UNIVERSITAT POLITÈCNICA DE VALÈNCIA

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Politécnica Superior de Gandia

Aplicación de un entorno de aprendizaje interactivo adaptable a las necesidades del estudiante

Trabajo Fin de Grado

Grado en Tecnologías Interactivas

AUTOR/A: Belloch Martinez, Luis

Tutor/a: Palanca Cámara, Javier

Cotutor/a externo: LOPEZ BALLESTER, JHOAN MANUEL

CURSO ACADÉMICO: 2022/2023

Resumen

Las aplicaciones educativas están cada vez más presentes en las aulas y en los teléfonos móviles de los alumnos. Comúnmente estas aplicaciones están diseñadas con el único propósito de enseñar al usuario el temario que tiene preparado o apoyar al docente en su labor. El objetivo de este proyecto es que la aplicación sea capaz de acompañar al alumno y ofrecer herramientas al docente para facilitar el aprendizaje de una forma digital e interactiva. En lugar de únicamente centrarse en contenidos esta aplicación se centra en interactividad como medio de aprendizaje con el potencial de implementar cualquier contenido.

Palabras clave: ludificación, móvil, física, aprendizaje, interactividad.

Abstract

Educational applications are increasingly present in classrooms and on students' mobile phones. Commonly these applications are designed with the sole purpose of teaching the user the subject matter they have prepared or to support the teacher in their work. The aim of this project is for the application to be able to accompany the student and offer tools to the teacher to facilitate learning in a digital and interactive way. Instead of focusing solely on content, this application focuses on interactivity as a means of learning with the potential of implementing any kind of content.

Keywords: gamification, mobile, physics, learning, interactivity.

Índice general

Índice de figuras

Índice de tablas

1 Introducción

Este Trabajo de Fin de Grado describe el proceso de desarrollo de un corte vertical de un videojuego educativo o aplicación educativa que aplica elementos de ludificación para apoyar a alumnos de la ESO y bachiller en el aprendizaje de conceptos de física y matemáticas.

Como anexo a este trabajo se ha creado una guía docente que sirve como ayuda para implementar esta aplicación en las aulas como herramienta de aprendizaje con menos esfuerzo por parte del personal docente.

En este capítulo se introducen la motivación para realizar el proyecto y los objetivos del mismo.

1.1 Motivación

La ludificación o gamificación como concepto surgió en 2008 y desde entonces nunca se han dejado de discutir sus posibles aplicaciones en la educación. Sin embargo en más de una década no se ha visto una voluntad generalizada de añadir ludificación a las aulas en gran medida (Zaman & van Roy, 2016, 2-4) por la falta de aplicaciones dedicadas al apoyo en la educación y errores fundamentales al diseñar la metodología educativa. Sí se utilizan algunas aplicaciones de apoyo en la organización de clases como Class Dojo o en enseñanza de conceptos de informática como Minecraft Classroom pero no ha llegado a extenderse a la enseñanza de otras asignaturas, al menos de una forma generalizada.

1.2 Objetivos

El objetivo principal de este proyecto es el desarrollo de un corte vertical de una aplicación educativa que aplique estrategias de ludificación para alumnos de bachillerato y ESO en asignaturas de ciencias. Para cumplir este objetivo se plantean los siguientes subobjetivos:

- Análisis del estado del arte de las aplicaciones educativas para descubrir fortalezas y

carencias.

- Analizar los requisitos de funcionalidad y estética de la aplicación a desarrollar.
- Diseñar e implementar la experiencia de usuario de la aplicación.
- Desarrollar la aplicación móvil.
- Evaluar la efectividad de la aplicación como herramienta de aprendizaje.

2 Marco teórico

En este capítulo se detalla el análisis del estado del arte de las aplicaciones educativas más relevantes para este proyecto y una comparativa de los motores para decidir cual usar para el desarrollo de la aplicación.

2.1 Estado del arte

El análisis del panorama de las aplicaciones educativas se puede desglosar en dos categorías: las aplicaciones de apoyo al profesorado y las de enseñanza. Las aplicaciones de apoyo requieren de la presencia del personal docente para su correcto funcionamiento, tales como Kahoot o Google Classroom. Aunque son útiles para organizar y dinamizar el aula, no ofrecen una estrategia pedagógica y en ocasiones, carecen de contenidos relevantes. Por otro lado, existen aplicaciones que sí presentan una metodología para enseñar a los usuarios habilidades y conocimientos específicos, entre las que se destacan Khan Academy, Lumosity y Brilliant (Cherner, 2014).

Como investigación preliminar se han analizado varias de estas aplicaciones con el objetivo de encontrar carencias y similitudes para comprender mejor el estado actual de las aplicaciones educativas. Se han seleccionado estas aplicaciones ya que son las más publicitadas y a priori las que más usuarios simultáneos tienen. En particular se van a analizar las que no requieren de un docente para ser funcionales.

2.1.1 Khan Academy

Khan Academy es una plataforma en línea que fue creada en 2008. Ofrece una variedad de videos instructivos y ejercicios de práctica en materias como matemáticas, ciencias, economía y computación. Es adecuada para estudiantes de todas las edades, ya que las lecciones de matemáticas comienzan desde el nivel primario con operaciones básicas como sumas y restas, y llegan hasta la universidad con temas como ecuaciones diferenciales.

El temario es generalista y se enfoca en enseñar todos conceptos separados por lecciones, temáticas y cursos educativos. Esto mismo sucede con los apartados de ciencia (física, química, cosmología e ingeniería eléctrica), economía y computación. Aunque estos dos últimos dan la apariencia de estar en desarrollo todavía ya que hay pocos contenidos.

Al seleccionar una lección empieza el proceso.

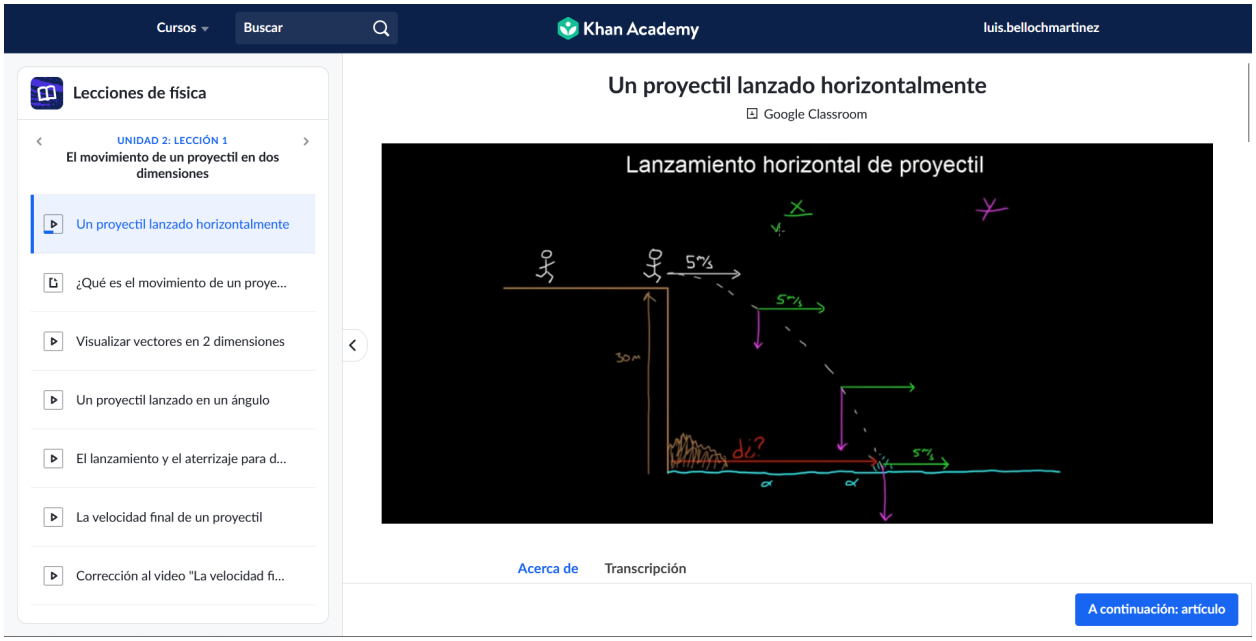

Figura 2.1. Captura de pantalla del primer apartado.

Al iniciar cada lección la pantalla muestra a un lado los próximos apartados para poder ver el progreso actual y saltar a otros apartados al instante y una grabación explicativa, en este caso “El movimiento de un proyectil en dos dimensiones” (Figura 2.1).

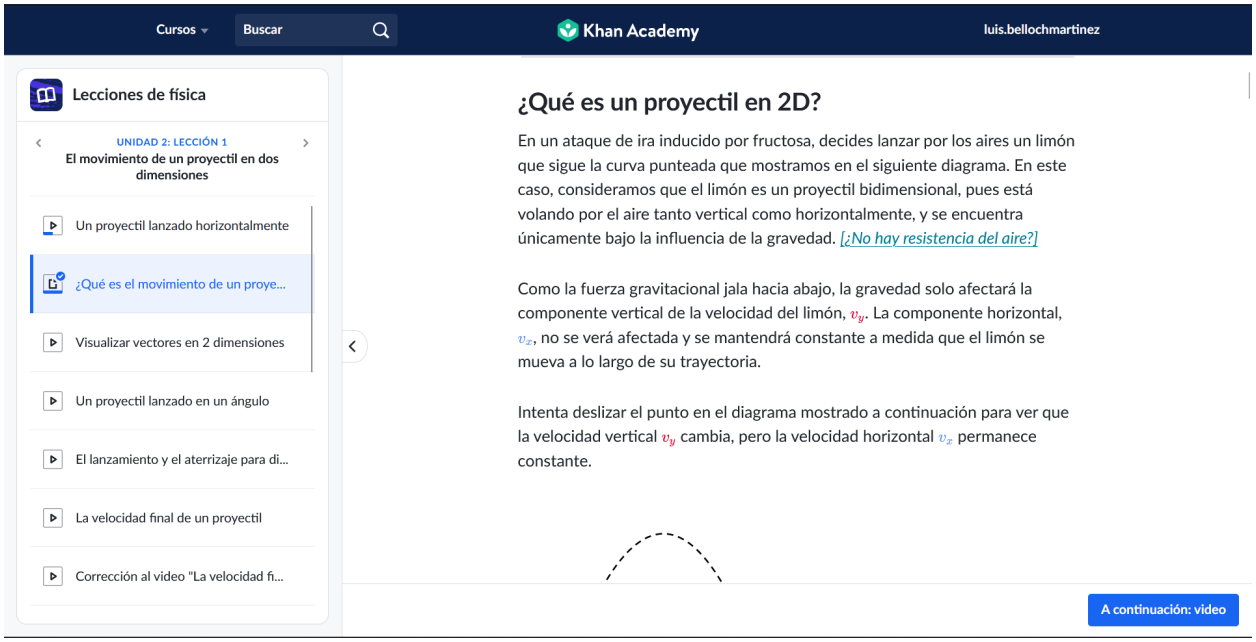

Figura 2.2. Captura de pantalla de una página de lección de texto.

En este primer apartado explicativo contiene una gran cantidad de texto plano con algunos cambios de color en las letras para señalar elementos relevantes (Figura 2.2).

Figura 2.3. Captura de pantalla del enunciado de un problema.

Hay 3 tipos de apartados: de vídeo, de texto (Figura 2.2) y de ejercicio (Figura 2.3). En los apartados de texto se explican en varios párrafos con diagramas y símbolos los conceptos

en más profundidad que en el vídeo explicativo. En este caso el vídeo sirve de explicación práctica y el texto de explicación teórica. Es destacable que en el apartado de texto (Figura 2.2) se usan colores para resaltar los símbolos y conceptos importantes lo que hace que una lectura rápida sea más efectiva, ya que puedes buscar dónde están las partes importantes de un vistazo. También es útil para evitar el cansancio visual al leer un bloque de texto negro.

