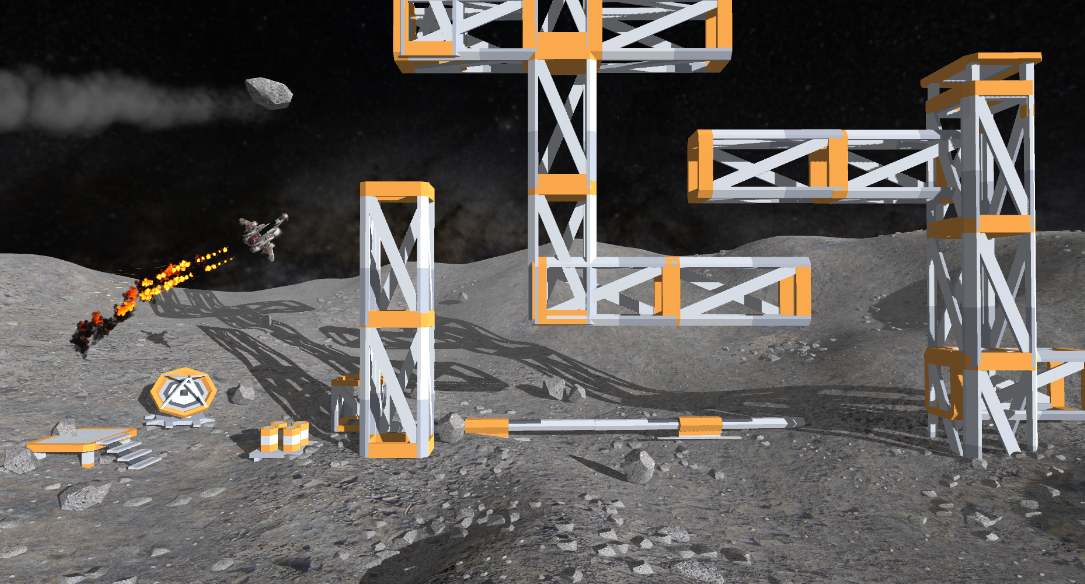
2024

Igor Gebauer Garcia

Proyecto Final de Ciclo

22-2-2024

MoonLanding



**Índice de contenidos**

1. Introducción........................................................................................................... 3

1.1. Motivación.......................................................................................................... 3

1.3. Objetivos propuestos (generales y específicos). ................................................ 5

2. Metodología utilizada............................................................................................. 6

3. Tecnologías y herramientas utilizadas en el proyecto............................................ 7

4. Estimación de recursos y planificación. ................................................................. 9

5. Concepto del juego…….......................................................................................... 10

5.1. Historia y personajes..........................................................................................10

5.2. Mecánicas del juego...........................................................................................10

5.3. Niveles y diseño del entorno..............................................................................10

5.4. Arte y sonido......................................................................................................12

5.5. Interfaz de usuario (UI)………..............................................................................12

5.6. Progresión del jugador.......................................................................................15

6. Implementación.................................................................................................... 17

7. Despliegue y pruebas ............................................................................................ 27

7.1. Plan de pruebas.................................................................................................27

8. Conclusiones. ........................................................................................................ 31

8.1. Objetivos alcanzados. ........................................................................................31

8.2. Conclusiones del trabajo. ..................................................................................31

8.3. Vías futuras...................................................................................................... 31

9. Glosario................................................................................................................ 33

10. Webgrafía........................................................................................................... 34

**1. Introducción**

Mi proyecto final para el Ciclo Formativo de Grado Superior en Desarrollo de Aplicaciones Multiplataforma tiene como meta principal el diseño y la creación de una versión actualizada del famoso juego "MoonLanding". Mi intención con este proyecto va más allá de simplemente traer de vuelta la magia de uno de los juegos fundacionales de la industria; busco fusionarlo con elementos educativos que aborden la física espacial y la navegación, para así brindar una experiencia tanto interactiva como instructiva.

He decidido reinterpretar "MoonLanding" para adaptarlo a los nuevos tiempos, integrando mecánicas de juego intuitivas junto con una representación precisa de las leyes de la física, lo que supone un reto para los jugadores al intentar alunizar. Este enfoque permitirá a los usuarios adentrarse en conceptos como la gravedad, el impulso, y el manejo de distancias y nombres espaciales, enfrentándose a retos similares a los que encontrarían astronautas e ingenieros en misiones lunares actuales.

