

**ESCUELA POLITÉCNICA
SUPERIOR DE CÓRDOBA**
Universidad de Córdoba



TRABAJO FIN DE GRADO

Grado en Ingeniería Informática

**Desarrollo de aplicación móvil para divulgación de
astronomía mediante realidad aumentada**

Manual técnico

Autor: Carlos Checa Moreno

Director: Cristóbal Romero Morales

Septiembre de 2024



UNIVERSIDAD DE CÓRDOBA

ÍNDICE GENERAL

CAPÍTULO 1 INTRODUCCIÓN	7
CAPÍTULO 2 DEFINICIÓN DEL PROBLEMA	9
2.1 DEFINICIÓN DEL PROBLEMA REAL	9
2.2 DEFINICIÓN DEL PROBLEMA TÉCNICO	9
2.2.1 <i>Funcionamiento</i>	10
2.2.2 <i>Entorno</i>	10
2.2.3 <i>Esperanza de vida</i>	10
2.2.4 <i>Ciclo de mantenimiento</i>	11
2.2.5 <i>Calidad y Fiabilidad</i>	11
2.2.6 <i>Ciclo de desarrollo</i>	11
CAPÍTULO 3 OBJETIVOS.....	13
CAPÍTULO 4 ANTECEDENTES.....	14
4.1 ANTECEDENTES TEMÁTICOS	14
4.2 ANTECEDENTES CONCEPTUALES.....	15
CAPÍTULO 5 LIMITACIONES	16
5.1 FACTORES DATO.....	16
5.2 FACTORES ESTRATÉGICOS.....	17
CAPÍTULO 6 RECURSOS	18
6.1 RECURSOS HUMANOS.....	18
6.2 RECURSOS SOFTWARE	18
6.3 RECURSOS HARDWARE	20
CAPÍTULO 7 ESPECIFICACIÓN DE REQUISITOS.....	21

7.1	REQUISITOS FUNCIONALES.....	22
7.2	REQUISITOS NO FUNCIONALES.....	23
7.3	REQUISITOS DE LA INFORMACIÓN	23
CAPÍTULO 8 ANÁLISIS DEL SISTEMA.....		26
8.1	ACTORES	26
8.2	CASOS DE USO	26
8.3	DIAGRAMA DE CASOS DE USO	40
8.4	VALIDACIÓN DE CASOS DE USO	43
CAPÍTULO 9 DISEÑO DEL SISTEMA.....		45
9.1	DISEÑO DE MARCA.....	45
9.1.1	<i>Nombre.....</i>	45
9.1.2	<i>Paleta de Colores.....</i>	46
9.1.3	<i>Marca Gráfica</i>	46
9.2	DISEÑO DE INTERFAZ	47
9.3	USABILIDAD DEL SISTEMA.....	54
9.4	PROTOTIPADO	60
9.5	DISEÑO DE ARQUITECTURA.....	65
9.5.1	<i>Recursos</i>	66
9.5.2	<i>Entorno de Desarrollo</i>	69
9.5.3	<i>Clases.....</i>	79
9.5.4	<i>Servidor de Base de Datos</i>	92
9.5.5	<i>Stellarium API.....</i>	93
CAPÍTULO 10 PRUEBAS		98
10.1	PRUEBAS DE CAJA BLANCA	98
10.2	PRUEBA DE CAJA NEGRA	99
10.2.1	<i>Caso de Prueba 1. Registro e inicio de sesión.....</i>	100
10.2.2	<i>Caso de Prueba 2. Visualización de astros.....</i>	101

10.2.3	<i>Caso de Prueba 3. Desbloqueo de logros</i>	<i>101</i>
10.2.4	<i>Caso de Prueba 4. Navegación entre menús</i>	<i>102</i>
10.2.5	<i>Caso de Prueba 5. Dimensiones adaptables</i>	<i>102</i>
10.2.6	<i>Caso de Prueba 6. Guardado y recuperación de datos I</i>	<i>103</i>
10.2.7	<i>Caso de Prueba 7. Guardado y recuperación de datos II</i>	<i>103</i>
10.2.8	<i>Caso de Prueba 8. Actualización en vivo</i>	<i>103</i>
10.2.9	<i>Caso de Prueba 9. Sonido</i>	<i>104</i>
10.3	PRUEBAS DE SEGURIDAD	104
10.3.1	<i>Caso de Prueba de Seguridad 1. SQL Injection</i>	<i>105</i>
CAPÍTULO 11 CONCLUSIONES		106
CAPÍTULO 12 FUTURAS MEJORAS		108
BIBLIOGRAFÍA		109

ÍNDICE DE FIGURAS

FIGURA 1: DIAGRAMA E-R	25
FIGURA 2: ACTOR USUARIO	26
FIGURA 3: COMPONENTES DIAGRAMAS DE CASOS DE USO	41
FIGURA 4: DIAGRAMA DE CASOS DE USO.....	42
FIGURA 5: PALETA DE COLORES	46
FIGURA 6: LOGOTIPO	46
FIGURA 7: ISOTIPO.....	46
FIGURA 8: LOGOTIPO E ISOTIPO.....	47
FIGURA 9: LOGOTIPO E ISOTIPO COLOR INVERTIDO	47
FIGURA 10: MAQUETA INICIAR SESIÓN Y REGISTRO	48
FIGURA 11: MAQUETA JUEGO	49
FIGURA 12: MAQUETA MENÚ DE OPCIONES	50
FIGURA 13: MAQUETA DE BIBLIOTECA PROGRESO, APRENDIZAJE Y INFORMACIÓN	51
FIGURA 14: MAQUETA DE UBICACIÓN Y LOGRO	52
FIGURA 15: MAQUETA DE TUTORIAL	53
FIGURA 16: MAQUETA DE AJUSTES	54
FIGURA 17: UNITY LOCALIZATION. LOCALE GENERATOR.....	56
FIGURA 18: TABLA DE CADENAS TUTORIAL	58
FIGURA 19: LOCALIZE STRING EVENT.....	59

FIGURA 20: INICIO DE APLICACIÓN.....	61
FIGURA 21: LOOK UP VISTAS PRINCIPALES	62
FIGURA 22: BIBLIOTECA E INFORMACIÓN	63
FIGURA 23: MENÚ UBICACIÓN	64
FIGURA 24: LOGRO CONSEGUIDO, AJUSTES Y TUTORIAL	65
FIGURA 25: ARQUITECTURA DEL DESARROLLO.....	66
FIGURA 26: ADOBE STOCK FONDOS	67
FIGURA 27: FLATICON ICONO LOGROS.....	67
FIGURA 28: ADOBE PHOTOSHOP PARA TRATAMIENTO DE IMÁGENES	68
FIGURA 29: TIPOS DE LOGROS	68
FIGURA 30: ELEMENTOS PRINCIPALES UNITY	70
FIGURA 31: CONTENTTYPE	71
FIGURA 32: INTERFAZ UNITY.....	72
FIGURA 33: ESCENA LOGIN	74
FIGURA 34: ESCENA REGISTER	75
FIGURA 35: ESCENA JUEGO	77
FIGURA 36: NGROK EN FUNCIONAMIENTO	93
FIGURA 37: PLUGIN CONTROL REMOTO STELLARIUM.....	94
FIGURA 38: EJEMPLO LLAMADA API PARA SOL	95
FIGURA 39: INFORMACIÓN UBICACIÓN ACTUAL STELLARIUM	96

ÍNDICE DE TABLAS

TABLA 1: TIPO DE ENTIDAD USUARIOS.....	25
TABLA 2: CASOS DE USO. EJEMPLO	27
TABLA 3: CASO DE USO. INICIO DE SESIÓN	28
TABLA 4: CASOS DE USO. CREACIÓN DE CUENTA	29
TABLA 5: CASOS DE USO. CIERRE DE SESIÓN	30
TABLA 6: CASOS DE USO. VISUALIZAR ASTROS	31
TABLA 7: CASOS DE USO. DESCUBRIR NUEVOS ASTROS	32
TABLA 8: CASOS DE USO. DESBLOQUEAR LOGROS	33
TABLA 9: CASOS DE USO. CAMBIAR UBICACIÓN DEL USUARIO.....	34
TABLA 10: CONSULTAR ASTROS DESCUBIERTOS	35
TABLA 11: CASOS DE USO. CONSULTAR INFORMACIÓN ADICIONAL	36
TABLA 12: CASOS DE USO. CONSULTAR LOGROS	37
TABLA 13: CASOS DE USO. AJUSTAR SISTEMA.....	38
TABLA 14: CASOS DE USO. VISUALIZAR MULTIMEDIA	39
TABLA 15: CASOS DE USO. VER TUTORIAL	40
TABLA 16: MATRIZ DE TRAZABILIDAD	44
TABLA 17: FORMATO TABLA DE LOCALIZACIÓN DE CADENAS	57

Capítulo 1

INTRODUCCIÓN

El ser humano lleva interesándose y estudiando el cielo desde las primeras civilizaciones de la historia. Este comportamiento ha sido motivado, no solo por la magnificencia y belleza del universo, sino por la necesidad de crear calendarios, orientarse, navegar e incluso por motivos religiosos.

Conocer las temporadas de lluvia o cuando serían las subidas de los ríos era un conocimiento de suma importancia para la supervivencia de los primeros pueblos, permitiendo una buena planificación de las actividades agrícolas. Por esta razón, las primeras civilizaciones de la historia, egipcia y mesopotámica [1], crearon una serie de calendarios y tablas donde medían ciclos astronómicos.

Más adelante, los griegos tomarían el relevo en el estudio del cielo. Partiendo de las observaciones realizadas por las anteriores civilizaciones buscarían una armonía matemática en el cosmos siendo precursores en la astronomía teórica.

Pasando por grandes científicos de la historia como Galileo Galilei, Nicolás Copérnico, Johannes Kepler e Isaac Newton, entre otros, que harían grandes avances en el área de la astronomía, llegaríamos hasta nuestros días.

Siendo la divulgación científica una tarea realmente importante para nuestra sociedad [2]. Una vez tenemos claro la importancia de la astronomía en la historia de la humanidad podemos concluir que es un conocimiento definitivamente digno de divulgar. Como plantea la Dra. Victoria Espinosa (2010) “La divulgación del conocimiento científico es una responsabilidad de todo aquel que

investiga, porque contribuye a la democratización del conocimiento, realimentar las desigualdades preexistentes o comunicar resultados a la comunidad formada por los especialistas en la materia”.

Con este proyecto se pretende realizar una aplicación móvil que mediante realidad aumentada te muestre todos los astros: estrellas, planetas, la luna... marcando también constelaciones que serían visibles desde tu localización de no ser por la contaminación.

Sin embargo, se procurará buscar la experiencia que tuvieron los astrónomos antaño y siguen teniendo los investigadores que trabajan estudiando el cosmos. Mostrar simplemente toda la información no está mal si lo que buscas es crear una enciclopedia interactiva, pero no cumple el requisito de brindar a los usuarios una experiencia comparable con la realidad. Con este propósito se le dará a la aplicación un enfoque de videojuego por el que el usuario deberá descubrir los diferentes elementos del universo interactuando con ellos a través de la realidad aumentada.

Se seguirá una filosofía de *learn by doing* (aprender haciendo) como promulgaba el Dr. John Dewey [3]. Mediante un aprendizaje basado en videojuegos se dinamizará la educación y se incrementará la motivación de los usuarios con logros o trofeos. Llevar a la práctica el hecho de aprender del cielo será indudablemente más fácil con esta aplicación que sin ella.

Capítulo 2

DEFINICIÓN DEL PROBLEMA

La necesidad de reconectar a las personas con el cosmos es evidente, especialmente en entornos urbanos donde la observación directa del cielo nocturno se encuentra severamente limitada por la contaminación lumínica. Esta desconexión no solo afecta nuestro aprecio por la belleza del universo, sino que también impacta nuestra comprensión y curiosidad por los fenómenos astronómicos, disminuyendo el interés general en la astronomía y las ciencias espaciales.

2.1 Definición del problema real

El problema real yace en la creciente barrera entre el ser humano y el vasto universo que nos rodea. La contaminación lumínica en las ciudades no solo ha robado a muchas personas la posibilidad de maravillarse ante la visión de la Vía Láctea, sino que también ha creado una generación desvinculada de los ritmos naturales y los eventos celestes que nuestros ancestros conocían bien. Esta desconexión cultural y educativa con el firmamento limita nuestra capacidad de entendimiento del lugar que ocupamos en el universo.

2.2 Definición del problema técnico

Desde un enfoque técnico, el desafío consiste en desarrollar una aplicación móvil que, empleando realidad aumentada, se muestren los cuerpos celestes que deberíamos ser capaces de ver, así como información adicional. Todo esto aplicando un sistema de progresión similar al de un videojuego para crear una experiencia más educativa y disfrutable.

2.2.1 Funcionamiento

El funcionamiento de la aplicación se puede resumir en las siguientes funcionalidades:

- Cuentas de usuario. Deben existir unas cuentas de usuario en las que poder guardar tu progreso personal.
- Mostrar astros. La aplicación será capaz de mostrar diferentes astros al apuntar al cielo con la cámara de nuestro móvil. También habrá datos adicionales con los que poder estudiar más a fondo los diferentes elementos del universo.
- Logros y progresión. La aplicación tendrá un sistema de logros o progresión que pueda servir como hoja de ruta para desenvolverse en la tarea de estudiar el cielo.
- Material educativo. La aplicación también ofrecerá la posibilidad de acceder a material educativo como vídeos sobre astronomía.

2.2.2 Entorno

Es esencial tener claro desde el principio a que plataformas estará dirigida a nuestra aplicación, en este caso estará destinada a Android.

Para su desarrollo se usará el entorno de desarrollo Unity. Este es concretamente un motor de videojuegos creado por Unity Technologies.

Unity nos ofrece el marco de trabajo AR Foundation para el desarrollo de realidad aumentada. Este paquete nos permite acceder a características de la realidad aumentada de tal forma que podamos introducirlas en los diferentes elementos de nuestro proyecto.

2.2.3 Esperanza de vida

La esperanza de vida de este proyecto es bastante grande puesto que se trata de una aplicación educativa sobre un tema que no está en constante cambio. En otras palabras, la invariabilidad de los datos que usa la aplicación la hace muy poco perecedera.

2.2.4 Ciclo de mantenimiento

La necesidad de mantenimiento de la aplicación vendrá dada por actualizaciones del sistema operativo, Android en este caso, y del AR SDK, el cual es el encargado de la realidad aumentada.

2.2.5 Calidad y Fiabilidad

La aplicación debe presentar un funcionamiento correcto y adecuado para ser usada en los dispositivos que se consideren que presentan unos requisitos mínimos. Para esta tarea la aplicación será probada y depurada de forma exhaustiva.

2.2.6 Ciclo de desarrollo

2.2.6.1 Preparación

Estudio de plataformas de desarrollo apropiadas para el desarrollo de aplicaciones móviles que usen realidad aumentada. También será necesario la recopilación datos astronómicos que usará la aplicación.

2.2.6.2 Análisis

En esta fase se profundará que requisitos debe cumplir nuestra aplicación teniendo en cuenta el material con el que se trabajará.

2.2.6.3 Diseño

Se realizarán prototipos de la aplicación simulando lo máximo, en la medida de lo posible, lo que se pretende que sea el producto final. De esta forma podremos refinar el diseño de la aplicación.

2.2.6.4 Implementación

Se trata de la fase en la que se realizará la codificación producto software haciendo uso de los datos y plataforma elegida en la fase de preparación, y ajustando la aplicación a los requisitos y diseño decididos en las anteriores fases de desarrollo.

2.2.6.5 Pruebas

Esta etapa consiste en llevar a cabo pruebas de forma exhaustiva al producto desarrollado en la fase de implementación. De tal forma que tengamos una aplicación final integra y carente de fallos.

2.2.6.6 Documentación

Como parte final del proyecto se llevará a cabo la memoria de este.

Capítulo 3

OBJETIVOS

El propósito fundamental de este proyecto es desarrollar una aplicación móvil basada en realidad aumentada que permita a los usuarios reconectar con el universo desde cualquier lugar, especialmente en entornos urbanos afectados por la contaminación lumínica. La aplicación no solo se limitará a mostrar astros y constelaciones invisibles a simple vista, sino que también transformará la observación del cielo en una experiencia interactiva y educativa, equiparable a un videojuego.

Se espera que esta iniciativa aumente el interés y la motivación hacia la astronomía y las ciencias del espacio, ofreciendo a los usuarios una manera dinámica y atractiva de aprender sobre el cosmos. Además, se integrará un sistema de logros y progresión personal junto a material educativo que incentivarán la exploración continua y el aprendizaje autodidacta.

Por último, como objetivos personales se espera:

- Ganar experiencia desarrollando una aplicación simulando lo máximo posible un escenario real de cualquier aplicación del mercado. Concretamente, una aplicación que necesite acceso a un servidor desde cualquier red.
- Aprender y adaptarme a una nueva tecnología para mí tal como es la realidad aumentada .
- Mantener un proyecto ordenado y bien documentado independientemente de la magnitud de este.
- Poner en uso todos los conocimientos posibles adquiridos durante la carrera.

Capítulo 4

ANTECEDENTES

Los antecedentes de este proyecto de desarrollo se pueden dividir entre las siguientes 2 categorías:

4.1 Antecedentes Temáticos

Por antecedentes temáticos se tratarán las aplicaciones enfocadas en la temática del universo y la astronomía.

En el mercado actual, existen diversas aplicaciones dedicadas a la exploración del cosmos, tales como *Star Walk 2*, *Solar Walk Lite: Planetario 3D* y *Stellarium - Mapa de Estrellas*, que utilizan mapas 3D para navegar por ellos. Sin embargo, no hacen uso de la realidad aumentada como se busca en este proyecto.

A diferencia de las aplicaciones comentadas, *SkyView* sí emplea realidad aumentada para integrar la observación de estrellas y planetas directamente en el entorno del usuario, proporcionando una experiencia más parecida a la que se busca en este trabajo.

No obstante, el enfoque de mi proyecto se distingue significativamente de estas propuestas existentes por su integración de elementos de videojuegos y un sistema de logros que no solo premia, sino que también orienta a los usuarios hacia objetivos específicos, fomentando el aprendizaje gradual y la superación de retos.

4.2 Antecedentes Conceptuales

En este punto se tratan productos que siguen el modelo de una aplicación que haga uso de la realidad aumentada para motivar a sus usuarios a manejar su producto de forma más interactiva.

Una gran referencia para este proyecto es el videojuego *Pokemon Go* que revolucionó la industria del entretenimiento móvil al combinar la geolocalización y la realidad aumentada para crear una experiencia de juego inmersiva y globalmente accesible. *Pokemon Go* no solo motivó a los usuarios a explorar su entorno físico en busca de criaturas virtuales, sino que también demostró el potencial de la realidad aumentada para fomentar la interacción social y el ejercicio físico, al tiempo que proporcionaba una experiencia de usuario envolvente y motivadora.

Siguiendo este modelo, nuestro proyecto aspira a aplicar conceptos similares a la astronomía. Al igual que *Pokemon Go* utiliza puntos de interés locales para motivar a los jugadores, nuestra aplicación utilizará eventos astronómicos, como eclipses, lluvias de meteoros y alineaciones planetarias, como incentivos para que los usuarios exploren y aprendan sobre el cosmos de manera activa y participativa.

Además, al igual que en *Pokemon Go*, donde los jugadores pueden acumular y evolucionar sus criaturas, en nuestra aplicación, los usuarios podrán desbloquear logros y obtener recompensas a medida que aprenden sobre diferentes astros y fenómenos astronómicos. Esto no solo enriquece la experiencia educativa, sino que también introduce un aspecto competitivo y lúdico que puede aumentar el compromiso y la retención de los usuarios.

Capítulo 5

LIMITACIONES

Este capítulo detalla los distintos factores y restricciones a considerar durante la ejecución del proyecto, diferenciando entre dos categorías principales:

- Factores dato: Estos elementos son intrínsecos al problema o son establecidos por el cliente y no están sujetos a cambio. Reflejan condiciones preexistentes o requisitos inamovibles que deben respetarse durante el desarrollo del proyecto.
- Factores estratégicos: Estos son aspectos del diseño que ofrecen múltiples opciones de acción. Es crucial evaluar cada opción cuidadosamente, ya que la elección de una u otra alternativa puede influir significativamente en el desempeño y resultados finales del producto.

5.1 Factores dato

Los factores dato que vienen impuestos en el desarrollo de nuestro sistema son:

- El tiempo asignado al desarrollo del TFG está limitado a 300 horas, correspondientes a 12 créditos ECTS. Esto impone una restricción en cuanto a la profundidad y alcance del proyecto que puedes realizar.
- El lenguaje de programación de desarrollo será C#, lenguaje característico de los *scripts*, ficheros de código, de Unity.

- Dado que es un TFG con presupuesto limitado, las herramientas y recursos utilizados serán principalmente de software libre o, como en el caso de Unity, bajo licencias educativas que minimicen los costos.

5.2 Factores estratégicos

- El entorno de desarrollo a usar será Unity, en combinación con Visual Studio, debido a su amplio uso y recomendación en proyectos que involucran realidad aumentada y desarrollo de videojuegos.
 - AR Foundation será el *framework*, esquema o marco de trabajo que ofrece una estructura base para elaborar un proyecto con objetivos específicos, utilizado para desarrollar la funcionalidad de realidad aumentada de la aplicación. Este framework es seleccionado por su compatibilidad tanto con Android como con iOS, permitiendo un desarrollo más eficiente y una mayor accesibilidad del producto final.
 - Para la elaboración de la documentación se usará Microsoft Word, aprovechando la licencia de Microsoft 365 proporcionada por la cuenta universitaria. Word es elegido por su facilidad de uso, amplia aceptación y experiencia personal utilizándolo.
 - XAMPP será implementado para alojar la base de datos en el ordenador de desarrollo, utilizando Apache y MySQL. Esta elección estratégica permite un entorno de desarrollo local robusto y eficiente, reduciendo la dependencia de servicios externos y mejorando la seguridad y control durante la fase de desarrollo.
 - Ngrok será utilizado para permitir el acceso a la base de datos desde otras redes. Esta herramienta es crucial para poder probar la aplicación debidamente desde nuestro dispositivo móvil en ubicaciones remotas.
-

Capítulo 6

RECURSOS

En este apartado se describen los recursos que estarán disponibles durante la realización de este Trabajo Fin de Grado, categorizados en recursos humanos, recursos software y recursos hardware.

6.1 Recursos Humanos

- Autor: Carlos Checa Moreno. Alumno de 4º de Grado de Ingeniería Informática con Mención en Computación.
- Director: Dr. Cristóbal Romero Morales. Profesor Titular de Universidad, Departamento de Informática y Análisis Numérico.

6.2 Recursos Software

- Sistemas Operativos: Se empleará Windows 11 para el desarrollo y Android para la implementación y pruebas de la aplicación. Windows 11 es el sistema operativo del ordenador de desarrollo. Android será crucial para probar la aplicación en el ambiente operativo en el que se usará.
- Navegador: Google Chrome será el navegador principal utilizado para la investigación y acceso a documentación en línea.
- Correo electrónico: Se utilizarán Gmail para comunicaciones personales y UCOWebMail.2v para comunicaciones oficiales con la universidad y el director del proyecto, facilitando el intercambio formal de documentos y feedback.

- Ofimática: Microsoft Office, a través de la suscripción de Microsoft 365 de la cuenta universitaria, será utilizado para toda la documentación del proyecto. Esto incluye la redacción del informe final, preparación de presentaciones para la defensa del TFG y la creación de cualquier material de apoyo necesario.
 - Servicio de almacenamiento y control de versiones: GitHub será utilizado para el manejo del código fuente del proyecto, así como para almacenar documentos y otros recursos relacionados.
 - Entorno de Desarrollo Integrado (IDE) y Herramientas de Realidad Aumentada:
 - Unity: Será el IDE principal para el desarrollo de la aplicación. Unity es ampliamente reconocido por su robusta plataforma de desarrollo para juegos y aplicaciones interactivas, incluyendo soporte integral para realidad aumentada.
 - AR Foundation: Este paquete de Unity permite la creación de experiencias de realidad aumentada que funcionan tanto en iOS como en Android. AR Foundation integra las capacidades de ARKit y ARCore, facilitando el desarrollo de aplicaciones de AR que pueden ejecutarse en una amplia gama de dispositivos móviles. Aunque en este caso nos enfocaremos en Android, tendremos la posibilidad de ampliar a dispositivos iOS en un futuro.
 - Servidor y Base de Datos: XAMPP será utilizado para alojar la base de datos en el ordenador de desarrollo, utilizando Apache y MySQL.
 - Acceso Remoto: Ngrok será utilizado para permitir el acceso a la base de datos desde otras redes, facilitando pruebas y demostraciones en tiempo real desde ubicaciones remotas.
 - Adobe Photoshop, Adobe Stock y Flaticon para tratamiento de imágenes y recopilación de ilustraciones e iconos para la construcción de la interfaz de usuario.
-

6.3 Recursos Hardware

Para el desarrollo de este proyecto, se utilizarán dos equipos informáticos principales que constituyen los entornos de desarrollo donde se llevarán a cabo todas las fases de diseño, implementación y prueba del software:

- Ordenador de sobremesa Epical-Q Nighthawk
 - Procesador Intel Core i7 11700F.
 - Disco duro Disco SSD NVME 1 TB HP EX900Plus PCIe 3x4 PLUS 3.300MBs/2.700MBs.
 - Memoria HP V6 16 GB 3200MHz DDR4.
 - Gráfica NVIDIA RTX 3060 12 GB Dual ASUS.
- Ordenador portátil HP Pavilion Power Laptop 15-cb0xx:
 - Procesador Intel Core i7-7700HQ.
 - Disco duro Disco SSD Kingston 960 GB.
 - Memoria 8 GB DDR4-2400 SDRAM.
 - Gráfica NVIDIA Geforce GTX 1050.

