UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Técnica Superior de Ingeniería Informática

Diseño e implementación de un videojuego con Godot Engine: Ascent Tower

Trabajo Fin de Grado

Grado en Ingeniería Informática

AUTOR/A: Pérez Asensio, Javier

Tutor/a: Abad Cerdá, Francisco José

CURSO ACADÉMICO: 2023/2024

Resumen

Este Trabajo de Fin de Grado consiste en la creación de un mínimo producto viable de un videojuego 2D desde cero. Este desarrollo se ha realizado con el motor de videojuegos *Godot Engine*. Se ha empleado la plataforma *HacknPlan* para facilitar la planificación y el desarrollo ágil del proyecto y *GitHub* para gestionar el proyecto bajo un control de versiones.

El videojuego trata sobre un guerrero que encuentra una torre de un castillo medieval muy alta; tan alta que las nubes no le dejan ver su final. Observando dicha torre escucha la voz de una persona que está pidiendo auxilio y el guerrero decide adentrarse para rescatarla sin saber que está habitada por monstruos y enemigos que le impedirán el ascenso y rescate de la persona en apuros. Durante esta aventura el guerrero irá aumentando su fuerza con los artefactos que vayan dejando los enemigos al ser derrotados para, finalmente, cumplir su objetivo de poner a salvo a la persona que suplicaba por su vida.

Durante el desarrollo del videojuego se aplicarán técnicas de patrones de diseño especializadas para videojuegos y los niveles se irán generando proceduralmente conforme el jugador vaya avanzando por los pisos de la torre del castillo.

Palabras clave: Godot, videojuego 2D, generación procedural, desarrollo de *software*, metodología ágil, patrones de diseño, control de versiones.

Abstract

This Final Degree Project consists of creating a minimum viable product of a 2D video game from scratch. This development has been carried out using the Godot Engine*. The* HacknPlan *platform was used to facilitate the planning and agile development of the project, and* GitHub *was used to manage the project under version control.*

The video game is about a warrior who finds a very tall tower of a medieval castle; so tall that the clouds obscure its top. Observing the tower, he hears the voice of a person calling for help, and the warrior decides to enter the tower to rescue them, unaware that it is inhabited by monsters and enemies that will hinder his ascent and the rescue of the person in distress. During this adventure, the warrior will increase his strength with the artifacts left by the defeated enemies, to ultimately achieve his goal of saving the person pleading for their life.

Throughout the development of the video game, specialized design pattern techniques for video games will be applied, and the levels will be procedurally generated as the player advances through the floors of the castle tower.

Keywords: Godot*, 2D video game, procedural generation,* software *development, agile methodology, design patterns, version control.*

Tabla de contenidos

Tabla de figuras

1. Introducción

Este Trabajo de Fin de Grado (TFG) aborda el desarrollo de un videojuego, para dispositivos móviles, desde su concepción inicial hasta la obtención de un Mínimo Producto Viable (Minimum Viable Product, MVP). Se tratarán aspectos fundamentales como la funcionalidad, objetivos y desafíos, asegurando una experiencia coherente con un comienzo y un final definidos. Para este fin, se empleará Godot Engine, una herramienta de desarrollo especializada en la creación de videojuegos (Hernández Hernández, 2021). Previo al desarrollo, se elaborará un Documento de Diseño de Juego (*Game Design Document*, GDD) que servirá como referencia para establecer la trama y los objetivos principales, previniendo posibles incompatibilidades o malentendidos durante las fases de desarrollo.

El método seleccionado para la gestión del proyecto es el desarrollo ágil, una metodología adaptativa frecuentemente aplicada en la creación de software. Esta aproximación facilita la adaptación a cambios y permite una evolución continua del proyecto a través de iteraciones cortas, conocidas como *sprints*, con una duración de dos semanas cada una. Dentro de esta metodología se utilizará el modelo de historias de usuario (*User Stories*, US) para definir y describir los requerimientos del sistema. Las US son descripciones concisas desde la perspectiva del usuario final, facilitando así un enfoque centrado en las necesidades del usuario (Izaurralde, 2013).

Esta memoria incluye un glosario de términos técnicos al final, con el objetivo de facilitar la comprensión de conceptos especializados que puedan presentar dificultades al lector, o servir como recordatorio si ya han sido explicados previamente en el documento.

Este proyecto pretende contribuir a la literatura existente sobre el desarrollo de videojuegos, proporcionando un caso práctico de metodologías ágiles en un contexto de desarrollo real. Además, se hará uso de patrones de diseño para estructurar el código y mejorar la mantenibilidad del proyecto. Asimismo, se tomó la decisión de utilizar la tecnología Godot Engine, con el propósito de aprender y explorar las capacidades de esta herramienta en el desarrollo de videojuegos.

1.1. Motivación

Este proyecto se ha desarrollado debido a mi interés en el ámbito de la creación de videojuegos, un área que combina habilidades técnicas con entretenimiento digital. La posibilidad de diseñar un espacio donde los usuarios puedan explorar y disfrutar, rememorando la magia de sus primeras experiencias con videojuegos, ha sido una motivación fundamental. Este trabajo ha ofrecido también la oportunidad de aplicar de manera práctica los conocimientos adquiridos durante la formación académica, contribuyendo así al campo de desarrollo de videojuegos con la implementación de nuevas técnicas y herramientas.

1.2. Objetivos

El objetivo principal de este proyecto es la creación de un videojuego 2D, para dispositivos móviles, desde su concepción inicial hasta alcanzar el MVP utilizando el entorno de desarrollo de videojuegos Godot Engine. Durante el desarrollo se realizará un estudio en profundidad de las características que esta herramienta proporciona al desarrollador.

A la hora de desarrollar el objetivo principal, se desarrollarán también los siguientes objetivos secundarios:

- durante toda la duración del proyecto, hacer uso de metodologías ágiles para optimizar el proceso de desarrollo,
- el uso de patrones de diseño durante el proceso de desarrollo del código del videojuego para mantener la estructura del código lo más coherente y eficiente posible, con un código limpio y aplicando buenas prácticas, e
- implementar un sistema de niveles con generación procedural que permita la creación dinámica de los escenarios a medida que el jugador avanza por el juego.

1.3. Impacto Esperado

El impacto esperado para el usuario consistirá en la experiencia de participar en una aventura ambientada en la época medieval, dentro de una torre de un castillo de la época. Inicialmente, el usuario puede experimentar sentimientos de inmersión, desafío y soledad debido a la música, los enemigos y la estética del juego.

A medida que avance en el videojuego, el usuario podrá alcanzar los objetivos planteados, como lograr superar su anterior récord, generando una sensación de satisfacción y cumplimiento, fomentando así su compromiso continuo con la exploración del videojuego.

El videojuego proporcionará al usuario un escape temporal de la realidad, permitiéndole sumergirse en una narrativa ficticia repleta de desafíos y metas a superar, lo que puede contribuir a su entretenimiento y bienestar.

El proyecto *Ascent Tower* se alinea con diversos Objetivos de Desarrollo Sostenible (ODS). Contribuye de manera indirecta al ODS 3 (Salud y Bienestar) al ofrecer una experiencia de juego que puede mitigar el estrés y potenciar habilidades cognitivas. En relación con el ODS 4 (Educación de Calidad), el juego fomenta el desarrollo de habilidades estratégicas y cognitivas mediante la resolución de problemas, facilitando el aprendizaje activo y el crecimiento personal. En cuanto al ODS 9 (Industria, Innovación e Infraestructura), Ascent Tower emplea tecnologías avanzadas como generación procedural creado en Godot Engine y uso de metodologías ágiles para el desarrollo de videojuegos, impulsando la innovación y una industrialización más inclusiva y sostenible. Además, el juego integra características accesibles como ajustes de botones personalizables y soporte para diversos dispositivos de entrada, promoviendo así el ODS 10 (Reducción de las Desigualdades) al fomentar la inclusión y reducir las barreras de acceso al entretenimiento digital.

1.4. Metodología

En este proyecto de creación de un videojuego, se aplicará la metodología ágil de software tipo *Scrum* y *Kanban*.

Scrum es una de las metodologías ágiles más populares hoy en día, especialmente en el ámbito del desarrollo y mantenimiento de proyectos de software. Se basa en un proceso iterativo que divide el desarrollo del producto en ciclos llamados sprints. En cada sprint, el equipo trabaja con una lista priorizada de requisitos, entregando al final de cada ciclo un producto funcional. Esto permite optimizar la previsibilidad y el control de riesgos. Los principales beneficios de Scrum incluyen: entrega de resultados prioritarios ya finalizados; gestión continua de las expectativas del cliente basada en resultados tangibles; reducción del tiempo de comercialización (*time to market*); flexibilidad y adaptación a las necesidades del cliente y cambios del mercado; gestión sistemática del Retorno de Inversión (ROI); mitigación de riesgos del proyecto; aumento de la productividad y calidad; y una mejor alineación entre el cliente y el equipo de desarrollo (Riano Nossa, 2021).

Por otro lado, Kanban, una metodología ágil visual, destaca por su capacidad para mejorar el flujo de trabajo y promover la adaptación continua en los equipos de desarrollo. La representación del flujo de trabajo mediante un tablero Kanban proporciona una transparencia que permite al equipo entender el estado de las tareas y las prioridades. Esto facilita una distribución más equilibrada de la carga de trabajo y la detección temprana de cuellos de botella (Pérez Tapia, 2023).

Para asegurar el correcto funcionamiento de la metodología Scrum, se realizará primero un análisis del problema mediante Casos de Uso (*Use Cases*, UC). Posteriormente, se definirán las US, que describirán las funcionalidades a implementar junto con estimaciones de tiempo y sus posibles dependencias, como se observa en la Ilustración 1. Cada US se desglosará en tareas específicas cuya ejecución y finalización llevará a la realización completa de dicha US. Para garantizar la calidad, cada tarea estará acompañada por una o varias Pruebas de Aceptación (*User Acceptance Testing*, UAT) que deberán cumplirse antes de considerarla finalizada (ver Ilustración 2). Estas UAT asegurarán que los requisitos se implementen correctamente y funcionen según lo esperado.

Cada US se planificará con una duración inicial, lo que permitirá tener una estimación temporal de la nueva función. Una vez que se definan las tareas específicas, estas se ajustarán dentro de ese límite de horas estimadas en la US. Se contempla la posibilidad de reevaluar estas horas, ya sea incrementándolas o reduciéndolas, según sea necesario.

La gestión de este proceso se llevará a cabo a través de la plataforma HacknPlan, una herramienta que facilita el seguimiento, la planificación y la organización de proyectos. Está especialmente diseñada para la industria de los videojuegos y es utilizada por numerosos estudios de desarrollo. Este sitio *web* permite crear tareas y asignarles categorías, además de permitir al desarrollador establecer la prioridad de la tarea y su estado, ya sea pendiente, en progreso, pendiente de revisión o completada (Raurell Gomis, 2022).

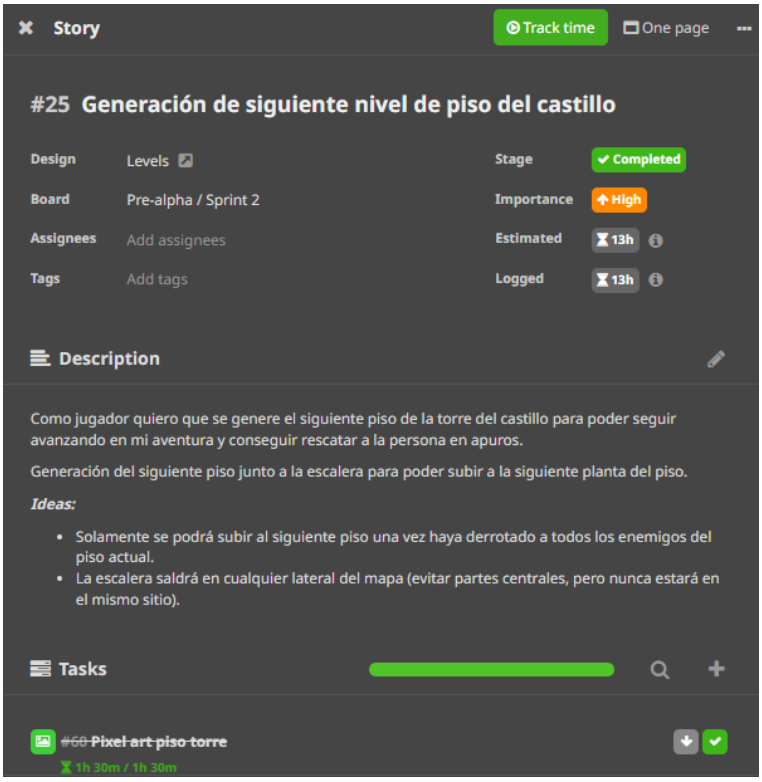

Ilustración 1. Ejemplo de US con nivel de prioridad, estimación de horas, descripción y tareas definidas en HacknPlan.

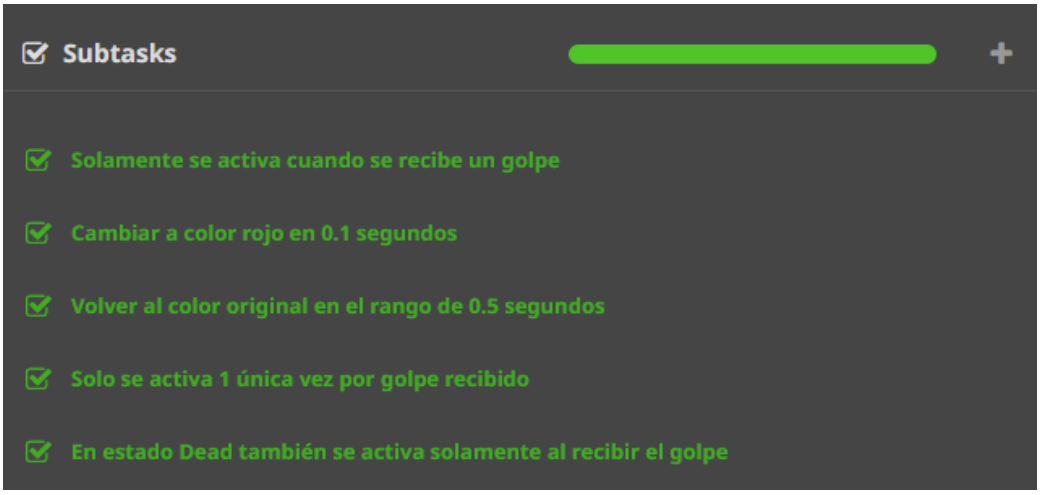

Ilustración 2. Ejemplo de UAT cumplidas de una US de HacknPlan.

El proyecto se organizará en iteraciones de dos semanas conocidas como sprints, cada una con una dedicación de 35 horas-hombre. La herramienta HacknPlan permite establecer un *backlog* que contiene tareas a realizar en futuros sprints o en el sprint actual, ordenado conforme la prioridad definida en las tareas. En total, se han planificado ocho sprints con el objetivo de alcanzar un MVP, lo que marcará la finalización de la primera versión completa del videojuego.

Con cada finalización del sprint, el avance será mostrado al *Product Owner* (PO), quien dará el visto bueno de las tareas realizadas y finalizadas. Este proceso es crucial, ya que la aprobación del PO permite continuar con la planificación establecida del siguiente sprint, creando así una rama específica en la plataforma GitHub junto con una etiqueta de *release* de la versión de revisión del sprint.

En cada sprint, HacknPlan ofrece una estructura de tareas organizada según el método Kanban, dividida en cuatro secciones clave, como se muestra en la Ilustración 3, siendo estas: *Planned* (Planificado), *In Progress* (En progreso), *Testing* (Pruebas) y *Completed* (Completado). Se ha decidido seguir el siguiente esquema para cada uno de los apartados que contiene HacknPlan:

- Planned: en esta sección estarán planificadas todas las US a realizar en dicho sprint, será en este momento en el que, por cada US, se definirán sus tareas y por cada tarea creada se definirá sus UAT correspondientes para cumplir con el objetivo de la tarea. La tarea se mantendrá en esa sección mientras no se empiece. Este apartado estará ordenado por prioridad de US, que determinará el orden de implementación.
- In Progress: esta es la segunda sección del Kanban proporcionado por HacknPlan. Solo habrá una tarea de la US por persona asignada al proyecto en todo momento. A esta sección se llevarán las tareas que se estén realizando en cada momento. Una vez finalizada esta tarea, será llevada al siguiente estado del Kanban, siendo este el de comprobación de cumplimiento de las UAT, llamado Testing.
- Testing: en este penúltimo apartado del Kanban se analizará si la tarea desarrollada cumple con los objetivos propuestos en la tarea y en las UAT creadas anteriormente. En caso de que alguna de ellas no se cumpla, esta tarea volverá a la sección anterior de In Progress. De lo contrario, si todas las UAT se cumplen satisfactoriamente, la tarea se pasará al siguiente estado, el de Completed.
- Completed: este es el último estado del Kanban, donde se almacenarán las tareas que ya hayan sido completadas y hayan pasado satisfactoriamente por todos los estados anteriores. El orden en el que se almacenarán las tareas y US en este estado será por orden de completado, que deberá tener una estructura similar a la inicial de cada US salvo que la US, en este caso, en vez de encontrarse en la parte superior de la planificación, se encontrará bajo sus tareas completas. Esto es debido a que una US no se dará por finalizada hasta que todas sus tareas no estén completas, quedando la US en el estado In Progress hasta que finalmente se complete la última tarea de la US y se hayan hecho todas las UAT correspondientes a dicha US para comprobar que se cumplen todas las condiciones en su conjunto y se ha realizado exactamente lo que la US solicitaba en su descripción.

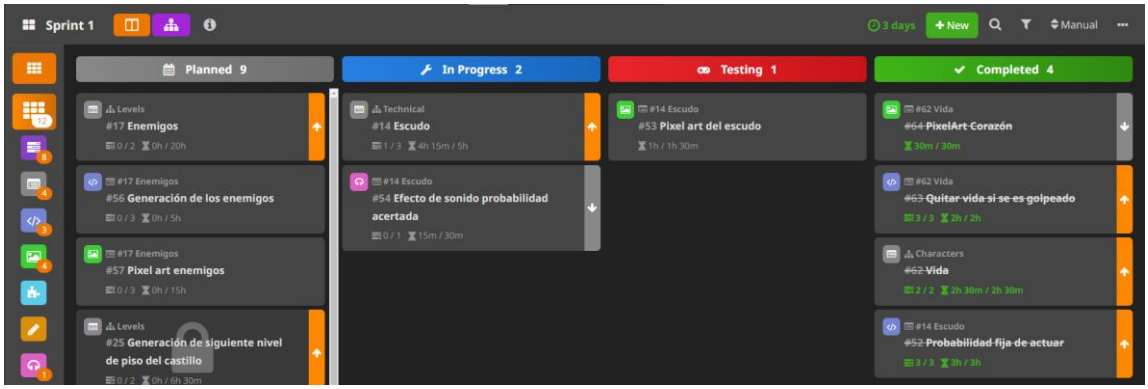

Ilustración 3. Distribución de los posibles estados de una tarea o US en HacknPlan.

1.5. Estructura

La estructura de este TFG se ha diseñado para guiar al lector a través del proceso de desarrollo y las decisiones tomadas en la creación del videojuego Ascent Tower. A continuación, se describe brevemente el contenido de cada capítulo y los anexos.

- **Introducción:** este capítulo establece el contexto del proyecto, incluyendo la motivación, los objetivos y el impacto esperado. Se presenta una visión general de la metodología utilizada y se describen las colaboraciones y convenciones adoptadas durante el desarrollo del TFG.
- **Estado del arte:** aquí se realiza un análisis de los motores de videojuegos disponibles, así como un estudio de la competencia. Se justifica la elección de Godot Engine y se discute la propuesta del proyecto en relación con el estado actual de la industria.
- **Análisis del problema:** este capítulo se centra en identificar y analizar el problema que el videojuego pretende resolver. Se incluyen estudios sobre la eficiencia algorítmica, análisis de dispositivos, factores importantes de la construcción del proyecto y la solución propuesta en base a US. Además, se presenta el plan de trabajo que guiará el desarrollo del proyecto.
- **Diseño de la solución:** se describe la arquitectura del sistema y la distribución de las clases en las diferentes capas lógicas del software. Este capítulo también detalla las técnicas de patrones de diseño aplicadas y las diferentes herramientas usadas para lograr el desarrollo correcto del proyecto.
- **Desarrollo de la solución propuesta:** en este capítulo, se documenta el proceso de implementación del videojuego. Se presentan las diferentes fases del desarrollo, los sprints realizados, y se analiza el progreso del proyecto a lo largo del tiempo.
- **Implementación:** en este capítulo, se documenta el proceso de puesta en marcha del videojuego.
- **Pruebas:** aquí se detallan las pruebas realizadas para comprobar la calidad del videojuego con los jugadores. Se describen los resultados obtenidos de los jugadores tras la implantación del juego.
- **Conclusiones:** este capítulo resume los logros del proyecto y la relación que hay entre el trabajo realizado y los estudios cursados.

- **Trabajos futuros:** reflexiona sobre los desafíos encontrados y propone posibles mejoras y extensiones futuras para el videojuego.
- **Anexos:** los anexos proporcionan información adicional que complementa el contenido principal del TFG. Incluyen documentación sobre la relación del proyecto con los objetivos de desarrollo sostenible, así como el documento de diseño de videojuegos, en el cual se profundiza en aspectos más específicos sobre el videojuego.

1.6. Colaboraciones

Durante el desarrollo de este TFG, ha habido una colaboración destacada que ha sido esencial para llegar al resultado final. Esta colaboración ha consistido en la composición y producción completa de la música del juego, llevada a cabo por un agente externo al proyecto, Mario Mocholi Rubio.

Mario Mocholi Rubio ha creado todas las piezas musicales que se escuchan en el videojuego, cubriendo diversos aspectos y momentos clave de la experiencia de juego. Esto incluye:

- Tema principal: la música que suena en la pantalla inicial de menú principal, diseñada para dar la bienvenida al jugador y establecer el tono de espera antes de jugar una partida.
- Música de juego: las piezas que acompañan al jugador durante las partidas, adaptadas para intensificar la inmersión y la emoción del juego, provocando en el jugador una sensación de soledad y temor.
- Música de la tienda de artefactos: una composición específica que se reproduce cuando el jugador accede a la tienda dentro del juego, proporcionando un ambiente adecuado para la selección y compra de nuevos y mejores equipos.

La colaboración con Mario Mocholi Rubio ha sido fundamental para añadir una capa adicional de calidad y profesionalismo al videojuego, enriqueciendo la experiencia del usuario y complementando los demás elementos del proyecto con una banda sonora cuidadosamente elaborada.

1.7. Convenciones

En este TFG, se seguirá una convención específica en cuanto al marcado de palabras extranjeras. La primera aparición de todas las palabras que no pertenezcan al idioma principal del documento se resaltarán en cursiva.

Esta convención tiene como objetivo principal facilitar la comprensión del texto para los lectores y se aplicará de manera consistente a lo largo de todo el documento, asegurando una comunicación efectiva y precisa con los lectores, independientemente de su nivel de familiaridad con ciertos términos o con los idiomas extranjeros.

2. Estado del arte

En los siguientes apartados, se realizará un estudio de los diferentes motores de videojuegos y de los videojuegos similares al que se presenta que se encuentran disponibles en el mercado.

2.1. Análisis de motores de videojuegos

En el ámbito del desarrollo de videojuegos, hay varios motores que facilitan su creación y desarrollo.

Un motor de videojuegos es un programa informático diseñado para proporcionar las herramientas esenciales para la creación de un videojuego. Incluye un motor de renderizado que soporta gráficos tanto en 2D como en 3D, sistemas de detección de colisiones, gestión de sonidos, *scripting*, animación, inteligencia artificial, funcionalidades de red, *streaming*, manejo de memoria, entre otras capacidades (Arce, 2011).

Las principales tecnologías que se encuentran en el mercado son Godot, *Unity* y *Unreal Engine* (Solís Flores, 2023).

Godot (Godot Engine, 2024) es un proyecto de código abierto y gratuito que sigue evolucionando gracias a las contribuciones de su comunidad. Originalmente, estaba enfocado en la creación de juegos 2D, pero recientemente ha ampliado sus capacidades para incluir también el desarrollo de juegos 3D. Al igual que Unity, permite exportar juegos a diversas plataformas, incluyendo *Windows*, *macOS*, *Linux* y BSD (Hernández Hernández, 2021).

Para crear juegos en Godot, es necesario aprender a manejar *GDScript*, su lenguaje específico, aunque ahora también se permite el uso de C#, C++ y el scripting visual. A pesar de ser un motor relativamente reciente, lanzado por primera vez en 2014, Godot está en constante crecimiento y se está posicionando como una alternativa cada vez más viable para ciertos proyectos, aunque aún necesita desarrollo para alcanzar el nivel de otros motores establecidos como Unity o Unreal Engine (Hernández Hernández, 2021).

Unity (Unity, 2024) es una plataforma de desarrollo de videojuegos desarrollada por Unity *Technologies*, disponible para Windows, macOS y Linux. Esta plataforma soporta la compilación en múltiples tipos de dispositivos y ofrece dos ediciones: Unity *Professional* y Unity Personal. Unity Technologies, fundada en 1988 en Copenhague, Dinamarca, por David Helgason, Nicholas Francis y Joachim Ante, surgió tras el fracaso de su primer videojuego, *GooBall*. Reconociendo el potencial de las herramientas de desarrollo que habían creado, decidieron ofrecer su motor de juego a un precio asequible para facilitar su uso por parte de desarrolladores independientes, quienes generalmente no tienen los recursos para desarrollar sus propios motores o adquirir herramientas costosas. Unity busca facilitar y democratizar la creación de juegos y contenidos interactivos en 2D y 3D para desarrolladores alrededor del mundo (Llubes Cano, 2022).

Por último, el motor Unreal Engine (Unreal Engine, 2024), desarrollado por *Epic Games* (Epic Games, 2024) y lanzado inicialmente en 1998, se enfocaba en juegos de disparos en primera persona y ha sido fundamental en títulos como Unreal *Tournament, BioShock, PUBG* y *Mass Effect*. Desde la versión 4, este motor se ofrece de manera gratuita, incluyendo todas sus actualizaciones. Entre sus características destacadas están su avanzado motor de físicas, notables capacidades gráficas y estéticas, reflejos en tiempo real, un eficiente sistema de *raycasting* y una gestión de transparencias mejorada. Además, el motor está diseñado para facilitar la localización de los juegos (Lizarraga, 2017).

Unreal Engine se puede emplear en múltiples ámbitos como el desarrollo de videojuegos, la educación y la arquitectura. No obstante, si algún proyecto usando este motor se comercializa, Epic Games exige un 5% de los ingresos trimestrales de la obra, siempre que estos superen los 3,000 dólares.

Finalmente, se ha decidido utilizar Godot Engine para el desarrollo del juego en 2D destinado a dispositivos móviles con requisitos gráficos moderados. Esta elección se debe a su facilidad de uso, la abundante documentación disponible en internet y el valor del aprendizaje de esta tecnología.

2.2. Análisis de la competencia

En este apartado se comparan diferentes videojuegos disponibles en el mercado similares al que se desarrollará.

Ilustración 4. Portada del videojuego *Tower of Hero*.

Ilustración 5. Videojuego Tower of Hero.

Como competencia directa se encuentra el videojuego Tower of Hero (Google Play, 2024) (ver Ilustración 4 e Ilustración 5), diseñado exclusivamente para dispositivos móviles. Este juego desafía a los jugadores a ascender por una torre llena de enemigos que deben ser derrotados para avanzar al siguiente piso, combinando elementos de acción y estrategia incremental. A medida que el jugador progresa y derrota enemigos, obtiene oro que se puede utilizar para mejorar las habilidades de los héroes, reclutar aliados como magos, guerreros, ninjas, entre otros, y comprar objetos en la tienda del juego. La jugabilidad de los héroes en Tower of Hero es completamente automática; solo es necesario posicionarlos en el piso deseado y ellos atacan automáticamente, permitiendo al jugador concentrarse en mejorar sus personajes con el oro obtenido.

Ilustración 6. Portada del videojuego *Sling Kong*.

Ilustración 7. Videojuego Sling Kong.

Otro ejemplo notable con características similares y exclusivo para dispositivos móviles es Sling Kong (Google Play, 2024), mostrado en la Ilustración 6 e Ilustración 7, donde el objetivo es ascender lo más alto posible mientras se esquivan obstáculos y se recolectan monedas y *power-ups*. La mecánica se centra en la sincronización y la precisión: los jugadores deben soltar al personaje en el momento adecuado para agarrarse a la siguiente plataforma y evitar caídas. Ha sido desarrollado por *Protostar*, un estudio *indie* que también tiene otros juegos con mecánicas comparables.

Ilustración 8. Portada del videojuego *Super Starfish*.

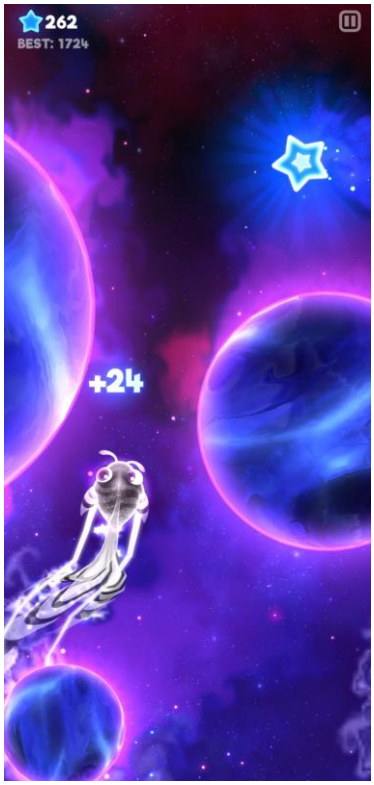

Ilustración 9. Videojuego Super Starfish.

Por último, otro competidor destacado, que se puede observar en la Ilustración 8 e Ilustración 9, es Super Starfish (Google Play, 2024), también exclusivo para dispositivos móviles y desarrollado por los mismos creadores de Sling Kong. En este juego, el objetivo es viajar lo más lejos posible en un recorrido infinito, recolectando estrellas y esquivando obstáculos. Su destacado diseño visual, junto con una música y Efectos de Sonido (*Sound Effects*, SFX) envolventes, contribuyen a una experiencia relajante para los jugadores. La mecánica principal implica deslizar el dedo por la pantalla para guiar al personaje a través del espacio, evitando asteroides y otros peligros.

2.3. Propuesta

Este TFG se enfoca en el desarrollo de un videojuego utilizando la tecnología Godot Engine, desde su concepción hasta la creación de un MVP. La singularidad de este videojuego, en comparación con otros del mercado, reside en su enfoque en la aleatoriedad como elemento fundamental durante la experiencia de juego. Esta característica confiere una mayor dinámica y jugabilidad al título, asegurando la singularidad de cada partida y evitando la repetición monótona.

La generación de mapas se llevará a cabo de manera procedural, lo que implica que la disposición y características de los escenarios se generarán automáticamente, contribuyendo así a la variabilidad y frescura de cada sesión de juego. Además, el videojuego incluirá eventos aleatorios, como la aparición de cofres, cuya ocurrencia estará determinada por diversos factores, como el nivel de la torre en la que se

encuentre el jugador. Asimismo, todos los elementos que los enemigos dejen caer al ser derrotados seguirán también un patrón de aleatoriedad en cuanto a cantidad y rareza, agregando un nivel adicional de imprevisibilidad y emoción al juego. Esto se asemeja mucho a los videojuegos Sling Kong y Super Starfish, donde el contenido se genera dinámicamente conforme el jugador avanza, creando así un ambiente menos repetitivo y más dinámico.

Aunque existen videojuegos que abordan temáticas similares, el mercado de los dispositivos móviles en este ámbito es relativamente limitado debido a las complejidades inherentes a su implementación. Los desafíos significativos de este contexto incluyen el desarrollo y ajuste constantes necesarios para garantizar un funcionamiento óptimo y una experiencia de juego satisfactoria (Tovstochub & Zinchenko, 2024).

Por otra parte, se desea realizar que el videojuego se avance solamente en ascenso, como ocurre en Super Starfish. A diferencia de Tower of Hero y Sling Kong, donde se permite el retroceso a puntuaciones o niveles anteriores, en el videojuego propuesto no se desea esta mecánica.

Además, se destaca que las recompensas obtenidas tras gastar la economía conseguida en las partidas puedan beneficiar al jugador en futuras partidas, mediante la compra de pociones en la tienda del juego y su uso en cualquier momento durante una partida. Esto es similar a lo que ocurre en Tower of Hero, donde los recursos obtenidos se pueden gastar en mejorar héroes o comprar nuevos, facilitando la superación de niveles. En contraste, en Sling Kong y Super Starfish, los recursos se gastan en obtener nuevos personajes que no aportan beneficios adicionales a la jugabilidad o, en el caso de Super Starfish, en la decoración del menú principal.