En el apartado de ejercicio se proponen unas pocas preguntas a modo de test con varias opciones. En este ejercicio en particular todas las opciones son gráficas. De esta manera es muy visual con el inconveniente de resultar muy superficial. Ya que no se muestran posibles soluciones numéricas sólo se evalúa la comprensión superficial sobre el tema del alumno, no se le reta realmente en este nivel.

2.1.2 Brilliant

Brilliant fue lanzada en 2012 y cuenta con el reconocimiento de haber formado parte de una “revolución matemática” en Estados Unidos (Tyre & Buder, 2016). Los contenidos de Brilliant se centran en matemáticas y física con explicaciones sencillas y muy gráficas para conceptos avanzados como álgebra matricial o sistemas de engranajes.

Los contenidos están enfocados a alumnos de secundaria y universidad con un aspecto de profesionalidad y seriedad.

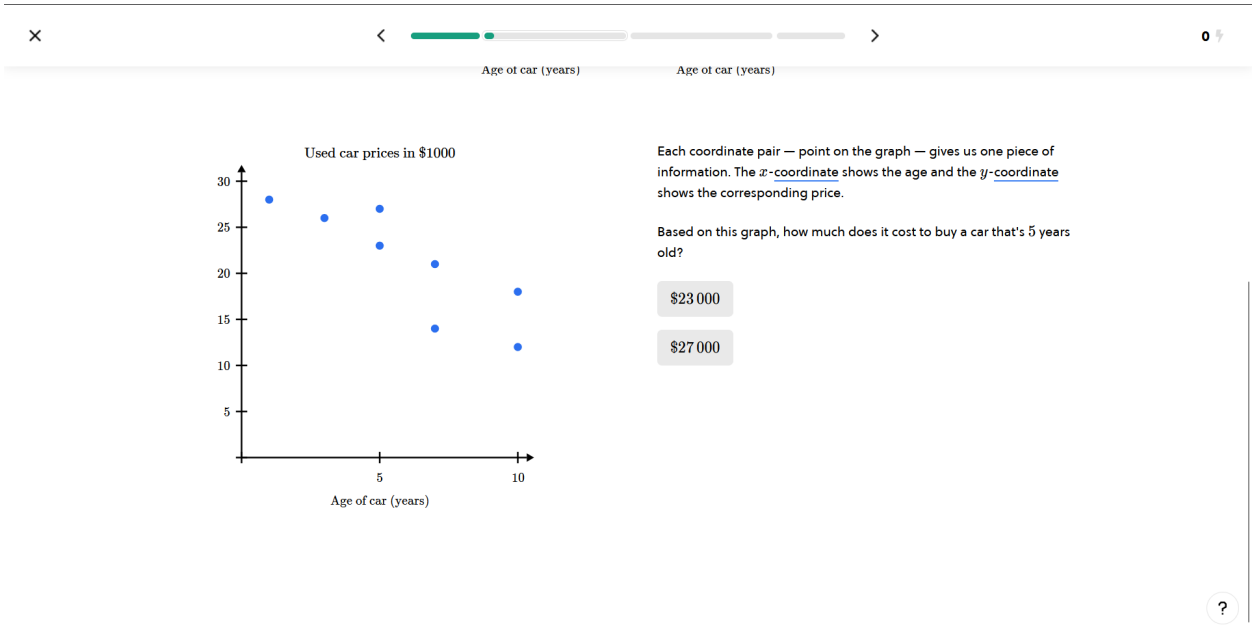

Figura 2.4. Captura de pantalla de una pregunta

El proceso de aprendizaje de los contenidos es pausado y paso a paso. Cada vez que presenta algo nuevo propone una pregunta sencilla o un ejercicio con una simulación para asegurarse de que el lector participa activamente en el proceso (Figura 2.4). Esto forma parte de la aspiración de sus creadores de enseñar usando ejemplos prácticos en los que el usuario puede participar de forma activa y no solo como lector.

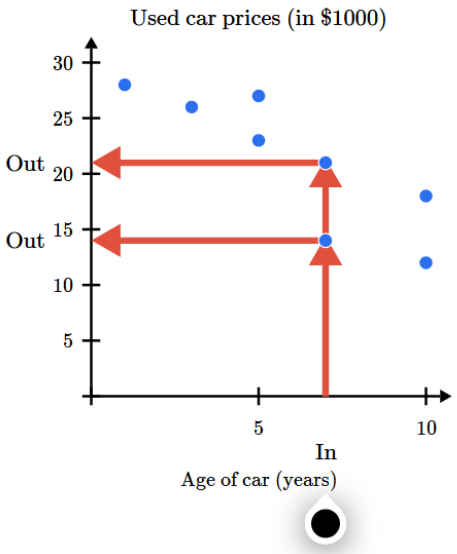

Figura 2.5. Gráfica con deslizador interactivo

En algunos apartados además de preguntas se presentan explicaciones interactivas o simulaciones. En esta gráfica (Figura 2.5) se presenta un deslizador de color negro que muestra las flechas y los valores de cada punto de la gráfica según se mueve en el eje horizontal. Enseña de forma dinámica qué significa cada parte de la gráfica. En los apartados de texto intenta tener bloques de texto cortos ya que no usa colores diferentes o símbolos gráficos para las explicaciones.

2.1.3 Lumosity

Lumosity fue lanzada en 2007 con juegos cognitivos y problemas de matemáticas como contenido principal. Actualmente es una aplicación centrada en juegos cognitivos que entrenan la velocidad de reacción, la memoria, atención, flexibilidad de razonamiento y resolución de problemas. La versión gratuita de la aplicación permite usar una serie aleatoria de ejercicios interactivos de estas habilidades diariamente.

Para este proyecto, es de interés analizar el apartado interactivo de estos ejercicios, a pesar de que su supuesta efectividad ha sido cuestionada ampliamente (Noë, 2017).

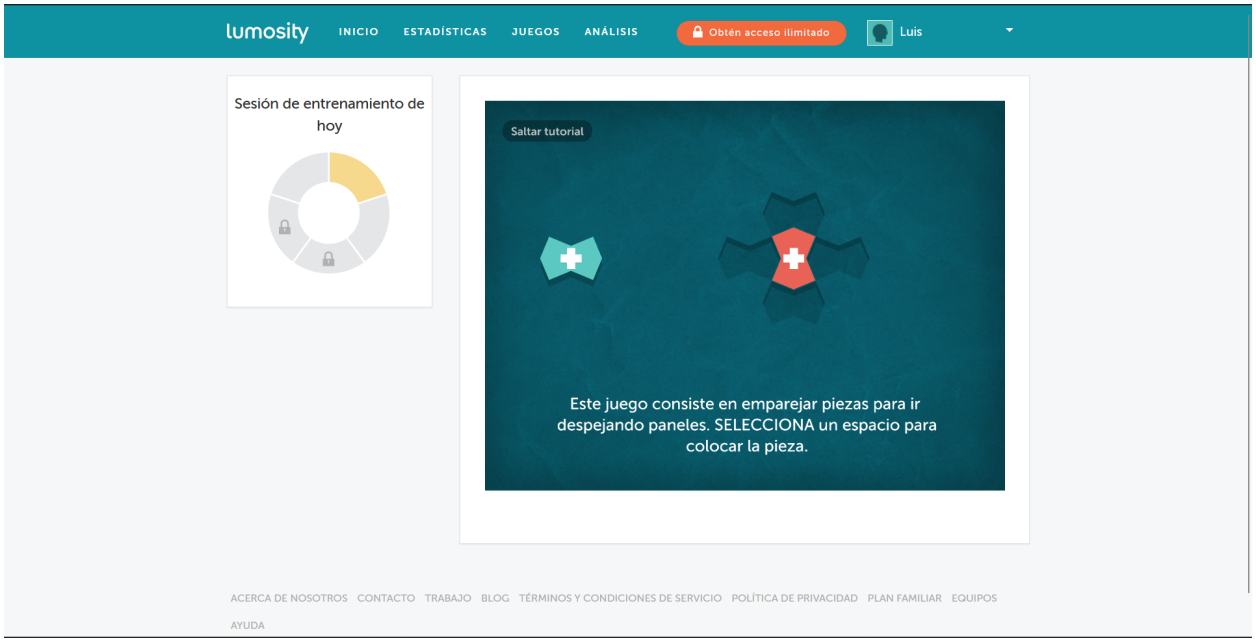

Figura 2.6. Captura de un ejercicio diario de flexibilidad

Cada día puedes realizar 3 ejercicios (Figura 2.6) que después se clasifican según tu estado de ánimo y horas dormidas, introducidos voluntariamente al iniciar los ejercicios. Estos problemas son completamente interactivos y ajustan su dificultad al rendimiento del usuario. Cuanto mejor lo haces más complicado se vuelve el ejercicio de forma progresiva, de la misma forma que fallar reduce la dificultad al instante.

Figura 2.7. Captura del apartado de estadísticas.

Lumosity tiene potencial de gamificación pero no la contiene en sí mismo. Es cierto que los ejercicios pueden considerarse juegos pero fuera de ellos no hay un sistema de gamificación en particular del aprendizaje que incentive el volver cada día a usarlo de nuevo. Únicamente se tiene un sistema de rachas y analíticas de rendimiento con valores aparentemente arbitrarios (Figura 2.7).

	Interactiva	Explicativa	Público	Dirigido a móviles	Gamificación	Acceso
Khan Academy	No	Sí	Joven y adulto	No	Medallas	Gratuito y sin ánimo de lucro
Brilliant	Sí	Sí	Joven	Sí	No	Prueba gratuita
Lumosity	Sí	No	Adulto	No	Sí	Prueba gratuita
Kahoot	No	No	Joven	Sí	Sí	Gratuito con apps de pago
Google Classroom	No	No	Todos	No	No	Gratuito
Minecraft Education	Sí	Sí	Joven	Sí	Sí	De pago
LuDOS	Sí	Sí	Joven	Sí	Sí	Bajo consideración

Tabla 1.1. Comparativa de aplicaciones educativas

En el presente análisis comparativo se ha podido constatar que la gran mayoría de las aplicaciones objeto de estudio tienen un público objetivo similar y cuentan con una explicación previa en los casos que resultan de mayor interés (Tabla 1.1). No obstante, se ha observado que existe una discrepancia en lo que respecta a la explicación y la gamificación, puesto que solo dos aplicaciones, Minecraft Education y LuDOS, coinciden en ambos aspectos. Sin embargo, se ha de tener en cuenta que Minecraft Education es una aplicación de pago que se centra principalmente en la enseñanza de contenidos relacionados con la informática en un entorno 3D, lo que la diferencia de las demás opciones. En consecuencia, se concluye que LuDOS sigue siendo una alternativa destacada respecto al resto de aplicaciones analizadas.

2.2 Análisis de las herramientas

En la presente sección nos centramos en analizar las herramientas y lenguajes idóneos para el desarrollo del trabajo. Para ello hemos definido una serie de requisitos mínimos y

hemos analizado un conjunto de herramientas que se han considerado candidatas para este trabajo. Los requisitos mínimos establecidos para seleccionar una herramienta de desarrollo para esta aplicación móvil deben ser los siguientes:

- Desarrollo multiplataforma (Windows, Android e IOS).
- Motor de gráficos 2D.
- Interactividad mediante interfaz táctil.
- Tener un programa final lo más ligero posible, ya que en móviles el espacio de instalación es más reducido.

Siguiendo estos requisitos se han considerado tres herramientas para usar: Android Studio, Unity y Godot.

	Multiplataforma	2D	Interactividad	Tamaño aproximado
Android Studio	No	Sí	Limitada	Pesado
Unity	Sí	Sí	Sí	Moderado
Godot	Sí	Sí	Sí	Ligero

Tabla 1.2. Comparativa de las aplicaciones de desarrollo

En la Tabla 1.2 se muestra un resumen de las características analizadas para cada herramienta en la que Godot y Unity cumplen los requisitos pero Godot tiene las ventajas de ser más ligero y de código abierto, por lo que será la elección para el desarrollo de la aplicación.

