UNIVERSITAT POLITÈCNICA DE VALÈNCIA

_ TELECOM ESCUELA TÉCNICA VLC SUPERIOR DE INGENIERÍA DE TELECOMUNICACIÓN

Mejora de la usabilidad de las aplicaciones del Museo de la Telecomunicación

María Martínez Simó

Tutor: Maria Carmen Bachiller Martin

Cotutor: Beatriz Rey Solaz

Trabajo Fin de Grado presentado en la Escuela Técnica Superior de Ingeniería de Telecomunicación de la Universitat Politècnica de València, para la obtención del Título de Graduado en Ingeniería de Tecnologías y Servicios de Telecomunicación

Curso 2020-21

Valencia, 23 de Junio de 2021

Escuela Técnica Superior de Ingeniería de Telecomunicación
Universitat Politècnica de València
Edificio 4D. Camino de Vera, s/n, 46022 Valencia
Tel. +34 96 387 71 90, ext. 77190
www.etsit.upv.es

Agradecimientos

La elaboración del TFG ha sido la última fase de mi formación como ingeniera técnica según el plan de estudios del grado de Tecnologías y Servicios de Telecomunicación. Por ello quiero agradecer a mi tutora del proyecto Carmen Bachiller y así como a mi cotutora Beatriz Rey, primero por haberme dado la oportunidad de realizar mi trabajo en el Museo de la Escuela y en segundo lugar el apoyo incondicional recibido por las dos al darme instrucciones en los momentos que no sabía como solucionar los problemas que iban surgiendo. Así como atendiendo constantemente a mis consultas de forma inmediata independientemente de las horas en que las realizaba, ello me ha permitido poder avanzar en el trabajo de forma ininterrumpida pudiendo cumplir mi objetivo de poder presentarlo en la convocatoria de julio. También agradecer al personal de la escuela el cual me ha facilitado el acceso para realizar el trabajo presencial. Por último, me gustaría agradecer a mi familia, en concreto a mi madre y a mi hermana por ayudarme a cumplir el horario de trabajo y animarme cuando no podía solucionar los problemas que iban surgiendo.

Resumen

En este trabajo de final de grado se han desarrollado una serie de proyectos con el fin de divulgar la existencia del Museo Vicente Miralles y los objetos tan singulares que están expuestos y que forman parte de la historia de la Telecomunicación mediante las nuevas tecnologías. Por una parte, se ha adaptado la versión virtual del Museo a los nuevos modelos, en concreto a las gafas Oculus Rift S y Oculus Quest que se han adquirido recientemente. Además, se ha modificado este proyecto para hacer posible su visualización desde la página web del Museo y así dotar de una nueva forma de visita gracias a las actuales técnicas de realidad virtual.

Por otra parte, se ha ampliado las funcionalidades de la aplicación de realidad aumentada, con la cual se pretende reforzar la visita presencial al Museo. Dicho de otro modo, a través de la visualización de vídeos, imágenes y audios donde se observa el funcionamiento de los objetos expuestos y se detalla algunos momentos históricos de la Telecomunicación en la Comunidad Valenciana para así dar a conocer la importancia de los ingenieros de telecomunicación en nuestra vida diaria.

Resum

En aquest treball de final de grau s'ha desenvolupat un conjunt de projectes amb el fi de divulgar l'existència del Museu Vicente Miralles y els objectes tan singulars exposats i que formen part de la història de la Telecomunicació a través de les noves tecnologies. Per una banda, s'ha adaptat la versió virtual del museu als nous models, les Oculus Rift S i Oculus Quest, que s'han adquirit recentment. A més, s'ha modificat aquest projecte per a fer possible la seua visualització des de la pàgina web del museu i així dotar d'una nova forma de visita gràcies a les actuals tècniques de realitat virtual.

Per l'altra, s'han ampliat les funcionalitats de l'aplicació de realitat augmentada, amb la qual es pretén reforçar la visita presencial del museu. Dit d'una altra manera, a través de la visualització de vídeos, imatges, i àudios on s'observa el funcionament dels objectes exposats i es detallen alguns dels moments històrics de la Telecomunicació en la Comunitat Valenciana per tal de donar a conèixer la importància dels enginyers de telecomunicació al voltant del nostre dia a dia.

Abstract

In order to spread the existence of the Vicente Miralles Museum and its unique objects that are part of the Telecommunications history, in this project I have carried out different tasks using new technologies. On one hand, as the museum is in possession of the newest versions of Oculus headsets, the Oculus Rift S and Oculus Quest, I have adapted the virtual reality project to be use with these. Moreover, this same project has been modified in order to be able to insert it into the official web page so it would increase the types of visits thanks to the new technical support of virtual reality.

On the other hand, the last task done has been to expand the functionalities of the augmented reality app which has the aim to strengthen the in-person visit. In other words, the goal is to make a little bit more known the historical moments of Telecommunications in our region. This is done through the display of videos, images and audios that explain how the unique objects on the exhibition

work and to emphasize the importance of the existence of telecommunications engineers in our daily lives.

Índice general

Índice de figuras

Índice de tablas

Listado de siglas empleadas

CMS Content Management System.

ETSIT Escuela Técnica Superior de Ingenieros de Telecomunicación.

JS JavaScript.

PHP Hypertext Preprocessor.

RA Realidad Aumentada.

RV Realidad Virtual.

Parte I

Memoria

Capítulo 1

Introducción

1.1. Objetivos

Este trabajo final de grado tiene como objetivo mejorar los trabajos de Realidad Virtual que se han hecho anteriormente sobre el Museo para así dar una mayor visibilidad a este. Además, se pretende hacer una divulgación de la historia de la Telecomunicación en la Comunidad Valenciana la cual está narrada a través de los objetos expuestos en el edifico de la ETSIT. Por un lado, en trabajos anteriores se ha creado una versión virtual del edificio 4D y de los expositores que conforman el Museo. Este proyecto se adaptó para usarse con las gafas de realidad virtual llamadas Oculus Rift. A partir de este trabajo se pretende realizar las siguientes tareas:

- Adaptar el trabajo a una versión web para poder ponerlo en la página del Museo y así que esté al alcance de todo el mundo.
- Recientemente el museo ha obtenido dos modelos nuevos de las gafas, en concreto el modelo Oculus Rift S y el modelo Oculus Quest. Es por ello que se debe adaptar el proyecto anterior para que pueda ser usado por estas.
- Hacer una recopilación de las nuevas adiciones al museo, hacer las fotografías y mediciones pertinentes para importarlos al proyecto web y al de las gafas de realidad virtual.

Por otro lado, también con trabajos anteriores se ha hecho una aplicación de realidad aumentada para dispositivos móviles. Con esta aplicación, a través de la cámara del dispositivo se reconoce una imagen y salta a un evento definido previamente. Además, la aplicación tiene otras funcionalidades, como son la información de la historia del Museo y actividades que pueden ser realizadas por los visitantes. Para mejorar la aplicación se han definido las siguientes propuestas:

- Mejorar la usabilidad de las funciones de la aplicación, comprobar los links asociados.
- Mejorar los siguientes eventos: código morse, el vídeo del telégrafo, el evento del televisor entre otros.
- Añadir nuevos eventos para completar la visita.
- Adaptar la aplicación para publicarla en AppStore y PlayStore.
- Publicar en la web la versión beta de la aplicación Andorid.

1.2. Motivación

El museo de la telecomunicación Vicente Miralles Segarra está ubicado en el edificio 4D Escuela Técnica Superior de Ingenieros de Telecomunicación, concretamente en la planta baja y primera planta del edificio sito en Camino de Vera S/n en Valencia.

La idea de crear un Museo de Telecomunicaciones en la escuela ETSIT nace alrededor del año 1992, cuando el profesor D. Luis Sempere conversó con D. Eduardo Fernández Nieto el cual poseía una colección de equipos antiguos de telecomunicación. De aquel primer contacto nació la primera donación de equipos de telecomunicación antiguos a la escuela. En un primer momento, los objetos fueron distribuidos por distintos lugares y despachos del edificio pero sin catalogar ya que en esa época no existía un proyecto expositivo ni divulgativo.

Con motivo del quinto aniversario de la creación de la escuela, los objetos fueron expuestos. Posteriormente en la celebración del 40 aniversario de la Universidad Politécnica de Valencia (1968-2008) se realizaron una serie de exposiciones donde cada escuela de la UPV tenía que aportar dos objetos relacionados con su área. La Escuela de Telecomunicaciones aportó unos heliógrafos de campaña de finales del siglo XIX y un teleimpresor de los años 40-50. Para aquella exposición se hicieron unas urnas para exponer los aparatos y con las cuales, una vez terminada la exposición de la UPV, se empezaron a usar en la colección de la ETSIT.

A partir del año 2010 un grupo de profesores se unió a la iniciativa del profesor Luis Sempere con el propósito de la creación del Museo que pretendía cumplir tres objetivos:

- La conservación del patrimonio tecnológico.
- La creación de un espacio expositivo.
- Y la difusión a la sociedad de la colección de aparatos de telecomunicaciones.

Actualmente cuenta con 750 piezas con gran valor patrimonial, ya que son la prueba de una de las mayores revoluciones tecnológicas de la humanidad. Esta colección es un reflejo de las sociedades de los siglos XIX, XX y XXI y su forma de comunicarse con sus semejantes, por ello es necesario conservarla para transferir esa parte de nuestra historia a las nuevas generaciones. Las piezas que componen la colección se han distribuido por cinco áreas diferentes: telegrafía, telefonía, equipos audiovisuales, de radiocomunicación y de instrumentación. A mediados del año 2014 se iniciaron los trámites para constituir el Museo y que formara parte de la Red de Museos de la Generalitat Valenciana. En el año 2015 se obtuvo el reconocimiento y se decidió ponerle el nombre de Museo de Telecomunicación Vicente Miralles Segarra en honor al primer ingeniero de telecomunicación valenciano.[1]

Actualmente se puede realizar la visita presencial del museo, para más detalles sobre el horario y el tipo de visitas guiadas se puede visitar la página web oficial del museo. [2]

El avance de las nuevas tecnologías hace que haya una necesidad de remodelar y modernizar los diferentes aspectos de nuestra vida, entre los que se incluye cómo se puede experimentar una visita a un museo.
Es por ello, que para mejorar la calidad del Museo y hacer posible su divulgación en la sociedad tecnológica en la que vivimos, en este trabajo se pretende hacer uso de tecnologías como es la realidad virtual para aumentar el alcance del museo y hacer posible su visita sin necesidad de estar

en las instalaciones de forma presencial, solo haría falta un pc y una conexión a internet. Además, se pretende complementar las visitas virtuales con la ayuda de una aplicación de realidad aumentada con la que se puede observar el funcionamiento de algunos aparatos expuestos y saber más de la historia de la Telecomunicación.

1.3. Plan de trabajo

El trabajo se ha ido realizando siguiendo los objetivos planteados inicialmente. Se ha previsto hacer el proyecto en cinco meses de trabajo, dedicando entre 4-5horas diarias de lunes a sábado para así presentarlo en la convocatoria de julio.

Para desarrollar el plan a seguir, se ha dividido el trabajo en tres tareas.

La primera tarea se refiere a los trabajos preliminares que han sido necesarios antes de poder empezar a desarrollar el proyecto. Estos trabajos preliminares son los siguientes:

- **Primera subtarea:** Lectura de trabajos anteriores.
  Los dos proyectos que componen mi TFG han sido anteriormente realizados por otros alumnos en años anteriores como trabajos de fin de grado. Para poder empezar a trabajar he tenido que leer los cuatro TFGs anteriores para saber lo que se ha hecho con anterioridad. La realización de esta subtarea ha costado alrededor de unas 20h en total.

- **Segunda subtarea:** Familiarización con Unity.
  Al empezar el proyecto no estaba familiarizada con el programa que se ha utilizado para la realización del proyecto de realidad virtual y aumentada, la cotutora me indicó un tutorial de edx de la universidad para formarme en el formato en el que se han realizado los anteriores trabajos. La formación en Unity duró cinco días, es decir, un total de 35 horas. [3]

- **Tercera subtarea:** Vuforia.
  Además del programa de Unity, se ha utilizado el software de Vuforia en conjunto con Unity. Me he tenido que documentar sobre esta herramienta para poder usarla en la aplicación de realidad aumentada. El tiempo empleado para la realización de estas tareas ha sido el siguiente.

Primera Tarea	Trabajos anteriors	Unity	Vuforia
Fecha Inicio	25 Enero	3 Marzo	31 Marzo
Duración(días)	10-15	8	3

Tabla 1: Plan de trabajo, Tarea 1

Dada la situación sanitaria actual provocada por el COVID-19 y debido a las restricciones de aforo, la tutora y yo acordamos que el desarrollo principal de los objetivos se haría de forma remota. Por tanto, la segunda tarea se ha realizado de forma remota, conectándome desde mi domicilio al ordenador del museo situado en la ETSIT, trabajando con los siguientes proyectos:

- **Proyecto Web.** Se pretende crear una versión en plataforma web del proyecto de realidad virtual para poder integrarlo en la página web del Museo. Además, se actualizará este proyecto con los nuevos expositores del Museo.
- **Proyecto Oculus.** Se pretende solucionar problemas y errores existentes del proyecto y al igual que con el proyecto web, añadir los nuevos expositores. También adaptarlo para las gafas Oculus Rift S y Oculus Quest.
- **Proyecto aplicación AR** Mejorar la usabilidad de la aplicación y algunos eventos ya existentes los cuales no se visualizan de manera correcta. También se pretende añadir eventos nuevos y preparar la aplicación para su publicación en App Store y Play Store.

Segunda Tarea	Proyecto Web	Proyecto Oculus	App AR
Fecha Inicio	9 Marzo	15 Abril	5 Abril
Duración(días)	75	62	47

Tabla 2: Plan de trabajo, Tarea 2

La tercera y última tarea del plan se ha realizado de forma presencial en el museo, ya que se necesitaba realizar trabajo de campo. El trabajo hecho presencialmente ha sido:

- **Medición de expositores nuevos.** Se ha comprobado las dimensiones de los expositores nuevos y se han tomado las fotografías pertinentes para luego poder implementarlas tanto en el proyecto web como en el de Oculus
- **Compilación Realidad Virtual.** Se ha trabajado presencialmente para probar el proyecto con las gafas Oculus Rift, Oculus Rift S y Oculus Quest. Cada tipo de gafa necesita de una compilación e instalación diferente.
- **Aplicación AR.** Desarrollo de la aplicación de realidad aumentada, así como de las pruebas hechas para perfeccionar su funcionamiento.

Tercera Tarea	Expositores Nuevos	Compilación VR	Pruebas aplicación AR
Fecha Inicio	19 Abril	26 Abril	3 Mayo
Duración(días)	12	15	20

Tabla 3: Plan de trabajo, Tarea 3

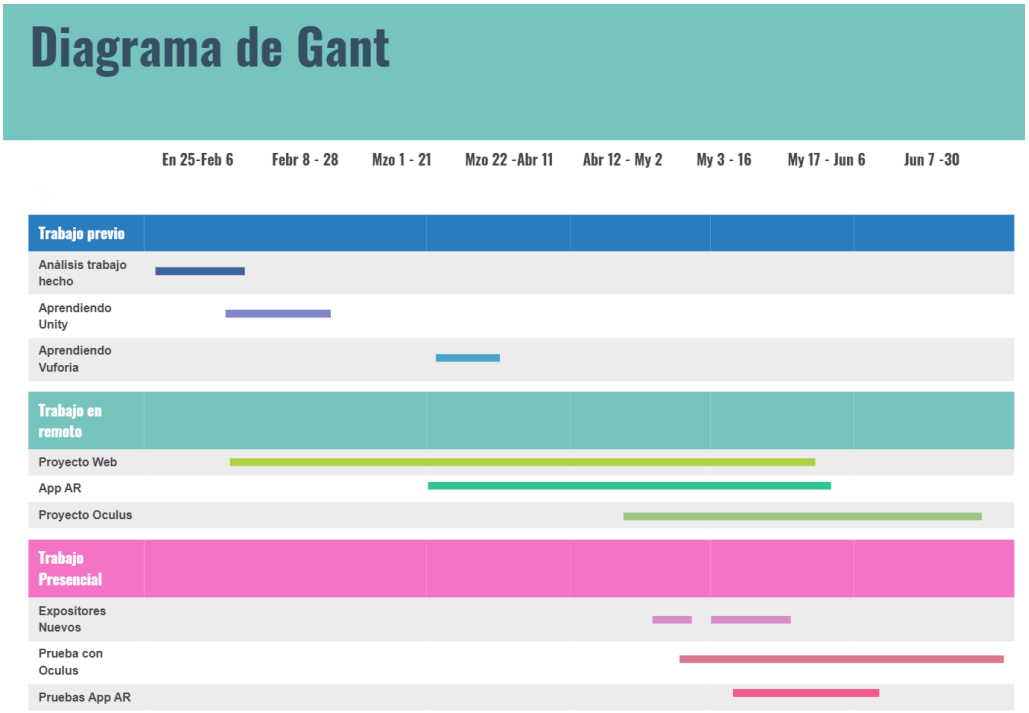

Figura 1: Diagrama de Gant

Capítulo 2

Museos y el estado del arte

El museo de ciencia y tecnología de Munich también conocido como Deutsches Museum von Meisterwerken der Naturwissenschaft und Technik, está situado en una isla en el río Isar que atraviesa el centro de Munich Es el museo más grande del mundo de esta temática y se sitúa como uno de los museos más visitados el mundo, es por ello un museo de referencia. Por su dimensión es necesario utilizar alrededor de 8 días para realizar la visita completa del mismo.[4]

El propósito de este museo es facilitar al visitante el acceso a la ciencia y a la ingeniería, por ello debemos tomarlo como referente para mejorar el museo de la escuela de ingenieros de telecomunicaciones Vicente Miralles.

Los visitantes del Deutsches Museum pueden encontrar simuladores, así como pantallas táctiles, y presentación de fenómenos físicos en determinados horarios. En cuanto a los objetos del museo se hallan recopilados más de 60.000 objetos y unos 850.000 libros y textos originales sobre las ciencias, materiales y producción, energía, transporte, comunicaciones e información.

También existen exposiciones permanentes en torno a los siguientes temas: Técnica agrícola y de alimentos, las cuevas de Altamira, radioaficionados, astronomía, minería, construcción de puentes, química, la imprenta, el ferrocarril, tecnología energética, el petróleo, gas natural, el péndulo de Foucault, la geodesia, el vidrio, la informática, la cerámica, las carrozas y bicicletas, los motores, la aviación, los elementos de maquinaria, las medidas y peso, gabinete matemático, los metales, la microelectrónica, los instrumentos musicales, la historia del museo, el papel , la farmacia, la física, la técnica de las corrientes fuertes eléctricas, los juguetes técnicos, las técnicas de la telecomunicación, la técnica textil, la construcción de túneles, el medio ambiente, la construcción hidráulica, los instrumentos.

Entre los elementos más destacados del museo se encuentran el primer avión a motor de los hermanos Wright, el primer automóvil de Kalrl Benz o el submarino U1.

Debido a la gran extensión del museo y al gran volumen de objetos que en él se pueden encontrar resulta bastante pesado si se pretende ver todo su contenido, por ello el museo ha hecho una app interactiva para usarla cuando vas a visitarlo. En ella hay diferentes recorridos. Se elige el recorrido que se quiere hacer y este te indica el lugar donde empieza. Una vez allí, en el móvil se muestra un vídeo explicativo de esa exhibición y luego mientras haces el recorrido, van saliendo más contenidos. Esta app está diseñada para detectar una señal de bluetooth y así mostrar un contenido u otro.