Otro aspecto destacado es que en este videojuego solo hay un personaje jugable, el cual debe ser el jugador el que mueva a ese personaje, a diferencia de Tower of Hero, que cuenta con varios personajes con diferentes habilidades y su jugabilidad es automática, el jugador solo necesita posicionarlos en el lugar del mapa más adecuado. Esta diferencia también se refleja en la recolección de la economía, mientras que en Tower of Hero esta recolección es automática, en juegos como Sling Kong y Super Starfish, al igual que en Ascent Tower, es responsabilidad del jugador recoger estos recursos.

Con este proyecto, se pretende no solo presentar una propuesta innovadora en el ámbito de los videojuegos móviles, sino también explorar las posibilidades y limitaciones técnicas asociadas a la integración de la aleatoriedad como elemento central de diseño en este tipo de plataformas. Al mismo tiempo, se pretende adquirir conocimientos y experiencia en el uso de una herramienta emergente para el desarrollo de videojuegos como Godot Engine, que actualmente está ganando popularidad en la industria (Holfeld, 2023).

3. Análisis del problema

Para realizar el análisis del problema de este proyecto, se han elaborado diferentes diagramas, incluyendo el diagrama de actores, el diagrama de contexto y los diagramas de UC. Asimismo, para definir las tareas, se ha optado por el uso de US con sus respectivas descripciones, tareas y UAT. Los diagramas desarrollados son los siguientes:

Ilustración 10. Diagrama de actores.

En relación con el diagrama de actores de la Ilustración 10, se identifica únicamente los siguientes tres actores:

Jugador: persona que juega al videojuego e interactúa con el sistema realizando todas las acciones manuales, como pulsar los diferentes botones para moverse, atacar o navegar entre menús.

Enemigo: sistema automático controlado por el propio dispositivo. Este controlará, mediante un algoritmo, todos los movimientos de los enemigos.

Sistema: también es un sistema controlado automáticamente por el dispositivo del usuario. Este gestionará todas las acciones que se realizan al pulsar los botones de una partida o de menús y se encargará de actualizar los valores correspondientes en la economía del juego, así como de los diferentes artefactos que el jugador vaya recolectando. Además, se encargará de posicionar y gestionar los *timeouts* necesarios para realizar las acciones necesarias.

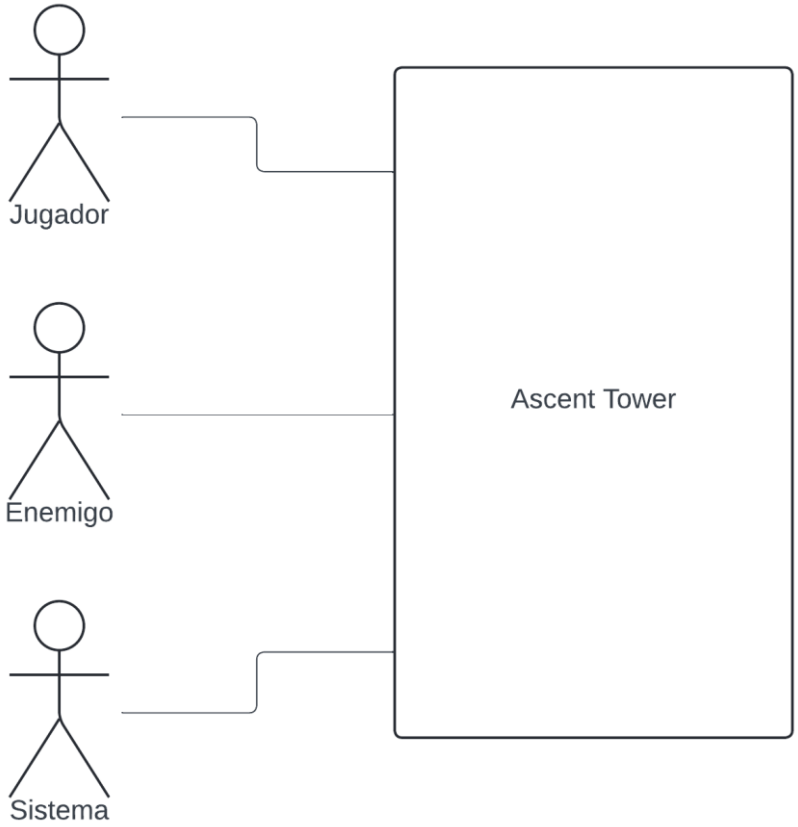

Ilustración 11. Diagrama de contexto del sistema Ascent Tower.

En cuanto al diagrama de contexto de la Ilustración 11, se observa cómo los diferentes actores interactúan con el sistema del videojuego Ascent Tower. Este sistema no tiene agentes externos, lo cual se debe a la decisión de desarrollar el videojuego de manera *offline* y para un solo jugador (*single-player)*. Debido a esta configuración, solo interaccionan el jugador y el propio sistema, que controla los diferentes enemigos presentes en la partida y gestiona las acciones realizadas por el usuario a la hora de pulsar los botones a lo largo del videojuego.

Por último, se presentan los diagramas de UC de los actores ya analizados anteriormente. Para las descripciones de estos UC se ha decidido seguir la estructura de guion narrativo.

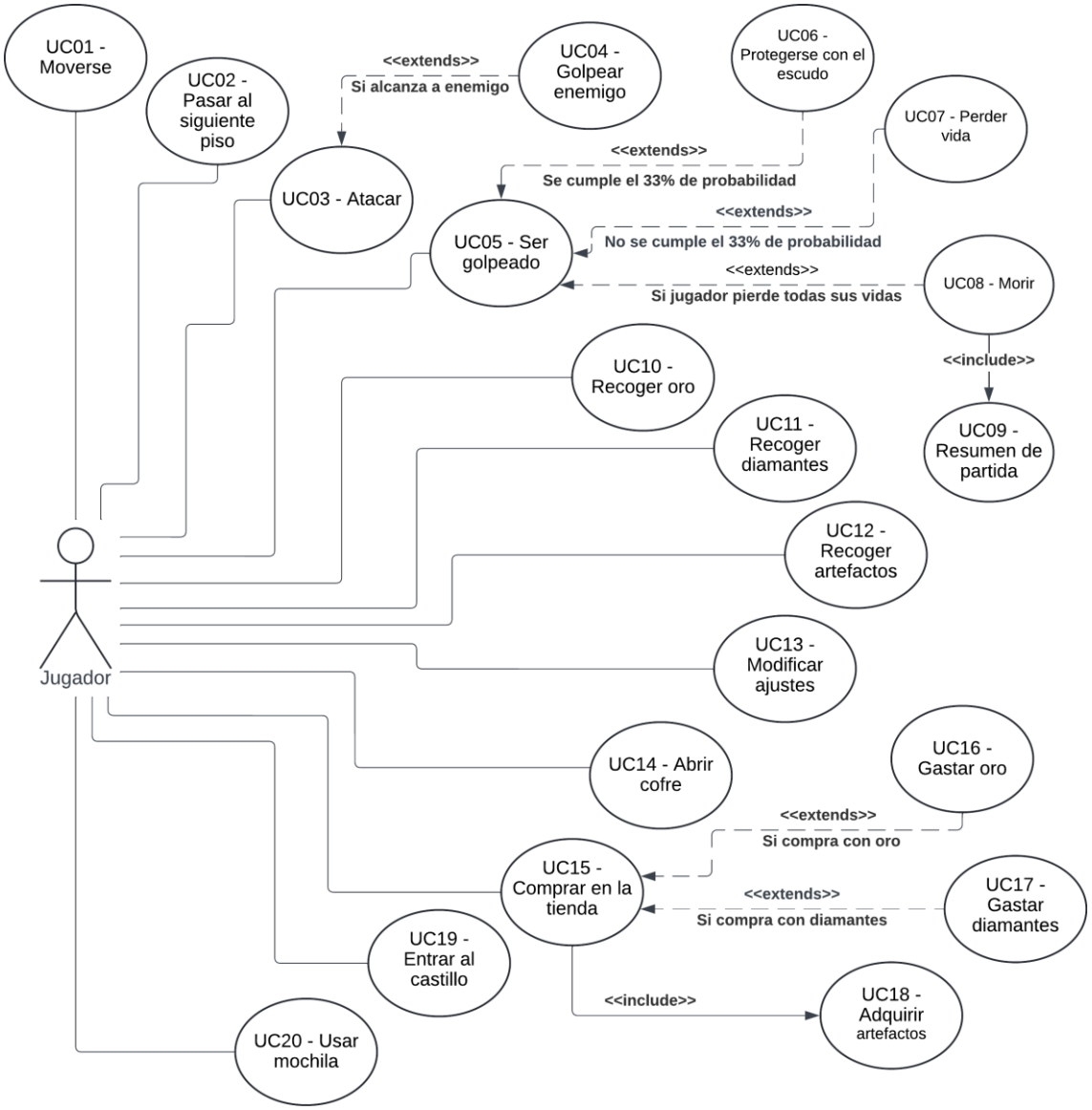

Ilustración 12. Diagrama de UC del jugador.

En cuanto al diagrama de UC del jugador, en la Ilustración 12 se observa que es el que contiene más UC en comparación con los demás actores del sistema. Los UC se definen de la siguiente manera:

UC01 – Moverse.

- Descripción: El usuario podrá moverse de lado a lado dentro de los límites permitidos del mapa utilizando los botones de movimiento.

UC02 – Pasar al siguiente piso.

- Descripción: El jugador podrá avanzar en la torre del castillo tras la generación del siguiente piso, incluyendo la escalera y el *pop-up* permitiendo ascender por ella y alcanzar la siguiente planta de la torre.

UC03 – Atacar.

- Descripción: El jugador podrá atacar, activando su animación correspondiente. Se podrá atacar tanto si hay enemigos cercanos como si no. Entre ataque y ataque habrá una pequeña pausa.

UC04 – Golpear enemigo.

- Descripción: Si el ataque del jugador alcanza a un enemigo, el enemigo sufrirá los daños del ataque recibido.

UC05 – Ser golpeado.

- Descripción: El personaje del jugador puede recibir golpes de los enemigos que le hayan lanzado un ataque.

UC06 – Protegerse con el escudo.

- Descripción: En caso de que se cumpla la probabilidad del 35%, el jugador no recibirá daños de los ataques enemigos, ya que se habrá hecho uso de su escudo, el cual no se desgastará.

UC07 – Perder vida.

- Descripción: Si no se cumple la probabilidad del 35% (65% restante), el ataque de los enemigos afectará al jugador y se perderá una de las vidas del personaje.

UC08 – Morir.

- Descripción: Si el jugador se queda sin vidas tras recibir un ataque de los enemigos, se representará la muerte del personaje, terminando así la partida, sin posibilidad de continuación.

UC09 – Resumen de partida.

- Descripción: Tras la muerte del jugador, se mostrará un resumen de la partida realizada. Se mostrarán diferentes datos como la duración de la partida, el oro conseguido, los diamantes conseguidos, la cantidad de pisos superados, la cantidad de enemigos derrotados, etc.

UC10 – Recoger oro.

- Descripción: El jugador podrá recoger las bolsas de oro que dejen los enemigos al ser derrotados. Estas bolsas contendrán una cantidad variable de monedas de oro, entre 5 y 20.

UC11 – Recoger diamantes.

- Descripción: El jugador podrá recoger los diamantes que los enemigos dejen caer al ser derrotados. Estos diamantes tendrán más valor en la tienda y serán más difíciles de conseguir que el oro.

UC12 – Recoger artefactos.

- Descripción: Se podrá recoger los diferentes artefactos dejados por los enemigos al ser derrotados, como armaduras, armas, escudos, pociones y amuletos.

UC13 – Modificar ajustes.

- Descripción: Se podrá modificar los ajustes del juego, afectando la música, los SFX y la posición de los botones. Esta función también sirve como botón de pausa durante la partida.

UC14 – Abrir cofre.

- Descripción: El usuario podrá encontrar cofres de forma aleatoria en los pisos del castillo (máximo un cofre por piso). Si hay enemigos en el piso, deberán ser derrotados antes de interactuar con el cofre. Una vez abierto se recibirán recompensas de oro, diamantes y/o artefactos dependiendo del nivel en el que se encuentre.

UC15 – Comprar en la tienda.

- Descripción: En la tienda del juego, el jugador puede gastar el oro y los diamantes obtenidos para comprar nuevo equipamiento, como pociones. Al comprar con diamantes, se podrá adquirir objetos más raros y mejores.

UC16 – Gastar oro.

- Descripción: El jugador puede usar el oro para comprar diferentes artículos disponibles en la tienda.

UC17 – Gastar diamantes.

- Descripción: El jugador puede usar los diamantes para comprar artículos de mayor calidad y rareza que los disponibles con oro en la tienda.

UC18 – Adquirir artefactos.

- Descripción: Tras gastar oro o diamantes en la tienda, el jugador adquirirá los artefactos comprados, los cuales podrán ser equipados desde la mochila.

UC19 – Entrar al castillo.

- Descripción: Algunos pisos del castillo tendrán una puerta que lleva a una zona interna del castillo. En esta zona, el jugador puede avanzar más tranquilamente, ya que no hay enemigos.

UC20 – Usar mochila.

- Descripción: El jugador dispondrá de una mochila que tendrá que administrar para no quedarse sin espacio durante la partida y seleccionar los objetos que más le convenga en cada momento. Además, contará con un acceso rápido de tres espacios donde podrá colocar las pociones que desee utilizar de manera más accesible.

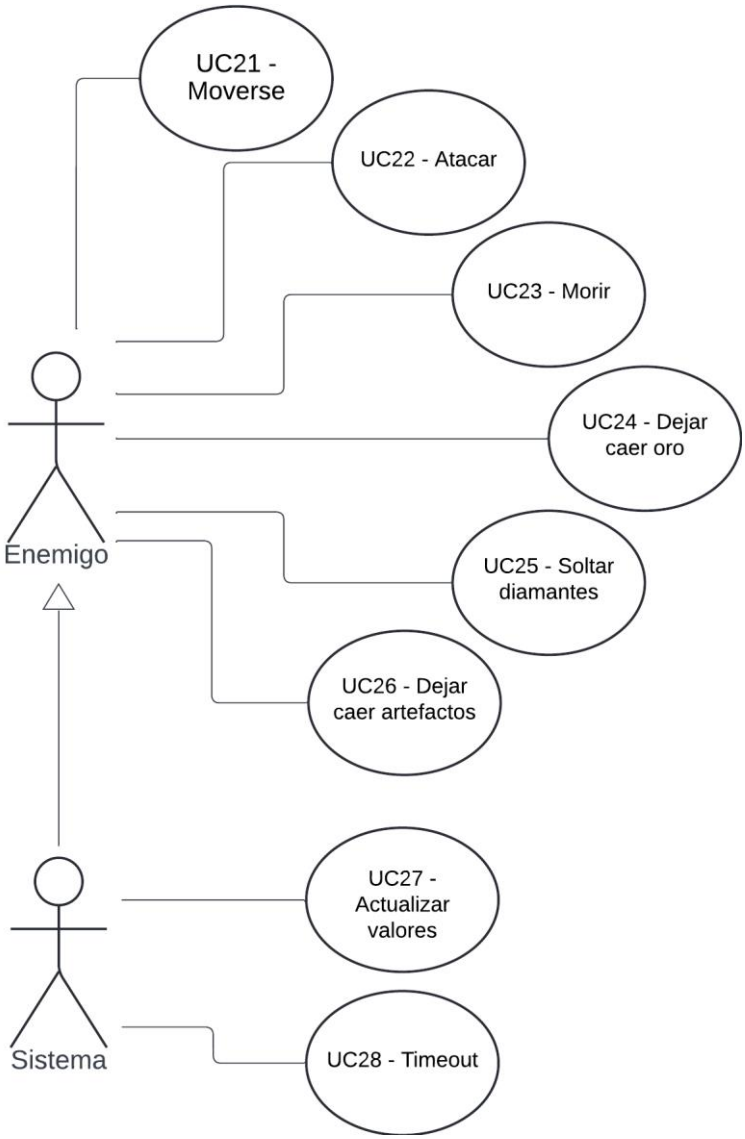

Ilustración 13. Diagrama de UC del enemigo/sistema.

Respecto a los UC del enemigo de la Ilustración 13, estos serán controlados por el sistema y las acciones de los usuarios, por lo que se ha decidido separar a estos dos actores en lugar de agruparlos. Por ejemplo, el enemigo solo atacará al jugador si este entra en su zona de ataque; de lo contrario, simplemente se moverá por el piso en el que se encuentre. Los UC se definen a continuación:

UC21 – Moverse.

- Descripción: El enemigo tendrá libertad de moverse dentro de los límites del mapa. Este movimiento será totalmente aleatorio, alternando entre caminar y quedarse quieto.

UC22 – Atacar.

- Descripción: Siempre que el jugador se encuentre dentro del rango de ataque del enemigo, este intentará golpearlo para defender la torre del castillo.

UC23 – Morir.

- Descripción: Cuando la vida del enemigo llegue a cero, morirá, representándose con una animación de muerte y dejando caer sus recompensas de oro y diamantes.

UC24 – Dejar caer oro.

- Descripción: El enemigo siempre dejará caer una bolsa de oro al ser derrotado por el jugador. La cantidad de oro varía entre 5 a 20 monedas por bolsa.

UC25 – Soltar diamantes.

- Descripción: El enemigo tendrá una probabilidad del 20% de dejar caer un diamante al ser derrotado.

UC26 – Dejar caer artefactos.

- Descripción: Cuando el enemigo es derrotado, dejará caer una de las cuatro partes de los artefactos.

En cuanto al sistema, la definición de sus UC es la siguiente:

UC27 – Actualizar valores.

- Descripción: Cuando el jugador consiga una bolsa de oro, diamantes, los valores deberán representarse en pantalla. Esto también ocurre con el número de piso en el que se encuentre y el espacio de la bolsa o los objetos que tenga en ella.

UC28 – Timeout.

- Descripción: El sistema es el encargado de controlar los temporizadores que contiene el videojuego. Estos temporizadores gestionan diferentes aspectos del juego, tales como la duración de las animaciones de los enemigos, el intervalo entre ataques, el tiempo transcurrido desde que un enemigo muere hasta que se dejan caer sus recompensas, la duración total de una partida, entre otros.

3.1. Análisis energético o de eficiencia algorítmica

Dado que este videojuego se genera proceduralmente a medida que se avanza por los niveles, se ha optado por simplificar y optimizar este aspecto para evitar un uso excesivo de recursos y batería del dispositivo de juego.

La utilización de Godot Engine ha facilitado la implementación de la generación procedural, gracias a una característica que integra el motor. Esta característica permite la generación procedural de los escenarios de juego mediante patrones creados en base a los *tiles* de un *tileset* almacenado en el motor de desarrollo.

Un tileset es un archivo en el que se almacenan todos los componentes que conforman el *tilemap* (Morales Cortés, 2016). Un tileset está conformado por diferentes tiles, siendo cada tile una imagen, en videojuegos 2D normalmente cuadrada o rectangular, que al combinarse permiten crear diferentes representaciones gráficas, como el fondo del castillo, el suelo, la escalera o las diferentes decoraciones encontradas en la pared del castillo. En la Ilustración 14 se muestra un ejemplo de tileset utilizado en la creación de este videojuego.

Ilustración 14. Tileset de algunos de los elementos utilizados en Ascent Tower.

El uso de los patrones definidos en base a los tiles de los tilesets permite reutilizar modelos existentes y ya definidos en lugar de generarlos dinámicamente en tiempo de ejecución mediante bucles, mejorando el rendimiento y la eficiencia del proceso de generación de mapas.

Estos patrones se crean directamente del archivo del tileset, específicamente de los tiles del tileset, almacenándose en memoria y posicionándose en el tilemap del juego según sea necesario. En la Ilustración 15 se muestra un ejemplo de esto, donde abajo se encuentran algunos de los patrones creados para realizar la generación de escenarios de Ascent Tower, arriba a la izquierda diferentes tiles del tileset posicionados en el tilemap y a la derecha los patrones posicionados en el tilemap.

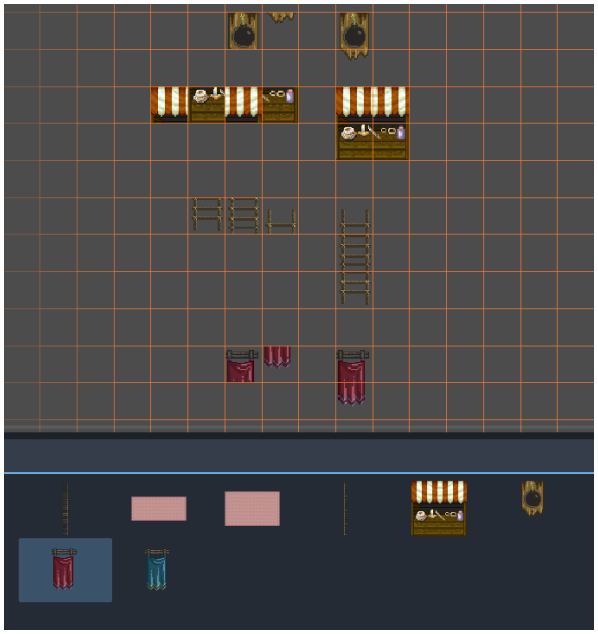

Ilustración 15. Representación de diferentes patrones creados a partir de diferentes tiles del tileset y posicionados en el tilemap.

Un tilemap, representado en la Ilustración 16, es un conjunto de texturas, generalmente de forma cuadrada o hexagonal, que se colocan en una cuadrícula para construir los escenarios de ciertos juegos en 2D (Morales Cortés, 2016).

Ilustración 16. Tilemap de un piso del videojuego Ascent Tower con patrones y componentes de diferentes tilesets.

Godot Engine se encarga de localizar estos objetos prefabricados, los patrones de tiles, y de posicionarlos en las ubicaciones especificadas en el tilemap.

Este proceso es muy similar a la utilización de *prefabs* comúnmente encontrados en los motores de desarrollo de videojuegos.

Un prefab es un recurso que permite guardar un objeto con todos sus componentes y características. Este prefab actúa como una plantilla que permite generar nuevas instancias del objeto en la escena. Lo especial de los prefabs es que cualquier cambio realizado en el recurso del prefab se verá reflejado automáticamente en todas las instancias creadas a partir de él. Sin embargo, también se puede modificar componentes y configuraciones específicas para cada instancia, brindando así un nivel de personalización independiente de la plantilla original (Morales Cortés, 2016).

Sin embargo, a diferencia de un objeto prefab, los patrones del tileset son un componente prefabricado en base a un conjunto de tiles localizados en un tileset almacenado, conformando un único elemento que se puede utilizar en el tilemap del videojuego. Aunque la imagen prefabricada también dispone de diferentes atributos, estos son más limitados en comparación con un prefab, lo que dificulta su interacción con eventos dinámicos durante una partida.

3.2. Identificación y análisis de soluciones posibles

Tras el análisis de los UC y las tareas a realizar, se procede a examinar el contexto más adecuado para la ejecución del videojuego en cuestión. En los siguientes apartados se exponen los puntos más decisivos considerados al realizar este proyecto en cuanto a la temática y jugabilidad del videojuego. Estos apartados incluyen los dispositivos en los que se podrá jugar, la orientación de la pantalla y la disposición de algunos de los elementos del juego en pantalla.

3.2.1. Dispositivos disponibles

Desde el inicio se decidió implementar el videojuego para dispositivos táctiles, por algunas funcionalidades específicas que se deseaba desarrollar, como la disposición de los botones en pantalla y la forma de juego. También se consideró desarrollar el videojuego para dispositivos portátiles que permitieran jugar en cualquier lugar y momento, sin la limitación de tener que estar exclusivamente en casa.

Teniendo en cuenta lo anterior, se identificaron dos tipos de dispositivos que cumplen con estas condiciones: los teléfonos móviles táctiles y las consolas de videojuegos portátiles actuales, como la *Nintendo Switch* o la *Steam Deck*, entre otras.

La Nintendo Switch (ver Ilustración 17) es una consola híbrida lanzada en marzo de 2017 que puede usarse tanto con un televisor como en modo portátil. La consola principal incluye una pantalla LCD multitáctil y puede conectarse a un televisor mediante un soporte. En la versión original, los controles Joy-Con se conectan a ambos lados de la consola, permitiendo diferentes formas de juego. Los Joy-Con pueden estar acoplados a la consola, funcionando como una consola portátil, o desacoplados para su uso en modo televisor o de sobremesa (Pérez Luque & Valverde Bourgon, 2021).

Ilustración 17. Consola de videojuegos Nintendo Switch.

Finalmente, se concluye que el mejor dispositivo para este videojuego son los teléfonos móviles. Actualmente, la mayoría de las personas tienen un teléfono móvil a su alcance en todo momento, a diferencia de las consolas de videojuegos portátiles que, aunque están ganando relevancia en el mercado, no todos tienen una en casa. Esto permitirá que el jugador pueda jugar siempre que quiera y sin limitaciones de dispositivos adicionales.

3.2.2. Orientación del dispositivo móvil

Este videojuego está diseñado para ser jugado exclusivamente en dispositivos móviles. En estos dispositivos, es común que la orientación de la pantalla se pueda limitar a formato vertical u horizontal, e incluso hay aplicaciones donde es el usuario quien decide cómo desea visualizar el contenido.

Tas un estudio del mercado de juegos similares, se ha decidido limitar la orientación de la pantalla únicamente a formato vertical. Esta decisión se debe a que el juego se genera de manera procedural de forma ascendente en relación con la pantalla del dispositivo. Además, esta orientación mejora la experiencia del usuario al optimizar la accesibilidad de los botones en pantalla y la visualización del entorno del jugador.

3.2.3. Puntuaciones

Debido a la modalidad del videojuego, en la que el jugador debe superarse partida a partida, se ha decidido mostrar en pantalla la puntuación del nivel actual junto con la cantidad de oro y diamantes recolectados durante la partida.

Esto mismo se refleja en los juegos de la competencia, como se puede ver en la Ilustración 18, Ilustración 19 e Ilustración 20, donde el usuario puede visualizar constantemente cuántos puntos ha acumulado en función de su progreso en el juego.

Ilustración 18. Puntuación y acumulación de recursos en el videojuego Tower of Hero.

Ilustración 19. Puntuación en partida y récord en el videojuego Super Starfish.

Ilustración 20. Puntuación en partida, récord y acumulación de oro del videojuego Sling Kong.

Asimismo, las puntuaciones en una partida reflejarán únicamente lo acumulado en esa partida, sin considerar las cantidades acumuladas en partidas anteriores. Esto se debe a que el juego tiene dos tiendas: una tienda dentro de la partida, donde se puede comprar utilizando el oro y los diamantes recolectados en la misma partida, y otra tienda fuera de la partida, donde se pueden adquirir artilugios con el oro y los diamantes acumulados a lo largo de todas las partidas. En ambas tiendas se pueden comprar los mismos objetos, pero esta estructura permite que el jugador pueda gastar su economía en la tienda dentro del juego durante una partida, debiendo ser más cuidadoso al elegir qué productos desea comprar.

Por lo tanto, se ha decidido mostrar constantemente las cantidades acumuladas en una partida para que el jugador sea consciente de los recursos disponibles para gastar en la tienda del juego y del nivel en el que se encuentra. Esto le permitirá tratar de superar su récord anterior y, a medida que ascienda por los niveles del juego, obtener mejores y mayores recompensas.

3.3. Solución propuesta

Tras el análisis del problema y la identificación y análisis de soluciones posibles, se ha decidido adoptar un formato de US para la planificación de tareas. Esta metodología permite realizar una estimación de tiempo en horas-persona, asegurando que se aborden cada uno de los UC mencionados anteriormente.

A continuación, se detallan las descripciones, estimaciones de tiempo y tareas de cada US necesarias para cubrir todos los UC, con el objetivo de lograr el desarrollo completo del MVP del videojuego.

Las US que no disponen de tareas son debido a que han surgido de un *brainstorming* y no están dentro del alcance del MVP.

US01 – Mover al jugador en un entorno representativo.

- Descripción: Como usuario quiero poder mover al jugador de lado a lado en los límites permitidos para explorar el escenario del videojuego.
- Estimación: 17 horas.
- Tareas:
    - Movimiento personaje (4 horas).
    - *Sprites* del personaje (3 horas).
    - Animaciones del personaje (6 horas).

US02 – Vida.

- Descripción: Como jugador, quiero que mi personaje tenga un sistema de vidas para proporcionar un desafío adicional en cada partida. Este sistema de vidas aumentará la dificultad y la emoción del juego, ya que tendré que ser cuidadoso para no perder todas mis vidas.
- Estimación: 2 horas y 30 minutos.
- Tareas:
    - Quitar vida si se es golpeado (2 horas).
    - Píxel *art* corazón (30 minutos).

US03 – Escudo.

- Descripción: Como jugador quiero poder tener un escudo para defenderme de los ataques de los enemigos.
- Estimación: 5 horas.
- Tareas:
    - Probabilidad fija de actuar (3 horas).
    - Píxel art del escudo (1 hora y 30 minutos).
    - Efecto de sonido probabilidad acertada (30 minutos).

US04 – Enemigos.

- Descripción: Como jugador quiero que haya enemigos durante mi exploración por la torre para darle más dinamismo y jugabilidad al videojuego.
- Estimación: 28 horas.
- Tareas:
    - Generación de los enemigos (10 horas).
    - Píxel art enemigos (15 horas).

- Máquina de estado enemigos (3 horas).

US05 – Generación de siguiente nivel de piso del castillo.

- Descripción: Como jugador quiero que se genere el siguiente piso de la torre del castillo para poder seguir avanzando en mi aventura y conseguir rescatar a la persona en apuros.
- Estimación: 13 horas.
- Tareas:
  - Generar a los enemigos (5 horas y 30 minutos).
  - Generar el terreno del siguiente piso (6 horas).
  - Píxel art piso torre (1 hora y 30 minutos).

US06 – Pasar siguiente piso (Escalera).

- Descripción: Como jugador necesito que se genere la escalera que enlaza los pisos para poder ascender por la torre y continuar mi recorrido por el castillo.
- Estimación: 21 horas.
- Tareas:
  - Generación de la escalera (5 horas).
  - Indicador de ascenso por la escalera (8 horas).
  - Mover mapa subiendo por la escalera (4 horas).
  - Píxel art escalera (1 hora).
  - Animación subir por la escalera (3 horas).

US07 – Atacar.

- Descripción: Como jugador me gustaría poder atacar a los enemigos para poder acabar con ellos consiguiendo así no morir tan fácilmente a la hora de explorar el castillo.
- Estimación: 6 horas y 30 minutos.
- Tareas:
  - Atacar tras pulsar botón de acción (3 horas).
  - Matar enemigos al atacar (2 horas y 30 minutos).
  - Botón atacar (1 hora).

US08 – Canciones.

- Descripción: Como jugador me gustaría tener canciones acordes a la temática del videojuego para sumergirme más en la experiencia del juego y experimentar las emociones que el videojuego pretende transmitir.
- Estimación: 2 horas.
- Tareas:
  - Arreglar canciones y *Sound Effects* (SFX) (1 hora) – Mario.
  - Preparar música para el menú de configuración (1 hora).

US09 – Efectos de sonido.

- Descripción: Como jugador quiero tener SFX que ambienten las acciones en el videojuego.
- Estimación: 3 horas y 30 minutos.
- Tareas:
  - Elegir nuevos SFX (1 hora y 30 minutos).
  - Introducir nuevos SFX respecto a los eventos correspondientes (2 horas).

US10 – Mejorar enemigos.

- Descripción: Como jugador quiero que los enemigos tengan implementada una Inteligencia Artificial (IA) y sean más dinámicos a la hora de interactuar con el jugador.
- Estimación: 18 horas y 30 minutos.
- Tareas:
    - Añadir AI (6 horas).
    - Vidas (30 minutos).
    - Mostrar barra de vida restante (1 hora).
    - Cambiar el color al ser golpeado (1 hora).
    - Girar enemigo en caso de recibir ataque y no estar viendo al jugador (1 hora).
    - Enemigo ataca a jugador (2 horas).
    - Añadir diferentes enemigos (7 horas).

US11 – Ser golpeado.

- Descripción: Como jugador me gustaría que los enemigos me pudiesen golpear para perder las vidas que tengo y así haya un reto en el videojuego.
- Estimación: 5 horas.
- Tareas:
    - Cambiar el color al ser golpeado (30 minutos).
    - Perder vida al ser golpeado (30 minutos).
    - Fin de partida (4 horas).

US12 – Economía.

- Descripción: Como jugador necesito recoger diamantes y bolsas de oro para poder comprar mejores artefactos y armamento, así como mejorar los que ya tengo en la tienda del juego.
- Estimación: 9 horas.
- Tareas:
    - Contador de oro (2 horas).
    - Dejar caer oro (3 horas).
    - Contador de diamantes (1 hora).
    - Soltar diamantes (3 horas).

