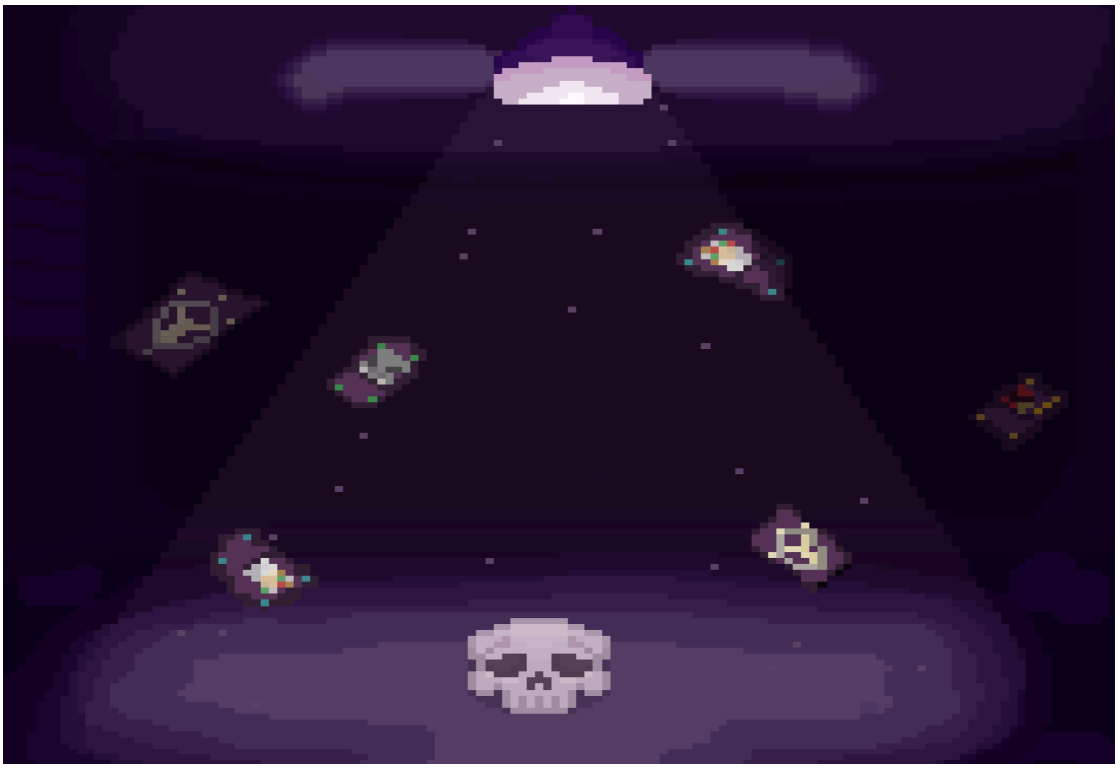
Institut Puig Castellar
Santa Coloma de Gramenet

Deck | Aaron Pérez Vargas, Jorge Rocano Da Silva

TENEBRIS

Proyecto de desarrollo



Miembros:

Jorge Rocano Da Silva | Aaron Pérez Vargas

Curso: 2SMX

Grupo: D

Tutor: Yago Morales ❤️

https://github.com/AaronPeva/Tenebris

Deck | Aaron Pérez Vargas, Jorge Rocano Da Silva

Deck | Aaron Pérez Vargas, Jorge Rocano Da Silva

Resumen del proyecto

Nuestro proyecto consiste en desarrollar nuestro propio videojuego con el motor de videojuegos Godot, utilizando el lenguaje de programación GDScript, diseñado específicamente para este entorno.

Para facilitar el trabajo colaborativo, implementamos Git como sistema de control de versiones y utilizamos GitHub como plataforma para alojar y coordinar repositorios.

Esta combinación nos permitió integrar el trabajo individual de cada miembro, organizar el código eficientemente y mantener copias de seguridad constantes, lo que optimizó la sincronización y el seguimiento del progreso.

Ante el reto de crear un videojuego completo en menos de un año, optamos por dividir las tareas entre los miembros del equipo, trabajando individualmente y unificando el progreso con Git. Esta metodología nos permitió avanzar de forma más ágil y eficiente, aprovechando al máximo las herramientas de desarrollo colaborativo.

Palabras clave:

#Videojuego #Cartas #Godot #Indie #Juego de cartas

Deck | Aaron Pérez Vargas, Jorge Rocano Da Silva

Project Abstract

Our project consists of developing our own video game using the Godot Engine, using the GDScript programming language, designed specifically for this environment.

To facilitate collaborative work, we implemented Git as a version control system and used GitHub as a platform for hosting and coordinating repositories.

This combination allowed us to integrate the individual work of each member, organize the code efficiently and maintain constant backups, which optimized synchronization and progress tracking.

Given the challenge of creating a complete video game in less than a year, we opted to divide the tasks among the team members, working individually and unifying progress with Git. This methodology allowed us to move forward in a more agile and efficient way, taking full advantage of collaborative development tools.

Keywords:
#Videogame #Cards #Godot #Indie #Card game

https://github.com/AaronPeva/Tenebris

Deck | Aaron Pérez Vargas, Jorge Rocano Da Silva

Índex

Deck | Aaron Pérez Vargas, Jorge Rocano Da Silva

1. Introducción

- Motor de desarrollo y lenguaje de programación

Nuestro proyecto consiste en crear un videojuego desde cero utilizando el motor de desarrollo gratuito y de código abierto Godot Engine, empleando el lenguaje de programación GDScript, el cual presenta ciertas similitudes con Python y está diseñado específicamente para integrarse con este motor.

