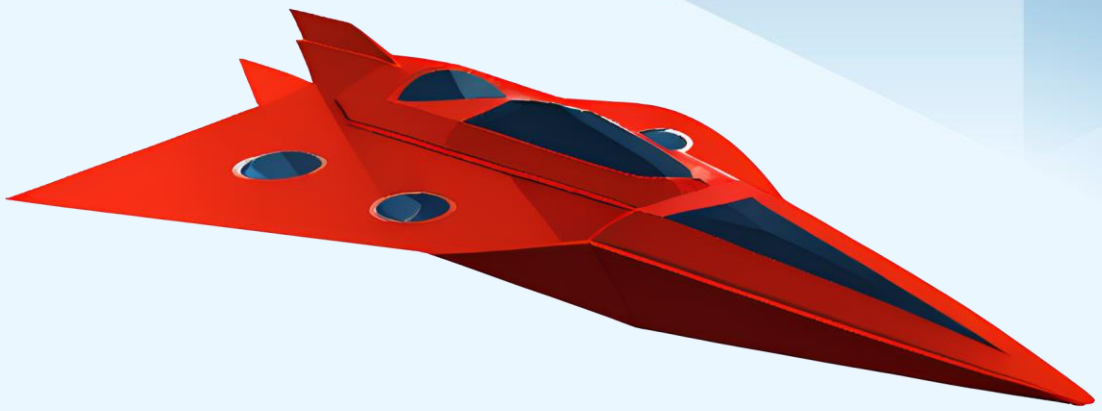
Proyecto de Fin de Grado (DAM2)



GALACTIC REBELION

MIEMBROS DEL PROYECTO

Alejandro Palá

Carlos Gamboa

TUTOR DE PROYECTO

Jose Federico Rodriguez de la Maya

INSTITUTO

IES LOPE DE VEGA

CURSO ACADÉMICO

(2023-2025)

ÍNDICE

1. Introducción

1.1 Introducción General

A lo largo de este tiempo los videojuegos tipo arcade han sido una referencia vital en este mundo para el entretenimiento destacando su simplicidad, dinamismo y una gran capacidad de captar la concentración a los jugadores en sesiones cortas de juego pero intensas.Sin embargo la mayoría de estos complementos se han rezagado por la aparición de nuevas tecnologías mucho más modernas y por la evolución de juegos más complejos, lo que ha provocado que esa esencia clásica se vea muy lastrada por el desarrollo de propuestas más modernas adaptándolas especialmente a todo tipo de dispositivos móviles.

En este documento comenzaremos con la explicación previa a nuestra memoria del proyecto “Videojuego para móvil”(GALACTIC REBELION).Este proyecto forma parte de nuestro TFG y tiene como finalidad aplicar todo el aprendizaje adquirido durante este ciclo formativo (DAM).

Este juego implementa una jugabilidad rápida, fluida y dinámica perfectamente acorde a su nombre(GALACTIC REBELION) inspirado en juegos arcade 2D con una excelente adaptabilidad para la usabilidad de móviles.Este proyecto busca ofrecer a los jugadores que están constantemente activos y concentrados a las fases que surjan en pantalla.Cada elemento ha sido diseñado para equilibrar el ritmo y dinamismo de este juego logrando mantener una curva de dificultad bien equilibrada para los jugadores.

Empezaremos con una explicación muy breve sobre el proyecto.

Como se ha dicho previamente el proyecto se trata de un juego para móvil tipo arcade que consta de unas 9 pantallas que se explicarán en profundidad más adelante a lo largo de este documento. En esta memoria se explicará detalladamente todo el proceso completo del desarrollo: el diseño , el código, pruebas realizadas , errores cometidos , sus etapas de seguimiento así como sus conclusiones de forma detallada y concisa.

1.2 Objetivos

El principal objetivo de este proyecto de fin de grado es implementar mediante herramientas de programación , un juego arcade 2D para móvil con C# usando el IDE de Unity.Se busca desarrollar un videojuego funcional sin errores que demuestre nuestros conocimientos y aprendizajes adquiridos a lo largo del curso en cuanto a diseño y desarrollo de software.

En este proyecto se desarrollarán los siguientes objetivos:

- Poner en práctica todos nuestros conocimientos aplicados durante el ciclo formativo orientado en el tema de videojuegos y desarrollo de software.
- Conseguir una jugabilidad fluida , dinámica y atractiva para el usuario
- Una navegación de pantallas sencilla y eficaz para el usuario.
- Conectarse a una API en la que se integrará la información de los usuarios y sus partidas en una base de datos.
- Efectos de sonido tipo colisión/destrucción para una mejor experiencia del usuario así como música de fondo etc.
- Conseguir implementar niveles de dificultad para lograr un mayor dinamismo en cuanto a jugabilidad y variedad.
- Implementar un ranking de puntuación. Así se logra incentivar a otros jugadores a superar puntuaciones de récord ampliando mucho más a un ámbito competitivo
- Diseño responsive para todos los dispositivos móviles. Ya que el juego está hecho para móvil es muy importante que el diseño de la interfaz del juego tenga una buena estructura visual para todas las resoluciones de pantalla móvil. Con esto se logra una buena experiencia visual y consistente para todo tipo de dispositivos.