2.2.1 Android Studio

Android Studio (Hohensee, B. 2014) es una herramienta de desarrollo que se enfoca en la creación de aplicaciones móviles, y que además utiliza el patrón Material Design (*Material Design*, Google), lo que le confiere una serie de características y particularidades que la hacen muy útil para la creación de aplicaciones móviles de alta calidad. Sin embargo, debido a que los componentes prefabricados de la herramienta están creados específicamente para aplicaciones móviles, puede complicarse el proceso de programación de un videojuego muy

interactivo. Es decir, aunque es posible utilizar Android Studio para la creación de videojuegos, conlleva un mayor esfuerzo y trabajo que si se utilizara un motor de videojuegos específico y dedicado a tal fin.

2.2.2 Unity

Unity (Erosa García, 2019) se centra en el desarrollo de videojuegos 3D para Windows y consolas pero también para Android o iOS. A pesar de no haber sido diseñado con la intención de usar gráficos 2D, pueden crearse aplicaciones 2D con Unity, aunque con algunos inconvenientes en la producción y la necesidad de realizar algunos ajustes sobre la marcha por las limitaciones que impone la herramienta principalmente en el desarrollo de la interfaz de usuario. El principal inconveniente de Unity es que su flujo de trabajo es complejo y, si no requieres de herramientas específicas de Unity, cualquier otra opción que lo facilite será preferible.

Además Unity requiere una licencia de pago a partir de ciertos umbral de beneficios obtenidos por el proyecto, aunque para este proyecto, dada su escala, no será necesario. Sin embargo, estos costes podrían resultar un impedimento si más adelante se pretende monetizar de alguna forma el resultado del proyecto.

2.2.3 Godot

Godot (*Godot Engine Features*, Linietsky, J. y Manzur A.) es una herramienta de desarrollo que se encuentra diseñada específicamente para la creación de juegos y aplicaciones en 2D, aunque también cuenta con la posibilidad de crear proyectos en 3D. Una de las principales ventajas de Godot es que su enfoque de producción es más atomizado, lo que significa que el bloque básico de diseño es el nodo, y cada nodo actúa dentro de una jerarquía de nodos. Esto facilita enormemente el diseño de interactividad y la reutilización de código, lo que se traduce en un proceso de desarrollo más eficiente y rápido.

Además, otra gran ventaja de Godot es que se trata de una herramienta de código abierto y de uso libre, lo que significa que los desarrolladores no tienen que preocuparse por

licencias o costos asociados a la herramienta. Esto permite que se puedan enfocar en el desarrollo del proyecto en sí, sin tener que preocuparse por costos adicionales.

En cuanto a la elección de Godot para la creación de un proyecto específico, destaca la ligereza del programa final y la sencillez de su flujo de trabajo (Henrique & Silva, 2023). Esto significa que, no solo se obtiene un resultado final más liviano, sino también que se puede trabajar de manera más eficiente y sin complicaciones innecesarias.

Es por esta razón y por las ventajas destacadas que para este proyecto se ha decidido utilizar la herramienta Godot haciendo uso de patrones de desarrollo apropiados para el motor, como se verá más adelante en los capítulos de diseño e implementación.

3 Metodología

En este capítulo se describen las metodologías de desarrollo, trabajo y educativa seguidas para el desarrollo del proyecto.

3.1 Metodología de desarrollo

El proceso de desarrollo de esta aplicación móvil ha constado de varias fases:

1. Idea: en esta fase se define la idea general de la aplicación, se analiza su viabilidad y se estudia el mercado al que estará destinada.
2. Análisis de requisitos: en esta fase se establecen los requisitos que debe cumplir la aplicación móvil y se definen las funciones que debe tener.
3. Diseño: en esta fase se desarrolla el diseño de la aplicación, tanto a nivel visual como funcional.
4. Desarrollo: en esta fase se procede a la programación y desarrollo de la aplicación.
5. Pruebas: en esta fase se comprueba el correcto funcionamiento de la aplicación y se llevan a cabo diferentes pruebas para detectar posibles errores.

Para poder acotar el desarrollo dentro de un plazo de 6 meses establecemos algunas limitaciones de plazos para cada fase: 1 mes para las fases 1 y 2, 1 mes para el diseño, 3 para el desarrollo y 1 para las pruebas necesarias. El tiempo de pruebas y desarrollo se solapan al final y se alterna a medida que se recibe información de las pruebas para mejorar la aplicación.

3.2 Metodología de trabajo

Para el desarrollo de la aplicación hemos usado el método Agile como referencia (Cunningham, 2001), aplicando procesos necesarios para el desarrollo de una aplicación web.

Se han usado los fundamentos de los sprints de la metodología SCRUM como guía para establecer plazos y el bucle general de trabajo.

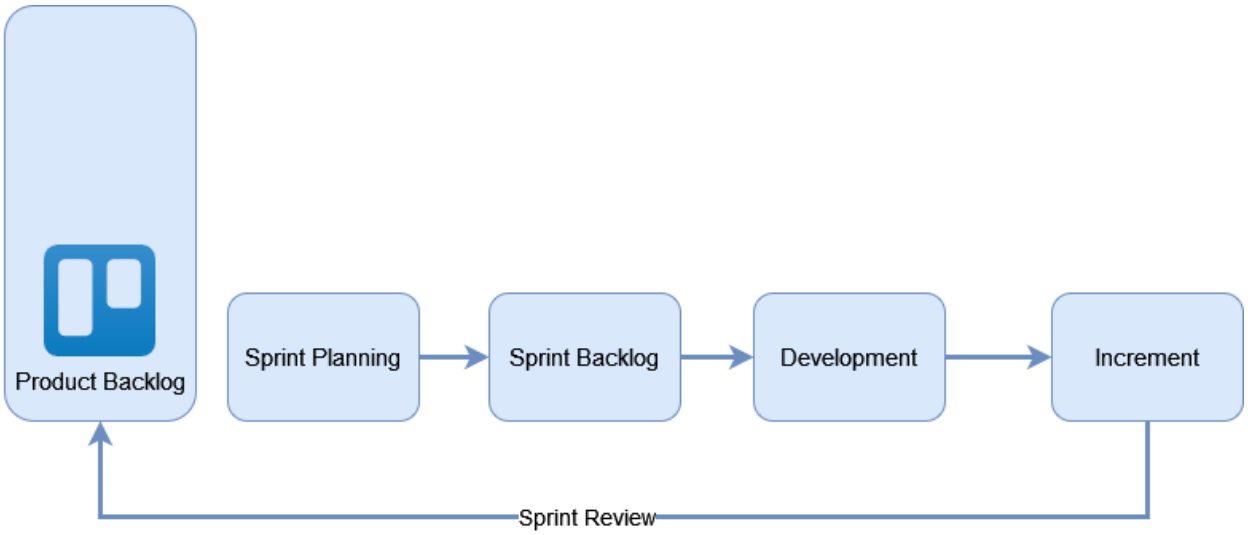

Figura 3.1. Proceso de trabajo

El proceso consiste en *sprints* de dos semanas, con un *sprint planning* al inicio en el que se detalla el *sprint backlog* para guiar el desarrollo el resto del periodo (Figura 3.1). Una vez terminado, el objetivo es conseguir un incremento en la aplicación. Este incremento debe ser testable y funcional.

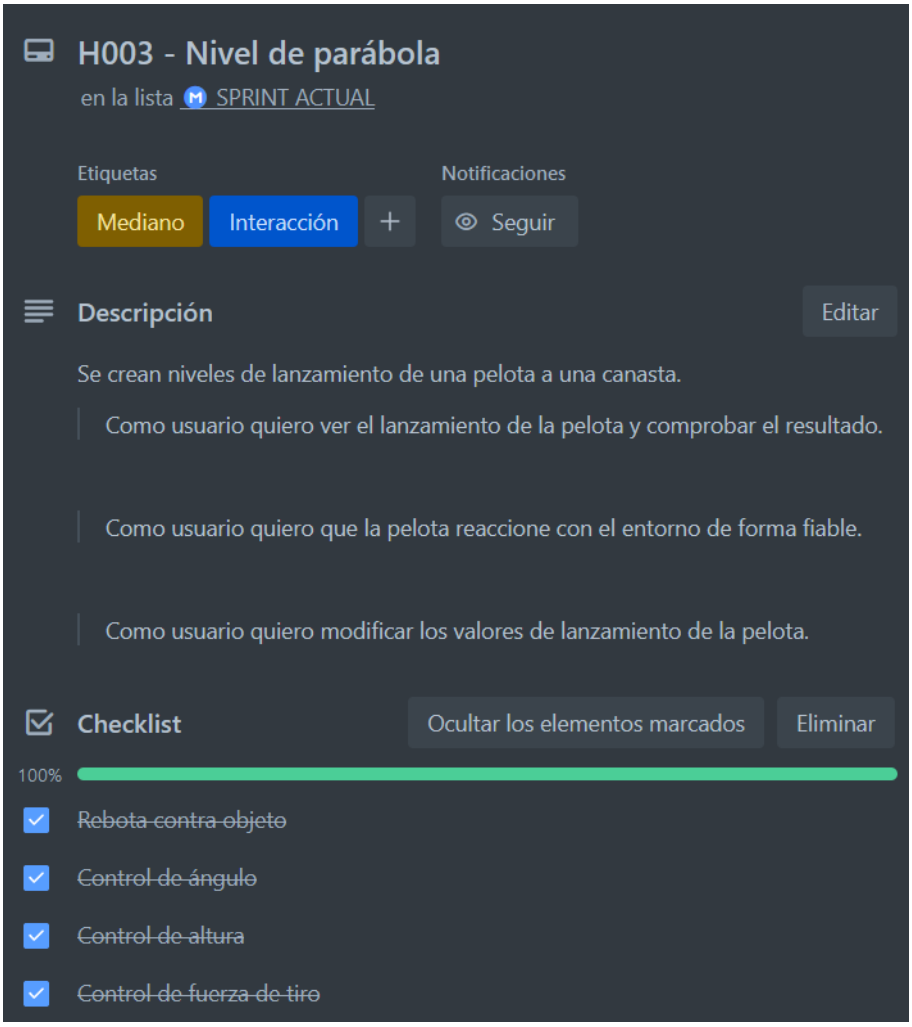

Figura 3.2. Ejemplo de historia en Trello

Cada historia se describe con un tamaño según su dificultad y tiempo de trabajo requerido y un ámbito entre: diseño, interacción, documentación o arte (Figura 3.2). Además se incluyen algunas historias de usuario para definir necesidades del diseño y la implementación. Estos son, a grandes rasgos, los componentes del proyecto.

