Facultad de Ciencias

APLICACIÓN MÓVIL DE REALIDAD AUMENTADA PARA LAS OBRAS DEL PARANINFO DE LA UC

(Augmented reality mobile application for the frescoes from the Paraninfo of the UC)

Trabajo de Fin de Grado
para acceder al

Grado en Ingeniería Informática

Autor: Raúl Pariente González

Director: Carlos Blanco Bueno

Codirector: Óscar Ruiz López

Julio-2022

Agradecimientos

Llegados a este punto, me gustaría agradecer a todas las personas que me han acompañado durante estos años en la Universidad y en el desarrollo de este proyecto.

Primero, agradecer a mi familia por su incondicional apoyo, comprensión en los momentos difíciles y confianza en mí desde el primer día.

También me gustaría agradecer a mis amigos, que me han apoyado y animado cuando lo necesitaba, tanto los de toda la vida como los nuevos que he hecho durante estos años en el Grado.

Por último, pero no menos importante, a la Universidad de Cantabria y a 3DIntelligence por permitirme realizar el proyecto sobre esta aplicación. Finalmente, a mis tutores Carlos y Óscar por su ayuda y orientarme cuando surgían dificultades.

Índice de contenido

Índice de figuras

Índice de tablas

Resumen

La realidad aumentada es una tecnología que está cobrando mucha importancia en los últimos años y que nos acompaña en nuestro día a día. Podemos verla en cualquier parte, desde en nuestro salón en la televisión (deportes, noticiarios...) hasta en nuestros teléfonos, en juegos (*Pokémon GO*), redes sociales (filtros de cámara de *Instagram*/*Snapchat*...), e incluso en aplicaciones de todo tipo como *Google Lens* para identificar objetos y traducir texto en tiempo real, *Star Walk 2* para identificar estrellas y planetas, entre otras.

El proyecto consiste en desarrollar una aplicación móvil (Android e iOS) de realidad aumentada con el fin de divulgar información sobre los frescos de Luis Quintanilla expuestos en el Paraninfo de la Universidad de Cantabria.

La aplicación es capaz de reconocer los frescos expuestos y, dependiendo de la opción que se elija, se podrá identificar puntos de interés, zonas restauradas, jornadas de trabajo o conocer información general del fresco en cuestión. También se podrá conocer la biografía de Luis Quintanilla y todo lo relacionado con los frescos que forman el conjunto "Ama la Paz, Odia la Guerra".

La aplicación está desarrollada con *Unity*, utilizando *ARCore* (dispositivos Android) y *ARKit* (dispositivos iOS) para implementar las funcionalidades en realidad aumentada. También se usarán otros programas para la creación de contenido como *Photoshop* y *Blender*.

Palabras clave: Aplicación, Realidad aumentada, Unity, Luis Quintanilla, frescos, arte.

Abstract

Augmented reality is a technology that is becoming very important in recent years and that accompanies us in our day to day. We can see it anywhere, from our living room on TV (sports, news...) to our phones, in games (*Pokémon GO*), social networks (*Instagram*/*Snapchat* camera filters...), and even in applications of all kinds such as *Google Lens* to identify objects and translate text in real time, *Star Walk 2* to identify stars and planets, among others.

The project consists of developing an augmented reality mobile application (Android and iOS) in order to disseminate information about the frescoes by Luis Quintanilla exhibited in the Paraninfo of the Universidad de Cantabria.

The application can recognize the exposed frescoes and, depending on the option chosen, it will be possible to identify points of interest, restored areas, work shifts or find out general information about the fresco in question. It will also be possible to know the biography of Luis Quintanilla and everything related to the frescoes that make up the "Love Peace, Hate War" group.

The application is developed with Unity, using *ARCore* (Android devices) and *ARKit* (iOS devices) to implement the functionalities in augmented reality. Other content creation programs such as *Photoshop* and *Blender* will also be used.

Keywords: Application, Augmented reality, Unity, Luis Quintanilla, frescoes, art.

1. Introducción

En este apartado se exponen las líneas generales del proyecto, las bases teóricas de la realidad aumentada, el contexto y los objetivos de la aplicación, y la empresa en la que se ha realizado.

1.1. Realidad Aumentada

Una de las tecnologías más llamativas en los últimos años es la "realidad extendida" (Extended Reality, XR). Este término es un concepto complejo que se atribuye a las tecnologías que crean entornos y objetos de forma digital y que permite interactuar con ellos.

El conjunto de tecnologías que forman la realidad extendida está compuesto por: Realidad virtual (RV), realidad aumentada (RA) y realidad mixta (RM).

Figura 1 – Realidad Extendida. Elaboración propia a partir de [1]

En este proyecto nos centraremos en la realidad aumentada (RA), que es la tecnología que será utilizada.

La RA es una tecnología que nos permite añadir capas de información visual sobre el mundo real que nos rodea. Esto nos permite crear experiencias que aportan un conocimiento relevante sobre nuestro entorno, siendo esta información recibida en tiempo real [2].

Dentro de la RA se distinguen dos tipos de dispositivos: visores con forma de gafas o casco (Head Mounted Displays, HMD) y móviles y tablets (Handheld Displays, HHD), aunque también podemos observar realidad aumentada en televisores (en deportes o noticiarios, por ejemplo) u otros dispositivos con pantalla.

Un ejemplo de HMD podrían ser las Google Glass que acaba de anunciar la compañía estadounidense en el Google I/O de 2022. Estas gafas pretenden traducir y mostrar

subtítulos cuando se habla con una persona en otro idioma distinto al propio. A día de hoy, los HMD aún están lejos de ser un tipo de dispositivo práctico y de fácil acceso.

Los HHD son los dispositivos con los que experimentar la RA está al alcance de todo el mundo, ya que sólo con el teléfono y sin necesidad de disponer de un dispositivo de gama alta, se tiene la posibilidad de acceder a decenas de aplicaciones y juegos a través de las tiendas de aplicaciones (Play Store/App Store).

El caso más conocido podría ser el de *Pokémon GO,* un videojuego que consiste en buscar y capturar criaturas de la saga Pokémon escondidas en ubicaciones del mundo real. Además de juegos, hay otros tipos de aplicaciones como *Google Lens* para identificar objetos y traducir textos, *Ikea Place* para comprobar el resultado de una posible compra en la vivienda, *Star Walk 2* para reconocer estrellas y planetas o *Instagram/Snapchat* para aplicar distintos filtros a las fotografías o videos. [3]

Varios sectores están destinando muchos recursos al desarrollo de las tecnologías basadas en la realidad aumentada. Cabe señalar, entre otros, el ámbito de los videojuegos, la educación, la salud, la arquitectura, la publicidad y el turismo.

1.2. 3DIntelligence

En este contexto se sitúa la empresa 3DIntelligence, en la que el firmante realiza prácticas. 3DIntelligence es una empresa centrada en el desarrollo de aplicaciones de realidad aumentada y realidad virtual, normalmente relacionadas con el sector del turismo.

Algunos ejemplos de aplicaciones desarrolladas serían: *Entre Siglos AR Tour* y *Cimavilla AR Tour* en la que el usuario se desplaza por la ciudad de Gijón (Asturias) aprendiendo sobre los distintos personajes y lugares emblemáticos de la zona, o *Secretos del Botánico*, una aplicación para el Jardín Botánico Atlántico (Gijón) donde el usuario puede descubrir el jardín botánico a través de distintos seres mitológicos.

Habiendo ya trabajado previamente con la Universidad de Cantabria en la realización del museo virtual “Luis Quintanilla, arte y memoria” [4], se nos pidió el desarrollo de una aplicación móvil de realidad aumentada para los frescos de Luis Quintanilla expuestos en el Paraninfo de la Universidad de Cantabria.

1.3. Luis Quintanilla y “Ama la Paz, Odia la Guerra”

Luis Quintanilla Isasi (1893-1978) fue un pintor español nacido en Santander. Una de sus obras más famosas fue el conjunto de frescos denominado “Ama la Paz, Odia la Guerra”. Este conjunto está compuesto por los frescos: “Huida”, “Dolor”, “Hambre”, “Destrucción” y “Soldados”.

Estos cinco murales iban destinados al pabellón español de la Exposición Internacional de 1939 en Nueva York. El artista se desplazó a la ciudad norteamericana para realizarlos, pero nunca llegaron a exponerse, ya que la Guerra Civil concluyó antes del comienzo de la muestra. Los frescos fueron gestados por Quintanilla tras conocer en primera persona

la contienda española, denunciando con ellos el horror y la destrucción que acarreaba el conflicto.

Durante medio siglo se les dio por desaparecidos, redescubriéndose en 1990 en Nueva York. En 2007 fueron rescatados (comprados, trasladados a Cantabria y restaurados) gracias al mecenazgo del Banco Santander. Siete años más tarde fueron declarados Bien de Interés Cultural por la Consejería de Educación, Cultura y Deporte del Gobierno de Cantabria.

Figura 2 – Composición del conjunto "Ama la Paz, Odia la Guerra"

1.4. Objetivo

El proyecto pretende desarrollar una aplicación para móviles (Android e iOS) de realidad aumentada cuyo objetivo principal es la divulgación de los frescos que componen el conjunto "Ama la Paz, Odia la Guerra" realizados por Luis Quintanilla y expuestos en el Paraninfo de la Universidad de Cantabria.

