UNIVERSIDAD DE TALCA
FACULTAD DE INGENIERÍA
ESCUELA DE INGENIERÍA EN DESARROLLO DE VIDEOJUEGOS Y REALIDAD VIRTUAL

Videojuego con dificultad dinámica en base a la destreza del jugador en un ambiente no competitivo.

IGNACIO FABIÁN GAJARDO ORTIZ

Profesor Guía: PABLO IGNACIO ROJAS VALDÉS

Memoria para optar al título de
Ingeniero en Desarrollo de Videojuegos y Realidad Virtual

Talca – Chile
Septiembre, 2021

CONSTANCIA

La Dirección del Sistema de Bibliotecas a través de su unidad de procesos técnicos certifica que el autor del siguiente trabajo de titulación ha firmado su autorización para la reproducción en forma total o parcial e ilimitada del mismo.

Talca, 2021

UNIVERSIDAD DE TALCA
FACULTAD DE INGENIERÍA
ESCUELA DE INGENIERÍA EN DESARROLLO DE VIDEOJUEGOS Y REALIDAD VIRTUAL

Videojuego con dificultad dinámica en base a la destreza del jugador en un ambiente no competitivo.

IGNACIO FABIÁN GAJARDO ORTIZ

Profesor Guía: PABLO IGNACIO ROJAS VALDÉS

Profesor Oponente: FELIPE ANDRES BESOAIN PINO

Miembro Comisión Examinadora: CLAUDIA ELIZABETH DE LA FUENTE BURDILES

Memoria para optar al título de
Ingeniero en Desarrollo de Videojuegos y Realidad Virtual

El presente documento fue calificado con nota: _____

Talca – Chile

Septiembre, 2021

Este documento está dedicado a mi familia, por apoyarme durante toda mi vida especialmente durante mi formación profesional y a todos los profesores, profesoras, estudiantes, amigos y amigas que tuve el agrado de conocer durante mi periodo universitario.

AGRADECIMIENTOS

El presente trabajo de memoria tecnológica fue realizado bajo la supervisión del profesor Pablo Ignacio Rojas Valdés, al cual deseo agradecer profundamente por hacer esta memoria posible.
Gracias a todas las personas que participaron en las pruebas del proyecto, ya sean usuarios anónimos que entregaron un poco de su tiempo para probar un reflejo de meses de trabajo, y especialmente a conocidos y amigos que incondicionalmente entregaron su apoyo cuando fue requerido.

TABLA DE CONTENIDOS

página

ÍNDICE DE FIGURAS

página

ÍNDICE DE TABLAS

página

RESUMEN

Con el paso de los años y las distintas generaciones se ha presentado con mayor frecuencia el progreso de dificultad en los videojuegos como factor en su diseño, en muchos casos siendo el punto de ventas de ciertas franquicias o la razón de su fracaso. Si bien existen distintos factores que influyen en diseñar la mejor dificultad para el jugador y/o producto, uno de los puntos más importantes es el *gameplay* en sí, específicamente, el control que el jugador tiene disponible para completar un nivel o vencer un enemigo. Una de las tareas de un diseñador de videojuegos es crear una ruta de aprendizaje para el usuario, donde las mecánicas son presentadas y entrenadas acorde a la dificultad con la intención de mantener al jugador envuelto su progresión. Si bien la mayoría de los juegos entregan recompensas por completar obstáculos y misiones ¿Cuántos juegos presentan el dominio de mecánicas como desafío y su obtención como recompensa? Este trabajo consiste en la creación de un videojuego de plataformas en 2D el cual evalúa el uso de mecánicas del jugador, entregándoles una valoración invisible para definir si esta ha sido dominada, y luego de completar una cantidad de niveles en forma de salas el jugador recibe una nueva habilidad o una mejora a una ya existente en base a la evaluación realizada.

El alcance proyectado consiste en el lanzamiento de un demo público con una duración aproximada de 20 minutos que cuente con las mecánicas principales encontradas en el diseño, estas siendo el sistema de niveles separados en salas, la evaluación de destreza del jugador y la mejora y adición de habilidades en base a esta evaluación. Además de un énfasis en la fase de recolección de datos posteriór al lanzamiento donde se llevan a cabo dos tipos de análisis, estas siendo analisis de datos por medio de alogritmos de clasificación y preguntas directas a usuarios por medio de encuestas. Utilizando procesos de KDD (*Knowledge Discovery in Databases*) en estos datos junto con el perfil del usuario entregarán la información necesaria para llegar a un futuro producto completo y satisfactorio. El proyecto cuenta con un gran foco pruebas de mecánicas y progresión de juego como su punto de desarrollo principal. Específicamente hablando del aspecto de dominio de mecánicas y su uso como recompensa dentro del juego, un área que no se encuentra explorada en juegos de un jugador pero es usualmente vista en tipos de juego donde la competencia es su objetivo principal.

1. Introducción

1.1. Motivación

Desde la concepción de los videojuegos han existido dos conceptos que se consolidan como necesarios en cuanto su creación, los cuales son dificultad y recompensa. Por un lado, la dificultad se define como la distribución progresiva de obstáculos que el jugador debe completar para avanzar por el videojuego de forma balanceada con el propósito de mantener al usuario entretenido, maximizando su motivación para completarlo o volver a jugarlo [1]. Por otro lado, recompensa se entiende como el premio obtenido resultado de completar los desafíos que son presentados en la dificultad diseñada, con premios variando entre obtención de puntajes (experiencia y valoración), desbloqueos de mecánicas, adquisición de recursos dentro del juego, contenido audiovisual y avances en la trama del juego [12]. Los conceptos de dificultad y recompensa pueden afectar directamente al diseño del videojuego para los desarrolladores de forma directa añadiendo nuevo contenido o indirecta mediante la entrega de logros y coleccionables. Aunque estos puntos han son encontrados en la mayoría de los videojuegos existentes, su versatilidad, conceptualización y propósito han creado distintas formas de ser implementados y trabajados por desarrolladores como motivador de juego, demostrando de que aún existen formas de ser mejorados [13]. Conociendo que los conceptos de dificultad y recompensa han tenido una evolución en el tiempo, estos son considerados la motivación principal para el trabajo de memoria y reflejarlos en proyecto con una envergadura acorde a los tiempos definidos para el proceso de titulación e implementarlos como base de ideas para la creación de un videojuego, además de pulir las habilidades de desarrollo en motores de videojuego de dos dimensiones.

En la actualidad el desarrollo de videojuegos se encuentra en su punto más alto en cuanto al tamaño de mano de obra y recursos monetarios [1]. Si bien esto demuestra el gran crecimiento de la industria en los últimos años, también presiona a los desarrolladores independientes que no disponen de los mismos recursos a buscar formas alternativas de competir.

Existen distintos factores que limitan los objetivos esperados en proyectos de cualquier escala, específicamente relacionados al tamaño de un equipo de desarrollo, falta de apoyo monetario y tiempo. Es por esto al tomar en cuenta estos factores la propuesta de crear un producto completo de calidad comercial es descartada por una opción más condensada.

1.2. Solución Propuesta

Teniendo en consideración las restricciones de tiempo del trabajo de titulación se determinó en realizar un demo jugable con el objetivo de funcionar como prueba de concepto para posible videojuego que a futuro implemente las ideas de dificultad y recompensa como núcleo de desarrollo. El demo propuesto cuenta con 5 niveles, 3 tipos de enemigos y 6 habilidades desbloqueables como objetivos del videojuego. Para realizar esta prueba de concepto se definió que posteriormente al lanzamiento del videojuego se deberá realizar un proceso de minería de datos con la aplicación publicada, la cual consiste en probar obtener resultados de partidas con usuarios reales y opiniones en cuanto al demo lanzado.
Esta demo tiene dos propósitos:

- Validar que, por medio de los datos recolectados automáticamente por el juego, clasificar la habilidad de los jugadores en el uso de las mecánicas con tres categorías: bueno, intermedio, malo.

- Obtener opiniones en cuanto al proyecto relacionadas a su aspecto gráfico, sonoro, mecánicas, dificultad y entretención en general.

[1] AAA Games. Everything You Need to Know About the Budget https://kevurugames.com/blog/why-are-aaa-games-so-expensive-everything-you-need-to-know-about-the-budget/

1.3. Objetivos

1.3.1. General

Diseñar, desarrollar y validar una demo de un videojuego de plataformas en 2D que implemente los elementos de dificultad y recompensa y clasifique a los usuarios según su nivel de experticia utilizando los datos de juego extraídos de usuarios únicos, y así usar esta información para a futuro implementar en el proyecto una dificultad adaptiva en base a la habilidad del usuario.

1.3.2. Específicos

- Creación de un documento de planificación del proyecto considerando la duración del ramo de memoria, que cuente con los elementos necesarios para el desarrollo del videojuego (niveles, enemigos, mecánicas, etc.).
- Creación de un documento de diseño que defina las características del videojuego.
- Definición de la metodología de trabajo del proyecto.
- Desarrollo del videojuego finalizando con la versión demo para ser publicada.
- Lanzamiento de videojuego en medios de distribución gratuitos.
- Extracción de datos con usuarios reales.
- Análisis de los datos encontrados en la fase de extracción.
- Conclusión del proyecto en base a los resultados encontrados durante la producción y el análisis de datos.
- Redacción del informe de memoria integrando la información obtenida.

1.4. Alcance

Tomando en cuenta las restricciones expresadas al momento de planificar el proyecto en cuestión, el alcance del proyecto tiene como meta que el videojuego a desarrollar finalice con un demo jugable, lo cual como desarrollador implica un producto

que contenga las mecánicas necesarias para comprender las características principales del videojuego, contenido desbloqueable dentro del juego que entregue una duración relativamente corta de juego (aproximadamente de 10 a 20 minutos) y una cantidad de elementos gráficos y sonoros suficientes para convalidar las ideas planteadas en la fase de diseño.

Así mismo, en el periodo posterior al lanzamiento del videojuego se espera tener una cantidad de datos a analizar cercano a 30 usuarios únicos, con la condición de que dichos jugadores que no tengan conocimiento previo al desarrollo del juego para asegurar que estos sean imparciales y reales.

1.5. Estructura del documento

Este documento cuenta con las distintas investigaciones, trabajos y resoluciones encontradas en las distintas fases del desarrollo y obtención de datos, las cuales fueron segmentadas y ordenadas de acuerdo con su orden de realización y su comodidad en cuanto a la lectura de la memoria.

El capítulo de **Estado del arte** muestra ejemplos de videojuegos que formaron la inspiración al momento de definir las distintas características del proyecto, contando con comparaciones en como estos videojuegos se asemejan y diferencian entre ellos y el videojuego desarrollado.

El **Marco Teórico** presenta la investigación y recopilación de antecedentes realizada para el desarrollo del proyecto, lo cual incluye una inmersión en las distintas áreas que forman el desarrollo de videojuegos, la creación de bases de datos para almacenar información conseguida de forma remota junto con su análisis de las variables obtenidas, una introducción al uso de encuestas en líneas para investigaciones y un estudio de las distintas metodologías de desarrollo de software existentes.

En **Metodología** se puede apreciar la implementación del modelo de desarrollo seleccionado y como este fue adaptado para el desarrollo del videojuego.

En el capitulo titulado **Diseño** se explica la construcción de las ideas del proyecto previas a su desarrollo práctico, centrándose en los elementos relacionados las

mecánicas, estéticas y niveles del videojuego, y que herramientas fueron seleccionadas para trabajar dichas características.

La sección de **Preproducción** muestra la organización de las tareas realizadas posteriormente en la fase de desarrollo, aquí se presenta una línea de tiempo basada en la división de fases y factores definieron su orden.

El capítulo denominado **Implementación** incluye toda la documentación relacionada al trabajo práctico realizado en el desarrollo del videojuego y la recopilación de información por medio de minería de datos y la encuesta.

Análisis de datos y resultados demuestra como fue realizada la evaluación de datos con sus etapas en cuanto a la evaluación de partidas, la limpieza de datos y como estos fueron analizados. Además, se presentan los resultados obtenidos al obtener las respuestas de usuarios por medio de la encuesta.

Finalmente, en **Conclusiones** se comenta como fueron estimados los resultados obtenidos en cuanto al producto final desarrollado y la evaluación de usuarios posterior. Además, se concluye explicando las posibles direcciones en la cual el proyecto y/o los conocimientos aprendidos de este se pueden expandir e implementar a futuro.

2. Estado del arte

En el desarrollo de videojuegos es necesario encontrar y analizar los distintos videojuegos pasados y la competencia actual para tener una idea de cómo implementar las ideas del proyecto de en base a ejemplos reales y entender cómo obtener un resultado favorable. En el estado del arte se presentan distintos videojuegos con características similares al proyecto final esperado que sirven de inspiración para el proyecto y como ejemplos al distinguir las particularidades principales, mecánicas y apartados visuales. Si bien ninguna de las características analizadas puede ser lo suficientemente semejante en el proyecto para asegurar un éxito similar, es importante comparar las distintas mecánicas y elementos de diseño presentes en la industria para tener una idea de cómo enfrentar el desarrollo del proyecto.

2.1. Megaman X

Cuadro 2.1: Información videojuego Megaman X

Plataforma	Super Nintendo
Desarrollador	Capcom, Nintendo (EU)
Publicador	Capcom

Es un videojuego del género plataformas 2D donde el jugador luego de finalizar el nivel introductorio debe completar 8 niveles sin un orden predeterminado y vencer a sus respectivos jefes, que al ser destruidos le entregan al jugador nuevos poderes para completar el juego. MegaMan X (MMX) es conocido y respetado por sus controles responsivos y la libertad que entrega para manejarse en los distintos niveles.

Figura 2.1: Captura del videojuego Megaman X

En cuanto al estado del arte, MMX implementa el concepto de "mejoras" de mecánicas en capsulas escondidas en distintos niveles, estas pueden entregar nuevas habilidades como el poder de realizar pequeños impulsos de moviemiento y mejoras directas como recibir menos daño. Aun así, el objetivo de estas mejoras es recompensar exploración y rejugabilidad, y no contiene ninguna mecánica de "dominación" de habilidades.

2.2. Hades

Cuadro 2.2: Información videojuego Hades

Plataforma	Steam, Epic Games Store, Nintendo Switch
Desarrollador	Supergiant Games
Publicador	Supergiant Games

Hades es un videojuego de acción *rogue-like* con vista isométrica donde se transita por distintas salas creadas de forma procedural y obteniendo mejoras de poderes de forma al azar al completar dichas salas, siendo estas cualidades su enfoque el diseño de juego promueve al jugador realizar numerosas partidas entregándole mejoras

Figura 2.2: Imagen promocional de Hades

permanentes y múltiples modos de juego.
Las mecánicas destacadas son el sistema de salas y los poderes entregados por su competición, siendo su segmentación el núcleo principal de su diseño en cuanto a balance en dificultad y punto de interés para el jugador.

2.3. Cave Story

Figura 2.3: Imagen promocional de Cave Story

Cuadro 2.3: Información videojuego Cave Story

Plataforma	Windows, PSP, Wii, Nintendo Switch
Desarrollador	Studio Pixel, Nicalis
Publicador	Studio Pixel, Nicalis

Cave Story es un videojuego *indie* 2D con elementos del género *metroidvania*, donde el jugador desbloquea distintas áreas y armas mientras avanza la historia del juego.

Además de ser un juego *indie* con un arte *pixel art*, su mecánica más destacable es la de mejoras de armas, esta consiste en que todas las armas encontradas en el juego tienen una barra de experiencia y un sistema de niveles, el cual aumenta al tomar las pirámides que los enemigos sueltan al ser destruidos. Subir de nivel un arma mejora su daño y en algunos casos cambia sus características, pero la diferencia recae en que este sistema se encuentra limitado a armas las cuales todas funcionan como proyectiles y no incluye mecánicas de movimiento ni mecánicas defensivas.

Cuadro 2.4: Comparación de características

	MegaMan X	**Hades**	**Cave Story**	**Proyecto**
Género	Plataformas	*Roguelike*	Plataformas	Plataformas
Visual	*Pixel art*	3D	*Pixel art*	*Pixel art*
Obtención de habilidades	Exploración	Compleción de salas	Experiencia	Uso de habilidades

3. Marco Teórico

En este segmento definido como marco teórico se presentará todo el contenido previo a la producción del videojuego, específicamente la información relacionada a la teoría de las dos áreas principales del proyecto en general, estas siendo el desarrollo del videojuego y el trabajo de recolección de datos y su análisis.