US13 – Mejorar jugabilidad.

- Descripción: Como jugador quiero tener un mapa de juego interactivo y extenso, en el que ocurran eventos inesperados para crear un ambiente de jugabilidad más entretenido y diferenciador respecto a otros videojuegos.
- Estimación: 23 horas.
- Tareas:
    - Modificar cámara (2 horas).
    - El jugador podrá moverse libremente (1 hora).
    - Generación de enemigos en zonas no visibles (2 horas).
    - Crear salas (15 horas).
    - No se verá lo que hay en la parte superior del piso (1 hora).
    - El mapa será más grande (2 horas).

US14 – Resumen de partida.

- Descripción: Como jugador deseo poder visualizar un resumen de la partida al perder, incluyendo mis estadísticas de juego, para poder mejorar mi rendimiento y seguir mi progreso de partida en partida.
- Estimación: 6 horas.
- Tareas:
    - Mostrar el tiempo de duración de la partida (30 minutos).
    - Mostrar el oro conseguido (30 minutos).
    - Mostrar los diamantes conseguidos (30 minutos).
    - Mostrar la cantidad de pisos superados (15 minutos).
    - En caso de ser un nuevo récord, se avisará al jugador (45 minutos).
    - Mostrar la cantidad de enemigos derrotados (1 hora).
    - Mostrar el número de ataques realizados (1 hora).
    - Botón Reintentar (15 minutos).
    - Botón Volver (15 minutos).
    - Las estadísticas estarán ocultas (1 hora).

US15 – Ajustes.

- Descripción: Como jugador me gustaría tener la capacidad de ajustar diversos parámetros del juego, como el volumen de la música o la disposición de los botones, para adaptarlo a mis necesidades en cada momento.
- Estimación: 6 horas y 30 minutos.
- Tareas:
    - Añadir ajustes en pantalla de partida (1 hora).
    - Añadir ajustes en menú principal (1 hora).
    - Modificar máster (1 hora).
    - Modificar música (30 minutos).
    - Modificar SFX (30 minutos).
    - Modificar botones de pop-up de acción (2 horas).
    - Modificar botones de posición (modo zurdo) (30 minutos).

US16 – Cofres.

- Descripción: Como jugador quiero encontrarme cofres en las partidas para conseguir mejores recompensas.
- Estimación: 7 horas.
- Tareas:
    - Generación de cofres (4 horas).
    - Máximo un único cofre por piso (2 horas).
    - SFX de abrir cofre (1 hora).

US17 – Pantalla de inicio.

- Descripción: Como jugador me gustaría tener un menú principal antes de comenzar una partida, permitiéndome acceder de manera organizada a las diferentes pantallas del videojuego.
- Estimación: 5 horas y 30 minutos.
- Tareas:
    - Fondo de pantalla del menú principal (1 hora).
    - Botón jugar (1 hora).
    - Botón ajustes (1 hora).

  - Botón salir (2 horas).
  - Añadir canción de menú principal (30 minutos).

US18 – Créditos.

- Descripción: Como jugador me gustaría poder observar los créditos de los creadores del videojuego para informarme sobre qué personas lo han desarrollado.
- Estimación: 1 hora y 30 minutos.
- Tareas:
  - Crear pantalla de créditos (45 minutos).
  - Botón volver (45 minutos).

US19 – Tienda.

- Descripción: Como jugador deseo tener acceso a una tienda dentro del juego donde pueda gastar el oro y los diamantes obtenidos durante la partida para adquirir diferentes artículos, como pueden ser pociones, y darle mayor jugabilidad a las partidas.
- Estimación: 19 horas y 30 minutos.
- Tareas:
  - Píxel art pociones (1 hora).
  - Botón volver (30 minutos).
  - Botón ir a la tienda (1 hora).
  - Creación de la escena de la tienda (4 horas).
  - Pagar con oro (2 horas).
  - Pagar con diamantes (2 horas).
  - Pociones (9 horas).

US20 – Mochila.

- Descripción: Como jugador desearía contar con una mochila en la que pueda almacenar todos los artículos comprados en la tienda del juego, facilitando así la administración de mis recursos.
- Estimación: 8 horas.
- Tareas:
  - Creación de la escena de la mochila (3 horas).
  - Botones de acceso rápido (2 horas).
  - Formato de la bolsa en forma de rejilla (3 horas).

A partir de este punto, en las US ya no se definen tareas, ya que las siguientes no forman parte del MVP del proyecto. No obstante, estas tareas se incluyen en el backlog para ser consideradas en futuros sprints. El enfoque para abordar estas US se detalla en el apartado “Trabajos futuros”.

US21 – Armadura del jugador.

- Descripción: Como jugador quiero tener una armadura para poder tener un medio de defensa ante los ataques de los enemigos.
- Estimación: 8 horas.

US22 – Dejar caer artefactos.

- Descripción: Como jugador necesito que los enemigos suelten una variedad de artefactos, como armaduras, armas, pociones y amuletos, para mejorar mi rendimiento y facilitar mi progreso a través de los niveles de la torre del castillo.
- Estimación: 10 horas.

US23 – Puerta a sala interna al castillo.

- Descripción: Como jugador quiero poder acceder a una habitación interna del castillo para explorarlo en detalle. Dado que esta habitación tiene un acceso más difícil, preferiría que no hubiera enemigos y que fuera un espacio más tranquilo en comparación con las salas comunes encontradas en el castillo.
- Estimación: 15 horas y 30 minutos.

US24 – Pantallas de carga.

- Descripción: Como jugador me gustaría que se mostrase una pantalla de carga en momentos donde, una vez el juego ya esté iniciado, se tenga que cargar nuevo contenido.
- Estimación: 6 horas.

US25 – Pantalla de carga de inicio.

- Descripción: Como jugador me gustaría que hubiese una pantalla de carga al iniciar el juego para ver el estado de carga de este.
- Estimación: 4 horas.

US26 – Logros.

- Descripción: Como jugador desearía tener diferentes logros que cumplir para tener retos que superar a cambio de recompensas.
- Estimación: 15 horas.

US27 – Ventana.

- Descripción: Como jugador me gustaría que el juego tuviese ventanas por las que pueda ver el temporal detrás del castillo y además pueda caerme por ellas para dar más dinamismo al juego.
- Estimación: 7 horas y 30 minutos.

US28 – Carga asíncrona de las pantallas.

- Descripción: Como jugador quiero que la carga entre las pantallas se realice de forma asíncrona para así poder visualizar el avance de carga y que el juego no parezca que se queda parado.
- Estimación: 2 horas.

US29 – *Ranking* global.

- Descripción: Como jugador me gustaría disponer de un ranking global en el que compararme con jugadores de todo el mundo para ver cuál es el número de pisos más alto alcanzado y quién ha sido el que lo ha conseguido.
- Estimación: 20 horas.

US30 – Ranking amigos.

- Descripción: Como jugador quiero tener un ranking donde pueda ver los récords de mis amigos para poder comparar sus puntuaciones máximas con las mías.
- Estimación: 35 horas.

US31 – Idioma.

- Descripción: Como jugador quiero tener diferentes idiomas en el videojuego para poder elegir el idioma que prefiera.
- Estimación: 9 horas.

US32 – Cambio de personaje y artefactos.

- Descripción: Como jugador me gustaría poder cambiar al personaje principal por otro y los artefactos equipados para no tener siempre al mismo jugador y poder disfrutar de otro tipo de personajes.
- Estimación: 30 horas.

US33 – Modalidad de juego: Arquero.

- Descripción: Como jugador desearía tener otro tipo de modalidad de juego a parte del cuerpo a cuerpo con espada para añadir más funcionalidad al videojuego y que se pueda atacar a distancia a los enemigos.
- Estimación: 25 horas.

US34 – Boss.

- Descripción: Como jugador me gustaría tener que luchar contra mejores y más difíciles enemigos que sirvan como puntos de guardado para poder continuar mi aventura desde ese piso, si es que se consigue derrotar al enemigo en cuestión, y no tener que empezar siempre desde el piso más bajo.
- Estimación: 50 horas.

US35 – Temporal.

- Descripción: Como jugador me gustaría que el juego, por detrás de la escena del castillo, mostrase el temporal que hace fuera de este y con ello el escenario donde me encuentro se adapte a los diferentes temporales para darle más realismo y nuevas mecánicas al videojuego.
- Estimación: 60 horas.

US36 – Reto semanal.

- Descripción: Como jugador desearía tener retos semanales donde competir con otros jugadores a nivel mundial para superar mi técnica en el videojuego, mejorar como jugador y tener un interés extra de entrar al juego semanalmente.
- Estimación: 40 horas.

US37 – Animación inicio del juego.

- Descripción: Como jugador me gustaría tener una animación la primera vez que inicio el videojuego para poder saber la historia del juego, de qué trata y cuáles son los objetivos a cumplir.
- Estimación: 25 horas.

US38 – Animación rescate persona en peligro.

- Descripción: Como jugador me gustaría que, si rescato a la persona que se encuentra en peligro, el objetivo del juego, se reproduzca una animación en la que salga la persona siendo rescatado por mi héroe para dar por finalizada la trama principal del videojuego.
- Estimación: 30 horas.

US39 – Vincular cuenta *Play* Games.

- Descripción: Como jugador desearía poder vincular mi cuenta de Play Games para guardar mi progreso en el juego y así comparar mis puntuaciones y logros con los de mis amigos de forma más fácil.
- Estimación: 7 horas.

US40 – Anuncios *AdMob*.

- Descripción: Como jugador me gustaría poder visualizar anuncios en el juego para apoyar al creador del videojuego dado que se trata de un videojuego gratuito.
- Estimación: 15 horas.

US41 – Guerrero acompañante.

- Descripción: Como jugador quiero poder rescatar guerreros de la torre y que estos se unan a mi equipo para tener una forma de ataque pasiva y darle más realismo al videojuego.
- Estimación: 20 horas.

US42 – Modalidad de juego: Mago.

- Descripción: Como jugador quiero tener una modalidad de juego de mago donde pueda atacar a distancia usando hechizos para añadir nuevas formas de jugar al juego.
- Estimación: 20 horas.

US43 – Elegir conjunto aleatorio.

- Descripción: Como jugador me gustaría que el videojuego dispusiera de una opción para poder asignar armaduras aleatorias al personaje para así no tener que cambiársela manualmente cada vez que quiera modificarla.
- Estimación: 12 horas.

US44 – Expediciones de personajes.

- Descripción: Como jugador me gustaría que el videojuego tuviese expediciones donde mandar a los personajes rescatados de la torre del castillo para poder conseguir recompensas de forma pasiva.
- Estimación: 15 horas.

US45 – Eventos.

- Descripción: Como jugador quiero tener diferentes eventos, como puede ser de navidad, *Halloween*, pascua, aniversario, entre otros, que tematicen el videojuego.

- Estimación: 50 horas.

US46 – Puesta en marcha.

- Descripción: Como jugador quiero que se realice la puesta en marcha del videojuego para poder descargarlo desde una página web de videojuegos y jugarlo.
- Estimación: 11 horas.
- Tareas:
    - Arreglo de fallos conocidos y mejoras (8 horas).
    - Sacar el videojuego en itch.io (3 horas).

US47 – Cambios o mejoras de la encuesta.

- Descripción: Como jugador quiero que se integren los cambios que los usuarios han propuesto a la hora de realizar la encuesta para mejorar el videojuego con diferentes opiniones de jugadores reales.
- Estimación: 22 horas.
- Tareas:
    - Añadir nuevas pociones (5 horas).
    - Modificar interfaz del menú principal respecto al botón de la tienda (2 horas).
    - Modificar icono de la tienda (30 minutos).
    - Modificar botones de acción por defecto (1 hora y 30 minutos).
    - Apertura de cofres más vistosa (4 horas).
    - Tienda en partida (8 horas).
    - Reducir el rango de ataque (1 hora).

US48 – Funcionalidad de salto.

- Descripción: Como jugador quiero poder saltar en una partida para poder esquivar pinchos y diferentes obstáculos, dándole así mayor jugabilidad al personaje.
- Estimación: 8 horas.

3.4. Plan de trabajo

Con las US a realizar ya definidas, es momento de planificar el trabajo para realizar el MVP completo.

La planificación se ha estructurado en sprints de 2 semanas cada uno, con un total de 8 sprints desde el inicial, conocido como sprint 0, hasta el sprint 7, que representa la fase final para alcanzar el MVP. Además, se ha gestionado un backlog donde se han registrado todas las US que están fuera del alcance del MVP. No obstante, se considera interesante considerar estas US para su implementación en futuros sprints, con el objetivo de continuar el desarrollo del proyecto, expandir sus funcionalidades y ofrecer nuevas experiencias a los usuarios a través de actualizaciones del videojuego existente.

Para la gestión de los sprints se ha utilizado la herramienta HacknPlan, que facilita la planificación como se detalla en el apartado “Metodología” del documento.

La distribución de sprints y backlog que se ha seguido en este proyecto, así como las fechas de inicio y finalización de cada sprint, ha sido la siguiente:

Sprint 0 (12/03/2024 – 26/03/2024):

- Aprendizaje de HacknPlan (3 horas y 30 minutos).
- Versión preliminar GDD (8 horas).
- Aprendizaje de Godot (12 horas).
- US01 – Mover al jugador en un entorno representativo.

Sprint 1 (26/03/2024 – 09/04/2024):

- US02 – Vida.
- US03 – Escudo.
- US04 – Enemigos.
- Actualizar GDD (3 horas y 30 minutos).

Sprint 2 (09/04/2024 – 23/04/2024):

- US05 – Generación de siguiente nivel de piso del castillo.
- US06 – Pasar siguiente piso (Escalera).
- Actualizar GDD (2 horas y 30 minutos).

Sprint 3 (23/04/2024 – 07/05/2024):

- US07 – Atacar.
- US08 – Canciones.
- US09 – Efectos de sonido.
- Acabar GDD (1 hora y 30 minutos).

Sprint 4 (07/05/2024 – 21/05/2024):

- US10 – Mejorar enemigos.
- US11 – Ser golpeado.
- US12 – Economía.

Sprint 5 (21/05/2024 – 04/06/2024):

- US13 – Mejorar jugabilidad.
- US14 – Resumen de partida.
- US15 – Ajustes.
- US16 – Cofres.

Sprint 6 (04/06/2024 – 18/06/2024):

- Fleco: US13 – Mejorar jugabilidad (Tarea: Crear salas) (2 horas).
- Fleco: US15 – Ajustes (Tarea: Modificar botones de pop-up de acción) (30 minutos).
- Fleco: US15 – Ajustes (Tarea: Modificar botones de posición (modo zurdo)) (30 minutos).
- US17 – Pantalla de inicio.
- US18 – Créditos.
- US19 – Tienda.
- US20 – Mochila.

Sprint 7 (18/06/2024 – 28/06/2024):

- US46 – Puesta en marcha.
- US47 – Cambios o mejoras de la encuesta.

Backlog (Ordenado por prioridad):

- US21 – Armadura del jugador.
- US22 – Dejar caer artefactos.
- US23 – Puerta a sala interna al castillo.
- US24 – Pantallas de carga.
- US25 – Pantalla de carga de inicio.
- US26 – Logros.
- US27 – Ventana.
- US28 – Carga asíncrona de las pantallas.
- US29 – Ranking global.
- US30 – Ranking amigos.
- US31 – Idioma.
- US32 – Cambio de personaje y artefactos.
- US48 – Funcionalidad de salto.
- US33 – Modalidad de juego: Arquero.
- US34 – Boss.
- US35 – Temporal.
- US36 – Reto semanal.
- US37 – Animación inicio del juego.
- US38 – Animación rescate persona en peligro.
- US39 – Vincular cuenta Play Games.
- US40 – Anuncios AdMob.
- US41 – Guerrero acompañante.
- US42 – Modalidad de juego: Mago.
- US43 – Elegir conjunto aleatorio.
- US44 – Expediciones de personajes.
- US45 – Eventos.

4. Diseño de la solución

En los siguientes apartados se muestra la estructura que se ha llevado en este proyecto en cuanto a capas lógicas de software y la distribución de las clases en cada capa.

4.1. Arquitectura del Sistema

No se ha integrado una base de datos con conexión a internet para el almacenamiento del progreso y las puntuaciones, ya que el videojuego está diseñado para ser jugado de manera offline. Por este motivo, la arquitectura de la aplicación se ha diseñado en un formato de dos capas: la capa de presentación y la capa de lógica de negocio.

En la imagen Ilustración 21 se muestra la distribución por capas y la interacción entre ellas, tal como se ha definido en este proyecto:

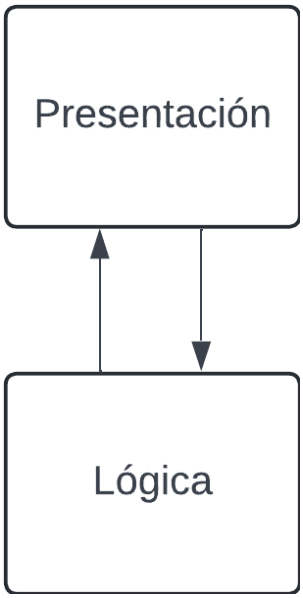

Ilustración 21. Distribución e interacción entre las capas del videojuego Ascent Tower.

Las distintas capas del sistema se han desarrollado como módulos independientes en proyectos separados, todos ellos integrados dentro de la misma solución. Esta arquitectura modular ha permitido una mayor modularidad y mantenibilidad del código. Además, esta estructura facilita la escalabilidad del sistema, ya que cada capa puede ser actualizada o reemplazada de forma independiente sin afectar al resto del sistema.

4.1.1. Capa de presentación

La capa de presentación constituye el componente del sistema que se encarga de la representación visual de la información y los elementos interactivos para el usuario. Esta capa abarca todos los elementos con los que interactúa el jugador en la pantalla del juego.

Además, esta capa se conecta con la capa lógica para obtener la información a mostrar por pantalla, como la puntuación actual o el número de vidas restantes.

4.1.2. Capa de lógica de negocio

La capa lógica de un videojuego es el componente encargado de gestionar la funcionalidad del juego. Esta capa incluye todas las reglas, cálculos, procesos y algoritmos que determinan cómo se desarrolla el juego. Mientras que la capa de presentación se ocupa de lo que el jugador ve y con lo que interactúa, la capa lógica se centra en cómo esas interacciones y elementos del juego se comportan y responden.

Dentro de esta capa, están métodos diseñados para realizar los cálculos requeridos cuando la capa de presentación lo solicite.

4.2. Diseño Detallado

Cada capa está compuesta por varias clases. En la capa de presentación estas clases estarán relacionadas con las escenas propias de Godot.

En Godot un juego se puede estructurar en base a escenas, que podrían representar, por ejemplo, un nivel específico dentro del juego. Dentro de una escena, se encuentran diversos elementos que la conforman, como objetos interactuables, componentes del nivel y el propio personaje, entre otros. Cada escena está compuesta por un conjunto de nodos, que representan los elementos de la escena, y scripts, que son código asociado a ciertos nodos que determinan su comportamiento. En este contexto, cada escena tiene un nodo principal, del cual dependen todos los demás, y que podríamos considerar como el nodo padre (Culiáñez Llorca, 2023).

En este proyecto se ha decidido estructurar cada componente independiente en escenas. Por ejemplo, el jugador, que el usuario puede mover y con el que puede atacar, constituye una escena. Los enemigos conforman otra escena que se reutiliza para cada enemigo que aparece. Los cofres, en los que puede haber oro y diamantes, representan una escena adicional. El oro y los diamantes que pueden encontrarse en los cofres o que pueden ser obtenidos de los enemigos también se consideran escenas individuales, entre otros.

A continuación, en la Ilustración 22, se presenta el diagrama de clases que ilustra los componentes del videojuego y sus relaciones:

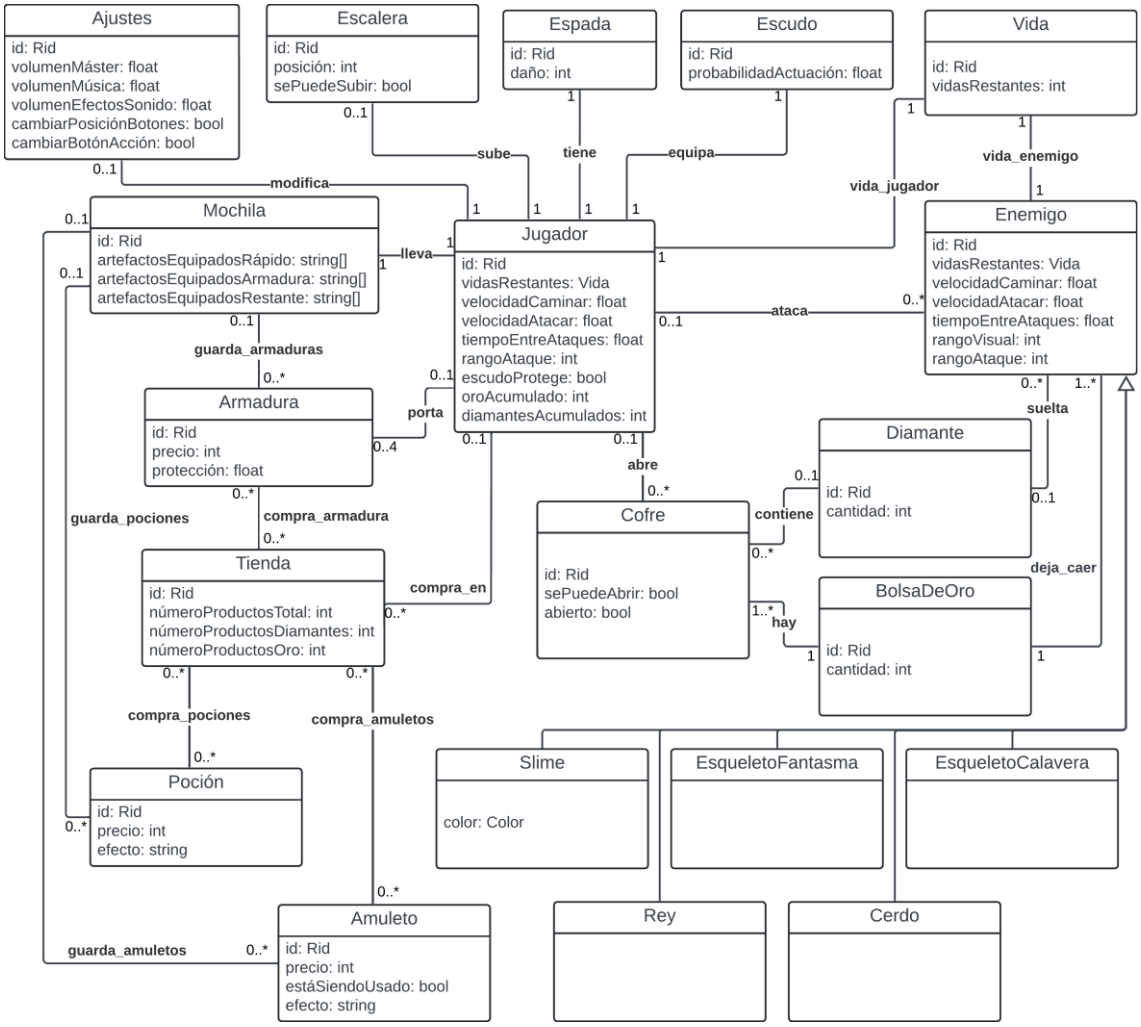

Ilustración 22. Diagrama de clases del videojuego Ascent Tower.

Cada clase creada en la capa de presentación, con su correspondiente escena, tiene una clase asociada en la capa lógica que realiza todos los cálculos necesarios, si es requerido. La comunicación entre capas se realiza a través de la clase que contiene la escena de Godot y una interfaz con los métodos públicos usables de la capa de lógica, que la capa de presentación utiliza para mostrar los diferentes resultados por pantalla.

Por ejemplo, la escena Jugador está asociada a la clase Jugador en la capa de presentación. Esta clase realiza llamadas a la capa de lógica de negocio, que contiene la clase Jugador donde se realizan los cálculos necesarios para la capa de presentación. Esta comunicación se efectúa mediante una interfaz intermedia que expone todos los métodos utilizables por la capa de presentación.

En el diagrama de la Ilustración 23, se muestra cómo la capa de presentación y la capa lógica se comunican a través de la interfaz IJugador. La flecha punteada indica que la clase Jugador en la capa de presentación depende de la clase Jugador en la capa lógica, pero esta dependencia es indirecta y está mediada por la interfaz IJugador. La flecha con línea discontinua entre la interfaz IJugador y la clase Jugador en la capa lógica representa las llamadas a métodos que se realizan dentro de la capa lógica para llevar a cabo los cálculos necesarios.

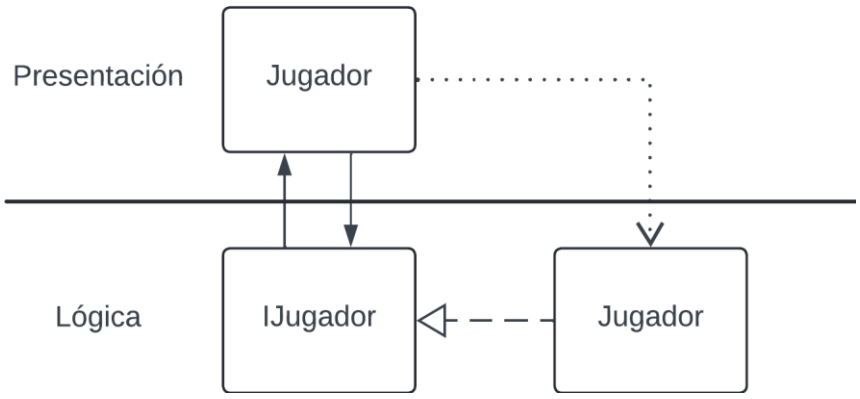

Ilustración 23. Comunicación entre capas de la clase Jugador.

También se ha diseñado una estructura en la que cada valor numérico de importancia se almacena en una clase enumerada ubicada en la capa de lógica. En este enumerado se encuentra toda la información relevante para realizar ajustes en el videojuego, como la vida máxima del jugador, el número mínimo y máximo de enemigos, la cantidad de oro que se puede obtener (mínima y máxima) y la cantidad de oro que se puede perder al perder una vida, entre otros muchos ajustes.

Para mejorar la legibilidad, se ha optado por organizar cada grupo relacionado en categorías específicas por regiones. Por ejemplo, el oro y los diamantes se encuentran en la región económica, los enemigos tienen su propia región exclusiva, al igual que los jugadores, y así sucesivamente.

Se ha decidido implementar esta estructura para unificar todos los valores que afectan la jugabilidad del videojuego en un solo lugar. De esta manera, es posible reajustar el videojuego sin necesidad de modificar varias clases del proyecto; solo sería necesario modificar una clase, lo que agiliza cualquier cambio. Este enfoque no solo mejora la eficiencia en el desarrollo, sino que también desempeña un papel fundamental en el mantenimiento del juego. Al centralizar todos los valores de ajuste en una sola clase, se simplifica enormemente la tarea de realizar cambios o actualizaciones en la jugabilidad del juego.

A continuación, se muestra la Ilustración 24 a modo de ejemplo de comunicación entre la capa de presentación accediendo al enumerado que se encuentra en la capa lógica.

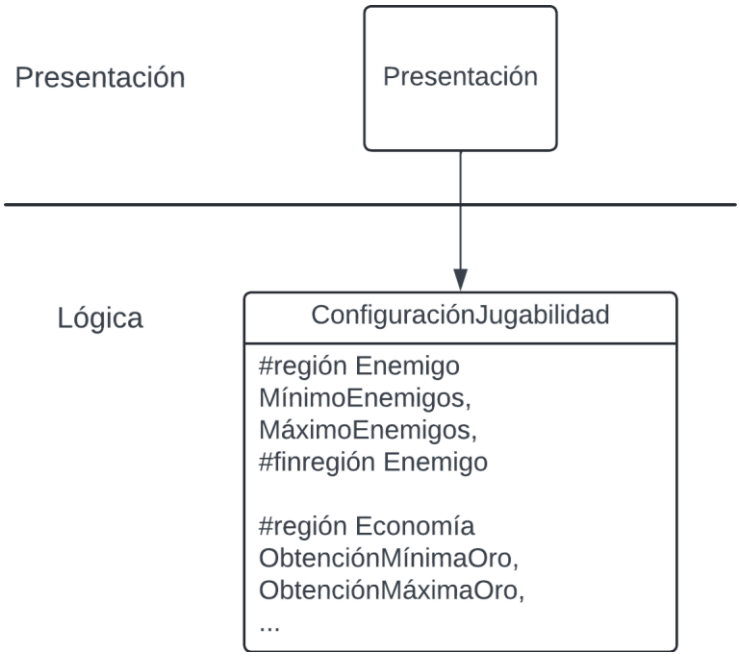

Ilustración 24. Comunicación entre la capa de presentación y la clase enumerada en la capa lógica.

Finalmente se ha hecho uso de patrones de diseño para mejorar la estructura, la flexibilidad y la reutilización del código. Los patrones utilizados han sido el *singleton* y el observador.

El patrón de diseño singleton, representado en la Ilustración 25, asegura que una clase tenga solo una instancia, y proporciona un punto de acceso global a ella (Gamma, Helm, Johnson, Vlissides, & Patterns, 1995).

Este patrón ha sido muy útil para gestionar los ajustes del usuario desde el menú de configuración, los cuales se guardan localmente en el dispositivo del jugador. Además, se ha utilizado para guardar la puntuación máxima del jugador, permitiendo informarle si ha superado su récord anterior. Dado que se puede acceder y modificar los ajustes desde varias escenas, se decidió utilizar el singleton para mantener una instancia global y consistente.

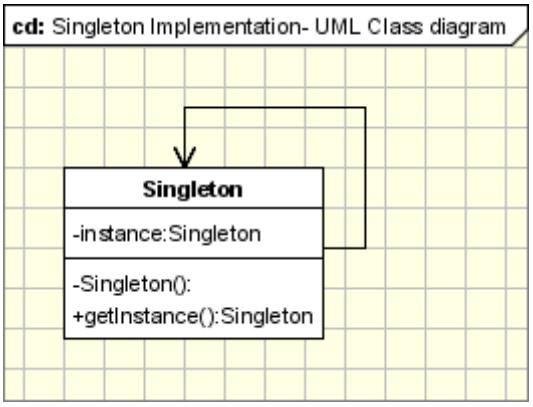

Ilustración 25. Ejemplo de estructura del patrón de diseño singleton.

Por otro lado, el patrón de diseño observador, representado en la Ilustración 26, define una dependencia uno a muchos entre objetos, de modo que cuando un objeto cambie de estado, todos sus dependientes sean notificados y actualizados automáticamente (Gamma, Helm, Johnson, Vlissides, & Patterns, 1995).

Este patrón ha sido de mucha utilidad para gestionar eventos en el juego, dado que, tras su implementación, se dispone de una forma muy sencilla de notificar a otros componentes del juego sobre la ocurrencia de eventos. Este patrón se ha utilizado en la mayoría de los eventos del videojuego, tales como la pérdida de todas las vidas del jugador, lo que activa la pantalla de fin de partida, y la recogida de objetos valiosos como bolsas de oro o diamantes, que se muestran en las secciones correspondientes de puntuación, entre otros.

Godot proporciona una base sencilla para la implementación del patrón observador. Dado que cada evento en el juego, como la pulsación de un botón, debe ser controlado mediante este patrón, su uso se vuelve esencial. Sin embargo, la funcionalidad ofrecida por el editor de Godot es limitada. Por ello, se ha optado por implementar el patrón observador directamente en el código, lo que ofrece más opciones y accesibilidad para notificar a otras clases sobre los eventos ocurridos en el juego.

