UNIVERSIDAD DEL VALLE DE GUATEMALA
Facultad de Ingeniería

***Platyfa*: Caso de estudio del desarrollo de un Videojuego en un Entorno Universitario Colaborativo**

Trabajo de graduación en modalidad de Megaproyecto Tecnológico
presentado por

Maria Isabel Solano Bonilla
Jessica Pamela Ortiz Ixcot
José Jorge Pérez Aldana
Mariana David Sosa
José Rodrigo Barrera García

para optar al grado académico de Licenciado en Ingeniería en Ciencias de la Computación y Tecnologías de la Información

Guatemala
2024

UNIVERSIDAD DEL VALLE DE GUATEMALA
Facultad de Ingeniería

Platyfa: Caso de estudio del desarrollo de un Videojuego en un Entorno Universitario Colaborativo

Trabajo de graduación en modalidad de Megaproyecto Tecnológico
presentado por

Maria Isabel Solano Bonilla
Jessica Pamela Ortiz Ixcot
José Jorge Pérez Aldana
Mariana David Sosa
José Rodrigo Barrera García

para optar al grado académico de Licenciado en Ingeniería en Ciencias de la Computación y Tecnologías de la Información

Guatemala
2024

Vo.Bo.:

(f) 

Ing. Alhvi Balcárcel

Tribunal Examinador:

(f) 

Ing. Alhvi Balcárcel

(f) X 

Ing. Eddy Castro

(f) 

Ing. Douglas Barrios

Fecha de aprobación: Guatemala, 29 de noviembre de 2024.

Agradecimientos

Expresamos nuestro más sincero agradecimiento a Alhvi por el invaluable apoyo, dedicación y guía brindados a lo largo de este proyecto. Asimismo, extendemos nuestra gratitud a los asesores, quienes con generosidad compartieron su conocimiento y experiencia, contribuyendo significativamente al desarrollo de este trabajo. Finalmente, agradecemos a la Universidad del Valle de Guatemala por proporcionarnos el espacio, los recursos y la oportunidad de aprender y colaborar en un entorno académico que fomenta la creatividad y el crecimiento profesional.

Índice

Lista de figuras

Lista de cuadros

Resumen

El proyecto *Platyfa: Caso de Estudio del Desarrollo de un Videojuego en un Entorno Universitario Colaborativo* representa una iniciativa interdisciplinaria orientada al diseño, desarrollo e implementación de un videojuego que integra elementos de programación avanzada, modelado y animación 3D, diseño de interfaces y experiencias de usuario (UI/UX/GX), y desarrollo web. Este megaproyecto tiene como objetivo principal ofrecer una experiencia de juego inmersiva y divertida, mientras fomenta el aprendizaje técnico, la colaboración entre disciplinas y la innovación en la emergente industria de videojuegos en Guatemala.

En el módulo de **Análisis y Diseño UI/UX/GX**, se creó una interfaz de usuario intuitiva y atractiva que permitió una experiencia fluida e inmersiva para los jugadores. Este proceso incluyó la ideación de la narrativa, el diseño de personajes, mecánicas y niveles, así como la implementación de un HUD funcional. Las sesiones de *playtesting* y encuestas proporcionaron retroalimentación clave que permitió ajustar elementos de diseño para alinear mejor las expectativas de los jugadores con la experiencia final.

El módulo de **Animación 3D** se enfocó en transmitir la narrativa del juego mediante animaciones caricaturescas de alta calidad. Usando herramientas como *Blender*, *Mixamo* y *Unity*, se crearon personajes animados con movimientos fluidos e integraciones eficientes, mejorando significativamente la conexión emocional y la inmersión del jugador. Las pruebas de usabilidad confirmaron que las animaciones cumplían con los estándares establecidos y enriquecieron la experiencia de juego.

El desarrollo del módulo **Web** permitió complementar el videojuego con una plataforma digital dinámica que ofrece estadísticas en tiempo real y mejora la experiencia del usuario. A través de una landing page y un sistema de análisis basado en datos recopilados mediante una API, los jugadores pueden visualizar su rendimiento mientras los desarrolladores analizan métricas clave para optimizar el equilibrio y la jugabilidad del juego.

El módulo de **Programación de Enemigos** integró inteligencia artificial avanzada para crear comportamientos dinámicos y desafiantes en los enemigos, ajustando su dificultad para jugadores casuales y experimentados. Este sistema permitió enriquecer la narrativa y jugabilidad del videojuego, ofreciendo interacciones significativas y satisfactorias.

Finalmente, el módulo de **Modelado 3D** destacó la creación de personajes y objetos optimizados para garantizar tanto el rendimiento técnico como la calidad estética del juego. Con el uso de *Blender*, se trabajó en topologías eficientes, mapeo UV preciso y técnicas de optimización que aseguraron la integración fluida de los modelos en el entorno de desarrollo.

En conclusión, *Platyfa* no solo alcanzó sus objetivos técnicos y narrativos, sino que también

promovió el crecimiento personal y profesional de los integrantes del proyecto, fortaleciendo sus habilidades en diseño, programación y trabajo colaborativo. Este megaproyecto establece un precedente para la industria de videojuegos en Guatemala, posicionándola como un sector emergente con gran potencial creativo y tecnológico.

CAPÍTULO 1

Introducción

El megaproyecto *Platyfa: Caso de Estudio del Desarrollo de un Videojuego en un Entorno Universitario Colaborativo* representa una iniciativa interdisciplinaria que integra elementos fundamentales del desarrollo de videojuegos, como el diseño UI/UX/GX, el modelado y la animación 3D, la programación de inteligencia artificial para enemigos y el desarrollo de módulos web. Este proyecto tiene como objetivo principal diseñar y desarrollar un videojuego inmersivo y divertido que combine aspectos técnicos y creativos, mientras fomenta el aprendizaje colaborativo, la innovación y el fortalecimiento de habilidades entre sus participantes. En un contexto como el guatemalteco, donde la industria de los videojuegos se encuentra en una etapa emergente, este trabajo busca no solo crear un producto de calidad alineado con estándares internacionales, sino también contribuir al desarrollo local del sector.

El diseño de la interfaz de usuario (UI), la experiencia de usuario (UX) y la experiencia de jugabilidad (GX) fue fundamental para establecer un medio de interacción intuitivo y atractivo entre los jugadores y el sistema. Este módulo se centró en el desarrollo de interfaces claras y funcionales que conectaran a los jugadores con la narrativa del juego, asegurando una experiencia de juego satisfactoria. Herramientas iterativas permitieron ajustar elementos clave, como menús, pantallas de configuración y la HUD, con base en pruebas de *playtesting*. Estas decisiones no solo facilitaron la navegación y el entendimiento de las mecánicas del juego, sino que también ayudaron a integrar la historia de *Platyfa* de forma coherente, creando un diseño que capturó la atención de los jugadores y reforzó su inmersión.

El modelado 3D, por su parte, fue un componente esencial en la creación de personajes, enemigos y objetos tridimensionales. Este módulo se apoyó en técnicas avanzadas como el *blocking*, la retopología y el mapeo UV utilizando *Blender*. Estas herramientas permitieron diseñar activos visuales estilizados, optimizados para garantizar un equilibrio entre estética y rendimiento. El ornitorrinco Gaus, los enemigos dingos y búhos, y diversos elementos del entorno fueron desarrollados con un enfoque centrado en la calidad visual y narrativa. Este trabajo evidenció cómo el modelado tridimensional contribuye significativamente a la inmersión del jugador y a la coherencia del universo del videojuego, proporcionando un mundo visualmente atractivo que refuerza la narrativa y la jugabilidad.

El módulo de animación 3D aportó dinamismo al videojuego mediante herramientas como *Mixamo*, *Blender* y *Unity*, que permitieron la creación de movimientos fluidos y transiciones coherentes. Los sistemas de rigging y árboles de animación integraron de manera efectiva las interacciones de los personajes con su entorno, mejorando la jugabilidad y la narrativa.

De manera destacada, el módulo de programación de enemigos fue esencial para enriquecer la jugabilidad y ofrecer desafíos equilibrados y dinámicos. Este módulo se enfocó en el desarrollo de inteligencia artificial avanzada para crear comportamientos adaptativos, ajustando patrones de ataque, defensa y movimiento según el rendimiento del jugador. A través del análisis de videojuegos existentes y de la implementación de redes neuronales en el motor de desarrollo *Unity*, se diseñaron enemigos que no solo representaron obstáculos significativos, sino que también reforzaron la narrativa del juego y aumentaron la inmersión de los jugadores. Enemigos como los dingos y los búhos, con patrones de comportamiento únicos, añadieron profundidad y variedad al juego, mientras que se trabajó en un sistema de IA adaptable que promete incrementar el nivel de personalización y reto en niveles futuros.

Finalmente, el desarrollo del módulo web complementó al videojuego al proporcionar herramientas adicionales para la promoción y la recopilación de datos. La creación de una *landing page* dinámica, un sistema de análisis estadístico y una API propia permitió ofrecer a los jugadores estadísticas detalladas sobre su desempeño, mientras que los desarrolladores pudieron analizar métricas clave para optimizar la jugabilidad y planificar actualizaciones futuras.

En conjunto, *Platyfa* es un esfuerzo integral que combina diseño, programación y narrativa para desarrollar un videojuego de calidad que trasciende lo técnico, ofreciendo una experiencia inmersiva y significativa para los jugadores. Este proyecto no solo alcanzó su meta de crear un producto que destaca por su cohesión y atractivo visual, sino que también fortaleció las habilidades técnicas, creativas y colaborativas de su equipo, posicionándose como un ejemplo valioso para el desarrollo de la industria de videojuegos en Guatemala.

CAPÍTULO 2

Objetivos

2.1. Objetivo general

Diseñar, desarrollar e implementar un videojuego colaborativo en un entorno universitario que integre elementos de programación de inteligencia artificial para enemigos, modelaje 3D, animación, diseño UI/UX/GX y desarrollo web, con el objetivo de ofrecer una experiencia de juego inmersiva y divertida, mientras se fortalece el aprendizaje interdisciplinario, las habilidades técnicas y la capacidad de trabajo en equipo de los integrantes del proyecto.

2.2. Objetivos específicos

- Desarrollar e implementar una interfaz de usuario (UI) y experiencia de usuario (UX) en el videojuego a desarrollar, garantizando que sean intuitivas, atractivas y capaces de transmitir eficazmente los sentimientos y emociones deseados a través de la jugabilidad (*Gameplay Experience*, GX), para así lograr una integración armoniosa de la UI, UX y GX, y ofrecer una experiencia de juego inmersiva y satisfactoria para el usuario final.
- Desarrollar un módulo de programación avanzado para los enemigos de un videojuego que permita crear comportamientos complejos e inmersivos, mejorando la experiencia de juego mediante desafíos más dinámicos y reales.
- Validar la aplicabilidad y el impacto del modelado 3D en el desarrollo de videojuegos, enfocándose en el uso de *Blender* como herramienta esencial para la creación de personajes, objetos y entornos tridimensionales de alta calidad.
- Animar modelos 3D de manera caricaturesca para contar la historia de Gaus y la Revolución Ornitorense.[en] un videojuego de ciencia ficción, y ofrecer una experiencia visualmente atractiva.
- Desarrollar una herramienta digital complementaria para el videojuego Platyfa que sirva como un punto de promoción, brinde información relevante sobre el juego y permita a los jugadores consultar estadísticas de sus partidas.

CAPÍTULO 3

Justificación

El desarrollo del videojuego Platyfa: Caso de Estudio del Desarrollo de un Videojuego en un Entorno Universitario Colaborativo surge de la necesidad de integrar conocimientos interdisciplinarios en un contexto práctico, fomentando la creación de un producto de alta calidad que combine innovación, aprendizaje técnico y colaboración. Este proyecto busca contribuir al crecimiento de la industria de videojuegos en Guatemala, un sector emergente con más de 200 videojuegos desarrollados localmente, que presenta un potencial significativo para fortalecer la economía creativa y tecnológica del país [66].

El módulo de Análisis y Diseño UI/UX/GX justifica su importancia por la necesidad de crear interfaces y experiencias de usuario funcionales, intuitivas y estéticamente atractivas. Tal como lo plantea Turunen, J [120], el diseño de una interfaz de usuario eficaz requiere planificación, iteración y pruebas continuas. Asimismo, el énfasis en la experiencia de usuario y la experiencia de juego (GX) es fundamental para ofrecer una narrativa inmersiva y una jugabilidad satisfactoria. Este módulo busca, además, que el diseño del videojuego cumpla con los estándares actuales, tomando en cuenta principios como la Ley de Hick y los lineamientos de Jakob Nielsen [140], que orientan hacia una interacción fluida y efectiva.

El módulo de Modelado 3D evidencia su relevancia en la creación de personajes, objetos y entornos tridimensionales que contribuyan a la narrativa visual y al desempeño técnico del videojuego. A nivel global, la industria de los videojuegos alcanzará un valor de mercado de 426.02 mil millones de dólares para 2029, con un crecimiento anual proyectado del 9.32 % [132]. Este crecimiento destaca la demanda de modelado 3D de calidad para la producción de contenido innovador. En Guatemala, herramientas como Blender permiten democratizar el acceso a tecnologías avanzadas, impulsando el desarrollo de habilidades técnicas locales y fomentando la internacionalización de los desarrolladores guatemaltecos.

El módulo de Animación 3D se justifica por su impacto en la narrativa y la experiencia del usuario. Según datos de la Entertainment Software Association (ESA), el 75 % de los jugadores consideran los gráficos y animaciones como factores decisivos al elegir un videojuego [58]. Herramientas como Blender, Mixamo y Unity no solo facilitan la creación de animaciones fluidas y expresivas, sino que también optimizan los flujos de trabajo, garantizando una integración eficiente entre narrativa y jugabilidad [79] . Este módulo permite transmitir emociones y dinamismo en el juego, enriqueciendo la conexión del jugador con la historia.

El Módulo Web aporta valor al proyecto mediante la creación de una plataforma complementaria

para la promoción y el análisis de datos del videojuego. La implementación de una página web dinámica mejora la experiencia del usuario al centralizar información relevante, como actualizaciones y estadísticas de juego, y fomenta la interacción social mediante funcionalidades como el compartir logros en redes sociales [81]. Desde la perspectiva del equipo de desarrollo, el módulo web facilita el análisis de métricas clave para optimizar la jugabilidad y el equilibrio del juego, además de servir como herramienta para la toma de decisiones estratégicas y el marketing digital.

Finalmente, el módulo de Programación de Enemigos es crucial para garantizar un nivel de desafío balanceado que mantenga la jugabilidad interesante y accesible para jugadores casuales. Según Rollings y Adams [117], la variedad y diseño de enemigos son esenciales para la satisfacción del jugador. Este módulo incorpora inteligencia artificial avanzada para crear comportamientos dinámicos e inmersivos que mejoren la experiencia de juego y enriquezcan la narrativa, mientras se consideran las necesidades y habilidades de diferentes perfiles de jugadores.

En conjunto, Platyfa representa un esfuerzo interdisciplinario que integra diseño, programación, animación, modelado y desarrollo web para crear un videojuego innovador y educativo. Este proyecto no solo impulsa el aprendizaje y la colaboración, sino que también posiciona a Guatemala como un actor emergente en la industria de los videojuegos, promoviendo el crecimiento de la economía creativa y tecnológica en la región.

CAPÍTULO 4

Marco teórico

4.1. Historia y evolución de los videojuegos

4.1.1. Orígenes y primeros desarrollos

Según M. G. [92], "los videojuegos han experimentado una evolución impresionante a lo largo de los años". Desde los simples juegos de arcade hasta los complejos mundos virtuales de hoy en día, la industria ha crecido exponencialmente. La creación de enemigos dentro de un videojuego es crucial para proporcionar desafíos a los jugadores y mantener su interés a lo largo de la experiencia de juego.

Los videojuegos tienen sus raíces en la década de 1950, pero no fue hasta los años 70 que comenzaron a ganar popularidad con la aparición de juegos como "Pongz "Space Invaders". "Pong", desarrollado por Atari en 1972, es considerado uno de los primeros videojuegos comerciales y marcó el inicio de la industria del entretenimiento digital. Este período se caracterizó por juegos de mecánicas simples y gráficos básicos, diseñados para máquinas arcade. [84]

El honor del primer videojuego se discute, ya que algunos aseguran que fue “Tennis for two” que fue creado en una computadora analógca por el físico William Higinbotham en el Brookhaven National Laboratory en 1958. Otros dicen que fue un grupo de estudiantes en el MIT a principios de 1960 con un juego llamado “Spacewar”. [73] Las máquinas de monedas estaban dispersas en todos los lugares urbanos, y se les conocía como “penny arcades”. A partir de 1890 se encontraban en parques de atracciones, balnearios y circos ambulantes. Luego progresivamente se convirtieron en salas de videojuegos, esta transición tuvo lugar durante la década de 1970 y fue gradualmente. Las máquinas tragamonedas y las interfaces con “joysticks” y simuladores de conducción coexistieron hasta cierto en estas salas de juegos. En esta época se escuchaban fuertes críticas por parte de los padres y autoridades preocupados por la cordura y la moralidad de las personas más jóvenes. Así comenzaron a dispersar ideas malas sobre las salas de videojuegos, por lo que luego se comenzaron a construir las consolas de videojuegos para jugar en casa y se popularizaron para mantener a los niños en casa y fuera de las salas de videojuegos fuera de casa donde pasaban muchas horas al día. Los centros de juegos comenzaron a lanzar campañas para limpiar su nombre, y estas no tuvieron éxito. Ahora e dice que la continuidad de los arcades puede encontrarse en los reinos de los juegos de rol en internet, por las comunidades que se forman y la atmósfera.[73]

En 1977 Nintendo publicó el primer videojuego para jugar en casa, lo cual fue un gran éxito, dos años después se fundó “Capcom” en Japón una empresa conocida por publicar videojuegos famosos, como “Street Fighter” y “Resident Evil”. Luego en el año 1980 se publicó “Pac-man” que es el juego más popular de arcade de toda la historia de los videojuegos, más de 300 mil unidades se vendieron en todo el mundo. Nintendo publicó para su venta “The legend of Zelda” en 1987, y un año despues Square Soft una empresa también japonesa publicó “Final Fantasy” ambos son juegos de rol. En 1989 Nintendo sacó a la venta la consola llamada “Game boy” a nivel mundial y ese mismo año Nintendo y Atari fueron a la corte para pelear los derechos sobre “Tetris”. Luego de un tiempo ya que las consolas se habían comercializado, y los videojuegos se viralizaron a nivel mundial empezaron a haber demandas sobre la violencia de los videojuegos, por lo que en 1993 hubieron bastantes audiencias sobre este tema. Más adelante se creó la Asociación Interactiva de Softwares Digitales para poder responder a estas demandas. En ese mismo año Sony sacó la consola “PlayStation” a la venta en Japón y en Estados Unidos y ese mismo año Nintendo publicó a la venta “Virtual Boy” en Estados Unidos. Así como este último año continúo la competencia de estas marcas en todo el mundo hasta la actualidad, sacando nuevas consolas y juegos para ganar la audiencia y permanecer como número uno en el mercado. [84]

4.1.2. Evolución del diseño y programación de enemigos

El diseño y la programación de enemigos han evolucionado considerablemente desde los primeros juegos. Para 1970 los videojuegos estaban muy limitados a poca capacidad computacional de la era, así que era muy evidente predecir cómo iba a atacar el enemigo, y las interacciones eran mucho más simples, por ejemplo en “Space Invaders” donde los enemigos al atacar una vez podían eliminar en un par de golpes al jugador. Además, si se incluyen las gráficas el hardware limitaba a los creadores acerca de cómo diseñar a los enemigos, así que resultaban en simples íconos minimalistas o simples figuras [152]. En los años 80, títulos como "Pac-Man" introdujeron enemigos con patrones de movimiento predefinidos que agregaban un nivel básico de desafío y estrategia. Con el avance de la tecnología, especialmente durante los años 90, los videojuegos comenzaron a incorporar inteligencia artificial (IA) más sofisticada para los enemigos. Juegos como "Doom" y "Quake" mostraron enemigos que podían reaccionar dinámicamente a las acciones del jugador, utilizando técnicas como las máquinas de estados finitos para modelar comportamientos complejos. [19]

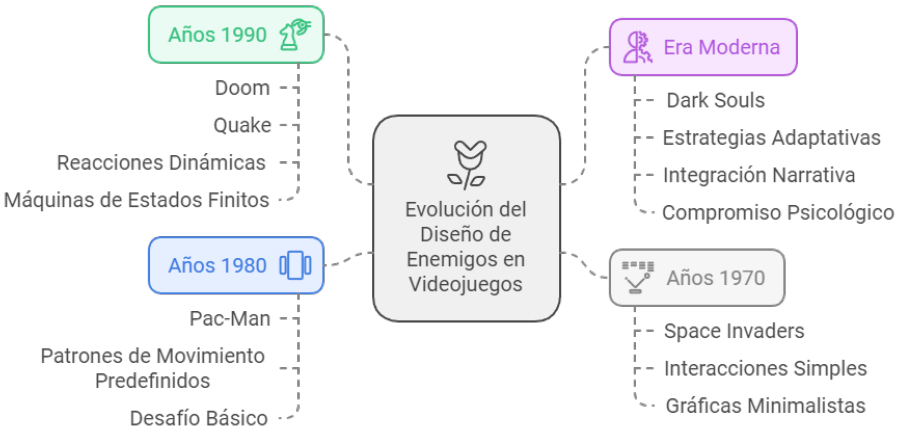

Figura 4.1: Evolución del diseño de enemigos en videojuegos.

Actualmente los videojuegos modernos han mejorado la inteligencia artificial que utilizan para los enemigos, ya que se pueden adaptar a las estrategias que utiliza el jugador para vencerlo. Esto se puede notar más en juegos como “Dark Souls” donde los patrones de ataque y estrategias son desafiantes y cambian según el patrón de ataque del jugador. Además los enemigos continúan con

una línea de narración lo que quiere decir que ya tienen una historia de trasfondo y motivaciones para envolver al jugador. Juegos como "BioShockïntegran enemigos que logran que el usuario se envuelva psicológica y emocionalmente en el juego [32].

4.1.3. Impacto de los avances tecnológicos en los videojuegos

El desarrollo de hardware más potente y herramientas de desarrollo avanzadas ha permitido una evolución continua en el diseño de videojuegos y la programación de enemigos. En la actualidad, los motores de juego como Unity y Unreal Engine facilitan la creación de enemigos con IA avanzada, utilizando técnicas como el aprendizaje profundo y los algoritmos de comportamiento adaptativo. Esto ha llevado a experiencias de juego más inmersivas y desafiantes, donde los enemigos pueden adaptarse y aprender de las estrategias del jugador en tiempo real. Los avances en la tecnología de renderizado gráfico también han mejorado significativamente la apariencia y el comportamiento de los enemigos, haciendo que sean más realistas y convincentes. [127]

En el pasado no fue tan sencillo, por ejemplo en las consolas de videojuegos de 1970 como "Magnavox Odyssey" y "Atari 2600" fue una gran revolución introducir cartuchos con juegos intercambiables para permitirles a los jugadores tener múltiples juegos. En el software de estos videojuegos los lenguajes de programación eran mucho más simples al igual que los motores. La inteligencia artificial como tal no estuvo presente, ya que solo existían las respuestas programables en los juegos como "Pong" donde la paleta simplemente reflejaba los movimientos del jugador. [84]

Figura 4.2: Evolución de los videojuegos a lo largo de las décadas.

En 1980 se logró mejorar la experiencia del jugador, con mejores gráficas y mejoras en el sonido por medio del hardware. El desarrollo de la tecnología de la programación logró la creación de juegos icónicos como "Super Mario Bros" ya que el diseño de herramientas para las gráficas mejoró exponencialmente. Por otro lado ya se podía ver la inteligencia artificial en juegos como "Pac-Man" donde los enemigos persiguen al jugador y ya son capaces de reaccionar a las acciones que toma el jugador [54].

El almacenamiento de la tecnología mejoró a CD-ROM lo que mejoró la capacidad en general y la fidelidad de la información. Así se pudo introducir Sony al mercado con el nuevo "PlayStation". Además en esta década de los 90 nacieron las gráficas en 3D lo que permitió crear ambientes totalmente en 3D como "Tom Raider" y "Doom". Las capacidades y comportamientos complejos de los enemigos avanzaron en esta década, y se podía apreciar en juegos como "Half-Life" [109]. En los 2000 se introdujo el almacenamiento en DVD mejorando las gráficas y complejidad de los juegos, esto surgió con las consolas: "PlayStation 2" de Sony y "Xbox". Para esta década también surgieron los teléfonos móviles capaces de tener videojuegos instalados. Las plataformas en linea surgieron con "Xbox Live" lo que permitió juegos multijugador. Además surgió el realismo en los juegos como "World of Warcraft" por las mejoras e interacciones dinámicas con ambientes con NPCs [82].

4.1.4. Términos de desarrollo de videojuegos

Gameplay: Es la forma característica en la que los jugadores interactúan con un juego. En esencia, es todo lo que involucra reglas, objetivos, limitaciones, desafíos, narrativa, y el resultado de las acciones de los jugadores. El *gameplay* es el corazón de cualquier experiencia de juego y provee una única mezcla de estrategias, desafíos e historia. [107]

NPC: Proveniente del término en inglés *Non-playable character* o también conocido como CPU, es cualquier personaje controlado por una computadora o cualquier personaje que no está bajo el control del jugador. Generalmente, es posible interactuar con estos personajes hablando con ellos, o incluso intercambiando recursos.

Figura 4.3: Captura de pantalla de la interacción entre Link y Beedle (NPC) en *Breath of the Wild*

Enemigos: En los videojuegos, los enemigos son todo aquel NPC que es hostil contra el jugador o que está presente para evitar que el jugador cumpla su objetivo. Estos generalmente están ligados al contexto e historia detrás del videojuego.

Figura 4.4: Evolución del enemigo "Goomba"de la saga de juegos de Mario Bros.

Jefe: Es un tipo de enemigo que suele ser más difícil que el enemigo normal. Tienden a estar relacionados al final de un nivel, progresión significativa dentro de la historia, o a la obtención de recursos más valiosos que los de un enemigo normal. También existe el término de "Jefe Final", el cual, como el nombre lo indica, el jefe que se enfrenta de último para terminar el juego y generalmente resolver el conflicto de la premisa del juego.

Figura 4.5: Batalla contra Asgore en *Undertale*

Rigging: En la animación, el *rigging* es el proceso por el cual un modelo plano, tridimensional o escultura digital se logra deformar para obtener movimiento para una animación. Para ello se crea una estructura de huesos y controles digitales para poder manipular la geometría de los objetos y personajes como si fuesen marionetas [100] .

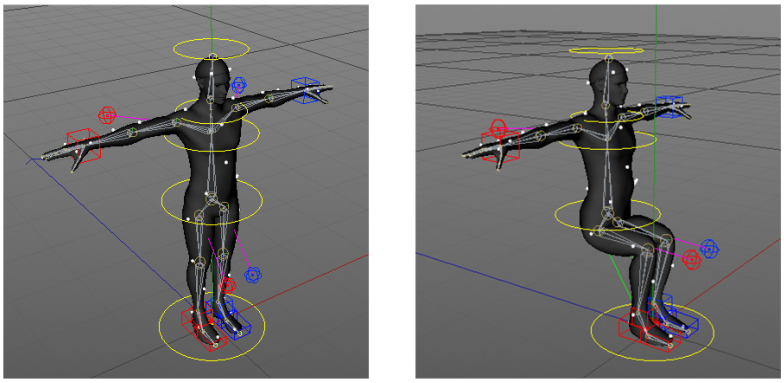

Figura 4.6: Personaje con *rig* completo colocado en pose T y pose T sentado

Play testing: Son sesiones de juego realizadas con usuarios determinados a realizar pruebas sobre un videojuego. En ellas se busca recibir retroalimentación sobre el juego, ya sea sobre algo en específico o sobre la experiencia general. También se busca hallar errores que se necesitan corregir antes de publicar el juego.

HUD: De las siglas de *Head-up Display*, y en español pantalla de visualización frontal, es el método por el cual visualmente se le muestra información a los jugadores. Entre los elementos más comunes que pueden mostrarse se encuentran: temporizadores, vidas, habilidades, armas y municiones, menús, progresión del juego, mini-mapas, información relevante al contexto, y cursores.

Figura 4.7: Ejemplo de HUD en el juego Fornite

Moodboard: Es una colección de imágenes, videos, y otros elementos gráficos recopilados con el fin de comunicar una dirección visual, reflejar un ánimo, o reflejar un estilo. [45]

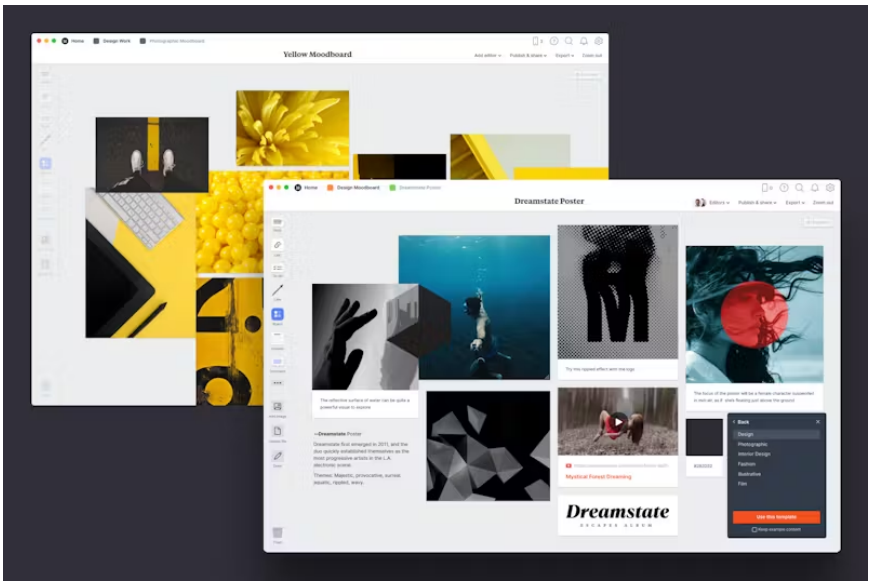

Figura 4.8: Ejemplos de Moodboars realizados en la aplicación Milanote

4.2. Importancia de los enemigos en los videojuegos

4.2.1. Rol de los enemigos en la jugabilidad

Los enemigos en los videojuegos desempeñan un papel crucial en la creación de desafíos, promoviendo la participación activa del jugador y determinando la dinámica que se tendrá a lo largo del juego. Los enemigos no solo sirven como obstáculos que deben ser superados, sino que también son fundamentales para el diseño de niveles, narrativa y la progresión del juego. Según Salen y Zimmerman [122], "los enemigos actúan como catalizadores que impulsan al jugador a utilizar habilidades aprendidas y a explorar mecánicas de juego de maneras más complejas y estratégicas"(p. 315). Este diseño intencional obliga a que los jugadores se adapten y mejoren continuamente, manteniendo la experiencia de juego interesante gracias a desafíos de la habilidad e ingenio del jugador. Además, los enemigos pueden variar en dependiendo de su comportamiento, dificultad y papel en la narrativa, lo que enriquece la experiencia de juego. Esto es especialmente evidente en los juegos con inteligencia artificial avanzada, donde los enemigos pueden adaptarse a las acciones del jugador, creando una experiencia de juego más dinámica y personalizada [113]. La presencia de enemigos bien diseñados puede transformar un juego simple en una experiencia memorable, como lo observan Linderoth y Bennerstedt [89]: "los enemigos no solo definen el tono y la atmósfera del juego, sino que también pueden influir directamente en la inmersión y la percepción de logro del jugador"(p. 246).

Por ejemplo, en *Bloons Tower Defense*, los enemigos son simples globos denominados como *Bloons*, con un comportamiento sencillo y un simple objetivo: pasar las defensas del jugador para quitarle corazones. Pero, a pesar de esa simplicidad, han demostrado ser un reto prominente para muchos jugadores, ya que sus números aumentan por cada ronda que se sobrevive, desafiando la capacidad del jugador para gestionar recursos e idearse estrategias de defensa efectivas. Asimismo, en *Elden Ring*, se tiene una complejidad mucho mayor en los enemigos ya que algunos de estos tienen comportamientos impredecibles y ataques variados que requieren una capacidad constante de adaptarse por parte del jugador, sin mencionar que deben conocer el entorno [24]. Por eso mismo, los enemigos en los videojuegos son elementos esenciales que afectan directamente la experiencia, dificultad y satisfacción del jugador. Su diseño y comportamiento no solo aportan desafíos, sino que también permiten una mayor profundidad y complejidad en la narrativa e interacción del jugador

con el entorno del juego.

4.2.2. Influencia en la narrativa y experiencia del jugador

La principal característica que distingue a los videojuegos de otros medios narrativos es su interactividad. Esta permite que los jugadores tengan una experiencia inmersiva en la historia, y además pueden influir en el desarrollo. Según Gemma López Canicio, la interactividad transforma la narrativa en un proceso comunicativo bidireccional, donde el jugador puede actuar como co-productor de la historia[76]. Esto genera una experiencia narrativa más rica y personalizada que permite a los jugadores experimentar sentimientos a través de la interactividad con el juego. Los videojuegos tienden a abandonar las estructuras narrativas lineales típicas del cine y la literatura. En lugar de seguir un camino predefinido, permiten a los jugadores explorar mundos abiertos y tomar decisiones que afectan el curso del relato. Esto se observa en juegos como RiME, donde el entorno no solo sirve como escenario, sino que también refleja los estados psicológicos de los personajes.

Los videojuegos crean espacios que cuentan historias a través de su diseño y mecánicas. Por ejemplo, elementos como cartas o correos electrónicos dentro del juego pueden ofrecer contexto adicional sin interrumpir la acción principal, permitiendo que los jugadores completen la narrativa a su propio ritmo[98]. Esta capacidad de inmersión se traduce en una conexión más profunda entre el jugador y el mundo del juego. La integración de mecánicas de juego con la narrativa también es crucial. Un estudio sobre videojuegos móviles sugiere que sumar elementos narrativos a las mecánicas aumenta el desafío creativo del jugador, haciendo que las decisiones tengan un impacto significativo en la historia[114]. Esto contrasta con la narrativa tradicional, donde el desarrollo es fijo y no puede ser alterado por el usuario.

La narrativa en los videojuegos no solo se trata de contar una historia; también busca construir empatía entre el jugador y los personajes. Al permitir que los jugadores tomen decisiones críticas, se fomenta una conexión emocional más fuerte. Esto es evidente en juegos donde las elecciones morales afectan el desenlace, lo que puede llevar a experiencias profundamente personales[18]. La inmersión generada por esta narrativa interactiva estimula una participación activa del jugador, lo cual es fundamental para mantener su interés. Los entornos digitales permiten experimentar resultados significativos a partir de acciones dentro del juego, reforzando así la ilusión de realidad y aumentando el placer asociado con la interacción[18]. La influencia de la narrativa en los videojuegos se manifiesta a través de su interactividad, la construcción dinámica de mundos narrativos y el impacto emocional en los jugadores. Estos elementos combinados crean experiencias únicas que no solo entretienen, sino que también invitan a una reflexión más profunda sobre las historias que se cuentan.

4.2.3. Estudios sobre diseño de enemigos

El diseño de enemigos en los videojuegos es un área clave que impacta directamente en la jugabilidad y en la experiencia del jugador. Diversos estudios han explorado este tema desde diferentes perspectivas, ofreciendo un marco conceptual importante para comprender el impacto de los enemigos en la mecánica del juego.

Uno de los trabajos relevantes en esta área es el de UPC [115], que analiza las mecánicas y comportamientos de los enemigos en videojuegos del género Roguelite. Este estudio clasifica diferentes tipos de ataques, movimientos y comportamientos, proponiendo un esquema que facilita el diseño de enemigos, y además explora la percepción de dificultad a través de sesiones de playtesting. Por otro lado, un estudio sobre la generación de comportamientos de enemigos en videojuegos 2D [112] propone un sistema de inteligencia artificial que permite a los enemigos reaccionar de manera dinámica y desafiante. Este tipo de sistemas mejora la interacción entre el jugador y los enemigos, lo que resulta en una experiencia más envolvente y retadora.

Adicionalmente, en el ámbito de los videojuegos de lucha, el trabajo de ULPGC [77] examina la implementación de inteligencia artificial para los enemigos en un videojuego 2D. Se destaca cómo los enemigos interactúan con el jugador, lo cual es crucial para mantener el equilibrio entre la dificultad y el entretenimiento. Otro enfoque es el de los juegos de rol (RPG), donde el diseño de los enemigos está ligado a decisiones tácticas por parte del jugador. El estudio de [67] detalla cómo los enemigos poseen mecánicas únicas que añaden profundidad estratégica al juego, exigiendo al jugador analizar y adaptar sus movimientos.

Finalmente, una revisión sistemática sobre videojuegos [52] subraya la importancia del diseño de enemigos como una parte integral del desarrollo técnico de los juegos, destacando su influencia en la complejidad y calidad general de la experiencia del jugador. Estos estudios proporcionan una visión amplia sobre el diseño de enemigos en videojuegos, subrayando su relevancia tanto en aspectos técnicos como en la construcción de experiencias inmersivas para los jugadores.

4.3. Teorías y técnicas de inteligencia artificial

4.3.1. Redes neuronales y aprendizaje profundo

El aprendizaje profundo, es una subcategoría del aprendizaje automático, que ha logrado revolucionar la inteligencia artificial (IA) mediante el uso de redes neuronales profundas. Estas redes son capaces de aprender y realizar tareas complejas a partir de grandes volúmenes de datos, imitando el funcionamiento del cerebro humano. Las redes neuronales son modelos computacionales inspirados en la estructura del cerebro humano. Están compuestas por neuronas artificiales organizadas en capas: la capa de entrada recibe los datos iniciales, las capas ocultas procesan la información a través de múltiples transformaciones y por último la capa de salida que genera el resultado final. Cada neurona en una red realiza cálculos simples y transmite su salida a las neuronas de la siguiente capa. Este proceso se repite hasta que se produce la salida final del modelo[17, 40].

El aprendizaje profundo se basa en redes neuronales con múltiples capas ocultas, lo que permite la extracción automática de características complejas a partir de datos sin procesar. Se diferencia del aprendizaje automático tradicional, ya que en este se requiere la selección manual de características, el aprendizaje profundo puede aprender directamente de los datos[31, 129, 17]. Es posible distinguir varias arquitecturas de redes neuronales utilizadas en el aprendizaje profundo:

1. **Redes Neuronales Convolucionales (CNN)**, que son efectivas para tareas de visión por computadora, como el reconocimiento y la clasificación de imágenes [53].
2. **Redes Neuronales Recurrentes (RNN)**, empleadas para el procesamiento de secuencias, como texto o audio, lo que permite modelar dependencias temporales [53].
3. **Redes Generativas Antagónicas (GAN)**, compuestas por dos redes que compiten entre sí, utilizadas para generar nuevos datos sintéticos a partir de un conjunto de entrenamiento [53].

Entrenar modelos profundos requiere de un hardware avanzado y grandes cantidades de datos etiquetados[31, 129]. Los modelos a menudo son considerados çajas negras", ya que es un poco difícil de entender cómo toman decisiones específicas[17, 40]. Además existe el riesgo de que un modelo aprenda demasiado bien los datos de entrenamiento, lo que puede perjudicar su rendimiento en datos no vistos[17]. Las redes neuronales y el aprendizaje profundo han transformado la forma en que las máquinas procesan información y aprenden. Con aplicaciones que abarcan desde la atención médica hasta la conducción autónoma, su impacto es significativo. Sin embargo, es crucial abordar los desafíos asociados con su implementación para maximizar su potencial en el futuro.

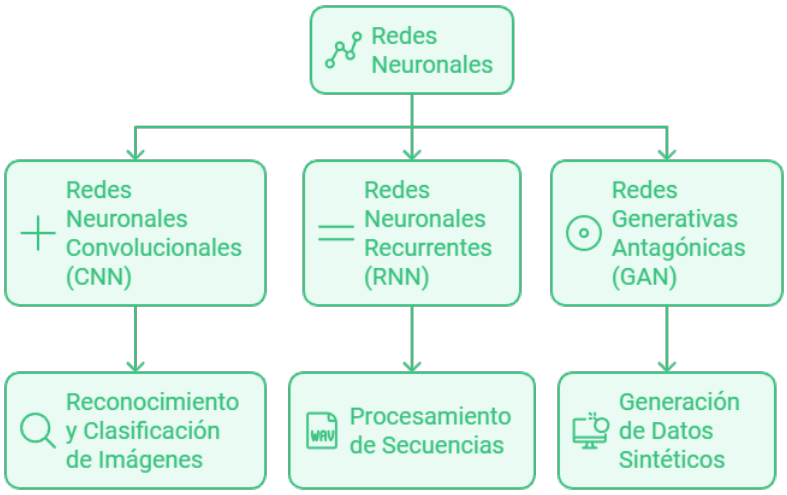

Figura 4.9: Mapa conceptual de los tipos de redes neuronales.

4.3.2. Comparación de técnicas y su aplicabilidad

Las redes neuronales han revolucionado el campo del desarrollo de videojuegos al ofrecer herramientas avanzadas para la inteligencia artificial y la generación de contenido automatizado. A continuación, se comparan algunas de las principales técnicas de redes neuronales y su aplicabilidad en diferentes áreas del desarrollo de videojuegos.

Redes Neuronales Convolucionales (CNN)

Las Redes Neuronales Convolucionales (CNN) son especialmente adecuadas para el análisis de datos visuales debido a su capacidad para aprender patrones jerárquicos. En el contexto de los videojuegos, las CNN son ampliamente utilizadas en tareas como el reconocimiento de imágenes y la mejora de gráficos. Asimismo, se han implementado en agentes inteligentes que utilizan la entrada de píxeles del juego como fuente de información para la toma de decisiones, como es el caso de algunos proyectos en juegos de disparos en primera persona, donde los agentes aprenden a jugar utilizando solo imágenes del entorno del juego [53].

Redes Neuronales Recurrentes (RNN) y LSTM

Las Redes Neuronales Recurrentes (RNN) son útiles para procesar secuencias de datos, aunque presentan limitaciones como la pérdida de información en secuencias largas debido al desvanecimiento del gradiente. Las Long Short-Term Memory (LSTM) son una variante de las RNN que solventan este problema, permitiendo a los modelos mantener información a largo plazo. En videojuegos, las RNN y LSTM son utilizadas para modelar comportamientos dinámicos en personajes no jugables (NPCs), lo que resulta en una interacción más realista y adaptativa con los jugadores. Además, han sido aplicadas para mejorar la toma de decisiones en tiempo real, como se ha visto en algunos proyectos desarrollados en motores como Unity [105].

Aprendizaje por Refuerzo Profundo (Deep Reinforcement Learning)

El Aprendizaje por Refuerzo Profundo combina redes neuronales con algoritmos de aprendizaje por refuerzo, permitiendo que los agentes en los videojuegos aprendan a través de la exploración y explotación de su entorno. Este enfoque ha sido clave en la creación de agentes competitivos capaces de superar a jugadores humanos en juegos estratégicos, como lo demuestran los avances de AlphaGo y AlphaZero. Además, los agentes que utilizan aprendizaje por refuerzo son capaces de adaptarse a distintos estilos de juego, mejorando significativamente la experiencia del usuario al ajustar su comportamiento a las estrategias del jugador [53, 44].

Generación procedural de contenido

Otra aplicación relevante de las redes neuronales en los videojuegos es la generación procedural de contenido, que permite la creación automática de niveles, personajes o escenarios. Esta técnica reduce el tiempo de desarrollo y ofrece experiencias únicas y personalizadas a los jugadores. Las redes neuronales pueden aprender patrones a partir de datos existentes para generar mapas o entornos dinámicos, lo que aumenta la diversidad y la adaptabilidad del juego a las preferencias individuales del usuario [53, 143].

4.4. Contexto histórico sobre el diseño de interfaces y de experiencia

Nasir Uddin (2023) [141], cofundador de Musimundo (Agencia de diseño UX), al relatar brevemente la historia y evolución del diseño UX/UI expresa que para crear experiencias de usuario centradas en los humanos es necesario primero comprender cómo es que funcionan los humanos y su psicología. En su artículo concluye que conforme la tecnología evoluciona con nosotros, los humanos seguiremos produciendo nuevas metodologías y generando nuevos conocimientos con el objetivo de la mejora de la experiencia humana.

Uddin habla sobre historia y nota que desde los inicios de la humanidad se han buscado formas de vivir de manera más armoniosa. Una de las prácticas más reconocidas y antiguas de la historia se desarrolló en China por el 4,00 a.C, conocida como Feng Shui. A pesar de que esta se basa en el *qi* (energías) y busca su equilibrio, su objetivo final y modo de uso general es mejorar los ambientes en los que las personas se encuentran. Para ello se toma en cuenta el balance entre objetos, la funcionalidad de los mismos y el placer que causan en las personas.

Otra práctica relevante de analizar es una que Nicolas Marmaras (1997) [104] describe en un documento defendiendo el diseño ergonómico en la Antigua Grecia, hace 25 siglos. Es importante saber que el término "ergonómico" es uno reciente que comenzó a ser utilizado hasta el siglo 20, pero incluso en la antigüedad se pueden ver la aplicación de sus bases y principios. En la literatura de la época se pueden encontrar varias referencias a este tipo de diseños de artefactos y espacios. También puede ser reconocido en los artefactos y ejemplares de arquitectura que han sobrevivido hasta la fecha y demostrado un diseño centrado en su uso humano. Ejemplos que Marmaras menciona de este tipo de enfoque en el diseño son los utensilios del día a día y las inteligentes soluciones que fueron adoptadas en su diseño. Además de lo duraderas que han demostrado ser, manteniendo su forma hasta hoy en día.

Siglos más tarde sobresale un nombre en la literatura del diseño. El nombre, Henry Dreyfuss, un pionero del diseño industrial. El libro "Las medidas del Hombre y la Mujer" publicado por la asociación Henry Dreyfuss en 1993 es un compendio de treinta años de investigación y recopilación de datos vitales que tienen el fin de asistir al diseño de productos y ambientes. Entre estos datos

se encuentran estadísticas y medidas de productos y espacios. Ejemplares de estos son medidas de hombres, mujeres, niños, estaciones de computadoras, escaleras, dormitorios y mucho más. Para cada una de ellas brinda recomendaciones y medidas estándar para que los diseñadores puedan tomarlas en cuenta y realizar diseños enfocados en la interacción del humano con cada uno de estos objetos.

En el contexto de la tecnología, en los años 70 surgieron las primeras computadoras personales. y junto a ellas surgieron preguntas de cómo interactuar con las computadoras. De esta necesidad surgieron las interfaces de usuario gráficas, o GUI (*Graphical User Interface*) que permitieron a los usuarios interactuar con la computadora haciendo uso de ratones y teclados. Fue hasta en 1984 que Donald Norman, científico cognitivo de Apple, acuñó el término de "Experiencia de Usuario" y se tomó un diseño centrado en el usuario.

Los juegos han sido parte esencial de nuestra vida y humanidad. Es una de las mejores herramientas que los humanos utilizan para aprender en una edad temprana. Por lo mismo, es natural que los juegos evolucionen junto a la tecnología. A partir de ello se crearon los videojuegos, que son cualquier tipo de juego que se realiza en un medio virtual o digital. Estos vienen desde que los computadores comenzaron a tener GUI, siendo "Tenis para Dos" el primer videojuego realizado solo con el fin de entretener. Desde ese momento han evolucionado en gráficos, narrativa, funcionalidades y todos sus aspectos.

4.5. Prácticas UX/UI

Primero es necesario comprender que los términos "Experiencia de Usuario" e "Interfaz de Usuario" no son términos mutuamente excluyentes, por el contrario, estos son términos complementarios e imposibles de separar en su totalidad uno del otro. Aunque tienen enfoques y acercamientos distintos, la realidad es que siempre que el usuario observe un elemento de diseño, este causará una reacción en el usuario que determinará cuál es su experiencia en general con la plataforma. (Bank, Cris, Cao, Jerry, s.f.) [48]

Con respecto al diseño de interfaces, este busca construir el puente de comunicación entre una persona y un computador. Se enfoca principalmente en todos los elementos que conforman la estética de la interfaz, su forma, orden, jerarquía y se basa en 7 principios.

Conocimiento del mundo real: Basarse en modelos análogos, tal como un botón.

Dejar al usuario cometer errores: Los errores son muy comunes, por lo que es importante y necesario implementar políticas de recuperación.

Convenciones y comportamientos aprendidos: Acciones con las que los usuarios ya se encuentran familiarizados, como deslizar un dedo en una pantalla para cambiar de pantallas o regresar.

Causa y efecto: Es necesario siempre brindar algún tipo de *feedback* al usuario para que esté consciente de que las cosas están actuando como deberían actuar

Consistencia: Los usuarios deben estar seguros de que las acciones que hacen y lo que han aprendido aún funciona y no cambia arbitrariamente.

Agilidad: Presentar al usuario soluciones rápidas y fáciles con la menor cantidad de pasos para todas sus necesidades. Mientras más importante y necesaria, muchos menos pasos.

Satisfacción: Brindarle al usuario pequeñas sensaciones y resultados de gratificación que le invitan a seguir utilizando la aplicación.

Sin embargo, uno de los principios más importantes que es necesario tomar siempre en cuenta al momento de diseñar una interfaz es "Un mismo tamaño no les queda a todos", el cual hace alusión

a que las interfaces pueden ser muy variadas y las decisiones que se toman para ellas pueden variar mucho dependiendo de los usuarios destinatarios, metas y contenido.

Una decisión que divide a las interfaces es si se diseña utilizando plantillas o si se diseña basado en el contenido. Por un lado, el uso de una plantilla es indiferente al tipo de contenido que muestra; un ejemplo es una grilla, la cual es muy genérica y con ella se pueden mostrar muchos tipos de contenido. Al usar una plantilla es fácil de implementar la interfaz y perfecto para contenido repetitivo, como objetos en una lista, pero puede resultar muy aburrido para el usuario. Por el contrario, cuando se diseña basándose en el contenido, los diseños pueden resultar atractivos para el usuario gracias a su variación y originalidad, pero no solo es más difícil de implementar, sino que puede requerir de mayor esfuerzo por parte del usuario para comprender su uso si difiere mucho de las convenciones actuales.

Por otro lado, para un análisis UX, existen varios acercamientos que se pueden utilizar, pero uno de los más usados y relevantes para el proyecto a trabajar es uno que se centra en el usuario final de los productos. Para ello se crean personas imaginarias que representan a los usuarios ideales que utilizarán la aplicación, se definen cuáles son sus necesidades, expectativas, deseos, motivaciones, aspiraciones, preocupaciones y pensamientos profundos. Para ello se puede hacer uso de herramientas y diagramas como mapas de empatía. A partir de toda la información sobre estos usuarios ideales se toman decisiones sobre diseño para acoplarse a sus intereses y las posibles experiencias que pueden crear al interactuar con la aplicación.

No solo es necesario tomar en consideración todos los aspectos internos y psicológicos de los usuarios finales ideales, sino también es necesario tomar en cuenta el contexto global en el que se encuentran. Haciendo énfasis en aspectos culturales y socioeconómicos de los usuarios. Pero más importante que todas estas consideraciones y herramientas, es de suma importancia obtener constante retroalimentación por parte de usuarios que representan al público objetivo y conforme a sus críticas, realizar cambios necesarios y relevantes para su satisfacción.

4.6. Investigación de estudios sobre la experiencia del usuario

4.6.1. Ley de Hick-Hyman

La **Ley de Hick-Hyman**, también conocida simplemente como Ley de Hick, es un principio de la psicología cognitiva que describe la relación entre el número de opciones disponibles y el tiempo que una persona tarda en tomar una decisión. Formulada en 1952 por los psicólogos William Edmund Hick y Ray Hyman, esta ley establece que el tiempo de reacción aumenta logarítmicamente a medida que se incrementa el número de alternativas. En términos prácticos, esto significa que cuantas más opciones se presenten a un individuo, más tiempo necesitará para decidirse por una de ellas [149].

La fórmula matemática que representa esta relación es:

$$T = b \cdot \log_2(n + 1)$$

Donde:

- T es el tiempo de reacción.
- b es una constante que depende de las condiciones específicas del experimento.
- n es el número de opciones disponibles.

Esta ley tiene aplicaciones significativas en campos como la interacción humano-computadora y el diseño de interfaces de usuario, donde se busca minimizar el tiempo de decisión del usuario al reducir la cantidad y complejidad de las opciones presentadas [78].

4.6.2. Estudios de Jakob Nielsen

Jakob Nielsen es un destacado experto en usabilidad y experiencia de usuario. Con un doctorado en interacción humano-computadora por la Universidad Técnica de Dinamarca, Nielsen ha dedicado gran parte de su carrera al estudio y mejora de la usabilidad en interfaces digitales. En 1998, cofundó el *Nielsen Norman Group*, una firma de consultoría especializada en usabilidad y diseño centrado en el usuario [148].

Entre sus contribuciones más influyentes se encuentran los **10 principios heurísticos de usabilidad**, formulados en 1994, que sirven como directrices para diseñar interfaces más intuitivas y eficientes [128]. Estos principios incluyen:

1. **Visibilidad del estado del sistema**: Mantener al usuario informado sobre lo que está ocurriendo en el sistema.
2. **Relación entre el sistema y el mundo real**: Utilizar un lenguaje y conceptos familiares para el usuario.
3. **Control y libertad del usuario**: Permitir al usuario deshacer y rehacer acciones fácilmente.
4. **Consistencia y estándares**: Seguir convenciones y estándares reconocidos.
5. **Prevención de errores**: Diseñar el sistema para minimizar la posibilidad de errores.
6. **Reconocer antes que recordar**: Reducir la carga de memoria del usuario mostrando opciones y acciones visibles.
7. **Flexibilidad y eficiencia de uso**: Ofrecer atajos y opciones para usuarios avanzados.
8. **Estética y diseño minimalista**: Evitar la sobrecarga de información irrelevante.
9. **Ayudar a los usuarios a reconocer, diagnosticar y recuperarse de errores**: Proporcionar mensajes de error claros y soluciones.
10. **Ayuda y documentación**: Ofrecer asistencia y documentación accesible cuando sea necesario.

4.7. Metodología de diseño centrado en el usuario

El **Diseño Centrado en el Usuario** (DCU), conocido en inglés como *User-Centered Design* (UCD), es una metodología de diseño que coloca al usuario final en el centro de todo el proceso de desarrollo de un producto o servicio. Su objetivo principal es crear soluciones que satisfagan las necesidades, expectativas y limitaciones de los usuarios, garantizando una experiencia de uso óptima y efectiva.

4.7.1. Principios fundamentales del DCU

- **Enfoque en el usuario**: Comprender profundamente a los usuarios, incluyendo sus características, necesidades y contextos de uso.

- **Participación activa del usuario**: Involucrar a los usuarios en todas las etapas del diseño, desde la investigación inicial hasta las pruebas finales.
- **Diseño iterativo**: Implementar un ciclo continuo de diseño, prototipado, evaluación y refinamiento basado en la retroalimentación de los usuarios.
- **Consideración del contexto de uso**: Tener en cuenta el entorno físico, social y técnico en el que el producto será utilizado.

4.7.2. Etapas del proceso de DCU

1. **Investigación y análisis**
   - **Identificación de usuarios y contextos**: Determinar quiénes son los usuarios y en qué situaciones utilizarán el producto.
   - **Recopilación de requisitos**: Entender las necesidades y expectativas de los usuarios mediante entrevistas, encuestas y observaciones.
2. **Diseño**
   - **Generación de ideas**: Desarrollar múltiples soluciones potenciales que respondan a los requisitos identificados.
   - **Prototipado**: Crear representaciones tangibles de las soluciones propuestas, que pueden variar desde bocetos hasta prototipos funcionales.
3. **Evaluación**
   - **Pruebas de usabilidad**: Evaluar los prototipos con usuarios reales para identificar problemas y áreas de mejora.
   - **Análisis de feedback**: Recopilar y analizar la retroalimentación para refinar y mejorar el diseño.
4. **Implementación**
   - **Desarrollo del producto final**: Construir el producto basándose en el diseño validado.
   - **Monitoreo post-lanzamiento**: Supervisar el uso del producto para detectar posibles mejoras futuras.

4.7.3. Beneficios del DCU

- **Mejora de la usabilidad**: Los productos diseñados bajo esta metodología suelen ser más intuitivos y fáciles de usar.
- **Aumento de la satisfacción del usuario**: Al atender las necesidades reales de los usuarios, se incrementa su satisfacción y lealtad.
- **Reducción de costos a largo plazo**: Detectar y corregir problemas en etapas tempranas del diseño evita costosas modificaciones posteriores.

4.7.4. Principios básicos de la teoría del color

- **Círculo cromático:** Representa la relación entre los colores primarios, secundarios y terciarios, facilitando la creación de paletas armoniosas.
- **Armonías cromáticas:** Combinaciones de colores que resultan visualmente agradables, como las armonías complementarias, análogas y triádicas.
- **Propiedades del color:** Incluyen el tono (hue), la saturación y el valor (luminosidad), que determinan la apariencia y el impacto de un color en la interfaz [88].

4.7.5. Psicología del color en el diseño

La psicología del color estudia cómo los diferentes tonos afectan las emociones y comportamientos humanos. En el diseño UX/UI, es fundamental comprender estas asociaciones para seleccionar colores que refuercen el mensaje y la funcionalidad del producto. Por ejemplo, el azul suele asociarse con confianza y seguridad, mientras que el rojo puede evocar urgencia o pasión [42].

4.7.6. Aplicaciones de la teoría del color en UI/UX

Para aplicar eficazmente la teoría del color en el diseño de interfaces:

- **Definir una paleta de colores consistente:** Seleccionar una gama de colores que refleje la identidad de la marca y sea coherente en toda la interfaz.
- **Utilizar el contraste para la legibilidad:** Asegurar que haya suficiente contraste entre el texto y el fondo para facilitar la lectura.
- **Considerar la accesibilidad:** Evitar combinaciones de colores que puedan ser problemáticas para personas con daltonismo u otras deficiencias visuales [36].
- **Probar con usuarios:** Realizar pruebas de usabilidad para evaluar cómo los usuarios perciben y reaccionan a las elecciones de color en la interfaz.

4.7.7. Herramientas para selección de colores

Existen diversas herramientas que asisten a los diseñadores en la creación de paletas de colores efectivas:

- **Adobe color:** Permite generar esquemas de color basados en diferentes reglas de armonía.
- **Coolors:** Facilita la creación y exploración de paletas de colores.
- **Contrast Checker:** Ayuda a verificar el contraste entre colores para garantizar la accesibilidad.

4.8. Prácticas GX

El término GX proviene de las palabras "*Gameplay Experience*z su prácticas hacen alusión a enfoques similares a los de UI y UX, pero en un contexto de videojuegos. Por ello, lo que se busca

es realizar un *Gameplay design* (Diseño de juego) el cual engloba todas las decisiones tomadas para determinar aspectos como las funciones dentro del juego, la tecnología a utilizar, controles, y la implementación de la historia.

Para el diseño de un videjuego se pueden tomar distintos acercamientos. De entre ellos, uno muy conocido y popular es el *player-centric approach* o un acercamiento centrado en el jugador. Este acercamiento, al igual que el acercamiento centrado en el usuario final del UX, busca tomar las mejores decisiones al momento de realizar el diseño de un videojuego tomando en cuenta todos los aspectos internos y externos de la psicología y contexto del usuario. Las prácticas GX prestan conceptos de las prácticas de UX, basándose 7 pilares similares, los cuales están descritos en el libro "La mente del jugador" escrito por Cecilia Hodent (2018) [71] :

Señales de retroalimentación, el usuario debe saber que las cosas funcionan correctamente.

- **Claridad**, de instrucciones y de lo que está ocurriendo en el videojuego.
- **La forma sigue a la función**, es importante representar de manera correcta las funciones de los elementos del videojuego y que estos tengan una forma clara.
- **Consistencia**, para que los jugadores estén seguros de qué hacen ciertos elementos del juego y ciertas acciones y que siempre que sean necesitados funcionen de la misma manera.
- **Carga de trabajo mínima**, para que los juegos puedan correr sin mayor problema en los dispositivos de los jugadores.
- **Prevención de errores**, se le debe de permitir al usuario fallar y cometer errores, por ello es necesario implementar estrategias de recuperación.
- **flexibilidad**, para permitir que el jugador cree sus propias experiencias.

4.9. Diseño de niveles

Para el diseño de niveles es recomendable hacer uso de un documento especial específico para la definición de los niveles que conlleva al videojuego. Este documento se titula *Level Design Document*, existen una infinidad de plantillas que abarcan desde ser de lo más general hasta lo más específico [34]. Cada videojuego requerirá de distintos elementos a tener en cuenta, por lo que se recomienda acoplar la plantilla al tipo y género de videojuego que se está desarrollando. Entre los aspectos a tomar en cuenta para un nivel se pueden mencionar; El segmento de la historia que conlleva el nivel, los objetivos dentro del nivel, el tiempo estimado para completar el nivel, los tipos de enemigos a enfrentar y sus cantidades, el mapa a utilizar, las posiciones del jugador, enemigos, objetos y objetivos, música de fondo, poderes, habilidades, peligros, mecánicas, entre muchos otros.

4.10. Narrativa

Estrechamente relacionado al diseño y la escritura del contexto del personaje, si un juego cuenta una historia, debe estar diseñada esta esté diseñada a medida para el jugador ideal. Como Mikolás Márton (2015) expresa en su tesis "Videojuegos como un género narrativo", la estructura de *storytelling* de un videojuego, aplicándole la topología de Gennette, demuestra que los videojuegos pueden pertenecer a cualquier género narrativo. Uno de sus principios, la interacción es la que lo diferencia de otras formas narrativas. Las propiedades de acción permiten que la narrativa de personajes y espacios pueda ser modificada y expandida. Se le pueden agregar distintos finales a un juego, o incluso

si el final es el mismo, las experiencias en el camino cambian. Con la presencia de videojuegos orientados al *storytelling* se puede llegar a reinterpretar la literatura lineal combinada con la exploración de espacios narrativos, creando así una meta-narrativa al reflejar las contradicciones en la narrativa del videojuego.

4.11. Storyboarding

El *storyboarding*, o guion gráfico en español, es un proceso en el cual se realizan una serie de bosquejos cronológicos con el objetivo de establecer un guion visual [59]. Este puede ser desarrollado en papel o haciendo uso de herramientas digitales. Por lo mismo, este es un proceso que no requiere muchos recursos, por lo que es ideal para hacer correcciones antes de enviar el guion a producción. Esto permite ahorrar tiempo y recursos al evitar realizar trabajo que luego de alguna corrección sería descartado. Los *storyboards* no tienen un formato en específico sino que son muy flexibles para el tipo de trabajo que se está realizando, pero entre las cosas que se pueden considerar al crear un *storyboard* son las siguientes:

- Continuidad: Las ilustraciones o bosquejos se encuentran de manera cronológica para tener una mejor comprensión del orden de los eventos.
- Diálogos: Cualquier conversación entre los personajes de la escena o el narrador.
- Sonido: Todos los efectos de sonidos que se utilizarán durante la secuencia y en qué momento sonarán.
- Música: Se puede indicar si se tiene música de fondo.
- Composición: De suma importancia, se indica el ángulo de la cámara real o ficticia, con el objetivo de dar cierto sentimiento específico a la escena.
- Mood: Se pueden indicar el *mood* general de la escena y cómo obtenerlo.

4.12. Introducción al modelado 3D

4.12.1. Definición del modelado 3D

El modelado 3D es un proceso de creación de una representación tridimensional para un objeto, un personaje o de un entorno en una lógica especializada. En este proceso, los modelos 3D se construyen con geometrías elevadas y llenos de detalles, en donde se conforman por caras, aristas y vértices que forman una red o también conocida en inglés como *wireframe*, representando la superficie de los objetos en un entorno 3D. Es, por tanto, uno de los elementos clave para crear mundos virtuales realistas y detallados, especialmente para industrias como el entretenimiento y los juegos, que requieren una precisión visual que les permita, a su vez, crear experiencias inmersivas. El dominio de esto no se limita al desarrollo de juegos, también tiene muchas otras áreas donde se aplican[110]:

- Medios y entretenimiento, Figura4.10: Está infiltrado en las industrias del cine, la televisión, lo que permite la creación de efectos visuales difíciles de realizar, personajes animados y entornos muy detallados. Esto permite conseguir un realismo visual especialmente para categorías de ficción[20].

- Arquitectura y concepción inmobiliaria, Figura 4.10: En este dominio, los arquitectos y los conceptualizadores adoptan la modelización 3D para visualizar los edificios, los interiores y los paisajes antes de la construcción. Esto permite comunicar mejor a los clientes y a los inversores y mejorar la concepción para responder a las exigencias del proyecto[20].

- Concepción y fabricación de productos, Figura 4.10: Aquí es una herramienta potente para la realización de prototipos y la visualización de productos. Una técnica que ofrece a los conceptualizadores la posibilidad de definir sus ideas gracias a un cierto número de interacciones antes del producto final; De esta manera, aún no está comprometido en la creación de modelos físicos finales[20].

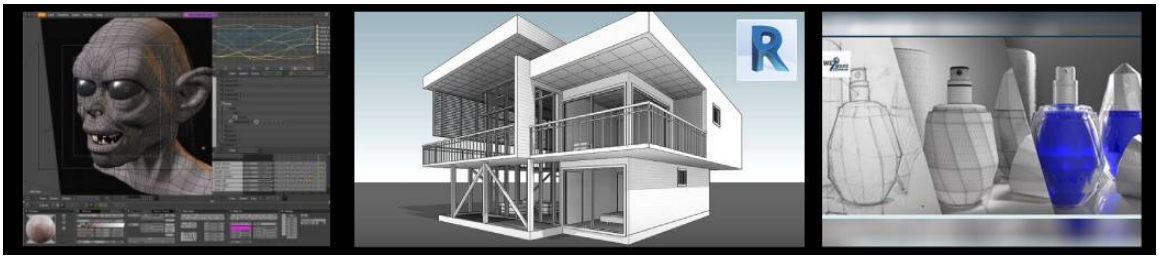

Figura 4.10: Entretenimiento, arquitectura, productos

- Para las industrias automovilística y aeroespacial, Figura 4.11: La modelización 3D se aplica desde la concepción de los vehículos y la publicidad, en la medida que es fácil de simular todos los componentes en detalle y representar los vehículos en la publicidad. Existe gran número de publicidades con automóviles que utilizan imágenes realizadas por los asistentes de ordenador[20].

- La salud y la medicina, Figura 4.11 Son una de las aplicaciones más grandes del modelo 3D, encuentra su aplicación en la planificación quirúrgica, la creación de réplicas numéricas de órganos y las simulaciones de procedimientos médicos. De esta manera, ella contribuye grandemente a la formación y a la precisión en la práctica de la medicina. Esto permite plantear diagnósticos precisos con una gran probabilidad y obtener un tratamiento muy eficaz[20].

- Realidad virtual (RV) y realidad aumentada (RA), Figura 4.11: Es fundamental en el desarrollo de experiencias VR y AR para obtener experiencias inmersivas, simulaciones de educación y formación profesional en diversos campos. La interacción con los modelos ayuda a mejorar la comprensión y la experiencia del usuario[20].

Figura 4.11: Automovilística, medicina, realidad virtual

La ventaja son varias también es un avance innovador en muchos dominios tecnológicos. Su capacidad para generar modelos muy detallados y exigentes lo convierte en una herramienta importante

para la simulación, la concepción y la comunicación visual de proyectos complejos. La modelización 3D es especialmente crucial para la creación de un mundo atractivo, interactivo e inmersivo en la industria del entretenimiento, donde el interés del público se conserva y se mejora la experiencia.

4.12.2. Evolución del modelado 3D

La historia del modelado 3D comenzó en la década de 1972, cuando Edwin Catmull y Fred Parke crearon la primera mano animada digital en 3D por computadora titulada *A Computer Animated Hand* mientras estudiaban en la Universidad de Utah fueron capaz de ejemplificar el potencial que tenía la computadora para crear objetos 3D. Mostrando la flexión de los dedos de la mano, y rostros humanos digitalizados. Esto también sirvió como prueba de un concepto que sería la base en las futuras técnicas de modelado y renderizado 3D. El trabajo fue innovador porque utilizaron los conceptos de *wireframe* y mapeo de texturas para permitir la representación exacta de la estructura y la superficie de un modelo tridimensional. Catmull luego se convertiría en presidente de *Pixar* y *Walt Disney Animation Studios*, abordó conceptos que se convertirían en principios fundamentales de los gráficos por computadora.[144]

Cabe mencionar que este proyecto se adelantó a su tiempo en gráficos por computadora para el cine y cualquier otro campo visual. Este avance técnico se enmarcaba más en el ámbito de la investigación científica que en el del arte. Esta situación aumentó la importancia de la precisión, a la que se prestaba bastante bien, y el desarrollo de algoritmos que trabajaban con superficies curvas.[144] Sin duda, esto seguiría ejerciendo una profunda influencia sobre todo el desarrollo posterior del modelado 3D en estos años. Su contribución fue fundamental para el desarrollo de los efectos visuales actuales que se utilizan tanto en películas como en juegos.

Las décadas siguientes a la primera película de animación llevaron a la tecnología de modelado 3D a través de una enorme evolución y desarrollo hasta la posición que ahora ocupa en la actualidad como una herramienta completa en la creación digital. Los ordenadores personales de la época se volvieron más potentes y económicos, y el modelado 3D abrió nuevas dimensiones para el entretenimiento, especialmente en el cine y los juegos de ordenador, hacia una experiencia visual más realista e inmersiva. Todo esto se hizo no sólo para mejorar la calidad de las texturas y su uso en superficies, sino para avanzar modelos totalmente optimizados para un mejor rendimiento, y que dan una técnica avanzada para retopologizar y hacer mapas *UV*, que hoy en día es un requisito en el desarrollo de juegos.[144]

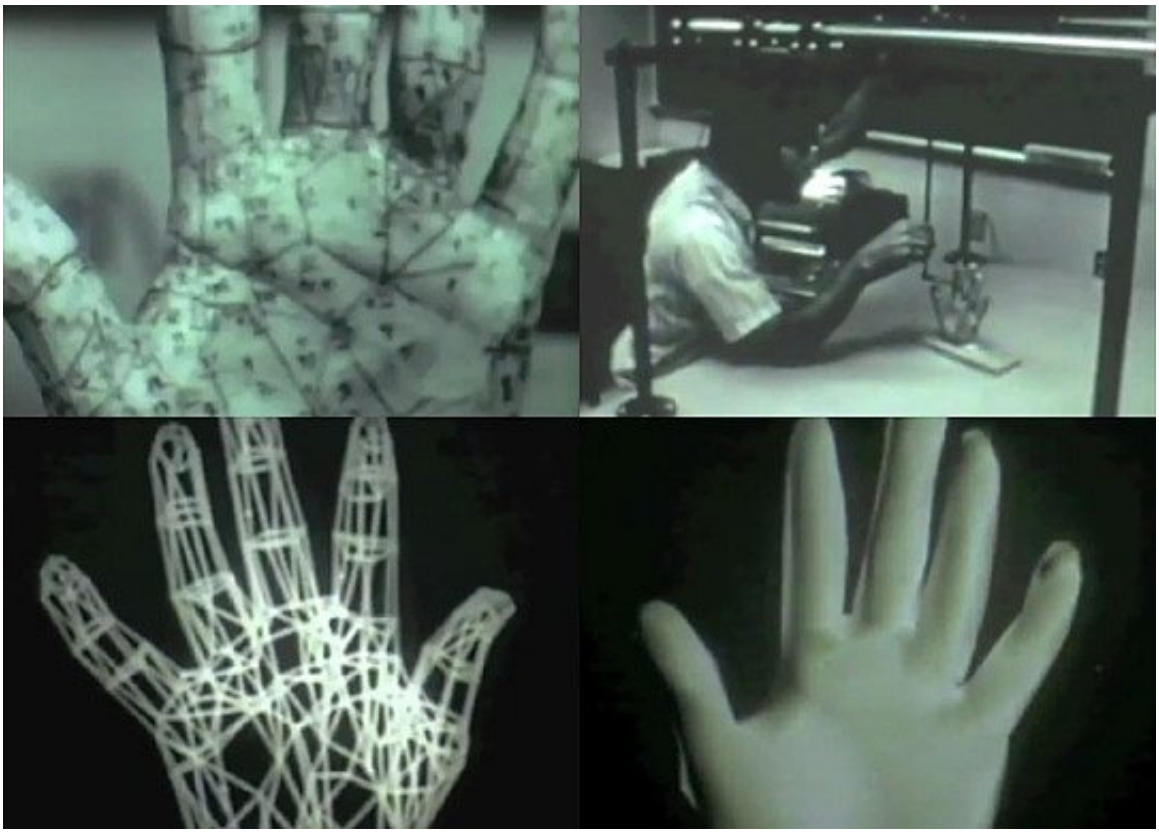

Figura 4.12: Imágenes del video *A Computer Animated Hand*

4.12.3. El modelado 3D en los videojuegos

En cierto modo, el modelado 3D en los juegos fue una especie de fuerza transformadora; fue la redefinición de las diferentes formas en las que los jugadores podían interactuar con el juego y el origen de nuevas posibilidades que permitieron los mundos de juego tridimensionales verdaderamente inmersivos. Esta tecnología en el desarrollo de juegos ha impulsado algunos cambios muy radicales a lo largo de los años en esta industria.

Cambiaria el aspecto del diseño de juegos en 3D a mediados de los años 90. Antes de eso, la historia de los videojuegos se basaba principalmente en gráficos 2D. Esto se debe a que la tecnología poligonal se introdujo en la quinta generación de consolas, comúnmente conocida como la generación 3D. Esta generación fue mejorada aún más por la *Sony PlayStation*, *Nintendo 64* y *Sega Saturn*. El paso de un mundo 2D a uno 3D cambió no solo la apariencia de los videojuegos sino también la jugabilidad misma. El uso del 3D en este caso hizo un llamado a la redefinición de la cámara en los juegos. En este sentido, permitió automáticamente diferentes perspectivas que cambiarían la forma en que el jugador experimentaba el juego. Se desarrollaron tres enfoques principales para la cámara[116]:

- Cámara de seguimiento: Esta es similar a la utilizada en obras de teatro en 2D, que tenía un efecto de paralaje, que estaba en el personaje frontal, lo que permitía que las escenas se enfocaran en el mismo. El hecho de que el personaje est*UV*iera en movimiento hizo que este tipo de cámara de seguimiento se utilizara en juegos como *Crash Bandicoot*, Figura 4.13 [116].

- Cámara estacionaria: Similar a la utilizada en *Resident Evil (1996)*, Figura 4.14, se colocaba

en un área fija de tal manera que cada vez que un jugador se movía del marco, solo cambiaba el ángulo. Puede que haya limitado la visión del jugador, pero tenía el valor añadido de la cinematografía porque añadía un elemento de suspenso y tensión[116].

- Cámara interactiva: Las primeras implementaciones se dieron en títulos como Super Mario 64 en 1996, para dar al jugador un control más directo sobre las vistas, presumiblemente otorgando la posibilidad de explorar escenarios tridimensionales. Se puede decir que este tipo de cámara presenta un avance muy importante al facilitar la navegación por un mundo en tres dimensiones[116].

Figura 4.13: Imágenes del juego *Crash Bandicoot*

Figura 4.14: Imágenes del juego *Resident Evil (1996)*

La Sony PlayStation fue definitivamente una de las consolas que se destacó en esta primera era del 3D, contando con una asombrosa cantidad de juegos que pusieron el listón muy alto para la base del diseño de la tridimensionalidad en los videojuegos. Juegos como *Tomb Raider*, Figura 4.15 y *Metal Gear Solid*, Figura 4.16 comenzaron a utilizar cinemáticas pre-renderizadas, formas de permitir historias más profundas y visualmente mejoradas. Estas eran escenas realizadas dentro del juego, que marcaron una diferencia en la narrativa de los videojuegos, desarrollando empatía hacia los personajes y mejor entendimiento de la historia[70].

Figura 4.15: Imágenes del juego *Tomb Raider*

Figura 4.16: Imágenes del juego *Metal Gear Solid*

Por otra parte, con la *Nintendo 64* estableció el estándar con títulos como *Super Mario 64*, Figura 4.17 y *The Legend of Zelda: Ocarina of Time*, Figura 4.18. Las dimensiones 3D encontraron un

significado completamente nuevo para controlar un personaje en ellos. Esos juegos no solo dominaban el espacio en tres dimensiones; se introdujeron muchas cosas nuevas, como la puntería durante la batalla, que realmente puso los juegos en perspectiva dentro de esos reinos complejos[70].

Figura 4.17: Imágenes del juego *Super Mario 64*

Figura 4.18: Imágenes del juego *The Legend of Zelda: Ocarina of Time*

En esa etapa temprana de la vida, el modelado 3D tenía algunas limitaciones críticas en la apariencia tosca de los modelos, que estaba en las memorias y el procesamiento, característico de esa época. La primera generación de juegos tridimensionales presentaría gráficos 3D que eran en su mayoría "toscos" y "puntiagudos" considerando el hecho de que incluso la texturización era imposible ya que las consolas no podían soportar texturas complejas y tampoco podían soportar muchos polígonos. Estos desafíos iniciales facilitaron la innovación y, muy pronto, gracias a ellos, la tecnología avanzó para garantizar una mejora en la calidad de los gráficos y el buen funcionamiento de los juegos[116].

Evolucionó y, con el avance de la tecnología, se hicieron posibles juegos mucho más realistas y cercanos a la vida real. A medida que las consolas crecieron en potencia, los motores gráficos avanzados desarrollaron una importante herramienta de narración visual en los juegos capaces de proporcionar una experiencia real a través de la interacción dentro del juego. Hoy en día, el 3D no solo se trata de modelar personajes o escenas; se utiliza para modelar entornos complejos, iluminación del mundo real y sistemas físicos.

El legado del 3D sigue vivo y muy presente en los juegos, ya que la industria sigue avanzando y revolucionando. Esta transición en la década de 1990 al 3D ha sido solo el comienzo de una revolución visual que ha ido creciendo a partir de ese día, con cada nuevo detalle y desarrollo realista en los gráficos que desafía la imaginación del jugador y del desarrollador por igual.

4.12.4. Técnicas populares

1. Polígonos, Figura:13.9

   Sin duda, la técnica que más fama tiene en el desarrollo de modelos de videojuegos es la del modelado poligonal. Se trata de una forma de construir formas tridimensionales mediante una serie de puntos, líneas y planos bajo un control y un orden que define una malla poligonal. Añadiendo y manipulando los polígonos se construirá la forma más compleja, dando lugar a modelos sencillos o muy detallados.[27]

   Las ventajas son que permite controlar la topología del modelo, que posteriormente puede optimizarse para la variedad de plataformas en las que se puede ejecutar el juego. Los modelos poligonales funcionan notablemente bien para fines de animación y simulación, ya que se puede

cambiar fácilmente el nivel de detalle utilizando estos modelos sin deteriorar la calidad visual. Esta será la aplicación más general para la construcción de modelos orgánicos en la industria de los videojuegos.[108]

2. Fotogrametría/Modelado basado en imágenes, Figura:13.10

   Esta es una técnica que se utiliza para el modelado basado en imágenes, donde se toman una gran cantidad de fotografías de un objeto físico desde perspectivas variadas y distintivas. De esta manera, se modela un espacio tridimensional aproximado a partir de dichas imágenes. Luego, estas imágenes se fusionan en un modelo tridimensional mediante un *software* específico.[108]

   Es capaz de capturar texturas realistas y detalles finos de la escala de objetos del mundo real, incluso imperfecciones de la textura o la apariencia de la superficie. La fotogrametría, por lo tanto, permite una recreación muy precisa de objetos del mundo real, incluso hasta detalles tan minuciosos como la textura de la superficie y las imperfecciones. Es utilizado en los videojuegos realistas y proyectos que tratan con recreaciones históricas donde la mayoría de las veces, el fondo visual importa. Sin embargo, sus aplicaciones para la animación de personajes son limitadas ya que el rendimiento requiere la optimización del contenido.[108]

3. Escultura, Figura:13.11

   La escultura digital, hasta el último detalle, es una técnica donde los artistas pueden ejecutar modelado similar a la forma en que se moldea la arcilla con la ayuda de las herramientas de escultura. Las herramientas de *software* de escultura se utilizan para modelar y esculpir detalles minuciosos en millones de polígonos.[137]

   Esto lo hace realmente útil para crear formas orgánicas complejas, como personajes o criaturas que pueden tener detalles finos esculpidos directamente en la malla de alta resolución. A partir de ahí, más adelante en el modelo, se puede preparar para juegos con retopología. Esto se ha utilizado mucho para el modelado de personajes con detalles más finos. El procedimiento general que sigue suele estar precedido por la etapa inicial que incluye la preparación de modelos para videojuegos, el uso de procesos como el modelado poligonal y la optimización de la topología.[61]

4. Escaneo 3D, Figura:13.12

   Es uno de los procesos, donde se aplican dispositivos especiales como como escáneres láser, escáneres de luz estructurada o técnicas de fotogrametría, para capturar la forma, el color y la textura de objetos o entornos del mundo real. Es el proceso ideal para reproducir objetos del mundo real con una precisión extremadamente alta. Permite hacer modelos altamente fotorrealistas en tiempo récord, lo cual es efectuado para el proyecto que se detallará, una vez que se realice la reproducción de un objeto, como puede ser un efecto especial en una película o la creación de una réplica de una persona o cosa. Las aplicaciones son videojuegos muy específicos, quizás en tiempo real, que requieren ajustes y optimización para funcionar incluso en motores de juegos que exigen mucho rendimiento. [108]

5. *NURBS*, Figura:13.13

   Las *NURBS* son un enfoque para el modelado geométrico de un objeto que utiliza una curva matemática para establecer superficies suaves y precisas; no se aplican a una malla de polígonos. Por lo tanto, las *NURBS* se diferencian del modelado poligonal, en el que se sabe que la malla no está definida en absoluto por cada vértice solo.[137]

   Sus beneficios son una precisión inigualable en la curva y la superficie detallada, lo que lo hace ideal para el diseño en las áreas industrial y automotriz. Además, las *NURBS* se modifican fácilmente y, por lo tanto, se pueden usar para el desarrollo de formas complejas. Sin embargo, es menos común para los videojuegos, ya que el proceso de convertir superficies *NURBS* en polígonos optimizados es muy costoso en términos de complejidad. Por otro lado, para el modelado con requisitos de geometrías suaves, es bueno para vehículos y elementos de diseño.[108]

6. Modelado procedural, Figura:13.14

   El modelado procedural es el que crea modelos 3D mediante algoritmos, modela automáticamente entornos y objetos que de otra manera resultarían difíciles. Utiliza parámetros preestablecidos para generar contenido 3D. Es la solución más adecuada para la creación altamente acelerada, rápida y eficiente de cantidades masivas de contenido 3D, como terrenos, edificios o una ciudad. La generación procedural permite agregar detalles a una escala tal que mantenga la coherencia con el diseño.[108]

   Se utiliza en videojuegos de mundo abierto y videojuegos que requieren entornos de juego grandes y extensos, que a menudo necesitan una gran cantidad de variedad visual, pero donde el costo de creación de contenido no se puede aumentar y por no tener largos tiempos de producción.[108]

7. Modelado de bordes (*Edge Modeling*), Figura:13.15

   Se deriva de fijar la definición del modelo 3D con respecto a contornos clave, teniendo en cuenta todos los bordes y vértices. De esta manera, el artista puede tener un control preciso sobre la forma final del objeto y la distribución de su topología. Esto sería lo más útil para proyectos altamente detallados topológicamente, especialmente en personajes animados, donde un flujo de bordes bien definido aliviará algunos de los problemas asociados con la deformación de la geometría bajo animación.[108]

   Se requiere en la generación de modelos que requieren una amplia animación, ya que permite definir un flujo de polígonos para resaltar características realistas, móviles y expresivas.

8. *Kitbashing*, Figura:13.16

   El *Kitbashing* es una de las características que acelera el proceso de modelado porque permite que las partes que ya están diseñadas se recombinen en nuevas configuraciones para ahorra tiempo en el desarrollo de modelos complejos como entornos de construcción, estructuras mecánicas y otros detalles finos. Aprovecha los recursos que ya tienes para mantener la coherencia visual en tus proyectos.[39]

   Es habitual usarlo en la creación de entornos de juegos, en los que se pueden construir rápidamente entornos completos a partir de piezas prefabricadas modulares.[39]

9. Modelado de caja/Subdivisión (*Box modeling/Subdivision*), Figura:13.17

   Esta técnica comienza con una forma básica que se puede esbozar rápidamente y luego refinar con subdivisiones (como cubos). Eficiente, permite un buen control sobre la calidad del modelo final y es más apropiado para proyectos que se ejecutan bajo un cronograma ajustado y/o con restricciones de recursos. Este método también va a hacer que la creación de personajes y objetos tanto orgánicos como inorgánicos sea más fácil, por lo tanto, muy eficiente en el flujo de trabajo.[99]

Para este proyecto se han elegido las técnicas de Modelado Poligonal y Caja/Subdivisión por ser eficientes en el proceso y control de detalles en la creación de modelos. De esta forma, a partir de las formas más simples, estas técnicas se van refinando progresivamente, por lo que es más fácil optimizar la malla en diferentes niveles de detalle sin perjudicar el rendimiento. En la industria del desarrollo de videojuegos, estas son las técnicas preferidas, ya que son compatibles con los motores de juegos modernos, lo que permite mantener la calidad visual requerida manteniendo el rendimiento.

El uso de escultura se limitó específicamente al personaje Dingo secuaz, debido a la particularidad compleja de su cabeza y dedos, lo que requeriría un enfoque mucho más detallado. Este método se adaptó en este caso especial solo porque el trabajo de retopología, para la optimización de los modelos esculpidos, resultó ser demasiado laborioso para el cronograma de este proyecto. Por esta razón, se abandonó el esculpido de los personajes restantes para concentrarse en otros métodos más rápidos y eficientes que pudieran soportar el cronograma.

No se habían considerado técnicas como la fotogrametría, el escaneo 3D y el modelado *NURBS*; después de todo, el enfoque debía estar en la precisión realista en comparación con los requisitos del proyecto de una estética estilizada y caricaturesca. El modelado procedimental y el *Kitbashing*, en un sentido similar, fueron excluidos, ya que las metodologías de ese tipo restringían el grado de personalización que necesitaban los personajes y objetos únicos del juego. Por otro lado, el modelado poligonal y la subdivisión de cajas brindaron la flexibilidad y adaptabilidad que se requerían: un flujo de trabajo tolerable y liviano que mantendría la consistencia y calidad visual para el alcance necesario para el proyecto.