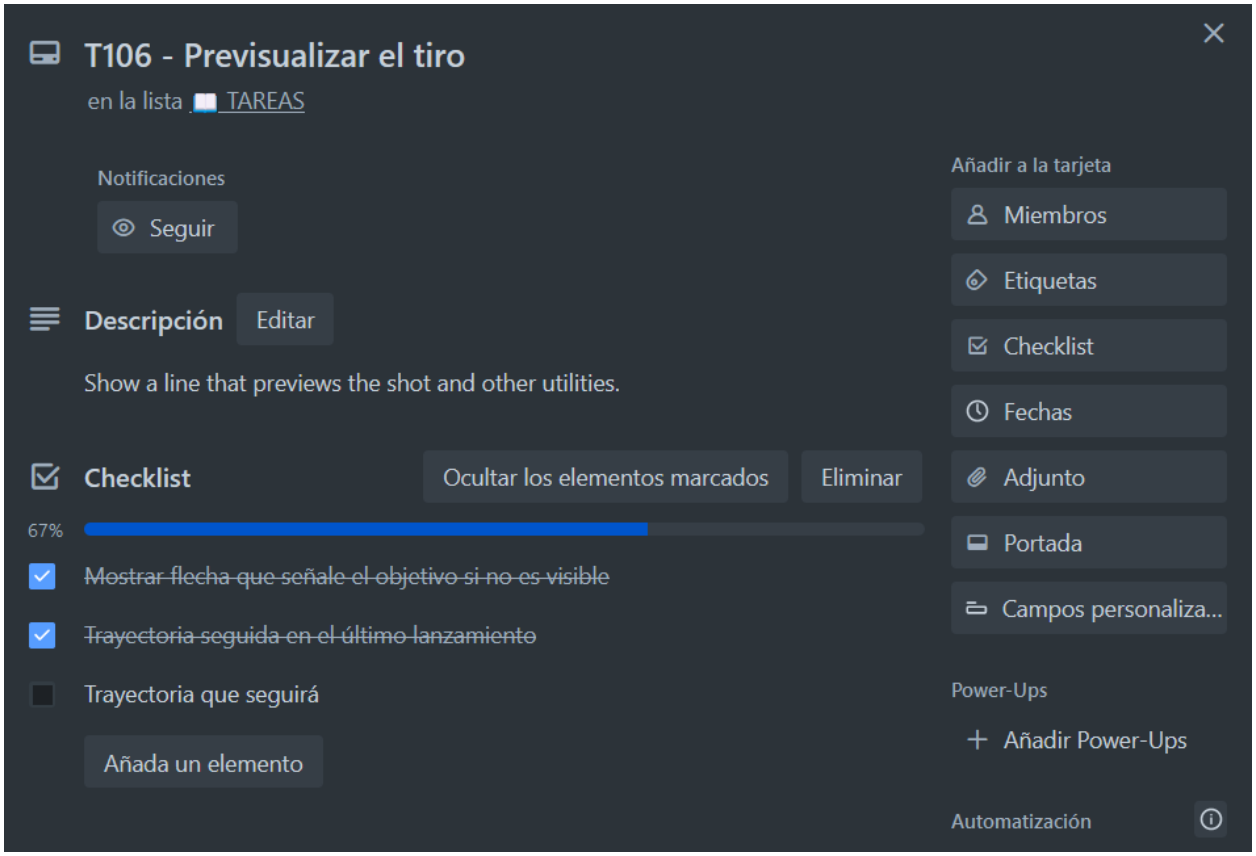

Figura 3.3. Ejemplo de tarea

Cada historia se divide en tareas que se dividen a su vez en una lista de objetivos a cumplir para poder completar la tarea (Figura 3.3). Una vez se cumplen los objetivos se realizan comprobaciones y las pruebas de validación que sean necesarias.

3.3 Metodología educativa

La metodología seguida para diseñar los contenidos y la presentación de los mismos es la de la gamificación. La gamificación se define como la realización de un trabajo seguido de una recompensa o *feedback* directamente relacionado. En resumen, aprender jugando (Werbach, 2020). De esta relación surge la motivación extrínseca de la que depende el éxito de la gamificación. En lugar de depender exclusivamente de la motivación intrínseca del alumno por aprender los conceptos se le premia de forma extrínseca como apoyo. Un ejemplo común

de gamificación es el de las medallas por cooperar, excelencia o realizar correctamente ciertas tareas (Contreras Espinosa & Eguia, 2017, 19).

Comúnmente se plantea la motivación extrínseca como única forma de gamificación viable ya que es la única que puede medirse y mostrarse en un programa, sin embargo sin motivación intrínseca la gamificación puede no tener el éxito deseado. Para conseguir crear una motivación intrínseca en el alumnado la metodología debe considerar factores culturales y la disposición personal de cada alumno hacia el aprendizaje de la materia (Rusk & Larson, 2011, 89). Para lograrlo, el diseño de la gamificación debe permitir cierto nivel de competitividad y cooperatividad, además de no restringir la creatividad del alumnado al intentar resolver los problemas presentados.

La creatividad es un gran generador de motivación intrínseca pero requiere de cierto nivel de empoderamiento y libertad dentro del sistema (Salge et al., 2014), de cierta capacidad de cambio en el ambiente. En el caso de esta aplicación se pretende conseguir no penalizando los intentos, por muy numerosos que sean, y ofreciendo la ayuda justa según el alumno lo necesite haciendo uso de consejos o sugerencias en el campo de juego para evitar que pierda la motivación completamente.

Una vez la motivación queda resuelta queda definir qué métodos se usarán para facilitar la comprensión y aprendizaje de los contenidos. En cuanto a estos se usarán principalmente dos métodos: diálogos y animaciones.

Los diálogos dirigidos a uno mismo y fragmentados resultan más fáciles de leer que un párrafo explicativo, es el método en el que se basa el género de videojuegos de novelas visuales, que también ayuda si quien lee tiene dificultades de algún tipo ya que no se ve tan fácilmente abrumado por una gran cantidad de texto permanente. Las animaciones sirven para ejemplificar de forma dinámica las explicaciones. Una imagen estática puede describir el movimiento que sigue una parábola pero una animación lo muestra, haciendo hincapié en la accesibilidad de la aplicación y la atención a la diversidad.

Los conceptos de física y matemáticas aplicables a este prototipo son los siguientes:

- Vector Posición y Vector Desplazamiento.
- Trayectoria y Espacio Recorrido.
- Velocidad media y velocidad instantánea.
- Aceleración media y Aceleración instantánea.
- Movimiento Rectilíneo Uniforme MRU.
- Gráficas MRU.
- Composición de Movimientos MRU.
- MRUA.
- Caída Libre.
- Gráficas MRUA.
- Tiro Parabólico.
- Tiro Horizontal.
- MRU y MRUA.

Sin embargo a fin de probar el concepto de aplicación solo se implementa el apartado de tiro parabólico ya que utiliza la mayoría de conceptos de la lista en un solo tipo de problema.

3.3.1 Guía docente

Como ayuda adicional al profesorado se incluirá una guía docente para servir de apoyo, donde se incluyen explicaciones de los problemas, las soluciones y el flujo de juego detallado para que el profesorado pueda usar la aplicación de forma efectiva sin necesidad de realizar pruebas preliminares o preparar material adicional.

4 Análisis de requisitos y diseño

En el presente capítulo vamos a realizar el análisis de requisitos, el diseño de casos de uso y el diseño de la aplicación y de la interfaz gráfica.

4.1 Requisitos

En este apartado se especifican los requisitos de la aplicación.

- Simulador de tiro parabólico: el simulador debe permitir al usuario ingresar los valores de velocidad inicial, gravedad y ángulo de lanzamiento, y luego calcular y mostrar la trayectoria del proyectil en tiempo real.
- Diálogos con explicación del problema: presentación de los problemas y la metodología para resolverlos de forma dialogada con uno o varios personajes.
- Fórmulas a usar en el problema: un libro de fórmulas accesible en cualquier momento que muestre los valores de forma dinámica.
- Menú de niveles: organización visual de los niveles disponibles para jugar.
- Puntuaciones: recompensa visual que sirve como registro para el progreso del usuario.
- Pantalla de final de nivel: resumen del progreso y resultado de un problema terminado.
- Pantalla de logros conseguidos: almacén de recompensas donde mostrar el progreso dentro de la aplicación.

4.2 Casos de uso

Aquí se desarrollan más en profundidad cada caso de uso derivado de los requisitos. Cada caso de uso cuenta con una precondición que debe cumplirse, un flujo básico de acciones por parte del usuario y una postcondición o resultado.

Caso de uso	Consultar logros
Descripción	El usuario lee los logros que ha conseguido
Precondición	Ha accedido a la aplicación
Flujo básico	1. En el menú pulsa el botón logros 2. Se despliega para mostrar los logros conseguidos y los no conseguidos
Postcondición	Ha visto sus logros

Tabla 4.1. Caso de uso de los logros

Una de las partes fundamentales de la gamificación del proceso de aprendizaje es establecer metas alternativas en forma de logros (Tabla 4.1). Estas funcionan a la vez como guía y como recompensa. Los logros, o medallas, son una forma eficaz de dirigir el comportamiento del usuario cuando no sabe qué debe hacer o no tiene la proactividad de buscar objetivos por sí mismo (Anderson et al., 2013).

Caso de uso	Reintentar un nivel ya completado
Descripción	El usuario ve la puntuación obtenida en cada nivel y puede reintentar un nivel
Precondición	Ha accedido a la aplicación y ha terminado al menos un nivel
Flujo básico	1. Pulsa al botón de ruta de niveles desde el menú principal 2. Encima de cada nivel completado puede ver las estrellas conseguidas 3. Cuando pulsa en un nivel se le permite reintentarlo.
Postcondición	Reintenta el nivel

Tabla 4.2. Caso de uso de reintentar un nivel

Al final de cada nivel completado es importante que el usuario conozca su resultado y puntuación como respuesta a la actividad realizada (Tabla 4.2). En esta se muestra lo cercano

que ha estado de un resultado óptimo y se le da la oportunidad de reintentar un nivel específico.

Caso de uso	Diálogos con la explicación del problema
Descripción	El usuario recibe una explicación previa al nivel
Precondición	Ha accedido a la ruta de niveles
Flujo básico	1. Pulsa en un nivel 2. Se abre la escena de diálogo 3. Se muestra un diálogo explicativo del problema. 4. Para avanzar en las frases del diálogo el usuario toca la pantalla.
Postcondición	Puede leer y seguir la explicación

Tabla 4.3. Caso de uso de los diálogos

Los diálogos son la explicación de los problemas y una forma de ofrecer contexto al alumnado. En este momento se muestran consejos visuales y de texto (Tabla 4.3) para ayudar en el siguiente problema sin dar la solución.

Caso de uso	Resolver un problema de tiro parabólico
Descripción	El usuario completa un tiro parabólico en el simulador de tiro a canasta
Precondición	Ha accedido a la aplicación y ha entrado al nivel
Flujo básico	1. Observa el estado del problema 2. (Opcional) Lee las fórmulas para resolver el problema 3. Manipula los valores de ángulo, gravedad y/o velocidad 4. Lanza la pelota 5. La pelota entra en la canasta 6. Se muestra la pantalla de final de nivel
Postcondición	Ha completado el nivel y accede al siguiente

Tabla 4.4. Caso de uso del tiro parabólico

El tiro parabólico consta de dos fases principales: establecer valores y observar el resultado. Para establecer los valores el usuario cuenta con deslizadores que manipulan el valor de las incógnitas y en cualquier momento puede pulsar el botón para lanzar y observar el resultado (Tabla 4.4).

4.3 Diseño

En esta sección se describen los patrones de diseño de código y diseño gráfico usados en el desarrollo de la aplicación.

4.3.1 Diseño de código

Para diseñar la aplicación hemos seguido los patrones de diseño recomendados por Godot (Linietsky et al., 2014). La razón es el uso de nodos, un tipo de estructura de proyecto propio de Godot.

En Godot, los nodos son los componentes básicos utilizados para construir una escena. Cada nodo tiene una función específica y se puede asignar como hijo de otro nodo para construir una jerarquía de nodos en forma de árbol. Los nodos pueden ser de diferentes tipos, como nodos de control de cámara, nodos de objetos 2D o 3D, efectos especiales, luces, etc.

Además, las escenas en Godot están compuestas por una o más jerarquías de nodos conectados. Las escenas también pueden contener otras escenas como nodos, lo que permite crear una estructura más compleja de nodos y escenas. Los nodos se conectan entre sí con señales, lo que permite la comunicación y el intercambio de datos entre ellos.

