GRADO EN INGENIERÍA EN TECNOLOGÍAS INDUSTRIALES

TRABAJO FIN DE GRADO

APLICACIÓN DE LA REALIDAD AUMENTADA PARA EL MANTENIMIENTO DE PLANTAS INDUSTRIALES

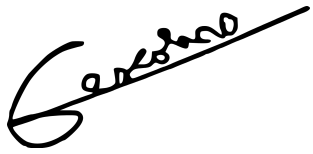
Autor: Guillermo Escolano Hovine

Director: José Antonio Rodríguez Mondéjar

Madrid

Declaro, bajo mi responsabilidad, que el Proyecto presentado con el título Aplicación de la realidad aumentada para el mantenimiento de plantas industriales en la ETS de Ingeniería - ICAI de la Universidad Pontificia Comillas en el curso académico 2020/21 es de mi autoría, original e inédito y no ha sido presentado con anterioridad a otros efectos.
El Proyecto no es plagio de otro, ni total ni parcialmente y la información que ha sido tomada de otros documentos está debidamente referenciada.

Fdo.: Guillermo Escolano Hovine *Fecha: 24/ 08/ 2021*



Autorizada la entrega del proyecto
EL DIRECTOR DEL PROYECTO

Fdo.: José Antonio Rodríguez Modéjar *Fecha: 24/ 08/ 2021*

Firmado por RODRIGUEZ MONDEJAR JOSE ANTONIO - 29057313X el día 24/08/2021 con un certificado emitido por AC FNMT Usuarios

GRADO EN INGENIERÍA EN TECNOLOGÍAS DE TELECOMUNICACIÓN

TRABAJO FIN DE GRADO

APLICACIÓN DE LA REALIDAD AUMENTADA PARA EL MANTENIMIENTO DE PLANTAS INDUSTRIALES

Autor: Guillermo Escolano Hovine

Director: José Antonio Rodríguez Mondéjar

Madrid

Agradecimientos

Quiero agradecer a todas las personas que me han acompañado en este gran proyecto. Sobre todo, a mis padres que siempre han confiado en mi y me han apoyado.

APLICACIÓN DE LA REALIDAD AUMENTADA PARA EL MANTENIMIENTO DE PLANTAS INDUSTRIALES

Autor: Escolano Hovine, Guillermo.
Director: Rodríguez Mondéjar, José Antonio.
Entidad Colaboradora: ICAI – Universidad Pontificia Comillas

RESUMEN DEL PROYECTO

En este proyecto se ha creado una aplicación que permite obtener información sobre las máquinas de la minifabrica a través del reconocimiento de imágenes, ayudando a posibles operarios a solucionar problemas con los que se puedan encontrar.

Palabras clave: Realidad Aumentada, Industria 4.0

1. Introducción

La cuarta revolución industrial, también conocida como Industria 4.0 esta marcada por la aparición de nuevas tecnologías, como pueden ser la robótica, el Big Data, la Inteligencia Artificial (AI), el Machine Learning (ML) o el Internet of Things (IoT) entre muchas otras. Esta revolución busca combinar estas tecnologías con los métodos de producción tradicionales y transformarlos para obtener sistemas conectados más eficientes [1].

Se prevé que otro de los pilares de este cambio sea la Realidad Aumentada (AR), una tecnología que permite visualizar el mundo real añadiendo información que mejora la interacción del usuario con el exterior en tiempo real.

2. Definición del proyecto

El objetivo de este proyecto es hacer una aplicación basada en la realidad aumentada que permita a sus usuarios (técnicos, operarios, estudiantes, etc...) una realización más eficiente de su trabajo aportando por ejemplo información sobre el estado de la actual de la maquina. Se busca hacer también una herramienta que permita la familiarización con la maquinaria a trabajadores que se enfrentan a ella por primera vez dando información básica sobre esta. Esta aplicación permitirá reconocer imágenes que el usuario elija, y que servirá como identificador de la maquina de interés. Al reconocer la imagen, la aplicación dará al usuario la opción de obtener más información sobre la maquina. Se crear también un sistema que permita crear plantillas para cada maquina que el usuario desee, conteniendo la información útil de cada una, implementando la posibilidad al usuario de acceder a su galería y escoger la imagen de reconocimiento o haciéndola con la cámara de fotos del dispositivo.

3. Descripción del modelo/sistema/herramienta

La aplicación esta basada en el motor de videojuegos de Unity que permite crear aplicaciones de forma sencilla. La aplicación estará dividida en tres secciones. La primera será una ventana que permitirá ver la información de la maquina una vez esta haya sido reconocida.

La segunda ventana es en la que se realiza la detección de la maquina con la ayuda del software de Vuforia que permite reconocer objetivos que están guardados en una base de datos online. Una vez reconocido el objetivo, la aplicación da la opción de obtener más información sobre la maquina asociada a ese objetivo (redirige a la ventana 1).

Por ultimo, la tercera ventana permite crear una plantilla para una nueva maquina. Para ello es necesario crear un nuevo objetivo, es decir una imagen que estará vinculada a una única maquina. Se ha desarrollado la opción de crear esta imagen usando la cámara de fotos del dispositivo o accediendo a los archivos guardados en el dispositivo. Una vez elegido el objetivo, el usuario introduce el nombre de la maquina y su información. Al crear la nueva plantilla, la aplicación se comunica con la base de datos de Vuforia y añade la nueva información. Se ha diseñado la plantilla de forma que, si un error se produce en alguna etapa de su creación, un mensaje de error podrá ser visualizado por el usuario, describiéndole el tipo de error del que se trata, mejorando substancialmente la accesibilidad de la aplicación. La arquitectura de la aplicación se puede ver en la figura 1.

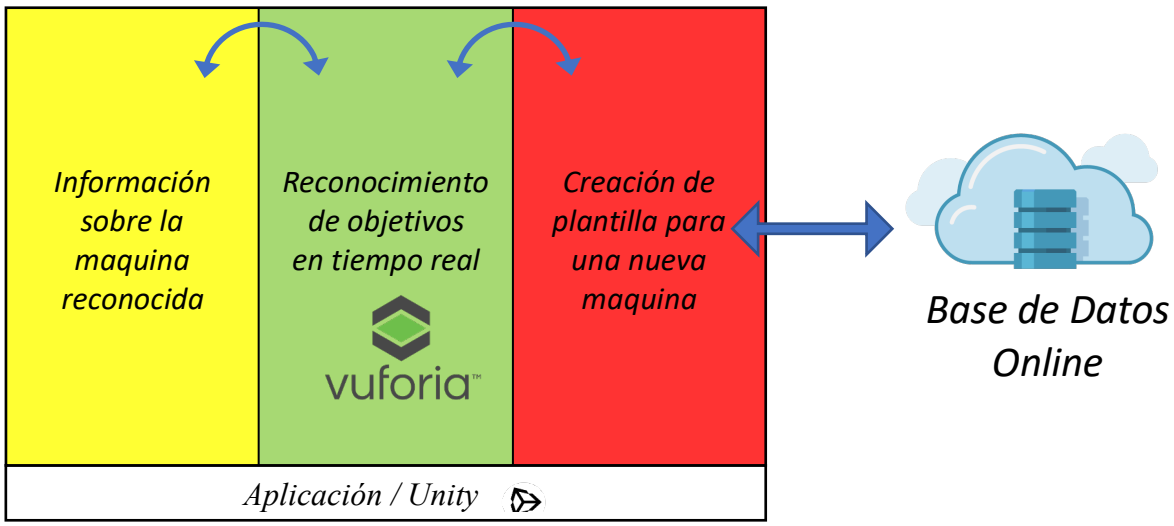


Figura 1 - Arquitectura de la aplicación

4. Resultados

En este proyecto se ha conseguido crear una aplicación que responde a las especificaciones deseadas, es decir, permite la identificación de imágenes seleccionadas por el usuario que identifican una maquina en particular y habilita la posibilidad de obtener información sobre estas en busca de una mejor experiencia de los alumnos en la minifabrica. Esto es conseguido con la ayuda de una interfaz interactiva y fácil de usar para evitar que el alumno tenga problemas adicionales.