Con el conjunto de estos objetivos no solo se demuestra nuestra capacidad técnica y creativa como desarrolladores ,sino que busca una combinación de entretenimiento y calidad para el impacto de los usuarios. El enfoque principal de este proyecto está orientado a la mejora y optimización de la experiencia del usuario tanto a nivel visual como funcional cumpliendo los requisitos presentes en el tema de desarrollo de videojuegos 2D.

1.3 Descripción

Galactic Rebelion es un juego adaptado a dispositivo móvil con un encaminamiento inspirado en los juegos arcade 2D para transmitir una experiencia de juego intensa,fluida,impactante y adictiva.Este proyecto se ha llevado a cabo como una idea que combina aprendizaje adquirido como el diseño de interfaces, lógica de programación , gestión de datos y buena experiencia para el usuario.

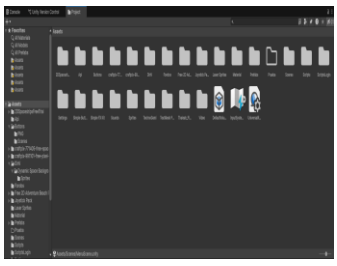
Galactic Rebelion es un juego 2D en el que el jugador/usuario cuenta con una nave espacial que se puede elegir en un selector de naves que controla con un joystick virtual. El juego consta de tres fases las cuales duran un periodo determinado con el fin de obtener la máxima puntuación derrotando a enemigos y jefes finales que disparan láseres sobreviviendo cada oleada para pasar a la siguiente fase.Los enemigos y jefes van variando a medida que se vaya pasando de fase conforme Implementa sistemas de explosión cuando las naves enemigas y jefes finales.

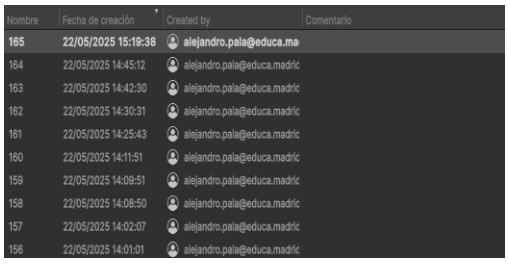
Por último cabe destacar de que incorpora dos cinemáticas incluidas(Introducción y Final feliz) en el juego con una música ambiental acorde al escenario para la mejora de experiencia del usuario y que así se meta en la historia del juego así como pantallas de de derrota y victoria.

2. Tecnologías empleadas

2.1 Unity Hub

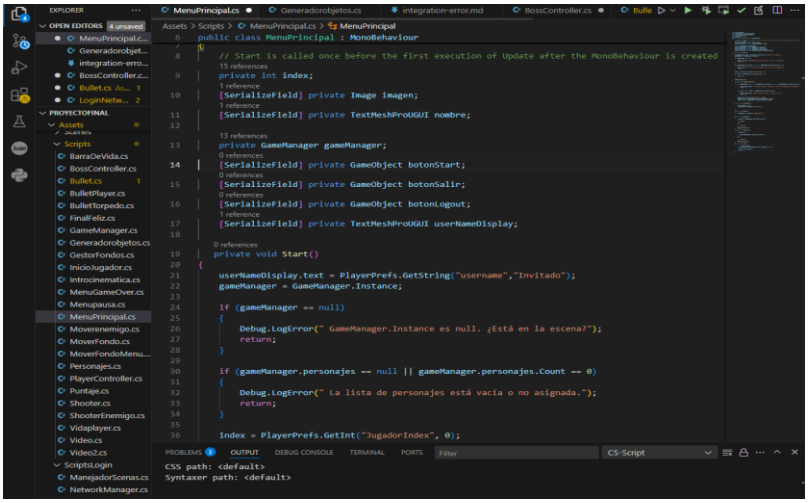
Para el desarrollo de nuestro proyecto hemos utilizado el IDE de Unity Hub, la cual nos permite actualizar los cambios aplicados de cada uno con mucha facilidad.También proporciona la capacidad de los changesets el cual ayuda mucho en el caso de que si se ha extraviado un cambio puedes volver a la versión o cambio funcional.





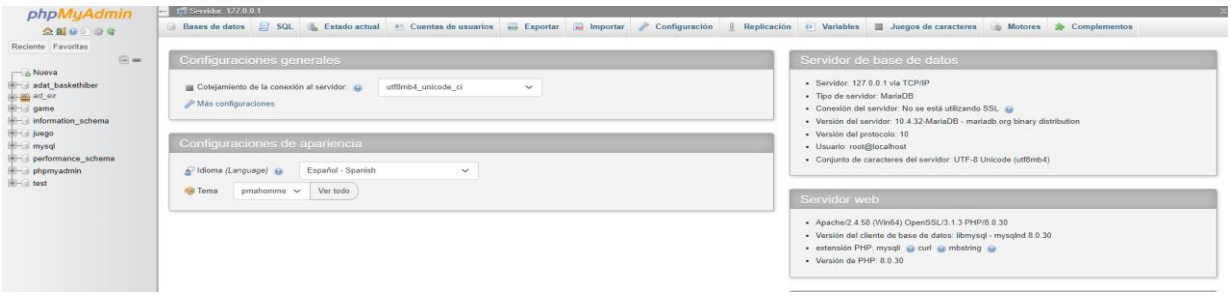
2.2 VSCode(C#)

Para la programación del juego se ha usado Visual Studio Code(VS Code) junto con el lenguaje de programación C# que es el lenguaje más indicado soportado por el motor de Unity. VS Code es un editor de código altamente personalizable y con la capacidad de instalar una gran cantidad de extensiones que facilitan el desarrollo de videojuegos



2.3 MySQL(phpMyAdmin)

Para la gestión y almacenamiento de los datos de los usuarios y sus respectivas partidas se ha usado MySQL la herramienta de gestión de bases de datos relacionales y phpMyAdmin como interfaz de administración.Esto es debido a la facilidad de uso ,fiabilidad y compatibilidad con múltiples entornos.