Para la consecución del objetivo del proyecto se plantean los siguientes objetivos secundarios:

- **Puntos de interés**, al enfocar un fresco, resaltar elementos y poder conocer información sobre ellos.
- **Zonas restauradas**, al enfocar un fresco, destacar las zonas del fresco que tuvieron que ser restauradas y la técnica utilizada para ello. Además, permitir visualizar el estado previo del mural a la restauración.
- **Jornadas de trabajo**, al enfocar un fresco, distinguir las partes pintadas en cada distinta jornada.
- **Información**, al enfocar un fresco, mostrar información general del fresco.
- **Introducción AR**, al escanear el suelo, colocar una recreación 3D de Luis Quintanilla que introduzca al usuario en la aplicación.
- **Biografía e historia**, enseñar la biografía de Luis Quintanilla y la historia de los cinco frescos.
- **Información Feria Mundial NY (1939)**, documentar sobre la participación de "Ama la Paz, Odia la Guerra" en la Feria Mundial de Nueva York de 1939.

2. Herramientas, tecnologías y materiales utilizados

En este apartado se describen las herramientas, las tecnologías y los materiales utilizados para el desarrollo de la aplicación.

2.1. Herramientas

Las herramientas software utilizadas durante el desarrollo de la aplicación han sido las mostradas a continuación.

2.1.1. Unity

Unity [5] es un motor de videojuego multiplataforma creado por Unity Technologies que permite desarrollar juegos o aplicaciones para distintas plataformas (Windows, macOS, iOS, Android, PlayStation 5, entre otras. Hay una licencia gratuita disponible para uso personal, y otras previo pago dirigidas a empresas, con características adicionales respecto a la gratuita. Para este proyecto se ha utilizado el plan "Plus", que principalmente permite modificar la pantalla de carga por defecto por una personalizada.

Es importante definir algunos conceptos sobre Unity, lo que ayudará al lector a una mejor comprensión del desarrollo de la aplicación detallada posteriormente [6]:

- **Game Object:** Un *Game Object* es la representación básica de cualquier elemento establecido en el motor. Todo *Game Object* viene con un componente *Transform*, que permite indicar posición, rotación y escala del elemento dentro de la aplicación. Por sí solos, los *Game Object* no desarrollan una gran función, su mayor capacidad de uso e interés viene dado por los componentes que puede contener. El buen uso y elección de estos componentes son clave a la hora de desarrollar los diferentes elementos (imágenes, botones, texto…) que componen la aplicación.

- **Canvas:** El *Canvas* es el área donde se deben colocar todos los elementos que forman la interfaz de usuario (UI). Es un *Game Object* con un componente *Canvas* en él, y todos los elementos UI deben proceder de dicho *Canvas.*

- **Scripts:** Un *Script* sirve para controlar el comportamiento de distintos *Game Objects* y sus componentes. Algunas funciones a destacar podrían ser *Awake() y Start(),* que son llamadas al crearse el objeto, o *Update(),* la cual se ejecuta por cada fotograma de la aplicación.

- **Vistas:** Las vistas no es un término propio de Unity, realmente es un *Game Object* con un *Canvas*. Las vistas son un elemento clave en el desarrollo de la aplicación, cada vista se corresponde con el aspecto visual de una pantalla de la aplicación y tiene su propio *Script* que controla todos los elementos que aparecen en él.

- **Prefabs:** Los *Prefabs* permiten almacenar por completo un *Game Object* con sus componentes y propiedades. Para esta aplicación cada elemento colocado en realidad aumentada es un *Prefab.*

2.1.2. Microsoft Visual Studio

Microsoft Visual Studio es un entorno de desarrollo integrado (IDE) para Windows y macOS y el cual es compatible con múltiples lenguajes de programación (C++, C#, Java, Python…). Se utiliza esta herramienta debido a que es la que el propio Unity proporciona para realizar los scripts.

2.1.3. Adobe Photoshop CS6

Adobe Photoshop, desarrollado por Adobe Systems Incorporated, es uno de los principales softwares de creación y edición de gráficos.

Contiene una enorme cantidad de opciones y herramientas sobre varios formatos de archivos. Por este motivo ha sido el software utilizado para desarrollar todo el contenido de la aplicación, desde la creación de los diseños gráficos (logo, iconos, botones, cuadros de texto…) hasta cada una de las imágenes en RA.

2.1.4. Xcode

Xcode es otro entorno de desarrollo integrado que contiene un conjunto de herramientas creadas por Apple. Están destinadas al desarrollo de software para macOS, iOS, watchOS y tvOS. Este software solo se utiliza como medio para compilar el proyecto creado en Unity para iOS y publicar la aplicación en el App Store.

2.1.5. Google Cloud Text-to-Speech

Text-to-Speech es una API que convierte texto en voz. Se eligió el uso de esta herramienta debido a que la voz generada suena "natural" gracias a que usa las tecnologías de inteligencia artificial de Google. Además, permite personalizar la voz gracias a algunos parámetros como el idioma, género, velocidad y tono.

Esta API se utiliza para que toda la información que se pueda mostrar en la aplicación (en formato texto) pueda ser escuchada y mejorar así la accesibilidad de la aplicación.

2.1.6. Audacity

Audacity es un software de edición de audio y grabación de sonido digital. Es un programa gratuito y de código abierto disponible para Windows, macOS y Linux.

Para el proyecto se utiliza para grabar los audios generados con *Google Cloud Text-to-Speech* y para editar algún fragmento si fuese necesario.

2.1.7. DaVinci Resolve

DaVinci Resolve es un software de edición de video no lineal desarrollado por Blackmagic Design. Originalmente fue lanzado como un software de etalonaje digital, pero ha evolucionado hasta el punto de convertirse en un editor de video referente en la industria cinematográfica.

Este software se ha utilizado para crear la pantalla de carga de la aplicación.

2.2. Tecnologías

Las tecnologías utilizadas para la realización de este proyecto han sido C# y AR Foundation.

2.2.1. C#

C# (C Sharp) es un lenguaje de programación desarrollado y estandarizado por Microsoft como parte de su plataforma .NET. Su sintaxis básica deriva de C/C++ y utiliza un modelo orientado a objetos, basado en clases y de tipado fuerte.

Es el lenguaje que proporciona Unity de forma nativa para la realización de aplicaciones y juegos.

2.2.2. AR Foundation

AR Foundation es un marco de trabajo creado especialmente para el desarrollo de experiencias en realidad aumentada e implementarlas en diferentes dispositivos AR distinguiendo entre *ARCore* (Android), *ARKit* (iOS) y *Magic Leap* y *HoloLens* (HMD concretos).

AR Foundation cuenta con multitud de características, pero son tres las que se utilizan en la aplicación:

- **2D Image Tracking,** se utiliza para escanear y reconocer cada fresco con el fin de poder superponer los distintos elementos en realidad aumentada sobre él.

- **Plane Tracking,** para escanear el suelo con el fin de colocar el modelo 3D de Luis Quintanilla.

- **Raycast,** con este componente se mide la distancia entre el dispositivo y el suelo para posicionar la recreación 3D a una distancia concreta.

2.3. Materiales

Toda la documentación correspondiente a los frescos, a la biografía de Luis Quintanilla y a la exposición sobre "Ama la Paz, Odia la Guerra" han sido proporcionada a 3DIntelligence por parte de la Universidad de Cantabria.

El modelo y animación en 3D de Luis Quintanilla ha sido creado por una tercera persona, compañero de trabajo en 3DIntelligence, utilizando programas como *Character Creator 3* y *Blender* entre otros.

Como se ha mencionado previamente todos los diseños de vistas (con sus correspondientes botones, iconos, etc.) y elementos en RA son de elaboración propia utilizando Photoshop.

3. Metodología

En este capítulo se describe la metodología iterativa incremental y cómo se ha empleado para la planificación y el correcto desarrollo de la aplicación.

3.1. Metodología iterativa incremental

Este procedimiento permite planificar un proyecto en distintos bloques temporales denominados iteración. De este modo, este tipo de metodología permite dividir el desarrollo del proyecto en etapas en las que en cada una de ellas se implementan diferentes funcionalidades, permitiendo así una evolución continua del proyecto. [7]

Cada iteración es entendida como un bloque completo de un desarrollo de software que cuenta con sus propias fases de análisis, diseño, implementación y pruebas.