El ordenador de sobremesa será el principal entorno de desarrollo debido a sus altas especificaciones técnicas y el portátil será utilizado como un entorno de desarrollo secundario, proporcionando flexibilidad para continuar trabajando cuando sea necesario trasladarse a diferentes ubicaciones.

Puesto que este proyecto se basa en la creación de una aplicación móvil también será necesario el uso de uno de estos dispositivos para poder probar el producto de forma adecuada.

- Teléfono móvil Xiaomi Mi 9 Lite:
 - Procesador: Octa-core Max 2,2 GHz.
 - RAM: 6,00 GB.
 - Almacenamiento: 64 GB.

Capítulo 7

ESPECIFICACIÓN DE REQUISITOS

En esta sección se procederá a detallar técnicamente lo que el sistema a desarrollar deberá realizar. La especificación de requisitos se basa en la información recopilada durante el análisis y es una fase crucial para el desarrollo del producto software, ya que definirá en gran medida el resultado final del producto. El objetivo es definir claramente la información que el software manipulará y las operaciones que realizará.

Dividiré los requisitos en tres categorías principales:

- **Requisitos Funcionales:** Estos describen las interacciones específicas entre el sistema y su entorno, detallando las funciones que el software debe ejecutar. Por ejemplo, acciones que el usuario espera que el sistema realice en respuesta a entradas específicas, como la creación de cuentas de usuario, la visualización de astros y la gestión de logros.
- **Requisitos No Funcionales:** Estos requisitos especifican criterios que pueden ser usados para juzgar la operación del sistema, en lugar de comportamientos específicos. Incluyen atributos como la seguridad, la usabilidad, la accesibilidad, y el rendimiento, entre otros. Estos aspectos son esenciales para asegurar que el software funcione de manera eficiente, segura y accesible para todos los usuarios.
- **Requisitos de la Información:** Se refieren a los tipos de información que el software debe manejar, cómo se almacena, accede y actualiza. Esto incluye la gestión de datos de los astros, los datos personales de los usuarios, y el seguimiento de los logros alcanzados por los usuarios dentro de la aplicación.

Para la elaboración de esta especificación, seguimos las recomendaciones del estándar IEEE 830 [4], que proporciona un marco detallado para la redacción de requisitos de software. Este estándar es ampliamente reconocido por promover la claridad, la completa definición y la verificabilidad de los requisitos, elementos esenciales para el éxito del desarrollo del software.

7.1 Requisitos Funcionales

- RF1 - El usuario podrá crear una cuenta personal donde se guardarán su progreso y logros.
 - RF2 - El usuario podrá iniciar sesión con su cuenta personal ya registrada en el sistema.
 - RF3 - El usuario podrá cerrar sesión.
 - RF4 - El usuario podrá visualizar astros como estrellas, planetas y constelaciones a través de la cámara de su dispositivo móvil,
 - RF5 – El usuario podrá interactuar con los astros mediante toques en la pantalla.
 - RF6 - El usuario podrá cambiar su ubicación.
 - RF7 - El sistema guardará el progreso del usuario siempre que este descubra un nuevo cuerpo celeste o consiga un logro.
 - RF8 - El sistema contará con un sistema de logros que reaccionará al progreso del usuario en el estudio del cielo.
 - RF9 – El sistema ofrecerá al usuario una recopilación de su progreso, tanto en logros como descubrimientos.
 - RF10 - El sistema proporcionará información detallada sobre los astros u elementos que muestre como historia y datos astronómicos.
 - RF11 - El sistema mostrará astros de acuerdo con datos reales de los cuerpos respecto a la ubicación del usuario.
 - RF12 - El sistema ofrecerá opciones de personalización al usuario.
 - RF13 - El sistema ofrecerá material multimedia al que el usuario podrá acceder.
 - RF14 - El usuario podrá ver un tutorial sobre como usar la aplicación dentro del sistema.
-

7.2 Requisitos No Funcionales

- RNF1 - La interfaz de usuario será intuitiva y fácil de usar para garantizar una experiencia satisfactoria para todos los usuarios.
- RNF2 - La aplicación funcionará fluidamente en dispositivos compatibles, con tiempos de carga y respuesta rápidos.
- RNF3 - La aplicación garantizará la protección de los datos personales de los usuarios conforme a las normativas de protección de datos.
- RNF4 - La arquitectura de la aplicación permitirá futuras expansiones y modificaciones sin degradar el rendimiento.

7.3 Requisitos de la Información

En primer lugar, se trabajará con archivos **json** [5] en los cuales se almacenará el progreso del usuario puesto que estos son fácilmente manipulables en Unity.

En estos archivos de guardado se tendrán 2 *arrays*: “astrosDescubiertos” y “listaLogros”. El primero estará compuesto por cada uno de los astros con los que haya interactuado el usuario, almacenando su nombre, altitud, azimuth, distancia, localización y fecha. Cabe destacar que la fecha y hora es un dato esencial para que el usuario pueda constatar que la aplicación está ofreciendo información real.

Por otra parte, los objetos que almacenará “listaLogros” serán todos los logros disponibles que pueda conseguir o ya haya conseguido el jugador. Cada logro tiene los campos nombre, conseguido y progreso. A diferencia de los astros descubiertos, los cuales se irán añadiendo medida que el jugador avance, todos los logros estarán presentes desde el principio ya que está presente el campo “conseguido”.

Se ha decidido realizar de esta manera puesto que es necesario guardar también el progreso de los logros. En contraposición, los astros solo se pueden encontrar en dos estados: descubiertos o no.

Por esto mismo simplemente se irán añadiendo al archivo de guardado a media que se vayan descubriendo.

Así pues, se presenta un ejemplo de la estructura básica que tendrá el json, junto a los tipos de cada campo:

```
{
  "astrosDescubiertos ": [
    {
      "name": "",
      "altitude": 0,
      "azimuth": 0,
      "distance": 0,
      "location": "",
      "date": "dd/mm/aaaa hh:mm:ss"
    },
  ],
  "listaLogros": [
    {
      "name": "",
      "conseguido": true/false,
      "progreso": 0
    },
  ]
}
```

Estos ficheros se almacenarán en la base de datos junto a sus correspondientes correos electrónicos y contraseñas debidamente cifradas.

Una vez explicado el tratamiento que tendrá la información de progreso del jugador se continuará con la definición de la **base de datos**. Esta se explicará mediante el **modelo entidad-relación**.

Este proyecto cuenta con una base de datos notablemente sencilla con un solo **tipo de entidad**. Una **entidad** es un tipo de objeto definido en base a la agregación de una serie de atributos [6] mientras que el tipo de entidad representa “el posible conjunto de objetos definidos en base a la

agregación de un mismo conjunto de atributos; es decir, en términos de abstracción, un tipo de entidad representa la clasificación de las entidades individuales” (Luque Ruiz et al., 2001, p. 41). En este caso tenemos el tipo de entidad **usuarios** el cual está compuesto por los atributos: correo, contraseña y archivoGuardado.

En cuanto a los **atributos** de un dominio, nos estaríamos refiriendo a la intención de ese dominio, y el valor del atributo será la extensión del dominio. A continuación, detallaremos el nombre y dominio de cada atributo, junto a una pequeña descripción de este, si puede ser nulo o no y su tipo.

Nuestro tipo de entidad cuenta con correo como **clave primaria**, es decir, este atributo no tomará el mismo valor para más de dos entidades de usuarios. Mientras que contraseña y archivoGuardado serán atributos simples, los cuales simplemente aportan un significado semántico.

Atributo	Descripción	Dominio	Nulo	Tipo
correo	Correo electrónico de identificación del usuario	varchar(255)	No	Clave primaria
contraseña	Clave de acceso del usuario	varchar(255)	No	Atributo simple
archivoGuardado	Datos de progreso del usuario en la aplicación	longtext	No	Atributo simple

Tabla 1: Tipo de entidad usuarios

El **diagrama entidad relación** de nuestra base de datos resulta obvio y bastante sencillo, pues cuenta con una sola entidad y no presenta relaciones, como se observa a continuación:

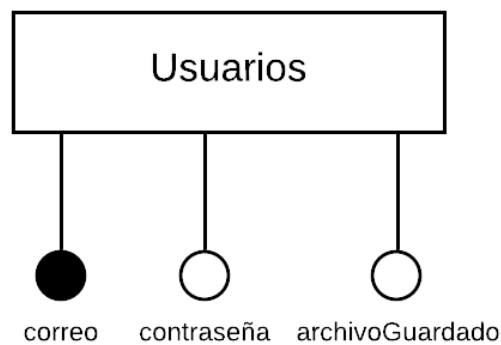


Figura 1: Diagrama E-R

Capítulo 8

ANÁLISIS DEL SISTEMA

En este apartado se llevará a cabo una descripción funcional de la aplicación mediante el desarrollo de Casos de Uso siguiendo el lenguaje *UML, Unified Modeling Language* [7].

8.1 Actores

Un actor es un componente principal de los casos de uso. Este es algo o alguien externo al sistema que interactúa de forma directa con el sistema. Este se representa con una imagen de un “muñeco de palo” con el nombre del actor debajo.

En este caso tendremos un solo actor que será el usuario.



Figura 2. Actor usuario

8.2 Casos de Uso

A continuación, se llevará a cabo la especificación de los casos de usos. Sobre cada caso de uso se especificará su objetivo, el actor principal, las precondiciones necesarias para que suceda, el escenario principal con los pasos a seguir, las excepciones que pueden ocurrir y las postcondiciones tras la ejecución del caso.

Nombre	Nombre del caso de uso.
Identificador	Identificador del caso de uso.
Objetivo	Objetivo del caso de uso, es decir, propósito del actor o actores.
Actor	Actor que interviene en el caso de uso.
Precondición	Condiciones que se deben cumplir para que se pueda llevar a cabo el escenario principal del caso de uso.
Escenario Principal	Serie de pasos que describen la sucesión de hechos que se darán con normalidad en el caso de uso.
Excepciones	Errores que pueden ocurrir durante el escenario principal y su consecuencia.
Postcondición	Condición que se debe satisfacer una vez haya terminado el escenario principal.
Importancia	Importancia que tiene el caso de uso en el sistema. Pudiendo ser baja, media o alta. Si es alta estamos hablando de un caso de uso imprescindible, mientras que si fuese baja tendríamos un caso de uso que no tiene una gran valía en el sistema.
Urgencia	Urgencia con la que se debe desarrollar el caso de uso.

Tabla 2: Casos de Uso. Ejemplo

En primer lugar, tenemos 2 casos de usos que tienen relación con la base de datos. En estos el usuario interactuará con el servidor usando el sistema como intermediario.

Nombre	Inicio de sesión
Identificador	CU1
Objetivo	Acceder al sistema
Actor	Usuario
Precondición	El usuario debe tener una cuenta registrada.
Escenario Principal	<ol style="list-style-type: none"> 1. El usuario abre la aplicación. 2. El usuario selecciona la opción "Iniciar sesión". 3. El usuario ingresa sus credenciales (nombre de usuario y contraseña). 4. El sistema verifica las credenciales. 5. El sistema permite el acceso al usuario si las credenciales son correctas.
Excepciones	<p>E5. Las credenciales son incorrectas, el sistema muestra un mensaje de error.</p> <p>E5. El sistema no puede verificar las credenciales debido a problemas de conexión, muestra un mensaje de error y solicita reintentar.</p>
Postcondición	El usuario ha iniciado sesión exitosamente en el sistema.
Importancia	Alta
Urgencia	Alta

Tabla 3: Caso de uso. Inicio de sesión

Nombre	Creación de cuenta
Identificador	CU2
Objetivo	Crear una cuenta personal
Actor	Usuario
Precondición	Ninguna
Escenario Principal	<ol style="list-style-type: none"> 1. El usuario abre la aplicación. 2. El usuario selecciona la opción "Crear cuenta". 3. El usuario ingresa los datos requeridos (nombre, correo electrónico, contraseña, etc.). 4. El usuario confirma la creación de la cuenta. 5. El sistema guarda la nueva cuenta y muestra un mensaje de confirmación.
Excepciones	E4. Los datos ingresados no cumplen con los requisitos establecidos: correo único y contraseña seguro. Por consiguiente, el sistema muestra un mensaje de error.
Postcondición	El usuario tiene una nueva cuenta creada exitosamente.
Importancia	Alta
Urgencia	Media

Tabla 4. Casos de uso. Creación de cuenta

Una vez el usuario hubiese iniciado sesión, surgiría naturalmente la necesidad del siguiente caso de uso, el cual podríamos considerar su opuesto:

Nombre	Cierre de sesión
Identificador	CU3
Objetivo	Cerrar sesión de la cuenta personal
Actor	Usuario
Precondición	El usuario debe haber iniciado sesión.
Escenario Principal	<ol style="list-style-type: none">1. El usuario selecciona la opción "Cerrar sesión".2. El sistema confirma la acción y cierra la sesión del usuario.
Excepciones	Ninguna
Postcondición	El usuario ha cerrado sesión exitosamente.
Importancia	Media
Urgencia	Media

Tabla 5: Casos de uso. Cierre de sesión

Continuamos revisando los casos de uso que definirán el principal uso de la aplicación, la interacción con los astros en realidad aumentada y el descubrimiento de logros.

Nombre	Visualizar astros
Identificador	CU4
Objetivo	Visualizar astros al apuntar con la cámara al cielo
Actor	Usuario
Precondición	El usuario debe haber iniciado sesión, la cámara del dispositivo debe estar funcionando y la aplicación debe tener permisos para acceder a la cámara del dispositivo.
Escenario Principal	<ol style="list-style-type: none"> 1. El sistema obtiene los datos de los astros del servidor. 2. El usuario apunta con la cámara de su dispositivo móvil al cielo. 3. La aplicación muestra los astros en la pantalla.
Excepciones	E1. El sistema no puede obtener los datos de los astros necesarios para calcular su posición, por lo tanto, no se instancian y se muestra un mensaje de error.
Postcondición	El usuario puede ver constelaciones y planetas en su entorno mediante el uso de la cámara de su móvil.
Importancia	Alta
Urgencia	Alta

Tabla 6: Casos de uso. Visualizar astros

CU5	Descubrir nuevos astros
Objetivo	Retroalimentación por parte de la aplicación al pulsar un astro.
Actor	Usuario
Precondición	El usuario debe haber iniciado sesión (CU1) y la Visualización de astros (CU3) debe estar activa.
Escenario Principal	<ol style="list-style-type: none"> 1. La aplicación muestra los astros en realidad aumentada (CU3). 2. El usuario interactúa con un astro por primera vez. 3. La aplicación le notifica de este suceso. 4. La aplicación guarda que este evento ha sucedido.
Excepciones	<p>E1. El sistema no puede obtener los datos de los astros necesarios para calcular su posición, por lo tanto, no se instancian y se muestra un mensaje de error.</p> <p>E4. El sistema no puede establecer una conexión con el servidor. Por lo tanto, no puede guardar el progreso correctamente. En este caso se informaría al usuario del error.</p>
Postcondición	La aplicación ha registrado el nuevo astro con el cual se ha interactuado como progreso del usuario.
Importancia	Media
Urgencia	Media

Tabla 7: Casos de uso. Descubrir nuevos astros

Nombre	Desbloquear logros
Identificador	CU6
Objetivo	Reconocimientos de logros al conseguir un cierto progreso establecido por el desarrollador.
Actor	Usuario
Precondición	El usuario debe haber iniciado sesión.
Escenario Principal	<ol style="list-style-type: none"> 1. El usuario realiza un progreso, descubre un astro, consulta información adicional o cambia de ubicación, desbloqueando un logro. 2. El sistema guarda el logros del usuario. 3. El sistema notifica al usuario del logro.
Excepciones	E2. El sistema no es capaz de establecer conexión con el servidor para guardar el progreso del jugador. Se informará al usuario del error.
Postcondición	El sistema debe guardar un registro del nuevo logro
Importancia	Media
Urgencia	Baja

Tabla 8: Casos de uso. Desbloquear logros

Por otra parte, debemos tener en cuenta todas la acciones que los usuarios realizarán separadamente de la mecánica principal de la aplicación. Tal es el caso de modificar su ubicación:

Nombre	Cambiar la ubicación del usuario
Identificador	CU7
Objetivo	Cambiar la ubicación del usuario
Actor	Usuario
Precondición	El usuario debe haber iniciado sesión en la aplicación.
Escenario Principal	<ol style="list-style-type: none"> 1. El usuario accede a la configuración de la aplicación. 2. El usuario selecciona la opción "Cambiar ubicación". 3. El usuario ingresa o selecciona la nueva ubicación. 4. El sistema actualiza la ubicación del usuario y muestra un mensaje de confirmación. 5. El sistema actualiza la ubicación de todos los objetos del cielo.
Excepciones	E3. La ubicación ingresada no es válida, la aplicación muestra un mensaje de error.
Postcondición	La ubicación del usuario ha sido actualizada correctamente.
Importancia	Baja
Urgencia	Baja

Tabla 9: Casos de uso. Cambiar ubicación del usuario

En el conjunto de actos que el usuario realiza más allá de la realidad aumentada podemos incluir todas las consultas que este puede efectuar para revisar información significativa para todo usuario. Así como sus avances o información suplementaria sobre los mismos.

Nombre	Consultar astros descubiertos
Identificador	CU8
Objetivo	Acceder y visualizar los astros descubiertos e información adicional de estos
Actor	Usuario
Precondición	El usuario debe estar logueado en la aplicación.
Escenario Principal	<ol style="list-style-type: none"> 1. El usuario accede a la vista en la cual pueda consultar la lista de astros descubiertos. 2. El sistema muestra la lista de astros descubiertos.
Excepciones	E2. Problemas de conexión, la aplicación muestra un mensaje de error y solicita reintentar.
Postcondición	Ninguna
Importancia	Baja
Urgencia	Baja

Tabla 10: Consultar astros descubiertos

Nombre	Consultar información adicional
Identificador	CU9
Objetivo	Obtener información y datos adicionales sobre los astros descubiertos
Actor	Usuario
Precondición	El usuario debe haber iniciado sesión en el sistema.
Escenario Principal	<ol style="list-style-type: none"> 1. El usuario accede a la vista en la cual pueda consultar la lista de astros descubiertos. 2. El sistema muestra la lista de astros descubiertos 3. El usuario selecciona un elemento de la lista. 4. El sistema muestra información adicional sobre el elemento seleccionado.
Excepciones	E4. No se encuentra los datos del elemento seleccionado, se muestra mensaje indicando el error.
Postcondición	Ninguna
Importancia	Baja
Urgencia	Baja

Tabla 11: Casos de uso. Consultar información adicional

Nombre	Consultar logros
Identificador	CU10
Objetivo	Acceder y visualizar al progreso realizado por el usuario en los logros
Actor	Usuario
Precondición	El usuario debe haber iniciado sesión en el sistema.
Escenario Principal	<ol style="list-style-type: none"> 1. El usuario accede a la vista en la cual pueda consultar sus logros. 2. El sistema muestra los logros realizados.
Excepciones	Ninguna
Postcondición	Ninguna
Importancia	Baja
Urgencia	Baja

Tabla 12: Casos de uso. Consultar logros

Por su parte, es reseñable que un sistema ofrezca la posibilidad de modificarlo a él mismo. El siguiente caso de uso contempla esta acción.

Nombre	Ajustar sistema
Identificador	CU11
Objetivo	Modificar variables del sistema
Actor	Usuario
Precondición	El usuario debe haber iniciado sesión en el sistema.
Escenario Principal	<ol style="list-style-type: none"> 1. El usuario accede a la vista en la cual pueda modificar variables del sistema. 2. El sistema muestra los variables modificables. 3. El usuario modifica las variables disponibles. 4. El sistema aplica los cambios.
Excepciones	Ninguna
Postcondición	Las variables modificadas deben presentar el nuevo valor dado por el usuario
Importancia	Baja
Urgencia	Baja

Tabla 13: Casos de Uso. Ajustar sistema

Una de las funcionalidades que el sistema ofrecerá será la capacidad de acceder a material educativo sobre astronomía. El caso de uso restante contempla esta posibilidad:

Nombre	Visualizar multimedia
Identificador	CU12
Objetivo	Ver material multimedia accesible desde el sistema
Actor	Usuario
Precondición	El usuario debe haber iniciado sesión en el sistema.
Escenario Principal	<ol style="list-style-type: none"> 1. El usuario accede a la vista en la cual se le presente multimedia relacionada con la astronomía. 2. El usuario selecciona el elemento al cual quiera acceder. 3. El sistema redirecciona al usuario al elemento seleccionado.
Excepciones	E3. El elemento seleccionado no está disponible por lo que el usuario no será redirigido. Se le informará de lo sucedido.
Postcondición	Ninguna
Importancia	Baja
Urgencia	Baja

Tabla 14: Casos de Uso. Visualizar multimedia

Finalizando con las descripciones de los casos de uso, un tutorial presente en el sistema siempre presenta una facilidad para el usuario que podrá acceder a una pequeña guía sin necesidad de informarse por terceros. Este último caso de uso contempla justamente este comportamiento.

Nombre	Ver tutorial
Identificador	CU13
Objetivo	Ver tutorial de la aplicación dentro del sistema
Actor	Usuario
Precondición	El usuario debe haber iniciado sesión en el sistema.
Escenario Principal	<ol style="list-style-type: none">1. El usuario selecciona la opción de ver tutorial2. El sistema le muestra el tutorial al usuario
Excepciones	Ninguna
Postcondición	Ninguna
Importancia	Baja
Urgencia	Baja

Tabla 15: Casos de Uso. Ver tutorial

8.3 Diagrama de Casos de Uso

Un diagrama de casos de uso es una representación visual de las interacciones entre los usuarios (o actores) y un sistema. Es utilizado para capturar los requisitos funcionales de un sistema, describiendo los escenarios en los que el sistema será utilizado. Los componentes principales de un diagrama de casos de uso que se han utilizado en este caso los podemos diferenciar en la siguiente figura:

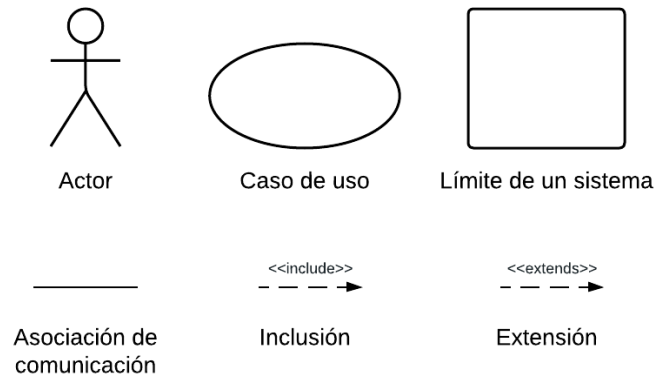


Figura 3: Componentes Diagramas de Casos de Uso

Estos son:

- Sistema: Se representa como un rectángulo grande que contiene los casos de uso. El sistema delimita el alcance de lo que se está modelando.
- Casos de uso: Representan las funcionalidades o servicios que el sistema proporciona a los actores. Se dibujan como óvalos dentro del rectángulo del sistema.
- Relaciones:
 - Asociación: Es una línea sólida que conecta a un actor con un caso de uso, indicando que el actor participa en ese caso de uso.
 - Inclusión (<<include>>): Es una línea punteada con una flecha que conecta dos casos de uso. Indica que un caso de uso incluye el comportamiento de otro caso de uso. Por ejemplo, si Iniciar sesión incluye a Ajustar sistema, quiere decir que no podrá ajustar el sistema sin haber iniciado sesión antes.
 - Extensión (<<extend>>): Es una línea punteada, al igual que la inclusión, con una flecha que también une dos casos de uso. A diferencia de la inclusión, la extensión define un comportamiento opcional que puede derivar de otro caso de uso. Tal es el caso de desbloquear logros, que es una acción que solo podrá llevarse a cabo al interactuar con otros sistema, pero no

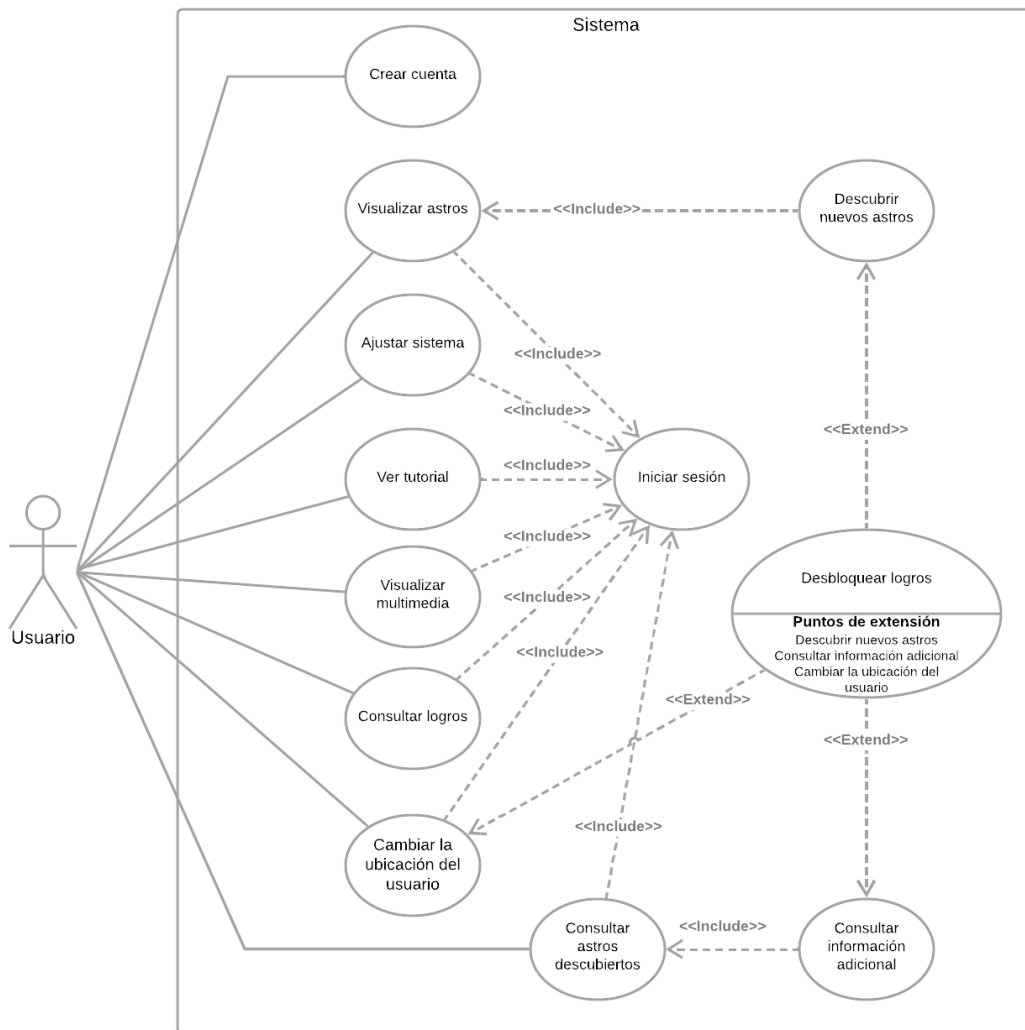


Figura 4: Diagrama de casos de uso

El diagrama representa cómo el usuario interactúa con nuestro sistema para realizar diversas tareas relacionadas. A continuación, se describirán las interrelaciones más destacables del diagrama.