4.13. Conceptos clave para el desarrollo de modelos 3D para videojuegos

4.13.1. *Blocking*

El *blocking* es la primera etapa técnica del modelado 3D después del desarrollo del arte conceptual. El arte conceptual es una representación visual de la etapa inicial que define las características estéticas y los aspectos clave del personaje u objeto dentro del juego. Este arte generalmente consiste en bocetos, ilustraciones o gráficos que describen la dirección visual, detallando elementos esenciales del personaje como la forma, la postura y los accesorios. Es la base sobre la que sigue el resto del proceso de modelado para asegurarse de que el entorno tridimensional se traduce correctamente para dar forma a la visión artística.[146]

La etapa de *blocking* es la creación de la estructura general muy aproximada del modelo 3D utilizando formas geométricas sencillas como cubos, cilindros y esferas una vez que se ha definido el arte conceptual. El *blocking* ayuda a establecer rápidamente la silueta y las proporciones del modelo sin preocuparse por la delicadeza de los detalles en esta etapa del proceso. Esto resulta muy útil para ver si la forma general ha resultado realmente como se espera en el concepto. Aquí es donde se define la estructura primaria del modelo para que se pueda asegurar el equilibrio y las proporciones correctas en esta etapa antes de ingresar la frase de detalle del personaje u objeto.[146]

También es el momento en el que se puede experimentar con las proporciones y la posición de los diferentes elementos, de manera que todos ellos se alineen con la visión inicial del proyecto. Esto significa un ajuste de las proporciones básicas del personaje: qué tan largas deben ser las extremidades o qué tamaño debe tener la cabeza con respecto a las otras partes. Se pueden realizar cambios muy fácilmente sin arriesgar el flujo de trabajo, lo que permite la iteración y la corrección de posibles problemas estructurales antes de avanzar. Una base sólida en el *blocking* garantiza que los pasos que siguen se realicen con precisión y eficiencia en línea con la visión del proyecto.[146]

4.13.2. Retopologia y flujo de bordes (Edge flows)

La retopología es una fase crítica, que generalmente se realiza después de un modelo 3D ya existente y bastante detallado, creado mediante las diversas técnicas de escultura digital. La retopología se basará en la reconstrucción de una malla 3D para mejorar su geometría, lo que se refiere a una reducción de la densidad de polígonos y al ajuste de la ubicación de los bordes en una configuración más funcional. Será aún más crítico en los videojuegos, ya que el rendimiento dependerá en gran medida de cuántos polígonos se deban renderizar en tiempo real.[29]

La retopología permite que el modelo mantenga un alto nivel de detalle visual al tiempo que reduce la complejidad. En general, se construye una malla de alta resolución que definirá todos los detalles que estarán bien, y luego se construye una versión que contiene menos polígonos y está optimizada. Lo que conlleva a seguir el flujo de bordes adecuado, que indica la disposición correcta

de los bordes de la malla, se puede deformar suavemente mientras se anima. El beneficio del flujo de bordes adecuado se puede ver en que permite la creación de animaciones realistas y suaves mucho más fácilmente en articulaciones de personajes complejas como codos, rodillas y configuraciones faciales.[22]

Un flujo de bordes bien creado normalmente es utilizando polígonos de 4 lados, también llamados cuadriláteros; son más fáciles de usar y no causan la resultante del bucle de bordes, que despeja el camino para subdivisiones suaves que evitan la distorsión. Los cuadriláteros son buenos para usar en la creación de un bucle de borde ya que siguen la anatomía del modelo. Con esto, podemos obtener una estructura clara para editar los detalles y agregar geometría de manera precisa. El bucle es necesario para las áreas de alta tensión en la malla, donde se garantiza que el flujo y el aspecto de las animaciones sean suaves y naturales.[22]

Hay dos tipos de bordes que uno puede encontrar en el proceso de modelado 3D, estructurales y de soporte. Los bordes estructurales establecen la forma general del modelo y son los más importantes para la silueta. El otro tipo son los bordes de soporte que ayudarán a uno a mantenerse nítido dentro de ciertas áreas cuando se subdivide la malla, o ayudarán a preservar la nitidez de los bordes alrededor de las esquinas afiladas u otros detalles que se consideran importantes. Los bordes de soporte garantizan que se mantenga la geometría y evitan que los detalles se suavicen en exceso. Además, existe una técnica (*creasing*/el plegado) que mantendrá los bordes afilados en su lugar sin agregar pliegues de soporte adicionales.[65]

Como se había mencionado el modelado 3D prefiere fuertemente el uso de quads porque brinda control sobre el modelo y la subdivisión de la malla. Aún así, el uso de tris (triángulos) se puede aplicar en varios casos; principalmente, esto ayuda a optimizar la geometría para los videojuegos, ya que de esta manera se puede reducir la cantidad de polígonos en los objetos estáticos. Los n-gons, por otro lado, definitivamente deben evitarse (los n-gons son polígonos con más de cuatro lados). Pueden causar problemas durante la subdivisión y la renderización, y es probable que provoquen errores en la interpretación de la geometría por parte de los motores de juegos. Por lo tanto, mantenerlo limpio con una topología basada en quads y un mínimo de tris conduciría a modelos fuertes y eficientes.[65]

4.13.3. Simetría

Simetría en el modelado 3D Esta es a menudo una técnica utilizada al comienzo de un proceso de modelado donde una persona puede generar solo un lado de un modelo y luego duplicarlo o reflejarlo. Es útil con los personajes porque la mayoría de ellos tienen una estructura simétrica, lo que ahorrará mucho tiempo y garantizará que las proporciones sean las mismas en ambos lados.[22]

En los videojuegos, la simetría es una práctica común al desarrollar personajes u objetos porque facilita su edición y garantiza que ambos lados del modelo sean perfectamente iguales, minimizando las posibilidades de error. La simetría se vuelve importante sobre todo durante la fase de bloqueo y retopologización, ya que permite la construcción de modelos de forma muy rápida y eficiente. Pero, una vez que esta base simétrica está ahí, es bastante común poner detalles asimétricos para mejorar visualmente el interés y el realismo en los personajes y objetos.[22]

4.13.4. Pose T

La pose T es una posición inicial utilizada en el mundo del modelado 3D. El personaje tiene tanto los brazos como las piernas un poco estiradas del cuerpo para colocarlos en forma de letra "T". Esta posición es importante para fines de *rigging*. El *rigging* implica configurar un esqueleto digital que hará que el modelo sea capaz de ser animado. La pose T es más sencilla para asignar pesos a las articulaciones y también garantiza que la malla esté en la posición correcta para la deformación durante la animación.[123]

Facilita el proceso de animación porque todas las partes del modelo se vuelven accesibles, minimizando así los problemas de deformación de las partes en consideración, como los hombros y las caderas. Otra ventaja es que la pose T es estándar en la mayoría del *software* de modelado 3D, lo que facilita la integración de un modelo ya realizado en el *software* utilizado por los motores de juegos o cualquier otro *software* de animación.[123]

4.13.5. Mapeos UV / UV Mapping y texturizado

Otro punto importante son los mapeos *UV*, que permiten adjuntar elementos bidimensionales texturas a la superficie de un modelo tridimensional. En pocas palabras, es la malla 3D dispuesta en un espacio 2D para crear un mapa *UV*, que actúa como una especie de plantilla para las texturas. El proceso de mapeo *UV* es importante porque asegura que las texturas de los objetos se apliquen con precisión y sin ningún tipo de distorsión o desalineación. Por lo tanto, es importante con los modelos porque logrará el mayor nivel de eficiencia visual. Un buen mapa *UV* optimiza el uso de la textura, aportando más detalles sin aumentar la cantidad de polígonos en el modelo. La texturización también agrega color, detalles de la superficie y efectos visuales, que agregan realismo y estilo a un modelo.[139]

La texturización es el proceso de añadir textura, que son estas apariencias y cualidades de la superficie, a las coordenadas *UV* de un modelo mediante algunas herramientas que darían patrones, colores y ciertos detalles a la superficie. Ayuda a construir personajes y escenarios interesantes dentro del videojuego. Estos conceptos hacen que los modelos 3D sean estéticos y se puedan preparar adecuadamente para la animación y para su uso dentro de los motores de juegos modernos, creando un equilibrio entre la calidad visual y el rendimiento técnico en el desarrollo de juegos.[22]

4.14. Animación 3D

La animación 3D se presenta como una forma de arte y creatividad que emplea herramientas digitales para infundir vida a personajes, objetos y universos fantásticos en una variedad de plataformas, incluyendo películas, series, videojuegos y otros formatos visuales. Su versatilidad permite narrar historias de manera visualmente impactante, además de ofrecer una amplia variedad de estilos artísticos. [30]

4.14.1. Definición de animación 3D

El término "animación" proviene del latín anima, que significa "alma". En este sentido, el concepto de animar se refiere a dotar de vida o alma a un personaje u objeto, haciéndolo parecer que piensa, actúa y se mueve de manera autónoma. En el caso de la animación 3D, este proceso no solo se enfoca en dar movimiento, sino que también implica la capacidad de girar y manipular los objetos dentro de un espacio tridimensional. [47]

Por ende, la animación 3D involucra un proceso que permite otorgar movimiento y vida a objetos inanimados dentro de un espacio tridimensional. A diferencia de la animación tradicional en dos dimensiones (2D), la animación 3D se caracteriza por su capacidad de manipular objetos en tres ejes: X, Y y Z, lo que proporciona una sensación de profundidad y realismo. Este tipo de animación se realiza mediante el uso de softwares especializados que permiten modelar y esculpir digitalmente personajes y objetos. A través de estos programas, los diseñadores pueden dotar de personalidad y expresividad a los elementos animados, logrando transmitir emociones y narrar historias. [138]

Aplicaciones de la animación 3D

La animación 3D se ha convertido en una herramienta fundamental en diversas industrias, transformando la manera en que se crean y visualizan contenidos. En el cine y la televisión, permite la creación de mundos y personajes con un realismo inigualable, como lo demuestra la película "Toy Story", pionera en el uso de la animación por ordenador. En los videojuegos, esta técnica ha sido clave para el desarrollo de entornos inmersivos y personajes detallados, como se puede ver en "The Legend of Zelda: Breath of the Wild". También ha revolucionado campos como la publicidad, la arquitectura y la educación, ofreciendo soluciones visuales creativas y precisas que mejoran la presentación de productos, la visualización de proyectos arquitectónicos y la comprensión de conceptos complejos. Estas aplicaciones se reflejan en múltiples sectores, mostrando la versatilidad de la animación 3D no solo como herramienta técnica, sino también como un medio para potenciar la creatividad y el impacto visual en diferentes contextos.[62]

Como es notorio, la animación 3D tiene una amplia gama de aplicaciones, ya que su propósito principal es dar vida a objetos inertes. Esta técnica es extremadamente flexible y puede adaptarse a las necesidades específicas de cada sector, utilizando diferentes metodologías según los objetivos y el contexto de aplicación (Torres, 2024). Algunos de sus usos más destacados incluyen la creación de películas, series animadas, videojuegos, publicidad, efectos visuales (VFX), realidad virtual y aumentada, video mapping, aplicaciones y simulaciones científicas. [47]

Estilos de animación 3D

Existen innumerables estilos dentro de la animación 3D, ya que cada artista y cada estudio pueden interpretar y aplicar las técnicas según sus necesidades artísticas o presupuestarias. A pesar de esta variedad, se identifican tres grandes categorías de animación 3D: realista, cartoon y snappy. [47]

• Animación realista: Este estilo busca emular los movimientos naturales de seres vivos, como humanos o animales. Aunque se puede crear de manera manual, la complejidad del movimiento hace que sea común el uso de sistemas de captura de movimiento (motion capture), especialmente en cine y videojuegos [47].

• Animación estilo cartoon: Este enfoque simplifica la realidad de manera estilizada y atractiva, utilizando principios básicos de la animación para dar credibilidad a los personajes y escenarios. Ejemplos notables de este estilo se pueden encontrar en películas como Angry Birds y Ugly Dolls. [47]

• Animación estilo snappy: Este tipo de animación es una versión exagerada de la animación cartoon, caracterizada por movimientos muy marcados y dinámicos. El uso acentuado de los principios de estiramiento, aplastamiento y exageración es clave en este estilo. Ejemplos incluyen producciones como The Food Thief y trabajos de animadores como Jorge Vigara. [47]

4.14.2. Tipos y técnicas de animaciones

La animación 3D abarca una variedad de técnicas que se adaptan a distintos objetivos creativos y necesidades de producción. A continuación, se describen algunas de las técnicas más representativas:
• Motion Design: El Motion Design es una técnica de animación que combina diseño gráfico con movimiento, creando piezas audiovisuales visualmente impactantes. Utiliza elementos como formas, colores, música y efectos sonoros para generar una experiencia sensorial integrada. [62]

• Stop Motion: El Stop Motion es una técnica de animación que utiliza objetos físicos, como muñecos o figuras, que son fotografiados fotograma a fotograma mientras se realizan ligeros cambios

en su posición. Al juntar las imágenes, se crea la ilusión de movimiento continuo. Esta técnica, aunque laboriosa, ha sido utilizada durante décadas en cine y televisión, destacando por su encanto artesanal y su capacidad para crear una animación con un estilo único. [62]

• Pixilación: Esta técnica puede ser un poco similar a la de Stop Motion, aunque se diferencia en que la pixilación no simula el movimiento de objetos o entidades. La pixilación es una técnica que emplea a personas reales como personajes animados. A través de la captura fotográfica de diferentes posiciones de las personas, se puede crear una secuencia en la que parecen moverse de manera inusual o antinatural. Este estilo de animación es útil para lograr efectos visuales que serían difíciles de recrear con animación tradicional o digital, ofreciendo una estética única y creativa. [62] La técnica se dedica a capturar imágenes estáticas para que continuamente, estas sean juntadas a una velocidad de 24 frames por segundo, lo que simula el movimiento. [62]

• Animación hiperrealista: La animación 3D hiperrealista tiene como objetivo recrear imágenes que sean casi indistinguibles de la realidad. Se utiliza para generar personajes, entornos y objetos con un nivel de detalle tan preciso que puede confundirse con imágenes reales. Esta técnica es común en publicidad, cine y videojuegos, especialmente cuando se requiere la recreación de entornos y personajes realistas que interactúan en mundos imaginarios o escenas de acción. [62]

• Animación de caricatura: La técnica de animación 3D de caricatura recrea el estilo exagerado de los dibujos animados en un entorno tridimensional. Los personajes suelen tener rasgos distorsionados, proporciones fuera de lo común y movimientos exagerados para producir un efecto humorístico y visualmente llamativo. Esta técnica es utilizada en producciones de entretenimiento, donde la creatividad y el estilo cómico son esenciales. El objetivo de este radica en hacerlos sencillos y divertidos.[62]

Keyframing

La animación conocida como keyframe se refiere a una técnica en la que se introduce "keys" (claves) en frames (fotogramas) a lo largo de una línea de tiempo. Esta técnica se distingue de otras formas de animación, como el stop motion y el motion capture, en su enfoque y metodología.

La animación keyframe se utiliza en software de animación 3D y 2D, donde el animador debe tomar decisiones sobre qué claves aplicar y en qué fotogramas hacerlo. En el contexto de la animación de personajes, esto implica seleccionar poses específicas que serán registradas en momentos determinados de la secuencia a animar. [69]

Los keyframes actúan como puntos de referencia dentro del software 3D, definiendo la posición, rotación y escala del objeto animado. A medida que el software interpola entre estas claves, se genera el movimiento del objeto, logrando así el efecto de animación deseado. Para establecer un keyframe, lo primero que se debe hacer es ubicarlo en la línea de tiempo del software de animación. En este punto se selecciona el instante en el que el efecto inicializará su transición y se fija el fotograma clave. Hasta esa marcación, la animación avanzará sin modificaciones; los cambios se activarán después de este punto. [106]

A continuación, se debe identificar el momento en que el efecto concluirá, lo que se logra mediante la creación de un segundo fotograma clave. Este segundo keyframe delimitará el intervalo del movimiento o cambio. Después de establecer ambos puntos, se procederá a modificar los valores deseados que se aplicarán en el intervalo seleccionado, permitiendo así que la animación se desarrolle de manera fluida y coherente. [106]

La animación keyframe se desarrolla a través de varias fases:

• Blocking: En esta etapa, el animador establece únicamente las keys más significativas de la secuencia, con el objetivo de contar la historia utilizando la menor cantidad de claves posible. Esta

fase es fundamental para sentar las bases de la animación. [69]

• Blocking Plus: En esta fase, el animador añade keys adicionales que ayudan a definir con mayor precisión el movimiento de los objetos. Esto permite una mayor claridad en la acción y en la narrativa visual. [69]

• Refine: Durante la etapa de refinamiento, el animador revisa y ajusta las keys, añadiendo, eliminando o modificando las claves existentes para lograr un movimiento más pulido y creíble. Esta fase es crucial para asegurar que la animación sea fluida y convincente. [69]

Motion Capture

Este proceso consiste en registrar el movimiento de seres humanos, animales o elementos físicos y trasladar esos datos a un modelo en dos o tres dimensiones. La captura de movimiento o mejor conocido como Motion Capture busca animar estos modelos utilizando los movimientos reales capturados.

Las aplicaciones de esta técnica son amplias y variadas. Uno de los usos más conocidos se encuentra en la industria del entretenimiento, donde se emplean para grabar las acciones de actores y replicarlas en modelos 3D generados digitalmente. Esta práctica permite que tanto las animaciones como los videojuegos alcancen niveles de realismo impresionantes, enriqueciendo así la experiencia del espectador. [51]

Como Andreú menciona en su blog investigativo "Motion Capture en la industria de la Animación." Esta técnica implica que una persona use un traje especial que está equipado con sensores extremadamente sensibles. Estos sensores están estratégicamente colocados en distintas partes del cuerpo para registrar incluso los movimientos más sutiles. La finalidad de estos sensores es proporcionar a los artistas digitales, como animadores 3D y productores de efectos visuales, datos precisos y realistas sobre los movimientos del actor que lleva el traje. La información recopilada sirve como una base para los movimientos animados, ya que incluye detalles sobre el tiempo de cada acción, lo que facilita el proceso de postproducción.

El proceso de motion capture comienza con la creación de un modelo 3D por parte de los artistas digitales. Una vez que el modelo está diseñado, pasa al departamento de rigging, donde se le otorgan las estructuras necesarias para que pueda ser deformado y animado. Con el modelo listo, se lleva a cabo la captura de movimiento, donde los movimientos del actor se registran en tiempo real y se vinculan al modelo 3D. Esto permite que el modelo replique los movimientos del actor de manera instantánea, creando una animación más realista. [25]

Después de realizar la captura de movimiento, los datos obtenidos se envían al departamento de animación, donde los animadores trabajan en perfeccionar las secuencias. En esta etapa, se crean keyframes y se ajustan los tiempos de los movimientos para asegurar una fluidez adecuada en las animaciones. Este meticuloso proceso, que combina la captura de movimiento con la animación y la expresión emocional, es fundamental para dar vida a las escenas que vemos en el cine hoy en día.

Una vez que el modelo 3D está listo, los artistas de efectos vinculan los movimientos registrados por el traje de motion capture a dicho modelo. Esto significa que, a medida que el actor se mueve, el modelo 3D responde en tiempo real, imitando esos movimientos de manera precisa. Posteriormente, la información es enviada al departamento de animación, donde comienza el proceso de refinamiento de las animaciones obtenidas. Durante esta fase, se generan keyframes y se desglosan las animaciones, refinando cada movimiento y ajustando los tiempos correspondientes para lograr una representación natural y fluida. [25]

Las escenas que observamos en el cine son el resultado de un meticuloso proceso de captura de movimiento, animación y expresión emocional. Un ejemplo reconocido a nivel mundial de la captura

de movimiento es “Avatar”, una película dirigida por James Cameron, que se estrenó en 2009. Esta película es célebre por su innovador uso de tecnología de captura de movimiento y efectos visuales, lo que le permitió crear un mundo completamente nuevo y realista.[43]

En la actualidad, el uso del motion capture se ha vuelto cada vez más común en diversas áreas del entretenimiento. A medida que la tecnología avanza e inclusive haciendo uso de la inteligencia artificial, también se está aplicando en campos como la ciencia y la biomecánica, entre otros.

4.14.3. Principios de animación

Como bien se conoce la animación es el arte de dar vida a objetos inanimados, creando movimiento a través de diversas técnicas como el dibujo, la pixilación, el stop motion y la animación digital en 2D y 3D. Estas técnicas permiten contar historias y transmitir incluso emociones. [86]

El primer texto que establece una base conceptual sólida sobre los principios de la animación se publicó en 1981 por Ollie Johnston y Frank Thomas, y se considera fundamental para cualquier artista en este campo. Sin embargo, existen variedad de autores como John Lasseter y otros, que han ampliado y adaptado estos conceptos a la animación digital. [93]

Los 12 principios de la animación, inicialmente diseñados para aumentar el realismo, siguen siendo relevantes en la actualidad. Estos principios, que se describen en la tesis doctoral de Cuesta Martínez (2015), incluyen:

• Estirar y encoger (squash and strech): Deformar los objetos para añadir dramatismo y transmitir su peso y flexibilidad, observable en la Figura 4.18.

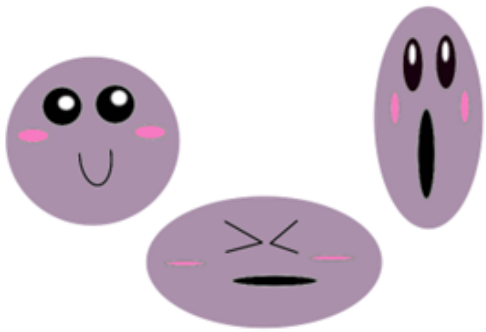

Figura 4.19: Principio “Estirar y encoger”

• Anticipación (anticipation): Preparar al espectador para una acción, guiando su atención hacia el lugar donde ocurrirá, observable en la Figura 4.19.

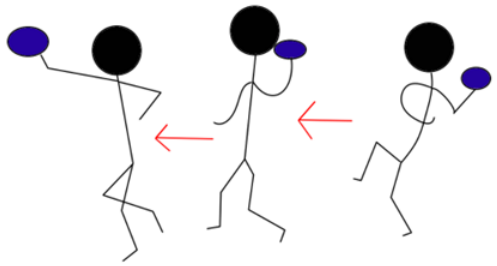

Figura 4.20: Principio “Anticipación”

• Puesta en escena (staging): Disposición de objetos y personajes para que la acción sea fácilmente comprensible, observable en la Figura 4.20.

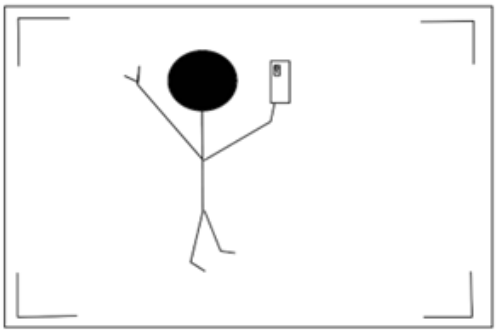

Figura 4.21: Principio “Puesta en escena”

• Animación directa y pose a pose (pose to pose): La animación directa se crea fotograma a fotograma de manera fluida, mientras que pose a pose implica definir primero los fotogramas clave y luego crear los movimientos intermedios, observable en la Figura 4.21.

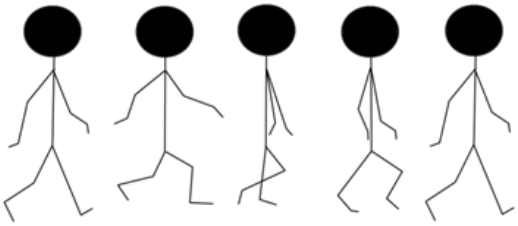

Figura 4.22: Principio “Animación directa y pose a pose”

• Acción continuada y superpuesta (follow through and overlapping action): Este principio asegura que las acciones del personaje se mantengan coherentes y realistas, mostrando que los movimientos siguen las leyes de la física, observable en la Figura 4.22.

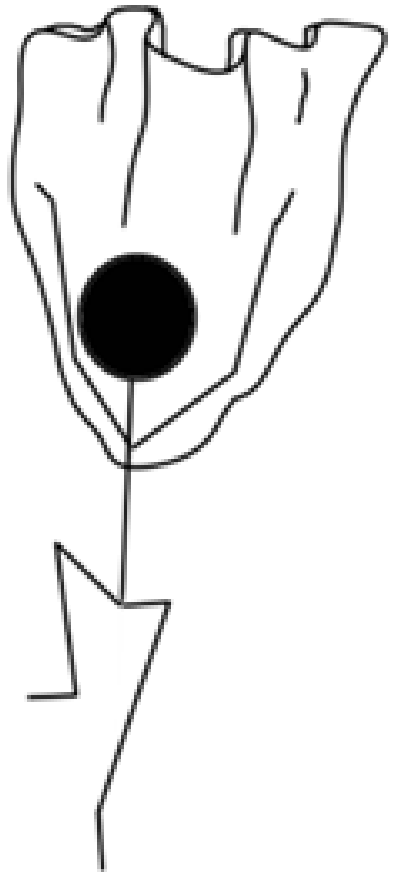

Figura 4.23: Principio “Acción continuada y superpuesta”

• Entradas lentas y salidas lentas (slot in and slow out): Los personajes u objetos necesitan tiempo para acelerar al inicio y desacelerar al final de una acción, ver en la Figura 4.23.

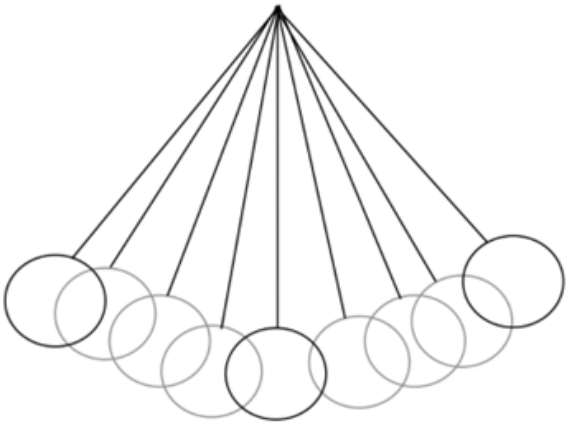

Figura 4.24: Principio “Entradas lentas y salidas lentas”

• Arcos (arc): Las acciones naturales se describen mediante trayectorias en arco, como el movimiento de los brazos o el salto, observable en la Figura 4.24.

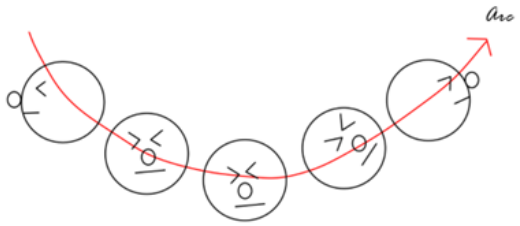

Figura 4.25: Principio “Arcos”

• Acción secundaria (secondary action): Agregar movimientos sutiles que complementen la acción principal, haciendo que esta sea más comprensible y natural, ver en la Figura 4.25.

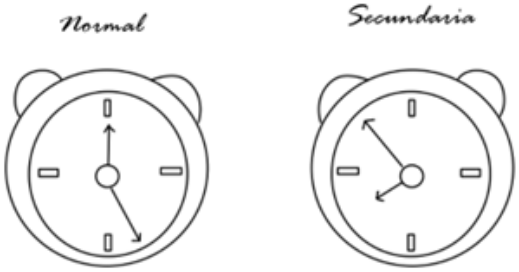

Figura 4.26: Principio “Acción secundaria”

• Temporización (timing): Se refiere a la cantidad de fotogramas utilizados en una acción, influyendo en la percepción de peso, velocidad y expresión, observable en la Figura 4.26.

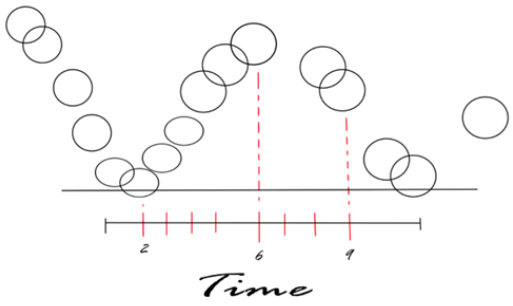

Figura 4.27: Principio “Temporización”

• Exageración (exaggeration): Ampliar los movimientos de los personajes para lograr un efecto dramático, observable en la Figura 4.27.

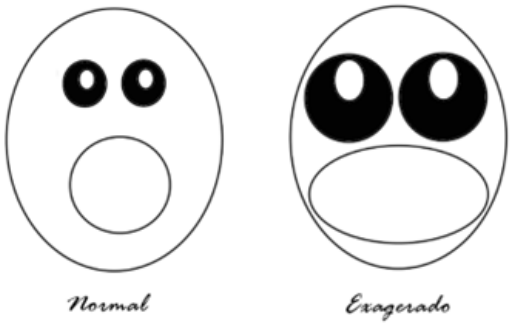

Figura 4.28: Principio “Exageración”

• Dibujo sólido (solid drawing): Presentar a los personajes de manera coherente, manteniendo profundidad y perspectiva durante todas sus acciones, ver en la Figura 4.28.

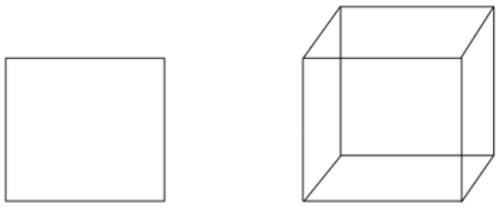

Figura 4.29: Principio “Dibujo sólido”

• Apelación (appeal): Los personajes deben tener una personalidad que conecte con el espectador, logrando una distinción basada en su apariencia, vestimenta y movimientos, observable en la Figura 4.29.

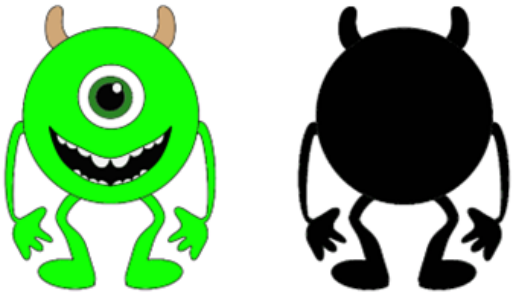

Figura 4.30: Principio “Apelación”

Estos principios, utilizados por los animadores de Disney desde finales de los años 20 y 30, son fundamentales para dar vida a los personajes y escenas animadas [91]

4.15. Diseño de personajes y caracterización de personajes en videojuegos

4.15.1. Definición de diseño de personajes

A pesar de que no todos los juegos tienen personajes, si un videojuego los utiliza, un buen diseño es esencial para brindarle una buena experiencia de juego al jugador. El diseño de un personaje se compone de dos partes principales: estética, y contexto. [12]

Del lado de la estética, similar a cuando se crea una UI, el enfoque se encuentra en el diseño gráfico de todos los elementos visuales que componen a un personaje. En esta parte se discute el lenguaje de la forma, gestos, colores, vestimenta, y rasgos. La estética del personaje es muy importante para un videojuego ya que un personaje, especialmente el personaje principal, similar a la portada de un libro, es la cara de un videojuego. Si este no tiene un buen diseño, puede llegar a alejar al público deseado o a dar un mensaje que no se quería transmitir.

Por otro lado, cuando se habla sobre el contexto del personaje, este toma en cuenta factores tales como su trasfondo, personalidad, metas, aspiraciones, relaciones interpersonales, percepción del mundo, y muchas otras cosas que permiten que el personaje "cobre vida" en la mente de los jugadores. Nuevamente, este aspecto también es de suma importancia porque los personajes deben ser consistentes dentro de su escritura, porque también se rigen dentro de los principios del diseño de videojuegos antes establecidos. Si el personaje no es consistente o presenta muchas contradicciones, este puede llegar a frustrar a los jugadores y no conectar con su historia, lo que los lleva a ignorarla o dejar el juego sin completar.

A pesar de todo lo dicho, es necesario recalcar que la complejidad de un personaje no significa que un personaje sea mejor, al mismo tiempo un personaje simple no significa un mal personaje, siempre es importante considerar el contexto del videojuego, especialmente haciendo uso del análisis UX para determinar al personaje o conjunto de personajes ideales que pertenecerán al juego- A diferencia de otros medios de *storytelling* como una película o un libro, un videojuego es interactivo, y si se tiene un acercamiento enfocado al jugador/usuario es necesario hacerle cambios a los personajes para que los jugadores disfruten de mejor manera el juego resultante.

4.15.2. Definición de caracterización de personajes

La caracterización de personajes es un proceso fundamental en la creación y presentación de personajes dentro de una obra, ya sea teatral, cinematográfica o de cualquier otra forma narrativa. Este proceso es clave para que el público logre comprender y apreciar la profundidad de los personajes, ya que involucra la construcción de figuras complejas y realistas que establecen una conexión emocional con la audiencia. Para lograrlo, actores y directores colaboran en desarrollar personajes con gran detalle y autenticidad. [57]

La caracterización se construye mediante la combinación de elementos físicos, psicológicos y emocionales. Los actores investigan y exploran las características internas de los personajes, como su personalidad, motivaciones, historias personales y las relaciones que mantienen con otros personajes. Esta investigación es esencial para que los actores puedan interpretar de manera creíble y profunda a sus personajes en escena.

Los aspectos físicos también juegan un papel crucial en el proceso de caracterización. El vestuario, el maquillaje y los peinados son herramientas importantes que ayudan a dar vida a los personajes y a transmitir su identidad de manera visual. Además, trabajar en detalles como gestos, posturas y movimientos corporales, que no solo reflejan la apariencia del personaje, sino también sus emociones y estado mental.

Otro aspecto relevante es la creación de la voz y el acento del personaje. La forma en que un personaje habla, incluyendo su tono, ritmo y lenguaje, es crucial para hacerlo más creíble y auténtico. [57]

En el caso de los videojuegos, las animaciones, es decir los movimientos de los modelos también pueden brindar caracterización a los personajes. Desde la manera en que caminan, pueden generar visualmente una proyección de emociones de felicidad, ánimo etc. Los movimientos corporales también indican que sucede con el personaje. [57]

4.15.3. ¿Cómo la animación ayuda a la caracterización de personajes?

La animación 3D ha revolucionado la forma en que se cuentan historias, especialmente en el ámbito de los videojuegos. Esta técnica permite que las experiencias narrativas sean más inmersivas, así como también la caracterización de este, haciendo que el jugador se sienta parte activa de la historia. En los videojuegos creados con animación 3D, el usuario no solo observa, sino que interactúa con la trama, convirtiéndose en el protagonista de una película interactiva. [135]

En efecto, la animación juega un papel crucial en la caracterización de personajes en los videojuegos. A través de la animación, los personajes cobran vida de una manera que complementa su diseño visual, personalidad y rol en la historia. Algunas características de cómo estas pueden ayudar son:

• Expresión de emociones y personalidad: Las animaciones faciales y corporales permiten que los personajes expresen una amplia gama de emociones, como felicidad, tristeza, miedo o ira. Esto ayuda a que los jugadores se conecten emocionalmente con los personajes. Los movimientos corporales también refuerzan la personalidad de un personaje. Por ejemplo, un héroe confiado podría caminar con pasos firmes y postura erguida, mientras que un villano furtivo podría moverse de manera sinuosa o cautelosa. [55]

• Estilo de movimiento único: La forma en que un personaje se mueve, cómo corre, salta, ataca o interactúa con el entorno puede contar mucho sobre quién es. Un personaje torpe o cómico tendrá animaciones más exageradas y descoordinadas, mientras que un personaje ágil tendrá movimientos fluidos y precisos.

• Refuerzo de habilidades y características: Las animaciones de combate o de uso de habilidades ayudan a reflejar las capacidades del personaje. Por ejemplo, un mago que utiliza hechizos podría tener animaciones elegantes y místicas, mientras que un guerrero con fuerza bruta podría tener movimientos más contundentes y agresivos.

• Contexto narrativo: Las escenas animadas o cinemáticas permiten que los personajes interactúen entre sí, revelando aspectos clave de su historia, relaciones y motivaciones. Las animaciones en estas escenas ayudan a contar la historia de manera más vívida y a desarrollar los personajes.

• Comportamiento no verbal: A veces, lo que un personaje no dice es tan importante como lo que dice. Las pequeñas animaciones o gestos, como una mirada desconfiada, un suspiro o una postura de defensa, pueden añadir capas a la personalidad del personaje y transmitir información que el diálogo no cubre.

• Inmersión y realismo: La animación contribuye a la sensación de realismo e inmersión, haciendo que los personajes se sientan vivos y reactivos al mundo que los rodea. Esto aumenta la conexión del jugador con los personajes y el mundo del juego. [133]

Por ello, la animación es un componente fundamental en la caracterización de personajes, ya que refuerza visualmente su personalidad, emociones, habilidades y su rol en la historia, haciendo que se sientan más auténticos y creíbles dentro del juego.

4.15.4. Ejemplos de personajes de videojuegos con buena caracterización

Las animaciones son fundamentales para la caracterización de personajes en los videojuegos, ya que permiten transmitir emociones, personalidades y características distintivas a través de acciones físicas. Los gestos, posturas y maneras de moverse pueden ofrecer a los jugadores una comprensión intuitiva de un personaje, complementando otros aspectos como el diseño visual y el diálogo.

Un buen uso de la animación no solo hace que los personajes se sientan más realistas, sino que también añade capas de profundidad a su caracterización. Por otro lado, existen las expresiones faciales y corporales. Las animaciones que capturan las emociones y expresiones de los personajes ayudan a los jugadores a conectarse emocionalmente con ellos. Esto puede incluir gestos, movimientos de la cabeza o cambios en la postura que reflejan la personalidad o el estado emocional del personaje. [33]

Esto es particularmente evidente en personajes que, a través de su forma de moverse o interactuar con el entorno, revelan rasgos específicos como vulnerabilidad, fuerza o agilidad. A continuación, se presentan algunos ejemplos de personajes de videojuegos cuya caracterización ha sido reforzada por la calidad y la coherencia de sus animaciones:

1. Spider-Man - Marvel's Spider-Man (2018)

• Animaciones de movimiento: Las animaciones de balanceo en la ciudad son fundamentales para caracterizar a Spider-Man como ágil y fluido. El movimiento es rápido, pero también tiene una ligereza que refleja su habilidad como héroe. Además, sus animaciones de combate muestran un estilo acrobático y flexible, haciendo que el jugador sienta que está controlando a alguien extremadamente hábil, pero con un toque juvenil y relajado.

• Caracterización: Las animaciones no solo muestran sus poderes, sino que también destacan su confianza y experiencia como superhéroe. [85]

2. Link - The Legend of Zelda: Breath of the Wild

• Animaciones de movimiento: En este juego, las animaciones de Link, como trepar montañas, esquivar ataques o correr, transmiten una sensación de fragilidad y humanidad. Su animación al escalar es lenta y cansada, mostrando esfuerzo, lo cual refuerza la idea de que Link, aunque valiente, es un héroe con limitaciones físicas.

• Caracterización: Las animaciones sutiles como la respiración pesada después de correr largas distancias o el temblor al colgarse de los acantilados hacen que el jugador sienta que Link está constantemente desafiando sus propios límites. [75]

3. Lara Croft - Tomb Raider (2013)

• Animaciones de movimiento: Las animaciones de movimiento de Lara Croft, especialmente las de sus interacciones con el entorno, son clave para mostrar su vulnerabilidad y evolución. Las animaciones de caída, saltos difíciles y sus movimientos cuando está herida transmiten a la perfección su lucha por sobrevivir.

• Caracterización: Sus animaciones de movimiento están diseñadas para mostrar su transformación de una arqueóloga inexperta a una sobreviviente decidida. A través de su movimiento, el jugador puede sentir su crecimiento tanto física como emocionalmente.

4. Kratos - God of War (2018)

• Animaciones de movimiento: Las animaciones de Kratos al luchar son pesadas y llenas de fuerza, lo que refuerza su carácter violento y poderoso. Sus ataques tienen un peso significativo, y su forma de moverse, lenta pero precisa, muestra su experiencia y fuerza física. [75]

• Caracterización: Sus movimientos reflejan su carácter: calculado, controlado, y lleno de ira contenida. Las animaciones transmiten mucho sobre su estado emocional, especialmente cuando interactúa con su hijo, Atreus. [46]

5. Geralt de Rivia - The Witcher 3: Wild Hunt

• Animaciones de movimiento: Las animaciones de combate de Geralt, como el uso de espadas y magia, son fluidas y detalladas, mostrando su destreza como cazador de monstruos. Además, sus gestos sutiles en momentos tranquilos, como encender una hoguera o preparar pociones, le dan vida al personaje más allá de las batallas.

• Caracterización: Geralt se mueve de manera eficiente y calculada, lo que coincide con su naturaleza de cazador veterano y experto en su oficio. Las animaciones sutiles también muestran su actitud pragmática y a menudo fría, pero también su lado reflexivo. [75]

Estos personajes están magistralmente caracterizados a través de sus movimientos, que no solo aportan realismo, sino que también transmiten sus emociones, experiencia y personalidad. [75]

4.16. Animación y jugabilidad

4.16.1. ¿Cómo las animaciones afectan la jugabilidad?

Las animaciones en los videojuegos son un componente esencial que influye directamente en la jugabilidad de diversas maneras. A continuación, se presentan los aspectos más relevantes sobre cómo las animaciones afectan la experiencia del jugador:

Inmersión y realismo

La inmersión y el realismo en los videojuegos se pueden potenciar significativamente a través de animaciones bien diseñadas. Algunas maneras en que las animaciones contribuyen a estos aspectos, es en el movimiento natural. Las animaciones fluidas y naturales hacen que los personajes y objetos se sientan más reales. Esto incluye no solo los movimientos de caminar o correr, sino también cómo interactúan con el entorno, como saltar, agacharse o reaccionar a impactos.

La interacción dinámica en los videojuegos es un factor clave para aumentar la inmersión del jugador. Las animaciones que responden a sus acciones o a eventos dentro del juego generan una experiencia más auténtica. Un ejemplo común es cuando el jugador dispara un arma; una animación que represente de manera convincente el retroceso del arma y el movimiento del personaje contribuye a una sensación de realismo, mejorando la experiencia de juego. [33]

Por otro lado, un entorno vivo también juega un papel importante en la inmersión. Animar elementos del entorno, como árboles que se balancean con el viento o luces que parpadean, ayuda a crear una atmósfera más dinámica y vibrante. Este tipo de detalles hacen que el jugador perciba el mundo del juego como un espacio real y activo, generando una mayor conexión con el entorno virtual.

Las cinemáticas y la narrativa son otra herramienta eficaz para contar historias dentro de los videojuegos. Las animaciones en escenas cinemáticas no solo ayudan a avanzar la trama, sino que también permiten que el jugador se sienta parte de la historia. De este modo, el uso de animaciones cuidadosamente elaboradas en secuencias narrativas puede profundizar la inmersión y el compromiso emocional del jugador con el juego. [33]

El feedback visual es fundamental para la comprensión del estado del juego. Animaciones que

brindan retroalimentación, como un efecto visual al recoger un objeto o al completar una tarea, permiten al jugador recibir información clara sobre su progreso. Este tipo de animaciones, aunque sutiles, aumentan el sentido de logro y contribuyen a una mayor interacción y satisfacción del jugador.[33]

Feedback visual

El feedback visual en los videojuegos es fundamental, ya que permite a los jugadores comprender las consecuencias de sus acciones y fortalecer su conexión con el entorno del juego. Existen diversas formas en que las animaciones contribuyen a este feedback, proporcionando una respuesta visual clara e inmediata. Una de estas formas es a través de las reacciones de los personajes. Cuando un jugador realiza una acción, como golpear a un enemigo o recoger un objeto, una animación de reacción, como un enemigo tambaleándose o el personaje celebrando, ofrece una respuesta inmediata. Este tipo de animaciones refuerzan la idea de que la acción del jugador ha tenido un impacto real en el mundo del juego, mejorando la sensación de interacción. [151]

Los efectos visuales también juegan un papel importante en el feedback visual. Animaciones que acompañan acciones específicas, como explosiones, destellos de luz o partículas en movimiento, añaden un nivel adicional de información visual. Por ejemplo, al disparar un arma, la combinación de una animación de retroceso con efectos tomar el arma, respiración, entre otros, hace que la acción se sienta más satisfactoria y realista, aumentando la inmersión.[151]

Las animaciones también son esenciales en la interacción con objetos. Acciones como abrir una puerta, presionar un botón o recoger un ítem se ven reforzadas por animaciones que confirman el éxito de la interacción. Estos movimientos suaves, acompañados de efectos visuales y sonoros, ofrecen al jugador un feedback inmediato y satisfactorio sobre su acción. El estado de salud y condiciones del personaje es otro elemento que puede comunicarse mediante animaciones. Por ejemplo, el parpadeo de la pantalla al recibir daño o una animación de curación proporcionan información clara al jugador sobre su estado actual, permitiéndole tomar decisiones rápidas y estratégicas.[151]

El cierre de acciones también es relevante en el feedback visual. Animaciones que indican la finalización de una tarea, como un gesto de victoria o un efecto de desenfoque al completar un nivel, ayudan a los jugadores a entender que han alcanzado con éxito sus objetivos, reforzando el sentido de logro.

Por último, la consistencia y claridad en el uso de animaciones es clave para crear un lenguaje visual coherente dentro del juego. Si las animaciones son predecibles y los jugadores pueden anticipar cómo reaccionan los personajes u objetos, se sentirán más en control y seguros en su experiencia de juego. [151]

Por ello, la implementación de un feedback visual efectivo a través de animaciones no solo mejora la jugabilidad, sino que también aumenta la satisfacción del jugador al proporcionar una experiencia más rica y envolvente.

Construcción de personajes

La construcción de personajes en los videojuegos se ve profundamente enriquecida por el uso de animaciones. Estas no solo dan vida a los personajes, sino que también ayudan a contar su historia, definir su personalidad y mejorar la inmersión del jugador en el mundo del juego. A continuación, se destacan algunos aspectos clave sobre cómo las animaciones contribuyen a este proceso.

Otro aspecto relevante es cómo las animaciones reflejan la personalidad y el estilo de un personaje. Un guerrero fuerte y robusto puede tener movimientos pesados y contundentes, mientras que un

personaje más ágil y astuto podría presentar animaciones rápidas y fluidas. Este contraste ayuda a los jugadores a identificar rápidamente la naturaleza de los personajes, facilitando su comprensión del papel que juegan en la historia.

En cuanto a las interacciones sociales, las animaciones también juegan un papel importante. Las reacciones de los personajes durante conversaciones o al interactuar con otros refuerzan las relaciones y la dinámica entre ellos. Esto aporta credibilidad a las escenas y mejora la inmersión, haciendo que las interacciones entre personajes sean más naturales y envolventes. El desarrollo de arcos narrativos es otro elemento que se ve potenciado por las animaciones. A medida que los personajes evolucionan a lo largo de la historia, sus animaciones pueden reflejar estos cambios. Por ejemplo, un personaje que comienza siendo tímido y luego se vuelve valiente puede mostrar una evolución en su postura y en la forma en que se mueve, evidenciando su transformación a través de la narrativa. [33]

La identidad visual de un personaje se ve fortalecida por sus animaciones, ya que movimientos únicos y patrones de comportamiento característicos permiten que un personaje sea fácilmente reconocible y memorable, incluso en juegos con numerosos personajes. Esta distinción contribuye a la creación de personajes icónicos que dejan una impresión duradera en los jugadores. Además, la conexión con el jugador se intensifica cuando los personajes se mueven y reaccionan de manera convincente, un aspecto crucial en juegos de rol o narrativos donde los personajes son el núcleo de la experiencia. Animaciones efectivas permiten a los jugadores sentirse más involucrados con los personajes, enriqueciendo así la experiencia general del juego.

Finalmente, las acciones y habilidades de los personajes son otro punto clave. Animaciones que muestran habilidades especiales o ataques pueden hacer que las acciones se sientan más satisfactorias. Por ejemplo, una animación espectacular durante un ataque puede hacer que el jugador se sienta poderoso y efectivo, mejorando la experiencia de juego.[33]

Por ello, la integración efectiva de las animaciones en la construcción de personajes puede llevar la experiencia del jugador a un nivel superior, haciendo que los personajes sean más reales, memorables y significativos.

Fluidez y Dinámica del Juego La fluidez y la dinámica del juego son esenciales para mantener a los jugadores comprometidos y garantizar una experiencia satisfactoria. Las animaciones juegan un papel crucial en estos aspectos, El control responsivo también es un aspecto importante. Animaciones que responden de manera inmediata a las entradas del jugador, como los saltos, ataques o movimientos, crean una sensación de fluidez y precisión en el control del personaje. [151]

La cadencia y el ritmo de las animaciones pueden influir directamente en la dinámica del juego. Por ejemplo, en un juego de acción, animaciones rápidas generan una sensación de urgencia e intensidad, mientras que, en juegos más pausados, animaciones lentas pueden crear una atmósfera más reflexiva y calmada. Este uso del ritmo en las animaciones ayuda a establecer el tono del juego y la forma en que los jugadores lo experimentan. Los efectos de impacto son fundamentales para la dinámica del juego, ya que las animaciones que acompañan ataques y colisiones ofrecen un feedback visual claro sobre las acciones de los jugadores, lo que no solo ayuda a entender lo que ocurre, sino que también hace que las interacciones sean más satisfactorias. Además, las animaciones que reflejan la vida en el entorno, como NPCs caminando y elementos naturales reaccionando al viento, añaden dinamismo y autenticidad al mundo del juego, creando una experiencia más envolvente y rica. [151]

Otro aspecto importante son las acciones encadenadas. Animaciones que permiten a los jugadores encadenar acciones de manera fluida, como pasar de un ataque a un combo o de un salto a un giro, fomentan un estilo de juego más dinámico. Esto les permite a los jugadores experimentar con diferentes estrategias y tácticas, lo que enriquece la jugabilidad y aporta profundidad a la experiencia del juego.

Las interacciones contextuales también se ven mejoradas por las animaciones. Cuando un personaje reacciona de manera distinta según el contexto, como agacharse para esquivar un disparo o

interactuar de manera diferente con el entorno, mejora la sensación de realismo e inmersión. Esto hace que los jugadores se sientan más conectados con el juego, ya que las reacciones del personaje se alinean con las situaciones que enfrentan.[151]

Por último, el feedback visual inmediato que proporcionan las animaciones es crucial para mejorar la fluidez de la experiencia de juego. Animaciones que ofrecen retroalimentación visual clara y rápida, como un destello al recibir daño o un efecto de sonido al completar una acción, permiten a los jugadores entender las consecuencias de sus acciones de manera instantánea, manteniendo el ritmo del juego sin pausas innecesarias. Por eso, la implementación efectiva de animaciones mejora tanto la fluidez como la dinámica del juego, haciendo que la experiencia sea más entretenida y envolvente para los jugadores. [151]

Mecánicas de juego

Las animaciones son fundamentales para implementar y mejorar las mecánicas de juego en los videojuegos, ya que aportan vida a las acciones de los personajes y hacen que las interacciones sean más fluidas y comprensibles. [33]

Las animaciones son fundamentales para representar acciones en los videojuegos, ofreciendo una representación visual clara de lo que los personajes pueden hacer, como saltar, atacar o ejecutar habilidades especiales. Esto facilita a los jugadores la comprensión de las mecánicas del juego y proporciona una referencia visual inmediata para interactuar con el mundo del juego. Además, las animaciones ofrecen feedback inmediato; por ejemplo, tras un ataque, la animación de impacto refuerza la sensación de éxito y efectividad de la acción del jugador, mejorando así la experiencia de juego al informar instantáneamente sobre la efectividad de sus acciones.

Algunas mecánicas de juego dependen de condiciones de activación vinculadas a animaciones. Por ejemplo, una habilidad especial que solo puede ejecutarse después de una animación de carga introduce un elemento estratégico, ya que los jugadores deben prestar atención al momento y lugar adecuado para ejecutar ciertas habilidades.[33] Las animaciones también mejoran las interacciones contextuales. En situaciones donde un personaje entra en un área específica o interactúa con un objeto, las animaciones activadas en función del contexto permiten a los jugadores experimentar diferentes mecánicas de manera más inmersiva, como al abrir puertas o recoger objetos.

El encadenamiento de acciones es otra área en la que las animaciones son esenciales. Permiten que los jugadores combinen varias acciones sin interrupciones, como en un juego de combate donde un jugador puede enlazar un ataque, esquivar y luego realizar una habilidad especial. Esto crea combos fluidos y dinámicos, lo que hace que la jugabilidad sea más emocionante y rápida. La consistencia en el juego es clave para una experiencia clara y predecible. Si las animaciones para ciertas mecánicas son coherentes a lo largo del juego, los jugadores pueden anticipar cómo funcionarán sus acciones en diferentes contextos. Esto les permite dominar más fácilmente las mecánicas y optimizar sus estrategias.

Las animaciones permiten la creación de mecánicas de juego innovadoras. Por ejemplo, un juego que manipule el tiempo a través de animaciones puede permitir que los jugadores ralenticen o aceleren el tiempo, afectando de manera creativa la jugabilidad y brindando nuevas formas de interactuar con el mundo del juego. Las reacciones de personajes a las acciones del jugador, como cuando sufren daño o interactúan con el entorno, también son facilitadas por animaciones. Estas reacciones mejoran la inmersión y ayudan a los jugadores a entender mejor las consecuencias de sus decisiones dentro del juego.

Por último, la variabilidad en el juego es posible gracias a animaciones alternativas. Tener diferentes animaciones para la misma acción en distintos contextos ofrece más opciones y estrategias a los jugadores, lo que enriquece la jugabilidad y añade variedad a la forma en que se abordan los

desafíos. Por ello, las animaciones son un componente esencial en el diseño de mecánicas de juego, mejorando la jugabilidad, la inmersión y la satisfacción de los jugadores.[33]

Identidad y estilo visual

La identidad y el estilo visual de un videojuego son elementos esenciales que definen la experiencia del jugador y contribuyen a que un juego sea memorable y atractivo. Las animaciones juegan un papel clave en la creación y el mantenimiento de esta identidad visual, aportando coherencia y profundidad al estilo del juego. A continuación, se destacan varias formas en que las animaciones contribuyen a este aspecto.

En primer lugar, las animaciones aseguran la consistencia estilística a lo largo del juego. Si los personajes, objetos y entornos comparten una misma estética animada, el juego resulta visualmente más armónico y atractivo. Esto es crucial para mantener una experiencia inmersiva, ya que una inconsistencia en las animaciones podría romper la cohesión visual. [28]

Además, que el diseño de personajes también se ve altamente influenciados movimientos. Un personaje cómico, por ejemplo, podría tener animaciones más exageradas y dinámicas, mientras que un personaje más serio mostraría movimientos más controlados y sutiles, ayudando a comunicar su esencia sin necesidad de palabras. Las animaciones refuerzan el estilo de arte general del juego. Ya sea que se trate de gráficos 2D, 3D, pixel art o incluso stop-motion, la manera en que los personajes y objetos son animados debe complementar y realzar el estilo artístico elegido. Esto crea una experiencia visual más coherente y envolvente, que facilita la inmersión del jugador.

La cultura y temática del juego también se ven enriquecidas por las animaciones. Un juego ambientado en una cultura específica puede incorporar movimientos, gestos y posturas que reflejen los valores o tradiciones de esa cultura, añadiendo autenticidad y profundidad al juego.

A nivel narrativo, las animaciones contribuyen a la narrativa visual. Los movimientos y las interacciones de los personajes con su entorno cuentan una historia en sí mismos, comunicando emociones, intenciones y conflictos sin necesidad de diálogos. Esto permite que los jugadores interpreten la historia de manera más intuitiva y emocional.

La animación puede incluso contribuir a la identidad de marca. Un estilo de animación distintivo hace que un juego sea fácilmente reconocible y recordado por los jugadores, lo que puede aumentar su popularidad y establecer una conexión más fuerte entre el juego y su audiencia. [23]

Por último, las animaciones que transmiten emociones y sentimientos profundos permiten a los jugadores conectar con los personajes y la historia en un nivel más personal. Esto crea una experiencia más significativa y duradera, generando una conexión emocional que deja una impresión positiva en los jugadores. [28]

Por ende, las animaciones son fundamentales para la jugabilidad en los videojuegos, afectando la inmersión, el feedback, la narrativa, la fluidez y las mecánicas del juego. La calidad de las animaciones puede determinar en gran medida cómo los jugadores experimentan y disfrutan de un juego. A medida que la tecnología de animación continúa evolucionando, su importancia en el diseño de juegos sólo aumentará.

4.16.2. Variables que pueden influir o afectar la jugabilidad

Las animaciones y transiciones en videojuegos tienen un impacto significativo en la jugabilidad, afectando tanto el tiempo de respuesta del jugador como la fluidez de las interacciones dentro del juego. Una de las variables clave es la duración de las animaciones. La velocidad de estas es crucial; si son demasiado largas o lentas, pueden ralentizar la jugabilidad y frustrar a los jugadores. Esto

se hace evidente en juegos de acción, donde las animaciones de ataque o recuperación que tardan demasiado pueden interrumpir el ritmo del juego. Además, la sincronización de las acciones es esencial. Animaciones bien cronometradas permiten una mejor percepción de control, mientras que tiempos prolongados o imprecisos pueden generar una sensación de control reducido. Por ejemplo, en juegos de ritmo rápido, las animaciones de salto o esquiva deben ejecutarse rápidamente para mantener la dinámica del juego. [102]

Otro aspecto importante son las transiciones entre animaciones. La fluidez en la transición entre diferentes estados de animación, como pasar de caminar a correr o de correr a saltar, es fundamental. Si las transiciones son abruptas o rígidas, pueden crear una sensación antinatural que afecta la inmersión del jugador. Implementar transiciones suaves ayuda a mantener la coherencia de los movimientos y la experiencia general. Además, en algunos juegos, es vital que estas transiciones se realicen mediante interpolaciones suaves, lo que afecta la percepción de control y realismo. Si son torpes o inexistentes, el jugador podría sentir que el personaje es difícil de manejar.

La inclusión de frames de invulnerabilidad también es relevante. En juegos de combate o plataformas, estas animaciones permiten que el jugador no reciba daño durante momentos específicos, como al esquivar o saltar. La gestión de estos tiempos y su coherencia con las animaciones impactan directamente en la jugabilidad estratégica, ya que los jugadores deben ser conscientes de cuándo pueden actuar sin riesgo. [83]

Finalmente, la carga cognitiva y los tiempos de respuesta son factores a tener en cuenta. Las animaciones demasiado rápidas o complejas pueden sobrecargar la percepción visual del jugador, mientras que las que son demasiado lentas pueden provocar aburrimiento o desconexión. Encontrar un equilibrio adecuado es esencial para mantener la atención del jugador sin causar fatiga.

Como es notorio, tanto la duración de las animaciones como las transiciones tienen un efecto profundo en el flujo, la estrategia y la percepción del jugador dentro de los videojuegos. Por lo tanto, es fundamental que estas se equilibren cuidadosamente para no afectar negativamente la jugabilidad y garantizar una experiencia más fluida y satisfactoria.

4.17. Herramientas y tecnologías para el desarrollo de videojuegos

En la actualidad, existe una gran variedad de programas tanto de modelado y animación en 3D que permiten crear objetos tridimensionales y generar imágenes fotorrealistas. Sin embargo, son pocos los que combinan tres características esenciales que resultan muy atractivas para los usuarios: ocupan poco espacio en el disco duro al ser instalados, requieren escasos recursos del ordenador y son de código abierto, lo que significa que su código fuente está disponible para el público; permitiendo a cualquier persona ver, modificar y distribuir el software.

4.17.1. Unity

El desarrollo de videojuegos ha experimentado un avance significativo en las últimas décadas, y el motor de desarrollo Unity ha emergido como una de las herramientas más utilizadas en la industria. Su versatilidad y accesibilidad lo han convertido en una opción preferida tanto por desarrolladores independientes como por estudios profesionales. A continuación, se exploran sus principales características, ventajas y herramientas complementarias.

Características de Unity

Unity destaca por su capacidad multiplataforma, lo que permite a los desarrolladores crear videojuegos que pueden ser exportados a una amplia variedad de plataformas, tales como PC, consolas, dispositivos móviles y aplicaciones web. Esta capacidad facilita la distribución de los juegos a un público más diverso. Además, Unity cuenta con una interfaz de usuario intuitiva, diseñada para que los nuevos desarrolladores puedan adaptarse rápidamente al entorno de trabajo. Esta característica, junto con una comunidad de usuarios activa y numerosos recursos disponibles en línea, como tutoriales y foros, permite a los desarrolladores resolver problemas y mejorar sus habilidades de manera efectiva [136, 63].

Ventajas de usar Unity

Unity no solo permite el desarrollo multiplataforma, sino que también destaca por su integración de herramientas de colaboración en la nube, como Unity Collaborate, que permite a los equipos de desarrollo trabajar de manera remota en tiempo real. Esto es fundamental en proyectos modernos, donde los equipos pueden estar distribuidos en diferentes zonas geográficas. La integración de sistemas de control de versiones también asegura que los desarrolladores puedan rastrear y revertir cambios fácilmente, lo que es esencial para evitar errores críticos durante el desarrollo [87].

Unity también destaca por su compatibilidad con tecnologías de realidad virtual y aumentada, lo que lo convierte en una opción ideal para el desarrollo de experiencias inmersivas. La facilidad con la que los desarrolladores pueden integrar kits de desarrollo para VR (Oculus, HTC Vive) y AR (ARKit, ARCore) ha permitido que Unity sea una opción líder en este mercado emergente [136].

Herramientas complementarias para el desarrollo de videojuegos

Si bien Unity es una herramienta poderosa, existen otras plataformas de desarrollo que también destacan en la industria. Unreal Engine, por ejemplo, es conocido por su motor gráfico de alta calidad y su sistema *Blueprint*, que permite la programación visual sin necesidad de escribir código. Esto lo convierte en una opción ideal para proyectos complejos y de gran envergadura [87, 125]. Por otro lado, GameMaker Studio está más orientado a la creación de juegos en 2D y ofrece una interfaz basada en el sistema de arrastrar y soltar, lo que facilita su uso para desarrolladores con poca experiencia en programación [63]. Finalmente, RPG Maker es una opción popular para el desarrollo de juegos de rol (RPG), ya que permite la creación de eventos predefinidos sin necesidad de programar, lo que lo convierte en una herramienta accesible para usuarios con conocimientos técnicos limitados [68, 63].

En resumen, Unity se posiciona como uno de los motores de desarrollo más versátiles y accesibles en la actualidad, adecuado tanto para principiantes como para desarrolladores avanzados. Su capacidad multiplataforma, interfaz intuitiva y recursos complementarios lo convierten en una opción atractiva dentro del desarrollo de videojuegos.

4.17.2. Implementación de IA en Unity

La inteligencia artificial (IA) en videojuegos ha evolucionado de manera significativa, permitiendo crear experiencias más inmersivas e interactivas. Unity ofrece herramientas robustas para la implementación de sistemas de IA, tales como navegación, decisiones de comportamiento y sistemas complejos basados en redes neuronales o algoritmos de aprendizaje automático. La facilidad con la que Unity permite integrar estas tecnologías ha hecho que sea una plataforma popular para el desarrollo de videojuegos que requieren interacciones avanzadas con el entorno o con otros personajes no jugadores (NPCs) [74].

Sistemas de navegación y pathfinding

Uno de los pilares de la IA en Unity es el sistema de navegación y pathfinding, que utiliza mallas de navegación (NavMesh). Estas mallas permiten a los personajes NPC moverse dentro de un escenario con obstáculos y terrenos complejos de forma eficiente. Unity proporciona herramientas integradas para generar NavMesh en tiempo real y ajustar la navegación a medida que el entorno cambia. Esto es esencial para crear entornos dinámicos donde los NPCs deben reaccionar a cambios inesperados, como la destrucción de objetos o la aparición de nuevos obstáculos [19].

Además, el algoritmo A (A-star)* es ampliamente utilizado en Unity para encontrar rutas óptimas en escenarios complejos. Este algoritmo se basa en buscar el camino más corto de un punto a otro, evaluando nodos y caminos de acuerdo a su costo. Unity permite una implementación sencilla de este tipo de algoritmos gracias a su compatibilidad con lenguajes de programación como C# y su integración con sistemas de mallas y físicas [74].

Comportamientos complejos y toma de decisiones

Unity facilita la creación de comportamientos complejos para los NPCs utilizando herramientas como behavior trees y finite state machines (FSM). Los árboles de comportamiento son utilizados para definir de manera jerárquica las decisiones que toma un NPC en función de las condiciones del entorno. Este enfoque modular permite a los desarrolladores crear interacciones realistas y dinámicas sin requerir un código excesivamente complejo. Unity ofrece plugins y frameworks que simplifican la creación de estos sistemas, permitiendo una personalización y optimización profunda [19].

Otra herramienta relevante son las finite state machines, que permiten gestionar las transiciones entre diferentes estados de los personajes en respuesta a estímulos. Por ejemplo, un NPC puede pasar de un estado de patrullaje a uno de ataque si detecta la presencia del jugador. Unity soporta la implementación de estas máquinas de estados mediante scripting en C#, lo que brinda flexibilidad para desarrollar IA reactiva e inteligente [74].

Aprendizaje automático en videojuegos

Con el avance del aprendizaje automático, Unity ha comenzado a incorporar herramientas para integrar redes neuronales y otros algoritmos de IA más avanzados en videojuegos. El framework ML-Agents de Unity permite entrenar agentes con técnicas de aprendizaje reforzado para que aprendan comportamientos complejos en entornos simulados. Esta herramienta ha abierto nuevas posibilidades para el desarrollo de videojuegos donde los NPCs pueden adaptarse y aprender del comportamiento del jugador en tiempo real, generando interacciones más profundas y dinámicas [74].

El uso de aprendizaje automático también se extiende a la creación de NPCs más "humanos", con comportamientos menos predecibles y más variados. A través de la implementación de redes neuronales, los desarrolladores pueden crear NPCs que ajustan su estrategia en tiempo real, lo que añade un nivel de dificultad y realismo a los videojuegos [19].

Integración de IA con sistemas físicos y gráficos

La IA en Unity no solo se limita a decisiones o navegación, sino que también interactúa con otros sistemas, como el motor gráfico y el sistema de físicas. Los personajes controlados por IA pueden reaccionar de manera realista a las físicas del entorno, como las colisiones o el impacto de objetos. Unity permite sincronizar las acciones de la IA con animaciones y efectos visuales en tiempo real,

creando una experiencia cohesiva entre la inteligencia del personaje y la respuesta visual en pantalla [74].

4.17.3. Blender

Blender es un software gratuito y de código abierto que se utiliza para generar contenido en 3D. Esta herramienta todo en uno combina funciones de modelado, animación, simulación, edición de vídeo, creación de videojuegos y más. Su capacidad para integrarse con otras aplicaciones lo hace muy atractivo para diversas industrias, incluyendo la animación, el cine, el diseño de videojuegos y la visualización arquitectónica. [37]

¿Por qué utilizar *Blender*?

De hecho, hay algunas razones muy importantes por las que *Blender* fue la opción más adecuada para este proyecto. La primera es que es un *software* gratuito con código abierto y, en general, muy disponible, por lo que se puede trabajar en varios proyectos a la vez sin incurrir en costes adicionales. Esto unifica una inmensa cantidad de herramientas en una sola interfaz, lo que lo convierte en un *software* todo en uno para funciones de modelado avanzado de polígonos, escultura, texturizado y animación sin necesidad de cambiar nunca de *software*. En ese sentido, es un flujo de trabajo eficaz. Además, *Blender* es parte de los conjuntos de herramientas estándar, trabajando en la dirección del modelado 3D con respecto a la flexibilidad y el desarrollo activo. La comunidad activa de usuarios y desarrolladores garantiza una mejora continua junto con el soporte para nuevas tecnologías. Finalmente, la compatibilidad con motores de juegos, como *Unity* y *Unreal*, y la capacidad de exportar en muchos formatos convierten a *Blender* en una solución seria y profesional que se puede utilizar para manejar la calidad y el rendimiento de acuerdo con las necesidades específicas de desarrollo en videojuegos.[64]

Principales aplicaciones

Blender cuenta con diversas aplicaciones ya que es un programa versátil que encuentra uso en múltiples campos creativos y técnicos. A continuación, se presentan algunas de sus principales aplicaciones:

• Modelado 3D: Permite la creación de modelos tridimensionales de objetos, personajes y entornos, abarcando tanto el modelado poligonal (basado en vértices y aristas) como el modelado basado en curvas y superficies.

• Animación: Facilita la animación de personajes, la pixilación y la animación por fotogramas clave, así como el rigging para la configuración de esqueletos y animaciones físicas. Su sistema de armaduras y controladores ayuda a lograr movimientos realistas y dinámicos.

• Simulación: Ofrece diversos motores de simulación para crear efectos físicos realistas, como fluidos, humo, fuego, telas, partículas y cuerpos rígidos. • Creación de videojuegos: Integra un motor de videojuegos (Blender Game Engine - BGE) que permite desarrollar juegos interactivos directamente en el software.

• Realidad virtual y aumentada: Gracias a su capacidad de exportar en formatos compatibles con dispositivos de VR/AR, Blender se ha consolidado como una herramienta esencial en este campo emergente. [37]

Características destacadas

Blender es compatible con todos los sistemas operativos, lo que significa que puede ser usarlo sin inconvenientes en Mac, Windows o Linux. Características destacadas: • Herramientas potentes para renderizado, modelado y escultura. • Funciones avanzadas para animación y rigging. • Capacidad para diseñar y editar en un entorno 3D. • Edición de vídeo y efectos visuales (VFX), que incluyen seguimiento de movimiento, enmascaramiento y composición. • Herramientas de simulación que ofrecen efectos realistas. • Amplia API de Python para personalizar y desarrollar scripts. • Interfaz de usuario y diseño de ventanas personalizables, así como accesos directos ajustables. [130]

Auto Rig Pro

Auto-Rig Pro es una herramienta que sirve como vía para extender las funcionalidades de Blender; conocidos como Add-ons. Este es altamente reconocido para el rigging de personajes en Blender, diseñado para optimizar el proceso de creación de esqueletos y controladores. Este complemento se distingue por su algoritmo inteligente, que permite a los artistas 3D configurar rigs (esqueleto del personaje) con unos pocos clics, sin comprometer la precisión y el control necesarios para proyectos de animación de alta calidad. Su compatibilidad con plataformas de animación, como Mixamo, lo convierte en una herramienta versátil y eficiente. [60]

Los beneficios que se obtienen al utilizar Auto-Rig Pro son significativos. En primer lugar, su interfaz rápida y fácil de usar permite realizar un rigging a cualquier personaje en cuestión de minutos, lo que ahorra tiempo valioso en el flujo de trabajo del artista. Además, a pesar de ser un proceso automatizado, el rig resultante es de calidad profesional, listo para la animación y con una gran precisión en los movimientos. La herramienta también facilita la integración de animaciones externas, lo que permite incorporar recursos adicionales sin complicaciones. [60]

Con Auto-Rig Pro, los artistas no solo optimizan su flujo de trabajo, sino que también mejoran la calidad de sus proyectos, haciendo de esta herramienta una opción indispensable para aquellos que buscan agregar movimiento a sus personajes 3D de manera efectiva y eficiente. Al ser accesible de forma gratuita, permite que los usuarios comiencen a trabajar de inmediato, maximizando así su productividad y creatividad en el proceso de animación. Ver Figura 4.30. [60]

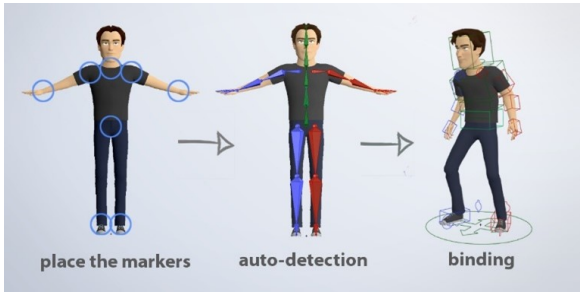

Figura 4.31: Rigging de personaje con Auto-Rig Pro

4.17.4. Mixamo

Mixamo es una herramienta creada por Mixamo Inc., una compañía fundada en 2008. Adobe Mixamo es una plataforma innovadora para la animación en 3D que permite a los usuarios crear y diseñar personajes tridimensionales para diversas aplicaciones, como películas, videojuegos y experiencias interactivas. Ofrece una amplia biblioteca de personajes 3D de alta calidad, completamente

texturizados, y animaciones de cuerpo completo obtenidas a través de la captura de movimiento de actores profesionales.[101]

Características

Una de las características más destacadas de Mixamo es su capacidad para cargar cualquier modelo 3D, al que se le aplica automáticamente un esqueleto humano, ajustándose según sus características específicas. Este proceso simplifica el rigging, permitiendo a los artistas centrarse en la animación y el diseño sin complicaciones técnicas.

Además, Mixamo facilita la creación y animación rápida de personajes, incluso para aquellos que no cuentan con un conocimiento profundo de animación. La plataforma ofrece un método ágil y personalizable, donde los usuarios pueden mezclar diferentes clips de movimiento de su biblioteca y redirigirlos a otros personajes en aplicaciones como Cinema 4D.

Finalmente, Mixamo permite exportar personajes y animaciones en múltiples formatos compatibles con plataformas como Unity, Blender, entre otros, lo que lo convierte en una herramienta valiosa para desarrolladores y animadores que buscan integrar personajes animados de manera eficiente en sus proyectos. [96]

4.17.5. Figma

Figma es un editor web de gráficos vectoriales desarrollado por Figma, Inc. y lanzado inicialmente en 2016. Esta herramienta es principalmente utilizada para el diseño y prototipado de interfaces gráficas y experiencias de usuario, permitiendo crear versiones semi funcionales de *frontend* de aplicaciones y páginas web. [56] La versatilidad de esta herramienta permite realizar cualquier tipo de diseño; desde diseñar la interfaz de una aplicación móvil o una aplicación web, hasta realizar presentaciones y *pitches* de proyectos. Gracias a esta versatilidad y, con un poco de creatividad, es posible diseñar el *Heads-up Display* (HUD) o pantalla de visualización frontal que se mostrará durante el juego, o diseñar todas las pantallas y menús necesarios para interactuar con y modificar datos. Puntos a favor del uso de esta herramienta es que es una aplicación gratis, y se puede trabajar desde un navegador o descargando su aplicación de escritorio o móvil. Además, permite trabajar colaborativamente con otros miembros a tiempo real.

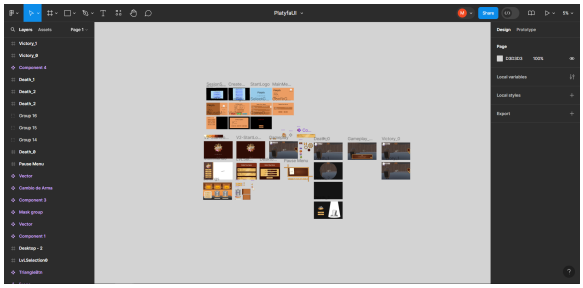

Figura 4.32: Interfaz gráfica de Figma

Productos

Figma tiene actualmente 3 productos, los cuales son:

- **Figma Design**: Utilizado para crear, compartir y testear el diseño de aplicaciones móviles,

páginas web, programas, y cualquier otro producto digital o experiencia. Permite a cualquier persona involucrada en el diseño contribuir y dar retroalimentación para realizar cambios en los productos y obtener mejores versiones de las mismas.

- **Figma Slides**: Permite crear presentaciones e integrar prototipos para exponerlos a personas de interés o al equipo. Estas pueden crearse de forma colaborativa con todas las herramientas que ofrece Figma.

- **FigJam**: son pizarrones virtuales que permiten colaborativamente aportar ideas durante la etapa de ideamineto de un diseño con texto, notas, imágenes y figuras.

4.18. Investigación de tecnologías *Frontend*

En el desarrollo de interfaces de usuario modernas, elegir la tecnología adecuada para el frontend es un factor crucial que afecta tanto la escalabilidad del proyecto como la experiencia del usuario. Entre las opciones más destacadas en el mercado se encuentran React, Vue.js y Angular, tres herramientas que han ganado popularidad por sus distintas ventajas y enfoques. A continuación, se presenta un análisis comparativo de estas tecnologías, evaluando sus características más relevantes, ventajas, desventajas y su aplicabilidad para proyectos web modernos.

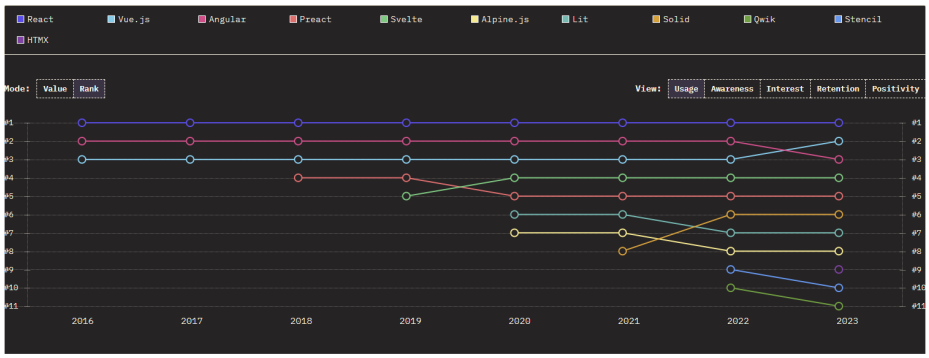

Figura 4.33: Top de tecnologías frontend a lo largo del tiempo

4.18.1. Características clave

React, desarrollado por Facebook, es una biblioteca que se centra en la creación de interfaces de usuario mediante componentes reutilizables y la gestión eficiente del estado de la aplicación a través de un **DOM Virtual**. Esto permite optimizar el rendimiento actualizando solo los elementos que han cambiado [121].

Vue.js, por otro lado, es un *framework* progresivo creado por Evan You que también se basa en la idea de componentes reutilizables, pero se destaca por su simplicidad y capacidad de adopción gradual. Es especialmente conocido por su curva de aprendizaje suave, lo que facilita su integración en proyectos de cualquier tamaño [118].

Finalmente, Angular, respaldado por Google, es un *framework* completo para aplicaciones web que utiliza TypeScript y una arquitectura basada en módulos. Ofrece una solución robusta para el desarrollo de aplicaciones de gran escala, con funcionalidades integradas como la inyección de dependencias y el enrutamiento dinámico [38].

4.18.2. Ventajas comparativas

Al evaluar las ventajas de cada tecnología, es importante considerar tanto su capacidad técnica como la facilidad de uso y el soporte de la comunidad. React destaca por su gran flexibilidad y el apoyo de una comunidad extensa, permitiendo una gran personalización a través de bibliotecas externas como Redux y React Router . Además, su enfoque en un flujo de datos unidireccional facilita la depuración y el manejo del estado [121].

Por otro lado, Vue.js es preferido por su enlace de datos bidireccional y su estructura ligera, lo que permite un desarrollo más rápido en proyectos pequeños y medianos sin sacrificar la escalabilidad . También cuenta con una documentación muy completa que facilita la curva de aprendizaje, haciéndolo accesible para desarrolladores con menos experiencia [118].

En cuanto a Angular, su arquitectura modular y su uso de TypeScript proporcionan una estructura sólida y bien organizada para proyectos grandes y complejos. Además, Angular ofrece un sistema robusto de inyección de dependencias, lo que permite una mejor gestión de los servicios y componentes en aplicaciones de gran escala [38].

4.18.3. Desventajas y retos

Cada tecnología también tiene sus desafíos. React, aunque muy popular, puede ser intimidante para desarrolladores nuevos debido a su curva de aprendizaje en torno a conceptos como JSX y el Virtual DOM [121]. Además, la rápida evolución de sus librerías y herramientas puede dejar la documentación obsoleta en poco tiempo.

Vue.js, aunque más fácil de aprender, presenta ciertas limitaciones en proyectos de gran escala. Su flexibilidad puede llevar a inconsistencias si no se siguen estrictamente las buenas prácticas de desarrollo, lo que podría complicar el mantenimiento en proyectos grandes [118].

Finalmente, Angular es conocido por tener una curva de aprendizaje más pronunciada debido a su complejidad, especialmente para desarrolladores que no están familiarizados con TypeScript. Además, es más verboso y requiere más configuración en comparación con React y Vue.js, lo que puede ralentizar el desarrollo en las etapas iniciales [38].

4.18.4. Popularidad y uso en el mercado laboral

En el mercado laboral de 2024 [14], React sigue siendo la opción más demandada por su versatilidad y el gran número de empresas que lo utilizan para construir aplicaciones web interactivas. Vue.js, aunque menos popular, ha ganado terreno en *startups* y proyectos más pequeños debido a su simplicidad [5]. Angular, por su parte, sigue siendo muy valorado en entornos empresariales donde la robustez y la estructura son primordiales para el desarrollo de aplicaciones de gran escala [49].

4.18.5. Conclusión

Tras analizar las ventajas, desventajas y la popularidad de React, Vue.js y Angular, se concluye que React es la tecnología más adecuada para proyectos que requieren un alto grado de interactividad y flexibilidad, como el desarrollo de un módulo web para *Platyfa*. Su capacidad para gestionar componentes dinámicos y su integración con APIs hacen de React una opción sólida y escalable que garantiza una experiencia de usuario fluida y eficiente.

4.19. Investigación de tecnologías para el manejo de estilos en el *Frontend*

El manejo de estilos en el desarrollo de interfaces de usuario es esencial para garantizar una presentación atractiva y funcional. Actualmente, existen diversas tecnologías que permiten definir y personalizar el diseño visual de una aplicación web. Entre las más populares se encuentran CSS, TailwindCSS y Bootstrap, cada una con características y ventajas específicas que las hacen adecuadas para distintos tipos de proyectos. A continuación, se presenta un análisis comparativo de estas tecnologías, evaluando sus características, ventajas, desventajas y aplicabilidad en el desarrollo de proyectos web.

4.19.1. Características clave

CSS es el lenguaje estándar para definir la apariencia y el diseño de los documentos web. Su principal fortaleza radica en la separación del contenido HTML de su presentación, lo que permite un control detallado y específico sobre los estilos aplicados a cada elemento en una página web [124]. CSS puro sigue siendo esencial en proyectos que requieren personalización profunda o compatibilidad con navegadores más antiguos.

Por otro lado, TailwindCSS es un *framework* de utilidades primero que permite aplicar estilos directamente en el HTML mediante clases predefinidas. Esto facilita un desarrollo rápido y eficiente, reduciendo la necesidad de escribir CSS personalizado para cada componente [8].

Bootstrap, un *framework* de CSS desarrollado por X (Twitter), ofrece un enfoque diferente al proporcionar un sistema de componentes predefinidos, como botones, formularios y modales, así como un sistema de cuadrículas que facilita la creación de diseños responsivos. Bootstrap es ampliamente utilizado en proyectos que requieren una solución rápida y estructurada para el diseño web [90].

4.19.2. Ventajas comparativas

Cada tecnología presenta ventajas únicas que la hacen atractiva para distintos contextos. CSS puro ofrece la mayor flexibilidad y control sobre el diseño, lo que lo convierte en la opción ideal para proyectos pequeños o donde se necesita una personalización específica y detallada. Además, permite separar completamente el contenido de su presentación, lo que facilita la reutilización de hojas de estilo en múltiples páginas [80].

TailwindCSS destaca por su capacidad para acelerar el desarrollo mediante el uso de clases de utilidad predefinidas, lo que permite aplicar estilos directamente en el HTML sin necesidad de escribir CSS adicional. Esto garantiza una consistencia en el diseño y facilita la optimización del CSS eliminando las clases no utilizadas en producción, lo que reduce el tamaño del archivo CSS y mejora el rendimiento [11].

Bootstrap, en cambio, es conocido por su facilidad de uso y por ofrecer una solución completa para el diseño responsivo mediante su sistema de cuadrículas y componentes predefinidos. Esto permite un desarrollo rápido de interfaces estilizadas sin necesidad de diseñar desde cero, lo que es particularmente útil en proyectos que requieren una implementación inmediata [72].

4.19.3. Desventajas y retos

Aunque CSS puro ofrece la mayor flexibilidad, su uso en proyectos grandes puede volverse complejo debido a la necesidad de gestionar múltiples hojas de estilo y manejar la especificidad y herencia de reglas [80]. Esto puede complicar la implementación y el mantenimiento en proyectos más avanzados.

En el caso de TailwindCSS, uno de los principales retos es la verbosidad en el HTML. El uso de múltiples clases de utilidad puede hacer que el código HTML sea difícil de leer y mantener, especialmente en proyectos grandes [11]. Además, la curva de aprendizaje puede ser pronunciada para quienes no están familiarizados con este enfoque basado en utilidades.

Por su parte, Bootstrap puede generar sitios web uniformes si no se personalizan los estilos, ya que muchos proyectos que lo utilizan tienden a tener una apariencia similar. Además, el peso de sus archivos puede afectar negativamente el rendimiento si no se optimizan adecuadamente, y algunos componentes requieren JavaScript adicional, lo que añade complejidad al proyecto [72].

4.19.4. Popularidad y uso en el mercado laboral

En 2024, el mercado laboral muestra una clara demanda de habilidades relacionadas con el manejo de estilos ([16]), con CSS siendo una habilidad fundamental para cualquier desarrollador *frontend*. El conocimiento de CSS es un requisito básico en la mayoría de las ofertas de empleo, especialmente en proyectos que requieren personalización detallada y control sobre el diseño [97].

TailwindCSS ha ganado popularidad en *startups* y proyectos modernos debido a su enfoque eficiente y su capacidad para crear diseños únicos sin sacrificar flexibilidad. La demanda de desarrolladores con experiencia en TailwindCSS sigue en aumento, especialmente en proyectos donde la optimización del CSS y la consistencia del diseño son factores clave [145].

Bootstrap, aunque ha visto un declive en popularidad frente a TailwindCSS, sigue siendo una opción fuerte en empresas más establecidas que priorizan el desarrollo rápido de interfaces funcionales con un sistema de diseño predefinido. La demanda de Bootstrap se mantiene, sobre todo en proyectos *legacy* y en empresas que valoran la consistencia y facilidad de uso del framework [15].

4.19.5. Comparativa general

- **Facilidad de aprendizaje**: Bootstrap es más fácil de aprender para principiantes debido a su amplia documentación y sus componentes predefinidos. TailwindCSS requiere más tiempo para aprender a usar sus clases de utilidad, pero puede acelerar el desarrollo una vez dominado [72]. CSS puro ofrece una curva de aprendizaje más suave en términos de fundamentos, pero se vuelve más complejo en proyectos grandes.
- **Flexibilidad y personalización**: TailwindCSS es la opción más flexible para quienes buscan personalización rápida y control granular sobre los estilos sin escribir CSS adicional. CSS puro ofrece el máximo control, pero requiere más tiempo y esfuerzo. Bootstrap, aunque es fácil de implementar, ofrece menos flexibilidad sin una personalización significativa [8].
- **Eficiencia en el desarrollo**: Bootstrap permite una implementación rápida de interfaces funcionales, pero TailwindCSS ofrece un desarrollo más eficiente a largo plazo en términos de mantenimiento y tamaño del archivo CSS. CSS puro es más eficiente en proyectos pequeños donde no es necesaria una estructura compleja [80].

4.19.6. Conclusión sobre qué tecnología utilizar para el manejo de estilos

Tras analizar las tecnologías disponibles, TailwindCSS se presenta como la mejor opción para proyectos que requieren alta flexibilidad y un desarrollo rápido sin sacrificar la personalización. Su enfoque de utilidades primero garantiza consistencia y eficiencia en el desarrollo, lo que lo convierte en una herramienta ideal para el manejo de estilos en aplicaciones modernas y escalables.

4.20. Investigación de tecnologías *Backend*

La elección de la tecnología *backend* sigue siendo crucial para el rendimiento, escalabilidad y eficiencia de una aplicación web. Actualmente, los *frameworks* y entornos más populares incluyen Express, Nest, y Fastify, cada uno con características distintivas que lo hacen adecuado para diferentes tipos de proyectos. Este análisis se centra en su rendimiento, flexibilidad, y aplicabilidad.

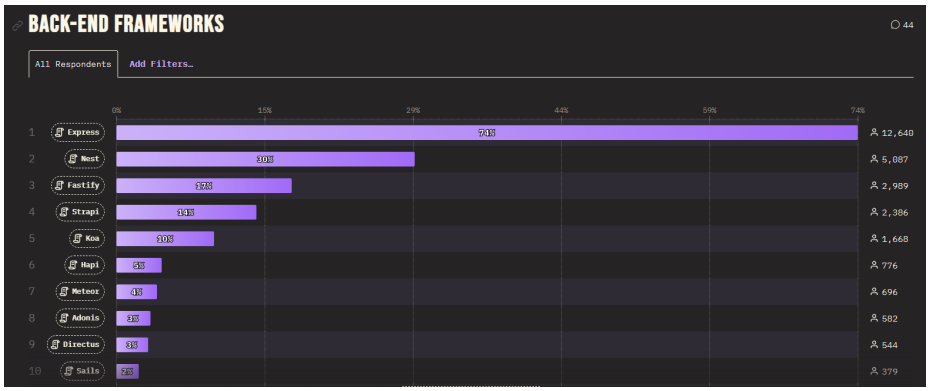

Figura 4.34: Top de tecnologías *backend* a lo largo del tiempo

4.20.1. Características clave

Express es un *framework* minimalista para Node.js que proporciona una estructura ligera y rápida para construir aplicaciones web. Es ampliamente utilizado debido a su simplicidad y velocidad, lo que permite a los desarrolladores crear APIs robustas y escalables rápidamente. Se enfoca en el desarrollo rápido con un enfoque no bloqueante y cuenta con una gran comunidad de soporte [2].

Nest es un *framework* progresivo de Node.js para el desarrollo de aplicaciones *backend*. Basado en TypeScript, utiliza el patrón de diseño modular y la arquitectura en capas, lo que facilita la escalabilidad de grandes proyectos. Nest está diseñado para la creación de microservicios, APIs RESTful y GraphQL, y proporciona un robusto soporte para TypeORM, Mongoose y otros ORM populares [4].

Fastify es un *framework* web Node.js altamente eficiente diseñado para aplicaciones de alto rendimiento. Su arquitectura se enfoca en la velocidad, con un énfasis en la menor cantidad de *overhead* posible, lo que lo hace ideal para aplicaciones de baja latencia. Además, cuenta con una amplia gama de *plugins* que permiten la personalización sin sacrificar la eficiencia [3].

4.20.2. Ventajas comparativas

Cada tecnología ofrece ventajas para diferentes escenarios:

Express se destaca por su simplicidad y capacidad de adaptarse a cualquier tipo de proyecto. Su estructura flexible permite a los desarrolladores personalizar la arquitectura del servidor sin restricciones, siendo ideal para prototipos y aplicaciones de tamaño mediano [147].

Nest proporciona una estructura más organizada y modular en comparación con Express. Su uso de TypeScript mejora la escalabilidad, la mantenibilidad y la seguridad del código, siendo adecuado para aplicaciones empresariales y de gran escala [131].

Fastify ofrece un rendimiento superior en aplicaciones de alta concurrencia. Gracias a su enfoque en la eficiencia, es una opción ideal para sistemas que requieren baja latencia y manejo de grandes cantidades de datos [94].

4.20.3. Desventajas y retos

Express puede volverse complicado de mantener en aplicaciones de gran escala debido a su enfoque minimalista, lo que puede requerir más código personalizado para ciertas funcionalidades avanzadas [147].

Nest, aunque es más estructurado, tiene una curva de aprendizaje más empinada debido a su uso de TypeScript y su arquitectura más compleja. Requiere más tiempo de configuración inicial en comparación con Express [131].

Fastify, si bien es extremadamente rápido, puede carecer de algunas funcionalidades integradas que otros *frameworks* ofrecen de manera predeterminada, lo que puede requerir la instalación de *plugins* adicionales para ciertas características [94].

4.20.4. Popularidad y uso en el mercado laboral

En 2024, las tecnologías de *backend* más utilizadas incluyen:

Express sigue siendo uno de los frameworks más populares debido a su simplicidad y rapidez en el desarrollo de aplicaciones. Es ampliamente adoptado por startups y empresas tecnológicas que buscan una solución rápida y eficaz para sus APIs y microservicios [14].

Nest ha ganado popularidad en empresas que desarrollan aplicaciones a gran escala y buscan una arquitectura robusta y mantenible. Su soporte para TypeScript lo hace especialmente atractivo para desarrolladores que prefieren un código más seguro y tipado [14].

Fastify ha sido adoptado por empresas que buscan optimizar el rendimiento de sus APIs y microservicios, especialmente en sectores que requieren una alta concurrencia y baja latencia [14].

4.20.5. Comparativa general

- **Rendimiento**: Fastify es la opción más eficiente para aplicaciones con alta concurrencia y baja latencia, mientras que Nest y Express ofrecen un rendimiento robusto y confiable para la mayoría de los proyectos.
- **Flexibilidad**: Express es altamente flexible y fácil de adaptar a diferentes arquitecturas, mientras que Nest proporciona una estructura modular que mejora la escalabilidad y mantenibilidad. Fastify destaca en términos de eficiencia sin sacrificar la personalización.
- **Facilidad de desarrollo**: Express y Nest son conocidos por su capacidad para acelerar el desarrollo gracias a su simplicidad y estructura modular, respectivamente, mientras que Fastify requiere más configuración inicial, pero ofrece un rendimiento superior a largo plazo.

4.20.6. Conclusión sobre qué tecnología utilizar para el *Backend*

Tras un análisis exhaustivo de las tecnologías *backend* disponibles, Express se presenta como la opción más adecuada para este proyecto. Su enfoque minimalista y simplicidad lo convierten en una excelente elección para la creación de APIs robustas y escalables, algo fundamental para la fase inicial de desarrollo de la aplicación.

El uso de **Express** permitirá aprovechar el modelo no bloqueante de Node.js, lo cual es esencial para manejar múltiples conexiones simultáneas de manera eficiente, un requisito crítico para este proyecto. Dado que la estructura planificada de la aplicación se enfoca en ofrecer respuestas rápidas y capacidad de escalabilidad sin comprometer el rendimiento, la flexibilidad de Express será clave para adaptar la arquitectura del servidor a las necesidades específicas del sistema [2].

Además, al utilizar un stack completo en JavaScript, Express facilitará la unificación del código entre el *frontend* y el *backend*, lo cual es beneficioso tanto para la simplicidad como para la mantenibilidad a largo plazo. Aunque frameworks como Nest y Fastify ofrecen ventajas en términos de modularidad y rendimiento extremo, respectivamente, la simplicidad, velocidad de implementación y gran soporte comunitario de Express lo hacen una opción óptima para el desarrollo inicial de este proyecto.

4.21. Investigación de tecnologías para bases de datos

La selección de la base de datos adecuada es crucial para asegurar el rendimiento, la escalabilidad y la eficiencia de una aplicación. Existen dos categorías principales de bases de datos: relacionales (RDBMS) y no relacionales (NoSQL). A continuación, se presenta una comparativa entre estas tecnologías, evaluando sus ventajas, desventajas, y cuándo es más adecuado utilizar cada una, con un enfoque particular en la decisión de utilizar PostgreSQL, un sistema de gestión de bases de datos relacionales.