Figura 2 - Ejemplo de reconocimiento de imagen exitosa

5. Conclusiones

Esta aplicación es un gran paso en la integración de tecnologías innovadoras como puede ser la Realidad Aumentada en el entorno de la minifabrica. Debido a su carácter genérico, esta aplicación se puede incorporar en todo tipo de laboratorios. Un buen ejemplo sería el laboratorio de máquinas eléctricas, en la que los alumnos también están expuestos a máquinas complejas y esta aplicación podría solventar algunos problemas. De cara al futuro, esta aplicación podría diseñarse para permitir la visualización información dinámica, es decir información sobre el estado de la maquina en tiempo real. Esto integraría de forma completa esta aplicación en el uso cotidiano de las máquinas del laboratorio.

6. Referencias

[1] Deloitte.com, "¿Qué es la Industria 4.0?"
https://www2.deloitte.com/es/es/pages/manufacturing/articles/que-es-la-industria-4.0.html
Última visita: 24/08/2021

AUGMENTED REALITY APPLICATION FOR THE MAINTENANCE OF INDUSTRIAL PLANTS

Author: Escolano Hovine, Guillermo.
Supervisor: Rodríguez Mondéjar, José Antonio.
Collaborating Entity: ICAI – Universidad Pontificia Comillas

ABSTRACT

In this project, an application has been created that allows obtaining information about the mini-factory machines through image recognition, helping potential operators to solve problems they may encounter.

Keywords: Augmented Reality, Industry 4.0

1. Introduction

The fourth industrial revolution, also known as Industry 4.0, is marked by the appearance of new technologies, such as robotics, Big Data, Artificial Intelligence (AI), Machine Learning (ML) or the Internet of Things (IoT) among many others. This revolution seeks to combine these technologies with traditional production methods and transform them to obtain more efficient connected systems [1].
Another of the pillars of this change is expected to be Augmented Reality (AR), a technology that allows visualizing the real world by adding information that improves user interaction with the outside world in real time.

2. Project summary

The aim of this project is to make an application based on augmented reality that allows its users (technicians, workers, students, etc ...) a more efficient performance of their work, providing, for example, information on the current state of the machine. It also seeks to make a tool that allows familiarization with the machinery to workers who face it for the first time giving basic information about it. This application will allow to recognize images that the user chooses, and that will serve as identifier of the machine of interest. By recognizing the image, the application will give the user the option of obtaining more information about the machine. A system will also be created that allows creating templates for each machine that the user wishes, containing the useful information of each one, implementing the possibility for the user to access their gallery and choose the recognition image or doing it with the device's camera.

3. Model/System/Tool description

The application is based on the Unity video game engine that allows you to create applications easily. The application will be divided into three sections. The first will be a window that will allow you to see the information of the machine once it has been recognized.

The second window is where the machine is detected with the help of the Vuforia software that allows to recognize targets that are stored in a n online database. Once the target is recognized, the application gives the option of obtaining more information about the machine associated with that target (redirects to window 1).

Finally, the third window allows you to create a template for a new machine. For this, it is necessary to create a new target, that is, an image that will be linked to a single machine. The option to create this image using the device's photo camera or accessing the files saved on the device has been developed. Once the target has been chosen, the user enters the name of the machine and its information. When creating the new template, the application communicates with the Vuforia database and adds the new information. The template has been designed in such a way that, if an error occurs at any stage of its creation, an error message can be viewed by the user, describing the type of error involved, substantially improving the accessibility of the application. The architecture of the application can be seen in figure 3.

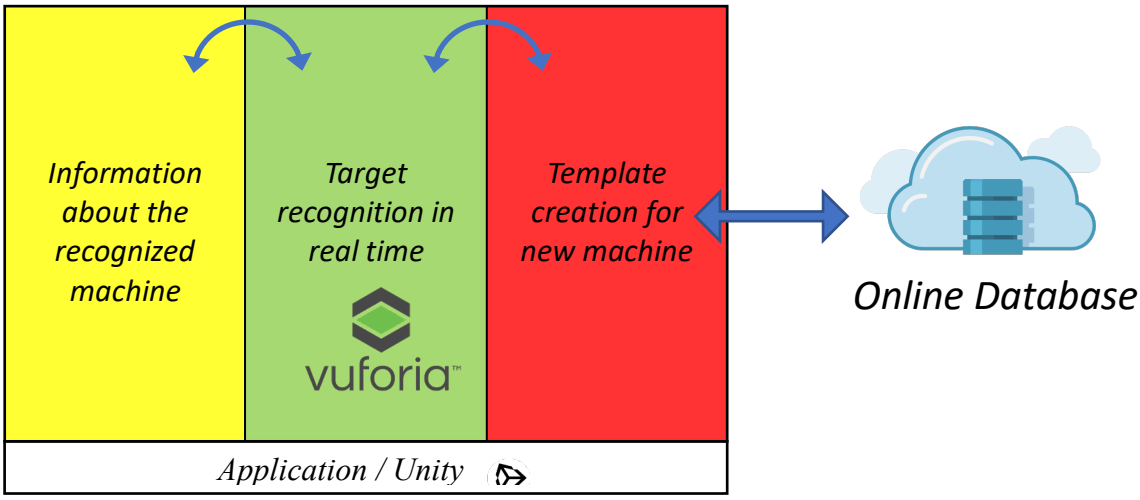


Figure 3 - Architecture of the application

4. Results

In this project, we have been able to create an application that meets the desired specifications, that is, it allows the identification of images selected by the user to identify a particular machine and enables the possibility of obtaining information about it, in search of a better user experience by the students in the mini factory. This is achieved with the help of an interactive and easy-to-use interface to avoid students having additional problems.

Figure 4 - Example of a successful image detection

5. Conclusions

This application represents a great step in the integration of innovative technologies such as Augmented Reality in the mini-factory environment. Due to its generic nature, this application can be incorporated in all types of laboratories. A good example would be the electrical machines laboratory, in which students are also exposed to complex machines and this application could solve some problems. With a view to the future, this application could be designed to allow the display of dynamic information, that is, information about the status of the machine in real time. This would fully integrate this application in the daily use of laboratory machines.