Muestra de estructura de GDScript

```
23  func _on_atacar_pressed() -> void:
24      print(Global.attack_cost)
25      var final_damage = await damage_prob()
26      if not fallado:
27          AnimacionDolido()
28          attack_sound.play()
29      if critico:
30          critical_sound.play()
31      critico = false
32      fallado = false
33      boss_health.value -= final_damage
34      await get_tree().create_timer(2.0).timeout
35      AnimacionAtacar()
36      await get_tree().create_timer(0.4).timeout
37      var damage_received = damage_boss_prob()
38      Global.hp_carta_jugador -= damage_received
39      if Global.hp_carta_jugador < 0:
40          Global.hp_carta_jugador = 0
41
42  func _on_saltar_pressed() -> void:
43      await get_tree().create_timer(2.0).timeout
44      AnimacionAtacar()
```

Deck | Aaron Pérez Vargas, Jorge Rocano Da Silva

- Mecánica principal y enfrentamiento estratégico

El juego será un videojuego de cartas en el que el jugador se enfrentará a una entidad poderosa con características de jefe o “boss”, diseñada para ofrecer un desafío estratégico. Durante la partida, el jugador deberá planificar cuidadosamente sus jugadas para superar a esta entidad, aprovechando las mecánicas del juego y la estructura de su mazo de cartas. La dificultad estará orientada a fomentar la toma de decisiones tácticas y la adaptación a las condiciones cambiantes del combate.

Vista del “BOSS” o “BOT”

Deck | Aaron Pérez Vargas, Jorge Rocano Da Silva

- Narrativa y estilo visual

Además del aspecto estratégico, el juego también contará con una historia de fondo y una atmósfera oscura, lo que se reflejará tanto en su narrativa como en su estética. Hemos optado por un estilo tétrico en pixel art, con una paleta de colores dominada por tonos morados, que contribuye a crear una ambientación inquietante y envolvente.

Imagen del menú del juego

- Organización del trabajo y control de versiones

Dado el tiempo limitado del desarrollo, estamos trabajando de forma individual pero sincronizados mediante el uso de GIT, un sistema de control de versiones que nos permite llevar un seguimiento detallado de los cambios en el código, mantener copias de seguridad y coordinar el progreso de manera eficiente.

https://github.com/AaronPeva/Tenebris

1.1 Contexto

- La evolución de los videojuegos

Los videojuegos han crecido significativamente durante las últimas décadas, hoy en día no solo es el entretenimiento de muchos, sino que también te pueden enseñar algo o bien contarte una historia.

- Influencia de los juegos de cartas

Dentro del inmenso mundo de los videojuegos, los videojuegos de cartas siempre han estado presentes desde los inicios hasta hoy, como por ejemplo los grandes títulos como Pokémon TCG Pocket o Balatro, que están logrando un alto nivel de popularidad últimamente.

Imagen del videojuego Balatro

Deck | Aaron Pérez Vargas, Jorge Rocano Da Silva

- Inspiración para nuestro proyecto

El videojuego está inspirado entre el título “Buckshot Roulette”, que aunque no sea un videojuego de cartas, nos inspiró en darle ese toque tétrico o turbio, y “Pokémon TCG Pocket”, el cual nos inspiró en hacer un juego de cartas entretenido.

Imagen del videojuego Pokémon TCG Pocket

- Una historia oscura e inmersiva

Siguiendo el estilo oscuro del juego, la historia no iba a ser menos, imagina que eres secuestrado por un grupo de desconocidos que te arrastran a un escenario siniestro donde tu única oportunidad de sobrevivir es jugar un juego de cartas. Cada partida que juegas es una apuesta de vida o muerte mientras eres el entretenimiento para unos psicópatas, observando cada movimiento, disfrutando de tu sufrimiento mientras el destino de tu vida depende de unas cartas. Tendrás que usar tus mejores estrategias y una pizca de suerte para sobrevivir.

1.2 Justificación

Siempre hemos tenido la espinilla de crear un videojuego, un sueño que nos ha acompañado durante años, alimentado por nuestra pasión por los juegos y la creatividad. Sin embargo, hasta ahora no habíamos encontrado el momento adecuado para hacerlo. Ya fuera por falta de tiempo, conocimientos o simplemente por no haber dado el primer paso.

Sabemos que no será un camino fácil. Desarrollar un videojuego desde cero implica enfrentarse a obstáculos técnicos, creativos y organizativos, pero estamos convencidos de que esta es una gran oportunidad para aprender, crecer y demostrar de lo que somos capaces.

Sin embargo, para nosotros, cumplir este reto no será solo una victoria técnica, sino también un logro personal. Más allá del resultado final, lo que más valoraremos será el esfuerzo invertido, las dificultades superadas y la satisfacción de haber construido algo único con nuestras propias manos.