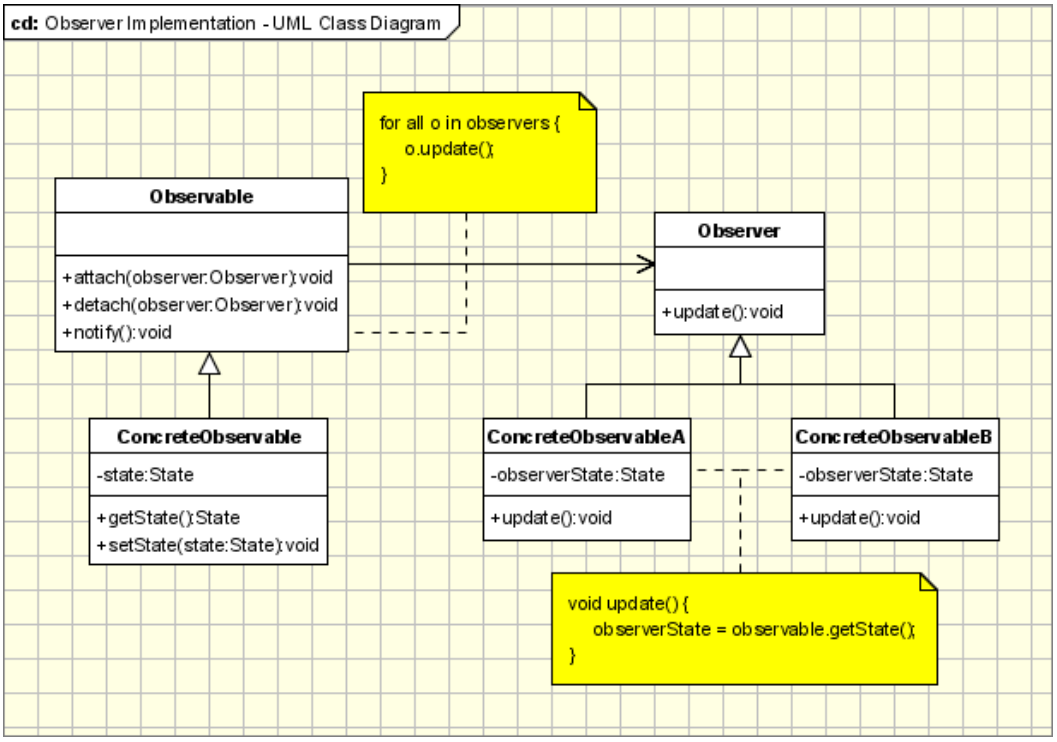

Ilustración 26. Ejemplo de estructura del patrón de diseño Observador.

4.3. Tecnología Utilizada

Para la realización de este proyecto se han empleado diversas tecnologías en el diseño, la creación de la música, la programación del videojuego, el control de versiones y la gestión del tiempo y las tareas asociadas a cada entregable.

4.3.1. Diseños

Para los diseños se han empleado diferentes tecnologías según las necesidades específicas del desarrollo del proyecto. Entre ellas se encuentran *Lucidchart*, *Figma* e *Inkscape*.

Lucidchart (Lucid, 2024), cuyo logotipo se representa en la Ilustración 27, es una plataforma de diagramación en línea que permite a los usuarios crear y colaborar en una amplia variedad de diagramas y visualizaciones (Lucid, 2024).

La herramienta ha sido utilizada para crear diagramas, como el diagrama de clases o los diagramas de interacción entre las distintas capas que componen el videojuego. Estos diagramas facilitan la comprensión del sistema que se pretende desarrollar, sirviendo como una guía para mantener los objetivos iniciales del videojuego en mente y asegurando así la coherencia en su desarrollo.

Ilustración 27. Logotipo de la herramienta Lucidchart.

En la Ilustración 28 se encuentra el logotipo de la herramienta Figma. Figma es una herramienta de diseño colaborativo basada en la nube que se utiliza para la creación de interfaces de usuario, prototipos y gráficos. Proporciona funcionalidades para el diseño de interfaces y experiencias de usuario, la creación de prototipos interactivos, el uso de componentes reutilizables y la integración de comentarios en tiempo real (Muñoz Jiménez, 2024).

En este proyecto, se ha utilizado la herramienta Figma (Figma, 2024) para la creación de bocetos y mapas conceptuales, como el mapa de navegación entre pantallas incluido en el GDD. Figma ha sido elegida por su facilidad de uso, lo que permite la realización de prototipos rápidos, facilitando así la clarificación de ideas y la planificación del proyecto.

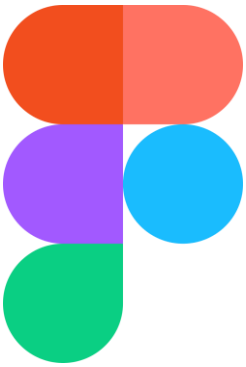

Ilustración 28. Logotipo de la herramienta Figma.

Inkscape (Inkscape Draw Freely, 2024) (ver logotipo en la Ilustración 29) es un editor de gráficos vectoriales libre y de código abierto. Este programa, que se caracteriza por su ligereza y gratuidad, está en constante desarrollo y es utilizado a nivel mundial. Tanto diseñadores profesionales como aficionados recurren a él para crear una amplia variedad de gráficos, utilizando principalmente el formato SVG (Muñoz Jiménez, 2024).

Se ha optado por emplear esta herramienta para diseñar algunos de los elementos específicos del videojuego. Entre estos elementos se incluyen los botones de acción, como los de movimiento y ataque, así como el pop-up de acción que se despliega en escaleras, cofres y tiendas.

Ilustración 29. Logotipo de la herramienta Inkscape.

4.3.2. Música

Para la creación de las diferentes canciones que componen el videojuego, se ha utilizado el programa *MuseScore* (MuseScore, 2024) (ver Ilustración 30). Mario Mocholí ha empleado este software para crear composiciones con diversos instrumentos, sin necesidad de disponer físicamente de ellos.

MuseScore permite la reproducción de composiciones musicales sin requerir el uso de instrumentos físicos, ofrece la opción de seleccionar diferentes instrumentos, imprimir partituras y almacenarlas en diversos formatos (Vázquez-Sánchez & Chao-Fernández, 2019).

Ilustración 30. Logotipo de la herramienta Musescore.

4.3.3. Motor de videojuegos

Como motor, y tras el análisis realizado en la sección “Análisis de motores de videojuegos”, se ha escogido Godot Engine (Godot Engine, 2024) debido a que se pretende desarrollar un juego en 2D con requisitos gráficos moderados, y por su facilidad de uso. Su logotipo se muestra en la Ilustración 31.

Ilustración 31. Logotipo del motor de videojuegos Godot Engine.

4.3.4. Programación

En cuanto a la elección del entorno de desarrollo, se optó por utilizar *Visual Studio* (Microsoft, 2024) (ver Ilustración 32) por la facilidad de integración en Godot y por la comodidad de uso e integración de nuevas características, como las librerías de clases, que permiten la separación del código en diferentes capas lógicas.

Además, se seleccionó el lenguaje C# (ver Ilustración 33) por su versatilidad como lenguaje orientado a objetos. Esta decisión fue posible gracias a la Interfaz de Programación de Aplicaciones (*Application Programming Interface*, API) específica que Godot ofrece para C#, la cual presenta un rendimiento muy similar al del lenguaje por defecto GDScript proporcionado por Godot. En la documentación de Godot se puede encontrar todas las diferencias respecto a la API en cuanto a los nombres de métodos, atributos y clases entre el lenguaje GDScript y C#, facilitando así la implementación en ambos lenguajes de programación.

Visual Studio, desarrollado por *Microsoft*, es un entorno integrado de desarrollo empleado para crear una amplia gama de aplicaciones de software, como aplicaciones web, móviles, de escritorio y otras plataformas. Es reconocido como una de las suites de desarrollo más exhaustivas y populares en los ámbitos de la programación y la ingeniería de software (Muñoz Jiménez, 2024).

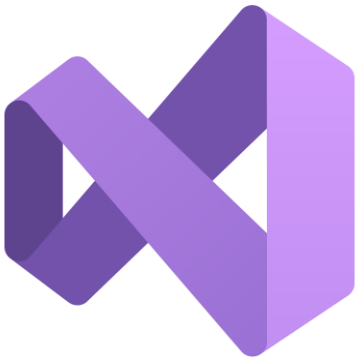

Ilustración 32. Logotipo de la herramienta Visual Studio.

C# (Microsoft, 2024) es un lenguaje de programación orientado a objetos creado por Microsoft. Originalmente, se diseñó para el desarrollo de aplicaciones en el entorno Windows, aunque también se utiliza para crear aplicaciones web, móviles y de consola. Como lenguaje de alto nivel, C# resulta accesible y fácil de aprender, particularmente para quienes ya tienen experiencia en programación. A pesar de su similitud con otros lenguajes como *Java*, C++ y Visual *Basic*, C# posee características distintivas que lo hacen especialmente apropiado para el desarrollo de aplicaciones en Windows y para la web (Wagner, 2024).

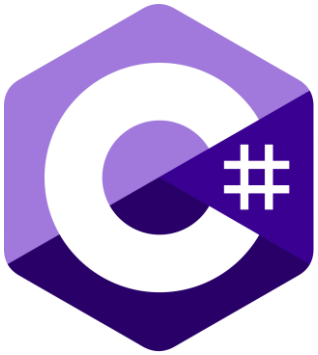

Ilustración 33. Logotipo del lenguaje de programación C#.

4.3.5. Control de versiones

Para gestionar el control de versiones de este proyecto, se ha optado por utilizar GitHub, cuyo logotipo se representa en la Ilustración 34, debido a su facilidad de uso. Esta elección se basa en la capacidad de GitHub para proporcionar una plataforma intuitiva y eficiente, lo que permite establecer un control de versiones de manera efectiva sin invertir un tiempo considerable en su configuración. Esto a su vez, nos permite dedicar más recursos al desarrollo del proyecto.

Github (GitHub, 2024) es un servicio en línea que utiliza un sistema de control de versiones llamado Git (Git, 2024) (ver logotipo en la Ilustración 35), que facilita la gestión

de proyectos de software. Permite la creación de ramas de trabajo y el acceso al proyecto desde múltiples dispositivos de manera sencilla (Muñoz Jiménez, 2024).

Ilustración 34. Logotipo de la herramienta GitHub.

Ilustración 35. Logotipo de Git.

4.3.6. Gestión del tiempo y tareas

Como herramienta para la gestión del tiempo y organización de tareas se ha usado HacknPlan (ver Ilustración 36), debido a su gran facilidad de gestión especializada en proyectos de videojuegos en formato de desarrollo ágil, su interfaz intuitiva y facilidad de uso.

HacknPlan (HacknPlan, 2024) es una plataforma web diseñada para gestionar y planificar las tareas necesarias en el desarrollo de un videojuego. Las tareas se organizan en cuatro fases: Planeada, En progreso, Probando y Completada, y se pueden clasificar en diversas categorías como Programación, Arte, Diseño, Escritura, Marketing, Sonido, Ideas o *Bug*. Además, es posible asignar un tiempo específico a cada tarea para registrar las horas dedicadas al proyecto. HacknPlan es ideal para el desarrollo de videojuegos, ya que contempla numerosos aspectos útiles para los desarrolladores, como la integración con cuentas de GitHub y los repositorios del proyecto (Pousa Barros, 2022).

Ilustración 36. Logotipo de la herramienta de gestión ágil de proyectos de videojuegos HacknPlan.

5. Desarrollo de la solución propuesta

Dado que en este proyecto se ha seguido un desarrollo ágil, iterativo e incremental, utilizando sprints de una duración de dos semanas, el proyecto ha evolucionado progresivamente con cada entrega. A continuación, se describen, iteración por iteración, los cambios que se han ido incorporando al juego a medida que se completaban los sprints.

5.1. Evolución del proyecto por sprints

5.1.1. Sprint 0

Este es el primer sprint del proyecto. Dado que se va a hacer uso de una tecnología en la que no se tenía experiencia, este sprint ha sido útil como toma de contacto con las nuevas herramientas y para un estudio más profundo de las ya utilizadas. Este sprint ha tenido el menor desarrollo debido al aprendizaje de las herramientas escogidas, con una única US a realizar, siendo esta la US01.

Durante este sprint se definieron tareas de aprendizaje de HacknPlan y Godot Engine, las cuales ocuparon aproximadamente la mitad del tiempo del sprint. Se siguieron los tutoriales definidos en la guía de Godot Engine, que muestran paso a paso la creación de un videojuego 2D. También se profundizó en el uso de HacknPlan, una herramienta muy intuitiva, utilizando vídeos informativos para obtener más información.

Además, se elaboró una versión preliminar del GDD, documento vital para definir los objetivos iniciales del videojuego, su contenido y lo que se espera obtener con el producto final. Gracias a este documento, se pueden evitar desviaciones de la esencia del juego respecto a nuevas ideas y proporciona una base para la creación de las US correspondientes.

En este sprint también se definieron la mayoría de las US, con sus estimaciones de tiempo, que finalmente se han llevado a cabo en el alcance del MVP tras la escritura inicial del GDD. Las estimaciones iniciales permitieron calcular el alcance de los sprints y definir el posible MVP del videojuego.

Finalmente, se completó la US01, que consistía en implementar un personaje jugable, como se observa en la Ilustración 38, con movimiento lateral y una animación de caminar (ver Ilustración 39). Gracias a Godot, esta implementación se simplificó mediante la configuración de *input maps* como se muestra en la Ilustración 37. Dado que el juego se lanzará para plataformas *Android*, se consideraron jugadores que prefieren conectar un mando de videojuegos al dispositivo o jugar con emuladores desde un ordenador.

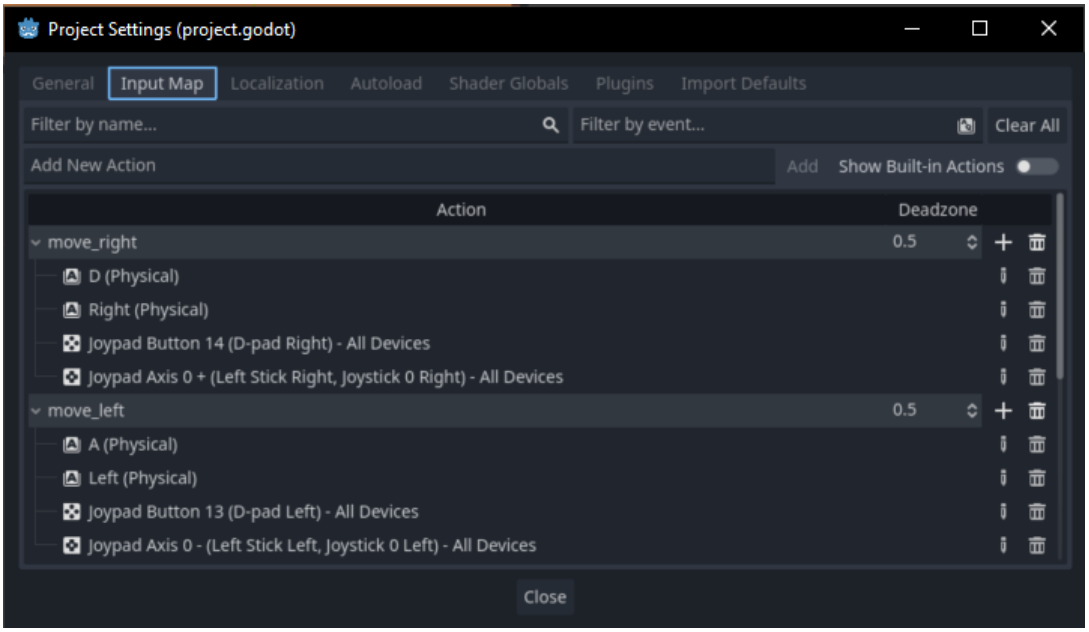

Ilustración 37. Inputs maps definidos para los diferentes dispositivos con el editor de Godot Engine.

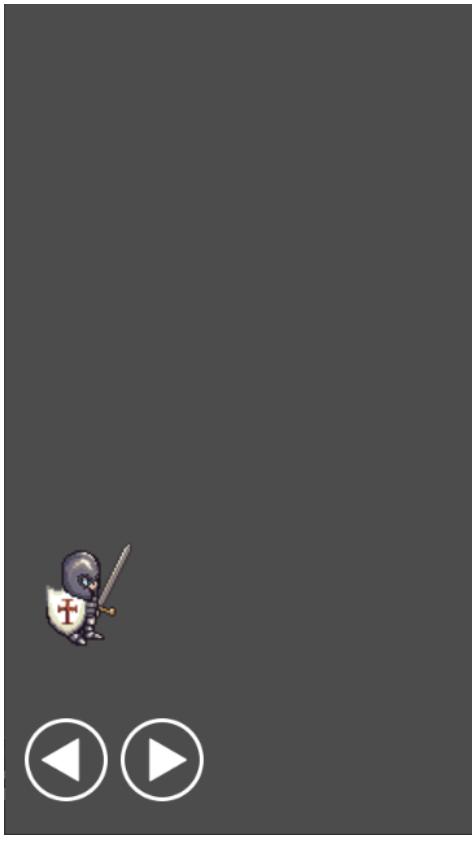

Ilustración 38. Avance del sprint 0 del personaje en estado *idle* (sin animación de movimiento activa).

Ilustración 39. Avance del sprint 0 del personaje moviéndose a la izquierda.

5.1.2. Sprint 1

En este segundo sprint se adquirió un mayor conocimiento del funcionamiento de Godot. Se completaron tres US, siendo estas las US02, US03 y US04. La más costosa fue la US04, que involucró la adición de enemigos al juego.

Tras completar estas US, el jugador podía perder vidas al chocar con un enemigo, a menos que el escudo, implementado en este incremento, lo protegiera, de forma pasiva, con una probabilidad del 65%. Se añadieron SFX para diferenciar entre recibir un golpe y la protección del escudo. Además, los enemigos podían aparecer con diferentes sprites aleatoriamente, como muestra la Ilustración 40, atacar al jugador, aún sin la animación correspondiente, y, al morir, dejar caer una bolsa de oro representado en la Ilustración 41. También se creó la máquina de estados de los enemigos, aunque no se implementó en esta iteración.

Se actualizó el GDD evaluando la viabilidad de las ideas propuestas inicialmente y añadiendo nuevas ideas y detalles específicos.

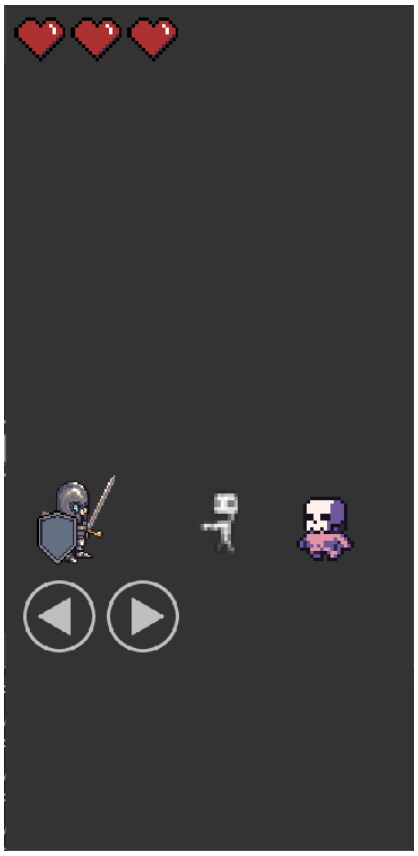

Ilustración 40. Avance del sprint 1, implementación de los enemigos, los corazones y el escudo.

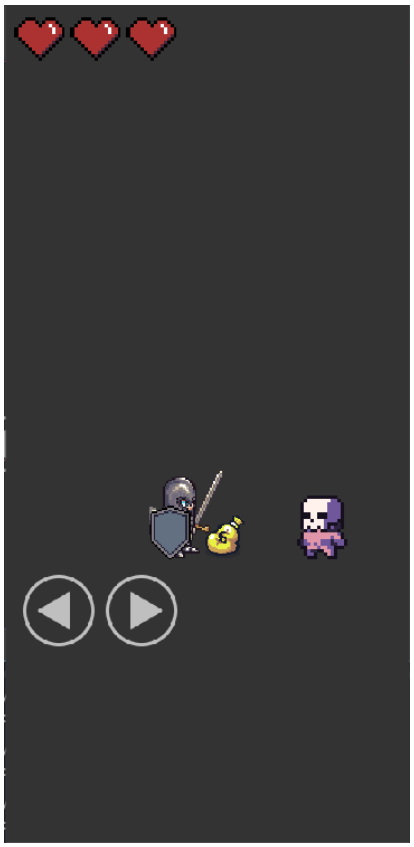

Ilustración 41. Avance del sprint 1, bolsa con monedas de oro y enemigo eliminado.

5.1.3. Sprint 2

Este fue uno de los sprints más costosos debido a que fue el primer sprint donde hubo un primer contacto con la generación procedural por la realización de las US05 y US06.

Para la generación procedural en videojuegos 2D, Godot tiene un nodo llamado tilemap, el cual facilita la creación de mapas en tiempo de ejecución. En este sprint se estuvo haciendo un análisis de su uso e implementación para poder crear mapas dinámicamente, siendo esta una de las tareas más costosas en el avance del proyecto. La dificultad radicaba en cómo crear en tiempo de ejecución e interactuar con los objetos creados por el tilemap, debido a que son imágenes posicionadas en una rejilla bidimensional. Godot aporta una manera de hacerlo a base de *colliders*, pero igualmente se dificultaba debido al acceso que se debe realizar para acceder a las propiedades del tilemap y la diferenciación entre colliders de un objeto u otro.

Cuando se comprendió completamente la herramienta, se implementaron las tareas de generación de escaleras para pasar al siguiente piso, junto con un indicador, como se muestra en la Ilustración 42, que permitía al jugador subir con una animación, como se enseña en la Ilustración 43, generando nuevos enemigos, mostrado en la Ilustración 44, y preparando el siguiente nivel fuera del campo de visión de la pantalla.

Por último, se actualizó nuevamente el GDD añadiendo nuevas funciones y personajes.

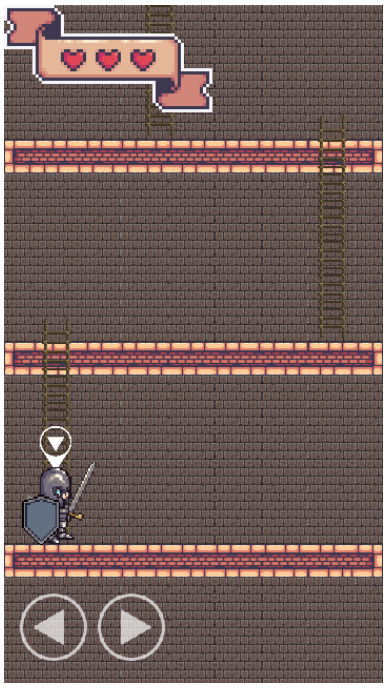

Ilustración 42. Avance del sprint 2, generación del mapa inicial y pop-up de la escalera para subir al siguiente nivel.

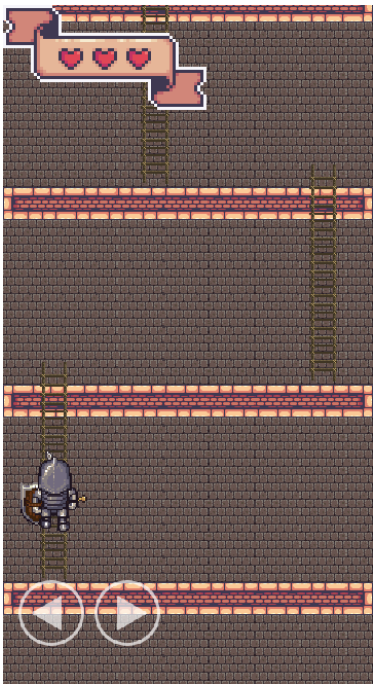

Ilustración 43. Avance del sprint 2, jugador subiendo por la escalera.

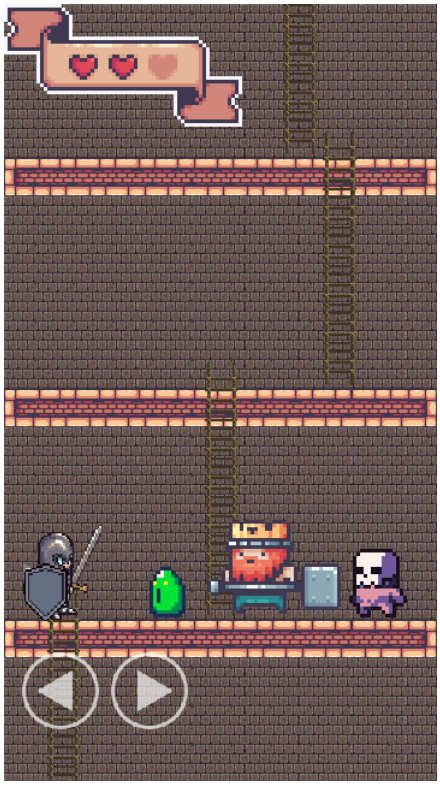

Ilustración 44. Avance del sprint 2, generación de enemigos tras subir al siguiente piso y pérdida de vida al ser golpeado.

5.1.4. Sprint 3

En este sprint se implementaron las US07, US08 y US09, las cuales no tuvieron mucha complicación a la hora de desarrollarlas.

En la US07 se añadió un nuevo botón de acción de ataque en la pantalla, que activaba una señal, correspondiente al patrón observador, para que el jugador ejecutara la animación de ataque, restando una vida al enemigo si estaba en el rango. Esto mismo se puede visualizar en la Ilustración 45.

Las US08 y US09 implicaron la incorporación de música y SFX del videojuego. Estas US no tuvieron mucha complicación dado que, en Godot, la incorporación de la música se puede realizar con un nodo ya preparado para esta función, llamado *AudioStreamPlayer*. Para los SFX se necesitaba que un mismo elemento reproduzca diferentes SFX. Godot tiene una función dentro del nodo AudioStreamPlayer la cual reproduce un sonido aleatorio entre los diferentes sonidos que se le especifique, simplificando bastante la realización de esta US.

En este mismo sprint se implementó la recogida de las bolsas de dinero al pasar por encima y se añadieron nuevos enemigos junto con nuevas animaciones. También se modificó la representación de los corazones en pantalla.

Por último, en este sprint se realizaron las últimas modificaciones al GDD añadiendo los diferentes elementos, acabando con la redacción de este documento.

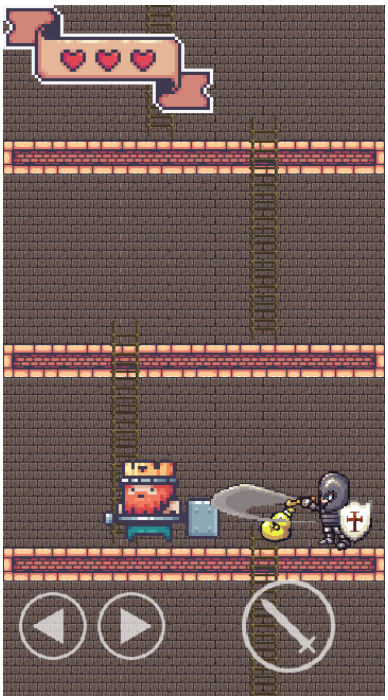

Ilustración 45. Avance del sprint 3, jugador atacando junto con el botón de acción de atacar.

5.1.5. Sprint 4

En este sprint se implementó la IA de los enemigos junto con diversas mejoras en su funcionamiento, todo ello correspondiente a la US10. Además de la implementación de la IA, esta US permitió dotar a cada enemigo de atributos únicos, haciendo que cada uno se diferenciase del resto en aspectos como alcance de ataque, velocidad de movimiento, tamaño, entre otros. También se añadió una barra de vida para los enemigos y un cambio de color de los enemigos al recibir daño, mostrado en la Ilustración 46.

En la US11, se implementó la funcionalidad de que el jugador pueda recibir golpes lanzados por los enemigos, no solo al chocar con ellos. Cada golpe recibido resta una vida, a menos que el escudo esté activo. Además, se introdujo un cambio de color en el jugador al ser golpeado, indicando visualmente cuándo se ha recibido un ataque. Cuando el jugador se queda sin vidas, se muestra una pantalla de fin de partida, representada en la Ilustración 47, que incluye el número de pisos superados y dos botones: uno para reiniciar la partida y otro para cerrar la aplicación. Dado que en esta nueva pantalla se muestra los pisos superados, se decidió incluir en la partida un marcador con el piso en el que se encuentra el jugador.

Finalmente, en la US12, se implementó el funcionamiento de las bolsas de oro y se crearon los diamantes como se puede observar en la Ilustración 46 e Ilustración 48. Las bolsas de oro siempre se generan al morir los enemigos y contienen una cantidad variable de monedas, que inicialmente estaba entre 200 y 450, pero se redujo a un rango de 5 a 20 monedas. Los diamantes tienen una probabilidad de aparecer del 20% al

eliminar un enemigo, y siempre será solo uno. Se añadieron contadores para las monedas de oro y diamantes acumulados durante la partida, los cuales comienzan desde cero en cada partida. Si el jugador pierde una vida, también puede perder entre 10 y 50 monedas de oro, pero los diamantes no se pierden debido a su rareza.

Ilustración 46. Avance del sprint 4 donde el enemigo recibe un ataque y se visualizan los contadores de oro, diamantes y pisos superados.

Ilustración 47. Avance del sprint 4 de la pantalla de fin de partida con los botones y el número de pisos superados.

Ilustración 48. Avance del sprint 4, soltar diamantes y bolsas de oro.

5.1.6. Sprint 5

En esta iteración se realizaron las US13, US14, US15 y US16. Este fue el sprint más complejo debido a que la US13, específicamente la tarea de "Crear salas", implicaba modificaciones y adiciones significativas a la generación procedural, lo cual ya había sido costoso en su implementación inicial.

La tarea "Crear salas" buscaba incorporar diferentes salas dentro de un mismo piso de la torre, de las que ni el jugador ni los enemigos pudiesen salir como se muestra la

Ilustración 50. Esto planteaba problemas, como la necesidad de bajar a pisos anteriores con escaleras para evitar que el jugador quedase atrapado, y asegurar que hubiera al menos un camino válido para seguir ascendiendo, ya que dentro de estas salas no debería haber ninguna escalera generada o muy pocas para no saturar el mapa de escaleras. Dado que la eliminación de todos los enemigos y la recolección de recursos en el piso actual eran requisitos para avanzar, la división en salas podía resultar tediosa para el jugador.

Tras revisar el GDD y considerando la dificultad de la tarea dentro del alcance temporal del MVP, se decidió simplificar la idea original. Las salas ahora se diseñaron para que los enemigos no pudieran salir, pero el jugador sí, así se evitaba el tener que hacer bajar al jugador a una sala ya completada, logrando así mantener la esencia de videojuego de ascenso continuo, como indica su nombre Ascent Tower.

Debido a que se inició la tarea sin tener estas consideraciones en cuenta, esta creó retrasos en el sprint, teniendo que reestimar la duración de la US y modificar la tarea con las consideraciones anteriores. Esto provocó que no se completaran por completo 3 tareas programadas para el sprint, creando flecos para el siguiente sprint como se muestra en la Ilustración 49.

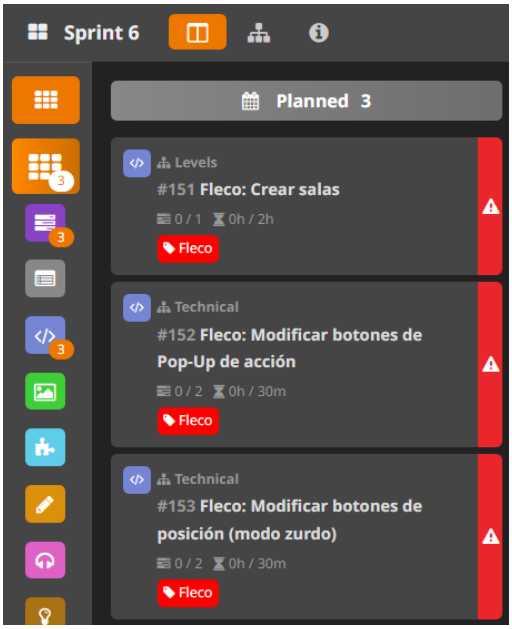

Ilustración 49. Imagen del inicio del siguiente sprint con las 3 tareas como flecos del sprint 5 y puestas con el estado Urgente proporcionado por HacknPlan.

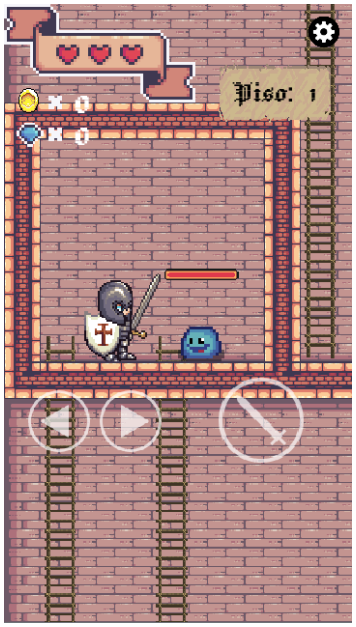

Ilustración 50. Avance sprint 5, creación de las salas con un enemigo en su interior.

A pesar de lo anterior, las US14 y US16 se completaron en el sprint asignado sin necesidad de la creación de flecos para estas.

En cuanto a la US14, esta no tuvo mucha complicación debido a que era representar datos por pantalla que ya habían sido calculados anteriormente durante la ejecución de la partida. Al finalizar la partida, estos datos se enviaban mediante señales de Godot

(patrón observador) y se mostraban en pantalla tal como muestra la Ilustración 51, Ilustración 52, Ilustración 53 e Ilustración 54. También se guardó en el dispositivo localmente el récord para así poder avisar al usuario en caso de haberlo superado.

