

Discovering 2.0

La aplicación que enseña curiosidades sobre el mundo animal
haciendo uso de realidad virtual

Trabajo de Fin de Grado

GRADO EN INGENIERÍA INFORMÁTICA



**VNiVERSiDAD
D SALAMANCA**

Julio de 2023

Autora

Cristina Alejandra Crespo Jiménez

Tutores

Gabriel Villarrubia González

André Filipe Sales Mendes



Certificado de los tutores

D. Gabriel Villarrubia González y D. André Filipe Sales Mendes, profesor del Departamento de Informática y Automática de la Universidad de Salamanca,

HACEN CONSTAR:

Que el trabajo titulado “Discovering 2.0: La aplicación que enseña curiosidades sobre el mundo animal haciendo uso de realidad virtual”, que se presenta, ha sido realizado por Cristina Alejandra Crespo Jiménez, con DNI 70824499R y constituye la memoria del trabajo realizado para la superación de la asignatura Trabajo de Fin de Grado de la Titulación Grado en Ingeniería Informática de esta Universidad.

Y para que así conste a todos los efectos oportunos.

En Salamanca, a 3 de Julio de 2023

Fdo.:

Resumen

Gracias a los nuevos avances tecnológicos, los videojuegos han experimentado una gran evolución, han pasado de ser simples píxeles en 2D a parecer hiperrealistas o incluso ser experiencias inmersivas, como es el caso del proyecto que se presenta. Los videojuegos educativos en realidad virtual (VR) proporcionan una herramienta innovadora para el aprendizaje y una gran ayuda en las aulas, el jugador se sumerge en un entorno de 360° donde se pueden enseñar cosas complejas de explicar, facilita la comprensión, los conocimientos adquiridos son más duraderos, hace que los alumnos estén más atentos y que aprendan de forma más rápida.

“*Discovering*” se ha diseñado para brindar una experiencia auténtica en el fondo marino donde tendrás que ir buscando a los animales, representados de una forma bastante realista y a tamaño real, los cuales serán indicados por una libreta donde podrás ir viendo tus logros y objetivos restantes. Se proporcionan curiosidades y datos interesantes sobre cada especie de animal encontrada, lo que promueve el aprendizaje de manera entretenida y emocionante. También incorpora elementos de *gamificación** para mantener el interés y la motivación del jugador que consiste en encontrar objetos escondidos por la escena.

La combinación de la tecnología de VR, la representación visual realista y la *gamificación** estimula la curiosidad, el pensamiento crítico y la adquisición de conocimientos de una manera divertida y motivadora. Este juego contribuye a despertar el interés de los jugadores por los animales y promueve el desarrollo de habilidades cognitivas.

Se han empleado como gafas de VR las Oculus Quest, junto con el motor de videojuego Unity para el diseño de las escenas. La autenticación, gestión de usuarios y almacenamiento de datos se ha realizado con Firebase, una plataforma en la nube creada por Google.

Palabras clave: videojuego, realidad virtual (VR), Unity, C#, Oculus, Firebase, Cloud Firestore

*En el contexto de los videojuegos, la gamificación implica la inclusión de elementos como desafíos, recompensas, puntajes, niveles o tablas de clasificación, entre otros, para mantener el interés del jugador y promover su compromiso activo con el juego. Estos elementos se utilizan para crear una experiencia más atractiva y motivadora, estimulando el progreso, la competencia sana y el logro de metas dentro del juego.



Abstract

Thanks to recent technological advances, videogames have undergone an enormous evolution, transitioning from simple 2D pixels games to hyper-realistic visuals or even immersive experiences, as is the case of the presented project. Educational virtual reality (VR) videogames provide an innovative tool for learning and a great help in classrooms. By immersing players in a 360° environment, complex concepts can be effectively taught, facilitating understanding and fostering long-lasting knowledge retention. VR gaming captivates students' attention, leading to faster and more efficient learning.

“*Discovering*” has been designed to offer an authentic experience in the underwater world where players will have to search for animals, represented in a realistic and life-sized way, which are indicated by a notebook where you can track your achievements and remaining objectives. Interesting facts about each encountered animal will be provided, promoting an enjoyable and exciting learning experience. Additionally, *gamification** elements are incorporated to maintain the interest and the motivation of the player, involving the search for hidden objects within the scene.

The combination of VR technology, realistic visual representation and *gamification** elements stimulates curiosity, critical thinking and knowledge acquisition in a fun and motivating manner. This game contributes to cultivating players’ interest in marine animals and promotes the development of cognitive skills.

For the VR experience, Oculus Quest headsets were employed, along with the Unity game engine for scene design. Authentication, user management and data storage were implemented with Firebase, a cloud platform created by Google.

Key words: videogame, virtual reality (VR), Unity, C#, Oculus, Firebase, Cloud Firestore.

*In the context of video games, gamification involves incorporating elements such as challenges, rewards, scores, levels, or leaderboards, among others, to maintain player interest and promote active engagement with the game. These elements are used to create a more engaging and motivating experience, stimulating progress, healthy competition, and goal achievement within the game.



Índice

1. Introducción.....	1
2. Objetivos del proyecto.....	4
2.1. Objetivos del sistema.....	4
2.2. Objetivos personales.....	4
3. Conceptos teóricos.....	5
3.1. Herramientas SDK.....	5
3.2. Firebase.....	5
3.3. Herramientas CASE.....	5
3.4. Framework.....	5
4. Técnicas y herramientas.....	6
4.1. Entorno videojuego.....	6
4.1.1. C#.....	6
4.1.2. Unity.....	6
4.1.3. Oculus Integration.....	7
4.1.4. Firebase Authentication.....	7
4.1.5. Firebase Cloud Firestore.....	8
4.2. Entorno desarrollo.....	9
4.2.1. GitHub.....	9
4.2.2. Source Tree.....	9
4.2.3. Visual Studio Code.....	9
4.3. Herramientas CASE.....	9
4.3.1. Draw.io.....	9
4.3.2. EZEstimate.....	9
4.3.3. Microsoft Project.....	9
4.4. Herramientas Documentación.....	10
4.4.1. Doxygen.....	10
4.4.2. Google Document.....	10
4.4.3. MyBib.....	10
5. Aspectos relevantes del desarrollo.....	11
5.1. Marco de trabajo.....	11
5.2. Estimación de costes del proyecto.....	12
5.3. Planificación temporal.....	13
5.4. Especificación de requisitos.....	15
5.4.1. Participantes.....	16
5.4.2. Objetivos del sistema.....	16
5.4.3. Requisitos del sistema.....	17
5.4.3.1. Requisitos de información.....	17



5.4.3.2. Requisitos funcionales.....	18
5.4.3.3. Diagrama de paquetes.....	18
5.4.3.4. Definición de actores.....	19
5.4.3.5. Casos de uso del sistema.....	20
5.4.3.6. Requisitos no funcionales.....	22
5.4.4. Matriz de rastreabilidad.....	22
5.5. Análisis de requisitos.....	22
5.5.1. Modelo de dominio.....	23
5.5.2. Realización de casos de uso - Análisis.....	23
5.5.3. Clases de análisis.....	24
5.5.4. Vista arquitectónica modelo de análisis.....	25
5.6. Diseño del sistema.....	25
5.6.1. Patrones arquitectónicos.....	25
5.6.1.1. Patrón MVC.....	25
5.6.1.2. Patrón Entidad-Componente (EC).....	26
5.6.2. Subsistemas de diseño.....	27
5.6.3. Clases de diseño.....	28
5.6.4. Vista arquitectónica modelo de diseño.....	28
5.6.5. Realización de casos de uso - Diseño.....	29
5.6.6. Diseño de base de datos.....	30
5.6.7. Modelo de despliegue.....	31
5.7. Implementación.....	31
5.8. Pruebas.....	32
6. Funcionalidad del sistema.....	33
7. Limitaciones de la aplicación.....	39
8. Conclusiones y líneas de trabajo futuras.....	40
8.1. Conclusiones.....	40
8.2. Líneas de trabajo futuras.....	40
9. Referencias bibliográficas.....	42

ANEXO I: Plan del proyecto

ANEXO II: Especificación de requisitos software

ANEXO III: Análisis de requisitos

ANEXO IV: Diseño del sistema

ANEXO V: Documentación técnica

ANEXO VI: Manual de usuario



Índice de tablas

Tabla 1: Participante Cristina A. Crespo Jiménez.....	16
Tabla 2: OBJ-002 Gestión de elementos encontrados.....	17
Tabla 3: IRQ-001 Información sobre los usuarios.....	18
Tabla 4: ACT-001 Usuario anónimo.....	20
Tabla 5: UC-008 Consultar libreta.....	22
Tabla 6: Matriz de rastreabilidad, Objetivo / Requisito.....	22

Índice de figuras

Figura 1: El aumento visualizado de los ingresos de los juegos.....	2
Figura 2: Interfaz de Unity.....	7
Figura 3: Firebase Authentication	8
Figura 4: Firebase Cloud Firestore	8
Figura 5: Ciclo de vida.....	12
Figura 6: Estimación de costes.....	13
Figura 7: Planificación temporal.....	14
Figura 8: Diagrama de Gantt.....	15
Figura 9: Diagrama de paquetes.....	19
Figura 10: Actores.....	19
Figura 11: Diagrama de casos de uso.....	20
Figura 13: Diagrama de clases.....	23
Figura 14: Diagrama de secuencia UC-008.....	24
Figura 15: Diagrama de comunicación, Gestión de Usuarios.....	25
Figura 16: Esquema patrón MVC.....	26
Figura 17: Problema herencia múltiple.....	27
Figura 18: Subsistema de diseño.....	27
Figura 19: Clase de diseño - Principal Scene.....	28
Figura 20: Diagrama de la base de datos.....	29
Figura 21: Diagrama de secuencia de diseño UC-013.....	30
Figura 22: Diagrama de la base de datos.....	30
Figura 23: Diagrama de despliegue.....	31
Figura 24: Controles mandos de las Oculus Quest.....	33
Figura 25: Pantalla inicial.....	34
Figura 26: Pantalla inicial.....	35
Figura 27: Pantalla océano.....	36
Figura 28: Pantalla océano - selección animal.....	36
Figura 29: Pantalla océano - información animal.....	37
Figura 30: Vista del escenario del océano.....	38

1. Introducción

Desde los años 70 hemos convivido con los videojuegos y hemos visto su evolución a lo largo de los años, como vemos en la *Figura 1*, desde las máquinas de arcade, pasando por las primeras consolas (Atari 2600, NES, Game Boy, PlayStation...), por los primeros juegos de ordenador (PC), consolas más nuevas, la aparición de los móviles y su adaptación de los juegos, hasta la actualidad con las gafas de realidad virtual (VR) [1].

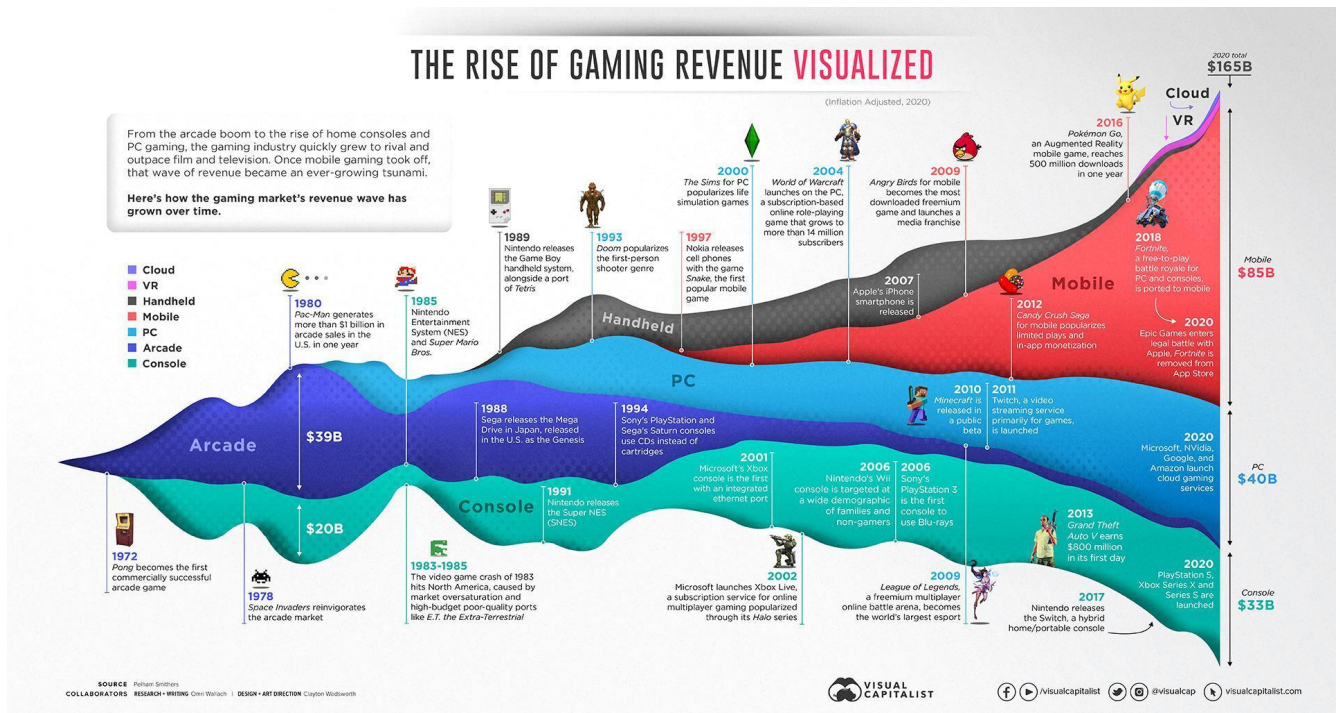


Figura 1: El aumento visualizado de los ingresos de los juegos (Garijo, 2020) [2]

Estos enormes avances tecnológicos han impulsado el crecimiento exponencial de los videojuegos, llevándolos a evolucionar rápidamente en diversos aspectos, aprovechando de manera efectiva el potencial que estos tienen para su aplicación educativa. De esta forma se atrae y motiva a los estudiantes a través de juegos interactivos que se combinan con diversión y aprendizaje. La toma de decisiones, resolución de problemas, enfrentarse a desafíos dentro de los juegos promueven el pensamiento crítico, el razonamiento lógico y estimula la curiosidad. Los conocimientos adquiridos son más duraderos, mejora su aprendizaje al poder ser más inclusivo e individualizado, facilita la comprensión y les ayuda a mantener la atención [3].