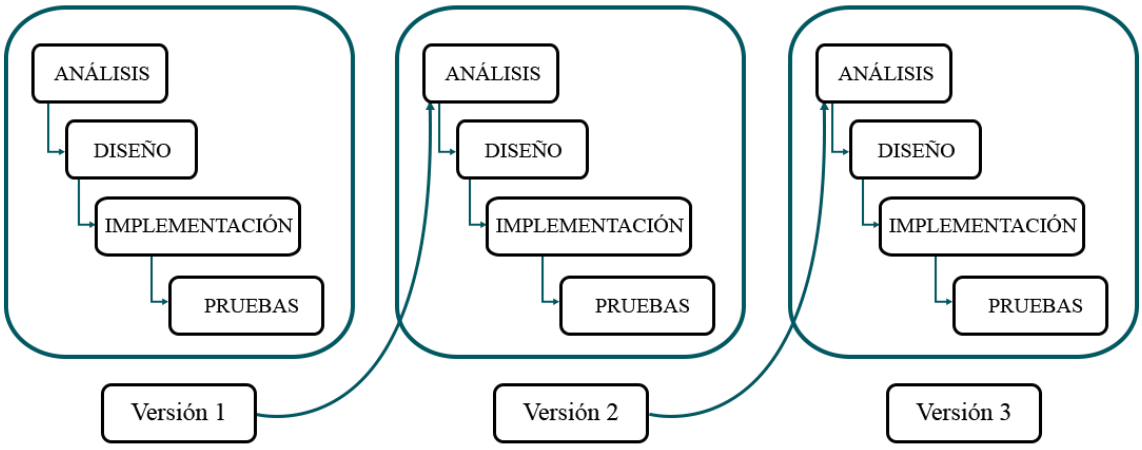
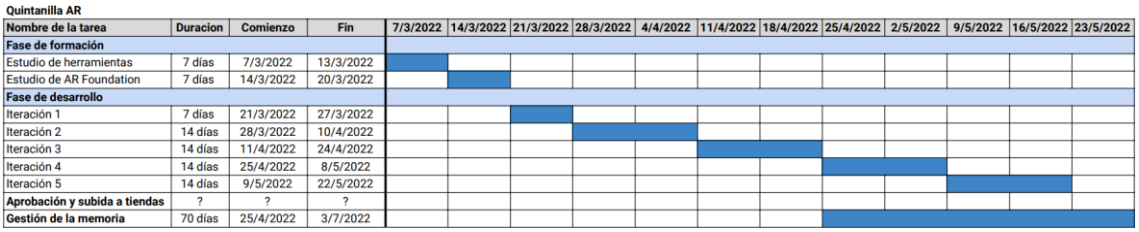


Figura 3 – Modelo iterativo incremental

3.2. Planificación

El proyecto se planificó estableciendo unos plazos orientativos para su desarrollo. En la siguiente figura se muestra la planificación de las fases llevadas a cabo.



Quintanilla AR

Nombre de la tarea	Duracion	Comienzo	Fin	7/3/2022	14/3/2022	21/3/2022	28/3/2022	4/4/2022	11/4/2022	18/4/2022	25/4/2022	2/5/2022	9/5/2022	16/5/2022	23/5/2022
Fase de formación															
Estudio de herramientas	7 días	7/3/2022	13/3/2022												
Estudio de AR Foundation	7 días	14/3/2022	20/3/2022												
Fase de desarrollo															
Iteración 1	7 días	21/3/2022	27/3/2022												
Iteración 2	14 días	28/3/2022	10/4/2022												
Iteración 3	14 días	11/4/2022	24/4/2022												
Iteración 4	14 días	25/4/2022	8/5/2022												
Iteración 5	14 días	9/5/2022	22/5/2022												
Aprobación y subida a tiendas	?	?	?												
Gestión de la memoria	70 días	25/4/2022	3/7/2022												

Figura 4 – Diagrama de Gantt

3.2.1. Formación

Antes de empezar con el desarrollo del proyecto se realizó una fase previa de formación. Esta fase se basó en la decisión, instalación y estudio de uso de las herramientas software necesarias. También se dedicó un tiempo al estudio de la tecnología de AR Foundation y sus características necesarias para la aplicación.

3.2.2. Desarrollo

Esta fase se ha realizado en cuatro iteraciones, generando un prototipo en cada una de ellas con el fin de ir mejorándolo de manera progresiva tanto gráfica como funcionalmente. La fase de desarrollo se realizó en cinco iteraciones, habiendo en cada una varias etapas:

- **Análisis:** En esta etapa de la iteración se definían los objetivos a realizar en la misma.
- **Diseño:** Durante esta etapa se organizaban las escenas, sus respectivas vistas y la estética de cada una de estas (organización de botones, iconos, textos…)
- **Implementación:** En esta tercera etapa, se crearon los materiales y se implementó la lógica establecida previamente.
- **Pruebas:** Finalmente, en la última etapa se comprobaba el correcto funcionamiento de las funcionalidades implementadas.

A continuación, se describen las iteraciones utilizadas y las funcionalidades implementadas en cada una de ellas.

3.2.2.1. Iteración 1

La primera iteración se utilizó para crear una demo sencilla en la que se viese la cámara del teléfono y un par de botones. El fin de la demo era comprobar el correcto funcionamiento de reconocimiento de imagen, la superposición de elementos sobre la misma y la posible navegación entre las diferentes imágenes superpuestas,

3.2.2.2. Iteración 2

En la segunda iteración se organizó la estructura de la aplicación, las vistas necesarias con una apariencia temporal y la navegación entre ellas gracias a un menú principal para elegir las diferentes opciones. También se amplió el controlador de AR Foundation para que reconociese los distintos frescos y añadiese las distintas opciones que permite la aplicación (puntos de interés, zonas restauradas, jornadas de trabajo e información).

3.2.2.3. Iteración 3

La tercera iteración se usó para sustituir la vista de menú principal por una barra inferior con accesos a las diferentes opciones (Frescos, Luis Quintanilla y Feria Mundial de NY). Este cambio y la adición de algunas vistas más de información provocó reorganizar ligeramente la estructura y la navegación entre vistas. A su vez, se definió la apariencia final de la interfaz de cada vista.

3.2.2.4. Iteración 4

En esta iteración se adaptó la aplicación para poder ser utilizada en un segundo idioma (inglés) y se añadió toda la información necesaria, tanto en texto como en audio, gracias a la documentación proporcionada por parte de la Universidad de Cantabria.

3.2.2.5. Iteración 5

En esta última iteración se añadió el modelo de Luis Quintanilla con su respectivo controlador, el cual incluye tanto el funcionamiento del modelo como el de las características de AR Foundation necesarias para escanear el suelo y lanzar el personaje.

También se agregaron modificaciones necesarias para el correcto funcionamiento de la aplicación en dispositivos iOS.

3.2.3. Aprobación y subida a tiendas de aplicaciones

Una vez finalizada la aplicación y tras recibir la aprobación por parte de la Universidad de Cantabria se procederá a la subida de la aplicación a las tiendas de aplicaciones, Google Play para los dispositivos con el sistema operativo de Android y App Store para los de iOS. [1]

[1] Aunque ya ha sido recibido el visto bueno, están aún por definir las fechas para el proceso de subida a las diferentes tiendas de aplicaciones. Dicha decisión compete a la Universidad de Cantabria y es probable que se espere a septiembre para llevar a cabo este proceso.

4. Análisis de requisitos

En este capítulo se presentan todos los requisitos capturados durante las distintas fases de análisis del proyecto. Se muestra una especificación de requisitos detallada, tanto de los requisitos funcionales como de los no funcionales.

4.1. Requisitos funcionales

Los requisitos funcionales son aquellos que definen las funcionalidades que debe tener la aplicación al finalizar el desarrollo. [8]

Los requisitos funcionales de este proyecto se presentan en la Tabla 1, enumerados mediante un identificador y una descripción sobre las acciones que el usuario puede realizar durante el uso de la aplicación.

ID	Descripción
RF01	Al iniciar la aplicación el usuario deberá elegir un idioma entre español o inglés.
RF02	El usuario será capaz de cambiar el idioma de la aplicación en cualquier momento.
RF03	El usuario será capaz de activar o desactivar la opción de narración por voz de la información en cualquier momento.
RF04	El usuario será capaz de activar o desactivar la opción de subtítulos del avatar en cualquier momento.
RF05	El usuario podrá acceder a la política de privacidad de la aplicación.
RF06	El usuario podrá conocer las entidades que han participado en el desarrollo de la aplicación.
RF07	El usuario podrá visualizar mediante RA los distintos puntos de interés de un fresco del conjunto.
RF08	El usuario podrá visualizar mediante RA las zonas del fresco restauradas y el estado previo a dicha restauración.
RF09	El usuario podrá visualizar mediante RA las diferentes jornadas de trabajo en las que fue realizado el fresco.
RF10	El usuario podrá acceder a una descripción de un fresco al enfocarlo con la cámara mediante RA.
RF11	El usuario podrá colocar un modelo 3D de Luis Quintanilla mediante RA.
RF12	El usuario podrá leer la biografía de Luis Quintanilla.
RF13	El usuario podrá conocer la historia de los frescos.
RF14	El usuario tendrá a su disposición información sobre el conjunto “Ama la Paz, Odia la Guerra”.
RF15	El usuario ha de ser capaz de pausar, reanudar o reiniciar el audio en todas las vistas en las que se muestre información sobre los frescos.

RF16	El usuario será capaz de ampliar cualquiera de las imágenes incluidas en la documentación proporcionada por la UC.
RF17	El usuario podrá cerrar la aplicación en todo momento.

Tabla 1 – Requisitos funcionales

4.2. Requisitos no funcionales

Los requisitos no funcionales son aquellos que sirven para indicar cómo se ha de implementar la aplicación, poniendo restricciones en el diseño e implementación de esta.

Los requisitos no funcionales de este proyecto se presentan en la Tabla 2, enumerados mediante identificador, descripción, tipo e importancia para la aplicación.