6. References

[1] Deloitte.com, "¿Qué es la Industria 4.0?"
https://www2.deloitte.com/es/es/pages/manufacturing/articles/que-es-la-industria-4.0.html
Last visit: 24/08/2021

Índice de la memoria

Índice de figuras

Índice de tablas

Capítulo 1. INTRODUCCIÓN

En los dos últimos siglos, hemos sido testigos de grandes avances en los ámbitos industriales y tecnológicos. En especial, ha habido momentos en los que nuevas tecnologías han sido el origen de un proceso de transformación económica, social y tecnológica. Como se puede ver en la figura 1, se han producido tres revoluciones industriales que cambiaron nuestra la forma en la que vivimos desde entonces, mejorando nuestra calidad de vida y abriéndonos puertas nunca antes exploradas.

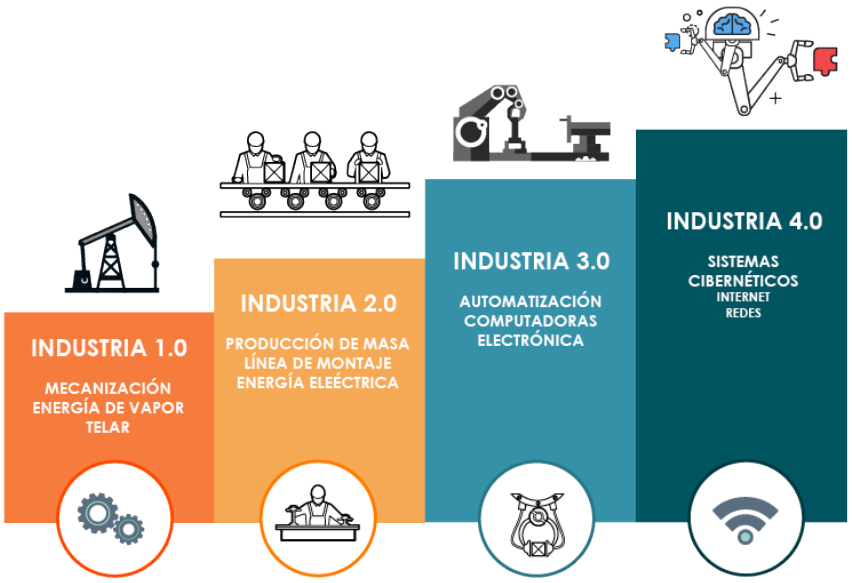


Figura 1 - Diferentes revoluciones industriales [1]

Se dice que en el presente estamos experimentando la cuarta revolución industrial. La cuarta revolución industrial, también conocida como Industria 4.0 esta marcada por la aparición de nuevas tecnologías, como pueden ser la robótica, el Big Data, la Inteligencia Artificial (AI), el Machine Learning (ML) o el Internet of Things (IoT) entre muchas otras. Esta revolución busca combinar estas tecnologías con los métodos de producción tradicionales y transformarlos para obtener sistemas conectados más eficientes.

Se prevé que otro de los pilares de este cambio sea la Realidad Aumentada (AR), una tecnología que permite visualizar el mundo real añadiendo información que mejora la interacción del usuario con el exterior en tiempo real.

Esta tecnología esta siendo usada en el mundo de la industria en diversas tareas de mantenimiento y de control de calidad. Este proyecto busca usar esta tecnología para ayudar a alumnos que usen la minifabrica a identificar problemas y fallos que pueden ocurrir al operar con las máquinas presentes en la minifabrica. También busca que los alumnos puedan familiarizarse con estas máquinas teniendo acceso a información sobre cada una de estas. Todo esto se consigue de forma interactiva a través de una aplicación que despertará el interés de los alumnos en conocer más la minifabrica. Para ello se creará una aplicación que permite de forma automática identificar una maquina y dar su información. También se dará la posibilidad de añadir máquinas si el profesor lo desea (por ejemplo, una maquina nueva se ha instalado en la minifabrica).

Capítulo 2. DESCRIPCIÓN DE LAS TECNOLOGÍAS

Para este proyecto se usarán varios recursos, en particular, distintos softwares relacionados con la realidad aumentada. El primero es Unity, que será la base del proyecto. Unity es un motor grafico que permite la creación de un entorno interactivo tanto en 2D como 3D. Dentro de Unity se usara una extensión llamada Vuforia que permite implementar herramientas relacionadas con la realidad aumentada y detección de objetos (visón artificial).

2.1 VUFORIA

Vuforia es un kit de desarrollo de software (SDK). Un SDK permite generalmente a desarrolladores de software crear aplicaciones a partir de las herramientas que esta ofrece, ahorrándoles tiempo y recursos. En este caso, Vuforia sirve para crear aplicaciones basadas en la realidad aumentada (AR) de manera fácil y flexible. [2]

La arquitectura de vuforia es compleja pero se puede simplicar en tres partes importantes:

1 – Las bases de datos: Existen tres tipos de bases de datos en Vuforia:

- una base de datos online: Permite tener una base de datos que esta conectada a internet y por lo tanto es capaz de buscar millones de imágenes de reconocimiento, además ofrece la opción de utilizar los servicios API del los cuales se habla en el apartado 2.3, para borrar, añadir, cambiar todas las partes de tu base de datos a través de conexiones HTTP. Esto facilita muchísimo el trabajo y permite automatizar el proceso de creación de imágenes de reconocimiento.

- una base de datos interna: Se crea en el propio dispositivo al descargarte la aplicación. Tiene ventajas y inconvenientes respecto a la base de datos online. La ventaja es que, al estar en la memoria del dispositivo, las respuestas son más inmediatas, es decir el reconocimiento de imágenes es más rápido. Sin embargo, esto supone que tienes un numero limitado de almacenamiento disponible (el del dispositivo) y para proyectos grandes de empresas eso no seria viable.

- una base de datos para VuMarks: VuMarks son la nueva tecnología de Vuforia que se aproximan mucho a un código QR, como se puede ver en la figura 2. Son muy útiles y pueden servir para identificar diferentes partes de un mismo objeto. Se pueden diseñar con facilidad, pero hace falta tener licencia de Vuforia para poder utilizarlos de forma fluida por lo que en este proyecto no son utilizados.

Figura 2 - Ejemplo de VuMark [3]

2 – Los "targets" (objetivos de reconocimiento): Existen diferentes tipos de objetivos que Vuforia puede detectar y rastrear:

- Imágenes
- Objetivo múltiple
- Objetivo cilíndrico
- Objeto
- VuMark

Al crear un objetivo de detección es necesario elegir el tipo que más le corresponda. En el caso de este proyecto todos los objetivos serán imágenes. Como se puede ver en la figura 3, Vuforia nos da a seleccionar los diferentes tipos de objetivos al añadir uno a nuestra base de datos. Es importante tener claro que tipo de objetivos vas a reconocer, dado que esto afecta mucho en la rapidez de detección. Esto es debido a que un objeto 3D es mucho más complejo que una simple imagen 2D, y por lo tanto tarda más en detectarse.