3.1. Desarrollo Videojuego

Mayoritariamente, se acuerda que el desarrollo de videojuegos se segmenta en las distintas áreas que implican el juego como producto completo, desde los pilares principales como programación, diseño y arte hasta factores externos que pueden afectar el producto como publicación, compatibilidad de hardware, etc [1].
Específicamente hablando de las áreas de programación, diseño y arte, estas son estructuradas en equipos de desarrollo, el cual involucra un grupo de desarrolladores organizados por una jerarquía determinada por roles[2]. Si bien estas definiciones son claramente vistas en las grandes empresas de desarrollo de videojuegos, la mayoría de los proyectos del género *indie* son producidos por un grupo limitado de integrantes exigiendo participar de forma multidisciplinaria; especialmente tomando en cuenta que el videojuego presentado en este documento cuenta con un solo desarrollador, se usará el término para presentar las distintas partes que forman la producción de un juego para facilitar la emisión de su información.

[1] How video games are made: the game development process https://www.cgspectrum.com/blog/game-development-process

3.1.1. Diseño de videojuegos

El área de diseño de videojuegos es la más particular comparado con otras artes, solamente encontrada en este medio gracias a su interactividad. El diseño de videojuegos abarca exactamente todos los factores que afectan la interactividad del usuario[16], en términos simples las misiones del área de diseño son, principalmente, la definición de mecánicas principales y secundarias, el balanceo de la dificultad, la construcción de niveles, organización de objetivos y sus respectivas recompensas y, podría decirse la más importante en términos internos al desarrollo, asegurar la cohesión entre las otras áreas de producción y la visión del proyecto. El diseño no solo implica mantener una visión acorde al producto final esperado en base a la propuesta entregada en la preproducción, también requiere un vasto conocimiento en las áreas paralelas al arte y programación ya que en el caso de que estas no sean compatibles con los conceptos presentados en el diseño deben ser corregidos los antes posible. Entre las subsecciones de diseño podemos encontrar:

Diseño Principal: Esta rama de diseño abarca la idea inicial que comienza el proceso de producción, estableciendo los principales conceptos estéticos y mecánicos que afectan a todo el equipo de desarrollo, por lo que cambios en estos conceptos también implican una alteración en todo el proyecto.

Mecánicas de juego: Las mecánicas del juego refieren a las herramientas que crean situaciones de *gameplay* e interacciones entre el jugador y los otros elementos del juego. Usualmente trabajar en mecánicas requiere un conocimiento en programación para asegurar de que su implementación sea correcta en relación con la dirección del juego.

Diseño de niveles/misiones: El diseño de niveles y/o misiones es una forma simplificada de definir la extensión de desafíos que el juego debe presentar al jugador, ya sea en forma de niveles que el jugador debe transitar o misiones que este debe completar[15].

3.1.2. Programación

La programación abarca llevar a cabo de forma práctica todas las ideas encontradas en el área de diseño. La implementación de mecánicas a base de código y el uso de motores de juego para la reproducción de físicas y mundos a crear, además de implementación de *assets* y corrección de errores son parte de las tareas del equipo de programación.
Específicamente hablando de videojuegos, para ser desarrollados es necesario el uso de un motor de juego, que consiste en una arquitectura que permite el uso de herramientas para programar, generar contenido gráfico, configurar físicas e iniciar correctamente el juego. En este caso los desarrolladores existen dos opciones, programar su propio motor desde cero o utilizar un motor de juego ya existente que se encuentre disponible, donde ambas opciones implican sus propias ventajas y desventajas. Por un lado, crear un motor desde cero entrega una completa libertad del proyecto, pero toma mayor tiempo y es considerada una habilidad más allá del desarrollo en sí, mientras que usar un motor ya existente corta toda una parte del desarrollo al no tener que programar herramientas adicionales, pero también requiere tiempo para aprender a usar estos motores, especialmente en el caso que usen su propio lenguaje de programación.

Al momento de seleccionar el motor de desarrollo de videojuegos existente en el mercado es importante tener en cuenta varios aspectos que afectan el producto final como el desarrollo de este[2]. Entre estos se encuentran:

- **Licencias:** Al lanzar cualquier producto es importante determinar hasta qué punto uno es propietario de su videojuego, por lo que hay que conocer si existen tarifas a pagar al momento de usar cualquier tecnología. Estas pueden variar desde pagar un porcentaje de las ganancias hasta ser completamente libre de uso (también conocidas como *open-source*).

- **Documentación:** Una de las características cruciales al momento de elegir un motor de juego es la documentación disponible para su uso, especialmente al usar uno por primera vez. Entre más información se encuentre disponible, ya

[2]Choosing the right game engine https://www.gdquest.com/tutorial/getting-started/learn-to/choosing-a-game-engine/

sea encontrada en el soporte del distribuidor o en foros libres para usuarios es un requerimiento para instruirse en las peculiaridades de cada motor.

- **Editor:** A menos que se desee trabajar únicamente con código se debe seleccionar un motor que permita desarrollar con un editor visual. Estos tipos de motores entregan una forma más cómoda de trabajar, con cada motor implementando flujo de trabajo distinto a su desarrollo.
- **Lenguaje de programación:** Dependiendo de los conocimientos del desarrollador, se tendrá que trabajar con un motor de juego que sea compatible con el lenguaje de programación a elección del equipo de desarrollo. También, en el caso que no se tenga un integrante con vasto conocimiento de programación, se puede optar a un motor que cuente con un sistema de programación visual.
- **Rendimiento:** Si el proyecto a trabajar requiere que el hardware objetivo rinda una gran cantidad de datos con modelos detallados y ambientes enormes, o simplemente se desea hacer un proyecto con una escala reducida y una duración corta, existe un motor de juego que se adapte a sus necesidades.

Para las características de este proyecto, es mejor enfocarse en optar por un motor de juego que cuente con un editor gráfico para facilitar el trabajo, ya que el desarrollo del proyecto cuenta con tareas adicionales a la programación, una abundante documentación para hacer su aprendizaje expedito y un rendimiento relativamente bajo ya que los recursos usados en un videojuego en 2D no requieren de procesos pesados y complicados en comparación a trabajar en 3D.

3.1.3. Arte

La sección de arte abarca toda la parte visual del proyecto, específicamente se explaya el área conceptual del diseño donde se definen las temáticas y los principios que estos conllevan hasta el estilo que los *assets* llevan ya insertados dentro del juego, tomando en cuenta el diseño, programación y las otras áreas. Esta etapa es muy importante ya que, sin contar algunas excepciones, los videojuegos son considerados un arte visual y de por sí este se transforma en uno de los puntos a destacar al momento de evaluar un proyecto como viable.

3.1.4. Sonido

El otro apartado artístico y técnico encontrado en los videojuegos es el área de los *assets* de sonido. Principalmente en videojuegos el sonido es segregado en dos secciones separadas pero conectadas a la vez. La primera siendo los efectos de sonido, que funcionan como un complemento al área visual con la idea de representar correctamente una acción llevada o reacción ocurrida. Mientras que la segunda hace referencia a la música encontrada en el juego, que tiene la misión de camuflarse con el ambiente del videojuego con la intención de provocar un cierto sentimiento al jugador, como motivación, miedo, pena, entre otros.

3.2. Recopilación de datos

Para mejorar la experiencia de juego en el futuro, el proyecto también consta con una recopilación de datos luego de lanzar el juego de forma pública, donde los mismos usuarios ayudan al proceso de refinamiento con solo jugar.

3.2.1. Minería y análisis de datos

Esta recopilación de datos se refiere al descubrimiento y su posterior análisis de datos duros encontrado dentro del juego al realizar partidas. En este proyecto en particular, su potente uso de variables con cambios constantes hace el uso de herramientas relacionadas a esta área especialmente útiles, así que el trabajo requiere el uso de una base de datos y KDD (*Knowledge Discovery in Databases*).
Antes de realizar cualquier tipo de análisis se necesita almacenar las variables en sí y para esto se precisa el uso de base de datos, que en términos simples existe como un espacio que almacena datos de cualquier tipo facilitando su acceso y uso, funcionando como, por ejemplo, una biblioteca lo hace para libros, documentos, textos, etc [6]. Obviamente para poder ingresar y eliminar información se requiere seguir una lógica de orden al momento de comenzar a usar una base de datos, entre estos tipos se encuentran[3]:

- **Bases de datos relacionales:** Los elementos de una base de datos relacional se organizan como un conjunto de tablas formadas por columnas y filas.

[3]Definición de base de datos https://www.oracle.com/cl/database/what-is-database/

- **Bases de datos orientadas a objetos:** La información en una base de datos orientada a objetos se representa en forma de objetos, como en la programación orientada a objetos.
- **Bases de datos distribuidas:** Una base de datos distribuida consta de dos o más archivos ubicados en diferentes sitios y puede ser almacenada en múltiples computadoras, ubicadas en la misma ubicación física o en diferentes redes.
- **Almacenes de datos:** Un almacén de datos es un tipo de base de datos diseñada específicamente para consultas y análisis rápidos, y funciona como un depósito central de datos.
- **Bases de datos NoSQL:** También conocida como base de datos no relacional, permite que los datos no estructurados y semiestructurados se almacenen y manipulen, a diferencia de una base de datos relacional, que define cómo deben componerse todos los datos insertados en la base de datos.
- **Bases de datos orientadas a grafos:** Una base de datos orientada a grafos almacena datos en base a entidades y las relaciones entre entidades.
- **Bases de datos OLTP:** Es una base de datos diseñada para un gran número de transacciones realizadas por múltiples usuarios.

La gran ventaja de la base de datos en videojuegos es su uso para almacenar datos relacionados a partidas e información de usuarios, todo de forma compacta y remota. Esta información es altamente valorada al momento de realizar nuevas iteraciones de los proyectos ya que permite a los desarrolladores solucionar errores que no fueron encontrados previo a la publicación del videojuego, los cuales incluyen errores técnicos por medio de *bugs* y fallos que pueden interrumpir el juego y errores de diseño como balance de dificultad y problemas en el *Game Engagement* del usuario[9].
Al obtener una cantidad satisfactoria de datos obtenida de distintos jugadores el análisis de datos es realizado por *Knowledge Discovery in Databases*, definido como un proceso de identificación no trivial de patrones válidos y, principalmente, útiles en la información encontrada en la base de datos [4]. Al ser un proceso este requiere un número definido de pasos luego de obtener los datos brutos.

Selección de datos: Se reciben las variables encontradas en la base de datos y se seleccionan y ordenan las relevantes para el análisis requerido.

Limpieza de datos: La limpieza consiste en la verificación de que los datos sean reconocibles al momento del análisis.

Transformación de datos: Se transforman los datos a información simple (como variables numéricas) para que el algoritmo de análisis las reconozca fácilmente y las relacione con él resultado a buscar.

Reconocimiento de datos: Aquí es donde el resultado se transforma en conocimiento por medio de patrones que entrega el análisis.

3.2.2. Encuesta en línea

Para complementar la información entregada por la minería de datos, se necesita recibir cierta información que se considera subjetiva, especialmente si se requiere obtener información relacionada a la experiencia de usuario. Por esto, adicionalmente se complementa la información obtenida por la misma aplicación entregando una encuesta a los usuarios centrada en temas que varían de jugador a jugador como dificultad y entretención del juego, y también tener conocimiento de la opinión de los usuarios en cuanto a las mecánicas presentadas y el apartado gráfico y sonoro del videojuego[14].

Ventajas:

- No requiere costos mayores en cuanto a mano de obra, solo siendo necesario para crear las preguntas y la página web donde esta se llevará a cabo.
- Mayor accesibilidad al momento de importar los resultados.
- Aumenta la cantidad de posibles encuestados, al solo requerir una forma de acceder a internet para ingresar a la página web que hospeda la encuesta.
- No requiere mayor sustento, con herramientas gratis y de libre uso disponibles.

Desventajas:

- Los encuestados se pueden sentir obligados a inclinarse por una opción sobre la otra, ya sea porque la pregunta no se presenta correctamente planteada o requiere más información.
- Se debe mantener un equilibrio en la cantidad de preguntas, para obtener una cantidad de respuestas respetable sin comprometer la tasa de posibles encuestados.
- Es necesario limitar la cantidad de información personal que el encuestado debe entregar, para evitar abrumarlo y no alentar la entrega de datos falsos.

3.3. Metodología de desarrollo de software

En el desarrollo de software, la metodología de desarrollo consiste en la división del trabajo en procesos reducidos con la intención de tener una mejor administración del proyecto en general y definir objetivos continuos para que todos los integrantes del equipo de desarrollo se encuentren informados de la fase en la cual se encuentra el proyecto[5].
Existen distintos tipos de metodologías de desarrollo, por lo que se debe elegir correctamente cuál de estas es la mejor para el proyecto a realizar. Entre las más populares se encuentran:

- **Metodología Ágil:** Metodología en la cual se caracteriza por adaptar y cambiar las soluciones y objetivos de las tareas con el tiempo y según la evolución del proyecto, todo esto con la intención de incentivar la cooperación del equipo de desarrollo y tener al cliente como prioridad.
- **Modelo Cascada:** El modelo de cascada consiste tener un flujo prestablecido de tareas que se realizan en un orden estricto, por lo que no se puede avanzar a una fase de desarrollo si la finalización de la fase previa no ha sido aprobada. El modelo clásico sigue el orden de análisis, diseño, implementación, pruebas, despliegue y mantenimiento.
- **Modelo Iterativo:** Derivado del modelo de cascada, el modelo iterativo consiste en la continua creación de prototipos que son probados y analizados de

forma iterativa para ser mejorados en la siguiente versión implementando cambios y refinamientos, con una evolución constante del proyecto y manteniendo una gran cantidad de versiones del mismo producto.

- **Modelo en Espiral:** Una combinación del modelo iterativo y el modelo de cascada, el modelo espiral consiste en la realización de tareas de forma cíclica, pasando a la primera tarea después de finalizar la última haciendo que el proyecto crezca ciclo por ciclo y evolucione de forma incremental en cada iteración con tareas definidas. Las fases básicas del modelo son planificación, análisis de riesgo, implementación y evaluación.

Tomando en cuenta que el desarrollo de videojuegos es multidisciplinario y que el desarrollo de este proyecto cuenta con un solo integrante, al conocer y analizar los modelos mencionados se seleccionó usar el modelo espiral como base para trabajar en las tareas del proyecto, con modificaciones que se adapten a los puntos mencionados relacionados al desarrollo de videojuegos y la cantidad de integrantes en el proyecto. Así se puede iniciar el desarrollo trabajando exclusivamente en las características básicas y principales del proyecto para luego pasar a las tareas que integran las mecánicas distintivas del videojuego, teniendo una versión del modelo para programación y otro para el apartado artístico.

4. Metodología

En esta sección se presentan los distintos pasos seguidos en la metodología usada en la construcción del proyecto, específicamente los pasos llevados en el modelo de espiral que fue adaptado para funcionar específicamente en el desarrollo del videojuego. Debido a que el desarrollo del videojuego se llevó a cabo por una sola persona las cuatro fases de espiral se llevaron a cabo de forma continua y en base a la planificación de tareas.

4.1. Diseño

En esta fase se diseña la tarea a realizar, que dependiendo si esta conlleva programación o arte se desarrolla de forma distinta. En cuanto a programación primero se identifica la siguiente mecánica necesaria en el proyecto a implementar, luego se define si esta cambia o afecta una mecánica programada previamente y si alguna función o elemento es necesario para su implementación. En el caso de trabajar en la parte artística del juego, el primer paso es identificar donde el elemento visual se encuentra al momento de ser implementado para definir tamaño y colores, definir si necesita algún tipo de animación y finalmente si alguna mecánica o aspecto de programación puede afectarla.

4.2. Programación / Arte

Como su nombre lo indica, esta es la parte práctica de la metodología, donde se trabaja lo planteado en la fase de diseño con todos los requerimientos definidos. Esta etapa se trabajó de forma aislada, es decir, se requirió completar las tareas

fuera del proyecto que se esperaba subir al finalizar la etapa, en programación esto significó probar que esta funcione en una escena del motor separada del proyecto antes de integrarla, mientras que en el arte del videojuego fue posible tener una idea del resultado final usando la herramienta de dibujo y probar su funcionalidad en la escena de pruebas mencionada.