Este proyecto se ha realizado desde este enfoque, para agilizar y mejorar el aprendizaje en las aulas. Es por ello que se ha desarrollado el videojuego en VR empleando las herramientas

que Oculus proporciona para el motor de juegos de Unity. *Discovering* consta de dos pantallas, la pantalla principal donde se gestiona a los usuarios, y el escenario del fondo marino donde se podrán ver, en un limitado espacio, varias especies marinas a tamaño real. El usuario tendrá a su disposición una libreta la cual le irá informando sobre los animales que debe buscar por la escena, mostrando una estrella bajo aquellos que ya ha descubierto. Una vez encontradas, se mostrarán diez curiosidades y datos interesantes sobre cada uno de ellos. A mayores habrá objetos escondidos que tendrá que encontrar, mostrando en este caso cinco curiosidades sobre ellos, promoviendo así la motivación y el interés.

A lo largo de la memoria se explicarán los conceptos y aspectos más importantes del desarrollo del proyecto, organizándose de la siguiente manera:

- Objetivos del proyecto: se exponen los objetivos que se deben satisfacer en la creación del proyecto.
- Conceptos teóricos: se recogen los conceptos básicos para poder entender el funcionamiento del proyecto.
- Técnicas y herramientas: se documentan las técnicas y herramientas empleadas para realizar el proyecto.
- Aspectos relevantes del desarrollo: se detallan las tareas principales llevadas a cabo en el proyecto.
- Funcionalidad del sistema: se explica paso por paso la dinámica del juego y el funcionamiento de las diferentes interacciones con la escena.
- Limitaciones de la aplicación: se especifican los obstáculos encontrados en la realización del proyecto.
- Conclusiones y líneas de trabajo futuras: se presentan las conclusiones y se identifican mejoras para futuras implementaciones.
- Referencias bibliográficas: se listan todas las fuentes consultadas a lo largo del proyecto.

Adicionalmente, se entrega el código fuente del desarrollo del proyecto y el conjunto de anexos que recogen la documentación técnica, organizada de la siguiente manera:

- Anexo I: Plan del proyecto, se realiza una estimación de la planificación temporal del proyecto.
- Anexo II: Especificación de requisitos software, se desglosa el proceso de recogida y documentación de requisitos.
- Anexo III: Análisis de requisitos, se explican las tareas relacionadas con el análisis de los requisitos especificados en el *Anexo II*.



- Anexo IV: Diseño del sistema, se documenta el diseño del sistema a partir de las especificaciones realizadas en el *Anexo III*.
- Anexo V: Documentación técnica, se detalla la documentación relevante del código fuente del proyecto.
- Anexo VI: Manual de usuario, se especifica el proceso a seguir en cada pantalla para que los usuarios puedan entender el correcto funcionamiento de cada interacción con el videojuego.

2. Objetivos del proyecto

En este apartado se exponen los objetivos que se pretenden alcanzar para poder llevar a cabo el desarrollo del proyecto, tanto los objetivos del sistema como los personales.

2.1. Objetivos del sistema

El principal objetivo del proyecto es el desarrollo de un videojuego educativo junto con las finalidades que conlleva el mismo. Estos objetivos son:

- Gestión de usuarios: el sistema debe gestionar el registro de usuarios, así como el inicio de sesión si previamente te has registrado.
- Gestión de elementos encontrados: el sistema debe gestionar el guardado de los datos, tanto de sus claves como de su progreso.
- Gestión de la información: el sistema debe mostrar la información relativa al progreso del usuario dentro del videojuego.
- Gestión de las escenas: el sistema debe permitir que los usuarios se muevan por la escena así como interactuar con los objetos contenidos en la misma.

2.2. Objetivos personales

La principal finalidad por la cual se lleva a cabo este proyecto es la motivación para poder realizar un videojuego interactivo e inmersivo que enseñe mientras te diviertes. Poder llevar a las aulas nuevas tecnologías que despierten la curiosidad de los más pequeños desde otra perspectiva de la informática que todo el mundo conoce como ceros y unos.

De pequeña siempre me apasionó el mundo que nos rodea y me imaginaba cómo sería el mundo a través de los ojos de otro ser vivo, como sería vivir bajo el mar y observar el fondo oceánico en todo su esplendor. Resulta gratificante poder desarrollar ese sueño y verlo hacerse realidad de forma virtual, es por eso que se han elegido modelos de animales realistas a tamaño real, y me gustaría poder mostrarles a los niños lo que una vez yo deseé experimentar.

3. Conceptos teóricos

En este apartado se recogen los conceptos básicos empleados a lo largo del proyecto, que se explicarán a continuación para un mejor entendimiento de la memoria presentada.

3.1. Herramientas SDK

Software Development Kit, o en español, Kit de Desarrollo de Software, es un conjunto de herramientas, bibliotecas, documentación y ejemplos que se proporcionan a desarrolladores para facilitar la creación de aplicaciones para una plataforma o tecnología específica. De esta forma se les permite ahorrar tiempo en las implementaciones que puedan ser más complejas [4].

3.2. Firebase

“*Firebase es una plataforma de desarrollo de apps que te ayuda a compilar y desarrollar las apps y los juegos que les encantan a los usuarios. Con el respaldo de Google y la confianza de millones de empresas de todo el mundo.*” [5]. Ofrece varios servicios como son los productos de compilación, de lanzamiento y supervisión, o de participación, siendo compatibles con diferentes plataformas como iOS, Android, Unity y web. Incluye también herramientas SDK.

Algunos de estos servicios son *Authentication*, facilita la creación y autenticación segura de usuarios, *Cloud Firestore*, proporciona una base de datos NoSQL en la nube, *Realtime Database*, igual que la anterior pero almacenando y sincronizando los datos en tiempo real, *Google Analytics*, proporciona informes ilimitados de hasta 500 eventos diferentes.

3.3. Herramientas CASE

Son un conjunto de aplicaciones informáticas que automatizan actividades en el desarrollo de software. Son utilizadas por directores de proyectos, analistas e ingenieros para agilizar el ciclo de vida del software, incluyendo análisis, diseño, gestión de proyectos, bases de datos, y documentación. Estas herramientas aceleran el desarrollo, ayudan a encontrar errores y mejoran la eficiencia del proceso[6].

3.4. Framework

Se trata de un conjunto de herramientas, bibliotecas y pautas de diseño que proporciona una estructura para el desarrollo de aplicaciones, permitiendo a los desarrolladores llevar a cabo tareas comunes o complejas de una manera más rápida gracias a los componentes reutilizables,

las arquitecturas predefinidas. De esta forma, se reducen los errores, se obtiene un mejor resultado y facilita el desarrollo colaborativo [7].

4. Técnicas y herramientas

En este apartado se documentan las técnicas, las herramientas de desarrollo, los lenguajes de programación y los frameworks que se han empleado para llevar a cabo el proyecto. Se distinguirá entre los entornos de videojuego y de desarrollo, las herramientas CASE y las de documentación.

4.1. Entorno videojuego

4.1.1. C#

Este lenguaje de programación (pronunciado "C sharp") está desarrollado por Microsoft como parte de su plataforma .NET. Es un lenguaje de programación orientado a objetos que combina elementos de C y C++. C# se diseñó principalmente para el desarrollo de aplicaciones de software en el entorno de Microsoft, incluyendo aplicaciones de escritorio, aplicaciones web y juegos. Tiene recolección de basura, integra el framework .NET, soporta programación concurrente, es muy versátil, sencillo y de tipado seguro [8].

Se ha elegido este lenguaje en concreto debido al soporte y la documentación que ofrece al usarlo en Unity.

4.1.2. Unity

Es un motor de desarrollo de juegos, consiste en un software que incluye un conjunto de rutinas de programación para facilitar la creación, diseño y ejecución de un entorno interactivo. Permite el desarrollo de juegos en 3D, como es la VR, proporcionando el desarrollo de sistemas exportables a diferentes plataformas (PC, Android, iOS...) [9].

Se ha elegido este motor en concreto gracias a los recursos que tiene de Oculus, en particular, el paquete de software *Oculus Integration* que a continuación se explicará.

En la *Figura 2* podemos ver la interfaz de Unity, teniendo a la izquierda la jerarquía de los componentes usados en la escena, a la derecha el inspector que muestra detalladamente el componente seleccionado, abajo el directorio donde se encuentran todo lo empleado en el desarrollo del videojuego, y por último, en el centro, la escena con la que se trabaja.

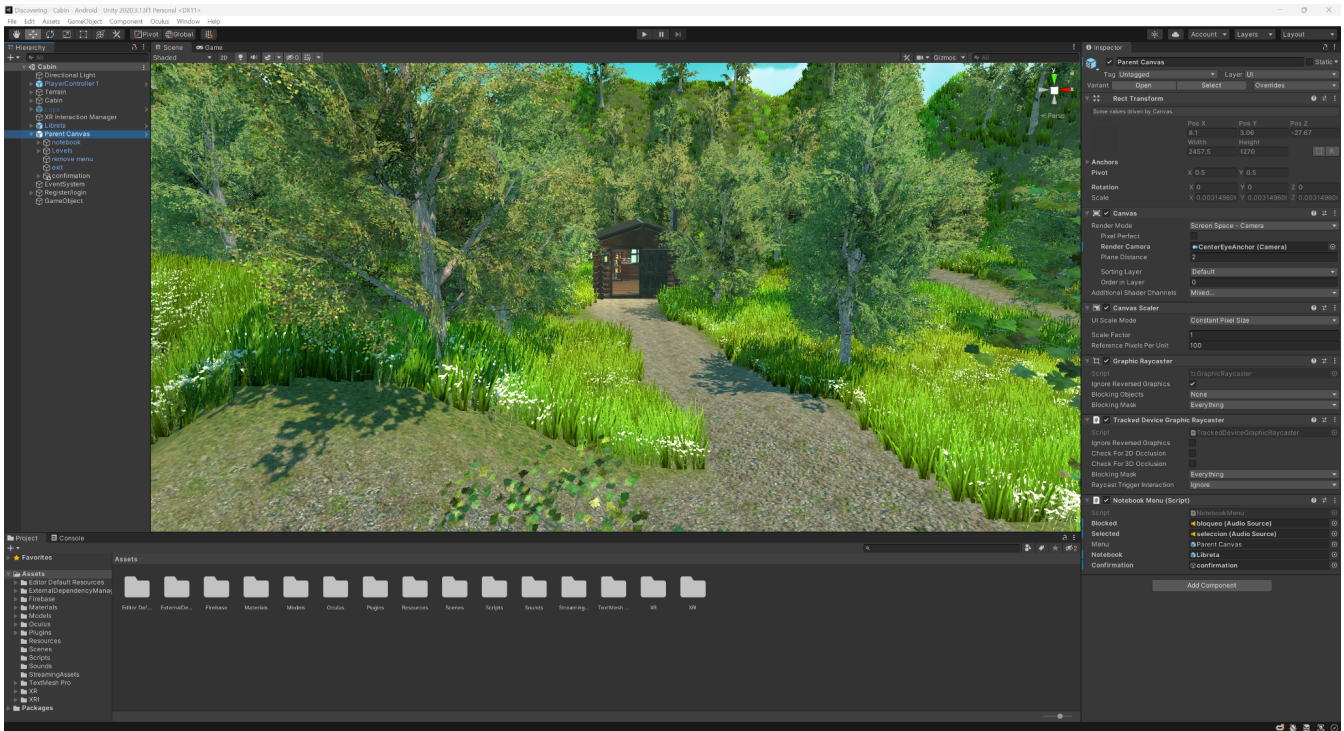


Figura 2: Interfaz de Unity

4.1.3. Oculus Integration

Es un paquete de software que facilita la integración de la tecnología de VR de Oculus en proyectos de Unity. Proporciona herramientas y recursos para configurar y optimizar aplicaciones de VR, soporte para controladores y seguimiento de manos, renderizado optimizado, audio espacial y facilita la implementación de interacciones VR. [10]

Entre las características y funcionalidades que ofrece, hay que destacar el soporte para el seguimiento de manos y los controladores Oculus Touch, lo que permite interactuar con el entorno virtual mediante gestos y movimientos de las manos.

4.1.4. Firebase Authentication

Es un servicio proporcionado por Firebase que te permite simplificar la creación de sistemas de autenticación seguros, al tiempo que mejora la experiencia de integración y acceso para los usuarios finales. La autenticación es compatible con usuarios anónimos, cuentas de correo y contraseñas, acceso mediante Google, Twitter, Facebook, etc. [11].

Se ha usado para la autenticación la cuenta de correo electrónico y contraseña. En la *Figura 3* podemos ver cómo se guardan los datos en la nube.

