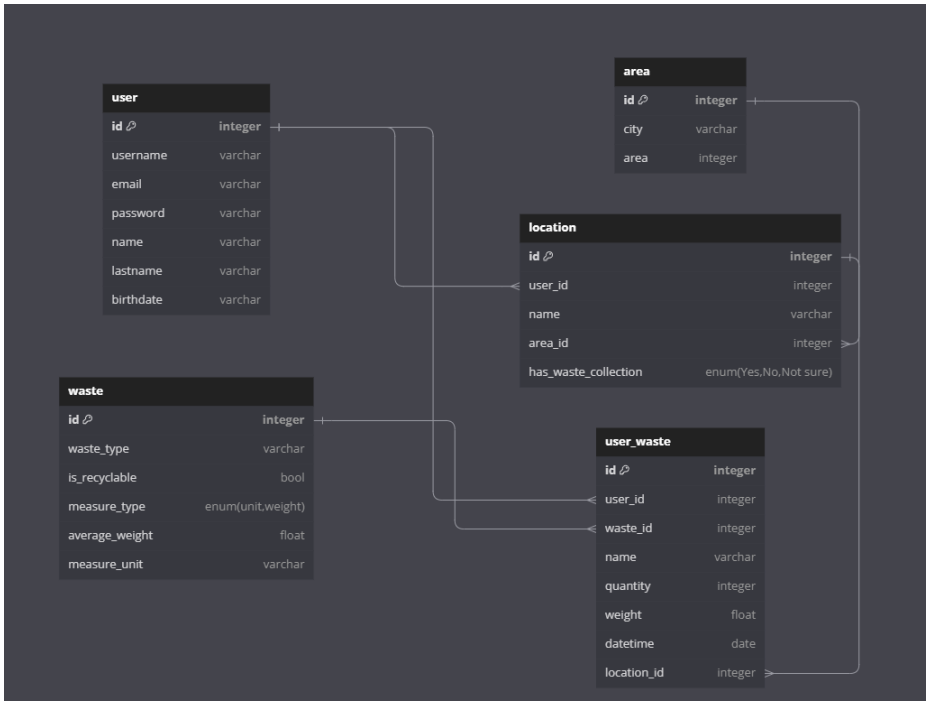


Figura 5.12. Diagrama entidad-relación de la cuarta iteración

5.16.2. *API*: ampliación de controladores

En esta iteración se implementaron diversas operaciones clave para asegurar un manejo adecuado de las operaciones del *API* y optimizar su funcionalidad.

Durante este proceso, se ampliaron las funciones de los controladores de cada uno de los modelos, atendiendo a los diferentes casos de uso previstos en la gestión de datos de los usuarios. Esto incluyó la incorporación de validaciones para cada entrada en las rutas del *API*, además de la implementación de un manejo de errores robusto y el uso de códigos de respuesta HTTP específicos para cada situación, asegurando así una comunicación clara y precisa con los clientes de la *API*.

Se implementaron también funciones de *hash* para almacenar las contraseñas de forma segura, preservando la integridad y privacidad de los datos de los usuarios. Mediante el uso de *middleware*, se estandarizaron las validaciones necesarias para las entradas de datos de los usuarios, logrando así una mayor consistencia y control en los datos procesados por el sistema.

Durante esta iteración se crearon dos módulos adicionales:

- En la carpeta `analysis`, se desarrolló el archivo `analysisController.js`, encargado de realizar las operaciones principales de análisis. Además, dentro de esta carpeta se incluyó un archivo adicional que proporciona las consultas necesarias al controlador.
- El *middleware* de validación para rutas de acceso se organizó en la carpeta `middleware` permitiendo una estructura más modular y facilitando la interacción de los usuarios con la *API*.

A continuación se presenta un ejemplo de un *query* de análisis de datos:

Listing 5.4. Archivo `userWasteAnalysis.js`

```
// Obtener estadisticas de desechos reciclables y no reciclables por usuario (ultimo mes)
exports.getRecyclableWasteStatsQuery = async (sequelize, userId) => {
    const query = `
        SELECT
            COALESCE(SUM(CASE WHEN w.is_recyclable = TRUE THEN uw.weight ELSE 0 END), 0) AS recyclable_weight,
            COALESCE(SUM(CASE WHEN w.is_recyclable = FALSE THEN uw.weight ELSE 0 END), 0) AS non_recyclable_weight
        FROM user_waste uw
        JOIN waste w ON uw.waste_id = w.id
        WHERE uw.user_id = :userId
        AND uw.datetime >= NOW() - INTERVAL '1 MONTH';
    `;

    const result = await sequelize.query(query, {
        type: QueryTypes.SELECT,
        replacements: { userId }
    });

    return {
        recyclable_weight: result[0]?.recyclable_weight || 0,
        non_recyclable_weight: result[0]?.non_recyclable_weight || 0
    };
};
```

5.16.3. Análisis de datos: creación de métricas iniciales

Para la **cuarta iteración** del análisis de datos, se consideró la estructura base de la aplicación con el objetivo de realizar un análisis sobre cómo la información almacenada podría proporcionar métricas significativas. Gracias a la colaboración de la Unidad de Reciclaje de la Municipalidad de Guatemala, se definieron las principales métricas que permitirían demostrar a los usuarios la importancia del reciclaje y la correcta clasificación de los desechos.

A continuación se presenta una foto de la reunión donde se mostró la estructura y se explicaron los datos que se guardaban de los usuarios:

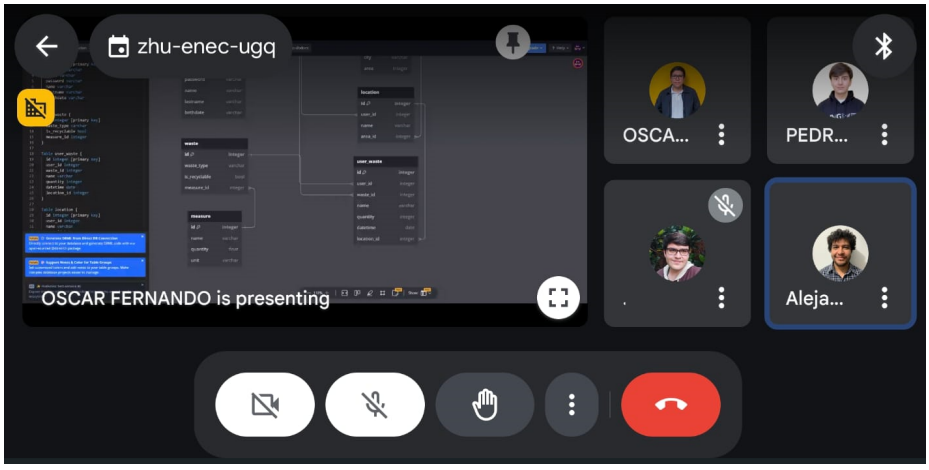


Figura 5.13. Reunión para Estructurar el Análisis de Datos

Durante este proceso, se identificaron las siguientes métricas clave:

- **Análisis de la cantidad de agua ahorrada**: calcula el impacto positivo en términos de ahorro de agua derivado de la correcta clasificación y reciclaje.
- **Análisis de la cantidad de emisiones de CO_2**: evalúa la reducción de emisiones de CO_2 lograda mediante las prácticas de reciclaje.
- **Análisis por ubicaciones del usuario**: proporciona información detallada del impacto de reciclaje basado en las distintas ubicaciones registradas por el usuario.

5.17. Quinta iteración

5.17.1. Base de datos: creación de tabla para tipos de desecho

En la **quinta iteración**, se realizó un cambio importante en la estructura de la base de datos al agregar una nueva tabla denominada ***waste_ type***, que tiene como objetivo principal mejorar la capacidad de análisis de los datos relacionados con los tipos de residuos que maneja la aplicación. Este cambio fue motivado por la necesidad de tener una estructura más clara y eficiente para realizar análisis de impacto ambiental y para generar métricas más precisas que ayuden a los usuarios a tomar mejores decisiones sobre la gestión de sus residuos.

Esta nueva tabla almacena información detallada sobre los tipos de residuos con los que los usuarios pueden interactuar en la aplicación. Los campos principales incluyen:

- `id`: identificador único para cada tipo de residuo.
- `type_name`: nombre del tipo de residuo (por ejemplo, plástico, vidrio, orgánico, etc.).
- `water_savings_index`: indice de ahorro de agua asociado a este tipo de residuo, que permite calcular el impacto positivo en términos de ahorro de agua cuando el residuo es gestionado correctamente.
- `co2_emission_index`: indice de emisiones de CO_2 asociado a la gestión de este tipo de residuo, utilizado para calcular el impacto ambiental en términos de emisiones de carbono.

A continuación se presenta el diagrama actualizado de la estructura de la base de datos en la quinta iteración:

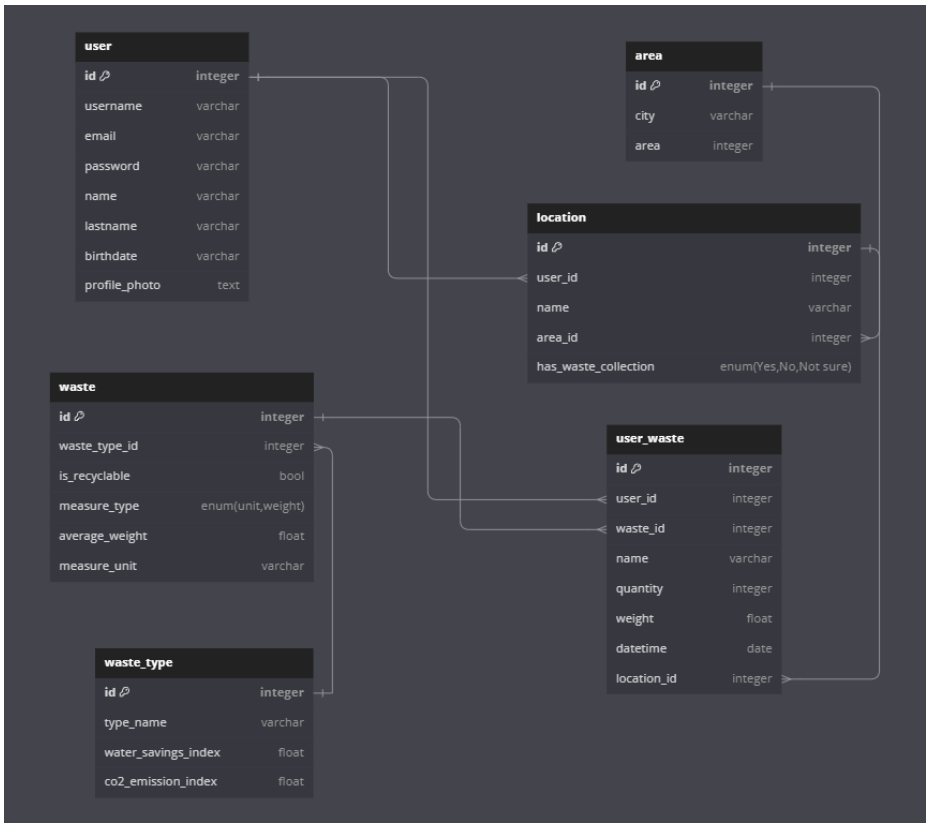


Figura 5.14. Diagrama entidad-relación de la quinta iteración

5.17.2. *API*: implementación de autenticación con *JWT*

Durante la **quinta iteración**, los esfuerzos se centraron en implementar autenticación para cada una de las rutas del *API*, utilizando *JSON Web Tokens* (*JWT*) para mejorar los procesos de seguridad en las peticiones hacia el sistema. Este mecanismo asegura que solo los usuarios autenticados puedan acceder a los recursos y realizar operaciones, fortaleciendo así la protección de los datos.

El enfoque principal consistió en configurar un *middleware* de autenticación que verificara el *token* generado durante el proceso de inicio de sesión. Este *token* se incluye en la cabecera de cada solicitud y es validado por el *middleware* antes de permitir el acceso a las rutas. La configuración permite un control de acceso específico para cada usuario, ya que el *token* contiene información única que identifica al usuario y sus permisos.

Si una solicitud no contiene el *token* en la cabecera, o si el *token* es inválido o ha expirado, el sistema responde con un error de autenticación, evitando el acceso no autorizado a la información y resguardando los datos sensibles. Esta implementación asegura que cada recurso del *API* esté disponible únicamente para los usuarios que han pasado por el proceso de autenticación, manteniendo la privacidad y la integridad de los datos.

Este enfoque también establece una estructura flexible y escalable para futuras ampliaciones de seguridad, permitiendo la incorporación de diferentes niveles de acceso y políticas de autorización adicionales en próximas iteraciones.

A continuación se presenta la función de autenticación:

Listing 5.5. Archivo `jwtAuth.js`

```
// middleware/jwtAuth.js
const jwt = require('jsonwebtoken');

const jwtAuth = (req, res, next) => {
    // Definir las rutas que deseas excluir del middleware de autenticacion
    const excludedPaths = ['/auth/token', '/user/login'];
    // Excluir por rutas o por el tipo de entorno
    if (excludedPaths.includes(req.path) || process.env.NODE_ENV !== '
        production') {
        return next(); // Omitir autenticacion para estas rutas
    }

    // Obtener el encabezado de autorizacion
    const authHeader = req.header('Authorization');
    // Verificar si el encabezado existe y tiene el formato adecuado
    if (!authHeader?.startsWith('Bearer ')) {
        return res.status(401).json({ error: 'Authorization header missing
            or malformed' });
    }

    // Extraer el token del encabezado
    const token = authHeader.replace('Bearer ', '');
    // Verificar el token JWT
    try {
        const decoded = jwt.verify(token, process.env.JWT_SECRET);
        req.user = decoded; // Agregar los datos decodificados del usuario
        next(); // Pasar al siguiente middleware o ruta
    } catch (error) {
        // Manejar errores especificos de JWT
        if (error.name === 'TokenExpiredError') {
            return res.status(401).json({ error: 'Token expired' });
        }
        if (error.name === 'JsonWebTokenError') {
            return res.status(401).json({ error: 'Invalid token' });
        }
        // Otro tipo de error
        return res.status(500).json({ error: 'Failed to authenticate token'
            });
    }
};
module.exports = jwtAuth;
```

5.17.3. Análisis de datos: diseño de funciones principales

Durante la **quinta iteración** del análisis de datos se implementaron cambios significativos en la estructura de la base de datos, permitiendo así la realización de cálculos precisos sobre la reducción de emisiones de CO_2 y el ahorro de agua mediante el reciclaje y la clasificación de residuos. Este enfoque se inspiró en metodologías empleadas en países de Europa y América Latina, donde se aplica un factor específico al peso y al tipo de material de los residuos clasificados para evaluar el impacto ambiental.

Este enfoque resultó fundamental, ya que permitió agregar a la estructura de la base de datos una nueva tabla diseñada para el análisis específico de los materiales reciclados y clasificados. Además, proporciona flexibilidad para realizar estudios de impacto ambiental sobre las zonas donde se

encuentran los usuarios, evaluando así cómo las buenas prácticas de gestión de residuos contribuyen a la sostenibilidad en distintas ubicaciones.

A continuación se demuestra el cálculo que se toma en cuenta para cada métrica:

Ahorro de agua:

$$\text{Ahorro de agua} = \text{Peso de la basura} \times \text{Factor de conversión de agua} \tag{5.1}$$

Emisiones de CO_2:

$$\text{Emisiones de } CO_2 = \text{Peso de la basura} \times \text{Factor de conversión de } CO_2 \tag{5.2}$$

5.18. Sexta iteración

5.18.1. Base de datos: creación de tabla de bitácora de auditoría

En la **sexta iteración**, se introdujo una tabla adicional denominada ***audit_log***, cuyo propósito es servir como una bitácora de auditoría. Este cambio responde a la necesidad de llevar un registro detallado de las operaciones críticas realizadas en la base de datos, es fundamental para garantizar la integridad de los datos y para identificar posibles problemas relacionados con el acceso no autorizado o modificaciones incorrectas.

La bitácora de auditoría no solo facilita el seguimiento de los cambios, sino que también se convierte en una herramienta crucial para el cumplimiento de normativas y estándares de seguridad, especialmente en sistemas que manejan información sensible. Al mantener un historial de operaciones, se puede revisar en cualquier momento quién realizó una modificación, en qué momento y qué datos se vieron afectados, lo cual es esencial para el mantenimiento de la confiabilidad del sistema.

- `id`: identificador único de cada registro de auditoría.
- `operation_type`: indica el tipo de operación realizada (*CREATE, UPDATE, DELETE*).
- `table_name`: nombre de la tabla en la cual se realizó la operación.
- `record_id`: identificador del registro afectado por la operación.
- `old_values`: almacena los valores anteriores a la operación (en caso de actualizaciones o eliminaciones).
- `new_values`: almacena los nuevos valores después de la operación (en caso de creaciones o actualizaciones).
- `performed_by`: identificador del usuario que realizó la operación.
- `created_at`: marca temporal que indica cuándo se realizó la operación.

A continuación se presenta el diagrama actualizado de la estructura de la base de datos en la sexta iteración:

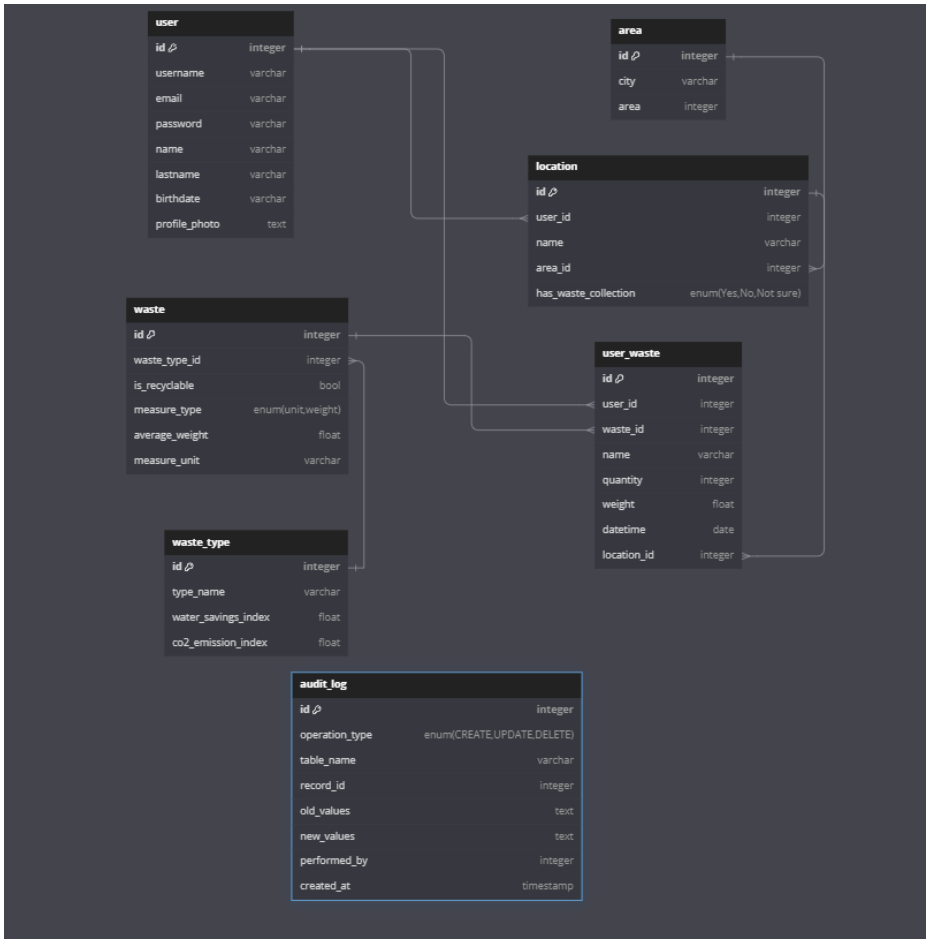


Figura 5.15. Diagrama entidad-relación de la sexta iteración

5.18.2. *API*: implementación de registros en bitácora de auditoría

En la **sexta iteración**, se integraron funciones adicionales para gestionar el registro de auditoría mediante el modelo ***audit_ log***. El objetivo principal de esta iteración fue mejorar la trazabilidad de las operaciones realizadas en el sistema, permitiendo registrar todas las acciones *CRUD* en las entidades clave de la base de datos para un seguimiento detallado de cada cambio.

Para implementar esta funcionalidad, se creó un nuevo módulo denominado `utils`, en el cual se definió una función principal llamada `logAudit`. Esta función se encarga de realizar el registro de cada operación de agregación, actualización y eliminación realizada en el *API*, generando un historial de cambios y facilitando la supervisión de las actividades del sistema.

Se realizaron actualizaciones en cada uno de los controladores para integrar esta funcionalidad, habilitando así el almacenamiento automático de los registros en el cuadro `audit_log` y garantizando que los datos se registren de forma consistente en cada operación relevante.

A continuación se presenta un ejemplo de la función:

Listing 5.6. Archivo `auditLogger.js`

```
1 // utils/auditLogger.js
2 const AuditLog  = require('../models/AuditLog');
```

```
3
4  // Funcion para registrar las operaciones en el cuadro audit_log
5  exports.logAudit = async (operation, table, recordId, oldValues, newValues,
       performedBy) => {
6      try {
7          await AuditLog.create({
8              operation_type: operation,
9              table_name: table,
10             record_id: recordId,
11             old_values: oldValues ? JSON.stringify(oldValues) : null,
12             new_values: newValues ? JSON.stringify(newValues) : null,
13             performed_by: performedBy
14         });
15     } catch (error) {
16         console.error('Error logging audit:', error);
17     }
18 };
```

5.18.3. Análisis de datos: diseño de funciones de análisis

Durante la **sexta iteración**, se diseñaron nuevas funciones que permiten proporcionar a los usuarios datos detallados sobre sus comportamientos y patrones de consumo en relación con el reciclaje y la gestión de residuos. Estas funciones buscan informar a los usuarios sobre sus prácticas actuales y fomentar una mayor conciencia ambiental a través de un seguimiento preciso de sus hábitos.

En esta iteración se propuso un conjunto de métricas y funciones clave que se aplicarían al análisis de datos, incluyendo:

- **Análisis de la cantidad de desechos reciclables y no reciclables**: proporciona un desglose de los residuos en función de la forma en la que fueron clasificados en el registro.
- **Análisis de los tipos de desechos más consumidos por el usuario**: identifica los materiales que el usuario consume en mayor medida, facilitando el reconocimiento de hábitos específicos.
- **Análisis del promedio diario de desechos en la última semana**: calcula el promedio diario de residuos gestionados por el usuario en los últimos siete días.
- **Análisis del promedio diario de desechos en el último mes**: proporciona una visión más amplia del consumo promedio diario de desechos en el último mes.
- **Porcentaje de consumo con respecto a la última semana**: evalúa el porcentaje de consumo actual del usuario en relación con el promedio de la semana anterior, permitiendo una comparación rápida de cambios en el comportamiento de gestión de residuos.

Porcentaje del consumo del día con respecto a la semana anterior:

$$\text{Porcentaje de Consumo} = \left(\frac{\text{Consumo del Día Actual}}{\text{Promedio de Consumo de la Semana Anterior}} \right) \times 100 \tag{5.3}$$

5.19. Séptima iteración

5.19.1. Base de datos: optimización y mejora de la estructura

En la **séptima iteración**, se realizaron ajustes adicionales en las tablas ***waste*** y ***user_waste***, enfocados en simplificar la estructura de los datos, almacenando únicamente la información necesaria para llevar a cabo las acciones requeridas y realizar un análisis más eficiente de la gestión de residuos por parte de los usuarios. Estos cambios fueron impulsados por la necesidad de optimizar el rendimiento del sistema, eliminando cualquier redundancia y asegurando que solo se almacenaran los datos esenciales. Al reducir la cantidad de datos almacenados en esta tabla, se optimizó la capacidad de realizar consultas más rápidas sobre los registros de desechos del usuario.

Se eliminaron los campos `measure_type` y `measure_unit`, que anteriormente se usaban para gestionar las unidades de medida de los residuos. Estos campos ya no eran necesarios, dado que la estructura del sistema fue simplificada para centrarse en los tipos de residuos y sus relaciones con las métricas ambientales almacenadas en el cuadro `waste_type`.

Se eliminó el campo `quantity`, que anteriormente almacenaba la cantidad de residuos generados por los usuarios. Ahora, únicamente se mantiene el campo `weight`, que almacena el peso exacto del residuo, ya que este es el dato más relevante para el análisis de la huella de carbono y las métricas de ahorro de recursos.

Este ajuste permite simplificar el cálculo de métricas, ya que los análisis se enfocan en el peso total de los residuos gestionados, eliminando la necesidad de manejar cantidades adicionales que podrían generar complejidad en el procesamiento de los datos.

A continuación se presenta el diagrama actualizado de la estructura de la base de datos en la séptima iteración:

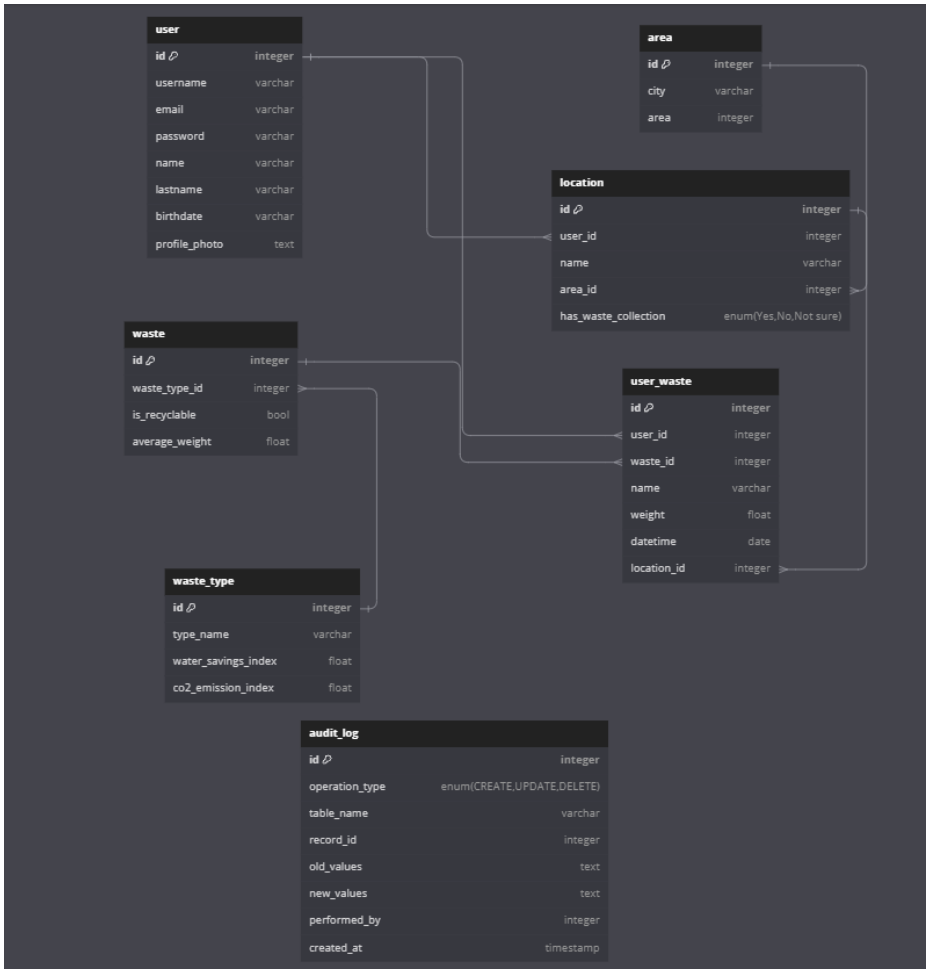


Figura 5.16. Diagrama entidad-relación de la séptima iteración

5.19.2. *API*: optimización y mejora del *API*

La **séptima iteración** se enfocó en ajustar los controladores y las rutas en función del diseño final de la base de datos, implementando las operaciones necesarias para desarrollar los controladores de análisis de datos y adaptando la estructura del sistema para una mayor coherencia y funcionalidad.

En esta fase, se realizaron cambios para alinear los modelos y relaciones de acuerdo con el esquema final. Estas modificaciones incluyeron ajustes en las asociaciones entre tablas, garantizando que los modelos reflejaran con precisión la nueva lógica y soportaran las consultas complejas necesarias para el análisis de datos.

Además, se implementó una función específica para obtener el *token* de autenticación de un usuario con privilegios administrativos. Esta funcionalidad permite el acceso seguro a rutas especiales y prepara el sistema para futuras integraciones con un tablero de visualización de datos. Con esta visión, se espera que el *API* pueda conectarse fácilmente a plataformas de análisis de datos en el futuro, proporcionando un flujo de datos seguro y directo hacia herramientas de visualización para el monitoreo y análisis en tiempo real.

Con esta iteración, el desarrollo del *API* se considera completo, cumpliendo con los requisitos establecidos y proporcionando una estructura sólida y escalable que soporta tanto la seguridad como

la integridad de los datos.

5.19.3. Análisis de datos: creación de las rutas de análisis en *API*

Durante esta iteración, los esfuerzos se centraron en adaptar y replicar las funciones y métricas en la *API*, permitiendo que los usuarios pudieran consultar y visualizar estos datos de manera accesible y dinámica.

Para lograr este objetivo, fue necesario desarrollar funciones en *SQL* que extrajeran datos específicos de cada usuario, asegurando que la información fuera precisa y relevante. Estas funciones de consulta se integraron en el *API* para que los datos se procesaran de manera eficiente y segura.

Además, se implementaron funciones auxiliares que ejecutan las consultas y procesan los resultados, proporcionando un flujo de datos optimizado para cada métrica definida. Finalmente, se crearon controladores específicos para cada una de las funciones de análisis propuestas, facilitando el acceso a través de rutas en la *API* y permitiendo una interacción fluida con la interfaz de usuario.

Este enfoque garantiza que todas las métricas y funciones analíticas puedan ser consultadas en tiempo real por los usuarios, proporcionando información clara y detallada sobre sus hábitos de consumo y reciclaje.

A continuación se muestra un ejemplo de la implementación de una de las funciones de análisis de datos:

Listing 5.7. Archivo `userWasteAnalysis.js`

```
// Obtener el ahorro de emisiones de di xido de carbono para el usuario (
    ltimo  mes)
exports.getCO2SavingsQuery = async (sequelize, userId) => {
    const query = `
        SELECT
            COALESCE(SUM(uw.weight * wt.co2_emission_index), 0) AS
               total_co2_savings
        FROM user_waste uw
        JOIN waste w ON uw.waste_id = w.id
        JOIN waste_type wt ON w.waste_type_id = wt.id
        WHERE uw.user_id = :userId
        AND uw.datetime >= NOW() - INTERVAL '1 MONTH';
    `;

    const result = await sequelize.query(query, {
        type: QueryTypes.SELECT,
        replacements: { userId }
    });

    return result[0]?.total_co2_savings || 0;
};
```

5.20. Octava iteración

5.20.1. Análisis de datos: creación de análisis de datos global de los usuarios

Durante esta última iteración del análisis de datos, los esfuerzos se centraron en proporcionar un análisis general sobre los usuarios y sus datos relacionados con la gestión de desechos. Para ello, se adaptaron y extendieron las funciones previamente utilizadas para análisis individual, modificándolas para calcular métricas y estadísticas a nivel global.

Para facilitar una visualización clara y comprensible de estos resultados agregados, y con el objetivo de analizar los comportamientos y registros de una población específica, se decidió implementar un tablero en ***Power BI***. Este tablero se conecta directamente con el *API*, permitiendo presentar los datos en gráficos interactivos y visualizaciones fáciles de interpretar. De esta forma, se simplifica el análisis y seguimiento de los datos, brindando una herramienta eficaz para identificar tendencias y patrones en la gestión de desechos.

A continuación se presenta un ejemplo de las funciones globales:

Listing 5.8. Archivo `userWasteAnalysis.js`

```
// Obtener estadisticas de desechos reciclables y no reciclables de todos
    los usuarios (ultimo mes)
exports.getRecyclableWasteStatsQueryAll = async (sequelize) => {
    const query = `
        SELECT
            COALESCE(SUM(CASE WHEN w.is_recyclable = TRUE THEN uw.weight
                ELSE 0 END), 0) AS recyclable_weight,
            COALESCE(SUM(CASE WHEN w.is_recyclable = FALSE THEN uw.weight
                ELSE 0 END), 0) AS non_recyclable_weight
        FROM user_waste uw
        JOIN waste w ON uw.waste_id = w.id
        WHERE uw.datetime >= NOW() - INTERVAL '1 MONTH';
    `;

    const result = await sequelize.query(query, {
        type: QueryTypes.SELECT
    });

    return {
        recyclable_weight: result[0]?.recyclable_weight || 0,
        non_recyclable_weight: result[0]?.non_recyclable_weight || 0
    };
};
```

Y se presenta el ejemplo de conexión por medio de *Power BI*:

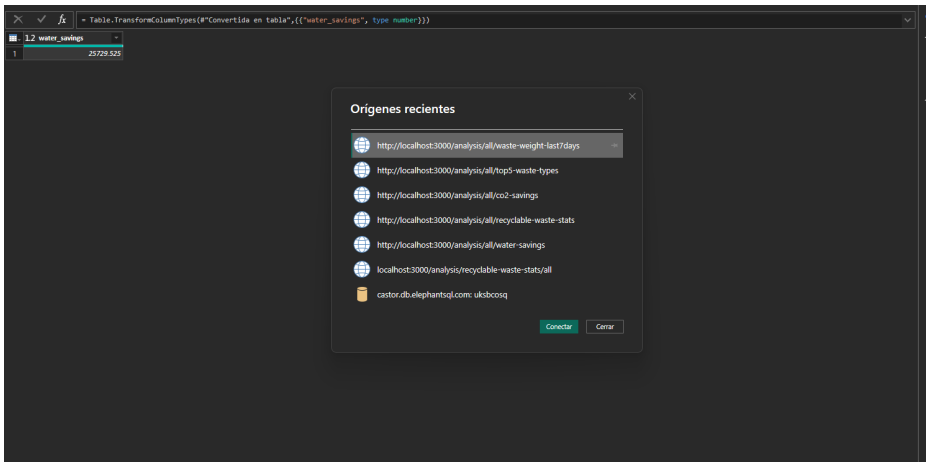


Figura 5.17. Ejemplo de solicitudes en origines de conexión

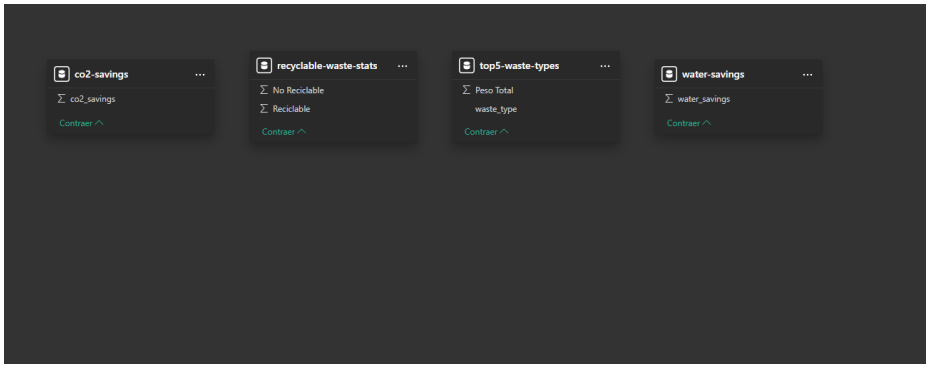


Figura 5.18. Tablas resultantes de la conexión del *API*

5.21. Implementación de *CI/CD*

Al completar las tres principales secciones necesarias para almacenar, gestionar y proporcionar datos a los usuarios, los esfuerzos se orientaron hacia la creación de una base sólida para el desarrollo y expansión futuros del *backend*. Mediante un análisis exhaustivo, se decidió adoptar buenas prácticas de desarrollo de *software*, las cuales contribuyen a mantener un código limpio, modular y sostenible a largo plazo.

Por lo tanto, se implementó un *pipeline* de Integración y Despliegue Continuo (*CI/CD*), permitiendo la automatización de pruebas, la detección temprana de errores y un proceso de despliegue optimizado. Esta estructura facilita una gestión eficiente del proyecto, al tiempo que prepara el sistema para futuras iteraciones y ampliaciones de la aplicación. Con estas prácticas, se estableció una infraestructura robusta que no solo agiliza el desarrollo actual, sino que también sienta las bases para un crecimiento escalable y una evolución continua de la aplicación.

Ejemplo del *pipeline* de *CI/CD*:

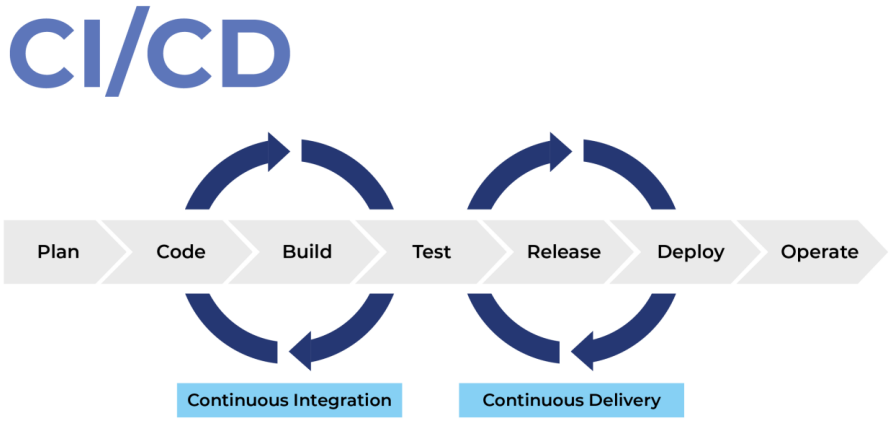


Figura 5.19. *Pipeline* de *CI/CD* [9]

5.21.1. Pruebas automatizadas

La primera etapa para la implementación del *pipeline CI/CD*, una vez finalizado el desarrollo del código, fue la de construcción y prueba. Esta etapa es fundamental para verificar el correcto funcionamiento del sistema después de cada integración de código, permitiendo identificar y corregir cualquier error que pudiera surgir. La construcción asegura que el entorno y las dependencias se configuren adecuadamente, mientras que la fase de pruebas valida la estabilidad y fiabilidad del código.

Durante esta etapa, se diseñaron, implementaron y configuraron tres tipos de pruebas fundamentales para obtener información detallada sobre el comportamiento del *API* y asegurar su fiabilidad en la gestión y procesamiento de datos. Estas pruebas permiten evaluar el rendimiento del sistema bajo diversas condiciones de uso, verificando su capacidad para operar de manera consistente y precisa.

Pruebas unitarias

Durante esta etapa de pruebas, se desarrollaron y estructuraron pruebas unitarias específicas para validar el funcionamiento de las funciones clave del *API*. Estas pruebas se centraron principalmente en verificar el comportamiento de los controladores encargados de gestionar la información, asegurando que cada controlador procesara los datos de forma precisa y consistente.

Se definieron diversos escenarios para simular posibles situaciones de uso, incluyendo casos en los que el usuario proporciona valores incorrectos o fuera de los parámetros esperados. En estos casos, las pruebas unitarias evaluaron cómo los controladores respondían a entradas erróneas, tomando en cuenta el tipo de error, la validación de los datos, y los mensajes de respuesta generados. Esto permitió identificar y manejar de forma efectiva los errores, asegurando que el sistema ofrezca una respuesta confiable

Para construir estas pruebas, se utilizaron varias dependencias de *Node.js* que facilitaron las simulaciones necesarias para verificar el correcto funcionamiento de los controladores. La lista de dependencias empleadas es la siguiente:

- ***Sinon.js***: esta herramienta fue fundamental para la simulación de tablas y funciones durante las pruebas. *Sinon* permite realizar *mocks* y *stubs* de funciones y tablas de la base de datos, lo

que facilita la creación de datos de prueba sin necesidad de interactuar con la base de datos real.

- ***Chai.js***: esta biblioteca de aserciones permite comparar los datos de prueba o simulados con los resultados obtenidos de las funciones. Su sintaxis facilita una validación clara y legible de los resultados esperados y reales.
- ***Mocha.js***: este marco de pruebas facilita la organización y ejecución de pruebas unitarias e integración, generando reportes detallados sobre los resultados. En caso de fallos, *Mocha* proporciona mensajes de error específicos que ayudan a identificar los problemas en el código.

A continuación se muestra un ejemplo de una prueba para el controlador de usuario:

Listing 5.9. Archivo `userControllerTest.js`

```
it('Deberia crear un nuevo usuario y registrar auditoria', async () => {
    const req = { body: { email: 'test@example.com', password: 'password123'
        }, user: { id: 999 } };
    const res = { status: sinon.stub().returnsThis(), json: sinon.stub() };
    const userMock = { id: 1, email: 'test@example.com' };

    sinon.stub(User, 'findOne').resolves(null); // Simula que el email no
        esta en uso.
    sinon.stub(User, 'create').resolves(userMock);
    sinon.stub(AuditLog, 'create').resolves(); // Simula la creacion del
        registro de auditoria

    await userController.createUser(req, res);

    expect(res.status.calledWith(201)).to.be.true;
    expect(res.json.calledWith(sinon.match.has('data'))).to.be.true;
    expect(AuditLog.create.calledOnce).to.be.true; // Verifica que se
        registro la auditoria
});
```

Pruebas de integración

Como siguiente paso en el proceso de pruebas, se procedió al diseño y ejecución de las pruebas de integración. Estas pruebas se enfocaron en evaluar la interacción entre las rutas de la *API*, los valores requeridos, y las funciones de los controladores, garantizando una integración coherente y fluida de los distintos componentes del sistema. A través de estas pruebas, se verificaron los flujos completos de datos y las respuestas que recibirán los usuarios al utilizar el *API*, permitiendo validar el funcionamiento del sistema de extremo a extremo.

A diferencia de las pruebas unitarias, que evalúan componentes aislados, las pruebas de integración comprueban la interacción de múltiples elementos, simulando el flujo real que el usuario experimentaría al interactuar con el sistema. Estas pruebas también se diseñaron para confirmar que los mensajes de respuesta, tanto de éxito como de error, se generen de manera clara y consistente. Esto incluye la verificación de que los mensajes de error contengan información útil y adecuada para guiar al usuario en caso de entradas incorrectas, sin exponer detalles técnicos innecesarios.

Para construir estas pruebas, se utilizaron varias dependencias de *Node.js* que facilitaron las simulaciones necesarias para verificar el correcto funcionamiento de las integración de los controladores con las rutas que el *API* brinda. La lista de dependencias empleadas es la siguiente:

- ***Sinon.js***: esta herramienta fue fundamental para la simulación de tablas y funciones durante las pruebas. *Sinon* permite realizar *mocks* y *stubs* de funciones y tablas de la base de datos, lo que facilita la creación de datos de prueba sin necesidad de interactuar con la base de datos real.
- ***Chai.js***: esta biblioteca de aserciones permite comparar los datos de prueba o simulados con los resultados obtenidos de las funciones. Su sintaxis facilita una validación clara y legible de los resultados esperados y reales.
- ***Mocha.js***: este marco de pruebas facilita la organización y ejecución de pruebas unitarias e integración, generando reportes detallados sobre los resultados. En caso de fallos, *Mocha* proporciona mensajes de error específicos que ayudan a identificar los problemas en el código.
- ***Supertest.js***: esta herramienta permite simular las solicitudes *HTTP* realizadas al *API*, proporcionando información detallada sobre el comportamiento de las respuestas y los resultados obtenidos. Es esencial para verificar el funcionamiento y la correcta respuesta del *API* ante diferentes tipos de solicitudes.

A continuación se presenta un ejemplo de prueba de integración:

Listing 5.10. Archivo `userRoutesTest.js`

```
it('Deberia iniciar sesion correctamente', async () => {
    const userMock = { id: 1, username: 'testuser', email: 'test@example.com
        ', password: 'hashedPassword' };

    // Simulamos la busqueda del usuario y la comparacion de passwords
    sinon.stub(User, 'findOne').resolves(userMock);
    sinon.stub(bcrypt, 'compare').resolves(true); // Simulamos que la
        password es valida

    const res = await request(app)
        .post('/api/users/login')
        .send({
            email: 'test@example.com',
            password: 'password123'
        });

    expect(res.status).to.equal(200);
    expect(res.body.success).to.be.true;
    expect(res.body.data.username).to.equal('testuser');
});
```

Pruebas de carga y rendimiento

Para evaluar el comportamiento de la *API* bajo condiciones de alta demanda y monitorear su rendimiento en diversos escenarios, se implementaron pruebas de carga y rendimiento. Estas pruebas permiten analizar la capacidad de respuesta de la *API*, midiendo el tiempo que tarda en procesar solicitudes y la eficacia de sus respuestas bajo distintos niveles de carga. Además, proporcionan información valiosa sobre la estabilidad y escalabilidad del sistema, aspectos fundamentales para asegurar un rendimiento confiable y adecuado.

En estas pruebas, se configuró un entorno de prueba específico, cargando datos representativos en la base de datos y simulando un tráfico de solicitudes similar al uso real que podría experimentar la *API* en un ambiente de producción. Al iniciar la *API* en este entorno, se envían múltiples solicitudes simultáneas, simulando condiciones de carga que varían desde niveles bajos hasta picos de alta

demanda. Esto permite observar cómo el sistema maneja la concurrencia y si existen cuellos de botella que afectan el rendimiento.

Durante las pruebas de carga, se registran métricas clave, como el tiempo promedio de respuesta, el porcentaje de solicitudes exitosas, y los tiempos de espera en diferentes *endpoints*, brindando datos que ayudan a identificar y optimizar los puntos críticos en la arquitectura de la *API*.

Cálculo de solicitudes por segundo para una aplicación móvil

Para determinar la cantidad de solicitudes que la *API* debe soportar por segundo en un entorno de alta demanda, se realizó el siguiente cálculo basado en la población objetivo y un supuesto de uso promedio. Este cálculo se basa en una estimación de la población de la Ciudad de Guatemala en 2022, obtenida de [398].

Paso 1: Cálculo de usuarios activos Dado que la población total es:

$$P = 1,213,651$$

Supongamos una tasa de adopción del 20 % de la población. Esto nos da:

$$U = P \times 0.20$$

Sustituyendo el valor de P:

$$U = 1,213,651 \times 0.20 = 242,730 \text{ usuarios activos}$$

Paso 2: cálculo de solicitudes diarias Si los usuarios realizan en promedio 5 solicitudes por día, la cantidad total de solicitudes al día es:

$$SD = U \times 5$$

Sustituyendo el valor de U:

$$SD = 242,730 \times 5 = 1,213,650 \text{ solicitudes diarias}$$

Paso 3: distribución de solicitudes por segundo Para calcular la cantidad de solicitudes por segundo, dividimos las solicitudes diarias por el número de segundos en un día (24 horas multiplicado por 60 minutos y 60 segundos):

$$SPS = \frac{SD}{24 \times 60 \times 60}$$

Sustituyendo el valor de SD:

$$SPS = \frac{1,213,650}{24 \times 60 \times 60} \approx 14 \text{ solicitudes por segundo}$$

Este cálculo permite establecer un objetivo claro para las pruebas de carga, asegurando que la *API* esté preparada para manejar un promedio de 14 solicitudes por segundo en condiciones de alta demanda. Esta preparación garantiza que el sistema pueda escalar de manera eficiente, ofreciendo un rendimiento estable y óptimo para los usuarios en producción.

A continuación se muestra el ejemplo de una prueba para las rutas de análisis de datos:

Listing 5.11. Archivo `performance.yml`

```
config:
  target: 'http://localhost:3000'
  phases:
```

```
    - duration: 60  # Duracion de la prueba en segundos
      arrivalRate: 20  # Numero de usuarios simulados por segundo
  defaults:
    headers:
      content-type: "application/json"

scenarios:
  - name: Test de analisis de desechos reciclables
    flow:
      - get:
          url: "/analysis/recyclable-waste/1"
  - name: Test de las 5 principales ubicaciones por desechos
    flow:
      - get:
          url: "/analysis/top5-locations/1"
  - name: Test de ahorro de agua
    flow:
      - get:
          url: "/analysis/water-savings/1"
  - name: Test de ahorro de CO2
    flow:
      - get:
          url: "/analysis/co2-savings/1"
  - name: Test de los 5 tipos de desechos mas comunes
    flow:
      - get:
          url: "/analysis/top5-waste-types/1"
  - name: Test de peso de desechos de los ultimos 7 dias
    flow:
      - get:
          url: "/analysis/waste-last7days/1"
  - name: Test de comparacion de desechos de hoy con el promedio mensual
    flow:
      - get:
          url: "/analysis/compare-waste-today/1"
```

5.21.2. Manejo y monitoreo de errores con *Sentry*

Para asegurar un monitoreo continuo del rendimiento y detectar errores en el funcionamiento de la *API*, se decidió integrar una herramienta especializada: *Sentry*. *Sentry* permite realizar un seguimiento exhaustivo de los errores en tiempo real, capturando información detallada sobre los fallos y excepciones que puedan surgir en el entorno de producción, los cuales podrían no haberse detectado en las pruebas iniciales.

Durante el proceso de integración de *Sentry*, se configuraron diversas funcionalidades de monitoreo, como el *profiling*, que permite analizar el rendimiento y los tiempos de respuesta en tiempo real, y el monitoreo automático de errores, que facilita la detección de fallos en el sistema. Sin embargo, debido a que la estructura de los controladores ya capturaba los errores de forma interna, estos no generaban mensajes que *Sentry* pudiera interceptar automáticamente, lo cual limitaba la visibilidad de los problemas en la herramienta de monitoreo.

Para solucionar este inconveniente, se implementó una función personalizada, `captureError`, en cada controlador donde se manejaban errores. Esta función permite enviar manualmente a *Sentry* la información relevante sobre cada error, incluso cuando este es capturado y manejado en el código. Al utilizar `captureError`, se asegura que los errores que ocurren en la *API* sean registrados en *Sentry*

con el contexto completo del fallo, incluyendo detalles como el controlador específico, el *endpoint* afectado, y los datos de la solicitud.

Para realizar la configuración de *Sentry* se hicieron las configuraciones correspondientes en la plataforma.

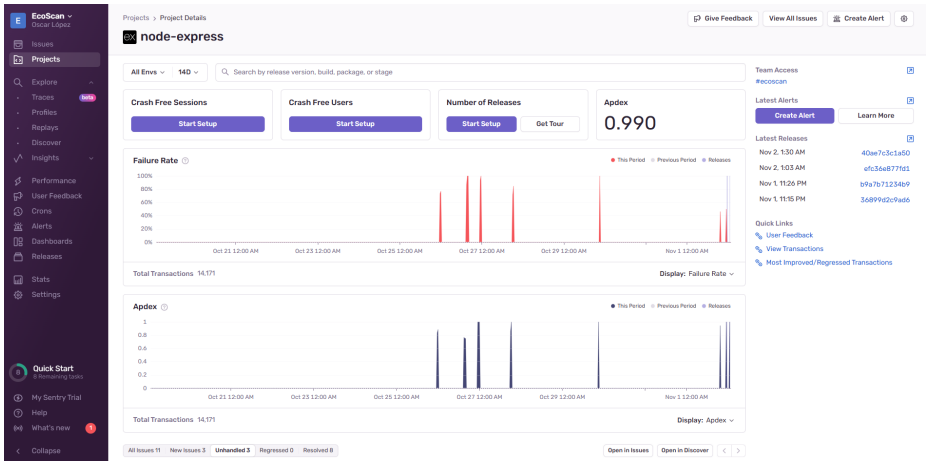


Figura 5.20. Tablero del proyecto en *Sentry*

Y se realizaron las configuraciones correspondientes en el *API*, para poder conectar la herramienta y capturar los errores.

Configuración principal de *Sentry* en el proyecto:

Listing 5.12. Archivo `instrument.js`

```
// middleware/instrument.js
require('dotenv').config(); // Cargar las variables de entorno

const Sentry = require("@sentry/node");
const { nodeProfilingIntegration } = require("@sentry/profiling-node");

Sentry.init({
    dsn: process.env.SENTRY_DSN, // Usar la variable de entorno
    integrations: [
        nodeProfilingIntegration(),
    ],
    tracesSampleRate: 1.0,
    profilesSampleRate: 1.0,
});
```

Ejemplo de captura de errores:

Listing 5.13. Archivo `captureError.js`

```
// midleware/captureError.js
const Sentry = require("@sentry/node");

// Funcion para capturar y manejar errores con Sentry
const captureError = (error) => {
    // Verificar si el entorno no es de pruebas
    if (process.env.NODE_ENV !== 'test') {
        // Capturar el error con Sentry solo si no es un entorno de pruebas
```

```
        Sentry.captureException(error);
    }
};

module.exports = captureError;
```

5.21.3. Implementación y automatización con *GitHub Actions*

Por último, en el proceso de implementación de un *Pipeline* de *CI/CD* se consideró la automatización de tareas clave para asegurar la calidad y el correcto funcionamiento del *API*. En este caso, se utilizó ***GitHub Actions*** debido a su integración con el repositorio en *GitHub*, donde se encuentra almacenado el código principal del *API*. Esta integración permite que, al realizar cambios en las funcionalidades del *API*, dichos cambios se sometan automáticamente a un conjunto de pruebas, asegurando que las nuevas implementaciones o modificaciones mantengan la estabilidad y calidad del sistema.

La configuración del *pipeline* se llevó a cabo mediante la creación de *workflows* en *GitHub*, los cuales definen los pasos necesarios para instalar dependencias, ejecutar pruebas, y generar reportes.

Para la ejecución, se configuraron dos *workflows*. En el primero, se llevan a cabo las pruebas unitarias y de integración, las cuales validan la funcionalidad de cada componente del *API* y verifican la interacción entre los distintos módulos del sistema. En el segundo *workflow*, se ejecutan las pruebas de carga y rendimiento, evaluando la capacidad del *API* para manejar múltiples solicitudes simultáneas y midiendo su desempeño bajo diferentes niveles de carga.

A continuación se muestra una parte de las configuraciones para los dos flujos:

Listing 5.14. Archivo `test-deploy.yml`

```
- name: Navigate to eco-\textit{scan}-backend folder
  run: cd eco-\textit{scan}-backend

- name: Install dependencies
  run: npm install
  working-directory: ./eco-\textit{scan}-backend

- name: Set up environment variables
  run: |
    // Realizar la configuracion de variables de entorno
  working-directory: ./eco-\textit{scan}-backend

- name: Wait for Postgres
  run: |
    until pg_isready -h localhost -p 5432 -U postgres; do echo waiting for
        postgres; sleep 5; done

- name: Run tests
  run: npm run test
  working-directory: ./eco-\textit{scan}-backend

- name: Show logs if test fails
  if: failure()
  run: cat ./eco-\textit{scan}-backend/npm-debug.log
```

Listing 5.15. Archivo `test-deploy.yml`

```yaml
deploy:
needs: test
runs-on: ubuntu-latest
if: github.ref == 'refs/heads/master'

steps:
- name: Checkout code
  uses: actions/checkout@v3

- name: Set up Node.js
  uses: actions/setup-node@v3
  with:
    node-version: '20'

- name: Deploy to Elastic Beanstalk
  env:
    AWS_ACCESS_KEY_ID: ${{ secrets.AWS_ACCESS_KEY_ID }}
    AWS_SECRET_ACCESS_KEY: ${{ secrets.AWS_SECRET_ACCESS_KEY }}
    AWS_REGION: 'us-east-1'
  run: |
    zip -r deploy.zip ./*
    aws elasticbeanstalk create-application-version --application-name "
        appName" --version-label version-1 --source-bundle S3Bucket="s3Name
        ",S3Key="deploy.zip"
    aws elasticbeanstalk update-environment --application-name "appName" --
        environment-name "appEnv" --version-label version-1
  working-directory: ./eco-\textit{scan}-backend
```

5.22. Metodología de infraestructura en la nube

Para poder proporcionar las herramientas necesarias para la ejecución de la aplicación en la nube, se siguieron los siguientes pasos fundamentales que permitieron estructurar y poner en marcha un entorno de trabajo óptimo y escalable. Esta metodología asegura que cada fase del proyecto cuenta con una planificación estratégica y que los recursos se aprovechan de la manera más eficiente posible. El proceso se organizó en los siguientes apartados principales:

- **Identificación de los recursos necesarios**: en esta fase, se realizó un análisis detallado de los requerimientos técnicos y operativos del proyecto. Se identificaron los tipos de servicios necesarios, como bases de datos, almacenamiento, balanceadores de carga, y herramientas de monitoreo, para asegurar que la arquitectura en la nube pudiera cumplir con los requisitos de rendimiento, escalabilidad y seguridad. Además, se evaluaron las necesidades específicas de desarrollo y las dependencias con otras aplicaciones o servicios.

- **Comparación de proveedores de nube**: una vez definidos los recursos necesarios, se procedió a comparar diferentes proveedores de nube para seleccionar la opción más adecuada. Se analizaron aspectos como costos, disponibilidad de servicios, soporte técnico, seguridad, y opciones de personalización. Entre los proveedores evaluados se incluyeron opciones como *AWS, Azure*, y *Google Cloud Platform*, considerando también características de sus programas de sostenibilidad y sus servicios de alta disponibilidad.

- **Implementación y configuración de recursos**: con el proveedor seleccionado, se llevó a cabo la implementación de los servicios en la nube. Esta etapa incluyó la creación y configuración de instancias de servidores, bases de datos, y redes virtuales. Cada recurso fue configurado de

acuerdo con las mejores prácticas recomendadas por el proveedor, asegurando así un ambiente de alta disponibilidad y resiliencia ante posibles fallos.

- **Análisis de costos y mantenimiento**: finalmente, se realizó un análisis de costos detallado para prever el gasto asociado con el uso de los recursos en la nube. Se identificaron los principales factores de costos y se implementaron prácticas de optimización, como el escalado automático y el monitoreo del uso de recursos, para reducir gastos.

A continuación, se explica en detalle cada apartado para profundizar en su relevancia y ejecución en el proyecto.

5.23. Identificación de los recursos necesarios

5.23.1. Evaluación de las pantallas

Este análisis de necesidades comenzó con una revisión de las pantallas desarrolladas en el prototipo de *Figma*, que representan las interacciones clave del usuario con la aplicación. Las pantallas destacadas incluyen los módulos de autenticación (*login* y *signin*), opciones de usuario, visualización de estadísticas, formulario de ingesta de datos y pantallas de inteligencia artificial, como se ilustra en las siguientes figuras:

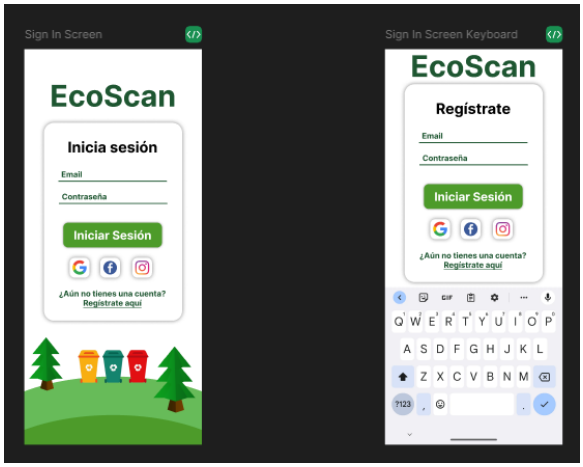


Figura 5.21. Pantallas *login* - *signin*

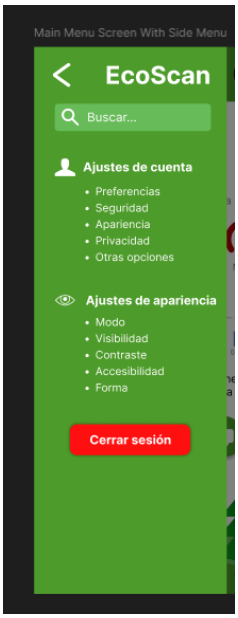


Figura 5.22. Opciones usuario

Figura 5.23. Pantalla estadísticas

Figura 5.24. Formulario de ingesta

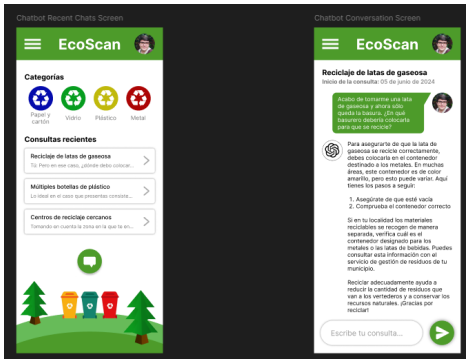


Figura 5.25. Pantallas Inteligencia artificial

5.23.2. Requerimientos derivados del prototipo

A partir de estas pantallas, se definieron necesidades específicas para implementar servicios de soporte a las funcionalidades principales:

- **Servicios de autenticación**: con el objetivo de proteger el acceso a la aplicación y gestionar eficientemente a los usuarios, se requiere un servicio de autenticación robusto, capaz de integrarse con proveedores externos (por ejemplo, autenticación social como *Google* o *Facebook*) y protocolos de autenticación seguros.

- **Almacenamiento y gestión de datos**: la aplicación necesita bases de datos escalables que permitan almacenar y gestionar grandes volúmenes de información de manera eficiente. Esto incluye la implementación de bases de datos relacionales para las transacciones y para las interacciones en tiempo real y el manejo de datos no estructurados.
- ***API* para comunicaciones**: es fundamental una *API* para que los distintos componentes (*frontend* y *backend*) interactúen y compartan datos de forma eficiente. Esta *API* debe ser capaz de manejar múltiples solicitudes simultáneas y ofrecer una alta disponibilidad para soportar las operaciones en tiempo real de la aplicación.
- ***Deployment* de modelos de inteligencia artificial**: dado que una parte de la funcionalidad se basa en inteligencia artificial (IA), se necesita infraestructura para el *deployment* de modelos. Esto implica servicios especializados en IA que garanticen tanto el procesamiento como el escalado de los modelos de manera continua.
- **Infraestructura interna para la prevención de colisiones**: para evitar conflictos en la manipulación de recursos, es crucial implementar una arquitectura que gestione adecuadamente la concurrencia. Esto incluye el uso de contenedores y tecnologías de virtualización que permitan aislar procesos, mejorar la eficiencia en el uso de recursos y evitar colisiones entre operaciones de los usuarios.

Este enfoque holístico asegura que todos los aspectos de la aplicación se encuentren respaldados por una infraestructura robusta y escalable, que permita tanto su crecimiento a futuro como la mejora continua de la experiencia del usuario.

5.24. Comparación de proveedores de nube

Tras definir los recursos requeridos, se procedió a realizar una evaluación de varios proveedores de servicios en la nube (*AWS, Google Cloud, Azure*) que ofrecieran las funcionalidades y niveles de servicio necesarios para soportar la aplicación. Se tomaron en cuenta factores como facilidad de integración, soporte técnico, nivel de personalización, y costos de servicio. Además, se evaluaron aspectos específicos de cada proveedor, como niveles de seguridad, opciones de administración, y capacidad de escalabilidad horizontal y vertical, con el fin de seleccionar la mejor opción.

Servicio *IAM*, cuentas internas

Iniciando con el servicio para manejo de cuentas internas, que se le puede conocer como IAM, se realizo la siguiente comparación entre los principales proveedores de este servicio

Cuadro 5.4. Comparación de Servicios *IAM* entre *AWS*, *Azure*, *Google Cloud* y *AWS IAM Identity Center*

Característica	***AWS IAM***	***Azure Active Directory (AAD)***	***Google Cloud IAM***	***AWS IAM Identity Center***
Administración de Identidades	Usuarios, grupos y roles para gestionar permisos de acceso a recursos	Usuarios, grupos, roles y dominios gestionados mediante *AAD* y *Microsoft Entra ID*	Usuarios, grupos y funciones, gestionados con permisos a nivel de recursos	Gestión centralizada de usuarios y permisos en múltiples cuentas y aplicaciones de *AWS*
Autenticación Multifactor (MFA)	Integración nativa para *MFA* y soporte para autenticadores de terceros	*MFA* mediante *AAD*, con soporte para autenticadores biométricos y autenticadores externos	*MFA* disponible, principalmente mediante *Google Authenticator* y aplicaciones de terceros	*MFA* integrado con opciones para autenticadores de terceros y compatibilidad con políticas de seguridad avanzadas
Integración de Directorio	Soporte para *Active Directory* y *LDAP*, y federación con otros *IDPs*	Soporte completo para *AD*, además de integración con otros directorios de identidades como *AD FS*	No ofrece integración directa con *AD*, pero permite federación mediante *SAML* y *OpenID*	Integración nativa con *AWS SSO* y sincronización con *AD* y otros *IDPs* mediante *SAML*
Gestión de Accesos Condicionales	Limitada; configuraciones basadas en roles y políticas	Acceso condicional avanzado basado en ubicación, dispositivo y riesgo de usuario	Limitada; control granular de políticas de acceso basadas en roles	Control de acceso avanzado mediante políticas basadas en condiciones y directivas
Control de Acceso Basado en Roles *(RBAC)*	Amplia gama de roles predefinidos y posibilidad de roles personalizados	Amplia gama de roles y políticas *RBAC* con personalización avanzada	Amplio conjunto de roles predefinidos y soporte para políticas detalladas	Soporte para *RBAC* y *ABAC* (basado en atributos) con personalización de permisos para aplicaciones de *AWS*
Escalabilidad	Escalabilidad horizontal y vertical para grandes organizaciones	Muy escalable, con soporte a nivel de empresa para sincronización de identidades	Escalable, aunque limitado en opciones de integración de directorios	Escalabilidad para organizaciones que manejan múltiples cuentas y aplicaciones en *AWS*
Soporte para Aplicaciones Externas	*SAML*, *OpenID Connect*, *OAuth*, con soporte nativo para integraciones de terceros	Soporte completo para aplicaciones externas mediante *OAuth*, *SAML* y *OpenID Connect*	Soporte para *OAuth* y *OpenID Connect*, con opciones limitadas en *SAML*	Integración sencilla con aplicaciones compatibles con *SAML 2.0* y soporte para *OpenID Connect*
Costo	Tarificación por usuario y política de uso de permisos	Tarificación variable; modelo gratuito básico y premium para empresas	Tarificación gratuita para recursos básicos, con costo para características avanzadas	Incluido en *AWS* para la gestión de accesos en cuentas y aplicaciones de *AWS*; costos adicionales para integración avanzada
Mejor Uso	Gestión avanzada de accesos y políticas en múltiples cuentas y regiones	Organizaciones que ya usan *Microsoft* y buscan integración con *Office 365* y *AD*	Usuarios que dependen de *Google Workspace* y aplicaciones en *Google Cloud*	Organizaciones que necesitan una solución centralizada para gestionar accesos en cuentas y aplicaciones de *AWS*

Servicio *CIAM*

En este caso también se analizaron los servicios *CIAM* tentativamente, que corresponden a servicios de autenticación para que el resultado se guarde de manera administrada por la nube.

Cuadro 5.5. Comparación de Servicios *CIAM* entre *AWS, Azure* y *Google Cloud*

Características	***AWS Cognito***	***Azure AD B2C***	***Auth0***
Autenticación de usuarios	Autenticación mediante usuario y contraseña, integración con proveedores de identidad social, *MFA* opcional	Autenticación con usuario/contraseña, *MFA*, integración con múltiples proveedores sociales	Amplia variedad de opciones de autenticación, como sin contraseña, redes sociales, y autenticación biométrica
Gestión de usuarios	*User Pools* para almacenar y gestionar usuarios con atributos personalizados	Almacenamiento de usuarios con opciones de personalización avanzada en el *Tenant* de *Azure AD*	Opciones avanzadas de perfil de usuario con reglas personalizables
Integración con redes sociales	*Facebook, Google, Amazon, Apple* y otros proveedores de identidad	*Facebook, Google, Microsoft, LinkedIn*, y otros a través de *OpenID Connect*	*Facebook, Google, Twitter, LinkedIn*, y otros a través de *OpenID Connect* y *SAML*
Autenticación multifactor	Compatible con *MFA* (*SMS, TOTP*)	Soporte de *MFA* con métodos como *TOTP* y *push notifications*	*MFA* disponible con varios métodos: SMS, e-*mail*, *OTP*, y autenticación biométrica
Escalabilidad	Escalabilidad alta, compatible con otros servicios de *AWS*	Escalable, configuraciones en *Azure AD* para alta disponibilidad	Alta escalabilidad para cualquier volumen de usuarios
Precios	Basado en el número de usuarios activos mensuales (MAU), cargos adicionales para *MFA* y autenticación social	Precios por número de usuarios y autenticaciones, con opciones de pago por uso o licencias mensuales	Basado en usuarios activos mensuales (MAU) y tarifas para *MFA* y *SSO*
Compatibilidad con *SDK*	*SDK* disponible para *iOS, Android, JavaScript* y otros	*SDK* para *iOS, Android* y aplicaciones *web*	*SDK* para *iOS, Android, JavaScript* y otros
Personalización de la experiencia de usuario	Interfaz personalizable, aunque limitada en opciones avanzadas de *UI*	Alta personalización con *HTML* y *CSS* para interfaces de usuario	Personalización avanzada mediante temas, plantillas y diseño adaptable
Gestión de roles y permisos	Manejo básico de grupos y permisos en Cognito; *IAM* de *AWS* para roles avanzados	Control avanzado de permisos y roles mediante grupos y políticas de *Azure AD*	Control granular de roles y permisos mediante reglas personalizables
Recuperación de cuenta	Recuperación de cuenta mediante SMS, correo y preguntas de seguridad	Opciones avanzadas de recuperación de cuenta mediante métodos como SMS y autenticación secundaria	Soporte completo para recuperación de cuenta mediante autenticación multifactor
Disponibilidad global	Disponible en la mayoría de las regiones de *AWS*	Disponibilidad en múltiples regiones de *Azure*	Disponible en múltiples regiones mediante *AWS, Azure* y *Google Cloud*
Idiomas soportados	Soporte multilingüe con localización de interfaz de usuario	Soporte multilingüe con personalización de idiomas adicionales	Soporte multilingüe completo con opciones avanzadas de personalización

Análisis de las soluciones *CIAM*

Las soluciones *CIAM* proporcionadas por las distintas plataformas en la nube tienen características que pueden adaptarse a diferentes tipos de necesidades empresariales. A continuación, se presenta un análisis de cada plataforma basada en la información comparativa de la tabla anterior.

AWS Cognito

AWS Cognito es una solución completa para la autenticación y gestión de usuarios en aplicaciones *web* y móviles, con alta escalabilidad e integración nativa con otros servicios de *AWS*. *Cognito* es especialmente adecuado para empresas ya involucradas en el ecosistema de *AWS*, permitiendo una implementación rápida y eficiente mediante el uso de otros servicios de *AWS*, como Lambda para lógica de negocio, *S3* para almacenamiento y *API Gateway* para la gestión de *API*.

A nivel de personalización, aunque ofrece opciones para interfaces de usuario, estas pueden ser limitadas en comparación con otras soluciones, lo cual puede ser un factor a considerar en aplicaciones que requieren una apariencia de marca más específica. Además, *AWS Cognito* es económico y flexible, con precios basados en el número de usuarios activos mensuales (MAU) y configuraciones de *MFA* opcionales.

Azure AD B2C

Azure AD B2C destaca por su capacidad de personalización y su integración profunda con los servicios de *Microsoft*. Esta solución es particularmente beneficiosa para aplicaciones empresariales que requieren autenticación multifactor avanzada, soporte *B2B* y *B2E*, y opciones de gestión de roles y permisos a nivel granular. La plataforma permite la personalización detallada de la experiencia de usuario con *HTML* y *CSS*, facilitando la alineación con la identidad visual de la empresa.

Además, *Azure AD B2C* es altamente escalable y puede adaptarse a empresas de distintos tamaños. La estructura de precios se basa en el número de usuarios y autenticaciones, lo cual puede ser beneficioso para aplicaciones que manejan volúmenes significativos de autenticación.

Google Identity Platform

Google Identity Platform es ideal para desarrolladores que buscan una integración simple y rápida, especialmente en aplicaciones móviles y *web* que ya utilizan *Firebase* como *backend*. Aunque ofrece funcionalidad básica en comparación con otras opciones, su escalabilidad a través de *Google Cloud* y su facilidad de uso lo hacen adecuado para aplicaciones de alto tráfico que requieren una administración sencilla de usuarios y autenticación social.

Sin embargo, puede tener limitaciones en la personalización de la interfaz de usuario y en la gestión avanzada de roles, por lo que se recomienda principalmente para aplicaciones que no requieren configuraciones de permisos complejas.

Auth0

Auth0 es una de las soluciones más completas y versátiles en el ámbito de *CIAM*. Ofrece una amplia gama de opciones de autenticación, soporte de *MFA*, y una experiencia de usuario altamente personalizable. Su capacidad para integrarse con servicios en *AWS*, *Azure* y *Google Cloud*, junto con su escalabilidad, lo convierte en una opción sólida para empresas que manejan altos volúmenes de usuarios.

Además, *Auth0* permite una gestión avanzada de roles y permisos mediante reglas personalizadas, lo cual es beneficioso para aplicaciones que requieren un control detallado sobre el acceso y los permisos de usuario. No obstante, su modelo de precios puede ser más alto en comparación con otras plataformas, lo que lo hace más adecuado para empresas que pueden invertir en funcionalidades avanzadas de autenticación y seguridad.

La elección de una solución *CIAM* depende en gran medida de las necesidades específicas de la aplicación, el volumen de usuarios, y el presupuesto disponible.

- ***AWS Cognito*** es ideal para quienes ya usan servicios de *AWS* y necesitan una solución económica y escalable.
- ***Azure AD B2C*** es adecuada para empresas que requieren personalización avanzada y compatibilidad con otras aplicaciones empresariales.
- ***Google Identity Platform*** es una opción ligera, ideal para aplicaciones de alto tráfico que necesitan una integración rápida y fácil con *Firebase*.
- ***Auth0*** ofrece una experiencia completa, con opciones avanzadas de autenticación y personalización, siendo óptima para empresas que necesitan una solución *CIAM* robusta y escalable.

Cada plataforma tiene puntos fuertes que la hacen destacar en distintos aspectos, por lo que una evaluación cuidadosa de estos factores permitirá elegir la solución *CIAM* más adecuada para cada proyecto.

Bases de datos *SQL*

El crecimiento de los servicios en la nube ha permitido a las organizaciones acceder a herramientas avanzadas para la gestión de bases de datos, independientemente de su tamaño o industria. Entre las plataformas líderes en este campo se encuentran *Amazon Web Services* (*AWS*), *Microsoft Azure* y *Google Cloud Platform* (*GCP*). Cada una de estas plataformas ofrece una variedad de servicios *SQL* que facilitan la administración de datos de manera escalable y segura.

La comparación de los servicios *SQL* de estas plataformas es fundamental debido a las diferencias en funcionalidades, costos, escalabilidad y opciones de migración que ofrecen. Las decisiones sobre qué plataforma utilizar pueden depender de la especificidad de cada aplicación, del presupuesto disponible, y de los requisitos de alta disponibilidad y seguridad que impone cada organización. Al comparar estas características clave, las empresas pueden seleccionar la opción que mejor se adapte a sus necesidades actuales y a su estrategia a largo plazo.

Cuadro 5.6. Comparación de Servicios *SQL* entre *AWS, Azure* y *Google Cloud*

Característica	***AWS* (*Amazon Web Services*)**	***Microsoft Azure***	***Google Cloud Platform***
Servicios Principales	*Amazon RDS, Amazon Aurora, Amazon Redshift*	*Azure SQL Database, Azure Database for MySQL, Azure Database for PostgreSQL*	*Cloud SQL, BigQuery, Spanner*
Tipos de Bases de Datos Soportadas	*MySQL, PostgreSQL, MariaDB, Oracle, SQL Server, Amazon Aurora*	*SQL Server, MySQL, PostgreSQL, MariaDB, Cosmos DB*	*MySQL, PostgreSQL, SQL Server, Spanner*
Escalabilidad	Escalado automático con *Amazon Aurora*; soporte para instancias de hasta 32 *vCPUs* y 244 GiB de *RAM*	Escalado elástico con *Azure SQL*; instancias hasta 80 *vCPUs* y 400 GB de *RAM*	Escalado horizontal con *Cloud Spanner* y vertical con *Cloud SQL*; instancias hasta 96 *vCPUs* y 624 GB de *RAM*
Alta Disponibilidad	*Multi-AZ* en *Amazon RDS, Amazon Aurora* soporta recuperación automática	*Azure SQL* con alta disponibilidad integrada y soporte de geo-replicación	*Cloud SQL* y *Cloud Spanner* soportan zonas múltiples y replicación global
Precios	Paga solo por lo que usas, opciones de precios *on-demand* y reservados	Basado en *DTU* (*Database Transaction Units*) o *vCores*; opciones de precios escalonados	Paga por uso y almacenamiento, descuentos por uso continuo en servicios específicos
Seguridad y Cumplimiento	Integración con *AWS IAM*, soporte de cifrado en reposo y en tránsito, certificado *PCI DSS*	Autenticación *Azure AD*, cifrado *TDE*, cumplimiento con *HIPAA, ISO, PCI*	Cifrado en reposo y en tránsito, *IAM* detallado, cumplimiento con *GDPR, ISO, PCI*
***Backups* y Recuperación**	*Backups* automáticos y *snapshots*; restauración en tiempo específico	*Backup* automático y georedundante; restauración de punto en el tiempo	*Backups* automáticos y restauración de punto en el tiempo hasta por 7 días en *Cloud SQL*
Migración de Datos	*AWS Database Migration Service* para migración heterogénea y homogénea	*Azure Database Migration Service* para migración desde *SQL Server* y otros motores	*Database Migration Service* de *Google* para migración de *MySQL, PostgreSQL* y *SQL Server*
Funciones Avanzadas	*Amazon Aurora Machine Learning*, soporte para transacciones ACID en *Aurora* y *RDS*	Integración con *Power BI, Elastic Pool* para optimización de recursos	*BigQuery* para análisis en tiempo real, *Spanner* para transacciones globales

En resumen, cada una de las plataformas (*AWS, Azure* y *Google Cloud*) proporciona servicios *SQL* robustos y avanzados que responden a distintas necesidades de escalabilidad, disponibilidad y seguridad. *AWS* destaca por su flexibilidad en la migración y la integración con servicios de aprendizaje automático en *Amazon Aurora*, lo cual es ideal para aplicaciones que requieren análisis avanzados. Por su parte, *Microsoft Azure* se enfoca en la integración nativa con herramientas de análisis como *Power BI*, y su enfoque en la seguridad con autenticación *Azure AD* puede ser especialmente atractivo para empresas con necesidades de cumplimiento normativo. *Google Cloud*, por otro lado, ofrece una excelente capacidad de escalado horizontal con *Spanner* y opciones avanzadas de análisis en *BigQuery*, convirtiéndose en una plataforma sólida para aplicaciones de datos a gran escala.

La selección de la plataforma adecuada dependerá de las necesidades específicas de cada empresa y de su estrategia de crecimiento en la nube. Consideraciones como el tipo de base de datos requerida, el presupuesto disponible y la necesidad de integración con otros servicios en la nube deben guiar la elección para maximizar el valor y la eficiencia.

Interfaz de programación de aplicaciones en la nube

En la era digital, las *API* (*Application Programming Interfaces*) se han convertido en un componente clave para la interoperabilidad de servicios, tanto dentro de organizaciones como en aplicaciones de terceros. Con el crecimiento de la nube, la creación y gestión de *API* ha evolucionado hacia servicios avanzados y altamente configurables, permitiendo a los desarrolladores crear, monitorear, asegurar y optimizar *API* de manera más sencilla y escalable. A continuación, se realiza una comparación de los servicios de *API* que ofrecen las principales plataformas en la nube: *AWS, Azure* y *Google Cloud Platform*.

Cuadro 5.7. Comparación de servicios de *API* entre *AWS*, *Azure* y *Google Cloud Platform*

Característica	***AWS (Amazon Web Services)***	***Microsoft Azure***	***Google Cloud Platform***
Servicios principales para *API*	*Amazon API Gateway, AWS AppSync, AWS Lambda, EC2* para *hosting* de *API* personalizadas	*Azure API Management, Azure Functions, Logic Apps, Azure VM* para *API* personalizadas	*Google Cloud Endpoints, Apigee, Cloud Functions, Compute Engine* para *hosting* de *API* personalizadas
Tipos de *API* soportadas	*REST, WebSocket, HTTP, GraphQL (con AppSync), API* personalizadas con *EC2*	*REST, SOAP, WebSocket, HTTP, GraphQL* (a través de *API Management*), *API* personalizadas con *Azure VM*	*REST, HTTP, gRPC, GraphQL* (con *Apigee*), *API* personalizadas con *Compute Engine*
Opciones de implementación de *API*	*Amazon API Gateway* para *API* gestionadas; *AppSync* para *GraphQL*; *EC2* para *API* personalizadas, soportando cualquier *framework* o lenguaje compatible	*Azure API Management* para centralizar *API*; *Azure Functions* para *serverless*; *Azure VM* para *API* personalizadas y control directo del entorno	*Cloud Endpoints* para *API* gestionadas; *Apigee* para *API* empresariales y análisis; *Compute Engine* para *API* personalizadas, permitiendo cualquier *stack* o *framework*
Escalabilidad	Escalado automático en *API Gateway* y *AppSync*; capacidad de escalado para instancias *EC2* según demanda	Escalado elástico con *API Management*; escalabilidad en *Azure VM* y *Azure Functions* para ajustar recursos según uso	Escalado automático en *Cloud Endpoints* y *Apigee*; *Compute Engine* permite escalado personalizado para *API* dedicadas
Seguridad y autenticación	Autenticación con *IAM, API Keys*, integración con *AWS Cognito* para *OAuth* 2.0 y *OpenID Connect*; configuraciones avanzadas en *EC2* con autenticación personalizada	Autenticación con *Azure AD, API Keys*, soporte para *OAuth* 2.0 y *OpenID Connect*; configuraciones personalizables en *Azure VM* para autenticación personalizada	Autenticación con *IAM, API Keys*, integración con *Identity Platform* para *OAuth* 2.0 y soporte de autenticación detallada en *Compute Engine*
Monitoreo y análisis	*Amazon CloudWatch* para monitoreo de *API Gateway*, métricas avanzadas y registros de actividad en *AppSync* y *CloudWatch* para *EC2*	*Azure Monitor* y *Application Insights* para monitoreo y análisis en tiempo real en *API Management*; insights detallados en *Azure VM*	*Cloud Monitoring* y *Stackdriver* para monitoreo en *Cloud Endpoints* y Apigee; métricas personalizadas en *Compute Engine* para *API* específicas
Gestión de versiones	Versiones múltiples en *Amazon API Gateway*; control manual en *API* hospedadas en *EC2*	Gestión de versiones en *API Management* con control detallado de revisiones; versiones manuales en *API* de *Azure VM*	Control de versiones en *Cloud Endpoints* y *Apigee*; versiones personalizadas en *Compute Engine* para *API* manuales
Optimización y rendimiento	Cacheo en *Amazon API Gateway*, optimización de tráfico y latencia con *CloudFront*; configuración avanzada de balanceo en *EC2* para *API* de alto rendimiento	Cacheo integrado en *Azure API Management*; *Azure Front Door* para optimización de tráfico; balanceo de carga configurable en *Azure VM*	Cacheo en *Apigee*; balanceo de carga global en *Cloud Endpoints*; configuraciones avanzadas de tráfico en *Compute Engine*
Precios	Pago por uso en *API Gateway* y *AppSync*; costos adicionales para *EC2* según consumo y reserva de instancias; precios escalonados en Lambda	Basado en consumo en *API Management*, precios en *Functions* y *Logic Apps* según llamadas, y precios de *VM* según configuración y uso	Pago por uso en *Cloud Endpoints* y *Apigee*, costos por instancia en *Compute Engine*, opciones de precios escalonados para *Cloud Functions*
Migración de datos y *API*	*AWS Database Migration Service* facilita la migración de datos y *API*; configuraciones avanzadas en *EC2* para *API* que requieren mayor control	*Azure Database Migration Service* para migración desde otras plataformas; *Azure Migrate* para configuraciones avanzadas de *API* y *VM*s	*Database Migration Service de Google* para *MySQL, PostgreSQL* y *SQL Server*; opciones en Compute *Engine* para migraciones avanzadas de *API*
Funciones Avanzadas	*Amazon AppSync* para *GraphQL* y sincronización en tiempo real; *Machine Learning* en *Amazon SageMaker* y *Lambda*; personalización avanzada en *EC2*	*Azure API Management* con integración a *Power BI* y soporte de analítica avanzada; *Azure Cognitive Services* en *API* personalizadas en *VM*	*Apigee* para análisis en tiempo real; funciones avanzadas de inteligencia en *BigQuery* y personalización en *Compute Engine*

Cada plataforma en la nube ofrece una amplia variedad de servicios para la creación y gestión

de *API*, proporcionando opciones que van desde simples *API REST* hasta opciones avanzadas de *GraphQL* y *WebSocket*. *AWS* destaca por su integración con Lambda y sus opciones de escalado automático, mientras que *Azure* ofrece soluciones de integración robustas con *Azure Functions* y una gestión eficiente de versiones. Por otro lado, *Google Cloud Platform* proporciona una potente combinación con *Apigee* y *Cloud Endpoints*, ideal para servicios de alto rendimiento y una latencia mínima. La elección de la plataforma ideal dependerá de las necesidades específicas de cada proyecto y del ecosistema de herramientas que mejor se integre con la infraestructura existente.

Deployment **de modelos de inteligencia artificial**

La inteligencia artificial (IA) ha evolucionado rápidamente en los últimos años, transformando múltiples industrias y generando nuevas oportunidades para optimizar procesos, mejorar la experiencia del usuario y resolver problemas complejos. Sin embargo, el desarrollo y despliegue de IA puede ser intensivo en términos de recursos y demanda una infraestructura robusta y escalable. En este contexto, las plataformas de nube, como *AWS, Azure* y *Google Cloud Platform*, han facilitado significativamente el acceso a herramientas avanzadas de IA, permitiendo que empresas de todos los tamaños puedan adoptar y escalar sus modelos de *machine learning* y *deep learning* sin necesidad de una infraestructura física costosa.

Cada uno de estos proveedores de nube ofrece una variedad de servicios específicos de IA, que abarcan desde herramientas para la preparación y almacenamiento de datos hasta plataformas para la creación y despliegue de modelos de aprendizaje automático. *AWS* destaca con servicios como *Amazon SageMaker* y *Amazon Rekognition*, permitiendo a los desarrolladores crear modelos de aprendizaje supervisado, no supervisado y profundo, y ofrece soluciones específicas de procesamiento de imágenes y videos, esenciales en aplicaciones de visión por computadora. *Azure*, por su parte, combina sus servicios de IA con su integración en el ecosistema empresarial de *Microsoft*, ofreciendo capacidades avanzadas en *Azure Machine Learning*, *Cognitive Services* y *Azure Bot Service* para IA conversacional, lo cual facilita una adopción sinérgica para organizaciones que ya utilizan el ecosistema de *Microsoft*. *Google Cloud Platform* también se destaca en el campo de la IA, con una fuerte apuesta por el aprendizaje automático y el procesamiento de datos masivos a través de servicios como *TensorFlow Enterprise, Vertex AI*, y *AutoML*, siendo particularmente eficiente para tareas que requieren procesamiento de *big data* y análisis en tiempo real.

A través de estas plataformas, los usuarios pueden acceder a funciones avanzadas de procesamiento de datos, modelos preentrenados para reconocimiento de voz, análisis de imágenes y procesamiento de lenguaje natural, así como capacidades para el despliegue de modelos en tiempo real. Los servicios de IA en la nube también permiten la creación de *pipelines* de *machine learning* en los que los datos pueden ser almacenados, procesados y analizados en un solo entorno, reduciendo significativamente el tiempo de desarrollo y facilitando la colaboración entre equipos de ciencia de datos, desarrollo y operaciones.

Cuadro 5.8. Comparación de servicios de inteligencia artificial entre *AWS, Azure* y *Google Cloud Platform*

Característica	***AWS* (*Amazon Web Services*)**	***Microsoft Azure***	***Google Cloud Platform***
Servicios principales de IA	*Amazon SageMaker, AWS Deep Learning AMIs, AWS Lambda* para inferencia en tiempo real, *Amazon Rekognition* para visión por computadora	*Azure Machine Learning, Cognitive Services, Azure Synapse Analytics, Azure Bot Service* para IA conversacional	*Google AI Platform, TensorFlow Enterprise, AutoML, Vertex AI, Vision AI* para procesamiento de imágenes
Procesamiento y almacenamiento de datos	*Amazon Redshift, Amazon S3* para almacenamiento de *big data*; *Glue* para *ETL* y preparación de datos para *machine learning*	*Azure Data Lake, Azure Synapse Analytics* para análisis y almacenamiento masivo; *Azure Databricks* para procesamiento de datos en tiempo real	*BigQuery* para análisis masivo de datos, *Cloud Storage* para almacenamiento escalable; *Dataflow* para procesamiento y *ETL* en tiempo real
***Frameworks* soportados**	*TensorFlow, PyTorch, MXNet, ONNX, Apache Spark*; soporte para múltiples *frameworks* en *SageMaker* y *AMIs* personalizables	*TensorFlow, PyTorch, Scikit-Learn, Keras*; soporte completo en *Azure Machine Learning* y *Azure Databricks*	*TensorFlow, PyTorch, Scikit-Learn, Keras, XGBoost*; optimización en AI *Platform* y *Vertex AI* para *TensorFlow*
Capacidades de *AutoML*	*SageMaker Autopilot* para generar modelos automáticamente, soportando análisis y recomendaciones de modelos optimizados	*Azure Machine Learning AutoML* para entrenar modelos con selección automática de algoritmos y optimización de hiperparámetros	*Google AutoML* para modelos personalizados de visión, lenguaje natural y tablas, optimizados automáticamente con *Vertex AI*
Implementación de modelos	*SageMaker* para despliegue de modelos en contenedores, *AWS Lambda* para inferencia sin servidor, *Elastic Inference* para escalado de *GPUs*	*Azure Kubernetes Service* (*AKS*) para contenedores, *Azure Functions* para inferencia sin servidor, opciones en *VM* para *GPUs* dedicadas	*Google Kubernetes Engine* (*GKE*) para contenedores; despliegue con *Vertex AI* y soporte para inferencia en *Cloud Functions*
Herramientas de IA conversacional	*Amazon Lex* para *chatbots* y asistentes virtuales; *Polly* para conversión de texto a voz; Transcribe para reconocimiento de voz	*Azure Bot Service* para *bots* personalizados, integración con *Azure Cognitive Services* para lenguaje y voz	*Dialogflow* para creación de *chatbots* y asistentes, integración con *Text-to-Speech* y *Speech-to-Text* en *Google AI*
Análisis de imágenes y videos	*Amazon Rekognition* para reconocimiento de imágenes y videos; integración con *S3* para almacenamiento masivo de archivos multimedia	*Computer Vision* en *Azure Cognitive Services* para análisis de imágenes; Video *Indexer* para procesamiento de contenido multimedia	*Vision AI* y Video *Intelligence* para reconocimiento y clasificación de imágenes y videos; integración con *Cloud Storage* para datos multimedia
Soporte para *machine learning* en tiempo real	*Amazon Kinesis* para análisis en tiempo real; integración con *SageMaker* y *Lambda* para inferencia inmediata en *streaming*	*Azure Stream Analytics* para procesamiento de datos en tiempo real; integración con *Azure Machine Learning* para inferencia instantánea	*Cloud Dataflow* para flujos de datos en tiempo real; soporte para inferencia en *Vertex AI* y *Cloud Functions* en aplicaciones de *streaming*
Seguridad y cumplimiento	*IAM* para control de acceso, cifrado de datos en reposo y en tránsito, cumplimiento con *HIPAA, GDPR, ISO* y *SOC*	*Azure AD* para gestión de identidades, cifrado y soporte de cumplimiento con *HIPAA, GDPR, ISO, SOC* y más	*IAM* para autenticación y autorización, cifrado en reposo y en tránsito, cumplimiento con *HIPAA, GDPR, PCI, ISO* y *SOC*
Optimización y rendimiento	*AWS Inferentia* para optimización de inferencia de IA; *Elastic Inference* para mejorar rendimiento en *GPUs* compartidas	*Azure Machine Learning Compute* para rendimiento optimizado en *VM*s y soporte de *GPU*; *AKS* para escalado de recursos	TPUs para rendimiento en modelos de *TensorFlow*; *Vertex AI* para optimización en tareas de inferencia y entrenamiento
Precios y modelos de pago	Pago por uso en *SageMaker*, precios escalonados para instancias *EC2*; descuentos por uso reservado y volumen en *Amazon Rekognition*	Pago por uso en *Azure Machine Learning*; precios escalonados en *Cognitive Services* y opciones de ahorro en *VM*s	Pago por uso en *Vertex AI* y *AutoML*; precios basados en almacenamiento en *BigQuery* y opciones de descuento por uso prolongado

La elección de una plataforma de nube para implementar IA dependerá en gran medida de los requerimientos específicos del proyecto, el nivel de personalización deseado y el ecosistema tecnológico en el que ya opera una organización. *AWS*, con su infraestructura robusta y su amplia gama

de servicios de IA como *Amazon SageMaker* y *Recognition*, es ideal para proyectos que requieren escalabilidad y una integración sencilla con otros servicios en la nube de *AWS*. Su compatibilidad con múltiples *frameworks* de *machine learning* y su enfoque en la optimización del rendimiento a través de *AWS* Inferentia y *Elastic Inference* lo convierten en una opción sólida para proyectos de IA que necesitan alto rendimiento.

Azure, con una oferta centrada en la interoperabilidad con el ecosistema de *Microsoft* y una gran variedad de servicios de IA como *Azure Machine Learning* y *Cognitive Services*, es una excelente elección para empresas que ya operan en entornos de *Microsoft*. Su integración con herramientas empresariales como *Power BI* y *Azure Synapse Analytics* facilita el análisis y la visualización de datos en tiempo real, lo que resulta útil en sectores donde la toma de decisiones informada es esencial. Además, su soporte para diversos *frameworks* y su capacidad de *AutoML* permiten que los desarrolladores creen y desplieguen modelos de manera rápida y eficiente.

Por otro lado, *Google Cloud Platform* ha desarrollado una infraestructura avanzada que combina servicios de IA altamente optimizados, como *Vertex AI* y *AutoML*, con su experiencia en *big data* y análisis en tiempo real a través de *BigQuery* y *Dataflow*. Esto la convierte en la opción ideal para organizaciones que buscan soluciones de análisis de datos masivos y requieren procesamiento de IA en tiempo real. Además, la presencia de herramientas como *TPUs*, diseñadas específicamente para *TensorFlow*, le otorgan una ventaja en aplicaciones que requieren entrenamiento intensivo en recursos de modelos de *deep learning*.

En resumen, *AWS*, *Azure* y *Google Cloud Platform* ofrecen una gama completa de soluciones de IA que pueden adaptarse a las necesidades de diferentes proyectos y organizaciones. La elección de la plataforma adecuada dependerá de factores como el nivel de personalización requerido, el entorno tecnológico actual de la organización y los recursos necesarios para entrenamiento y despliegue de modelos. A medida que la tecnología de IA continúa evolucionando, es probable que estas plataformas sigan innovando, ofreciendo nuevas capacidades y mejoras de rendimiento para facilitar el desarrollo de aplicaciones de IA cada vez más avanzadas y accesibles.