ID	**Descripción**	**Tipo**	**Importancia**
RNF01	La aplicación podrá utilizarse en iOS y Android.	Portabilidad	Muy alta
RNF02	La aplicación estará disponible en el Play Store para dispositivos con una versión de Android igual o superior a 9.0 y que sean compatibles con los servicios de Google Play para RA (AR Core).	Compatibilidad	Muy alta
RNF03	La aplicación estará disponible en la App Store para dispositivos con una versión de iOS igual o superior a 9.0.	Compatibilidad	Alta
RNF04	El dispositivo ha de tener el sensor de cámara funcional.	Hardware	Alta
RNF05	La aplicación no bajará de 30 FPS mínimos.	Rendimiento	Media
RNF06	La aplicación ha de responder en menos de 2 segundos.	Rendimiento	Alta
RNF07	La aplicación hará un uso menor de 2 GB de RAM.	Rendimiento	Alta
RNF08	El dispositivo deberá contar con un almacenamiento libre de al menos 250MB para el correcto funcionamiento	Disponibilidad	Media
RNF09	La aplicación debe ser fácil de usar e intuitiva.	Usabilidad	Alta
RNF10	La aplicación ha de ser accesible para personas con dificultades auditivas y/o visuales	Accesibilidad	Alta

Tabla 2 – Requisitos no funcionales

5. Diseño e implementación

En este capítulo se explica en primer lugar la arquitectura del sistema utilizada para detallar posteriormente el proceso de diseño e implementación.

5.1. Arquitectura de tres capas

La arquitectura empleada para este proyecto es la de tres capas [9]. Hace referencia a una arquitectura de software de aplicación que divide las aplicaciones en tres capas de informática lógica y física, estas capas se ejecutan en su propia infraestructura permitiendo modificar cada una de ellas sin que afecte a los demás.

Las tres capas son la capa de presentación, la capa de aplicación y la capa de datos.

Figura 5 – Arquitectura de tres capas

5.2. Capa de presentación

La capa de presentación es la interfaz de usuario y de comunicación de la aplicación, donde el usuario interactúa con la aplicación. Su objetivo principal es mostrar información al usuario y recopilar datos de este.

Esta capa está formada por dos módulos que gestionan la interacción de la aplicación con los diferentes componentes de entrada y salida del dispositivo.

- **Módulo de entrada.** Es el encargado de recoger la información que produce el usuario en la aplicación mediante diferentes componentes de entrada del teléfono móvil.
    - **Pantalla táctil:** Recoge la interacción que produce el usuario cuando éste usa la pantalla para navegar por la aplicación.
    - **Sensor de cámara:** Es el componente principal que posibilita que la RA sea funcional. Se utiliza para reconocer diferentes patrones (frescos en este caso) y superficies.
- **Módulo de salida.** Se encarga de proporcionar la información de la aplicación al usuario utilizando componentes de salida del dispositivo.

- **Pantalla:** Muestra gráficamente lo que sucede durante el uso de la aplicación.
- **Altavoz:** Reproduce toda la información que aparece escrita sobre los frescos o el conjunto. Además, se generan otros sonidos para enriquecer el uso.
- **Sensor de vibración:** Con el fin de mejorar la experiencia de uso de la aplicación el dispositivo vibrará en algunas ocasiones.

5.2.1. Diagrama de flujo de navegación

A continuación, se muestra el diagrama de navegación entre las diferentes vistas y una descripción detallada de cada una de ellas.

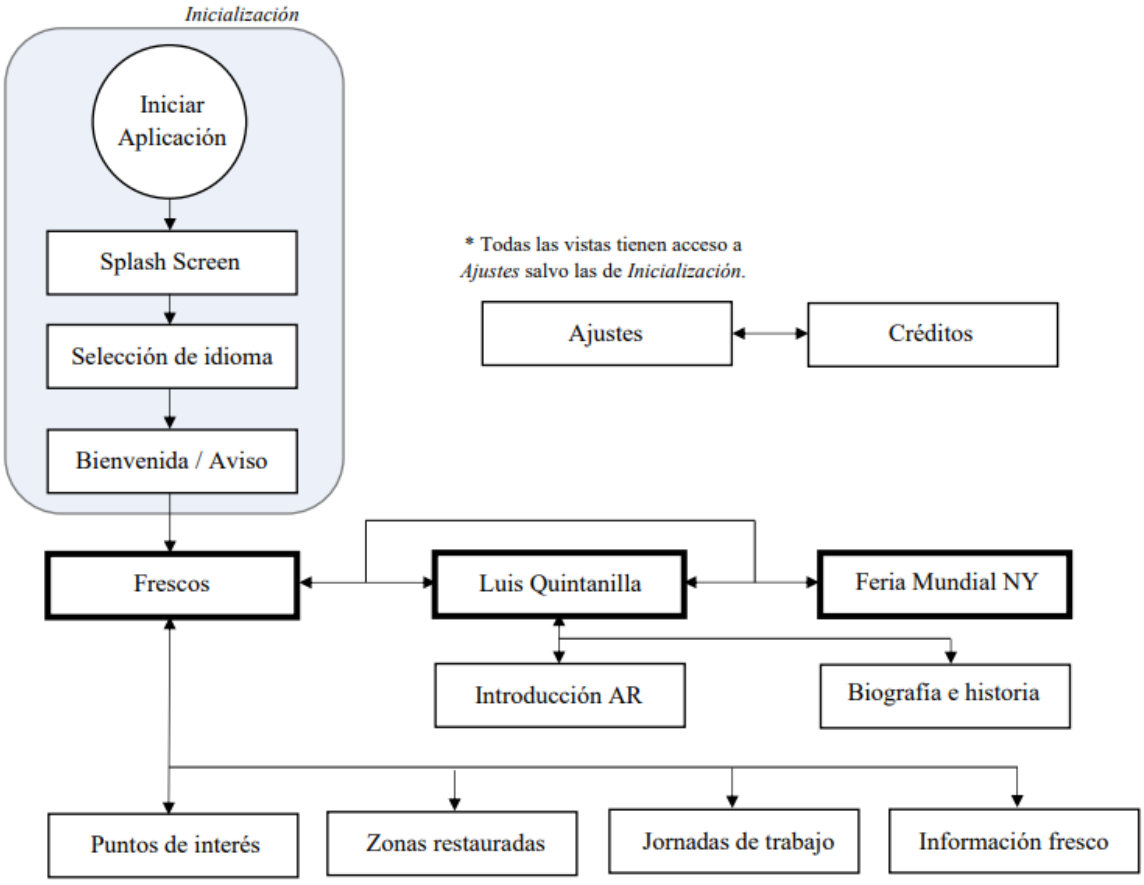


Figura 6 – Diagrama de flujo de navegación

Splash Screen

Es la pantalla de carga de la aplicación, se muestra un recorte del fresco "Soldados" y título "Ama la Paz, Odia la Guerra. Los frescos de Luis Quintanilla".

Selección de idioma

Se utiliza para pedir al usuario que seleccione el idioma que quiera utilizar, a elegir entre español o inglés. También se muestran los logos de la *Universidad de Cantabria*, del *Campus Cultural* y del *Ministerio de la Presidencia, Relaciones con las Cortes y Memoria Democrática*.

Figura 7 – "Selección de idioma"

Bienvenida / Aviso

Muestra un mensaje de bienvenida y advierte al usuario para que tenga precaución con su alrededor mientras utiliza la aplicación. Este mensaje es obligatorio en todas las aplicaciones de RA en el caso de decidirse subir la aplicación a las diferentes tiendas.

Figura 8 – "Bienvenida/Aviso" 1

Figura 9 – "Bienvenida/Aviso" 2

Frescos

Presenta una pequeña descripción de uso y permite al usuario elegir entre las diferentes opciones de RA para cada fresco, cambiar de menú con los botones de la barra inferior o acceder a ajustes con el icono de la esquina superior derecha (el cual estará presente en todas las vistas siguientes). Estas opciones de RA utilizan la característica *2D Image Tracking* (detección de imagen) de AR Foundation.

Figura 13 – "Frescos"

Puntos de interés

Esta opción de RA permite al usuario acceder a información (escrita o en imágenes mediante bocetos) sobre los personajes que aparecen en los frescos. Antes de mostrar la cámara para que el usuario elija el contenido en RA al que desea acceder se muestran unas instrucciones.

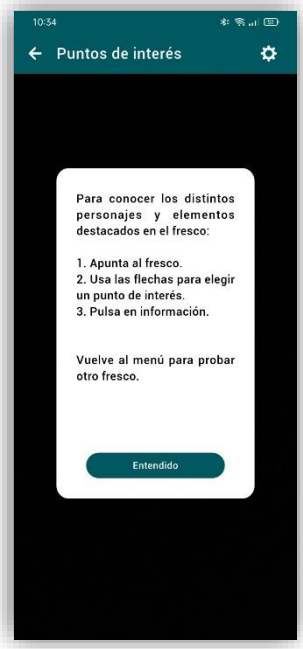


Figura 10 – "Puntos de interés" 1

Figura 11 – "Puntos de interés" 2

Figura 12 – "Puntos de interés" 3

Zonas restauradas

En esta vista los elementos en RA se corresponden con áreas delimitadas dentro del fresco en las que fue necesaria una restauración. Estas zonas se acompañan de una breve descripción, junto a la técnica empleada para restaurar esa parte. También cuenta con unas instrucciones al principio y la posibilidad de ver el estado del fresco antes de ser restaurado.

Figura 14 – "Zonas restauradas" 1

Figura 15 – "Zonas restauradas" 2

Figura 16 – "Zonas restauradas" 3

Jornadas de trabajo

Al tratarse de un mural al fresco en el que la pintura se seca con rapidez, se puede apreciar con facilidad qué partes fueron realizadas durante una misma jornada. Esta tercera opción de RA se encarga de mostrar las zonas correspondientes a cada jornada.