Además, este museo ha hecho una visita virtual. Esta creada a partir de la toma de imágenes del museo físico, el cual lo han reconstruido como los pasillos y aparatos mostrados. Es decir, su funcionamiento es parecido al que ha hecho Google con la herramienta Google maps, imágenes estáticas las cuales se ponen en orden y te mueves con el ratón y pulsando las flechas. Esta realidad virtual es diferente a la que se ha hecho en el museo de la ETSIT, ya que en nuestro caso se ha construido de cero una versión virtual del edificio y de los expositores, todo en un entorno gráfico, no se ha trabajado a partir de la toma de imágenes del edificio.

Por otra parte, dentro del museo, hay una exhibición de RV, esto quiere decir que permite a los visitantes crear una nueva experiencia a través del uso de las nuevas tecnologías como el modelado 3D y gafas con controladores.

Asimismo, hay que señalar la existencia del Museo Nacional de Ciencia y Tecnología de España (MUNCYT), el cual es un museo de titularidad estatal adscrito a la Secretaría General de Investigación del Ministerio de Ciencia e Innovación y gestionado por la Fundación Española para la Ciencia y la Tecnología (FECYT). Actualmente custodia y conserva más de 18000 objetos de patrimonio histórico científico y tecnológico.[5]

El proyecto de creación comienza en 1962, pero este proyecto no tuvo continuidad. La iniciativa se retomó a partir del año 1970, pero no fue hasta el año 1975, cuando comenzaba el nacimiento de la democracia, cuando el Ministerio de Presidencia del Gobierno instituye un Patronato interministerial para la Ciencia y la Tecnología, surgiendo contactos a partir del año 1976 con la UNESCO y creando una dinámica de viajes a otros centros como los canadienses, estadounidenses y europeos con el objeto de recabar contactos y modelos para crear el museo español.

Este museo, tiene dos versiones de visita virtual, una en la sede de A Coruña y la otra en la sede de Alcobendas. Estas están hechas de forma similar a la visita virtual del museo de Munich donde se ha reconstruido las instancias del museo a partir de fotografías de este y se puede navegar por los pasillos a través de pulsar flechas en el suelo.

Capítulo 3

Herramientas de desarrollo

En este capítulo se va a proceder a introducir el software y herramientas utilizadas tanto para desarrollar el proyecto de realidad virtual del museo como para la aplicación de realidad aumentada. Se van a exponer las razones por las cuales se decidió usar estos softwares como sus características principales.

3.1. Unity

En los proyectos anteriores se decidió usar Unity como motor de desarrollo del proyecto ya que se necesitaba un entorno de desarrollo 3D.

Los motivos por las cuales se eligió este software fueron las siguientes.

Unity es un software que principalmente se ha usado para crear y desarrollar videojuegos el cual tiene la posibilidad de renderizar gráficos en 2D y 3D. Además, tiene un motor físico que imita las leyes de la física del mundo real y otras opciones como el uso de animaciones, sonidos y la programación del juego a través de scripts en lenguaje C, C++ y C#.

Este software fue creado por la empresa Unity Technologies y su primera versión fue lanzada en 2005. Desde ese momento, se ha creado una gran comunidad de usuarios de esta herramienta. Por otra parte, Unity permite el acceso gratuito a toda su documentación y ha creado foros y comunidades tales como el *UnityANSWERS* donde los usuarios comparten sus preguntas y/o errores que les van surgiendo mientras crean su proyecto.

Estos foros pueden ser de mucha ayuda para los principiantes con el programa ya que una duda que puedas tener, alguien ya la ha tenido antes. Además, es muy fácil crearse una cuenta en estos foros y buscar información en la documentación de Unity dependiendo de qué versión se esté utilizando.

Otra de las razones por las que se eligió este motor de desarrollo fue su compatibilidad en diferentes plataformas. Unity se puede usar en plataformas tan diferentes como son IOS, Android, Windows, Mac OS, Linux entre otras. Cabe destacar que también es compatible con plataformas de realidad virtual como son las gafas Oculus o Gear VR. Esto es un punto a favor de usar esta herramienta ya que el museo tiene a su disposición las gafas Oculus.

3.2. Vuforia

Como se ha explicado en puntos anteriores, se va a trabajar sobre una aplicación de realidad aumentada que ya está creada, donde se pretende mejorar su funcionalidad, adaptarla para dispositivos IOS y publicarla tanto en la Play Store como en el App Store.

Para cumplir con los objetivos establecidos, se ha seguido usando el software de Vuforia, como se ha hecho en los TFGs anteriores, siendo este un Software Development Kit (SDK) que permite a dispositivos móviles la creación de aplicaciones de realidad aumentada. Esto se refiere a que permite crear una aplicación que reconoce objetos en 2D y 3D en tiempo real. Los tipos de objetos que puede reconocer estos pueden ser VuMark e ImageTargets los cuales son objetos en 2D o un objeto real como puede ser un cilindro en 3D.

Cabe destacar que, este SDK necesita una licencia para su uso. Hay una opción de descarga gratis la cual ha sido la utilizada para la creación de esta aplicación.

Paralelamente, se eligió un entorno de desarrollo que fuese compatible con el SDK de Vuforia. Se decidió usar el software de Unity, el mismo que se ha usado para crear el proyecto de realidad virtual. Las razones por las cuales se decidió son parecidas a las expuestas en el apartado anterior.

La comodidad de poder pasar un mismo proyecto a diferentes plataformas como puede ser Android e IOS. La programación de los eventos que ocurren a través de scripts en lenguaje C#, los cuales se editan en el programa Visual Studio.

Es por ello que se ha desarrollado la aplicación de realidad aumentada en las plataformas de Unity y usando el SDK de Vuforia para el reconocimiento de imágenes y activación de eventos simultáneamente.

3.3. Oculus

Las gafas Oculus fueron creadas por la empresa Oculus VR, fundada en 2012 por Palmer Luckey en California, Estados Unidos. En este mismo año, la empresa lanzó un casco de realidad virtual llamado Oculus Rift, creado para dotar de una experiencia más inversiva y realista de videojuegos. Este producto tuvo mucho éxito en el mercado y se lanzaron para los desarrolladores dos modelos nuevos, el Oculus VR DK1 y el Oculus VR DK2.

En marzo de 2014, Mark Zuckerberg, el CEO de Facebook, compró las acciones de Oculus VR y luego en 2015 esta compañía adquirió una start up británica llamada Surreal Vision, la cual, estaba especializada en la reconstrucción 3D y realidad mixta.

En 2016, se presentó al mercado una nueva versión del casco Oculus. Esta versión se diferencia de la anterior por su nuevo diseño, la introducción de pantallas VR especializadas, audio posicional y un sistema de seguimiento por infrarrojos. Desde este año la empresa Oculus VR ha ido sacando nuevos tipos de gafas, entre ellas las Samsung Gear VR, las cuales son una colaboración con Samsung y otras gafas como son las Oculus Rift S y Oculus Quest.

Actualmente, el museo posee tres modelos diferentes de estas gafas. En primer lugar, tenemos las Oculus Rift, las cuales se usaron para el proyecto anterior. Estas están compuestas por las gafas que se conectan al ordenador, dos mandos joystick y dos sensores de movimiento. Este modelo para poder reconocer el entrono usa los sensores de movimiento junto con la tecnología llamada

’Outside in’. También tiene instalados dos pequeños altavoces en cada lateral del headset.

Figura 2: Oculus Rift

En segundo lugar, tenemos las gafas Oculus Rift S, las cuales son la versión más nueva de las anteriores. Este modelo contiene una mejor resolución de imágenes, una nueva tecnología llamada ‘Inside Out’ con la cual se reconoce el entorno que rodea al usuario y algoritmos de visión computacionales que determinan la posición del headset.

Figura 3: Oculus Rift S

Y por último, tenemos las gafas Oculus Quest. Estas están compuestas por el headset y dos controladores, no necesitan de una conexión constante a un PC para poder usarlas. Al igual que las Oculus Rift S, us la tecnología ’Inside Out’ para reconocer el entrono. Además tiene integrado un software de ’Hand Tracking’ para poder localizar y seguir el movimiento de los controllers. Por último, cabe añadir que el headset contiene unos altavoces para sumergir al usuario en el entorno virtual.

Figura 4: Oculus Quest

Capítulo 4

Realidad Virtual

En este capítulo voy a exponer qué es la realidad virtual y cómo se hecho una versión virtual del Museo para mejorar la experiencia de un visitante.

4.1. Descripción del proyecto anterior

La realidad virtual se puede definir como un conjunto de escenas que tratan de imitar al mundo real, las cuales están creadas a partir de programas informáticos. Esta realidad es totalmente perceptiva, no contiene ningún respaldo en formato físico, es decir, no se pueden tocar ni sentir los objetos que existen en una escena virtual.

Para poder experimentar el mundo virtual se necesita de un respaldo de dispositivos externos, como son las gafas de realidad virtual. Así pues, el usuario puede verse sumergido visual y auditivamente en el entorno gráfico. Además, con la ayuda de unos controladores existe la posibilidad de interactuar con el entorno virtual.

En esta primera parte del proyecto he trabajado a partir de dos proyectos de final de grado anteriores [6] [7]. El objetivo de los trabajos anteriores era crear una versión virtual e interactiva del edificio 4D donde se encuentra el museo Vicente Miralles Segarra. Para conseguir esto, se ha usado el programa Blender, herramienta que permite modelar objetos en tres dimensiones y los planos del edificio para reproducir las estancias del edificio. Además, con esta misma herramienta se modelaron otros objetos como son los extintores, bancos, sillas y mesas. Cabe destacar que, aunque estos objetos pueden parecer innecesarios, ya que no forman parte de la exposición, son indispensables para dar una visión más real.

Una vez hecho el modelado, se eligió trabajar con Unity con el objetivo de crear una visita virtual interactiva y accesible a diferentes plataformas. Dentro de Unity, se han usado objetos llamados GameObject 3D para crear lo que son los expositores y carteles y se han añadido las texturas respetando los colores originales. Para añadir los diferentes objetos que forman parte de la historia de la Telecomunicación y que están mostrados en la exposición, se han incluido en el proyecto de Unity a través de fotografías individuales de cada expositor.

Para completar la experiencia de RV, el museo compró unas gafas de RV de la marca Oculus, estas se llaman Oculus Rift. Es por ello que se adaptó este proyecto para poder usarlas. Para hacerlo, hay que descargarse los assets de Oculus, llamados Oculus Integration Tools, desde Unity y luego,

posicionar la cámara y controlador de las Oculus dentro del museo virtual y configurarlo para que se pueda usar en este nuevo entorno.

Finalmente, se añadió la posibilidad de que el usuario interactuara en el entorno haciendo saltar vídeos o audios, los cuales forman parte de la aplicación de móvil del museo.

4.2. Proyecto Web

Se pretende adaptar el proyecto existente para poder implementarlo en una página web, en concreto, en la página web del Museo de la ETSIT.

En primer lugar, antes de empezar a trabajar dentro del entorno de Unity, se ha comprobado la versión con la que se hizo el trabajo anterior, en este caso se utilizó Unity 2018. La versión con la que se hace un proyecto nos define qué versiones se van a poder utilizar sobre este, ya que Unity solo da la opción de utilizar la misma o una versión superior.

Al abrir el proyecto con Unity 2018, el programa da muchos errores sobre las API y otras librerías que están en desuso. Por lo tanto, para no tener estos problemas, decidí usar la versión 2019.4 la cual ya estaba instalada en el ordenador.

Como no había utilizado anteriormente esta herramienta realicé un tutorial de Edx recomendado por la cotutora del TFG. [3] A partir de este creé un juego simple de una bola que se puede mover por un plano con la intención de familiarizarme con Unity y ver cuánto podía ocupar este juego simple en la versión compilada a Web.

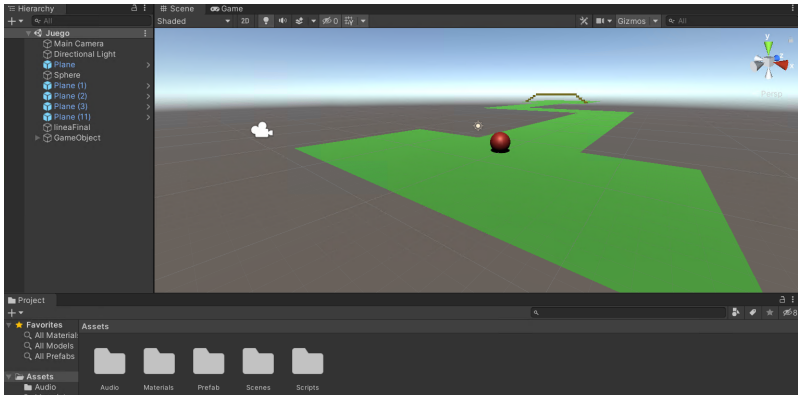

Figura 5: Juego de la Bola

Una vez hecho el juego de la bola intenté compilar a web desde Build Setting>WebGL>Build. Al hacer esto salían unos errores, los cuales parecían bastante extraños ya que dentro del entorno de Unity en el estado PlayMode y al compilarlo para entorno de PC no salía ningún error.

Como Unity tiene una gran comunidad de usuarios, busqué en sus foros este problema de compilación para web y me encontré que este programa en algunas versiones tiene problemas para compilar a la plataforma WEBGL.[8]

Con la ayuda de la cotutora, llegamos a la conclusión que esto se debía a un bug del programa y que la única solución era cambiar de versión. Entre la cotutora y yo, buscamos una versión con la que se pudiera compilar a web. Tras realizar diferentes pruebas con diferentes versiones, se vio que la compilación a plataforma WebGL funcionaba con Unity 2019.2.0f1.

4.2.1. Desarrollo proyecto web

Una vez elegida la versión y comprobado que se podía utilizar para compilar a web, se procedió a trabajar sobre el proyecto.

Cuando se abre el proyecto, salen algunos errores dentro de los scripts, a causa del cambio de versión desde el 2018 a 2019.2. Esto se debe a que con la mejora de las versiones, se han cambiado o eliminado el nombre de las clases que se usan en los scripts. Para resolverlos hay que ir a cada uno de los archivos y editar lo que haga falta. En concreto, salían errores mostrados en la Figura 6

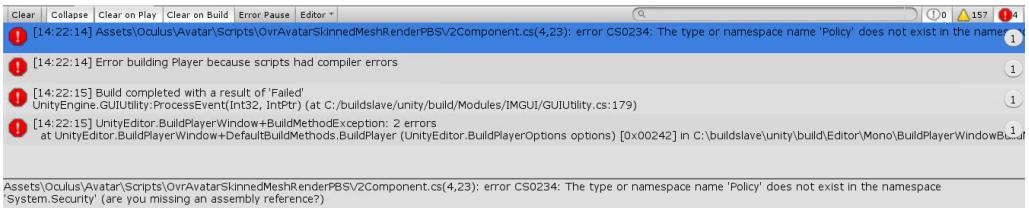

Figura 6: Errores Policy Unity 2019.2

El primer error viene del GameObject *ParticleRenderer*, el cual dejó de ser usado en la versión 2018.3, para resolverlo se cambia este Game Object por el *ParticleSystemRenderer*.

Por otro lado, aparecen errores que se generan por los controladores de las Oculus. Como se ha expuesto antes, se está adaptando el trabajo hecho para Oculus a formato web. Es por eso que está integrado el controlador de las Oculus a este.

Aquí, el problema parece ser que los assets y scripts de las OVRplayer están en desuso. Las librerías de Oculus se integraron al anterior proyecto sobre el año 2018-2019, parece ser que Unity solo acepta los assets del OVRplayer más recientes. Para la versión web de la visita virtual no nos interesa tener integradas estas librerías y como ocupan mucho espacio, se han eliminado de este proyecto.

Ahora, para poder navegar a través de las instalaciones virtuales del museo, se ha activado la opción FPScontroller, con la cual se puede controlar la posición y rotación con el ratón y el teclado.

4.2.1.1. Optimización peso

Se pretende hacer una optimización general de este proyecto en Unity para que pese lo menos posible y se pueda poner en la página web. En un primer paso, se ha quitado todas las librerías de las Oculus y esto ha permitido disminuir la ocupación total de la compilación a WebGL.

Para saber qué objetos ocupan más tamaño, debemos mirar un archivo llamado Editor.log, este se crea cuando se compila un proyecto. Hay que compilar desde Build Setting>WebGL>Build,

esperar a que termine el programa, cosa que puede tardar entre unos 10 y 30 minutos. Una vez terminado, desde la ventana de Console abrimos el Log y lo guardamos en un archivo de texto en el ordenador. Lo vemos explicado en la siguiente Figura 7.

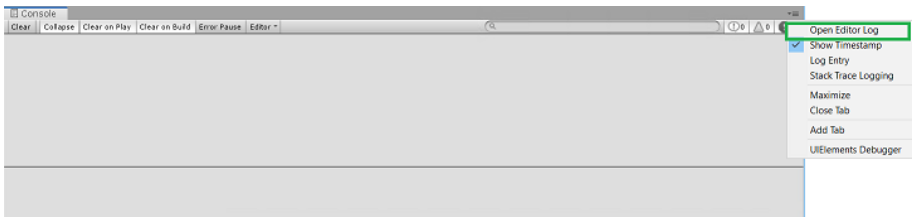

Figura 7: Acceso editor log desde Console

En el momento que está guardado, se abre el archivo del Log y se busca la palabra *Build Report*, así aparecerá un resumen de los archivos que contiene el proyecto y estarán listados en orden de su tamaño, además se encuentra detallada la ruta donde están guardados. De esta forma, se pueden identificar los archivos más grandes y así poder reducir su tamaño. Ejemplo de ello es la siguiente imagen, Figura 8.

```
Build Report\par
Uncompressed usage by category (Percentages based on user generated assets only):\par
Textures                498.4 mb\tab  56.4% \par
Meshes                  18.2 mb\tab  2.1% \par
Animations              0.0 kb\tab  0.0% \par
Sounds                  7.1 mb\tab  0.8% \par
Shaders                 759.9 kb\tab  0.1% \par
Other Assets            345.7 mb\tab  39.1% \par
Levels                  2.5 mb\tab  0.3% \par
Scripts                 1.1 mb\tab  0.1% \par
Included DLLs           9.4 mb\tab  1.1% \par
File headers            48.9 kb\tab  0.0% \par
Total User Assets       883.2 mb\tab  100.0% \par
Complete build size     1.4 gb\par
Used Assets and files from the Resources folder, sorted by uncompressed size:\par
 52.7 mb\tab  3.7% Assets/Resources/Video/a bordo del castillo de olmedo\par
 44.6 mb\tab  3.1% Assets/Resources/Video/Magnetofono\par
 43.5 mb\tab  3.1% Assets/Resources/Video/Gramofono\par
 43.1 mb\tab  3.0% Assets/Resources/Video/Picu\par
 42.9 mb\tab  3.0% Assets/Resources/Video/CasetteCinta\par
 40.6 mb\tab  2.9% Assets/Resources/Video/Fonografo\par
 28.9 mb\tab  2.0% Assets/Resources/Video/VideoTransparente_v6\par
 22.5 mb\tab  1.6% Assets/Resources/Video/Un dos tres... responda otra vez - Don Cicuta y Kiko Ledgard - El terror\par
\highlight2  16.8 mb\tab  1.2% Assets/Alex/Fotos objetos/Magnetofono.jpg\par
\highlight0  16.7 mb\tab  1.2% Assets/ETSIT/Universidad.fbx\par
 16.1 mb\tab  1.1% Assets/Resources/Video/Sebastian olive telegrafista\par
 10.5 mb\tab  0.7% Assets/Resources/Video/TelefonoQuijote\par
 \highlight2 5.3 mb\tab  0.4% Assets/Objects/Textures/Carteles/cartel_info_225x115_new222opt.jpg\highlight0\par
 5.3 mb\tab  0.4\highlight2 % Assets/Objects/Textures/Carteles/ChartOfElectromagneticRadiation.jpg\highlight0\par
 5.3 mb\tab  0.4\highlight2 % Assets/Objects/Textures/Carteles/morse_manipulador.jpg\par
 5.3 mb\tab  0.4% Assets/Alex/Fotos objetos/Radio Onteniente 4.jpg\par
 5.3 mb\tab  0.4% Assets/Alex/Fotos objetos/Radio Onteniente 3.jpg\par
\highlight3  5.3 mb\tab  0.4% Assets/Objects/Textures/Carteles/telegrafistas.jpg\par
 5.3 mb\tab  0.4% Assets/Objects/Textures/Carteles/radio_onteniente.jpg\par
 5.3 mb\tab  0.4% Assets/Objects/Textures/Carteles/morse_receptor.jpg\par
 5.3 mb\tab  0.4% Assets/Objects/Textures/Carteles/instrumentacion.jpg\par
\highlight0  \highlight3 5.3 mb\tab  0.4% Assets/Objects/Textures/Carteles/centralitas_3.jpg\par
```

Figura 8: Texto Build Report

Se puede observar que los archivos con más peso son los vídeos seguidos de las imágenes de los carteles y expositores. En cuanto a los vídeos se ha decidido eliminarlos completamente ya que no forman parte del objetivo de esta parte del proyecto.