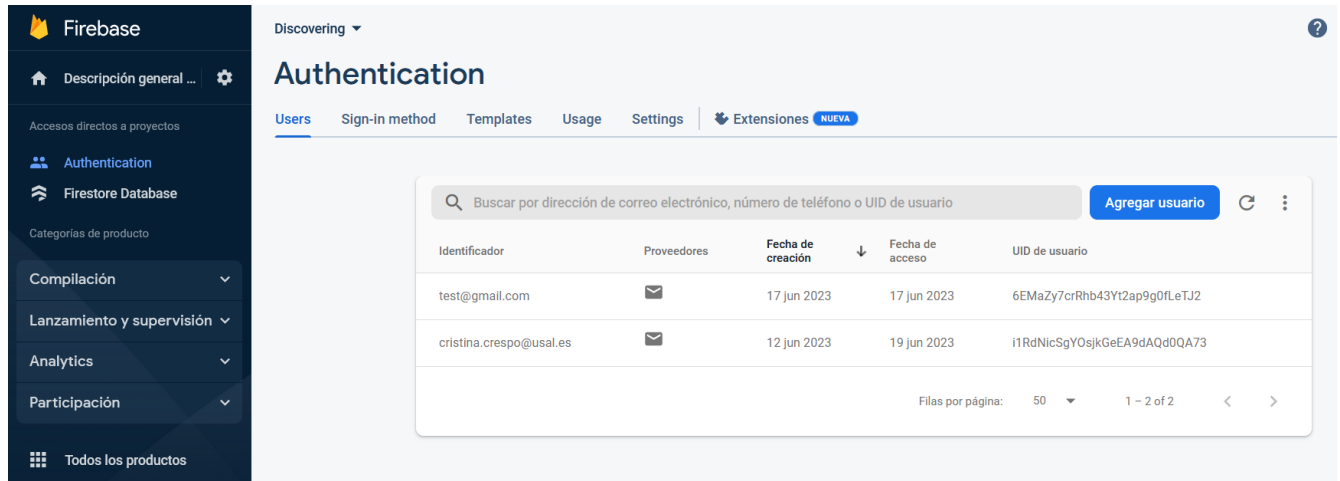


Figura 3: Firebase Authentication [12]

4.1.5. Firebase Cloud Firestore

Es un servicio proporcionado por Firebase que te permite crear una base de datos NoSQL para almacenar, sincronizar y consultar fácilmente datos a gran escala. Ofrece características como sincronización en tiempo real, escalabilidad, consultas avanzadas, seguridad y se integra fácilmente con otros servicios de Firebase [13].

En la *Figura 4* podemos ver cómo se guardan y se estructuran los datos guardados.

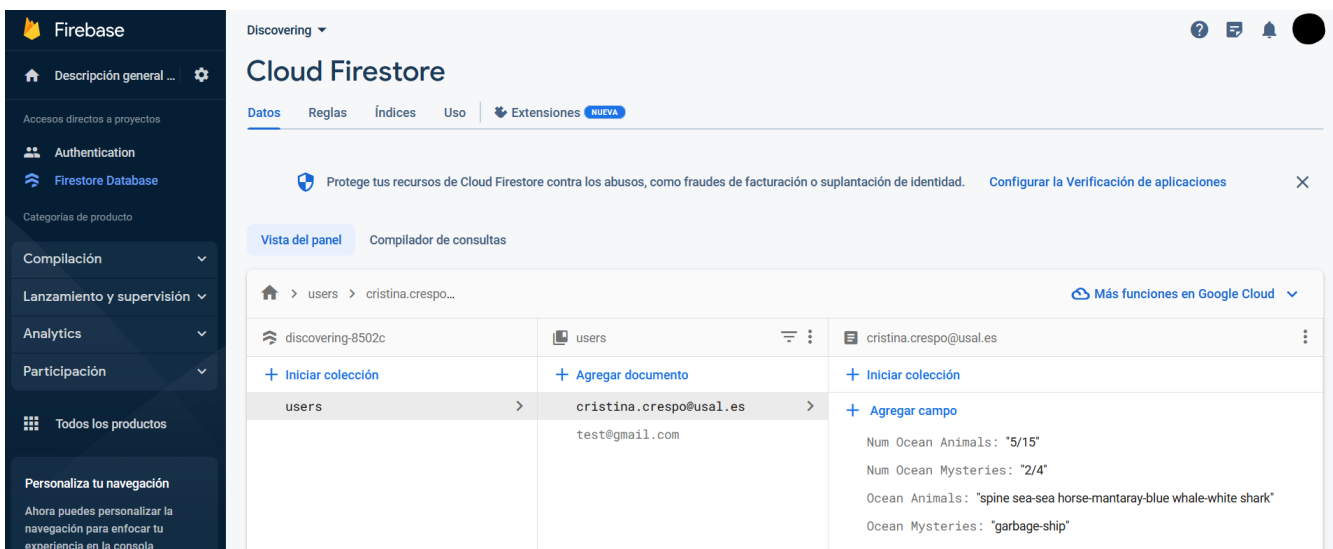


Figura 4: Firebase Cloud Firestore [12]

4.2. Entorno desarrollo

4.2.1. GitHub

Es una de las principales plataformas web que junto con el sistema de control de versiones Git permite crear proyectos abiertos, de esta forma, permite a los desarrolladores trabajar de manera colaborativa en el mismo proyecto. También se pueden crear proyectos privados [14].

4.2.2. Source Tree

Se trata de una aplicación de escritorio que proporciona una interfaz de usuario que ayuda, de una forma más accesible y visual, a trabajar con repositorios Git y/o Mercurial, en este caso, se ha usado Git. Permite realizar operaciones como clonar repositorios, crear ramas, fusionar cambios o gestionar conflictos [15].

4.2.3. Visual Studio Code

Es un editor de código desarrollado por Microsoft. Se ha convertido en uno de los más usados al ser software libre, tener soporte para múltiples lenguajes, estar disponible para diferentes sistemas operativos, contar con integración con Git y una amplia variedad de extensiones [16].

4.3. Herramientas CASE

4.3.1. Draw.io

Es un software libre que te permite diseñar cualquier tipo de diagrama y esquema, como por ejemplo, diagramas de clases, diagramas de secuencia, etc..

4.3.2. EZEstimate

Es una herramienta que nos permite realizar estimaciones de los costes y esfuerzos que supone el desarrollo de un proyecto, basándose en la complejidad de los casos de uso, las interacciones de los actores, y los factores técnicos y del entorno [17].

4.3.3. Microsoft Project

Es una herramienta que nos facilita la planificación del desarrollo de un proyecto mediante la asignación de tareas y recursos, pudiendo así hacer un seguimiento del proyecto.



4.4. Herramientas Documentación

4.4.1. Doxygen

Es una herramienta que te permite generar documentación automatizada de código fuente de un proyecto, incluyendo información sobre clases, funciones, variables y las relaciones entre ellos. Apto para un amplio abanico de lenguajes de programación.

4.4.2. Google Document

Es una herramienta de Google que permite a los usuarios crear, editar y colaborar en documentos de texto en la nube de manera simultánea.

4.4.3. MyBib

Se trata de una herramienta para generar de una forma más rápida todo tipo de citas bibliográficas.

5. Aspectos relevantes del desarrollo

En este apartado se detallan de forma minuciosa todas las tareas y etapas importantes que se han realizado durante el desarrollo del proyecto.

5.1. Marco de trabajo

Un marco de trabajo es un conjunto de conceptos y prácticas estandarizados que ayudan a desarrollar un sistema. Se pueden reducir los costes, dar más consistencia al sistema al usar el marco correcto.

Se ha optado por el Proceso Unificado (PU) al ser el que mejor se adapta a las necesidades del proyecto. Las características que lo definen son:

- Iterativo e incremental, en cada iteración se identifican y definen los casos de uso relevantes, cuando alcanza su objetivo, pasa a la siguiente. Se desarrolla de forma incremental.
- Centrado en arquitectura, representada por las vistas del modelo.
- Conducido por casos de uso, ya que afectan a las fases al contener las descripciones de las funciones.

El PU se repite en varios ciclos de desarrollo que conforman el ciclo de vida del sistema. Cada ciclo consta de cuatro fases:

- Inicio, se define el alcance del proyecto y se desarrollan los casos de negocio.
- Elaboración, se planifica y especifican los detalles de los casos de uso y se diseña la arquitectura.
- Construcción, se construye el producto.
- Transición, se corrigen los problemas y se incorporan mejoras.

Cada fase se divide en una serie de disciplinas, que son:

- Modelado del negocio, donde se procederá a investigar, buscar recursos, definir riesgos, costes y detalles del sistema.
- Requisitos, donde se llevará a cabo el desarrollo de los casos de uso.
- Análisis, donde se analizarán los requisitos y se definirá el comportamiento de los casos de uso.
- Diseño, donde se realizará el modelado de diseño, incluyendo los casos de uso,

diagramas de clase y definición de patrones arquitectónicos.

- Implementación, donde se harán las tareas de programación del sistema.
- Pruebas, donde se efectuarán las pruebas del sistema para comprobar su correcto funcionamiento.

En la *Figura 5* podemos observar las diferentes fases e iteraciones del PU [18].

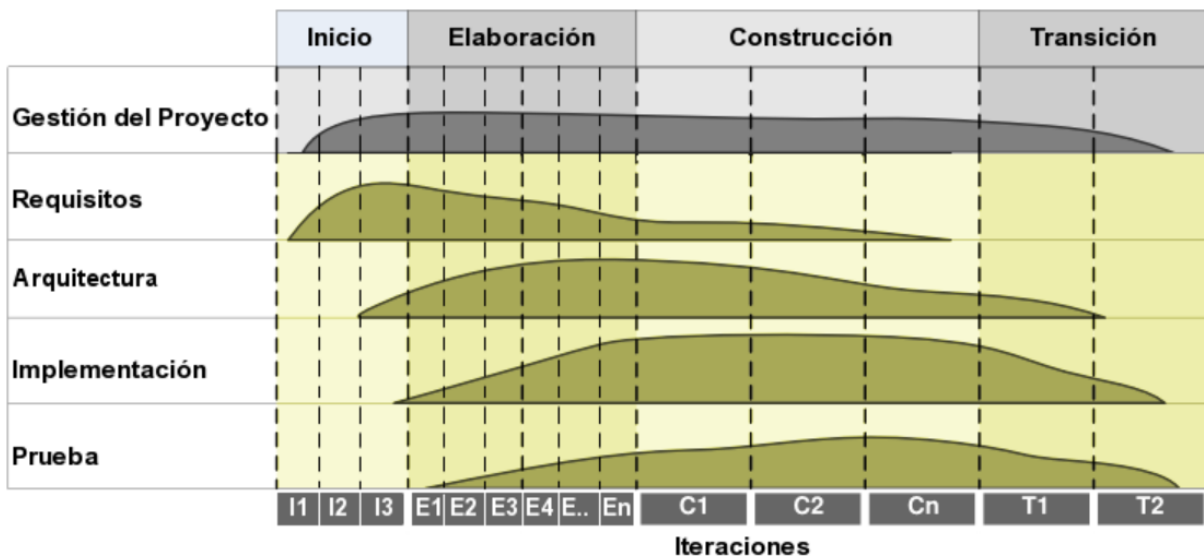
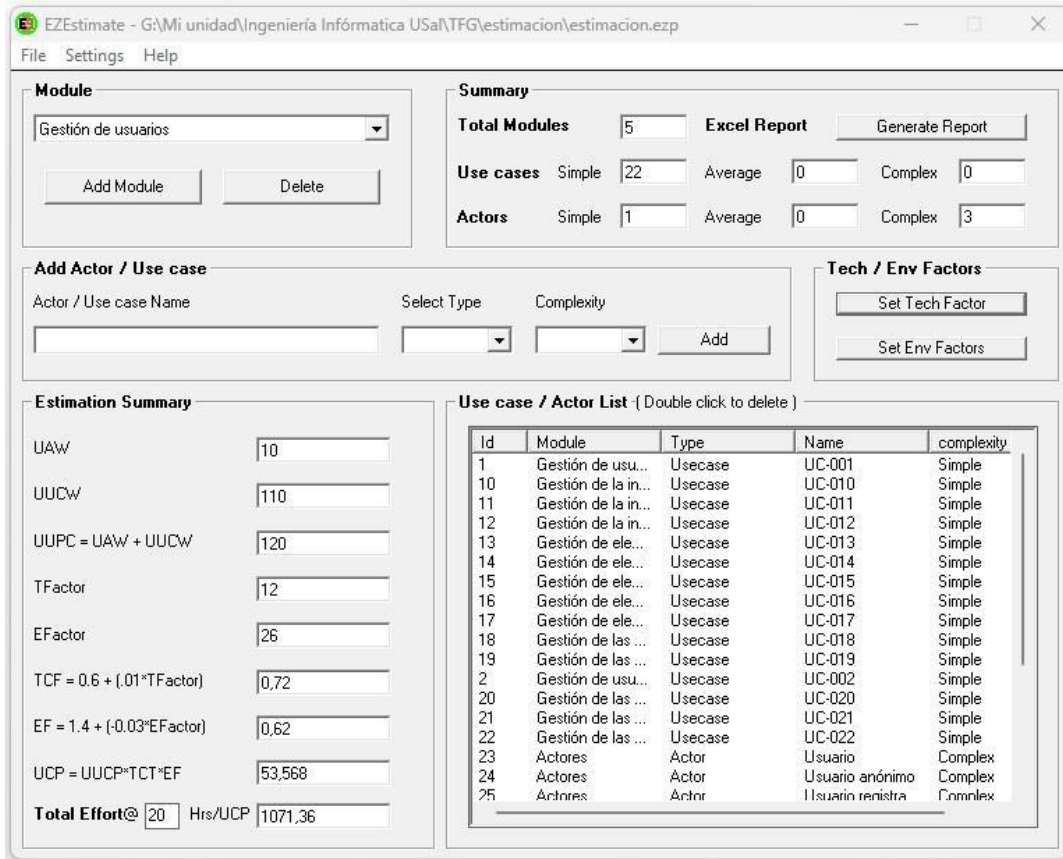


Figura 5: Ciclo de vida

5.2. Estimación de costes del proyecto

El principal objetivo es tener una estimación del tiempo requerido para el desarrollo del proyecto, así la planificación temporal se puede definir de una manera más correcta. Se ha calculado la estimación con ayuda de la herramienta *EZEstimate*, explicada en el apartado 4. *Técnicas y herramientas*, a partir de los casos de uso definidos en el *Anexo II: Especificación de requisitos software*. Si se desea conocer todo el proceso de estimación, puede consultar el *Anexo I: Plan del proyecto*.