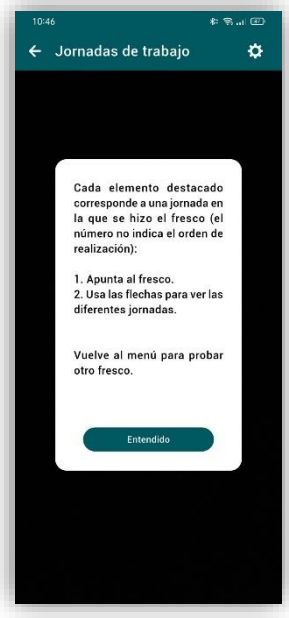


Figura 17 – "Jornadas de trabajo" 1

Figura 18 – "Jornadas de trabajo" 2

Información fresco

Esta última opción de RA sirve para, una vez enfocado y reconocido uno de los frescos con la cámara, mostrar información relevante sobre el fresco en cuestión.

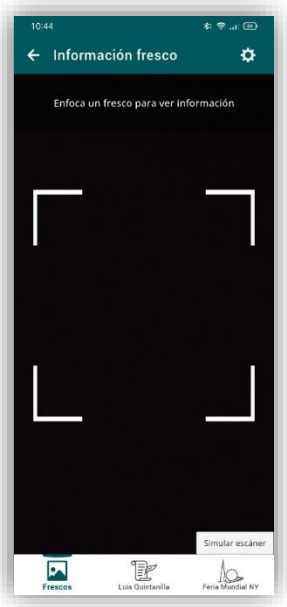


Figura 19 – "Información fresco" 1

Figura 20 – "Información fresco" 2

Luis Quintanilla

Aquí se presenta un nuevo menú en el que, al igual que en **Frescos**, el usuario dispone de una barra inferior para cambiar de menú y un acceso a los ajustes. Las dos opciones de este menú se corresponden por un lado con una introducción en RA al conjunto "Ama la Paz, Odia la Guerra" y por otro lado con la biografía Luis Quintanilla y la historia de los frescos.

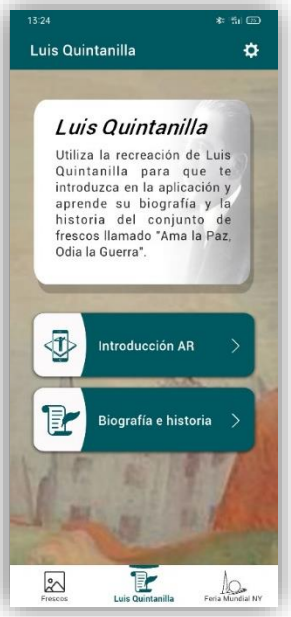


Figura 21 – "Luis Quintanilla"

Introducción AR

En esta vista se utilizan las otras características de AR Foundation: *Plane Tracking* y *Raycast*. En primer lugar se muestra una animación en el centro de la pantalla que sirve como guía para posicionar el modelo 3D. Una vez detecta la superficie a una distancia de 3 metros se coloca el modelo automáticamente y aparecen distintas opciones para interactuar con él.

Figura 22 – "Introducción AR" 1

Figura 23 – "Introducción AR" 2

Biografía e historia

Se detalla la biografía de Luis Quintanilla y la historia de los cinco frescos que forman el conjunto.

Figura 24 – "Biografía e historia"

Feria Mundial de NY

Cuenta cómo el conjunto fue creado con objeto de ser expuesto en la Feria Mundial de Nueva York celebrada en 1939, así como que tuvo que volver a España debido al fin de la Guerra Civil española y finalmente, habiendo estado desaparecidos durante medio siglo en la ciudad de NY, cómo los frescos fueron restaurados y expuestos en el Paraninfo de la Universidad de Cantabria.

Figura 25 – "Feria Mundial NY"

Ajustes

Permite al usuario cambiar de idioma, activar o desactivar tanto la narración por voz de la información como los subtítulos del modelo 3D, acceder a los créditos y visualizar la política de privacidad de la aplicación.

Figura 26 – "Ajustes"

Créditos

Se muestran todas las entidades que han tomado parte en el desarrollo de este proyecto.

Figura 27 - "Créditos"

5.3. Capa de aplicación

La capa de aplicación, también conocida como la capa lógica o media, es el núcleo de la aplicación y el nivel en el que se procesa la información recopilada en la capa de presentación. En esta capa, se almacena toda la lógica del sistema, incluyendo tanto las funcionalidades que debe implementar la aplicación, como las interacciones y reglas entre elementos.

A continuación, se muestran todas las clases utilizadas en la aplicación.

- Máquina de estados
  - StateMachine.cs
  - NavigationStateMachine.cs
- Utilidades
  - PlayerPreferences.cs
  - AndroidUtility.cs
  - PanZoom.cs
  - TextClass.cs
  - Toast.cs
- Manager de RA
  - ARManagerFrescoes.cs
  - ARManager.cs
- Estados
  - State.cs
  - InitialState.cs
  - LangauageState.cs
  - WelcomeState.cs
  - FrescoesMenuState.cs
  - POIState.cs
  - RestorationsState.cs
  - WorkingDaysState.cs
  - InformationFrescoState.cs
  - LuisQuintanillaState.cs
  - IntroductionARState.cs
  - BiographyState.cs
  - NYFairState.cs
  - ConfigurationState.cs
  - CreditsState.cs
- Controladores
  - Controller.cs
  - LanguageController.cs
  - WelcomeController.cs
  - FrescoesMenuController.cs
  - POIController.cs
  - RestorationsController.cs
  - WorkingDaysController.cs
  - InformationFrescoController.cs
  - LuisQuintanillaController.cs
  - IntroductionARController.cs
  - BiographyController.cs
  - NYFairController.cs
  - ConfigurationController.cs
  - CreditsController.cs

Figura 28 – Clases de la aplicación

5.3.1. Navegación entre vistas (Máquina de estados)

Toda la estructura lógica de la aplicación está basada en una máquina de estados en la que cada estado se corresponde con una vista. Las clases que se encargan de llevar a cabo las funciones de navegación entre vistas son *NavigationStateMachine.cs* y *StateMachine.cs* (de la cual deriva la primera).

StateMachine.cs además de ser una clase abstracta cuya función es cambiar de estado, también mantiene una referencia al estado anterior, una característica fundamental que permitirá al controlador de cada vista tomar unas decisiones u otras dependiendo del estado previo. Por otro lado, *NavigationStateMachine.cs*, es el encargado de iniciar la máquina de estados y el objeto al que se llame cuando se quiera cambiar de estado si éste también requiera un cambio en la escena de Unity.

5.3.2. Estados

Como se ha mencionado previamente cada pantalla tiene una clase estado y otra controlador. La clase abstracta *State.cs* es de la que derivan todos los demás estados, cuenta con atributos y métodos para gestionar el estado actual (iniciar, finalizar y devolver nombre y máquina de estados).

Todas las clases estado de cada vista realizan la misma acción: al iniciar buscan el tag asociado a la vista en el editor de Unity para acceder e iniciar el controlador, y al finalizar lo desactivan y cambian de estado al que se pase por parámetro.

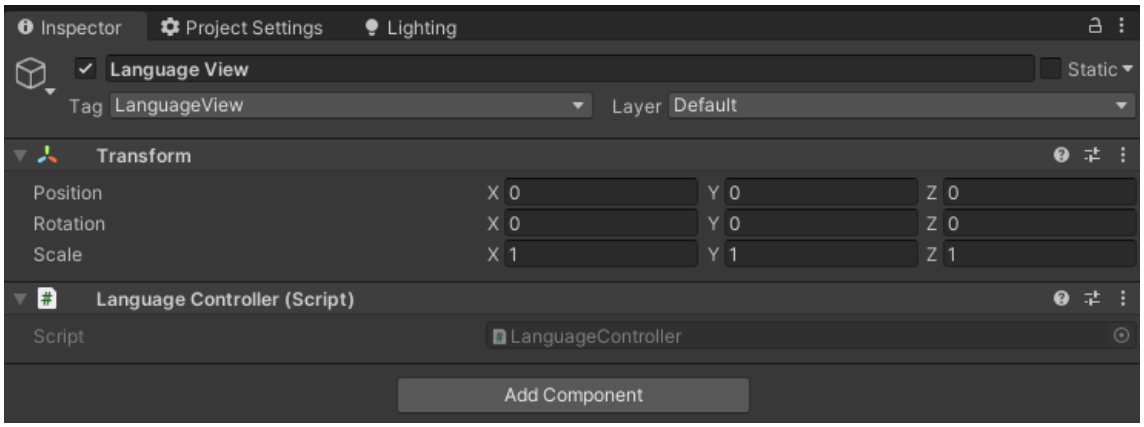


Figura 29 – Propiedades objecto Language View

Existe un estado especial que es el *InitialState.cs*, este es el primero al que se accede al iniciar la aplicación, decide si la primera vista es la de selección de idioma o la de bienvenida/aviso. Permite evitar una pantalla innecesaria si el usuario ya ha usado la aplicación con anterioridad.

5.3.3. Controladores

Los controladores de cada vista también derivan de una clase abstracta, en este caso *Controller.cs*. Esta clase establece el color de la barra de estado (donde se muestran las notificaciones, hora y batería) y cuenta con el método para activar o desactivar el primer hijo de la vista, este siempre será el canvas donde se encuentran todos los elementos de la vista.