Seguidamente, para seguir optimizándolo se ha bajado la resolución de las imágenes de forma individual. Para que la reducción de la calidad no afectara a la experiencia del usuario de forma general, se ha optado por bajar la resolución solo de algunas imágenes para no disminuir la calidad de visualización.

Para hacerlo, se ha abierto una nueva vista Proyecto, que es donde se encuentran las carpetas del proyecto creado. Se ha seleccionado la imagen deseada y desde la vista de Hierarchy se ha cambiado su resolución, originalmente todas estaban con un tamaño de 4096. Dependiendo de la fotografía se ha dejado el valor original o se ha reducido.

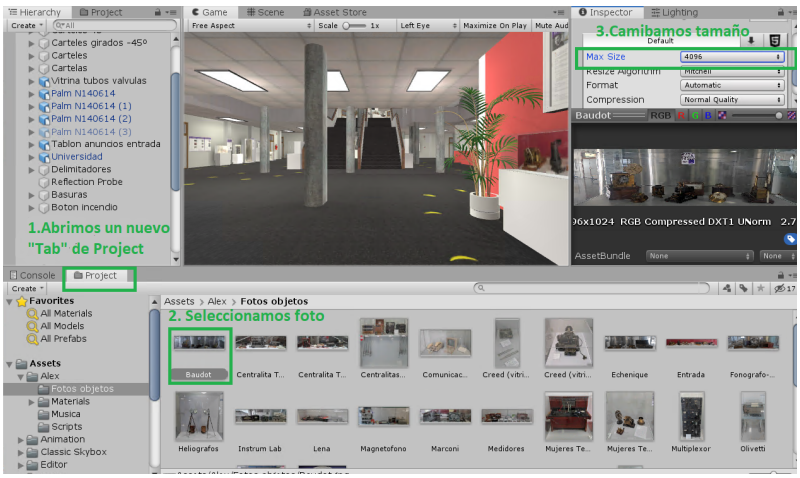

Figura 9: Cambio resolución imágenes

Al hacer esto con todos los objetos y archivos que tienen un tamaño mayor, se ha conseguido reducir la visita compilada desde unos 400 Mb que tenía originalmente a menos de 100 Mb. Esta reducción nos facilitará el subir el archivo de la visita a la página web.

4.2.1.2. Plataforma WebGL y la iluminación

Una vez se tiene el proyecto compilado, se han hecho pruebas de forma local para comprobar cómo quedaría en la página web, es decir, cómo funciona la navegación con el teclado y cómo se ven los objetos expuestos. Para ello, se ha hecho uso de un servidor local en concreto el Tomcat.

Para instalarse este servidor, se descarga la aplicación Xampp desde la página oficial y se aceptan los términos de uso. Seguidamente, en nuestro equipo aparecerá una nueva carpeta dentro de una ruta similar a C:/xampp/tomcat/webapps Aquí guardamos la carpeta del output de la compilación del proyecto web. Para probarlo, en primer lugar, hay que abrir la aplicación de Xampp e iniciar el servidor Tomcat. En segundo lugar, se abre un navegador preferentemente Google Chrome y entramos en la siguiente dirección: http://localhost:8080/<nombre carpeta>/index.html

Al entrar en el enlace, el juego puede tardar alrededor de un minuto para cargarse en el navegador, mientras tanto aparece el logo de Unity. Una vez cargada la página, se puede ver que no hay luces en la versión compilada para Web mientras que el mismo proyecto se compila para versión PC, se ve una correcta iluminación de todo el edificio y los expositores se ven sin problema. Vemos la comparación en las siguientes Figuras 10 y 11.

Figura 10: Compilación a PC

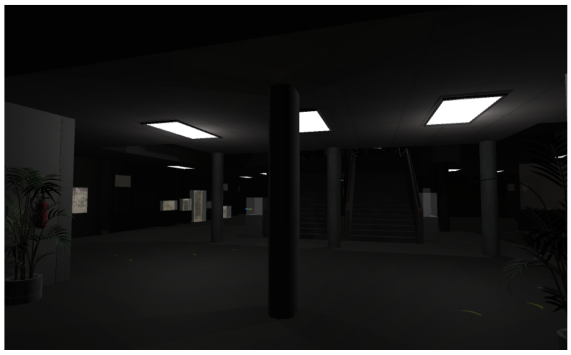

Figura 11: Compilación a WebGL

Como se puede observar, en una versión hay luces y se ven todos los elementos correctamente mientras que en la otra no se observa ningún tipo de iluminación aparte de las luces del techo que no son suficientes para hacer visible la escena.

Cabe destacar que Unity es un programa hecho para el desarrollo de videojuegos, por lo tanto, estas herramientas son muy útiles a la hora de crear un ambiente misterioso típico de los juegos de hoy en día. Aun así, podemos usar estas herramientas a nuestro favor para crear una iluminación lo más cercana al mundo real posible.

Según la documentación [9], los objetos de luz de Unity son:

- *Directional Light*: un GameObject que pretende imitar a la luz del sol. Suelen ser el objeto elegido para dotar de luz escena en Unity. Una de sus características principales es que la fuerza en que ilumina este objeto no disminuye con la distancia, es decir, actúa como la luz solar. Normalmente solo hay un GameObject de este tipo por escena, para más iluminación se puede combinar el uso de point lights o hacer un cocinado de luces, esto se explicará más adelante. Como vemos en la Figura 12, con solamente la directional light se consigue iluminar toda la escena y los objetos dentro de ella.

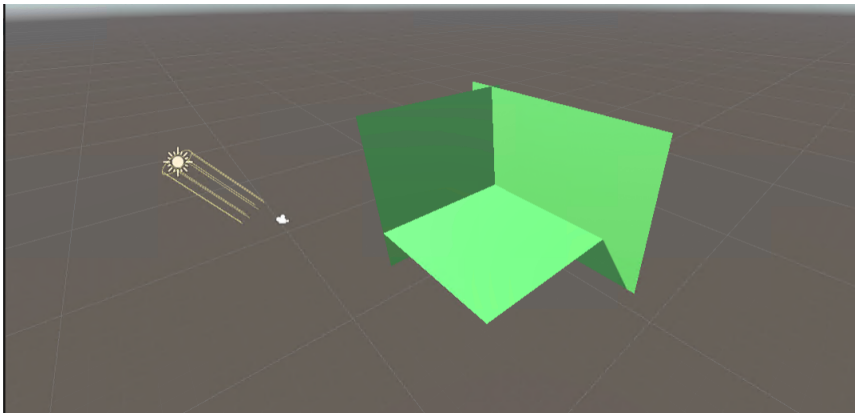

Figura 12: Ejemplo funcionamiento Directional Light

- *Point Lights*: se sitúan en un punto del espacio y emiten en todas las direcciones de forma equivalente. La intensidad de la luz disminuye proporcionalmente a la distancia y llega a cero en un punto determinado. Además, esta imita el comportamiento de las ondas de luz en el mundo real ya que la intensidad es inversamente proporcional a la distancia al cuadrado. Se suelen utilizar para imitar el comportamiento de lámparas u otras fuentes de luz locales a la escena. También, se pueden controlar a partir de los scripts para hacer que imiten una chispa en un momento concreto.

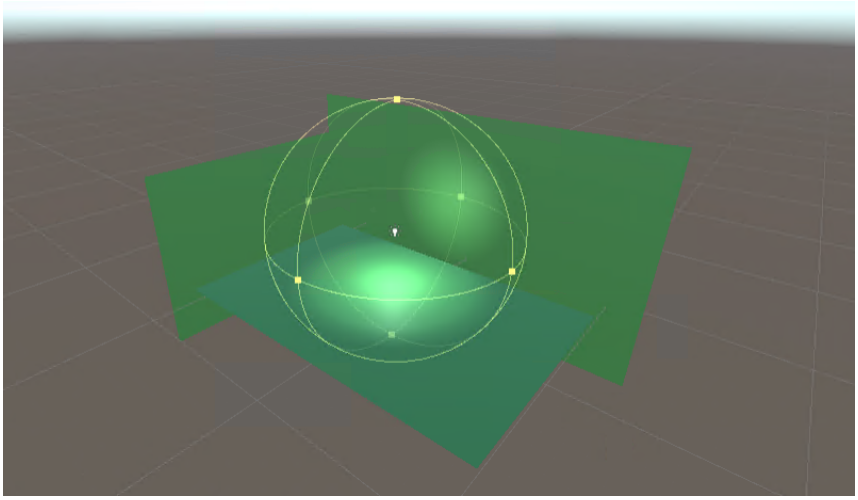

Figura 13: Ejemplo funcionamiento Point Light

- *Spot Lights*: al igual que las *point lights* se sitúan en un punto específico del espacio desde el que se emite la luz y esta disminuye proporcionalmente a la distancia. A diferencia de la anterior, emite luz dentro de un rango específico definido por un ángulo. Si se aumente este ángulo, se incrementa el ancho del cono que forma la luz y se hace más grande la zona de disipación, la cual se llama ‘penumbra’.

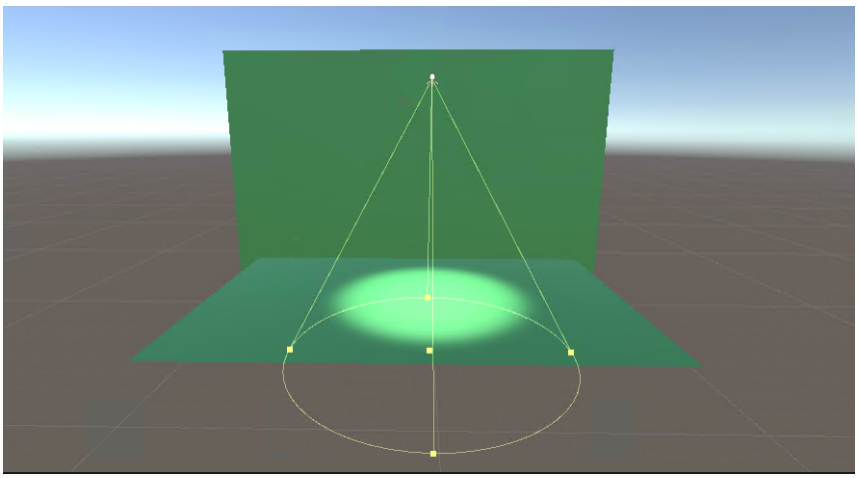

Figura 14: Ejemplo funcionamiento Spot Light

- *Area Lights*: se define por un rectángulo en el espacio, este tipo de objeto-luz emite unifor-

memente en todas las direcciones del espacio sobre su área pero solamente por una de las caras del rectángulo. Dado que, este tipo de luz, ilumina a un objeto u objetos desde diferentes puntos al mismo tiempo, la formación de sombras suele ser más suave que si se utilizase los otros tipos de luz. Es importante destacar que las área light no se verán a no ser que se haga un lightmap en la escena y se indique como estático los objetos a los que va a afectar esta luz.
Las *area lights* se suelen utilizar para dar a la escena un ambiente que se adecúe al tipo de proyecto. Como Unity es uno de los programas más utilizados para el desarrollo de juegos, esta luz ayuda a crear ese ambiente misterioso, cambiando los colores de estas luces, para dar una mejor experiencia de usuario. En nuestro caso en concreto, no se usan estas luces con colores como puede ser el rojo, naranja o azul ya que nuestra intención es tener una experiencia lo más parecido a la realidad posible.

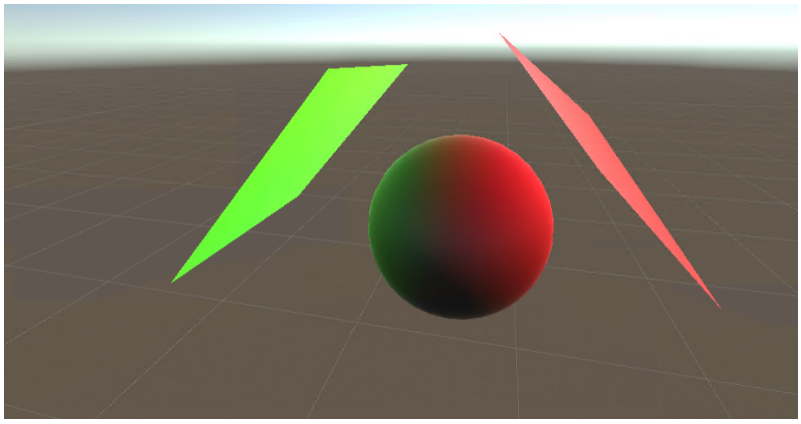

Figura 15: Ejemplo funcionamiento Area Light

4.2.1.3. Identificación luces en el proyecto

Una vez ya sabemos los tipos de luz que existen en Unity, vamos a identificar los elementos que iluminan nuestra escena.

SPOT LIGHT
En primer lugar, tenemos un spot light situado en el techo del edificio para imitar que entre luz por los cristales situados en el techo. Lo vemos en la Figura 16.

Esta luz ilumina parte de las escaleras y suelo de la planta baja, simulando la luz del sol que entra por los cristales del techo.

AREA LIGHT
En segundo lugar, podemos ver que hay luces área situadas como las luces del techo del edificio. Estas se pueden apreciar en la imagen donde se ve el edificio todo oscuro (Figura 11), se puede apreciar que las luces del techo funcionan, es decir, dan un poco de iluminación a la escena son de tipoa aea light, las cuales suelen ser planos que emiten luz por una parte y en todas las direcciones. Pero no tienen mucha fuerza de para iluminar el escenario completo. Originalmente, no se crearon estas bombillas de luz para que fueran las encargadas de dotar la iluminación principal, sino que

Figura 16: Spot Light en el proyecto

se crearon para dar un atisbo de realidad y crear la sensación de estar andando por los pasillos del museo. Es por eso que estas no iluminan mucho la escena por si mismas, pero pueden ayudar para dar luz a la escena en un futuro.

POINT LIGHTS
Paralelamente, hay dos point lights situados en la entrada antes de los pilares y delante de las escaleras, que se pusieron para dotar de una mayor claridad nada más empezar el juego. En nuestro caso, tienen tan poca intensidad y rango que apenas se ven sus efectos, igual que ocurre con la spot light mencionada anteriormente.

DIRECTIONAL LIGHT
También hay una directional light, la cual está presente en todas las escenas de Unity. Esta simula la luz del sol y por ello no importa donde esté situada exactamente pero sí su orientación.

En la figura 17, se aprecia que entra la luz pero esta no afecta a los objetos que no son tocados por ella directamente, al contrario de lo que ocurre con la luz del sol natural, que entra por la ventana e ilumina toda la habitación y los objetos que hay dentro aunque no les dé la luz directamente.

Para imitar el efecto de la luz natural dentro del programa, se debe hacer un “baked” o cocinado de la escena. Cabe destacar que esto se puede hacer con los elementos que son estáticos ya que consiste en hacer un cálculo del lightmap (mapa de luz) en los objetos de la escena. Para hacer esto, indicamos en la ventana de Inspector, los objetos que son estáticos, como es el edificio y sus componentes. Luego, abrimos el panel que se encuentra en Windows>Lightning y ajustamos el parámetro ‘ambiental light’ para hacer que se ilumine toda la escena y no solo la luz que entra directamente por la puerta de cristal.

Al hacerlo vemos el cambio en el editor de Unity y para comprobar que funciona en formato web, vamos a Build Settings>Build y lo compilamos.

La duración de esta compilación puede variar entre 10 min o 45 minutos a causa de que Unity debe hacer los cálculos de los lightmap y hacer dicho cocinado. Si se hace algún cambio en la posición de los objetos, esto se debe volver a calcular.

Figura 17: Directional Light en el Proyecto

Una vez compilado, se comprueba con el servidor local. Al abrir la página, el ajuste del 'ambient light' y el cocinado no ha hecho ningún cambio en la escena compilada a web. Por tanto, se ha comprobado que esta opción del "baked" no funciona para la versión WebGL 2.

Buscando información en el foro de Untiy, me encuentro que la mayoría de posts de los usuarios definen cómo usar estos objetos para crear el ambiente deseado en la escena y no errores de por qué no funciona el cocinado. Pero encontré un post [10] donde se explica qué luces usar dependiendo de qué versión de WebGL se tenga. En este caso, Unity 2019.2 ya usa el formato WebGL 2 mientras que en versiones de Unity más antiguas se usaba WebGL 1. La diferencia básica entre estas es que la primera versión sí que aceptaba hacer un cocinado de las luces mientras que la segunda no. Entonces para dotar de iluminación en WebGL 2 se debe usar el realtime lightning con combinación de point lights.

4.2.1.4. Como iluminar la escena

La mejor forma de proceder, será haciendo uso de diferentes point light puestas dentro del edificio. Al hacer esto, en el techo se queda un punto más blanquecino en el lugar que se ha puesto dicho GameObject de point light. Con el fin de disimular la posición de este objeto, se va a intentar poner un point light por bombilla que haya en el edificio, es decir a las area light que se han mencionado anteriormente. Para poder tener una buena visualización de toda la planta y saber dónde poner los point light se va a cambiar la vista de la escena para verla desde abajo, tal y como se ve en la siguiente imagen, Figura 18. La intensidad de cada de point light y su rango de luz es diferente en cada uno de ellos ya que depende del tamaño de la zona a iluminar. Además, hay que tener en cuenta la proximidad de las paredes ya que no se puede poner mucha intensidad porque se reflejaría un punto blanco en dichas paredes. También es importante saber que las superficies con una textura rojiza reflejan más la luz, por ello, los point light que hay cerca de una pared roja son de menos rango e intensidad. Una vez terminada, se puede apreciar el resultado en las figuras 19 y 20. De forma similar se ha trabajado para realizar el posicionamiento de los point lights en la primera planta.