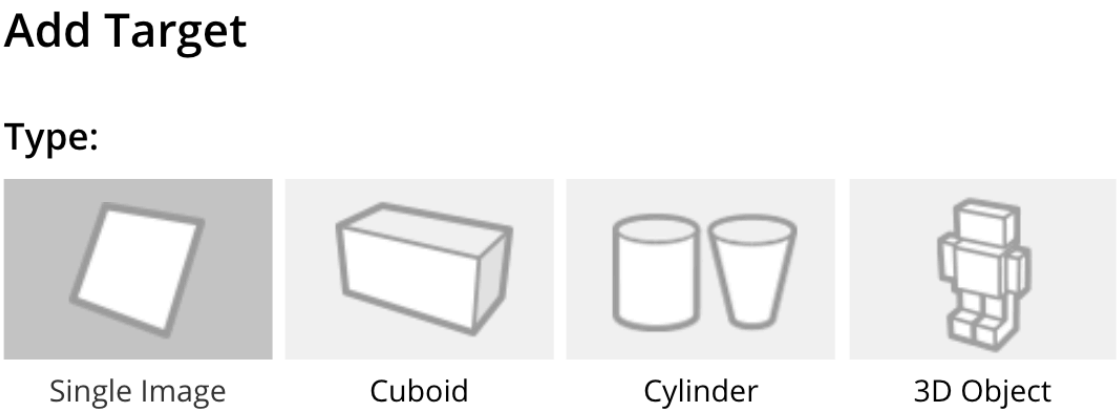


Figura 3 - Menu de creación de objetivo de reconocimiento [4]

Vuforia proporciona a estos objetivos una nota (rating) sobre cinco, como se puede ver en la figura 4, que indica la facilidad con la que la aplicación podrá reconocer ese objetivo. Es importante tener en cuenta esta nota a la hora de crear imágenes objetivo ya que la fluidez de la aplicación esta muy correlacionada con la velocidad de detección, y si esta parte funciona de forma mediocre, la experiencia del usuario se vería muy deteriorada.

Figura 4 - Ejemplo de imágenes con su nota de rating [5]

3 – La aplicación: Por ultimo, para poder usar Vuforia hace falta tener una aplicación en la que poder usar esta tecnología. Para ello se puede usar el motor de videojuegos Unity, que es el caso de este proyecto. Como se puede ver en la figura 5, se puede ver como es la SDK de Vuforia y como esta conectada con la aplicación del movíl. La información procedente de la cámara del móvil es procesada y el rastreador (Tracker en la imagen) al tener acceso a la base de datos con los objetivos (tanto online como las internas del dispositivo) es capaz de comparar ambos y hacer una detección en caso de que sean semejantes. Una vez realizada la detección, la aplicación puede añadir información a la pantalla y "renderizarla" (generar la imagen) para que el usuario la pueda ver, que es el principio de la realidad aumentada, es decir añadir información virtual a un mundo real.

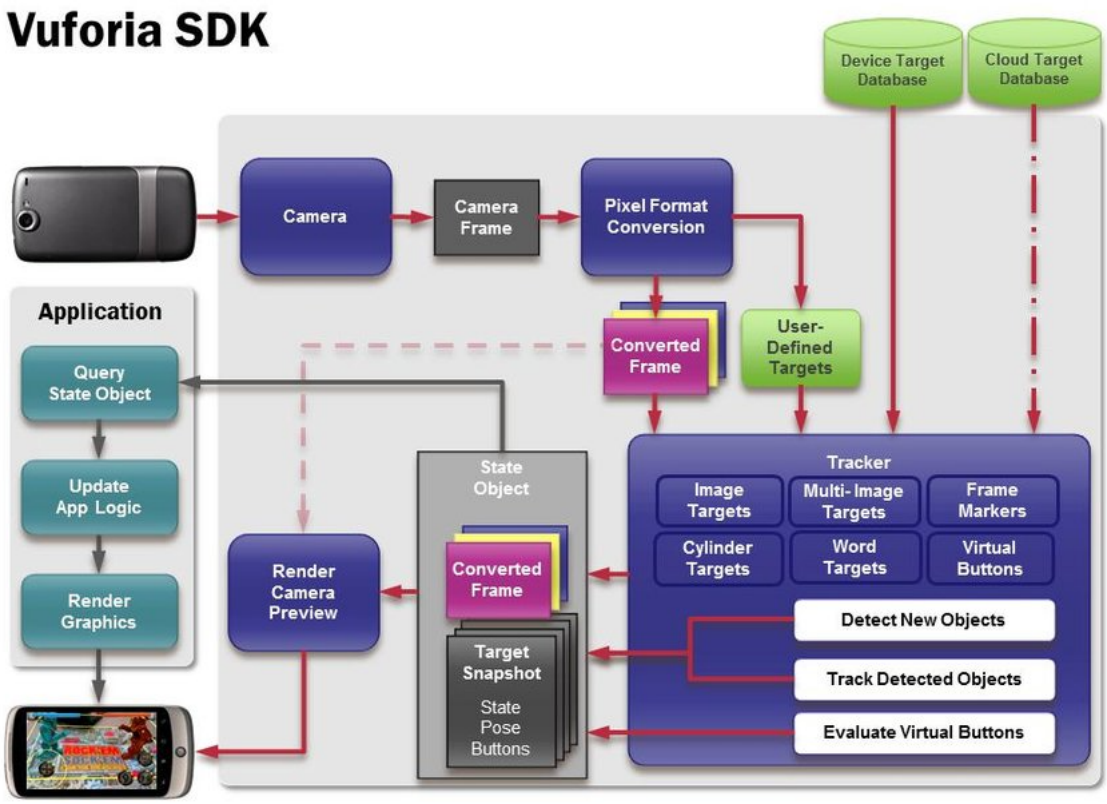


Figura 5 - SDK de Vuforia [2]

2.2 UNITY

Unity es un motor de videojuegos multiplataforma, lo que significa que un equipo de desarrolladores puede desarrollar un videojuego para diferentes consolas y dispositivos a partir de la misma plataforma. Esto de una gran versatilidad a Unity, diferenciándola de muchos de sus competidores al poder reducir significativamente los costes de traspaso de una base de código entre diferentes plataformas. Se puede ver en la figura 6 todos los dispositivos compatibles con Unity.

Figura 6 - Dispositivos compatibles con Unity [6]

Esta plataforma esta basada en el lenguaje de programación C# (Csharp) que esta entre los diez idiomas informáticos más hablados en el mundo. Esta es una de las razones por las que Unity es accesible a muchas personas y muchas otras pueden acceder fácilmente, al poder utilizar sus conocimientos de este lenguaje en otras aplicaciones [7].

La principal utilidad de Unity es proporcionar a desarrolladores una plataforma que les permita enfocar su atención en la experiencia del juego y no en su construcción. Unity cuenta con muchísimas herramientas ya integradas que permiten ahorrar tiempo en el desarrollo de cualquier proyecto. Esto permite que las personas utilizando esta tecnología se puedan concentrar en el diseño de la aplicación y como va a quedar el producto final, y perder menos tiempo creando la infraestructura de la aplicación. Otra forma de Unity de ayudar a sus usuarios es a través de su editor visual que permite a través de componentes conectar diferentes partes de la aplicación de forma muy intuitiva, reduciendo así de forma

significativa el código que se debería escribir si no se usase esta plataforma. Los componentes son una parte esencial de Unity, son equivalentes a tuercas y tornillos de los objetos, tienen diferentes propiedades y variables que se pueden cambiar fácilmente y se pueden añadir cuantos uno desee.