En la *Figura 6* podemos ver la interfaz de *EZEstimate*, que nos muestra la estimación realizada a partir de los casos de uso, los actores y los factores de complejidad técnica y del entorno.



The screenshot shows the EZEstimate software interface. The 'Module' dropdown is set to 'Gestión de usuarios'. The 'Summary' section shows 5 total modules, 22 simple use cases, 1 simple actor, and 3 complex actors. The 'Estimation Summary' section shows a total effort of 1071.36 hours. The 'Use case / Actor List' table is as follows:

Id	Module	Type	Name	complexity
1	Gestión de usu...	Usecase	UC-001	Simple
10	Gestión de la in...	Usecase	UC-010	Simple
11	Gestión de la in...	Usecase	UC-011	Simple
12	Gestión de la in...	Usecase	UC-012	Simple
13	Gestión de ele...	Usecase	UC-013	Simple
14	Gestión de ele...	Usecase	UC-014	Simple
15	Gestión de ele...	Usecase	UC-015	Simple
16	Gestión de ele...	Usecase	UC-016	Simple
17	Gestión de ele...	Usecase	UC-017	Simple
18	Gestión de las ...	Usecase	UC-018	Simple
19	Gestión de las ...	Usecase	UC-019	Simple
2	Gestión de usu...	Usecase	UC-002	Simple
20	Gestión de las ...	Usecase	UC-020	Simple
21	Gestión de las ...	Usecase	UC-021	Simple
22	Gestión de las ...	Usecase	UC-022	Simple
23	Actores	Actor	Usuario	Complex
24	Actores	Actor	Usuario anónimo	Complex
25	Actores	Actor	Usuario registra	Complex

Figura 6: Estimación de costes

5.3. Planificación temporal

Una vez realizada la estimación, procedemos a realizar la planificación temporal siguiendo el esquema del PU y usando la herramienta *Microsoft Project*, explicada en el apartado 4. *Técnicas y herramientas*. Si se desea conocer todo el proceso de planificación, puede consultar el *Anexo I: Plan del proyecto*.

En la *Figura 7* podemos observar la planificación realizada siguiendo el esquema del PU, siendo así:

- Fase de Inicio, con dos iteraciones.
- Fase de Elaboración, con 3 iteraciones.
- Fase de Construcción, con 4 iteraciones.
- Fase de Transición, con 3 iteraciones.

Y en la *Figura 8*, vemos el diagrama de Gantt asociado a las tareas vistas en la anterior imagen, que nos ofrece una visualización rápida del desarrollo del proyecto.



	Nombre de tarea	Duración	Comienzo	Fin	Predecesoras
1	Planificación	127 días	mié 08/02/23	lun 19/06/23	
2	Inicio	37 días	mié 08/02/23	vie 17/03/23	
3	Iteración 1	8 días	mié 08/02/23	jue 16/02/23	
16	Hito Fin Iteración 1	0 días	jue 16/02/23	jue 16/02/23	3
17	Iteración 2	29 días	jue 16/02/23	vie 17/03/23	16
32	Hito Fin Iteración 2	0 días	vie 17/03/23	vie 17/03/23	17
33	Hito Fin Fase de Inicio	0 días	vie 17/03/23	vie 17/03/23	2
34	Elaboración	29 días	sáb 18/03/23	lun 17/04/23	33
35	Iteración 1	22 días	sáb 18/03/23	lun 10/04/23	
48	Hito Fin Iteración 1	0 días	lun 10/04/23	lun 10/04/23	35
49	Iteración 2	14 días	sáb 18/03/23	sáb 01/04/23	
62	Hito Fin Iteración 2	0 días	sáb 01/04/23	sáb 01/04/23	49
63	Iteración 3	15 días	sáb 01/04/23	lun 17/04/23	62
76	Hito Fin Iteración 3	0 días	lun 17/04/23	lun 17/04/23	63
77	Hito Fin Fase de Elaboración	0 días	lun 17/04/23	lun 17/04/23	34
78	Construcción	45 días	lun 17/04/23	vie 02/06/23	77
79	Iteración 1	8 días	lun 17/04/23	mar 25/04/23	
92	Hito Fin Iteración 1	0 días	mar 25/04/23	mar 25/04/23	79
93	Iteración 2	7 días	mar 25/04/23	mar 02/05/23	92
106	Hito Fin Iteración 2	0 días	mar 02/05/23	mar 02/05/23	93
107	Iteración 3	13 días	mié 03/05/23	mar 16/05/23	106
120	Hito Fin Iteración 3	0 días	mar 16/05/23	mar 16/05/23	107
121	Iteración 4	17 días	mar 16/05/23	vie 02/06/23	120
134	Hito Fin Iteración 4	0 días	vie 02/06/23	vie 02/06/23	121
135	Hito Fin Fase de Construcción	0 días	vie 02/06/23	vie 02/06/23	78
136	Transición	16 días	vie 02/06/23	lun 19/06/23	135
137	Iteración 1	8 días	vie 02/06/23	dom 11/06/23	
148	Hito Fin Iteración 1	0 días	dom 11/06/23	dom 11/06/23	137
149	Iteración 2	6 días	dom 11/06/23	sáb 17/06/23	148
160	Hito Fin Iteración 2	0 días	sáb 17/06/23	sáb 17/06/23	149
161	Iteración 3	2 días	sáb 17/06/23	lun 19/06/23	160
166	Hito Fin Iteración 3	0 días	lun 19/06/23	lun 19/06/23	161
167	Hito Fin Fase de Transición	0 días	lun 19/06/23	lun 19/06/23	136
168	Fin del Proyecto	0 días	lun 19/06/23	lun 19/06/23	1

Figura 7: Planificación temporal

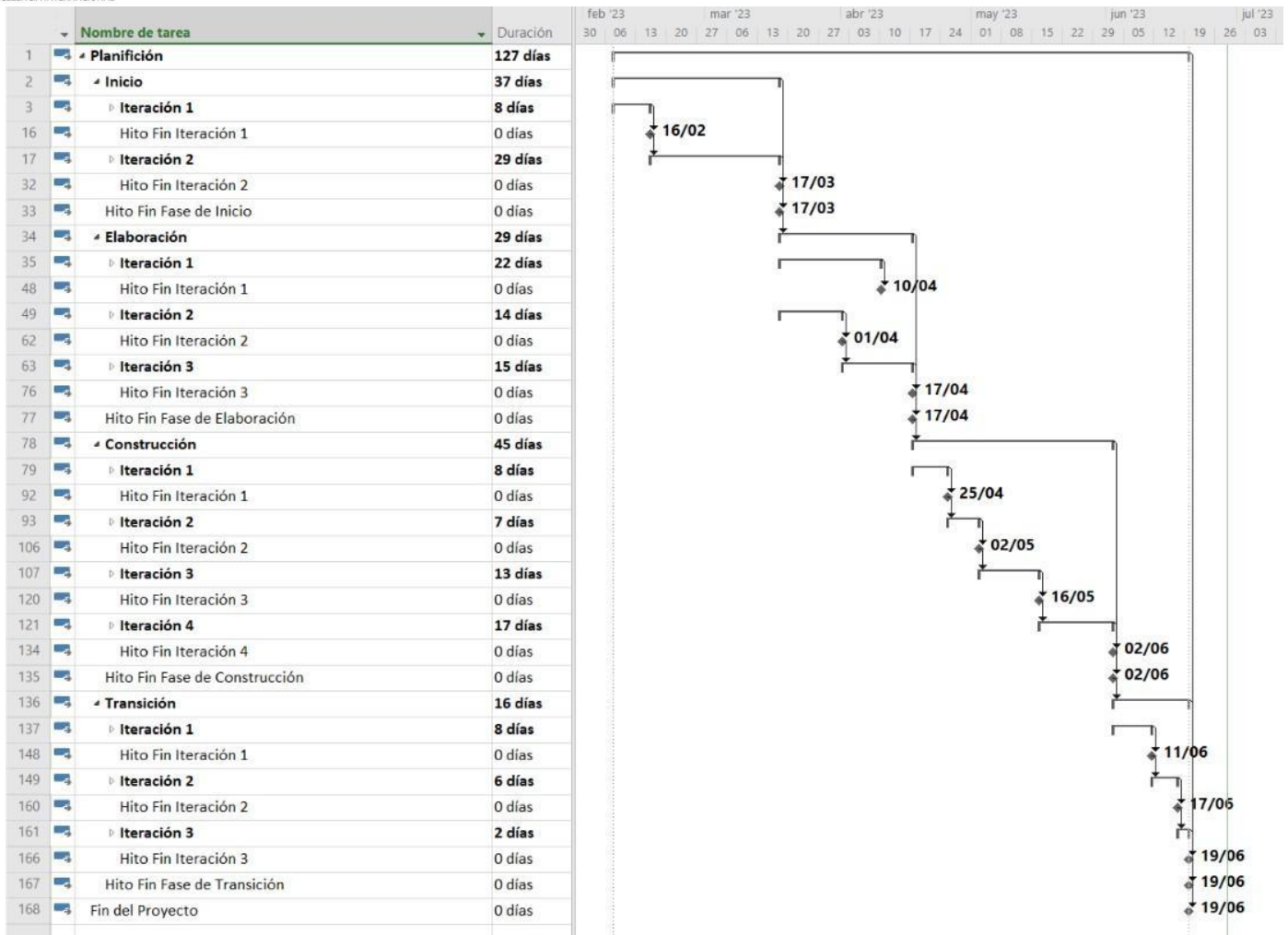


Figura 8: Diagrama de Gantt

5.4. Especificación de requisitos

En este apartado se detallarán los requisitos software del sistema que se va a desarrollar. Siguiendo la estructura del *Método Durán y Bernárdez* [19], se recogerán los siguientes datos:

- Participantes
- Objetivos del sistema
- Requisitos del sistema
 - Requisitos de información
 - Requisitos funcionales
 - Requisitos no funcionales
- Matriz de rastreabilidad

Si se desea conocer todo el proceso del desarrollo de los requisitos, puede consultar el *Anexo II: Especificación de requisitos software*.

5.4.1. Participantes

El proyecto cuenta con tres participantes, dos de ellos los tutores encargados de la supervisión del propio desarrollo, y uno la alumna autora del proyecto. Siendo estos:

- Cristina Alejandra Crespo Jiménez
- Gabriel Villarrubia González
- André Filipe Sales Mendes

La *Tabla 1* nos muestra un ejemplo de cómo se recogen los datos de los participantes.

Participante	Cristina Alejandra Crespo Jiménez
Organización	Universidad de Salamanca
Rol	Desarrolladora
Desarrollador	Sí
Cliente	No
Usuario	No
Comentarios	Ninguno

Tabla 1: Participante Cristina A. Crespo Jiménez

5.4.2. Objetivos del sistema

Los objetivos que debe cumplir el sistema son los siguientes:

- Gestión de usuarios
- Gestión de elementos encontrados
- Gestión de la información
- Gestión de las escenas

La *Tabla 2* mostrada a continuación es un ejemplo de cómo se recogen los datos de estos objetivos.

OBJ-002	Gestión de elementos encontrados
Versión	1.0
Autores	Cristina A. Crespo Jiménez
Descripción	El sistema deberá actualizar la base de datos cada vez que un usuario encuentre animales u objetos ocultos

Subobjetivos	Ninguno
Importancia	Alta
Urgencia	Alta
Estado	Validada
Estabilidad	Alta
Comentarios	Ninguno

Tabla 2: OBJ-002 Gestión de elementos encontrados

5.4.3. Requisitos del sistema

5.4.3.1. Requisitos de información

Estos requisitos nos indican la información que se debe almacenar del proyecto. Siendo los siguientes:

- Información sobre los usuarios
- Información sobre los animales descubiertos
- Información sobre los objetos ocultos descubiertos

La *Tabla 3* mostrada a continuación es un ejemplo de cómo se recogen los datos de estos requisitos de información.

IRQ-001	Información sobre los usuarios
Versión	1.0
Autores	Cristina A. Crespo Jiménez
Fuentes	
Objetivos asociados	<ul style="list-style-type: none"> ● OBJ-001 Gestión de usuarios ● OBJ-004 Gestión de las escenas
Requisitos asociados	<ul style="list-style-type: none"> ● IRQ-002 Información sobre los animales descubiertos ● IRQ-003 Información sobre los objetos ocultos descubiertos ● UC-001 Registro de usuarios ● UC-002 Login ● UC-003 Verificar datos ● UC-004 Comprobar datos ● UC-005 Cerrar sesión ● UC-020 Señalar con raycast ● UC-021 Cambio de color del raycast

	<ul style="list-style-type: none"> ● UC-022 Interactuar con el teclado 	
Descripción	El sistema deberá almacenar la información correspondiente a cada usuario registrado.	
Datos específicos	<ul style="list-style-type: none"> ● Email ● Contraseña 	
Tiempo de vida	Medio	Máximo
Ocurrencias simultáneas	Medio	Máximo
Importancia	Alta	
Urgencia	Alta	
Estado	Validada	
Estabilidad	Alta	
Comentarios	Ninguno	

Tabla 3: IRQ-001 Información sobre los usuarios

5.4.3.2. Requisitos funcionales

Estos requisitos definen cómo se comporta el sistema cuando los actores interactúan.

5.4.3.3. Diagrama de paquetes

El sistema se estructura en diferentes paquetes, lo primero de todo será realizar el diagrama que muestre cómo es esta jerarquía. En la *Figura 9* podemos observar que se conforma de *Gestión de usuarios*, *Gestión de la información*, *Gestión de elementos encontrados* y *Gestión de las escenas*.