4.21.1. Bases de datos relacionales

Las bases de datos relacionales, como PostgreSQL, son utilizadas comúnmente en aplicaciones que requieren consistencia e integridad de datos. Utilizan el lenguaje SQL (Structured Query Language) para gestionar y consultar los datos, lo que las hace ideales para aplicaciones que necesitan transacciones fiables y consultas complejas.

Ventajas

1. **Integridad y consistencia de los datos**: Las bases de datos relacionales aseguran que los datos se mantengan coherentes y válidos mediante el uso de transacciones ACID (Atomicidad, Consistencia, Aislamiento y Durabilidad), lo cual es fundamental para aplicaciones críticas como las financieras y de gestión empresarial [1].

2. **Consultas complejas con SQL**: El lenguaje SQL es potente y estandarizado, lo que permite realizar consultas avanzadas para obtener y analizar datos de múltiples tablas y relaciones, haciendo que las bases de datos relacionales sean ideales para entornos de negocios que manejan datos estructurados [1].

3. **Escalabilidad vertical**: Aunque las bases de datos relacionales tienen limitaciones en cuanto a la escalabilidad horizontal, pueden manejar grandes cantidades de datos mediante la adición de recursos como CPU y memoria a un servidor, mejorando así el rendimiento [1].

4. **Amplio soporte y comunidad**: Bases de datos como PostgreSQL gozan de un sólido respaldo de la comunidad y cuentan con una extensa documentación y herramientas de terceros, lo que facilita su uso y adaptación en una amplia gama de aplicaciones [1].

Desventajas

1. **Escalabilidad horizontal limitada**: Aunque algunas soluciones permiten escalar horizontalmente (añadiendo más servidores), este proceso puede ser más complicado y costoso en bases de datos relacionales en comparación con las bases de datos NoSQL [1].
2. **Rigidez del esquema**: Las bases de datos relacionales requieren un esquema predefinido, lo que puede hacer más difícil adaptar o cambiar la estructura de datos durante el desarrollo [1].
3. **Desempeño en grandes volúmenes de datos**: Aunque son eficientes para manejar datos estructurados, las bases de datos relacionales pueden experimentar una disminución del rendimiento cuando se manejan volúmenes de datos extremadamente grandes sin una correcta optimización [1].

4.21.2. Bases de datos No Relacionales (NoSQL)

Las bases de datos no relacionales están diseñadas para manejar grandes volúmenes de datos no estructurados o semiestructurados. Su flexibilidad en cuanto al esquema y su capacidad de escalar horizontalmente las hace populares en aplicaciones modernas que manejan grandes cantidades de datos, como redes sociales o sistemas de análisis en tiempo real.

Ventajas

1. **Escalabilidad horizontal**: NoSQL permite escalar fácilmente añadiendo más servidores, lo que facilita el manejo de aplicaciones con gran volumen de datos y alta concurrencia [134].
2. **Flexibilidad del esquema**: NoSQL permite trabajar sin un esquema rígido, lo que resulta útil cuando se manejan datos no estructurados o cuando los requisitos de datos pueden cambiar frecuentemente [134].
3. **Rendimiento en tiempo real**: Las bases de datos NoSQL están diseñadas para operaciones rápidas de lectura y escritura, lo que es ideal para aplicaciones que requieren alta concurrencia o tiempos de respuesta rápidos, como sistemas de recomendaciones o análisis en tiempo real [134].

Desventajas

1. **Consistencia eventual**: Muchas bases de datos NoSQL ofrecen consistencia eventual, lo que significa que no garantizan que todas las operaciones se reflejen instantáneamente en todas las réplicas, lo que puede no ser adecuado para aplicaciones críticas que requieren transacciones ACID [134].
2. **Consultas complejas**: Aunque son rápidas para operaciones simples, las consultas complejas pueden ser menos eficientes y más difíciles de implementar en NoSQL en comparación con SQL en bases de datos relacionales [134].
3. **Falta de estandarización**: No existe un lenguaje de consulta estandarizado en NoSQL, lo que puede complicar el desarrollo y mantenimiento a largo plazo, especialmente en proyectos grandes [134].

4.21.3. Comparación y casos de uso

Las bases de datos relacionales y no relacionales son adecuadas para distintos tipos de aplicaciones, dependiendo de la estructura de los datos, la escalabilidad requerida y las necesidades de transacciones.

Bases de datos Relacionales (PostgreSQL)

Las bases de datos relacionales son ideales para:

- **Aplicaciones transaccionales**: En aplicaciones que requieren mantener la consistencia e integridad de los datos, como sistemas bancarios, ERP o plataformas de e-commerce, PostgreSQL garantiza transacciones ACID [1].
- **Consultas complejas**: Cuando es necesario realizar operaciones avanzadas de consulta y análisis de datos, SQL proporciona una solución eficiente y estandarizada [1].
- **Estructura de datos bien definida**: Si los datos son altamente estructurados y las relaciones entre ellos son importantes, como en bases de datos de productos, clientes o inventarios, PostgreSQL es una excelente opción [1].

Bases de datos No Relacionales (NoSQL)

Por otro lado, las bases de datos NoSQL son preferibles para:

- **Aplicaciones con grandes volúmenes de datos**: En entornos como redes sociales, donde se manejan datos no estructurados en grandes volúmenes y alta concurrencia, NoSQL sobresale [134].
- **Escalabilidad horizontal**: NoSQL es ideal para aplicaciones que necesitan distribuir la carga de datos entre varios servidores para manejar millones de operaciones por segundo [134].
- **Flexibilidad y agilidad**: Si se espera que los requisitos de datos cambien rápidamente o si se están manejando datos no estructurados como JSON o XML, NoSQL ofrece una mayor flexibilidad para adaptarse a esos cambios [134].

4.21.4. Conclusión sobre qué tipo de base de datos utilizar

Tras analizar las distintas tecnologías de bases de datos, se ha decidido utilizar PostgreSQL para este proyecto debido a sus características que mejor se ajustan a los requisitos. PostgreSQL proporciona una estructura robusta que garantiza la consistencia y fiabilidad de los datos, y es capaz de manejar consultas complejas de manera eficiente, lo cual es crucial para el análisis de datos y las transacciones en tiempo real que se requieren.

Además, PostgreSQL ofrece la flexibilidad necesaria para manejar grandes volúmenes de datos estructurados, con la opción de escalar verticalmente para satisfacer las demandas de rendimiento del sistema. Su capacidad de integrarse de manera eficiente con otras tecnologías *backend*, como NodeJS, facilita la creación de una API robusta para el análisis estadístico y la gestión de datos.

4.22. Reglamento general de protección de datos

4.22.1. ¿Qué es el RGPD?

El Reglamento General de Protección de Datos (RGPD) es una normativa de la Unión Europea que establece un marco legal para la protección de los datos personales de los ciudadanos europeos. Entró en vigor el 25 de mayo de 2018 y tiene como objetivo principal otorgar a los individuos un mayor control sobre su información personal, así como armonizar las regulaciones de protección de datos en toda la UE.

El RGPD define los principios y obligaciones que deben seguir las organizaciones al recopilar, procesar y almacenar datos personales. Entre sus disposiciones clave se incluyen:

- **Consentimiento explícito**: Las organizaciones deben obtener un consentimiento claro y afirmativo de los individuos para procesar sus datos.
- **Derecho al olvido**: Los individuos pueden solicitar la eliminación de sus datos personales cuando ya no sean necesarios para los fines para los que fueron recopilados.
- **Portabilidad de datos**: Los individuos tienen el derecho de recibir sus datos personales en un formato estructurado y comúnmente utilizado, y de transferirlos a otro controlador.
- **Notificación de violaciones de seguridad**: Las organizaciones deben informar a las autoridades de protección de datos y a los individuos afectados sobre cualquier violación de seguridad que comprometa datos personales.

4.22.2. ¿Por qué es importante implementar el RGPD?

La implementación del RGPD es crucial por varias razones:

- **Protección de la privacidad**: En una era digital donde la recopilación y el uso de datos personales son omnipresentes, el RGPD garantiza que los derechos de privacidad de los individuos sean respetados y protegidos.
- **Confianza del consumidor**: Las organizaciones que cumplen con el RGPD demuestran un compromiso con la protección de datos, lo que puede fortalecer la confianza y lealtad de sus clientes.
- **Armonización legal**: El RGPD unifica las regulaciones de protección de datos en toda la UE, facilitando a las empresas operar en múltiples países europeos bajo un conjunto común de reglas.
- **Sanciones por incumplimiento**: El incumplimiento del RGPD puede resultar en multas significativas, que pueden alcanzar hasta 20 millones de euros o el 4 % de la facturación anual global de la empresa, lo que resalta la importancia de su implementación.
- **Adaptación a nuevas tecnologías**: El RGPD aborda desafíos contemporáneos relacionados con tecnologías emergentes y el manejo masivo de datos, asegurando que las prácticas de protección de datos evolucionen junto con el panorama tecnológico.

4.23. Metodologías de desarrollo colaborativo

El desarrollo de software ha evolucionado hacia un enfoque más colaborativo y ágil, donde las metodologías y las plataformas juegan un papel crucial para garantizar la eficiencia, calidad y transparencia del proceso. En esta sección se exploran las metodologías ágiles, el uso de plataformas colaborativas y las herramientas para la gestión de versiones y seguimiento de errores.

4.23.1. Metodologías ágiles: Scrum

Scrum es una de las metodologías ágiles más adoptadas en la industria del desarrollo de software, ya que promueve un enfoque iterativo e incremental. Esta metodología divide el trabajo en ciclos cortos llamados sprints, generalmente de dos a cuatro semanas de duración. Durante cada sprint, el equipo se enfoca en completar un conjunto específico de tareas, lo que permite una mayor flexibilidad y la capacidad de adaptarse a los cambios en los requisitos del proyecto [126].

Scrum se estructura en roles, eventos y artefactos. Los principales roles incluyen el Scrum Master, responsable de facilitar el proceso; el Product Owner, quien representa los intereses del cliente; y el Development Team, que es el equipo encargado de implementar las funcionalidades. Los eventos clave incluyen la Daily Standup, una reunión diaria para revisar el progreso, y la Sprint Review, donde se presenta el trabajo completado [126, 119].

El uso de Scrum mejora la comunicación entre los miembros del equipo y con los interesados del proyecto, al mismo tiempo que permite una entrega continua de valor a través de entregas incrementales. Esta metodología es especialmente efectiva en proyectos con alta incertidumbre o requisitos cambiantes, ya que facilita la adaptación continua [119].

4.23.2. Uso de plataformas colaborativas: GitHub, Trello

El desarrollo de software colaborativo a menudo implica la integración de plataformas que permiten la gestión eficiente del código, la comunicación entre los miembros del equipo y la organización de tareas. Dos de las plataformas más utilizadas en la industria son GitHub y Trello.

GitHub es una plataforma de control de versiones que permite a los desarrolladores colaborar en proyectos de manera eficiente utilizando Git. GitHub facilita la gestión de versiones a través de características como pull requests, que permiten a los miembros del equipo revisar cambios en el código antes de que sean fusionados a la rama principal. Además, GitHub permite la integración con herramientas de CI/CD (Integración Continua y Entrega Continua), lo que garantiza que el software se pruebe y despliegue automáticamente [41, 6].

Por otro lado, Trello es una herramienta de gestión de tareas basada en tableros, donde cada tarea se representa como una tarjeta que puede ser movida entre diferentes listas que representan el estado del trabajo. Esta plataforma permite a los equipos visualizar el flujo de trabajo de manera clara, delegar tareas y establecer fechas de vencimiento, lo que contribuye a una mejor organización y coordinación del equipo [10].

El uso combinado de GitHub y Trello en proyectos colaborativos permite una gestión más eficiente tanto del código como de las tareas, asegurando que todos los miembros del equipo estén alineados con los objetivos del proyecto [10, 41].

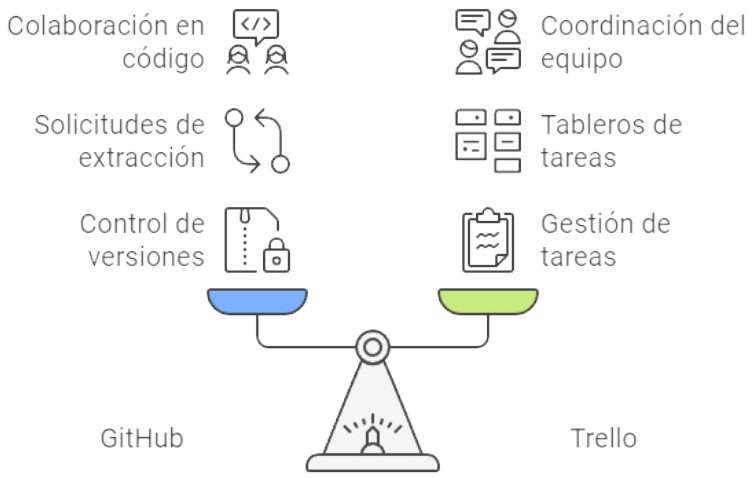

Figura 4.35: Comparación de Github y Trello para la colaboración en equipo.

4.23.3. Gestión de versiones y seguimiento de errores

La gestión de versiones es un aspecto fundamental en el desarrollo de software, ya que permite controlar los cambios realizados en el código a lo largo del tiempo. El uso de sistemas de control de versiones distribuidos, como Git, es esencial para mantener un registro de las modificaciones, revertir cambios no deseados y colaborar con otros desarrolladores sin conflictos. Git permite a los equipos trabajar en múltiples ramas simultáneamente, lo que facilita la implementación de nuevas funcionalidades sin afectar la estabilidad del código base [41].

El seguimiento de errores es otro componente crucial en el desarrollo colaborativo. Herramientas como GitHub Issues permiten registrar, rastrear y priorizar errores, asegurando que los problemas se resuelvan de manera oportuna. Cada error puede ser asignado a un miembro del equipo y vinculado a una rama específica del proyecto, lo que facilita la resolución rápida y eficiente. Integrar el seguimiento de errores con la gestión de versiones garantiza que el equipo esté siempre al tanto del estado del proyecto y pueda reaccionar de manera proactiva a los problemas [6].

4.24. Contexto local y relevancia en Guatemala

El desarrollo de videojuegos en Guatemala es una industria emergente que, aunque aún se encuentra en una etapa temprana, ha comenzado a mostrar un crecimiento sostenido en los últimos años. Esta sección analiza el estado actual de la industria de videojuegos en el país, las oportunidades y desafíos locales, y el impacto potencial de proyectos innovadores en este sector.

4.24.1. Estado de la industria de videojuegos en Guatemala

La industria de los videojuegos en Guatemala ha tenido un crecimiento lento en comparación con otros países de la región. Sin embargo, en la última década, ha surgido una comunidad de desarrolladores independientes que han comenzado a posicionarse tanto a nivel local como internacional. Aunque el mercado de consumo de videojuegos es considerable, especialmente en dispositivos móviles, el desarrollo local ha enfrentado retos como la falta de financiamiento, recursos tecnológicos limitados y escaso apoyo institucional [9].

A pesar de estos desafíos, iniciativas como la creación de pequeños estudios independientes y eventos como Guatemala Game Devs Meetup han fomentado la colaboración y el intercambio de conocimientos entre desarrolladores. Estas comunidades están trabajando en conjunto para fortalecer la presencia del país en la industria y atraer inversiones externas [13].

4.24.2. Oportunidades y desafíos locales

El entorno guatemalteco presenta tanto oportunidades como desafíos para los desarrolladores de videojuegos. Una de las principales oportunidades es el creciente acceso a herramientas de desarrollo accesibles, como Unity y Unreal Engine, que permiten a los creadores locales competir en igualdad de condiciones con estudios internacionales. Además, el bajo costo de operación en comparación con otros países de la región hace que Guatemala sea un lugar atractivo para el desarrollo de videojuegos independientes [13, 7].

Sin embargo, los desafíos son considerables. La falta de inversión en tecnología y educación especializada en el desarrollo de videojuegos sigue siendo un obstáculo importante. Además, la percepción social del desarrollo de videojuegos como una industria viable aún es limitada, lo que restringe el crecimiento de este sector. La necesidad de alianzas con instituciones académicas y el apoyo gubernamental es crucial para consolidar la industria local [153].

4.24.3. Impacto potencial del proyecto en la industria local

El proyecto en desarrollo tiene el potencial de contribuir significativamente a la industria de videojuegos en Guatemala, no solo por su innovación tecnológica, sino también por su capacidad para abrir nuevas oportunidades de colaboración y aprendizaje. La implementación de tecnologías avanzadas como la inteligencia artificial y los sistemas colaborativos en el desarrollo de videojuegos puede servir como un ejemplo para otros estudios locales y generar interés entre inversionistas nacionales e internacionales [13].

Además, la creación de videojuegos que resuenen con la identidad cultural guatemalteca podría ayudar a diferenciar los productos locales en el mercado global. Proyectos como este tienen el potencial de mostrar que Guatemala puede ser un actor relevante en la industria de videojuegos de América Latina, atrayendo talento y capital hacia este sector [153, 7].

CAPÍTULO 5

Antecedentes

5.1. Estudio del mercado web en el gaming

El mercado de los videojuegos ha integrado el uso de sitios web para mejorar la experiencia del jugador, ofrecer soporte, y centralizar la información sobre los juegos. Esta investigación busca analizar cómo los principales juegos del mercado utilizan sus sitios web para interactuar con la comunidad, gestionar estadísticas y ofrecer una experiencia web complementaria. Para este análisis, se han considerado los cinco juegos más jugados de **Steam** y el mundialmente conocido **Grand Theft Auto V (GTA V)**.

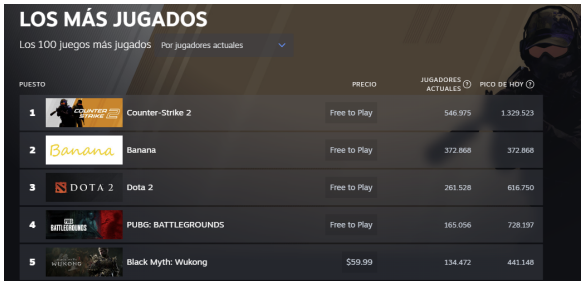

Figura 5.1: Top cinco de los juegos más jugados en Steam

5.1.1. Análisis de páginas web de los mejores juegos en Steam

Counter Strike 2

El sitio web de **Counter Strike 2**, desarrollado por Valve Corporation, se enfoca en ser un centro de noticias, preguntas frecuentes y un portal que redirige a Steam para descargar el juego de manera gratuita. Además, proporciona estadísticas sobre la cantidad de jugadores mensuales, lo que permite a los usuarios tener una idea del tamaño de la comunidad activa del juego. Este enfoque promueve tanto la información como la accesibilidad del juego.

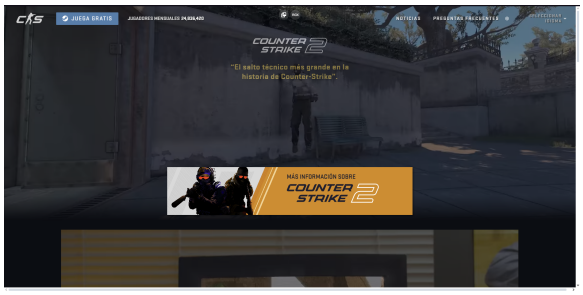

Figura 5.2: Sitio oficial de Counter-Strike 2

Banana

El juego **Banana**, desarrollado por Sky, fue observado durante el proceso de investigación y se determinó que está creado con el motor Unity por desarrolladores independientes. El juego se basa en una mecánica sencilla que consiste en hacer clic sobre una banana. Actualmente, no cuenta con un sitio web oficial, y su distribución se realiza exclusivamente a través de la plataforma Steam.

Figura 5.3: Publicación en Steam de Banana

Dota 2

El sitio web oficial de **Dota 2**, desarrollado por Valve Corporation, ofrece información detallada sobre el juego. Incluye secciones dedicadas a las actualizaciones recientes y anteriores, así como los parches implementados. Además, cuenta con una sección de noticias donde los jugadores pueden mantenerse al tanto de las novedades y eventos más recientes. En cuanto a la jugabilidad, el sitio proporciona información sobre los héroes (personajes del juego). Finalmente, además de servir como enlace directo para descargar el juego a través de Steam, la página también destaca el entorno de los *esports* relacionados con Dota 2.

PUBG: BATTLEGROUNDS

El sitio web oficial de **PUBG: BATTLEGROUNDS**, desarrollado por PUBG Studios, se centra en proporcionar información, novedades, eventos y contenido relacionado con el juego. También tiene un enfoque promocional, ofreciendo en la sección de ayuda acceso a las distintas plataformas donde el juego está disponible. Además, cuenta con una sección de comunidad que permite acceder directamente a los tweets y publicaciones en Facebook realizados a través de los canales oficiales del estudio.

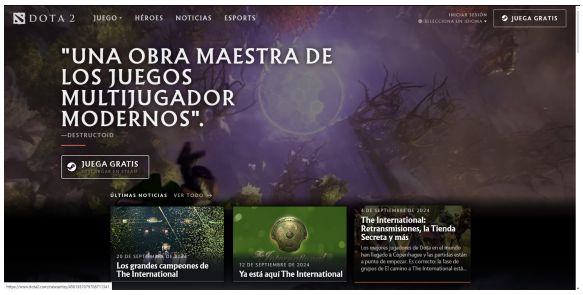

Figura 5.4: Sitio oficial de Dota 2

Figura 5.5: Sitio oficial de PUBG

Black Myth: Wukong

El sitio web oficial de **Black Myth: Wukong**, tiene un enfoque ligeramente distinto pero alineado con su objetivo principal, ya que se utiliza principalmente para la promoción del juego. A través de este sitio, se ofrecen vistas previas de niveles mediante contenido multimedia, además de wallpapers y arte conceptual. Una característica distintiva es que el sitio también muestra las calificaciones y reseñas que ha recibido el juego en diversas plataformas de críticas, lo que refuerza su posición como uno de los títulos mejor valorados en la actualidad. Finalmente, cuenta con una sección de ayuda donde los jugadores pueden contactar con el soporte técnico en caso de algún problema.

Figura 5.6: Sitio oficial de Black Myth: Wukong

5.1.2. Análisis de la página web de Grand Theft Auto V (GTA V)

Para profundizar en el análisis, se incluyó **Grand Theft Auto V (GTA V)**, desarrollado por Rockstar Games, uno de los juegos más vendidos de todos los tiempos. Rockstar utiliza la web para ofrecer una plataforma llamada **Social Club**, que permite a los jugadores ver estadísticas detalladas

tanto de su progreso en el modo offline como online. Los jugadores pueden acceder a información sobre sus misiones completadas, logros y rendimiento en general. Además, el sitio web de GTA V ofrece información general sobre el juego, asistencia técnica, y comunidad, fomentando la interacción constante entre los jugadores y el juego.

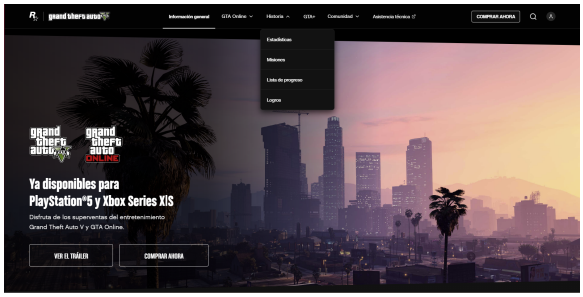

Figura 5.7: Página de inicio de Grand Theft Auto V

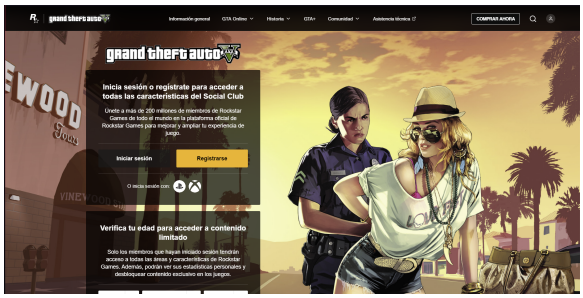

Figura 5.8: Página de inicio de sesión a Social Club de Rockstar

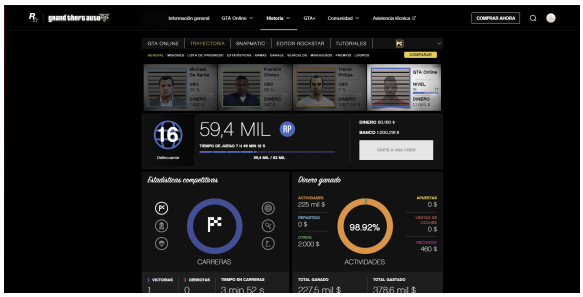

Figura 5.9: Apartado estadístico para el modo online de GTA V

5.1.3. Conclusiones del análisis del mercado web en el gaming

El análisis de las páginas web de los principales juegos del mercado demuestra que estas plataformas son fundamentales para la experiencia del jugador. Las compañías desarrolladoras utilizan sus sitios web no solo para promocionar y ofrecer descargas, sino también para gestionar comunidades, proporcionar soporte técnico, y ofrecer a los jugadores acceso a estadísticas sobre su rendimiento. Este enfoque aumenta la participación de los jugadores al permitirles un mayor control sobre su experiencia de juego y mantenerlos informados de las últimas actualizaciones y eventos.

En el caso de **Grand Theft Auto V**, la integración con **Social Club** agrega una capa adicional de interacción, permitiendo a los jugadores analizar su progreso en el juego con gran detalle. Esta

capacidad de gestionar y visualizar estadísticas de manera dinámica es un ejemplo destacado de cómo los desarrolladores pueden usar la web para fortalecer la conexión entre el jugador y el juego.

Este análisis proporciona una base sólida para la creación del módulo web de **Platyfa**, el cual buscará integrar muchas de estas funciones clave, como el acceso a estadísticas y la promoción del juego, para mejorar la experiencia del usuario y fomentar una mayor participación de la comunidad.

CAPÍTULO 6

Alcance

La iniciativa Platyfa: El caso de estudio de la creación de un videojuego en un ambiente universitario colaborativo comprende todos los elementos vinculados con la elaboración de un videojuego, desde su primera idea hasta su ejecución final. Esta labor se concentró en fusionar diversas áreas del desarrollo de videojuegos, que incluyen el diseño de las mecánicas de juego, la programación, el arte conceptual, el modelado en 3D, la texturización, la animación, el diseño de niveles y las evaluaciones de usuario.

El objetivo del proyecto fue explorar y poner en práctica técnicas útiles que faciliten a los programadores el aprendizaje y uso de herramientas como Blender, Unity y otros recursos pertinentes para la creación de activos y la programación. Se destacó la importancia de perfeccionar las capacidades particulares requeridas para elaborar un videojuego de alta calidad, fusionando de forma consistente elementos visuales, narrativos y mecánicos.

Es crucial destacar que la extensión de este estudio se restringe al desarrollo del videojuego Platyfa en un entorno universitario. No contempla elementos asociados a la distribución comercial ni a tecnologías de vanguardia que superan las habilidades propuestas en este proyecto, como la realidad aumentada o la inteligencia artificial de vanguardia.

CAPÍTULO 7

Metodología

7.1. Reuniones iniciales y de seguimiento

Estas, como el título lo indica, se realizaron durante los primeros días de planeación con el objetivo de definir a grandes rasgos el videojuego que se desarrollaría grupalmente, conociendo las distintas perspectivas y opiniones de todos los integrantes del proyecto. También se utilizaron para definir las metas y objetivos del proyecto, como también la visión del mismo. Las cuales fueron definidas como las siguientes.

7.1.1. Metas

- Que el proyecto sea de mucho aprendizaje para todos los integrantes del grupo.
- Publicar una versión del juego en plataformas digitales como Steam o similares.
- Que el proyecto sea de mucho valor para el portafolio de todos los integrantes del grupo.
- Fortalecer habilidades, talentos y capacidades de los integrantes del grupo.
- Tener el escenario y contexto más realista posible sobre el desarrollo de un videojuego.
- Formar un modelo de marketing para poder dar a conocer sobre el juego y su producción.

7.1.2. Visión

- Crear un espacio de *networking* durante la producción del juego para participar en más proyectos a futuro.
- Crear un juego de calidad, y divertido que deje huella y aporte en la vida de los jugadores.
- Que el alcance del videojuego llegue al público que se desea satisfacer.

Se definió un cronograma general para el avance esperado durante el desarrollo del videojuego, enfocado en entregas, prototipos y pruebas. 13.18

7.2. Narrativa

Para la definición de la narrativa del videojuego se utilizó el espacio dentro de las reuniones descritas en la sección anterior. En ellas se definieron las bases narrativas del proyecto tales como el personaje principal, el mundo en el que se sitúa la historia, y las escenas que ocurrirán en la historia.

7.2.1. Ideación

La ideación y escritura de la narrativa se llevó a cabo en varias fases. La primera fase fue el establecimiento de la historia literal que el videojuego contaría. En un principio, de qué trataría el videojuego fue ideado, discutido y debatido durante las reuniones iniciales de planeación entre todo el equipo de trabajo. Durante estas reuniones se utilizó la lluvia de ideas para realizar una ideación inicial, y luego se discutió sobre esas ideas para considerar su viabilidad y las preferencias de los integrantes del grupo sobre ellas. Entre todas las ideas propuestas, la idea seleccionada fue la de un videojuego de tipo *shooter* con un estilo caricaturesco. Este juego tendría a un ornitorrinco como personaje principal. Las municiones de las armas del juego serían hechas de comida. A partir de esta idea planteada se generó una imagen haciendo uso de la inteligencia artificial para tener un concepto más claro de cómo podría verse el juego. Esta imagen puede verse en la Figura7.1.

Figura 7.1: Idea original generada con inteligencia artificial

A partir de ello, se continuó desarrollando y modificando la idea, considerando siempre el diseño del modo de juego y su viabilidad.

7.2.2. Versiones preliminares

A partir de la idea inicial surgida durante la lluvia de ideas, se prosiguió a escribir la narrativa de la historia, comenzando desde los términos más generales sobre la estructura, hasta luego definir los elementos más específicos como lo que ocurre en cada escena. Como apoyo se utilizaron diagramas y bocetos como los de la Figura 7.2 para visualizar y organizar ideas. A continuación se encuentra una versión previa de la historia. En esta versión se introduce la premisa inicial del juego, la cual

es una revolución de ornitorrincos contra sus depredadores, y también se introduce a los enemigos principales, basados en especies de animales del continente australiano que se alimentan de animales como los ornitorrincos.

En un mundo gobernado por animales antropomórficos, en las regiones más incógnitas de Australia, habitan un grupo de ornitorrincos que vivían pacíficamente en su propia utopía. Todo iba bien hasta que un día, un frente frío azotó toda la zona provocando que varias especies se mudaran cerca de los ornitorrincos. Aves rapaces, zorros y los más problemáticos los dingos, arrasaron con todos los recursos del ecosistema. Los dingos se aprovecharon de su fuerza para esclavizar y obligar a los ornitorrincos a prepararles platillos que lograran satisfacerlos, de lo contrario acabarían con cada uno de ellos.

La historia se centra en un joven Ornitorrinco llamado Gaus, quien es un aprendiz de chef que sirve a uno de los líderes de los dingos. Desde muy pequeño, Gaus siempre quiso ser un inventor, por lo que en la actualidad siempre busca combinar esta afición con su profesión. Llegó el día de la semana en que los ornitorrincos presentaban sus platillos y los entregaban. Gaus cometió el error de intentar entregar su platillo utilizando una especie de cañón, lo que termina enfureciendo al líder de los dingos y sentenciándolo a ser comida de dingos. Amigos chefs de Gaus saltaron a ayudarle lo que dio inicio a la "Revolución Ornitorisense".

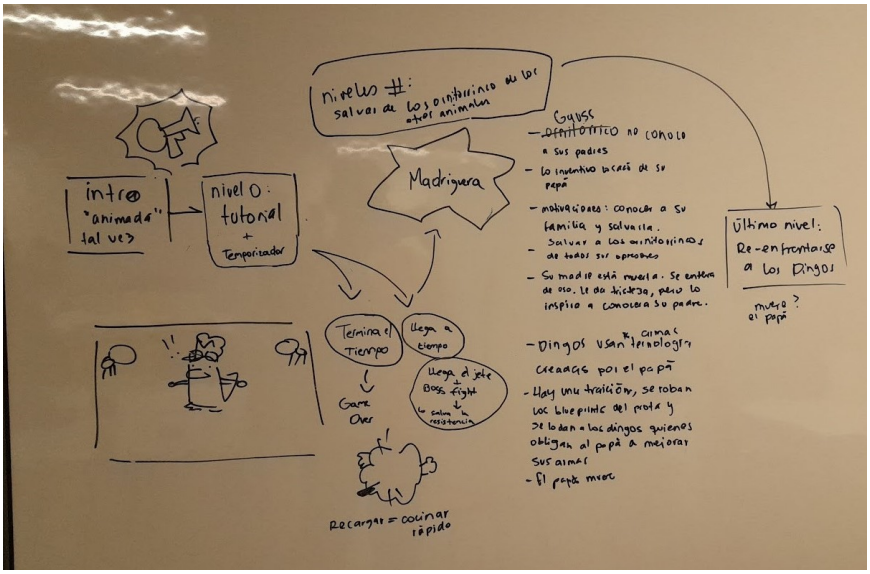

Figura 7.2: Pizarra de planeación de la historia 1

7.2.3. Versión final

En discusión con los integrantes del grupo, se redefinieron varios aspectos de la primera versión historia y de los personajes, agregando mucha más profundidad a ellos. Esto con el objetivo de crear una historia mucho más inmersiva y entretenida que la de la primera versión. Nuevamente, el uso de diagramas tales como el que se puede observar en la Figura7.3 fueron necesarios para explicar y comprender la historia escrita.

Como base de la relación entre los distintos grupos, personajes y enemigos, se decidió utilizar una temática inspirada en las cartas de *poker* para ellas. Cada jefe de cada grupo existente en el juego (dingos, búhos, cocodrilos y ornitorrincos) representaría una carta del mazo; el primer jefe, el capitán de los dingos, representaría a la carta Jota (J). El segundo jefe, la reina de los búhos, representaría a la Reina (Q). El tercer jefe, el líder de los cocodrilos, representaría al Rey, mientras

que la última carta a utilizar sería el comodin (*Joker*), que sería el líder de los ornitorrincos.

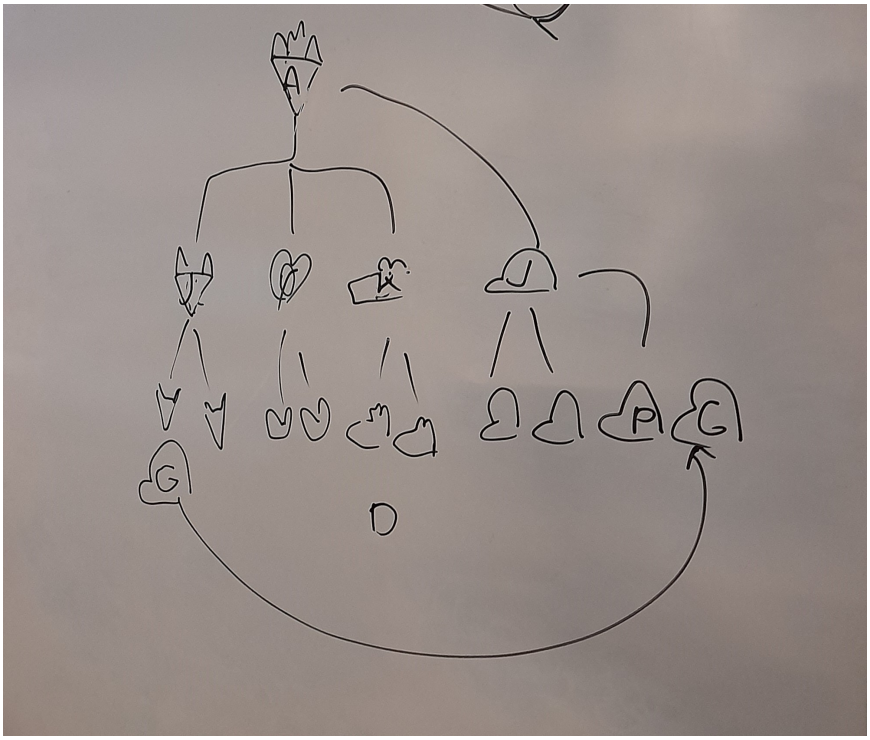

Figura 7.3: Ideación de relación entre personajes y jerarquías dentro de la historia

Gracias a la idea de las cartas, se escribió una nueva versión de la historia, la cual se ve resumida en la siguiente premisa:

En un mundo gobernado por animales antropomórficos, en las regiones más recónditas del continente Australiano, habitan un grupo de ornitorrincos que buscaban vivir pacíficamente. A pesar de ser constantemente acechados por sus depredadores como dingos, búhos y cocodrilos, los ornitorrincos simplemente buscaban vivir tranquilos en sus madrigueras. Escarbando en las profundidades, un ornitorrinco descubrió un misterioso gas natural, el cual prendía fácilmente al entrar en contacto con el fuego. Muchos ornitorrincos le temían a esta temible bestia sin forma. Pero gracias a la astucia del ornitorrinco que lo descubrió, los ornitorrincos domaron a la bestia para utilizarlo como energía, ¡especialmente para cocinar! A partir de ese momento los ornitorrincos se dedicaron a la cocina. Probando hacer cada vez más platillos más complejos, y siempre muy deliciosos.

A los dingos, búhos y cocodrilos a la distancia se les hacía agua la boca, ya no por los ornitorrincos sino por lo exquisito que olían sus creaciones. Angrif, el rey dingo, al ver una oportunidad decidió tomar acción al respecto. Juntó a la manada de dingos más fuertes que tenía, se dirigió al hogar de los ornitorrincos ofreciéndoles protección a cambio de sus servicios de cocina. Los ornitorrincos, con mucha desconfianza, rápidamente rechazaron la oferta. Angrif, muy molesto, no podía tomar un no como respuesta, así que comandó a su manada a atrapar a todos los ornitorrincos, y así hicieron. Los ornitorrincos no pudieron detener la invasión de los dingos y sucumbieron ante su ataque.

Los búhos y cocodrilos se molestaron con los dingos; no era justo que ellos se quedaran con todos los ornitorrincos. Pero esta vez de declarar la guerra a las otras especies y demostrar su superioridad, Angrif lo vio como una oportunidad de unir fuerzas y ofreció dividir los ornitorrincos de manera equitativa entre los grupos. Los cocodrilos y búhos no pudieron rechazar la oferta, y una alianza fue creada a costa de los ornitorrincos, que a partir de ese momento se vieron obligados a trabajar y

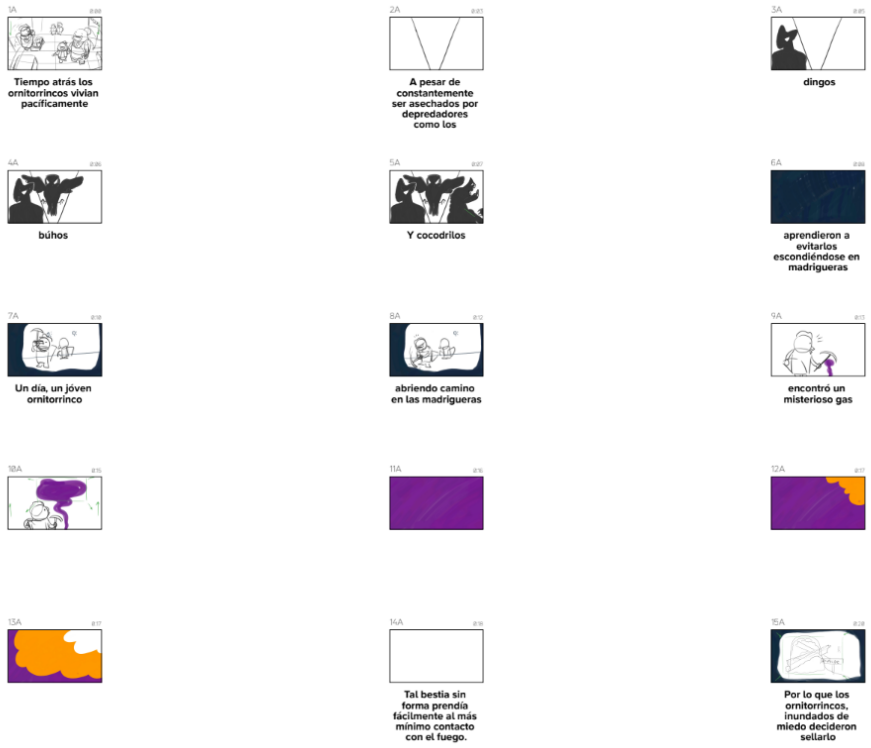

Figura 7.4: Página uno del *storyboard* de la escena inicial

cocinar para sus depredadores.

Años después de este catastrófico evento para los ornitorrincos, existía un ornitorrinco que solo conocía la vida como cocinero de los dingos. A pesar de sus circunstancias, Gaus amaba su trabajo de cocinero, e incluso buscaba mejorar sus platillos fuera de la cocina creando inventos innovadores para la cocina. Sus compañeros no lo comprendían pero a veces se veían beneficiados de sus inventos, así que lo dejaban ser. Fue hasta que un día se acercaba la fecha de un banquete y de celebración para los dingos, y el nuevo invento de Gaus estaba listo; un cañón que sirviera la comida por los ornitorrincos. Pero en el momento de ser usado, en vez de servir la comida, la disparó rápidamente contra Jagger, el capitán dingo, accidentalmente entrando a una batalla y declarando el inicio de una revolución.

7.2.4. Implementación

Para la implementación de la historia de forma explícita, primero, se decidió introducir al jugador con la ayuda de una escena en la que presentara los eventos que ocurrieron antes de la historia de nuestro protagonista, Gaus. Para ello, se realizó un *storyboard* que contuviese todos los acontecimientos relevantes que se le presentarían al jugador, tal como en la Figura 7.4, enfatizando en el descubrimiento del gas de los ornitorrincos, cómo fueron atrapados, y quién es nuestro protagonista. A partir del *storyboard* se crearon todos las ilustraciones necesarios 7.5 para utilizarlos en la edición un video que se reproduciría al inicio de una nueva partida.

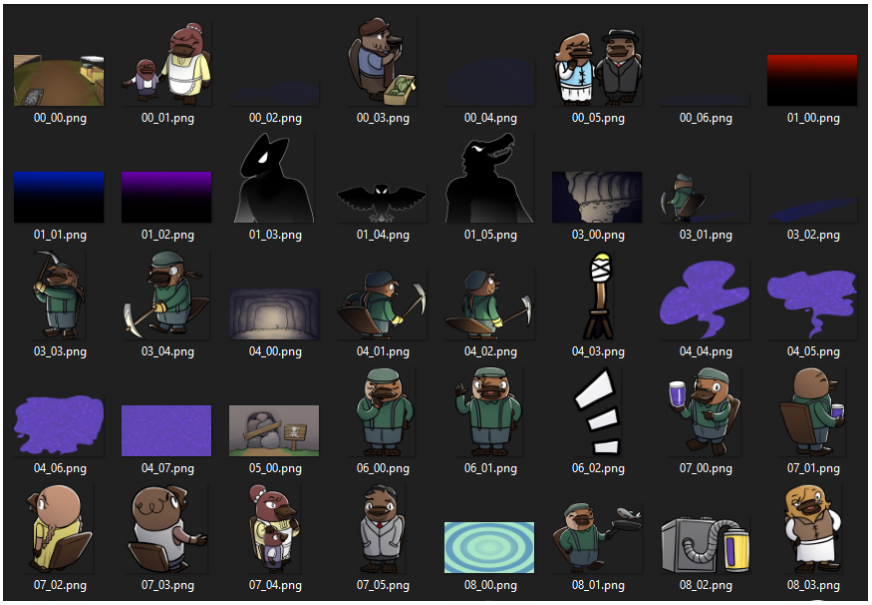

Figura 7.5: Assets ilustrados para el video de introducción de la historia

7.3. Diseño conceptual de enemigos y análisis de videojuegos

El primer paso para comenzar el desarrollo del módulo de enemigos en el videojuego *Platyfa* fue con una investigación y análisis de videojuegos referentes en el diseño de enemigos. Esta fase permitió entender cómo los enemigos impactan la jugabilidad y la satisfacción del jugador, así como identificar las mejores prácticas de diseño para adaptarlas al contexto del juego y poder darle una mejor experiencia al público objetivo. A continuación, se describen los componentes de esta etapa.

7.3.1. Análisis de videojuegos

Para construir una base teórica sólida, se realizó un análisis comparativo de videojuegos clásicos y modernos, explorando la evolución en el diseño de enemigos y jefes, también conocidos como *bosses*. Durante dicho estudio, el objetivo principal fue compender cómo el diseño de los enemigos influye en la jugabilidad, la narrativa, y la satisfacción del jugador, además de identificar prácticas destacadas que podrían aplicarse a *Platyfa*. Entre los videojuegos estudiados se incluyen:

- **Pac-Man** (1980): Este juego revolucionó el diseño de enemigos mediante comportamientos únicos que incrementan progresivamente la dificultad, creando una experiencia desafiante [50].
- **The Last of Us** (2013): Los enemigos en este juego contribuyen a la narrativa y generan tensión, requiriendo que el jugador utilice diversas estrategias para superarlos.
- **Elden Ring** y **Warhammer 40k: Space Marine 2**: Estos juegos modernos incorporan *bosses* con patrones de ataque complejos y fases diferenciadas, desafiando al jugador a adaptar constantemente sus tácticas.

Este análisis ayudó a extraer elementos clave en el diseño de enemigos que se integrarían en *Platyfa*, buscando crear una experiencia equilibrada y emocionante para el jugador.

7.3.2. Diseño conceptual de los enemigos en *Platyfa*

Tras el análisis, se procedió a la fase de diseño conceptual de los enemigos, definiendo dos tipos principales: dingos y búhos. Estos enemigos fueron seleccionados por ser depredadores naturales del ornitorrinco en Australia, lo cual añade coherencia y realismo al contexto del juego. Cada tipo posee características visuales y comportamentales diseñadas para desempeñar roles específicos en el juego, basados en los aprendizajes obtenidos en la fase anterior. Esta elección no solo enriquece la narrativa, sino que también permite que los enemigos reflejen amenazas reales para el personaje principal, aportando una dimensión adicional al enfrentamiento entre especies.

1. **Dingos**: Enemigos rápidos y ágiles que atacan en grupo, lo cual exige que el jugador se mantenga en movimiento y emplee ataques a distancia. Su diseño se centra en mantener la acción constante y generar tensión.
2. **Búhos**: Con la capacidad de volar y atacar desde diferentes ángulos, estos enemigos fuerzan al jugador a tener una visión más amplia del entorno y mantenerse alerta en todo momento .

Cada uno de estos diseños pasó por múltiples iteraciones (ver Anexos figs. 13.1 and 13.5), donde se realizaron ajustes tanto en los elementos visuales como en las mecánicas de juego. Estas iteraciones buscaron afinar un equilibrio entre el nivel de desafío y la jugabilidad, asegurando que cada enemigo no solo complementara el entorno del juego, sino que también ofreciera una experiencia variada y retadora. La combinación de estos elementos ha dado como resultado un diseño que maximiza el potencial del jugador para disfrutar, aprender, dominar las mecánicas del juego y comprender la historia, creando una experiencia de juego inmersiva que invita a la exploración, la estrategia y la mejora constante.

7.4. Definición de comportamiento del jugador

Luego de haber decidido cómo se verán los enemigos y comenzar el proceso de diseño, es posible iniciar a definir de qué manera actuarán los antagonistas a lo largo de los niveles. El comportamiento del jugador en *Platyfa* se define en función de sus interacciones con los enemigos principales, el Dingo Pirata, el Boss Dingo, y los Búhos Soldados. Cada uno de estos enemigos presenta desafíos específicos y fases de combate que requieren diferentes estrategias y habilidades del jugador; elementos cruciales que deben definirse **antes** de comenzar con la programación de cualquier tipo. A continuación, se detalla el comportamiento esperado del jugador para interactuar eficazmente con cada tipo de enemigo.

7.4.1. Interacciones con el Dingo Pirata

El Dingo Pirata utiliza ataques a distancia y la destreza para manterer esa distancia, lo que requiere que el jugador alterne entre esquivar y utilizar herramientas para detener al enemigo. Las interacciones clave incluyen:

- **Ataque a distancia**: El Dingo Pirata lanza su espada en un ataque tipo boomerang. El jugador debe esquivar este ataque manteniendo la distancia o buscando cobertura.
- **Enredar al Dingo**: En el primer nivel, el jugador puede utilizar mecánicas de enredo para incapacitar temporalmente al Dingo Pirata, dejándolo vulnerable a más disparos o dando tiempo para retroceder y atacar desde mas lejos.
- **Derrota del Dingo Pirata**: Para derrotar al Dingo, el jugador debe realizar diez ataques de barringtonia para un daño gradual y así eliminar al enemigo.

7.4.2. Interacciones con el *boss* dingo

El Boss Dingo presenta un mayor desafío, siendo una versión mejorada del Dingo Pirata, con una estructura más robusta y ataques adicionales. El comportamiento del jugador debe adaptarse de la siguiente manera:

- **Estrategias de combate**: Dado que el Boss Dingo posee una segunda espada y un tamaño mayor, el jugador debe esquivar y atacar en intervalos precisos.
- **Ataques y enredo**: A diferencia del Dingo Pirata, el Boss puede liberarse del enredo despúes de un tiempo (6 segundos) o luego de recibir un máximo de 4 golpes. Para avanzar en el combate, el jugador debe acertar un total de 12 golpes para debilitar al Boss y provocar un *cambio de fase* donde aumentará la agresividad del Boss.
- **Uso del entorno**: El jugador puede interactuar con varias cajas para dañar al Boss Dingo cuando este se encuentre en su segunda fase. Esto requiere que el jugador explore el área para localizar las cajas y destruir $\frac{3}{4}$ del total, activando una en la que el personaje principal, Gaus, aprovecha la oportunidad para derrotar al Boss Dingo.

7.4.3. Interacciones con los búhos soldados

Los Búhos Soldados son los enemigos principales del segundo nivel y poseen un comportamiento aéreo y comienzan en un estado dormido, activándose al detectar la presencia del jugador cuando se acerque demasiado a ellos. Las interacciones clave incluyen:

- **Activación y patrullaje**: Inicialmente dormidos, los Búhos despertarán y atacarán si el jugador se acerca demasiado. Si el jugador sale de su campo visual, el Búho seguirá patrullando en el área asignada, permitiendo al jugador opciones de sigilo para evadir o preparar el combate.
- **Ataque con ballesta**: Cuando el jugador está en el punto de mira del Búho, este comenzará a disparar una ballesta, con un ciclo de recarga de 2-3 segundos. El jugador debe esquivar los proyectiles o usar el entorno como protección.
- **Explotar las debilidades**: El jugador puede usar tres elementos para atacar al Búho:
  1. **Barringtonia**: Este ataque realiza un daño normal al Búho, por lo que deberá darle 10 golpes para eliminar a uno.
  2. **Espagueti**: Permite enredar al Búho, haciéndolo caer al suelo donde permanece incapacitado por 3-4 segundos. La caída puede causar daño adicional dependiendo de la altura.
  3. **Gelatina**: Al impactar al Búho con gelatina, se ralentiza sin causarle daño, permitiendo al jugador moverse con ventaja.

El jugador tiene la opción de enfrentarse directamente a los enemigos o emplear tácticas de sigilo para evitar encuentros innecesarios. Esta flexibilidad en el comportamiento del jugador permite una experiencia de juego adaptativa, donde cada enfrentamiento puede abordarse de acuerdo con el estilo y las estrategias personales del jugador. Pero no todo es tan sencillo como suena; ya que, los Búhos Soldados serán un poco más inteligentes que los Dingos, gracias a una red neuronal con la que se adaptarán basada en el desempeño del jugador.

7.5. Capacitación en inteligencia artificial

Para mejorar las habilidades técnicas necesarias para implementar un sistema de inteligencia artificial (IA) robusto, se realizó un curso especializado de IA en la plataforma *Udemy*, centrado en el uso de redes neuronales en el motor de desarrollo Unity. Este curso abordó los principios del aprendizaje supervisado y no supervisado, y cómo estos pueden aplicarse al diseño de enemigos en videojuegos [74].

La capacitación incluyó módulos sobre la creación de agentes controlados por IA que aprenden y adaptan su comportamiento en tiempo real, ajustando la dificultad y el desafío que representan para el jugador. Este enfoque fue fundamental para implementar la capacidad de los enemigos en Platyfa de reaccionar y adaptarse a las acciones del jugador, lo que proporcionó una experiencia de juego dinámica y personalizada.

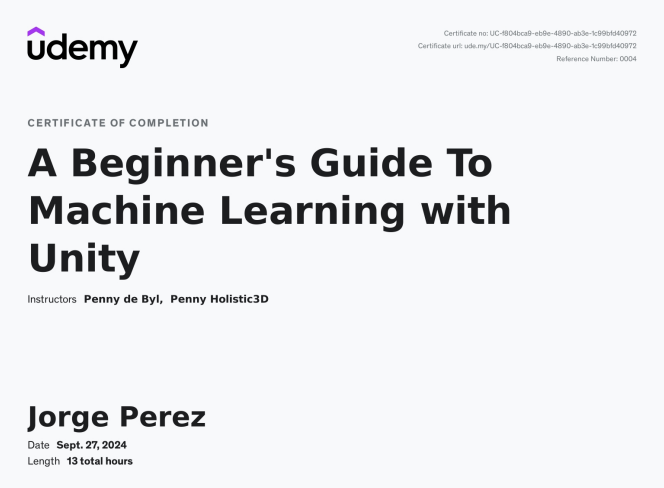

Figura 7.6: Certificado obtenido de la plataforma Online *Udemy*

7.6. Desarrollo y programación de los enemigos

La programación de los enemigos en *Platyfa* se dividió en dos fases principales: el desarrollo de comportamientos básicos predefinidos y la integración de un sistema de aprendizaje basado en redes neuronales. Esta estructura de programación asegura que los enemigos presenten un reto adaptativo y escalable a lo largo del juego.

7.6.1. Fase 1: Algoritmo de programación de enemigos de nivel 1

En la primera fase, se desarrollaron algoritmos específicos para los enemigos básicos del nivel 1, particularmente los Dingos. Estos enemigos siguen comportamientos predefinidos y están programados para reaccionar a la proximidad del jugador. A continuación se describe el algoritmo de programación básico para los enemigos del nivel 1:

- **Dingos**: Cuando el jugador entra en el rango de visión de un Dingo y este mismo puede verlo sin algún obstáculo de por medio siempre dentro del cono de visión, este cambia de estado a *persecución* y avanza en su dirección. Si el jugador entra en el rango de ataque del Dingo, el

enemigo lo marca como objetivo, pasando del estado de *persecución* a *ataque*. Si la distancia entre el Dingo y el jugador es acortada, el dingo retrocoderá para poder mantener una distancia que le permita seguir disparando sus espadas; si la distancia es mayor, realizará un ataque de largo alcance.

7.6.2. Fase 1: Algoritmo del Boss del primer nivel

El Boss Dingo en el primer nivel representa una versión avanzada del Dingo estándar, con comportamientos y ataques adicionales. A continuación se describe el algoritmo para su programación:

- **Ataques mejorados**: El Boss Dingo utiliza dos espadas, disparando ambas con una fuerza y velocidad mucho mayor que el un Dingo normal, sin mencionar que irán una seguida de la otra. En su estado *normal*, el Boss realiza este ataque dependiendo de la proximidad del jugador. Además, si el jugador intenta enredarlo, el Boss puede liberarse después de 6 segundos o un máximo de 4 golpes.
- **Segunda Fase**: Al recibir daño continuo, el Boss entrará en una segunda fase, durante la cual se volverá más agresivo y aumentará su velocidad. Debido al cambio, la barringtonia del jugador ya no le hará daño al Boss, por lo que deberá buscar diferentes maneras para derrotarlo.
- **Uso del entorno**: En esta segunda fase, el jugador debe aprovechar el entorno para atacar al Boss. Si el jugador destruye $\frac{3}{4}$ de las cajas cercanas, se activa una *cutscene* en la que el personaje principal, Gaus, aprovecha la situación para atacar al Boss Dingo mientras está debilitado.

7.6.3. Fase 2: Integración de inteligencia artificial mediante redes neuronales

La segunda fase del desarrollo consistió en integrar una red neuronal para adaptar el comportamiento de los enemigos en función de la habilidad del jugador. Esta red neuronal fue diseñada para analizar los datos de desempeño del jugador, como el tiempo de eliminación de enemigos y la cantidad de enemigos que han sido derrotados, ajustando así el nivel de dificultad.

- **Entrenamiento de la red neuronal**: La red se entrenó usando datos recolectados del rendimiento del jugador en eliminar a los enemigos. Por ejemplo, si el jugador derrotaba rápidamente a los Búhos, el sistema aumentaba la dificultad activando enemigos adicionales en encuentros futuros.
- **Ajuste de la dificultad**: En caso de un rendimiento bajo del jugador, la red neuronal ajusta los parámetros de dificultad para ofrecer una experiencia equilibrada y evitar frustraciones.
- **Implementación**: La red neuronal se implementa en tiempo real, evaluando constantemente el desempeño del jugador y ajustando los patrones de comportamiento de los enemigos de manera dinámica.

Diagrama de funcionamiento de la red neuronal

A continuación se muestra un diagrama que representa el flujo de funcionamiento de la red neuronal para adaptar el comportamiento de los enemigos.

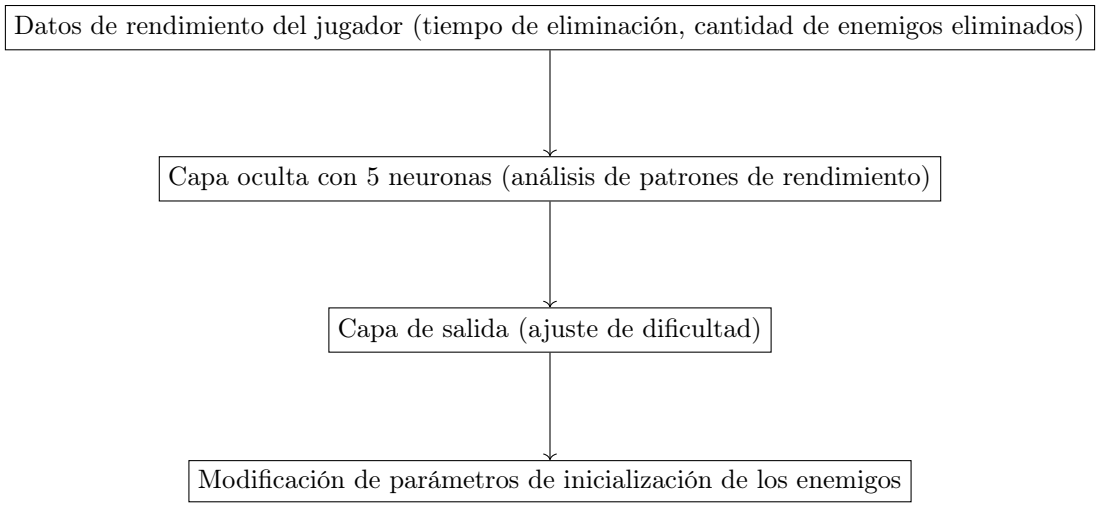

Figura 7.7: Diagrama de flujo de la red neuronal para el ajuste de dificultad en la inicialización de enemigos

Este flujo ilustra cómo la red neuronal procesa los datos de rendimiento del jugador, analiza patrones en una capa oculta y luego ajusta los parámetros de dificultad en la capa de salida, modificando en tiempo real el comportamiento de los enemigos.

Con este sistema, los enemigos presentan un desafío adaptativo, respondiendo al estilo y habilidad del jugador para mantener una experiencia de juego dinámica y equilibrada.

7.7. Diseño de personajes y objetos

Durante la escritura de la historia se definió a todos los personajes que tendrían cualquier tipo de participación en la historia, determinando su personalidad y su diseño a grandes rasgos. Los personajes creados pueden ser descritos en las siguientes categorías: Personaje principal, aliados ornitorrincos, enemigos, y jefes. Para cada uno de ellos, se tomó en cuenta las bases del diseño de personaje, pensando siempre en su propósito, obteniendo información e inspiración para su diseño, y tomando en cuenta la facilidad de reconocer su diseño y silueta. Con esto en cuenta se realizaron bocetos y pruebas de su diseño, buscando mantener consistencia de diseño y lógica dentro del ambiente de la historia. A su vez, por la naturaleza del proyecto (limitaciones de tiempo y cantidad de personal involucrado) también se buscó que el diseño de todo los personajes fuese simple y limpio para facilitar el proceso de modelado, y que tuviese una cantidad moderada de vértices para mejorar su rendimiento dentro del juego, debido a que una mayor cantidad de vértices requiere más cálculos que se deben de realizar para renderizar los modelos. Además de que es una buena práctica el no utilizar modelos demasiado pesados (millones de vértices) (Unity, 2016). [142]

7.7.1. Gaus, el protagonista

Se inició creando muchos bosquejos rápidos para poder definir y transmitir una idea fácil y comprensible al resto del equipo. Además de también determinar el estilo de caricatura simple del que todos los personajes estarían basados. Tal como se indicó desde la primera premisa, el personaje principal sería un ornitorrinco llamado Gaus. Su personalidad inventiva y culinaria se vería reflejado en su atuendo. Como inspiración se utilizó la estética *Steampunk*. Estas primeras ideas y pruebas pueden observarse en la Figura 7.9.

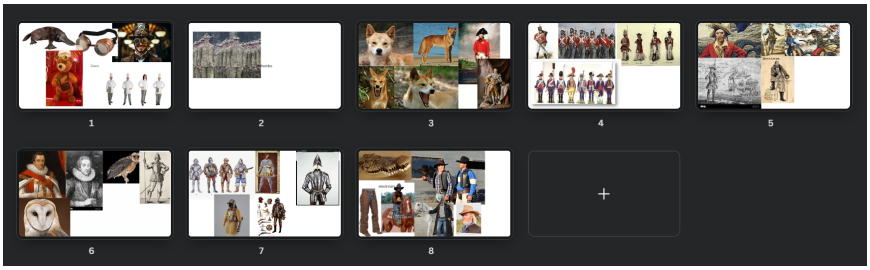

Figura 7.8: Referencias de todos los personajes

Figura 7.9: Diseño inicial de gaus

Para la primera iteración formal del diseño de Gaus, que puede ser apreciado en la Figura 7.10, se buscó tener formas y elementos básicos fáciles de modelar, pero aún manteniendo los elementos básicos de su diseño como su gorro, gafas y delantal. La primera iteración fue descartada debido a que su forma poseía demasiados elementos que podían ser interpretados como cuadrados. La forma cuadrada puede representar un balance y fuerza, pero también puede representar inflexibilidad (Naghdi, A., 2021)[103]. Al ser el personaje principal se buscaba un personaje más accesible y amigable para los jugadores. por lo que se optó por cambiar la forma a una más circular. Al hacer este cambio se permite que el personaje principal sea interpretado de esa manera gracias a la redondez y forma orgánica de los círculos. (Naghdi, A. 2021). Estos cambios se pueden observar en la Figura 7.10, que mantienen la estética general, pero alteran la forma completa del personaje.

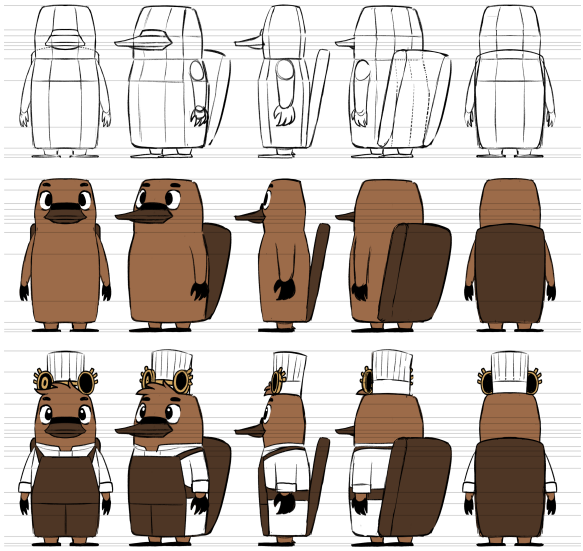

Figura 7.10: Primer diseño para modelar a Gaus

Con el segundo diseño oficial, presentado en la Figura 7.11 también se buscó tener un personaje fácil de modelar en 3D tomando en cuenta varios tipos de movimiento como el “agacharse”. Otro aspecto que se tuvo en cuenta durante el diseño fue la silueta del personaje. Esta es de suma importancia debido a que, al igual que las formas básicas, esta transmite las características más básicas del personaje. Y para que un personaje sea verdaderamente memorable, una silueta única y reconocible es indispensable. (Pulse College, s.f.). [111] Gracias al pico y cola del personaje principal ya se presentaban características reconocibles, pero también se hizo uso de la forma del gorro de chef, detalles en los lentes y las mangas del traje para agregar más forma y distinguirlo de mejor manera de otros personajes similares. La silueta puede observarse en la Figura 7.12.

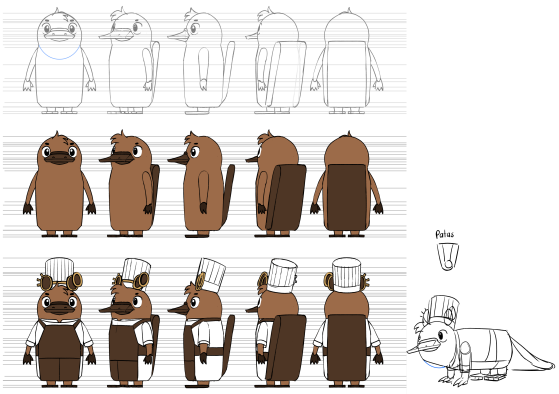

Figura 7.11: Segundo diseño para modelar a Gaus

Figura 7.12: Silueta de Gaus

Con el diseño establecido, se prosiguió a compartirlo con los otros miembros del equipo para que se creara el modelo 3D y el *rigging* del personaje principal.

7.7.2. Enemigos

Etapas similares de diseño fueron utilizadas para los enemigos. Se crearon varias iteraciones y bosquejos de los mismos, cada una realizando distintas modificaciones para tener distintos resultados y evocar distintos sentimientos y opiniones en el jugador. Además, para cada uno de los grupos enemigos, con fines de ser distinguibles y únicos, se le determinó a cada uno de ellos un estilo y estética distinta que determinaría su vestimenta, características especiales, y estética del nivel.

Dingos

El diseño de los enemigos de tipo dingo es crucial, ya que representan el primer desafío que el jugador encuentra y, por tanto, son fundamentales para establecer la primera impresión del juego. Al ser los primeros oponentes, su diseño no solo debe captar la atención del jugador, sino también comunicar claramente las mecánicas básicas, el estilo visual y el tono general del juego. A través de sus características, los dingos permiten al jugador experimentar las dinámicas de combate, las opciones de interacción y el nivel de desafío esperado, brindando así una introducción intuitiva a la experiencia de juego. Su diseño, por lo tanto, actúa como un punto de referencia inicial que prepara al jugador para los desafíos y estilos de juego que se encontrarán en niveles posteriores.

La primera versión de su diseño buscaba indicar que estos eran malvados y se decidió utilizar picos, ya que estos indican peligro. También se usaron colores fríos y apagados con el mismo objetivo. Pero el diseño poseía demasiadas curvas y formas circulares y ovaladas, que les daba una apariencia un poco tierna. Por lo que fue descartado.

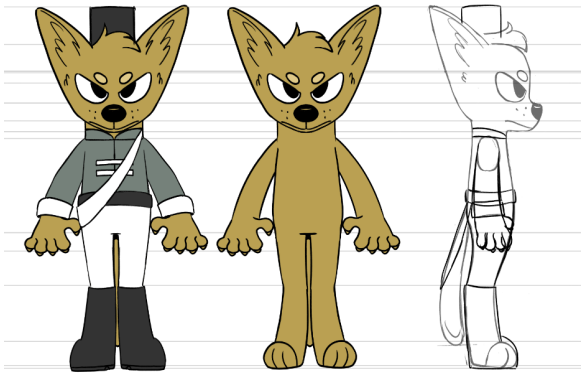

Figura 7.13: Primer diseño de los enemigos de tipo dingo

El segundo diseño, que se puede observar en la Figura 7.14 , buscó construir un poco sobre el anterior y cambiar la estética a una más brutal, como la de un pirata. Pero nuevamente sufría de los mismos problemas, presentando al enemigo con muchos elementos redondos (especialmente en la zona de la cabeza y la cara) que no permite expresar mucho peligro.

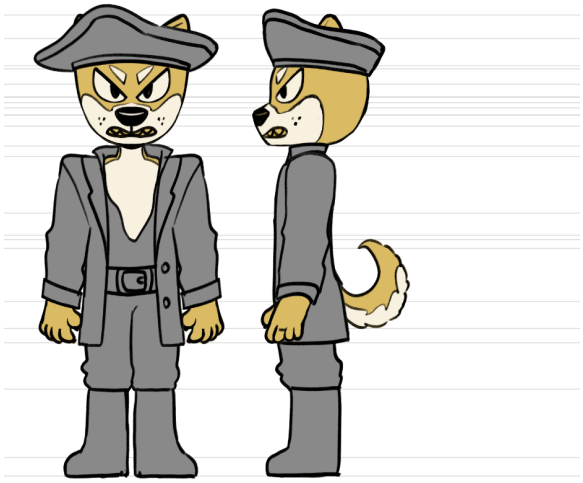

Figura 7.14: Segundo diseño de los enemigos de tipo dingo

El tercer diseño buscó otro acercamiento, utilizando más formas triangulares y con picos para indicar el peligro que presentan estos enemigos, siempre teniendo en cuenta la limitación de vértices anteriormente mencionada. Las proporciones del cuerpo fueron modificadas para hacerlos ver más rudos y fuertes, como se observa en la Figura 7.15.

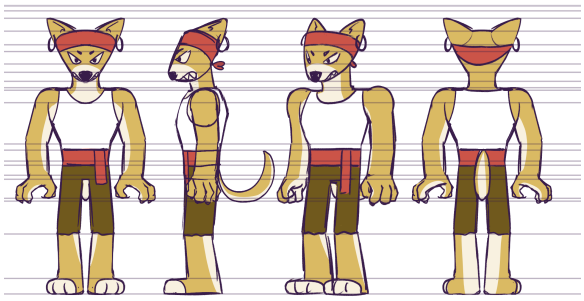

Figura 7.15: Tercer diseño de los enemigos de tipo dingo

El tercer diseño del enemigo fue transferido a los otros miembros del equipo de trabajo para realizar el modelo 3D y *rigging*.

Jefe Dingo

A partir de los enemigos base, se prosiguió a definir al jefe del nivel, el cual sería una versión más grande y fuerte de los enemigos base. Como se puede observar en la Figura 7.16, el jefe posee un aspecto similar al de los dingo base, pero para verse más poderoso es mucho más alto. Mantiene unos colores similares a los otros dingos, nuevamente utilizando el rojo para evocar peligro en los jugadores. Su aspecto es más similar al de un capitán pirata, utilizando elementos como su traje, parche y sombrero. Al ser capitán, destaca entre los otros dingos, que llevan vestimenta de tripulación de barco pirata. Este diseño también fue enviado al resto del equipo para realizar el modelo 3D y *rigging*.

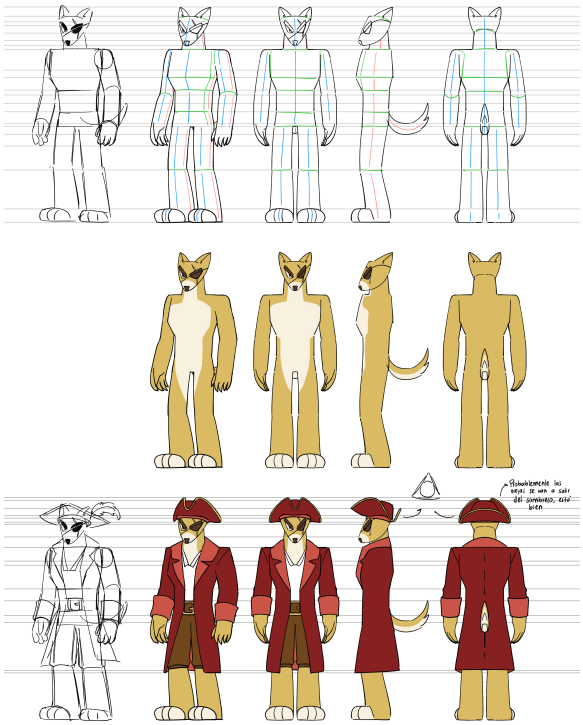

Figura 7.16: Diseño del jefe dingo

Búhos

Para los enemigos de tipo búho, se decidió utilizar como base la forma de los búhos del género *Tyto*. Estos búhos se caracterizan por tener un rostro con forma de corazón y varias especies de ellos habitan en las tierras del continente australiano. Se tomó la idea de la cara en forma de corazón y se mantuvo la simplicidad de la forma del cuerpo para facilitar el modelado 3D. La estética que ellos poseen es la de caballeros de un castillo, por ello se les agregó elementos como armadura y casco.

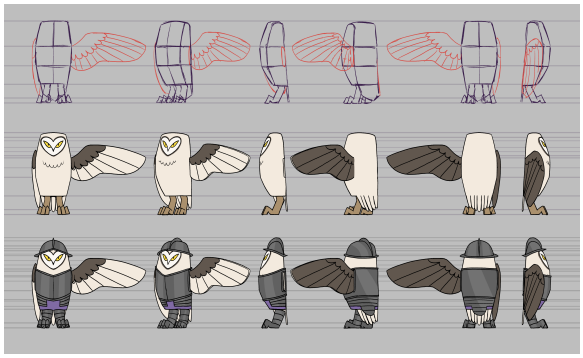

Figura 7.17: Diseño de enemigos de tipo búho

Jefe Búho

En el caso de la jefa búho, el diseño base es similar al de los enemigos normales de tipo búho, manteniendo elementos como la armadura. Pero al contrario de los otros búhos, la jefa toma el rol de monarca y porta una corona. Su diseño también busca demostrar elegancia, por lo que se le agregaron elementos como las plumas en la cabeza y la cola larga, las cuales tienen figuras curvas que pueden ser asociadas a una figura más femenina.

En su diseño, y el de los búhos normales, también se utilizó el color morado debido a su asociación con la realeza. Esta asociación data desde la Edad de Bronce, debido a que en la antigüedad este era posible de conseguir solamente por medio del tinte púrpura de Tiro, producido a base de mariscos murícidos en la ciudad de Tiro. Este tinte era caro de producir, por lo que solo la clase alta tenía los medios de comprarlos, y por ende nace su conexión con la realeza, quienes se daban el lujo de portar ropa de ese color. [150]

Figura 7.18: Diseño de la jefe búho

7.7.3. NPCs

Para los NPC, se tomó como base el modelo de Gaus para facilitar el proceso de modelado 3D. Por ello, se enfocó en su vestuario y las características que los distinguirían del protagonista, especialmente en su sombra y formas generales.

El primer personaje diseñado fue Powel, quien se encuentra en la Figura 7.19. Este personaje tiene un rol importante en la historia como la mano derecha del siguiente personaje, y futuro amigo y compañero de Gaus. Su diseño trató de reflejar un espíritu rebelde y alguien luchador. Esto se puede observar en sus vestimentas inspiradas en la película *Rambo* y las vendas que lleva puestas que cubren golpes de batallas pasadas.

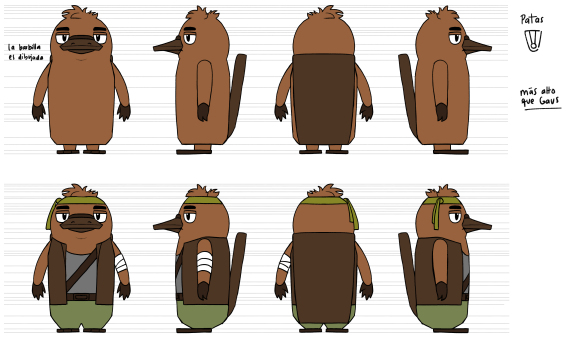

Figura 7.19: Diseño del aliado, Powel

El segundo personaje diseñado de los ornitorrincos fue el jefe de la rebelión, Jol, quien se puede apreciar en la Figura 7.20. Su diseño varía del de Powel, debido a que este quería representar elementos más misteriosos y serios. Por ello se optó darle una vestimenta inspirada en ropa elegante que fuese de combate. Su vestimenta es muy formal, con elementos defensivos, y sus ojos se encuentran cerrados para cuando estos se abran se pueda tener otra sensación y opinión sobre el personaje.

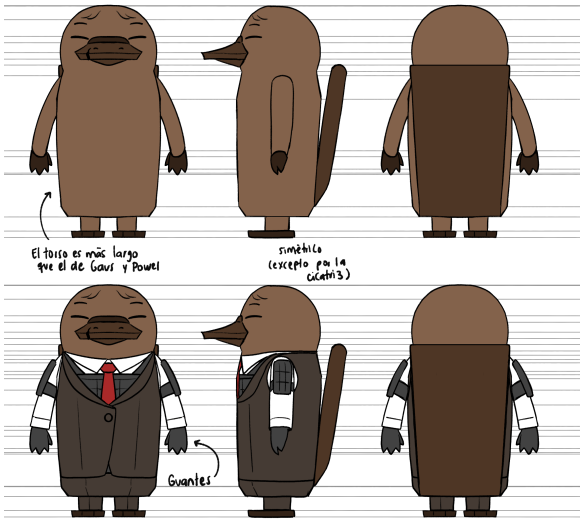

Figura 7.20: Diseño del aliado, Jol

7.7.4. Armas

Para el diseño de las armas se tomó en cuenta dos cosas de suma importancia, la estética y la funcionalidad. El arma más importante a diseñar fue el arma portada por el protagonista, Gaus, ya que esta es la herramienta más importante en el juego, con la cual los jugadores avanzarán por todos los niveles. Debido que es el arma con la que los jugadores se verían más familiarizados durante toda la duración del vide0juego. Para su diseño, primero se buscó inspiración para estética de la misma, haciendo uso de *moodboards* 7.21. El primer diseño, observado en la Figura 7.22, tenía la idea de combinar un cañón de mano y una pistola estilo *steampunk* que esta lanzara comida. Pero rápidamente fue descartada debido a la incompatibilidad entre el diseño y el modo de uso. Especialmente por el almacenamiento de comida en el arma y el pico del ornitorrinco.

Figura 7.21: *Mood board* de inspiración para el arma, primera versión

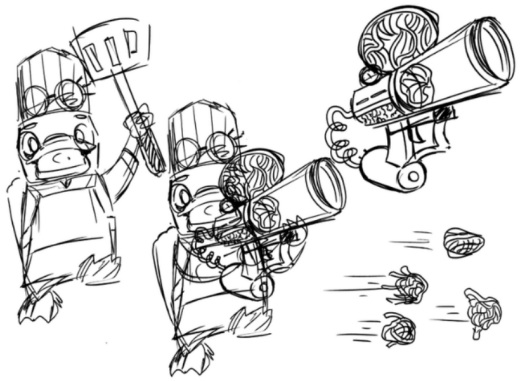

Figura 7.22: Bosquejos del modo de uso del arma diseñada

A partir del fallo anterior, se decidió probar desde otro acercamiento utilizando otro artefacto como base del diseño. Este debía ser más fácil para modelar y se debería poder ajustar para la anatomía del ornitorrinco. Por este motivo se tomó en cuenta como base un soplador de hojas y se mantuvo la estética anterior, pero modificándola a la nueva base. El segundo diseño oficial puede observarse en la Figura 7.24 . En la parte superior se podría observar la munición actual del jugador.

Figura 7.23: *Mood board* de inspiración para el arma, segunda versión

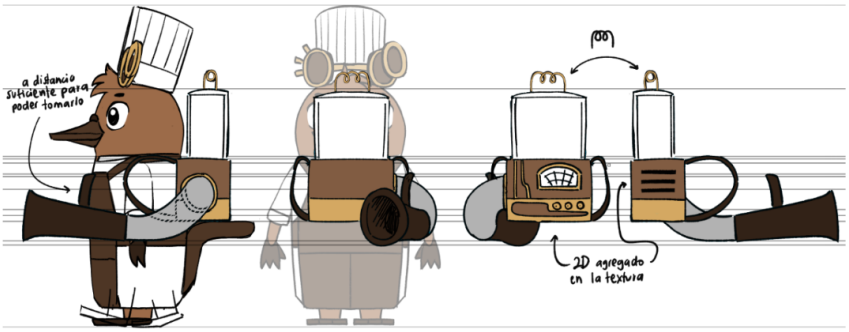

Figura 7.24: Diseño del arma portada por Gaus

En el caso de los enemigos, sus armas fueron diseñadas de acuerdo con la estética de cada grupo de depredadores. Dado que *Platyfa* es un juego de tipo *shooter*, se buscó crear armas de largo alcance que mantuvieran coherencia con la narrativa del juego. Para los enemigos de tipo dingo, inicialmente se exploró el uso de armas simples, como lanzas y otras armas de asta, aprovechando su potencial para ser lanzadas, tal como se muestra en la Figura 7.25.

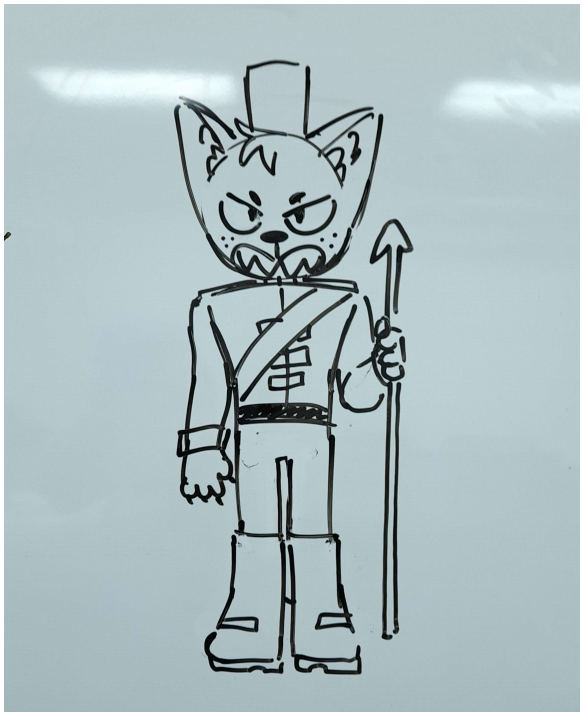

Figura 7.25: Diseño antiguo de enemigo de tipo dingo utilizando una lanza

Sin embargo, al cambiar la temática de estos enemigos a una inspiración de "dingos piratas", se consideraron nuevas alternativas para el diseño de sus armas. Inspirados en juegos como *Breath of the Wild*, se decidió combinar dos elementos, una cimitarra pirata y un búmeran, permitiendo que el arma tuviera no solo un estilo acorde a su identidad de pirata, sino también un mayor alcance al ser lanzada, brindando una experiencia de combate dinámica y variada para el jugador. Se tomó inspiración de armas reales y ficticias 7.26 y se llegó al diseño en la Figura 7.27.

Figura 7.26: *Mood board* de de inspiración de para el arma de los enemigos de tipo dingo. Se busca combinar una cimitarra con un búmeran

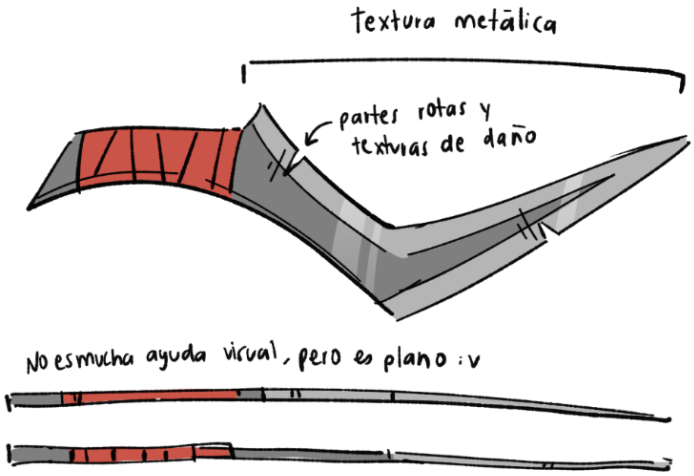

Figura 7.27: Diseño de arma dingo, combinación de cimitarra y búmeran

Para los enemigos de tipo búho, se necesitaba de otra arma de largo alcance que combinara con su temática de castillos y caballeros. Por ello se escogió la ballesta, un de origen desconocido, pero evidencias lo remontan a al menos el siglo 4 en China [26]. A partir de la selección del arma se recopiló inspiración en un *mood board* 7.28 para posteriormente realizar ilustraciones para determinar el diseño del arma. Para el diseño era necesario cambiar la forma base del arma a modo que se adapte a la anatomía de los búhos, los cuales vuelan al momento de atacar. Al volar, tomarán el arma con sus patas. Por ello, en lugar de tener un tablero plano o un mango inferior, se propuso colocar el mango en la parte superior para poder ser sostenido desde arriba. Asimismo, se emplearon formas básicas y sencillas para facilitar el proceso de modelado.

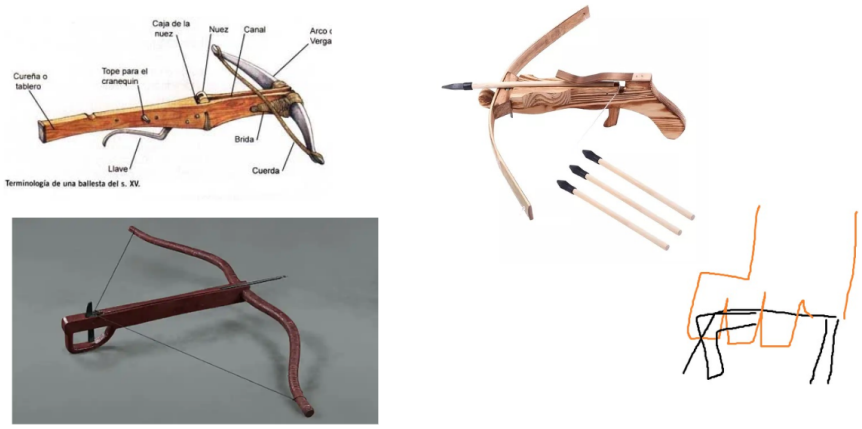

Figura 7.28: *Mood board* de inspiración para el arma de los enemigos de tipo búho. Ballesta

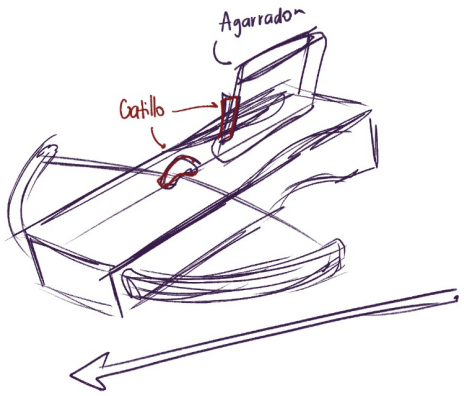

Figura 7.29: Diseño de exploración del arma de los enemigos de tipo búho

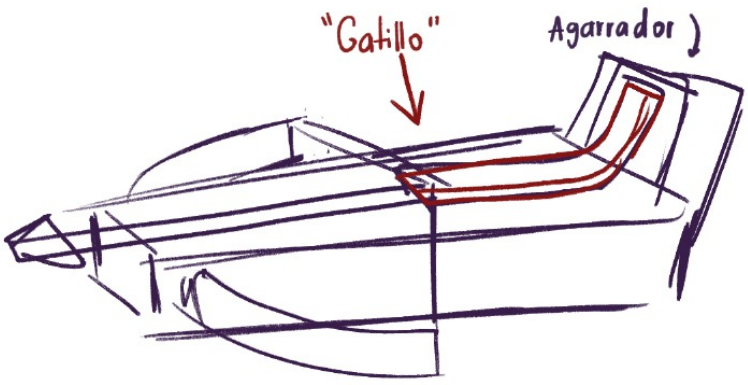

Figura 7.30: Diseño de exploración del arma de los enemigos de tipo búho

7.7.5. Objetos del nivel

Municiones

Uno de los elementos más importantes para cualquier *shooter* es recargar las municiones del arma que portan los personajes. En el caso específico del juego *Platyfa*, todas las municiones se encuentran inspiradas en comida. Como munición estándar se tendría un fruto llamado barringtonia el cual puede hallarse en el continente australiano. Cada nivel introduciría un tipo de munición con el cual se podrá derrotar enemigos y progresar el nivel.

El primer nivel, la introducción, introducirá al jugador al proyectil de tipo espagueti, con el cual se puede inmovilizar enemigos temporalmente, e interactuar con tuercas. El segundo nivel le presentará al jugador las municiones de tipo gelatina para poder saltar en ellas y llegar a lugares más altos, introduciendo el elemento de altura en el nivel. El tercer nivel enseña el uso de *cupcakes* explosivos para derrotar enemigos y destruir objetos. Y por último, el siguiente nivel utilizará té caliente para derretir estructuras e interactuar con el nivel.

Figura 7.31: Diseño de proyectiles

7.8. Modelaje 3D

7.8.1. Preparación previa al modelado 3D

Antes de empezar con el trabajo de modelado en *Blender*, se dedicó un tiempo a obtener conocimientos sobre el modelado 3D. Luego de elegir *Blender* por las razones expuestas en el marco teórica, comencé a investigar sobre herramientas y enfoque general para trabajar en él a través de un curso llamado *Creación de personajes en 3D con Blender* en *Domestika*, donde se desarrolló una base para trabajar con el *software* y técnicas para crear un personaje en 3D.

Aparte de eso, como apoyo se utilizaron otros recursos obtenidos de YouTube que se centraban en vídeos sobre modelado optimizado para videojuegos. Me ayudó a entender mejor la importancia de mantener una cantidad mínima de polígonos y algunos otros conceptos básicos para la optimización de juegos.

Con esa base, se comenzó a hacer modelado simple, tomando un ornitorrinco de plastilina hecho por la hermana de un integrante del equipo como ejemplo inicial, tal como se puede ver en la Figura 7.32. El poder observar y girar el modelo fue de gran ayuda para comprender la forma y el volumen, obteniendo el resultado de la Figura 7.33. En esta fase de aprendizaje se solicitó la ayuda y el consejo del asesor, Aníbal Gramajo, quien me sugirió que intentara ampliar las técnicas de *blocking* para simplificar la geometría inicial de los modelos. El secuaz dingo era un personaje un poco más complejo, por lo que en una reunión el asesor mostró cómo llevar a cabo la técnica de retopología, que luego se investigué más a profundidad y se aplicó para poder optimizar el modelo sin perder calidad visual.