1.3 Objetivos

1.3.1 Objetivo general

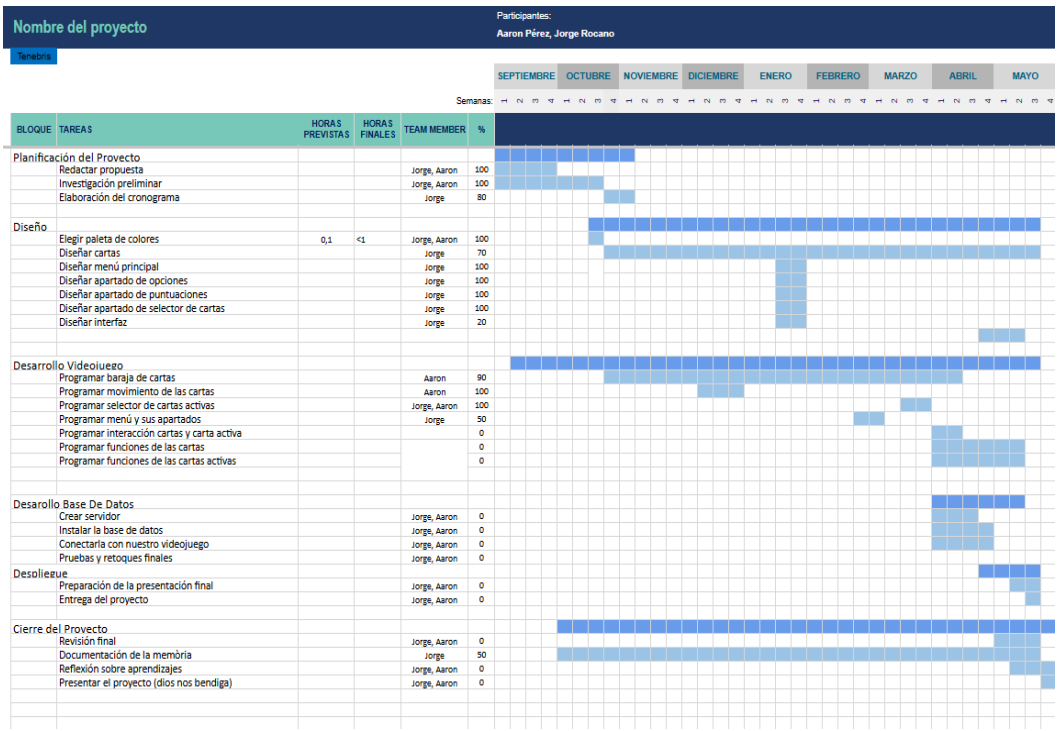
El objetivo general del proyecto consiste en desarrollar un videojuego de cartas utilizando el motor de desarrollo 'Godot Engine' y el lenguaje de programación 'GDScript'. La meta es no solo crear un producto funcional y entretenido, sino también ofrecer una experiencia única con mecánicas de juego estratégicas que desafíen al jugador. Además, se pondrá un énfasis especial en el diseño gráfico, buscando lograr una estética visual atractiva y coherente con la atmósfera del juego.

1.3.2 Objetivos Específicos

- Diseñar y pensar una idea original para el videojuego, incluyendo una narrativa, personajes y mecánicas del juego.
- Programar todas las funciones básicas del juego.
- Crear e integrar elementos artísticos, en este caso, el estilo pixel art, animaciones y efectos visuales, que se adapten al estilo y ambientación del juego.
- Desarrollar una interfaz de usuario atractiva.
- Implementar una conexión con una base de datos para gestionar la puntuación de los jugadores.
- Probar el videojuego para asegurar que el producto final es estable y jugable.

1.4 Estratègia i planificación del proyecto

Para la planificación del proyecto utilizamos un diagrama de Gantt con el objetivo de organizarnos mejor. Sin embargo, al final no resultó muy útil, ya que muchas tareas surgían de forma imprevista y era complicado seguir una planificación fija. Es prácticamente imposible planear algo que no sabemos hacer, por lo que la mayoría del trabajo se resolvió improvisando y aprendiendo sobre la marcha.



Nombre del proyecto

Participantes:
Aaron Pérez, Jorge Rocano

Tenebris

	SEPTIEMBRE	OCTUBRE	NOVIEMBRE	DICIEMBRE	ENERO	FEBRERO	MARZO	ABRIL	MAYO
Semanas:	1 2 3 4	1 2 3 4	1 2 3 4	1 2 3 4	1 2 3 4	1 2 3 4	1 2 3 4	1 2 3 4	1 2 3 4

BLOQUE	TAREAS	HORAS PREVISTAS	HORAS FINALES	TEAM MEMBER	%
Planificación del Proyecto					
	Redactar propuesta			Jorge, Aaron	100
	Investigación preliminar			Jorge, Aaron	100
	Elaboración del cronograma			Jorge	80
Diseño					
	Elegir paleta de colores	0,1	<1	Jorge, Aaron	100
	Diseñar cartas			Jorge	70
	Diseñar menú principal			Jorge	100
	Diseñar apartado de opciones			Jorge	100
	Diseñar apartado de puntuaciones			Jorge	100
	Diseñar apartado de selector de cartas			Jorge	100
	Diseñar interfaz			Jorge	20
Desarrollo Videojuego					
	Programar baraja de cartas			Aaron	90
	Programar movimiento de las cartas			Aaron	100
	Programar selector de cartas activas			Jorge, Aaron	100
	Programar menú y sus apartados			Jorge	50
	Programar interacción cartas y carta activa				0
	Programar funciones de las cartas				0
	Programar funciones de las cartas activas				0
Desarollo Base De Datos					
	Crear servidor			Jorge, Aaron	0
	Instalar la base de datos			Jorge, Aaron	0
	Conectarla con nuestro videojuego			Jorge, Aaron	0
	Pruebas y retoques finales			Jorge, Aaron	0
Desolieque					
	Preparación de la presentación final			Jorge, Aaron	0
	Entrega del proyecto			Jorge, Aaron	0
Cierre del Proyecto					
	Revisión final			Jorge, Aaron	0
	Documentación de la memòria			Jorge	50
	Reflexión sobre aprendizajes			Jorge, Aaron	0
	Presentar el proyecto (dios nos bendiga)			Jorge, Aaron	0

- Inspiración en videojuegos indie y enfoque minimalista

La estrategia elegida para este proyecto se basa en inspirarnos en otros videojuegos indie para crear el nuestro propio, con mecánicas, historia y estilo únicos. Apostamos por la simplicidad tanto en las mecánicas como en los gráficos, evitando la sobrecarga que ha llevado a muchos proyectos a quedar inconclusos por culpa de la avaricia y la falta de enfoque.

- Trabajo en equipo y colaboración equilibrada

Cada uno de nosotros aporta lo mejor de sí mismo al proyecto. Somos conscientes de que no podemos hacerlo todo por nuestra cuenta, por lo que nos complementamos mutuamente, equilibrando nuestras fortalezas y cubriendo nuestras carencias. De esta manera, conseguimos que el proyecto sea más sólido.