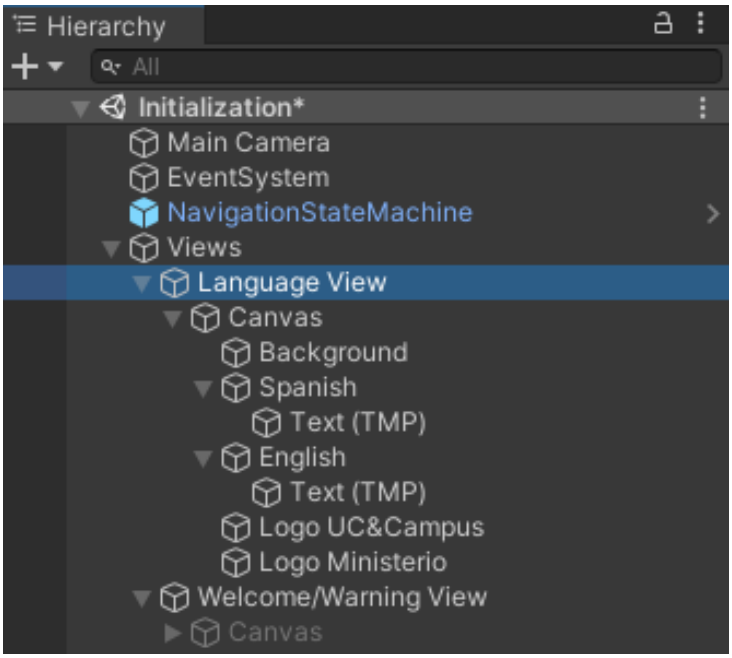


Figura 30 – Jerarquía Language View

Al igual que los estados, todos los controladores tienen un bloque común de código en el que se traducen los textos al idioma elegido y se activa la vista con la función que hereda de *Controller.cs*. Aparte de esto, cada controlador se encarga de indicar cómo debe reaccionar su vista a diferentes interacciones del usuario, véase acciones al pulsar botones, carga y manejo de contenido (imágenes y audio), mostrar u ocultar elementos, etc.

Los controladores más complejos de desarrollar son las que corresponden con vistas de RA, ya que además de lo mencionado anteriormente, tienen que controlar el correcto funcionamiento de los diferentes elementos de RA.

5.3.4. Manager de RA

Se utilizan dos clases manager con el fin de facilitar la organización de las funcionalidades necesarias para implementar la RA en los distintos casos.

- ***ARManagerFrescoes.cs*:** Esta clase se encarga de la gestión de la RA correspondiente con las vistas de “Puntos de interés”, “Zonas restauradas” y “Jornadas de trabajo”.

Para estas tres vistas la característica de AR Foundation utilizada es únicamente *2D Image Tracking* (detección de imagen). Por ello necesita una librería con las imágenes de referencia que utilizará, estas imágenes son los frescos "Dolor", "Destrucción", "Huida", "Soldados" y "Hambre", a los que hay que añadir sus dimensiones originales.

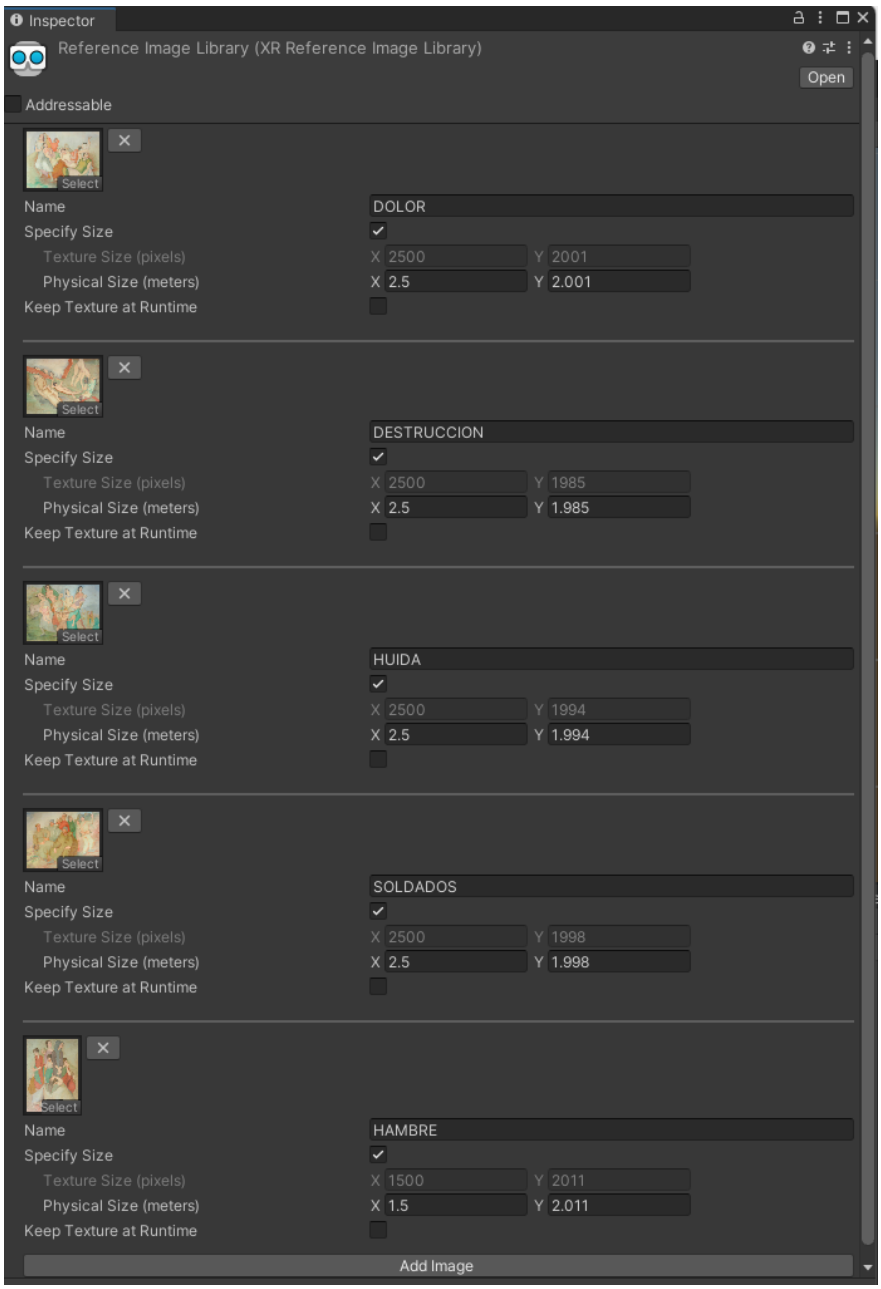


Figura 31 – Librería de imágenes de referencia

Cuando detecta que lo que se está mostrando por la cámara coincide con uno de los frescos, superpone sobre la imagen de referencia el elemento en RA correspondiente. Este elemento es un *prefab* cuyo nombre está compuesto por "nombreFresco_opciónRA_contador" y el cual se utiliza para buscar dentro de una lista formada por ciento cuatro posibles imágenes de realidad aumentada.

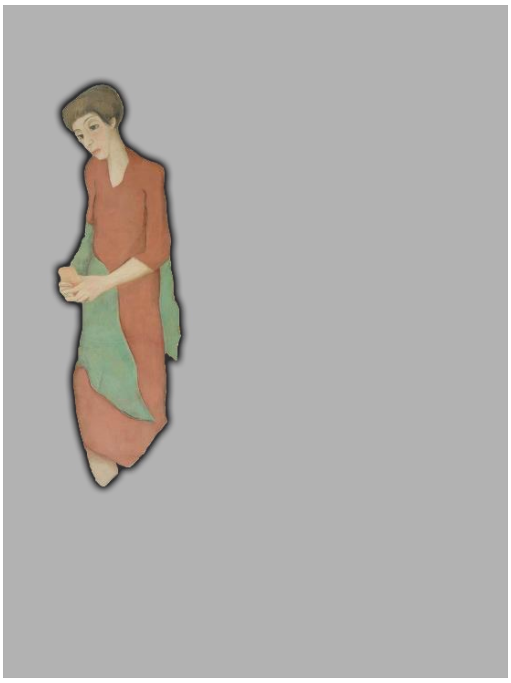


Figura 32 - HAMBRE_POI_1

El nombre del fresco es el que obtiene al detectar la imagen de referencia, la opción de RA dependerá de en qué opción -entre "Puntos de interés", "Zonas restauradas" o "Jornadas de trabajo"- esté, y el contador, que es el número que se muestra en la vista entre las flechas y que aumenta o disminuye a medida que se usan.

- ***ARManager.cs*:** Por otro lado en esta clase se implementa la RA para las vistas de "Información fresco" y "Introducción AR". Aunque todas las vistas se podrían haber implementado un un único gestor de RA se decidió implementar estas dos vistas en otra clase por evitar tener una clase demasiado compleja.

  Para la primera vista el funcionamiento inicial es el mismo al mencionado anteriormente: utilización de la carácteristica de detección de imagen y la librería de referencia de imágenes para que cuando coincida lo mostrado en cámara con uno de los frescos se muestre automáticamente la información general del fresco correspondiente.

  En la vista "Introducción AR" se utilizan tanto *Plane Tracking* como *Raycast*. La detección de plano consiste en apuntar al suelo con la cámara para que el dispositivo reconozca la superficie, es importante que no haya obstáculos en medio. El *Raycast* se utiliza para, una vez detectada la superficie, medir la distancia entre en el dispositivo y el punto céntrico de lo que se muestra en pantalla. La distancia se calcula continuamente gracias a la función *Update()* para que cuando sea de 3,5 metros se coloque el modelo 3D. Esta condición se implementa para que el modelo no se posicione ni cerca ni lejos y la experiencia sea lo más atractiva posible.