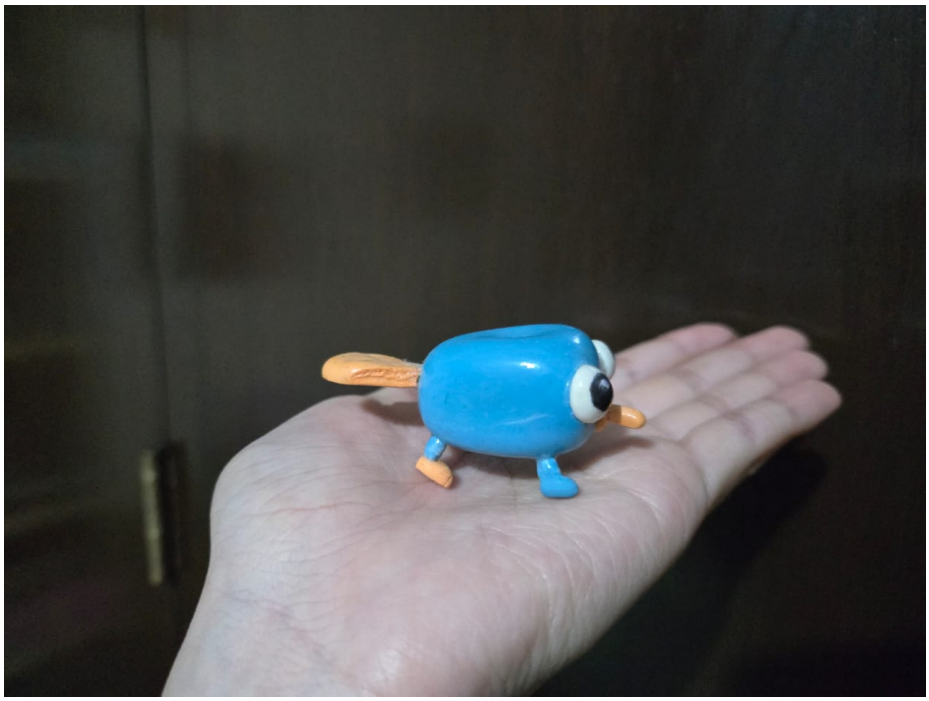

Figura 7.32: Ornitorrinco físico de referencia

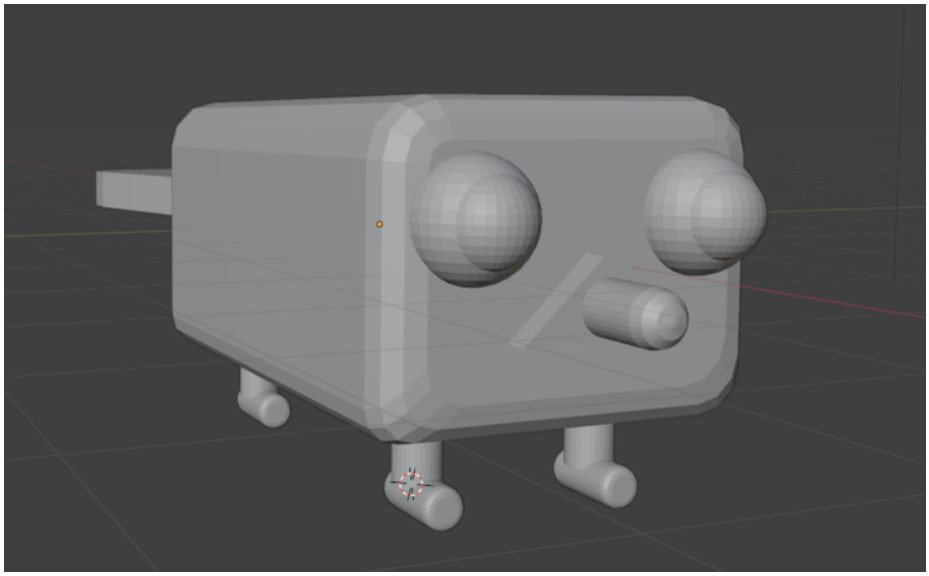

Figura 7.33: Ornitorrinco creado a partir de la referencia

7.8.2. *Blocking*

Para explicar el desarrollo de los modelos dentro de *Blender* se utilizará a Gaus, el ornitorrinco y personaje principal, como ejemplo ya que el proceso fue similar al de todos los otros personajes y objetos. Solo para destacar diferentes enfoques de otros modelos, se mencionarán más adelante cuando sean relevantes.

Se empezó con la técnica de *blocking*, en la que se utilizaron formas primitivas que compondrían la estructura general del personaje y asegurarían que las proporciones encajaran con el arte conceptual establecido anteriormente. Para ello se utilizaron cubos y cilindros, como se puede ver en la Figura 7.35. Es importante mencionar que cuando se agrega una figura en *Blender* se crean con una densidad alta de geometría como se puede notar en la Figura 7.36 , por eso, se le bajó a 8 de geometría al cilindro ya que es un número par y con las suficientes caras para modificar al inicio como se ve en la imagen, para luego modificar los vértices, aristas y caras utilizando el modo Edición de *Blender* y darle la forma que se desea. Se construyó la forma general del cuerpo utilizando un cubo escalado y deformado para obtener una forma simple a partir de la referencia visual. Se utilizaron cilindros para los brazos y las piernas, y para la cabeza una esfera. Este paso fue útil para desarrollar rápidamente una base del personaje 3D y sirvió como guía para el desarrollo de más detalles.

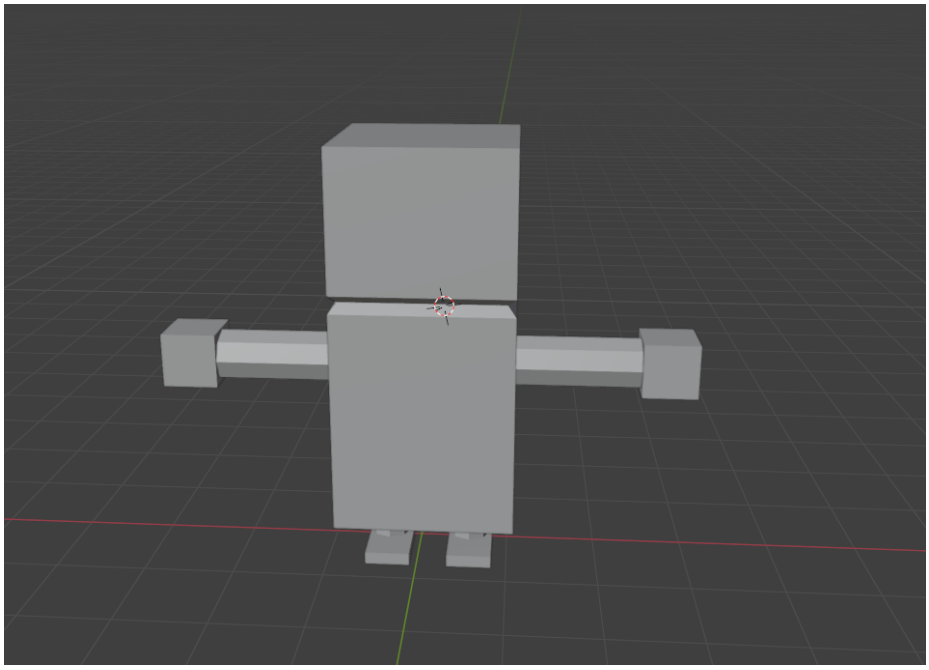

Figura 7.34: Primera fase del *blocking*

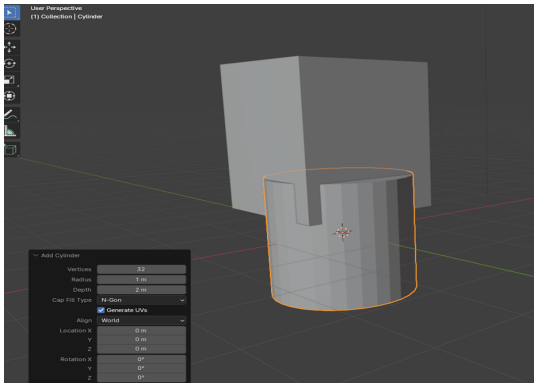

Figura 7.35: Creando figuras

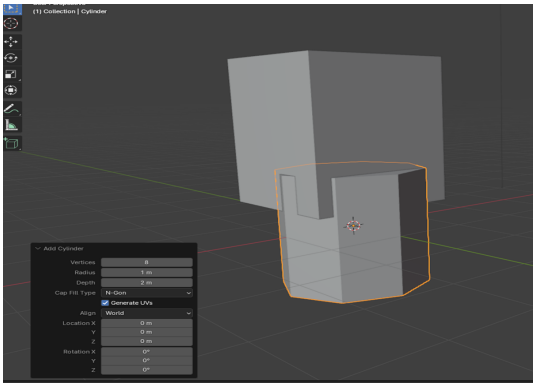

Figura 7.36: Reduciendo su densidad

Después del *blocking* y de agregar detalles, se dividió a la mitad el personaje y se eliminó una mitad de las caras seleccionándolas y usando *Ctrl + X*, seguido a eso, se aplicó el modificador *Mirror*, para poder trabajar en simetría y asegura que ambos lados del modelo sean exactamente iguales. Esto es muy importante en términos de precisión y eficiencia en la etapa de refinamiento. El modificador hará que cualquier ajuste necesario en un lado del modelo se refleje automáticamente en el otro, lo que ayudará a eliminar la necesidad de repetir trabajo, a su vez, aumentará la rapidez del flujo del proceso. Hecho lo anterior, se prosiguió a corregir la pose de Gaus a la pose T, la cual es una pose en la que los brazos están nivelados paralelos al suelo y las piernas están un poco abiertas. Esto se hace ya que más adelante en el desarrollo se debe incluir la creación de un esqueleto simétrico y evitar más problemas en la deformación del modelo.

Con la pose y la estructura primaria en su lugar, se pudo avanzar en el detalle de las transiciones y curvas del cuerpo, haciendo uso de la herramienta Bisel para suavizar los bordes, especialmente en lugares que necesitaban tener un perfil más orgánico y menos angular. Los avances se pueden

notar de la Figura 7.37 a la Figura 7.38. Gracias a las curvas de bisel y sus configuraciones se puede agregar la cantidad de segmentos necesarios. Posteriormente se aplicó el modificador *Mirror* para obtener la malla de la geometría reflejada.

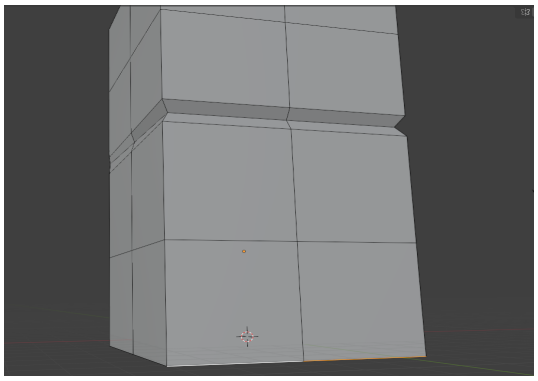

Figura 7.37: Mostrando un borde seleccionado de la figura

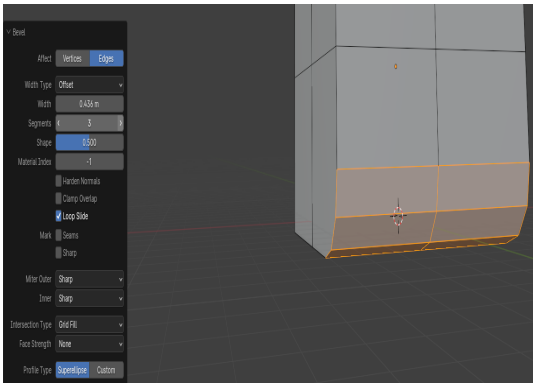

Figura 7.38: Aplicando Bisel

A partir de ese momento se puede pasar a comprobar cómo se ve el flujo general de los bordes del modelo después de aplicar el modificador *Mirror*. En otras palabras, la topología debería fluir naturalmente por todo el modelo sin brechas aplicadas en las áreas de transición entre el lado izquierdo y el derecho. La distribución del flujo de bordes o *edge flow* se puede describir bien utilizando la orientación de los bordes de la malla geométrica. Para lograr deformaciones apropiadas durante la animación y un aspecto fluido y suave del modelo, es importante tener un flujo de bordes apropiado que siga de un vértice a otro. Esto implicaba tener que revisar las áreas más importantes que son las uniones de las extremidades y el cuello. También necesitaba priorizar el uso de cuadriláteros y polígonos de cuatro lados en las áreas que se esperan que tendrán más movimiento en otro módulo de animación.

Se continuó con la mochila de Gaus, y se creó un nuevo archivo de *Blender* para mantener un orden. El proceso fue bastante similar al de Gaus iniciando con el *blocking* mientras se ajustaba proporcionalmente el objeto en relación al modelo de Gaus para garantizar la coherencia visual y la integración en la escena. En este punto cabe resaltar que en la mayoría de los casos, se crearon objetos separados en lugar de tener una malla geométrica para todos, y luego los fueron fusionados todos usando *Ctrl+J* para que se convirtieran en un solo objeto.

En ciertos momentos fue necesario partir de diferentes objetos para mantener una geometría correcta y en otras ocasiones se requirió mantener todo en una sola malla de geometría. Esto era altamente dependiente de los movimientos se le quisieran dar en la animación y se buscó que objetos como la manguera no afectara y deformara la base de la mochila y que se moviera de manera independiente.

7.8.3. Ropa de los modelos

En el caso de algunos modelos, se añadió ropa para crear más detalles. No fue necesario en el caso de Gaus (ornitorrinco) y el secuaz Dingo, ya que el efecto se daba con las texturas. Sin embargo, se añadió en el caso del jefe Dingo, el secuaz Búho y el jefe Búho. Después de realizar lo que describí en la sección de *blocking* seleccioné todas las caras que quería que fueran la ropa, las dupliqué con 'Shift + D' y luego fue escalado un poco la para que se ajustara encima del cuerpo. Se usó 'Ctrl + P' para que la selección fuera un objeto independiente del personaje. Posteriormente se apoyó en función de visibilidad de rayos X, para poder ver las capas debajo de la nueva prenda de vestir

y eliminar mucha geometría que quedaba debajo de la ropa. Si esta se dejaba, tendería a causar colisiones y provocaría distorsiones en las texturas durante la animación. Además, la reducción de la cantidad de geometría mejora el rendimiento del motor del juego al descartar la representación de caras que los usuarios no pueden ver.

7.8.4. Mapeo UV y texturización

Con un modelo simétrico y con los bordes fluyendo con claridad, se seleccionó el modelo realizando un clic en el menú de objetos, y en la configuración del origen, donde se selecciona la opción '*Origin to center of mass*'/'Origen al centro de masa'. Es necesario en este paso aplicar los transformadores. Se procedió a empezar las costuras de marcado o '*mark seem*'. Este es un proceso bastante importante cuando se trata de mapeo *UV* en *Blender*.

Se desplegó el modelo en partes de una manera similar a como se extendería una figura de papel para aplanarlo y luego se procede a dibujar la textura correctamente en el mapa 2D. Esto implicó marcar el modelo en lugares estratégicos donde las costuras no serían tan evidentes en el render final, para lograr una integración visual más limpia. Se aplicaron las costuras en zonas menos visibles como por ejemplo debajo de los brazos y la parte interna de la pierna. Esto minimiza la visibilidad de los cortes después de aplicar la textura y el modelo final se verá suave.

Luego se estableció una costura que cortara todo el cuerpo. Con este corte se permite que la geometría se desenrede en el espacio *UV* en una sola pieza. Se continuó esa costura con la misma lógica hasta los pies y alrededor del cuello. Para la cabeza, que es un área orientada a los detalles y sensible a la expresión, solo aplicó la costura en la parte posterior. En la parte de enfrente de la cara no se realizó ningún corte. Esto garantiza el potencial de aplicación de textura en la cara de una manera no distorsionada o estirada, lo que es muy importante para la demostración final del modelo.

Un punto clave que ayuda a revisar que no esté distorsionada las partes separadas por costuras o saber si se necesita agregar o quitar alguna costura, se puede crear una textura *UV* que nos muestra un mapa *UV* y ayuda a visualizar cómo quedará la textura sin aplicarla. Como se puede ver en la Figura7.39, así se configura en el modo '*Shader*' en *Blender*

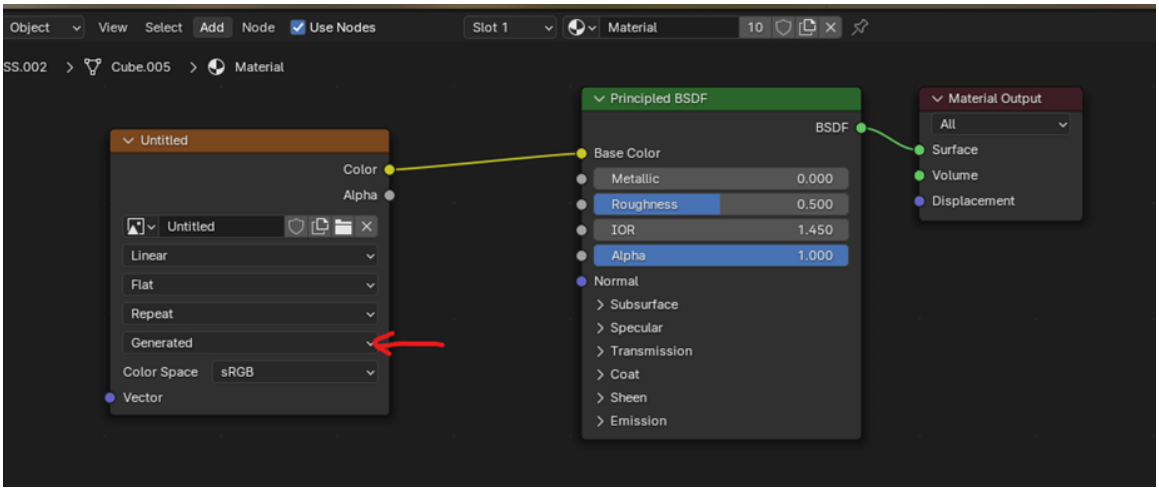

Figura 7.39: Creación de textura *UV* de guía

En la Figura7.40 se puede obserbar lo que ocurre cuando se aplica mal una costura y al desenvolver la malla geométrica; queda extraña.

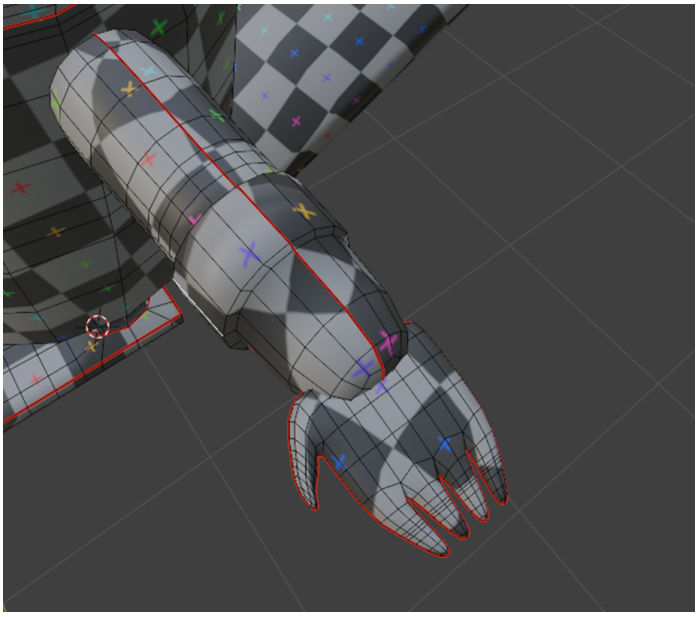

Figura 7.40: Ejemplo de un mal mapeo UV con distorsiones

Pero al compararlo con la siguiente imagen la cual muestra el UV arreglado, se puede notar cómo el brazo es un tipo de cilindro y se tiene que cortar las dos caras y luego separar el cuerpo del cilindro con un corte de costura. En la Figura7.41, entonces se puede notar una gran diferencia la cual es más efectiva si el corte queda en las partes menos visibles Figura7.42.

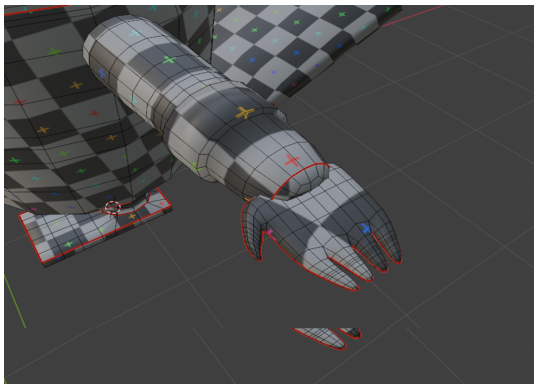

Figura 7.41: Brazo corregido primera perspectiva

Figura 7.42: Brazo corregido segunda perspectiva

Después de todo ese proceso para asegurarse que el mapa UV este correcto, se pudo empezar a texturizar. La imagen de la base del mapa UV fue enviado a un iPad y con ayuda del software de dibujo Procreate se tuvo más control a la hora de pintar la textura. Para este paso fue esencial el uso de un lápiz digital para simplificar bastante el proceso de hacer trazos muy finos y controlados. Primero, se aplicó un color base a las distintas secciones del modelo y se usó la herramienta de "rellenar" para llenar rápidamente cada sección con algunos tonos particulares, basándome en la imagen de referencia que tenía disponible. De esta manera, se hizo uso de la herramienta de cuentagotas para

elegir colores y obtener una paleta que se ajustara perfectamente al diseño conceptual del personaje y cómo quedó como se muestra en la Figura 7.43.

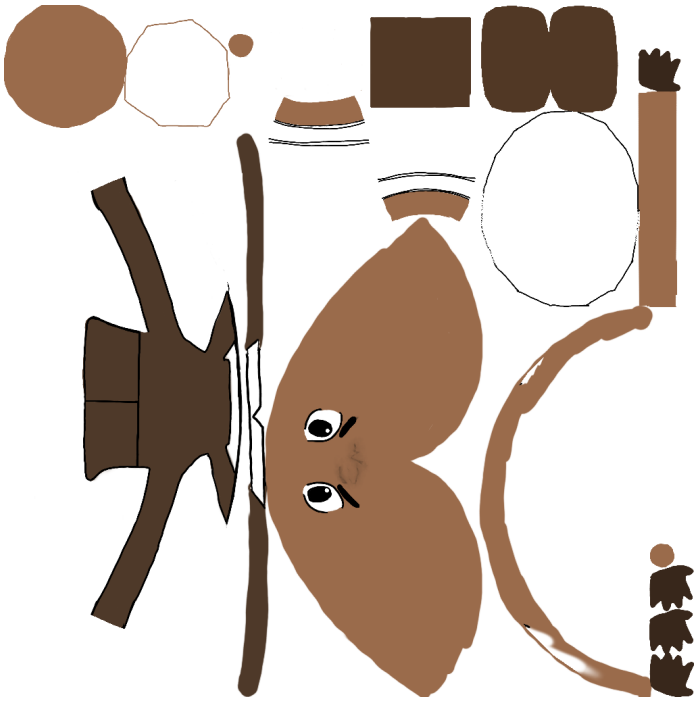

Figura 7.43: Textura final del cuerpo de Gaus (ornitorrinco)

Una vez realizada la texturización, la imagen se importó nuevamente a *Blender*. en donde se realizaron unos ajustes finales con el pincel de *Blender*. Entre ellos, detalles como sombras, especularidad y gradientes directamente en el modelo 3D, de modo que la textura fuera uniforme y sin problemas visuales. En general, este método combinado de uso de herramientas permitió hacer que este proceso fluyera sin problemas, tomando en cuenta el control creativo en cada paso de la textura, ajustando cada herramienta para su mejor uso según las necesidades del proyecto.

Figura 7.44: Textura final cargada de regreso a *Blender*)

7.8.5. Escultura, Dingo secuaz

Para el modelo del secuaz dingo, a diferencia del ornitorrinco, se tuvo que modelar la cabeza mediante una ruta de escultura digital. Todo se debe a que este modelo tiene una forma inusualmente extraña, principalmente en las orejas y la cabeza, y se necesitaba una alta densidad de geometría para modelar correctamente. Se empezó con esferas de alta resolución en *Blender*; la resolución es lo suficientemente alta como para que pueda empujar y tirar de estas áreas sin límites.

Pude ajustar algunas de las características definitorias de la cabeza con el pincel de Agarre en el modo escultura, las orejas y el nudo trasero del pañuelo también que le darían el aspecto de dingo. Aunque el detalle en la parte delantera del pañuelo se logró más adelante con la textura, este enfoque de escultura permitió trabajar con mayor libertad con formas realmente orgánicas y complejas que habrían sido difíciles de lograr mediante el modelado poligonal básico, obteniendo el resultado que se muestra en la figura .

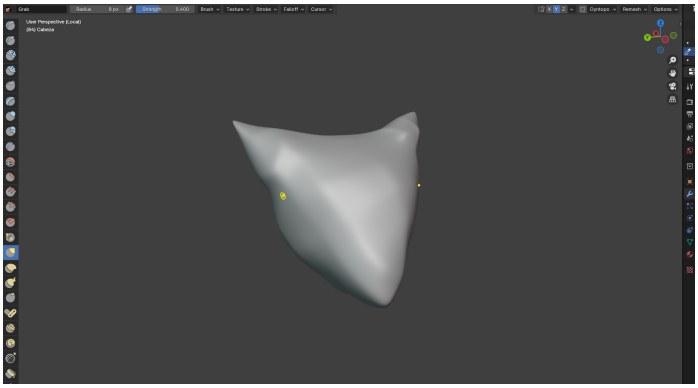

Figura 7.45: Cabeza Dingo con la técnica de escultura)

Continuando con el modelo, se utilizó el modificador '*Mirror*' para modelar simétricamente ambas mitades de la cabeza. También se modeló la cola del dingo por separado, creándola a partir de una curva biselada y ajustándola a la forma correcta. Luego, se convirtió esta curva con la herramienta 'Transformar en malla'. Esto se debe de realizar para cuando se haga el desenrollado lo encuentre correctamente el programa y se aplique la textura.

Una vez hecho el modelo a base de escultura, se procedió a iniciar un proceso muy importante para hacer un modelo para un juego, la retopología. Dado que el modelo estaba esculpido con una alta densidad de polígonos como se puede ver en la Figura 7.46, no era adecuado en términos de rendimiento. Es por eso que creó la versión "*low-poly*"del modelo, al mismo tiempo que mantenía las formas principales y los detalles importantes con una carga de polígonos menor.

Para comenzar con la retopología, primero se colocó un nuevo plano sobre el modelo ya existente. Para pegar el plano al modelo original, se utilizó el modificador *Shrinkwrap* en *Blender*. Esto eventualmente permitió construir una nueva geometría a lo largo de las curvas y contornos del modelo de alta densidad. Se aplicó la operación de extrusión con la tecla 'e' en los bordes para formar nuevas caras repitiendo la operación varias veces hasta cubrir todo el modelo por una geometría nueva y más simple. Después de obtener la versión optimizada, se repitió lo realizado sobre el ornitorrinco.

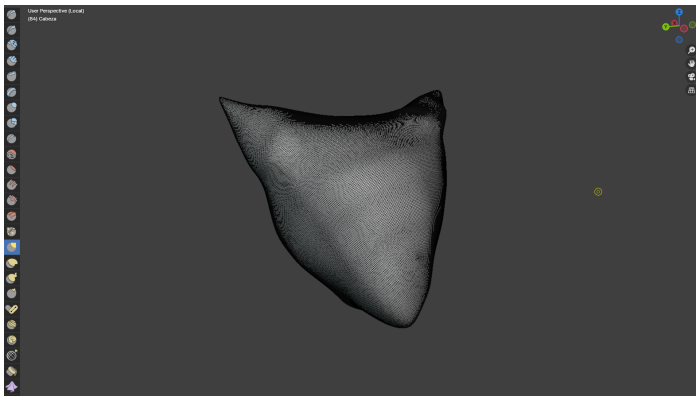

Figura 7.46: La malla de geometría de la cabeza del dingo)

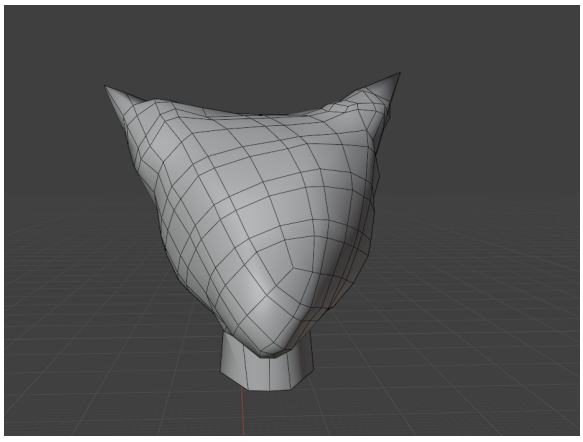

Figura 7.47: Resultado de la retopología en la malla de la cabeza del dingo

7.8.6. Pruebas de integración

Las pruebas de integración son otro paso para verificar que los modelos importados desde *Blender* se comporten correctamente en *Unity*. Esto incluye asegurarse de que los modelos se vean según lo previsto y se rendericen correctamente con sus respectivos *shaders* y que no haya problemas con la física o las colisiones.

Cabe resaltar que todos los modelos se veían de manera en *Unity* menos el Dingo jefe, que al importarlo a *Unity* se pudo detectar un espacio transparente como se puede ver en la Figura 7.48. Debido a eso, se tuvo que modificar las configuraciones en *Unity*, crear un hijo del material principal de la chaqueta, y cambiar el renderizado de caras de *Front* Figura 7.49 a *Both* Figura 7.50 y así no se transparentaba la chaqueta y dando un resultado como la Figura 7.51.

Durante el proceso de animación, también se realizaron ciertos cambios que al aplicar el esqueleto y el movimiento al personaje se hacían notar con mayor facilidad y eran necesario de corregir.

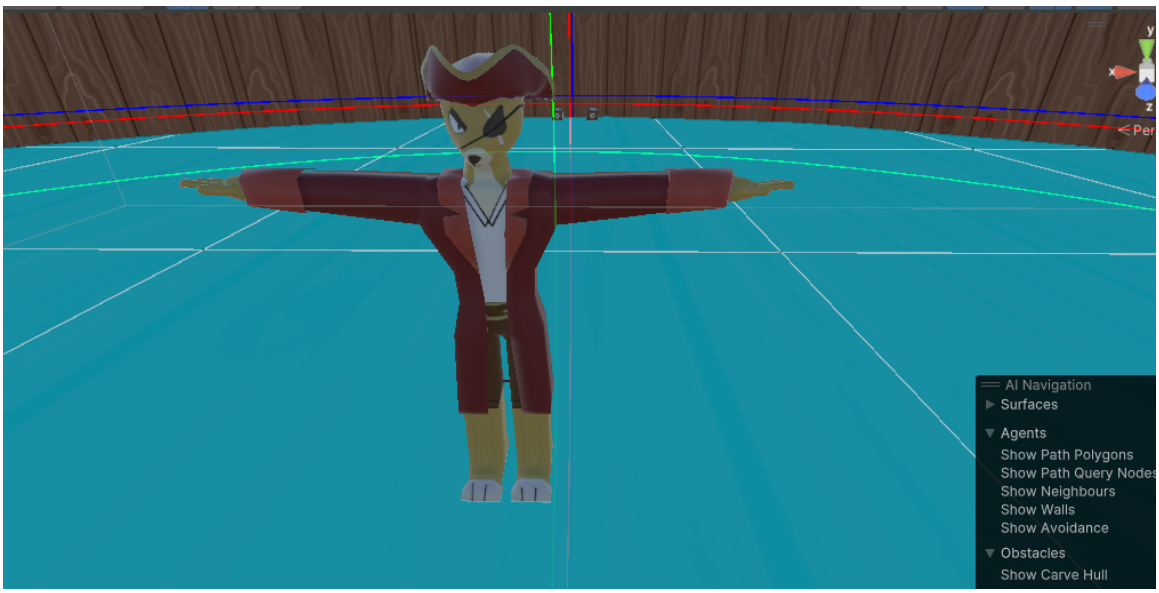

Figura 7.48: Dingo jefe con parte transparente

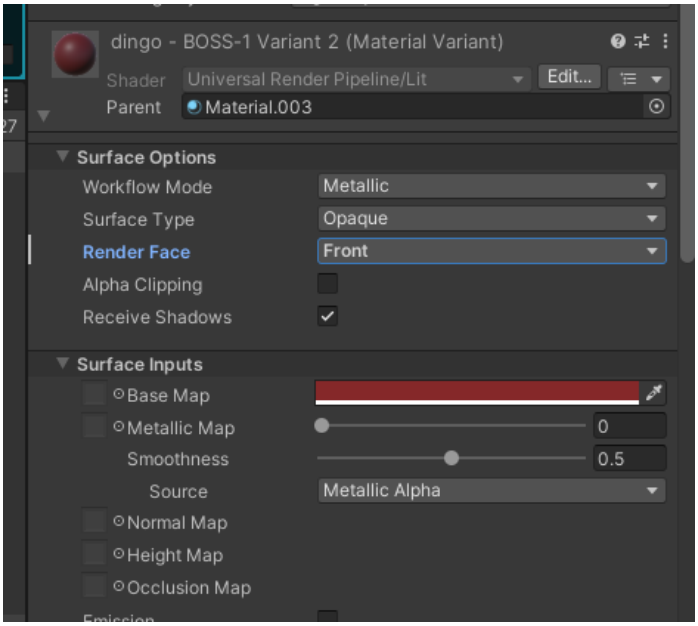

Figura 7.49: Configuración *Front* en material

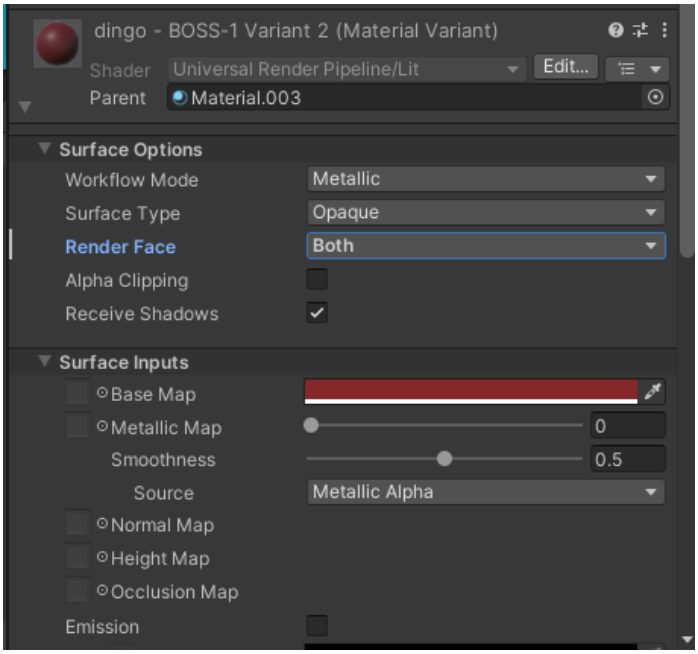

Figura 7.50: Configuración *Both* en material

Figura 7.51: Dingo jefe sin la parte transparente

7.9. Animación 3D

7.9.1. Recopilación de recursos

La primera etapa fue crucial para poder establecer la base creativa y técnica del proyecto. Para ello, se llevó a cabo una recopilación exhaustiva de recursos visuales, conceptuales y narrativos que informarán la dirección artística y el desarrollo de las animaciones.

Arte Conceptual

Para esta parte se realizó una búsqueda de imágenes de referencia que inspiraron el diseño visual de los personajes, escenarios y objetos dentro del mundo de Gaus. Esto se llevó a cabo con la finalidad de poder crear una caracterización más pertinente para cada uno de los modelos que se animaron.

Como se estudió con anterioridad el tener acceso al contexto del personaje, brindó una mejor información para poder caracterizar el mismo e ingeniar las animaciones que se solicitaron en cada modelo.

Guiones y narrativas

Durante este punto se realizó una revisión de los guiones del juego. Esto, porque se pretendió indicar los momentos claves que debían ser reforzados por las animaciones. La sinopsis del videojuego es el siguiente: En un mundo donde los ornitorrincos vivían en paz y habían dominado el arte de la cocina, sus depredadores deciden invadirlos y obligarlos a cocinar para ellos. Uno de estos ornitorrincos, Gaus, ama tanto a la cocina como a sus ingeniosas invenciones. Un día, una de estas invenciones no sale como lo planea, y termina envuelto en un plan para llevar a cabo la revolución Ornitorense. La narrativa fue de importancia ya que para ello se desarrolló una lista preliminar de las animaciones requeridas, ver Figura 7.50.

	Animaciones	IDLE	KO	Caminar	Correr	Saltar	Disparar	Extra
Personajes	Gaus	X	X	X	X	X	X	X
	Dingo	X	X	X	X		X	X
	Dingo Boss	X	X	X	X		X	
	Buho							X

Figura 7.52: Lista preliminar de las animaciones requeridas por personaje

Nota: La columna de EXTRA se refiera a una animación fuera de lo normal. Ejemplo: Búho – tiene una animación de volar. Mas adelante del documento se describe con exactitud la animación de estas.

Referencias históricas y estilísticas

Para este apartado se hizo una investigación analítica de estilos de animaciones caricaturescos que fueran aptos para la visión del proyecto. Además, se consideró cómo los estilos influyen en la percepción del jugador y en la narrativa general.

Como parte de dicha investigación se definió la animación como caricaturesca y tridimensional. Tratándose de que estas mismas cumplieran con algunos de los principios básicos de la animación. Estos fueron los siguientes:

- Estirar y encoger
- Anticipación
- Puesta en escena
- Animación directa y pose a pose
- Acción continuada y superpuesta
- Acelerar y desacelerar
- Arcos
- Acción secundaria
- Exageración
- Atractivo

Estos procesos no solo facilitaron la obtención de una visión clara y compartida entre los miembros del equipo, sino que también se establecieron las bases para decisiones informadas en las siguientes etapas del desarrollo.

Investigación y experimentación con herramientas de animación

Con los recursos recopilados, se procedió a la investigación de las herramientas para llevar a cabo el proyecto.

Las herramientas que se investigaron al inicializar el proceso fueron las siguientes: *Blender*, Mixamo y Unity. Como primera instancia se investigaron los fundamentos de la animación con *Blender*. Aquí surgieron conceptos claves como: Keyframes y Rigging.

Por otra parte, las herramientas de mixamo fueron de ayuda, ya que brindó una solución de animación en 3D. Permitiendo ajustar animaciones de personaje de cuerpo completo capturadas por acores de movimientos profesionales.

Mixamo: Flujo de trabajo

Mixamo, mantiene un flujo mucho más simple y fácil de usar, que cualquiera puede hacer uso de ella sin mucha capacitación o conocimiento. Además, cuenta con una interfaz de usuario más optimizada e intuitiva, ver Figura 7.51.

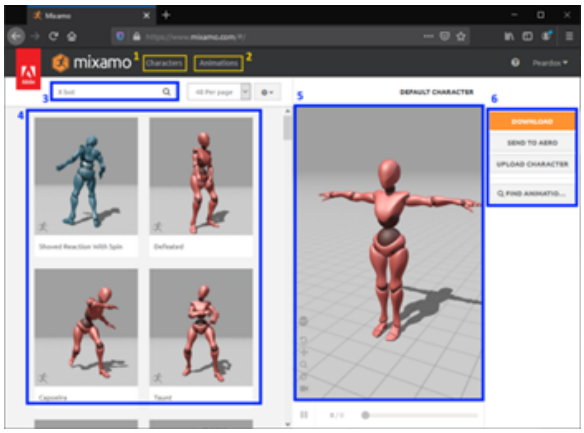

Figura 7.53: Interfaz de Mixamo

Como consiguiente, se experimentó con Unity, uno de los motores de creación de juego más usados en la actualidad [95]. Esta es una herramienta que ayudó con la integración de las animaciones al videojuego como tal. En este se investigaron sobre las herramientas que usa de animación, las cuales son: Componente *Animator* y máquina de estados.

Aparte de esta investigación se procedió a la experimentación con las herramientas de animación, la cual trajo como consiguiente el uso de una nueva herramienta que no estuvo definido en la inicialización de esta. Y este es una herramienta de *Blender* conocida como Add-ons llamada AutoRig Pro el cual ayudó para el procedimiento del Rigging que se presenta más adelante.

Desarrollo de animaciones

En esta sección, se detalla el proceso de desarrollo de las animaciones del personaje principal. Cabe mencionar que los modelos utilizados para el desarrollo de las animaciones fueron brindados por otra compañera como parte del megaproyecto. Posteriormente, se aplicaron los mismos procedimientos a los demás modelos, teniendo en cuenta sus características individuales para lograr animaciones proporcionales y coherentes en cada uno de ellos.

Creación del *rigging*

Para este primer proceso se optó por asegurar que el modelo utilizado viniese en posición de “T”. Esta posición se utiliza comúnmente en las animaciones y modelado por ser una postura neutral antes de aplicar cualquier tipo de animación. Además, que este facilita el establecimiento de un

punto de partida, a verificar la simetria y poder hacer de manera generar una evaluación visual de forma y proporciones para garantizar un Rigging exitoso, ver Figura 7.52. .

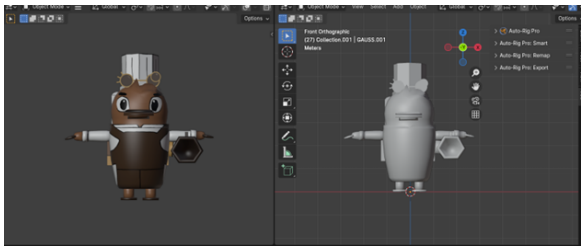

Figura 7.54: Modelo Gaus en posición “T”.

Colocación de marcadores

Después de hacer una breve evaluación visual y colocación del posicionamiento correcto se procedió a la creación del Rigging con el uso de la herramienta Auto-Rig Pro de *Blender*. Para ello se realizó la colocación de marcadores en el cuerpo completo del modelo, ver Figura 7.53.

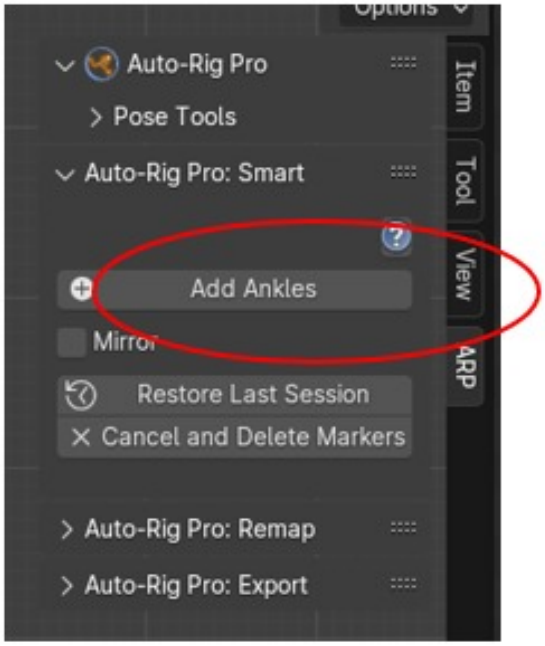

Figura 7.55: Opción de agregación de marcadores para el modelo

Se procedió a colocar todos los marcadores que por defecto ofrece la herramienta de Auto-Rig Pro hasta terminar, ver Figura 7.54.

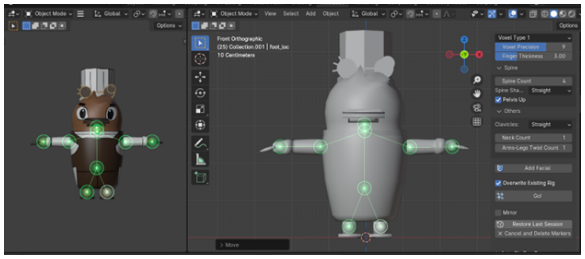

Figura 7.56: Colocación de marcadores corporales en el modelo

Detección automática

La característica de detección automática se realizó por parte del Auto-Rig Pro generando los huesos en el modelo, facilitando la configuración inicial del *rig* (esqueleto) al detectar la estructura básica del personaje y posicionar los huesos principales de la manera más precisa posible.

Durante esta etapa sucedieron varios procedimientos generales. El primero, consiste en que el plugin analiza la forma general del modelo para identificar las partes principales del cuerpo. Continuamente el sistema realizó una detección de las proporciones y alineamientos del modelo para ajustar la posición y así redujo la necesidad del ajuste manual, ver Figura 7.55.

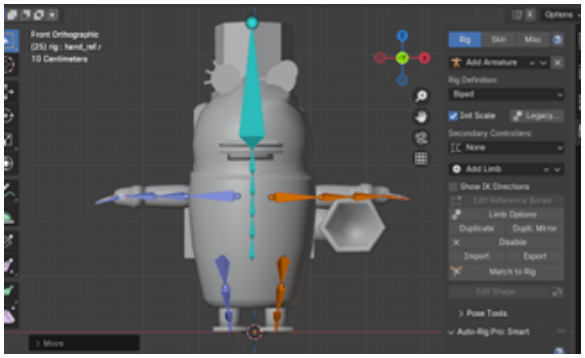

Figura 7.57: Detección automática y asignación de huesos

Vinculación *armature* vs *mesh*

El proceso de Binding es donde el esqueleto (*rig*) se conecta con el modelo 3D o bien el *mesh* (malla). Esto permitió que los huesos controlen la deformación de la malla. En otros términos, este procedimiento es el paso que asoció los vértices del modelo con los huesos del *rig*, para que, al realizar un movimiento a un hueso la malla también se deforme. Durante este proceso suceden acontecimientos importantes. El primero es la asignación de piel. El proceso de vinculación implica aplicar una piel virtual al modelo 3D. Esta piel está compuesta por vértices de la malla, como se mencionó con anterioridad, que se asignan a diferentes huesos del esqueleto. Como segundo punto se generó el pesado de vértices. Aquí, cada vértice del modelo se asocia con uno o más huesos mediante "pesos" que determina cuanta influencia tiene cada hueso sobre ese vértice. Por ejemplo, un vértice en el codo puede estar influenciado en un 70 porciento por el hueso del brazo superior y en un 30 porciento por el hueso del antebrazo. Esto permite una transición suave y natural entre las diferentes partes del cuerpo, ver Figura 7.56.

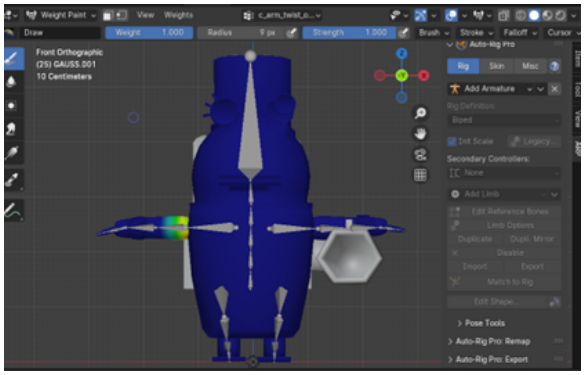

Figura 7.58: Pesado de vértices

Como se muestra en la Figura 21, el color rojo indica el mayor nivel de influencia, con un valor de 1.0. Esto significa que los vértices pintados de rojo se moverán completamente con el hueso correspondiente. Conforme el color cambia a naranja o amarillo, la influencia del hueso disminuye, pero sigue siendo significativa, afectando de manera parcial los vértices. Los vértices pintados de verde tienen un peso moderado, lo que indica que se moverán menos que los amarillos, pero aún responderán al movimiento del hueso.

A medida que el color se aproxima al azul claro, el peso es bajo, lo que significa que los vértices apenas se ven afectados por el hueso. Finalmente, el azul oscuro indica que el peso es cero, y los vértices que están pintados de este color no serán influenciados por el hueso en absoluto, permaneciendo fijos durante los movimientos del *rig*.

Este sistema de colores es clave para asegurar que las deformaciones en la malla sean controladas de manera precisa y que los movimientos resultantes sean fluidos y coherentes con la animación. Este sistema de pesos lo ejecutó de manera optimizada el procedimiento de vinculación.

Ajustes finales/extras

Después del *binding* , el *rig* está completamente funcional y listo para ser animado. Los movimientos que se apliquen a los huesos del esqueleto afectarán directamente la malla, deformándola de manera que se vea natural en la mayoría de los casos. Sin embargo, se ajustó en todos los modelos el esqueleto que este proyecto. En ocasiones la herramienta no nos arroja resultados correctos, por lo cual se realizó una corrección manual. A continuación, en la Figura 22. Se muestra del lado izquierdo la generación del esqueleto incorrecto y del lado derecho su debida corrección, ver Figura 7.57.

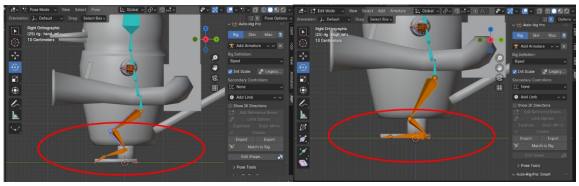

Figura 7.59: Corrección del esqueleto tras vinculación automática

Adicionalmente se agregó un hueso extra como parte del esqueleto base generado. Esto se debe a que Auto-Rig Pro genera el *rig* como si el modelo fuera un humanoide. Sin embargo, el modelo con el que se cuenta es antropomorfo, es decir es aquel que combinan características humanas y animales. Aunque el ornitorrinco es un animal y podría ser considerado como un modelo cuadrúpedo, el

hecho de que esté en dos patas y use sus patas delanteras para disparar como un humano le otorga características humanoides, ver Figura 7.58.

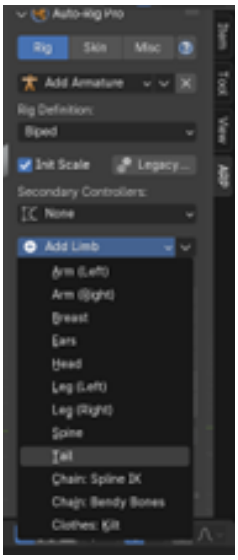

Figura 7.60: Opción de agregación de cola

Además de la colocación de la cola extra al *rig*, también se hizo una asignación de jerarquía de huesos. A este procedimiento se le conoce también como *Parent* o "aparentar huesos". Esto se refiere al proceso de establecer relaciones jerárquicas entre los huesos de un esqueleto (o rig) en un modelo 3D, de modo que los huesos secundarios (hijos) sigan el movimiento de los huesos primarios (padres). En este caso se aparentó en la cola al hueso de la cadera, ver Figura 7.59.

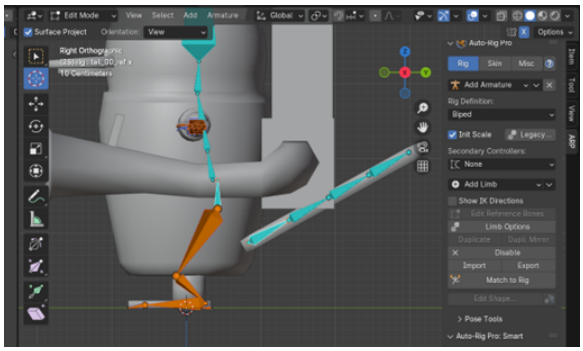

Figura 7.61: Agregación de cola y asignar jerarquía de huesos

Luego de la ejecución de correcciones se seleccionó la opción de Match to *rig* para unificar el objeto con el *rig* y se obtiene el modelo preparado para poder ser animado.

Búsqueda de animaciones base Mixamo

Se utilizó Mixamo como una herramienta clave para la búsqueda y descarga de animaciones base. Mixamo ofreció para el proyecto animaciones predefinidas que sirvieron como punto de partida para la creación de las animaciones específicas requeridas en el videojuego, ver Figura 7.60.

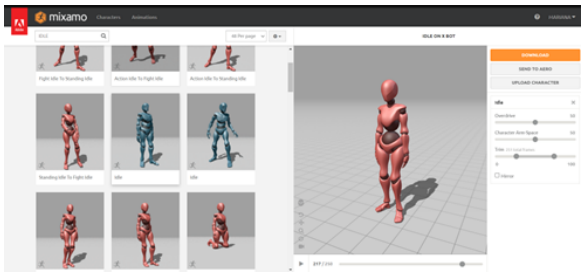

Figura 7.62: Búsqueda de animaciones base en Mixamo

Estas animaciones se adaptaron a los modelos del juego, conservando ciertos aspectos fundamentales, como el posicionamiento y movimiento corporal, mientras que otros componentes se ajustaron o personalizaron manualmente para cumplir con las exigencias particulares de la jugabilidad, como el uso de armas y otras interacciones especializadas. Esto permitió agilizar el proceso de animación sin comprometer la precisión ni la coherencia con las mecánicas del juego.

Descarga y exportación animación Mixamo

En la etapa de descarga y exportación de animaciones desde Mixamo a *Blender*, se seleccionaron animaciones base específicas de la biblioteca de Mixamo que se alinearan con las necesidades del videojuego (IDLE, KO, JUMP, RUN, WALKING, SHOTING, THROWING). Una vez elegidas, las animaciones fueron descargadas en formato FBX, garantizando la compatibilidad con *Blender*.

Este flujo de trabajo facilitó la integración eficiente de animaciones de alta calidad en el entorno de *Blender*, permitiendo una edición precisa para personalizar las acciones y movimientos requeridos en el proyecto.

Unificación de armaduras

En la etapa de unificación de armaduras, se utilizó el complemento Auto-Rig Pro en *Blender* para combinar el *rig* de Mixamo, previamente descargado, con el *rig* personalizado del modelo del videojuego. Para llevar a cabo esta integración, primero se accedió a la pestaña Skin del complemento Auto-Rig Pro. Allí, se seleccionaron ambas armaduras, tanto la de Mixamo como la del modelo propio. Luego, se generó la lista de huesos mediante la opción "*Build Bones List*". Ver Figura 7.61.

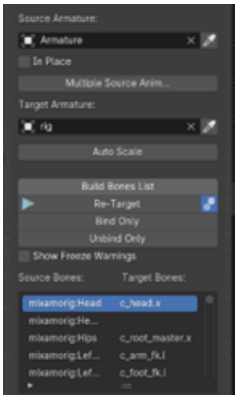

Figura 7.63: Construcción de huesos al unificar armadura

Después de generar la lista de huesos, se seleccionó el hueso base principal *(root bone)* del *rig*, designando el *rootmaster* como "*Set as Root*"para asegurarse de que ambas estructuras estuvieran correctamente alineadas. Una vez realizados estos ajustes, se ejecutó la opción *Re-target* , lo que permitió adaptar la animación de Mixamo al *rig* personalizado.

Finalmente, para vincular la animación al modelo, se seleccionaron tanto la armadura como el modelo y se utilizó la opción Bind, logrando que la animación importada se ajustara de manera óptima al modelo final, manteniendo coherencia y funcionalidad dentro del flujo de trabajo de animación del proyecto.

Edición de animación

En esta área se abordó el proceso de edición de animaciones dentro de *Blender*, donde se realizaron ajustes y personalizaciones a las animaciones previamente importadas y unificadas como se mostrará más adelante. Este proceso es fundamental para la adaptación de las animaciones a las especificaciones del videojuego, garantizando que los movimientos de los modelos se alineen con las mecánicas y dinámicas del juego.

Eliminación/Agregación *KeyFrames* Se llevaron a cabo modificaciones en los *keyframes* , permitiendo el ajuste preciso de las trayectorias y posiciones de los personajes durante las interacciones, como el uso de armas y otras acciones específicas.

Algunos de estos ajustes adicionales involucraron el eliminar por completo los *keyframes* de animación que traía el esqueleto de Mixamo por partes del personaje y realizar las animaciones propias. Un ejemplo de este es para la animación del disparo de Gaus. A continuación, se dará a conocer una de ellas.

Como se observa en la Figura 7.62, se descargó una animación básica de disparo de Mixamo.

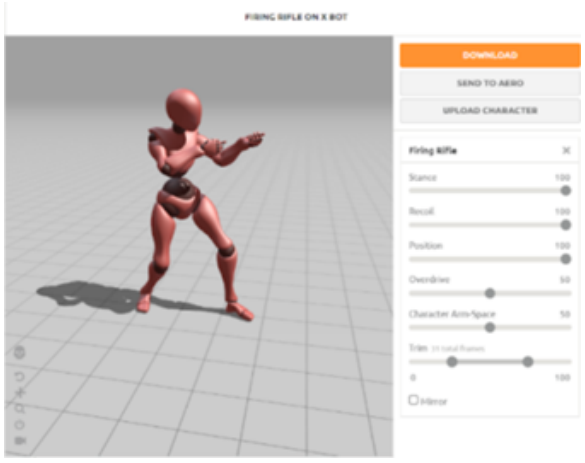

Figura 7.64: Animación de disparo descargada de MIXAMO

Sin embargo, el modelo de Gaus maneja un arma diferente a la de un rifle como se muestra en la Figura 26. Su arma es una mochila que cuenta con un expulsor en forma de tubo en la parte inferior o baja del lado izquierdo por lo cual todos los *keyframes* con respecto a ambos brazos eran inservibles. Para este caso específico se eliminaron toda la animación que correspondía a los brazos para que de manera propia se pudieran animar con el arma específica de Gaus, ver Figura 7.63 y 7.64.

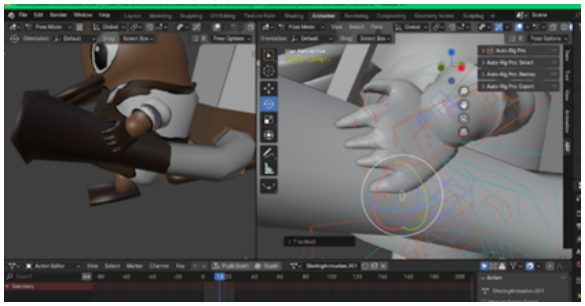

Figura 7.65: Animación de disparo mano izquierda

Figura 7.66: Animación de disparo mano derecha

Además, se abordó el uso del *Action Editor* para la eliminación y agregación de *keyframes* , un paso esencial para personalizar las animaciones base importadas desde Mixamo. Este editor permitió una manipulación precisa de los *keyframes*, facilitando la eliminación de aquellos que no se alineaban con la visión deseada para la animación. Además, se agregaron nuevos *keyframes* para introducir ajustes específicos que enriquecieran la acción y la expresividad del modelo, ver Figura 7.65.

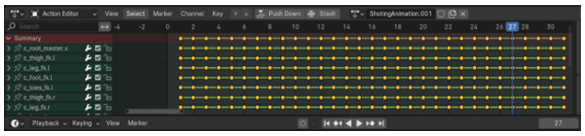

Figura 7.67: Action Editor para colocación de keyframes

Para definir claramente el inicio y el final de la animación, se utilizó la herramienta Manual Frame Range, que permitió establecer el rango de fotogramas en el que se ejecutaría la animación. Este enfoque no solo mejoró la fluidez y la coherencia de los movimientos, sino que también garantizó que las animaciones personalizadas se integraran de manera efectiva en la narrativa y mecánicas del videojuego, ofreciendo así una experiencia más envolvente para el jugador, ver Figura 7.66.

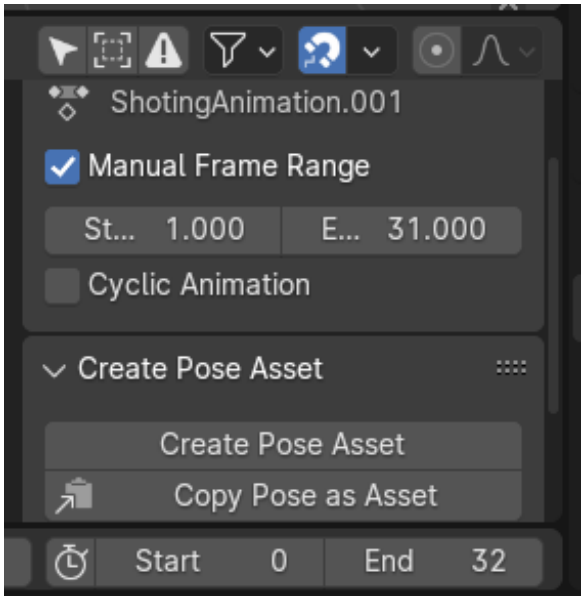

Figura 7.68: Establecimiento de rango en el que se ejecuta la animación

Pruebas

Para finalizar la etapa del uso de *Blender*, se realizaron pruebas de las animaciones ajustadas. Este paso es crucial porque ayudó a verificar que todas las acciones y movimientos se ejecuten de manera adecuada y natural dentro del contexto del videojuego. Durante las pruebas, se prestó especial atención a elementos como el movimiento de la cola, que debía reflejar la agilidad y el carácter del modelo, así como la correcta articulación de los brazos y otras extremidades.

Cada animación se revisó minuciosamente para asegurar que todos los aspectos, desde la sincronización hasta la fluidez de los movimientos, estuvieran alineados con las expectativas de diseño. Este proceso no solo permitió identificar y corregir posibles errores en la animación, sino que también garantizó que el resultado final contribuyera a una experiencia de juego inmersiva y visualmente atractiva para el jugador, ver Figura 7.67. Lo cual nuevamente influye en darle una caracterización al personaje por medio de los movimientos realizados.

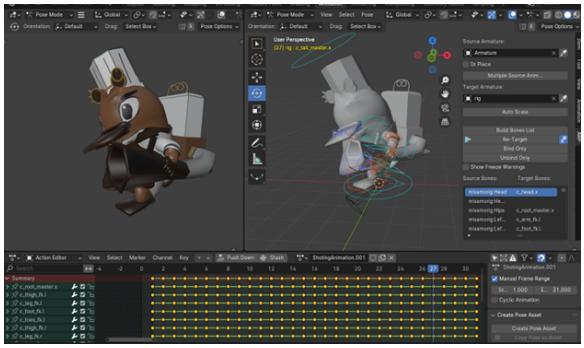

Figura 7.69: Resultado de animación en *Blender*

Integración a Unity Se abordó el proceso de integración de las animaciones finalizadas en *Blender* al motor de desarrollo Unity. Esta etapa es esencial para garantizar que las animaciones se implemen-

ten correctamente dentro del entorno del videojuego, permitiendo su uso en diversas interacciones y mecánicas del juego. La integración requiere la exportación de las animaciones en un formato compatible, asegurando que todas las características y ajustes realizados en *Blender* se mantengan intactos durante el proceso de importación.

Exportación archivo

Las animaciones se exportaron en formato FBX, el cual es ampliamente reconocido y soportado por Unity. Este formato permite incluir tanto la geometría del modelo como las animaciones, asegurando que todos los ajustes realizados se conserven en la transición. Además, que también se configura para que mantuviera la textura.

Al exportar, se prestó atención a los parámetros de configuración, como la escala y la orientación, para evitar problemas durante la importación en Unity. Este proceso no solo facilitó la integración de las animaciones en el motor, sino que también sentó las bases para su implementación efectiva en la dinámica del videojuego.

Creación de *Game Object*

Se procedió a la creación de un *Game Object* en Unity, que es fundamental para organizar y gestionar los modelos y sus respectivas animaciones dentro del proyecto. Al abrir el proyecto en Unity, se inició la creación de un nuevo *Game Object* que serviría como contenedor para el modelo animado.

Este *Game Object* permitió agrupar todas las componentes necesarias, facilitando la manipulación y control del modelo durante la ejecución del juego. Al estructurar el *Game Object* de esta manera, se establece un marco organizado en el que se integrarán las animaciones y se gestionarán las interacciones del modelo dentro del entorno del videojuego.

Componente *animator*

Se abordó la incorporación del componente *Animator* al *Game Object*previamente creado. Este componente ayudó para la gestión de las animaciones del modelo, ya que permitió la interacción con el *Animator Controller*, que alberga el árbol de estados de animación. Al agregar el componente *Animator*, se configuró para utilizar el controlador específico que se había creado previamente, lo que permite que todas las animaciones importadas sean accesibles y controlables dentro de *Unity* , ver Figura 7.68.

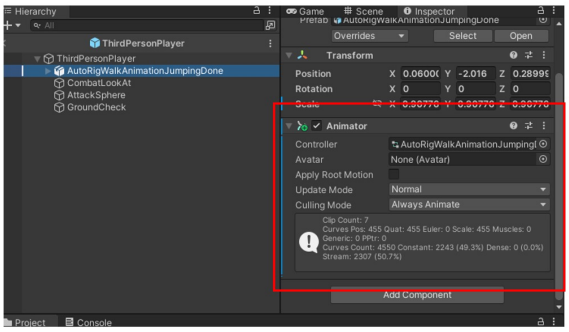

Figura 7.70: Pantalla del componente *Animator* con el **Game Object** como **controller**.

Esta integración asegura que las transiciones entre animaciones se realicen de manera fluida y que las interacciones del modelo con el entorno del juego respondan correctamente a las acciones del jugador.

Árbol de estados de animación

Se exploró el diseño del árbol de estados de animación, una herramienta clave para gestionar las diferentes animaciones que el modelo puede ejecutar. El árbol de estados ayudó a definir y organizar las transiciones entre las animaciones, lo que es vital para la coherencia del comportamiento del personaje dentro del juego.

Creación del árbol de estados

Continuamente se realizó la creación del árbol de estados de animación en *Unity*. Utilizando la interfaz de *Animator*, se comenzaron a añadir los diferentes estados que representan cada una de las animaciones del modelo, ver Figura 7.69.

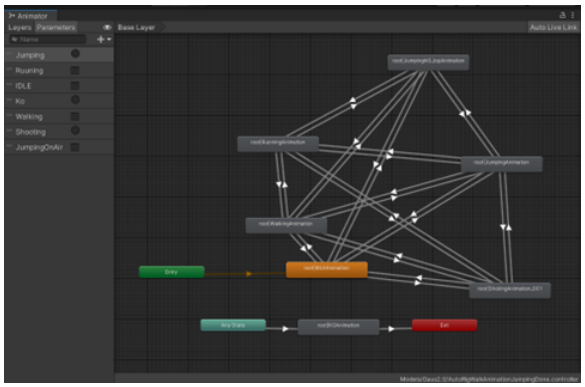

Figura 7.71: Resultados del árbol de estados de animación

Este árbol se construyó de manera que cada estado estuviese conectado a transiciones que facilitan el cambio de una animación a otra, dependiendo de las condiciones predefinidas, ver Figura 7.70.

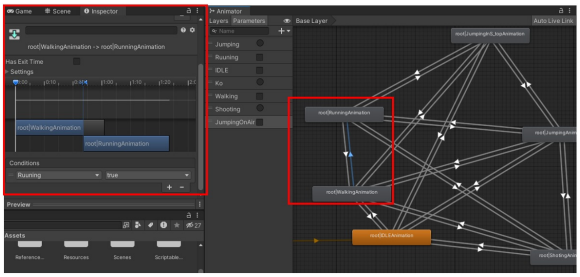

Figura 7.72: Configuración de flecha de transición

La correcta estructuración de este árbol es crucial para asegurar que las animaciones se ejecuten en el momento adecuado, mejorando la fluidez del movimiento y la respuesta del personaje a las acciones del jugador.

Definición de parámetros

Se colocaron parámetros que controlan las transiciones dentro del árbol de estados de animación. Se implementaron booleanos y *triggers* para gestionar las condiciones que activan las diferentes animaciones, ver Figura 7.71.

Figura 7.73: Parámetros colocados

Por ejemplo, un parámetro booleano puede utilizarse para determinar si el personaje está en movimiento o en reposo, mientras que un *trigger* podría activarse al ejecutar una acción específica, como un salto o un ataque. Esta configuración nos permitió una gestión dinámica y precisa de las animaciones, asegurando que el personaje responda adecuadamente a las acciones del jugador y mantenga una coherencia en su comportamiento, ver Figura 7.72.

Figura 7.74: Índice para identificar *bools* y *triggers*

Uso de *scripts*

Se centró en el uso de *scripts* para controlar las animaciones y la lógica del juego, permitiendo una interacción más rica y compleja entre el jugador y el entorno.

Creación de *scripts* principal Se llevó a cabo la generación de varios *scripts* principales para cada personaje, que se designó con un nombre representativo, como "GausAnimationz así por cada nombre

de los correspondientes modelos. Estos *scripts* son el núcleo que gestiona todas las interacciones del personaje, incluyendo las llamadas a las diferentes funciones de animación definidas anteriormente. A través de estos *scripts*, se pueden controlar las animaciones en respuesta a las entradas del jugador, permitiendo que el personaje se mueva, ataque o realice otras acciones de acuerdo con la lógica del juego. Esta estructura de *scripting* es fundamental para garantizar que las animaciones se integren de manera efectiva con la jugabilidad, ver Figura 7.73.

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class GausAnimations : MonoBehaviour
{
    [SerializeField] private Animator _animator;

    public void WalkingAnimation(bool walkingAnimation)
    {
        _animator.SetBool("Walking", walkingAnimation);
    }

    public void RuningAnimation(bool runningAnimation)
    {
        _animator.SetBool("Ruuning", runningAnimation);
    }

    public void Jumping()
    {
        _animator.SetTrigger("Jumping");
    }

    public void JumpingInStop(bool jumpingOnAirAnimation)
    {
        _animator.SetBool("JumpingOnAir", jumpingOnAirAnimation);
    }

    public void ShootAnimation()
    {
        _animator.SetTrigger("Shooting");
    }

    public void KO()
    {
        _animator.SetTrigger("Ko");
    }
}
```

Figura 7.75: Script principal – Ejemplo: GausAnimations

Llamada de métodos

Se procedió a la realización de las llamadas a los métodos dentro del *script* principal. Cada función de animación creada (Ejemplo: en "GausAnimation") se invoca en momentos específicos del juego, como cuando el jugador presiona un botón o al alcanzar ciertos eventos dentro de la lógica del juego, ver Figura 7.74.

```
private void HandleInputs()
{
    //Jump -space
    if(Input.GetButtonDown("Jump") && isGrounded)
    {
        velocity.y = Mathf.Sqrt(jumpHeight * -2f * gravity);
        GameManager.Instance.PlayerDataUpdate(PlayerDataEnum.jumps);
        AudioManager.Instance.PlayOneShot(FMODEvents.instance.gaussJump, transform.position);
        _animations.Jumping();
    }
    velocity.y += gravity * Time.deltaTime;
```

Figura 7.76: *Script* controlador del personaje Gaus – Ejemplo: Animación de salto

Esta invocación de métodos no solo permite que las animaciones se ejecuten en el momento adecuado, sino que también asegura que el modelo reaccione de manera fluida y coherente a las acciones del jugador, mejorando así la experiencia de juego.

Evaluación de calidad y retroalimentación

Por último, se abordó la evaluación de calidad y retroalimentación, un paso crucial en el proceso de desarrollo para garantizar que las animaciones y mecánicas del juego cumplan con los estándares deseados. Durante esta fase, se realizaron pruebas exhaustivas de las animaciones en el contexto del juego, observando su rendimiento y la respuesta del jugador. La retroalimentación recogida de las pruebas ayuda a identificar áreas de mejora y ajustes necesarios, asegurando que el producto final sea de alta calidad y ofrezca una experiencia de usuario envolvente. Se decidió hacer un *"PlayTesting"* en la Universidad del Valle de Guatemala, en donde a cada participante se le brindó un formulario para que llenara antes y después de haber jugado el videojuego y que evaluara las diversas áreas de este; entre ellos el de las animaciones. Este proceso de evaluación y ajuste es fundamental para optimizar la jugabilidad y el disfrute del jugador, y para garantizar que las animaciones se integren sin problemas en la narrativa y mecánicas del videojuego.

7.10. Diseño de niveles y mecánicas

Con la historia establecida, esta se separó en niveles, cada uno determinando cuál es el alcance de la historia. Al final se definieron 5 niveles. Cada nivel con una estética y enemigos distintos, que fueran progresando la historia. Para ello, se hizo uso del documento *Game Design Document* el cual nos permitió por niveles determinar sus objetivos, modo de progreso, *power-ups*, enemigos, mecánicas, habilidades, música, objetos, ambientación, *layouts*, y otros detalles importantes del nivel.

7.10.1. Nivel 1

El primer nivel se estableció como un nivel tutorial que sirviera para que el jugador se acoplara al mundo del juego y sus mecánicas, por lo que se optó a que este fuese sencillo y rápido de vencer. Por ello, su estructura y mapa observado en la Figura 7.77 se nota que es muy directa y corta de atravesar. La estética y temática del nivel será de pirata y madera, esto debido a que está directamente ligada a la temática utilizada para el diseño de los enemigos del nivel.

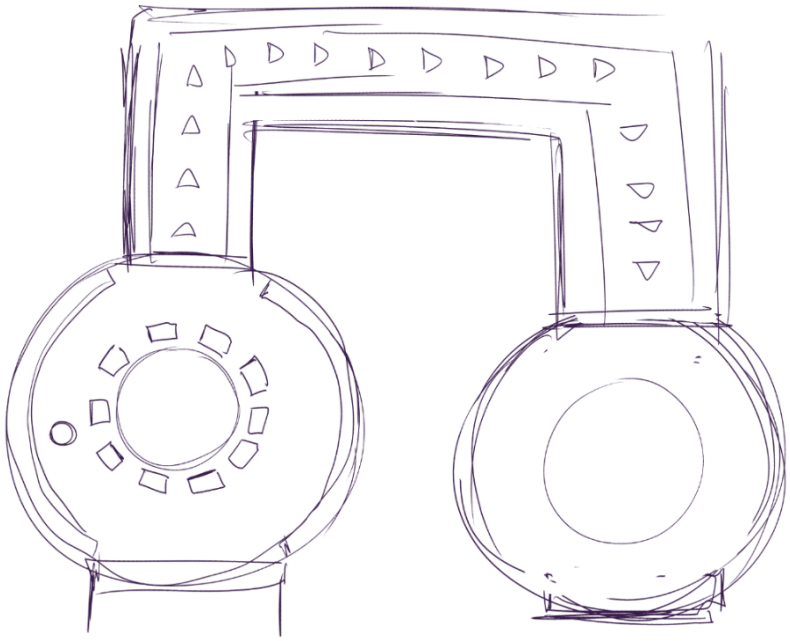

Figura 7.77: Diseño del *layout* del primer nivel

El nivel 1 de diseñó para estar dividido dentro de 3 secciones principales. La primera sección es un cuarto redondo en donde se iniciará el nivel. En este cuarto no habrá mucho peligro para poder explicarle los controles básicos al jugador. La siguiente sección será un pasillo que el jugador tendrá que derrotar grupos de dingos. Al final del pasillo habrá otro cuarto en el cual ocurrirá la batalla contra el jefe del nivel, Jagger. Esta batalla estará dividida en 2 fases. La primera, una lucha directa contra el jefe, y la segunda fase, usando elementos del nivel como un timón pirata 7.78 y la munición de tipo *spaghetti* para derrotarlo.

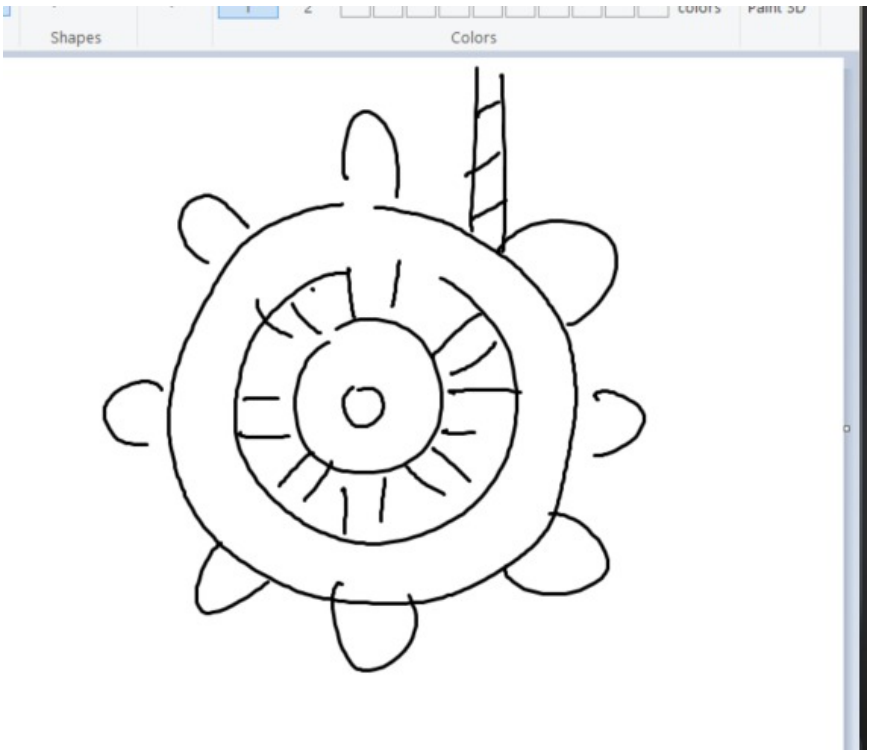

Figura 7.78: Ideación de elementos dentro del nivel 1

7.10.2. Nivel 2

Basado en la estética de los enemigos de tipo búho, se prosiguió a utilizar como inspiración dos elementos en los cuales basarse para el diseño del nivel; castillos y casas del árbol. Por ello se creó un *mood board* con distintas imágenes de castillos ficticios y reales, y de casas en el árbol. 7.79

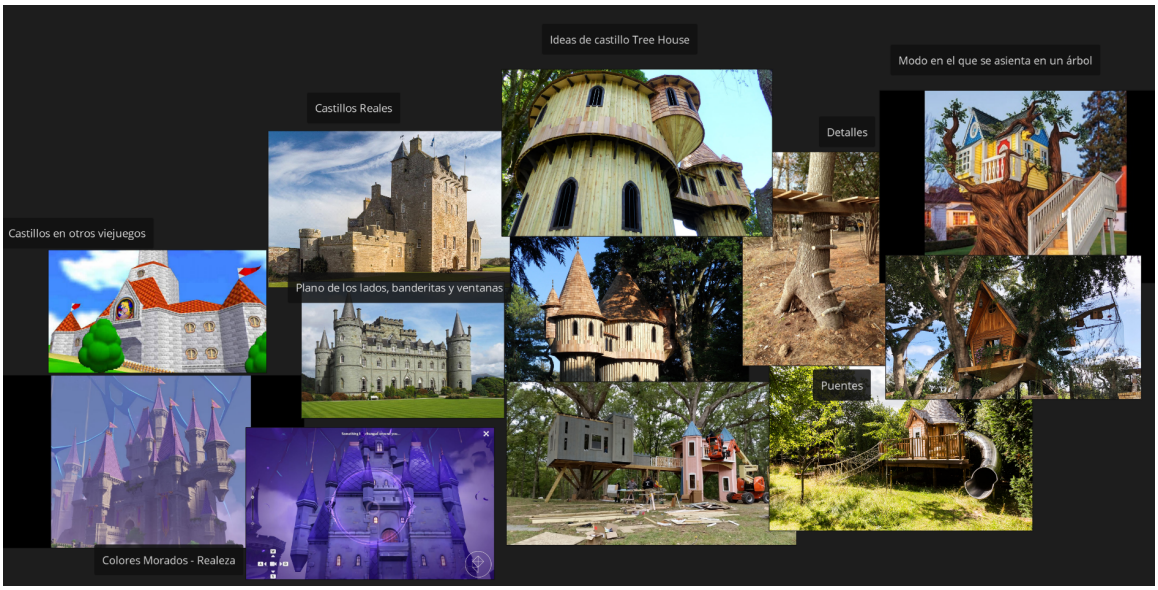

Figura 7.79: *Moodboard* de inspiración del segundo nivel

Con el *mood board* creado, se realizaron varios boquejos de los pisos y cuartos que llevaría el nivel 2. Gracias a que el nivel está basado en un árbol, se consideró tener un nivel vertical en el cual se fuese subiendo de cuarto en cuarto, avanzando y luchando con los enemigos de tipo búho. Además, de esa manera se podrá utilizar el potencial de la munición introducida en este nivel, la gelatina, la cual permite al jugador rebotar sobre en ella y llegar a lugares más altos. El la Figura 7.80 se puede observar el diseño final de todos los cuartos correspondientes al nivel 2 y cómo se atraviesan para pasar de uno a otro.

Figura 7.80: Diseño del *layout* del segundo nivel

7.11. Colores

Para la definición de la paleta de colores se tuvo en cuenta varios aspectos relevantes. Con el objetivo de lograr una inmersión total en la historia del juego, se tomó la decisión de utilizar una paleta de colores que reflejara la personalidad y esencia del personaje principal, Gaus. Rápidamente se estableció la paleta de colores mostrada en la Figura 7.81. Estos colores seleccionados no solo se relacionan con Gaus, sino que también se buscó tonalidades cafés y cálidas con el fin de representar confort y *groundedness*, especialmente porque los ornitorrincos viven bajo tierra.

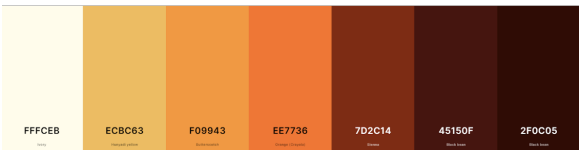

Figura 7.81: Paleta de colores principales seleccionados para los elementos de la interfaz gráfica

Figura 7.82: Combinaciones de los colores seleccionados para los elementos de la interfaz gráfica

A partir de la paleta de colores seleccionada se realizaron pruebas de contraste para determinar qué combinación de colores deberían ser utilizadas y cuáles no. Y cómo se verían las combinaciones de los mismos. Para cada combinación se realizó una prueba de contraste. Esto con el objetivo de tener una interfaz de usuario que pueda ser considerada como limpia, y que pueda ser interpretada y leída por la totalidad de los posibles jugadores.

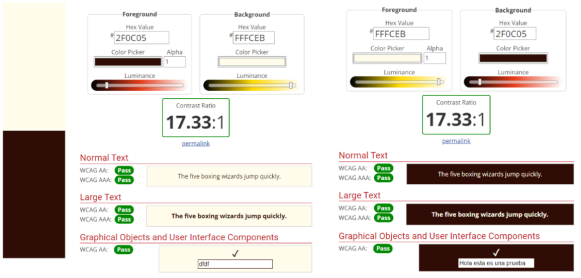

Figura 7.83: Revisión de contraste de dos colores de la paleta, test pasado

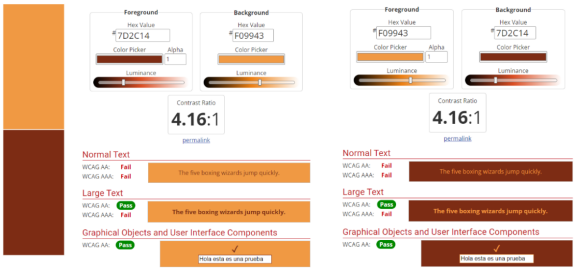

Figura 7.84: Revisión de contraste de dos colores de la paleta, test fallido

7.12. Diseño e implementación de la interfaz

La implementación de la interfaz gráfica fue de suma importancia y requirió de varias versiones y pasos.

7.12.1. Prototipos iniciales

A partir de los requerimientos básicos y juegos similares, se determinó que las pantallas necesarias eran la pantalla principal, pantallas de configuración, inicio de sesión, inicio de partida, pausa, *game over* y finalización de nivel. A partir de ello se creó un prototipo minimalista de todas estas pantallas.

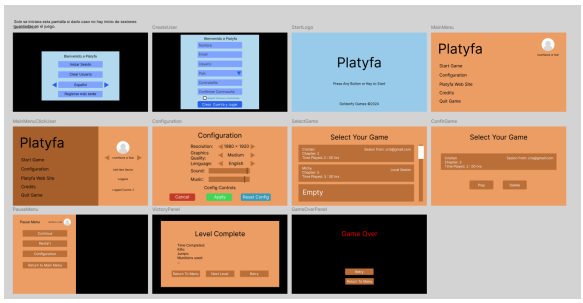

Figura 7.85: Primer diseño de la interfaz. Definición de pantallas necesarias

7.12.2. Diseño final

El segundo prototipo tuvo el objetivo de ser más similar a la idea final a implementar en el juego buscando ser más creativo que el primer prototipo. Con él se pudo determinar los elementos necesarios a ilustrar y crear, y se pudo definir las rutas de los botones e interacciones para los jugadores. Para este paso fue de suma importancia la paleta de colores anteriormente establecida para utilizar esos colores en todos los elementos que componen la interfaz. Además, se realizaron varios cambios en varias pantallas para que estas fuesen más creativas y acordes a la idea central del juego.

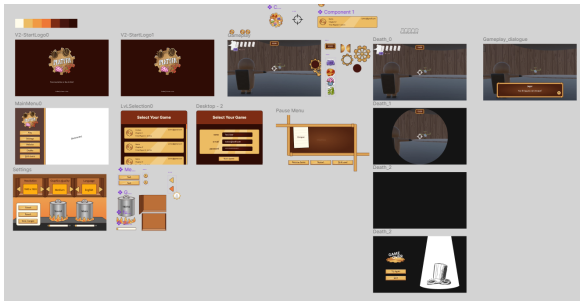

Figura 7.86: Nueva interfaz, similar al producto final

7.12.3. Implementación

Para la implementación de la interfaz gráfica dentro del proyecto de Unity se utilizaron diversas herramientas. Para la creación de *assets* se hizo uso de un programa de dibujo llamado *Clip Studio*.

En él se ilustraron todos los elementos necesarios y se guardaron como archivos png. Para la manipulación y animación de elementos se hizo uso de un paquete descargable de la tienda de *assets* de Unity llamado *LeanTween*. Gracias a este paquete, las pantallas pueden verse más dinámicas y llamativas para los jugadores.

7.13. Fase de desarrollo: Primer prototipo (*Backend*)

El desarrollo del primer prototipo se enfocó inicialmente en el *backend*, dado que constituye uno de los componentes más críticos del proyecto. Al establecer la lógica del servidor y la base de datos en esta etapa temprana, se sientan las bases para la recolección de datos y el análisis del rendimiento del sistema. Este enfoque es esencial para obtener información preliminar sobre el comportamiento de los usuarios y la respuesta del sistema, que será crucial para la evolución posterior del videojuego *Platyfa*.

Para el desarrollo del backend, se decidió utilizar NodeJS con el *framework* Express, una combinación que permite construir un servidor robusto y eficiente, ideal para manejar la lógica de la API del videojuego. Asimismo, se optó por MySQL como sistema de gestión de bases de datos, debido a su capacidad para gestionar datos estructurados y proporcionar consultas rápidas, lo que permite almacenar y recuperar los datos de los jugadores de manera eficiente.

7.13.1. Razones para desarrollar primero el *backend*

El desarrollo del *backend* se priorizó debido a su rol fundamental en el proyecto. Al ser una de las partes más críticas de la infraestructura del videojuego, el *backend* debe gestionar eficientemente las operaciones relacionadas con los usuarios, como el registro e inicio de sesión, además de proporcionar un entorno estable para la gestión de datos.

El enfoque inicial en el *backend* también permite capturar datos desde los primeros usuarios que testean el juego, lo que facilita el análisis del comportamiento tanto de la API como del rendimiento del servidor y la base de datos. Esta información temprana es crucial para realizar ajustes y mejoras antes de avanzar en el desarrollo del *frontend* o en otras características del videojuego.

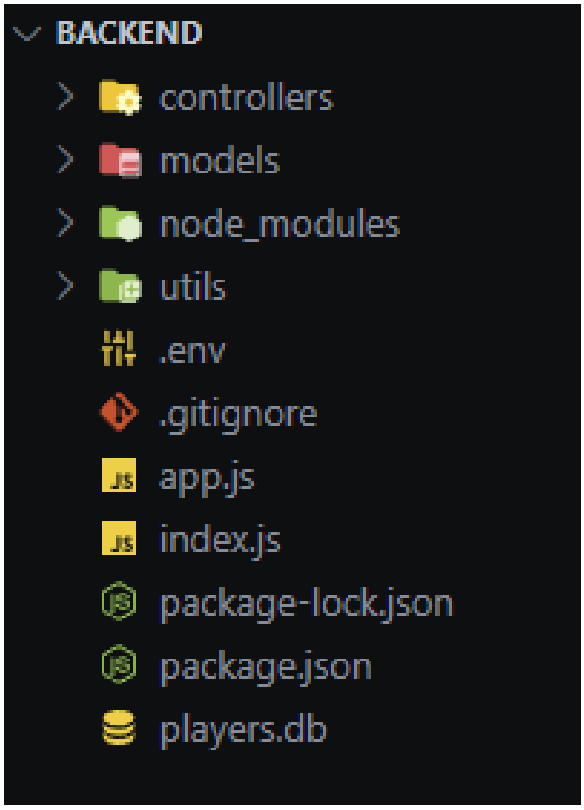

Figura 7.87: Estructura de carpetas para el *backend*

7.13.2. Estructura del *backend*

La estructura de carpetas del primer prototipo de backend fue diseñada con el fin de mantener una organización clara y eficiente de los archivos. A continuación se detalla cómo fue implementada esta estructura:

- **package.json**: Archivo principal que contiene la información del proyecto y las dependencias necesarias.
- **index.js**: Archivo de entrada para crear la conexión inicial al servidor.
- **app.js**: Archivo que configura el servidor web utilizando NodeJS y Express.
- **models**/: Directorio destinado a la creación de los modelos y esquemas de la base de datos MySQL, permitiendo la interacción con los datos de los usuarios.
- **controllers**/: Directorio donde se desarrollan las operaciones sobre los datos, gestionando las acciones de los *endpoints*.
- **utils**/: Directorio donde se pueden encontrar utilidades como el manejo de la estructura para enviar un *mail*, como también, *loggers* predefinidos para evitar el uso de *console logs*, como también, para tener un archivo con la configuración del puerto, llaves secretas, saltos, etc... Que servirán para la conexión y autenticación de peticiones y usuarios en la API.

Esta estructura no solo garantiza un flujo de trabajo organizado, sino que también facilita la expansión futura del sistema a medida que se añaden más funcionalidades al videojuego.

7.13.3. Desarrollo de *endpoints*

El objetivo inicial del desarrollo del *backend* fue la creación de dos *endpoints* clave, destinados a gestionar el *login* y el registro de los usuarios. Estos *endpoints* mejoran la lógica de autenticación, permitiendo una experiencia más fluida para los jugadores desde el primer contacto con el juego.

1. **loginUser** (POST)
   - Parámetros de entrada: `username` o `email`, `password`
   - Detalles: Este *endpoint* permite que los usuarios inicien sesión en el sistema. En caso de que los datos proporcionados sean incorrectos, se devolverá el mensaje "*Invalid User Data*".
2. **createUser** (POST)
   - Parámetros de entrada:
     - `username`: String
     - `name`: String
     - `email`: String
     - `login date`: Date
     - `password`: String
     - `country`: String
     - `language`: String
   - Detalles: Este *endpoint* gestiona el registro de nuevos usuarios en la base de datos. Los datos proporcionados por el usuario, como su nombre, país, y preferencia de idioma, se almacenan para crear una experiencia personalizada.

Tras haber definido las tablas de la base de datos a las cuales apuntan los *endpoints*, el siguiente paso en el desarrollo del primer prototipo fue la programación de la lógica de cada uno de estos *endpoints*.

7.13.4. Proceso de desarrollo y validación

El desarrollo de los *endpoints* estuvo centrado en dos funcionalidades principales: el registro de usuarios y la autenticación (*login*). Para garantizar la seguridad y validez de los datos ingresados por los usuarios, se implementaron varias medidas de control en el *backend*.

1. **Validación de datos en el *endpoint* de registro (*Register*)**

   Al momento de diseñar el *endpoint* de registro, se estableció la necesidad de validar los datos recibidos, asegurando que cada campo contenga valores adecuados y dentro de los parámetros permitidos (por ejemplo, que los nombres tengan un largo mínimo, que los correos sean válidos, etc.). Para este propósito, se implementó el *middleware express-validator*, que permite definir reglas de validación para cada uno de los parámetros ingresados por el usuario.

   - ***Middleware* utilizado**: *express-validator*
   - **Objetivo**: Validar que los campos ingresados cumplan con los requisitos establecidos (tamaño, tipo de dato, etc.) antes de ser procesados por el servidor y almacenados en la base de datos.

2. **Protección de contraseñas**

   Como medida de seguridad adicional, se implementó el *hashing* de las contraseñas antes de almacenarlas en la base de datos. Para ello, se utilizó el *middleware bcrypt*, el cual realiza un proceso de *hash* con un número determinado de saltos (10 en este caso), asegurando que las contraseñas almacenadas no sean fácilmente vulnerables en caso de un ataque.

   - ***Middleware* utilizado**: *bcrypt*
   - **Objetivo**: *Hashear* las contraseñas de los usuarios con una cantidad de 10 saltos, incrementando la seguridad al almacenar dicha información en la base de datos.

3. **Gestión de variables sensibles con .env**

   Las variables sensibles, tales como las configuraciones del servidor, claves de seguridad y las credenciales para la base de datos, se almacenaron en un archivo **.env** para evitar que dicha información sea expuesta en el código fuente. Esto se logró mediante el uso del *middleware* **dotenv**, que permite acceder a estas variables desde el archivo de entorno sin necesidad de colocarlas directamente en el código, lo cual mejora la seguridad general del proyecto.

   - ***Middleware* utilizado**: *dotenv*
   - **Objetivo**: Proteger las claves y configuraciones sensibles mediante el uso de archivos de entorno, evitando su exposición en el código público.

7.13.5. Generación y uso de tokens

Tras validar y *hashear* los datos del usuario en el registro, se implementó un sistema de generación de tokens para identificar de manera única a los usuarios en sus sesiones de juego. Para ello, se utilizó el *middleware jsonwebtoken*, el cual toma el `id` del usuario registrado y genera un token con una clave secreta. Este token será utilizado para validar futuras solicitudes del usuario a la API.

- ***Middleware* utilizado**: *jsonwebtoken*
- **Objetivo**: Generar un token único para cada usuario registrado, el cual se utilizará para identificar al jugador en sus sesiones de juego.

7.13.6. Testeo con Postman

Una vez implementados los *endpoints*, se realizaron pruebas mediante Postman para verificar el correcto funcionamiento de la inserción de datos en la base de datos. Estas pruebas se realizaron tanto para el *endpoint* de *register* como para el de *login*.

- ***Endpoint Register* (POST)**: Se validó la creación de nuevos usuarios en la base de datos.