Ilustración 51. Avance sprint 5, pantalla de fin de partida sin récord y sin estadísticas.

Ilustración 52. Avance sprint 5, pantalla de fin de partida sin récord, pero con estadísticas.

Ilustración 53. Avance sprint 5, pantalla de fin de partida con nuevo récord, pero sin estadísticas.

Ilustración 54. Avance sprint 5, pantalla de fin de partida con nuevo récord y con las estadísticas.

Respecto a la US16, dado que la funcionalidad de los cofres era muy similar a los objetos soltados por los enemigos al morir, y que el pop-up de acción era muy similar al de la escalera, salvo que este era más sencillo debido a que no se realizaba mediante el tilemap, no tuvo mucha complicación (ver Ilustración 55, Ilustración 56, Ilustración 57 e Ilustración 58).

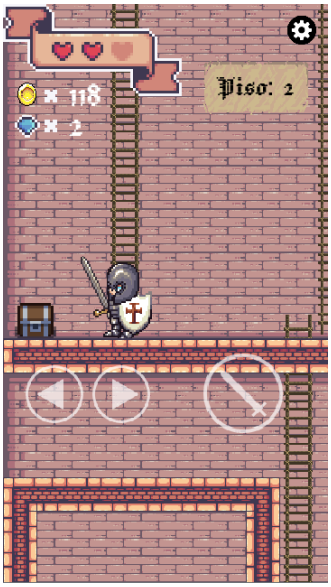

Ilustración 55. Avance sprint 5, cofre sin abrir.

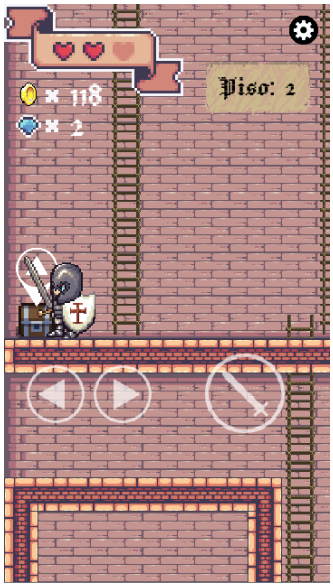

Ilustración 56. Avance sprint 5, pop-up del cofre.

Ilustración 57. Avance sprint 5, recompensas del cofre.

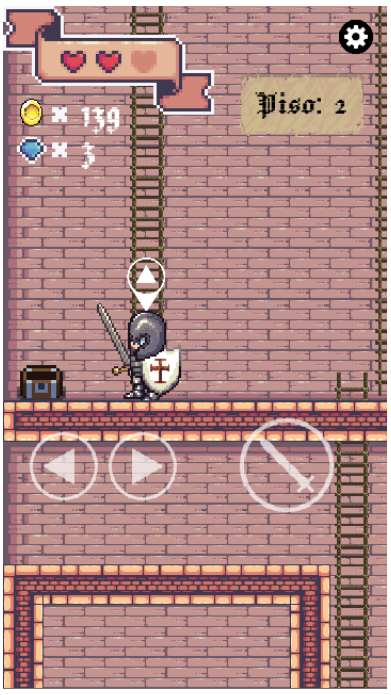

Ilustración 58. Avance sprint 5, cofre abierto y posibilidad de seguir avanzando al siguiente piso de la torre.

Finalmente, la US15 no se completó totalmente debido a la falta de tiempo retirado por la tarea de la US13. Se priorizaron las tareas de mayor importancia, siendo estas la gestión del volumen de la música, SFX y volumen general del juego (el cual incluye a ambos conjuntamente), como se puede ver en la Ilustración 59. Las tareas restantes quedaron como flecos, con el código mayoritariamente finalizado, faltando solo la representación visual de los cambios (los botones de los ajustes ya están realizados pero ocultos).

Ilustración 59. Avance sprint 5, menú de ajustes con los diferentes *sliders* para modificar los sonidos.

5.1.7. Sprint 6

En el penúltimo sprint del MVP, se priorizaron las tareas pendientes que no se pudieron completar en el sprint anterior. Estas tareas eran: “Crear salas”, “Modificar botones de pop-up de acción” y “Modificar botones de posición (modo zurdo)”.

La US13 contenía la tarea “Crear salas” como fleco. Estaba casi finalizada, solo quedaba corregir algunos fallos detectados que surgieron durante su desarrollo.

Por otro lado, la US15 tenía dos flecos, “Modificar botones de pop-up de acción” y “Modificar botones de posición (modo zurdo)”. Estas tareas ya tenían la mayor parte del código completado y la interfaz creada en la sección de ajustes, pero aún faltaba representar los cambios en pantalla para que el jugador pudiera interactuar con ellos. La finalización de estas tareas no fue muy costosa.

A continuación, se muestra el avance de estas tareas completadas. En la Ilustración 60 se presenta el menú de ajustes con los botones visibles. La Ilustración 61 muestra los botones intercambiados de posición. En la Ilustración 62 se muestran los botones de acción (pop-ups) cambiados a una posición estática en pantalla. Finalmente, en la Ilustración 63 se observa ambas opciones activadas simultáneamente.

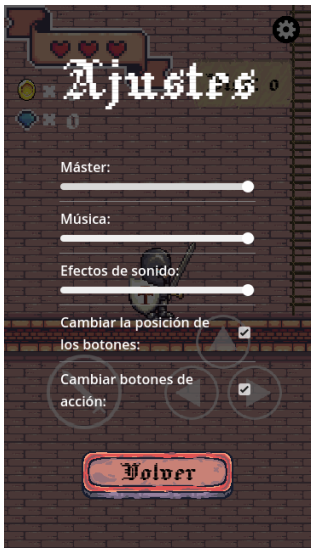

Ilustración 60. Menú de ajustes con las opciones “Cambiar la posición de los botones” y “Cambiar botones de acción” añadidas y activadas.

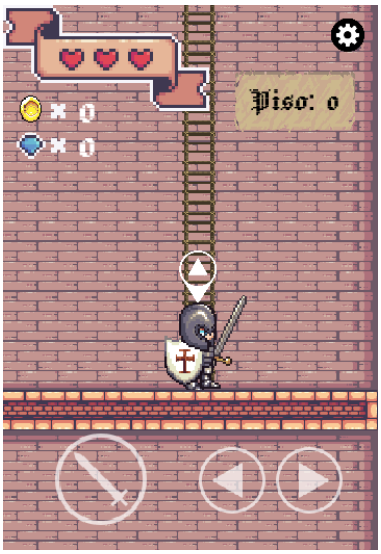

Ilustración 61. Opción “Cambiar la posición de los botones” activa en partida.

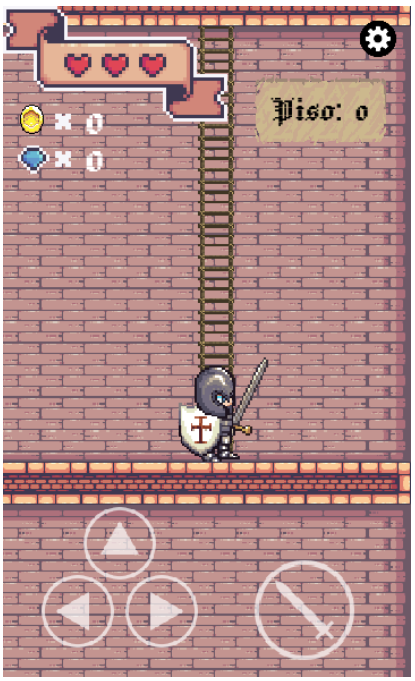

Ilustración 62. Opción “Cambiar botones de acción” activa en partida.

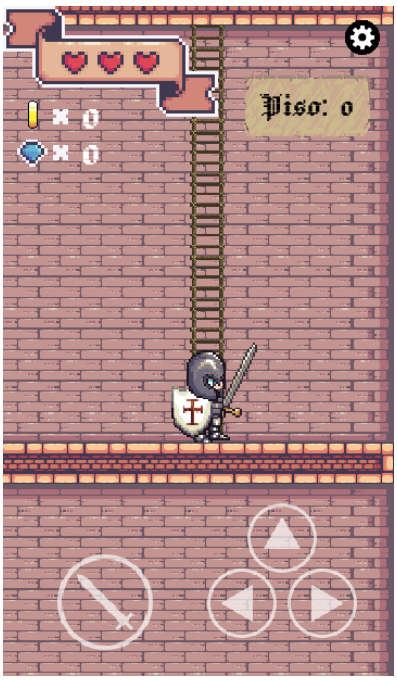

Ilustración 63. Opciones “Cambiar la posición de los botones” y “Cambiar botones de acción” activas simultáneamente en partida.

Tras la finalización de los flecos, y dado que en este sprint se pretendía tener el MVP prácticamente completado para realizar la implantación del videojuego y su disfrute por los usuarios, se decidió realizar las US17, US18, US19 y US20, que daban los

últimos retoques al videojuego para que los usuarios percibieran un producto finalizado. El siguiente sprint se destinó principalmente a implementar propuestas de mejora y corregir los fallos encontrados por los usuarios.

La US17 consistía en la creación del menú principal del videojuego, que no fue complicado de realizar, ya que implicaba implementar botones con funcionalidades sencillas. En la Ilustración 64 se muestra el menú principal integrado en el videojuego, con los botones de “Jugar”, que inicia una nueva partida, “Créditos”, creado por la US18, “Salir”, que presenta un mensaje de confirmación como se observa en la Ilustración 65, el botón de ajustes reutilizado de la US15 y el botón de la tienda, que se detallará en la explicación de la US19.

Ilustración 64. Menú principal de Ascent Tower.

Ilustración 65. Mensaje de confirmación del botón "Salir".

En la US18 se crearon los créditos del videojuego, mostrados en la Ilustración 66, que incluye a los desarrolladores: Mario Mocholí Rubio, como compositor, y Javier Pérez Asensio, como programador.

Ilustración 66. Créditos del videojuego Ascent Tower.

La US19 se centró en la creación de la tienda, como se muestra en la Ilustración 67, donde los jugadores pueden gastar el oro y los diamantes recolectados en partidas anteriores para comprar diferentes tipos de pociones: la poción roja (vida extra), la poción amarilla (más velocidad) y la poción gris (reintento en caso de quedarse sin vidas). La poción roja se puede comprar con oro o diamantes, mientras que las pociones amarilla y gris, debido a su rareza y efectos, y a la rareza de los diamantes, solo están disponibles a cambio de diamantes.

Ilustración 67. Tienda del juego con opciones de compra tanto con diamantes como con oro.

Tanto en la Ilustración 68 como en la Ilustración 69 se muestran los mensajes de aviso que la tienda despliega cuando no hay suficiente oro o diamantes, y en la Ilustración 70, el mensaje de aviso por falta de espacio en la bolsa.

Ilustración 68. Mensaje de aviso en la tienda de diamantes insuficientes.

Ilustración 69. Mensaje de aviso en la tienda de oro insuficiente.

Ilustración 70. Mensaje de aviso en la tienda de espacio insuficiente en el inventario.

La última US del sprint fue la US20, que integró la representación de las compras de la tienda durante la partida,y que se pudiese hacer uso de los artículos comprados.

En las primeras partidas, antes de comprar por primera vez artículos en la tienda o cuando se haya consumido todos los artículos comprados, la mochila se representa vacía, pero igualmente se puede interactuar con esta, teniendo dos estados: la mochila cerrada, como se ve en la Ilustración 71, y la mochila abierta, como se ve en la Ilustración 72.

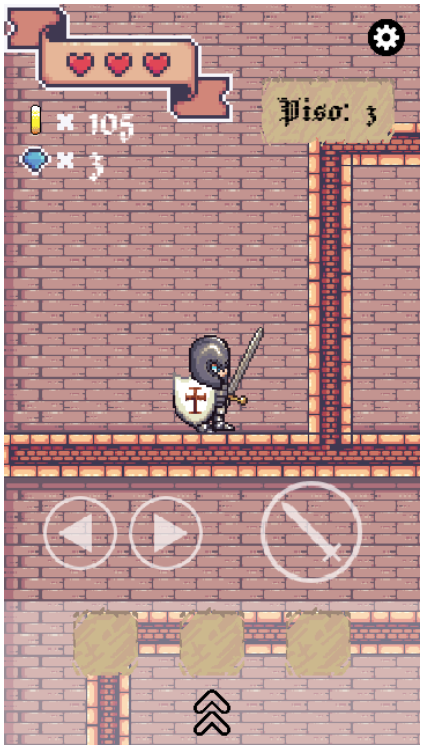

Ilustración 71. Mochila cerrada sin objetos.

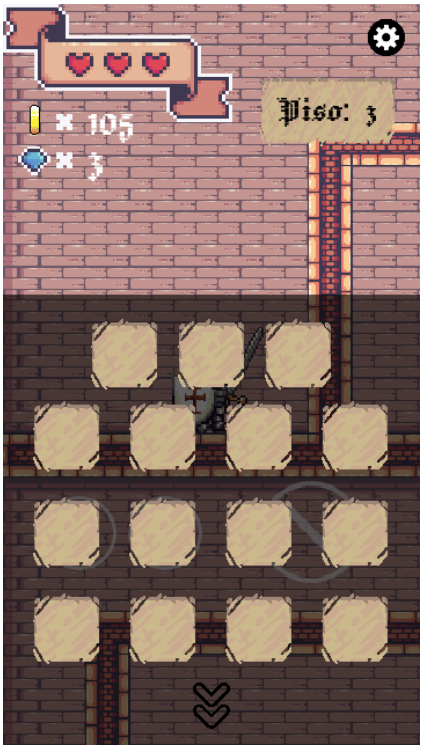

Ilustración 72. Mochila abierta sin objetos.

Los artículos comprados se representan por orden de compra en la mochila, posicionando los tres primeros en las ranuras de acción, como se muestra en la Ilustración 73, y el resto guardados en la mochila como se muestra en la Ilustración 74.

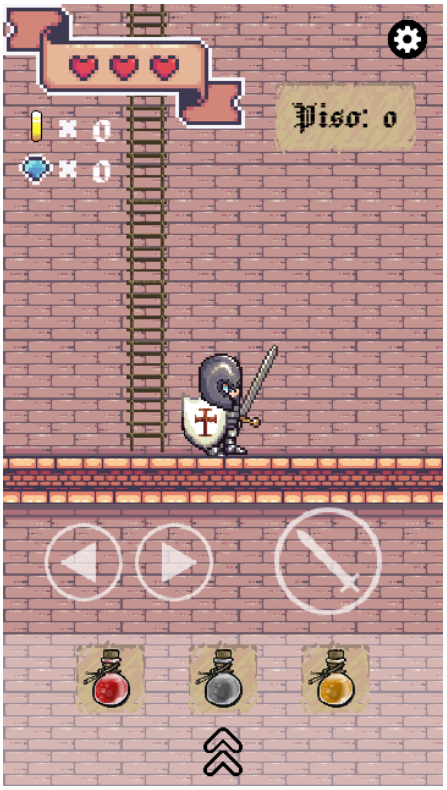

Ilustración 73. Mochila cerrada con objetos.

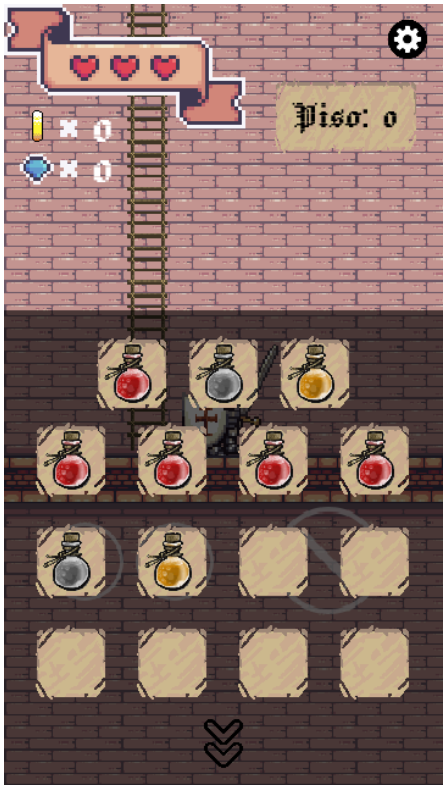

Ilustración 74. Mochila abierta con objetos.

Con la mochila cerrada se pueden usar las pociones pulsando en las tres ranuras de acceso rápido durante la partida. Con la mochila abierta, se pueden reorganizar las pociones, pudiéndolas poner en las ranuras de acceso rápido.

5.1.8. Sprint 7

Al inicio del último sprint, se preparó el videojuego para su lanzamiento, creando la US46. Esta US incluía tareas destinadas a corregir los errores conocidos del videojuego y a mejorarlo antes de su publicación. La mayor parte del tiempo se dedicó a la corrección de errores, puliendo el juego para asegurar su correcto funcionamiento.

En el apartado de mejoras, se realizaron ajustes visuales significativos, como se muestra en la Ilustración 75. Esto incluyó tanto la visualización del estado del juego como mejoras en el entorno de juego. A destacar los botones de la partida, que ahora muestran en pantalla un efecto cuando son pulsados, informando al jugador sobre las acciones que está realizando.

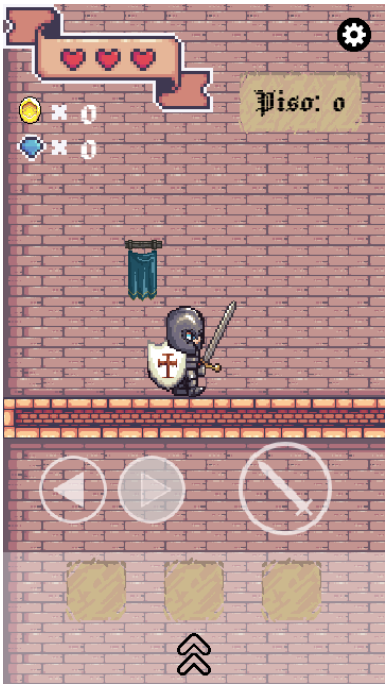

Ilustración 75. Mejoras visuales del videojuego.

Una vez finalizadas estas tareas, se investigó cómo publicar un videojuego en la plataforma itch.io. Se elaboró una descripción detallada del videojuego, incluyendo capturas de pantalla, y se creó la página principal de descarga (ver Ilustración 82). Esta página se compartió a través de diversos medios para informar a los usuarios sobre la disponibilidad del videojuego.

Pasados unos días, se recopilaron las respuestas de la encuesta, como se detalla en el apartado “Pruebas”, y se planteó la US47 con las nuevas tareas a realizar. Estas tareas fueron en general sencillas, excepto la de “Tienda en partida”, que implicaba modificar la generación procedural para incluir la tienda durante una partida.

En cuanto a la tarea de “Añadir nuevas pociones”, se decidió incorporar dos nuevas pociones y reorganizar la estructura de la tienda, como se ilustra en la Ilustración 76. La poción de “+ velocidad” ahora se compra con oro a un precio más reducido, ya que muchos jugadores no le veían utilidad a esa poción debido a un precio excesivo.

Ilustración 76. Tienda modificada tras los comentarios recibidos en la encuesta.

Las nuevas pociones son “+ protección”, que aumenta la probabilidad de activación del escudo, y “No perder oro”, que impide la pérdida de oro tras recibir golpes durante la partida, dando así mayor jugabilidad con el uso de nuevas pociones en el juego.

Respecto a las tareas “Modificar interfaz del menú principal respecto al botón de la tienda” y “Modificar icono de la tienda”, se hizo que el icono sea más visible y atractivo, como se observa en la Ilustración 77. Se añadió una animación y se reubicó el icono cerca del botón de “Jugar” en el menú principal. Además, se cambió el icono que representa la tienda, ahora representado por una poción, lo cual resulta más adecuado para la temática del juego.

Ilustración 77. Botón de la tienda modificado tras incorporar los cambios recomendados de la encuesta.

También se sugirió en los comentarios la aparición de la tienda de forma aleatoria a lo largo de una partida. Esta tarea, denominada “Tienda en partida”, ha sido una de las más complicadas de implementar debido a la modificación de la generación procedural. La tienda se mostraría tal y como observa en la Ilustración 78 y llevaría al jugador a la tienda que se encuentra en la Ilustración 76, pero con una diferencia. En el caso de acceder a la tienda a través de una partida en vez desde el menú principal, solo se podrán comprar artículos con el oro y diamantes acumulados en esa partida. En cambio, si se accede desde el menú, se tendrá la cantidad total de oro y diamantes acumulados de partidas anteriores. Con esto se consigue que el jugador pueda ahora gastar sus recursos en la tienda del juego en mitad de una partida, pero tenga que ser más cuidadoso al elegir qué producto desea comprar.

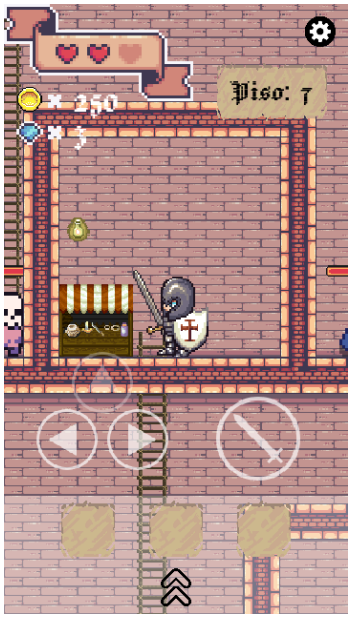

Ilustración 78. Representación de la tienda en un piso de una partida.

Finalmente, respecto a otros comentarios recibidos, se decidió realizar modificaciones a los cofres y a los botones de acción por defecto. En cuanto a los cofres, se mejoró el *feedback* al recoger las recompensas obtenidas, añadiendo SFX. Por varios comentarios recibidos, se configura por defecto el botón de acción estático en pantalla en vez de mostrar los pop-ups en los eventos correspondientes. Debido a que esta es ahora la configuración por defecto del juego, se mejoró este botón, haciendo que se muestre cuando no pueda ser pulsado y cuándo sí, como se muestra en la Ilustración 79 e Ilustración 80.

Ilustración 79. Botón de evento deshabilitado.

Ilustración 80. Botón de evento habilitado.

6. Implantación

En esta sección se llevará a cabo la puesta en producción del videojuego.

Para la puesta en marcha del videojuego, el editor Godot, ofrece una herramienta denominada "*Export*". Esta herramienta permite exportar el videojuego desarrollado a diversas plataformas, entre las cuales se encuentra Android, que es la plataforma de interés para este proyecto. En la Ilustración 81 se muestra una imagen de la herramienta que ofrece Godot para exportar los videojuegos.

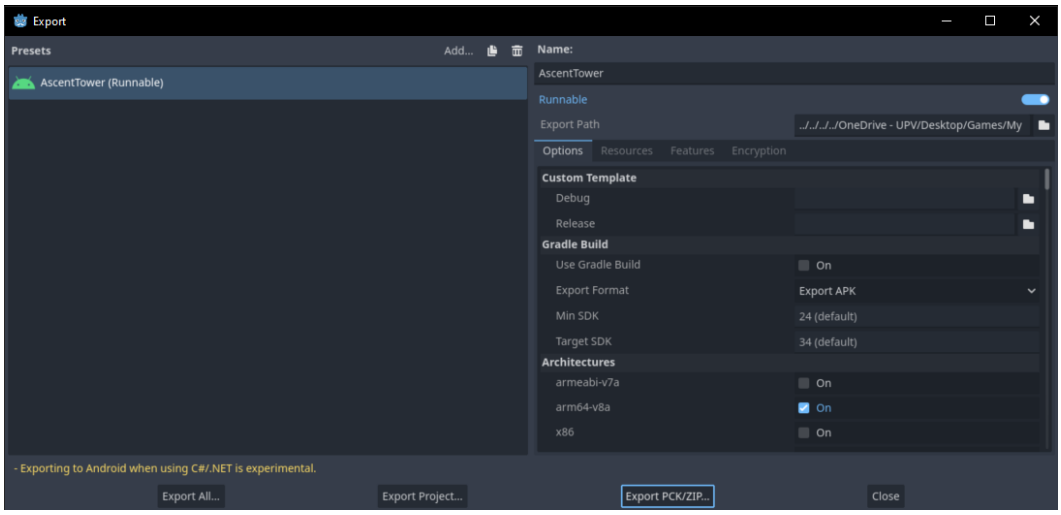

Ilustración 81. Herramienta export de Godot.

Gracias a esta herramienta, se simplifica el proceso de llevar el videojuego a las plataformas deseadas. Para Android, se requiere la provisión del Android SDK, una *Debug Keystore* y el Java SDK. A diferencia de otros entornos de desarrollo como Unity, Godot, al ser de código abierto, no los implementa nativamente. Sin embargo, Godot simplifica este proceso solicitando únicamente las ubicaciones de instalación de los programas requeridos.

Con estos requisitos cumplidos, el videojuego está listo para ser lanzado al mercado. Además, se recopilarán las opiniones de los jugadores a través de una encuesta publicada en página de descarga del juego.

Para este proyecto, el videojuego será lanzado en la plataforma web de itch.io como se ve en la Ilustración 82, conocida por ser un espacio para encontrar y compartir juegos indie en línea de forma gratuita (itch.io, 2024).

La descarga del videojuego está disponible en el siguiente enlace: https://estimp.itch.io/ascent-tower.

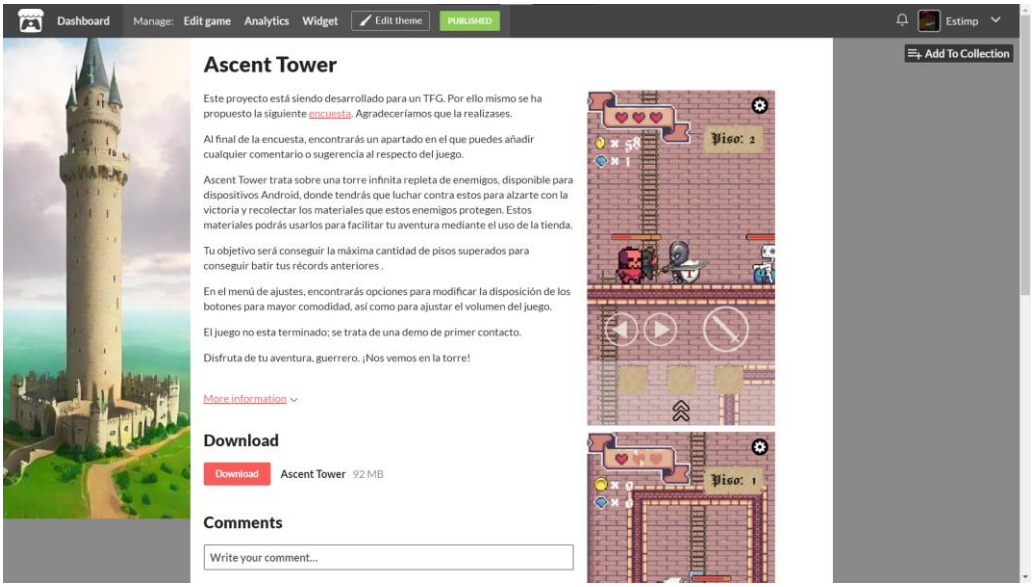

Ilustración 82. Captura de pantalla de la página web de itch.io del videojuego Ascent Tower.

7. Pruebas

En esta sección se analizarán los resultados de la encuesta que acompañaba al videojuego, dirigida a los usuarios que lo han descargado y jugado. Esta encuesta permite identificar puntos de mejora y detectar las funcionalidades más apreciadas durante la experiencia de juego.

La primera pregunta planteada fue "¿Qué te ha parecido el nivel de dificultad del juego?", con opciones de respuesta del 1 al 5, donde 1 significa "Muy fácil" y 5 "Muy difícil". Como se puede observar en la Ilustración 83, en general, el nivel de dificultad del juego ha estado balanceado, evitando que sea injugable por su dificultad o demasiado fácil, lo que podría llevar al jugador a aburrirse.

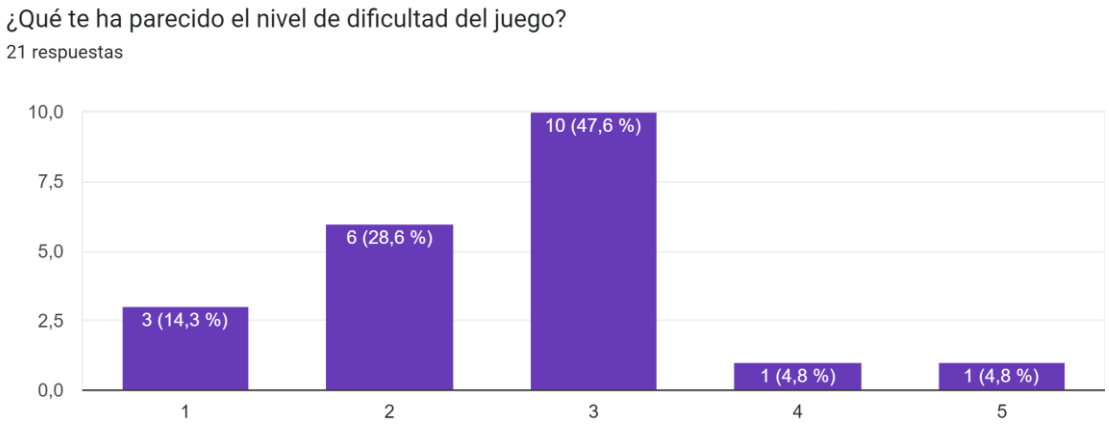

Ilustración 83. Encuesta sobre la dificultad del juego, siendo 1 "Muy fácil" y 5 "Muy difícil".

La siguiente pregunta de la encuesta, representada en la Ilustración 84, trata sobre el movimiento del personaje, con opciones del 1 al 5, significando 1 "Injugable" y 5 "Muy cómodo". En general, el movimiento del personaje ha sido bien valorado, con respuestas mayoritariamente positivas.

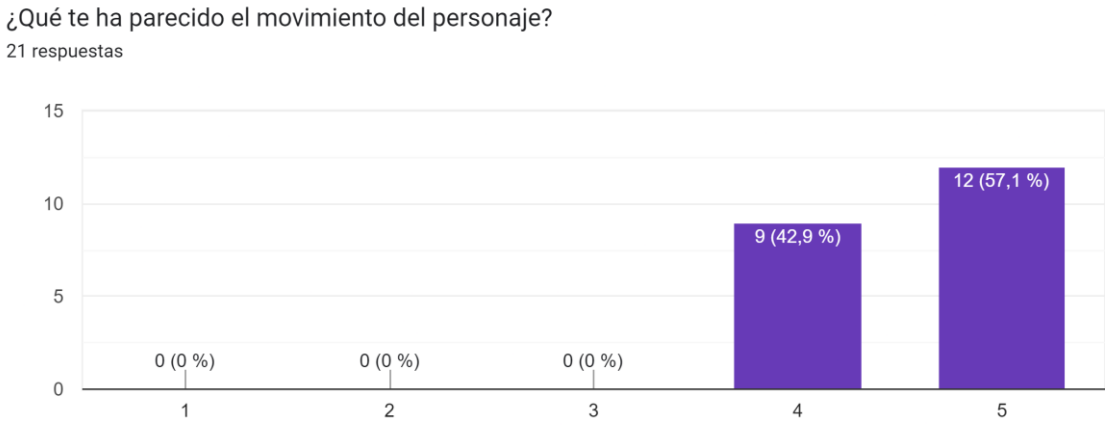

Ilustración 84. Encuesta sobre el movimiento del personaje, siendo 1 "Injugable" y 5 "Muy cómodo".

En la Ilustración 85 se representa la encuesta sobre el diseño de nivel, con respuestas del 1 al 5, donde 1 significa “Muy malo” y 5 “Muy bueno”. Las respuestas indican que el diseño del nivel también ha sido bien recibido por los usuarios.

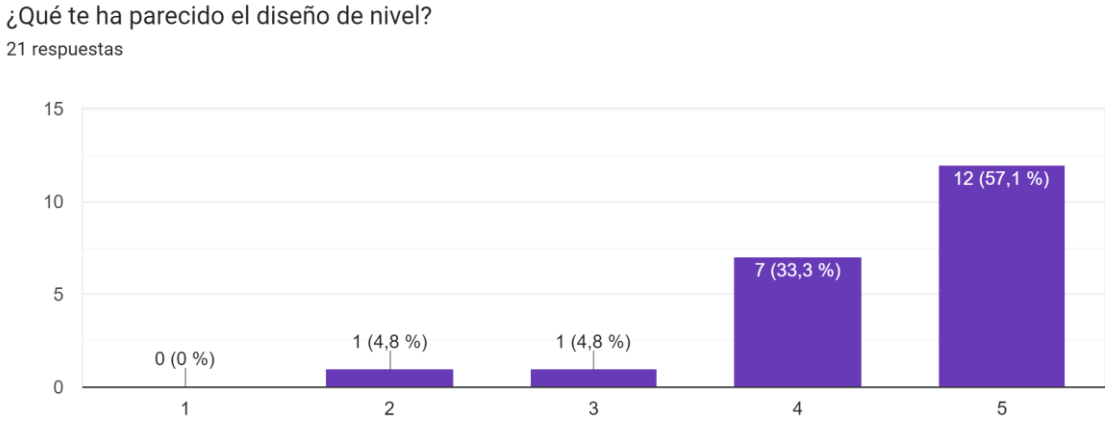

Ilustración 85. Encuesta sobre el diseño de nivel, siendo 1 "Muy malo" y 5 "Muy bueno".