5.3.5. Utilidades

- ***PlayerPreferences.cs*:** Aquí se almacenan las preferencias del usuario de idioma, narración de voz y subtítulos automáticos. Todas son configurables desde la vista de "Ajustes".

- ***AndroidUtility.cs*:** Se encarga de mostrar la barra de estado en los dispositivos Android, una funcionalidad que estaba implementada en anteriores versiones de Unity pero que quitaron por algunos errores de funcionamiento (para iOS sí está disponible).

- ***Toast.cs*:** Esta clase nos permite evitar que el usuario cierre la aplicación pulsando "atrás" en las vistas "Frescos", "Luis Quintanilla" y "Feria Mundial NY". Al realizar esa acción aparece un mensaje de aviso durante un segundo en la parte inferior de la pantalla y si el usuario repite la acción en este tiempo entonces sí se cerrará la aplicación.

Figura 33 – Vista "Frescos" con aviso

- ***PanZoom.cs:*** Cuando el usuario pulsa sobre una de las imágenes de información, esta se abre en una vista nueva y permite ampliar o encoger la imagen usando dos dedos.

- ***TextClass.cs*:** Es un diccionario, toda la documentación escrita de la aplicación se encuentra en esta clase. Cuenta con una única función que devuelve un string dado un identificador y otro entero que se corresponde con el idioma elegido.

5.4. Capa de datos

La capa de datos, a veces denominada capa de base de datos, capa de acceso a datos o backend, es donde se almacena y gestiona la información procesada por la aplicación.

La documentación en formato de texto se encuentra almacenada en la clase *TextClass.cs*. Como se ha mencionado previamente, los controladores tienen un bloque común de código en el que se activa la vista y se traducen los todos los textos. Para traducir los textos se llama a la clase *TextClass.cs* indicando el id deseado y el idioma escogido.

```
private void TranslateView()
{
    int lang = PlayerPreferences.GetInt(Preference.Language);
    welcomeTitle.text = TextClass.GetLocalizedString(0, lang);
    welcomeText.text = TextClass.GetLocalizedString(1, lang);
    warningTitle.text = TextClass.GetLocalizedString(2, lang);
    warningText.text = TextClass.GetLocalizedString(3, lang);
    buttonContinue.GetComponentInChildren<TextMeshProUGUI>().text = TextClass.GetLocalizedString(4, lang);
}
```

Figura 34 – Fragmento de código de la clase WelcomeController.cs

```
public static class TextClass
{
    public static Dictionary<int, string[]> Dictionary = new Dictionary<int, string[]> {
        { 0, new string[]{
            "¡Bienvenido!",
            "¡Welcome!"}},

        { 1, new string[]{
            "Adéntrate en esta experiencia y descubre los frescos de Luis Quintanilla a través de la Realidad Aumentada.",
            "Inmerse yourself into this experience and discover Luis Quintanilla's frescoes through Augmented Reality."}},

        { 2, new string[]{
            "¡Advertencia!",
            "¡Caution!"}},

        { 3, new string[]{
            "Recuerda prestar atención a tu alrededor mientras usas la aplicación.",
            "Remember to pay attention to your surroundings while using the application."}},

        { 4, new string[]{
            "Comenzar",
            "Start"}},
```

Figura 35 – Fragmento de código de la clase TextClass.cs

Por otro lado, para la información en formato de imagen y audio se utiliza *Unity Addressable Asset System* [10] un sistema que permite cargar archivos mediante direccionamiento.

Al activar la casilla “Addressable” en el inspector de las imágenes y audios, nos proporciona la ruta de direccionamiento donde se encuentra almacenado, esta dirección se utiliza posteriormente para cargar el contenido en el controlador de la vista deseada.

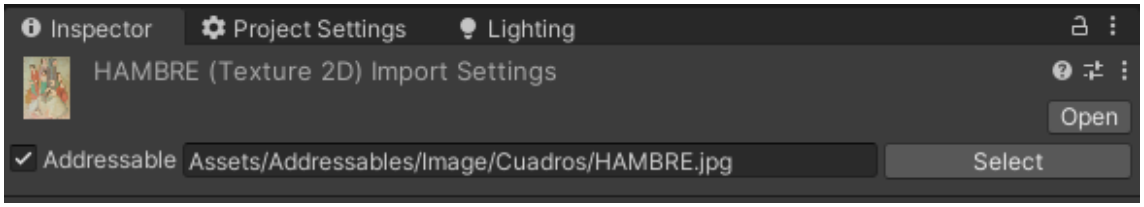


Figura 36 – Fragmento del inspector del fresco Hambre

```
private void LoadImage(string file)
{
    Addressables.LoadAssetAsync<Sprite>("Assets/Addressables/Image/Cuadros/" + file + ".jpg").Completed += OnImageLoaded;
}
1 referencia
private void OnImageLoaded(UnityEngine.ResourceManagement.AsyncOperations.AsyncOperationHandle<Sprite> obj)
{
    SetImage(obj.Result);
    imageLoaded = true;
}
6 referencias
private void SetImage(Sprite sprite)
{
    imageInfo.sprite = sprite;
}
```

Figura 37 – Fragmento de código de la clase InformationFrescoController.cs

6. Evaluación y pruebas

En este capítulo se realizan las pruebas para validar el correcto funcionamiento de la aplicación y comprobar el cumplimiento de los requisitos especificados.

6.1. Pruebas unitarias y de integración

Las pruebas unitarias consisten en técnicas empleadas para la comprobación del correcto funcionamiento de fragmentos de código. Gran parte de estas pruebas se han llevado a cabo mediante los mensajes que genera el propio motor de Unity cuando algo no funciona correctamente y la clase propia de Unity *Debug* (usando las funciones *Log()* , *LogError()* y *LogWarning*) que mandan mensajes a la consola de Unity mientras se ejecuta la aplicación.

Sin embargo, esta técnica no es válida cuando se quiere comprobar el funcionamiento de las vistas de RA, ya que el ordenador no es compatible y no puede ejecutarlas desde el editor de Unity. Es por esto por lo que se ha empleado otro método que permite ver los mismos mensajes de la clase *Debug* en una terminal del sistema operativo mientras se ejecuta la aplicación en el dispositivo móvil. Para esto, hay que mantener conectado por cable el dispositivo al ordenador, tener la opción de “Depuración por USB” activada y escribir en la terminal del ordenador el comando:

“adb logcat -s unity activitymanager packagemanager dalvikvm debug”

Por otro lado, existen también las pruebas de integración. El objetivo de estas pruebas es que, una vez verificados los módulos unitarios por separado, ampliar el rango de alcance de las pruebas.

Cada vez que se carga una vista nueva en la aplicación se considera una prueba de integración. De este modo y gracias a la previsualización que ofrece el propio motor de Unity se puede detectar, de manera ágil, si los elementos desarrollados para cada vista y la manera de relacionarlas tienen cualquier comportamiento erróneo.

Al finalizar la total implementación de la aplicación y para realizar la comprobación general con una correcta integración de todas las partes, se ejecuta la aplicación recorriendo cada vista y utilizando todos sus componentes, con el fin de asegurar el correcto funcionamiento.

6.2. Pruebas de sistema

Para comprobar el total comportamiento del sistema y verificar el cumplimiento de algunos de los requisitos no funcionales establecidos con anterioridad, se ha hecho uso de *Unity Profiler*, una herramienta propia de Unity. Esta herramienta realiza un análisis de la aplicación mientras está siendo ejecutada, calculando de esta forma los tiempos y uso de recursos.

Uno de los requisitos establecidos para la aplicación es que esta no baje de 30 FPS durante la ejecución (RNF05), como se observa en la gráfica de la Figura 38, este requisito se cumple holgadamente al igual que el RNF06 que exige un tiempo de respuesta menor a 2 segundos.

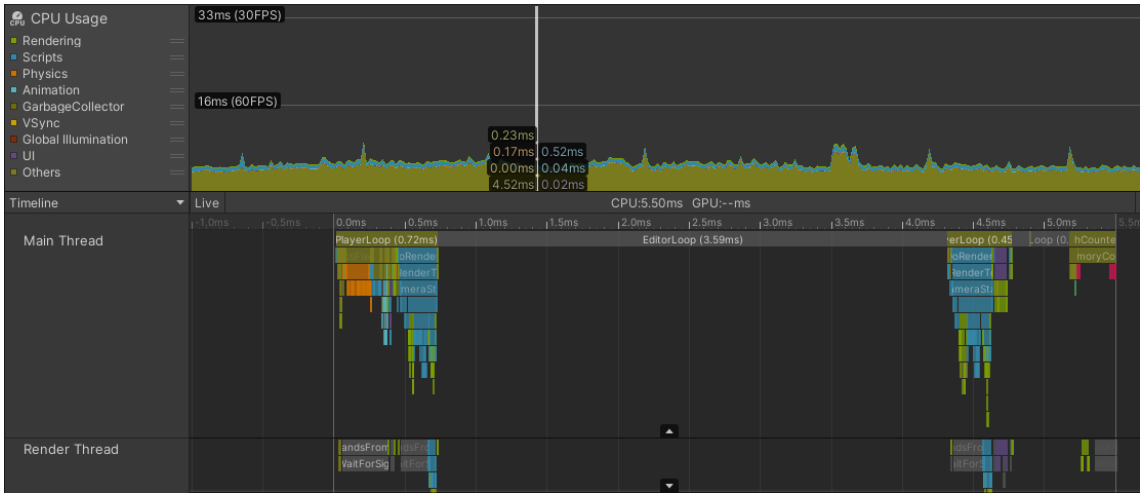


Figura 38 – Gráficas generadas por Unity Profiler

Asimismo, en el RNF07 se pide que no se haga uso de más de 2 GB de memoria durante la ejecución. Para comprobar este requisito, se vuelve a hacer uso de Unity Profiler, donde se detalla el consumo de memoria en cada componente y el uso empleado por todo el sistema, que en este caso es de 1,76 GB.

