Sistema de medición continua de contaminantes atmosféricos en Carretera a El Salvador, Centro Histórico de la capital y Mixco, para análisis estadístico-científico

Francisco Molina Jiménez, Ana Lucía Hernández Ordoñez, María Fernanda López Díaz

UVG UNIVERSIDAD DEL VALLE DE GUATEMALA

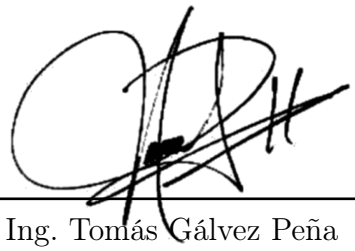
UNIVERSIDAD DEL VALLE DE GUATEMALA
Facultad de Ingeniería

Sistema de medición continua de contaminantes atmosféricos en Carretera a El Salvador, Centro Histórico de la capital y Mixco, para análisis estadístico-científico

Trabajo de graduación en modalidad de Megaproyecto presentado por
Francisco Molina Jiménez, Ana Lucía Hernández Ordoñez,
María Fernanda López Díaz
Para optar al grado académico de Licenciados en Ingeniería en Ciencias de la Computación y Tecnologías de la Información

Guatemala,
2021

Vo.Bo.:



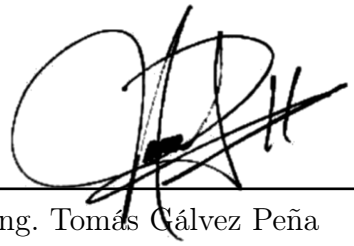
(f) _____

Ing. Tomás Gálvez Peña

Tribunal Examinador:

(f) _____

MSc. Douglas Barrios



(f) _____

Ing. Tomás Gálvez Peña

Fecha de aprobación: Guatemala, 6 de diciembre de 2021.

Prefacio

Conforme el tiempo avanza y las actividades humanas tienen un impacto ambiental. La calidad del aire decrece a medida que no se hace la transición hacia energías renovables que reduzcan los contaminantes atmosféricos. Esto tiene un impacto negativo a largo plazo en la salud de las personas, y en el cambio climático a través de la producción de los gases invernaderos.

El presente trabajo surge de la creciente necesidad de monitorear nuestro entorno ambiental del daño que hacemos y de la gran oportunidad que nos brinda la tecnología de mejorar nuestras vidas de manera sostenible. Con este proyecto se busca crear una herramienta que permita fomentar la plática sobre la contaminación en áreas urbanas en Guatemala y mejorar las decisiones que se toman al respecto. A continuación, los agradecimientos de cada uno de los integrantes del equipo de trabajo.

Yo, Francisco Molina, quiero agradecer en primer lugar a mi familia por ser mis referentes académicos, laborales y éticos a lo largo de mi vida, y por hacer posible la oportunidad de poder acceder a estudios superiores privados. Especialmente a mi madre, Ana Lucrecia, a mi padre, Francisco y a mi hermana Ana Sofía. Agradezco a mis abuelos por su incondicional apoyo, cuidado y cariño: Armila Juárez España, Benicio Molina, Blanca Gudiel y Mario Jiménez.

En segundo lugar, a las personas que contribuyeron en la realización de este trabajo, a Mercedes Pérez y a su padre José Pérez por apoyarme con su vasto conocimiento en química y medición de gases y hacer posible este trabajo de graduación a través de su conocimiento, consejos y equipo de medición. Adicionalmente quiero agradecer a Hee Chan Kim por su apoyo en el diseño 3D del contenedor que fue utilizado para este trabajo.

En tercer lugar, a mis profesores y compañeros que me han inspirado en mi trayectoria como estudiante, a Walter Cervantes, por inspirarme en mis estudios de las matemáticas y a perseguir un título como ingeniero. A mis profesores de computación Borís Paz, William Girón, Byron Bolaños, Romeo Mendoza, Dennis Aldana, Hector Hurtarte y Douglas Barrios.

En cuarto lugar, a mis amigos que me han acompañado en mi trayectoria como estudiante, principalmente a María Fernanda López, Ana Lucía Hernández, Paola Mendizabal, Odalis Reyes, Maryandré Salguero, Mayra Silva, Katarzyna Polkowska y Gabriela Przestrzelska, Sebastián Arriola, Rodrigo Zea, Hayri Yigit, Dejan Čančar, Nena Petrović y Tanja Petronijević, por haberme acompañado durante el transcurso de mi carrera y ser personas que han aportado tanto a mi vida. Finalmente, a todas las personas que han influido positivamente en mi vida y servir de inspiración en mi vida hasta este momento, especialmente a William Navarro.

Yo, Ana Lucía Hernández, agradezco a mis padres y hermanas, por darme la oportunidad de estar en este largo viaje y haberme dado su apoyo y amor incondicional siempre. A mis amigos

Francisco Molina y María Fernanda López, juntos hemos logrado cerrar esta etapa de nuestras vidas y espero compartir muchos años más con ustedes.

Yo, María Fernanda López, dedico el presente trabajo de graduación a mi madre, en muestra de todo el esfuerzo realizado por brindarme la oportunidad de estudios universitarios. Le agradezco a todas aquellas personas que se interpusieron en el camino de ella para ayudarla a llegar a donde hoy se encuentra. Sobre todo a Dios por derramar siempre bendiciones a nuestra familia y ampararnos en momentos difíciles. Le agradezco a mis compañeros de carrera sobre todo a Francisco Molina, Ana Lucía Hernández, Paul Belches y Gustavo de León por su apoyo incondicional y por los momentos compartidos. Por último, a mí novio por siempre estar pendiente de mí y apoyarme en todo momento.

Índice

Lista de figuras

Lista de cuadros

Resumen

Se desarrolló una estación de gases contaminantes, una infraestructura para el procesamiento y recopilación de datos y una plataforma web para la visualización, distribución y análisis de estos contaminantes. Los contaminantes a medir por la estación fueron SO_2, O_3, NO_2 y CO, se realizaron las mediciones en tres puntos del país, siendo estos Carretera a el Salvador, Ciudad San Cristóbal Mixco y Centro Histórico de la ciudad de Guatemala.

A través de IoT (Internet of Things, por sus siglas en inglés), esta estación permitirá compartir datos a través de protocolos como MQTT a plataformas en la nube con el fin de compartir los datos. Los sensores utilizados son electroquímicos, algunos responden con cambios en carga (SPEC) o cambios de resistencia como los sensores MQ y MICS. Esto fue implementado en un SoC (System on a Chip) ESP32 con un firmware de MicroPython para el desarrollo de los módulos. Al comparar el sensor MQ135 con un sensor profesional Drager 5500 se obtuvo porcentaje de error de $1.38 * 10^5$ %.

La infraestructura en la nube fue la intermediaria entre la fase de generación de datos (sensores que miden las concentraciones de gases contaminantes específicos) y la fase de presentación de resultados (página web) con la función de realizar el procesamiento de datos necesario para ser presentado al usuario de manera entendible y útil. Se optó por utilizar Microsoft Azure como proveedor de servicios en la nube y IoT Hub, Databricks y SQL Server como herramientas principales. Se evaluó la escalabilidad y tolerancia a fallos de la infraestructura a través de pruebas de carga sobre las herramientas que se utilizan, las cuales concluyeron de manera exitosa. También se validó el desempeño del procesamiento de datos realizando comparaciones de los valores AQI generados con los proveídos por la calculadora oficial de AQI de la Agencia de Protección Ambiental de Estados Unidos obteniendo resultados satisfactorios para 3 de los 4 contaminantes evaluados, con un porcentaje de error promedio de 5.91 %.

A través de un proceso de Design Thinking se identificaron los casos de uso. La infraestructura de la plataforma es de tipo nube, utilizando Amazon Web Services para servir una interfaz de aplicaciones desarrollada en Express JS, la cual permite a la interfaz web conectarse y consultar datos de la base de datos. Se desarrollo la interfaz de usuario en React JS y se integro y sirvió al internet utilizando Netlify. Utilizando una escala de usabilidad de sistemas, se efectuó una prueba sobre la usabilidad de la interfaz web obteniendo un 85.25 % de aceptación sobre esta. Por último, a través de una serie de diez preguntas sobre identidad y contenido de la página se obtuvo que 70 % de los usuarios opinaba que los elementos del tablero tienen un nivel 5 de utilidad y un 80 % opina que la información es suficiente para un análisis diario de calidad del aire.

CAPÍTULO 1

Introducción

El monitoreo y la cuantificación de los efectos de la exposición a la contaminación del aire es vital para la discusión de políticas ambientales según la Organización Mundial de la Salud [66]. La contaminación del aire provoca alrededor de 7 millones de muertes alrededor del mundo, afectando a países ricos y pobres [66]. En Guatemala, los programas que velan por obtener esta información se encuentran inactivos o bien no hacen de carácter público sus datos, por lo que la realización de estudios se ve limitado.

El presente proyecto busca informar a la población en general el estado actual de la calidad de aire en la ciudad, a través de una recolección continua de datos obtenidos por dispositivos electrónicos estratégicamente ubicados para proveer una muestra lo más significativa posible. El concepto general es tener dispositivos electrónicos dedicados a tomar mediciones de las concentraciones de los 4 principales gases tóxicos en áreas urbanas: dióxido de nitrógeno (NO_2), dióxido de azufre (SO_2), monóxido de carbono (CO) y ozono (O_3), luego los datos emitidos serán enviados a una plataforma en la nube en donde se centralizan, procesan y almacenan para que, por último, los resultados puedan ser presentados de manera gráfica e interactiva a usuarios interesados en obtener y analizar esta información. Estos sensores individuales para cada contaminante son ensamblados en un solo componente el cual se trasladará periódicamente en tres puntos estratégicos del departamento de Guatemala: el Centro Histórico de la ciudad, Carretera a El Salvador y Mixco.

El proyecto se dividió en tres módulos individuales, siendo: recopilación de datos (IoT), procesamiento de datos (plataforma en la nube) y presentación de datos a usuarios (página web). Este proyecto brinda la oportunidad para académicos de poder acceder a datos libres, realizar estudios y crear recomendaciones a las autoridades competentes. Adicionalmente brinda la oportunidad para la población de la Ciudad de Guatemala conocer la calidad de aire y sus posibles complicaciones a futuro.

CAPÍTULO 2

Objetivos

2.1. Objetivo general

Proveer información de libre acceso sobre el estado actual de contaminantes atmosféricos en la Ciudad de Guatemala a través de una solución que permita recopilar, distribuir y analizar datos para la implementación de estudios de polución del aire.

2.2. Objetivos específicos

- Ensamblar y programar un dispositivo electrónico que permita medir a un costo de aproximadamente $150USD las concentraciones de contaminantes atmosféricos como CO, O_3 NO_2, SO_2 y compartir sus concentraciones a una plataforma en la nube.
- Garantizar la comunicación continua del dispositivo con la nube en condiciones óptimas de red.
- Construir un contenedor que proteja al dispositivo del ambiente.
- Diseñar una infraestructura escalable que permita centralizar la información emitida por los dispositivos electrónicos.
- Establecer un API para que los dispositivos electrónicos puedan utilizar y comunicarse con la infraestructura.
- Determinar un pipeline para procesar los datos que continuamente se reciben.
- Implementar una base de datos en la nube para almacenar la información recolectada y procesada.
- Identificar el contenido de una plataforma web que permita visualizar, interactuar, y descargar los datos almacenados.
- Diseñar una interfaz gráfica usable que permita visualizar, interactuar, y descargar los datos almacenados.

- Desarrollar una interfaz de programación de aplicaciones web que permita obtener los datos almacenados y utilizarlos en una plataforma.
- Implementar la técnica de diseño web adaptable para que la interfaz sea compatible con dispositivos móviles.
- Implementar una arquitectura en Amazon Web Services que permita dar acceso público y continua actualización a cambios a la plataforma desarrollada.

CAPÍTULO 3

Justificación

La contaminación del aire es un problema que ha aumentado debido a la continua urbanización y actividad humana que puede tener consecuencias letales como infecciones respiratorias, enfermedades cardiovasculares, accidentes cerebrovasculares y cáncer de pulmón. Según la Organización Mundial de la Salud (OMS), alrededor del 91 % de la población mundial reside en lugares donde la calidad del aire excede los límites de contaminantes aéreos definidos por dicha organización. En el caso de Guatemala, según BreatheLife Org, en el 2016 la contaminación de partículas en suspensión era 4.1 veces mayor al límite establecido [59].

Guatemala ha implementado previamente estudios sobre contaminantes del aire, como el Programa de Calidad de Aire impulsado por el Instituto de Sismología, Vulcanología, Meteorología e Hidrología (INSIVUMEH) en el 2010, el cual reportaba datos diarios de cuatro estaciones de medición ubicadas en distintos puntos de la República sobre gases tóxicos como (CO_2), (NO_2), (O_3) y partículas menores a 10 micras. Los datos recopilados eran publicados en un boletín diario, semanal o anual, en formato PDF. Sin embargo, este programa se encuentra deshabilitado desde el 2016, por mantenimiento de las estaciones, por lo que la información ya no está actualizada. Por otra parte, la Universidad de San Carlos de Guatemala (USAC), ha realizado mediciones de calidad del aire desde 1995, pero dicha información recopilada no está disponible para acceso público [46]. Adicionalmente, según una entrevista realizada con el Dr. Edwin Castellanos, director del Instituto de Investigaciones de la Universidad del Valle de Guatemala, muchos de los problemas de continuidad con estos estudios tienen como origen el hecho de que los centros de medición se arruinan y son difíciles de reemplazar por razones de costo. Por lo que existe la necesidad de realizar una alternativa de bajo costo y fácil mantenimiento.

Los datos que han sido publicados relacionados con mediciones de calidad del aire en Guatemala, se encuentran de tal forma que resulta ineficiente su futuro uso para propósitos de estudio o se encuentran des-actualizados o privatizados, limitando así futuras investigaciones, récords históricos o posibles trabajos de Ciencias de la Computación sobre predicción de la calidad del aire y al acceso libre de datos.

Según la OMS, cuantificar y monitorear los efectos de la exposición a la contaminación del aire en términos de salud pública es un componente crítico en la discusión de políticas [66]. Debido a que actualmente los programas relacionados con el monitoreo de calidad de aire no están actualizados, es necesario dimensionar y visibilizar el problema. Por esto se identificó la oportunidad de realizar un proyecto que permita la fácil recolección de datos, el reportaje y distribución libre de los mismos que incentiven la realización de estudios que contribuyan con la discusión de políticas ambientales.

CAPÍTULO 4

Marco teórico

4.1. Contaminación atmosférica

La contaminación del aire se define como la presencia de elementos contaminantes que alteran su composición y que afectan a cualquier individuo dentro de un ecosistema. También se refiere a los agentes contaminantes que afectan la salud o bienestar humano [29]. Según su origen los contaminantes pueden ser:

- Antropogénicos: derivados de la actividad humana.
- Naturales: que resultan de procesos de la naturaleza como erupciones volcánicas o polen en suspensión.

4.1.1. Principales contaminantes atmosféricos en áreas urbanas

Los contaminantes de mayor interés determinados por EPA (Environmental Protection Agency, por sus siglas en inglés) son:

- Ozono a nivel de suelo (O_3)
- Material particulado (PM 2.5 o PM 10)
- Monóxido de carbono (CO)
- Dióxido de sulfuro (SO_2)
- Dióxido de nitrógeno (NO_2)

4.1.2. Ozono

La presencia de ozono (O_3) puede ser considerada positiva o negativa según en que parte de la atmósfera se encuentre. La presencia de ozono en la estratosfera es considerado natural, y de hecho

es beneficioso debido que bloquea rayos ultravioleta que son responsables de causar cáncer en la piel y otras complicaciones de salud. [10]

En caso de que exista presencia de ozono a nivel del suelo, o nivel troposférico, es considerado como un contaminante debido a que tiene efectos perjudiciales en la salud de las personas, de hecho es uno de los principales componentes del Smog:. [16]

El ozono se forma debido a la interacción de óxidos de nitrógeno (NO_x), compuestos orgánicos volátiles (COV) y luz solar. El origen de estos contaminantes puede ser varios tales como: carros, fábricas, refinerías, etc. [16].

Existen diversos efectos negativos en la salud de las personas al ser expuestas a concentraciones altas de ozono, entre ellas se encuentran:

- Tos y ardor en garganta
- Dificultad para respirar
- Inflamación de las vías respiratorias.
- Susceptibilidad a infecciones de pulmón
- Aumentos de ataques de asma

Los grupos más afectados son personas asmáticas o personas que padecen de enfermedades respiratorias. [23]

4.1.3. Monóxido de carbono

El monóxido de carbono (CO) es un gas inodoro e incoloro que puede ser dañino si es respirado en grandes concentraciones. El monóxido de carbono es emitido cada vez que ocurre una combustión. Este gas es encontrado en exteriores principalmente por la actividad de motores a combustión, que se encuentran en carros, aviones, motocicletas, etc. En ambientes interiores, las lamparas de queroseno, estufas a gas natural, y chimeneas emiten monóxido de carbono. [21]

Los efectos negativos que tiene el monóxido de carbono pueden ser variados, debido a que la inhalación del mismo reduce la cantidad de oxígeno que entra al flujo de sangre a través de los pulmones, esto puede afectar órganos críticos como el cerebro y el corazón. En muy altas concentraciones puede causar la muerte, entre otros síntomas también puede causar:

- Muerte en personas con sistema respiratorio y/o cardiaco comprometido.
- Ataques cardiacos
- Arritmia cardiaca
- Reducción de la capacidad pulmonar
- Irritación de las vías respiratorias.

Las personas que sufren de más riesgo son aquellas que tienen comprometida la salud cardiaca. [21]

4.1.4. Dióxido de sulfuro

El dióxido de sulfuro (SO_2) es considerado uno de los contaminantes más preocupantes para la EPA. La mayoría de fuentes de emisión de SO_2 son las combustiones de combustibles fósiles, y procesos industriales como la extracción de metales por medio de la refinación de minerales. Este gas también puede ser emitido por actividad volcánica. Los efectos del SO_2 ocurren a nivel de salud de las personas y nivel ambiental. La presencia en el aire de este contaminante puede causar la reacción química con los demás elementos en la atmósfera y crear otros óxidos de sulfuro (SO_x).

A nivel ambiental este gas causa la lluvia ácida, la cual afecta el crecimiento de las plantas y fertilidad del suelo.

En cuestiones de salud, este gas es principalmente dañino en niños, causando dificultad para respirar en exposiciones al gas a corto plazo. [24]

4.1.5. Dióxido de nitrógeno

El dióxido de nitrógeno (NO_2) pertenece al grupo de gases altamente reactivos conocidos como óxidos de nitrógeno (NO_x). La presencia de este contaminante proviene de la quema de combustibles de carros y camiones.

Los síntomas frente a exposición a altas concentraciones de este gas en cortos periodos de tiempo, se encuentran:

- Irritación de las vías respiratorias
- Tos, dificultad para respirar

Si existe una exposición a largo plazo frente a estos gases existen riesgos de:

- Asma
- Infecciones respiratorias

El grupo de riesgo frente a estos contaminantes son personas con asma, niños y personas de edad avanzada.

Adicionalmente, al igual que el SO_2, existen efectos negativos sobre el ambiente, particularmente estos gases son también responsables de la lluvia ácida y contaminación de nutrientes en los océanos. [22]

4.2. Indice de calidad del aire (AQI)

EPA desarrolló un índice de calidad de aire, llamado también AQI (Air Quality Index, por sus siglas en inglés). Este índice permite determinar que tan peligroso es el aire que está siendo respirado por las personas en una determinada área. El índice se reporta en rangos, cada rango indica el nivel de calidad del aire, según el siguiente cuadro [5]:

Categoría o nivel de calidad	Rango de valores	Color designado	Descripción de categoría
Bueno	0-50	Verde	La calidad del aire es satisfactoria y la contaminación presenta un peligro mínimo o nulo.
Moderado	51-100	Amarillo	La calidad del aire es aceptable. Sin embargo, es posible que exista un riesgo par algunos individuos, particularmente aquellos que son sensibles a la contaminación aérea.
No saludable para grupos sensibles	101-150	Anaranjado	Los miembros de grupos sensibles pueden experimentar efectos adversos en su salud.
Insalubre	151-200	Rojo	Algunas personas del público general pueden experimentar efectos adversos en su salud; miembros de grupos sensibles pueden experimentar efectos mucho más serios en su salud.
Muy insalubre	201-300	Púrpura	El riesgo de enfermedad aumenta para todos.
Peligroso	301-400	Granate	Alta probabilidad para todos de tener serios efectos de salud.

Tabla 4.1: Categorías del Índice de Calidad de Aire.

4.2.1. Cálculo del ICA

Según el Instituto Nacional de Sismología, Vulcanología, Meteorología e Hidrología de Guatemala (INSIVUMEH) – el cual es la organización gubernamental encargada de estudiar y monitorear fenómenos y eventos atmosféricos en el territorio nacional – el cálculo del Índice de Calidad de Aire en la región se realiza de acuerdo con la metodología establecida por la Agencia de Protección Ambiental de los Estados Unidos para el Air Quality Index (AQI por sus siglas en inglés). El cálculo es el siguiente:

$$I_x = \frac{I_{Hi} - I_{Lo}}{BP_{Hi} - BP_{Lo}} (C_p - BP_{Lo}) + I_{Lo} \tag{4.1}$$

En donde:

Ix = Índice de Calidad del Aire para contaminante x.

C_p = La concentración (truncada) del contaminante x.

BP_{Hi} = Concentración de corte que es mayor o igual a C_p.

BP_{Lo} = Concentración de corte que es menor o igual a C_p.

I_{Hi} = Valor AQI correspondiente a BP_{Hi}.

I_{Lo} = Valor AQI correspondiente a BP_{Lo}.

Para cada uno de los contaminantes medidos, se realizan mediciones continuas por ventanas de tiempo establecidas, dentro de las cuales se toma el valor más alto reportando como medición de ese

rango de tiempo. Para cada contaminante, la ventana de tiempo es la siguiente:

- Ozono: 8 horas o 1 hora.
- Monóxido de Carbono: 8 horas.
- Dióxido de azufre: 1 hora.
- Dióxido de nitrógeno: 1 hora.

A continuación, se presentan las concentraciones de corte previamente mencionadas en el cálculo y sus respectivos rangos AQI.

O_3 (ppm) 8 hr	O_3 (ppm) 1 hr	CO (ppm) 8 hr	SO_2 (ppb) 1 hr	NO_2 (ppb) 1 hr	Rango AQI	Categoría
0.000 – 0.054	-	0.0 – 4.4	0 – 35	0 – 53	0-50	Bueno
0.055 – 0.070	-	4.5 – 9.4	36 – 75	54 – 100	51-100	Moderado
0.071 – 0.085	0.125 – 0.164	9.5 – 12.4	76 – 185	101 – 360	101-150	No saludable para grupos sensibles
0.086 – 0.105	0.165 – 0.204	12.5 – 15.4	186 – 304	361 – 649	151-200	Insalubre
0.106 – 0.200	0.205 – 0.404	15.5 – 30.4	305 – 604	650 – 1249	201-300	Muy insalubre
-	0.405 – 0.504	30.5 – 40.4	605 – 804	1250 – 1649	301-400	Peligroso

Tabla 4.2: Concentraciones de corte para los distintos contaminantes.

En cuanto a las concentraciones truncadas (Cp), hacen referencia a la aproximación de las mediciones obtenidas de los distintos contaminantes a cierta cantidad de decimales. El criterio de aproximación varía dependiendo del contaminante y es el siguiente:

- Ozono: truncar a 3 decimales.
- Monóxido de Carbono: truncar a 1 decimal.
- Dióxido de azufre: truncar a 0 decimales (aproximar a número entero).
- Dióxido de nitrógeno: truncar a 0 decimales (aproximar a número entero).

Con lo anterior, el cálculo del AQI – para cada contaminante, individualmente – consiste en los siguientes pasos:

- De todas las estaciones reportando mediciones, escoger la medición más alta y truncar su valor de acuerdo con los criterios previos.
- Utilizando la Tabla 4.2, identificar los dos cortes o breakpoints dentro de los cuales está la medición.
- Utilizando la Ecuación 4.1, calcular el índice.
- Aproximar el valor resultante al entero más cercano.

En caso de que se tengan reportes de AQI de múltiples contaminantes y se desee tener un valor general, este será el AQI máximo entre los AQI individuales de cada contaminante.

Adicionalmente, en caso de que algunas mediciones dentro del rango de tiempo especificado para un contaminante no sean válidas, basta con tener datos válidos del 75 % de la ventana de tiempo para considerar el cálculo del AQI válido [3]. Por ejemplo, si utilizamos ventanas de 8 horas para obtener el AQI del ozono, para tener un cálculo correcto del AQI individual se deben tener 6 horas o más de datos válidos.

4.3. Internet de las cosas

IoT (Internet of Things) es una tendencia tecnológica que se basa en permitir interconectar elementos cotidianos a Internet, tales como cafeteras, bombillas de luz hasta dispositivos como sensores. Los dispositivos tendrían capacidades de formar intranets e incluso conectarse a internet. Los sistemas complejos de IoT funcionan con poca intervención humana. Existen múltiples objetivos que pueden cumplirse con un proyecto IoT, desde automatización como preparar el clima de la casa según la ubicación de un celular, hasta generar datos para *machine learning*. En el ámbito empresarial también se utiliza IoT con propósitos monetarios o de seguridad como implementación de controles de tarjetas *RFID*, o monitoreo de humedad y temperatura en cultivos que permiten programar riegos y optimizar el uso del agua. Las características principales de un dispositivo IoT son por lo general: Computadoras de bajo consumo energético, de tamaño pequeño y bajo consumo de *bandwidth*, por lo tanto esto sacrifica la capacidad de computación de cada uno de los dispositivos. Según las necesidades de cada proyecto, es probable que sea necesario implementar *edge computing*, que es agrupar datos de docenas de dispositivos recopilando datos y enviarlos en conjunto a un servidor capaz de procesar, distribuir y/o guardar los datos recopilados. [64]

4.4. Lenguaje de desarrollo: Micropython

Micropython es un lenguaje diseñado específicamente para dispositivos IoT, siendo un subconjunto de Python3. Debido a que los recursos de los microcontroladores son sustancialmente más limitados que una computadora de escritorio, este subconjunto de Python fue diseñado tomando en cuenta los recursos disponibles, según el fabricante requiere 256kb de memoria y ocupa 16kb de RAM, (Random Access Memory, por sus siglas en inglés). Adicionalmente, MicroPython proporciona una terminal interactiva, conocida como REPL, la cual cuenta con funcionalidades como historial de comandos, auto identación, y entre otras funcionalidades. MicroPython viene con sus librerías de código propias, adicionalmente incluye la librería *machine* para interactuar con el hardware de bajo nivel. [65]

4.4.1. Comparativa: Micropython y Arduino

En el contexto de desarrollo de Sistemas embebidos: existen varios lenguajes utilizados, cómo estándar está actualmente de uso popular Arduino y Micropython. Arduino es un lenguaje basado en C++:, utilizado en aplicaciones de sistemas embebidos, MicroPython busca cumplir el mismo objetivo, sin embargo ambos lenguajes difieren en como entregan el código al microcontrolador. Esto está relacionado a las diferencias fundamentales entre los lenguajes, Arduino funciona compilado, MicroPython como interpretado. Andreas Spiess, Ingeniero en electrónica y experto en IoT menciona en su vídeo titulado: *Time to Say Goodbye to Arduino and Go On to Micropython/ Adafruit Circuitpython?* las características de ambos lenguajes [72]. Arduino siendo un lenguaje compilado, es un lenguaje donde el código fuente es compilado en la computadora host, y lo que se envía al microcontrolador son los binarios ejecutables. En el caso de Micropython, al ser interpretado, se requiere que el código fuente exista dentro del microcontrolador para que sea interpretado por el ambiente de micropython y luego enviado a ejecución, esto hace que sea más intenso en el uso de

memoria, esto no es un problema para el microcontrolador, debido a que este cuenta con 520 Kb de RAM [26], y según los requisitos para la instalación de MicroPython este ambiente requiere 16 kb de RAM [65].

4.5. System on a chip

SoC (System on a Chip, por sus siglas en inglés) es un sistema completo de computación contenido en un paquete, no solo contiene dentro de esto un CPU, adicional a esto contiene múltiples partes de procesamiento tales como memorias, modems, GPUs, codificadores de vídeo, etc. No necesariamente debe tener todas estas funcionalidades presentes. Un chip que cumpla con múltiples propósitos distintos, lo hace por lo tanto, un SoC. El caso del SoC a utilizar es el ESP32 diseñado por Espressif [75]

El dispositivo utiliza varias interfaces de comunicación; DAC, SPI, I2C, I2S, Wi-Fi, sensores captativos e incluso Bluetooth. Para este módulo la comunicación entre sensores y SoC se utilizará DAC, UART y I2C.

4.6. UART

La comunicación (UART universal asynchronous receiver / transmitter, por sus siglas en inglés), define un protocolo o un conjunto de normas para el intercambio de datos en serie entre dos dispositivos. La comunicación entre dispositivos es asíncrona, a través de dos puertos identificados como TX (Transmisión) y RX (Recepción) estos deben de ir a puertos cruzados entre dispositivos. De tal forma, que si se desea enviar datos de un sensor al SoC, debe conectarse el cable Tx del sensor al Rx del SoC, de igual forma con el otro cable, de forma opuesta. La comunicación puede funcionar a varias velocidades, siendo estas: 9600, 19200, 38400 (medidas en Baudios:) por nombrar algunas de ellas. [47]

4.7. DAC

DAC (Digital Analog Converter, por sus siglas en inglés), es un circuito que se encarga de transformar señales análogas en señales digitales. Esto es necesario porque el procesador es únicamente capaz de procesar información binaria, la cual es digital, además la información comunicada por medio digital es menos suceptible a ruido externo que puede alterar los datos medidos. En el caso del ESP32, este utiliza un DAC de 12 bits, el cual enviará valores entre 0 y 4095, aunque también tiene canales de 10 bits el cual envía valores de 0 a 1024 bits. Esto se utiliza para obtener la información del sensor MQ135 y el sensor DHT-22 [40]

4.8. I2C

El bus de comunicación I2C (Inter-Integrated Circuit, por sus siglas en inglés) es un bus de comunicación serial de 8 bits que utiliza dos cables SDA (Serial data wire) y SCL (Serial clock wire). Este bus de comunicación permite la comunicación entre dispositivos independientemente de CPU clock del dispositivo maestro. I2C se maneja de manera maestro-esclavo, donde cada dispositivo conectado al bus tiene una dirección única. Existen distintas velocidades de comunicación, este puede

ir desde 100Kbits/s hasta 5Mbits/s. En este proyecto, los sensores MQ131 y el sensor MICS-6814 utilizan este bus para enviar sus mediciones al SoC. [74]

4.9. Wi-Fi

Wi-Fi es un nombre dado al estándar IEEE: 802.11 que define las reglas a seguir para realizar una conexión entre dos dispositivos a través de una red inalámbrica. A través de esta interfaz se puede realizar comunicación a un Router:, el cual proveerá internet al dispositivo conectado. Esto será utilizado en el proyecto para comunicar a un Endpoint: los resultados de las concentraciones obtenidas por los distintos sensores que serán implementados [11]

4.10. Unidad de concentración (ppm/ppb)

En situaciones de concentraciones diluidas, se utilizan medidas como ppm (partes por millón) o ppb (partes por billón). Esto se expresa por la ecuación:

$$C_{ppm} = (Ms/Md) * 1 * 10^6 ppm \tag{4.2}$$

Donde Ms, es masa del soluto y Md es la masa de disolución medidas en kilogramos.

Para entender el concepto de partes por millón o billón se puede ejemplificar como partir un pastel en 1 millón de partes, y se sustituye una pieza del pastel original por otra pieza de un tamaño idéntico, entonces se puede decir que ese pastel tiene 1ppm de material ajeno al pastel. [27] [6]

Esta unidad de medida que se utiliza para reportar presencia de contaminantes atmosféricos, es utilizada por la EPA (Environmental Protection Agency, por sus siglas en inglés) y posteriormente utilizada para calcular AQI (Air Quality Index, por sus siglas en inglés). [19]. Calculado por la ecuación:

$$I_p = (I_{Hi} - I_{Lo}/BP_{Hi} - BP_{Lo}) * (C_p - B_{Lo} \tag{4.3}$$

4.11. Sensores de gas

Los sensores de gas son dispositivos electrónicos que comunican a través de un voltaje, resistencia o carga eléctrica la presencia de un gas o incluso varios de ellos. Los sensores que son utilizados son los sensores semiconductores que interactúan de manera eléctrica con un gas reactivo (SO_2, CO, etc).

El funcionamiento de estos sensores, también conocidos como sensores de óxido de metal, miden la conductividad de la corriente que pasa a través de ellos. La capa sensitiva del dispositivo cambia en sus propiedades conductivas cuando entra en contacto con un gas en específico, estos sensores son principalmente utilizados para gases reactivos [8]

4.11.1. Sensores MQ

Los sensores MQ, se caracterizan todos por proveer un voltaje de entrada que calientan una resistencia, por ejemplo: el sensor MQ135 utilizado en el proyecto para la detección de CO (Monóxido de carbono) estos sensores requieren un tiempo de conexión al controlador de 24 horas para alcanzar

condiciones óptimas de medición. Los sensores MQ trabajan con un calentador, que se encarga de crear condiciones óptimas del circuito. La capa utilizada para detectar el gas objetivo de está fabricada de SnO_2 los valores retornados por el sensor son análogos, los valores retornados por el sensor deben ser convertidos a un R_s o resistencia sensorial. El mismo proceso ocurre para el MQ131, utilizado para detectar concentraciones de O_3 en el ambiente. Adicionalmente es importante mencionar que los sensores de esta familia tienen como ventaja ser de bajo costo y de alta durabilidad, sin embargo nunca serán igual de precisos que los sensores de grado profesional como la marca Dräger, por lo que no se espera que exista una calibración perfecta, pero si se espera tener aproximados de concentraciones de los gases. [39]

4.11.2. Cálculo de PPM en sensores MQ

Para el cálculo de ppm (partículas por millón) se requiere obtener información de como se comporta la relación de RS/R0 en las distintas concentraciones de los distintos gases, en este caso se tomará información sobre el sensor MQ135. Dicho procedimiento puede ser repetido con todos los sensores de la familia MQ, dado que el funcionamiento es el mismo y los cambios son únicamente los valores de la gráfica y constantes que se deben encontrar por cada sensor.

Variables a considerar

Variable	Descripción de variable
Rs	Valor de la resistencia medida por el sensor (en ohms)
RL	Valor de la resistencia presente en la placa del sensor (en ohms)
R0	Coeficiente único por cada sensor

Tabla 4.3: Tabla de variables relevantes en el cálculo de ppm

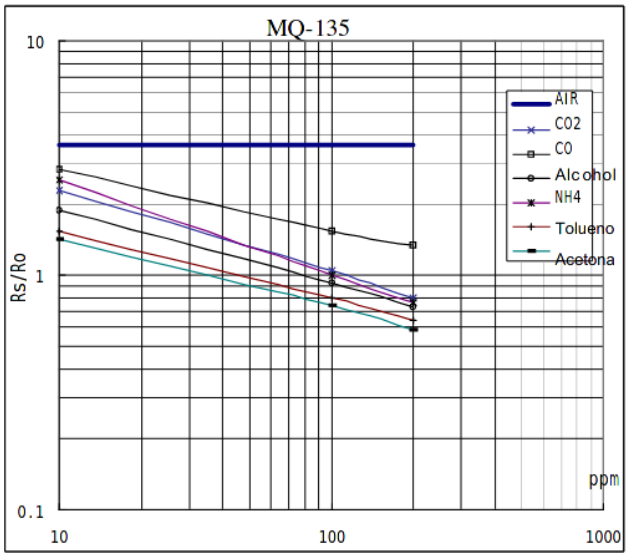


Figura 4.1: Gráfica de curva de sensibilidad para sensor MQ135

Al observar la Figura 4.1, la cual será llamada también curva de sensibilidad, se puede determinar que se observa una gráfica log-log, la cual tendrá tiene la función Rs/R0 contra ppm, por lo que se observa una función de tipo exponencial $y(x) = ax^b$, dada la pendiente negativa, es evidente que es una función $y(x) = ax^{(1/b)}$. Sustituyendo las variables en la función se puede determinar que la gráfica representa:

$$Rs/R0(ppm) = a * (ppm)^{1/b} \tag{4.4}$$

Dado que se requiere la inversa de la ecuación 4.4, es decir, una función que dado un RS/R0 esta retorne un ppm, se requiere despejar la ecuación para ppm, por lo que se obtiene esta nueva ecuación:

$$ppm(Rs/R0) = a * (Rs/R0)^{b} \tag{4.5}$$

Para obtener los valores a y b, es necesario realizar una regresión exponencial, para ello se debe contar con los valores de la gráfica 4.1. Finalmente, es necesario encontrar un R0, el cuál es único por cada sensor, por lo que se necesita calcular R0. Teóricamente, R0 puede ser calculado despejando la misma ecuación 4.4 para lo cual se obtiene:

$$R0 = Rs * (a/ppm)^{b} \tag{4.6}$$

4.11.3. Sensor MICS-6814

El sensor MICS-6814 es un sensor con capacidad de detección de múltiples gases, entre ellos CO, NO_2, H_2 y CH_4. La diseñadora de este sensor es Grove, vendida por la compañía Seeed. El sensor utiliza comunicación I2C, mencionado previamente. Según la documentación de este sensor, se requiere un periodo de precalentado de 30 segundos antes de utilizarlo para obtener mediciones más precisas. A diferencia de los sensores MQ, este tiene múltiples canales de comunicación, cada uno para reportar la resistencia obtenida en el ambiente medido. El fabricante menciona posibles efectos por los cambios de temperatura, o humedad y presencia de vapores de agua y VOCs (Volatile Organic Compounds, por sus siglas en inglés) sin embargo en el Datasheet: no muestra ninguna gráfica que permita obtener relación entre el cambio de RS y RO, por lo que no serán ajustados los valores obtenidos con las curvas de temperatura y humedad, a diferencia de como fue realizado con los sensores de la familia MQ. [69]

4.11.4. Cálculo de ppm en sensor MICS-6814

Para calcular el ppm del sensor MICS-6814, basta con observar la gráfica proporcionada por el fabricante, la cual lista en una misma gráfica los distintos valores de Rs/R0 según las distintas concentraciones de gas, para los gases detectables por el sensor. Debido a que la gráfica es log-log y presenta mismo comportamiento que para sensores de la familia MQ, las ecuaciones 4.4, 4.5 y 4.6 aplican de igual forma para este sensor.

4.11.5. Sensores SPEC

SPEC SO2 es un sensor electroquímico que emite un voltaje al entrar en presencia con el gas objetivo, en este caso en particular SO_2. El sensor al estar conectado a Vcc y GND entra en espera. La comunicación entre el dispositivo Host: y el sensor es a través de UART. El sensor tiene tres modos de consumo de energía: en espera, durmiente y activo. El fabricante indica que es común observar valores iniciales altos, por lo que recomienda dejar el sensor en proceso de calentado por un periodo de 1 hora por lo menos, hasta encontrar un punto de equilibrio en las mediciones. [68]

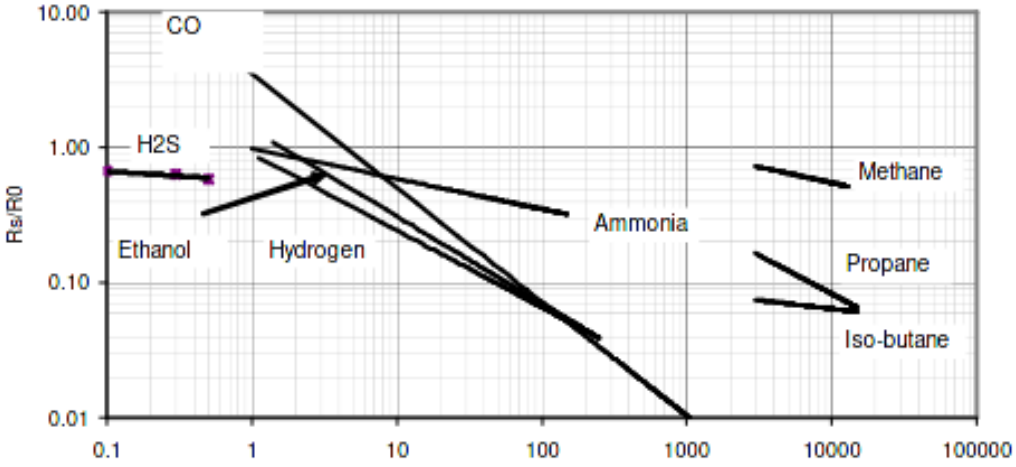


Figura 4.2: Gráfica Log-Log de Rs/R0-ppm para sensor MICS-6814

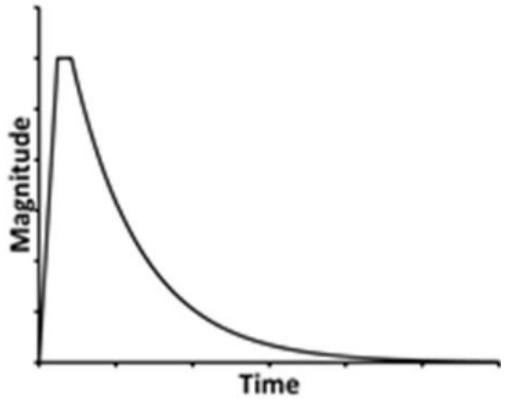


Figura 4.3: Gráfica de mediciones sobre tiempo DGS

4.11.6. Sensor DHT22

El sensor de humedad y temperatura digital DHT22, también conocido como AM2302 es un sensor digital que permite conocer las condiciones ambientales en las que se encuentra el mismo. Se comunica a través de un chip de 8 bits para transmitir datos al canal DAC del SoC al que esté conectado el sensor. El sensor DHT22 tiene capacidad de operar en condiciones de -40°c a 80°c según el datasheet del fabricante. Adicionalmente el fabricante menciona que debe tomarse datos por lo menos cada dos segundos para que el sensor tenga tiempo de reportar un nuevo valor de humedad y temperatura. [18]

4.12. Plataformas en la nube

La computación en la nube es la entrega de servicios computacionales tales como servidores, almacenamiento, bases de datos, manejo de red, Software:, etc. Todo esto es entregado y enviado a través de Internet. Esto permite tercerizar los servicios de despliegue de plataformas, disminuyendo

el tiempo de desarrollo y despliegue de proyectos tecnológicos. [52]

4.12.1. Azure IoT Hub

Azure IoT hub, es una plataforma que permite la comunicación entre muchos dispositivos IoT y la aplicaciones terceras a Azure, como portales web, aplicaciones móviles, etc. Cada dispositivo que es registrado en esta plataforma requiere autenticación generada por el administrador de la infraestructura. El administrador permite manejo de dispositivos conectados a esta red y autenticación de los clientes que deseen conectarse a la plataforma y análisis estadístico del uso de la red. El dispositivo funciona como el servicio mensajero a través de un protocolo MQTT, este es quien alimenta a la plataforma de los datos que se buscan transmitir a otros servicios dentro o fuera de Azure. [49]

4.13. Protocolo MQTT

El protocolo de comunicación MQTT es un protocolo de bajo consumo de Bandwidth: y mínima implementación de código, desarrollado específicamente para comunicación de dispositivos IoT a otras computadoras. Adicionalmente, tiene pensado utilizar la mínima cantidad de recursos con el fin que esta sea implementable a través de múltiples dispositivos IoT. El protocolo utiliza varias maneras de garantizar la comunicación y envío correcto de paquetes, minimizando la pérdida de los mismos, por ello se puede configurar modos de envío: Exactamente un paquete, al menos un paquete, como máximo un paquete. Adicionalmente, la comunicación puede ser de ambas vias, entre Broker: y dispositivo.

El protocolo MQTT permite encripción por TLS: (Transport layer security, por sus siglas en inglés), conexión de millones de dispositivos a través de redes inestables como las celulares, haciendo el uso de este protocolo ideal para dispositivos que requieran seguridad, escalabilidad y no tengan acceso a redes rápidas y/o estables.

Existen acciones que se realizan en este protocolo: publicar y subscribir. Por lo general, un dispositivo IoT se dedica a publicar, aunque también se puede subscribir a ciertos temas. Los temas son básicamente rutas definidas dentro de la conexión del protocolo.

Existen diversos actores en un protocolo MQTT:

- Editor: Publica información a la red, a través de un tema.
- Broker: Intermediario que permite que se comuniquen las distintos actores de la red.
- Cliente: Cualquier dispositivo que reciba información a la cual está suscrita dicho dispositivo a través de un tema

Un actor puede ser híbrido, por ejemplo un dispositivo IoT puede publicar y estar suscrito a un tema, de tal forma que sería editor y cliente al mismo tiempo. [60]

4.14. Análisis de datos

El análisis de datos o *data analytics* hace referencia al uso de poder computacional para descubrir e interpretar patrones significativos en los datos, es una herramienta importante para la toma de decisiones efectiva. En el caso de *big data analytics*, se refiere al análisis sobre un gran volumen de datos. Dentro de este concepto, existe la ingeniería de datos, en donde el objetivo es diseñar

y construir sistemas para recolectar, almacenar y analizar información de manera escalable; estos sistemas son comúnmente conocidos como *Data pipelines*: (Russom, 2011).

Dependiendo del caso de uso, la información que reciben los *data pipelines* de insumo tiene dos maneras de ser procesadas: en *batch* y en *streaming*. El primero hace referencia a data histórica o información que se obtiene por grupos o lotes; la lógica que se maneja con este tipo de procesamiento asume que las transformaciones se realizarán de manera periódica (por ejemplo, una vez al día, a la semana o al mes). Por otro lado, el *streaming* de datos indica que la información está siendo recibida de manera continua y que debe ser constantemente procesada. Se deben tener en consideraciones particulares como el hecho de que las agregaciones (sumas, máximos, promedios) deben ser realizadas en ventanas o rangos de tiempo y que es posible que algunos datos que deban ser incluidos en una ventana en específico lleguen en un momento posterior al cierre de esta; debido a lo anterior, la lógica interna de este tipo procesamiento cambia en comparación a *batches*.

Por lo general, debido al alto volumen de datos – de cantidades iguales o mayores a 1TB – que se manejan a la vez es necesario mucho poder computacional para lograr transformar la información. Es posible que este poder computacional deba ser tan alto que el hardware necesario para soportarlo aún no exista o requiera una inversión monetaria más alta de lo que las empresas puedan pagar. Para resolver este problema se ha creado la programación distribuida, la cual consiste en tener múltiples máquinas o nodos que son coordinados y trabajan al unísono dedicados a realizar una tarea.

En la actualidad, Apache Spark se ha convertido en la herramienta por excelencia utilizada para *big data analytics*, sobresale por su capacidad de ejecutar tareas más rápido que otras herramientas gracias a la optimización de *queries* y el almacenamiento temporal en memoria caché o *in-memory caching*. Spark puede ser ejecutado en lenguajes como Python, Java, Scala y R (Apache Foundation, s.f.).

4.15. Servicios en la nube o *Cloud Computing*

4.15.1. ¿Qué es la computación en la nube?

La nube es un gran conjunto de computadoras interconectadas con múltiples propósitos, que pueden ser públicos o privados. En la mayoría de los casos de proveedores de servicios en la nube, el hardware que conforma su nube es privado (es decir, son propiedad del proveedor únicamente) pero son públicamente accesibles a través del internet. Cualquier usuario autorizado puede acceder estos documentos y aplicaciones desde cualquier computadora a través de Internet. Y, para el usuario, la tecnología y la infraestructura detrás de la nube son invisibles [70]. Desde la perspectiva de uno de los proveedores de la nube más grandes en la actualidad, Google, *cloud computing* contiene las siguientes propiedades claves:

1. Centrado en el usuario: una vez un usuario se conecta a la nube, todo lo que almacene ahí – documentos, mensajes, imágenes, aplicaciones, etc – pasan a ser propiedad del usuario.
2. Centrado en tareas: en lugar de enfocar una aplicación que vive en la nube en lo que puede hacer, el enfoque está en lo que el usuario necesita hacer y lo que puede hacer con las aplicaciones.
3. Poderoso: conectar miles de computadoras en la nube genera una riqueza de poder computacional que puede ser accedido a través de una única PC. Los usuarios pueden utilizar cada vez más información en cuestión de segundos.
4. Inteligente: debido a la cantidad de información que se almacena en las computadoras en la nube, el análisis de datos es importante para acceder a esa información de manera eficiente e inteligente.

5. Programable: muchas de las tareas necesarias para *cloud computing* deben estar automatizadas. Por ejemplo, para proteger la integridad de los datos, si una computadora que almacena información queda fuera de servicio, la información debe ser replicada y redistribuida para evitar que se pierda.

La nube también da la posibilidad de escalar recursos de manera casi inmediata e infinita. Existen dos tipos de escalamiento de recursos: vertical y horizontal. Escalamiento vertical significa hacer crecer el hardware al darle más recursos al existente y hacerlo más potente, como aumentar el disco duro o memoria RAM a un servidor. Por otro lado, el escalamiento horizontal implica agregar más instancias de hardware a la red existente, coordinando las tareas para que los nodos trabajen como uno solo [9].

4.15.2. Modelos de despliegue

Comúnmente conocidos como *Deployment Models* por su nombre en inglés, los modelos de despliegue indican en dónde residirá la infraestructura en la nube que el cliente usará y quién tiene el control sobre la misma.

Cada modelo de despliegue en la nube está diseñado para satisfacer distintas necesidades dentro de una organización, por lo que existe cierta importancia en analizar bien el modelo que realmente satisfará las necesidades de la organización. Quizás aún más importante es el hecho de que cada modelo de implementación en la nube tiene una propuesta de valor diferente y diferentes costos asociados. Hasta ahora existen cuatro tipos, los cuales son descritos a continuación.

Pública

Una nube pública se define como servicios de cómputo ofrecidos por proveedores terceros a través de la internet pública, siendo accesibles para cualquier persona que desee usarlos, ya sea de manera pagada o gratuita. Por lo general, empresas en donde la seguridad de la información no es una prioridad fundamental escogen esta opción, principalmente porque no genera gastos relacionados a la compra, manejo y mantenimiento de hardware ubicado en instalaciones propias (también llamadas *on-premise*) y la infraestructura de las aplicaciones que desarrollan [56].

En este caso, el proveedor es el responsable por el manejo y el mantenimiento de todo el sistema, incluyendo su disponibilidad, fiabilidad, seguridad, redundancia y tolerancia a fallos. Las implementaciones en la nube también tienen la ventaja que pueden ser aprovisionadas más rápido que infraestructuras *on-premise* y con escalabilidad casi infinita. Por el aspecto público estas implementaciones suelen tener preocupaciones de seguridad (ya que, a través del internet, se podría decir que cualquier persona puede accederlo); sin embargo, los proveedores se han empeñado en desarrollar componentes que protejan lo más que se pueda los recursos que cada cliente tenga en la nube, así que cuando se implementa correctamente, un *deployment* en la nube pública puede ser tan seguro como uno en nube privada.