Para llevar a cabo este proyecto, opté por utilizar Unity, que facilita la creación de juegos contemporáneos y aseguran su disponibilidad tanto en PC como en dispositivos móviles.

* 1. **Motivación**

La motivación detrás de este proyecto es ofrecer una experiencia de juego entretenida y competitiva, mientras se aprenden algunas nociones del espacio. También pretende educar sobre la física y los desafíos de la navegación espacial de una manera accesible y atractiva(funcionalidades que se irán implementando en futuras versiones, no son prioritarias en esta primera versión beta). Los usuarios adquirirán conocimientos fundamentales sobre gravedad, trayectoria y control de naves espaciales, todo ello dentro de un contexto emocionante y desafiante.

* 1. **Objetivos propuestos (generales y específicos)**
* **Gráficos y Animaciones Mejoradas**: Actualización gráfica y de animaciones respecto la versión original, usando asets 3D y partículas de alta calidad.
* **Niveles con Diferentes Grados de Dificultad**: Diseño de niveles que varían no solo en la disposición de obstáculos y la ubicación de la zona de aterrizaje, sino también en condiciones ambientales que afectan la física del juego, como tormentas solares o asteroides.
* **Modo Educativo**: Sistema de información basado en interfaces que proporciona información sobre la física espacial, la historia de la exploración lunar y datos curiosos sobre el espacio. Se pretende implementar a través de mini-lecciones accesibles en puntos específicos del juego o como recompensas por completar niveles.
* **Personalización de la Nave:** Permite a los jugadores personalizar su nave espacial, no solo estéticamente sino también en aspectos funcionales que afectan al juego, como el tipo de combustible, la resistencia del casco o la potencia del motor.
* **Logros y Sistema de Puntuación:** Sistema de logros y puntuación que motiva a los jugadores a mejorar sus habilidades y a completar objetivos específicos, como aterrizar sin daños, conservar el máximo de combustible posible o completar un nivel en un tiempo récord.
* **Modo Multijugador Competitivo y Cooperativo:** Los jugadores pueden competir por la mejor puntuación en aterrizajes o colaborar en misiones espaciales y ver la repetición del adversario con un fantasma de tu recorrido.
* **Interactividad con el Entorno:** Elementos interactivos en los niveles, como estaciones de combustible para reabastecerse o monedas que los jugadores pueden recoger para obtener puntos extra o información.
* **Adaptabilidad a Diferentes Plataformas**: El juego esta optimizado para funcionar en una variedad de dispositivos, como PCs y dispositivos móviles, adaptando los controles y la interfaz según sea necesario.

1. **Metodología utilizada**

La metodología adoptada en el desarrollo de "MoonLanding" se basa en principios ágiles, enfocándose en la iteración rápida, la flexibilidad en la planificación, y la respuesta ante el cambio. Este enfoque permite adaptar el proyecto a nuevos requerimientos o desafíos que surgen durante el desarrollo. Se seguirán estos pasos principales:

* **Planificación Inicial:** Definición de los objetivos del juego y las características clave que debería tener el juego.
* **Diseño Iterativo:** Empleo de ciclos de diseño, desarrollo y prueba iterativos. Cada ciclo incluye la creación de prototipos, pruebas, y recopilación de feedback para mejorar(lo monto, se prueba, no gusta, se cambia, y así).
* **Desarrollo Incremental:** Construcción del juego por módulos, permitiendo enfocar en una característica o componente a la vez y asegurando que cada parte funcione correctamente antes de pasar a la siguiente.
* **Revisión y Refinamiento**: Al final de cada iteración, se revisa el progreso y se refinan los objetivos y prioridades basado en lo concluido.
* **Pruebas Continuas:** Para identificar y corregir errores en el proceso de desarrollo.