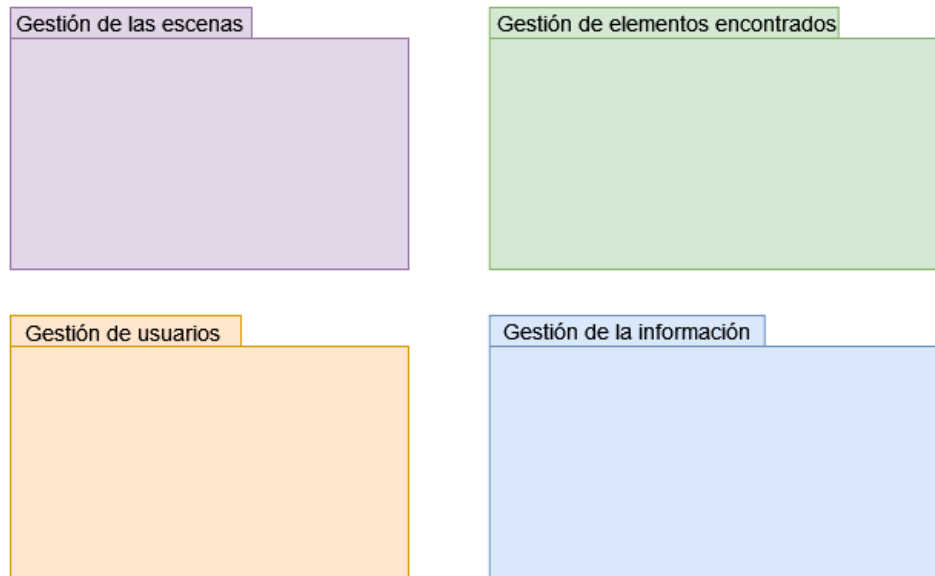


Figura 9: Diagrama de paquetes

5.4.3.4. Definición de actores

Lo segundo que haremos será detallar los actores que forman parte del sistema. En la *Figura 10* podemos observar que son *Usuario*, *Usuario anónimo*, *Usuario registrado* y *Sistema*. Y en la *Tabla 4*, un ejemplo de las tablas donde se recoge la especificación de cada actor.

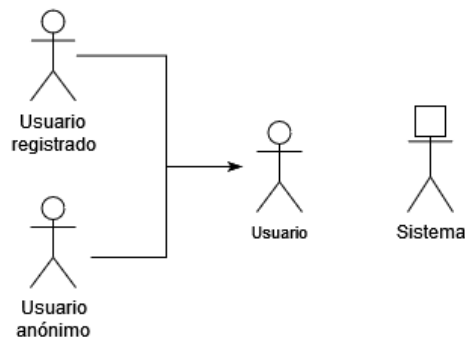


Figura 10: Actores

ACT-001	Usuario anónimo
Versión	1.0
Autor	Cristina A. Crespo Jiménez
Fuentes	
Descripción	Este actor representa a cualquier usuario que accede al sistema y puede que todavía no se haya registrado.

Comentarios	Ninguno
-------------	---------

Tabla 4: ACT-001 Usuario anónimo

5.4.3.5. Casos de uso del sistema

Lo tercero es definir los escenarios en los que se desarrollan e interactúan los actores predefinidos realizando diagramas de casos de uso como el de la *Figura 11*. En tablas, como la mostrada en la *Tabla 5*, se van especificando los pasos que sigue cada caso de uso para su desarrollo.

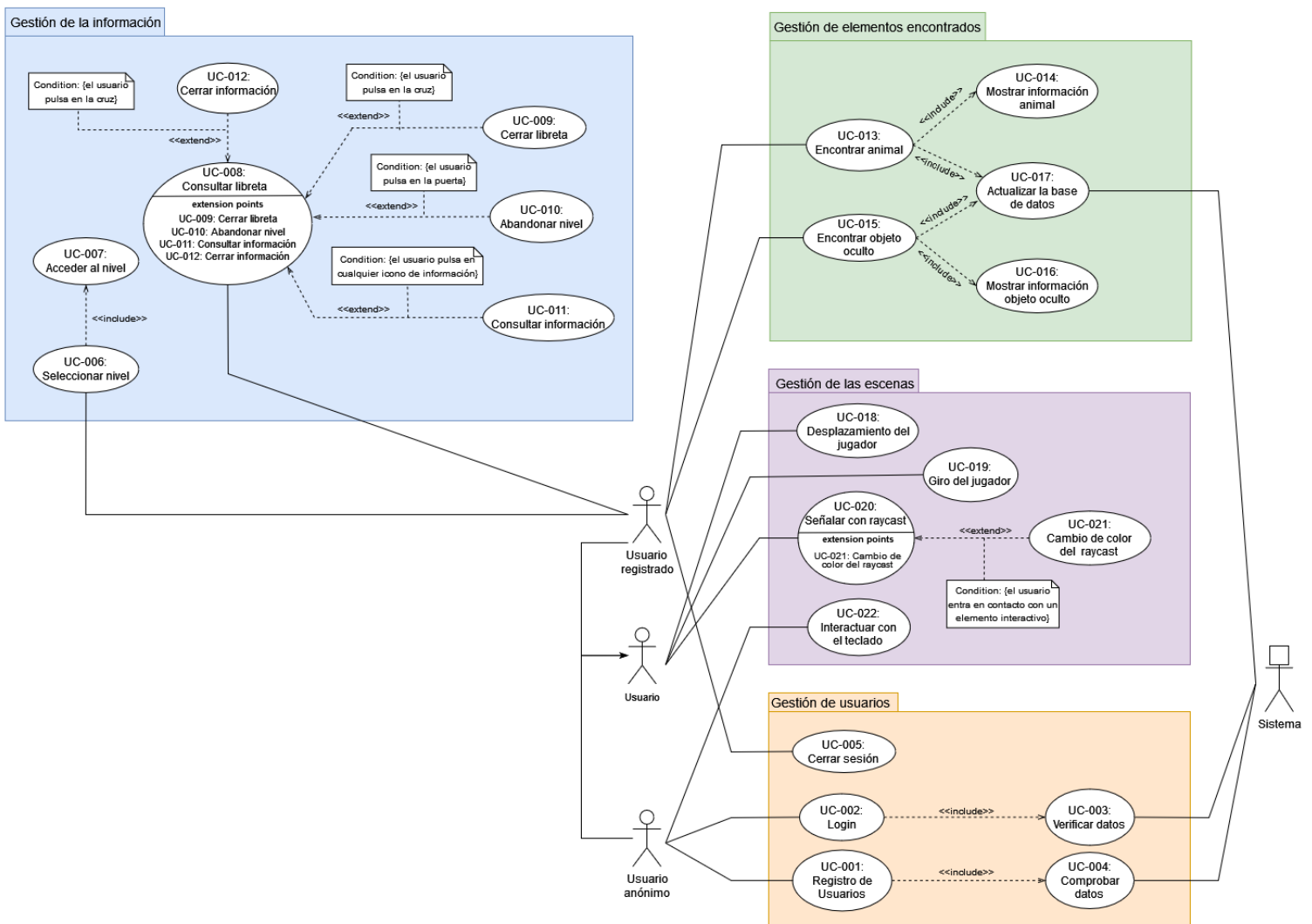


Figura 11: Diagrama de casos de uso



UC-008	Consultar libreta	
Versión	1.0	
Autor	Cristina A. Crespo Jiménez	
Fuentes		
Objetivos asociados	<ul style="list-style-type: none"> ● OBJ-003 Gestión de la información ● OBJ-004 Gestión de las escenas 	
Requisitos asociados	<ul style="list-style-type: none"> ● IRQ-002 Información sobre los animales descubiertos ● IRQ-003 Información sobre los objetos ocultos descubiertos ● UC-002 Login ● UC-006 Seleccionar nivel ● UC-008 Cerrar libreta ● UC-010 Abandonar nivel ● UC-011 Consultar información ● UC-012 Cerrar información ● UC-018 Desplazamiento del jugador ● UC-019 Giro del jugador ● UC-020 Señalar con raycast ● UC-021 Cambio de color del raycast 	
Descripción	El sistema mostrará al usuario la libreta.	
Precondición	El usuario debe haber realizado login.	
Secuencia normal	Paso	
	1	El actor usuario registrado <i>ACT-002</i> solicita consultar la libreta.
	2	El sistema mostrará la libreta.
Postcondición	Al usuario se le mostrará la libreta.	
Excepciones	Paso	
Rendimiento	Paso	
Frecuencia		
Importancia	Alta	
Urgencia	Alta	

5.5.1. Modelo de dominio

Durante esta fase es fundamental conceptualizar adecuadamente los objetos del sistema y sus relaciones, lo que permitirá determinar las entidades del modelo de negocio basándose en el proceso realizado hasta el momento. En la *Figura 13* podemos observar el diagrama de clases.

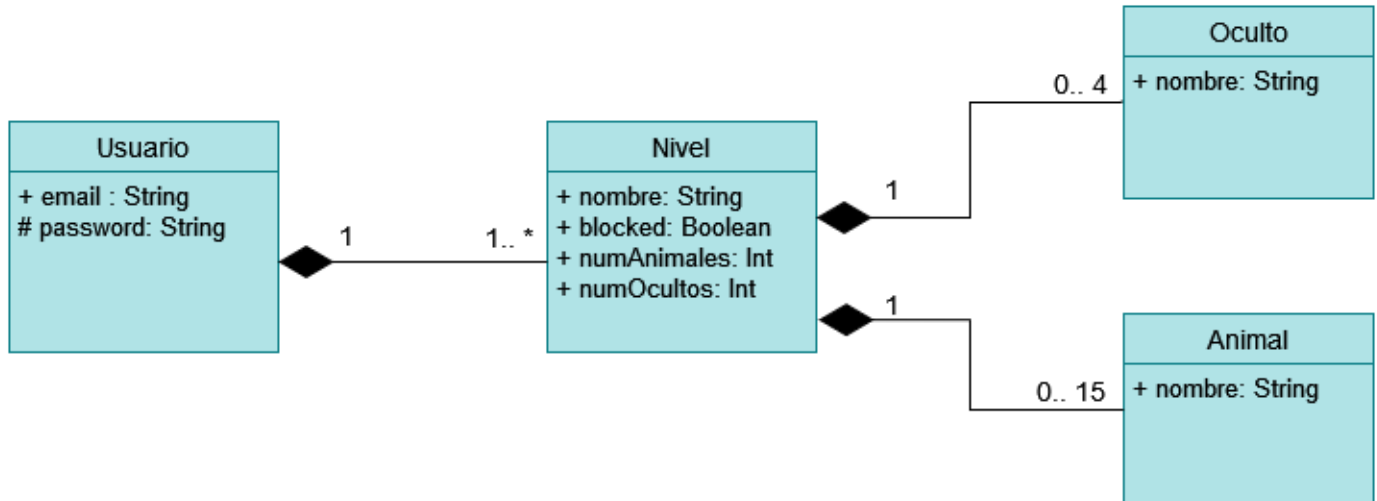


Figura 13: Diagrama de clases

5.5.2. Realización de casos de uso - Análisis

Los diagramas de secuencia, sacados de los casos de uso, permiten representar las interacciones entre los objetos del sistema mediante el intercambio de mensajes de envío y respuesta, ayudando así a definir las relaciones entre ellos. En la *Figura 14* podemos ver un ejemplo de estos diagramas.

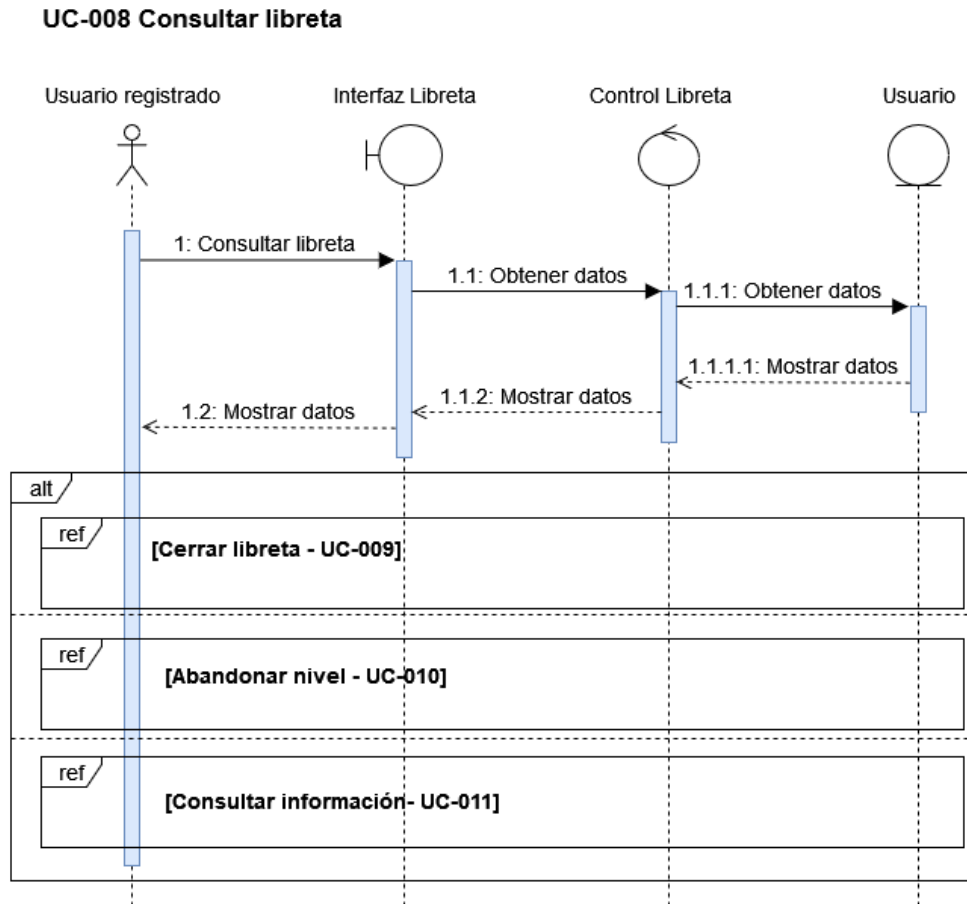


Figura 14: Diagrama de secuencia UC-008

5.5.3. Clases de análisis

Para poder visualizar la comunicación y distribución de las clases de análisis que conforman los diagramas de secuencia, se crean los diagramas de comunicación organizados en paquetes, los mismos en los que se divide el sistema. Un ejemplo de estos diagramas lo podemos ver a continuación en la *Figura 15*.