- Sinergia y compromiso del equipo

Las diferentes habilidades y perspectivas de cada miembro del equipo nos permiten trabajar de forma más eficiente. La colaboración es, sin duda, uno de los pilares fundamentales del proyecto, y es lo que nos impulsa a seguir adelante a pesar de cualquier dificultad.

1.5 Metodología del trabajo

Se ha decidido optar por la metodología ágil en el proyecto debido a la flexibilidad y capacidad de adaptación a los cambios. Este método es muy útil, ya que permite recibir constante retroalimentación y mejorar el juego. Además, fomenta la colaboración entre los integrantes, lo que enriquecerá el proceso.

2. Descripción del proyecto

2.1 Diseño del videojuego

2.1.1 Diseño artístico

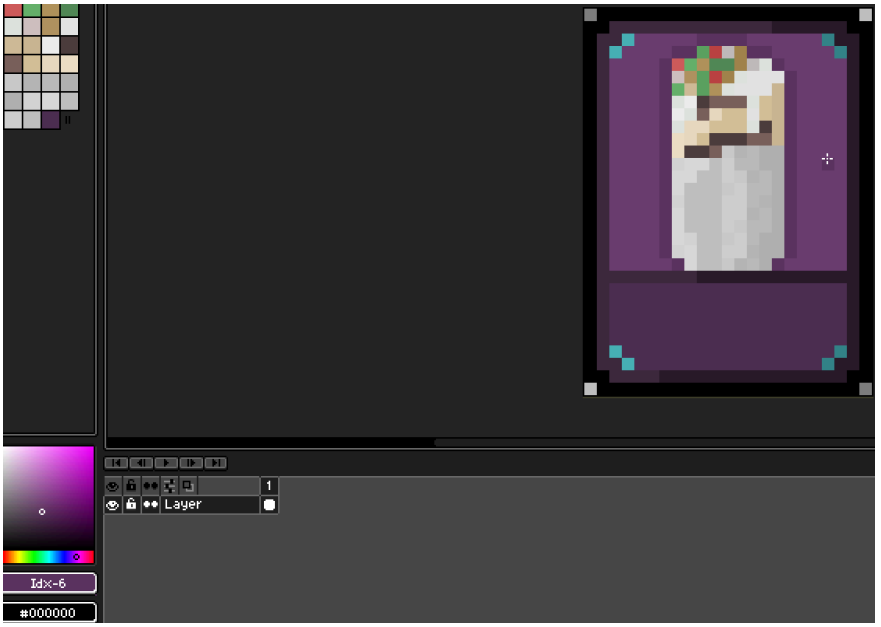
Aprovechando que es un juego en 2D, decidimos utilizar el estilo pixel art, al igual que otros títulos como Celeste o Balatro. Además, para añadirle un toque oscuro, diseñamos todo el juego con una paleta de colores basada en tonos lilas, lo que finalmente le ha dado una estética única y original.

2.1.1.1 Animaciones y efectos visuales

- Creación de animaciones con LibreSprite

Para la creación de las animaciones y efectos visuales del videojuego, se utilizó la herramienta LibreSprite al igual que los demás sprites del juego, la cual permitió dibujar de manera detallada cada frame de los personajes y elementos del juego. El proceso consistía en diseñar cuadro a cuadro las distintas poses y movimientos, logrando así animaciones fluidas y personalizadas.

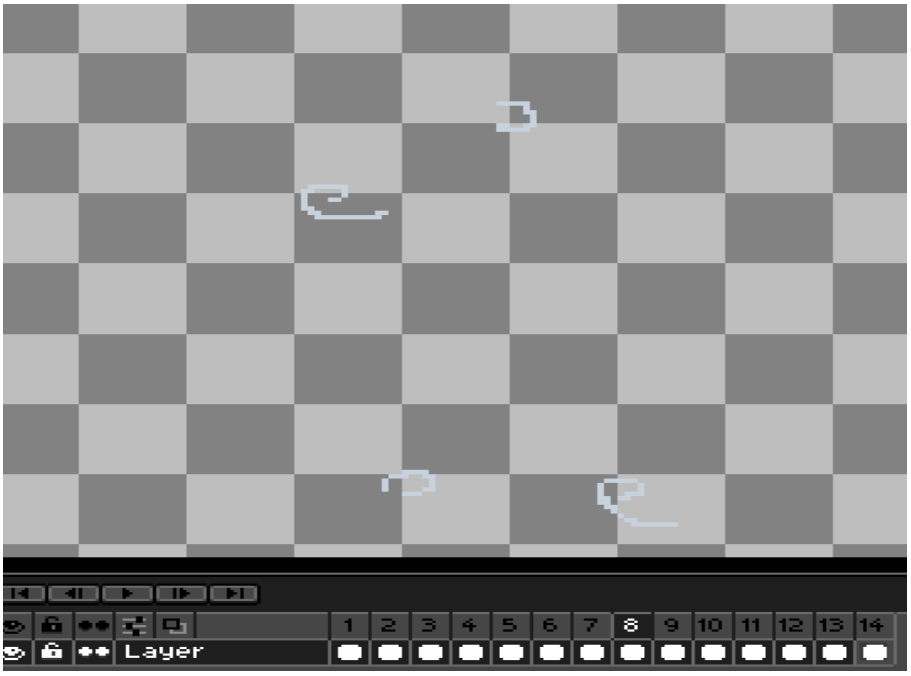
Imagen del software LibreSprite



- Integración de animaciones en Godot

Una vez finalizados los frames correspondientes a cada animación (por ejemplo, atacar, eventos, mensajes), estos se exportaban y almacenaban en una carpeta específica dentro del proyecto en Godot. Posteriormente, se utilizaban estos archivos de imagen para construir animaciones dentro del motor Godot, usando el nodo AnimatedSprite, según las necesidades de cada caso.