Los proveedores de este tipo de nubes utilizan el concepto de virtualización para poder brindar los recursos a cualquier cliente que desee comprarlos. La virtualización permite separar los recursos físicos de los servicios que se necesita que ejecuten, de manera que un solo recurso puede ejecutar múltiples servicios y así utilizarse el máximo de su potencial. Consiste en particionar un equipo o servidor físico en varias máquinas virtuales, cada una puede interactuar de forma independiente a las demás y ejecutar aplicaciones diferentes mientras comparten los recursos de una sola máquina [57]. Esto significa que una sola máquina física puede estar ejecutando aplicaciones o trabajos de una empresa A y al mismo tiempo estar ejecutando cargas de trabajo de una empresa B.

Privada

La nube privada se define como "los servicios informáticos que se ofrecen a través de Internet o de una red interna privada solo a algunos usuarios y no al público general" [58]. Por lo general, indica que todos los recursos dentro de esta red se encuentran ya sea dentro de las instalaciones empresariales (también denominado *on-premise*) y son completamente manejados por el cliente o en centros de datos de otros proveedores, pero contenido de tal manera que el acceso está mucho más limitado comparado a la nube pública y tiene privacidad con *firewalls* y hospedaje interno del proveedor, lo que da un nivel más alto de seguridad.

Similar a las nubes públicas, las privadas ofrecidas por terceros también utilizan tecnologías de virtualización para manejar sus recursos, con la diferencia que, por seguridad, los recursos físicos son exclusivos de un único cliente, no existe la opción que varios clientes compartan una sola máquina huésped. Esta exclusividad significa que el costo de este tipo de nube es un poco más alto que su contraparte, la nube pública [58].

Esta opción es ideal para clientes que tienen estrictas políticas de seguridad, requisitos de cumplimiento o normativas o para aquellos que, por limitaciones de presupuesto, no tienen los recursos para implementar su infraestructura *on-premise* pero la seguridad de los datos es de alta prioridad.

Híbrida

La nube híbrida es la combinación de la nube pública y la privada, en la que una parte de la lógica de una solución es manejada en una nube pública y el resto manejada en una nube privada o viceversa. Funcionalmente, una nube híbrida debe poder realizar lo siguiente:

1. Conectar sin problema varias computadoras u otros recursos físicos a través de una red.
2. Escalar horizontalmente y enlazar o añadir recursos nuevos con rapidez.
3. Tener la capacidad de trasladar cargas de trabajo entre los entornos.
4. Tener una única herramienta de gestión unificada para toda la infraestructura.
5. Organizar los procesos de manera automatizada.

Las nubes independientes (pública y privada) se consideran híbridas cuando se conectan de la manera más sencilla y sin problemas, ya que esta interconectividad es esencial para el funcionamiento correcto de una nube híbrida [63].

4.15.3. Tipos de servicios

Infrastructure-as-a-Service [IaaS]

En este tipo de soluciones, el servicio que se compra es sobre hardware (servidores, almacenamiento y redes) y su software asociado (como aplicaciones virtualizadas en sistemas operativos o sistemas de archivos. El proveedor de IaaS pone a disposición del cliente el hardware y los servicios de administración necesarios y deja a control del cliente el uso de dichos servicios de administración. Se incluye también el escalamiento de memoria, almacenamiento y ancho de banda ya que todo el equipo necesario es propiedad del proveedor. El beneficio clave de este tipo de servicio es la flexibilidad de los precios, ya que el cliente paga únicamente por los recursos que su aplicación utiliza y ampliarlos y reducirlos conforme se necesite; comparado a montar una infraestructura *on-premise*,

en donde el cliente se responsabiliza de todos los costos asociados con el alojamiento, ejecución y mantenimiento del sistema y una vez amplia los recursos, no puede reducirlos fácilmente.

En la actualidad existen muchas configuraciones para IaaS, desde computación completa como servicio hasta una infraestructura parcial como la obtención de solo almacenamiento, solo servidores, solo alojamiento web o una combinación de los anteriores. Los proveedores de este servicio pueden ofrecer un arrendamiento único o uno multiusuario; multiusuario siendo en donde múltiples clientes comparten los recursos físicos, aunque los sistemas de cada uno se mantienen separados y contenidos – esta es la forma más común de entrega de IaaS ya que es más eficiente y escalable, lo que permite que sea más barato. Por otro lado, en el arrendamiento único el proveedor alquila un espacio dedicado en su centro de datos a un único cliente [38].

Platform-as-a-Service [PaaS]

En este modelo, el proveedor proporciona a los clientes una plataforma de nube completa (hardware, software e infraestructura) para desarrollar, ejecutar y administrar aplicaciones sin el costo, la complejidad y la inflexibilidad que a menudo conlleva la creación y el mantenimiento de esa plataforma en las instalaciones; las ofertas de PaaS a menudo incluyen las herramientas necesarias para monitoreo, administración e implementación de la infraestructura [17]. En PaaS, el consumidor no administra ni controla la infraestructura física subyacente de la nube, como la red, los servidores, los sistemas operativos o el almacenamiento, pero sí tiene control sobre las aplicaciones implementadas y posiblemente las configuraciones para el entorno de la aplicación [38]. Cualquier oferta PaaS debe incluir necesariamente los recursos de IaaS requeridos para albergar la solución, aunque estos no sean explícitamente mencionados o referidos como IaaS.

Software-as-a-Service [SaaS]

Un servicio Software-as-a-Service es aquel en el que un software es entregado al usuario de la nube a través de un navegador web y puede hacer uso del software casi inmediatamente. Toda la infraestructura subyacente, el middleware, el software y los datos de las aplicaciones se encuentran en el centro de datos del proveedor, quien se encarga de administrar el hardware y el software y, con el contrato de servicio adecuado, garantizará también la disponibilidad y la seguridad de la aplicación y de sus datos. Las aplicaciones de este tipo se conocen también como software basado en la web, software bajo demanda o software alojado. La ventaja clave de este tipo de servicios es que permite que una organización se ponga en marcha rápidamente y pueda ejecutar aplicaciones con un costo inicial mínimo [30]. Es especialmente beneficioso para organizaciones con las siguientes características:

- Son organizaciones pequeñas que necesitan lanzar sus proyectos rápidamente y no tienen tiempo disponible para montar servidores o software y manejarlos.
- Los proyectos por desarrollar son de corta duración y necesitan una colaboración rápida, fácil y económica.
- Sus aplicaciones no son utilizadas con poca frecuencia, como por ejemplo un software para impuestos.
- Sus aplicaciones necesitan acceso desde plataformas móviles y web.

Comparación entre servicios

Los distintos servicios mencionados previamente se distinguen definiendo el nivel de control que tiene el usuario sobre cada grupo de componentes que conforman el servicio, y el nivel de control sobre los mismos que le queda al proveedor. La siguiente figura muestra dichos niveles de control.

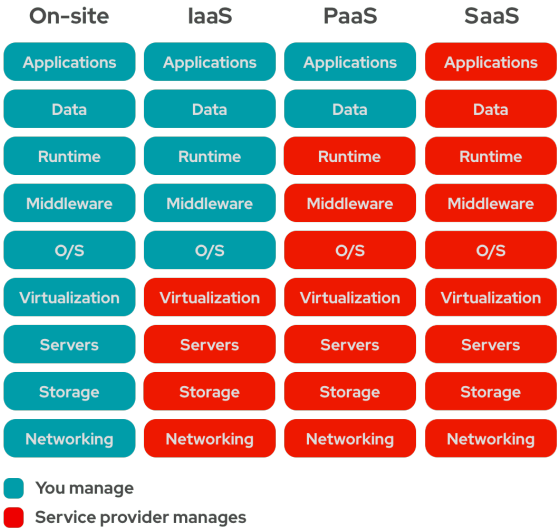


Figura 4.4: Alcance de gestión usuario-proveedor en los distintos servicios.

Al momento de escoger un servicio, es importante considerar estos niveles de control que se le delegan al usuario, ya que se deben contar con personas que conozcan sobre la infraestructura que están utilizando, para que el desarrollo sea menos propenso a errores. Por ejemplo, en un entorno IaaS el usuario debe estar atento a la seguridad de las aplicaciones instaladas, qué versión de aplicación instalar y a que la misma permanezca actualizada.

Adicional al alcance del usuario, se puede considerar que estos servicios ocupan distintos "niveles de jerarquía", ya que un servicio puede requerir implícitamente otro. En la siguiente imagen se ejemplifica esta jerarquía, mostrando qué componentes de computación utiliza cada servicio.

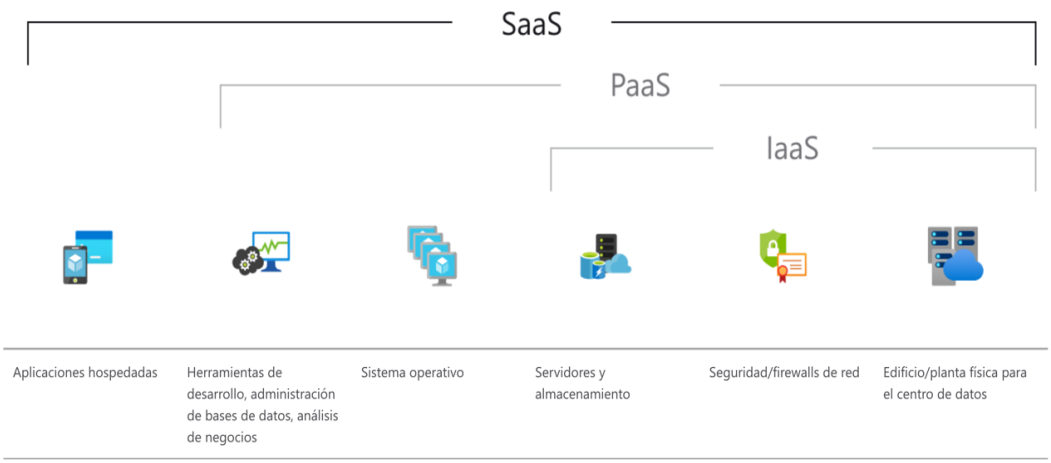


Figura 4.5: Dependencia entre ofertas as-a-service.

Se puede observar que los servicios PaaS utilizan los recursos IaaS necesarios para montar la solución, y SaaS necesita incluir los componentes PaaS necesarios para ejecutar las aplicaciones.

4.15.4. Principales proveedores

Amazon Web Services (AWS)

AWS es la plataforma en la nube desarrollada por Amazon Inc., ofrece más de 200 servicios desde 25 Zonas geográficas o regiones: alrededor del mundo. Estos servicios incluyen tecnologías de infraestructura como computación, almacenamiento y bases de datos, para desarrollar tecnologías emergentes como inteligencia artificial, *Data lakes*:, analítica de datos e Internet de las cosas o *Internet of Things*: (IoT). Tienen una de las infraestructuras en la nube más extensas del mundo, teniendo 81 *Availability zones*: dentro de las regiones geográficas y ha anunciado que está planificando expandirse y agregar 21 zonas más en 7 regiones geográficas adicionales: Australia, India, Indonesia, Israel, España, Suiza y Emiratos Árabes Unidos. Es una de las plataformas preferidas debido al amplio catálogo de servicios y aplicaciones que ofrecen, a un precio competitivo: los servicios más populares usualmente tienen un *Free Tier* que es una versión básica del servicio que es gratis, diseñado para que los clientes puedan probar la plataforma y decidir montar el resto de su infraestructura en AWS [7]. Amazon también ofrece los siguientes planes de pago:

- *Pay-as-you-go*: en esta forma de pago, el cliente paga por lo que ha consumido en un período de tiempo establecido, usualmente es un mes. Cada servicio tiene precios establecidos dependiendo de los recursos que se utilizan, por lo que este plan es ideal para clientes que no saben su consumo mensual, es demasiado irregular y no desean pagar de más o no quieren depender de predicciones para limitar los recursos que pueden usar.
- *Saving Plans*: es un modelo de pago flexible en donde se reserva una cantidad de recursos específica (medido en dólar estadounidense por hora) por un período de uno o tres años; al realizar esta reserva, se ofrecen precios más bajos comparado a si se comprara conforme se usa.
- Descuentos por volumen: Amazon ofrece una rebaja en los precios si los recursos se compran en volumen, entre más recursos se utilizan, menos se paga.

Microsoft Azure

Azure es la plataforma de computación en la nube proveída por la empresa tecnológica Microsoft. Está compuesta por más de 200 productos y servicios en la nube, diseñadas para construir, probar, desplegar y administrar distintas aplicaciones y servicios teniendo Azure como proveedor. Pone a disponibilidad de sus clientes los tres modelos de servicios: IaaS, PaaS y SaaS. Sus servicios ofrecen una alta disponibilidad (del 99.99 %) y un amplio portafolio de cursos educativos y más de 100 certificaciones de alto nivel, clasificadas en los distintos roles laborales que se tienen en el ámbito técnico. Adicionalmente, cuenta con un programa estudiantil denominado *Microsoft Student*, en donde ofrece un crédito de USD$100 para su uso en 12 meses, además de 25 servicios sin costo, disponibles mientras se tenga crédito restante [54].

Google Cloud Platform (GCP)

Google Cloud Platform es el servicio PaaS para computación en la nube creada por el gigante tecnológico Google. Se encuentra presente en más de 200 países y territorios, con una red de 228 regiones y 85 zonas, altamente aprovisionada y de baja latencia – la misma que se utiliza para productos populares como Gmail, búsqueda de Google y Youtube. Sus centros de datos se encuentran entre las instalaciones más seguras y de mayor eficiencia energética, utilizan seguridad en varias capas, redundancia integrada y tolerancia a errores. A diferencia de los proveedores anteriores, GCP únicamente tiene la metodología de pago de *pay-as-you-go* el cual funciona similar a Azure y AWS. Ofrece USD$300 de créditos gratuitos para cuentas primerizas y más de 20 productos que son gratuitos todo el tiempo – siempre y cuando no se excedan sus límites de uso mensuales [33].

Comparación de servicios

Este año (2021), Gartner – la empresa líder consultora y de investigación de las tecnologías de la información con sede en Estados Unidos – publicó un reporte en donde se evaluaban los distintos proveedores de tecnologías en la nube en función de su madurez y visión (planes a futuro y capacidad de ejecutarlos) para identificar y analizar a los proveedores de tecnologías en la nube más relevantes y sus productos en el mercado. En este reporte, AWS, Azure y GCP fueron todas clasificadas como líderes en la industria [62].

4.16. Bases de datos

Una base de datos es un conjunto de datos englobados por un mismo contexto y almacenados sistemáticamente para posteriormente acceder y utilizarlos de manera eficiente.

4.16.1. Tipos de bases de datos

Relacionales (SQL)

Las bases de datos relacionales se basan en el uso de objetos que gráficamente están representados como una tabla con filas y columnas y cómo se relacionan estos distintos objetos. Cada columna almacena información sobre una propiedad (también llamado atributo) específica de la tabla y cada fila representa una ocurrencia de la instancia representada por la tabla [73].

Documentales (No-SQL)

Este tipo de bases de datos utiliza documentos como la estructura de almacenamiento y consulta de datos. Un documento está compuesto por registros y datos los cuales conforman la información que luego es indexada para su acceso y uso. Están construidas con base en lenguaje NoSQL, lo que permite que maneje grandes volúmenes de información en períodos mínimos de tiempo [25].

4.17. Design thinking

El design thinking es un método de resolución de problemas, enfocado a un sistema de soluciones creativas con procesos pertinentes y de carácter social. Consiste en analizar y trabajar a través diferentes perspectivas o puntos de vista, en conjunto con diversas profesiones, problemas interdisciplinarios complejos y crear un proceso de aprendizaje que permita generar nuevos conocimientos que generen algún rendimiento [48].

4.17.1. Etapas fundamentales del Design Thinking

- Empatizar: adquirir los conocimientos básicos sobre los usuarios, pero también sobre el problema general, logrando empatía con las necesidades de los usuarios.
- Idear: generar todas las ideas posibles.
- Prototipar: construir prototipos reales de algunas de las mejores ideas y con más potencial.
- Evaluar: aprender partiendo de las reacciones de los usuarios con respecto a los diferentes prototipos.

En resumen mediante este proceso iterativo, tomando en cuenta la experiencia del usuario, el uso de la creatividad y la ejecución y el testeo, permite crear innovaciones y soluciones que se centran en los usuarios y no en su totalidad en los productos [28].

4.17.2. Herramientas del Design Thinking

- Focus Group: método cualitativo para obtención de datos.
- Shadowing: observar cuidadosamente las situaciones de la vida real en un determinado tiempo para comprender el comportamiento de las personas en un determinado contexto.
- Fichas de personas: modelos de caracterización que toman en cuenta la descripción física, psicológica y sociológica.
- Perfiles de segmentos de clientes: proceso de dividir un mercado en distintos subconjuntos de consumidores con características en común.
- Mapa de empatía: comprender la experiencia de otras personas sin realmente pasar por dicha experiencia para convertirse en el segmento del cliente deseado.
- Mapa del viaje del cliente: permite construir una imagen realista de lo que realmente sucede en torno al usuario.
- Árbol de problemas: herramienta visual de análisis usada para identificar el problema objeto de estudio. Permite analizar las causas y efectos de primer y segundo nivel de un problema central.

- Curvas de valor: o Strategy Canvas, representación gráfica de la dinámica competitiva del mercado actual.
- Mapa de contexto: herramientas utilizadas para mostrar factores externos, tendencias y fortalezas que rodean a la institución.
- Prototipo: modelo para desarrollar nuevas ideas.
- Mapa de oferta: atributos que crean las diferencias entre dos productos genéricos similares.
- Modelo de negocios: es la forma en la que la empresa lleva a cabo su negocio.
- Testeo de prototipos: evalua si el prototipo de adecua a las necesidades iniciales del proyecto
- Testeo de usabilidad beta en entorno real: también conocido como Test Driven Development (TDD), prátia de desarrollo de software en el que los casos de prueba de unidad se escriben gradualmente antes de la implementación del código.

4.18. Sistemas y tecnologías web

Los sistemas web se refieren a la manera en como las computadoras y dispositivos se comunican entre ellas [76].

4.18.1. Página Web

Es un documento digital de carácter multimediático capaz de incluir diferentes tipos de contenidos y se adapta a estándares de World Wibe Web. Este documento puede ser accedido a través de un navegador web [76].

4.18.2. Servidor Web

Un servidor web es un programa o una máquina que genera y transmite respuestas al clientes a través de consultas o peticiones a través de un navegador web utilizando un protocolo HTTP o HTTPS [76].

4.18.3. Protocolo

Un protocolo es un conjunto de reglas que gobierna la manera en que los datos se comunican. Un protocolo define qué, cómo y cuándo se realiza la comunicación [76].

4.18.4. Interfaz de aplicación de programación

Una interfaz de aplicación de programación, o por sus siglas en ingles API, es un conjunto de estructuras de datos, rutinas o protocolos, utilizados por una aplicación para comunicarse con otras [76].

4.18.5. Entornos de trabajo

Un entorno de trabajo es una estructura sobre la cual se puede crear y desarrollar código para la creación de sistemas. Sirve como base para no empezar de cero. Estos suelen estar asociados con un lenguaje de programación específico y se adaptan a diferentes tipos de tareas . Utilizar un entorno de trabajo hace el desarrollo mas rápido y reduce el riesgo de errores. [12].

React JS

React JS es una librería de código abierto para desarrollar interfaces de usuarios. Es frecuentemente utilizada en el desarrollo de páginas de una sola aplicación, en donde la página se esta actualizando dinámicamente con distintos datos. React solo se ocupa de la administración del estado y la representación de ese estado en el DOM, por lo que la creación de aplicaciones React generalmente requiere el uso de bibliotecas adicionales para el enrutamiento, así como ciertas funciones del lado del cliente [45].

Document Object Model

El Document Object Model (DOM) es una interfaz multiplataforma e independiente del lenguaje que trata un documento XML o HTML como una estructura de árbol en la que cada nodo es un objeto que representa una parte del documento. El DOM representa un documento con un árbol lógico. Cada rama del árbol termina en un nodo y cada nodo contiene objetos. Los métodos DOM permiten el acceso programático al árbol; con ellos se puede cambiar la estructura, el estilo o el contenido de un documento [15].

Cuando se carga una página web, el navegador crea un DOM de la página, que es una representación orientada a objetos de un documento HTML que actúa como una interfaz entre JavaScript y el documento en sí. Esto permite la creación de páginas web dinámicas, porque dentro de una página JavaScript puede:

- Modificar y eliminar los elementos HTML
- Cambiar los estilos CSS
- Reaccionar a los eventos existentes
- Crear nuevos eventos

DOM Virtual

Es un concepto de programación donde una representación ideal de la interfaz de usuario se mantiene en memoria y sincronía con el DOM real, mediante una biblioteca como ReactDom [42]. Dentro de React, el término *DOM virtual* es asociado con elementos de React ya que son objetos representando la interfaz de usuario.

Componentes de React

Los componentes son un conjunto de código independientes y reutilizables. Tienen el mismo propósito que las funciones de JavaScript, pero funcionan de forma aislada y devuelven HTML [41].

Existen dos tipos de componentes, componentes de clase y componentes de función. Los componentes de clase deben incluir la declaración extiende *React.Component*. Esta declaración crea una herencia para React.Component y le da a su componente acceso a las funciones de React.Component. El componente también requiere un método *render ()*, este método devuelve HTML.

Los componentes de función también devuelve HTML y se comporta de la misma manera que un componente de clase, pero los componentes de función se pueden escribir usando menos código, son más fáciles de entender.

Axios

Axios es una biblioteca de Javascript que se utiliza para realizar solicitudes HTTP desde node.js o XMLHttpRequests desde el navegador. Es isomórfico, lo cual quiere decir que se puede ejecutarse en el navegador y en Nodejs con la misma base de código. En el lado del servidor usa el módulo http nativo node.js, mientras que en el cliente (navegador) usa XMLHttpRequests [**?**]. Entre sus principales funcionalidades se encuentra:

- Hacer XMLHttpRequests desde el navegador.
- Realizar solicitudes http desde node.js
- Interceptar solicitudes y respuestas.
- Transformar los datos de solicitud de respuesta.
- Cancelar solicitudes.
- Transformaciones automáticas para datos JSON.

Express JS

Express es una infraestructura de aplicaciones web Node.js mínima y flexible que proporciona un conjunto de características para las aplicaciones web y móviles. Cuenta con una interfaz de programación de aplicaciones que tiene disponibilidad de métodos y aplicaciones http, lo cual permite la creación de una interfaz rápida y sencilla [14]. Los mecanismos que proporciona son:

- Escritura de manejadores de peticiones con diferentes verbos HTTP en diferentes caminos URL (rutas).
- Integración con motores de renderización de vistas para generar respuestas mediante la introducción de datos en plantillas.
- Establecer ajustes de aplicaciones web como qué puerto usar para conectar, y la localización de las plantillas que se utilizan para renderizar la respuesta.
- Añadir procesamiento de peticiones (middleware) adicional en cualquier punto dentro de la tubería de manejo de la petición.

4.18.6. Experiencia de usuario

La experiencia del usuario es cómo se sienten las personas cuando usan un producto o servicio. En la mayoría de los casos, el producto será un sitio web o una aplicación. Cada elemento que tiene

interacción con el usuario tiene una experiencia asociada, la experiencia de usuario se enfoca como los usuarios interactúan con estos elementos a través de una computadora [28].

Centrarse en la experiencia de usuario permite que el diseño se centre en el cliente final esto aumenta las posibilidades de éxito de un proyecto.

Diseño web adaptable

El diseño web adaptable es un enfoque de diseño de interfaz gráfica de usuario (GUI) que se utiliza para crear contenido ajustable a varios tamaños de pantalla. Se utilizan consultas de medio para que los diseños adapten el contenido a los distintos tamaños de pantalla [28].

Escala de usabilidad del sistema

La Escala de Usabilidad del Sistema, SUS por sus siglas en inglés (System Usability Scale), fue una de las primeras escalas que surgieron para evaluar la usabilidad de una interfaz, la cual surgió en el año 1986. Existen tres aspectos clave que no deben perderse de vista durante su implementación: eficacia, eficiencia y satisfacción [2]. Estos aspectos se definen como:

- La eficacia por el grado de precisión y totalidad del usuario para lograr actividades u objetivos específicos.
- La eficiencia como el grado en que los recursos son utilizados para que el usuario logre sus objetivos con precisión y totalidad.
- La satisfacción como la libertad del usuario para mostrarse incómodo o mostrar actitud positiva utilizando el sistema.

Google Search Console

Google Search Console es un servicio gratuito de Google que ayuda a supervisar, mantener y solucionar problemas de aparición de un sitio web en los resultados de Búsqueda de Google [32]. Entre sus propósitos se encuentra:

- Solucionar problemas relacionados con la usabilidad móvil y otras funciones de la Búsqueda.
- Confirmar que Google pueda encontrar y rastrear la página web.
- Solucionar problemas de indexación y solicitar que se indexe contenido nuevo o actualizado.

Google Search Console cuenta con un test llamado *Mobile-Friendly Test* el cual permite escanear una pagina web y evaluar su usabilidad móvil a través de tres términos:

- Adaptación correcta en dispositivos móviles.
- Tamaño de letra.
- Separación entre contenido clickeable.

CAPÍTULO 5

Antecedentes

Existen trabajos previos sobre tomar medidas de concentración con los sensores de la familia MQ, durante la investigación de este trabajo, se encontró el sitio web del Ing. Davide Gironi, graduado como Computer Scientist en la Università degli Studi. El trabajo de Davide Gironi demuestra que tomando medidas del sensor MQ135 es posible obtener valores aproximados a las mediciones de ppm de CO2 por un sensor infrarrojo modelo MH-Z14. Adicionalmente, David Gironi propone un flujo de trabajo de como analizar, determinar R0 para cada sensor y calibrarlos. Por lo que su trabajo tiene una gran influencia sobre este mismo para poder determinar concentraciones bajo un método ya probado. [31]

Adicionalmente, existen trabajos previos en la implementación de estaciones de medición de calidad del aire en el país, sin embargo, este se encuentra en mantenimiento desde 2016, según la página del INSIVUMEH por lo cual no ha generado datos desde ese entonces. [44]

CAPÍTULO 6

Alcance

El alcance de este trabajo es determinar si es factible desarrollar un sistema de monitoreo y reporte que permita compartir datos de contaminantes atmosféricos a través de internet, con el fin de reportar indices ICA. Con este propósito, se determina la factibilidad tomando 6 horas de datos continuos en las distintas ubicaciones delimitadas del proyecto: Centro Histórico, Carretera a el Salvador y Mixco.

CAPÍTULO 7

Marco metodológico

Para la elaboración del proyecto se realizaron distintas investigaciones sobre el estado actual de sistemas o programas en Guatemala que distribuyeran y proporcionaran datos recientes de la calidad del aire en cualquier punto del país. Una vez completadas las investigaciones, se procedió a la realización del proyecto.

La sección de IoT fue desarrollada comenzando por la selección de equipo a utilizar, donde las características de los sensores respondieran a las necesidades técnicas y económicas del proyecto, tomando en cuenta que se debe asegurar la comunicación entre computadora y microcontrolador. Para esto, se utilizaron herramientas como esptool, rshell y ampy, las cuales tienen la funcionalidad de configurar firmware, enviar comandos al microcontrolador y enviar y recibir archivos entre microcontrolador y computadora, respectivamente. Seguido de esto, se procedió a la calibración individual de cada sensor según las recomendaciones del fabricante, seguido de pruebas individuales y finalmente pruebas piloto con comunicación a la infraestructura en la nube.

Para la implementación de la infraestructura en la nube, primero se definió cuál sería el proveedor de servicios en la nube sobre el cual se haría el *deployment*, analizando la oferta de servicios de los tres proveedores más populares hoy en día: Amazon Web Services [AWS], Microsoft Azure y Google Cloud Platform [GCP]. Con base en esos resultados, se diseñó la infraestructura como tal, determinando los recursos que se utilizarían y sus conexiones. Seguido de esto, se realizó el diseño de la base de datos y el *pipeline* que realizaría el procesamiento de datos. Por último, se ejecutaron las pruebas, evaluando el insumo de los datos provenientes del módulo de IoT y la conexión con la plataforma web.

Para el desarrollo de la plataforma web, se comenzó con la identificación de casos de uso a través de un proceso de Design Thinking. Tras los resultados, se realizaron distintos prototipos, que se validaron a través de una serie de preguntas que buscaban enfocarse a la identidad y contenido de la página web final. Después, se procedió a elegir las tecnologías a utilizar para el desarrollo, la tecnología elegida para la interfaz de usuario fue ReactJS y ExpressJS para la interfaz de programación. Se construyó una arquitectura en la nube con AWS, utilizando un servidor EC2 y un proxy inverso de Nginx para que la interfaz de programación fuera accesible desde la interfaz de usuario. Seguido de esto, se utilizó Netlify para la implementación de integración continua y acceso público de la interfaz de usuario. Por último, se realizaron pruebas utilizando una escala de usabilidad de sistemas.

A continuación, se detallan las distintas fases empleadas a lo largo del desarrollo del proyecto.

7.1. Selección de sensores

Para iniciar el proyecto fue necesario realizar una investigación sobre los sensores usados para el cálculo de concentraciones de gas. Para ello se buscó sensores que fueran compatibles con Arduino y tuvieran implementaciones o documentación en MicroPython. Otro factor que se tomó en cuenta fue la investigación sobre los sensores existentes de concentraciones de gas, donde los sensores de la familia MQ y el sensor de la familia SGX, al cual pertenece el MICS-6814. Todos estos sensores tienen un amplio rango de temperatura de operación [-20°C, 50°C] y bajo consumo energético $<$ 200mA a 5V. El sensor SPEC fue utilizado porque fue el único que se encontró en Stock: que tenía las capacidades de medir SO2, pero se pudo haber utilizado el sensor MQ136. [8]

7.2. Comunicación entre ESP32 y PC

El dispositivo para comunicarse a la computadora host utiliza un cable Micro USB: para poder comunicarse al microcontrolador. La computadora necesita tener instalados los controladores CP210X de Silicon Silabs. De lo contrario no será posible realizar comunicación con el dispositivo. En el ambiente de desarrollo, Manjaro Linux: el dispositivo fue listado exitosamente utilizando el comando lsusb, por lo que no requirió instalación de controladores.

7.3. Selección de ambiente de desarrollo

Debido a que la cantidad de RAM: requerida para la instalación de MicroPython era suficiente para este microcontrolador en particular, siendo como requisito 16Kilobytes: de RAM para poder ejecutar el ambiente de MicroPython. [65], puesto que el microcontrolador cuenta con 520Kb de RAM [26].

Considerando que MicroPython es un lenguaje que tiene menos implementaciones de librerías esto es una desventaja frente a Arduino, sin embargo la programación de MicroPython requiere una corta curva de aprendizaje debido a que es un subconjunto del lenguaje Python:. Debido a la potencial reducción de tiempo en el desarrollo general del proyecto se optó por la instalación de MicroPython sobre el firmware existente de la ESP32.

7.4. Instalación de Micropython en ESP32

Debido a que el ESP32 viene por defecto de fábrica instalado con firmware compatible con Arduino, fue necesario eliminar el firmware existente de Arduino.

Fue necesario tener conectado el ESP32 a la computadora través de un cable micro-usb y tener instalado Python3, configurado como variable de entorno. Se realizaron los siguientes pasos para poder sustituir el firmware de Arduino a uno de MicroPython.

Para estos pasos es necesario tener control a una terminal con acceso privilegiado para realizar cambios necesarios en el sistema y en el microcontrolador.

- pip install esptool

- sudo chmod 666 /dev/ttyUSB0
- esptool.py –port /dev/ttyUSB0 erase_flash
- Descarga de imágen: https://micropython.org/download/#esp32
- Navegar al path donde se encuentra el .bin descargado
- esptool.py –chip esp32 –port /dev/ttyUSB0 write_flash -z 0x1000 “esp32-20210623-v1.16.bin”

7.5. Comunicación entre PC y REPL

A continuación se detalla los pasos que se tomaron para establecer comunicación entre la computadora y el REPL ya instalado en el microcontrolador ESP32 (Read, eval, print loop por sus siglas en inglés). Para establecer la comunicación entre el sensor ESP32 y la computador fue necesario hacer uso de RShell, la cual es una herramienta desarrollada en Python que permite realizar comunicación con la REPL del microcontrolador desde la computadora host.

Adicionalmente, fue utilizado Ampy, una paquete de Adafruit para Python que permitió el manejo de envío y recepción de archivos entre la computadora host y el microcontrolador. A continuación se detalla como se instaló y se comunicó con el dispositivo.

- pip install rshell
- pip install Ampy
- rshell –buffer-size=30 -p /dev/ttyUSB0

Para el envío y recepción de archivos se ejecutaron los siguientes comandos:

- ampy -p /dev/ttyUSB0 put <archivo>
- ampy -p /dev/ttyUSB0 get <archivo>

7.6. Cálculo de ppm en sensores MQ

Para el cálculo de las partículas por millón con los sensores MQ implementados, MQ135 y MQ131 fue necesario utilizar los parámetros mostrados en las gráficas Log-Log ppm-Rs/R0 para ambos sensores.

Los sensores MQ poseen tres variables relevantes para este cálculo:

- R0: Resistencia del sensor sin presencia del gas objetivo, valor en ohms.
- Rs: Resistencia del sensor con alguna concentración de gas objetivo, valor en ohms.
- RL: Resistencia conectada al sensor, valor en ohms.

Estas variables vienen dadas por la ecuación 4.5.

Se obtuvo las siguientes R0, a y b con el fin de obtener la ecuación que permite el cálculo de ppm 4.5 del gas objetivo, siendo en este caso para los sensores MQ135 y MQ131 que miden los gases CO y O3 respectivamente.

7.6.1. Calentamiento del sensor

Es importante mencionar, que se requirió un periodo de calentamiento, durante el primer uso del sensor de un periodo de 10 minutos continuos antes de realizar cualquier cálculo.

Esto implica dejar el sensor conectado a tierra y un voltaje de 5V durante este tiempo para que provea mediciones correctas. Esto no es mencionado por el fabricante en su datasheet, pero fue mencionado por el distribuidor, por lo que esta práctica fue aplicada con ambos sensores MQ131 y MQ135. [13]

7.6.2. Determinando ecuación de la gráfica de sensibilidad

Para determinar la ecuación de la curva de CO es necesario conocer los puntos de la Figura 4.1. Los puntos extraídos de las gráficas son obtenidos utilizando WebplotDigitalizer. Esta herramienta fue utilizada debido a que las gráficas listadas en el datasheet de los sensores son ser difíciles de leer dada a la baja resolución utilizada en las imágenes. WebplotDigitalizer permitió obtener los puntos de la gráfica una vez fueron ingresados los valores iniciales y finales de ambos ejes, eliminando potencial error humano. Los siguientes puntos representan la curva para el gas CO. Los puntos fueron extraídos de la Figura 4.1.

ppm	**Rs/R0**
9.9375	2.8405
20.0466	2.3509
30.3133	2.1079
39.9356	1.9600
50.0403	1.8492
80.0574	1.6341
100.9443	1.5305
199.8390	1.3525

Tabla 7.1: Puntos obtenidos de gráfica log-log sensor MQ135

7.6.3. Cálculo de RS

Obteniendo los valores devueltos por el DAC de la ESP32, los cuales son de 0 a 4095 debido a la resolución de 12 bits del microcontrolador, se calculó la resistencia con base a ese rango de valores.

La variable x, representa el valor en bruto que retorna el sensor a través del pin A0, en este caso los valores entre 0 y 4095.

La variable RL indica el valor de la resistencia a la cual está conectado el sensor. Se determinó que el valor de la resistencia es de 1,000 ohms debido al código impreso en la resistencia 102.

La ecuación utilizada para calcular la resistencia actual del sensor (Rs) fue entonces:

$$Rs(x) = ((4096 * 1000)/x) - 1000 \tag{7.1}$$

7.6.4. Cálculo de ppm

Con los puntos obtenidos de la Tabla 7.1. Una vez estos puntos fueron obtenidos, se utilizó el Script: de R: de acceso público propuesto y desarrollado por el Ing. Davide Gironi. Dicho script

tiene el fin de automatizar el proceso de recolección de datos de la gráfica de curva de sensibilidad, tales como: R0, los valores a y b que son sustituidos de la ecuación 4.5. El acceso a este script puede encontrarse en la parte de anexos.

El script toma como parámetros los siguientes valores:

Variable	Valor	Significado
ppm	0.001	Concentración (en ppm) conocida de gas
Rs	8000	Resistencia (en ohms) a una concentración de gas
Puntos de recta	Tabla 7.1	Puntos de la curva del fabricante (en *string* por ",")

Tabla 7.2: Valores de entrada para script R

Se obtuvo la curva y su respectiva ecuación que representa los valores de concentración de ppm en condiciones ambientales 20°C y 60 % humedad relativa:

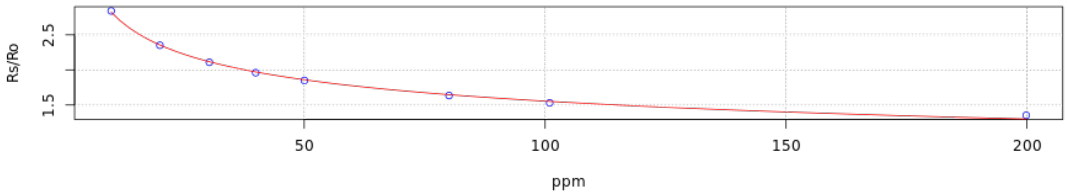


Figura 7.1: Gráfica de ppm - Rs/R0 para sensor MQ135

Los puntos azules indican los valores del fabricante, y la curva roja indica la función que describe esos puntos. Se observó que la curva pasa por todos los puntos, por lo tanto los valores retornados por el algoritmo son sustituidos en la ecuación 4.5. La ecuación de la curva es dada por:

$$ppm(Rs) = 764.4864 * (Rs/2603)^{-4.525248} \tag{7.2}$$

7.6.5. Determinando el factor de corrección

El fabricante del sensor especifica que los factores ambientales tales como temperatura y humedad tienen influencia en los valores de RS/R0, por lo que la ecuación descrita anteriormente 7.2, deja de ser válida conforme se alejan las condiciones ambientales de 20°C y 60 % humedad relativa. Los valores de temperatura y humedad son obtenidos por el sensor DHT22. Para determinar el cambio entre Rs/R0, el datasheet aporta la siguiente gráfica. [39]

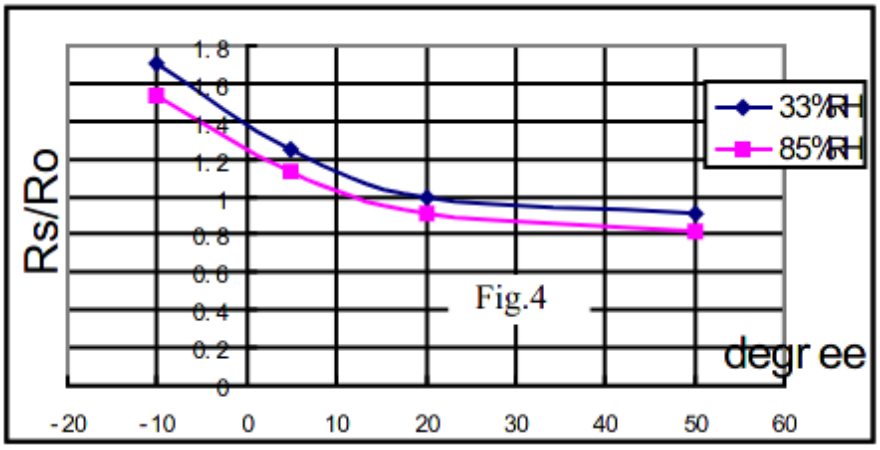


Figura 7.2: Gráfica Rs/R0 - temperatura - humedad para sensor MQ135

La siguiente tabla define la nomenclatura utilizada para las variables que fueron requeridas para realizar los cálculos del factor de corrección.

Variable	**Descripción de variable**
Rs_{TxHx}	Resistencia medida por el sensor a una temperatura y humedad determinada
$R0_{TxHx}$	R0 a una temperatura y humedad determinada

Tabla 7.3: Tabla de variables relevantes en la corrección de RS/R0

Se replicó la metodología de ajuste de temperatura y humedad para Rs/R0. Esta vez se correlacionó Rs y R0 a nuevas condiciones de temperatura y humedad, para lo cual, Gironi propone la existencia de una proporción entre Rs a 20°C y 60 % de humedad relativa y Rs a una temperatura X en una humedad relativa X, a distintas concentraciones del mismo gas, mostrado en la Figura 7.3.

Basado en el artículo de Gironi, donde propone encontrar la relación entre la temperatura actual para ambas curvas de la Figura 7.2, la relación se calculó de la siguiente forma:

$$m = Rs_{TxHx}/Ro_{20C33} \tag{7.3}$$

$$n = Rs_{T20H65}/Ro_{T20H33} \tag{7.4}$$

Se utilizaron las ecuaciones 7.3, 7.4 y 4.6 de las cuales se obtuvo:

$$R0_{HxTx} = (n/m) * Rs_{TxHx} * (a/ppm)^b \tag{7.5}$$

La siguiente lista de pasos indica la computación del factor de corrección:

- Se encontró la proporción entre las curvas de 33 % y 85 % de humedad, con la temperatura actual y la temperatura de referencia, en las ecuaciones 7.4 y 7.3.
- El valor encontrado, es el factor de corrección.
- Se agregó el término del factor de corrección a R0 en la ecuación 7.5 y se realizó el despeje para ppm y substitución de variables ya obtenidas en la sección anterior.

De esta forma, la ecuación 7.2 pasa a tener la forma de:

$$ppm_{TxHx} = (m/n) * 764.4864 * Rs_{TxHx} * (a/2603)^{-4.525248} \tag{7.6}$$

Dicha ecuación es utilizada para reportar las concentraciones de CO, ya tomando en cuenta los factores ambientales de temperatura y humedad como describe finalmente Gironi. [31].

Este proceso es repetido de la misma forma para el sensor MQ131, los valores para las ecuaciones cambian, pero la metodología no varía debido a que los sensores funcionan exactamente igual.

7.7. Comunicación y configuración sensor SPEC DGS-SO2

El sensor SPEC utilizado es la versión digital, por lo que este se conectó por medio de UART. El fabricante menciona en el datasheet del mismo que el sensor, al igual que los sensores de la familia MQ es afectado por factores de temperatura y humedad. El sensor está equipado con sus propios sensores de temperatura y humedad, según se menciona en el datasheet esto ya es tomado en cuenta, es decir aplicado un factor de corrección con base a los valores de temperatura y humedad recopilados por el mismo.[68].

Para reportar efectivamente los datos obtenidos por el sensor se realizó la siguiente serie de pasos:

- Calentar el sensor por 1 hora, enviar el caracter c
- Si se considera el aire no contaminado, calibrar a cero el sensor, enviando el caracter Z
- Después de la hora de calentamiento, los datos obtenidos por el sensor ya son válidos y se puede comenzar a reportar.

7.7.1. Calibración

Para calibrar el sensor, si se cuenta con el módulo USB, en un ambiente Windows, el sensor aparece con el nombre Silicon Labs UART USB:. Es necesario tener instalado el controlador UART CP210X instalado para que sea reconocido. Se procede a reiniciar la computadora y ejecutar el proveído por el fabricante. En la sección de recursos utilizados se dejará el enlace del programa utilizado.

En el administrador de dispositivos de Windows se observó a que puerto COM estaba utilizando el sensor, este se encontró listado con el nombre Silicon Labs CP210x USB to UART Bridge (COM4).

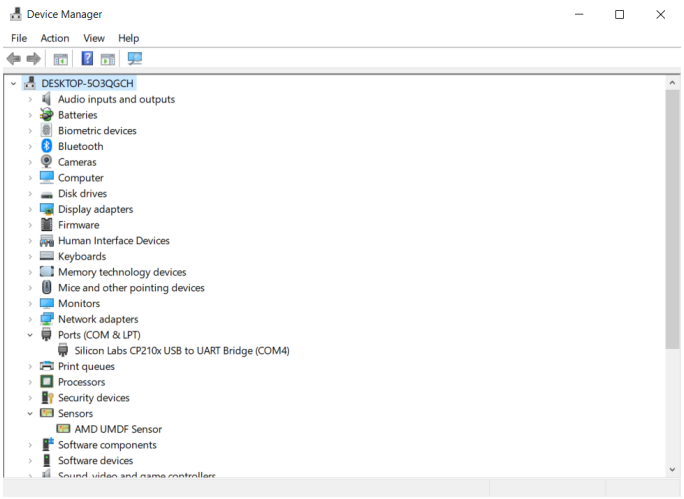


Figura 7.3: Interfáz Administrador de dispositivos

Una vez se encuentra en la interfaz gráfica del programa DSDK, se procedió:

- Open COM Port
- En input de set average, se colocó el valor 300 y luego click en set average.
- Click en Continous Read

En el proceso anterior se dejó el sensor por 12 horas continuas, y al día siguiente por otras 12 horas continuas. Finalmente, se hizo click en Set Zero, en la Figura 7.4 se puede observar la interfaz del programa.

Es importante notar que el sensor tiene la particularidad de mostrar valores altos (e incorrectos) en ambientes cerrados, por lo que es recomendable calibrar dicho sensor en condiciones abiertas, esto también aplica para el sensor ULPSM SO2 que será discutido en la siguiente sección.

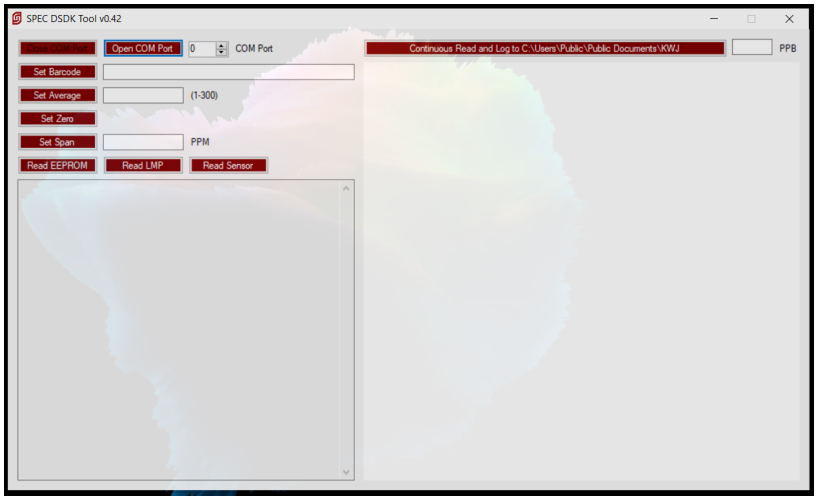


Figura 7.4: Interfaz DSDK, para sensores SPEC

7.7.2. Cálculo ppm con sensor SPEC DGS-SO2

El sensor SPEC provee en una cadena de caracteres la información de las mediciones obtenidas por el mismo, cada una separada por comas. Por lo que reportarla solo requirió partir dicha cadena, acceder al segundo elemento de la cadena y dividirlo entre 1000, puesto que el valor reportado es en ppb (partes por billón).

7.8. Cálculo ppm con sensor SPEC ULPSM-SO2

El sensor SPEC ULSPM-SO2 no es utilizado en este proyecto, sin embargo este fue planificado para utilizarse. Este modelo no fue utilizado por problemas de envío fue recibido el sensor DGS-SO2. A pesar de este improvisto, se lista como se implementaría el sensor ULSPM SO2.

Según es mencionado, existen tres pines tienen distintas señales:

- Pin #1: V_{gas}
- Pin #2: V_{ref}
- Pin #3: V_{temp}

7.8.1. Calibración

El sensor debe ser calibrado a 0 ppm, para ello se deja conectado el sensor por al menos una hora, a partir de esa hora se tomarán 300 muestras de voltaje en un ambiente de aire limpio. De estas 300 muestras se ejecuta un promedio y se obtiene V_{gas0}.

7.8.2. Cálculo de concentración de gas (ppm)

Utilizando la ecuación propuesta por el datasheet, se determina la concentración del gas:

$$C_x = 1/M * (V_{gas} - V_{gas0}) \tag{7.7}$$

Sin embargo, de momento se desconoce el valor M, el cuál es calculado por la siguiente ecuación.

Del mismo datasheet se obtiene la información del sensor:

Variable	Valor
TIA Gain (SO2)	100
Sensivity Code	26.01

Tabla 7.4: Información relevante del sensor

Nota: Cada sensor tiene su propio código de sensibilidad. Este es obtenido en el sticker en la parte trasera de cáda sensor, o en el código QR en el que viene empacado el sensor vendrá el customer part number, donde se debe consultar el código de sensitividad al correo de sales@spec-sensors.com.

Figura 7.5: Sticker de información del sensor

$$M = SensivityCode * TIAGain * 1 * 10^{-9} * 10^{3} \tag{7.8}$$

Una vez se cuenta con la información para el cálculo de M, se puede computar 7.7.

El fabricante menciona que los factores ambientales como temperatura afectan el cero del sensor, por lo que para mejorar las medidas de ppm, es necesario aplicar un factor de corrección tanto al cero del sensor como a la sensibilidad del mismo. [67]

Para ello, se cita los offsets a tomar en cuenta del datasheet.

Temperature Coefficient of Span (%/°C) ***(Typical)***	-20 °C to 20 °C	-0.33%/°C
	20 °C to 40 °C	0.26%/°C
Temperature Coefficient of Zero Shift (ppm/°C) ***(Typical)***	-20 °C to 0 °C	0.012 ppm/°C
	0 °C to 25 °C	0.056 ppm/°C
	25 °C to 40 °C	0.46 ppm/°C

Figura 7.6: Offsets mencionados por el fabricante del ULPSM-SO2

Adicionalmente, al código de sensibilidad se le resta, o suma el porcentaje del sensivity code que está en la tabla. Luego de estos pasos, el cálculo de concentración sigue siendo el mismo 7.7 pero los valores de M y cero han sido corregidos tomando en cuenta el factor ambiental de temperatura.

7.9. Implementación de MICS-6814

El sensor MICS-6814 implementado por Seeedstudio, llamado Grove Multichannel Gas Sensor provee información sobre múltiples gases, se utilizaron las mediciones de CO y NO2. La comunicación entre el SoC y el sensor es manejada a través de I2C, esto por diseño del fabricante de la placa [1].

7.9.1. Calibración

En la página web del fabricante se referencia el repositorio oficial para controlar el sensor. Se utilizó el programa *calibration.ino* con el fin de poder calibrar las lecturas reportadas. Se decidió utilizar este programa puesto que supone una reducción en el tiempo de desarrollo. No se podrá utilizar el ESP32 donde se ya se han implementado el resto de sensores debido a que este ya está flasheado con firmware de Micropython. En este caso, se utilizó un Arduino Uno y se ejecutó el programa como es recomendado por el fabricante, configurando la constante PRE_HEAT_TIME a 30, lo cual equivale a 30 minutos de calentamiento. Para la comunicación del sensor, se tuvo que utilizar una función en desuso I2C, en lugar de SoftI2C puesto que no se podía lograr comunicación con el sensor, esto fue descubierto al implementar un script desarrollado por Paul Spooren, el acceso al mismo puede encontrarse en la sección de recursos utilizados. Finalmente, luego de obtener los resultados de R0 para cada uno de los gases, se prosiguió a la toma de datos.