3. Desarrollo del proyecto(estructura y diseño)

El desarrollo del juego se dió mediante el IDE de Unity que es utilizado mayormente en la creación de videojuegos. Al tener nuestra primera y además

única experiencia en este IDE se decidió implementar el proyecto dándole uso a las herramientas proporcionadas especializadas en esta área.

3.1 Estructura

1. Idea General

Los primeros pasos para el desarrollo de nuestro videojuego fue la búsqueda de sprites.Unity facilita mucho esto puesto que tiene una herramienta llamada Asset store que nos lleva directamente a la pagina oficial de Unity en la que hay una gran cantidad de Assets gratos que podemos implementar en nuestro proyecto a nuestro antojo.
Buscamos una gran cantidad de sprites los cuales nos facilitaran la construcción principal del juego, buscando así sprites de naves espaciales, enemigos, fuentes de texto, botones, diseños y así como muchas más para implementar el esqueleto del juego.

2. La creación de las distintas escenas

La cuales serían cada una de las distintas pantallas que necesitaríamos para que el usuario pudiera trasladarse por el juego. Se creó un mapa mental de funciones esenciales que el juego requeriría. Una pantalla de login, de registro, un menú y las distintas pantallas que necesitaría el juego en funcionamiento en general.

3. El funcionamiento de el login y la base de datos

Al querer hacer un juego en base a puntuaciones necesitábamos la manera de poder almacenar esos datos y además de cada jugador que estuviera acumulando dichos puntos. De esa forma nos pusimos con la creación de login y el registro de aquellos jugadores que quisieran registrarse al juego.

4. El gameplay

Cuando hablamos del gameplay se refiere al contenido del juego, la experiencia del usuario al jugarlo. La idea en general era crear un gameplay semejante al de los arcades ochenteros. Así que con el uso de los Sprites le dimos las principales acciones que tenían los mismos. Lo cual era disparar, moverse, atacar enemigos,ganar puntuación, tener efectos y trasladarse de un nivel a otro cuando se ganaba.

5. Leve mejora de el diseño de las escenas

Básicamente hemos hecho ligeros cambios en cuanto a diseño de las pantallas.Por ejemplo se cambiaron algunas estructuras de las escena como el login y registro para una mejor coherencia visual , el cambio a unos sprites de botones que encajen con el diseño general de la escena , que la puntuación se vea un poco más nítida y quitar el contorno del botón al saltar las cinemáticas.

6. Implementación de videos, imágenes, diseños y bosses que aviven el juego.

Para que el juego tuviera un poco más de dificultad y tuviera una historia optamos implementar bosses finales y videos estilo cinemática.
Básicamente el boss tiene un script en el que tiene un gameobject para añadirle el prefab ya que se va a instanciar varias veces y un array de Transform que son los waypoints por los que va pasar, luego también se le ha asignado sus puntos de vida y el daño.Luego se han añadido cinemáticas estilo video las cuales son generadas por una IA de videos, básicamente
Se ha usado un componente Video player para meter el archivo del video y un render texture para el componente DRawImage donde se va a reproducir el video en el que a la vez está dentro de un Canvas que es la UI.

7. La creación de un ranking de puntuaciones

El ranking de puntuaciones fue pensado desde el inicio del juego como una herramienta para que los jugadores pudieran retarse entre ellos y así motivar a los demás generando un sentimiento de competencia.
Las puntuaciones fueron guardadas desde el inicio en la base de datos y fue creada en la API dos scripts los cuales son “getTopScores” y “saveScores” los cuales funcionaban para recolectar los mejores scores y guardarlos en la base de datos.
La implementación de el scene de el ranking fue bastante sencillo, donde tenemos un título y 10 textos que se llenarán con el script creado en Unity. La función del script es hacer un select de los 10 mejores scores de cada usuario y eso se guardará en un array que se mostrará en pantalla.

8. Colisiones entre entre naves enemigas , bosses y su artillería con el jugador.

Para darle un toque mucho más realista y natural se han incluido colisiones entre GameObjects el cual se hace mediante un método que se llama (OnCollisionEnter(Collision collision)) pero como al quitarle el trigger a los enemigos y al jugador hacían movimientos extraños se ha cambiado por el

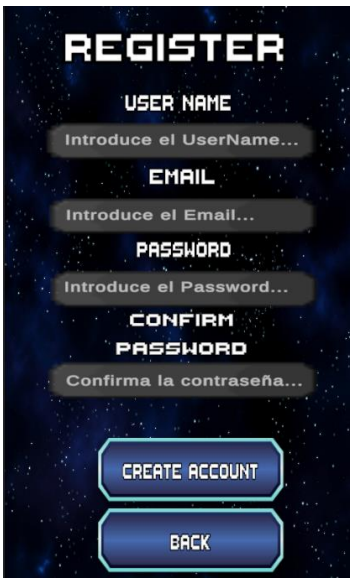
(OnTriggerEnter(Collider other)) y dentro de esa función implementar la lógica ya que nos permitía controlarlo con más naturalidad y facilidad.Por supuesto también hemos tenido que ajustar colliders en los sprites para el área de contacto y asemejarse lo más real posible.