5.24.1. *Templates*, paquetes y *all in one*

El despliegue de aplicaciones en la nube ha evolucionado significativamente, permitiendo que las empresas no solo se enfoquen en el desarrollo, sino también en la administración y optimización de sus aplicaciones sin preocuparse por la infraestructura subyacente. Los principales proveedores de nube, como *AWS, Azure* y *Google Cloud Platform*, ofrecen soluciones integrales de despliegue de aplicaciones que facilitan la creación, el monitoreo y el escalado automático. Estas soluciones, como *Elastic Beanstalk* en *AWS*, *App Service* en *Azure* y *App Engine* en *Google Cloud*, proporcionan un entorno completo para administrar el ciclo de vida de una aplicación, incluyendo desde configuraciones básicas hasta optimizaciones avanzadas de rendimiento y escalabilidad.

Cada una de estas plataformas ofrece soporte para una amplia variedad de lenguajes de programación y *frameworks*, como *Python, Java, .NET* y *Node.js*, y permite a los desarrolladores implementar aplicaciones sin preocuparse por la administración de servidores, el balanceo de carga o la gestión de recursos de red. *Elastic Beanstalk* en *AWS*, por ejemplo, permite el despliegue rápido de aplicaciones con escalado automático y administración simplificada, integrándose con otros servicios de *AWS*, como *Amazon CloudWatch*, para el monitoreo y análisis de aplicaciones. De manera similar, *Azure App Service* en *Microsoft Azure* proporciona un entorno altamente configurable que soporta aplicaciones en múltiples lenguajes y permite integrarse con otros servicios, como *Azure DevOps*, para el despliegue continuo. Por su parte, *Google App Engine* facilita la creación y despliegue de aplicaciones escalables en un entorno sin servidor, permitiendo a los desarrolladores enfocarse en el código y las funcionalidades de la aplicación en lugar de en la infraestructura.

Estas soluciones también ofrecen compatibilidad con el despliegue de aplicaciones basadas en

contenedores, ya sea mediante *Docker* o *Kubernetes*, lo cual permite escalar y administrar aplicaciones complejas de manera eficiente. La integración con otros servicios de *CI/CD* en cada plataforma, como *CodePipeline* en *AWS*, *Azure DevOps* en *Azure* y *Cloud Build* en *Google Cloud*, permite automatizar el flujo de trabajo y actualizar aplicaciones de manera continua. Al mismo tiempo, cada una de estas plataformas garantiza un monitoreo y registro de actividades detallado, con capacidades de análisis en tiempo real para detectar problemas de rendimiento y responder a eventos sin interrumpir el servicio. En esta tabla, se comparan estas y otras características clave de los servicios integrales de despliegue de aplicaciones en *AWS, Azure* y *Google Cloud Platform*, brindando una visión general de sus ventajas y capacidades únicas.

Cuadro 5.9. Comparación de Soluciones Integrales de Despliegue de Aplicaciones entre *AWS, Azure* y *Google Cloud Platform*

Característica	**AWS (*Amazon Web Services*)**	***Microsoft Azure***	***Google Cloud Platform***
Servicio Integral de Despliegue	*Elastic beanstalk* para despliegue de aplicaciones sin administración de infraestructura	*Azure app service* para despliegue de aplicaciones con administración automática de recursos	*App engine* para despliegue de aplicaciones sin servidor y escalado automático
Compatibilidad con lenguajes	Soporte para *Python, Java, .NET, PHP, Node.js, Ruby* y *Go*; despliegue mediante contenedores y Docker	Soporte para *.NET, Java, Ruby, Node.js, PHP, Python* y contenedores; integración con *Visual Studio*	Soporte para *Python, Java, Node.js, PHP, Go, Ruby y .NET*; despliegue mediante contenedores y *Docker*
Configuración y administración	Automatización de configuración, monitoreo y escalado en *Elastic Beanstalk*; integración con *CloudWatch* para métricas y registros	Gestión integrada de configuración, escalado y monitoreo en *App Service*; integración con *Application Insights* y *Azure Monitor*	Escalado y monitoreo automáticos en *App Engine*; administración de configuración integrada con *Cloud Monitoring*
Despliegue de aplicaciones en contenedores	Compatibilidad con *Docker* y *Kubernetes* a través de EKS, además de despliegues en *Beanstalk* con *Docker*	Compatibilidad con contenedores en *App Service*; integración avanzada con *AKS* (*Azure Kubernetes Service*)	Compatibilidad con *Docker* y *Kubernetes* a través de *GKE* (*Google Kubernetes Engine*); *App Engine*
Escalabilidad	Escalado automático en *Elastic Beanstalk*; personalización avanzada de instancias *EC2*	Escalado automático en *App Service* con opciones para balanceo de carga y control granular de recursos	Escalado automático y balanceo de carga en *App Engine*; configuraciones avanzadas en *GKE* para cargas de trabajo complejas
Integración con *CI/CD*	Integración con *AWS CodePipeline* y *CodeBuild* para flujos de *CI/CD*; soporte para repositorios externos	*Azure DevOps* y *GitHub Actions* para *CI/CD* con integración completa; opciones de automatización en *pipelines*	*Cloud Build* y *Cloud Source Repositories* para *CI/CD*; soporte para *GitHub* y otros sistemas de control de versiones
Monitoreo y registro	*Amazon CloudWatch* para monitoreo y alertas, registro detallado en *CloudTrail*	*Azure Monitor* y *Application Insights* para monitoreo y análisis detallado de aplicaciones	*Cloud Monitoring* y *Stackdriver* para métricas y registros de actividad en tiempo real
Precios y modelos de pago	Pago por uso en *Elastic Beanstalk* según consumo de *EC2* y otros recursos; opciones de precios reservados	Pago por uso en *App Service*, con escalado de precios según consumo de recursos; opciones de ahorro en instancias dedicadas	Pago por uso en *App Engine*; precios variables según el uso de *GKE* y recursos de procesamiento y almacenamiento
Soporte para aplicaciones sin servidor	Integración con *AWS Lambda* para aplicaciones sin servidor y eventos automáticos en i	Soporte para funciones sin servidor en *Azure Functions*; integración con *App Service* para eventos en tiempo real	Compatibilidad con *Cloud Functions* para aplicaciones sin servidor; integración con App Engine para eventos sin servidor
Soporte para microservicios	Despliegue de microservicios en *Elastic Beanstalk* y *Amazon ECS*; compatibilidad con arquitecturas basadas en eventos con Lambda	Soporte para microservicios en *App Service* y *AKS*; *Azure Functions* para arquitecturas basadas en eventos	Implementación de microservicios en *App Engine* y *GKE*; integración con *Pub/Sub* para arquitecturas de eventos
Personalización de recursos	Permite personalización avanzada de configuraciones en *EC2* dentro de *Beanstalk*	Control detallado de recursos en *App Service*; personalización en *Azure VM* si se requiere mayor control	Personalización avanzada de configuraciones en *App Engine*; opciones detalladas en *Compute Engine* y *GKE*

Las soluciones integrales de despliegue en la nube permiten a las empresas acelerar el desarrollo de aplicaciones y optimizar su administración sin incurrir en los costos y complejidades de manejar infraestructura física. *AWS Elastic Beanstalk*, *Azure App Service* y *Google App Engine* ofrecen un conjunto de herramientas y funcionalidades que hacen posible desplegar, monitorear y escalar aplicaciones con facilidad. Cada uno de estos servicios ha sido diseñado para integrarse con el ecosistema de su respectiva plataforma de nube, brindando una experiencia optimizada y adaptada a las necesidades del usuario. *Elastic Beanstalk* se destaca por su flexibilidad en la personalización de recursos a través de instancias *EC2*, proporcionando opciones avanzadas para quienes requieren mayor control sobre su entorno. *Azure App Service*, con su enfoque en la interoperabilidad con herramientas empresariales como *Azure DevOps*, es ideal para empresas que buscan una integración fluida en entornos corporativos y desean una gestión eficiente de aplicaciones mediante un ecosistema completo de herramientas de desarrollo y despliegue.

Por otro lado, *Google App Engine*, con su enfoque en un entorno sin servidor y la capacidad de escalar automáticamente en función de la demanda, resulta una opción atractiva para desarrolladores que desean enfocarse en la funcionalidad y el rendimiento de sus aplicaciones sin preocuparse por la infraestructura. *Google Cloud Platform* también ofrece ventajas para aplicaciones que requieren procesamiento intensivo de datos, gracias a su integración con servicios de *big* data como *BigQuery*.

En general, la elección de una plataforma de despliegue dependerá de las necesidades y prioridades específicas de cada proyecto. *AWS* es una excelente opción para quienes buscan control y flexibilidad en la infraestructura; *Azure*, para aquellos que requieren una integración sólida con herramientas empresariales y un entorno corporativo completo; y *Google Cloud Platform*, para desarrolladores que buscan un enfoque sin servidor y escalabilidad sencilla. Estas soluciones de despliegue continuarán evolucionando a medida que la tecnología avance, y es probable que sigan ofreciendo nuevas capacidades y optimizaciones para facilitar el desarrollo de aplicaciones innovadoras, escalables y de alto rendimiento en la nube.

5.25. Implementación y configuración de recursos

Luego de una evaluación exhaustiva de los servicios y capacidades ofrecidas por cada proveedor de nube, se tomó la decisión de descartar *Google Cloud Platform*, a pesar de sus potentes capacidades en procesamiento de *big data*, escalado sin servidor y herramientas como *BigQuery* y *Cloud AI*. Esta decisión estuvo motivada principalmente por la necesidad de contar con una integración más directa y soporte robusto para tecnologías específicas, en especial para el desarrollo con *React Native*. Además, en el análisis comparativo, *Google Cloud* mostró limitaciones en cuanto a herramientas de autenticación optimizadas para aplicaciones móviles multiplataforma, lo que hubiera requerido dependencias y configuraciones adicionales que no se alineaban con los objetivos de simplicidad y control de la arquitectura deseados, sin embargo, partes de sus servicios se terminaron utilizando. En cambio, *AWS* y *Microsoft Azure* demostraron tener un ecosistema mejor integrado y mayor soporte para herramientas empresariales y de autenticación necesarias en el proyecto.

Con el objetivo de establecer un entorno seguro y bien gestionado, el primer paso fue la configuración de *Microsoft* Entra ID y luego *Azure AD B2C*, que, aunque después fue descartado por limitaciones en su integración con *React Native*, formó parte de la fase inicial de pruebas. A continuación, se detallan estos procesos de configuración.

Configuración de *Microsoft* Entra *ID*

Para la administración de usuarios y la implementación de políticas de seguridad robustas, se configuró *Microsoft Entra ID* con una licencia *Premium* que ofrece herramientas avanzadas en la gestión de identidades. La configuración de *Microsoft* Entra ID consistió en una serie de pasos

detallados para garantizar una administración estructurada y segura de los recursos de autenticación:

- **Creación y configuración del inquilino (*Tenant*):** se inició el proceso creando un inquilino de *Microsoft Entra ID*. Este *tenant* actúa como una instancia única dentro de *Azure AD,* proporcionando un espacio de administración de identidades y accesos exclusivo para la aplicación. La configuración del *tenant* incluyó la personalización de los dominios asociados y la activación de políticas de uso para establecer los términos y condiciones que deben cumplir todos los usuarios.
- **Organización de usuarios y grupos:** para mantener un orden en la administración de permisos, se crearon grupos de usuarios basados en roles, incluyendo roles específicos para desarrolladores, administradores y usuarios finales de prueba. Estos grupos permiten asignar permisos de forma masiva y estructurada, facilitando el control sobre los niveles de acceso. Cada grupo fue configurado con parámetros específicos para restringir o permitir acceso a recursos según la necesidad.
- **Sincronización de usuarios con *Azure AD Connect*:** la sincronización de usuarios fue configurada mediante *Azure AD Connect*, una herramienta que permite sincronizar el *Active Directory* local con *Microsoft Entra ID* en la nube. Esta configuración se realizó para garantizar que las credenciales de los usuarios internos se mantuvieran actualizadas y consistentes tanto en los sistemas locales como en la nube, asegurando una experiencia fluida en la autenticación. *Azure AD Connect* también permitió configurar el inicio de sesión único (*SSO*), facilitando el acceso a aplicaciones mediante una sola autenticación.
- **Asignación de roles y control de permisos:** para asegurar un control de acceso detallado, se asignaron roles específicos a cada grupo de usuarios. *Microsoft Entra ID Premium* permite asignar roles integrados como el de Administrador de Aplicaciones, Administrador de Seguridad y Lector, cada uno con diferentes niveles de permisos. Esta segmentación de roles permitió limitar el acceso a recursos sensibles y proteger la infraestructura en desarrollo. Además, se crearon políticas de acceso condicional, que restringen el acceso a la aplicación dependiendo de la ubicación geográfica del usuario o el tipo de dispositivo utilizado.
- **Activación de autenticación multifactor (*MFA*):** como medida adicional de seguridad, se configuró la autenticación multifactor (*MFA*) para los usuarios con roles administrativos y de desarrollo. Esto incluyó la habilitación de *MFA* en dispositivos móviles y correos electrónicos, garantizando que cualquier acceso no autorizado sea detectado y bloqueado antes de comprometer la seguridad de la aplicación.

Implementación y evaluación de *Azure AD B2C*

Como parte de los requerimientos iniciales, se intentó implementar *Azure Active Directory B2C* para la autenticación y gestión de usuarios externos en la aplicación. *Azure AD B2C* fue elegido en esta fase de pruebas debido a su capacidad de manejar autenticación de usuarios externos y ofrecer opciones avanzadas de personalización de la experiencia de usuario en el flujo de inicio de sesión. Además, *Azure AD B2C* soporta protocolos estándar como *OAuth* 2.0 y *OpenID Connect*, los cuales son esenciales para garantizar una autenticación segura y compatible en múltiples plataformas. El proceso de configuración de *Azure AD B2C* consistió en los siguientes pasos:

- **Creación del inquilino de *Azure AD B2C*:** se configuró un *tenant* específico para *B2C*, separado del *tenant* principal de *Microsoft Entra ID*, con el fin de gestionar usuarios externos sin interferir con la administración de usuarios internos de la organización. Este inquilino *B2C* permitió establecer un entorno dedicado exclusivamente a la autenticación de usuarios de la aplicación, facilitando la segmentación de permisos y políticas.

- **Definición de flujos de usuario personalizados:** *Azure AD B2C* permite definir flujos de usuario que controlan la experiencia en el registro, inicio de sesión y recuperación de contraseñas. En este caso, se configuraron flujos de registro e inicio de sesión personalizados, donde se incluyeron validaciones de seguridad adicionales y pasos de verificación, como el envío de códigos de autenticación a correos electrónicos. Estos flujos se personalizaron con logotipos y colores de la marca de la aplicación para brindar una experiencia de usuario consistente.
- **Integración con proveedores de identidad externos:** se configuraron proveedores de identidad externos como *Google* y *Facebook* para ofrecer opciones de inicio de sesión social en la aplicación. Esto implicó registrar la aplicación en las consolas de desarrollador de *Google* y *Facebook* y obtener los identificadores de cliente y secretos necesarios para la integración. *Azure AD B2C* soporta estos proveedores mediante configuraciones estándar, lo que facilita la autenticación sin necesidad de desarrollar integraciones complejas.
- **Evaluación de la compatibilidad con *React Native*:** durante la fase de pruebas, se encontró que la integración de *Azure AD B2C* con *React Native* era limitada, especialmente debido a la falta de librerías oficiales compatibles con el *framework*. Aunque existen librerías de terceros para integrar *Azure B2C* en *React Native*, estas no están oficialmente mantenidas por *Microsoft*, lo cual representa un riesgo de seguridad y de compatibilidad a largo plazo. La dependencia de librerías externas no oficiales podría causar problemas en futuras actualizaciones y requiere mayor esfuerzo de mantenimiento, por lo que se decidió no utilizar *Azure AD B2C* para esta implementación.
- **Descartes y alternativas propuestas:** ante la falta de soporte oficial para *React Native* y la necesidad de garantizar una autenticación segura y confiable, se descartó *Azure AD B2C* para la autenticación de usuarios externos en la aplicación. En su lugar, se exploraron alternativas que ofrecen integración directa con *React Native* y cuentan con librerías oficiales y mantenidas, lo que simplifica el desarrollo y reduce los riesgos asociados a la seguridad y compatibilidad de la aplicación.

La decisión de no implementar *Azure AD B2C* en la aplicación fue tomada para asegurar la estabilidad y seguridad del sistema. Aunque *Azure AD B2C* ofrece opciones avanzadas de personalización y administración de usuarios, su falta de integración oficial con *React Native* y la dependencia de librerías externas no oficiales representaban un obstáculo considerable para el desarrollo. Con estos ajustes, se logró un entorno de autenticación más confiable y seguro, adecuado para las necesidades específicas del proyecto y compatible con las herramientas de desarrollo utilizadas.

Para asegurar una gestión robusta de usuarios y permisos en *AWS*, se decidió utilizar *IAM Identity Center* junto con *IAM*, debido a la flexibilidad que proporcionan en configuraciones avanzadas de acceso y autenticación. Además, se evaluó el servicio *AWS Amplify* en su versión Gen 1 para comprobar su adecuación en el desarrollo y despliegue de la aplicación. *Amplify* Gen 1 ofrece capacidades simplificadas de integración de *backends*, lo que facilitó la fase inicial de validación de servicios, permitiendo una integración fluida con *IAM* para gestionar la seguridad y autenticación.

Configuración de *IAM Identity Center* e *IAM*

La implementación de *Identity Center* junto con *IAM* en *AWS* permitió una administración detallada de usuarios y roles, lo cual fue esencial para gestionar permisos de manera estructurada y controlada. A continuación se detallan los pasos de configuración y las políticas aplicadas para una administración efectiva de identidades y acceso:

- **Configuración de *IAM Identity Center* (anteriormente *AWS SSO*):** *IAM Identity Center* ofrece una plataforma centralizada para la administración de identidades y acceso, facilitando el inicio de sesión único (*SSO*) y la administración de permisos basados en roles. La

configuración inicial incluyó la creación de un *Identity Center* desde la consola de *AWS*, estableciendo el *tenant* que centralizaría todos los accesos a recursos de *AWS*. Este paso permitió definir políticas de inicio de sesión único y controlar el acceso a aplicaciones sin necesidad de credenciales individuales, lo cual simplificó el acceso seguro para los miembros del equipo.

- **Integración con *IAM* para control de acceso detallado:** aunque *IAM Identity Center* cubre gran parte de las necesidades de acceso de usuarios y grupos, se integró con *IAM* para configurar permisos específicos que requieren una mayor personalización. *IAM* proporcionó la flexibilidad necesaria para crear políticas detalladas, asignando permisos a nivel de recurso para servicios individuales como *Lambda* y *DynamoDB*. Estas políticas *IAM* se aplicaron tanto a usuarios como a roles específicos dentro del *Identity Center*, permitiendo un control granular en funciones sensibles.
- **Creación de grupos y roles para acceso específico:** se crearon grupos y roles específicos dentro de *IAM Identity Center* para segmentar los permisos según el rol de cada miembro del equipo, como desarrolladores, administradores y analistas. Los desarrolladores recibieron permisos para acceder a recursos como Lambda y *DynamoDB*, mientras que los administradores contaban con permisos de administración para gestionar instancias *EC2* y bases de datos. Los roles de *IAM* se configuraron para ajustar permisos adicionales necesarios en casos específicos, asegurando así una separación de privilegios adecuada.
- **Activación de políticas de seguridad y auditoría en *IAM*:** además de las políticas de acceso, se implementaron configuraciones de seguridad en *IAM*, como la autenticación multifactor (*MFA*) obligatoria para usuarios con permisos elevados. Esto ayudó a asegurar que solo los usuarios verificados pudieran realizar cambios en la configuración de los recursos y sistemas. Para llevar un control exhaustivo, se activaron políticas de auditoría mediante *CloudTrail*, permitiendo registrar y *monitorear* todas las actividades relacionadas con *IAM* e *Identity Center*. Esto facilita la identificación de accesos no autorizados o actividades sospechosas.

Evaluación y configuración de *AWS Amplify Gen* 1

Para la fase inicial de desarrollo de la aplicación y la validación de servicios en *AWS*, se utilizó *AWS Amplify Gen* 1, una versión de *Amplify* que facilita la integración de *backends* y *frontends* en aplicaciones móviles y *web*. *Amplify Gen* 1 fue seleccionado en esta etapa debido a sus capacidades de integración rápida, permitiendo un desarrollo ágil y eficiente. La configuración de *Amplify* incluyó los siguientes aspectos:

- **Creación y configuración del proyecto en *Amplify Console*:** se creó un proyecto en *Amplify Console*, configurando un entorno de desarrollo con despliegue continuo desde un repositorio de *GitHub*. Esta integración permitió automatizar el despliegue de la aplicación en cada actualización de código, acelerando el flujo de trabajo y facilitando pruebas frecuentes. *Amplify Console* también ofreció herramientas para monitorear el estado de la aplicación y el historial de despliegues.
- **Autenticación con *Amplify Auth* e integración con *IAM*:** *Amplify Auth* fue configurado para gestionar la autenticación de usuarios, integrándose con *IAM* para aplicar políticas de acceso específicas en base a roles. Esto incluyó la creación de un grupo de usuarios en *Amazon Cognito*, donde se registraron las credenciales de prueba de los usuarios y se configuraron flujos de autenticación como el inicio de sesión por correo electrónico y la recuperación de contraseñas. La integración con *IAM* permitió aplicar políticas de acceso a estos usuarios, asegurando que solo los usuarios autenticados pudieran acceder a ciertos recursos.
- **Implementación de servicios de *backend* (*API* y base de datos):** se configuraron servicios de *backend* directamente desde *Amplify*, incluyendo la creación de una *API GraphQL*

para gestionar la comunicación entre la aplicación y la base de datos. *Amplify* simplificó el proceso de creación de esquemas *GraphQL* y resoluciones de datos, permitiendo que la *API* se conectara a una base de datos *DynamoDB* sin requerir configuraciones avanzadas. Esto proporcionó un *backend* escalable y seguro para el desarrollo inicial de la aplicación.

- **Monitoreo y administración de *Amplify*:** *AWS Amplify Console* incluye opciones de monitoreo que permitieron realizar un seguimiento detallado del uso y el rendimiento de la aplicación. Estas herramientas fueron utilizadas para identificar problemas de rendimiento y optimizar los tiempos de respuesta de la *API*. Además, *Amplify Console* ofreció métricas sobre las tasas de éxito en autenticación y acceso a la base de datos, facilitando el análisis y diagnóstico de cualquier inconveniente en el flujo de usuarios.
- **Limitaciones de *AWS Amplify Gen* 1 y evaluación para el futuro:** aunque *Amplify Gen* 1 proporcionó una base sólida para el desarrollo inicial, se identificaron algunas limitaciones en la personalización y el control de ciertos componentes, especialmente para casos de uso avanzados o configuraciones de seguridad más estrictas. En futuras fases de desarrollo, se considerarán versiones actualizadas de *Amplify*, así como configuraciones adicionales en servicios de *backend* para ampliar la flexibilidad y escalabilidad del sistema.

La configuración combinada de *IAM Identity Center, IAM* y *Amplify Gen* 1 en *AWS* permitió establecer una base de autenticación segura y un *backend* funcional que facilitó el desarrollo rápido de la aplicación. Estas configuraciones iniciales sirvieron para validar los servicios de *AWS* en un entorno de desarrollo, proporcionando al equipo un entorno seguro y optimizado para la creación y despliegue de aplicaciones escalables en la nube.

5.25.1. Comparativa entre desarrollos

Luego de la fase inicial de configuración y validación de servicios, se decidió continuar con el desarrollo en *AWS*, dada su compatibilidad directa y las ventajas específicas que ofrece para aplicaciones desarrolladas en *React Native*. La elección de *AWS* para el desarrollo completo resultó beneficiosa al proporcionar un entorno de integración fluido con herramientas orientadas a aplicaciones móviles y *frontends* basados en *JavaScript*, como es el caso de *React Native*, en comparación con las opciones que ofrece *Azure* para este tipo de tecnologías.

Ventajas de *AWS* para aplicaciones en *React Native*

Al optar por *AWS*, se identificaron varias ventajas significativas en cuanto a soporte y facilidad de integración con aplicaciones basadas en *React Native*. Estas ventajas, en gran medida, facilitaron el desarrollo, asegurando una experiencia de usuario optimizada y reduciendo la necesidad de configuraciones adicionales. A continuación, se detallan algunos de estos beneficios:

- **Integración con *Amplify Libraries* para *React Native*:** *AWS Amplify* proporciona una serie de librerías específicas para *React Native*, diseñadas para facilitar la implementación de autenticación, almacenamiento de datos, notificaciones *push* y gestión de archivos directamente en aplicaciones móviles. Estas librerías son oficialmente mantenidas por *AWS*, lo que garantiza compatibilidad continua con actualizaciones del *framework React Native* y reduce el esfuerzo de mantenimiento.
- **Autenticación simplificada mediante *Amplify Auth* y *Cognito*:** *AWS Cognito*, en combinación con *Amplify Auth*, ofreció un sistema de autenticación de usuarios seguro y fácil de implementar en *React Native*. A diferencia de *Azure*, que en ciertos casos requiere la utilización de librerías externas para aplicaciones móviles, *AWS* proporciona una solución nativa y

bien documentada que permite a los desarrolladores configurar autenticación y autorización de usuarios sin la dependencia de servicios adicionales o librerías no oficiales.

- **Servicios de *backend* sin servidor:** la infraestructura sin servidor de *AWS*, en particular *AWS Lambda* y *DynamoDB*, complementó perfectamente el desarrollo de *backend* para la aplicación en *React Native*. Estos servicios permiten una escalabilidad automática y un coste optimizado, ajustándose a las demandas de carga en tiempo real de la aplicación. En comparación, aunque *Azure* también cuenta con funciones sin servidor (*Azure Functions* y *Cosmos DB*), la integración nativa en *AWS* con *Amplify* facilitó la configuración y el despliegue sin necesidad de pasos adicionales de configuración.
- **Soporte para notificaciones *push* y sincronización en tiempo real:** *AWS Amplify* facilita la integración de notificaciones *push* y la sincronización en tiempo real, utilizando servicios como *Amazon SNS* y *AppSync*. Estas funcionalidades son esenciales para aplicaciones móviles, especialmente en *React Native*, donde es fundamental mantener a los usuarios informados y proporcionar una experiencia interactiva sin latencias. En el caso de *Azure*, aunque ofrece *Azure Notification Hubs*, el proceso de configuración para *React Native* suele requerir pasos adicionales y ajustes de compatibilidad, lo cual puede incrementar la complejidad del desarrollo.
- **Amplia documentación y comunidad de soporte:** *AWS Amplify* cuenta con una extensa documentación y una comunidad activa de desarrolladores enfocados en *React Native*. La documentación oficial incluye guías específicas para integrar *Amplify* con *React Native*, lo que aceleró el proceso de implementación y permitió resolver dudas rápidamente. *Azure*, aunque ofrece buenas opciones para aplicaciones móviles, cuenta con una comunidad más limitada y menos recursos específicamente orientados a *React Native*, lo cual puede ser una desventaja en la solución de problemas y optimización de la integración.

Comparativa entre *AWS* y *Azure* en el desarrollo de *React Native*

En resumen, la decisión de continuar el desarrollo en *AWS*, en lugar de *Azure*, se basó en la compatibilidad y eficiencia que *AWS* ofrece para aplicaciones en *React Native*. Las librerías oficiales de *Amplify* para *React Native* brindaron una solución nativa para funcionalidades esenciales como autenticación, almacenamiento y sincronización en tiempo real. La infraestructura sin servidor de *AWS*, además, proporcionó escalabilidad y flexibilidad sin requerir configuraciones adicionales o librerías externas.

En contraste, *Azure*, aunque cuenta con una gama de herramientas robustas para aplicaciones empresariales y la integración con herramientas como *Visual Studio*, presenta algunas limitaciones en el soporte directo para *React Native*, lo que en ciertos casos requiere el uso de soluciones no oficiales. Esto puede impactar en el mantenimiento y la actualización de la aplicación, especialmente cuando se trabaja en entornos dinámicos y se busca una experiencia de usuario fluida en dispositivos móviles.

Con *AWS*, se logró un entorno de desarrollo optimizado para *React Native*, permitiendo una implementación eficiente y asegurando la compatibilidad a largo plazo con las actualizaciones de este *framework*. Esta elección facilitó el desarrollo continuo y aseguró una plataforma escalable y bien documentada para respaldar el crecimiento de la aplicación y estas herramientas estan disponibles para el equipo del megaproyecto para que trabajen con ellas o tengan ese espacio.

5.25.2. Configuración de inicio de sesión con *Google* y *Facebook*

Como parte de la estrategia para optimizar la experiencia de usuario y facilitar el registro e inicio de sesión en la aplicación, se incluyó la autenticación mediante cuentas de *Google* y *Facebook*.

Esta funcionalidad permite a los usuarios autenticarse sin la necesidad de crear una cuenta específica para la aplicación, simplificando así el proceso de acceso y brindando una capa adicional de confianza mediante el uso de proveedores de identidad conocidos. La configuración de estas opciones de inicio de sesión se llevó a cabo a través de las plataformas de *Google Cloud* y *Facebook Developer*, donde se habilitaron y configuraron los parámetros necesarios para integrar el inicio de sesión social con el *backend* y el *frontend* de la aplicación.

Configuración de inicio de Sesión en *Google Cloud*

Para habilitar el inicio de sesión con *Google*, se utilizó la plataforma de *Google Cloud Console*, que proporciona herramientas para configurar credenciales de cliente *OAuth* 2.0 y gestionar la integración de autenticación en aplicaciones móviles. La configuración en *Google Cloud Console* incluyó los siguientes pasos:

- **Creación del proyecto en *Google Cloud Console*:** se inició el proceso creando un proyecto específico en *Google Cloud Console*, que permite gestionar todos los recursos asociados a la autenticación de la aplicación. Este proyecto centraliza la administración de credenciales y facilita la supervisión de todas las solicitudes de autenticación realizadas por los usuarios a través de *Google*.
- **Configuración de credenciales *OAuth* 2.0:** en la sección de credenciales de *Google Cloud Console*, se generaron las credenciales *OAuth* 2.0 necesarias para la autenticación. Este proceso incluyó la creación de un ID de cliente y un secreto de cliente, ambos necesarios para que la aplicación pueda solicitar permisos de autenticación en nombre de los usuarios. Las credenciales generadas permiten que la aplicación utilice el flujo de autorización de *OAuth* 2.0, gestionando de manera segura el acceso a los datos de autenticación de los usuarios.
- **Especificación del *URI* de redirección (*Redirect URI*):** como parte de la configuración, se especificó el *URI* de redirección (*Redirect URI*) que permite a *Google* redirigir a los usuarios de vuelta a la aplicación tras un intento de autenticación. Este *URI* es fundamental para completar el flujo de autenticación de *OAuth* 2.0, ya que asegura que los usuarios sean redirigidos correctamente una vez que *Google* haya autorizado el acceso. *Google Cloud* facilita la configuración de este *URI*, permitiendo una integración sin complicaciones con aplicaciones desarrolladas en *frameworks* como *React Native*.
- **Configuración del *SHA-1* del certificado de firma:** *Google Cloud* requiere un certificado de firma *SHA-1* para autenticar solicitudes provenientes de aplicaciones *Android*. Para obtener este certificado, se utilizó *Expo* en el *frontend*, junto con *EAS* (*Expo Application Services*), que permite generar certificados de firma automáticamente. El *SHA-1* se configuró en *Google Cloud Console* como un requisito para identificar de forma única la aplicación en dispositivos *Android*, lo cual garantiza que solo aplicaciones firmadas puedan solicitar autenticación a *Google*.

Configuración de inicio de sesión en *Facebook Developer*

Para el inicio de sesión mediante *Facebook*, se utilizó la plataforma *Facebook Developer*, que permite configurar aplicaciones y gestionar permisos de autenticación mediante la *API* de *Facebook Login*. La configuración en *Facebook Developer* se llevó a cabo con los siguientes pasos:

- **Creación de una aplicación en *Facebook Developer*:** se creó una aplicación en *Facebook Developer Console*, la cual actúa como un contenedor que gestiona los permisos y datos de autenticación solicitados a los usuarios de *Facebook*. Esta aplicación también permite monitorear el rendimiento de la autenticación y los datos de acceso proporcionados por los usuarios.

- **Generación de la clave de la aplicación (*App ID*) y el secreto de la aplicación (*App Secret*):** *Facebook Developer* proporciona un *App ID* y un *App Secret*, necesarios para realizar solicitudes de autenticación de usuarios. Estos identificadores permiten a la aplicación realizar peticiones de autenticación seguras y válidas, asegurando que los datos de los usuarios sean gestionados correctamente. Tanto el *App ID* como el *App Secret* fueron almacenados de manera segura en el *backend* de la aplicación para evitar accesos no autorizados.
- **Configuración del *URI* de redirección (*Redirect URI*):** al igual que en *Google Cloud*, *Facebook Developer* requiere especificar un *Redirect URI* para redirigir a los usuarios de vuelta a la aplicación después de autenticar su cuenta de *Facebook*. Este *URI* asegura que los usuarios regresen a la aplicación de manera segura, y que se complete el flujo de autenticación correctamente. *Facebook* facilita la configuración de este *URI*, proporcionando opciones específicas para aplicaciones móviles, lo que permite una integración sencilla con *Expo* y *React Native*.
- **Certificados de firma y *SHA-1* para integración en *Android*:** la autenticación mediante *Facebook* en aplicaciones *Android* requiere la inclusión de un certificado de firma *SHA-1*, similar al proceso en *Google Cloud*. *EAS* (*Expo Application Services*) fue utilizado para obtener y configurar este certificado en la consola de *Facebook Developer*, garantizando que únicamente las solicitudes de autenticación provenientes de la aplicación oficial sean procesadas. Esta configuración contribuyó a una autenticación segura y precisa en dispositivos *Android*.

Implementación en el proyecto con *Expo* y *EAS*

Dado que el *frontend* de la aplicación fue desarrollado en *React Native* utilizando *Expo*, fue necesario realizar configuraciones adicionales para asegurar una integración exitosa con los servicios de autenticación de *Google* y *Facebook*. *Expo Application Services* (*EAS*) facilitó el proceso de obtención de certificados y paquetes requeridos para completar la autenticación de manera segura en ambas plataformas. A continuación, se describen los aspectos específicos en la implementación con *Expo* y *EAS*:

- **Generación de certificados con *EAS* para *Android* y *iOS*:** *EAS* simplificó la generación de certificados necesarios para configurar la autenticación en dispositivos móviles. Para *Android*, *EAS* permitió obtener el *SHA-1* necesario tanto para *Google* como para *Facebook*, sin necesidad de configuraciones manuales complejas. Para *iOS*, *EAS* facilitó la obtención del certificado de desarrollo y el perfil de aprovisionamiento necesarios para la autenticación de usuarios en dispositivos *Apple*.
- **Integración de paquetes de Autenticación para *React Native*:** *Expo* proporciona paquetes oficiales para integrar *Google* y *Facebook Login* en aplicaciones *React Native*. Estos paquetes permiten gestionar el flujo de autenticación de manera nativa, facilitando el proceso de autorización y garantizando una experiencia de usuario fluida. La integración de estos paquetes en el proyecto permitió realizar solicitudes de autenticación seguras, procesar *tokens* de acceso y acceder a la información de perfil básico de los usuarios de manera sencilla y segura.
- **Configuración de la redirección y manejo de *tokens* en el *frontend*:** la autenticación mediante *Google* y *Facebook* requiere manejar *tokens* de acceso en el *frontend*. *Expo* simplificó este proceso, permitiendo almacenar y gestionar los *tokens* de autenticación de manera segura en la aplicación. Estos *tokens* son utilizados para validar las sesiones de los usuarios y autenticar su identidad al interactuar con el *backend* de la aplicación, garantizando la protección de la información del usuario y manteniendo la integridad de las sesiones activas.

La implementación de *Google* y *Facebook Login* en la aplicación, gracias a *Google Cloud* y *Facebook Developer*, fue un proceso relativamente sencillo debido a las herramientas y documentación ofrecidas

por ambas plataformas. La integración de *Expo* y *EAS* en el desarrollo con *React Native* facilitó la generación de certificados y configuraciones necesarias, asegurando una autenticación fluida y segura. Esta solución no solo mejoró la experiencia de usuario, sino que también fortaleció la seguridad y confiabilidad de los accesos en la aplicación, utilizando proveedores de autenticación ampliamente reconocidos.

Despliegue de la *API* con *Node.js* y *Express* en *Elastic Beanstalk*

El equipo de *backend* desarrolló una *API* en *Node.js* utilizando el *frameworks Express*, destinada a gestionar las solicitudes y manejar la lógica del negocio. La implementación en *Elastic Beanstalk* permitió desplegar esta *API* de manera rápida, al tiempo que se aseguraba la escalabilidad y la flexibilidad para manejar la carga variable. *Elastic Beanstalk* simplificó el proceso de despliegue mediante la automatización de configuraciones esenciales como el balanceo de carga, el escalado automático y la supervisión de rendimiento.

- **Creación y configuración del entorno en *Elastic Beanstalk*:** para desplegar la *API*, se creó un entorno en *Elastic Beanstalk* configurado para ejecutar *Node.js*. *Elastic Beanstalk* permite seleccionar el *runtime* de Node.js y configurar ajustes básicos de la instancia, como el tipo de instancia *EC2*, la disponibilidad de balanceo de carga y el escalado automático, proporcionando un entorno completamente administrado. Esta configuración facilitó el despliegue de la *API* con capacidades automáticas de recuperación y escalado.
- **Configuración del origen del código y despliegue continuo:** la *API* en *Node.js* fue vinculada a un repositorio de *GitHub* para facilitar el despliegue continuo y la actualización de código. *Elastic Beanstalk* permite integrar directamente el origen del código desde servicios de repositorios y gestionar el flujo de despliegue sin necesidad de intervenciones manuales. Cada vez que el equipo realizaba cambios en el código y se enviaban actualizaciones al repositorio, *Elastic Beanstalk* se encargaba de desplegar automáticamente la nueva versión en el entorno productivo, asegurando un proceso de actualización sin interrupciones.

Problemas y reconfiguración en *Elastic Beanstalk* debido a cambios en *AWS*

El 1 de octubre, *AWS* implementó ciertos cambios en *Elastic Beanstalk*, lo cual afectó el despliegue inicial de la *API*. Estos cambios, relacionados con configuraciones predeterminadas en *Elastic Beanstalk*, introdujeron ajustes en los roles de permisos y en las configuraciones de red, lo que generó problemas de compatibilidad con la infraestructura previamente configurada. Como resultado, el equipo de desarrollo se vio obligado a reconfigurar varios aspectos del entorno en *Elastic Beanstalk* para asegurar el funcionamiento correcto de la *API*.

- **Reconfiguración de roles y permisos:** uno de los principales problemas derivados de los cambios en *Elastic Beanstalk* fue la necesidad de actualizar los roles y permisos en *IAM*. *Elastic Beanstalk* comenzó a requerir permisos adicionales para acceder a ciertos servicios de red y monitoreo, lo que requirió la modificación de las políticas asignadas a los roles de *Elastic Beanstalk* y a las instancias *EC2* asociadas. Esta reconfiguración garantizó que la *API* tuviera acceso adecuado a los recursos necesarios y que pudiera ejecutarse sin problemas en el entorno.
- **Ajustes en la configuración de red y seguridad:** los cambios también afectaron las configuraciones de red del entorno en *Elastic Beanstalk*, como las reglas de seguridad en los grupos de seguridad y el acceso a *VPCs* específicas. Se realizó una actualización en las configuraciones de red para permitir las conexiones adecuadas entre la *API* y la base de datos *RDS*, asegurando que *Elastic Beanstalk* pudiera gestionar correctamente las solicitudes de entrada y salida sin conflictos de permisos o limitaciones de acceso.

Configuración de *RDS* con *PostgreSQL* para almacenamiento de datos

Para el almacenamiento de datos de la aplicación, se utilizó *Amazon RDS (Relational Database Service*) con *PostgreSQL* como motor de base de datos. *RDS* fue configurado para facilitar la administración de la base de datos sin necesidad de realizar tareas de mantenimiento manuales, como la instalación de parches, la configuración de *backups* automáticos y el escalado de recursos.

- **Creación de la instancia *RDS* (*PostgreSQL*):** se creó una instancia de *RDS* con *PostgreSQL* configurada en una instancia de tamaño "*small*"para ajustarse a las necesidades iniciales del proyecto. Este tamaño de instancia proporcionó el rendimiento y el almacenamiento necesarios para manejar las consultas y datos de la *API* sin incurrir en altos costos. La elección de *PostgreSQL* se basó en su robustez y compatibilidad con *Node.js,* lo cual facilitó la integración entre la *API* y la base de datos.
- **Configuración de conexiones seguras y acceso restringido:** la instancia de *RDS* fue configurada con un grupo de seguridad personalizado que permitía conexiones únicamente desde el entorno de *Elastic Beanstalk*, asegurando que solo la *API* tuviera acceso directo a la base de datos. Esta configuración fue importante para garantizar la seguridad de los datos y prevenir accesos no autorizados. Además, se habilitó el cifrado en tránsito para todas las conexiones a la base de datos, proporcionando una capa adicional de seguridad.
- ***Backups* automáticos y monitoreo:** *Amazon RDS* ofrece la opción de *backups* automáticos y recuperación ante desastres, lo cual se configuró para garantizar la disponibilidad de los datos en caso de fallos. Estos *backups* automáticos permitieron establecer puntos de restauración diarios de la base de datos, minimizando la pérdida de datos ante posibles problemas. *RDS* también ofrece monitoreo detallado de rendimiento, permitiendo al equipo observar métricas críticas como el uso de *CPU*, la latencia de las consultas y la capacidad de almacenamiento utilizada.

Beneficios de la implementación en *Elastic Beanstalk* y *RDS*

La combinación de *Elastic Beanstalk* y *Amazon RDS* para el despliegue de la *API* y el almacenamiento de datos resultó en una solución altamente escalable y segura, que facilitó la administración de la infraestructura sin la necesidad de configurar manualmente cada aspecto del *backend*. *Elastic Beanstalk* ofreció una plataforma gestionada con opciones de escalado y balanceo de carga, permitiendo al equipo de desarrollo centrarse en el código y la funcionalidad de la *API*. La integración de *RDS* con *PostgreSQL* proporcionó un entorno de base de datos confiable y optimizado, con capacidades de *backup* automático y configuraciones de seguridad avanzadas, ideales para una *API* de alto rendimiento y accesible.

Con esta implementación, el equipo de desarrollo pudo gestionar el despliegue de la *API* en un entorno controlado y seguro, con la capacidad de escalar según las necesidades de la aplicación. *Elastic Beanstalk* y *RDS* juntos formaron una infraestructura sólida, preparada para el crecimiento y fácil de mantener, incluso ante cambios en las configuraciones y políticas de *AWS*.

5.25.3. Despliegue de un modelo de IA

Como parte de la funcionalidad avanzada de la aplicación, se desplegó un modelo de inteligencia artificial en *AWS SageMaker* que permite procesar tanto imágenes como texto y generar respuestas en formato de texto. Este modelo fue diseñado para llevar a cabo tareas de procesamiento de lenguaje natural y análisis de contenido visual, brindando la capacidad de responder a las solicitudes de los usuarios en función de los datos visuales y textuales proporcionados. El modelo fue previamente

entrenado y optimizado por el equipo de IA, y en esta fase simplemente se tomó el modelo final y se desplegó en el entorno productivo de *AWS SageMaker*.

Entrenamiento y optimización del modelo

El entrenamiento del modelo fue realizado íntegramente por el módulo de IA, quien se encargó de desarrollar una red neuronal que pudiera analizar datos multimodales, es decir, datos que incluyen tanto imágenes como texto. Este equipo llevó a cabo la preparación de los datos y la elección de la arquitectura adecuada para el modelo, con el fin de asegurar un rendimiento óptimo en tareas de reconocimiento y generación de respuestas.

- **Preparación del conjunto de datos:** el equipo de IA preparó un conjunto de datos balanceado que consistía en pares de imágenes y texto, donde cada imagen estaba asociada con una descripción textual correspondiente. Este conjunto de datos fue preprocesado y almacenado en *Amazon S3*, permitiendo que *SageMaker* accediera fácilmente a los datos durante el proceso de entrenamiento. La estructura de estos datos incluía imágenes etiquetadas y secuencias de texto, lo cual permitió al modelo aprender las relaciones entre características visuales y contenido textual relevante.
- **Selección de la arquitectura del modelo:** para la implementación del modelo, el equipo de IA optó por una arquitectura de red neuronal convolucional para el procesamiento de imágenes, combinada con un modelo de lenguaje basado en *transformers* para el análisis de texto. Esta arquitectura multimodal permitió que el modelo no solo interpretara el contenido visual, sino también el texto asociado, generando respuestas en texto en función del contexto total de la entrada.

Despliegue en *AWS SageMaker*

Una vez que el módulo de IA completó el entrenamiento y optimización del modelo, el equipo de desarrollo se encargó de tomar el modelo final y desplegarlo en *AWS SageMaker* para su uso en producción. *AWS SageMaker* facilitó el proceso de despliegue al proporcionar una infraestructura gestionada y escalable para alojar modelos de IA, permitiendo una integración fluida con el resto de la aplicación.

- **Carga del modelo en *SageMaker*:** el modelo entrenado y optimizado por el equipo de IA fue almacenado en *Amazon S3* como un archivo serializado. Desde *AWS SageMaker*, se creó un *endpoint* para servir el modelo y se cargó el archivo desde *S3* a *SageMaker*, utilizando el entorno administrado para desplegar el modelo sin requerir configuraciones complejas de servidores o infraestructura adicional.
- **Configuración del *endpoint* de *SageMaker*:** para el despliegue en producción, se configuró un *endpoint* en *SageMaker* que permite a la aplicación enviar solicitudes de entrada y recibir respuestas en tiempo real. Este *endpoint* fue configurado para recibir datos en formato *JSON*, permitiendo que la aplicación enviara tanto imágenes como texto como parte de la entrada. *AWS SageMaker* maneja la lógica de inferencia, lo que simplifica la implementación y permite al modelo responder rápidamente a las consultas de los usuarios.
- **Escalabilidad y monitoreo:** *AWS SageMaker* proporciona opciones de escalado automático, lo que permite ajustar la capacidad del *endpoint* según la demanda de los usuarios. Además, se habilitó el monitoreo de rendimiento utilizando *Amazon CloudWatch*, lo cual permite rastrear métricas como el tiempo de respuesta y el uso de *CPU*, garantizando que el modelo funcione de manera eficiente y pueda escalarse si el volumen de solicitudes aumenta.

Ventajas de utilizar *AWS SageMaker* para el despliegue

La elección de *AWS SageMaker* para el despliegue del modelo de IA trajo múltiples beneficios, ya que *SageMaker* proporciona un entorno completamente gestionado que facilita el despliegue, escalado y monitoreo de modelos de aprendizaje automático sin necesidad de gestionar la infraestructura. Además, el proceso de integración con otros servicios de *AWS*, como *S3* para el almacenamiento de datos y *CloudWatch* para el monitoreo, hizo que el despliegue fuera ágil y adaptable a los cambios en el uso de la aplicación. Con *SageMaker*, el equipo pudo desplegar el modelo final del módulo de IA en un entorno productivo robusto y con la flexibilidad necesaria para ajustarse a las necesidades de la aplicación.

5.25.4. Análisis de costos y mantenimiento

5.26. Estimación de costos y mantenimiento para diferentes escenarios de afluencia

Para establecer un presupuesto de costos y mantenimiento en *AWS*, se realizó un análisis detallado de los servicios involucrados en la aplicación en función de tres niveles de uso o afluencia: baja, media y alta. Cada nivel de afluencia representa un rango de interacciones y cargas de trabajo, y se han configurado los recursos de *AWS* para ajustar el rendimiento de la aplicación a la demanda de cada escenario. A continuación, se presentan las estimaciones de costos y los requisitos de mantenimiento para cada servicio implementado.

5.26.1. Interfaz de programación de aplicaciones desarrollada

AWS Elastic Beanstalk permite desplegar y gestionar aplicaciones *web* utilizando *Amazon EC2*. Este servicio facilita la configuración automática de servidores y el escalado de instancias según la carga de trabajo, por lo que se eligió para alojar la *API* desarrollada en *Node.js* con el *framework Express*.

- **Afluencia baja**:
  - **Uso**: se utiliza una instancia *EC2* de tipo `t3.micro` (1 *vCPU*, 1 GB de *RAM*) en ejecución continua.
  - **Costo aproximado**: la instancia `t3.micro`, ejecutándose de manera continua en *Elastic Beanstalk*, tiene un costo estimado de aproximadamente **$8.50 al mes**.
- **Afluencia media**:
  - **Uso**: se implementan dos instancias `t3.small` (2 *vCPU*, 2 GB de *RAM*) con balanceador de carga para manejar un aumento de solicitudes.
  - **Costo aproximado**: con dos instancias `t3.small` y un balanceador de carga en *Elastic Beanstalk*, el costo aproximado es de **$35 al mes**.
- **Afluencia alta**:
  - **Uso**: se configuran cuatro instancias `t3.medium` (2 *vCPU*, 4 GB de *RAM*) con balanceador de carga y autoescalado, lo cual permite soportar una alta demanda.
  - **Costo aproximado**: el costo estimado para cuatro instancias `t3.medium` en *Elastic Beanstalk* con balanceador de carga y escalado automático es de aproximadamente **$140 al mes**.

5.26.2. Base de datos relacional

Amazon RDS (Relational Database Service) se emplea para gestionar la base de datos *PostgreSQL* que almacena la información crítica de la aplicación. Los costos se ajustan de acuerdo con el tipo de instancia, el tamaño de almacenamiento y la configuración de alta disponibilidad según el nivel de afluencia.

- **Afluencia *aaja***:
  - **Uso**: se utiliza una instancia `db.t3.micro` con 20 GB de almacenamiento *SSD*, adecuada para una carga ligera de solicitudes.
  - **Costo aproximado**: la instancia `db.t3.micro` en *RDS* tiene un costo aproximado de **$15 al mes**.
- **Afluencia media**:
  - **Uso**: se configura una instancia `db.t3.small` con 50 GB de almacenamiento *SSD* para soportar una mayor cantidad de transacciones.
  - **Costo aproximado**: con una instancia `db.t3.small` y 50 GB de almacenamiento, el costo estimado es de **$40 al mes**.
- **Afluencia alta**:
  - **Uso**: para alta disponibilidad y desempeño, se emplea una instancia `db.t3.medium` con 100 GB de almacenamiento *SSD* y configuración *Multi-AZ*.
  - **Costo aproximado**: la instancia `db.t3.medium` con configuración *Multi-AZ* y 100 GB de almacenamiento tiene un costo estimado de **$200 al mes**.

5.26.3. Modelo de inteligencia artificial

AWS SageMaker se utiliza para alojar el modelo de IA que procesa imágenes y texto. *SageMaker* permite inferencias en tiempo real, y los costos dependen del tipo de instancia y el tiempo de uso diario.

- **Afluencia baja**:
  - **Uso**: instancia `ml.t2.medium` operando para inferencia en tiempo real durante 4 horas al día.
  - **Costo aproximado**: el costo aproximado para una instancia `ml.t2.medium` con 4 horas de uso diario es de **$50 al mes**.
- **Afluencia media**:
  - **Uso**: instancia `ml.m5.large` operando 8 horas diarias para cubrir un aumento de solicitudes de inferencia.
  - **Costo aproximado**: el costo estimado es de **$200 al mes** para una instancia `ml.m5.large` operando 8 horas diarias.
- **Afluencia alta**:
  - **Uso**: instancia `ml.m5.xlarge` operando 24 horas al día para una alta demanda de procesamiento en tiempo real.
  - **Costo aproximado**: para una instancia `ml.m5.xlarge` con operación continua, el costo estimado es de **$600 al mes**.

5.26.4. Servicios de autenticación

AWS Amplify facilita la autenticación de usuarios mediante proveedores como *Google* y *Facebook*. Los costos varían según la cantidad de solicitudes de autenticación.

- **Afluencia baja**:
  - **Uso**: hasta 50,000 solicitudes de autenticación mensuales, dentro del nivel gratuito de *AWS*.
  - **Costo aproximado**: el uso de *Amplify* para esta carga está cubierto por el nivel gratuito, por lo que el costo es de **$0 al mes**.
- **Afluencia media**:
  - **Uso**: hasta 500,000 solicitudes de autenticación mensuales.
  - **Costo aproximado**: para una carga media, el costo estimado es de **$25 al mes**.
- **Afluencia alta**:
  - **Uso**: hasta 5,000,000 solicitudes de autenticación mensuales.
  - **Costo aproximado**: para una alta afluencia, el costo estimado es de **$250 al mes**.

5.26.5. Almacenamiento de datos

Amazon S3 proporciona almacenamiento de datos escalable. Los costos se calculan en función de la cantidad de datos almacenados y el número de solicitudes de lectura y escritura.

- **Afluencia baja**:
  - **Uso**: almacenamiento de hasta 50 GB de datos.
  - **Costo aproximado**: para 50 GB en *S3*, el costo estimado es de aproximadamente **$1.15 al mes**.
- **Afluencia media**:
  - **Uso**: almacenamiento de hasta 200 GB de datos.
  - **Costo aproximado**: para 200 GB en *S3*, el costo estimado es de aproximadamente **$4.60 al mes**.
- **Afluencia alta**:
  - **Uso**: almacenamiento de hasta 1 TB de datos.
  - **Costo aproximado**: para 1 TB en *S3*, el costo estimado es de aproximadamente **$23 al mes**.

5.26.6. Resumen de costos mensuales estimados

- **Afluencia baja**: aproximadamente **$75 al mes**.
- **Afluencia media**: aproximadamente **$300 al mes**.
- **Afluencia alta**: aproximadamente **$1,200 al mes**.

5.26.7. Consideraciones de mantenimiento

Para mantener un entorno optimizado y seguro, se han incluido los siguientes aspectos en el plan de mantenimiento:

- **Monitoreo y alertas**: utilizar *Amazon CloudWatch* para monitorear el rendimiento y establecer alertas que permitan identificar y resolver problemas de manera proactiva.
- **Actualizaciones de seguridad**: mantener actualizados los servicios y aplicar parches de seguridad de acuerdo con las recomendaciones de *AWS* para evitar vulnerabilidades.
- **Optimización de costos**: revisar periódicamente el uso de recursos y ajustar los servicios en función de la demanda, aplicando configuraciones de autoescalado para evitar costos adicionales innecesarios.

Estas estimaciones son aproximadas y pueden variar según la región y el uso específico. Para una precisión mayor, se recomienda emplear la calculadora de precios de *AWS*.

5.27. Metodología de seguridad

El desarrollo del proyecto se llevó a cabo en varias fases, cada una enfocada en la implementación de buenas prácticas de seguridad y la mejora continua en el análisis de la calidad del código. La metodología adoptada se basó en el marco de trabajo *OWASP SAMM*, abarcando prácticas en gobernanza, diseño, implementación y verificación para garantizar la seguridad del *software*. A continuación, se detalla el proceso en cada una de las fases:

5.28. Fase 1: selección de tecnologías

El primer paso fundamental del proyecto consistió en realizar una investigación exhaustiva sobre las herramientas disponibles para el análisis estático y dinámico del código, lo cual permitió identificar las tecnologías que mejor se ajustaban a los objetivos de seguridad y calidad establecidos. Las herramientas seleccionadas se centraron en dos áreas clave: análisis estático (*SAST*) y análisis dinámico (*DAST*), cada uno con enfoques específicos para aplicaciones de *backend* y *frontend* móvil.

5.28.1. Herramientas de análisis estático de código

Para el análisis estático, se investigaron varias opciones reconocidas en la industria, tales como *SonarQube*, *Collaborator* y *Embold*. Cada una de estas herramientas ofrece funcionalidades avanzadas para detectar vulnerabilidades y mejorar la calidad del código, aunque presentan diferencias significativas en términos de integración, facilidad de uso y cobertura de estándares.

- ***SonarQube*:** *SonarQube* fue evaluada como la herramienta con mayores ventajas para el análisis estático de código en este proyecto. Su facilidad de configuración e integración con repositorios como *GitHub* permite la exportación y análisis directo del código fuente. *SonarQube* también incluye de forma nativa reglas y estándares de calidad como el *clean code* de Robert C. Martin, *OWASP Top 10*, *SANS Top 25* y *CWE (Common Weakness Enumeration)*, lo que proporciona una cobertura completa de los aspectos más críticos de seguridad. Además, *SonarQube* identifica diversas categorías de problemas, tales como *bugs*, *code smells*, y vulnerabilidades, detallando el origen de cada problema, su impacto en la seguridad y brindando recomendaciones sobre cómo resolverlo. Esta capacidad de análisis detallado facilita una gestión efectiva de las vulnerabilidades, razón por la cual se seleccionó *SonarQube* como la herramienta principal para el análisis estático en el *backend*.

SonarQube ha sido utilizado por grandes empresas como ***eBay*** y ***NASA***, quienes lo emplean para garantizar la seguridad y la calidad del código en sus desarrollos. En el caso de *eBay*, por ejemplo, *SonarQube* contribuyó a mantener altos estándares de seguridad en su plataforma de comercio electrónico, una aplicación que procesa millones de transacciones diarias y debe cumplir con estrictos controles de seguridad para proteger la información de los usuarios. La robustez de *SonarQube* y su capacidad de integración con herramientas de *CI/CD* lo convierten en una solución ideal para proyectos de gran escala.

- ***Collaborator*:** esta herramienta está diseñada para facilitar revisiones de código colaborativas, destacándose en entornos en los que el análisis manual y el trabajo en equipo son prioritarios. Aunque *Collaborator* ofrece características útiles para la revisión y documentación de problemas, no cuenta con las capacidades integradas de detección de vulnerabilidades y estándares de seguridad que ofrece *SonarQube*. Debido a esta limitación en la cobertura automática de estándares y su menor facilidad de integración, *Collaborator* fue descartada para el análisis estático de seguridad automatizado.

  Ejemplos de uso de *Collaborator* incluyen a ***Micro Focus*** y ***Volkswagen***, quienes lo emplean en entornos de desarrollo donde las revisiones de código son colaborativas y requieren un enfoque personalizado. *Volkswagen*, por ejemplo, lo utiliza para revisar componentes de *software* en sus sistemas de vehículos, asegurando una calidad y coherencia en el código mediante revisiones manuales detalladas.

- ***Embold*:** *Embold* es una plataforma orientada a la mejora de la calidad del código, enfocada principalmente en la detección de *code smells* y problemas de mantenibilidad. Aunque es útil para optimizar el código en términos de rendimiento y legibilidad, *Embold* no cuenta con la misma capacidad de detección de vulnerabilidades de seguridad que *SonarQube*. Por esta razón, *Embold* no fue seleccionada como la herramienta principal de *SAST*, aunque sigue siendo una opción viable para equipos que buscan optimizar el código desde una perspectiva de calidad en lugar de seguridad.

  Embold ha sido implementado en empresas como ***Siemens*** y ***Nokia***, quienes buscan mejorar la mantenibilidad de su código y reducir el costo de desarrollo a largo plazo. *Nokia*, por ejemplo, lo utiliza para mejorar la calidad del código en sus soluciones de telecomunicaciones, optimizando el rendimiento y la eficiencia del desarrollo.

5.28.2. Herramientas de análisis dinámico de código para aplicaciones móviles

Para el análisis dinámico (*DAST*) de la aplicación móvil, se exploraron herramientas como *AppSweep*, *MobSF*, *Arachni*, *OWASP ZAP* y *Burp Suite*. Las opciones seleccionadas se eligieron con base en su efectividad para analizar aplicaciones móviles y su capacidad para detectar vulnerabilidades durante la ejecución de la aplicación.

- ***AppSweep*:** *AppSweep* se destacó como la herramienta ideal para el análisis dinámico de aplicaciones móviles. Desarrollada específicamente para entornos móviles, *AppSweep* permite la detección de vulnerabilidades durante la ejecución de la aplicación, identificando problemas relacionados con permisos, almacenamiento inseguro y configuraciones deficientes en tiempo de ejecución. Además, *AppSweep* ofrece una interfaz intuitiva que facilita la interpretación de los resultados y permite integrar las pruebas en el ciclo de desarrollo. Su capacidad para identificar vulnerabilidades específicas de aplicaciones móviles la hace especialmente útil para el análisis *DAST*, lo cual justifica su elección como la herramienta primaria en este ámbito.

  Empresas como ***Samsung*** y ***Facebook*** han optado por soluciones similares en seguridad de aplicaciones móviles, aprovechando *AppSweep* para analizar vulnerabilidades en aplicaciones de uso masivo. La capacidad de *AppSweep* para integrarse en *pipelines* de *CI/CD* y su adaptabilidad para aplicaciones móviles de alto rendimiento permiten un ciclo de revisión de seguridad constante y efectivo.

- ***MobSF (Mobile Security Framework)*:** aunque *MobSF* también ofrece análisis dinámico, fue seleccionado principalmente como la herramienta de análisis estático para la aplicación móvil, debido a su enfoque integral en la seguridad de aplicaciones móviles y su capacidad para detectar una amplia variedad de problemas de seguridad antes de la ejecución. *MobSF* permite identificar vulnerabilidades de configuración, análisis de permisos, y detecta prácticas de desarrollo inseguras en el código. Este enfoque preventivo complementa el análisis *DAST* de *AppSweep*, permitiendo una cobertura más completa de los riesgos de seguridad en la aplicación móvil.

  Empresas como ***Paytm*** y ***Grab*** han implementado *MobSF* en sus procesos de desarrollo de aplicaciones móviles para garantizar que los elementos de seguridad, desde permisos hasta configuraciones, se revisen exhaustivamente antes de lanzarlas al mercado. Esto asegura un nivel de protección robusto frente a amenazas específicas de entornos móviles.

- ***Arachni*:** esta herramienta es útil para la detección de vulnerabilidades en aplicaciones *web*, aunque carece de características específicas para entornos móviles. Dado que el proyecto requiere una cobertura robusta de aplicaciones móviles, *Arachni* fue descartada en favor de opciones más especializadas como *AppSweep* y *MobSF*.

  Arachni ha sido adoptado por empresas como ***Adobe*** y ***Oracle***, que lo utilizan en sus *pipelines* de desarrollo *web* para identificar y mitigar vulnerabilidades en entornos de producción. *Adobe*, por ejemplo, lo utiliza en sus productos de *software* en la nube para identificar amenazas de seguridad y mejorar sus defensas ante posibles ataques en sus aplicaciones *web*.

- ***OWASP ZAP*:** *OWASP ZAP* es una herramienta poderosa para realizar pruebas de seguridad en aplicaciones *web*, con un enfoque en la identificación de vulnerabilidades como inyección *SQL* y *XSS*. Sin embargo, al no estar optimizada para el análisis dinámico de aplicaciones móviles, su uso fue considerado menos adecuado en comparación con *AppSweep*.

  OWASP ZAP es utilizado ampliamente en organizaciones como ***Mozilla*** y ***GitLab***. *Mozilla* lo utiliza como una herramienta esencial para pruebas de penetración en sus servicios *web*, y *GitLab* lo ha integrado en su *CI/CD* para realizar escaneos de seguridad en sus aplicaciones *web*, contribuyendo así a la seguridad de sus proyectos de código abierto y de sus clientes.

- ***Burp Suite*:** *Burp Suite* es ampliamente utilizada para pruebas de seguridad *web* y permite analizar aplicaciones móviles en ciertos contextos, aunque su enfoque principal no es el entorno móvil. Dado que *AppSweep* y *MobSF* proporcionan una mejor adaptación para los requerimientos específicos del proyecto en cuanto a aplicaciones móviles, *Burp Suite* fue descartada como herramienta principal para el análisis *DAST* en móviles.

  Burp Suite es ampliamente adoptada por empresas de seguridad como ***Trustwave*** y ***Verizon***. *Trustwave* la emplea en auditorías de seguridad de redes y aplicaciones, mientras que *Verizon* utiliza *Burp Suite* en sus servicios de consultoría de seguridad para realizar análisis de vulnerabilidades *web* y asegurar los sistemas de sus clientes frente a posibles ataques.

De los cuales se escogieron las siguiente herramientas:

SonarQube, con su capacidad para detectar *bugs*, *code smells* y vulnerabilidades a partir de una configuración que incluye *OWASP Top 10* y otros estándares relevantes, facilita una evaluación exhaustiva de posibles vulnerabilidades en el código y brinda recomendaciones claras que apoyan los esfuerzos de mitigación. Esta herramienta permite además generar informes detallados sobre las vulnerabilidades y riesgos detectados, promoviendo la corrección temprana de los problemas en el ciclo de vida del *software*, lo cual contribuye a reducir riesgos en las versiones finales de producción y fomenta el cumplimiento de los estándares de seguridad establecidos.

Por su parte, *AppSweep* realiza evaluaciones dinámicas en tiempo de ejecución de la aplicación móvil, detectando problemas de permisos, almacenamiento inseguro y configuraciones deficientes que podrían afectar la seguridad de los datos de los usuarios. Esta herramienta asegura que la aplicación cumpla con las prácticas de seguridad móvil más estrictas. *MobSF* complementa estos análisis proporcionando una revisión estática exhaustiva de permisos, configuraciones y el uso de

prácticas seguras en el desarrollo móvil, generando un entorno de confianza y robustez desde la fase de planificación hasta el despliegue, y siguiendo las recomendaciones de seguridad de *OWASP*.

La elección de estas herramientas también apoya la creación de una cultura de seguridad proactiva dentro del equipo de desarrollo, ya que los reportes específicos y detallados permiten revisiones semanales de los riesgos y vulnerabilidades, facilitando una capacidad de respuesta rápida y efectiva. En conjunto, *SonarQube*, *AppSweep* y *MobSF* no solo ofrecen una cobertura completa de las vulnerabilidades críticas, sino que también proporcionan una base sólida para implementar prácticas avanzadas de autenticación, autorización y protección de datos (como la autenticación multifactor y el control de acceso basado en roles), contribuyendo así a garantizar la integridad, confidencialidad y disponibilidad de la información de los usuarios.

Con esta combinación de herramientas, el proyecto se encuentra en una posición óptima para alcanzar el objetivo de cumplir con los más altos estándares de seguridad a lo largo de todo el ciclo de vida del *software*, logrando una plataforma segura y confiable que protege y prioriza la privacidad de los datos del usuario.

5.29. Fase 2: preparación e instalación de herramientas

El primer paso del proyecto consistió en la instalación y configuración de las herramientas necesarias para el análisis estático del código. Se implementó la versión *Community* de *SonarQube* debido a que es gratuita y ofrece funcionalidades esenciales para realizar un análisis *SAST (Static Application Security Testing)*. Esta elección se debió a la limitación presupuestaria y a que las funcionalidades adicionales de la versión *Developer* no eran críticas para los objetivos del proyecto.

Posteriormente, se configuró *SonarScanner* en las variables de entorno del sistema para su uso con *SonarQube*. Una vez finalizada la configuración, se creó un nuevo proyecto en *SonarQube* para servir como el espacio destinado a cargar y analizar el código fuente a lo largo del desarrollo del proyecto.

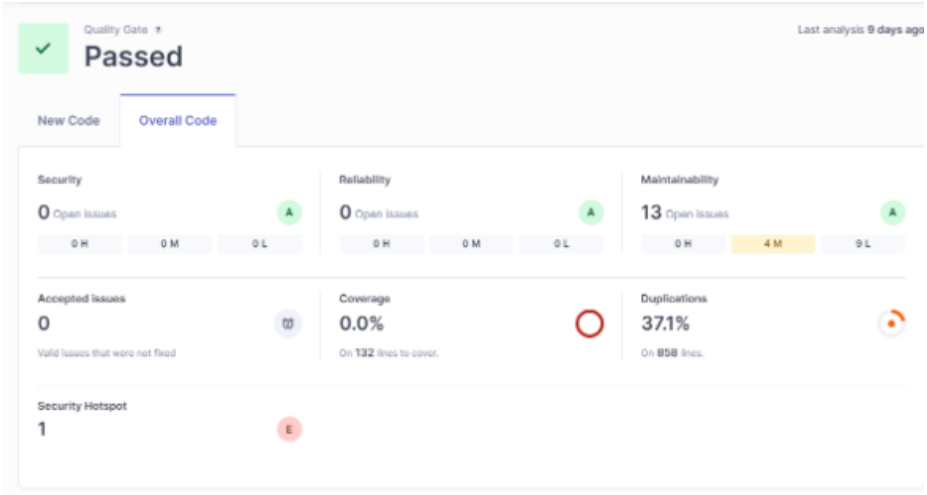


Figura 5.26. Resultado de *scan* del 5 de agosto.

Para el análisis dinámico de la aplicación móvil, se incorporaron las herramientas *AppSweep* y *MobSF*.

En el caso de *AppSweep*, la configuración inicial fue rápida y sencilla. Solo fue necesario ingresar con una cuenta de usuario en la plataforma, la cual proporciona una opción gratuita para escanear aplicaciones móviles, tanto en *Android* como en *iOS*. Esto permitió realizar evaluaciones de seguridad completas sin costos adicionales. La facilidad de uso de *AppSweep* para subir y analizar aplicaciones directamente desde la plataforma facilita la incorporación de pruebas de seguridad dentro del ciclo de desarrollo de forma ágil, como se muestra en la siguiente imagen:

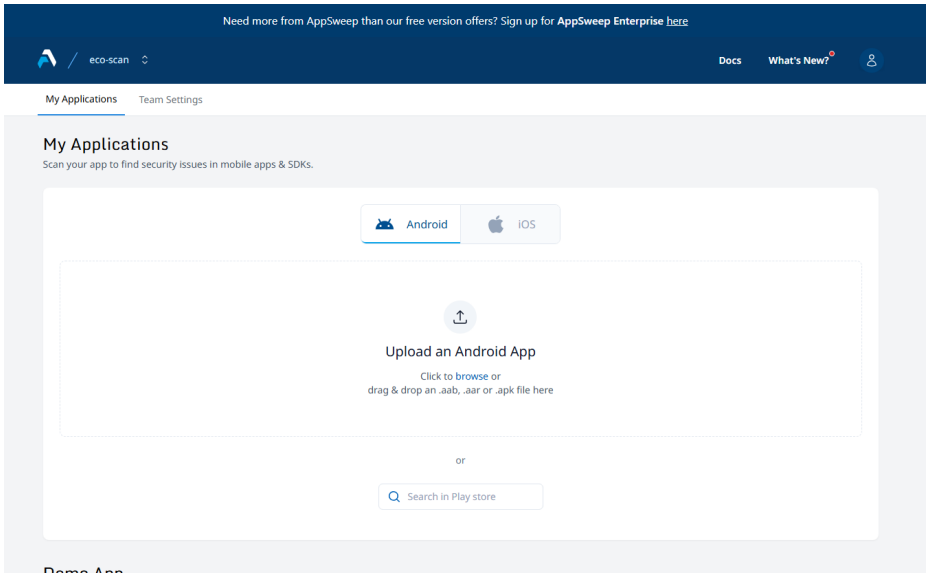


Figura 5.27. Página de inicio de *AppSweep*.

Por otro lado, *MobSF* también ofrece un portal en el que es posible cargar aplicaciones en formato `.apk` o `.ipa`, correspondientes a los sistemas *Android* e *iOS* respectivamente. Esta herramienta permite analizar la seguridad de los archivos de instalación directamente desde una interfaz *web*, proporcionando un análisis detallado de los permisos y configuraciones que podrían poner en riesgo la seguridad del usuario. La opción de *MobSF* de analizar aplicaciones sin necesidad de integración compleja agiliza su uso para evaluaciones de seguridad en entornos móviles.

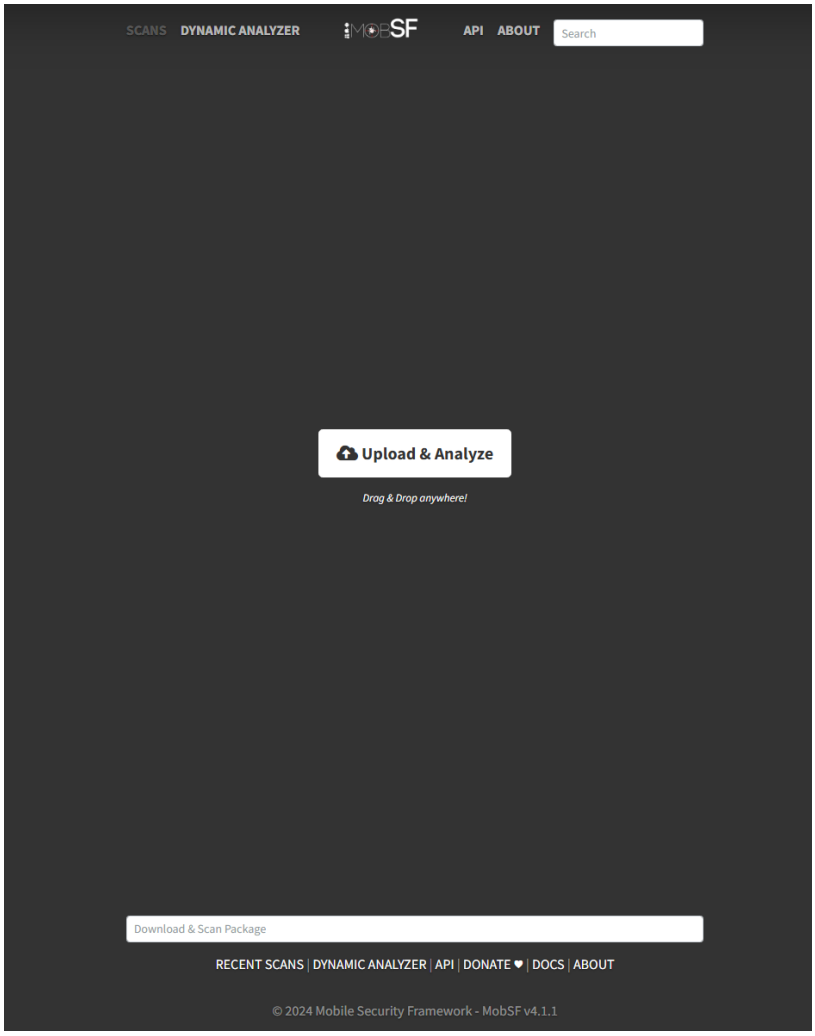


Figura 5.28. Página de carga de archivos en *MobSF*.

Además, se planificaron las tecnologías fundamentales para el proyecto, incluyendo tanto el entorno de infraestructura en la nube como los lenguajes y *frameworks* específicos para el desarrollo *frontend* y *backend*. Las decisiones se basaron en consideraciones de seguridad, compatibilidad y rendimiento, alineadas con los requisitos de la aplicación.

Infraestructura en la nube: *AWS*

El equipo encargado de la infraestructura optó por utilizar *AWS* en lugar de *Azure*, debido a la robustez de los servicios de seguridad y la facilidad de integración con herramientas de *DevOps*. Me informaron que *AWS* proporciona opciones avanzadas de seguridad, incluyendo un control detallado sobre los permisos y el cifrado de datos sensibles. Además, mencionaron que las capacidades de personalización de políticas de seguridad de *AWS* son esenciales para configurar un entorno seguro, acorde con las mejores prácticas.

Backend*: *Node.js* con *Express.js

El equipo de desarrollo *backend* seleccionó *Node.js* junto con el *framework Express.js* por sus ventajas en términos de escalabilidad y soporte para manejar múltiples solicitudes simultáneas. Me explicaron que estas características son cruciales para aplicaciones que podrían recibir un alto volumen de tráfico en tiempo real. También discutieron que *Node.js* y *Express.js* cuentan con módulos y herramientas diseñadas para seguir prácticas de codificación segura, lo cual contribuye a la protección contra ataques comunes como inyecciones *SQL* y *Cross-Site Scripting (XSS)*.

Cifrado: *Bcrypt*

El encargado del *backend* me consultó sobre la elección de un algoritmo para gestionar contraseñas y datos sensibles, considerando varias opciones. Después de investigar, recomendé *bcrypt* debido a sus ventajas de seguridad, especialmente su uso de *salting* para prevenir la reutilización de *hashes* y hacer que los ataques de fuerza bruta sean más difíciles. Esta recomendación se basó en las mejores prácticas de seguridad de *OWASP* para el manejo de contraseñas. El algoritmo fue implementado en *JavaScript*.

Frontend: React Native

El equipo de desarrollo *frontend* decidió utilizar *React Native* por su capacidad de optimizar el rendimiento en dispositivos móviles y su soporte para actualizaciones de seguridad en tiempo real. Me explicaron que *React Native* permite una fácil integración de medidas de seguridad, como el manejo seguro de la comunicación con el *backend* mediante *HTTPS* y el cifrado de datos en dispositivos. Además, destacaron su eficiencia al trabajar con bibliotecas de autenticación segura, lo cual es fundamental para proteger la información del usuario.

Con esta configuración de infraestructura y tecnologías de desarrollo, se estableció una base sólida y segura para el proyecto, optimizando tanto el rendimiento como la seguridad desde el inicio.

5.29.1. Reuniones semanales de seguimiento y planificación

Con el objetivo de asegurar el progreso continuo y la calidad del proyecto, se establecieron reuniones semanales de seguimiento y planificación, las cuales se llevaron a cabo cada viernes. Durante estas reuniones, se abordaron varios aspectos clave para el desarrollo seguro de la aplicación.

5.30. Fase 3: definición de la estrategia de seguridad

Para garantizar la seguridad y calidad del proyecto, se adoptó una estrategia basada en *OWASP SAMM*, enfocándose en las siguientes áreas:

5.30.1. Gobernanza: educación y orientación

Se priorizó la capacitación en seguridad para el equipo, proporcionando orientación en la adopción de prácticas seguras y asegurando que todos comprendieran las implicaciones de las decisiones tomadas. Esto incluyó la organización de sesiones formativas y el apoyo continuo durante el desarrollo.

5.30.2. Diseño: arquitectura segura

En colaboración con el encargado de la infraestructura, se realizaron revisiones exhaustivas de la arquitectura del sistema para asegurar que las configuraciones cumplieran con los estándares de seguridad necesarios. Se puso un énfasis particular en la implementación de controles de acceso basados en roles (*RBAC*), que permitieron asignar permisos específicos a los usuarios en función de sus responsabilidades dentro del proyecto. Este enfoque garantizó que cada usuario tuviera acceso únicamente a los recursos y funciones necesarios para llevar a cabo sus tareas, minimizando el riesgo de acceso no autorizado a información crítica o funciones sensibles.

Además, se establecieron políticas claras para la asignación y revisión periódica de los roles, asegurando que cualquier cambio en las responsabilidades de los usuarios fuera reflejado de manera oportuna en el sistema. Esto ayudó a mantener una gestión adecuada de los permisos y reforzó la protección de la infraestructura al limitar la exposición de los recursos a los privilegios mínimos necesarios.

5.30.3. Implementación: gestión de defectos

Para la gestión de defectos, se emplearon varias herramientas que permitieron un enfoque integral en la detección y corrección de vulnerabilidades. Se utilizaron *SonarQube* para el análisis *SAST (Static Application Security Testing)* y *AppSweep* para el análisis *DAST (Dynamic Application*

Security Testing), lo que permitió la detección continua de problemas de seguridad en el código fuente.

Adicionalmente, se integró *MobSF (Mobile Security Framework)* para el análisis estático de aplicaciones móviles.

5.30.4. Verificación: pruebas de seguridad

Se estableció un proceso de verificación continua, que incluyó la ejecución de pruebas unitarias e integrales para garantizar el correcto funcionamiento del código. Las auditorías periódicas permitieron revisar la seguridad de los entornos de despliegue y la implementación de configuraciones seguras.

5.31. Fase 4: evaluaciones de seguridad y mejora continua

5.31.1. Capacitación del equipo en *clean code* y *OWASP Top* 10

Como parte del esfuerzo para crear una cultura de seguridad y mejorar la calidad del desarrollo, se implementó un programa de capacitación semanal enfocado en las mejores prácticas de codificación y seguridad. Esta capacitación abarcó dos temas principales: *clean code* y los *OWASP Top 10*, los cuales fueron introducidos de manera gradual para asegurar una asimilación efectiva por parte del equipo.

- **Principios de *clean code***: la primera fase de la capacitación se centró en los principios de *clean code*, abordando aspectos clave como la simplicidad y claridad del código, con el objetivo de promover un desarrollo más estructurado y fácil de mantener. Se discutieron conceptos como:
  - Mantener el código lo más simple posible sin sacrificar la funcionalidad.
  - Uso adecuado de comentarios en áreas críticas del código para clarificar la funcionalidad y el propósito de los bloques de código.
  - Nombres descriptivos para variables, funciones y clases para mejorar la legibilidad.
  - Evitar duplicaciones y asegurar que cada fragmento de código tenga una sola responsabilidad.

  Estos principios fueron presentados durante varias sesiones semanales, permitiendo que el equipo los asimilara y aplicara de manera gradual. La introducción progresiva de estos conceptos no solo permitió reforzar su importancia en la práctica diaria, sino que también promovió una cultura de calidad en el desarrollo, alineada con los estándares de *OWASP SAMM* en términos de implementación y gobernanza.
- **Capacitación en *OWASP Top 10***: además de los principios de *clean code*, se dedicaron sesiones a la revisión de los *OWASP Top 10*, un conjunto de las vulnerabilidades de seguridad más comunes en aplicaciones *web*. Cada semana se presentó uno de los riesgos enumerados en *OWASP*, junto con ejemplos específicos proporcionados por la organización. Los temas abordados incluyeron:
  - Descripción de cada vulnerabilidad, su impacto en el sistema y ejemplos reales.
  - Buenas prácticas para evitar estos riesgos, adaptadas a los módulos del proyecto en desarrollo.
  - Casos prácticos para ilustrar cómo una vulnerabilidad podría comprometer la seguridad de la aplicación si no se gestiona correctamente.

  Esta capacitación semanal permitió al equipo absorber los conceptos clave sin sobrecargarlo con demasiada información a la vez. Esta metodología aseguró una comprensión sólida de los riesgos y cómo mitigarlos en sus tareas cotidianas, promoviendo una cultura de seguridad proactiva y alineada con los objetivos específicos del proyecto.

5.31.2. Reuniones semanales de seguimiento y planificación

Con el objetivo de asegurar el progreso continuo y la calidad del proyecto, se establecieron reuniones semanales de seguimiento y planificación, las cuales se llevaron a cabo cada viernes. Durante estas reuniones, se abordaron varios aspectos clave para el desarrollo seguro de la aplicación:

- **Revisión de avances y dificultades**: cada miembro del equipo compartía los avances alcanzados durante la semana, identificando logros, desafíos y obstáculos que requerían apoyo adicional. Esta práctica fomentó un ambiente colaborativo donde se discutían soluciones y se distribuían tareas de acuerdo a las necesidades actuales del proyecto.
- **Planificación de tareas y tiempos**: en función de los puntos tratados, se establecían metas semanales con plazos definidos, lo cual facilitaba la organización del trabajo y aseguraba que todos los integrantes tuvieran claridad sobre los próximos pasos y sus responsabilidades específicas.
- **Actualización de estado de la seguridad y calidad del código**: como líder del proyecto, presentaba un resumen del estado general del código, incluyendo vulnerabilidades detectadas y aspectos de cobertura de pruebas. También se detallaban áreas que necesitaban atención en términos de seguridad y buenas prácticas, destacando los puntos críticos para garantizar el cumplimiento de los estándares establecidos en *OWASP SAMM*.
- **Retroalimentación y resolución de dudas**: estas sesiones se aprovecharon para aclarar dudas del equipo sobre las recomendaciones y configuraciones de seguridad necesarias. Se brindaba retroalimentación específica sobre las vulnerabilidades detectadas y la cobertura, proporcionando una guía para resolver los problemas detectados y mejorar las prácticas de codificación.

Las reuniones semanales no solo fomentaron una cultura de seguridad proactiva, sino que también facilitaron la alineación continua del equipo con los objetivos del proyecto. Esta práctica contribuyó directamente al área de gobernanza dentro de *OWASP SAMM*, asegurando una comunicación clara y una gestión eficiente de la seguridad en cada fase del desarrollo.

5.31.3. Prueba de *phishing*

Como parte de la evaluación inicial, se llevó a cabo una prueba de *phishing* para medir la capacidad del equipo en la identificación de correos electrónicos maliciosos. La prueba mostró un desempeño inicial con un promedio de 4.5 respuestas correctas, indicando la necesidad de mejorar en esta área. Durante esta etapa, se organizaron sesiones de capacitación específicas para el equipo, enfocadas en identificar señales comunes de *phishing* y reforzar prácticas de seguridad en el manejo del correo electrónico.

La razón para incluir esta prueba de *phishing* es que, dentro de las prácticas de seguridad, no solo me enfoco en la protección de la aplicación y el desarrollo, sino también en la seguridad del propio equipo. En el contexto de ciberseguridad, los miembros del equipo pueden ser un punto de vulnerabilidad, ya que existe la posibilidad de que caigan en un ataque de *phishing*. Esto podría resultar en la divulgación de credenciales sensibles, como las necesarias para acceder a *AWS* u otros servicios críticos. La capacitación y las pruebas de este tipo ayudan a reducir el riesgo de accesos no autorizados derivados de errores humanos y fortalecen la postura de seguridad del equipo en general.

Después de las sesiones de capacitación, se llevó a cabo una segunda prueba el 14 de septiembre, en la que se observó una mejora significativa con un promedio de 8.25 respuestas correctas. Esto reflejó el impacto positivo de las capacitaciones, ya que el equipo demostró una mayor habilidad para reconocer correos electrónicos maliciosos, mejorando la postura de seguridad general del proyecto.

Finalmente, se realizó una evaluación sorpresa el 19 de octubre para asegurar que el equipo aún recordaba los consejos y técnicas impartidos en las sesiones de capacitación para diferenciar entre correos electrónicos legítimos y maliciosos. En esta evaluación, el equipo obtuvo un promedio de 9 respuestas correctas, lo cual reflejó que las capacitaciones y los tips compartidos fueron efectivamente retenidos y aplicados.

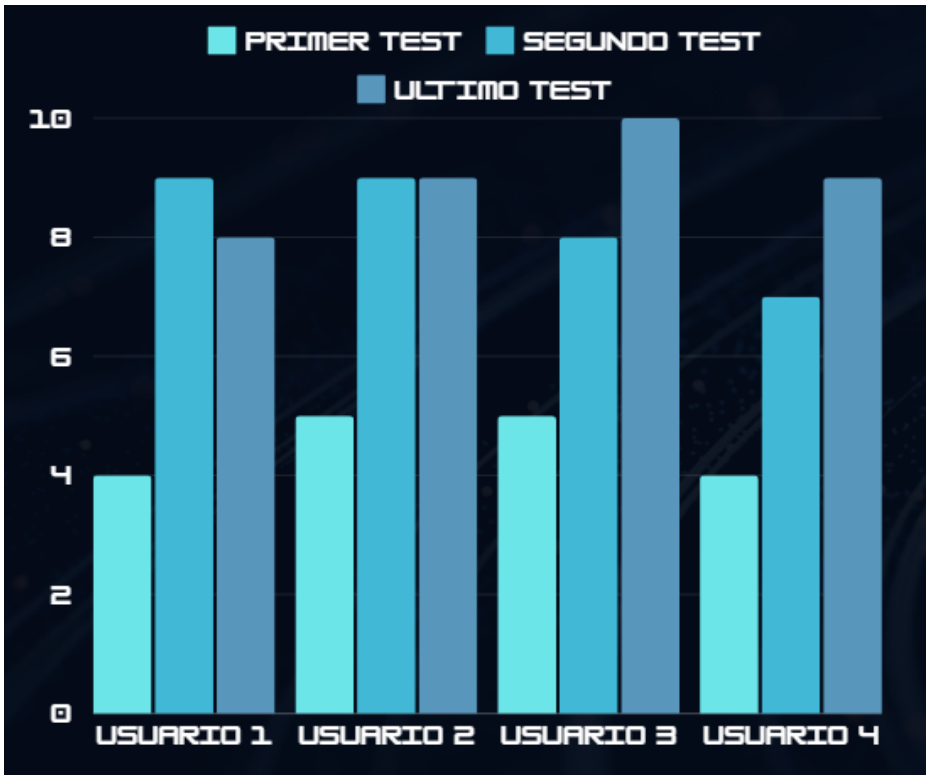


Figura 5.29. Resultados de phishing del equipo.

5.31.4. Análisis estático del código

Se llevaron a cabo análisis estáticos en diferentes momentos clave, comenzando el 5 de agosto, lo que permitió detectar problemas iniciales de mantenibilidad y seguridad. Tras varias rondas de corrección y reescaneos, el código mostró mejoras significativas en su calidad. Durante el proyecto, se realizaron capacitaciones regulares para asegurar que el equipo aplicara las mejores prácticas de desarrollo seguro, mejorando continuamente la calidad del código y asegurando la implementación correcta de las recomendaciones derivadas de los análisis.

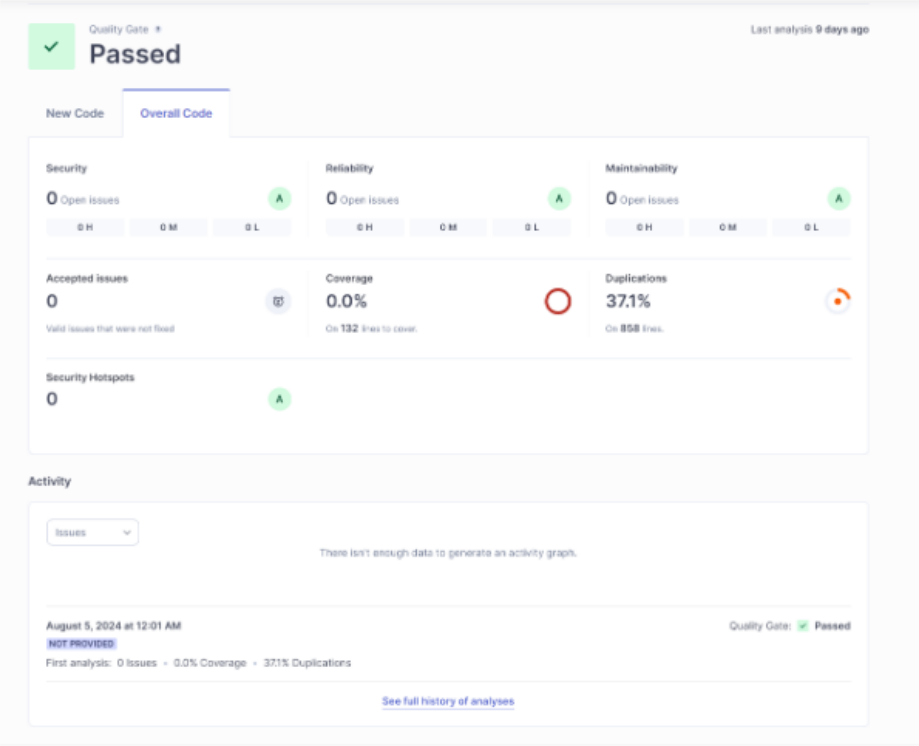


Figura 5.30. Resultado de *scan* del 14 de agosto.

En el análisis del 22 de agosto, se identificó que *SonarQube* requería un porcentaje de cobertura de pruebas mayor al 80 %. Para cumplir con este criterio, se configuró el proyecto para excluir ciertas carpetas que no contribuían a la cobertura relevante. La revisión constante de los resultados permitió ajustar las estrategias de prueba y mejorar la precisión de los informes.

5.31.5. Migración a *SonarCloud* y resultados de cobertura

Para mejorar la automatización del análisis estático, se migró el proceso a *SonarCloud*, lo que permitió integrar la plataforma con *GitHub* y utilizar un dominio público para los análisis. Una de las principales razones de esta migración fue que *SonarCloud* proporcionaba un dominio público, lo cual facilitó la automatización de los escaneos mediante *CI/CD* utilizando *GitHub Actions*. Esto permitió configurar flujos de trabajo automáticos para que los análisis de calidad del código se ejecutaran cada vez que se realizaban cambios en la rama principal del proyecto.

Este cambio no solo mejoró la visibilidad de los resultados del análisis, sino que también facilitó una mayor colaboración en el equipo, ya que todos los desarrolladores podían ver los resultados del análisis y recibir capacitaciones sobre cómo interpretar los hallazgos y actuar sobre ellos.

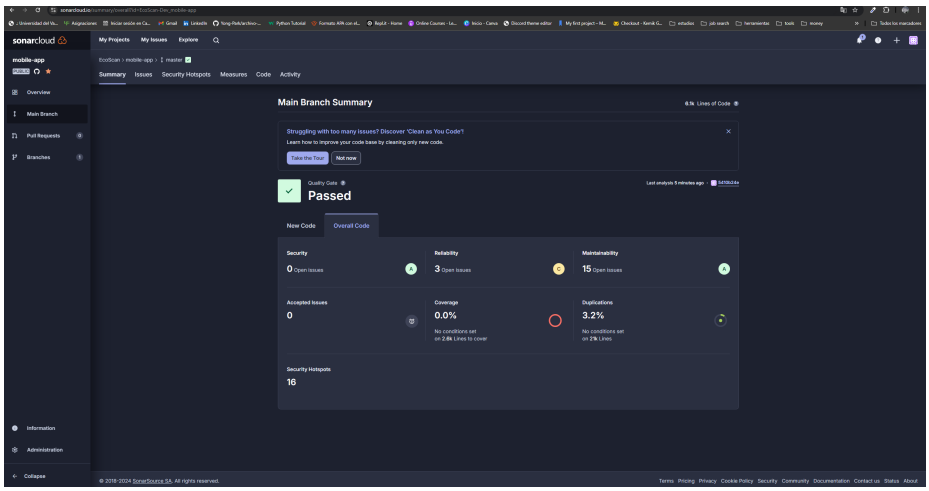


Figura 5.31. Primer escaneo en *SonarCloud*.