1. **Tecnologías y herramientas utilizadas en el proyecto**

El desarrollo de "MoonLanding" se apoya en varias tecnologías y herramientas clave para facilitar el diseño, desarrollo, y pruebas del juego:

* **Unity**: Como plataforma de desarrollo principal, Unity permite crear una experiencia de juego rica e interactiva gracias a su potente motor de física, sistema de partículas para efectos visuales, y compatibilidad con múltiples plataformas.
* **C#:** Se utiliza como lenguaje de programación principal para el desarrollo de scripts en Unity, permitiendo implementar la lógica del juego, controles de la nave, y mecánicas de juego.
* **Plugins:** Para la creación de modelos 3D, como la nave espacial y el terreno lunar, se emplea principalmente plugins de unity. Esto permite dar al juego un estilo visual único y atractivo con bastante rapidez.
* **Git:** Se usa para el control de versiones y la colaboración en equipo, facilitando el seguimiento de cambios y la gestión del código fuente del proyecto.
* **Visual Studio Code**: Como editor de código, proporciona un entorno de desarrollo integrado (IDE) eficiente para escribir y depurar los scripts de C#.
* **Audacity:** Para la edición de efectos de sonido y música de fondo, se utiliza Audacity, mejorando la atmósfera y la experiencia inmersiva del juego.
* **FireBase:** Como backend para el sistema login y base de datos para la recopilación de datos de los jugadores.

1. **Estimación de recursos y planificación**

* Recursos Humanos: El proyecto se desarrolla individualmente, con el apoyo ocasional de mentores y compañeros para revisión de código y feedback.
* Recursos de Software: Se utiliza “Unity 2022.3.20f1” como motor de juego, plugins para assets 3D y particulas, Audacity para edición de audio, FireBase como backend para el login y base de datos y Visual Studio Code como IDE para programación en C#.
* Recursos de Hardware: Un ordenador con especificaciones capaces de ejecutar Unity sin problemas, y dispositivos móviles y PC para pruebas de juego.
* Tiempo: El proyecto se divide en fases: una semana para el diseño y prototipado inicial, una semana para el desarrollo principal y la integración de mecánicas de juego, y una semana final para pruebas, pulido y documentación.

1. **Concepto del juego**

"MoonLanding" es una modernización del clásico juego de arcade. En este juego, los jugadores asumen el rol de astronautas encargados de pilotar una nave espacial para aterrizar en varios sitios lunares distintos. El juego pone a prueba la habilidad del jugador para controlar la nave bajo la influencia de la gravedad lunar, gestionar el combustible y navegar entre obstáculos hasta llegar a la plataforma de aterrizaje.

**Características Principales:**

* **Mecánicas de Física Realista:** El juego simula la gravedad lunar, el impulso y la inercia.
* **Niveles Progresivamente Difíciles:** Los niveles aumentan en dificultad, introduciendo obstáculos dinámicos, como meteoritos o cráteres, y condiciones variables, como tormentas solares.
* **Modo Educativo:** El juego incluye un modo educativo que proporciona información sobre la física espacial, la historia de la exploración lunar y curiosidades sobre el espacio en determinados puntos de la partida.
* **Personalización de la Nave:** Los jugadores pueden personalizar su nave espacial, ajustando aspectos como la apariencia y las características técnicas para mejorar la maniobrabilidad o la eficiencia del combustible.
  1. **Historia y personajes**

**Historia:**

En una futura era de exploración espacial, la humanidad ha vuelto su vista hacia las Lunas como un paso crucial para la expansión y exploración espacial. Los jugadores asumen el papel de pilotos de élite en la "Corporación Lunar de Exploración" (CLE), una organización encargada de establecer bases avanzadas en Lunas. La misión es clara: aterrizar con seguridad en designadas zonas lunares, superando desafíos naturales y artificiales..

**Personajes:**

* Axel Bauer: El protagonista principal, un piloto experimentado con un talento natural para la navegación espacial y una pasión por descubrir los secretos del universo.
* Dra. Jessenia: La científica jefe de la CLE, guiando a Axel con información crucial sobre la superficie lunar y consejos sobre cómo manejar las mecánicas en juego.
  1. **Mecánicas del juego**
* Control de la Nave: Los jugadores deben usar las teclas de flecha o controles táctiles para manejar la dirección y la potencia del impulso de la nave, teniendo en cuenta la gravedad lunar y la inercia.
* Gestión de Combustible: Un recurso limitado que los jugadores deben elegir y administrar para completar sus misiones.
* Navegación y Exploración: Identificar la ruta óptima hacia las zonas de aterrizaje, evitando obstáculos.
* Sistema de Puntuación: Ganan puntos por uso eficiente del combustible, las vidas totales y la rapidez en completar el nivel.
  1. **Niveles y diseño del entorno**