Pasando a las encuestas con respuesta de “Sí” o “No”, se incluyen dos preguntas: “¿Te parece un juego entretenido?”, representada en la Ilustración 86, y “¿Volverías a jugar en futuras actualizaciones?”, representada en la Ilustración 87. Ambas encuestas recibieron respuestas afirmativas en su totalidad, lo que indica que el juego ha sido del agrado de los jugadores y que existe interés en su evolución futura.

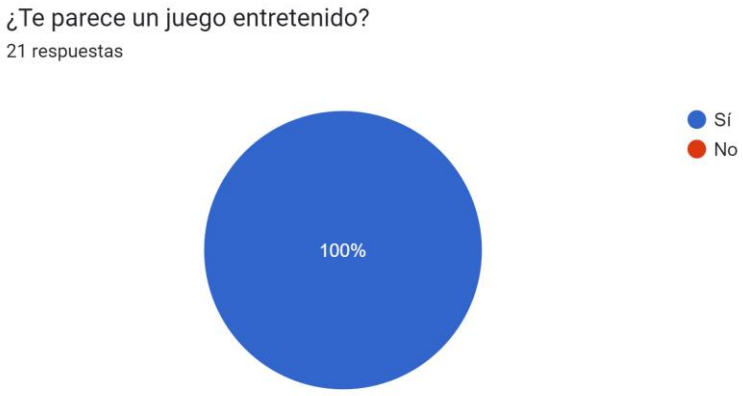

Ilustración 86. Encuesta sobre si el videojuego es entretenido, con respuesta de "Sí" o "No".

Ilustración 87. Encuesta sobre si se volvería a jugar al videojuego en un futuro, con respuesta de "Sí" o "No".

Asimismo, se realizó una encuesta de respuesta múltiple, representada en la Ilustración 88, sobre las mecánicas del juego que más han gustado a los usuarios. La mecánica de “Atacar” fue la más votada, seguida por “Moverse” y “Recoger oro y diamantes”.

Ilustración 88. Encuesta con respuesta múltiple sobre las mecánicas favoritas de los usuarios.

Para finalizar, se incluyó una pregunta de respuesta abierta, mostrada en la Ilustración 89, donde los jugadores podían aportar propuestas de cambios o mejoras a incorporar en futuras actualizaciones. De esta pregunta se extrajeron las tareas de mejora a realizar en la US47 en el sprint 7:

1. “Añadir más pociones con diferentes efectos”.
2. “Cambiar la interfaz porque no se ve que exista la tienda […]”.
3. “[…] Y otra propuesta sería la de, que en lugar de que hubiese un carro como ícono de la tienda, que hubiese algo más medieval, como por ejemplo una poción.”
4. “[…] la otra forma de subir es mucho más cómoda por lo que la pondría por defecto. […]”.
5. “Quizás se podría hacer la apertura de cofres un poco más vistosa. Al abrirlos, recoges casi al instante las monedas y los diamantes y prácticamente no los ves. Que caigan al suelo primero o que suenen las monedas lo haría más satisfactorio.”.
6. “Diría que cada ciertos niveles subidos, estuviese la posibilidad de comprar en la tienda, ya que si no, tienes que salir y volver a empezar después dd comprar. […]”.
7. “[…] También le metería menos rango a la espada al atacar para mayor dificultad […]”.

Dado que la propuesta de incluir la capacidad de saltar se repitió, se ha valorado la inclusión de una nueva US, siendo esta la US48, para una funcionalidad futura, aportada por las respuestas “Molaría que pudieses saltar pinchos o algo así. […]” y “Que el muñeco salte”.

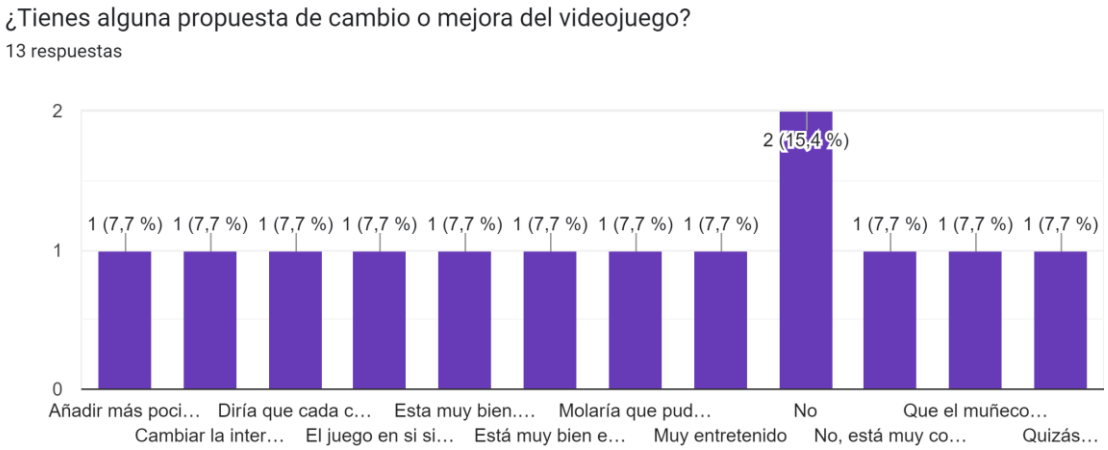

Ilustración 89. Pregunta de respuesta abierta sobre propuestas de cambio o mejora.

8. Conclusiones

Tras la finalización de este TFG y repasando los objetivos establecidos inicialmente, se puede afirmar que el proyecto ha cumplido todos los objetivos propuestos.

El objetivo principal de este proyecto era la creación de un videojuego 2D, para dispositivos móviles, desde su concepción inicial hasta alcanzar el MVP utilizando el entorno de desarrollo de videojuegos Godot Engine, junto con un estudio de las funcionalidades que proporciona esta herramienta. Este objetivo se ha logrado completamente gracias a los numerosos recursos disponibles en internet y a las guías ofrecidas en la página web oficial de Godot Engine. Durante el desarrollo, he aprendido gradualmente a utilizar esta tecnología, consiguiendo así que al inicio del sprint 7 se realizara la puesta en marcha del MVP del videojuego en la plataforma itch.io, donde los jugadores pudieron probar el videojuego.

Los objetivos secundarios también se han cumplido satisfactoriamente. Durante todo el proyecto, se ha utilizado metodologías ágiles y se han profundizado los conocimientos en este ámbito con el uso de la plataforma HacknPlan, la cual ha sido muy útil para planificar el trabajo y entregarlo al PO a tiempo, incluso con reestimaciones y modificaciones acordadas. Además, se han desarrollado diferentes patrones de diseño a medida que se avanzaba en las tareas de creación del proyecto.

Uno de los logros más complejos ha sido la creación de la generación procedural de los escenarios, que constituía la base del videojuego. Esta característica ha sido la que más tareas y dificultades ha presentado, debido a que era la primera vez que trataba con algo así y había poca información especializada en Godot al respecto. No obstante, el estudio realizado ha permitido adquirir un gran conocimiento sobre este tema.

Después del desarrollo de la aplicación y al evaluar los resultados obtenidos, considero que ha sido un proyecto muy enriquecedor. El proceso de desarrollo de un videojuego utilizando este motor gráfico ha resultado especialmente interesante. Mirando hacia el futuro, sería interesante continuar con el desarrollo del videojuego para lograr un producto final mucho más elaborado, con la intención de publicarlo en la plataforma de Play Store.

8.1. Relación del trabajo desarrollado con los estudios cursados

El desarrollo de este proyecto ha estado muy relacionado con varias asignaturas cursadas durante la realización de este grado. A continuación, se presentará de manera detallada una serie de asignaturas y conceptos aprendidos junto con su vínculo con el proyecto.

- **Diseño de Software**: en esta asignatura se adquirió agilidad en la comprensión e implementación de patrones de diseño en un proyecto software, así como en la realización de refactorizaciones al proyecto.
- **Proceso de Software**: se desarrolló un proyecto software utilizando metodologías ágiles, lo cual fue fundamental para entender la importancia de seguir metodologías establecidas y no ejecutar un proyecto sin un plan claro.

- **Análisis y Especificación de Requisitos**: se obtuvieron conocimientos sobre cómo adquirir y especificar los diferentes requisitos de un proyecto software, utilizando correctamente UC y US, entre otros modelos.
- **Mantenimiento y Evolución de Software**: se aprendió el uso de ramas y etiquetas en GitHub y la importancia de mantener un control de versiones activo en un proyecto.
- **Proyecto de Ingeniería de Software**: esta asignatura, similar a la asignatura Proceso de Software, profundizó en el uso de las metodologías ágiles y permitió desarrollar un nuevo proyecto, aportando soltura y conocimiento en el uso de dichas metodologías.
- **Desarrollo de Videojuegos 3D**: se expusieron las funcionalidades del motor gráfico Unity, que son muy similares a las de Godot, proporcionando las bases para la creación de videojuegos y las guías a seguir en el desarrollo de un nuevo videojuego.

Finalmente, es importante destacar las siguientes competencias transversales, las cuales han sido fundamentales para que este proyecto se haya llevado a cabo en el tiempo definido en los diferentes sprints.

- **Análisis y resolución de problemas**: para poder evaluar y diseñar soluciones efectivas a los desafíos que se presentan durante la ejecución del proyecto, transformando un problema complejo en problemas más manejables.
- **Innovación, creatividad y emprendimiento**: detectando necesidades de mejora y, ante ello, se han propuesto ideas novedosas y bien detalladas que pueden generar valor. Principalmente, esto se ha reflejado en el desarrollo del videojuego de forma procedural, lo cual está estrechamente relacionado con esta competencia transversal.
- **Trabajo en equipo y liderazgo**: esta competencia ha sido fundamental para para entenderme y coordinarme eficazmente con el PO.
- **Comunicación efectiva**: donde se ha demostrado la capacidad de transmitir convicción y seguridad a la hora de comunicarse con el equipo de desarrollo, ilustrando el discurso para facilitar su comprensión.

9. Trabajos futuros

Dado que la idea inicial del videojuego era su creación y lanzamiento al mercado en plataformas como *PlayStore*, se llevó a cabo una tormenta de ideas en la que se definieron futuras US para desarrollar un juego más dinámico, con más eventos y una mejora sustancial tanto en la generación procedural como en la jugabilidad. Estas US corresponden a las US desde la US21 hasta la US45, incluyendo también la US48.

Se planea incluir nuevos modos de juego para evitar que exista una única modalidad de combate cuerpo a cuerpo, incorporando también ataques a distancia con arcos o hechizos mágicos, como se describe en las US33, US41 y US42.

Asimismo, se pretende añadir nuevas funcionalidades no tan relacionadas con las características principales, pero que ofrecerían mayores opciones de jugabilidad, como se observa en las US21, US22, US23, US27, US32, US34, US35, US36, US44, US45 y US48 y mayor competitividad dentro del juego, tal como se detalla en las US26, US29, US30 y US36.

Para otorgar un mayor grado de profesionalismo al videojuego, se definieron US como las US24, US25, US28, US31, US37 y US38. Estas US incluirían pantallas de carga para informar al usuario sobre el estado del videojuego durante los tiempos de carga, diferentes animaciones que mejorarían el aspecto visual del juego, una narrativa más profunda y la posibilidad de cambiar el idioma del juego.

Además, se podrían mejorar los niveles del juego con nuevas texturas y sprites que lo hagan más atractivo visualmente. Sin embargo, para optimizar el diseño de los sprites, sería ideal contar con un diseñador gráfico especializado en Bellas Artes y con experiencia en diseño digital para videojuegos.

Bibliografía

Arce, L. J. (2011). Desarrollo de videojuegos. *Mendoza: Universidad de Aconcagua*.

Culiáñez Llorca, A. (2023). Desarrollo del videojuego educativo PLMan en Godot Engine.

Epic Games. (26 de junio de 2024). Obtenido de https://www.epicgames.com/site/es-ES/home

Figma. (26 de junio de 2024). Obtenido de https://www.figma.com/es-es/

Gamma, E., Helm, R., Johnson, R., Vlissides, J., & Patterns, D. (1995). Elements of reusable object-oriented software. *Design patterns.*

Git. (26 de junio de 2024). Obtenido de https://git-scm.com/

GitHub. (26 de junio de 2024). Obtenido de https://github.com/

Godot Engine. (26 de junio de 2024). Obtenido de https://godotengine.org/

Google Play. (26 de junio de 2024). Obtenido de Tower of Hero: https://play.google.com/store/apps/details?id=com.Tatsuki.Tower&hl=es_419

Google Play. (26 de junio de 2024). Obtenido de Sling Kong: https://play.google.com/store/apps/details?id=com.protostar.sling

Google Play. (26 de junio de 2024). Obtenido de Super Starfish: https://play.google.com/store/apps/details?id=com.protostar.starfish

HacknPlan. (26 de junio de 2024). Obtenido de https://hacknplan.com/

Hernández Hernández, A. (2021). *Desarrollo de un videojuego en Unity con editor de niveles en línea* (Doctoral dissertation, Universitat Politècnica de València).

Holfeld, J. (2023). On the relevance of the Godot Engine in the indie game development industry. *arXiv preprint arXiv:2401.01909.*

Inkscape Draw Freely. (26 de junio de 2024). Obtenido de Inkscape: https://inkscape.org/es/

itch.io. (12 de 06 de 2024). Obtenido de https://itch.io/

Izaurralde, M. P. (2013). Caracterización de Especificación de Requerimientos en entornos Ágiles: Historias de Usuario. *Trabajo de especialidad, Febrero.*

Lizarraga, J. I. L. (2017). Motores de desarrollo de videojuegos más populares. *Universidad Politécnica de Sinaloa.*

Llubes Cano, X. (2022). Hit or Run: desarrollo de un videojuego 2D en Unity.

Lucid. (10 de junio de 2024). Obtenido de Lucidchart: https://www.lucidchart.com/pages/es

Microsoft. (26 de junio de 2024). Obtenido de Visual Studio: https://visualstudio.microsoft.com/es/

Microsoft. (26 de junio de 2024). Obtenido de C#: https://dotnet.microsoft.com/es-es/languages/csharp

Morales Cortés, D. (2016). *Steel Soldier: Un juego de plataformas-shooter desarrollado en Godot* (Doctoral dissertation, Universitat Politècnica de València).

Muñoz Jiménez, R. (2024). *Desarrollo del juego Sushi Go para móviles Android* (Doctoral dissertation, Universitat Politècnica de València).

MuseScore. (26 de junio de 2024). Obtenido de https://musescore.org/es

Pérez Luque, D., & Valverde Bourgon, H. (2021). Uso del movimiento de los Joy-Con de Nintendo Switch en el motor Unity.

Pérez Tapia, S. J. (2023). *Estudio comparativo de las metodología Kamban y Devops como estrategia de desarrollo ágil de proyectos de software* (Bachelor's thesis, Babahoyo: UTB-FAFI. 2023).

Pousa Barros, G. (2022). *Neyteria, un juego 2D de plataformas. Aspectos técnicos y gestión* (Doctoral dissertation, Universitat Politècnica de València).

Raurell Gomis, P. (2022). *Desenvolupament d'un personatge basat en l'estil artístic de Gaudí* (Bachelor's thesis, Universitat Politècnica de Catalunya).

Riano Nossa, N. D. (2021). Estudio comparativo de metodologías tradicionales y ágiles aplicadas en la gestión de proyectos.

Solís Flores, J. A. (2023). *Desarrollo de un videojuego roguelike en Unity* (Doctoral dissertation, Universitat Politècnica de València).

Tovstochub, I., & Zinchenko, O. (2024). OPTIMIZATION OF GENERATION OF IN-GAME CONTENT. *Collection of scientific papers «ΛΌГОΣ»*, (April 26, 2024; Bologna, Italy), 245-246.

Unity. (26 de junio de 2024). Obtenido de https://unity.com/es

Unreal Engine. (26 de junio de 2024). Obtenido de https://www.unrealengine.com/es-ES

Vázquez-Sánchez, R., & Chao-Fernández, R. (2019). MuseScore: crea y edita tus propias partituras.

Wagner, B. (14 de 05 de 2024). *Guía de C#: lenguaje administrado de .NET | Microsoft Learn*. Obtenido de Paseo por el lenguaje C#: https://learn.microsoft.com/es-es/dotnet/csharp/tour-of-csharp/overview

Glosario de términos

Android SDK: conjunto de herramientas de desarrollo proporcionado por *Google* para crear aplicaciones para dispositivos Android. Incluye bibliotecas, herramientas de depuración, emuladores y documentación.

Capas: separación lógica del sistema en niveles, donde cada capa tiene una responsabilidad específica. Esto facilita la organización, mantenimiento y escalabilidad del código.

Caso de uso (UC): UC, del inglés, Use Cases, describe cómo un usuario interactúa con un sistema para lograr un objetivo específico. Es un escenario detallado que define las interacciones y los resultados esperados.

Colliders: componentes que definen la forma y las dimensiones de los objetos para la detección de colisiones. Permiten que los objetos interactúen físicamente en el entorno del juego.

Debug Keystore: archivo utilizado durante el desarrollo de aplicaciones Android para firmar el código de forma provisional. Permite probar y depurar la aplicación en dispositivos reales y emuladores antes de la firma final para su distribución.

Documento de Diseño de Videojuegos (GDD): del inglés, Game Desing Document, es un documento detallado que describe todos los aspectos del diseño de un videojuego, incluyendo mecánicas, historia, personajes, niveles y reglas. Sirve como guía y referencia para todo el equipo de desarrollo.

Efectos de Sonido (SFX): del inglés, Sound Effects, utilizados en videojuegos para mejorar la inmersión y la experiencia del jugador. Incluyen sonidos de ambiente, efectos de acciones, interacciones y otros eventos del juego.

Engine: plataforma de software que proporciona herramientas y bibliotecas para el desarrollo de juegos. Incluye componentes para gráficos, física, sonido, inteligencia artificial y más, facilitando la creación y optimización de juegos.

Fleco: pequeño detalle o tarea pendiente que queda después de completar la mayor parte del trabajo. Puede ser una corrección menor o un ajuste final necesario antes de la entrega.

Generación procedural: técnica de desarrollo de videojuegos que crea contenido de manera algorítmica en lugar de manual. Esto permite generar automáticamente mapas, niveles, personajes y otros elementos del juego, proporcionando variabilidad y complejidad sin necesidad de diseñar cada detalle a mano.

Historia de Usuario (US): del inglés, User Story, una historia de usuario es una forma de describir qué quiere un usuario que haga una aplicación, de una manera corta y fácil de entender para todos en el equipo de desarrollo. Es como una pequeña historia sobre lo que alguien necesita.

Input maps: configuraciones que asignan las entradas de los dispositivos de control (teclados, ratones, gamepads) a acciones específicas dentro de un videojuego. Permiten personalizar y gestionar cómo las interacciones del jugador se traducen en el juego.

Interfaz de Programación de Aplicaciones (API): del inglés, Application Programming Interface, conjunto de funciones y protocolos que permiten a diferentes programas comunicarse entre sí. Facilita la integración y el intercambio de datos entre aplicaciones diversas.

Java SDK: conjunto de herramientas y bibliotecas proporcionadas por *Oracle* para desarrollar aplicaciones en el lenguaje de programación Java. Incluye el compilador, el entorno de ejecución y otros utilidades necesarias para la programación en Java. Dado que Android se basa en el lenguaje Java, este componente es necesario a la hora de desarrollar cualquier aplicación para estos dispositivos.

Kanban: metodología ágil para gestionar el flujo de trabajo mediante tarjetas visuales en un tablero. Cada tarjeta representa una tarea y se mueve a través de diferentes columnas que indican el estado del trabajo, facilitando la organización y el seguimiento del progreso.

Mínimo Producto Viable (MVP): del inglés, Minimum Viable Product, versión básica de un producto que incluye solo las características esenciales para que funcione. Se utiliza para probar la idea del producto en el mercado y obtener retroalimentación de los usuarios, lo que ayuda a mejorar y desarrollar el producto final con menor riesgo y costo.

Offline: capacidad de una aplicación o videojuego para funcionar sin necesidad de estar conectado a internet. Permite a los usuarios acceder a funciones y contenido sin depender de una conexión en línea.

Orientado a Objetos: paradigma de programación que organiza el software en objetos, cada uno con atributos (propiedades) y métodos (comportamientos). Facilita la reutilización, modularidad y mantenimiento del código.

Power-ups: en videojuegos, los power-ups son ítems que otorgan habilidades temporales o mejoras a los personajes del jugador, como invulnerabilidad, velocidad aumentada o poderes especiales, mejorando la experiencia de juego.

Prefab: abreviatura de *prefabricated*, es una plantilla que agrupa una colección de objetos y sus configuraciones, permitiendo su reutilización en diversas partes del juego. Los prefabs pueden incluir elementos como modelos 3D, scripts, sonidos y efectos visuales, todos configurados con propiedades específicas. Esta técnica simplifica la creación y manejo de múltiples instancias de objetos complejos, mejorando la eficiencia y consistencia en el desarrollo.

Product Owner (PO): rol en la metodología ágil, responsable de definir la visión del producto, priorizar el backlog y asegurar que el equipo de desarrollo entrega valor en cada iteración.

Prueba de Aceptación (UAT): del inglés, User Acceptance Testing, se realiza al final del desarrollo para verificar que lo que se ha construido es lo que el usuario quería y funciona como se esperaba.

Script: conjunto de instrucciones escritas en un lenguaje de programación o scripting que controla el comportamiento y la lógica en un videojuego. Se utiliza para definir acciones, eventos y reglas en el juego.

Single-player: videojuego diseñado para ser jugado por una sola persona. A diferencia de los juegos multijugador, el enfoque está en la experiencia individual del jugado.

Sprite: imagen 2D que representa un personaje, objeto o elemento gráfico en un videojuego. Se usa principalmente en juegos 2D para animar y visualizar diferentes entidades del juego.

Sprints: ciclos de trabajo cortos y repetitivos, generalmente de dos a cuatro semanas, durante los cuales se completa una parte específica del proyecto. Cada sprint comienza con una planificación y termina con una revisión y una retrospectiva.

Tilemap: matriz bidimensional que organiza pequeñas imágenes llamadas tiles para crear escenarios 2D. Cada celda del tilemap contiene un identificador que señala un tile específico, permitiendo construir niveles detallados y variados de manera eficiente mediante la reutilización de un conjunto reducido de gráficos. Los tilemaps son esenciales para diseñar entornos en juegos de plataformas, *RPGs* y otros géneros 2D, ya que optimizan el uso de memoria y simplifican la creación y modificación de los niveles.

Tileset: colección de pequeñas imágenes usadas para construir niveles en videojuegos. Cada imagen, llamada tile, representa una parte del entorno, como el suelo o las paredes. Se combinan para crear mapas detallados y optimizar el rendimiento del juego.

ANEXO

OBJETIVOS DE DESARROLLO SOSTENIBLE

Grado de relación del trabajo con los Objetivos de Desarrollo Sostenible (ODS).

Objetivos de Desarrollo Sostenibles	**Alto**	**Medio**	**Bajo**	**No Procede**
ODS 1. **Fin de la pobreza.**				**x**
ODS 2. **Hambre cero.**				**x**
ODS 3. **Salud y bienestar.**			**x**	
ODS 4. **Educación de calidad.**			**x**	
ODS 5. **Igualdad de género.**				**x**
ODS 6. **Agua limpia y saneamiento.**				**x**
ODS 7. **Energía asequible y no contaminante.**				**x**
ODS 8. **Trabajo decente y crecimiento económico.**				**x**
ODS 9. **Industria, innovación e infraestructuras.**		**x**		
ODS 10. **Reducción de las desigualdades**.		**x**		
ODS 11. **Ciudades y comunidades sostenibles**.				**x**
ODS 12. **Producción y consumo responsables.**				**x**
ODS 13. **Acción por el clima.**				**x**
ODS 14. **Vida submarina.**				**x**
ODS 15. **Vida de ecosistemas terrestres.**				**x**
ODS 16. **Paz, justicia e instituciones sólidas.**				**x**
ODS 17. **Alianzas para lograr objetivos.**				**x**

Reflexión sobre la relación del TFG/TFM con los ODS y con el/los ODS más relacionados.

El ODS 3 se centra en asegurar una vida sana y promover el bienestar para todos en todas las edades. La relación de “Ascent Tower” con este objetivo es indirecta y de bajo impacto porque el juego no aborda directamente la salud y el bienestar físico o mental de manera explícita. Sin embargo, a través del juego se pueden fomentar ciertos aspectos del bienestar mental, como la reducción del estrés y la mejora de habilidades cognitivas, dado que los videojuegos pueden ofrecer una forma de entretenimiento que puede ayudar a aliviar el estrés y fomentar el bienestar psicológico.

El proyecto “Ascent Tower” también contribuye indirectamente al ODS 4 (Educación de Calidad) al fomentar la adquisición de habilidades cognitivas y estratégicas a través de la resolución de problemas en el juego. A medida que los jugadores se enfrentan a diversos desafíos y enemigos en su ascenso por la torre, deben desarrollar y aplicar habilidades de pensamiento crítico, toma de decisiones y planificación estratégica. Estos procesos mentales son esenciales en la educación de calidad, ya que promueven el desarrollo cognitivo y el aprendizaje activo.

El ODS 9 se dedica a desarrollar infraestructuras resilientes, impulsar una industrialización inclusiva y sostenible, y promover la innovación. La relación de “Ascent Tower” con este objetivo es de nivel medio debido a varios factores explicados a continuación:

- **Innovación en el Desarrollo de Videojuegos:** El uso de Godot Engine y la implementación de técnicas avanzadas como la generación procedural de niveles representan innovaciones en el campo del desarrollo de videojuegos.
- **Metodología Ágil:** El empleo de metodologías ágiles para gestionar el desarrollo del juego fomenta prácticas modernas y eficientes en la industria tecnológica, contribuyendo a la innovación y mejora de procesos.

Por último, el ODS 10 se centra en reducir las desigualdades dentro y entre los países, promoviendo la inclusión y la equidad en todos los aspectos de la vida. "Ascent Tower" contribuye a este objetivo a un nivel medio al incorporar características de accesibilidad y personalización que aseguran que el juego sea inclusivo para personas con diversas necesidades y preferencias. Estas características están diseñadas para reducir las barreras y permitir que una mayor diversidad de jugadores disfrute y participen plenamente en el juego. A continuación, se enumeran algunas pautas que se han desarrollado en el videojuego:

1. *Accesibilidad para Personas con Movilidad Reducida:*
    - **Ajustes de Botones Personalizables:** "Ascent Tower" permite a los jugadores ajustar la configuración de los botones para que se adapten a sus necesidades individuales. Esta funcionalidad es crucial para personas con movilidad reducida, permitiéndoles configurar los controles de una manera que les resulte más cómoda y manejable.

- **Compatibilidad con Diferentes Dispositivos de Entrada:** El juego está diseñado para ser compatible con una variedad de dispositivos de entrada, incluyendo mandos de consolas de videojuegos y teclados. Esta flexibilidad asegura que los jugadores puedan elegir el dispositivo que mejor se adapte a sus capacidades físicas.

2. *Inclusión de Jugadores Zurdos y Preferencias Personales:*
   - **Opciones para Voltear la Posición de los Botones:** "Ascent Tower" ofrece la posibilidad de invertir la disposición de los botones, facilitando el juego a personas zurdas o a quienes prefieran esta configuración por comodidad. Esto asegura que todos los jugadores, independientemente de su dominancia manual, puedan tener una experiencia de juego óptima y personalizada.
   - **Interfaz de Usuario Adaptable:** El diseño del interfaz de usuario es adaptable, permitiendo ajustes que se alineen con las preferencias y necesidades individuales de cada jugador, promoviendo así una experiencia inclusiva y equitativa.

3. *Compatibilidad con Diversos Dispositivos:*
   - **Jugar con Mandos de Consolas y Teclados:** La capacidad de jugar "Ascent Tower" utilizando mandos de consolas o teclados conectados a dispositivos móviles amplía las opciones de accesibilidad. Esta flexibilidad permite a los jugadores seleccionar el método de entrada que les resulte más cómodo y accesible, eliminando barreras y fomentando una experiencia de juego inclusiva.

ASCENT TOWER

Índice

Presentación

Título del proyecto

Ascent Tower.

Concept

El rescate de una persona en apuros de un castillo está en tus manos, aunque el castillo parece que no tenga fin.

Metáfora

Avanzar por los pisos de la torre mientras derrotas a los enemigos.

Narrativa

Eres un guerrero el cual decides avanzar por los pisos de una torre debido a que has escuchado que hay una persona en peligro y deseas rescatarla y ponerla a salvo lo antes posible. Durante esta aventura te irás encontrando enemigos a los que tendrás que derrotar y tal vez te encuentres algún que otro guerrero que se quiera unir a tu aventura con el mismo objetivo que tú.

Objetivo del juego

Desarrollar los reflejos y velocidad mecánica en el movimiento de los dedos de las manos.

Mecánicas básicas

Moverse a izquierda o derecha, atacar, subir escaleras, abrir cofres, adentrarse por las puertas del castillo, consumir pociones, uso de amuletos, evitar ataques, comprar artefactos y mejorar artefactos.

Obstáculos

Enemigos, dificultad creciente, falta de oro, carencia de diamantes, escasez de armamento, ausencia de amuletos, insuficiencia de pociones.

Recursos

Diferentes armas para acabar con los enemigos, armaduras con la que protegerse de sus ataques, escudos para evitar ciertos ataques y amuletos y pociones para conseguir diferentes efectos secundarios deseados.

Formas en que se producen y consumen

Las armas podrás usarlas siempre que quieras en el momento que quieras y las armaduras realizarán su efecto de forma pasiva, al igual que el escudo, pero el escudo a diferencia de las armaduras no siempre te protegerá, dependiendo del escudo que lleves será más probable que te proteja y no recibas ningún tipo de daño o recibir el golpe y ya dependería de la armadura que llevases puesta y el enemigo que te está atacando para ver cuánto daño te realiza.

Por otra parte, los amuletos también se consumen de forma pasiva, una vez equipados, realizan la función propia del amuleto en cuestión. Sin embargo, las pociones tienen que ser consumidas por el propio jugador, teniendo este que hacer uso de ellas en el momento que más las necesite.

Concluye

Si consigues alcanzar a la persona en peligro, podrás ponerla a salvo y llevarla a un lugar seguro, pero tus ansias por seguir explorando la torre no se acabarán ahí. Sin embargo, si los enemigos te derrotan antes, tu aventura acabará y no habrás podido cumplir con tu principal objetivo.

Planificación

Equipo de trabajo

Javier Pérez Asensio: Desarrollador, diseñador de niveles, diseñador gráfico, planificador.

Mario Mocholí Rubio: Compositor.

Tareas

Llevadas a cabo en un contexto de desarrollo ágil a través de la página web HakenPlan, la cual facilita el desarrollo ágil en el contexto de los videojuegos gracias a un diagrama *Kanban* en la cual representar las historias de usuario y tareas que se han de realizar distribuidos por *Sprints* y *Backlog*.