El caso de uso de **Creación de cuenta** no está conectado a otros casos de uso mediante relaciones de inclusión o extensión. Esto indica que es una funcionalidad autónoma que el usuario puede realizar directamente sin depender de otras funcionalidades del sistema.

Por otro lado, **Inicio de sesión** es un caso de uso central e imprescindible en el diagrama. Se distingue como la gran mayoría de casos lo incluyen, es decir, para estos casos será necesario iniciar sesión para que se puedan llevar a cabo.

También encontramos **Consultar información adicional** y **Descubrir nuevos astros** con relaciones de inclusión con **Consultar astros descubiertos** y **Visualizar astros** respectivamente. Estos includes se dan a cause de que para consultar la información adicional de algún astro primero deberemos consultar los astros que hayamos descubiertos. Similarmente, no podremos descubrir nuevos astros si no los podemos visualizar.

Por último, el caso de uso **Desbloquear logros** también destaca por sus tres relaciones de extensión. Estas quieren decir que solo podremos desbloquear un logro cuando consultemos información adicional, al cambiar la ubicación del usuario o al descubrir nuevos astros. Sin embargo, es una acción complementaria y opcional a los casos de uso con los que se relaciona, es decir, no siempre que descubramos un astro se desbloqueará un logro, pero eventualmente se podrá dar esta situación.

8.4 Validación de Casos de Uso

Con el fin de validar nuestros casos de uso se hará uso de una **matriz de trazabilidad**. En esta matriz enfrentaremos los requisitos funcionales con los casos de uso. De esta manera podremos evaluar correctamente si hemos cubierto todas las funcionalidades especificadas en los requisitos.

CU RF	01	02	03	04	05	06	07	08	09	10	11	12	13
01		⊗											
02	⊗												
03			⊗										
04				⊗									
05					⊗								
06							⊗						
07					⊗	⊗							
08						⊗							
09								⊗		⊗			
10									⊗				
11				⊗									
12											⊗		
13												⊗	
14													⊗

Tabla 16: Matriz de Trazabilidad

En conclusión, los casos de uso sí cubren todos los requisitos funcionales que se especificaron anteriormente.

Capítulo 9

DISEÑO DEL SISTEMA

En este capítulo se procederá con la especificación del diseño de nuestra aplicación. Comenzando por ponerle nombre y crearle una marca visual, para, a continuación, seguir detallando como será visualmente el sistema y finalizando con una explicación de la arquitectura de nuestro proyecto en conjunto a todos los diferentes sistemas o elementos de este.

9.1 Diseño de Marca

En el desarrollo de la marca para nuestra aplicación he puesto especial énfasis en la creación de una identidad visual que no sea demasiado compleja, pues el diseño gráfico no es el objeto de este proyecto, y que tenga relación con el tema tratado. Para ello, se ha definido una paleta de colores la cual se usará en los diferentes elementos gráficos de la aplicación, como los botones. En este punto también se presentan los iconos que servirán como principal identificador de la aplicación junto con su nombre.

9.1.1 Nombre

El nombre que tendrá nuestro producto es una decisión de vital importancia no solo para una aplicación móvil, sino para cualquier producto que se quiera incorporar en el mercado. De hecho, esta característica formará las primeras opiniones no neutrales sobre nuestro producto, que a posteriori serán complicadas de cambiar (Zinkhan y Martin, 1987) [8].

Esta aplicación se llamará: “Look Up”. Se ha buscado que sea sencillo de recordar, fácil de pronunciar y que, obviamente, de una idea sobre qué trata nuestro software.

9.1.2 Paleta de Colores

Una paleta de colores bien definida ayudará a mantener una mayor coherencia del diseño de la interfaz. Mediante el uso de la herramienta Adobe Stock, que ofrece una librería de paletas de colores, se ha elegido la paleta de la Figura 5. Se ha buscado un conjunto de colores con tonos azulados que recuerden al cielo, estando así en sintonía con la temática del proyecto.

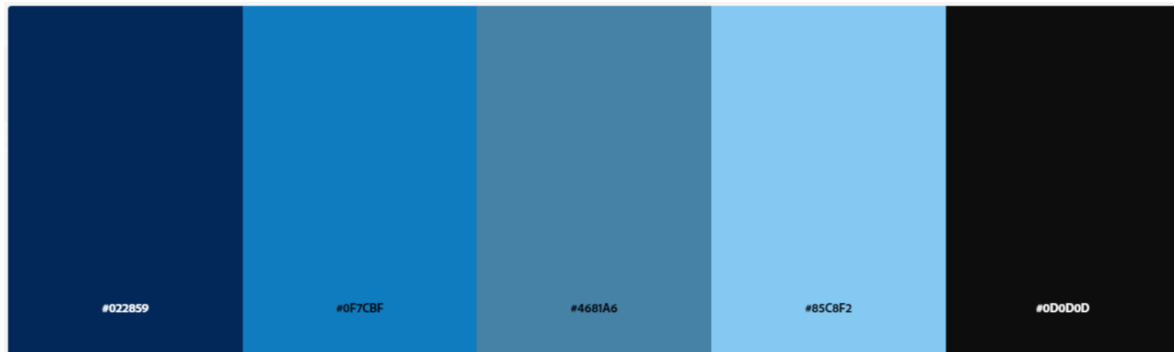


Figura 5: Paleta de colores

9.1.3 Marca Gráfica

Por último, se definirán los principales elementos gráficos que servirán para identificar nuestro proyecto. Los dividiré en logotipo, isotipo e isologo [9]:

- Logotipo, letras o cifras sin ícono o imagen:

LOOK UP

Figura 6: Logotipo

- Isotipo, icono o imagen sin letras o cifras:



Figura 7: Isotipo

- Isologo, combinación de un logotipo con un isotipo, pero fundidos en un solo elemento gráfico:



Figura 8: Logotipo e Isotipo



Figura 9: Logotipo e isotipo color invertido

9.2 Diseño de Interfaz

El diseño de la interfaz de usuario es un componente crítico en el desarrollo de nuestra aplicación Look Up. Una interfaz bien diseñada no solo mejora la experiencia del usuario, sino que también garantiza que todas las funcionalidades de la aplicación sean accesibles y fáciles de usar.

En este contexto, se han maquetado una serie de vistas y menús que permiten a los usuarios interactuar con la aplicación de manera intuitiva y eficiente. Estas son:

- **Pantallas de inicio de sesión y registro:** Ambos presentan prácticamente los mismos elementos. Dos campos para introducir datos, concretamente el correo electrónico y contraseña. Aunque en el caso de Registro serán 3 campos, pues se requerirá volver a introducir la contraseña. Bajo estos campos encontraremos un mensaje de retroalimentación donde la aplicación nos irá informando de los posibles errores que

puedan ir surgiendo. Por último, el botón homónimo a la ventana en la que se encuentre servirá para realizar la correspondiente consulta a la base de datos, mientras que el botón sobrante nos conducirá a la ventana que indique.

La imagen muestra una maqueta de la interfaz de usuario dividida en dos paneles por una barra vertical central. El panel izquierdo está encabezado por 'Iniciar Sesión' y el panel derecho por 'Registro'. Ambos paneles contienen los siguientes elementos de interfaz:

- Un campo de entrada para 'Introducir correo electrónico'.
- Un campo de entrada para 'Introducir contraseña'.
- Un botón etiquetado como 'Retroalimentación'.
- Un botón para 'Iniciar sesión'.
- Un botón para 'Ir a registro'.

Adicionalmente, el panel de 'Registro' incluye un campo de entrada para 'Repetir contraseña' situado entre los campos de contraseña y el botón de retroalimentación.

Figura 10: Maqueta Iniciar Sesión y Registro

- **Pantalla principal del juego:** esta será la vista donde podremos usar la principal función de la aplicación, usar la realidad aumentada para obtener información adicional del cielo, en cumplimiento de RF2. Además, se incluirá un botón de menú para acceder a opciones adicionales. También nos mostrará la ubicación que hayamos elegido.

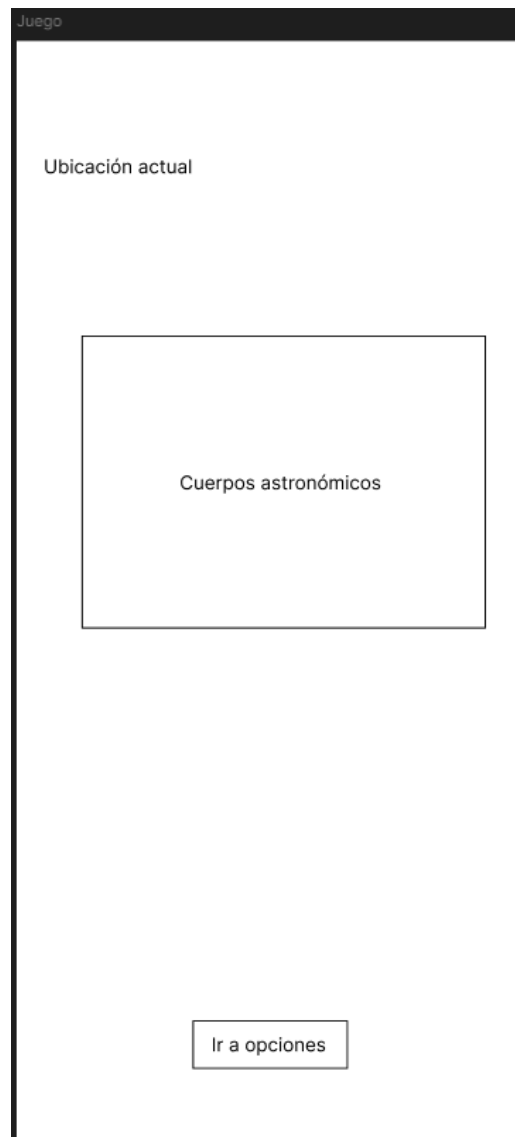


Figura 11: Maqueta Juego

- **Menú de opciones:** El menú proporcionará acceso a la biblioteca de astros descubiertos, logros obtenidos, ajustes, menú de cambio de ubicación y la opción de cerrar sesión. También estará presente a partir de ahora los botones “Ir al juego”, el cual nos devolverá a la pantalla de juego, y “Volver atrás” que nos devolverá a la pantalla desde la que hayamos accedido a la que nos encontremos.

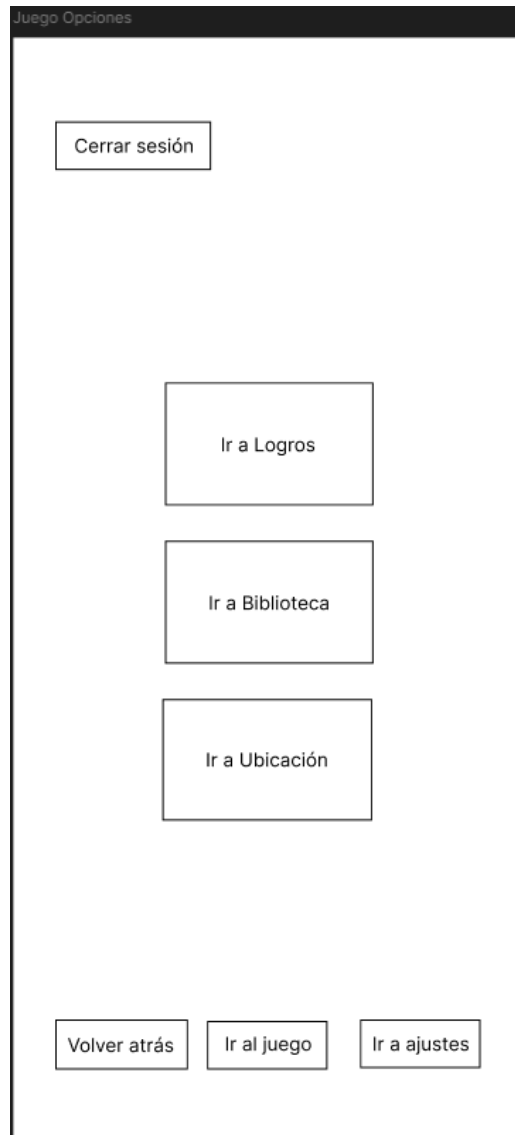


Figura 12: Maqueta Menú de Opciones

- **Pantallas de biblioteca:** Estas se dividen en Progreso y Aprendizaje. Podremos alternar entre ambas pantallas mediante sus botones correspondientes. Desde Progreso se podrá acceder a la pantalla de información interactuando con algún elemento de la lista de cuerpos astronómicos.

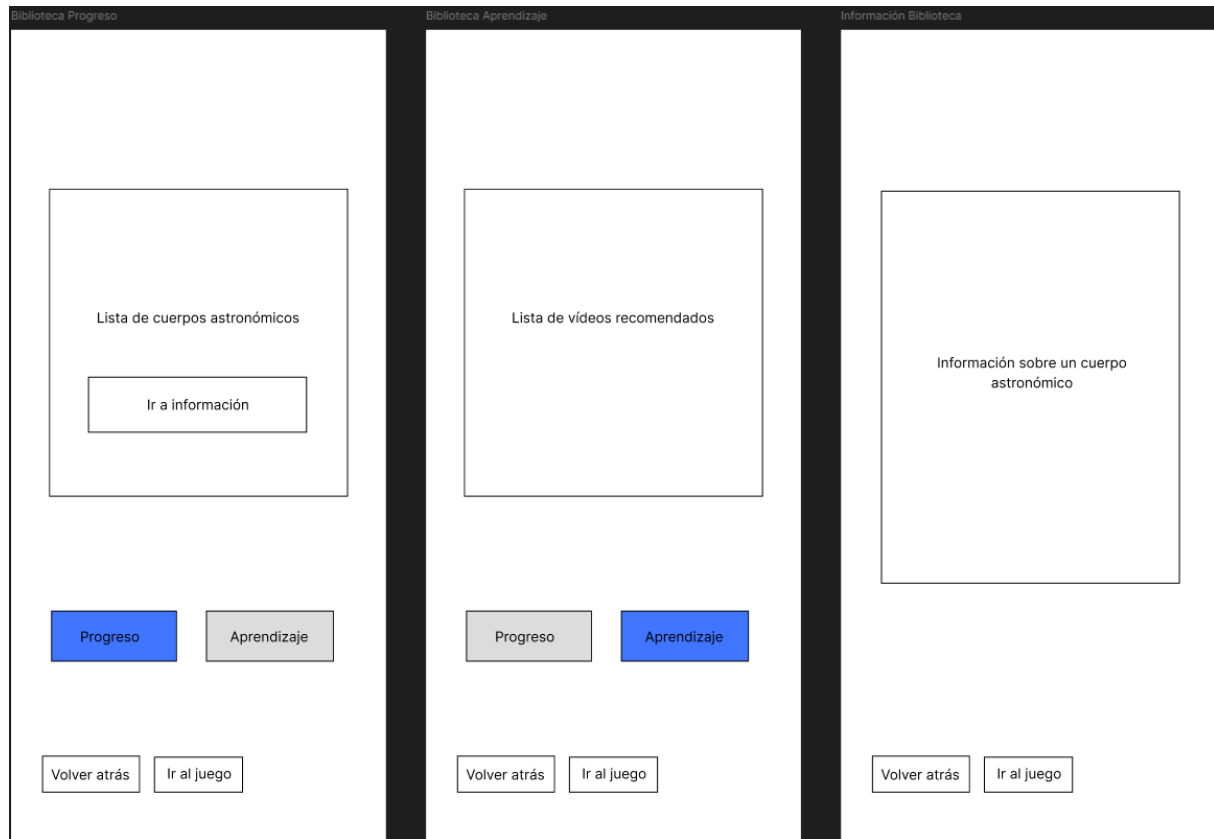


Figura 13: Maqueta de Biblioteca Progreso, Aprendizaje e Información

- **Pantalla de logros y ubicación:** En la pantalla de ubicación podremos buscar las ciudades por sus nombre para después recibir una lista con las coincidencias, estas serán seleccionables para cambiar la ubicación a esas localizaciones. En cuanto a la pantalla de logros, simplemente servirá como expositor de los resultados de los usuarios.

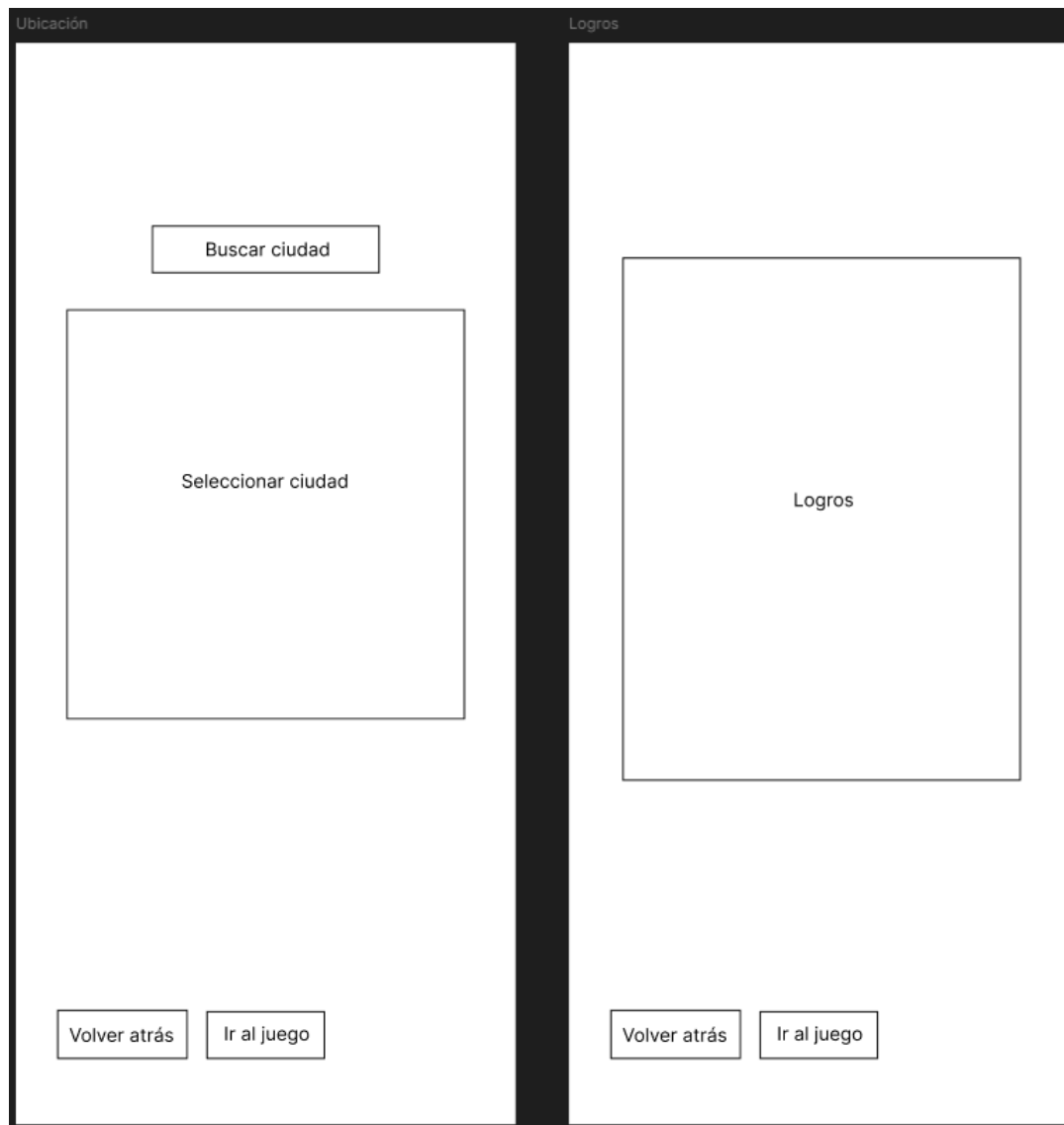


Figura 14: Maqueta de Ubicación y Logro

- **Pantalla de tutorial:** esta pantalla ofrecerá una serie de diapositivas que podremos encontrar en el centro de la pantalla, para explicar el correcto funcionamiento de la aplicación y facilitar su uso. En las esquina inferiores izquierda y derechas se localizan 2 botones para retroceder o avanzar una diapositiva respectivamente. Por último, el botón “Saltar tutorial” nos sacará del tutorial y nos llevará a la pantalla de juego.

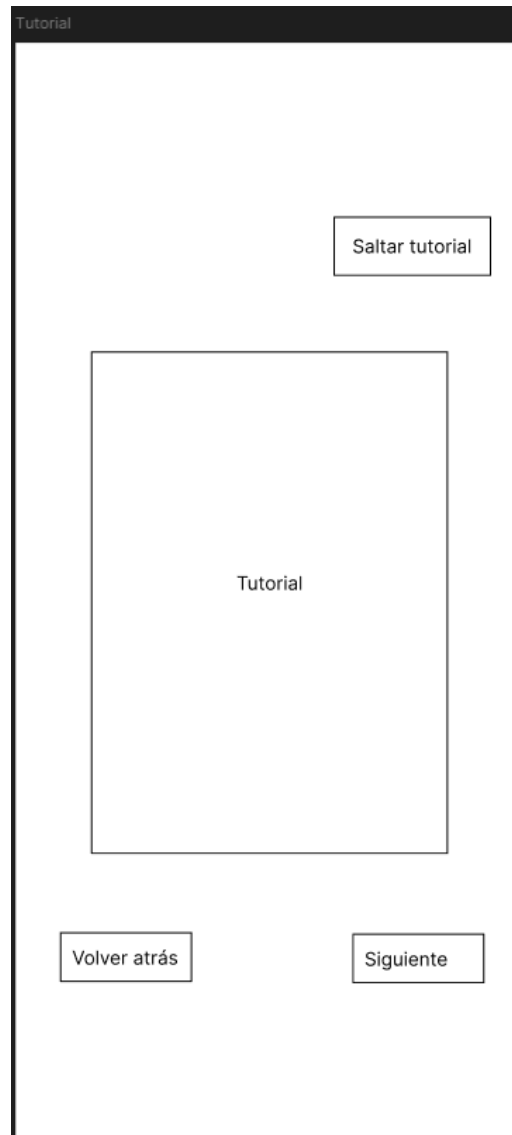


Figura 15. Maqueta de Tutorial

- **Pantalla de ajustes:** Esta será la pantalla que mayor número de opciones ofrecerá. Se podrá variar el valor de diferentes variables del sistema como los efectos de sonido, música, idioma y opacidad de cielo. También se podrá acceder al tutorial desde esta misma vista.

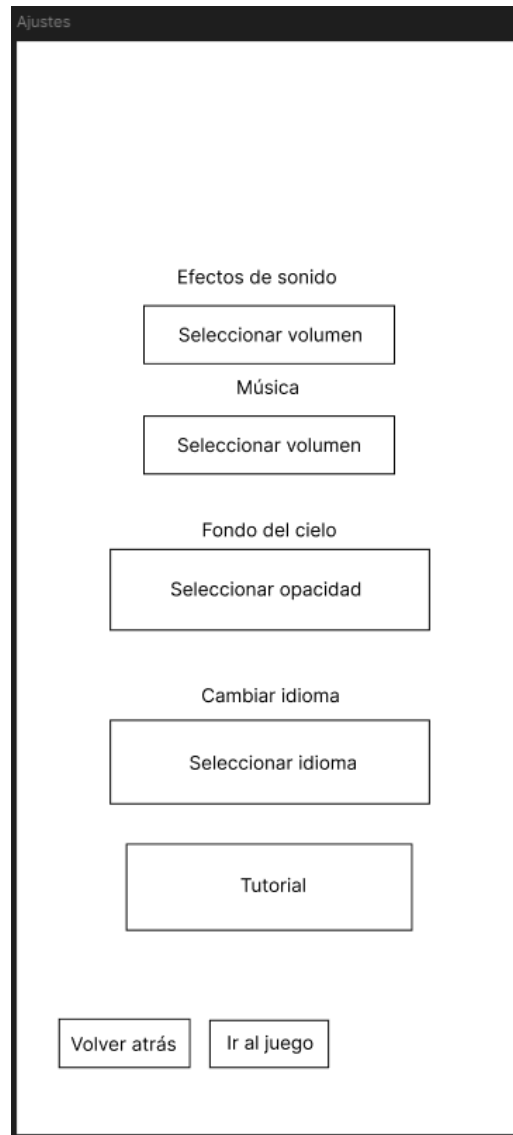


Figura 16: Maqueta de Ajustes

9.3 Usabilidad del Sistema

La usabilidad de un sistema se refiere a la capacidad de un software de ser comprendido, aprendido, usado y ser atractivo para el usuario [10]. Esta es un factor clave en el éxito de cualquier aplicación, y en el caso de *Look Up*, cobra aún más relevancia debido a la necesidad de ofrecer una experiencia educativa que, además, debe ser intuitiva y atractiva para el usuario.