4.3. Integración

Al corroborar de que el trabajo realizado funcionaba por sí solo, este se integra al proyecto principal y se arregla cualquier tipo de problema inmediato que puede ser identificado.

4.4. Testing

En la etapa final se hacen distintas pruebas del programa o arte implementado en el ambiente de trabajo del proyecto principal, específicamente se revisa como esta interactúa con el resto del proyecto y si algún problema que no fue identificado en las etapas de diseño o implementación aparece para pasar a solucionarlo. Al completar una cantidad de pruebas razonable sin problemas aparentes se da por completada la implementación y se pasa a la siguiente mecánica a programar o arte a integrar al proyecto.

5. Diseño

En este capítulo presenta la planificación y construcción del proyecto previa a su implementación. Principalmente esta porción del documento entrega una introducción a las mecánicas previstas en el proyecto con su implementación esperada, las reglas y herramientas utilizadas para el diseño de niveles en el juego, las decisiones estéticas que fueron seleccionadas como base al momento de trabajar en el aspecto gráfico del videojuego y una descripción de las herramientas utilizadas en el trabajo de los aspectos mencionadas.

5.1. Características principales

Antes de comenzar con cualquier tarea o planificación, es necesario concretar las características básicas para el juego, esto se refiere a las ideas iniciales del proyecto que definen concepto del videojuego ya que lógicamente no se puede definir un tipo de trabajo, periodos de desarrollo, áreas a destacar u objetivos sin tener una idea clara del producto final deseado, al menos en su estado más básico. Luego de analizar las limitaciones de tiempo, habilidades del desarrollador y requerimientos en el contexto de la memoria de título, la primera decisión fue trabajar en un juego en 2 dimensiones con un estilo de arte *pixel art* por temas de comodidad del desarrollador tomando en cuenta su relativa simplicidad en implementación y conocimiento previo en la técnica.
Así mismo, tomando estas características y comparándolas con los productos vistos en el estado de arte se consolidó el apuntar al género de plataformas, ya que este posibilita un mejor uso en las herramientas que motores entregan en diseño de niveles, y acción como subgénero para manejar su dificultad progresiva ya que al

integrar combate directo con enemigos se agrega una capa de mecánicas que pueden ser trabajadas y mejoradas.
Finalmente, la plataforma seleccionada fue PC, específicamente publicar el juego en compatible en el sistema operativo Windows ya que el proyecto sería trabajado en PC, siendo la plataforma en la cual se desarrolla el proyecto y se desea evitar la necesidad hacer cambios para transportarlo a consolas u otros sistemas operativos.

Cuadro 5.1: Ideas concretadas incialmente como base del proyecto

Características iniciales	
Plataforma	PC
Género	Platafomras/Acción
Estilo de arte	*Pixel art*
Niveles	2D

5.2. Mecánicas

Al momento de plantear posibles mecánicas se seleccionaron dos ideas principales que consolidaron la idea del proyecto, ambas relacionadas a los conceptos de dificultad y recompensa.

Dificultad dinámica en base a control del jugador: Como se puede apreciar en juegos multijugador-competitivos, la cantidad y complejidad de mecánicas aumenta el número de opciones y estrategias posibles del usuario, pero esta también tiene un impacto directo en la dificultad al incrementar la cantidad de variables del juego y pidiendo un mayor nivel de ejecución al jugador. La idea es llevar estos conceptos a un juego *single-player* en un ambiente no competitivo.

Entrenamiento y recompensa: La idea es unir estos dos conceptos al evaluar la destreza del jugador y su consecuente progresión en distintas habilidades encontradas en el juego, y al determinar de que la habilidad ha sido aprendida, recompensar al usuario con una expansión y/o mejora dicha habilidad la cual amplía las opciones de juego y su dificultad, creando una conexión reciproca entre ambas.

Al tener las características principales del proyecto y las ideas que el videojuego abarca se pasó a definir las mecánicas en un aspecto práctico. Para esto se clasificaron las mecánicas del juego en dos grupos, el grupo de mecánicas principales que refiere a las características claves usadas para describir términos propios del género y completar el contenido necesario en cuanto a dificultad y progresión, todas estas fueron implementadas basadas en la idea inicial. Por otro lado, el segundo grupo consiste en puntos de evaluación del personaje jugable que definen como y cuando las habilidades disponibles son obtenidas y/o mejoradas durante el juego.

Cuadro 5.2: Mecánicas principales que definen el proyecto

Mecánicas Principales	**Descripción**
Barra de vida	Numero finito de daño recibido y condición de derrota.
Salas de nivel	Nivel de plataformas y enemigos compacto, segmentación de locaciones encontradas.
Combate	Enemigos encontrados en niveles pueden ser atacados y destruidos.
Plataformas	Desafíos en el diseño de niveles que pone a prueba las habilidades de movimiento, son necesarias para completar los niveles.
Evaluación de habilidades	Cada vez que se usa una habilidad esta es evaluada, si su uso es satisfactorio esta puede ser mejorada.
Mejora de habilidades	Algunas habilidades al ser evaluadas positivamente son mejoradas al completar una cantidad definida de salas.
Obtención de habilidades	Algunas habilidades al ser evaluadas positivamente recompensan con una nueva al completar una cantidad definida de salas.
Guardado de habilidades	Si se cumple la condición de derrota, las habilidades obtenidas se mantienen en una nueva partida.

Como se puede apreciar, las primeras mecánicas principales presentadas en la tabla son estándar en el género de plataformas, incluyendo un sistema de vidas para tener una constante condición de derrota, combate con enemigos simples y niveles diseñados con plataformas para mantener un desafío constante donde el jugador debe usar sus habilidades para evitar la condición de derrota. Esto nos lleva a la otra parte de mecánicas que son específicas para el proyecto, la evaluación, mejora y obtención de habilidades hacen que el jugador no solo se encuentre motivado a intentar jugar lo más perfectamente posible, sino que también incita volver a jugar en caso de que su propia habilidad no le permita completar un nivel o quiera desbloquear una habilidad que no obtuvo en una partida previa.

Cuadro 5.3: Puntos de evaluación

Tipo	**Descripción**
Movimiento	Acción de desplazo en direcciones laterales y vertical por medio de saltos.
Ataque	Ataque cuerpo a cuerpo con un *hitbox* desconectado.
Proyectil	Ataque de proyectil a distancia.

Las tres características mencionadas son puntos en los cuales el jugador es evaluado, estas definen la efectividad de su uso y si estas son mejoradas. Movimiento analiza la proeza del jugador para completar niveles lo más rápido posible, mientras que ataque y proyectil se activan al dañar exitosamente a los enemigos con las habilidades mencionadas. Al inicio del proyecto se consideró la idea de situaciones donde los puntos de evaluación serian disminuidos, específicamente al completar un nivel en un tiempo sobre un máximo requerido para movimiento y al fallar ataques y proyectiles, pero se encontró que el balance del juego y la limitada duración de partidas impedía al jugador recibir habilidades de forma constante, por lo que se terminó descartando de la versión lanzada.

5.3. Estética

En esta sección se presentan las decisiones visuales que pasan a ser implementadas en el diseño del videojuego, específicamente una descripción del estilo artístico que fue seleccionado durante el periodo de preplanificación y la temática utilizada para diseñar los personajes y ambientes en el juego.

5.3.1. Temática

Se definió como base de la temática encontrada como el género de entretenimiento de origen japonés "Sentai". Estos son definidos como superhéroes en uniformes de colores con poderes que combaten fuerzas sobrenaturales (figura 5.1). Usualmente lo compone un grupo de fuerzas especiales [19], pero este proyecto se centra en las estéticas del género más que las reglas que lo definen. El personaje principal toma

Figura 5.1: Captura de la serie Power Rangers uno de los íconos mundiales del género Sentai

inspiración de los héroes "Sentai", específicamente las características más icónicas como el diseño de casco siendo de cabeza completa con un solo color, un visor que cubre completamente los ojos y una pieza de otro color en el área bucal, además de un uniforme que complementa el casco y otros accesorios característicos (como correas y botas). Un punto importante representa es que los héroes "Sentai" representan un color en el grupo, por lo que es necesario llevar una paleta de colores limitada al momento de diseñar al personaje principal.

5.3.2. Estilo de arte

El estilo de arte seleccionado es el *pixel art*, un medio de arte exclusivamente digital usado en consolas de generaciones pasadas por las limitaciones encontradas en estas en tamaños, colores, etc[18]. En la actualidad es usado, eliminando las res-

tricciones para modernizar y darle un nuevo estilo. Este se encuentra principalmente en videojuegos *indie*, ya que este requiere menos conocimiento técnico comparado a las alternativas tomando en cuenta los equipos de desarrollo, y se mantiene relevante por su estética agradable y nostálgica.

Figura 5.2: Ejemplo del estilo Pixel art moderno encontrado en el videojuego indie Fez

5.4. Niveles

La segmentación de niveles en el proyecto es diseñada en base a un sistema de etapas, el cual es usualmente encontrado en géneros de plataformas. Este consiste en la creación de distintos niveles con un predeterminado punto de inicio y un objetivo final físico encontrado en el nivel, con distintos obstáculos diseñados para ser encontrados por el jugador mientras transita la diferencia de espacio entre los dos puntos, siendo estos posibles obstáculos desafíos de movimiento mediante plataformas o enemigos que impiden el paso y dañan al personaje principal, los cuales deben ser evitados o destruidos. El uso de estos elementos permite la creación de patrones en diseño de niveles que guían al jugador y clarifican los objetivos y obstáculos presentados anteriormente[8]. Específicamente se quiere resaltar estos patrones:

- **Foreshadowing:** Este patrón se refiere al uso de objetos y situaciones para introducir nuevos elementos encontrados en el juego que serán utilizados posteriormente, con la idea de crear un tutorial integrado en el juego sin la necesidad de texto adicional.
- **Layering:** *Layering* es el uso de elementos encontrados anteriormente en el juego y combinarlos de distintas formas mientras se avanza en el juego, creando nuevas experiencias en el juego y variar su dificultad sin la necesidad de crear contenido adicional.
- **Branching:** Como su nombre lo indica, *branching* se refiere a entregarle al usuario distintas formas de completar su objetivo para que este se acomode su propio estilo de juego. La idea se encuentra en diseñar niveles los cuales puedan ser completados con todas las herramientas iniciales del juego y que sea decisión del usuario cual sea utilizada para su propio beneficio.

5.5. Herramientas

5.5.1. Motor de juego

En este tema desde el inicio se descartó la idea de crear un motor de juego propio, tomando en cuenta del tiempo de trabajo disponible siendo limitado junto con el hecho de que se debe trabajar simultáneamente en las otras áreas de desarrollo,

y el resultado esperado solicita entregar prioridades equitativas a todas estas. Al momento de seleccionar un motor de juego existente, se llegó a la conclusión de trabajar en el motor de juego "GameMaker Studio 2" esto se debe a distintas razones.

- **Lenguaje de GameMaker:** GameMaker Studio 2 cuenta con su propio lenguaje de programación (GML), basado en los conceptos encontrados en los lenguajes de C y javascript de programación imperativa y los adapta a su propio Entorno de Desarrollo Integrado (IDE) (figura 5.3) especializado para el desarrollo de videojuegos 2D[7].

- **Sistema de salas:** En GameMaker Studio 2 la estructura del juego se implementa por el sistema de salas propio del motor. Este funciona como un ambiente donde los objetos creados son posicionados y organizados de acuerdo con las necesidades del juego además de ser encargados de presentar la visualización de este[10]. El uso de salas es usado para segmentar los niveles implementados en el juego y las distintas pantallas de menús presentados al usuario.

- **Programación orientada a objetos:** Al igual que los lenguajes que GML se basa, se implementa el concepto de objetos como entidades que contienen propiedades entregadas por código para realizar alguna actividad especifica. Estos se presentan desde plataformas y efectos hasta los enemigos con su comportamiento y el control de menús.

- **Foros y comunidad:** GameMaker Studio 2 cuenta con una comunidad activa de usuarios dispuestos a ayudar y contribuir con contenido fuera del entregado por la misma plataforma por medio de foros y otros canales de la comunidad[22].

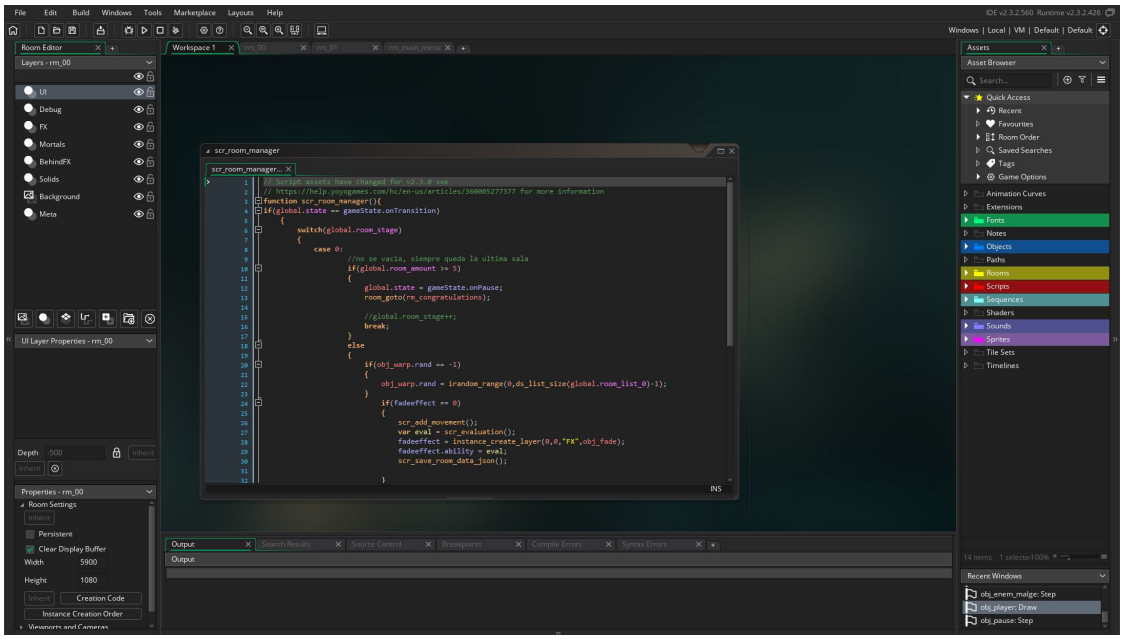

Figura 5.3: Captura de pantalla de Gamemaker 2 mostrando su IDE

5.5.2. Herramientas complementarias

Además del motor de juego se definieron las herramientas que serán utilizadas para complementar el trabajo, ya sea porque las características encontradas en este son limitadas para el objetivo necesario o simplemente inexistentes. Específicamente estas herramientas se centran en ayudar en las áreas de arte. Las herramientas fueron usadas para trabajar en la parte visual y sonora del juego.

Aseprite: Herramienta de dibujo y animación especializada en el estilo de arte conocido como *pixel art*. Elegida por su peso ligero de instalación y características como creación de paletas de colores, líneas de tiempo para animación, exportación de animaciones como *spritesheet*, creación de *tiles*, entre otras.

Figura 5.4: Logo de la herramienta aseprite

FL Studio: Una herramienta del tipo estación de trabajo de audio digital (DAW) que entrega el servicio para producir distintos temas musicales además de editar o crear sonidos para el juego por medio de conversión de audio digital y uso de grabaciones.

Figura 5.5: Logo de la herramienta FL Studio

Audacity: Adicionalmente, Audacity es una herramienta usada para grabación de audio y edición de sonidos. Especialmente es útil para complementar aplicaciones DAW, teniendo una vista más accesible y conteniendo más opciones de edición, especialmente al momento de grabar efectos de sonido.