- *Sprint* 0 (12/03/2024 – 26/03/2024):
    - Aprendizaje de HacknPlan. (3 horas y 30 minutos) – Javier
    - Versión preeliminar GDD. (8 horas) – Javier
    - Aprendizaje de Godot. (12 horas) – Javier
    - Mover al jugador en un entorno representativo. (17 horas) – Javier
        - Movimiento personaje. (4 horas) – Javier
        - *Sprites* del personaje. (3 horas) – Javier
        - Animaciones del personaje. (6 horas) – Javier
- *Sprint* 1 (26/03/2024 – 09/04/2024):
    - Vida. (2 horas y 30 minutos)
        - Quitar vida si se es golpeado. (2 horas) – Javier
        - Píxel art corazón. (30 minutos) – Javier
    - Escudo. (5 horas)
        - Probabilidad fija de actuar. (3 horas) – Javier
        - Píxel art del escudo. (1 hora y 30 minutos) – Javier
        - Efecto de sonido probabilidad acertada. (30 minutos) – Javier
    - Enemigos. (28 horas) – Javier
        - Generación de los enemigos. (10 horas) – Javier
        - Píxel art enemigos. (15 horas) – Javier
        - Máquina de estado enemigos. (3 horas) – Javier
- *Sprint* 2 (09/04/2024 – 23/04/2024):
    - Generación de siguiente nivel de piso del castillo. (13 horas) – Javier
        - *Spawnear* a los enemigos. (5 horas y 30 minutos) – Javier
        - *Spawnear* el terreno del siguiente piso. (6 horas) – Javier
        - Píxel art piso torre. (1 hora y 30 minutos) – Javier
    - Pasar siguiente piso (Escalera). (21 horas) – Javier

        - Generación de la escalera. (5 horas) – Javier
        - Indicador de ascenso por la escalera. (8 horas) – Javier
        - Mover mapa subiendo por la escalera. (4 horas) – Javier
        - Píxel art escalera. (1 hora) – Javier
        - Animación subir por la escalera. (3 horas) – Javier
- *Sprint* 3 (23/04/2024 – 07/05/2024):
    - Atacar. (6 horas y 30 minutos) – Javier
        - Atacar tras pulsar botón de acción. (3 horas) – Javier
        - Matar enemigos al atacar. (2 horas y 30 minutos) – Javier
        - Botón atacar. (1 hora) – Javier
    - Canciones. (2 horas) – Javier y Mario
        - Arreglar canciones y SFX ya metidos. (1 hora) – Javier y Mario
        - Preparar para menú de configuración. (1 hora) – Javier
    - Efectos de sonido. (3 horas y 30 minutos) – Javier
        - Elegir nuevos SFX. (1 hora y 30 minutos) – Javier
        - Introducir nuevos SFX respecto a los eventos correspondientes. (2 horas) – Javier
- *Sprint* 4 (07/05/2024 – 21/05/2024):
    - Mejorar enemigos. (18 horas y 30 minutos) – Javier
        - Añadir AI. (6 horas) – Javier
        - Vidas. (30 minutos) – Javier
        - Mostrar barra de vida restante. (1 hora) – Javier
        - Cambiar el color al ser golpeado. (1 hora) – Javier
        - Girar enemigo en caso de recibir ataque y no estar viendo al jugador. (1 hora) – Javier
        - Enemigo ataca a jugador. (2 horas) – Javier
        - Añadir diferentes enemigos. (7 horas) – Javier
    - Ser golpeado. (5 horas)
        - Cambiar el color al ser golpeado. (30 minutos) – Javier
        - Perder vida al ser golpeado. (30 minutos) – Javier
        - Fin de partida. (4 horas) – Javier
    - Economía. (9 horas)
        - Contador de oro. (2 horas) – Javier
        - Dropeo de oro. (3 horas) – Javier
        - Contador de diamantes. (1 hora) – Javier
        - Dropeo de diamantes. (3 horas) – Javier
- *Sprint* 5 (21/05/2024 – 04/06/2024):
    - Mejorar jugabilidad. (23 horas) – Javier
        - Modificar cámara. (2 horas) – Javier
        - El jugador podrá moverse libremente. (1 hora) – Javier
        - *Spawn* de enemigos en zonas no visibles. (2 horas) – Javier
        - Crear laberinto. (15 horas) – Javier
        - No se verá lo que hay en la parte superior del piso. (1 hora) – Javier

        - El mapa será más grande. (2 horas) – Javier
    - Resumen de partida. (6 horas) – Javier
        - Mostrar el tiempo de duración de la partida. (30 minutos) – Javier
        - Mostrar el oro conseguido. (30 minutos) – Javier
        - Mostrar los diamantes conseguidos. (30 minutos) – Javier
        - Mostrar la cantidad de pisos superados. (15 minutos) – Javier
        - En caso de ser un nuevo récord, se avisará al jugador. (45 minutos) – Javier
        - Mostrar la cantidad de enemigos derrotados. (1 hora) – Javier
        - Mostrar el número de ataques realizados. (1 hora) – Javier
        - Botón Reintentar. (15 minutos) – Javier
        - Botón Volver. (15 minutos) – Javier
        - Las estadísticas estarán ocultas (1 hora) – Javier
    - Ajustes. (6 horas y 30 minutos) – Javier
        - Añadir ajustes en pantalla de partida. (1 hora) – Javier
        - Añadir ajustes en menú principal. (1 hora) – Javier
        - Modificar máster. (1 hora) – Javier
        - Modificar música. (30 minutos) – Javier
        - Modificar SFX. (30 minutos) – Javier
        - Modificar botones de Pop-Up de acción. (2 horas) – Javier
        - Modificar botones de posición (modo zurdo). (1 hora) – Javier
    - Cofres. (7 horas) – Javier
        - *Spawn* de cofres. (4 horas) – Javier
        - Máximo un único cofre por piso. (2 horas) – Javier
        - SFX de abrir cofre. (1 hora) – Javier
- *Sprint* 6 (04/06/2024 – 18/06/2024):
    - Fleco: Crear salas. (2 horas) – Javier
    - Fleco: Modificar botones de pop-up de acción. (30 minutos) – Javier
    - Fleco: Modificar botones de posición (modo zurdo). (30 minutos) – Javier
    - Pantalla de inicio. (5 horas y 30 minutos) – Javier
        - Fondo de pantalla del menú principal. (1 hora) – Javier
        - Botón jugar. (1 hora) – Javier
        - Botón ajustes. (1 hora) – Javier
        - Botón salir. (2 horas) – Javier
        - Añadir canción de menú principal. (30 minutos) – Javier
    - Créditos. (1 hora y 30 minutos) – Javier
        - Crear pantalla de créditos. (45 minutos) – Javier
        - Botón volver. (45 minutos) – Javier
    - Tienda. (19 horas y 30 muntos) – Javier
        - Píxel art pociones (1 hora) – Javier
        - Botón volver. (30 minutos) – Javier
        - Botón ir a la tienda. (1 hora) – Javier

        - Creación de la escena de la tienda. (4 horas) – Javier
        - Pagar con oro. (2 horas) – Javier
        - Pagar con diamantes. (2 horas) – Javier
        - Pociones. (9 horas) – Javier
    - Mochila. (8 horas) – Javier
        - Creación de la escena de la mochila. (3 horas) – Javier
        - Botones de acceso rápido. (2 horas) – Javier
        - Formato de la bolsa en forma de rejilla. (3 horas) – Javier
- *Sprint* 7 (18/06/2024 – 30/06/2024):
    - Puesta en marcha. (11 horas) – Javier
        - Arreglo de fallos conocidos y mejoras. (8 horas) – Javier
        - Sacar el videojuego en itch.io. (3 horas) – Javier
    - Cambios o mejoras de la encuesta. (22 horas) – Javier
        - Añadir nuevas pociones. (5 horas) – Javier
        - Modificar interfaz del menú principal respecto al botón de la tienda. (2 horas) – Javier
        - Modificar icono de la tienda. (30 minutos) – Javier
        - Modificar botones de acción por defecto. (1 hora y 30 minutos) – Javier
        - Apertura de cofres más vistosa. (4 horas) – Javier
        - Tienda en partida. (8 horas) – Javier
        - Reducir el rango de ataque. (1 hora) – Javier
- *Backlog* (Ordenado por prioridad):
    - Armadura del jugador. (8 horas)
    - Dropeo de artefactos. (10 horas)
    - Puerta a sala interna al castillo. (15 horas y 30 minutos)
    - Pantallas de carga. (6 horas)
    - Pantalla de carga de inicio. (4 horas)
    - Logros. (15 horas)
    - Ventana. (7 horas y 30 minutos)
    - Carga asíncrona de las pantallas. (2 horas)
    - Ranking global. (20 horas)
    - Ranking amigos. (35 horas)
    - Idioma. (9 horas)
    - Cambio de personaje y artefactos. (30 horas)
    - Funcionalidad de salto. (8 horas)
    - Modalidad de juego Arquero. (25 horas)
    - Boss. (50 horas)
    - Temporal. (60 horas)
    - Reto semanal. (40 horas)
    - Animación inicio del juego. (25 horas)
    - Animación rescate persona en peligro. (30 horas)
    - Vincular cuenta *Play Games*. (7 horas)
    - Anuncios *AdMob*. (15 horas)

- Guerrero acompañante. (20 horas)
- Modalidad de juego: Mago. (20 horas)
- Elegir conjunto aleatorio. (12 horas)
- Expediciones de personajes. (15 horas)
- Eventos. (50 horas)

General

Género

El género de este videojuego sería *Action RPG* debido a que se trata de un videojuego donde los jugadores deben mejorar su equipamiento y enfrentarse a enemigos en un entorno de juego que fomenta la estrategia y la planificación cuidadosa, en caso de no ser precavidos, la probabilidad de fallo y derrota en el juego es mucho más elevada.

Plataformas

Imagen 1. Logotipo de sistema operativo Android.

La plataforma en la que se pueda jugar a este juego sería exclusivamente teléfonos móviles Android, incluidas tabletas, desde la versión de Android 5.1 hasta la última versión disponible.

Público objetivo

Este videojuego está pensado para que la gran parte de su público objetivo principalmente venga de los jugadores de videojuegos casual en Android.

Otra gran parte del público vendría de juegos similares o del mismo género los cuales también son exclusivos de dispositivos móviles y que tengan fama o sean bastantes conocidos como puede ser "*Tower of Hero*", para que el jugador de esta clase de juegos ya tenga una idea de qué es lo que se puede llegar a encontrar en este videojuego y, debido a sus gustos, desee empezar a jugar a este juego.

Un público que también sería atraído por este videojuego sería los jugadores de *RPG*s de consolas u ordenadores, en este caso, al ser un teléfono móvil más accesible que una consola u ordenador, si el juego llamase la atención a una persona de esta categoría, no habría mucho problema en probarlo y averiguar si es el videojuego que están buscando desde la comodidad de su bolsillo y con ello mantener el juego en su dispositivo jugando regularmente o si esperaban otra experiencia y eliminar el juego o jugarlo cada mucho tiempo.

Por último, se desearía que cualquier persona, ya tenga sus gustos en esta clase de videojuegos o no, le llame la atención este videojuego y llegue a jugarlo, aunque no sepa mucho de videojuegos o su interés no esté tan centrado en esta categoría.

Clasificación

Imagen 2. Logotipo de la clasificación PEGI 7.

Este juego estaría clasificado como PEGI 7 debido a que "El contenido del juego con escenas o sonidos que pueden atemorizar a los niños más pequeños debería incluirse en esta categoría. Las formas muy suaves de violencia (violencia implícita, no detallada o no realista) son aceptables para un juego con una clasificación PEGI 7." (Referencia).

Competencia

Imagen 3. Videojuego "*Tower of Hero*".

Como competencia directa nos encontramos el videojuego "*Tower of Hero*", este es un videojuego exclusivo para dispositivos móviles, el cual trata en ascender por una torre repleta de enemigos los cuales tienes que derrotar para poder avanzar al siguiente piso, este juego combina elementos de acción y estrategia incremental. En este juego, a medida que avanzas y derrotas enemigos, vas obteniendo oro que pueden usar para mejorar las habilidades de tus héroes, reclutar aliados, como pueden ser magos, guerreros, ninjas, entre muchos otros, y comprar objetos útiles en la tienda del juego. La jugabilidad de los héroes en este videojuego es completamente automática, el jugador solamente tiene que preocuparse en mejorar sus héroes con el oro obtenido al derrotar a los enemigos.

Imagen 4. Imagen ilustrativa del videojuego "*Sling Kong*".

Otro videojuego destacable por el parecido a su funcionalidad y a su exclusividad en dispositivos móviles es "*Sling Kong*", donde el objetivo de este videojuego es subir lo más alto posible mientras esquivas obstáculos recolectando monedas y *power-ups*. La mecánica de juego se basa en el tiempo y la precisión: los jugadores deben soltar al personaje en el momento adecuado para que pueda agarrarse a la siguiente plataforma y evitar caer al vacío. Este juego es de los desarrolladores "*Protostar*", siendo un estudio *indie*, los cuales tiene otro videojuego con una mecánica similar a este el cual se explicará a continuación.

Imagen 5. Ilustración del videojuego "*Super Starfish*".

Por último, un competente también destacable encontraríamos el videojuego "*Super Starfish*", este videojuego es de los mismos creadores que el anterior, "*Sling Kong*", siendo este también exclusivo para dispositivos móviles. El objetivo de este videojuego es viajar lo más lejos posible, en un recorrido infinito, mientras recolectas estrellas y esquivas obstáculos. Uno de los aspectos más destacados de "*Super Starfish*" es su diseño visual, además, la música atmosférica y los efectos de sonido contribuyen a crear una experiencia relajante y envolvente para los jugadores. La mecánica central del juego implica deslizar el dedo por la pantalla para guiar al pez estrella a través del espacio, evitando asteroides y otros peligros que pueden interponerse en su camino.

Diferencias destacables

Las diferencias respecto a la competencia directa, siendo en este caso "*Tower of Hero*", serían que en este juego, el jugador ha de ser el que dicte qué realizar en cada momento, es decir, el jugador tiene libertad total en decidir si es el momento de subir al siguiente piso, el moverse para esquivar un ataque de un enemigo, el atacar a un enemigo e intentar acabar con él o si prefiere quedarse en el piso actual para simplemente moverse para descansar un rato de recibir los ataques de los enemigos u observar los movimientos que realiza el jugador en cada escenario, entre muchas otras cosas. Este juego no tendría la modalidad de juego de recolección de recursos de manera automática como es el caso de "*Tower of Hero*".

Gracias a esta diferencia conseguimos una interacción del usuario mucho más profunda y cercana en el videojuego debido a que en todo momento es el usuario el que tiene que tomar una decisión sobre qué debería hacer y qué sería lo mejor para lograr sus objetivos, siendo este otro punto diferenciador, en "*Tower of Hero*" el principal objetivo es el de ascender por la torre sin importar nada más que conseguir el oro correspondiente para mejorar a tus actuales guerreros y conseguir aumentar aún más en los pisos de la torre. En este juego ese no es el único objetivo, aquí también nos encontramos el objetivo de ir en búsqueda de los cofres o enemigos brillantes que pueden aparecer de manera aleatoria a lo largo de una partida y con ello conseguir recompensas mucho mayores.

Otro gran punto diferenciador sería que en este videojuego solamente se tiene un único personaje jugable, a diferencia de "*Tower of Hero*" el cual tiene múltiples personajes, con habilidades diferentes los cuales tienes que mejorar para que sean más fuertes y así avanzar por los niveles de una forma más sencilla. Este único personaje tendría su armadura, escudo y arma, los cuales, estos artefactos, se podrían mejorar por separado para aguantar más golpes de los enemigos o poder hacer más daño.

Gameplay

Acciones

Como solamente hay un único personaje jugable, las acciones principales en el transcurso de cada partida van a ser siempre las mismas, estas acciones son las básicas de cualquier juego de este tipo, siendo estas las de movimiento y ataque.

En el caso de las acciones de movimiento, el jugador podrá desplazarse de forma horizontal con unos botones puesto a su disposición posicionados inicialmente en el lado izquierdo inferior de la pantalla (con posibilidad de modificación al lado derecho de la pantalla). El movimiento vertical en este videojuego estará limitado condicionalmente a si el jugador a conseguido derrotar a todos los enemigos del piso en el que se encuentre actualmente en el castillo, en caso de que haya logrado derrotarlos a todos, el jugador podrá avanzar al siguiente piso gracias al desbloqueo del ascenso por las escaleras.

Por otro lado, en el caso de la acción de ataque, esta estará dispuesto en el lado derecho inferior de la pantalla, y será un botón notoriamente más grande que los botones de movimiento. Este botón podrá ser accionado por el jugador en cualquier momento y, en caso de activarlo y encontrarse a un enemigo enfrente, este será golpeado y se le reducirá la vida correspondiente al daño del arma llevada en ese momento.

Por último, las acciones extra en este videojuego serán llevadas a cabo del uso de pociones o amuletos encontrados en el transcurso de una partida, estas nos pueden otorgar vida extra, recuperación de vida, mayor resistencia a golpes, aumento de las recompensas obtenidas, aumento de la probabilidad de obtención de mejores recompensas, aumento en el daño otorgado a los enemigos, entre muchas otras. En el caso de las pociones, estas solamente tendrán un único uso, en comparación a los amuletos que el efecto de estos será duradero conforme lo tengan equipado. El uso de las pociones y amuletos se podrá observar en la parte superior de la pantalla, donde tendremos un acceso rápido al uso de estos simplemente clicando sobre las pociones, el efecto de los amuletos se activa solamente poniéndolo en una de las tres ranuras disponibles.

Objetos

Durante una partida en el juego, nos iremos encontrando diferentes objetos coleccionables los cuales, en caso de tener espacio en nuestra bolsa, los iremos almacenando, estos son:

- **Oro**: Esta es la moneda principal del juego, con la que vas a poder comprar los objetos básicos y mejorarlos en la tienda.

- **Diamantes**: Esta piedra preciosa es muy valiosa en el reino debido a su dificultad de obtención, por eso mismo, los mercaderes van a tratarla con un valor superior, con esta moneda podrás comprarte objetos ya mejorados, objetos exclusivos, imposibles de adquirir con oro o intercambiarla por oro en el caso que sea necesario.

Armamento

El armamento también pertenecería al grupo de objetos coleccionables que se irían almacenando en la bolsa según tengamos espacio o no. El armamento disponible en el juego se corresponde al siguiente:

- **Armaduras**: Gracias a las armaduras nuestro personaje podrá recibir daño significativamente inferior dependiendo de qué tan buena y mejorada se encuentre la armadura. La armadura se subdivide en cuatro secciones, siendo estas el casco, la pechera, los pantalones y las botas. En caso de equiparse una armadura completa, es decir, los cuatro componentes del mismo tipo, se le proporcionará una bonificación al jugador dependiendo del tipo de armadura con la cual haya decidido realizar el conjunto.

- **Armas**: Sin un arma no se puede atacar, este es el componente básico si lo que se quiere es ser capaz de derrotar a los enemigos que nos vayamos encontrando en nuestra aventura, cada arma tendrá diferentes características, unas pueden realizar más daño, otras atacar más seguido y algunas atacar más lejanamente. Al igual que las armaduras, estas también se pueden mejorar, en este caso, para otorgar más daño a los enemigos.

- **Escudos**: La utilidad de los escudos sería la de poder esquivar los ataques de los enemigos, este objeto sería un objeto pasivo, al igual que las armaduras, es decir, el jugador no tiene que realizar nada para activar la función del escudo, este se activaría en algunas ocasiones cuando el jugador recibiese un golpe, dependiendo del nivel de mejora, la probabilidad de protección variaría. En caso de activarse el escudo al recibir un golpe, este no afectaría al jugador, es decir, no se le disminuiría el nivel de vida, reproduciendo un sonido de efecto especial para indicar que le golpe ha sido esquivado gracias al escudo.

Potenciadores

Tanto las pociones como los amuletos se colocan en el mismo sitio, teniendo tres espacios disponibles para colocar amuletos o pociones, en partida se puede modificar qué amuletos o pociones lleva el jugador, pero no se detendrá el tiempo, con lo cual, puede complicarse si se deseaba tener otra poción u otro amuleto asignado cuando se está luchando contra los enemigos de ese nivel.

- **Pociones**: Las pociones tendrán un único uso en toda su vida útil, una vez te lo encuentras, compras o mejoras, si lo usas, se consumirá y su efecto no podrá ser revertido hasta que no se finalice este. Los efectos de las pociones estarán relacionados con efectos temporales o inmediatos, siendo estos de recuperación de vida, poder recibir menor daño, realizar más daño, entre muchos otros efectos.

- **Amuletos**: Los amuletos son similares a las pociones, pero en este caso su uso será ilimitado y su efecto durará mientras se lleve el amuleto equipado. Los amuletos ocuparán un acceso rápido a una poción, pero tal vez su efecto sea más interesante que el de cualquier poción disponible. Los efectos de los amuletos son efectos constantes mientras se lleve equipado el amuleto, estos efectos estarán relacionados con aumento de recompensas como el oro, diamantes, aumentos de probabilidad de aparición de mejores artefactos, aumento del número de artefactos obtenidos, entre muchas otras opciones.

Efectos negativos

Este juego dispondría de muy pocos efectos negativos, solamente encontramos que el jugador, al recibir un golpe de un enemigo, en caso de no ser esquivado por el jugador, afectaría al jugador disminuyendo su nivel de vida, el nivel de disminución dependería del enemigo que le haya atacado.

Otro efecto negativo, en este caso secundario, sería que, en caso de que el jugador decida no mejorar o comprar mejor equipamiento a través de la tienda y continúe su aventura por los pisos del castillo, debido a la dificultad ascendente del videojuego, tarde o temprano no podrá seguir avanzando a menos que compre, mejore o cambie su equipamiento actual por una combinación más efectiva o mejores atributos, así se le facilitará más el juego y podrá seguir avanzando por lo diferentes niveles sin tanta dificultad.

Historia

Prólogo

Eres un caballero cualquiera, el cual, mientras realizas un paseo explorando tierras que nunca habías visto, te topas con una torre, al intentar mirar a lo alto de la torre, te das cuenta de que no logras verlo debido a una cantidad considerable de nubes ocultando el final, lo único que puedes ver es el cuerpo de la torre, que ya de por sí te parece impresionantemente alto, y las nubes tapando la punta de dicha torre.

Al cabo de un rato de estar observando y analizando el castillo te das cuenta de que de la cima de la torre puedes escuchar a alguien pidiendo ayuda a gritos desesperados, sin pensártelo dos veces, corres a adentrarte a la torre en búsqueda de aventuras emocionantes y ayudar a dicha persona que rogó por su vida.

Tramo

Dentro de la torre notas un ambiente muy oscuro y solitario donde, como sucedía desde fuera, tampoco eres capaz de ver qué hay más allá de los pisos donde te encuentras, sin embargo, esto no es un impedimento hacia tu curiosidad y decides avanzar y tratar de ascender por dicha torre, logras subir los escalones de la primera escalera con éxito hasta que te encuentras con ser que no debería estar ahí, la primera reacción de este, una vez te ha visualizado, es la de atacarte a pesar que tú inicialmente te mostrabas amigable, ahora mismo estás entre la vida o la muerte, dejarte ser atacado por dicha criatura e intentar llegar a un acuerdo o atacarla tú primero para no recibir ningún tipo de daño y con ello pasar el mal trago, decides optar por la segunda opción y finalmente eres el vencedor de esta batalla.

Esto te recuerda a la persona que suplicó que la rescatasen desde lo alto de la torre y te anima a seguir tu aventura en búsqueda de esa persona para rescatarla y ponerla a salvo, aún sin saber cuán larga será la aventura durante el ascenso de la torre.

Por cada piso que subes, encuentras nuevos y más enemigos, consigues poco a poco derrotarlos a todos, a veces son oleadas de tres enemigos, otras de dos, hay otras que de cinco e incluso de siete, tú único objetivo ahora mismo es lograr alcanzar a la persona en apuros y ponerla a salvo cuanto antes, sin importar cuántos enemigos derrotar o cuántos pisos asciendas.

Sin embargo, no todo son malas noticias, durante tu ascenso, en algunos pisos encuentras cofres, los cuales te otorgan una muy gran fortuna, eso sí, no podrás ver lo que hay dentro de ellos sin antes acabar con todos los enemigos de dicho piso, debido a que, si te ven usurpando sobre sus posesiones, se pueden llegar a poner más violentos, eso si alcanzas a tocar el cofre antes de que te ataquen.

También encontrarás pisos donde el mal no reine, si no que puedas descansar y tomarte un respiro, en algunos encontrarás mercaderes, los cuales, a cambio de oro o diamantes, te darán materiales necesarios si quieres continuar tu aventura sin problemas, en otros encontrarás puertas, las cuales te dan acceso al interior del castillo que, como verás al meterte por una de ellas, será un lugar de paz, sin enemigos y sin peligro, donde podrás ascender unos cuantos pisos sin preocuparte de nada más que de alcanzar cuanto antes a la persona en apuros.

Poco a poco irás avanzando, irás encontrando que cada vez te cuesta más avanzar, no es como al principio, aquí la cosa se va poniendo cada vez más y más difícil, mejorar y comprar mejores materiales en las tiendas será tu objetivo para avanzar más fácilmente, pero de vez en cuando accederás a salas donde solamente habrá un único enemigo inicialmente, pero este no es como los demás, este muestra todo su poder y será mucho más difícil que los anteriores, por suerte y gracias a tu valía consigues derrotarlo, dicho enemigo te otorga recompensas aún mucho más grandes las cuales te permitirán seguir y seguir avanzando por la torre.

Epílogo

Tras un muy largo ascenso por la torre y habiendo derrotado a innumerables enemigos por el camino, finalmente consigues alcanzar tu objetivo, encontrar a dicha persona para poder rescatarla y que pueda vivir sana y salva. Esta te agradece enormemente tu labor por dicho trabajo realizado por ella y serás recompensado debidamente a tu decisión tomada.

Sin embargo no todo acaba aquí, esa persona está a salvo, pero te das cuenta que la torre tiene aún más pisos y decides que tu aventura no finaliza aquí, pero primero deberás dejar a dicha persona en un lugar alejado de la torre para que no corra más peligro, de momento abandonas tu exploración en la torre y diriges a la persona a un su reino, esta te otorgará la prometida recompensa pero tú no te quitas la sensación de querer volver a la torre y seguirla explorando para resolver el misterio que oculta bajo sus muros, por esto mismo, vuelves al castillo a continuar por donde has dejado tu aventura y ver qué te depara el nuevo camino escogido.

Niveles de juego

Objetivo

El objetivo principal del juego es el de ascender lo máximo posible por la torre, debido a que es un juego infinito, nunca nos encontraremos con un final y siempre podremos seguir avanzando hasta que finalmente un enemigo acabe con nosotros.

Cuantos más pisos consigas avanzar, tu número de ascensos máximo podrá aumentar en caso de que lo superes, este es el punto clave de este juego, el de proponerse una meta personal de conseguir superar un número de pisos superados máximo y poder competir con tus amigos o mundialmente a ver quién es el que consigue llegar más lejos.

Retos a superar

No en todos los pisos se encuentra este reto, pero sí en la gran mayoría y es uno de los puntos más notorios del juego, este es el derrotar a los enemigos que te vayas encontrando según vayas ascendiendo por los diferentes pisos, este es el principal reto para superar de todo el juego y en cada piso, pero no siempre será acción y aventura.

También habrá puntos de estrategia donde en un piso no te saldrán enemigos, si no que tendrás la opción de comprar en la tienda del juego y aprovisionarte bien para continuar tu aventura y con ello, tu ascenso por el castillo.

Relación entre ellos

En cada reto superado nos iremos encontrando escaleras, las cuales nos guiarán al siguiente piso, donde tendremos que volver a derrotar a los enemigos correspondientes y con ello ascender por las escaleras de ese piso, así continuamente hasta que nuestra partida finalice debido a que hemos sido derrotados por algún enemigo.

Justificación con la historia

Debido a nuestro ansia en proteger a la persona en peligro, nuestro objetivo será el de ascender por el castillo hasta que hallemos a la persona y la pongamos a salvo, por eso tendremos que ir ascendiendo por las escaleras, para alcanzar nuevos pisos y con ello, poco a poco lograr acercarnos a nuestro objetivo, el de rescatar a dicha persona.

Nuevos personajes que aparecen

Durante nuestra aventura encontraremos pisos clave, donde habrá otros guerreros que hayan escuchado la llamada de auxilio de la persona en peligro y hayan ido en búsqueda de ella, pero que, debido a la dificultad de la torre, han decidido rendirse y retroceder en su aventura. Es aquí cuando os cruzáis y ambos pensáis que sois enemigos, lucharéis en una larga batalla hasta que finalmente uno de los dos sale vencedor.

Dicha batalla finaliza antes de acabar con la vida del desconocido guerrero, el cual, ruega que no le mates y que te ayudará a seguir tu aventura por el castillo. Pero ten cuidado, tú le has perdonado la vida, pero no esperes que todos sean como tú, tal vez si no le derrotas antes de que él lo haga, no piense igual que tú y decida que vencerte sea la mejor opción para evitar males mayores.

En caso de salir victorioso de la lucha, es aquí cuando este se unirá a tu equipo y te acompañará a lo largo de tu travesía, lucharéis juntos y se facilitarán las cosas, pero no te confíes, en cualquier momento puede decidir marcharse o incluso si los enemigos le atacan mucho puede llegar a perder su vida.

Personajes

Descripción

En este videojuego encontramos diferentes personajes tanto jugables como no jugables.

Como personaje jugable tenemos al **protagonista**, este es una persona cualquiera, al que le emociona vivir nuevas aventuras, superar los retos que se le interpongan y proteger y mejorar el bien estar de los de su alrededor. Esta persona es como otra cualquiera, de estatura media, pelo corto, sin ninguna habilidad destacable salvo su gran capacidad de atención y resistencia, con muchas ganas de aventura.

Por otro lado, los personajes no jugables son los siguientes:

- **Enemigos:** durante nuestra aventura por la torre del castillo nos iremos encontrando seres que no parecen ni reales, estos son fantasmas, esqueletos, *slimes*, ogros, orcos, y un largo etcétera. Estos lo único que desean es proteger el castillo a toda costa, sin importarles nada más, deberemos acabar con ellos si no queremos ser golpeados y que nos eliminen ellos a nosotros antes que nosotros a ellos.
- **Héroe perdido:** este es un personaje que se enteró que dentro del castillo había alguien en apuros y decidió en embarcarse en esta aventura tal y como lo hizo el protagonista, sin embargo, este no fue derrotado por los enemigos, pero tampoco llegó a cumplir su objetivo de proteger a la persona en apuros, este se quedó en un piso de la torre, sucumbido por la oscuridad y la soledad que esta emana. Cuando el jugador protagonista se acerca al héroe, inicialmente percibe al jugador como un enemigo más, pero más adelante se da cuenta de la idea que tenía en su cabeza era errónea.
  El héroe tiene características físicas y psicológicas muy similares al protagonista, este era otro héroe que compartía la misma mentalidad del protagonista.
- **Dragón:** enemigo muy poderoso muy difícil de ver, pero que, si te lo cruzas, no será tarea fácil librarte de él. Este es muy grande, puede llegar a superar tu altura cuatro veces, además expulsa fuego por su boca y tiene unas garras tan afiladas como cuchillas. Solo querrá quitarte de su vista, pero si consigues derrotarle tú a él, ten por seguro que estaría escondiendo algo muy grande para protegerlo con tanta ansia.
- **Vendedor:** durante nuestro recorrido no todo lo que encontremos será malo, también encontraremos vendedores, pero estos no serán vendedores comunes como los que encontraríamos en la villa, estos tienen forma de rana humanoide, estos solo nos querrán vender y ya habrán hecho limpieza del piso donde se encuentren para que el comprado pueda gastar la mayor cantidad de dinero posible en su tienda.

También nos encontramos el vendedor de la villa, este es un vendedor normal el cual ni si quiera sabe de la existencia de esa torre de la que tanto habla el jugador protagonista.

- **Herrero:** este solamente se dedica a su trabajo, fabricar nuevas piezas de metal como armas, armaduras y escudos, durante sus tiempos libres irá creando nuevas piezas que venderá en su propia tienda, pero también aceptará que le lleves armaduras, armas y escudos para mejorar. Esta persona es una persona que está físicamente muy fuerte, pero tal vez no quieras estar hablando mucho tiempo con él, debido a que solamente se centra en su trabajo.

Historia

El jugador protagonista escucha gritos pidiendo auxilio desde lo alto de la torre por donde está paseando, este decide adentrarse a la misma donde se encuentra con enemigos que nunca antes había visto, durante su aventura también encontrará otros héroes que tenían su misma idea, pero se quedaron a medio camino, y este intentará convencerle a toda costa para que se una a él y siga con el cometido que deseaba inicialmente.

Pese a todos los obstáculos, este personaje querrá seguir avanzando más y más por la torre, y no parará hasta que no encuentre a la persona deseada, poco a poco irá notándose más fuerte, lo cual le animará a seguir la aventura y con ello rescatar a la persona y cualquier otro héroe que se encuentre durante su ascenso.

Héroes

Los únicos héroes que encontramos en esta historia sería el jugador protagonista y el héroe perdido, el cual tenía tu mismo objetivo de rescatar a la persona que gritaba desconsoladamente por auxilio, pero este no llegó a lograrlo.

Enemigos

Los enemigos de los héroes serán los seres que se vaya encontrando por el ascenso del castillo, estos son desde fantasmas, esqueletos, *slimes*, ogros, orcos, arañas, hasta dragones, entre muchos otros.

Estos enemigos intentarán siempre que puedan defender la torre del castillo a toda cosa, sin importarles si su vida está en juego y lucharán hasta la muerte por protegerla.

Listado de enemigos:

- Rey:

Imagen 6. *SpriteSheet* de todas las animaciones de los reyes.