Para evitar que se escanearan los archivos de pruebas unitarias y aparecieran como vulnerabilidades, se configuró *SonarCloud* para excluir estas carpetas del análisis. Además, algunos archivos se excluyeron de la cobertura de código, ya que su lógica no requería cobertura, decisión tomada en consulta con cada especialista del área. Esta configuración específica se realizó dentro de los ajustes de *SonarCloud*, como se muestra en la imagen a continuación:

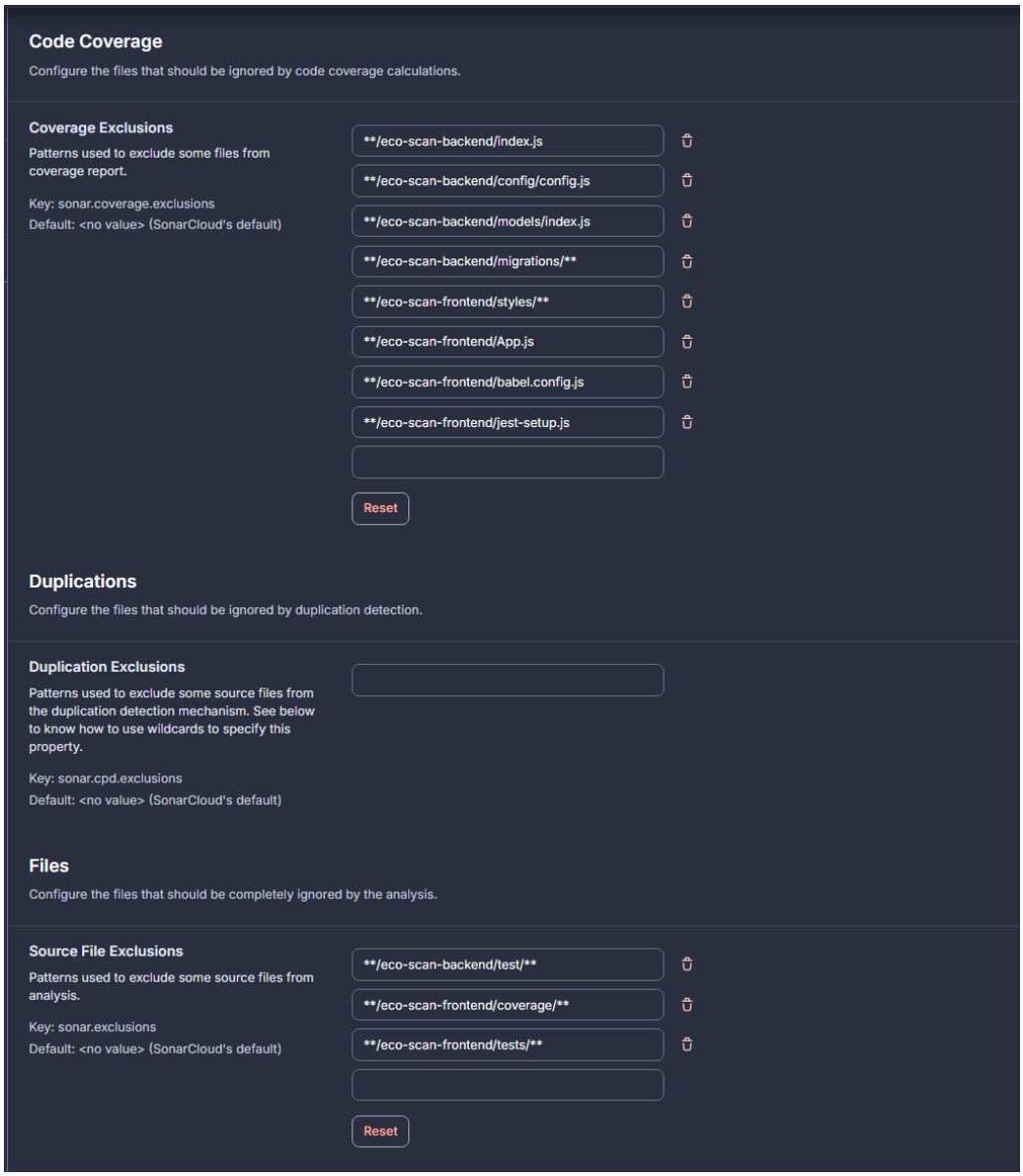


Figura 5.32. Configuración de exclusiones en *SonarCloud*.

Además, se creó un *quality check* manual en *SonarCloud* para asegurar el cumplimiento de los estándares definidos para el proyecto. La mayoría de los criterios de calidad proporcionados por *SonarCloud* fueron mantenidos, y se añadieron dos requisitos adicionales: primero, que la cobertura del código completo (no solo del nuevo) supere el 80 %; y segundo, que no existan *security hotspots* en ninguna parte del código, ya que es esencial identificar y resolver estos puntos críticos de seguridad lo más pronto posible.

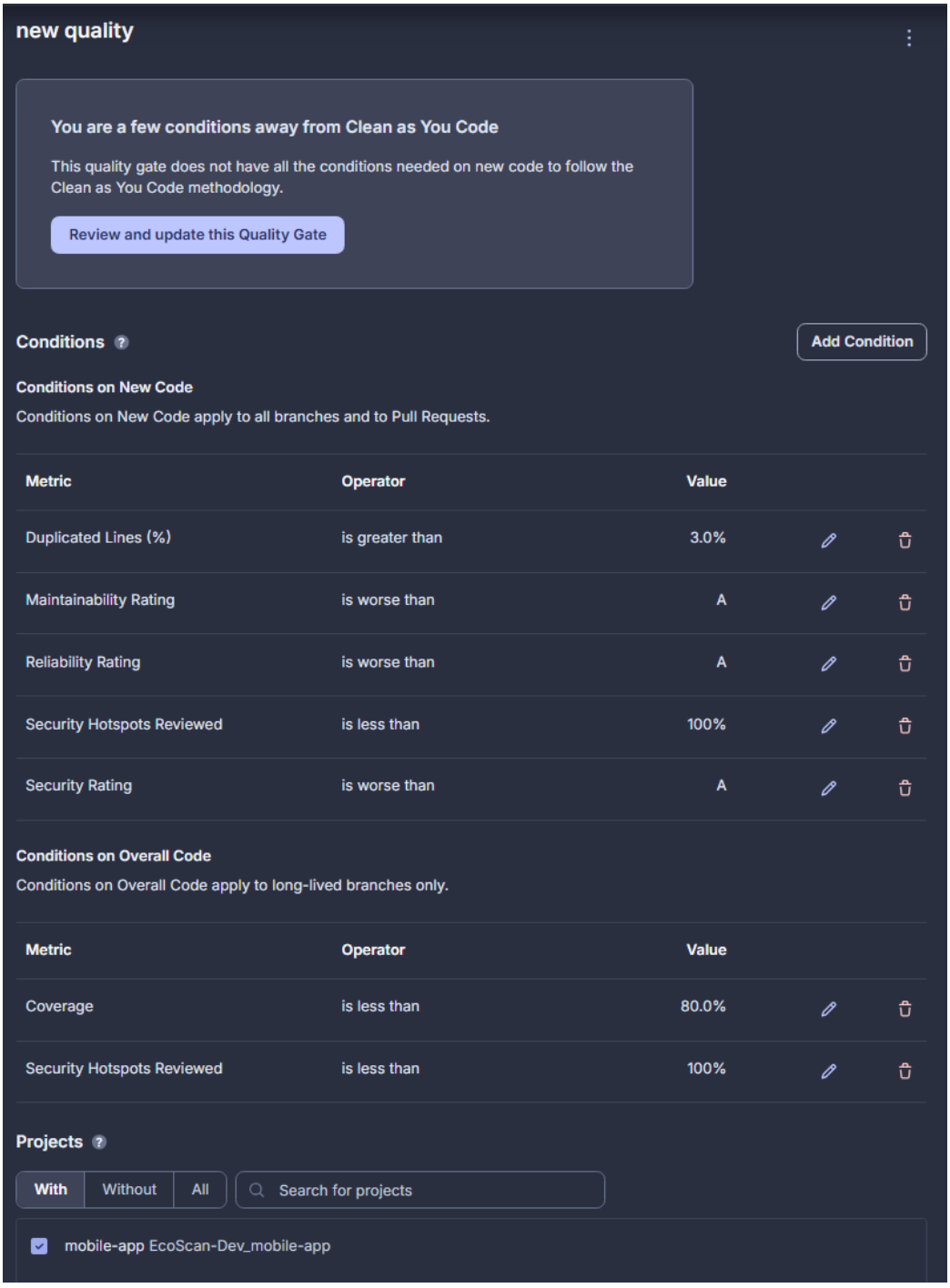


Figura 5.33. Configuración de calidad en *SonarCloud*.

Después de ajustar la configuración, la cobertura del *backend* alcanzó un 82.3 %, mientras que el *frontend* logró un 80.4 %, resultando en una cobertura total del 82 % del proyecto. Este ajuste permitió reflejar de manera más precisa la calidad y el alcance de las pruebas realizadas en el código relevante, excluyendo elementos que no aportan al funcionamiento directo de la aplicación.

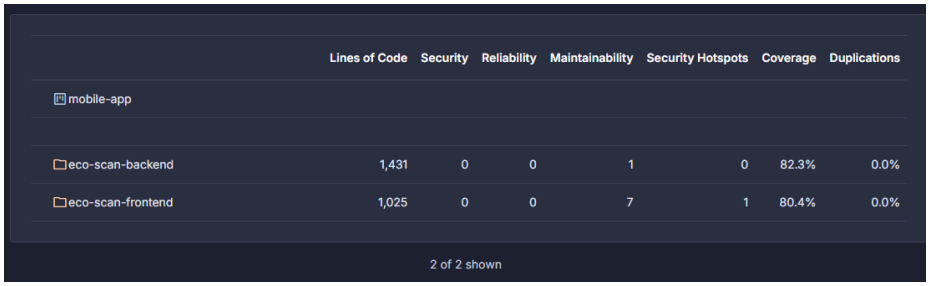


Figura 5.34. Cobertura del código en *SonarCloud*.

Figura 5.35. Grafica de cobertura en *sonarCloude*.

A lo largo de esta fase, se continuaron realizando capacitaciones periódicas para asegurar que el equipo adoptara un enfoque proactivo hacia la mejora continua y la resolución de problemas, garantizando que se mantuviera un ciclo de desarrollo seguro y de alta calidad.

5.31.6. Configuración de *CI/CD* en *SonarCloud*

Para implementar el proceso de *CI/CD* en *SonarCloud* junto con *GitHub*, fue necesario configurar el archivo `sonar-project.properties` para definir parámetros específicos de análisis y asegurar la sincronización entre ambas plataformas. En este archivo se configuraron aspectos clave como el *`projectKey`*, *`organization`*, y las rutas de cobertura. La siguiente imagen muestra la configuración final realizada en el archivo.

```
1   sonar.projectKey=EcoScan-Dev_mobile-app
2   sonar.organization=ecoscan-dev
3
4   # This is the name and version displayed in the SonarCloud UI.
5   sonar.projectName=mobile-app
6   #sonar.projectVersion=1.0
7
8   # Path is relative to the sonar-project.properties file. Replace "\" by "/" on Windows.
9   sonar.sources=.
10
11  # Path to the lcov coverage report
12  sonar.javascript.lcov.reportPaths=eco-scan-backend/test/reports/coverage/lcov.info,eco-scan-frontend/coverage/lcov.info
```

Figura 5.36. Configuración de *SonarCloud*.

5.31.7. Revisión de seguridad: detección y mitigación de vulnerabilidades

Como parte del compromiso con la seguridad, se monitorizó constantemente la cantidad de vulnerabilidades detectadas a lo largo del tiempo, tanto en *SonarQube* como en *SonarCloud*. Desde los primeros análisis en *SonarQube* hasta los escaneos automatizados en *SonarCloud*, la cantidad de vulnerabilidades ha disminuido progresivamente, lo cual demuestra la implementación efectiva de las buenas prácticas de seguridad y la gestión de defectos, en línea con el marco *SAMM*.

Figura 5.37. Gráfica de vulnerabilidades detectadas en *SonarQube*.

Figura 5.38. Gráfica de vulnerabilidades detectadas en *SonarCloud*.

Estas gráficas muestran que las vulnerabilidades detectadas en análisis iniciales fueron efectivamente corregidas, y que no reaparecieron los mismos problemas. Esto valida el éxito en la implementación de la gestión de defectos de *SAMM*, asegurando una mejora continua en la seguridad del código a lo largo del proyecto.

5.31.8. Evaluación de seguridad en modelos de *IA* y *chatbot*

Se revisaron los modelos de *IA*, identificando algunos falsos positivos en el uso de bibliotecas y funciones. La revisión confirmó que no había problemas críticos, lo cual fue validado mediante un escaneo sin alertas.

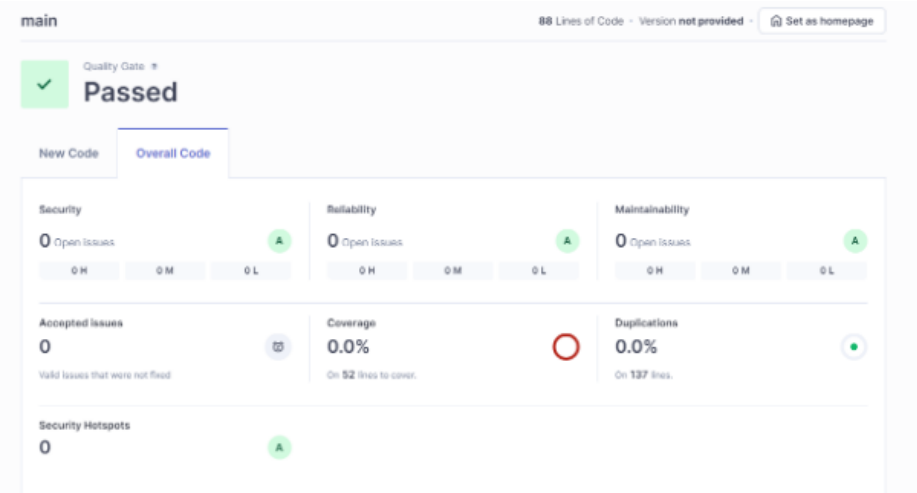


Figura 5.39. Resultado del escaneo del 20 de octubre para el segundo modelo.

Como parte de esta fase, se incluyó el código del modelo de *chatbot* desarrollado por el compañero Pedro Pablo. En el escaneo inicial del código del modelo a utilizar, no se detectaron vulnerabilidades en *SonarCloud*. Debido a que este código corresponde a un modelo de *IA*, no se requiere cobertura de pruebas; sin embargo, se realizó el análisis debido al uso de *endpoints* y *API*, cuya correcta implementación era necesaria para asegurar la seguridad y efectividad de la comunicación con servicios externos.

A continuación, se realizó un primer escaneo en *SonarCloud* específicamente para el código del *chatbot*. Durante esta inspección inicial, se identificó un error mínimo junto con una vulnerabilidad de seguridad menor que necesitaba ser atendida.

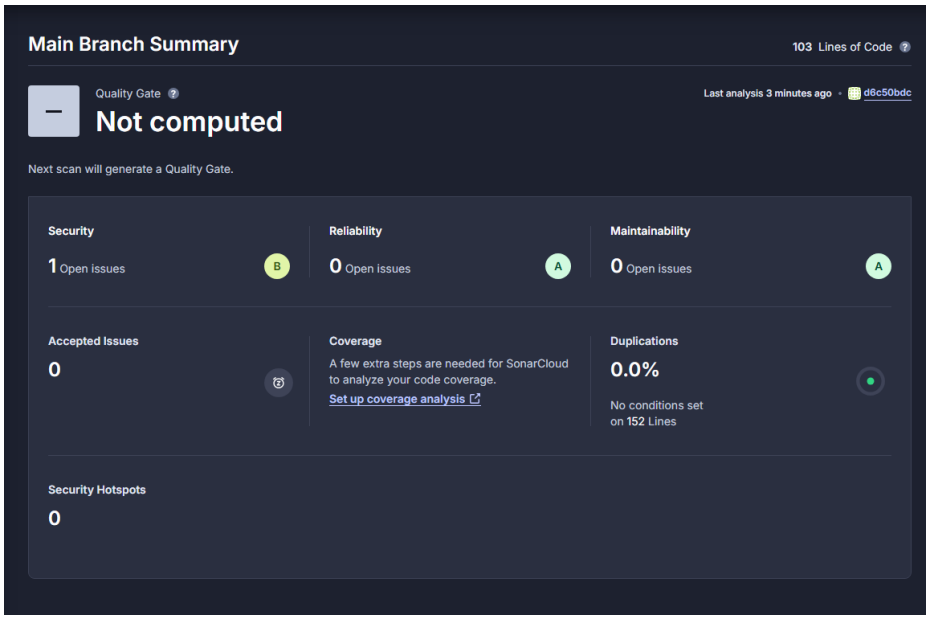


Figura 5.40. Primer escaneo del *chatbot* en *SonarCloud*.

Pedro Pablo procedió a realizar los ajustes necesarios en el código del *chatbot* para corregir la vulnerabilidad detectada. Como se observa en el siguiente escaneo, tras los cambios aplicados, el sistema no reportó ninguna vulnerabilidad adicional, asegurando así que el código cumpliera con los estándares de seguridad y calidad establecidos para el proyecto. Dado que este código corresponde a la lógica de un modelo, no se requirió cobertura adicional.

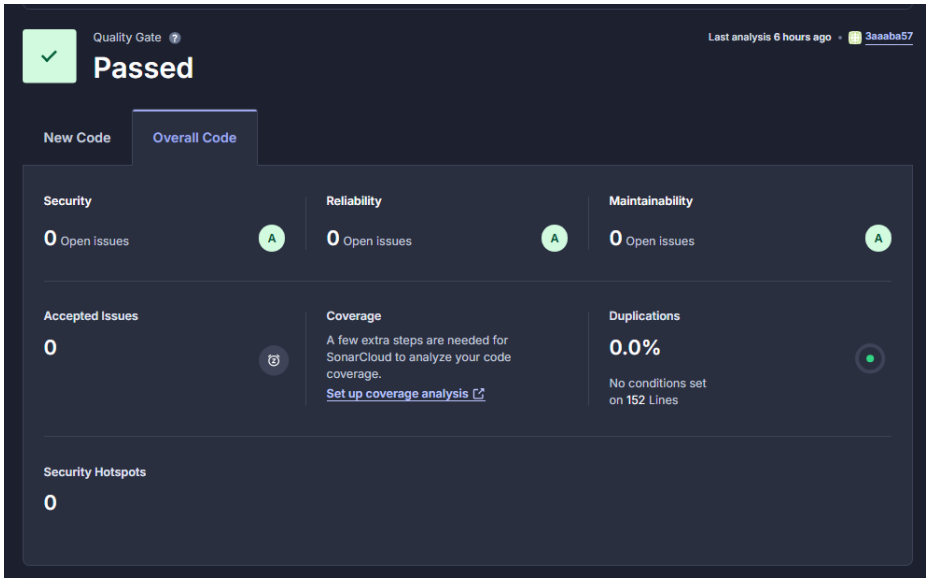


Figura 5.41. Último escaneo del *chatbot* en *SonarCloud*.

Con esta última revisión y sin vulnerabilidades reportadas, se dio por completada la fase de verificación de seguridad del código del *chatbot*, finalizando así la contribución del compañero Pedro Pablo en esta parte del proyecto.

5.32. Fase 5: Escaneo y resultados de la aplicación móvil

Con el objetivo de asegurar la seguridad de la aplicación móvil, se realizaron escaneos tanto estáticos como dinámicos utilizando *MobSF* y *AppSweep*, respectivamente. A continuación, se describen los procedimientos y resultados obtenidos en esta fase.

5.32.1. Escaneo estático con *MobSF*

Para el análisis estático de la aplicación móvil, se utilizó ***Mobile Security Framework (MobSF)***. Inicialmente, se intentó realizar el escaneo del archivo `.aab` (*Android App Bundle*) mediante la instancia *web* de *MobSF*. Sin embargo, se encontró un problema recurrente de pérdida de conexión, lo cual interrumpía el proceso de carga y análisis del archivo.

Para solventar esta limitación, se optó por descargar el repositorio de *MobSF* y ejecutarlo localmente. Los pasos seguidos para la configuración local fueron los siguientes:

- `setup.bat`: este *script* verifica los requisitos previos y, en caso de que falten dependencias, solicita su instalación. Una vez cumplidos los requisitos, procede a instalar todas las librerías necesarias para el funcionamiento de *MobSF*.
- `run.bat`: una vez que las dependencias están instaladas, este comando inicia *MobSF* para que se ejecute localmente, permitiendo el análisis sin interrupciones de conexión.

En este caso, para ejecutar correctamente `setup.bat`, fue necesario contar con las siguientes herramientas preinstaladas en el sistema: ***Git***, ***Python 3.8/3.9*** y ***OpenSSL***. Una vez que `setup.bat` completó la instalación de las librerías necesarias, al ejecutar `run.bat`, *MobSF* se activó localmente, siendo accesible a través de `localhost:8000`. Las credenciales de acceso configuradas de forma predeterminada eran ***mobsf*** tanto para el usuario como para la contraseña.

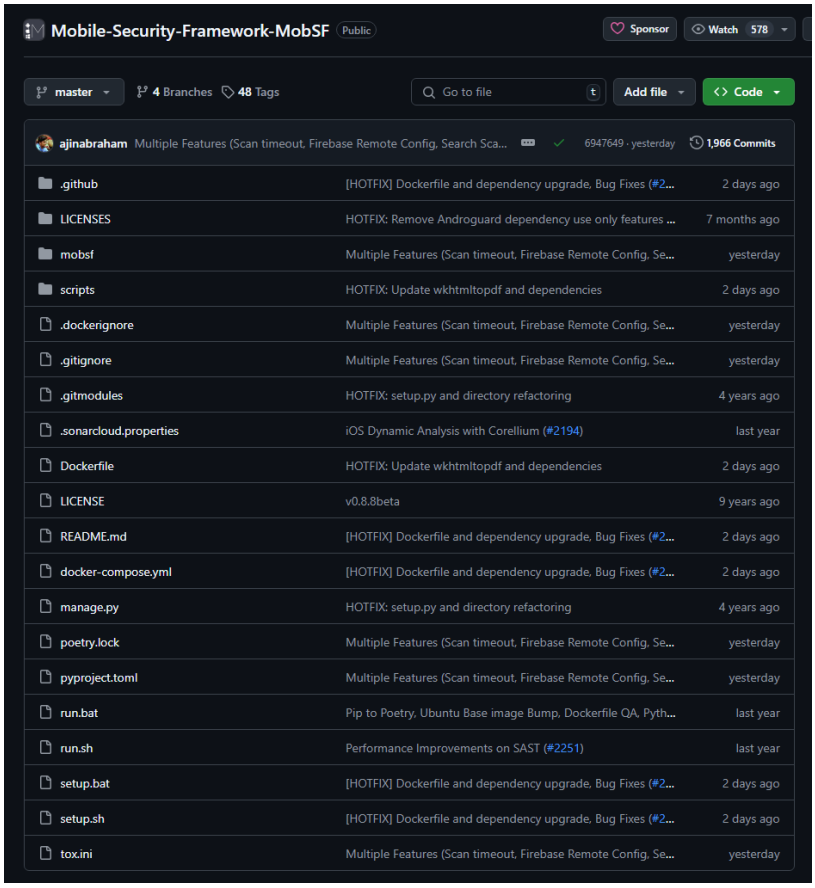


Figura 5.42. Repositorio de *MobSF*.

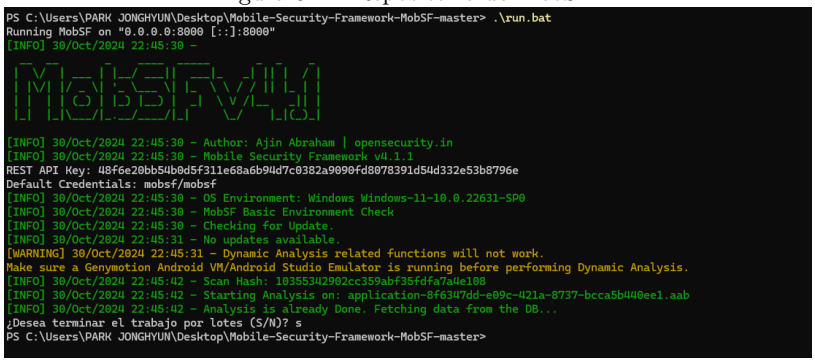


Figura 5.43. *MobSF* ejecutándose localmente.

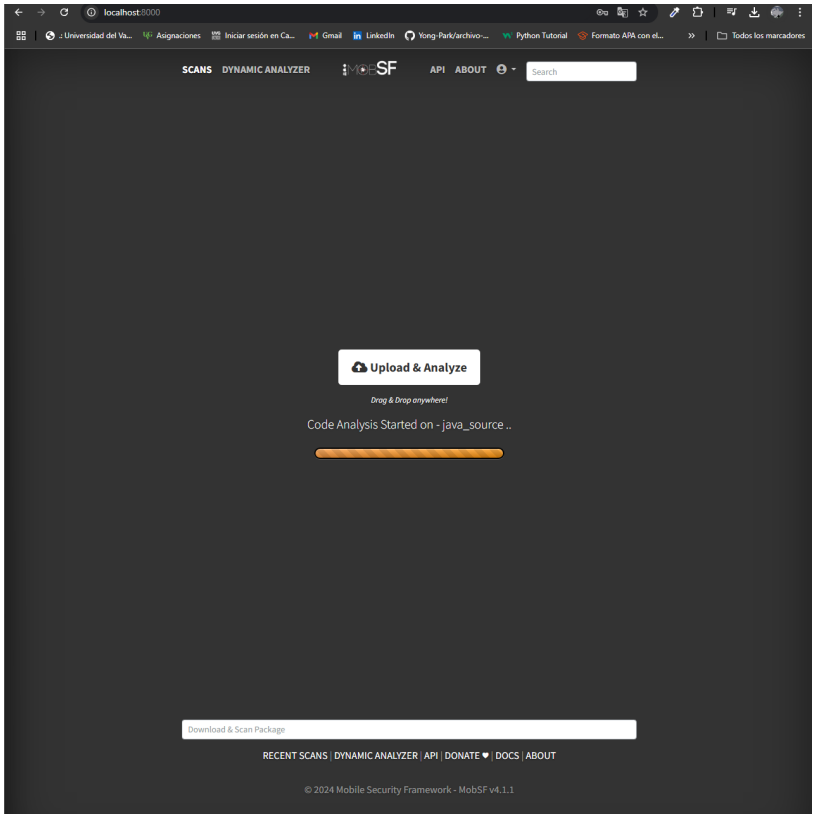


Figura 5.44. Interfaz gráfica de *MobSF*

Después de la configuración y ejecución de *MobSF* en un entorno local, se procedió a cargar el archivo `.aab` de la aplicación móvil para su análisis. Los resultados iniciales del escaneo estático revelaron diversos aspectos de seguridad en el código de la aplicación, que fueron posteriormente evaluados y optimizados según las recomendaciones de *MobSF*.

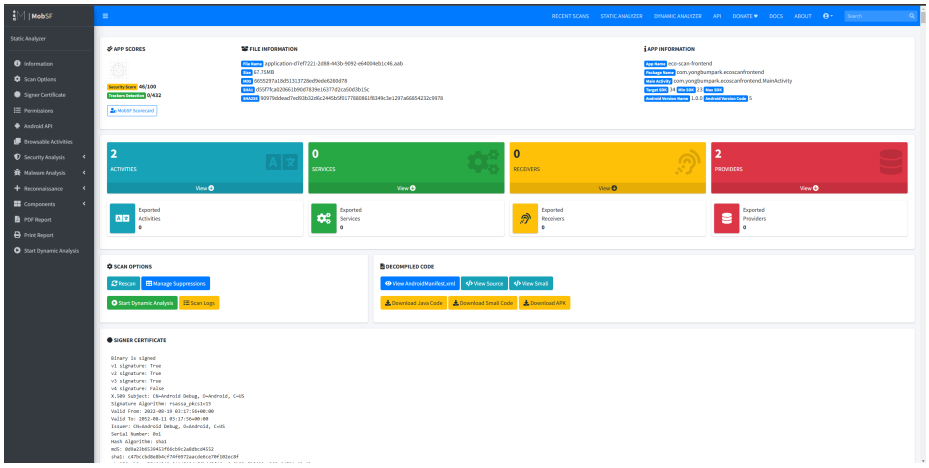


Figura 5.45. Resultado del escaneo inicial en *MobSF*.

5.32.2. Escaneo dinámico con *AppSweep*

Para el análisis dinámico de la aplicación móvil, se seleccionó ***AppSweep*** debido a su enfoque específico en la detección de vulnerabilidades en aplicaciones móviles en tiempo de ejecución. Es-

te escaneo permitió identificar problemas relacionados con permisos, configuraciones deficientes y almacenamiento inseguro.

El análisis *DAST* en *AppSweep* fue exitoso y se ejecutó sin problemas, validando aspectos críticos de seguridad durante la ejecución de la aplicación móvil.

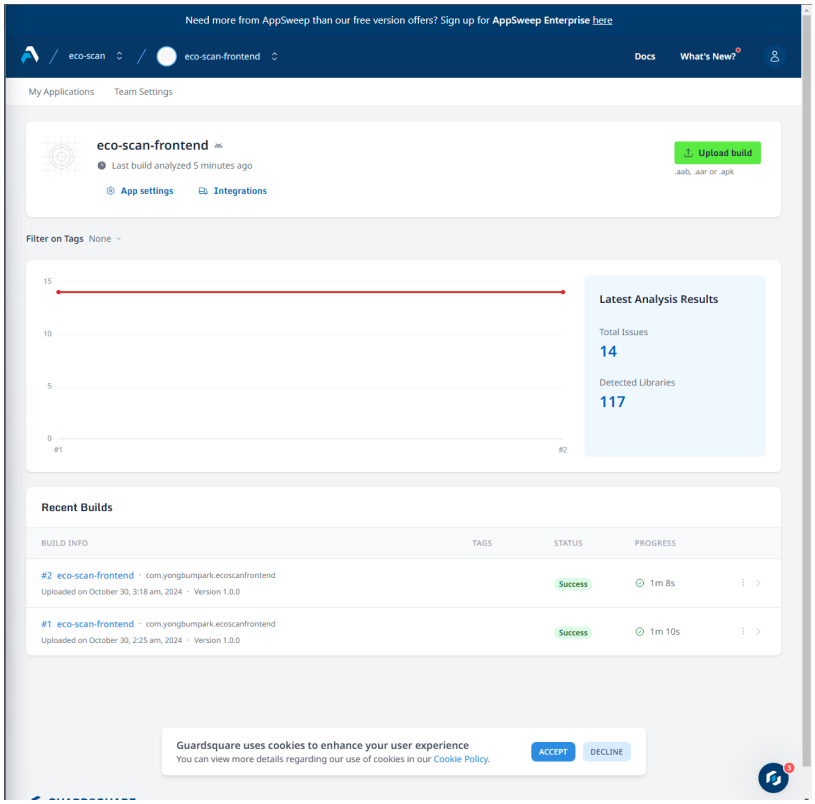


Figura 5.46. Resultado del escaneo en *AppSweep*.

5.33. Fase 6: revisión y optimización de permisos en *AWS IAM Identity Center*

Como parte de las medidas de seguridad y buenas prácticas en la gestión de acceso, se llevó a cabo una revisión de los permisos configurados en el *AWS IAM Identity Center*. Este proceso fue esencial para asegurar que los accesos a la infraestructura de *AWS* estuvieran debidamente controlados y alineados con los principios de privilegios mínimos.

5.33.1. Revisión de grupos y permisos

La revisión comenzó con una inspección detallada de los grupos predeterminados generados por *AWS* en el *IAM Identity Center*. Estos grupos contienen configuraciones estándar proporcionadas por *AWS*, las cuales fueron evaluadas para determinar su adecuación y relevancia en el contexto del proyecto.

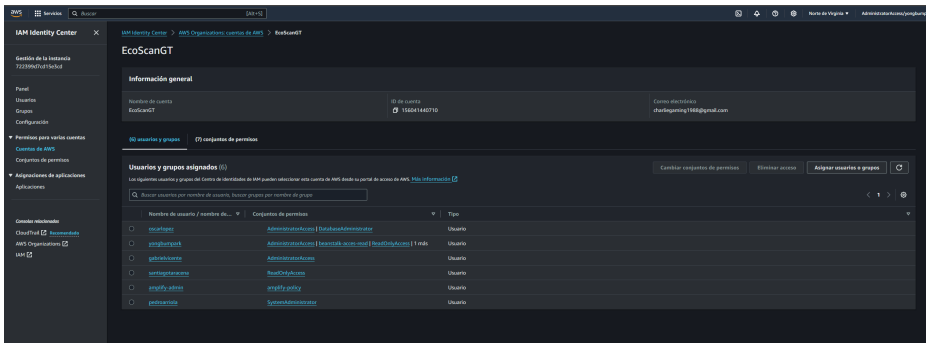


Figura 5.47. *AWS IAM Identity Center*: cuentas y grupos predeterminados.

Cada grupo de permisos fue revisado cuidadosamente, y se documentaron las asignaciones de permisos para cada usuario. Este análisis permitió identificar permisos que podrían no ser necesarios para el rol asignado, lo cual podría representar un riesgo potencial de seguridad si no se gestionaba adecuadamente.

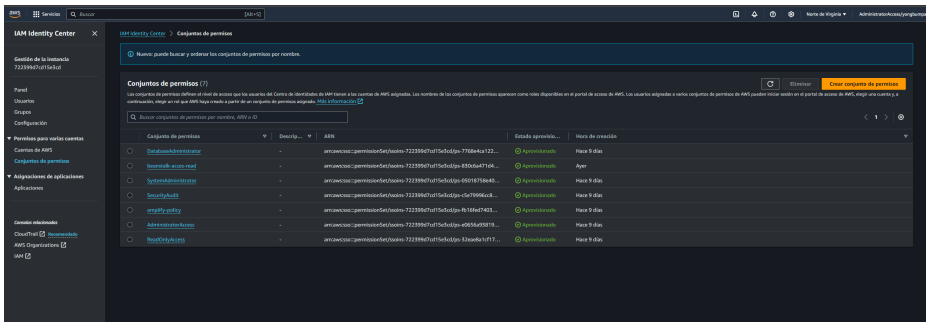


Figura 5.48. *AWS IAM Identity Center*: permisos de usuario revisados.

5.33.2. Proceso de validación y ajuste de permisos

Durante la revisión, cuando se identificaba un permiso que no era crítico o que representaba un posible exceso de privilegios, se realizaba una consulta directa con el encargado de la infraestructura, Gabriel. Las decisiones sobre cada permiso se tomaban en colaboración, considerando el impacto en las operaciones y la seguridad general del sistema. Esto incluyó las siguientes acciones:

- **Eliminación de permisos no necesarios:** en casos donde se determinó que ciertos permisos eran superfluos, se procedió a su eliminación para fortalecer la seguridad.
- **Restricción de permisos excesivos:** algunos permisos fueron restringidos aún más mediante la aplicación de políticas de acceso más limitadas, alineadas con el principio de privilegios mínimos.
- **Mantenimiento de permisos esenciales:** los permisos críticos para la operación se mantuvieron, pero se documentaron adecuadamente para futuras auditorías y revisiones.

Este proceso contribuyó a la seguridad de la infraestructura al garantizar que solo se mantuvieran los permisos absolutamente necesarios y que cualquier exceso de privilegios fuera gestionado de manera proactiva. La colaboración continua con Gabriel aseguró que las decisiones fueran bien informadas y equilibraran la seguridad con la funcionalidad requerida.

5.34. Metodología de inteligencia artificial

5.35. Selección de tecnologías

En este proyecto, se han seleccionado diversas tecnologías y *frameworks* para implementar tanto el modelo de clasificación de imágenes mediante redes neuronales como el sistema de chat inteligente. La selección de estas herramientas se ha realizado con base en criterios de rendimiento, soporte en la comunidad, flexibilidad y facilidad de integración en una aplicación móvil destinada a promover prácticas sostenibles. A continuación, se explican las tecnologías seleccionadas y las razones para su elección.

5.35.1. Plataforma de entrenamiento y desarrollo: *Google Colab*

Para el entrenamiento y desarrollo del modelo de clasificación de imágenes, se optó por utilizar ***Google Colab*** en lugar de servicios de computación en la nube de pago o máquinas locales. La decisión de emplear *Google Colab* se basa en una combinación de criterios de costo, disponibilidad de recursos de cómputo y compatibilidad con la arquitectura utilizada en este proyecto.

Ventajas de *Google Colab* en el contexto del proyecto

- **Acceso a recursos de cómputo potentes**: *Google Colab* proporciona acceso gratuito a *GPU* (como las *NVIDIA Tesla T4* o *K80*) y ocasionalmente a *TPU*, recursos fundamentales para entrenar modelos de redes neuronales profundas de manera eficiente. Esta capacidad de cómputo es clave para realizar entrenamientos en menos tiempo, algo que no sería viable en una máquina local sin tarjeta gráfica dedicada.
- **Optimización de costo-beneficio**: a diferencia de servicios de nube dedicados como *AWS, Azure* o *Google Cloud Platform*, el uso de *Google Colab* no incurre en costos significativos para la mayoría de los recursos necesarios en este proyecto. Esto es especialmente relevante dada la necesidad de optimizar el presupuesto, ya que los servicios de nube suelen implicar tarifas elevadas para instancias con *GPU* o *TPU*.
- **Compatibilidad y facilidad de uso**: *Google Colab* está basado en *Jupyter Notebooks*, lo que facilita la iteración, prueba y desarrollo de código en un entorno interactivo. Esta plataforma permite la ejecución de código en *Python*, simplificando la implementación y prueba de los modelos mediante una interfaz amigable y ampliamente utilizada en investigación y educación.
- **Limitaciones de *hardware* local**: el equipo de desarrollo es una *MacBook Pro* con arquitectura *M1*, que si bien es potente para tareas generales, presenta limitaciones para el entrenamiento de redes neuronales profundas. La arquitectura *ARM* de los procesadores *M1*, además de la falta de una tarjeta gráfica dedicada, hace que los tiempos de entrenamiento se prolonguen notablemente si se utiliza el *CPU*. Además, las librerías principales como *TensorFlow* y *PyTorch* aún tienen soporte limitado en *ARM* para tareas de entrenamiento en *GPU*, lo cual refuerza la elección de *Google Colab* como una solución más eficiente.

Desventajas de alternativas consideradas

- **Servicios de nube**: a pesar de ofrecer recursos de alta calidad y soporte avanzado, el costo de plataformas como *AWS* y *Google Cloud Platform* resulta prohibitivo para entrenamientos de larga duración y de alta demanda computacional en *GPU*. La opción de usar *Google Colab* permite reducir estos costos sin sacrificar el rendimiento.
- **Máquina local (*MacBook Pro* con *M1*)**: el uso de una máquina local con arquitectura *M1* no es viable para este proyecto, debido a las limitaciones en rendimiento y soporte de *GPU*. Los tiempos de entrenamiento en *CPU* serían sustancialmente más largos, lo cual afectaría el proceso iterativo de desarrollo y ajuste de los modelos.

En conclusión, la selección de *Google Colab* como plataforma de desarrollo y entrenamiento permite aprovechar recursos computacionales avanzados sin incurrir en altos costos. Esto garantiza una solución económica y accesible, capaz de gestionar los requisitos computacionales del modelo de clasificación de residuos desarrollado en este proyecto.

5.35.2. *Framework* de redes neuronales: *TensorFlow*

Para el desarrollo del modelo de clasificación de imágenes, se ha elegido ***TensorFlow*** como el *framework* de *Deep Learning* principal, descartando otras opciones como ***PyTorch***. La elección de *TensorFlow* se justifica en los siguientes aspectos [399]:

- **Rendimiento y eficiencia**: *TensorFlow* es ampliamente reconocido por su capacidad de realizar cálculos de alto rendimiento mediante el soporte de procesamiento en *GPU* y *TPU*, lo cual es esencial para entrenar redes neuronales convolucionales (*CNNs*) que requieren gran poder de cómputo.
- **Compatibilidad y portabilidad**: *TensorFlow* facilita la exportación de modelos a distintos formatos, como *TensorFlow* Lite, lo cual permite desplegar modelos en dispositivos móviles con eficiencia en el uso de recursos. Esta funcionalidad es crítica en el contexto de una aplicación móvil para la clasificación de residuos en tiempo real.
- **Soporte en la comunidad y documentación**: *TensorFlow* cuenta con una amplia comunidad de desarrolladores y un extenso soporte de documentación y recursos, lo cual facilita el desarrollo y la resolución de problemas, maximizando la eficiencia en el desarrollo.

Aunque *PyTorch* es una alternativa popular y altamente valorada en investigación, se optó por *TensorFlow* debido a su mejor soporte para despliegue en producción y optimización en dispositivos móviles, factores determinantes en este proyecto.

5.35.3. Manipulación y preprocesamiento de imágenes: *PILLOW*

Para el preprocesamiento y la manipulación de imágenes se ha seleccionado ***PILLOW*** (*Python Imaging Library*) [400]. Esta biblioteca permite realizar operaciones esenciales de preprocesamiento, tales como redimensionamiento, rotación, ajuste de brillo y normalización de las imágenes de entrada. Estas funciones son indispensables para preparar los datos que se ingresan al modelo de clasificación de residuos.

- **Facilidad de uso**: *PILLOW* es una herramienta sencilla y altamente compatible con otras bibliotecas de *Machine Learning* y *Deep Learning*, lo cual facilita su integración en el flujo de trabajo.
- **Optimización y velocidad**: la biblioteca está optimizada para el procesamiento rápido de imágenes, lo cual es ideal para su uso en aplicaciones que requieren análisis en tiempo real.

La elección de *PILLOW* sobre otras librerías como *OpenCV* responde a su simplicidad en la manipulación de imágenes en el contexto específico de este proyecto, donde no se requieren las capacidades avanzadas de visión por computadora que ofrece *OpenCV*.

5.35.4. Modelo de chat inteligente: *OpenAI API* con *GPT*-3.5 Turbo

Para el sistema de chat inteligente se ha seleccionado la ***API* de OpenAI**, específicamente el modelo ***GPT-3.5 Turbo*** [401]. Este modelo fue preferido sobre alternativas como **GPT-4** o modelos de *Hugging Face*, como *Llama 2* y *Llama 3*, por las siguientes razones:

- **Eficiencia en tiempo de respuesta**: *GPT-3.5 Turbo* presenta un balance entre precisión y velocidad, lo cual permite responder rápidamente a las consultas de los usuarios. Dado que la aplicación se orienta a proporcionar consejos e información de manera ágil, esta característica es fundamental.

- **Costo-beneficio**: *GPT-3.5 Turbo* tiene un menor costo de operación comparado con *GPT-4*, lo que resulta adecuado para el presupuesto del proyecto.
- **Fiabilidad y flexibilidad**: la *API* de *OpenAI* ofrece estabilidad en producción y se encuentra optimizada para el procesamiento de lenguaje natural en tareas de asistencia y generación de respuestas. La disponibilidad de *OpenAI* asegura un servicio continuo y confiable.

A pesar de que modelos como Llama 2 y 3 presentan capacidades avanzadas y son opciones de código abierto, la *API* de *OpenAI* se ha elegido por su rendimiento en producción, facilidad de implementación y menor demanda de infraestructura, lo cual resulta ideal para este proyecto.

5.35.5. *Framework* para *endpoints* de chat: *FastAPI*

Para la implementación de los *endpoints* del sistema de chat, se ha seleccionado ***FastAPI*** sobre otras opciones como ***Flask***, debido a sus ventajas en términos de rendimiento y facilidad de desarrollo [402].

- **Alto rendimiento**: *FastAPI* está optimizado para manejar múltiples peticiones concurrentes con una latencia mínima, lo cual es crítico para un *chatbot* que interactúa con varios usuarios en tiempo real.
- **Compatibilidad con asincronía**: a diferencia de *Flask, FastAPI* está diseñado para soportar operaciones asincrónicas, lo que permite gestionar eficientemente las consultas a la *API* de *OpenAI* y mejorar la capacidad de respuesta del sistema.
- **Facilidad de uso y documentación automática**: *FastAPI* genera automáticamente documentación de los *endpoints* utilizando *Swagger*, lo cual facilita las pruebas y la integración del sistema.

La decisión de elegir *FastAPI* sobre *Flask* se basa en su mejor rendimiento para aplicaciones de tiempo real y su compatibilidad con arquitecturas asincrónicas, lo cual contribuye a un sistema de chat eficiente y escalable.

5.35.6. Resumen de ventajas de las tecnologías seleccionadas

Las herramientas seleccionadas cumplen con los requisitos de eficiencia, escalabilidad y facilidad de integración en un entorno de producción. La combinación de *TensorFlow* [399], *PILLOW* [400], la *API* de *OpenAI* [401] y *FastAPI* [402] permite implementar una solución robusta y optimizada para la clasificación de residuos y la interacción mediante un *chatbot* inteligente en una aplicación móvil.

5.36. Descripción del *dataset* Utilizado

Para el entrenamiento y la validación del modelo de clasificación de residuos, se utilizó el *dataset Garbage Classification - 6 classes (775/class)*, disponible en *Kaggle* [403]. Este *dataset* globalizado, con imágenes de residuos provenientes de múltiples países, representa una amplia gama de contextos y condiciones ambientales, proporcionando así una base sólida para la generalización del modelo. El uso de un conjunto de datos internacional es particularmente relevante, ya que permite capturar las variaciones en las características visuales de los residuos que podrían encontrarse en distintas regiones, incluido Guatemala.

5.36.1. Características principales del *dataset*

El *dataset* consta de un total de 4650 imágenes, distribuidas uniformemente en seis categorías principales de residuos, con 775 imágenes por clase. Las características de cada clase se describen a continuación:

- ***Plastic***: compuesta de botellas, bolsas y envases de plástico. Los objetos de esta clase presentan bordes definidos y una textura translúcida o brillante que los distingue visualmente.
- ***Paper***: incluye papeles, cartones y otros productos fibrosos. Esta clase es caracterizada por superficies mate y bordes irregulares debido a la flexibilidad del material.
- ***Organic***: residuos biodegradables como restos de alimentos y vegetación. Los residuos orgánicos presentan una gran variedad de formas y colores naturales, con texturas menos uniformes y, a menudo, superficies rugosas.
- **Metal**: principalmente latas y piezas metálicas. La clase de metal es fácilmente reconocible por sus superficies reflectantes y a menudo está acompañada de signos de oxidación o desgaste.
- ***Glass***: incluye botellas, frascos y otros objetos de vidrio. El vidrio se caracteriza por su transparencia o translucidez, reflejos y bordes lisos.
- ***Battery***: residuos peligrosos que incluyen baterías de distintos tamaños y formas. Las baterías suelen tener colores y etiquetas distintivas, además de una superficie generalmente lisa y regular.

La distribución balanceada entre las clases asegura que el modelo aprenda de manera equitativa las características distintivas de cada tipo de residuo, favoreciendo una clasificación precisa y sin sesgo de clase.

5.36.2. Relevancia del *dataset* en el contexto Guatemalteco

A pesar de su carácter internacional, el *dataset* ha demostrado ser adecuado para el contexto guatemalteco tras realizar pruebas exitosas con imágenes de productos locales. Las similitudes en las características visuales de residuos comunes en Guatemala y en otros países sugieren que este conjunto de datos es representativo y adecuado para capturar patrones visuales relevantes en el contexto local. Además, la diversidad en las condiciones de iluminación y variabilidad de texturas y colores en el *dataset* refuerza su capacidad de generalización, lo cual es crítico para aplicaciones en el mundo real.

5.37. Ejemplos visuales de imágenes en el *dataset*

Para ilustrar la variabilidad y las características visuales de cada clase de residuos, la Figura 5.49 muestra ejemplos representativos de las seis categorías en el *dataset*. Estas imágenes destacan los patrones y texturas característicos que el modelo debe aprender para diferenciar entre tipos de residuos.

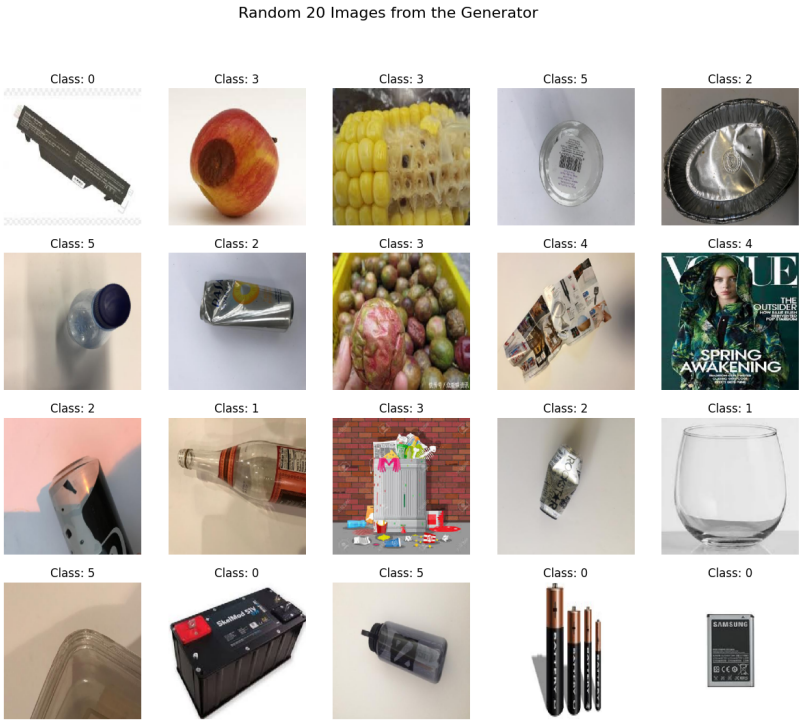


Figura 5.49. Ejemplos de imágenes de cada clase en el *dataset*: plástico, papel, orgánico, metal, vidrio y baterías

La diversidad en las imágenes, desde residuos orgánicos hasta materiales peligrosos como baterías, enfatiza la necesidad de un modelo robusto que pueda identificar con precisión diferentes materiales en diversas condiciones de captura. Esta variabilidad es clave para el éxito del modelo en un entorno de implementación en tiempo real.

5.38. Justificación del uso de Redes Neuronales Convolucionales (*CNN*) y *Transfer Learning*

Para el desarrollo del modelo de clasificación de residuos, se seleccionaron **Redes Neuronales Convolucionales (*CNN*)** debido a su reconocida eficacia en tareas de visión por computadora, particularmente en la clasificación de imágenes. Las *CNNs* han mostrado un alto rendimiento en la identificación de patrones visuales complejos, como formas, texturas y contornos, permitiendo distinguir entre diferentes materiales de residuos con precisión.

5.38.1. Ventajas del uso de *CNNs* en clasificación de imágenes

Las *CNNs* están estructuradas en capas convolucionales, que extraen características locales de una imagen, como bordes, formas y texturas, permitiendo que el modelo aprenda representaciones jerárquicas de las características de los residuos. Esto resulta particularmente útil en la clasificación de residuos, ya que distintos materiales presentan características visuales distintivas que pueden ser aprendidas y generalizadas por el modelo.

- **Extracción automática de características**: a diferencia de otros enfoques que requieren un diseño manual de características, las *CNNs* pueden aprender directamente de los datos las características relevantes para la tarea de clasificación.

- **Invarianza a transformaciones**: las capas convolucionales permiten que el modelo sea invariante a pequeñas transformaciones como rotaciones o cambios de escala, lo cual es útil para clasificar residuos en diferentes posiciones y condiciones.
- **Eficiencia computacional**: aunque las *CNNs* requieren un poder de procesamiento significativo, son más eficientes que otros modelos complejos en la clasificación de imágenes, lo cual es ideal para implementar un modelo en un entorno de producción.

5.38.2. *Transfer Learning* para mejorar la precisión del modelo

El ***Transfer Learning*** se utilizó en el proyecto como una estrategia para mejorar la precisión del modelo y reducir el tiempo de entrenamiento. Esta técnica implica utilizar una *CNN* preentrenada en un conjunto de datos de gran tamaño (como *ImageNet*) y ajustar sus capas finales para la tarea de clasificación de residuos. El uso de *Transfer Learning* ofrece varias ventajas:

- **Reducción del tiempo de entrenamiento**: al utilizar un modelo preentrenado, solo es necesario reentrenar las capas finales, reduciendo significativamente el tiempo de entrenamiento en comparación con un modelo entrenado desde cero.
- **Mejora en la generalización**: los modelos preentrenados ya han aprendido características visuales generales (como bordes y texturas) que son útiles para múltiples tareas de clasificación, lo cual mejora la capacidad de generalización del modelo en el contexto de residuos.
- **Optimización de recursos computacionales**: *Transfer Learning* permite obtener un modelo preciso utilizando un conjunto de datos relativamente pequeño, lo cual es beneficioso en este proyecto debido a la cantidad limitada de imágenes en el *dataset* de residuos.

5.39. Justificación del Uso de Redes Neuronales Convolucionales (*CNN*), *Transfer Learning* y *Fine-Tuning*

Para el desarrollo del modelo de clasificación de residuos, se seleccionaron **Redes Neuronales Convolucionales (*CNN*)** debido a su reconocida eficacia en tareas de visión por computadora, particularmente en la clasificación de imágenes. Las *CNNs* han mostrado un alto rendimiento en la identificación de patrones visuales complejos, como formas, texturas y contornos, permitiendo distinguir entre diferentes materiales de residuos con precisión.

5.39.1. Ventajas del Uso de *CNNs* en Clasificación de Imágenes

Las *CNNs* están estructuradas en capas convolucionales, que extraen características locales de una imagen, como bordes, formas y texturas, permitiendo que el modelo aprenda representaciones jerárquicas de las características de los residuos. Esto resulta particularmente útil en la clasificación de residuos, ya que distintos materiales presentan características visuales distintivas que pueden ser aprendidas y generalizadas por el modelo.

- **Extracción Automática de Características**: a diferencia de otros enfoques que requieren un diseño manual de características, las *CNNs* pueden aprender directamente de los datos las características relevantes para la tarea de clasificación.
- **Invarianza a Transformaciones**: las capas convolucionales permiten que el modelo sea invariante a pequeñas transformaciones como rotaciones o cambios de escala, lo cual es útil para clasificar residuos en diferentes posiciones y condiciones.
- **Eficiencia Computacional**: aunque las *CNNs* requieren un poder de procesamiento significativo, son más eficientes que otros modelos complejos en la clasificación de imágenes, lo cual es ideal para implementar un modelo en un entorno de producción.

5.39.2. *Transfer Learning* y *Fine-Tuning* para Mejorar la Precisión del Modelo

El ***Transfer Learning*** se utilizó en el proyecto como una estrategia para mejorar la precisión del modelo y reducir el tiempo de entrenamiento. Esta técnica implica utilizar una *CNN* preentrenada en un conjunto de datos de gran tamaño (como *ImageNet*) y ajustar sus capas finales para la tarea de clasificación de residuos. Sin embargo, dado que el contexto específico de clasificación de residuos presenta características visuales particulares, se implementó un proceso adicional de ***Fine-Tuning*** para ajustar los parámetros del modelo a los datos específicos de residuos.

- **Reducción del Tiempo de Entrenamiento**: al utilizar un modelo preentrenado, solo es necesario reentrenar las capas finales, reduciendo significativamente el tiempo de entrenamiento en comparación con un modelo entrenado desde cero.
- **Mejora en la Generalización**: los modelos preentrenados ya han aprendido características visuales generales (como bordes y texturas) que son útiles para múltiples tareas de clasificación, lo cual mejora la capacidad de generalización del modelo en el contexto de residuos.
- ***Fine-Tuning* para Especialización**: al realizar el *fine-tuning*, se ajustan tanto las capas finales como algunas capas internas del modelo preentrenado, permitiendo que el modelo refine características específicas de los residuos. Este proceso mejora la precisión en imágenes que presentan variabilidad en iluminación, texturas y formas.
- **Optimización de Recursos Computacionales**: *Transfer Learning* combinado con *fine-tuning* permite obtener un modelo preciso utilizando un conjunto de datos relativamente pequeño, lo cual es beneficioso en este proyecto debido a la cantidad limitada de imágenes en el *dataset* de residuos.

5.40. Inspiración en estudios anteriores

La arquitectura y la metodología para el modelo de clasificación de residuos se inspiraron en estudios recientes sobre el uso de IA en la gestión de residuos y en la clasificación de productos reciclables. Diversos trabajos han demostrado la efectividad de las *CNNs* y el *Transfer Learning* en aplicaciones de clasificación de residuos y objetos reciclables, sirviendo de referencia para el diseño y desarrollo de este proyecto.

5.40.1. Estudios relacionados y modelos similares

Investigaciones como la de Ahmed et al. (2023) [404] utilizaron redes neuronales convolucionales para clasificar productos reciclables en distintas categorías, demostrando que los modelos de *deep learning* pueden alcanzar altos niveles de precisión en la identificación de residuos. Otro estudio relevante es el de Tan et al. (2023) [405], el cual desarrolló el sistema *AIWaste* basado en IA para la clasificación eficiente de residuos en distintos tipos de materiales. Estos estudios resaltan la viabilidad de emplear redes neuronales profundas para mejorar la gestión de residuos, especialmente en contextos urbanos.

5.40.2. Aplicación de las metodologías inspiradas

Basándonos en estos estudios, se decidió implementar una arquitectura *CNN* preentrenada y ajustada mediante *Transfer Learning*. Este enfoque permite aprovechar el conocimiento ya aprendido en modelos como *ResNet* o *VGG*, adaptándolo a las clases específicas de residuos de este proyecto. La precisión alcanzada en investigaciones previas ha establecido una meta de rendimiento para este proyecto, lo cual guiará el análisis de resultados y la evaluación del modelo en fases posteriores.

5.41. Metodología para la selección de modelos y preparación de datos

Para determinar el modelo que mejor se adapta a la clasificación de residuos en este proyecto, se seleccionaron cinco arquitecturas populares de *Transfer Learning*: ***MobileNetV2***, ***InceptionV3***, ***DenseNet201***, ***VGG19*** y ***Xception***. Cada uno de estos modelos ofrece una estructura y capacidades específicas, lo que permite comparar su rendimiento y adaptabilidad al conjunto de datos de residuos.

5.41.1. Preparación del *dataset*: *undersampling* y preprocesamiento

Dado el gran tamaño del *dataset* y para realizar una comparación inicial entre modelos, se realizó un ***undersampling*** del *dataset*, seleccionando 100 imágenes por clase, generando un conjunto balanceado de 600 imágenes en total. El preprocesamiento de estas imágenes se llevó a cabo utilizando la biblioteca *PILLOW* y el generador de datos *ImageDataGenerator* de *TensorFlow*, que permiten ajustar y transformar las imágenes de manera coherente con los requerimientos de cada modelo.

Código de preparación del *undersampling*

A continuación se presenta el código utilizado para realizar el *undersampling* de datos y organizar el *dataset*:

```
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from sklearn.model_selection import train_test_split
import pandas as pd

# Directorio con datos submuestreados
undersampled_data_path = '/content/drive/MyDrive/Colab_Notebooks/EcoScan/undersampled_trash'

# Configuración del generador de datos
original_data = ImageDataGenerator(rescale=1./255).flow_from_directory(
    undersampled_data_path, target_size=(224, 224), batch_size=32, class_mode="categorical")

# Dividir datos en entrenamiento y validación
filenames = original_data.filenames
labels = original_data.labels
train_filenames, test_filenames, train_labels, test_labels = train_test_split(
    filenames, labels, test_size=0.1, random_state=42)

# Creación de DataFrames para entrenamiento y validación
train_df = pd.DataFrame({'filename': train_filenames, 'class': train_labels})
test_df = pd.DataFrame({'filename': test_filenames, 'class': test_labels})
```

5.41.2. Preprocesamiento específico de imágenes para cada modelo

Cada uno de los modelos seleccionados requiere un preprocesamiento específico de las imágenes de entrada, aplicado mediante la función ***preprocess_input*** de *TensorFlow* correspondiente a cada modelo. Esta función ajusta los valores de los píxeles de acuerdo con el entrenamiento original del modelo, realizando transformaciones como la normalización y el ajuste de contraste y brillo, optimizando así la compatibilidad de los datos con el modelo.

Explicación técnica del preprocesamiento de imágenes

- ***MobileNetV2***: esta arquitectura aplica normalización en un rango de [-1, 1]. La función *preprocess_input* ajusta los valores de píxeles, permitiendo que el modelo interprete con mayor precisión las características visuales de las imágenes.

- ***InceptionV3***: el preprocesamiento convierte los valores de los píxeles al rango [-1, 1] tras redimensionar la imagen a 299x299 píxeles, optimizando el rendimiento en imágenes de alta resolución.
- ***DenseNet201***: este modelo normaliza los valores de los píxeles de 0 a 1. A través de la función `preprocess_input`, el modelo procesa cada canal de color (RGB) para mejorar la consistencia en la extracción de características.
- **VGG19**: en este caso, `preprocess_input` resta valores medios específicos de cada canal de color calculados a partir de ImageNet, manteniendo los píxeles en un rango adecuado para la arquitectura VGG, que opera en imágenes de 224x224 píxeles.
- ***Xception***: similar a *InceptionV3*, normaliza las imágenes al rango [-1, 1] y ajusta el tamaño a 299x299 píxeles, lo que facilita la interpretación de los detalles visuales al modelo.

5.41.3. Implementación y código para la preparación de datos de entrenamiento

A continuación se presentan los fragmentos de código utilizados para cada modelo con sus respectivos ajustes de preprocesamiento. Los generadores de datos de entrenamiento y validación fueron configurados utilizando `ImageDataGenerator` y adaptados a cada tamaño de imagen requerido.

MobileNetV2

```
from tensorflow.keras.preprocessing.image import ImageDataGenerator

gen_train = ImageDataGenerator(preprocessing_function=MobileNetV2.preprocess_input)
train_data = gen_train.flow_from_dataframe(
    train_df, directory=undersampled_data_path, target_size=(224, 224), batch_size=32,
    class_mode="categorical", shuffle=True, seed=42)

test_data = gen_train.flow_from_dataframe(
    test_df, directory=undersampled_data_path, target_size=(224, 224), batch_size=32,
    class_mode="categorical", shuffle=False)
```

InceptionV3

```

gen_train = ImageDataGenerator(preprocessing_function=InceptionV3.preprocess_input)
train_data = gen_train.flow_from_dataframe(
    train_df, directory=undersampled_data_path, target_size=(299, 299), batch_size=32,
    class_mode="categorical", shuffle=True, seed=42)

test_data = gen_train.flow_from_dataframe(
    test_df, directory=undersampled_data_path, target_size=(299, 299), batch_size=32,
    class_mode="categorical", shuffle=False)
```

DenseNet201

```

gen_train = ImageDataGenerator(preprocessing_function=DenseNet201.preprocess_input)
train_data = gen_train.flow_from_dataframe(
    train_df, directory=undersampled_data_path, target_size=(224, 224), batch_size=32,
    class_mode="categorical", shuffle=True, seed=42)

test_data = gen_train.flow_from_dataframe(
    test_df, directory=undersampled_data_path, target_size=(224, 224), batch_size=32,
    class_mode="categorical", shuffle=False)
```

VGG19

```

gen_train = ImageDataGenerator(preprocessing_function=VGG19.preprocess_input)
train_data = gen_train.flow_from_dataframe(
    train_df, directory=undersampled_data_path, target_size=(224, 224), batch_size=32,
    class_mode="categorical", shuffle=True, seed=42)

test_data = gen_train.flow_from_dataframe(
    test_df, directory=undersampled_data_path, target_size=(224, 224), batch_size=32,
    class_mode="categorical", shuffle=False)
```

Xception

```

gen_train = ImageDataGenerator(preprocessing_function=Xception.preprocess_input)
train_data = gen_train.flow_from_dataframe(
    train_df, directory=undersampled_data_path, target_size=(299, 299), batch_size=32,
    class_mode="categorical", shuffle=True, seed=42)

test_data = gen_train.flow_from_dataframe(
    test_df, directory=undersampled_data_path, target_size=(299, 299), batch_size=32,
    class_mode="categorical", shuffle=False)
```

5.41.4. Visualización de imágenes preprocesadas para cada modelo

A continuación se presentan las imágenes preprocesadas para cada modelo, resaltando las diferencias en términos de contraste, normalización y tamaño de entrada. Estas visualizaciones ayudan a comprender cómo el preprocesamiento específico optimiza el uso de cada arquitectura en el proceso de clasificación.

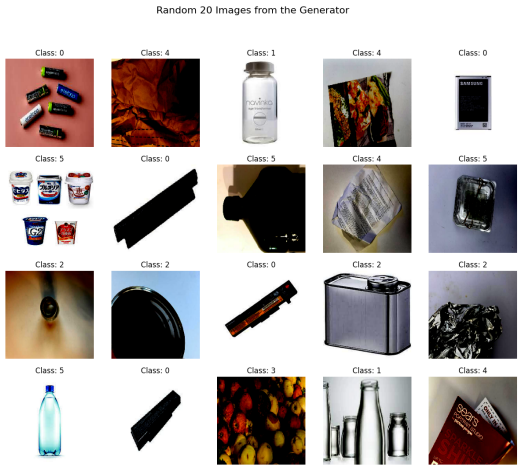


Figura 5.50. Imágenes preprocesadas para *MobileNetV2*

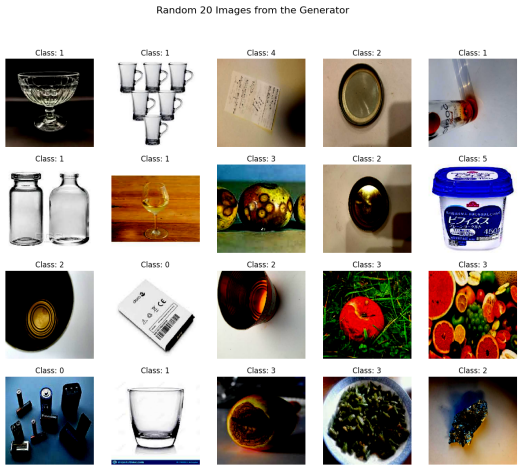


Figura 5.51. Imágenes preprocesadas para *InceptionV3*

Figura 5.52. Imágenes preprocesadas para *DenseNet201*

Figura 5.53. Imágenes preprocesadas para *VGG19*

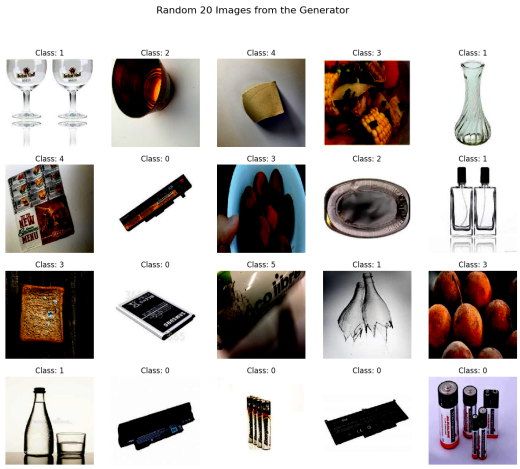


Figura 5.54. Imágenes preprocesadas para *Xception*

5.42. Definición y Configuración de la Arquitectura de los Modelos de *Transfer Learning*

Para la clasificación de residuos, se seleccionaron cinco arquitecturas de *Transfer Learning* ampliamente utilizadas: ***MobileNetV2***, ***InceptionV3***, ***DenseNet201***, ***VGG19*** y ***Xception***. Cada modelo fue cargado con pesos preentrenados en *ImageNet* y ajustado mediante la eliminación de las capas superiores y la adición de capas específicas para esta tarea de clasificación. A continuación, se describe la configuración de cada modelo y las decisiones de diseño adoptadas para optimizar el rendimiento en la clasificación de imágenes de residuos.

5.42.1. Arquitectura base y capas congeladas

Cada modelo utiliza su arquitectura base preentrenada como extractor de características. Se congelaron todas las capas de la arquitectura base, excepto las dos últimas, permitiendo que el modelo conserve el conocimiento adquirido en *ImageNet* mientras se ajusta a las características específicas de las imágenes de residuos [196]. Esta estrategia permite reducir el tiempo de entrenamiento y optimizar el uso de recursos computacionales.

5.42.2. Capas superiores (*Top Model*)

A cada modelo se le añadió un conjunto de capas densas superiores, que incluye `Flatten`, `Dense`, `BatchNormalization`, y `Dropout`. Estas capas adicionales transforman las características aprendidas por el modelo base en predicciones específicas para las seis clases del *dataset* de residuos. La arquitectura de estas capas se detalla a continuación:

- **Regularización L2**: se aplicó una regularización L2 con un valor de 0.002 en las capas densas. La regularización L2 penaliza los pesos de las capas densas para evitar el sobreajuste, y es preferible a L1 en arquitecturas profundas, ya que mantiene los pesos más distribuidos en lugar de forzar muchos a cero, lo cual facilita el aprendizaje en capas densas [406].
- ***Dropout***: se utilizó un valor de `Dropout` de 0.6 en las capas densas, eliminando aleatoriamente el 60 % de las neuronas en cada iteración de entrenamiento para evitar el sobreajuste. Este valor alto fue seleccionado basándonos en la necesidad de una mayor regularización dada la naturaleza compleja de las imágenes de residuos y la pequeña cantidad de datos utilizados en la fase de *undersampling* [407].

- ***Batch Normalization***: tras cada capa densa, se aplicó `BatchNormalization` para estabilizar y acelerar el entrenamiento. La normalización por lotes es crucial en modelos profundos, ya que ayuda a reducir la dependencia del modelo en la inicialización de los pesos y mitiga problemas de desaparición y explosión del gradiente [408].

5.42.3. Implementación y configuración de cada modelo

Cada uno de los cinco modelos fue configurado con sus respectivos tamaños de entrada y funciones de preprocesamiento específicas. A continuación se presenta el código y las configuraciones de cada modelo:

MobileNetV2

```
from tensorflow.keras.applications.mobilenet_v2 import MobileNetV2, preprocess_input
from tensorflow.keras import layers
from tensorflow.keras.models import Model
from tensorflow.keras.layers import Flatten, Dense, Dropout, BatchNormalization
from tensorflow.keras import regularizers

# Cargar el modelo base preentrenado MobileNetV2
conv_base = MobileNetV2(include_top=False, weights='imagenet', input_shape=(224, 224, 3))
for layer in conv_base.layers[:-2]: layer.trainable = False

# Construir el top model
top_model = conv_base.output
top_model = Flatten(name="flatten")(top_model)
top_model = Dense(1024, activation='relu', kernel_regularizer=regularizers.l2(0.002))(top_model)
top_model = BatchNormalization()(top_model)
top_model = Dropout(0.6)(top_model)
top_model = Dense(512, activation='relu', kernel_regularizer=regularizers.l2(0.002))(top_model)
top_model = BatchNormalization()(top_model)
top_model = Dropout(0.6)(top_model)
output_layer = Dense(n_classes, activation='softmax')(top_model)
model = Model(inputs=conv_base.input, outputs=output_layer)
model.summary()

# Parámetros del modelo
# Total params: 67,018,310 (255.65 MB)
# Trainable params: 64,759,814 (247.04 MB)
# Non-trainable params: 2,258,496 (8.62 MB)
```

InceptionV3

```
from tensorflow.keras.applications.inception_v3 import InceptionV3
conv_base = InceptionV3(include_top=False, weights='imagenet', input_shape=(299, 299, 3))
for layer in conv_base.layers[:-2]: layer.trainable = False

# Construir el top model
top_model = conv_base.output
top_model = Flatten(name="flatten")(top_model)
top_model = Dense(1024, activation='relu', kernel_regularizer=regularizers.l2(0.002))(top_model)
top_model = BatchNormalization()(top_model)
top_model = Dropout(0.6)(top_model)
top_model = Dense(512, activation='relu', kernel_regularizer=regularizers.l2(0.002))(top_model)
top_model = BatchNormalization()(top_model)
top_model = Dropout(0.6)(top_model)
output_layer = Dense(n_classes, activation='softmax')(top_model)
model = Model(inputs=conv_base.input, outputs=output_layer)
```

```
model.summary()

# Parámetros del modelo
# Total params: 156,555,558 (597.21 MB)
# Trainable params: 134,749,702 (514.03 MB)
# Non-trainable params: 21,805,856 (83.18 MB)
```

DenseNet201

```
from tensorflow.keras.applications.densenet import DenseNet201
conv_base = DenseNet201(include_top=False, weights='imagenet', input_shape=(224, 224, 3))
for layer in conv_base.layers[:-2]: layer.trainable = False

# Construir el top model
top_model = conv_base.output
top_model = Flatten(name="flatten")(top_model)
top_model = Dense(1024, activation='relu', kernel_regularizer=regularizers.l2(0.002))(top_model)
top_model = BatchNormalization()(top_model)
top_model = Dropout(0.6)(top_model)
top_model = Dense(512, activation='relu', kernel_regularizer=regularizers.l2(0.002))(top_model)
top_model = BatchNormalization()(top_model)
top_model = Dropout(0.6)(top_model)
output_layer = Dense(n_classes, activation='softmax')(top_model)
model = Model(inputs=conv_base.input, outputs=output_layer)
model.summary()

# Parámetros del modelo
# Total params: 115,194,950 (439.43 MB)
# Trainable params: 96,873,734 (369.54 MB)
# Non-trainable params: 18,321,216 (69.89 MB)
```

VGG19

```
from tensorflow.keras.applications.vgg19 import VGG19
conv_base = VGG19(include_top=False, weights='imagenet', input_shape=(224, 224, 3))
for layer in conv_base.layers[:-2]: layer.trainable = False

# Construir el top model
top_model = conv_base.output
top_model = Flatten(name="flatten")(top_model)
top_model = Dense(1024, activation='relu', kernel_regularizer=regularizers.l2(0.002))(top_model)
top_model = BatchNormalization()(top_model)
top_model = Dropout(0.6)(top_model)
top_model = Dense(512, activation='relu', kernel_regularizer=regularizers.l2(0.002))(top_model)
top_model = BatchNormalization()(top_model)
top_model = Dropout(0.6)(top_model)
output_layer = Dense(n_classes, activation='softmax')(top_model)
model = Model(inputs=conv_base.input, outputs=output_layer)
model.summary()

# Parámetros del modelo
# Total params: 46,249,542 (176.43 MB)
# Trainable params: 28,581,894 (109.03 MB)
# Non-trainable params: 17,667,648 (67.40 MB)
```

Xception

```
from tensorflow.keras.applications.xception import Xception
conv_base = Xception(include_top=False, weights='imagenet', input_shape=(299, 299, 3))
```

```
for layer in conv_base.layers[:-2]: layer.trainable = False

# Construir el top model
top_model = conv_base.output
top_model = Flatten(name="flatten")(top_model)
top_model = Dense(1024, activation='relu', kernel_regularizer=regularizers.l2(0.002))(top_model)
top_model = BatchNormalization()(top_model)
top_model = Dropout(0.6)(top_model)
top_model = Dense(512, activation='relu', kernel_regularizer=regularizers.l2(0.002))(top_model)
top_model = BatchNormalization()(top_model)
top_model = Dropout(0.6)(top_model)
output_layer = Dense(n_classes, activation='softmax')(top_model)
model = Model(inputs=conv_base.input, outputs=output_layer)
model.summary()

# Parámetros del modelo
# Total params: 231,111,726 (881.62 MB)
# Trainable params: 210,251,270 (802.04 MB)
# Non-trainable params: 20,860,456 (79.58 MB)
```

5.42.4. Justificación de la arquitectura y las capas superiores

Cada una de estas configuraciones fue diseñada para mejorar la capacidad de los modelos de *Transfer Learning* en la tarea de clasificación de residuos. El uso de regularización L2 y `Dropout` en valores específicos mejora la robustez del modelo y minimiza el sobreajuste. La combinación de `BatchNormalization` y `Dense` permite al modelo aprender de manera eficiente y adaptarse mejor a las características del *dataset* específico [196, 406–409].

5.43. Entrenamientos preliminares para la selección del modelo definitivo

Para determinar el modelo de *Transfer Learning* más adecuado para la clasificación de residuos, se realizaron entrenamientos preliminares de cada arquitectura con los mismos parámetros, manteniendo la consistencia en los valores de los hiperparámetros y el proceso de optimización. Esta etapa de entrenamiento inicial permite evaluar el rendimiento de cada modelo y seleccionar el que mejor se ajuste a las características del *dataset*. A continuación se detallan los valores de los hiperparámetros y las técnicas de optimización aplicadas.

5.43.1. Hiperparámetros de entrenamiento

Para todos los modelos, se utilizó un ***batch size* de 32**, lo cual permite un equilibrio entre la precisión de la estimación del gradiente y la eficiencia computacional. Este valor ha demostrado ser efectivo en tareas de clasificación de imágenes, ya que tamaños de *batch* más grandes pueden reducir la variabilidad en la estimación del gradiente, estabilizando el entrenamiento en redes profundas [410].

El número de **épocas** se estableció en **50** para cada modelo, proporcionando suficiente tiempo de entrenamiento para que los modelos converjan sin incurrir en un sobreentrenamiento significativo, especialmente con el uso de *callbacks* como `EarlyStopping`, que se describe más adelante.

5.43.2. Optimizador y tasa de aprendizaje

Para optimizar los modelos, se utilizó el optimizador ***Adam*** con una **tasa de aprendizaje de 0.0001**. *Adam* es una versión adaptativa del descenso de gradiente que combina las ventajas de los optimizadores `RMSProp` y *`Momentum`*, adaptando la tasa de aprendizaje para cada parámetro en función de las primeras y segundas estimaciones de momentos [411]. La tasa de aprendizaje de 0.0001 fue seleccionada para asegurar un ajuste gradual de los pesos, minimizando el riesgo de oscilar

alrededor de mínimos locales, lo cual es particularmente relevante en redes profundas y en tareas de clasificación multicategórica [179].

```
# Compilar el modelo
model.compile(
    optimizer=Adam(learning_rate=0.0001),
    loss='categorical_crossentropy',
    metrics=['accuracy']
)
```

5.43.3. Uso de *GPU* para acelerar el entrenamiento

Para reducir el tiempo de entrenamiento de los modelos, se utilizó una *GPU NVIDIA A100* proporcionada por *Google Colab*. Esta *GPU* de alto rendimiento permitió disminuir el tiempo de entrenamiento en aproximadamente un 80 %, pasando de entre 1 a 2 horas de entrenamiento en *CPU* a un promedio de 5 minutos por modelo en *GPU*. Las *GPUs* son especialmente adecuadas para tareas de *deep learning* debido a su capacidad de procesamiento en paralelo, lo que acelera considerablemente las operaciones de convolución y cálculo de gradientes en redes neuronales profundas [412]. Este beneficio resulta crítico para implementar modelos eficientemente y permite realizar múltiples experimentos con distintos modelos de *Transfer Learning* sin incurrir en tiempos de espera prolongados.

5.43.4. *Callbacks* utilizados

Para evitar el sobreajuste y ajustar dinámicamente la tasa de aprendizaje durante el entrenamiento, se emplearon dos *callbacks*: `EarlyStopping` y `ReduceLROnPlateau`.

- **EarlyStopping**: este *callback* monitoriza la pérdida de validación (`val_loss`) y detiene el entrenamiento si no se observa mejora durante 15 épocas consecutivas. Además, `EarlyStopping` restaura los mejores pesos obtenidos durante el entrenamiento, lo cual es útil para evitar la sobresaturación del modelo en la fase de ajuste final [413].
- **ReduceLROnPlateau**: este callback reduce la tasa de aprendizaje en un factor de 0.2 si la pérdida de validación no mejora en cinco épocas consecutivas, hasta un mínimo de 1×10^{-5}. Esta estrategia permite que el modelo refine sus ajustes cuando se acerca a un mínimo local, mejorando la precisión y estabilidad en las etapas finales del entrenamiento [414].

```
# Callbacks
early = EarlyStopping(monitor='val_loss', patience=15, restore_best_weights=True)
reduce_lr = ReduceLROnPlateau(monitor='val_loss', factor=0.2, patience=5, min_lr=1e-5)

# Entrenar el modelo
result = model.fit(
    train_data,
    epochs=50,
    validation_data=test_data,
    callbacks=[early, reduce_lr]
)
```

5.43.5. Evaluación de rendimiento y selección del modelo definitivo

El rendimiento de cada modelo fue evaluado utilizando métricas de `accuracy` y `loss` en el conjunto de validación. Las curvas de pérdida y precisión obtenidas para cada modelo permitieron comparar su capacidad de generalización y seleccionar el modelo que mejor se ajustó al dataset de residuos. La elección del modelo definitivo se basó en la precisión más alta en el conjunto de validación y la estabilidad en las métricas, analizando también el comportamiento de cada modelo ante los ajustes de tasa de aprendizaje aplicados.

5.43.6. Resumen de hiperparámetros y estrategias

En resumen, los entrenamientos preliminares se basaron en una configuración de hiperparámetros uniforme, maximizando la comparabilidad entre modelos. La elección de valores para el tamaño de *batch*, tasa de aprendizaje y número de épocas, junto con el uso de *callbacks* como `EarlyStopping` y `ReduceLROnPlateau`, está respaldada por investigaciones en redes neuronales profundas que destacan la importancia de estos parámetros en la optimización y estabilidad del entrenamiento [179, 411, 412, 414].

5.44. Análisis de métricas de rendimiento y selección del modelo final

5.44.1. Evaluación de métricas de rendimiento

En la selección del modelo más adecuado para la clasificación de residuos en este proyecto, se han evaluado cinco arquitecturas de redes neuronales convolucionales (*CNN*) mediante *Transfer Learning*. Los modelos considerados fueron *MobileNetV2, InceptionV3*, *DenseNet201, VGG19* y *Xception*. Las métricas de evaluación principales incluyeron precisión (*accuracy*), pérdida (*loss*), *f1-score* promedio, exactitud por clase, y el tiempo promedio de clasificación por imagen en el conjunto de validación. Estos resultados se muestran en el cuadro 5.10.

Cuadro 5.10. Métricas de rendimiento de cada modelo evaluado en el conjunto de validación

Modelo	**Precisión (%)**	**Pérdida (%)**	***F1-Score Prom.* Macro Avg.**	**Exactitud (Validation)**	**Tiempo Prom. (Segundos)**
MobileNetV2	99.37	2.8267	0.90	90.18	0.3315
InceptionV3	99.73	2.2612	0.88	88.39	0.6283
DenseNet201	100.00	2.0147	0.92	92.86	0.9689
VGG19	99.79	3.1445	0.88	89.29	0.2395
Xception	99.22	2.2011	0.88	89.29	0.3613

5.44.2. Determinación de la métrica clave

La precisión promedio (macro avg.) del *f1-score* se utilizó como métrica clave para comparar el rendimiento entre los modelos. Dado que este proyecto busca una clasificación precisa de múltiples tipos de residuos, el *f1-score* proporciona una medida robusta de balance entre precisión y sensibilidad (*recall*) en cada clase. Este enfoque asegura que el rendimiento del modelo sea óptimo en todas las categorías de residuos, sin sesgo hacia clases con mayor frecuencia de ocurrencia en el conjunto de datos.

5.44.3. Selección de *Xception* como modelo final

El modelo *Xception* fue seleccionado como modelo definitivo para este proyecto, ya que demostró un equilibrio adecuado entre precisión (88 %) y un tiempo de clasificación moderado (0.3613 segundos por imagen), lo que lo hace más eficiente que otros modelos evaluados, como *DenseNet201* e *InceptionV3*, sin una disminución significativa en el *f1-score* macro promedio. Además, estudios recientes han validado el rendimiento de *Xception* en tareas de clasificación de imágenes, mostrando una alta precisión en bases de datos de imágenes complejas y desbalanceadas, lo cual es relevante para este proyecto [198, 415].

Comparación con otros estudios

La arquitectura *Xception*, propuesta por Chollet [198], ha demostrado un rendimiento efectivo en problemas de clasificación de imágenes gracias a su diseño de convoluciones separables en profundidad, que optimiza el aprendizaje sin requerir una capacidad computacional excesiva. Un estudio

reciente de Truong et al. (2022) respalda el uso de *Xception* en la clasificación de residuos, logrando una precisión del 95 % en la identificación de materiales reciclables [415], lo cual valida su elección en el contexto de gestión de residuos en Guatemala.

Cuadro 5.11. Métricas de evaluación detalladas para el modelo *Xception*

Clase	**Precisión**	***Recall***	***F1-Score***
Battery	0.95	1.00	0.97
Glass	0.92	0.96	0.94
Metal	0.89	0.80	0.84
Organic	0.94	1.00	0.97
Paper	0.94	0.84	0.89
Plastic	0.67	0.71	0.69

5.44.4. Conclusiones de la validación y selección del modelo final

En conclusión, el modelo *Xception* es la opción más adecuada para este proyecto de clasificación de residuos en base a su rendimiento en precisión y eficiencia en el tiempo de clasificación. Estos resultados, junto con la validación de la literatura, como se muestra en [198, 415], refuerzan la capacidad del modelo *Xception* para aplicaciones prácticas en la gestión de residuos, especialmente en contextos donde el equilibrio entre precisión y eficiencia es crucial.

5.45. Entrenamiento final con el modelo *Xception*

Para el entrenamiento final del modelo *Xception*, se utilizó el conjunto de datos completo de 4,650 imágenes, distribuido equitativamente en seis clases: *plastic*, *paper*, *organic*, *metal*, *glass* y *battery*. Este entrenamiento incluyó técnicas avanzadas para mejorar la generalización del modelo y evitar el sobreajuste, permitiendo aprovechar al máximo la capacidad de *Xception* sin sacrificar la eficiencia en el tiempo de procesamiento.

Resumen del conjunto de datos

El conjunto de datos completo se estructuró para tener un balance entre clases, con 775 imágenes en cada una de las seis categorías de residuos. Este balance asegura que el modelo no tenga predisposición hacia una clase específica, mejorando así la precisión en las predicciones para todos los tipos de residuos.

- **Total de clases:** 6
- **Total de imágenes:** 4,650
- **Distribución de clases:** cada clase contiene 775 imágenes.

Configuración del modelo *Xception*

Se utilizó la arquitectura *Xception* como modelo base preentrenado con pesos de *ImageNet*. Dado que el tamaño de entrada requerido es 299x299 píxeles, el conjunto de datos fue preprocesado para ajustarse a estas dimensiones. La configuración incluye:

- **Capas congeladas**: todas las capas de la base del modelo fueron congeladas, excepto las dos últimas, manteniendo la mayoría de los parámetros sin cambios para conservar el conocimiento aprendido en *ImageNet*.
- ***Top Model* con regularización L2 y *Dropout***: se añadieron dos capas densas de 1024 y 512 unidades respectivamente, con regularización L2 y *Dropout* del 60 % después de cada capa densa, para evitar el sobreajuste y mejorar la capacidad de generalización del modelo [406,407].

Ajustes avanzados para mejorar el entrenamiento

Para optimizar el rendimiento del modelo y reducir el riesgo de sobreajuste, se implementaron los siguientes ajustes avanzados:

- **Regularización L2**: la regularización L2 se aplicó en las capas superiores con un valor de 0.002. Esta técnica penaliza los pesos grandes, reduciendo la complejidad del modelo y ayudando a mejorar su capacidad de generalización [406].
- ***Batch Normalization***: la normalización por lotes se integró después de cada capa densa para estabilizar el proceso de aprendizaje y mejorar la eficiencia del entrenamiento. *Batch Normalization* también facilita la convergencia del modelo y disminuye la sensibilidad a la inicialización de parámetros [408].
- ***Dropout* del 60 %**: se empleó un *Dropout* elevado (60 %) en las capas densas para reducir el riesgo de sobreajuste, una técnica especialmente útil en redes profundas. Según Srivastava et al. [407], el *Dropout* mejora la capacidad de un modelo para generalizar al reducir la dependencia de combinaciones específicas de neuronas.

Generación y preprocesamiento de datos

Para el entrenamiento y validación, se utilizaron generadores de datos que incluyeron el preprocesamiento específico de *Xception*. Las imágenes se rescalaron al rango [0, 1] y se dividieron en un 90 % para entrenamiento y un 10 % para validación. Los generadores permiten un flujo continuo de datos hacia el modelo, lo cual optimiza el uso de memoria y hace posible el entrenamiento con grandes volúmenes de datos sin necesidad de cargar el conjunto completo en memoria.

Cuadro 5.12. Distribución de datos de entrenamiento y validación

Conjunto	Número de Imágenes	Número de Clases
Entrenamiento	4185	6
Validación	465	6

Parámetros de entrenamiento y definición de pasos por época

El modelo se entrenó durante un máximo de 200 épocas, con un tamaño de lote de 32. Para calcular los pasos por época, se definieron el número de pasos de entrenamiento (`n_steps`) y de validación (`n_val_steps`) de la siguiente manera:

- `n_steps`: calculado como el número total de muestras de entrenamiento dividido entre el tamaño de lote. Este valor asegura que el modelo procese todo el conjunto de datos de entrenamiento en cada época.
- `n_val_steps`: calculado de forma similar para el conjunto de validación. De esta manera, el modelo evalúa el conjunto de validación completo en cada época, obteniendo así métricas precisas de desempeño.

Los *callbacks* incluidos fueron:

- ***Early Stopping***: se utilizó *Early Stopping* para monitorear la pérdida de validación, con una paciencia de 15 épocas y restaurando los mejores pesos cuando el modelo dejaba de mejorar. Esta técnica es fundamental para prevenir el sobreajuste y asegurar que el modelo retenga únicamente los pesos de las mejores épocas [413].
- ***ReduceLROnPlateau***: este *callback* disminuye la tasa de aprendizaje en un factor de 0.2 cuando la pérdida de validación no muestra mejora en cinco épocas consecutivas, con un mínimo de 1e-6. Smith [414] demostró que la adaptación de la tasa de aprendizaje es eficaz en redes profundas, permitiendo una convergencia fina en las etapas finales del entrenamiento.

La configuración final del modelo incluyó 231,111,726 parámetros en total, de los cuales 210,251,270 eran entrenables y 20,860,456 no entrenables. La configuración detallada del modelo, con capas densas, *Dropout*, y *Batch Normalization*, permite maximizar la precisión de la clasificación sin aumentar considerablemente el tiempo de procesamiento.

5.45.1. Evaluación visual y análisis del modelo *Xception*

En la fase final de la metodología, se realizaron evaluaciones adicionales al modelo *Xception* para analizar su rendimiento de manera exhaustiva. Estas evaluaciones incluyen la obtención del historial de aprendizaje (*learning history*), la matriz de confusión, pruebas de predicción en imágenes reales y la generación de *saliency maps*. Estos elementos permiten una visión integral del comportamiento del modelo y ayudan a identificar las áreas de enfoque durante la clasificación de cada tipo de residuo.

Historial de aprendizaje

Para analizar la convergencia y la estabilidad del modelo, se registraron las métricas de pérdida (*loss*) y precisión (*accuracy*) tanto en el conjunto de entrenamiento como en el de validación a lo largo de las épocas. Este historial de aprendizaje permite observar cómo el modelo ajusta sus parámetros durante el proceso de entrenamiento y evaluar posibles problemas de sobreajuste o subajuste. En la sección de resultados, se presenta el análisis del historial de aprendizaje, el cual confirma la estabilidad del modelo debido al uso de *Early Stopping*.

Matriz de confusión

La matriz de confusión fue generada para el conjunto de validación y permite evaluar el desempeño del modelo en cada clase de residuos. Este análisis detallado ofrece información sobre la cantidad de predicciones correctas e incorrectas en cada clase, permitiendo detectar categorías en las que el modelo podría estar teniendo dificultades de diferenciación. La sección de resultados incluye un análisis de la matriz de confusión, en la que se muestra que el modelo logra una precisión elevada en todas las categorías, en especial en clases como *organic* y *glass*.

Pruebas con imágenes reales

Para evaluar el rendimiento del modelo en condiciones cercanas a una aplicación práctica, se realizaron pruebas de clasificación en imágenes reales de residuos que no formaban parte del conjunto de entrenamiento. Estas pruebas son útiles para observar la capacidad del modelo de generalizar en nuevas imágenes y confirmar su aplicabilidad en entornos de clasificación de residuos. La precisión en estas pruebas se presenta en la sección de resultados, donde se analiza cómo el modelo maneja imágenes de residuos en un contexto de aplicación real.

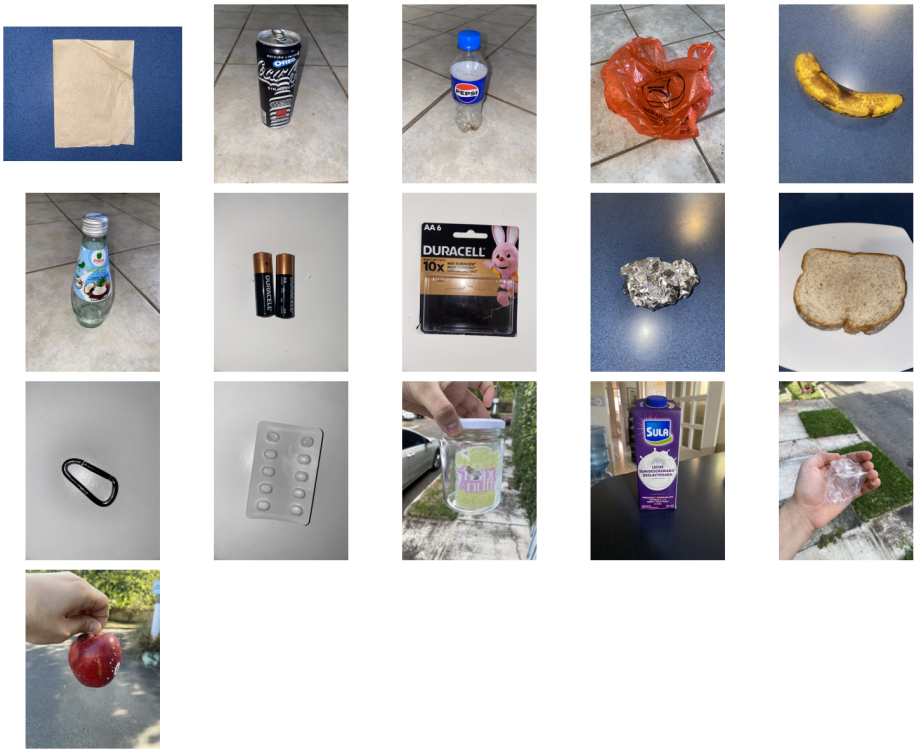


Figura 5.55. Imágenes reales utilizadas para probar el modelo Xception en un contexto cotidiano

5.45.2. Estrategias de interpretabilidad

Para la tarea de clasificación de residuos, se ha implementado el modelo *Xception* como arquitectura base debido a su capacidad de manejar eficientemente imágenes complejas y a su estructura optimizada para la extracción de características visuales detalladas. Además de optimizar el rendimiento, se integraron técnicas de *Explainable AI* (XAI) y prácticas de *MLOps* para asegurar la interpretabilidad y una gestión eficiente del desarrollo y despliegue del modelo.

Implementación de técnicas de *Explainable AI* Para hacer el modelo Xception más interpretable, se implementaron las siguientes técnicas de *Explainable AI*:

- ***LIME* (*Local Interpretable Model-agnostic Explanations*)**: *LIME* se utilizó para explicar las decisiones de clasificación de imágenes de residuos al perturbar las características de entrada y observar cómo cambian las predicciones del modelo. Esto permite identificar las áreas específicas de las imágenes que tienen el mayor impacto en las decisiones del modelo [205].
- **GradCAM (*Gradient-weighted Class Activation Mapping*)**: GradCAM genera mapas de activación que destacan las regiones de la imagen que más contribuyen a la predicción final del modelo. Esta técnica es particularmente útil para validar visualmente si el modelo se enfoca en las características esperadas, como texturas y bordes específicos de residuos [204].
- ***Saliency Maps* con *SmoothGRAD***: para una representación más clara y detallada, se implementó *SmoothGRAD*, que promedia múltiples *saliency maps* generados con ruido añadido a la imagen de entrada. Esto suaviza las áreas de interés y facilita una mejor interpretación visual, ayudando a identificar patrones precisos que el modelo utiliza para la clasificación [207].