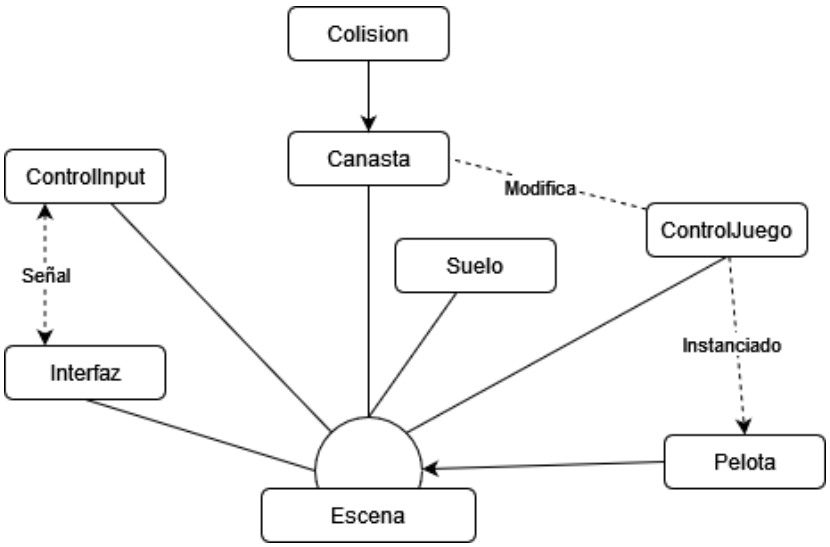

Figura 4.1. Diseño de la escena del juego de tiro parabólico

En la figura 4.1 se muestran los nodos que forman la escena del nivel de tiro parabólico y las principales interacciones entre ellos. Las líneas simbolizan la jerarquía de nodos mientras que las líneas punteadas son o bien señales o bien instancias creadas por otro nodo.

Por ejemplo los nodos *Interfaz* y *ControlInput* de la figura 4.1 tienen varias señales en ambas direcciones que usan para comunicar los cambios en la interfaz y los botones que pulsa el usuario.

Para programar en Godot se usa el lenguaje de alto nivel y tipado dinámico GDScript. Al ser el lenguaje propio de Godot cuenta con un amplio soporte en la aplicación. Este tiene una sintaxis similar a Python.

```
1  func _on_Player_body_entered(body):
2      hide() # El usuario se esconde al recibir el impacto.
3      emit_signal("hit")
4
```

Figura 4.2. Ejemplo de código.

En este ejemplo de código GDScript (Figura 4.2) hay una función llamada *_on_Player_body_entered* que llama a otra función y emite una señal “hit”. Esta es la forma más básica de comunicación entre nodos de Godot.

En el diseño de niveles y el flujo de juego se ha seguido el siguiente diseño.

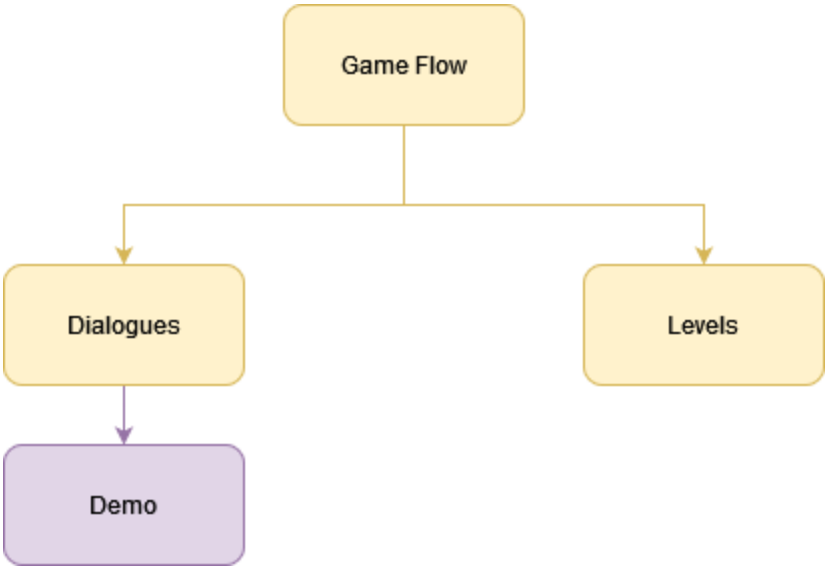

Figura 4.3. Árbol de archivos del flujo de juego

En la figura 4.3 se muestran los 3 archivos en formato JSON mostrados en color amarillo y los archivos de escenas en morado. El archivo *Dialogues* contiene los diálogos que se muestran en el juego con identificadores para poder referenciarlos. *Levels* contiene los niveles organizados por tipo y los datos de cada nivel: incógnitas, datos iniciales y resultado. Finalmente *Game Flow* describe qué diálogo y qué nivel se muestra en cada momento en un orden lineal.

Demo es un conjunto de escenas que se muestran junto a los diálogos a modo de demostración gráfica para apoyar las explicaciones.

4.3.2 Diseño de interfaz de usuario

En esta sección se detalla el proceso de diseño de la interfaz y los gráficos de la aplicación. Empezamos el diseño con una fase de *greyboxing*. El *greyboxing* se refiere al proceso de crear maquetas básicas interactivas de pantallas o interfaces de usuario en el desarrollo de aplicaciones móviles. Estas maquetas son esquemas en blanco y negro que representan las características y la funcionalidad de la aplicación, y se utilizan para probar la interacción del usuario y las funcionalidades sin invertir tiempo y dinero en detalles visuales

complejos. Esto nos permite enfocarnos en la funcionalidad y experiencia del usuario en lugar de la estética de la aplicación durante las primeras etapas de desarrollo.

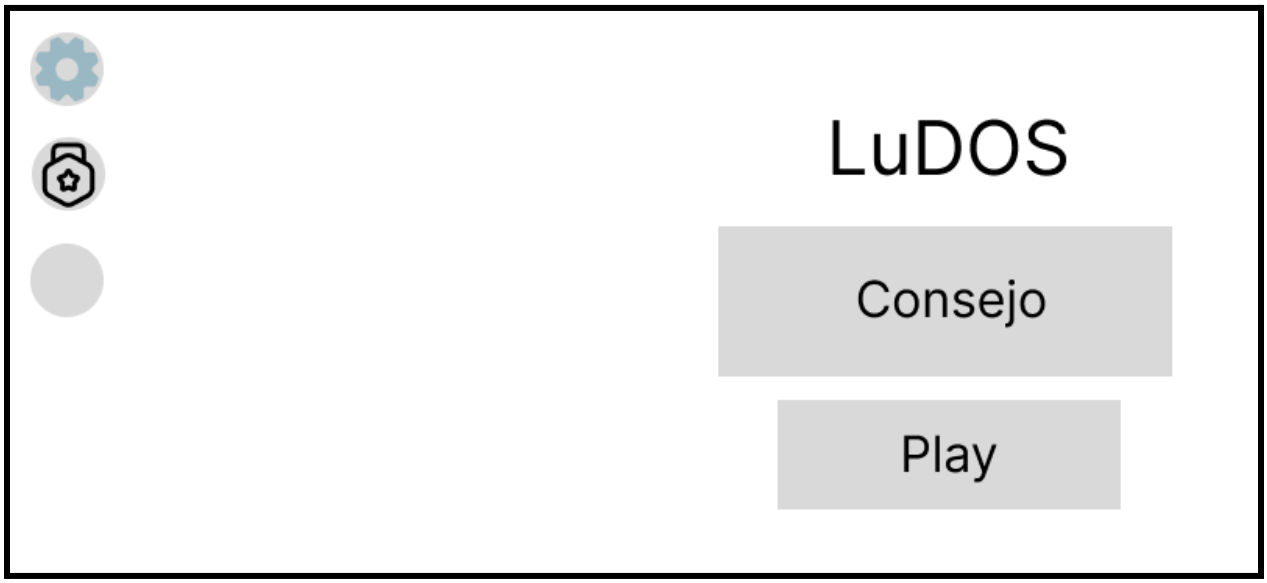

Figura 4.4. *Greyboxing* del menú principal.

En la figura 4.4 se muestran los elementos de la pantalla principal que vería el usuario. Se muestra un botón para comenzar el juego, un panel que muestra un consejo aleatorio, el título de la aplicación y tres botones a la izquierda: ajustes, logros y perfil.

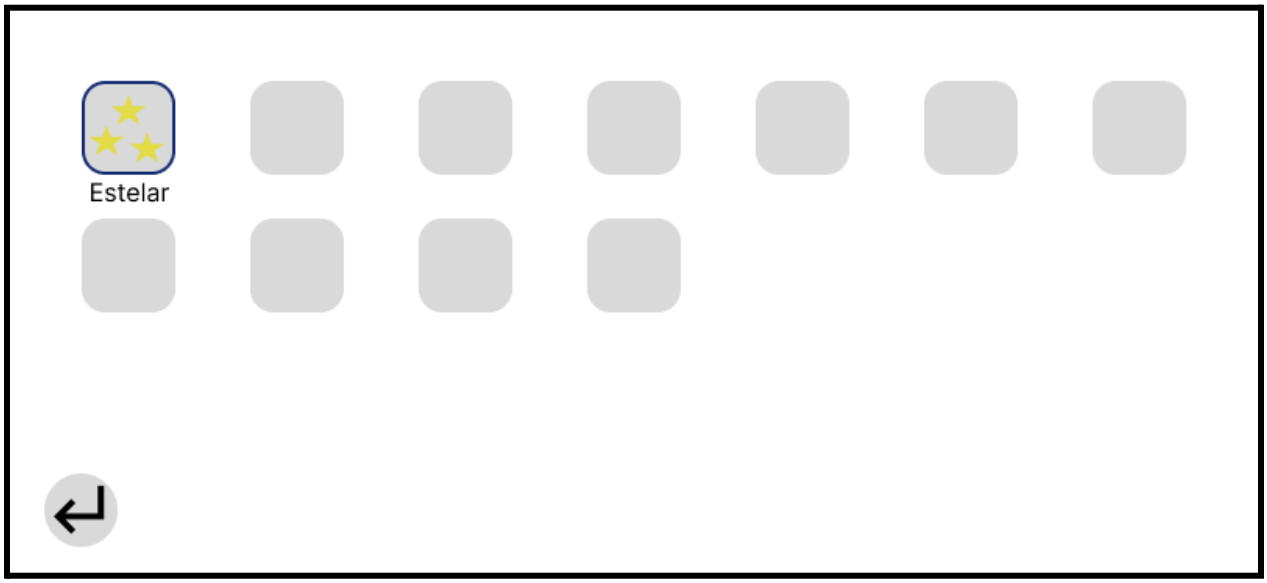

Figura 4.5. *Greyboxing* de la pantalla de logros.

En la figura 4.5 se muestran los logros que el usuario puede obtener y los obtenidos. Los que puede obtener aparecen ocultos y los obtenidos tienen un dibujo y un texto. Abajo a la izquierda hay un botón flotante para volver a la pantalla del menú principal.

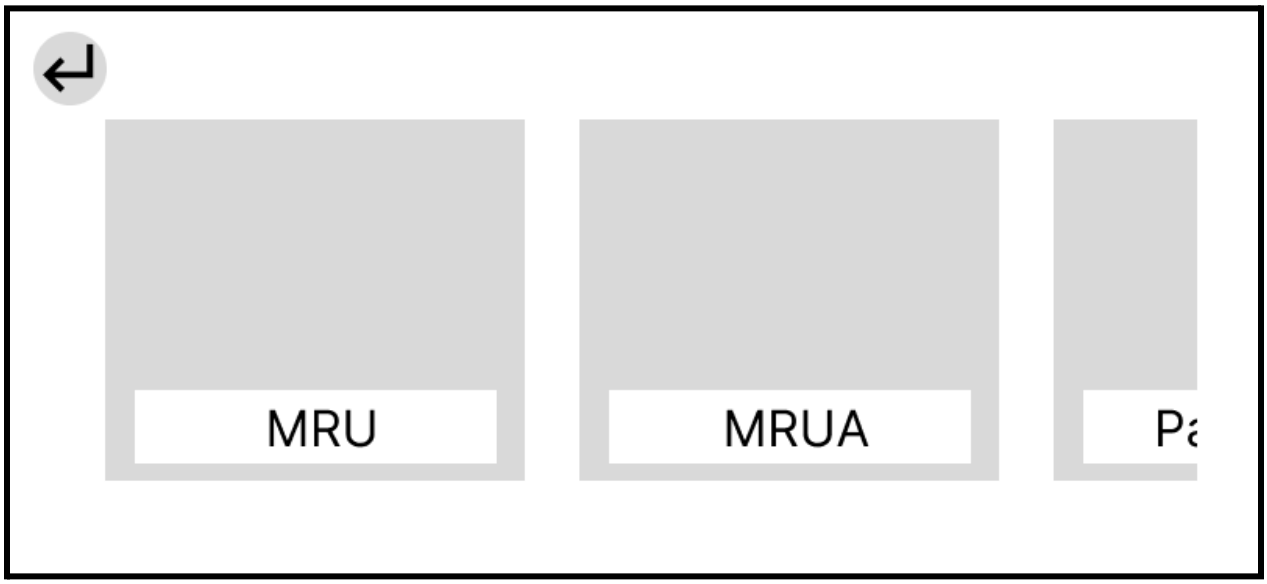

Figura 4.6. *Greyboxing* de la selección de tema.