7.9.2. Cálculo de concentración de gas (ppm)

El archivo *MutichannelGasSensor.h* contiene información sobre los distintos segmentos de memoria los cuales contienen los valores de R0 para cada gas y resistencias configuradas para el sensor. En el programa *MutichannelGasSensor.cpp* se observó como realizar los cálculos pertinentes para obtener la concentración del gas objetivo, esto es sugerido por el fabricante y estas funciones serán utilizadas para reportar las concentraciones de CO y NO_2. Las funciones que fueron utilizadas para el cálculo de concentración fueron las siguientes:

$$ppmNO2(Rs/R0) = (Rs/R0)^{1.007}/6.855 \tag{7.9}$$

$$ppmCO(Rs/R0) = (Rs/R0)^{-1.179} * 4.385 \tag{7.10}$$

Los valores Rs/R0 para cada gas son calculados según los valores obtenidos en la calibración y los datos reportados por el ADC del sensor, en este caso se calcula según la siguiente ecuación, que el fabricante define:

$$RS/R0 = Rs/R0_{gas} * (1023.0 - R0_{gas})/(1023.0 - Rs) \tag{7.11}$$

Computando la ecuación 7.11 para los casos de NO_2 y CO permite obtener el valor a ingresar a las funciones 7.9 7.10 finalmente se obtiene el valor de ppm. El fabricante no menciona influencias de ambiente en el datasheet por lo cual temperatura y humedad no fueron tomadas en consideración para efectuar el cálculo de factor de corrección.[69].

El valor de ppm reportado es un promedio entre los valores obtenidos por el sensor MICS-6814 y el sensor MQ135.

7.10. Ambiente de calibración

Idealmente, todos los gases que se están midiendo deberían ser 0ppm dado que la presencia de ellos es considerado contaminación del ambiente. [20].

Se decidió usar este ambiente donde se calibrarían los sensores. Esto fue realizado para poder, en un ambiente de jardín exterior a una hora similar (9 am - 10 am) cada vez que estos sensores

eran configurados, con el fin de tener un proceso estándar de medición. A continuación se muestra el lugar donde se realiza la calibración a cero de los sensores.

Figura 7.7: Ambiente de calibración

7.11. Conexión a Internet y Azure IoT Hub

El dispositivo se conecta a internet a través de una red Wi-Fi, en este caso se utilizó el celular con su función Hotspot: para poder compartir internet al ESP32. Para conectar el sensor a la plataforma IoT Hub en Azure, el cual fue implementado por el módulo complementario a este trabajo. El código ejemplo a utilizar fue el proporcionado por Azure-Samples, el cual fue obtenido de la documentación, se requirió un *SAS TOKEN:* para poder ingresar a la infraestructura, estos SAS token tienen una expiración la cual es determinada por el administrador de la infraestructura. La implementación del código fue extraída y adaptada según las necesidades del proyecto, el acceso a este código estará en la sección de recursos utilizados.

7.12. Conexión entre sensores

Se utilizó una Protoboard: para conectar los cinco sensores al microcontrolador ESP32. Para la conexión se utilizaron cables jumpers Existieron casos particulares como el sensor MQ131 y MICS-6814 para que pudieran ser conectados al ESP32.

En el caso del sensor MICS-6814, este requirió cortar los cables originales y soldarlos a jumpers machos para conectarlo a las líneas del protoboard. Esto se debió al grosor de los pines originales que eran muy delgados para hacer contacto correctamente con el protobard.

De forma similar, el sensor MQ131 tiene una conexión compatible, pero que requirió de agregar jumpers macho a macho, puesto que este sensor tenia conexión hembra-hembra. Esto no requirió soldar los cables, únicamente se necesitó agregar cables entre el microcontrolador y el sensor.

Nombre de sensor	Medición	Costo (USD)
MQ135	CO	$2.00
MQ131	O3	$49.00
ULPSM-SO2	SO2	$50.00
MICS-6814	NO2, CO	$38.40
DHT22	Temperatura y humedad	$10.99

Tabla 7.5: Sensores implementados, funcionalidad y costos

A continuación se muestra la esquemática de conexión de los sensores en el protoboard.

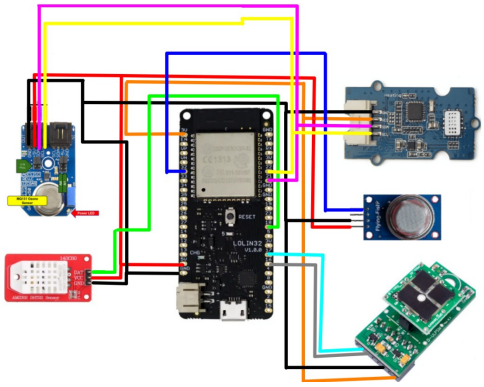


Figura 7.8: Esquemática de conexión

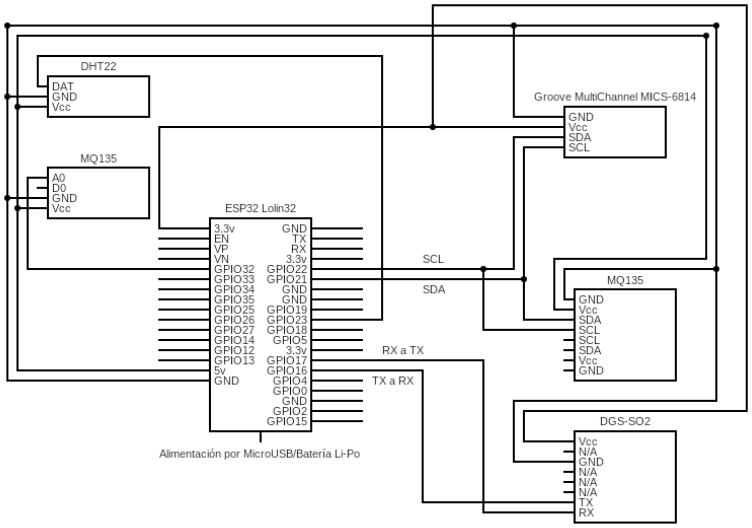


Figura 7.9: Diagrama de conexión técnico

Color	Tipo de señal
Negro	GND
Rojo	5V
Verde	DAC, Pin 18
Amarillo	SCL
Lila	SDA
Azul	DAC, Pin 32
Naranja	3V
Gris	TX
Celeste	RX

Tabla 7.6: Códigos de colores y sus significados

7.13. Validación de datos obtenidos

Para la validación de datos, se realizó experimentos de emisión de CO, manteniendo encendido un carro Yaris, modelo 2007. Se realizaron dos pruebas, colocando el sensor a aproximadamente a 3 metros del carro, a nivel del suelo con un portón cerrado, en un ambiente abierto. La segunda prueba fue realizada con el portón abierto, en el mismo ambiente. Los datos fueron comparados con un sensor Dräger Pac 5500 de uso profesional.

Cabe destacar que debido a la falta de equipo especializado, no se pudo validar los demás gases medidos.

$$Err\,\% = (V_{real} - V_{res})/(V_{real} * 100) \tag{7.12}$$

La ecuación 7.12 fue utilizada para determinar los porcentajes de error cuando se comparó el sensor MQ135 con el sensor Dräger 5500.

A continuación se incluyen imágenes comparando los resultados obtenidos con el sensor MQ135, y un sensor de uso profesional.

7.14. Recopilación de datos y pruebas piloto

Una vez la estación fue armada y configurada, se procedió a la toma de datos en los tres lugares a los que está delimitado este proyecto: Centro Histórico de la ciudad de Guatemala, Carretera el Salvador, y Mixco. Se definió un estándar de prueba, el cual sería implementado en los tres lugares mencionados. El tiempo de medición de cada prueba fue de alrededor de 2 horas y media. Estas pruebas buscaron determinar la estabilidad del sensor, bajo condiciones óptimas: Red Wifi y alimentación eléctrica estable a través de conexión a una computadora. Adicionalmente se probó con un powerbank de 10,000 mA.

La estación de medición fue probada en: ambiente cerrado, ambiente abierto, y en movimiento en un ambiente abierto. Los ambientes que fueron seleccionados respectivamente fueron: sótanos, una avenida o calle principal y en tráfico, con la estación midiendo afuera del carro.

A continuación se muestra fotografía de la recopilación de datos para las pruebas piloto, en Mixco.

Figura 7.10: Recopilación de datos en Mixco, prueba al aire libre

7.15. Contenedor de la estación

Debido a la naturaleza de los sensores, la estación de medición debe un espacio de entrada de aire para que esta pueda tomar medidas adecuadamente, adicionalmente que tuviera fácil acceso y cerrado fácil para el contenedor.

- Tomar medidas del protoboard
- Búsqueda / adaptaciones de diseños existentes
- Impresión del contenedor

7.15.1. Toma de medidas

Antes de buscar un contenedor que pudiera adecuarse a las necesidades del proyecto, se tomaron las medidas de la estación de medición, desde su tamaño en protoboard, las medidas obtenidas fueron:

- Largo: 21.3cm
- Ancho: 11.6cm

Con estas medidas, se estableció un contenedor de 28cm * 17 cm * 11 cm. Debido a las dimensiones de este contenedor y el tiempo que requeriría imprimir uno fue decidido utilizar MDF (Medium Density Fibreboard). Este material tiene como desventaja que no resiste a los elementos como el agua, por lo que se tuvo que planear una fase de impermeabilizar dicho contenedor.

7.15.2. Búsqueda de diseños existentes

El contenedor de la estación fue obtenido de internet según las necesidades planteadas. La página de diseño MakerCase fue utilizada para obtener un diseño rápido. Se utilizó el diseño de Caja básica o BasicBox como le llama el programa, y se configuraron las siguientes medidas: 280 * 170 * 110 milímetros (interior), con un grosor de material de 3 milimetros, con unión de borde configurada a la opción de dedos.

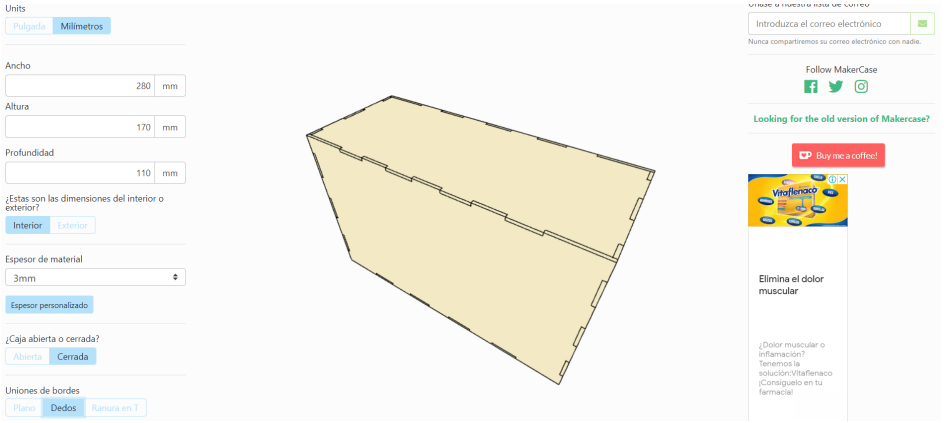


Figura 7.11: Construcción de contenedor en MakerCase

7.15.3. Modificaciones al diseño

Como se puede observar, el contenedor no tiene de entrada de aire, para resolver este problema, fue necesario utilizar InkScape. Para esta sección, se requirió la asistencia por una colaboradora de la Universidad del Valle de Guatemala, una vez fue agregado al diseño dichas aperturas se procedió al proceso de impresión. Utilizando este programa, se le realizaron aperturas al contenedor.

Figura 7.12: Contenedor impreso

7.15.4. Ensamblaje e impermeabilización

El ensamblaje de este material requirió utilizar goma blanca de carpintería y dejar reposar por cuatro horas, luego de esto se prosiguió al proceso de impermeabilización. Dicho proceso cual requirió de barniz y un catalizador. La mezcla utilizada para el barniz un vaso de ocho onzas y medio vaso de catalizador, aproximadamente cuatro onzas. Se procedió a mezclar durante dos minutos y se tiene que observar una consistencia viscosa del líquido. Con esto se utilizó una brocha y se comenzó a aplicar la mezcla sobre el contenedor, se dejó secar en un ambiente libre de polvo, debido a que el polvo puede dañar la calidad del proceso de impermeabilización. Luego de dos horas, se prosiguió a agregar otra capa de esta mezcla de barniz, este proceso fue repetido tres veces y se dejó en secado.

7.16. Diseño de la infraestructura

7.16.1. Definición de proveedor de servicios en la nube

Para definir la infraestructura en la nube que soportaría la plataforma, era necesario hacer un comparativo de los proveedores de servicios en la nube disponibles, analizar sus diferencias y similitudes y cuál oferta de servicios se adaptaba mejor a los propósitos de este proyecto. Para esto se tomaron en cuenta las siguientes características:

- Costo de cada herramienta a utilizar
- Facilidad de acoplamiento de las herramientas con los demás módulos

- Curva de aprendizaje para el manejo de las herramientas

De todos los proveedores existentes, se establecieron como candidatos aquellos que fueran catalogados como líderes en la industria: Amazon Web Services, Microsoft Azure, Google Cloud Platform.

A continuación, se detallan estas características para cada proveedor escogido y la selección definitiva del proveedor a utilizar.

Oferta de servicios

Para la selección del proveedor, se realizó una lista sobre las herramientas o servicios que cada uno ofrecía, la cantidad y complejidad de estas y qué tan bien se podían acoplar con el resto de los módulos. Se necesitaría que las herramientas en conjunto provean las tres principales:

1) Conexión a dispositivos en la nube (IoT)
2) Procesamiento de los datos de manera distribuida, preferiblemente con soporte a Spark.
3) Almacenamiento de datos, preferiblemente datos estructurados o SQL.

Amazon Web Services

En Amazon Web Services, el flujo de datos streaming a lo largo de sus herramientas sería el siguiente:

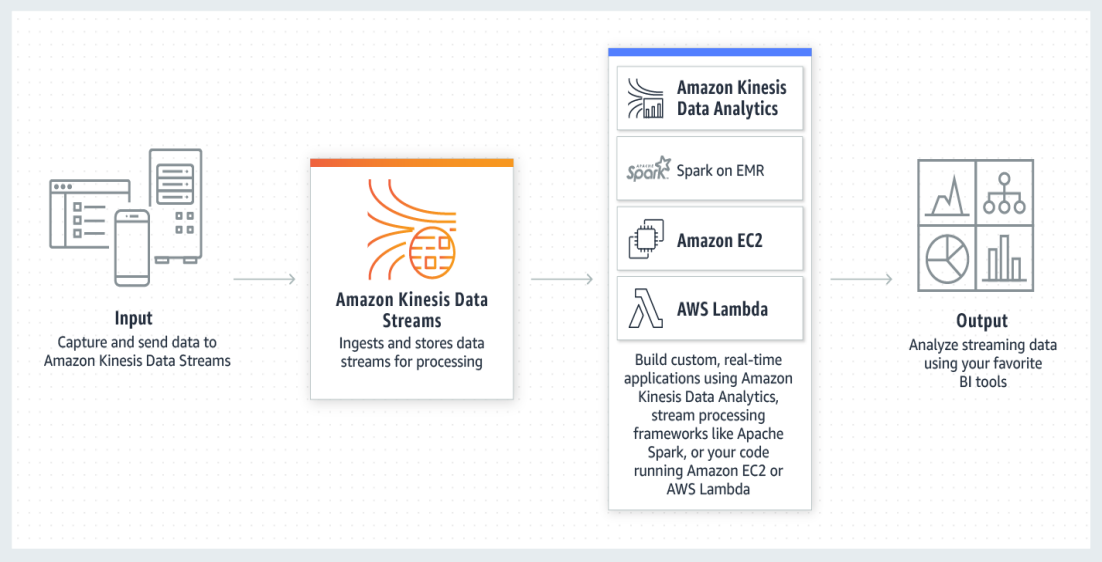


Figura 7.13: Flujo de datos streaming en Amazon Web Services.

- **Conexión a dispositivos en la nube (IoT)**

  En el caso de Amazon, este tiene una *Suite*: general de servicios que engloban las funcionalidades de *Data streaming*:, el cual es Amazon Kinesis. Amazon Kinesis Data Streams (o KDS por sus siglas en inglés), es un servicio de streaming de datos en tiempo real con un alto nivel de escalabilidad y durabilidad. Es capaz de registrar de manera continua un alto volumen de

datos por segundo provenientes de múltiples orígenes; estos datos pueden estar disponibles en cuestión de milisegundos para casos de uso de análisis en tiempo real. Aunque no aparece en la figura, previo a Kinesis estaría AWS IoT Core, que sería la herramienta que habilitaría la conexión entre los dispositivos con el resto de la plataforma en la nube.

- **Procesamiento de datos**

  Amazon EMR es una plataforma de administración de clústers, utilizada para procesar y analizar grandes cantidades de datos de manera distribuida. En esta herramienta se tiene soporte para marcos de trabajo como Apache Hadoop y Apache Spark. Esta herramienta sería la que realizaría el procesamiento de datos.

- **Almacenamiento de datos**

  Amazon EMR permite transformar y trasladar grandes cantidades de datos hacia y desde otros almacenes de datos o bases de datos. En AWS, la herramienta designada para la administración de bases de datos SQL en cualquiera de sus distribuciones (como MariaDB, PostgreSQL, etc) es Amazon Relactional Database Service o RDS.

Microsoft Azure

El flujo completo de datos que se manejaría con Azure sería el siguiente:

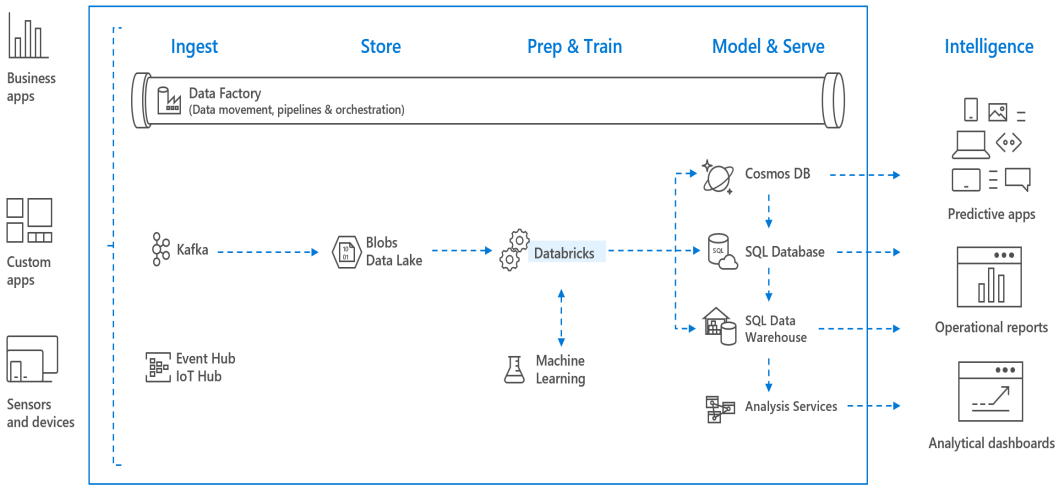


Figura 7.14: Flujo de datos streaming en Microsoft Azure.

- **Conexión a dispositivos en la nube (IoT)**

  IoT Hub es un servicio manejado en la nube que actúa como una central de comunicación entre los dispositivos IoT enviando mensajes y las aplicaciones que consumen estos datos. Tiene alta flexibilidad de conexión, tanto en los dispositivos que se conectan como en los protocolos de envío de mensajes [50]. Este servicio sería el que se usaría para conectar los sensores al resto de la plataforma en la nube.

- **Procesamiento de datos**

  Azure Databricks es una plataforma de análisis de datos optimizada para fácil acoplamiento con los demás servicios de Azure. Ofrece tres entornos principales: SQL, Data Science Engineering

y Machine Learning. Data Science Engineering – que sería el entorno de utilidad al proyecto – es una plataforma de análisis basada en Apache Spark. Provee las siguientes funcionalidades: clústeres de Spark completamente administrados, áreas de trabajo interactivas de exploración y visualización de datos, una plataforma de ejecución de aplicaciones Spark [55]. Esta sería la herramienta de procesamiento de datos.

- **Almacenamiento de datos**

  SQL Database es un motor de base de datos completamente administrado, se encarga de la mayoría de las funciones administrativas como actualizar, aplicar revisiones, crear copias de seguridad y supervisión, todo sin intervención del usuario. Está basado en la versión estable más reciente de Microsoft SQL Server [53]. Esta sería la herramienta de almacenamiento de datos y también tendría la conexión con el servidor web del siguiente módulo.

Google Cloud Platform

Para Google Cloud Platform o GCP, el flujo de datos para aplicaciones streaming sería el siguiente:

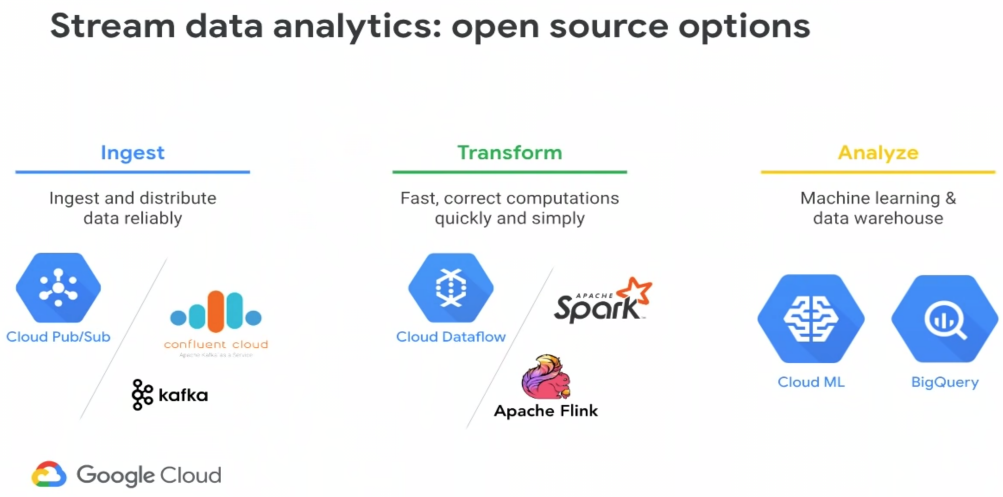


Figura 7.15: Flujo de datos streaming en Google Cloud Platform.

- **Conexión a dispositivos en la nube (IoT)**

  Cloud Pub/Sub es una herramienta de mensajería que permite que distintos servicios se comuniquen de forma asíncrona, con latencias de alrededor de 100 milisegundos [34]. No está diseñado específicamente para dispositivos físicos, si no para servicios en general que se deseen que envíen datos de un lado a otro de manera continua. Aunque no aparece en el diagrama, se debe usar la herramienta de Cloud IoT Core, que es donde se registran los dispositivos para habilitar la comunicación entre los sensores, Pub/Sub y el resto de la plataforma.

- **Procesamiento de datos**

  Dataflow es un servicio dedicado a ejecutar una amplia variedad de patrones de procesamiento de datos, para transformar y enriquecer datos en tiempo real (streaming) e histórica (batch). Entre sus funciones principales se encuentran: ajustar la capacidad de procesamiento de cada trabajador de manera dinámica, ajusta de manera integrada la escala automática vertical con

la horizontal, sincronizar o replicar datos de manera confiable y con latencia mínima [36]. Esta herramienta sería la destinada para el procesamiento de datos.

- **Almacenamiento de datos**

  BigQuery es el almacén de datos rentable, a escala de petabytes y completamente administrado por Google Cloud. Permite ejecutar estadísticas en grandes cantidades de datos con baja latencia [35]. Esta sería la herramienta de almacenamiento de datos y conexión con el servidor del siguiente módulo, que sería la página web.

Comparativa de precios

Siguiendo la investigación de los servicios relevantes al proyecto que cada uno de los proveedores tenía por ofrecer, se procedió a generar la estimación de costos que tendría el ciclo de vida del proyecto dentro de la nube. En todos los casos, los precios presentados son de periodicidad mensual, utilizando la metodología de pago de *pay-as-you-go*, suponiendo que los servicios estarán en funcionamiento todo el día, todos los días del mes y utilizando la calculadora de precios de cada proveedor. A continuación, se presentan los resultados de la estimación de costos:

Group hierarchy	Region	Description	Service	Upfront	Monthly	First 12 months total	Currency	Configuration summary
My Estimate	US East (Ohio)		Amazon Kinesis Data Streams	0	0	0	USD	Number of records (2 per minute), Number of Consumer Applications (1)
	US East (Ohio)		Amazon EMR master node on EC2	0	31.536	378.43	USD	Number of master EMR nodes (1), EC2 instance (c6gn.xlarge), Utilization (730 Hours/Month)
	US East (Ohio)		Amazon EMR core node on EC2	0	63.072	756.86	USD	Number of core EMR nodes (2), EC2 instance (c6gn.xlarge), Utilization (730 Hours/Month)
	US East (Ohio)		Amazon EC2	0	262.29	3147.48	USD	Operating system (Linux), Quantity (2), Pricing strategy (On-Demand Instances), Storage amount (50 GB), Instance type (c6gn.xlarge)
	US East (Ohio)		Amazon RDS for SQL server	0	60.87	730.44	USD	Storage for each RDS instance (General Purpose SSD (gp2)), Storage amount (250 GB), Number of nodes (1), Instance type (db.t2.small), Utilization (On-Demand only) (730 Hours/Month), Deployment option (Single-AZ), License (License included), Database edition (Express), Pricing strategy (OnDemand)
				Total	417.768	5013.21		

Acknowledgement

* AWS Pricing Calculator provides only an estimate of your AWS fees and doesn't include any taxes that might apply. Your actual fees depend on a variety of factors, including your actual usage of AWS services.

Figura 7.16: Estimación de costos por servicio generado para Amazon Web Services.

Microsoft Azure Estimate

Your Estimate

Service type	Custom name	Region	Description	Estimated monthly cost
Azure Databricks		East US	Jobs Compute Workload, Standard Tier, 2 F4SV2 (4 vCPU(s), 8 GB RAM) x 730 Hours, Pay as you go, 1.00 DBU x 730 Hours	$410.99
Azure IoT Hub		East US	Standard Tier, Free: 500 devices, 8,000 msgs/day, $0.00/mo, 1 IoT Hub Units	$0.00
Azure SQL Database		East US	Single Database, DTU Purchase Model, Standard Tier, S0: 10 DTUs, 250 GB included storage per DB, 1 Database(s) x 730 Hours, 5 GB Retention	$14.72
Support			Support	$0.00
			Licensing Program	Microsoft Customer Agreement (MCA)
			Billing Account	
			Billing Profile	
			Total	$425.71

Disclaimer

All prices shown are in United States – Dollar ($) USD. This is a summary estimate, not a quote. For up to date pricing information please visit https://azure.microsoft.com/pricing/calculator/
This estimate was created at 10/21/2021 5:09:35 PM UTC.

Figura 7.17: Estimación de costos por servicio generado para Microsoft Azure.

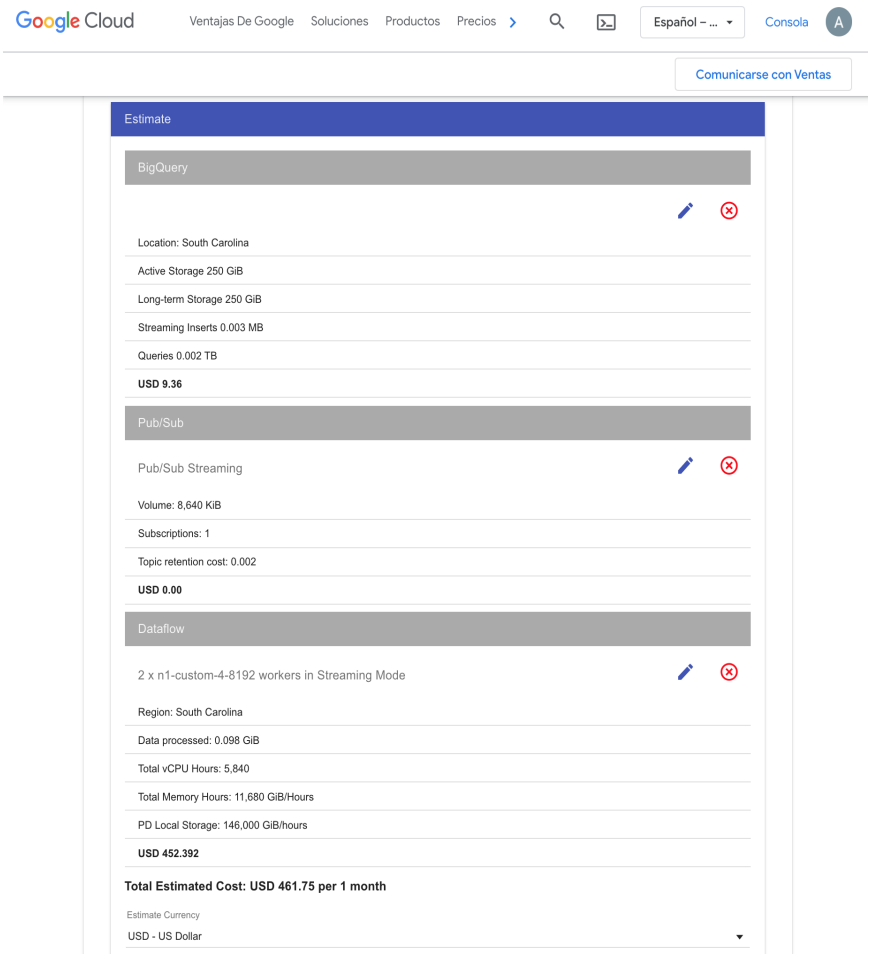


Figura 7.18: Estimación de costos por servicio generado para Google Cloud Platform.

En orden ascendente, los costos totales para cada una de las plataformas son: USD $425.71 para Microsoft Azure, USD $461.75 para Google Cloud Platform y USD $417.68 para Amazon Web Services. Cabe resaltar que, para cada GCP y Azure, existen "créditos" disponibles para cuentas primerizas; en el caso de GCP, nuevas cuentas son otorgadas $300 válidos por 90 días para explorar la plataforma [37]. Azure tiene USD $200 de créditos válidos por un mes adicionalmente de servicios populares gratuitos por 12 meses y más de 25 servicios que son gratis siempre [51]; a parte de los USD $200 dados a toda cuenta primeriza, Azure tiene disponible un programa de Azure for Students, el cual provee USD $100 adicionales de crédito válidos por un año, los cuales son renovables cada año, siempre que el usuario sea parte de la organización educativa.

Facilidad de acoplamiento de las herramientas con los demás módulos

Este módulo es el punto intermedio entre el segmento de Internet de las Cosas – es decir, los sensores que realizan las mediciones – y el segmento de la plataforma web. Debido a esto, es importante que las herramientas del proveedor de servicios en la nube que se escoja tengan disponible una conexión fácil desde tecnologías fuera de la nube. El siguiente cuadro describe los aspectos necesarios de las conexiones que se deben realizar a la plataforma en la nube, tanto del lado del Internet de las Cosas como de la plataforma web.

		Azure	AWS	GCP
Sensores o Internet de las Cosas	Forma de conexión	Conexión a IoT Hub a través de un *connection string*. Autenticación por un *SAS token* generado.	Conexión a AWS IoT Core a través de un *Message broker*: suscrito a un *topic*[1]. Autenticación por pares de llaves criptográficas: Se usa un objeto proveedor de credenciales.	Se necesita crear un registro de dispositivos para poder conectar los sensores. Autenticación por pares de llaves criptográficas.
	Librerías externas para Python 3.9	-	awsiotsdk, awscrt	pyjwt, paho-mqtt, cryptography
Plataforma web	Forma de conexión	Conexión directa a base de datos SQL a través de un *connection string*. Solo se necesitan enviar SQL *queries*. Registro de IP de origen al servidor de SQL Server.	Conexión directa a AWS RDS. Solo se necesitan enviar SQL *queries*.	Conexión es a través de un cliente que envía un objeto tipo JSON con un SQL *Query*: parámetros y una región definida para solicitar datos a BigQuery. Autenticación a través de un *service account* con credenciales asignadas.
	Librerías externas para Express.js	mssql	aws-sdk/client-rds node-mssql	google-cloud/bigquery

Tabla 7.7: Características de la conexión para cada módulo por proveedor.

7.16.2. Infraestructura final

Teniendo en cuenta la información encontrada – resumida en las secciones anteriores – el proveedor de servicios en la nube considerado como el mejor adaptado a las necesidades y características de este proyecto es Microsoft Azure, por las siguientes razones:

1. Aunque Azure no reporta el menor costo total por mes – la opción más económica siendo AWS –, la diferencia entre costos es mínima y, por otro lado, Azure tiene el beneficio de créditos que se pueden utilizar en la plataforma; aunque estos son temporales y limitados, se consideró que eran suficientes para el desarrollo de este proyecto.
2. El acceso a la infraestructura montada en Azure por parte de los dos módulos que lo rodean es la más simple: se deben instalar la menor cantidad de librerías para poderse usar y la conexión es realizada a través de connection strings que ya provee cada uno de los servicios dentro de la plataforma – es decir, no es necesario que sean generados manualmente.
3. Azure puede ejecutar el pipeline de procesamiento de datos con la menor cantidad de herramientas, lo que significa que se debe aprender sobre el funcionamiento de menos cosas y la curva de aprendizaje disminuye.

Por último, se diseñó la infraestructura que soportaría el proyecto en Azure, la cual se resume en la siguiente figura.

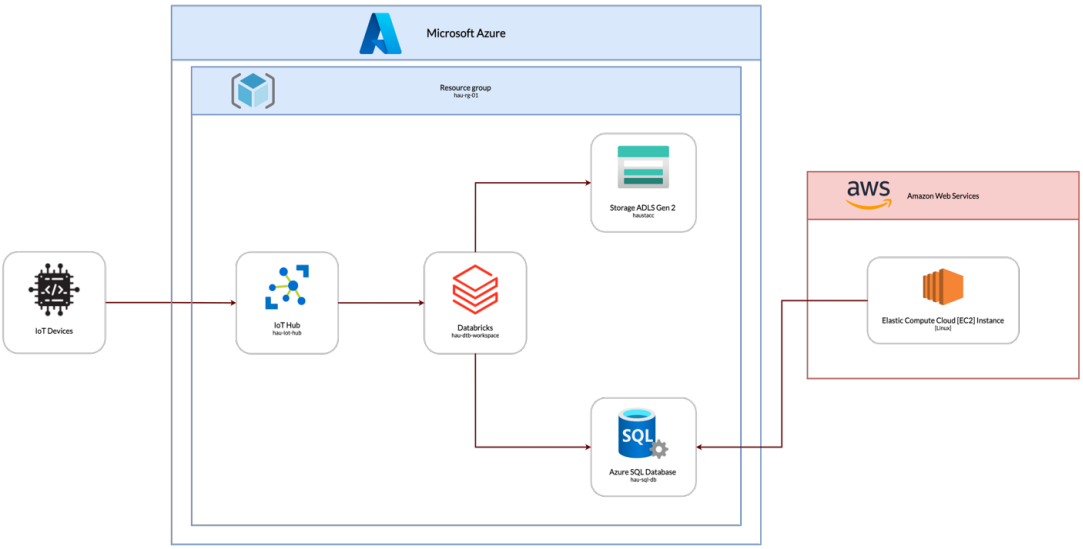


Figura 7.19: Recursos que serán desplegados en las plataformas en la nube.

En la figura anterior, se observa los distintos componentes o recursos que estarían en funcionamiento dentro de Azure, y cómo se relacionan entre sí. El flujo de la información es el siguiente:

1. Los sensores envían las mediciones de los contaminantes establecidos en intervalos de 30 segundos.
2. IoT Hub recibe la información y la envía a las herramientas conectadas al Hub.
3. Databricks ingesta la información de IoT Hub y realiza el procesamiento necesario de limpieza de datos y cálculo del AQI por ventanas de 1 y 8 horas – dependiendo del contaminante.

4. La información procesada se almacena en la base de datos SQL Server.
5. La aplicación web del módulo siguiente está alojada en una máquina virtual de AWS – se conecta a la base de datos SQL para obtener los datos sobre las mediciones y desplegarlos gráficamente al usuario.

Cada una de las herramientas en Azure se creó con los siguientes límites de recursos:

- IoT Hub: F1 (*Free tier*) con un máximo de 8,000 mensajes al día, un tamaño máximo de 0.5 KB por mensaje y un máximo de una unidad IoT Hub. Esta configuración es gratuita.
- Databricks: un *Clúster*: con escalabilidad dinámica habilitada y las siguientes características para sus nodos:
  - Nodos *worker*:: instancias tipo Standard_F4s (8GB de memoria y 4 vCores), 1 instancia mínimo y 2 instancias máximo.
  - Nodo *master/driver*:: una instancia tipo Standard_F4s (8GB de memoria y 4 vCores).

  Las configuraciones anteriores resultan en un precio de USD$1 – USD$1.5 por cada hora de uso. Cabe resaltar que estas configuraciones pueden ser modificadas para agregar más nodos o mejorar el tipo de instancias y así reducir el tiempo de trabajo o procesamiento, aunque esto sube el costo total de la infraestructura.
- SQL Server: versión estándar con una capacidad máxima de 250GB entre todas las bases de datos que estarán dentro del servicio y un máximo de 10 unidades de procesamiento. Esta configuración es gratuita durante un año.

7.16.3. Diseño de bases de datos

Figura 7.20: Diagrama Entidad-Relación: de la base de datos diseñada

La información que se guardaría en la base de datos serían los datos que envían los sensores continuamente – para que los usuarios puedan descargar la información recopilada – además de los resultados del cálculo periódico para obtener el AQI y el significado de estos valores AQI que obtienen. En la figura anterior, se observa los campos de cada una de las tablas, aquellos que están resaltados en azul indican que son referencias directas al campo de otra tabla – *Foreign keys*: – y los campos en negritas indican la Llave primaria o *primary key*: – o campos con valores únicos – de la tabla.

7.17. Implementación de la infraestructura

7.17.1. Procesamiento de datos

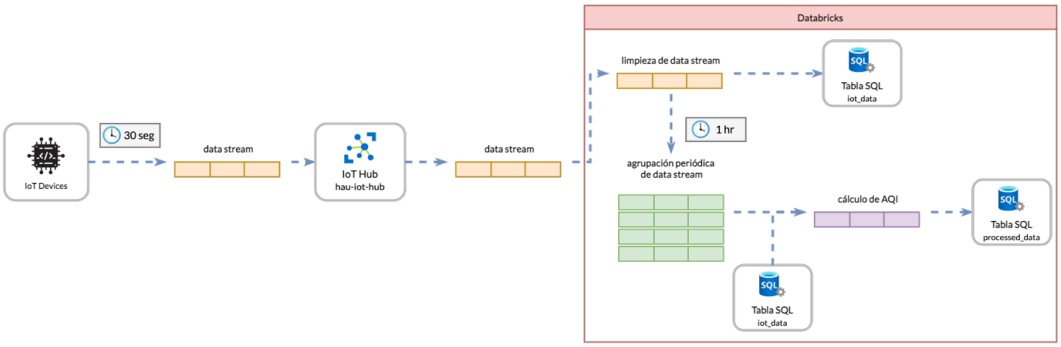


Figura 7.21: Flujo de los datos a lo largo de su procesamiento.

En la figura anterior, se observa cómo los datos se mueven y manejan desde que son enviados por los sensores hasta que se almacenan en la base de datos. Estos datos se envían cada 30 segundos por los sensores y llegan al IoT Hub, el cual los redirige inmediatamente a cualquier aplicación que esté pendiente de esta conexión; en este caso, es el proceso que se encuentra corriendo en Databricks. Cada envío contiene una única fila de mediciones de los contaminantes, la temperatura, la humedad y la hora a la cual fueron tomadas. En el momento que llegan a Databricks, los datos recibidos son formateados para adaptarse a la estructura de la tabla **iot_data** para posteriormente guardarlos en la misma tabla, fila a fila.

Por otro lado, tomando el mismo stream de datos, se realiza un corte cada hora y se agrupan los datos que se han tomado hasta el momento y se obtiene la hora a la que fue enviado el último registro previo al corte, considere esta hora como t. Con esta hora, se lee la tabla **iot_data** y se obtienen todos los registros que entren dentro del rango de t – 8 (horas previas) y se reúnen grupos de 1hr, obteniendo el valor máximo de cada contaminante y el promedio para la temperatura y humedad para cada grupo. En el caso de los contaminantes cuyo AQI se genera con un rango de 1hr, la concentración que entra al cálculo AQI es la de t. En el caso de los contaminantes de rango de 8hrs, la concentración que entra al cálculo de AQI es la concentración máxima entre todos los grupos que estén entre t y t - 8. Se utiliza la Ecuación 1 para obtener el AQI resultante para cada contaminante; de estos resultados, se obtiene el valor máximo entre los AQI de cada contaminante, este es el AQI general para la estación. Adicionalmente se obtiene el rango en el que cae este AQI general. Por último, se agregan los valores obtenidos en la tabla **processed_data**.

7.17.2. Ejecución de pruebas

Las pruebas de integración de la infraestructura implementada con los demás módulos se planificaron para realizarse de manera periódica, en lugares con ambiente variado (contaminado y no contaminado) con intervalos de tiempo limitados a algunas horas por las siguientes razones:

- No fue posible diseñar una fuente de energía independiente – como una batería – para los sensores, estos debían estar conectados a una computadora para funcionar. Esto implicó que,

para poder tomar mediciones continuas la mayor parte del día, los integrantes del proyecto debían permanecer en la intemperie en toda la duración de las mediciones.

- El costo de la plataforma era mayor a la cantidad de créditos de Azure que tenemos disponible si se quedaba activa todo el tiempo, por lo que era mejor si se limitaba el tiempo diario de pruebas.

Por lo anterior, se decidió realizar mediciones de 6 horas mínimo en cada estación para verificar el comportamiento de la infraestructura al tener múltiples horas de actividad. Las mediciones obtenidas y los AQIs resultantes fueron comparados con la siguiente ecuación de porcentaje de error:

$$Error\ porcentual = \frac{|valor\ experimental - valor\ teorico|}{valor\ teorico} * 100 \tag{7.13}$$

7.18. Proceso de Design Thinking

Se empleó el proceso de Design Thinking para entender y encontrar las necesidades del usuario y así lograr definir los casos de uso a emplear dentro de la plataforma a crear. Este es un proceso iterativo del cual se emplearon las fases descritas en el siguiente diagrama.

Figura 7.22: Proceso de Design Thinking empleado

Cabe destacar que este proceso no necesariamente se ejecuta de forma lineal, ya que varias veces se necesita regresar a una fase anterior para lograr definir de mejor manera el caso de uso.

7.18.1. Empatizar

Debido al COVID-19, esta fase de realizó de manera virtual con el apoyo de la plataforma Zoom, la cual no tiene límite de tiempo para las llamadas al ser utilizada con el correo brindado por la Universidad del Valle. Las llamadas se hicieron en conjunto con expertos sobre el tema ambiental, contando con el apoyo de trabajadores del Centro de Estudios Ambiental de la Universidad del Valle. Durante estas llamadas, se observaron distintos trabajos realizados por ellos, la mayoría de los trabajos fueron realizados con datos no pertenecientes a Guatemala o con datos de un satélite que visita el país una vez cada año. También se empatizó con una persona egresada de Ingeniería Química de dicha casa de estudio, que labora en el área de uso de sensores para la recopilación de datos de gases contaminantes. Por último, se realizó un grupo focal entre estudiantes de Biología y Biotecnología de la Universidad del Valle. Este grupo se fue consultado únicamente para esta fase en las fases siguientes todo retroalimentación se obtuvo de manera individual.

7.18.2. Definir

La fase de definición se realizaron diferentes mapas de empatía resultantes de la fase anterior, para entender lo que el usuario necesita y siente. Se definieron casos de uso para suplir dichas necesidades. Estos mapas se pueden observar en la sección de resultados.

7.18.3. Idear

La definición del proyecto tomó en cuenta los resultados y perspectivas obtenidos de la fase anterior. Para encontrar una solución a las necesidades, se realizó una lluvia de ideas de posibles gráficas, cuadros y funcionalidades que se pudiesen implementar para suplir estas. También se realizó una investigación comparativa de plataformas relacionadas con el tema que estuvieran accesibles al público, con el fin de determinar y definir aquellos datos de mayor relevancia para el público objetivo y púbico en general que pudiesen ser replicados para el contexto del proyecto a desarrollar. Así mismo en esta fase se realizaron mapas de inspiración para definir el diseño y paletas de colores a utilizar en el proyecto.

Las fases de prototipar y de pruebas con usuarios se definen más adelante.

7.19. Diseño de interfáz gráfica y experiencia de usuario

Se realizó un prototipo en papel con una esquematización de como quedarían los elementos definidos en la fase de Design Thinking, este primer prototipo en papel fue compartido con los integrantes del proyecto. A través de una llamada por la plataforma Zoom, se discutió respecto a como estos impactaban en la información a ser almacenada y recopilada. Así mismo se definieron los diferentes cálculos a realizar sobre la datos, para poder ser presentados en la plataforma. Esto se realizó con el fin de determinar si los elementos propuestos se podían lograr desarrollar tomando en cuenta los datos de ingreso y el proceso de almacenamiento de los gases.

Junto con el mapa de inspiración obtenido en la fase de ideación, se definió de manera puntual el diseño y paleta de colores a implementar dentro de la interfaz. Habiendo obtenido esto, se realizó una investigación de herramientas para realizar prototipos comparando ventajas y desventajas entre las más populares. Se decidió utilizar la herramienta gratuita Figma, la cual permite edición colaborativa en tiempo real, facilita el desarrollo brindando código CSS para cada elemento realizado y permite el uso de un repertorio amplio de iconos y formas previamente diseñadas.

Se realizaron dos versiones del prototipo, en cada una se prototipó una pantalla llamada "Tablero"la cual contiene información y gráficos pertinentes a los datos recopilados y una segunda pantalla llamada "Datos.[en] la cual el usuario podía ver las diferentes secciones de filtros para descargar los datos de acceso públicos que habían sido recopilados. Con cada una de las versiones del prototipo se realizaron pruebas con usuarios para validar los elementos y la experiencia de usuario.

7.20. Diseño de Arquitectura del Proyecto

El diagrama muestra la estructura y conexión de los diferentes componentes del proyecto, a continuación se detalla cada uno de ellos.

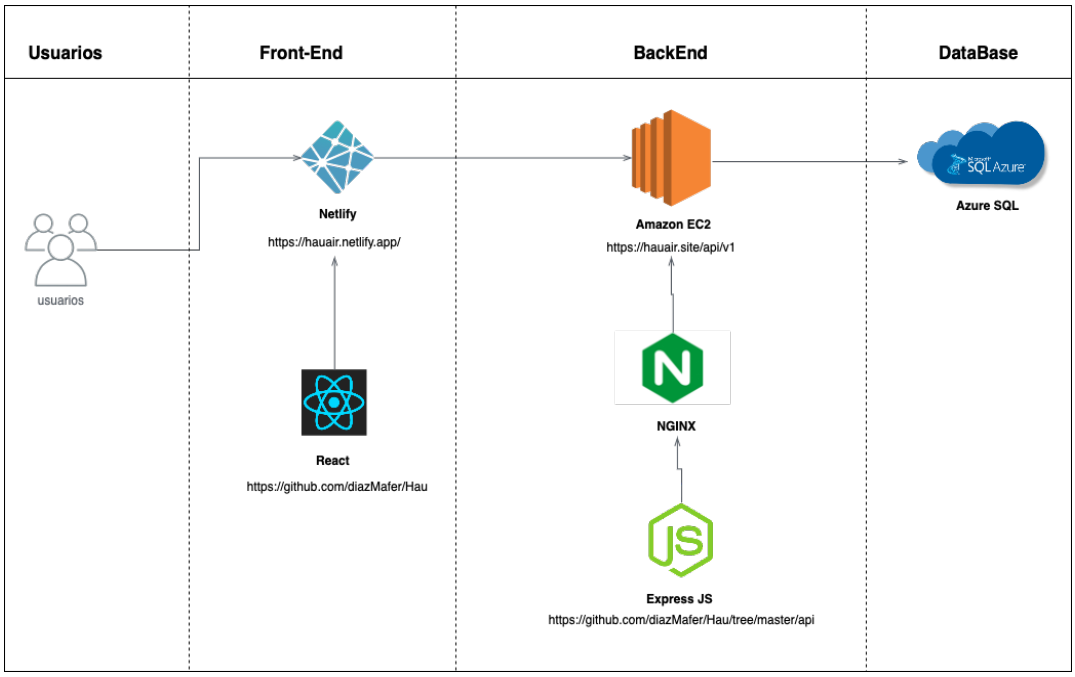


Figura 7.23: Arquitectura del proyecto

7.20.1. Interfaz de programación de aplicaciones

Se desarrolló utilizando Express, el uso de un framework como este permite la implementación de protocolos http y https en las rutas. Se implementó el patrón de diseño, modelo, vista, controlador, siendo el modelo quien se conecta a la base de datos y realiza la consultas, el controlador es quien se encarga de servir la ruta a internet y proporcionar los datos que la vista devuelve, por último la vista es quien se encarga de recibir los distintos parámetros enviados a través de las rutas para este comunicárselos al modelo y obtener los datos solicitados. En el siguiente diagrama se detalla la interacción entre los diferentes elementos del modelo de la interfaz de programación.

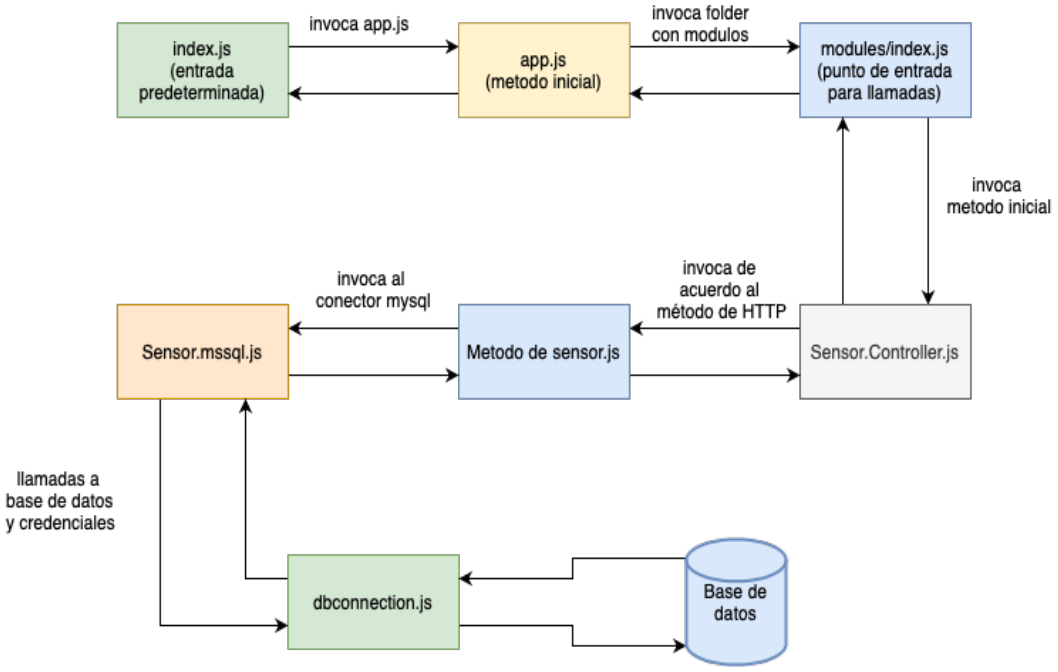


Figura 7.24: Conexión de componentes de la interfaz de programación

Para la configuración de esta se descargó previamente una versión de Node y NPM. Seguido de esto se instaló el framework de Express y la librería de Express Generator utilizando el manejador de paquetes NPM. A continuación se generó un proyecto a través de la instrucción *'npx express-generator api'*. Dentro del proyecto creado previamente, se instaló un conector mssql para la conexión a la base de datos Microsoft SQL Server, esto se realizó ejecutando la siguiente instrucción *'npm i mssql'*. Por último, se empleó una librería que facilita el uso compartido de recursos entre orígenes (CORS) para lograr la comunicación entre las aplicaciones web construidas y los ordenadores.