Figura 5.6: Logo de la herramienta Audacity

6. Preproducción

Antes de comenzar con la producción del proyecto, es necesario tener una organización planteada en las tareas a realizar para mantener un orden de trabajo, tener metas fijas y continuas de trabajo para incentivar productividad. En el caso del videojuego, fue realizada en una carta Gantt que presenta una fecha de inicio y fin para las distintas tareas designadas durante la pre-planificación, también para facilitar el trabajo en términos de tiempo estas tareas se separaron en segmentos relacionados a las distintas etapas exclusivas del trabajo. Además, tomando en cuenta de que solo hay un integrante en el equipo de desarrollo en vez de asignar una persona a cada tarea se le asignó el área a la cual pertenece siendo estas las de programación, diseño, arte visual y sonoro como fueron mencionadas en el marco teórico del documento, esto para que el desarrollador pueda realizar tareas simultaneas con tal de que, idealmente, estas no se apliquen en la misma área de desarrollo. Se llegó a la conclusión de que el desarrollo del videojuego daría inicio oficialmente en noviembre del año 2020, con un receso después de la primera etapa fuera finalizada para consecuentemente retomar el trabajo durante el semestre universitario y seguir con este hasta que oficialmente concluya al término del mes de junio del año 2021.
Las etapas designadas en la preproducción fueron las siguientes:

- **Base del proyecto:** En esta etapa la producción se encarga de formar el esqueleto del proyecto, específicamente, se trabaja en los puntos principales que el género requiere siendo en este caso las mecánicas básicas encontradas en juegos de plataforma. En programación mecánicas de movimiento, colisiones y funcionalidades de cámara son los objetivos encontrados, mientras que arte y diseño toman un rol secundario enfocándose en el diseño visual del personaje principal y se definen cuáles serán las habilidades iniciales del jugador, las que

serían implementadas en la siguiente etapa. Esta etapa finaliza a mediados de noviembre 2020.

- **Mecánicas principales del género:** Luego del periodo de receso se toma lo trabajado en la etapa anterior y se profundiza para tener una base completa de los dos géneros en los cuales el videojuego se categoriza. En la parte del género de plataformas se programa un sistema de *inputs* para teclado y control, completamente personalizable para el desarrollador en el caso de hacer cambios necesarios, y para tener una condición de derrota se agrega un sistema de vidas al juego dándole el empujón inicial para trabajar en el género de acción, mientras que en el área artística se crea la primera animación de correr del personaje principal y su *sprite* de salto. Se inicia la implementación del segundo género este siendo el de acción, principalmente con la programación de un sistema de *hitbox* utilizado tanto por el jugador como por los enemigos. Para poder probar este sistema también se completa el desarrollo del primer enemigo encontrado en el juego siendo diseñado, programado y animado en esta etapa. Además, se desarrollan las habilidades iniciales diseñadas en la etapa anterior, como los ataques *slash*, proyectiles y sus combinaciones con las habilidades de movimiento ya disponibles, estas siendo necesarias para pasar a las mecánicas especificas del proyecto. Por otro lado, en el área de diseño se define las habilidades que podrán ser obtenidas por el jugador y se completa la idea de los otros enemigos que serán creados

- **Mecánicas específicas del proyecto:** Esta es considerada la etapa más grande e importante en el desarrollo del proyecto, ya que, como su nombre lo indica se trabajan las distintas tareas que son consideradas esenciales para la construcción del videojuego como las mencionadas características de evaluación y obtención de habilidades. Como los puntos más fuertes de esta etapa se encuentra la programación del gestor de salas que tiene distintas labores en el proyecto, los objetivos encontrados al final de cada etapa usados para definir el momento de evaluación del usuario y el mismo sistema de evaluación y obtención de habilidades que juntos mantienen el loop de juego. Por esta misma razón también en esta etapa se trabaja la programación y animación de las distintas habilidades obtenibles durante el juego y adicionalmente para complementar el sistema de evaluación se desarrollan nuevos enemigos y plataformas

con características especiales que serán usadas al diseñar los niveles.

- **Adiciones varias:** Como su nombre lo indica, en esta etapa se encuentran las tareas que son consideradas separadas de las mecánicas principales y específicas, y de cierta forma se pueden considerar refinamientos del proyecto centrados en crear un producto más completo en vez de enfatizar sus características principales. La programación de parámetros de cambio de teclas y completo soporte de control se encuentran para darle más opciones al usuario de como jugar, además de la programación de un escritor para que el usuario indique su nombre al inicio de la partida. Aquí se da un enfoque en la parte de audio del juego con la composición de su música y la edición e implementación de efectos de sonido para el juego. Finalmente se concretan y diseñan los distintos niveles que se encuentran disponibles en el juego y se crea el *tilemap* para su parte gráfica.

- **Base de datos y lanzamiento:** En la etapa final se encuentran dos objetivos esenciales, la programación e implementación de la base de datos enlazada con el juego y la publicación del juego en si en su plataforma. La primera requiere una cantidad de pruebas locales y externas antes de su implementación completa y consecuentemente es necesario preparar el lanzamiento del juego para que distintos usuarios lo prueben y se obtengan los datos necesarios para encontrar las conclusiones necesarias.

En en apéndice del documento se encuentra el enlace que dirige a la carta gantt que fue usada para la producción del videojuego, con detalles en las especificas tareas de cada etapa y su duración en los meses trabajados.

7. Implementación

En este capítulo se explican en detalle todos los elementos implementados en la versión final del producto, tomando en cuenta las características presentadas en el diseño del juego, las tareas designadas en el proceso de preproducción y los objetivos del proyecto en general.

7.1. Mecánicas

En esta sección se presentan las mecánicas encontradas en el juego desde las consideradas mecánicas principales, que son propias del género y definen las condiciones de victoria, derrota y los objetivos básicos, hasta las especificas del proyecto enfocadas en los objetivos exclusivos del proyecto; centradas en la evaluación del jugador, el desbloqueo de habilidades y su uso.

7.1.1. Objetivos y condiciones de victoria/derrota

El objetivo principal del videojuego es completar los 5 niveles presentados al jugador, esto se realiza desplazándose de derecha a izquierda por el nivel, evitando los ataques de enemigos, completando los desafíos encontrados en plataformas y finalmente realizando contacto con el portal encontrado al final de cada nivel (figura 7.1).

En paralelo existe el objetivo exclusivo del juego, que consiste en desbloquear las distintas habilidades ocultas en el juego. Esto se realiza al aumentar los 3 puntos de evaluación existentes y si se alcanza la cantidad requerida para ser desbloqueados este se encontrará disponible en el siguiente nivel.

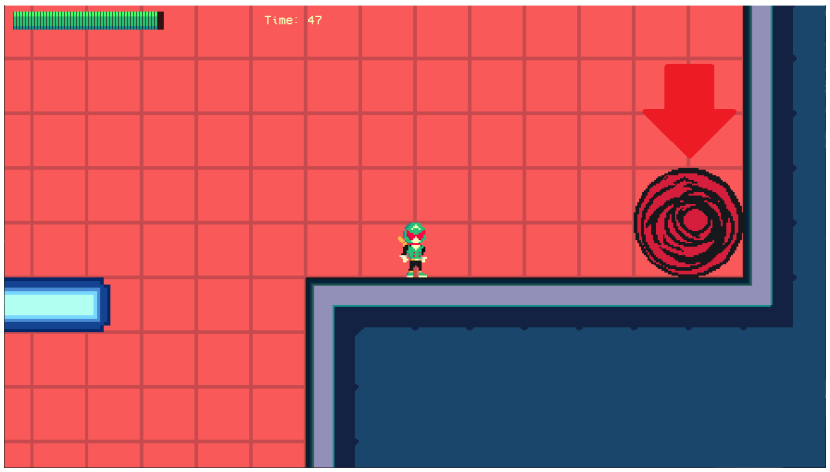

Figura 7.1: Portal al final de cada nivel que representa el objetivo principal

Finalmente, el videojuego contiene un simple sistema de vida como condición de derrota. El jugador cuenta con una vida máxima de 50 puntos, es decir, este puede recibir hasta 49 puntos de daño, y pasarse de este valor lo lleva automáticamente a la pantalla de derrota donde puede decidir si desea jugar otra partida inmediatamente, volver al menú principal o cerrar el juego.

Como fue mencionado, recibir daño resulta en perder vida y son los enemigos encontrados en el juego el obstáculo que impacta la condición de derrota. Existen dos formas en las cuales los enemigos pueden realizar daño, la primera es con sus ataques singulares (esta información se encuentra expandida en la sección de combate y la sección de enemigos) o haciendo contacto directo con ellos, lo cual realiza un daño fijo de 3 puntos y entrega al jugador un periodo de invulnerabilidad equivalente a dos segundos representado por su sprite tomando una tonalidad transparente (figura 7.2).

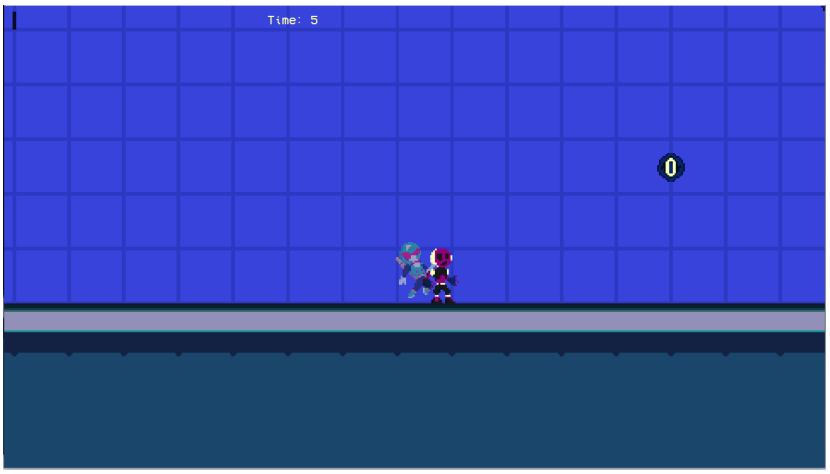

Figura 7.2: Personaje principal recibiendo daño por tocar a un enemigo

Figura 7.3: Al perder todos los puntos de vida, el fondo se colorea negro representando derrota

7.1.2. Habilidades

Por supuesto, para comenzar a evaluar al jugador es necesario implementar habilidades que se encuentren disponibles desde el inicio del juego. Para esto se tomaron en cuenta las mecánicas principales estándar iniciales (vida, plataformas y combate) para moldear el arsenal inicial del jugador.

Cuadro 7.1: Habilidades iniciales

Habilidad	**Descripción**	**Tipo**
Correr	Desplazamiento horizontal del personaje.	Movimiento
Salto	Movimiento vertical realizado al desplazarse del suelo.	Movimiento
Slash 1	Ataque realizado con espada cuando el jugador se encuentra estático en el suelo.	Ataque
Slash 2	Otro ataque con espada que solo puede ser realizado después de Slash 1.	Ataque
Slash 3	Ataque final con espada que solo puede ser realizado después de Slash 2.	Ataque
Run Slash	Ataque realizado con espada cuando el jugador se encuentra corriendo en el suelo	Ataque
Air Slash	Ataque realizado con espada cuando el jugador se encuentra en el aire.	Ataque
Shoot	Disparo de proyectil a distancia, veloz pero débil.	Proyectil

En movimiento, correr y salto son dos habilidades necesarias para desplazarse por plataformas y alcanzar el final de cada nivel, los ataques *slash* existen como las habilidades principales para dañar y eliminar enemigos, siendo *slash* 1, 2 y 3 parte de un *combo*, *run slash* para atacar enemigos sin interrumpir su movimiento y air slash para dañar enemigos aéreos o caer al suelo con un ataque, así mismo existe la opción de *shoot 1* como un proyectil de daño reducido a distancia. Para complementar las habilidades iniciales, entre la lista de posibles habilidades para el juego se tomaron las que fueron consideradas levemente más avanzadas para presentarlas como habilidades desbloqueables, estas son las que pueden ser obtenidas después de alcanzar una cierta cantidad de puntos en su respectivo tipo.

Cuadro 7.2: Habilidades desbloqueables

Habilidad	Descripción	Tipo
Crouch	El personaje se agacha para evitar ciertos ataques.	Movimiento
Double Jump	El personaje puede saltar una segunda vez en medio del aire.	Movimiento
Fast Fall	Si se encuentra en el aire, el personaje desciende rápidamente cancelando su momentum vertical.	Movimiento
Massive Slash	El jugador realiza un ataque lento pero grande que realiza mucho daño.	Ataque
Chargeshoot 1	Ataque de proyectil que debe ser cargado, haciendo más daño que shoot.	Proyectil
Chargeshoot 2	Igual que chargeshoot 1, pero requiere más tiempo de carga y realiza más daño	Proyectil

Cabe destacar que ninguna de las habilidades desbloqueables reemplaza las habilidades iniciales u otras habilidades desbloqueables, si no que buscan complementarse al entregar más opciones para que el jugador complete objetivos y evite derrotas.

7.1.3. Evaluación de habilidades

Al tener definidas las habilidades desbloqueables por el jugador se elaboró un sistema de evaluación para entregar estas habilidades. El sistema consiste en el uso de 3 funciones y una variable de evaluación por cada tipo de habilidad (movimiento, ataque y proyectil) que aumenta si dependiendo de su requerimiento. Al final de cada nivel los puntos son evaluados, y si el jugador a alcanzado los puntos requeridos para una o más habilidades, una de estas es elegida, los puntos de su tipo son reiniciados y la habilidad es desbloqueada al comienzo del siguiente nivel.

Movimiento: La variable de movimiento cambia en base a un temporizador encontrado en cada nivel, este comienza en 0 y avanza en segundos. Al completar un nivel el temporizador se detiene y se evalúa con la siguiente lógica: existe un valor predeterminado equivalente al “mejor tiempo aceptable”, si el temporizador finalizó antes o exactamente al llegar a este tiempo entonces su variable obtiene el máximo de

puntos entregables. Si este no es el caso sus puntos de evaluación van disminuyendo por cada segundo sobre el mejor tiempo hasta llegar a un mínimo predeterminado, donde se recibe la menor cantidad de puntos posible independiente de cuanto el temporizador aumente sobre este valor.

Ataque: Ataque se evalúa en el daño realizado mediante el uso de habilidades del tipo ataque, ya sea las habilidades iniciales como las desbloqueables.La cantidad de puntos aumentados es proporcional a la cantidad de daño realizado con estas habilidades al completar un nivel.

Proyectil: Proyectil se evalúa en el daño realizado mediante el uso de habilidades del tipo proyectil, ya sea las habilidades iniciales como las desbloqueables.La cantidad de puntos aumentados es proporcional a la cantidad de daño realizado con estas habilidades al completar un nivel.

En cuanto a las tres funciones utilizadas, la primera fue denominada habilidad, que funciona como plantilla genérica para la creación de habilidades y define sus puntos requeridos. La segunda función es la adición de habilidades, también creada de forma genérica y replicada para cada tipo de habilidad, y como su nombre lo indica, esta agrega los puntos de evaluación recibidos al total de cada una. Y la última función es el evaluador de habilidades, que tiene la función de revisar si los requerimientos de las habilidades han sido alcanzados al final de cada nivel y elige una de estas para ser desbloqueadas al inicio del siguiente.

7.1.4. Combate

El combate implementado en el juego consiste en dos estados que el jugador y los enemigos pueden encontrarse, realizando un ataque o recibiendo daño; ambos siendo importantes para el jugador ya que eliminar enemigos facilita la finalización de un nivel y recibir daño implica acercarse a la condición de derrota. El sistema usado para el combate utiliza dos términos conocidos en el mundo de los videojuegos (especialmente los competitivos) pero adaptado a el género de plataformas.

Hitbox

Hitbox son cajas virtuales invisibles para el usuario final creadas por un jugador, enemigo u objeto en el juego con la misión de detectar colisiones específicas que gatillan un ataque en el objetivo. Los *hitbox* contienen un dueño que sería su creador y están asignadas a un estado de ataque, el cual define sus características como tamaño, daño y duración. Además, los *hitbox* tienen un oponente el cual aplica el tipo de objeto en el juego que es considerado como objetivo, es decir, los enemigos para el jugador y el jugador para los enemigos.

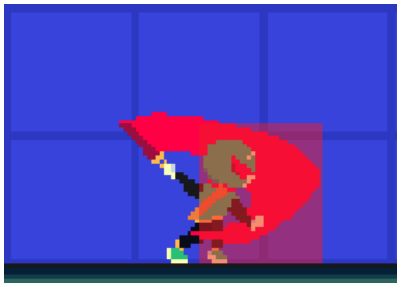

Figura 7.4: Ejemplo de hitbox de forma visible

Knockback

La otra cara del combate, *knockback* es un resultado de ser dañado donde el recibidor ve su movimiento interrumpido y es empujado a una dirección definida por el ataque. *Knockback* solamente ocurre en el estado de hit, por lo que no puede hacer otra acción hasta que este acabe.