Para seleccionar un tema tras pulsar “Play” se muestra una lista con los temas disponibles ordenados (Figura 4.5). Cuando se pulse uno cualquiera se abrirá el panel de niveles en el que puede elegir si saltar a un nivel ya jugado o empezar por uno anterior (Figura 4.6).

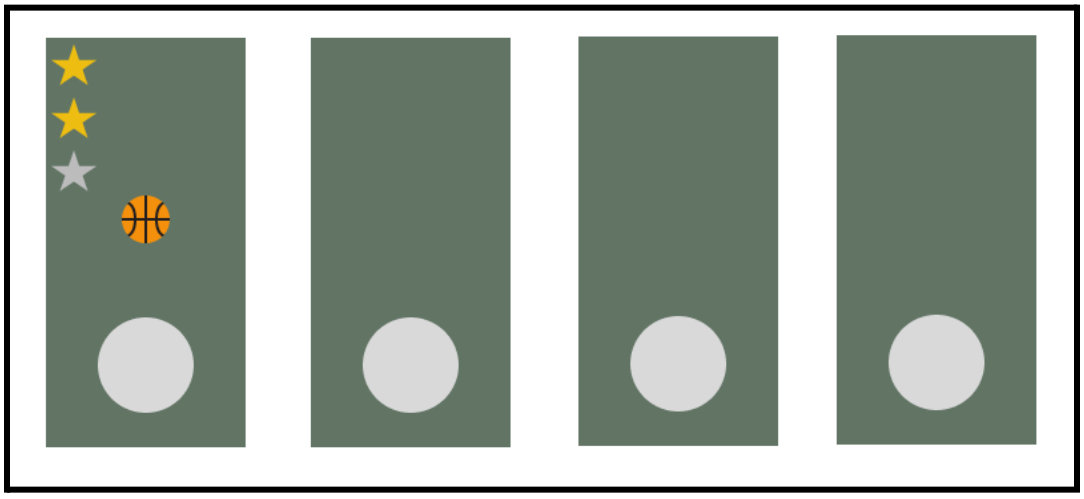

Figura 4.7. *Greyboxing* de selección de nivel.

En la selección de nivel (Figura 4.7) se muestran los niveles disponibles como superados u ocultos (no superados). El usuario puede seleccionar uno y desplazarse en el eje horizontal para verlos todos.

Figura 4.8. *Greyboxing* de los diálogos.

Al iniciar un nivel empieza una explicación sobre el nivel con animaciones como apoyo gráfico (Figura 4.8). El objetivo de esta explicación en formato de diálogo es que el usuario tenga un contexto previo y reciba información básica para afrontar el problema siguiente. Hacerlo de forma dialogada es parte de la metodología de gamificación descrita en la sección 3.3 ya que los diálogos resultan más cómodos de comprender que una explicación impersonal.

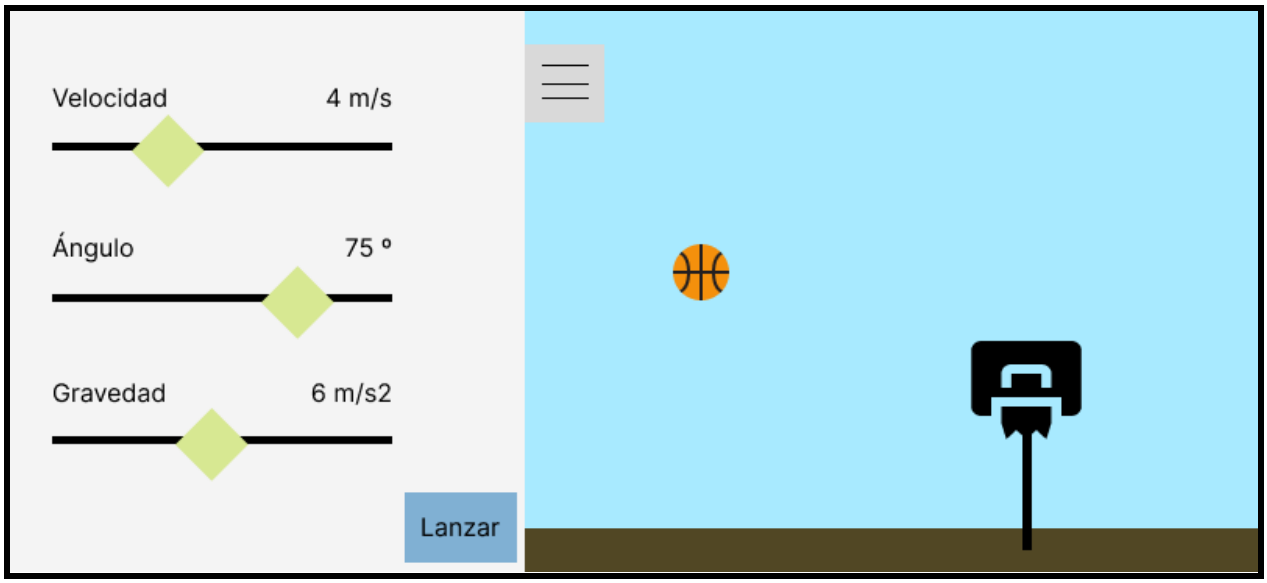

Figura 4.9. *Greyboxing* de la pantalla de nivel de tiro parabólico.

En este escenario el usuario puede modificar los valores de los deslizadores que alteran los valores de las incógnitas en una fórmula de tiro parabólico. Al tener controles en esta pantalla (Figura 4.9) para modificar los datos de entrada al problema tenemos en cuenta que ocupan más de un tercio de la pantalla. Por esta razón decidimos ofrecer la opción de ocultar y mostrar el panel de controles parcialmente.

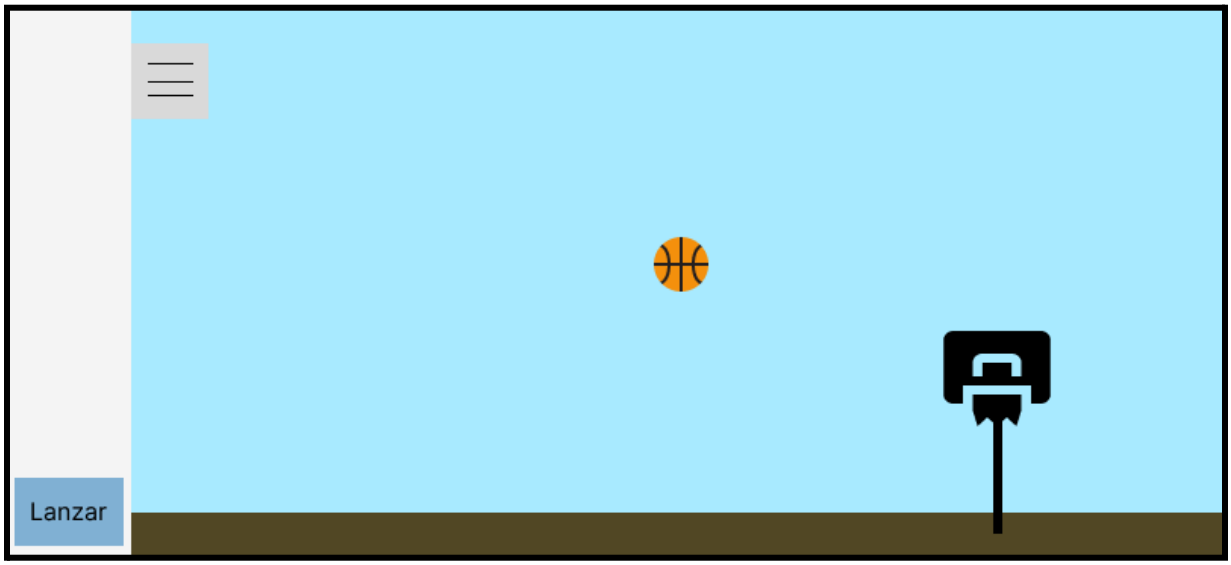

Figura 4.10. *Greyboxing* de la figura 6.6 con el panel oculto.

Ocultar el panel de control sirve como ayuda visual para tener menos objetos en pantalla y concentrarse en observar el resultado del lanzamiento como se muestra en la tabla 4.4 y la figura 4.10.

5 Implementación

En esta sección se detalla la implementación de los sistemas descritos en la sección 4.3.1.

Para la implementación de los diseños en el proyecto se han seguido las recomendaciones de diseño antes mostradas en la Figura 4.1 y descritas en la documentación oficial de Godot (Linietsky et al., 2014).

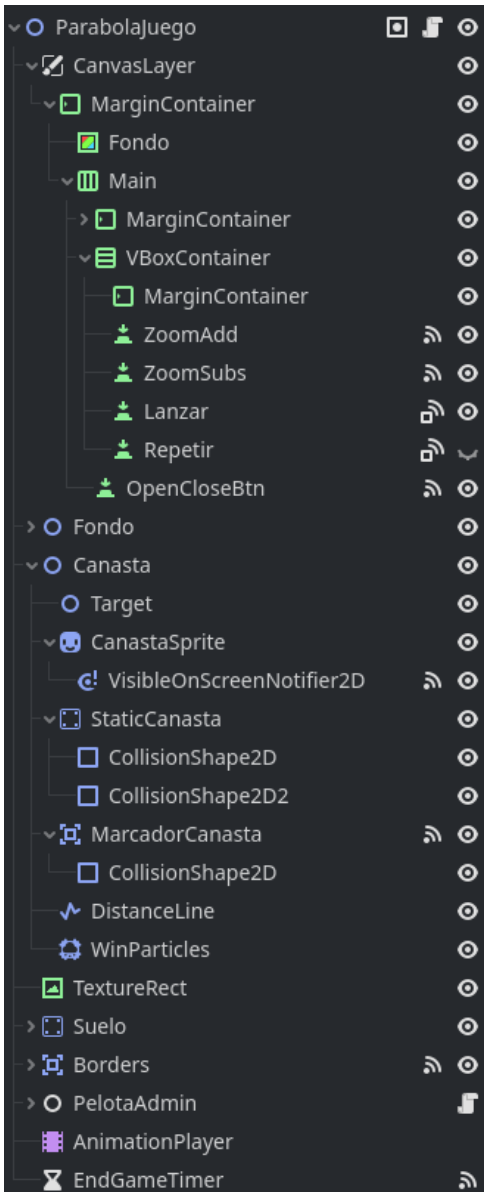

Figura 5.1. Árbol de nodos del ejercicio de tiro parabólico.

En el ejercicio de tiro parabólico que muestran las figuras 4.1, 4.8 y 4.9 se ha implementado por una parte la interfaz de usuario dentro del nodo “CanvasLayer” y el resto de elementos visuales en nodos varios debajo del anterior. En la figura 5.1 vemos la variedad de nodos que pueden implementarse usando las herramientas de Godot. Algunos muy simples como nodos de imagen “CanastaSprite” y otros con interacciones únicas como una línea que se dibuja detrás de la pelota (DistanceLine) o partículas que se ejecutan al superar el nivel (WinParticles).

En cada uno de los nodos con interacciones puede verse a su derecha el símbolo de una señal lo que indica que ese nodo emite señales (Godot community, 2014).