Figura 18: Vista edificio desde abajo

Figura 19: Imagen Planta baja entrada

Figura 20: Imagen Planta baja fondo

Por otra parte, es muy importante que estos objetos de luz estén en modo realtime. En caso contrario, Unity intentará hacer una renderización de las luces y como WebGL 2 no lo soporta aparecen unos cuadrados oscuros por todo el edificio. Esto son los lightmaps que no se han renderizado correctamente, como se puede observar en la imagen siguiente (Figura 21):

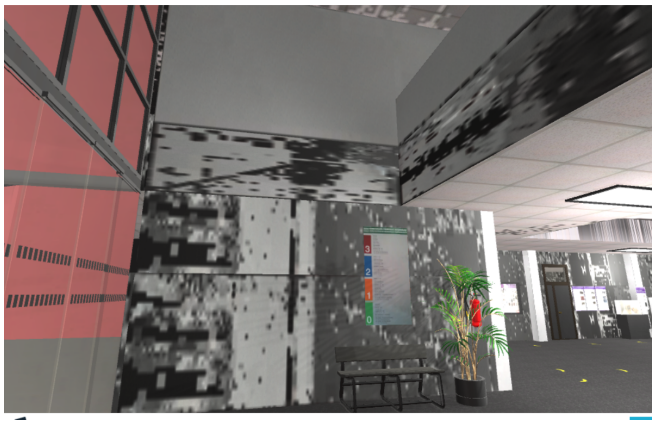

Figura 21: Cuadros negros en proyecto

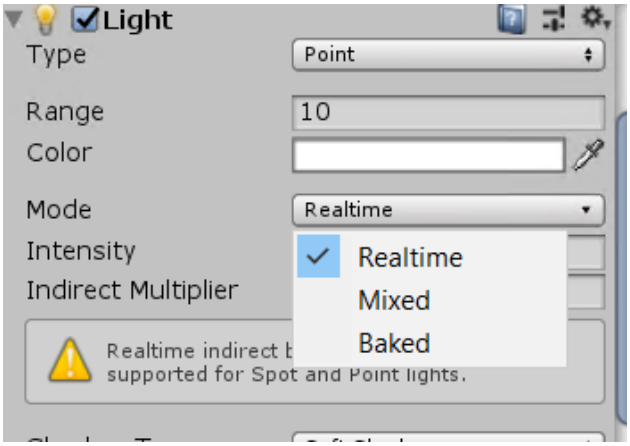

Figura 22: Selección Mode Realtime

Para evitar este efecto, se debe seleccionar el objeto point light y desde Hierarchy se indica el Mode como Realtime, como se indica en la imagen 22.

Seguidamente, para iluminar la primera planta se procede de una forma similar a como se ha trabajado en la planta baja.

Por último, se han puesto luces de point ligth en zonas que no son accesibles por el usuario pero que a lo lejos si son visibles y por tanto si se dejaran en oscuridad, darían la impresión de un edificio fantasma lo cual se aleja de nuestro propósito que es acercarnos lo máxima a una iluminación natural.

Al finalizar la parte de la iluminación, se ha actualizado el proyecto con los expositores y objetos nuevos que se han añadido al museo desde el año 2019. Esto se explicará con más detalle en los puntos siguientes, pero es importante comentarlo aquí ya que al añadir un objeto nuevo ya sea un Game Object para hacer la figura del expositor o una fotografía de un cartel parecía que tenían una mayor luz que el resto de objetos que ya existían previamente dentro del proyecto.

A causa de esta diferencia, mirando en las propiedades dentro del Hierarchy entre un objeto nuevo y otro ya existente, se ha visto que los objetos añadidos no tenían la propiedad Contribute Global Illumination activada, como se observa en la Figura 23.

La iluminación general se refiere a la luz que va saltando de un lugar a otro, es decir, si un rayo de luz le da a una pared, esta rebota en los objetos que tiene alrededor con menor intensidad. En el caso de que esta propiedad esté activada, al hacer el cálculo de la luz indirecta se tendrá en cuenta dicho objeto, en caso contrario se ignorará al objeto por completo. Para el proyecto web no podemos hacer el cocinado de las luces y por tanto no hay mapas de luz ni luces indirectas a tener en cuenta. Entonces se puede desactivar esta opción y así dará una sensación de iluminación completa del proyecto web. Esta propiedad se debe desactivar de todos los objetos que se ven en la escena, desde los expositores hasta las mesas, sillas, puertas y estructura general del edificio.

Una vez realizados los ajustes descritos anteriormente, las estancias del museo quedarían iluminadas tal como se ve en la imagen 24. Esta es una visualización muy parecida a la que se tiene con el proyecto original hecho para las gafas Oculus.

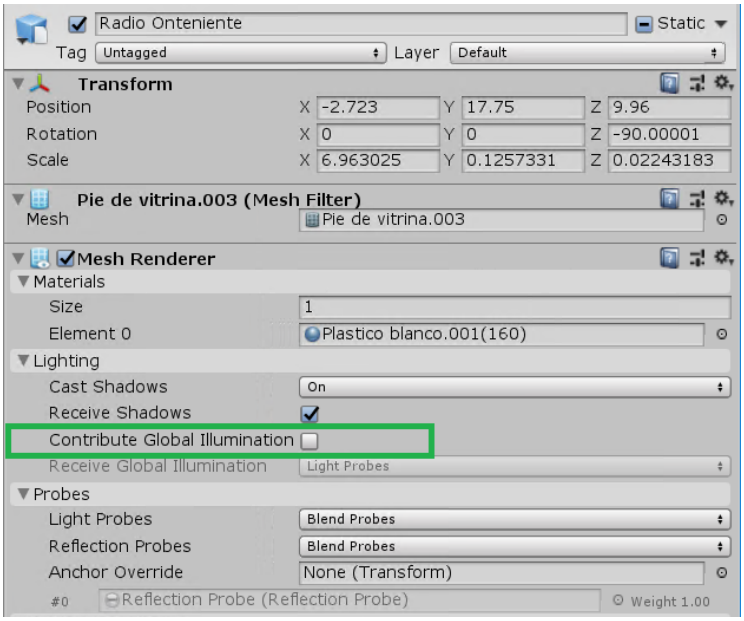

Figura 23: Global Illumination en Hierarchy

Figura 24: Resultado al cambiar la opción Global Illumination

Al hacer este cambio, no haría falta la presencia de la mayoría de los point light que se han puesto en un primer momento. Algunos objetos de este tipo se han eliminado mientras que otros se han recolocado y cambiado su intensidad para iluminar alguna de las zonas que se podían quedar un poco más oscuras, como se ha hecho en los expositores de la primera planta donde la radio de Onteniente y los expositores de alrededor se quedaban ensombrecidos, Figura 25.

Figura 25: Point Lights finales

4.2.2. Integración en la página web del museo

Una vez conseguida una iluminación óptima y hechas pruebas con un servidor local para ver que funciona en el navegador, se va a proceder a subirlo a la página web del museo.

La página web del museo Vicente Miralles Segarra está hecha con un sistema de gestión y creación de contenidos (CMS) web llamado Wordpress. Este sistema está diseñado con el lenguaje PHP para entornos que ejecuten MySQL y Apache y es un software libre. Sus creadores Matt Mullenwegg y Mike Little lanzaron en el año 2003 su primera versión con el fin de dar la oportunidad a cualquier persona de crear su espacio web de una forma personalizada y con una buena arquitectura. Una de las cualidades más destacables de Wordpress es su constante actualización, la última versión es la 5.7. Con el cambio y la intención de renovación constante, Wordpress se ha situado como uno de los sistemas más populares para el mantenimiento y desarrollo de páginas web entre bloggers y otras entidades. Actualmente, la página web del museo está actualizada con la versión 5.7, esto nos da la ventaja en cuanto a poder utilizar las últimas novedades como son la mejora en la usabilidad.

Otra característica de esta plataforma es el uso de plugins. Los plugins son aplicaciones hechas con software que contiene funcionalidades que se pueden añadir a una página Wordpress sin la necesidad de cambiar el código PHP original de la página. Es importante mantener los plugins que estén funcionando con la versión de Wordpress que se tiene actualmente, en caso contrario, no se podrían utilizar.

4.2.2.1. Compilación en Unity

El primer paso para subir la visita virtual es compilar el proyecto del museo en formato WebGL. Para ello, abrimos el proyecto con Unity y en la barra de menú vamos a Files>Build Settings, y nos saldrá una ventana como vemos en la Figura 26.

Seguidamente, indicamos la plataforma a la cual deseamos pasarlo, en nuestro caso es WebGL. Antes de compilar, el programa nos da la opción de cambiar algunas características del proyecto, para ello vamos a la parte inferior izquierda y entramos en Player Settings

En la ventana siguiente nos aparecerá un menú lateral y vamos al apartado Player. Dentro de este, vamos a cambiar la configuración del proyecto que tiene que salir. Las opciones que se pueden cambiar son las dimensiones del proyecto resultante, la Splash Image (la cual en nuestro caso no se puede quitar ya que estamos usando la versión gratuita de Unity), el tipo de codificación, entre

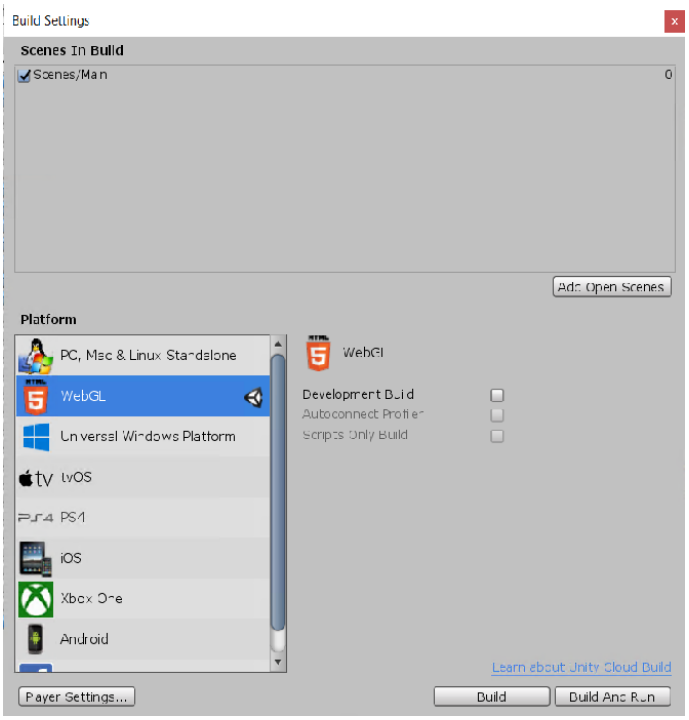

Figura 26: Ventana BuildSettings

muchas otras. En este proyecto se han cambiado dos cosas, las dimensiones y el "template" del documento html para hacer que no aparezca un botón de pantalla completa ya que al integrarse en la página web de Wordpress este no funciona.

Dentro de Player Settings se puede cambiar el tamaño final del juego, el formato de compresión entre otros. En un primer momento, vamos a dejar todas las opciones igual, sin cambiar nada.

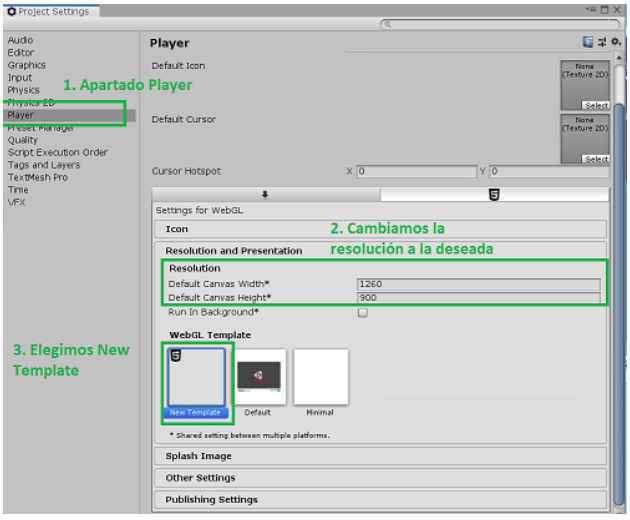

Figura 27: Propiedades que se han editado en PlayerSettings

Las dimensiones puestas en el apartado Width y Height son 1260 y 900 correspondientemente, se

ha elegido un tamaño superior al que había originalmente para que la ventana donde se vea la visita virtual tenga una mejor visibilidad.

Otro parámetro que se ha cambiado es el usar un Template diferente al que viene por defecto para el archivo html resultante. La diferencia entre el Template nuevo que se ha creado y el que está hecho por Unity es la eliminación de una línea donde se especifica un botón de pantalla completa.

Para crear este nuevo template, dentro de la carpeta de Assets se crea una nueva carpeta llamada WebGLTemplates y dentro de esta, otra llamada NewTemplates. Finalmente, se crea un archivo html. En este caso particular, se ha creado el nuevo template html con la misma estructura que el original a excepción de la línea del botón de pantalla completa.

Cuando ya se han cambiado todas las opciones de configuración deseadas, se cierra la ventana que se ve en la Figura 26, se pulsa el botón Build con la casilla de Development Build desmarcada y se indica una carpeta donde guardarlo. Una vez haya terminado, el resultado de la compilación estará compuesto por una carpeta llamada Build, donde está la información del proyecto, una carpeta llamada Templates y un archivo html. Todo esto está explicado en la Figura 27.

4.2.2.2. Publicación en Github

Con el trabajo compilado, el objetivo es ponerlo en la página web. Hay varias formas de hacerlo entre ellas están usando un plugin de Wordpress o publicándolo en una página web y luego insertándolo en la web del museo. En un primer momento, se intentó usar un plugin de Wordpress, en su página oficial. Hay poca existencia de plugins que sirvan para subir proyectos WebGL de Unity. Aun así, se ha encontrado un plugin llamado *Howescpae Unity3d WebGL*. En la descripción de este se explica cómo subir el proyecto a través del plugin y luego cómo ponerlo en la página o post deseado usando un lenguaje denominado short text, esto es una forma de referencia de código dentro de Wordpress.

Este plugin lleva ya instalado ejemplos de juegos WebGL. Al probarlos dentro de la página del museo, o no sale nada directamente o sale un error de referencia. Además, se han hecho varios intentos de publicar la visita virtual compilada con diferentes versiones de Unity, pero sin ningún resultado.

Es por ello que, finalmente, se ha decidido proceder a publicar el proyecto en una página secundaria para luego insertarlo en la web del museo. Dentro de la comunidad de usuarios de Unity que han publicado su proyecto en una página web, la forma más utilizada ha sido utilizar páginas como son simmer.io o itch.io en las cuales el programador se registra y sube al servidor su juego. Seguidamente, con el lenguaje html se usa la etiqueta <iframe>para insertarlo en la página deseada. Este formato es muy común por su comodidad, pero si se usan páginas como estas, mientras se carga el juego saldrá el logo de la página donde está publicado originalmente. Para intentar evitar que salga este logo, he buscado otra manera de poder publicarlo y he encontrado la página de Github. Github es un repositorio online de gestión y desarrollo de proyectos basado en la nube, que te permite hacer un control de versiones a través de Git. Esta plataforma es una de las más usadas por los desarrolladores y por grandes empresas para compartir su trabajo. Su función colaborativa facilita a otras personas el trabajar en ellos.

Para subir el proyecto a Github en primer lugar, se ha creado una cuenta con el correo electrónico del museo y seguidamente se ha creado un nuevo repositorio. Para hacerlo, entramos en el perfil de la cuenta creada en esta plataforma y en la parte superior derecha pulsamos el icono del perfil

y seleccionamos la opción “New Repository”, al hacerlo saldrá una ventana como en la siguiente imagen.

Como nombre de ’Owner’ debe aparecer el nombre del usuario de la cuenta de Github y en ’Repository Name’ se debe escribir el nombre del repositorio que se quiere crear. Luego, se indica si será público o privado y seleccionamos la opción “Add a read me file”. Por último, se pulsa el botón verde que se encuentra al final de la pantalla para su creación. (Figura 28).

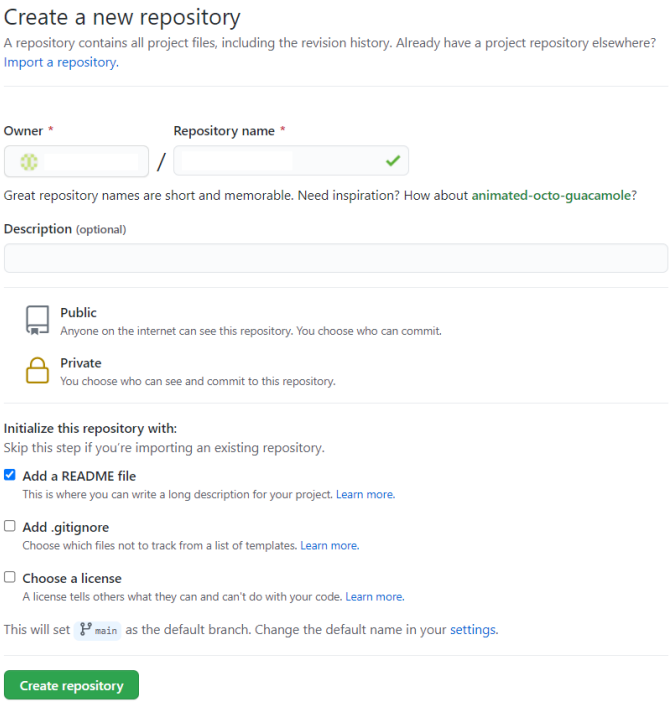

Figura 28: Ejemplo como crear un repositorio en Github

Una vez creado el repositorio, el siguiente paso es subir los archivos a este. Como el proyecto es de un tamaño de 68 Mb, para subir el proyecto compilado a web se debe clonar el repositorio en nuestro ordenador, esto se llamará nuestro repositorio local. Entramos en el repositorio y copiamos en portapapeles el enlace para clonarlo, como se ve en la Figura 29.

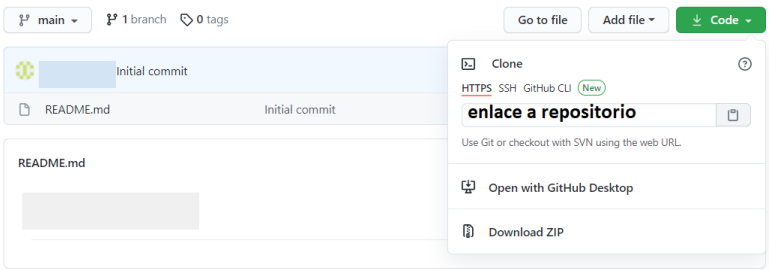

Figura 29: Copiar url repositorio

Luego, para hacer la clonación, el ordenador debe tener instalado GitBash o escribir los comandos

desde un terminal dentro del programa Visual Studio Code, donde habrá que instalar la extensión Git y sincronizar una cuenta. En ambos casos, se tiene que crear una carpeta en el escritorio y se abre el terminal. En caso de usar GitBash, será botón derecho>New GitBash y en caso de usar VSC será abrir un terminal en la carpeta abierta. Finalmente, se clona el repositorio con el enlace que está en el portapapeles con la instrucción: git clone url.

Donde url es el enlace copiado en el portapapeles. Al hacer esto, en la carpeta local del ordenador se hará una copia de los archivos que haya en el repositorio. Para añadir la compilación web del proyecto se copian los archivos dentro de esta clonación local del repositorio. Los archivos que se deben añadir son la carpeta Build, Template y el archivo index.html resultados de la compilación anterior.

Ahora tenemos que subirlo desde nuestro repositorio local al de la nube, para ello dentro del terminal usamos las instrucciones de Github para actualizar primero el repositorio local y luego subir los cambios. Las instrucciones usadas son:

```
git add .
git commit -m 'comentario'
git push
```

Una vez hecho esto, actualizamos la página de Github donde se encuentra el repositorio creado y se verán los archivos subidos.

Para publicarlo en esta página, navegamos a Settings>Pages (se encuentra en el menú lateral). Dentro de Pages, se indica la rama main y la root del repositorio. Es importante indicar el tema de la página que tendrá la publicación de la visita virtual.Se puede observar los pasos a seguir en la siguiente Figura 30 y 31.