7.20.2. Sistemas en la nube

Para hacerla pública y que fuera accesible desde cualquier punto del internet, se utilizó el servicio de máquinas virtuales en la nube de Amazon Web Services (AWS) denominado, Elastic Cloud Compute (EC2). Para la creación de la máquina virtual se creó una cuenta dentro de la plataforma de AWS, ya creada la cuenta se realizaron los siguientes pasos para la creación del EC2:

1. Se escogió una imagen de Ubuntu Server 18.0.4 de 64 bits como sistema operativo de la maquina virtual.
2. Para el tipo de instancia, se seleccionó una t2.micro que cuenta con un rendimiento de red moderada. Así mismo, se escogió un almacenamiento de estado sólido, para que la aplicación pudiera procesar y responder de manera más rápida.
3. Habiendo elegido las configuraciones del sistema, se configuró un grupo de seguridad para poder habilitar y acceder a la máquina utilizando distintos protocolos. Los puertos que se habilitaron fueron los siguientes:
   a) Tipo SSH con protocolo TCP en el puerto 22 con opción de ip local, esto tomara la ip pública de la computadora en donde se esta realizando la configuración.

b) Tipo HTTP con protocolo TCP en el puerto 80 con opción de que todas las ips (0.0.0.0/0) puedan acceder a él.

c) Tipo HTTPS con protocolo TCP en el puerto 443 con opción de que todas las ips (0.0.0.0/0) pueden acceder a él.

IP version	Type	Protocol	Port range	Source
IPv4	SSH	TCP	22	190.99.116.188/32
IPv4	HTTPS	TCP	443	0.0.0.0/0
IPv6	HTTP	TCP	80	::/0
IPv6	HTTPS	TCP	443	::/0
IPv4	HTTP	TCP	80	0.0.0.0/0

Figura 7.25: Configuración de protocolos a través de puertos en la instancia EC2

Por último, se eligió la opción de generar una nueva llave de seguridad tipo pem para la máquina creada previamente. Este archivo generado es único por cada una y sirvió para conectarse a esta a través de un puerto ssh. Dentro de la máquina virtual, se actualizaron todos los paquetes fuentes del sistema operativo utilizando el comando *'sudo apt-get update'*, esto se realizó para obtener una versión actualizada de todas las dependencias y paquetes del sistema. Seguido de esto se instalo Node JS y NPM a través de los siguientes comandos 'curl -sL https://deb.nodesource.com/setup_11.x | sudo -E bash -' y 'sudo apt-get install -y nodejs'.

Para la instalación de Nginx se utilizó la guía de instalación y configuración oficial del sitio de Nginx, en donde se corrieron los siguientes comandos para instalarlo en la máquina virtual:

1. *sudo wget http://nginx.org/keys/nginxsigning.key*
2. *sudo apt-key add nginxsigning.key*
3. *cd /etc/apt*
4. *sudo nano sources.list*
5. Dentro del archivo que se abrió con el comando anterior se agregaron las siguientes lineas al final del archivo:
   - deb http://nginx.org/packages/ubuntu xenial nginx
   - deb-src http://nginx.org/packages/ubuntu xenial nginx
6. *sudo apt-get update*
7. *sudo service nginx start*

Para verificar que se haya instalado correctamente Nginx se abrió la ip pública de la máquina virtual en un navegador y se comprobó que lo mostrado en pantalla fuera una pagina web con instrucción de bienvenida de Nginx. Seguido de esto se procedió a clonar el repositorio del API, una vez realizado esto se instalaron todas las dependencias ejecutando el comando 'npm install' dentro de la carpeta del API. Teniendo ya el API corriendo dentro del servidor se realizo la configuración de un Reverse Proxy, editando los archivos de configuración de Nginx. Para que la aplicación pudiera utilizar un protocolo https se adquirió el dominio 'hauair.site' y se configuró un DNS utilizando la ip pública de la máquina creada. Seguido de esto se instaló certbot para generar un certificado ssl, el cual estuviera asociado con el dominio adquirido. Para la generación del certificado se ejecutaron los siguientes comandos:

1. sudo apt-get update
2. sudo apt-get install software-properties-common
3. sudo add-apt-repository universe
4. sudo add-apt-repository ppa:certbot/certbot
5. sudo apt-get update
6. sudo apt-get install certbot
7. sudo certbot certonly –standalone

Para finalizar se accedió a la máquina virtual a través del dominio hauair.site utilizando el protocolo SSL y se accedió a una de las rutas del API para validar su funcionamiento.

7.20.3. Interfaz de usuario

Se realizó una investigación sobre tecnologías para desarrollo de interfaces de usuario tomando en cuenta que estas fueran de código abierto, altamente mantenibles y con una alta disponibilidad de recursos. Esto para realizar el desarrollo de manera más rápida, segura y sostenible en un futuro. Entre las tecnologías encontradas durante la fase de investigación se concluyo que React Js era la más adecuada para un desarrollo rápido y sostenible.

Para el diseño de la interfaz de usuario se llevó a cabo una serie de fases, las cuales se demuestran en el siguiente diagrama.

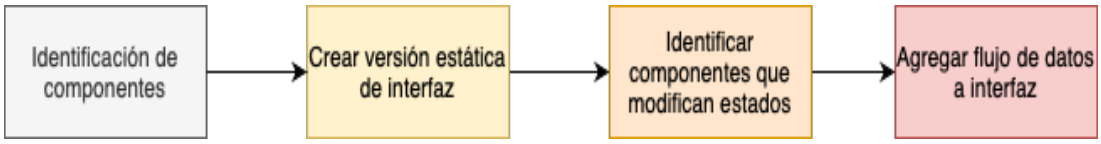


Figura 7.26: Proceso de diseño para la creación de la interfaz de usuario

La identificación de componentes a desarrollar se realizó tomando como base el último prototipo desarrollado, dentro de este se determinaron cuáles elementos de la interfaz gráfica debían de convertirse en un componente independiente dentro del código a desarrollar. La forma en que se identificó esto fue tomando en cuenta la funcionalidad que debía de tener cada elemento y la cantidad de elementos que dependían o interactuaban con estos. A su vez se identificó la jerarquía entre los componentes, dicho proceso tomó en cuenta el tamaño de dicho elemento y la cantidad de otros elementos de interfaz gráfica que se encontraban dentro de este.

Para la creación del proyecto en React Js se utilizó una plantilla de código abierto llamada React-Bolt para obtener una estructura del proyecto con distintas librerías para manejo y buenas prácticas de desarrollo de código, generador de código para ambientes de producción y librerías para la implementación de rutas, pruebas y componentes estilizados dentro del proyecto. Esta plantilla se obtuvo a través de su repositorio oficial en GitHub. Seguido de esto se desarrolló el código para cada uno de los componentes identificados para obtener la versión estática de la aplicación. Habiendo obtenido esto, se procedió a conectar el Front-End con la interfaz de programaciones que se encontraba en la nube. Esto se hizo utilizando peticiones de Axios, en donde en cada estado u actualización de los componentes se hace la llamada a la ruta que devolvería los datos necesarios para ser mostrados en dicho componente. Por último, se implementó un diseño adaptable, el cual consiste que a medida que la resolución de pantalla se vuelve menor, el contenido se adapta a una sola

columna. Esto se implementó a través del uso de *media queries* de css e implementando propiedades *Flex Box* para adecuar los elementos de las filas en columnas. Para validar que la implementación fuera correcta se utilizó el servicio Search Console de Google *Mobile-Friendly Test* para realizar pruebas sobre la página.

7.21. Pruebas con usuarios

Las pruebas con usuarios se realizaron de forma virtual haciendo uso de la plataforma Zoom. Para validar los prototipos realizados se diseñó una serie de preguntas las cuales abarcan aspectos de identidad y contenido de la plataforma, estas preguntas se validaron con la ayuda de un experto en diseño de experiencias. Dichas preguntas, contienen una escala del 1 al 5, esto se definió de esta manera para medir las habilidades, perspectivas y conocimientos del usuario relacionados con lo que se le está preguntando.

Las preguntas que se le brindaban al usuario fueron las siguientes:

1. ¿De qué se trata el tablero?
2. ¿Qué lo hizo determinar sobre que trata el tablero?
3. En una escala del 1 al 5 ¿qué tan cómodo se siente utilizando una herramienta de visualización de datos en línea?
4. ¿Hacia qué tipo de audiencia cree que está dirigido el sitio?
5. En una escala del 1 al 5 ¿qué utilidad tienen los elementos en el tablero?
6. ¿Usarías el tablero en tu día a día?
7. ¿Cree que es suficiente información para un análisis diario de la calidad del aire?
8. ¿Logró navegar a la página de descarga de datos y encontrar el botón para descargarlos?
9. En una escala del 1 al 5 ¿qué tan cómodo se sintió utilizando el tablero?

También se realizó una prueba de usabilidad, utilizando una serie de preguntas que conforman la escala de usabilidad del sistema. La prueba está conformada por 10 preguntas que pretenden ofrecer una visión global de la usabilidad del sistema. Se utilizaron las preguntas estándares de la prueba.

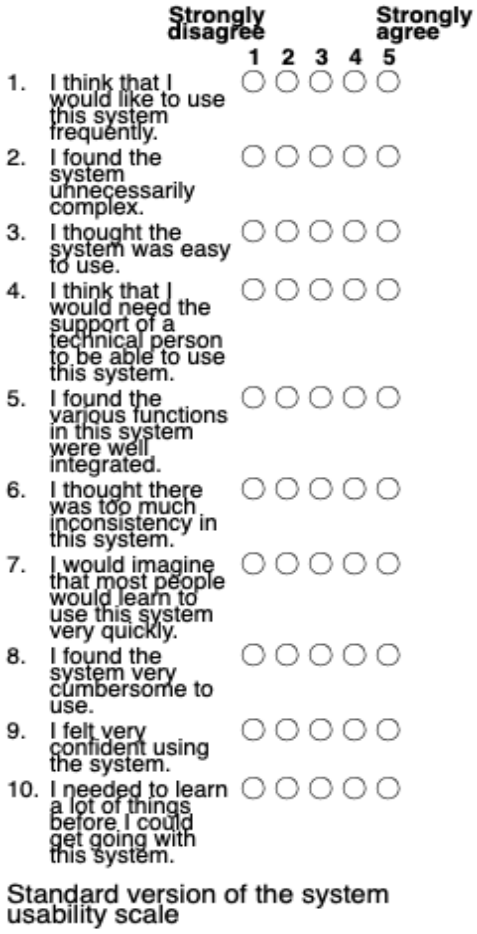


Figura 7.27: Versión estándar de escala de usabilidad del sistema

Con los resultados obtenidos por pregunta se realizó el siguiente procedimiento para obtener el resultado de la prueba:

- Se sumaron los resultados finales obtenidos por cada pregunta y se dividieron dentro de 10 para obtener el promedio por pregunta.
- Se sumó el valor obtenido para cada pregunta de número impar (1,3,5,7,9) y se le restó cinco.
- Se sumó el valor obtenido para cada pregunta de número par (2,4,6,8,10).
- Se restó de 25 el valor obtenido al sumar preguntas pares.
- Se sumó el valor obtenido en el paso 2 con el valor obtenido en el paso 4 y se multiplicó por 2.5.

Al finalizar se clasificó el resultado obtenido con base en una escala de usabilidad.

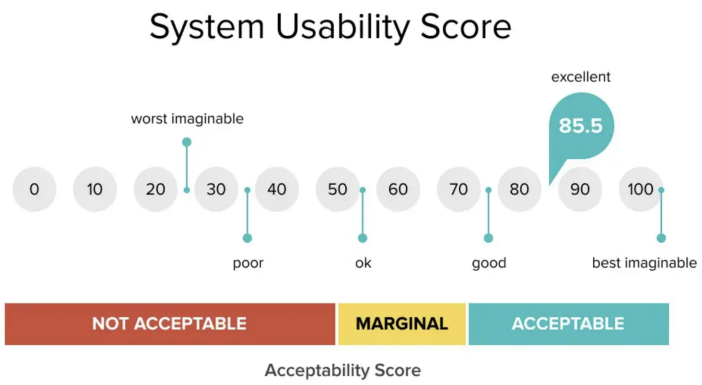


Figura 7.28: Rango de aceptabilidad prueba SUS.

Fuente: Adobe XD [71]

Dentro de la reunión de Zoom se compartió el link al prototipo y se le solicitó al usuario que compartiera su pantalla para observar cómo este usaba la plataforma. Seguido de esto se le pidió al usuario su opinión y recomendaciones respecto a lo que había observado y por último se le brindaron ambos pruebas para que las contestara.

CAPÍTULO 8

Resultados: Estación de medición

Los resultados obtenidos por parte de la estación de medición fueron clasificados según el tipo de gas medido. Cada gas fue reportado en partículas por millón (ppm) y se generaron datos de incertidumbre para el sensor de CO (monóxido de carbono). Debido a limitantes de falta de equipo de medición, los otros gases no fueron sometidos a este tipo de pruebas, sin embargo fue realizado el proceso de preparación y calibración recomendados por los fabricantes de cada uno de los sensores. Para determinar la efectividad de cada uno de los sensores, se compararon los valores recolectados y se evaluó si estaban dentro del rango aceptable del cálculo promedio de AQI de una u ocho horas, dependiendo del tipo de gas. Los resultados finales fueron, el sensor MQ135 (monóxido de carbono) tiene porcentaje de error de $3.33 * 10^{-}5\,\%$ y una validez promedio de 99.57 %. El sensor DGS-SO2 (dióxido de sulfuro) tiene una validez promedio de 79.27 %, el próximo sensor MICS-6814 (dióxido de nitrógeno y monóxido de carbono), este sensor fue utilizado para extraer valores de ambos gases, como redundancia lecturas de CO. Este sensor tiene una validez promedio de 100 %. Finalmente el sensor MQ131 (Ozono) tiene una validez promedio 99.32 %. Estos promedios fueron extraídos de tres mediciones de aproximadamente 6 horas de duración cada una de ellas.

La infraestructura en la nube tuvo un desempeño satisfactorio, en las pruebas de estrés en donde se agregaron hasta 20 fuentes distintas de datos y se observó que esta aprovisionaba más recursos para acomodarse a la carga de trabajo de manera fluida y sin interrupciones visibles. Se estableció de manera exitosa una vía de comunicación entre la plataforma de procesamiento de datos y la estación de medición a través del API proveído por Azure IoT Hub a través de la cual se obtuvieron las concentraciones periódicas reportadas por los sensores para cada locación, los AQI de estas concentraciones se compararon - mediante el cálculo de porcentajes de error - con los AQI obtenidas por la calculadora de Índice de Calidad de Aire oficial de la Agencia de Protección Ambiental de Estados Unidos. En total, para los contaminantes de CO, SO2 y NO2, se obtuvieron porcentajes de error de 2.23 %, 8.99 % y 6.45 %, respectivamente; en el caso del O3, no se consideró el porcentaje de error obtenido, ya que las magnitudes utilizadas por la calculadora diferían de las magnitudes utilizadas por la estación de medición, resultando en *error matemático*. Por último, se realizó el diseño e implementación de la base de datos en un SQL Server en Azure la cual almacenó la información recolectada y procesada y se realizó la conexión entre esta y el servidor de la página web para atender poder leer y escribir sobre la información almacenada y ser presentada al usuario final.

Con el proceso de Design Thinking se obtuvieron cinco casos de usos, los cuales cubrían las necesidades del usuario. Dentro de ellos el caso de uso de brindar la información de la calidad del aire en un plataforma, cuyos índices se obtuvieron del programa del INSIVUMEH. Para el primer

prototipo realizado se obtuvo poca aceptación de los usuarios y mucha critica respecto a la forma en que los datos eran presentados en la plataforma. El segundo prototipo realizado obtuvo una mayor aceptación por parte de los usuarios obteniendo. Esto se puede ver reflejado que un 60 % respondió que los elementos tienen un nivel 4 de utilidad. Para el desarrollo del API se obtuvieron 10 rutas de tipo POST, las cuales se desarrollaron de manera facil y rapida gracias al uso de ExpressJS. La infraestructura en AWS resulto en una instancia EC2 Micro, la cual se le instalo un servidor inverso proxy que permitió la apta comunicación entre el API y la interfaz de usuario. Con ReactJS se obtuvieron dos pantallas que fueron servidas a través de Netlify de manera sencilla permitiendo una integración continua desde el repositorio de desarrollo. Para las pruebas finales con usuarios se obtuvo que un 80 % se encuentra cómodo utilizando la plataforma y un 70 % opino que los elementos del tablero tenían un nivel cinco de utilidad. Un 80 % opina que la información es suficiente para realizar un análisis diario de la calidad del aire. Por lo que el contenido de la plataforma es suficiente para visualizar y analizar datos de calidad del aire. Por otra parte nuevamente 100 % de los usuarios lograron navegar y utilizar la página de descarga datos haciendo que se logre distribuir y visualizar estos. Para las pruebas de usabilidad se obtuvo un 85.25 %.

8.1. Datos reportados por MQ135

Las mediciones obtenidas en las pruebas referidas en la sección 7.10 (Validación de datos obtenidos) dieron resultados con un bajo porcentaje de error, a continuación se muestran las imágenes tomadas durante la prueba.

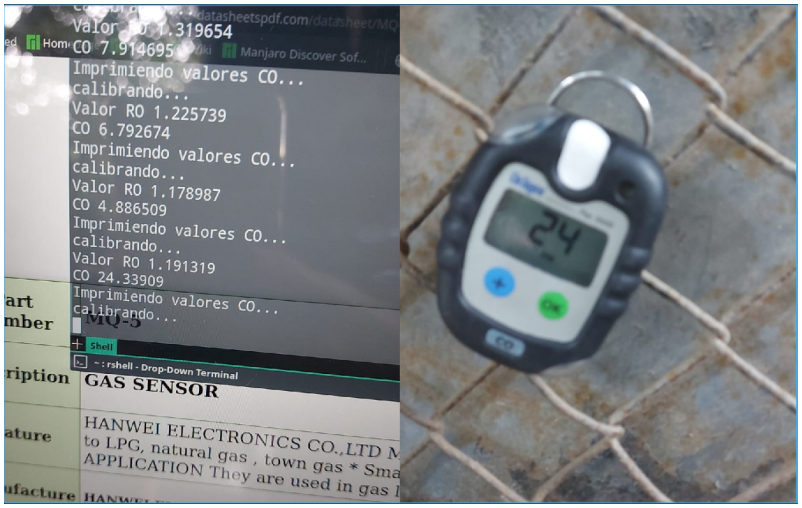


Figura 8.1: Comparación MQ135 - Dräger 5500

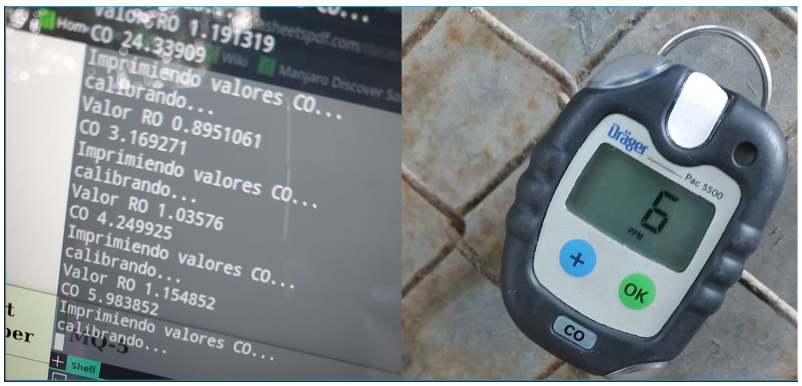


Figura 8.2: Comparación 2 MQ135 - Dräger 5500

De las figuras 8.1 y 8.2 se obtienen los valores observados: 24.339090ppm y 5.983852ppm. Al observar los resultados obtenidos y utilizando la ecuación 7.12 se obtienen los porcentajes de error: $1.38 * 10^{-}5\,\%$ y $3.33 * 10^{-}5\,\%$ respectivamente, por lo que se puede considerar que los valores son certeros, en las mediciones de CO reportadas por MQ135.

8.2. Validez de los datos

Debido a la ausencia de equipo especializado de medición del resto de los gases, se utilizaron los datos de concentración para determinar la validez de los datos reportados por la estación en cada uno de los lugares donde fue delimitado este proyecto. Estas mediciones fueron comparadas con la calculadora AQI desarrollada por AirNow. La calculadora AQI define limites superiores para los distintos AQIs calculados.

Gas	**AQI**	**Límite superior de concentración (ppm)**
SO_2	SO_2, 1h avg	0.3
NO_2	NO_2, 1h avg	2.05
O_3	O_3, 8h avg	0.2
CO	CO, 8h avg	50

Tabla 8.1: AQIs y sus límites superiores

8.3. Datos reportados por DGS SO2

Con el caso del sensor que captura valores de concentración de SO_2 inicialmente se observaron valores grandes de hasta 15ppm de SO_2, y conforme se mantenía conectado a corriente iban decreciendo los valores hasta llegar a valores cercanos a 0ppm, tal y como el fabricante mencionó en su datasheet [68]. Sin embargo, el sensor pasados 15 minutos, comenzó a reportar valores erróneos. Existieron casos donde el sensor reportó valores negativos. Para mitigar esto, se realizó un offset restando el valor más grande obtenido y comenzar a reportar desde ese valor, por ejemplo: Se reportaron valores de hasta -7043 ppb por el sensor. Se utilizó este valor negativo y se sumó, para tener un valor 0 y comenzar a reportar desde ahí. Inicialmente esto funcionó, sin embargo cuando el sensor se normaliza comienza a reportar valores como 2.45ppm, lo cual nuevamente es erróneo dado que esos valores fueron observados en ambiente interior, donde no debería de existir emisión de SO_2. Una de los posibles razones por las que estos valores pueden salir del rango válido es por

tomar mediciones en ambientes interiores, esto mencionado por PhD David Peaslee, en un GitHub issue. [61].

A continuación la tabla muestra los resultados obtenidos según los tres lugares de medición a los que fue delimitado, así como los porcentajes de validez de datos según la tabla de validez de AQI. 8.1.

Fecha de medición	Ubicación	Reportes válidos	Reportes enviados	Validez (%)
2021/10/01 al 2021/10/23	Ciudad San Cristóbal	713	771	92.48 %
2021/10/01 al 2021/10/23	Centro Histórico	923	1115	82.78 %
2021/10/01 al 2021/10/23	Carretera al Salvador	416	665	62.56 %

Tabla 8.2: Tabla de validez de datos: DGS-SO2

Al realizar un promedio sobre los porcentajes de validez de datos se obtiene que los datos reportados por el sensor DGS-SO2 tienen un 79.27 % de validez. Como ya fue mencionado, resulta imposible determinar el porcentaje de error en el resto de los datos teóricamente validos.

8.4. Datos reportados por MICS-6814

Los valores de concentración reportados por este sensor son para los gases CO y NO_2. Durante la recolección de datos, se observó que los valores para las concentraciones de NO_2 nunca son 0ppm, ya mencionado anteriormente es imposible determinar el porcentaje de error de estas mediciones debido a la falta de equipo especializado. Analizando la totalidad de datos recolectados entre las fechas de 20 de octubre y 23 de octubre, en todas las estaciones se obtienen los valores para las siguientes tablas de validez de NO_2 y CO respectivamente:

Fechas de medición	Ubicación	Reportes válidos	Reportes enviados	Validez (%)
2021/10/01 al 2021/10/23	Ciudad San Cristóbal	771	771	100.00 %
2021/10/01 al 2021/10/23	Centro Histórico	1115	1115	100 %
2021/10/01 al 2021/10/23	Carretera al Salvador	665	665	100 %

Tabla 8.3: Tabla de validez de datos (NO_2) MICS-6814

Los datos de la tabla reportan validez del 100 %, sin embargo resulta sospechoso, que nunca existan valores muy cercanos a 0ppm, como si ocurre con los demás gases. Esto podría indicar valores que los valores reportados son una sobre estimación de la cantidad real, aunque todos siguen siendo válidos. El fabricante del board del sensor, Seeed indica que los valores reportados por este sensor son con el método de calibración empleado no espera reportar valores exactos, por lo que es esto es un comportamiento aceptado por el fabricante. [1].

Fecha de medición	Ubicación	Reportes válidos	Reportes enviados	Validez (%)
2021/10/20	Ciudad San Cristóbal	770	771	99.87 %
2021/10/20	Centro Histórico	1109	1115	99.46 %
2021/10/23	Carretera al Salvador	661	665	99.39 %

Tabla 8.4: Tabla de validez de datos (CO) MICS-6814

La plataforma de AQI para CO toma como reporte inválido cualquier medición por encima de 50ppm. Los valores reportados son un promedio entre los valores del sensor MQ135 y el MICS-6814. El porcentaje de error del MICS-6814 es de 10 % respecto al sensor MQ135. Se observa que en promedio el 99.57 % de los datos son válidos. Los datos inválidos reportados en el centro histórico

ocurrieron puesto que se tomaron mediciones dentro de un carro encendido en el tráfico y al no tener ventilación existió un periodo donde se reportaron valores por encima de 100ppm, esto ocurrió por medir en un ambiente cerrado y con fuentes de emisión de CO, por lo que aún al parecer inválidos para el cálculo de AQI, se consideran correctos, especialmente porque se tiene porcentajes de error para estos datos.

8.5. Datos reportados por MQ131

Este sensor se encarga de reportar concentraciones de O_3. La calculadora de AQI para el promedio de una hora de O_3 toma como máximo un valor de 200 ppb. Para analizar la validez de los datos enviados, se realiza la siguiente tabla:

Fecha de medición	Ubicación	Reportes válidos	Reportes enviados	Validez (%)
2021/10/01 al 2021/10/23	Ciudad San Cristóbal	771	771	100 %
2021/10/01 al 2021/10/23	Centro Histórico	1100	1116	98.56 %
2021/10/01 al 2021/10/23	Carretera al Salvador	661	665	99.39 %

Tabla 8.5: Tabla de validez de datos: MQ131

En promedio se obtuvo 99.31 % de validez de las concentraciones de O_3 enviadas, al igual que los otros sensores no se tiene acceso a equipo especializado que permita determinar porcentajes de error contra sensores profesionales.

8.6. Estructura de envío de datos

Cada mensaje enviado a la infraestructura Azure tiene la forma de la Tabla 8.6. El dispositivo guarda las primeras 150 observaciones en un archivo CSV (Comma separated value, por sus siglas en inglés). En la siguiente tabla se muestran tres de los valores obtenidos, la estructura de la tabla, es la misma a la que se envía al IoT Hub.

Timestamp	DeviceId	CO	SO2	O3	NO2	temperature	humidity
2021-10-2 17:11:22	s02	0.000522	0.000999	0.010818	0.211513	23.3	58.3
2021-10-2 17:11:55	s02	0.000393	0.0	0.011334	0.218295	23.4	58.1
2021-10-2 17:12:28	s02	0.000476	0.021000	0.011692	0.223224	23.4	58.1

Tabla 8.6: Resultados y mediciones enviadas

Todas las mediciones de concentración de los gases son reportados en ppm, temperatura es reportada en grados celsius y la humedad es reportada como humedad relativa, siendo esta un porcentaje. El comportamiento de los sensores es variado, cada uno presenta particularidades, por ejemplo los valores del SO2 en ambientes cerrados son demasiado altos e incorrectos, dado que en ocasiones se reportan concentraciones de 1ppm de SO2, las cuales según la calculadora de AQI de AirNow son consideradas peligrosas. Adicionalmente, se observa que las concentraciones de NO2 se encuentran particularmente altas y estas conforme avanza el uso del sensor aumentan hasta 0.50, luego de 2 horas continuas de medición, esto sugiere revisar como calibrar el sensor, particularmente sugiere que el sensor debería ser calibrado luego de 2 horas continuas de mediciones. Por otra parte los valores de O3, nunca son 0ppm y se encuentran en concentraciones bajas, sin embargo requiere de más estudios y equipos de medición para poder determinar, como se determinó en el caso de las concentraciones de CO los porcentajes de error del sensor.

8.7. Conectividad con infraestructura

Utilizando el protocolo MQTT, se realizaron pruebas donde existiera una pérdida en la conexión entre la estación y la infraestrucura en Azure. La prueba fue ejecutada desconectando el cable coaxial del *router* al cual estaba conectado la estación temporalmente. La estación continuó funcionando de manera normal hasta que intentó enviar un mensaje con los datos recopilados, se observó que se reportó un error desde MQTT intentando realizar una recolección. Al reconectar el cable coaxial al router el dispositivo continuó enviando datos, sin embargo cabe destacar que el proceso fue repetido en varias ocasiones y en una de ellas la estación no salió del proceso de recuperación de conexión, por lo que fue requerido reiniciar el dispositivo. Esta prueba donde no se logró reconectar la estación fue experimentando con un largo periodo sin conexión. A continuación se muestra una imagen con el error reportado cuando se pierde conectividad entre la estación y el dispositivo.

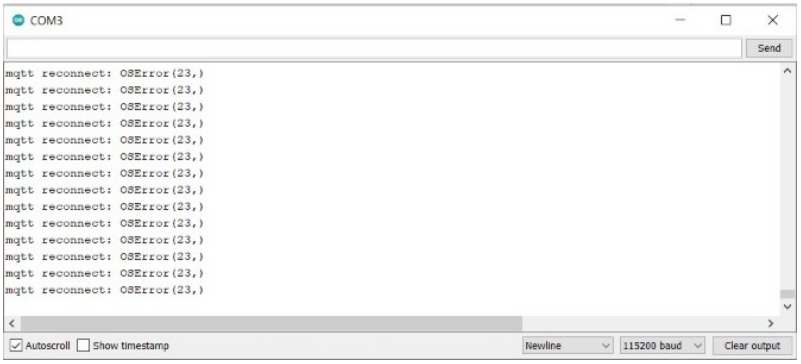


Figura 8.3: Prueba de reconexión

8.8. Pruebas del contenedor

El contenedor desarrollado para la estación fue sometido a una prueba donde se expuso el contenedor desarrollado a la lluvia. Para ello el día 28 de octubre del 2021 a las 7:48pm se expuso a la lluvia el contenedor sin la estación adentro, por los costos que podría tener si se expone la estación al agua. Por aproximadamente 15 minutos, fue expuesto el contenedor. A continuación se muestra una imagen del contenedor bajo la lluvia.

Figura 8.4: Prueba de contenedor

Al observar la Figura 8.4 se ve que se forman cúmulos de agua, esto sugiere que la película de barniz agregada al sensor funciona para impermeabilizar el contenedor. Esto aumenta la tolerancia del mismo a la lluvia. Además se hace un análisis del estado del interior del contenedor desarrollado

Figura 8.5: Resultados de prueba de resistencia al agua

Al observar el interior se observa una pequeña cantidad de agua que se filtró en el contenedor, esto quizás debido a las entradas de aire colocadas en el mismo. Esta cantidad es pequeña, pero podría aumentar en casos de bastante lluvia, o lluvia con viento, lo cual podría darle un ángulo de inclinación a la caída de las gotas de agua, aumentando la probabilidad de entrar en el sensor. Adicional a esto no se observó humedad en el contenedor desarrollado, por lo que el material impermeabilizante funcionó adecuadamente.

CAPÍTULO 9

Resultados: infraestructura en la nube

La infraestructura fue diseñada con la intención de ser fácilmente escalable, si en algún momento en el futuro se desea agregar más sensores todas las herramientas que viven en la nube deben adaptarse a la carga de trabajo sin permitir fallas en ningún punto del pipeline. A pesar de que las herramientas sí tienen límites de recursos que pueden aprovisionar que depende de la configuración que se escoge para cada una – por ejemplo, IoT Hub tiene un límite diario de 8,000 mensajes intercambiados con los dispositivos registrados – se realizaron pruebas para verificar que la plataforma lograra adaptarse a mayor carga de trabajo, siempre dentro de sus límites. Para IoT Hub, se utilizó un simulador de Raspberry Pi que actúa como un sensor extra y se comunica con la nube. Este simulador envía datos cada 30 segundos.

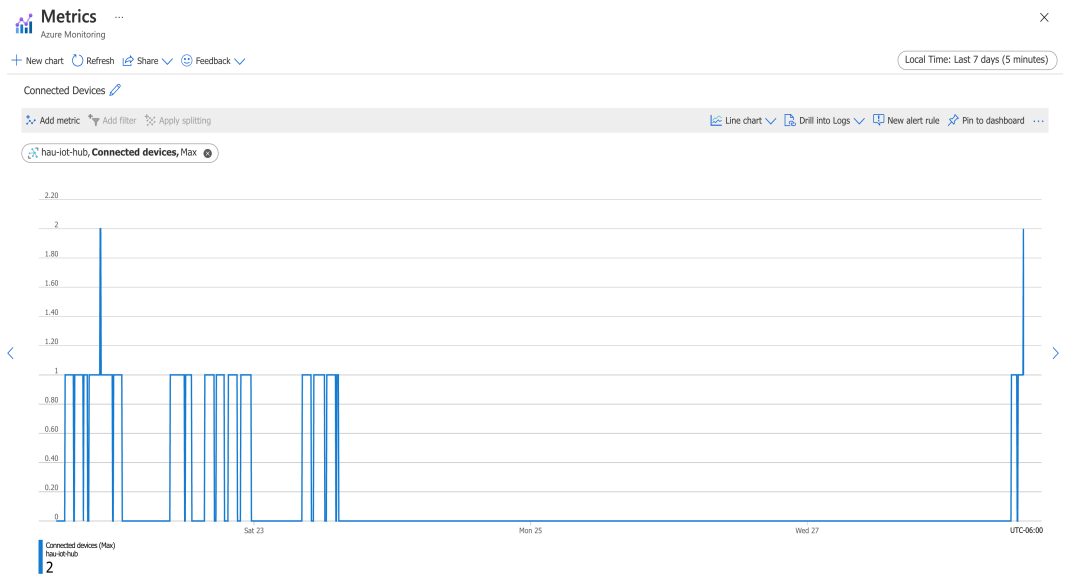


Figura 9.1: Cantidad de dispositivos en comunicación con IoT Hub en un período de 7 días.

La figura anterior demuestra la cantidad de dispositivos comunicándose con IoT Hub para un momento dado, se puede ver que en algunas instancias la cantidad de dispositivos conectados sube a 2; en dichas instancias se mantuvo un monitoreo constante para verificar que en Databricks se estuviera recibiendo los datos enviados tanto por la estación como por el sensor y se observó que en efecto los datos provenían de distintas fuentes, confirmando que logra recibir conexiones de múltiples dispositivos y centralizar la información sin problema. En este monitoreo también se observó que Databricks proporcionaba más nodos (1 más) del clúster para acomodarse a la mayor cantidad de datos que se estaban recibiendo. En el caso de la base de datos SQL Server, también se realizaron pruebas de carga que resultaron exitosas, estas se describirán más adelante. De acuerdo con estas pruebas, se determinó que la plataforma escala dinámicamente cuando lo necesita de manera satisfactoria y que las configuraciones establecidas son suficientes para el propósito del proyecto.

Al implementar la infraestructura fue evidente que no era necesario programar manualmente un API que realizara la comunicación entre los dispositivos electrónicos o sensores y IoT Hub, ya que esta herramienta ya contaba un API ya definida y fue la que se utilizó. Este API: funciona con *Requests* y *responses*: que acatan el Protocolo HTTPS: enviados a través del internet – utilizando el puerto 8883 de los dispositivos –, los mensajes son intercambiados directamente con los sensores. La autenticación de los sensores se realiza con un *SAS TOKEN:* válido por cierta cantidad de tiempo y debe ser actualizado manualmente al expirar, de lo contrario la conexión es denegada. Con la comunicación ya establecida, se obtuvieron las concentraciones periódicas reportadas por los sensores para cada estación, las cuales se visualizan en las siguientes figuras.

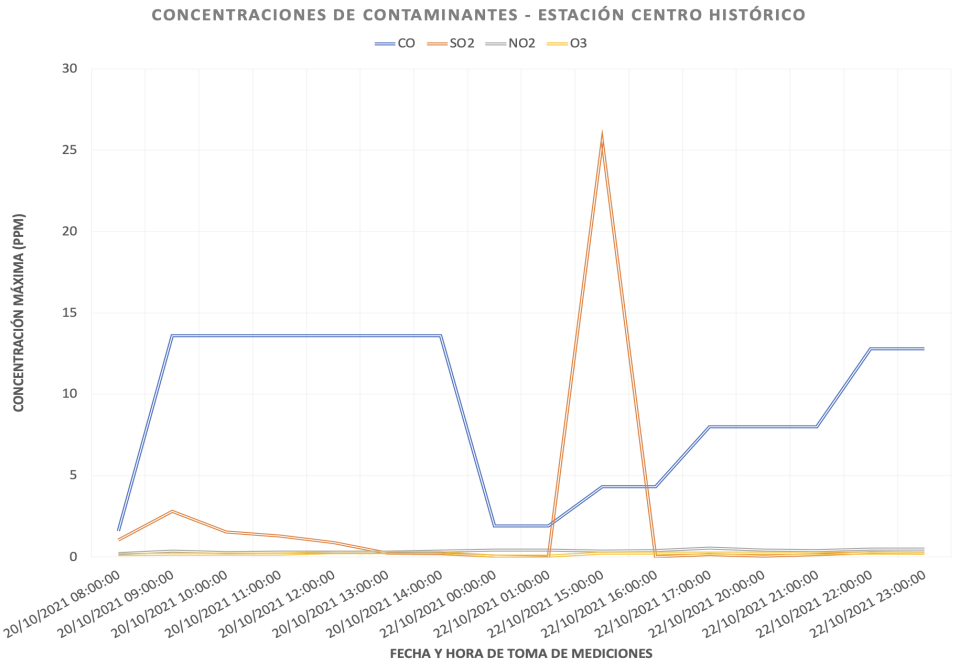


Figura 9.2: Contaminantes obtenidos para la estación del Centro Histórico agrupadas por hora.

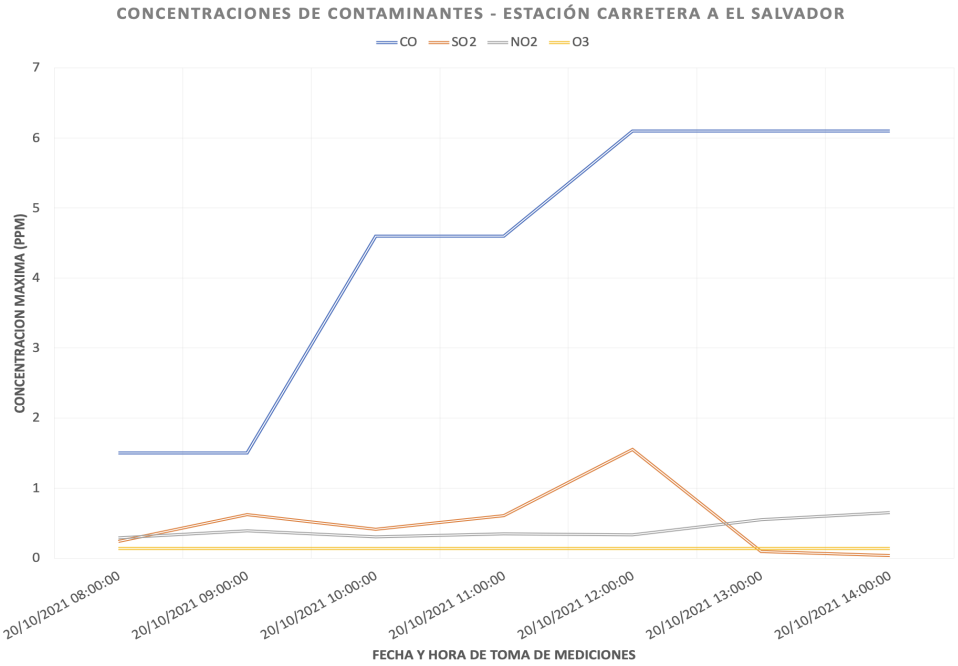


Figura 9.3: Contaminantes obtenidos para la estación del Carretera Salvador agrupadas por hora.

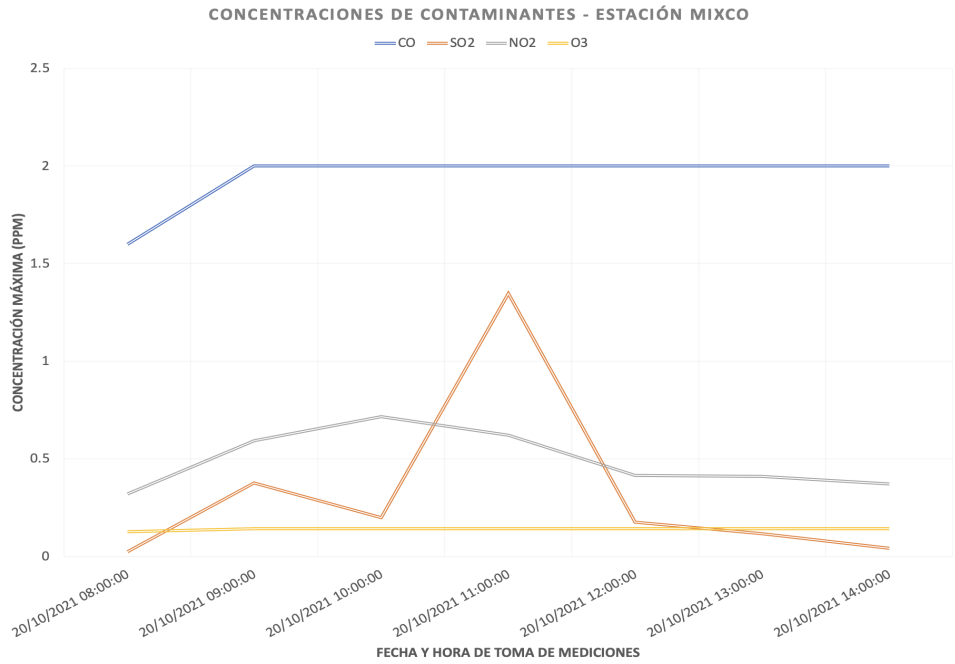


Figura 9.4: Contaminantes obtenidos para la estación de Mixco agrupadas por hora.

En cada una de las estaciones, se puede observar el ozono (O_3) es el único gas que se mantiene en concentraciones muy bajas y con poca variación; esto se debe a que el ozono, aunque es uno de los gases que conforman la atmósfera y es natural tenerlo presente, es altamente tóxico en la superficie

terrestre, por lo que se considera congruente las cifras reportadas en ventanas de 8 horas para este gas. El siguiente índice de gas se calcula en ventanas de 8 horas es el monóxido de carbono (CO) el cual es bastante común en áreas urbanizadas ya que es producido en la combustión o quema de material orgánico – como madera, la gasolina en vehículos o estufas – y se espera que esté en mayores concentraciones que los demás gases; su concentración guía es de 9ppm en promedio de 8 horas. Los contaminantes de dióxido de azufre (SO_2) y dióxido de nitrógeno (NO_2) ambos son calculados en ventanas de 1 hora y también son resultado de la quema de combustibles fósiles que contienen azufre y nitrógeno, pero tienen concentraciones guía bajas, de 0.2ppm y 0.1ppm, respectivamente. Los valores de los dos gases anteriores que la infraestructura reportó se encuentran alrededor de las concentraciones guía, por lo que se estima que los valores obtenidos son correctos. Cabe resaltar que el sensor de SO_2 tiene la particularidad de fábrica que cuando se coloca en una ubicación cerrada (por ejemplo, una habitación), se reportan concentraciones relativamente elevadas; esto explica los picos de este gas que se observan en las figuras.

Adicionalmente, durante el período de prueba se obtuvo la diferencia entre el momento en el que una medición fue registrada y enviada por los sensores y el momento en el que la medición era recibida por Databricks; esta diferencia daría una referencia de qué tanta era la demora para la transferencia de datos y evaluar si IoT Hub era un facilitador o, si la demora era demasiada, era más bien un obstáculo.

	Estación Centro Histórico	Estación Carretera a El Salvador	Estación Mixco
Diferencia promedio entre momento de medición y llegada a IoT Hub	1.52 segundos	1.67 segundos	1.06 segundos
Diferencia promedio entre llegada a IoT Hub y recepción de dato en Databricks	0.52 segundos	0.54 segundos	0.54 segundos
Diferencia promedio total entre el momento de medición y recepción en Databricks	2.04 segundos	2.21 segundos	1.60 segundos

Tabla 9.1: Diferencias de tiempo en la transferencia de datos de la fase previa al procesamiento.

El cuadro anterior muestra el tiempo promedio que se toma una medición en viajar desde el sensor hasta IoT Hub, luego desde IoT Hub a Databricks para ser procesado y por último el tiempo promedio que se toma en hacer el viaje completo. En general, el tiempo promedio del viaje para una medición es de 1.95 segundos, por lo que esta transferencia es considerada rápida.

De acuerdo con los resultados anteriores, las mediciones obtenidas por los sensores son valores coherentes y se considera la comunicación entre los dispositivos y la infraestructura en la nube exitosa y eficaz.

El siguiente paso en el flujo de datos es su procesamiento, para esto se calculó el Índice de Calidad de Aire en ventanas de 1 y 8 horas – dependiendo del contaminante – cada 1 hora, utilizando la Ecuación 4.1. Similar a los resultados en cuanto a las concentraciones recibidas, en las siguientes figuras se visualizan los valores AQI calculados para cada contaminante, en cada estación. En las mismas también se grafica el AQI general, el cual representa el AQI máximo de la combinación de los 4 gases evaluados, se puede observar en las figuras que este AQI general aumenta y disminuye conforme lo hace el AQI del gas más alto.

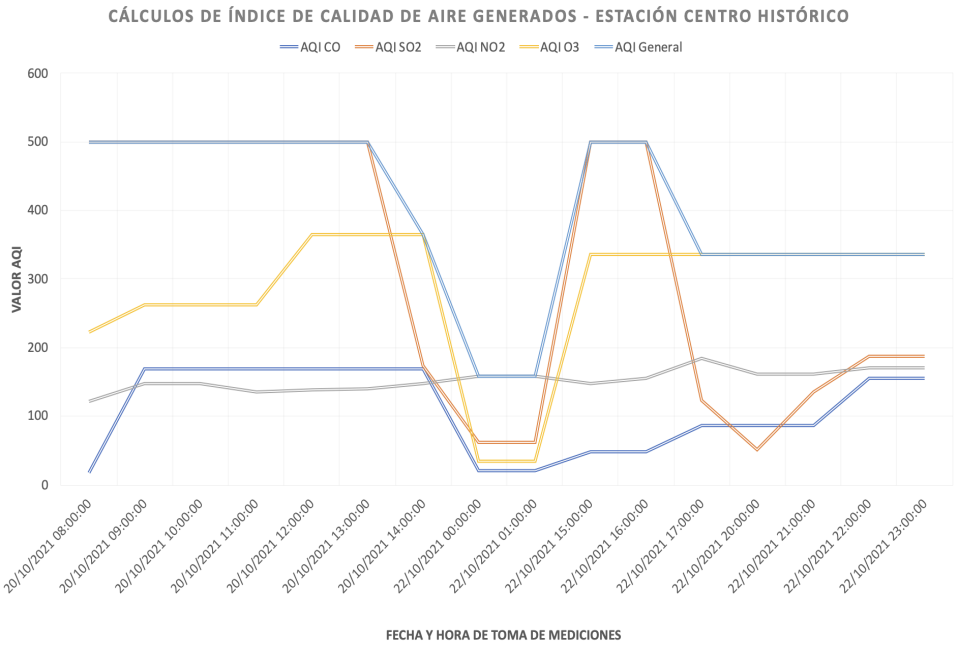


Figura 9.5: AQI obtenido por hora de medición para la estación del Centro Histórico.

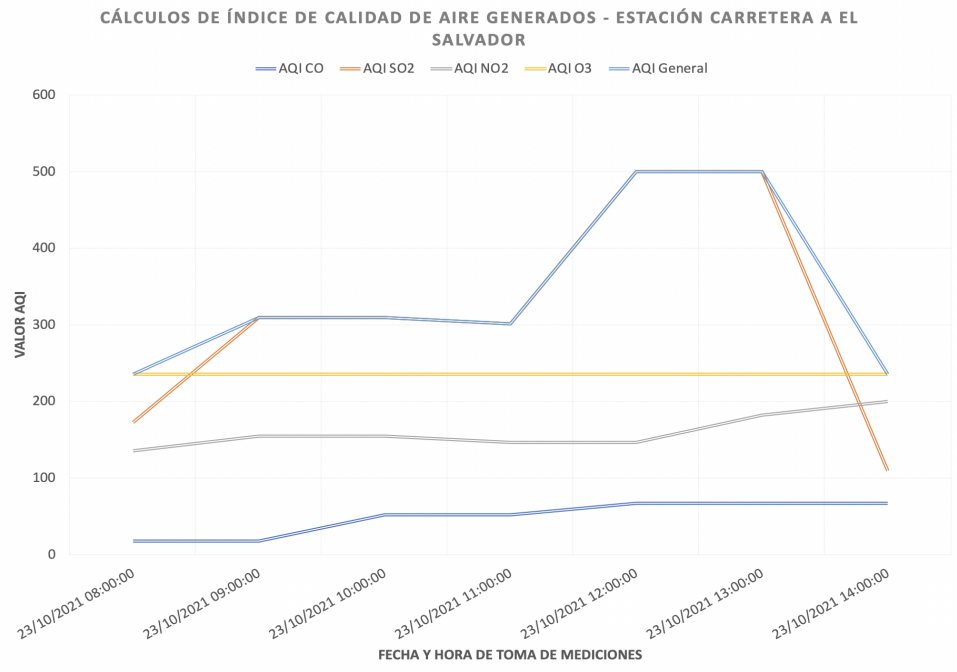


Figura 9.6: AQI obtenido por hora de medición para la estación de Carretera a El Salvador.