```
// createUser endpoint
playersRouter.post(
  "/createUser",
  [
    // validation rules
    check("username").isLength({ min: 3 }),
    check("name").isLength({ min: 3 }),
    check("email").isEmail(),
    check("login_date").isISO8601(),
    check("password").isLength({ min: 5 }),
    check("role").isIn(["Player", "Admin"]),
    check("kills").custom((value) => {
      if (Number.isInteger(value)) {
        return true;
      } else {
        throw new Error("kills must be an integer");
      }
    }),
    check("ko").custom((value) => {
      if (Number.isInteger(value)) {
        return true;
      } else {
        throw new Error("ko must be an integer");
      }
    }),
    check("jumps").custom((value) => {
      if (Number.isInteger(value)) {
        return true;
      } else {
        throw new Error("jumps must be an integer");
      }
    }),
    check("frequency_main_arm").custom((value) => {
      if (Number.isInteger(value)) {
        return true;
      } else {
        throw new Error("frequency_main_arm must be an integer");
      }
    }),
    check("frequency_second_arm").custom((value) => {
      if (Number.isInteger(value)) {
        return true;
      } else {
```

Figura 7.88: Lógica para el primer *endpoint* de *register*

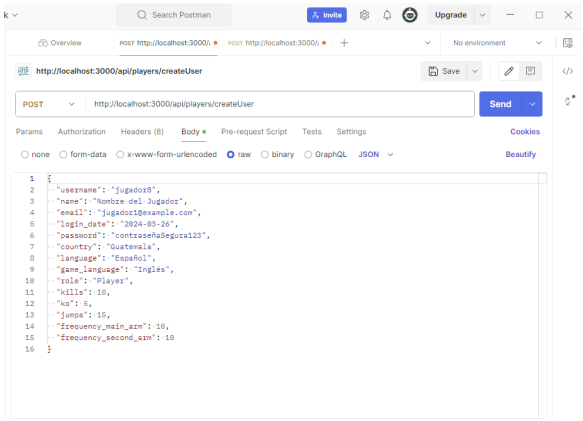

Figura 7.89: Prueba de inserción de nuevos jugadores a la base de datos

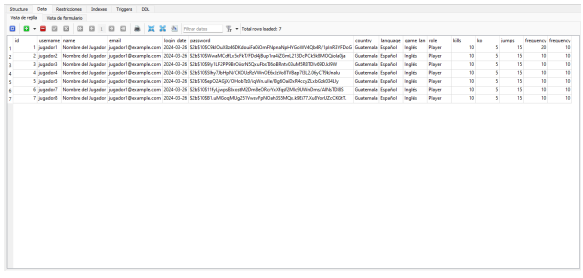

Figura 7.90: Base de datos con jugadores nuevos de prueba

- ***Endpoint Login* (POST)**: Se validó el proceso de autenticación comparando la contraseña ingresada con la almacenada, y la generación del token correspondiente.

Se realizaron inserciones exitosas en la base de datos, como se puede ver en la prueba de inserción de nuevos jugadores en la tabla *players*, almacenando los datos correctamente.

Figura 7.91: Lógica para el primer *endpoint* de *login*

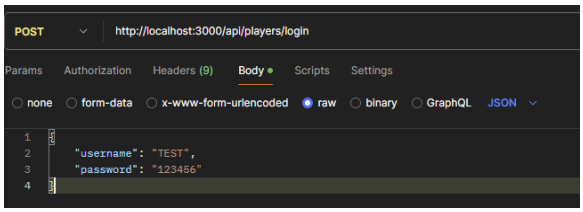

Figura 7.92: Test de funcionamiento para *endpoint* de *login*

Creación del *endpoint* para registrar sesiones de juego El siguiente paso fue el desarrollo de un tercer *endpoint* destinado a registrar las sesiones de juego de los jugadores. Este *endpoint* debía capturar información clave sobre el desempeño de los jugadores durante sus partidas, como el nivel jugado, la duración del nivel, y el resultado del juego (victoria, derrota, cancelado, o abandono).

Parámetros del *endpoint createGameSession* (POST):

- `id player`
- `date`
- `level`
- `duration level`
- `game result`
- Verificaciones adicionales:
  - Verificar que el jugador exista.
  - Asegurar que el nivel sea válido.
  - Permitir que el resultado solo sea (*win*, *lose*, *canceled*, *quitted*).
  - Validar que la fecha no sea mayor a la fecha actual.

Se diseñó una nueva tabla en la base de datos, *game_sessions*, para almacenar la información generada por este *endpoint*, y se implementó una relación de llaves foráneas con la tabla *players*, garantizando que solo se almacenen datos de jugadores existentes.

7.13.7. Cambio de enfoque en el desarrollo del *backend*

Durante el análisis del flujo de trabajo y tras varias pruebas, se identificaron problemas relacionados con la eficiencia y el rendimiento. Se determinó que tener una sola API registrando cada evento del juego podía causar una sobrecarga de peticiones, afectando tanto el servidor como el juego. Además, se detectó que la base de datos SQLite no era adecuada para manejar grandes volúmenes de datos en producción, por lo que se decidió buscar una alternativa más robusta.

Tras una reunión con el asesor del proyecto, se acordó cambiar el flujo de trabajo, separando las bases de datos y reorganizando el flujo de las peticiones. El nuevo enfoque contempla que el juego tenga su propia base de datos local utilizando SQLite para registrar eventos temporales, y que posteriormente se sincronice con la base de datos principal, alojada en PostgreSQL, mediante un nuevo *endpoint* llamado Sync_data el cual se encuentra dentro de la lógica del juego elaborado por el equipo de desarrollo del mismo. Esto evita la saturación del servidor al manejar todas las solicitudes en tiempo real.

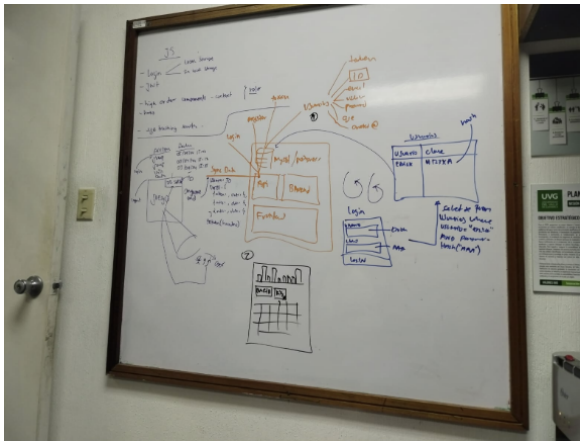

Figura 7.93: Nueva definición del enfoque a tomar en la metodología del desarrollo

7.13.8. Implementación y pruebas en producción

Para probar la API en un entorno de producción, se utilizó el servicio de *hosting* Vercel, subiendo el *backend* a producción. Sin embargo, se enfrentaron problemas iniciales relacionados con la comunicación entre el *backend* y la base de datos local, lo que llevó a configurar una base de datos en la nube para garantizar su correcto funcionamiento en línea.

Finalmente, los *endpoints* se probaron satisfactoriamente en producción, y se verificó el funcionamiento correcto del *login*, *registro* y *game_sessions* mediante Postman.

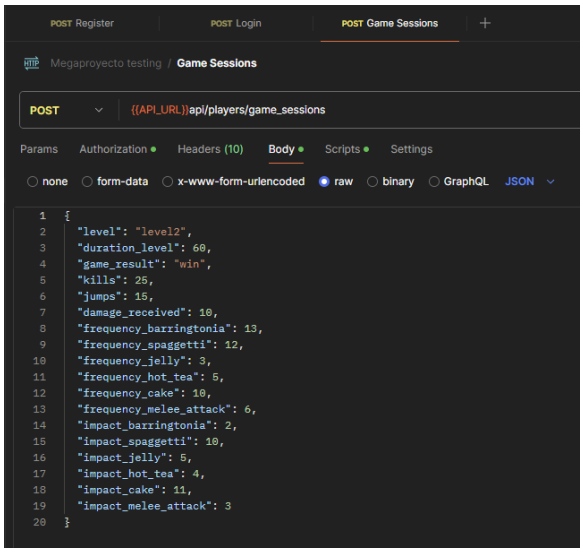

Figura 7.94: Test de implementación de la primera API en producción

7.13.9. Flujo de trabajo

El desarrollo del *backend* se dividió en dos fases principales:

1. **Fase 1: Diseño y creación de la base de datos**

   Se diseñaron las tablas necesarias para almacenar la información de los usuarios, estableciendo relaciones y estructuras de datos que aseguren una gestión eficiente de las operaciones de registro e inicio de sesión.

2. **Fase 2: Implementación de los *endpoints* y manejo de datos**

   Durante esta fase, se crearon los *endpoints* mencionados anteriormente para permitir la inserción y recuperación de datos de la base de datos. Se realizaron pruebas iniciales para evaluar el rendimiento del servidor y la eficiencia de la base de datos al manejar múltiples solicitudes simultáneas.

7.14. Fase de desarrollo: Prototipo de diseño (*Frontend*)

El desarrollo del prototipo del *frontend* de la página web comenzó con un análisis detallado de las webs estudiadas durante la fase de investigación. El objetivo de este análisis fue identificar tanto los puntos fuertes como las áreas de mejora en términos de *diseño* y *funcionalidad*, con el fin de aplicarlos en la creación de nuestra página web para el videojuego *Platyfa*.

El desarrollo de este prototipo seguirá un enfoque de "diseño centrado en el usuario"(UCD), con el objetivo de asegurar una navegación intuitiva y una integración clara de contenido, facilitando una experiencia de usuario sencilla y efectiva. Además, se incorporó una arquitectura de información modular, lo cual permite presentar el contenido de manera organizada, dinámica y adaptable a futuras actualizaciones.

7.14.1. Proceso de diseño

El primer paso para la realización del prototipo fue la creación de bocetos a mano. Este enfoque permitió esbozar ideas rápidamente sin la necesidad de utilizar aplicaciones o herramientas de diseño digital, lo que facilitó la conceptualización del posicionamiento de los elementos, acciones y la estructura general de las distintas secciones de la página web. Este proceso sirvió como una base inicial para definir cómo se organizarían los componentes visuales y funcionales en la web, asegurando que la jerarquía visual sea clara y fácil de entender para los usuarios.

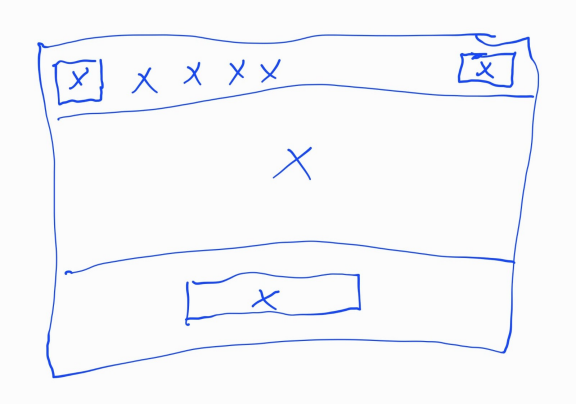

Figura 7.95: Boceto a mano para el prototipo *frontend*

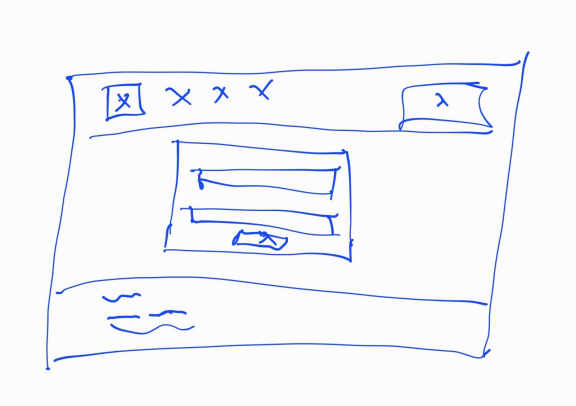

Figura 7.96: Boceto a mano para el prototipo *frontend*

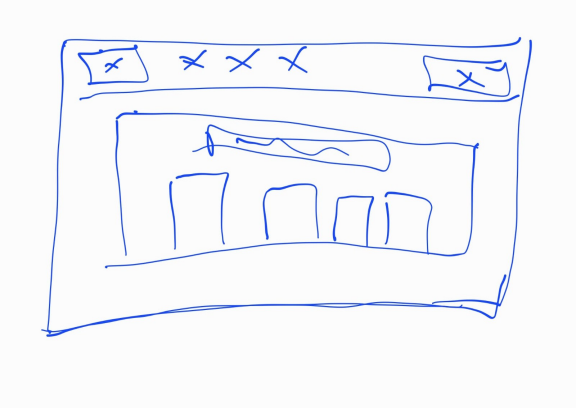

Figura 7.97: Boceto a mano para el prototipo *frontend*

Una vez establecido el esquema general, se identificó la necesidad de definir una paleta de colores coherente con la estética del videojuego. La selección de colores fue crucial para reflejar la atmósfera del juego y conectar la página web con los personajes, el entorno y la interfaz de usuario presentes en *Platyfa*. Se optó por utilizar la misma paleta de colores que los personajes y el entorno del juego, lo que ayuda a transmitir la identidad visual del proyecto.

Paleta de colores seleccionada: Predominan los tonos cálidos, como el amarillo, que evocan sensaciones de diversión, y colores oscuros que aportan un tono más serio y permiten una inmersión en la narrativa del juego.

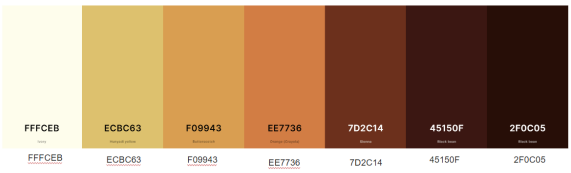

Figura 7.98: Paleta de colores elegida para el diseño *frontend*

7.14.2. Diseño digital con Figma

Con los bocetos y la paleta de colores definidos, se optó por utilizar la herramienta Figma para comenzar a diseñar de manera digital el prototipo de la página web. Figma fue elegida por su versatilidad en el diseño colaborativo y la facilidad para crear prototipos interactivos, permitiendo realizar ajustes de diseño de forma eficiente y visualizar cómo se comportarían los elementos en un entorno interactivo.

Se decidió que el diseño del *frontend* abarcaría tres secciones principales:

1. ***Home***
2. ***About***
3. ***News***

El objetivo principal de esta fase fue diseñar la estructura y visualización de la *landing page*, asegurando que refleje la identidad visual del juego mientras se mantiene una navegación clara y funcional.

7.14.3. Diseño de la página principal (Home)

La página *Home* fue diseñada para ofrecer una experiencia visual limpia y centrada en la promoción del juego, siguiendo un enfoque modular. Se priorizó el uso de imágenes representativas del juego y un acceso rápido que redireccionara a los usuarios a Steam para descargar *Platyfa*. La idea era proporcionar información clave de manera directa y atractiva, sin sobrecargar al usuario con detalles.

Elementos clave: Imagen destacada del juego, botón de descarga/redirección a Steam. Adicionalmente, se decidió que la barra de navegación sería visible en todas las páginas, facilitando el acceso a las distintas secciones del sitio en cualquier momento.

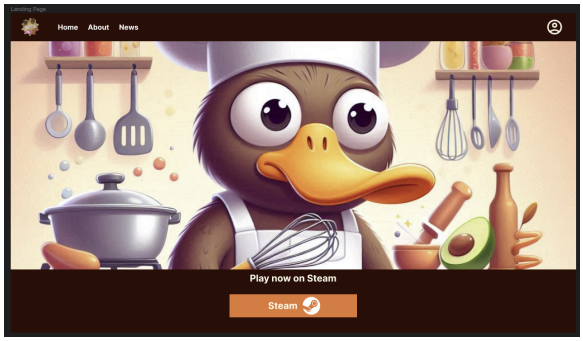

Figura 7.99: Prototipo de diseño *frontend* para la página *Home*

7.14.4. Diseño de la página (*About*)

La página *About* sigue el mismo esquema de colores y estilo general, pero con un enfoque en la presentación de información sobre la empresa desarrolladora del juego y su historia. Se utilizó un diseño basado en *cards*, lo que permite mostrar la información de manera estructurada y visualmente atractiva, manteniendo la coherencia con la experiencia del usuario.

Cada *card* presenta información sobre los inicios de la empresa, sus proyectos y su visión. Esta disposición permite a los usuarios conocer más sobre el equipo detrás del desarrollo de *Platyfa* de forma clara y ordenada.

Figura 7.100: Prototipo de diseño *frontend* para la página *About*

7.14.5. Diseño de la página de (*News*)

La página *News* fue diseñada para ofrecer a los usuarios actualizaciones y noticias sobre el videojuego, manteniendo el diseño basado en *cards*. Cada *card* incluye una imagen y un breve resumen de las novedades, como actualizaciones o nuevos contenidos del juego. Además, se añadió una pestaña en cada *card* que indica el nombre del administrador que realizó la publicación, junto con la fecha y hora de la actualización.

Este diseño no solo facilita la lectura de las noticias, sino que también ofrece transparencia sobre el origen de la información, lo cual es importante para mantener la confianza de los usuarios.

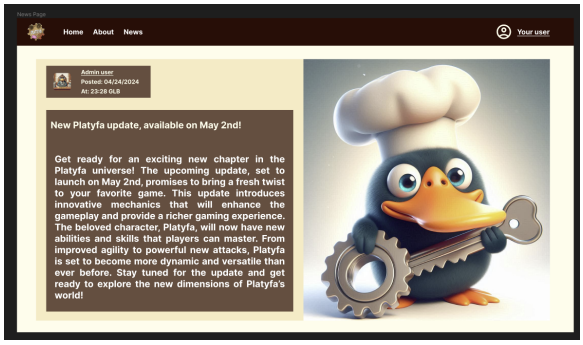

Figura 7.101: Prototipo de diseño *frontend* para la página *News*

7.14.6. Conclusión del prototipo de diseño

Con la finalización del prototipo de diseño frontend, que incluye las pantallas *Home*, *About* y *News*, se logró una estructura coherente y visualmente atractiva, alineada con la estética del videojuego *Platyfa*. El diseño de la página web no solo cumple con los requisitos funcionales, sino que también refuerza la identidad visual del juego y se basa en un enfoque de diseño centrado en el usuario, garantizando una navegación sencilla y una integración dinámica de contenido.

7.15. Fase de desarrollo: Segundo prototipo (*Backend*)

Después de realizar el *testing* del primer prototipo de la API para nuestro *backend* y de haber hecho cambios en su estructura, se identificaron varias deficiencias en su funcionamiento. Una de las principales fue la falta de una lógica adecuada para la creación y verificación de tokens. Además, la migración de una base de datos SQLite a PostgreSQL reveló la inexistencia de un modelo funcional compatible con esta nueva base. Otro problema detectado fue que cualquier usuario podía ingresar datos a través del *endpoint game sessions* sin restricciones. A esto se sumó la poca legibilidad del código debido al mal enfoque del proyecto y a las numerosas comparaciones con la librería *express validato*r. Como resultado, se decidieron implementar los siguientes cambios:

- Migración a una estructura modular utilizando el formato de *imports* de ES6.
- Creación y verificación de tokens para el inicio de sesión.
- Desarrollo de un modelo compatible con la base de datos PostgreSQL.
- Organización de componentes *middleware* en carpetas específicas.

Figura 7.102: Implementación del formato ES6

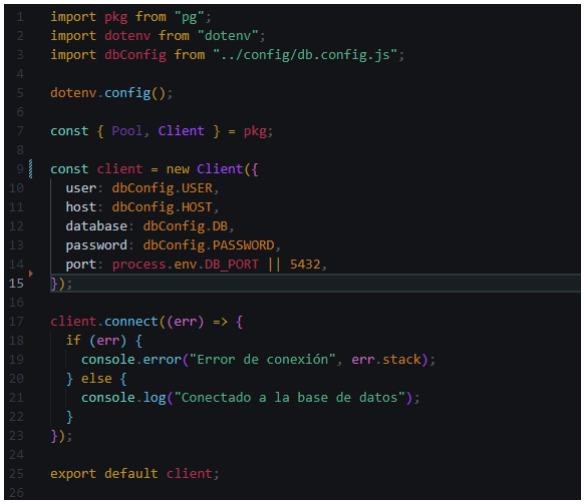

Figura 7.103: Implementación del modelo funcional con Postgresql

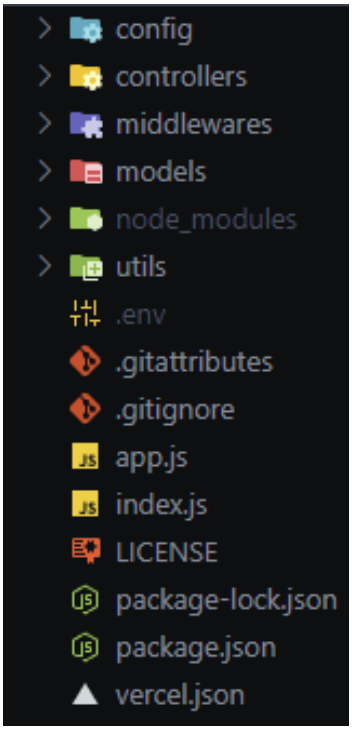

Figura 7.104: Organización de componentes *middleware* en carpetas específicas

Figura 7.105: Creación y verificación de tokens para el inicio de sesión

Para la migración de la base de datos de SQLite a PostgreSQL, se decidió mantener la misma estructura de tablas que se había utilizado en el primer prototipo de la API de nuestro *backend*. La tabla principal es *player*, la cual será responsable de almacenar toda la información y los roles de cada jugador en nuestro sitio en línea.

Tras realizar estos cambios, se procedió a desarrollar nuevos *endpoints* que permitieran operaciones CRUD para los usuarios en la plataforma. Los *endpoints* añadidos fueron:

- Eliminar cuenta (*Delete*)
- Actualizar perfil (*Update profile*)
- Restaurar cuenta (*Restore account*)

Para cada uno de estos *endpoints*, se utiliza un token generado al iniciar sesión. Este token contiene el ID del usuario, el nombre de usuario y una clave secreta. La nueva funcionalidad de verificación se basa en la validación de que el usuario esté correctamente autenticado. Una vez verificado, se extraen los campos necesarios del token, como el ID, para realizar las operaciones de actualización o restauración de la cuenta.

7.15.1. ¿Por qué se realizaron cada uno de estos cambios?

En primer lugar, la decisión de migrar a una nueva base de datos se tomó porque era necesario contar con una base capaz de manejar grandes volúmenes de datos sin comprometer el rendimiento de las consultas. Al evaluar diferentes opciones, se determinó que una base de datos no relacional no sería adecuada para este proyecto debido a su limitada capacidad para gestionar grandes cantidades de información de manera eficiente. Aunque las bases no relacionales suelen ser flexibles, presentan una mayor dificultad para realizar consultas complejas en grandes volúmenes de datos, lo que podría afectar el rendimiento general del sistema.

Inicialmente, se usó SQLite, ya que era adecuado para el desarrollo del primer prototipo al estar enfocado en un entorno local y con un volumen de datos limitado. Sin embargo, al proyectar el crecimiento del proyecto, se evidenció que SQLite no era una solución escalable, debido a sus limitaciones en el manejo de grandes cantidades de información y en la concurrencia de múltiples usuarios. Estas razones llevaron a la decisión de migrar a PostgreSQL, una base de datos relacional que ofrece mayor escalabilidad y robustez para manejar grandes volúmenes de datos y consultas complejas, características esenciales para un proyecto que busca crecer y operar a gran escala.

7.15.2. ¿Y por qué se eligió Postgresql y no MySQL?

La elección de PostgreSQL sobre MySQL se basó principalmente en las capacidades avanzadas que PostgreSQL ofrece en comparación con MySQL. Una de las principales ventajas de PostgreSQL es su capacidad para manejar no solo datos relacionales, sino también tipos de datos complejos y objetos, lo que brinda mayor flexibilidad al momento de almacenar información diversa, como estructuras JSON, datos geoespaciales o incluso matrices y tipos de datos personalizados. Esta versatilidad es crucial en un entorno donde se podrían manejar distintos tipos de información y objetos relacionados con el juego y los usuarios.

Por otro lado, aunque MySQL es una base de datos relacional robusta y ampliamente utilizada, está más limitada en cuanto a su capacidad de almacenar tipos de datos avanzados o de realizar operaciones complejas con ellos. MySQL se centra exclusivamente en el manejo de datos relacionales, lo cual puede ser eficiente en ciertos contextos, pero carece de la flexibilidad que ofrece PostgreSQL al combinar la potencia relacional con el soporte de datos no estructurados. Además, PostgreSQL tiene una mejor gestión de transacciones y soporte completo para el estándar SQL, lo que garantiza mayor consistencia y fiabilidad en el manejo de datos críticos para el proyecto.

Por estas razones, se optó por PostgreSQL, ya que no solo permite un almacenamiento eficiente de grandes volúmenes de datos, sino que también ofrece la posibilidad de gestionar datos más complejos y realizar consultas avanzadas con mayor facilidad y eficiencia.

Figura 7.106: Elección del tipo de base de datos PostgreSQL

7.15.3. ¿Y por qué se eligió el estándar ES6?

La adopción del estándar ES6 (ECMAScript 2015) se realizó principalmente para mejorar la legibilidad y la limpieza del código. El uso de las características modernas de ES6 ofrece una sintaxis más clara y concisa, en comparación con la forma tradicional de JavaScript, que en ocasiones puede llevar a código desordenado y difícil de mantener. Al utilizar ES6, se introducen diversas mejoras que optimizan la estructura del código y facilitan su mantenimiento a largo plazo.

Uno de los aspectos clave fue la introducción de las funciones asíncronas y el uso de promesas, que reemplazan el estilo anterior de anidación excesiva de *callbacks*, conocido como *callback hell*. En lugar de manejar grandes bloques de código con múltiples promesas anidadas que dificultaban su comprensión, ES6 permite gestionar flujos asíncronos de manera mucho más limpia y ordenada. Con la sintaxis *async* y *await*, es posible escribir código asíncrono de forma secuencial y más legible, lo que facilita tanto el desarrollo como la depuración.

Además, ES6 introdujo otras características útiles, como la desestructuración de objetos, *let* y *const* para el manejo adecuado de variables, clases para un enfoque orientado a objetos más intuitivo,

y módulos nativos para una mejor organización del código. Estas mejoras permiten escribir código más robusto y modular, facilitando la colaboración entre diferentes desarrolladores y reduciendo errores comunes que solían surgir con las versiones anteriores de JavaScript.

Figura 7.107: Estandar ES6 (ECMAScript 2015)

7.16. Fase de desarrollo: Primer prototipo programado (*Frontend*)

Para la realización del primer prototipo programado del *frontend*, se decidió utilizar la librería React en conjunto con la librería de personalización TailwindCSS, siguiendo el diseño del primer prototipo desarrollado en Figma. Hasta el momento, no se realizaron cambios en este diseño debido a que, durante la presentación del proyecto, que incluía tanto el juego como la página web, varios usuarios mencionaron que el estilo del prototipo era adecuado. Sin embargo, expresaron su deseo de interactuar con el sitio para poder proporcionar un mejor feedback sobre su funcionalidad. Por esta razón, se decidió no modificar el diseño inicial y proceder con su implementación en un prototipo programado.

Una vez establecido este objetivo, el desarrollo del *frontend* se llevó a cabo utilizando React, aplicando los principios SOLID para asegurar la reutilización de componentes. El enfoque principal fue la creación de las pantallas orientadas al jugador, por lo que se implementaron las siguientes vistas:

- Pantalla principal (*Home*)
- Pantalla sobre nosotros (*About*)
- Pantalla de noticias (*News*)
- Pantalla de inicio de sesión (*Login*)
- Pantalla de registro (*Register*)
- Pantalla de estadísticas (*Dashboard*)

El despliegue del *frontend* se realizó utilizando la herramienta Vercel, que también fue empleada para gestionar la API. Esta API es fundamental para la funcionalidad del *login*, ya que cada *endpoint* requiere la verificación de un token para permitir el acceso del jugador. La validación de tokens y la autenticación del usuario en el *Dashboard* se gestionaron mediante promesas y funciones asíncronas que interactúan con la API en producción de nuestro *backend*. Además, para el registro de nuevos

usuarios, se implementó un *endpoint* POST en *Register*, que envía un JSON con la información del nuevo usuario.

En el caso de la pantalla de *News*, que será actualizada por los administradores, se utilizó una implementación provisional utilizando la librería *json-server*. Esta herramienta permitió simular la adición de varias *cards* de noticias para observar el comportamiento y la disposición del diseño cuando se añaden nuevos contenidos.

En cuanto al diseño, también se creó una versión beta del diseño *responsive* de la página. Aquí es donde TailwindCSS mostró sus ventajas frente a otras librerías o *frameworks* de diseño. Al ser *responsive mobile-first*, TailwindCSS facilita la adaptación de los elementos a diferentes tamaños de pantalla mediante el uso de clases predefinidas. Esto permitió que cada pantalla del prototipo, independientemente de su contenido, fuera completamente *responsive* y se adaptara a cualquier tamaño de dispositivo, ofreciendo una experiencia de usuario optimizada en todas las resoluciones.

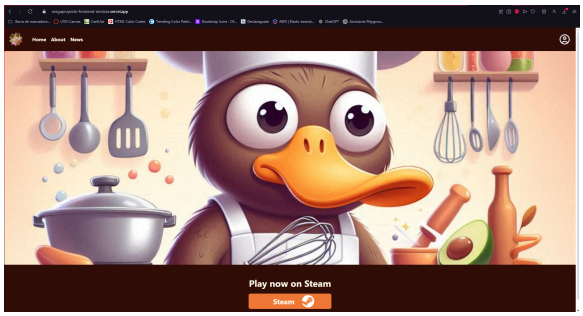

Figura 7.108: Primer prototipo *frontend* de *Home Page* en Producción

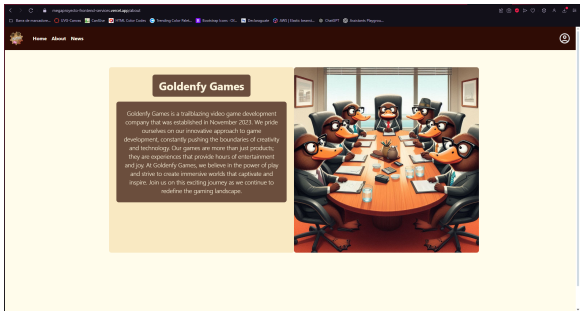

Figura 7.109: Primer prototipo *frontend* de *About Page* en Producción

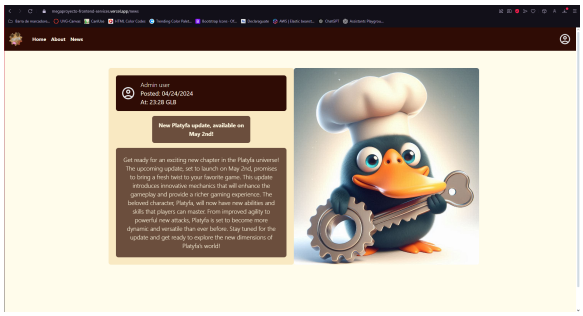

Figura 7.110: Primer prototipo *frontend* de *News Page* en Producción

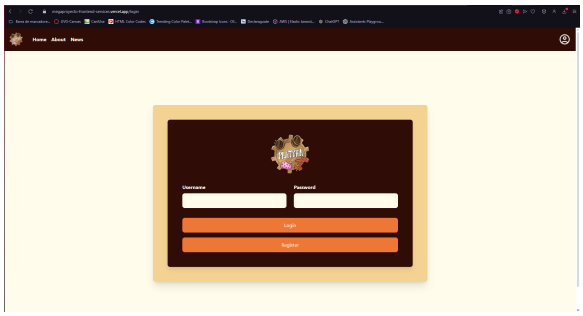

Figura 7.111: Primer prototipo *frontend* de *Login Page* en Producción

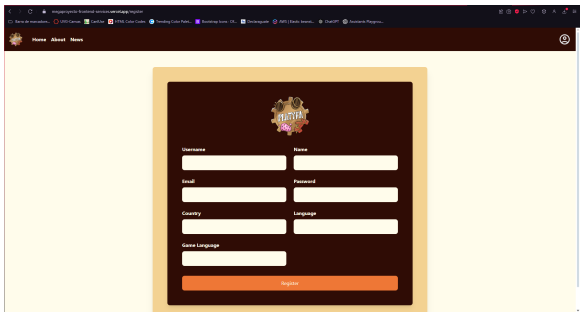

Figura 7.112: Primer prototipo *frontend* de *Register Page* en Producción

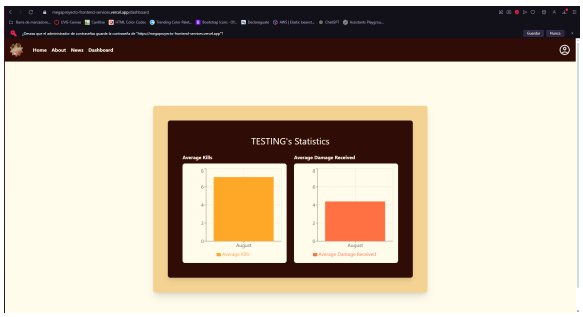

Figura 7.113: Primer prototipo *frontend* de *Dashboard Page* en Producción

7.17. Fase de *testing*: Primer *testing* con usuarios

Una vez que el *frontend* y el *backend* fueron implementados y unificados en el entorno de producción, se decidió realizar un *testing* con usuarios. Para ello, se participó en el evento GameDev Quest.

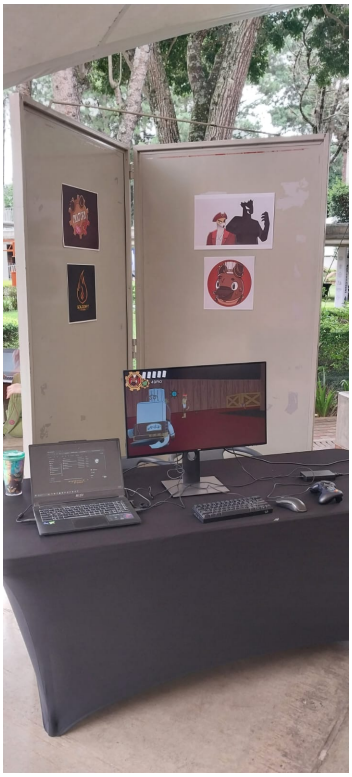

Figura 7.114: Colocación de la página con el juego

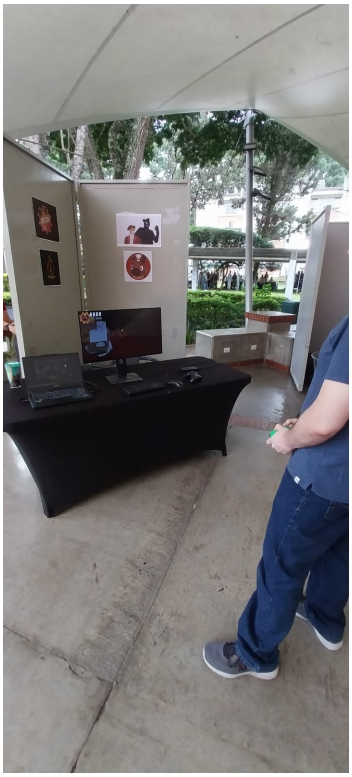

Figura 7.115: Usuario probando el juego con la página

Durante el evento, basado en la Ley de Hick, cómo también, en la metodología UCD, lo que se consideró realizar fue la toma del primer feedback de la página, esto mediante la interacción con el juego y el funcionamiento de la página.

Al finalizar dicha presentación se obtuvo que, varios les hubiera gustado que se implementara una pantalla para que el usuario pudiera editar la información de su perfil, una función que aún no ha sido desarrollada. Además, señalaron que sería ideal aplicar las mismas tonalidades de colores y texturas del juego a los componentes de la página, para lograr una mayor cohesión estética.

Los usuarios también sugirieron que las imágenes mostradas en la página web fueran propias del juego y no generadas por IA, lo que reforzaría la autenticidad visual del sitio. En la sección *About*, recomendaron incluir información sobre las personas que participaron en el desarrollo del proyecto, mientras que en la sección *News*, señalaron que preferirían ver noticias relacionadas con el desarrollo del juego y su contenido, en lugar de información generada por IA.

En cuanto a la funcionalidad futura de *EditProfile*, los usuarios sugirieron que esta incluyera la opción de subir una foto de perfil, eliminar la cuenta, y permitir al usuario cambiar tanto su nombre de usuario como su contraseña.

En términos de diseño, varios usuarios mencionaron que sería útil contar con un indicador visual que separara la barra de navegación del contenido principal para mejorar la claridad en la navegación. También sugirieron que, al subir una foto de perfil, esta apareciera en una de las esquinas de la página, lo que mejoraría la visualización y personalización del perfil del usuario.

Como observaciones adicionales, algunos jugadores sugirieron incluir contenido multimedia sobre el juego en la página, una recomendación alineada con lo que se observó en otras páginas web analizadas previamente durante la investigación de antecedentes.

7.18. Fase de desarrollo: Tercer prototipo (*Backend*)

Tras la primera etapa de *testing* con usuarios, se identificaron diversas mejoras necesarias en la gestión de cuentas, enfocadas en permitir una mayor personalización y control para los usuarios en el *backend*. Esta fase se centró en implementar funcionalidades que permitieran a los usuarios gestionar sus perfiles y cuentas de manera más flexible, asegurando una experiencia de usuario más completa y segura. Para lograr esto, se realizaron modificaciones tanto en los *endpoints* existentes como en la base de datos, incluyendo la creación de nuevos campos y funcionalidades.

7.18.1. Implementación de la funcionalidad de eliminación de cuenta

Se añadió un *endpoint* que permite a los usuarios marcar sus cuentas como eliminadas. Este cambio se refleja en el *backend* al modificar el estado del campo is *deleted* a TRUE en la base de datos. Aunque la cuenta se deshabilita temporalmente, los datos del usuario no se eliminan de forma permanente, lo que permite la posibilidad de una restauración futura.

```
// Endpoint para marcar un jugador como eliminado
playersRouter.delete("/delete/:id", authenticateToken, async (req, res) => {
  const playerId = req.params.id;

  try {
    const query = `
      UPDATE player SET is_deleted = TRUE WHERE id = $1
    `;
    const result = await clientDB.query(query, [playerId]);

    if (result.rowCount === 0) {
      return res
        .status(404)
        .json({ success: false, message: "Player not found" });
    }

    res.status(200).json({
      success: true,
      message: "Player deleted successfully",
    });
  } catch (err) {
    console.error(err);
    res.status(500).json({ success: false });
  }
});
```

Figura 7.116: *Endpoint* para marcar una cuenta de usuario como eliminada

7.18.2. Funcionalidad de restauración de cuenta

Para permitir la reactivación de cuentas deshabilitadas, se añadió un *endpoint* que permite a los usuarios restaurar su acceso. La restauración se basa en el ID del usuario almacenado en el token de autenticación, lo que permite verificar la identidad del usuario de manera segura y precisa. El campo *is deleted* se actualiza a FALSE, lo que reactiva la cuenta en la base de datos.

```
// Endpoint para restaurar una cuenta eliminada
playersRouter.post("/restore", authenticateToken, async (req, res) => {
  const { playerId } = req.body;
  const requesterId = req.userId; // ID del usuario del token
  const isAdmin = req.isAdmin; // Verificar si es administrador desde el token

  try {
    // Verificar si el usuario es administrador
    const targetPlayerId = isAdmin ? playerId : requesterId;

    // Restaurar la cuenta del usuario objetivo
    const query = `
      UPDATE player SET is_deleted = FALSE WHERE id = $1
    `;
    const result = await clientDB.query(query, [targetPlayerId]);

    if (result.rowCount === 0) {
      return res
        .status(404)
        .json({ success: false, message: "Player not found" });
    }

    res
      .status(200)
      .json({ success: true, message: "Account restored successfully" });
  } catch (err) {
    console.error(err);
    res.status(500).json({ success: false, message: "Server error" });
  }
});
```

Figura 7.117: *Endpoint* para restaurar una cuenta de usuario eliminada

7.18.3. Modificación del inicio de sesión para detectar cuentas eliminadas

El sistema de inicio de sesión fue modificado para incluir una verificación adicional del estado de la cuenta del usuario. Si el campo *is deleted* está marcado como TRUE, se deniega el acceso y se devuelve un mensaje informando al usuario que su cuenta está deshabilitada.

```
// Endpoint para iniciar sesión
playersRouter.post("/login", async (req, res) => {
  const { username, password } = req.body;

  try {
    const query = `
      SELECT * FROM player WHERE username = $1
    `;
    const result = await clientDB.query(query, [username]);

    if (result.rows.length === 0) {
      return res.status(403).json({ message: "User incorrect" });
    }

    const user = result.rows[0];
    const token = jwt.sign(
      {
        id: user.id,
        username: user.username,
        is_deleted: user.is_deleted,
        is_admin: user.is_admin,
      },
      process.env.JWT_SECRET,
      {
        expiresIn: "1d",
      }
    );

    // Verificar si la cuenta está eliminada
    if (user.is_deleted) {
      return res
        .status(403)
        .json({ message: "Account is deleted", token: token });
    }
```

Figura 7.118: Verificación de cuentas eliminadas al iniciar sesión

7.18.4. Acceso a la información del usuario

Se implementó un *endpoint* que permite a los usuarios acceder a su información personal mediante su ID. La información recuperada incluye detalles como nombre de usuario, correo electrónico, país, imagen de perfil y preferencias de idioma. Estos datos se obtienen de la tabla *player*, lo cual permite mostrar información personalizada en el *frontend*.

```
// Endpoint para obtener la información de un jugador por ID
playersRouter.get("/:id", authenticateToken, async (req, res) => {
  const playerId = req.params.id;

  try {
    const query =
      "SELECT username, email, country, profile_image, language FROM player WHERE id = $1";
    const result = await clientDB.query(query, [playerId]);

    if (result.rows.length === 0) {
      return res
        .status(404)
        .json({ success: false, message: "Player not found" });
    }

    const player = result.rows[0];

    res.status(200).json({ success: true, player });
  } catch (err) {
    console.error("Error fetching player data:", err);
    res.status(500).json({ success: false, message: "Server error" });
  }
});
```

Figura 7.119: Acceso a la información del usuario por ID

7.18.5. Funcionalidad para actualizar el perfil del usuario

Se implementó un *endpoint* para permitir la actualización de varios aspectos del perfil del usuario, como nombre de usuario, país, correo electrónico, imagen de perfil y contraseña. Para garantizar la seguridad, las contraseñas se encriptan antes de almacenarse en la base de datos, y solo se actualizan los campos especificados por el usuario.

```
playersRouter.put("/update_profile", authenticateToken, async (req, res) => {
  const { username, country, password, profileImage, email } = req.body;
  try {
    let updates = [];
    let values = [];
    let index = 1;

    if (username) {
      updates.push(`username = $${index}`);
      values.push(username);
      index++;
    }

    if (country) {
      updates.push(`country = $${index}`);
      values.push(country);
      index++;
    }

    if (profileImage) {
      updates.push(`profile_image = $${index}`);
      values.push(profileImage); // URL de la imagen
      index++;
    }

    if (email) {
      updates.push(`email = $${index}`);
      values.push(email);
      index++;
      //updates.push(`is_verified = $${index}`);
      //values.push(false);
      //index++;
    }
```

Figura 7.120: Funcionalidad para la actualización del perfil del usuario

7.18.6. Modificación en la base de datos para soportar los estados de cuenta

Para soportar las nuevas funcionalidades de gestión de cuentas, se añadió el campo *is deleted* de tipo booleano en la tabla de usuarios (*player*). Este campo se utiliza para indicar si una cuenta está activa (FALSE) o deshabilitada (TRUE), lo cual permite una administración más eficiente y segura de los perfiles de usuario.

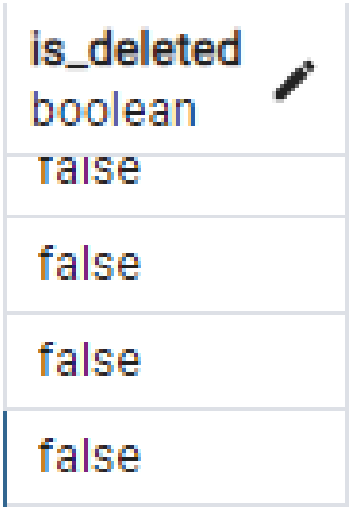

Figura 7.121: Campo *deleted* en la base de datos

7.19. Fase de desarrollo: Segundo prototipo programado (*Frontend*)

Siguiendo las opiniones y sugerencias obtenidas del lado del *frontend* durante la primera etapa de *testing* con usuarios, se realizaron una serie de cambios en el diseño y funcionalidad del *frontend* de la página web para mejorar la experiencia de usuario y alinear la estética de la web con el videojuego *Platyfa*.

7.19.1. Ajuste de la estética general

Uno de los principales cambios implementados fue la modificación de la paleta de colores, las texturas y las sombras de la página para que combinaran con la estética visual del juego. Esto incluyó la incorporación de tonos, texturas y efectos que reflejan la atmósfera y el estilo visual de *Platyfa*, lo que mejora la cohesión estética entre la página y el videojuego.

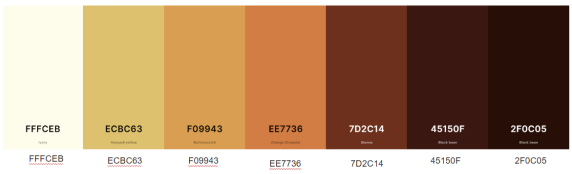

Figura 7.122: Nueva paleta de colores y tonalidades en el *frontend*

7.19.2. Implementación de la sección de noticias

Se añadió una sección completamente funcional de noticias, la cual ahora muestra contenido generado directamente por el equipo de desarrollo, en lugar de utilizar información generada por inteligencia artificial. Esto mejora la autenticidad del contenido presentado, proporcionando a los usuarios información actualizada y relevante sobre el desarrollo del juego.

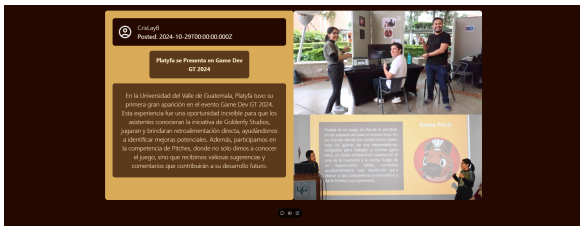

Figura 7.123: Primera noticia generada por el equipo de desarrollo

7.19.3. Mejoras en la sección (*About*)

En el apartado *About*, se incluyeron imágenes e información auténtica, asegurando que no fueran generadas por inteligencia artificial, lo cual responde directamente a las sugerencias de los usuarios sobre la autenticidad visual del contenido. Además, se agregó una sección dedicada al equipo de desarrollo, mostrando a todos los integrantes del proyecto con su respectiva foto y rol en el desarrollo del videojuego.

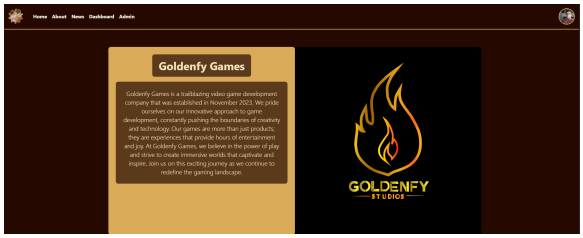

Figura 7.124: Contenido no generado por IA en la sección *About*

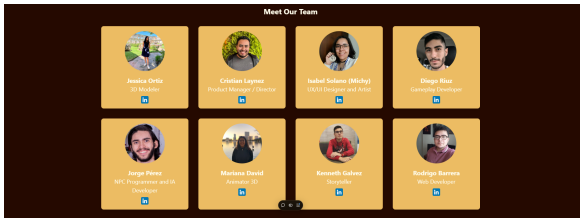

Figura 7.125: Sección con integrantes del equipo de desarrollo en *About*

7.19.4. Funcionalidad de edición de perfil

Se añadió una funcionalidad completa para la personalización del perfil del usuario, permitiendo que los usuarios puedan editar su perfil. Esta nueva funcionalidad incluye la capacidad de:

- Subir una foto de perfil.
- Eliminar la cuenta de usuario.
- Actualizar el nombre de usuario, país, idioma de preferencia y lenguaje del juego.

Además, se implementó la opción de que la foto de perfil subida por el usuario se muestre en la esquina superior de la página, mejorando así la personalización y representación del usuario en la web.

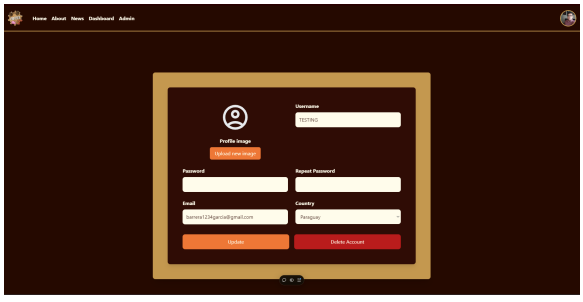

Figura 7.126: Interfaz de edición de perfil con opciones de personalización

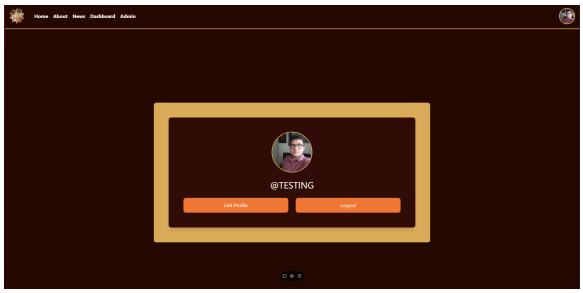

Figura 7.127: Visualización de la foto de perfil en la página

7.19.5. Mejoras en la interfaz de navegación

Para mejorar la claridad en la navegación, se añadió un indicador visual que separa la barra de navegación del contenido principal de la página, respondiendo directamente a las sugerencias de los usuarios sobre la necesidad de una separación más clara entre estas secciones.

Figura 7.128: Separación visual entre la barra de navegación y el contenido principal

7.19.6. Cambios en el apartado de estadísticas (*Dashboard*)

Utilizando el diseño centrado en el usuario (UCD) y una arquitectura de información modular, se implementaron cambios significativos en el diseño del *Dashboard*. Ahora, cada tipo de estadística tiene su propia *card*, lo cual permite una visualización clara y organizada de la información para los usuarios. Este diseño modular facilita la comprensión y acceso rápido a los diferentes tipos de datos estadísticos presentados en la página.

Además, se realizó un ajuste en la paleta de colores del *Dashboard* para que esté alineada con la estética del juego, asegurando una experiencia de usuario más coherente y visualmente atractiva.

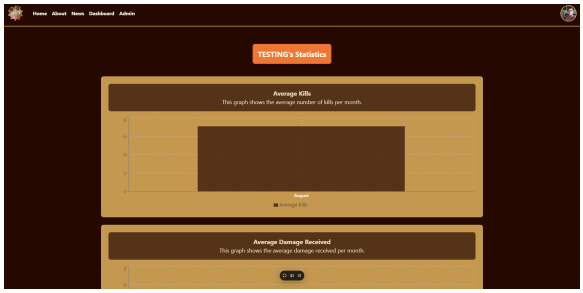

Figura 7.129: Nuevo diseño del *Dashboard* con *cards* para cada tipo de estadística

7.19.7. Detección de cuentas deshabilitadas

Durante el desarrollo del segundo prototipo, se añadió una funcionalidad adicional para que el *frontend* detecte cuando una cuenta de usuario se encuentra deshabilitada. Esta funcionalidad permite al usuario optar por restaurar su cuenta si lo desea.

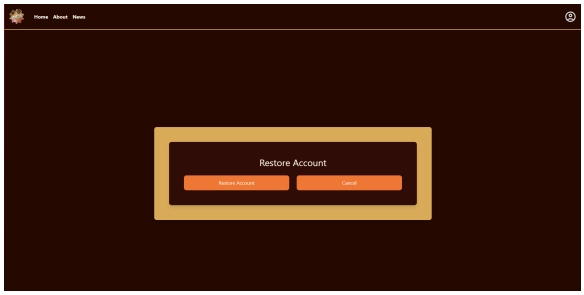

Figura 7.130: Interfaz para restaurar cuentas deshabilitadas

7.20. Fase de *testing*: Segundo *testing* con usuarios

Una vez finalizada la primera etapa de *testing* y la etapa de desarrollo posterior a ella, se determinó que era necesario establecer un tamaño de muestra específico para el siguiente *testing* con usuarios. Para definir la cantidad mínima de encuestados requerida, se utilizó el cálculo de muestreo estadístico, considerando tanto el nivel de confianza como el margen de error para asegurar que los resultados sean representativos de la población objetivo.

7.20.1. Población objetivo

Este estudio se realizó en Guatemala, considerando una población total de 15,841,063 personas, correspondiente a individuos en un rango de edad de 6 a 65 años. La elección de este rango de edad se debe a las siguientes razones:

- **Acceso a la tecnología**: Se asumió que la mayoría de las personas dentro de este rango de edad tiene acceso a dispositivos electrónicos, como computadoras, tabletas y teléfonos móviles, lo cual es fundamental para la interacción con la web en estudio.
- **Usuarios potenciales**: El rango de edad seleccionado representa a la mayoría de los usuarios potenciales del sitio web, ya que incluye tanto a usuarios jóvenes (en edad escolar), como a adultos en edad laboral e individuos mayores que siguen utilizando la tecnología de manera activa. Este rango permite captar una diversidad de opiniones y comportamientos en cuanto al uso y percepción del sitio web.
- **Relevancia de usabilidad**: Personas a partir de los 6 años tienen la capacidad de realizar interacciones básicas con la interfaz, mientras que hasta los 65 años, la mayoría de las personas se considera familiarizada con la navegación web. De este modo, se puede evaluar la usabilidad de manera más precisa para quienes realmente van a utilizar la web.

7.20.2. Cálculo del tamaño de la muestra

El tamaño de la muestra se calculó utilizando la siguiente fórmula para poblaciones grandes, que no requieren ajuste por ser finitas:

$$n = \frac{Z^2 \cdot p \cdot (1-p)}{E^2} \tag{7.1}$$

Donde:

- n: Tamaño de la muestra inicial.
- Z: Valor Z del nivel de confianza. En este caso, para un nivel de confianza del 90 %, se usó $Z = 1.645$.
- p: Proporción esperada de variabilidad en la población. Se asume $p = 0.5$ para maximizar el tamaño de la muestra y considerar la mayor variabilidad posible.
- E: Margen de error aceptado. Se estableció un margen de error del 10 %, lo cual equivale a $E = 0.10$.

Al sustituir los valores en la fórmula, se obtiene:

$$n = \frac{1.645^2 \cdot 0.5 \cdot (1-0.5)}{0.10^2} = 67.65 \tag{7.2}$$

Esto indica que el tamaño de muestra inicial es de aproximadamente **68 personas**.

7.20.3. Ajuste para población finita

Dado que la población objetivo es finita, con un tamaño total de 15,841,063 personas, se aplicó un ajuste para obtener el tamaño de muestra más preciso:

$$n_{ajustado} = \frac{n}{1 + \left(\frac{n-1}{N}\right)} \tag{7.3}$$

Donde:

- $n_{ajustado}$: Tamaño de la muestra ajustado para población finita.
- n: Tamaño de la muestra inicial calculado previamente (68 personas).
- N: Tamaño de la población objetivo, en este caso $15,841,063$.

Sustituyendo los valores:

$$n_{ajustado} = \frac{67.65}{1 + \left(\frac{67.65-1}{15,841,063}\right)} \approx 67.65 \tag{7.4}$$

En este caso, el tamaño de la muestra ajustado sigue siendo prácticamente el mismo, dado que la población total es muy grande en comparación con el tamaño de la muestra inicial.

El tamaño de la muestra ajustado de **68 personas** es válido y representativo para este segundo testing con usuarios en Guatemala. Un margen de error del 10 % es apropiado para estudios exploratorios como el testing de usabilidad de una web, ya que permite obtener conclusiones generales sobre la experiencia de los usuarios sin requerir una precisión extrema. Además, el nivel de confianza del 90 % asegura que, en 9 de cada 10 encuestas similares, los resultados estarán dentro del margen de error especificado.

Tras finalizar la primera etapa de *testing* y la fase de desarrollo posterior, se realizó un segundo **testing** con usuarios para evaluar la usabilidad y funcionalidad del sitio web. Para definir la cantidad de participantes necesaria, se utilizó un cálculo estadístico de muestra, considerando un nivel de confianza del 90 % y un margen de error del 10 %, lo cual resultó en una muestra mínima de 68 personas. Sin embargo, se obtuvieron 94 respuestas en total, superando el tamaño de muestra ajustado.

7.20.4. Metodología del segundo *testing* con usuarios

Una vez definida la cantidad de muestra, se procedió a realizar el segundo *testing* de virtual. A cada participante se le proporcionó un formulario con el enlace de acceso a la página web, permitiéndoles navegar e interactuar con las diversas secciones. Se establecieron tres puntos clave para la evaluación, que sirvieron de base para el diseño de las preguntas:

- **Usabilidad general:** Evaluar la facilidad de navegación y la claridad de la información.
- **Experiencia del Usuario (UX):** Evaluar la percepción estética y la adaptabilidad de la página en dispositivos móviles y tablets.
- **Funcionalidad específica:** Evaluar el correcto funcionamiento de formularios y elementos interactivos.

Con base en estos puntos, se plantearon las siguientes preguntas:

1. ¿Qué tan fácil fue navegar por la página web?
2. ¿Consideras que la información está organizada de manera clara y comprensible?
3. ¿Cómo calificarías la apariencia del sitio en tu dispositivo móvil/tablet?
4. ¿Los botones y enlaces eran fáciles de hacer clic en la versión móvil/tablet?

Una vez definidas las preguntas, se tomó la decisión de realizar la encuesta de manera general, permitiendo alcanzar el tamaño de muestra necesario. La encuesta fue distribuida a través del correo universitario, brindando acceso a un enlace directo a la página web para que los usuarios pudieran interactuar libremente con ella.

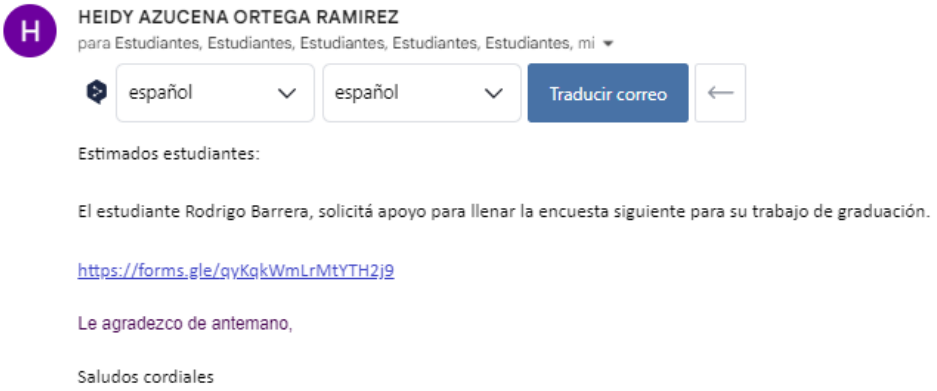

Figura 7.131: Divulgación de encuesta mediante el correo universitario

Esto con el objetivo de poder obtener a través del encuestado y cada una de las preguntas enfocadas a la usabilidad de la página web, poder realizar el análisis del feedback correspondiente para poder mejorar cada una de las áreas a las que están enfocadas dichas preguntas, desde la parte de navegación, hasta la parte de diseño y con esto ir orientando a tener un resultado mucho más enfocado en el usuario objetivo que se quiere cumplir.

7.21. Fase de desarrollo: Cuarto prototipo (*Backend*)

En esta fase, se implementaron mejoras en el *backend* a raíz de las sugerencias obtenidas en el segundo *testeo* de usuarios. Entre las recomendaciones destacadas en los resultados, los usuarios sugirieron la posibilidad de visualizar múltiples noticias y recibir soporte en caso de no poder reabrir sus cuentas. Con base en estas solicitudes, se decidió incorporar una sección de administración que permitiera una mayor flexibilidad en el manejo de la información y el soporte al usuario.

Para lograr esto, se añadió un campo `is_admin` a la tabla `Players`. La inclusión de este campo directamente en la tabla de usuarios permite una gestión centralizada de la información y simplifica la arquitectura de la base de datos al evitar la creación de una tabla adicional para roles de usuarios. Esto facilita el acceso y verificación del rol de *admin* sin necesidad de realizar consultas adicionales, optimizando así el rendimiento y la eficiencia del sistema.

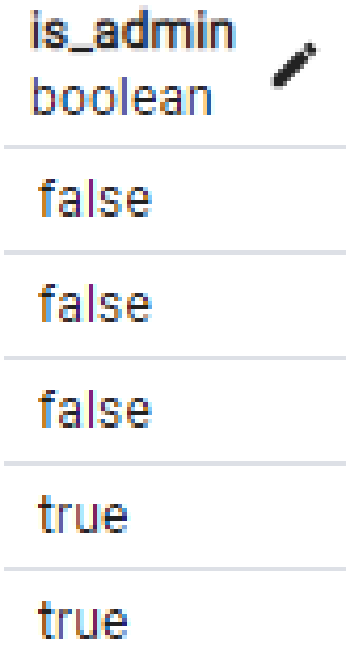

Figura 7.132: Campo para el manejo de rol administrador

Dado que no existía previamente un usuario con rol de administrador, se implementó un 'usuario quemado' o predeterminado con permisos de administrador. Esta práctica es útil en entornos de desarrollo y pruebas, ya que garantiza que haya un acceso de administrador disponible para configurar y probar las funcionalidades críticas de administración desde el inicio. Al tener este usuario predeterminado, fue posible verificar y asegurar el correcto funcionamiento de las nuevas funcionalidades sin depender de la creación manual de un *admin* en cada entorno.

Como primer paso en la implementación de la sección de administración, se creó un controlador exclusivo para el manejo de los *endpoints* específicos de los administradores. Este enfoque por componentes es recomendable, ya que organiza el código en módulos específicos que se encargan de funcionalidades particulares, en este caso, el rol de administrador. Esto no solo mejora la legibilidad del código, sino que también facilita el mantenimiento y la expansión del sistema, permitiendo modificar o añadir funcionalidades sin afectar otros módulos.

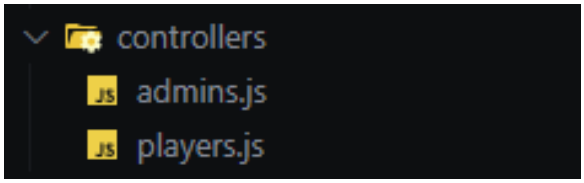

Figura 7.133: Nuevo controlador para manejar peticiones *admin*

Una vez definido el controlador de administración, se implementó la lógica para identificar si el usuario tenía rol de *admin*. Para ello, se modificó el *endpoint* de `login` para que recuperara el campo `is_admin` del usuario y lo incluyera en el token de autenticación. Este valor se adjunta en el token, lo cual permite una verificación constante del rol del usuario sin tener que consultar la base de datos en cada solicitud. Esta práctica de incluir roles o permisos en el token es eficiente y segura, ya que minimiza el número de consultas a la base de datos y facilita el control de accesos de forma ágil y confiable.

```
// Middleware para restringir acceso solo a administradores
const isAdmin = (req, res, next) => {
  if (!req.isAdmin) {
    return res
      .status(403)
      .json({ message: "Access restricted to admins only" });
  }
  next();
};
```

Figura 7.134: Interfaz para restaurar cuentas deshabilitadas

Con la autenticación de administradores establecida, se procedió a desarrollar diversos *endpoints*. El primero de estos fue un *endpoint* que permite a los administradores obtener un listado de todas las cuentas almacenadas en la base de datos. En este caso, se verifica que el usuario que realiza la solicitud tenga el rol de *admin*, utilizando la información del token, y, tras esta verificación, se le proporciona acceso a la lista completa de usuarios.

Además, se implementó un *endpoint* para que los administradores pudieran agregar noticias. Para ello, fue necesario añadir una nueva tabla a la base de datos que almacene las noticias. Esta tabla incluye campos para mostrar la imagen de perfil del administrador que publica la noticia, su nombre de usuario, la fecha de publicación, así como el contenido y la imagen de la noticia. El *endpoint* ***`news`*** se encarga de validar primero que el usuario sea *admin* y, posteriormente, inserta la nueva noticia en la tabla correspondiente.

```
// Endpoint para agregar una noticia
adminsRouter.post("/add_new", authenticateToken, isAdmin, async (req, res) => {
  const { title, author, content, image } = req.body; // Se eliminan 'date' y 'time' del body

  try {
    const query = `
      INSERT INTO news (title, author, date, time, content, image)
      VALUES ($1, $2, CURRENT_DATE, CURRENT_TIME, $3, $4) RETURNING *
    `;
    const values = [title, author, content, image];
    const result = await clientDB.query(query, values);

    res.status(201).json({
      success: true,
      message: "News added successfully",
      news: result.rows[0],
    });
  } catch (err) {
    console.error("Error adding news:", err);
    res.status(500).json({ success: false, message: "Server error" });
  }
});
```

Figura 7.135: *Endpoint* de administración para publicar noticias

Adicionalmente, se creó un *endpoint* de ***`news`*** que no requiere autenticación, permitiendo a cualquier visitante de la página web ver las noticias publicadas. Esta decisión facilita el acceso público a la información sobre el proyecto, fomentando el interés y la participación de usuarios potenciales, quienes pueden mantenerse informados sin necesidad de registrarse o autenticarse en el sistema.

```
// Endpoint para obtener todas las noticias
playersRouter.get("/news", async (req, res) => {
  try {
    const query = `
      SELECT id, title, author, date, content, image
      FROM news
      ORDER BY date DESC, id DESC
    `;
    const result = await clientDB.query(query);

    if (result.rows.length === 0) {
      return res.status(404).json({ success: false, message: "No news found" });
    }

    res.status(200).json({ success: true, news: result.rows });
  } catch (err) {
    console.error("Error fetching news:", err);
    res.status(500).json({ success: false, message: "Server error" });
  }
});
```

Figura 7.136: *Endpoint* de administración para publicar noticias

Finalmente, para facilitar la administración y los permisos de otros usuarios, el usuario administrador predeterminado fue utilizado para probar la funcionalidad de autenticación de *admin* y para gestionar los roles de otros usuarios. A través de un *endpoint* `make_admin`, se permite seleccionar a un usuario, obtener su `id` y actualizar su rol para otorgarle permisos de administrador. Adicionalmente, el administrador tiene la posibilidad de *banear* o desactivar cuentas, permitiendo así una gestión integral del sistema.

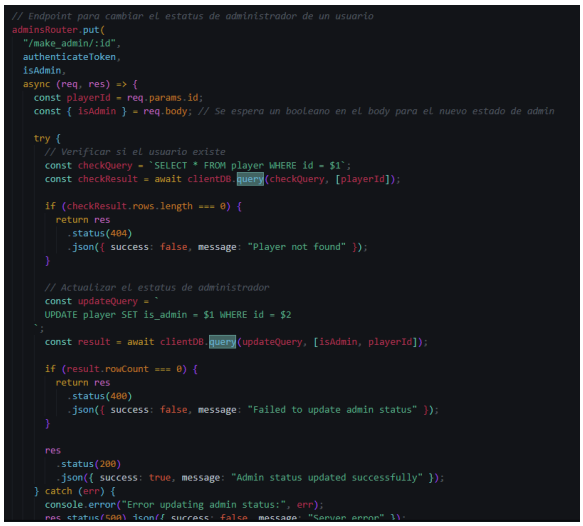

Figura 7.137: *Endpoint* de administración para dar rol al usuario

Es una buena práctica mantener un único rol de administrador con todos los permisos necesarios para la gestión del sistema. Esto centraliza el control en una sola entidad autorizada, lo cual simplifica el manejo de permisos y reduce el riesgo de inconsistencias o errores en la administración. Al contar con un rol de *admin* que pueda publicar noticias, asignar roles de *admin* a otros usuarios y gestionar (*banear* o desactivar) cuentas, se asegura que la administración del sistema sea coherente y eficiente, manteniendo un flujo de control claro y sin fragmentación en las responsabilidades de administración.

7.22. Fase de desarrollo: Tercer prototipo programado (*Frontend*)

En esta fase, se desarrolló la interfaz de usuario para el manejo de la administración de la página web, implementando diversas modificaciones que permiten una diferenciación de roles entre usuarios y administradores. Estas modificaciones adaptan la experiencia del *frontend* de acuerdo con el rol del usuario, proporcionando un acceso exclusivo a las funcionalidades de administración. Todo el desarrollo de estas implementaciones sigue el patrón de diseño centrado en el usuario, asegurando que tanto la administración como la experiencia general en el sitio sean intuitivas y accesibles.

Para lograr esta adaptación, se realizó una verificación mediante el token de autenticación. Primero, el sistema valida que la cuenta del usuario no esté eliminada. Luego, confirma que el token corresponda al usuario que intenta acceder. Finalmente, se añade una comprobación que verifica si el token incluye el rol de administrador, siguiendo la implementación realizada en el backend. Esta secuencia de validaciones asegura que solo los usuarios con privilegios de administrador puedan acceder a las funcionalidades adicionales de administración.

Una vez que el *frontend* verifica que el usuario tiene rol de administrador, el *Header* de la página muestra una nueva pestaña llamada *Admin*. Esta pestaña sirve como punto de entrada a

las funcionalidades administrativas. Dentro de la sección de administración, se implementaron dos componentes principales:

Figura 7.138: Panel de administración de usuarios

7.22.1. *Admin* panel

El `Admin Panel` utiliza el *endpoint* correspondiente para obtener y mostrar toda la información de los usuarios almacenados en la base de datos. Junto a cada usuario, se despliegan dos botones: `Deleted` y `Make Admin`. Si no se ha interactuado previamente con un usuario, los botones aparecerán en su estado inicial (`Deleted` y `Make Admin`). Sin embargo, al cambiar el estado de un usuario, el botón de `Deleted` se transformará en `Active`, y el de `Make Admin` pasará a `Revoke Admin`, reflejando los cambios en el rol o estado del usuario.

El botón de `Deleted` utiliza el mismo *endpoint* de eliminación implementado para los usuarios en el *backend*, pero ahora se accede desde el panel de administración. En cuanto al botón de `Make Admin`, se emplea el *endpoint* de otorgamiento de privilegios de administrador desarrollado en la fase de *backend* anterior.

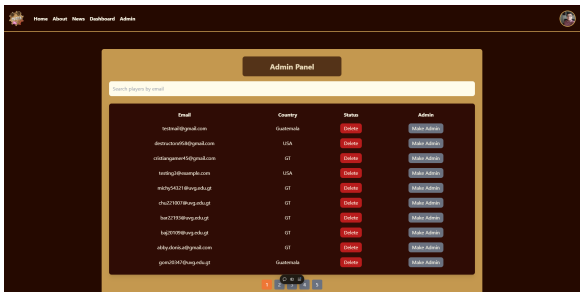

Figura 7.139: Panel de administración de usuarios

7.22.2. Panel de publicación de noticias

Debajo del `Admin Panel`, se incluyó otro panel que permite a los administradores publicar noticias en la página. Este panel consiste en un formulario que solicita un título, una descripción y una imagen para la noticia. Al presionar el botón `Add News`, el formulario envía estos datos mediante el *endpoint* de creación de noticias elaborado en la fase anterior del *backend*.

Con estas funcionalidades, el *frontend* no solo permite una administración efectiva de usuarios y roles, sino que también facilita la publicación de noticias por parte de los administradores, brindando a la página web una estructura robusta y adaptable para su gestión.

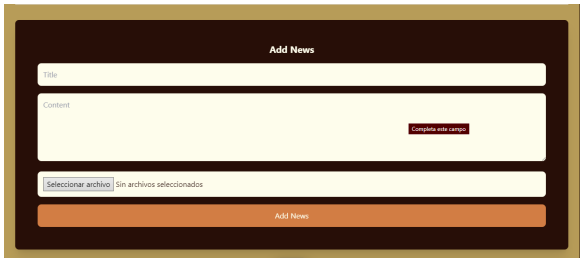

Figura 7.140: Panel de administración de noticias

7.23. Fase de *testing*: Tercer *testing* con usuarios

Para esta fase de *testing*, y tras analizar los resultados obtenidos en la segunda etapa de testeo, se optó por realizar un enfoque mucho más centralizado en la usabilidad de la página web. Este tercer testeo tuvo como objetivo específico evaluar la facilidad de uso y la efectividad en la navegación e interacción del usuario con la web. Para lograrlo, se partió de las mismas cuatro preguntas planteadas en las fases anteriores:

1. ¿Qué tan fácil fue navegar por la página web?
2. ¿Consideras que la información está organizada de manera clara y comprensible?
3. ¿Cómo calificarías la apariencia del sitio en tu dispositivo móvil/tablet?
4. ¿Los botones y enlaces eran fáciles de hacer clic en la versión móvil/tablet?

Con el objetivo de obtener datos más precisos sobre la usabilidad, se definió un grupo de prueba reducido y seleccionado. Basándonos en el estudio de Jakob Nielsen sobre pruebas de usabilidad, se estableció que un grupo de cinco personas es suficiente para detectar la mayoría de los problemas de usabilidad [21, 35]. Considerando, además, el tamaño de muestra calculado en la segunda fase de *testing*, se decidió realizar este tercer testeo con el 10 % de la muestra total, lo cual corresponde aproximadamente a 7 personas. Este número no solo cumple con la recomendación de Nielsen, sino que también respeta la cantidad mínima de muestreo establecida previamente.

Este testing fue realizado tanto de manera virtual como presencial. Para el testing virtual, se utilizó la plataforma Zoom. Se envió un enlace de invitación a los participantes y, una vez dentro de la reunión, se solicitó al moderador, Cristian Laynez, que creara una sala privada para cada participante. Esto permitió llevar a cabo el testing en un entorno más privado y sin interrupciones, lo que facilitó la concentración y mejoró la calidad de la retroalimentación.

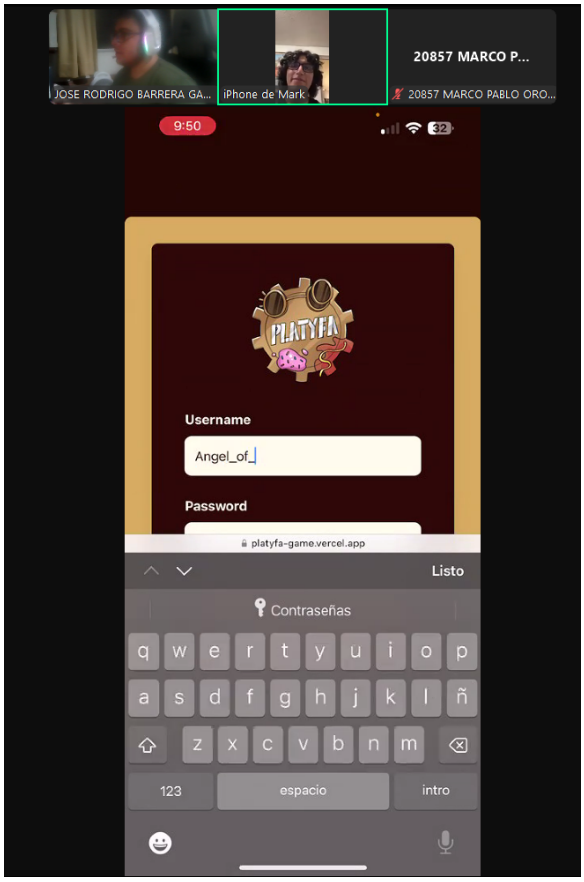

Figura 7.141: Sala de testing donde se estaba probando la usabilidad móvil.

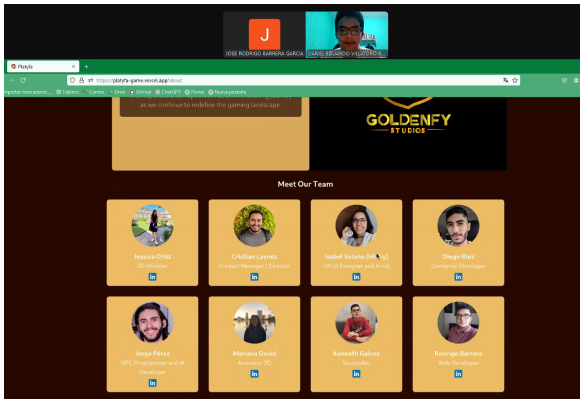

Figura 7.142: Sala de testing donde se estaba probando la usabilidad desktop

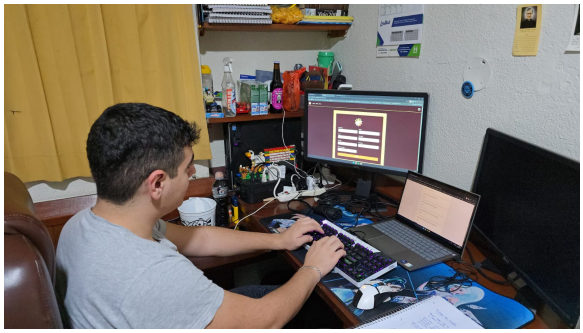

Figura 7.143: *Testing* de usabilidad *desktop* de manera presencial

Una vez definida la cantidad de participantes para el *testing*, se procedió a evaluar el tiempo que cada uno de ellos necesitaba para completar diversas tareas específicas dentro de la página web, tanto en su versión de escritorio como en la versión móvil. Estas tareas fueron diseñadas para cubrir cada una de las secciones de la web y proporcionar información detallada sobre la facilidad de uso de cada una. Las tareas asignadas fueron las siguientes:

- Encontrar en una de las imágenes el título *game pitch*.
- Identificar quién es la persona que tiene el rol de *Animator 3D*.
- Ubicar el gráfico de partidas ganadas y perdidas.
- Encontrar la forma de registrarse en la página.
- Encontrar la forma de iniciar sesión en la página.
- Ubicar la opción para editar el perfil en la página.

Con las tareas y las preguntas de retroalimentación establecidas, cada participante completó el testeo en ambas versiones de la web. Con el objetivo de poder analizar cada uno de los resulados en matrices de retroalimentación 2x2, en las cuales se evaluaron la eficacia y la rapidez en la realización de las tareas con base al tiempo que le tomó a cada usuario en realizar cada una de las tareas, y con ello cumplir con otra sección del estudio de Jakob Nielsen, en donde establece que el rango para cumplir una tarea dentro de un software o aplicación en general debe estar entre 10 a 20 segundos, y con base a ello lograr identificar áreas específicas de mejora en la usabilidad de la web y asegurar una experiencia de usuario optimizada.

7.24. Fase de desarrollo: Quinto prototipo (*Backend*)

En esta fase, se implementaron funcionalidades avanzadas en el *backend* con el propósito de cumplir dos objetivos clave en la gestión y promoción de la página web de *Platyfa*.

El primer objetivo se centró en la creación de estadísticas administrativas en un dashboard interactivo, alineado con el siguiente objetivo específico:

Gestionar la página web de Platyfa, incluyendo un dashboard interactivo con análisis de datos del último mes del juego. Enfocándose en métricas clave como el crecimiento de la base de jugadores, la actividad general de los mismos, y las tendencias generales de juego.

Para la implementación de esta funcionalidad, se desarrolló un *endpoint* `admin-stats` encargado de generar estadísticas relevantes a partir de las tablas principales en la base de datos. El objetivo

es analizar la evolución del juego en términos de crecimiento de usuarios y su actividad general. A continuación, se describen las principales consultas y sus propósitos:

- **Crecimiento mensual de la base de jugadores:** Mediante una consulta de agrupación por mes, se cuenta el número de jugadores nuevos registrados cada mes en el último año.
- **Actividad general (número de sesiones de juego):** Se calcula el total de sesiones de juego mensuales, reflejando la actividad global de los usuarios.
- **Duración promedio de las sesiones:** Una consulta que calcula el tiempo promedio de duración de las sesiones por mes, proporcionando datos sobre el tiempo de interacción de los jugadores.
- **Tendencias de niveles jugados y resultados de juego:** Identifica los niveles más populares y calcula la tasa de éxito o "*win rate*" en cada uno de ellos, dando una visión de los niveles más jugados y de los resultados de los usuarios.
- **Frecuencia de uso de elementos del juego:** Recopila datos sobre la frecuencia de uso de elementos clave del juego, tales como "Barringtonia", "*Spaghetti*", entre otros, lo cual permite analizar las preferencias de los jugadores.

Este conjunto de estadísticas proporciona una visión detallada y actualizada de la actividad en la plataforma, permitiendo que los administradores analicen y respondan a las tendencias observadas.