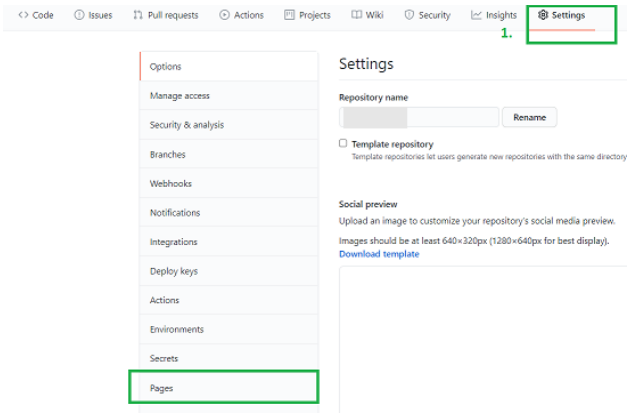

Figura 30: Navegar a la página Pages

Una vez hecho esto, en la parte superior debe aparecer un enlace, el cual estará compuesto por el nombre del usuario de la cuenta de git y el nombre del repositorio. Si lo abrimos veremos la visita virtual publicada dentro de Github.

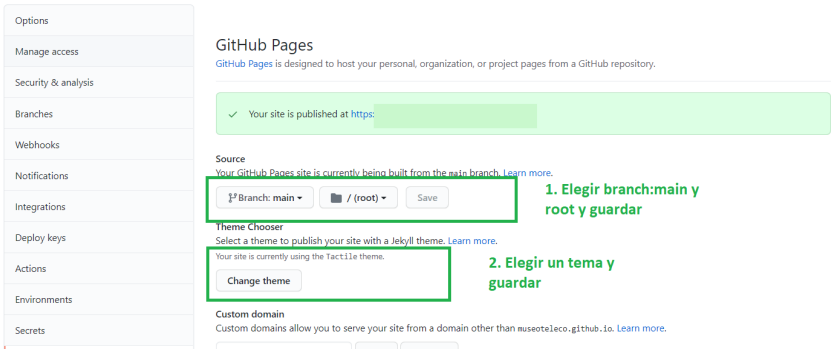

Figura 31: Crear página Github

4.2.2.3. Página WordPress

Como ya se ha indicado, la página del museo de Vicente Miralles Segarra está hecha con Wordpress, una herramienta de gestión de contenido para crear, personalizar y mantener páginas web. En este apartado se va a explicar cómo se ha creado la página dentro de la web del museo y cómo se ha puesto la visita virtual para que cualquier usuario pueda disfrutar del museo en cualquier momento.

Previamente a subir la visita, la tutora me ha dado permisos de administradora dentro de la cuenta de Wordpress y además la tutora me ha indicado que se debe poner la página de la visita virtual dentro de la pestaña Visitas.

Figura 32: Menú inicial página web

Para crear esta página de Visita Virtual hay que entrar en la página de edición de Wordpress y poner el usuario y contraseña facilitados previamente. Aquí dentro saldrá un menú lateral donde están las variadas opciones que se pueden hacer en esta herramienta de gestión de contenido. Como se puede ver en la Figura 33, en este apartado, se crea la página de Visita Virtual y se guarda en la carpeta correspondiente.

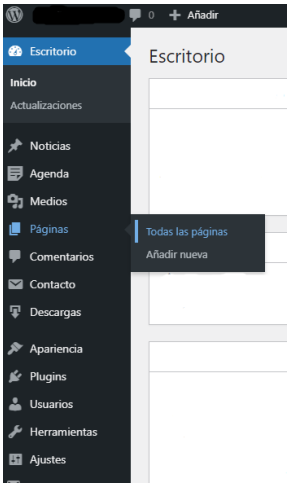

Figura 33: Crear página en Wordpess

Ahora ya existe la página, pero esta no es visible aún en el menú inicial de la web. Para hacerlo accesible al público hay que ir al apartado de Apariencia>Menu, el cual también esta en el menú lateral. En este apartado aparecerá una pestaña con la lista de las páginas existentes y a la derecha otra pestaña que conforma el menú inicial. Se selecciona la página que se quiere añadir desde la lista y esta aparecerá al final de la pestaña del menú. Finalmente se arrastra la página dentro del apartado del menú deseado, en este caso, se ha puesto dentro de Visitas>Visita Virtual.

En el apartado anterior se ha publicado la visita de realidad virtual del museo en Github, pero el objetivo final es tener esta visita dentro de la página web oficial del museo. Para hacerlo se va insertar la visita virtual dentro de la web utilizando código html y JavaScript.

Si se quiere editar una página hay que volver al apartado de Páginas del menú lateral (Figura 33) y pulsar la opción “editar” que sale al lado del nombre de la página donde se quieren hacer los cambios. Una vez hecho esto, debe salir una nueva ventana donde se pueda escribir texto, añadir contenido entre otras cosas. Si se quiere añadir código html, se debe cambiar la vista del editor de Editor Visual a Editor Código, y luego dentro de las etiquetas <p>se añade el html. Los pasos los vemos explicados en la Figura 34

El código html añadido es el siguiente:

Figura 34: Edición de una página

```
 <p><iframe id='embededGame' title="Iframe Museo" src="url_git"
width="1260" height="720"></iframe><br><button onclick="goFullscreen('embededGame');
return false" style="background-color:#6C2DC7; border-color:#6C2DC7;
color:white">Pantalla Completa</button></p>
```

Se usa la etiqueta *iframe*, la cual te permite insertar un trozo de una página existente a la tuya. Tiene atributos como son el título, *id* y *width* y *height* que sirven para definir el tamaño del recuadro que se va a incrustar dentro de nuestra web. Cabe destacar que el atributo *src* sirve para definir la url de la página que se pretende insertar. En este caso es la url que se ha obtenido cuando se ha publicado la visita en Github.

En este código html también se ha definido un botón para poder poner la pantalla completa y disfrutar de una experiencia con mejor visualización y movilidad. Para hacer que este botón funcione, se ha hecho uso de código JavaScript complementario. Wordpress no entiende el código JavaScript si este está puesto de la misma forma que se ha puesto el código html anterior, es decir, no se puede poner dentro de las etiquetas <p>y esperar que este CMS lo entienda. Para ello, una de las formas de escribir JS es haciendo uso de un plugin.

Para poder añadir JS se ha utilizado el plugin llamado *Header and Footer Script*. Para instalar este plugin desde el menú lateral de la página de wordpress (Figura 33) se pulsa en Plugins>Añadir Nuevo Plugin y se busca por nombre el que se desea instalar. Una vez hecho esto, hay que activarlo y ya se puede usar.

Cuando se activa este plugin, durante la edición de cualquier página saldrá una caja en la parte inferior del editor. En esta caja se puede escribir tanto código JS como html, lo que hará el plugin es insertar el contenido de la caja en el html de esa página. Deberá quedar como se ve en la Figura 35.

Figura 35: Código JS insertado con el plugin

El código JS debe ir dentro de la etiqueta <script></script>y está compuesto por:

```
<script>
function goFullscreen(id) {
    var element = document.getElementById(id);
    if (element.mozRequestFullScreen) {
      element.mozRequestFullScreen();
    } else if (element.webkitRequestFullScreen) {
      element.webkitRequestFullScreen();
    }
}
```

```
</script>
```

Este lo que hace es que si se hace click en el botón se llama a la función de FullScreen, la cual pone en modo pantalla completa el elemento con el *id*, en este caso es el elemento del *iframe*.

4.3. Proyecto Oculus

El proyecto original sobre el que se ha trabajado para hacer la versión web de la visita virtual estaba originalmente hecho para usarlo con unas gafas de realidad virtual de marca Oculus, en concreto con las Oculus Rift. En este apartado se va a explicar qué se ha hecho para adaptar el mismo proyecto para dos gafas nuevas, las Oculus Rift S y las Oculus Quest, además de los problemas que han ido surgiendo y como se han solucionado.

4.3.1. Estado del proyecto

Unity es una herramienta que se está actualizando constantemente, es decir, cada año sale una versión nueva y mejorada para ampliar los recursos y formas de trabajar de los usuarios. El proyecto sobre el que se va a trabajar está hecho con la versión 2018, se ha decidido pasar a utilizar una versión más reciente, en concreto la 2020.3 para evitar problemas de no compatibilidad con librerías que Unity puede marcar como en desuso.

Al migrar a una versión más reciente, pueden aparecer errores de algunas clases que han cambiado de nombre o que ya no están reconocidas por Unity, ocurrió algo similar cuando se hizo el proyecto web y se ha explicado en el punto anterior como resolverlo.

Sin embargo, una vez resueltos al intentar compilar el proyecto aparecen en la consola otros errores distintos. Estos están relacionados con las librerías de Oculus, en concreto con archivos del OVRplayer el cual define el comportamiento y adaptación de un juego para su uso con las gafas Oculus de realidad virtual.

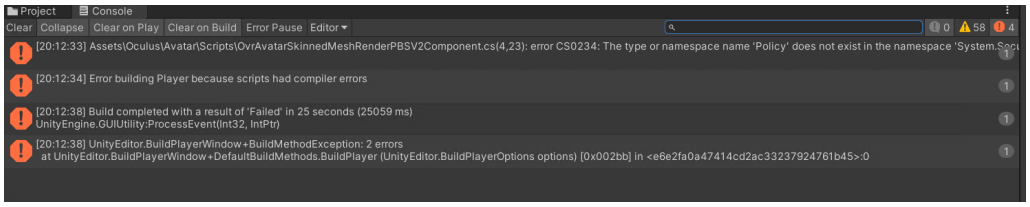

Figura 36: Errores consola proyecto Oculus

Consultando los foros de Unity, su documentación y la documentación de las librerías de Oculus no se ha encontrado ninguna referencia a este problema o algún caso similar. Estos errores no eran muy comunes, ya que se ha trabajado desde la versión funcional del trabajo anterior y no se ha editado nada. Se ha comprobado que estos mismos errores también aparecían usando la versión 2018 de Unity, con la que se ha hecho originalmente el proyecto. Entonces, se ha llegado a la conclusión de que el problema no reside en algún cambio hecho en el código de estos archivos *.cs* sino en que las librerías de Oculus que se han usado son muy antiguas y por tanto Unity no las

reconoce como debería. La solución a este problema ha sido eliminar todas las librerías de Oculus del proyecto y volverlas a instalar. Se ha seguido las instrucciones recomendadas por la página oficial de las gafas de realidad virtual. [11]

Aquí se detalla como desinstalar el OVRplayer y todos sus archivos eliminándolos directamente de la carpeta del trabajo y además eliminando todos los archivos que empiecen por OVR y Oculus que pueden quedar aun instalados en otras carpetas.

A continuación, desde el menú de Unity se entra en Window>Asset Store. La asset store es una biblioteca donde se encuentran elementos para poder extender las características de un juego, hay assets que contienen texturas y materiales para decorar el trabajo, animaciones y ejemplos de tutoriales de cómo crear un proyecto desde cero. La mayoría de assets suelen ser gratuitos pero los que son de pago tienen un precio asequible al mercado. Para instalarlos hay que entrar en la cuenta de Unity y seleccionar el asset deseado, en nuestro caso será el Oculus Integration Package.

Ya se ha descargado el package de las Oculus ahora falta instalarlo en Unity, se va a Window>Package Manager. Aquí saldrá todos los assets que tiene Unity instalados, como hemos descargado las librerías desde el asset store estas estarán en el apartado My Assets, para cambiar a este apartado se pulsa en la pestaña superior.

Aparecerá una lista de los assets descargados de la asset store, pulsamos en Oculus Integration y luego importamos. Una vez termine de instalarse en el editor, nos preguntará que si se quiere actualizar los plugins, a esto se seleccionará la opción de sí.

Una vez hecho esto, en la layout de Hierarchy de Unity debe aparecer un nuevo elemento llamado OvrPlayerController, se selecciona y en Inspector se pueden definir sus propiedades. Se han dejado las mismas propiedades que había en el proyecto anterior a excepción de la velocidad de movimiento que se ha bajado a 32 puntos para disminuir la sensación de mareo que puede experimentar el usuario que lleva las gafas, estas propiedades se pueden ver en la Figura 37.

Cabe destacar que se ha activado la opción de poder rotar con el controlador derecho, esto se hace para poder moverse mejor dentro del entorno virtual.

Una vez puestas las propiedades del Ovr y situado dentro del edificio virtual, se pone la OvrCameraRig para indicar cuál es el recuadro a visualizar dentro de las gafas. Además, se va a hacer uso de un prefab de Oculus. Los prefabs son elementos con funcionalidades y características concretas que ya están creados y se pueden hacer uso en cualquier proyecto. El prefab que se va a usar está dentro de la carpeta de Assets>Oculus>Avatar>Content>Prefabs y se llama LocalAvatar, al añadir este elemento a la escena deberá aparecer un muñeco que representa la cabeza y manos del jugador. En un primer momento, al añadir el prefab al Hierarchy y ponerlo dentro de OvrCameraRig, este no se visualizaba al iniciar el juego desde el editor. Para resolver este problema se ha desinstalado una segunda vez las carpetas relacionadas con Oculus y se han vuelta a instalar siguiendo los pasos explicados anteriormente.

Una vez se ha vuelto a instalar, se han seguido los pasos para poner las propiedades del OvrPlayerController y el OvrCameraRig. Finalmente, se ha puesto el LocalAvatar, con la propiedad de StartWithControllers activada y en una posición Z=1.5 para poder verlo de forma completa. De esta manera sí que se ha podido ver desde el editor de Unity, el avatar se puede ver en la Figura 38.

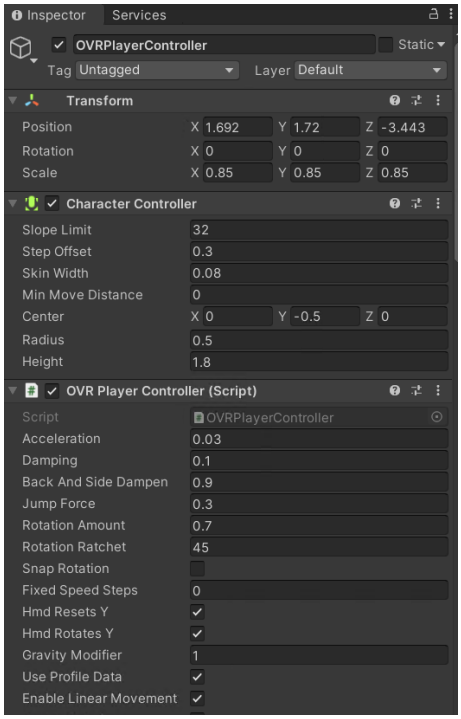

Figura 37: Ventana Inspector

Figura 38: Avatar dentro del entorno de Unity

La característica principal que se hizo en este proyecto fue la integración en el entorno virtual de los vídeos que se han usado en la aplicación de realidad aumentada del museo. En esta aplicación de RA, dada una Image Target, salta un vídeo en la pantalla del dispositivo. Para trasladar el funcionamiento de la app de RA al proyecto de RV, se ha hecho que el avatar controlado por el usuario pulse una imagen de la Image Target (en el proyecto a este GameObject se le ha llamado código QR) y salga delante de la vista del usuario un recuadro donde se inicia el vídeo.

Esto ocurre ya que existe una colisión entre la mano del avatar y el objeto que representa la Image Target. En el trabajo anterior se editaron algunos ficheros *.cs* originales de la integración de la librería Oculus para redefinir la mano no solamente como un elemento entero, sino que estuviera compuesta por los cinco dedos. Así pues, al dedo índice se le marcó con el nombre de index3 para que solamente ocurriera una colisión entre este dedo y el código QR.

Además, se creó un script nuevo donde se ha definido que hacer cuando ocurre la colisión, en primer lugar, se hace parar la reproducción de un vídeo o audio que pueda haber en curso y también la música ambiental. Seguidamente, según qué QR se haya seleccionado con el dedo índice se reproducirá el vídeo o audio que proceda.

Originalmente, en este proyecto se hicieron cambios en los archivos *.cs* de las librerías de Oculus integradas. La solución que estaba hecha tenía muy buen resultado ya que se consiguió hacer que saltara la colisión deseada, pero tiene un inconveniente, no es escalable. Es decir, la integración de Oculus necesita ser actualizada cada cierto tiempo para poder seguir usándose y si se cambian los scripts originales, en cada actualización se deben volver a realizar los cambios, en caso contrario se tendrá un proyecto que no será usable cada cierto tiempo.

Se ha buscado una solución que no conllevara la edición de los scripts de la librería de Oculus, con la ayuda de la cotutora se decidió hacer uso de una propiedad de Unity, los tags. Estos son etiquetas que puedes poner a un elemento y desde un script controlar qué le puede ocurrir a dicho elemento que contiene un tag en concreto.

Se ha creado un tag llamado index3, para hacer una prueba inicial, desde el script (QR_Video y QR_Audio) que controla el lanzamiento de los videos y audios se ha puesto el código que se puede observar en la Figura 39.

```
void OnTriggerEnter(Collider other)
{
    Debug.Log("ha entrado en ontrigger"+ other.gameObject.name);  1
    if (other.gameObject.CompareTag("index3"))  2
    {
        Debug.Log("ha entrado en el if del QR_video con el objeto " + other.gameObject.name);  3
        if (!videoPlayer.isPlaying)
        {
            StopAllVideo();
            StopAllAudio();

            Transform baseT = Instantiate(prefabBase);
            baseT.transform.parent = transform;
            baseT.position = spawnPointBase.position;
            rotationBaseTx = transform.localEulerAngles.x;
            rotationBaseTy = transform.localEulerAngles.y;
            rotationBaseTz = transform.localEulerAngles.z;
            baseT.Rotate(rotationBaseTx, rotationBaseTy + 90, rotationBaseTz);
```

Figura 39: Script de QR_Audio y QR_Video

En el número uno se indica la línea de código que hará que salga un mensaje en la consola en caso se haya entrado en la clase de *OnTriggerEnter*, esto ocurre si se ha producido una colisión, además también se saca por la consola el nombre del objeto que ha colisionado. El número 2 indica el condicional *if* el cual solo dejará ejecutar el código de dentro si el objeto que ha colisionado lleva la etiqueta index3. De esta forma evitamos que se ejecute el código si la colisión no es con el avatar (que tiene el tag). Y en el número 3 se ha usado la instrucción *debug* para saber si cumple la condición if. Para poder simplificar la prueba, se ha hecho inicialmente con un Game Object al cual se le ha puesto esta etiqueta. Al colisionar el Game Object con un QR sí que se producía la colisión y se empezaba a reproducir el vídeo.

En cuanto se comprobó que estas pruebas sencillas funcionaban se pasó a probar el objeto tag con el prefab del LocalAvatar. Para ello fue necesario hacer las pruebas con el PC del museo y las gafas

Oculus conectadas a este.

Con las gafas Rift S conectadas al ordenador y el proyecto en funcionamiento, se ha puesto el tag index3 al objeto LocalAvatar, se inicia la visita desde el modo Play del editor de Unity y se hacen las pruebas. Al hacerlo esto no funcionaba, es decir, aunque el avatar tenía la etiqueta puesta, esta no hacía que se produjera la colisión. Para resolverlo se ha puesto un componente nuevo al LocalAvatar, un *box collider*, esto es como una caja invisible que rodea al avatar y con el cual se puede hacer un seguimiento de cuando se acerca a otro objeto. Es importante señalar que tanto el avatar como el objeto del código QR deben tener este componente para hacer que salte la colisión. Es por ello que el código QR también tiene puesto el *collider* pero en dimensiones más pequeñas.