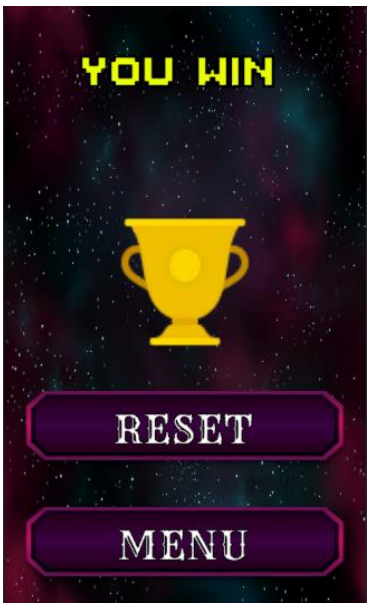
9. Navegabilidad entre pantallas

Uno de los rasgos más importantes de nuestro videojuego ha sido garantizar una navegación fluida y coherente entre las diferentes pantallas del videojuego. El usuario siempre empieza su primera toma de contacto con la aplicación , por lo que es fundamental organizar una interfaz visual clara y sencilla para el usuario con accesos rápidos a las funcionalidades principales del juego.

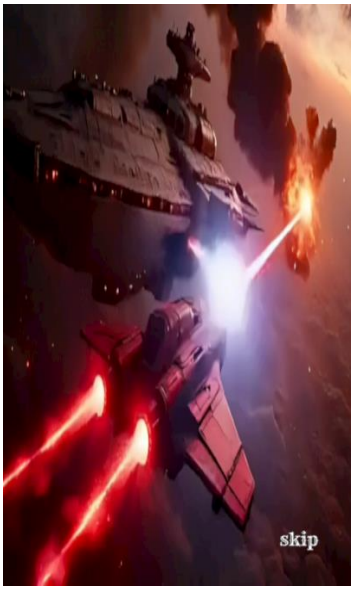
3.2 Diseño

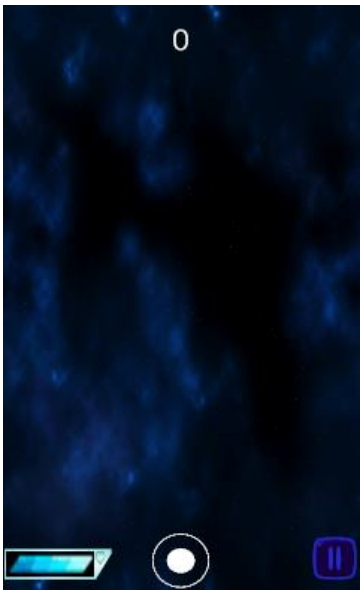
Aquí se mostrarán las capturas de pantalla más relevantes:






skip


skip


0


0

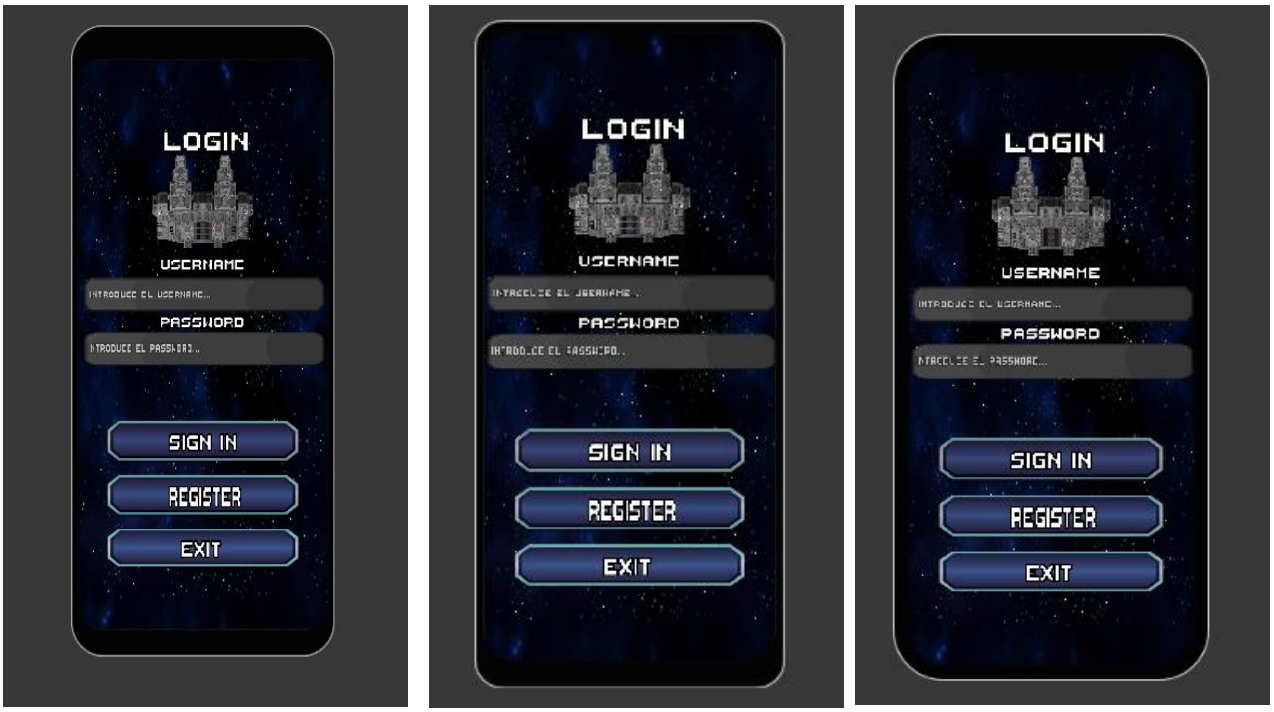

0


TOP 10 MEJORES JUGADORES
New Text
New Text
New Text
New Text
New Text
New Text
New Text
New Text
New Text
New Text