El nodo padre ParabolaJuego configura los valores iniciales del nivel, sin embargo con esto lo único que tenemos es un nivel creado así que requerimos de algún sistema para poder crear y modificar niveles. Para esto creamos un sistema de carga de niveles que, en base a un archivo JSON, configura el nivel con sus datos iniciales y sus incógnitas disponibles.

```
{
    "parabola-01": [
        {
            "starting_values" : {
                "speed": 3,
                "angle": 30,
                "gravity": 9.8,
                "distance": 3.9,
                "height": 2
            },
            "parameters": {
                "speed": true,
                "angle": false,
                "gravity": false
            },
            "solution": {
                "speed": 5,
                "angle": 30,
                "gravity": 9.8
            }
        }
    ],
```

Figura 5.2. Fragmento del archivo JSON de niveles.

En este archivo tenemos una serie de objetos con un identificador, en este caso “parabola-01” (Figura 5.2) seguido de una lista de posibles niveles consecutivos. En el mismo se ven los elementos “starting_values” que define los valores iniciales del problema, “parameters” que configura qué incógnitas pueden modificarse y “solution” que define una solución de entre las posibles (en este caso solo es una ya que hay una única incógnita). Este último objeto se utiliza en la implementación para guiar al jugador si está lejos de acertar la solución.

Este mismo método se ha utilizado para crear diálogos y el bucle de juego.

```
{
    "parabola-01" : {
        "speakers": [
            {
                "name": "SOC-86",
                "flipH": 0
            }
        ],
        "lines": [
            {
                "speaker": 0,
                "line": "Ahora que entiendes el MRU y MRUA vamos a por algo más difícil."
            },
            {
                "speaker": 0,
                "line": "En un lanzamiento de pelota a una canasta la trayectoría no es recta nunca
                "demo": "res://Scenes/Demos/ParabolicDemo.tscn"
            },
            {
                "speaker": 0,
                "line": "A ese movimiento lo llamamos parábola."
            },
            {
                "speaker": 0,
                "line": "En el eje X se moverá a la misma velocidad siempre, la del lanzamiento, es
            },
            {
                "speaker": 0,
                "line": "Pero en el eje Y la gravedad del planeta hace que caiga, acelera la pelota
                "demo": "res://Scenes/Demos/FormulasDemo.tscn"
            },
```

Figura 5.3. Fragmento del JSON de diálogos.

Para escribir y organizar los diálogos se describen de forma lineal con parámetros de texto, hablante y animaciones a ejecutar en la zona derecha de la pantalla (Figura 4.7) según se muestra en la figura 5.3. De esta forma podemos crear y modificar diálogos rápidamente e incluso abrimos la posibilidad de permitir a un usuario crear sus propios diálogos y problemas y cargarlos en la aplicación, lo que ofrece la oportunidad de crear problemas y diálogos personalizados para estudiantes con necesidades diferentes en el aprendizaje. A pesar de que no está implementado de forma sencilla en el corte vertical, sí puede modificarse únicamente reescribiendo un archivo de texto plano con datos en formato JSON.

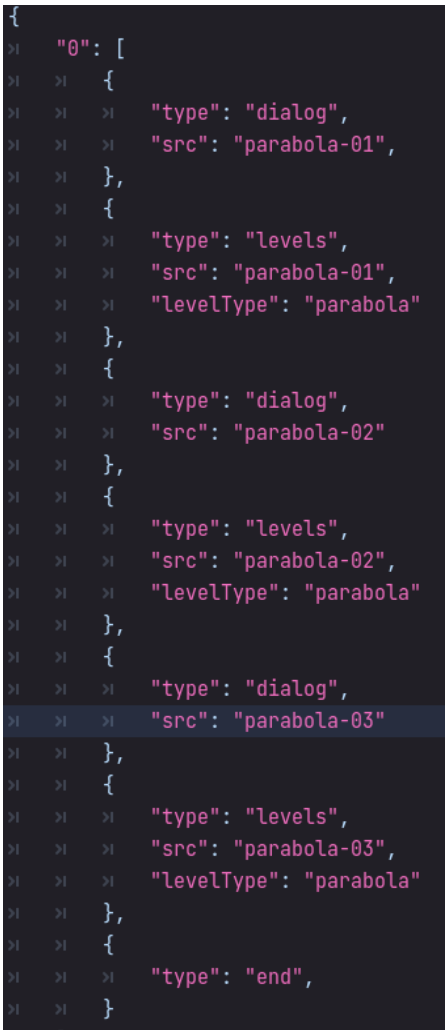

Figura 5.4. Fragmento de JSON de *game flow*.

El game flow se entiende como la secuencia de partes dentro de un juego que se adaptan a la habilidad del jugador, sus conocimientos en este caso. Para implementar la secuencia de diálogos y niveles creamos un sistema de lectura (Figura 5.4) que muestra de forma lineal qué apartado se muestra en cada momento y un nodo carga las escenas convenientes según lo escrito en el archivo.

6 Pruebas de validación

En este capítulo se detallan los procesos de validación de la aplicación y sus resultados.

6.1 Objetivos

Las pruebas de validación en el desarrollo de esta aplicación comprueban los siguientes apartados de la aplicación:

1. Funcionalidad: Verificar las funciones de la aplicación. En este caso se debe analizar que los requisitos establecidos en el capítulo 4 funcionan como se esperaba.
2. Usabilidad: Evaluar la facilidad de uso de la aplicación y su accesibilidad, con especial énfasis hacia la diversidad funcional.
3. Compatibilidad: Probar la aplicación en diferentes dispositivos móviles (específicamente Android para esta aplicación) y comprobar que es igualmente usable en diferentes resoluciones de pantalla y modelos.
4. Rendimiento: Detectar posibles errores de código que puedan producir retrasos en la carga de la aplicación o en la fluidez de sus gráficos.
5. Pruebas de contenido: Verificar la implementación de la metodología educativa desarrollada en la sección 3.3 y estudiar su grado de éxito tras utilizar la aplicación.

6.2 Testeos

En este capítulo se detallan los procesos de testeo seguidos en el desarrollo de la aplicación interactiva para comprobar funcionalidades y detectar posibles errores. Los testeos seguidos son comúnmente usados en videojuegos (Hamilton, 2023) por lo que también servirán para esta aplicación.

Testear un videojuego requiere de pruebas continuas respecto a los cambios y adiciones que se van realizando a lo largo del desarrollo por lo que estas pruebas se han repetido en las varias ocasiones en las que han sido necesarias.

6.2.1 Pruebas de funcionalidad

El método más rápido de verificar funcionalidades es usar la aplicación en dispositivos diferentes y con usuarios diferentes comprobando en tiempo real que las funciones implementadas tengan el comportamiento esperado.

6.2.2 Pruebas de compatibilidad

Para comprobar la compatibilidad de la interfaz y del sistema se instala en dispositivos con diferentes sistemas operativos y resoluciones de pantalla.

Al finalizar estas pruebas se encontró que en dispositivos iOS la instalación producía errores de licencia por lo que se decidió reducir el desarrollo a únicamente sistemas Android en dispositivos móviles.

6.2.3 Pruebas de rendimiento

Al igual que con las pruebas de compatibilidad, cuando se comprueba el rendimiento se instala en dispositivos diferentes y se anotan principalmente los valores de batería tras un uso prolongado y la fluidez de los frames mientras se utiliza.

Los resultados de esta prueba han sido poco destacables, la aplicación no tiene un consumo que se pueda considerar elevado y ni siquiera ha activado las alarmas de consumo elevado de los dispositivos móviles en los que se ha probado.

6.2.4 Pruebas de experiencia de usuario

Las pruebas de experiencia de usuario se han realizado con usuario monitorizados para identificar errores en el diseño.

Generalmente las pruebas han sido satisfactorias a pesar de que sí han sido necesarios algunos cambios tras identificar que la mayoría de usuarios no identifican los deslizadores bloqueados como bloqueados y tampoco abren el libro de fórmulas.

6.3 Encuesta

Para validar los objetivos de la aplicación de funcionalidad y usabilidad y comprobar su efectividad se ha creado una encuesta con 2 preguntas de opinión sobre la experiencia de usuario general, 1 pregunta de trasfondo, 2 de problemas para comprobar la efectividad del aprendizaje con la aplicación y una última pregunta de sensaciones tras jugar tres niveles de tiro parabólico.

La encuesta ha tenido como objetivo personas de todas las edades priorizando aquellas que no tuvieran estudios de física o matemáticas ya que son las personas que más pueden aprender usando la aplicación.

Para los puntos 2 a 5 de la sección 6.1 se ha creado una encuesta que realizan todos los usuarios de la aplicación.

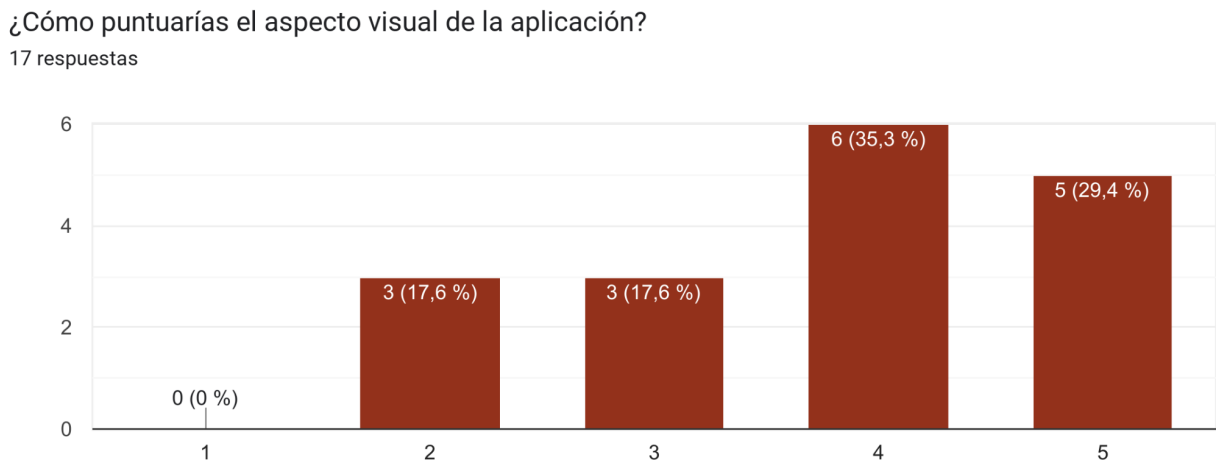

Figura 6.1. Resultado de la primera pregunta del cuestionario

En la primera pregunta (Figura 6.1) se recogen las opiniones del aspecto gráfico de la aplicación. En general las opiniones son positivas pero tiene bastante margen de mejora. El principal problema con los gráficos en la aplicación era la falta de énfasis de algunos botones como el del libro o los de zoom, que se solucionan teniendo iconos con colores e iconos en lugar de texto.

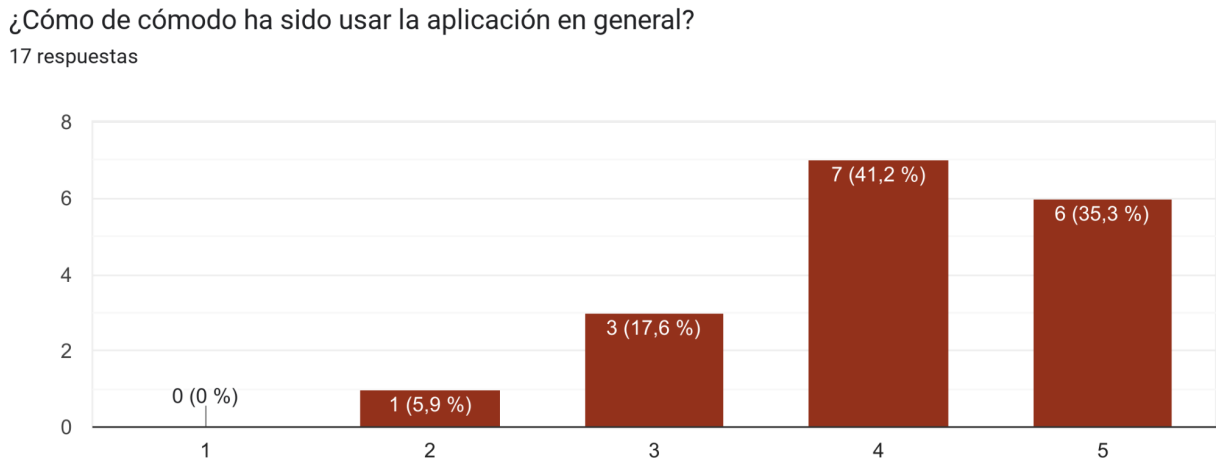

Figura 6.2. Resultados de la pregunta de usabilidad.