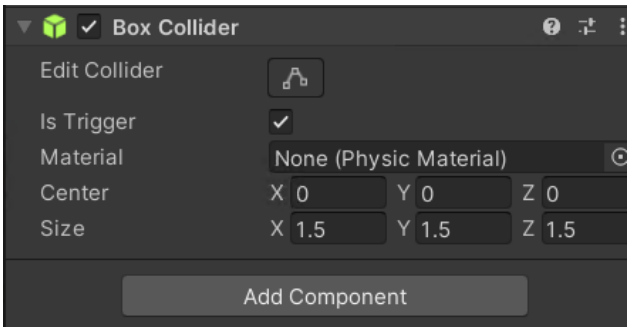

Figura 40: Componente box collider

Con el *box collider* y el tag puesto en el local avatar, para hacer saltar la colisión y que aparezca el evento solamente hay que acercarse al código QR, no hace falta tocarlo con las manos virtuales ni con los controladores.

4.3.2. Oculus Rift y Rift S

El Museo tiene a su disposición dos ordenadores para poder trabajar. En el más antiguo se han realizado los trabajos anteriores y se han usado las Oculus Rift ya que tienen un conector que solo se puede utilizar en este PC. Cabe señalar que el ordenador con el cual se usa este modelo debe tener una memoria mínima de 8GB+ RAM, un procesador Intel i5-4590 / AMD Ryzen 5 1500X y una tarjeta gráfica NVIDIA GTX 1060 / AMD Radeon RX 480 o mayor, sistema operativo Windows 10, tres puertos USB 3.0, un puerto USB 2.0 y HDMI 1.3 para la salida del vídeo.

Las Rift están compuestas por el casco con las gafas, dos controladores joystick y además dos sensores de movimientos los cuales se tienen que situar a una distancia mínima de 1m entre el ordenador. La primera vez que se usan se debe definir el sistema guardián y recoger la zona donde se va a jugar para evitar tropiezos o caídas del usuario.

El segundo ordenador es el más nuevo y contiene una tarjeta gráfica mejor. En concreto, para usar las Rift S, el ordenador debe tener una memoria 8GB+ RAM, sistema operativo Windows 10, un puerto USB 3.0, output de video DisplayPort, una tarjeta gráfica NVIDIA GTX 1060 / AMD Radeon RX 480 y un procesador Intel i5-4590 / AMD Ryzen 5 1500X. Estas gafas estan compuestas solamente por el headset y dos joystick.

Se ha usado el mismo proyecto para probarlo con los dos tipos de gafas anteriores. En primer lugar, se hicieron pruebas con las Rift S y cuando se tuvo en funcionamiento todo, se pasó a usar las Rift.

4.3.3. Oculus Quest

Las gafas Oculus Quest son un modelo de las versiones más nuevas, está pensado para usarlo sin la necesidad de tener que estar conectado con un cable a un PC para disfrutar de la realidad virtual. Esto hace que a la hora de probar nuestra visita se tenga que proceder de manera distinta. Al igual que las Rift S, este modelo está compuesto por el casco y dos joystick pero el casco no tiene un cable que vaya conectado directamente al ordenador.

Para probarlo, se debe compilar la visita en formato .apk, es decir como si se fuera a instalar a un dispositivo móvil. Y luego se debe instalar en el headset de las gafas. Se ha usado el mismo proyecto que en las anteriores gafas pero optimizado, es decir se ha bajado la calidad de algunas fotografías y algunos vídeos para que al compilarlo este no pesara tanto.

4.4. Actualización cambios en el museo

Desde el año 2019 la entidad del museo ha obtenido nuevos objetos en su colección y se han hecho cambios en la disposición de los expositores. Por ende, se ha tenido que trasladar dichos cambios a las versiones virtuales del museo, es decir, al proyecto web y al proyecto de las gafas Oculus.

4.4.1. Medidas y fotografías

Para poder replicar los expositores en la versión virtual del museo tanto en el proyecto Web como en el de las gafas Oculus, se ha tenido que identificar los objetos nuevos y también la dimensión de los expositores y donde están posicionados. Para poder hacerlo, se quedó un día con la tutora Carmen Bachiller en el edificio 4D. Durante esta reunión se tomaron medidas de los expositores, se midió la altura, ancho y profundidad para poder replicarlos con mayor exactitud. Además, este mismo día se tomaron fotografías de los objetos para luego poder incluirlas en la versión virtual.

Las medidas de los expositores añadidos son:

- Expositor tocadiscos: 172x156,5x56,5 cm
- Expositores de cámaras de grabación: 145x66,5x66,5 cm
- Expositor TVs. Base primera: 120cm diámetro y 41 cm de altura. Segunda base: 100 cm diámetro y 10 cm de altura. Tercera base: 80 cm diámetro y 10 cm altura.
- Expositor centralita costurera: 173x82x92 cm
- Expositor Vicente Miralles: 141x227x46,5 cm

4.4.2. Importación a Unity

En el entorno de Unity, se ha decidido seguir el método de trabajo que se ha hecho hasta ahora cuando se ha tenido que poner un objeto nuevo del museo. Se pueden crear desde cero o duplicar alguno ya existente.

En primer lugar, se han modelado las vitrinas donde se exponen los objetos. Estas están formadas por una base con un material grisáceo y la parte superior está hecha con un material transparente para crear la sensación de un cristal.

Con el fin de dar una sensación de continuidad al proyecto, se ha seguido la forma de trabajo hecha anteriormente. Con esto, se ha conseguido hacer cinco vitrinas nuevas, cada cual se ha modelado al tamaño adecuado. Las vitrinas hechas son las siguientes:

- **Vitrina y cartel de Vicente Miralles.** Expositor donde se recogen objetos personales de Vicente Miralles. Además, en este también hay un cartel informativo donde se explica brevemente su historia, además de una fotografía suya y sus objetos personales. (Figura 41).

Figura 41: Expositor Vicente Miralles

- **Vitrina Centralita costurera.** Situada enfrente de las escaleras de la entrada principal al edificio. (Figura 42).
- **Vitrinas de las grabadoras de vídeo.**Para añadir estas dos nuevas vitrinas, ha habido un reposicionamiento del expositor de la televisión y se han añadido en su lugar original dos nuevos que contienen aparatos de grabación de vídeo. (Figura 43).
- **Vitrina tocadiscos vinilo.** Expositor de la parte de imagen y sonido donde se exhibe un reproductor de tocadiscos con sus altavoces a conjunto. (Figura 44).
- **Expositor de televisores en forma de "tarta".** Al lado de la radio Onteniente se ha puesto un expositor muy singular. Tiene forma circular y contiene diferentes modelos de televisores que se han ido usando a través de los años. Hay desde los televisores grandes antiguos que había en las casas hace ya más de 60 años hasta algunos más pequeños y portables. La forma del expositor tiene un gran parecido a una tarta con tres pisos de altura, es por ello que se ha nombrado como "expositor tarta". Para modelarlo, se han utilizado seis Game Object en forma de cilindros, tres para hacer la base donde se posicionarán los televisores y los tres restantes para hacer los pilares que sujetan las bases. Una vez hecho el modelo del expositor

Figura 42: Expositor Centralita Manual

"tarta", se ha procedido a ajustar su tamaño dentro del pasillo de la primera planta y se ha añadido las texturas que se han considerado lo más cercanas a los colores del expositor real. Finalmente, se ha añadido los televisores uno a uno. Estos están compuestos por un Game Object de un cubo y un plano donde se sitúa la imagen del aparato. Este proceso se ha repetido 14 veces para añadir todos los televisores. El resultado final se ve en la Figura 45.

Asimismo, en la planta baja se ha añadido un mural donde se ha representado una cronología de mujeres que realizaron diversos avances científicos.

Con la finalidad de añadirlo dentro de la realidad virtual se ha hecho uso de un Game Object en forma de plano, al cual se le ha asociado la fotografía del mural. Seguidamente, se ha puesto sobre la pared donde está el mural, se ha intentado juntarlo lo máximo posible para que pareciese que está pintado sobre esta. Es necesario resaltar que se ha puesto un point light enfrente del cartel para evitar un ensombrecimiento.

Figura 43: Expositor Grabadoras de vídeo

Figura 44: Expositor Tocadiscus Picu

Figura 45: Expositor Televisores

Figura 46: Cronología Mujeres

Capítulo 5

Aplicación de Realidad Aumentada

La realidad aumentada (RA) se puede definir como la mejora del entorno en el que vivimos a través de la superposición de elementos digitales como pueden ser imágenes y vídeos en el mundo real. Este término se puede confundir con la realidad virtual la cual sumerge al usuario en un entorno gráfico creado por herramientas informáticas. Es importante diferenciarlo de la RA ya que esta segunda añade elementos a un lugar, objeto o persona ya existente. Para poder apreciar la realidad aumentada, se necesita de un dispositivo con una cámara que retransmita a tiempo real y además de un software que permita el reconocimiento del entorno para poder mostrar los elementos virtuales a través de la pantalla del dispositivo.[12]

Cuando se habla de realidad aumentad no se puede obviar el término de realidad virtual ya que la historia de ambos va de la mano. En 1962 el cinematógrafo Morton Heiling inventó un aparato llamado Sensorama, el cual incluía una pantalla de color, ventiladores, sonido estéreo, motores emitentes de olores y un motor de movimiento para crear una nueva experiencia. Aunque la idea general de este prototipo se acercaba más a lo que es la definición de realidad virtual, se considera el primer paso hacia la RA.

Figura 47: Sensorama

Más tarde, en 1968 el ingeniero americano Ivan SHuterland diseño un software llamado Sketchpad

Figura 48: Gafas de Democles

de modelaje y estimulación visual en 3D. Este diseño cobró vida a través de unas gafas de realidad aumentada, las cuales tenían una instalación tan pesada que se ganaron el nombre de las "gafas de Democles".

Con el tiempo se han hecho más investigaciones sobre la realidad aumentada y cómo introducirla en nuestro día a día pero no ha sido hasta la década del 2010 cuando esta nueva tecnología ha sido lanzada al público. En 2015, la red social de Snapchat, famosa por el envío de mensajes, vídeos y fotografías de corta duración hizo el debut de la realidad aumentada a través de los famosos filtros. Con un software que reconoce la cara de una persona, se pueden añadir elementos visuales tales como orejas de perro y ver el resultado desde un dispositivo móvil. Más tarde, en 2016 la compañía de desarrollo de software americana, Niantic, lanzó el juego Pokemon Go. En este el jugador puede convertir el mundo que le rodea con la aparición de un Pikachu o Bulbassur delante de sus ojos. [13]

5.1. Estado original de la aplicación

La aplicación del museo Vicente Miralles se ha hecho a partir de dos trabajos finales de grado anteriores. [14] [15]

El objetivo de este trabajo es mejorar la usabilidad de la aplicación al añadir algunos botones o elementos que pueden enriquecer el aspecto y uso de esta, así como añadir nuevos elementos gráficos. Para hacerlo se parte de una aplicación que consta de una pantalla inicial donde hay un menú con 4 opciones. Se puede ver en la Figura 49.

- Cámara AR: la aplicación reconoce el entorno y objetos grabados por la cámara y hace saltar el evento correspondiente.
- Información: se pasa a una ventana nueva donde aparece un texto que da la bienvenida al museo y se explican las razones de por qué se ha hecho la aplicación.

Figura 49: Menú principal aplicación AR

- Actividades: es un enlace a la página web oficial del museo en el apartado de vistitas y actividades que se pueden realizar.
- Video: pasa a otra ventana donde se reproduce un vídeo de unos 2 minutos

Para hacer que la aplicación reconozca una imagen y objeto en concreto y salte un evento, en este caso un audio, video o un elemento gráfico, se necesita de un software específico. Se decidió usar Vuforia como software de reconocimiento del entorno real en combinación del programa Unity para poder diseñar la aplicación y todos sus elementos. Vuforia tiene diferentes softwares que permiten el reconocimiento y seguimiento de objetos, desde imágenes estáticas en 2D hasta objetos con forma compleja en 3D. Las diferentes herramientas que permiten hacer esto posible son: [16]

- Image Target: reconoce imágenes en 2D, para hacerlo el software compara la imagen que ve en la cámara con una base de datos previamente asignada. Las Image Target son normalmente usadas para campañas publicitarias y promoción de videojuegos.
- Multi Target: parecido a la Image Target pero compuesta por una combinación de imágenes dentro de un mismo objeto geométrico.
- Model Targets: reconoce y hace un seguimiento un objeto del mundo real por su tamaño. Para ello, se debe tener un modelo en 3D del objeto real en cuestión y proporcionarlo al software.
- Object Target: usa una herramienta llamada reconocimiento de objeto para poder reconocer objetos en 3D con una geometría más compleja como puede ser un juguete o un producto de consumo pequeño. Esta herramienta puede servir para crear nuevas y diferentes actividades interactivas con los objetos, por ejemplo, se puede hacer que se aumente el tamaño de un juguete y darle vida.
- Cylinder Targets: permite detectar objetos con forma cilíndrica o con forma de cono, lo hace al reconocer los laterales y la superficie superior plana. Es muy común usar este software para hacer que una app reconozca el embalaje de productos como son bebidas de soda y tazas de café entre otros.

- VuMarks: las VuMarks son imágenes características de Vuforia. Deben tener una forma en concreto pero dejando espacio para poder crearlas y diseñarlas al gusto del programador. Se pueden definir como el futuro del código de barras por su gran rango de posibilidades y autenticidad. Para crearlas se puede usar Adobe Illustrator con el Vuforia VuMark Designer plugin. Una VuMark siempre debe tener un contorno y un borde seguido de un espacio en blanco y luego un código de elementos, finalmente en el centro se encuentra la zona de background donde normalmente se pone el logo de la empresa o producto.

En los trabajos anteriores se decidió usar la Image Target como la imagen que debe reconocer la aplicación del museo para hacer saltar un evento. Esta está definida por un contorno negro, en la parte inferior se encuentra un título seguido del logo del Museo. En la parte superior hay un conjunto de bits aleatorios diferentes para cada Image Target.

Figura 50: Ejemplo de Image Target del museo

La aplicación del Museo tiene originalmente dieciséis eventos, los cuales se pueden ver cuando se reconoce una Image Target previamente definida. Estos eventos pueden ser audios, vídeos o imágenes gráficas creadas por un artista profesional. Los eventos que tenía la aplicación originalmente son:

- TV Philips: se muestra una parte de un programa de televisión de la misma época en que se comercializó este modelo de televisor, el programa se llama “Un dos tres, responda otra vez”.
- Tocadiscos Picu: se muestra un vídeo del funcionamiento de este tocadiscos, formado por dos altavoces y un reproductor donde se sitúa el disco de vinilo. Este tocadiscos es muy característico, ya que tiene la posibilidad de recogerlo y convertirse en un estuche portable.
- Magnetófono: donde se muestra un vídeo de cómo se usa este aparato, el cual fue el primero en grabar sonido sobre soporte magnético.
- Gramófono: se muestra un vídeo de cómo montar el gramófono y cómo se escuchaba la música a finales del siglo XIX.
- Fonógrafo: se muestra un vídeo de cómo se monta la bocina y se coloca el cilindro que contiene el sonido grabado previamente. Este aparato constituyó una revolución tecnológica ya que fue el primero en poder grabar y reproducir la voz humana.

- Telégrafo: constituye de un vídeo explicativo de cómo funciona el telégrafo.
- Teléfono sobremesa disco: consta de un vídeo de este teléfono que recibe una llamada y se escucha un extracto del libro de Don Quijote, en este se puede apreciar la calidad del audio que daba este aparato.
- Radio Onteniente: se muestra un video transparente donde habla Héctor García Miquel, profesor de la ETSIT y nieto de Salvador Miquel, creador de la emisora de radio Onteniente.
- Siphon Recorder: se muestra el vídeo El castillo de Olmedo donde se narra la historia de este objeto.

En cuanto a los eventos de audio, hay los siguientes:

- Receptor Radio: se muestra un audio de cómo se escuchaba la radio a través de este aparato.
- Decreto Telégrafo: donde se muestra un audio que cita el decreto de mujeres telegrafistas.
- Estación radiotelegráfica: se escucha un audio del código morse que significa Bienvenidos al museo de la telecomunicación.

Por último, hay dos eventos de ilustraciones:

- Centralita mujeres: se muestra un conjunto de ilustraciones de las mujeres telegrafistas, estas ilustraciones cambian de estilo dependiendo del estilo gráfico de la época.
- Lena: se superpone la imagen de Lena del museo con otras versiones donde se ha realizado procesado de imagen.

En el trabajo anterior, al añadir nuevos eventos y corregir algunos errores de funcionamiento se consiguió tener una aplicación funcional.

5.2. Mejora usabilidad

En cuanto a la usabilidad, se ha mejorado la aplicación tanto para el disfrute del usuario como para la edición de futuras versiones. En primer lugar, se han hecho pruebas de la aplicación inicial para ver donde haría falta la adición de un botón u otra funcionalidad. Se ha visto que, en la pantalla de información, no hay un botón visible para volver al menú inicial, es por ello que se ha añadido un recuadro junto con una flecha de color negro que representa al botón. Esto se puede ver en la Figura 51.

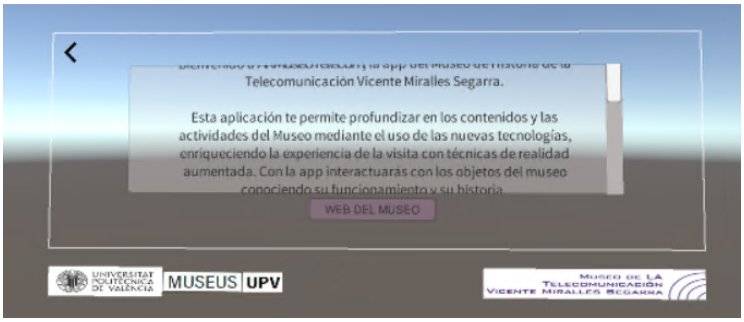

Figura 51: Boton añadido

Se ha definido, un Game Object vacío el cual tiene asignado dos scripts, el Menu (script) y el Navigator Handler (script) los cuales controlan las diferentes escenas que hay definidas en el proyecto y también un método público llamado *LoadMenuScene()* el cual se usará en el botón para volver al menú inicial.

```
public void LoadMenuScene()
{
    // called by "Vuforia Samples" button in AR scene UI menu
    // also called below in Update() if Android Back button pressed
    Menu.LoadScene(Menu.MenuScene);
}
```

Figura 52: Clase LoadMenuScene()

Dentro de este Game Object vacío, se define un canvas que será el recuadro que se ve en la imagen anterior. Dentro de este, se define el objeto botón que será transparente y encima se pone una imagen de una flecha de color negro. A este botón se le indica que cuando es pulsado se ejecute el método mencionado anterior, el cual hará que se cambie de pantalla.

En segundo lugar, desde el menú inicial, cuando se pulsa el botón de Actividades, la aplicación redirige a la página web del museo, en este caso al apartado de Actividades. Al probarse el funcionamiento de esta, se ha visto que la dirección de la web ha cambiado y por tanto este enlace no lleva a ningún sitio. Se ha tenido que cambiar la dirección web definida, esta se encuentra en el script Menu. Hay un método público llamado GoToActiv() donde se define el comportamiento de que ocurre si se pulsa el este botón del menú inicial. Se ha cambiado la url ahí y se ha puesto la correcta, se puede ver en la Figura 53.

```
public void GoToActiv()
{
    //UnityEngine.SceneManagement.SceneManager.LoadScene(PruebaScene);
    Application.OpenURL("http://museotelecomvlc.webs.upv.es/actividades/");
}
```

Figura 53: Clase donde se especifica la URL

Estas dos mejoras de la aplicación se pueden ver como usuario, es decir, cualquier persona que se descargue e instale la aplicación podrá tener una mejor experiencia de la visita al añadir el botón

de cambio de escena y arreglar el link. En último lugar, el cambio que se ha hecho no será notable por un usuario, sino que solo se verá por el programador de la aplicación.