Imagen de una animación en LibreSprite



2.1.1.2 Sonido y música

- Uso de música libre de derechos y generación por IA

Para la ambientación sonora del juego, optamos por utilizar música con licencia libre de derechos, asegurando así el cumplimiento de los requisitos legales y éticos del proyecto. La música fue generada con herramientas de inteligencia artificial, lo que nos permitió adaptar el resultado exactamente al ambiente que buscábamos.

- Creación de una atmósfera inmersiva

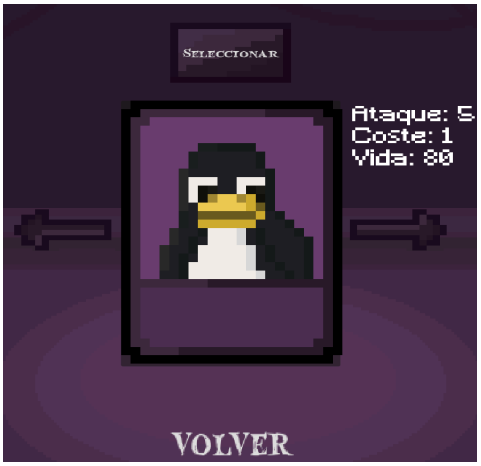
En concreto, solicitamos a la IA una música tétrica pero tranquila, que pudiera pasar desapercibida durante la partida sin distraer al jugador, pero que, a la vez, ayudará a crear una atmósfera inmersiva y envolvente. Este enfoque permitió integrar una banda sonora que complementa la experiencia de juego, aportando profundidad ambiental sin sobresalir ni saturar la escena.

2.2 Mecánicas

El juego implementa diversas **mecánicas estratégicas** en el uso de cartas. El jugador selecciona una carta activa desde el menú, la cual actúa como su personaje principal durante la partida. Para mejorar su rendimiento, se pueden **utilizar cartas** de apoyo, cada una con un costo de energía tomado del contador de energía. Esto requiere gestionar el **suministro de energía** a lo largo de la partida para optimizar el uso de las cartas y derrotar al BOT enemigo.

2.2.1 Cartas activas

Son los personajes que usas para atacar directamente al boss. Cada uno tiene diferente daño, coste de energía y vida.



2.2.2 Cartas objeto

Son las cartas que tienes en la mano para apoyar a la carta activa en la pelea contra el boss.

2.2.3 Críticos y fallos

Al atacar, tanto tú como el boss podéis hacer un golpe crítico que duplica el daño, o fallar el ataque y que no haga nada.

Imagen de un golpe crítico

Mensaje de un golpe fallado

2.2.4 Eventos

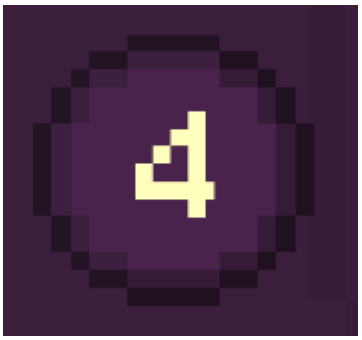
Al cambiar de turno, puede activarse un evento que cambia las condiciones del campo, pudiendo ser beneficioso o perjudicial.

Imagen de un evento activo

2.2.4 Energía

La energía se usa para atacar con las cartas activas y para jugar las cartas objeto. Se recarga pasivamente al empezar tu turno y también con algunas cartas específicas.

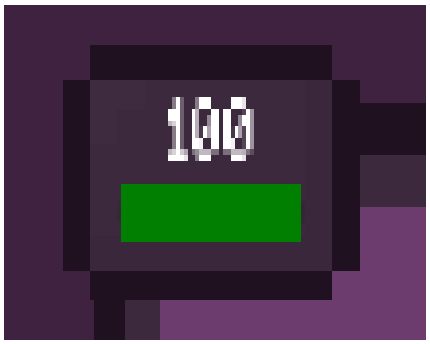
Contador de energía



2.2.5 Progresión del jugador

La progresión de la partida dependerá de las decisiones estratégicas del jugador. A medida que avance, el **contador de energía** se recargará parcialmente al final de cada turno. El jugador deberá administrar sus recursos considerando el daño recibido, pudiendo **utilizar cartas** de curación o bien optar por cartas ofensivas para aumentar su daño.

Barra de vida del jugador



Mazo de cartas

2.3 Análisi de requisits

2.3.1 Requisitos funcionales

Sistema de juego por turnos:

El videojuego debe implementar una mecánica de juego basada en turnos, donde el jugador y la entidad rival alternen acciones de manera ordenada. Cada turno debe permitir al jugador analizar la situación, tomar decisiones y ejecutar movimientos estratégicos antes de ceder el turno al oponente.

Interacción mediante arrastre de cartas:

Durante el desarrollo de la partida, el jugador debe tener la posibilidad de interactuar con las cartas a través de un sistema de arrastrar y soltar. Esta funcionalidad busca ofrecer una experiencia más intuitiva y dinámica, facilitando la colocación de cartas en el tablero o la activación de sus efectos.

Uso de cartas

https://github.com/AaronPeva/Tenebris

Deck | Aaron Pérez Vargas, Jorge Rocano Da Silva

Cartas de personaje con atributos definidos:

Las cartas correspondientes a personajes deben contar con atributos claramente definidos como ataque, vida y habilidades especiales. Estos atributos influirán directamente en las mecánicas de combate y en el resultado de los enfrentamientos, siendo fundamentales para la planificación estratégica del jugador.

Cartas de utilidad con efectos de soporte:

El juego debe incluir cartas de utilidad que proporcionen efectos complementarios o de apoyo, tales como curación, mejoras temporales de atributos o manipulación del mazo. Estas cartas permitirán al jugador adaptar su estilo de juego y reaccionar ante situaciones adversas.