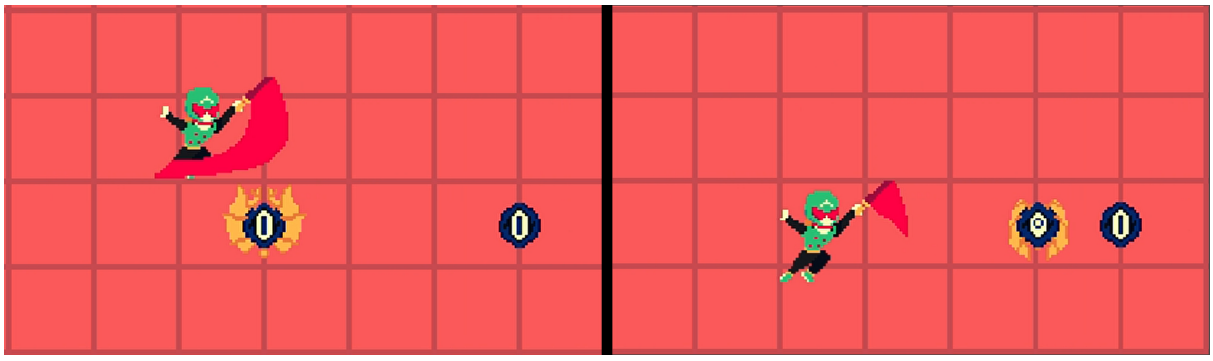

Figura 7.5: Ejemplo de knockback, se puede apreciar que el enemigo se desplaza al ser atacado

7.1.5. Enemigos

El videojuego cuenta con 3 enemigos, cada uno con su propia cantidad de vida, estado de ataque y características en cuanto a combate, pero todos fueron diseñados con el objetivo de dañar al personaje jugable. Antes de mencionar a los enemigos, se definieron dos tipos de enemigos en base a sus físicas.

- *Flying* son enemigos que se mueven libremente en el mapa en cualquier dirección e ignoran colisiones con objetos solidos en el nivel. Como su nombre lo indica, estos enemigos vuelan y se encuentran mayormente en el aire careciendo de gravedad, por lo que su knockback es más alto al ser dañados.

- *Grounded* se refiere a los enemigos encontrados en superficies que enfrentan al personaje jugable al encontrarlo en su campo de visión. Estos colisionan con los objetos solidos en el mapa ya sea de forma vertical u horizontal y se aferran a la gravedad en el juego, por lo que los hace más susceptibles a recibir *combos*.

Karv: Diseñado con el nombre clave "volador", este enemigo cuenta con un estado inicial, donde busca instancias del personaje jugable en un área circular y en el caso en que este colisione con el área pasa a su estado de ataque. Su estado de ataque consiste en continuamente obtener la posición del personaje jugable y acercarse a este para forzar contacto y dañarlo, ignorando otras colisiones de plataformas y enemigos. Este enemigo es de tipo *flying* y cuenta con 10 puntos de vida.

Deirr: Diseñado con el nombre clave "trooper", este enemigo se mueve de un lado a otro y tiene un rango de visión cónico en frente de él que busca al personaje jugable, que al ser encontrado los enemigos deirr disparan un proyectil con impulso horizontal a su dirección.
Este enemigo es de tipo *grounded* y cuenta con 25 puntos de vida.

Malge: Diseñado con el nombre clave "lancero", Malge es un enemigo alto que cuenta con una gran lanza como arma principal. En su estado inicial estos enemigos se mantienen estáticos y siempre se encuentran de cara al personaje jugable y al igual que los enemigos Deirr, Malges cuentan con un rango de visión cónico en busca del jugador, que al ser encontrado hace que estos enemigos se desplacen rápidamente a su dirección con el objetivo te atacarlo con la lanza como un caballero de justa.

Cabe destacar que la lanza contiene un hitbox continuo, por lo que el jugador puede ser dañando por ella si la toca en cualquier estado excepto hit.
Además, la lanza de los enemigos Malge cuentan una la característica única, una caja de colisión que destruye proyectiles, lo que evita que sea interrumpido por uno al atacar.
Este enemigo es de tipo *grounded* y cuenta con 30 puntos de vida.

7.1.6. Estados

En este proyecto los objetos más complejos realizan sus acciones en base a un sistema de estados basado en modelo de Máquinas de Estados Finitos (FSM). Las máquinas de estados finitos consisten en un modelo de computación basado en una máquina hipotética hecha de uno o más estados donde solo un estado individual se puede encontrar activo al mismo tiempo, haciendo que la máquina pase de un estado a otro para realizar sus distintas acciones[3].
En videojuegos, su aplicación se ve al tratar a un objeto como actor y entregarle distintos estados a los cuales puede acceder y condiciones para poder cambiar a otros estados. En este proyecto existen dos elementos que cuentan con este sistema, el personaje principal que cuenta con estados en base a los comandos entregados por el usuario y otros factores externos, y los enemigos que realizan acciones con una inteligencia artificial básica.

En el caso del personaje principal este presenta una variedad de estados, tomando en cuenta que estos son usados para realizar habilidades y responder a cambios en el ambiente del juego.

Estados base: Los "estados base" están definidos como los estados que responden a un cambio en el ambiente del juego, ya sean gatillados por el jugador o por factores externos. De estos estados los considerados principales son los llamados *simple* y *on-air* que son ingresados cuando el jugador se encuentra en una superficie o en el aire respectivamente y no se realiza ningún tipo de input por parte del jugador, estos son usados principalmente para trasladarse a estados de habilidades y no confundir las que usan el mismo *input* y restricción (por ejemplo, *slash* 1 y *air slash*).
Adicionalmente existen los estados *hit* y *dead* , *hit* es usado cuando el jugador es dañado por un ataque enemigo haciendo que su vida disminuya, sea empujado por

knockback, no pueda realizar acciones por un tiempo determinado, y en el caso en que su vida disminuya a 0, pasar al estado *dead* donde se pasa a implementar los elementos necesarios para acabar la partida.

Estados de habilidades: Como su nombre lo indica la mayoría de las habilidades implementadas, ya sean iniciales o desbloqueables, están asignadas a un estado. Específicamente, todas las habilidades tipo ataque, proyectil y la habilidad de movimiento *crouch* tienen su propio estado, mientras que las no mencionadas se encuentran adheridas a los estados simple u onair dependiendo de si deben ser accionadas en el aire o no. Tener estas habilidades en estados facilita al jugador moverse entre estos y realizar acciones específicas como combos en estados de ataque o ser interrumpidos por ataques evitando los errores que pueden ocurrir si existieran en un mismo estado.
En cuanto a los enemigos del videojuego, al realizar acciones independientes de los inputs directos del jugador, estos funcionan con una máquina de estados autónoma en base a interacciones con el personaje jugable. Los enemigos cuentan con 3 estados, uno donde buscan una instancia del jugador y no presentan una amenaza directa, uno de ataque que se inicia al encontrar la instancia y realizan un ataque dependiendo del tipo de enemigo (revisar la sección de enemigos para más información) y uno estado al recibir daño donde se aplican las mismas características que el estado *hit* del personaje jugable recibe al ser dañado.

7.2. Diseño de niveles

Para la versión publicada del juego se diseñaron cinco niveles distintivos que usan toda la variedad enemigos y plataformas diseñadas.
En cuanto a plataformas además de las colisiones sólidas que funcionan como superficie básica del juego se designaron plataformas *one-way*, que corresponden a plataformas donde su colisión solo afecta si el objeto a colisionar ingresa por encima de ella, por lo que el personaje jugable puede ingresar a ellas al saltar estando debajo. Estas plataformas se encuentran en los mapas en tres tipos, estáticas, con movimiento horizontal y con movimiento vertical.
Para crear estos niveles se definieron distintas reglas que todos deben seguir para mantener coherencia en diseño, entregar un nivel de dificultad similar y no variar

mucho en los resultados entregados.

- Todos los niveles deben tener una duración similar.
- Todos los niveles deben contener al menos una instancia de cada tipo de plataforma o enemigos.
- Todos los niveles deben ser posibles de completar solo con las habilidades iniciales.
- Todos los niveles deben contener una dificultad moderadamente alta.

7.3. Arte

Como su nombre lo indica, la sección de arte se centra en el desarrollo de los recursos artísticos visuales que fueron implementados en el videojuego. Se presenta la transformación de las ideas de estética explicada en el capítulo de diseño al resultado final con ejemplos para respaldar.

7.3.1. Personaje jugable

El primer trabajo de arte planteado y el más importante para el proyecto en su totalidad fue el relacionado al personaje principal. Al trabajar con *pixel art*, su diseño fue inicialmente engendrado en una cuadricula donde sus dimensiones no podían pasar el máximo de 32 x 32 y con una edición personalizada de la paleta de colores denominada "Journey" encontrada en la página web de recursos gratuitos Lospec[17] la cual fue utilizada en todos los sprites creados para el videojuego con la intención de mantener coherencia.

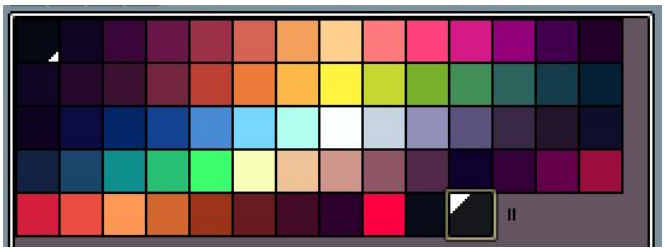

Figura 7.6: Paleta de colores utilizada para los sprites del videojuego

Al tener estos aspectos definidos se pasó al diseño del personaje principal con el desarrollo de su *sprite* inicial, la "pose *idle*" que muestra el cuerpo completo del personaje. Primero tomando los puntos encontrados en la temática, se encontró que las características a acentuar se centraban en su cabeza (su casco y visor), por lo que se tomó la decisión de proporcionar su cabeza de forma caricaturesca en comparación al resto del cuerpo y centrar los colores principales en esta área, mientras que otros aspectos como cinturón, botones en el uniforme y el uso de una espada como arma principal quedan para entregar más detalles y acentuar el uso de la paleta de colores en el sprite. En cuanto a colores, se utilizaron dos colores complementarios para que funcionaran como sus colores principales, estos siendo rojo y verde (*red torch* y *jungle green* específicamente) y adicionalmente complementado con tonos blancos, negros y amarillos para finalizar los detalles en su diseño.

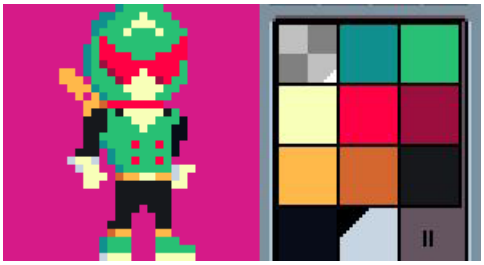

Figura 7.7: Pose idle del personaje principal junto con su paleta de colores

Al finalizar el diseño de la pose *idle*, los siguientes *sprites* usan lo usan como base para crear las distintas animaciones que el personaje jugable posee además de unos sprites adicionales, tomando en cuenta de que estos no se verán por su cuenta si no que en movimiento y que la nueva tecnología no presenta grandes limitantes en gráficas, especialmente en *pixel art*, algunas reglas en cuanto dimensiones y silueta del personaje son exceptuadas para darle prioridad a efectos de impacto y a la propia fluidez de la animación como será mencionado a continuación.

Figura 7.8: Animación de correr del personaje principal

Figura 7.9: Animación ataque slash 1 del personaje principal

Aquí se pueden apreciar dos láminas de sprites distintas que forman dos animaciones encontradas en la versión final del videojuego, la figura 7.8 muestra los 6 sprites que forman la animación de correr usada para el movimiento horizontal en superficies, y están diseñadas para ser usada en bucle con el ultimo sprite conectándose con el primero.
Por otro lado la figura 7.9 muestra un ejemplo de cómo los ataques slash son animados, en específico estos sprites forman la habilidad *Slash* 1. Como se puede apreciar el personaje principal solo realiza dos poses con la segunda repitiéndose en la mayoría de los sprites, ya que al ser un movimiento veloz los cuadros deberían aparecer entre el movimiento son tan rápidos que pasarían a ser irreconocibles a simple vista, por lo que no son necesarios de dibujar. Pero el foco de esta animación se encuentra en el mismo ataque donde el movimiento de la espada es representado por una estela de color rojo que deja al pasar, que también resalta el tamaño del ataque y su duración. Esta misma filosofía de animación es utilizada para todos los ataques del tipo *slash*.

7.3.2. Enemigos

Los otros personajes que requieren la máyor cantidad de trabajo al ser diseñados y animados son los enemigos. Aquí se presentará como pasaron de sus aspectos claves y mecánicas planteadas a un diseño coherente a estas características.

Karv

Los enemigos Karv son simples criaturas voladoras, por lo que al diseñarlos se tomó énfasis en sus alas y el núcleo que las conecta. Como la regla de los enemigos es tener tres tipos de estados (estático, ataque y dañado) su número de *sprites* es limitado en comparación a los del personaje jugable, con todos los enemigos poseyendo solo una animación y una variedad de *sprites* que reflejen los distintos estados. En el caso de Karvs, su diseño principalmente utiliza los colores azul y amarillo para

el cuerpo y las alas respectivamente con toques de blanco y negro para los detalles, además, con la intención de entregarle más expresión al personaje y facilitar de entender el estado en el que se encuentra, se le entregó un ojo en medio de su núcleo como punto de foco al momento de diseñar sus distintos *sprites* con tal de que sean fáciles de diferenciar. La animación de los enemigos karv es usada para trasladarse entre su estado estático al de ataque, demostrando que ha encontrado la instancia del personaje principal.

Figura 7.10: Sprites y paleta de colores enemigo Karv

Deirr

Al igual que el personaje jugable, los enemigos Deirr también toman inspiración del género "Sentai", con la idea de ser soldados rasos que no poseen un gran peligro para el héroe. Diseñados como enemigos humanoides delgados, toman un estilo robótico en sus animaciones y expresiones de rostro, que tiene un aspecto digital. Al igual que los enemigos Karv, se usan colores blancos y negros para destacar detalles en su diseño y apoyar a los colores purpuras y azules que tienen la función de ser los principales. La animación elegida para los enemigos Deirr es un *loop* de caminar usado en los momentos donde se mueven de un lado a otro buscando la instancia del personaje jugable.

Figura 7.11: Sprites y paleta de colores enemigo Deirr

Malge

Finalmente se encuentran los enemigos Malge, que se destacan en su diseño por sus tonos oscuros, gran altura y una lanza aún más grande. Al tener su diseño basado en caballeros lanceros su cuerpo se encuentra protegido por una armadura y un casco que tapa su rostro, solo dejando un pequeño punto de luz que simula un ojo, en contraste, su arma presenta detalles luminosos en la punta que solo se apagan al ser atacados, demostrándole al usuario que puede recibir daño por parte de la lanza siempre y cuando se encuentre encendida. A pesar de su tamaño estos enemigos son veloces ya que se propulsan por medio de sus talones, lo cual les ayuda a atacar al personaje jugable cuando se encuentra su instancia (siendo el fuego de la propulsión su animación de ataque). Como fue mencionado los enemigos Malge usan tonos claros y oscuros para resaltar sus rasgos más importantes, con colores rojizos y morados en su armadura demostrando su pose rígida mientras que su ojo y punta de lanza mantiene tonos azules claros para demostrar vida y poder.

Figura 7.12: Sprites y paleta de colores enemigo Malge

7.3.3. Tileset

Se crearon *tilesets* relativamente simples para el juego integrado en los niveles disponibles, todos con una dimensión de 32x32 por *tile*. El primero es usado para los bloques solidos de colisión siendo el más grande con 47 *tiles* al tener que contar las distintas esquinas que pueden ser encontradas al momento de diseñar los niveles del juego mientras que diseño una *tileset* más diminuto de 16 *tiles* que corresponde a las plataformas *one-way* y así sean diferenciadas del *tileset* sólido.
Adicionalmente, se crearon 2 *tiles* pequeños para diferenciar las plataformas móviles de los otros objetos de colisión.

7.3.4. Logo

El último *asset* gráfico destacable encontrado es el logo del juego, que se puede apreciar en la pantalla principal del juego. También es utilizado como material visual en la página de publicación del videojuego.