9.3.1 Captación Visual

La interfaz de usuario se ha diseñado teniendo en cuenta principios de simplicidad y consistencia. Cada elemento visual y funcional ha sido cuidadosamente ubicado para facilitar la navegación, minimizando la cantidad de pasos necesarios para acceder a las diferentes secciones de la aplicación. El menú principal, accesible en cualquier momento a través de un botón estático en la pantalla, permite al usuario cambiar de una funcionalidad a otra sin perder el contexto actual, lo que mejora la experiencia de exploración.

Además, la aplicación ofrece retroalimentación constante. Por ejemplo, cuando el usuario descubre un astro, no solo se le informa de su logro mediante notificaciones claras, sino que también se actualizan automáticamente los registros de progreso en tiempo real, reforzando así la sensación de avance y motivación en el uso de la aplicación.

De igual modo, hay que tener en cuenta que si el usuario juega en un entorno muy iluminado le sería más complicado diferenciar correctamente las representaciones de los diferentes cuerpos, por esto se ofrecerá la opción de oscurecer el fondo sobre el cual se construyen los diferentes cuerpos.

Asimismo, se ha usado un diseño responsivo que garantiza una correcta visualización en diferentes tamaños de pantalla y resoluciones, manteniendo la coherencia estética y funcional en todos los dispositivos Android que cumplan los requisitos mínimos para ejecutar a la aplicación.

9.3.2 Localización

Look Up ha sido localizada a inglés y francés, de tal forma que más usuarios puedan usarla entendiendo todo a la perfección y no solo hispanoparlantes.

Para esta tarea se ha usado el paquete de Unity **Localization** [11]. El cual nos permite primeramente seleccionar los idiomas que usaremos para que el IDE cree los Locale de cada idioma. Un **Locale** es simplemente la representación de un idioma dentro del entorno.

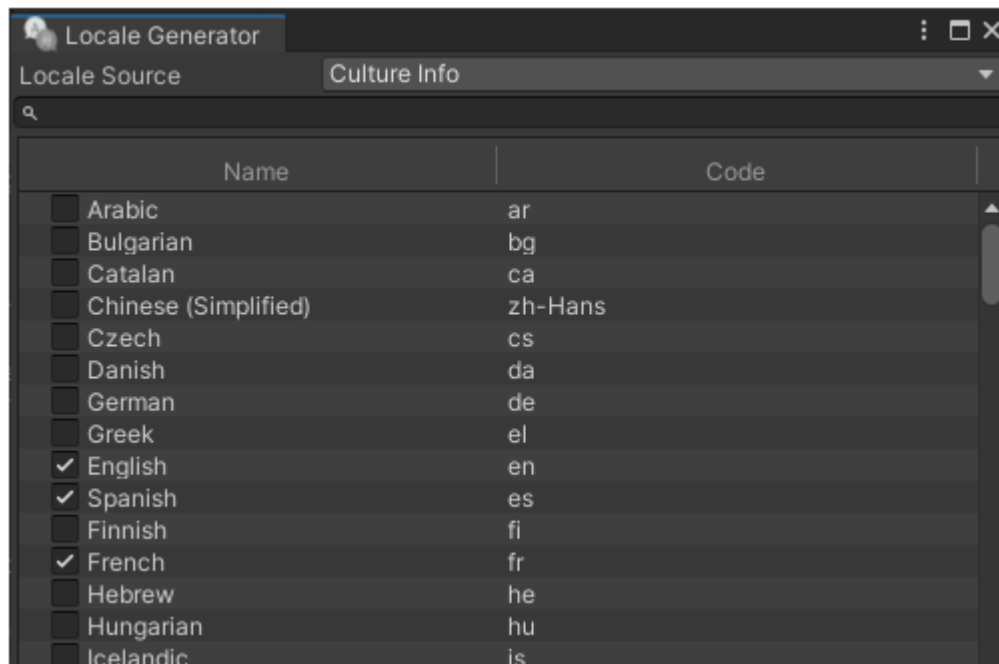


Figura 17: Unity Localization. Locale generator

Una vez hayamos creado los correspondientes Locales de los tres idiomas elegidos, será el momento de crear las tablas de cadenas que usará Unity para traducir los texto debidamente. Siendo más específicos, se han creado las siguientes 6 tablas:

- **Biblioteca:** en la cual se almacenarán todas las descripciones que se obtienen de los astros al entrar en sus detalles.
- **Errores:** para guardar los errores que se puedan producir en la diferentes pantallas del sistema.
- **Logros:** usada para todo el texto con relación a los logros, ya sea. sus nombres o descripciones.
- **Pre-Juego:** esta tabla servirá para las cadenas presentes en las pantallas de inicio de sesión y registro.
- **Tutorial:** como su nombre indica estará enfocada en almacenar toda la información del tutorial.
- **General:** el texto que no pertenezca a ninguna de las tablas indicadas irá en general. Estos son generalmente el texto de los botones de los menús.

Cada una de estas tablas tiene el siguiente formato:

Columna	Descripción
Key	Identificador de la cadena el cual indicaremos en los diferentes campos de texto de la aplicación
Id	Número asignado por Unity
Spanish(es)	Cadena en español
English(en)	Cadena en inglés
French(fr)	Cadena en francés

Tabla 17: Formato Tabla de Localización de Cadenas

Unity nos ofrece la posibilidad de importar CSV o directamente ir añadiendo registros desde el entorno. En la siguiente figura podemos ver, por ejemplo, la tabla de tutorial:

Key	Spanish (es)	English (en)	French (fr)
tuto1	Bienvenido a Look Up	Welcome to Look Up	Bienvenue à Look Up
tuto2	A continuación, le explicaremos las diferentes funciones de esta aplicación.	Next, we will explain the different functions of this application.	Ensuite, nous expliquerons les différentes fonctions de cette application.
tuto3	Pulsa...	Press...	Appuyez sur...
tuto4	Para acceder al menú de ajustes desde el cual podrás configurar el sonido, fondo del cielo e	To access the settings menu, where you can configure the sound, sky background, and	Pour accéder au menu des paramètres, où vous pourrez configurer le son, le fond du ciel et
tuto5	Cualquier constelación o planeta para	Any constellation or planet to 'discover' and add to your	Toute constellation ou planète pour 'découvrir' et ajouter

Figura 18: Tabla de Cadenas Tutorial

Podemos contemplar como tenemos por un lado las Keys, que no tiene porque tener relación con el texto que identifique (en este caso es “tuto” más el orden en el que sale el texto en el tutorial), y por el otro lado hallamos el mismo texto traducido a los tres idiomas seleccionados.

Después de tener listas todas las tablas necesarias se debe añadir el componente “Localize String Event” a los objetos que contengan los texto que queramos localizar. Como se observa en la siguiente figura a este componente se le debe indicar el texto a actualizar en “Update String” y la key de las cadenas correspondientes al objeto que hayamos seleccionado.

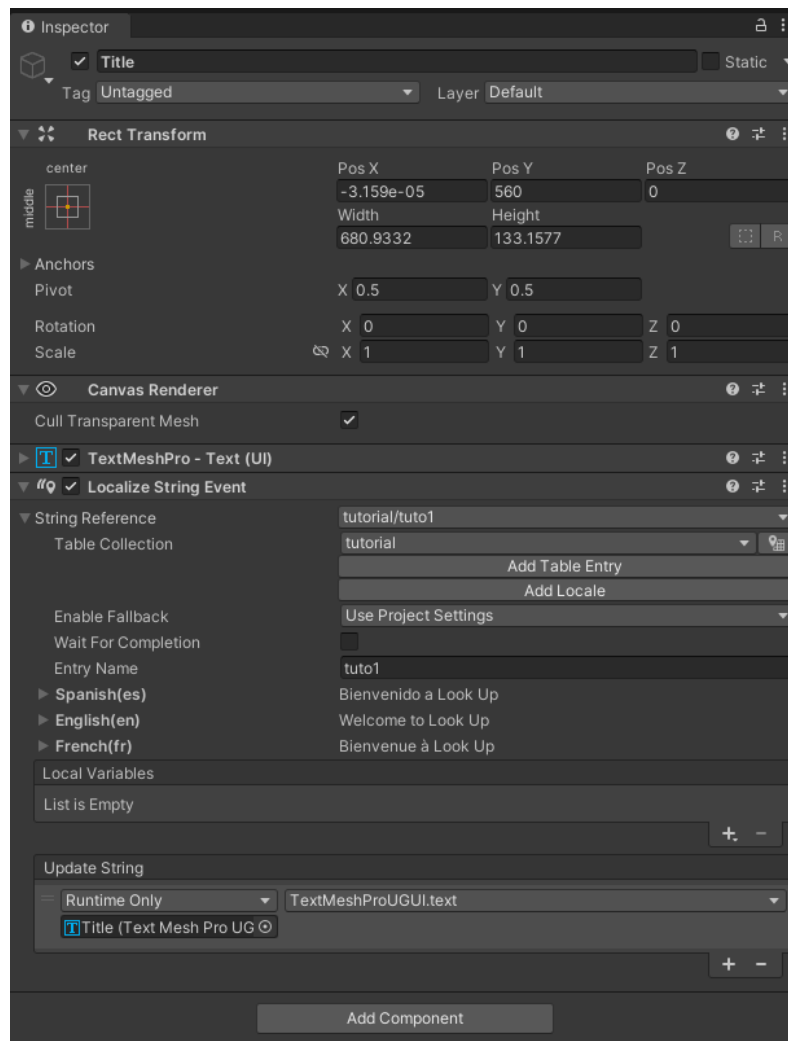


Figura 19: Localize String Event

Analizando la anterior figura, se le ha indicado el componente de Text del mismo objeto y la key “tuto1” perteneciente a la tabla del tutorial. Este proceso se ha seguido con todos los textos de Look Up, exceptuando las cadenas de objetos que se construyen en ejecución como los paneles de la biblioteca o logros.

Para el caso en el que tengamos que localizar objetos que no estén en escena se puede recurrir a la clase **LocalizationSettings**, por ejemplo: ‘LocalizationSettings.StringDatabase.GetLocalizedString(“biblioteca”, title)’ se utiliza para modificar la información adicional sobre el astro al que hayamos accedido en la pantalla de detalles. Cada vez que accedemos a los detalles dentro de la biblioteca, LocalizationSettings se encargará de buscar en la tabla de la biblioteca la entrada con la key que coincida con title, siendo este último el nombre del

astro que se habrá pasado como argumento. Como se puede intuir, en la tabla de biblioteca se ha usado el nombre de los astros como keys para sus respectivas descripciones.

9.3.3 Accesibilidad

Por último, se ha prestado especial atención a la accesibilidad, garantizando que los textos, iconos y elementos interactivos de la aplicación sean lo suficientemente claros y grandes para facilitar su uso por personas con posibles limitaciones visuales o motrices. Igualmente se ha incorporado una opción de ajustes desde la cual se podrán modificar ciertos valores de la aplicación como el sonido, idioma y fondo del cielo como se explicó anteriormente. Desde este mismo menú también se podrá ver un tutorial sobre la aplicación, para que los usuarios lo puedan revisar cuando lo necesiten.

9.4 Prototipado

Realizar un prototipo de la interfaz es una buena práctica para un desarrollo más eficiente. Tener un diseño decidido agilizará la codificación del *frontend*, parte de una aplicación que interactúa con los usuarios, pues, como resulta evidente, es más fácil realizar cambios en una herramienta de prototipado como Figma o Pencil que probar diseños directamente desde el entorno de desarrollo.

Para empezar, en la Figura 20 podemos apreciar las tres vistas por las que cualquier usuario pasará normalmente para iniciar la aplicación. Las ventanas de **Inicio de Sesión**, **Registro** y **Carga**.

Esta última ventana de carga se mostrará una vez se haya iniciado sesión correctamente. Servirá de telón para permitir a nuestro programa cargar el progreso del usuario y sus correspondientes logros, calcular la ubicación de todos los elementos y colocar los diferentes objetos en el entorno.

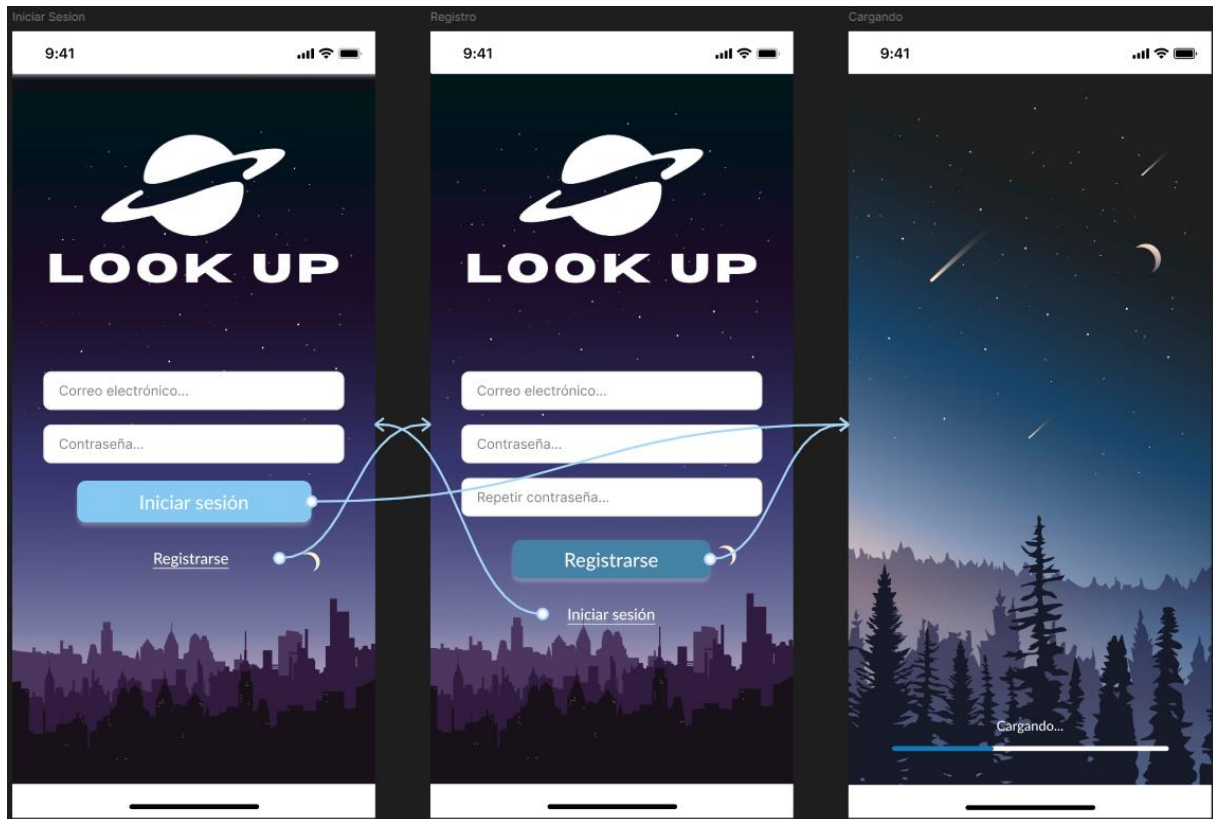


Figura 20: Inicio de aplicación

A continuación, la Figura 21 nos muestra las dos principales escenas del sistema por las que se moverá el usuario.

A la izquierda podemos observar una recreación de cómo se vería la **pantalla principal** de la aplicación. Observamos como podemos ver unas constelaciones las cuales al interactuar con ellas serían añadidas a nuestra biblioteca, pues habrían sido descubiertas. En la esquina superior izquierda se nos informa de nuestra ubicación actual mientras que en la parte inferior de la pantalla podemos encontrar un botón blanco con el isotipo de Look Up en negro, este será el diseño del botón por el que accederemos al menú de opciones.

El menú de opciones se muestra a la derecha de la Figura 21. Tenemos todas las opciones que se han visto en la maqueta, simplemente se han reorganizado los botones de biblioteca, logros y ubicación para tener un mejor aspecto.

También cabe recalcar que en la parte inferior encontramos 2 botones que se mantendrán estáticos durante los siguientes menús como se explicó en los bocetos. El botón de “volver atrás” se

representa con una flecha blanca sobre fondo negro apuntando hacia la izquierda, mientras que el botón de “volver al juego” es igual al botón para acceder al menú, pero con colores invertidos.

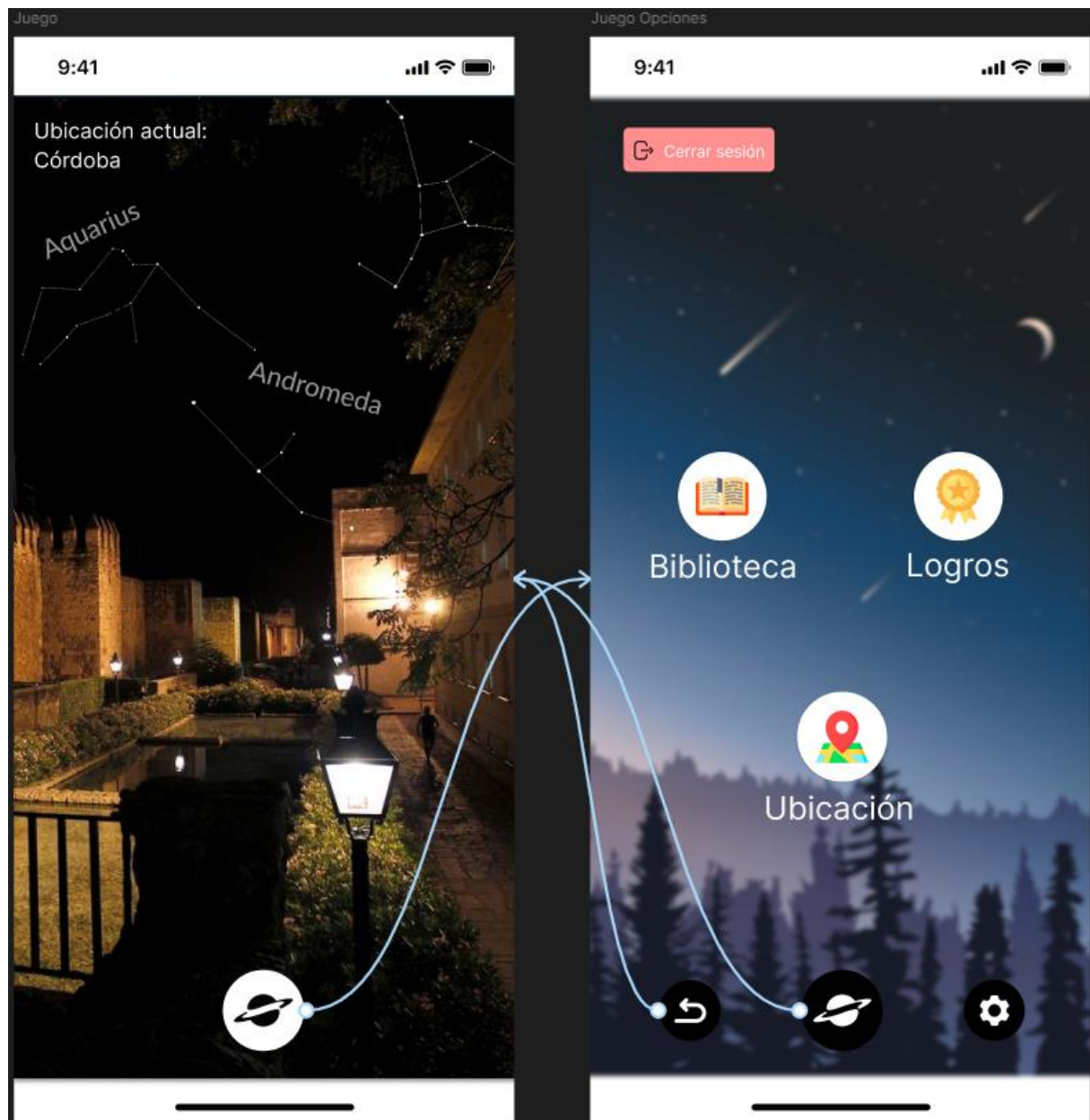


Figura 21: Look Up vistas principales

A continuación, en la Figura 22 vemos de izquierda a derecha: **Biblioteca/Aprendizaje**, **Biblioteca/Progreso** e **Información**. Se han añadido ejemplos a todas la pantallas para ser más ilustrativo.

En estas pantallas tenemos los botones Progreso y Aprendizaje. Siempre deberá haber uno de los dos activos, pero nunca los dos a la vez. Para representar la inactividad del que no esté en pantalla se ha optado por reducir su opacidad, lo que fácilmente es entendible visualmente.

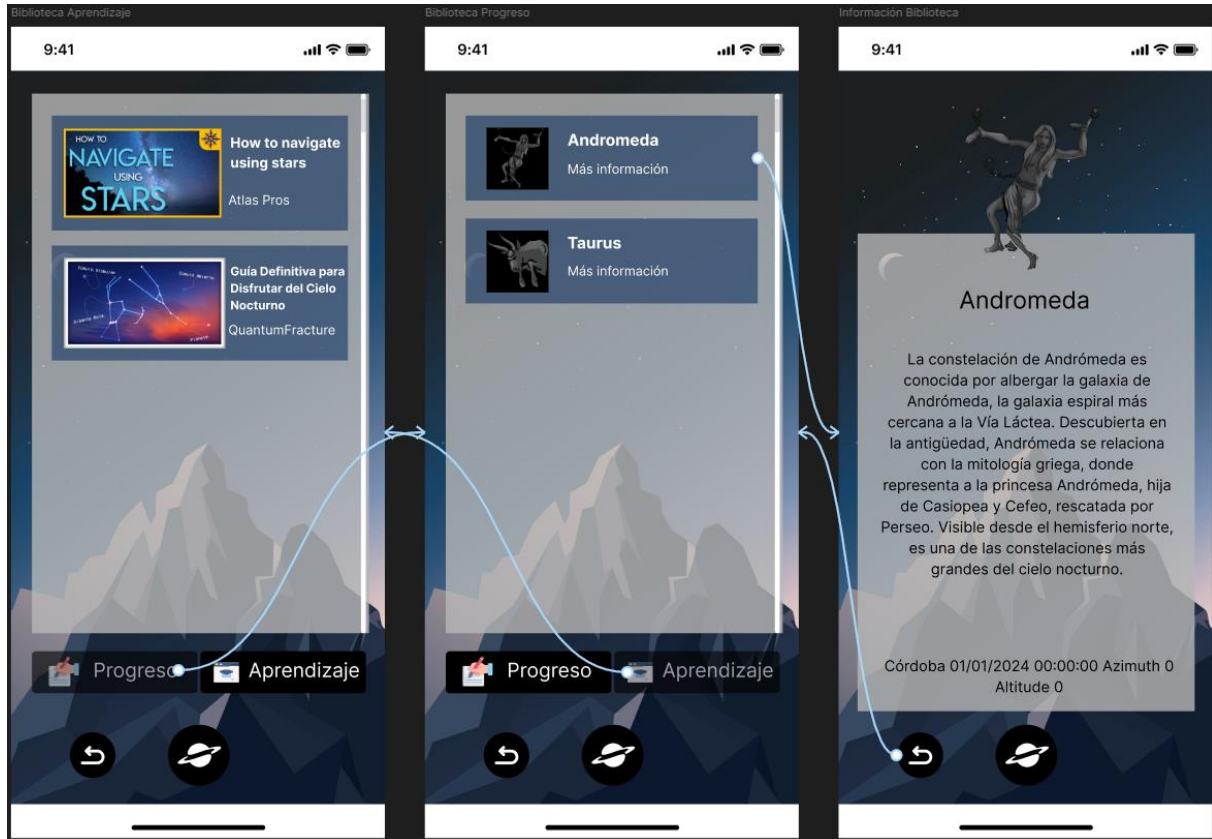


Figura 22: Biblioteca e información

En la siguiente figura encontramos el menú de **ubicación** que nos permite buscar una ciudad y nos ofrece una lista de ubicaciones con el nombre buscado. Una vez seleccionemos la nuestra, el sistema nos notificará con un diálogo de confirmación. Al aceptar dicho diálogo confirmaremos nuestro cambio de ubicación y se nos llevará de vuelta a la pantalla de carga mientras la aplicación recalcula la posición de los astros.

La última vista de esta misma figura nos ofrece la pantalla de los **logros**. Se presenta su icono, título y el hecho por el cual se ha conseguido.