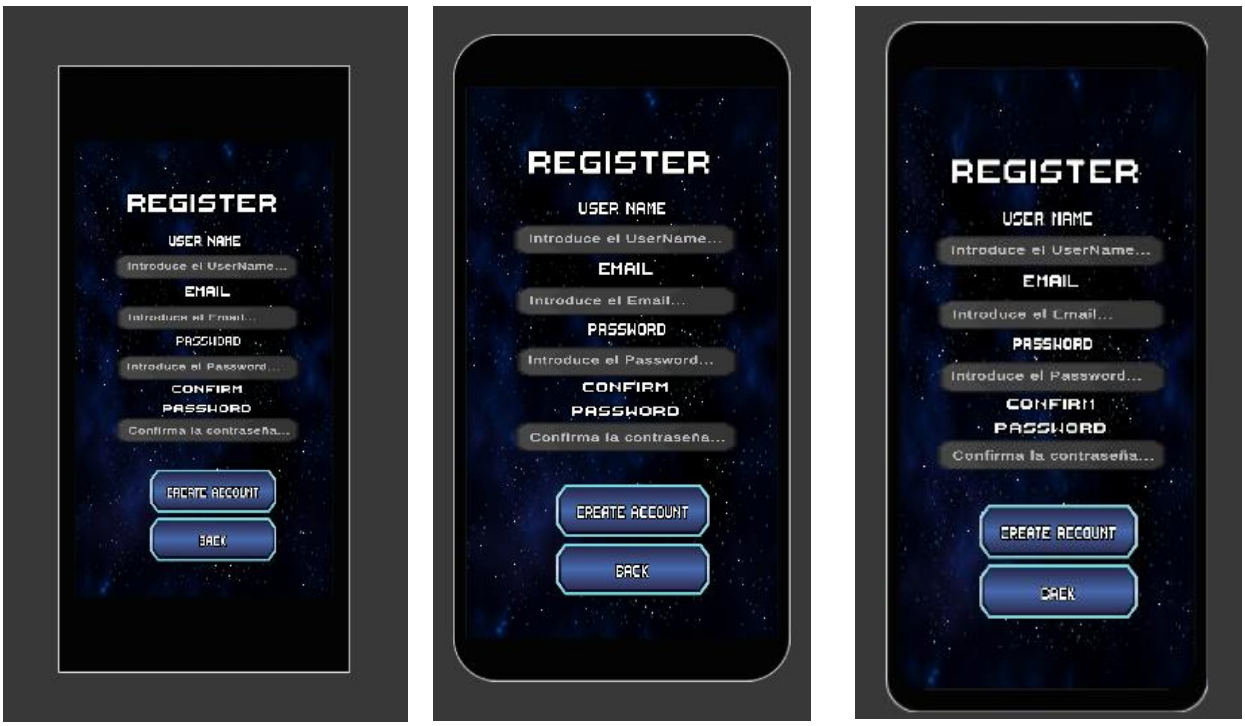
Responsive

A continuación se mostraran algunos ejemplos de algunas pantallas adaptadas a distintas resoluciones en móvil:

Samsung Galaxy S 9, Xiaomi Redmi Note 7, iPhone 12 Pro



Sony Xperia XZ Premium, Motorola Nexus 6, Google Pixel 4



4. Desarrollo del proyecto(funcionalidades y etapas)

4.1 Funcionalidades

Las funcionalidades implementadas abarcan desde sistemas básicos (movimiento, disparos) hasta elementos complejos (autenticación segura, ranking online,persistencia de datos), combinando técnicas de programación en *C#* (Unity) y *PHP* (backend). El resultado es un juego arcade funcional, con una jugabilidad retro, pero con tecnología moderna que garantiza escalabilidad y seguridad.

4.2 Etapas

Etapas Clave

1. **Prototipado Inicial**:
    - Creación del gameplay básico inspirado en arcades clásicos: movimiento del jugador, disparos, colisiones y sistema de puntuación.
    - Implementación de mecánicas como niveles progresivos y efectos visuales/auditivos.
2. **Sistemas de Usuario**:
    - **Login y Base de Datos**: Se desarrolló un sistema de autenticación con PHP y MySQL para almacenar perfiles y puntuaciones, usando prepared statements para seguridad.
    - **Ranking**: Integración de un top de puntuaciones consultable desde la interfaz, ordenado de forma descendente.
3. **Refinamiento**:
    - **Colisiones**: Uso de OnTriggerEnter para gestionar interacciones entre naves, proyectiles y enemigos, ajustando *colliders* para un comportamiento realista.
    - **Navegabilidad**: Transiciones fluidas entre pantallas (login, menú, partida) mediante eventos de UI y gestión de escenas en Unity.