Variedad y diversidad en el mazo de cartas:

El sistema debe ofrecer una amplia variedad de cartas tanto de personaje como de utilidad, cada una con características únicas. Esta diversidad garantizará la rejugabilidad y fomentará la exploración de distintas estrategias por parte del jugador, además de enriquecer el diseño general del juego.

https://github.com/AaronPeva/Tenebris

2.3.2 Requisitos no funcionales

Mecánica intuitiva:
El juego debe ser fácil de entender y jugar, con una interfaz clara que facilite la interacción sin necesidad de explicaciones extensas.

Rendimiento estable:
Debe garantizar un funcionamiento fluido, sin caídas de rendimiento ni retrasos que afecten la experiencia del jugador.

Seguridad e integridad:
El sistema debe proteger los datos del juego, asegurando que la información y el progreso del jugador estén seguros y no puedan ser alterados de forma indebida.

Escalabilidad:
La arquitectura del juego debe permitir la incorporación futura de nuevas cartas, niveles o funcionalidades sin afectar el rendimiento ni la estabilidad.

Usabilidad:
La interfaz y controles deben ser accesibles, permitiendo que jugadores con distintos niveles de experiencia puedan disfrutar del juego sin dificultades.

2.4 Tecnologías

Para el desarrollo del proyecto se han utilizado diversas tecnologías que permiten implementar sus mecánicas y funcionalidades de manera eficiente.

- Motor de desarrollo

Godot Engine: Se ha elegido este motor debido a su ligereza, código abierto y compatibilidad con GDScript, un lenguaje optimizado para el desarrollo de videojuegos.

- Lenguaje de programación

GDScript: Un lenguaje inspirado en Python, diseñado específicamente para Godot, que permite una programación rápida y flexible.

- Herramientas adicionales

Git: Se utiliza para el control de versiones y la colaboración en el desarrollo del proyecto.

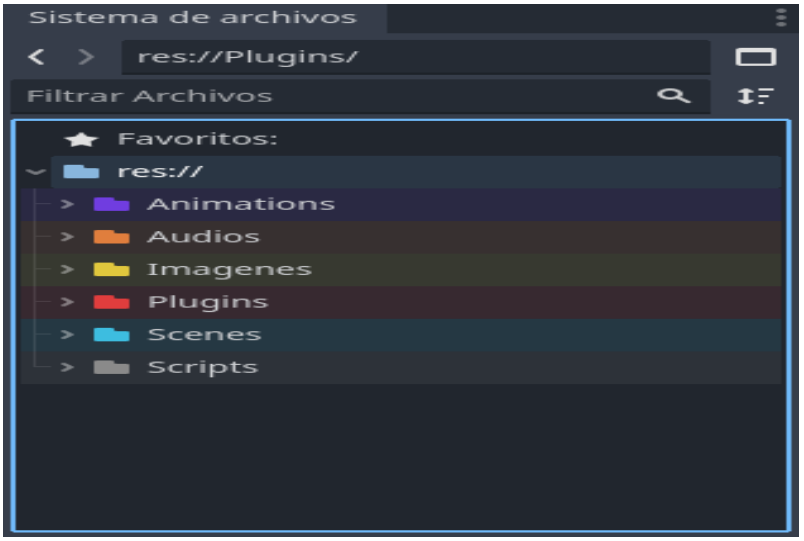
- Software de edición gráfica

LibreSprite: Para la creación de sprites de todo el videojuego.

2.5 Estructura y arquitectura del proyecto

Durante el desarrollo del videojuego en Godot Engine, organizamos el proyecto siguiendo una estructura de carpetas clara y modular, con el objetivo de facilitar la navegación, el mantenimiento y la escalabilidad del código y los recursos. Esta estructura responde a las convenciones recomendadas por la comunidad de Godot y se adaptó a las necesidades específicas de nuestro juego:

- **/Animations:** Recursos de animaciones para personajes, cartas y eventos.
- **/Audios:** Efectos de sonido del juego.
- **/Imágenes:** Sprites y elementos gráficos personalizados.
- **/Plugins:** Plugins externos o personalizados, como el plugin de GitHub para Godot.
- **/Scenes:** Escenas principales del juego, incluyendo niveles y menús.
- **/Scripts:** Código en GDScript que controla la lógica del juego y la interfaz.