Las sensaciones de la usabilidad son buenas, se valoran en gran medida el uso de deslizadores en lugar de teclado para manejar los controles y la cantidad de elementos táctiles de control (Figura 6.2).

Figura 6.3. Resultado de la pregunta de trasfondo.

Entre los encuestados hay un tercio que, según su criterio, saben resolver problemas de tiro parabólico y otra cantidad igual que no saben resolverlos previamente como se muestra en la gráfica de la figura 6.3.

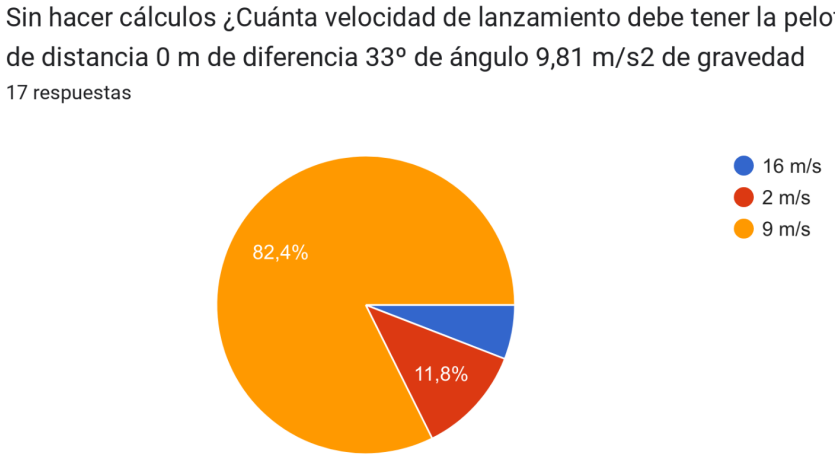

Figura 6.4. Resultado del primer problema.

El primer problema analiza la intuición desarrollada tras usar la simulación del tiro parabólico. En esta pregunta (Figura 6.4) deben elegir entre tres valores muy dispares de los cuales el tercero es el correcto. Los resultados de esta pregunta son buenos ya que los metros

por segundo es una medida de velocidad que no se utiliza de forma cotidiana y aún así la amplia mayoría de usuarios han sabido contestar correctamente.

Figura 6.5. Resultado del segundo problema.

El segundo problema comprueba la comprensión del simulador y de los diálogos con una pregunta de mayor peso teórico. La respuesta correcta es la tercera (Figura 6.5), mencionada en los diálogos, pero la segunda recibe casi un cuarto de los votos ya que en un lanzamiento parabólico la altura máxima suele parecer suceder en la mitad del tiempo. Esta pregunta comprueba la efectividad de la explicación por medio de diálogos y gráficos en movimiento.

Figura 6.6. Resultado de las sensaciones tras usar la aplicación.

La última pregunta es de opinión para averiguar con qué sensaciones queda el usuario tras terminar los problemas (Figura 6.6). Todos los usuarios lograron terminar los 3 problemas propuestos y en comparación a los resultados de la misma pregunta antes de empezar a usar el problema (Figura 6.3) hay cierta mejoría.

Los resultados de la encuesta cuentan con 17 respuestas de las 30 personas a las que se les ofreció realizar la encuesta. Los resultados están dentro de lo esperado ya que la opinión de saber resolver problemas de tiro parabólico son mejores después de jugar y las dos preguntas de teoría tienen más de dos tercios de acierto.

7 Conclusiones

En este proyecto se ha desarrollado un corte vertical de una aplicación de apoyo a estudiantes de la ESO y bachillerato con la posibilidad de adaptarse a su aprendizaje y servir de simulador para realizar pruebas, ejercicios y aprender procesos de resolución de problemas.

En cuanto a los objetivos se puede afirmar que se han cumplido los objetivos mínimos para un corte vertical de la aplicación descritos en la sección 1.2. Asimismo estos objetivos tienen la posibilidad de ser ampliados si se quisiera convertir en una aplicación comercial. Entre los subobjetivos el primero en realizarse fue el análisis del estado del arte donde se identificó la carencia de aplicaciones educativas interactivas utilizables en un aula, descrito en la sección 2.1.

Posteriormente se abordaron las tareas de diseño de experiencia de usuario y ludificación simultáneamente con la que se consigue una interfaz rápida de usar, con su posterior análisis de eficacia en la encuesta de la sección 6.3.

Una vez establecido el diseño se implementó todo lo escrito anteriormente en la aplicación con algunas particularidades que se describen en el capítulo 5. Finalmente la aplicación ha sido validada usando diferentes pruebas y una encuesta cuyos resultados han sido satisfactorios en base a los objetivos.

Este TFG ha resultado un reto en cuanto a la creación de un videojuego educativo, ya que es un concepto que no suele ponerse sobre la mesa cuando se enseña a diseñar videojuegos. El uso de Godot como motor también ha sido una novedad por la apuesta personal de usar un motor de código abierto en lugar de uno privado y comercial como Unity o Unreal, además de las ventajas que ofrece el propio motor en cuanto al desarrollo.

Finalmente la decisión de crear una guía docente ha resultado ser un reto interesante para ampliar las posibilidades de usar el juego como herramienta educativa en un aula. Ya que

es un anexo que no suele presentarse con aplicaciones móviles resulta ser una particularidad de este proyecto.

7.1 Relación entre el trabajo y los estudios cursados

Para el desarrollo de esta aplicación han sido de gran utilidad los conocimientos aprendidos en el Grado en Tecnologías Interactivas. A continuación se describe una lista con los conocimientos prácticos y teóricos más relevantes:

- Programación en Python por sus similitudes con GDScript impartida en varias asignaturas.
- Inglés técnico para la comprensión de la documentación de las herramientas utilizadas cuya traducción al castellano es, en ocasiones, escasa.
- Conocimientos de desarrollo de interfaces con el fin de crear una experiencia al usuario usable e intuitiva, impartidos en la asignatura de Diseño de interfaces y experiencia de usuario.
- Conocimientos de desarrollo de un proyecto desde cero, impartidos en todas las asignaturas de proyecto.
- Conocimientos de desarrollar un videojuego en la asignatura Proyecto Aplicaciones Multimedia Interactivas, comprendiendo también métodos de gamificación extraídos de la teoría de diseño de juegos.
- Conocimientos para aplicar la metodología Scrum a proyectos de software, impartidos en todas las asignaturas de proyecto y especialmente en la asignatura de Desarrollo de un proyecto electrónico utilizando metodología CDIO.

7.2 Objetivos de desarrollo sostenible

El presente proyecto se relaciona con el objetivo de desarrollo número 4: una educación de calidad. Los objetivos de este juego y aplicación se alinean con los objetivos de los ODS de permitir a alumnos y alumnas de secundaria terminar la educación obligatoria

independientemente de su entorno o problemas de accesibilidad. Este compromiso además es especial en este proyecto ya que la accesibilidad y el diseño de contenidos para que sean comprensibles por cualquiera han sido uno de los pilares de su desarrollo.

7.3 Trabajos futuros

En la realización de esta aplicación se han alcanzado los objetivos mínimos creando un corte vertical de lo que puede llegar a ser. En la sección 3.3 se establecen una serie de posibles ampliaciones de la aplicación para llegar a ser usable en aulas como método de aprendizaje gamificado. Algunas ampliaciones posibles son ampliar los contenidos, crear una herramienta para editar niveles y crear nuevos niveles, ampliar la variedad de problemas con sus respectivos controles y arte e implementar una forma de registro de usuarios para guardar sus datos y poder llevar un seguimiento más preciso por parte del profesorado.

Además la aplicación requiere de un rediseño de la experiencia de usuario que se ha podido comprobar poco efectiva en el uso de la aplicación, ya que al realizar pruebas monitorizadas la mayoría de usuarios tenían problemas para identificar los controles inactivos y el libro de fórmulas.

8 Referencias

Anderson, A., Huttenlocher, D., Kleinberg, J., & Leskovec, J. (2013). *Steering User Behavior with Badges*. https://www.cs.cornell.edu/home/kleinber/www13-badges.pdf

Cherner, Todd; DIX, Judy; LEE, Corey. Cleaning up that mess: A framework for classifying educational apps. Contemporary issues in technology and teacher education, 2014, vol. 14, no 2, p. 158-193.

Contreras Espinosa, R. S., & Eguia, J. L. (2017). *Experiencias de gamificación en aulas*. InCom-UAB Publicacions, 15. Bellaterra: Institut de la Comunicació, Universitat Autònoma de Barcelona. 978-84-944171-6-0

Cunningham, W. (2001). *Manifesto for Agile Software Development*. Manifesto for Agile Software Development. Retrieved March 3, 2023, from https://agilemanifesto.org/

Erosa García, D. (2019, June 10). Qué es Unity y características principales. *OpenWebinars*. https://openwebinars.net/blog/que-es-unity/

Godot community. (2014). *Using signals — Godot Engine (stable) documentation in English*. Godot Docs. Retrieved June 2, 2023, from https://docs.godotengine.org/en/stable/getting_started/step_by_step/signals.html

Google. (2014). *Material Design*. Material Design. Retrieved April 2, 2023, from https://m3.material.io/

Hamilton, T. (2023, June 24). *Game Testing: Types & How to Test Mobile/Desktop Apps*. Guru99. Retrieved July 11, 2023, from https://www.guru99.com/game-testing-mobile-desktop-apps.html

Henrique, J., & Silva, M. (2023, March 18). Godot Engine 4.0 vs Unity: A Comprehensive Comparison of Game Engines. *Medium*. https://medium.com/the-polyglot-programmer/godot-engine-4-0-vs-unity-a-comprehensive-comparison-of-game-engines-592abc466f32

Hohensee, B. (2014). *Introducción a Android Studio. Incluye proyectos reales y el código fuente.* Babelcube Inc.

Linietsky, J., Manzur, A., & Godot community. (2014). *Features*. Godot Engine. Retrieved April 2, 2023, from https://godotengine.org/features/

Linietsky, J., Manzur, A., & Godot community. (2014). *Godot's design philosophy — Godot Engine (stable) documentation in English*. Godot Docs. Retrieved April 2, 2023, from https://docs.godotengine.org/en/stable/getting_started/introduction/godot_design_philosophy.html

Noë, A. (2017, July 14). More Bad News For Brain-Training Games : 13.7: Cosmos And Culture. *NPR*. https://www.npr.org/sections/13.7/2017/07/14/536759455/more-bad-news-for-brain-training-games

Rusk, N., & Larson, R. W. (2011). Intrinsic Motivation and Positive Development. *Advances in Child Development and Behavior*, *41*(5), 89-130. https://www.sciencedirect.com/science/article/abs/pii/B9780123864925000051?via%3Dihub

Salge, Christoph; Glackin, Cornelius; Polani, Daniel. Changing the environment based on empowerment as intrinsic motivation. Entropy, 2014, vol. 16, no 5, p. 2789-2819.

Tyre, P., & Buder, E. (2016, March 15). A Group of American Teens Are Excelling at Advanced Math. *The Atlantic*. https://www.theatlantic.com/magazine/archive/2016/03/the-math-revolution/426855/

Werbach, Kevin. (Re) defining gamification: A process approach. En Persuasive Technology: 9th International Conference, PERSUASIVE 2014, Padua, Italy, May 21-23, 2014. Proceedings 9. Springer International Publishing, 2014. p. 266-272.

Zaman, B., & van Roy, R. (2016). Why Gamification Fails in Education-And How to Make it Successful. *Serious Games and Edutainment Applications*, *II*, 485-509.