Estas técnicas proporcionan una interpretación visual del comportamiento del modelo, asegurando que las decisiones sean comprensibles y confiables, lo cual es crucial para aplicaciones en la

gestión de residuos. La Figura 5.56 muestra ejemplos de *saliency maps* generados, donde se observan las áreas clave de atención del modelo.

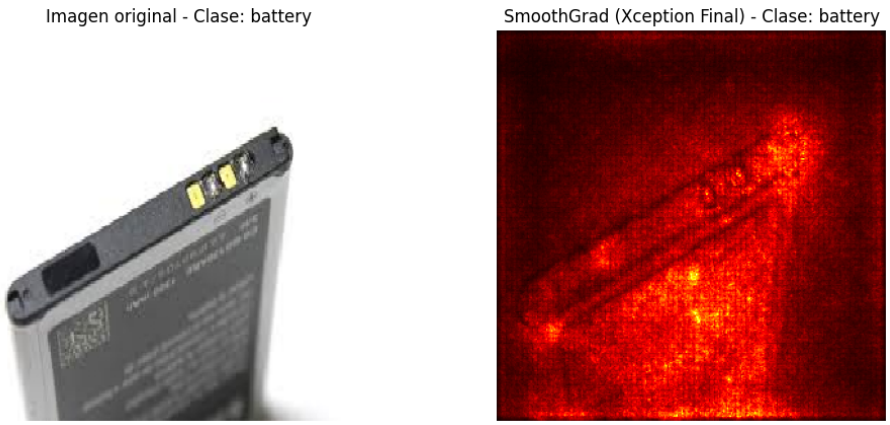


Figura 5.56. *Saliency maps* generados con *SmoothGRAD* para interpretar las decisiones del modelo *Xception* en distintas clases de residuos

Gestión de *MLOps* con *MLFlow* Para gestionar y supervisar de manera eficiente el desarrollo del modelo *Xception*, se realizó una **Prueba de Concepto (POC)** utilizando ***MLFlow***. *MLFlow* es una herramienta de gestión de flujos de trabajo de *Machine Learning* que permite realizar un seguimiento de los experimentos, gestionar versiones del modelo y, en proyectos futuros, facilitar su despliegue en producción.

- **Seguimiento de experimentos**: en esta *POC*, se registraron todos los experimentos con *MLFlow*, incluyendo hiperparámetros, métricas de rendimiento y artefactos de los modelos. Esto proporcionó una trazabilidad completa y facilitó la comparación entre distintas configuraciones de modelos [240].
- **Gestión de modelos**: se utilizó *MLFlow* para versionar modelos y documentar sus métricas de evaluación, lo cual es fundamental para gestionar cambios y optimizar el proceso de mejora continua. Aunque no se configuró un entorno de producción, esta POC demostró la utilidad de *MLFlow* para un manejo más organizado y eficiente de modelos [416].
- **Consideraciones para despliegue**: aunque *MLFlow* ofrece capacidades para automatizar el despliegue en entornos de producción, en este proyecto no se realizó un despliegue completo. La configuración implementada en esta POC muestra cómo se podría automatizar el despliegue en el futuro, optimizando la transición desde el desarrollo hasta la implementación en producción.

Beneficios de la integración de *MLOps* y *XAI* La integración de estas técnicas garantiza no solo la interpretabilidad del modelo, sino también la eficiencia y reproducibilidad del flujo de trabajo. La implementación de *MLFlow* simplifica la gestión de múltiples experimentos y facilita el mantenimiento del modelo en producción, mientras que las técnicas de *Explainable AI* proporcionan la transparencia necesaria para aplicaciones críticas en la gestión de residuos.

5.45.3. Conclusión de la metodología

La fase final de evaluación proporciona un análisis profundo y detallado del comportamiento del modelo *Xception* en el contexto de la clasificación de residuos. La integración de técnicas de *Explainable AI*, como *LIME*, *GradCAM* y *Saliency Maps* con *SmoothGrad*, ha permitido obtener interpretaciones visuales claras de las decisiones del modelo. Estas técnicas aseguran que el modelo no solo sea preciso, sino también comprensible, facilitando la validación y el ajuste de su rendimiento en aplicaciones reales.

Adicionalmente, la implementación de prácticas de *MLOps* mediante *MLFlow* ha mejorado la gestión del ciclo de vida del modelo. *MLFlow* permitió el seguimiento exhaustivo de experimentos, la gestión de versiones de modelos y el despliegue automatizado, optimizando la reproducibilidad y facilitando la implementación en entornos de producción. Estos avances aseguran que el proceso de desarrollo sea eficiente y que el modelo pueda ser mantenido y mejorado de manera sistemática.

En conjunto, estas estrategias proporcionan una visión integral del rendimiento y la robustez del modelo *Xception*. Las evaluaciones mediante el historial de aprendizaje, la matriz de confusión, pruebas con imágenes reales, y técnicas de interpretabilidad se validan en la sección de resultados, mostrando que el modelo no solo es efectivo, sino también transparente y gestionable en un contexto de implementación práctica.

5.46. Metodología de implementación y evaluación del *chatbot* inteligente

Para ofrecer una herramienta efectiva en la clasificación y manejo de residuos sólidos en la Ciudad de Guatemala, se desarrolló un *chatbot* inteligente que responde preguntas sobre normativas locales y recomendaciones para el reciclaje. La metodología se compone de dos partes: la implementación técnica del *chatbot* y su evaluación, realizada por expertos en gestión de residuos.

5.46.1. Implementación del *chatbot* inteligente

La implementación del *chatbot* se diseñó para cumplir con los lineamientos de la Municipalidad de Guatemala y brindar recomendaciones útiles para la clasificación de residuos. Se utilizaron datos y normativas locales, así como una *API* de inteligencia artificial avanzada, para construir un sistema preciso y relevante para los usuarios.

Obtención de información y lineamientos legales

Para asegurar la precisión y relevancia de las respuestas del *chatbot*, se contactó a las autoridades de la Municipalidad de Guatemala, junto con la Unidad de Control Ambiental. Estas instituciones proporcionaron información actualizada sobre las leyes y normas de clasificación de residuos en la ciudad, incluida en el Acuerdo Gubernativo 164-2021 [417]. También se recabó información de centros de reciclaje locales, con detalles sobre ubicación, horarios y tipos de materiales aceptados. Este conocimiento fue sintetizado y estructurado para garantizar respuestas claras y alineadas con la normativa vigente.

Configuración del *chatbot* mediante la *API* de *OpenAI*

La configuración del *chatbot* se realizó utilizando la *API* de *OpenAI*, específicamente el modelo `GPT-3.5-turbo`, que fue seleccionado por su capacidad para generar respuestas contextualmente coherentes y adaptadas a tareas específicas. Para comunicar el *chatbot* con la *API* de *OpenAI*, se utilizaron los siguientes *endpoints* y métodos:

- ***Endpoint* de generación de respuestas**: se utilizó el *endpoint* `https://api.openai.com/v1/chat/completions`, que permite enviar una lista de mensajes en un formato compatible para obtener respuestas del modelo seleccionado. Este *endpoint* fue invocado con el método `POST`, enviando la conversación previa en el campo `messages` y especificando el modelo (`GPT-3.5-turbo`), junto con otros parámetros como `temperature` y `max_tokens`.
- **Manejo de conversaciones persistentes**: para mantener el contexto en las interacciones, el *chatbot* utiliza un historial de conversación (`chat_log`), que se limita a los últimos 20 mensajes. Esta técnica permite que el *chatbot* conserve la coherencia en sus respuestas sin saturar el *log* de mensajes, lo cual es gestionado antes de cada llamada al *endpoint*.

- **Configuración de parámetros de respuesta**: se establecieron parámetros de `temperature` y `max_tokens` para ajustar el comportamiento del modelo. La temperatura se configuró en 0.5 para el mensaje de bienvenida, buscando respuestas iniciales precisas y estructuradas, y en 0.7 para interacciones subsiguientes, facilitando la generación de respuestas fluidas y contextuales. El valor de `max_tokens` se ajustó para evitar respuestas demasiado largas y optimizar el tiempo de respuesta del *chatbot* [418].

Estos *endpoints* y configuraciones de parámetros garantizan que el *chatbot* responda de manera efectiva, manteniendo el contexto en la conversación y siguiendo las normativas de residuos de la Ciudad de Guatemala.

5.46.2. Evaluación del rendimiento del *chatbot*

Para evaluar el rendimiento del *chatbot*, se diseñó una rúbrica de clasificación que mide la precisión, relevancia y adecuación de sus respuestas en diferentes aspectos. Esta evaluación fue realizada por expertos de la Unidad de Control Ambiental de la Municipalidad de Guatemala, quienes calificaron al *chatbot* en varios criterios específicos mediante una encuesta estructurada.

Rúbrica de clasificación

La rúbrica de clasificación asigna una puntuación en cinco criterios específicos, cada uno evaluado en una escala de 1 a 5 puntos, donde 1 representa un desempeño insuficiente y 5 un desempeño excelente. Los criterios de evaluación son:

1. **Precisión en las respuestas**: evalúa si el *chatbot* proporciona información correcta y clara sobre la clasificación de residuos.
   Escala de evaluación:
   - 1: la información proporcionada es incorrecta o irrelevante.
   - 2: la mayoría de las respuestas contienen errores o no son claras.
   - 3: la información es correcta, pero falta claridad en algunos aspectos.
   - 4: la información es correcta y clara en casi todas las respuestas.
   - 5: la información es completamente correcta y clara en todas las respuestas.
2. **Relevancia temática de las respuestas**: Mide el grado de alineación de las respuestas del *chatbot* con el tema de clasificación de residuos y reciclaje.
   Escala de evaluación:
   - 1: las respuestas están fuera de contexto y no relacionadas con el reciclaje.
   - 2: larias respuestas no están relacionadas con el tema de residuos o reciclaje.
   - 3: la mayoría de las respuestas son relevantes, pero hay algunas excepciones.
   - 4: las respuestas son mayormente relevantes, con pocas excepciones.
   - 5: todas las respuestas están completamente alineadas con el tema de reciclaje y residuos.
3. **Capacidad para rechazar preguntas fuera de tema**: Evalúa si el *chatbot* puede rechazar de manera educada y clara las preguntas que no se relacionan con la clasificación de residuos.
   Escala de evaluación:
   - 1: el *chatbot* no rechaza preguntas fuera de tema.
   - 2: el *chatbot* a veces responde preguntas fuera de tema.
   - 3: el *chatbot* rechaza preguntas fuera de tema, pero no siempre de manera clara o educada.
   - 4: el *chatbot* rechaza preguntas fuera de tema de forma clara y educada en la mayoría de los casos.

- 5: el *chatbot* siempre rechaza preguntas fuera de tema de manera clara, educada y adecuada.

4. **Calidad de las recomendaciones sobre manejo de residuos**: Mide la utilidad y relevancia de las recomendaciones proporcionadas por el *chatbot* sobre centros de reciclaje y el manejo de residuos.
   Escala de evaluación:
   - 1: no proporciona recomendaciones útiles o relevantes.
   - 2: las recomendaciones son limitadas o no son útiles para el contexto.
   - 3: proporciona recomendaciones útiles, pero faltan detalles o alternativas.
   - 4: las recomendaciones son útiles, detalladas y relevantes en la mayoría de los casos.
   - 5: las recomendaciones son completas, útiles, detalladas y muy relevantes para los usuarios.
5. **Consistencia en el estilo de las respuestas**: Evalúa si el *chatbot* responde de manera coherente, clara y respetuosa, manteniendo un estilo uniforme en sus respuestas.
   Escala de evaluación:
   - 1: las respuestas son inconsistentes y carecen de claridad o respeto.
   - 2: varias respuestas son inconsistentes o carecen de claridad.
   - 3: las respuestas son mayormente consistentes, pero con algunas excepciones.
   - 4: las respuestas son consistentes, claras y respetuosas en la mayoría de los casos.
   - 5: todas las respuestas son consistentes, claras y respetuosas.

Cada criterio puede recibir un puntaje máximo de 5 puntos, sumando un total de 25 puntos. Los resultados obtenidos se interpretan de la siguiente manera:

- **21-25 puntos**: excelente. El *chatbot* cumple con todas las expectativas.
- **16-20 puntos**: muy bueno. El *chatbot* es funcional y útil, con algunas áreas de mejora.
- **11-15 puntos**: adecuado. El *chatbot* funciona, pero requiere mejoras en precisión o claridad.
- **6-10 puntos**: insuficiente. El *chatbot* necesita ajustes importantes en la calidad de las respuestas.
- **0-5 puntos**: deficiente. El *chatbot* no cumple con los requisitos mínimos.

5.46.3. Interfaz gráfica para pruebas externas

Para facilitar las pruebas y evaluaciones del *chatbot* por expertos de la Unidad de Control Ambiental de la Municipalidad de Guatemala y otros evaluadores externos, se desarrolló una interfaz gráfica sencilla. Esta interfaz permite a los usuarios interactuar con el *chatbot* en tiempo real, proporcionando una plataforma accesible para analizar su desempeño en la clasificación de residuos y gestión de consultas.

Objetivo de la interfaz gráfica

El objetivo principal de la interfaz gráfica es ofrecer un entorno de pruebas en el que los evaluadores puedan interactuar con el *chatbot* sin necesidad de configuraciones técnicas complejas. La interfaz permite que los evaluadores envíen consultas directamente al *chatbot*, visualicen las respuestas generadas en tiempo real y observen el flujo de la conversación en un formato claro y accesible. Esta configuración simplifica el proceso de recopilación de retroalimentación sobre el desempeño del *chatbot* en términos de precisión y relevancia.

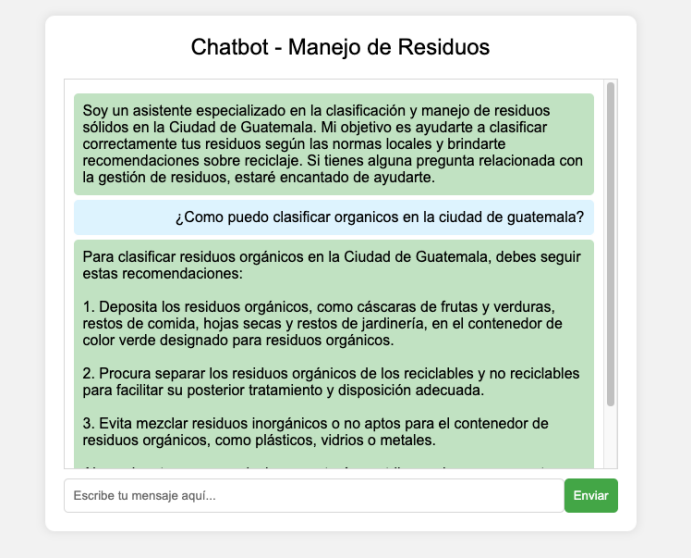


Figura 5.57. Interfaz gráfica desarrollada para pruebas del *chatbot*

Tecnologías utilizadas

La interfaz gráfica fue implementada utilizando `FastAPI` para el *backend* y `Jinja2` para la generación de plantillas *HTML*, mientras que las comunicaciones en tiempo real se gestionaron mediante *WebSocket*. Estas tecnologías ofrecen una solución ligera y eficiente, ideal para la ejecución de pruebas de conversación en tiempo real. El diseño se mantuvo simple y funcional, con elementos mínimos que facilitan la visualización de preguntas y respuestas sin distracciones.

Flujo de pruebas y evaluación

Al acceder a la interfaz gráfica, los evaluadores pueden:

- Ingresar consultas relacionadas con la clasificación de residuos y observar las respuestas del *chatbot* en tiempo real.
- Visualizar el historial de la conversación de forma organizada, lo cual facilita el análisis de la coherencia y consistencia en las respuestas del *chatbot*.
- Realizar múltiples pruebas consecutivas sin necesidad de reiniciar el sistema, lo que permite una evaluación continua del rendimiento.

La interfaz también registra cada interacción en un *log* de conversaciones, permitiendo a los desarrolladores revisar posteriormente los datos de las pruebas y analizar áreas potenciales de mejora en las respuestas del *chatbot*. Esta funcionalidad es especialmente útil para identificar patrones en las consultas y optimizar la precisión del modelo.

Conclusión sobre la interfaz gráfica

La implementación de esta interfaz gráfica facilita considerablemente el proceso de evaluación externa, permitiendo a los expertos en residuos realizar pruebas exhaustivas en un entorno controlado y de fácil acceso. La recopilación de retroalimentación mediante esta herramienta es esencial para ajustar y mejorar el rendimiento del *chatbot* en su tarea de asistencia en la clasificación de residuos en la Ciudad de Guatemala.

Análisis estadístico de la evaluación

Los datos recopilados mediante la rúbrica fueron analizados estadísticamente para obtener una métrica de precisión del *chatbot* en función de la calidad de sus respuestas. Se utilizaron medidas de tendencia central (media y mediana) para calcular el puntaje promedio de cada criterio, lo cual permitió identificar las áreas de fortaleza y las áreas de mejora.

Además, se evaluó la eficiencia del *chatbot* mediante la medición del tiempo de respuesta promedio. Este tiempo se calculó a partir de un conjunto de 10 preguntas diferentes, utilizando la siguiente fórmula:

$$\text{Tiempo promedio} = \frac{\sum_{i=1}^{n} T_i}{n} \tag{5.4}$$

donde T_i representa el tiempo de respuesta de cada pregunta y n es el número total de preguntas analizadas (en este caso, 10). Este cálculo proporciona una métrica objetiva de la eficiencia del *chatbot*, la cual es relevante en herramientas de asistencia en tiempo real.

5.46.4. Conclusión de la metodología de evaluación

La metodología de evaluación mediante una rúbrica estandarizada y el análisis estadístico de los resultados proporciona una medida objetiva del rendimiento del *chatbot*. Este enfoque permite validar que el *chatbot* cumpla con las expectativas de precisión, relevancia y consistencia en la clasificación de residuos, asegurando su aplicabilidad en el contexto de la Ciudad de Guatemala.

CAPÍTULO 6

Resultados

6.1. Resultados de experiencia de usuario

A lo largo de las múltiples etapas por las que pasó el desarrollo del proyecto, se pudo observar que varias de ellas producían resultados importantes para la toma de decisiones a lo largo del mismo desarrollo del prototipo, aplicación en código y validaciones finales del proyecto. Cada conjunto de resultados obtenido por una sección del desarrollo de la metodología del proyecto produjo información de suma importancia para la continuación de la siguiente etapa a trabajar. Entre las secciones que brindaron resultados importantes se encuentran la investigación inicial de usuarios, prototipado iterativo de la solución, y desarrollo de la interfaz gráfica de la aplicación. Finalmente, también se generaron resultados finales probando y validando la aplicación finalizada en código, los cuales también fueron extremadamente importantes para la generación de material para discusión y conclusiones respecto al proyecto.

6.1.1. Investigación inicial de usuarios

La investigación inicial de usuarios realizada consistió en dos encuestas de *Google Forms* que se mencionaron anteriormente, las cuales tenían como objetivo recaudar información sobre conocimientos y comportamiento actual sobre la gestión de residuos por parte de los potenciales usuarios de la aplicación, y también información sobre necesidades y expectativas para la aplicación por parte de los mismos potenciales usuarios. Según cálculos realizados, para la obtención de información significativa eran necesarias aproximadamente 384 respuestas a cada una de las encuestas. Se logró obtener 448 respuestas en la primera encuesta y 427 respuestas en la segunda, por lo que los resultados de las mismas representan efectivamente la población que constituye el público objetivo de la aplicación.

Resultados de la investigación sobre conocimientos y comportamientos actuales sobre la gestión de residuos de los usuarios

A continuación se presentan los resultados provistos por parte de las respuestas a la primera encuesta elaborada, la cual tuvo como objetivo obtener resultados sobre conocimientos y comportamientos en lo que respecta la gestión de residuos por parte de la muestra establecida de usuarios potenciales de la aplicación.

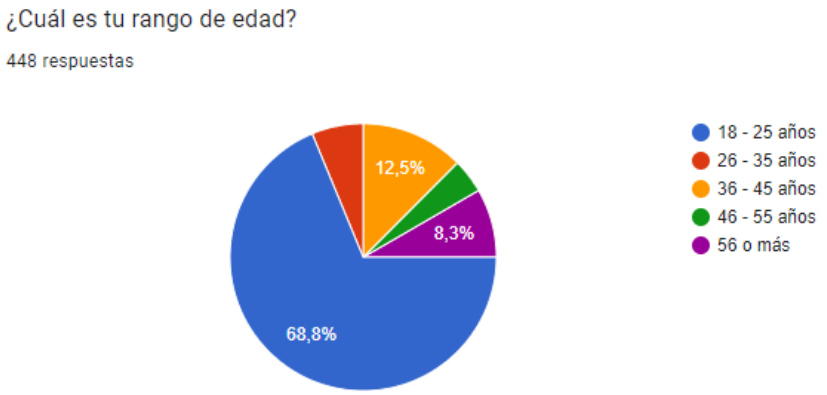


Figura 6.1. Resultados a la pregunta: ¿Cuál es tu rango de edad?

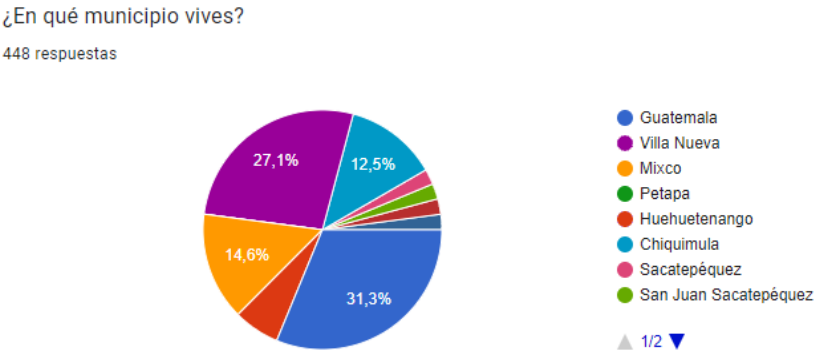


Figura 6.2. Resultados a la pregunta: ¿En qué municipio vives?

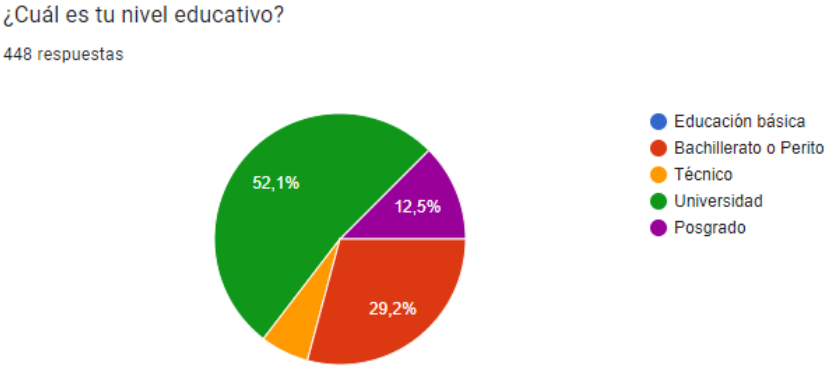


Figura 6.3. Resultados a la pregunta: ¿Cuál es tu nivel educativo?

Figura 6.4. Resultados a la pregunta: ¿Cuánto sabes sobre la clasificación de residuos?

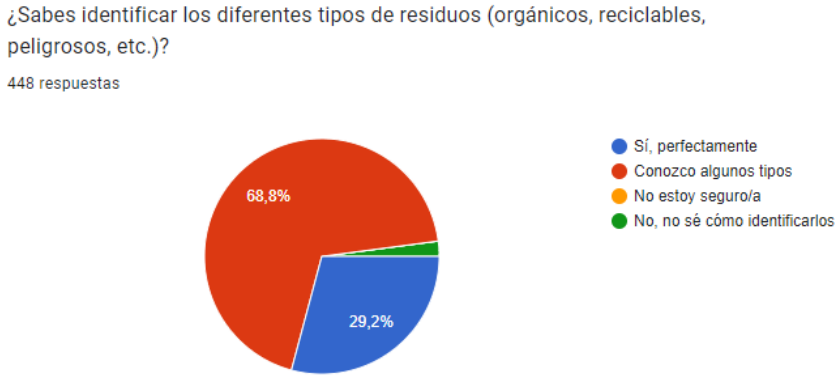


Figura 6.5. Resultados a la pregunta: ¿Sabes identificar los diferentes tipos de residuos (orgánicos, reciclables, peligrosos, etc.)?

Figura 6.6. Resultados a la pregunta: ¿Estás familiarizado/a con los símbolos de reciclaje en los productos?

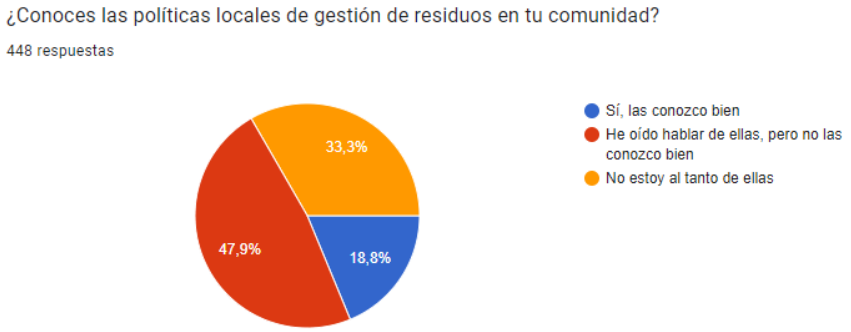


Figura 6.7. Resultados a la pregunta: ¿Conoces las políticas locales de gestión de residuos en tu comunidad?

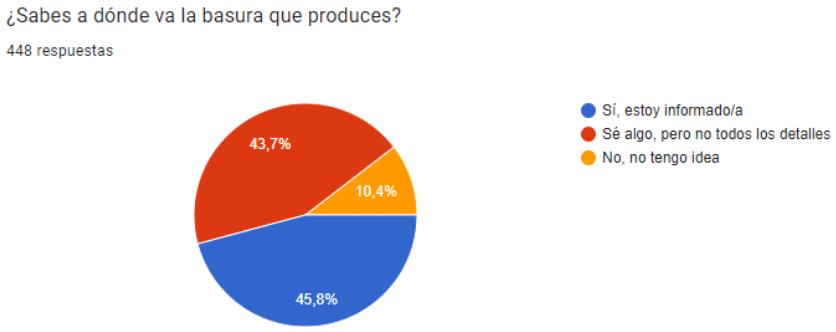


Figura 6.8. Resultados a la pregunta: ¿Sabes a dónde va la basura que produces?

Figura 6.9. Resultados a la pregunta: ¿Con qué frecuencia reciclas?

Figura 6.10. Resultados a la pregunta: ¿Tienes un sistema de clasificación de residuos en tu hogar?

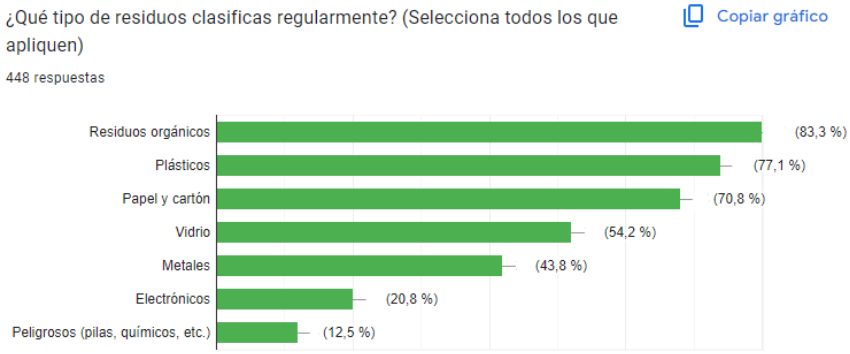


Figura 6.11. Resultados a la pregunta: ¿Qué tipo de residuos clasificas regularmente?

Figura 6.12. Resultados a la pregunta: ¿Utilizas alguna aplicación o recurso digital para ayudarte en la gestión de residuos?

Figura 6.13. Resultados a la pregunta: ¿Qué es lo más desafiante al momento de gestionar tus recursos?

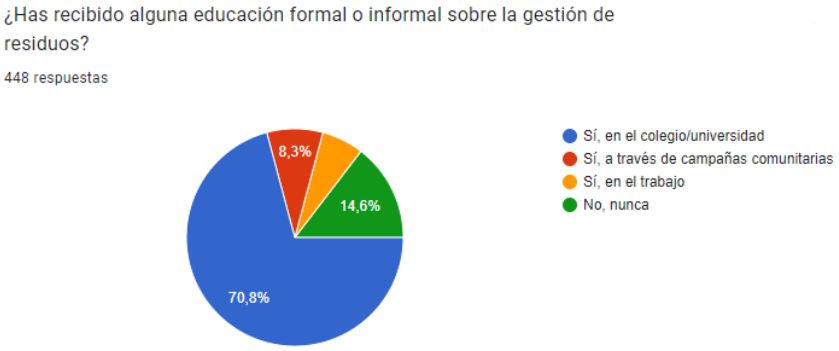


Figura 6.14. Resultados a la pregunta: ¿Has recibido alguna educación formal o informal sobre la gestión de residuos?

Figura 6.15. Resultados a la pregunta: ¿Qué tan importante crees que es reciclar y gestionar los residuos correctamente?

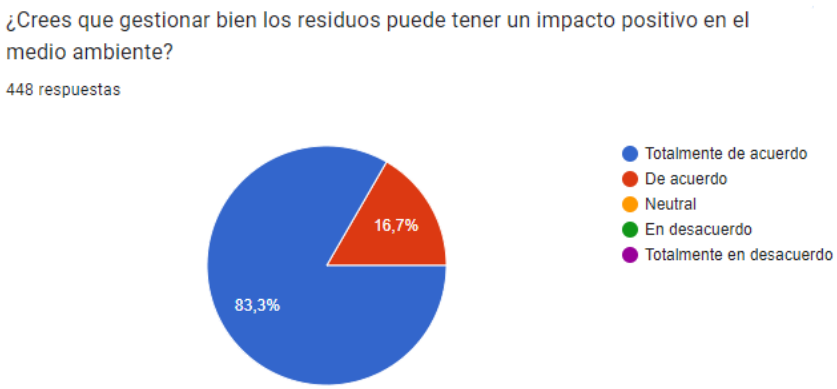


Figura 6.16. Resultados a la pregunta: ¿Crees que gestionar bien los residuos puede tener un impacto positivo en el medio ambiente?

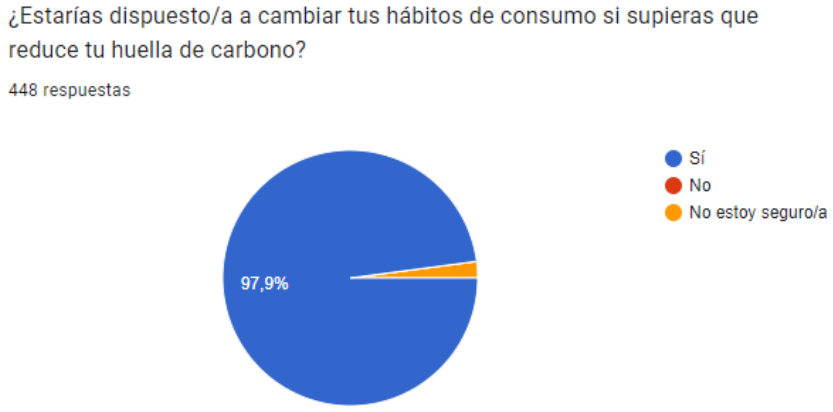


Figura 6.17. Resultados a la pregunta: ¿Estarías dispuesto/a a cambiar tus hábitos de consumo si supieras que reduce tu huella de carbono?

Resultados de la investigación sobre necesidades y expectativas de los usuarios sobre la aplicación

Por otro lado, también se obtuvieron los siguientes resultados en lo que respecta a la investigación sobre necesidades y expectativas por parte de los potenciales usuarios sobre la aplicación a desarrollar.

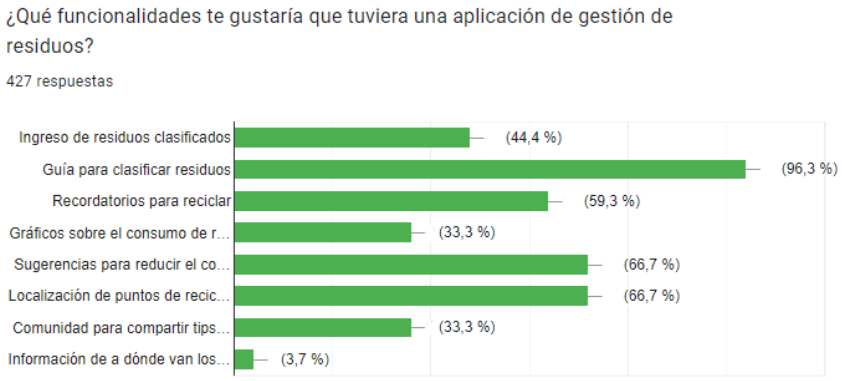


Figura 6.18. Resultados a la pregunta: ¿Qué funcionalidades te gustaría que tuviera una aplicación de gestión de residuos?

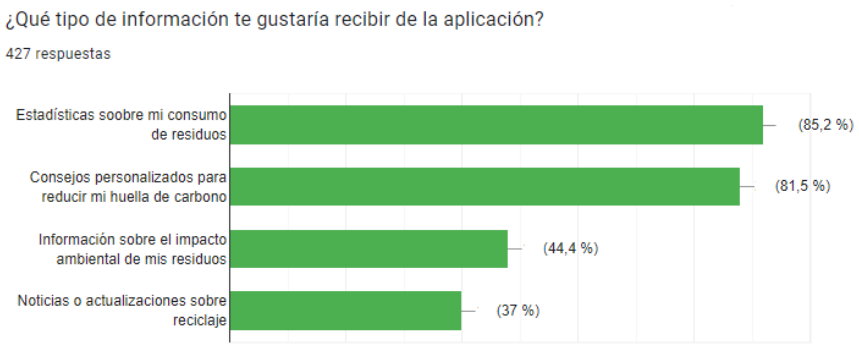


Figura 6.19. Resultados a la pregunta: ¿Qué tipo de información te gustaría recibir de la aplicación?

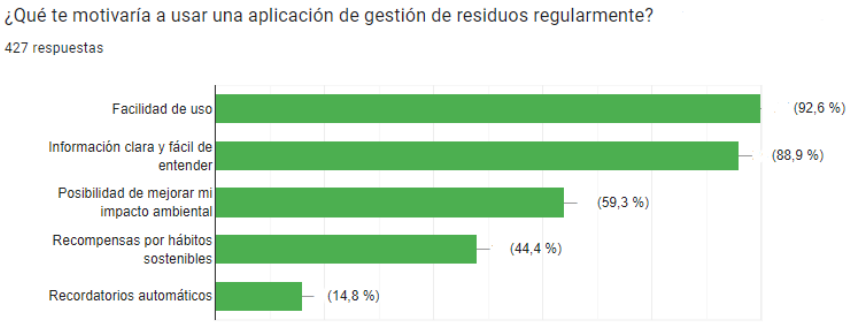


Figura 6.20. Resultados a la pregunta: ¿Qué te motivaría a usar una aplicación de gestión de residuos regularmente?

Figura 6.21. Resultados a la pregunta: ¿Qué tan importante es para ti que la aplicación sea fácil de usar?

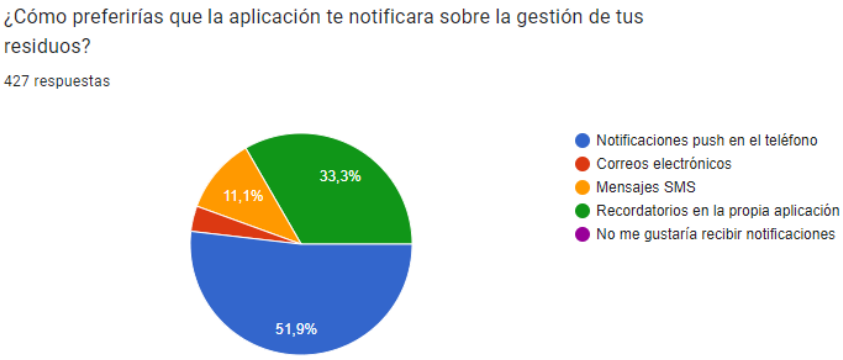


Figura 6.22. Resultados a la pregunta: ¿Cómo preferirías que la aplicación te notificara sobre la gestión de tus residuos?

Figura 6.23. Resultados a la pregunta: ¿Qué tan importante es la personalización en la experiencia de la aplicación?

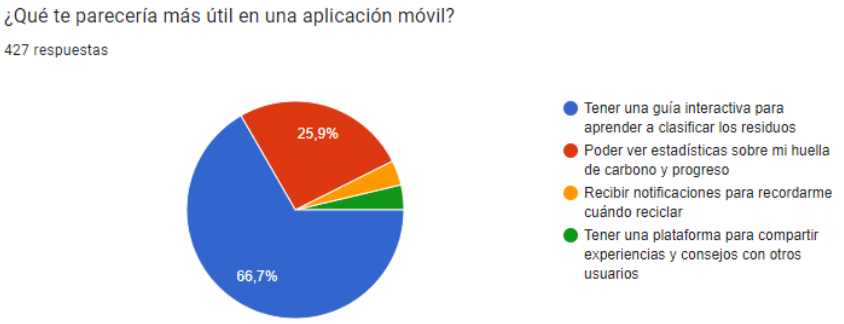


Figura 6.24. Resultados a la pregunta: ¿Qué te parecería más útil en una aplicación móvil?

Figura 6.25. Resultados a la pregunta: ¿Qué tan importante es para ti que la aplicación tenga un diseño visualmente atractivo?

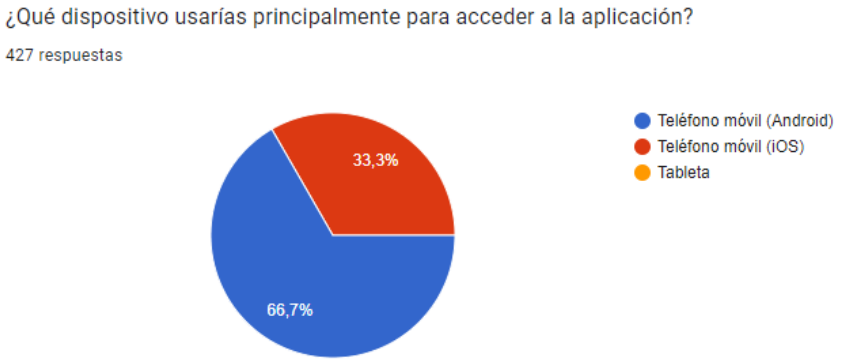


Figura 6.26. Resultados a la pregunta: ¿Qué dispositivo usarías principalmente para acceder a la aplicación?

6.1.2. Desarrollo iterativo del prototipo

Con los perfiles de los potenciales usuarios identificados correctamente, en conjunto de la investigación, recabación y análisis de resultados detallados en la sección anterior, fue posible el desarrollo iterativo del prototipo de alta fidelidad de la solución planteada. Como se detalló anteriormente, los resultados provienen tanto de encuestas realizadas durante cada iteración, como de pruebas con usuarios puntuales. Las encuestas fueron respondidas por 289 personas en la primera iteración, 314 personas en la segunda iteración, y finalmente 295 personas en la tercera y última iteración. También se evaluó puntualmente a 5 personas, una que entraba en cada uno de los perfiles definidos anteriormente.

Resultados de la primera iteración del prototipo

La siguiente figura indica los resultados en lo que respecta al puntaje promedio recibido por cada una de las pantallas de la aplicación que fueron colocadas en la encuesta. Estas pantallas, de manera respectiva, son la pantalla inicial, la pantalla de inicio de sesión, el menú principal, la pantalla de configuración de perfil, la pantalla de analíticas, la pantalla de ingreso de datos, la pantalla de conversaciones con el modelo para dudas sobre clasificación y la pantalla para conversar con el modelo en sí.

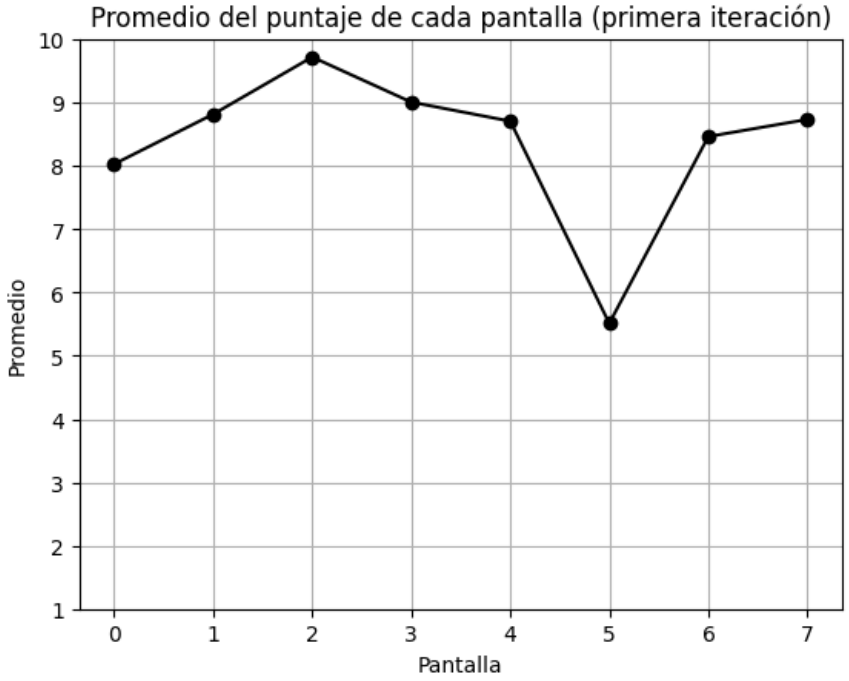


Figura 6.27. Resultados de las calificaciones de la primera iteración del prototipo

Al mismo tiempo, junto con los resultados del puntaje de cada pantalla presentada, también se tienen como resultados múltiples comentarios para cada una de estas pantallas. A continuación se incluye un cuadro con varios de los comentarios más frecuentes para cada una de las pantallas presentadas, ya sean opiniones positivas o sugerencias de cambios.

Pantalla	Comentarios
Pantalla inicial	En general los comentarios a la pantalla fueron muy positivos. Sin embargo, una sugerencia que se recibió fue que el botón para ingresar se encontraba muy abajo en la pantalla.
Pantalla de inicio de sesión	Muy buenos comentarios. Se recibieron sugerencias sobre cómo los botones son del mismo color que el fondo de la pantalla, lo que puede llegar a ser un poco confuso.
Pantalla de menú principal	La pantalla principal fue la que tuvo mejores calificaciones con diferencia. Se recibieron comentarios sobre cómo la información desplegada era compacta y fácil de comprender.
Pantalla de configuración de perfil	Buenos comentarios. Se comentó de forma positiva la distribución de los botones.
Pantalla de analíticas de consumo	Buenos comentarios en general. Se recibieron comentarios que indicaban que no se comprendía el objetivo del botón .ªceptar".
Pantalla de ingreso de datos	Con diferencia la pantalla peor recibida. Se comentó que era complicado entender el flujo de la inserción de información debido a la distancia entre los campos para ingreso de información y los íconos también resultaban confusos.
Pantalla de chats con el modelo	Se recibieron comentarios positivos. Una de las sugerencias recibidas consistió en explicar qué significan los botones de la parte de arriba de la pantalla.
Pantalla para conversar con el modelo	Los comentarios fueron muy positivos. Se comentó la similitud con otras pantallas de aplicaciones para modelos similares.

Cuadro 6.1. Comentarios y retroalimentación sobre la primera iteración del prototipo

Resultados de la segunda iteración del prototipo

En la segunda iteración al prototipo se hicieron varios de los cambios sugeridos durante la retroalimentación recibida con respecto a la primera iteración del prototipo. Los resultados de la encuesta durante la segunda iteración fueron los siguientes.

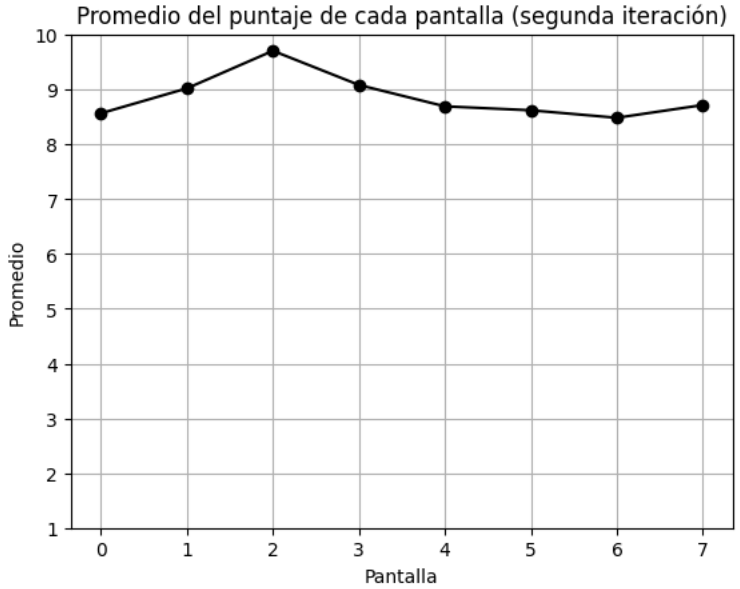


Figura 6.28. Resultados de las calificaciones de la primera iteración del prototipo

Junto con los resultados obtenidos por parte de la encuesta respectiva a la segunda iteración del prototipo, también se obtuvieron nuevos comentarios respecto a los cambios que se implementaron en las pantallas nuevas. Algunos de estos comentarios se encuentran resumidos en el siguiente cuadro.

Pantalla	**Comentarios**
Pantalla inicial	Se obtuvieron más comentarios positivos. No hubo mención de posibles mejoras.
Pantalla de inicio de sesión	Los comentarios volvieron a ser muy positivos. Se sugirió dejar más espacio entre algunos elementos.
Pantalla de menú principal	Esta pantalla volvió a ser la que mejor se recibió. No hubieron mejoras notables sugeridas.
Pantalla de configuración de perfil	Se recibieron buenos comentarios. Se sugirió cambiar el color de botones que hacen acciones importantes como cerrar sesión.
Pantalla de analíticas de consumo	Se obtuvieron comentarios muy positivos y no se comentó más sobre botones sin un sentido luego de los cambios.
Pantalla de ingreso de datos	Los comentarios sobre esta pantalla fueron mucho más positivos. Se sugirió agrandar un poco el botón para subir una fotografía del desecho a reportar.
Pantalla de chats con el modelo	Se recibieron comentarios muy positivos. No hubieron mejoras notables sugeridas.
Pantalla para conversar con el modelo	Se volvieron a recibir comentarios positivos sobre la similitud a otras aplicaciones.

Cuadro 6.2. Comentarios y retroalimentación sobre la segunda iteración del prototipo

Resultados de la tercera iteración del prototipo

Finalmente, la tercera iteración produjo resultados muy positivos que afirmaron que la usabilidad y la estética del prototipo estaba lista para ser llevada a la etapa de desarrollo. A continuación se encuentran los resultados obtenidos para las pantallas.

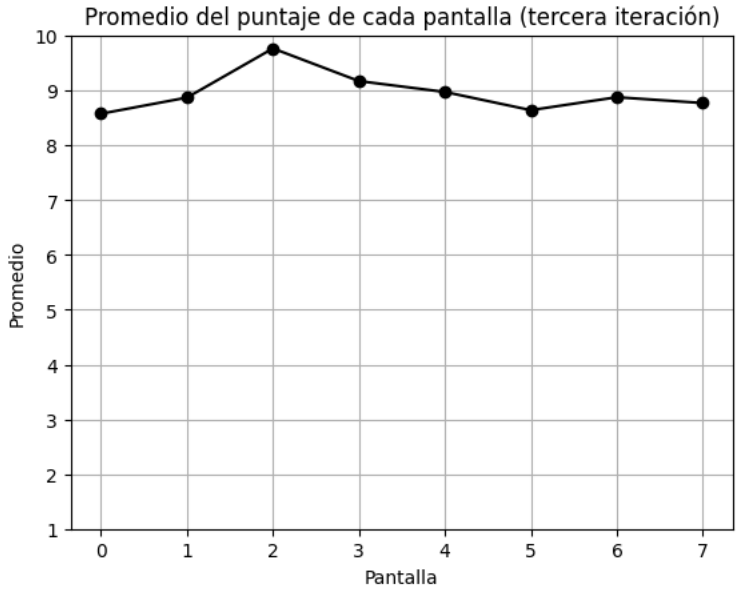


Figura 6.29. Resultados de las calificaciones de la primera iteración del prototipo

Los comentarios obtenidos sobre el resultado de la tercera iteración al prototipo también fueron muy positivos. A continuación se encuentra un cuadro con los comentarios recibidos.

Pantalla	**Comentarios**
Pantalla inicial	Se obtuvieron comentarios positivos y de satisfacción respecto al resultado final de la pantalla.
Pantalla de inicio de sesión	Los comentarios fueron muy positivos y no hubieron sugerencias particulares.
Pantalla de menú principal	La pantalla tuvo un excelente recibimiento y no hubieron posibles mejoras planteadas.
Pantalla de configuración de perfil	Se recibieron comentarios muy positivos, especialmente respecto al cambio de color de botones de acciones como cierre de sesión.
Pantalla de analíticas de consumo	Los comentarios volvieron a ser muy positivos. Sin embargo esta pantalla sí tuvo como posible mejora agregar más gráficos que permitan visualizar el consumo de basura de los usuarios.
Pantalla de ingreso de datos	La mejora en la pantalla fue notoria, los comentarios recibidos fueron muy positivos y no se sugirió algún cambio. Se comentó positivamente la simplicidad de la pantalla.
Pantalla de chats con el modelo	Los comentarios recibidos fueron positivos y no se plantearon posibles mejoras.
Pantalla para conversar con el modelo	Se volvieron a recibir comentarios muy positivos que apreciaban lo intuitivo de la pantalla.

Cuadro 6.3. Comentarios y retroalimentación sobre la tercera iteración del prototipo

6.1.3. Evaluación final a la aplicación funcional

Los resultados de las pruebas de usabilidad revelan que el tiempo que los usuarios requieren para completar ciertas acciones en la aplicación varía. Crear una cuenta tomó un promedio de 54.37 segundos. Iniciar sesión fue relativamente rápido, con un promedio de 50.86 segundos, y registrar un consumo de basura fue aún más ágil, completándose en 43.11 segundos en promedio. Formular una pregunta al *chatbot* tomó 62.09 segundos. En cuanto a la personalización del perfil, cambiar la foto de perfil fue una acción sencilla, con un promedio de 39.30 segundos, mientras que agregar una dirección fue la tarea más prolongada, con 68.44 segundos. Observar las analíticas del día fue la acción más rápida, con un promedio de 20.16 segundos. Estos resultados destacan las fortalezas y las áreas de mejora para lograr una experiencia de usuario más fluida. A continuación se presenta un cuadro con los tiempos promedio de cada una de las acciones realizadas en las pruebas de usabilidad. En cuestión a comentarios y resultados cualitativos, los comentarios por parte de los usuarios que pudieron probar la aplicación fueron muy positivos, destacando cómo la mayoría de acciones no tomaba mucho tiempo y las sesiones en promedio de uso de la aplicación eran rápidas.

Acción	**Tiempo Promedio (segundos)**
Crear una cuenta	54.37
Iniciar sesión	50.86
Registrar basura	43.11
Preguntar al *chatbot*	62.09
Cambiar foto de perfil	39.30
Agregar dirección	68.44
Observar analíticas	20.16

Cuadro 6.4. Resultados de las pruebas de usabilidad

Por otro lado también se obtuvieron resultados por parte del proceso de la realización de las pruebas utilizando la técnica de *eye tracking*. Dependiendo de las acciones a realizar, es posible observar cómo las miradas de los usuarios se dirigen a los lugares exactos que forman parte del procedimiento de las acciones a realizar. Durante las pruebas realizadas se pudo observar cómo los mapas de calor generados por el *eye tracking* mostraban que la gran mayoría de interacción por parte de los usuarios con las pantallas se realizaba directamente a los botones que eran necesarios para realizar las acciones solicitadas. También es posible observar cómo las miradas también se enfocaban a múltiples elementos complementarios como gráficos e incluso el teclado del emulador utilizado para las pruebas.

Figura 6.30. Resultados del *eye tracking* en pantallas como la principal y la de configuración de perfil

6.2. Resultados de *backend*, gestión de base de datos y análisis de datos

El desarrollo del módulo se completó en las fases establecidas en la metodología, y cada iteración contribuyó a obtener resultados concretos que demuestran la funcionalidad del API para gestionar, proporcionar y transformar la información de los usuarios.

A continuación, se presentan los resultados obtenidos en cada apartado:

6.2.1. Diseño y estructura de la base de datos

A través de las iteraciones, se logró definir una estructura final optimizada para gestionar los datos de los usuarios, desechos y ubicaciones relacionadas. Como resultado final, se establecieron las siguientes siete tablas principales:

- ***User***: guarda la información personal de los usuarios.
- ***Area***: almacena la información de las zonas disponibles para las ubicaciones.
- ***Location***: registra las ubicaciones con respecto a la zona y el usuario asociado.
- ***Waste Type***: almacena los tipos de desechos y sus propiedades para el análisis del impacto ambiental.
- ***Waste***: relaciona el tipo de desecho y guarda información detallada sobre un desecho específico.
- ***User Waste***: registra todos los datos de los residuos gestionados por el usuario.
- ***Audit Log***: registra todas las operaciones realizadas de creación, actualización y eliminación en la base de datos.

A continuación, se muestra el diagrama final de la estructura de la base de datos:

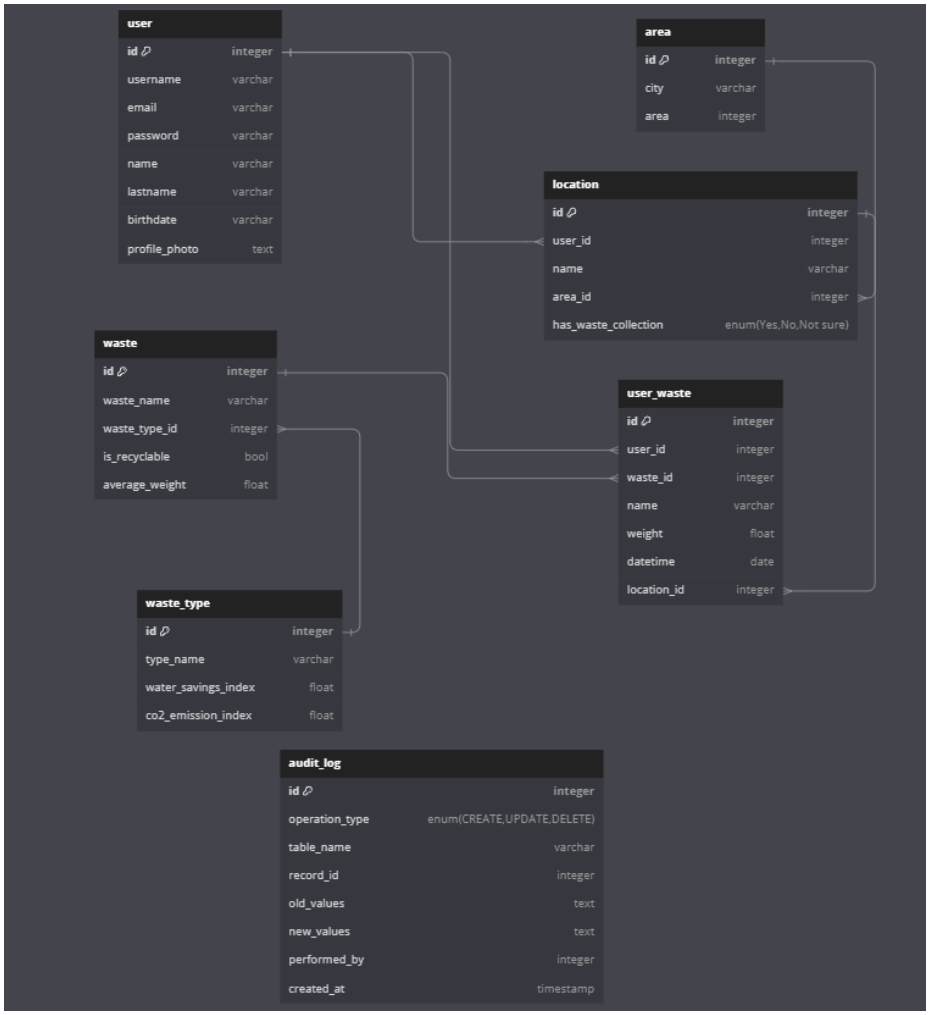


Figura 6.31. Diagrama entidad-relación final

6.2.2. *API*

A lo largo de las iteraciones, se desarrollaron los módulos del *API*, cada uno contribuyendo a una gestión eficiente de los datos y asegurando que el *API* responda adecuadamente a las solicitudes de información.

Los módulos finales son los siguientes:

- ***analysis***: proporciona las funciones y consultas necesarias para el análisis de datos del sistema.
- ***config***: gestiona la conexión a la base de datos, garantizando un acceso estable y seguro.
- ***controllers***: contiene las funciones principales que ejecutan las operaciones en cada una de las tablas de la base de datos.
- ***middleware***: incluye funciones auxiliares que facilitan el funcionamiento del *API*, tales como el monitoreo y la autenticación de usuarios.
- ***models***: define los modelos que representan la estructura de las tablas en la base de datos.

- ***routes***: gestiona las rutas que exponen la información del *API*, permitiendo acceder a los recursos de manera estructurada.
- **test**: almacena las pruebas unitarias, de integración y de rendimiento, que verifican la funcionalidad y estabilidad del sistema.
- ***utils***: contiene funciones auxiliares, como la encargada de registrar las entradas en la bitácora de auditoría.

Resultados de pruebas de funcionalidad

Se realizaron pruebas de funcionalidad para verificar el correcto desempeño del API en distintas operaciones. Los resultados de algunas de las rutas principales para el funcionamiento del *API* se muestran a continuación:

- **Registro de usuarios**:

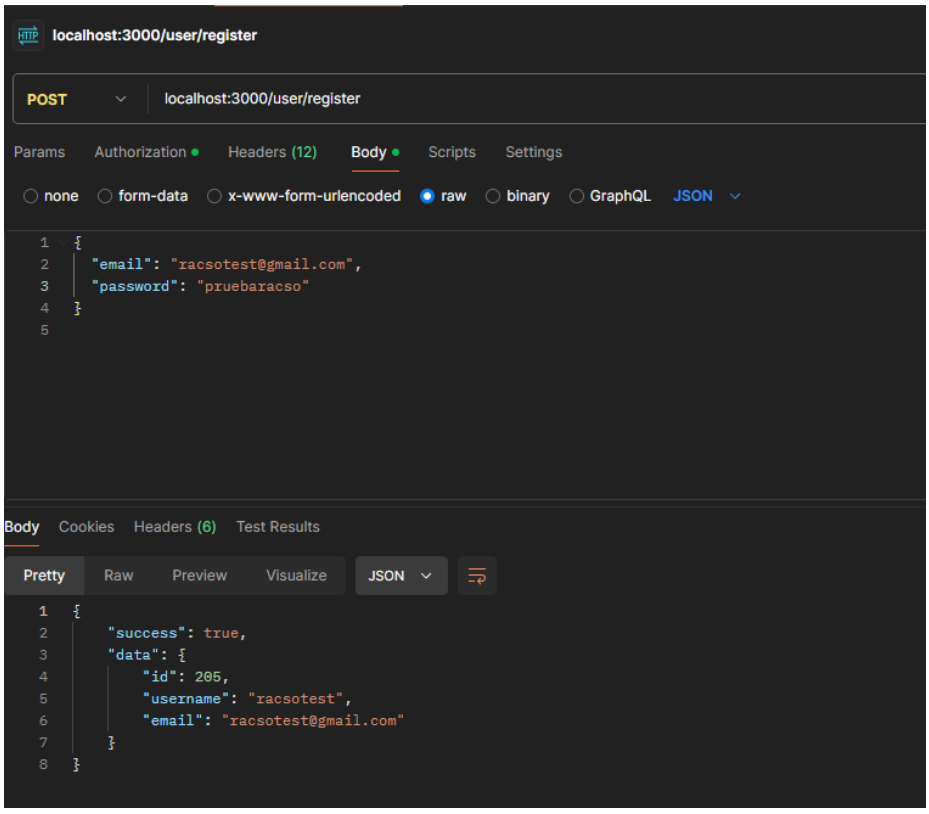


Figura 6.32. Registro de usuarios

- ***Login*** **de usuarios**:

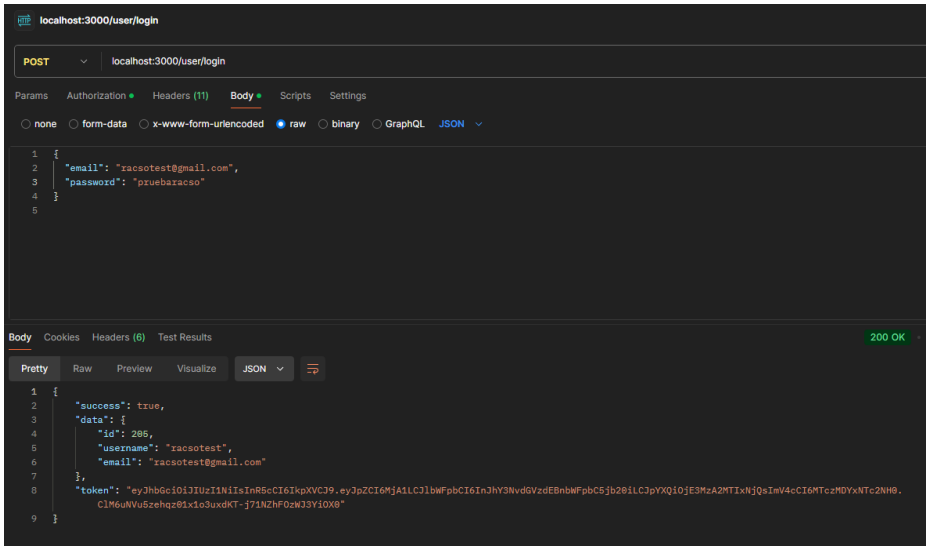


Figura 6.33. *Login* de usuarios

- **Registro en la bitácora**:

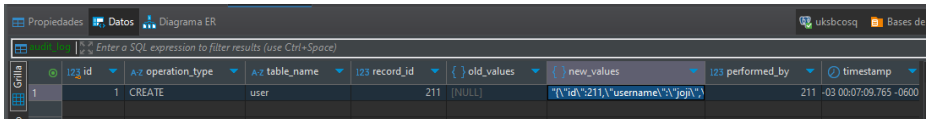


Figura 6.34. Registro en la bitácora de auditoría

- **Acceso no autorizado**:

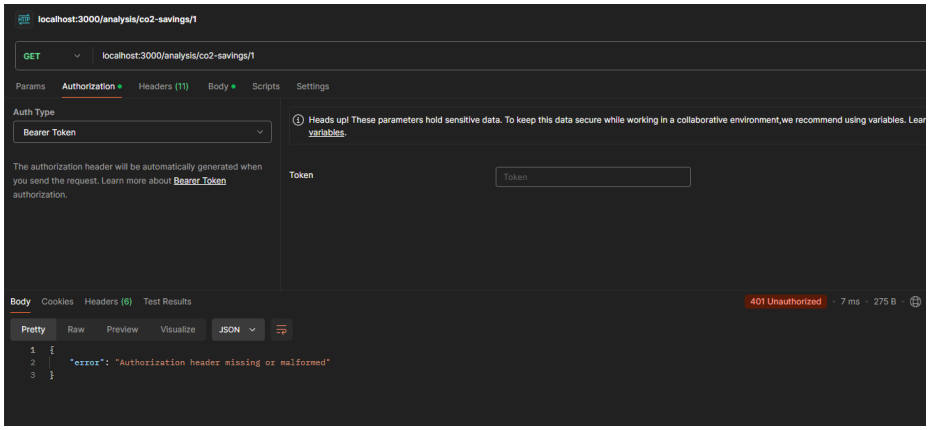


Figura 6.35. Intento de acceso no autorizado

- **Consulta de análisis**:

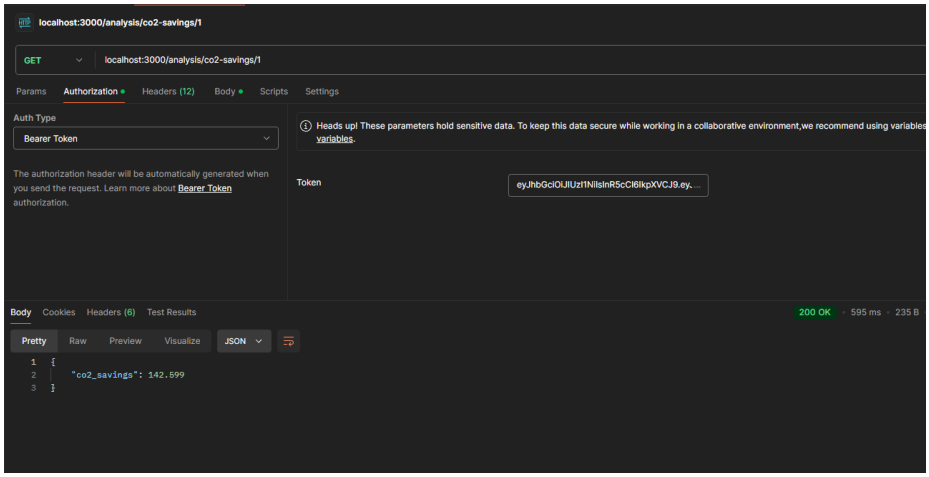


Figura 6.36. Consulta de análisis para usuarios autorizados

Pruebas unitarias y de integración

Con el objetivo de asegurar el correcto funcionamiento de todas las rutas y funcionalidades del *API*, se llevaron a cabo pruebas unitarias y de integración. Estas pruebas permitieron validar cada componente tanto de forma individual como en conjunto, garantizando que el sistema funcione conforme a lo esperado. Las pruebas se aplicaron directamente a los controladores de todos los modelos y a todas las rutas que proporciona el *API*.

En total, se realizaron 220 pruebas, todas las cuales se ejecutaron exitosamente, confirmando la efectividad y robustez del sistema. A continuación, se presentan los resultados detallados:

Figura 6.37. *Tests* realizados al *API*

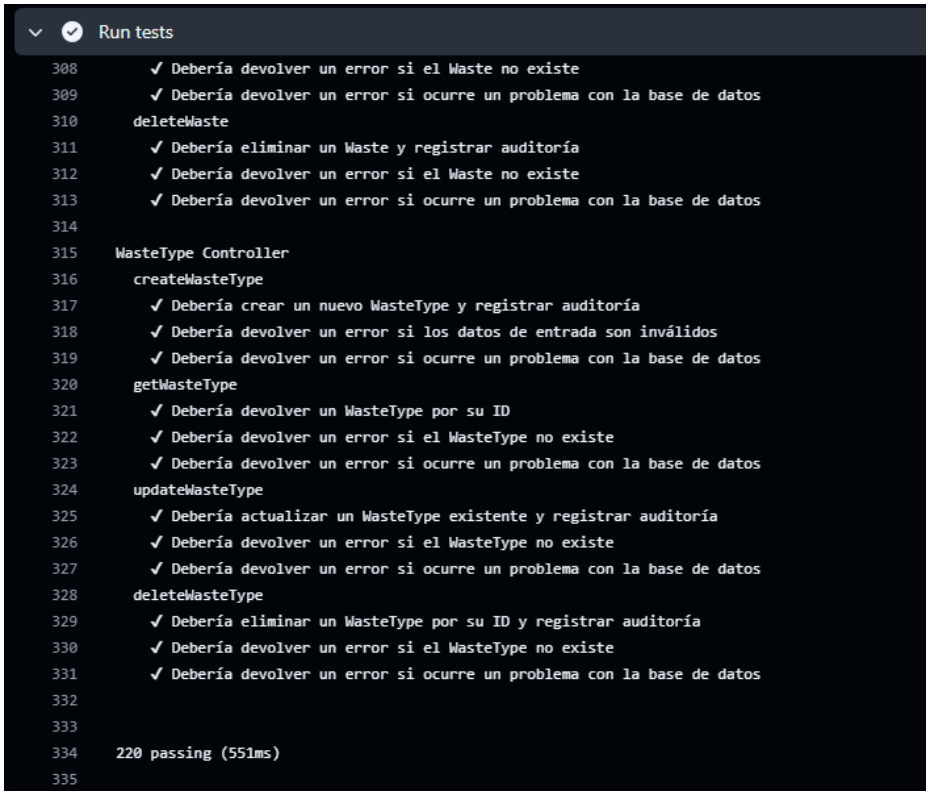


Figura 6.38. Resultado de los *tests* acertados

Pruebas de carga y rendimiento

Durante la fase de pruebas, se llevaron a cabo evaluaciones de carga y rendimiento para medir la capacidad de respuesta del *API* ante diferentes volúmenes de solicitudes. Estos escenarios permitieron observar el comportamiento y los tiempos de respuesta del sistema bajo distintas condiciones de carga.

Es importante destacar que las pruebas de rendimiento se enfocaron especialmente en las consultas de análisis de datos, ya que estas suelen presentar tiempos de respuesta más elevados debido a la necesidad de realizar búsquedas complejas en múltiples tablas. Estas consultas fueron seleccionadas por ser las más representativas del uso intensivo del sistema y por su relevancia en la evaluación de la capacidad del *API* para gestionar grandes volúmenes de datos.

En base a los cálculos de la metodología, se determinó que un 20 % de la población de la Ciudad de Guatemala generaría aproximadamente 14 solicitudes por segundo. Se realizó una prueba con una carga mayor, alcanzando 20 solicitudes por segundo, a continuación se presentan los resultados de la la prueba de rendimiento durante 1 minuto:

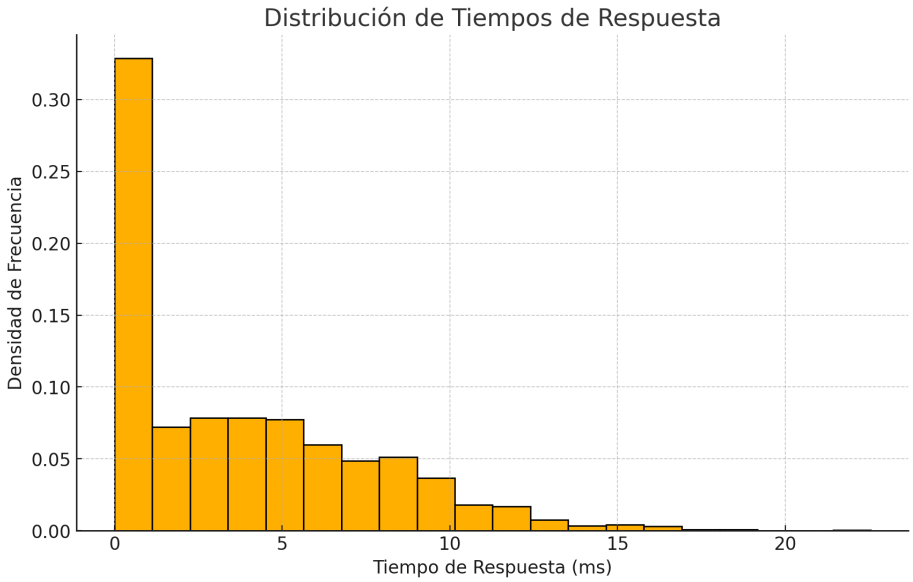


Figura 6.39. Resultados prueba de carga del 20 %

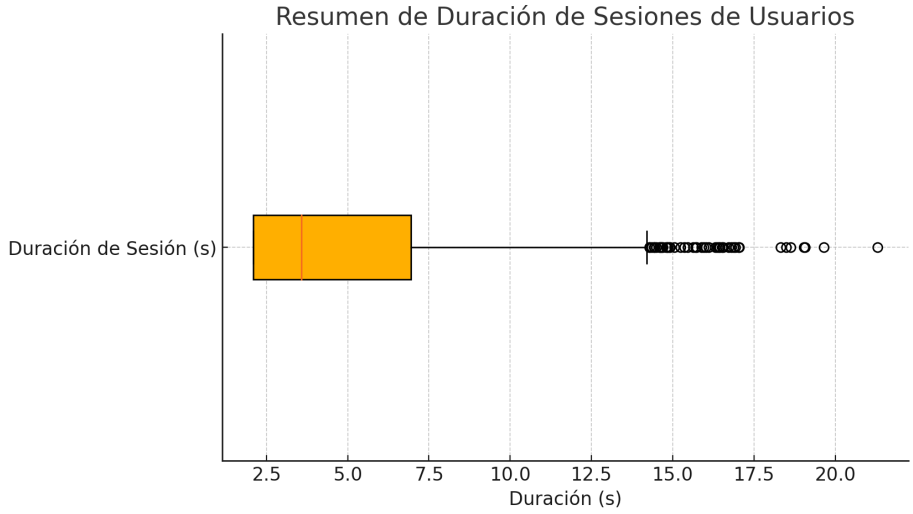


Figura 6.40. Resultados prueba de carga del 20 %

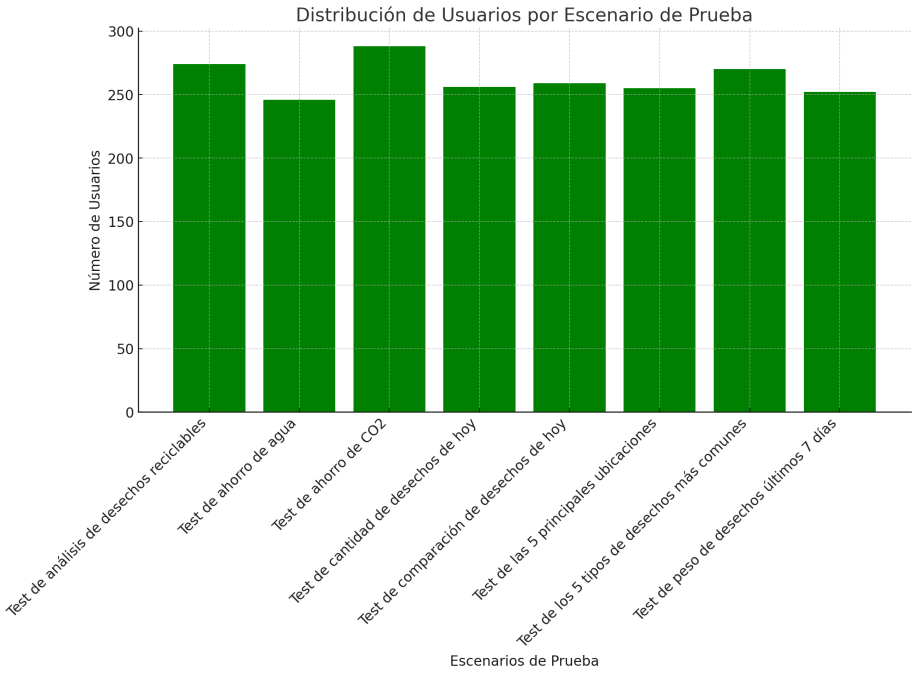


Figura 6.41. Resultados prueba de carga del 20 %

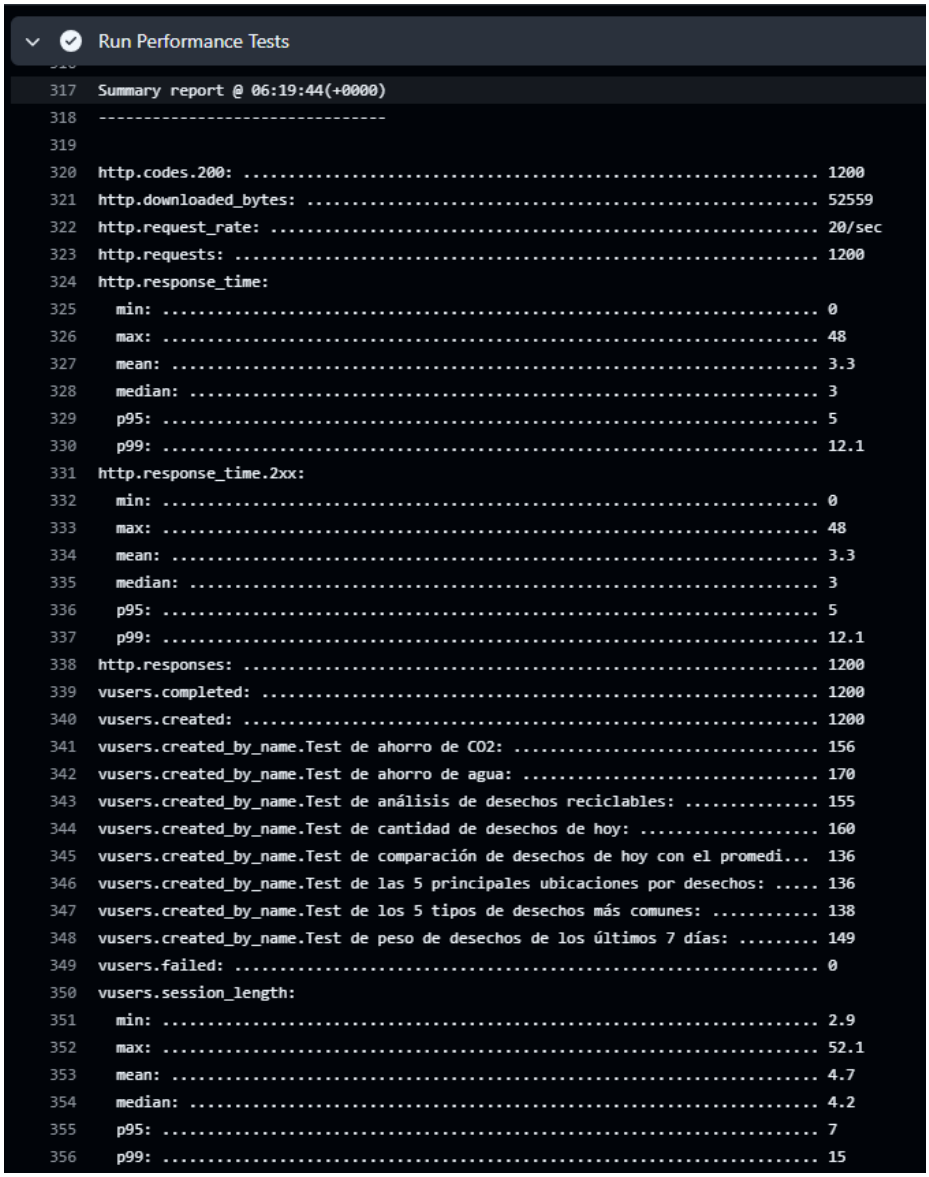


Figura 6.42. Resultados prueba de carga del 20 %

Además, se calculó que para un 50 % de la población de la Ciudad de Guatemala, el *API* debería soportar alrededor de 35 solicitudes por segundo. Los resultados de esta prueba también se analizaron para comprender el rendimiento del *API* bajo una carga considerable. Se presentan a continuación los resultados de la prueba de rendimiento durante 1 minuto:

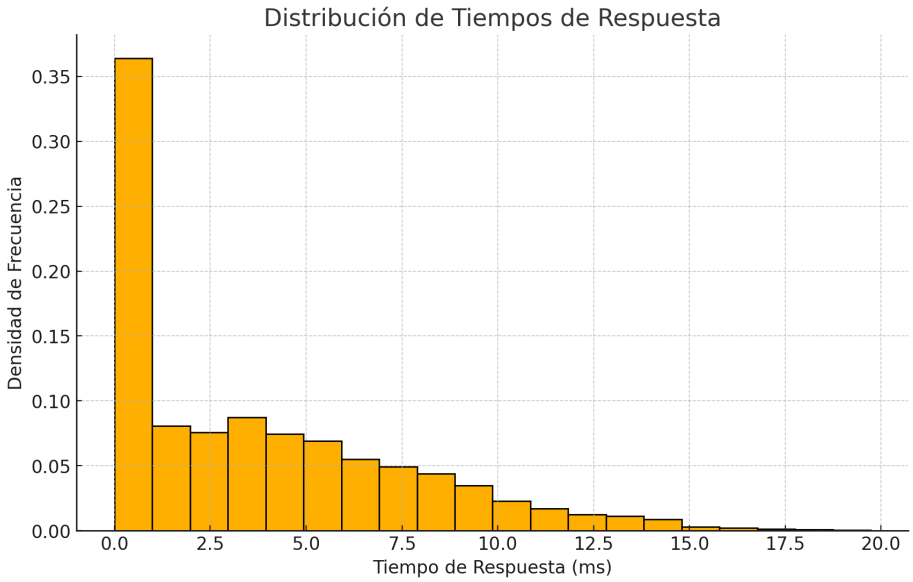


Figura 6.43. Resultados prueba de carga del 50 %

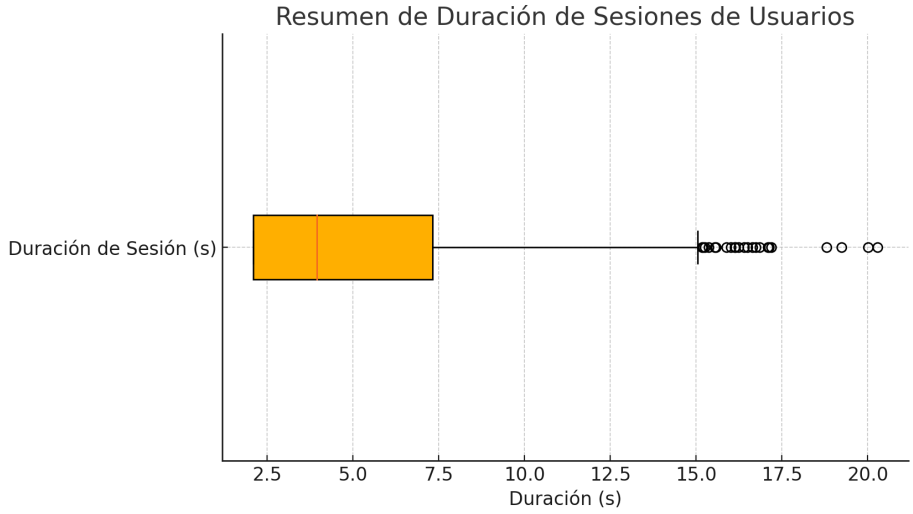


Figura 6.44. Resultados prueba de carga del 50 %

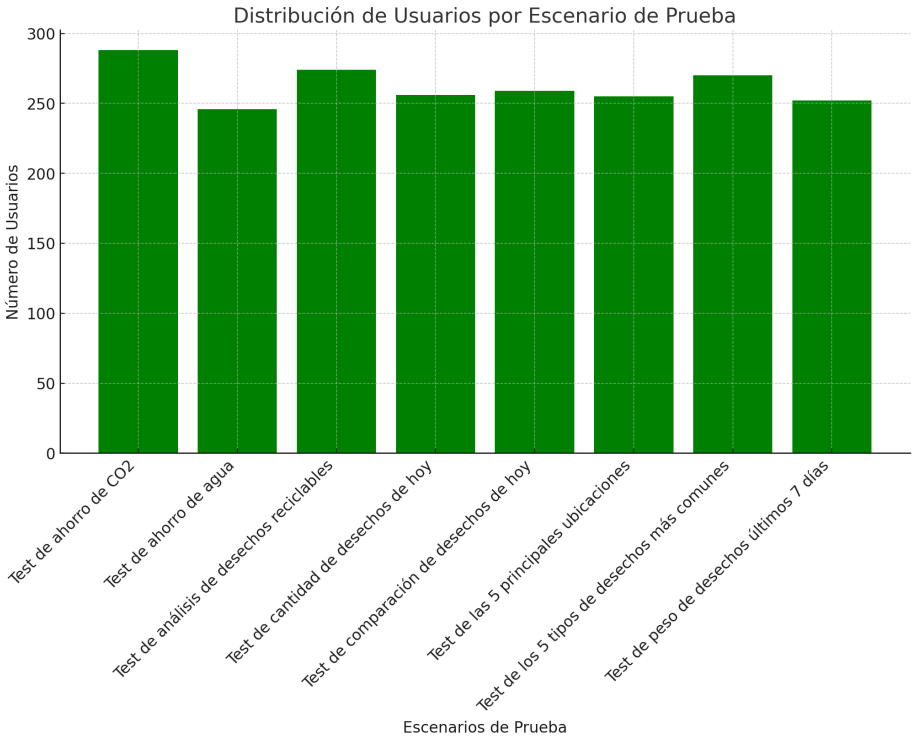


Figura 6.45. Resultados prueba de carga del 50 %

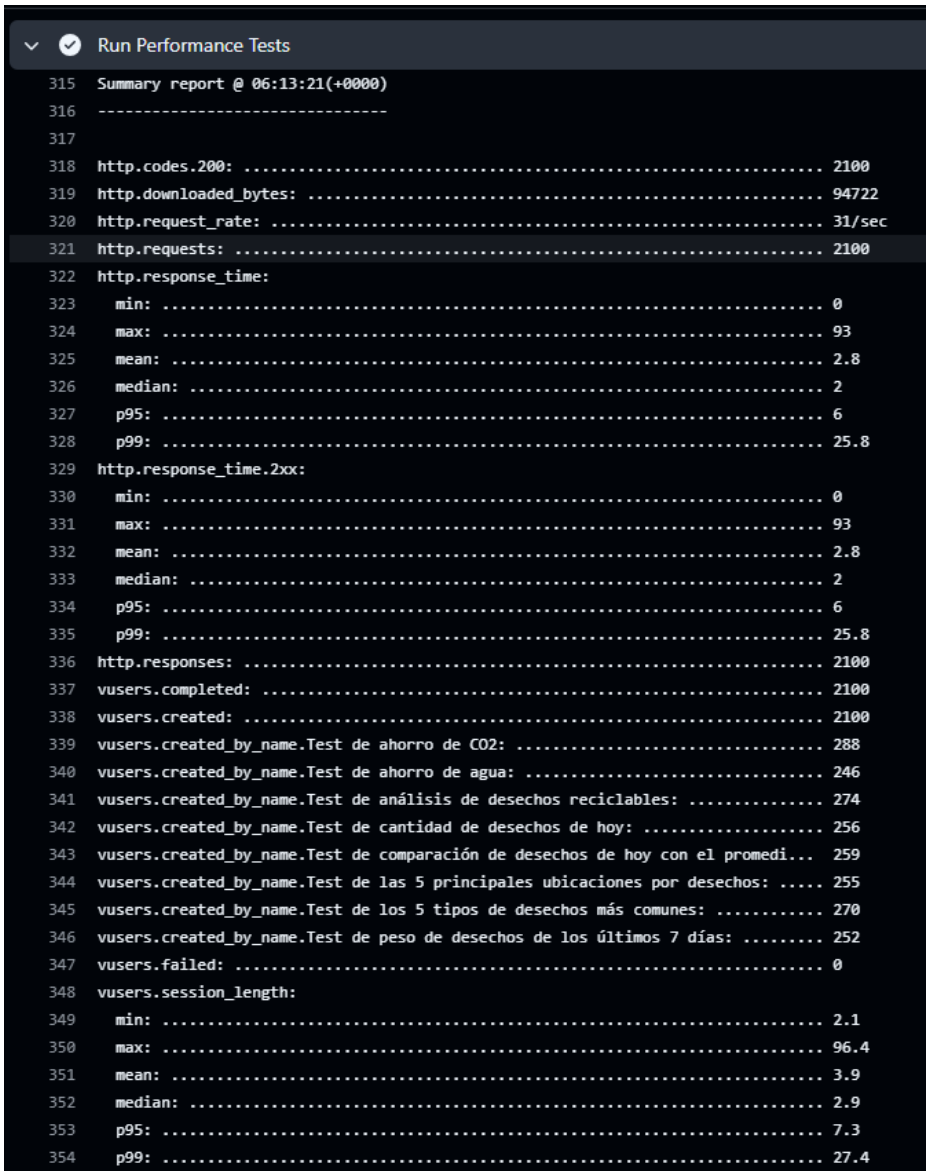


Figura 6.46. Resultados prueba de carga del 50 %

Por último, se llevó a cabo una prueba de carga para evaluar los tiempos de respuesta del *API*, simulando un escenario en el que el 100 % de la población de la Ciudad de Guatemala interactúa con el sistema. De acuerdo con los cálculos, esto representa aproximadamente 70 solicitudes por segundo. Los resultados de la prueba de rendimiento durante 1 minuto se presentan a continuación:

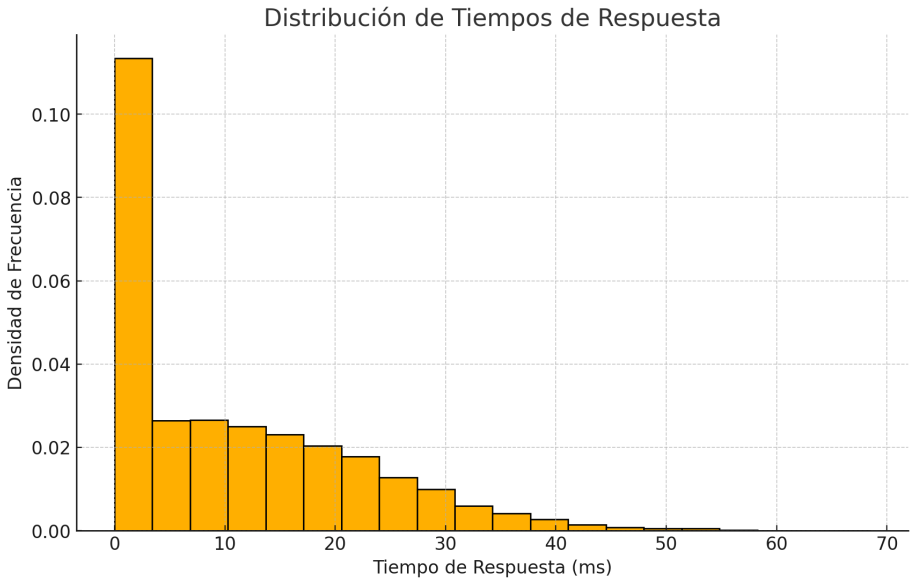


Figura 6.47. Resultados de la prueba de carga para el 100 % de la población

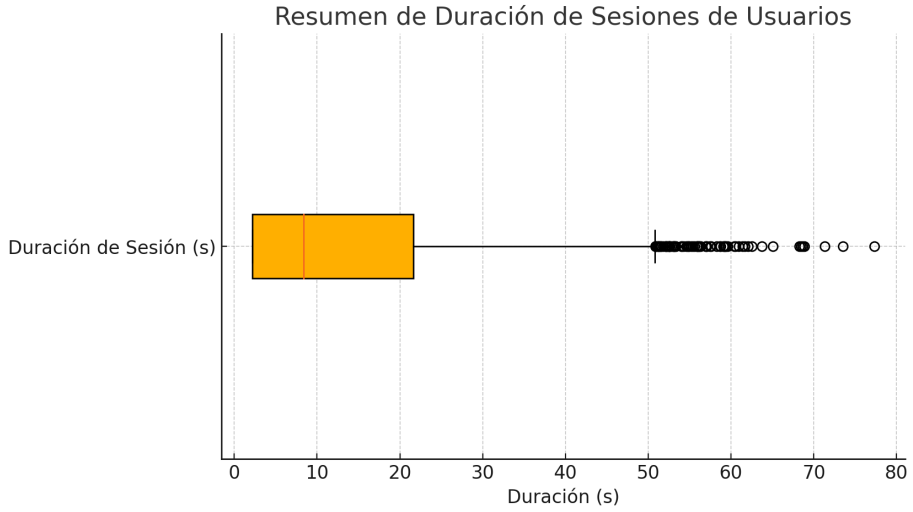


Figura 6.48. Resultados de la prueba de carga para el 100 % de la población

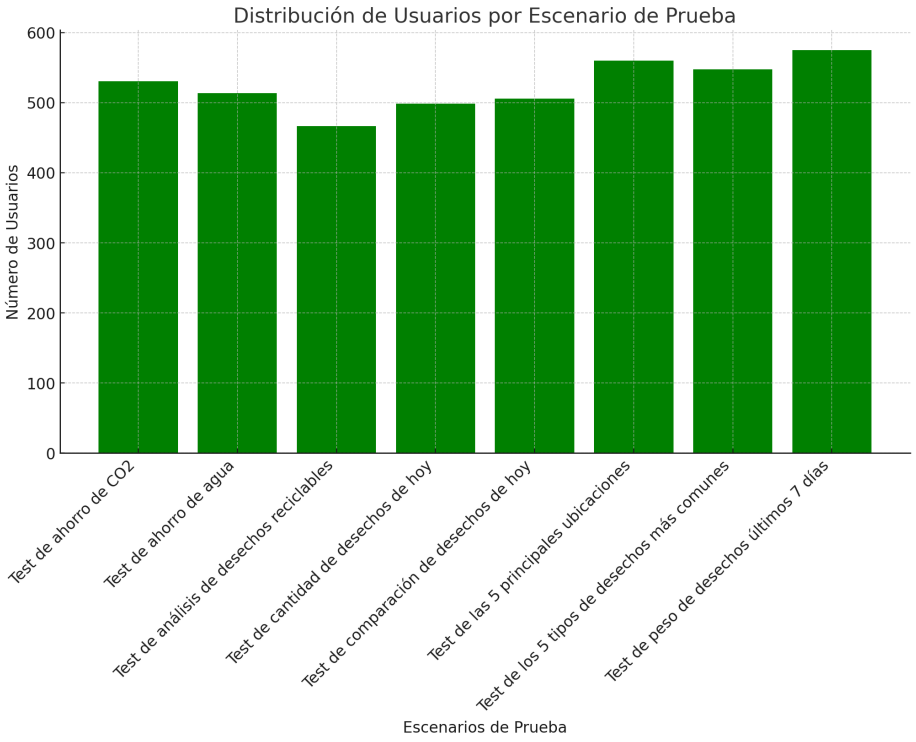


Figura 6.49. Resultados de la prueba de carga para el 100 % de la población

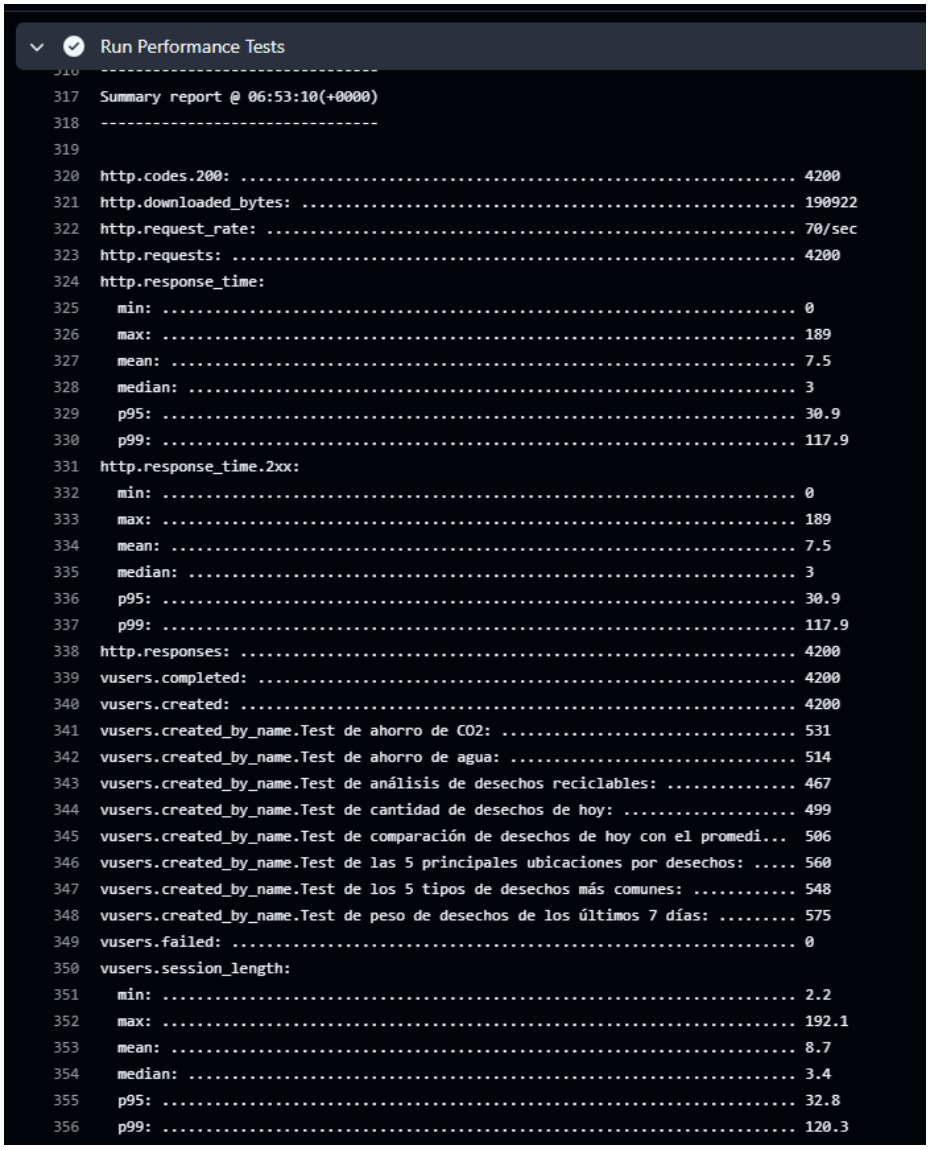


Figura 6.50. Resumen de resultados de la prueba de carga para el 100 %

Haciendo un resumen de los resultados se tiene lo siguiente:

Métrica	**20 solicitudes/seg**	**35 solicitudes/seg**	**70 solicitudes/seg**
Total de solicitudes	1200	2100	4200
Tiempo de respuesta (promedio)	3.5 ms	2.8 ms	7.5 ms
Tiempo de respuesta (mediana)	3 ms	2 ms	3 ms
Tiempo de respuesta (p95)	5 ms	6 ms	30.9 ms
Tiempo de respuesta (p99)	8.9 ms	25.8 ms	117.9 ms
Duración de sesión (promedio)	4.8 s	3.9 s	8.7 s
Duración de sesión (p95)	7.5 s	7.3 s	32.8 s
Duración de sesión (p99)	11.1 s	27.4 s	120.3 s

Cuadro 6.5. Resultados de pruebas de carga a diferentes tasas de solicitudes

Además, en el Anexo A, titulado *Pruebas de carga y rendimiento en* API *de producción*, se presentan los resultados obtenidos al realizar múltiples solicitudes a la *API*. En este anexo, se detallan los resultados de las pruebas realizadas para una carga de 35 solicitudes por segundo y 70 solicitudes por segundo.