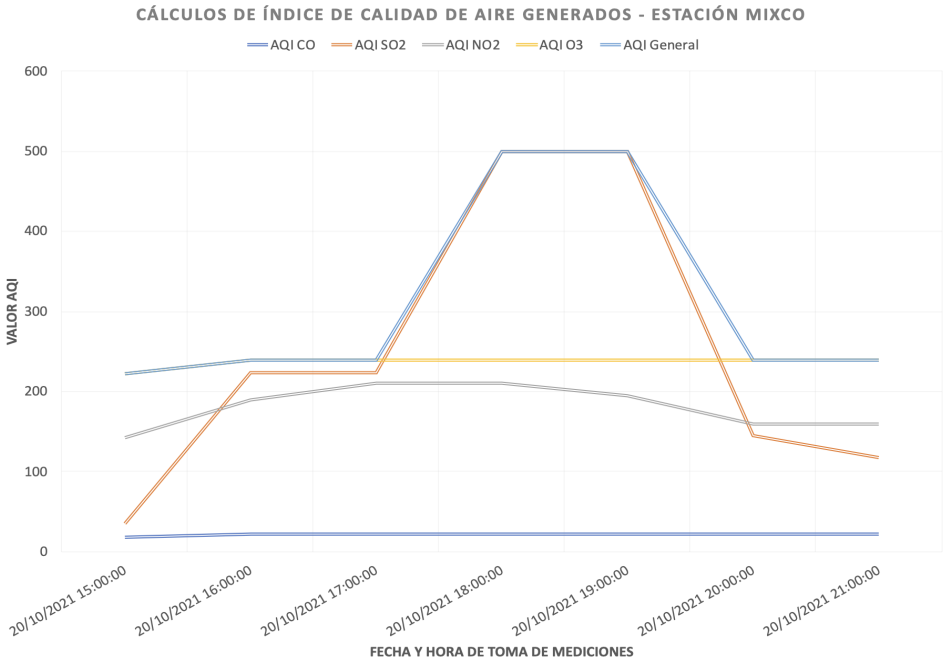


Figura 9.7: AQI obtenido por hora de medición para la estación de Mixco.

Adicional a las visualizaciones anteriores, se realizó una comparación de los datos del AQI obtenidos en las pruebas con los valores AQI dados por la calculadora de Índice de Calidad de Aire oficial de la Agencia de Protección Ambiental de Estados Unidos, estos últimos fueron considerados como los valores teóricos y, utilizando la Ecuación 7.13, se generó un porcentaje de error para evaluar qué tan acertado fue el cálculo generado por nuestra infraestructura. Los siguientes cuadros muestran los valores AQI experimentales – es decir, los índices calculados por la herramienta de Databricks – para cada contaminante, cada estación y cada hora de período de la prueba y los valores AQI teóricos obtenidos con la calculadora previamente mencionada. Por último, se muestra el porcentaje de error obtenido en la comparación de los valores experimentales y teóricos.

Estación	Fecha y hora de medición	AQI CO	AQI SO_2	AQI NO_2	AQI O_3
Centro Histórico	20/10/2021 08:00:00	18	500	122	222
Centro Histórico	20/10/2021 09:00:00	169	500	148	263
Centro Histórico	20/10/2021 10:00:00	169	500	148	263
Centro Histórico	20/10/2021 11:00:00	169	500	136	263
Centro Histórico	20/10/2021 12:00:00	169	500	138	365
Centro Histórico	20/10/2021 13:00:00	169	500	140	365
Centro Histórico	20/10/2021 14:00:00	169	174	147	365
Centro Histórico	22/10/2021 00:00:00	21	157	158	35
Centro Histórico	22/10/2021 01:00:00	21	62	158	35
Centro Histórico	22/10/2021 15:00:00	48	500	147	336
Centro Histórico	22/10/2021 16:00:00	48	34	156	336
Centro Histórico	22/10/2021 17:00:00	86	123	184	336
Centro Histórico	22/10/2021 20:00:00	86	51	162	336
Centro Histórico	22/10/2021 21:00:00	86	136	162	336
Centro Histórico	22/10/2021 22:00:00	156	187	170	336
Centro Histórico	22/10/2021 23:00:00	156	187	170	336
Carretera a El Salvador	23/10/2021 08:00:00	17	172	136	235
Carretera a El Salvador	23/10/2021 09:00:00	17	500	155	235
Carretera a El Salvador	23/10/2021 10:00:00	52	500	155	235
Carretera a El Salvador	23/10/2021 11:00:00	52	500	147	235
Carretera a El Salvador	23/10/2021 12:00:00	67	500	147	235
Carretera a El Salvador	23/10/2021 13:00:00	67	109	182	235
Carretera a El Salvador	23/10/2021 14:00:00	67	51	200	235
Mixco	20/10/2021 15:00:00	18	35	142	222
Mixco	20/10/2021 16:00:00	22	224	190	239
Mixco	20/10/2021 17:00:00	22	156	211	239
Mixco	20/10/2021 18:00:00	22	500	211	239
Mixco	20/10/2021 19:00:00	22	145	195	239
Mixco	20/10/2021 20:00:00	22	145	160	239
Mixco	20/10/2021 21:00:00	22	59	159	239

Tabla 9.2: AQIs experimentales resultantes de la segunda ronda de pruebas.

Estación	Fecha y hora de medición	AQI CO	AQI SO_2	AQI NO_2	AQI O_3
Centro Histórico	20/10/2021 08:00:00	18	500	122	0
Centro Histórico	20/10/2021 09:00:00	170	500	149	0
Centro Histórico	20/10/2021 10:00:00	170	500	133	0
Centro Histórico	20/10/2021 11:00:00	170	500	136	0
Centro Histórico	20/10/2021 12:00:00	170	500	138	0
Centro Histórico	20/10/2021 13:00:00	170	500	140	0
Centro Histórico	20/10/2021 14:00:00	170	174	147	0
Centro Histórico	22/10/2021 00:00:00	20	158	159	0
Centro Histórico	22/10/2021 01:00:00	20	62	158	0
Centro Histórico	22/10/2021 15:00:00	49	500	148	0
Centro Histórico	22/10/2021 16:00:00	49	34	156	0
Centro Histórico	22/10/2021 17:00:00	85	124	184	0
Centro Histórico	22/10/2021 20:00:00	85	50	162	0
Centro Histórico	22/10/2021 21:00:00	85	50	157	0
Centro Histórico	22/10/2021 22:00:00	154	187	170	0
Centro Histórico	22/10/2021 23:00:00	154	168	170	0
Carretera a El Salvador	23/10/2021 08:00:00	16	172	137	0
Carretera a El Salvador	23/10/2021 09:00:00	16	500	156	0
Carretera a El Salvador	23/10/2021 10:00:00	52	500	140	0
Carretera a El Salvador	23/10/2021 11:00:00	52	500	148	0
Carretera a El Salvador	23/10/2021 12:00:00	66	500	144	0
Carretera a El Salvador	23/10/2021 13:00:00	66	110	182	0
Carretera a El Salvador	23/10/2021 14:00:00	66	51	200	0
Mixco	20/10/2021 15:00:00	18	36	143	0
Mixco	20/10/2021 16:00:00	23	500	190	0
Mixco	20/10/2021 17:00:00	23	156	212	0
Mixco	20/10/2021 18:00:00	23	500	325	0
Mixco	20/10/2021 19:00:00	23	146	115	0
Mixco	20/10/2021 20:00:00	23	119	104	0
Mixco	20/10/2021 21:00:00	23	60	153	0

Tabla 9.3: AQIs teóricos de la segunda ronda de pruebas.

Estación	Fecha y hora de medición	AQI CO	AQI SO_2	AQI NO_2	AQI O_3
Centro Histórico	20/10/2021 08:00:00	0	0	0	100
Centro Histórico	20/10/2021 09:00:00	0.58823529	0	0.67114094	100
Centro Histórico	20/10/2021 10:00:00	0.58823529	0	11.2781955	100
Centro Histórico	20/10/2021 11:00:00	0.58823529	0	0	100
Centro Histórico	20/10/2021 12:00:00	0.58823529	0	0	100
Centro Histórico	20/10/2021 13:00:00	0.58823529	0	0	100
Centro Histórico	20/10/2021 14:00:00	0.58823529	0	0	100
Centro Histórico	22/10/2021 00:00:00	5	0.632911392	0.62893082	100
Centro Histórico	22/10/2021 01:00:00	5	0	0	100
Centro Histórico	22/10/2021 15:00:00	2.04081633	0	0.67567568	100
Centro Histórico	22/10/2021 16:00:00	2.04081633	0	0	100
Centro Histórico	22/10/2021 17:00:00	1.17647059	0.806451613	0	100
Centro Histórico	22/10/2021 20:00:00	1.17647059	2	0	100
Centro Histórico	22/10/2021 21:00:00	1.17647059	172	3.18471338	100
Centro Histórico	22/10/2021 22:00:00	1.2987013	0	0	100
Centro Histórico	22/10/2021 23:00:00	1.2987013	11.30952381	0	100
Carretera a El Salvador	23/10/2021 08:00:00	6.25	0	0.72992701	100
Carretera a El Salvador	23/10/2021 09:00:00	6.25	0	0.64102564	100
Carretera a El Salvador	23/10/2021 10:00:00	0	0	10.7142857	100
Carretera a El Salvador	23/10/2021 11:00:00	0	0	0.67567568	100
Carretera a El Salvador	23/10/2021 12:00:00	1.51515152	0	2.08333333	100
Carretera a El Salvador	23/10/2021 13:00:00	1.51515152	0.909090909	0	100
Carretera a El Salvador	23/10/2021 14:00:00	1.51515152	0	0	100
Mixco	20/10/2021 15:00:00	0	2.777777778	0.6993007	100
Mixco	20/10/2021 16:00:00	4.34782609	55.2	0	100
Mixco	20/10/2021 17:00:00	4.34782609	0	0.47169811	100
Mixco	20/10/2021 18:00:00	4.34782609	0	35.0769231	100
Mixco	20/10/2021 19:00:00	4.34782609	0.684931507	69.5652174	100
Mixco	20/10/2021 20:00:00	4.34782609	21.8487395	53.8461538	100
Mixco	20/10/2021 21:00:00	4.34782609	1.666666667	3.92156863	100

Tabla 9.4: Porcentaje de error en el cálculo del Índice de Calidad de Aire.

Para los contaminantes de CO, SO_2 y NO_2, se puede observar que los porcentajes de error obtenidos son relativamente bajos, el promedio entre todas las estaciones para cada contaminante siendo de 2.23 %, 8.99 % y 6.45 %, respectivamente. Esta diferencia entre cálculos se atribuye principalmente al hecho que las mediciones, previo a entrar como parámetros del cálculo del AQI, se aproximan a cierta cantidad de decimales – dependiendo del contaminante – y pequeñas diferencias en decimales puede ocasionar que el resultado experimental difiera del teórico por una o dos unidades. Entre los tres contaminantes previos, se obtuvo en promedio un porcentaje de error de 5.91 %, lo que se consideró como aceptable.

Por otro lado, el porcentaje de error obtenido para el contaminante de O_3 fue un caso particular. Evidentemente, un porcentaje de error del 100 % indican que ni uno de los cálculos experimentales estuvo por lo menos aproximado a su valor teórico. En realidad, este valor porcentual no es de 100 %, el resultado verdadero era un error matemático; esto debido a que, al momento de calcular

el porcentaje, el valor teórico en todos los casos fue de 0 y una división dentro de 0 resulta siempre en error matemático. La razón por la cual todos los valores AQI teóricos fueron 0 es porque la calculadora utilizada consideraba el ozono (O_3) como un contaminante que se reporta con magnitudes de partículas por billón (ppb), a pesar de que en documento técnico para calcular el Índice de Calidad del Aire especifica que este se reporta en magnitudes de partículas por millón (ppm). Como se puede observar en las Figuras 9.2-9.4, el ozono en partículas por millón resulta en mediciones menores a 1 (como 0.02, 0.59. etc); al ingresar valores de esa escala a la calculadora – la cual esperaba valores en escala de decenas y hasta cientos – el AQI resultante siempre era 0. A pesar de lo anterior, debido a los bajos porcentajes de error obtenidos, se consideró que el pipeline de procesamiento de datos implementado realizó los cálculos esperados de manera exitosa.

Por último, del almacén de datos – la base de datos SQL Server – se esperaba que pudiera soportar escrituras y lecturas continuas de manera paralela; en otras palabras, no debía dar error de conexión si Databricks estaba constantemente escribiendo las concentraciones de los contaminantes y al mismo tiempo que la página web estuviera realizando lecturas constantes sobre las tablas de la base de datos, sin importar qué tan frecuente fueran las escrituras y las lecturas y sin falla durante las múltiples horas del período de prueba. La siguiente figura grafica la cantidad de conexiones exitosas recibidas en SQL Server entre el 19 de octubre del presente año y el 23 de octubre.

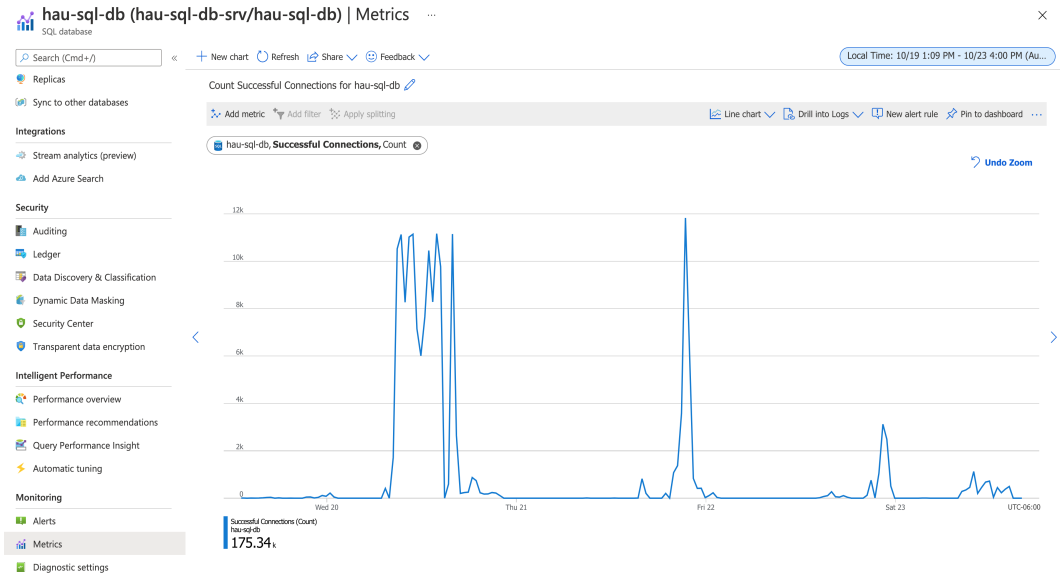


Figura 9.8: Cantidad de conexiones exitosas hacia SQL Server en el período de prueba.

Se observa que existen períodos en donde la base de datos recibía una cantidad de solicitudes de conexión atípica, estos picos llegan hasta 11,000 conexiones en un momento dado (no necesariamente simultáneas). Dichas cúspides se mantienen por varias horas hasta descender y mantenerse bajas por días antes de elevarse de nuevo. Estos cambios representan las horas de los días que se tuvo Databricks activo y en ejecución al mismo tiempo que la página web actualizaba sus visualizaciones con los datos generados. Como complemento, la siguiente figura muestra el porcentaje de CPU que SQL Server tiene en uso para un momento dado en el período de tiempo del 19 de octubre del presente año y el 23 de octubre.

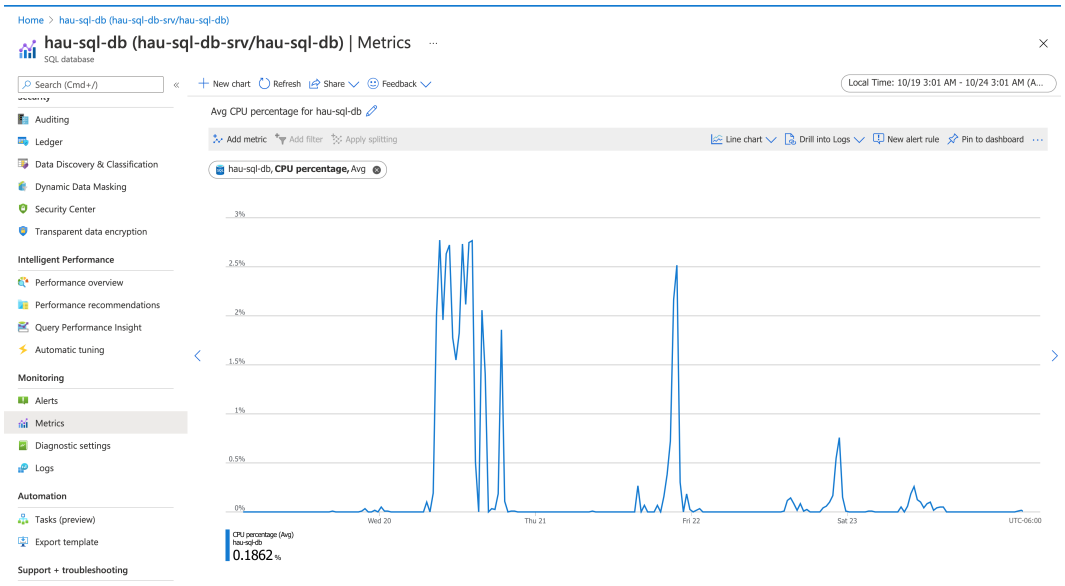


Figura 9.9: Capacidad computacional aprovisionada en SQL Server para ejecutar solicitudes.

Se puede ver que la Figura 9.8 es bastante parecida a la Figura 9.9, indicando que los períodos de tiempo en los que SQL Server ejecutaba queries – y, por tanto, utilizaba recursos computacionales – son los mismos períodos de tiempo en los que se solicitan conexiones. Esto confirma que la base de datos es capaz de acomodarse a cualquier cantidad de solicitudes de lectura/escritura por repentino que sea y provee los recursos necesarios para atender estas solicitudes correctamente y los desactiva o desmantela cuando ya no son necesarios.

CAPÍTULO 10

Resultados: plataforma web

10.1. Design Thinking

Como parte del proceso de Design Thinking se obtuvieron los siguientes mapas de empatía en donde se puede apreciar que los usuarios reconocen la importancia de la medición y análisis de gases contaminantes no solo para quienes se dedican a realizar estudios ambientales, si no también, para público en general para prevenir afecciones o problemas de salud relacionados con el tema. A través de los siguientes cuadros de empatía, se puede observar las distintas retroalimentaciones obtenidas según los tipos de usuario definidos:

- Lo que piensa y siente.
- Lo que dice y hace.
- Lo que ve.
- Lo que oye.

Esto se pregunta con el fin de poder obtener un mejor entendimiento respecto a la contaminación aérea en Guatemala, y que tan informado o no se sienten los perfiles de usuario definidos.

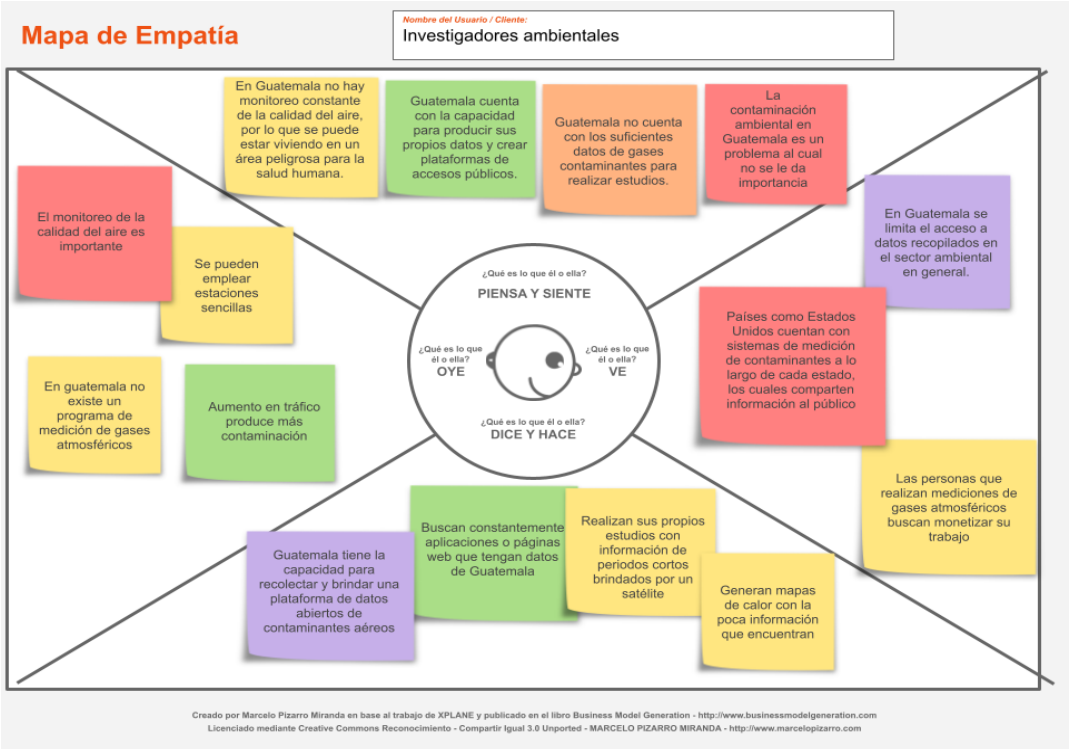


Figura 10.1: Mapa de empatía para investigadores ambientales

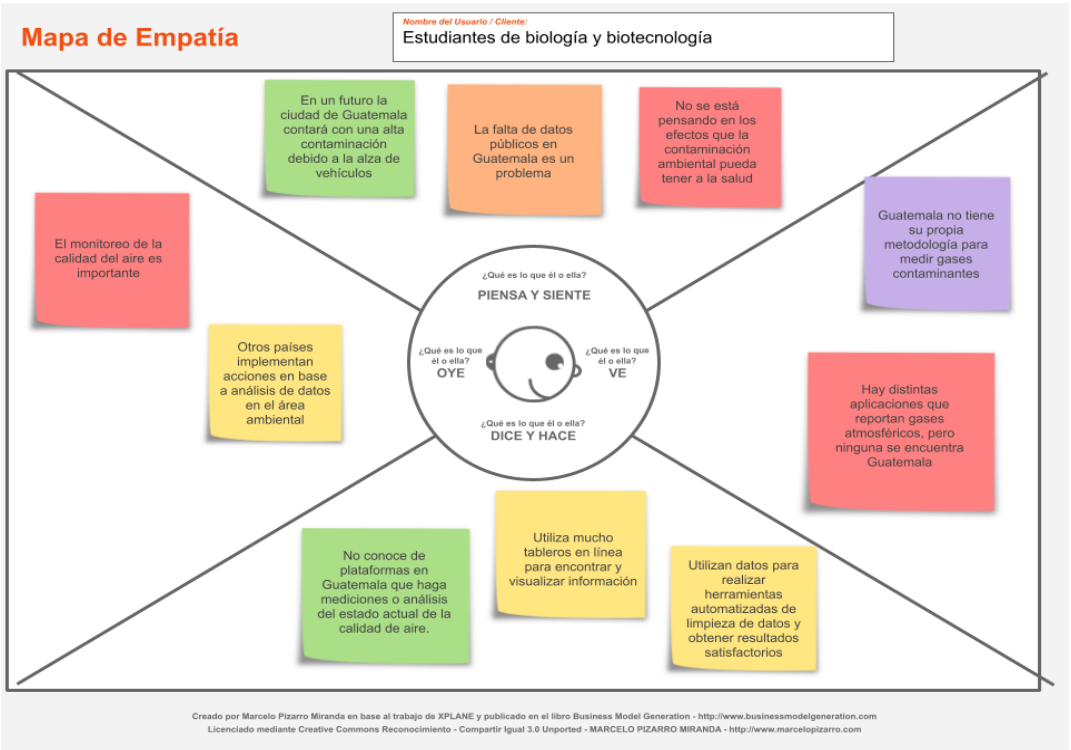


Figura 10.2: Mapa de empatía estudiantes de biología y biotecnología

Los usuarios que se definieron para el proyecto se encuentran en la siguiente tabla.

Perfiles de usuarios	
Nombre	Descripción
Investigadores	Personas afines al área ambiental interesadas en explorar datos de gases contaminantes y realizar análisis diarios de la calidad del aire
Usuarios comunes	Personas no afines al área ambiental que visiten la plataforma para conocer u informarse sobre el estado actual de la calidad del aire

Tabla 10.1: Perfiles de usuarios

Entre las observaciones y percepciones obtenidas de las entrevistas y grupos focales se encontraron las siguientes necesidades para ambos perfiles definidos.

1. Poder obtener los datos recolectados para un análisis futuro a fondo.
2. Visualizar el estado de contaminación del aire en un sector dado.
3. Obtener información de cambios en las concentraciones de los gases contaminantes medidos.
4. Identificar que sectores del país cuentan con una estación de medición de gases atmosféricos.

Para lograr definir los casos de uso puntuales se realizó una investigación para obtener información sobre como visualizar y analizar la calidad del aire, obteniendo que el índice de calidad del aire, AQI por sus siglas en inglés, es una medida para demostrar la calidad del aire en un ambiente tomando en cuenta las concentraciones de todos los gases que se tengan disponibles. Para esto se tienen disponibles las concentraciones de cuatro gases, las cuales son medidas a través del Módulo de Internet de las cosas con la creación de una estación de medición. El índice de calidad del aire cuenta con distintos rangos en los cuales se describen las condiciones del ambiente y las posibles consecuencias para la población en general [43]. Los rangos y descripciones que se utilizaron dentro en el proyecto son los definidos por el INSIVUMEH en [43].

Valor de ICA	Color	Calidad del Aire	Significado
0 - 50	Verde	Buena	La Calidad del aire es considerada satisfactoria.
51 - 100	Amarillo	Moderada	La Calidad del Aire es aceptable, sin embargo, algunos contaminantes pueden generar un efecto moderado en la salud de un muy pequeño número de personas usualmente sensibles a la contaminación del aire.
101 - 150	Naranja	No saludable para grupos sensibles	Miembros de grupos sensibles pueden experimentar efectos sobre su salud. El público general no es usualmente afectado.
151 - 200	Rojo	Insalubre	Cualquier persona comienza a experimentar efectos sobre su salud. Miembros de grupos sensibles pueden experimentar efectos más serios sobre su salud.
201 – 300	Púrpura	Muy Insalubre	Alerta de Salud: cualquier persona puede experimentar efectos serios sobre su salud.
301 - 500	Granate	Peligrosa	Advertencia de Condición de Emergencia: La salud de la población entera está en riesgo de ser afectada.

Figura 10.3: Rangos de índice de calidad del aire.

Fuente: INSIVUMEH

Dadas las necesidades encontradas se definieron los siguientes casos de uso para los actores del proyecto.

Caso de uso 1	Visualizar datos recopilados
Actor	Investigadores y Usuario Común
Flujo básico	El usuario ingresa a la plataforma y selecciona visualizar datos. El usuario selecciona que gases contaminantes, temperatura, humedad, estaciones y las fechas de inicio y fin de los datos que desea ver.

Tabla 10.2: Caso de uso: Visualizar datos recopilados

Caso de uso 2	Descargar datos recopilados
Actor	Investigadores y Usuario Común
Flujo básico	El usuario realizó el flujo básico del caso de uso de visualizar datos recopilados. El usuario selecciona el formato en el cual desea descargar sus datos y lo almacena en su computadora.

Tabla 10.3: Caso de uso: Descargar datos recopilados

Caso de uso 3	Analizar cambios en gases atmosféricos
Actor	Investigadores
Flujo básico	El investigador observa los datos de los diferentes gases atmosféricos. Para realizar el análisis de las concentraciones en los cambios selecciona los gases que desea ver en pantalla, el tiempo entre el cambio de cada concentración y la estación que desea analizar.

Tabla 10.4: Caso de uso: Visualizar datos recopilados

Caso de uso 4	Inspeccionar estaciones de medición
Actor	Investigadores
Flujo básico	El investigador selecciona la estación de la cual desea conocer toda la información. Cuando el investigador selecciona la estación se le despliegan en pantalla datos específicos de esta.

Tabla 10.5: Caso de uso: Inspeccionar estaciones de medición

Caso de uso 5	Visualizar calidad del aire
Actor	Investigadores y Usuario común
Flujo básico	Tanto el investigador como el usuario, ingresan a la plataforma y observan el rango en el que la calidad del aire se encuentra para dicho momento. Seguido de esto los usuarios observan recomendaciones de actividades que pueden realizar en función al rango del índice de calidad del aire en el que se encuentra.

Tabla 10.6: Caso de uso: Visualizar calidad del aire

10.2. API

Se desarrollaron en total 10 rutas de tipo POST dentro de la interfaz de programación, las cuales se describe su función a continuación.

Ruta	Parámetros	Función
/api/v1/sensors		Obtener la longitud, latidud y nombre de las estaciones activas.
/api/v1/gases	Gases, Temperatura, Humedad, Estación, Fecha de inicio y Fecha de fin	Devolver toda la información incluyendo fecha, medida de contaminante obtenida, temperatura, porcentaje de humedad y estación según los de entrada. Esta ruta es utilizada para mostrar los datos recopilados al usuario, los datos de entrada son los filtros seleccionados por él.
/api/v1/ozone	Estación	Obtener los valores de las concentraciones de ozono para la estación elegida por el usuario.
/api/v1/climates		Obtener la última información registrada de temperatura y humedad.
/api/v1/graph	Gases y Estación	Obtener el promedio de concentración registrada de los gases seleccionados en la estación preferida, por las ultimas 7 horas.
/api/v1/graphDay	Gases y Estación	Obtener el promedio de concentración registrada de los gases seleccionados en la estación preferida, por los últimos 7 días.
/api/v1/maps	Estación	Obtener las distintas estaciones con su respectiva información de aqi, temperatura, humedad, latitud, longitud, nombre completo de la estación
/api/v1/aqis		Obtener el índice de calidad de aire del día, su rango, descripción y recomendaciones basados en el valor del índice.
/api/v1/aqio	Estación	Obtener los valores del aqi del ozono por los últimos 20 días.
/api/v1/individuals		Obtener los valores más recientes reportados para cada gas contaminante almacenado.

Tabla 10.7: Rutas disponibles en interfaz de programación

Esta se encuentra disponible a través del enlace https://hauair.site/. Los resultados que se devuelven al acceder a las rutas se encuentran en formato JSON.

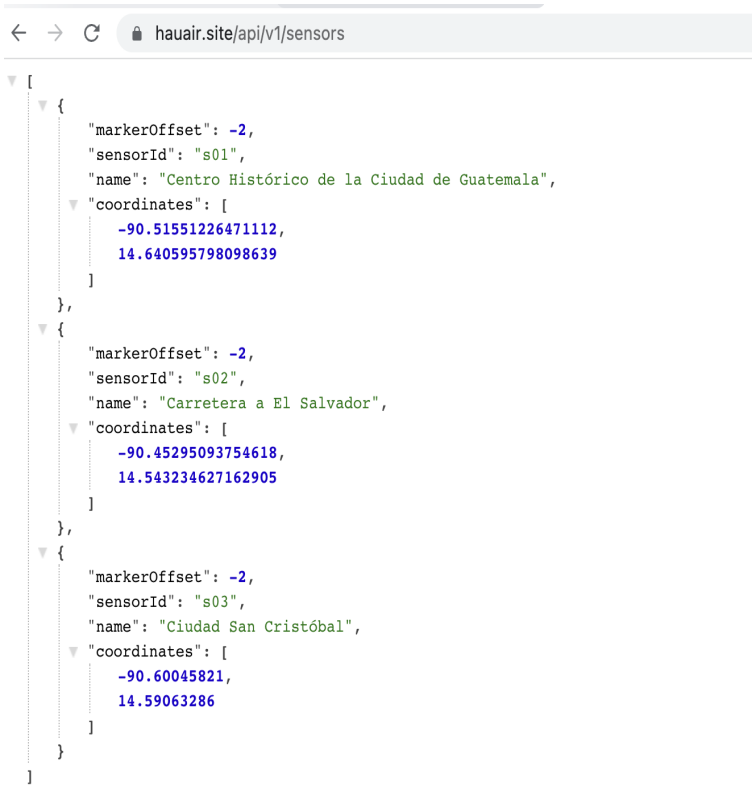


Figura 10.4: Ejemplo de resultados devueltos por interfaz de programación

10.3. Sistemas en la nube

Terminada la interfaz de programación y configurada la máquina virtual en AWS se obtuvo como resultados los detalles de esta.

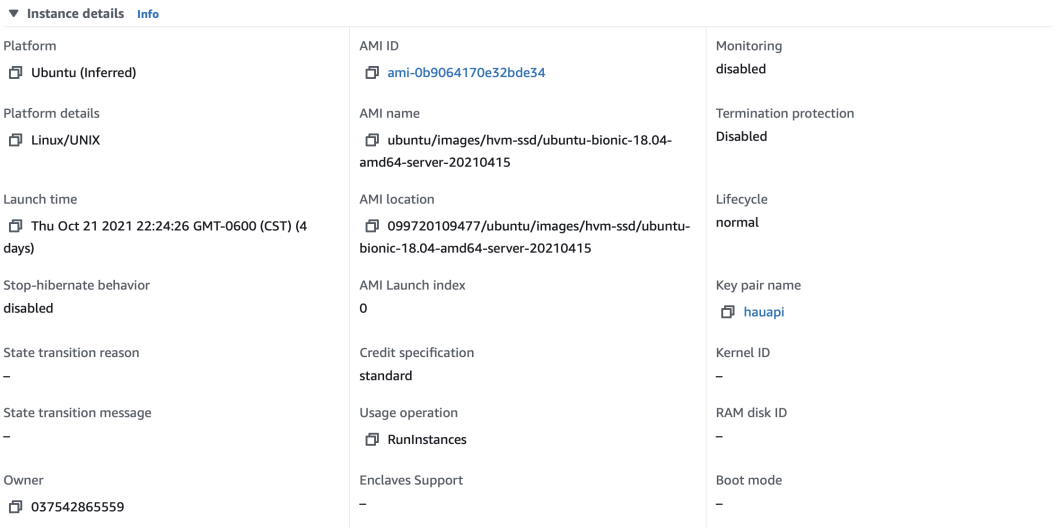


Figura 10.5: Detalles de Instancia EC2

La configuración de Nginx con la implementación de Reverse Proxy quedo de como se muestra en la siguiente figura.

```
# configuration file /etc/nginx/conf.d/default.conf:

log_format custom_log '"Request: $request\n Status: $status\n Request_URI: $request_uri\n Host: $host\n Clie
nt_IP: $remote_addr\n Proxy_IP(s): $proxy_add_x_forwarded_for\n Proxy_Hostname: $proxy_host\n Real_IP: $http
_x_real_ip\n User_Client: $http_user_agent"';


server {

    listen       443 ssl;
    server_name  hauair.site;
    root        /var/html/index.html;
        access_log /var/log/nginx/custom-access-logs.log custom_log;
    ssl_certificate      /etc/letsencrypt/live/hauair.site/fullchain.pem;
ssl_certificate_key  /etc/letsencrypt/live/hauair.site/privkey.pem;

        proxy_set_header Host $host;
             proxy_set_header X-Real-IP $remote_addr;
             proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
             add_header Test_header jay;


    location / {
        proxy_pass http://127.0.0.1:3000;

    }

}
```

Figura 10.6: Archivo de configuración Nginx para la implementación de Reverse Proxy

La implementación del certificado de seguridad demuestra que esta asociado al dominio del sitio obtenido, indicando que este es seguro y valido.

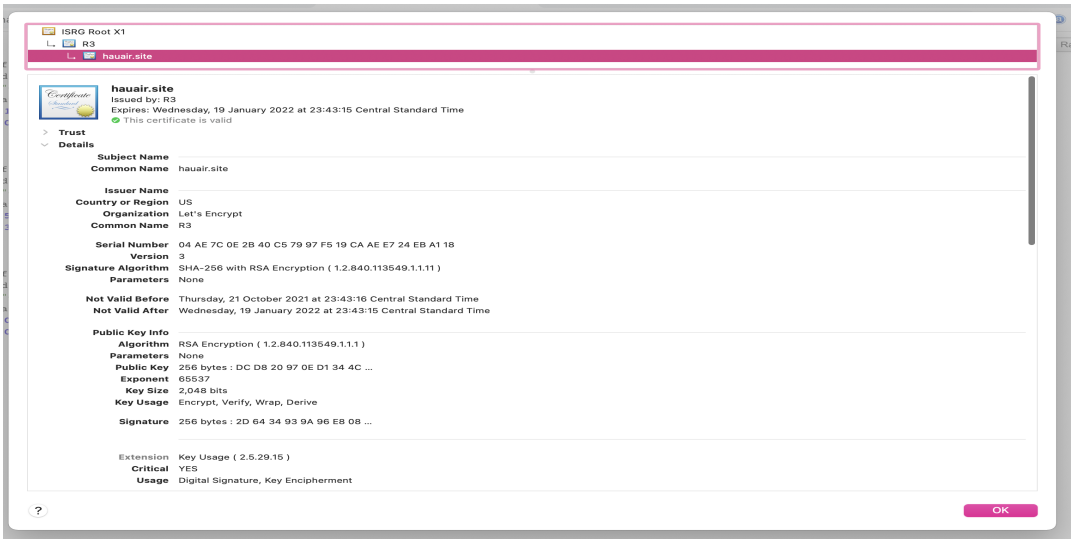


Figura 10.7: Detalles de certificado de seguridad para dominio hauair.site

10.4. Front-End

Utilizando el último prototipo elaborado se llevó a cabo la definición de componentes y su debida jerarquía para ser implementados posteriormente en React

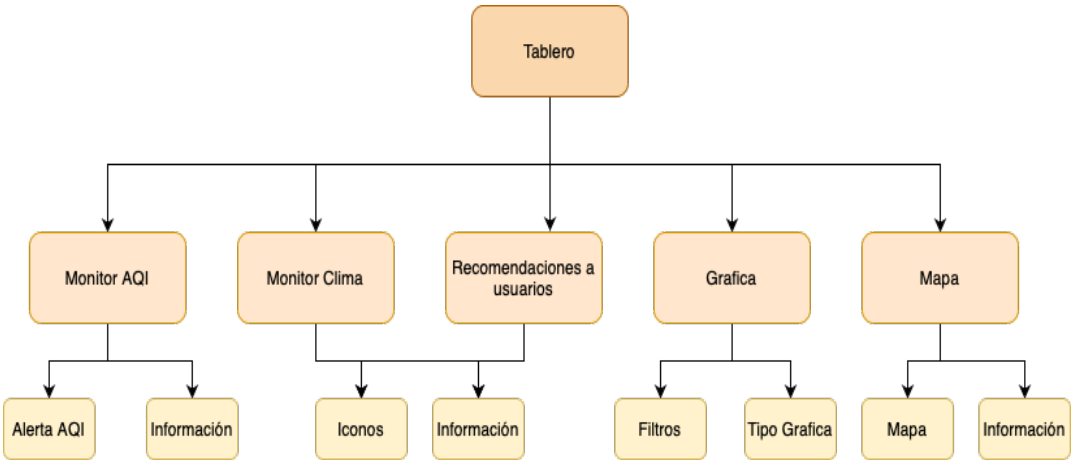


Figura 10.8: Jerarquía de componentes para pantalla Tablero

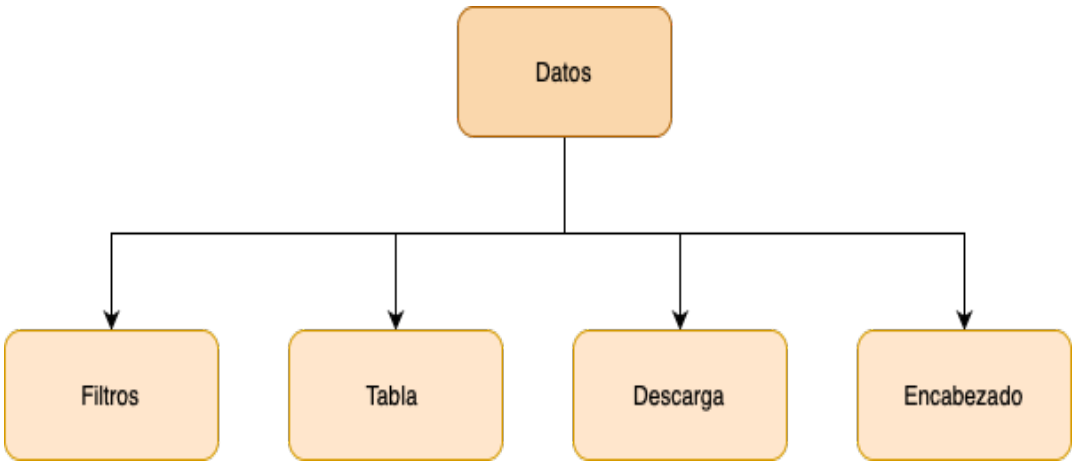


Figura 10.9: Jerarquía de componentes para pantalla Datos

Tras haber implementando los componentes con sus respectivos estilos y utilizando las rutas de la interfaz de programación para obtener los datos se obtuvo la siguiente página web que consta de dos pantallas.

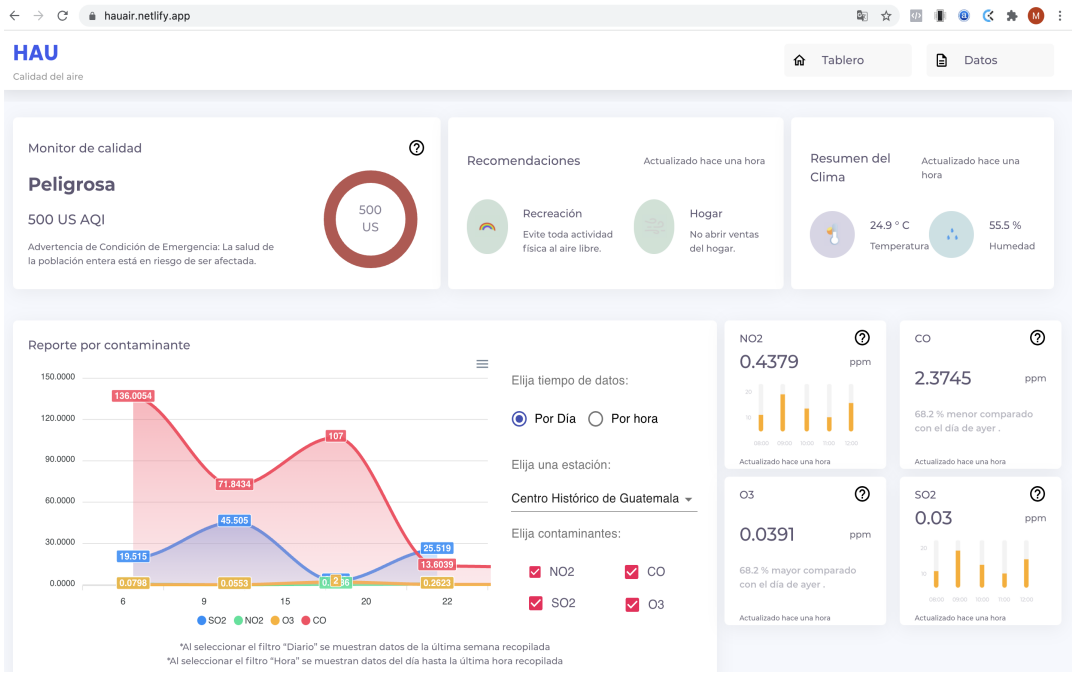


Figura 10.10: Pantalla Tablero página web

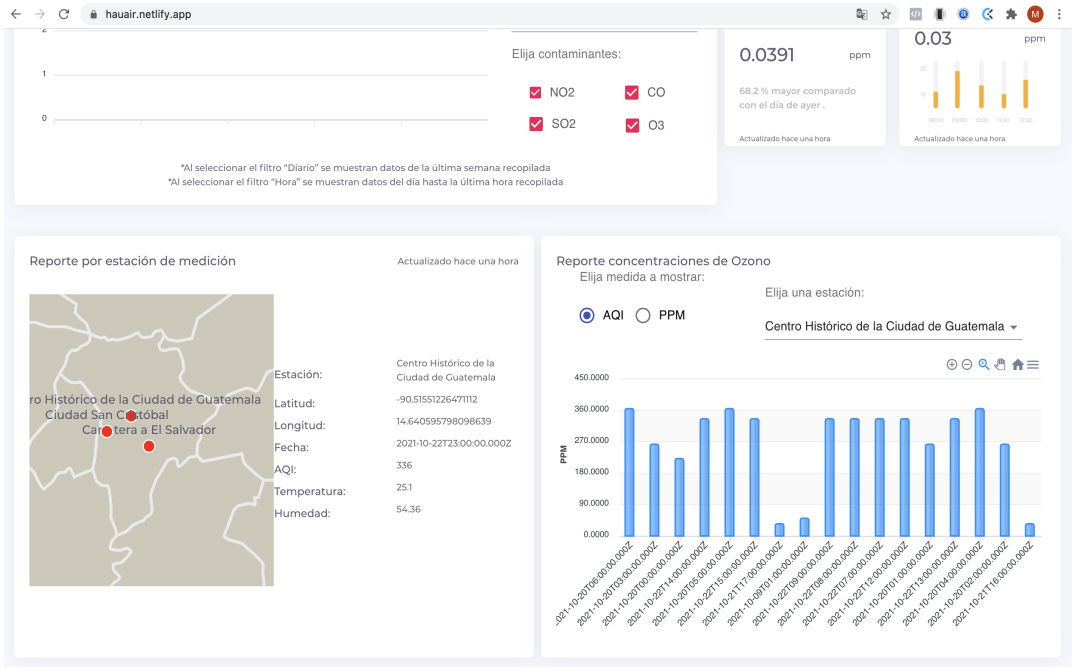
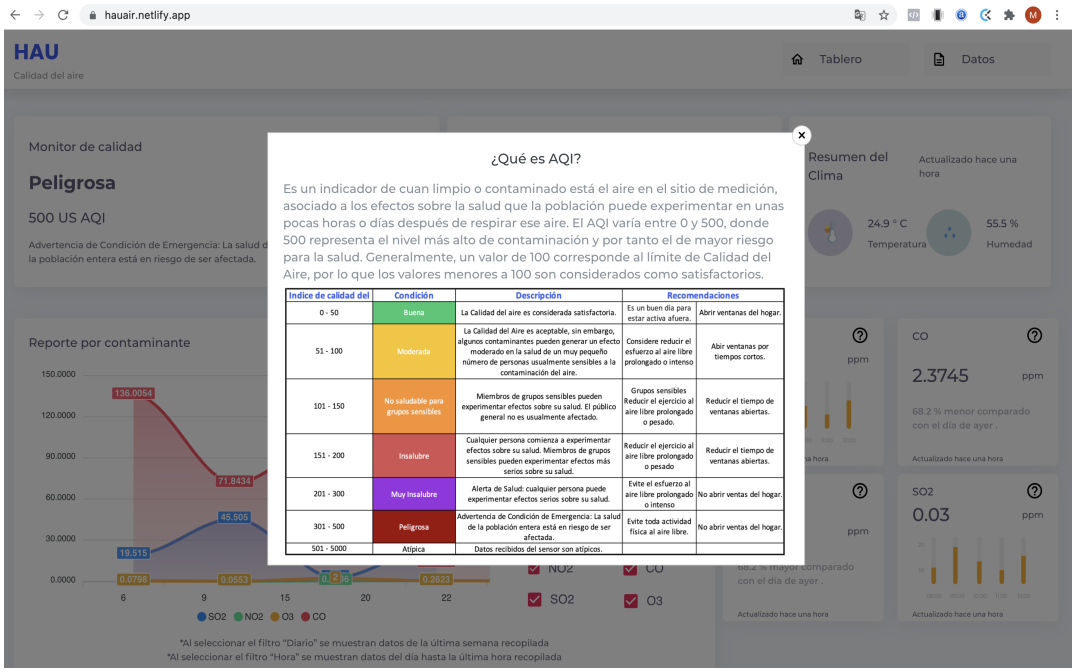


Figura 10.11: Pantalla Tablero página web



Indice de calidad del	Condición	Descripción	Recomendaciones	
0 - 50	Buena	La Calidad del aire es considerada satisfactoria.	Es un buen día para estar activa afuera.	Abrir ventanas del hogar.
51 - 100	Moderada	La Calidad del Aire es aceptable, sin embargo, algunos contaminantes pueden generar un efecto moderado en la salud de un muy pequeño número de personas usualmente sensibles a la contaminación del aire.	Considere reducir el esfuerzo al aire libre prolongado o intenso	Abir ventanas por tiempos cortos.
101 - 150	No saludable para grupos sensibles	Miembros de grupos sensibles pueden experimentar efectos sobre su salud. El público general no es usualmente afectado.	Grupos sensibles Reducir el ejercicio al aire libre prolongado o pesado.	Reducir el tiempo de ventanas abiertas.
151 - 200	Insalubre	Cualquier persona comienza a experimentar efectos sobre su salud. Miembros de grupos sensibles pueden experimentar efectos más serios sobre su salud.	Reducir el ejercicio al aire libre prolongado o pesado	Reducir el tiempo de ventanas abiertas.
201 - 300	Muy Insalubre	Alerta de Salud: cualquier persona puede experimentar efectos serios sobre su salud.	Evite el esfuerzo al aire libre prolongado o intenso	No abrir ventas del hogar.
301 - 500	Peligrosa	Advertencia de Condición de Emergencia: La salud de la población entera está en riesgo de ser afectada.	Evite toda actividad física al aire libre.	No abrir ventas del hogar.
501 - 5000	Atípica	Datos recibidos del sensor son atípicos.		