Dentro del editor de Unity se ha cambiado la organización de los eventos que saltan dado una Image Target. Los eventos se sitúan en una escena llamada ARenviroment, dentro de esta escena, por una parte, está definido un canvas con una flecha para volver atrás. Por otra parte, está definida la cámara AR de Vuforia, la cual es un Game Object de Unity que incluye el software de Vuforia-Behaviour, necesario para crear una aplicación de realidad aumentada. También en esta escena se han colocado los Image Target con sus respectivos eventos. Están posicionados dentro del margen de la cámara AR e inicialmente se encontraban amontonados. La posición donde se encuentran los Image Targets respecto dentro del contorno de la cámara AR es irrelevante, sin embargo, lo importante es la posición del objeto Image Target con respecto al audio, vídeo o ilustración del evento que corresponda.

Cuando se tienen que hacer algún cambio de las Image Target es difícil identificarlas dentro de la escena, ya que están todas de manera solapada. Mientras se realizaban las tareas de este proyecto se ha tenido dificultad a la hora de poder editar un evento a causa de esto. Es por ello que se ha cambiado la disposición de los objetos para tener una mejor visibilidad para una futura edición o mejora.

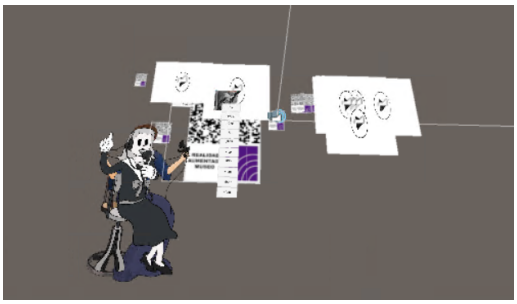

Figura 54: Antes

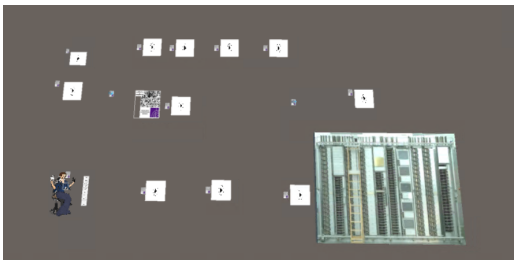

Figura 55: Después

Con estos cambios, se ha conseguido mejorar la aplicación tanto para la perspectiva de un usuario como para la perspectiva del programador el cual debe añadir y corregir la funcionalidad de esta.

5.3. Nuevos eventos

La aplicación de realidad aumentada tiene como principal atracción el reconocimiento de las Image Targets y la visualización de un vídeo o audio. En este punto, se van a explicar las nuevas adiciones de realidad aumentada que se han puesto en la aplicación y además de las correcciones pertinentes que se han hecho a eventos ya existentes. En total, se han editado cuatro eventos ya existentes y se ha creado uno nuevo. A continuación, se va a describir el trabajo hecho.

5.3.1. Estación radio morse

La image target llamada EstRadioMorse es la que hace que salte un audio de una retransmisión de código morse, se ha decidido editarlo y hacer que aparezcan las letras del abecedario español al mis-

mo tiempo que se salen las letras morse y se escucha el audio. En un primer intento, se utilizó una animación de Unity. Esta es una herramienta del programa que permite programar acontecimientos o comportamientos de un GameObject dada una condición, sin la necesidad de programarlo con scripts. Usando la animación, se ha intentado hacer que salieran las letras al mismo tiempo que se escuchan en el audio, es decir si sale la letra B, simultáneamente debe escucharse en el audio "-…". Al hacerlo, surge un problema si el evento se ve interrumpido y se vuelve a iniciar, el audio y la animación de las letras se desincroniza. Se ha intentado vincular el audio con las letras probando diferentes propiedades que había dentro de la ventana del Inspector pero sin ningún resultado. Por lo tanto, con el consejo de la tutora, se ha decidido hacer un vídeo transparente de forma similar a como funciona el vídeo de la radio Onteniente.

Antes de hacer el vídeo morse completo, se ha hecho una prueba con el audio y unas letras para saber de su funcionamiento y también saber qué formato de vídeo puede dar mejor resultado. Haciendo esta prueba, se ha visto que el mejor formato es el de mp4, el cual está reconocido tanto por un dispositivo Windows, Android Linux y IOS según la documentación de Unity. Además, se ha hecho uso del shader AlphaIsBlack el cual reconoce el fondo negro de un vídeo y lo trata como si fuera Alpha, es decir, como si fuera transparente. El vídeo se ha realizado utilizando un editor llamado "VSDC Free Editor Video" el cual se puede descargar de forma gratuita. Este editor te permite editar video y audio de forma gratuita, es compatible con los formatos de video más populares y además tiene funcionalidades que te permiten editar el vídeo y añadir los elementos que desees.

El editor VSDC está compuesto por un menú inicial, una pantalla donde se muestra el vídeo y un cronograma donde se sitúan los elementos dependiendo en que momento aparecen durante la reproducción. Se pretende hacer un vídeo donde aparezcan las letras en el momento exacto que se escuchan. Para ello, se debe hacer un trabajo inicial y apuntar en qué segundo sale cada letra de la frase *Bienvenidos al Museo de la Telecomunicación*. Esta parte ha sido bastante costosa ya que no estaba familiarizada con la escucha y reconocimiento del código morse. Finalmente, se ha podido tomar notas de cuando poner exactamente las letras y así reflejarlo en el editor. En la parte inferior del editor, se ha puesto el audio sobre el que se va a trabajar y encima se han puesto las letras, tanto la del abecedario español como la de morse. Cabe destacar que la misma letra en ambos alfabetos sale al mismo tiempo y según avanza el audio van saliendo las siguientes letras. Luego, en la parte del centro del editor, se puede cambiar la posición y tamaño de las imágenes puestas, en este caso es la frase que se escucha.

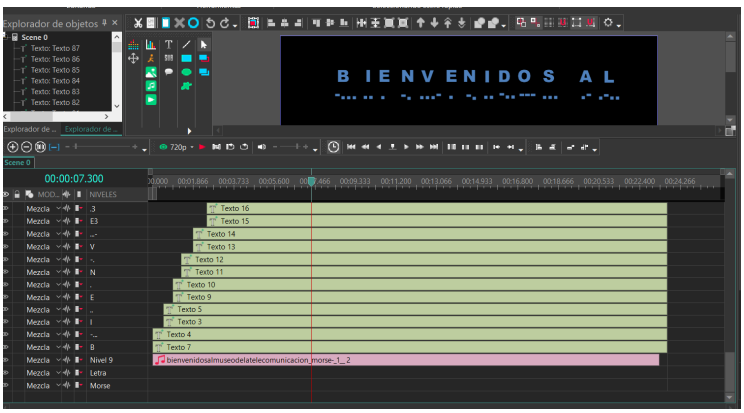

Figura 56: Ventana editor VSDC

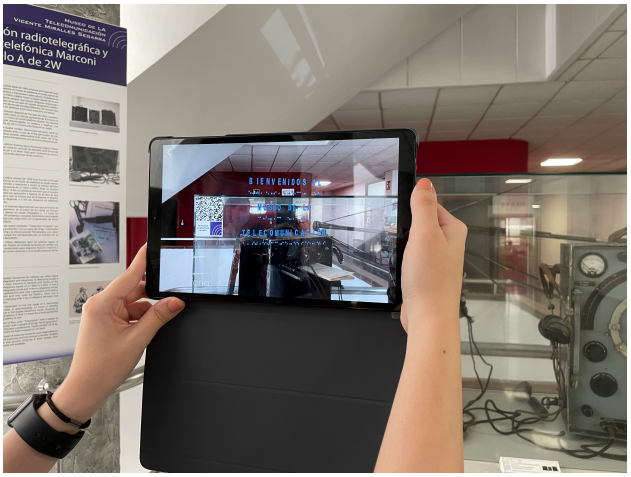

Figura 57: Evento Morse

Para añadirlo dentro del entorno del programa de Unity, se ha seguido los mismos pasos que con otros vídeos, donde se ha añadido la funcionalidad de pausa y play del vídeo al pulsar sobre este en la pantalla del dispositivo móvil. Hay que resaltar que se ha marcado la opción de Extended Tracking para hacer que el vídeo tuviera más estabilidad a la hora de su reproducción. El resultado final, se puede apreciar en la siguiente Figura 57.

5.3.2. Televisor Philips

Otro elemento que se ha corregido es la Image Target que corresponde al evento del Televisor Philips. Aquí debe salir un vídeo de un programa el cual debe estar superpuesto dentro del recuadro de la pantalla del televisor, así dando una sensación de estar viendo el aparato en funcionamiento. Se ha tenido que corregir la posición del vídeo según la Image Target ya que estaba inicialmente puesto en la parte media de un lateral de la vitrina y actualmente se ha movido y por lo tanto el vídeo ya no aparecía dentro del televisor. Para ello, en el editor de Unity, se ha movido el objeto correspondiente al vídeo. Como ahora la Image Target está situada en la parte superior izquierda. Dentro del editor de Unity, se ha movido el objeto del vídeo más hacia abajo para conseguir ponerlo en la posición deseada.

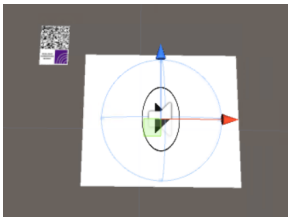

Figura 58: Cambio posición vídeo en Unity

Figura 59: Evento TV Philips

5.3.3. Ilustraciones mujeres telefonistas

La siguiente corrección corresponde al evento de las mujeres telefonistas. Aquí aparecen una ilustración que representa a una mujer telefonista, pero la ilustración cambia dependiendo del estilo artístico de la época. Para cambiar de ilustración hay una barra lateral con botones y pulsando uno se cambia de imagen. Cuando saltaba este evento, la posición del menú estaba bastante más arriba que donde se situaban las ilustraciones, haciendo difícil la visualización de ambas cosas al mismo tiempo. Se ha procedido de forma similar a como se ha corregido el evento anterior, dentro del editor de Unity se ha cambiado la posición de este menú.

Figura 60: Telefonistas en editor de Unity

Figura 61: Evento mujeres telefonistas

5.3.4. Vídeo telégrafo

Como última mejora de eventos ya existentes, se ha corregido el vídeo del evento del telégrafo. En este sale el profesor de la ETSIT, Luis Sempere, explicando el funcionamiento de un telégrafo y cómo eran capaces los telegrafistas de reconocer y escribir este idioma. Este vídeo, cuando se visualizaba con algunos dispositivos Android, se veía de color verde en los segundos iniciales. Para saber porque ocurría esto, se hicieron pruebas al cambiar algunas propiedades dentro de Unity, además también se consultó el foro de Unity para ver si algún usuario había tenido un problema similar. Haciendo estas pruebas y luego de consultar el foro, no se obtuvo ningún resultado. Este problema parecía un poco frecuente ya que existen otros eventos que hacen saltar vídeos y no tienen ningún problema durante su reproducción. Por ello, se reprodujo el vídeo en un reproductor del ordenador y este se veía y escuchaba perfectamente. Se llegó a la conclusión que el vídeo podía estar dañado de alguna forma y es por ello que en algunos dispositivos no se podía visualizar correctamente. Para comprobarlo, se ha recortado el vídeo quitando los primeros segundos donde aparecía de color verde, de esta forma se podía ver el vídeo correctamente.

Para solucionar este problema, la tutora me ha proporcionado el vídeo original, el cual se grabó con una cámara Sony y esta en formato MTS. Para pasarlo a un formato reconocido por Unity como es el formato MP4 se puede hacer de diferentes formas. Para estar seguros de que la conversión no afectaba a la codificación del vídeo se han hecho tres versiones usando tres conversores diferentes. Las dos primeras se han pasado a formato mp4 usando dos convertidores de uso gratis en internet y luego se han recortado usando el editor VSDC EDITOR. La tercera se ha usado el editor de imágenes y vídeo de Windows donde se ha recortado y guardado como formato MP4.

Una vez se ha tenido los vídeos preparados se han hecho pruebas con la Tablet del museo, la cual en un principio no se visualizaba correctamente el video. Se ha decidido usar el vídeo editado y convertido con el editor de Windows ya que su funcionamiento y visualización no han dado ningún problema. El resultado de la corrección se puede observar en la Figura 62.

Figura 62: Evento telégrafo

5.3.5. Centralita mecánica

Finalmente se ha añadido un nuevo evento con una nueva Image Target. Este evento es diferente a los hechos anteriormente ya que se pretende mostrar una imagen en 3D de una centralita mecánica. [17]

Esta centralita es una maqueta central de conmutación “Rotary”, está expuesta en el Museo Joaquín Serna de la Escuela Técnica Superior de Ingenieros de Telecomunicación de la UPM. Desde mediados del siglo XX hasta 1996 estuvo en servicio esta centralita, la cual era utilizada por Telefónica para dar servicio de comunicación entre dos abonados. Tiene unas dimensiones de 317x565x28.5cm, es decir unos tres metros de altura y casi seis metros de largo.

Se pretende hacer una versión digital de esta centralita y hacer que se muestre en la aplicación con un tamaño lo más próximo posible a la realidad, para así dar una idea al usuario de cuan grande es este aparato. Para obtener una imagen en 3D se ha trabajado a partir de una imagen de esta centralita de la parte delantera. Para obtener lo que serían las imágenes laterales, se ha hecho uso de un programa de edición de imágenes que se puede obtener por los programas disponibles para los alumnos de la UPV desde Polilabs, este editor se llama GIMP 2. Con él se ha recortado la imagen original y se ha editado para formar dos laterales:

Figura 63: Lateral izq y drcho

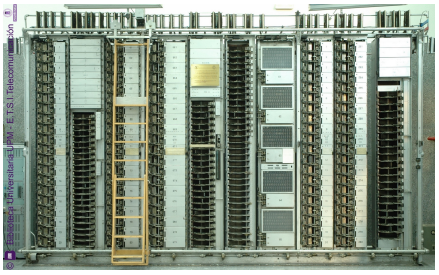

Figura 64: Frontal

A partir de estas tres imágenes se va a realizar el modelo en 3D. Se ha utilizado el programa Blender, el cual es un software de libre distribución utilizado para poder crear gráficos y diseños en 3D. Diseñadores gráficos emplean esta herramienta para dar forma y color a juegos, dotar de efectos a animaciones, modelar realidades virtuales entre otras. Este software no es una herramienta extraña para el museo ya que se ha usado anteriormente para dar forma y recrear el edificio de la ETSIT en el trabajo de realidad virtual.

La centralita se ha creado a partir de un cubo, con el cual se le ha dado una forma de rectángulo alargado y con profundidad, respetando las dimensiones reales. El siguiente paso es poner las imágenes anteriores como la textura de los seis lados. Este paso me ha resultado bastante complejo ya que no tenía ninguna experiencia previa trabajando con programas de modelado. Se ha buscado información de cómo usar Blender así como se han visualizado diversos tutoriales sobre cómo pintar y dar una textura a un objeto en concreto.

Según la información recogida, es necesario abrir una segunda ventana en el editor de Blender llamada UV editing donde aparecerá un esquema de los costados de la figura seleccionada.

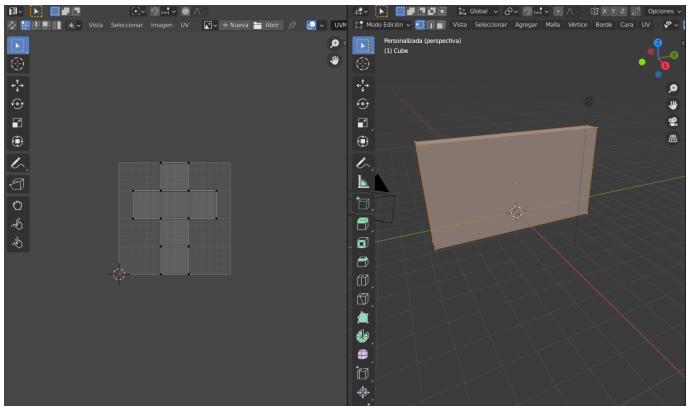

Figura 65: Ventana editor de Blender

En la nueva ventana, se puede seleccionar una imagen guardada en el equipo y definirla como la textura de un lado en concreto. Este proceso se sigue para las seis caras de la figura y queda un

resultado como se ve en la Figura 66.

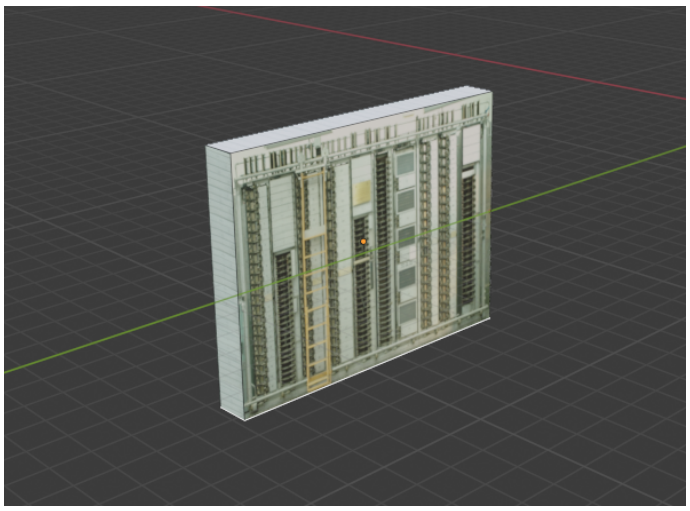

Figura 66: Modelado Centralita en Blender

Una vez terminada la edición del modelado, se debe exportar a Unity. Es importante saber que para integrarlo en el programa donde se ha realizado la aplicación, el modelo gráfico de la centralita debe estar exportado a un formato en concreto, este es el .dae. También cabe destacar que no solo se debe importar este archivo sino también las texturas con el cual se ha ilustrado, en caso contrario, en Unity solo se verá el objeto y no las imágenes que se han renderizado en el paso anterior.

El siguiente paso es abrir el proyecto de la aplicación de AR con el programa Unity, y guardar dentro el nuevo objeto y sus texturas. Seguidamente, se tiene que crear un nuevo objeto del tipo Image Target para así hacer que salte el nuevo evento. En el menú superior se abre GameObject>Vuforia>Image, de esta forma se incluye en la escena el objeto que representa la Image Target.

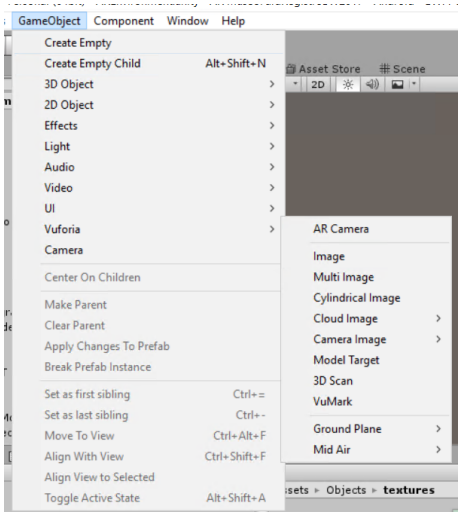

Figura 67: Añadir Image Target en Unity

Le ponemos el nombre de Centralita Mecanica y en la ventana de Inspector, indicamos que Image

Target debe reconocer el software. Se indica debajo de la opción de Database, donde previamente se ha añadido dicha imagen a la database de la cuenta de Vuforia del museo. Cuando se usa Vuforia en combinación con Unity, se debe indicar qué evento salta por cada imagen u objeto observado por el dispositivo. Se indica al poner el evento como un objeto hijo de la Image Target en concreto. En la jerarquía de la ventana Hierarchy debe quedar como se ve en la Figura 68.