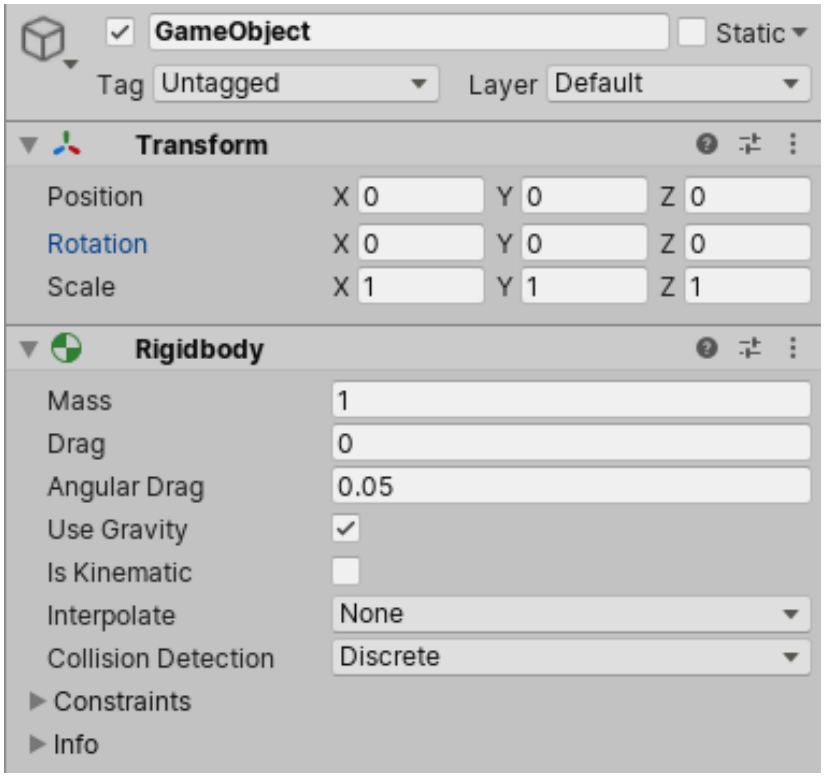


Figura 7 - Objeto de Unity con sus diferentes componentes [8]

Otra gran ventaja de usar Unity es su gran comunidad de usuarios. Esto significa que hay una multitud de personas activas en foros dedicados que se ayudan para resolver dudas, y comentan diferentes técnicas que usan a la hora de desarrollar sus proyectos. Esto es muy útil para personas que recién acaban de empezar y necesitan ayuda. Junto con la baja curva de aprendizaje, esto permite a Unity ser la plataforma ideal para desarrolladores novicios que se quieren adentrar en el mundo de la creación de aplicaciones [9].

2.3 API

El término API es una abreviatura de Application Programming Interfaces, que en español significa interfaz de programación de aplicaciones. Permite, a través de definiciones y

protocolos, la comunicación entre dos aplicaciones usando un conjunto de reglas [10]. Una buena analogía es la de un restaurante en la que un consumidor se comunica con la cocina a través del camarero que hace de intermediario entre ambos. Así pues, en el caso de la comunicación entre aplicaciones, el intermediario es una API, que se encarga de enviar mensajes entre todos los que se conecten a ella.

El uso de las API es muy extendido y se pueden encontrar en todas las empresas más grandes del mundo. Por ejemplo, twitter tiene una API que permite acceder a tweets o hacer publicaciones de forma automática. Como se puede ver en la figura 8, se puede observar como la API hace en efecto de intermediario entre una base de datos y diferentes programas que pueden acceder a ella con o sin acreditación según el tipo de API que se trate.

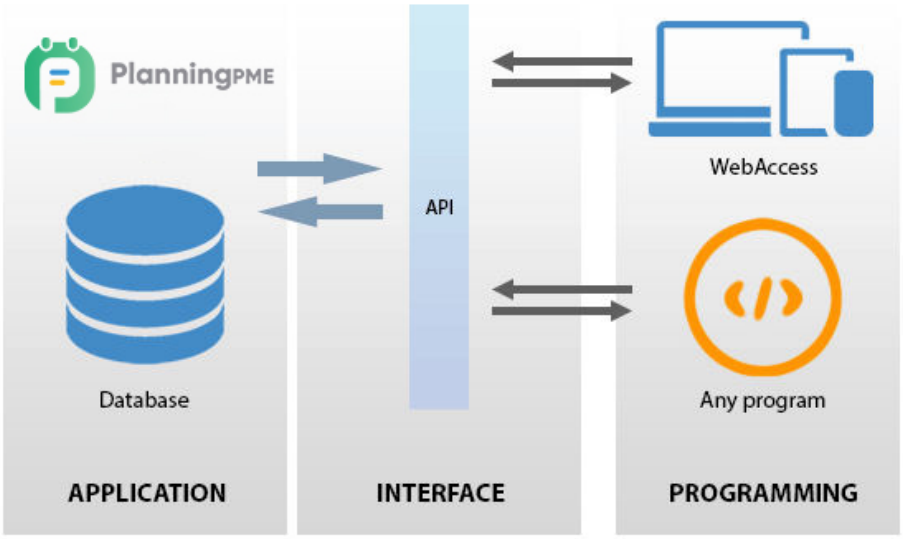


Figura 8 - Diagrama de una API [11]

2.4 APK

APK (Android Application Package) es un archivo que contiene datos comprimidos que permiten ejecutar una aplicación en el sistema operativo de Android. Estos archivos son similares a los archivos ZIP o RAR para otros sistemas operativos [12]. Hay que tener en cuenta que, en el sistema operativo de Android, estos APK pueden ser descargados por los usuarios desde fuentes externas que no sean del Play Store, al opuesto de iOS (sistema operativo de Apple), en la que no te puedes descargar ninguna aplicación fuera del Apple

Store. A la hora de querer crear una aplicación es importante tener este aspecto en cuenta ya que el sistema de iOS esta cerrado a fuentes externas y hace falta una licencia para poder publicar una aplicación, que, en el caso de querer hacer una aplicación de uso recreativo, no compensa su adquisición.

Capítulo 3. ESTADO DE LA CUESTIÓN

Los expertos en este sector prevén que de 2020 a 2023 el gasto mundial en realidad aumentada se multiplique por 10, pasando de un gasto de 18 600 millones de dólares a 160 000 millones de dólares, con una tasa de crecimiento anual del 78,3%, con Estados Unidos y China comprendiendo el 75% de ese gasto en 2023 [13]. En efecto, la realidad aumentada ya esta permeando muchos sectores, incluido el de la industria. Esta tecnología lleva varios años usándose en la industria de la aviación para proporcionar información crucial a los pilotos, pero también muchas tiendas de venta minorista usan la realidad aumentada para que el consumidor pueda probar como quedan ciertos productos, como zapatos o muebles, sin haberlos comprado y sin tener que desplazarse [14]. Un ejemplo sería el uso de esta tecnología en el sector de los videojuegos en la que muchas empresas del sector han optado por hacer juegos que incluyan tanto elementos virtuales como elementos reales. Así fue el caso de Pokemon GO, juego que fue lanzado en el año 2016, y obtuvo 232 millones de usuarios en su primer año de lanzamiento con unos ingresos de casi un billón de dólares, lo que da a entender que la aceptación por el publico de esta tecnología es fuerte y que esta demandada [15]. Se puede ver en la figura 9 como este juego de móvil utiliza la tecnología de realidad aumentada para simular una experiencia más realística al usuario.