Métrica	**20 solicitudes/seg**	**35 solicitudes/seg**	**70 solicitudes/seg**
Total de solicitudes	1200	2100	4200
Solicitudes exitosas (código 200)	1200	2096	4194
Errores (ETIMEDOUT)	0	4	6
Tiempo de respuesta (promedio)	3.5 ms	78.4 ms	4211.6 ms
Tiempo de respuesta (mediana)	3 ms	71.5 ms	4583.6 ms
Tiempo de respuesta (p95)	5 ms	120.3 ms	5168 ms
Tiempo de respuesta (p99)	8.9 ms	159.2 ms	6976.1 ms
Duración de sesión (promedio)	4.8 s	145 ms	4298 ms
Duración de sesión (mediana)	4.3 s	135.7 ms	4676.2 ms
Duración de sesión (p95)	7.5 s	202.4 ms	5378.9 ms
Duración de sesión (p99)	11.1 s	257.3 ms	6976.1 ms

Cuadro 6.6. Comparación de rendimiento en tres escenarios de carga en API de producción

6.2.3. Análisis de datos

La implementación de funciones para el análisis de datos de usuarios específicos y el análisis general permite aprovechar la información generada en los registros de desechos para obtener resultados significativos. Este análisis facilita la evaluación de los hábitos de los usuarios, el impacto ambiental de sus acciones, y su contribución en términos de reducción de residuos y reciclaje.

Estas funciones analíticas están diseñadas para ofrecer a cada usuario un resumen de su comportamiento en la gestión de desechos, así como métricas generales que abarcan todos los usuarios del sistema.

A continuación, se muestran los resultados de algunas solicitudes de análisis realizadas por usuarios individuales:

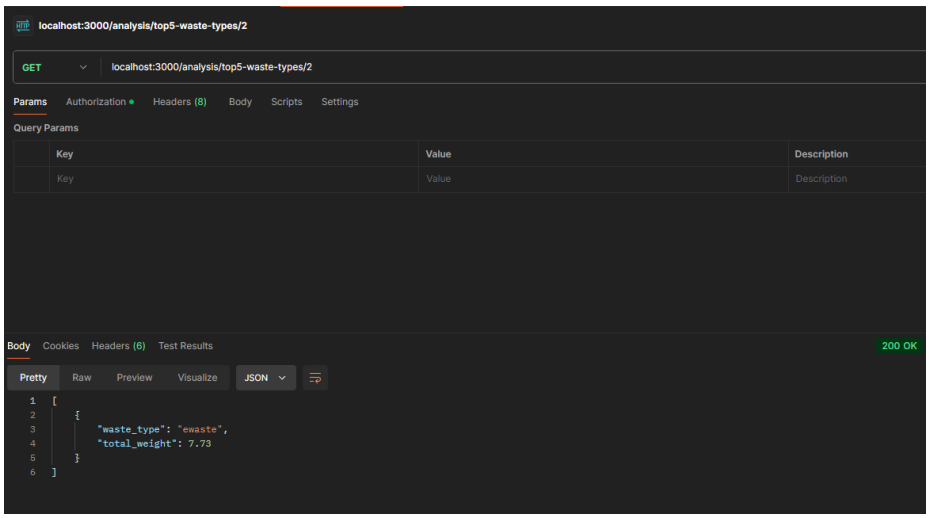


Figura 6.51. Resultado top 5 desechos usuario

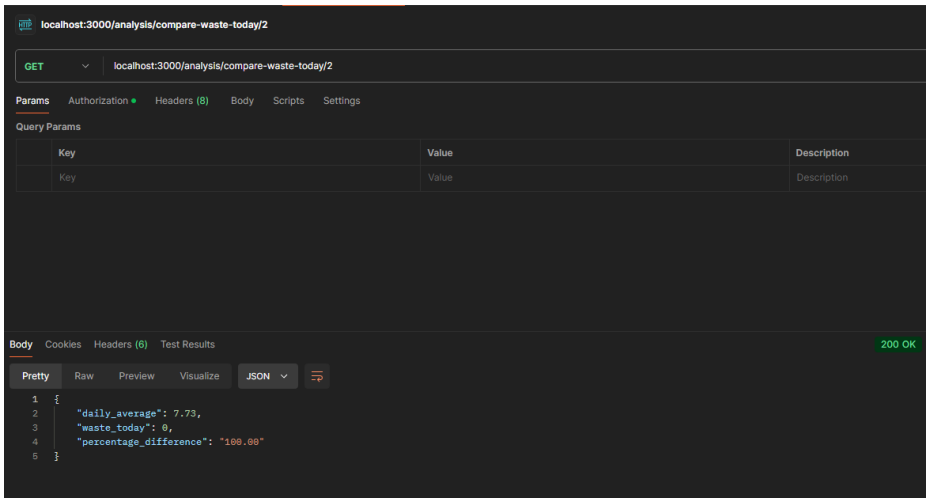


Figura 6.52. Resultado porcentaje comparación usuario

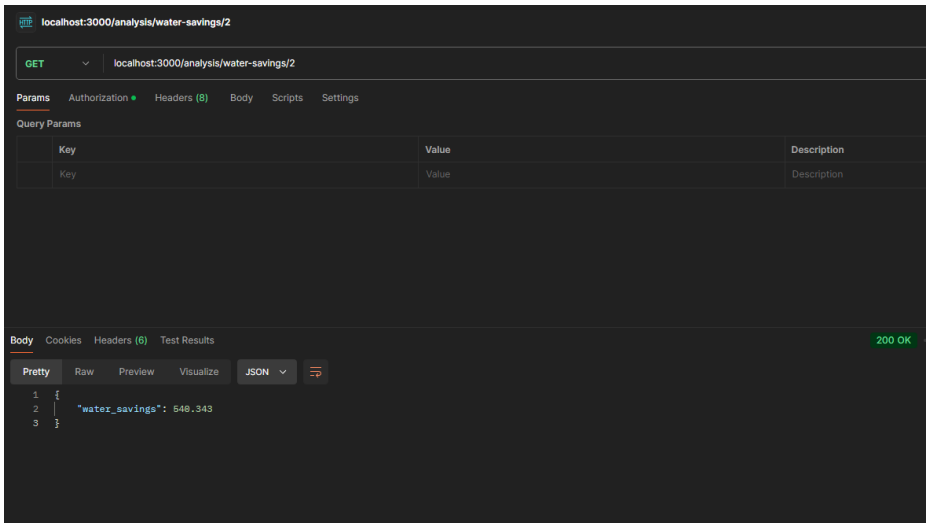


Figura 6.53. Resultado porcentaje comparación usuario

A continuación, se muestran algunos ejemplos de visualizaciones generadas en *Power BI* a partir de solicitudes al *API*:

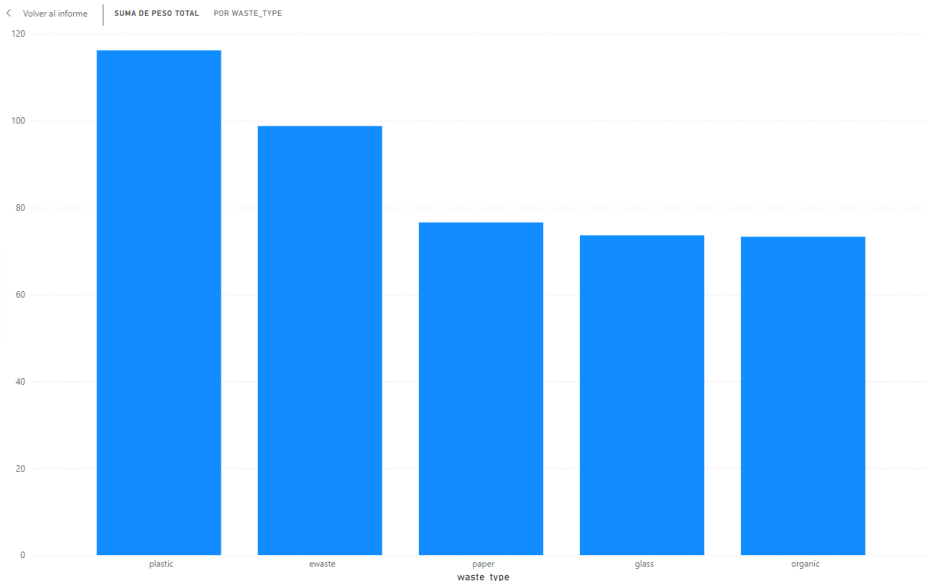


Figura 6.54. Resultado top 5 desechos generales

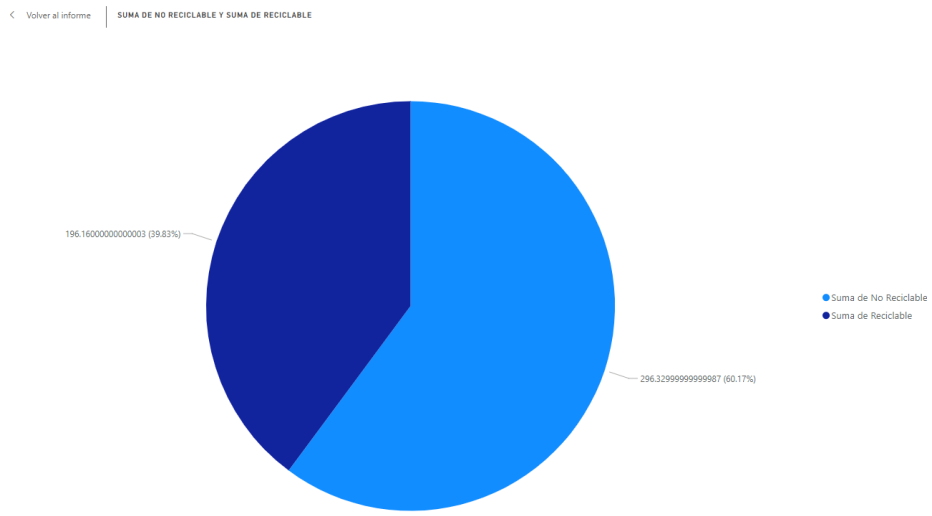


Figura 6.55. Resultado desechos reciclables generales

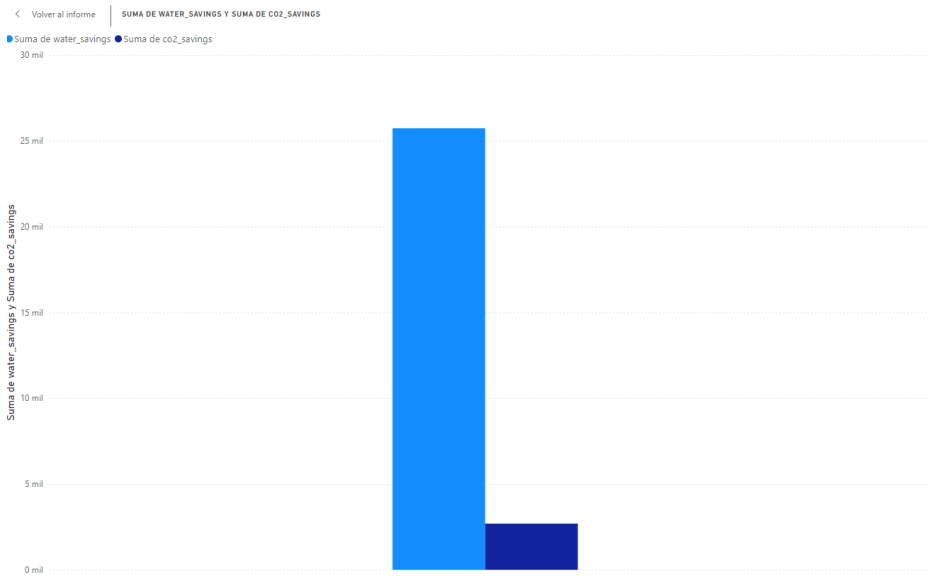


Figura 6.56. Resultado CO2 y Agua generales

6.2.4. Implementación de *CI/CD*

Como parte final del módulo y un componente crucial en el proceso de desarrollo, se presentan los resultados obtenidos a partir de la implementación del *pipeline* de *CI/CD*. Este *pipeline* fue diseñado para automatizar las tareas de integración y despliegue, asegurando que cada cambio en el código sea verificado y probado antes de ser incorporado en el sistema de producción. Su implementación no solo mejora la eficiencia del proceso de desarrollo, sino que también reduce el riesgo de errores y asegura la estabilidad del API en cada actualización.

Pruebas

Durante el proceso de pruebas, se evaluaron distintos casos que permitieron comprobar cómo el sistema responde ante situaciones comunes que pueden surgir al realizar modificaciones en el código. Estas pruebas ayudan a validar la robustez del *API*, identificando y corrigiendo problemas antes de que impacten a los usuarios finales. A continuación, se presentan los resultados de los dos escenarios principales, demostrando cómo el *pipeline* gestiona los cambios y asegura un despliegue confiable:

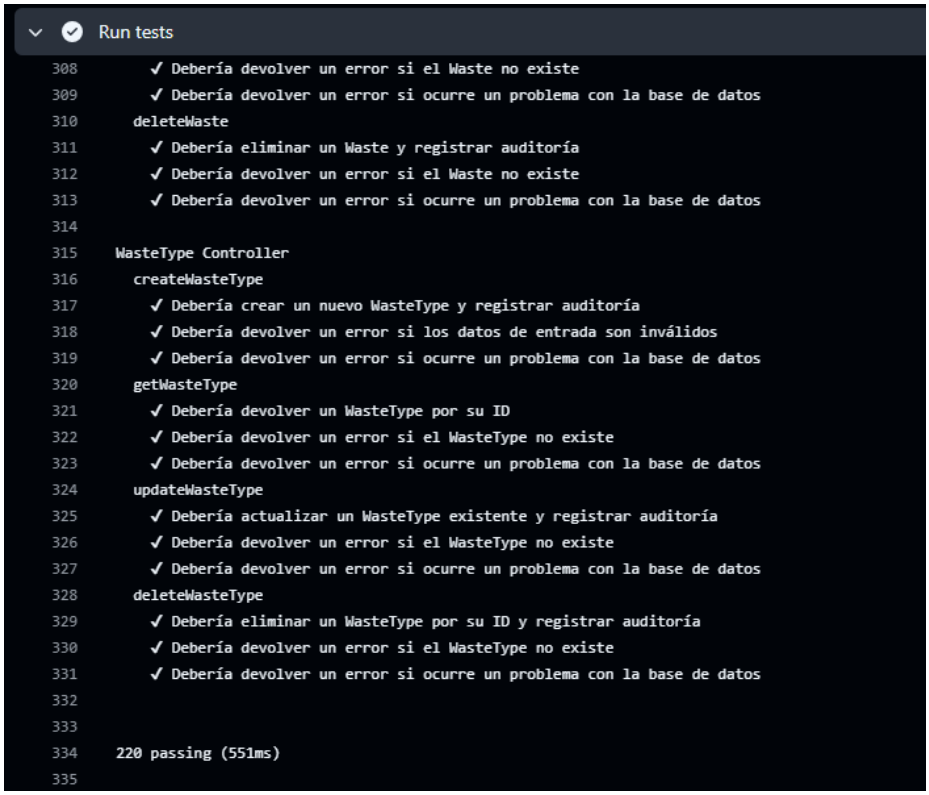


Figura 6.57. Resultado de los *tests* acertados

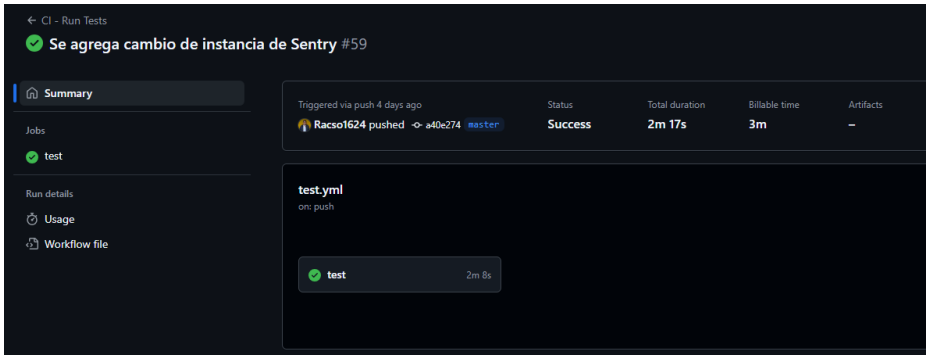


Figura 6.58. Resultado del *action* de los *tests* acertados

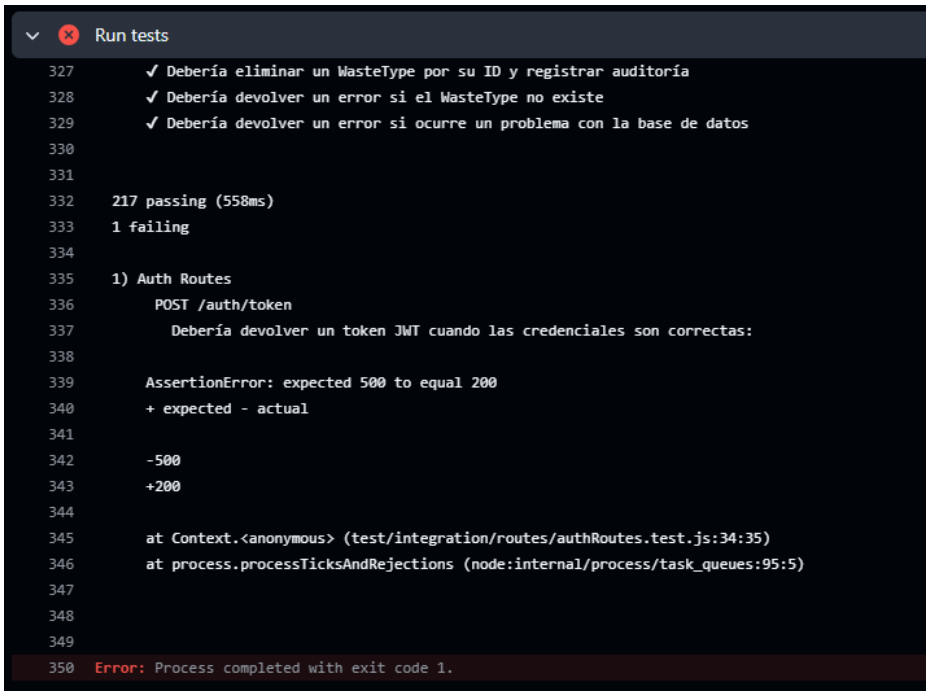


Figura 6.59. Resultado de los *tests* fallados

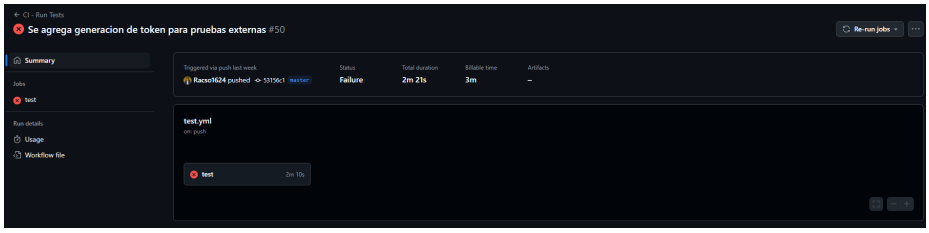


Figura 6.60. Resultado del *action* de los *tests* fallados

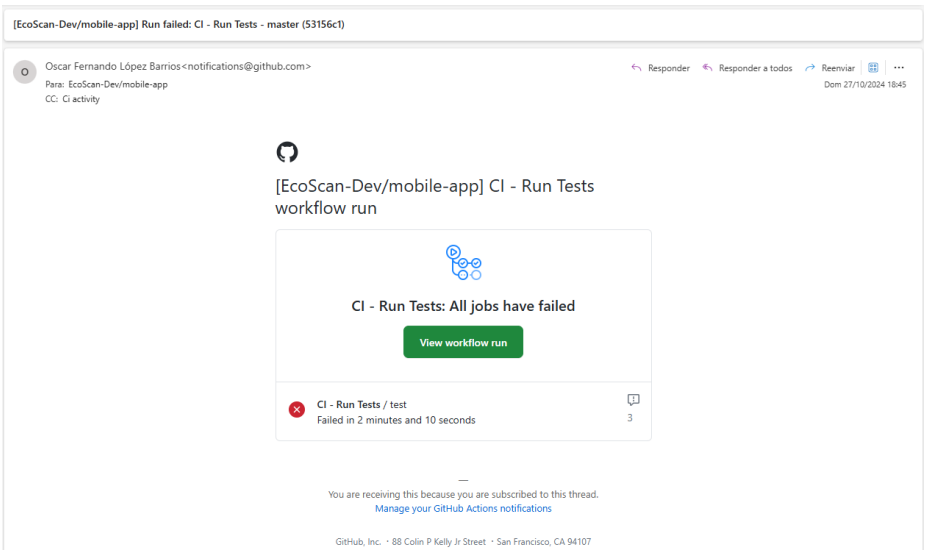


Figura 6.61. Resultado de los *tests* fallados en el *mail*

Monitoreo de errores con *Sentry*

La elección de una herramienta de monitoreo de errores es fundamental, ya que permite capturar datos precisos para identificar y resolver problemas que pueden haber pasado desapercibidos una vez que el *API* se encuentra en producción.

Además, ***Sentry*** no solo proporciona monitoreo de errores, sino que también ofrece una amplia gama de datos adicionales. En este caso, *Sentry* permite obtener trazas detalladas de ejecución y monitorear el rendimiento de las consultas (*queries*), proporcionando una visión integral del funcionamiento del sistema. Esto facilita la detección de cuellos de botella y problemas de rendimiento, lo cual es esencial para mantener la eficiencia y confiabilidad del *API* en entornos de producción.

A continuación, se presentan los resultados de los escenarios en los que **Sentry** ha sido utilizado para el monitoreo y diagnóstico de errores en el sistema:

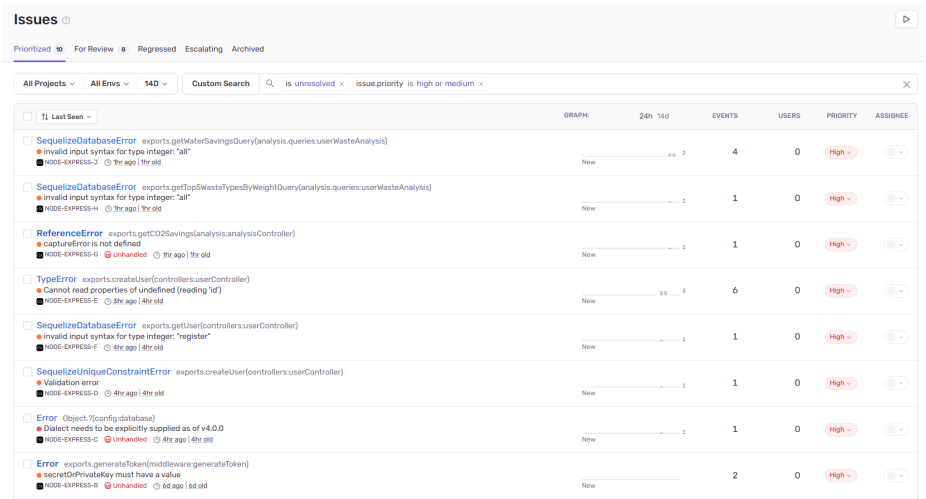


Figura 6.62. Resultado de los errores de *Sentry*

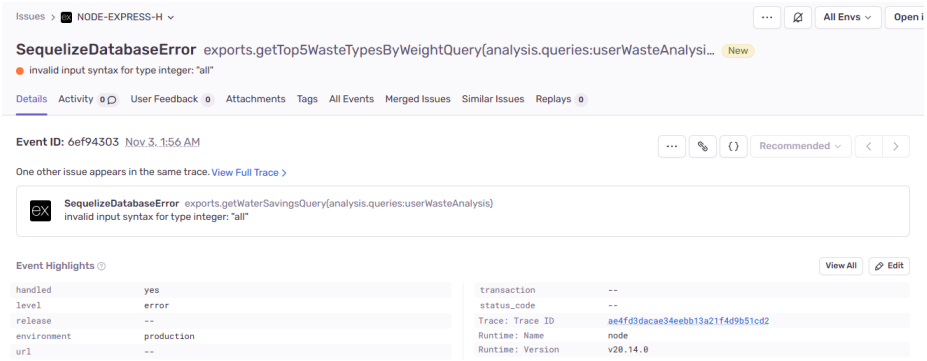


Figura 6.63. Resultado de la descripcion de un error en *Sentry*

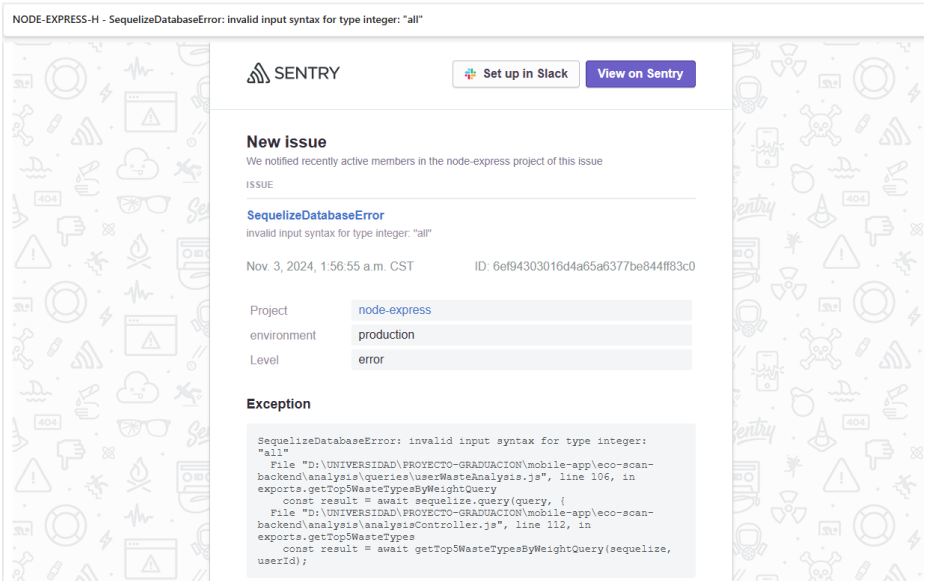


Figura 6.64. Resultado una notificacion de error

Figura 6.65. Resultado del apartado de *performance* de *Sentry*

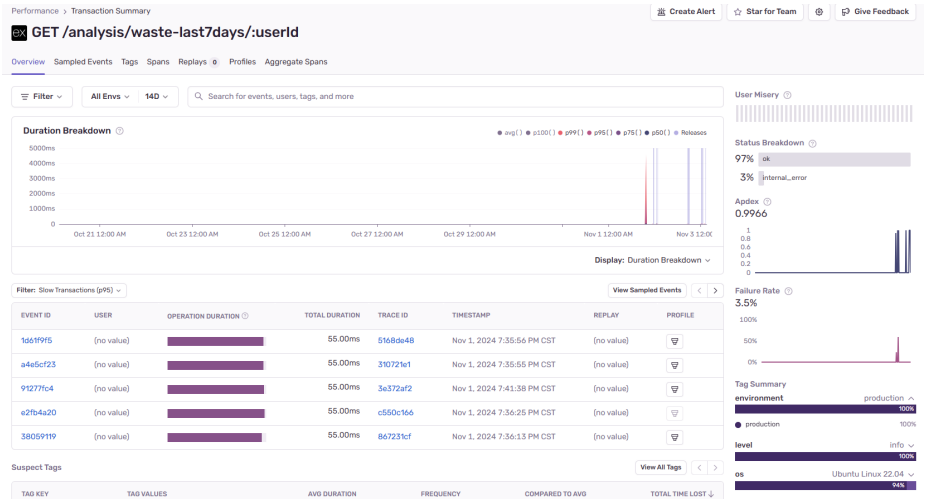


Figura 6.66. Resultado del detalle de descripción de *performance* de *Sentry*

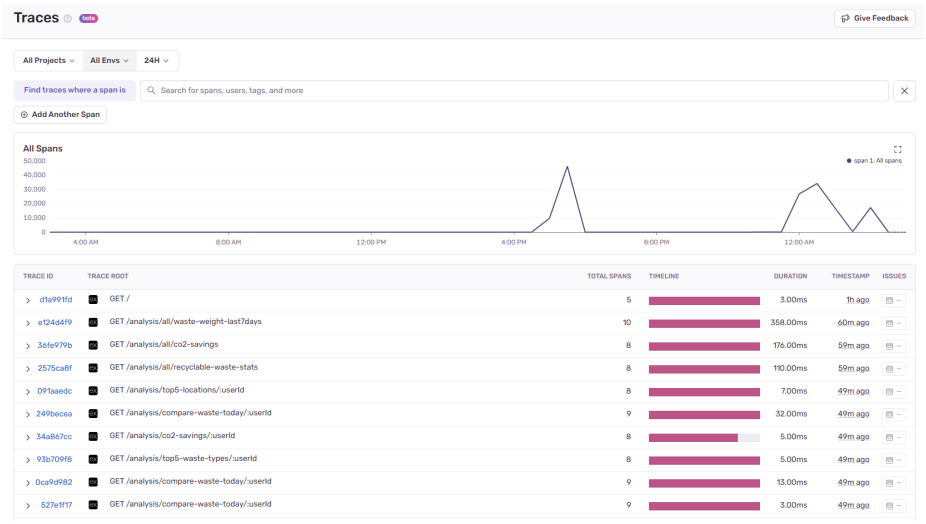


Figura 6.67. Resultado de los traces de *Sentry*

Automatización con *GitHub Actions*

La última funcionalidad necesaria para establecer una base sólida de *CI/CD* es la automatización de tareas. En este caso, se logró implementar exitosamente mediante **GitHub Actions**. Esta herramienta permite definir flujos de trabajo automatizados que verifican la calidad del código y facilitan el despliegue continuo. Con *GitHub Actions*, es posible configurar una serie de acciones para evaluar el código en cada cambio realizado, garantizando así un control de calidad constante y permitiendo una entrega continua y confiable del software.

A continuación se muestra el funcionamiento:

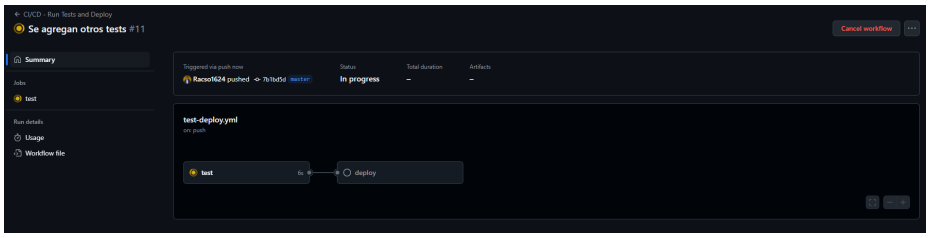


Figura 6.68. Resultado de *GitHub Actions*

Resultado del *deploy*:

Figura 6.69. Resultado del *deploy*

6.3. Resultados de inteligencia artificial

6.3.1. Resultados de *Xception*

En esta sección se presentan las métricas de rendimiento, pruebas realizadas y los resultados de la implementación de *MLOps* y técnicas de *Explainable AI*, como *MLFlow*, *GradCAM* y *LIME*, aplicadas al modelo *Xception*.

Historial de precisión y pérdida

Las figuras a continuación muestran la evolución de la precisión y la pérdida del modelo durante el entrenamiento y la validación.

Precisión del modelo La Figura 6.70 muestra cómo la precisión del modelo cambia a lo largo de las épocas, tanto en el conjunto de entrenamiento como en el de validación. El modelo alcanza una precisión promedio del 98 %, destacando un rendimiento excepcional en la clasificación de residuos.

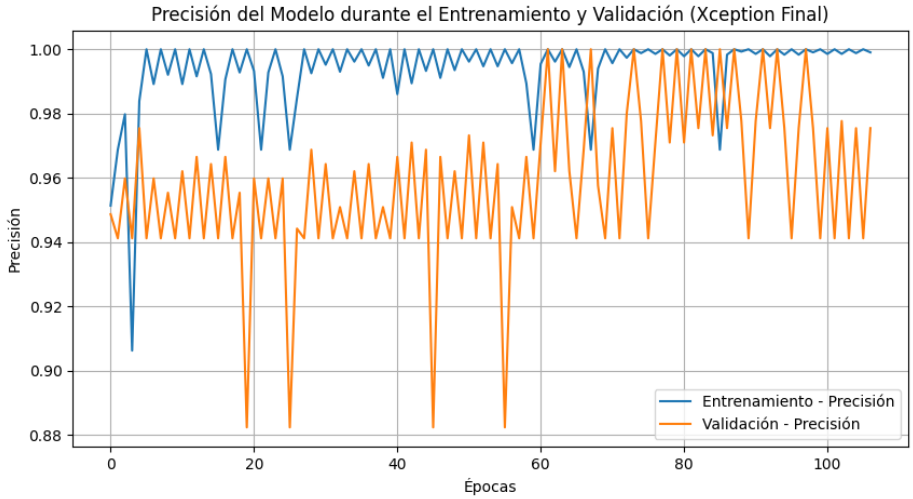


Figura 6.70. Evolución de la precisión del modelo durante el entrenamiento y validación (*Xception* Final)

Pérdida del modelo La Figura 6.71 representa la evolución de la pérdida durante el entrenamiento y la validación, proporcionando una medida de qué tan bien se ajusta el modelo a los datos.

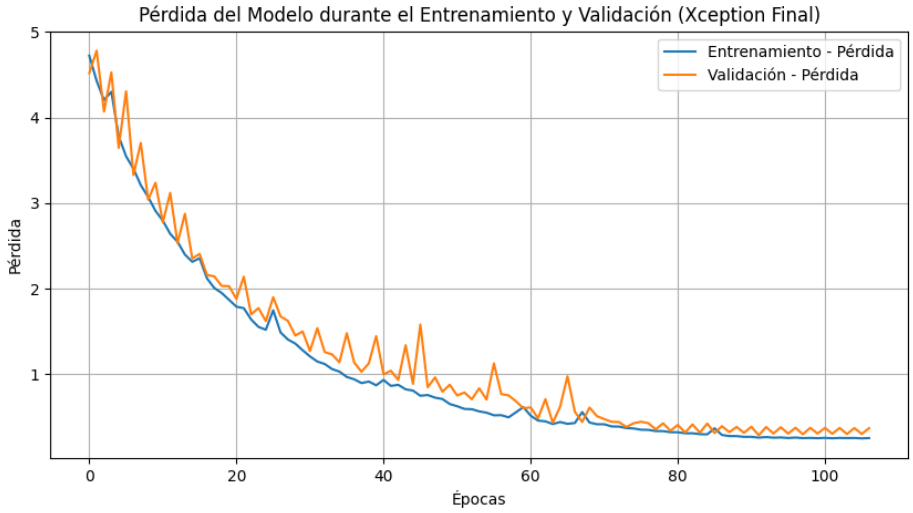


Figura 6.71. Evolución de la pérdida del modelo durante el entrenamiento y validación (*Xception* Final)

Matriz de confusión y reporte de clasificación

La Figura 6.72 muestra la matriz de confusión, y el cuadro 6.7 proporciona el reporte de clasificación con métricas detalladas para cada clase de residuos.

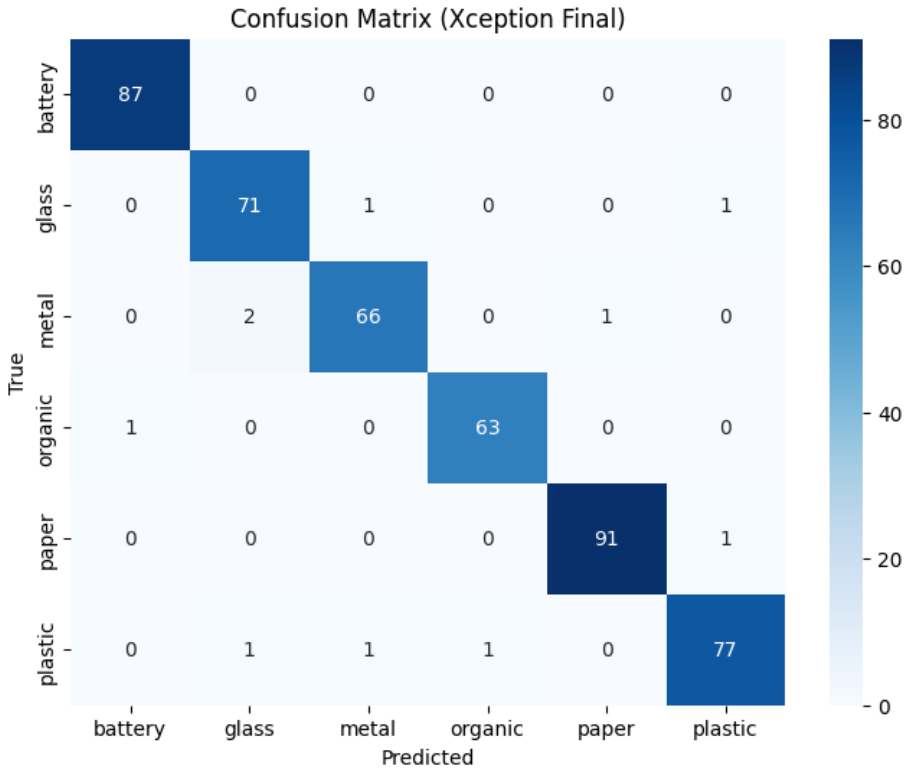


Figura 6.72. Matriz de confusión del modelo *Xception* en el conjunto de prueba

Clase	Precisión	Recall	F1-Score	Soporte
Battery	0.99	1.00	0.99	87
Glass	0.96	0.97	0.97	73
Metal	0.97	0.96	0.96	69
Organic	0.98	0.98	0.98	64
Paper	0.99	0.99	0.99	92
Plastic	0.97	0.96	0.97	80
Precisión Total	0.98 (465 muestras)			
Promedio Macro	0.98	0.98	0.98	
Promedio Ponderado	0.98	0.98	0.98	

Cuadro 6.7. Reporte de clasificación del modelo *Xception* en el conjunto de prueba

Prueba de concepto de *MLOps* con *MLFlow*

Para mejorar la trazabilidad y reproducibilidad de los experimentos de *Machine Learning*, se implementó una prueba de concepto (POC) utilizando *MLFlow*. Esta herramienta facilita el registro, monitoreo y versionamiento de modelos, lo cual es esencial en procesos de *MLOps* (*Machine Learning Operations*). A continuación, se describen los resultados obtenidos al integrar *MLFlow* en el ciclo de vida del modelo *Xception*.

Interfaz de experimentos en *MLFlow* La Figura 6.73 muestra la interfaz de *MLFlow*, en la cual se registraron dos ejecuciones de entrenamiento. En esta vista se pueden observar detalles im-

portantes como la duración de cada ejecución, el estado final (completado), y las métricas registradas para facilitar el análisis y comparación.

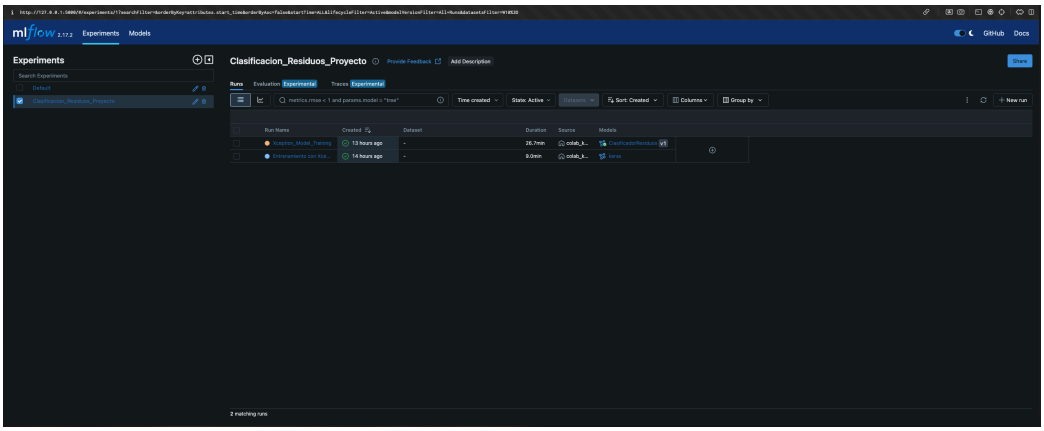


Figura 6.73. Interfaz de *MLFlow* mostrando los experimentos registrados. Aquí se observan las ejecuciones del modelo *Xception* y sus correspondientes métricas y parámetros

Detalles del primer experimento: entrenamiento del modelo *Xception* La Figura 6.74 presenta los detalles del primer experimento, titulado *Xception Model Training*. En esta ejecución, se registraron los siguientes parámetros clave: tasa de aprendizaje de 0.0001, tamaño de lote de 32, y un total de 200 épocas. Además, se muestra el valor de la precisión de validación (0.9732) y la pérdida de validación (0.5581), reflejando el rendimiento del modelo en esta configuración.

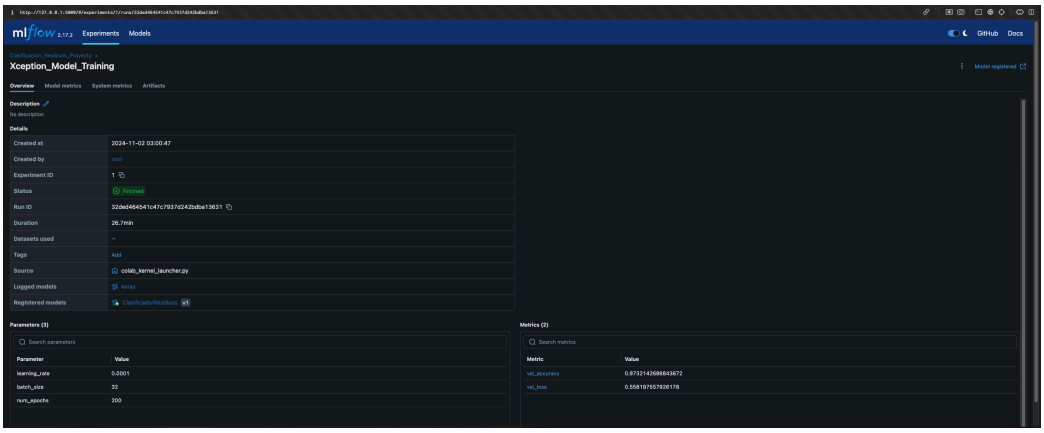


Figura 6.74. Detalles del experimento *Xception Model Training* en *MLFlow*. Se observan las métricas y los parámetros registrados durante el entrenamiento del modelo

Detalles del segundo experimento: entrenamiento con *GradCAM* En la Figura 6.75 se muestran los detalles del segundo experimento, titulado Entrenamiento con *Xception* y *GradCAM*. Esta ejecución tuvo una duración de 9 minutos, y se configuró con parámetros similares al primer experimento, con la adición del valor de regularización L2 (0.002). Las métricas de validación registradas fueron una precisión de 0.9732 y una pérdida de 0.4891.

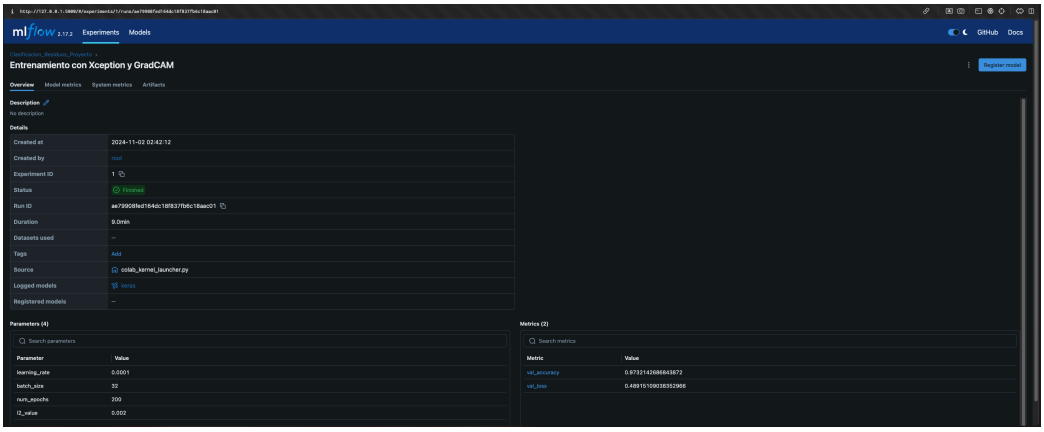


Figura 6.75. Detalles del experimento Entrenamiento con *Xception* y *GradCAM* en *MLFlow*. Se destacan las métricas de precisión y pérdida, junto con los parámetros utilizados

Resultados de *Explainable AI*: *GradCAM* y *LIME*

Interpretación con *GradCAM* Las siguientes figuras presentan ejemplos de interpretaciones generadas con *GradCAM* para las distintas clases de residuos. Los mapas de *GradCAM* resaltan las regiones cruciales que el modelo considera al realizar la clasificación, enfocándose en características visuales importantes, como bordes, contornos y texturas.

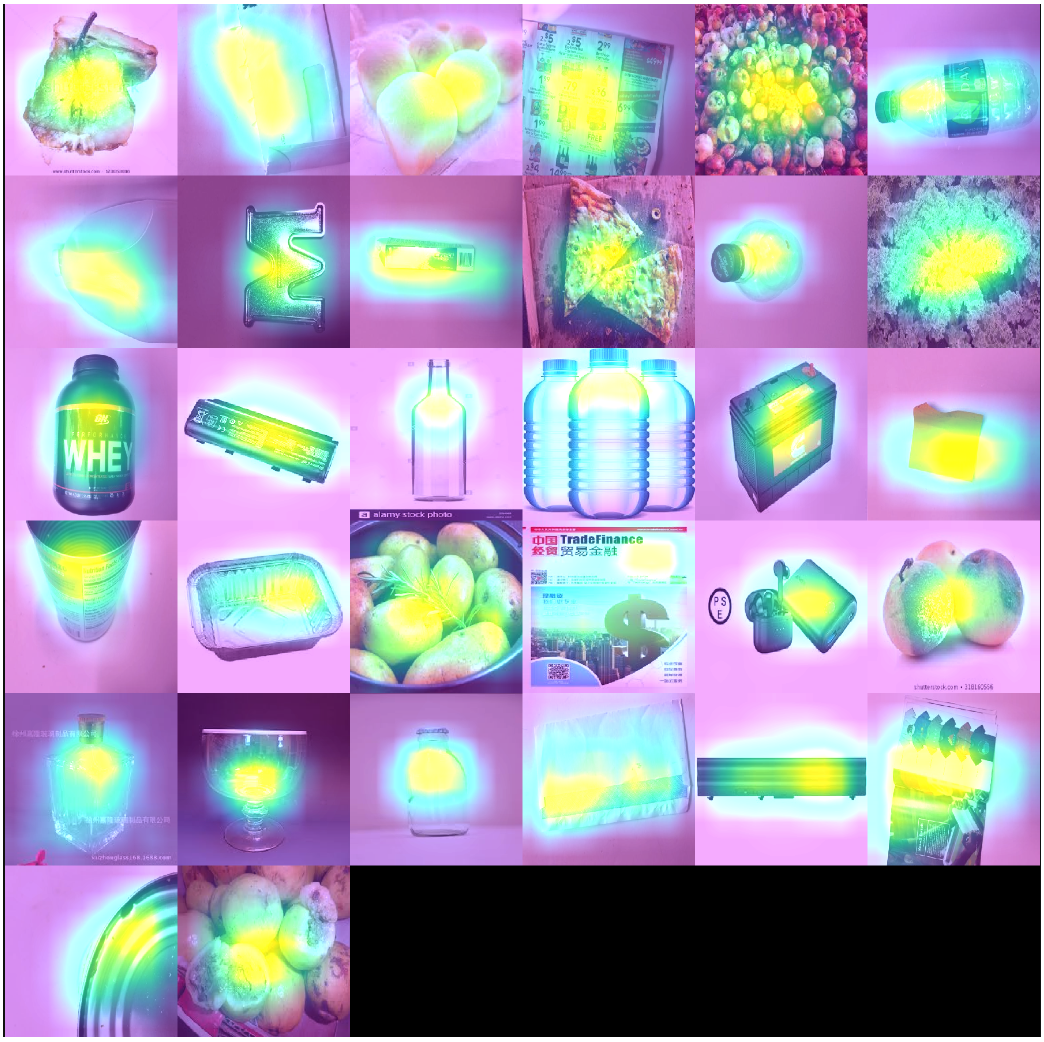


Figura 6.76. Ejemplos combinados de mapas de atención generados con *GradCAM* para las diferentes clases de residuos. Se observan las áreas de atención del modelo, que se enfocan en las características clave de cada residuo, como bordes, texturas y patrones específicos, esenciales para la correcta clasificación

Explicaciones generadas con *LIME* *LIME* (*Local Interpretable Model-Agnostic Explanations*) se utilizó para descomponer las decisiones del modelo en segmentos interpretables. A continuación, se presentan los resultados de *LIME* para cada una de las clases de residuos, destacando las áreas que influyen significativamente en las predicciones del modelo.

Figura 6.77. Explicación generada con *LIME* para la clase *Battery*. Las áreas resaltadas muestran las características que más influyeron en la predicción, indicando bordes y regiones específicas

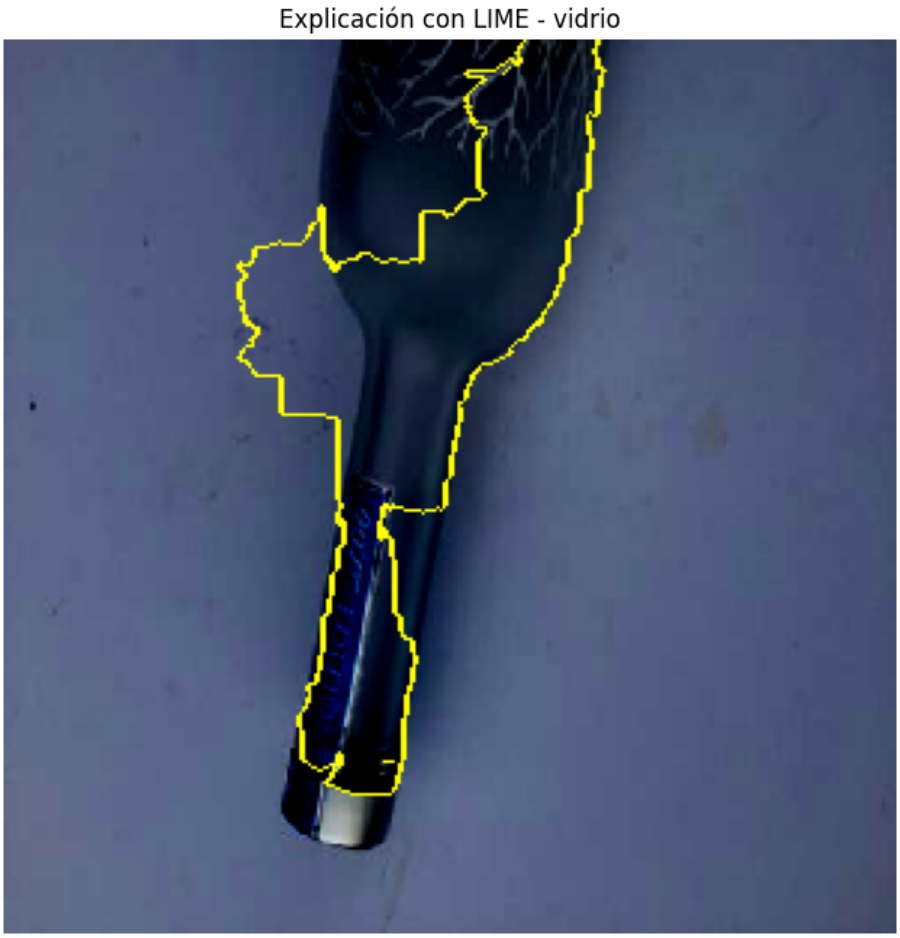


Figura 6.78. Explicación generada con *LIME* para la clase *Glass*. Las áreas resaltadas indican las texturas y bordes que el modelo consideró importantes para clasificar correctamente el residuo

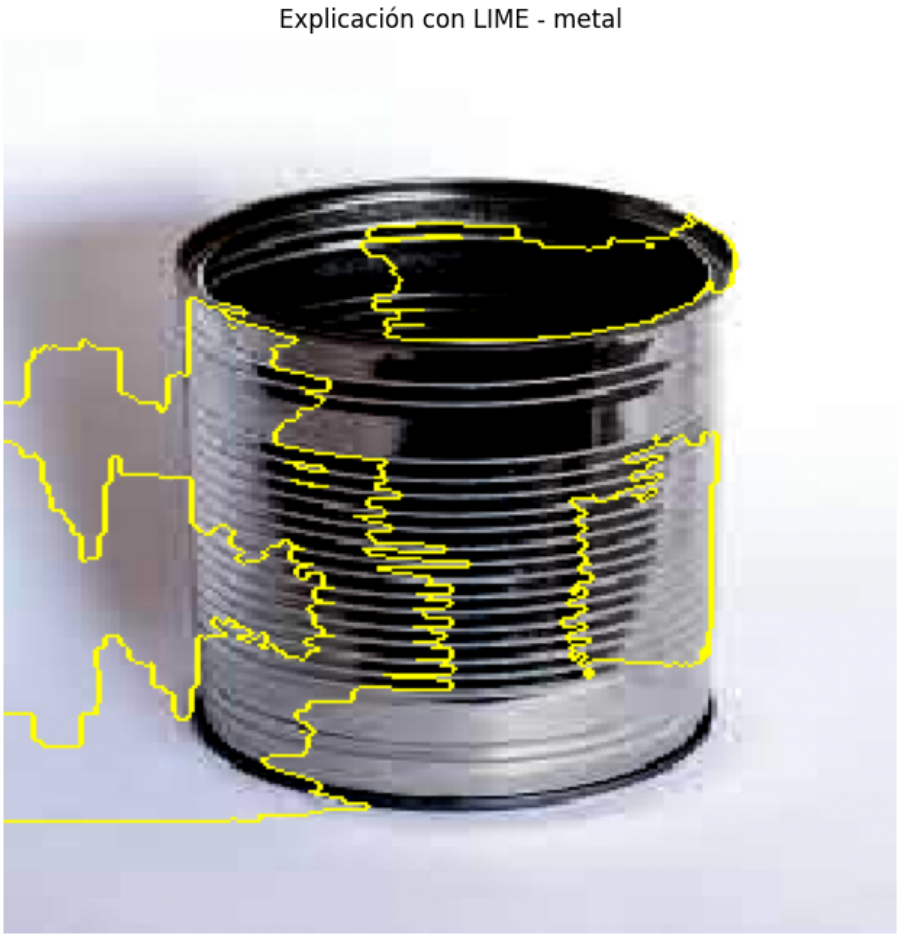


Figura 6.79. Explicación generada con *LIME* para la clase *Metal*. Las áreas destacadas muestran las regiones clave que ayudaron al modelo a identificar las propiedades del metal, como el brillo y las texturas metálicas

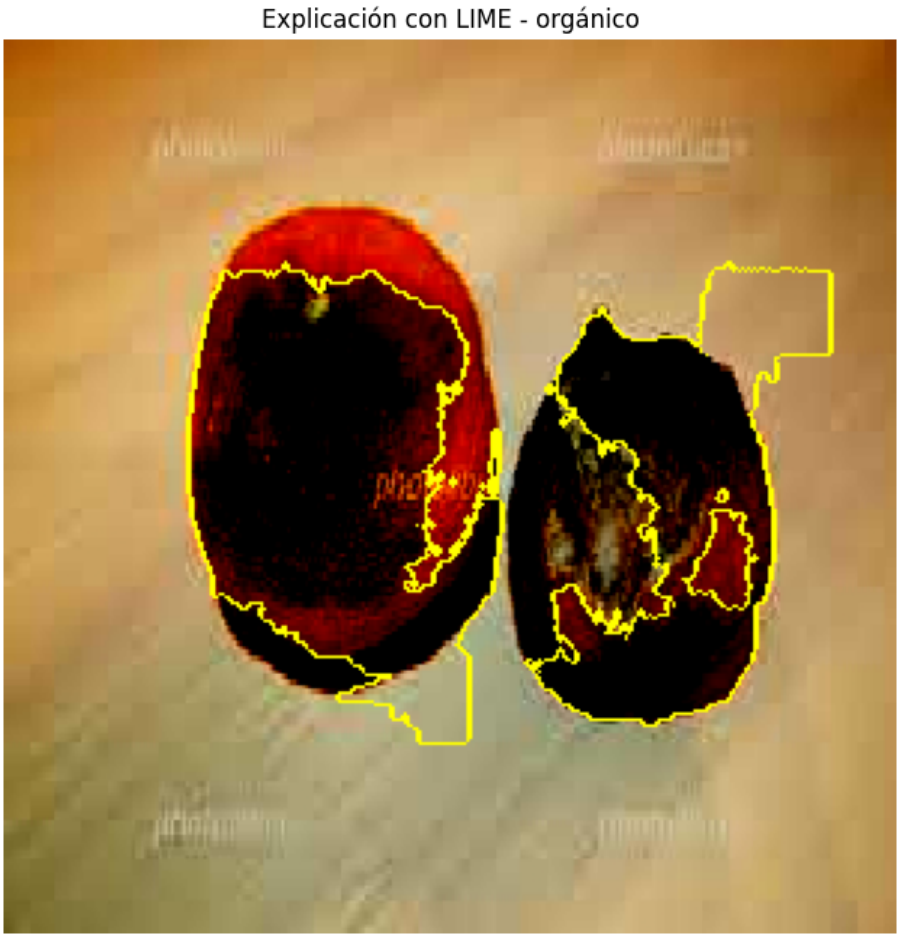


Figura 6.80. Explicación generada con *LIME* para la clase *Organic*. Las áreas resaltadas indican las texturas y patrones orgánicos que fueron más relevantes para la predicción

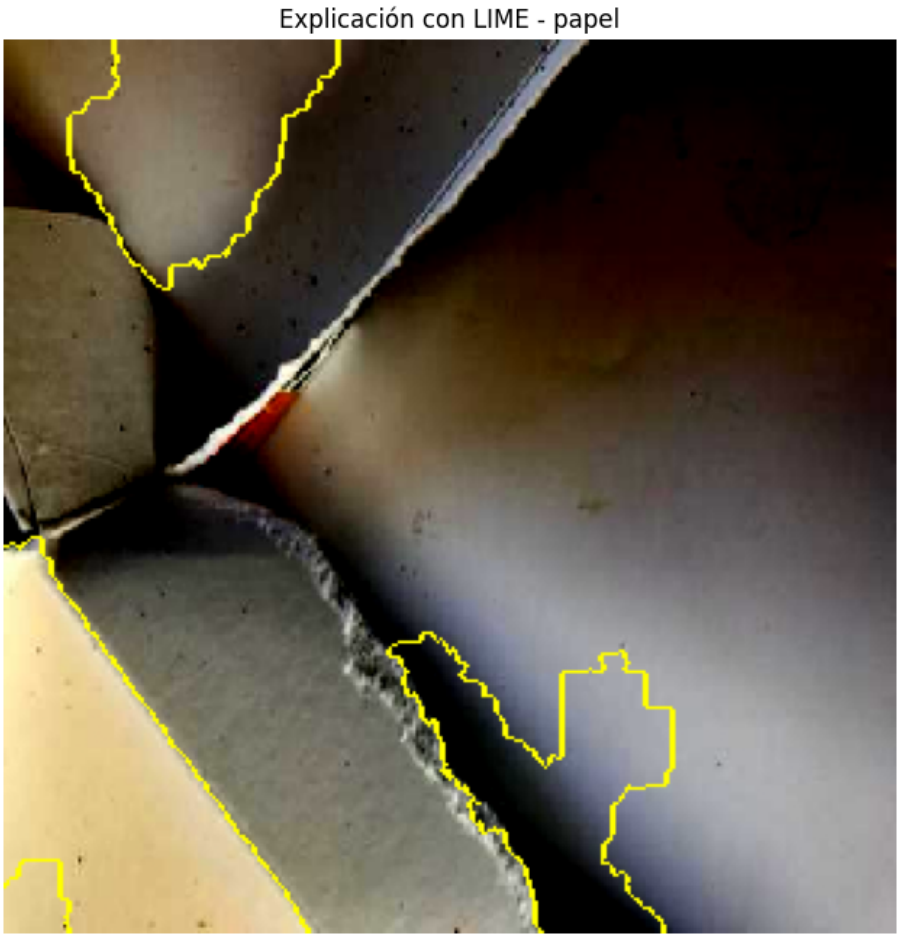


Figura 6.81. Explicación generada con *LIME* para la clase *Paper*. Las áreas importantes se centran en las texturas y bordes del papel que influenciaron la decisión del modelo

Figura 6.82. Explicación generada con *LIME* para la clase *Plastic*. Las áreas resaltadas muestran las regiones específicas que el modelo utilizó para identificar el plástico, como los contornos y las variaciones en la textura

Pruebas de *Saliency Maps* con *SmoothGrad* En esta subsección, se presentan los resultados de los *saliency maps* generados con *SmoothGrad* aplicados al conjunto de pruebas. *SmoothGrad* se utiliza para obtener mapas de atención más claros y menos ruidosos, resaltando las áreas que el modelo considera importantes para realizar la clasificación.

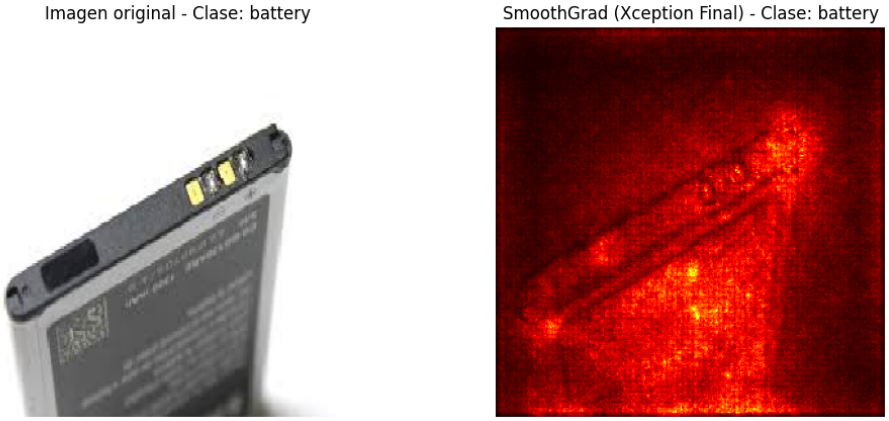


Figura 6.83. *Saliency Map* generado con *SmoothGrad* para la clase *Battery*

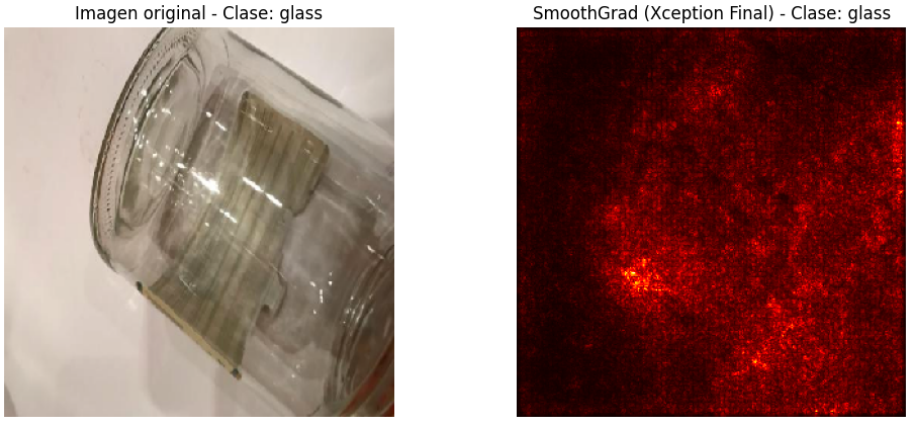


Figura 6.84. *Saliency Map* generado con *SmoothGrad* para la clase *Glass*.

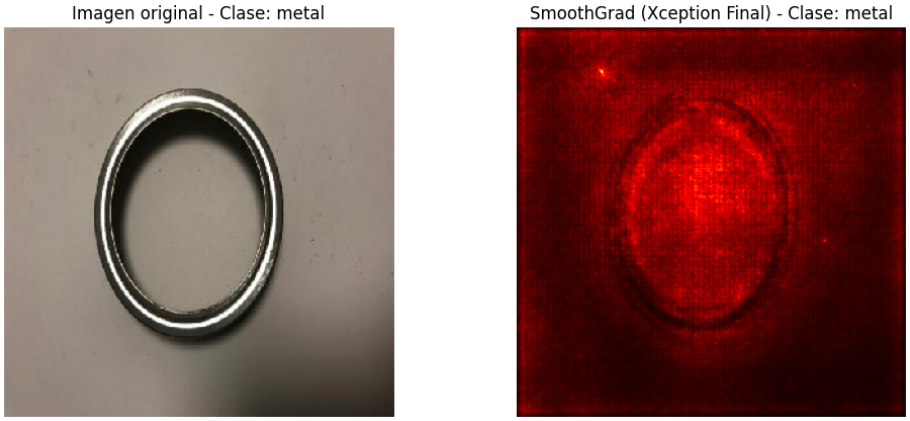


Figura 6.85. *Saliency Map* generado con *SmoothGrad* para la clase *Metal*

Figura 6.86. *Saliency Map* generado con *SmoothGrad* para la clase *Organic*

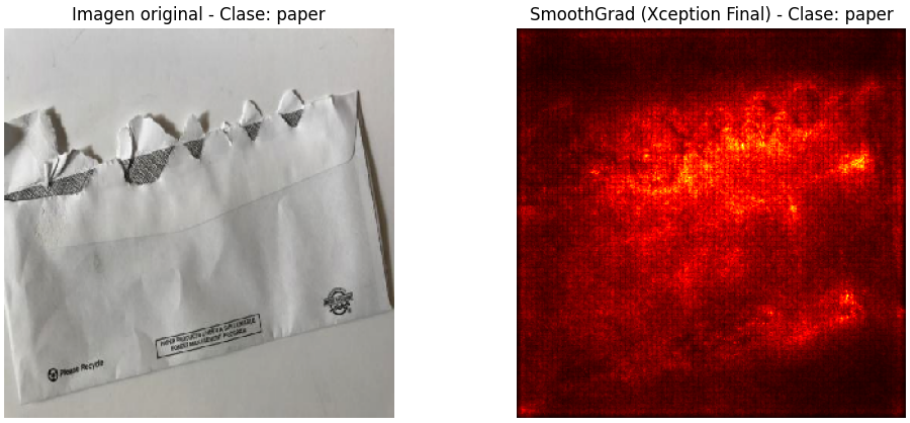


Figura 6.87. *Saliency Map* generado con *SmoothGrad* para la clase *Paper*

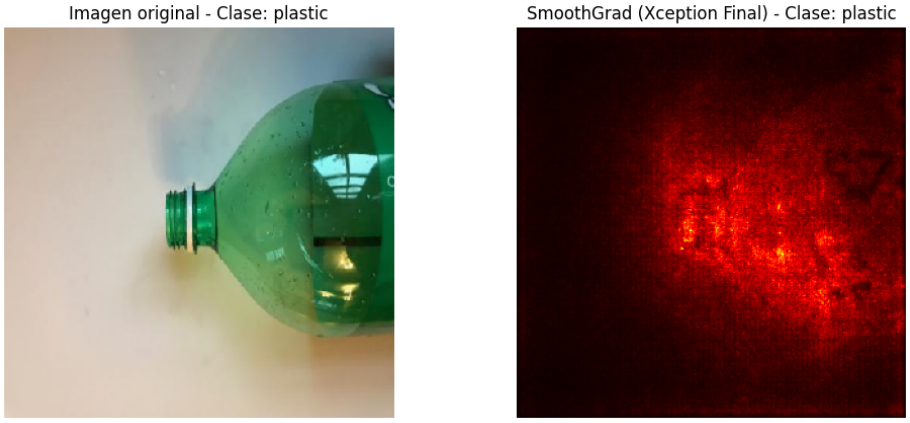


Figura 6.88. *Saliency Map* generado con *SmoothGrad* para la clase *Plastic*

Resultados de pruebas con imágenes reales Para evaluar el rendimiento del modelo en situaciones prácticas, se probaron 16 imágenes reales de residuos, con diferentes condiciones de fondo y contexto. Los resultados a continuación muestran cómo el modelo maneja estas situaciones, destacando tanto las imágenes correctamente clasificadas como los ejemplos más desafiantes.

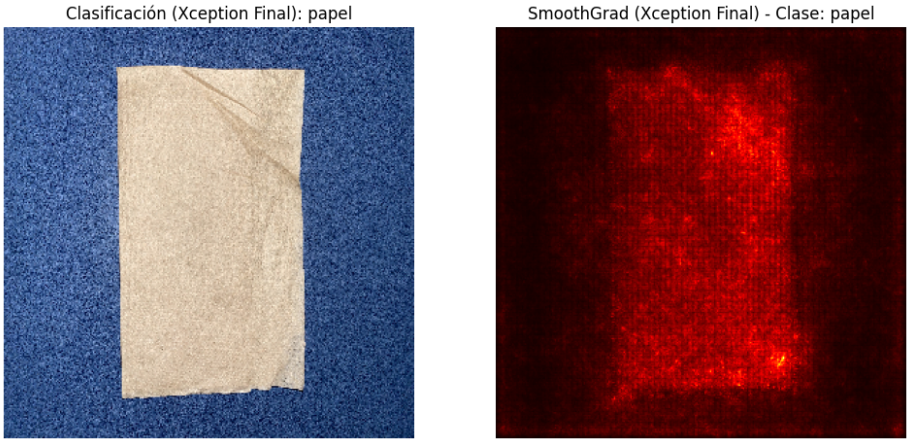


Figura 6.89. Clasificación de un residuo real perteneciente a la clase *Papel*

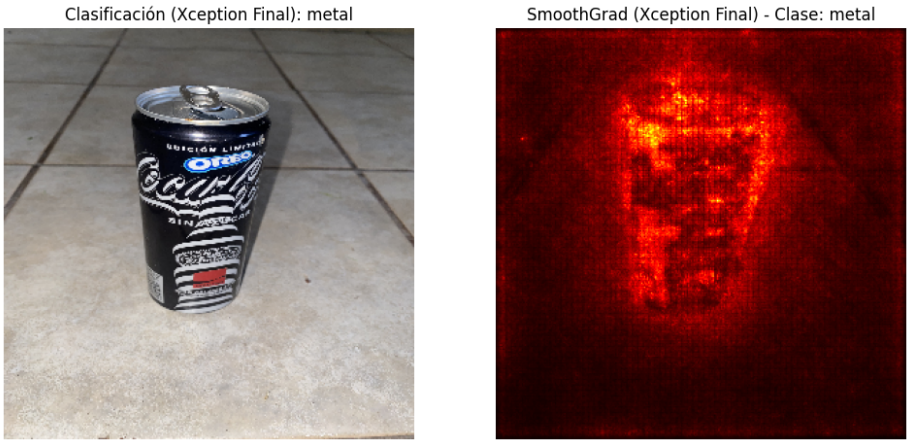


Figura 6.90. Clasificación de un residuo real perteneciente a la clase *Metal*

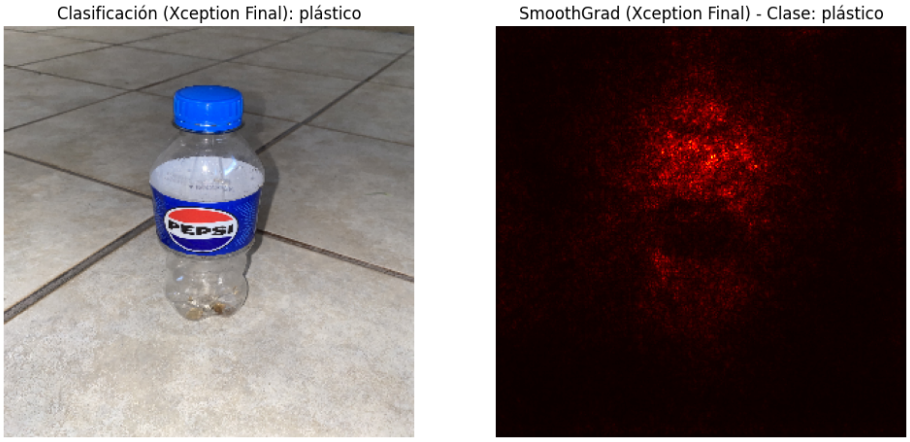


Figura 6.91. Clasificación de un residuo real perteneciente a la clase *Plastic*

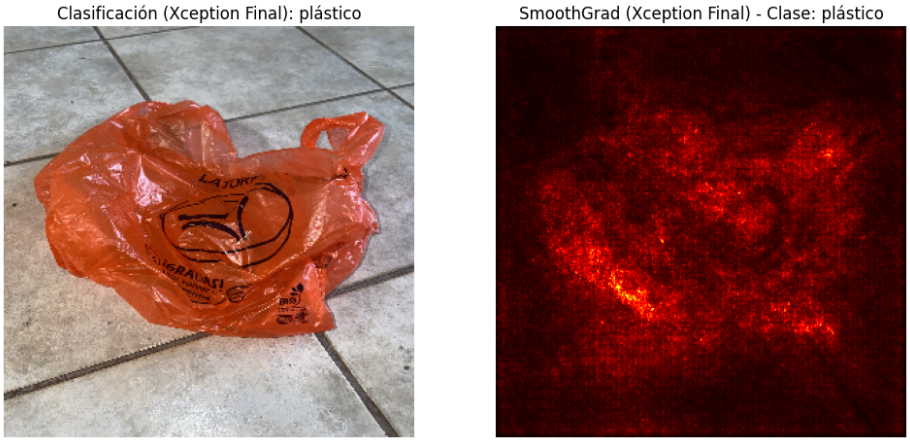


Figura 6.92. Clasificación de un residuo real perteneciente a la clase *Plastic*

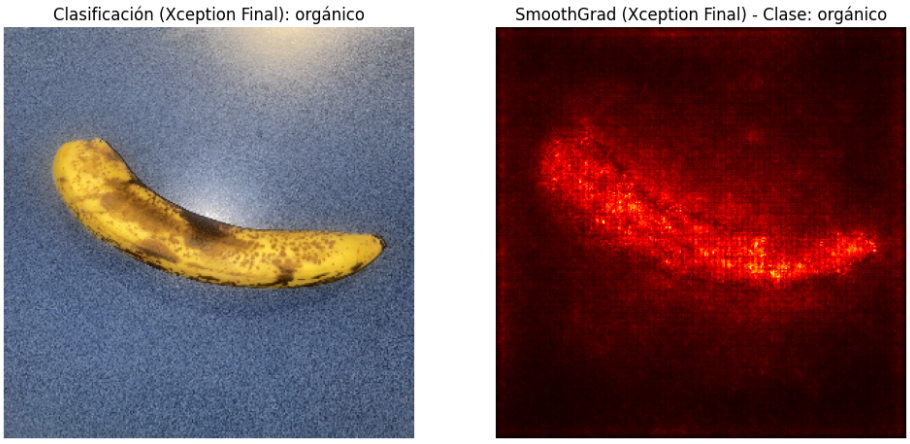


Figura 6.93. Clasificación de un residuo real perteneciente a la clase *Organic*

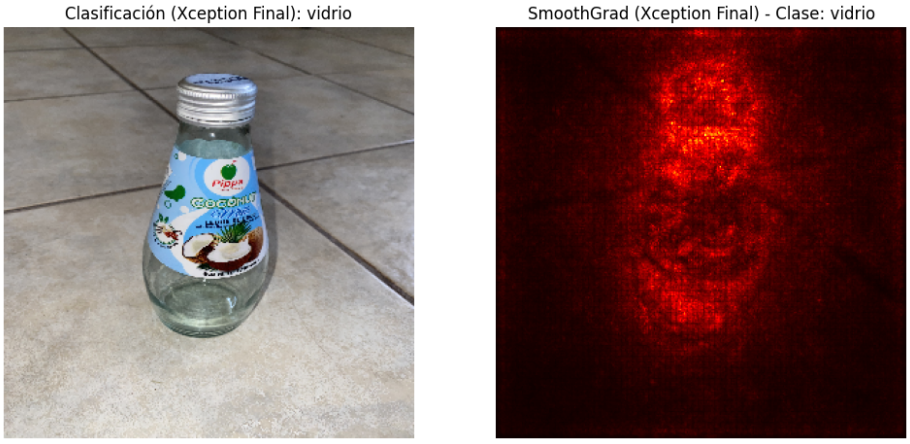


Figura 6.94. Clasificación de un residuo real perteneciente a la clase *Glass*

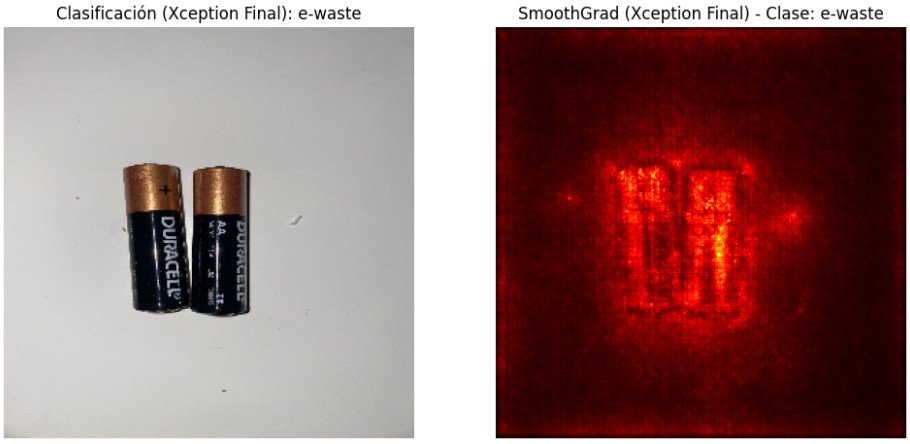


Figura 6.95. Clasificación de un residuo real perteneciente a la clase *Battery*

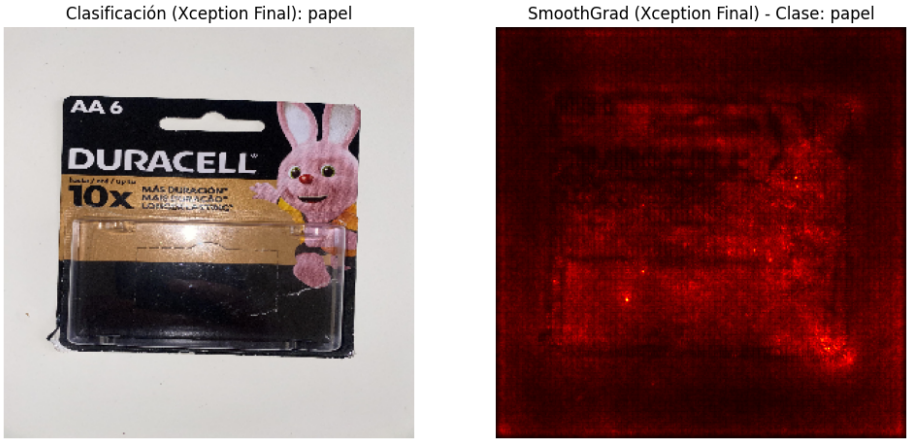


Figura 6.96. Clasificación de un residuo real perteneciente a la clase *Paper*

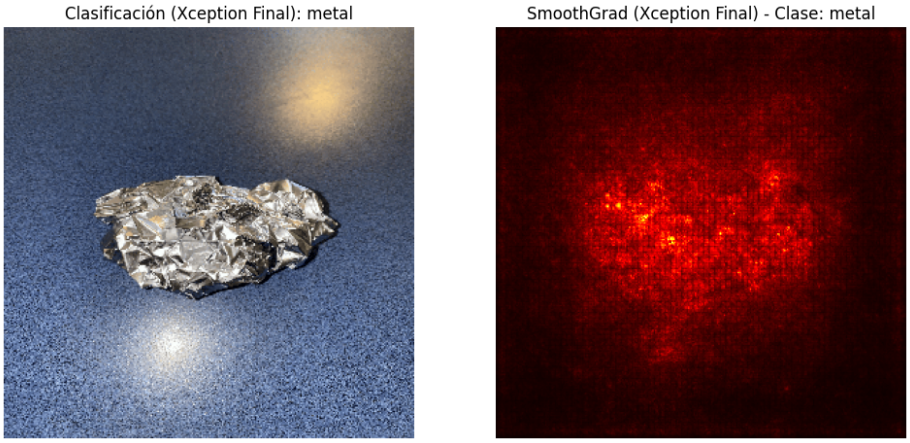


Figura 6.97. Clasificación de un residuo real perteneciente a la clase *Metal*.

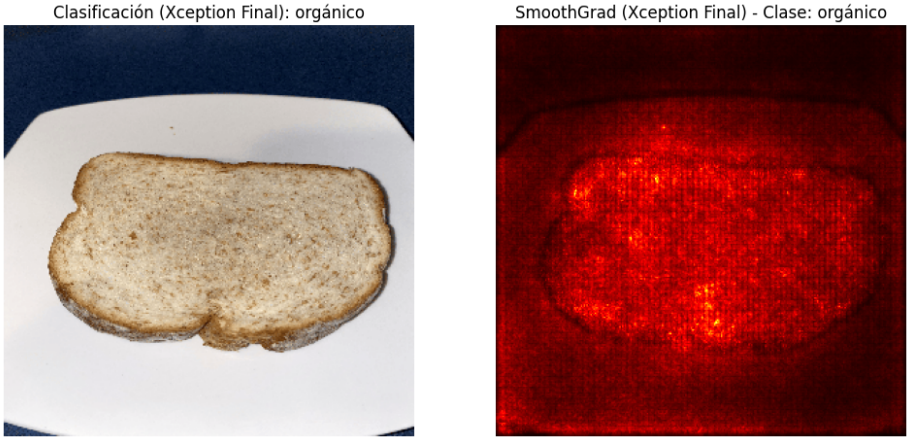


Figura 6.98. Clasificación de un residuo real perteneciente a la clase *Organic*

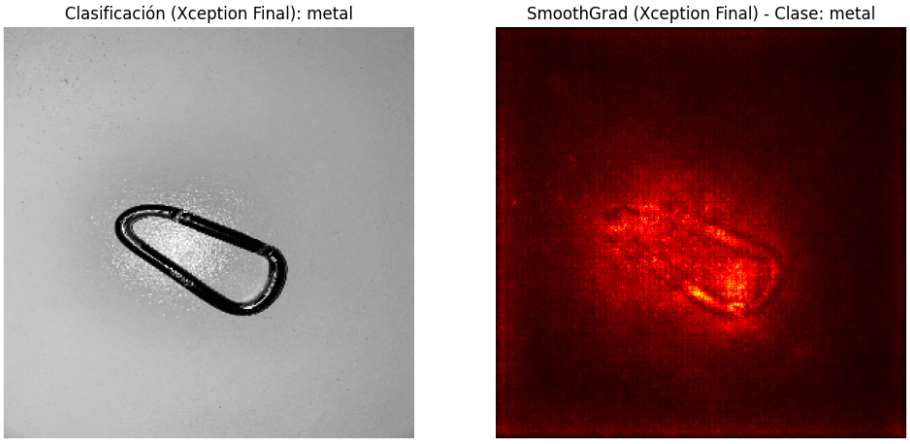


Figura 6.99. Clasificación de un residuo real perteneciente a la clase *Metal*

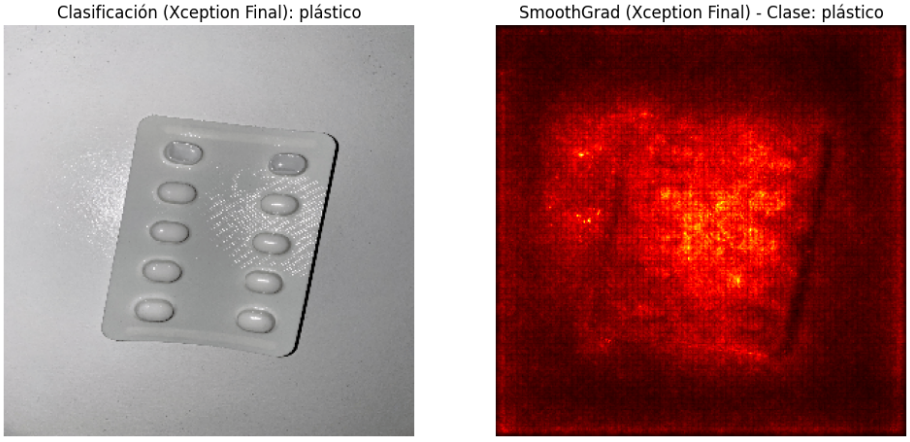


Figura 6.100. Clasificación de un residuo real perteneciente a la clase *Plastic*

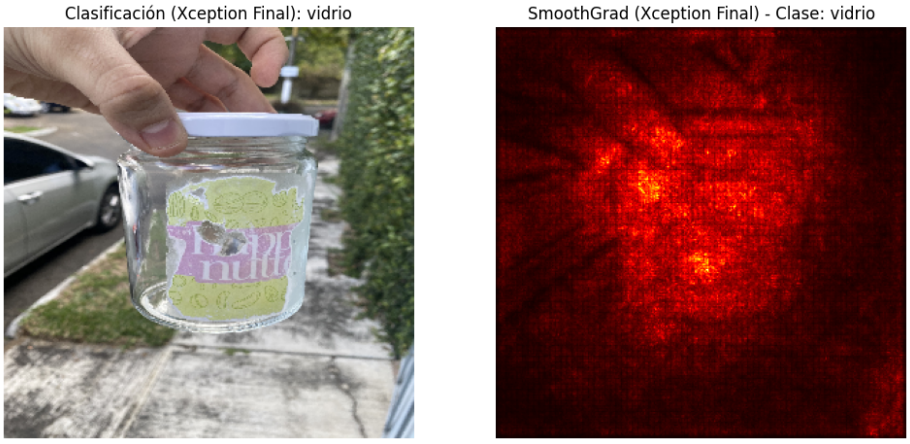


Figura 6.101. Clasificación de un residuo real perteneciente a la clase *Glass*

Figura 6.102. Clasificación de un residuo real perteneciente a la clase *Plastic*

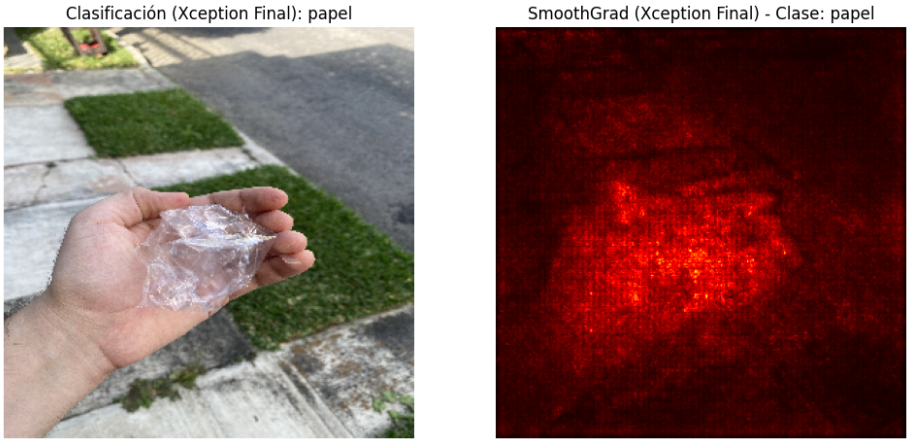


Figura 6.103. Clasificación de un residuo real perteneciente a la clase *Plastic*, pero que fue clasificado como *Paper*

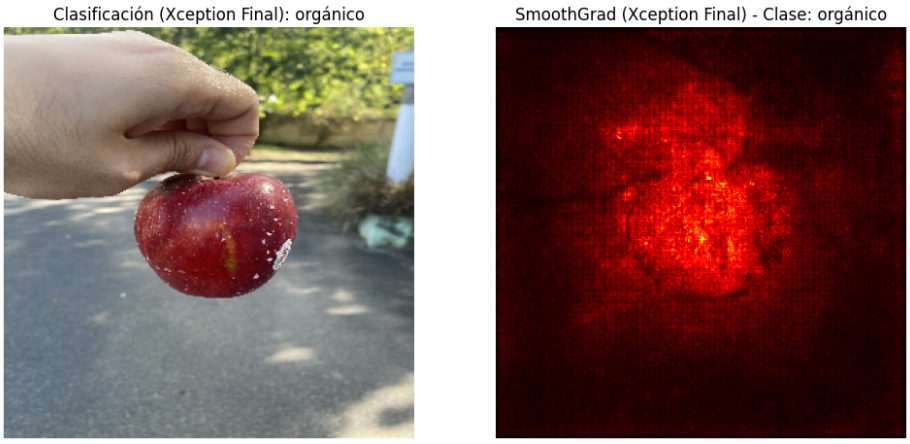


Figura 6.104. Clasificación de un residuo real perteneciente a la clase *Organic*

Tiempo promedio de clasificación por imagen El tiempo promedio de clasificación por imagen es una métrica importante para evaluar la eficiencia del modelo en aplicaciones prácticas. A continuación, se detalla el rendimiento temporal del modelo *Xception* final, destacando su viabilidad para su uso en sistemas en tiempo real.

```
# Función para predecir la clase de una imagen y mostrarla junto con su etiqueta
def classify_and_display_image(image_path):
    # Cargar y mostrar la imagen
    img = image.load_img(image_path, target_size=(299, 299))
    plt.imshow(img)
    plt.axis('off')

    # Convertir la imagen a array y preprocesarla
    img_array = image.img_to_array(img)
    img_array = np.expand_dims(img_array, axis=0)
    img_array = preprocess_input(img_array)

    # Tomar el tiempo de inicio
    start_time = time.time()

    # Hacer la predicción
    predictions = model.predict(img_array)

    # Tomar el tiempo de fin y calcular el tiempo
    end_time = time.time()
    classification_time = end_time - start_time
    classification_times.append(classification_time)

    # Obtener la clase predicha (índice) y traducirla al español
    predicted_class_idx = np.argmax(predictions[0])
    predicted_class = list(class_translation.keys())[predicted_class_idx]
    translated_class = class_translation[predicted_class]

    # Mostrar la etiqueta traducida
    plt.title(f'Clasificación (\textit{Xception} Final): {translated_class}')
    plt.show()

# Clasificar y mostrar cada imagen
for image_file in image_files:
    classify_and_display_image(image_file)

# Calcular el tiempo promedio de clasificación
average_time = np.mean(classification_times)
print(f'Tiempo promedio de clasificación por imagen (\textit{Xception} Final): {average_time:.4f} segundos'
```

Tiempo promedio de clasificación por imagen: 0.1945 segundos

6.3.2. Resultados del *chatbot* inteligente

En esta sección se presentan los resultados obtenidos a partir de la evaluación del *chatbot* desarrollado para la clasificación de residuos. Los resultados se basan en las respuestas de 7 expertos de la Unidad de Control Ambiental de la Municipalidad de la Ciudad de Guatemala, quienes evaluaron el desempeño del *chatbot* en cinco criterios clave mediante una rúbrica de 1 a 5 puntos.

Resultados de la encuesta por criterio

Precisión en las respuestas La precisión en las respuestas fue evaluada para determinar si el *chatbot* proporciona información correcta y clara en relación con la clasificación de residuos. De acuerdo con los resultados, el 57.1 % de los participantes otorgó una puntuación de 5, y el 42.9 % calificó con un 4.