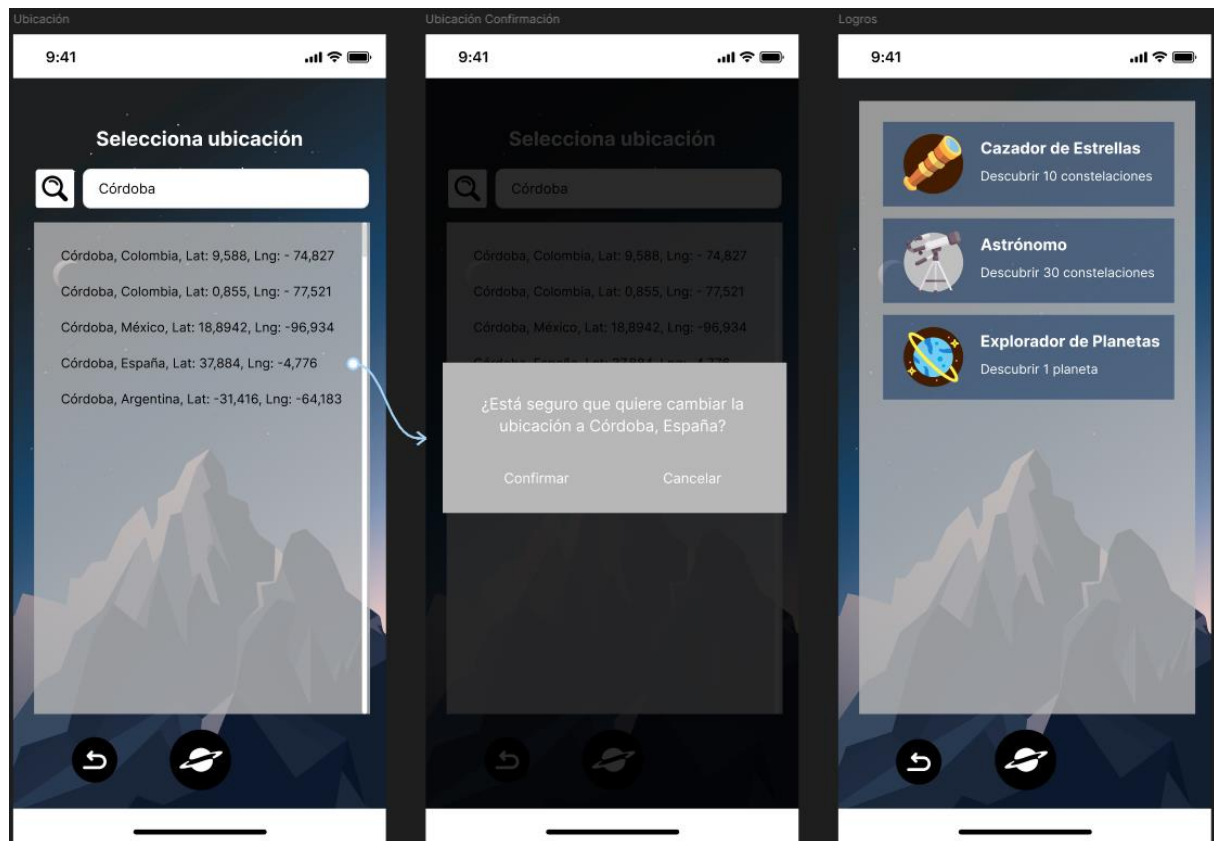


Figura 23: Menú ubicación

Por último, queda por señalar la vista que obtendríamos al conseguir un logro y los ajustes o configuración.

En cuanto a la **notificación de logro** es un diálogo que se superpondría a la vista en la que se encontrase el jugador al haberlo conseguido. Por ejemplo, en el prototipo se ha conseguido un logro obtenible por indagar en la Biblioteca en múltiples ocasiones.

En los **ajustes** tendremos todas las opciones que se indicaron en el punto de Usabilidad del Sistema. Se ofrecerán unas *sliders*, elemento de la interfaz de usuario que permite a los usuarios seleccionar un valor moviendo un control deslizante a lo largo de una barra, para configurar el valor de los diferentes parámetros especificados anteriormente. Podremos cambiar al idioma que deseemos pulsando en la bandera pertinente y será posible acceder al tutorial de la aplicación pulsando el texto subrayado “Ver tutorial”.

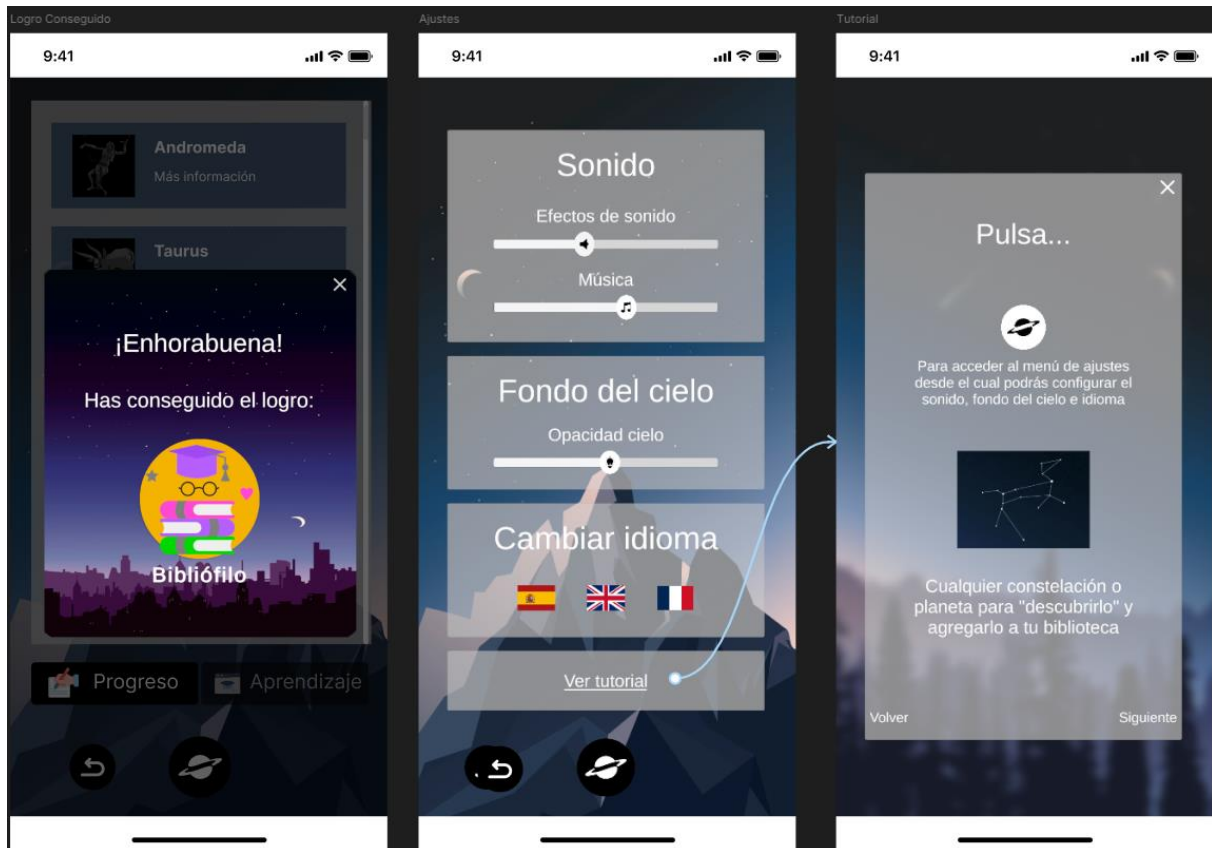


Figura 24: Logro conseguido, ajustes y tutorial

9.5 Diseño de Arquitectura

En este apartado se definirá la arquitectura del desarrollo de nuestra aplicación, es decir, el conjunto de aplicaciones o métodos que dan resultado al correcto ciclo de creación, uso y mantenimiento de nuestra aplicación.

La Figura 25 muestra dicha arquitectura. En esta toman partida 2 agentes, el desarrollador (yo en este caso) y los usuarios. A continuación, se indagará a fondo en las diferentes herramientas que he usado como desarrollador, cómo las he utilizado y su integración entre ellas. Por la parte del usuario se explicará la interacción que tendrá con la capa del desarrollador una vez tuviera la aplicación en su dispositivo.

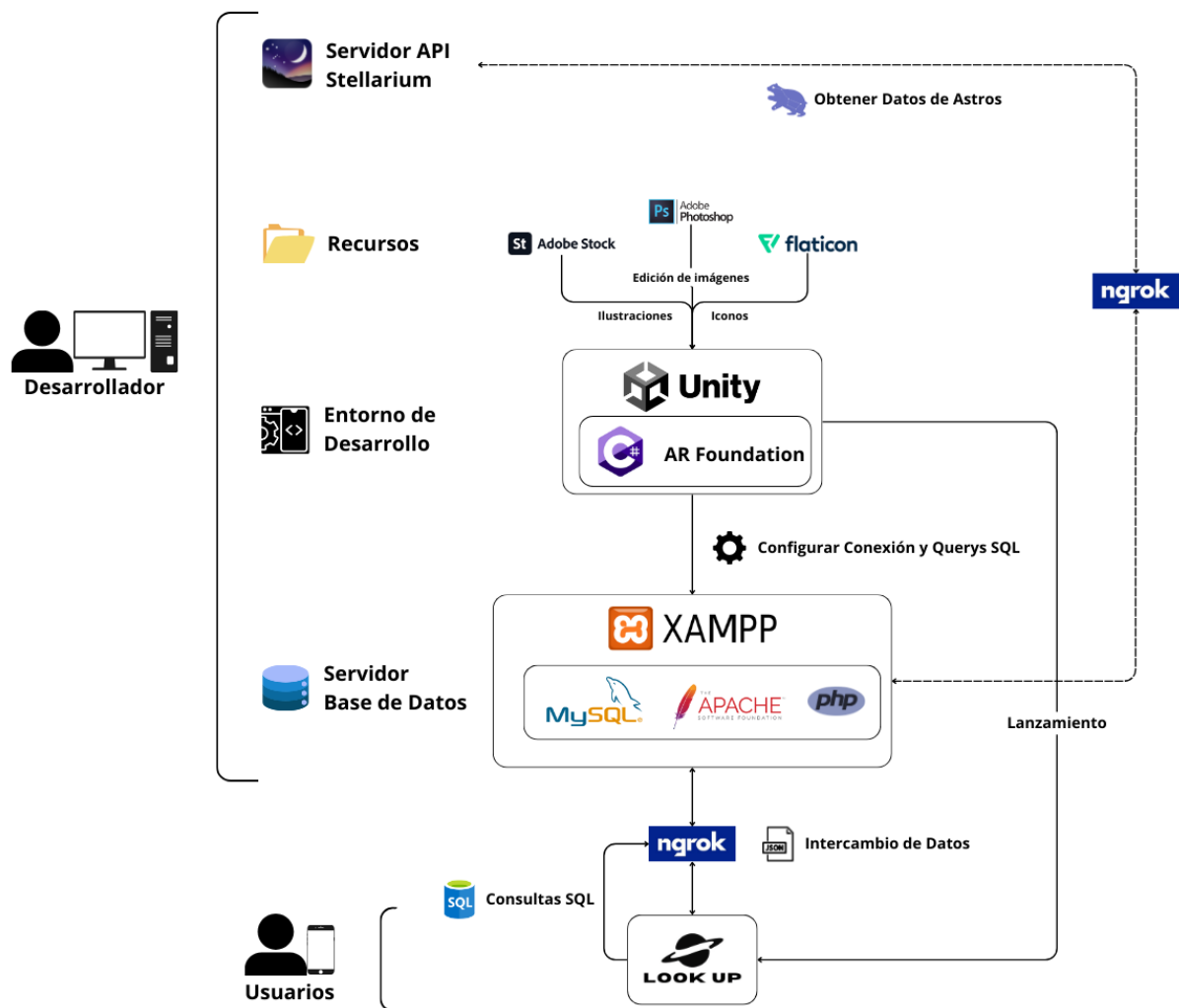


Figura 25: Arquitectura del desarrollo

9.5.1 Recursos

Para la creación de la aplicación se utilizan diversas herramientas y plataformas que proporcionan los recursos necesarios:

9.5.1.1 Adobe Stock

Fuente de imágenes y elementos gráficos de alta calidad. Concretamente se ha utilizado para descargar un conjunto de imágenes de fondo que se utilizan tanto en las pantallas de inicio de sesión,

registro, carga, biblioteca y logros. Cabe recalcar que han sido descargadas con una licencia estándar que permiten su integración en el proyecto sin problema alguno.

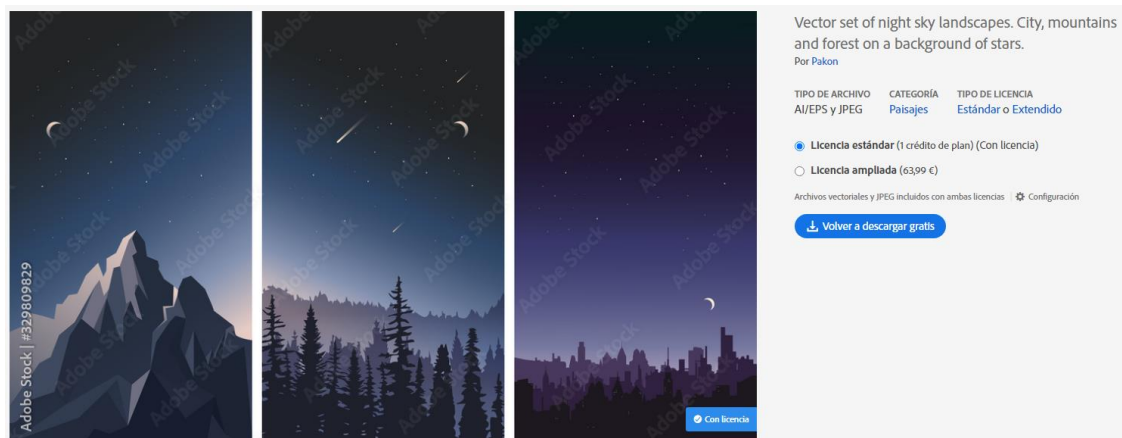


Figura 26: Adobe Stock fondos

9.5.1.2 Flaticon

Página web que proporciona íconos que se integran en la interfaz de usuario para mejorar la estética y la funcionalidad. Concretamente se ha usado en gran medida para las imágenes de los logros y los iconos de los botones de los diferentes menús.

Cabe recalcar que es importante la elección de estos iconos de tal forma que se guarde una coherencia visual. Por ejemplo, para Look Up se han buscado iconos poco detallados y sin contornos.

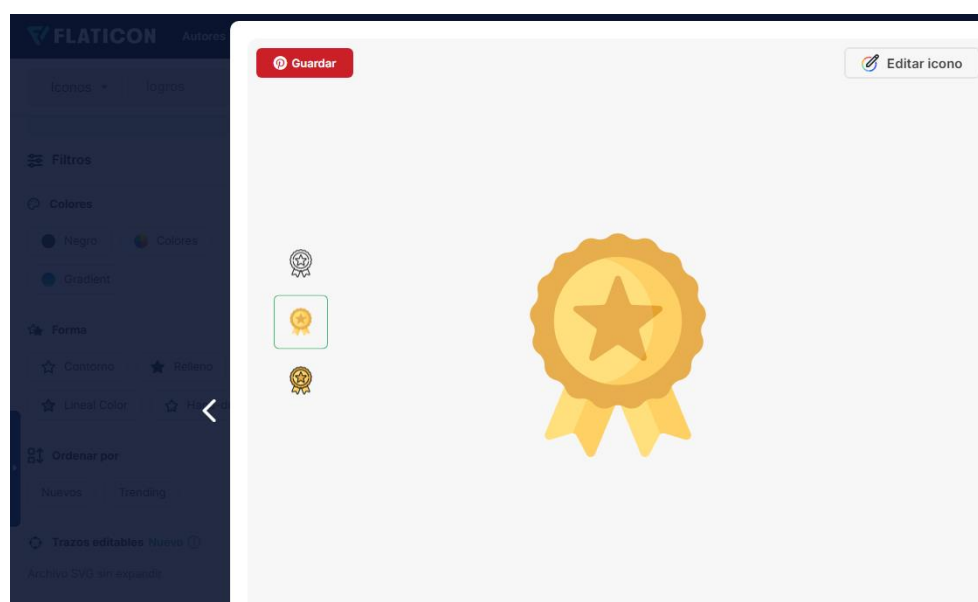


Figura 27: Flaticon icono logros

9.5.1.3 Adobe Photoshop

Utilizado para la edición de imágenes y la creación de gráficos personalizados. Esta ha sido especialmente útil para la creación de los iconos de los logros y botones.

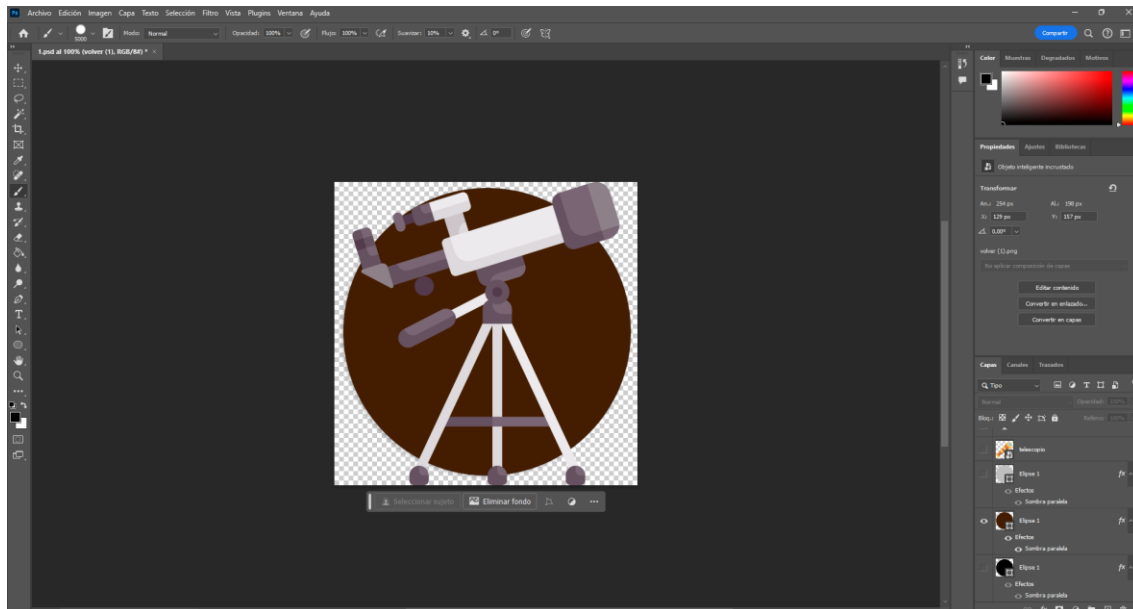


Figura 28: Adobe Photoshop para tratamiento de imágenes

Concretamente se han planteado cuatro tipos de logros en función de su dificultad: bronce, plata, oro y platino (Figura 29). Siendo los de bronce los más sencillos de obtener y los de platino los que un mayor trabajo requieren. Cada uno de esta categoría de logros tendrá un fondo de un color en concreto, de tal forma que todos los logros de plata tendrán el mismo fondo, los de oro igual, etc. Esta decisión, junto con la de elegir iconos con diseños parecidos, crearán una armonía visual destacable en el proyecto.

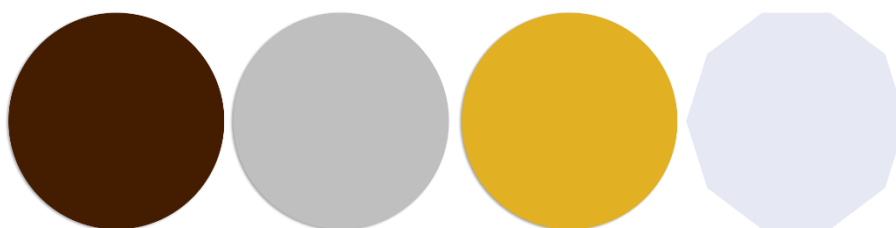


Figura 29: Tipos de logros

Estos recursos son esenciales para el desarrollo de ilustraciones y gráficos que se utilizarán dentro de la aplicación, asegurando un diseño atractivo y coherente.

9.5.2 Entorno de Desarrollo

El entorno de desarrollo como se ha explicado en múltiples ocasiones es Unity. En este IDE se utiliza **C#** como lenguaje de programación. Asimismo, tendremos acceso al paquete **AR Foundation** que nos proporcionará las herramientas necesarias para tratar con la realidad aumentada.

9.5.2.1 Conceptos Básicos Unity

Llevar a cabo una correcta explicación del diseño del proyecto a un menor nivel de abstracción es un ejercicio muy difícil sin antes dar una pequeña introducción a nuestro entorno de desarrollo. A parte, al hablar de una tecnología concreta es normal tener que recurrir a abundantes tecnicismos para poder explicar ciertos funcionamientos de la forma más fiel posible. Entonces, se haya la necesidad de explicar las bases de este motor de videojuegos.

Para empezar, los proyectos se dividen por **escenas**, las cuales se componen por **objetos** y estos últimos son constituidos por **componentes** [12]. Por ejemplo, en la Figura 30 se muestra la escena *LogIn*, en la cual se gestiona la totalidad, frontend y *backend* (parte lógica de un sitio, esta se encarga de la lógica de negocio, de recibir y devolver datos procesados a las apps y sitios web), de la vista de inicio de sesión, presentada anteriormente en el diseño de la interfaz. **LogIn** está compuesta por varios objetos como *Main Camera*, *Canvas* o *EventSystem*. Por último, si examinamos cualquiera de estos objetos, *InputField_Correo* en este caso, accedemos a sus componentes.

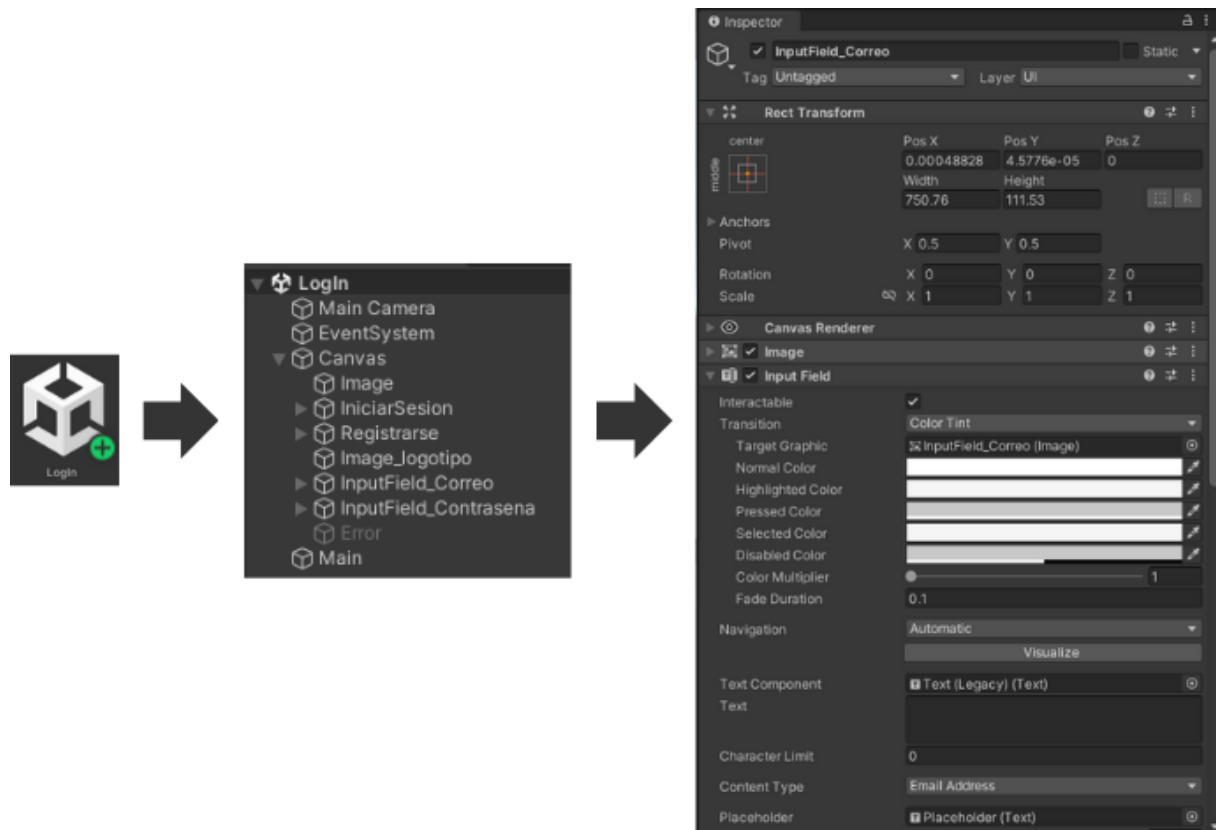
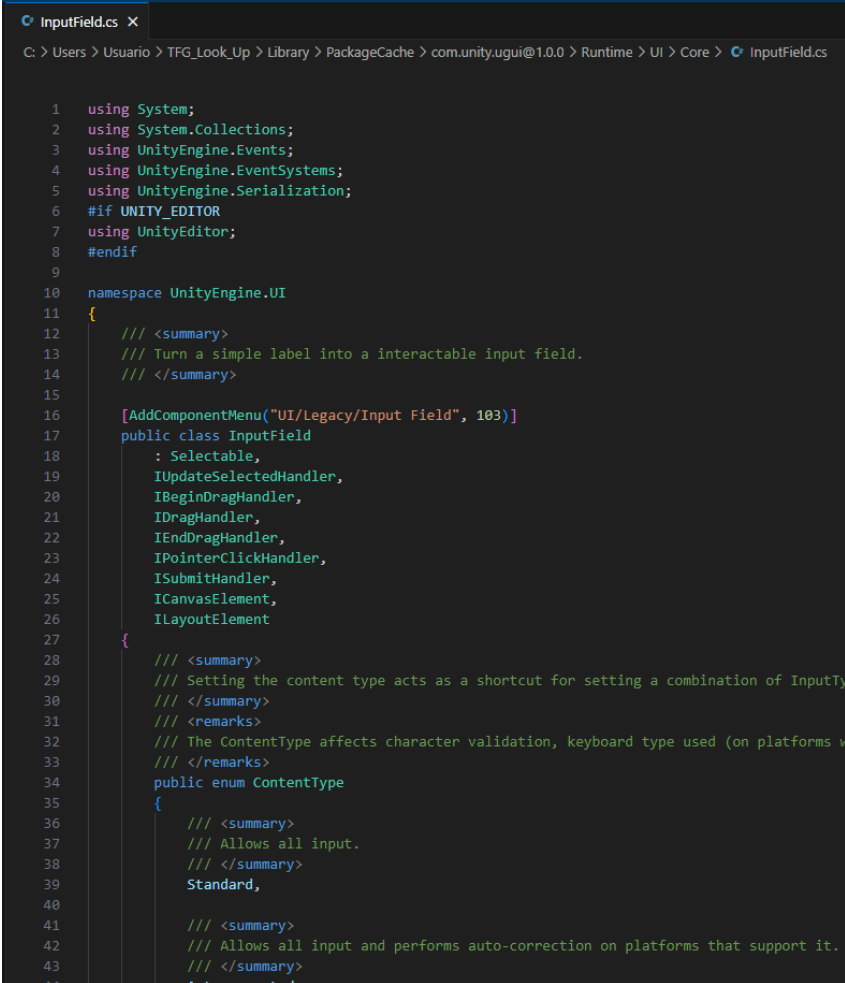


Figura 30: Elementos principales Unity

Estos componentes son fundamentalmente scripts de C#. La interfaz nos brinda la posibilidad de cambiar el valor de ciertas variables, esto es dado que han sido declaradas como públicas o, en su defecto, como SerializedField en sus respectivos ficheros de código. Así se puede observar en la Figura 31 como el campo *Content Type* del objeto Input Field ha sido declarado como “public enum”.