Para "MoonLanding", he diseñado una serie de niveles que aumentan progresivamente en dificultad, cada uno presenta desafíos únicos y un entorno visualmente distinto. Los niveles comienzan en áreas relativamente abiertas y simples para aterrizar, progresando hacia zonas más complejas con obstáculos He utilizado Unity y plugins de Unity para crear estos entornos**.**

**Nivel 1:** Nivel inicial con dificultad +.

**Nivel 2:** Nivel inicial con dificultad ++.

**Nivel 3:** Nivel inicial con dificultad +++.

**Nivel 4:** Nivel inicial con dificultad ++++.

**Nivel 5:** Nivel inicial con dificultad +++++.

**Nivel 6:** Nivel inicial con dificultad ++++++.

* 1. **Arte y sonido**

He utilizado plugins para añadir modelos 3D detallados de la nave y los elementos del entorno. En cuanto al sonido, he seleccionado una banda sonora que complementa la tensión y la emoción, junto con efectos de sonido para el motor de la nave, los aterrizajes y los choques, creados y editados algunos en Audacity para añadir profundidad a la experiencia de juego.

* 1. **Interfaz de usuario (UI)**

La interfaz de usuario en "MoonLanding" es intuitiva y minimalista para no distraer al jugador de la experiencia de juego. Consta de una HUD que muestra información crítica como el nivel de combustible, velocidad, altitud y vidas. Los menús de inicio, pausa y juego completado son accesibles y simples.

* 1. **Progresión del jugador**

La progresión en "MoonLanding" está planteada para ser desafiante pero justa, con el jugador desbloqueando nuevos niveles y mejoras para la nave a medida que demuestran su habilidad. Consta de un sistema de puntos basado en la eficiencia del combustible, la precisión del aterrizaje y el tiempo completado, incentivando a los jugadores a mejorar su técnica. Las mejoras incluirán mayor eficiencia de combustible, mejor manejo y resistencia a impactos, lo que permitirá a los jugadores personalizar su experiencia de juego según su estilo.

1. **Implementación**

Para la implementación de "MoonLanding", he seguido un enfoque iterativo, comenzando con el desarrollo de prototipos básicos de las mecánicas principales en Unity. Estas incluyen el control de la nave, la simulación de la gravedad lunar, y la detección de colisiones. A medida que validaba cada mecánica, integraba los modelos 3D y los efectos de sonido para empezar a dar vida al ambiente del juego.

**SC\_ControladorNave2D**

SC\_ControladorNave2D fue el corazón de mi implementación inicial, estableciendo cómo el jugador interactua con la nave. Desde el inicio, quise que la experiencia de control fuera intuitiva pero desafiante, equilibrando la física de la nave con la entrada del usuario para crear un sistema de vuelo interesante, asi que añadi las rotación y desplazamiento lateral obligando al jugador a utilizar las 2 manos si quiere tener el control total de la nave.

**SC\_Combustible**

La clase SC\_Combustible se desarrolló después, pero en conjunto con SC\_ControladorNave2D. La necesidad de hacer que el vuelo acabe en algún momento me llevo a añadir esta clase, definiendo un consumo determinado máximo que puede hacer la nave, luego para el impulso de la nave si no hay combustible. Esta clase también se comunica con SC\_Puntuacion para entregar el combustible restante y con la UI para reflejarlo en la barra de combustible del HUD.

**SC\_MusicController**

Para implementar la música y los efectos de sonido de "MoonLanding" desarrollé SC\_MusicController. Con ello pude controlar fácilmente la reproducción de audio dependiendo de las acciones del jugador y los eventos del juego, como al impulsar la nave, la música de fondo, las colisiones con los objetos o la victoria o la derrota también tienen sonidos únicos gestionados en esta clase.

**SC\_SpawnAsteroide**

Implemente esta clase a los inicios del desarrollo buscando tener un principal spawn de obstáculos móviles, de aquí extrapole al obstáculo “proyectil” reciclando esta clase y las del asteroide. Se encarga de spawnear el prefab asteroide en un intervalo de tiempo que podemos definir en el inspector. El resto de la magia la hacen las otras clases del asteroide.