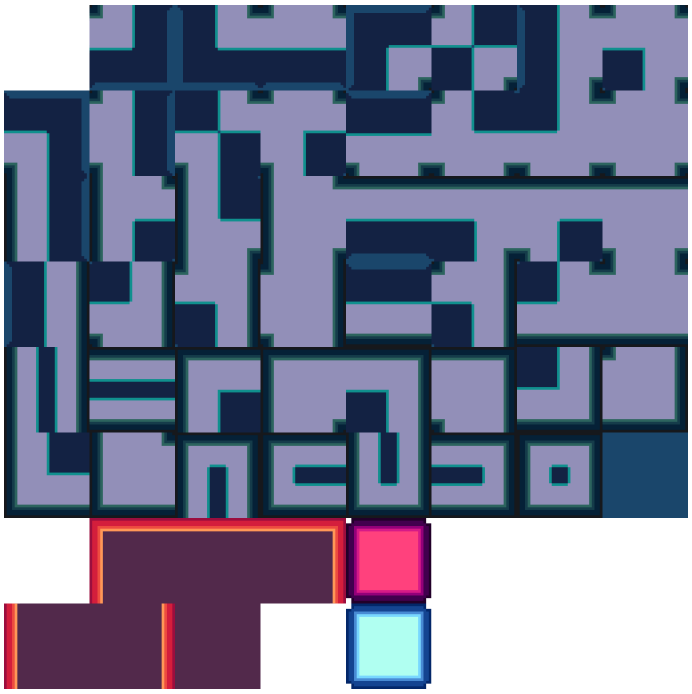

Figura 7.13: Tilesets utilizados en el videojuego

Figura 7.14: Logo del juego presentado como ícono web

7.4. Sonido

A continuación, se describirán los recursos sonoros implementados en el juego los cuales están separados en dos subsecciones, la música de fondo que tiene la función

de establecer un estado de ánimo en el jugador y los efectos de sonido que enfatizan las acciones realizadas en el juego y entregan realimentación de sus resultados.

7.4.1. Música

El videojuego cuenta con dos temas, uno usado para el menú principal y el otro en partidas. El tema del menú principal comienza con un *fade-in* que se transforma en un bucle de percusión junto con la línea de bajo principal y con acordes que llevan a la pista que lidera la canción. Al finalizar la pista principal de la canción hay una pausa que resulta en un cambio de percusión, acodes y línea de bajo la cual se repite un par de veces hasta volver a los elementos originales de la canción. En la segunda vuelta de la sección inicial en medio de la pista principal se produce un cambio, donde luego de repetir un par de notas la pista principal suena junto a la sección alterna que extiende la pista principal y que lleva al final de la canción con la sección inicial.
El tema presentado en partidas utiliza la misma línea de bajo, pero con un BPM (*Beats Per Minute*) más alto para presentar la acción del juego, es por esto que los acordes son más simples y suaves y la percusión presentada es más agresiva y toma el rol principal de la canción, este punto se solidifica cuando la canción integra un bucle de marcha. Adicionalmente esta canción cuenta con un arpegio que sigue el tono de los acordes, dándole una identidad distinta a la del menú principal y ayuda a completar el bucle del tema.

Figura 7.15: Captura de FLStudio mostrando la produccion del tema usado en gameplay

7.4.2. Efectos de sonido

En cuanto a los efectos de sonidos, el resultado esperado es contener sonidos que tengan un efecto robótico y representen correctamente su función.

Cuadro 7.3: Lista efectos de sonido

Nombre	Descripción
MenuSelect	Efecto accionado al seleccionar en el menú.
PlayerJump1	Sonido al presionar salto estando en una superficie.
PlayerJump2	Sonido al realizar un segundo salto en el aire.
PlayerDmg	El personaje jugable recibe daño.
PlayerSlash	Sonido al realizar cualquiera de los ataques slash excepto massive slash.
PlayerShoot1	Lanzamiento del proyectil relacionado a la habilidad shoot 1.
PlayerShoot2	Lanzamiento del proyectil relacionado a la habilidad shoot 2.
PlayerShoot3	Lanzamiento del proyectil relacionado a la habilidad shoot 3.
PlayerMassive	Sonido relacionado al realizar la habilidad massive slash.
PlayerHit	Sonido que ocurre cuando el jugador exitosamente daña a un enemigo.
DeirrShoot	Sonido utilizado para el ataque de los enemigos deirr, que consiste en el disparo de un proyectil.
MalgeAttack	Sonido utilizado para el ataque de los enemigos malge, que consiste en un impulso.

7.5. Recopilación de datos

Esta sección abarca la implementación de la minería de datos y las preguntas encontradas en la encuesta, que fueron el foco principal durante la fase de post-producción del proyecto. Se cuenta con la minería de datos obteniendo información directa de las partidas jugadas y la encuesta que entrega opiniones de *testers* en cuanto aspectos más subjetivos del videojuego.

7.5.1. Minería de datos

Antes de definir el método para la extracción de los datos del usuario, naturalmente hay que dejar concretos los datos que serán extraídos. Para esto se planteó que variables son las que permiten diferenciar cada partida y la habilidad del jugador requerida para completar los niveles, en base a esto se concluyó en una lista con las variables enviadas por el juego al finalizar un nivel, o en un intento fallido.

Cuadro 7.4: Lista de variables extraídas en cada partida

Nombre	**Descripción**
Level	Nombre del nivel que se ha completado o se haya perdido.
Timer	Tiempo en segundos que el usuario demoró para completar el nivel o momento en el que perdió.
HP	Cantidad de vida restante al finalizar el nivel, en el caso de ser menor a 1 significa que el jugador perdió.
Slash	Cantidad de puntos de evaluación en Slash al momento de mandar los datos.
Projectile	Cantidad de puntos de evaluación en Projectile al momento de mandar los datos.
Movement	Cantidad de puntos de evaluación en Movement al momento de mandar los datos.
SkillList	Lista con las habilidades desbloqueables y su estado al momento de enviar los datos.

Como se puede apreciar en la tabla 7.4 el último elemento en la lista es otra lista, esta con los estados de las habilidades desbloqueables. En cada una de las 6 habilidades desbloqueables se extrae la siguiente información.

Cuadro 7.5: Lista de variables extraídas encontradas en SkillList

Nombre	**Descripción**
SkillName	Nombre de la habilidad en cuestión.
SkillType	Valor que muestra si la habilidad corresponde al tipo Movimiento, Ataque o Proyectil.
SkillAvailable	Valor binario que explica si la habilidad a sido desbloqueada.

Al tener todos los datos necesarios definidos, estos se guardaron en un mapa de estructura de datos, conocido en Game Maker Language como “ds map”. “Ds maps” son mapas que guardan valores junto con una llave identificadora, usando la llave para encontrar el valor ya que estos no se encuentran ordenados[21].

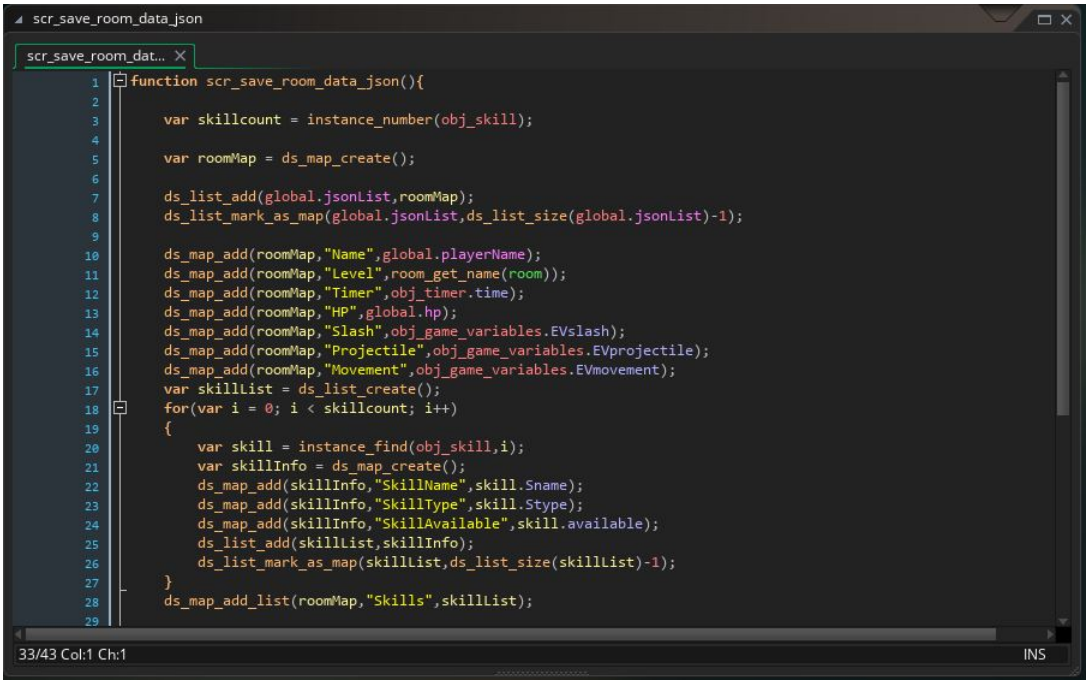

Figura 7.16: Estructura final de ds map en GameMaker Studio 2

La otra ventaja de trabajar con ds map es que estas pueden ser transformadas a JSON con la función propia de GML llamada `json_encode`, lo que entrega todo el contenido del mapa en un string con formato JSON y así trabajar el análisis de partidas en aplicaciones externas. Este string JSON es enviado si el jugador completa un nivel o pierde toda su vida.
La forma de conseguir los datos y almacenarlos en una base de datos es por medio de HTTP request, específicamente la función POST que translada la información de una aplicación (en este caso el juego desarrollado en GameMaker Studio 2) a una dirección externa.

Pruebas con localhost

Al tener completa la conexión por parte de gamemaker se comenzó a realizar pruebas para almacenar los datos enviados con una base de datos local creada en MySQL Workbench y usando el software gratuito XAMPP para gestionar la conexión entre la base de datos del proyecto y la base de datos del dispositivo local.
El último paso consiste en programar una aplicación que reconozca los envíos POST y los entregue a la base de datos local. Para esto se usaron dos herramientas, el entorno de desarrollo integrado IntelliJ y el asistente de aplicaciones *spring* denominado "Spring Boot", que permite fácilmente crear repositorios para los datos entregados en JSON por medio de las distintas peticiones a localhost.

```
125     @PostMapping(value="/test", produces = MediaType.TEXT_PLAIN_VALUE)
126 @   public String post(@RequestParam String content)
127     {
128         Gson gson = new Gson();
129         System.out.println("Content: " + content);
130
131         Contenedor contenedor = gson.fromJson(content.trim(), Contenedor.class);
132
133         Set<Escena> escenas = contenedor.getLevels();
134
135         for(Escena escena : escenas)
136         {
137             Set<Habilidad> habilidades = escena.getSkills();
138             if(habilidades != null)
139             {
140                 for (Habilidad habilidad : habilidades)
141                 {
142                     if(habilidad != null)
143                     {
144                         repoHabilidad.save(habilidad);
145                     }
146                 }
147             }
148             repoEscena.save(escena);
149         }
150         repoContenedor.save(contenedor);
151         return "";
152     }
```

Figura 7.17: Función POST en IntelliJ con uso de controladores Spring Boot

Conexión con base de datos externa

Para ya trabajar con usuarios reales, se obtuvo una base de datos web entregada por el profesor guía por medio de una cuenta de administrador que permite revisar y manejar contenedores de datos. Para esto fue necesario crear una aplicación similar a la usada para la conexión de localhost, pero con propiedades de conexión y seguridad distintas para evitar errores con los navegadores. Para esto es implementada una conexión cors (*cross-origin resource sharing*) en la aplicación, esta es una conexión cruzada al momento de encontrar una petición HTTP entrega acceso por ambas partes lo que permite que ocurra un intercambio de información entre ambos dominios[1].

Al tener la aplicación lista el siguiente pase consiste en hacer un *deploy* de un archivo WAR (*Web Application Archive*, el cual es un aplicación web que se encuentra empaquetada para ser subida a un servidor[2]) con la dirección URL de la base de datos, los datos verificadores y subirlo al servidor.

[1]Control de acceso HTTP (CORS) https://developer.mozilla.org/es/docs/Web/HTTP/CORS

[2]Archivos WAR (Web Archive) https://www.ibm.com/docs/es/rsas/7.5.0?topic=projects-web-archive-war-files

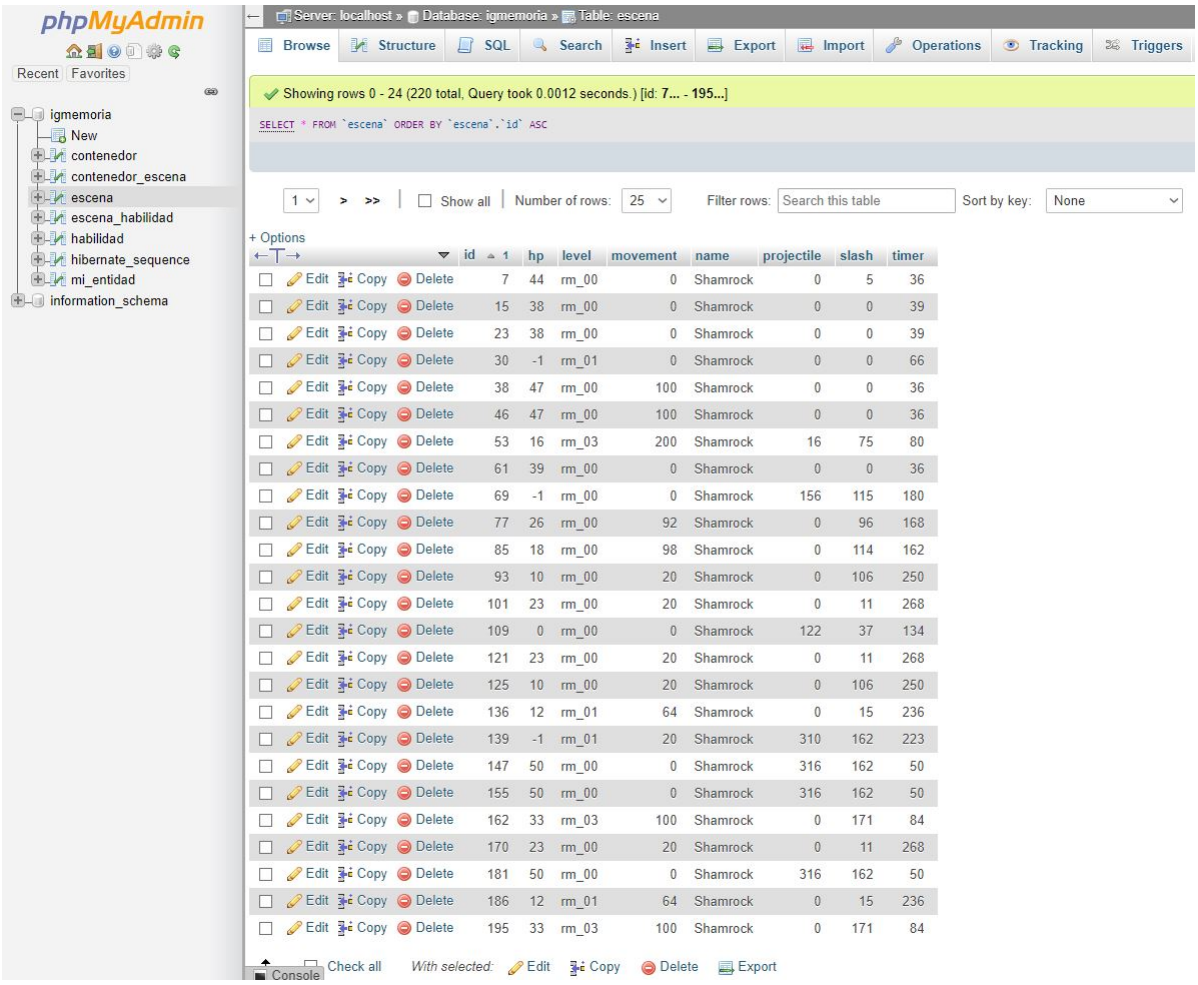

Figura 7.18: Captura de la base de datos funcionando en el servidor

7.5.2. Encuesta

Para complementar la información obtenida directamente de la minería de datos se entregó una encuesta opcional en inglés enlazada en la página web de descarga del juego donde usuarios pueden responder preguntas variadas en cuanto a su experiencia de juego y otros temas valiosos relacionados a la realimentación.
La intención de esta encuesta es comparar las respuestas entregadas con la información obtenida en la minería de datos y confirmar que estas alinean, y adicionalmente preguntar a los usuarios acerca de características del videojuego que no pueden ser obtenidas en datos, como opiniones relacionadas a las mecánicas y los apartados visuales y sonoros.