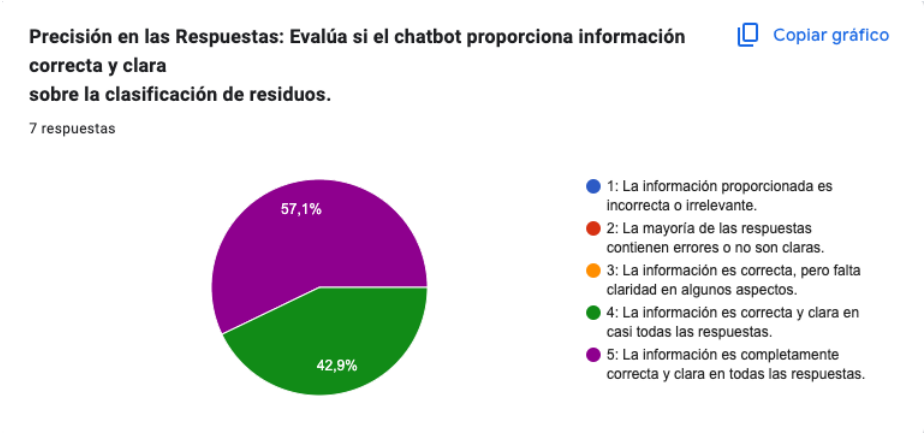


Figura 6.105. Evaluación de la precisión de las respuestas del *chatbot*

Relevancia temática de las respuestas Este criterio mide el grado de alineación de las respuestas con el tema de clasificación de residuos y reciclaje. Los resultados muestran que el 57.1 % de los evaluadores otorgó una puntuación de 5, mientras que el 42.9 % calificó con un 4.

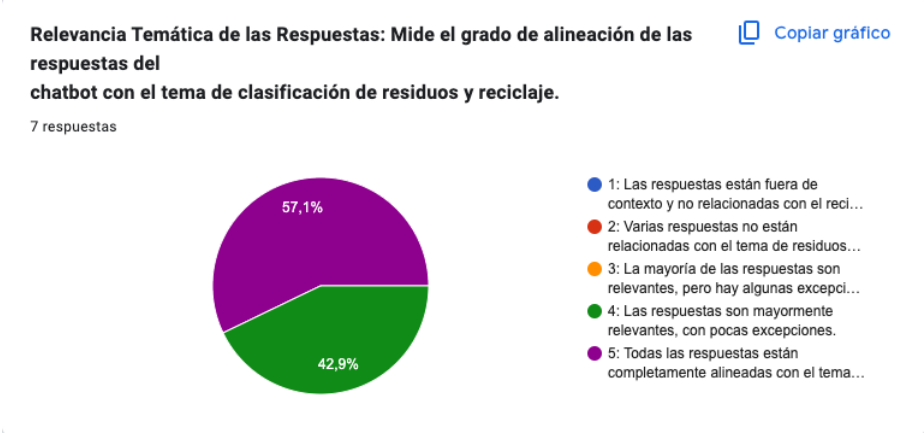


Figura 6.106. Evaluación de la relevancia temática de las respuestas

Capacidad para rechazar preguntas fuera de tema El 85.7 % de los participantes calificó esta capacidad con un 5, mientras que el 14.3 % la evaluó con un 4, lo que sugiere que el *chatbot* logra rechazar preguntas fuera de contexto de manera educada y clara.

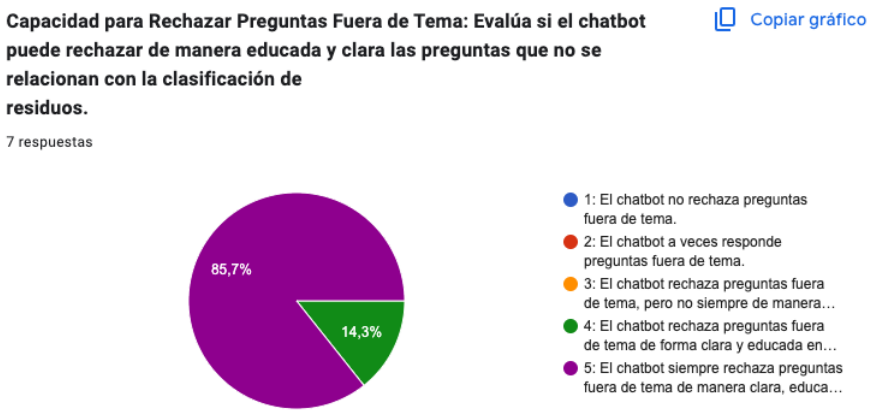


Figura 6.107. Evaluación de la capacidad del *chatbot* para rechazar preguntas fuera de tema

Calidad de las recomendaciones sobre manejo de residuos El 71.4 % de los evaluadores calificaron la calidad de las recomendaciones con un 4, mientras que el 28.6 % le otorgó un 5, destacando la utilidad y relevancia de las sugerencias proporcionadas.

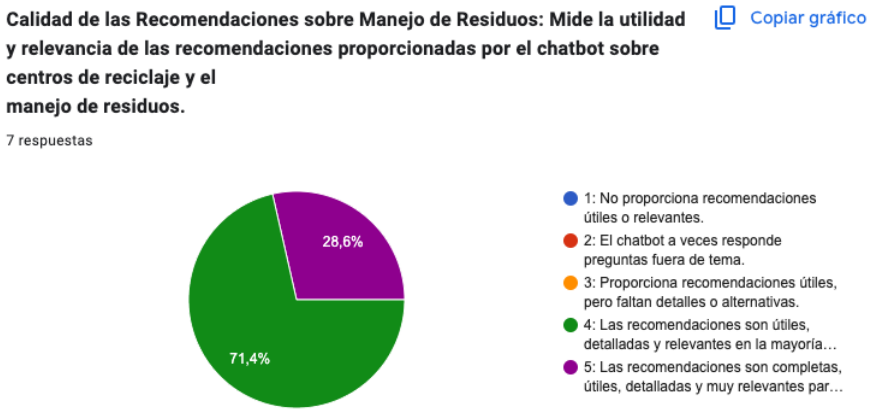


Figura 6.108. Evaluación de la calidad de las recomendaciones proporcionadas

Consistencia en el estilo de las respuestas El 57.1 % de los evaluadores calificaron este aspecto con un 5, mientras que el 42.9 % otorgó un 4, indicando que el chatbot mantiene una consistencia estilística adecuada.

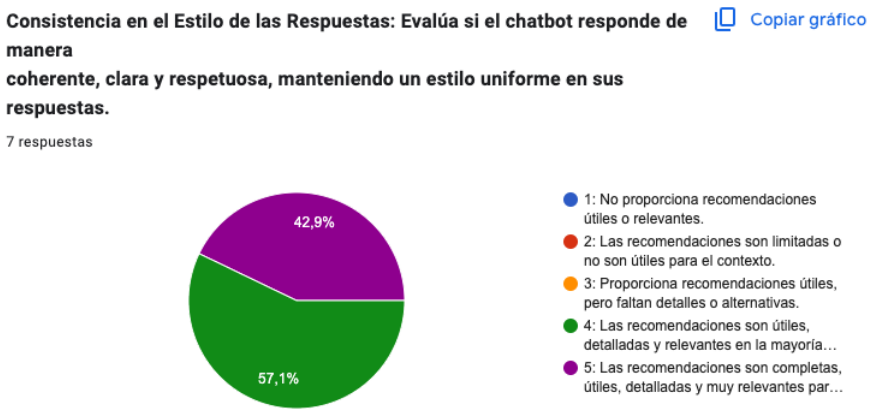


Figura 6.109. Evaluación de la consistencia en el estilo de las respuestas

Puntaje total y precisión general

Se calcularon los puntajes totales y el porcentaje de precisión de cada participante en la evaluación del *chatbot*. Los resultados se presentan en el cuadro 6.8.

Cuadro 6.8. Puntaje total y porcentaje de precisión de cada participante en la evaluación del *chatbot*

Participante	Precisión	Relevancia	Rechazo	Recomendaciones	Consistencia	Total	Promedio (%)
Experto 1	4	4	5	4	5	22	88.0
Experto 2	5	5	5	5	5	25	100.0
Experto 3	4	5	5	4	4	22	88.0
Experto 4	5	5	5	5	5	25	100.0
Experto 5	5	5	5	4	4	23	92.0
Experto 6	4	4	4	4	4	20	80.0
Experto 7	5	4	5	4	4	22	88.0

El puntaje total promedio de precisión y calidad del *chatbot* fue de **90.86 %**, indicando un alto nivel de precisión y efectividad en la asistencia para la clasificación de residuos.

Tiempos de respuesta del *chatbot*

Los tiempos de respuesta del *chatbot* se evaluaron a través de 10 pruebas consecutivas con la misma pregunta. Los resultados se resumen en el cuadro 6.9.

Cuadro 6.9. Tiempos de respuesta del *chatbot* en cada prueba realizada

Prueba	Tiempo de Respuesta (segundos)
1	2.67
2	0.82
3	2.09
4	1.08
5	2.52
6	0.72
7	1.79
8	0.93
9	2.44
10	0.96
Promedio	**1.70**

El tiempo promedio de respuesta fue de **1.7 segundos**, lo cual es adecuado para una experiencia de usuario rápida y fluida.

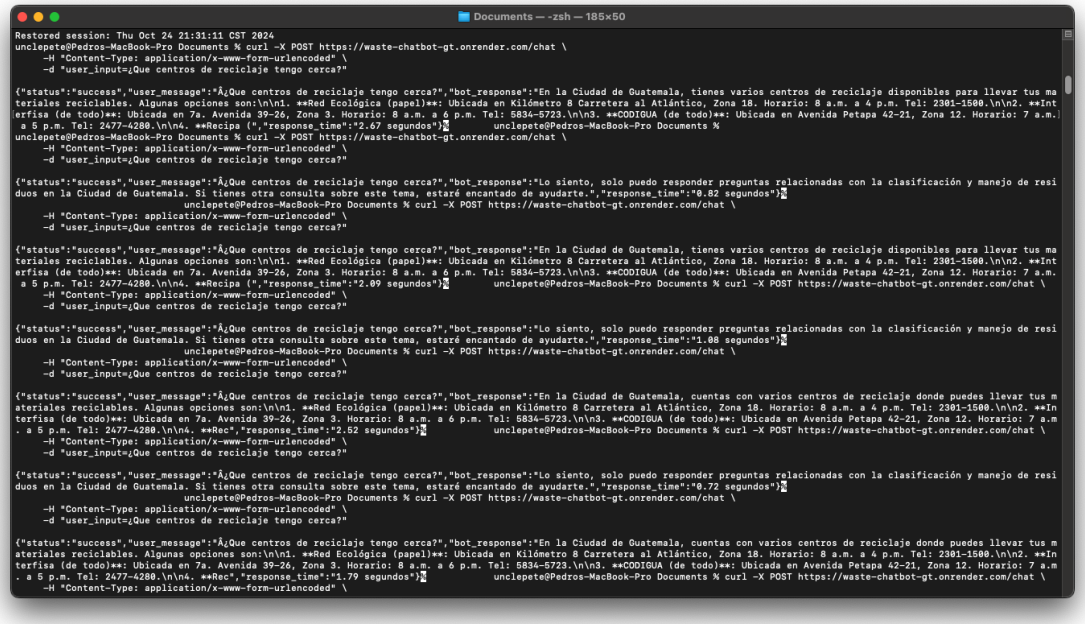


Figura 6.110. Tiempo promedio de respuesta del *chatbot*.

Pruebas de preguntas y respuestas del *chatbot*

A continuación, se muestran ejemplos de interacciones con el *chatbot*, junto con las respuestas proporcionadas. Estas pruebas demuestran cómo el *chatbot* gestiona consultas relacionadas con la clasificación de residuos y preguntas fuera de contexto.

Cuadro 6.10. Pruebas de interacción con el *chatbot* y sus respuestas

Pregunta del Usuario	**Respuesta del *Chatbot***	**Tiempo de Respuesta**
¿Que puedo hacer con mis residuos plásticos?	Puedes clasificar tus residuos plásticos como reciclables y llevarlos a un centro de reciclaje adecuado en la Ciudad de Guatemala. Algunos centros que reciben materiales plásticos son Interfisa, CODIGUA y Recipa. Recuerda que es importante que los residuos plásticos estén limpios y secos para facilitar su reciclaje. Si necesitas más información sobre cómo manejar tus residuos plásticos, no dudes en preguntar.	2.14 segundos
¿Cómo debo clasificar los residuos orgánicos?	Los residuos orgánicos deben ser clasificados en la categoría de .ºrgánicos"(Verde). Estos residuos incluyen materiales de origen animal o vegetal que se descomponen de forma natural, como cáscaras de frutas y verduras, restos de comida, hojas secas y restos de jardinería. Es importante separar los residuos orgánicos de los reciclables y no reciclables para facilitar su tratamiento adecuado. Si tienes más preguntas sobre la clasificación de residuos orgánicos o necesitas información adicional, estaré encantado de ayudarte.	1.70 segundos
¿Dónde puedo reciclar el vidrio en la Ciudad de Guatemala?	En la Ciudad de Guatemala, puedes reciclar el vidrio en el centro de reciclaje Interfisa, que acepta materiales de vidrio entre otros tipos de residuos reciclables. Interfisa está ubicada en 7a. Avenida 39-26, Zona 3, y su horario de atención es de 8 a.m. a 6 p.m. Si necesitas más información sobre el reciclaje de vidrio u otros materiales, no dudes en preguntar.	1.64 segundos
¿Cuáles son las reglas para desechar desechos electrónicos?	Las normas para desechar desechos electrónicos en la Ciudad de Guatemala implican no mezclarlos con la basura común, ya que contienen componentes que pueden ser perjudiciales para el medio ambiente si no se manejan correctamente. Para desechar desechos electrónicos como computadoras, teléfonos celulares, baterías, y otros aparatos electrónicos, se recomienda llevarlos a centros de reciclaje especializados en este tipo de residuos. En la Ciudad de Guatemala, puedes contactar con centros de reciclaje como Interfisa, CODIGUA y Recipa, quienes pueden recibir y manejar adecuadamente estos desechos electrónicos.	2.21 segundos
¿Cuál es la capital de Francia?	Lo siento, solo puedo responder preguntas relacionadas con la clasificación y manejo de residuos en la Ciudad de Guatemala. Si tienes alguna consulta sobre ese tema, estaré encantado de ayudarte.	0.89 segundos

6.4. Resultados de seguridad

6.4.1. Educación y orientación

Durante el proyecto, se logró un avance significativo en el área de educación y orientación, pasando del nivel 1 al nivel 2 en el marco de *OWASP SAMM*. Al inicio, se implementaron prácticas básicas de sensibilización en seguridad, y mi rol como campeón de seguridad fue fundamental para alcanzar este progreso.

Como parte de mis responsabilidades, apoyé al equipo en la adopción de prácticas seguras en el ciclo de vida del *software*. Esto incluyó la creación de guías específicas adaptadas a los roles de cada miembro y a las tecnologías empleadas. El avance al nivel 2 implicó la educación de todo el personal mediante materiales y guías personalizadas, según sus funciones y las tecnologías utilizadas. A lo largo de este proceso, los miembros del equipo me consultaban sobre diversas tecnologías y buenas prácticas en el desarrollo seguro. Para asegurar respuestas adecuadas, investigué las opciones disponibles y recomendé las mejores soluciones con argumentos basados en seguridad y eficiencia.

Este enfoque de educación y orientación demostró su efectividad mediante los resultados de las pruebas de *phishing* y el cumplimiento del estándar de cobertura de código. Ambas actividades reflejan la importancia de la gobernanza y el establecimiento de prácticas seguras, conforme a las directrices de *OWASP SAMM*.

Resultados en la prueba de *phishing*

Como parte de las actividades de capacitación, se llevaron a cabo pruebas de *phishing* para evaluar la capacidad del equipo en la identificación de correos maliciosos. La primera prueba mostró un promedio de 4.5 respuestas correctas, lo que indicó la necesidad de mejorar en esta área. Tras una sesión de formación enfocada en la identificación de *phishing*, se realizó una segunda prueba el 14 de septiembre, en la cual se observó una mejora significativa, alcanzando un promedio de 8.25 respuestas correctas. Finalmente se realizo una prueba sorpresa para asegurar que el equipo recordara los consejos y técnicas impartidas del cuál se observo un promedio de 9 respuestas correctas.

Nombre	**Primer test de *phishing***	**Segundo test de *phishing***	**Último test de *phishing***
Pedro	4	9	8
Oscar	4	8	10
Santiago	5	9	9
Gabriel	4	7	9

Cuadro 6.11. Resultados de los *tests* de *phishing*

La mejora en los resultados de las pruebas de *phishing* evidenció el impacto positivo de las actividades de educación y orientación en la capacidad del equipo para identificar amenazas, reduciendo así el riesgo de caídas en ataques de *phishing*. Este logro contribuye al cumplimiento del área de gobernanza en educación y orientación de *OWASP SAMM*.

6.4.2. Calidad del código y mejora continua

Además de la capacitación en seguridad, se realizaron mejoras significativas en la calidad del código y en las prácticas de desarrollo. A medida que se proporcionaban recomendaciones sobre mejores prácticas y mantenibilidad del código, utilizando herramientas como *SonarQube* para el análisis de calidad, el equipo adoptó consistentemente técnicas para reducir la deuda técnica.

La reducción de vulnerabilidades detectadas a lo largo del proyecto muestra el cumplimiento en la gestión de defectos de la categoría de implementación de *OWASP SAMM*. Esta práctica implicó que el equipo no solo resolviera vulnerabilidades detectadas sino que implementara mejoras preventivas para evitar futuros problemas de seguridad. Estas mejoras incluyeron la implementación de pruebas unitarias para asegurar que el código funcionara según lo esperado, lo cual no solo incrementó la cobertura de pruebas, sino que también permitió identificar y resolver problemas de manera proactiva.

Resultados en la gráfica de vulnerabilidades

La progresión de vulnerabilidades detectadas en *SonarQube* y *SonarCloud* mostró una disminución significativa en los problemas de seguridad, evidenciando el enfoque proactivo del equipo en la resolución de problemas y en la adopción de mejores prácticas de codificación segura. La siguiente gráfica muestra la reducción de vulnerabilidades a lo largo del proyecto.

Figura 6.111. Gráfica de vulnerabilidades detectadas en *SonarQube*

Figura 6.112. Gráfica de vulnerabilidades detectadas en *SonarCloud*

La disminución en la cantidad de vulnerabilidades evidencia el éxito en la gestión de defectos, cumpliendo con los principios de *OWASP SAMM* en la categoría de implementación.

Cobertura de código

El proyecto logró superar el 80 % de cobertura de código en *SonarCloud*, cumpliendo con los estándares de la herramienta y con los objetivos planteados en el área de Verificación de *OWASP SAMM*. La cobertura del *backend* alcanzó un 82.3 % mientras que el *frontend* logró un 80.4 %, logrando una cobertura general de 82 % en el proyecto.

	Lines of Code	Security	Reliability	Maintainability	Security Hotspots	Coverage	Duplications
mobile-app							
eco-scan-backend	1,431	0	0	1	0	82.3%	0.0%
eco-scan-frontend	1,025	0	0	7	1	80.4%	0.0%

2 of 2 shown

Figura 6.113. Cobertura del código en *SonarCloud*

Figura 6.114. Gráfica de cobertura en *SonarCloud*

Este logro no solo muestra la adopción de buenas prácticas de desarrollo seguro, sino que también cumple con el área de pruebas de seguridad en verificación de *OWASP SAMM*, garantizando una adecuada validación y cobertura en el proyecto.

Code smells

Los *code smells* son problemas de diseño o prácticas de programación que pueden afectar la calidad del código pero no necesariamente son errores de funcionamiento. A lo largo del proyecto, se realizó un monitoreo constante de estos *code smells*, lo cual permitió al equipo mejorar la estructura y diseño del código para mantenerlo limpio y eficiente.

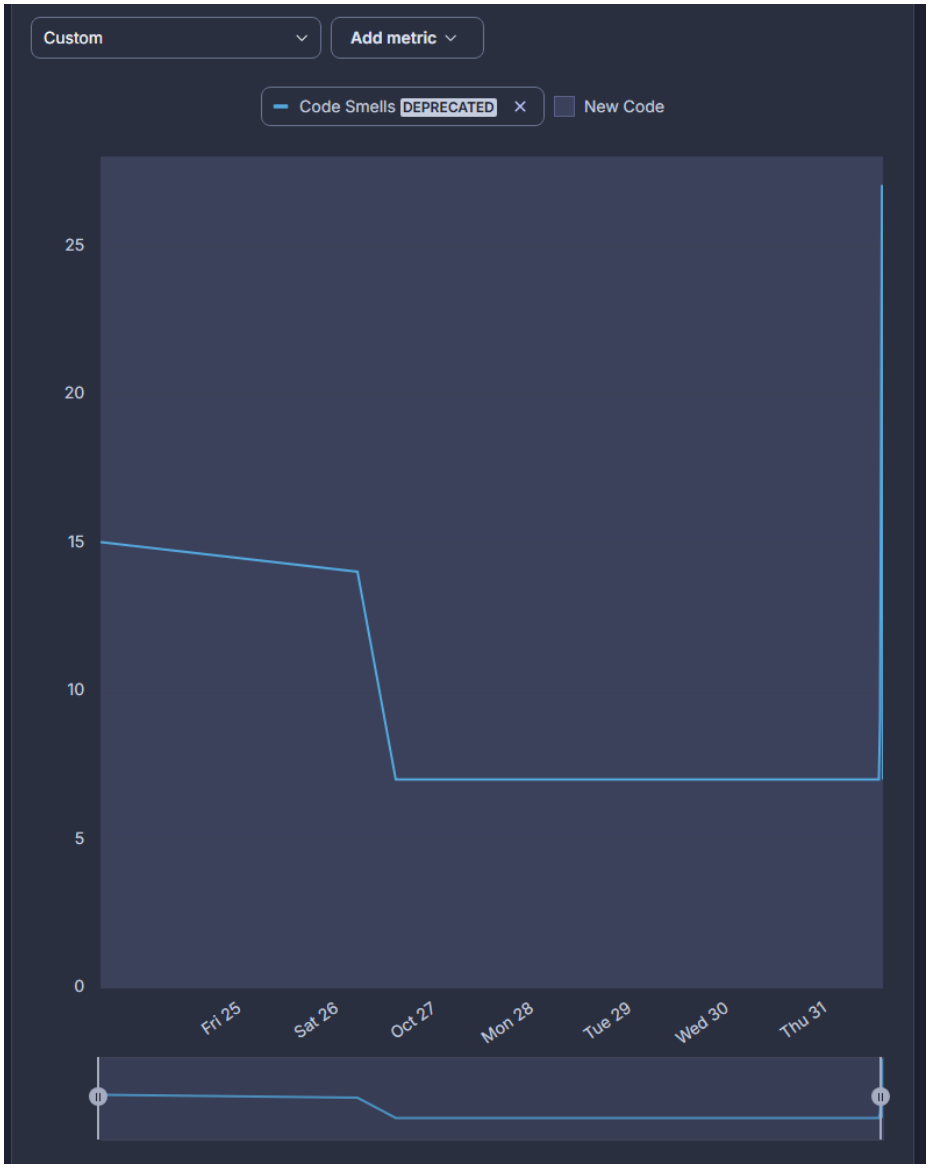


Figura 6.115. Gráfica de *code smell* en *SonarCloud*

Código duplicado

El código duplicado mide el porcentaje de líneas de código que están repetidas en la base de código. Durante el análisis, se identificaron y redujeron las duplicaciones, lo cual contribuyó a una mejor mantenibilidad y eficiencia del proyecto. Este proceso fue fundamental para optimizar el rendimiento y la claridad del código.

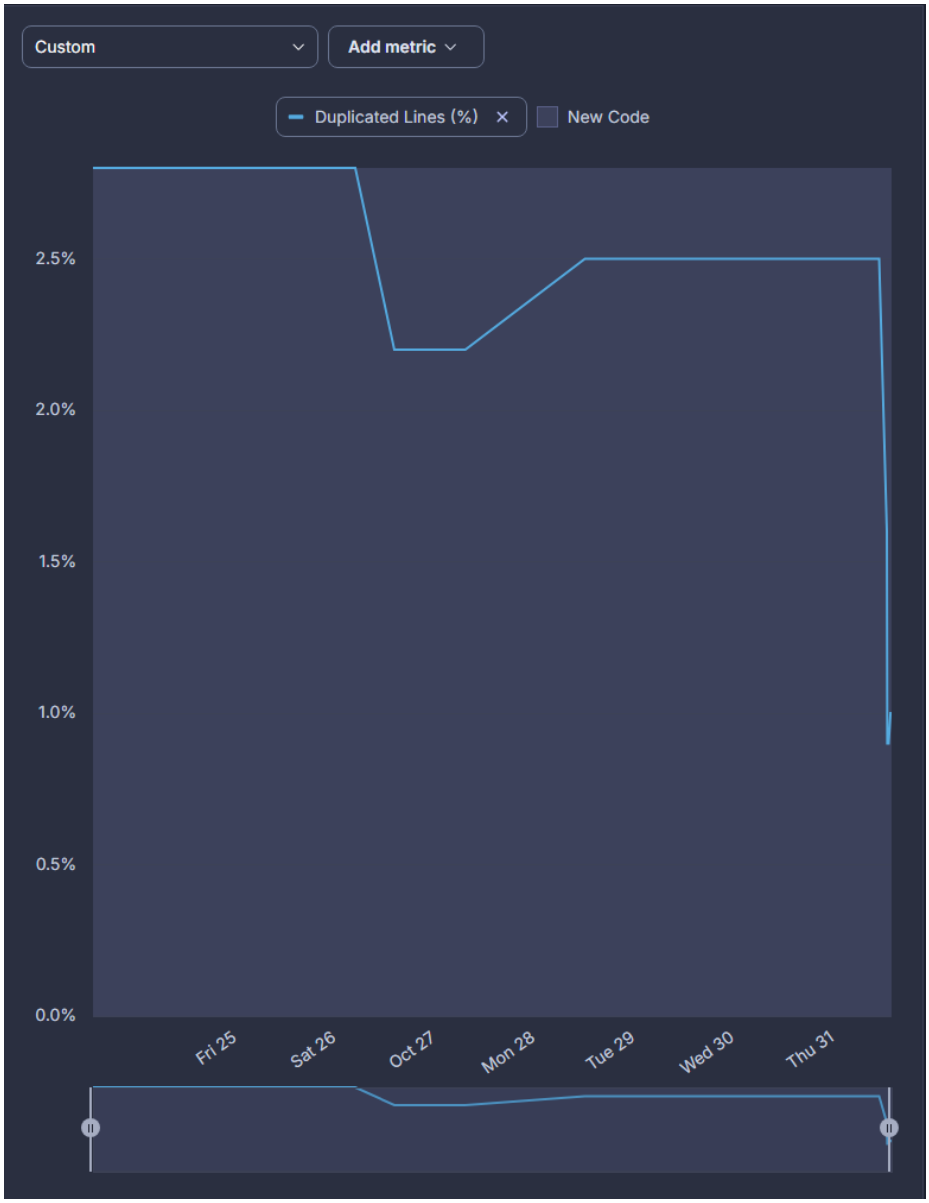


Figura 6.116. Gráfica de líneas de código duplicadas en *SonarCloud*

Complejidad ciclomática

La complejidad ciclómatica mide la complejidad de los métodos, clases o módulos del proyecto, calculando el número de caminos distintos que se pueden recorrer en el código. Mantener la complejidad bajo control es esencial para asegurar que el código sea fácil de entender, probar y mantener. Se realizaron esfuerzos para simplificar los métodos y módulos más complejos.

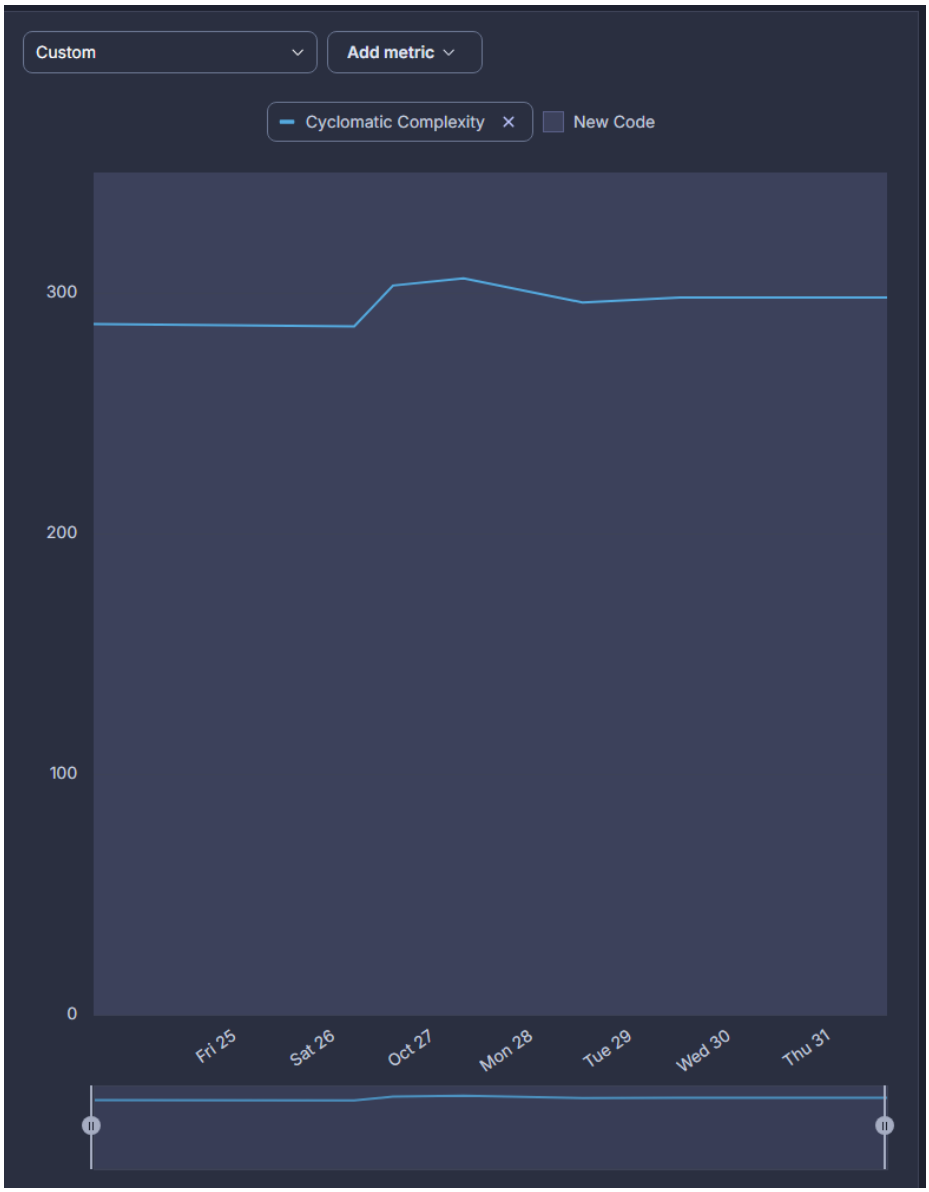


Figura 6.117. Gráfica de complejidad en *SonarCloud*

Densidad de comentarios

La densidad de comentarios mide el porcentaje de comentarios en el código. Tener un equilibrio en esta métrica es importante: demasiados comentarios pueden ser un signo de código complicado, mientras que muy pocos comentarios pueden indicar una falta de documentación. Se trabajó en mantener este equilibrio, asegurando que el código fuera comprensible sin necesidad de comentarios excesivos, pero con la documentación necesaria para la claridad.

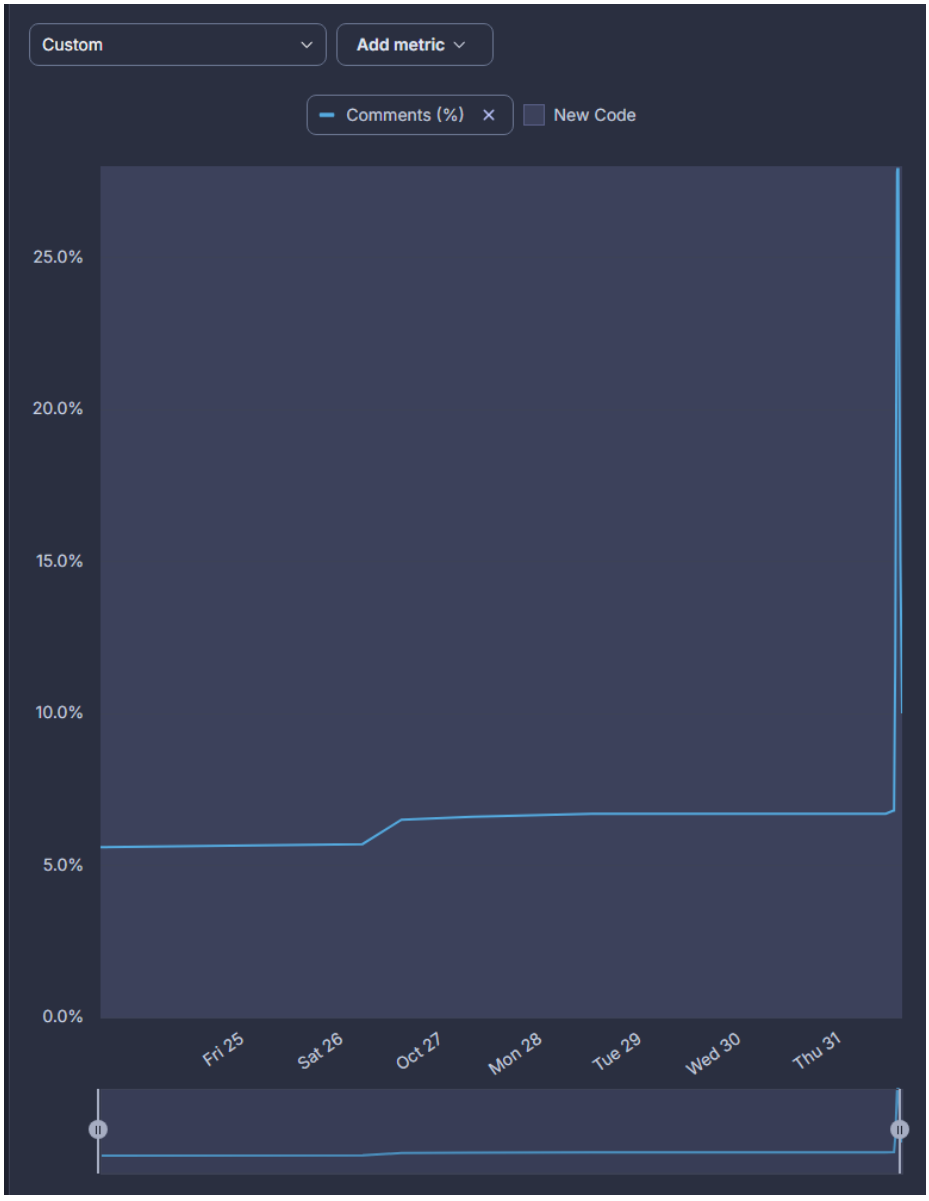


Figura 6.118. Gráfica de densidad de comentarios en *SonarCloud*

6.4.3. Escaneo de vulnerabilidades en aplicación móvil

Se logró realizar el escaneo de la aplicación móvil utilizando las herramientas *MobSF* y *AppSweep*, permitiendo tanto el análisis estático como el dinámico de la aplicación. Este proceso de escaneo reveló vulnerabilidades adicionales en el código móvil que no habían sido identificadas previamente en otras fases del proyecto.

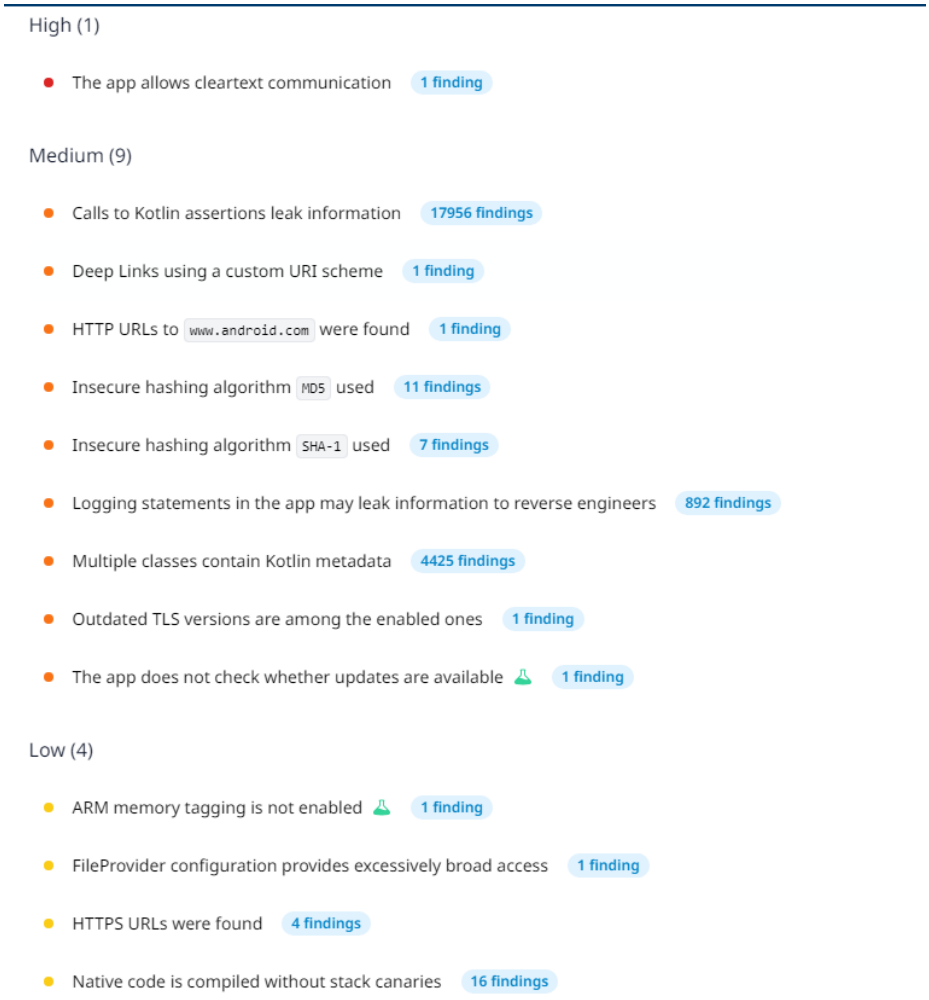


Figura 6.119. Vulnerabilidades detectadas en *AppSweep*

Estas nuevas vulnerabilidades detectadas abren el camino para una mejora continua del código y proporcionan una perspectiva valiosa sobre otras áreas de vulnerabilidades potenciales en el desarrollo de aplicaciones móviles. Este análisis contribuye no solo al cumplimiento de estándares de seguridad, sino también a un aprendizaje continuo, apoyando la implementación de buenas prácticas en código y configuraciones futuras que refuercen la protección y fiabilidad de la aplicación móvil.

6.4.4. Análisis del código de los modelos de IA

En el trabajo realizado por el encargado de los modelos de inteligencia artificial, Pedro Pablo, se observó un manejo adecuado y consistente en la implementación del código. A lo largo de todo el proyecto, Pedro Pablo siguió rigurosamente las buenas prácticas de desarrollo seguro, lo que incluyó la modularidad del código, el uso de comentarios claros y la implementación de técnicas de evaluación continua del modelo.

Análisis de seguridad y cumplimiento de estándares

Los análisis de seguridad realizados con *SonarQube*, tanto al inicio como al final del proyecto, no detectaron problemas ni vulnerabilidades en el código. Este resultado refleja un cumplimiento constante de los estándares de calidad y seguridad, atribuible al enfoque proactivo de Pedro Pablo en la implementación de técnicas de monitoreo, tales como gráficos para la precisión y la pérdida del entrenamiento, y la validación del rendimiento mediante una matriz de confusión y un reporte de clasificación.

Prácticas de seguridad en el *chatbot*

Las prácticas seguidas en el código del *chatbot* también evidenciaron un enfoque integral en la seguridad, con una gestión adecuada del registro de *logs* y el manejo seguro de excepciones. Estas medidas contribuyeron a evitar vulnerabilidades comunes, y la política de limitar el historial del chat y el uso seguro de las claves de la *API* reforzaron la protección del sistema, logrando un entorno seguro y confiable.

6.4.5. Análisis de permisos en *AWS*

Durante el desarrollo del proyecto, se realizó un análisis de los permisos otorgados a cada miembro del equipo en *AWS* para garantizar la aplicación del principio de mínimo privilegio. Este análisis se llevó a cabo en conjunto con el encargado de la infraestructura, Gabriel, y se identificaron varios permisos que podrían ser ajustados para mejorar la seguridad general.

Oscar López: permisos de *DatabaseAdministrator*

Oscar López, encargado del *backend* del proyecto, fue asignado al rol de ***DatabaseAdministrator***, el cual contiene permisos específicos para la gestión de bases de datos en *AWS*. Sin embargo, se detectaron algunos permisos que no eran necesarios para sus tareas, como:

- `rds:*` – permisos relacionados con *Amazon RDS* que podrían ser reducidos para limitar el acceso.
- `s3:CreateBucket` – permiso innecesario para la creación de *buckets* en *S3*.
- `iam` – permiso crítico que no debería ser accesible sin restricciones adicionales.

Después de analizar estos permisos, se discutió con Gabriel Vicente, encargado de la infraestructura sobre la necesidad de restringir o eliminar estas capacidades, ajustando el rol de Oscar para cumplir con las mejores prácticas de seguridad.

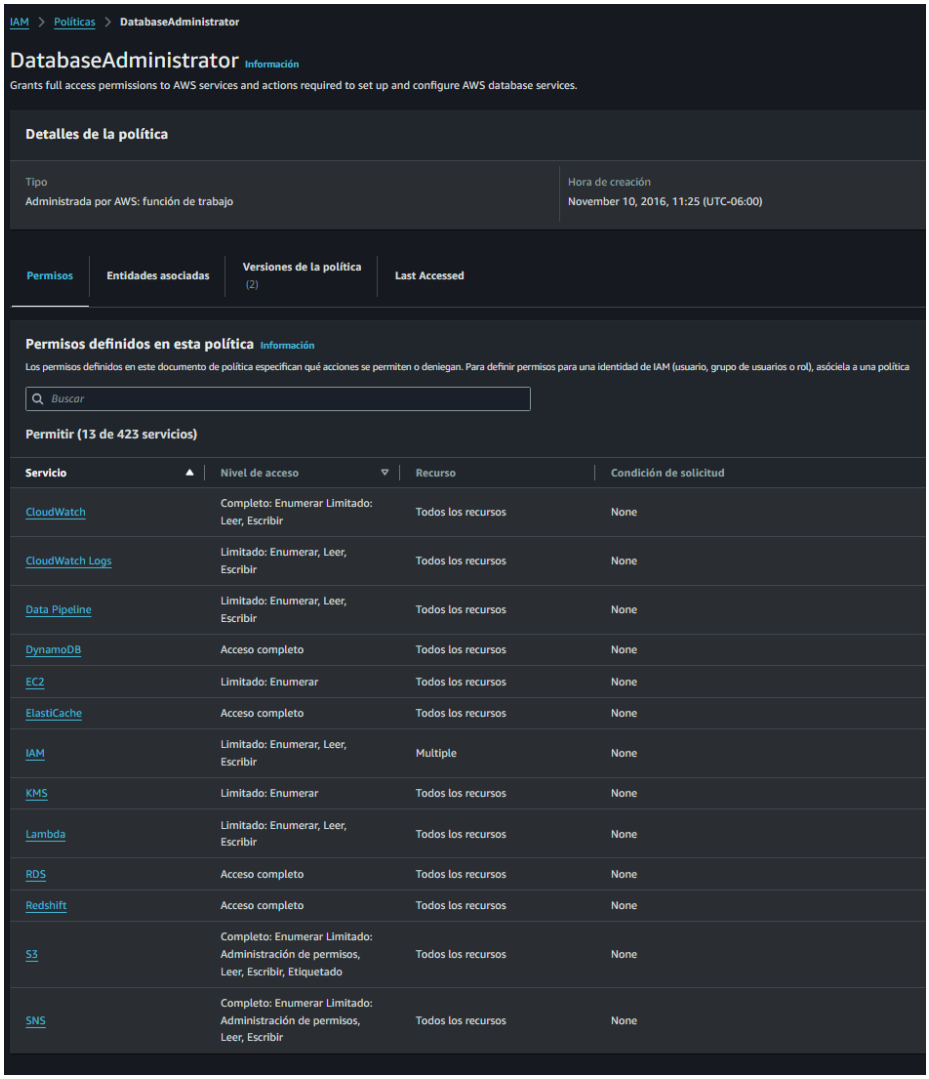


Figura 6.120. Permisos del rol *DatabaseAdministrator* asignados a Oscar López

Pedro Pablo: permisos de *SystemAdministrator*

Pedro Pablo, responsable de la implementación de los modelos de inteligencia artificial, fue asignado al rol de ***SystemAdministrator***. Durante el análisis, se encontraron permisos adicionales que no eran necesarios para sus responsabilidades, como:

- `trustadvisor:*` – permisos para *AWS Trusted Advisor* que no son relevantes para su función.
- `config:*` – permisos relacionados con *AWS Config* que tampoco son necesarios.
- `iam` – permiso que podría ser removido para fortalecer la seguridad.

Se discutieron estos hallazgos con Gabriel, y se acordó realizar las modificaciones pertinentes para asegurar que Pedro Pablo tuviera únicamente los permisos imprescindibles para su trabajo.

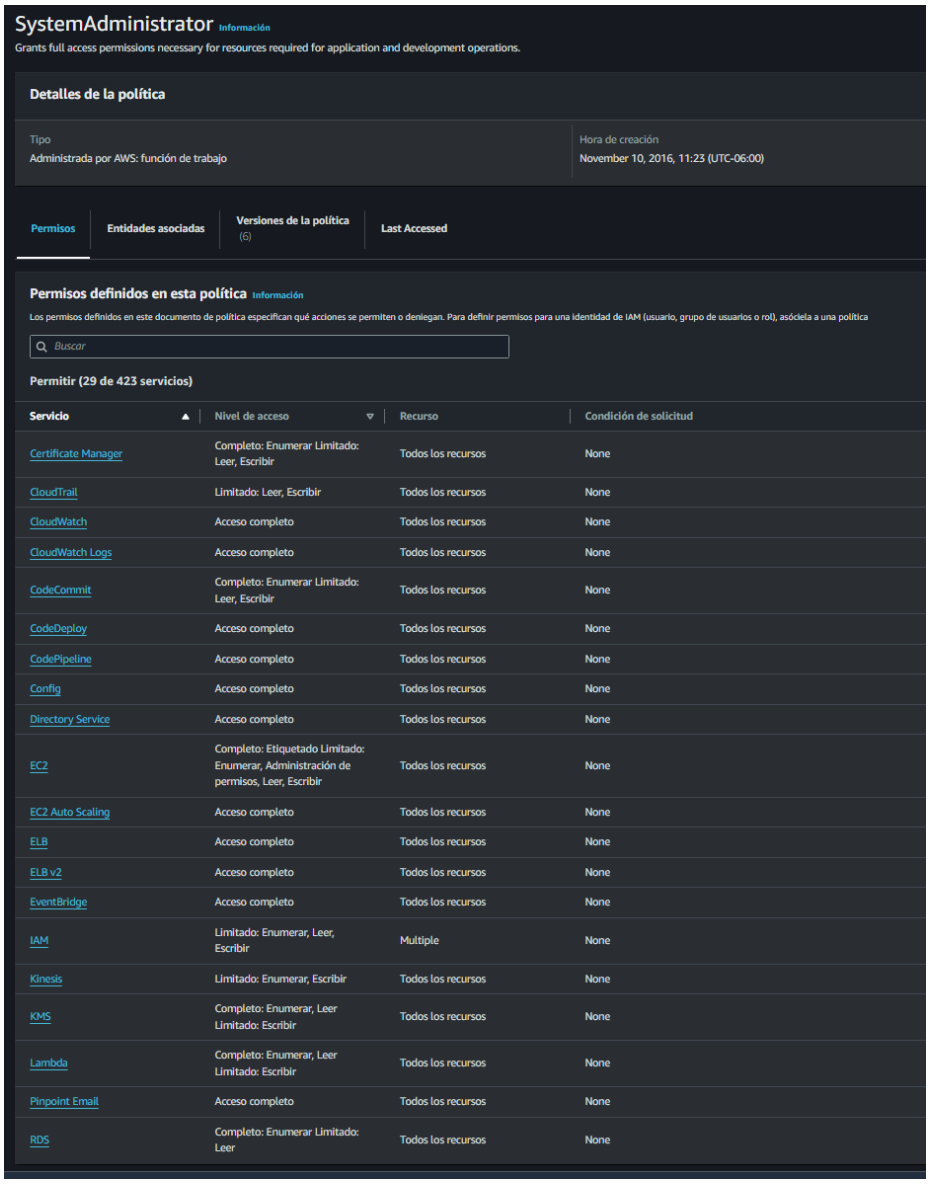


Figura 6.121. Permisos del rol *SystemAdministrator* asignados a Pedro Pablo

Santiago Taracena y Yongbum Park: permisos de *ReadOnlyAccess*

Santiago Taracena y yo (Yongbum Park) fuimos asignados al rol de ***ReadOnlyAccess***, el cual es adecuado para nuestras funciones, ya que no requerimos realizar cambios en el sistema. Santiago no necesita subir nada directamente, ya que todo se gestionará a través de *GitHub Actions* para su parte del código, por lo que este rol es suficiente para sus tareas de configuración y observación.

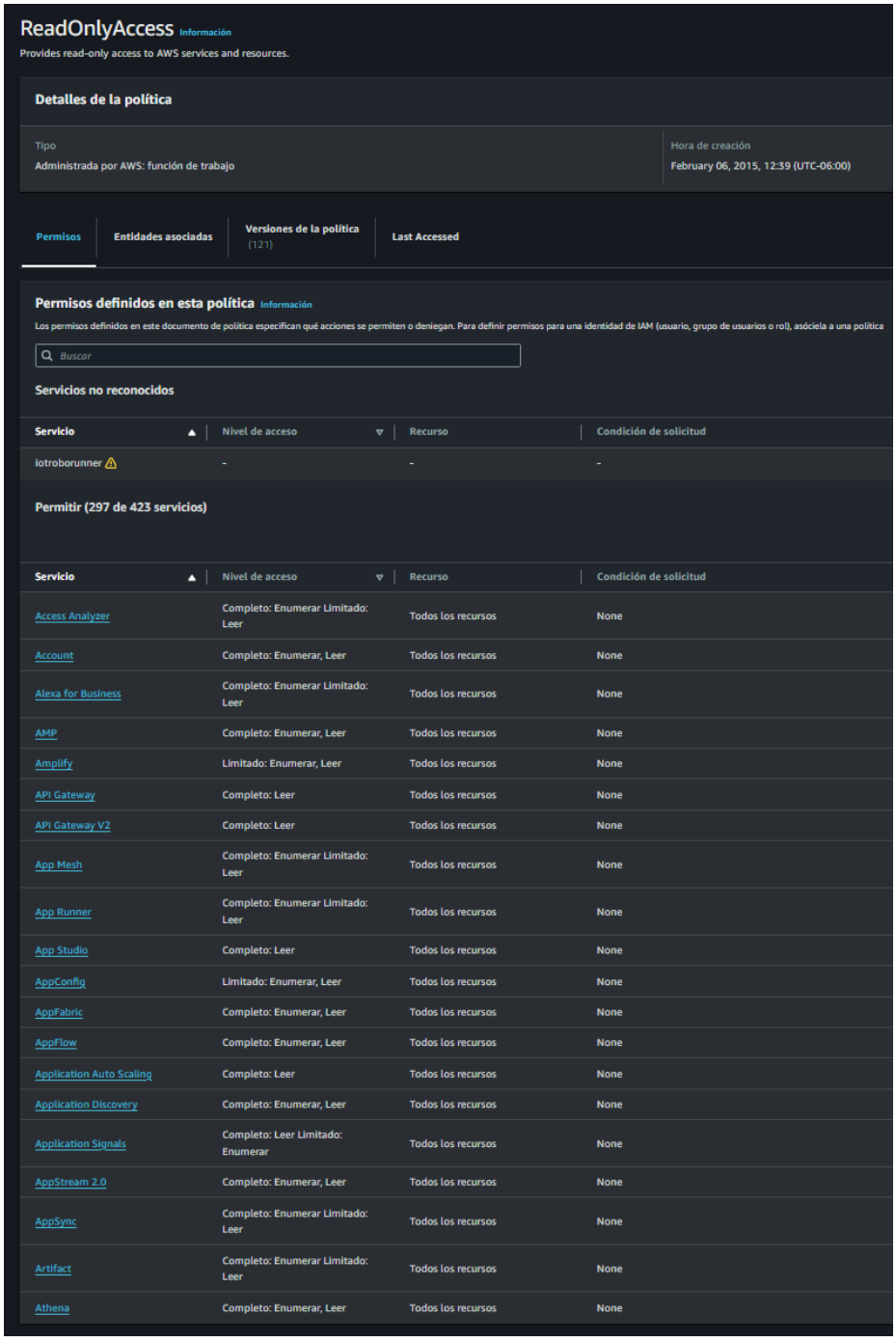


Figura 6.122. Permisos del rol *ReadOnlyAccess* asignados a Santiago Taracena y Yongbum Park

6.5. Resultados de infraestructura en la nube

6.5.1. Recursos utilizados

A continuación, se detalla el listado de recursos utilizados en la implementación de la solución en *AWS*. Estos recursos fueron seleccionados y configurados para cumplir con los requisitos de rendimiento, escalabilidad y seguridad necesarios para el desarrollo y despliegue de la aplicación.

- ***AWS Elastic Beanstalk***: utilizado para el despliegue de la *API* en un entorno administrado que permite el escalado automático y el balanceo de carga. *Elastic Beanstalk* proporciona una plataforma simplificada para gestionar el *backend* de la aplicación sin necesidad de administrar manualmente los servidores.
- ***Amazon EC2 (Elastic Compute Cloud)***: como parte de *Elastic Beanstalk*, se utilizaron instancias EC2 para ejecutar la *API* en *Node.js* con *Express*. Las instancias se configuraron de acuerdo a los requisitos de afluencia y se escalaron automáticamente en función de la demanda, utilizando tipos de instancia como `t3.micro`, `t3.small` y `t3.medium`.
- ***Amazon RDS* (*Relational Database Service*)**: implementado con *PostgreSQL* como motor de base de datos. *Amazon RDS* permite la gestión de bases de datos relacionales de forma escalable y segura. Las instancias de *RDS* fueron configuradas para soportar diferentes niveles de carga, utilizando configuraciones de almacenamiento *SSD* y opciones de *Multi-AZ* para garantizar la disponibilidad en caso de alta afluencia.
- ***AWS SageMaker***: utilizado para desplegar un modelo de inteligencia artificial que procesa tanto imágenes como texto y genera respuestas en formato textual. *SageMaker* facilitó la administración y escalabilidad del modelo, permitiendo el procesamiento de inferencias en tiempo real con instancias `ml.t2.medium`, `ml.m5.large`, y `ml.m5.xlarge` según el nivel de demanda.
- ***AWS Amplify***: implementado para facilitar la autenticación de usuarios mediante cuentas de *Google* y *Facebook*. *Amplify* ofrece una integración simplificada con estos proveedores de identidad a través de la *API* de autenticación, permitiendo gestionar las solicitudes de inicio de sesión y almacenar *tokens* de autenticación de forma segura.
- ***Amazon S3* (*Simple Storage Service*)**: utilizado para el almacenamiento de archivos, tales como imágenes y documentos asociados a la aplicación. *Amazon S3* proporciona almacenamiento escalable y seguro, permitiendo almacenar grandes volúmenes de datos y optimizando costos mediante el almacenamiento en clases como `Standard` y `Infrequent Access`.
- ***Amazon CloudWatch***: implementado para monitorear el rendimiento de los servicios, con métricas detalladas de uso de *CPU*, memoria, tiempo de respuesta, y otras estadísticas clave. *CloudWatch* permite configurar alertas para identificar posibles problemas de rendimiento y permite escalar recursos automáticamente en función de la carga.
- ***Amazon IAM (Identity and Access Management)***: utilizado para gestionar los permisos de acceso a los recursos de *AWS, IAM* permite configurar políticas detalladas de control de acceso y proporciona una capa adicional de seguridad, limitando el acceso a roles específicos en la infraestructura de la aplicación.
- ***Amazon Cognito***: empleado en combinación con *AWS Amplify* para gestionar la autenticación de usuarios de forma segura. *Cognito* simplifica la creación y gestión de usuarios y proporciona opciones de autenticación multifactor (MFA), lo cual añade una capa de seguridad adicional al proceso de inicio de sesión.

- ***Expo* y *EAS* (*Expo Application Services*)**: herramientas utilizadas en el *frontend* de la aplicación, desarrollado en *React Native*, para generar los certificados de firma y gestionar las configuraciones necesarias para los inicios de sesión con *Google* y *Facebook*. EAS facilita la generación de claves de firma y asegura que la aplicación cumpla con los requisitos de seguridad en plataformas *Android* e *iOS*.

Este listado de recursos fue fundamental para el desarrollo y despliegue exitoso de la aplicación en *AWS*, proporcionando una infraestructura escalable y segura que soporta las funcionalidades requeridas en el entorno de producción.

6.5.2. Resultados de funcionalidad

A continuación se presentan los resultados de haber implementado los recursos anteriormente mencionados.

IAM Identity Center

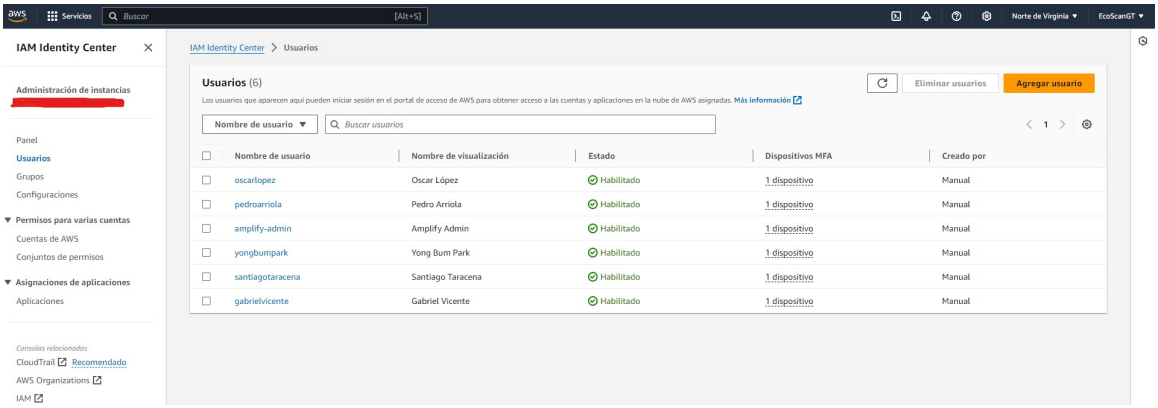


Figura 6.123. Usuarios *IAM Identity Center*

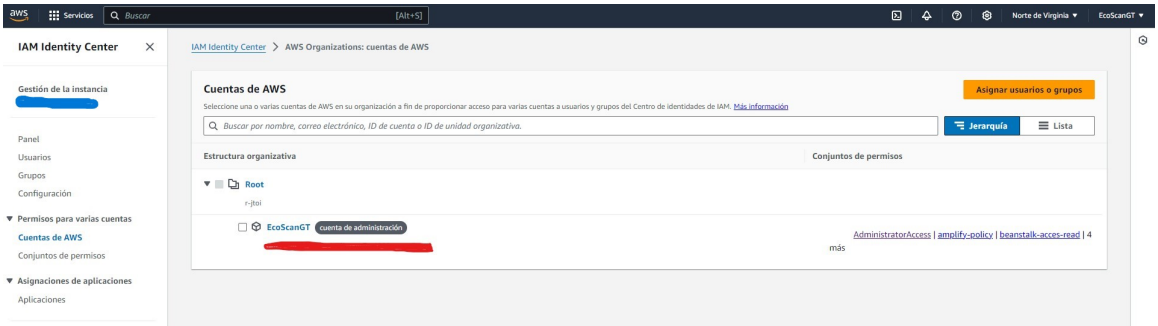


Figura 6.124. Administración de cuenta única por múltiples usuarios

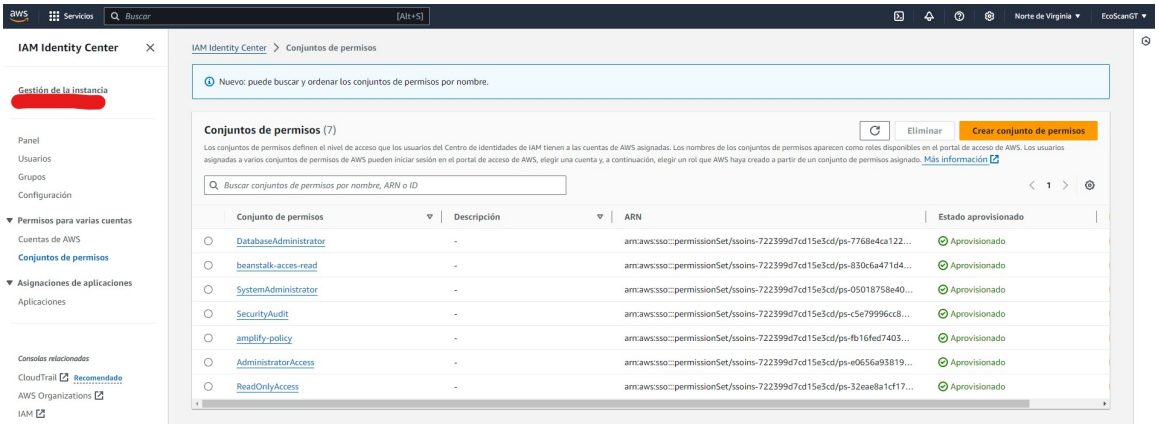


Figura 6.125. Configuración de políticas para cada usuario

Elastic Beanstalk

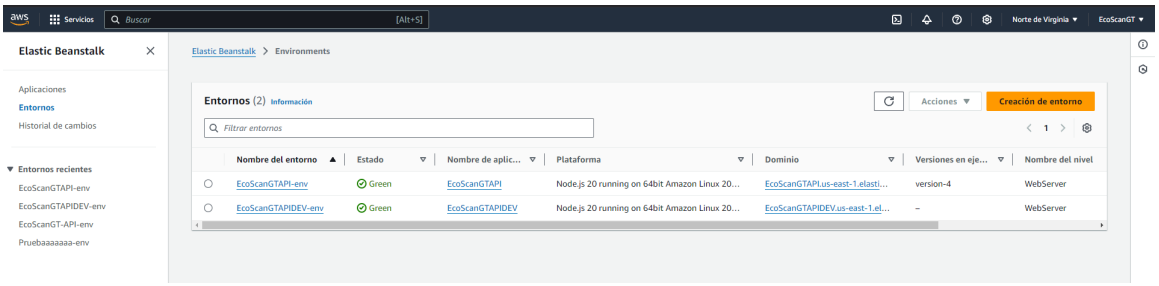


Figura 6.126. Entorno de creación de *Elastic Beanstalk*

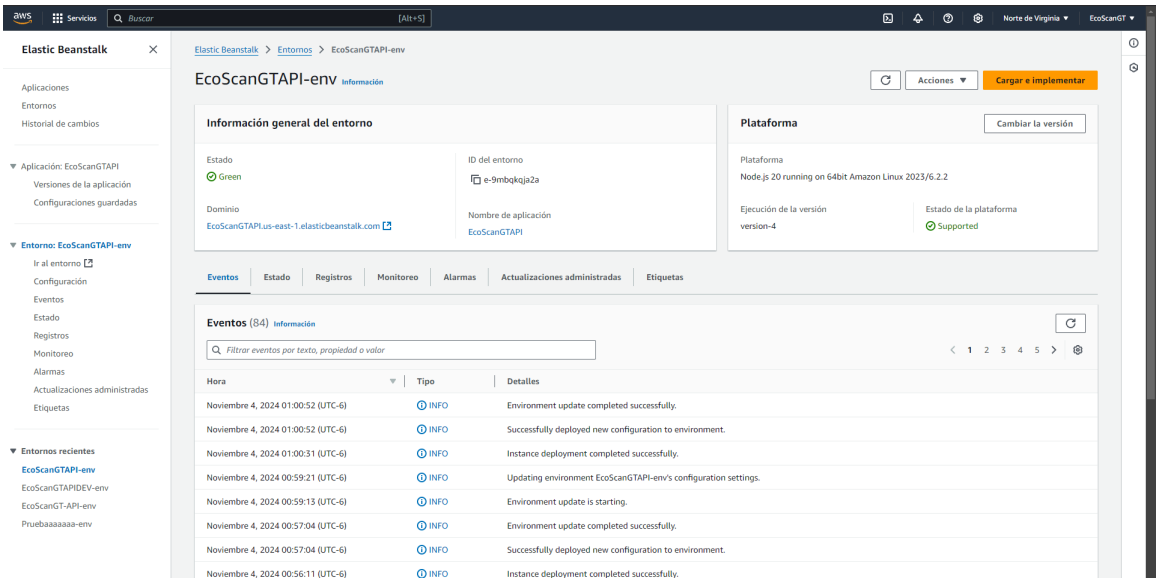


Figura 6.127. Estado de producción de interfaz de programación de aplicación

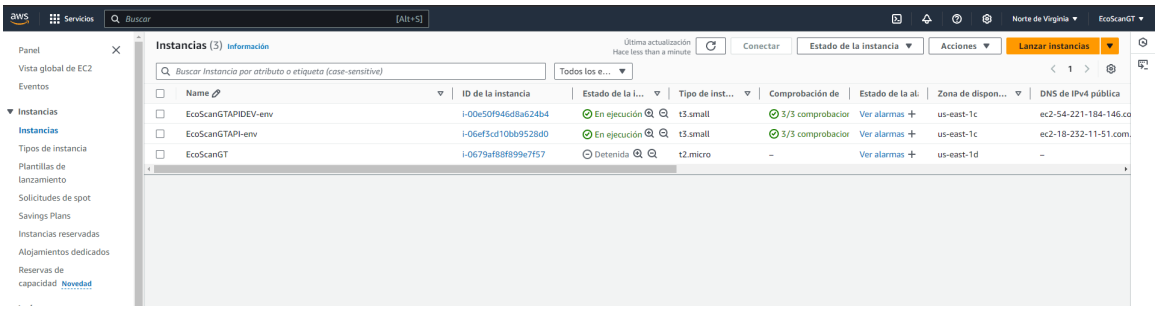


Figura 6.128. EC2 que aloja *API Express*

Recursos asociados a *Elastic Beanstalk*

RDS Postgres SQL

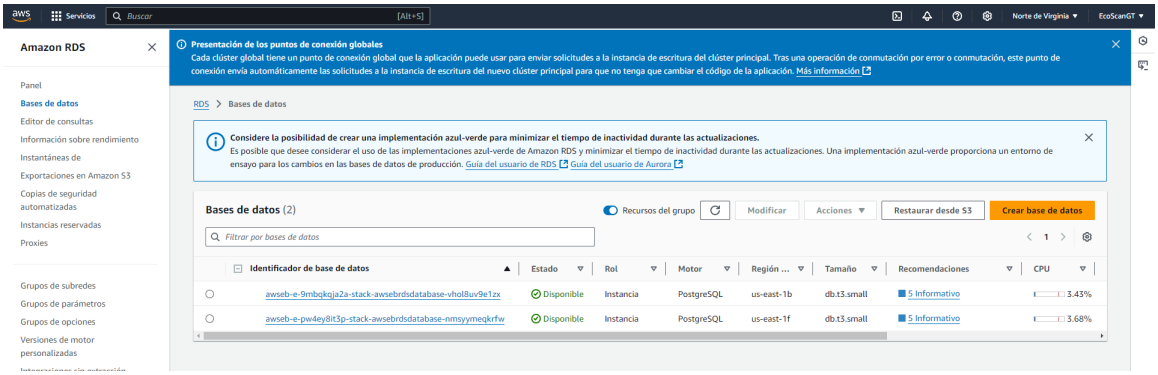


Figura 6.129. Base de datos disponibles para *Elastic Beanstalk*

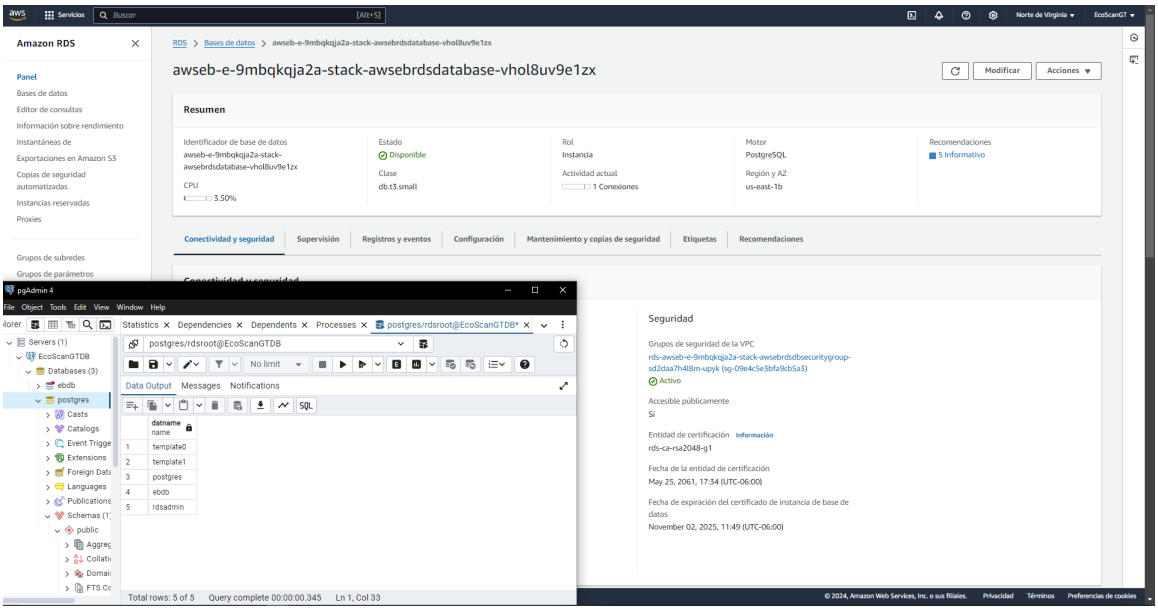


Figura 6.130. Conexión para base de datos de producción

Sage Maker

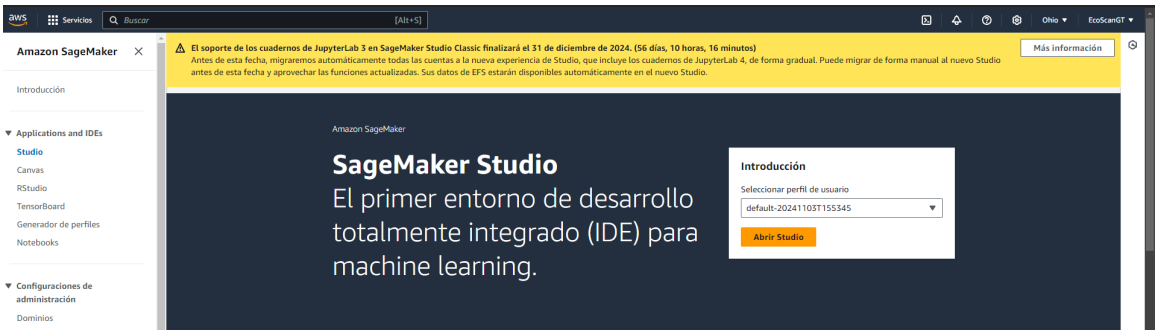


Figura 6.131. Configuracion de perfil para *Sage Maker*

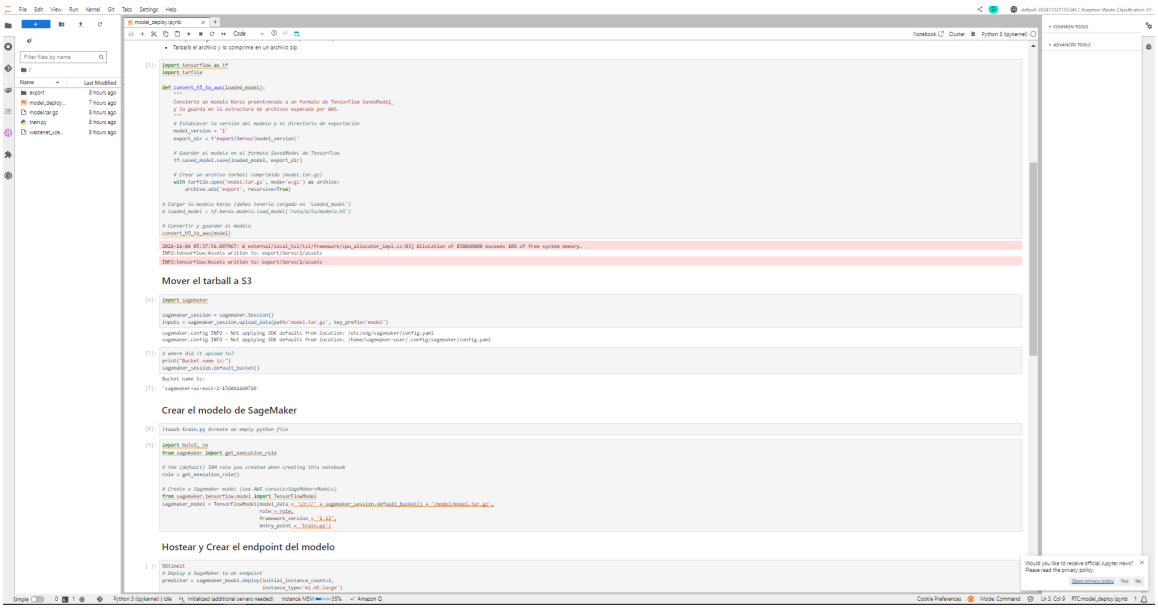


Figura 6.132. *Deployment* de modelo h5 para producción

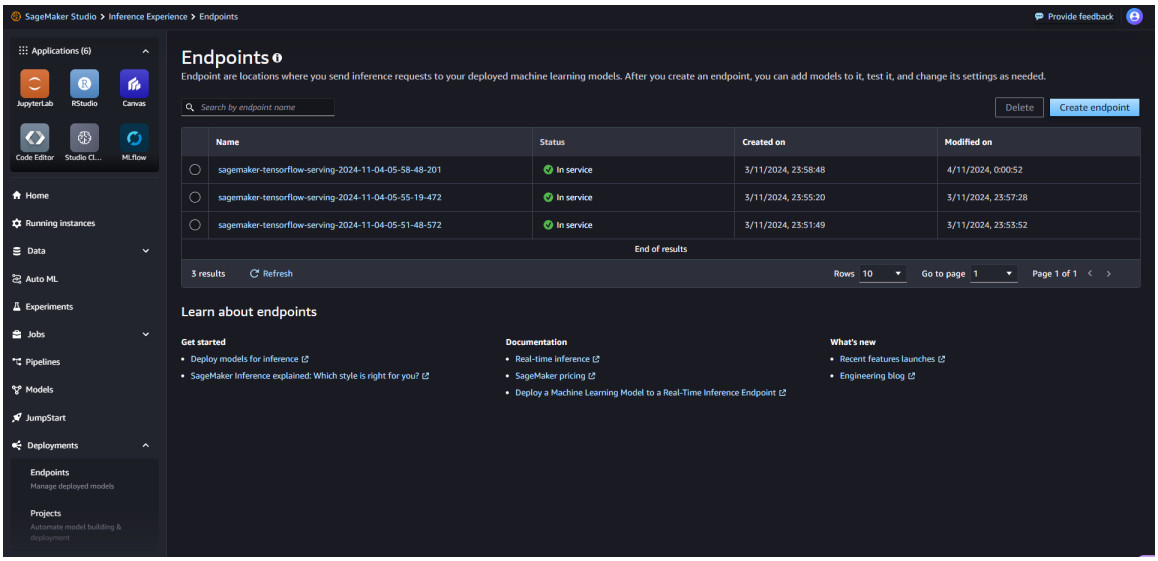


Figura 6.133. *Endpoints* disponibles para consultar el modelo

6.5.3. Diagrama de infraestructura final

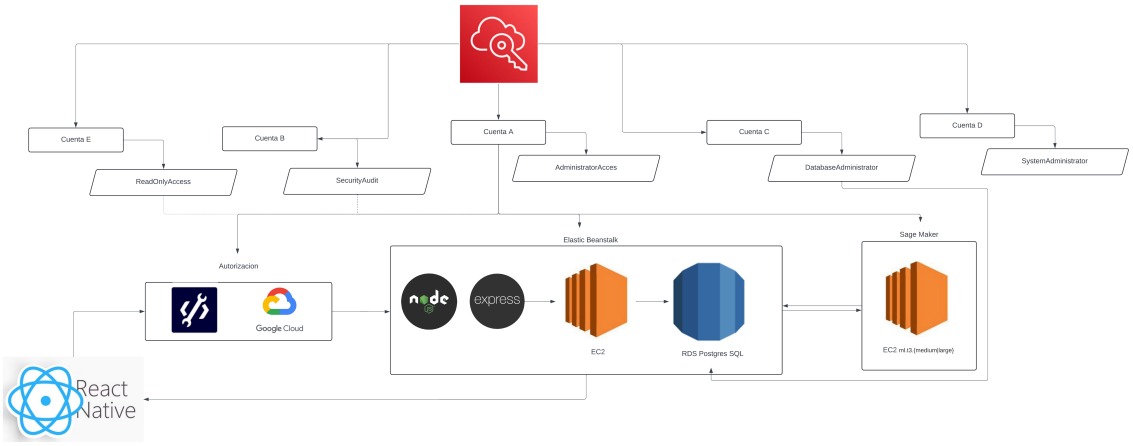


Figura 6.134. Diagrama de infraestructura en la nube

6.5.4. Pruebas de rendimiento

A continuación se presentan los resultados de pruebas de rendimiento que ponían bajo estrés el *API* y cómo respondió esta a las solicitudes. Durante la sesión de prueba, se crearon un total de 4200 usuarios virtuales (VUs), de los cuales 4197 completaron sus sesiones exitosamente. Se registraron 3 errores de tiempo de espera (*ETIMEDOUT*), reflejando incidencias mínimas durante la ejecución. El sistema manejó un volumen considerable de solicitudes, alcanzando una tasa de 59 solicitudes por segundo y un total de 4200 solicitudes HTTP enviadas.

Las respuestas HTTP exitosas (código 200) fueron 4197, lo que indica un alto grado de estabilidad y capacidad de respuesta bajo carga. Los datos descargados durante la prueba ascendieron a 186991 bytes, mostrando la eficiencia en la transferencia de datos bajo condiciones de estrés. En términos de tiempos de respuesta, el mínimo fue de 135 milisegundos y el máximo de 6998 milisegundos, con una media de 4211.6 milisegundos, lo que sugiere que aunque la *API* maneja bien las solicitudes bajo carga, hay margen de mejora para optimizar tiempos de respuesta más largos y mejorar la experiencia del usuario. Los detalles de los tiempos de respuesta y la longitud de las sesiones de los usuarios virtuales proporcionan una visión clara de cómo se comporta el sistema en diferentes escenarios de uso, facilitando así la identificación de áreas específicas para futuras optimizaciones.

```
All VUs finished. Total time: 1 minute, 8 seconds

------------------------------
Summary report @ 01:06:04(-0600)
------------------------------

errors.ETIMEDOUT: ..... 3
http.codes.200: ..... 4197
http.downloaded_bytes: ..... 186991
http.request_rate: ..... 59/sec
http.requests: ..... 4200
http.response_time:
  min: ..... 135
  max: ..... 6998
  mean: ..... 4211.6
  median: ..... 4583.6
  p95: ..... 5168
  p99: ..... 6976.1
http.response_time.2xx:
  min: ..... 135
  max: ..... 6998
  mean: ..... 4211.6
  median: ..... 4583.6
  p95: ..... 5168
  p99: ..... 6976.1
http.responses: ..... 4197
vusers.completed: ..... 4197
vusers.created: ..... 4200
vusers.created_by_name.Test de cantidad de ahorro de CO2 general: ..... 1091
vusers.created_by_name.Test de cantidad de ahorro de agua general: ..... 1042
vusers.created_by_name.Test de cantidad de desechos reciclables generales: ..... 1033
vusers.created_by_name.Test de top 5 desechos generales: ..... 1034
vusers.failed: ..... 3
vusers.session_length:
  min: ..... 205.6
  max: ..... 7312.5
  mean: ..... 4298
  median: ..... 4676.2
  p95: ..... 5378.9
  p99: ..... 6976.1
Log file: test/performance/artillery-report.json
```

Figura 6.135. Prueba de rendimiento, 70 solicitudes por segundo

En una prueba de rendimiento subsiguiente, se observó un comportamiento ligeramente diferente del sistema bajo condiciones de carga modificadas. Se generaron 4200 usuarios virtuales, y 4194 de ellos completaron sus sesiones, con 6 errores de tiempo de espera, un ligero incremento en comparación con la prueba anterior. A pesar de la disminución en la tasa de solicitud a 35 solicitudes por segundo, la *API* manejó el mismo número total de solicitudes, 4200, indicando una capacidad constante para manejar cargas elevadas.

El total de bytes descargados aumentó a 188502, y aunque se redujo la cantidad de respuestas exitosas a 4194, los tiempos de respuesta mostraron una mejora significativa. El tiempo de respuesta mínimo fue de tan solo 58 milisegundos, el máximo fue de 224 milisegundos, y la media fue considerablemente más baja, con 78.4 milisegundos. Estos resultados reflejan una mejora notable en la eficiencia de procesamiento y la rapidez de respuesta del sistema, con tiempos de respuesta mucho más cortos que en la prueba anterior. La longitud de las sesiones de los usuarios también varió, proporcionando datos valiosos para futuras optimizaciones en la gestión del tráfico y la capacidad de respuesta del servidor.

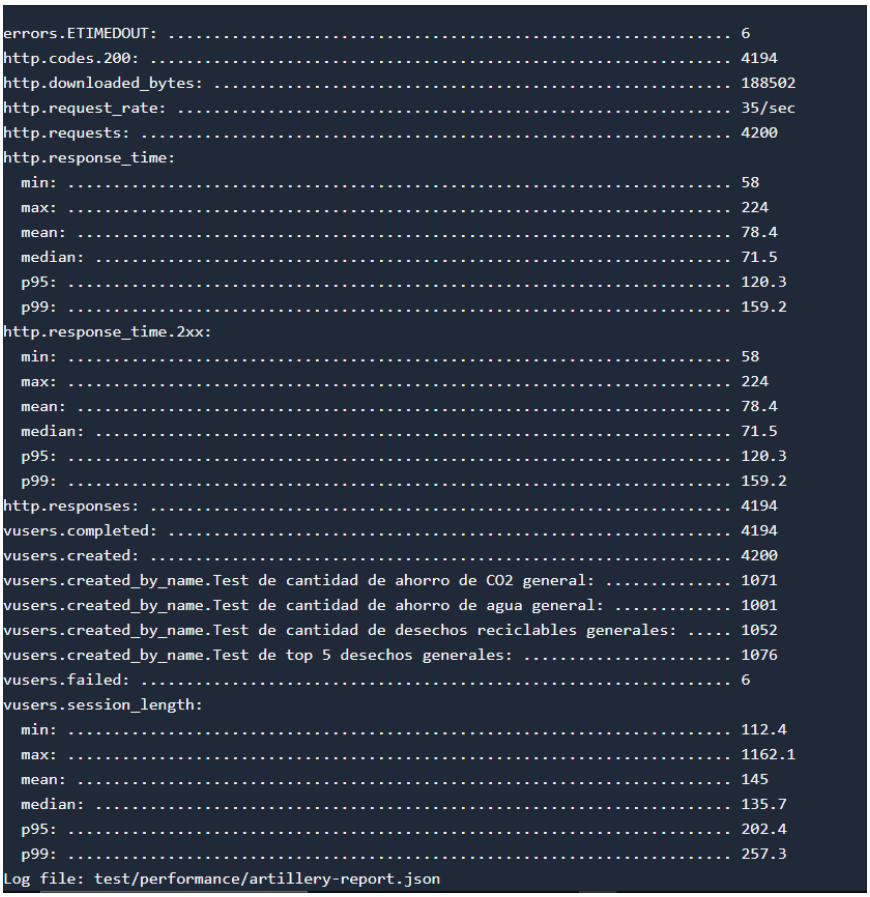


Figura 6.136. Prueba de rendimiento, 35 solicitudes por segundo

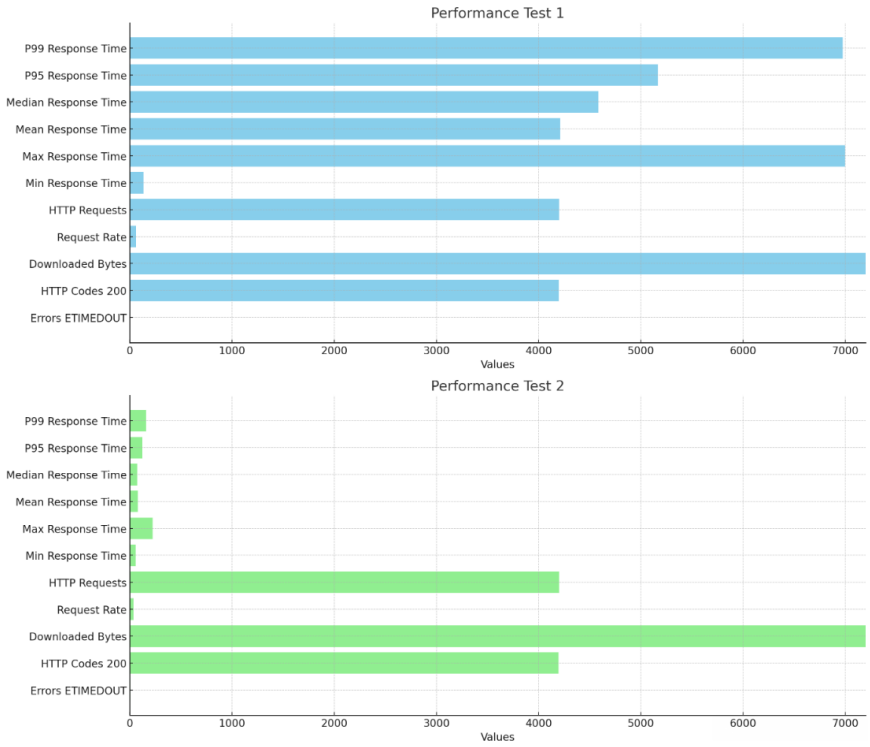


Figura 6.137. Resultados sobre las pruebas de rendimiento

Las pruebas comparativas son presentadas mediante gráficas de barras horizontales, las cuales ofrecen una visualización clara y directa de los resultados en diversas métricas clave:

- **Errores de tiempo de espera (*ETIMEDOUT*):** se observa un ligero aumento en los errores de la primera a la segunda prueba, indicativo de posibles puntos de tensión bajo ciertas cargas.
- **Códigos HTTP 200:** la cantidad de respuestas exitosas muestra una alta consistencia, sugiriendo que el *API* mantiene una fiabilidad sólida incluso bajo diferentes configuraciones de prueba.
- **Bytes descargados y tasa de solicitud:** aunque la tasa de solicitud disminuyó en la segunda prueba, la cantidad de datos procesados fue ligeramente superior, lo que puede reflejar un manejo eficiente de solicitudes más complejas o con mayor contenido de datos.
- **Tiempos de respuesta:** la mejora sustancial en los tiempos de respuesta mínimos, máximos, y promedios entre las pruebas es destacable. Los tiempos más cortos en la segunda prueba indican una capacidad mejorada para procesar solicitudes de manera rápida y eficiente.

La gráfica comparativa revela que el *API*, aunque expuesto a variaciones en la tasa de carga, demuestra una capacidad robusta para mantener la eficacia y eficiencia operativa.

6.5.5. Plan de mantenimiento y costos

- **Afluencia baja**: aproximadamente **$75 al mes**.
- **Afluencia media**: aproximadamente **$300 al mes**.
- **Afluencia alta**: aproximadamente **$1,200 al mes**.

AWS Elastic Beanstalk con *Amazon EC2* para la *API*

- **Afluencia baja**: instancia `t3.micro` con un costo aproximado de **$8.50 al mes**.
- **Afluencia media**: dos instancias `t3.small` y balanceador de carga, costo aproximado de **$35 al mes**.
- **Afluencia alta**: cuatro instancias `t3.medium`, balanceador de carga y autoescalado, costo aproximado de **$140 al mes**.

Amazon RDS para la base de datos *PostgreSQL*

- **Afluencia baja**: instancia `db.t3.micro`, costo aproximado de **$15 al mes**.
- **Afluencia media**: instancia `db.t3.small` con 50 GB SSD, costo aproximado de **$40 al mes**.
- **Afluencia alta**: instancia `db.t3.medium` con Multi-AZ y 100 GB SSD, costo aproximado de **$200 al mes**.

AWS SageMaker para el modelo de IA

AWS SageMaker se emplea para alojar el modelo de IA, permitiendo inferencias en tiempo real.

- **Afluencia baja**: instancia `ml.t2.medium` con 4 horas diarias de uso, costo aproximado de **$50 al mes**.
- **Afluencia media**: instancia `ml.m5.large` con 8 horas diarias de uso, costo aproximado de **$200 al mes**.
- **Afluencia alta**: instancia `ml.m5.xlarge` con 24 horas diarias de uso, costo aproximado de **$600 al mes**.

Servicios de autenticación

AWS Amplify facilita la autenticación de usuarios mediante proveedores externos.

- **Afluencia baja**: hasta 50,000 solicitudes de autenticación cubiertas por el nivel gratuito, **$0 al mes**.
- **Afluencia media**: hasta 500,000 solicitudes de autenticación, costo aproximado de **$25 al mes**.
- **Afluencia alta**: hasta 5,000,000 solicitudes de autenticación, costo aproximado de **$250 al mes**.

Amazon S3 para almacenamiento de datos

Amazon S3 proporciona almacenamiento escalable, con costos según la cantidad de datos y las solicitudes.

- **Afluencia baja**: hasta 50 GB, costo aproximado de **$1.15 al mes**.
- **Afluencia media**: hasta 200 GB, costo aproximado de **$4.60 al mes**.
- **Afluencia alta**: hasta 1 TB, costo aproximado de **$23 al mes**.

6.5.6. Consideraciones de mantenimiento

Para mantener un entorno optimizado y seguro, se han incluido los siguientes aspectos en el plan de mantenimiento:

- **Monitoreo y alertas**: utilizar *Amazon CloudWatch* para monitorear el rendimiento y establecer alertas que permitan identificar y resolver problemas de manera proactiva.
- **Actualizaciones de seguridad**: mantener actualizados los servicios y aplicar parches de seguridad de acuerdo con las recomendaciones de *AWS* para evitar vulnerabilidades.
- **Optimización de costos**: revisar periódicamente el uso de recursos y ajustar los servicios en función de la demanda, aplicando configuraciones de autoescalado para evitar costos adicionales innecesarios.

CAPÍTULO 7

Análisis de resultados

7.1. Análisis de resultados de experiencia de usuario

7.1.1. Usuarios identificados

Los resultados de las primeras encuestas realizadas proporcionaron información de suma importancia para poder definir no sólo los perfiles de los usuarios potenciales de la aplicación, sino también información que fue vital para definir las funcionalidades más importantes de la aplicación y la orientación que su desarrollo debía tomar en general. Los resultados demográficos permitieron confirmar los perfiles de usuario identificados luego de las asesorías por parte de la Municipalidad de Guatemala. Los resultados sobre conocimientos generales y hábitos de gestión de residuos y reducción de la huella de carbono permitieron la comprensión del nivel educativo respecto a los temas tratados por el proyecto por parte de la población. Finalmente, los resultados sobre las expectativas de la aplicación por parte del público objetivo proporcionaron información importante para lograr comprender las necesidades de los usuarios potenciales y priorizar las funcionalidades que resultarían más importantes para el proyecto.

Resultados demográficos sobre los usuarios identificados

En cuestión a la demografía que se logró obtener, es posible observar cómo la mayoría de personas encuestadas se encontraba en un rango de edad entre 18 y 25 años, lo cual inmediatamente nos da una idea de que la mayoría de personas que utilizarían la aplicación se encuentran en este rango de edad, lo cual cuadra con la investigación realizada sobre la distribución demográfica en cuestión a edades en el departamento de Guatemala. Estos resultados también se pueden deber al hecho de que la encuesta también fue compartida en redes sociales por parte de los miembros del equipo de trabajo, donde claramente la interacción con estas publicaciones fue por parte de personas que tienen una edad muy similar a las nuestras.

También se recibieron múltiples respuestas diferentes a la pregunta sobre el municipio en el que las personas encuestadas residían. La mayoría de personas se encontraban en los municipios más poblados del departamento de Guatemala, como lo son Guatemala, Villa Nueva y Mixco. Sin embargo, también se recibieron respuestas de departamentos que no son Guatemala, como Chiquimula y Huehuetenango. Pese a que estas personas se encuentran fuera del alcance del proyecto, las

respuestas fueron tomadas en cuenta con el objetivo de tener una mayor variabilidad a tener en mente.

Para finalizar, en lo que respecta al nivel educativo de las personas encuestadas, es posible observar cómo la mayoría de usuarios potenciales de la aplicación han recibido como mínimo un título universitario y algunos incluso un posgrado. Solamente el 35 % aproximadamente no contaban con una carrera universitaria. Este resultado es muy importante para cuadrar otro resultado que se mostrará más adelante, pero para resumir la gran cantidad de posibles usuarios potenciales con estudios universitarios brindará información valiosa sobre cómo la mayoría de potenciales usuarios también ha recibido educación ambiental en lugares como el colegio y la universidad.

Resultados sobre conocimientos y comportamientos sobre la gestión de residuos

Analizando los resultados obtenidos en la sección de conocimientos y comportamientos sobre la gestión de residuos revela un panorama mixto en cuanto al nivel de entendimiento y la confianza que la población tiene sobre este tema. En general, se observa que los encuestados poseen un entendimiento básico de los principios relacionados con la gestión de residuos y la reducción de la huella de carbono. Sin embargo, lo que destaca es que, a pesar de este conocimiento básico, prácticamente nadie se considera un experto en el área. Esta falta de auto percepción de competencia puede ser indicativa de una carencia de formación especializada o de experiencias previas que validen su comprensión.

La mayoría de las preguntas de las encuestas fueron respondidas de manera neutral, lo que sugiere que los encuestados reconocen que tienen conocimientos sobre la gestión de residuos, pero no a un nivel profesional que les otorgue la confianza necesaria para manejar el tema de manera efectiva. Esta percepción puede ser crucial, ya que implica que muchos de los participantes podrían sentirse inseguros o confundidos al momento de aplicar estos conocimientos en situaciones prácticas, especialmente cuando se trata de utilizar la aplicación que se desarrollará para la gestión de residuos. Es probable que esta falta de confianza genere dudas durante el proceso, lo que resalta la importancia de diseñar una aplicación intuitiva y educativa que guíe a los usuarios a través de las diversas funciones y recursos disponibles.