Figura 9 - Fotografía del videojuego móvil Pokemo GO [14]

Otro ejemplo que da a ver la importancia que se le esta dando a esta tecnología es la creación de google lens por Google que es una aplicación móvil que analiza el entorno con Machine Learning (ML) para detectar e identificar objetos a nuestros alrededores. Esta aplicación tiene muchos usos como por ejemplo el traducir texto en tiempo real, es decir, el usuario puede apuntar a un documento en un idioma extranjero, y esta aplicación superpondrá por encima el texto traducido al idioma deseado por el usuario. Otros usos pueden ser la obtención de información de libros, edificios o animales con tan solo apuntarlos [16]. Como se puede apreciar, los casos prácticos para la realidad aumentada son realmente numerosos y es una tecnología que tiene mucho recorrido por delante todavía.

En ultimó lugar, cabe destacar el auge de la Realidad Aumentada, más concretamente, en el sector de la industria. En este sector también se ha notado un aumento en el uso de la tecnología de Realidad Aumentada, y da lugar a una conexión entre máquinas y datos que permite a los operarios trabajar de forma más eficaz y fluida, aumentando así su productividad. Muchos procesos y operaciones pueden efectuarse de manera más eficiente usando esta tecnología, desde la formación de técnicos a labores de mantenimiento de equipos y máquinas, o dando un soporte online a particulares por expertos que ya no se ven con la obligación de desplazarse.

La Realidad Aumentada también tiene usos en otros campos de la industria como puede ser el de la logística permitiendo dar indicaciones visuales y la programación de tareas. También se puede usar en el sector del marketing al permitir visualizar modelos 3D de piezas o sistemas complejo. Podemos pensar en la visualización de una obra antes de esta haber sido empezada para convencer a los inversores de como será el resultado final. Varias empresas industriales han decidido dar ya el paso y adoptar la Realidad Aumentada a su cadena de producción. Una de estas empresas es la empresa española Unilever que distribuye bienes de consumo de alta rotación. Esta empresa usa la Realidad Aumentada para la localización dentro de la línea de producción a través de visión artificial y el guiado de alta precisión [17] como se puede ver en la figura 10.

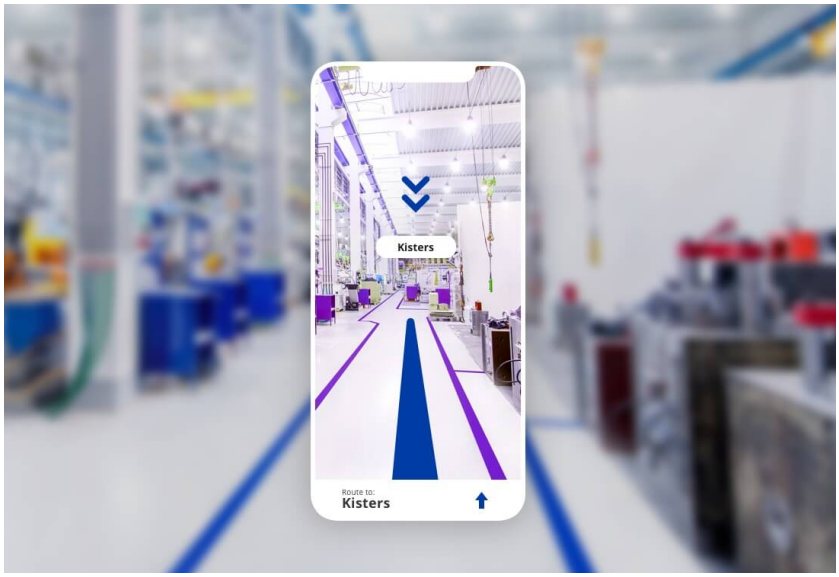


Figura 10 - Uso de la Realidad Aumentada con Unilever [17]

Capítulo 4. DEFINICIÓN DEL TRABAJO

4.1 JUSTIFICACIÓN

Este trabajo tiene como permite la introducción de tecnologías como la realidad aumentada a un entorno como puede ser el de la minifabrica de ICAI. Esto permitirá a alumnos de trabajar de manera más eficiente y sentirse más integrados en los laboratorios en los que estén operando con las máquinas de la minifabrica.

Es una oportunidad para todos ellos para tener un primer contacto con tecnologías de la nueva revolución industrial que está empezando y que tengan la oportunidad de verse involucrados. Muchas veces los laboratorios pueden considerarse como embudos para alumnos que se quedan atascados, y esta aplicación les permite salirse de la rutina y acudir a una ayuda que no tenga que depender siempre del profesor, que ha su vez ve su carga de trabajo disminuida. Este proyecto también hace ejercicio de prevención, al evitar que alumnos usen de forma incorrecta las máquinas y las dañen en el proceso. Por lo tanto, el gasto en mantenimiento y renovación de equipamiento se vería reducido.

4.2 OBJETIVOS

4.2.1 OBJETIVOS DE PRESTACIÓN

El objetivo de este proyecto es hacer una aplicación basada en la realidad aumentada que permita a sus usuarios (técnicos, operarios, estudiantes, etc…) una realización más eficiente de su trabajo aportando por ejemplo información sobre el estado de la actual de la maquina. Se busca hacer también una herramienta que permita la familiarización con la maquinaria a trabajadores que se enfrentan a ella por primera vez dando información básica sobre esta. Esta aplicación permitirá reconocer imágenes que el usuario elija, y que servirá como identificador de la maquina de interés. Al reconocer la imagen, la aplicación dará al usuario la opción de obtener más información sobre la maquina. Además, esta aplicación permitirá al usuario acceder a información de todas las máquinas que desee al añadir la posibilidad de crear perfiles de la máquinas a partir de una misma plantilla. Esta plantilla será capaz de

acceder a la galería de fotos del dispositivo y la cámara de fotos y también comunicarse con la base de datos de Vuforia a partir de la API.

4.2.2 OBJETIVOS DE DESARROLLO SOSTENIBLE

Uno de los objetivos del desarrollo Sostenible es el de desarrollar infraestructuras fiables, sostenibles, resilientes y de calidad (ODS 9). Este proyecto responde perfectamente a este propósito al permitir desarrollar una herramienta que, basándose en la realidad aumentada, permite a técnicos de diferentes industrias familiarizarse y adaptarse a las máquinas con las que trabajan, al aportar información especifica y en tiempo real de estas mismas. Esto se traduce en una reducción de averías y fallos en las máquinas que pueden ser reparadas antes y con mejor precisión [18].

La mejora de la fiabilidad de las máquinas supone también una reducción de consumo de energía y recursos, que es otro de los objetivos de desarrollo sostenible (ODS 7). Las máquinas, más eficientes y con una vida útil más larga, no tendrán que ser reemplazadas con tanta frecuencia, disminuyendo así gastos en infraestructura de las empresas y por lo tanto de materias primas [19].