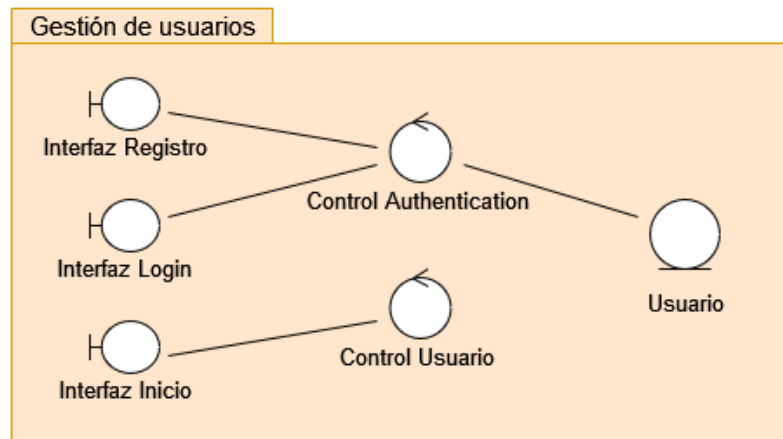


Figura 15: Diagrama de comunicación, Gestión de Usuarios

5.5.4. Vista arquitectónica modelo de análisis

Para concluir esta parte del análisis, obtenemos la vista arquitectónica inicial del sistema siguiendo el modelo de *Modelo-Vista-Controlador*.

5.6. Diseño del sistema

Esta fase se enfoca en el dominio de la solución, lo cual implica una aproximación cercana a la implementación. Los nombres de las clases, los métodos y los atributos presentados a continuación reflejarán una visión próxima al resultado final del sistema [20]. Se procederá al refinamiento de las especificaciones realizadas en etapas anteriores. Si desea conocer todo el proceso de diseño, puede consultar el *Anexo IV: Diseño del sistema*.

5.6.1. Patrones arquitectónicos

Elegir los patrones correctos es crucial ya que implica considerar diversos aspectos para su elección, como es el número y la estructura de los paquetes que conforman el sistema así como las relaciones entre ellos [21].

5.6.1.1. Patrón MVC

El patrón Modelo-Vista-Controlador (MVC) divide el trabajo en tres partes, podemos visualizarlo mejor en la *Figura 16*, por una parte están los datos de la aplicación, por otra la interfaz de usuario, y por último la lógica de control:

- Modelo, encapsula los datos y la funcionalidad central de forma independiente a la entrada y salida.
- Vista, presenta la información al usuario obteniendo los datos del modelo.
- Controlador, sirve peticiones del modelo o de la vista, propagándose al sistema a través de los controladores.

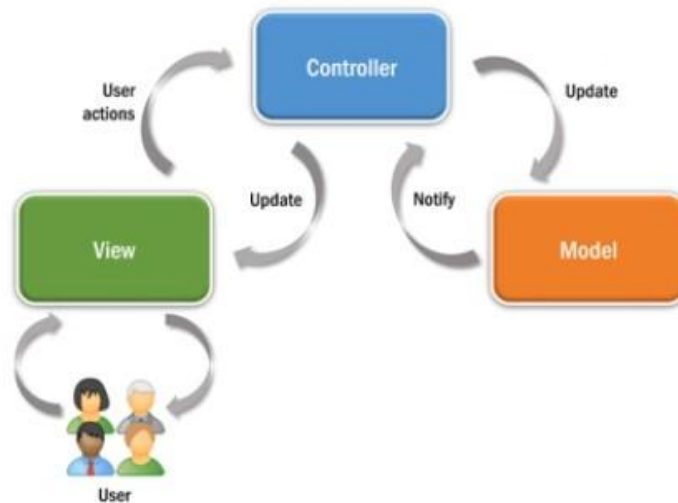


Figura 16: Esquema patrón MVC [22]

Es importante destacar que en Unity, las referencias de las clases MVC están dispersas en todo el código, por lo que es necesario tener un objeto de referencia raíz que funcione como contenedor de los datos relacionados. A mayores, se emplean componentes para acciones específicas que no notifican ni almacenan información.

5.6.1.2. Patrón Entidad-Componente (EC)

Se emplea principalmente en el desarrollo del videojuego. Basado en el principio de composición sobre herencia, crea una jerarquía de elementos formando entidades (GameObject), que a su vez cada una está compuesta por uno o más componentes (Rigidbody, Collider, Scripts...) que añaden la funcionalidad.

Este patrón resuelve el problema de herencia múltiple, que considera la situación de la *Figura 17*. Supongamos que HijaA e HijaB quieren modificar el mismo dato en Madre, eso podría causar un conflicto. Al segmentar las funcionalidades de las entidades y reutilizar componentes, evitamos que esto pase. Pero si empleáramos un gran número de componentes, podría dificultar la comprensión del proyecto, para abordar esto, se utiliza el patrón MVC.

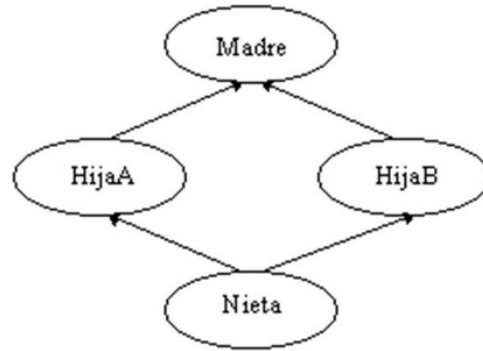


Figura 17: Problema herencia múltiple (Castro, R.) [23]

5.6.2. Subsistemas de diseño

A continuación, en la *Figura 18*, se procederá a descomponer el sistema en paquetes y subpaquetes para que sean más sencillos de manejar.

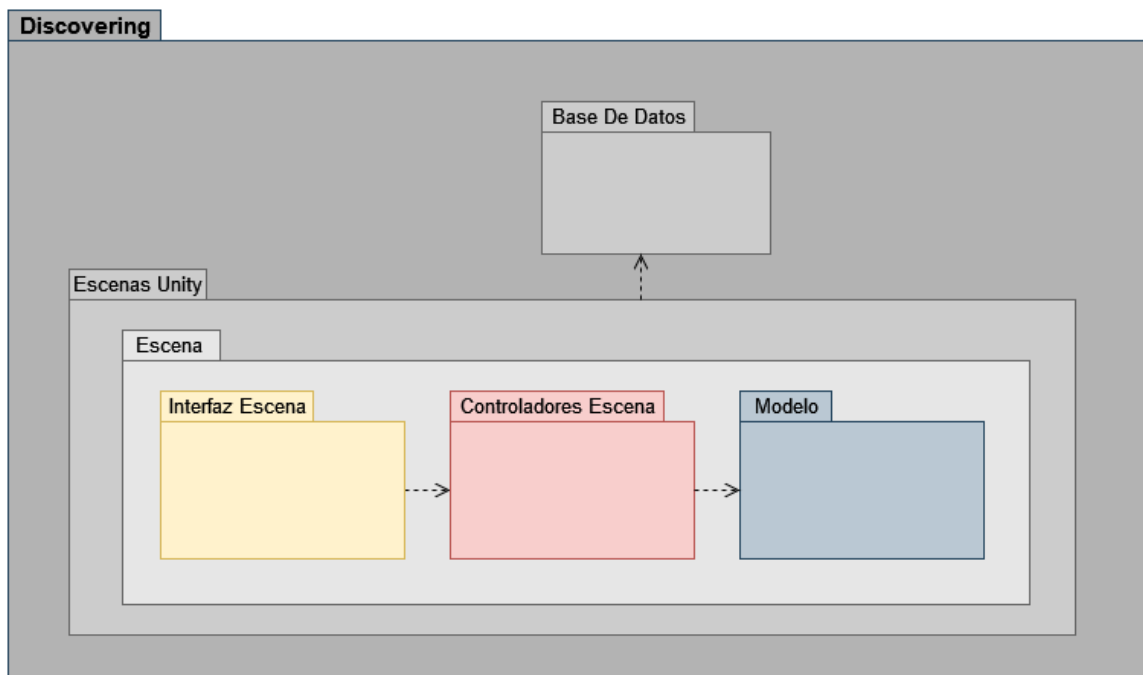


Figura 18: Subsistema de diseño

Discovering hace referencia al videojuego en sí que se genera con Unity. Dentro se encuentran dos paquetes:

- Escenas Unity, donde se efectúa la representación general del contenido de cada escena que conforma la aplicación del MVC adaptado.
- Base De Datos, se ocupa de la comunicación con la base de datos para el guardado y la extracción de datos.

5.6.3. Clases de diseño

Debido a que Unity sigue el patrón MVC, en este apartado se van a mostrar las clases de diseño por escenas, en la *Figura 19* se muestra un ejemplo de una de las escenas, haciendo hincapié en la estructura de vista, controlador y modelo de cada escena.

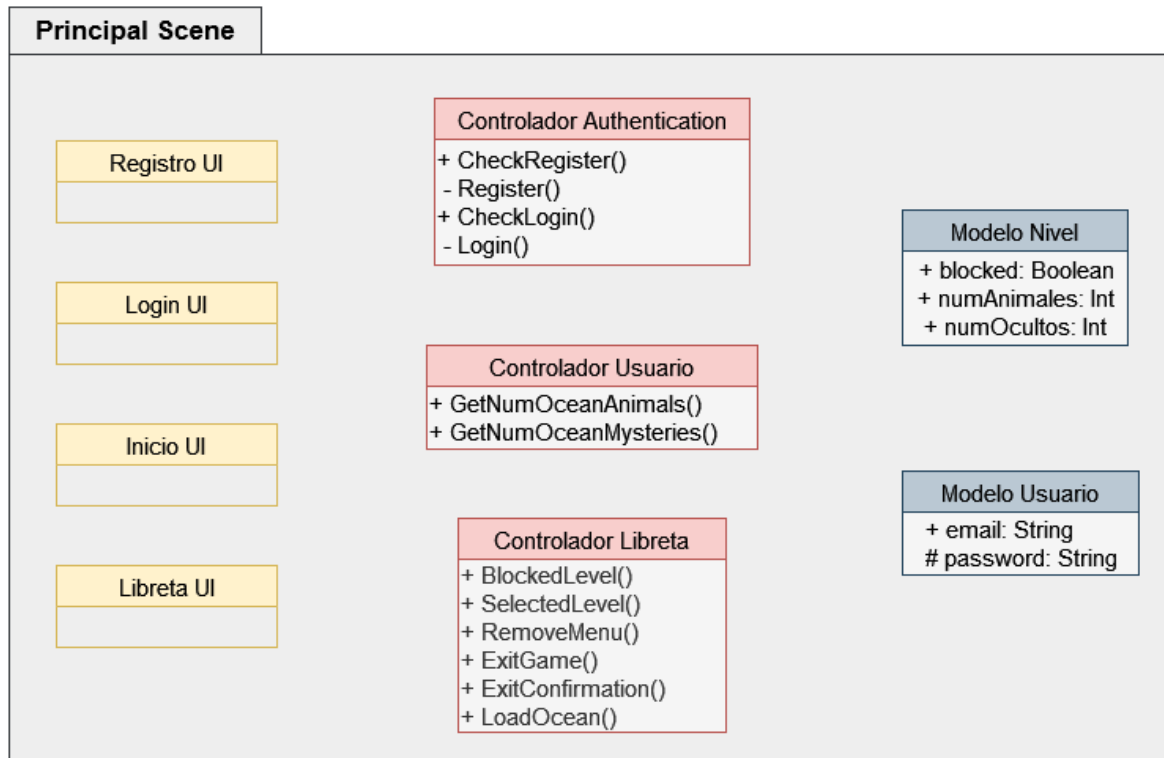


Figura 19: Clase de diseño - Principal Scene

5.6.4. Vista arquitectónica modelo de diseño

A continuación, podemos ver en la *Figura 20*, la vista arquitectónica del sistema que se ha compuesto basándose en el patrón MVC.