https://github.com/AaronPeva/Tenebris

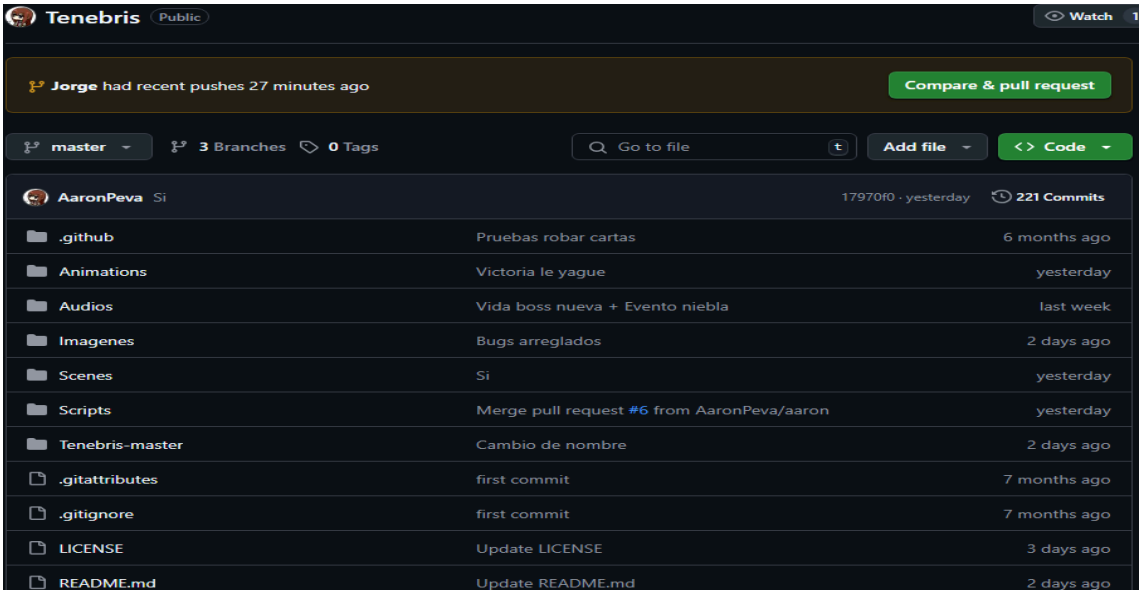
2.6 Desarrollo y control de versiones

Para el control de versiones se ha utilizado GitHub, permitiendo llevar un registro detallado y organizado de los avances y cambios realizados en el proyecto a lo largo del tiempo.

Esta herramienta facilitó el trabajo colaborativo, permitiendo que ambos integrantes del equipo pudieran trabajar en paralelo sin conflictos, gracias al uso de ramas para el desarrollo de nuevas funcionalidades o la corrección de errores.

Se realizaron commits frecuentes y descriptivos, lo cual no solo facilitó la trazabilidad de cada modificación, sino que también permitió documentar el proceso de desarrollo de manera clara y ordenada.

El versionado resultó fundamental para identificar y solucionar errores, así como para comparar distintas versiones del código y recuperar estados anteriores del proyecto en caso de fallos o regresiones, asegurando así un desarrollo más seguro y controlado.



https://github.com/AaronPeva/Tenebris

2.7 Pruebas y validación

El juego fue sometido a pruebas de jugabilidad, con sesiones de testeo con alumnos de nuestra clase o los propios miembros del proyecto para detectar errores y posibles mejoras. El feedback recibido sirvió para ajustar la dificultad, mejorar la usabilidad y pulir detalles tanto visuales como sonoros.

2.8 Accesibilidad y aspectos éticos

Se procuró que el juego fuera accesible, utilizando colores contrastados y controles simples para facilitar la experiencia a todo tipo de jugadores. El contenido es adecuado para todos los públicos y no incluye mensajes ofensivos o discriminatorios.

2.8.1 Configuración del videojuego

El juego incluye un apartado de configuración que permite activar el modo de pantalla completa y ajustar el volumen. Estas opciones están pensadas para mejorar la comodidad del usuario y ofrecer una experiencia más personalizada durante el juego.

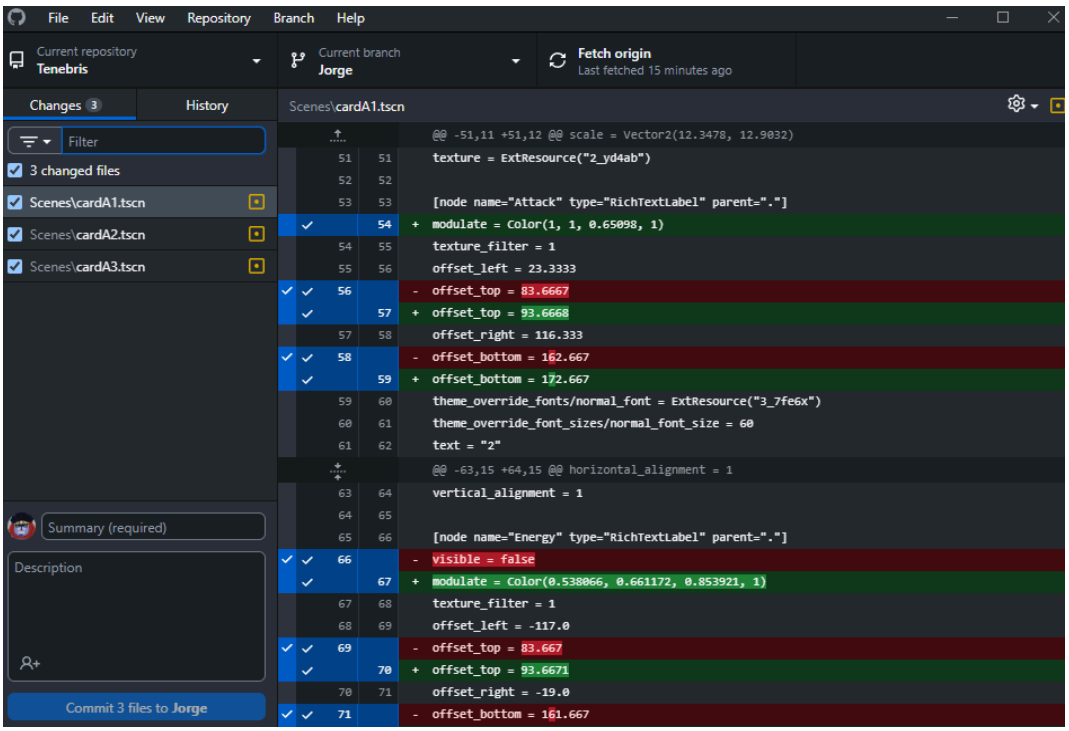
2.9 Compatibilidad

Nuestro proyecto es compatible con múltiples sistemas operativos, incluyendo Linux, Windows y Mac. Esto permite una mayor accesibilidad y flexibilidad para los usuarios, ya que puede ejecutarse correctamente en diferentes entornos sin necesidad de realizar cambios significativos en su configuración.