4.3 METODOLOGÍA Y PLANIFICACIÓN

Estos son los siguientes pasos que se siguieron durante este proyecto:

1. Familiarización con las herramientas que se van a usar (Unity, Vuforia).
2. Diseño de la interfaz de la aplicación.
3. Sistema de detección de imágenes
4. Conexión de la aplicación con la base de datos en la nube de Vuforia
5. Sistema de creación de una nueva maquina (elegir imagen del dispositivo) a partir de una plantilla
6. Opción de subir una imagen con la cámara del dispositivo

El cronograma relacionado con los pasos anteriores es el siguiente:

Tabla 1 - Cronograma del Proyecto

Pasos	Enero	Febrero	Marzo	Abril	Mayo	Junio	Julio
1							
2							
3							
4							
5							
6							

4.4 ESTIMACIÓN ECONÓMICA

La razón por la que se decidieron usar varias de las plataformas escogidas en el proyecto es debido a que ofrecen muchas funcionalidades de forma gratuita. El desarrollo de este proyecto se ha realizado por lo tanto sin necesidad de ningún gasto. Pero en el caso de que se quisiese extender esta aplicación a los dispositivos del alumnado, sería necesario que pagar varias licencias. Por ejemplo, Vuforia tiene un limite de reconocimientos en su versión gratuita y tarde o temprano, la aplicación dejaría de funcionar debido a que los alumnos no podrían reconocer la imagen objetivo. En el caso de querer poner la aplicación en las diferentes plataformas de descarga (sea Apple o Android), también seria necesario pagar licencias en Vuforia, y en caso de Apple sería necesario pagar la licencia del Apple Store ya que se necesita para poder subir aplicaciones. Para poner ciertos números a lo dicho anteriormente:

- Licencia Apple Store [20]: **100$ anuales**, permite subir aplicaciones a la Apple Store.
- Licencia básica Vuforia [21]: **42$ mensuales**, permite la publicación para uso comercial, reconocimiento de objetivos básicos.
- Licencia básica + nube Vuforia: **100$ mensuales**, añade a la licencia básica la opción de tener 10.000 reconocimientos en la nube mensuales (en vez de 1000)
- Licencia Plus Unity [22]: **400$ anuales**, ofrece más servicios que la versión gratuita.
- Licencia Pro Unity: **1.800$ anules**, ofrece aun más servicios.

Por lo tanto, el coste mínimo que costaría sacar la aplicación de forma oficial en todas las plataformas sería de alrededor de **600 euros**.

Capítulo 5. SISTEMA/MODELO DESARROLLADO

5.1 DISEÑO

5.1.1 DISPOSICIÓN DE LAS VENTANAS

Esta aplicación esta dividida en tres secciones, cada una de ellas con un rol muy distinto, pero, aun así, todas están conectadas como se vera en el apartado 5.1.3. Se decidió dividir en tres ventanas la aplicación ya que correspondía con los tres principales objetivos del proyecto:

- Poder crear una plantilla para diferentes máquinas
- Poder detectar e identificar las máquinas en cuestión
- Poder acceder a los datos respectivos de cada maquina

Para ello se ha diseñado cada ventana con el objetivo de que el usuario tenga la mejor experiencia y que la aplicación sea fácil de usar e intuitiva. A continuación, serán presentadas las disposiciones de cada ventana con sus diferentes componentes.