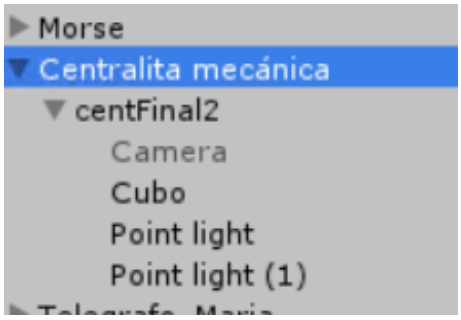

Figura 68: Hierarchy

Con estos Game Objects definidos, la aplicación ya puede mostrar ante los ojos del usuario la figura gráfica de la centralita mecánica.

Junto con la tutora, se ha decidido imprimir la Image Target con un tamaño de dos hojas A3 para que sea visible y reconocible por el dispositivo móvil desde una gran distancia. También se han hecho muchas pruebas en cuánto a que tamaño debe tener la figura para que se pueda observar de manera cómoda por el usuario.

Realizando estas pruebas, se han hecho algunos cambios sobre el objeto Camara AR. A este se le ha definido un tamaño más grande en cuanto se refiere a al marco de este, se ha definido como 1000 para que la figura no saliera recortada y mejorar su estabilidad.

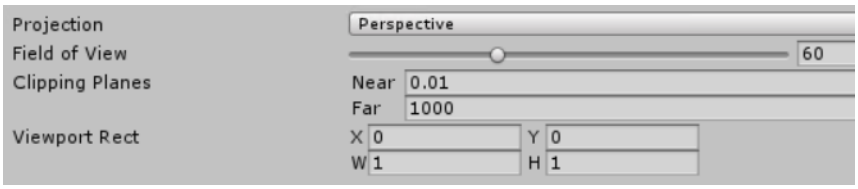

Figura 69: Propiedades de Camara AR

Después de realizar las pruebas pertinentes se ha conseguido tener un evento estable de la visualización de la centralita, se ha situado la Image Target en un lateral de la escalera principal del edificio 4P. El resultado es el siguiente:

Figura 70: Evento Centralita Mecánica

5.3.6. Publicación app en página web

El objetivo inicial de haber creado esta aplicación es poder hacerla disponible para cualquier usuario del museo. Es por ello que cuando se ha tenido la aplicación de realidad aumentada para Android terminada y en funcionamiento, se ha publicado en la página web del Museo para que cualquier persona pueda descargarla e instalársela.

La aplicación se ha subido en una página que se encuentra en el apartado Visitas>App de Realidad Aumentada del menú principal de la página web. Cabe destacar que la aplicación es un archivo .apk y Wordpress no deja subir archivos de un formato que la herramienta no ha reconocido. Los formatos que reconoce Wordpress se muestran en la Figura 71. [18]

Por lo tanto, para subir el archivo .apk hay que comprimirlo previamente en un archivo .zip. Esto hace que para instalarse la app, un usuario debe descargarse el comprimido, descomprimirlo e instalarlo, un proceso que se hace largo. Por esa razón, se ha buscado una forma de poder subir otros formatos de archivos a esta plataforma. En un primer lugar, la solución más común es la de editar el código PHP, el cual define la forma de visualización y uso de las páginas Wordpress. Editar estos archivos sin tener previa formación con el lenguaje PHP y experiencia en como usarlo dentro de Wordpress, puede llevar al deterioro de la página web.

Finalmente se ha hecho uso de un plugin, este se llama *WP Add Mime Types*. Este plugin, permite

Images	Documents	Audio	Video
.jpg	.pdf (Portable Document Format; Adobe Acrobat)	.mp3	.mp4, .m4v (MPEG-4)
.jpeg	.doc, .docx (Microsoft Word Document)	.m4a	.mov (QuickTime)
.png	.ppt, .pptx, .pps, .ppsx (Microsoft PowerPoint Presentation)	.ogg	.wmv (Windows Media Video)
.gif	.odt (OpenDocument Text Document)	.wav	.avi
.ico	.xls, .xlsx (Microsoft Excel Document)		.mpg
	.psd (Adobe Photoshop Document)		.ogv (Ogg)
			.3gp (3GPP)
			.3g2 (3GPP2)

Figura 71: Archivos reconocidos por Wordpress

subir a una página de Wordpress un archivo que no está originalmente en la lista de la Figura 71. Para indicar que se quiere permitir la subida de la aplicación en formato .apk, se va al apartado de Ajustes>Mime Tipe Settings. Dentro debe aparecer un apartado con una lista de nombres y definiciones de diferentes tipos de archivos que gracias al plugin ahora sí que se podrán subir. En un principio, esta lista no contiene la definición del archivo .apk, hay que añadirlo. Para ello, se va al final de esta página donde pone Add Values y se escribe el siguiente párrafo:

```
apk = application/vnd.android.package-archive
```

Se guardan los cambios y deberá aparecer en la lista mencionada anteriormente este formato de archivo. Finalmente, se va a la página App de Realidad Aumentada y se edita su contenido para cambiar el archivo .zip que había previamente por el archivo .apk que contiene la aplicación desarrollada.

Capítulo 6

Publicación en Google Play Store y App Store

En esta sección se va a explicar los pasos a seguir para publicar la aplicación de realidad aumentada hecha con el programa Unity y Vuforia para los dispositivos con el sistema operativo de Android e iOS.

6.1. Publicación en Google Play Store

Para publicar una aplicación hecha con Unity en Google Play Store, esta debe estar firmada desde dentro de la propia herramienta de Unity. Para ello, se abre el proyecto con Unity y se va al apartado de Build Settings>Player Settings, aquí se pueden cambiar distintos aspectos de la compilación y la publicación del proyecto. En este caso se va al apartado de Publishing Settings.

El primer paso es crear un archivo que será el contenedor de las llaves también conocido como *keystore*. Para crearlo, se pulsa la pestaña de *Create a new keystore* y poner una contraseña a este contenedor. Seguidamente, se crea un nuevo alias y se escribe su contraseña. Es muy importante, guardar bien ambas contraseñas para poder actualizar la publicación de la aplicación en Google Play store. Esto se ve en la Figura 72. En el caso de que se usara una *keystore* y un alias ya existente se debe seleccionar la pestaña de *Use Existing Keystore* y escribir las contraseñas pertinentes. [19] [20]

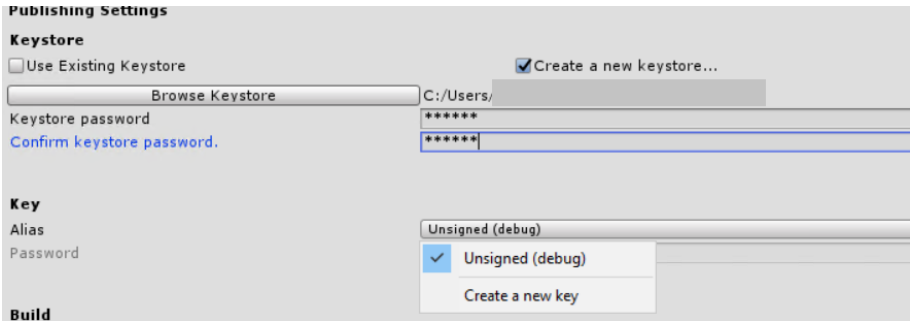

Figura 72: Firmar con Unity

6.2. Publicación en App Store

La ventaja de haber usado Unity como herramienta para desarrollar la aplicación de realidad aumentada es la facilidad en que puedes adaptar un proyecto a diferentes plataformas. Para pasar la aplicación a plataforma iOS, se debe entrar en Build Settings e indicar que se quiere compilar a la plataforma de iOS, esto se hace pulsando el icono de la plataforma que se desea y pulsando el botón Switch Platform. Finalmente, el apartado Build Settings debe quedar como el de la Figur 73.

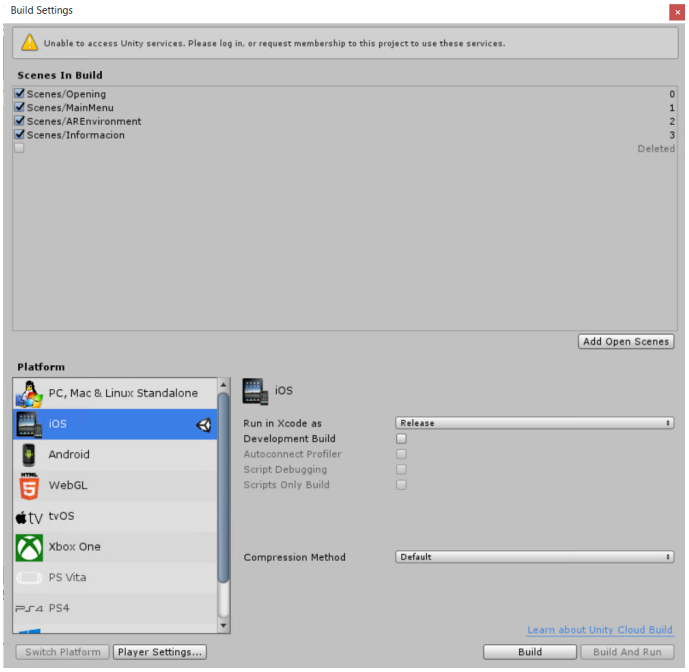

Figura 73: Build Settings para iOS

El siguiente paso será compilar el proyecto para obtener la aplicación iOS, para ello, se deben ajustar algunos parámetros del Player Settings, en caso contrario saltarían errores en la consola. Dentro del apartado de *Other Settings* se deben definir las siguientes propiedades:

- Camera Usage Description. Donde se va a escribir lo siguiente:" Camera access required for target detection and tracking"
- Target Device. Donde se indica que el dispositivo de uso será de iPad y iPhone.
- Target SDK. Donde se indica que será Device SDK.
- Target mínimum iOS version. El cual se pondrá como 9.0.

Se puede ver mejor en la Figura 74.

Una vez cambiado esto, se puede pasar a su compilación. Para ello, desde Build Settings>Build y se elije la carpeta donde se quiere guardar el resultado de la compilación.

Cuando se compila para dispositivos Android, el resultado es un solo archivo en formato .apk el cual se puede instalar y probar en un cualquier dispositivo con este sistema operativo. En el caso

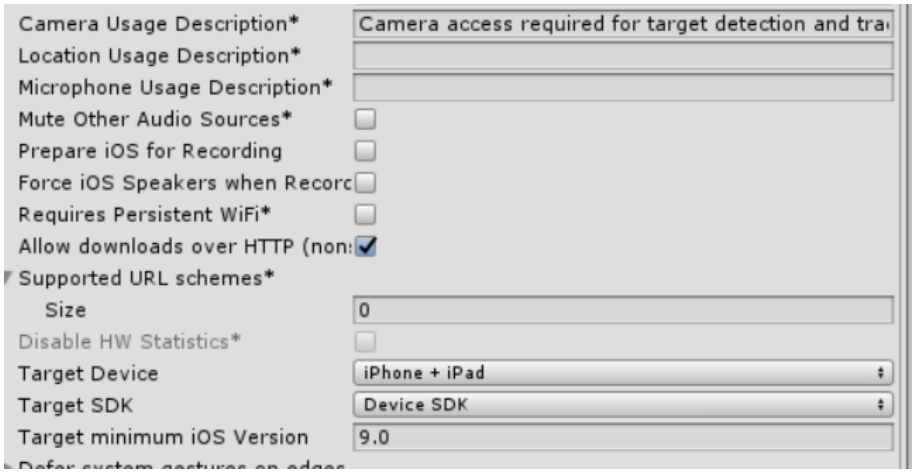

Figura 74: Propiedades de Player Settings

la compilación a iOS, el resultado son diferentes archivos y carpetas, como se ve en la ilustración 75.

Apple tiene sus propias políticas de seguridad y no permite instalar una aplicación a uno de sus dispositivos sin que esta se haya descargado directamente del App Store. Aun así, se puede hacer una prueba con un iPhone o un iPad. Para hacerlo se necesita además del dispositivo Apple de un Mac el cual debe tener instalada la última versión de software además de tener instalado el programa Xcode (se puede descargar desde el App Store). También es importante que el dispositivo donde se va a probar la aplicación tenga una versión de software 14.5.

Para instalar la aplicación, en primer lugar, se abre Xcode y se inicia sesión con un ID de Apple, como se ve en la Figura 76. Se abre General y en el apartado *Signing* se introduce la cuenta de Apple.

Seguidamente, se debe abrir el proyecto completo compilado, es decir la carpeta con los archivos que se muestran en la Figura Anterior. Una vez abierto, se debe conectar el iPhone al ordenador mediante un cable y dentro de Xcode seleccionar dicho dispositivo desde el menú superior. Finalmente, se desbloque el iPhone y se hace un run de la aplicación desde Xcode, esto hará que se instale dicha aplicación en el dispositivo y se pueda probar.

En el caso que se quiera publicar la aplicación en App Store, se debe tener un ID de Apple y además este debe ser de una cuenta de desarrollador, esto significa que se tiene que dar de alta en el programa de desarrolladores de Apple el cual tiene un coste de unos 100$ al año. Luego, hay que subir la aplicación al entorno del programa de desarrolladores. Esto se puede hacer desde Xcode y seguidamente hay que esperar para la aprobación por parte de Apple de nuestra app. Cuando se tenga el visto bueno se puede publicar en la tienda de App Store. [21] [22]

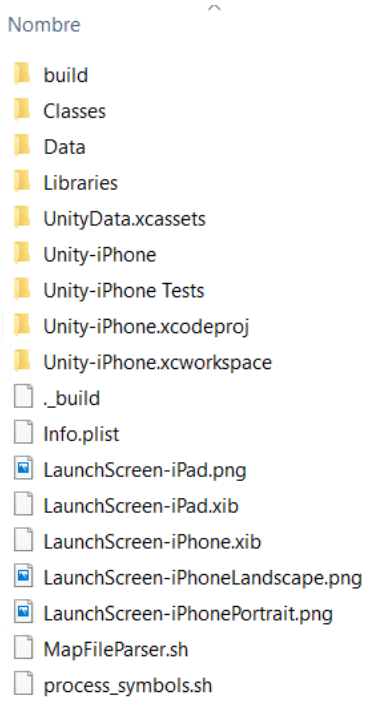

Figura 75: Resultado compilación

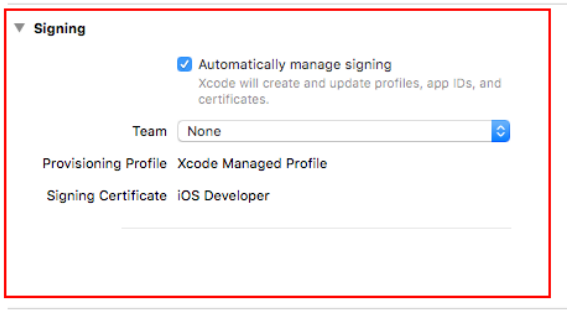

Figura 76: Añadir ID Apple

Capítulo 7

Conclusiones y líneas futuras

En cuanto al objetivo principal de este trabajo, el cual es dar a conocer y mejorar la calidad de las visitas del museo, se puede decir que se ha conseguido gracias a la puesta en marcha de la visita virtual en la página web, la adaptación y actualización del proyecto a las nuevas gafas de realidad virtual. Por último, la mejora de la funcionalidad de la aplicación de realidad aumentada, para enriquecer la visita presencial a las instalaciones del museo.

Es importante destacar que la tecnología de la realidad virtual y la aumentada son muy novedosas y con las que no había trabajado previamente. Al realizar este trabajo, he podido aprender, no solamente el alcance que tienen las nuevas tecnologías, sino que también he podido mejorar mis habilidades como ingeniera al aprender a enfrentarme a problemas desconocidos, mejorar mi comunicación tanto oral como escrita y usar mi creatividad.

Se ha conseguido poner en marcha tres aplicaciones diferentes relacionadas con el museo, aunque siempre se puede hacer sitio para una mejora. Con respecto al proyecto de realidad virtual adaptado a las gafas, se podría añadir los nuevos eventos de vídeo que hay en la aplicación RA. En el proyecto web se pueden expandir las utilidades para permitir una interacción con el entorno al igual que se ha hecho en el proyecto adaptado a las gafas virtuales.

Bibliografía

[1] María de los Desamparados Bachiller Martin Maria Carmen Romero Mora. *Los inicios de la telecomunicación en la Comunidad Valenciana. Una publicación del Museo de la Telecomunicación Vicente Miralles Segarra*.

[2] *Página oficial Museo Vicente Miralles Segarra*. URL: `https://museotelecomvlc.webs.upv.es/`.

[3] UPValenciaX. *Introduction to video game development with Unity*. URL: `https://www.edx.org/es/course/introduction-to-video-game-development-with-unity`.

[4] *Página oficial del museo de Munich*. URL: `http://www.deutsches-museum.de/en/information/`.

[5] *Página oficial del museo MUNCYT de Madrid*. URL: `http://www.muncyt.es/`.

[6] Alejandro Vecina Dolz. *DESARROLLO DE UNA VERSIÓN VIRTUAL DEL MUSEO DE LA TELECOMUNICACIÓN VICENTE MIRALLES SEGARRA*. 2019.

[7] Luis Pérez Pau. *VISITA VIRTUAL AL MUSEO DE LAS TELECOMUNICACIONES VICENTE MIRALLES*. 2015.

[8] *Página oficial del foro de Unity*. URL: `https://forum.unity.com/`.

[9] *Manual Unity versión 2019.2*. URL: `https://docs.unity3d.com/2019.2/Documentation/Manual/UnityManual.html`.

[10] user: UnityLightning. *WebGL Lighting Tips*. 21st of April de 2021.

[11] *Página oficial de la documentación de Oculus*. URL: `https://developer.oculus.com/documentation/unity/unity-import/`.

[12] Ana Javornik. "The Mainstreaming of Augmented Reality: A Brief History". En: (oct. de 2016). URL: `https://hbr.org/2016/10/the-mainstreaming-of-augmented-reality-a-brief-history`.

[13] *A brieg Hisotry of Augmented Reality*. URL: `https://numerized.com/augmented-reality/history-augmented-reality/`.

[14] Antonio Castellanos López. *REALIDAD AUMENTADA PARA LA MEJORA DE LA VISITA AL MUSEO DE LA TELECOMUNICACIÓN*. 2017-2018.

[15] Ángel Martínez Muñoz. *DESARROLLO DE UNA APLICACIÓN DE REALIDAD AUMENTADA PARA LA VISITA ENRIQUECIDA AL MUSEO DE LA TELECOMUNICACIÓN*. 2019.

[16] *Manual uso de Vuforia*. URL: `https://library.vuforia.com/features/objects/vumark.html`.

[17] bibliotecaetsit. “Maqueta de central de conmutación “Rotary””. En: (Ene de 2017). URL: http://biblog.etsit.upm.es/?p=17269.

[18] *How to upload APK file in WordPress 2021?* URL: https://wpintensity.com/upload-apk-file-wordpress/.

[19] “Como colgar una aplicación en Google Play Store”. En: (sep. de 2016). URL: https://www.antevenio.com/blog/2016/09/como-colgar-una-aplicacion-en-google-play/.

[20] Marta. “Como firmar una .apk de Unity”. En: (mar. de 2018). URL: https://martra.uadla.com/como-firmar-un-apk-de-unity-para-subirlo-a-la-google-play-store/.

[21] “Cuenta desarrollador Apple”. En: (). URL: https://www.devsdna.com/como-publicar-una-app-en-app-store/.

[22] “Como probar una app en dispositivo Apple”. En: (). URL: http://taniacastellano.com/compilar-app-ios-en-un-dispositivo-sin-licencia-de-apple/.

Parte II

Anexos

Figura 77: Image Targer y evento centralita mecánica ’Rotary’