Figura 10.12: Ejemplo diálogos informativos página web

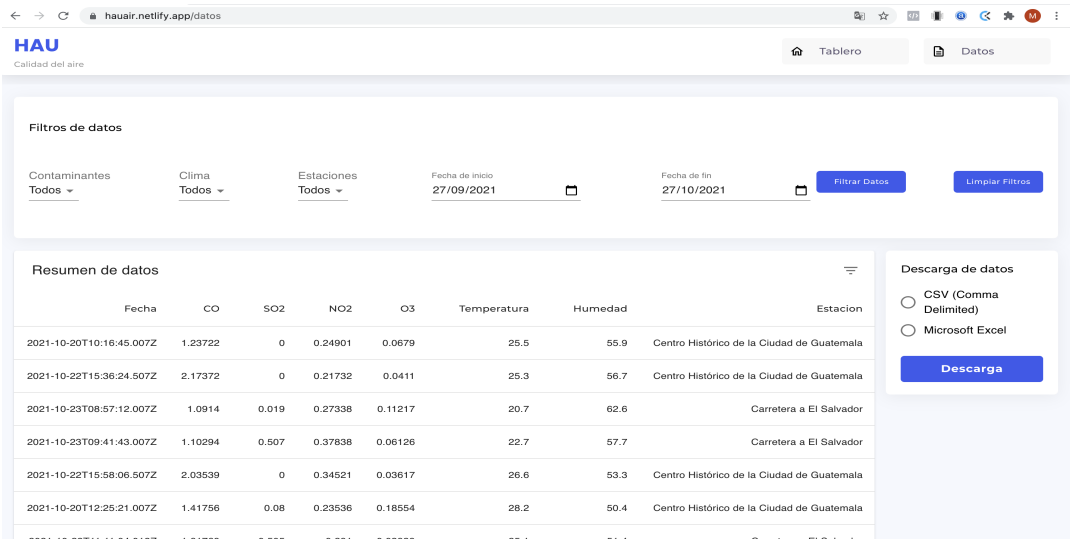


Figura 10.13: Pantalla Datos página web

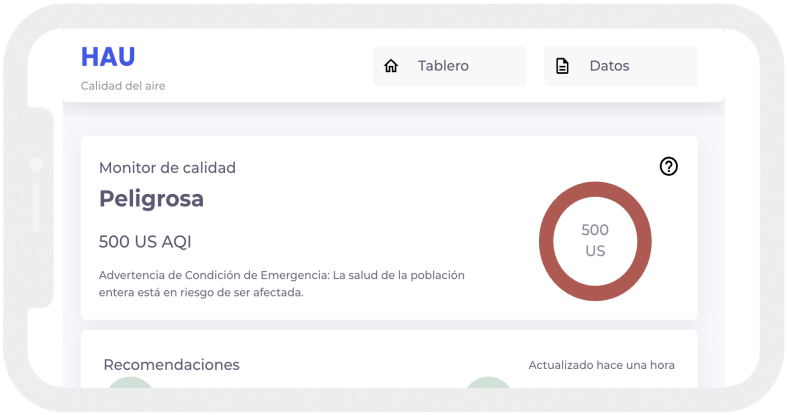


Figura 10.14: Pantalla Tablero en dispositivo móvil página web

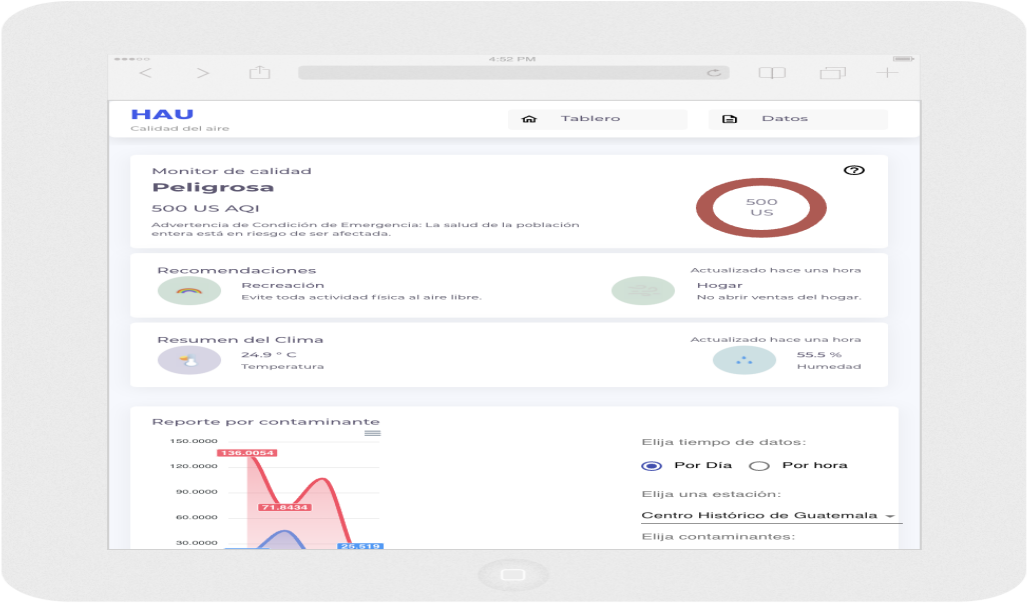


Figura 10.15: Pantalla Tablero en Tabletas página web

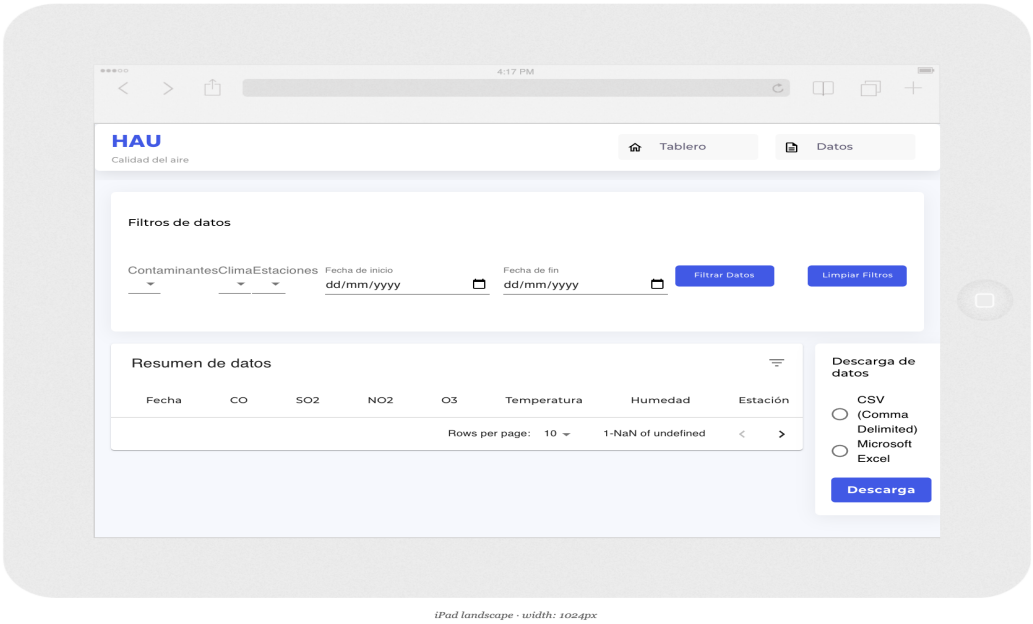


Figura 10.16: Pantalla Tablero en Tabletas página web

Los resultados obtenidos en la herramienta de *Mobile-Friendly Test* fueron exitosos, pasando la prueba y obteniendo que la página en efecto lo era.

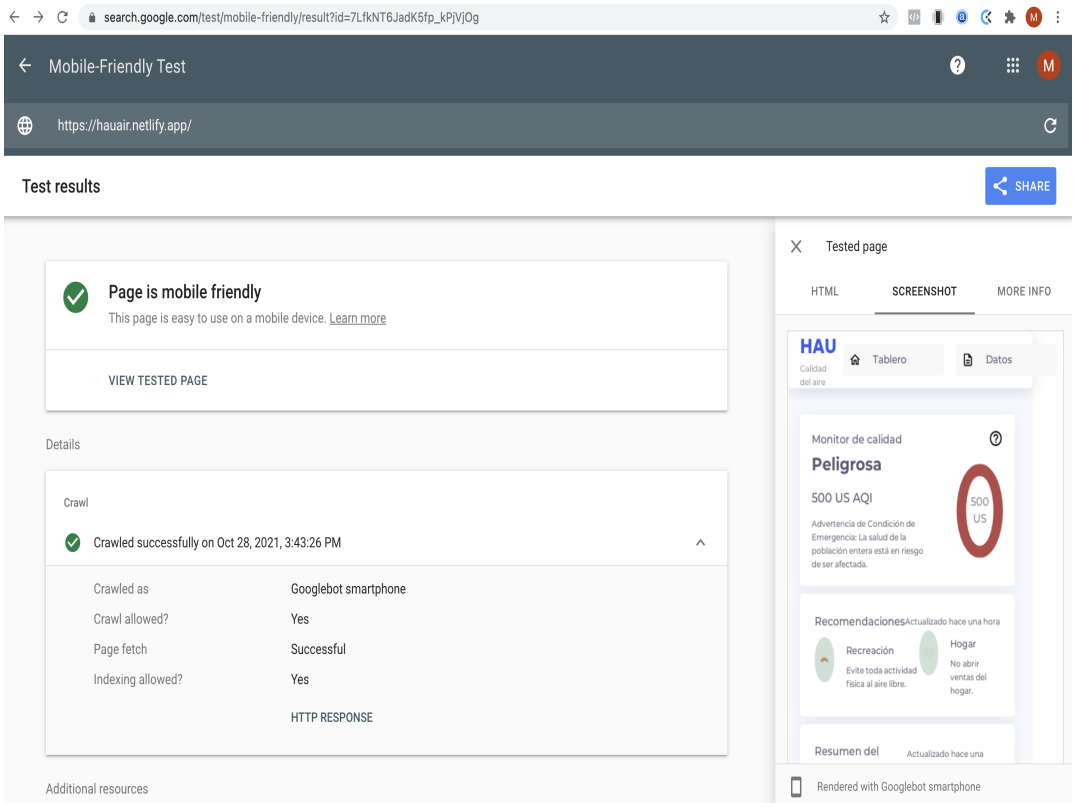


Figura 10.17: Resultados test de compatibilidad de diseño movil

10.5. Pruebas con usuario

El primer prototipo desarrollado se validó junto con Ana Lucía Hernández quien desarrolló el módulo de *Cloud Computing*.

La retroalimentación obtenida por su parte fue:

1. Los datos de recomendación sobre el rango del AQI, se deben de recopilar para ser agregados desde antes a la base de datos.
2. Las gráficas sobre los contaminantes y cambios en sus concentraciones se pueden realizar haciendo consultas a múltiples tablas al mismo tiempo.
3. La descarga de datos debe de realizarse desde el frontend dado que en la plataforma en la nube consume recursos y aumenta los costos de esta.
4. Colocar cuando fue actualizado un dato o los datos de las gráficas, dado que los valores del AQI son calculados en distintos lapsos de tiempo.

El segundo prototipo fue realizado en la plataforma Figma, este contaba con dos pantallas principales una llamada Tablero y la otra llamada Datos. La pantalla llamada Tablero contenía todos los gráficos, cuadros y despliegue de información relacionados con los casos de uso identificados. Por otra parte, la pantalla *Datos* era por la cual el usuario podría consultar y descargar la información recopilada por los sensores, siendo eso a través de los siguientes filtros:

- Contaminantes
- Clima (temperatura y humedad)
- Estación
- Rango de fechas

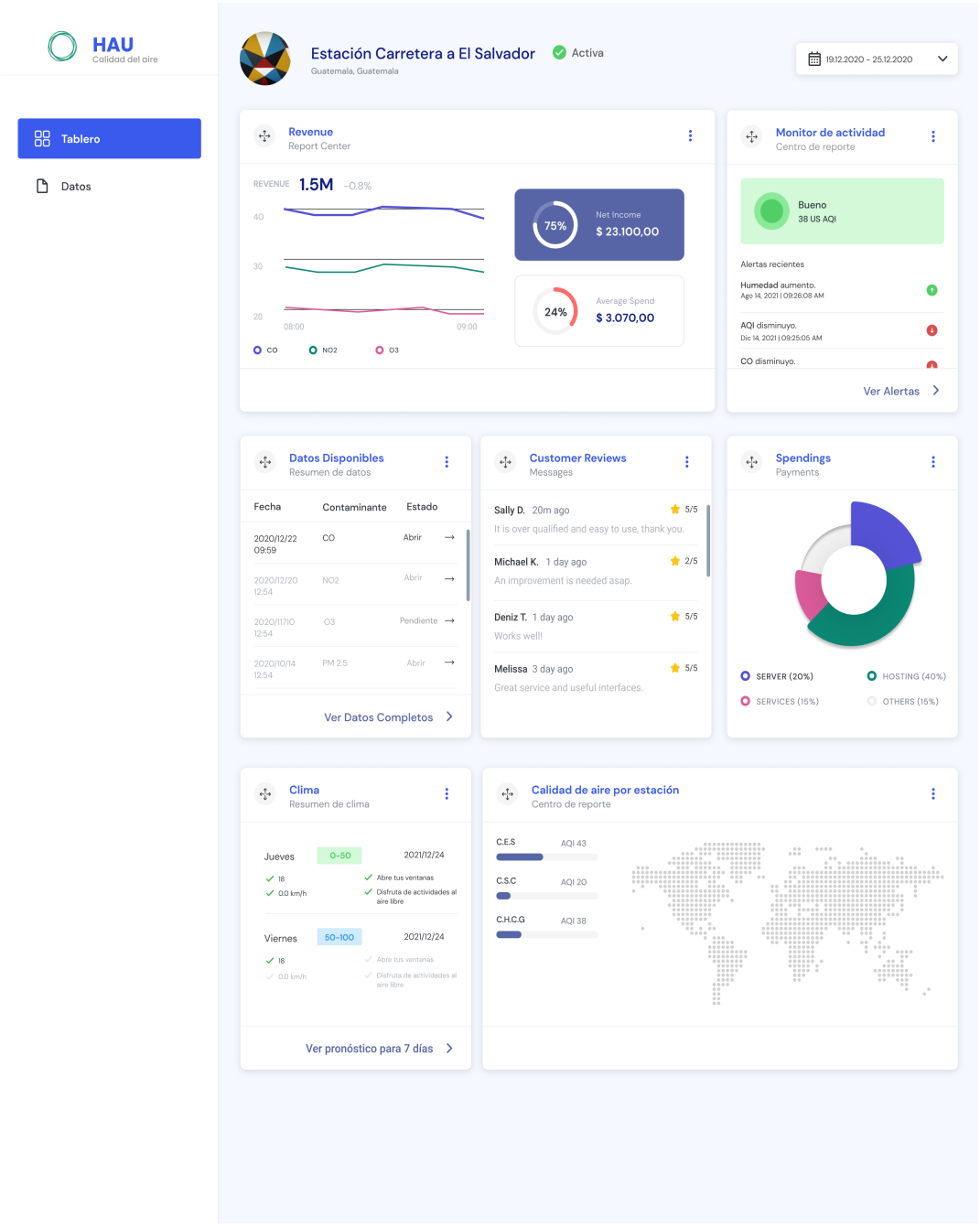


Figura 10.18: Pantalla Tablero primer prototipo

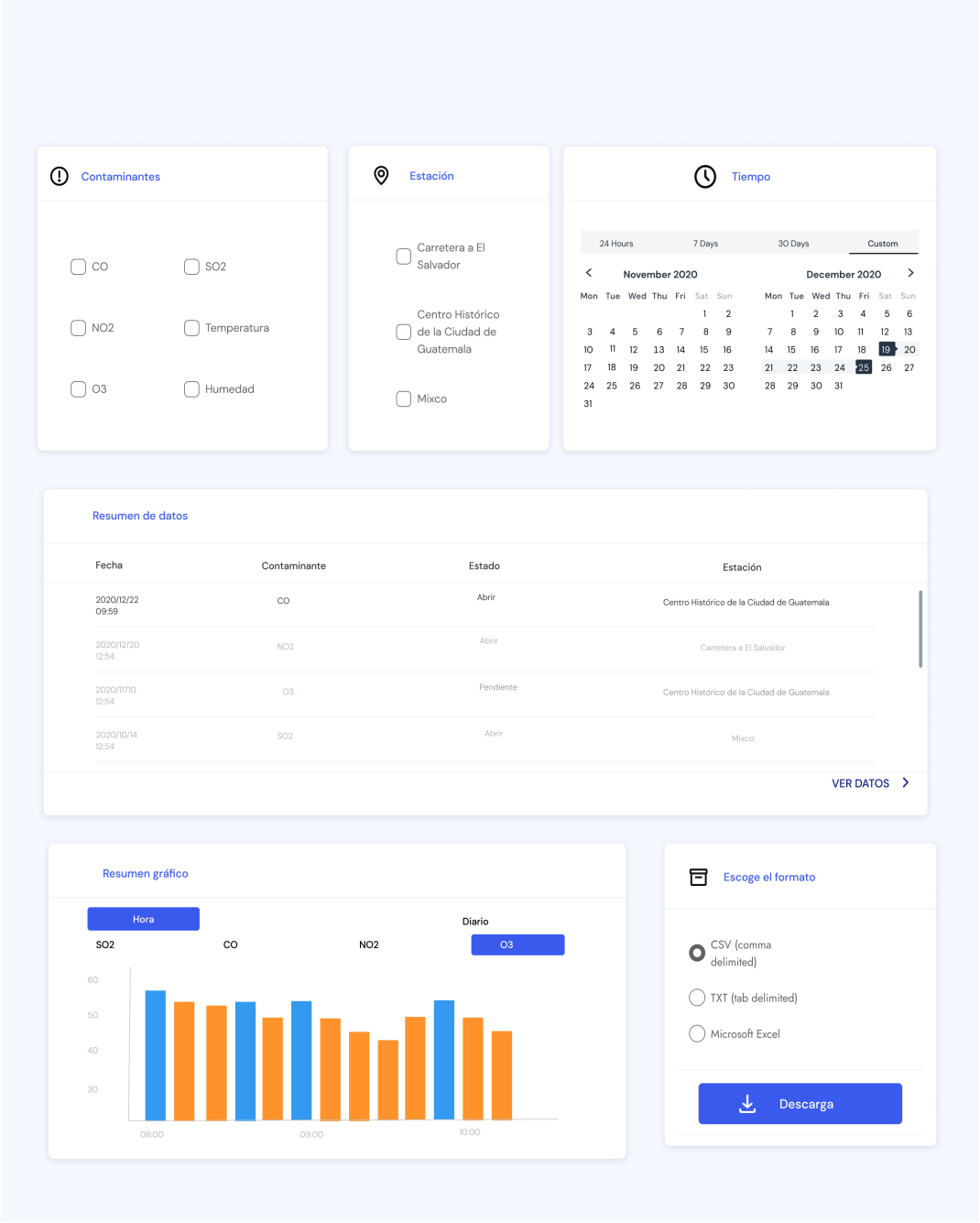


Figura 10.19: Pantalla Datos primer prototipo

Se realizaron las pruebas con 10 usuarios de los perfiles definidos, las recomendaciones a implementar derivadas de las pruebas con usuarios de este prototipo son las siguientes:

1. Enfocar el tablero a personas no conocedoras del tema incluyendo información general de lo que esta presentado en pantalla.
2. El monitor de actividad debe de llamar más la atención.
3. Los datos presentados en pantalla son confusos.

4. Aprovechar de mejor forma el espacio en pantalla para visualizar mejor las gráficas.

En esta fase se observó que el perfil de investigadores era más exigente respecto a la funcionalidad del prototipo, aunque a estos se les aclaró que aún era un prototipo no funcional y que contenía datos que no eran reales.

Todas de personas sabían sobre lo que trataba el tablero presentado y un 90 % se sentía cómodo utilizando plataformas de visualización de datos en línea. Por otra parte un 50 % de ellos respondió que la utilidad de los elementos del tablero no era la mejor, seleccionando un nivel 3. 40 % de los entrevistados respondieron que no utilizarían el tablero en su día a día, un 40 % tal vez y un 20 % sí. 10 % no lograron navegar hacia la página de datos y descargarlos y un 10 % indicó que no se siente tan cómodo utilizando una plataforma de visualización de datos en línea. Por último, un 70 % de ellos respondieron que se sintieron cómodos utilizando el tablero.

Para el segundo prototipo se realizaron cambios en la posición de los elementos, gráficas a mostrar en pantalla y se incluyeron pantallas informativas para dar más conocimiento con respecto a lo que se mostraba en pantalla.

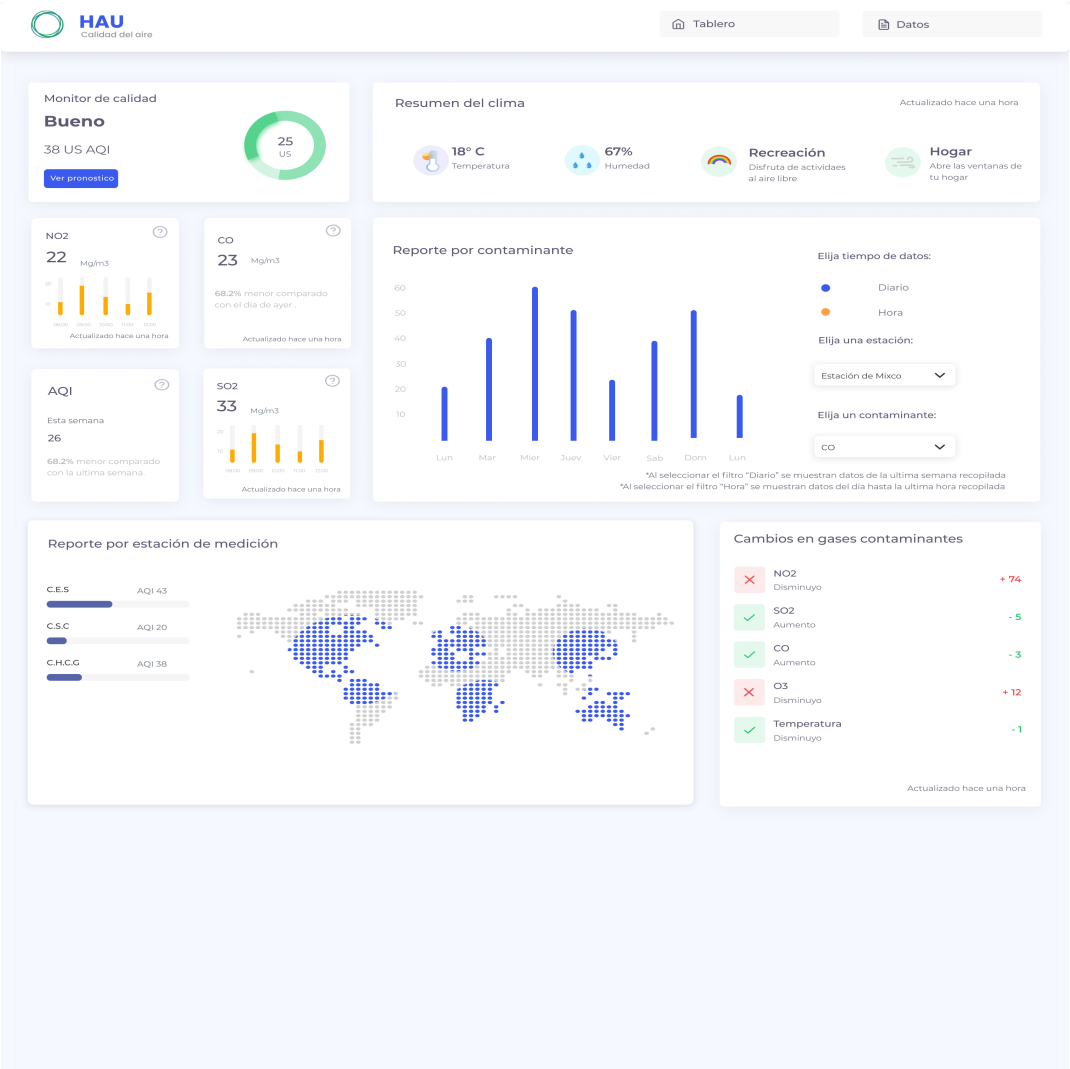


Figura 10.20: Pantalla Tablero segundo prototipo

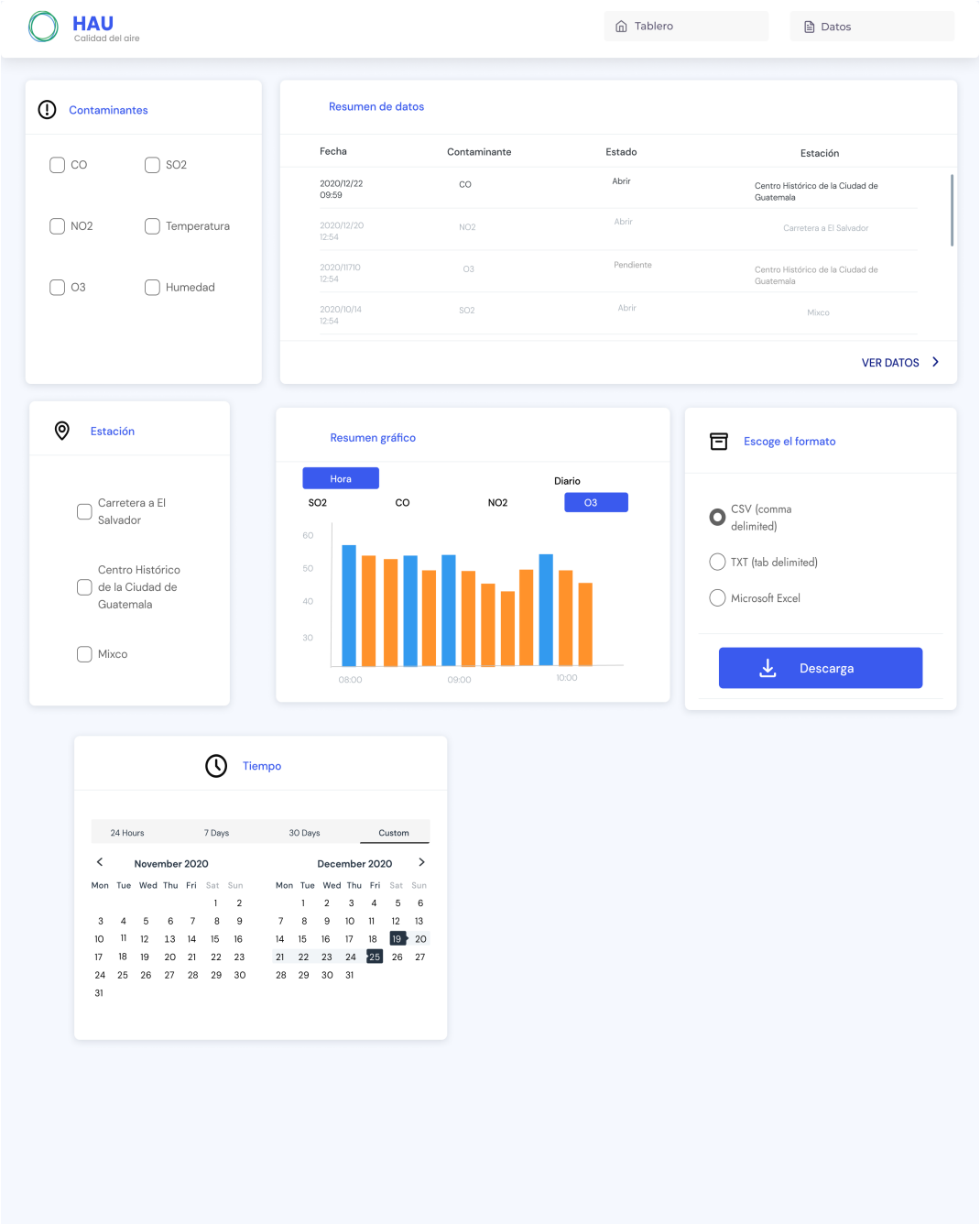


Figura 10.21: Pantalla Datos segundo prototipo

Se observó una mejor aceptación de esta versión del prototipo al observar que los comentarios realizados por los encuestados eran directamente relacionados con el tema y funcionalidad de los elementos vistos. Entre las recomendaciones a implementar se tomaron:

1. Eliminar elemento de cambios de contaminante pues está confuso.
2. Agregar una opción para seleccionar todos los filtros y/o limpiar filtros.
3. Acomodar de mejor forma los filtros para que estos se encuentren visibles en pantalla.

4. Incluir dimensiones de concentraciones medidas.

Un 60 % se sintió cómodo utilizando el tablero. Por otra parte un 40 % de ellos respondió que la utilidad de los elementos del tablero no era la mejor. 40 % de los entrevistados respondieron que no utilizarían el tablero en su día a día, un 40 % tal vez y un 20 % sí. Un 30 % cree que los elementos del tablero no son suficientes para un análisis diario de la calidad del aire. Un 100 % de los usuarios logro navegar a la página de datos. Solo un 30 % de los usuarios respondieron que se sentían cómodos utilizando la plataforma.

La última prueba se realizó a través del link https://hauair.netlify.app/, solicitándole al usuario que contestara la encuesta. Entre los comentarios recibidos se encuentran:

1. Hacer más grande el mapa para que se aprecien mejor las estaciones.
2. Los diálogos de ayuda complementan los datos calculados presentados en pantalla.

Escala de Usabilidad Prototipo Final			
Suma preguntas impares	22,5	17,5	
Suma preguntas pares	8,4	16,6	
		34,1	2,5
Resultado final		85,25	

Figura 10.22: Resultados prueba de usabilidad página web

Un 90 % de los encuestados se sintió cómodo utilizando el tablero. Un 70 % respondió que los elementos del tablero tienen una muy buena utilidad. 60 % respondieron que tal vez usarían el tablero en su día a día y un 40 % que sí. Un 80 % respondió que la información presentada sí era suficientes para un análisis diario de la calidad del aire y un 20 % respondió que no.

CAPÍTULO 11

Discusión de resultados

Con el proceso de Design Thinking se obtuvieron cinco casos de uso, los cuales cubrían las necesidades de los usuarios. Dentro de estas se encontraba una plataforma en la cual se pudieran acceder a datos públicos para ser utilizados y distribuidos de manera libre. Seguido de esto se encontró la necesidad de brindar información acerca de la calidad del aire para que tanto, usuarios del área ambiental o de biología y población en general pudieran informarse acerca de esta y tomar precauciones en caso la calidad del aire cerca de su área no se encuentre bien. Estos índices de calidad del aire eran brindados por el programa de calidad del aire del INSIVUMEH en un boletín diario, dichos boletines se dejaron de generar dado que las estaciones de medición se encuentran en mantenimiento desde el año 2016. A partir de dicho año, no hay una plataforma la cual realice estas mediciones y las brinde al público.

La primera prueba con el prototipo no obtuvo mucha aceptación por parte de los usuarios, ya que solamente un 70 % de ellos se sintió cómodo utilizando el tablero. Se observó que en esta fase de pruebas los usuarios comentaban mucho sobre los datos que se mostraban en este, los datos utilizados en el prototipo eran ficticios por lo que no se asemejaban a valores reales. También las personas esperaban que el prototipo tuviera ya todas las funcionalidades implementas a pesar de que se les aclaró que este aún no contaba con ellas. Esto se pudo haber debido a que el prototipo contaba con una resolución alta en donde los usuarios quienes lo probaban se enfocaban más en la interfaz, por ejemplo, datos, colores y tamaños y no en la funcionalidad e importancia de los elementos e información que este tuviera. A pesar de esto un 50 % de los usuarios afirmó que los elementos mostrados en pantalla, entiéndase gráficas y cuadros informativos, no eran de utilidad.

Un dato para resaltar es que 10 % de los entrevistados no logro navegar a la segunda página del tablero, la cual mostraba las opciones de descarga de datos. Sin embargo, hubo un 10 % de los entrevistados que respondió que no se siente cómodo utilizando plataformas de visualización de datos en línea. Esto es relevante para el proyecto, ya que uno de los objetivos es proveer datos de acceso público los cuales puedan ser distribuidos, si los usuarios no logran encontrar esta pestaña de datos dicho objetivo no se logrará cumplir. Por otro lado, no hay forma de asumir o relacionar que el mismo 10 % que no logro navegar a dicha pantalla es el mismo 10 % que se siente incómodo utilizando plataformas de visualización de datos en línea. La razón de por qué no se logra relacionar es dado que la plataforma de Google Form, la cual se utilizó para pasar dichas encuestas, se configuró para no tomar información personal de los usuarios.

Como se observa en la Figura 10.20, uno de los principales cambios fue ajustar la posición de los elementos para aprovechar de mejor forma el espacio en pantalla y que las gráficas disponibles en el tablero se pudieran manipular de mejor forma, tomando en cuenta la cantidad de variables por las cuales los datos de estas pudieran cambiar. Junto con ese cambio se aseguró que los datos mostrados en pantalla se asemejaran más a los valores reales de estos índices o concentraciones, para asegurarse que los usuarios se enfocaran en la utilidad y funcionalidad de los elementos. La cantidad de personas que se sintió cómoda utilizando este prototipo vario a comparación de la prueba anterior, se obtuvo que un 50 % se sentía en un nivel 4 de comodidad con este tablero, un 30 % en un nivel 5 y 20 % en un nivel 3. La razón por la cual se puede ver más variados los resultados que la primera prueba, puede ser porque el tablero ya mostraba datos más asemejados a la realidad por lo que los usuarios le prestaron más atención a esto. También ya no había gráficas o cuadros informativos que no presentaran una funcionalidad especifica como tal, a diferencia del prototipo anterior que había cuadros informativos que confundían al usuario. Esto pretende que los usuarios ya no se enfocaron en la interfaz y detalles de su apariencia si no más en la funcionalidad e importancia de los elementos. Esto se puede ver reflejado que un 60 % respondió que los elementos tienen un nivel 4 de utilidad.

A pesar de haber obtenido un 10 % de personas que se sienten incómodas utilizando plataformas de visualización en línea, el 100 % de los usuarios logro navegar hacia la pantalla de datos. Indicando que las recomendaciones adoptadas y cambios implementados ayudaron a construir de mejor manera la experiencia del usuario.

En cuanto al desarrollo del backend, utilizar Express para esto facilitó la conexión hacia la base de datos, ya que únicamente se debió de descargar una librería que permitiera conectarse con un Microsoft SQL Server a través de una cadena de caracteres la cual contiene las credenciales para acceder a ella. De igual forma permitió un manejo fácil y adecuado de los datos en cada controlador permitiendo devolver distintas estructuras de datos a través de las rutas, para que estos fueran utilizados sin necesidad de manipularse en las vistas de usuarios desarrolladas en la fase de interfaz de usuario. El modelo de programación utilizado para la interacción interna de la interfaz de usuario permitió que cada ruta pudiese recibir distintos parámetros y procesarlos de tal manera que llegaran al conector SQL listos para utilizarse dentro de una consulta a base de datos.

La construcción de la infraestructura para alojar y servir públicamente el backend se realizó de manera rápida debido a que Amazon Web Services provee una interfaz con bastante información y con pasos guiados de como crear distintas instancias. De igual forma uno puede utilizar la máquina virtual creada media vez se finalice su configuración. Amazon permite una escalabilidad sencilla de las instancias permitiendo aumentar en cantidad y tamaño los recursos de estas Al haber utilizado el servicio gratuito que este proveía el aumentar los recursos de esta se vería afectado en cuanto al costo de la instancia. El utilizar una instancia gratuita permitía el no incurrir en costos. Dado que que la plataforma backend no ocupa mucho espacio en disco y las herramientas para su funcionamiento tampoco, no se necesita de una gran cantidad de este y debido a que este no es quien realiza los cálculos de los índices, si no, que únicamente los consulta a la base de datos, utilizar una instancia gratuita era suficiente mientras que la plataforma no contara con tanto tráfico hacia ella. Sin embargo, pese a haber configurado protocolos y puertos de seguridad, las rutas de la API no cuentan con ninguna seguridad, permitiendo que cualquiera desde el puerto 80 (HTTP) o 443 (HTTPS) pueda acceder a esta.

El desarrollo de la interfaz de usuario utilizando un proyecto como archivo acelero el tiempo de desarrollo de este. Al tener la interfaz estática se observó que había por lo menos tres componentes, que indicaban el cambio en las concentraciones de los gases. Se decidió cambiar el componente de la Figura 10.20 llamado *Cambios en contaminantes*, por una gráfica de barras en donde se pudiera observar el cambio en cuanto a las concentraciones y valores del AQI del Ozono. Esto se decidió así, ya que este gas produce una de las concentraciones más peligrosas a nivel bajo, perjudicando la salud de grupos sensibles [4]. La retroalimentación obtenida se enfocó en el mapa de estaciones. Diciendo que este era costoso de navegar y acceder a la información que se desplegaba. Esto se debió a que el mapa no contaba con un acercamiento automático al departamento de Guate mala y ocupaba

un espacio reducido. Lo que pudo afectar a que la visualización e interacción del mapa no fuera la mejor, fue haber hecho uso de una librería de React JS en la cual se necesitaba de un archivo en formato topo.json con la topografía del país o ciudad que se deseaba mostrar. Se tomó la decisión de realizar esto de esta manera para que el diseño del mapa se viera similar al resto de elementos de la interfaz, sin embargo, perjudico a la experiencia del usuario.

El haber utilizado Netlify como servicio para integración continua y servir el Front End a través de él, permitió que el desarrollo fuera más rápido y que el servidor de AWS fuera dedicado únicamente para el API, permitiendo que esta respondiera de mejor manera.

Para las pruebas con usuarios se obtuvo que un 80 % se encuentra cómodo utilizando la plataforma y un 70 % opinó que los elementos del tablero tenían un nivel cinco de utilidad. Un 80 % opina que la información es suficiente para realizar un análisis diario de la calidad del aire. Por lo que el contenido de la plataforma es suficiente para visualizar y analizar datos de calidad del aire. Por otra parte nuevamente 100 % de los usuarios lograron navegar y utilizar la página de descarga datos haciendo que se logre distribuir y visualizar estos.

Para las pruebas de usabilidad se obtuvo un 85.25 % que según la Figura 7.28 el resultado de la prueba se encuentra dentro de un rango aceptable estando cerca de un puntaje excelente. Lo cual indicia que, desde la perspectiva del usuario, en términos generales, la plataforma es fácil de utilizar.

CAPÍTULO 12

Conclusiones

- Se logró desarrollar una estación que permite medir y compartir a una plataforma en la nube las concentraciones de contaminantes aéreos como CO, SO_2, NO_2 y O_3 en ambientes cerrados y abiertos. Debido a la falta de equipo especializado de medición de cada uno de los contaminantes, exceptuando el CO, actualmente resulta imposible calibrar los sensores a como fue calibrado el sensor MQ135, por lo que no se puede calcular la precisión de los sensores.
- Dado que los sensores muestran tendencias a reportar los aumentos de concentración de contaminantes y reportar resultados válidos en al menos el 79.27 % de las observaciones para SO_2, 100 % en NO_2, 99.57 % en CO y 99.31 % en O_3, la estación puede servir como una herramienta referente de las concentraciones que esta misma reporta. Es importante enfatizar que se requiere de mayor calibración a los sensores. Se considera completado el objetivo principal al haber desarrollado una estación de medición de contaminantes con costos totales de $150.34 USD al sumar la totalidad de costo de equipo de sensores referido por la Tabla 7.5.
- El contenedor desarrollado demostró que es capaz de soportar y proteger con resistencia frente a lluvias ligeras a la estación de medición, por lo que se considera que se desarrolló un contenedor capaz de soportar climas leves, sin embargo en tormentas más fuertes es probable que no pueda proteger del agua dado a las entradas de aire y requeriría el diseño de un contenedor más sofisticado.
- La estación fue capaz compartir datos a plataformas externas en la nube de manera continua bajo condiciones óptimas de red y suministro eléctrico, y en caso de fallas en el servicio de internet se logró además recuperar en la mayoría de casos se considera que el equipo es capaz de reportar de manera continua a plataformas externas.
- Se observó que la infraestructura implementada en la nube tuvo un desempeño satisfactorio al momento de tener que aprovisionar más recursos para acomodarse a una mayor carga de trabajo, deshabilitarlos cuando esta carga disminuía y aceptar datos provenientes de múltiples fuentes para centralizarlos.
- Se estableció la comunicación entre los dispositivos que recolectan las concentraciones de los contaminantes y Databricks a través de la interfaz de comunicación – API – ya establecida en IoT Hub. Se pudo observar que esta comunicación es correcta y rápida, con un tiempo promedio de transferencia de datos de 1.95 segundos.
- Se estableció que el pipeline implementado ejecuta el procesamiento de los datos recibidos por los sensores continuamente de manera exitosa. Las concentraciones de los contaminantes

son almacenadas en la base de datos y el Índice de Calidad de Aire es calculado cada hora correctamente, con un porcentaje de error general de 5.91 %.

- Se concluyó que la base de datos implementada en SQL Server de Azure almacena la información recolectada y procesada y atiende las solicitudes para poder leer esta información o escribir sobre las tablas de manera escalable y automatizada.
- Habiendo obtenido que el 70 % de los usuarios opinaba que los elementos del tablero tienen un nivel 5 de utilidad y un 80 % opina que la información es suficiente para un análisis diario de calidad del aire, se determina que se logró identificar el contenido de una plataforma web que permita visualizar, interactuar y descargar los datos almacenados.
- Con un 85.25 % de aceptación en la prueba de usabilidad, se determinó que se logró desarrollar una interfaz gráfica usable que permita visualizar, interactuar y descargar los datos almacenados.
- Se logró desarrollar una interfaz de programación de aplicaciones web, que consiste de diez rutas, las cuales permiten obtener los datos almacenados y utilizarlos en la página web.
- Se determinó que la implementación de una técnica de diseño web adaptable permite que la interfaz sea compatible con dispositivos móviles.
- La arquitectura implementada en Amazon Web Services, fue capaz de permitir la comunicación entre la interfaz de programación y la página web, haciendo que esta pueda hacer consultas constantes a ella cada vez que un usuario la visita.

CAPÍTULO 13

Recomendaciones

- Crear un sistema de alimentación eléctrica para la estación. Dado a que se cree importante desarrollar un sistema de alimentación eléctrico independiente para poder evitar el uso de una computadora que funcione como fuente de alimentación, se utilizó también una batería externa pero sigue siendo una solución temporal para hacer más independiente a esta estación.
- Utilizar una implementación del sensor MQ131 más económica. Debido a que existen versiones más baratas del mismo que podrían tener un rendimiento similar e incluso facilitar la implementación de código.
- Buscar sensores alternativos al Groove Multichannel o mejorar la implementación oficial del mismo. El sensor MICS-6814 implementado por la empresa Seeed da indicios que pueda existir una sobre estimación de la concentración de gas NO2, pero esta no pudo ser probada por falta de equipo. Según comunicación con Veselin Hadzhiyski, un investigador que afirma haber encontrado merjores resultados para valores de CO en el sensor MICS-6814 implementado por Seeed. Su implementación no fue utilizada en este trabajo por limitaciones de tiempo pero si existe interés en mejorar la calidad de las lecturas, se puede comunicar con él al correo: vcoder@abv.bg.
- Implementar acceso a internet distinto a Wi-fi, utilizar SIM para dispositivos IoT o una red LoRa, para comunicar a otro dispositivo los datos. Esto permitiría al proyecto a operar en áreas donde no existan redes Wi-Fi disponibles.
- Para la comparación de valores generados del cálculo del Índice de Calidad de Aire, se recomienda utilizar una fuente cuyas concentraciones insumo se encuentren en la misma escala que las utilizadas por la infraestructura, de acuerdo con la escala por contaminante establecida por la Agencia de Protección Ambiental de Estados Unidos.
- Se recomienda que se automatice a través de un proceso programado calendarizado la generación del SAS token que autentica la comunicación entre IoT Hub y los dispositivos, ya que, si el token llega a expirar y los sensores intentan comunicarse, la solicitud será denegada y se interrumpe la transmisión de datos.
- Se recomienda realizar pruebas de carga con más fuentes de datos para evaluar si la infraestructura diseñada es suficiente o si necesita ser mejorada.

- Se recomienda utilizar por lo menos un sensor por estación para obtener mediciones entre las distintas estaciones lo más similar posible y realizar comparaciones más acertadas.
- Se recomienda utilizar los datos presentados por la plataforma para futuros estudios ambientales sobre el impacto de la actividad humana sobre el entorno ambiental.
- Se recomienda el uso de prototipos en papel o de baja definición, para recibir retroalimentación enfocada a la utilidad y funcionalidad de los elementos y así evitar que los usuarios se enfoquen de primero en detalles de interfaz.
- Se recomienda ampliar la fase de Design Thinking para encontrar cálculos o índices de mayor importancia que puedan ser incluidos dentro de la plataforma.
- Se recomienda implementar seguridad a la interfaz de programaciones, dado a que actualmente se encuentra accesible para todo publico corre riesgo de una inyección SQL a través de los parámetros que reciben las rutas.
- Se recomienda utilizar una librería distintas para la generación del mapa de estaciones, la cual permite tener un mejor manejo de este y una interacción más agradable.
- Para la expansión del proyecto se recomienda:
  - Aumentar la capacidad de la máquina virtual EC2 para que soporte tráfico constante.
  - Aumentar la capacidad de ancho de banda que ofrece Netlify para que la interfaz de usuario este siempre accesible.
  - Si el proyecto se desea expandir a mas países se recomienda utilizar un mapa global y no únicamente de Guatemala.

Bibliografía

[1] *Grove - Multichannel Gas Sensor*. Seeed Studio, NA. https://wiki.seeedstudio.com/Grove-Multichannel_Gas_Sensor/#resources.

[2] Administration, U.S General Services: *System Usability Scale (SUS)*, 2021. https://www.usability.gov/how-to-and-tools/methods/system-usability-scale.html, visitado el 2021-08-12.

[3] Agency, U.S Environmental Protection: *A guide to Air Quality Index and Your Health*, 2014. https://www.airnow.gov/sites/default/files/2018-04/aqi_brochure_02_14_0.pdf, Recuperado el 22 de Octubre de 2021.

[4] Airnow: *Air Quality Guide For Ozone*, 2015. https://www.airnow.gov/sites/default/files/2021-03/air-quality-guide_ozone_2015.pdf, visitado el 2021-10-30.

[5] AirNow: *Air Quality Index Basics*, s.f. https://www.airnow.gov/aqi/aqi-basics/, Recuperado el 14 de Octubre de 2021.

[6] al., Skoog D. West D. et: *Fundamentos de química analítica*. Cengage Learning, 2015.

[7] Amazon: *Cloud Services - Amazon Web Services*, s.f. https://aws.amazon.com/, Recuperado el 24 de Octubre de 2021.

[8] Anand, Nayyar, Puri Vikram y Le Dac-Nhuong: *A Comprehensive Review of Semiconductor-Type Gas Sensors for Environmental Monitoring*. Review of Computer Engineering Research, 3:55–64, 2016.

[9] Blancarte, Oscar: *Escalabilidad Horizontal y Vertical*, 2017. https://www.oscarblancarteblog.com/2017/03/07/escalabilidad-horizontal-y-vertical/, Recuperado el 24 de Octubre de 2021.

[10] CDC: *¿Qué es el cáncer de piel?* 2021. https://www.cdc.gov/spanish/cancer/skin/basic_info/what-is-skin-cancer.htm.

[11] Cisco: *What is Wi-fi?* 2021. https://www.cisco.com/c/en/us/products/wireless/what-is-wifi.html.

[12] code, Learning to: *What is a framework?*, 2021. https://www.codecademy.com/resources/blog/what-is-a-framework/, visitado el 2021-06-22.

[13] Components, Quartz: *MQ-135 Air Quality Gas Sensor Module*. 2021. https://quartzcomponents.com/products/mq-135-air-quality-gas-sensor-module.

[14] Contributors, MDN: *Introducción a Express/Node*, 2021. https://developer.mozilla.org/es/docs/Learn/Server-side/Express_Nodejs/Introduction, visitado el 2021-06-22.

[15] Contributors, MDN: *¿Qué es el DOM?*, 2021. https://developer.mozilla.org/es/docs/Web/API/Document_Object_Model/Introduction, visitado el 2021-06-22.

[16] Dylan, James: *Ozono*, 2010. https://www.miteco.gob.es/es/calidad-y-evaluacion-ambiental/temas/atmosfera-y-calidad-del-aire/calidad-del-aire/salud/ozono.aspx, visitado el 2021-08-04.

[17] Education., IBM Cloud: *PaaS (Platform-as-a-Service)*, 2021. https://www.ibm.com/cloud/learn/paas#toc-paas-iaas-\/-xsjkA5Od, Recuperado el 19 de septiembre de 2021.

[18] Electronics, Aosong: *Digital-output relative humidity & temperature sensor/module DHT22 (DHT22 also named as AM2302)*. NA. https://www.sparkfun.com/datasheets/Sensors/Temperature/DHT22.pdf.

[19] EPA: *Technical Assistance Document for the Reporting of Daily Air Quality – the Air Quality Index (AQI)*. 2018. https://www.airnow.gov/sites/default/files/2020-05/aqi-technical-assistance-document-sept2018.pdf.

[20] EPA: *Air Pollution: Current and Future Challenges*. EPA, 2021. https://www.epa.gov/clean-air-act-overview/air-pollution-current-and-future-challenges.

[21] EPA: *Basic Information about Carbon Monoxide (CO) Outdoor Air Pollution*. 2021. https://www.epa.gov/co-pollution/basic-information-about-carbon-monoxide-co-outdoor-air-pollution.

[22] EPA: *Basic Information about NO2*. 2021. https://www.epa.gov/no2-pollution/basic-information-about-no2.

[23] EPA: *Health Effects of Ozone Pollution*. 2021. https://www.epa.gov/ground-level-ozone-pollution/health-effects-ozone-pollution.

[24] EPA: *Sulfur Dioxide Basics*. 2021. https://www.epa.gov/so2-pollution/sulfur-dioxide-basics.

[25] Ernest Abadal, Lluís Codina: *Bases de datos documentales: características, funciones y métodos*. Síntesis, Madrid, 2005.

[26] Espressif: *ESP32 Series Datasheet*. 2021. https://www.espressif.com/sites/default/files/documentation/esp32_datasheet_en.pdf.

[27] EXTOXNET, Oregon State University: *HOW MUCH IS A PART PER MILLION?* 1993. http://extoxnet.orst.edu/tibs/partperm.htm.

[28] FOUNDATION, INTERACTION DESIGN: *The Basics of User Experience Design*. INTERACTION DESIGN FOUNDATION Est 2002. INTERACTION DESIGN FOUNDATION, 2002.

[29] G, MANUEL OYARZÚN: *Contaminación aérea y sus efectos en la salud*. Universidad de La Serena. Chile, 26(1):1–3, 2010.

[30] Gabriel Costa Silva, Louis M. Rose, Radu Calinescu: *IEEE International Conference on Cloud Computing Technology and Science (CloudCom)*. En *Towards a Model-Driven Solution to the Vendor Lock-In Problem in Cloud Computing*, páginas 711–716, Bristol, Reino Unido, 2013.

[31] Gironi, David: *MQ gas sensor correlation function against temperature and humidity*. 2017. https://davidegironi.blogspot.com/2017/07/mq-gas-sensor-correlation-function.html#.YT4b_PdMHb0.

[32] Google: *About Google Search Console*, 2021. https://support.google.com/webmasters/answer/9128668?hl=es, visitado el 2021-10-30.

[33] Google: *Acelera tu transformación con Google Cloud*, s.f. https://cloud.google.com/?utm_source=google&utm_medium=cpc&utm_campaign=latam-LATAM-all-es-dr-BKWS-all-all-trial-p-dr-1009897-LUAC0014410&utm_content=text-ad-none-any-DEV_c-CRE_512379899447-ADGP_Hybrid%20%7C%20BKWS%20-%20PHR%20%7C%20Txt%20~%20GCP_General-, Recuperado el 23 de Octubre de 2021.

[34] Google, Inc.: *¿Qué es Pub/Sub?*, 2021. https://cloud.google.com/pubsub/docs/overview#comparing_to_other_messaging_technologies, Recuperado el 20 de Octubre de 2021.

[35] Google, Inc.: *BigQuery*, s.f. https://cloud.google.com/bigquery/docs, Recuperado el 20 de Octubre de 2021.

[36] Google, Inc.: *Dataflow*, s.f. https://cloud.google.com/dataflow#all-features, Recuperado el 20 de Octubre de 2021.

[37] Google, Inc.: *Precios de Google Cloud*, s.f. https://cloud.google.com/pricing?skip_cache=true, Recuperado el 24 de Octubre de 2021.

[38] Guevara, R. E.: *Servicios de cómputo en la nube (cloud computing)*. Universidad de Quintana Roo., Quintana Roo, México, 2018.

[39] Hianwei Electronics CO., LTD: *TECHNICAL DATA MQ-5 GAS SENSOR*. NA. https://datasheetspdf.com/pdf-file/904638/HANWEIELETRONICS/MQ-5/1.