Cabe mencionar que algunos componentes como Main Camera, Directional Light o EventSystem serán comunes entre todas las escenas de este proyecto. Estos indican el punto de vista del usuario conforme al entorno del Unity, la fuente de luz que afectará a los objetos visibles y un controlador de eventos, como podría ser un toque en la pantalla, respectivamente.



```

1  using System;
2  using System.Collections;
3  using UnityEngine.Events;
4  using UnityEngine.EventSystems;
5  using UnityEngine.Serialization;
6  #if UNITY_EDITOR
7  using UnityEditor;
8  #endif
9
10 namespace UnityEngine.UI
11 {
12     /// <summary>
13     /// Turn a simple label into a interactable input field.
14     /// </summary>
15
16     [AddComponentMenu("UI/Legacy/Input Field", 103)]
17     public class InputField
18         : Selectable,
19         IUpdateSelectedHandler,
20         IBeginDragHandler,
21         IDragHandler,
22         IEndDragHandler,
23         IPointerClickHandler,
24         ISubmitHandler,
25         ICanvasElement,
26         ILayoutElement
27     {
28         /// <summary>
29         /// Setting the content type acts as a shortcut for setting a combination of InputTy
30         /// </summary>
31         /// <remarks>
32         /// The ContentType affects character validation, keyboard type used (on platforms w
33         /// </remarks>
34         public enum ContentType
35         {
36             /// <summary>
37             /// Allows all input.
38             /// </summary>
39             Standard,
40
41             /// <summary>
42             /// Allows all input and performs auto-correction on platforms that support it.
43             /// </summary>
44             AutoCorrected
45         }
46     }

```

Figura 31: ContentType

Seguidamente, conociendo ya los elementos básicos de Unity podemos introducir la interfaz de un proyecto. Esta se divide en **cuatro zonas principales**, marcadas en la Figura 32. Siendo estas:

1. **Hierarchy**. Se presentan las escenas cargadas con sus correspondientes objetos.
2. **Project**. Directorios que componen el proyecto. Muchas de estas carpetas son creadas automáticamente por los paquetes utilizados en el desarrollo como *Plugins* o *XRi*.
3. **Scene**. Vista actual de la escena. Si ejecutamos la aplicación para probarla en el entorno de desarrollo, Scene se cambiará por *Game* y podremos interactuar con la aplicación.
4. **Inspector**. Componentes del objeto que estemos seleccionando. Desde esta interfaz podremos modificar valores de parámetros públicos de los objetos, como se explicó anteriormente, y añadir nuevos componentes o scripts.

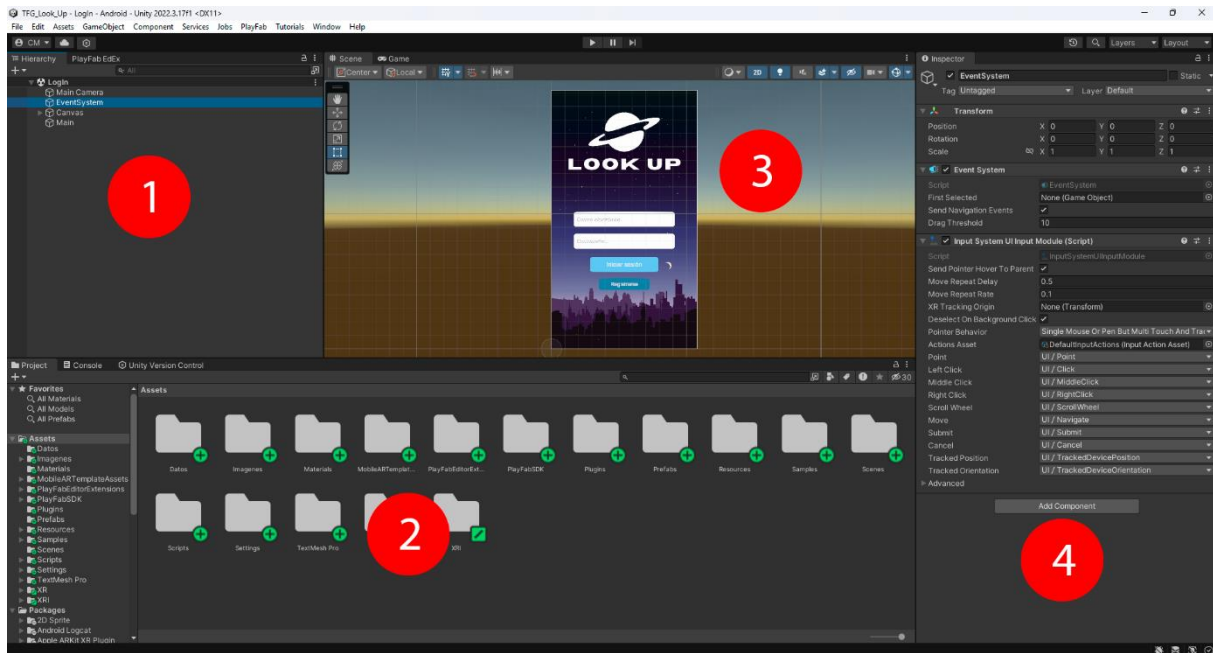


Figura 32: Interfaz Unity

Finalmente, como conceptos adicionales tenemos los **Prefabs** y **Resources**. Siendo los **Prefabs** objetos que hemos preconstruido con unos componentes concretos [15], esto nos será útil, por ejemplo, para construir todas las diferentes constelaciones a partir de un único Prefab y no tener que crear manualmente un objeto para cada una. Por otra parte, la carpeta **Resources** es muy importante pues los elementos que no estén asignados a un objeto que esté en una escena no serán accesibles una vez compilada la aplicación, a excepción de los ficheros que se encuentren en esta carpeta [16]. Estos serán accesibles mediante la clase homónima, utilizando el método “load”.

9.5.2.2 Escenas.

Una vez explicados los conceptos podemos empezar a abordar nuestro proyecto por las escenas en las que este se divide. Look Up consta de 3 escenas: LogIn, Register y Juego.

En la escena **LogIn**, a parte de los componentes comunes explicados anteriormente, podemos diferenciar:

- Canvas: Contiene todos los elementos visuales de la interfaz, asegurando que estos se adapten correctamente a diferentes resoluciones de pantalla.
 - Image: Proporciona el fondo visual de la aplicación.

- IniciarSesion y Registrarse: Son los botones principales que permiten al usuario iniciar sesión o dirigirse a la opción de registro.
 - Image_logotipo: Es el logotipo de la aplicación, que se coloca de forma prominente para reforzar la identidad visual de Look Up.
 - InputField_Correo e InputField_Contraseña: Estos son los campos donde el usuario ingresa su información de inicio de sesión. Incluyen placeholders para guiar al usuario sobre los datos que debe introducir.
 - Error: Componente que se activa cuando el usuario introduce credenciales incorrectas, mostrando un mensaje que facilita la corrección del error.
- Main: Componente que agrupa las funciones principales de la aplicación.
 - AudioManager: Este componente gestiona los efectos de sonido, como los clics en los botones o las notificaciones al usuario.
-

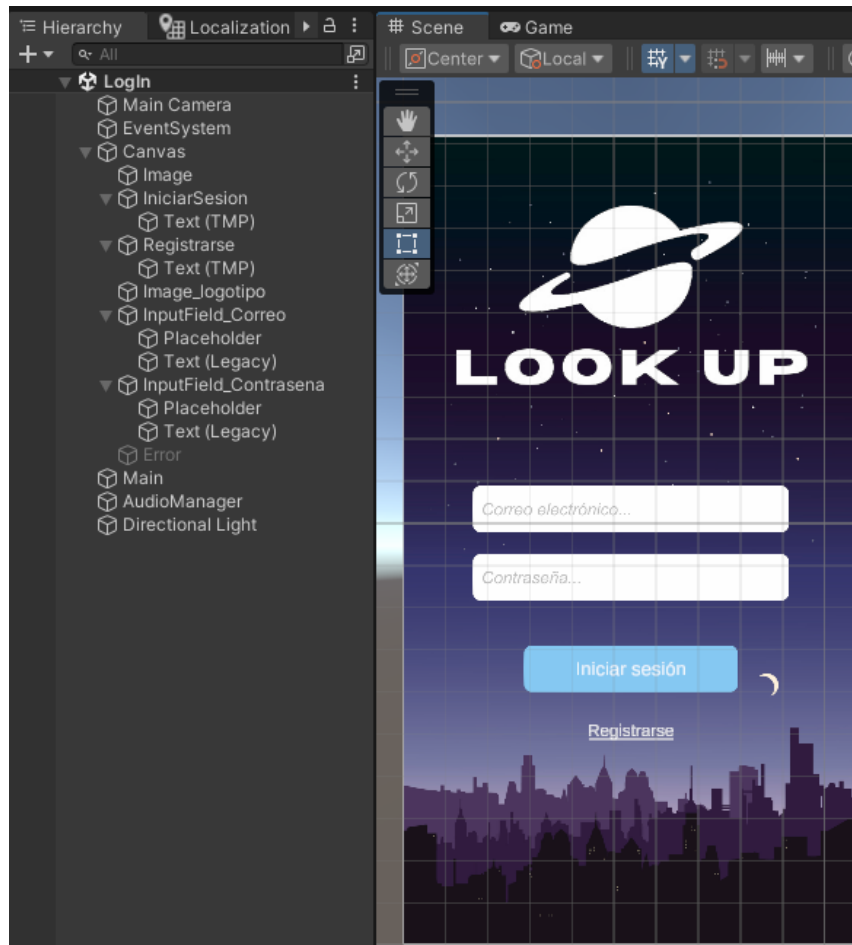


Figura 33: Escena LogIn

En cuanto a **Register** se puede discernir como es prácticamente idéntica a LogIn. Esto es porque a nivel visual simplemente cambia en el campo de “Repetir Contraseña”. Sin embargo, encontraremos los cambios en los componentes de Main que se explicarán en el punto 9.5.3 Clases.

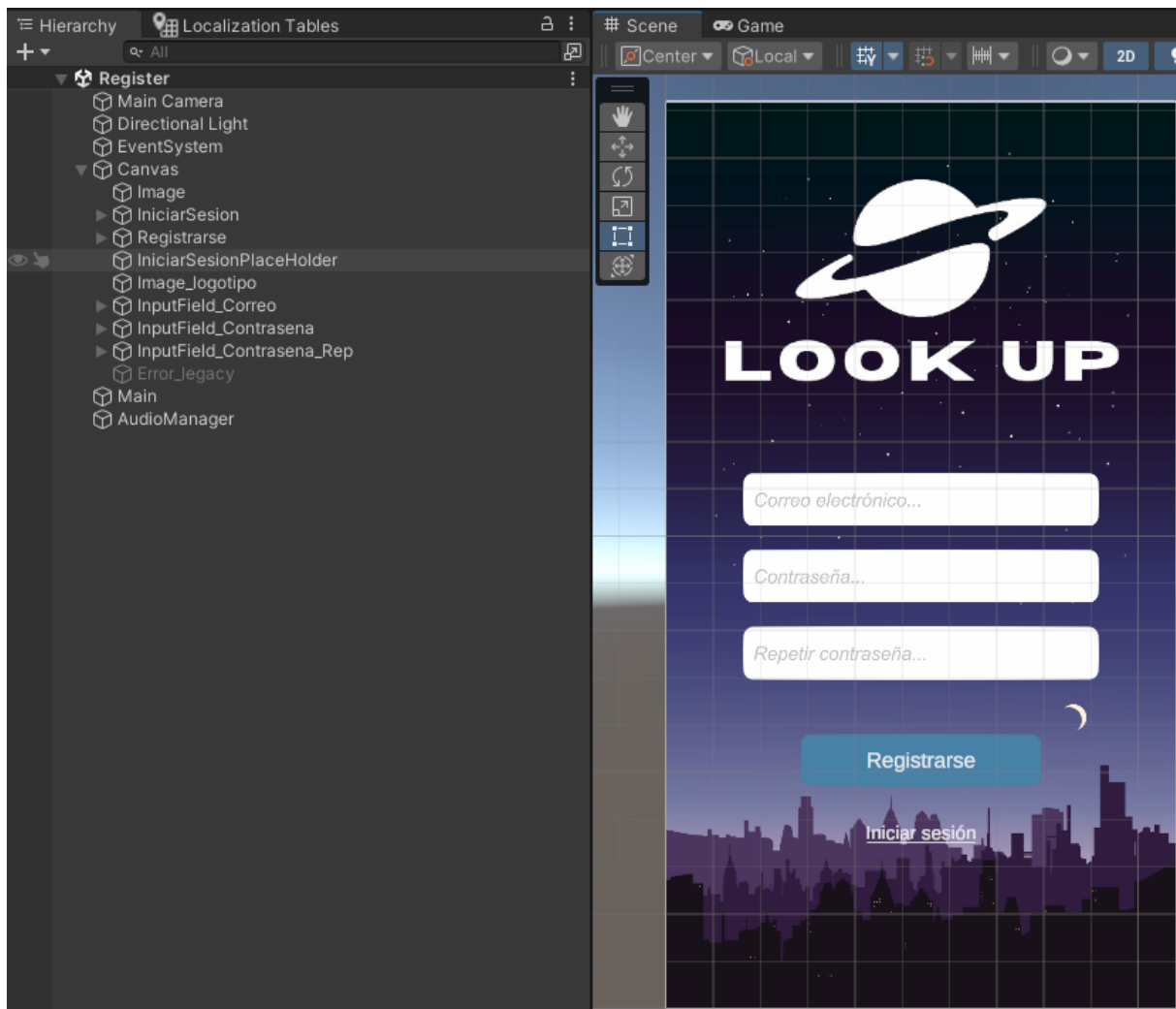


Figura 34: Escena Register

La escena **Juego**, Figura 35, es la más importante dentro de *Look Up*, ya que es donde el usuario interactúa directamente con el contenido astronómico a través de la realidad aumentada. A continuación, se detallan los componentes clave de esta escena:

- **PantallaCarga:** Esta pantalla aparece brevemente al iniciar la escena de juego, mostrando una animación de carga mientras el sistema se prepara para la experiencia AR. Se compone de una Image y un LoadPanel que cubren la interfaz principal durante el proceso de carga.
- **UIManager:** Es el encargado de gestionar los diferentes menús e interfaces con los que interactúa el usuario durante el juego. Dentro de este, se incluyen los submenús:

- **MainMenu** y **OptionsMenu**: MainMenu se refiere al botón el cual activa OptionMenu, el cual es el menú principal que permite al usuario acceder a las opciones de la aplicación, como los logros o la biblioteca de astros.
- **Biblioteca** y **Logros**: Estas secciones permiten al usuario consultar los astros descubiertos y los logros desbloqueados hasta el momento.
- **GONotificacionLogro**: Componente que muestra una notificación visual cuando el usuario desbloquea un logro.
- **AR Session** y **XR Origin (XR Rig)**: Estos son los componentes que permiten la experiencia de realidad aumentada. AR Session es responsable de gestionar las sesiones de realidad aumentada, mientras que XR Origin controla la posición del dispositivo en el entorno virtual, ajustando la cámara y los elementos visibles en función de los movimientos del usuario. Ambos trabajan en conjunto para mapear el mundo real con el entorno virtual, haciendo que los astros y constelaciones aparezcan correctamente superpuestos sobre el cielo real.
- **XR Interaction Manager**: Este componente es crucial para gestionar la interacción del usuario con los objetos en AR. Permite que el usuario "toque" los astros o constelaciones que aparecen en la pantalla, lo que desencadena eventos como el desbloqueo de logros o la aparición de información adicional.
- **Ubicación actual**: Simplemente informa al usuario de su ubicación. Es el texto que se encuentra en la esquina superior izquierda en la pantalla de juego.
- **Ajustes**: Esta sección ofrece la pantalla en la cual el usuario puede modificar valores del sistema.

Finalmente, AudioManager gestiona los sonidos de la escena, incluidos los efectos de audio cuando el usuario interactúa con los objetos o menús.

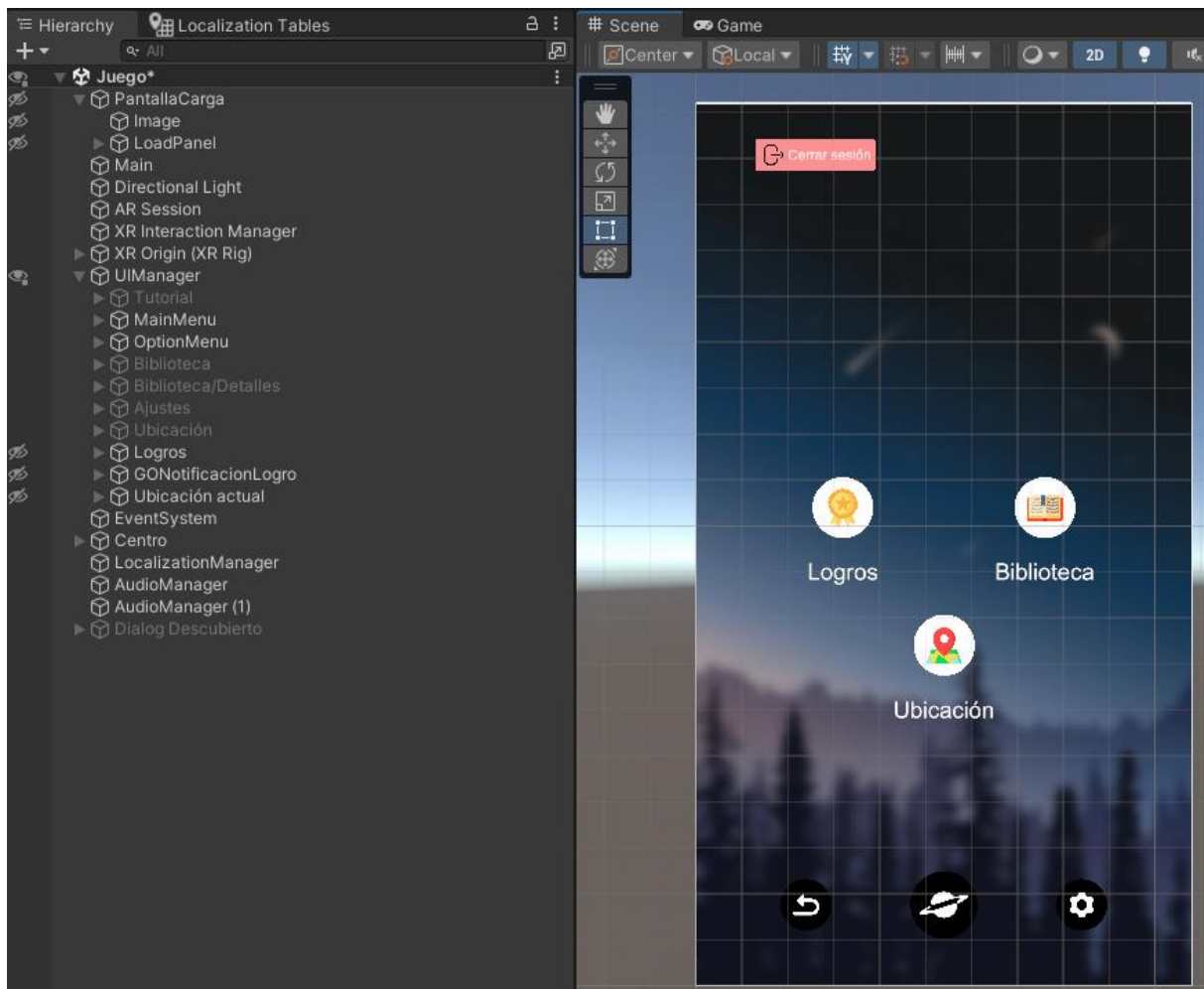


Figura 35: Escena Juego

9.5.2.3 Sistemas de Guardado de Datos.

Unity nos ofrece 3 formas de tratar con datos ya sea entre escenas o entre diferentes sesiones de juego.

Para empezar, una de las formas más simples es declarar una **clase** y sus respectivas **variables estáticas**. Sin embargo, este método solo nos servirá para mantener datos entre escenas, pero una vez se cierre la aplicación se deberán inicializar de nuevo. Mas con la cantidad de clases que se usan en este proyecto y con la existencia de datos que son importantes para varios objetos, tener una clase estática nos será de mucha utilidad. Esta se explicará en profundidad en el punto 9.5.3 Clases.

Un modo específico para guardar datos de manera persistente, es decir, que se mantengan entre diferentes sesiones de juego, es mediante el uso de **PlayerPrefs**. PlayerPrefs es una herramienta

proporcionada por Unity que permite almacenar y recuperar datos sencillos en el dispositivo del usuario, como enteros, flotantes y cadenas de texto [17]. Estos datos se almacenan en el sistema operativo del dispositivo y persisten incluso cuando la aplicación se cierra. En Android, concretamente se almacenarán en la siguiente ruta: “/data/data/pkg-name/shared_prefs/pkg-name.v2.playerprefs.xml”.

En Look Up, PlayerPrefs se utiliza para almacenar información relacionada con la configuración del usuario y algunas variables que deben permanecer constantes entre sesiones, como: volumen de efectos de sonido, volumen de música, opacidad del plano del cielo e idioma.

Se puede observar cómo es información que no llega a ser tan esencial e imprescindible como podría ser el progreso del jugador. Hay que tener en cuenta que si desinstalamos la aplicación de nuestro dispositivo también perderemos nuestros PlayerPrefs, pero perder esta información no nos supondría un gran conveniente. Simplemente tendríamos que volver a entrar a los ajustes y ponerlos a nuestro gusto de nuevo. Esta es la razón por la que estos datos se guardan mediante PlayerPrefs.

Por último, la opción más completa y robusta sería usar una **base de datos**. Esta la he usado para guardar los datos de juego del jugador, descubrimientos y logros. A parte de, obviamente, el correo y contraseña del usuario para que pueda iniciar sesión y acceder a sus datos. Se abordará la implementación de esta en el punto 9.5.4 Servidor de Base de Datos.

En resumen:

- Clase estática: usada para valores de ajustes. Datos entre escenas y clases.
 - PlayerPrefs: usada para valores de ajustes. Datos entre sesiones de juego en el mismo dispositivo.
 - Base de datos: usada para progreso del jugador. Datos entre dispositivos y persistentes en el servidor.
-

9.5.3 Clases

En este punto abordaré todas las clases creadas por mí para el correcto funcionamiento del proyecto.

9.5.3.1 Estructuras

El fichero Estructuras define varias estructuras de datos que se utilizan para almacenar el progreso del jugador, logros y objetos celestes.

Variables:

- infoLogros, PlayerLogros: Estructuras que almacenan información sobre los logros del jugador.
- infoCuerpoProgreso, PlayerProgress: Estructuras que almacenan el progreso del jugador con relación a las constelaciones descubiertas.
- CelestialData, ConstellationDataList: Estructuras que contienen los datos de los objetos celestes, como su tipo y posición.
- CiudadData: Estructura que contiene información sobre ciudades, incluyendo nombre, país, latitud y longitud.

Funciones: La clase no contiene funciones, ya que su objetivo es definir la estructura de los datos utilizados en otras partes de la aplicación.

9.5.3.2 Clase Register

La clase Register se encarga del proceso de registro de nuevos usuarios en la aplicación. Esta clase interactúa con la interfaz de usuario, recopilando la información ingresada por el usuario, y luego envía los datos a través de una solicitud web al servidor para crear una nueva cuenta.

Variables:

- UsernameInput, PasswordInput, PasswordRepeatInput: Son los campos de entrada donde el usuario introduce su nombre de usuario, contraseña y la repetición de la contraseña para confirmar que coincidan.
- RegisterButton: Botón que el usuario presiona para enviar la solicitud de registro.