```
// Endpoint para obtener estadísticas del último mes
adminsRouter.get("/stats", authenticateToken, isAdmin, async (req, res) => {
  try {
    // Consulta 1: Crecimiento mensual de la base de jugadores en el último año
    const newPlayersQuery = `
      SELECT DATE_TRUNC('month', login_date) AS month, COUNT(*) AS new_players
      FROM player
      WHERE login_date >= CURRENT_DATE - INTERVAL '1 year'
      GROUP BY month
      ORDER BY month;
      `;
    const newPlayersResult = await clientDB.query(newPlayersQuery);
    const newPlayers = newPlayersResult.rows;

    // Consulta 2: Actividad general (número de sesiones de juego) en el último año por mes
    const totalSessionsQuery = `
        SELECT DATE_TRUNC('month', date) AS month, COUNT(*) AS total_sessions
        FROM game_sessions
        WHERE date >= CURRENT_DATE - INTERVAL '1 year'
        GROUP BY month
        ORDER BY month;
        `;
    const totalSessionsResult = await clientDB.query(totalSessionsQuery);
    const totalSessions = totalSessionsResult.rows;

    // Consulta 3: Duración promedio de las sesiones en el último año por mes
    const avgDurationQuery = `
      SELECT DATE_TRUNC('month', date) AS month, AVG(duration_level) AS avg_duration
      FROM game_sessions
      WHERE date >= CURRENT_DATE - INTERVAL '1 year'
      GROUP BY month
      ORDER BY month;
      `;

    const avgDurationResult = await clientDB.query(avgDurationQuery);
    const avgDuration = avgDurationResult.rows;
```

Figura 7.144: *Enpoint* de estadísticas administrativas

El segundo objetivo consistió en la implementación de un sistema de notificaciones promocionales por correo electrónico, con el siguiente objetivo específico:

Utilizar tecnologías web modernas para mejorar la promoción y mantener informados a los jugadores sobre actualizaciones y eventos relevantes del juego.

Para lograr esto, se implementó un *endpoint* que utiliza la API de *Resend* para enviar correos electrónicos de promoción de manera automatizada. Estos correos son enviados desde un dominio personalizado (`platyfa-game.com`), adquirido mediante *NameCheap*, y su formato de contenido incluye un diseño personalizado en HTML y CSS incrustado para reflejar la identidad visual de la página y el juego. El proceso para el envío de correos promocionales se resume a continuación:

- **Recuperación de correos:** Se consulta la tabla `Players` para obtener los correos electrónicos de todos los usuarios activos (excluyendo cuentas eliminadas).
- **Contenido del correo:** El contenido incluye el título de la promoción, el nombre del autor y una descripción de la notificación.
- **Envío secuencial:** Los correos se envían de manera individual a cada usuario con un intervalo de 2 segundos para asegurar una entrega controlada y evitar restricciones del proveedor de correo.

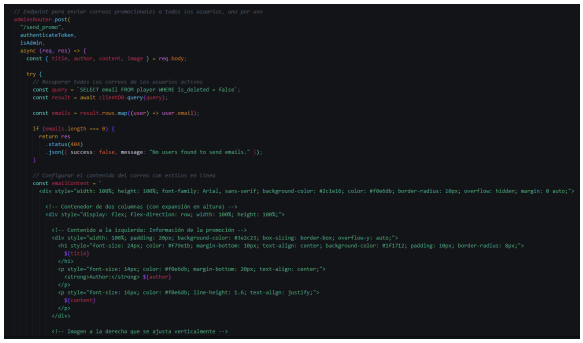

Figura 7.145: *Enpoint* para envío de notificaciones sobre promoción o evento

7.25. Fase de desarrollo: Cuarto prototipo programado (*Frontend*)

En esta fase se realizaron modificaciones enfocadas principalmente en mejorar la funcionalidad del sistema y resolver problemas de usabilidad identificados en la tercera fase de *testing*. A continuación, se detallan los cambios implementados, tanto para la experiencia de usuario general como para la parte administrativa.

7.25.1. Mejoras en la experiencia del usuario

Con base en el feedback recibido, se identificaron tres problemas principales en la interfaz de usuario:

1. **Persistencia de funcionalidades con token expirado:** Algunos usuarios reportaron que al volver a ingresar a la página, seguían viendo funcionalidades que deberían estar accesibles solo con un token válido. Para resolver este problema, se implementó una verificación para asegurarse de que el token no esté expirado al momento de cargar el sitio. Utilizando el siguiente código, se comparó la fecha de expiración del token con la fecha actual:

```
// Verificar si el token ha expirado
const currentTime = Date.now() / 1000; // Tiempo actual en segundos
if (decodedToken.exp < currentTime) {
  // Token ha expirado
  handleLogout();
  return;
}
```

Figura 7.146: Verificación de token válido

Si el token ha expirado, se llama a la función *handleLogout()* para cerrar la sesión del usuario, garantizando así que solo los usuarios con tokens válidos puedan ver las secciones protegidas.

2. **Formato de fecha en noticias:** Los usuarios también sugirieron mejorar la visualización de la fecha en la sección de noticias, ya que se mostraba en un formato poco amigable (e.g., 2024-10-29T00:00:00.000Z). Para mejorar la legibilidad, se implementó un formateo de fecha utilizando la función `Date` de JavaScript, con el siguiente código:

Figura 7.147: Formateo de fecha en noticias

Este cambio permite que las fechas se muestren en un formato más claro y comprensible para el usuario, mejorando la estética y la accesibilidad de la información.

3. **Actualización de perfil sin cierre de sesión innecesario:** Otro inconveniente reportado fue el cierre de sesión automático al actualizar cualquier información del perfil. Para mejorar esta experiencia, se implementó una verificación de cambios sensibles. Ahora, solo cuando se modifican campos críticos como la contraseña, el correo electrónico o el nombre de usuario, se elimina el token y se redirige al usuario al *login*. En caso de modificar otros campos no sensibles, como la foto de perfil o el país, simplemente se recarga la página. El siguiente código muestra cómo se detectan los cambios sensibles:

```
const sensitiveFieldsChanged =
  formData.username !== originalData.player.username ||
  formData.email !== originalData.player.email ||
  (formData.password && formData.password !== originalData.player.password);

// Enviar la solicitud para actualizar el perfil
const updateResponse = await fetch(`https://megaproyecto-backend-services.vercel.app/api/players/update_profile`, {
  method: 'PUT',
  headers: {
    'Content-Type': 'application/json',
    Authorization: `Bearer ${token}`,
  },
  body: JSON.stringify({
    username: formData.username,
    email: formData.email,
    country: formData.country,
    password: formData.password ? formData.password : undefined,
    profileImage: formData.profileImage, // Enviar la URL de la imagen al backend
  }),
});

const updateData = await updateResponse.json();
if (updateResponse.ok) {
  alert('Profile updated successfully!');

  if (sensitiveFieldsChanged) {
    localStorage.removeItem('token'); // Remover el token antiguo solo si los campos sensibles cambiaron
    window.location.href = '/login'; // Redirigir a login
  } else {
    window.location.reload(); // Recargar la página si no se cambian los campos sensibles
  }
} else {
  setError(updateData.message || 'Failed to update profile');
}
```

Figura 7.148: Detección de cambios sensibles en el perfil

7.25.2. Implementaciones para la administración

Para la parte administrativa, se añadieron dos nuevas funcionalidades que expanden las capacidades de gestión para los administradores.

1. **Sección de estadísticas administrativas:** Se agregó una nueva pestaña en el *Header* llamada *Admin Stats*, que permite a los administradores acceder a estadísticas clave de la plataforma. Esta sección incluye gráficos sobre el crecimiento de jugadores, el número de sesiones y otras métricas del último mes, obtenidas mediante el *endpoint* */stats*. A continuación se muestra una captura de esta sección en el *Header*:

Figura 7.149: Pestaña de *Admin Stats* en el *Header*

En la página ***Admin Stats***, cada gráfica se presenta en su propia tarjeta para una mejor organización visual:

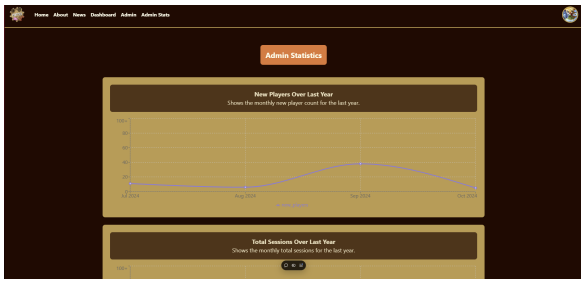

Figura 7.150: Sección de estadísticas administrativas

2. **Envío de correos promocionales:** En el ***Admin Panel***, se añadió una funcionalidad que permite al administrador enviar correos masivos a los usuarios registrados para informar sobre eventos o promociones. Esta funcionalidad utiliza el *endpoint* ***send-promo***, y el administrador puede especificar un título, autor, contenido y una imagen que se incluirán en el correo. A continuación se muestra la interfaz para el envío de correos promocionales:

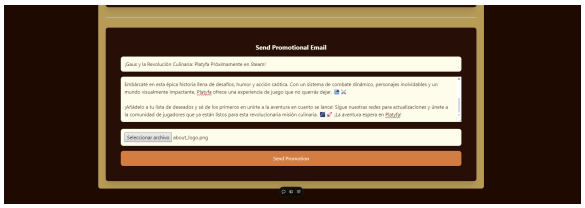

Figura 7.151: Panel de envío de correos promocionales

Para optimizar el almacenamiento, solo se envía la imagen sin almacenarla en el servidor, facilitando así la gestión de recursos.

7.26. Fase de *testing*: *Testing* final con usuarios

En esta fase de *testing*, se aplicó un enfoque similar al de la tercera fase de *testing* de usabilidad, utilizando las mismas cuatro preguntas clave para evaluar la experiencia del usuario:

1. ¿Qué tan fácil fue navegar por la página web?
2. ¿Consideras que la información está organizada de manera clara y comprensible?
3. ¿Cómo calificarías la apariencia del sitio en tu dispositivo móvil/tablet?
4. ¿Los botones y enlaces eran fáciles de hacer clic en la versión móvil/tablet?

Se definió y mantuvo la misma cantidad de participantes que en la tercera fase de *testing*, lo cual permite una evaluación consistente y comparable entre ambas etapas. Al igual que en la fase anterior, se evaluó el tiempo que cada participante requería para completar diversas tareas específicas dentro

de la página web, tanto en su versión de escritorio como en la versión móvil. Las tareas asignadas fueron diseñadas para abarcar cada sección de la web y proporcionar información detallada sobre la facilidad de uso y navegabilidad de cada una. Las tareas específicas fueron:

- Encontrar en una de las imágenes el título *game pitch*.
- Identificar quién es la persona que tiene el rol de *Animator 3D*.
- Ubicar el gráfico de partidas ganadas y perdidas.
- Encontrar la forma de registrarse en la página.
- Encontrar la forma de iniciar sesión en la página.
- Ubicar la opción para editar el perfil en la página.

Sin embargo, en esta fase se introdujo un nuevo elemento de análisis: la evaluación mediante un mapa de calor. Este enfoque adicional tuvo como objetivo verificar que el usuario visualizara la información relevante sin distracciones en otros elementos de la página, y con ello velar por el cumplimiento de la Ley de Hick, en dónde lo que se quiere lograr es que el usuario se enfoque solo en lo esencial. Para llevar a cabo esta evaluación, se utilizó un dispositivo llamado `Tobii`, el cual, mediante tecnología de infrarrojos, captura el movimiento de los ojos de los usuarios, permitiendo así generar un mapa de calor de las áreas más visualizadas en la página.

El uso de mapas de calor proporcionó una perspectiva adicional sobre la usabilidad de la página, al mostrar cuáles secciones captan la atención de los usuarios y cuáles podrían necesitar ajustes para mejorar la focalización de la información. Con estos datos, se puede mejorar la disposición de los elementos en la página y asegurar que la información clave sea fácilmente accesible para los usuarios.

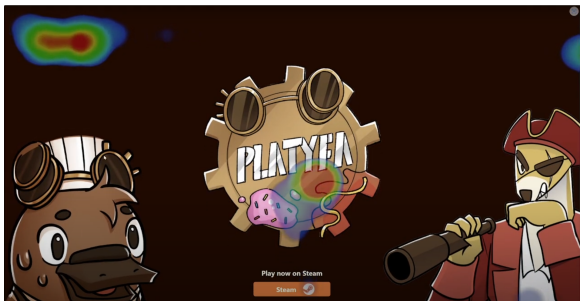

Figura 7.152: Mapa de calor visible en la página

Figura 7.153: Mapa de calor visible en la página versión móvil

7.27. Fase de *testing*: Test de *performance* a la API

En esta fase de *testing*, se llevó a cabo un test de *performance* de la API para evaluar su capacidad de respuesta bajo cargas masivas de usuarios simultáneos. El objetivo de este test fue analizar los tiempos de respuesta de la API y su estabilidad cuando se encuentra bajo estrés, simulando situaciones de uso intensivo.

Para realizar el test de *performance*, se utilizó la herramienta `k6` en un entorno Linux. Con `k6`, fue posible establecer un flujo controlado de usuarios concurrentes y definir distintas etapas de carga para simular picos de tráfico. A continuación, se muestra la configuración utilizada:

```
export let options = {
  stages: [
    { duration: "10s", target: 20 }, // Aumenta a 20 usuarios simultáneos en 10 segundos
    { duration: "1m", target: 50 },   // Mantiene 50 usuarios por 1 minuto
    { duration: "10s", target: 0 },   // Reduce a 0 usuarios
  ],
};
```

En este caso, la carga se incrementa a 20 usuarios simultáneos en los primeros 10 segundos, luego se incrementa a 50 usuarios por un periodo de 1 minuto, y finalmente se reduce a 0 usuarios. Este patrón de carga permitió observar cómo respondía la API en diferentes niveles de uso, desde un aumento gradual de usuarios hasta un periodo de alta demanda y, posteriormente, una reducción a un estado inactivo.

Se establecieron tareas específicas para cada uno de los *endpoints* de la API, incluyendo el registro de usuarios, el inicio de sesión, la actualización de datos de perfil, eliminar usuarios, el almacenamiento de los datos obtenidos en las sesiones de juego, y el acceso a las estadísticas administrativas.

Esto permitió evaluar el rendimiento de la API en distintas áreas de funcionalidad, verificando que cada *endpoint* pueda manejar cargas masivas de datos y solicitudes de manera eficiente.

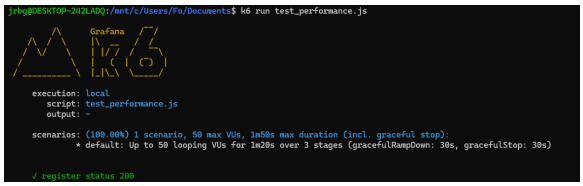

Figura 7.154: Visualización del test *performance* utilizando k6

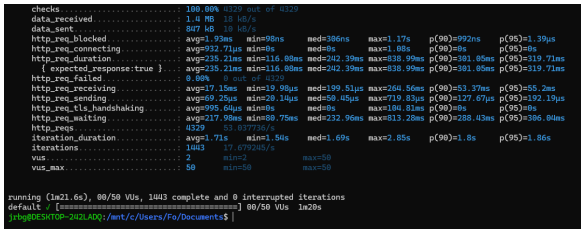

Figura 7.155: Visualización de como se brindan los resultados k6

7.28. Última fase: Implementación y desarrollo final

En esta fase final, se enfocó en la optimización de la API basada en los resultados obtenidos en el test de *performance*, realizado previamente para evaluar su rendimiento bajo cargas elevadas. Con esta optimización, el objetivo fue mejorar los tiempos de respuesta en los *endpoints* más críticos de la API, asegurando una experiencia de usuario fluida y sin interrupciones.

Para lograr esta optimización, se identificaron los *endpoints* que mostraron mayor demanda y fueron clave para el funcionamiento del sistema. Entre estos *endpoints* críticos se incluyeron `login`, `register`, `game_sessions`, `news`, `update_profile` y `game_statistics`. La optimización se realizó mediante la mejora de las consultas (*queries*) de la base de datos utilizadas en estos *endpoints*, buscando reducir el tiempo de ejecución y minimizar el consumo de recursos.

Una vez optimizados estos *endpoints*, los cambios finales se desplegaron en el entorno de producción. Esta implementación aseguró que la API estuviera lista para soportar el tráfico real de usuarios y manejar cargas masivas de manera eficiente. Con esta última fase completada, se dio por finalizado el desarrollo del proyecto, logrando un sistema robusto y preparado para el uso real, con un rendimiento optimizado y alineado a los objetivos de usabilidad y experiencia de usuario establecidos desde el inicio.

7.29. Fase final de *testing*: Test de *performance* final a la API

En esta fase, se realizó un test final de *performance* a la API, siguiendo el mismo procedimiento utilizado en la primera fase de pruebas de rendimiento. La diferencia clave en esta etapa fue que los *endpoints* ya han sido optimizados con base a los resultados y observaciones obtenidos en el test inicial. El objetivo de esta prueba fue verificar los tiempos de respuesta de cada uno de los *endpoints* tras las optimizaciones, asegurando que la API pudiera manejar cargas masivas de manera eficiente y mantener un rendimiento estable.

Para este test, se utilizó nuevamente la herramienta `k6` en un entorno Linux, configurando una carga de usuarios similar a la prueba inicial para obtener resultados comparables. Cada uno de los *endpoints* críticos, como ***`login`***, ***`register`***, ***`game_sessions`***, ***`news`***, ***`update_profile`***, ***`game_statistics`***, entre otros, fue evaluado individualmente para observar su rendimiento y verificar si las mejoras implementadas cumplían con los tiempos de respuesta esperados. Los resultados obtenidos proporcionaron una validación final de la eficiencia de la API bajo condiciones de uso intensivo, confirmando que las optimizaciones realizadas lograron reducir los tiempos de respuesta y mejorar la experiencia del usuario.

Figura 7.156: Visualización del test *performance* utilizando k6

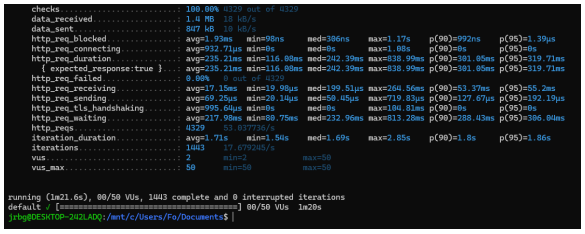

Figura 7.157: Visualización de como se brindan los resultados k6

7.30. Implementación del consentimiento activo para cumplimiento del GDPR

En el marco del desarrollo del proyecto web, se implementó una funcionalidad de **consentimiento activo** para cumplir con las regulaciones establecidas en el Reglamento General de Protección de Datos (GDPR, por sus siglas en inglés). Esta implementación asegura que los usuarios otorguen su consentimiento explícito antes de procesar su información personal, en conformidad con los principios de transparencia, legalidad y responsabilidad establecidos en el reglamento.

7.30.1. Identificación de la necesidad

El GDPR establece que todo procesamiento de datos personales debe estar basado en un fundamento legal, siendo el consentimiento informado uno de los más comunes en aplicaciones web. Dado que el proyecto requiere el registro de datos como nombre, correo electrónico y país, era necesario implementar un mecanismo que garantizara el consentimiento explícito del usuario antes de almacenar cualquier información.

7.30.2. Definición de los requisitos

Los requisitos específicos para la implementación del consentimiento activo fueron:

- Un ***checkbox* no preseleccionado** para que el usuario pueda aceptar la política de privacidad.
- Un **mensaje claro y explícito** indicando que al registrarse el usuario acepta la política de privacidad, con un enlace directo a la misma.
- La **validación obligatoria** del consentimiento antes de permitir el envío del formulario de registro.

7.30.3. Desarrollo e implementación técnica

Para la implementación técnica, se realizaron las siguientes modificaciones en el componente de registro (`Register.jsx`):

- Se añadió un estado de React (*`useState`*) para gestionar si el usuario ha marcado la casilla de consentimiento.
- Se implementó un input checkbox no preseleccionado, asociado al estado de consentimiento.
- En la función *`handleRegister`*, se incluyó una condición que evita el envío del formulario si el consentimiento no ha sido aceptado, mostrando una notificación de error en caso de incumplimiento.
- Se agregó un enlace a la página de política de privacidad (`/privacy-policy`).

El siguiente fragmento de código ejemplifica la implementación descrita:

```
const [consent, setConsent] = useState(false);

const handleRegister = async (e) => {
    e.preventDefault();
    if (!consent) {
        setNotification({
            message: 'You must accept the Privacy Policy to register.',
            type: 'error',
        });
        return;
    }
    // Continuar con el registro...
};

<label className="flex items-center space-x-2">
    <input
        type="checkbox"
        checked={consent}
        onChange={(e) => setConsent(e.target.checked)}
        className="form-checkbox h-5 w-5 text-orangePlatyfa"
    />
    <span>
        By registering, you agree to our{' '}
        <Link to="/privacy-policy" className="underline text-orangePlatyfa">
            Privacy Policy
        </Link>.
    </span>
</label>
```

7.30.4. Validación y control

Para garantizar el correcto funcionamiento de la funcionalidad implementada, se realizaron las siguientes pruebas de validación:

- **Intento de registro sin marcar la casilla de consentimiento:** El sistema bloquea el registro y muestra un mensaje de error.
- **Intento de registro con consentimiento marcado:** El sistema permite el envío del formulario y registra al usuario correctamente.
- **Visualización del enlace a la política de privacidad:** El enlace redirige correctamente a la página de términos y condiciones.

7.30.5. Cumplimiento con el GDPR

Esta implementación asegura el cumplimiento con los siguientes principios del GDPR:

- **Consentimiento activo:** El usuario debe realizar una acción afirmativa para aceptar la política.
- **Claridad y transparencia:** Se proporciona un mensaje claro y un enlace visible a la política de privacidad.
- **Minimización de datos:** Solo se recopilan los datos estrictamente necesarios para el registro.

7.31. Implementación del nuevo estilo de *Dashboard*

El nuevo estilo del Dashboard fue diseñado y desarrollado con el objetivo de mejorar la presentación de estadísticas del usuario, enfocándose en una interfaz clara, organizada y funcional para la experiencia de visualización de datos. La implementación se centró en un diseño de tarjetas (*cards*) con gráficos distribuidos uniformemente para asegurar la legibilidad y facilitar el análisis visual de datos clave.

7.31.1. Objetivo del cambio

El rediseño del *Dashboard* se implementó para:

- **Mejorar la experiencia del usuario (UX):** Facilitar la comprensión rápida de estadísticas importantes mediante un diseño visualmente limpio y centrado en gráficos.
- **Claridad y consistencia:** Todas las tarjetas gráficas se redimensionaron para ser uniformes en tamaño, permitiendo una distribución simétrica y balanceada en la pantalla.
- **Accesibilidad:** Se utilizaron colores contrastantes con un fondo oscuro y tonalidades neutras para mantener la accesibilidad visual y evitar confusión al interpretar los gráficos.

7.31.2. Elementos implementados

1. Diseño de tarjetas (*cards*) uniformes

- Se utilizaron tarjetas con un tamaño uniforme para cada estadística.
- Cada tarjeta contiene un título, una breve descripción y un gráfico representativo del dato mostrado.
- El encabezado de cada tarjeta utiliza un color ligeramente más claro para destacar la información sin perder contraste.

2. Gráficos y visualización de datos

- Los gráficos fueron implementados utilizando la biblioteca Recharts.
- Cada gráfica utiliza una paleta de colores consistente (`rgba(37, 9, 0, 0.7)`) para mantener uniformidad visual.
- **Tipos de gráficos usados:**
  - Gráfica de líneas para representar estadísticas con tendencia en el tiempo.
  - Gráfica de área para valores acumulativos o progresivos.
  - Gráfica de barras para comparar valores individuales.
  - Gráfica de radar para distribuciones circulares de datos.
  - Gráfica de barras apiladas para comparación de dos conjuntos de datos.

3. Encabezado y titulación

- Un encabezado destacado con el nombre del usuario actual y el título *"Dashboard"* centrado en la parte superior.
- El nombre de usuario se obtiene dinámicamente del token almacenado localmente.

4. Distribución y responsividad

- La distribución se organizó en una cuadrícula flexible (`grid grid-cols-1 sm:grid-cols-2 lg:grid-cols-3`) que ajusta la cantidad de columnas según el tamaño de la pantalla.
- Esto asegura una experiencia de usuario óptima tanto en dispositivos móviles como de escritorio.

5. *Footer* consistente

- Se integró un *footer* uniforme al final de la página con información sobre derechos de autor y un enlace directo a la política de privacidad.

7.31.3. Justificación de diseño

Este estilo de *Dashboard* fue seleccionado y desarrollado con base en los siguientes criterios:

- **Claridad visual:** El uso de gráficos y la disposición uniforme de las tarjetas permiten al usuario interpretar los datos de forma rápida y efectiva.
- **Consistencia visual:** Se implementó una paleta de colores homogénea con tonos marrones y beige que se alinean con la identidad visual del proyecto.
- **Jerarquía de información:** La disposición en cuadrícula y los títulos destacados guían al usuario a priorizar la información más importante.
- **Accesibilidad:** Se optaron por contrastes adecuados y tamaños de fuente legibles para garantizar una experiencia inclusiva para todo tipo de usuarios.
- **Experiencia de usuario mejorada:** El diseño minimalista y la eliminación del desplazamiento vertical excesivo mejoran la experiencia general al acceder al *Dashboard*.

Figura 7.158: Visualización del nuevo *dashboard* implementado

7.32. Pruebas de usabilidad y ajuste de dificultad

Para evaluar la experiencia del jugador y realizar ajustes en la dificultad y jugabilidad, se aplicaron dos encuestas a los usuarios: una antes de la sesión de juego y otra al finalizarla. Las preguntas a continuación detallan los aspectos evaluados en cada fase.

7.32.1. Encuesta previa a la sesión de juego

Esta encuesta se aplicó antes de la prueba para obtener información sobre el perfil del jugador y sus expectativas.

1. ¿Está de acuerdo en participar en la investigación?
   a) Sí

b) No

2. EVENTO
    a) Game Showcase - Gamedev Quest
    b) Playtesting 28 / 10
3. Nombre y apellido
4. Carrera
5. ¿Has participado antes en una sesión de *playtesting* de videojuegos?
    a) Sí
    b) No
6. ¿Tienes experiencia previa jugando juegos de tipo shooter 3D?
    a) Sí
    b) No
7. Si la respuesta a la pregunta anterior fue sí, ¿puedes mencionar alguno?
8. ¿Qué esperas encontrar en un juego con un protagonista como un ornitorrinco chef e inventor?
9. ¿Qué aspectos de los juegos valoras más?
    a) Jugabilidad
    b) Historia
    c) Gráficos
    d) Controles
    e) Diseño de personajes
    f) Otra...
10. ¿Cuánto tiempo dedicas generalmente a jugar videojuegos por semana?
    a) Menos de una hora.
    b) Entre 1 a 3 horas.
    c) Entre 3 a 5 horas.
    d) Entre 5 a 10 horas.
    e) Entre 10 a 20 horas.
    f) Más de 20 horas.
11. ¿Qué es lo que más te atrae cuando decides probar un nuevo juego?
12. ¿Qué juegos has jugado recientemente que te impresionaron por su modelado 3D?
13. ¿Tienes experiencia previa en el modelado 3D o el desarrollo de videojuegos?
    a) Sí
    b) No
14. Si respondiste "sí", ¿qué herramientas o software has utilizado?

7.32.2. Encuesta posterior a la sesión de juego

Esta segunda encuesta fue aplicada al finalizar la sesión, con el objetivo de obtener retroalimentación sobre la experiencia de juego y la interacción con los enemigos.

Sección 1 de 4: Preguntas - Experiencia

1. En una escala de 1 a 5, ¿cuánta diversión tuvo probando el juego?

1	2	3	4	5
○	○	○	○	○
Poca diversión				Mucha diversión

2. ¿Con qué controles probó el juego?
   a) Teclado
   b) Control
   c) Ambas
3. ¿Los controles del juego fueron comprensibles?
   a) Sí
   b) No
4. ¿Cree que todos los íconos e instrucciones dentro del juego son claros?
   a) Sí
   b) No
   c) La mayoría
   d) Muy pocas
5. ¿Qué tan satisfactorio es el feedback dentro del juego?
6. ¿Tuvo algún problema navegando entre las distintas pantallas del juego?
   a) Sí
   b) No
7. ¿Considera que la estética visual del juego ayuda a la inmersión de la historia?
   a) Sí
   b) No
   c) Parcialmente
8. ¿En algún momento hubo algo que quería hacer pero no pudo?
   a) Sí
   b) No
9. Si la respuesta anterior fue sí, ¿qué fue?
10. ¿Qué cambios le haría al juego?

Esta sección busca evaluar la experiencia general del usuario y su percepción de los controles y la estética del juego. Las preguntas están diseñadas para medir la diversión, comprensión de instrucciones, facilidad de navegación y la calidad visual, lo que es fundamental para establecer una base sólida de jugabilidad y asegurar que los aspectos técnicos básicos no interfieran con la experiencia.

Sección 2 de 4: Preguntas - Gameplay

1. ¿Cree que el nivel de dificultad indicado es apropiado para un juego de este estilo?
    a) Sí, me parece perfecto
    b) No, debería de ser más fácil
    c) No, debería de ser más difícil
2. ¿En qué dificultad sintió a los enemigos dentro del nivel?
    a) Fácil
    b) Intermedio
    c) Difícil
3. ¿La velocidad de ataque de los enemigos es apropiada?
    a) Sí
    b) No
4. Justifique.
5. Con respecto al comportamiento de los enemigos, ¿cambiaría algo de ellos?
    a) Sí
    b) No
6. Justifique.
7. ¿Durante el periodo de prueba, encontró algún bug con respecto a los enemigos?
    a) Sí
    b) No
8. Si su respuesta anterior fue sí, por favor explique la situación en la que encontró el bug.
9. ¿Qué le pareció la cantidad de enemigos en el nivel?
    a) Muy pocos, deberían de haber más
    b) Adecuada
    c) Demasiados, deberían de haber menos

La sección de Gameplay se enfoca en la dificultad, comportamiento y cantidad de enemigos, elementos directamente relacionados con el desarrollo de los enemigos en el juego. Las preguntas sobre dificultad y velocidad de ataque permiten evaluar si los enemigos ofrecen un desafío equilibrado y adaptativo. Además, las preguntas sobre el comportamiento de los enemigos y la cantidad en el nivel permiten ajustar las mecánicas para que sean satisfactorias y coherentes con las expectativas del jugador. Esta información es crucial para refinar el diseño de los enemigos, asegurando que cada encuentro sea desafiante y estimulante.

Sección 3 de 4: Preguntas - Modelos 3D y animación

1. ¿Qué opina sobre la estética del juego?
2. En una escala de 1 a 5, ¿los modelos 3D son visualmente atractivos y están bien diseñados?

1	2	3	4	5
◯	◯	◯	◯	◯
Poco atractivos				Muy atractivos

3. ¿Cree que el modelado 3D mejoró su experiencia de juego?
   a) Sí
   b) No
4. Si respondió "sí", ¿puede proporcionar ejemplos específicos de cómo el modelado 3D influyó en su experiencia?
5. ¿Cree que el modelado 3D aporta un impacto significativo a la historia y experiencia del juego, en comparación con un posible diseño en 2D?
   a) Sí
   b) No
6. Justifique.
7. En una escala de 1 a 10, ¿el modelado 3D aumentó su nivel de inmersión en el juego?

1	2	3	4	5	6	7	8	9	10
◯	◯	◯	◯	◯	◯	◯	◯	◯	◯
Muy poco									Mucho

8. Justifique.
9. En una escala del 1 al 10, ¿los personajes y objetos se alinearon con lo que imaginaba a partir de la cinemática y la introducción?

1	2	3	4	5	6	7	8	9	10
◯	◯	◯	◯	◯	◯	◯	◯	◯	◯
Muy en desacuerdo									Muy de acuerdo

10. En una escala del 1 al 10, ¿qué tan inmersiva le pareció la experiencia visual del juego gracias a las animaciones de los personajes?

1	2	3	4	5	6	7	8	9	10
◯	◯	◯	◯	◯	◯	◯	◯	◯	◯
Poco inmersiva									Muy inmersiva

11. ¿Cree que las animaciones de los personajes lograron transmitir la narrativa de "Gaus y la revolución de los ornitorrincos¿
    a) Sí
    b) No
12. ¿Se sintió más conectado o empático hacia los personajes debido a sus animaciones?
    a) Sí
    b) No

13. En una escala del 1 al 5, ¿considera que las animaciones fueron fluidas y se adaptaron adecuadamente al ritmo del juego?

1	2	3	4	5
○	○	○	○	○
No se adaptaron				Sí se adaptaron

14. ¿Logró identificar los distintos estados de animación de los personajes, como correr, caminar, saltar, etc.?

    a) Sí

    b) No

15. En una escala del 1 al 10, ¿qué tan satisfactorias le parecieron las transiciones entre las animaciones (por ejemplo, de caminar a correr)?

1	2	3	4	5	6	7	8	9	10
○	○	○	○	○	○	○	○	○	○
Insatisfactorio									Satisfactorio

16. ¿Notó la aplicación de los principios de animación como anticipación y encogerse en los movimientos de los personajes?

    a) Sí

    b) No

17. ¿Percibió que las animaciones del juego contribuían al atractivo visual general?

1	2	3	4	5	6	7	8	9	10
○	○	○	○	○	○	○	○	○	○
Contribuyen									No contribuyen

18. En una escala del 1 al 5, ¿considera que las animaciones mejoran su comprensión de la trama y la lucha entre los ornitorrincos y sus depredadores?

1	2	3	4	5
○	○	○	○	○
No ayudan				Sí ayudan

19. En una escala del 1 al 5, ¿las animaciones le parecieron realistas o creíbles dentro del contexto caricaturesco y de ciencia ficción del juego?

1	2	3	4	5
○	○	○	○	○
Poco creíbles				Apropiadas

En esta sección, las preguntas se centran en cómo los modelos 3D y las animaciones contribuyen a la inmersión y narrativa. Evaluar la estética, el impacto visual y la calidad de las animaciones permite entender si los diseños apoyan la conexión emocional y comprensión de la historia. Esto es importante en juegos donde la narrativa y la atmósfera visual enriquecen la experiencia del jugador.

Sección 4 de 4: Agradecimiento

1. Comentarios adicionales.

Finalmente, esta sección invita a los usuarios a brindar comentarios adicionales. Esta retroalimentación abierta permite capturar detalles o sugerencias no cubiertos en las preguntas anteriores, brindando al equipo de desarrollo una perspectiva más completa de la experiencia del jugador.

Todas estas preguntas ayudaron a obtener una visión detallada de la experiencia del usuario y a realizar ajustes en la dificultad y el diseño visual del juego para mejorar la jugabilidad y la inmersión, asegurando así una experiencia de juego gratificante y adaptativa.

7.33. Recopilación e implementación de retroalimentación

Una de las partes más importantes del proceso de creación de un videojuego el el obtener retroalimentación del público objetivo, especialmente si se está teniendo un acercamiento centrado en el jugador. Por ello es necesario escuchar comentarios sobre jugadores que no se encuentran familiarizados con el juego para poder obtener información sobre primeras impresiones, qué tan intuitivos son los controles y sistemas del juego, y qué tan claras son las instrucciones que se le brindan al jugador. Para ello se hacen sesiones de *playtesting* las cuales tienen el objetivo de contestar las preguntas anteriores y observar si los sentimientos objetivos del juego están siendo vividos por los jugadores. Para ello se realizaron varias sesiones de *playtesting* donde los jugadores probaron versiones específicas del juego para definir los aspectos necesarios a cambiar y la prioridad de estos cambios. Además, se hizo uso de 2 formularios de Google. El primer formulario tiene el objetivo de obtener más información sobre los jugadores, el tipo de jugador que son y su experiencia con juegos de tipo *shooter*. El segundo formulario tiene el objetivo de obtener comentarios sobre el *gamplay* y sus opiniones sobre los controles, el aspecto visual del juego y el nivel de dificultad.

7.33.1. Preguntas para la retroalimentación

Para las 2 encuestas se realizaron preguntas con el objetivo de determinar el éxito o el fallo de ciertos elementos dentro del juego. Las preguntas realizadas son las siguientes.

En una escala de 1 a 5 ¿cuánta diversión tuvo probando el juego? Para esta pregunta se busca determinar si efectivamente el juego se puede considerar como una experiencia placentera y divertida.

¿Con qué controles probó el juego? Esta nos permite obtener 2 datos. Cuál es la preferencia de controles para el juego por parte de los jugadores, y nos permite asociar comentarios solicitados luego en la encuesta con el modo de juego de usuario. Por ejemplo, si un usuario dice que tuvo problemas comprendiendo los controles, podemos determinar si se jugó con un control o con un teclado.

¿Los controles del juego fueron comprensibles? Esta pregunta busca comprender si los jugadores consideran que los controles son intuitivos y si tienen sentido dentro del contexto del juego.

¿Tuvo algún problema navegando entre las distintas pantallas del juego? Asociada a la experiencia de usuario, esta nos permite comprender si la interfaz del videojuego es intuitiva y si el orden de los elementos es comprensible.

¿Considera que la estética visual del juego ayuda a la inmersión de la historia? Esta pregunta nos responde qué tan integrada se encuentra el contexto de la historia dentro de las

mecánicas del juego, y cómo la estética visual se responsabiliza de ello.

En algún momento hubo algo que quería hacer pero no pudo? Una pregunta típica de las sesiones de *playtesting*, seguida de una pregunta para justificarla, nos ayuda a determinar puntos de frustración para los jugadores. También es posible determinar si el jugador comprendió correctamente todas las instrucciones dentro del juego.

En general, ¿qué cambios le haría al juego? Ayuda a obtener nuevas ideas sobre el juego, o nuevamente determinar otros puntos de frustración y cómo los jugadores desearían que se arreglaran.

7.33.2. Primera sesión de *playtesting*

El primer *playtesting* oficial del juego ocurrió a finales del mes de agosto. Para ello se utilizaron las computadoras del laboratorio de UI/UX del edificio CIT de la Universidad del Valle de Guatemala luego de establecer su disponibilidad. Para obtener jugadores se llamó a estudiantes conocidos con el horario disponible. Se charló con los estudiantes, y se les explicó un poco del proyecto y el fin de la sesión de *playtesting*. Antes de jugar se les requirió llenar el primer formulario. Durante su tiempo de juego se les explicaron las instrucciones, controles y el objetivo del juego debido a que las versiones testeadas no poseían ninguna de estas explícitamente en el juego. Se fueron anotando ciertas ideas sobre los comentarios que realizaban jugando. Y para finalizar la sesión se les requería llenar el segundo formulario. Este playtesting fue corto, con un número limitado de personas, pero fue muy enriquecedor para conocer otros puntos de vista.

7.33.3. Segunda sesión de *playtesting*

La segunda sesión oficial de playtesting ocurrió a finales del mes de septiembre, dentro de la celebración del décimo aniversario de la comunidad de GameDev GT, llevado a cabo dentro de las instalaciones de la Universidad del Valle de Guatemala. Para ello, se reservó espacio dentro de los puestos durante el *showcase* de juegos guatemaltecos. Se invitó a los visitantes del evento a probar el juego y dar su opinión. Nuevamente, se requirió llenar el primer formulario antes de jugar, y el segundo luego de la prueba. La versión de prueba no poseía las instrucciones ni los controles descritos explícitamente en el juego, por lo que estos fueron descritos mientras se jugaba. Se obtuvo un aproximado de 20 respuestas en los formularios y se obtuvo más *feedback* enriquecedor y distintos puntos de vista.

Figura 7.159: Puesto de prueba del juego dentro del evento de GameDev Quest

7.33.4. Tercera sesión de *playtesting*

Para la tercera sesión de *playtesting*, nuevamente se hizo uso del laboratorio de computadoras de UI/UX del edificio CIT de la Universidad del Valle de Guatemala. Se invitó a un grupo de conocidos para probar el juego y recibir su retroalimentación. Al igual que en las sesiones pasadas, se les requirió llenar un formulario antes y después de realizar pruebas sobre el juego. El objetivo principal de la tercera sesión era obtener comentarios sobre la implementación de elementos de otros módulos del proyecto.

CAPÍTULO 8

Resultados

8.1. *Gameplay* final

A continuación se pueden observar capturas de pantalla del *gameplay* final del juego de Platyfa, haciéndo especial énfasis en los elementos de la interfaz gráfica.

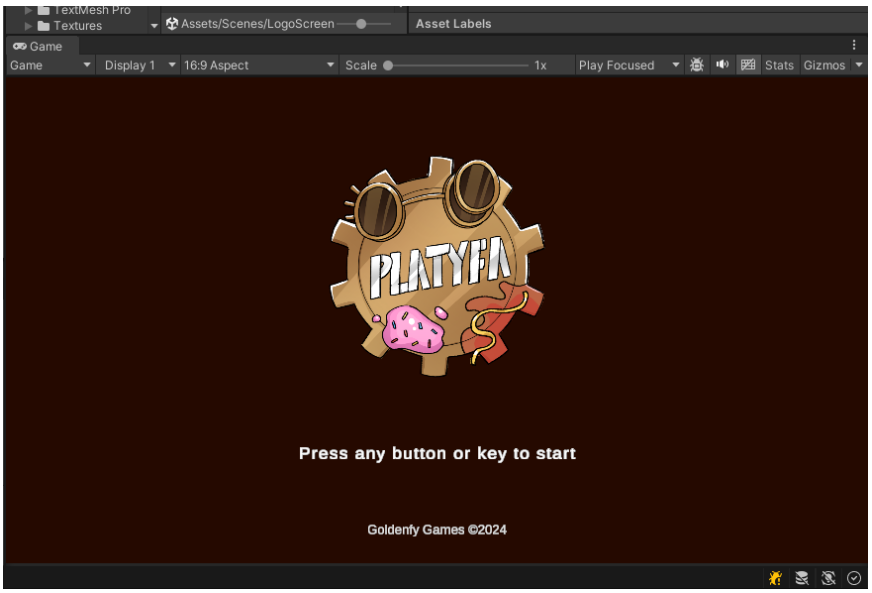

Figura 8.1: Pantalla de logo

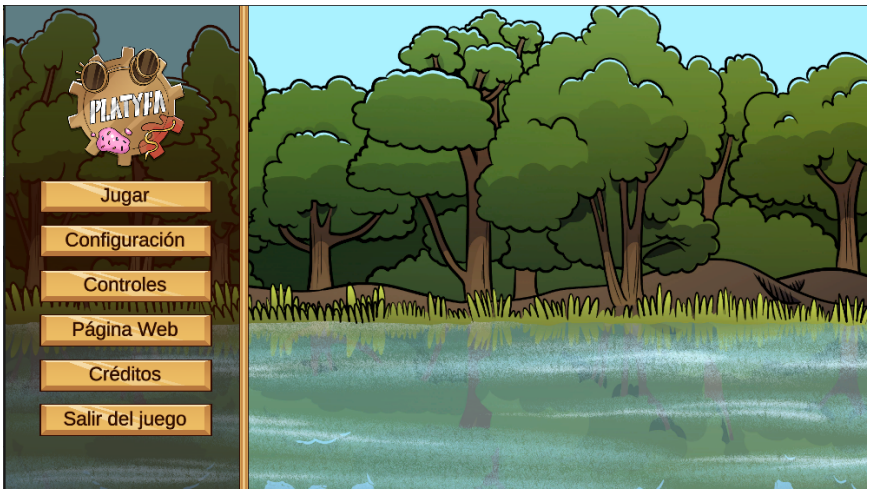

Figura 8.2: Pantalla de menú principal

Figura 8.3: Pantalla de selección de partirda

Figura 8.4: Pantalla de creación de cuenta

Figura 8.5: Pantalla de inicio de sesión

Figura 8.7: HUD principal

Figura 8.6: Pantalla de error al iniciar sesión

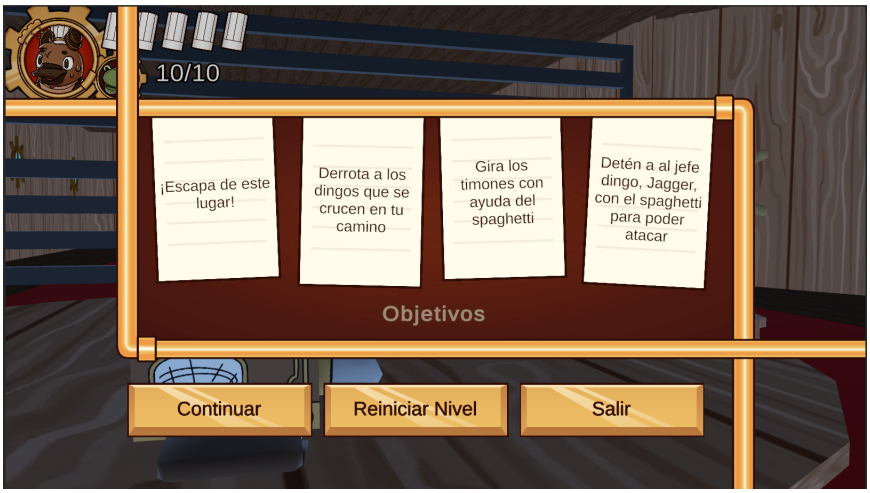

Figura 8.8: Pantalla de pausa

Figura 8.9: Panel de instrucciones sobre controles del juego al lado derecho

Figura 8.10: Barra de vida de los enemigos

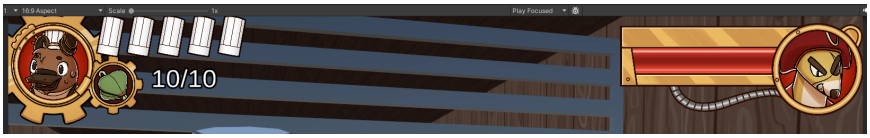

Figura 8.11: HUD superior cuando se está en una batalla contra un jefe

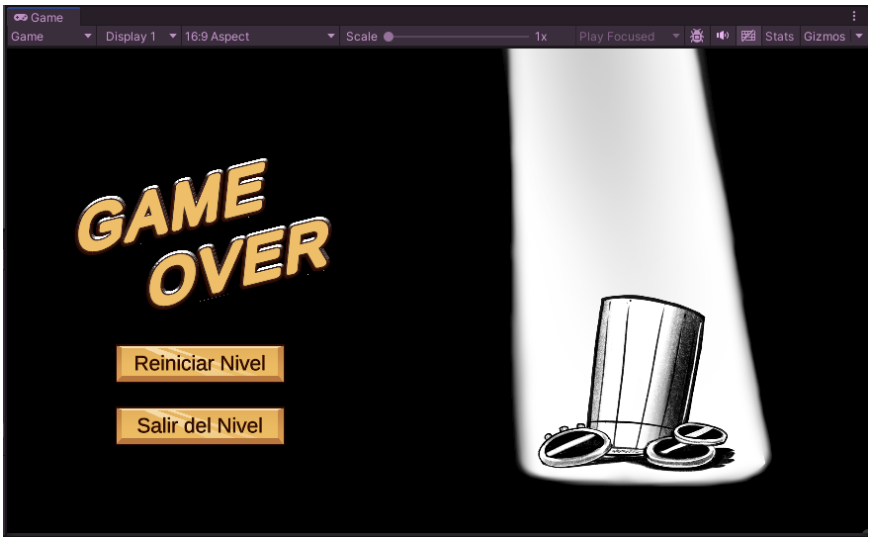

Figura 8.12: Pantalla de derrota

Figura 8.13: Pantalla de victoria

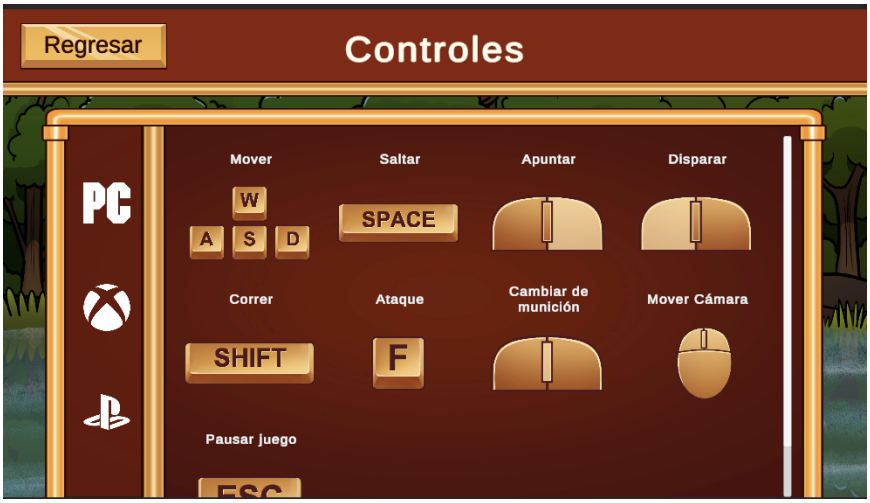

Figura 8.14: Pantalla de explicación de controles

Figura 8.15: Pantalla de configuración de settings

8.2. Dificultad de los enemigos

Una de las preguntas clave para evaluar la dificultad percibida fue:

1. ¿Cree que el nivel de dificultad indicado es apropiado para un juego de este estilo?
   a) Sí, me parece perfecto
   b) No, debería de ser más fácil
   c) No, debería de ser más difícil

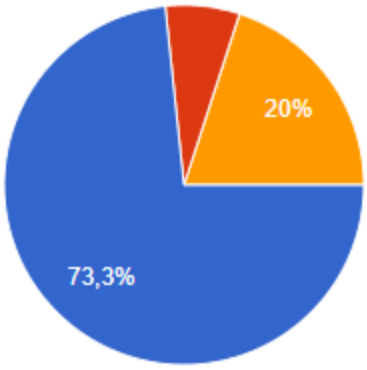

Figura 8.16: Respuestas de los jugadores respecto al nivel de dificultad de los enemigos

Los resultados mostraron que el 73.3 % de los 55 jugadores consideró que la dificultad fue adecuada, mientras que el 6.7 % sugirió que debería ser más fácil y el 20 % opinó que debería ser más difícil. Esto indica que el comportamiento desarrollado de los enemigos fue balanceado para la mayoría de los jugadores, pero algunos ajustes podrían mejorar la experiencia.

8.3. Velocidad y comportamiento de los enemigos

Para entender mejor el comportamiento de los enemigos, se incluyeron las siguientes preguntas relacionadas con la velocidad y la respuesta de los enemigos durante el combate:

1. ¿La velocidad de ataque de los enemigos es apropiada?
    a) Sí
    b) No

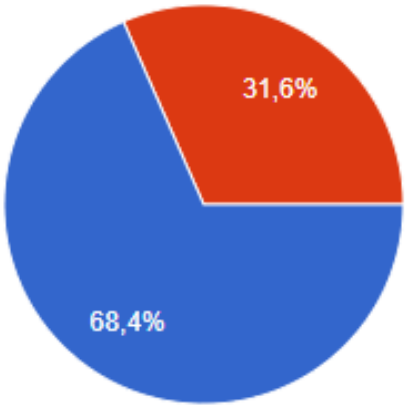

Figura 8.17: Respuestas de los jugadores respecto a la velocidad de ataque de los enemigos

2. Con respecto al comportamiento de los enemigos, ¿cambiaría algo de ellos?
    a) Sí
    b) No

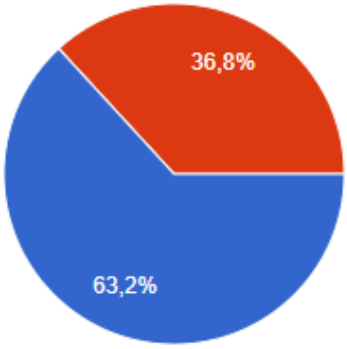

Figura 8.18: Respuestas de los jugadores respecto a si quisieran cambios en los enemigos

El 68.4 % de los 55 jugadores opinó que la velocidad de ataque fue adecuada, mientras que el 31.6 % sugirió que debería ajustarse. Además, el 63.2 % mencionó posibles cambios en el comportamiento de los enemigos, destacando aspectos como la dificultad de anticipar los movimientos y la falta de variación en las estrategias enemigas.

8.4. Bugs relacionados con los enemigos

Otro aspecto importante fue la detección de posibles errores (bugs) en el comportamiento de los enemigos. A través de la siguiente pregunta:

1. ¿Durante el periodo de prueba, encontró algún *bug* con respecto a los enemigos?
    a) Sí
    b) No

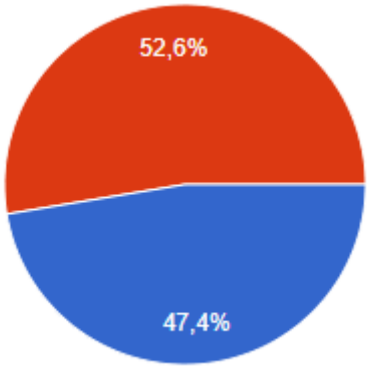

Figura 8.19: Respuestas de los jugadores respecto a si hubo *Bugs* con los enemigos

Casi la mitad de los 55 jugadores (47.4 %) de los participantes reportó haber encontrado bugs en el comportamiento de los enemigos, describiendo problemas como un error en el comportamiento del Boss, el cual no atacaba al pasar a su segunda fase hasta que se eliminaban los timones, o que en ciertos momentos los enemigos no se movían hasta que el jugador se acercaba a ellos.

8.5. Cantidad de enemigos

Para evaluar la percepción de la cantidad de enemigos en los niveles, se utilizó la siguiente pregunta:

1. ¿Qué le pareció la cantidad de enemigos en el nivel?
    a) Muy pocos, deberían de haber más
    b) Adecuada
    c) Demasiados, deberían de haber menos

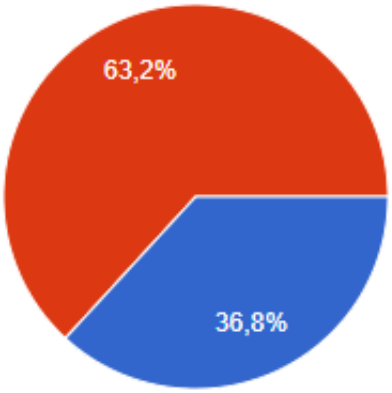

Figura 8.20: Respuestas de los jugadores respecto a si la cantidad de enemigos era adecuada para el nivel

Los resultados mostraron que el 63.2 % de los 55 jugadores encontró que la cantidad de enemigos fue adecuada, mientras que el 36.8 % sugirió que deberían incluirse más enemigos en el nivel.

8.6. Personajes 3D

8.6.1. Personaje principal: Gauss (Ornitorrinco)

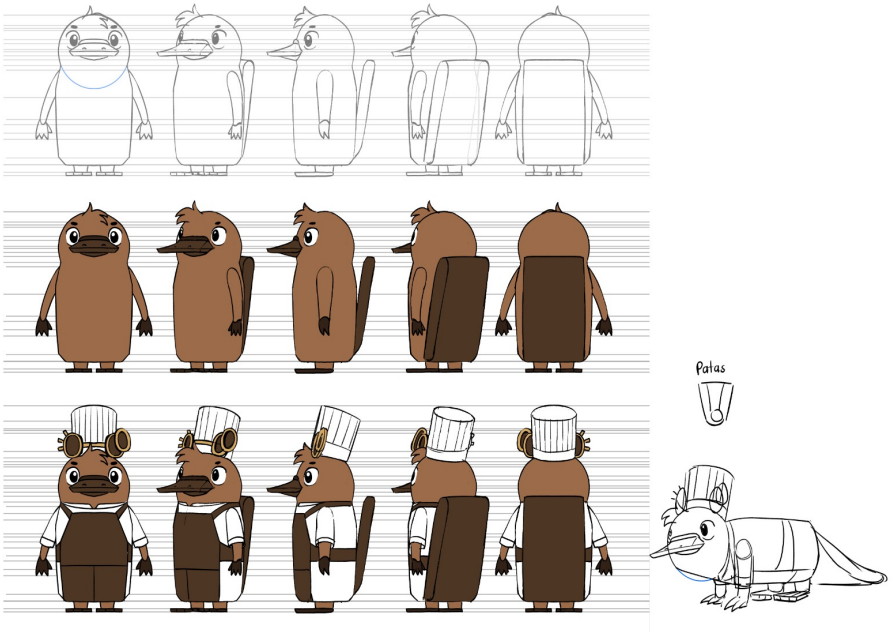

Figura 8.21: Imagen de referencia para Gauss

Figura 8.22: Gauss versión 1

La versión 1 de Gauss fue mi primer modelo y aún estaba entendiendo mi metodología regresando a ver este modelo no usé todas las técnicas y herramientas correctas como en los siguientes modelos que me di cuenta por la práctica. Y se notó la falta de experiencia ya que cuando se quiso utilizar este modelo para animaciones tenía muchas deformaciones extrañas, así que con mi equipo decidimos que lo volvería a hacer y es por eso que esta versión quedó descartada y la versión dos fue la que se usó en el videojuego.

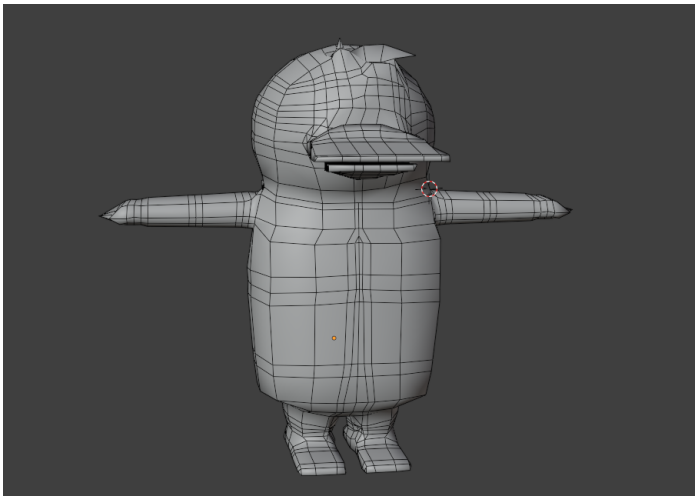

Figura 8.23: Gauss version 1

Figura 8.24: Vistas relevantes de Gauss v2

La versión 2 de Gauss fue el tercer modelo que hice después de Dingo secuaz, es por eso con un poco más de practica logre un personaje que cumplía con el arte conceptual y los más importante la geometría mejoro mucho, por lo que cuando lo pase al módulo de animación no me reportaron ningún problema.

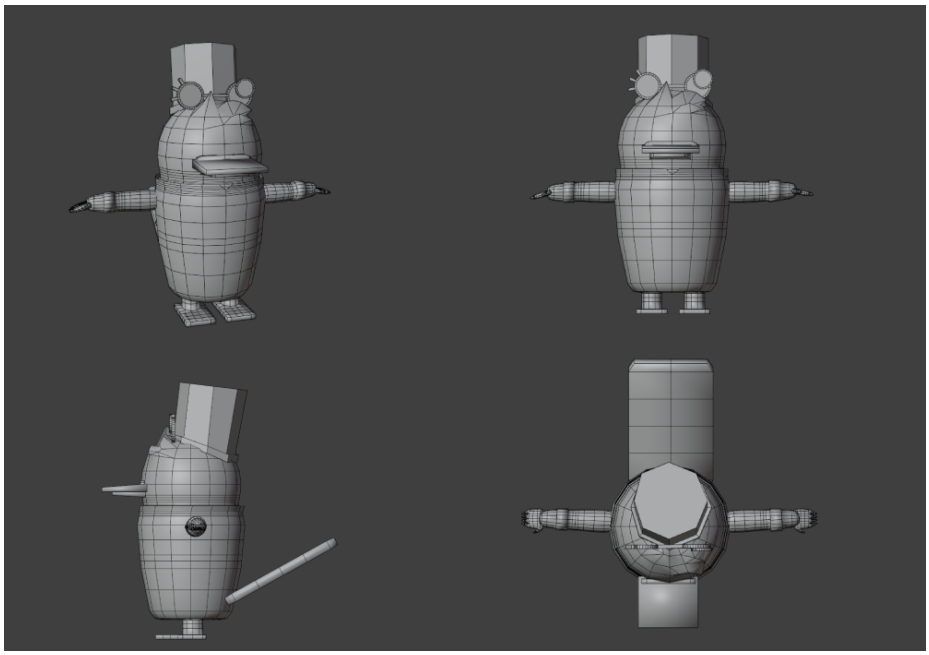

Figura 8.25: Vistas relevantes de la estructura geométrica de Gauss v2

8.6.2. Arma de Gauss

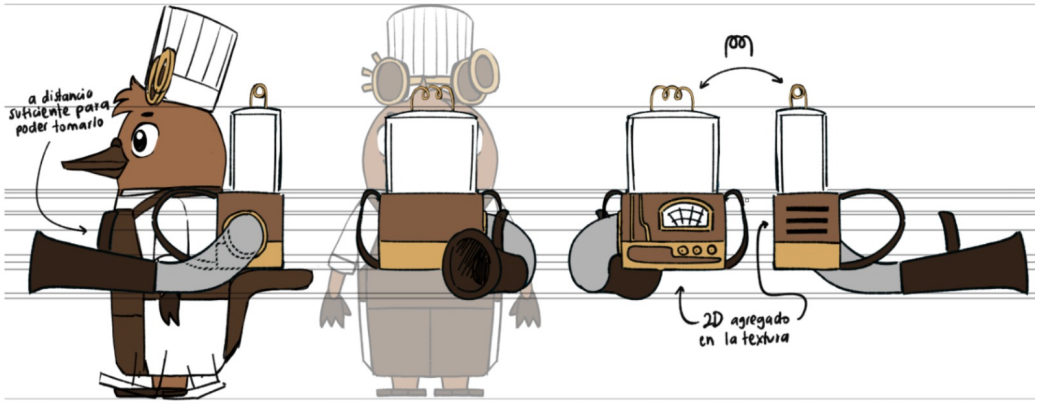

Figura 8.26: Imagen de referencia para el arma de Gauss

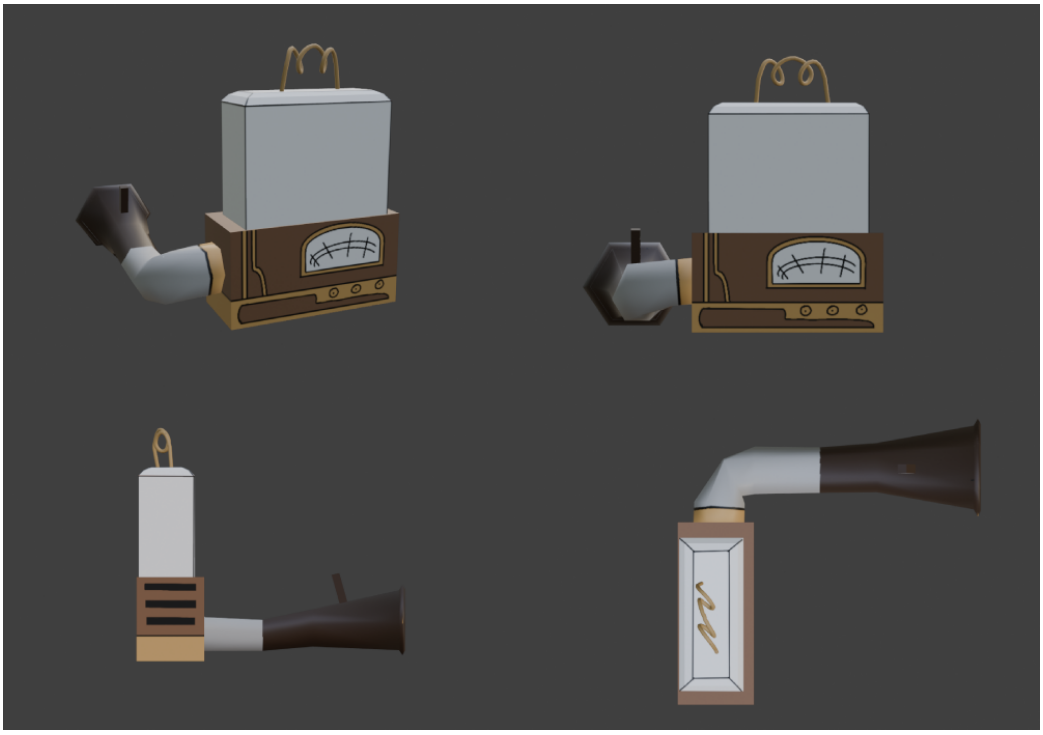

Figura 8.27: Vistas relevantes del arma de Gauss

El primer objeto realizado fue el arma de Gauss ya que es el entorno inmediato a Gauss, este objeto presentaba una variedad de texturas ya que esta encargada de informarle visualmente al jugador que munición esta usando en el momento ya que cada munición tiene un efecto diferente hacia los enemigos, la técnica que utilice fue el de modelación por cajas y subdivisiones.

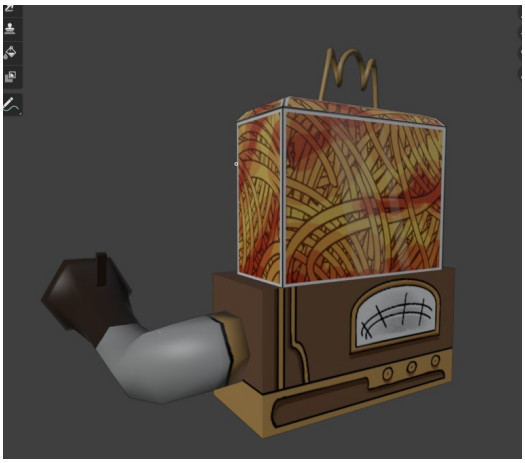

Figura 8.28: Arma de Gauss con munición de spaguetti

Figura 8.29: Arma de Gauss con munición de barrigtonia

Figura 8.30: Arma de Gauss con munición de gelatina

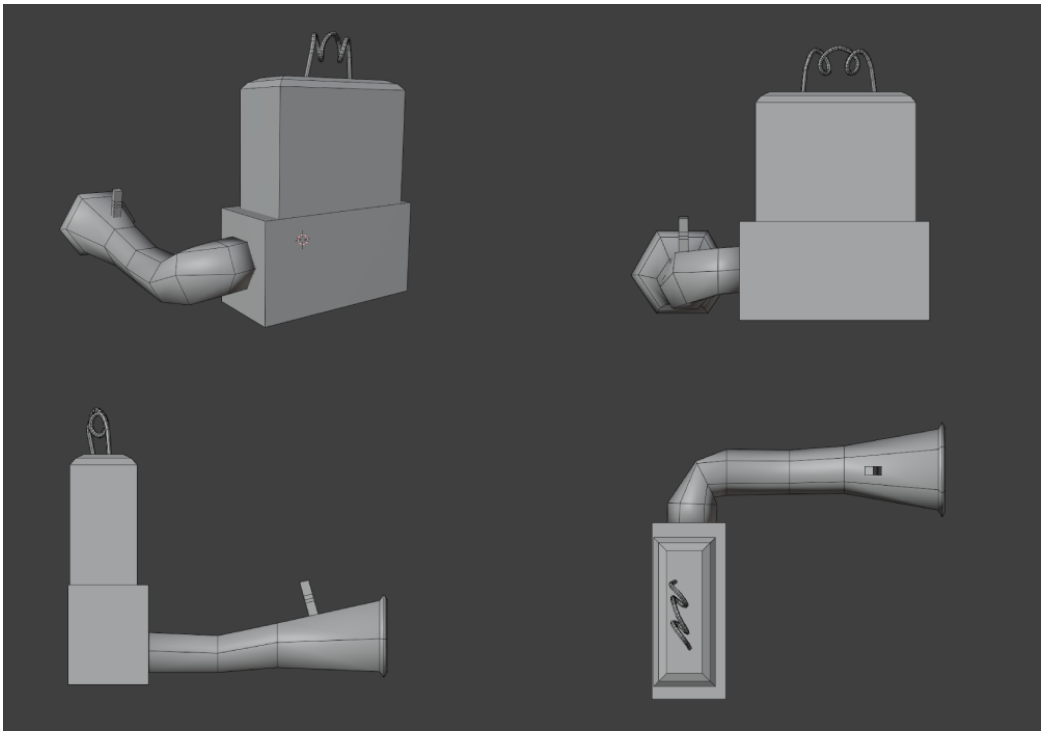

Figura 8.31: Vistas relevantes de la estructura geométrica del arma de Gauss

8.6.3. Primer enemigo: Dingo

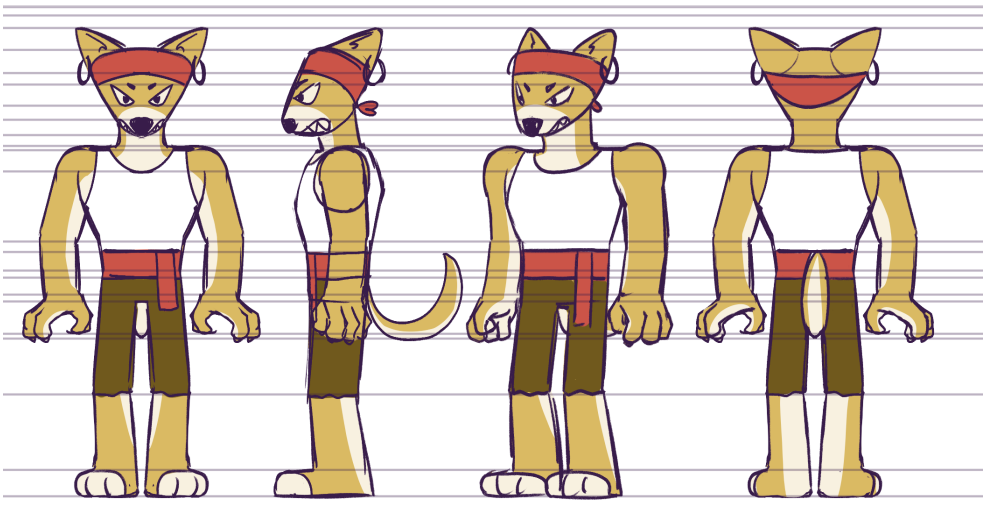

Figura 8.32: Imagen de referencia para el Dingo secuaz

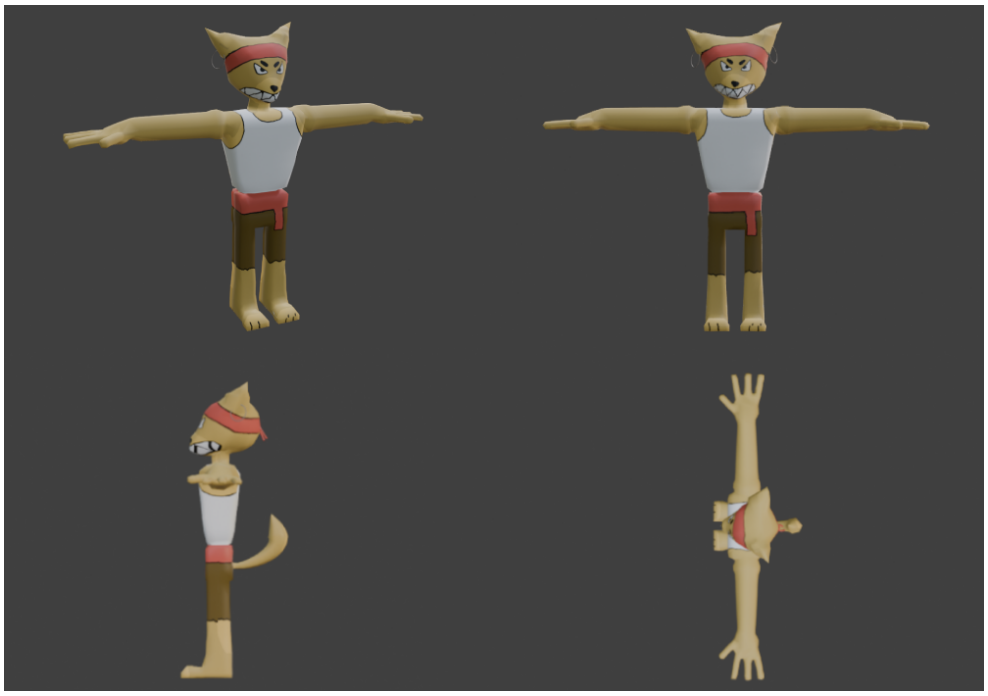

Figura 8.33: Vistas relevantes del Dingo secuaz

El dingo secuaz fue el segundo modelo que se realizó, como ya fue mencionado anteriormente, la cabeza fue hecha con la técnica de escultura y el resto del cuerpo con modelación de caja.

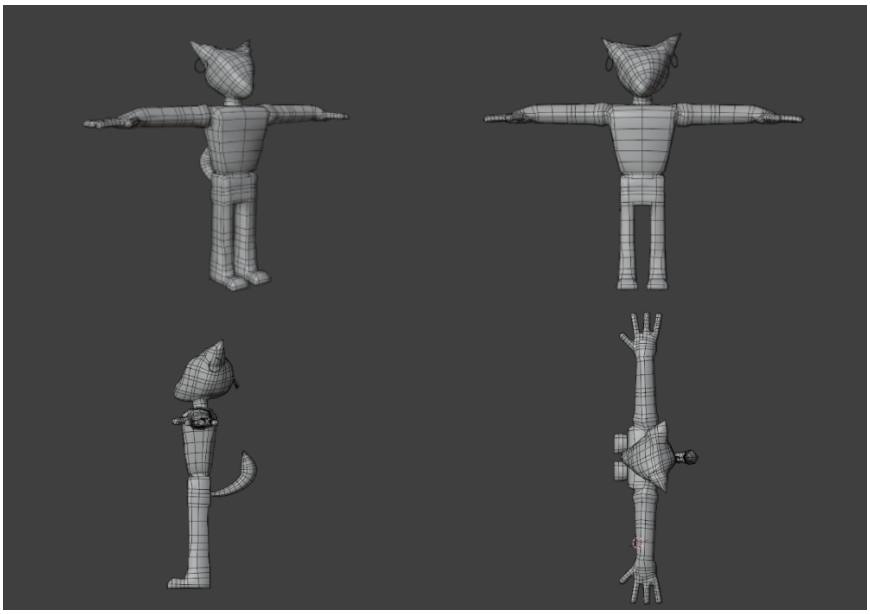

Figura 8.34: Vistas relevantes de la estructura del Dingo secuaz

8.6.4. Arma de los Dingos

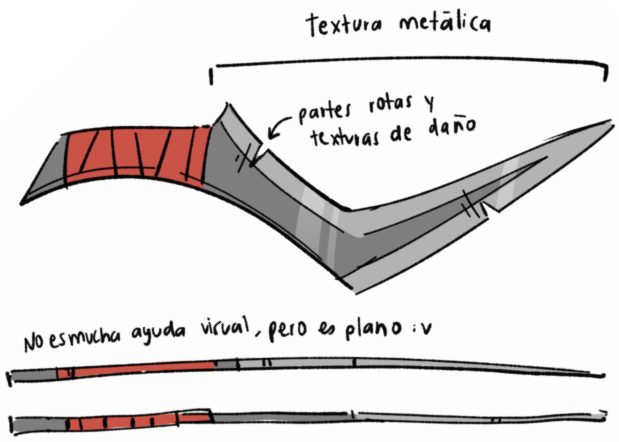

Figura 8.35: Imagen de referencia para la daga

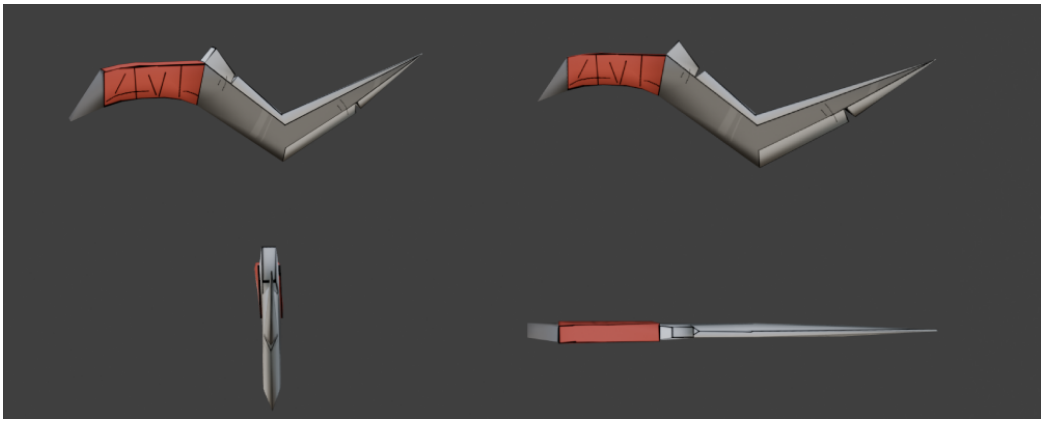

Figura 8.36: Vistas relevantes de la daga

La daga fue creada con el modelado poligonal, ya que al no parecerse a primera vista en una figura primaria se decidió aplicar el modelado poligonal que constó en tener la imagen de referencia posicionada en Bender, colocar un plano enfrente y seguir extruyendo bordes de este para ir formando la silueta de la daga, luego solo aplicar el modificador de *solidify* y afinar detalles. Continuando con el resto de los pasos explicados en el capítulo de metodología.

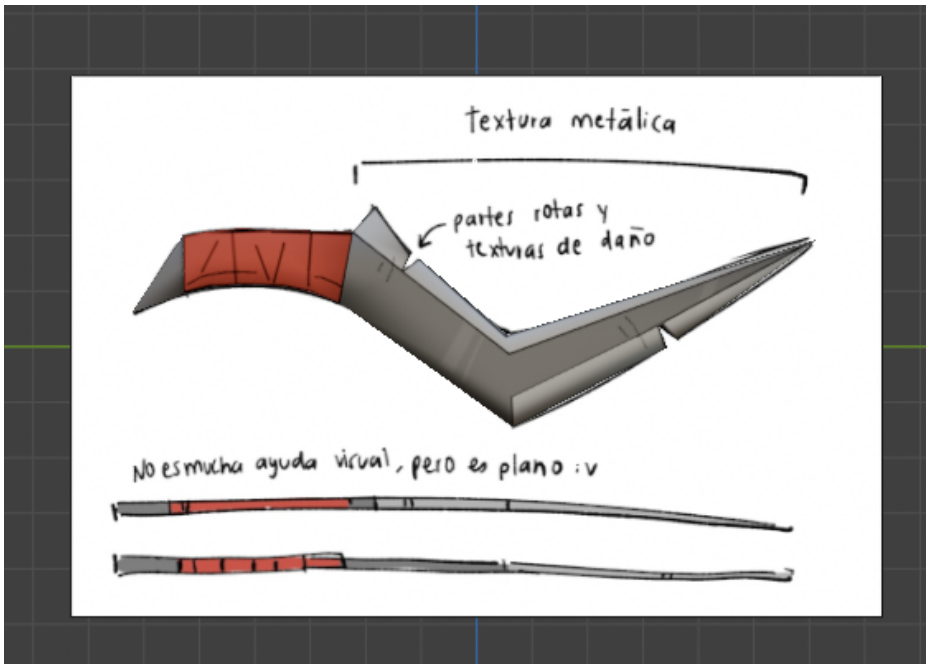

Figura 8.37: Imagen de como quedo la daga por ecima de la imagen de referencia

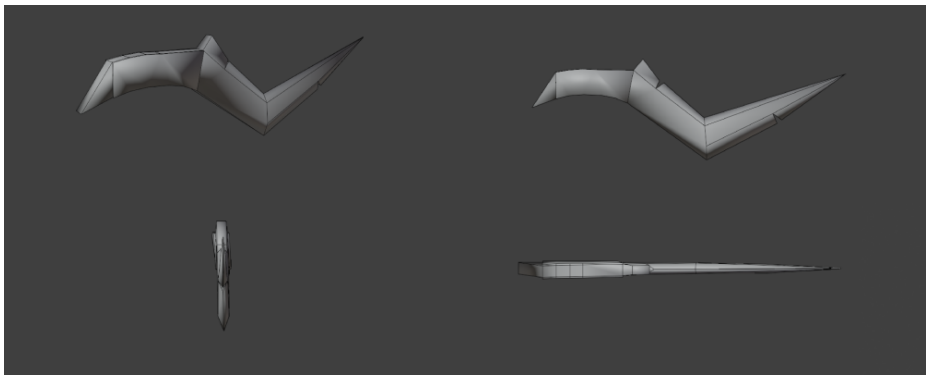

Figura 8.38: Vistas relevantes de la estructura de la daga

8.6.5. Primer jefe enemigo: Dingo Boss

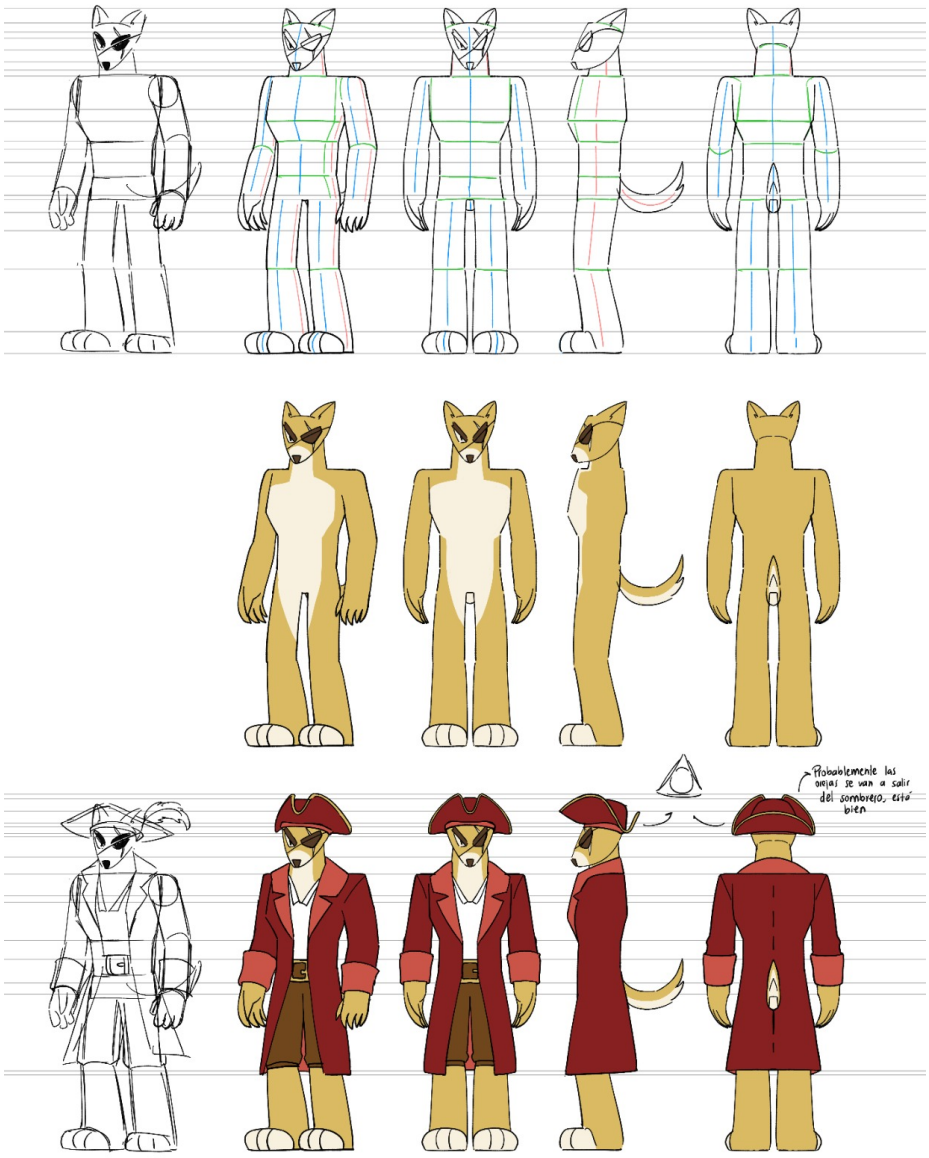

Figura 8.39: Imagen de referencia para el dingo

Figura 8.40: Vistas relevantes de Dingo jefe

Para el jefe Dingo, este fue el primer modelo al que se le le añadió ropa siguiendo el proceso mencionado en la metodología. Se procedió con los detalles, agregando algunos pliegues al área de la manga y diseñando el cuello de la chaqueta. Para el sombrero, se hizo un objeto separado. Sus curvas lo hicieron un poco problemático, al trabajar con muchos bordes, descubrí que se puede usar el modo Esculpir, sin sobrecargar el modelo con geometría adicional. Hice uso de pinceles con tamaños grandes en la herramienta de esculpir y manipulé uniformemente cada vértice disponible para que la curvatura del sombrero pudiera ser suave sin aumentar la densidad de la geometría. Esto me permitiría mantener un modelo de polígonos bajos y optimizado, para brindar suficiente detalle para el sombrero y la ropa del personaje.

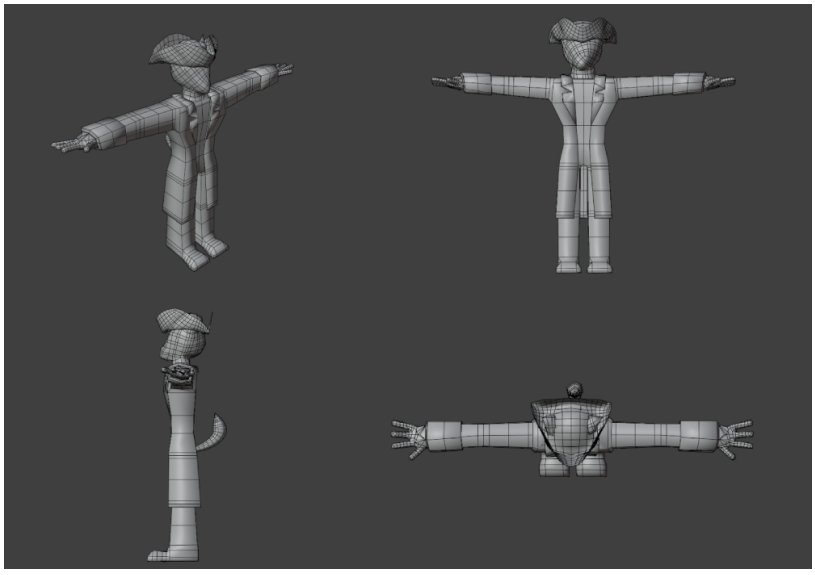

Figura 8.41: Vistas relevantes de la estructura del Dingo jefe

8.6.6. Segundo enemigo: Búho

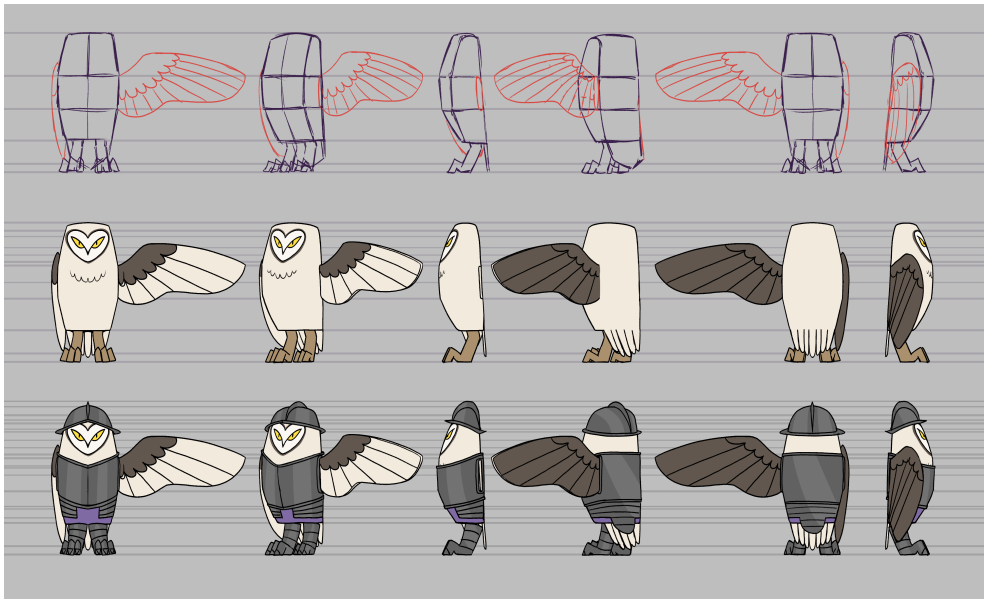

Figura 8.42: Imagen de referencia para el búho

Figura 8.43: Vistas relevantes del búho

Para el modelo del búho secuaz, seguí la misma rutina que para los otros modelos, salvo por una excepción: las plumas. Se me pidió que las plumas se dividieran en partes para poder asignarles pesos individuales en la animación, para que su movimiento fuera más fluido al hacerlo volar. Para ello, comencé el modelo de las alas como un solo objeto, asegurándome primero de que la malla geométrica se prestara a cortar y seleccionar cada pluma, de modo que se pudiera convertir en objetos individuales con bastante facilidad.

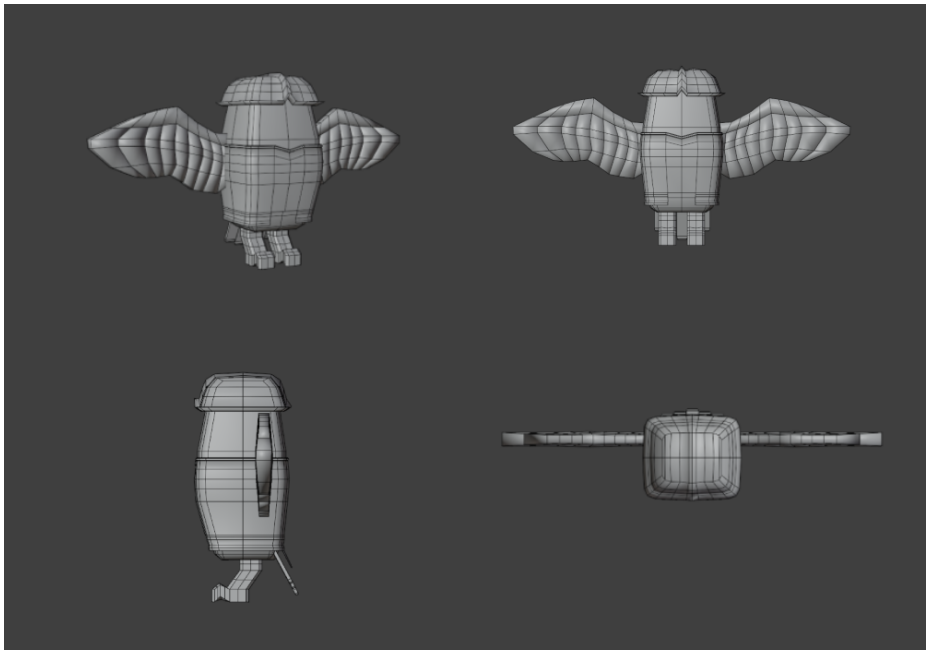

Figura 8.44: Vistas relevantes de la estructura del búho

8.6.7. Segundo jefe enemigo: Búho

Figura 8.45: Imagen de referencia para la jefa búho

Figura 8.46: Vistas relevantes de búho jefe

Tomé algunas de las partes del búho secuaz, como el cuerpo y las garras, para el modelo del búho jefe. Solo las tuve que escalar y ajustar un poco como el cuerpo un poco más grande y las garras un poco más altas como lo definía el arte conceptual. Todo lo demás en el búho jefe se hizo desde cero. Esto también implico la ropa y accesorios que se deben colocar para el modelo. Además, las alas son diferentes, por lo que decidí que modelarlas sería más apropiado que usar las que están diseñadas previamente.

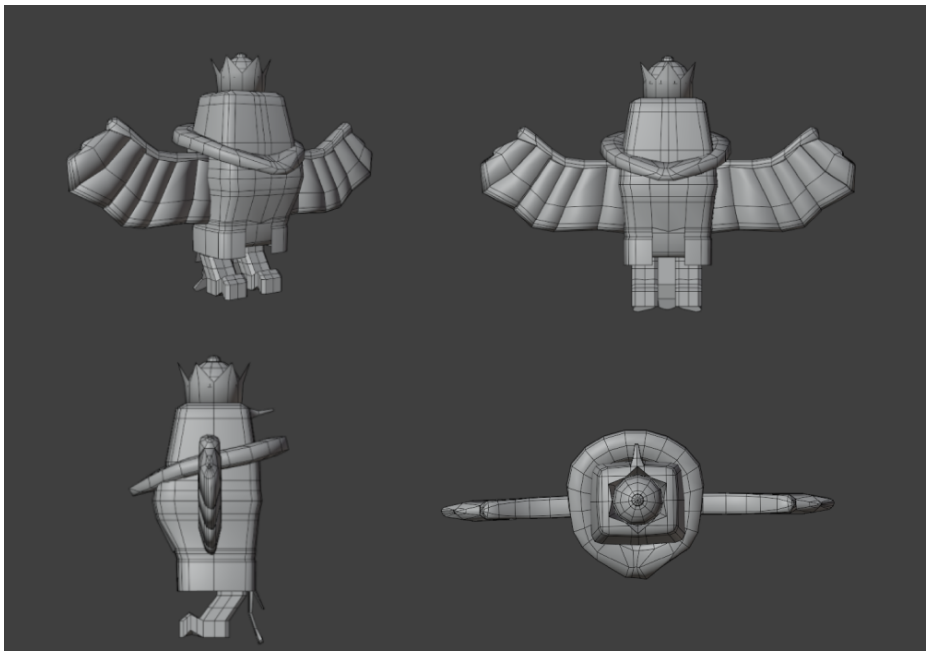

Figura 8.47: Vistas relevantes de la estructura de la búho jefe

8.6.8. Arma de los búhos

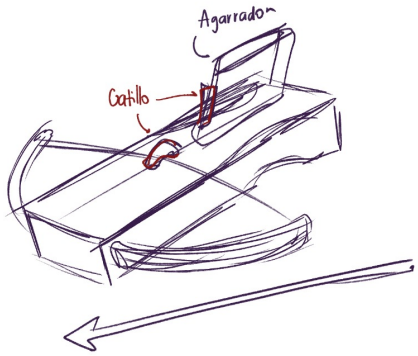

Figura 8.48: Arte conceptual básico para la ballesta

Figura 8.49: Referencia secundaria

De todos los modelos, este fue el último que trabajé. La artista conceptual me dio el diseño base y me pidió que le diera textura. Tuve que hacer algunos cambios a su diseño para que el modelo quedara bien. Además, obtuve algunas referencias de internet adicionales para obtener el mejor acabado posible, como se ve en la referencia secundaria.

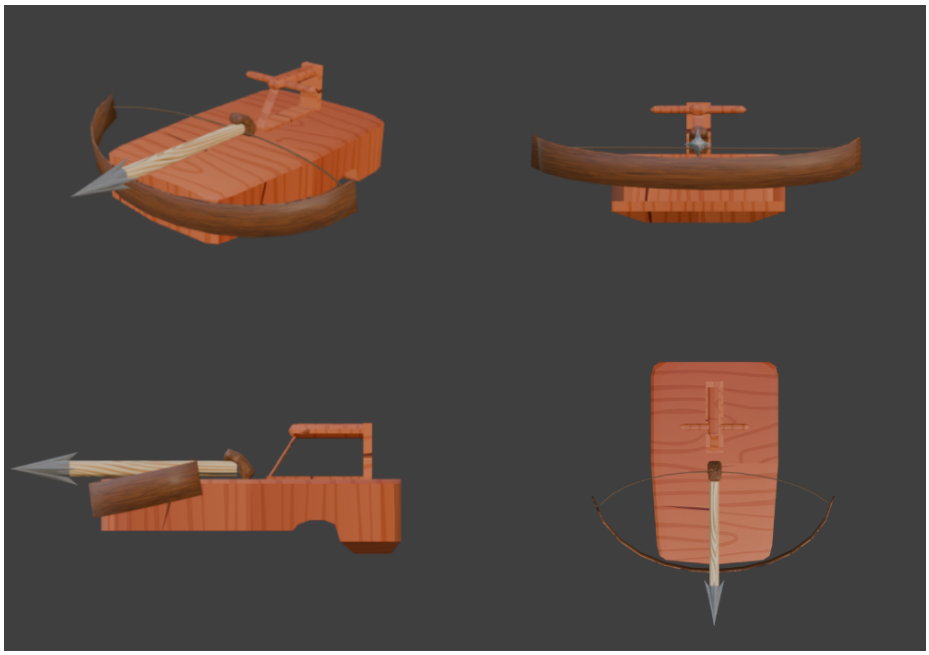

Figura 8.50: Vistas relevantes de la ballesta

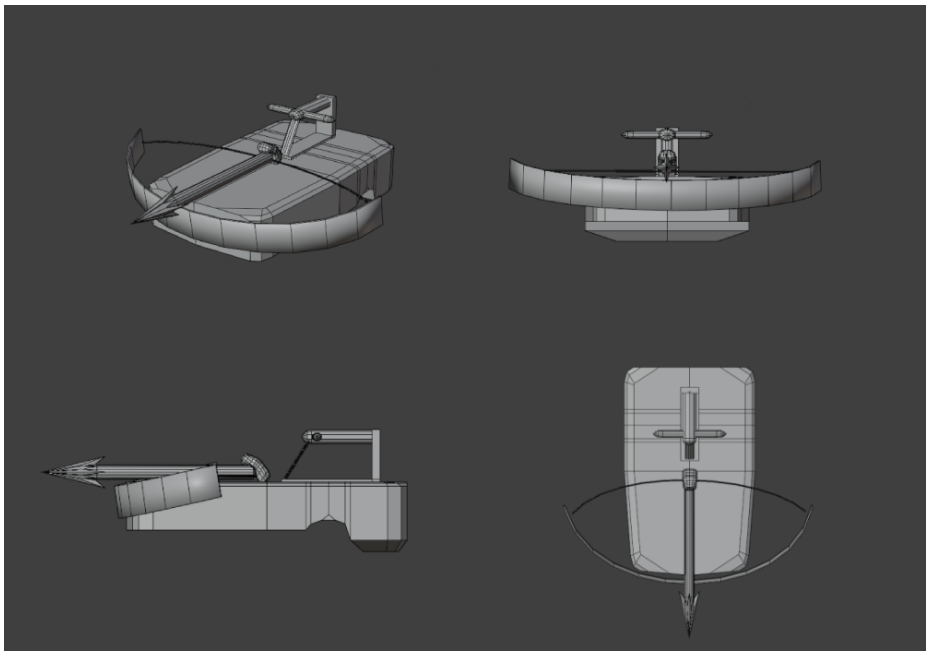

Figura 8.51: Vistas relevantes de estructura de la ballesta

8.6.9. Objetos de municiones

Para las municiones tampoco recibí un arte conceptual muy detallado, ya que iban a ser iguales del lado que se vieran y eran diseños muy simples, por lo que no resultaban en una complejidad al pasarlos a modelado 3D y mis compañeros de trabajo me dieron libertad de dejarlos como a mí me gustaran más, por lo que a partir de estas referencias presentadas conseguí los siguientes resultados.

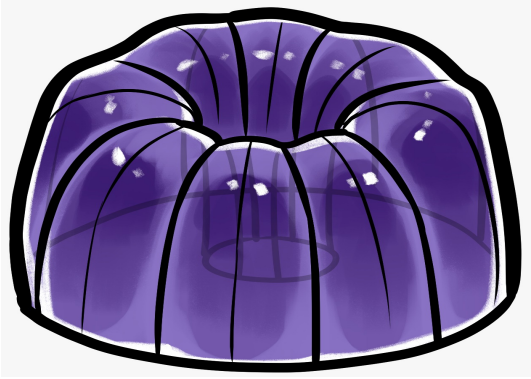

Figura 8.52: Imagen de referencia para la munición de la gelatina

Figura 8.53: Imagen de referencia para la municion de la pasta

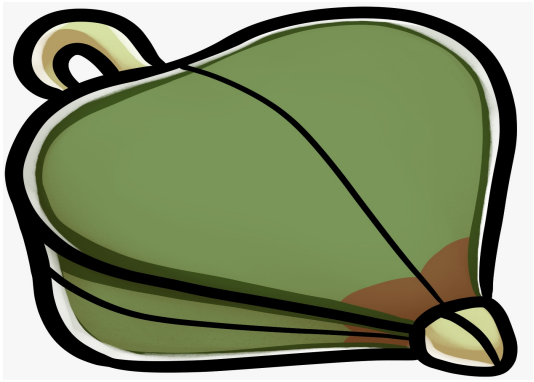

Figura 8.54: Imagen de referencias de las municiones

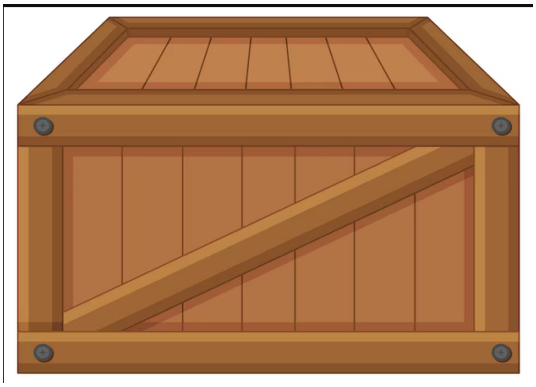

Figura 8.55: Imagen de referencia para la municion de la gelatina

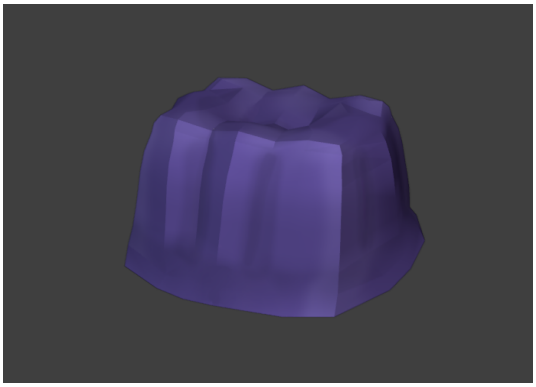

Figura 8.56: Munición: Gelatina

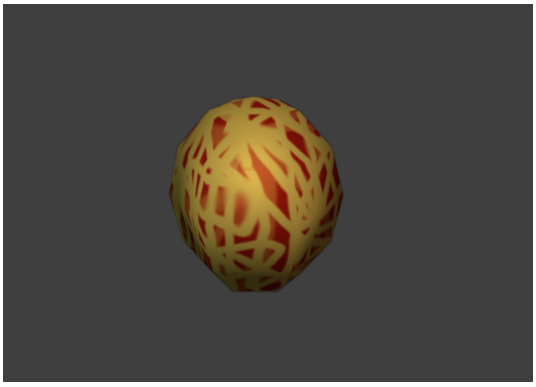

Figura 8.57: Munición: Spaguetti

Figura 8.58: Munición: Barringtonia

Figura 8.59: Caja de municion

8.7. Resultados animación 3D

Se completó el proceso de *rigging* para los cuatro modelos seleccionados, permitiendo que cada uno contara con una estructura esquelética funcional. Esta estructura proporcionó la base para la movilidad y la manipulación de los modelos en las animaciones posteriores.

Para realizar el *rigging* de los cuatro modelos seleccionados, se utilizó la herramienta Auto-Rig Pro de Blender. Primero, se colocaron los indicadores de posición en las áreas clave del modelo, permitiendo a la herramienta realizar una detección automática de los huesos principales en la estructura. Durante este proceso, algunos huesos no coincidieron de forma precisa con la anatomía de los modelos, por lo que fue necesario realizar ajustes manuales para optimizar la alineación y asegurar que el esqueleto se adaptara correctamente al cuerpo de cada modelo.

Asimismo, se añadieron huesos adicionales en áreas específicas que requerían movimientos complejos. Posteriormente, se completó el proceso de binding para vincular el mesh del modelo al rig, asegurando que los movimientos de la estructura esquelética se transmitieran fielmente al exterior del modelo y quedaran listos para las animaciones. A continuación, se presentarán los resultados del *rigging* de cada modelo.

El proceso de *rigging* del personaje principal, Gaus, un ornitorrinco, requirió una adaptación considerable debido a las particularidades anatómicas del modelo en comparación con un humanoide. Al iniciar con el primer modelo de Gaus, se observó que la estructura ósea debía ajustarse proporcionalmente, ya que el cuerpo de un ornitorrinco es más pequeño y sus huesos vertebrales son relativamente más largos en comparación con un modelo humano estándar, ver Figura 8.60.

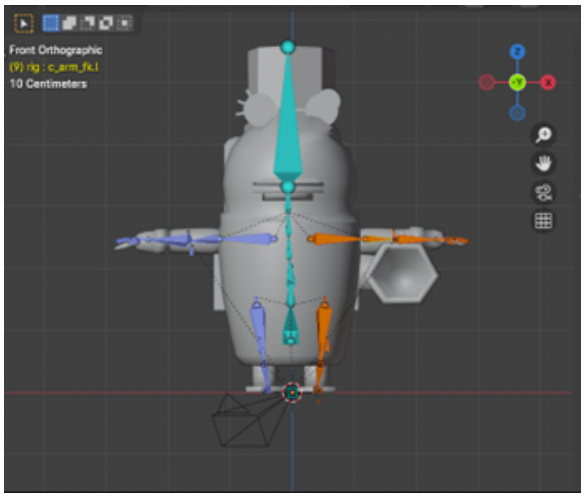

Figura 8.60: Resultados del rigging del cuerpo de Gaus

Además, se modificaron las extremidades, dado que las patas de Gaus son más cortas y robustas que las de un humano. Así como también se ajustaron las manos, ver Figura 8.61.

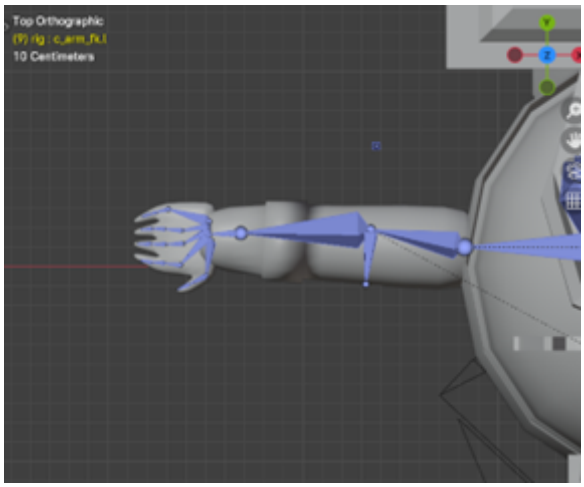

Figura 8.61: Resultados del rigging de las manos de Gaus

Fue necesario agregar huesos específicos en la zona de la cola, ya que la herramienta Auto-Rig Pro, que se empleó para el *rigging*, no detecta correctamente las estructuras de ciertos animales salvo casos específicos, como aves, entre otros. Por lo tanto, el *rigging* de Gaus se realizó siguiendo un enfoque humanoide, al cual se le añadieron huesos adaptados al tamaño y proporciones del ornitorrinco, sin necesidad de eliminar huesos existentes, sino ajustándolos para mantener la coherencia con la anatomía del personaje. La Figura 8.62 muestra el *rigging* de la cola en vista de perfil.

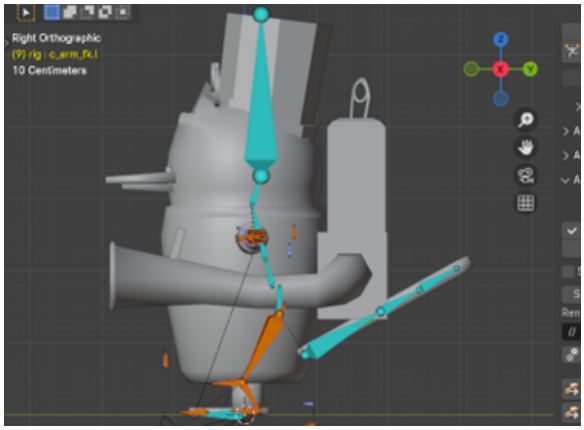

Figura 8.62: Resultados del rigging de las manos de Gaus

Por otro lado, en el caso del segundo y tercer modelo, correspondientes al Dingo normal y el Dingo jefe, se aplicaron ajustes similares en el *rig*, adaptándolo a sus tamaños respectivos y agregando huesos adicionales, ver Figura 8.63.

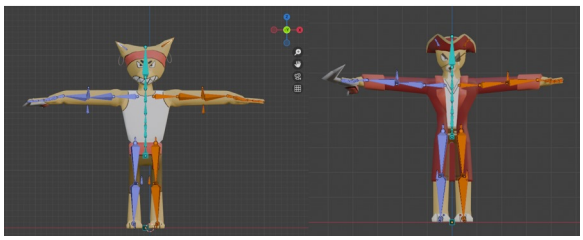

Figura 8.63: Resultados del *rigging* del cuerpo de ambos Dingos

Sin embargo, para estos modelos fue necesario un mayor número de modificaciones, agregando más huesos y eliminando algunos. Además de incorporar huesos en la cola, se añadieron huesos en las orejas para permitir un movimiento sutil en esta área, ver Figura 8.64.

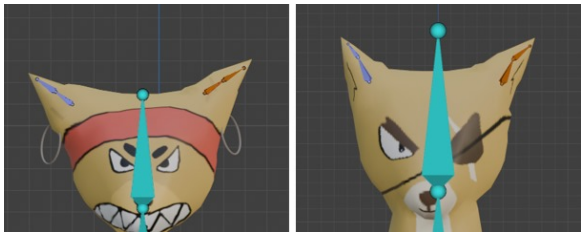

Figura 8.64: Resultados del rigging de las orejas de ambos Dingos

En cuanto a la simplificación, se eliminó el hueso correspondiente al dedo medio de ambas manos, dado que tanto el Dingo normal como el Dingo jefe solo tienen cuatro dedos en sus manos, ver Figura 8.65.

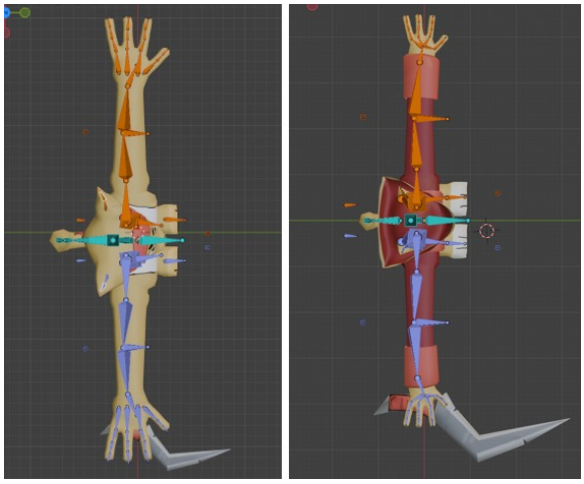

Figura 8.65: Resultados del rigging de las manos de ambos Dingos

En el cuarto modelo, correspondiente al búho (otro personaje villano), el rigging fue completamente diferente debido a que la estructura del búho no tiene similitud con un humanoide. Auto-Rig Pro ofrece una herramienta específica para generar un rigging básico para aves, lo cual facilitó la creación de la estructura inicial. En este modelo también se ajustó el posicionamiento del cuerpo y se agregó un hueso en la cola ver Figura 8.66.

Figura 8.66: Resultados del *rigging* de la cola del búho

Además, los huesos de las alas se ajustaron cuidadosamente para alinearse con las alas del modelo. Para las plumas, se añadieron los huesos necesarios para lograr una animación fluida, optimizando la cantidad de huesos, ya que agregar un hueso para cada pluma en un modelo 3D caricaturesco podría distorsionar la apariencia del modelo. Por ello, se eliminaron algunos huesos en las alas, en áreas de plumas donde no eran esenciales para la animación ver Figura 8.67.

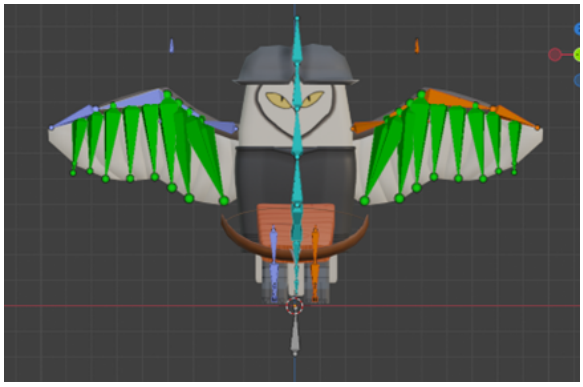

Figura 8.67: Resultados del rigging del búho

Finalmente, para este modelo fue necesario agregar un hueso adicional en la zona inferior, cerca de las patas, específicamente en el centro. Este hueso se utilizó para vincular el arma del personaje, una ballesta, permitiendo una integración y movimiento coherentes entre el arma y el personaje durante la animación, ver Figura 8.68.

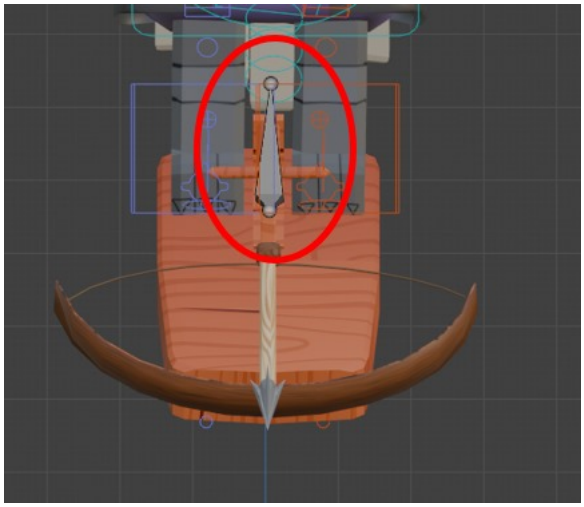

Figura 8.68: Resultados del hueso extra para el arma del búho

En cuanto a el resultado de las animaciones, para cada modelo se crearon animaciones específicas que complementan su personalidad y función dentro del juego. En el caso de Gaus, el ornitorrinco, se desarrollaron las siguientes animaciones: Idle (inactivo), KO (muerte), salto, correr, caminar, apuntar el arma y una animación extra, que consiste en un salto prolongado para simular una caída larga. Ver Figura 8.69, que ejemplifica la animación de KO (morir).

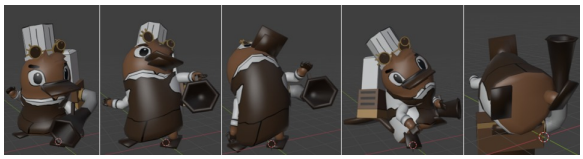

Figura 8.69: Resultados de animación KO (morir) de Gaus

Para los modelos del Dingo normal y el Dingo jefe, se realizaron animaciones similares, adaptadas a su rol en el juego: Idle, KO, correr, caminar y lanzar el arma. Ver Figuras 8.70 y 8.71, que ejemplifican una de las animaciones realizadas por cada uno

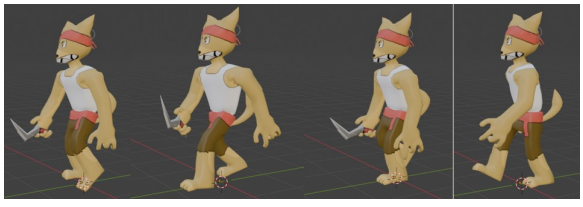

Figura 8.70: Resultados de animación caminar del Dingo normal

Figura 8.71: Resultados de animación de lanzar arma del Dingo jefe

Por último, para el modelo del búho, se llevó a cabo únicamente la animación extra que consistió en volar mientras sostiene su arma, ver Figura 8.72.

Figura 8.72: Resultados de animación extra de volar del Búho

Por otro lado, los árboles de transiciones de animaciones en Unity, gestionados a través del componente Animator, permitieron controlar de manera efectiva el flujo de animaciones de cada modelo. A continuación, se muestran los árboles de transiciones correspondientes a cada uno de los personajes.

Para el modelo de Gaus, las transiciones entre los estados de Correr, Idle y Caminar y saltar en el aire utilizaron parámetros booleanos, lo que permitió activar o desactivar estos estados en función del movimiento del personaje. Además, las animaciones de Saltar, KO y Disparar se gestionaron mediante *triggers*, que se activaron en momentos específicos, como al recibir daño o al ejecutar un ataque y saltar rápidamente, ver Figura 8.73.

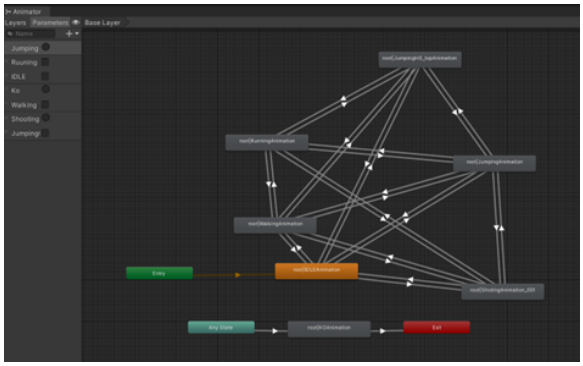

Figura 8.73: Árbol de transiciones de animaciones de Gaus

En el caso de los modelos del Dingo, tanto el normal como el jefe, el árbol de transiciones abarcó los estados de Idle, Caminar, Correr, Lanzar arma y KO. Para estos modelos, se utilizaron parámetros con variables booleanas para las transiciones de Idle, Caminar y Correr, lo que permitió activar o desactivar estos estados según el movimiento del personaje. Además, las animaciones de KO y Lanzar arma se gestionaron mediante triggers que se activaron en momentos específicos, como cuando el personaje recibía daño o realizaba un ataque, ver Figura 8.74.

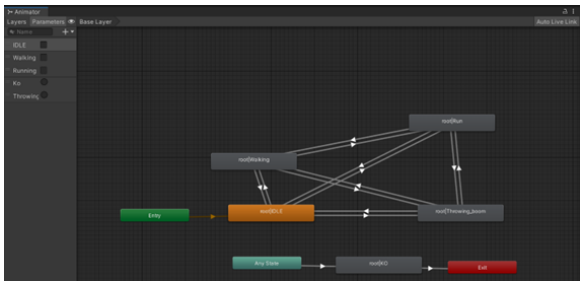

Figura 8.74: Arbol de transiciones de animaciones de Dingo y Dingo jefes

Para el modelo del búho, se desarrolló un árbol de transiciones más simplificado, dado que la única animación ejecutada fue la animación extra que consistía en volar mientras sostenía su arma. Esta animación se activó bajo condiciones específicas, dependiendo de la interacción del jugador y la lógica del juego, ver Figura 8.75.

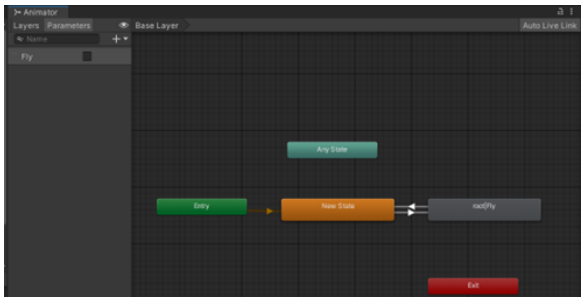

Figura 8.75: Árbol de transiciones de animación del Búho

Se desarrollaron diversos scripts en Unity, para integrar las animaciones a los movimientos de los personajes, garantizando una interacción fluida y dinámica durante el juego. Cada script fue diseñado con una clase correspondiente al nombre del modelo, como *GausAnimations*, *DingoAnimations* y *BossDingoAnimations*, entre otros. Dentro de cada clase, se definieron métodos específicos para cada animación, a excepción de la animación *Idle*, que se inicializa automáticamente al comenzar el juego.

Por ejemplo, en el *script* de *GausAnimations*, se implementaron métodos como *WalkingAnimation*, *RuningAnimation*, *Jumping*, *JumpingInStop*, *ShootAnimation* y *KO*. Dependiendo de la naturaleza de la animación, se utilizaron parámetros booleanos o *triggers* para activar las transiciones. Un ejemplo de esta implementación es el siguiente, ver Figura 8.76:

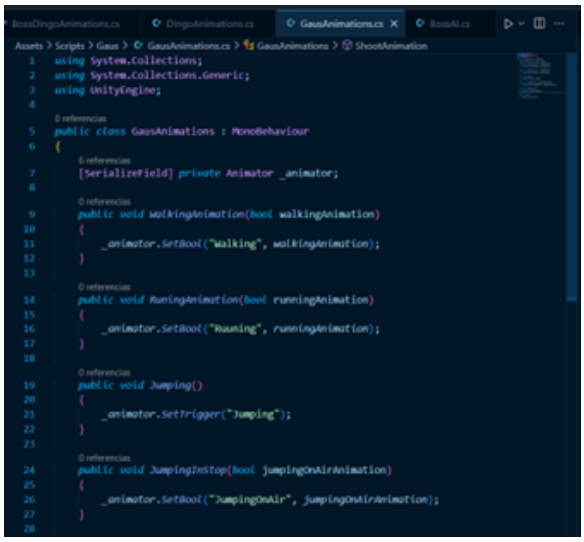

Figura 8.76: Script de animaciones de Gaus

Posteriormente, en el script que maneja la mecánica del modelo, se inicializó la conexión con las animaciones utilizando la declaración: *private GausAnimations animations;*. Esto permitió definir el componente responsable de llamar a las animaciones: *_animations = GetComponent<GausAnimations>();*.

A continuación, se llamaron los métodos correspondientes según las entradas del jugador. Por ejemplo, al recibirla entrada de salto, se invocó el método, ver Figura 8.77:

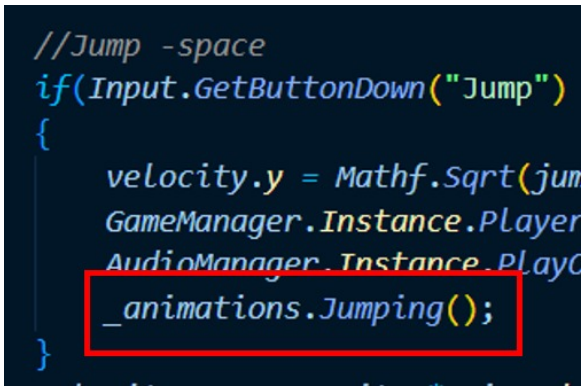

Figura 8.77: Llamada a los métodos correspondientes de animaciones de Gaus

De manera similar, para las animaciones de correr y caminar, se utilizaron los métodos, ver Figura 8.78.

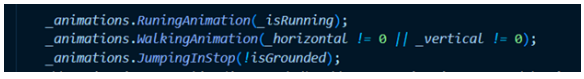

Figura 8.78: Llamada a los métodos correspondientes de animaciones de Gaus

En el caso de los villanos, como el Dingo normal, el Dingo jefe y el búho, que operan con inteligencia artificial, la implementación se realizó de manera similar, pero con ajustes específicos

para reflejar su comportamiento autónomo. Para estos modelos, la condición que activa la animación de caminar se estableció de la siguiente manera, ver Figura 8.79.