[40] Huang, Dapeng Zhu Todd Sifleet Travis Nunnally Yachun: *Analog to Digital Converters*. Georgia Tech, 2017. https://ume.gatech.edu/mechatronics_course/ADC_F08.pdf.

[41] Inc, Facebook: *Componentes y propiedades*, 2021. https://es.reactjs.org/docs/components-and-props.html, visitado el 2021-06-22.

[42] Inc, Facebook: *DOM virtual y detalles de implementación*, 2021. https://es.reactjs.org/docs/faq-internals.html, visitado el 2021-06-22.

[43] INSIVUMEH: *Indice de calidad del aire*, 2010. http://hidromet.insivumeh.gob.gt/calidadaire/indiceconcepto.htm, visitado el 2021-08-04.

[44] INSIVUMEH: *PROGRAMA DE CALIDAD DEL AIRE DE INSIVUMEH PARA LA REPÚBLICA DE GUATEMALA*. INSIVUMEH, 2021. http://hidromet.insivumeh.gob.gt/calidadaire/.

[45] Krill, Paul: *React: Making faster, smoother UIs for data-driven Web apps*, 2014. https://www.infoworld.com/article/2608181/react--making-faster--smoother-uis-for-data-driven-web-apps.html, visitado el 2021-06-22.

[46] MARN, USAC: *PRIMER INFORME INDICATIVO DE MEDICIÓN DE LA CALIDAD DEL AIRE AMBIENTE EN LAS CABECERAS DEPARTAMENTALES DE LA REPÚBLICA DE GUATEMALA*. MARN, USAC, 2013. https://www.marn.gob.gt/Multimedios/7907.pdf.

[47] Mary Grace Legaspi, Eric Peña: *UART: A Hardware Communication Protocol Understanding Universal Asynchronous Receiver/Transmitter*. AnalogDialogue, 54, 2020. https://www.analog.com/en/analog-dialogue/articles/uart-a-hardware-communication-protocol.html.

[48] Mauricio Castillo-Vergara, Alejandro Alvarez-Marin, Ricardo Cabana Villca: *Design thinking: como guiar a estudiantes, emprendedores y empresarios en su aplicación*. Universidad de La Serena. Chile, 35(3):5–6, 2014.

[49] Microsoft: *Azure IoT Hub*. 2021. https://azure.microsoft.com/en-us/services/iot-hub/#overview.

[50] Microsoft: *Conceptos de IoT y Azure IoT Hub*, 2021. https://docs.microsoft.com/es-mx/azure/iot-hub/iot-concepts-and-iot-hub, Recuperado el 20 de Octubre de 2021.

[51] Microsoft.: *Otorgar acceso limitado a recursos de Azure Storage con firmas de acceso compartido*, 2021. https://docs.microsoft.com/es-es/azure/storage/common/storage-sas-overview, Recuperado el 22 de Octubre de 2021.

[52] Microsoft: *What is cloud computing?* 2021. https://azure.microsoft.com/en-us/overview/what-is-cloud-computing/#benefits.

[53] Microsoft.: *¿Qué es Azure SQL Database?*, 2021. https://docs.microsoft.com/es-mx/azure/azure-sql/database/sql-database-paas-overview, Recuperado el 20 de Octubre de 2021.

[54] Microsoft: *Cree su cuenta gratuita de Azure hoy mismo*, s.f. https://azure.microsoft.com/es-es/free/, Recuperado el 23 de Octubre de 2021.

[55] Microsoft: *Servicios de informática en la nube Azure*, s.f. https://azure.microsoft.com/es-es/, Recuperado el 24 de Octubre de 2021.

[56] Microsoft: *What is a public cloud?*, s.f. https://azure.microsoft.com/en-us/overview/what-is-a-public-cloud/, Recuperado el 19 de Septiembre de 2021.

[57] Microsoft: *¿Qué es la virtualización?*, s.f. https://azure.microsoft.com/es-es/overview/what-is-virtualization/, Recuperado el 19 de septiembre de 2021.

[58] Microsoft: *¿Qué es una nube privada?*, s.f. https://azure.microsoft.com/es-es/overview/what-is-a-private-cloud/, Recuperado el 19 de septiembre de 2021.

[59] Org, Breathe: *The Air Pollution in Guatemala, Guatemala.* Breathe, 2016. https://breathelife2030.org/city_data/guatemala/.

[60] Org, MQTT: *MQTT: The Standard for IoT Messaging.* 2020. https://mqtt.org/.

[61] Peaslee, D.: *I cant set to Zero DGS S02.* GitHub, 2021. https://github.com/SPEC-Sensors/DSDK_Tool/issues/1.

[62] Raj Bala, Bob Gill, et. al: *Magic Quadrant for Cloud Infrastructure and Platform Services.* Gartner, 2021.

[63] Red Hat, Inc.: *¿Qué es una nube híbrida?*, s.f. https://www.redhat.com/es/topics/cloud-computing/what-is-hybrid-cloud, Recuperado el 19 de septiembre de 2021.

[64] RedHat: *¿Qué es el Internet de las cosas (IoT)?* 2021. https://www.redhat.com/es/topics/internet-of-things/what-is-iot.

[65] Robotics, George: *MicroPython.* 2018. https://micropython.org/.

[66] salud, Organización mundial de la: *OMS estima que 7 millones de muertes ocurren cada año debido a la contaminación atmosférica*, 2014. https://www3.paho.org/hq/index.php?option=com_content&view=article&id=9406:2014-7-million-deaths-annually-linked-air-pollution&Itemid=135&lang=es, visitado el 2021-04-06.

[67] SENSORS, SPEC: *Ultra-Low Power Analog Sensor Module for Sulfur Dioxide.* 2016. https://www.spec-sensors.com/wp-content/uploads/2016/10/ULPSM-SO2-968-006.pdf.

[68] SENSORS, SPEC: *Digital Gas Sensor – Sulfur Dioxide*. SPEC SENSORS, 2017. https://www.spec-sensors.com/wp-content/uploads/2017/01/DGS-SO2-968-038.pdf.

[69] Sensortech, SGX: *Datasheet MiCS-6814 1143 rev 8*. 2015. https://www.sgxsensortech.com/content/uploads/2015/02/1143_Datasheet-MiCS-6814-rev-8.pdf.

[70] Shivaji Mirashe, N.V. Kalyankar: *Cloud Computing*. Journal of Computing, 2(3), 2010.

[71] Smyk, Andrew: *The System Usability Scale How It's Used in UXL*, 2019. https://xd.adobe.com/ideas/process/user-testing/sus-system-usability-scale-ux/, visitado el 2021-04-06.

[72] Spiess, Andreas: *Time to Say Goodbye to Arduino and Go On to Micropython/ Adafruit Circuitpython?*, 2018. https://www.youtube.com/watch?v=m1miwCJtxeM, visitado el 2021-10-31.

[73] Sánchez, Jorge: *Principios sobre las Bases de Datos Relacionales*. Creative Commons, California, Estados Unidos, 2004.

[74] Thornton, Scott: *What is I2C a.k.a. "I-squared-C"?* Microcontroller Tips, 2017. https://www.microcontrollertips.com/i2c-k-squared-c/.

[75] Triggs, Robert: *What is an SoC? Everything you need to know about smartphone chipsets*. Android Authority, 2021. https://www.androidauthority.com/what-is-an-soc-smartphone-chipsets-explained-1051600/.

[76] UNECA: *Web Technology*, 2021. https://archive.uneca.org/sites/default/files/uploaded-documents/SROs/SA/GIS-SP2018/introduction_to_web_technology.pdf, visitado el 2021-06-22.

ANEXO A

Recursos utilizados

Los recursos utilizados para este proyecto fueron principalmente scripts proveídos por los fabricantes o desarrolladores independientes.

A.1. Scripts

A.1.1. Automatización de extracción de datos para gráfica de sensores MQ

El script de R, publicado por el Ing. Davide Gironi permitió obtener de manera automatizada los valores de R0, a y b para la ecuación 4.5 a partir de la gráfica de 7.1. El script de acceso público puede ser obtenido en el siguiente enlace `https://gist.github.com/davidegironi/b7be6b7cace6b475dd42c48c3e62fcf4`

A.1.2. Calibración de MICS-6814

El script programado en Arduino fue utilizado para calibrar por primera vez el sensor MICS-6814. Este script fue desarrollado por Seeed, la empresa fabricante de la placa madre del sensor. El script de acceso público puede ser obtenido en el siguiente enlace `https://github.com/Seeed-Studio/Mutichannel_Gas_Sensor/blob/master/examples/calibration/calibration.ino`

A.1.3. Comunicación con MICS-6814

Se utilizó una traducción de MicroPython de la librería proveída por el fabricante Seeed que originalmente fue desarrollada en Arduino. Esta es de acceso público y se puede encontrar en el siguiente enlace. `https://github.com/aparcar/groove-multichannel-gas-sensor-micropyton/blob/master/gas.py`

A.1.4. Script de conexión con Azure IoT Hub

Se utilizó el script que permite realizar la conexión a través del protocolo MQTT al IoT Hub, naturalmente este fue alterado para las necesidades del proyecto. Este es de acceso público y puede ser obtenido en el siguiente enlace https://github.com/Azure-Samples/IoTMQTTSample/blob/master/src/MicroPython/main.py

A.2. Programas y aplicaciones utilizados

A.2.1. WebPlotDigitalizer

Esta aplicación web fue utilizada con el fin de poder extraer puntos de gráficas utilizadas en el trabajo y eliminar potencial error humano, especialmente en gráficas donde la calidad de la misma es baja. La aplicación web puede encontrarse en el siguiente enlace https://apps.automeris.io/wpd/

A.2.2. SPEC DSDK TOOL

Este programa fue utilizado en conjunto con el sensor SPEC DGS SO2 con el fin de dejar en un tiempo de calentado el sensor y realizar una calibración set Zero. En el siguiente enlace de descarga se puede encontrar el programa https://www.spec-sensors.com/wp-content/uploads/2019/03/DSDK_Tool_v0.42.zip

A.2.3. MakerCase

Esta aplicación web fue utilizada con el fin de obtener un diseño para un contenedor que protegiera la estación de medición. Esta aplicación web puede encontrarse en el siguiente enlace https://es.makercase.com/#/

A.2.4. AQI Calculator

Esta aplicación web recomendada por EPA, es utilizada para validar los valores obtenidos por la estación. Esta aplicación web puede encontrarse en el siguiente enlace: https://www.airnow.gov/aqi/aqi-calculator-concentration/

A.2.5. Circuit Diagram

Aplicación web utilizada para crear diagrama técnico de conexión, puede encontrarse en el siguiente enlace: https://www.circuit-diagram.org/

A.3. Repositorio de proyecto

El repositorio de Github donde fue desarrollado el proyecto es público y se puede acceder a él bajo el siguiente enlace: https://github.com/molinajimenez/airQualityIOT/tree/master

A.4. Datos utilizados

El acceso a los datos utilizados en la discusión de resultados pueden ser descargados en el siguiente enlace: https://drive.google.com/drive/folders/138XsU6bsrc9dfzM_fLzr8eumurhOKhzN?usp=sharing