Funciones:

- Start: Se encarga de añadir un listener al botón de registro. Cuando el botón es presionado, llama a una corrutina que gestiona el proceso de registro, validando los campos de entrada y comunicándose con el servidor.

9.5.3.3 Clase Login

La clase Login se encarga del proceso de autenticación de los usuarios en la aplicación. Recoge las credenciales ingresadas, las envía al servidor para validarlas, y gestiona la respuesta correspondiente, permitiendo o denegando el acceso al sistema.

Variables:

- UsernameInput, PasswordInput: Campos de entrada donde el usuario introduce su nombre de usuario y contraseña.
- LoginButton: Botón que el usuario presiona para iniciar el proceso de inicio de sesión.

Funciones:

- Start: Añade un listener al botón de inicio de sesión. Al hacer clic, ejecuta una corrutina que envía las credenciales al servidor y espera la respuesta. Si las credenciales son correctas, carga la escena de juego. Si no, muestra un mensaje de error.

9.5.3.4 Clase Main

La clase Main sirve como controlador principal para la aplicación. Actúa como un singleton, permitiendo que sus instancias sean accesibles desde cualquier otra clase. Esta clase también gestiona

el control del volumen del audio y la inicialización de otros componentes clave como las funciones web.

Variables:

- Instance: Implementación del patrón Singleton, permitiendo que esta clase sea accesible globalmente.
- myAudioSource: Controla el volumen de los efectos de sonido en la aplicación.
- Web: Instancia de la clase Web, que se encarga de gestionar todas las comunicaciones con el servidor.

Funciones:

- Start: Inicializa la instancia de la clase y configura el volumen de los efectos de sonido utilizando los valores almacenados en PlayerPrefs. También asigna la referencia para la clase Web.

9.5.3.5 Clase Web

La clase Web se encarga de todas las interacciones entre la aplicación y el servidor. Desde la gestión de login, registro, hasta la actualización y obtención de los datos de progreso del jugador, esta clase maneja las solicitudes y respuestas HTTP, asegurando la correcta comunicación con la base de datos remota.

Variables:

- feedbackLogin, errorTextPanel, iniciarSesion, registrarse, errorText: Son elementos de la interfaz que se activan o desactivan para mostrar retroalimentación al usuario durante el proceso de inicio de sesión o registro, como errores o confirmaciones.

Funciones:

- **GetGameData:** Envía una solicitud al servidor para obtener los datos del jugador (progreso, astros descubiertos, etc.) y los guarda en un archivo JSON en el dispositivo local.
- **Login:** Envía las credenciales del usuario al servidor para autenticarlas. Si son correctas, carga la escena de juego y recupera los datos del jugador.
- **Register:** Valida las credenciales ingresadas por el usuario (nombre de usuario, contraseña y confirmación) y las envía al servidor para registrar una nueva cuenta. Si el registro es exitoso, también inicia sesión automáticamente.
- **DatosEstrellas:** Solicita información relacionada con los astros al servidor.
- **UpdateGameData:** Envía al servidor los datos de progreso actualizados del jugador, como los astros descubiertos.
- **UpdateLogroProgreso:** Actualiza el progreso de los logros del jugador tanto en el archivo local como en la base de datos remota.

9.5.3.6 Clase MainManager

La clase **MainManager** es responsable de gestionar los componentes principales de la aplicación, como la biblioteca de astros, los logros, la ubicación y los objetos celestes en el cielo. Centraliza las operaciones que manejan la descarga de datos del servidor y la creación de paneles de información para el jugador.

Variables:

- **bibliotecaManager, ubicacionManager, logrosManager, cieloManager, stellariumManager:** Managers responsables de gestionar las diferentes áreas del juego (biblioteca de astros, ubicación, logros, cielo y conexión con Stellarium).
 - **celestialObjects:** Lista que contiene información sobre los objetos celestes.
-

- tutorial: Referencia al objeto que gestiona el tutorial, que se activa o desactiva según las preferencias del jugador.

Funciones:

- Start: Inicializa los componentes principales, como el tutorial y los datos del jugador (progreso y logros). También configura la notificación de logros.
- Inicio: Corrutina que gestiona la descarga de los datos celestes y la creación de paneles de la biblioteca, logros y ciudades, así como la representación de los objetos del cielo.

9.5.3.7 Clase LocaleSelector

La clase LocaleSelector se encarga de gestionar el cambio de idioma en la aplicación. Permite modificar el idioma de la interfaz y guarda la selección en las preferencias del usuario.

Variables:

- active: Booleano que impide que se realicen varios cambios de idioma simultáneamente.

Funciones:

- Start: Carga el idioma guardado en PlayerPrefs al iniciar la aplicación.
- ChangeLocale: Cambia el idioma de la aplicación según el identificador proporcionado, guardándolo en PlayerPrefs.
- SetLocale: Corrutina que aplica el cambio de idioma de manera asíncrona utilizando el sistema de localización de Unity.

9.5.3.8 Clase FondoSettings

La clase FondoSettings gestiona las configuraciones del fondo de la aplicación, permitiendo al usuario modificar la transparencia de este. También guarda y recupera las preferencias del usuario.

Variables:

- fondoMaterial: Material asignado al fondo de la escena.
-

- `translucencySlider`: Slider que permite al usuario ajustar la transparencia del fondo.
- `handleImage`: Imagen que muestra visualmente el estado de la transparencia.

Funciones:

- `Start`: Inicializa el nivel de transparencia del fondo utilizando los valores guardados en `PlayerPrefs`.
- `SetTranslucencyLevel`: Ajusta la transparencia del material del fondo según el valor del slider.
- `GuardarPreferencias`: Guarda el nivel de transparencia actual en `PlayerPrefs`.
- `UpdateHandleImage`: Actualiza la imagen de referencia del slider según el nivel de transparencia seleccionado.

9.5.3.9 Clase AprendizajeManager

La clase `AprendizajeManager` gestiona la apertura de enlaces educativos externos, como videos de YouTube, desde la aplicación.

Variables: La clase no contiene variables adicionales.

Funciones:

- `OpenLink`: Abre un enlace en el navegador del sistema si la URL proporcionada es válida.

9.5.3.10 Clase Miscelanea

La clase `Miscelanea` contiene utilidades que manejan la carga y actualización del progreso del jugador y la gestión de los datos de los astros descubiertos y logros.

Variables:

- Las principales variables que utiliza son externas, almacenadas en `DatosEntreEscenas`.

Funciones:

- LoadPlayerProgress: Carga el progreso del jugador desde el archivo local.
- LoadPlayerLogros: Carga los logros del jugador desde el archivo local.
- UpdatePlayerProgress: Actualiza el archivo local con los nuevos astros descubiertos.
- CombinePlayerData: Combina los datos de logros y astros descubiertos en un solo archivo JSON.
- UpdateLogroProgreso: Actualiza el progreso de un logro y comprueba si ha sido completado, guardando los datos en el archivo local.

9.5.3.11 Clase UbicacionManager

La clase UbicacionManager gestiona la selección y cambio de ubicación del usuario dentro de la aplicación. Se encarga de mostrar una lista de ciudades, permitir búsquedas, y actualizar los datos de ubicación seleccionados.

Variables:

- botonSi, botonNo: Botones para confirmar o cancelar la selección de ubicación.
 - textoPrefabCiudades, parentElementCiudades: Prefab y elemento padre donde se crean los paneles de las ciudades.
 - ubicacionText, textoPlaceholder: Elementos de la interfaz para mostrar la ubicación seleccionada y un marcador de posición.
 - listaCiudades: Lista que almacena los datos de las ciudades cargadas desde un archivo CSV.
 - ciudadBuscar, panelMsgConfirmacion, textoConfirmacion: Componentes utilizados para la búsqueda de ciudades y la confirmación de selección de una ciudad.
 - nameObjetivo, longitudObjetivo, latitudObjetivo: Variables que almacenan la ubicación objetivo seleccionada por el usuario.
-

Funciones:

- Start: Inicializa la interfaz de la ubicación con los datos actuales del usuario.
- CreatePanelsCiudades: Corrutina que carga los datos de las ciudades desde un archivo CSV y crea los paneles de selección de ciudades.
- BuscarCiudad: Filtra la lista de ciudades según el texto ingresado en el campo de búsqueda y muestra los resultados.
- CambiarDatosUbicacion: Actualiza los datos de ubicación seleccionados por el usuario y los refleja en la interfaz.

9.5.3.12 Clase StellariumManager

La clase StellariumManager se encarga de obtener los datos astronómicos del servidor y transformarlos en objetos celestes que se pueden mostrar en el juego.

Variables: No contiene variables adicionales relevantes.

Funciones:

- FetchCelestialData: Corrutina que envía una solicitud al servidor para obtener los datos de los astros visibles desde la ubicación actual del usuario. Los datos recibidos se procesan y se transforman en objetos que se pueden representar en Unity.

9.5.3.13 Clase SceneLoadManager

La clase SceneLoadManager gestiona la carga de escenas, mostrando una barra de progreso mientras se cargan datos importantes en segundo plano.

Variables:

- loadbar: Barra de progreso visual que muestra el avance de la carga.
 - loadPanel: Panel que contiene la barra de progreso.
 - mainManager: Referencia al MainManager para iniciar el proceso de carga de datos.
-

- `canvasToDisable`: Canvas que se desactiva una vez que se completa la carga.

Funciones:

- `Start`: Inicia la corrutina que gestiona el proceso de carga de datos.
- `LoadStellariumAPI`: Corrutina que simula la carga progresiva hasta el 50%, y luego ejecuta la función de inicio del `MainManager` para completar el proceso.

9.5.3.14 Clase ProgresoLogrosManager

La clase `ProgresoLogrosManager` gestiona el progreso y la visualización de los logros dentro del juego. Actualiza el progreso de cada logro y muestra notificaciones cuando se alcanza un logro.

Variables:

- `logrosObjeto`, `panelPrefabBiblioteca`, `parentElementBiblioteca`: Componentes que gestionan la visualización de los logros en la interfaz de usuario.
- `webScript`: Referencia al componente Web, que se utiliza para actualizar los logros en el servidor.
- `valoresParaConseguirLogro`: Diccionario que contiene los valores necesarios para conseguir cada logro.

Funciones:

- `Start`: Inicializa las referencias a los objetos relacionados con los logros en la interfaz.
 - `IncrementarProgreso`: Actualiza el progreso de un logro y, si se completa, muestra la notificación correspondiente.
 - `NotificarLogro`: Activa la notificación visual de un logro conseguido.
 - `UpdateNotificacionLogro`: Actualiza el panel de notificación con la información del logro conseguido.
-

- `CreatePanelsLogroNuevo`: Crea un nuevo panel en la interfaz de logros para mostrar el logro recién conseguido.

9.5.3.15 Clase `CieloManager`

La clase `CieloManager` gestiona la representación y la interacción de los objetos celestes (constelaciones y planetas) en la escena del cielo. Permite la creación de objetos 2D y 3D en la escena, además de gestionar la lógica relacionada con el descubrimiento de astros por parte del jugador.

Variables:

- `constelacionPrefab`, `PlanetaPrefab`: Prefabs utilizados para representar constelaciones y planetas en la escena.
- `parentElementConstelacion`: Elemento padre bajo el cual se crean las constelaciones y planetas.
- `webScript`: Referencia a la clase `Web`, que gestiona las interacciones con el servidor.
- `AudioSuccess`, `AudioYaDescubierto`: Sonidos que se reproducen al descubrir un astro o si ya ha sido descubierto.
- `dialogoDescubierto`: Diálogo que aparece cuando un astro ya ha sido descubierto.
- `canvasGroup`: Controla la visibilidad del diálogo.

Funciones:

- `Start`: Inicializa el `canvasGroup` para el diálogo de descubrimiento.
 - `ShowDialogo`: Muestra el diálogo de descubrimiento por unos segundos y luego lo oculta con una animación de desvanecimiento.
 - `CreateObjetosCielo`: Corrutina que crea objetos celestes (constelaciones y planetas) en la escena, asignándoles sus respectivas posiciones y sprites.
 - `UpdateGameData`: Se llama cuando se descubre un astro y se actualizan los datos del jugador en el servidor, además de reproducir el sonido correspondiente y actualizar la interfaz de logros.
-

9.5.3.16 Clase BibliotecaManager

La clase BibliotecaManager gestiona la biblioteca de astros descubiertos por el jugador. Se encarga de crear los paneles correspondientes para cada astro en la interfaz y mostrar detalles adicionales cuando se seleccionan.

Variables:

- titleText, descriptionText, fechaYDatosText: Campos de texto que muestran la información del astro seleccionado.
- content: Contenedor que organiza los elementos visuales de la interfaz.
- rawImageComponent: Componente que muestra la imagen del astro seleccionado.
- bibliotecaObject, bibliotecaDetallesObject: Objetos que representan la interfaz principal de la biblioteca y los detalles de los astros.
- efectoSonido: Sonido que se reproduce al seleccionar un astro.
- panelPrefabBiblioteca: Prefab utilizado para crear los paneles de astros descubiertos.
- parentElementBiblioteca: Elemento padre bajo el cual se crean los paneles de astros.

Funciones:

- Start: Inicializa los componentes de texto que mostrarán la información de los astros.
 - CreatePanelsBiblioteca: Corrutina que recorre la lista de constelaciones descubiertas y crea un panel para cada una en la interfaz.
 - CrearPanelIndividual: Crea un panel individual para un astro descubierto, asignando su nombre e imagen y configurando un botón para mostrar los detalles del astro seleccionado.
 - SetContent: Establece la información de un astro en la interfaz de detalles, incluyendo su nombre, ubicación, fecha, y posición en el cielo.
-

9.5.3.17 Clase LogrosManager

La clase LogrosManager se encarga de gestionar los logros del jugador. Muestra los logros conseguidos en la interfaz y permite su localización y visualización mediante la integración con el sistema de localización de Unity.

Variables:

- panelPrefabBiblioteca: Prefab utilizado para crear los paneles de logros.
- parentElementBiblioteca: Elemento padre bajo el cual se crean los paneles de logros.

Funciones:

- CreatePanelsLogros: Corrutina que recorre la lista de logros del jugador y crea un panel para cada logro conseguido. Asigna los textos de localización y las imágenes correspondientes, eliminando los botones si no son necesarios.

9.5.3.18 Clase CoordsManager

La clase CoordsManager es responsable de convertir las coordenadas astronómicas (azimuth y altitud) en coordenadas cartesianas para posicionar los objetos celestes en la escena.

Variables: No contiene variables adicionales relevantes.

Funciones:

- EquatorialToCartesian: Convierte las coordenadas de azimuth y altitud en coordenadas cartesianas para posicionar los objetos en el espacio 3D.

9.5.3.19 Clase DatosEntreEscenas

La clase DatosEntreEscenas es una clase estática que se utiliza para almacenar y transferir datos importantes entre las diferentes escenas de la aplicación, como se explicó anteriormente. Almacena información relacionada con el estado del jugador, la ubicación y otros datos persistentes.

Variables:

- correo, pathDatosUsuario, urlToLocalHost, ubicacion: Almacenan información del usuario, la ubicación y el servidor.
- longitud, latitud: Coordenadas geográficas del usuario.
- horaDescargaDatos: Almacena la hora en que se descargaron los datos astronómicos.
- logged: Bandera que indica si el usuario ha iniciado sesión.
- biblioLogro, constDescLogro, generalLogro, planetLogro, ubiLogro: Almacenan el progreso de logros del usuario.
- playerProgress, playerLogros: Estructuras que contienen los datos del progreso del jugador y sus logros.
- logros, notificacionLogros: Referencias a objetos de la interfaz para mostrar logros.
- conseguido: Bandera que indica si un logro ha sido conseguido.

Funciones: No contiene funciones, ya que es una clase estática diseñada para almacenar datos globales.

9.5.3.20 Clase Navigator

La clase Navigator se encarga de gestionar la navegación entre diferentes escenas de la aplicación.

Variables: No contiene variables adicionales.

Funciones:

- toRegister: Cambia la escena a la de registro.
- toLogIn: Cambia la escena a la de inicio de sesión.

9.5.3.21 Clase ControlVolumen

La clase ControlVolumen permite al usuario ajustar los niveles de volumen de los efectos de sonido y la música en la aplicación.

Variables:

- efectos: Indica si el control ajusta el volumen de los efectos de sonido o de la música.
- myAudioSource: Componente de audio que se controla.
- musicSlider: Slider que permite al usuario ajustar el volumen.
- handleImage: Imagen que cambia dependiendo del nivel de volumen (muted o activo).

Funciones:

- Start: Inicializa el volumen de la música o efectos según las preferencias guardadas en PlayerPrefs.
- SetMusicVolume: Ajusta el volumen de la música o efectos en tiempo real.
- GuardarPreferencias: Guarda el volumen ajustado en PlayerPrefs.
- UpdateHandleImage: Cambia la imagen del control deslizante dependiendo de si el volumen está activado o muteado.

9.5.4 Servidor de Base de Datos

El servidor de base de datos es un componente crucial para el almacenamiento y gestión de los datos de la aplicación. En este proyecto se utilizará **XAMPP** para la configuración de un servidor local con **Apache** que nos sirva para hospedar la base de datos.

Para poder acceder a la base de datos desde cualquier red se ha usado **ngrok**. Este software nos proporciona un dominio estático gratuito que utilizaremos para que cualquiera pueda acceder a nuestro servidor Apache, en este paso hospedado en “<http://localhost:80>”.

```

ngrok

New guides https://ngrok.com/docs/guides/site-to-site-apis/

Session Status      online
Account             carloschecamoreno@gmail.com (Plan: Free)
Update              update complete, restart for new version!
Version             3.11.0
Region              Europe (eu)
Latency              50ms
Web Interface        http://127.0.0.1:4040
Forwarding           https://rested-fox-accurately.ngrok-free.app -> http://localhost:80

Connections          ttl      opn      rt1      rt5      p50      p90
                    1         0         0.00     0.00     11.38    11.38

HTTP Requests
-----

10:24:40.599 CESTPOST /lookupbbdd/DatosEstrellas.php 200 OK

```

Figura 36: Ngrok en funcionamiento

9.5.5 Stellarium API

Para acceder a la API de Stellarium deberemos instalar este software de forma local en nuestro dispositivo. Una vez instalada podemos activarla usando el plugin “Control Remoto” [19] el cual abrirá la API en nuestra red local al puerto 8090.

En este caso no será necesario acceder a este puerto desde cualquier red, pues los usuarios interactuarán con la API de manera involuntaria al hacer peticiones al servidor Apache. Este último servidor será el que accederá a la API de Stellarium desde su misma red local.

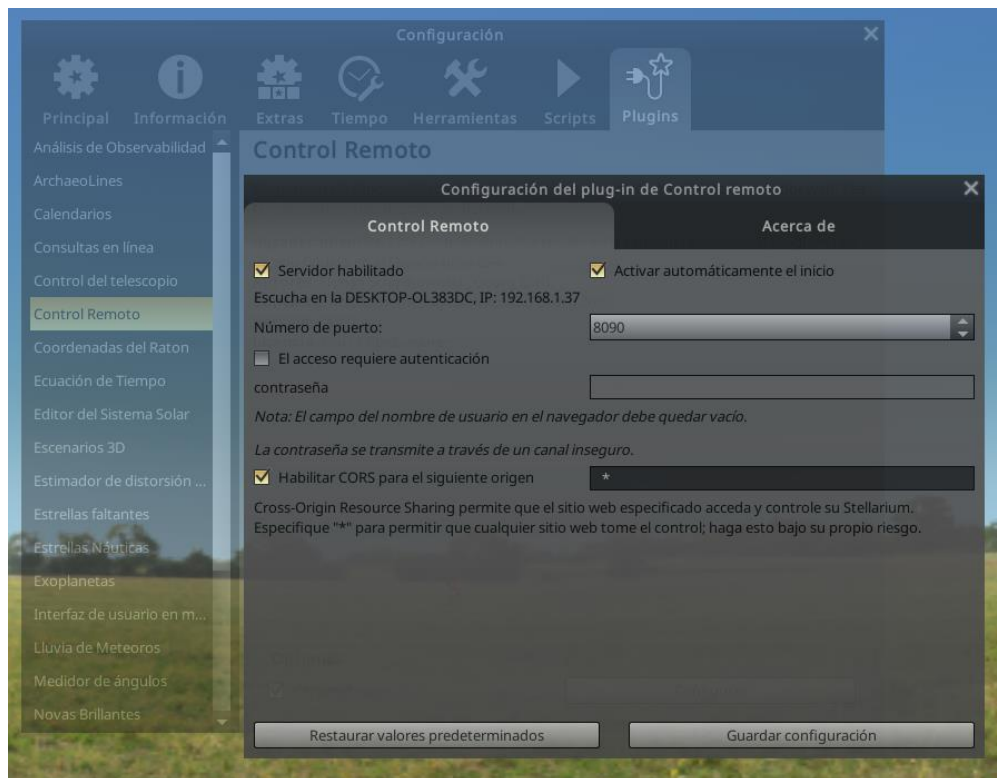
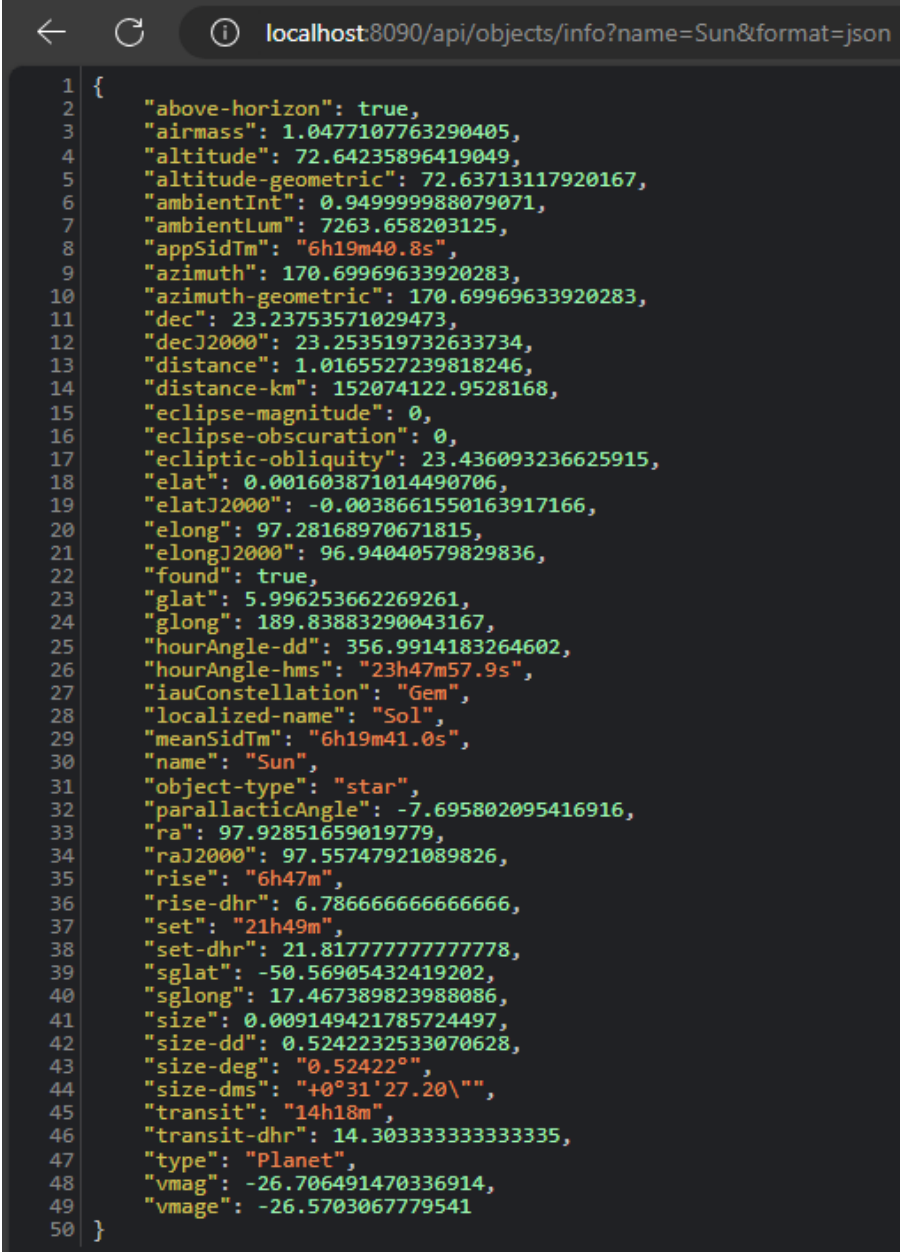


Figura 37: Plugin Control Remoto Stellarium

Esta API nos proporciona una gran cantidad de información del objeto que solicitamos. Por ejemplo, para obtener información sobre el sol deberíamos acceder a la siguiente url: <http://localhost:8090/api/objects/info?name=Sun&format=json>



```

1 {
2   "above-horizon": true,
3   "airmass": 1.0477107763290405,
4   "altitude": 72.64235896419049,
5   "altitude-geometric": 72.63713117920167,
6   "ambientInt": 0.949999988079071,
7   "ambientLum": 7263.658203125,
8   "appSidTm": "6h19m40.8s",
9   "azimuth": 170.69969633920283,
10  "azimuth-geometric": 170.69969633920283,
11  "dec": 23.23753571029473,
12  "decJ2000": 23.253519732633734,
13  "distance": 1.0165527239818246,
14  "distance-km": 152074122.9528168,
15  "eclipse-magnitude": 0,
16  "eclipse-obscuration": 0,
17  "ecliptic-obliquity": 23.436093236625915,
18  "elat": 0.001603871014490706,
19  "elatJ2000": -0.0038661550163917166,
20  "elong": 97.28168970671815,
21  "elongJ2000": 96.94040579829836,
22  "found": true,
23  "glat": 5.996253662269261,
24  "glong": 189.83883290043167,
25  "hourAngle-dd": 356.9914183264602,
26  "hourAngle-hms": "23h47m57.9s",
27  "iauConstellation": "Gem",
28  "localized-name": "Sol",
29  "meanSidTm": "6h19m41.0s",
30  "name": "Sun",
31  "object-type": "star",
32  "parallacticAngle": -7.695802095416916,
33  "ra": 97.92851659019779,
34  "raJ2000": 97.55747921089826,
35  "rise": "6h47m",
36  "rise-dhr": 6.786666666666666,
37  "set": "21h49m",
38  "set-dhr": 21.817777777777778,
39  "sglat": -50.56905432419202,
40  "sglong": 17.467389823988086,
41  "size": 0.009149421785724497,
42  "size-dd": 0.5242232533070628,
43  "size-deg": "0.52422°",
44  "size-dms": "+0°31'27.20\"",
45  "transit": "14h18m",
46  "transit-dhr": 14.303333333333335,
47  "type": "Planet",
48  "vmag": -26.706491470336914,
49  "vmage": -26.5703067779541
50 }

```

Figura 38: Ejemplo llamada API para sol

Cabe recalcar que la API nos dan los datos referentes a la ubicación establecida en Stellarium. Por lo tanto, antes de pedir los datos de los diferentes objetos tendremos que cambiar la ubicación de Stellarium.