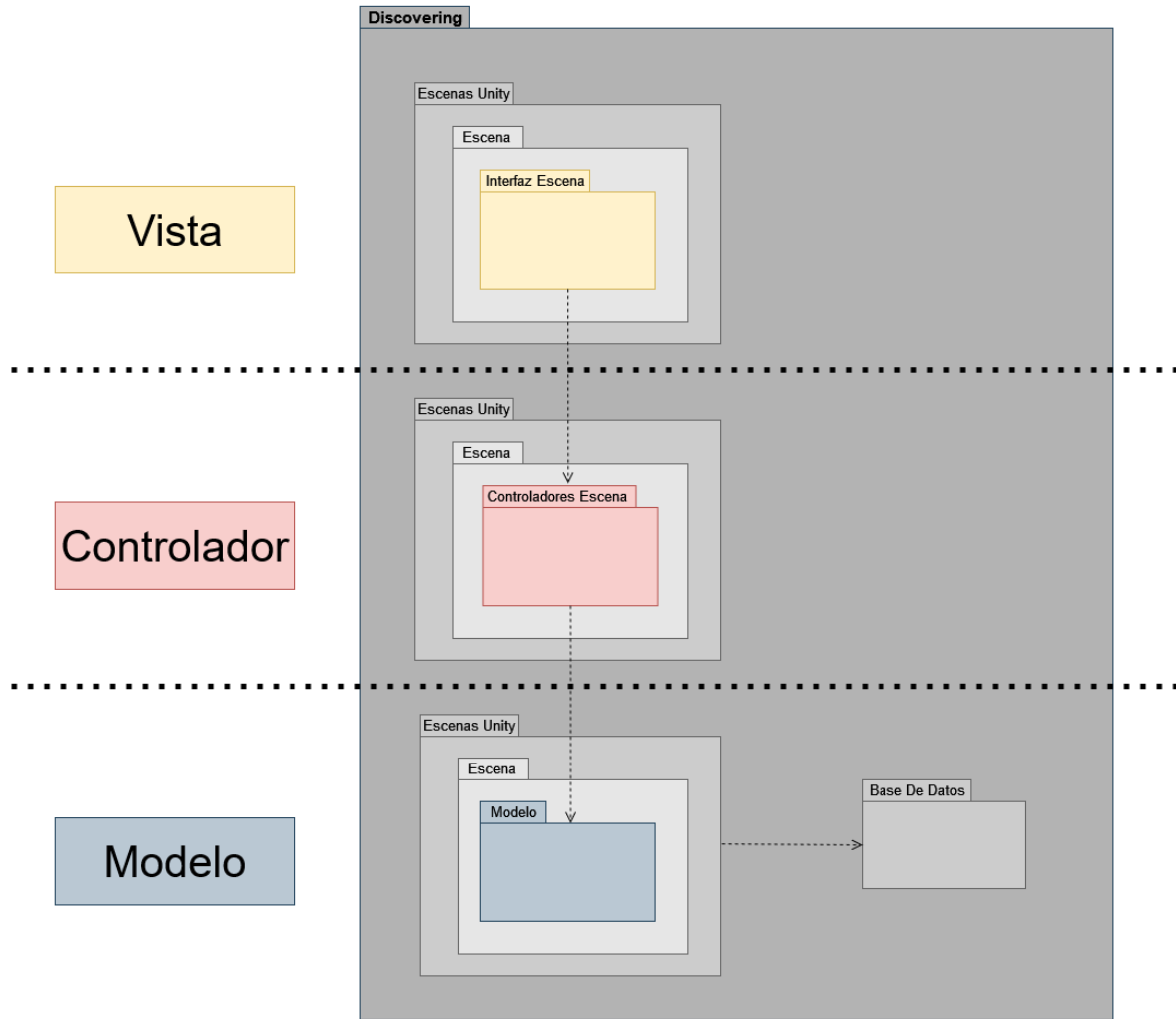


Figura 20: Diagrama de la base de datos

5.6.5. Realización de casos de uso - Diseño

Los diagramas de secuencia de diseño, sacados de los casos de uso, permiten representar las interacciones entre los objetos del sistema mediante el intercambio de mensajes de envío y respuesta, ayudando así a definir las relaciones entre ellos.

En la *Figura 21* podemos observar un ejemplo de estos diagramas.

UC-013 Encontrar animal

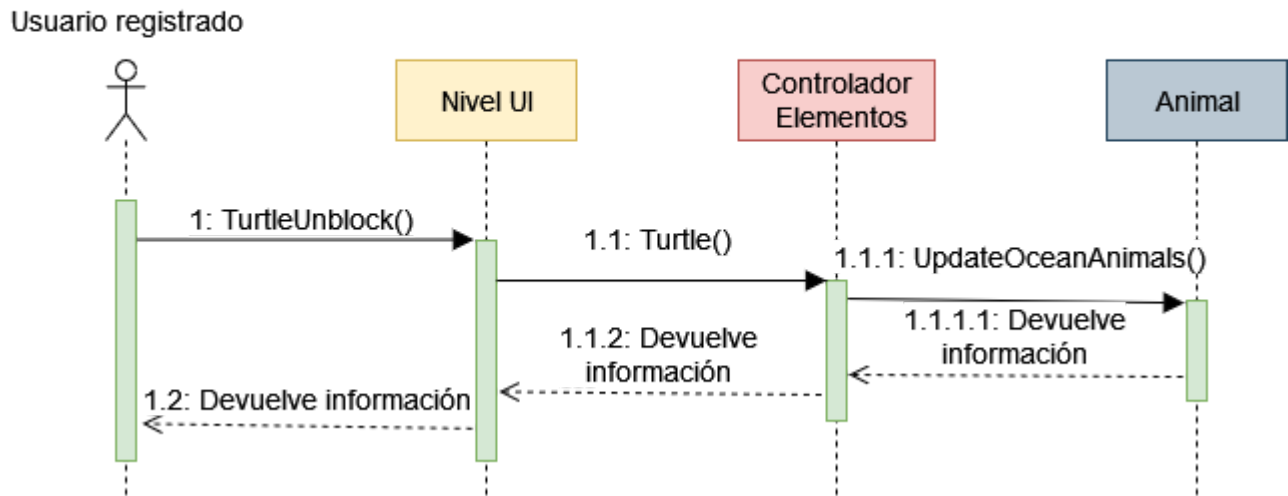


Figura 21: Diagrama de secuencia de diseño UC-013

5.6.6. Diseño de base de datos

La base de datos que se ha utilizado es la proporcionada por *Firebase*, una base de datos NoSQL que se ordena por colecciones.

En la siguiente *Figura 22* se muestra cómo es la base de datos creada.

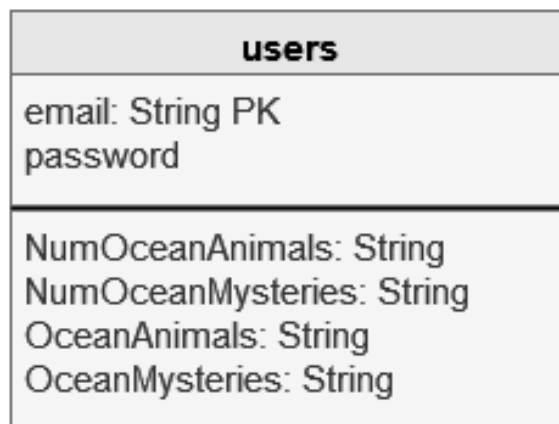


Figura 22: Diagrama de la base de datos

Se ha decidido poner todos los datos en una misma colección y no separados por animales y objetos ocultos puesto que cuanto más grande sea la colección más lento será el envío de datos, y los costes serán más grandes a medida que aumenta el envío de datos. *Firebase* proporciona un límite de uso de hasta 3000 usuarios activos por día.

5.6.7. Modelo de despliegue

A continuación, en la *Figura 23*, se detalla el diagrama de despliegue del sistema:

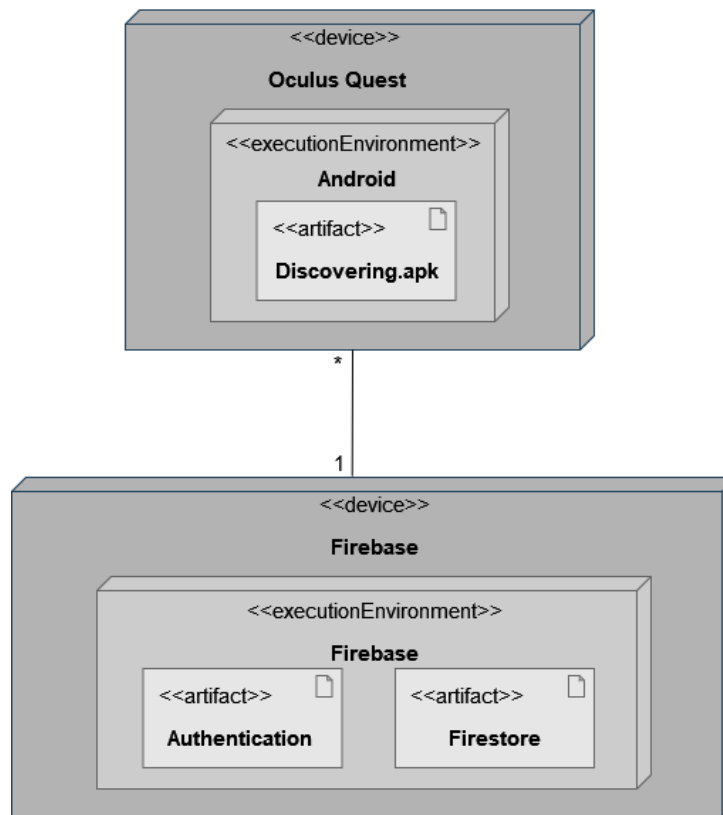


Figura 23: Diagrama de despliegue

Se observa que el sistema se compone de dos nodos:

- Oculus Quest, es el dispositivo empleado para la ejecución del *apk* en entorno Android. Será utilizado por los usuarios que ejecuten el videojuego.
- Firebase, es el servicio que nos ofrece autenticación y base de datos para el videojuego.

5.7. Implementación

En esta fase se ha realizado la codificación del sistema a partir de los resultados del apartado anterior, la fase de diseño. Se ha hecho uso de las herramientas especificadas en el apartado 4. *Técnicas y herramientas*. Si desea conocer con detalle la implementación, puede consultar el *Anexo V: Documentación técnica*.

Lo primero de todo ha sido familiarizarse con el entorno de Unity y el paquete de *Oculus Integration* para poder vincular las gafas Oculus al videojuego [24]. Una vez que se consiguió la vinculación correctamente, se continuó con el diseño de la primera escena. Todas las texturas y modelos empleados se han descargado de *Asset Store* y *Sketchfab* [25][26].

Se implementaron todas las funcionalidades correspondientes al movimiento del jugador, el raycast (es un rayo virtual con origen en los mandos y se usa para detectar colisiones, interactuar o seleccionar objetos de la escena), recoger objetos del entorno y las interacciones y funcionalidades con respecto a las interfaces de usuario creadas para el registro e inicio de sesión de usuarios así como la interacción con las libretas.

Cuando funcionaron correctamente todos los componentes, se continuó con el diseño de la segunda escena, se incorporaron los animales y su animación del movimiento por el entorno. Siendo el desarrollo de esta escena más rápido que la primera al tener todas las funcionalidades básicas funcionando correctamente.

Una vez implementadas las escenas y todas las funcionalidades, se vinculó con *Firestore*. Se implementaron las conexiones necesarias para que el paquete *Firestore Authentication* autentificase de manera adecuada los datos introducidos por el usuario, al igual que con el paquete *Firestore Cloud Firestore* para el correcto funcionamiento del guardado y de la extracción de datos.

5.8. Pruebas

Esta fase desempeña un papel crucial en el desarrollo de un software, ya que pueden usarse para verificar la implementación y garantizar que el sistema funciona correctamente y cumple con los objetivos establecidos. Es por eso que la carga de trabajo se reparte a lo largo del ciclo de vida del proyecto, siendo mayor y más significativo a la hora de implementar código.

Se han realizado dos tipos de pruebas:

- Unitarias, aquellas que se efectúan durante la implementación de un componente para averiguar si funciona o no correctamente.
- Globales, aquellas que se ejecutan al finalizar la implementación de un componente que se ha unido con el resto de componentes, para verificar el funcionamiento del sistema en su conjunto.

6. Funcionalidad del sistema

A continuación, se explicará la dinámica del juego y el funcionamiento de las diferentes interacciones que se pueden hacer dentro del mismo, de esta manera se facilita la total comprensión del juego desarrollado. Si desea conocer todo el proceso de diseño, puede consultar el *Anexo VI: Manual de usuario*.

Antes de comenzar nos tienen que quedar claros los controles de los mandos de las gafas Oculus Quest, se muestran en la *Figura 24*.

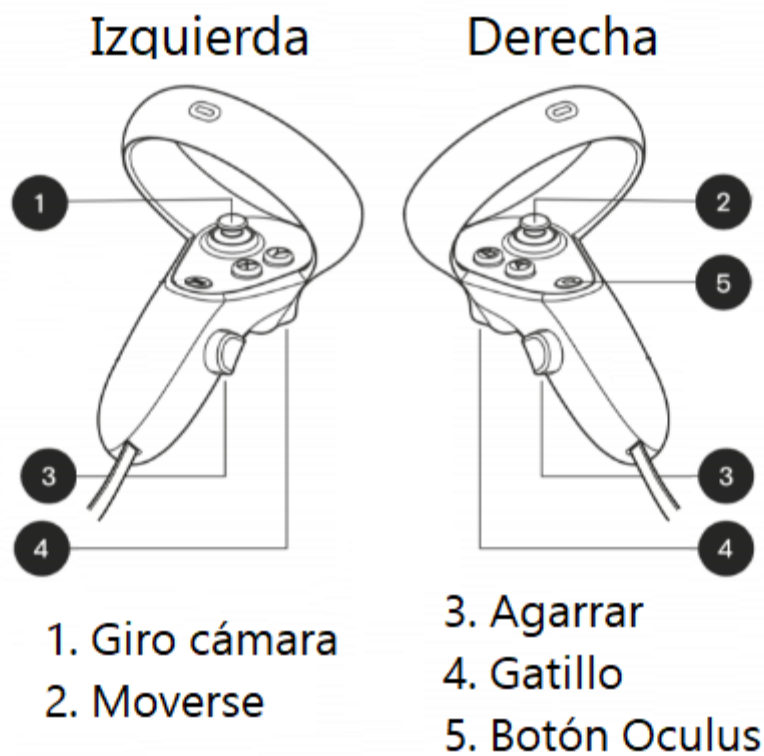


Figura 24: Controles mandos de las Oculus Quest [27]

Al cargar el videojuego se nos mostrará la pantalla de la *Figura 25*.