ANEXO B

Evidencia pruebas con usuarios

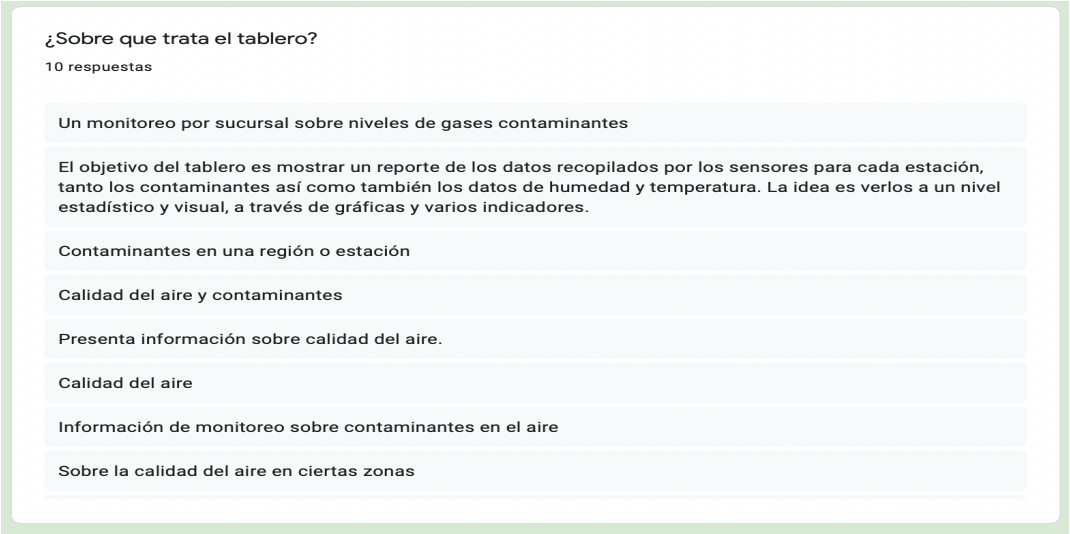


Figura B.1: Resultados primera pregunta primer prototipo

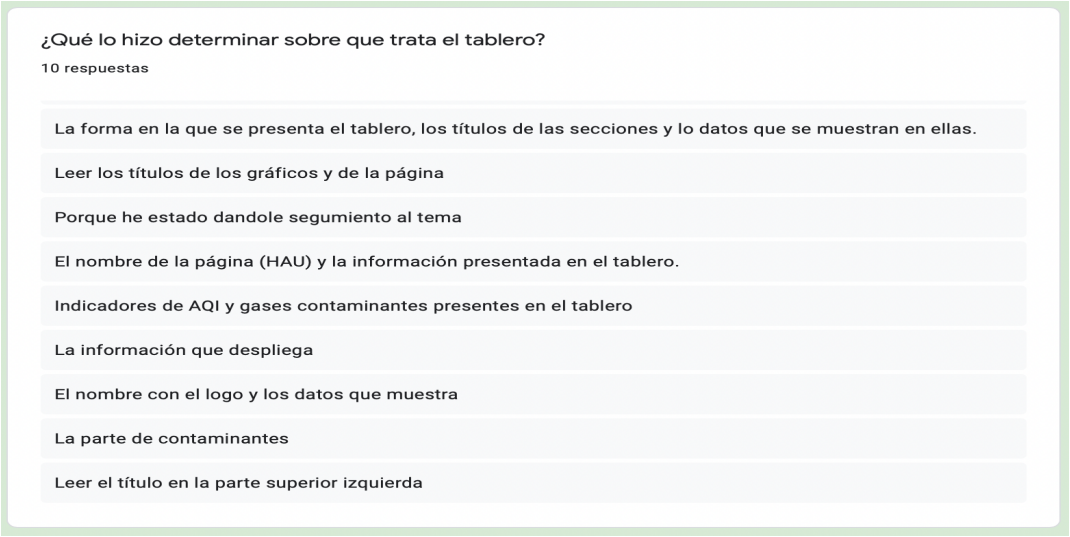


Figura B.2: Resultados segunda pregunta primer prototipo

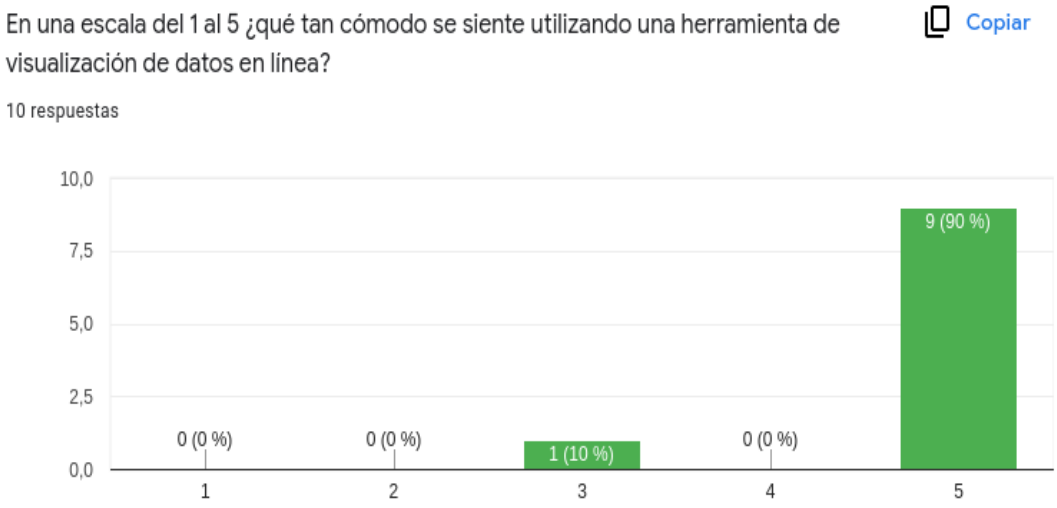


Figura B.3: Resultados tercera pregunta primer prototipo

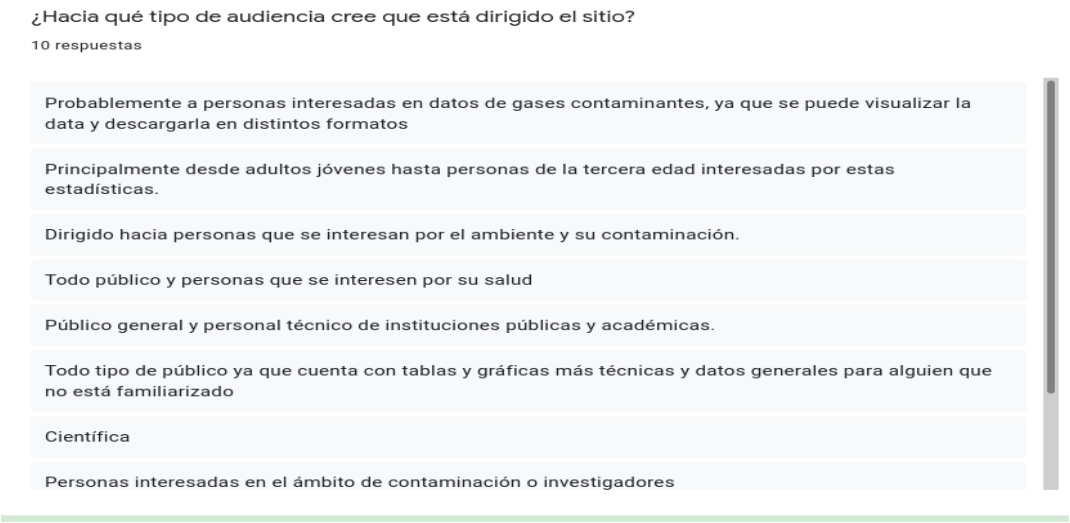


Figura B.4: Resultados cuarta pregunta primer prototipo

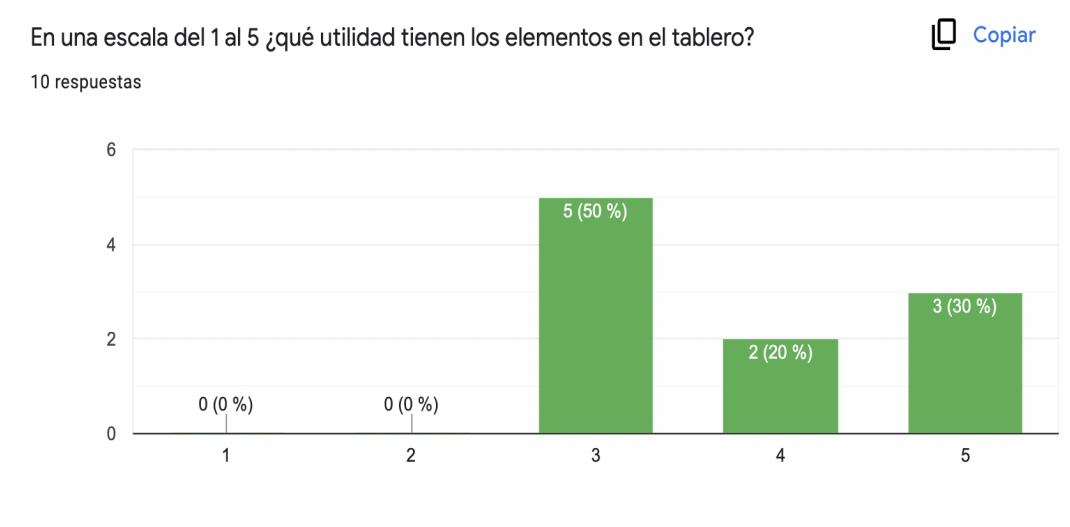


Figura B.5: Resultados quinta pregunta primer prototipo

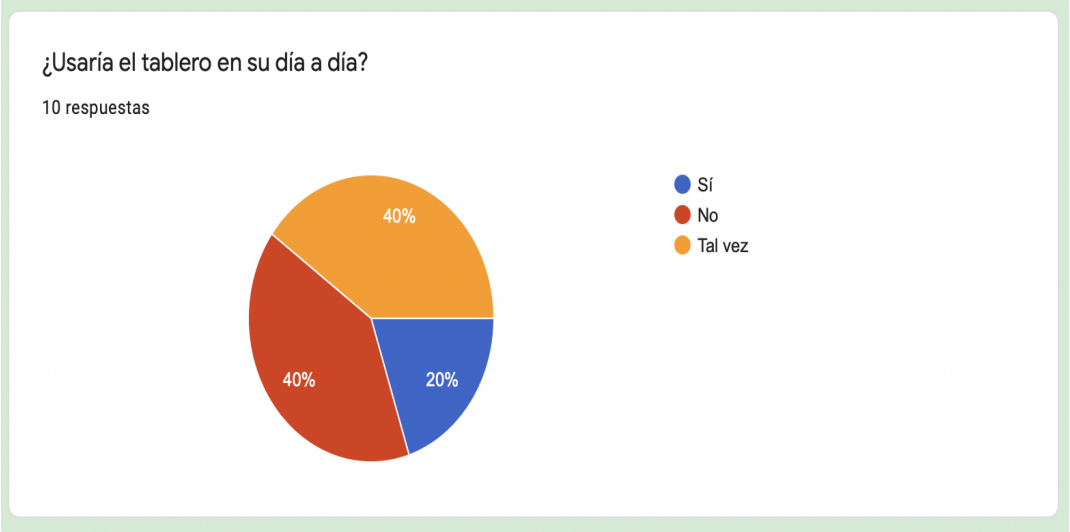


Figura B.6: Resultados sexta pregunta primer prototipo

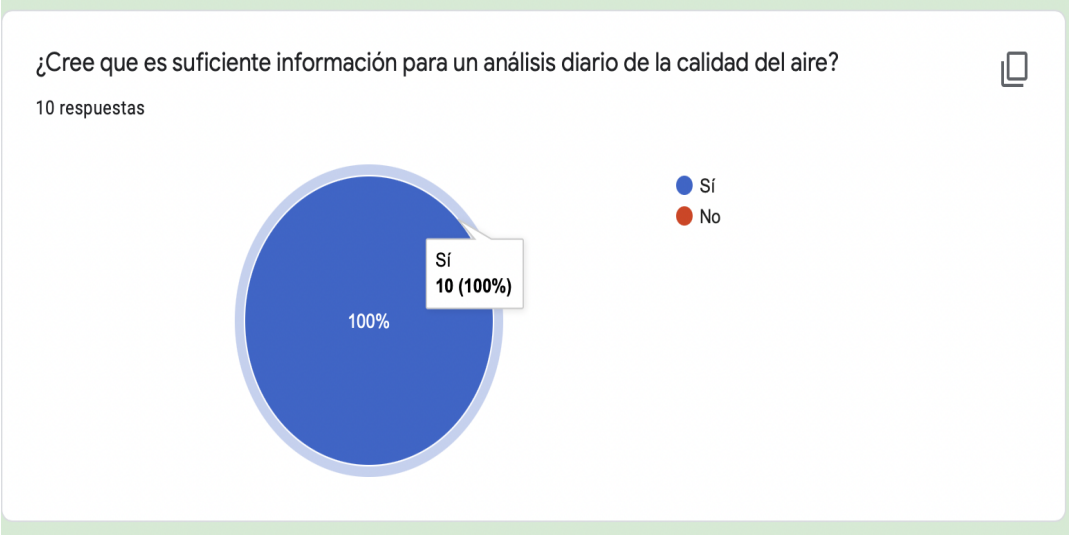


Figura B.7: Resultados séptima pregunta primer prototipo

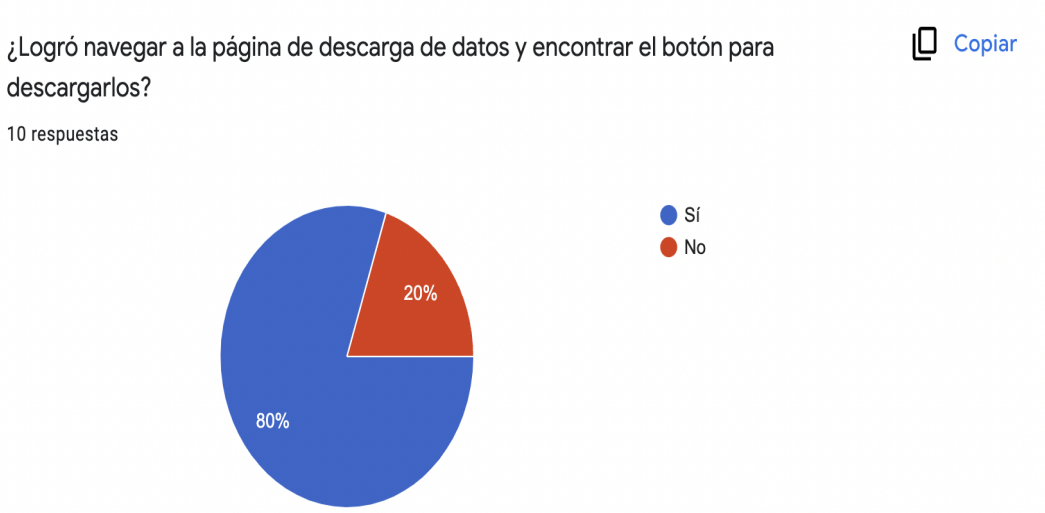


Figura B.8: Resultados octava pregunta primer prototipo

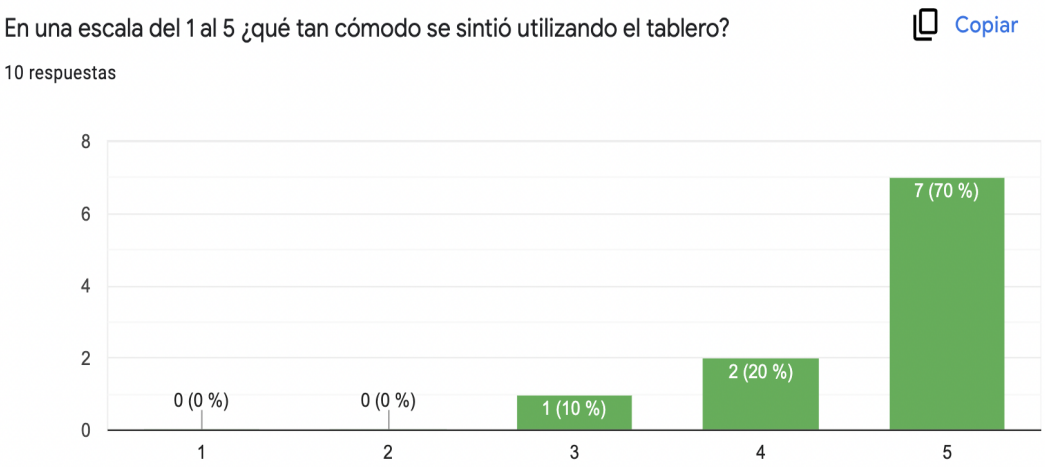


Figura B.9: Resultados novena pregunta primer prototipo

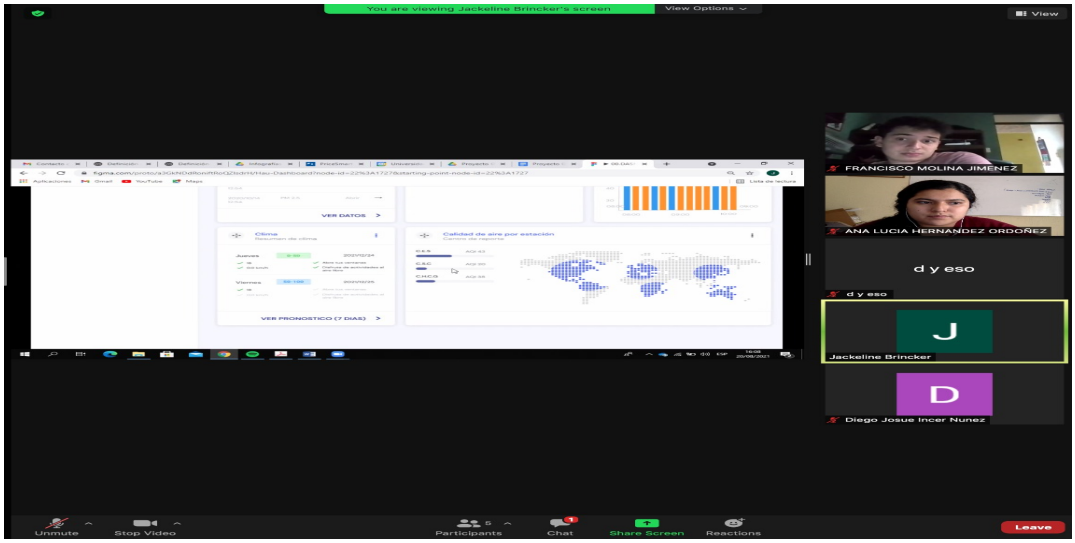


Figura B.10: Prueba virtual primer prototipo Jackelyn Brincker y Diego Incer

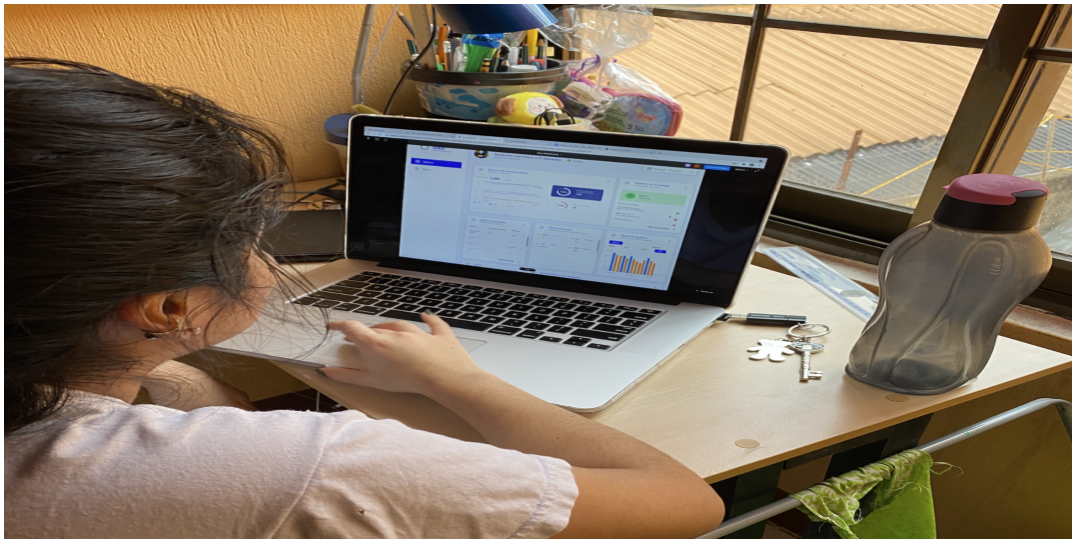


Figura B.11: Prueba virtual primer prototipo Jennifer Hernández

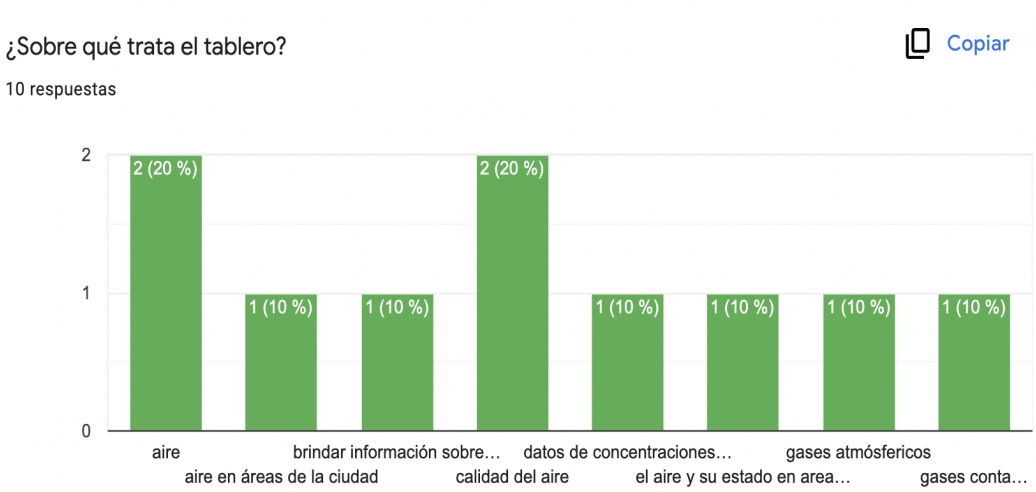


Figura B.12: Resultados primera pregunta segundo prototipo

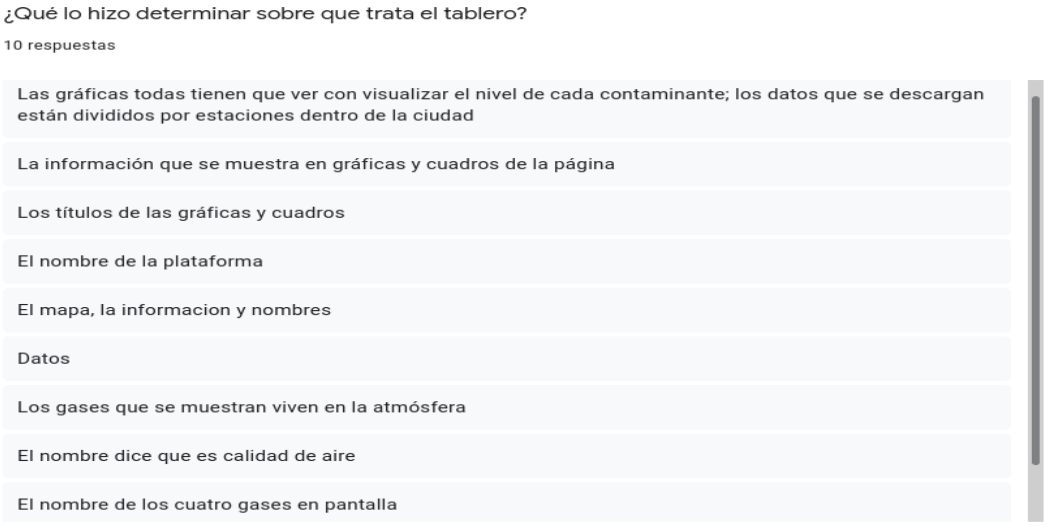


Figura B.13: Resultados segunda pregunta segundo prototipo

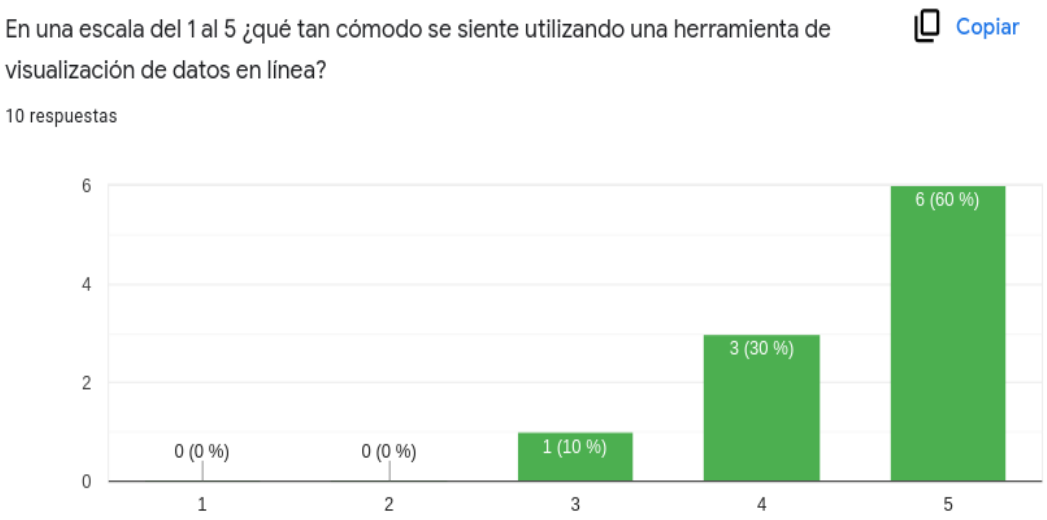


Figura B.14: Resultados tercera pregunta segundo prototipo

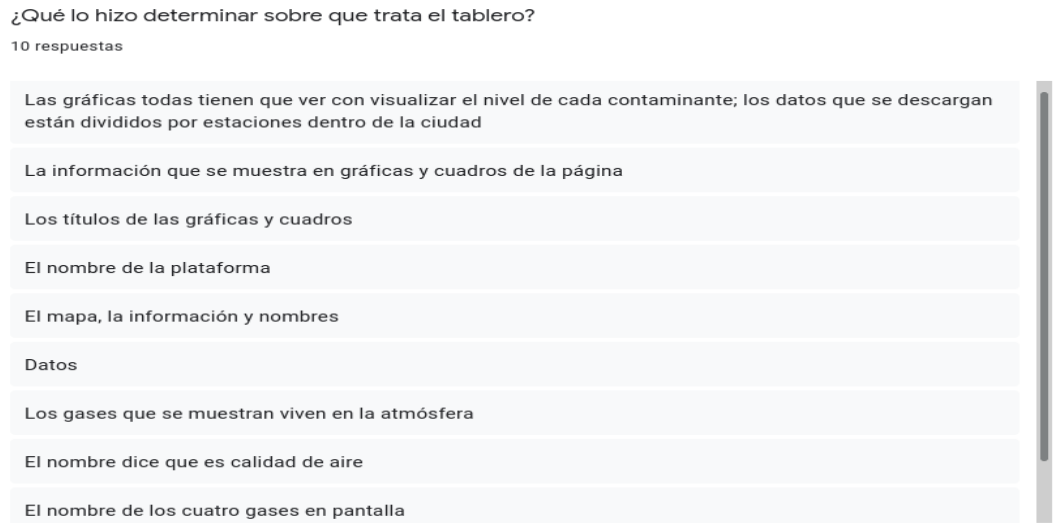


Figura B.15: Resultados cuarta pregunta segundo prototipo

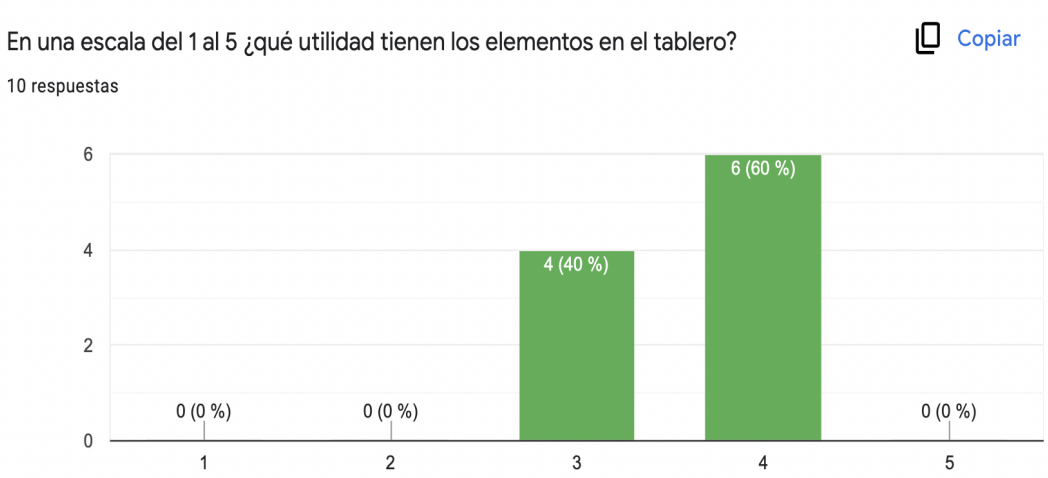


Figura B.16: Resultados quinta pregunta segundo prototipo

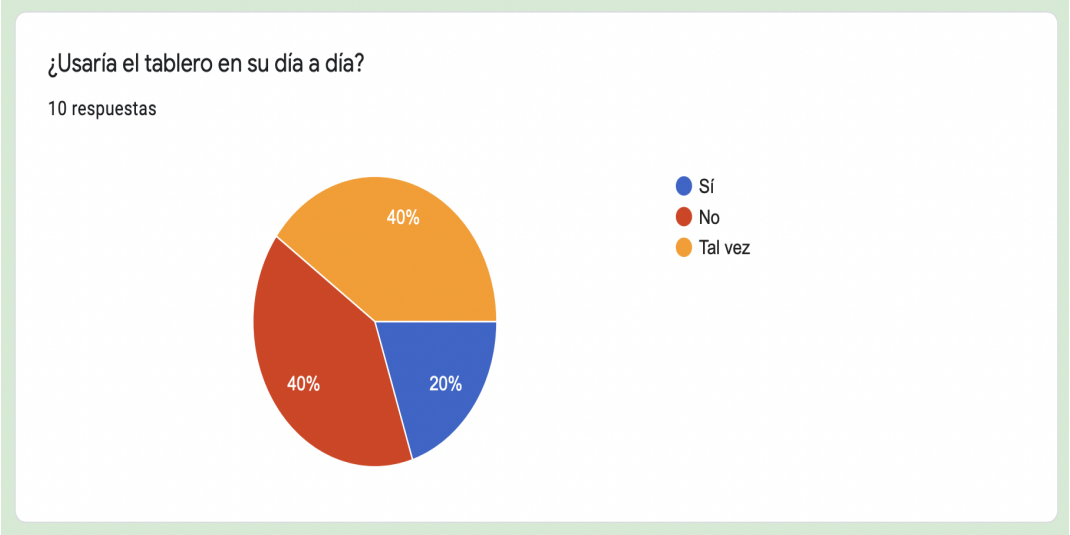


Figura B.17: Resultados sexta pregunta segundo prototipo

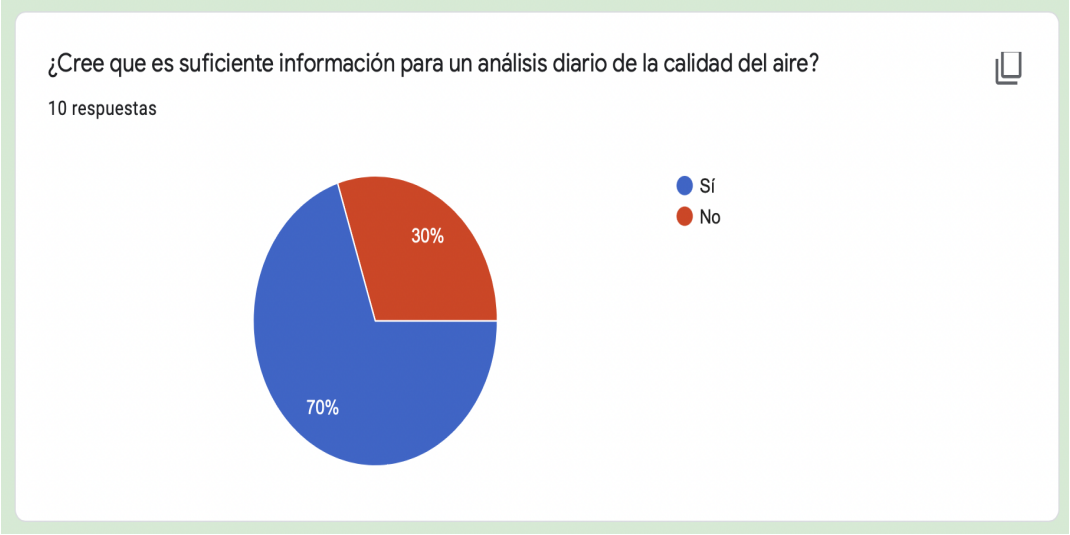


Figura B.18: Resultados séptima pregunta segundo prototipo

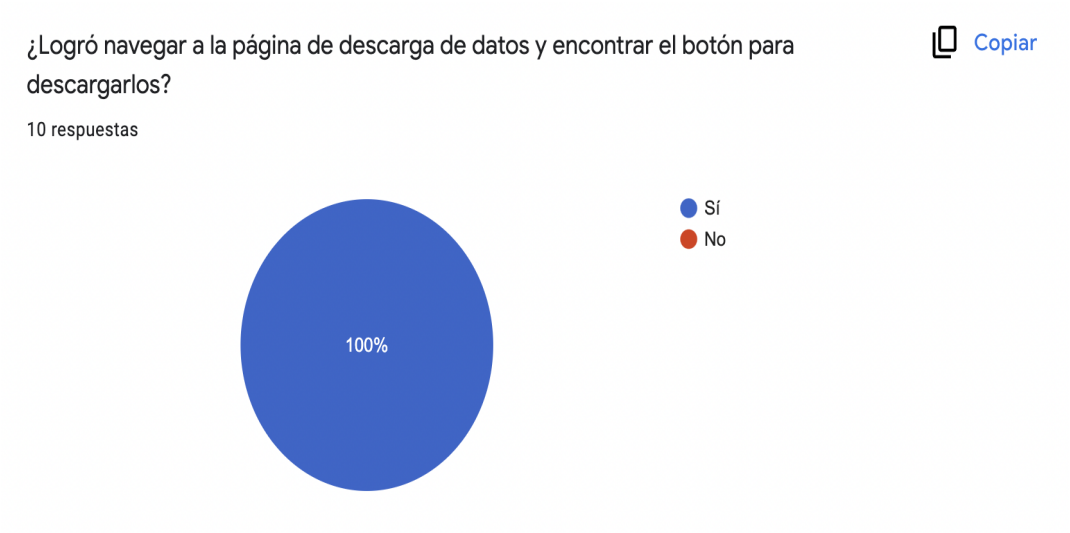


Figura B.19: Resultados octava pregunta segundo prototipo

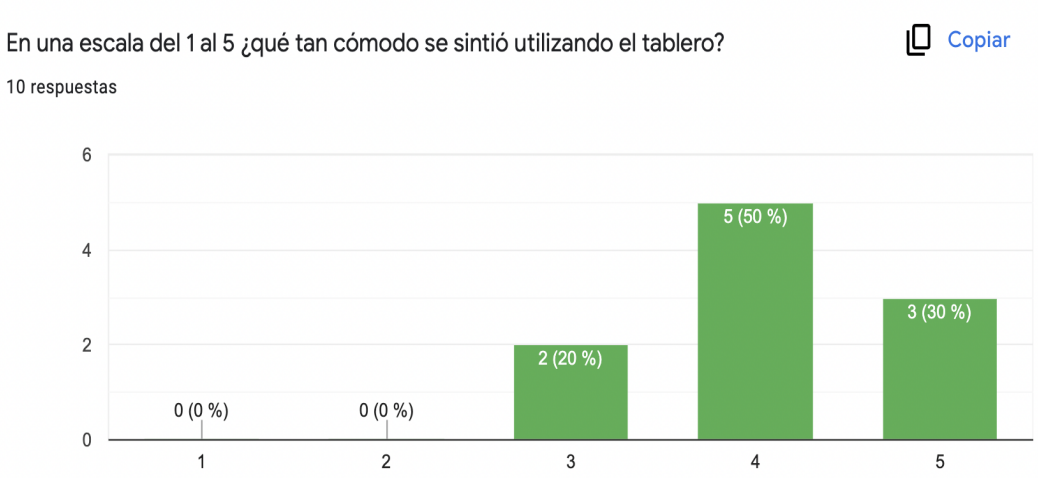


Figura B.20: Resultados novena pregunta segundo prototipo

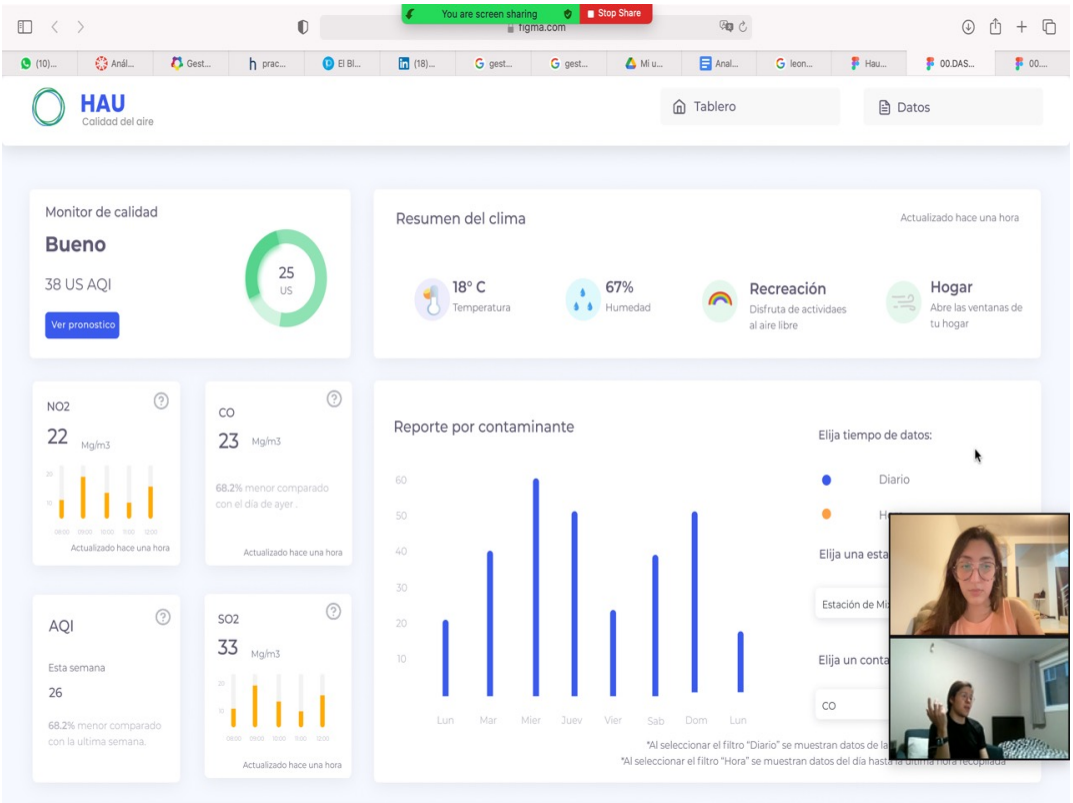


Figura B.21: Prueba virtual segundo prototipo Javier Anleu

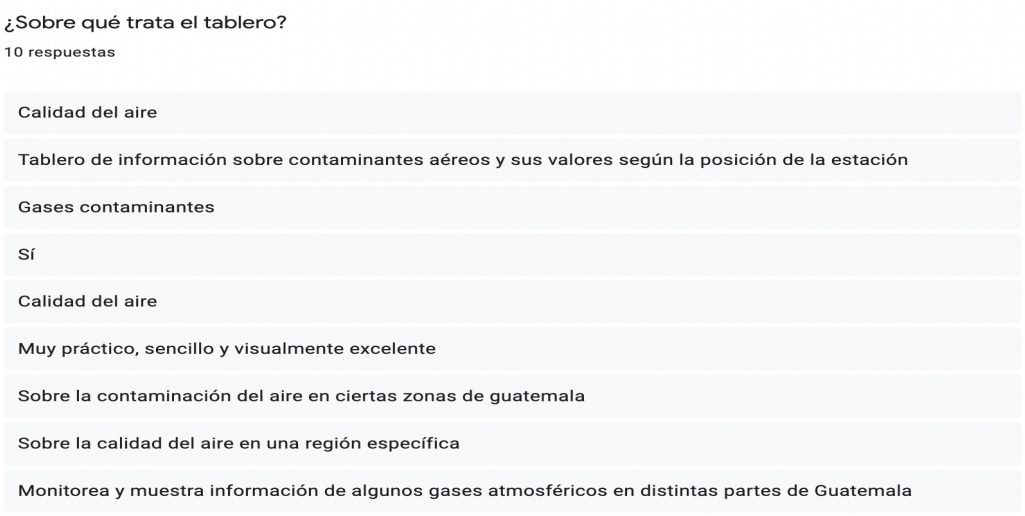


Figura B.22: Resultados primera pregunta prueba final

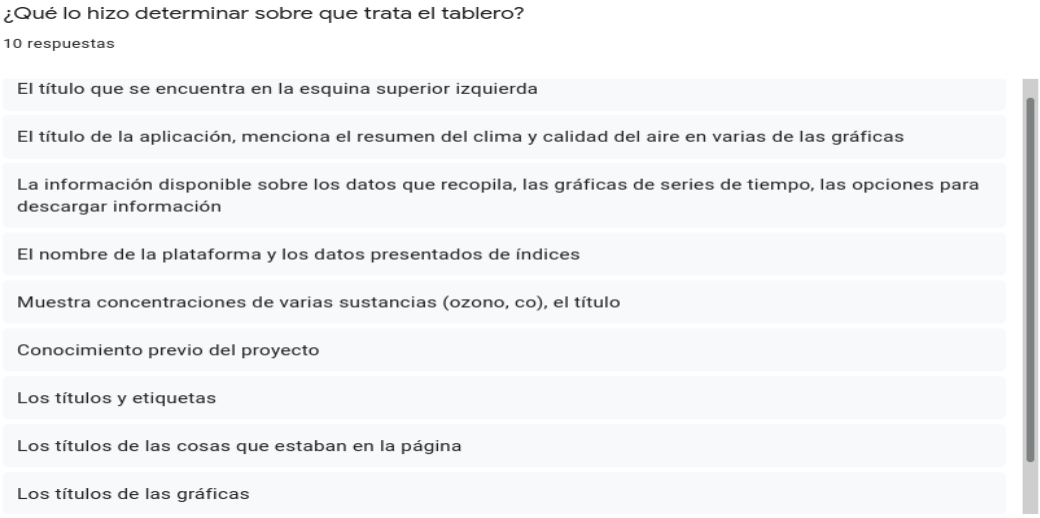


Figura B.23: Resultados segunda pregunta prueba final

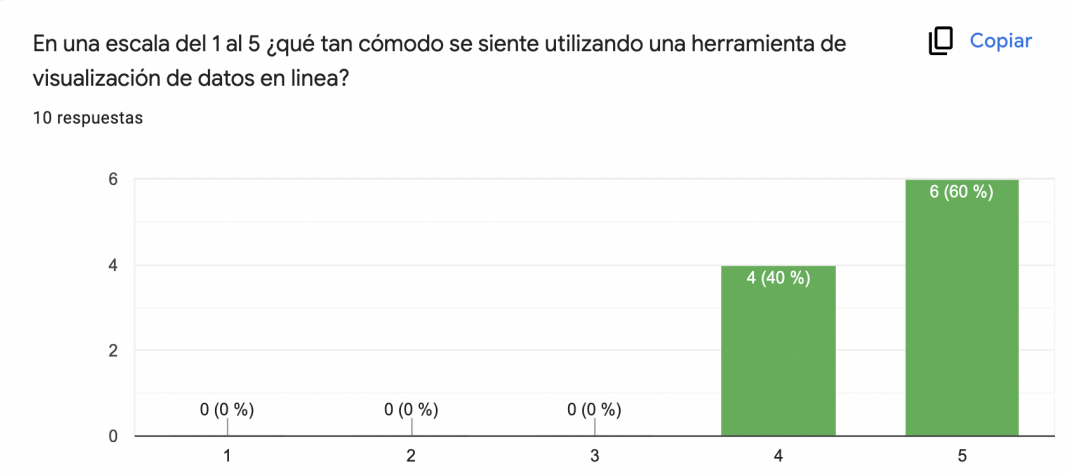


Figura B.24: Resultados tercera pregunta prueba final

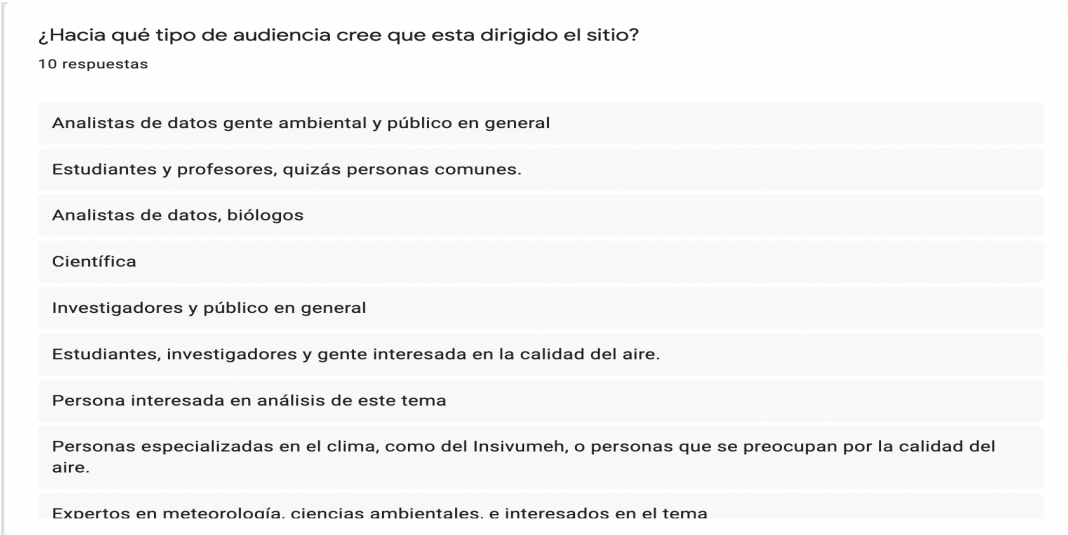


Figura B.25: Resultados cuarta pregunta prueba final

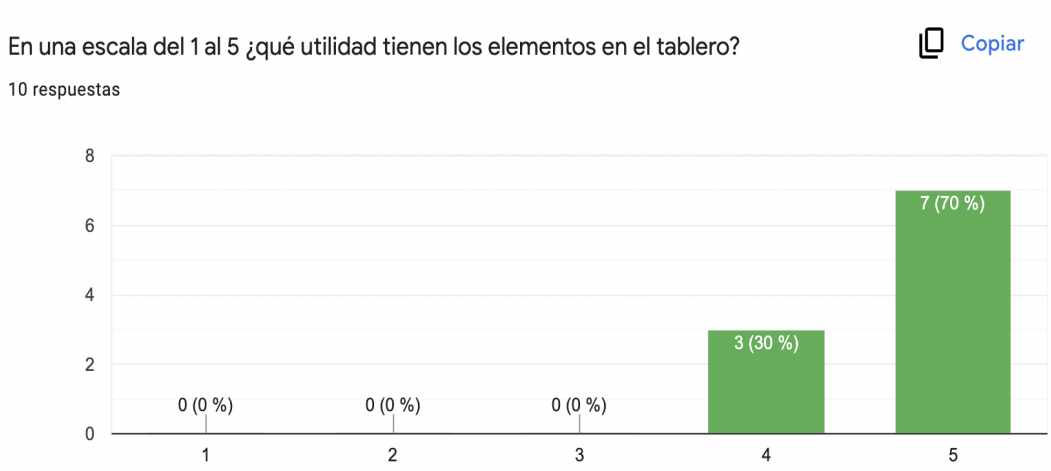


Figura B.26: Resultados quinta pregunta prueba final

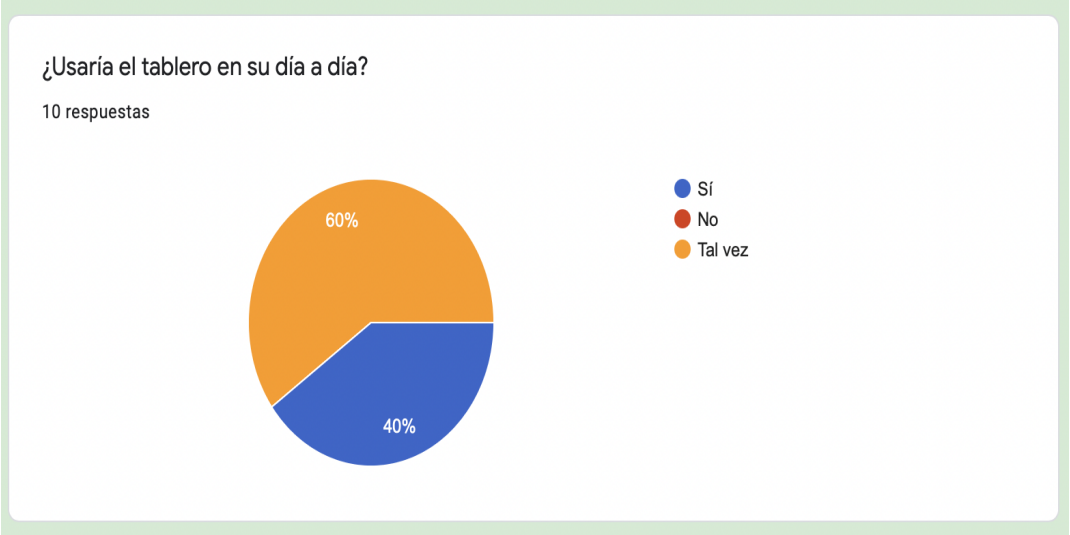


Figura B.27: Resultados sexta pregunta prueba final

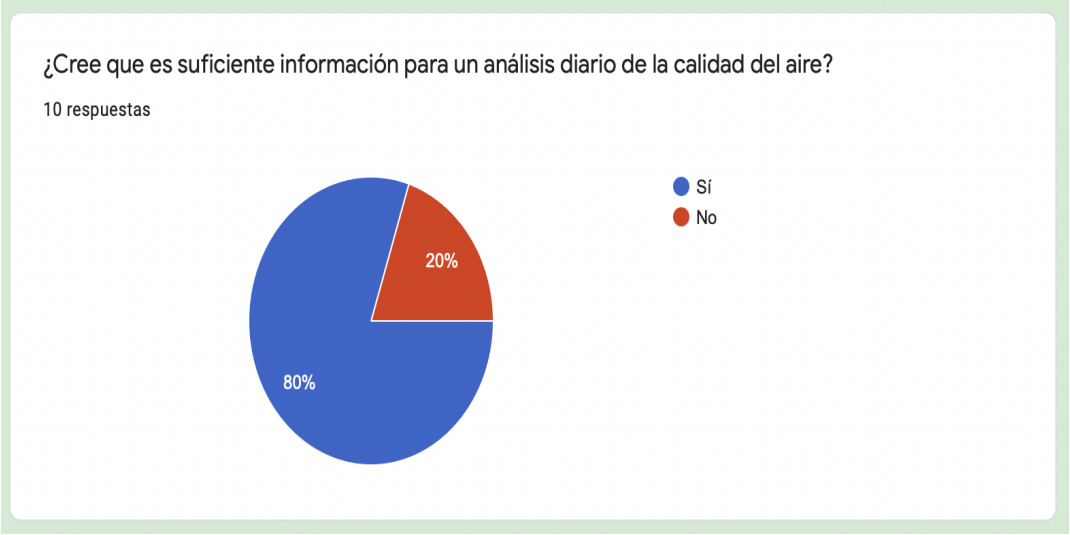


Figura B.28: Resultados séptima pregunta prueba final

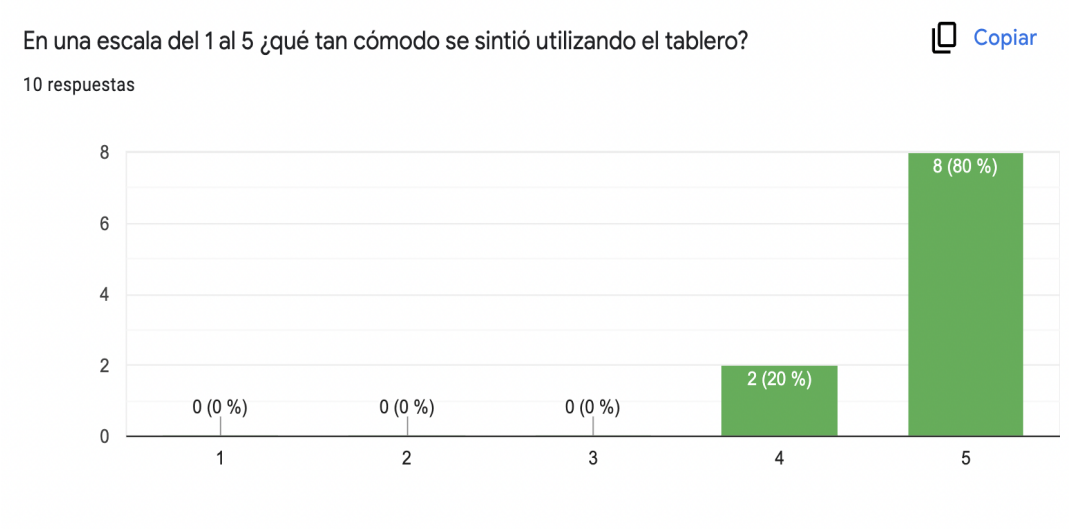


Figura B.29: Resultados octava pregunta prueba final

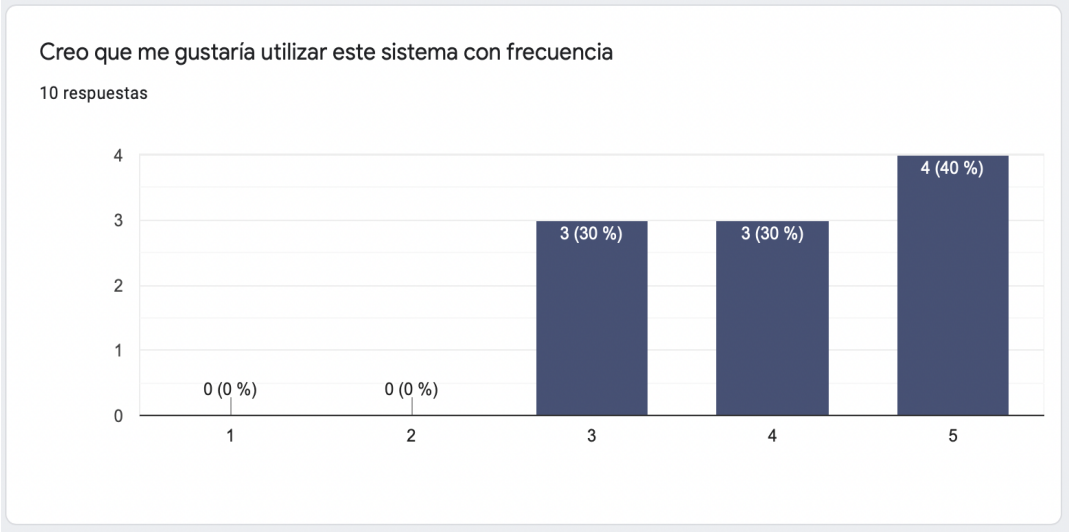


Figura B.30: Resultados primera pregunta SUS

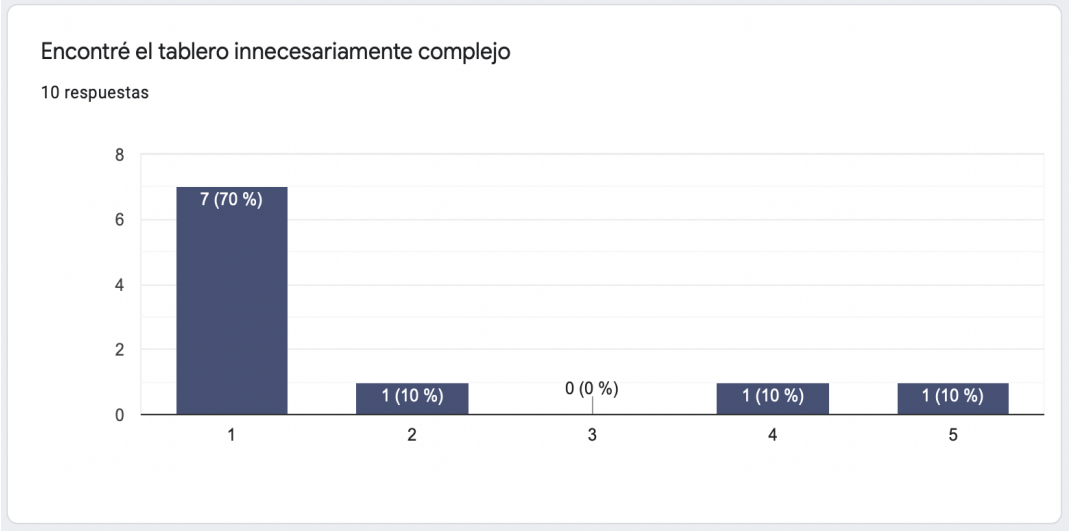


Figura B.31: Resultados segunda pregunta SUS

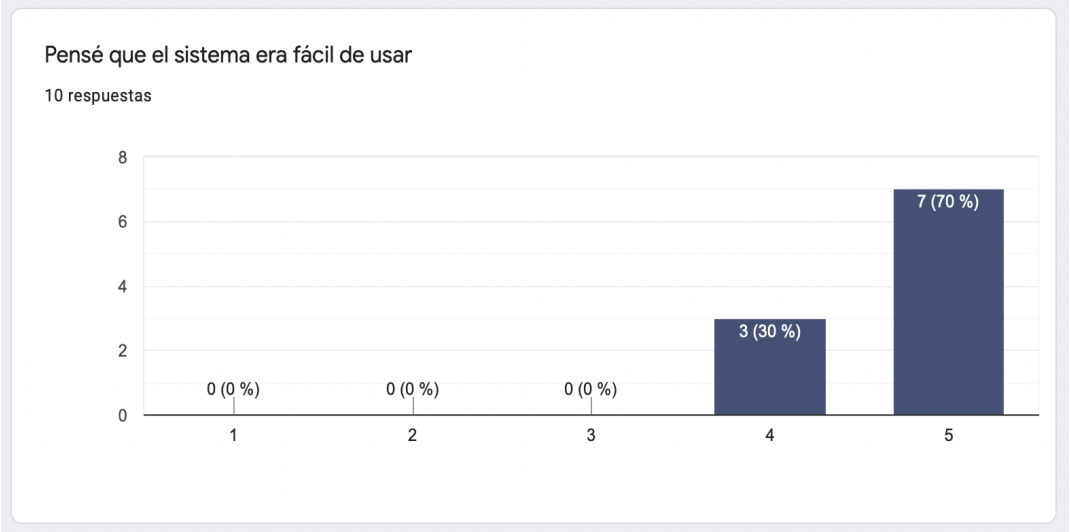


Figura B.32: Resultados tercera pregunta SUS

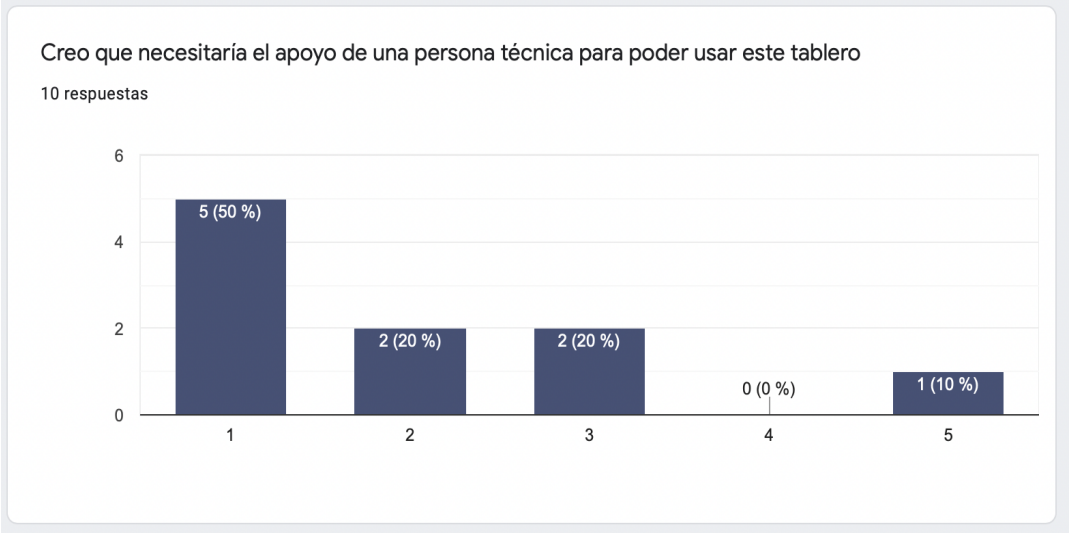


Figura B.33: Resultados cuarta pregunta SUS

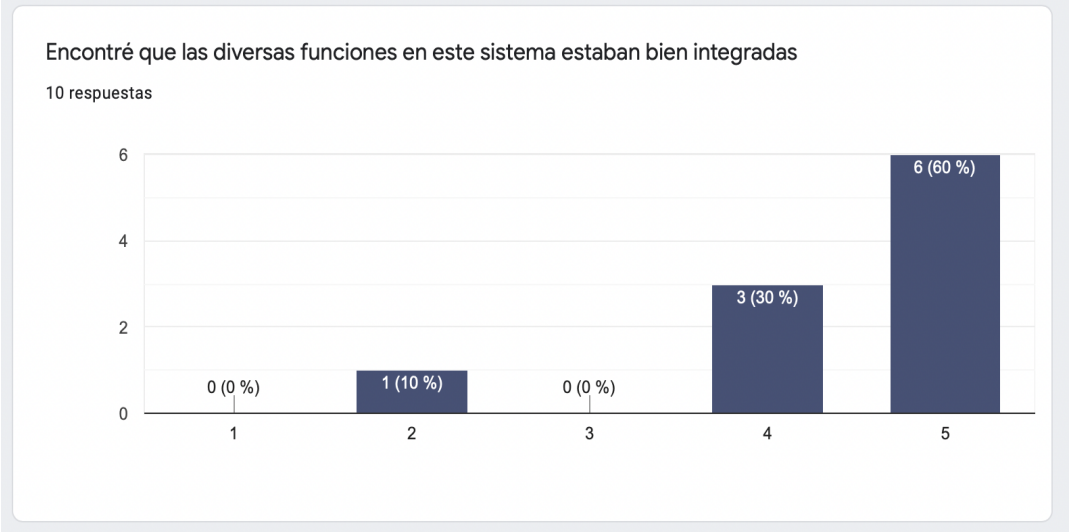


Figura B.34: Resultados quinta pregunta SUS

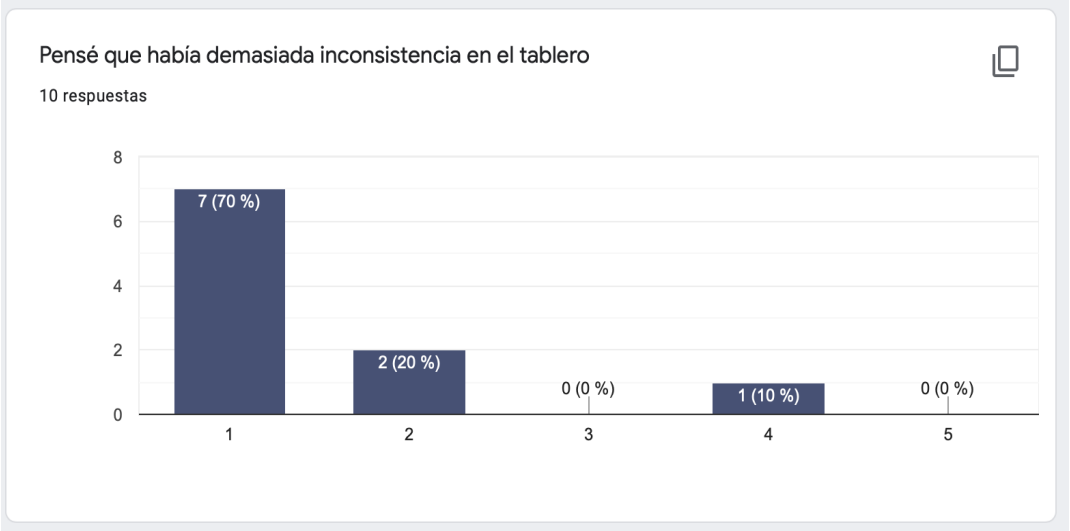


Figura B.35: Resultados sexta pregunta SUS

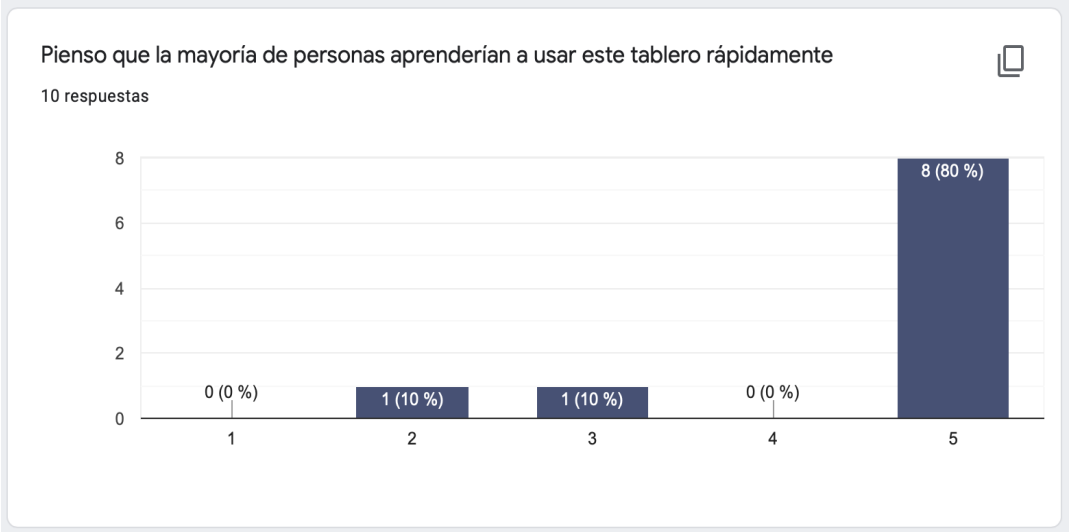


Figura B.36: Resultados séptima pregunta SUS

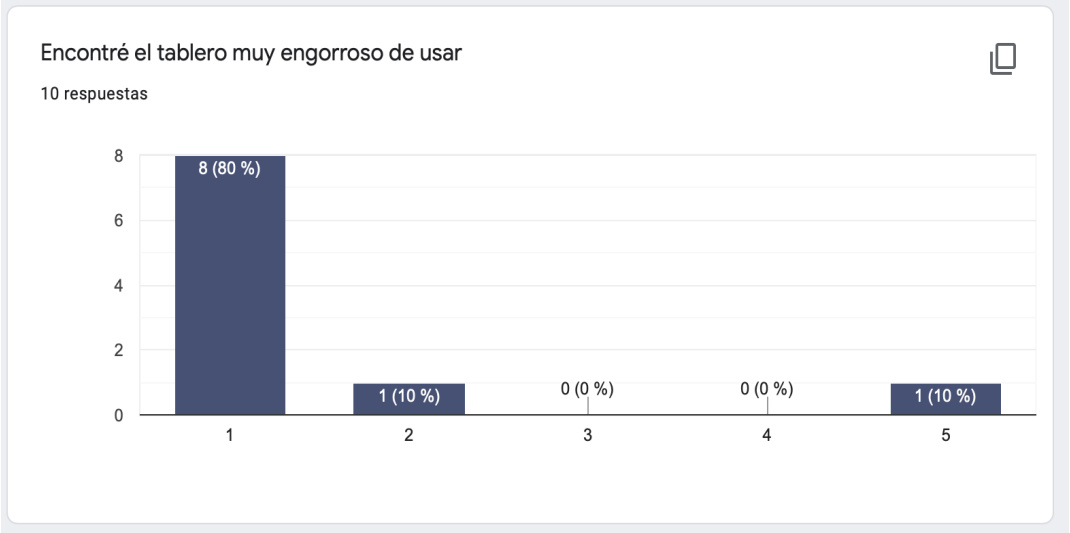


Figura B.37: Resultados octava pregunta SUS

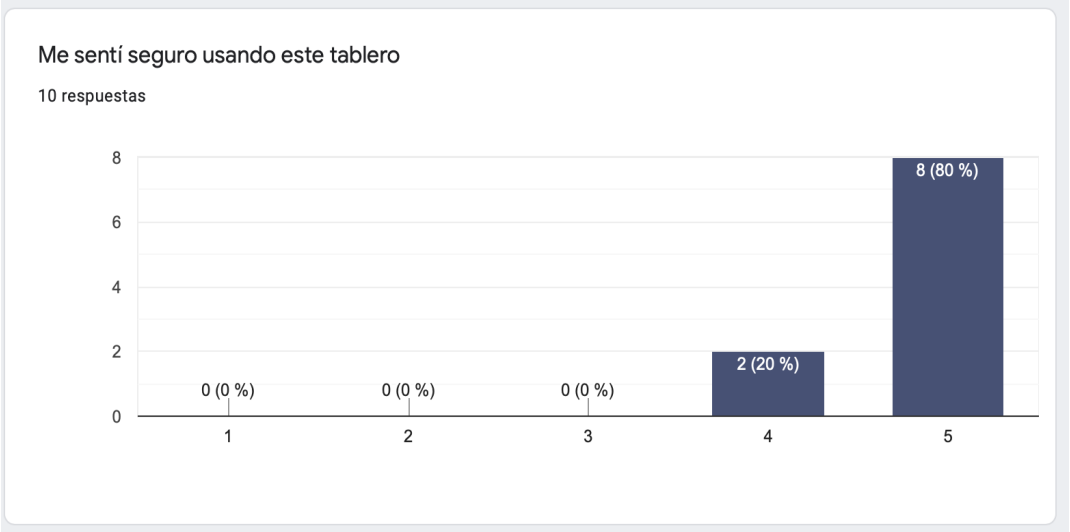


Figura B.38: Resultados novena pregunta SUS

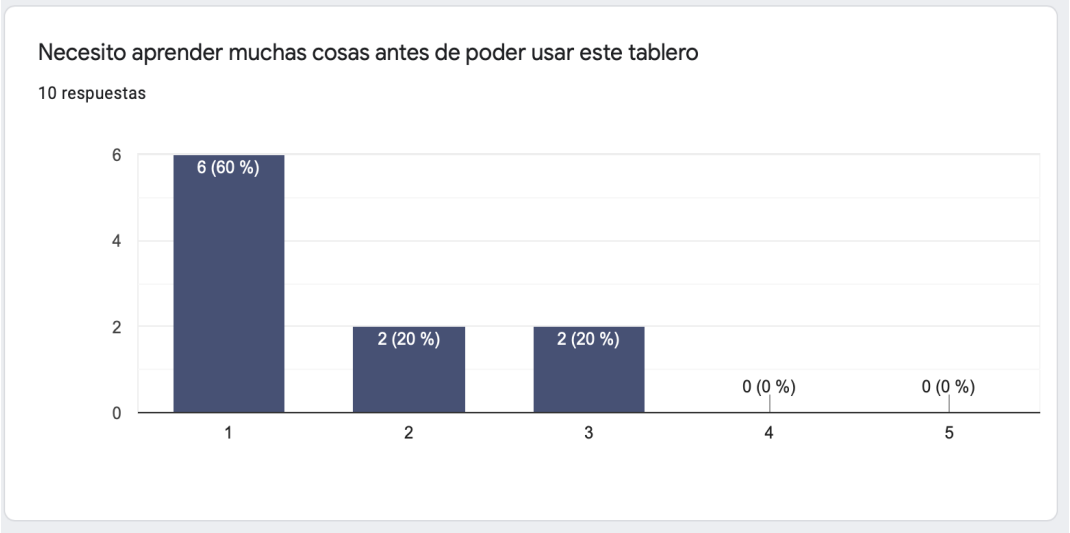


Figura B.39: Resultados decima pregunta SUS

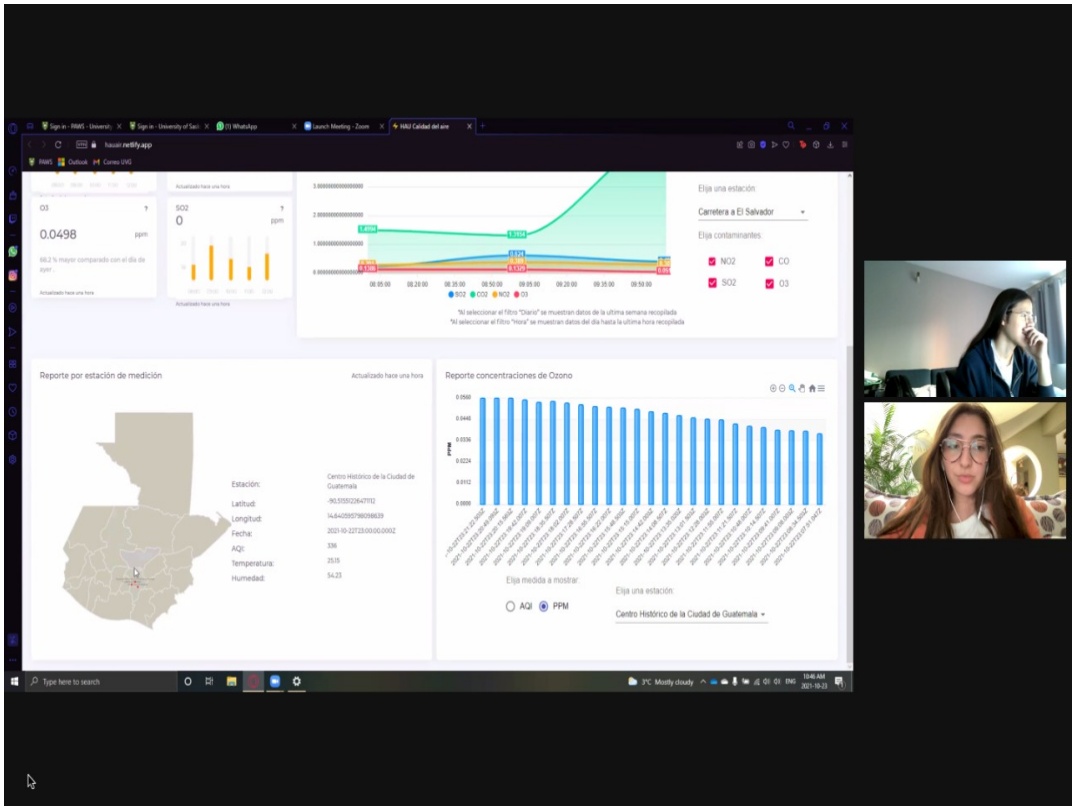


Figura B.40: Prueba virtual último prototipo Javier Anleu estudiante de Biotecnología

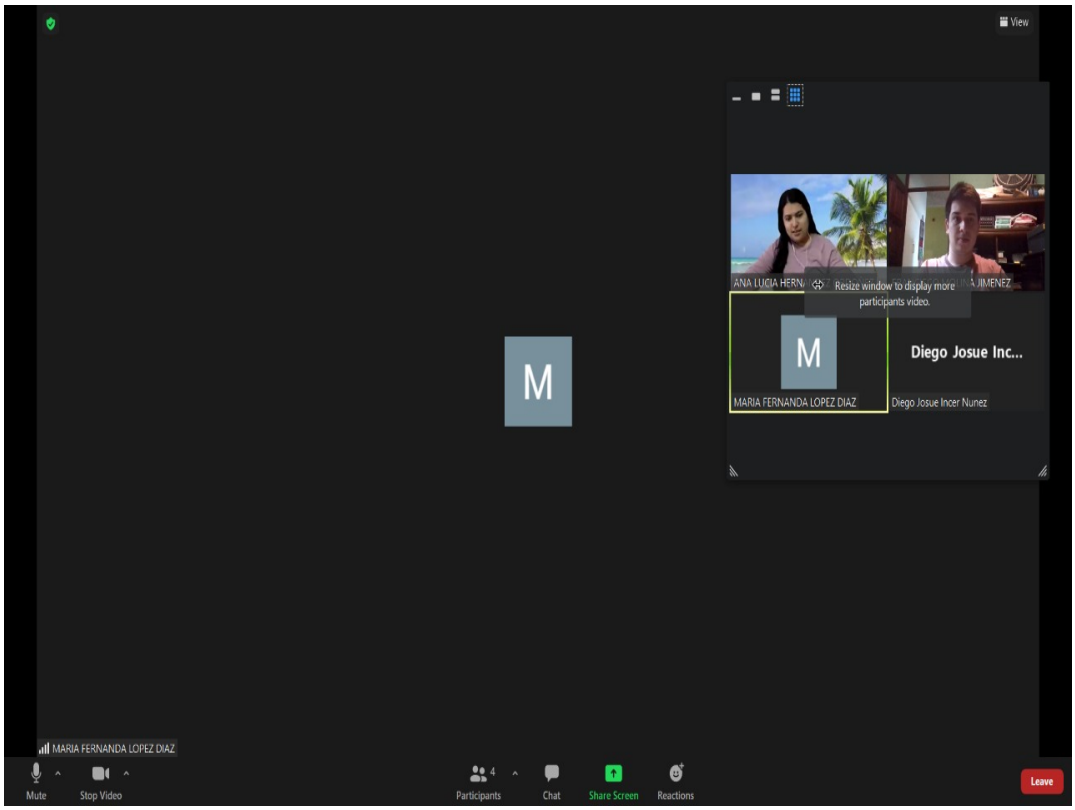


Figura B.41: Prueba virtual último prototipo Diego Incer Investigador, UVG

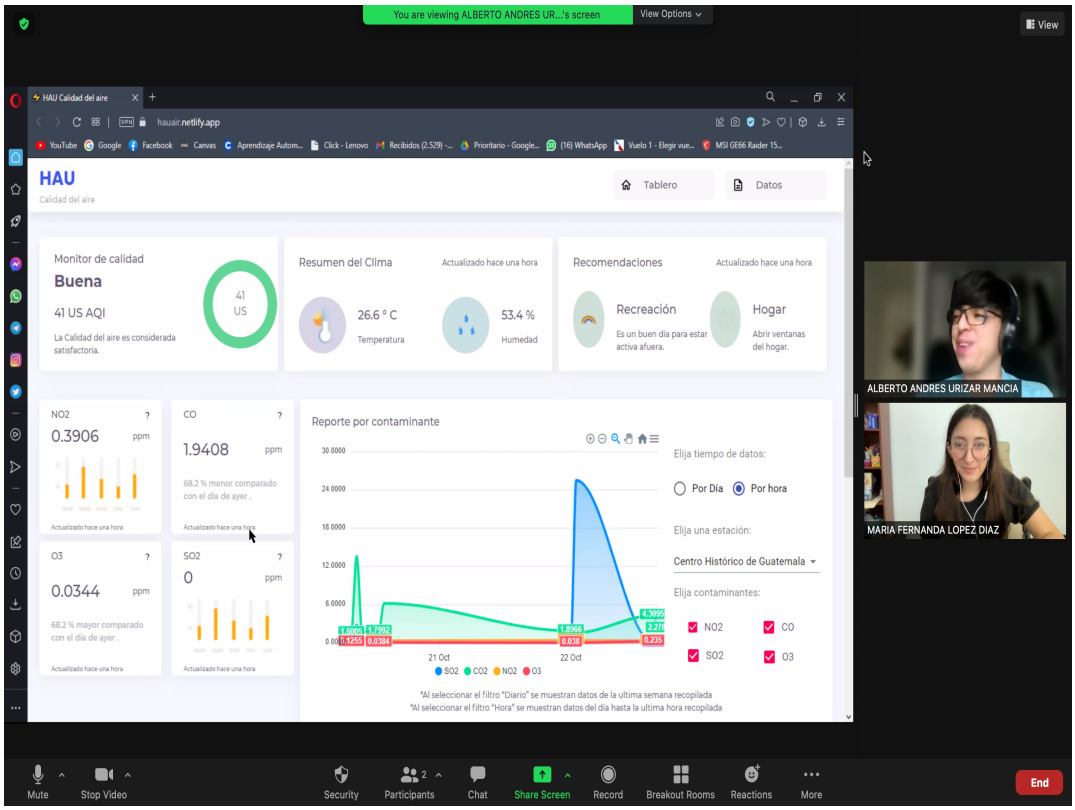


Figura B.42: Prueba virtual último prototipo Andrés Urizar público en general

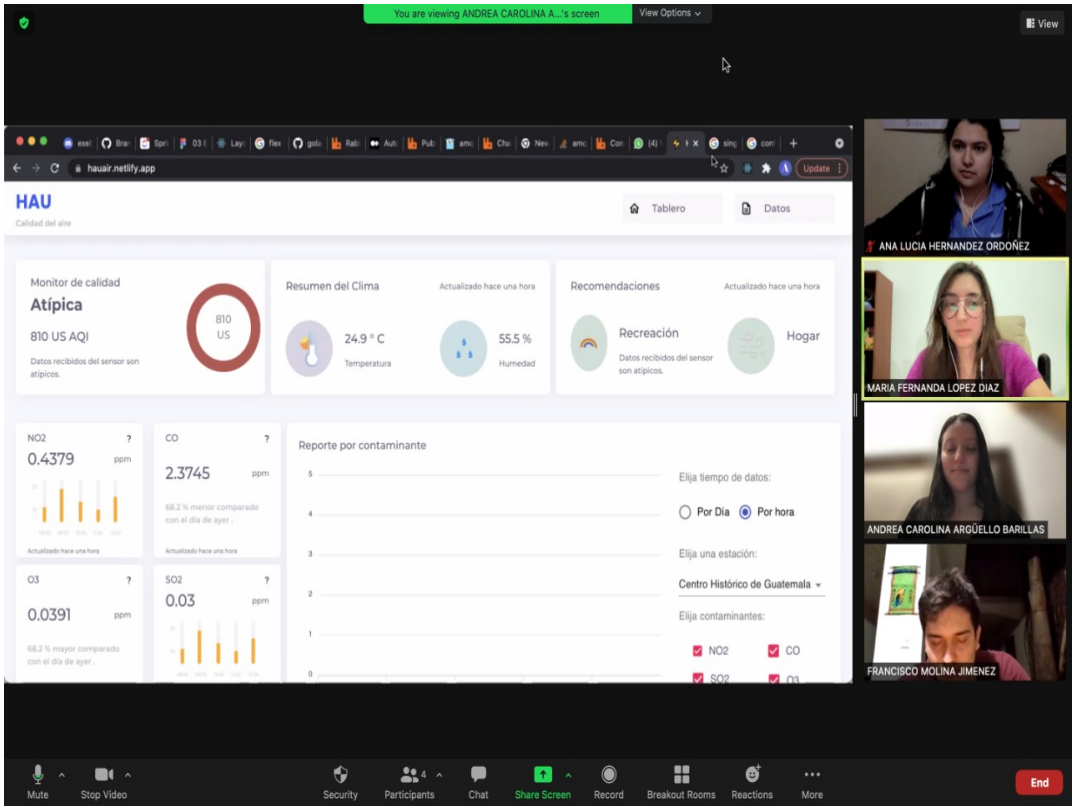


Figura B.43: Prueba virtual último prototipo Andrea Argüello público en general

Glosario

API: Una interfaz de programación de aplicaciones (Application Programming Interface, API por sus siglas en inglés) es un conjunto de definiciones y protocolos que permiten que distintos productos o servicios se comuniquen entre sí y con otros, sin necesidad de saber cómo están implementados. 76

***Availability zones*:** Las zonas de disponibilidad o availability zones son ubicaciones que están físicamente separadas dentro de cada región del proveedor de servicios en la nube, diseñadas para ser tolerante a fallos y asegurar resiliencia. 22

Bandwidth: Máxima capacidad de información que puede ser enviada a través de una red en un periodo de tiempo dado.. 16

Baudios: Unidad de velocidad de transmisión de datos medida en cantidad de símbolos por segundo. 11

Broker: Programa intermediario entre dos entidades independientes. 16

***Clúster*:** En el contexto de programación distribuida, un clúster es una agrupación de instancias computacionales que operan al unísono y de manera coordinada para realizar una carga de trabajo. 55

C++: Lenguaje de programación basado en C, desarrollado en los años 80s. Basado en el paradigma de programación por objetos. 10

***Data lakes*:** Repositorio de almacenamiento centrado que contiene un alto volumen de datos provenientes de varis fuentes en un formato granular y sin procesar. 22

Datasheet: Documento que provee especificaciones técnicas para un producto.. 14

***Data pipelines*:** En el contexto de *data analytics*, un *pipeline* o *data pipeline* es una serie de pasos tomados para procesar la data, desde que se toma la data como insumo hasta que se hace disponible al usuario final, se almacena o queda en su transformación final. 17

***Data streaming*:** Datos que se generan continuamente por uno o varios orígenes de datos, que normalmente envían los registros de datos simultáneamente y en tamaños pequeños. 48

Diagrama Entidad-Relación: Un diagrama entidad-relación – también conocido como modelo entidad relación o ERD por su significado en inglés – es un tipo de diagrama de flujo que indica como las entidades (modeladas como tablas dentro de una base de datos relacional) se relacionan entre sí dentro de un sistema. ix, 56

Endpoint: Dispositivo remoto que se comunica con una red a la cual está conectado.. 12

***Foreign keys*:** Una llave foránea o *foreign key* es una columna o combinación de columnas de una tabla (origen) que hacen referencia hacia columna o columnas de otra tabla (destino), imponen un enlace directo entra las columnas de las tablas para controlar la información que se almacena en la tabla origen. 56

Host: Dispositivo que tiene la capacidad de permitir conectarse a otro dispositivo a una red.. 14

Hotspot: Ubicación física donde se pueden conectar a internet múltiples dispositivos.. 42

IEEE: Asociación global de profesionales especializados en la ingeniería electrónica dedicada a la estandarización de tecnologías.. 12

Internet de las cosas o *Internet of Things*: Comúnmente conocido como IoT por sus siglas en inglés, se define como una red de dispositivos físicos que se conectan e intercambian información con otros dispositivos o servicios a través de una conexión a internet. 22

Kilobytes: Unidad de medida de espacio ocupado por una entidad de software, medido en miles de bytes, que son 8 bits, cada bit representa un caracter 1 o 0.. 32

Llaves criptográficas: Hace referencia al par de llaves (pública y privadas) que se intercambian dos servicios para encriptar su comunicación. 53

Llave primaria o *primary key*: Por lo general, una tabla relacional tiene una columna o una combinación de columnas que contienen valores únicos y que identifican cada fila de la tabla. Esta columna o grupo de columnas conforman lo que se le conoce como llave primaria de la tabla, se utiliza para garantizar la integridad de esta. 56

Manjaro Linux: Distribución del sistema operativo GNU/Linux, basado en la distribución padre Arch.. 32

***Message broker*:** Un bróker de mensajería o *message broker* es un programa que trabaja como intermediario traduciendo los mensajes de comunicación entre sistemas distintos; se encarga de validar, transformar (si es necesario) y verificar el ruteo de mensajes. 53

Micro USB: Puerto de conexión que se encuentra en dispositivos móviles, especialmente los que no son desarrollados por la compañía Apple.. 32

Nodo *master/driver*: Es la instancia computacional encargada de coordinar y manejar el trabajo que se distribuye a lo largo del clúster, este trabajo es ejecutado por los nodos *workers*. 55

Nodos *worker*: Son todas las instancias encargadas de realizar una parte de la carga de trabajo en un clúster. 55

Protoboard: Tablero con orificios que permite interconectar dispositivos a través de cables sin necesidad de soldar, funcionando como una placa madre para dispositivos prototipo.. 42

Protocolo HTTPS: HTTPS (HyperText Transfer Protocol Secure, protocolo seguro de transferencia de hipertexto) es un protocolo de comunicación de Internet que protege la integridad y la confidencialidad de los datos de los usuarios entre sus ordenadores y el sitio web. 76

Python: Lenguaje de alto nivel interpretado, de sistema de tipos suave.. 32

***Query*:** En el contexto de bases de datos – específicamente de relacionales – un *query* son consultas para obtener o escribir información del almacén de datos, estos son escritos en lenguajes de consulta (*query language*). 53

R: Lenguaje de alto nivel interpretado, utilizado principalmente para análisis estadístico y de datos.. 34

RAM: Random Access Memory por sus siglas en inglés. Se refiere a un tipo de memoria presente en una computadora. Aquí se guardan variables, instrucciones y programas cargados. Esta memoria se vacía por completo cada vez que el dispositivo se desconecta de alimentación eléctrica.. 32

***Requests* y *responses*:** En el contexto de tecnologías web, una solicitud o *request* es una petición de un cliente hacia un servidor por información. Una vez el servidor recibe la petición, le procesa una respuesta o *response*, que envía por el mismo canal de comunicación en el que recibió la petición. 76

Router: Dispositivo de red que se encarga de reenviar paquetes de datos desde internet a redes locales.. 12

SAS TOKEN: El token de firma de acceso compartido (Shared Access Signature, SAS por sus siglas en inglés) es una cadena que se genera mediante alguna de las bibliotecas cliente de Azure Storage. Se utiliza como parte de una solicitud para acceder a elementos almacenados dentro de Azure Storage, verificar si la firma es válida y luego autorizar la solicitud. 42, 76

Script: Pequeña pieza de software de un solo propósito.. 34

Sistemas embebidos: Sistema de computación desarrollado con el fin de realizar tareas específicas.. 10

Smog: Nube formada por contaminantes gaseosos como: Hollín, dióxido de carbono, ozono, etc.. 6

Software: Programa o conjunto de programas que permiten ejecutar tareas o instrucciones a una computadora.. 15

Stock: Disponible para su compra. En existencia.. 32

***Suite*:** Un *software suite* es una colección de programas computacionales o aplicaciones que tienen funcionalidades y objetivos relacionados, pueden compartir interfaces de usuario y los caracteriza la facilidad de intercambiar información o datos entre sí. 48

TLS: Protocolo de seguridad que encripta información en una conexión entre distintos dispositivos conectados a través de una red.. 16

USB: Universal serial bus (por sus siglas en inglés). Puerto de comunicación serial de uso estándar.. 37

Zonas geográficas o regiones: Una región dentro de un proveedor en la nube es un grupo de *datacenters* (instalaciones donde se alojan servidores) ubicados dentro de un perímetro definido por latencia en comunicación. Estos *datacenters* están físicamente conectados a través de una red regional de baja latencia. 22