Fases del proyecto

Fase	Periodo	Planificación
Organización de ideas y diseño general	Finales de marzo - primera semana de abril	Idea del concepto general del juego, herramientas a usar , elaborar bocetos de pantallas y diagramas E/R
Diseño y navegación entre las pantallas	Segunda - Tercera semana de abril	Creación de las escenas principales , menús y navegación entre las pantallas y funcionalidad de los botones mediante scripts
Implementación de la jugabilidad	Cuarta semana - principios de mayo	Desarrollar la lógica del juego mediante control de la nave , movilidad de los enemigos, joystick y otros elementos importantes
Implementación de API y conexión a una base de datos	Segunda semana de mayo	Integrar mediante los campos del login y registro a una base de datos , también registrar puntuaciones y lógica interna a la hora de registrarse.
Revisión , optimización de errores y mejora en algunas funcionalidades	Tercera semana de mayo	Corregir bugs , mejoras en el diseño de las pantallas , lógica general del juego y el rendimiento en sí.
Realización de la documentación	Cuarta semana de mayo-principios de junio	Redacción de la memoria del proyecto , elaborar todos los apartados correspondientes (capturas , bibliografía..) antes de la entrega
Elaboración de la defensa final	Primera semana de junio	Preparación para la exposición de la defensa final del proyecto.

5 Base de datos (diseño y estructura)

El sistema desarrollado gestiona partidas y puntuaciones de usuarios. Para ello, se ha diseñado una base de datos sencilla pero funcional, compuesta por tres tablas principales:

5.1 Estructura

1. Tabla usuarios

Esta tabla contiene los datos básicos de los usuarios registrados en el sistema. Su propósito es autenticar a los jugadores antes de permitirles acceder o registrar resultados.

Campos esperados:

- userName: nombre de usuario único.
- password: contraseña almacenada de forma segura mediante hash SHA-256.

Uso: En el archivo checkUser.php, se verifica la existencia del usuario y su contraseña para permitir el acceso.

2. Tabla partidas

Registra cada partida realizada por los usuarios, guardando los puntos obtenidos en cada una.

Campos:

- userName: usuario que realizó la partida.
- puntos: puntos obtenidos en la partida.

Uso: El archivo createMatch.php permite registrar una nueva partida si el usuario existe en la tabla usuarios.

Seguridad y Validaciones

- Se utilizan **prepared statements** para prevenir inyecciones SQL.

- Las contraseñas están protegidas mediante hashing SHA-256.

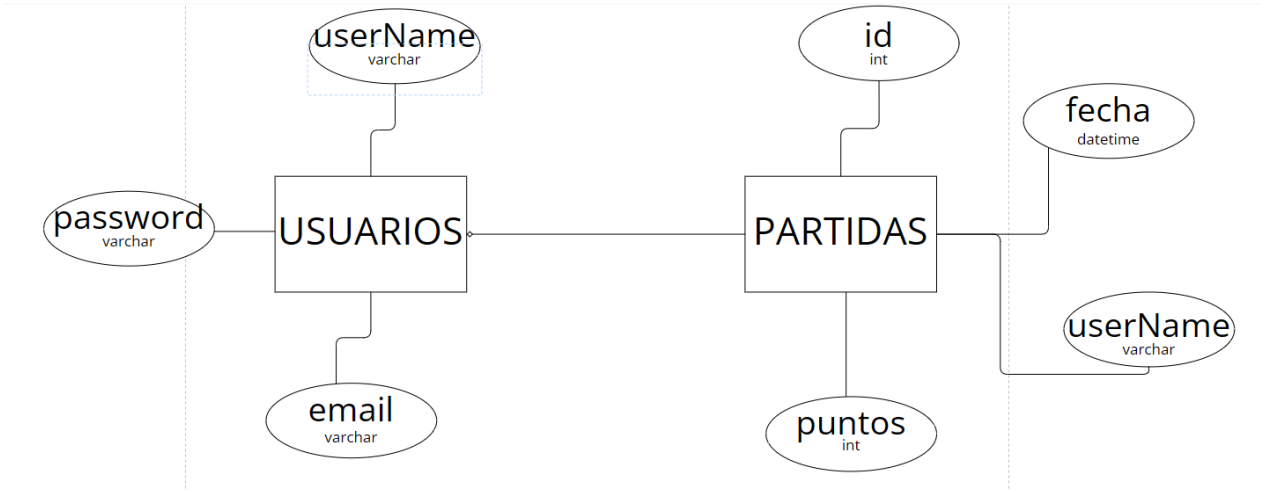
- Se realizan múltiples validaciones para asegurar que los datos estén completos y que el usuario exista antes de realizar operaciones.

Funcionalidades

1. **Gestión de usuarios**:
    - Registro de nuevos usuarios (createUser.php)
    - Autenticación de usuarios (checkUser.php)
2. **Gestión de partidas**:
    - Registro de partidas jugadas (createMatch.php)
    - Almacenamiento de puntuaciones (saveScore.php)
    - Obtención de puntuaciones más altas (getTopScores.php)
3. **Administración de la base de datos**:
    - Creación automática de tablas si no existen (getTopScores.php y saveScore.php crean la tabla scores si es necesario)
    - Manejo de conexiones centralizado (dbConnection.php)

5.2 Diseño

Modelo entidad relación:



Consideraciones de Diseño

1. **Normalización**: La base de datos sigue principios de normalización para evitar redundancias.
2. **Integridad referencial**: Se utilizan claves foráneas para mantener relaciones consistentes.
3. **Escalabilidad**: La estructura permite añadir fácilmente nuevas funcionalidades.
4. **Rendimiento**: Las consultas de puntuaciones utilizan ORDER BY y LIMIT para optimizar el rendimiento.