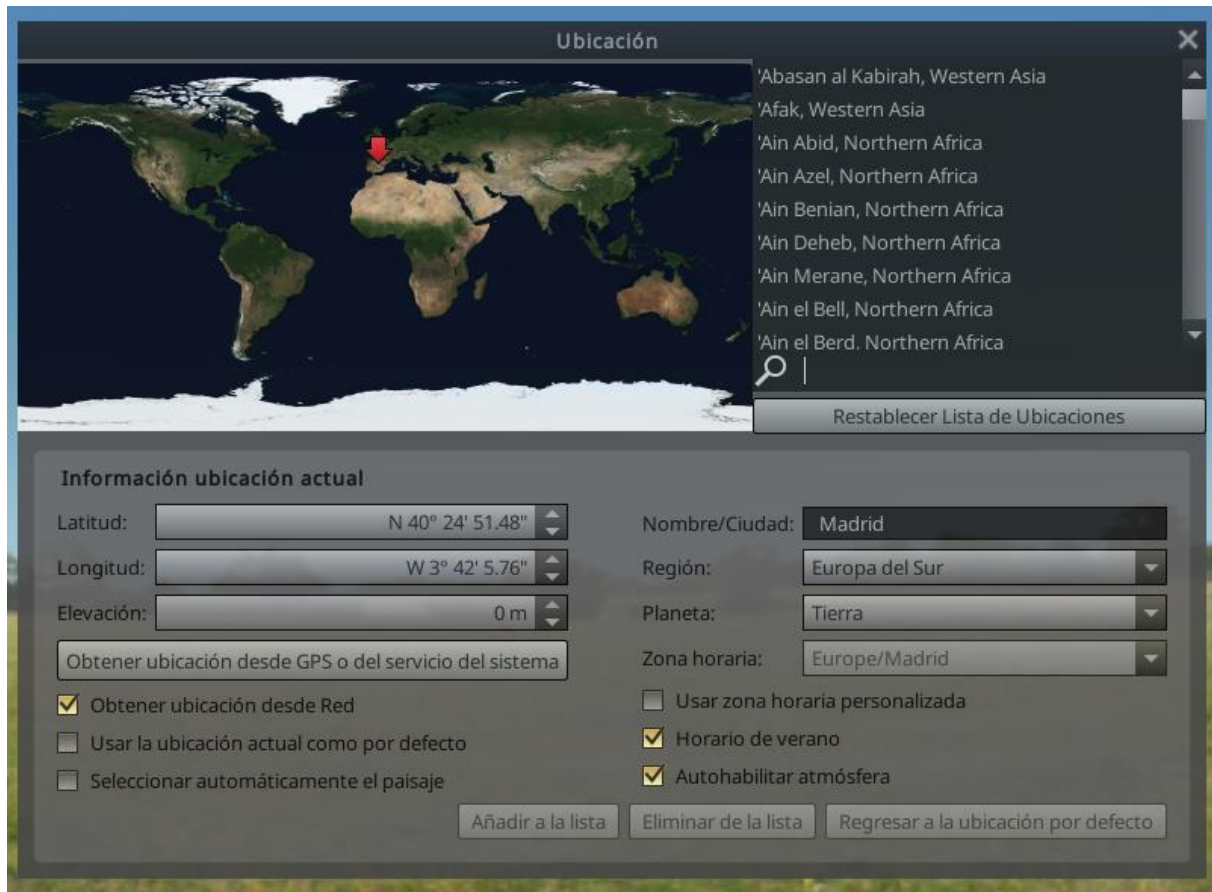


Figura 39: Información ubicación actual Stellarium

Podremos cambiar la ubicación de Stellarium de forma automática mediante la ejecución de un script de Stellarium, pues estos también pueden ser ejecutados desde la API, por ejemplo: “<http://localhost:8090/api/scripts/run?id=LookUpRequest.ssc>”. Sin embargo, será necesario tener un script que cambie la ubicación del software a la del usuario que esté realizando la petición.

Para realizar este fichero que cambie la ubicación tenemos que utilizar el lenguaje de scripting de Stellarium [20]. En este caso nos interesa la siguiente función:

```
core.setObserverLocation(
    longitude    ,
    latitude    ,
    altitude     ,
    duration     , // optional    , default = 1
    name         , // optional    , default = ""
    planet       // optional(?), default = ""
```

);

Teniendo claro estos puntos, se ha desarrollado “DatosEstrellas.php”, fichero guardado en XAMPP y accesible desde nuestro servidor. Este es el fichero encargado de devolver todos los datos de los cuerpos del cielo al usuario. Para esta tarea el usuario debe pasar como argumento su latitud y longitud. Una vez tenga los datos necesarios ejecuta los siguientes pasos:

1. Crear script de Stellarium para cambiar la posición. El contenido de este sería:
"core.setObserverLocation(\$longitud,\$latitud,0);".
 2. Ejecutar script
 3. Llamadas a API para obtener datos
 4. Devolver datos al usuario
-

Capítulo 10

PRUEBAS

Las pruebas son una fase crucial dentro del ciclo de vida del desarrollo de software, ya que permiten garantizar el correcto funcionamiento del sistema, identificando posibles fallos y optimizando su rendimiento. En el caso de Look Up, se han realizado diversos tipos de pruebas con el objetivo de asegurar que la aplicación cumpla con los requisitos establecidos. Estas pruebas se han realizado en dos enfoques principales: pruebas de caja blanca y pruebas de caja negra, además de pruebas de seguridad, como la detección de vulnerabilidades por inyecciones SQL.

10.1 Pruebas de Caja Blanca

Las **pruebas de caja blanca** se enfocan en el análisis interno del código, validando que las estructuras lógicas funcionen correctamente, y que las rutas de ejecución cubran todas las posibles ramas del programa [21]. Estas pruebas se llevaron a cabo sobre los módulos más críticos de la aplicación, como la creación de cuentas, el inicio de sesión y la visualización de astros.

Como herramientas para la realización de estas comprobaciones se ha usado la clase Debug que nos proporciona Unity, la cual nos permite depurar la aplicación con mensajes por consola del entorno de desarrollo mediante Debug.Log(), Debug.LogWarning() y Debug.LogError(); y el paquete Android Logcat que nos permite obtener los mensajes de depuración en la terminal de Unity mientras estemos usando la aplicación en nuestro móvil teniéndolo enchufado por medio de USB a nuestro ordenador.

Se ha seguido la siguiente metodología:

- **Cobertura de código:** Se revisaron las diferentes funciones del sistema para asegurar que todos los caminos de ejecución se verificasen correctamente, incluyendo la verificación de credenciales de usuario y el sistema de logros.
- **Validación de flujos lógicos:** Se realizaron pruebas exhaustivas para comprobar que cada bloque condicional (if-else) dentro de los controladores de usuario y las llamadas a la API se ejecutaran de acuerdo con los parámetros y estados recibidos.
- **Pruebas de interacción con la base de datos:** Se verificaron las conexiones y operaciones *CRUD* (Create, Read, Update, Delete) que el sistema realiza con la base de datos, prestando especial atención a la manipulación de datos de usuarios y astros descubiertos.

10.2 Prueba de Caja Negra

En las **pruebas de caja negra**, el enfoque fue probar la funcionalidad externa del sistema sin tener en cuenta la lógica interna [22]. Se llevaron a cabo simulaciones basadas en los casos de uso descritos previamente, poniendo énfasis en la interacción del usuario con la aplicación en escenarios reales.

Pruebas realizadas:

- **Registro y autenticación de usuarios:** Se simularon múltiples intentos de registro e inicio de sesión con diferentes combinaciones de credenciales, verificando tanto casos exitosos como fallidos (errores en las credenciales, cuentas inexistentes, etc.).
 - **Visualización de astros:** Se evaluó el comportamiento de la aplicación al utilizar la cámara del dispositivo para visualizar astros, verificando que las constelaciones y planetas se mostraran correctamente y se aprecian apropiadamente.
 - **Logros y progresión:** Se realizaron pruebas de desbloqueo de logros para asegurarse de que la aplicación notificase correctamente el progreso del usuario.
-

- **Interfaz de usuario:** Se realizaron pruebas de usabilidad para comprobar que todos los elementos de la interfaz fueran accesibles, funcionales y con dimensiones adecuadas a la resolución de las pantallas de los usuarios. Así como que la navegación entre los diferentes menús fuese fluida y libre de errores.
- **Sonido:** Se llevaron a cabo pruebas para comprobar que los diferentes elementos que deben reproducir sonidos lo hiciesen adecuadamente.
- **Actualización de pantallas:** Se ha comprobado que tanto la Biblioteca como los Logros se actualizan correctamente en el instante que el usuario realiza un progreso, esto es, que no le sea necesario reiniciar la aplicación para visualizar su nuevo éxito en la pantalla pertinente.
- **Datos:** Se ha realizado un arduo trabajo realizando pruebas con los diferentes sistemas de guardado de la aplicación: PlayerPrefs y base de datos. Es esencial guardar correctamente el progreso del jugador de tal forma que su información no se vea comprometida en ningún momento.

10.2.1 Caso de Prueba 1. Registro e inicio de sesión

10.2.1.1 Prueba CP1

Se llevará a cabo el registro de nuevos usuarios para probar si pueden iniciar sesión adecuadamente.

10.2.1.2 Resultado CP1

Correcto, no se han encontrado de funcionamiento. Sin embargo, la retroalimentación a la hora de registrarse no es del todo adecuada. Si todos los campos son correctos, al darle al botón Registrarse se necesitarán un par de segundos para registrarnos. Durante estos segundos no tenemos ninguna marca o señal visual que nos indique que todo va bien, lo cual puede producir confusión en el usuario.

10.2.1.3 Solución Aplicada CP2

El botón Registrarse ha sido actualizado para que no sea posible interactuar con él mientras se ejecuta la función a la que llame y, además, su color cambia a gris. Este cambio de color es esencial para que el usuario entienda visualmente que el botón no está operativo y que se están ejecutando procesos internamente.

10.2.2 Caso de Prueba 2. Visualización de astros

10.2.2.1 Prueba CP2

Comprobar que los diferentes objetos que deben verse mediante la cámara del dispositivo se aprecien correctamente.

10.2.2.2 Resultado CP2

Los objetos sí se contemplaban. Sin embargo, en ambientes muy luminosos es más complicado distinguirlos correctamente.

10.2.2.3 Solución Aplicada CP2

Se ha configurado la opción del menú de ajustes de Opacidad del Cielo. De tal forma que con esta opción podremos modificar la opacidad de un plano negro que se encuentra detrás de los objetos del cielo.

10.2.3 Caso de Prueba 3. Desbloqueo de logros

10.2.3.1 Prueba CP3

Verificar que los logros se desbloqueen correctamente al cumplir con los requisitos establecidos, y que el sistema notifique al usuario cuando esto ocurra. Concretamente se ha configurado manualmente un fichero de guardado con los datos de progreso justo para conseguir los logros rápidamente.

10.2.3.2 Resultado CP3

Correcto, los logros se desbloqueen según lo previsto, y el sistema notifica al usuario de manera adecuada.

10.2.4 Caso de Prueba 4. Navegación entre menús

10.2.4.1 Prueba CP4

Comprobar que la navegación entre los distintos menús de la aplicación es fluida, y que no se producen errores al acceder a las diferentes funcionalidades, como el cambio de ubicación o el acceso a la biblioteca de astros descubiertos.

10.2.4.2 Resultado CP4

Se detectaron un par de errores por despistes a la hora del desarrollo. En algunas pantallas el botón de volver hacia atrás o de volver al juego no estaban correctamente configurados.

10.2.4.3 Solución Aplicada CP4

Se ha corregido el funcionamiento de los diferentes botones usados para navegación, de tal forma que activen y desactiven los objetos necesarios.

10.2.5 Caso de Prueba 5. Dimensiones adaptables

10.2.5.1 Prueba CP5

Constatar que los diferentes elementos visuales se adaptan visualmente a dispositivos con pantallas con diferentes resoluciones. Durante el desarrollo se ha trabajado en un emulador con una resolución de 1920x1080px y en esta prueba se usará Look Up en un dispositivo con 2.670x1.200px.

10.2.5.2 Resultado CP5

Se han encontrado elementos que no están correctamente configurados: el texto sobresale por los lados en los diálogos de confirmación, el botón Cerrar Sesión se corta por la izquierda y la notificación de “astro ya encontrado” directamente se sale de la pantalla por ambos lados, tanto el texto como el panel de fondo.

10.2.5.3 Solución Aplicada CP5

El problema se ha solucionado aplicando la propiedad *AutoSize* a textos que no la tenían y configurando correctamente el componente *Canvas Scaler* para que escale con las resoluciones de los dispositivos.

10.2.6 Caso de Prueba 6. Guardado y recuperación de datos I

10.2.6.1 Prueba CP6

Se realizará un progreso en la aplicación descubriendo nuevas constelaciones y desbloqueando logros, asimismo, se cambiarán los ajustes. A continuación, se cerrará la aplicación completamente y se volverá a abrir iniciando sesión para comprobar si todo sigue como estaba.

10.2.6.2 Resultado CP6

Correcto, el progreso del usuario se guarda y se recupera de manera adecuada. No se encontraron problemas de pérdida de datos ni inconsistencias en las pruebas realizadas.

10.2.7 Caso de Prueba 7. Guardado y recuperación de datos II

10.2.7.1 Prueba CP7

Se realizará la Prueba CP6 con el distintivo que una vez se cierre sesión, se abrirá sesión en otro dispositivo. Especificando más, se realiza un progreso utilizando el simulador del ordenador de desarrollo y se comprobará que se ha guardado iniciando sesión desde el dispositivo móvil.

10.2.7.2 Resultado CP7

Correcto, los datos se mantienen entre diferentes dispositivos.

10.2.8 Caso de Prueba 8. Actualización en vivo

10.2.8.1 Prueba CP8

Se probará realizar un progreso descubriendo una constelación y consiguiendo un logro para luego acceder a sus respectivos menús y observar si ya se encuentran en estos.

10.2.8.2 Resultado CP8.

Correcto, cada progreso que se hace en Look Up se ve reflejado de manera visual en su correspondiente menú.

10.2.9 Caso de Prueba 9. Sonido

10.2.9.1 Prueba CP9

Se probará que todos los elementos interactivos que deban emitir sonidos lo hagan de forma correcta.

10.2.9.2 Resultado CP9

Se han encontrado elementos que no reproducían sonido por, como en el CP4, despistes u olvidos en la fase de desarrollo.

10.2.9.3 Solución Aplicada CP5

Estos elementos son botones a los cuales se les ha añadido al conjunto de funciones que ejecutan cuando son pulsados que también ejecuten la función Play() de la correspondiente Audio Source de la escena.

10.3 Pruebas de Seguridad

Uno de los aspectos clave en el desarrollo de Look Up fue la implementación de la base de datos, concretamente el acceso a la misma. Este acceso puede llevar a fallos de seguridad que requieren de mecanismos para prevenir vulnerabilidades de seguridad, como los ataques por inyección de SQL. Estas pruebas fueron esenciales para asegurar que la integridad de la base de datos de usuarios se mantuviera ante intentos malintencionados de acceder o manipular los datos.

SQL Injection es una vulnerabilidad de la seguridad por la cual se puede inyectar código malicioso en una consulta SQL [13]. Durante el proceso de desarrollo, se llevaron a cabo pruebas para prevenir esta vulnerabilidad en los módulos de registro e inicio de sesión, simulando ataques en los que se introducían consultas maliciosas en los campos de entrada de usuario.

10.3.1 Caso de Prueba de Seguridad 1. SQL Injection

10.3.1.1 Prueba CPS1

En la escena de LogIn se intentó iniciar sesión con el siguiente usuario: “ OR 1=1; DELETE FROM usuarios; --”.

10.3.1.2 Resultado CPS1

El sistema sí era vulnerable a este tipo de ataque.

En este ejemplo en concreto, así sería la consulta completa que acabaría recibiendo la base de datos: “SELECT * FROM users WHERE usuario = ' OR 1=1; DELETE FROM usuarios; -- \$username' AND password = '\$password'”, mientras que pretendíamos que recibiese solo la parte con letras verdes. El atacante conseguiría una lista entera de todos los usuarios, pues la condición 1=1 es correcta, y borraría después la misma tabla. El resto del código de la consulta original quedaría comentado por el doble guión al final de la inyección SQL.

10.3.1.3 Solución Aplicada CPS1.

Se implementaron consultas parametrizadas en PHP en todas las interacciones con la base de datos para eliminar la posibilidad de inyección de SQL. A continuación, se muestra un ejemplo del uso de consultas parametrizadas en la función de verificación de credenciales:

```
$stmt = $conn->prepare("SELECT * FROM usuarios WHERE correo = ? AND  
contraseña = ?");
```

```
$stmt->bind_param("ss", $correo, $contraseña);
```

```
$stmt->execute();
```

Gracias a este enfoque, se pudo prevenir exitosamente cualquier intento de inyección de SQL, asegurando que los datos de la base de datos de usuarios permanecieran seguros.

Capítulo 11

CONCLUSIONES

El desarrollo de Look Up ha permitido abordar diferentes aspectos tanto tecnológicos como de diseño, con el objetivo de ofrecer una experiencia educativa y lúdica para los usuarios interesados en la astronomía. A lo largo del proyecto, se han integrado múltiples sistemas que permiten la visualización en tiempo real de astros, su descubrimiento y el seguimiento del progreso del usuario mediante logros y una biblioteca personalizada.

A nivel técnico, uno de los principales logros de este proyecto ha sido la implementación efectiva de la realidad aumentada, unificando la visualización del cielo en tiempo real con una experiencia interactiva. Así como la utilización de la API de Stellarium para la obtención de los datos.

La integración de una base de datos remota para almacenar el progreso del usuario también ha sido un componente clave. Este sistema ha permitido que los usuarios puedan conservar su información a lo largo del tiempo, independientemente del dispositivo que utilicen, asegurando así una persistencia de los datos de vital importancia para la experiencia de usuario.

A nivel personal este proyecto me ha exigido conocimientos muy variados de diferentes campos de estudio de la ingeniería informática, como bases de datos, realidad aumentada, APIs, y desarrollo de backend y frontend. Mantener todos estos elementos cohesionados entre sí para dar lugar a este proyecto ha supuesto sin lugar a duda un reto que me ha brindado una gran experiencia y mejora en mis habilidades en los diferentes ámbitos mencionados.

Para terminar, podemos concluir que Look Up ha logrado cumplir con los objetivos iniciales propuestos, ofreciendo una aplicación robusta, interactiva y educativa que fomenta el interés en la astronomía.

Capítulo 12

FUTURAS MEJORAS

Como futura mejoras que supondrían un gran salto de calidad para Look Up se sugieren los siguientes puntos:

- Más astros. Añadir a la aplicación más cuerpos celestes de diferente índole a las constelaciones y planetas.
- Sistema de descubrimiento gradual. Look Up no ofrecería todos los objetos astronómicos desde el principio, si no que tendrías que ir consiguiendo logros para ir desbloqueando más cuerpos. Por ejemplo, al principio solo tendrías disponibles unas 15 constelaciones, cuando obtuvieses el primer logro pasarías a tener unas 25 disponibles y así gradualmente. De esta forma los logros ganarían más valor y serían más codiciados.
- Sistema de ubicación automático. Tener la opción de que la aplicación detecte tu ubicación automáticamente y no tener que buscarla.

BIBLIOGRAFÍA

- [1] Valenzuela Vila, M. del M. (2010). El nacimiento de la astronomía antigua: Estabilizaciones y desestabilizaciones culturales. *Gazeta de Antropología*, 26(2), Artículo 25. <https://digibug.ugr.es/handle/10481/6768>
- [2] Espinosa Santos, V. (2010). Difusión y divulgación de la investigación científica. IDESIA (Chile), 28(3), 5-6. <https://revistaschilenas.uchile.cl/handle/2250/57232>
- [3] Parra, E. y Torres, M. (2018). La gamificación como recurso didáctico en la enseñanza del diseño. *Educación artística: revista de investigación (EARI)*, 9, 160-173. <https://ojs.uv.es/index.php/eari/article/view/11473/12485>
- [4] Stephen, E. y Mit, E. (2020). Evaluation of Software Requirement Specification Based on IEEE 830 Quality Properties. *International Journal On Advanced Science, Engineering And Information Technology/International Journal Of Advanced Science, Engineering And Information Technology*, 10(4), 1396-1402. <https://ijaseit.insightsociety.org/index.php/ijaseit/article/view/10186>
- [5] JSON. (s.f.). Introducción a JSON. Recuperado el 7 de septiembre de 2024 de <https://www.json.org/json-es.html>
- [6] Luque Ruiz, I., Gómez-Nieto, M. Á., López Espinosa, E., & Cerruela García, G. (2001). Bases de datos: Desde Chen hasta Codd con Oracle (pp. 40-46). Ra-Ma.
- [7] Diagramas UML. (s.f.). Diagramas de casos de uso. Recuperado el 8 de junio de 2024 de <https://diagramasuml.com/casos-de-uso/>
- [8] Zinkhan, George M., Martin, Jr., Claude R. (1987). New brand names and inferential beliefs: Some insights on naming new products. *Journal of Business Research*, 15(2), 157-172. <http://hdl.handle.net/2027.42/26745>

-
- [9] Harada Olivares, E. (2014). Logotipos, isotipos, imagotipos e isologos: una aclaración terminológica. Mixcoac, 2(33), 36-47.
http://www.paginaspersonales.unam.mx/app/webroot/files/157/Logotipos_isotipos_imagotipos_e_isologos_e.pdf
- [10] Porta Simó, L., González Díaz, P., y Pueyo i Busquets, A. (2021) Usabilidad: ¿Qué es y cuáles son sus principios? Universitat Oberta de Catalunya (UOC).
<https://blogs.uoc.edu/informatica/es/usabilidad-que-es-y-cuales-son-sus-principios/>
- [11] Unity Technologies. (s.f.). Localization settings. Recuperado el 10 de junio de 2024 de <https://docs.unity3d.com/Packages/com.unity.localization@1.2/manual/LocalizationSettings.html>
- [12] Unity Technologies. (2022). Scenes. Unity User Manual.
<https://docs.unity3d.com/2022.3/Documentation/Manual/CreatingScenes.html>
- [13] Unity Technologies. (2022). GameObjects. Unity User Manual.
<https://docs.unity3d.com/2022.3/Documentation/Manual/GameObjects.html>
- [14] Unity Technologies. (2022). Introduction to components. Unity User Manual.
<https://docs.unity3d.com/2022.3/Documentation/Manual/Components.html>
- [15] Unity Technologies. (2022). Prefabs. Unity User Manual.
<https://docs.unity3d.com/2022.3/Documentation/Manual/Prefabs.html>
- [16] Unity Technologies. (2022). Resources. Unity User Manual.
<https://docs.unity3d.com/2022.3/Documentation/ScriptReference/Resources.html>
- [17] Unity Technologies. (2022). PlayerPrefs. Unity User Manual.
<https://docs.unity3d.com/ScriptReference/PlayerPrefs.html>
- [18] Coal, I. (2013). Latitud y longitud, azimuth y altitud. <https://es.slideshare.net/slideshow/latitud-y-longitud-azimut-y-altitud/24269934>
- [19] Stellarium. (s.f.). Remote control API. Recuperado el 10 de junio de 2024 de <https://stellarium.org/doc/head/remoteControlApi.html>
- [20] Nyffenegger, R. (s.f.). Set observer location - Stellarium scripting API. Recuperado el 10 de junio de 2024 de
-

<https://renenyffenegger.ch/notes/Wissenschaft/Astronomie/tools/Stellarium/script/API/core/setObserverLocation>

- [21] GeeksforGeeks. (2024). Software engineering | White box testing.
<https://www.geeksforgeeks.org/software-engineering-white-box-testing/>
 - [22] GeeksforGeeks. (2024). Software engineering | Black box testing.
<https://www.geeksforgeeks.org/software-engineering-black-box-testing/>
 - [23] W3Schools. (s.f.). SQL injection. Recuperado el 10 de junio de 2024 de
https://www.w3schools.com/sql/sql_injection.asp
-