**SC\_MovimientoAsteroide y SC\_RotacionAsteroide**

Desarrollé estas clases cuando comencé a introducir obstáculos en los niveles. Los asteroides, con su movimiento y rotación, añaden desafío y dinamismo al entorno de juego. Su implementación fue un punto de inflexión para aumentar la dificultad y

**SC\_CambiadorDeNave**

Implementada en una fase madura del desarrollo al verme con el problema de no poder cambiar el mesh de mi prefab según la elección del player, entonces decidí tener 3 prefabs y que esta clase spawnee el prefab elegido por el player en el menú configuración de la nave. Me llevo a otros problemas que resolvi con otras clases.

**SC\_Vida**

Implementé SC\_Vida después de establecer las mecánicas básicas de movimiento. Fue crucial para añadir desafío al juego, gestiona el sistema de vidas del jugador y responde a las colisiones con los obstáculos. Esta clase también actualiza la UI para reflejar el estado actual de vida del jugador y se comunica con SC\_Puntuacion para recoger las vidas que quedan y sacar el cálculo de la puntuación.

**SC\_ActivadorDeColliderYParticulas**

Implementada en mitad del desarrollo, activa los efectos visuales y colisiones de ciertos obstáculos con particulas de vapor o llamaradas, enriqueciendo las interacciones del juego y sincronizando la colision con el principio del pley de las particulas.

**SC\_CamaraFollow**

La necesidad de SC\_CamaraFollow apareció en el momento en el que decidí hacer que la nave spawneara según la que hayas elegido en el menú configuración, entonces nunca hay una nave en el nivel para indicarle a la cámara que siga a esa nave, asi que tuve que crear esta clase para indicarle por script que la cámara siga a la nave que spawnee en cada nivel. Solucion simple pero efectiva.

**SC\_ControladorLuz**

Introduje esta clase para darle más relevancia al punto de inicio y de aterrizaje. Su objetivo es oscilar la intensidad de la luz en bucle infinito haciendo que parpadee sobre las plataformas para llamar la atención(efecto emisivo pero con luces, ya que con los materiales me salió una cosa muy rara y fea). A través del Feedback recibido me obligaron a añadir esta solución.

variedad de los niveles. Tube que dividir la rotación y el movimiento en clases distintas por conflictos con las particulas del asteroide(el humo debia ir sin rotación).

**SC\_MenuInicial, SC\_MenuJuego y SC\_MenuPausa**

Las interfaces las deje casi para el final, son lo que menos me gusta. Estas clases tienen todos los métodos para el comportamiento de las interfaces, desde el menú de pausa dentro del juego, hasta el menú inicial que te lleva a elegir la nave y el menú de juego(tutorial) que solo aparece en el primer nivel.

**SC\_FadeIn & SC\_FadeOut**

Implementé los Fades a través de estas clases para mejorar las transiciones entre escenas y menús. Utilizando el Animator de Unity, creé animaciones suaves que se activan al iniciar cada nivel, al darle al botón pausa(ESC) y al interactuar con las interfaces.

**SC\_Puntuacion**

Esta clase se creó hacia la fase final del desarrollo. Su objetivo esta claro, calcular y mostrar la puntuación que la calcula llamando a la vida, el combustible y el tiempo tardado. Luego se comunica con la UI para mostrarlo en un TextBox del HUD.

1. **Despliegue y pruebas**

En esta sección describo el plan de pruebas del proyecto "MoonLanding", asegurando que cada aspecto funciona correctamente y ofrece una experiencia de usuario óptima. Las pruebas se dividen en: pruebas funcionales, pruebas de usabilidad, pruebas de rendimiento, pruebas de estrés y feedback.

**1.- Pruebas Funcionales:**

Con estas pruebas se pretende garantizar el correcto funcionamiento de todos los elementos del juego, incluidos controles, mecánicas y UI.

* **Test de Controles:**

Inicialmente empecé por la nave y la entrada del usuario, probando en la plataforma Windows una y otra vez la clase controladora del player SC\_ControladorNave2D con las teclas A,S,D y W y las flechas tanto el giro lateral como los impulsos frontales y laterales de la nave.