La encuesta fue lanzada junto al videojuego por medio de la herramienta "Google Forms", una herramienta web gratuita que entrega una forma simple y conveniente de crear encuestas permitiendo realizar preguntas de selección múltiple, valoración numérica, respuestas cortas, etc. Todo con acceso inmediato a las respuestas ingresadas ya sea de forma web o exportando una hoja de cálculo[3].

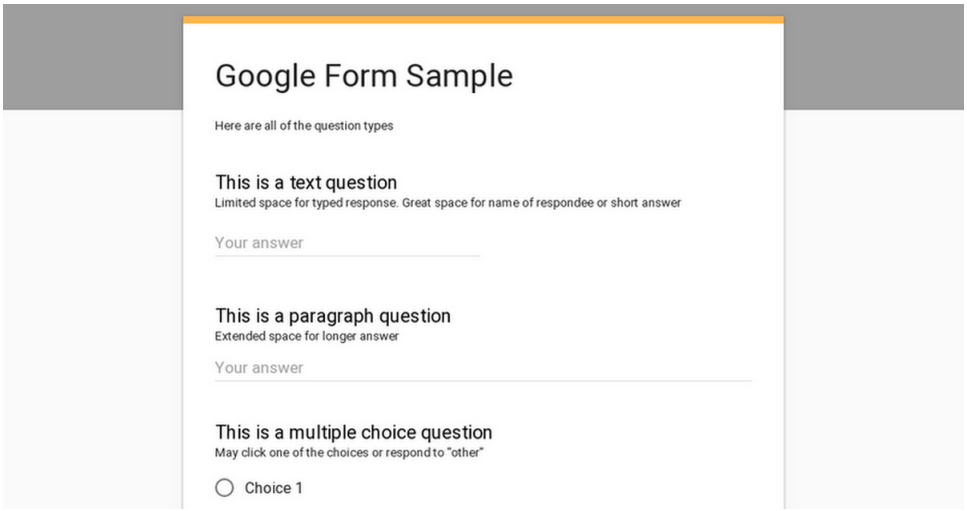

Figura 7.19: Ejemplo de la herramienta web Google Forms

La encuesta consiste en las siguientes preguntas:

1. **Un nombre de reconocimiento** con la intención de separar las preguntas obtenidas.
2. **Preguntas con respuesta si, no o no aplica** para conocer las mecánicas mantuvieron el *Game Engagement* del usuario.
   - ¿Volvió a jugar después de completar todos los niveles?
   - ¿Volvió a jugar después de morir?
3. **La cantidad de habilidades que desbloqueó en su primera partida de 0 a 6** para saber la destreza del jugador en su primera partida.
4. **Preguntas con respuesta si o no** para conocer hasta qué punto se jugó en cuanto a mecánicas.

[3]Página oficial Google Forms https://www.google.com/intl/es/forms/about/

- ¿Jugó hasta obtener cada habilidad?
- ¿Jugó hasta completar una partida sin morir?

5. **Evaluación de la experiencia de usuario en una escala del 1 al 10** evaluando el videojuego en base a su opinión personal.
    - ¿Qué tan divertido encontró el videojuego? (10 siendo lo más divertido)
    - ¿Qué tan difícil encontró el videojuego? (10 siendo lo máximo en dificultad)
6. **Preferencia personal al momento de lidiar con enemigos** para conocer que estrategia ofensiva le pareció más efectiva o divertida.
    - Disparar proyectiles.
    - Atacar con la espada
    - Evitar conflicto
    - Disparar y usar la espada
7. **Selección múltiple con opciones: Horrible, Malo, Ok, Bueno, Excelente** para evaluar de forma general las características principales del videojuego, la cual no puede ser extraída directamente de este.
    - ¿Cómo le pareció el movimiento en el juego?
    - ¿Cómo le pareció el combate del juego?
    - ¿Qué le pareció el diseño de los niveles?
    - ¿Cómo encontró la progresión de obtención de habilidades?
    - ¿Qué le pareció el pixel art del videojuego?
    - ¿Qué le parecieron los efectos de sonido del videojuego?
    - ¿Qué le pareció la música del videojuego?
8. **Interés por una versión más completa del juego: si, no o tal vez** para valorar el posible lanzamiento de una versión futura del proyecto.

9. **¿Qué adición encuentra que sea la más importante en el corto plazo para el desarrollo?** entregando opciones de características y mecánicas que no pasaron a la fase de desarrollo, para saber cuál debería ser implementada en una nueva iteración.
    - Alguna forma de recuperar vida.
    - Gráficos y efectos de sonido mejorados.
    - Puntos de guardado.
    - Modo historia / campaña.
    - Batallas con jefes.
    - Más contenido en general (habilidades, niveles y enemigos).
    - Balance en la dificultad.
    - Otro (con un campo para completar).
10. **Un campo para comentarios adicionales opcional** por si el encuestado desea realizar una realimentación que no puede ser entregada con las preguntas previas.

8. Análisis de datos y resultados

8.1. Base de datos

Finalmente, la base de datos llegó a recolectar un total de 158 instancias de partidas jugadas distintas con la información entregada por archivos JSON, esta vez recibido desde el mismo servidor como un archivo csv (valores separados por comas) el cual puede ser analizado en KDD. Para realizar el análisis se hiso uso de la aplicación gratuita WEKA, que contiene algoritmos de clasificación además de presentar opciones de limpieza y transformación de datos para aumentar el porcentaje de precisión de resultados[20].

8.1.1. Evaluación de usuarios

Al finalizar la fase de extracción de datos se inició un proceso de evaluación de partidas donde se tomó un resultado de usuarios reales encontrados fuera de la base de datos para definir la calidad de una partida, que luego sería ingresado a WEKA para identificar el nivel de precisión. Las partidas fueron evaluadas con la destreza esperada en un usuario común, quedando tres clasificaciones:

Bueno: El usuario mostró la destreza esperada al momento de completar el nivel, usando correctamente las mecánicas bases y especificas del juego.

Intermedio: El usuario entendió algunas de las mecánicas del videojuego al finalizar el nivel, pero mostró dificultad la mayoría del tiempo.

Malo: El jugador no pudo adaptarse a las mecánicas y no contó con la destreza necesaria para entregar un resultado satisfactorio.

Para determinar a qué clasificación se encontraba cada partida encontrada en la base de datos se llevaron los resultados de las pruebas y se identificaron los valores de cada variable necesaria para cada clasificación, los resultados fueron los presentados en el cuadro 8.1.

Cuadro 8.1: Clasificaciones de usuarios por partida

	B	**I**	**M**
Tiempo en segundos	0 - 160	161 - 299	300+
Puntos de vida nivel 1	50 - 40	39 - 26	25 - 0
Puntos de vida nivel 2	50 - 35	34 - 21	20 - 0
Puntos de vida nivel 3	50 - 30	29 - 16	15 - 0
Puntos de vida nivel 4	50 - 25	24 - 11	10 - 0
Puntos de vida nivel 5	5 - 20	19 - 6	5 - 0

Los datos usados para la evaluación fueron los de vida y tiempo, ya que ambos tienen un punto de inicio y fin en cada partida, mientras que la obtención de habilidades y los puntos de evaluación fueron analizados de forma externa a la evaluación.

Finalmente, para llevar a cabo la clasificación de partidas se decidió usar el algoritmo clasificatorio de Naive Bayes. Este algoritmo basado en el teorema del mismo nombre considera de forma independiente el valor de cada variable al momento de evaluar un elemento, por lo que no existen relaciones entre las distintas características presentadas en el elemento evaluado (en el caso de este proyecto en específico, los datos de vida no afectan a la evaluación de tiempo y viceversa) haciendo que cada una de las variables tengan el mismo peso al momento de seleccionar la clasificación[11].
La información relevante al momento de implementar la clasificación Naive Bayes viene de evidencias previas a la evaluación presentadas en la tabla 8.1, comparando su resultado por medio de probabilidad condicional.

8.1.2. Limpieza de datos

Con la misión de eliminar cualquier contenido de información que no aporte a la clasificación de partidas se comenzó la limpieza de datos. La primera parte consistió en reconocer cualquier dato que pueda ser considerado como clasificador, ya que este puede interferir con el clasificador de partidas. Inmediatamente se identificó la gran cantidad de datos relacionados a la lista de habilidades la cual fue creada como una lista dentro de la principal, ahí se concluyó que los únicos datos relevantes para las habilidades son sus nombres y disponibilidad, por lo que se eliminó la variable tipo. Aun así, la variable de nombre seguía siendo un problema ya que aparece como otro clasificador que además se aplica seis veces por cada partida de forma independiente, por esto se tuvo que editar la tabla de datos y sus atributos haciendo un atributo nuevo por cada habilidad y a cada uno se le asignó un valor booleano que representa si la habilidad fue desbloqueada.

Cuadro 8.2: Lista final de variables analizadas

Nombre	**Valor**
EscenaSlash	int
EscenaProjectile	int
EscenaMovement	int
EscenaTimer	int
Room	nom
EscenaHP	int
Crouch Available	int
Double Jump Available	int
Fast Fall Available	int
Massive Slash Available	int
ChargeShoot 1 Available	int
ChargeShoot 2 Available	int
Clasificación	nom

Posteriormente se decidió eliminar las partidas en las que el jugador haya sido derrotado, ya que al usar una partida de derrota hace que el valor de tiempo se haga irrelevante, ya que la partida puede finalizar muy prontamente al morir, lo que es considerado un factor positivo al completar el nivel. Esto llevó a reducir la cantidad de partidas evaluadas a 123.

8.1.3. Resultados

Al pasar la clasificación a WEKA este entregó un nivel de precisión del 74.7 %, el cual es considerado aceptable para el proyecto, específicamente 92 instancias fueron evaluadas correctamente y 31 de forma incorrecta. Con esta información se llegó a los siguientes resultados.

- Partidas Buenas : 64 (52 %)
- Partidas Intermedias: 37 (30 %)
- Partidas Malas : 22 (18 %)

Como se puede apreciar, más de un 50 % de las partidas pasaron la evaluación de habilidad esperada, un resultado positivo para la investigación y las mecánicas implementadas, los mismos resultados exitosos también se ver reflejados al ser comparado con la encuesta opcional que será presentada en la siguiente sección. Aquí se presenta información adicional que fue considerada importante encontrada en los resultados entregados por el análisis:

- La mayor cantidad de partidas clasificadas buenas fueron encontradas en los niveles 1 y 5, estos siendo los primeros y los últimos niveles presentados al jugador respectivamente.
- En ninguna partida un usuario desbloqueo la habilidad del tipo movimiento *fast fall*, que requiere un gran número de partidas jugadas para ser desbloqueada.
- Las habilidades *crouch* y *chargeshoot* 1 fueron las únicas que recibieron un porcentaje de desbloqueo mayor al 50 %.
- Todas las partidas consideradas malas terminaron con una puntuación de vida menor a 30.
- Hay una directa relación entre la cantidad de puntos de evaluación y la cantidad de habilidades de cada tipo, con un mayor promedio de puntos de evaluación entre menor cantidad de habilidades de este tipo (Por ejemplo, la puntuación máxima de movimiento alcanzó los 400 puntos con tres habilidades desbloqueables de este tipo, mientras que el tipo *slash* que solo tiene una habilidad desbloqueable obtuvo una puntuación máxima de 800).

8.2. Encuesta

La encuesta finalizó con un total de 12 usuarios únicos entregando su experiencia propia con el videojuego. Las observaciones muestran una interesante variedad en las respuestas de cada usuario, lo cual era lo esperado sabiendo que el proyecto contaba con la intención de que cada partida fuera distinta, dependiendo a como el usuario decida completar los niveles y como se adapte a las habilidades iniciales y desbloqueables. Esto es presentado en la primera pregunta que se encontró la misma cantidad de respuestas en sus 3 opciones.

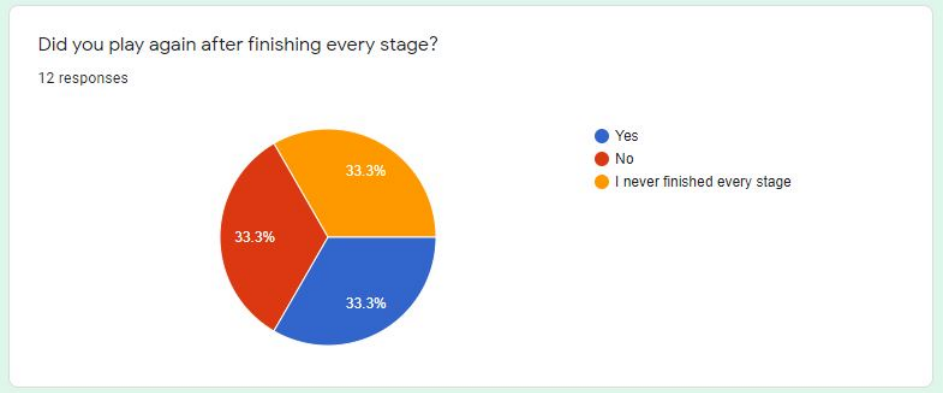

Figura 8.1: Resultados de la pregunta ¿Volvió a jugar después de completar todos los niveles?

Al contrario, la única respuesta que tuvo un resultado unánime fue la pregunta acerca del desbloqueo de habilidades, donde los usuarios respondieron que no desbloquearon todas las habilidades del juego. Esto concuerda con la información extraída en la base de datos, al saber que la habilidad *fast fall* nunca fue desbloqueada.

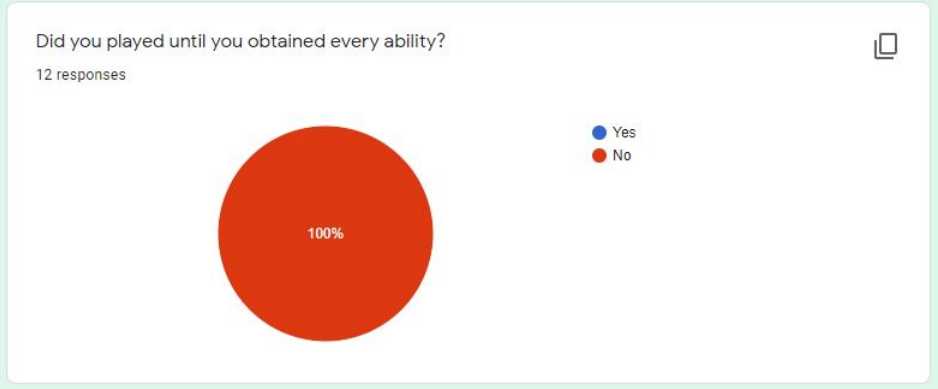

Figura 8.2: Resultados de la pregunta ¿Jugó hasta obtener cada habilidad?

En cuanto a entretenimiento y dificultad los resultados fueron positivos, con entretenimiento consiguiendo un sólido promedio de siete puntos del máximo de diez y teniendo una nota 3 como la más baja, mientras que la dificultad del juego fue evaluada de forma alta como era esperado en el diseño de los niveles, ya que ningún usuario la evaluó con una nota menor a 5, esto demuestra que la mayoría de los encuestados evaluaron positivamente la dificultad y el valor de entretención del videojuego en su opinión subjetiva.

Figura 8.3: Resultados de la evaluación de entretenimiento y dificultad

Las preguntas del tipo 7 (selección múltiple con opciones de horrible a excelente) también mostraron un resultado positivo al no recibir en ningún momento la respuesta más negativa, especialmente al ser comparado con el abundante número de respuestas buenas o excelentes.
El aspecto más criticado del juego fue sin lugar a dudas el diseño de niveles lo cual también puede ser reflejado en los comentarios adicionales que serán mencionados más adelante, y por otro lado los puntos mejores valorados fueron aspectos del videojuego que no pueden ser evaluados en la minería de datos, estos siendo apartado artístico del juego con el *pixel art* y la música del juego teniendo los mejores promedios.