```
if (!isTiedUp)
{
    agent.SetDestination(player.position);
    // ! ANIMATION:
     _animations.RuningAnimation(agent.velocity.magnitude > 0.1f);
}
```

Figura 8.79: Llamada a los métodos correspondientes de villanos con IA

Se accedió a la propiedad *"agent.velocity.magnitude"* para determinar la velocidad actual del personaje controlado por inteligencia artificial (IA) en el juego. Una vez implementadas y verificadas las animaciones de los personajes en Unity, se procedió a realizar un *GameTesting* de juego para evaluar la experiencia de los jugadores.

El juego fue puesto a disposición de un grupo aleatorio de participantes, quienes interactuaron con los diferentes modelos animados en diversas situaciones del entorno del juego. Después de la sesión de juego, los participantes completaron un formulario diseñado para recopilar sus impresiones sobre las animaciones. Este cuestionario incluyó preguntas específicas sobre la percepción de las animaciones, etc. La retroalimentación obtenida fue analizada para identificar áreas de mejora y confirmar la efectividad de las animaciones implementadas. Los resultados se muestran en la siguiente tabla. Ver Tabla 8.80.

Pregunta	Tipo de Respuesta	Resultado Promedio	Porcentaje de Respuestas Afirmativas
1. ¿Qué tan inmersiva le pareció la experiencia visual del juego gracias a las animaciones de los personajes?	Escala 1-10	Promedio: 7.6	-
2. ¿Las animaciones lograron transmitir la narrativa de "Gaus y la Revolución Ornitorense"?	Sí/No	-	80% Sí
3. ¿Se sintió más conectado o empático hacia los personajes debido a sus animaciones?	Sí/No	-	76% Sí
4. ¿Las animaciones fueron fluidas y se adaptaron al ritmo del juego?	Escala 1-5	Promedio: 4.0	-
5. ¿Identificó los distintos estados de animación (caminar, correr, saltar, etc.)?	Sí/No	-	80% Sí
6. ¿Qué tan satisfactorias le parecieron las transiciones entre animaciones?	Escala 1-10	Promedio: 8.0	-
7. ¿Notó los principios de animación como anticipación y encogerse en los movimientos de los personajes?	Sí/No	-	76% Sí
8. ¿Las animaciones contribuyeron al atractivo visual del juego?	Escala 1-10	Promedio: 7.3	-
9. ¿Las animaciones mejoraron su comprensión de la trama?	Escala 1-5	Promedio: 4.3	-
10. ¿Las animaciones fueron creíbles dentro del contexto caricaturesco y de ciencia ficción del juego?	Escala 1-5	Promedio: 4.2	-

Figura 8.80: Resultados de formulario del *GameTesting*

Para cada pregunta en escala (del 1 al 5 o del 1 al 10), el promedio fue calculado multiplicando cada opción por la cantidad de votos recibidos, sumando todos los valores y luego dividiendo por el total de votos (25). Para las preguntas de "Sí/No", el porcentaje de respuestas afirmativas fue calculado en base a 25 respuestas totales.

Por último, los principios de animación exitosos en el transcurso de las dieciocho animaciones generadas fueron: estirar y encoger, anticipación, puesta en escena, pose a pose, acción continua y superpuesta, exageración, acelerar y desacelerar movimiento en arcos, acción secundaria y atractivo.

8.8. Resultados: Primer testing con usuarios

En el primer testing con usuarios, no se obtuvieron resultados significativos debido a la naturaleza del evento en el que se llevó a cabo. Durante este evento, los usuarios tenían la oportunidad de observar la página web mientras jugaban, pero no interactuaron activamente con ella. En su lugar, solo ofrecieron algunas sugerencias sobre elementos que les hubiera gustado ver en el sitio. Esto limitó la obtención de datos concretos sobre la usabilidad y la experiencia de usuario en esta fase inicial, dado que la interacción directa y el uso activo de la página no formaron parte del proceso de testing en esta ocasión.

8.9. Resultados: Segundo testing con usuarios

En el segundo testing con usuarios, se recopilaron un total de 94 respuestas, lo que permitió obtener una visión más amplia de la usabilidad y satisfacción general con la página web. A continuación, se presentan los resultados obtenidos para cada una de las preguntas clave de este testeo, junto con los gráficos correspondientes que muestran la distribución de las respuestas.

8.9.1. Facilidad de navegación

A la pregunta *"¿Qué tan fácil fue navegar por la página web?"*, el 48.9 % de los usuarios reportaron que navegar por la página fue **muy fácil**, mientras que un 36.2 % la calificaron como **fácil**. Solo un 11.7 % consideró la navegación **regular** y un 3.2 % de los usuarios consideró la navegación **difícil**. (Ver Figura 8.81)

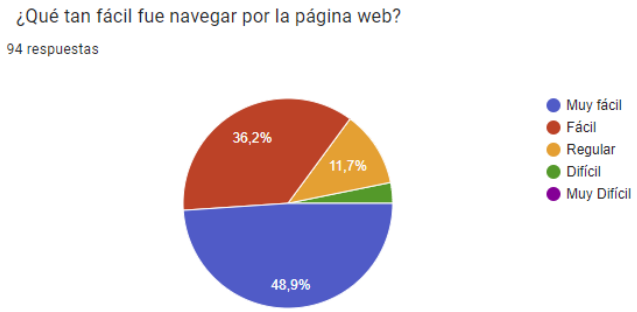

Figura 8.81: Facilidad de navegación en la página web

8.9.2. Organización de la información

En cuanto a la organización de la información, el 47.9 % de los participantes estuvieron **totalmente de acuerdo** que la información estaba organizada de manera clara y comprensible, y un 33 % se mostró **de acuerdo**. Sin embargo, un 11.7 % tuvo una percepción **neutral**, 3.2 % de los usuarios usuarios mostraron cierto desacuerdo y un 4.3 % de los usuarios se mostraron totalmente en desacuerdo. (Ver Figura 8.82)

Figura 8.82: Opinión sobre la organización de la información

8.9.3. Apariencia en dispositivos móviles/tablets

Respecto a la apariencia de la página en dispositivos móviles y tablets, el 45.7 % de los usuarios calificaron la apariencia como **muy buena**, mientras que un 42.6 % la consideró **buena**. Un 10.6 % indicó una experiencia **regular** en dispositivos móviles, con solamente un 1.1 % de los usuarios indicó una experiencia **muy mala**. (Ver Figura 8.83)

Figura 8.83: Calificación de la apariencia en dispositivos móviles

8.9.4. Facilidad para hacer clic en botones/enlaces en dispositivos móviles/tablets

Finalmente, en cuanto a la facilidad para hacer clic en botones y enlaces en dispositivos móviles, el 67 % de los usuarios afirmaron que los botones y enlaces eran **fáciles de hacer clic**, mientras que un 22.3 % indicó que esta experiencia era variable, dependiendo del caso. Un pequeño porcentaje, el 10.6 %, consideró que no era fácil hacer clic en estos elementos. (Ver Figura 8.84)

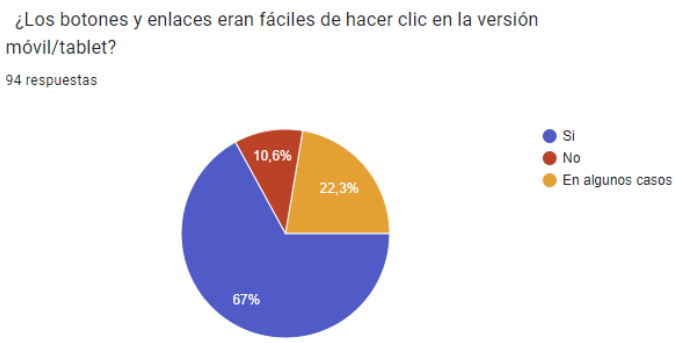

Figura 8.84: Facilidad para hacer clic en botones y enlaces en dispositivos móviles

8.10. Resultados: Tercer testing con usuarios

En el tercer testing con usuarios, se enfocó en evaluar el tiempo que cada usuario necesitaba para completar una serie de tareas específicas en la página web, tanto en su versión de escritorio como en la versión móvil. Los tiempos de cada usuario para las tareas se presentan a continuación:

Tabla 8.1: Tiempos de tareas en desktop y móvil

Usuario	**Tarea 1**	**Tarea 2**	**Tarea 3**	**Tarea 4**	**Tarea 5**
Versión desktop					
Alejandro Gómez	15.34s	4.12s	7.34s	2.56s	5.34s
Elean Rivas	9.54s	6.10s	3.66s	10.63s	7.34s
Marco Orozco	11.54s	9.98s	6.17s	16.62s	15.88s
Dariel Villatoro	12.55s	9.14s	6.54s	4.31s	5.21s
Paola Contreras	9.18s	7.47s	6.09s	7.10s	15.88s
Sergio Orellana	2.28s	13.70s	6.44s	8.80s	3.71s
Ángel Gabriel	5.64s	11.49s	13.88s	16.47s	10.19s
Versión móvil					
Alejandro Gómez	17.95s	6.23s	10.66s	5.36s	8.77s
Elean Rivas	12.95s	9.33s	5.59s	12.45s	8.64s
Marco Orozco	14.48s	9.60s	5.20s	18.45s	17.64s
Dariel Villatoro	15.71s	12.25s	9.73s	6.57s	7.78s
Paola Contreras	8.12s	13.24s	7.74s	7.19s	5.83s
Sergio Orellana	2.31s	8.80s	7.73s	10.64s	5.69s
Ángel Gabriel	4.36s	9.40s	3.44s	9.93s	7.29s

Además, con base en las respuestas obtenidas para cada una de las preguntas, se construyeron matrices de retroalimentación 2x2 para organizar los comentarios de los usuarios y proporcionar una visión clara de las áreas de mejora y los aspectos positivos observados.

8.10.1. Retroalimentación por pregunta

- **Pregunta 1: ¿Qué tan fácil fue navegar por la página web?**

Tabla 8.2: Matriz de retroalimentación sobre la facilidad de navegación

Fortalezas	**Oportunidades de mejora**
Muy fácil. 85 % sencillo.	Fácil, aunque faltan detalles del juego. Demasiado fácil, el diseño podría mejorar.
Sugerencias adicionales	**Problemas críticos**
Sencillo, pero podría tener una personalización más adecuada al juego. Buena navegación, pero algunos accesos fallan.	Fácil, aunque en algunas funcionalidades como Editar Perfil se cerraba sesión cuando actualizaba.

- **Pregunta 2: ¿Consideras que la información está organizada de manera clara y comprensible?**

Tabla 8.3: Matriz de retroalimentación sobre la organización de la información

Fortalezas	**Oportunidades de mejora**
Claro y comprensible, pero la página principal es simple. Bien organizada, aunque falta algo de contexto.	Sí, pero algunos títulos pueden mejorar. Está bien, pero la página principal debería de tener la sinopsis del juego.
Sugerencias adicionales	**Problemas críticos**
Sí, solo falta la sinopsis para facilitar información.	Sí, aunque podría agregar un poco más de información del juego.

- **Pregunta 3: ¿Cómo calificarías la apariencia del sitio en tu dispositivo móvil/tablet?**

Tabla 8.4: Matriz de retroalimentación sobre la apariencia en dispositivos móviles

Fortalezas	**Oportunidades de mejora**
10, aunque la imagen central estaba recortada. Me gusta, pero cambiaría alguna tipografía.	Se ve bien, pero en móvil la imagen central no aparece completa. 8, pero usaría otros colores en el menú.
Sugerencias adicionales	**Problemas críticos**
Buena en general, pero no tan bien como en la computadora. Llamativa y creativa, aunque hay detalles que ajustar.	Cómoda en móvil, pero la imagen central no se muestra bien.

- **Pregunta 4: ¿Los botones y enlaces eran fáciles de hacer clic en la versión móvil/tablet?**

Tabla 8.5: Matriz de retroalimentación sobre la facilidad para hacer clic en botones/enlaces

Fortalezas	**Oportunidades de Mejora**
Sí, aunque algunos botones podrían ser más grandes. Todo bien organizado, pero mejorar el tamaño ayudaría.	Sí, pero algunos enlaces se sienten pequeños. Sí, aunque en móvil algunos botones se ven muy justos.
Sugerencias Adicionales	**Problemas Críticos**
Funciona bien, aunque mejorar el tamaño facilitaría el uso.	Fácil de hacer clic, pero podrían ser un poco más grandes.

8.11. Resultados: *Testing* final con usuarios

En el *testing* final con usuarios, se evaluó nuevamente el tiempo que cada usuario necesitaba para completar una serie de tareas específicas en la página web, tanto en su versión de escritorio como en la versión móvil. Además, en esta fase se incorporó el uso de mapas de calor para analizar las áreas de interés visual de los usuarios y obtener información adicional sobre la interacción con la interfaz. Los tiempos de cada usuario para las tareas se presentan a continuación:

Tabla 8.6: Tiempos de tareas en desktop y móvil

Usuario	**Tarea 1**	**Tarea 2**	**Tarea 3**	**Tarea 4**	**Tarea 5**
Versión desktop					
Alejandro Gómez	9.24s	3.35s	6.54s	3.76s	2.70s
Elean Rivas	7.84s	5.43s	2.72s	9.43s	5.87s
Marco Orozco	9.43s	8.32s	5.40s	7.43s	6.38s
Dariel Villatoro	10.21s	7.34s	4.67s	4.25s	5.14s
Paola Contreras	8.78s	7.34s	5.76s	5.43s	9.98s
Sergio Orellana	3.79s	8.23s	6.36s	7.69s	3.65s
Ángel Gabriel	5.59s	9.95s	10.11s	12.45s	8.09s
Versión móvil					
Alejandro Gómez	10.63s	4.95s	8.46s	5.36s	4.91s
Elean Rivas	8.87s	7.42s	4.58s	10.65s	6.23s
Marco Orozco	7.41s	5.59s	4.87s	9.44s	7.98s
Dariel Villatoro	12.64s	9.45s	6.73s	5.32s	6.89s
Paola Contreras	7.31s	8.40s	6.92s	7.24s	5.71s
Sergio Orellana	2.29s	8.43s	6.87s	9.53s	5.12s
Ángel Gabriel	4.16s	7.87s	3.36s	9.93s	7.34s

Además, los comentarios de los usuarios para cada una de las preguntas reflejaron una percepción predominantemente positiva de la página web. En lugar de matrices de retroalimentación, por lo que se consideró realizar una síntesis de las observaciones y fortalezas en cada aspecto evaluado:

8.11.1. Facilidad de navegación

Los usuarios reportaron una experiencia de navegación fluida y clara en la página web. Comentarios como “La navegación es fluida y rápida”, “Muy fácil de usar, con una disposición clara de elementos” y “Navegar fue sencillo” destacan la facilidad para encontrar y acceder a las diferentes secciones de la página. En general, la estructura y distribución de contenido fueron bien recibidas, y los participantes indicaron que les resultó fácil y cómodo localizar cada sección.

8.11.2. Organización de la información

Los participantes también consideraron que la información está organizada de manera clara y comprensible. Comentarios como "Sí, es muy clara y todo está organizado de manera lógica" y "Toda la información está donde debería, sin confusión" reflejan una percepción positiva en cuanto a la estructura de la información. La simplicidad de la página fue especialmente valorada, con observaciones como "Muy comprensible, me gusta la simplicidad en la estructura" y "Excelente organización, el contenido es muy accesible".

8.11.3. Apariencia en dispositivos móviles

La apariencia del sitio en dispositivos móviles fue evaluada positivamente, con comentarios que elogiaron tanto el diseño minimalista como la paleta de colores. Los usuarios expresaron que la página se adapta bien a dispositivos móviles, destacando que "El diseño minimalista y la paleta de colores lucen muy bien en móvil" y "La disposición de los elementos es excelente en dispositivos móviles". En general, la estética del sitio fue calificada como atractiva y adecuada para el entorno móvil.

8.11.4. Facilidad para interactuar con botones y enlaces en móvil

Finalmente, los usuarios reportaron una experiencia positiva al interactuar con botones y enlaces en la versión móvil de la página. Comentarios como "Los botones son fáciles de presionar, muy bien adaptados" y "Todo es accesible y cómodo de usar en el celular" destacan la facilidad de uso en este aspecto. La respuesta de los botones y enlaces fue percibida como intuitiva y cómoda, contribuyendo a una experiencia de usuario satisfactoria.

8.11.5. Análisis de mapas de calor

Para el cuarto *testing*, se utilizaron mapas de calor para analizar las áreas de mayor interés visual en la interfaz, tanto en *desktop* como en móvil. A continuación, se presentan las imágenes de los mapas de calor obtenidos en varias secciones clave de la página:

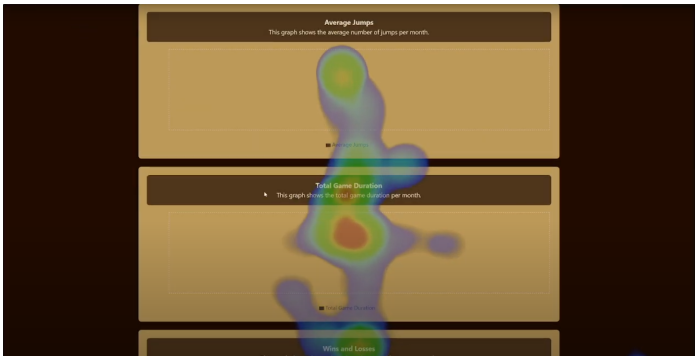

Figura 8.85: Mapa de calor del *dashboard* en escritorio

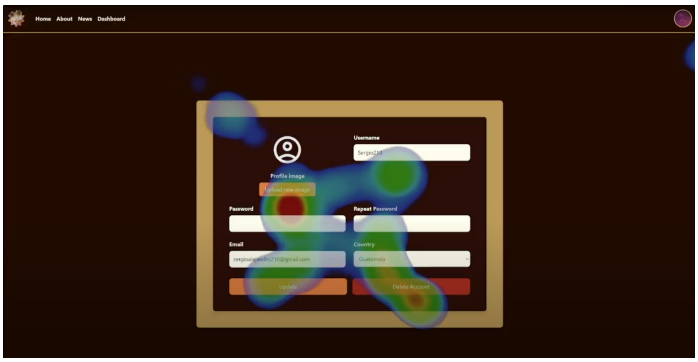

Figura 8.86: Mapa de calor de la edición de perfil en escritorio

Figura 8.87: Mapa de calor del *about* en escritorio

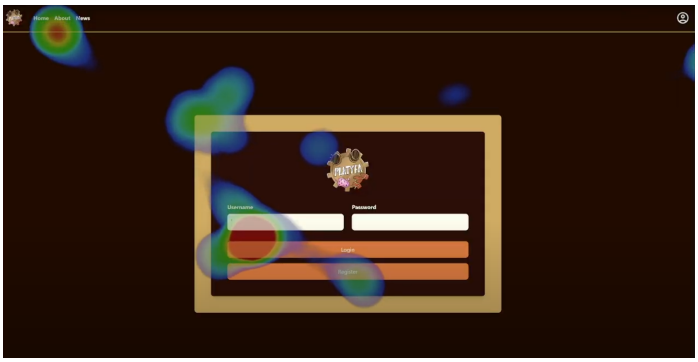

Figura 8.88: Mapa de calor del *login* en escritorio

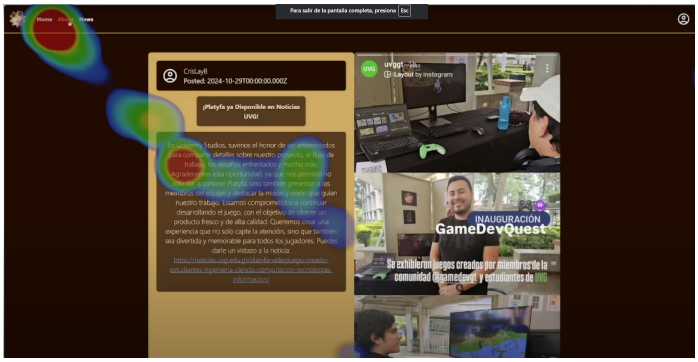

Figura 8.89: Mapa de calor de la sección de noticias en escritorio

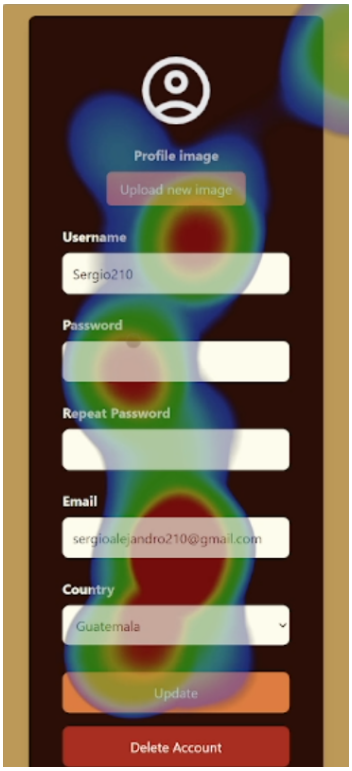

Figura 8.90: Mapa de calor de la edición de perfil en móvil

Figura 8.91: Mapa de calor de la sección de noticias en móvil

Figura 8.92: Mapa de calor del *about* en móvil

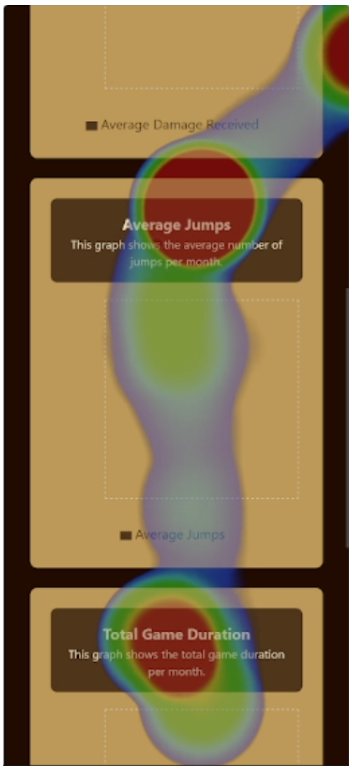

Figura 8.93: Mapa de calor del *dashboard* en móvil

Los mapas de calor indican que las áreas de mayor interés visual incluyen el logo, el menú de navegación en la parte superior, y las secciones de noticias y perfil en el *dashboard*. Esta información permitirá realizar ajustes específicos en el diseño para mejorar la experiencia del usuario.

8.12. Resultados: *Test* de *performance* a la API

En esta fase de *testing* se obtuvieron los siguientes resultados al ejecutar pruebas de carga sobre la API utilizando `k6`. Los resultados incluyen métricas clave como tiempos de respuesta, tasas de transferencia de datos y desempeño general de la API bajo una carga masiva.

- **Escenario de prueba:** 50 usuarios virtuales concurrentes, con una duración máxima de 1 minuto y 50 segundos, incluyendo una fase de finalización gradual de 30 segundos.

Tabla 8.7: Resumen de resultados del *test* de *performance* a la API

Métrica	**Promedio**	**Mínimo**	**Máximo**
`http_req_duration` (Duración de la solicitud)	242.78 ms	79.61 ms	2.71 s
`http_req_waiting` (Tiempo de espera)	223.08 ms	76.31 ms	2.71 s
`http_req_receiving` (Tiempo de recepción)	19.62 ms	4.81 ms	97.17 ms
`http_req_sending` (Tiempo de envío)	76.48 ms	21.4 ms	865.46 ms

Además, se lograron los siguientes resultados adicionales:

- **Peticiones exitosas:** 4266 de 4266, lo que indica una tasa de éxito del 100 %.
- **Datos recibidos:** 2.1 MB a una tasa de 26 kB/s.
- **Datos enviados:** 1.4 MB a una tasa de 18 kB/s.
- **Tasa de iteración:** Promedio de 17.45 iteraciones por segundo.
- **Latencias de conexión:** Mínima de 0 ms, con un promedio de 464.9 µs.

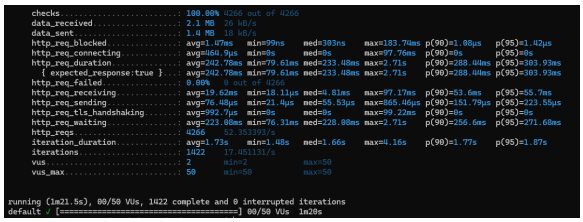

Figura 8.94: Resultados de la primera fase de prueba de *performance* a la API

8.13. Resultados: *Test* final de *performance* a la API

En esta fase de *testing*, se ejecutó una prueba de *performance* final sobre la API, enfocándose en verificar los tiempos de respuesta de los *endpoints* optimizados. Los resultados obtenidos confirman mejoras en la eficiencia de la API tras las optimizaciones implementadas a partir de la fase anterior. A continuación, se detallan las métricas clave obtenidas:

Tabla 8.8: Resumen de resultados del *test* final de *performance* a la API

Métrica	Promedio	Mínimo	Máximo
`http_req_duration` (Duración de la solicitud)	235.21 ms	116.08 ms	838.99 ms
`http_req_waiting` (Tiempo de espera)	217.98 ms	80.75 ms	813.28 ms
`http_req_receiving` (Tiempo de recepción)	17.15 ms	19.98 µs	264.56 ms
`http_req_sending` (Tiempo de envío)	69.25 ms	20.14 µs	719.83 µs

Otros resultados obtenidos en esta fase incluyen:

- **Peticiones exitosas:** 4329 de 4329, con una tasa de éxito del 100 %.
- **Datos recibidos:** 1.4 MB a una tasa de 18 kB/s.
- **Datos enviados:** 847 kB a una tasa de 10 kB/s.
- **Tasa de iteración:** Promedio de 17.68 iteraciones por segundo.
- **Latencias de conexión:** Promedio de 932.71 µs, con un mínimo de 0 ms.

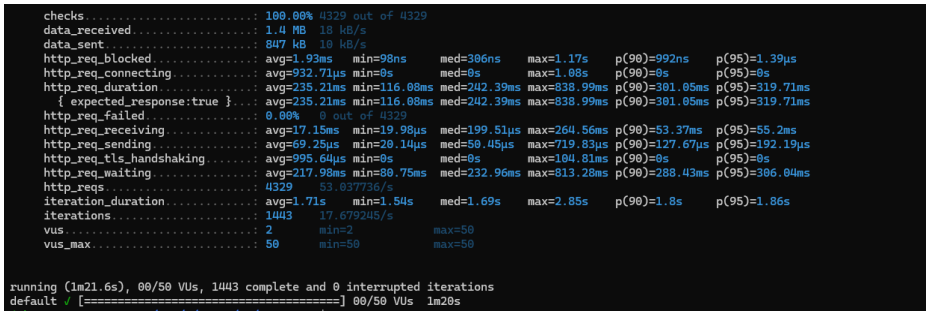

Figura 8.95: Resultados del *test* final de *performance* a la API tras optimización

8.14. Recepción y retroalimentación del público

Los participantes probaron el juego en un computador ya proveído para ellos, se les explicó el contexto del videojuego y se les ofreció utilizar control o teclado. En cualquier momento eran libres de realizar preguntas. Al finalizar la sesión contestaron una encuesta de satisfacción cuyos resultados pueden hallarse a continuación.

8.14.1. Primera sesión de *playtesting*

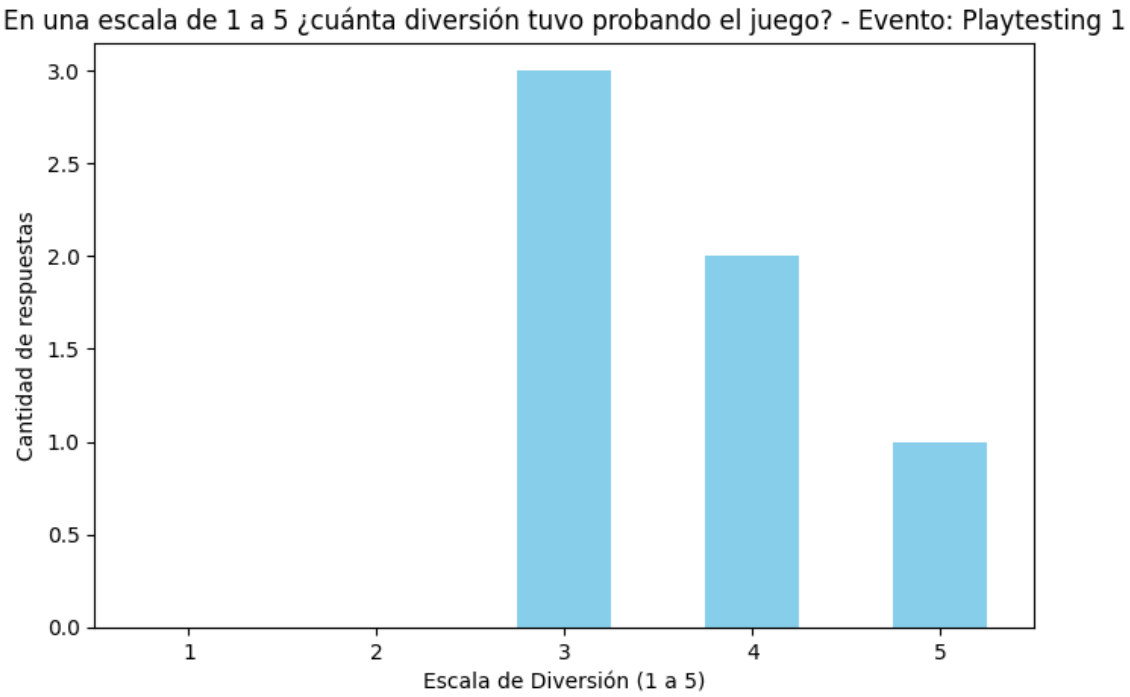

Figura 8.96: Escala de diversión, *Playtesting* 1

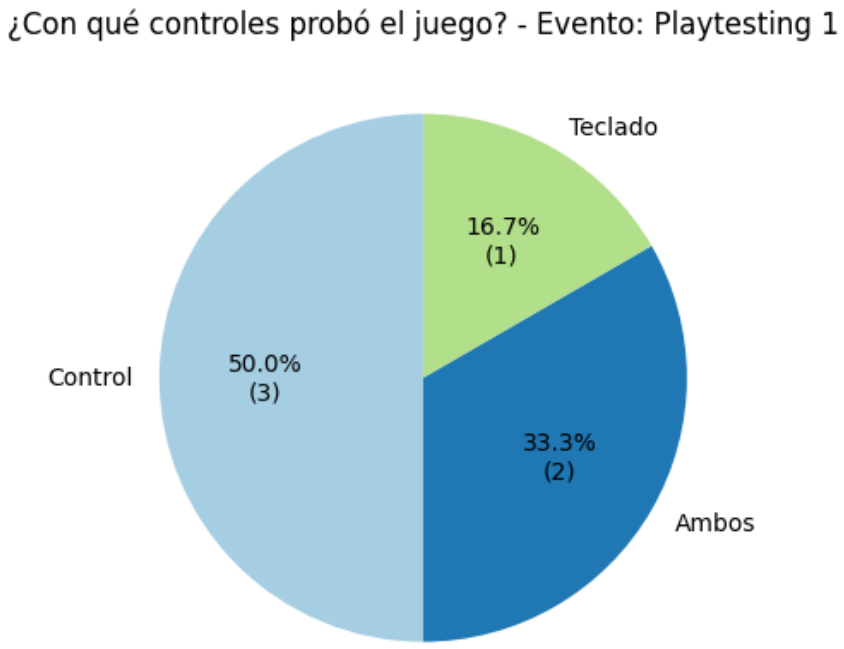

Figura 8.97: Porcentaje de uso de controles, *Playtesting* 1

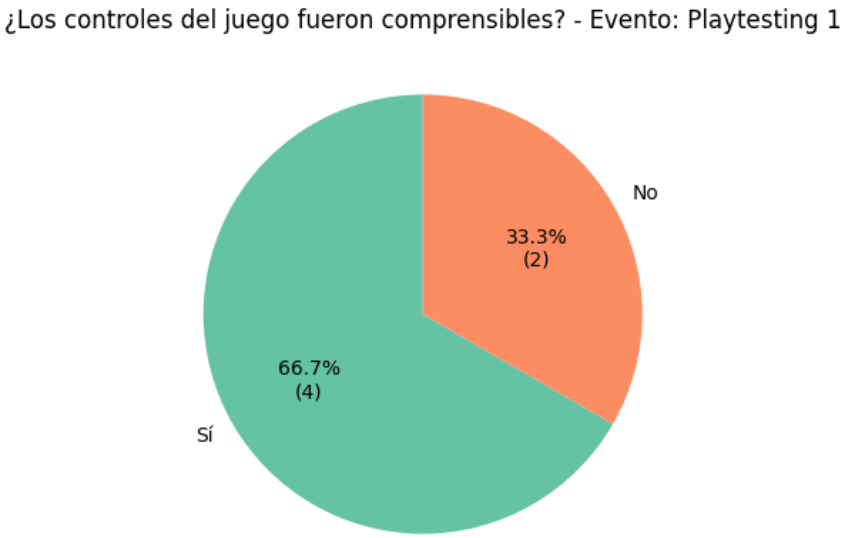

Figura 8.98: Comprensión de controles, *Playtesting* 1

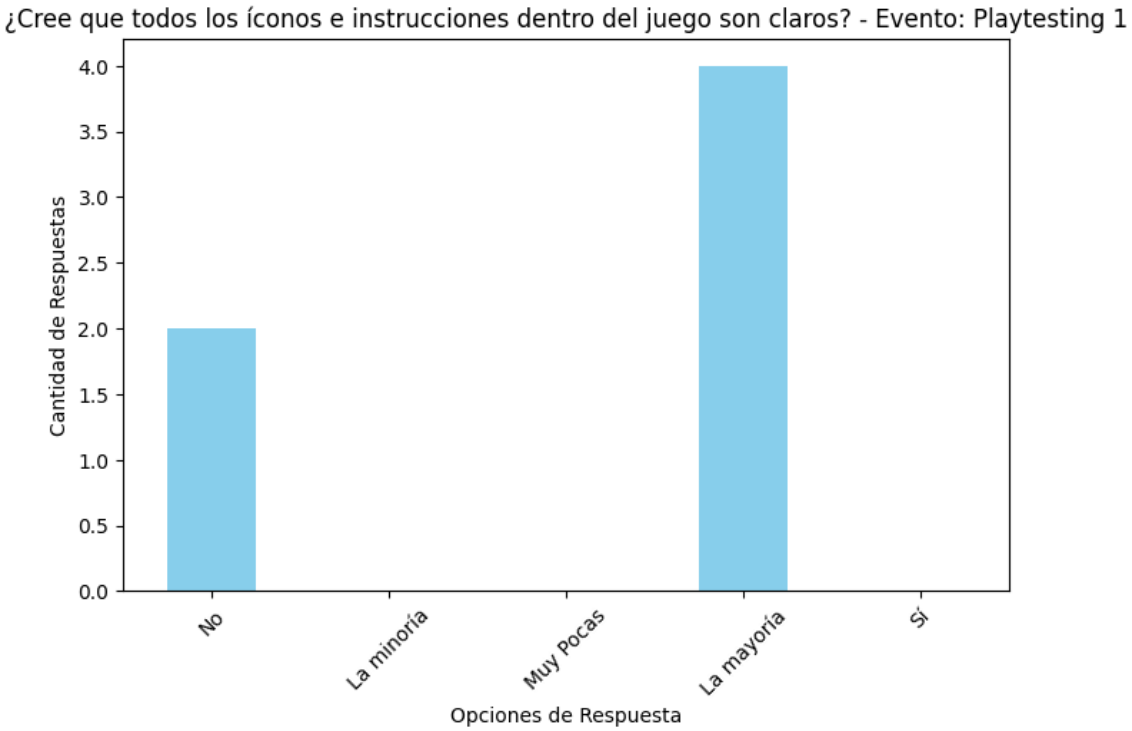

Figura 8.99: Claridad de iconos e instrucciones, *Playtesting* 1

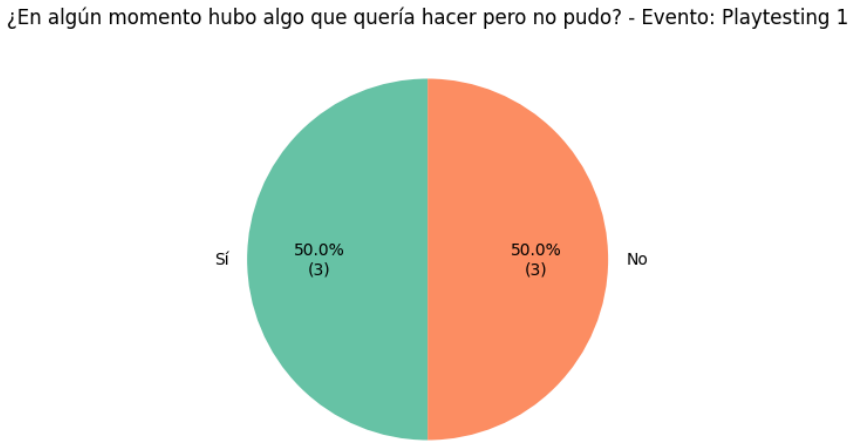

Figura 8.100: Inhabilidad de acción, *Playtesting* 1

Si tu respuesta anterior fue sí, ¿qué fue?: En general, los jugadores presentaron confusión con respecto al modo de cómo apuntar el arma del personaje y les frustró la inhabilidad de derrotar al jefe al final del nivel.

En general, ¿qué cambios le haría al juego? Entre el *feedback* brindado por los jugadores se encuentran ideas como centrar la mira, agregar animaciones, y agregar más sonidos. Existe la función de .ªgacharse"pero no parece ser útil.

8.14.2. Segunda sesión de *playtesting*

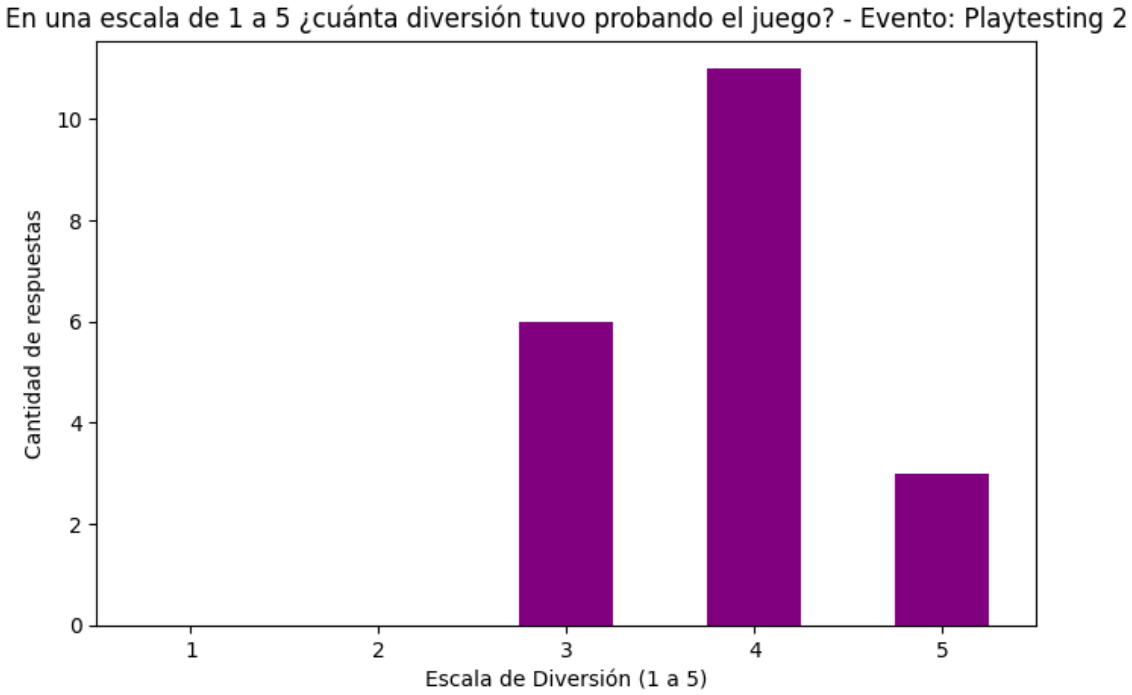

Figura 8.101: Escala de diversión, *Playtesting* 2

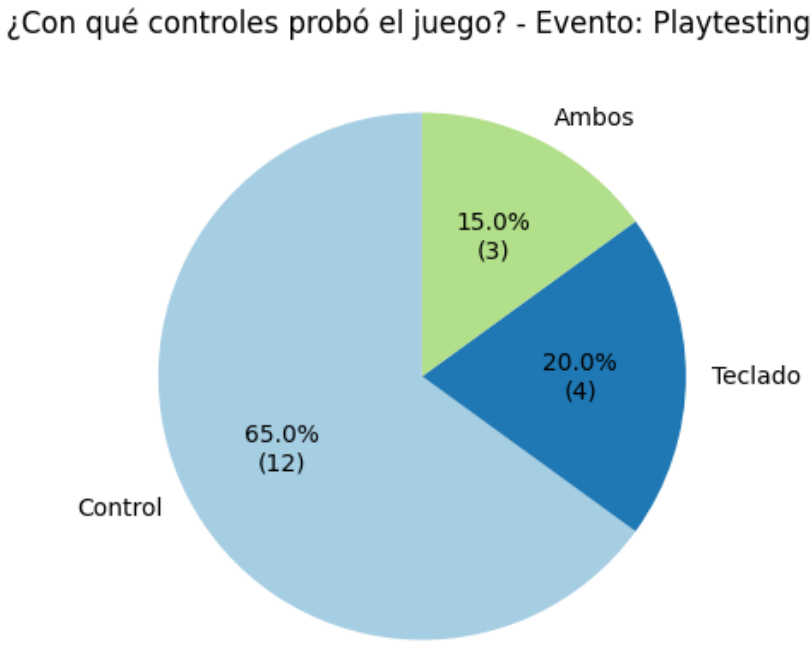

Figura 8.102: Porcentaje de uso de controles, *Playtesting* 2

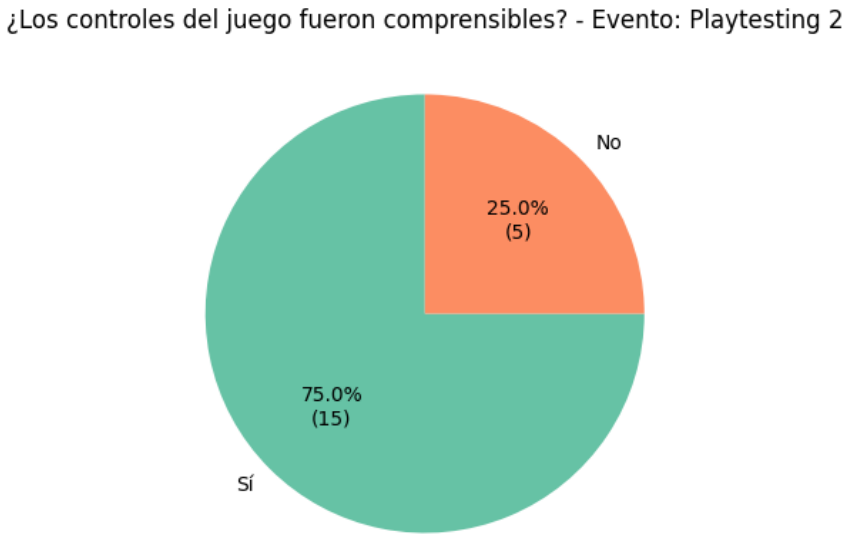

Figura 8.103: Comprensión de controles, *Playtesting* 1

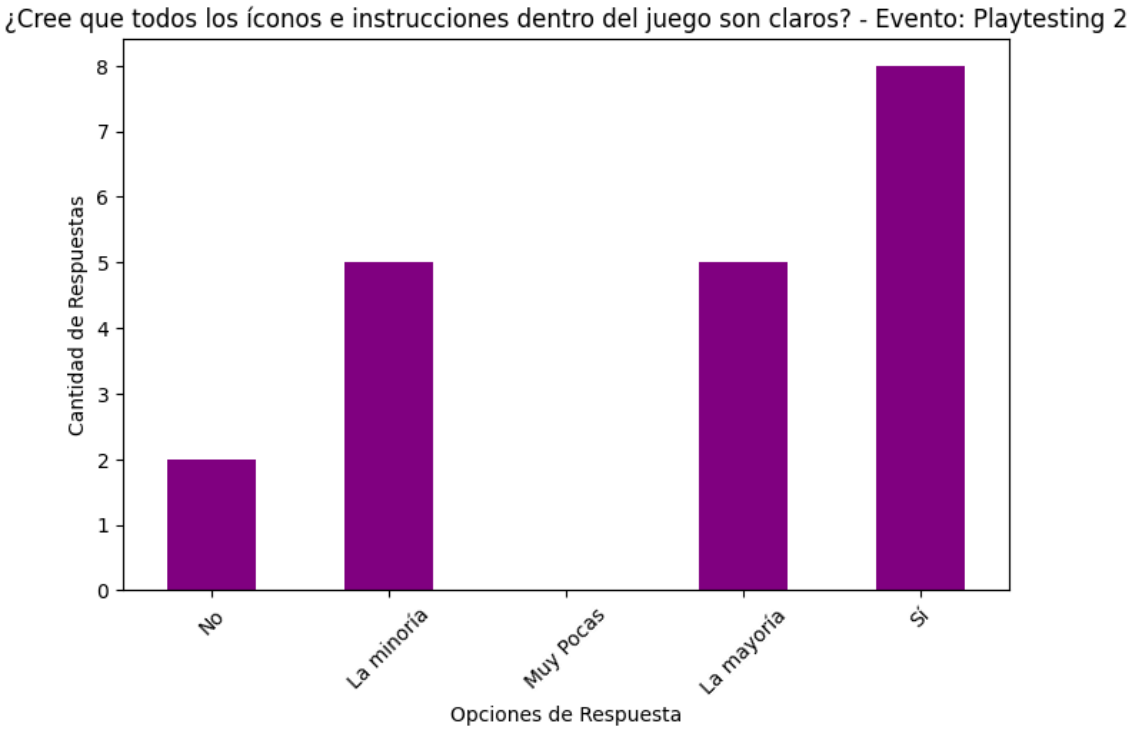

Figura 8.104: Claridad de íconos e instrucciones, *Playtesting 2*

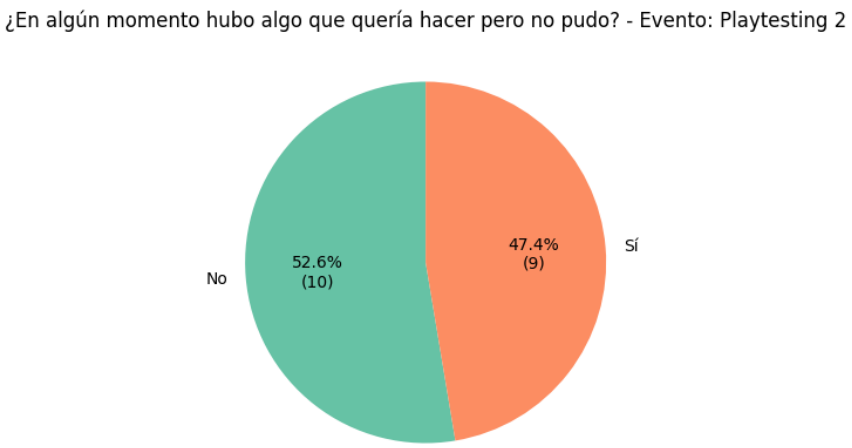

Figura 8.105: Inhabilidad de acción, *Playtesting 2*

Si tu respuesta anterior fue sí, ¿qué fue?: Para la segunda sesión, nuevamente se presentó dificultad con el modo de uso del arma y cómo se apunta con ella, y la comprensión de modo de derrota del jefe fina.

En general, ¿qué cambios le haría al juego? Se recomendó hacer arreglos sobre los puntos de frustración como arreglar la mira y agregar más *feedback* en la batalla contra el jefe. Además también se recomienda agregarle animaciones a los enemigos.

8.14.3. Tercera sesión de *playtesting*

Figura 8.106: Escala de diversión, *Playtesting* 3

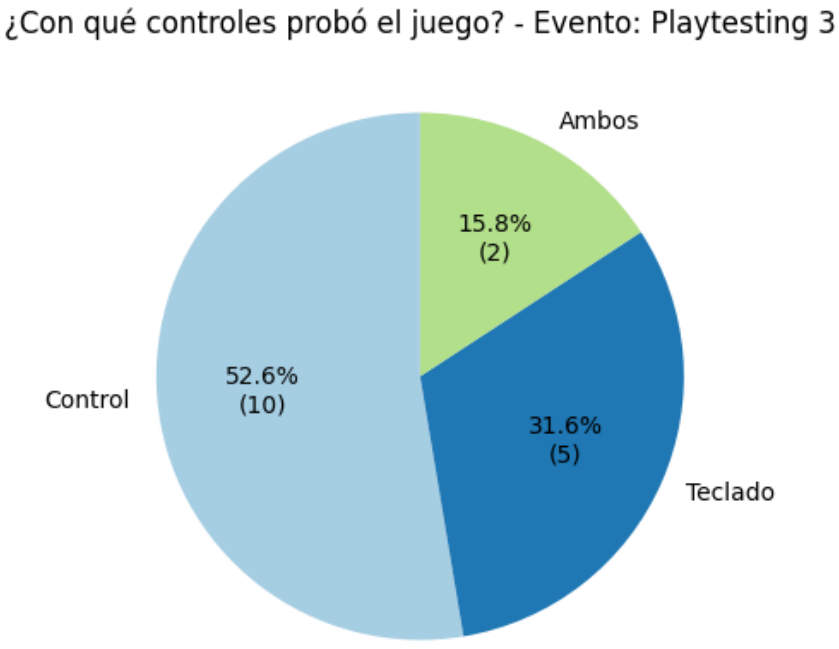

Figura 8.107: Porcentaje de uso de controles, *Playtesting* 3

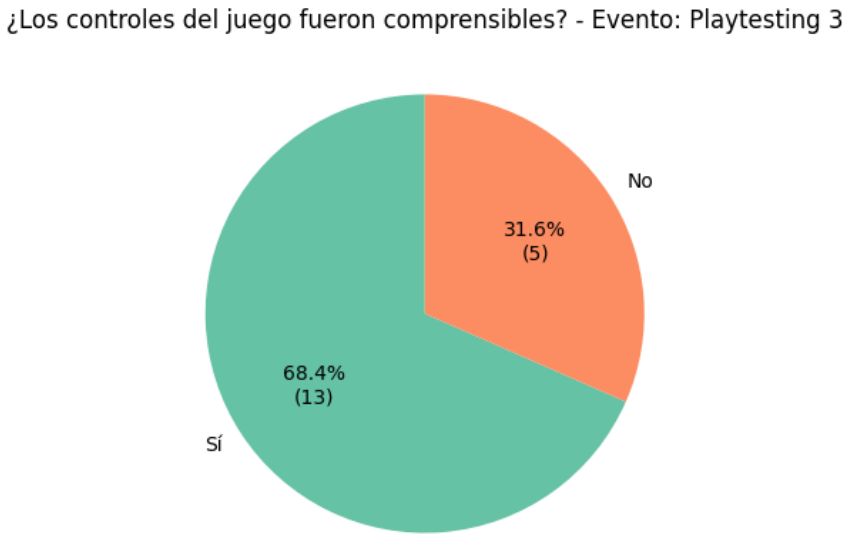

Figura 8.108: Comprensión de controles, *Playtesting* 3

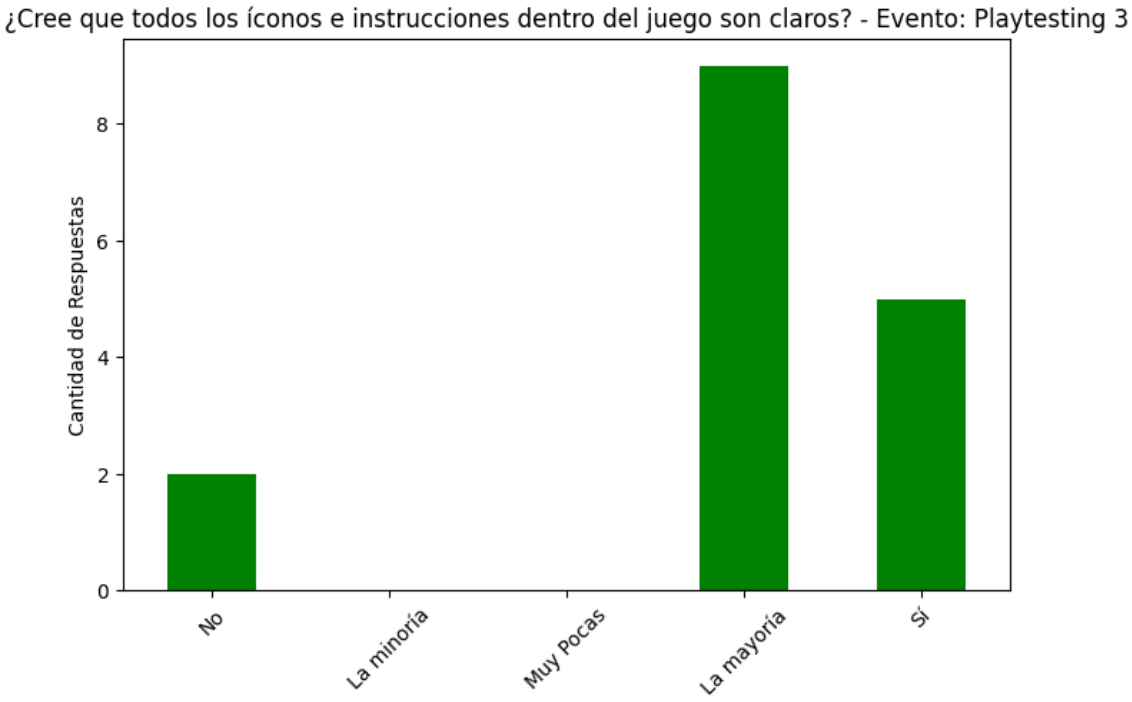

Figura 8.109: Claridad de instrucciones, *Playtesting* 3

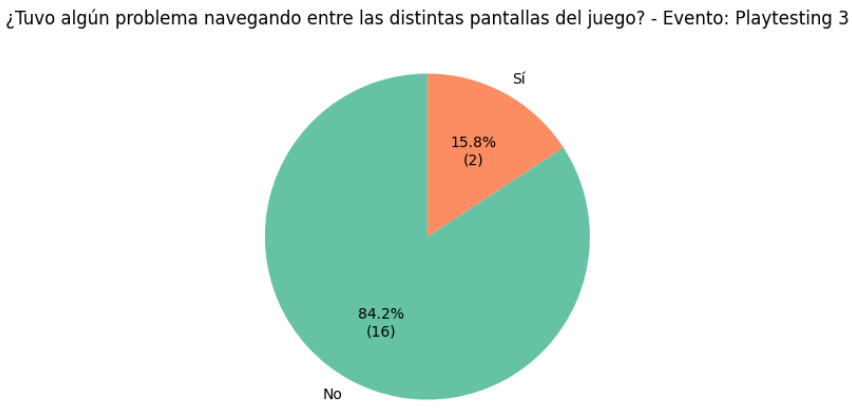

Figura 8.110: Facilidad de navegación, *Playtesting* 3

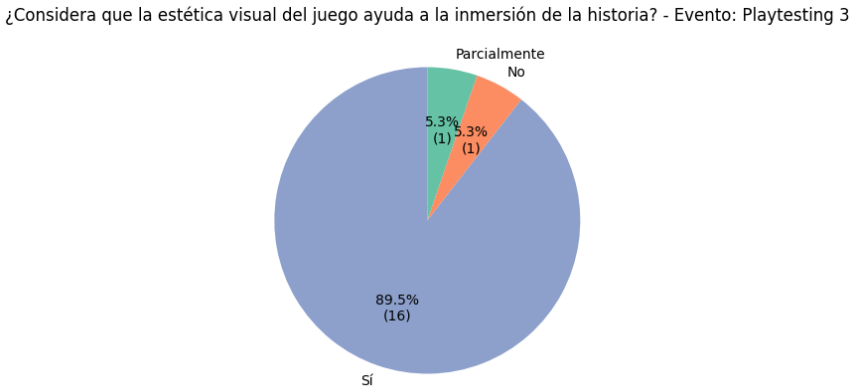

Figura 8.111: Inmersión por estética, *Playtesting* 3

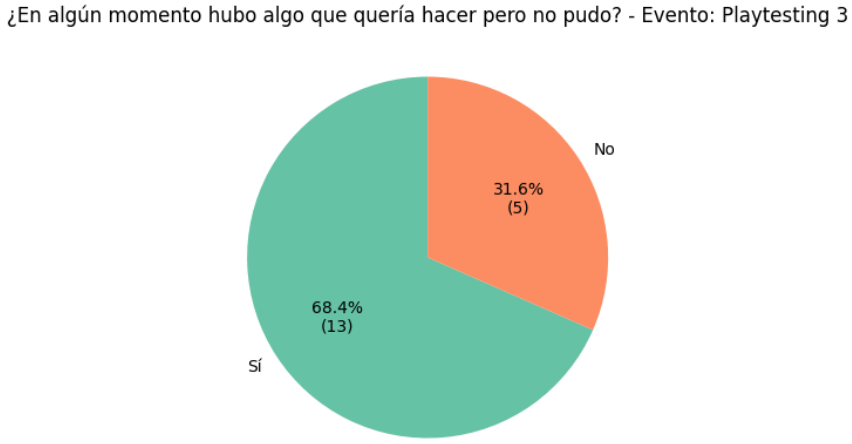

Figura 8.112: Inhabilidad de acción, *Playtesting* 3

Si tu respuesta anterior fue sí, ¿qué fue?: En la tercera sesión presentaron problemas con la mira e instrucciones de cómo derrotar al jefe.

En general, ¿qué cambios le haría al juego? Luego de la sesión de juego, se recomendó por medio de comentarios mejorar los problemas con respecto al punto anterior. Agregar más contexto e instrucciones. Bajarle a la sensibilidad de la cámara o permitir que esta sea ajustable.

CAPÍTULO 9

Análisis de resultados

9.1. UI, UX, GX

Este proyecto buscaba establecer una integración de forma efectiva entre los distintos elementos de UI, UX, y GX dentro del videojuego Platyfa. Se buscaba que cada aspecto de la interfaz y la jugabilidad contribuyese a una experiencia inmersiva, atractiva y divertida. Para lograrlo, se desarrollaron prototipos de la historia, la interfaz, personajes, armas y niveles. Esta, según los resultados (como las gráficas 8.96, 8.101, y 8.106) se obtuvo una recepción positiva del videojuego, brindando una experiencia divertida, y por medio de los elementos visuales como la interfaz y el diseño de los personajes y los modelos 3D se pudo crear una experiencia inmersiva, como lo indica la Figura 8.111 , que le permita a los jugadores vivir el inicio de la historia de Gaus y la accidental revolución de ornitorricos.

El diseño y prototipado de la interfaz de usuario (UI) incluyó la creación de menús principales, pantallas de configuración, pantallas de información, y la HUD, las cuales fueron evaluadas junto al juego durante las sesiones de *playtesting*. Según los jugadores, la mayoría reportó una experiencia clara y fácil de navegar 8.110 lo cual es un indicador positivo en términos de usabilidad.

El objetivo de diseñar y prototipar elementos esenciales para el videojuego de Platyfa; como personajes, narrativa, mecánicas, y niveles; fue alcanzado mediante un enfoque iterativo en el cual se evaluaron y ajustaron cada uno de estos elementos con base a retroalimentación recibida principalmente de manera interna. El personaje principal, Gaus, y los enemigos de tipo dingo, en particular, fueron diseñados para proporcionar una primera impresión fuerte que lograra transmitir el tono y las emociones del juego.

Se logró la colaboración interdisciplinaria para integrar el diseño UI, UX y GX en todas las áreas del desarrollo del juego. La comunicación entre los integrantes del equipo fue crucial para asegurar una correcta implementación para obtener un juego inmersivo y divertido de jugar. En general, la colaboración resultó en una implementación eficiente con la mayoría de módulos dentro del proyecto. Sin embargo, ciertos problemas técnicos en la coordinación con avances con otros módulos interrumpieron el progreso de la implementación de las mejoras. Debido a ellas, ciertas áreas de frustración de los jugadores, tales como arreglos en la cámara y la mirilla del arma no fueron realizados a tiempo.

La recopilación y el análisis del *feedback* de los jugadores fue esencial para identificar las nece-

sidades y expectativas del jugador. A parte de ser una herramienta fundamental para poder ver el juego desde ojos ajenos al mismo y notar con mayor facilidad problemas de diseño y código. Las preguntas diseñadas para medir el nivel de diversión, claridad de instrucciones e inmersión (tales como las que se encuentran en las gráficas 8.96, 8.99, 8.111) demuestran que en términos generales, Platyfa fue recibido de manera positiva como un juego entretenido.

9.2. Dificultad de los enemigos

La mayoría de los jugadores (73.3 %) consideró que la dificultad de los enemigos era adecuada, lo cual valida la efectividad del balance inicial implementado en la programación. No obstante, un 20 % expresó que el nivel de dificultad debería incrementarse. Este grupo, posiblemente conformado por jugadores con mayor experiencia, sugiere la necesidad de implementar un sistema de dificultad adaptable.

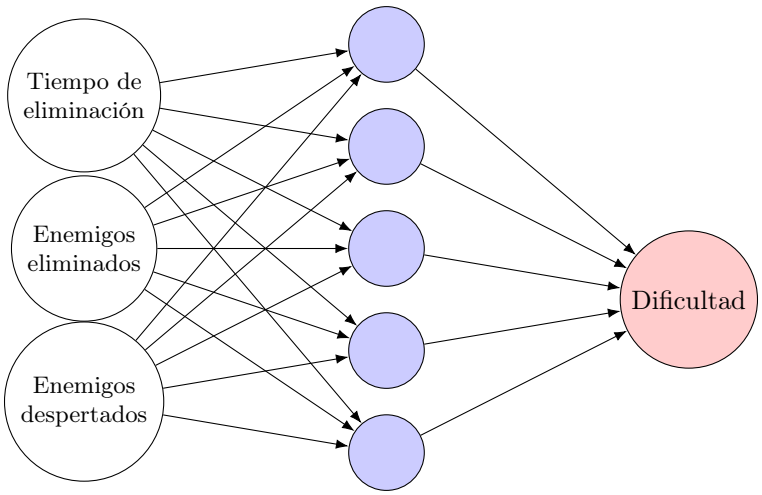

Figura 9.1: Arquitectura de la red neuronal para ajustar la dificultad en el segundo nivel con los enemigos búhos

Un enfoque sugerido para mejorar la dificultad es el uso de una red neuronal que modifique dinámicamente el comportamiento de los enemigos de acuerdo con el desempeño del jugador, como se muestra en la Figura 9.1. Esto permitiría que el juego ajuste la dificultad en tiempo real, adaptando los retos para mantener la experiencia desafiante y atractiva para una gama más amplia de jugadores.

9.3. Velocidad y comportamiento de los enemigos

El análisis de la velocidad de ataque mostró que mientras el 68.4 % la consideró adecuada, un 31.6 % recomendó ajustes. La percepción de una velocidad de ataque adecuada indica un balance inicial aceptable, aunque introducir variabilidad en la velocidad en función del contexto (por ejemplo, la proximidad del jugador o su nivel de habilidad) podría enriquecer la experiencia de combate.

En cuanto al comportamiento, el 63.2 % de los jugadores sugirió cambios, lo que resalta la necesidad de una mayor diversidad en las estrategias enemigas. Un enfoque viable sería implementar tácticas adaptativas en los enemigos, permitiéndoles ajustar sus acciones según el estilo de juego del jugador. Esto no solo aumentaría la dificultad percibida sino que también mejoraría la inmersión y

reduciría la previsibilidad de los combates, factores críticos en la experiencia de juego. Para mayor información respecto a los comentarios de los usuarios, dirigirse al Anexo 13.1

9.4. Bugs eelacionados con los enemigos

Casi la mitad de los jugadores (47.4 %) reportó la presencia de *bugs* en el comportamiento de los enemigos, lo cual es un área de oportunidad crítica. Estos errores, detallados en la Tabla 9.1, afectan negativamente la jugabilidad y pueden romper la inmersión. Un plan de acción podría incluir la implementación de un protocolo de pruebas más exhaustivo, acompañado de un sistema de retroalimentación para capturar y priorizar *bugs* reportados por los jugadores.

Específicamente, los problemas de detección y reacción de los enemigos indican que los algoritmos de percepción y respuesta pueden beneficiarse de una revisión y optimización. Además, los errores críticos, como los enemigos que fallan al cambiar de fase o se comportan erráticamente, pueden corregirse mediante pruebas unitarias y de integración orientadas a la estabilidad en escenarios de combate.

Bug	**Descripción**	**Frecuencia de aparición**
Bug 1	Al pasar a la segunda fase, el Boss no atacaba hasta eliminar los timones	Alta
Bug 2	Problemas de detección de los enemigos	Media
Bug 3	El Boss se cayó de frente	Baja

Tabla 9.1: Principales *bugs* reportados en el comportamiento de los enemigos

9.5. Cantidad de enemigos

Finalmente, la cantidad de enemigos fue considerada adecuada por el 63.2 % de los 55 jugadores, mientras que el 36.8 % sugirió una mayor densidad de enemigos. Esta diferencia de opiniones sugiere la necesidad de diseñar niveles que puedan adaptarse al estilo de juego de cada jugador. Una opción es implementar configuraciones de juego que permitan al jugador ajustar la densidad de enemigos o diseñar zonas de alta densidad en ciertos niveles para aquellos que buscan un desafío más intenso.

9.6. Modelos 3D

Con el fin de lograr los objetivos planteados al inicio de este trabajo, se pasaron formularios en la prueba del videojuego en los cuales iban incluidas preguntas que ayudarán a comprobar la hipótesis. Primero se hicieron preguntas antes de realizar la prueba del juego, para conocer los conocimientos previos al proyecto:

9.6.1. ¿Qué juegos has jugado recientemente que te impresionaron por su modelado 3D?

Las respuestas a esta pregunta revelan que los jugadores están familiarizados con las experiencias visuales de alta gama de los juegos 3D, los participantes mencionaron a *Elden Ring*, *God of War*, *Red Dead Redemption 2* y *Zelda: Tears of the Kingdom*. Teniendo en cuenta la buena calidad gráfica y la atención al detalle en los modelos de este lote de juegos indican un aprecio por los aspectos

visuales, a través del modelado 3D en un juego. Dicha experiencia y conocimiento previo con juegos visualmente atractivos dan inicio para validar la relevancia del modelado 3D para la creación de experiencias inmersivas y atractivas en videojuegos.

9.6.2. ¿Tienes experiencia previa en el modelado 3D o el desarrollo de videojuegos?

La mayoría de los participantes informaron que no tienen experiencia previa en modelado 3D o desarrollo de juegos, de las 26 respuestas conseguidas, el 65.4 % fueron los que respondieron que no. Lo que demuestra que muy pocos tienen antecedentes técnicos. Sin embargo, algunos de los participantes mencionaron que tenían experiencia en el uso de herramientas como *Blender* y *Unity*, pero a nivel básico. Por lo tanto, indica un inicio muy temprano en el uso de *Blender* y otras herramientas en la comunidad local. El registro de usuarios con experiencia anterior en *Blender*, independientemente de su nivel amateur, resalta que este proyecto puede eventualmente contribuir a transmitir las capacidades adquiridas para modelos 3d en videojuegos a un nivel mucho más amplio y profundo, gracias a un ejemplo local, especialmente para aquellos que no poseen experiencia en el asunto.

9.6.3. ¿Qué esperas encontrar en un juego con un protagonista como un ornitorrinco chef e inventor?

Las respuestas revelan que con 55 participantes el 60 % de los jugadores (33 de 55) esperan una gran creatividad visual y detalle en un videojuego de este tipo, que logre transmitir algo sobre el personaje y el entorno del protagonista Así, muchos pensaron en el arma, muchos otros pensaron en inventos relacionados con la cocina, combinando el tema de la cocina con la aventura. Estas respuestas muestran que los participantes esperaban un diseño tridimensional que fuera expresivo, innovador y al mismo tiempo coherente con la narrativa del juego. La anticipación previa a la creación de estos elementos indica que un modelo 3D no solo debe ser visualmente atractivo, sino que también debe tener una historia que contar, y por lo tanto, uno de los elementos más importantes serían los personajes y entornos que respaldan el carácter y el contexto del juego a través de sus elementos visuales.

Después de dejar a los participantes navegar libremente entre las diferentes pantallas de menús del juego y pasar el primer nivel del juego de *Platyfa* se les paso una segunda encuesta que contenía preguntas relacionadas después de la prueba de juego:

9.6.4. En una escala de 1 a 5 ¿los modelos 3D son visualmente atractivos y están bien diseñados? (considerando aspectos como la estética y la coherencia con el estilo del juego)

Esta pregunta resultó con muy buenos resultados, el 80 % de las 55 respuestas, evaluaron los modelos 3D con 4 y 5 puntos de calificación, lo que marcó que los modelos son muy atractivos y coherentes con el estilo del juego. Esto solo confirma que los recursos realizados en *Blender* están realmente a la altura de la alta deseabilidad estética en un videojuego. Esto se relaciona mucho con el objetivo del trabajo de hacer modelos 3D de alta calidad que los jugadores puedan visualizar de manera satisfactoria. Los comentarios también mostraron que a la mayoría de los participantes les

gustó la estética y la coherencia gráfica del juego, notando que el modelado 3D ciertamente afectó la percepción de las personas sobre su apariencia general.

9.6.5. ¿Cree que el modelado 3D mejoró su experiencia de juego?

Aquí, el 84 %, de los jugadores respondieron que sí, para quienes la experiencia de juego mejoró con la ayuda del modelado 3D dejaron comentarios que se refirieron al estilo realismo cartoon, inmersión y atractivo visual *agregada* a la experiencia mediante el modelado 3D. Algunos jugadores notaron que la interacción es mucho mejor con detalles y calidad de personajes y objetos.

9.6.6. Si respondió "sí", ¿puede proporcionar ejemplos específicos de cómo el modelado 3D influyó en tu experiencia?

Para 20 participantes de está pregunta mencionaron como el protagonista y los detalles de los objetos hicieron que su inmersión fuera más fácil de relacionar, con comentario como "*El darle profundidad al juego me hace sentir más espacioso el juego*", "*Creo que el modelado 3D ayuda mucho al combate, se siente mas fluido y uno puede visualizar los ataques de los enemigos de forma mas fácil.*" y muchos más. Estos resultados confirman la hipótesis del trabajo, de que un buen modelo 3D no solo tiene como objetivo mejorar la estética del juego, sino también mejorar la calidad general de la experiencia. Por lo tanto, estas mejoras elevan el producto a una mayor inmersión y una mejor calidad de disfrute desde el punto de vista del diseño hermoso. Algunos de estos participantes que mencionaron ejemplos específicos informaron que el modelo 3D había hecho que las acciones del personaje fueran claramente visuales y mejoró la comprensión de su ambiente.

9.6.7. ¿Cree que el modelado 3D aporta un impacto significativo a la historia y experiencia del juego, en comparación con un posible diseño en 2D?, Justifique

En el caso de esta pregunta, el 72 % de los jugadores está de acuerdo, pensando que el valor del modelado 3D sería mucho mayor que el de un diseño 2D. Las razones que dieron los encuestados son ilustrativas de cómo el 3D permite un detalle mucho más rico en la representación visual que también reforzará la narrativa y la inmersión del juego. Algunos jugadores sintieron que el personaje 3D realmente los ayudó, hasta cierto punto, a dibujar una imagen clara de la personalidad del protagonista y su mundo contextual; aunque para otros, los personajes se volvieron "vivosz "tangibles.ª través de la profundidad visual proporcionada por el 3D. Estos comentarios subrayan el enfoque del trabajo en el modelado 3D, que confirma que la elección de *Blender* para el desarrollo de los personajes y los escenarios detallados es una de las respuestas directas a la percepción del jugador sobre la importancia del 3D para obtener la experiencia más completa y significativa.

9.6.8. En una escala de 1 a 10 ¿el modelado 3D aumentó su nivel de inmersión en el juego? Justifique

El 78 % de los jugadores le dieron una puntuación entre 8 y 10. Un vistazo rápido a los comentarios muestra que a la mayoría de los jugadores les gustó mucho el efecto positivo que el modelado 3D le dio al aspecto gráfico. Muchos justificaron que esto los atrajo al mundo del juego a partir de los detalles de los personajes y los objetos. Una buena parte de los encuestados afirmó que la tridimensionalidad del espacio, junto con las texturas y los detalles de los modelos, los ayudó a ïnmersos.en la experiencia del juego y a sentirse como si fueran participantes activos de la historia.

Otros efectos 3D, como la profundidad de campo o el nivel de detalle de los elementos interactivos, aumentaron significativamente su sensación de estar presentes en el espacio real. Estos comentarios proporcionaron la justificación de la necesidad de tener un diseño tridimensional detallado porque el modelado 3D es el que mejor contribuye a una experiencia poderosa dentro del juego. De esta manera se logra el objetivo del trabajo de lograr una experiencia máxima y una representación visual poderosa del juego.

Toda está información fue fundamental para afirmar el exitos de todos los objetivos específicos para el modelado 3D. La creación de *UVs* adecuados, el control de la topología y las técnicas de gestión del flujo de bordes han sido indispensables para afectar la optimización tanto de la animación como del rendimiento dentro del entorno del juego. Estos métodos permitieron reducir el recuento de polígonos sin ninguna pérdida en la calidad visual necesaria, logrando así el objetivo de modelos de polígonos bajos efectivos y eficientes, manteniendo al mismo tiempo una alta fidelidad visual. El uso correcto de las normales y la optimización en *Blender* ayudó para un rendimiento suave.

Al mismo tiempo, ciertos métodos de modelado, como el modelado de cajas y los bucles de bordes para recibir curvas suaves y construir a partir de figuras primitivas, permitieron la creación de personajes y objetos estilizados, aunque visualmente coherentes y tratados al nivel que satisface la calidad exigida por un videojuego. Esto aseguró un equilibrio entre la estética atractiva y la estructura geométrica eficiente para aliviar la integridad visual sin una carga pesada en el rendimiento del juego. Todo este soporte de implementación de principios técnicos fue clave para el desarrollo de una experiencia atractiva y satisfactoria en línea con la visión del proyecto y reafirma la competencia de *Blender* como herramienta principal en el desarrollo de contenido visual de alta calidad.

9.7. Animación

La implementación de los sistemas de *rigging* y de animación demostró ser una herramienta clave para alcanzar los objetivos de diseño planteados. El *rigging* de cada personaje se llevó a cabo con un enfoque detallado, creando un esqueleto que otorgara flexibilidad y precisión en el control de las extremidades, la cola y las articulaciones principales. Este enfoque permitió que con la ayuda de herramientas como *AutoRig-Pro* los movimientos de cada personaje fueran fluidos y naturales, facilitando la implementación de diversas animaciones sin que se produjeran distorsiones en la malla. Al garantizar un control preciso, el *rigging* resultó esencial para mantener la coherencia visual y la integridad de los personajes dentro del entorno del juego, especialmente en situaciones complejas como los movimientos de combate o en las transiciones de estado en el árbol de animaciones.

El sistema de animaciones fue configurado cuidadosamente para lograr transiciones suaves y naturales entre diferentes acciones, como caminar, correr, saltar y disparar. Se implementó un árbol de animaciones organizado en nodos jerárquicos que facilitaron la transición entre acciones simples y complejas, logrando evitar interrupciones bruscas entre ellas. Cada nodo en el árbol de transiciones permitía que las animaciones se sucedieran de manera orgánica, contribuyendo a una experiencia de juego fluida y sin interrupciones. La estructura jerárquica del árbol no solo optimizó la progresión de movimientos, sino que también permitió una transición eficiente entre las animaciones de combate y las de interacción con el entorno, como cuando el personaje está en reposo o en movimiento activo como se observan en la sección de resultados.

La integración del sistema de animación con los *scripts* de control resultó también eficaz. Se programaron *scripts* que coordinaban las animaciones de acuerdo con las acciones del jugador y las situaciones contextuales en el juego. Esta integración aseguró que el personaje respondiera correctamente a las entradas del usuario y se moviera en sincronía con los eventos en pantalla, mejorando tanto la jugabilidad como la inmersión del jugador. Además, el *rigging* permitía que estas animaciones se adaptaran a distintos estados sin problemas técnicos, manteniendo la estabilidad de las transiciones y la coherencia visual del personaje.

Paralelamente, gracias a los resultados obtenidos en el *GameTesting*, estos indicaron que las animaciones implementadas lograron un alto nivel de aceptación entre los participantes, cumpliendo de manera exitosa con los objetivos de diseño planteados. En general, los puntajes promedio para las preguntas en escala se situaron consistentemente en el rango superior: en una escala del 1 al 10, la experiencia inmersiva recibió un promedio de 7.6, y la satisfacción con las transiciones de animación alcanzó un 8.0. Esto sugirió que las animaciones no solo proporcionaron un nivel de inmersión adecuado, sino que también cumplieron con las expectativas de continuidad y fluidez en los movimientos, un aspecto fundamental en la narrativa interactiva del juego.

Además, el 80 porciento de los participantes consideró que las animaciones lograron transmitir eficazmente la narrativa de "Gaus y la Revolución Ornitorense", lo que indica que los elementos visuales contribuyeron significativamente a la comprensión de la historia y a la conexión emocional con los personajes. La percepción positiva de los principios de animación aplicados (76 porciento los notó) y la valoración general del atractivo visual de las animaciones (promedio de 7.3) demuestran que las técnicas de anticipación y encogimiento, entre otras fueron efectivas, logrando una integración coherente con el entorno caricaturesco y de ciencia ficción. Por último, la calificación promedio de 4.2 en cuanto a la credibilidad de las animaciones en el contexto del juego y el alto puntaje (4.3) en la mejora de comprensión de la trama resaltan el éxito de las animaciones en su función narrativa y visual. Estos resultados reflejan que las animaciones contribuyeron de manera significativa al atractivo visual y a la experiencia general del juego, superando las expectativas medias y validando la efectividad de la implementación para generar una experiencia inmersiva y visualmente coherente.

Por otra parte, un aspecto adicional del análisis de resultados se centra en el cumplimiento de los principios de animación fundamentales, los cuales fueron integrados en diversas áreas del sistema de animaciones. El principio de “estirar y encoger” fue implementado en las animaciones de salto, donde los personajes levantan los brazos y flexionan las rodillas al caer, logrando un cambio dinámico en la postura y añadiendo fluidez a la animación. La anticipación fue aplicada en la animación de combate, donde el personaje adopta una posición estable para disparar, reforzando la intención de la acción y creando una expectativa visual clara. El principio de “puesta en escena” se mantuvo en las animaciones, evitando detalles superfluos que pudieran desviar la atención del jugador y logrando así que el foco visual permaneciera en la acción principal. Así como cada animación se hizo con el principio de pose a pose para alcanzar las animaciones obtenidas como se puede observar en resultados.

Principios adicionales como la acción continuada y superpuesta fueron incorporados en la animación de los personajes secundarios, como en el caso del búho, cuyas patas cuelgan para simular el efecto de la gravedad. La exageración se utilizó especialmente en las animaciones de *KO*, donde el movimiento abrupto y ampliado en la caída añade un toque caricaturesco y refuerza la narrativa de ciencia ficción. Se incorporaron también los principios de acelerar y desacelerar, evidentes en la animación de salto extendido, donde el personaje cae a una velocidad más lenta en la parte final del salto. El movimiento en arcos se hizo visible en la animación de la cola del personaje principal, que sigue un patrón de arco al caminar, añadiendo naturalidad a la acción y manteniendo el atractivo visual. Como acción secundaria, se animaron las orejas y cola del Dingo, agregando vivacidad y profundidad a los personajes, mientras que el atractivo se consolidó en el estilo y ritmo de las caminatas, contribuyendo a una estética coherente y visualmente llamativa.

Por ende, los resultados obtenidos reflejan un alto nivel de efectividad en la implementación de los sistemas de *rigging* y de animación. Estos sistemas no solo cumplieron con los principios fundamentales de la animación, sino que también contribuyeron significativamente a la inmersión y la narrativa del juego, como lo reflejan los altos puntajes en satisfacción general y conexión narrativa. La calificación promedio de 4.2 en cuanto a la credibilidad de las animaciones y el puntaje de 4.3 en la mejora de la comprensión de la trama resaltan el éxito de las animaciones en su función narrativa y visual. En general, los elementos implementados no solo superaron las expectativas medias, sino que también validaron la efectividad del enfoque en crear una experiencia inmersiva y visualmente coherente que enriquece la interacción del jugador con el mundo del juego.

9.8. Web

9.8.1. Evaluación de la usabilidad de la web

El objetivo específico de "asegurar una experiencia de usuario sencilla y efectiva" se abordó mediante un análisis constante de la usabilidad a lo largo de varias iteraciones de testing. En cada fase, se midieron los tiempos de realización de tareas y se recogieron comentarios de los usuarios. Para evaluar el avance en la usabilidad, se compararon los tiempos promedio de cada fase de testing.

Tabla 9.2: Tiempos promedio de tareas en desktop y móvil entre tercer y final testing

Versión	Tarea	Tercer testing (s)	Testing final (s)	Reducción (%)
Desktop	Tarea 1	9.44	7.84	16.9 %
Desktop	Tarea 2	8.86	7.14	19.4 %
Desktop	Tarea 3	7.16	5.94	17.1 %
Desktop	Tarea 4	9.50	7.21	24.1 %
Desktop	Tarea 5	9.08	5.97	34.2 %
Móvil	Tarea 1	11.52	9.17	20.4 %
Móvil	Tarea 2	9.94	7.41	25.4 %
Móvil	Tarea 3	7.73	6.36	17.7 %
Móvil	Tarea 4	10.73	7.83	27.0 %
Móvil	Tarea 5	8.62	6.73	21.9 %

Como se observa en la tabla 9.2, los tiempos promedio de cada tarea disminuyeron significativamente entre el tercer *testing* y el *testing* final, tanto en la versión *desktop* como en la versión móvil. En *desktop*, las reducciones de tiempo oscilaron entre el 16.9 % y el 34.2 %, siendo la Tarea 5 la que mostró la mayor mejora con una reducción del 34.2 %, pasando de 9.08 segundos a 5.97 segundos. En móvil, las mejoras variaron entre el 17.7 % y el 27.0 %, destacando la Tarea 4 como la que presentó la mayor reducción, con un 27.0 %, al disminuir de 10.73 segundos a 7.83 segundos.

9.8.2. Análisis estadístico: Prueba T para medidas repetidas

Para evaluar la efectividad de las optimizaciones realizadas en la página web, se realizó una prueba t de muestras pareadas para comparar los tiempos de cada tarea entre el tercer *testing* y el *testing* final. Esta prueba estadística permite determinar si las diferencias en los tiempos de realización de las tareas son significativas, proporcionando evidencia cuantitativa de la mejora en la experiencia de usuario.

La hipótesis nula (H_0) establece que no hay diferencia significativa en los tiempos de tarea entre el tercer *testing* y el *testing* final. La hipótesis alternativa (H_a) indica que los tiempos en el *testing* final son menores debido a las optimizaciones implementadas.

9.8.3. Resultados de la prueba T para cada tarea

A continuación, se presentan los resultados de la prueba t de muestras pareadas para cada tarea:

Tabla 9.3: Resultados de prueba t para reducción de tiempo en tareas (desktop y móvil)

Versión	Tarea	Prom. Tercer testing (s)	Prom. Testing Final (s)	Dif. Prom. ($\bar{d}$)	Desv. Est. (s_d)	Valor t
Desktop	Tarea 1	9.44	7.84	1.60	2.20	1.68
Desktop	Tarea 2	8.86	7.14	1.72	2.10	1.98
Desktop	Tarea 3	7.16	5.94	1.22	1.45	2.25
Desktop	Tarea 4	9.50	7.21	2.29	2.05	2.75
Desktop	Tarea 5	9.08	5.97	3.11	2.50	3.31
Móvil	Tarea 1	11.52	9.17	2.35	2.60	2.55
Móvil	Tarea 2	9.94	7.41	2.53	2.10	3.01
Móvil	Tarea 3	7.73	6.36	1.37	1.95	1.76
Móvil	Tarea 4	10.73	7.83	2.90	2.20	3.49
Móvil	Tarea 5	8.62	6.73	1.89	1.60	3.13

9.8.4. Interpretación de resultados

Los resultados de la prueba t para medidas repetidas muestran los siguientes hallazgos clave sobre la reducción de tiempos en las tareas entre el Tercer Testing y el Testing Final, como se resume en la Tabla 9.3:

- **Tareas 1 a 5 en *desktop* y móvil**: Las reducciones en los tiempos de algunas tareas son estadísticamente significativas ($p < 0.05$) tanto en la versión Desktop como en la versión Móvil, mientras que en otras no alcanzaron la significancia requerida. En *desktop*, las tareas 4 y 5 mostraron una reducción significativa en el tiempo (valores $t = 2.75$ y $t = 3.31$, respectivamente), indicando que estas tareas se beneficiaron particularmente de las optimizaciones implementadas. En la versión Móvil, las tareas 1, 2, 4 y 5 lograron reducciones significativas en el tiempo (valores t de 2.55, 3.01, 3.49 y 3.13, respectivamente). Este hallazgo sugiere que las optimizaciones realizadas tuvieron un impacto positivo más amplio en la versión Móvil, logrando una reducción considerable en los tiempos necesarios para completarlas.
- **Magnitud de las mejoras**:
  - **Desktop**: La Tarea 5 mostró la mayor reducción promedio de tiempo ($\bar{d} = 3.11\,\text{s}$), seguida por la Tarea 4 ($\bar{d} = 2.29\,\text{s}$). Esto sugiere que las tareas con mayores requerimientos de interacción y navegación se beneficiaron particularmente de las optimizaciones realizadas.
  - **Móvil**: La Tarea 4 tuvo la mayor mejora ($\bar{d} = 2.90\,\text{s}$), seguida por la Tarea 2 ($\bar{d} = 2.53\,\text{s}$). Esto podría estar relacionado con mejoras en la disposición de elementos o accesibilidad en dispositivos móviles.

  Estas reducciones reflejan una mayor eficiencia en el flujo de trabajo, especialmente en tareas complejas que previamente demandaban más tiempo de los usuarios.
- **Comparación entre plataformas**: Aunque ambas plataformas mostraron mejoras, las reducciones promedio fueron más amplias y significativas en la versión Móvil, especialmente en tareas como la Tarea 4 y Tarea 2, donde se registraron valores t más altos y diferencias promedio más amplias. Esto podría indicar que las optimizaciones enfocadas en móviles tuvieron un impacto más sustancial, posiblemente debido a las limitaciones iniciales de esta plataforma.

Conclusión: Los resultados de esta prueba t para muestras pareadas sugieren que las optimizaciones realizadas contribuyeron significativamente a mejorar el rendimiento en varias de las tareas evaluadas. Estas mejoras se traducen en una experiencia de usuario más fluida y eficiente tanto en *desktop* como en móvil. En particular, las tareas más complejas mostraron las mayores reducciones en tiempo, lo cual refuerza que las optimizaciones se enfocaron eficazmente en los puntos críticos de interacción. Este análisis cuantitativo, junto con los comentarios cualitativos de los usuarios, respalda que las modificaciones en la interfaz lograron reducir los tiempos de navegación de forma consistente y significativa.

9.8.5. Impacto de la interacción visual (mapas de calor)

El análisis de los mapas de calor permite identificar las áreas de mayor interés visual de los usuarios, destacando los elementos de la interfaz que reciben mayor atención. A continuación, se presentan los hallazgos obtenidos en las secciones clave de la página web para evaluar la efectividad en la distribución de contenido y la navegación en dispositivos móviles y de escritorio.