Posteriormente realice las mismas pruebas para Android desde Windows con el ratón sobre la interfaz diseñada para Android y sus controles, luego probándolo en mi dispositivo móvil verificando que reaccionaba correctamente a los toques de la pantalla, depurando con mucho Debug.Log para interpretar cuando se estaban llamando y cuando no los métodos de movimiento.

Por último, añadí partículas y sonidos para los motores de la nave controlándolos desde la clase SC\_ControladorNave2D y probándolo repetidamente hasta conseguirlo sin conflictos.

* **Test de Mecánicas de Juego:**

Construí la clase SC\_Combustible para darle “vida” al movimiento del jugador y realicé pruebas en el mismo nivel plantilla para comprobar el consumo de la nave que reaccionaba adecuadamente a las teclas y a los botones táctiles.

Luego construí la clase SC\_Puntuacion para el cálculo de puntuaciones basado en el combustible restante, tiempo empleado, y vidas testeándolo de la misma manera.

También añadí assets que hacían de obstáculos móviles (proyectil, asteroide) y reaccionaban con partículas al colisionar, me llevo muchísimas pruebas hasta conseguir el comportamiento aceptable al que he llegado.

También añadí algunos assets con colisiones y partículas de vapor temporizadas, me llevo un tiempo conseguir que la caja de colisión se sincronizara con las partículas de vapor.

* **Test de UI:**

Fui creando primero la interfaz de inicio del juego con el botón simple, luego añadí la pantalla de configuración de la nave, luego el login y luego el registro. Recorrí todos los menús y pantallas para confirmar que cada elemento es visible, accesible y funciona según lo diseñado en el momento que se espera.

* **Test FireBase:**

Primero fue FireBase.Analytics, comprobando con el proyecto compilado desde mi dispositivo Android que el cliente web de Google-Analitics mostraba los datos de uso y de conexión.

Luego fue FireBase.Authenticator para el login, las pruebas las hice desde el cliente web comprobando que el registro que hacia en mi App se reflejaban en el cliente web de FireBase exitosamente. El login de la misma manera, solo se logueaba con dichos registros.

Por último, fue FireBase.DataBase para la recopilación de los datos del jugador.

**2.- Pruebas de Usabilidad**

El objetivo es observar a jugadores interactuando con el juego e identificar áreas a mejorar, priorizando por las más importantes por si no llegaba en tiempo.

Principalmente mis betatester fue hermano, algunos amigos y compañeros del curro. Despues de cada compilado con mejoras, aleatoriamente obligaba a alguna de mis victimas a disfrutar el juego mientras escuchaba y anotaba tanto las sugerencias a mejora como los errores que detectaban. Corrijo y vuelvo a lo mismo.

**3.- Pruebas de Rendimiento**

El objetivo es asegurar que el juego funcione fluidamente en varios dispositivos en distintas plataformas con una tasa de frame sostenida.

Plataforma Android: Probado en mi RedMi Note 10 Pro, un Samgsumg Galaxy S21 y un Oppo A92. Todos fluían correctamente excepto en algún spawneo masivo de particulas en alguna compilación rara.

Plataforma Windows: Testeado en mi ordenador y varios más de otros compañeros, sin ningún problema por el sentido lowpoly del proyecto.

**4.- Pruebas de Estrés**

El objetivo de estas pruebas es poner el dispositivo en extremos para comprobar su comportamiento y corregirlo en la medida de lo posible.  
  
Probe el proyecto en Android y en Portátil con poca batería a ver qué pasaba sin ningún problema aparente, más que el móvil se acaba apagando y pc suspendiendo.

También se probó el máximo spawneo de asteroides para ver cuál era el límite del móvil gráficamente, y es bastante corto, al tercer asteroide que colisiona a la ves las partículas de explosión consumen demasiado para Android.

**5.- Feedback**

El objetivo del feedback es utilizar la retroalimentación para iteraciones futuras del juego. Principalmente me dedicaba a escuchar mientras se intentaban pasar mi juego las risas y cualquier emoción llamativa, generalmente ocurría cuando la nave se daba una ostia fuerte y giraba muy rápido sobre su eje mientras el asteroide vuela hacia otro lado, intente mejorar esa experiencia.