Los resultados también indican que los principales inconvenientes que enfrenta la población objetivo al gestionar residuos son, casi en partes iguales, la falta de recursos, la escasa información y la limitación de tiempo. La falta de recursos puede referirse tanto a la ausencia de infraestructura adecuada para la recolección y reciclaje de residuos, como a la falta de incentivos económicos que motiven a la población a participar activamente en la gestión de residuos. Por otro lado, la escasa información podría significar que la población no tiene acceso a materiales educativos suficientes que expliquen cómo gestionar sus residuos de manera efectiva, lo que puede llevar a prácticas inadecuadas. La limitación de tiempo también juega un papel crucial; muchas personas llevan un estilo de vida ajetreado que dificulta la dedicación necesaria para realizar prácticas sostenibles de gestión de residuos.

A pesar de estos desafíos, los resultados revelan un aspecto muy positivo, y es que la población es consciente de la importancia de la gestión de residuos. Este reconocimiento es fundamental, ya que sugiere que, aunque puedan existir barreras, hay una disposición general para aprender y mejorar sus prácticas. Los encuestados expresaron su interés en implementar cambios en su comportamiento diario que contribuyan a una mejor gestión de residuos. Esta actitud proactiva no solo refleja una creciente conciencia ambiental, sino también una apertura al cambio, lo cual es crucial para el éxito de cualquier programa o aplicación destinada a facilitar la gestión de residuos.

Resultados sobre necesidades y expectativas de los usuarios sobre la aplicación

Los resultados obtenidos en la sección de necesidades y expectativas de los usuarios sobre la aplicación proporcionan una visión clara de las funcionalidades más solicitadas por los potenciales usuarios. En primer lugar, se destaca la demanda de guías para clasificar residuos, que se consideran fundamentales para facilitar el correcto manejo de los desechos. Los usuarios también expresaron un gran interés en recibir sugerencias para reducir su huella de carbono, lo que indica una conciencia creciente sobre la importancia de adoptar prácticas sostenibles. Además, la posibilidad de ingresar datos sobre sus residuos y acceder a gráficos que visualicen el consumo de basura fue otra funcionalidad altamente valorada, ya que esto permitiría a los usuarios monitorear su comportamiento y realizar ajustes informados en sus hábitos diarios.

La mayor motivación que tienen los potenciales usuarios para utilizar la aplicación radica en su facilidad de uso. La interfaz debe ser intuitiva y accesible, lo que permitiría que personas con diferentes niveles de experiencia tecnológica puedan interactuar con la aplicación sin frustraciones. En segundo lugar, los usuarios subrayaron la importancia de recibir información clara y fácil de entender. Esto sugiere que las explicaciones y directrices proporcionadas dentro de la aplicación deben ser sencillas, evitando el uso de jerga técnica que pueda resultar confusa para quienes no están familiarizados con el tema de la gestión de residuos.

Otro aspecto fundamental que se identificó en los resultados es la utilidad de contar con una guía interactiva para aprender a clasificar residuos. Esta herramienta no solo ayudaría a los usuarios a gestionar sus desechos de manera más efectiva, sino que también fomentaría un aprendizaje continuo sobre la sostenibilidad y el reciclaje. Asimismo, la capacidad de visualizar estadísticas sobre la huella de carbono y los progresos individuales sería altamente valorada, ya que permitiría a los usuarios ver el impacto directo de sus acciones en el medio ambiente.

El diseño atractivo de la aplicación se considera un factor crucial para su éxito. Un entorno visualmente agradable puede aumentar la motivación y el interés de los usuarios, haciendo que la experiencia de uso sea más placentera. Un diseño bien pensado no solo contribuye a la usabilidad, sino que también puede transmitir un mensaje positivo sobre el compromiso con la sostenibilidad y la innovación.

7.1.2. Prototipado iterativo

Los resultados que se obtuvieron en la sección pasada sirvieron principalmente para la definición de funcionalidades y objetivos a cumplir por parte de la solución a desarrollar. Sin embargo, la sección actual que respecta el desarrollo del prototipo de alta fidelidad de la solución tuvo como meta generar resultados que permitieran mejorar la solución planteada y cumplir con el objetivo de que la misma fuera atractiva y fácil de intuir y utilizar por parte de los usuarios potenciales. Cada una de las tres iteraciones realizadas sobre el prototipo generó una gran cantidad de información diferente que permitió mejorar el prototipo conforme su avance y la retroalimentación constante por parte de los usuarios potenciales.

Primera iteración del prototipo

En la primera iteración del prototipo, la recepción de las pantallas fue en general positiva, con la mayoría superando el 80 % de aprobación. Sin embargo, la pantalla de ingreso de datos de basura destacó negativamente, alcanzando solo un 55 % de aceptación, lo que indica la necesidad de mejoras en esta área.

Los usuarios dieron una retroalimentación muy positiva sobre la pantalla inicial, aunque sugirie-

ron elevar el botón de ingreso para mejorar la accesibilidad. La pantalla de inicio de sesión recibió comentarios positivos, pero se mencionó la confusión generada por el color de los botones en relación con el fondo.

La pantalla de menú principal fue la mejor valorada, destacándose por su presentación clara y comprensible. La configuración de perfil también fue bien recibida, especialmente por la distribución de los botones. No obstante, en la pantalla de analíticas de consumo, algunos usuarios mostraron confusión sobre el botón "Aceptar", sugiriendo que se necesita clarificación.

La pantalla de ingreso de datos fue identificada como la de menor aceptación, con comentarios que subrayaron la dificultad en el flujo de inserción de información debido a la distancia entre los campos de entrada y los íconos. Finalmente, las pantallas de chat recibieron opiniones mayormente positivas, aunque se sugirió explicar mejor la funcionalidad de ciertos botones, y la similitud con otras aplicaciones fue bien valorada, indicando que los usuarios se sienten cómodos con ese tipo de diseño. Estos hallazgos destacan áreas críticas que requieren atención en futuras iteraciones para optimizar la experiencia del usuario.

Segunda iteración del prototipo

En la segunda iteración del prototipo, la aceptación general de las pantallas mejoró notablemente, superando todas el 80 % de aprobación. Esto refleja la efectividad de las modificaciones realizadas en la iteración anterior. La pantalla inicial recibió más comentarios positivos, sin sugerencias de mejora, lo que indica que es intuitiva. La pantalla de inicio de sesión también fue bien recibida, aunque se sugirió más espacio entre elementos para mayor comodidad.

La pantalla de menú principal continuó siendo la mejor valorada, lo que indica su diseño eficiente. La configuración de perfil recibió buenos comentarios, pero se recomendó cambiar el color de los botones críticos para aumentar su visibilidad. La pantalla de analíticas de consumo mostró mejoras, con menos confusión sobre la funcionalidad de los botones tras los cambios.

La pantalla de ingreso de datos también recibió comentarios más positivos, con la sugerencia de agrandar el botón para subir fotografías, mejorando así la usabilidad. Las pantallas de chat, tanto para interactuar con el modelo como para conversar, mantuvieron comentarios muy positivos y no se propusieron mejoras significativas, lo que sugiere que su diseño cumple con las expectativas de los usuarios y ofrece una experiencia familiar y confortable.

Tercera iteración del prototipo

En la tercera iteración del prototipo, la recepción de las pantallas mejoró aún más, reflejando el éxito de las modificaciones anteriores. Los comentarios fueron mayormente positivos, indicando que la aplicación se alinea con las expectativas de los usuarios.

La pantalla inicial y la de inicio de sesión recibieron elogios por su diseño atractivo y eficiente, sin sugerencias de mejora. La pantalla de menú principal tuvo un excelente recibimiento y se considera efectiva. La configuración de perfil también fue bien valorada, especialmente por el cambio en el color de botones críticos, lo que mejoró su visibilidad. La pantalla de analíticas de consumo mantuvo una buena recepción, aunque se sugirió añadir más gráficos para visualizar mejor el consumo de basura. La mejora en la pantalla de ingreso de datos fue notable, con comentarios positivos y sin recomendaciones de cambio, destacando su simplicidad. Para finalizar, las pantallas de chat y conversación con el modelo recibieron comentarios muy positivos, resaltando su diseño intuitivo y la facilidad de interacción, lo que es clave para fomentar el uso de la aplicación.

7.1.3. Evaluación final a la aplicación

Los resultados obtenidos de las pruebas de usabilidad y de *eye tracking* validan la efectividad de la aplicación desarrollada en el contexto de la problemática de la producción desmedida de basura y su gestión inadecuada. En primer lugar, el promedio de tiempo requerido por los usuarios para completar diversas acciones refleja una experiencia de usuario mayormente fluida, lo que sugiere que la interfaz ha sido diseñada con un enfoque intuitivo. La rapidez en acciones como registrar el consumo de basura y observar las analíticas indica que estas funcionalidades están bien optimizadas y cumplen con el objetivo de facilitar el proceso para los usuarios.

La interacción visual, evidenciada a través de los mapas de calor generados por el *eye tracking*, muestra que los usuarios dirigen su atención a los elementos clave de la interfaz, como los botones de acción. Este comportamiento no solo resalta la efectividad de la organización visual de la aplicación, sino que también sugiere que el diseño es atractivo y capta la atención de los usuarios en los lugares apropiados. Además, el hecho de que las miradas se dirijan a gráficos y otros elementos complementarios demuestra que los usuarios encuentran valor en la información presentada, lo cual es fundamental para fomentar prácticas sostenibles.

El cumplimiento de los objetivos del proyecto se manifiesta en la creación de una interfaz de usuario que no solo es atractiva visualmente, sino que también es funcional. La aplicación aborda de manera efectiva la problemática de la gestión de residuos al permitir que los usuarios registren su consumo de basura y accedan a analíticas, lo cual contribuye a una mejor comprensión de sus hábitos. Asimismo, el enfoque en el diseño centrado en el usuario se ha traducido en una aplicación que se adapta a las necesidades y preferencias de la audiencia objetivo, logrando una experiencia que se alinea con las mejores prácticas de usabilidad.

Para terminar, las pruebas piloto realizadas con un grupo seleccionado de usuarios, junto con las encuestas de satisfacción y la retroalimentación cualitativa, indican que la aplicación no solo cumple con los estándares de efectividad y usabilidad esperados, sino que también tiene el potencial de impactar positivamente en la reducción de residuos. La combinación de una interfaz intuitiva, tiempos de respuesta eficientes y un diseño atractivo refuerza la conclusión de que la aplicación es una herramienta viable para abordar la problemática ambiental actual.

7.2. Análisis de resultados de *backend*, gestión de base de datos y análisis de datos

Una vez obtenidos los resultados, es fundamental analizarlos para evaluar si cumplen con el enfoque y los objetivos planteados inicialmente. En los siguientes apartados se examinará cada una de las implementaciones y su contribución a la eficiencia, robustez y calidad del módulo final.

7.2.1. Elección de las tecnologías

Las tecnologías seleccionadas para el proyecto han demostrado ser adecuadas, proporcionando una integración fluida y efectiva entre sí. La elección de estas tecnologías fue acertada, ya que cada una facilita la operabilidad y permite la incorporación de herramientas externas que fortalecen y consolidan una base sólida para el módulo. Esta combinación de tecnologías no solo garantiza la estabilidad y eficiencia del sistema, sino que también facilita futuras expansiones y mejoras.

7.2.2. Diseño y estructura de la base de datos

El diseño final de la base de datos permite gestionar de manera eficiente la información de los usuarios. A lo largo de las iteraciones, se implementaron mejoras y ajustes específicos que dieron lugar a un modelo optimizado para almacenar los datos necesarios para el análisis. Este proceso permitió evaluar y ajustar cada aspecto de la estructura de la base de datos, asegurando que solo se almacene la información relevante y optimizando el rendimiento general del sistema.

La incorporación de una tabla de auditoría o bitácora fortalece la integridad de la información al permitir el monitoreo de todos los cambios realizados en los datos. Esto crea una base sólida para futuras expansiones de la aplicación, facilitando el manejo de roles y permisos para usuarios y administradores, y asegurando un control detallado de las operaciones dentro del sistema.

7.2.3. *API*

La versión final del *API* incorpora funcionalidades clave para el manejo seguro y eficiente de la información. Entre estas, destaca la integración de autenticación en las solicitudes mediante **JWT** (*JSON Web Token*), lo cual asegura que solo usuarios autorizados puedan acceder y manipular los datos, protegiendo la integridad de la información desde su uso en una aplicación hasta su almacenamiento en la base de datos.

Las pruebas de funcionalidad han verificado que estas restricciones de autenticación operan correctamente, permitiendo a los usuarios acceder mediante la aplicación sin exponer la información a personas no autorizadas.

Las funciones implementadas en cada modelo permiten el ingreso adecuado de los datos, asegurando el almacenamiento preciso de la información relacionada con los usuarios, ubicaciones y registros de desechos. Además, el registro de actividades en la bitácora de auditoría proporciona un historial detallado de las operaciones realizadas, incluyendo las acciones de agregación, actualización y eliminación, junto con la información de los usuarios y las fechas correspondientes. Esto garantiza un seguimiento completo y confiable de las actividades en el sistema.

Las funciones de análisis de datos permiten calcular con precisión la información relacionada con los usuarios en cuanto a sus registros de gestión de desechos. Estas funciones proporcionan una visión detallada del comportamiento y patrones de uso, facilitando el análisis de datos clave para evaluar el impacto de la gestión de residuos y mejorar la toma de decisiones.

Mediante una fase de pruebas exhaustiva del *API* final, se verificó que todas las funcionalidades fueron implementadas correctamente y funcionan conforme a lo esperado. Para alcanzar esta conclusión, se realizaron tres tipos de pruebas que confirman el correcto funcionamiento del sistema.

Las pruebas realizadas incluyen pruebas unitarias, las cuales validaron cada uno de los controladores, verificando que proporcionen la información de manera precisa y eficiente. Además, se llevaron a cabo pruebas de integración, que demostraron la coordinación adecuada entre las rutas y los controladores al procesar las solicitudes del *API*, garantizando una integración fluida y coherente.

Finalmente, se realizaron pruebas de rendimiento y carga, donde, a través de múltiples iteraciones, se evaluó la capacidad de respuesta del sistema ante diferentes niveles de demanda. Estas pruebas abarcaron tres escenarios específicos, simulando el 20 %, 50 % y 100 % de la carga esperada, y en todos los casos el sistema mostró un rendimiento robusto. De acuerdo con las recomendaciones de [419], se verificó que el tiempo de respuesta de las solicitudes se mantuviera por debajo de los 10 segundos, asegurando así una experiencia fluida para el usuario y minimizando la probabilidad de pérdida de atención.

7.2.4. Análisis de datos

El apartado de análisis de datos se implementó mediante una estructuración adecuada de la información de los usuarios. Cada una de las funciones diseñadas para el análisis fue construida de manera precisa, con el apoyo directo de la Unidad de Reciclaje de la Municipalidad de Guatemala.

Gracias a la orientación de la Unidad de Reciclaje, se lograron identificar las métricas necesarias para proporcionar información relevante al usuario en el contexto de la aplicación. Se definieron funciones que permiten a los usuarios comprender cómo sus prácticas de gestión de residuos pueden contribuir al cuidado del medio ambiente.

Entre las principales funcionalidades, destacan las métricas de reducción de emisiones de CO_2 y de ahorro de agua, que permiten a los usuarios evaluar el impacto positivo de sus acciones en la reducción de su huella de carbono. Esto es consistente con lo mencionado en [420], donde se indica que una de las maneras más efectivas de reducir la huella de carbono personal es a través de un reciclaje y una gestión de desechos adecuados. Además, la reducción de la huella de carbono se relaciona estrechamente con la disminución de emisiones de CO_2 y el ahorro de agua, promoviendo una gestión ambientalmente responsable.

Por otro lado, además de proporcionar resultados personalizados para cada usuario, se establecieron las bases para realizar análisis de datos a nivel global, considerando a todos los usuarios de la aplicación. Esto permite obtener una visión más amplia y comprender los patrones de comportamiento que pueden incentivar y guiar a los usuarios hacia un impacto ambiental positivo.

Como resultado de este proceso, se crearon tableros de análisis de datos en **Power BI**, lo cual permite a los administradores y responsables de áreas ambientales acceder a información detallada sobre el comportamiento de la población que utiliza la aplicación. Estos tableros ofrecen una visión completa y precisa, facilitando la toma de decisiones informadas y la implementación de estrategias para promover prácticas sostenibles.

7.2.5. Implementación de *CI/CD*

En el apartado de ***CI/CD***, se establecieron bases sólidas para que el proyecto esté preparado para futuras iteraciones y expansiones a largo plazo. Esto se logró mediante la aplicación de buenas prácticas de Ingeniería de Software, que permitieron diseñar un *pipeline* que abarca las fases de desarrollo, pruebas, lanzamiento y monitoreo.

Cada una de las etapas en este *pipeline* automatizado desempeña un rol clave en asegurar la robustez y adecuada gestión del *API*. Las herramientas implementadas en cada fase permiten detectar y corregir errores de manera temprana, garantizando la calidad y estabilidad del sistema en cada actualización y facilitando un crecimiento continuo y ordenado del proyecto.

La fase de pruebas permite automatizar la verificación del correcto funcionamiento de los controladores y rutas del *API*, lo cual facilita la detección temprana de errores y alerta al equipo en caso de problemas ocasionados por iteraciones de código incorrectas. Esto asegura una identificación de problemas generales y un manejo adecuado de errores antes de realizar el despliegue en producción.

Además, se implementaron pruebas de carga y rendimiento automatizadas, con el objetivo de evaluar el desempeño del *API* en futuras iteraciones donde se anticipe un mayor volumen de datos o la incorporación de nuevas funcionalidades. Estas pruebas permiten monitorear el rendimiento del *API* de manera continua, garantizando que mantenga su eficiencia y robustez a medida que evoluciona.

Gracias a la automatización y a la verificación adecuada, cada actualización permite que el *API* se despliegue de manera automática. Esto garantiza que cada cambio aprobado esté inmediatamente

disponible para los usuarios, facilitando un flujo continuo de mejoras y optimizaciones en tiempo real.

Por último, una vez que el *API* se encuentra en producción, es monitoreada constantemente mediante la herramienta **Sentry**. Esta herramienta permite gestionar cualquier error inesperado de manera eficaz, proporcionando información detallada sobre cada incidente para facilitar una resolución rápida y eficiente.

Además, *Sentry* permite analizar el comportamiento de las solicitudes realizadas al *API*, ofreciendo un monitoreo continuo de métricas clave. Esto facilita una evaluación constante del rendimiento del *API*, asegurando que se mantenga en óptimas condiciones y responda adecuadamente a las demandas de los usuarios.

7.3. Análisis de resultados de inteligencia artificial

7.3.1. Análisis de resultados del modelo *Xception*

Análisis de las métricas de precisión y pérdida

El rendimiento del modelo *Xception* fue evaluado mediante las métricas de precisión y pérdida tanto en el conjunto de entrenamiento como en el conjunto de validación. Estas métricas se analizaron para comprender la capacidad de aprendizaje y generalización del modelo.

Historia de precisión y pérdida Las gráficas de precisión y pérdida ilustran cómo el modelo *Xception* aprende y ajusta sus parámetros a lo largo de las épocas. Se observa que la precisión en el conjunto de entrenamiento aumenta rápidamente, alcanzando casi el 100 %, mientras que la precisión en el conjunto de validación se estabiliza en un valor elevado de aproximadamente 98 %. Este comportamiento refleja la capacidad del modelo para capturar patrones complejos en los datos de entrenamiento y mantener una alta precisión en datos no vistos.

La pérdida en el conjunto de entrenamiento disminuye de manera pronunciada en las primeras épocas y luego se estabiliza en un valor bajo. De forma similar, la pérdida en el conjunto de validación sigue una trayectoria descendente y se estabiliza sin mostrar un aumento significativo, lo que indica que el modelo no está sobreajustado. Este comportamiento es producto de una adecuada configuración de hiperparámetros y del uso de técnicas de regularización como *Dropout* y la regularización L2, las cuales ayudan a mitigar el sobreajuste y mejorar la capacidad de generalización.

Eficacia del aprendizaje El modelo *Xception*, conocido por su arquitectura basada en convoluciones separables en profundidad, es particularmente eficiente para la extracción de características. Esta capacidad ha sido esencial para manejar la variabilidad visual inherente a los residuos, permitiendo que el modelo distinga entre clases visualmente similares. La alta precisión de validación sugiere que el diseño del modelo ha sido adecuado para esta tarea de clasificación de residuos, aprovechando la eficiencia de las convoluciones para capturar detalles relevantes en las imágenes.

—

Análisis de la matriz de confusión

La matriz de confusión proporciona un análisis detallado de las predicciones correctas e incorrectas del modelo para cada clase de residuos. Las predicciones se concentran en gran medida en la

diagonal principal, lo que es indicativo de un rendimiento sobresaliente en la clasificación.

Rendimiento por clase

- **Clases con alto rendimiento**: las clases *Battery* y *Paper* destacan por tener casi todas las predicciones correctas, con tasas de precisión y *recall* cercanas al 100 %. Las características visuales distintivas de estas clases, como los contornos definidos y las texturas homogéneas, han sido capturadas eficazmente por el modelo, facilitando una clasificación precisa.
- **Clases con desafíos moderados**: se observan algunas confusiones en las clases *Plastic* y *Glass*. Estas confusiones pueden explicarse por similitudes visuales, como superficies reflectantes y traslúcidas que pueden compartir ciertos residuos plásticos y de vidrio. A pesar de esto, el modelo aún logra mantener un alto nivel de precisión, lo que es una evidencia de su capacidad para extraer y diferenciar características sutiles.
- **Observaciones generales**: la efectividad del modelo *Xception* se ve reforzada por su capacidad para manejar estas similitudes visuales mediante una representación de características robusta. Las capas convolucionales de *Xception*, optimizadas para la separación de canales espaciales y de profundidad, han permitido al modelo identificar patrones relevantes que son críticos para una clasificación precisa.

La matriz de confusión, en conjunto con las métricas asociadas, valida la eficacia del modelo en términos de su capacidad de diferenciación y generalización. El análisis de los casos específicos en los que ocurren errores de clasificación puede proporcionar información útil para futuras optimizaciones del modelo.

Análisis del reporte de clasificación

El reporte de clasificación presenta las métricas de *precision*, *recall* y *F1-score* para cada clase de residuos, lo que permite una evaluación granular del rendimiento del modelo.

Métricas de rendimiento

- **Precisión elevada**: la precisión promedio del 98 % indica que el modelo realiza predicciones correctas con una frecuencia muy alta. Esto es crucial en aplicaciones como la clasificación de residuos, donde las predicciones erróneas pueden afectar la eficacia del proceso de reciclaje. La alta precisión es especialmente destacable en las clases *Battery* y *Paper*, donde los patrones visuales específicos han sido correctamente aprendidos y reconocidos por el modelo.
- **Recall consistente**: un *recall* promedio del 98 % muestra que el modelo tiene una alta capacidad para identificar ejemplos positivos de cada clase, minimizando los falsos negativos. Esto es esencial en sistemas de reciclaje automatizado, ya que asegura que la mayoría de los residuos se clasifiquen correctamente sin perder instancias importantes.
- **F1-Score balanceado**: el *F1-score* promedio también se mantiene en un 98 %, reflejando un equilibrio ideal entre precisión y *recall*. Esto indica que el modelo no solo es preciso, sino que también mantiene una capacidad de detección uniforme en todas las clases. Incluso las clases que presentan ciertos desafíos, como *Plastic* y *Glass*, muestran un *F1-score* competitivo, lo que reafirma la solidez del modelo.

Estas métricas resaltan el éxito de la arquitectura *Xception* para esta tarea específica. La combinación de precisión, *recall* y *F1-score* elevados en todas las clases demuestra que el modelo es

robusto y confiable, con un desempeño uniforme que lo hace adecuado para aplicaciones prácticas en la gestión de residuos.

Análisis de *Explainable AI*: *GradCAM* y *LIME*

La interpretabilidad de los modelos de redes neuronales profundas es un tema crucial en la aplicación de sistemas de IA, especialmente en contextos críticos como la clasificación de residuos. En este proyecto, se utilizaron herramientas de *Explainable AI*, como *GradCAM* y *LIME*, para ofrecer una interpretación clara de las decisiones del modelo Xception. Estas técnicas no solo ayudan a validar el rendimiento del modelo, sino que también proporcionan una comprensión más profunda de cómo el modelo procesa las imágenes y qué características considera importantes.

Interpretaciones con *GradCAM* *GradCAM (Gradient-weighted Class Activation Mapping)* es una técnica que permite visualizar las áreas de una imagen que activan más intensamente las últimas capas convolucionales del modelo. En el caso del modelo *Xception*, se generaron mapas de calor para evaluar las regiones de las imágenes en las que el modelo se enfoca al realizar sus predicciones.

Observaciones y comportamiento del modelo Los mapas de *GradCAM* revelaron cómo el modelo *Xception* distribuye su atención al clasificar diferentes tipos de residuos. Se observó lo siguiente:

- **Clase metal**: los mapas de *GradCAM* destacan claramente los bordes brillantes y las superficies reflejantes de los objetos metálicos. Este enfoque es consistente con la naturaleza visual de los metales, que generalmente tienen superficies lisas y reflectantes. La arquitectura *Xception*, al descomponer las imágenes en componentes más simples mediante convoluciones separables en profundidad, es capaz de capturar estos patrones complejos de manera eficiente. Esto demuestra que el modelo ha aprendido a identificar correctamente las señales visuales clave para esta clase.
- **Clase vidrio**: para residuos de vidrio, *GradCAM* muestra una atención hacia las áreas de transparencia y los bordes del material. La capacidad del modelo para identificar las texturas translúcidas y las formas definidas del vidrio refuerza la idea de que *Xception* puede manejar variaciones visuales sutiles gracias a su estructura de extracción de características optimizada. La atención en las áreas correctas también sugiere que el modelo ha aprendido a ignorar elementos irrelevantes del fondo, lo que es crucial para aplicaciones en entornos reales.
- **Clase plástico**: aquí, los mapas de calor muestran una ligera dispersión en la atención del modelo, centrándose en las áreas de color y los contornos suaves. La variabilidad visual en los residuos plásticos, que pueden presentarse en diferentes colores, formas y texturas, introduce un desafío adicional. Esta dispersión en la atención sugiere que el modelo podría beneficiarse de una mayor diversidad en los datos de entrenamiento para mejorar su capacidad de generalización. Sin embargo, el enfoque en características distintivas como los contornos y las superficies de color uniforme es un indicio positivo de que el modelo está utilizando las pistas visuales adecuadas.
- **Clase orgánico**: los residuos orgánicos presentan un patrón visual caracterizado por texturas rugosas y formas irregulares. Los mapas de *GradCAM* muestran que el modelo *Xception* se centra en estas características, lo que es coherente con la naturaleza heterogénea de los desechos orgánicos. La habilidad del modelo para identificar estas texturas complejas demuestra la eficacia de las convoluciones separables en profundidad de *Xception*, que maximizan la extracción de detalles relevantes mientras minimizan el costo computacional.

- **Clase papel**: *GradCAM* revela que el modelo presta atención a las superficies lisas y los bordes definidos del papel, ignorando elementos de fondo irrelevantes. Esto es importante, ya que las características visuales del papel suelen ser menos complejas que las de otras clases. El modelo parece haber aprendido de manera eficiente a diferenciar el papel de otros materiales mediante la identificación de patrones visuales simples pero distintivos.
- **Clase *Battery***: los residuos electrónicos, como las baterías, se identifican a través de detalles específicos, como los terminales y las formas geométricas distintivas. Los mapas de *GradCAM* destacan estas áreas, indicando que el modelo está utilizando información visual precisa y relevante. La capacidad de *Xception* para detectar detalles finos y estructuras complejas se refleja en la atención que el modelo presta a las características esenciales de las baterías.

Importancia de la arquitectura *Xception* La arquitectura *Xception* se basa en convoluciones separables en profundidad, que descomponen el proceso de convolución en dos pasos: una convolución de profundidad y una convolución puntual. Esta estructura permite una extracción de características más eficiente y detallada. En este contexto, *GradCAM* ha demostrado que *Xception* es capaz de capturar patrones visuales relevantes para cada clase de residuos, lo que refuerza la idea de que este modelo es adecuado para manejar datos con una gran diversidad visual.

Interpretaciones con *LIME* *LIME* (*Local Interpretable Model-agnostic Explanations*) ofrece una forma de interpretar las predicciones del modelo al resaltar los segmentos de imagen que más contribuyen a una clasificación específica. LIME proporciona una visión más granular y detallada de cómo el modelo *Xception* toma decisiones basadas en las características visuales de las imágenes.

Desglose de las Explicaciones de *LIME* *LIME* fue utilizado para descomponer las imágenes en segmentos y evaluar la importancia de cada uno en la predicción del modelo. Aquí se detallan las observaciones clave:

- **Clase metal**: *LIME* destaca segmentos que contienen superficies brillantes y áreas reflectantes. Esto es coherente con el análisis de *GradCAM*, donde se observa que el modelo prioriza las características visuales asociadas con el metal. La redundancia en la atención a estas características refuerza la confianza en la precisión del modelo.
- **Clase vidrio**: los segmentos resaltados por *LIME* incluyen áreas transparentes y bordes definidos, lo que indica que el modelo *Xception* utiliza información visual significativa para clasificar correctamente los objetos de vidrio. La interpretación granular de *LIME* valida que el modelo no está sobreajustado a patrones irrelevantes y que realmente comprende las propiedades visuales de los residuos de vidrio.
- **Clase plástico**: las explicaciones de *LIME* muestran una mayor variabilidad en los segmentos importantes, lo que subraya la diversidad intrínseca de los residuos plásticos. Aunque el modelo identifica correctamente las áreas clave, la dispersión en la atención sugiere que el modelo aún puede beneficiarse de un conjunto de datos más representativo para mejorar su rendimiento en esta clase.
- **Clase orgánico**: *LIME* resalta segmentos que corresponden a texturas rugosas y colores naturales, lo que es indicativo de un modelo que ha aprendido a identificar la heterogeneidad de los residuos orgánicos. Esto es relevante porque demuestra que el modelo no se enfoca en artefactos visuales sino en las características esenciales de los desechos.
- **Clase papel**: los segmentos resaltados en las imágenes de papel muestran que el modelo utiliza bordes y áreas planas para realizar sus predicciones. Esto es consistente con las observaciones de *GradCAM* y refleja una comprensión adecuada de las características distintivas del papel.

- **Clase *Battery***: *LIME* destaca áreas como los terminales de las baterías y otros detalles específicos. Esto confirma que el modelo está utilizando información relevante para clasificar residuos electrónicos, lo cual es crucial para la precisión y fiabilidad en entornos reales.

Justificación y relevancia de las interpretaciones Las interpretaciones proporcionadas por *LIME* y *GradCAM* son fundamentales para entender y justificar las decisiones del modelo. En términos de *Explainable AI*, estos resultados demuestran que el modelo *Xception* no solo es preciso, sino también interpretativamente sólido, lo que es esencial para aplicaciones críticas como la gestión de residuos. Además, en el contexto de *MLOps*, la transparencia y trazabilidad proporcionadas por estas herramientas facilitan el monitoreo y la auditoría del modelo, garantizando que las decisiones sean confiables y explicables.

Impacto y comparación con otros estudios En comparación con estudios previos, donde los modelos de clasificación de residuos a menudo carecen de interpretabilidad, el uso de *Xception* junto con técnicas de *Explainable AI* en este proyecto representa un avance significativo. Mientras que otros enfoques pueden lograr precisión, este trabajo destaca por su capacidad para explicar las decisiones del modelo de manera clara y comprensible, lo que es crucial para la adopción en aplicaciones reales. La arquitectura *Xception*, combinada con estas herramientas de explicabilidad, proporciona un modelo que no solo es eficaz sino también transparente y auditable.

Conclusiones del análisis El análisis detallado con *GradCAM* y *LIME* confirma que el modelo *Xception* utiliza características visuales significativas para clasificar los residuos. Esto valida que las decisiones del modelo son razonables y justificadas, lo que es crucial para su implementación en sistemas de gestión de residuos. La estructura optimizada de *Xception* permite un aprendizaje efectivo y una clasificación precisa, lo que refuerza la elección de este modelo para esta tarea. Además, la capacidad de auditar y comprender el comportamiento del modelo a través de *Explainable AI* fortalece la confiabilidad del sistema.

Análisis de *Saliency Maps* con *SmoothGrad*

Para proporcionar una interpretación más precisa y menos ruidosa de las decisiones del modelo *Xception*, se utilizaron *saliency maps* generados con la técnica ***SmoothGrad***. *SmoothGrad* es una mejora de los mapas de saliencia tradicionales que añade ruido a la entrada y promedia los gradientes obtenidos. Esto ayuda a suavizar los mapas de saliencia y resaltar de manera más clara las áreas importantes en las que el modelo se enfoca. A continuación, se realiza un análisis detallado de cómo *SmoothGrad* contribuye a la comprensión del rendimiento del modelo en el set de validación/prueba.

Principios de *SmoothGrad* y su aplicación *SmoothGrad* funciona agregando ruido gaussiano a las imágenes de entrada y calculando los gradientes varias veces. Luego, los mapas de saliencia resultantes se promedian, lo que reduce el ruido y proporciona una visualización más coherente de las áreas de atención del modelo. Esto es particularmente útil en redes profundas como *Xception*, donde los mapas de saliencia originales pueden ser ruidosos y difíciles de interpretar.

Evidencia de aprendizaje significativo Los *saliency maps* generados con *SmoothGrad* mostraron patrones consistentes que validan el aprendizaje significativo del modelo *Xception*. A continuación, se describen las observaciones clave por clase:

- **Clase metal**: los mapas de saliencia de *SmoothGrad* destacaron las superficies brillantes y los bordes definidos de los objetos metálicos. Esta observación es consistente con las propiedades

visuales del metal, que a menudo se caracteriza por reflejar la luz de manera distintiva. La arquitectura *Xception*, con su enfoque en convoluciones separables en profundidad, parece ser particularmente efectiva para captar estas características visuales, lo que explica la alta precisión del modelo en esta clase. El enfoque en las áreas relevantes sugiere que el modelo no se ve distraído por características irrelevantes del fondo.

- **Clase vidrio**: para los residuos de vidrio, los *saliency maps* resaltaron las áreas donde la transparencia y los bordes de los objetos eran prominentes. Esta atención precisa indica que el modelo ha aprendido a distinguir los objetos de vidrio, que suelen ser más desafiantes debido a su naturaleza translúcida. La estructura optimizada de *Xception*, que permite una mejor extracción de características complejas, es claramente ventajosa en este contexto. El modelo ignora eficazmente los elementos de fondo, lo que demuestra un aprendizaje robusto y una capacidad de generalización adecuada.
- **Clase plástico**: los residuos plásticos presentaron un patrón más disperso en los mapas de saliencia, lo que refleja la diversidad visual de esta clase. Sin embargo, *SmoothGrad* ayudó a clarificar las áreas clave en las que el modelo se enfoca, como los contornos y las texturas homogéneas. Esta observación sugiere que, aunque el modelo tiene cierta dificultad con la variabilidad de los residuos plásticos, sigue utilizando características significativas para realizar sus predicciones. La dispersión moderada en la atención resalta la necesidad de más datos de entrenamiento que capturen mejor la diversidad de los residuos plásticos.
- **Clase orgánico**: los *saliency maps* de los residuos orgánicos mostraron un enfoque claro en las texturas rugosas y los colores naturales. Esto es importante, ya que los residuos orgánicos son visualmente heterogéneos y pueden ser difíciles de clasificar. El modelo parece haber aprendido a identificar estas características complejas, lo que es una señal positiva de su capacidad de generalización. La arquitectura de *Xception*, con su capacidad para extraer patrones espaciales detallados, es particularmente adecuada para esta tarea, lo que explica el buen rendimiento del modelo en esta clase.
- **Clase papel**: en el caso del papel, los *saliency maps* destacaron las áreas planas y los bordes claros, lo que es coherente con las características visuales simples de esta clase. El modelo *Xception* muestra un enfoque preciso y consistente, lo que sugiere que ha aprendido eficazmente las señales visuales necesarias para diferenciar el papel de otros materiales. Esto es una indicación de que la configuración del modelo es adecuada y que el aprendizaje ha sido efectivo.
- **Clase *battery***: los residuos electrónicos, como las baterías, son identificados a través de detalles finos y características específicas. Los mapas de saliencia de *SmoothGrad* resaltaron áreas como los terminales y las etiquetas de advertencia, lo que demuestra que el modelo utiliza información visual relevante y específica para esta clase. Esto es crucial, ya que los residuos electrónicos pueden tener un impacto ambiental significativo si no se gestionan adecuadamente. La capacidad del modelo para enfocarse en detalles clave refuerza la efectividad de la arquitectura *Xception* en la extracción de características complejas.

Explicabilidad y validación del modelo El uso de *SmoothGrad* proporciona una validación adicional de que el modelo *Xception* toma decisiones basadas en características visuales significativas y no en artefactos irrelevantes. Esta transparencia es fundamental en aplicaciones críticas como la gestión de residuos, donde la interpretabilidad del modelo es esencial para generar confianza en su uso. Además, el hecho de que el modelo se enfoque en las áreas correctas sugiere que las técnicas de regularización y ajuste de hiperparámetros han sido efectivas.

Comparación con otros métodos de interpretabilidad En comparación con otros enfoques de interpretabilidad, como los mapas de saliencia tradicionales, *SmoothGrad* ofrece una visualización mucho más clara y consistente. Esto es particularmente importante para modelos como *Xception*,

que tienen una gran cantidad de parámetros y capas complejas. Mientras que los mapas de saliencia tradicionales pueden ser ruidosos y difíciles de interpretar, *SmoothGrad* reduce este ruido y destaca las áreas de interés de manera más precisa.

Impacto en el ciclo de vida del modelo y *MLOps* Desde una perspectiva de *MLOps*, la capacidad de interpretar y validar las decisiones del modelo con herramientas como *SmoothGrad* es crucial para la gestión del ciclo de vida del modelo. La transparencia proporcionada por *SmoothGrad* facilita la monitorización y auditoría del modelo, asegurando que las decisiones sean confiables y explicables. Esto es especialmente importante cuando se considera la implementación del modelo en entornos de producción, donde la trazabilidad y la responsabilidad son esenciales.

Conclusiones del análisis con *SmoothGrad* El análisis con *SmoothGrad* confirma que el modelo *Xception* utiliza características visuales significativas para realizar sus predicciones. Esto valida que el modelo no solo es preciso, sino también explicablemente sólido, lo que es crucial para su implementación en sistemas de clasificación de residuos. La atención del modelo a las áreas correctas refuerza la confianza en su rendimiento y destaca la efectividad de la arquitectura *Xception* para manejar datos visualmente complejos. La capacidad de auditar y comprender las decisiones del modelo con *SmoothGrad* añade un nivel de confianza adicional, lo que es fundamental para aplicaciones prácticas en la gestión de residuos.

Análisis y discusión de resultados con imágenes reales

Las pruebas realizadas con el conjunto de imágenes reales proporcionan una evaluación crucial de la capacidad del modelo *Xception* para generalizar y realizar clasificaciones precisas en un entorno práctico, donde los residuos aparecen en contextos más complejos y variados en comparación con el set de prueba controlado. Se utilizaron 16 imágenes representativas de diferentes clases de residuos, y el modelo logró clasificar correctamente 15 de ellas, lo que corresponde a una precisión del 93.75 %. A continuación, se analizan los factores clave que explican este rendimiento y se discuten las implicaciones de los casos de éxito y los errores de clasificación observados.

Desempeño general y robustez del modelo El hecho de que el modelo haya clasificado correctamente el 93.75 % de las imágenes sugiere que la arquitectura *Xception* es adecuada para manejar datos visualmente complejos, lo cual se debe a su capacidad de extraer y procesar características relevantes mediante convoluciones separables en profundidad. Esta capacidad permite al modelo identificar patrones visuales significativos incluso en imágenes con fondos variados o desordenados.

Interpretación de imágenes con fondos complejos Uno de los desafíos más importantes al clasificar imágenes en entornos reales es la presencia de fondos complejos y visualmente distractores, como patrones de textura o elementos no relacionados con el objeto de interés. En las imágenes con fondos complejos, los *saliency maps* generados con *SmoothGrad* mostraron que el modelo *Xception* era capaz de enfocarse en las características críticas del objeto de la clase objetivo, como los contornos, las texturas únicas y las diferencias de color.

Por ejemplo, en una imagen de un residuo plástico colocado sobre un fondo de hierba y hojas secas, el modelo se centró principalmente en las características plásticas del objeto, ignorando eficazmente las distracciones del entorno. Este comportamiento demuestra que el modelo ha logrado un nivel notable de robustez y que la configuración de hiperparámetros y el uso de técnicas de regularización han sido efectivos para mejorar la capacidad de generalización.

Análisis de errores de clasificación De las 16 imágenes probadas, se observaron dos casos de errores de clasificación:

1. Una imagen de un residuo plástico fue clasificada erróneamente como papel. Esta imagen mostraba un pedazo de plástico sostenido en una mano humana. El *saliency map* de *SmoothGrad* para esta imagen reveló que el modelo enfocó parte de su atención en la textura y el color de la mano, lo cual pudo haber contribuido a la confusión con la clase papel. Además, las áreas del plástico que eran brillantes o reflectantes no se destacaron suficientemente como características relevantes, lo que probablemente influyó en el error de clasificación. Este error es consistente con las observaciones de la matriz de confusión, donde la clase *plastic* mostró algunas confusiones con otras clases.

2. El otro caso ya mencionado es el residuo plástico que el modelo clasificó como vidrio, debido a características visuales ambiguas como un brillo translúcido. Aquí, el modelo enfocó su atención en áreas del objeto que podían imitar las propiedades ópticas del vidrio.

Implicaciones de los errores Los errores de clasificación subrayan varias limitaciones inherentes a los modelos de visión por computadora cuando se enfrentan a imágenes reales con propiedades visuales ambivalentes. En el caso del plástico clasificado como papel, la interacción con la mano humana introdujo un sesgo visual, haciendo que el modelo interpretara erróneamente el material. Esto destaca un área clave de mejora, como la necesidad de un preprocesamiento adicional que pueda ayudar a desambiguar características visuales en contextos humanos.

Desde la perspectiva de la matriz de confusión, la clase *plastic* es una de las que presenta más confusiones, lo que ya se había observado en el análisis de las métricas. Este patrón confirma que, a pesar de los buenos resultados generales, la variedad de formas y texturas de los residuos plásticos sigue siendo un desafío para el modelo.

Validez de las explicaciones generadas con *SmoothGrad* Los *saliency maps* con *SmoothGrad* proporcionaron una herramienta crucial para validar las decisiones del modelo. En la mayoría de los casos, las áreas de atención resaltadas por *SmoothGrad* correspondieron con las características esperadas para cada clase de residuo, lo que sugiere que el modelo está tomando decisiones razonables y basadas en información relevante. Sin embargo, en los casos de error, se evidenció que el modelo puede ser influenciado por el contexto o características visuales no esenciales.

Impacto de fondos complejos Otra observación importante es el impacto de los fondos complejos en el rendimiento del modelo. Aunque el modelo mostró una robustez notable al ignorar muchos elementos de fondo, los errores de clasificación indican que el modelo puede ser susceptible a contextos visuales que introducen ambigüedad. Esto subraya la importancia de incluir más ejemplos en el *dataset* de entrenamiento que representen contextos diversos y escenarios donde las clases de residuos puedan ser confundidas debido a interacciones humanas o propiedades visuales similares.

Lecciones aprendidas y futuras direcciones Este análisis sugiere que, aunque el modelo *Xception* ha demostrado ser robusto y efectivo para la clasificación de residuos en entornos reales, hay margen de mejora en la gestión de casos ambiguos y fondos complicados. Las futuras iteraciones del modelo podrían beneficiarse de la integración de más ejemplos representativos de residuos ambiguos y de técnicas adicionales de *Explainable AI* que ofrezcan aún más claridad sobre las decisiones del modelo.

Desde una perspectiva de *MLOps*, estos hallazgos subrayan la importancia de seguir monitorizando y refinando el modelo en producción. La trazabilidad y las explicaciones proporcionadas por *SmoothGrad* serán herramientas valiosas para realizar ajustes informados y asegurar un rendimiento consistente del modelo a lo largo del tiempo.

Análisis de la implementación de *MLOps* con *MLFlow*

La implementación de *MLOps* utilizando ***MLFlow*** en este proyecto fue concebida como una **prueba de concepto (*PoC*)**, con el objetivo de evaluar la viabilidad y los beneficios potenciales de integrar prácticas de *MLOps* en el ciclo de vida del modelo de aprendizaje profundo *Xception* para la clasificación de residuos. Este análisis detalla cómo la configuración y el uso de *MLFlow* ayudaron a mejorar la gestión del proyecto, con implicaciones futuras que podrían facilitar la escalabilidad y el despliegue eficiente de modelos de *Machine Learning*.

Seguimiento de experimentos y viabilidad de implementación Uno de los aspectos más importantes de esta *PoC* fue demostrar la capacidad de *MLFlow* para realizar un seguimiento completo de los experimentos. Esto incluye el registro automático de hiperparámetros, métricas de rendimiento y artefactos de modelos. Durante el desarrollo del modelo *Xception*, la plataforma *MLFlow* permitió almacenar y organizar múltiples configuraciones de manera estructurada, lo que facilitó la comparación y selección de los mejores resultados.

Esta trazabilidad no solo es crucial para el desarrollo eficiente, sino que también tiene un impacto significativo en la reproducibilidad de los experimentos, un desafío común en proyectos de IA. La *PoC* demostró que, si este proyecto se implementa en un entorno de producción o se amplía, *MLFlow* proporcionaría una base sólida para garantizar que los modelos puedan ser auditados y replicados de manera confiable.

Gestión de modelos y versionamiento: prueba de concepto El versionamiento de modelos es una práctica esencial en *MLOps*, y en esta *PoC* se demostró la efectividad de *MLFlow* para gestionar versiones de manera eficaz. Cada modelo entrenado fue versionado junto con sus métricas de rendimiento, lo que simplifica el proceso de actualización o comparación de modelos. Aunque esta configuración no se llevó a un entorno de producción completo, la *PoC* mostró que *MLFlow* es capaz de soportar un flujo de trabajo robusto, donde las versiones del modelo pueden ser rastreadas y gestionadas con precisión.

Esta funcionalidad es especialmente relevante para proyectos como este, donde la mejora continua del modelo es crucial. La gestión de versiones asegura que los desarrolladores y científicos de datos puedan experimentar con nuevas configuraciones sin perder el contexto histórico ni comprometer la estabilidad del sistema.

Despliegue automatizado: un paso hacia la viabilidad en producción Aunque el despliegue del modelo *Xception* fue realizado como una prueba de concepto y no en un entorno de producción completo, el proceso automatizado configurado con *MLFlow* demostró el potencial de la herramienta para facilitar la transición de modelos desde el desarrollo hasta la producción. La *PoC* incluyó la configuración de flujos de trabajo que podrían automatizar el despliegue de modelos en servidores de inferencia, lo cual es un aspecto crítico para la implementación futura en sistemas de gestión de residuos en tiempo real.

La automatización y la infraestructura preparada para la integración con *MLFlow* aseguran que el proyecto esté bien posicionado para futuras escalas y actualizaciones. Esto es particularmente importante en aplicaciones de aprendizaje profundo donde los modelos deben ser regularmente re-entrenados y desplegados de manera eficiente.

Beneficios identificados en la *PoC* A través de esta prueba de concepto, se destacaron varios beneficios de la integración de *MLOps* con *MLFlow*:

- **Trazabilidad completa**: la capacidad de rastrear cada detalle del experimento proporciona confianza y control sobre el ciclo de vida del modelo. Esto es vital para asegurar la transparencia y la auditabilidad, especialmente en proyectos que pueden estar sujetos a escrutinio por la importancia ambiental y operativa.
- **Reproducibilidad mejorada**: con toda la información registrada de manera estructurada, cualquier experimento puede ser replicado de manera exacta. Esto es importante tanto para el desarrollo futuro como para la colaboración entre equipos.
- **Preparación para la escalabilidad**: aunque no se ha implementado en producción, el flujo de trabajo diseñado permite que el sistema esté preparado para escalar en un entorno operativo, haciendo que futuras implementaciones sean más sencillas y eficientes.

Desafíos y limitaciones de la *PoC* La prueba de concepto también reveló algunas limitaciones que deben ser consideradas en futuras implementaciones:

- **Complejidad de configuración**: la implementación inicial de *MLFlow* puede ser compleja y requiere una infraestructura adecuada. Para proyectos con recursos limitados, esto podría ser un obstáculo, aunque se justifica en proyectos que buscan tener un impacto a largo plazo.
- **Infraestructura de soporte**: un despliegue completo requeriría un *backend* más robusto y servicios adicionales para soportar las funciones automatizadas de *MLFlow*, lo que podría aumentar los costos y la complejidad operativa.

Implicaciones futuras y relevancia para *MLOps* A pesar de las limitaciones, esta *PoC* ha validado la importancia de *MLOps* en proyectos de aprendizaje profundo como este. La infraestructura y las prácticas demostradas con *MLFlow* aseguran que el proyecto esté listo para una implementación a mayor escala, con la capacidad de gestionar modelos de manera eficiente y mantener una mejora continua. Las técnicas y principios de *MLOps* aplicados aquí pueden servir como base para proyectos futuros, tanto en la gestión de residuos como en otras aplicaciones de IA donde la precisión, la reproducibilidad y la capacidad de actualización son esenciales.

En resumen, la implementación de *MLFlow* en este proyecto no solo ha mejorado la organización y el flujo de trabajo, sino que también ha proporcionado una visión clara de cómo un sistema completo de *MLOps* podría transformar la gestión y el impacto del modelo *Xception* en aplicaciones del mundo real.

Opiniones finales sobre el modelo *Xception*

El modelo *Xception* ha demostrado un rendimiento sobresaliente en la tarea de clasificación de residuos, alcanzando métricas de precisión y *recall* altamente satisfactorias. La arquitectura de convoluciones separables en profundidad utilizada por *Xception* ha mostrado su eficacia al extraer características complejas de imágenes de residuos, lo que ha sido esencial para lograr una capacidad de generalización robusta incluso en conjuntos de datos desafiantes.

El análisis de las métricas de precisión y pérdida ha revelado una estabilidad impresionante durante el entrenamiento y la validación, lo que indica que el modelo está bien ajustado y no sufre de un sobreajuste significativo. La precisión sostenida en el conjunto de validación es una señal de que el modelo es apto para ser desplegado en escenarios reales de clasificación de residuos, y el análisis de la matriz de confusión ha resaltado áreas específicas donde el modelo sobresale, así como las limitaciones que aún persisten.

Las técnicas de *Explainable AI*, como *GradCAM*, *LIME* y *SmoothGrad*, han proporcionado información crítica sobre cómo el modelo toma sus decisiones. La validación visual de estas explicaciones ha confirmado que *Xception* se enfoca en las características más relevantes de las imágenes, como texturas, bordes y colores, lo que respalda aún más la fiabilidad del modelo. Sin embargo, también se han identificado casos desafiantes, como la imagen de plástico clasificada erróneamente como papel, lo que subraya la necesidad de seguir investigando y refinando el modelo para manejar mejor las complejidades de ciertos escenarios.

El uso de *MLFlow* para una prueba de concepto ha subrayado la importancia de las prácticas de *MLOps* en el desarrollo de modelos de aprendizaje profundo. Aunque esta implementación fue limitada, ha demostrado cómo las herramientas de gestión y versionado de modelos pueden optimizar el flujo de trabajo y garantizar que las futuras iteraciones del sistema sean más manejables y escalables. Esto establece una base prometedora para una mayor integración de *MLOps* en futuros desarrollos.

Como conclusión, el modelo *Xception* ha cumplido con las expectativas en términos de precisión y capacidad de generalización, y el uso de técnicas de *Explainable AI* ha agregado un nivel valioso de transparencia y comprensión. Aunque hay áreas que podrían beneficiarse de un mayor refinamiento, los resultados actuales sugieren que este modelo es una herramienta poderosa para abordar problemas de clasificación de residuos, con un alto potencial para futuras aplicaciones y mejoras.

7.3.2. Análisis de resultados del *chatbot* inteligente

En esta sección, se presenta un análisis exhaustivo del rendimiento del *chatbot* inteligente, que fue implementado utilizando la *API* de *OpenAI*. El *chatbot* fue diseñado para asistir a los ciudadanos de la Ciudad de Guatemala en la clasificación de residuos, proporcionando respuestas precisas y relevantes basadas en la información disponible sobre el manejo y reciclaje de residuos.

Evaluación de la precisión y relevancia de las respuestas

La precisión y relevancia de las respuestas del *chatbot* fueron evaluadas mediante una encuesta realizada a expertos en gestión ambiental, así como por pruebas directas en escenarios simulados con diversas preguntas. Las evaluaciones indicaron que el *chatbot* es altamente efectivo en proporcionar información útil, aunque ciertas limitaciones fueron observadas en casos específicos.

Precisión de las respuestas La precisión fue uno de los aspectos más valorados por los expertos, con un 57.1 % otorgando una calificación de 5, lo que indica que las respuestas fueron completamente precisas y claras. El 42.9 % restante asignó una puntuación de 4, señalando que las respuestas eran generalmente correctas pero con margen de mejora en ciertos detalles o explicaciones.

El análisis del rendimiento del *chatbot* destaca la capacidad de la *API* de *OpenAI* para entender y procesar consultas complejas en lenguaje natural. Esto es posible gracias al extenso entrenamiento del modelo de lenguaje subyacente, que ha sido optimizado para la comprensión semántica y la generación de respuestas contextualmente apropiadas. Sin embargo, el modelo también presenta limitaciones inherentes a los modelos de lenguaje, como potenciales malentendidos de consultas ambiguas o la incapacidad de proporcionar respuestas completamente precisas cuando la información es escasa o no está directamente disponible.

Relevancia temática El *chatbot* fue evaluado en su capacidad para proporcionar respuestas alineadas con el tema de clasificación de residuos. Los expertos calificaron este criterio con un 57.1 % de 5 puntos y un 42.9 % de 4 puntos, lo que muestra un alto grado de satisfacción con la relevancia de

las respuestas. Esto valida la eficacia de la implementación para mantenerse enfocado en el dominio ambiental, un logro considerable dado el amplio conocimiento general que la *API* de *OpenAI* tiene en múltiples temas.

El éxito en este aspecto se debe a la implementación de estrategias de orientación temática que guían las respuestas del *chatbot* hacia un enfoque más específico, mitigando el riesgo de divagaciones hacia temas irrelevantes. La relevancia de las respuestas fue lograda a través de un entrenamiento y diseño cuidadoso de las instrucciones proporcionadas al modelo, que lo mantienen dentro del ámbito específico de la clasificación de residuos.

Capacidad para rechazar preguntas fuera de tema

El *chatbot* mostró una capacidad impresionante para manejar preguntas fuera de tema. En este criterio, el 85.7 % de los expertos calificó esta habilidad con un 5, destacando la cortesía y claridad con la que el *chatbot* rechaza consultas no relacionadas. Solo un 14.3 % de los evaluadores otorgó una puntuación de 4.

Esta funcionalidad es crítica para mantener la utilidad del *chatbot* dentro de su dominio específico, y fue implementada utilizando instrucciones claras que indican al modelo cuándo y cómo rechazar preguntas. La capacidad del modelo para rechazar adecuadamente preguntas no relacionadas es un reflejo de las habilidades de comprensión contextual de la *API* de *OpenAI*, las cuales permiten al *chatbot* identificar cuándo una consulta no se ajusta al ámbito de la gestión de residuos y responder de manera educada pero firme.

Implicaciones de *MLOps* y futuro despliegue Aunque el *chatbot* fue evaluado en un entorno controlado y sin un despliegue completo en un entorno de producción, los principios de *MLOps* también pueden aplicarse aquí para mejorar el mantenimiento y la mejora continua del sistema. En futuras iteraciones, la integración con herramientas de *MLOps* podría permitir un seguimiento en tiempo real del rendimiento del *chatbot*, versionado de actualizaciones de modelos, y pruebas A/B para optimizar las respuestas.

Calidad de las recomendaciones

El *chatbot* también fue evaluado por su capacidad para proporcionar recomendaciones útiles sobre el manejo de residuos. La calidad de estas recomendaciones fue bien recibida, con un 71.4 % de los expertos otorgando una puntuación de 4 y un 28.6 % calificando con un 5. Este resultado sugiere que, aunque las recomendaciones fueron útiles y claras, hay margen para mejorar en términos de detalle y especificidad.

La *API* de *OpenAI* ha sido entrenada con una vasta cantidad de datos textuales, lo que le permite proporcionar recomendaciones bien fundamentadas. Sin embargo, la calidad de estas respuestas también depende de la especificidad de la información proporcionada en las instrucciones iniciales al *chatbot* y del contexto en el que se utilizan. Esto resalta la importancia de la continua mejora y refinamiento de las instrucciones para garantizar que el *chatbot* siga proporcionando valor a los usuarios.

Consistencia en el estilo de las respuestas

La consistencia en el estilo de las respuestas también fue altamente valorada, con un 57.1 % de los expertos calificando con un 5 y un 42.9 % con un 4. Esto demuestra que el *chatbot* mantiene

un tono claro, educado y profesional en sus interacciones, lo cual es esencial para la confianza del usuario y la facilidad de uso.

El estilo de las respuestas fue diseñado para ser amigable y accesible, sin perder la formalidad necesaria para un asistente ambiental. Esta consistencia en el tono se debe a la configuración de la *API* de *OpenAI*, que permite definir el estilo de comunicación del modelo. Mantener un estilo consistente es importante para que los usuarios se sientan cómodos y seguros al interactuar con el *chatbot*, lo que podría ser un factor decisivo para la adopción del sistema.

Evaluación del tiempo de respuesta

El tiempo de respuesta promedio del *chatbot* fue de 1.7 segundos, como se muestra en la evaluación. Este tiempo es adecuado para la mayoría de las aplicaciones de asistencia en línea, proporcionando una experiencia de usuario fluida y eficiente. El tiempo de respuesta es un factor crucial, especialmente en entornos en los que los usuarios esperan respuestas rápidas y precisas.

El rendimiento en términos de tiempo es un testimonio de la optimización subyacente de la *API* de *OpenAI* y la infraestructura utilizada. La rapidez del *chatbot* no solo es importante para la experiencia del usuario, sino también para la escalabilidad del sistema si se implementa en un entorno de producción con un gran número de usuarios simultáneos.

Pruebas y ejemplos de interacciones

Se realizaron pruebas prácticas al *chatbot* para evaluar su capacidad de proporcionar respuestas precisas y relevantes a preguntas frecuentes sobre el manejo de residuos. A continuación, se muestran algunos ejemplos de las interacciones y análisis de las respuestas generadas:

Cuadro 7.1. Ejemplos de interacciones con el *chatbot* y su análisis

Pregunta	**Respuesta del *chatbot***	**Tiempo de respuesta**
¿Que puedo hacer con mis residuos plásticos?	Puedes clasificar tus residuos plásticos como reciclables y llevarlos a un centro de reciclaje adecuado en la Ciudad de Guatemala...	2.14 segundos
¿Cómo debo clasificar los residuos orgánicos?	Los residuos orgánicos deben ser clasificados en la categoría de *Orgánicos* (Verde)...	1.70 segundos
¿Dónde puedo reciclar el vidrio en la Ciudad de Guatemala?	En la Ciudad de Guatemala, puedes reciclar el vidrio en el centro de reciclaje Interfisa...	1.64 segundos
¿Cuáles son las reglas para desechar desechos electrónicos?	Las normas para desechar desechos electrónicos en la Ciudad de Guatemala implican no mezclarlos con la basura común...	2.21 segundos
¿Cuál es la capital de Francia?	Lo siento, solo puedo responder preguntas relacionadas con la clasificación y manejo de residuos...	0.89 segundos

El análisis de estas interacciones revela que el *chatbot* es eficaz en proporcionar respuestas claras y útiles dentro de su ámbito de conocimiento. La capacidad de rechazar preguntas fuera de tema también ha sido implementada con éxito, demostrando que el *chatbot* entiende el alcance de su funcionalidad.

Opiniones finales sobre el desempeño del *chatbot*

En general, el *chatbot* ha demostrado ser una herramienta valiosa para la clasificación de residuos, con respuestas precisas, relevantes y consistentes. La implementación de la *API* de *OpenAI* ha proporcionado un rendimiento sólido, aunque el sistema puede beneficiarse de actualizaciones y optimizaciones continuas basadas en el *feedback* de los usuarios y nuevas necesidades que puedan surgir.

El éxito del *chatbot* en esta etapa inicial valida la viabilidad de su uso a gran escala y sugiere un alto potencial para futuras aplicaciones en la gestión de residuos y otros ámbitos relacionados con la sostenibilidad y el medio ambiente.

7.3.3. Implicaciones de los agentes inteligentes en la Ciudad de Guatemala

La implementación de los agentes inteligentes desarrollados en este proyecto, el modelo *Xception* para la clasificación automática de residuos y el *chatbot* inteligente para la asistencia en la gestión de residuos, tiene un impacto significativo y potencialmente transformador en la Ciudad de Guatemala. Estas herramientas fueron diseñadas para abordar desafíos específicos en la educación y concienciación ambiental, y en esta sección se analizan las implicaciones prácticas, sociales y medioambientales de su aplicación en la vida cotidiana de los habitantes de la ciudad.

Facilitando la gestión eficiente de residuos

La gestión de residuos es uno de los mayores retos ambientales para la Ciudad de Guatemala, donde la falta de sistemas de reciclaje bien implementados y el desconocimiento de las prácticas sostenibles contribuyen a la contaminación y al deterioro de la calidad de vida. El modelo *Xception* ofrece una solución práctica para abordar este problema, ya que permite la clasificación precisa y rápida de residuos en seis categorías distintas: plástico, papel, vidrio, metal, orgánicos y desechos electrónicos. Este modelo puede integrarse en sistemas automatizados de separación de residuos en centros de reciclaje, mejorando la eficiencia y reduciendo la carga de trabajo manual.

El uso de técnicas de *Explainable AI* como *GradCAM* y *SmoothGrad* proporciona un nivel de transparencia esencial para ganar la confianza de los usuarios. Las autoridades municipales y los encargados de la gestión de residuos pueden ver y entender las razones detrás de las decisiones del modelo, lo que facilita su aceptación y uso en sistemas de reciclaje y clasificación de residuos.

Educación y concienciación ambiental a través del *chatbot* inteligente

El *chatbot* inteligente desarrollado para este proyecto representa una herramienta valiosa para educar y guiar a los ciudadanos sobre las mejores prácticas en la gestión de residuos. Al proporcionar respuestas precisas y relevantes en tiempo real, el *chatbot* puede integrarse en una aplicación móvil destinada a la ciudadanía, permitiendo un acceso sencillo a la información sobre reciclaje y sostenibilidad.

En la vida cotidiana, esto significa que cualquier residente de la Ciudad de Guatemala, independientemente de su nivel de conocimiento previo, puede obtener asesoramiento rápido sobre cómo clasificar sus residuos correctamente. La capacidad del *chatbot* para manejar consultas relacionadas con la ubicación de centros de reciclaje, las normas de desecho de residuos electrónicos y las mejores prácticas para el manejo de materiales reciclables promueve una cultura de responsabilidad ambiental.

Promoviendo hábitos sostenibles en la población

La combinación de estos dos agentes inteligentes tiene el potencial de generar un cambio profundo en los hábitos de los habitantes de la Ciudad de Guatemala. La disponibilidad de un sistema automatizado de clasificación, junto con un asesor virtual accesible, reduce las barreras para la adopción de prácticas sostenibles. Las personas pueden sentirse más empoderadas y seguras al saber que están clasificando sus residuos correctamente y contribuyendo al bienestar del medio ambiente.

Además, estas herramientas pueden ser implementadas en campañas de educación ambiental dirigidas por la Municipalidad, en escuelas, universidades y comunidades. Al tener una herramienta educativa y funcional que facilita la clasificación de residuos y fomenta la toma de decisiones informadas, se pueden alcanzar de manera más efectiva los objetivos de reducción de residuos y reciclaje establecidos por las autoridades locales.

Sostenibilidad y futuro de la gestión de residuos

El desarrollo de estos agentes inteligentes también tiene implicaciones positivas para el futuro de la sostenibilidad en la Ciudad de Guatemala. Con la creciente urbanización y la presión sobre los recursos naturales, es esencial contar con soluciones escalables y eficientes. La implementación de *MLOps* mediante *MLFlow*, aunque en este proyecto fue un prueba de concepto (PoC), sugiere que es posible gestionar el ciclo de vida de estos modelos de manera eficiente, facilitando futuras mejoras y adaptaciones.

A largo plazo, estas herramientas podrían integrarse en un sistema de gestión de residuos basado en IoT, donde sensores y dispositivos automatizados trabajen en conjunto con los modelos de IA para optimizar la recolección y el tratamiento de residuos. Esto no solo mejoraría la eficiencia, sino que también ayudaría a la ciudad a cumplir con los objetivos de sostenibilidad y a reducir su huella ecológica.

Desafíos y consideraciones finales

Si bien las implicaciones de estos agentes inteligentes son prometedoras, es importante considerar ciertos desafíos. La aceptación y adopción de estas herramientas por parte de la población dependerán en gran medida de la educación y sensibilización. Además, la infraestructura tecnológica de la ciudad debe estar preparada para soportar la integración de estos sistemas.

Otro factor crítico es la necesidad de políticas públicas que respalden la implementación de estas tecnologías, así como la inversión en infraestructura de reciclaje y programas de educación ambiental. La colaboración entre la Municipalidad, empresas privadas y organizaciones comunitarias será fundamental para asegurar el éxito de estas iniciativas.

Por lo tanto, los agentes inteligentes desarrollados en este proyecto se perfilan como una herramienta poderosa para transformar la gestión de residuos en la Ciudad de Guatemala, fomentando un cambio positivo en los hábitos de los ciudadanos y contribuyendo significativamente al bienestar del medio ambiente. La implementación de estas soluciones representa una vía prometedora hacia un futuro más sostenible, donde tecnología e innovación se conjugan para mejorar la calidad de vida urbana y proteger nuestro entorno.

7.4. Resultados de seguridad

A lo largo del proyecto, los resultados obtenidos evidencian un progreso significativo en la implementación de seguridad, desde la capacitación del equipo hasta la reducción de vulnerabilidades en el código y la cobertura de pruebas alcanzada. Este análisis reflexiona sobre los logros y desafíos del proyecto, abordando cómo cada aspecto se alineó con los objetivos propuestos y cómo los resultados obtenidos fortalecieron la postura de seguridad general de la aplicación.

7.4.1. Cultura de seguridad y capacitación en el equipo

Uno de los primeros logros clave fue el desarrollo de una cultura de seguridad entre los miembros del equipo, evidenciada en las pruebas de *phishing*. Inicialmente, los resultados mostraron un bajo desempeño, con un promedio de 4.5 respuestas correctas, lo cual indicó una falta de experiencia en la identificación de amenazas comunes, como correos de *phishing*. Tras implementar sesiones de capacitación específicas, el equipo logró una mejora notable, alcanzando un promedio final de 8.25 respuestas correctas en la segunda prueba y 9 en la prueba final de octubre, realizada sin previo aviso para evaluar el aprendizaje sostenido.

Además de las sesiones de *phishing*, se establecieron reuniones regulares para profundizar en temas clave de buenas prácticas y seguridad en el desarrollo, como los principios de *clean code* y el *OWASP Top 10*. Estas reuniones proporcionaron una base de conocimientos sobre la importancia de mantener un código claro y seguro, junto con la prevención de las vulnerabilidades más comunes en aplicaciones *web*. A lo largo de varias sesiones, el equipo fue instruido sobre.

Este enfoque continuo de capacitación no solo mejoró la habilidad del equipo para identificar amenazas, sino que también les brindó herramientas prácticas y conocimientos fundamentales para aplicar directamente en sus códigos, asegurando que las buenas prácticas de seguridad fueran consideradas durante todo el desarrollo. Este avance reflejó un compromiso con la seguridad desde la perspectiva de la gobernanza, alineándose con el marco *OWASP SAMM*, y contribuyendo a construir una base sólida de conocimientos de ciberseguridad dentro del equipo.

7.4.2. Reducción de vulnerabilidades mediante *SonarQube* y *SonarCloud*

La implementación de *SonarQube* y *SonarCloud* como herramientas de análisis de código fue determinante en el cumplimiento de los objetivos específicos de reducir vulnerabilidades y mejorar la calidad del código. A lo largo del proyecto, se observó una disminución gradual en el número de vulnerabilidades detectadas, lo cual evidencia que las prácticas de desarrollo seguro fueron integradas con éxito en el ciclo de vida del *software*. Este logro muestra cómo el equipo adoptó una mentalidad de mejora continua, que permitió no solo detectar y corregir errores, sino también prevenir su reaparición en etapas posteriores.

El uso de estas herramientas ha funcionado de manera destacada en la reducción de malas prácticas de codificación que provocan vulnerabilidades y otros problemas de seguridad. *SonarQube* y *SonarCloud* no solo identifican las áreas problemáticas del código, sino que también proporcionan ejemplos detallados que explican por qué una práctica es problemática y cómo puede ser corregida. Además, ofrecen referencias bibliográficas y enlaces a documentación adicional, permitiendo que los desarrolladores profundicen en el conocimiento de las mejores prácticas de seguridad y programación. Esto ha facilitado la capacitación personalizada de los miembros del equipo, ya que cada uno pudo identificar sus puntos débiles y mejorar sus habilidades de codificación.

Al recibir retroalimentación específica y ejemplos de solución, los desarrolladores se acostumbraron gradualmente a las buenas prácticas de codificación. Como resultado, el código se volvió

más limpio y seguro, reduciendo significativamente las vulnerabilidades a lo largo del desarrollo del proyecto. Este enfoque de mejora continua ha sido esencial para reforzar una cultura de desarrollo seguro y eficiente.

El cumplimiento del estándar de cobertura de código en *SonarCloud*, alcanzando un 82 %, complementó este logro al asegurar que el código estuviera correctamente probado y validado, incrementando así la confiabilidad de la aplicación. Este aspecto fue fundamental para garantizar un análisis exhaustivo y efectivo del código en consonancia con los principios del *OWASP Top 10*, aportando una base sólida en la seguridad y calidad de la aplicación.

7.4.3. Exclusiones de cobertura y configuración de calidad de código

En el contexto del análisis de *SonarCloud*, fue necesario realizar exclusiones específicas en la cobertura de código y en el análisis de algunos archivos. Los archivos excluidos de la cobertura contenían lógica que carece de aplicabilidad directa para pruebas de cobertura, lo que justifica su exclusión. Estos archivos no afectan la funcionalidad o seguridad del sistema, por lo que mantenerlos fuera del análisis de cobertura permitió una evaluación más precisa y relevante del código activo.

Además, los archivos excluidos del análisis corresponden a código de prueba cuyo propósito es exclusivamente de *testing* y, por lo tanto, no requieren de un análisis de seguridad ni de calidad de código, ya que su relevancia para los objetivos de seguridad es limitada. Estos archivos, al estar diseñados específicamente para probar componentes y funcionalidades del sistema, contienen configuraciones y dependencias que generan resultados que no aplican a la lógica principal del *software* en un entorno de producción.

Específicamente, el análisis de estos archivos en *SonarCloud* generaba una gran cantidad de vulnerabilidades e incidencias de seguridad catalogadas como *security hotspots*. Sin embargo, estas detecciones eran falsos positivos, ya que el código de *testing* contiene deliberadamente fragmentos y configuraciones que simulan condiciones de prueba, sin afectar la seguridad o funcionalidad del sistema final. Mantener estos archivos en el análisis hubiera inflado artificialmente los reportes de vulnerabilidad y reducido la precisión de los resultados.

Por tanto, excluir estos archivos de *testing* permitió obtener un reporte de calidad y seguridad más preciso y relevante, enfocado exclusivamente en el código que tiene impacto directo en la seguridad y funcionalidad de la aplicación.

Asimismo, la configuración personalizada de calidad de código fue una decisión estratégica clave. Se diseñó un *quality check* manual que incorporó dos requisitos adicionales a los criterios de *SonarCloud*: asegurar que la cobertura del código completo, no solo del nuevo, superara el 80 %, y que no existiera ningún *security hotspot* en el código. Estas condiciones adicionales fueron esenciales para reforzar el nivel de seguridad, asegurando que el código cumpliera consistentemente con altos estándares y que cualquier riesgo potencial fuera identificado y mitigado. Las demás condiciones de *SonarCloud* se mantuvieron, ya que cumplían con buenas prácticas de desarrollo y fortalecían el control de calidad del proyecto.

7.4.4. *Code smells* y resolución

Tal como se observó en la gráfica de los *code smells*, hubo un aumento significativo debido a un tipo de comentario repetitivo en varias partes del código. Sin embargo, estos problemas fueron resueltos de inmediato tras un análisis que determinó que dichos comentarios eran innecesarios. El equipo eliminó estos comentarios, logrando una reducción de la cantidad de *code smells*, aunque el cambio no se refleje perfectamente en la gráfica, esto demostrando mejoras en el diseño y en la aplicación de principios de *clean code*.

7.4.5. Código duplicado y principio *DRY*

Respecto al código duplicado, *SonarCloud* establece un estándar de calidad de no superar el 3 % en código duplicado. Al acercarse a este límite, se comunicó al equipo las áreas problemáticas, mayormente en los archivos de estilos del *frontend*. Se mostró al encargado cómo reestructurar y aplicar correctamente el principio *DRY* (*Don't Repeat Yourself*), reduciendo así la duplicación de código al 1 %. Aunque lo ideal es alcanzar un 0 %, se planea continuar aplicando estas mejoras conforme avanza el proyecto.

7.4.6. Complejidad ciclomática

La complejidad ciclomatica tuvo un breve aumento debido a la adición de nuevas líneas de código. Sin embargo, se logró reducirla, y el cálculo de la complejidad por línea mostró un valor de:

$$\text{Complejidad por Línea} = \frac{\text{Complejidad Total}}{\text{Número de Líneas de Código}} = \frac{298}{4019} = 0.074$$

Un ratio menor a 0.1 es considerado aceptable, indicando que la complejidad está bien distribuida y no representa un gran riesgo en términos de mantenibilidad. Esto demuestra que las prácticas de *clean code* se han aplicado adecuadamente, simplificando el código y facilitando su comprensión, prueba y mantenimiento.

Complejidad por línea de código (ratio)

- Menor a 0.1: se considera un rango aceptable y razonable para la mayoría de los proyectos grandes. Indica que la complejidad está bien distribuida y no representa un gran riesgo en términos de mantenibilidad.
- Entre 0.1 y 0.2: puede considerarse moderado, y es un signo de que se deben de empezar a revisar áreas específicas del código que podrían simplificarse. No es necesariamente grave, pero vale la pena monitorearlo.
- Mayor a 0.2: se considera alto y potencialmente problemático. Un ratio por encima de este valor sugiere que el proyecto tiene secciones complejas que pueden ser difíciles de mantener, entender y probar. A este nivel, es probable que se necesite una refactorización significativa para mejorar la calidad.

El valor obtenido de 0.074 en nuestro proyecto cae dentro del rango aceptable, lo que indica que el código está bien estructurado y no presenta riesgos significativos en términos de complejidad. Esto es un reflejo positivo del uso de prácticas de *clean code* y otras estrategias de simplificación, que han contribuido a mantener la complejidad del proyecto bajo control y a facilitar su mantenimiento.

7.4.7. Densidad de comentarios

La densidad de comentarios se mantuvo en un rango del 10 %, un valor adecuado para asegurar que el código esté bien documentado sin ser excesivamente comentado. Un porcentaje muy bajo puede indicar falta de documentación, mientras que un porcentaje muy alto puede sugerir un código complejo.

El aumento observado en la densidad de comentarios se debió a una mala comunicación de mi parte, haciendo pensar al encargado que era necesario comentar todo el código. Sin embargo, aclaré

que los comentarios solo debían agregarse en áreas ambiguas. Tras esta corrección, se logró mantener el porcentaje dentro del rango ideal.

Este ajuste en la densidad de comentarios, combinado con la aplicación de *clean code*, aseguró que el código fuera comprensible y fácil de mantener sin necesidad de explicaciones excesivas.

7.4.8. Integración de seguridad en componentes de inteligencia artificial y *chatbot*

Finalmente, la integración de prácticas de seguridad en los modelos de inteligencia artificial (IA) y en el código del *chatbot* refleja el compromiso del encargado con una protección integral de todos los componentes de la aplicación. Este trabajo fue desarrollado por mi compañero Pedro Pablo, quien desde el inicio aplicó rigurosamente las buenas prácticas de codificación. Los resultados de los escaneos de seguridad fueron una clara evidencia de esto: el modelo de IA desarrollado no presentó ningún problema de seguridad en los análisis iniciales, lo que demuestra la efectividad de las medidas preventivas que implementó.

En cuanto al escaneo de seguridad del *chatbot*, que hace uso del modelo de IA entrenado por Pedro Pablo, solo se detectó un problema menor, el cual fue corregido de inmediato. Este éxito refleja no solo la calidad del trabajo de Pedro Pablo, sino también el impacto positivo de las sesiones de capacitación en seguridad que se llevaron a cabo semanalmente. De mi parte, me aseguré de transmitirle de manera efectiva los conceptos y buenas prácticas discutidos durante las reuniones, lo cual él supo aplicar de manera óptima a lo largo de todo su desarrollo.

Este enfoque colaborativo y de aprendizaje continuo garantizó que los componentes críticos, como el *chatbot* y el modelo de IA, estuvieran alineados con los estándares de seguridad del proyecto. La aplicación de estas prácticas desde el principio no solo facilitó la identificación y corrección rápida de problemas, sino que también fortaleció la postura general de seguridad de la aplicación.

7.4.9. Análisis de vulnerabilidades en la aplicación móvil con *AppSweep* y *MobSF*

La incorporación de *AppSweep* y *MobSF* para el análisis de vulnerabilidades en la aplicación móvil fue un paso clave para cumplir con el objetivo de realizar evaluaciones exhaustivas de seguridad. Estas herramientas proporcionaron información crítica sobre posibles debilidades en el código y configuraciones específicas de la aplicación móvil, cubriendo aspectos de seguridad que complementan los análisis realizados en el *backend*.

Sin embargo, aunque se realizó un análisis y se identificaron vulnerabilidades, el proyecto aún no ha logrado aplicar correcciones para eliminar estos problemas de seguridad en la aplicación móvil. Este punto refleja una oportunidad de mejora, pues los hallazgos de *AppSweep* y *MobSF* constituyen un primer paso hacia la mitigación de riesgos, pero se necesita tiempo adicional para implementar las recomendaciones propuestas.

A pesar de no haber concluido con la remediación de todas las vulnerabilidades detectadas, esta etapa es un avance importante y forma parte del ciclo continuo de mejora del proyecto. Dado que este desarrollo se lleva a cabo en colaboración con la Municipalidad de Guatemala, se cuenta con la oportunidad de aplicar las correcciones necesarias en futuras fases del proyecto. Esto asegurará que la aplicación móvil cumpla con altos estándares de seguridad antes de ser lanzada al público, minimizando así los riesgos de filtración de información y mejorando la protección de los datos de los usuarios.

Esta situación también contribuye al aprendizaje continuo del equipo, ya que al aplicar las co-

rrecciones y buenas prácticas sugeridas por *AppSweep* y *MobSF*, será necesario capacitar al equipo en estas nuevas áreas de seguridad. Esto no solo permitirá fortalecer el conocimiento en el manejo de vulnerabilidades móviles, sino también consolidar una cultura de seguridad orientada a la proactividad y a la mejora continua.

7.4.10. Configuraciones de permisos en *AWS*

La gestión de permisos en *AWS* fue una parte crítica para asegurar que se cumpla con el principio de mínimo privilegio, minimizando así los riesgos de seguridad asociados a una configuración incorrecta o a la asignación excesiva de permisos. A continuación, se detalla el análisis y los cambios realizados en los permisos asignados a los miembros del equipo.

[node distance=2cm] role = [rectangle, rounded corners, minimum width=3cm, minimum height=1cm, text centered, draw=black, fill=cyan!30] responsibility = [rectangle, rounded corners, minimum width=4cm, minimum height=1cm, text centered, draw=black, fill=orange!20] arrow = [thick,->,>=stealth]
(oscar) [role] Oscar Lopez - Backend; (pedro) [role, below of=oscar] Pedro Pablo - Inteligencia Artificial; (santiago) [role, below of=pedro] Santiago Puga - Frontend; (gabriel) [role, below of=santiago] Gabriel Vicente - Infraestructura;
$(\text{oscar}_r es)[responsibility, rightof = oscar, xshift = 3cm]PublicarlaAPIenAWS; (pedro_r es)[responsibility, rightof = pedro, xshift = 5cm]LevantarelmodelodechatenAWS; (santiago_r es)[responsibility, rightof = santiago, xshift = 4cm]UX/UI(Sininteraccinconlanube); (gabriel_r es)[responsibility, rightof = gabriel, xshift = 6cm]Asegurarelfuncionamientodelanube, permisosyaccesoenAWS;$
[arrow] (oscar) – $(\text{oscar}_r es)$; $[arrow](pedro) - -(pedro_r es)$; $[arrow](santiago) - -(santiago_r es)$; $[arrow](gabriel) - -(gabriel_r es)$;

Figura 7.1. Diagrama de roles y responsabilidades del equipo

Oscar López: revisión de permisos de *DatabaseAdministrator*

En el caso de Oscar López, quien tiene el rol de ***DatabaseAdministrator***, se identificaron varios permisos innecesarios que se propusieron eliminar para mejorar la seguridad:

- `dynamodb:*` – este permiso permite gestionar *DynamoDB*, un recurso que no se utiliza en este proyecto, por lo que se decidió retirarlo.
- `s3:CreateBucket` – se propuso eliminar este permiso, ya que Oscar solo debe poder utilizar los *buckets S3* creados por el encargado de la infraestructura, Gabriel, en lugar de tener la capacidad de crear nuevos *buckets*.
- `iam` – este permiso fue considerado un riesgo, ya que permite a Oscar delegar permisos a funciones (roles). Se decidió quitarlo para evitar la posibilidad de un escalado de privilegios no autorizado. Limitar los roles específicos que Oscar pueda pasar es crucial para prevenir accesos indebidos y garantizar la seguridad del sistema.

Estos cambios aseguran que los permisos de Oscar se mantengan en un nivel mínimo necesario, evitando problemas potenciales derivados de un exceso de privilegios y asegurando un mejor cumplimiento de las prácticas de seguridad.

Pedro Pablo: revisión de permisos de *SystemAdministrator*

Para Pedro Pablo, quien tiene el rol de ***SystemAdministrator***, también se identificaron permisos que no eran necesarios:

- *trustadvisor:** – este permiso permite el acceso a *AWS Trusted Advisor*, que no es necesario para Pedro Pablo, ya que el monitoreo y asesoramiento sobre la configuración y el estado de la cuenta es responsabilidad de Gabriel.
- *config:** – permisos relacionados con *AWS Config*, que tampoco son relevantes para las tareas de Pedro Pablo.
- *iam* – al igual que con Oscar, se decidió eliminar este permiso para evitar riesgos de seguridad, ya que permite que Pedro Pablo delegue permisos a otros roles de entidades. Limitar esta capacidad es fundamental para proteger contra posibles escaladas de privilegios.

Estas modificaciones aseguran que Pedro Pablo solo tenga los permisos necesarios para su trabajo, manteniendo una configuración segura y eficiente.

Santiago Taracena y Yongbum Park: revisión de permisos de *ReadOnlyAccess*

Santiago Taracena y yo (Yongbum Park) tenemos asignado el rol de ***ReadOnlyAccess***, lo cual es adecuado para nuestras responsabilidades actuales, ya que ninguno de los dos necesita realizar modificaciones en la infraestructura de *AWS*. Esta configuración permite que Santiago trabaje a través de *GitHub Actions* para subir su parte del código sin intervenir directamente en *AWS*, y que yo supervise y verifique la infraestructura sin hacer cambios.

Sin embargo, se identificó que en mi caso, Yongbum Park, sería útil tener permisos adicionales para poder observar contraseñas y configuraciones relacionadas. Actualmente, ***ReadOnlyAccess*** no permite ver detalles como las contraseñas, lo cual es importante para revisar que se sigan las buenas prácticas de seguridad, como el uso de contraseñas complejas y únicas. Se planea ajustar mi configuración para incluir permisos de observación más completos, sin permitir ninguna modificación.

Estos ajustes en los permisos reflejan nuestro compromiso con una gestión de seguridad efectiva, alineada con el principio de mínimo privilegio, asegurando que cada usuario tenga solo los permisos estrictamente necesarios para desempeñar sus funciones sin comprometer la seguridad del sistema.

7.5. Resultados de infraestructura en la nube

El despliegue de la aplicación en *AWS* requirió la selección cuidadosa de recursos para cubrir necesidades de escalabilidad, seguridad y rendimiento. En esta sección, se analiza cómo cada uno de los recursos contribuyó al funcionamiento general de la aplicación, y se destacan las ventajas y desafíos encontrados al utilizar estos servicios en un entorno productivo.

7.5.1. *AWS Elastic Beanstalk* y *Amazon EC2*

AWS Elastic Beanstalk facilitó el despliegue de la *API*, proporcionando un entorno gestionado que permitió reducir el tiempo dedicado a la configuración de servidores y a la administración de infraestructura. La combinación de *Elastic Beanstalk* con instancias de *Amazon EC2* permitió gestionar automáticamente el escalado y el balanceo de carga, adaptándose eficientemente a variaciones en la demanda.

Ventajas:

- **Facilidad de uso**: *Elastic Beanstalk* simplificó la implementación de la *API* al automatizar la configuración del entorno, lo que permitió al equipo concentrarse en el desarrollo del código sin preocuparse por detalles de infraestructura.
- **Escalabilidad automática**: la capacidad de *Elastic Beanstalk* para escalar instancias *EC2* según la demanda resultó útil en los escenarios de alta afluencia, ya que ajustó la infraestructura sin intervención manual.

Desafíos:

- **Limitaciones en la personalización de la infraestructura**: aunque *Elastic Beanstalk* simplifica el despliegue, también limita la capacidad de personalización de algunos componentes específicos, lo cual podría ser una limitación para requisitos avanzados.
- **Reconfiguración de roles**: los cambios de permisos requeridos en *IAM* y las reconfiguraciones tras actualizaciones en *AWS* aumentaron la complejidad en la administración de permisos.

Amazon RDS para la base de datos *PostgreSQL*

Amazon RDS fue la opción seleccionada para gestionar la base de datos *PostgreSQL*, permitiendo delegar el mantenimiento y la administración de la base de datos a un servicio gestionado. *RDS* facilitó la implementación de la base de datos con configuraciones optimizadas para alta disponibilidad y redundancia.

Ventajas:

- **Administración de la base de datos simplificada**: al ser un servicio gestionado, *RDS* se encargó de tareas de mantenimiento, como la actualización de software y la creación de *backups* automáticos.
- **Alta disponibilidad y seguridad**: la configuración *Multi-AZ* brindó redundancia y disponibilidad, asegurando que los datos estuvieran protegidos y disponibles en caso de fallos.

Desafíos:

- **Costo de escalabilidad en alta afluencia**: aunque *RDS* facilita el escalado, el incremento de costos en escenarios de alta afluencia fue considerable, especialmente al usar instancias de mayor tamaño con configuraciones de alta disponibilidad.

AWS SageMaker para el despliegue del modelo de IA

AWS SageMaker se utilizó para alojar y servir el modelo de IA que procesa tanto imágenes como texto. *SageMaker* proporcionó un entorno escalable para gestionar inferencias en tiempo real, permitiendo optimizar el uso de recursos de procesamiento de IA.

Ventajas:

- **Escalabilidad para inferencias en tiempo real**: *SageMaker* permitió escalar el modelo de IA de acuerdo con la demanda de solicitudes de inferencia, asegurando una respuesta rápida en aplicaciones de alto rendimiento.

- **Integración con *S3* para datos de entrenamiento**: al integrarse con *Amazon S3, SageMaker* accedió a grandes volúmenes de datos de entrenamiento y almacenamiento de modelos de manera sencilla.

Desafíos:

- **Costos por uso continuo**: el costo por uso continuo en *SageMaker* aumentó considerablemente en alta afluencia, lo cual plantea la necesidad de optimizar los tiempos de actividad del *endpoint* en función de los patrones de uso.

AWS Amplify para la autenticación

AWS Amplify fue implementado para facilitar la autenticación de usuarios mediante servicios de terceros, como *Google* y *Facebook*. *Amplify* proporcionó un sistema de autenticación sencillo y seguro, reduciendo la complejidad de gestionar credenciales.

Ventajas:

- **Integración simplificada de autenticación**: *Amplify* permite conectar rápidamente la aplicación a proveedores de identidad externos, lo cual reduce significativamente el tiempo de desarrollo para gestionar autenticación de usuarios.
- **Escalabilidad con solicitudes de autenticación**: *Amplify* soporta una gran cantidad de solicitudes de autenticación sin requerir ajustes en la infraestructura, adaptándose bien a incrementos en el tráfico.

Desafíos:

- **Costos de autenticación en alta afluencia**: en niveles de afluencia alta, los costos por solicitud de autenticación se incrementan, lo que podría justificar una revisión de la estructura de autenticación para optimizar gastos.

Amazon S3 para almacenamiento de datos

Amazon S3 fue utilizado para el almacenamiento de archivos e imágenes. Este servicio proporcionó una solución de almacenamiento escalable y segura, con costos optimizados en función de las necesidades de acceso.

Ventajas:

- **Escalabilidad y costo eficiente**: *Amazon S3* facilita el almacenamiento de grandes volúmenes de datos y permite ajustar el tipo de almacenamiento en función de la frecuencia de acceso, lo cual resulta en una estructura de costos eficiente.
- **Integración con *SageMaker* y *Elastic Beanstalk***: la capacidad de integrar S3 con otros servicios de *AWS* simplificó el acceso a los datos y optimizó los tiempos de carga y procesamiento.

Desafíos:

- **Costos de salida de datos en alta afluencia**: aunque S3 es rentable para almacenar grandes volúmenes de datos, los costos de salida de datos aumentaron en función del tráfico, lo cual se debe considerar en afluencias elevadas.

Los recursos de *AWS* utilizados en el despliegue de la aplicación demostraron ser altamente efectivos en términos de escalabilidad y facilidad de administración. Cada servicio proporcionó una ventaja específica que facilitó el desarrollo y la puesta en marcha de la infraestructura, permitiendo manejar distintas cargas de trabajo. Sin embargo, el análisis muestra que a medida que aumenta la afluencia de usuarios, también se incrementan los costos, especialmente en servicios gestionados como *RDS, SageMaker* y *Amplify*.

CAPÍTULO 8

Conclusiones

8.1. Conclusiones de experiencia de usuario

- La aplicación desarrollada permitió abordar la problemática de la producción desmedida de basura, la falta de una gestión correcta de residuos y el aumento de la huella de carbono de la población. La aplicación constituye una solución que permite a sus usuarios gestionar sus residuos y aprender sobre temas como el reciclaje y la huella de carbono.
- Se investigó a la población del departamento de Guatemala, obteniendo información importante sobre su demografía, sus hábitos de gestión de residuos y sus expectativas sobre la solución planteada.
- La información obtenida sobre los usuarios potenciales de la aplicación permitió diseñar una experiencia de usuario basada en principios de *design thinking* que fue capaz de solventar las necesidades de los usuarios investigadas.
- Se implementó un diseño mediante un proceso iterativo clave para adaptar la aplicación a las necesidades de los usuarios a nivel visual. Este prototipo consistió en una interfaz gráfica intuitiva y atractiva según los resultados obtenidos de las pruebas realizadas durante el proceso iterativo.
- Las evaluaciones realizadas al prototipo desarrollado mostraron calificaciones positivas, validando tanto el diseño como el prototipo inicial y sentando una base sólida para el desarrollo en código de la aplicación.
- Con el prototipo finalizado después de las iteraciones realizadas, se desarrolló la interfaz gráfica de la aplicación utilizando tecnologías de desarrollo móvil adaptadas para el desarrollo de la solución implementada, priorizando las funcionalidades que tienen un impacto directo en la reducción de residuos y la promoción de prácticas sostenibles.
- Las evaluaciones realizadas durante todo el proceso de investigación, prototipado y desarrollo de la solución, especialmente las últimas pruebas de usabilidad y satisfacción, confirman que la interfaz gráfica desarrollada es intuitiva, haciendo de la aplicación una herramienta accesible y fácil de adoptar para diversos públicos.
- El proyecto desarrollado contribuye a la creación de tecnología que no solo facilita tareas, sino que también apoya el cumplimiento de los Objetivos de Desarrollo Sostenible, especialmente en cuanto a la gestión de residuos y la reducción de la huella de carbono.

- El proyecto presenta un enfoque innovador en el campo de la educación ambiental, proporcionando una experiencia interactiva que potencia el aprendizaje y la concientización sobre la gestión de residuos.
- El aprendizaje obtenido a nivel académico resultó sumamente valioso, desarrollando un proyecto que abarcó desde temas de experiencia de usuario y desarrollo hasta temas estadísticos, de investigación científica y análisis de resultados.

8.2. Conclusiones de *backend*, gestión de base de datos y análisis de datos

- **Eficiencia y robustez del sistema**: la elección de tecnologías adecuadas y la correcta estructuración de la base de datos han permitido construir un módulo de *Backend* eficiente y robusto, capaz de gestionar de manera óptima las solicitudes de los usuarios sobre sus registros y el análisis correspondiente.
- **Gestión y estructura de datos de residuos**: a lo largo del proyecto, se estableció una estructura sólida y eficiente para almacenar información detallada sobre los tipos de residuos y la frecuencia de desecho de los usuarios. Asimismo, se desarrolló un sistema de base de datos integral que permite clasificar y gestionar de manera óptima los datos relacionados con la gestión de residuos. Esta infraestructura no solo facilita el análisis tanto individual como global de la información, sino que también permite generar resultados analíticos detallados que ofrecen una visión clara del comportamiento y patrones de los usuarios en relación con la gestión de residuos.
- **Autorización e integridad de los datos**: la integración de una bitácora de auditoría asegura la integridad de los datos de los usuarios, permitiendo rastrear cualquier cambio en caso necesario. Además, la implementación de JWT proporciona un nivel de seguridad y autenticación avanzado para el *API*, protegiendo así los datos de accesos no autorizados.
- **Análisis de datos**: las funcionalidades del *API* permiten ofrecer a los usuarios información personalizada sobre sus registros de desechos, mostrando el impacto positivo de sus acciones en el medio ambiente. Además, el sistema facilita la generación de análisis globales, permitiendo examinar los comportamientos colectivos de los usuarios y apoyar la toma de decisiones estratégicas.
- **Mejora continua mediante *CI/CD***: la implementación de un *pipeline* de *CI/CD* prepara al sistema para una mejora continua en cada actualización y facilita su escalabilidad a largo plazo. Las pruebas automatizadas aumentan la eficiencia y permiten identificar y corregir errores antes de que afecten el funcionamiento del *API*. Además, este enfoque posibilita un despliegue rápido de cambios, asegurando que las mejoras estén disponibles para los usuarios en tiempo real.
- **Monitoreo y rendimiento del *API***: la integración de ***Sentry*** para monitorear el sistema en producción permite una respuesta rápida ante errores que puedan afectar el funcionamiento del *API*. Además, el monitoreo del rendimiento garantiza que el sistema pueda adaptarse al crecimiento y a la incorporación de nuevas funcionalidades sin comprometer la velocidad y la eficiencia.
- **Escalabilidad y futuras expansiones**: la implementación de *CI/CD*, el diseño optimizado de la base de datos y las herramientas de monitoreo posicionan al proyecto para futuras expansiones. Este diseño permite que el sistema se escale y se introduzcan nuevas funcionalidades sin afectar su estabilidad ni rendimiento, asegurando una evolución fluida y robusta.