Figura 25: Pantalla inicial

En esta escena nos aparece una hoja que nos dice que para comenzar la aventura o bien creamos una cuenta o iniciamos sesión. Cuando movemos los mandos para señalar, podemos ver que de cada mando nos salen unos rayos (raycast) que dependiendo de donde apuntes son de un color u otro. Si el rayo se muestra de color rosa, significa que a lo que apuntes no tiene ningún tipo de interacción; y si es amarillo, significa que lo que señalas es interactivo.

Independientemente del botón que seleccionemos nos saldrá una pantalla para introducir nuestros datos. Si los introducimos mal, nos saldrá una pantalla de error; pero si los introducimos correctamente, nos saldrá una de éxito.

Una vez registrados/logueados, nos dirigimos a la cabaña para encontrarnos con una libreta que tendremos que agarrar, y al hacerlo, nos aparecerá una pantalla con la libreta abierta que nos mostrará los diferentes niveles bloqueados y/o disponibles, *Figura 26*.



Figura 26: Pantalla inicial

Al acceder al nivel del océano, podremos ver otra libreta encima de una mesa y otra abajo a la izquierda, pero esta es fija, se mueve con el jugador. Para poder consultar la libreta, lo podemos hacer de cualquiera de las dos formas. Al consultarla nos aparecerá el listado de animales y objetos ocultos que tenemos que buscar, *Figura 27*. Si en la parte inferior derecha de la imagen de los animales aparece una estrellita, significa que ese animal ya ha sido encontrado y puedes consultar la información del mismo cuando quieras. Si en el apartado de los objetos ocultos, hay alguno que no tiene la imagen del interrogante, significa que ha sido encontrado, y al igual que en los animales, puedes consultar su información cuando quieras.

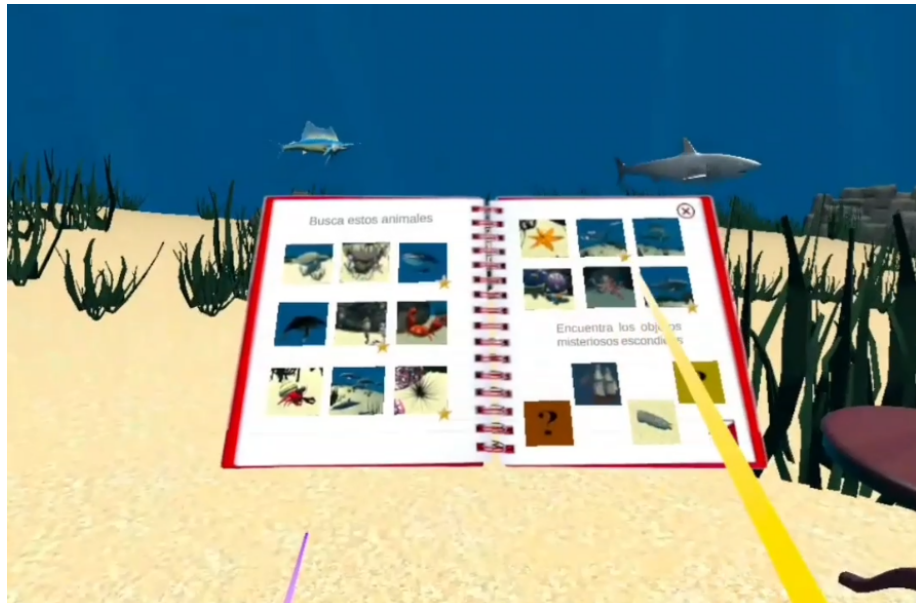


Figura 27: Pantalla océano

Cuando encontremos algún animal, apuntamos el raycast hacia él, en cuanto se vuelva amarillo, *Figura 28*, pulsamos el gatillo. En ese momento nos aparecerá en la pantalla la libreta con diez datos curiosos sobre el animal que hemos encontrado, *Figura 29*.

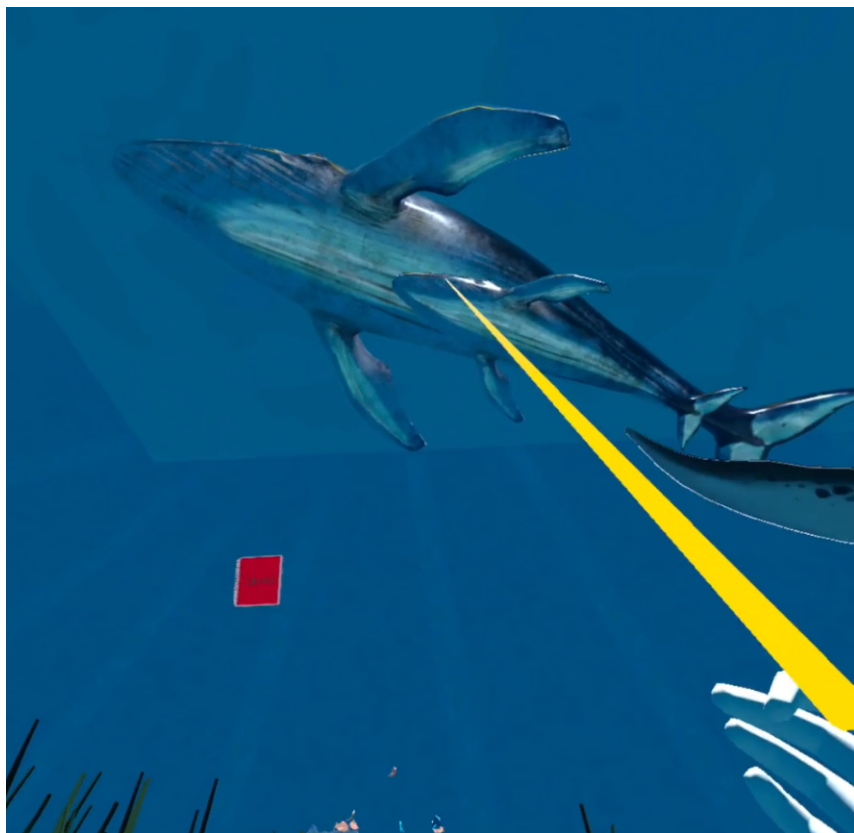


Figura 28: Pantalla océano - selección animal



Figura 29: Pantalla océano - información animal

Si se quiere cerrar la libreta valdría con seleccionar la flecha roja que aparece en la *Figura 29* abajo a la derecha, o seleccionar la cruz que aparece en la *Figura 27* arriba a la derecha.

Si se quiere salir del nivel o de la aplicación, tan sólo hay que seleccionar la puerta que aparece en la *Figura 27* en la esquina inferior derecha al consultar la libreta.

En la *Figura 30* podemos ver parte del escenario del océano.



Figura 30: Vista del escenario del océano

Se deja un enlace a continuación para poder visualizar el vídeo demostrativo del funcionamiento del videojuego: <https://vimeo.com/841439574?share=copy>



7. Limitaciones de la aplicación

Como todos las aplicaciones desarrolladas, tiene limitaciones, técnicas y partes que aún necesitan ser más refinadas. Algunas de ellas:

- A la hora de elegir los modelos, como se quiere que sean lo más realistas posibles, nos encontramos con una dificultad, ya que cuanto más renderizado, más pesa el modelo y más tarda en cargar el resto de la escena. Es por ello que se ha elegido usar modelos 2D o low poly (técnica de diseño gráfico que usa pocos polígonos para la creación de modelos 3D) en aquellos objetos secundarios, como son las algas marinas o el césped.
- Tampoco se han podido desarrollar más niveles debido a la dificultad de encontrar dichos modelos realistas de los diferentes animales.

8. Conclusiones y líneas de trabajo futuras

8.1. Conclusiones

Una vez finalizado el desarrollo del proyecto, se puede afirmar que se han conseguido cumplir todos los objetivos y requisitos establecidos al comienzo.

En referencia a los requisitos del sistema, se ha logrado:

- Diseñar e implementar un videojuego en realidad virtual funcional, donde el usuario puede moverse y girarse con total libertad por las escenas, al igual que interactuar con su entorno.
- Se ha conseguido que se gestione la verificación de los datos del usuario al registrarse o iniciar sesión, el guardado de datos de los usuarios registrados, así como actualizar los progresos realizados por ellos, animales y objetos ocultos encontrados.
- También se ha conseguido la visualización de los logros durante la ejecución del videojuego, de esta forma, los jugadores tendrán constancia en todo momento de los logros conseguidos y los que les quedan.

Acerca de los objetivos personales, se han alcanzado las metas propuestas y se ha logrado tanto la adquisición de nuevos conocimientos técnicos como el perfeccionamiento de los ya adquiridos, además del desarrollo de habilidades de gestión de proyectos. En consecuencia, estos avances han sido una fuente primordial de motivación e inspiración para lograr el éxito del proyecto.

8.2. Líneas de trabajo futuras

Como líneas de trabajo futuras, se proponen mejoras que añadan más funcionalidad y contenido interesante al videojuego. Como se ha dicho en el punto anterior, se han encontrado ciertas limitaciones, y si se consiguiera resolverlas añadiendo, por ejemplo, más niveles, o retos como resolver puzzles o cuentas matemáticas cuando se encuentran animales u objetos ocultos, o realizar un test al terminar un nivel para comprobar que se han adquirido conocimientos, se convertiría en un producto más valioso.

La idea de poder utilizarlo en las aulas de primaria como una herramienta educativa es muy prometedora, ya que los videojuegos pueden ser una forma efectiva y atractiva de enseñar y



aprender en el entorno educativo. Es importante considerar también aspectos como la accesibilidad y la usabilidad del juego para que pueda ser utilizado de manera efectiva en el entorno educativo.



9. Referencias bibliográficas

- [1] Garijo, C. (2020, 3 Diciembre).  *Historia de los VideoJuegos (1970-2020)*. “ON the AIR.” <https://carlosgarijo.substack.com/p/historia-de-los-videojuegos-1970>
- [2] Visual Capitalist. (2020). The rise of gaming visualized [Online]. In “ON the AIR”. <https://carlosgarijo.substack.com/p/historia-de-los-videojuegos-1970>
- [3] Carrasco, F. J. (2023, 3 Marzo). *Beneficios de la Realidad Virtual en la Educación*. Educa360. <https://educa360.com/blog/beneficios-de-la-realidad-virtual-en-la-educacion>
- [4] *¿Qué es un SDK?* (2020, 10 Junio). RedHat. <https://www.redhat.com/es/topics/cloud-native-apps/what-is-SDK>
- [5] Firebase, <https://firebase.google.com/?hl=es>
- [6] *Software - CASE Herramientas*. Tutorialspoint. https://www.tutorialspoint.com/es/software_engineering/case_tools_overview.htm
- [7] Framework: qué es, para qué sirve y algunos ejemplos. (2021, 19 Agosto). *Edix España*. <https://www.edix.com/es/instituto/framework/>
- [8] *¿Qué es C# en programación y para qué sirve?* (2020, 5 Mayo). Besoftware. <https://bsw.es/que-es-c/>
- [9] MasterD. *Qué es Unity y para qué sirve*. Master.d. <https://www.masterd.es/blog/que-es-unity-3d-tutorial>
- [10] Developer Oculus. *Oculus Integration | Integration | Unity Asset Store*. Asset Store. <https://assetstore.unity.com/packages/tools/integration/oculus-integration-82022>
- [11] *Firebase Authentication | Acceso multiplataforma simple y sin costo*. Firebase. <https://firebase.google.com/products/auth?hl=es>
- [12] *Firebase*. <https://console.firebase.google.com/u/0/project/discovering-8502c/overview>
- [13] *Cloud Firestore | Almacena y sincroniza los datos de tu app a escala global*. Firebase. <https://firebase.google.com/products/firestore?hl=es>
- [14] Fernández, Y. (2019, 30 Octubre). *Qué es Github y qué es lo que le ofrece a los desarrolladores*. Xataka.

<https://www.xataka.com/basics/que-github-que-que-le-ofrece-a-desarrolladores>

[15] Atlassian. *Sourcetree | Free Git GUI for Mac and Windows*. SourceTree.
<https://www.sourcetreeapp.com/>

[16] Flores, F. (2022, 22 Julio). *Qué es Visual Studio Code y qué ventajas ofrece*. OpenWebinars. <https://openwebinars.net/blog/que-es-visual-studio-code-y-que-ventajas-ofrece/>

[17] *EZEstimate Lite-Effort estimation tool*. Asignatura de Gestión de Proyectos.

[18] García Peñalvo, F., García Hogaldo, A., Vázquez Ingelmo, A. (2020). *Proceso Unificado*. Asignatura de Ingeniería de Software I.

[19] García Peñalvo, F., García Hogaldo, A., Vázquez Ingelmo, A. (2020). *Ingeniería de requisitos*. Asignatura de Ingeniería de Software I.

[20] Moreno García, M. (2020). *UML. Unified Modeling Language*. Asignatura de Ingeniería de Software II.

[21] Moreno García, M., García Peñalvo, F. (2020). *Patrones*. Asignatura de Ingeniería de Software II.

[22] Rodríguez-Aragón, J., Zato Domínguez, C. (2020). *Diseño*. Asignatura de Ingeniería de Software II.

[23] Castro, R. Problema de herencia múltiple. *Universidad Latina*.
<https://slideplayer.es/slide/1724738/>

[24] Casas, A. (2020, 5 Junio). *Nuestra primera aplicación con Unity y Oculus Quest*.
<https://angelcasas.es/2020/06/05/nuestra-primera-aplicacion-con-unity-y-oculus-quest/>

[25] *Unity Asset Store - The Best Assets for Game Making*. <https://assetstore.unity.com/>

[26] Sketchfab. <https://sketchfab.com/feed>

[27] VR Team Space (<https://www.vrteamspace.co.uk/setup/>). *Setup | VR TeamSpace*.
<https://mavink.com/post/837E2C25C5BF4723E10FF7B348B64BCDEDAM25DD2D/B85D75907F97492AF26E44E241D065EBD2AM8CE1D2>