* 1. **Plan de pruebas**
* Pruebas Funcionales: Verificar que todos los elementos del juego (controles, mecánicas, UI) funcionen como se espera.
* Pruebas de Usabilidad: Observar a los jugadores mientras interactúan con el juego para identificar errores o confusión.
* Pruebas de Rendimiento: Asegurar que el juego mantenga una tasa de fotogramas estable en diferentes dispositivos.
* Pruebas de Estrés: Evaluar cómo se comporta el juego bajo condiciones extremas, como bajos niveles de batería.

1. **Conclusiones**
   1. **Objetivos alcanzados**

A lo largo de este proyecto, logre varios objetivos cruciales para el desarrollo exitoso de "MoonLanding". Primero, se implementó una versión funcional y jugable del clásico, imitando mecánicas esenciales como el control de la nave, lo que me proporcionó una base para el proyecto. Segundo, se estableció una estructura de proyecto en Unity organizada a mi manera, facilitando el mantenimiento y el crecimiento del código. Tercero, se realizaron pruebas funcionales y de usabilidad, asegurando que el juego no solo cumple con las expectativas funcionales, también ofrece una experiencia divertida.

No se ha podido llegar a implementar el sistema multiplayer ni muchas otras cosas que me hubiera gustado tocar, pero es el inicio de algo más.

* 1. **Conclusiones del trabajo**

Este proyecto ha sido una gran oportunidad para aplicar y profundizar mis conocimientos en desarrollo de videojuegos con Unity. La implementación de mecánicas de juego y la resolución de problemas específicos, como la gestión de la interfaz de usuario y la optimización del comportamiento de las coliones, han añadido un par de sótanos más a mi paciencia y ha enriquecido mi comprensión de lo que implica desarrollar un juego en general. Este proyecto ha reforzado mi capacidad para trabajar de manera estructurada y adaptativa, preparándome para proyectos más complejos en el futuro .

Sobre todo, lo mas importante, si no fuera por este proyecto… actualmente en mi puesto de trabajo me hubiera cortado las venas, ya que me han cambiado las funciones recientemente y en ves de desarrollar en Unreal ahora me toca desarrollar lo que tenemos en Unreal pues en Unity, y gracias a este proyecto he podido entender más rápido Unity.

* 1. **Vías futuras**

1. **Glosario**

**Unity**: Motor de desarrollo de juegos ampliamente utilizado que permite la creación de videojuegos en 2D, 3D, realidad virtual y realidad aumentada, ofreciendo un conjunto robusto de herramientas para desarrolladores.

**Prefab**: Objeto de juego preconfigurado en Unity que puede incluir componentes y propiedades establecidas, permitiendo su reutilización a lo largo del proyecto.

**Scene**: Nivel dentro de Unity donde se colocan y organizan los elementos del juego, como personajes, objetos y cámaras.

**Canvas**: Componente de Unity utilizado para organizar y presentar elementos de la interfaz de usuario (UI) en una escena.

**Scripts**: Archivo de código, generalmente escrito en C#, utilizado en Unity para definir el comportamiento y la lógica de los objetos de juego y otros aspectos del proyecto.

**Firebase**: Plataforma de desarrollo de aplicaciones de Google que ofrece funcionalidades backend como bases de datos en tiempo real, autenticación de usuarios y análisis, utilizada en este proyecto para gestionar el login y recopilar datos de los jugadores.

1. **Webgrafía**

**Documentación oficial de Unity:** Proporciona una guía exhaustiva y tutoriales sobre cómo utilizar Unity y sus características.

**Documentación oficial Firebase para Unity:** Tutoriales y documentación sobre cómo integrar Firebase en proyectos de Unity, cubriendo autenticación, bases de datos y más.

**Stack Overflow:** Comunidad de desarrolladores con una vasta cantidad de preguntas y respuestas sobre programación en Unity y otros temas relacionados.

**YouTube** - Videos tutoriales de diferentes tipos de distintos canales.

**ChatGPT:** La herramienta más útil cuando no te quedan ideas, “la calidad de tu vida depende de la calidad de tus preguntas” – cita de no recuerdo quien que me apeteció añadir.

**Otras Webs:** Distintas webs googleadas como “redit” de las que se extrajeron recursos o información útil de algún tipo.