Sección de perfil

En la pantalla de edición de perfil (ver Figura 8.86 para escritorio y Figura 8.90 para móvil), se observa una alta concentración de atención en los campos *Username*, *Password* y *Update*, lo cual indica que estos elementos son de interés inmediato para el usuario. Sin embargo, el botón de *Delete Account* también muestra actividad considerable, lo que podría sugerir curiosidad o una intención de explorar la funcionalidad. Esta interacción sugiere que los usuarios valoran la facilidad para actualizar su perfil, aunque también se debe considerar que el botón de eliminación de cuenta podría necesitar una posición menos prominente para evitar activaciones accidentales.

Sección de noticias

En la sección de noticias (ver Figura 8.89 para escritorio y Figura 8.91 para móvil), las áreas con mayor interés se encuentran en los encabezados de las noticias y las imágenes destacadas. Esto sugiere que los usuarios se sienten atraídos principalmente por elementos visuales y títulos llamativos. Esta observación resalta la importancia de utilizar imágenes relevantes y titulares claros para captar la atención y guiar al usuario en esta sección. Adicionalmente, la estructura de información debe ser clara y accesible para mejorar la experiencia de lectura.

Dashboard de estadísticas

En el dashboard de estadísticas (ver Figura 8.85 para escritorio y Figura 8.93 para móvil), los gráficos que muestran datos de juego, como el promedio de saltos y duración de juego total, presentan una concentración de atención significativa. Esta interacción indica que los usuarios están interesados en visualizar métricas de rendimiento, lo cual valida la inclusión de estos elementos en el *dashboard*. No obstante, se sugiere destacar visualmente los títulos de cada métrica para facilitar la comprensión inmediata del contenido del gráfico y su propósito.

Sección *About Us*

La sección *About Us* (ver Figura 8.87 para escritorio y Figura 8.92 para móvil) muestra un enfoque importante en el logotipo de la empresa y el nombre de la misma. Esto indica que los usuarios buscan identificar la identidad y el propósito del sitio rápidamente. Para mejorar esta experiencia, se recomienda incluir una breve introducción o resumen visible junto al logotipo, lo cual permitiría a los usuarios obtener una visión clara de la misión del proyecto sin tener que desplazarse hacia abajo en la página.

Página de inicio

En la página de inicio (ver Figura 8.88), el logotipo de Platyfa y el menú de navegación superior atraen la mayor parte de la atención, lo que sugiere que los usuarios se apoyan en estos elementos para

orientarse en el sitio. Esta observación refuerza la importancia de una navegación clara y accesible. Sin embargo, también es recomendable que otros elementos de la página de inicio, como botones de llamada a la acción o enlaces a secciones destacadas, estén ubicados estratégicamente para guiar mejor la interacción del usuario desde el primer acceso.

Conclusión del análisis de mapas de calor

Se concluye por ende que los mapas de calor nos demuestra que los usuarios solamente se están enfocando en la información relevante, tanto como en la sección de noticias que se enfocan en la informaición, cómo también, en la parte del home la cual muestra una mayor concentración en el logo del juego, y con ello, se cumple con el enfoque de tener un diseño que sea sencillo y efecto.

9.9. Análisis comparativo de *performance* en la API

Se realizaron dos pruebas de *performance* a la API para evaluar su capacidad de respuesta y estabilidad bajo carga, antes y después de implementar optimizaciones en los *endpoints* críticos. En esta sección se presenta una comparación entre los resultados de ambas pruebas, enfocándose en las métricas de tiempos de respuesta, tasas de transferencia y eficiencia en el manejo de solicitudes concurrentes.

9.9.1. Comparación de tiempos de respuesta

En la Tabla 9.4 se comparan los tiempos de respuesta entre la prueba inicial y la prueba final. Como se observa, los tiempos promedio, mínimo y máximo de duración de las solicitudes (`http_req_duration`) y de espera (`http_req_waiting`) disminuyeron tras las optimizaciones, lo cual indica una mejora en la eficiencia de la API.

Tabla 9.4: Comparación de tiempos de respuesta entre pruebas de *performance*

Métrica	**Prueba inicial (ms)**	**Prueba final (ms)**	**Reducción (%)**
`http_req_duration` (Duración de la solicitud)	242.78	235.21	3.1 %
`http_req_waiting` (Tiempo de espera)	223.08	217.98	2.3 %
`http_req_receiving` (Tiempo de recepción)	19.62	17.15	12.6 %
`http_req_sending` (Tiempo de envío)	76.48	69.25	9.5 %

Las pruebas de rendimiento realizadas después de las optimizaciones en la API muestran una reducción en los tiempos de espera y una mayor consistencia en el rendimiento, con una disminución promedio del 3.1 % en la duración de las solicitudes y del 9.5 % en el tiempo de envío de datos. Esto asegura que la API puede manejar el tráfico concurrente de manera confiable, proporcionando a los jugadores una experiencia sin interrupciones al consultar estadísticas de sus partidas.

9.9.2. Tasas de transferencia de datos

La Tabla 9.5 resume las tasas de transferencia de datos y las iteraciones promedio por segundo. Aunque los tiempos de respuesta mejoraron, la tasa de datos transferidos disminuyó en la prueba final. Esto se debe a que las optimizaciones se centraron en reducir la carga de datos para mejorar la eficiencia general de la API.

Tabla 9.5: Comparación de tasas de transferencia de datos y eficiencia

Métrica	**Prueba inicial**	**Prueba final**	**Cambio (%)**
Datos recibidos	2.1 MB (26 kB/s)	1.4 MB (18 kB/s)	-33.3 %
Datos enviados	1.4 MB (18 kB/s)	847 kB (10 kB/s)	-39.5 %
Tasa de iteración	17.45 iter/s	17.68 iter/s	+1.3 %

La reducción en las tasas de transferencia de datos refleja una mejora en la eficiencia de la API, al requerir menos datos para responder a las solicitudes. Esta optimización también ayudó a incrementar ligeramente la tasa de iteración, pasando de 17.45 a 17.68 iteraciones por segundo, lo cual es beneficioso en contextos de alta concurrencia.

9.9.3. Análisis de latencias de conexión

Finalmente, la latencia de conexión promedio también mostró una mejora notable, con un promedio de 464.9 µs en la prueba inicial y una latencia de 932.71 µs en la prueba final, debido a las configuraciones ajustadas para reducir la carga en los *endpoints* de alto uso.

9.9.4. Resumen de la comparativa

Los resultados comparativos confirman que las optimizaciones implementadas en la API lograron reducir los tiempos de respuesta y aumentar la eficiencia en el manejo de datos, lo cual cumple con el objetivo de brindarle al usuario un *endpoint* de estadísticas mucho más optimizado. Estas mejoras permiten que la API mantenga una alta tasa de éxito en las solicitudes y una estabilidad sólida incluso durante picos de concurrencia, lo cual es crucial para garantizar la escalabilidad y rendimiento en el uso de la plataforma.

CAPÍTULO 10

Conclusiones

El proyecto *Platyfa: Caso de Estudio del Desarrollo de un Videojuego en un Entorno Universitario Colaborativo* alcanzó exitosamente el objetivo general de diseñar, desarrollar e implementar un videojuego colaborativo que integrara programación avanzada, modelado y animación 3D, diseño UI/UX/GX, inteligencia artificial para enemigos y desarrollo web, ofreciendo una experiencia de juego inmersiva y divertida. Este logro se obtuvo gracias al trabajo interdisciplinario que fortaleció habilidades técnicas y creativas, así como la capacidad de trabajo en equipo de los participantes. Los aportes de cada módulo permitieron una integración coherente y de alta calidad: el diseño UI/UX/GX garantizó una experiencia intuitiva y atractiva, el modelado y la animación 3D enriquecieron la narrativa y la estética visual, la inteligencia artificial de los enemigos brindó desafíos dinámicos, y el módulo web complementó el juego con estadísticas en tiempo real y herramientas de análisis. A través de sesiones de *playtesting* y retroalimentación, se realizaron ajustes que mejoraron la experiencia del jugador, cumpliendo con los estándares planteados. En conjunto, *Platyfa* no solo se consolidó como un videojuego innovador y técnicamente robusto, sino también como una experiencia educativa transformadora que contribuye al fortalecimiento de la industria de videojuegos en Guatemala y la experiencia de trabajo en equipo.

El proyecto de Platyfa logró una integración exitosa entre los elementos y el diseño de la UI, UX y GX, proporcionando una experiencia inmersiva y entretenida para el jugador. Los elementos del diseño de la interfaz, como personajes y la narrativa trabajaron en conjunto para introducir al jugador en un mundo coherente y atractivo. Las gráficas de retroalimentación sugieren que los jugadores percibieron al juego como uno divertido, atractivo y visualmente atractivo que les permitió experimentar el inicio de la historia de nuestro protagonista, Gaus.

Los resultados de los cuestionarios, tanto antes como después de la prueba del juego, permitieron concluir que el modelado 3D de \textit{Platyfa} cumplió con los objetivos de la calidad visual desarrollada y el aumento del nivel de experiencia en el juego. Los encuestados ya han experimentado la visualización moderna a través de juegos en 3D, por lo que establece un estándar alto en lo que se debe esperar con respecto al diseño tridimensional. Es desde esta perspectiva que se prueba la hipótesis: un buen modelado 3D es primordial para captar la atención de un jugador al que se le debe ofrecer una experiencia rica y atractiva.

El desarrollo de las animaciones y sistemas de rigging para el proyecto "Gaus y la Revolución Ornitorense"ha sido exitoso, cumpliendo con los objetivos generales y específicos planteados al inicio del proyecto. Los resultados obtenidos en las pruebas de *GameTesting* y el análisis de cada animación validan que los esfuerzos en rigging y diseño de animación no solo alcanzaron los estándares de calidad

deseados, sino que también mejoraron la inmersión y la narrativa del juego, elementos fundamentales para una experiencia interactiva efectiva.

El análisis de los resultados obtenidos en las fases de testing la plataforma web confirma el cumplimiento de los objetivos específicos planteados para la plataforma web de Platyfa. La optimización de la usabilidad, guiada por principios como la Ley de Hick y Jakob Nielsen, permitió una navegación eficiente, con una reducción significativa en los tiempos de tarea tanto en desktop como en móvil. Asimismo, la elección de tecnologías modernas como PostgreSQL aseguró una plataforma escalable y eficiente para la promoción y gestión del juego. Además, mediante un diseño iterativo centrado en el usuario y la recopilación constante de datos sobre las acciones de los jugadores, se logró implementar un dashboard interactivo que permite analizar métricas clave del rendimiento y actividad de los jugadores, brindando herramientas estratégicas para la toma de decisiones.

El análisis del módulo de desarrollo de enemigos en el videojuego ha permitido identificar aspectos clave sobre la percepción de los jugadores y la eficacia de los algoritmos utilizados. En general, el módulo cumplió con su objetivo principal de proporcionar un desafío equilibrado y atractivo. No obstante, se identificaron áreas de mejora, como la detección y resolución de bugs, la implementación de tácticas adaptativas y la inclusión de configuraciones de dificultad ajustables. Abordar estos aspectos permitirá optimizar tanto la inteligencia artificial como la jugabilidad, sentando una base sólida para futuras iteraciones y mejoras del proyecto.

CAPÍTULO 11

Recomendaciones

El megaproyecto *Platyfa: Caso de estudio del desarrollo de un Videojuego en un Entorno Universitario Colaborativo* ha permitido la integración de diversos módulos especializados, generando aprendizajes significativos en cada área. A continuación, se presentan las recomendaciones generales derivadas del trabajo en los diferentes módulos, destacando oportunidades de mejora y sugerencias para futuros proyectos interdisciplinarios:

1. **Fortalecimiento de la comunicación interdisciplinaria:** Es fundamental establecer canales de comunicación claros y fomentar su uso activo entre los equipos de los diferentes módulos. Además de crear los canales, es crucial desarrollar la costumbre de utilizarlos regularmente y con propósito, asegurando que los objetivos del proyecto se mantengan alineados. También se recomienda realizar reuniones de seguimiento periódicas con agendas claras para resolver problemas y monitorear avances. La implementación de incentivos por cumplimiento de metas y penalizaciones por falta de compromiso podría ayudar a mantener la disciplina en estas dinámicas.

2. **Iteración basada en retroalimentación de usuarios:** Todos los módulos se beneficiarían de ciclos de retroalimentación más frecuentes, obtenidos tanto de jugadores como del equipo interno. Estas iteraciones permitirían ajustar aspectos del diseño, programación y animación antes de la finalización del proyecto, asegurando que el producto cumpla con las expectativas de los usuarios. Además, incluir pruebas de accesibilidad y ajustes específicos para mejorar la experiencia de usuario (UX) y la experiencia de juego (GX) es clave.

3. **Optimización del producto para publicación:** Considerando el objetivo de publicar *Platyfa* en plataformas como Steam, se recomienda desarrollar un plan de marketing detallado que incluya análisis de mercado, definición de público objetivo y estrategias promocionales. Además, sería beneficioso investigar los requisitos técnicos y legales para la publicación en plataformas digitales, asegurando que el proyecto cumpla con todos los estándares necesarios.

4. **Análisis y diseño UI/UX/GX:** Para optimizar la experiencia de usuario, se recomienda ajustar la interfaz de usuario (UI) mediante pruebas de accesibilidad adicionales y una mejor organización de los elementos en pantallas clave como configuración y controles. Además, fomentar la colaboración activa entre equipos mediante canales de comunicación efectivos y reuniones regulares con agendas claras ayudará a monitorear avances y resolver problemas oportunamente. Es importante revisar el balance de dificultad del juego para garantizar una

curva de aprendizaje adecuada que mantenga el interés de jugadores de distintos niveles. Finalmente, se debe fortalecer el plan de marketing y considerar los requisitos de publicación en plataformas como Steam, realizando estudios de mercado que alineen el producto con las expectativas del público objetivo.

5. **Modelado 3D:** Optimizar los materiales y texturas en el motor de juego puede mejorar significativamente el rendimiento y la calidad visual. Se recomienda explorar *shaders* personalizados en *Blender* para efectos visuales avanzados y fomentar la retroalimentación de los usuarios para alinear el estilo artístico con las expectativas de los jugadores. Promover el uso de *Blender* en la comunidad mediante talleres y capacitaciones fortalecerá el ecosistema local de desarrollo. Además, aprovechar herramientas de IA para texturizado y modelado reducirá tiempos de desarrollo sin sacrificar calidad, lo que resulta esencial para proyectos futuros.

6. **Animación:** Capacitar al equipo en herramientas como *Blender* y Mixamo optimizará los flujos de trabajo y mejorará la calidad de las animaciones. Se recomienda documentar exhaustivamente cada etapa, desde el rigging hasta la integración en Unity, y establecer ciclos de retroalimentación frecuentes para ajustar detalles antes de la fase final del proyecto. La expansión del alcance del juego, como la creación de nuevos niveles y personajes, permitirá enriquecer la narrativa y la jugabilidad. Explorar técnicas como captura de movimiento o animación 2D puede añadir dinamismo y profundidad visual al juego. Además, recopilar feedback post-lanzamiento asegurará actualizaciones que mantengan el interés de los jugadores a largo plazo.

7. **Enemigos:** Para mejorar la inteligencia artificial de los enemigos, se sugiere implementar un sistema de dificultad adaptable que ajuste los desafíos según el desempeño del jugador. Diversificar tácticas y comportamientos de los enemigos incrementará la inmersión y reducirá la previsibilidad. Un protocolo robusto de pruebas continuas permitirá detectar y corregir errores en los algoritmos, mientras que la retroalimentación de los jugadores contribuirá a ajustar las interacciones y patrones de ataque. Finalmente, ofrecer configuraciones personalizables de dificultad mejorará la accesibilidad y satisfacción del usuario.

8. **Escalabilidad y mantenimiento del sistema web:** Para garantizar un sistema web robusto y preparado para el crecimiento, es importante planificar la escalabilidad del backend. Esto incluye la implementación de microservicios, balanceadores de carga y sistemas de caché avanzados. Continuar con pruebas iterativas garantizará que la plataforma web se adapte a las necesidades cambiantes de los usuarios. Integrar elementos de gamificación, como logros e insignias, aumentará el compromiso y retención. Además, la integración de funcionalidades sociales, como compartir logros en redes sociales o interactuar con otros jugadores, puede fortalecer la comunidad del juego y aumentar el interés en la plataforma.

9. **Promoción del uso de herramientas *Open Source*:** Dado el éxito obtenido con herramientas como *Blender*, se recomienda continuar fomentando su uso dentro de la comunidad de desarrollo local. Talleres, charlas y capacitaciones podrían ser una excelente manera de introducir a nuevos desarrolladores a estas herramientas, impulsando el crecimiento de la industria de videojuegos en Guatemala.

Estas recomendaciones buscan no solo mejorar el resultado de futuros proyectos, sino también fortalecer las habilidades y capacidades de los equipos multidisciplinarios involucrados. Al adoptar estas sugerencias, se espera que el desarrollo de videojuegos en entornos universitarios y colaborativos continúe creciendo en calidad y relevancia.

CAPÍTULO 12

Bibliografía

Bibliografía

[1] *Bases de Datos Relacional ¿Qué Es Y Sus Características?* https://ayudaleyprotecciondatos.es/bases-de-datos/relacional/, Recuperado de: https://ayudaleyprotecciondatos.es/bases-de-datos/relacional/. 62, 63, 64

[2] *Express - Node.js web application framework.* https://expressjs.com/, Recuperado de: https://expressjs.com/. 60, 62

[3] *Fastify: Fast and low overhead web framework, for Node.js.* https://www.fastify.io/, Recuperado de: https://www.fastify.io/. 60

[4] *NestJS: A Progressive Node.js Framework.* https://docs.nestjs.com/, Recuperado de: https://docs.nestjs.com/. 60

[5] *Empresas Que Usan Vue.js (33,254) | TheirStack.com*, 2014. https://theirstack.com/es/technology/vue-js. 57

[6] *GitHub Documentation: GitHub Issues*, 2022. https://docs.github.com/en/issues. 66, 67

[7] *Opportunities for Game Developers in Guatemala*, 2022. https://devguatemala.com/opportunities-2022. 68

[8] *Qué Es Tailwind CSS Y Por Qué Deberías Usarlo*, January 2022. https://openwebinars.net/blog/que-es-tailwind-css-y-por-que-deberias-usarlo/, Recuperado de: https://openwebinars.net/blog/que-es-tailwind-css-y-por-que-deberias-usarlo/. 58, 59

[9] *The Rise of Indie Game Development in Guatemala*, 2022. https://www.gamasutra.com/view/guatemala-indie-game-development.php. 67

[10] *What is Trello?*, 2022. https://trello.com. 66

[11] *¿Cómo Usar Tailwind CSS Para Desarrollar Rápidamente Sitios Web Elegantes?*, March 2022. https://kinsta.com/es/blog/tailwind-css/, Recuperado de: https://kinsta.com/es/blog/tailwind-css/. 58, 59

[12] *Character Design 101: Bringing Characters to Life*, 2023. https://www.bolton.ac.uk/blogs/character-design-101-bringing-characters-to-life. 42

[13] *Guatemala Game Devs Meetup: A Community on the Rise*, 2023. https://www.gameguatemala.com/dev-meetup-2023. 68

[14] *State of JavaScript 2023*, 2023. https://2023.stateofjs.com/en-US. 57, 61

[15] *Empresas Que Usan Bootstrap (46,641) | TheirStack.com*, 2024. https://theirstack.com/es/technology/bootstrap. 59

[16] *State of CSS*, 2024. https://stateofcss.com/en-US. 59

[17] 360, Innovación Digital: *Deep learning: qué es, cómo funciona y cuáles son los casos de aplicación*, August 2023. https://www.innovaciondigital360.com/i-a/deep-learning-que-es-el-aprendizaje-profundo-como-funciona-y-cuales-son-los-casos-de-aplica 14

[18] AAcademica: *Análisis de la influencia narrativa del videojuego en el cine*, February 2017. https://www.aacademica.org/000-020/32.pdf. 13

[19] Adams, Ernest y Andrew Rollings: *On Game Design*. New Riders, New Jersey, 2003. 7, 52

[20] Adobe: *What is 3D modeling used for?* Adobe, s.f. https://www.adobe.com/products/substance3d/discover/what-is-3d-modeling.html, Consultado en septiembre de 2024. 23, 24

[21] Aguayo: *Teoría Nielsen Norman Pruebas de Usabilidad*, 2023. https://aguayo.co/es/blog-aguayo-experiencia-usuario/teoria-nielsen-norman-pruebas-usabilidad-5-usuarios/. 165

[22] Alaiza, A.: *10 conceptos imprescindibles – Esenciales CGI II*. espai, 2020. https://www.espai.es/blog/2020/11/10-conceptos-imprescindibles-esenciales-cgi-ii/, Consultado en septiembre de 2024. 32, 33

[23] Aminian, P.: *Stylized Visionaries: artistic interpretations of games*. Pixune, August 6 2024. https://pixune.com/blog/stylized-art-style/. 49

[24] Anderson, John: *The Complexity of Enemy Design in Elden Ring: A Comprehensive Review*. Game Design Journal, 2022. 12

[25] Andreu, A. P.: *Motion Capture en la industria de la Animación*. Blog CEDIM, December 7 2022. https://blog.cedim.edu.mx/animacion/motion-capture-en-la-industria-de-la-animacion/, Accessed: 2022-12-07. 36

[26] ArcheryTag: *Historia de la ballesta: origen y curiosidades*, s.f. https://www.archerytagbarcelona.com/blog/historia-ballesta/. 96

[27] Arias, A.: *Introducción al modelado poligonal*. Tokio School, 2024. https://www.tokioschool.com/noticias/modelado-poligonal/, Consultado en septiembre de 2024. 28

[28] Artadmin: *Video Game Art Styles: A Glimpse into Gaming Aesthetics*. Gamepack Studio, November 15 2023. https://gamepackstudio.com/video-game-art-styles-a-glimpse-into-gaming-aesthetics/. 49

[29] Autodesk: *Retopology: Unlocking new horizons in 3D artistry*. Autodesk, s.f. https://www.autodesk.com/solutions/retopology, Consultado en septiembre de 2024. 31

[30] Azsan, F.: *Exploring CGI and 3D Animation*. Polydin, August 26 2024. https://polydin.com/cgi-and-3d-animation/, Accessed: 2024-08-26. 33

[31] Azure, Microsoft: *What is Deep Learning?*, January 2024. https://azure.microsoft.com/es-es/resources/cloud-computing-dictionary/what-is-deep-learning. 14

[32] Bates, Bob: *Game Design: The Art and Business of Creating Games*. Premier Press, 2004. 8

[33] Bisht, V.: *Animation in video games: creating immersive experiences.* aajjo.com, May 16 2024. https://blog.aajjo.com/post/animation-in-video-games-creating-immersive-experiences. 44, 45, 46, 47, 48, 49

[34] Bottomley, P.: *The Level Design Document.* https://petebottomley.wordpress.com/the-level-design-document/. 22

[35] Bueno, Julián: *¿Cuántos usuarios son necesarios para un test de UX y usabilidad?*, 2023. https://www.julianbueno.com/cuantos-usuarios-son-necesarios-para-un-test-de-ux-y-usabilidad/. 165

[36] Calderon, C. y K. Mahadevan: *Accessibility in design: Making the web accessible*, 2018. https://www.interaction-design.org/literature/article/accessibility-in-design-making-the-web-accessible, Interaction Design Foundation. 21

[37] Campus, C.: *¿Qué es el rigging en animación?* Universidad Europea Creative Campus, May 22 2024. https://creativecampus.universidadeuropea.com/blog/que-es-rigging/. 53

[38] Carlos, Juan: *Juan Carlos Yovera | Senior UI/UX Front End*, 2024. https://jconsoftwares.com/blog/article/11. 56, 57

[39] Centropixels: *Kitbashing: historia, técnicas y aplicaciones en el arte digital.* Centropixels, 2024. https://centropixels.com/kitbashing/, Consultado en septiembre de 2024. 30

[40] CESUMA: *Avances y aplicaciones del aprendizaje profundo*, October 2023. https://www.cesuma.mx/blog/avances-y-aplicaciones-del-aprendizaje-profundo.html. 14

[41] Chacon, Scott y Ben Straub: *ProGit.* Apress, 2014. 66, 67

[42] Chapman, C.: *The psychology of color in marketing and branding*, 2010. https://www.smashingmagazine.com/2010/01/the-psychology-of-color-in-marketing-and-branding/, Smashing Magazine. 21

[43] Chisaba, C. a. O.: *Captura de movimiento: La tecnología usada en Avatar.* Niixer. Portal De Noticias De Tecnología, Realidad Virtual, Aumentada Y Mixta, Videojuegos, February 22 2021. https://niixer.com/index.php/2021/02/22/captura-de-movimiento-tecnologia/, Accessed: 2021-02-22. 37

[44] Ciencias Informáticas (UCI), Universidad de las: *Redes Neuronales aplicadas a videojuegos*, October 2008. https://repositorio.uci.cu/bitstream/ident/TD_1609_08/1/TD_1609_08.pdf. 16

[45] Clancy, Marc: *How to make a moodboard in 10 easy steps*, 2024. https://milanote.com/guide/create-better-moodboards. 11

[46] Cooper, J.: *Making of God of War: Cinematics & Rigging.* Game Anim, October 3 2019. https://www.gameanim.com/2018/06/20/making-of-god-of-war-cinematics-rigging/. 45

[47] Cortés, J. y J. Cortés: *¿Qué es la Animación 3D? Tipos, Técnicas y Programas.* Notodoanimacion.es | Noticias, Recursos, Tutoriales y Empleo para Artistas Digitales, August 17 2024. https://www.notodoanimacion.es/que-es-la-animacion-3d-tipos-y-tecnicas/, Accessed: 2024-08-17. 33, 34

[48] Cris Bank, Jerry Cao: *Web UI design best practices.* https://github.com/hckhanh/ui-ux/blob/master. 17

[49] Cristancho, Felipe: *¿Qué Es Angular Y Para Qué Sirve? - Talently*, July 2022. https://talently.tech/blog/que-es-angular/. 57

[50] De, Jesse Schell: *The Art of Game Design: A Book of Lenses*. CRC Press, Boca Raton, FL, 2009. 80

[51] DeGuzman, K.: *What is Mocap — The Science and Art Behind Motion Capture*. StudioBinder, September 7 2023. https://www.studiobinder.com/blog/what-is-mocap-definition/, Accessed: 2023-09-07. 36

[52] Dialnet: *Estudio sobre el impacto de los videojuegos en la sociedad*, November 2021. https://dialnet.unirioja.es/descarga/articulo/7894709.pdf. 14

[53] Digital, Salud: *Aprendizaje profundo y sus aplicaciones en la atención médica*, September 2023. https://saluddigital.com/es/noticias/aprendizaje-profundo-y-sus-aplicaciones-en-la-atencion-medica/. 14, 15, 16

[54] Donovan, Tristan: *Replay: The History of Video Games*. Yellow Ant, 2010. 8

[55] Dumitru, C. Z.: *Where character animation meets human behavior sciences*. Cristina Teaching Art, June 26 2021. https://www.cristinateachingart.com/where-character-animation-meets-human-behavior-sciences/. 43

[56] Editores: *Figma*. https://es.wikipedia.org/wiki/Figma. 55

[57] Education, Euroinnova International Online: *Conoce qué es una diseñadora gráfica*. September 17 2024. https://www.euroinnova.com/blog/latam/caracterizacion-de-personajes. 42, 43

[58] ESA: *Essential facts: About the U.S Video Game Industry*. theesa.com, 2024. https://www.theesa.com/resources/essential-facts-about-the-us-video-game-industry/2024-data. 4

[59] Farhatullah, Rai: *Storiboarding*. Film and Television Production Process, 2017. 23

[60] Fernando y Fernando: *Auto-Rig Pro Descargar addon GRATIS*. Sketchitos.com, September 15 2024. https://sketchitos.com/auto-rig-pro-descargar-addon-gratis/. 54

[61] Fernández, E.: *Qué es el modelado 3D: tipos, herramientas y cómo dar vida a tus creaciones digitales*. Tokio School, 2022. https://www.tokioschool.com/noticias/tipos-modelado-3d/, Consultado en septiembre de 2024. 29

[62] Floriano, J. F. J. y J. F. J. Floriano: *Técnicas de animación 3D*. Blog de DSIGNO, December 19 2022. https://www.dsigno.es/blog/diseno-grafico/tecnicas-de-animacion-3d, Accessed: 2022-12-19. 34, 35

[63] Formación, Deusto: *Tipos de tecnologías para el desarrollo de videojuegos*, January 2024. https://www.deustoformacion.com/blog/diseno-produccion-audiovisual/tipos-tecnologias-desarrollo-videojuegos. 51

[64] Frey, Edge y: *Los mejores programas de modelado 3D gratis de 2024*. All3dp, 2024. https://all3dp.com/es/1/modelado-3d-programa-diseno-3d-principiantes/, Consultado en septiembre de 2024. 53

[65] Furneri, F.: *In this first part, learn how to create clean 3D models by following these simple tips and practical examples.* shutterstock, 2022. https://www.shutterstock.com/blog/building-clean-3d-models, Consultado en septiembre de 2024. 32

[66] GameDevGT: *Lista de videojuegos guatemaltecos*, 2024. https://docs.google.com/spreadsheets/d/1zloW0iGvD2NOWiqKADNfy8AwBoUJvJQSQxLXcPC5Gno/edit?gid=1938942876#gid=1938942876. 4

[67] Gao Nap, J.: *Memoria de trabajo final de máster: Desarrollo de videojuegos*, January 2024. https://openaccess.uoc.edu/bitstream/10609/149416/1/jgaonapTFM0124memoria.pdf. 14

[68] Genbeta: *10 herramientas de creación de videojuegos para programadores principiantes*, May 2023. https://www.genbeta.com/desarrollo/10-herramientas-creacion-videojuegos-para-programadores-principiantes. 51

[69] Giménez, R.: *Keyframe Animation: Aplicación en Producción 3D - Animum 3D.* Animum 3D, January 12 2024. https://www.animum3d.com/blog/keyframe-animation/, Accessed: 2024-01-12. 35, 36

[70] Herranz, S.: *Así jugábamos: 1996-1997. Llega la revolución 3D.* Hobbyconsolas, 2021. https://www.hobbyconsolas.com/reportajes/jugabamos-1996-1997-llega-revolucion-3d-818545, Consultado en septiembre de 2024. 27, 28

[71] Hodent, C.: *The Gamer's Brain: How Neuroscience and UX can Impact Video Game Design. CRC Press*, 2018. https://books.google.com.gt/books?hl=en&lr=&id=JzyhDwAAQBAJ&oi=fnd&pg=PP1&dq=video+games+ui+ux&ots. 22

[72] Horizonte: *Ventajas Y Desventajas de Usar Bootstrap*, August 2020. https://soyhorizonte.com/blog/ventajas-y-desventajas-de-usar-bootstrap/, Recuperado de: https://soyhorizonte.com/blog/ventajas-y-desventajas-de-usar-bootstrap/. 58, 59

[73] Huhtamo, Erkki: *Slots of Fun, Slots of Trouble: An Archaeology of Arcade Gaming.* En Raessens, Joost y Jeffrey Goldstein (editores): *Handbook of Computer Game Studies*, páginas 3–22. MIT Press, Cambridge, MA, 2005. 6

[74] IAT: *INTELIGENCIA ARTIFICIAL EN VIDEOJUEGOS: UNA MIRADA AL PASADO Y FUTURO DE LA INDUSTRIA*, March 19 2020. https://iat.es/tecnologias/inteligencia-artificial/videojuegos/. 51, 52, 53, 83

[75] IGN: *God of War Review.* IGN, April 13 2018. https://www.ign.com/articles/2018/04/12/god-of-war-review. 44, 45

[76] Institucional, UAM Repositorio: *Análisis de la influencia narrativa del videojuego en el cine*, March 2023. https://repositorio.uam.es/bitstream/handle/10486/689721/AN_3_23.pdf?isAllowed=y&sequence=1. 13

[77] Institucional, ULPGC Repositorio: *Estudio de videojuegos y comportamiento en el ámbito educativo*, March 2016. https://accedacris.ulpgc.es/bitstream/10553/25514/9/0742418_00000_0000.pdf. 14

[78] Interaction Design Foundation: *Hick's Law*, n.d. https://www.interaction-design.org/literature/topics/hick-s-law, Disponible en: https://www.interaction-design.org/literature/topics/hick-s-law. 19

[79] ITSQMETt: *Animación Digital Está Revolucionando el Diseño.* 2024. https://itsqmet.edu.ec/animacion-digital/. 4

[80] Jesús: *Ventajas Y Desventajas de CSS3: Una Mirada Crítica a CSS3*, December 2023. https://www.dongee.com/tutoriales/ventajas-y-desventajas-de-css3/. 58, 59

[81] Johnson, A.: *Data-Driven Game Design: Improving Player Retention.* En *Game Development Conference Proceedings*, 2018. 5

[82] Juul, Jesper: *A Casual Revolution: Reinventing Video Games and Their Players.* MIT Press, 2010. 9

[83] Kajala, N.: *Designing a Combat System for 3D Video Game*, 2022. https://www.theseus.fi/bitstream/handle/10024/746542/Kajala_Niilo.pdf?sequence=2, Project: Ripple in Dimensions. 50

[84] Kent, Steven L.: *The Ultimate History of Video Games: From Pong to Pokémon and Beyond*. Three Rivers Press, 2001. 6, 7, 8

[85] Kyle, J.: *Spider-Man's fluid animations make him feel agile and responsive, embodying the essence of the character as a superhero who swings through the city with grace*. Animation and Game Design: The Case of Marvel's Spider-Man, 12:45–50, 2019. 44

[86] Lasseter, J.: *Principles of traditional animation applied to 3D computer animation*. ACM SIGGRAPH Computer Graphics, 21(4):35–44, 1987. 37

[87] Lautrec, Toulouse: *Herramientas para crear un videojuego*, June 2023. https://www.toulouselautrec.edu.pe/blogs/herramientas-crear-videojuego. 51

[88] Lidwell, William, Kritina Holden y Jill Butler: *Universal principles of design*. Rockport Publishers, 2010. 21

[89] Linderoth, Jonas y Ulrik Bennerstedt: *Game Studies: The Game, the Player, the World: Looking for a Heart of Gameness*. En *Proceedings of the Digital Games Research Association (DiGRA) 2007 Conference*, 2007. 12

[90] Londoño, Pablo: *Qué Es Bootstrap, Para Qué Sirve Y Cómo Funciona*, 2023. https://blog.hubspot.es/website/que-es-bootstrap, Recuperado de: https://blog.hubspot.es/website/que-es-bootstrap. 58

[91] Lowe, R. K. y W. Schnotz: *Animation principles in multimedia learning*. En *Cambridge University Press eBooks*, páginas 513–546. 2014. 41

[92] M., A. G.: *¿Cuál fue el primer videojuego? Esta es la historia de un pasatiempo universal*, January 10 2024. https://historia.nationalgeographic.com.es/a/cual-fue-primer-videojuego-esta-es-historia-pasatiempo-universal_20123. 6

[93] Manuel, C. M. J.: *La reinterpretación de los principios clásicos de animación en los medios digitales*. August 31 2015. https://eprints.ucm.es/32970/. 37

[94] Martin, Javier: *Fastify for High-Performance Applications*, 2023. https://fastify.com/blog/performance/, Recuperado de: https://fastify.com/blog/performance/. 61

[95] MasterD: *Qué es Unity y para qué sirve*. https://www.masterd.es/blog/que-es-unity-3d-tutorial, November 8 2019. Accessed: 2024-09-20. 112

[96] Maxon: *Maxon - Cinema 4D - Mixamo Control Rig*. https://www.maxon.net/es/cinema-4d/features/mixamo-control-rig, 2024. Accessed: 2024-09-20. 55

[97] Medina, Noe: *CSS de Ahora Y Del Futuro*, May 2022. https://www.paradigmadigital.com/dev/css-ahora-futuro/, Accessed 17 Aug. 2024. 59

[98] Meristation: *Análisis de la influencia narrativa del videojuego en el cine*, November 20 2020. https://as.com/meristation/2020/11/20/reportajes/1605863800_319793.html. 13

[99] Mintforpeople: *Tipos de Modelado 3D: descubre las diferentes técnicas*. Mintforpeople, 2023. https://mintforpeople.com/noticias/tipos-de-modelado-3d/, Consultado en septiembre de 2024. 30

[100] Monsuton: *¿Qué es el rigging en animación? Definición y Usos*, s.f. https://www.monsuton.com/rigging/. 11

[101] Moody, A.: *3D Pioneers unveil new company - Mixamo*. Animation World Network, 2009. https://www.awn.com/news/3d-pioneers-unveil-new-company-mixamo. 55

[102] Murphy, C.: *DESIGN BETTER GAMES! FLOW, MOTIVATION, & FUN*. 2013. http://www.goodgamesbydesign.com/Files/Chapter5_Flow_Motivation_Fun_Final_WebVersion.pdf. 50

[103] Naghdi, A.: *How does shape language impact a character design? (with illustrated examples)*, 2021. https://dreamfarmstudios.com/blog/shape-language-in-character-design/. 86

[104] Nicolas Marmaras, George Poulakakis, Vasilis Papakostopoulos: *Ergonomic design in ancient Greece*. ELSEVIER, 1997. https://www.academia.edu/34119158/Ergonomic_design_in_ancient_Greece. 16

[105] Noguera, J.: *Redes Neuronales aplicadas a un videojuego en el entorno de Unity*, December 2023. https://riunet.upv.es/entities/publication/d7657d40-c6f9-402d-9023-d3f440d557a1. 15

[106] Oliveira, J.: *¿Qué es un keyframe?, claves para editar tus mejores videos*. Crehana, March 29 2021. https://www.crehana.com/blog/estilo-vida/que-es-un-keyframe/, Accessed: 2021-03-29. 35

[107] Plarium, s.f. https://plarium.com/en/glossary/gameplay-meaning/. 9

[108] Polydin: *Mastering Game Development | Top 3D Modeling Techniques for Video Games*. Polydin, 2023. https://polydin.com/3d-modeling-techniques/, Consultado en septiembre de 2024. 29, 30

[109] Poole, Steven: *Trigger Happy: Videogames and the Entertainment Revolution*. Arcade Publishing, 2000. 9

[110] Porokh, A.: *3D MODELING GAME: WHAT YOU NEED TO KNOW*. KEVURU Games, 2023. https://kevurugames.com/blog/3d-modeling-for-video-games/, Consultado en septiembre de 2024. 23

[111] PulseCollege: *Character Design - The Secrets To Creating Memorable Characters*. https://www.pulsecollege.com/character-design-the-secrets-to-creating-memorable-characters/. 87

[112] Quintero Bernal, J.: *Generador de comportamientos de enemigos para videojuegos 2D*, July 2020. https://eprints.ucm.es/61704/1/Quintero_Bernal_Generador_de_comportamientos_de_enemigos_para_videojuegos_2D_4398577_1546886327.pdf. 13

[113] Rabin, Steve: *AI Game Programming Wisdom 4*. Charles River Media, 2010. 12

[114] Redalyc: *Análisis de la influencia narrativa del videojuego en el cine*, June 2023. https://www.redalyc.org/journal/6887/688775175001/html/. 13

[115] Repositorio, UPC: *Trabajo de investigación en videojuegos*, July 2023. https://upcommons.upc.edu/handle/2117/393301. 13

[116] Rodríguez, M.: *La Primera Era del 3D: La generación que lo cambió todo*. MeriStation|as, 2018. https://as.com/meristation/2018/10/19/reportajes/1539948499_015444.html, Consultado en septiembre de 2024. 26, 27, 28

[117] Rollings, Andrew y Ernest Adams: *Andrew Rollings and Ernest Adams on Game Design*. New Riders, Indianapolis, 2003. 5

[118] Rootstack: *VueJS: Ventajas Y Desventajas de Este Framework*, 2021. https://rootstack.com/es/blog/vuejs-ventajas-desventajas, Recuperado de: https://rootstack.com/es/blog/vuejs-ventajas-desventajas. 56, 57

[119] Rubin, Kenneth S: *Essential Scrum: A Practical Guide to the Most Popular Agile Process*. Addison-Wesley, Boston, 2012. 66

[120] Rururen, Jyrki: *THE GOOD, THE BAD AND THE UNPLEASANT – A STUDY OF GRAPHICAL USER INTERFACES IN VIDEO GAMES*. Informe técnico, Tampereen Teknillinen Yliopisto, Tampere University of Technology, 2017. 4

[121] Saavedra, Jose Angel: *Qué Es React Y Para Qué Sirve*, July 2023. `https://ebac.mx/blog/que-es-react`, Recuperado de: https://ebac.mx/blog/que-es-react. 56, 57

[122] Salen, Katie y Eric Zimmerman: *Rules of Play: Game Design Fundamentals*. MIT Press, Cambridge, MA, 2004. 12

[123] Sanmartín, J.: *Qué es la pose en T, para qué sirve y por qué todos los personajes de los videojuegos la hacen*. vida extra, 2023. `https://www.vidaextra.com/industria/que-t-pose-sirve-que-todos-personajes-videojuegos-hacen`, Consultado en septiembre de 2024. 32, 33

[124] Santos, Diego: *Introducción al CSS: Qué Es, Para Qué Sirve Y Otras 10 Preguntas Frecuentes*, 2023. `https://blog.hubspot.es/website/que-es-css`, Recuperado de: https://blog.hubspot.es/website/que-es-css. 58

[125] School, Tokio: *Herramientas para crear videojuegos*, February 2024. `https://www.tokioschool.com/noticias/herramientas-crear-videojuegos/`. 51

[126] Schwaber, Ken y Mike Beedle: *Scrum Development Process*. Springer, New York, 2002. 66

[127] Seemann, Glenn y David M. Bourg: *AI for Game Developers*. O'Reilly Media, 2004. 8

[128] SEMrush: *Usabilidad Web: 10 principios de usabilidad de Jakob Nielsen*, n.d. `https://es.semrush.com/blog/usabilidad-web-principios-jakob-nielsen/`, Disponible en: https://es.semrush.com/blog/usabilidad-web-principios-jakob-nielsen/. 19

[129] Services, Amazon Web: *What is Deep Learning?*, January 2024. `https://aws.amazon.com/es/what-is/deep-learning/`. 14

[130] Sheikh, M. y M. Sheikh: *Programa de Animación: 15 Herramientas Para Principiantes*. Visme Blog, September 8 2024. `https://visme.co/blog/es/programa-de-animacion/#blender`. 54

[131] Smith, John: *Advantages of Using NestJS*, 2023. `https://nestjs.com/blog/advantages/`, Recuperado de: https://nestjs.com/blog/advantages/. 61

[132] Source, Mordor Intelligence™ Industry Reports: *Industria del juego - Análisis de tamaño y participación - Tendencias y pronósticos de crecimiento (2024 - 2029) Source: https://www.mordorintelligence.com/es/industry-reports/global-gaming-market*. Mordor Intelligence™ Industry Reports Source, 2024. `https://www.mordorintelligence.com/es/industry-reports/global-gaming-market`, Consultado en septiembre de 2024. 4

[133] Statham, N.: *Characterization through character design in competitive multiplayer games*. Tesis de Doctorado, UPPSALA UNIVERSITET, 2023. `https://uu.diva-portal.org/smash/get/diva2:1767167/FULLTEXT01.pdf`. 43

[134] Tablado, Fernando: *Base de Datos No Relacional. ¿Qué Es? Características Y Ejemplos*, September 2020. `https://ayudaleyprotecciondatos.es/bases-de-datos/no-relacional/#google_vignette`. 63, 64

[135] Tai: *¿Qué es la animación 3D y para qué sirve?* TAI ARTS, April 21 2022. `https://taiarts.com/blog/que-es-animacion-3d-y-para-que-sirve/`. 43

[136] Tech, Atlantic: *Desarrollo de videojuegos: tecnologías que se utilizan y tendencias en 2022*, July 2022. https://atlantictech.io/desarrollo-de-videojuegos-tecnologias-se-utilizan-y-tendencias-en-2022/. 51

[137] Tiigimägi, S.: *¿Cómo modelar en 3D? [Guía completa].* 3dstudio, s.f. https://3dstudio.co/es/how-to-3d-model/, Consultado en septiembre de 2024. 29

[138] Torres, A.: *¿Qué es la animación 3D y qué tipos existen?* ESDESIGN, August 5 2024. https://www.esdesignbarcelona.com/actualidad/animacion/que-es-la-animacion-3d-y-que-tipos-existen, Accessed: 2024-08-05. 33

[139] U-tad: *¿Qué son y para qué sirven las UVs?* U-tad, 2024. https://u-tad.com/que-son-y-para-que-sirven-las-uvs/, Consultado en septiembre de 2024. 33

[140] UC-San-Diego: *Why User Experience (UX) Design Matters: The Importance of User Experience and Design Thinking for Products and Services*, s.f. https://extendedstudies.ucsd.edu/news-and-events/blog/why-is-ux-design-important. 4

[141] Uddin, Nasir: *A Brief History and Evolution of UI UX Design*, 2024. https://musemind.agency/blog/ui-ux-design-history. 16

[142] Unity: *Optimizando el Rendimiento Gráfico.* https://docs.unity3d.com/es/530/Manual/OptimizingGraphicsPerformance.html. 85

[143] Uno, Prompt: *Redes neuronales profundas en los juegos de video*, November 2023. https://prompt.uno/redes-neuronales-profundas/redes-neuronales-profundas-en-los-juegos-de-video/. 16

[144] Utterson, Andrew: *A Computer Animated Hand*, s.f. https://www.loc.gov/static/programs/national-film-preservation-board/documents/computer_hand2.pdf, Consultado en septiembre de 2024. 25

[145] Vergara, Sergio: *¿Por Qué Tailwind CSS Es Tan Popular?*, November 2023. https://www.itdo.com/blog/por-que-tailwind-css-es-tan-popular/. 59

[146] Vittar, J.: *Box Modeling en 3D: Una Guía Completa.* fandel3D, 2023. https://fanaticosdel3d.com/box-modeling/, Consultado en septiembre de 2024. 31

[147] Webber, Alex: *Express.js Pros and Cons*, 2023. https://medium.com/express-js-pros-cons/, Recuperado de: https://medium.com/express-js-pros-cons/. 61

[148] Wikipedia: *Jakob Nielsen (usability consultant)*, n.d. https://en.wikipedia.org/wiki/Jakob_Nielsen_(usability_consultant), Disponible en: https://en.wikipedia.org/wiki/Jakob$_N ielsen_(usability_c onsultant)$. 19

[149] Wikipedia: *Ley de Hick*, n.d. https://es.wikipedia.org/wiki/Ley_de_Hick, Disponible en: https://es.wikipedia.org/wiki/Ley$_d e_H ick$. 18

[150] Wilson, Debra: *Born to the Purple*, s.f. https://carnegiemnh.org/born-to-the-purple/. 92

[151] Wirtz, B.: *Game Design with Impact: The Psychology of Video Game Immersion (Bonus: List of the Most Immersive Games).* Video Game Design and Development, July 4 2023. https://www.gamedesigning.org/learn/game-immersion/. 46, 47, 48

[152] Wolf, Mark J. P.: *The Medium of the Video Game.* University of Texas Press, 2001. 7

[153] Álvarez, Carlos: *Industria de Videojuegos en Latinoamérica: Oportunidades y Retos.* Universidad de San Carlos, Guatemala City, 2019. 68

CAPÍTULO 13

Anexos

13.1. Proceso para la creación de los modelos

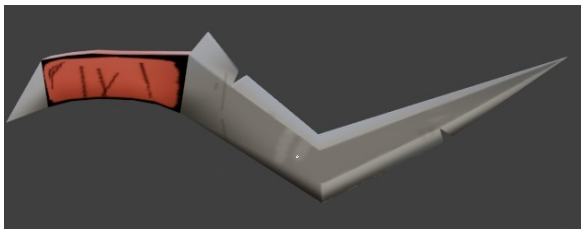

Figura 13.1: Modelo del arma del enemigo dingo.

Figura 13.2: Modelo del enemigo dingo.

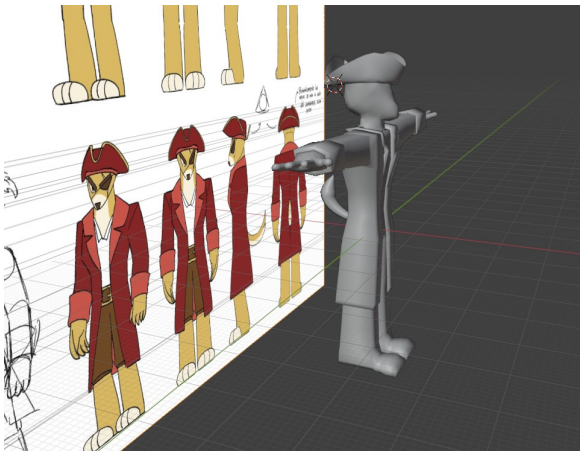

Figura 13.3: Boceto del modelo del *boss* dingo.

Figura 13.4: Modelo finalizado del *boss* dingo.

Figura 13.5: Modelo del enemigo búho.

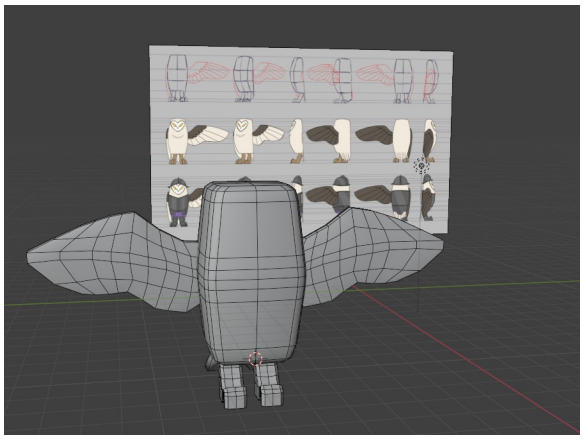

Figura 13.6: Boceto del modelo del enemigo búho.

Figura 13.7: Modelo del enemigo búho.

Figura 13.8: Modelo del *boss* búho.

13.2. Tablas de resultados

Categoría	Comentarios de los jugadores
Comportamiento adecuado	▪ Me parece el comportamiento adecuado. ▪ Los veo bien. ▪ Todo bien. ▪ Creo que tienen su estilo los enemigos. ▪ Me parecen bien.
Animación y naturalidad	▪ Que no estuvieran tan tiesos. ▪ Que sea un poco más animado. ▪ Animaciones más representativas. ▪ Más variedad de ataques o naturalidad de movimientos.
Mejora en la inteligencia artificial	▪ Su inteligencia ya que se quedaban quietos mucho tiempo. ▪ Que sí sigan al jugador. ▪ Me parece bien, mejoraría la inteligencia artificial de estos, pero creo que está lograda hasta cierto punto.
Respuesta ante cercanía del jugador	▪ Tal vez que si estás más cerca, te den golpes cuerpo a cuerpo. ▪ Tal vez podrían moverse. ▪ Que se muevan un poco más.
Reacción y detección	▪ Si hay algunos enemigos que tardan en detectarnos, lo que conlleva a que tarden en atacar. ▪ Tal vez que el ataque del boss se ajuste al movimiento del jugador.
Propuestas específicas para el Boss	▪ Al boss yo le agregaría un escudo para entender que debo usar primero el spaghetti y luego las balas. ▪ A menos que se quiera hacer más complejo el juego, creo que el comportamiento es bastante estándar.

Categoría	Comentarios de los jugadores

Tabla 13.1: Comentarios agrupados de los jugadores sobre el comportamiento de los enemigos en el videojuego

13.3. Técnicas populares

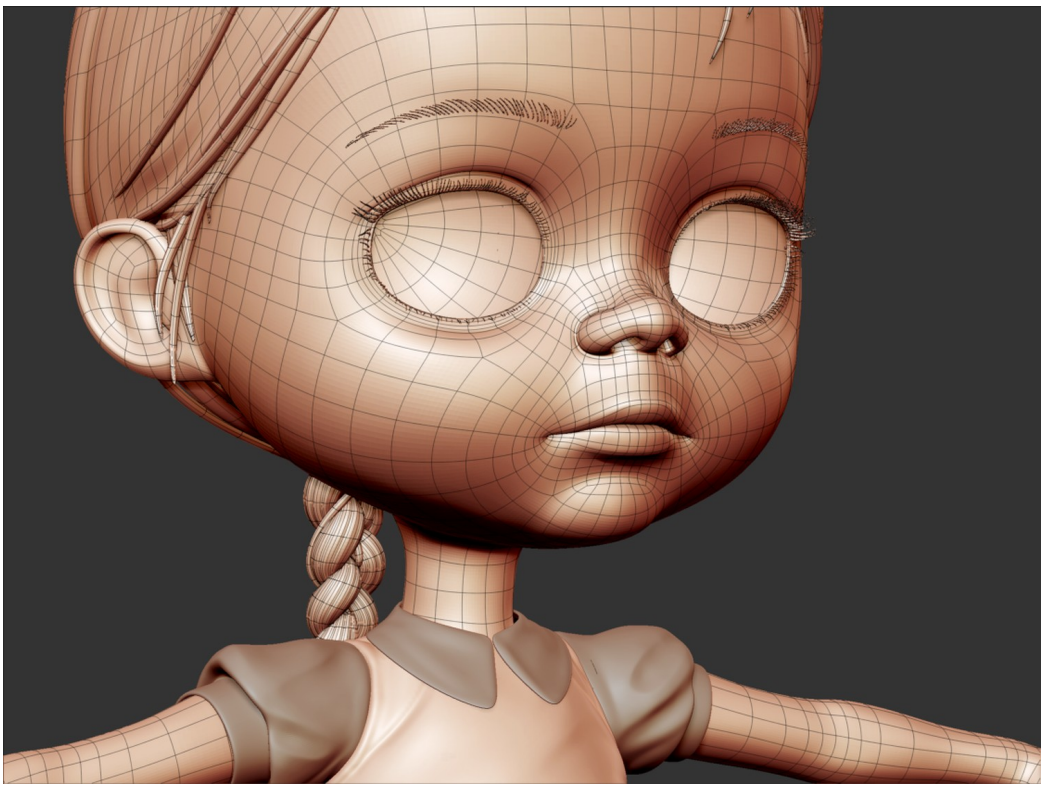

Figura 13.9: Modelado poligonal

Figura 13.10: Modelado basado en imágenes

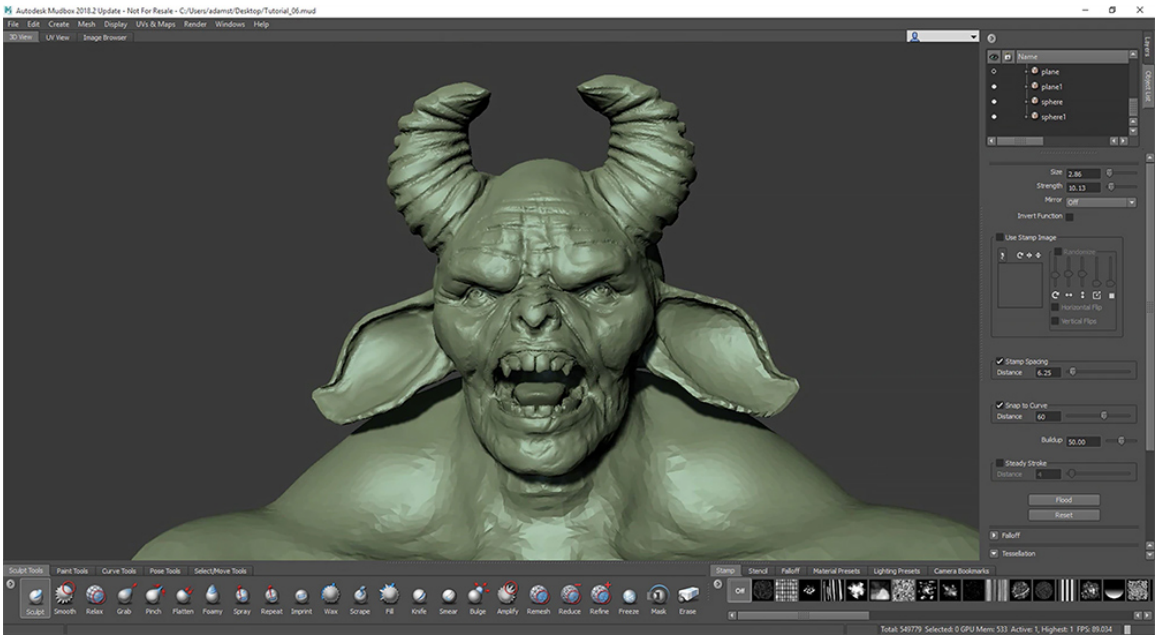

Figura 13.11: Modelado de escultura

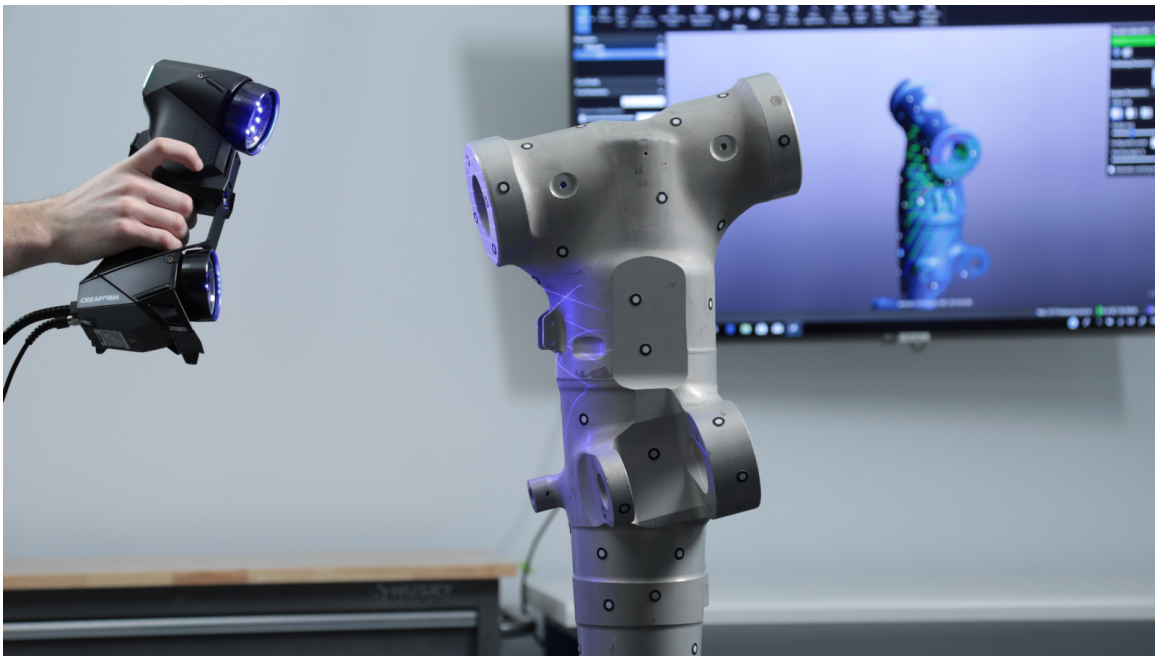

Figura 13.12: Modelado de escaneo

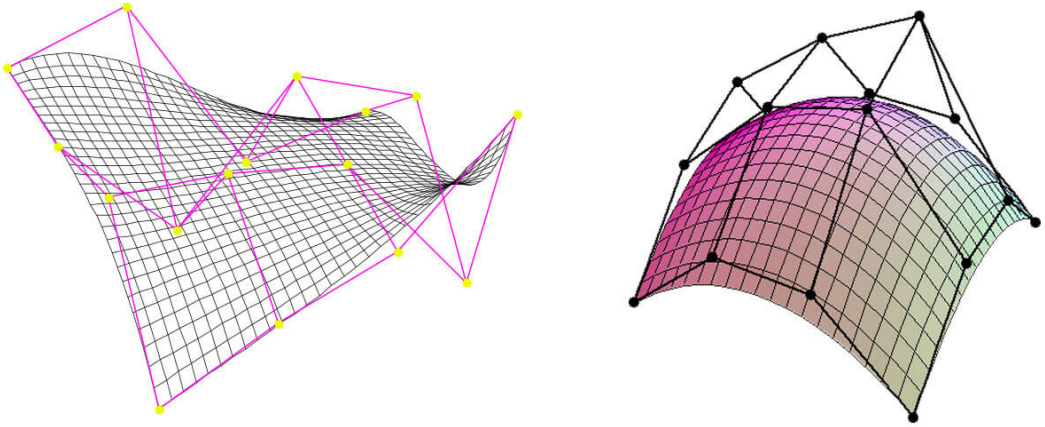

Figura 13.13: Modelado con Nurbs

Figura 13.14: Modelado procedural

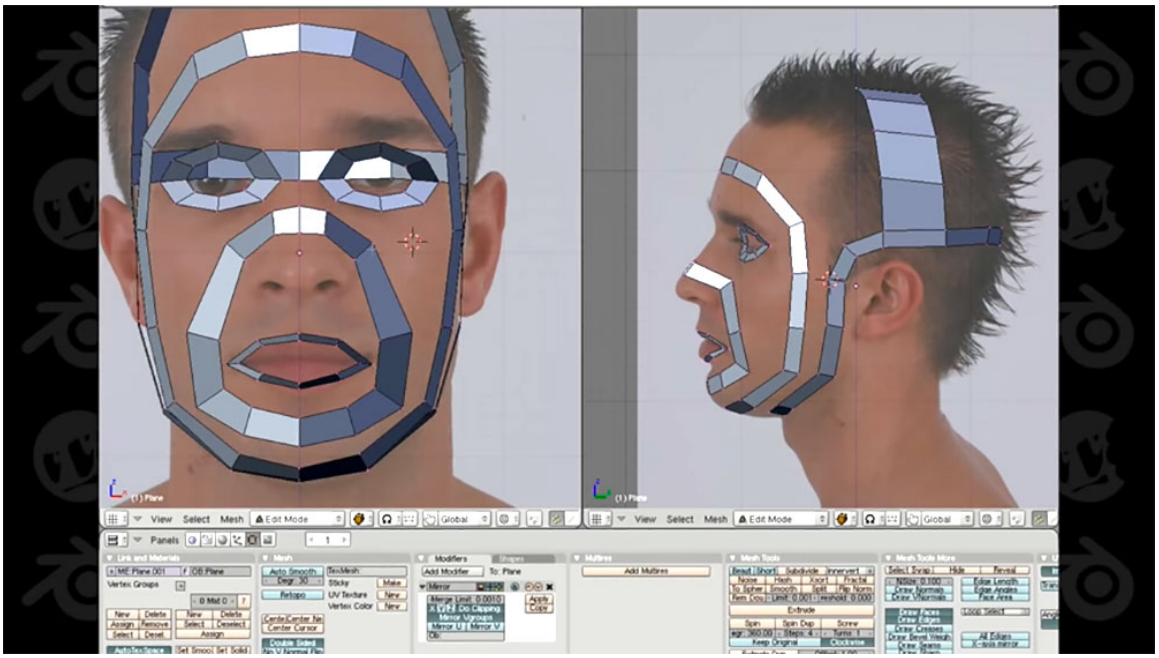

Figura 13.15: Modelado de bordes

Figura 13.16: Kitbashing

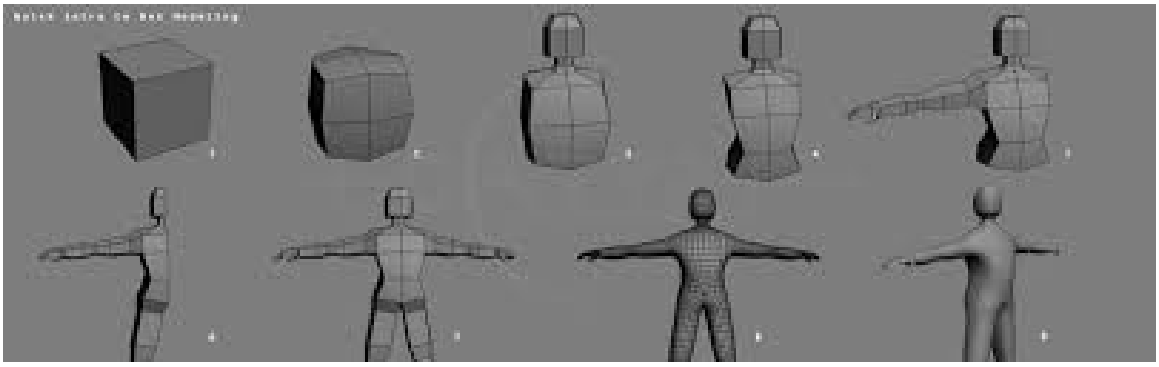

Figura 13.17: Modelado de caja/subdivisión

13.4. Diagrama de Gantt

Fecha de Inicio	Actividades	Nov	Dic	Enero	Feb	Marzo	Abril	Mayo	Junio	Julio	Agos	Sep	Octubre
27 al 29 de noviembre	Planificar este mismo cronograma (Empezar a poner las actividades a llevar a cabo y las fechas correspondientes (27 al 29 de noviembre)												
29 de noviembre	Hablar con Alhvi para que pueda ser nuestra asesora, mostrarle toda la idea y flujo de trabajo detrás. 29 de noviembre.												
30 de noviembre	Hablar con los otros profesores que podrían ser asesores y darles el contexto del proyecto.												
1 de diciembre	Reunión con los integrantes para explicar los siguientes pasos.												
1 al 4 de diciembre	Dejar como "tarea" las investigaciones propuestas y la lluvia de ideas.												
4 de diciembre	Reunirnos con el grupo para hablar sobre lo que encontramos y observamos con las investigaciones, también empezar a definir la lluvia de ideas.												
	Si es necesario dejar como tarea pulir mejor la lluvia de ideas con cada uno de los integreantes. (4 - 6 de diciembre).												
11 de diciembre	Reunirnos con el team para hablar sobre las ideas finales y en base a ello filtrar y empezar a definir la idea central del videojuego												
13 de diciembre	Empezar a definir el game design en base a la idea central y principal del videojuego												
12 de enero	Cuando se tenga el game design listo mostrar la idea del proyecto, objetivos, visiones y el game design para hablar sobre lo que tratará el juego para que nos aprueben la idea o que nos den ideas, sugerencias, arreglar elementos en específico o entre otros.												
5 de febrero	Realizar Reunión para detallar el flujo de trabajo para el prototipo.												
20 de febrero	Inicio oficial del desarrollo del prototipo V1.												
2 de marzo	Investigación y diseño de bases de datos para API y página web para el videojuego.												
5 de marzo	Propuesta y oficialización de las mecánicas propuestas para el gameplay de shooter.												
9 de marzo	Diseño y planificación del nivel 1.												
11 de marzo	Oficializar los modulos finales para cada integrante.												
17 de marzo	Documento de diseño de los personajes y enemigos.												
24 de marzo	Crear diagramas de administración del proyecto.												
25 de marzo	Creación de diseño de sonidos y animaciones.												
4 de abril	Diseño UI / UX para el videojuego en figma.												
4 de abril	Implementación de todos los elementos desarrollados.												
5 de abril	Reunión con los músicos que participaran y asignación de tareas.												
11 de abril	Implementación de FMOD para Unity.												
17 de abril	Implementación del nivel 1 para el prototipo del nivel 1.												
22 de abril	Implementación de Game Managers para la correcta conexión entre todos los elementos para el prototipo.												
22 de abril a 29 de abril	Pulir, mejorar gampelay y arreglar posibles bugs.												
26 de abril	Sesión de testing del prototipo 1 y demostración de avances para los asesores.												
29 de abril	Muestra del protitpo 1 con las mejoras mencionadas.												
1 de mayo	Comienzo de la segunda etapa para desarrollar el prototipo versión 2.												
13 de mayo	Semana de testing del segundo prototipo.												
18 de mayo	Arreglar posibles bugs encontrados en el prototipo 2.												
21 de mayo	Entrega y presentación del prototipo con el nivel 1 listo.												
29 de mayo	Revisar feedback de pruebas y comienzo del diseño del nivel 2.												
5 de junio	Retoques en el diseño del nivel 2 y comienzo del desarrollo.												
10 de junio	Diseño de estrategías de marketing en conjunto con las redes sociales, material audiovisual y videos promocionales.												
21 de junio	Pruebas y retroalimentación con el protipo del nivel 2.												
26 de julio	Presentación del prototipo 3 del proyecto.												
2 de agosto	Primer trailer del juego en las redes sociales y en la página web.												
23 de agosto	Presentación del prototipo 4 del proyecto.												
25 de septiembre	Presentación del prototipo 5 del proyecto.												
18 de octubre	Presentación Versión final del videojuego.												
19 de octubre	Feedback, pruebas y correción de bugs y/o detalles.												
28 de octubre	Lanzamiento del videojuego en las plataformas seleccionadas (Versión Beta).												

Figura 13.18: Diagrama de Gantt inicial

CAPÍTULO 14

Glosario

barringtonia Nombre de la munición principal del juego, que se utilizará en ambos niveles.. 81, 84

feedback El feedback en videojuegos es la respuesta inmediata del juego a las acciones del jugador, ayudando a que se sienta inmerso y conectado. Las animaciones del personaje también son una forma de feedback: si recibe daño, muestra dolor o retrocede; si dispara, muestra un retroceso de la simulación del disparo etc.. 181