- Cerdo:

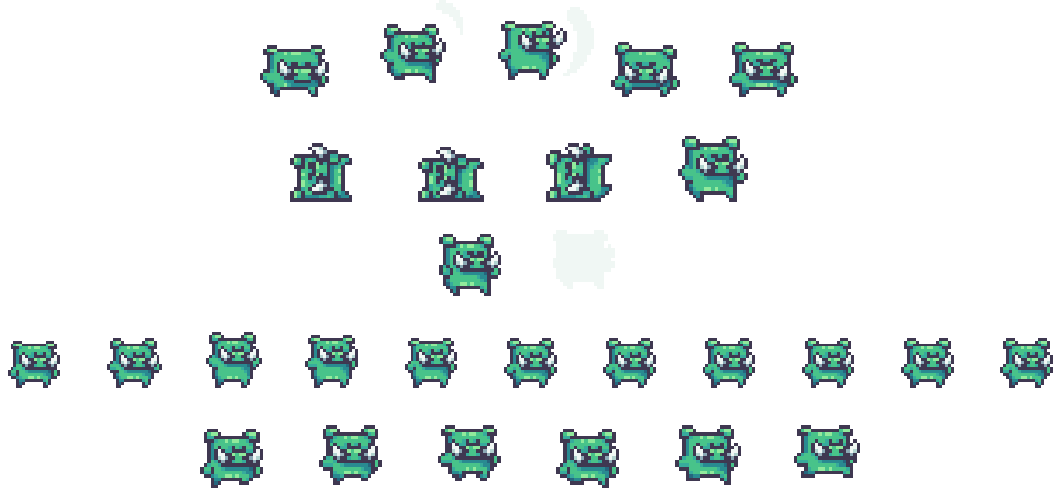

Imagen 7. *SpriteSheet* de todas las animaciones de los cerdos.

- *Slime*:

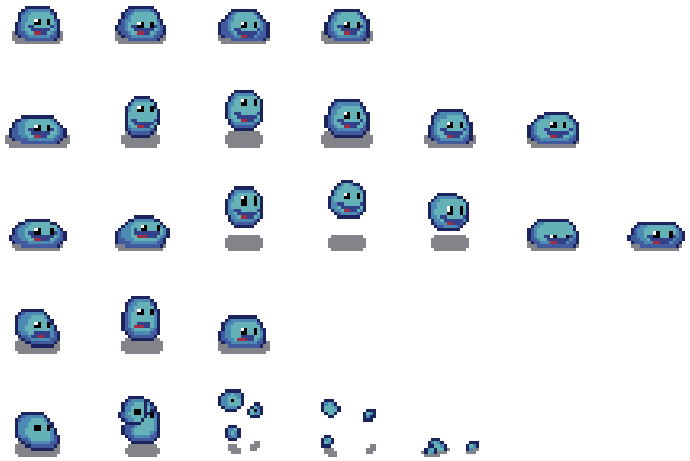

Imagen 8. *SpriteSheet* de todas las animaciones de los *slimes*.

- Esqueletos:

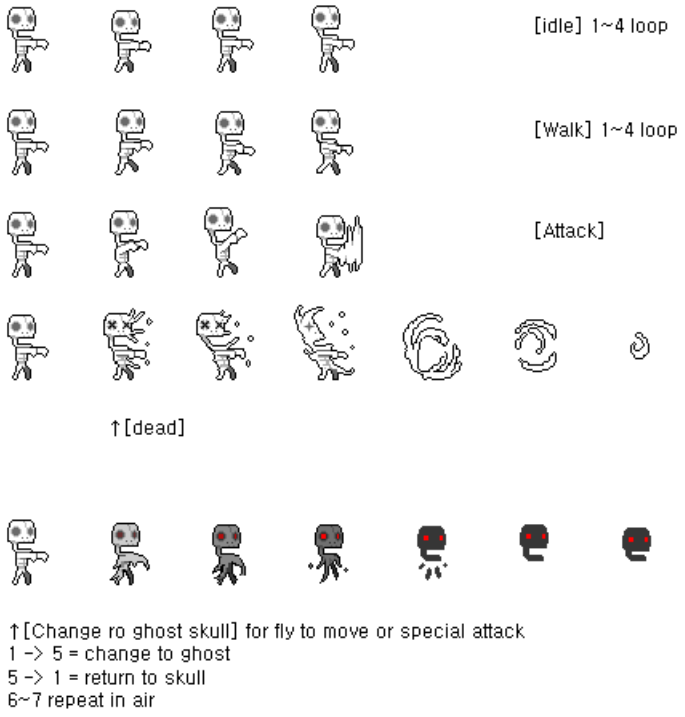

Imagen 9. *SpriteSheet* de todas las animaciones de uno de los esqueletos.

Imagen 10. *SpriteSheet* de todas las animaciones de otro de los esqueletos.

Las acciones de todos los enemigos estarán regidas por la siguiente máquina de estados:

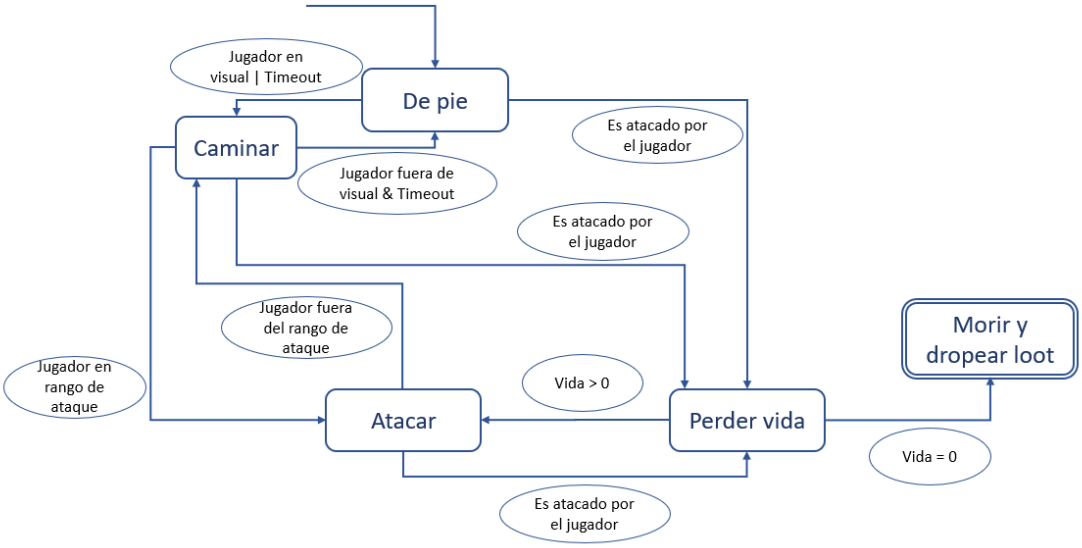

Imagen 11. Máquina de estados de los enemigos.

Donde aquí tenemos la explicación de los siguientes estados, eventos y transiciones:

Estados, eventos y transiciones

Estados	
Nombre	*Descripción*
De pie (Inicial)	El enemigo se encuentra de pie, respirando tranquilamente y esperando a la aparición de cualquier aventurero.
Caminar	Ha visualizado a un aventurero y se dirige a él para acabar con su vida.
Atacar	Intenta atacar al aventurero, para proteger la torre.
Perder vida	El enemigo pierde vida tras ser atacado por el aventurero.
Morir y dropear loot (Final)	Tras perder todas sus fuerzas (su vida), el enemigo queda derrotado y es posible recoger su recompensa.

Eventos	
Nombre	*Descripción*
Jugador en visual | Timeout	El enemigo tiene en visual al jugador o se ha acabado el timeout y se dirige a por él.
Jugador fuera de visual & Timeout	El enemigo pierde de vista al jugador y se ha acabado el timeout.
Jugador en rango de ataque	Una vez se ha acercado lo suficiente el enemigo al jugador, procede a atacarle.
Jugador fuera del rango de ataque	El jugador se aleja lo suficiente como para que el enemigo no intente atacarle, pero sí vaya tras él.
Es atacado por el jugador	En un descuido al enemigo le llega un ataque efectuado por el jugador.
Vida > 0	El enemigo ha recibido un ataque, pero sigue teniendo fuerzas para defender la torre.
Vida = 0	Al enemigo ya no le quedan fuerzas y se desploma junto a sus objetos.

Imagen 12. Explicación de los estados y eventos de la máquina de estados.

Eventos/Estados	De pie (Inicial)	Caminar	Atacar	Perder vida	Morir y dropear loot (Final)
Jugador en visual | Timeout	Caminar				
Jugador fuera de visual & Timeout		De pie			
Jugador en rango de ataque		Atacar			
Jugador fuera del rango de ataque			Caminar		
Es atacado por el jugador	Perder vida	Perder vida	Perder vida		
Vida > 0				Atacar	
Vida = 0				Morir y dropear loot	

Imagen 13. Transiciones de la máquina de estados.

Alianzas

El jugador podrá establecer alianzas con los héroes perdidos que vaya rescatando y animando a seguir su mismo objetivo, así, todos unidos podrán luchar para poder cumplirlo y cuantos más sean, más fácil será cumplir este cometido.

Objetivo

El objetivo principal que tiene el héroe protagonista es rescatar a la persona en apuros, pero también rescatará a todo aquél que se quiera poner de su parte y no quiera acabar con su vida, aunque dentro de una torre tan solitaria y oscura, con un aura tan extraño, será difícil encontrar a alguien así.

Elementos de juego

Armamento

Durante la aventura, el jugador, al derrotar a los enemigos que se vaya encontrando, irá encontrando y recolectando diferentes partes del armamento que se pueda equipar. Estas partes incluyen armas, escudos y armaduras, las cuales le otorgarán al jugador diferentes efectos dependiendo de qué sea lo que se equipe.

En el apartado de las armas nos encontramos con diferentes espadas, arcos y anillos mágicos, siendo las espadas ataques cuerpo a cuerpo y los arcos y anillos mágicos ataques a distancia.

Todas las armas tienen diferentes categorías, dependiendo de cuál se equipe, unas harán más daño, con otras se atacará más rápido, otras pesarán menos y te harán moverte más rápido, con otras atacarás más lejanamente, entre muchos otros estados, teniendo todas las armas puntos positivos y negativos.

Por otro lado, en el apartado de los escudos nos encontramos que estos nos protegen tanto de los ataques cuerpo a cuerpo como los ataques a distancia, este es un ser pasivo, el jugador no interactúa con este de ninguna forma.

Al igual que con las armas, también se encontrarán diferentes escudos los cuales tendrán mayor o menor probabilidad de actuación ante ataques físicos o de distancia, incluyendo también efectos como los de las armas, como puede ser que algunos escudos pesen menos que otros o que al ser atacados tengan una probabilidad de quemar, entumecer o electrocutar al enemigo, entre muchos otros.

Nuevamente todos los escudos tendrán tanto efectos positivos como efectos negativos y no habrá ninguno mejor que el otro, todo dependerá de los efectos que el jugador prefiera obtener.

Como último armamento encontramos las armaduras, estas también actuarán sin interacción del jugador y sus efectos serán completamente pasivos, pero a diferencia de los escudos, las armaduras tendrán porcentajes fijos en vez de probabilidades de eventos. Por ejemplo, al equiparse una parte de armadura nos puede aumentar la defensa un 20% y esa subida será permanente y constante respecto a un valor fijo preestablecido en el jugador, las únicas formas de modificar este valor son cambiando de armadura o mejorándola en el herrero.

En el apartado de las armaduras encontramos que se dividen en cuatro secciones, siendo estas el casco, la pechera, los pantalones y las botas.

En el caso de las armaduras, a parte de las subidas fijas de cada complemento que pueda tener la armadura, si se completa un grupo del mismo tipo, siendo un grupo las cuatro partes de la armadura, se darán bonificaciones extra, exclusivas de cada

grupo de armaduras, algunas darán más vida, otras más defensa, otras más ataque, entre muchos otros efectos, esto dependerá del tipo de armadura completa que se equipe.

Coleccionables

Durante la aventura se encontrarán muchos objetos coleccionables, entre ellos está el armamento, explicado en el apartado anterior, estos se podrán obtener al derrotar a enemigos y al abrir cofres. Estos podrán ser equipados para obtener diferentes bonificaciones y mejorados para aumentar dichas bonificaciones.

Al derrotar enemigos, estos también soltarán oro y diamantes, los cuales son muy útiles para ser usados en la villa, concretamente en el herrero para mejorar nuestro armamento y en la tienda para comprar nuevos artefactos.

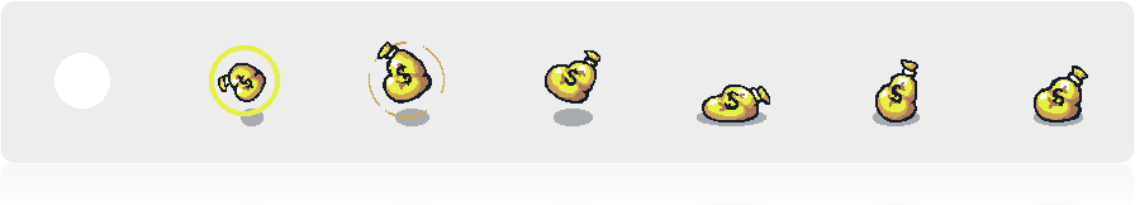

Imagen 14. Animación en *frames* del *spawn* de la bolsa de monedas (se le ha puesto fondo para no perder el primer *frame*).

Imagen 15. Bolsa de monedas en estado *idle*.

Otros elementos que podemos obtener al derrotar enemigos o abrir cofres son las pociones y los amuletos, estos nos aportan diferentes habilidades y aumentos de probabilidades, los cuales estarán explicados en cada frasco de pociones, siendo estos elementos consumibles, o efectos pasivos que se tendrán que equipar con los amuletos, los cuales por mucho uso que se les dé, nunca se desgastarán.

Para almacenar todos estos elementos coleccionables, el jugador dispondrá de una mochila con espacio limitado, dependiendo de qué mochila esté llevando y

cuán mejorada se encuentre esta, se podrán llevar más o menos artefactos en una misma aventura antes de venderlos o usarlos como mejoras en la villa o ser usados en partida.

Por último, también se podrán obtener corazones al derrotar enemigos los cuales nos regenerarán vida en caso de que no la tengamos ya al máximo.

Fondos

En el juego encontraremos principalmente dos fondos destacables, uno fuera del castillo, antes de entrar a este y otro dentro del castillo mientras se juega una partida.

En el fondo de fuera del castillo encontramos una pradera llena de césped muy amplia con diferentes desniveles, encontrándose a lo lejos y en mitad la torre, donde solamente se verá su inicio, con una puerta por donde entrar, y el cuerpo de la torre, no se podrá ver el final de esta.

En el fondo de dentro del castillo, lo que se puede apreciar es su estructura interna de ladrillos con los diferentes suelos de cada piso que contiene la torre, los ladrillos de las paredes tienen un color más oscuro que los representados por los suelos. Todos los suelos estarán conectados entre sí con el piso de arriba mediante una escalera de madera.

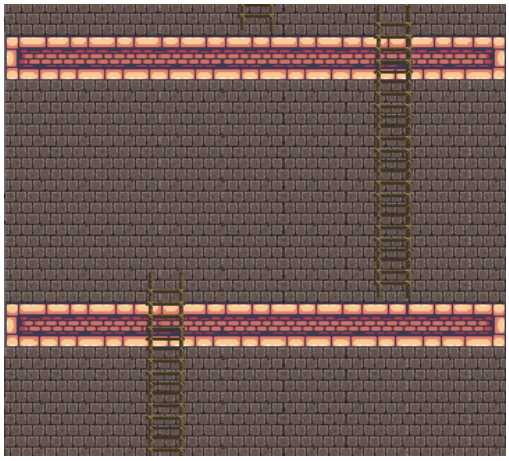

Imagen 16. Interior del castillo con el fondo de ladrillos, los suelos de los diferentes pisos conectados por la escalera de madera.

Objetos decorativos

Como objetos decorativos, durante la aventura, se encontrarán antorchas, ventanas, banderas, banderilles, emblemas, puertas, cofres, escaleras y ladrillos rotos o deteriorados.

Algunos de estos objetos son interactuables, como puede ser las escaleras para ascender al siguiente piso de la torre, las ventanas, por donde nos podremos caer o se podrá introducir un rayo en tomporales de tormenta, las puertas, por las cuales nos podremos adentrar al castillo pudiendo subir pisos más tranquilamente hasta un cierto nivel y los cofres, que podremos abrirlos para conseguir mayores recompensas. El resto de objetos son simplemente decoración para el juego.

A continuación se encuentras los diferentes sprites, tilemaps y animaciones utilizadas a lo largo del juego.

Imagen 17. *TileMap* utilizado para los objetos decorativos y fonde del castillo.

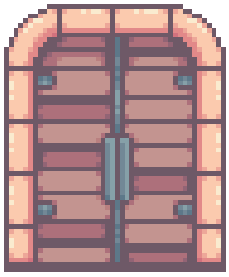

Imagen 18. Otra puerta del castillo en estado Idle.

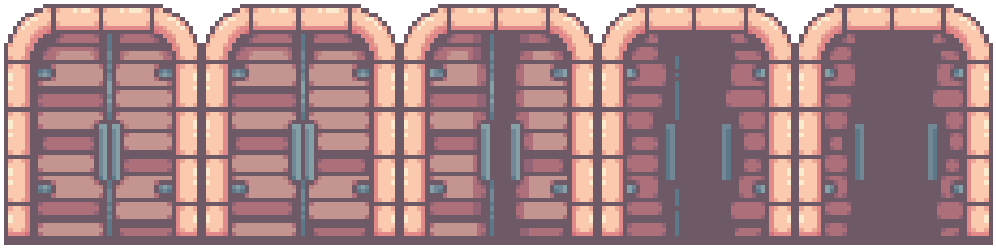

Imagen 19. Puerta del castillo abriéndose.

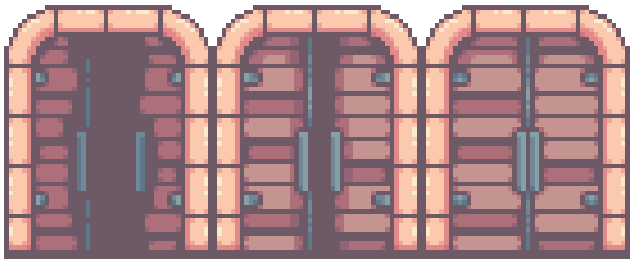

Imagen 20. Puerta del castillo cerrándose.

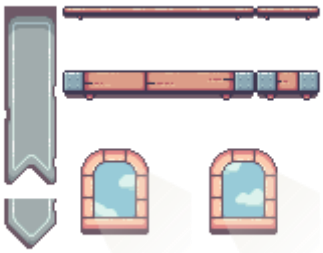

Imagen 21. Otro *TileMap* para las ventanas del castillo.

Imagen 22. Animación del cofre abriéndose.

Interacción

Diagrama de navegación entre pantallas de menú

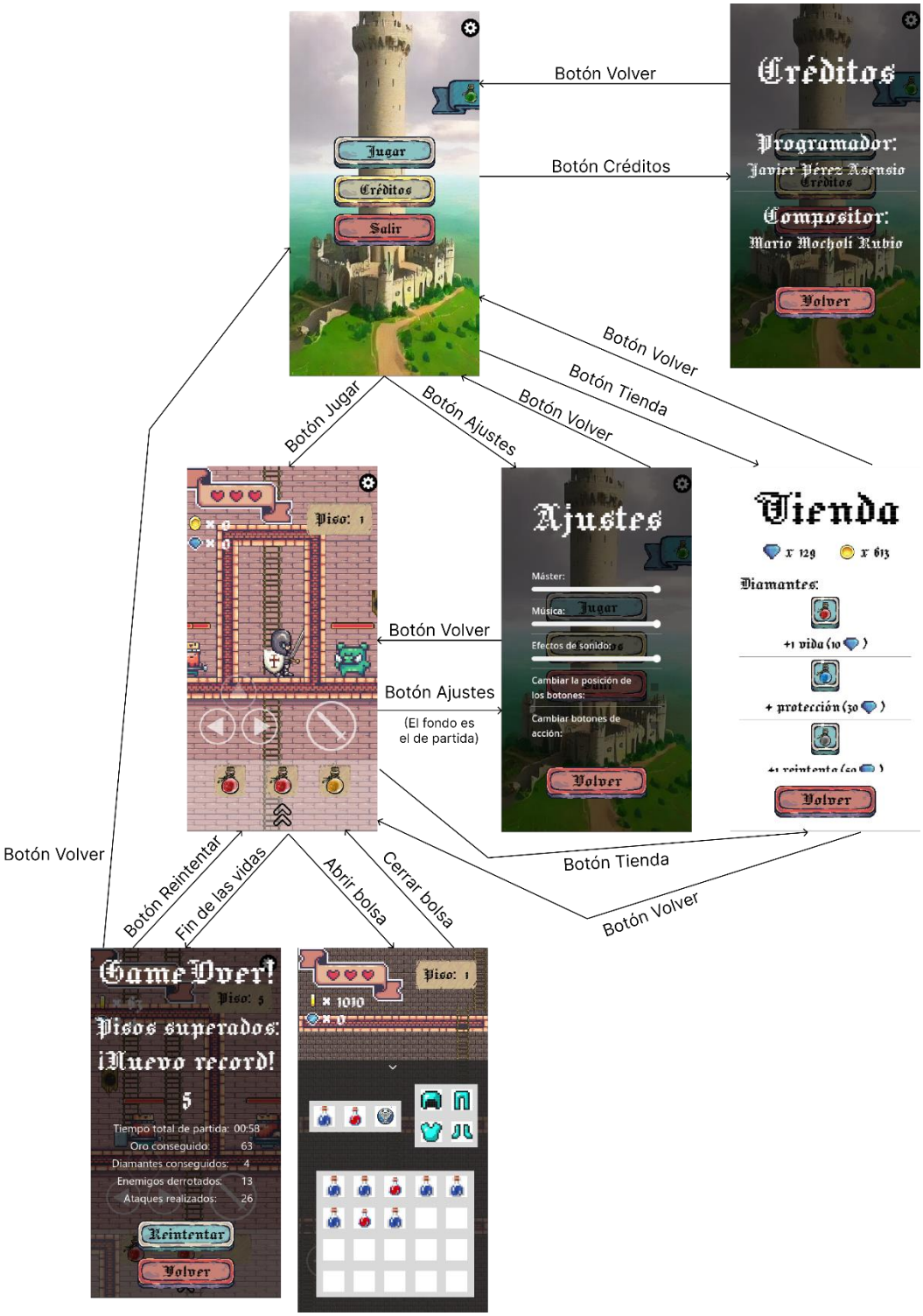

Imagen 23. Diagrama de navegación entre las diferentes pantallas de Ascent Tower.

Punto de vista de la cámara

Al tratarse de un videojuego 2D, se opta por el uso de cámara más común actualmente en esta categoría de videojuegos. Esta es el tipo de cámara paralela a la visión del juego, con movimiento vertical, simulando un seguimiento al jugador, en tercera persona, distribución de 2 capas, variando el ángulo de perspectiva entre fondo y el personaje jugable.

Mapa de teclas

En este caso, al tratarse de un videojuego exclusivo para dispositivos táctiles Android, se jugará con unos botones dispuestos en la pantalla los cuales el jugador tendrá que clicar en caso de querer realizar cualquier acción.

Estos botones serán de dos tipos:

- Estáticos en pantalla: donde encontraríamos las acciones básicas, estas son de movimiento y de ataque. Estos botones se encontrarían siempre en la parte inferior de la pantalla, estando los de movimiento al lado izquierdo y el de ataque en el lado derecho. Estos botones tienen la posibilidad de ser invertidos desde el menú de ajustes para jugadores zurdos o por mayor comodidad.
- Dinámicos: los cuales irían apareciendo en pantalla en forma de pop-up en función de las acciones del usuario o de su necesidad. Por ejemplo, el botón el cual te permite subir por la escalera para alcanzar el siguiente piso no aparecería hasta que el jugador haya derrotado a todos los enemigos en el piso actual, lo mismo ocurre con los cofres y con las puertas. Este botón es posible convertirlo en estático con un botón auxiliar, el cual se colocaría por encima de los botones de movimiento, este estaría desactivado a menos que se encuentre un objeto interactuable como son las puertas, los cofres y las escaleras las cuales, al pulsar sobre este botón, haría el mismo efecto que si se pulsara el botón pop-up emergente (al tener esta opción activa, el pop-up ya no se mostraría por pantalla).

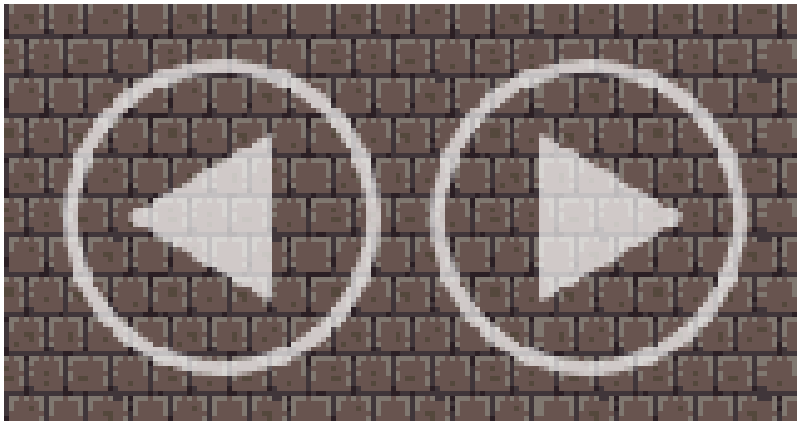

Imagen 24. Botones estáticos de movimiento.

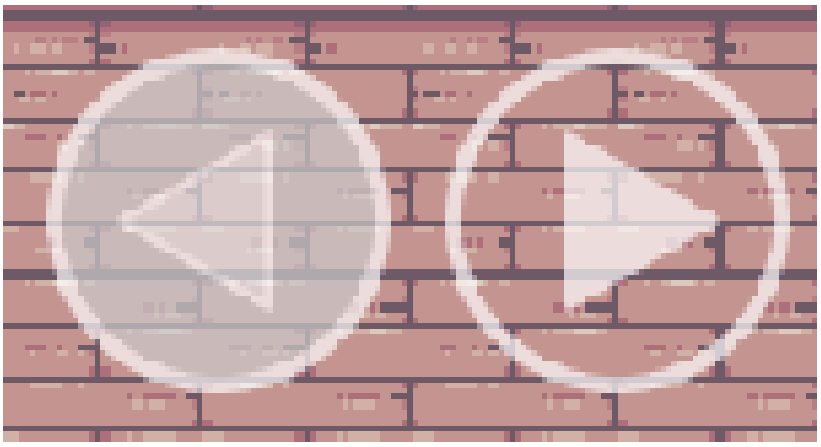

Imagen 25. Botones estáticos de movimiento con el botón izquierdo pulsado.

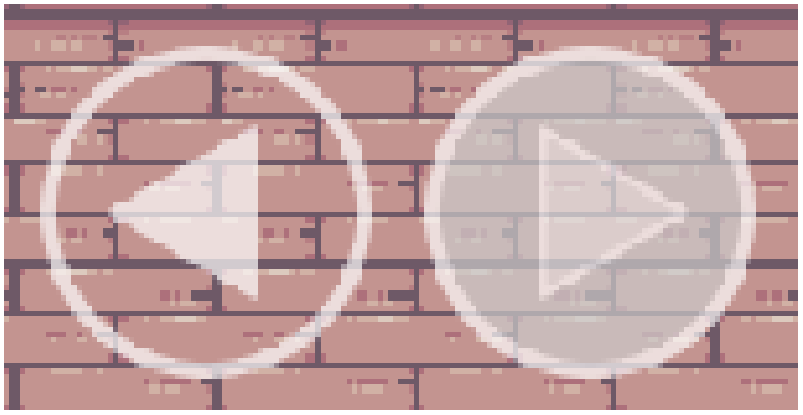

Imagen 26. Botones estáticos de movimiento con el botón derecho pulsado.

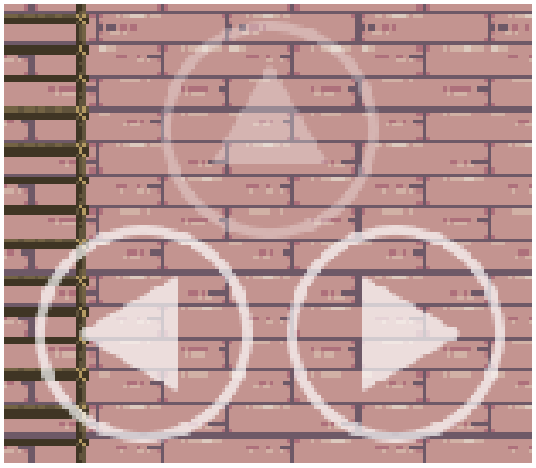

Imagen 27. Botón estático de evento deshabilitado.

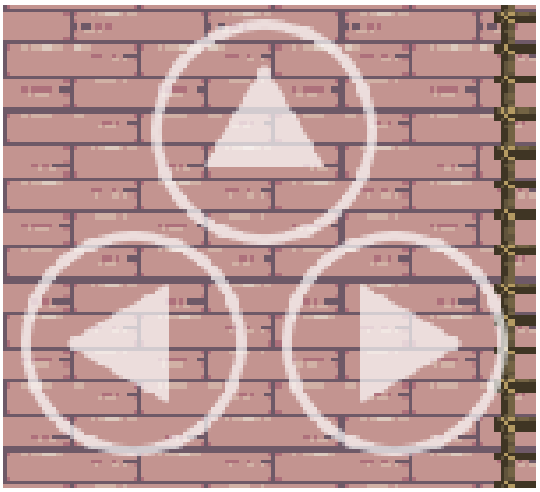

Imagen 28. Botón estático de evento habilitado.

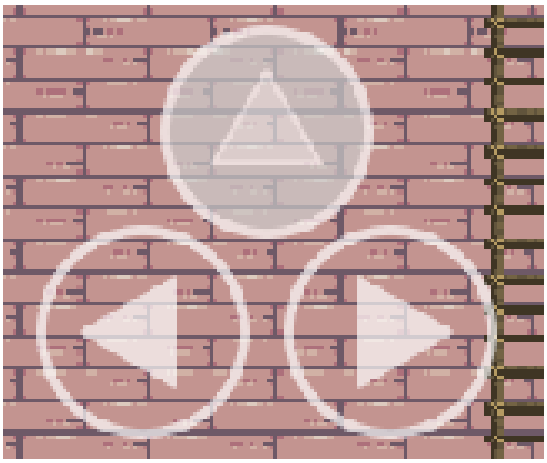

Imagen 29. Botón estático de evento pulsado.

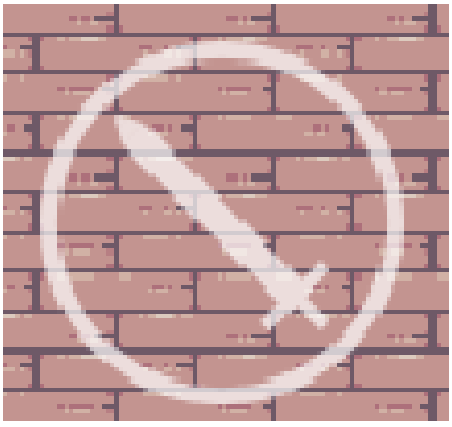

Imagen 30. Botón estático de ataque.

Imagen 31. Botón estático de ataque pulsado.

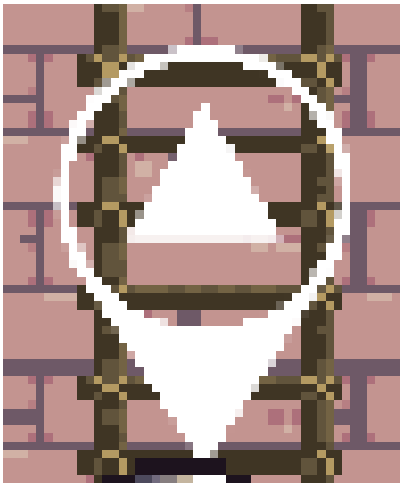

Imagen 32. Botón dinámico pop-up de evento.

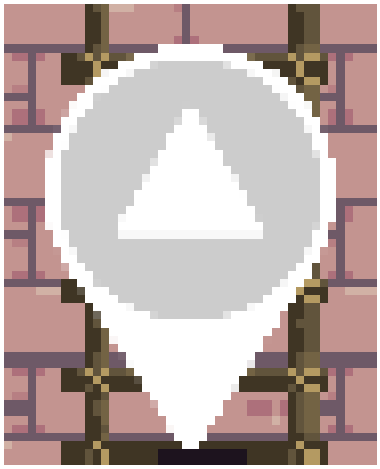

Imagen 33. Botón dinámico pop-up de evento pulsado.