Esta estructura proporciona una base sólida para el sistema de juego, permitiendo la gestión segura de usuarios, el registro de partidas y el seguimiento de puntuaciones, con capacidad para expandirse según las necesidades futuras del proyecto.

6. Pruebas, errores cometidos y soluciones planteadas

6.1 Pruebas

Pruebas Funcionales: Comprobar que todas las mecánicas principales (movimiento,disparo,puntuación, cambio de fases) cumplan con con el propósito del videojuego. Se realizaron desde el editor de unity y usando emuladores de dispositivos móviles.

Pruebas de Integración: Esta está enfocada en el correcto funcionamiento con la API y la base de datos en MySQL , incluyendo el almacenamiento y la recuperación de datos.

Pruebas de Regresión: Aquí básicamente nos hemos centrado en que cada cambio mínimo aplicado en el proyecto no afectará negativamente a la funcionalidad y que tuviera un correcto funcionamiento.

Pruebas de Rendimiento: Centradas en comprobar la fluidez del juego y el consumo de memoria cada vez que se instancian GameObjects en la escena así como el tiempo de carga entre escenas.

Pruebas de Usuario: Con el objetivo de que se nos de una perspectiva general de qué problemas se han encontrado , y comprobar la usabilidad para un correcto funcionamiento.

6.2 Errores cometidos y soluciones

Establecimiento de límites: La nave que se controlaba con el joystick se salía del rango de la pantalla de la escena lo cual generaba mucha desubicación. Como solución implementamos límites de (max eje X, min eje X , max eje Y, min eje Y) dentro del script del playercontroller para que no se saliera del campo de visión de la escena.

Consumo excesivo de memoria: Los enemigos , los láseres de los enemigos y los del jugador al instanciarse seguían infinitamente activos en la escena. Para solucionarlo hemos metido en cada fase de la escena unos colliders tanto arriba como abajo para que cuando el enemigo , bala enemiga o del jugador se destruya y no consuma mucha memoria.

Partículas del jefe final al morir. En los enemigos cuando morían las partículas saltaban pero en las del jefe no salían. La solución fue ir a la opción de render y luego en order in layer subirle el nivel de la capa para que sea visible.

Instanciación de disparos cuando el enemigo está muerto: Al usar un audioSource en los enemigos cuando morían ,(Destroy(gameObject, boom.length) seguían instanciando disparos sin ningún tipo de sentido pero si le quitamos el componente de audio en la función(Destroy(gameObject) no se reproducía el sonido de la explosión. Como alternativa decidimos reducir el tiempo de duración del audio para que el timing cuadrada más o menos de esta forma (Destroy(gameObject, boom.length-7).

Errores de navegación: Algunos botones dirigían a otras escenas incorrectas o quedaban inactivos directamente. Reorganizamos la lógica de navegación aplicando el SceneManager.LoadScene de manera eficiente.

Interactividad con la barra de vida: Al estar en la escena de juego el slider de la barra de vida se podía deslizar permitiendo al jugador llenarse la vida lo cual era ilógico. La solución fue desmarcar la casilla Interactable para que no se pueda interactuar con ella.

Elementos activos al pausar el juego: Al pausar el juego básicamente quedaban elementos activos como : la barra de vida , la puntuación y el propio joystick. Como solución hemos implementado desde código una lógica en la que cuando le des al botón del pause se desactiven esos elementos con un [nombre].SetActive(false).

Área de los colliders descompensada : En algunos enemigos tenían los colliders más grandes de lo habitual lo cual hacía que la física al impactar con una bala del jugador fuera ilógica además de los láseres. Desde el editor de Unity se ha ajustado con precisión el área de los colliders de cada sprite enemigo.

Ajustamiento de diseño Responsive : Al probar el juego en distintos emuladores de dispositivos móviles se podía observar que estaban muy descompensados visualmente en el que en todos no ocupaba la pantalla completa de su resolución. Como recurso usamos el Canvas scaler que nos permite compensar el área de espaciado de la pantalla y luego usamos una propiedad de unity que permitía cubrir todo el ancho y alto de la pantalla fuera la resolución que sea manteniendo la proporción de los botones al tenerlos como hijos en la jerarquía del fondo , esto se hizo con el Strech.

Nivelación de sonidos desequilibrado : Al escucharse los efectos de sonido a la hora de jugarlo había algunos que estorbaban el audio de otros.
Como solución calibramos los sonidos de manera equilibrada para un mejor balance auditivo y que se escuchen todos los efectos ya sea a más bajo o alto volumen.

6.3 Conclusiones de las pruebas

Gracias a este proceso de pruebas , correcciones y errores se ha logrado una versión final mucho más funcional y optimizada con una navegación eficiente , buena jugabilidad y un consumo de recursos de las escenas menor. Los fallos encontrados han permitido mejorar la experiencia de usuario , la consistencia visual y la calidad técnica del producto. También esta etapa fue esencial para integrar en distintos dispositivos móviles garantizando la persistencia y mantenimiento de datos.