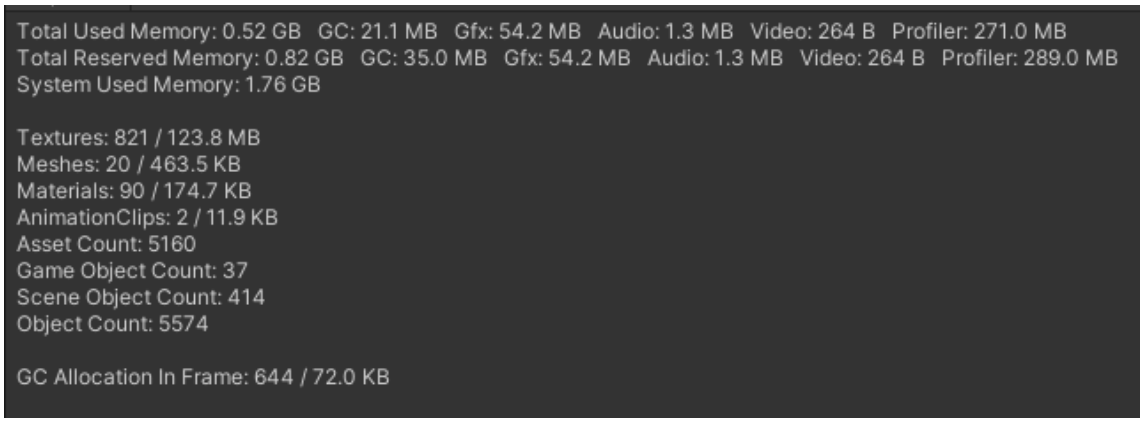


Figura 39 – Uso de memoria generado por Unity Profiler

El RNF08 pide 250MB de almacenamiento disponible para el correcto funcionamiento, esto se traduce en que la aplicación no puede exceder de esa cantidad de memoria. La Figura 40 muestra como desde un dispositivo, la aplicación no llega a esa cantidad de almacenamiento y por lo tanto cumple el requisito.

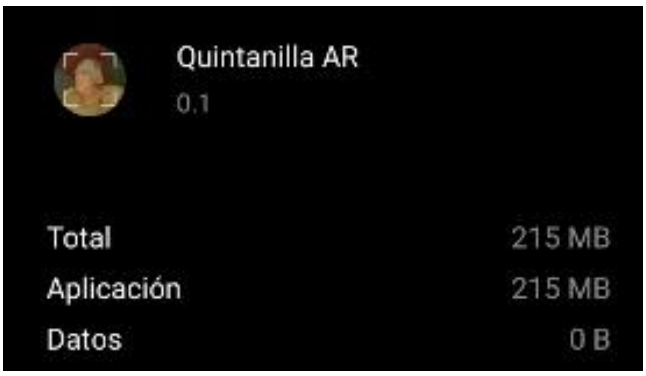


Figura 40 – Uso de almacenamiento en dispositivo

Con el fin de cumplir el RNF09, los elementos son claramente visibles gracias a que el tamaño y contraste de los iconos, botones e imágenes facilitan esta característica. La barra de navegación inferior y el botón de atrás ayudan al usuario a comprender y manejar la aplicación de manera intuitiva.

También se pide que la aplicación sea accesible para personas con dificultades auditivas y/o visuales (RNF10). Es por esto por lo que se ha implementado la funcionalidad de narración por voz en todas las vistas de información y los subtítulos en el modelo 3D.

Por último, en cuanto a la portabilidad y compatibilidad (RNF01, RNF02 y RNF03), Se limita la aparición de la aplicación en tiendas a dispositivos compatibles con la versión de Android e iOS requerida y en el caso de Android se excluye también a los que no sean compatibles con los Servicios de Google Play para RA.

6.3. Pruebas de aceptación

Durante el desarrollo de la aplicación se han ido llevando a cabo diferentes reuniones con personal del Paraninfo de la Universidad de Cantabria para ir validando el progreso y el cumplimiento de requisitos de la aplicación.

Para cada una de las reuniones se preparaba una versión funcional con lo desarrollado hasta el momento con el objetivo de hacer todas las pruebas que fuesen necesarias y de aportar los comentarios pertinentes.

La última reunión se realizó en el propio Paraninfo de la UC para comprobar una por una todas las posibles opciones de RA con los diferentes frescos, tras concluir la prueba de forma satisfactoria se determinó por finalizada la aplicación a falta de decidir cuándo se llevaría la subida a las diferentes tiendas de aplicaciones.

7. Conclusiones y trabajos futuros

Para finalizar, el objetivo de este último capítulo de la memoria es exponer las conclusiones obtenidas en el desarrollo del proyecto, y las posibles líneas de trabajo futuras de la aplicación.

7.1. Conclusiones

Antes de hablar de las conclusiones es necesario recordar que el objetivo principal de la aplicación era desarrollar una aplicación móvil de realidad aumentada que sirviese para divulgar toda la información posible sobre los cinco frescos que forman el conjunto "Ama la Paz, Odia la Guerra" de Luis Quintanilla. A la vista del resultado obtenido se considera que el objetivo ha sido cumplido con gran satisfacción, habiéndose además satisfecho todos los requisitos. Se ha conseguido desarrollar una aplicación móvil muy útil para todas aquellas personas que quieran saber más acerca de los frescos que se encuentran expuestos en la entrada del Paraninfo de la UC y sobre la figura de Luis Quintanilla.

Ha resultado un proyecto muy interesante de realizar, no solo por ser el primer proyecto completo que llevo a cabo durante mi transcurso en el grado sino también porque coincidió con el inicio de mi estancia en prácticas en la empresa 3DIntelligence. Por otra parte, he aplicado diversas técnicas adquiridas a lo largo del grado, además, de haber aprendido muchas otras a lo largo del proyecto que han ampliado mis conocimientos para poder seguir desarrollando aplicaciones en la empresa.

Por último, recalcar la satisfacción personal de ver gente ajena a la empresa y al Paraninfo probar la aplicación y de disfrutar la realidad aumentada y la experiencia que esta ofrece.

7.2. Trabajos futuros

Respecto a los trabajos futuros, la aplicación está terminada a falta de subirla a las tiendas de aplicaciones. Sin embargo, si recibiéramos reportes por algún error o porque la Universidad de Cantabria quisiera modificar algo de la información siempre se podría actualizar y subir una nueva versión a las tiendas.

Hay algunas mejoras que se podrían haber implementado pero que no eran factibles por presupuesto, tiempo y/o tamaño de aplicación, como por ejemplo contratar actores de voz para el modelo 3D de Luis Quintanilla y las narraciones de voz en ambos idiomas o utilizar también el modelo para que se fuese él quien te cuente la información de los puntos de interés, zonas restauradas, jornadas de trabajo o la información general del fresco.

Una vez la aplicación esté subida en las respectivas tiendas de aplicaciones, la Universidad de Cantabria procederá a promocionar la aplicación a través de sus canales de redes sociales, página web, etc.

8. Referencias

[1] Vicmix Reality. Noticias de Realidad Virtual, Metaverso y Meta Quest. *¿Qué es la Realidad Aumentada?* Disponible en: https://vicmixreality.com/2021/08/16/que-es-la-realidad-aumentada-actualidad-visores-relevantes/

[2] Neosentec: *Realidad Aumentada.* Disponible en: https://www.neosentec.com/realidad-aumentada/ Proyecto subvencionado por el Principado de Asturias y fondos FEDER de la UE.

[3] Tom's Guide (Future US Inc.). *Best AR apps in 2021: Augmented reality comes to your phone.* Disponible en: https://www.tomsguide.com/round-up/best-ar-apps.

[4] Campus Cultural de la Universidad de Cantabria: *Luis Quintanilla, arte y memoria*. Disponible en: https://uc360.unican.es/tourquintanillaES/

[5] Unity Technologies. Página web: https://unity.com/es

[6] Unity Documentation: Unity User Manual (2019.4 LTS). Disponible en: https://docs.unity3d.com/es/2019.4/Manual/index.html

[7] Mundo.erp | Tecnologías ERP: *Desarrollo iterativo e incremental.* Disponible en: https://www.mundoerp.com/blog/metodologia-iterativa-o-incremental-gestion-proyectos/

[8] Sommerville, I. Ingeniería del Software (Vol. 9). Addison-Wesley ed: Reading, 2012.

[9] IBM Cloud Learn Hub: *Arquitectura de tres niveles*. Disponible en: https://www.ibm.com/mx-es/cloud/learn/three-tier-architecture

[10] Unity Manual: *Addressables.* Disponible en: https://docs.unity3d.com/Packages/com.unity.addressables@1.18/manual/index.html