Los comentarios adicionales también reflejan los resultados de la encuesta, mencionando la dificultad del juego y sus criticas con el diseño de los niveles, dos áreas del juego en las cuáles se puede reconocer una relación en su respuesta relativamente negativa, en comparación a las otras características del videojuego.
Además, cabe destacar que usuarios comentaron acerca dos puntos importantes que

no podían ser evaluados en la encuesta ni pudieron encontrarse como puntos a analizar en la base de datos, estos fueron:

- Problemas con la cámara, específicamente siendo muy limitada en comparación a las opciones de movimiento del juego lo que resultó en jugadores recibiendo daño de enemigos sin la posibilidad de ser evitado.
- La velocidad de respuesta de algunos enemigos es muy alta haciendo muy difícil de esquivar sus ataques, especialmente si estos son encontrados de forma repentina.

Finalmente, todos los usuarios encuestados mostraron un interés en probar una versión futura del juego o específicamente una "versión completa" que integre los cambios sugeridos, demostrando que los conceptos del proyecto y la mayoría de sus implementaciones fueron exitosos.

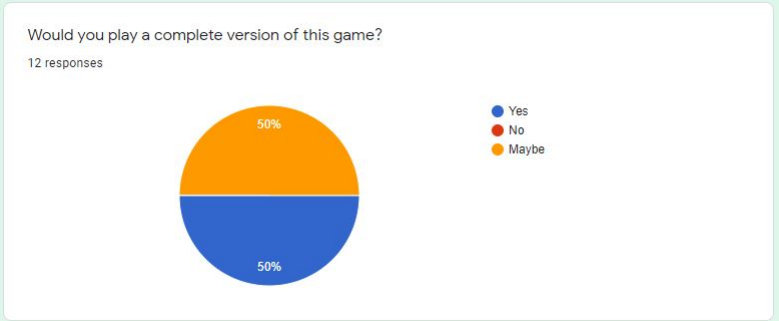

Figura 8.4: Resultados acerca del interés por una versión actualizada del videojuego.

9. Conclusiones

9.1. Desarrollo del videojuego

En cuanto al desarrollo del juego publicado, se estimó que las áreas aprendidas durante el transcurso de la carrera universitaria de diseño y programación fueron implementadas de forma exitosa en cuanto a los objetivos propuestos en especial la creación de una mecánica creativa que implementa los conceptos de dificultad y recompensa, por otro lado, las áreas que no forman parte de la formación académica (arte visual y sonido) también tuvieron un resultado efectivo, lo cual fue especialmente mencionado por los usuarios. Así, se puede concluir que las tareas relacionadas a su diseño, desarrollo y validación se realizaron efectivamente en cuanto a la duración del trabajo de memoria de título, y que el modelo de desarrollo junto con la planificación de los tiempos permitió la creación del videojuego planteado, con sus niveles, enemigos, habilidades y mecánicas correspondientes.

Las mecánicas principales relacionadas a la evaluación y obtención de habilidades pudieron ser implementadas de la forma que fueron visionadas en la fase de diseño, teniendo un número respetable de habilidades para desbloquear y de niveles donde estas pudieran ser usadas. Un gran factor para que estas funcionaran fue el motor de juego Game Maker Studio 2 que, gracias a su documentación, programación orientada a objetos y sistema de salas se transformó en el motor ideal para un proyecto de las características definidas en la tabla 5.1.

El uso de *pixel art* en el videojuego aseguró que este pudiera ser finalizado en los tiempos requeridos por la memoria, con el uso herramienta Aseprite facilitando su creación, especialmente al momento de crear las animaciones de los personajes, estos factores exitosos fueron reflejados en los comentarios positivos de la encuesta.

De la misma forma, el sonido del videojuego, que inicialmente no fue planificado

como uno de los factores principales para el desarrollo de las ideas del videojuego también fue recibido de forma positiva, con la creación de 12 sonidos únicos usados en el personaje principal, los enemigos y el menú del juego (cuadro 7.3) y la composición de 2 temas para el juego, que terminaron siendo lo necesario para el videojuego.

Una decisión importante en el desarrollo fue dejar el proyecto con mecánicas pulibles y permitiendo una expansión en su contenido, algo que fue altamente sugerido ya sea en habilidades, niveles u otras ideas que fueron descartadas antes del periodo de desarrollo en el caso de que se llegue a una nueva versión del videojuego que implemente los resultados del análisis y los comentarios de los jugadores haciendo cambios que mejoren la experiencia de usuario y creando nuevo contenido que enriquezca el juego.

9.2. Evaluación de usuarios

El trabajo posterior al lanzamiento del videojuego que constó en la obtención de datos de usuarios mediante la encuesta en línea y la obtención de variables de partidas junto con su análisis se pudo realizar sin problemas mayores, al conseguir la cantidad necesaria de datos en las partidas ingresadas a la base de datos y a los usuarios que respondieron voluntariamente la encuesta en línea. Teniendo en cuenta el alcance del proyecto y la cantidad de datos recopilada, el análisis de datos funcionó como una validación de las mecánicas del videojuego, pero dejando una amplia recopilación de información que permita pulir y mejorar el proyecto, lo cual es una de las posibles rutas a tomar a futuro.

La obtención de información de partidas de usuarios y su ingreso al servidor en línea fue realizada exitosamente, con la publicación del videojuego recibiendo usuarios únicos y resultando en más de 100 partidas disponibles para ser analizadas y llegar a una conclusión exitosa de evaluación, con el único posible impedimento en encontrar más partidas siendo que la aplicación no pudo ser publicada con una versión web ya que esta no pudo conectarse correctamente con el servidor que guardaba los datos y los usuarios requirieron descargar el videojuego para entregar sus datos, lo cual pudo desanimar a posibles usuarios a probar el juego.

Paralelamente, con doce usuarios la encuesta en línea demostró ser un recurso importante al momento de obtener realimentación de las características del videojuego que no pudieron ser obtenidas directamente de las partidas. El uso y obtención de habilidades concordaban con los datos evaluados, la dificultad y el entretenimiento videojuego fueron valoradas positivamente, que eran los puntos más importantes al momento de diseñar el videojuego y las características artísticas del juego fueron elogiadas lo cual se apreció, especialmente al saber de qué estas fueron diseñadas completamente por el estudiante y utilizaron conocimiento más allá del aprendido en la universidad. Si bien la cantidad de usuarios que respondieron la encuesta fueron suficientes para llegar a las conclusiones del proyecto, se pudo aumentar este número si se implementara la encuesta directamente al finalizar el juego en vez de mantenerlo en una página web separada.
Finalmente, el análisis de los datos obtenidos mostró un resultado positivo al recibir más de un 50 % de partidas consideradas buenas en la evaluación en base al uso de habilidades para completar niveles, demostrando que los usuarios entendieron las mecánicas principales del juego y como estas afectan la dificultad de este. El uso de KDD y algoritmos de clasificación demostró ser una posibilidad viable al momento de evaluar dificultad en videojuegos si se obtiene una cantidad razonable de datos y, de la misma forma, también puede ser implementada para probar mecánicas experimentales que no tienen una comparación suficiente en otros juegos que funcionen como evidencias para validar un concepto de diseño.
Teniendo estos conocimientos en cuenta, si en el futuro se desea realizar un análisis de datos similar se buscaría probar nuevas formas de evaluar al usuario, ya sea usando datos más específicos con las mecánicas del juego como la frecuencia en la cual se obtienen habilidades o simplemente utilizando distintos algoritmos de clasificación más avanzados que el de Naive Bayes, ya que, aunque el porcentaje de eficiencia se mostró aceptable este puede ser mejorado con una expansión del trabajo realizado.

9.3. Trabajo Futuro

Al llegar al final de todas las tareas planteadas en el proyecto y al ver que los resultados fueron contundentemente positivos se llegó a la conclusión de dos posibles pasos siguientes, que no son mutuamente exclusivos.
La primera es usar este tipo de evaluación de mecánicas en base a datos de partidas

y análisis de resultados en proyectos futuros, la cual demostró ser especialmente efectiva al momento de evaluar y equilibrar dificultad en videojuegos adaptándola a distintos géneros y mecánicas que se deseen trabajar.
La segunda es crear una nueva iteración del proyecto, implementando varios cambios y mejoras que fueron encontradas durante el desarrollo del videojuego y en las fases de análisis. Entre las distintas formas de mejorar el proyecto concluidas se encuentran:

- Implementar un sistema de dificultad adaptiva basada en la evaluación de jugadores, que consiste en evaluar como el usuario se desempeña nivel por nivel y en el caso de que no llegue a una evaluación buena", se realizan cambios en los enemigos y obstáculos del siguiente nivel para que el jugador tenga una mejor experiencia en general.

- Aumentar la cantidad de elementos en el videojuego, al tener un desarrollo expandible se puede agregar más contenido al videojuego. Entre estos elementos se encuentran habilidades desbloqueables, tipos de enemigos y niveles jugables.

- De la misma forma, el juego puede contener nuevas mecánicas sugeridas por los mismos usuarios. Para que se transforme en un producto más completo la implementación de un modo campaña, una tabla de clasificación basada en puntaje y un sistema de guardado son posibles características que implementar.

- Pulir las mecánicas ya existentes con los resultados recibidos en el análisis de datos y la encuesta, con un enfoque en el diseño de niveles, el comportamiento de los enemigos y la cámara del videojuego.

De cualquier forma, la experiencia del proyecto resultó ser gratificante y con un futuro extenso. Ya sea trabajar en el desarrollo del videojuego para una posible versión comercializable en tiendas virtuales como Steam o Epic Games Store, o la obtención y análisis de datos para ser aplicada en distintas iteraciones del proyecto, se demostró tener las habilidades necesarias para la realización de ambas tareas y como estas pueden ser aplicadas en otros videojuegos con una posible escala más grande.

Glosario

Gameplay: Término que define las reglas y características presentadas en su diseño para la interacción del jugador. Usada en la actualidad para categorizar distintos géneros de videojuegos.

Pixel art: Estilo de arte digital caracterizado por dibujar a resolución tamaño pixel, con el propósito de simular graficas de videojuegos encontrados en las primeras generaciones.

Tester: Encargado de probar el juego antes de su publicación para realizar control de calidad.

Hitbox: Técnica utilizada en detección de colisiones. Consiste en una caja invisible que representa el área donde un ataque puede dañar a su objetivo.

Rogue-like: Género de videojuegos caracterizado por creación de niveles procedurales, exploración de mazmorras y muerte permanente.

Slash: Palabra en inglés que refiere a un corte o un ataque realizado con un objeto con filo.

Combo: Secuencia de ataques consecutivos que, por el mayor de los casos, no pueden ser evitados por el receptor.

Assets: Material gráfico o sonoro utilizado para el desarrollo de contenido multimedia.

Demo: Demostración distribuida de un videojuego con la intención de que los consumidores tengan una idea del juego antes de decidir comprarlo.

Open-Source: Software de código abierto que entrega un acceso público, donde todos pueden verificar, modificar y distribuir el código de forma libre.

Input: Comandos entregados en el videojuego por parte del usuario.

Indie: Uno se refiere a un videojuego como *indie* cuando es desarrollado por un solo individuo o un equipo pequeño con un presupuesto limitado. También es usado como filtro de busqueda en videojuegos.

Tileset: Conjunto de texturas unidas en una misma imagen, usualmente usada para la creación grafica de escenarios.

Metroidvania: Subgénero que abarca juegos de plataformas con mapas y niveles no lineales centrados en exploración y desbloqueo de accesos a nuevas áreas. Inspirado por las series de videojuegos "Metroid" y "Castlevania".

Pose idle: En dibujo y animación se refiere a la pose que un personaje realiza al mantenerse quieto por un periodo de tiempo sin realizar ninguna otra acción.

Bug: Error o fallo en un sistema de software que entrega un resultado no deseado.

Game Engagement: Término que refiere a la experiencia individual del usuario cuando este se encuentra completamente envuelto en la actividad que el videojuego entrega.

KDD: *Knowledge Discovery in Databases.*

GML: *GameMaker Language*

IDE: *Integrated Development Environment.*

MMX: Mega Man X.

FSM: *Finite State Machine.*

Bibliografía

[1] Maria-Virginia Aponte, Guillaume Levieux, and Stéphane Natkin. Difficulty in video games : An experimental validation of a formal definition. 2011.

[2] Bethke, Erik. *Game development and production*. Wordware Publishing, Inc., 2003.

[3] Fernando Bevilacqua. Finite-state machines: theory and implementation. http://gamedevelopment. tutsplus. com/tutorials/finite-state-machines-theory-and-implementation-gamedev-11867, 2013.

[4] Fayyad, Usama M and Piatetsky-Shapiro, Gregory and Smyth, Padhraic and others. Knowledge Discovery and Data Mining: Towards a Unifying Framework. In *KDD*, volume 96, pages 82–88, 1996.

[5] Centers for Medicare & Medicaid Services et al. Selecting a development approach. *Centers for Medicare & Medicaid Services*, pages 1–10, 2008.

[6] Grapheverywhere. Tipos de bases de datos. https://www.grapheverywhere.com/tipos-bases-de-datos-clasificacion/, 2020.

[7] Hoganson, Ken. Teaching programming concepts with GameMaker. *Journal of Computing Sciences in Colleges*, 26(2):181–188, 2010.

[8] Ahmed Khalifa, Fernando de Mesentier Silva, and Julian Togelius. Level design patterns in 2d games. In *2019 IEEE Conference on Games (CoG)*, pages 1–8. IEEE, 2019.

[9] Diana Sofía Lora Ariza. Data mining techniques in video games, 2015.

[10] Overmars, Mark. Designing games with game maker, 2013.

[11] Alain Pereira-Toledo, José D. López-Cabrera, and Luis A. Quintero-Domínguez. Estudio experimental para la comparación del desempeño de Naive Bayes con otros clasificadores bayesianos. *Revista Cubana de Ciencias Informáticas*, 11, 2017.

[12] Cody Phillips, Daniel Johnson, and Peta Wyeth. Videogame reward types. 10 2013.

[13] Andrew Przybylski, C. Rigby, and Richard Ryan. A motivational model of video game engagement. *Review of General Psychology*, 14:154–166, 06 2010.

[14] Leonardo Rocco and Natalia Oliari. La encuesta mediante internet como alternativa metodológica. *VII Jornadas de Sociología. Buenos Aires: Facultad de Ciencias Sociales, Universidad de Buenos Aires*, 2007.

[15] Rogers, Scott. *Level Up! The guide to great video game design*. John Wiley & Sons, 2014.

[16] Salen, Katie and Tekinbas, Katie Salen and Zimmerman, Eric. *The game design reader: A rules of play anthology*. MIT press, 2006.

[17] Sam Keddy. Lospect's database of palletes for pixel art. https://lospec.com/palette-list, 2021.

[18] Silber, Daniel. *Pixel art for game developers*. CRC Press, 2015.

[19] Tv Tropes. Sentai Definition. https://tvtropes.org/pmwiki/pmwiki.php/Main/Sentai, 2021.

[20] Ian H Witten and Eibe Frank. Data mining: practical machine learning tools and techniques with java implementations. *Acm Sigmod Record*, 31(1):76–77, 2002.

[21] Yoyo Games. Game Maker Language manual. https://manual.yoyogames.com, 2021.

[22] Yoyo Games. Gamemaker Community. https://www.yoyogames.com/community, 2021.

A. Página de descarga The Shamrock Ranger

Esta es la página oficial donde la versión final del videojuego fue publicada, aquí se encuentra disponible su descarga para que este sea probado.

https://ignaciog.itch.io/the-shamrock-ranger

B. Carta Gantt desarrollo

En este link se encuentra acceso el archivo Excel de la carta grant utilizada para planificar el desarrollo del proyecto. Al contener meses de trabajo se decidió dejarla disponible para ser descargada y revisada por otros en vez de colocarla directamente en este documento. Cambiando el valor de semana se pueden revisar todas las fechas de desarrollo.

https://tinyurl.com/Gantt-TSR

C. Respuestas encuesta feedback

Aquí se pueden apreciar todas las respuestas ingresadas en la encuesta posterior al videojuego que fue entregada a los usuarios, incluyendo las preguntas que no fueron directamente mencionadas en la sección de resultados. Los nombres que fueron ingresados se han omitido ya que no se pidió el consentimiento directo de los que participaron.

https://tinyurl.com/thr-post-form