7. Conclusiones

Durante todo el desarrollo de este proyecto se ha buscado lograr un juego 2D para dispositivo móvil dinámico y funcional que permita que los usuarios estén enganchados.

Para conseguir este objetivo hemos trabajado con Unity version control , un sistema que permite trabajar remotamente con uno o varios compañeros en un solo proyecto y que además puede revertir a versiones anteriores en caso de fallos.

Hemos experimentado muchos problemas a la hora de aplicar la lógica de movimiento ,jugabilidad y fluidez eficiente cuyo arreglo nos ha permitido prepararnos mucho mejor de cara a futuros proyectos que realicemos.

Además se implementó una base de datos que está alojada en el servidor con el objetivo de ver las estadísticas de los jugadores , pero sobre para incluir un ranking de las 10 mejores puntuaciones.

Al finalizar el desarrollo de este videojuego , hemos podido lograr un producto fiable y eficiente que permite a los usuarios a conectarse online registrándose con sus cuentas para almacenar información sobre las estadísticas de sus partidas.

Con lo dicho previamente se puede decir que se han logrado todos los objetivos mencionados al principio del trabajo. Además por la investigación y manejo de las tecnologías mencionadas al principio se han obtenido conocimientos en este área para desarrollar videojuegos 2D.

Sin embargo el videojuego pese a cumplir con los requisitos propuestos en este proyecto, existen áreas que tienen margen de mejora que pueden añadirse y aplicarse a futuras versiones.Entre ellas añadir más animaciones , enemigos, una interfaz para configurar ajustes como: volumen, calidad , quitar música de fondo o incluso la capacidad de personalizar tus propias naves

Licencia

Este proyecto ha sido liberado como software libre bajo la licencia **MIT**. Esto permite su uso, modificación y distribución con fines personales, educativos o comerciales, siempre y cuando se conserve el aviso de copyright.

8.Bibliografía

Datos de la investigación:

Assets
Botones 2D | Unity Asset Store
Autor KartInnka, Abril 2, 2025
Buttons Set | 2D GUI | Unity Asset Store

Naves enemigas | Unity Asset Store
Autor MSGDI, Mayo 28, 2019
2D Spaceships Free Trial | 2D Environments | Unity Asset Store

Fondos | Unity Asset Store
Autor DinV Studio, Enero 17, 2018
Dynamic Space Background Lite | 2D Textures & Materials | Unity Asset Store

Botones 2D | Unity Asset Store
Autor Nayrissa, Septiembre 26, 2018
371 Simple Buttons Pack | 2D Icons | Unity Asset Store

Maquetación del diseño | Asset Free Space Shooter
Autor CraftPix, 30 Enero, 2019
Free Space Shooter Game GUI - CraftPix.net

Assets de fuentes y letras | Itch.io
Autor adamgda119, Febrero 01, 2025
https://itch.io/game-assets/free

Imagen espacio gratis | Google
fondo espacio - Búsqueda

Herramienta de Sprites

Creador de PixelArt | Piskel
Piskel - Free online sprite editor

Estilos
Fuentes de Google | Google Fonts
Browse Fonts - Google Fonts

Youtube Video

Touhou inspiración
Autor Phar111,24 Mayo 2009
Touhou 06 - EoSD Lunatic - Perfect Stage 6 Run

Patrones de Ataque inspiración Undertale
Autor Suwako Moriya,19 Septiembre 2015
[Undertale] Sans' boss fight - Genocide Run

Juego de naves inspiración video tutorial
Autor Ferry Dev, 8 Julio 2022
Prueba en unity con juego de naves

Tutorial de Base de datos de sistema de Login conectado a base de datos
Autor Unity Classroom, 20 Abril de 2018
Crear Sistema de Login y Registro Unity - 1 Instalando Herramientas

Tutorial de creación inicial de juegos de naves
Autor DelaloLab, 11 Enero de 2022
Como hacer un JUEGO de NAVES en UNITY

Tutorial de como crear una barra de vida
Autor BravePixelG, 5 Noviembre de 2021
Cómo crear una barra de vida en Unity

Tutorial para adaptar nuestro juego a cualquier resolución
Autor BAM APPS, 16 Octubre de 2023
https://www.youtube.com/watch?v=yciW-SWVpfY&t=103s

Tutorial de el Spawn de enemigos aleatorios

Autor Luis Canary, 16 Noviembre de 2021
Como CREAR un VIDEOJUEGO con SPAWN ENEMIGOS de manera ALEATORIA! □□□ (Mini-Serie) / 1-Capitulo

Tutorial de el Spawn de enemigos aleatorios (2)
Autor Luis Canary, 20 Noviembre de 2021
Como CREAR un VIDEOJUEGO con SPAWN ENEMIGOS de manera ALEATORIA! □□□ (Mini-Serie) / 2-Capitulo

Tutorial de como crear un Joystick
Autor DrosGame, 8 mayo 2020
□APRENDE como crear JOYSTICK TOUCH para CONTROL de PERSONAJES [BASICO] en UNITY 2D | FACIL [2022]□

Tutorial de como poner un video en 2D
Autor jhonosferatu, 24 de mayo de 2023
Como Poner Video en Unity 2D

Proyectos de Github

Inspiración de Github Space Shooter
Autor warekein, 2022
warekein/Space_Shooter: A simple space shooter demo