2.10 Integración de recursos externos

- Uso del plugin de GitHub en Godot

Durante el desarrollo del proyecto, se integraron diversos recursos externos que facilitaron tanto la gestión del código como la inspiración para el diseño de las mecánicas de juego. En primer lugar, se utilizó el plugin oficial de Godot para GitHub, lo que permitió una integración directa del motor con la plataforma de control de versiones. Gracias a este plugin, fue posible realizar commits, gestionar ramas y sincronizar cambios sin salir del entorno de Godot.



https://github.com/AaronPeva/Tenebris

- Investigación y referencia a videojuegos de cartas

Además, para la implementación de algunas mecánicas de juego relacionadas con los videojuegos de cartas, se recurrió a la visualización de vídeos en YouTube. Estos recursos audiovisuales resultaron de gran utilidad para analizar diferentes enfoques y dinámicas presentes en juegos de cartas digitales. A partir de estas referencias, se seleccionaron e implementaron aquellas ideas que mejor se adaptan a la experiencia jugable deseada en nuestro proyecto.

3. Plan a futuro

3.1 Mejoras y nuevas funcionalidades

- **Implementación del modo historia**
  Se incorporará un modo historia que permita a los jugadores sumergirse en el universo de *Tenebris* a través de misiones y narrativa progresiva, aportando contexto a los personajes y profundizando la experiencia de juego.

- **Ampliación del conjunto de cartas y efectos**
  Se añadirán nuevos tipos de cartas con efectos especiales y condiciones únicas, tanto para las cartas activas como las de objeto, ampliando así la variedad de estrategias disponibles.

- **Soporte multijugador en línea**
  Se desarrollará un sistema para partidas multijugador, permitiendo enfrentamientos entre jugadores en tiempo real y fomentando la competencia y la comunidad.

- **Mejoras en la inteligencia artificial del oponente**
  Se trabajará en una IA más sofisticada que analice jugadas anteriores, reaccione de forma dinámica y represente un desafío real, adaptándose a diferentes niveles de dificultad.

- **Sistema de logros y estadísticas**
  Se implementará un sistema que registre el progreso del jugador, logros desbloqueados, estadísticas de juego y otros hitos, incentivando la rejugabilidad y la mejora personal.

- **Editor visual de cartas**
  Se planea crear una herramienta integrada que permita a los usuarios diseñar sus propias cartas de manera visual, facilitando la personalización y fomentando la creatividad dentro de la comunidad.

3.2 Accesibilidad y usabilidad

- **Localización a otros idiomas**
  Para ampliar el alcance del juego, se traducirá el contenido a distintos idiomas, priorizando inglés y catalán, pero dejando abierta la posibilidad de añadir más en el futuro según la demanda.

- **Mejoras en la interfaz de usuario**
  Se revisará y optimizará la interfaz para que sea más intuitiva, accesible y adaptada tanto a jugadores nuevos como experimentados, asegurando compatibilidad con diferentes resoluciones y dispositivos.

4. Manual

Manual de instalación de Tenebris

1. Acceder al repositorio
   Ve al siguiente enlace de GitHub:
   https://github.com/AaronPeva/Tenebris

2. Descargar el juego
   Una vez en el repositorio, haz clic en el botón verde que pone "Code" y selecciona "Download ZIP".

3. Descomprimir el archivo
   Cuando se haya descargado el archivo ZIP, descomprímelo en la ubicación que prefieras.

4. Ejecutar el juego
   Entra en la carpeta descomprimida y abre el archivo ejecutable que encontrarás dentro.

5. Glosario

Gameplay: Experiencia de juego, incluyendo controles, interacciones, mecánicas y fluidez de la partida.

AnimatedSprite: Nodo del motor Godot utilizado para mostrar animaciones 2D basadas en múltiples fotogramas de una imagen.

Drag and Drop (Arrastrar y soltar): Sistema de interacción que permite al jugador mover cartas con el cursor y soltarlas en el tablero para activar su efecto.

GDScript: Lenguaje de programación de alto nivel y tipado dinámico, similar a Python, creado específicamente para su uso con el motor Godot.

Git: Sistema de control de versiones distribuido que permite gestionar y registrar los cambios del código fuente durante el desarrollo.

GitHub: Plataforma online donde se almacena el repositorio del proyecto, permitiendo colaboración, control de versiones y manejo de ramas de desarrollo.

Godot Engine: Motor de desarrollo de videojuegos open source utilizado para crear el proyecto, ideal para juegos 2D y 3D con herramientas integradas.

Indie: Término que describe videojuegos desarrollados de forma independiente, sin el respaldo financiero de grandes empresas.

LibreSprite: Programa gratuito utilizado para crear gráficos en pixel art y animaciones cuadro a cuadro para el videojuego.

Pixel art: Estilo gráfico retro basado en el uso de píxeles grandes y visibles, típico de videojuegos clásicos y modernos independientes.

Deck | Aaron Pérez Vargas, Jorge Rocano Da Silva

Proyecto: Conjunto organizado de tareas y recursos planificados con el fin de lograr un producto final, en este caso un videojuego de cartas.

Ramas (branches): Líneas independientes de desarrollo dentro de un repositorio Git. Permiten trabajar en nuevas funciones o corregir errores sin afectar al código principal.

6. Bibliografía

En esta bibliografía se recogen las fuentes consultadas para obtener información sobre las plataformas, herramientas y tecnologías utilizadas o mencionadas durante el desarrollo del proyecto, como navegadores, motores de desarrollo, foros y plataformas de colaboración.

Godot Engine. (2014). *Manual de usuario de Godot.*
https://docs.godotengine.org/es/4.x/

OpenAI. (2015). *Guía de Chat GPT*.
https://openai.com

YouTube. (2005). *Videos sobre desarrollo en Godot.*
https://www.youtube.com

Google. (2008). *Navegador.*
https://www.google.com

GitHub. (2008). *GitHub*
https://github.com/dashboard

Reddit. (2005.). *Foro en línea*
https://www.reddit.com