8.3. Conclusiones de inteligencia artificial

A partir de la implementación y análisis detallado de los agentes inteligentes desarrollados en este proyecto, se pueden destacar las siguientes conclusiones:

- El modelo de clasificación de residuos basado en la arquitectura *Xception* logró un desempeño sobresaliente, alcanzando una precisión promedio del 98 % en el conjunto de prueba. Esto valida que la elección de la arquitectura *Xception* fue adecuada para abordar la complejidad y variabilidad visual de los residuos urbanos, particularmente en el contexto de la Ciudad de Guatemala.
- La integración de técnicas de *Explainable AI* (*GradCAM, LIME* y *SmoothGrad*) permitió una comprensión profunda de cómo el modelo toma decisiones, aumentando la transparencia y confianza en el sistema. Estos métodos confirmaron que el modelo se enfoca en las características más relevantes de los residuos, validando su efectividad en escenarios con fondos simples y complejos.
- El análisis y validación del modelo en imágenes reales evidenció su robustez y capacidad de generalización en entornos no controlados. Aunque se observó un error de clasificación en una imagen con un contexto visual desafiante, el desempeño general se mantuvo alto, resaltando la aplicabilidad práctica del sistema.
- La implementación de prácticas de *MLOps* con *MLFlow*, aunque a nivel de prueba de concepto (PoC), demostró la viabilidad y la importancia de la gestión y seguimiento del ciclo de vida del modelo. Este enfoque asegura que futuros desarrollos y optimizaciones puedan realizarse de manera eficiente, facilitando el versionamiento y la trazabilidad de experimentos.
- El *chatbot* inteligente basado en la *API* de *OpenAI* mostró un desempeño altamente efectivo en términos de precisión, relevancia temática y tiempo de respuesta. La capacidad del *chatbot* para rechazar preguntas fuera de tema y proporcionar recomendaciones útiles respalda su potencial como herramienta educativa y de asistencia para la ciudadanía en temas de gestión de residuos.
- Las pruebas realizadas con expertos de la Unidad de Control Ambiental de la Municipalidad de la Ciudad de Guatemala validaron la eficacia y relevancia de ambas soluciones. Los agentes inteligentes fueron bien recibidos y se consideraron como herramientas valiosas para fomentar hábitos sostenibles y mejorar la gestión de residuos en la ciudad.
- La implementación de estos agentes inteligentes ofrece una solución viable y sostenible para la Ciudad de Guatemala, alineada con los objetivos de promover la educación ambiental y optimizar el manejo de residuos. La combinación de tecnologías avanzadas y un enfoque práctico crea un impacto positivo tanto en la conciencia ciudadana como en la infraestructura urbana.
- Se cumplieron los objetivos planteados al inicio del proyecto, logrando desarrollar un sistema integral que combina modelos de visión por computadora y un *chatbot* de asistencia con aplicaciones prácticas. La solución propuesta no solo es técnicamente sólida sino también escalable y adaptable a diferentes entornos, ofreciendo un marco que puede extenderse a otras ciudades con necesidades similares.

8.4. Conclusiones de seguridad

- El desarrollo de este proyecto permitió abordar la problemática de la falta de prácticas de seguridad en el ciclo de vida del *software*, logrando una implementación efectiva de medidas que fortalecen tanto la seguridad como la calidad del código en aplicaciones móviles y *backend*.

- El proyecto se basó en el marco de *OWASP SAMM*, aplicando metodologías de gobernanza, diseño, implementación y verificación, lo que contribuyó a una metodología de trabajo integral y proactiva para abordar la seguridad desde el inicio de la creación del *software*.
- Se realizó un análisis exhaustivo utilizando *SonarQube* y *SonarCloud* para la detección y mitigación de vulnerabilidades en el análisis estático del código, contribuyendo a una disminución continua de las fallas de seguridad a lo largo del proyecto. Estas herramientas no solo permitieron identificar problemas en el código, sino también aplicar parches cuando fue necesario, elevando el estándar de calidad y seguridad en el desarrollo del *backend*.
- Para el análisis de la aplicación móvil, se emplearon las tecnologías *MobSF* y *AppSweep*, que facilitaron la identificación de vulnerabilidades específicas del entorno móvil. Sin embargo, debido a limitaciones de tiempo, no se lograron implementar las correcciones recomendadas por estas herramientas. Este proyecto no se detendrá en esta etapa, ya que el equipo continuará trabajando en futuras mejoras y aplicará las recomendaciones obtenidas, con el fin de alcanzar los estándares de seguridad que *MobSF* y *AppSweep* requieren para una aplicación móvil robusta y confiable.
- El equipo experimentó un crecimiento significativo en su cultura de seguridad a través de capacitaciones regulares y pruebas de *phishing*. Las evaluaciones realizadas evidenciaron mejoras en la habilidad del equipo para identificar correos maliciosos, lo que redujo el riesgo de vulnerabilidades humanas en la seguridad del sistema.
- Se lograron detectar algunos permisos que no son necesarios para ciertos roles del equipo, lo cual resalta la importancia de revisar los permisos generados por defecto en *AWS*. Esto permitió aplicar principios de mínimo privilegio, evitando riesgos potenciales por configuraciones de acceso excesivas y mejorando la seguridad general de la infraestructura.
- La cobertura de código alcanzada en *SonarCloud* superó el 80 %, validando la calidad de las pruebas y el compromiso con un desarrollo seguro. La configuración de un *quality check* adicional aseguró que el código cumpliera consistentemente con altos estándares de seguridad y calidad.
- El análisis de seguridad aplicado al código de modelos de inteligencia artificial y *chatbot* desarrollados en el proyecto reflejó una implementación rigurosa de buenas prácticas, asegurando que estos componentes también cumplieran con altos estándares de protección de datos y seguridad operativa.
- Las herramientas y metodologías aplicadas en el proyecto, junto con las prácticas de capacitación continua, no solo contribuyeron a resolver el problema original planteado sino que también establecieron una base sólida para el desarrollo de futuras aplicaciones seguras y confiables.
- Este proyecto no solo facilita la identificación de vulnerabilidades en el código, sino que también apoya el desarrollo de una cultura de seguridad en el equipo y fortalece la postura de seguridad general de la organización, contribuyendo a los estándares de calidad en la industria de desarrollo de *software* seguro.
- El aprendizaje obtenido a nivel académico y profesional resultó invaluable, abarcando desde temas de gestión de accesos y configuraciones de seguridad en infraestructura en la nube hasta el análisis y mitigación de vulnerabilidades en el ciclo de desarrollo de *software*.

8.5. Conclusiones de infraestructura en la nube

La arquitectura en *AWS* ofrece una infraestructura completa y gestionada que incluyó procesamiento, almacenamiento, autenticación y despliegue. La selección de servicios, como *Elastic Beanstalk*, *Amazon RDS, SageMaker* y *S3*, demostró ser efectiva al permitir un desarrollo y despliegue ágil

y sin contratiempos. *AWS* fue una plataforma adecuada para satisfacer las necesidades de escalabilidad, rendimiento y seguridad que requiere una aplicación móvil de propósito social. La facilidad de integración y las capacidades de administración automatizadas permitieron al equipo de desarrollo concentrarse en la funcionalidad y experiencia del usuario, en lugar de en la administración de servidores y configuraciones complejas.

El análisis detallado de los requerimientos de la aplicación fue clave para seleccionar los servicios específicos de *AWS* que pudieran satisfacer las necesidades de procesamiento, almacenamiento y autenticación. Este análisis permitió definir los recursos y escalabilidad que se ajustarían a los patrones de uso previstos para la aplicación. *Elastic Beanstalk* fue seleccionado para el *backend* debido a su capacidad para gestionar el escalado y la carga automáticamente. *Amazon RDS* y *SageMaker* fueron implementados en base a sus especificaciones de administración de bases de datos, procesamiento de IA y autenticación, cumpliendo con las necesidades de cada componente de la aplicación.

La arquitectura desarrollada incorporó todos los componentes necesarios para una infraestructura en la nube completamente operativa y optimizada. *Elastic Beanstalk* y *EC2* gestionaron el *backend* de la aplicación, mientras que *Amazon RDS* facilitó el almacenamiento y gestión de datos mediante *PostgreSQL*. *AWS SageMaker* se utilizó para implementar el modelo de IA encargado de procesar imágenes y texto, mientras que *Amplify* simplificó el manejo de la autenticación de usuarios. *Amazon S3* se configuró para el almacenamiento de archivos e imágenes, integrándose de manera eficiente con los demás servicios. Esta arquitectura detallada permitió asegurar la conectividad y optimización del flujo de datos, proporcionando una experiencia de usuario fluida.

En conclusión, la arquitectura desarrollada proporcionó una plataforma segura y escalable que facilita el despliegue de una aplicación móvil capaz de gestionar funciones avanzadas de autenticación, procesamiento de IA y almacenamiento de datos. La arquitectura resultante no solo se ajusta a las necesidades funcionales de la aplicación, sino que también permite un control efectivo de costos y rendimiento mediante el uso de servicios gestionados y configuraciones de escalado automático. Las conclusiones sugieren que esta solución es adecuada para un crecimiento futuro y que el monitoreo y optimización de costos serán factores críticos en el mantenimiento de la infraestructura.

CAPÍTULO 9

Recomendaciones

9.1. Recomendaciones de experiencia de usuario

- Se recomienda ampliar el enfoque del proyecto a poblaciones más grandes y variadas. El presente proyecto toma como público objetivo la población del departamento de Guatemala, pero es posible ampliar esta población objetivo y de esta manera tener la capacidad de obtener información que permita idear nuevas funcionalidades para ampliar la aplicación y adaptarla a un uso más global.
- Se recomienda a futuros desarrolladores o personas interesadas en el proyecto en general el uso de dispositivos *iOS* para realizar más pruebas al producto final. La mayoría de pruebas fueron realizadas en dispositivos y emuladores de *Android*, por lo que, pese a que la proporción de pruebas cuadra con la información investigada de porcentajes de personas en el departamento de Guatemala que utilizan dispositivos *Android* e *iOS*, se resalta la posibilidad de hacer uso de más dispositivos *iOS* para realizar pruebas.
- Se recomienda a futuros desarrolladores interesados en el proyecto la inclusión de un sistema de actividades lúdicas y más mecánicas en la aplicación que motiven más a sus usuarios a hacer uso de la misma. Un buen ejemplo del enfoque que se le puede dar al proyecto es un enfoque como el de la aplicación Duolingo para aprender diferentes idiomas. Esta aplicación tiene sistemas de punteos, rachas de uso y más información que los usuarios pueden compartir en redes sociales, lo cual es un motivador muy grande a utilizar la misma aplicación.

9.2. Recomendaciones de *backend*, gestión de base de datos y análisis de datos

- **Ampliación de las métricas ambientales**: considerar la inclusión de nuevas métricas de análisis ambiental que permitan ofrecer a los usuarios información más detallada sobre cómo sus registros de desechos impactan positivamente en otras áreas ambientales. Esto podría incluir métricas adicionales relacionadas con la reducción de residuos, ahorro energético, o preservación de recursos naturales.
- **Análisis y fortalecimiento de la seguridad**: realizar auditorías de seguridad periódicas en el *API*, incluyendo pruebas específicas para evaluar su comportamiento frente a posibles

ataques o vulnerabilidades. Este enfoque permitirá identificar y mitigar riesgos proactivamente, asegurando la protección de los datos de los usuarios y la resiliencia del sistema frente a amenazas.

- **Funcionalidades administrativas**: implementar funcionalidades específicas para usuarios administrativos, como municipalidades o centros de reciclaje. Esto permitirá integrar el análisis de datos y facilitar la carga de información sobre desechos cuando sea necesario, mejorando la colaboración y gestión de datos ambientales a nivel institucional.
- **Aporte de los usuarios**: habilitar la opción para que los usuarios puedan sugerir la inclusión de nuevos tipos de desechos o registrar aquellos que no se encuentren disponibles en la aplicación. Esto fomentará la cooperación de la comunidad y permitirá el análisis de nuevos tipos de residuos, ampliando así la base de datos para futuras métricas y estudios.
- **Información adicional sobre los desechos**: evaluar la incorporación de información adicional sobre los desechos, como características específicas o impacto ambiental detallado, lo cual permitirá enriquecer las métricas de análisis de datos y proporcionar a los usuarios una comprensión más completa de sus hábitos de gestión de residuos.

9.3. Recomendaciones de inteligencia artificial

- Se recomienda continuar optimizando el modelo de clasificación *Xception* mediante la experimentación con técnicas avanzadas de *data augmentation* y el ajuste fino de los hiperparámetros. Además, explorar el uso de arquitecturas híbridas y aplicar transferencia de aprendizaje con modelos preentrenados en *datasets* más amplios puede mejorar la capacidad del modelo para generalizar, especialmente cuando se enfrenta a escenarios diversos y complejos en la clasificación de residuos.
- Ampliar el *dataset* con imágenes adicionales, particularmente aquellas capturadas en entornos reales con fondos urbanos complejos y condiciones de iluminación variables, es fundamental. Esto también debería incluir situaciones donde los residuos están parcialmente ocultos o en posiciones inusuales, lo que incrementará la robustez del modelo frente a escenarios prácticos y mejorará su precisión general.
- Aunque la implementación de *MLOps* con *MLFlow* se desarrolló como una prueba de concepto, se recomienda establecer un entorno de *MLOps* completo. Esto incluiría la automatización del ciclo de vida del modelo, el despliegue continuo, el monitoreo en tiempo real y la gestión de versiones. La adición de mecanismos para detectar el *drift* de datos y realizar *rollbacks* rápidamente asegurará la estabilidad y el rendimiento óptimo del sistema a lo largo del tiempo.
- Para el *chatbot* inteligente, sería beneficioso extender sus funcionalidades. Implementar soporte multilingüe y adaptar las respuestas en función de la ubicación geográfica de los usuarios dentro de la Ciudad de Guatemala aumentará la accesibilidad y relevancia del servicio. Además, desarrollar un módulo de aprendizaje continuo, que permita al *chatbot* mejorar sus respuestas a partir de la interacción y retroalimentación de los usuarios, fortalecerá la precisión y la utilidad del asistente.
- La educación y sensibilización de la población son clave para el éxito de estas herramientas. Se sugiere promover campañas educativas enfocadas en el correcto uso de las tecnologías propuestas y la importancia de la gestión adecuada de residuos. Estas iniciativas deben involucrar escuelas, comunidades y organizaciones locales para fomentar hábitos sostenibles y garantizar que la población aproveche al máximo las herramientas desarrolladas.
- Se recomienda fomentar la colaboración con la Municipalidad de la Ciudad de Guatemala y entidades privadas para asegurar un soporte adecuado en la implementación de estas soluciones

tecnológicas. La creación de políticas públicas que impulsen el uso de tecnologías de inteligencia artificial en la gestión de residuos, junto con la inversión en infraestructura de reciclaje, es esencial para el éxito a largo plazo del proyecto. Estas alianzas facilitarán la integración fluida de los sistemas en la vida urbana.

- Realizar evaluaciones periódicas del impacto de los agentes inteligentes en la gestión de residuos y en los hábitos de la población proporcionará datos valiosos para mejorar continuamente. Indicadores como el aumento en la cantidad de residuos correctamente clasificados, la reducción de desechos no reciclables y la satisfacción de los usuarios serán fundamentales para ajustar las herramientas según sea necesario y maximizar su efectividad.
- Se debe seguir mejorando la interpretabilidad del modelo de clasificación *Xception* mediante la investigación y la implementación de técnicas avanzadas de *Explainable AI*. Una mejor transparencia en las decisiones del modelo no solo aumentará la confianza de los usuarios, sino que también permitirá a los desarrolladores identificar y corregir posibles áreas problemáticas en el sistema.
- Finalmente, realizar pruebas adicionales en condiciones adversas, como ambientes con iluminación cambiante o la presencia de objetos distractores, garantizará que el modelo mantenga un desempeño robusto y confiable. También se podría considerar la expansión del proyecto a otras ciudades con desafíos similares en la gestión de residuos, dado que las soluciones aquí desarrolladas son flexibles y pueden adaptarse a diferentes contextos urbanos con éxito.

9.4. Recomendaciones de seguridad

- Se recomienda establecer un programa continuo de capacitación en ciberseguridad para el equipo, basado en las sesiones de formación que mejoraron la detección de *phishing* y fortalecieron la cultura de seguridad del equipo. Esto permitirá que el personal se mantenga actualizado frente a nuevas amenazas y adopte prácticas seguras de forma constante. Además, sería ideal implementar herramientas específicas para la capacitación, tales como simuladores de ataques de *phishing*, plataformas de evaluación de habilidades y recursos de entrenamiento interactivos, que faciliten la práctica de escenarios de seguridad en un entorno controlado y seguro.
- Para futuras etapas del proyecto, se sugiere investigar y evaluar otras herramientas de análisis de código que puedan complementar las capacidades de *SonarQube* y *SonarCloud*, particularmente en áreas donde se requiera mayor especificidad en la detección de vulnerabilidades o una integración más robusta en entornos cambiantes de desarrollo. Aunque *SonarQube* es una herramienta poderosa y ampliamente utilizada para el análisis de calidad de código y detección de vulnerabilidades, presenta algunas limitaciones. En primer lugar, su cobertura en la detección de vulnerabilidades específicas puede ser limitada, ya que se enfoca principalmente en la calidad del código y errores comunes, pero no siempre detecta vulnerabilidades avanzadas, especialmente aquellas relacionadas con el análisis dinámico o la seguridad de la infraestructura. Además, aunque *SonarQube* permite la integración continua con diversas plataformas, equipos de desarrollo que operan en entornos altamente cambiantes podrían encontrar dificultades al configurar y personalizar la herramienta para su flujo de trabajo o tecnologías emergentes. Por último, *SonarQube*, a pesar de ser compatible con muchos lenguajes, puede no ser tan eficiente al detectar problemas en lenguajes menos comunes o más nuevos, lo que podría requerir el uso de herramientas adicionales especializadas en esas áreas.
- Dado el uso de bibliotecas de terceros en el proyecto, se sugiere integrar herramientas de monitoreo de vulnerabilidades en dependencias externas en los *pipelines* de desarrollo. Estas herramientas permiten detectar vulnerabilidades en tiempo real y aplicar actualizaciones de seguridad de forma proactiva, reduciendo los riesgos asociados con dependencias externas en la aplicación.

- Para mejorar la confiabilidad del código, se sugiere aumentar la cobertura de pruebas en áreas críticas de la aplicación, superando el 82 % alcanzado en este proyecto. Enfocarse en módulos de alto impacto permitirá asegurar una mayor estabilidad y seguridad en componentes que puedan representar riesgos significativos para la aplicación.
- Se sugiere seguir los consejos y recomendaciones proporcionados por las tecnologías de *MobSF* y *AppSweep* en el análisis de aplicaciones móviles, ya que sus reportes detallados ayudan a identificar configuraciones deficientes y áreas de mejora en la seguridad del código. Además, para futuras etapas del proyecto, se recomienda investigar y evaluar herramientas adicionales de análisis de seguridad que profundicen en las evaluaciones de seguridad de aplicaciones móviles, ofreciendo mayor cobertura y especificidad en la detección de vulnerabilidades que sean críticas para este tipo de entornos.
- A fin de optimizar el flujo de trabajo y asegurar escaneos constantes de seguridad, se recomienda implementar un *pipeline* de *CI/CD* para la aplicación móvil. Esto permitirá que herramientas como *MobSF* y *AppSweep* se ejecuten automáticamente en cada despliegue o construcción de la aplicación, eliminando la necesidad de cargas manuales y garantizando que cada actualización pase por un análisis de seguridad inmediato. Esta integración continua contribuirá a mantener un alto estándar de seguridad en cada versión publicada de la aplicación.

9.5. Recomendaciones de infraestructura en la nube

En base a los resultados obtenidos y el análisis de los recursos utilizados en la implementación de la aplicación en *AWS*, se sugieren las siguientes recomendaciones para optimizar el rendimiento, gestionar los costos y mejorar la eficiencia de la infraestructura en la nube.

- **Monitoreo continuo y optimización de recursos**: se recomienda implementar un monitoreo constante a través de *Amazon CloudWatch* para supervisar el uso de los recursos, el tráfico de la aplicación y las métricas de rendimiento. Esto permitirá identificar patrones de uso, anticipar aumentos en la demanda y ajustar los recursos de manera proactiva. Además, el monitoreo de métricas ayudará a tomar decisiones informadas sobre el escalado y optimización de la infraestructura.
- **Planificación de escalado y tiempo de actividad de *SageMaker***: dado que *AWS SageMaker* incurre en costos significativos en uso continuo, es recomendable revisar los patrones de uso del modelo de IA y ajustar el tiempo de actividad del *endpoint* según las necesidades de la aplicación. Si la demanda varía a lo largo del día, se puede optar por activar el *endpoint* en horas de alta afluencia y desactivarlo durante periodos de baja demanda para reducir costos.
- **Evaluación periódica de las clases de almacenamiento en *Amazon S3***: para optimizar el almacenamiento de datos, se recomienda revisar la frecuencia de acceso a los archivos y considerar el uso de diferentes clases de almacenamiento en *Amazon S3*. Los datos de uso poco frecuente pueden trasladarse a clases de almacenamiento más económicas, como `Infrequent Access` o `Glacier`, lo cual permitirá reducir costos sin afectar el rendimiento.
- **Implementación de estrategias de *backup* y recuperación de RDS**: se sugiere establecer políticas de *backup* frecuentes para la base de datos en *Amazon RDS* y considerar la opción de copias de seguridad geo-redundantes en casos críticos. Las estrategias de *backup* y recuperación asegurarán la continuidad del servicio en caso de fallos y permitirán restaurar datos de forma rápida, minimizando el impacto en la operación.
- **Evaluación de costos y revisión de la infraestructura cada trimestre**: la demanda y uso de la aplicación pueden variar con el tiempo, por lo que se recomienda realizar una revisión trimestral de los costos y la infraestructura. Esta evaluación permitirá ajustar los

recursos en función de las necesidades actuales y explorar alternativas de configuración más rentables, asegurando que la aplicación mantenga una infraestructura eficiente sin incurrir en gastos innecesarios.

Estas recomendaciones apuntan a mejorar la eficiencia y control de costos de la infraestructura en *AWS*, permitiendo que la aplicación se mantenga escalable y rentable mientras ofrece un alto rendimiento y seguridad. Al implementar estas estrategias, el proyecto podrá adaptarse a cambios en la demanda y en el entorno de la nube, asegurando la sostenibilidad de la aplicación a largo plazo.

Bibliografía

[1] J. Rodríguez, *Análisis de la generación de desechos sólidos en Guatemala.* Revista Análisis de la Realidad Nacional, 2023. [Online]. Available: https://ipn.usac.edu.gt/wp-content/uploads/2023/09/Analisis-de-la-generacion-de-desechos-solidos-en-Guatemala.pdf

[2] H. Lone and F. Buchh, "Social causatums of improper waste management," *Innovative Waste Management Technologies for Sustainable Development*, 2019.

[3] S. Ramakrishna, M. Pervaiz, J. Tjong, P. Ghisellini, and M. Sain, "Low-carbon materials: Genesis, thoughts, case study, and perspectives," *Circular Economy and Sustainability*, vol. 2, 2022.

[4] S. Gibbons, "Design thinking 101," Nielsen Norman Group, abril 2024, consultado el 13 de octubre de 2024. [Online]. Available: https://www.nngroup.com/articles/design-thinking/

[5] Hack a Boss, "Frontend y backend explicados," 2023. [Online]. Available: https://www.hackaboss.com/blog/frontend-backend-explicados

[6] Nick, "What is rest api?" 2020. [Online]. Available: https://blog.iron.io/what-is-rest-api/

[7] P. Pedamkar, "Second normal form," 2020. [Online]. Available: https://www.educba.com/second-normal-form/

[8] PopulationPyramid.net, "Pirámides de población del mundo desde 1950 a 2100," 2020, s. f. [Online]. Available: https://www.populationpyramid.net/es/guatemala/2020/

[9] M. Ferris. (2023, Mar) How does automated testing for ci/cd help improve software quality and delivery. Accessed: 2023-10-31. [Online]. Available: https://qameta.io/blog/automated-testing-for-ci-cd/

[10] Intergovernmental Panel on Climate Change, *Global Warming of 1.5°C: Summary for Policymakers.* IPCC, 2018. [Online]. Available: https://www.ipcc.ch/sr15/

[11] Food and Agriculture Organization of the United Nations, *The State of the World's Forests 2020: Forests, Biodiversity and People.* FAO, 2020. [Online]. Available: https://doi.org/10.4060/ca8642en

[12] World Bank, *What a Waste 2.0: A Global Snapshot of Solid Waste Management to 2050.* World Bank Group, 2018. [Online]. Available: https://openknowledge.worldbank.org/handle/10986/30317

[13] Ministerio de Ambiente y Recursos Naturales de Guatemala, *Diagnóstico de la gestión integral de los residuos sólidos y su potencial aprovechamiento en Guatemala.* MARN, 2016. [Online]. Available: https://www.marn.gob.gt/s/portal/index.php/documentos/centro-de-documentacion/file/2650-diagnostico-de-la-gestion-integral-de-residuos-solidos-y-su-potencial-aprovechamiento-en-guatemala

[14] D. Hoornweg and P. Bhada-Tata, "What a waste: A global review of solid waste management," *World Bank Urban Development Series Knowledge Papers*, vol. 15, pp. 1–116, 2012. [Online]. Available: https://documents.worldbank.org/curated/en/302341468126264791/What-a-waste-a-global-review-of-solid-waste-management

[15] J. R. Jambeck, R. Geyer, C. Wilcox, T. R. Siegler, M. Perryman, A. Andrady, R. Narayan, and K. L. Law, "Plastic waste inputs from land into the ocean," *Science*, vol. 347, no. 6223, pp. 768–771, 2015.

[16] I. A. Al-Khatib, S. Kontogianni, and H. Abu Nabaa, "Solid waste characterization, quantification and management practices in developing countries. a case study: Nablus district - palestine," *Journal of Environmental Management*, vol. 91, no. 5, pp. 1131–1138, 2010.

[17] K. R. Smith, A. Woodward, D. Campbell-Lendrum, D. D. Chadee, Y. Honda, Q. Liu, J. M. Olwoch, B. Revich, and R. Sauerborn, "Human health: Impacts, adaptation, and co-benefits," *Climate Change 2014: Impacts, Adaptation, and Vulnerability. Part A: Global and Sectoral Aspects. Contribution of Working Group II to the Fifth Assessment Report of the Intergovernmental Panel on Climate Change*, pp. 709–754, 2014. [Online]. Available: https://www.ipcc.ch/site/assets/uploads/2018/02/WGIIAR5-Chap11_FINAL.pdf

[18] N. Watts, M. Amann, N. Arnell, S. Ayeb-Karlsson, J. Beagley, K. Belesova, M. Boykoff, P. Byass, W. Cai, and D. Campbell-Lendrum, "The 2018 report of the lancet countdown on health and climate change: Shaping the health of nations for centuries to come," *The Lancet*, vol. 392, no. 10163, pp. 2479–2514, 2018.

[19] C. Parmesan and G. Yohe, "A globally coherent fingerprint of climate change impacts across natural systems," *Nature*, vol. 421, pp. 37–42, 2003.

[20] G. Tchobanoglous and F. Kreith, *Handbook of Solid Waste Management*, 2nd ed. McGraw-Hill, 2002.

[21] L. M. Johannessen and G. Boyer, "Observations of solid waste landfills in developing countries: Africa, asia, and latin america," *Urban Development Division, The World Bank*, pp. 1–51, 1999. [Online]. Available: http://documents.worldbank.org/curated/en/217071468740982081/Observations-of-solid-waste-landfills-in-developing-countries-Africa-Asia-and-Latin-America

[22] O. Zerbock, "Urban solid waste management: Waste reduction in developing nations," Michigan Technological University, Tech. Rep., 2003. [Online]. Available: https://www.mtu.edu/peacecorps/programs/civil/pdfs/waste-reduction.pdf

[23] W. H. O. (WHO), "Environmental health inequalities in europe: Second assessment report," 2018. [Online]. Available: https://www.euro.who.int/en/publications/abstracts/environmental-health-inequalities-in-europe.-second-assessment-report-2018

[24] C. P. Baldé, V. Forti, V. Gray, R. Kuehr, and P. Stegmann, "The global e-waste monitor 2020: Quantities, flows, and the circular economy potential," *United Nations University (UNU), International Telecommunication Union (ITU) International Solid Waste Association (ISWA)*, pp. 1–120, 2020. [Online]. Available: https://ewastemonitor.info/

[25] European Environment Agency, "Hazardous waste in europe: Infographic," European Environment Agency (EEA), Tech. Rep., 2021. [Online]. Available: https://www.eea.europa.eu/publications/hazardous-waste-in-europe-infographic

[26] J. Hopewell, R. Dvorak, and E. Kosior, "Plastics recycling: Challenges and opportunities," *Philosophical Transactions of the Royal Society B: Biological Sciences*, vol. 364, no. 1526, pp. 2115–2126, 2009.

[27] N. Singh and P. Sharma, "Mechanistic implications of plastic degradation for environmental waste management," *Environmental Science & Technology*, vol. 54, no. 24, pp. 15 582–15 594, 2020.

[28] L. Matsakas, Q. Gao, S. Jansson, U. Rova, and P. Christakopoulos, "Waste biorefinery technologies for multiple bioproducts: Present and future prospects," *Bioresource Technology*, vol. 241, pp. 1084–1096, 2017.

[29] L. A. Wright, S. Kemp, and I. Williams, "'carbon footprinting': towards a universally accepted definition?" *Carbon Management*, vol. 2, no. 1, pp. 61–72, 2011.

[30] M. Hedayati and X. Li, *Sustainable Supply Chains: Strategies and the Role of Innovation.* Routledge, 2021.

[31] B. Owen and S. Kemp, "Individual contributions to carbon reduction," in *Environmental Footprints and Eco-design of Products and Processes.* Springer, 2020, pp. 25–36.

[32] D. Kammer and P. Knaus, "Digital tools for carbon footprint management," in *Green Technologies and Eco-Innovation.* Springer, 2022, pp. 112–125.

[33] S. Muller, "The role of the united nations in combating climate change," *Global Governance*, vol. 28, no. 2, pp. 83–97, 2022.

[34] T. Wiedmann and J. Minx, *A Definition of "Carbon Footprint".* Hauppauge, NY: Elsevier, 2008.

[35] D. Pandey, M. Agrawal, and J. S. Pandey, *Carbon Footprint: Current Methods of Estimation.* New York: Nova Science Publishers, 2011.

[36] International Renewable Energy Agency, "Global energy transformation: A roadmap to 2050," IRENA, Tech. Rep., 2018. [Online]. Available: https://www.irena.org/publications/2018/Apr/Global-Energy-Transition-A-Roadmap-to-2050

[37] K. Gillingham, D. Rapson, and G. Wagner, "The rebound effect and energy efficiency policy," *Review of Environmental Economics and Policy*, vol. 10, no. 1, pp. 68–88, 2016.

[38] R. Sullivan, "Corporate carbon footprint management: The role of transparency and standards," *Journal of Business Ethics*, vol. 150, no. 5, pp. 1019–1033, 2017.

[39] Ellen MacArthur Foundation, "The circular economy: A transformative covid-19 strategy," Ellen MacArthur Foundation, Tech. Rep., 2020. [Online]. Available: https://www.ellenmacarthurfoundation.org/publications/the-circular-economy-a-transformative-covid-19-strategy

[40] L. Whitmarsh, G. Seyfang, and S. O'Neill, "Public engagement with climate change: The role of perceived risk and affect," *Environmental Research Letters*, vol. 6, no. 4, p. 044017, 2011.

[41] M. Niero, H. Wenzel, L. Helsen, and al., "The role of waste management in the reduction of carbon footprint," *Journal of Cleaner Production*, vol. 171, pp. 1487–1495, 2018.

[42] N. J. Themelis and P. Ulloa, "The role of recycling in the reduction of greenhouse gas emissions," *Waste Management & Research*, vol. 25, no. 3, pp. 275–279, 2007.

[43] M. P. Bernal, J. A. Alburquerque, and R. Moral, "Composting as a tool for carbon sequestration in soils," *Waste Management*, vol. 63, pp. 208–216, 2017.

[44] Intergovernmental Panel on Climate Change, *Fifth Assessment Report: Climate Change 2014*. Cambridge: Cambridge University Press, 2014. [Online]. Available: https://www.ipcc.ch/report/ar5/

[45] L. Zhang, W. Zhao, F. Yu, and Y. Zhang, "Internet of things (iot) for environmental monitoring: A survey," *Environmental Monitoring and Assessment*, vol. 191, no. 2, p. 52, 2019.

[46] D. Rolnick, A. D'Amour, J. Maccarthy, and al., "Tackling climate change with machine learning," *ACM Computing Surveys*, vol. 52, no. 3, pp. 1–37, 2019.

[47] K. G. Liakos, P. Busato, D. Moshou, and al., "A review of the application of precision agriculture technologies for sustainable crop production," *Computers and Electronics in Agriculture*, vol. 151, pp. 114–132, 2018.

[48] H. Hao, L. Zhang, Q. Liu, and X. Li, "Energy management systems for sustainable development," *Sustainable Cities and Society*, vol. 54, p. 101969, 2020.

[49] Ellen MacArthur Foundation, *Completing the Picture: How the Circular Economy Tackles Climate Change*. Cowes: Ellen MacArthur Foundation, 2019. [Online]. Available: https://www.ellenmacarthurfoundation.org/publications/completing-the-picture-how-the-circular-economy-tackles-climate-change

[50] J. Murray, C. Kim, and R. Singh, "Blockchain technology for sustainable supply chains," *Journal of Cleaner Production*, vol. 245, p. 118824, 2020.

[51] L. Casado-Aranda, J. Sánchez-Fernández, and I. Álvarez Merino, "Gamification and mobile apps for environmental education: An analysis of effectiveness," *Computers in Human Behavior*, vol. 116, p. 106618, 2021.

[52] C. de la República de Guatemala, *Ley de Protección y Mejoramiento del Medio Ambiente (Decreto 68-86)*. Gobierno de Guatemala, 1986.

[53] G. de Guatemala, *Reglamento para la Gestión Integral de los Residuos y Desechos Sólidos Comunes (Acuerdo Gubernativo 164-2021)*. Gobierno de Guatemala, 2021.

[54] L. Gordillo and C. F. Presa, "Guatemala - new regulation for waste management," Lexology, Tech. Rep., August 17 2021. [Online]. Available: https://www.lexology.com/library/detail.aspx?g=6cac48fb-a179-4857-b257-38a07010efa3

[55] Ministerio de Ambiente y Recursos Naturales (MARN), *Política Nacional de Residuos Sólidos*. Gobierno de Guatemala, 2010.

[56] B. Legal, "The pending solution for garbage disposal in guatemala," October 25 2023. [Online]. Available: https://blplegal.com/regulatory-harmonization-the-pending-solution-for-garbage-disposal-in-guatemala/

[57] FAO, "Transforming food and agriculture to achieve the sdgs," Food and Agriculture Organization of the United Nations, Tech. Rep., 2018. [Online]. Available: http://www.fao.org/3/I9900EN/i9900en.pdf

[58] E. M. Foundation, *What is a Circular Economy?*, 2019, available at https://www.ellenmacarthurfoundation.org/circular-economy.

[59] European Commission, "A european strategy for plastics in a circular economy," European Commission, Tech. Rep., 2018. [Online]. Available: https://eur-lex.europa.eu/legal-content/EN/TXT/PDF/?uri=CELEX:52018DC0028&from=EN

[60] IPNUSAC, "Análisis de la generación de desechos sólidos en guatemala," Instituto de Problemas Nacionales de la Universidad de San Carlos de Guatemala (IPNUSAC), Tech. Rep., 2023. [Online]. Available: https://ipn.usac.edu.gt/wp-content/uploads/2023/09/Analisis-de-la-generacion-de-desechos-solidos-en-Guatemala.pdf

[61] J. J. Garrett, *The Elements of User Experience: User-Centered Design for the Web and Beyond*, 2nd ed. Berkeley, CA: New Riders, 2016.

[62] D. A. Norman, *The Design of Everyday Things*, revised and expanded edition ed. New York, NY: Basic Books, 2013.

[63] A. Cooper, R. Reimann, D. Cronin, and C. Noessel, *About Face: The Essentials of Interaction Design*, 4th ed. Indianapolis, IN: Wiley, 2014.

[64] Y. Rogers, H. Sharp, and J. Preece, *Interaction Design: Beyond Human-Computer Interaction*, 5th ed. Hoboken, NJ: Wiley, 2020.

[65] H. Beyer and K. Holtzblatt, *Contextual Design: Design for Life*, 2nd ed. San Francisco, CA: Morgan Kaufmann, 2019.

[66] J. Gothelf and J. Seiden, *Lean UX: Applying Lean Principles to Improve User Experience*, 2nd ed. Sebastopol, CA: O'Reilly Media, 2017.

[67] World Wide Web Consortium (W3C), "Web content accessibility guidelines (wcag) 2.1," 2021. [Online]. Available: https://www.w3.org/TR/WCAG21/

[68] S. Kujala, M. Kauppinen, A. Anttila, and H. Tokkonen, "User involvement: A review of the benefits and challenges," *International Journal of Human-Computer Interaction*, vol. 36, pp. 385–396, 2020.

[69] M. Hassenzahl, *Experience Design: Technology for All the Right Reasons.* San Rafael, CA: Morgan & Claypool Publishers, 2018.

[70] Google, "Mobile speed matters: How user perception impacts business outcomes," 2019. [Online]. Available: https://www.thinkwithgoogle.com/marketing-strategies/app-and-mobile/mobile-speed-matters/

[71] G. McLean and K. Osei-Frimpong, "Examining the determinants and outcomes of mobile app engagement – a longitudinal perspective," *Computers in Human Behavior*, vol. 84, pp. 392–403, 2019.

[72] D. A. Norman and J. Nielsen, "Emotion & design: Attractive things work better," *Interactions*, vol. 27, pp. 45–48, 2020.

[73] J. J. Garrett, *The Elements of User Experience: User-Centered Design for the Web and Beyond*, 3rd ed. Berkeley, CA: New Riders, 2020.

[74] T. Brown, *Change by Design: How Design Thinking Creates New Alternatives for Business and Society*, updated and expanded ed. New York, NY: HarperBusiness, 2019.

[75] International Organization for Standardization, *ISO 9241-210:2019 Ergonomics of human-system interaction — Part 210: Human-centred design for interactive systems.* International Organization for Standardization (ISO), 2019.

[76] D. A. Norman, *Emotional Design: Why We Love (or Hate) Everyday Things.* New York, NY: Basic Books, 2004.

[77] J. J. Garrett, *The Elements of User Experience: User-Centered Design for the Web and Beyond*, 2nd ed. Berkeley, CA: New Riders, 2011.

[78] J. Nielsen and D. Norman, "The definition of user experience (ux)," 2012, accessed: 2024-09-22. [Online]. Available: https://www.nngroup.com/articles/definition-user-experience/

[79] M. Hassenzahl, "The thing and i: Understanding the relationship between user and product," in *Funology: From Usability to Enjoyment*, M. A. Blythe, K. Overbeeke, A. F. Monk, and P. C. Wright, Eds. Dordrecht, Netherlands: Kluwer Academic Publishers, 2003, pp. 31–42.

[80] J. Nielsen, *Usability Engineering.* Morgan Kaufmann, 1994.

[81] B. Shneiderman, C. Plaisant, M. Cohen, and S. Jacobs, *Designing the User Interface: Strategies for Effective Human-Computer Interaction*, 5th ed. Boston, MA: Addison-Wesley, 2010.

[82] S. Krug, *Don't Make Me Think, Revisited: A Common Sense Approach to Web Usability*, 3rd ed. Berkeley, CA: New Riders, 2014.

[83] J. Johnson, *Designing with the Mind in Mind: Simple Guide to Understanding User Interface Design Guidelines*, 3rd ed. San Francisco, CA: Morgan Kaufmann, 2020.

[84] Google, "Material design guidelines," 2014, accessed: 2024-09-22. [Online]. Available: https://material.io/design

[85] J. Rubin and D. Chisnell, *Handbook of Usability Testing: How to Plan, Design, and Conduct Effective Tests*, 2nd ed. Indianapolis, IN: Wiley, 2008.

[86] M. Hassenzahl, *Experience Design: Technology for All the Right Reasons.* San Rafael, CA: Morgan & Claypool Publishers, 2010.

[87] A. Martínez, *Investigación cualitativa y análisis de contenido: Guía metodológica.* Editorial Alianza, 2019.

[88] J. W. Creswell, *Research Design: Qualitative, Quantitative, and Mixed Methods Approaches.* Thousand Oaks, CA: SAGE Publications, 2014.

[89] A. Fink, *How to Conduct Surveys: A Step-by-Step Guide.* Thousand Oaks, CA: SAGE Publications, 2015.

[90] R. A. Krueger and M. A. Casey, *Focus Groups: A Practical Guide for Applied Research.* Thousand Oaks, CA: SAGE Publications, 2014.

[91] M. Q. Patton, *Qualitative Research and Evaluation Methods.* Thousand Oaks, CA: SAGE Publications, 2015.

[92] C. Yom, *Secondary Data Analysis: Techniques for Comparing Social Phenomena.* New York, NY: Palgrave Macmillan, 2019.

[93] S. Lohr, *Sampling: Design and Analysis.* Cengage Learning, 2019.

[94] S. K. Thompson, *Sampling.* Wiley, 2012.

[95] E. Babbie, *The Basics of Social Research.* Cengage Learning, 2015.

[96] C. Noy, "Sampling knowledge: The hermeneutics of snowball sampling in qualitative research," *International Journal of Social Research Methodology*, vol. 11, no. 4, pp. 327–344, 2008.

[97] N. Golafshani, "Understanding reliability and validity in qualitative research," *The Qualitative Report*, vol. 8, no. 4, pp. 597–607, 2003.

[98] A. Field, *Discovering Statistics Using IBM SPSS Statistics*, 4th ed. SAGE Publications, 2013.

[99] J. E. DeVoe, "Understanding reliability and validity in qualitative research," *Journal of Social Work Education*, vol. 52, no. 2, pp. 158–165, 2016.

[100] J. Nielsen, *Usability Engineering.* Morgan Kaufmann, 2020.

[101] T. Brown, *Change by Design: How Design Thinking Creates New Alternatives for Business and Society.* HarperBusiness, 2009.

[102] C. Snyder, *Paper Prototyping: The Fast and Easy Way to Design and Refine User Interfaces.* Morgan Kaufmann, 2003.

[103] J. J. Garrett, *The Elements of User Experience: User-Centered Design for the Web and Beyond.* New Riders, 2010.

[104] R. A. Virzi, "Refining the test phase of usability evaluation: How many subjects is enough?" *Human Factors*, vol. 34, no. 4, pp. 457–468, 1992.

[105] E. Goodman, M. Kuniavsky, and A. Moed, *Observing the User Experience: A Practitioner's Guide to User Research.* Morgan Kaufmann, 2012.

[106] J. Preece, Y. Rogers, and H. Sharp, *Interaction Design: Beyond Human-Computer Interaction.* Wiley, 2015.

[107] T. Granollers, J. Lorés, and J. J. Cañas, *Diseño de interfaces de usuario: hacia un diseño centrado en el usuario.* Editorial UOC, 2005.

[108] J. Balliet, *Collaborative Design: Using Figma to Enhance Your Workflow.* Independently Published, 2020.

[109] M. Rodríguez, *Desarrollo de Aplicaciones Móviles para Android e iOS.* Editorial Técnica, 2020.

[110] J. Godfrey, "Mobile app development in 2020: Trends and techniques," *International Journal of Software Engineering*, vol. 12, no. 3, pp. 113–128, 2021.

[111] J. Cerdá and P. Navarro, *Aplicaciones Nativas vs. Aplicaciones Híbridas: Un análisis comparativo.* McGraw Hill, 2020.

[112] S. Kumar, *React Native and Flutter: A Comprehensive Guide to Mobile Development.* Packt Publishing, 2021.

[113] J. Silva and A. Martínez, "Hybrid apps: Balancing performance and flexibility," *Journal of Mobile Computing*, vol. 10, no. 1, pp. 65–89, 2021.

[114] S. Bohner and S. Roberts, "The mobile app development life cycle: Phases and challenges," *Software Engineering Today*, vol. 34, no. 5, pp. 35–54, 2020.

[115] M. Taylor, "Mobile backend as a service (mbaas): Integrating cloud solutions into mobile applications," *Journal of Cloud Computing*, vol. 8, no. 4, pp. 101–118, 2021.

[116] T. Myers, *Testing and Maintaining Mobile Applications: Ensuring Quality and Performance.* Springer, 2020.

[117] R. Sebesta, *Concepts of Programming Languages.* Pearson, 2019.

[118] A. Downey, *Think Python: How to Think Like a Computer Scientist.* O'Reilly Media, 2020.

[119] M. Fowler, *Patterns of Enterprise Application Architecture.* Addison-Wesley, 2004.

[120] M. Risch, *API Design Patterns: A Comprehensive Guide to API Development.* Addison-Wesley, 2020.

[121] B. Braithwaite, *Designing Web APIs: Building APIs That Developers Love.* Manning Publications, 2021.

[122] C. Johnson, *Building Microservices with ASP.NET Core: Develop, Test, and Deploy Cross-platform Services in the Cloud.* Packt Publishing, 2021.

[123] L. Johansson, *Effective Unit Testing: A Guide for Java Developers.* Manning Publications, 2021.

[124] D. Hunt, "Optimizing react native apps for performance," *Journal of Mobile Development*, vol. 9, no. 3, pp. 145–158, 2021.

[125] M. Q. Patton, *Utilization-focused evaluation.* Sage Publications, 2008.

[126] M. Friedman, *Trying hard is not good enough: How to produce measurable improvements for customers and communities.* Trafford Publishing, 2005.

[127] C. H. Weiss, *Evaluation: Methods for studying programs and policies.* Prentice Hall, 1998.

[128] M. Scriven, *Evaluation thesaurus.* Sage, 1991.

[129] D. Parmenter, *Key Performance Indicators: Developing, Implementing, and Using Winning KPIs.* John Wiley & Sons, 2015.

[130] T. Bovaird and E. Löffler, *Public Management and Governance.* Routledge, 2016.

[131] M. Bernardes and S. Zanoni, "Sustainability and waste management: Key performance indicators," *Journal of Environmental Sustainability*, vol. 8, no. 2, pp. 20–35, 2020.

[132] K. Doerner, *Project Management: Balancing Time, Cost, and Quality.* Springer, 2018.

[133] R. Hartson and P. S. Pyla, *The UX Book: Process and Guidelines for Ensuring a Quality User Experience.* Elsevier, 2012.

[134] M. Abrams and S. Helms, "Task completion time as a measure of usability: Practical implications," *International Journal of Human-Computer Interaction*, vol. 29, no. 4, pp. 220–234, 2013.

[135] J. Brooke, "Sus: A quick and dirty usability scale," *Usability Evaluation in Industry*, pp. 189–194, 1996.

[136] J. W. Creswell and J. D. Creswell, *Research Design: Qualitative, Quantitative, and Mixed Methods Approaches.* Sage Publications, 2017.

[137] R. K. Yin, *Case Study Research: Design and Methods.* Sage Publications, 2009.

[138] C. Seale, *The Quality of Qualitative Research.* Sage Publications, 1999.

[139] B. Flyvbjerg, "Five misunderstandings about case-study research," *Qualitative Inquiry*, vol. 12, no. 2, pp. 219–245, 2006.

[140] A. Kumar, *Backend Development Fundamentals.* New York, USA: Tech Books Publishing, 2020.

[141] L. Pacheco and J. Smith, "Understanding the role of backend in modern web applications," *Journal of Web Development*, vol. 12, no. 3, pp. 45–58, 2019.

[142] M. Karan, *Backend Architecture and Design.* San Francisco, CA: Coding Press, 2021.

[143] S. Williams and M. Johnson, "Essential components of backend development," *International Journal of Software Engineering*, vol. 18, no. 2, pp. 23–36, 2020.

[144] S. Tilkov and S. Vinoski, "Node.js: Using javascript to build high-performance network programs," *IEEE Internet Computing*, vol. 14, no. 6, pp. 80–83, 2010.

[145] A. Holovaty and J. Kaplan-Moss, *The Definitive Guide to Django: Web Development Done Right.* Apress, 2009.

[146] M. Hartl, *Ruby on Rails Tutorial: Learn Web Development with Rails*, 3rd ed. Boston, MA: Addison-Wesley, 2016.

[147] M. Stonebraker, "Sql databases v. nosql databases," *Communications of the ACM*, vol. 53, no. 4, pp. 10–11, 2010.

[148] G. Harrison, *Next Generation Databases: NoSQL and Big Data*. Apress, 2015.

[149] R. T. Fielding, "Architectural styles and the design of network-based software architectures," Ph.D. dissertation, University of California, Irvine, 2000.

[150] R. Elmasri and S. B. Navathe, *Fundamentals of Database Systems*, 7th ed. Pearson, 2015.

[151] E. F. Codd, "A relational model of data for large shared data banks," *Communications of the ACM*, vol. 13, no. 6, pp. 377–387, 1970.

[152] J. Gray and A. Reuter, *Transaction Processing: Concepts and Techniques*. Morgan Kaufmann, 1992.

[153] L. Richardson and S. Ruby, *RESTful Web Services*. Sebastopol, CA: O'Reilly Media, 2007.

[154] D. Hardt, "The oauth 2.0 authorization framework," 2012, rFC 6749, Internet Engineering Task Force (IETF). [Online]. Available: https://tools.ietf.org/html/rfc6749

[155] M. Jacobson, *RESTful Web Services Cookbook: Solutions for Improving Scalability and Simplicity*. O'Reilly Media, 2009.

[156] J. Han, J. Pei, and M. Kamber, *Data Mining: Concepts and Techniques*, 3rd ed. Morgan Kaufmann, 2011.

[157] W. McKinney, *Python for Data Analysis: Data Wrangling with Pandas, NumPy, and IPython*, 2nd ed. O'Reilly Media, 2017.

[158] M. Hailperin, B. Kaiser, and K. Pfister, *Operating Systems and Middleware: Supporting Controlled Interaction*. Princeton University Press, 2007.

[159] M. Fowler and J. Lewis, *Continuous Integration and Continuous Delivery: Principles and Practices*. New York, USA: Tech Books, 2018.

[160] J. Humble and D. Farley, *Continuous Delivery: Reliable Software Releases through Build, Test, and Deployment Automation*. Boston, MA: Addison-Wesley Professional, 2010.

[161] L. Bass, I. Weber, and L. Zhu, "Devops: A software architect's perspective," *IEEE Software*, vol. 33, no. 3, pp. 94–97, 2016.

[162] M. Feathers, *Test Driven Development for Embedded C*. Boston, MA: Addison-Wesley Professional, 2019.

[163] L. Meyer and F. O'Donnell, "Testing restful apis in ci/cd pipelines: Best practices and tools," *Journal of Software Testing*, vol. 18, no. 3, pp. 120–135, 2021.

[164] A. Kaplan and T. Roberson, "Automated testing in continuous integration: A practical guide," *Journal of Software Testing*, vol. 15, no. 2, pp. 90–105, 2020.

[165] J. Wallace and E. Gordon, "Scalable api testing in ci/cd environments," *Software Quality Journal*, vol. 27, no. 4, pp. 231–245, 2019.

[166] H. Demers and J. Kim, "Automated testing techniques for restful apis in ci/cd," *International Journal of Software Engineering and Applications*, vol. 11, no. 2, pp. 43–60, 2020.

[167] L. Jones, *Effective Monitoring and Alerting: For Web Operations*. Sebastopol, CA: O'Reilly Media, 2021.

[168] S. Kim and P. Allen, "Real-time monitoring in ci/cd pipelines: Techniques and tools," *Software Development Journal*, vol. 23, no. 4, pp. 210–225, 2022.

[169] C. Lewis and B. Thompson, "Observability and monitoring for high-performance apis," *Journal of DevOps and Observability*, vol. 5, no. 1, pp. 19–33, 2021.

[170] A. Patel and M. Lee, "Real-time monitoring techniques in ci/cd: Tools and methodologies," *Journal of Software Development and Operations*, vol. 12, no. 4, pp. 56–72, 2020.

[171] D. Meyer, "Automating deployment processes: Challenges and solutions in ci/cd," *Journal of DevOps and Agile Development*, vol. 10, no. 3, pp. 45–58, 2020.

[172] C. Brown and R. Allen, "Efficiency in automated ci/cd pipelines for restful apis," *Journal of Software Engineering and Automation*, vol. 17, no. 2, pp. 78–89, 2020.

[173] K. Smith and R. Williams, *CI/CD Tools and Pipelines: Best Practices for Automation.* San Francisco, CA: IT Books Press, 2021.

[174] R. Kim and P. Moore, "Continuous deployment strategies for microservices and apis," *International Journal of Software Engineering*, vol. 19, no. 3, pp. 150–168, 2021.

[175] E. Johnson and V. Sanders, "Automating ci/cd with github actions: Streamlined workflows for modern development," *Journal of DevOps and Automation*, vol. 13, no. 1, pp. 33–49, 2022.

[176] E. Stevenson and M. Johnson, "Devops and automation for continuous delivery: Tools and techniques," *IEEE Software*, vol. 36, no. 4, pp. 23–31, 2019.

[177] OpenJS Foundation, "Express - node.js web application framework," 2020. [Online]. Available: https://expressjs.com/

[178] The PostgreSQL Global Development Group, "Postgresql documentation," 2020. [Online]. Available: https://www.postgresql.org/docs/

[179] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning.* MIT Press, 2016.

[180] T. B. Brown *et al.*, "Language models are few-shot learners," *Advances in Neural Information Processing Systems*, vol. 33, pp. 1877–1901, 2020.

[181] M. Hajnas and A. Alqurashi, "Waste classification using deep learning for waste management in smart cities," *IEEE Access*, vol. 8, pp. 17 506–17 518, 2020.

[182] M.-T. Truong *et al.*, "Ai-driven waste management systems: A comprehensive review," *Journal of Cleaner Production*, vol. 250, p. 11952, 2022.

[183] A. Marin *et al.*, "Predicting urban waste generation with machine learning," *Waste Management*, vol. 116, pp. 170–182, 2021.

[184] C. de la República de Guatemala, "Ley de protección de datos personales (decreto 25-2012)," 2012. [Online]. Available: https://www.congreso.gob.gt/

[185] AGN, "Guatemala en el primer foro de Ética de la inteligencia artificial," 2023. [Online]. Available: https://agn.gt/guatemala-en-el-primer-foro-de-etica-de-la-inteligencia-artificial/

[186] U. Europea, "Reglamento de inteligencia artificial de la ue," 2021. [Online]. Available: https://eur-lex.europa.eu/legal-content/ES/TXT/?uri=CELEX%3A52021PC0206

[187] U. Government, "Guidelines for the regulation of artificial intelligence," 2021. [Online]. Available: https://www.whitehouse.gov/ai

[188] S. Russell, *Human Compatible: Artificial Intelligence and the Problem of Control.* Viking, 2019.

[189] GSMA, "ewaste en américa latina," 2015. [Online]. Available: https://www.gsma.com/latinamerica/wp-content/uploads/2015/11/gsma-unu-ewaste2015-spa.pdf

[190] DataScienceCentral, "Bias variance trade off," 2019. [Online]. Available: https://www.datasciencecentral.com/bias-variance-trade-off/

[191] U. de Valladolid, "Estudio de algoritmos de redes neuronales convolucionales en," 2021. [Online]. Available: https://uvadoc.uva.es/bitstream/10324/47137/1/TFG-G4811.pdf

[192] DataScienceCentral, "Bias variance trade off - datasciencecentral.com," 2019. [Online]. Available: https://www.datasciencecentral.com/bias-variance-trade-off/

[193] Medium, "Intro to deep learning — bias-variance tradeoff, regularizations," 2021. [Online]. Available: https://hk3342.medium.com/intro-to-deep-learning-bias-variance-tradeoff-regularizations-f0e7b7b4a4f2

[194] H. CS182, "Bias, variance, and regularization - cs 182," 2021. [Online]. Available: https://cs182sp21.github.io/static/slides/lec-3.pdf

[195] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 248–255, 2009.

[196] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson, "How transferable are features in deep neural networks?" in *Advances in Neural Information Processing Systems*, 2014, pp. 3320–3328.

[197] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Transactions on Knowledge and Data Engineering*, vol. 22, no. 10, pp. 1345–1359, 2009.

[198] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2017, pp. 1251–1258.

[199] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "Mobilenetv2: Inverted residuals and linear bottlenecks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 4510–4520.

[200] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 2818–2826.

[201] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[202] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 4700–4708.

[203] A. Adadi and M. Berrada, "Peeking inside the black-box: A survey on explainable artificial intelligence (xai)," *IEEE Access*, vol. 6, pp. 52 138–52 160, 2018.

[204] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, "Grad-cam: Visual explanations from deep networks via gradient-based localization," *Proceedings of the IEEE international conference on computer vision*, pp. 618–626, 2017.

[205] M. T. Ribeiro, S. Singh, and C. Guestrin, "Why should i trust you? explaining the predictions of any classifier," pp. 1135–1144, 2016.

[206] S. M. Lundberg and S.-I. Lee, "A unified approach to interpreting model predictions," *Advances in Neural Information Processing Systems*, vol. 30, pp. 4765–4774, 2017.

[207] D. Smilkov, N. Thorat, B. Kim, F. Viégas, and M. Wattenberg, "Smoothgrad: removing noise by adding noise," *arXiv preprint arXiv:1706.03825*, 2017.

[208] D. Gunning and D. Aha, "Darpa's explainable artificial intelligence (xai) program," *AI Magazine*, vol. 40, no. 2, pp. 44–58, 2019.

[209] J. Fandiño, "Desarrollo de una aplicación web para la clasificación de residuos sólidos a través de un modelo de machine learning," 2022. [Online]. Available: https://repository.universidadean.edu.co/bitstream/handle/10882/12381/FandinoJuan2022.pdf?sequence=1&isAllowed=y

[210] A. Marín, "Analítica para una gestión de residuos inteligente," 2020. [Online]. Available: https://blogs.sas.com/content/hiddeninsights/2020/07/13/modelos-analiticos-para-dotar-de-inteligencia-a-la-gestion-de-residuos/

[211] DoradoReciclaAI, "Sistema para clasificación de residuos: Project recicla ai," 2023. [Online]. Available: https://github.com/DoradoReciclaAI/Project_recicla_ai

[212] M. Cruz and L. Vega, "Wastenet: A smart waste management system for smart cities," *IEEE Access*, vol. 6, pp. 14 594–14 605, 2018.

[213] L. González, "Ecosort: Optimización de rutas de recolección de residuos usando machine learning," 2019. [Online]. Available: https://repositorio.uam.es/handle/10486/684402

[214] J. Smith, "Ai-waste: Advanced e-waste classification using deep learning," 2020. [Online]. Available: https://arxiv.org/abs/2003.12356

[215] S. Imaging, "Ventajas del machine learning y deep learning para la visión por computadora," 2024. [Online]. Available: https://es.stemmer-imaging.com/blog/tecnologia-tendencias/machine-learning-deep-learning/

[216] A. Torres Rodríguez, "Desarrollo e implementación de clasificadores de imágenes mediante el empleo de algoritmos de aprendizaje por transferencia," 2024. [Online]. Available: https://www.researchgate.net/publication/375863525

[217] CS-UCSP, "Diplomado en visión por computador con machine learning y deep learning," 2024. [Online]. Available: https://cs.ucsp.edu.pe/postgrado/diplomado-vision-computador-con-machine-learning-deep-learning/

[218] S. Institute, "Computer vision: Qué es y por qué es importante," 2024. [Online]. Available: https://www.sas.com/es_mx/insights/analytics/computer-vision.html

[219] InboundCycle, "Qué es el deep learning, cómo funciona y sus aplicaciones," 2024. [Online]. Available: https://www.inboundcycle.com/blog-de-inbound-marketing/deep-learning-que-es

[220] Medium, "Modelos de chatgpt y llm (modelos de lenguaje grande)," 2023. [Online]. Available: https://medium.com/chatgpt-learning/modelos-de-chatgpt-y-llm-modelos-de-lenguaje-grande-d8c5fc39f59e

[221] H. P. E. (HPE), "¿qué son los grandes modelos de lenguaje?" 2023. [Online]. Available: https://www.hpe.com/lamerica/es/what-is/large-language-model.html

[222] Kaggle, "Explora y domina kaggle para machine learning," 2023. [Online]. Available: https://www.kaggle.com/learn/overview

[223] Jupyter, "Jupyter notebooks para ciencia de datos," 2023. [Online]. Available: https://jupyter.org/

[224] Google, "Recursos y ventajas de google colab pro," 2023. [Online]. Available: https://colab.research.google.com/signup

[225] H. Face, "Implementación de modelos nlp con hugging face," 2023. [Online]. Available: https://huggingface.co/transformers/

[226] TensorFlow, "Tensorflow: Guía completa," 2023. [Online]. Available: https://www.tensorflow.org/

[227] PyTorch, "Tutorial de pytorch," 2023. [Online]. Available: https://pytorch.org/tutorials/

[228] OpenCV, "Opencv para visión por computadora," 2023. [Online]. Available: https://opencv.org/

[229] Keras, "Keras: La api de alto nivel de tensorflow," 2023. [Online]. Available: https://keras.io/

[230] Scikit-Image, "Scikit-image: Procesamiento avanzado de imágenes," 2023. [Online]. Available: https://scikit-image.org/

[231] A. Clark and contributors, "Pillow: Python imaging library (fork)," https://python-pillow.org/, 2015, available at: https://python-pillow.org/.

[232] NVIDIA, "Cuda y el aceleramiento de deep learning," 2023. [Online]. Available: https://developer.nvidia.com/cuda-zone

[233] TrashNet, "Trashnet: Dataset de clasificación de residuos," 2023. [Online]. Available: https://github.com/garythung/trashnet

[234] W. C. Data, "Waste classification data: Dataset de residuos," 2023. [Online]. Available: https://www.kaggle.com/datasets/alexattia/the-simpsons-characters-dataset

[235] TACO, "Taco: Dataset de clasificación y detección de residuos," 2023. [Online]. Available: https://www.kaggle.com/datasets/ahmedetefy/taco-trash-detection

[236] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Kaiser, and I. Polosukhin, "Attention is all you need," 2017. [Online]. Available: https://arxiv.org/abs/1706.03762

[237] S. Ramirez, "Fastapi: The high-performance web framework for building apis with python 3.7+," 2021, available at: https://fastapi.tiangolo.com/.

[238] A. Ronacher, *Jinja2 Documentation (2.10)*, 2010, available at: https://jinja.palletsprojects.com/.

[239] J. Author, "A primer on mlops: Machine learning operations," *Tech Journal of AI*, 2021.

[240] M. Zaharia, A. Chen, A. Davidson, A. Ghodsi, S. Hong, A. Konwinski, S. Murching, M. Nykodym, P. Ogilvie, S. A. Parkhe *et al.*, "Accelerating the machine learning lifecycle with mlflow," in *IEEE Data Engineering Bulletin*, vol. 41, no. 4, 2018, pp. 39–45.

[241] D. Sculley, G. Holt, D. Golovin, E. Davydov, T. Phillips, D. Ebner, V. Chaudhary, M. Young, J. Crespo, and D. Dennison, "Hidden technical debt in machine learning systems," *Advances in Neural Information Processing Systems*, 2015.

[242] E. Breck, S. Cai, M. Nielsen, M. Salib, and D. Sculley, "The ml test score: A rubric for ml production readiness and technical debt reduction," in *Proceedings of the 2017 Conference on Systems for ML*, 2017.

[243] N. Polyzotis, S. Roy, S. Whang, and M. Zinkevich, "Data management challenges in production machine learning: A survey," *ACM Data Engineering Bulletin*, 2018.

[244] Databricks, "Mlflow documentation," https://mlflow.org/docs, 2022.

[245] L. G. y Katya Seckar y Rebekah Church, "Manejo de desechos," 2006. [Online]. Available: https://www.mcgill.ca/pfss/files/pfss/GenericWasteGuide2006.pdf

[246] Ministerio de Ambiente y Recursos Naturales, "Guatemala tendrá una gestión integral de los residuos y desechos sólidos con el nuevo reglamento del marn," August 13 2021. [Online]. Available: https://guatemala.gob.gt/guatemala-tendra-una-gestion-integral-de-los-residuos-y-desechos-solidos-con-el-nuevo-reglamento-del-marn/

[247] ——, "Dirección para el manejo de residuos y desechos sólidos," March 10 2023. [Online]. Available: https://www.marn.gob.gt/viceministro-de-recursos-naturales-y-cambio-climatico/los-residuos-y-desechos-solidos/direccion-para-el-manejo-de-residuos-y-desechos-solidos/

[248] J. Rodríguez, "Análisis de la generación de desechos sólidos en guatemala," 2023. [Online]. Available: https://ipn.usac.edu.gt/wp-content/uploads/2023/09/Analisis-de-la-generacion-de-desechos-solidos-en-Guatemala.pdf

[249] ClimateTrade, "Mejores consejos para la reducción de carbono," March 2 2023. [Online]. Available: https://climatetrade.com/es/los-10-mejores-consejos-de-reduccion-de-carbono-para-empresas-y-particulares/

[250] Deloitte Spain, "La nube, una tecnología verde y sostenible," 2022. [Online]. Available: https://www2.deloitte.com/es/es/blog/sostenibilidad-deloitte/2022/la-nube-una-tecnologia-verde-y-sostenible.html

[251] Amazon, "Principios de diseño generales - aws well-architected framework," 2024. [Online]. Available: https://docs.aws.amazon.com/es_es/wellarchitected/latest/framework/general-design-principles.html

[252] The Kubernetes Authors, "Orquestación de contenedores para producción," 2024. [Online]. Available: https://kubernetes.io/es/

[253] Google Cloud, "¿qué es la arquitectura de nube? beneficios y componentes," 2024. [Online]. Available: https://cloud.google.com/learn/what-is-cloud-architecture?hl=es-419

[254] L. Bass, I. Weber, and L. Zhu, *DevOps: A Software Architect's Perspective*. Addison-Wesley Professional, 2015.

[255] J. Humble and D. Farley, *Continuous Delivery: Reliable Software Releases through Build, Test, and Deployment Automation*. Addison-Wesley Professional, 2010.

[256] C. Richardson, *Microservices Patterns: With examples in Java*. Manning Publications, 2018.

[257] M. T. Wilder, *Release It!: Design and Deploy Production-Ready Software*. Pragmatic Bookshelf, 2012.

[258] K. Morris, *Infrastructure as Code: Managing Servers in the Cloud*. O'Reilly Media, Inc., 2016.

[259] J. Allspaw and J. Robbins, *Web Operations: Keeping the Data on Time*. O'Reilly Media, Inc., 2010.

[260] iNBest, "Servicio de migración de infraestructura a la nube de aws," 2016. [Online]. Available: https://www.inbest.cloud/servicio-de-migracion-de-infraestructura-a-la-nube-aws-0

[261] U. Government, "Sustainability in cloud computing: Environmental and economic benefits," UK Government Digital Service, Tech. Rep., 2020, available at https://www.gov.uk/government/publications/sustainability-in-cloud-computing.

[262] A. W. Services, *AWS Customer Carbon Footprint Tool*, 2021, available at https://aws.amazon.com/sustainability/carbon-footprint-tool/.

[263] J. Wilson and K. Wright, "Artificial intelligence and data analytics for sustainable cloud operations," *Journal of Cloud Computing*, vol. 12, pp. 321–335, 2020.

[264] J. R. Gibbs, *Cloud FinOps: Collaborative, Real-Time Cloud Financial Management*. O'Reilly Media, Inc., 2020.

[265] I. Foster and C. Kesselman, "Cloud computing and sustainability: A new operational paradigm," *International Journal of Cloud Computing*, vol. 15, pp. 230–245, 2018.

[266] P. Ralph and D. Lucas, "The impact of cloud-based development on organizational sustainability," *Sustainable IT Journal*, vol. 5, pp. 78–95, 2021.

[267] W. Wright, *Cloud Architecture Patterns: Using Microsoft Azure*. O'Reilly Media, Inc., 2019.

[268] M. Loukides, "Choosing single-provider cloud strategies: Benefits and challenges," *Journal of Cloud Computing*, vol. 15, pp. 201–215, 2020.

[269] K. Hashizume and D. Rosado, "Multi-cloud storage and cost efficiency," *Journal of Cloud Security*, vol. 11, pp. 40–58, 2013.

[270] A. Stuckmann, *Vendor Lock-in and Cloud Portability: A Comprehensive Guide*. Addison-Wesley, 2018.

[271] D. Bernstein and D. Vij, "Multicloud deployment strategies and best practices," *Journal of Cloud Computing*, vol. 7, pp. 100–120, 2014.

[272] C. Moreno and M. Alvarez, "Cloud management for multicloud deployments: A resilient approach," *International Journal of Cloud Computing*, vol. 9, pp. 75–90, 2020.

[273] R. Marshall, "Cost and complexity in multicloud management," *Cloud Economics Journal*, vol. 3, pp. 65–80, 2020.

[274] S. Strickland and J. Lee, "Choosing the right cloud strategy for business resilience and flexibility," *Enterprise Cloud Journal*, vol. 6, pp. 15–30, 2021.

[275] P. Gillin, "Understanding cloud storage pricing models for better decision-making," *Journal of Cloud Economics*, vol. 10, pp. 115–130, 2023.

[276] H. Demirkan, "Key factors in enterprise cloud storage costs: An analytical overview," *Cloud Computing Journal*, vol. 15, pp. 55–75, 2022.

[277] J. Smith, "Data transfer and egress costs in cloud environments," *Journal of Cloud Networking*, vol. 7, pp. 45–60, 2023.

[278] Veritas Technologies, "The definitive guide to cloud storage pricing," 2024. [Online]. Available: https://www.veritas.com/es/mx/information-center/the-definitive-guide-to-cloud-storage-pricing

[279] R. Johnson, "Tiered storage strategies for cost optimization in the cloud," *International Journal of Cloud Optimization*, vol. 12, pp. 85–100, 2023.

[280] F. Foundation, *FinOps Handbook: Cloud Cost Optimization for the Modern Enterprise*. O'Reilly Media, Inc., 2023.

[281] F. Stojanović, "Market analysis of major cloud providers in 2023," *Cloud Computing Review*, vol. 8, pp. 150–170, 2023.

[282] G. Research, *2023 Cloud Infrastructure Market Share Report*, 2023, available at https://www.gartner.com/en/cloud-market-share-2023.

[283] VMware, *VMware's Approach to Private Cloud Security and Customization*, 2022, available at https://www.vmware.com/private-cloud-overview.

[284] P. Mell and T. Grance, "The nist definition of cloud computing," National Institute of Standards and Technology (NIST), Tech. Rep., 2020, available at https://nvlpubs.nist.gov/nistpubs/legacy/sp/nistspecialpublication800-145.pdf.

[285] A. W. Services, *AWS Security and Compliance Best Practices*, 2023, available at https://aws.amazon.com/security/.

[286] A. D. Rayome, "Hybrid cloud: The future of enterprise data management," *Enterprise Technology Journal*, vol. 12, pp. 300–320, 2021.

[287] J. Miller, "Aws: An overview of services and market position," *Cloud Computing Review*, vol. 10, pp. 20–45, 2023.

[288] A. W. Services, *AWS Pricing Overview and Cost Management*, 2023, available at https://aws.amazon.com/pricing/.

[289] M. Corporation, *Microsoft Azure Cloud Services and Hybrid Solutions*, 2023, available at https://azure.microsoft.com/overview.

[290] G. Cloud, *Google Cloud Platform and Machine Learning Solutions*, 2023, available at https://cloud.google.com/solutions/ai-machine-learning.

[291] O. Corporation, *Oracle Cloud for Enterprise Applications and Performance*, 2023, available at https://www.oracle.com/cloud/enterprise/.

[292] I. Corporation, *IBM Cloud Security and Risk Management Solutions*, 2023, available at https://www.ibm.com/cloud/security.

[293] G. Research, *2023 Cloud Market Share and Trends Analysis*, 2023, available at https://www.gartner.com/en/cloud-market-report-2023.

[294] W. Wright, *Cloud Architecture Patterns: Using Microsoft Azure and Other Platforms*. O'Reilly Media, Inc., 2019.

[295] D. Bernstein and D. Vij, "Challenges and strategies in multi-cloud architectures," *Journal of Cloud Computing*, vol. 7, pp. 120–140, 2014.

[296] C. Moreno and M. Alvarez, "Ensuring interoperability in cloud architectures across multiple platforms," *International Journal of Cloud Systems*, vol. 10, pp. 95–110, 2021.

[297] F. Flores, "Cloud computing: Tipos de nubes, servicios y proveedores," March 22 2021. [Online]. Available: https://openwebinars.net/blog/tipos-de-cloud-computing/

[298] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016.

[299] J. Smith and V. Rao, "Data collection and processing for ai: Best practices," *Journal of Data Science*, vol. 15, pp. 120–145, 2022.

[300] S. Olson, "Green technology and environmental sustainability," *Sustainable Technology Journal*, vol. 8, pp. 30–50, 2021.

[301] P. Del Rio, "The impact of cleantech solutions on climate change," *Journal of Environmental Technology*, vol. 22, pp. 60–80, 2019.

[302] G. E. Department, *Innovations in Renewable Energy Technology*, 2023, available at https://www.energy.gov/renewable-technology.

[303] ISO, *Environmental Management Systems – Requirements with Guidance for Use*, 2015, iSO 14001:2015 Standard, International Organization for Standardization.

[304] E. Commission, *EMAS - The Eco-Management and Audit Scheme Guide*, 2019, available at https://ec.europa.eu/environment/emas/.

[305] U. G. B. Council, *LEED Certification: An Overview*, 2020, available at https://www.usgbc.org/leed.

[306] M. . Company, "The role of digitalization in waste management," *Sustainable Technology Insights*, vol. 9, pp. 45–60, 2021.

[307] G. W. M. Institute, *Waste Management Systems for the Digital Era*, 2020, available at https://www.gwmi.org/waste-management-systems.

[308] E. Rodriguez and L. Marquez, "Using big data for waste optimization in industrial settings," *Journal of Environmental Data Science*, vol. 5, pp. 150–170, 2019.

[309] UNESCO, *Environmental Education for Sustainability*, 2020, available at https://unesdoc.unesco.org/environmental-education.

[310] ——, *Environmental Education and Waste Management*, 2020, available at https://unesdoc.unesco.org/waste-management-education.

[311] J. Smith and R. Kumar, "Corporate sustainability programs for waste management," *Journal of Environmental Management*, vol. 12, pp. 55–70, 2019.

[312] E. P. Agency, *Campaigns for Environmental Awareness and Impact Reduction*, 2021, available at https://www.epa.gov/environmental-awareness.

[313] T. Gower and A. Ray, "Closing the loop: Circular economy principles and practices," *Circular Economy Journal*, vol. 7, pp. 120–145, 2020.

[314] M. . Company, "Technology-driven circular economy solutions," *Sustainable Innovation Journal*, vol. 15, pp. 100–115, 2021.

[315] C. Systems, *IoT Solutions for Waste Management*, 2020, available at https://www.cisco.com/iot-waste-management.

[316] M. . Company, "Smart waste management with iot and real-time data," *Journal of Sustainable Technologies*, vol. 12, pp. 130–145, 2021.

[317] R. J. E. Team, "Automation and robotics in recycling: Current applications and future trends," *International Journal of Recycling Innovations*, vol. 8, pp. 80–100, 2020.

[318] C. Dupont and M. Ramirez, "The role of automation in reducing waste in industrial processes," *Journal of Industrial Ecology*, vol. 15, pp. 200–215, 2019.

[319] U. E. Programme, *Carbon Reduction Initiatives: Global Success Stories*, 2021, available at https://www.unep.org/carbon-reduction.

[320] U. N. F. C. on Climate Change, *Paris Agreement*, 2015, available at https://unfccc.int/process-and-meetings/the-paris-agreement.

[321] Amazon, *Amazon's Net Zero Carbon by 2040 Pledge*, 2022, available at https://sustainability.aboutamazon.com/net-zero-pledge.

[322] T. Inc., *Tesla Impact Report 2020: Sustainability and Innovation*, 2020, available at https://www.tesla.com/impact-report.

[323] Unilever, *Sustainable Supply Chain: Unilever's Approach*, 2021, available at https://www.unilever.com/sustainable-supply-chain.

[324] G. Research, *Cloud Pricing and Licensing Models: 2023 Report*, 2023, available at https://www.gartner.com/cloud-pricing-2023.

[325] A. W. Services, *Bring Your Own License (BYOL) on AWS*, 2023, available at https://aws.amazon.com/byol.

[326] M. Azure, *Azure Hybrid Benefit for Cost Optimization*, 2023, available at https://azure.microsoft.com/hybrid-benefit.

[327] M. Corporation, *Microsoft Azure Licensing Options*, 2022, available at https://docs.microsoft.com/azure/licensing.

[328] G. C. Platform, *Google Cloud Pricing Overview and Sustained Use Discounts*, 2023, available at https://cloud.google.com/pricing.

[329] G. Cloud, *Open Source Solutions on Google Cloud*, 2022, available at https://cloud.google.com/open-source.

[330] F. Research, *Cloud Cost Comparison: AWS, Azure, and GCP in 2023*, 2023, available at https://www.forrester.com/cloud-cost-comparison-2023.

[331] G. Research, *SaaS Licensing and Subscription Models: 2022 Guide*, 2022, available at https://www.gartner.com/saas-licensing-2022.

[332] L. Forrester, "The advantages of saas licensing models in modern business," *Journal of Cloud Computing*, vol. 10, pp. 55–72, 2021.

[333] G. Research, *PaaS Market Guide: Best Solutions and Providers*, 2023, available at https://www.gartner.com/paas-market-guide.

[334] F. Research, *SaaS Solutions and Market Trends 2022*, 2022, available at https://www.forrester.com/saas-trends-2022.

[335] I. Research, "Customer identity and access management solutions," *Journal of Digital Identity*, vol. 14, pp. 80–95, 2021.

[336] Cloudflare, *Security as a Service: Cloudflare Solutions*, 2023, available at https://www.cloudflare.com/secaas.

[337] A. W. Services, *AWS Lambda and Serverless Applications*, 2023, available at https://aws.amazon.com/lambda.

[338] P. Identity, *Ping Identity IDaaS: Identity and Access Management Solutions*, 2023, available at https://www.pingidentity.com/idaas.

[339] Z. V. Communications, *Zoom Communication as a Service Solutions*, 2023, available at https://zoom.us/caas.

[340] G. C. Platform, *Google Cloud Storage Solutions*, 2023, available at https://cloud.google.com/storage.

[341] G. Research, *Cloud Management and Security Considerations 2023*, 2023, available at https://www.gartner.com/cloud-management-2023.

[342] A. W. Services, *AWS CloudTrail and Compliance Solutions*, 2023, available at https://aws.amazon.com/cloudtrail.

[343] G. Cloud, *BigQuery and Machine Learning in Google Cloud*, 2023, available at https://cloud.google.com/bigquery.

[344] M. Azure, *Azure Site Recovery and Disaster Management Solutions*, 2023, available at https://azure.microsoft.com/recovery-services.

[345] I. Research, "Trends in identity and access management for cloud environments," *Journal of Identity Management*, vol. 18, pp. 50–67, 2021.

[346] N. I. of Standards and T. (NIST), *Role-Based Access Control (RBAC) Standards*, 2021, available at https://www.nist.gov/rbac-standards.

[347] P. Mell and T. Grance, "Attribute-based access control (abac) in cloud systems," *Journal of Cybersecurity*, vol. 22, pp. 85–100, 2021.

[348] T. Grance, *Policy-Based Access Control (PBAC): Standards and Applications*, 2020, available at https://www.nist.gov/pbac-standards.

[349] F. Research, *Federated Identity and Access Management in Multi-Cloud Environments*, 2022, available at https://www.forrester.com/federated-iam-2022.

[350] A. W. Services, *AWS Identity and Access Management (IAM) Overview*, 2023, available at https://aws.amazon.com/iam.

[351] ——, *AWS Organizations and Multi-Account Management*, 2023, available at https://aws.amazon.com/organizations.

[352] ——, *AWS Single Sign-On (SSO) Solutions*, 2023, available at https://aws.amazon.com/single-sign-on.

[353] G. C. Platform, *Google Cloud Identity and Access Management (IAM) Solutions*, 2023, available at https://cloud.google.com/iam.

[354] ——, *BeyondCorp Enterprise: Zero Trust Security for Google Cloud*, 2023, available at https://cloud.google.com/beyondcorp.

[355] ——, *Identity-Aware Proxy (IAP) for Secure Access*, 2023, available at https://cloud.google.com/iap.

[356] M. Azure, *Azure Active Directory Overview*, 2023, available at https://azure.microsoft.com/active-directory.

[357] ——, *Role-Based Access Control (RBAC) in Azure*, 2023, available at https://docs.microsoft.com/azure/role-based-access-control.

[358] ——, *Azure Conditional Access Policies: Dynamic Access Control*, 2023, available at https://docs.microsoft.com/azure/conditional-access.

[359] F. Research, "Advantages of cloud development vs on-premise solutions," *Cloud Computing Insights*, vol. 11, pp. 22–40, 2022.

[360] G. Research, *Global Access and Collaboration in Cloud Environments*, 2023, available at https://www.gartner.com/cloud-global-access.

[361] I. Research, "Ci/cd tools and integration in cloud platforms," *Journal of DevOps*, vol. 8, pp. 90–110, 2021.

[362] M. . Company, "The ecological impact of cloud computing," *Sustainability and Technology Journal*, vol. 9, pp. 35–50, 2023.

[363] Greenpeace, *E-Waste Reduction through Cloud Infrastructure*, 2022, available at https://www.greenpeace.org/ewaste-reduction-cloud.

[364] U. Colombia, "¿qué es la ciberseguridad? la importancia de la ciberseguridad," 2024. [Online]. Available: https://colombia.unir.net/actualidad-unir/que-es-ciberseguridad/#:~:text=La%20importancia%20de%20la%20ciberseguridad&text=La%20ciberseguridad%20no%20solo%20salvaguarda,la%20evoluci%C3%B3n%20de%20las%20amenazas

[365] U. de Cataluña, "Seguridad informática: la importancia y lo que debe saber," 2024. [Online]. Available: https://www.ucatalunya.edu.co/blog/seguridad-informatica-la-importancia-y-lo-que-debe-saber

[366] O. Foundation, "Owasp top 10 web application security risks," 2024. [Online]. Available: https://owasp.org/www-project-top-ten/

[367] T. Security, "Owasp top 10: Riesgos en aplicaciones móviles," 2024. [Online]. Available: https://www.tarlogic.com/es/blog/owasp-top-10-riesgos-aplicaciones-moviles/

[368] C. F. Bernal, "GuÍa de buenas prÁcticas de seguridad en el desarrollo de software con base en estÁndares reconocidos en empresas de desarrollo de software," 2024. [Online]. Available: https://repository.unad.edu.co/jspui/bitstream/10596/20449/1/1088303409.pdf

[369] C. Blog, "What is clean code? a guide to principles and best practices," 2024. [Online]. Available: https://blog.codacy.com/what-is-clean-code

[370] G. Pineda, "Conflictos en la gestión de los residuos y desechos sólidos: impactos económicos, ambientales y sociales de prácticas e implementación de políticas no consensuadas en guatemala," 2023. [Online]. Available: https://capitalisthistory.com/2023/08/07/conflictos-en-la-gestion-de-los-residuos-y-desechos-solidos-impactos-economicos-ambientales-y-sociales-de-pra

[371] M. de Ambiente y Recursos Naturales (MARN), "Reglamento para la gestión integral de los residuos y desechos sólidos comunes (acuerdo gubernativo 164-2021)," 2024. [Online]. Available: https://www.marn.gob.gt/reglamento-164-2021/

[372] P. Libre, "Separación de desechos en guatemala: así debe clasificar su basura con los cambios al reglamento y lo que será obligatorio en 2025," 2024. [Online]. Available: https://www.prensalibre.com/guatemala/comunitario/separacion-de-desechos-en-guatemala-asi-debe-clasificar-su-basura-con-los-cambios-al-reglamento-y-lo-que-se

[373] C. I. de Derechos Humanos (Corte IDH), "Ley de protección y mejoramiento del medio ambiente," 2024. [Online]. Available: https://www.corteidh.or.cr/tablas/20491a.pdf

[374] M. de Ambiente y Recursos Naturales (MARN), "Políticas ambientales y acuerdo gubernativo 189-2017," 2024. [Online]. Available: https://www.marn.gob.gt/wp-admin/admin-ajax.php?juwpfisadmin=false&action=wpfd&task=file.download&wpfd_category_id=96&wpfd_file_id=8524&token=&preview=1

[375] S. Seguras, "¿cuáles son los principales desafíos de ciberseguridad en guatemala en este 2022?" 2022. [Online]. Available: https://www.solucionesseguras.com/noticias/noticias-y-eventos/cuales-son-los-principales-desafios-de-ciberseguridad-en-guatemala-en-este-2022

[376] I. S. Guatemala, "Situación de guatemala en materia de seguridad cibernética," 2021. [Online]. Available: https://www.isoc.org.gt/wp-content/uploads/2021/11/Situacion-de-Guatemala-en-materia-de-Seguridad-Cibernetica.pdf

[377] K. R. López, "Ley contra ciberdelincuencia, la normativa que puso en riesgo la libertad de expresión y la crítica ciudadana," 2022. [Online]. Available: https://www.plazapublica.com.gt/content/ley-contra-ciberdelincuencia-la-normativa-que-puso-en-riesgo-la-libertad-de-expresion-y-la

[378] A. Caro, "La importancia del desarrollo seguro de software en la era digital," 2023. [Online]. Available: https://web.fdi.ucm.es/posgrado/conferencias/AndresCaroLindo-slides.pdf

[379] M. Micucci, "Metodologías de desarrollo seguro de software e integración de aplicaciones," 2023. [Online]. Available: https://www.welivesecurity.com/es/recursos-herramientas/metodologias-desarrollo-seguro-software-integracion-aplicaciones/

[380] A. W. S. (AWS), "¿qué es el ciclo de vida del desarrollo de software (sdlc)?" 2024. [Online]. Available: https://aws.amazon.com/es/what-is/sdlc/

[381] J. J. S. Chávez, "4 buenas prácticas para un desarrollo seguro de software: ¿por qué es importante?" 2024, publicado el 9 de febrero de 2024, actualizado el 25 de abril de 2024. Último acceso: 15 de octubre de 2024. [Online]. Available: https://www.deltaprotect.com/blog/desarrollo-seguro-de-software

[382] R. Hat, "Software development lifecycle security," 2024. [Online]. Available: https://www.redhat.com/es/topics/security/software-development-lifecycle-security#sdlc-devops-y-el-enfoque-%C3%A1gil

[383] S. Electro, "Principios de un diseño seguro," 2011. [Online]. Available: https://serzaelectro.wordpress.com/2011/03/10/principios-de-un-diseno-seguro/

[384] Laotshi, "Secure by design: Fundamentos del diseño seguro," 2020. [Online]. Available: https://laotshi.medium.com/secure-by-design-fundamentos-b4d25ed3a9ac

[385] Microsoft, "Principios de seguridad en el marco microsoft well-architected," 2021. [Online]. Available: https://learn.microsoft.com/es-es/azure/well-architected/security/principles

[386] O. Foundation, "Owasp software assurance maturity model (samm)," 2024. [Online]. Available: https://owaspsamm.org/

[387] T. Security, "Owasp samm: Cómo mejorar la seguridad del software empresarial," 2024. [Online]. Available: https://www.tarlogic.com/es/blog/owasp-samm-mejorar-seguridad-software-empresarial/

[388] I. Formación, "¿qué es sonarqube?" 2024, Último acceso: 15 de octubre de 2024. [Online]. Available: https://imaginaformacion.com/tutoriales/que-es-sonarqube

[389] R. Gupta, "Sonarqube security features," 2024. [Online]. Available: https://media.licdn.com/dms/document/media/D4D1FAQHlnis2swS0zg/feedshare-document-pdf-analyzed/0/1729849905729?e=1730937600&v=beta&t=BV9POAtmIfhLi6gMVyamdLwVKoAZSH13Vw6eCHl3Z_4

[390] Sonar. (2022) Sonarqube 10.7 documentation. [Online]. Available: https://docs.sonarsource.com/sonarqube/latest/

[391] I. Consulting. (2022) Art. 5 gdpr - principles relating to processing of personal data. [Online]. Available: https://gdpr-info.eu/art-5-gdpr/

[392] ——. (2022) Art. 6 gdpr - lawfulness of processing. [Online]. Available: https://gdpr-info.eu/art-6-gdpr/

[393] H. I. Hernández. (2021) Qué es el gdpr y cómo afecta a latinoamérica. [Online]. Available: https://news.registro.gt/2021/09/27/que-es-el-gdpr-y-como-afecta-a-latinoamerica/#:~:text=Guatemala%20carece%20de%20una%20normativa,datos%20personales%20y%20datos%20sensibles.

[394] CityAdapt, "Guatemala - cityadapt," October 2024, accessed: 2024-10-10. [Online]. Available: https://cityadapt.com/country/guatemala/

[395] D. Ragan, "Why figma is a better choice for design collaboration," *UX Design Magazine*, pp. 1–3, 2020.

[396] A. Singh and J. Smith, "Comparison of prototyping tools for ux designers," *Journal of Interactive Design*, pp. 45–53, 2021.

[397] S. G. Stats, "Mobile operating system market share guatemala," n.d., accessed: 2024-10-10. [Online]. Available: https://gs.statcounter.com/os-market-share/mobile/guatemala

[398] CityAdapt, "Habitantes ciudad de guatemala," https://cityadapt.com/ciudad/ciudad-de-guatemala/#:~:text=Habitantes%20Ciudad%20de%20Guatemala%3A%20La,2022%20es%20de%201%2C213%2C651%20habitantes, 2022, estimación de población en 2022 para la Ciudad de Guatemala. [Online]. Available: https://cityadapt.com/ciudad/ciudad-de-guatemala/#:~:text=Habitantes%20Ciudad%20de%20Guatemala%3A%20La,2022%20es%20de%201%2C213%2C651%20habitantes

[399] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, M. Kudlur, J. Levenberg, R. Monga, S. Moore, D. G. Murray, B. Steiner, P. Tucker, V. Vasudevan, P. Warden, M. Wicke, Y. Yu, and X. Zheng, *TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems.* Google Research, 2016. [Online]. Available: https://www.tensorflow.org/

[400] A. Clark and Contributors, "Pillow: The friendly PIL fork," *Python Imaging Library (PIL)*, 2021, available at: https://pillow.readthedocs.io/.

[401] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. M. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei, "Gpt-3: Language models are few-shot learners," *OpenAI Research*, 2023. [Online]. Available: https://www.openai.com/research/gpt-3

[402] S. Ramirez, "Fastapi: The high performance web framework for building apis with python 3.6+ based on standard python type hints," *FastAPI Documentation*, 2020, available at: https://fastapi.tiangolo.com/.

[403] Q. Theng, "Garbage classification - 6 classes (775/class)," 2023. [Online]. Available: https://www.kaggle.com/datasets/quangtheng/garbage-classification-6-classes-775class

[404] M. I. B. Ahmed, R. B. Alotaibi, R. A. Al-Qahtani, R. S. Al-Qahtani, S. S. Al-Hetela, K. A. Al-Matar, N. K. Al-Saqer, A. Rahman, L. Saraireh, M. Youldash, and G. Krishnasamy, "Deep learning for recyclable product classification: Enhancing efficiency and sustainability in waste management," *Journal of Cleaner Production*, 2023. [Online]. Available: https://www.semanticscholar.org/paper/c19b7f207a9462aae2658d9875d5149273d74cc5

[405] Z. Tan, C. W. Lee, S. Y. Ong, A. J. Chua, and T. S. Koh, "Aiwaste: An ai-based approach to waste classification for efficient waste management," in *Proceedings of the 2nd International Conference on Artificial Intelligence for Industries (AI4I).* ACM, 2023, pp. 12–18.

[406] A. Y. Ng, "Feature selection, l1 vs. l2 regularization, and rotational invariance," *Proceedings of the 21st International Conference on Machine Learning*, 2004.

[407] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *Journal of Machine Learning Research*, vol. 15, pp. 1929–1958, 2014.

[408] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *International Conference on Machine Learning*, 2015, pp. 448–456.

[409] N. Tajbakhsh, J. Shin, S. R. Gurudu, R. T. Hurst, C. B. Kendall, M. B. Gotway, and J. Liang, "Convolutional neural networks for medical image analysis: Full training or fine tuning?" *IEEE Transactions on Medical Imaging*, vol. 35, no. 5, pp. 1299–1312, 2016.

[410] D. Masters and C. Luschi, "Revisiting small batch training for deep neural networks," in *arXiv preprint arXiv:1804.07612*, 2018.

[411] D. P. Kingma and J. L. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[412] J. D. Owens, M. Houston, D. Luebke, S. Green, J. E. Stone, and J. C. Phillips, "Gpu computing," *Proceedings of the IEEE*, vol. 96, no. 5, pp. 879–899, 2008.

[413] L. Prechelt, "Early stopping—but when?" *Neural Networks: Tricks of the trade*, pp. 55–69, 1998.

[414] L. N. Smith, "Cyclical learning rates for training neural networks," *2017 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pp. 464–472, 2017.

[415] P. Truong, J. Hansen, and M. Weisser, "Ai in waste management: An advanced convolutional neural network model for waste classification," *Journal of Waste Management*, pp. 33–47, 2022.

[416] Databricks, *MLflow: A Platform for the Machine Learning Lifecycle*, 2023, https://mlflow.org/.

[417] G. de Guatemala, "Acuerdo gubernativo 164-2021," *Diario de Centro América*, 2021, en línea.

[418] OpenAI, "Openai api documentation," 2020, openAI.

[419] J. Nielsen, *Usability Engineering*. Amsterdam: Morgan Kaufmann, 1994.

[420] European Youth Portal, "How to reduce my carbon footprint," https://youth.europa.eu/get-involved/sustainable-development/how-reduce-my-carbon-footprint_en, 2023, accedido: 2023-11-03. [Online]. Available: https://youth.europa.eu/get-involved/sustainable-development/how-reduce-my-carbon-footprint_en