5.1.1.1 Ventana de creación de plantillas para máquinas

Figura 11 - Ventana de creación de plantillas para máquinas

5.1.1.2 Ventana de detección de máquinas

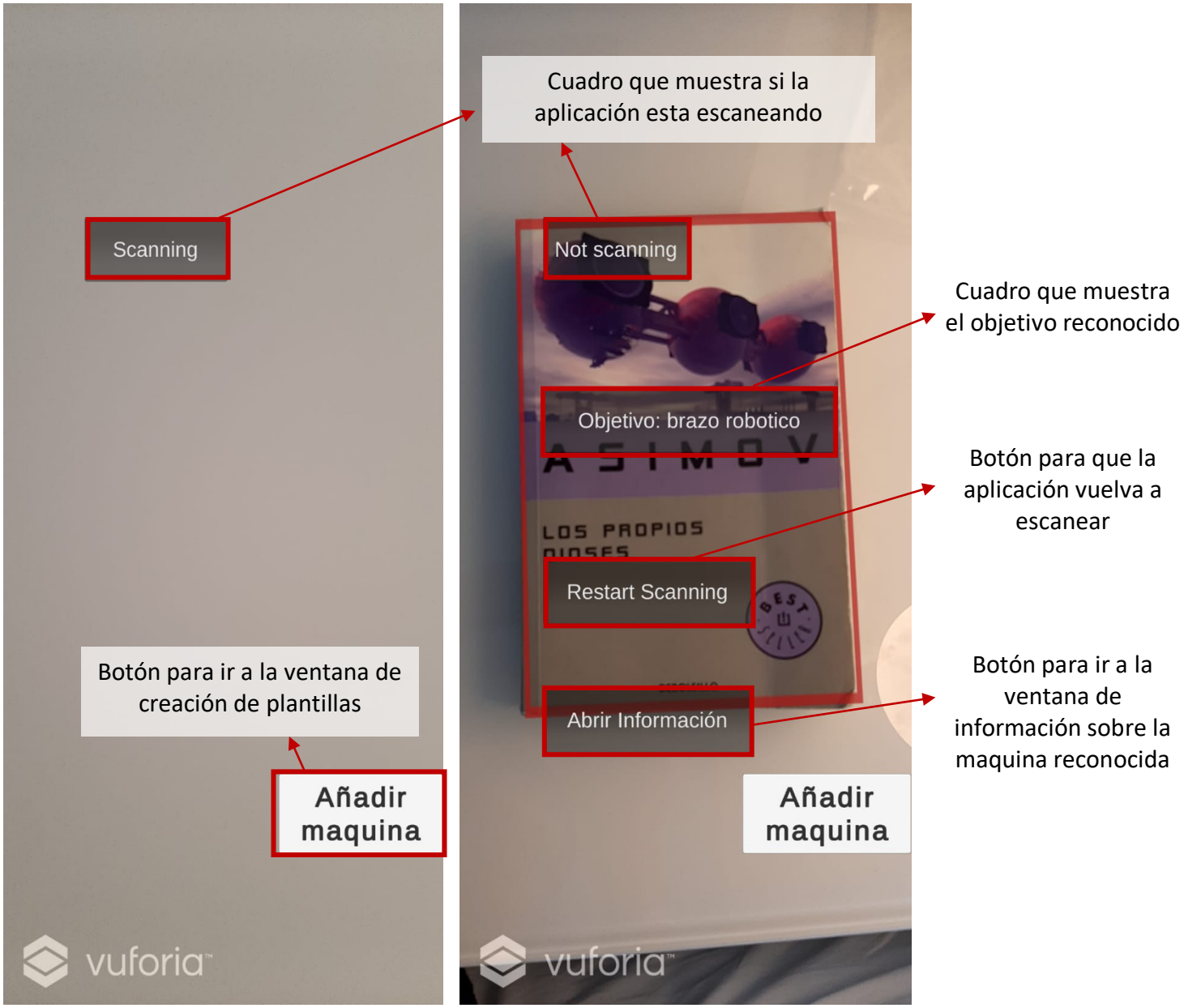


Figura 12 - Ventana de detección de máquinas

5.1.1.3 Ventana de visualización de información sobre la maquina

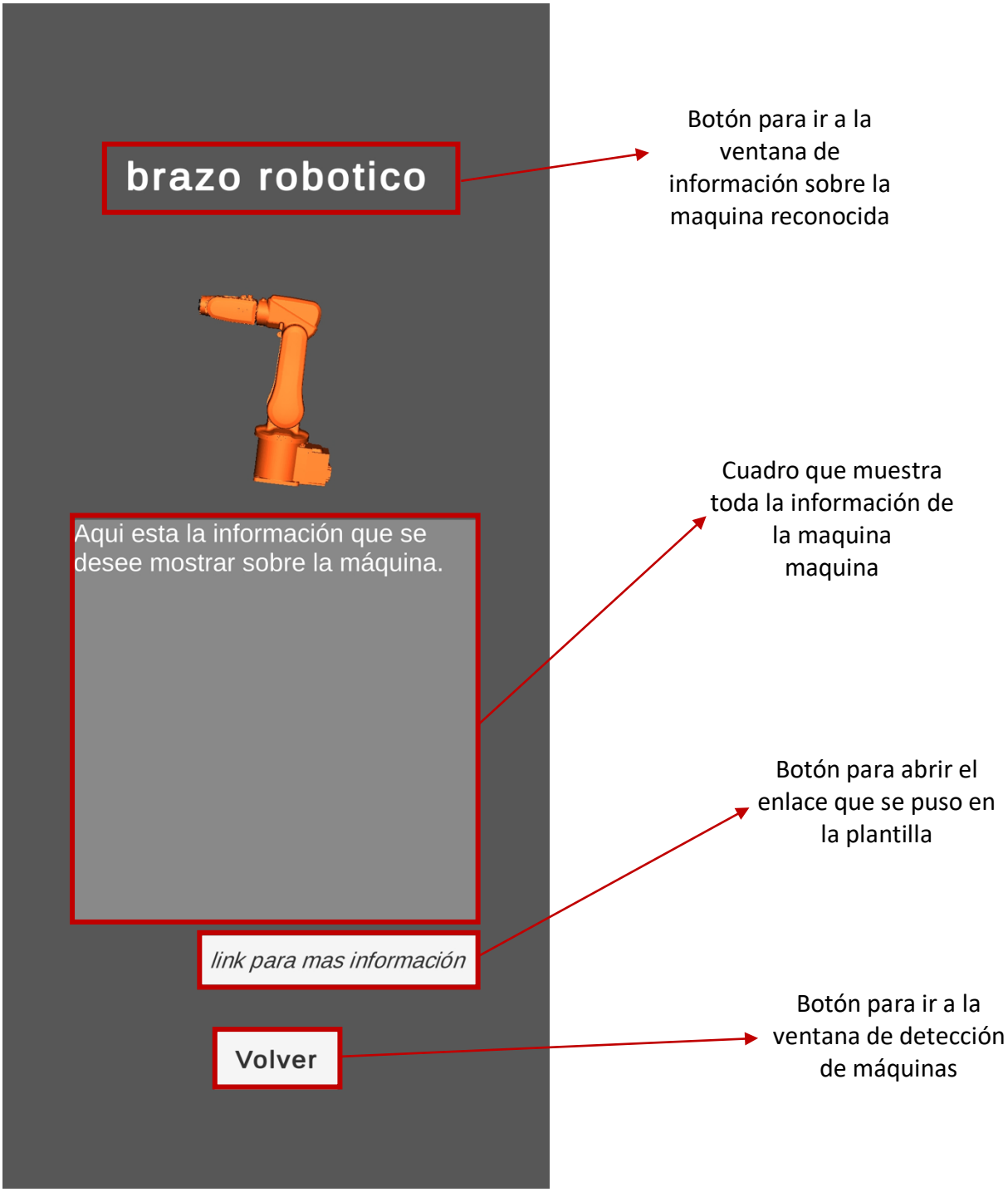


Figura 13 - Ventana de visualización de información sobre máquinas

5.1.2 SIMULADOR DE DISPOSITIVO MÓVIL

Al empezar este proyecto, el diseño se realizaba sobre todo para ordenador para ir probando las diferentes disposiciones de los componentes en la pantalla y comprobar el funcionamiento general de la aplicación. Para ello se usaba la pantalla de simulación de juego integrada en Unity desde el ordenador. Esto permitió un arranque en el proceso de diseño suave y sencillo, pero solo permitió avanzar en el diseño de la aplicación hasta cierto punto.

Cuando el proyecto empezó a tener la suficiente substancia como para empezar a querer ver el resultado en un dispositivo móvil o Tablet, se apercibió que la disposición de los diferentes componentes no en la pantalla no era la misma que la del ordenador. Por lo tanto, muchas de las funcionalidades de la aplicación (botones, "input fields", etc..) no eran accesibles ya que muchas de estas se salían de la región visible de la pantalla. Para solucionar este problema y permitir un diseño mucho más fiable y directo, se hizo uso de una extensión de Unity llamada "Device Simulator". Esta extensión permite visualizar en tiempo real como va a quedar la disposición de los componentes en todo tipo de dispositivos (móviles de muchas marcas y tablets). Se puede apreciar en la figura 14 encuadrado en rojo todo lo comentado anteriormente y el porque de la utilidad de esta extensión.

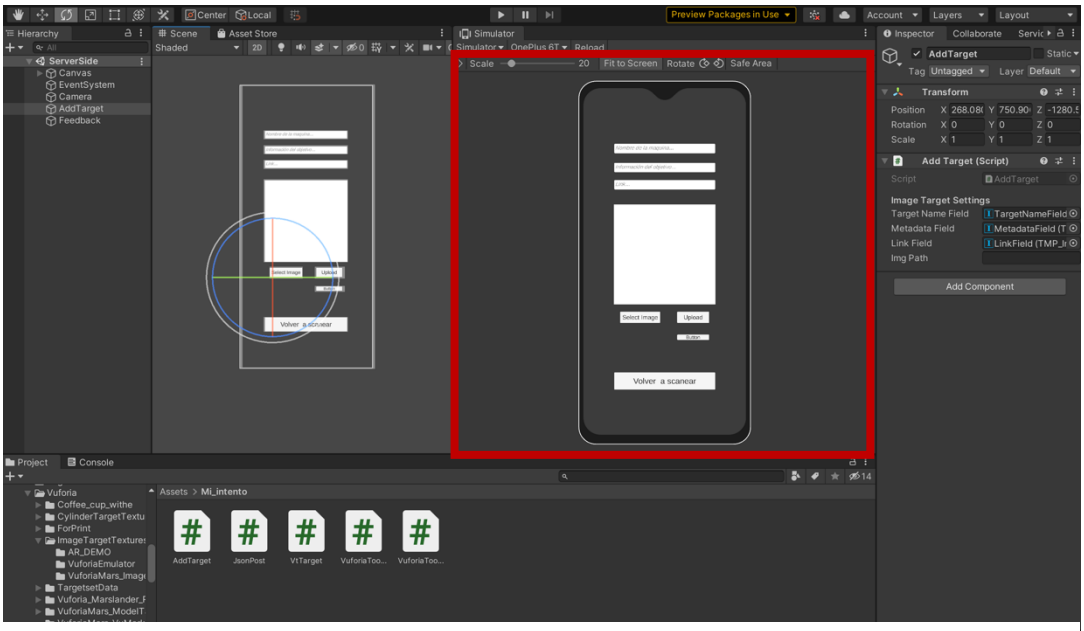


Figura 14 - Ventana de Unity con la extensión de simulación de móvil activada

5.1.3 INTERCONEXIÓN ENTRE LAS VENTANAS

Esta aplicación tiene ventanas (o escenas) y para ello es necesario tener un mecanismo que permita moverse de manera fluida entre estas ventanas según lo que el usuario requiera en ese momento. Para ello, se ha decido implementar una seria de botón, los cuales al pulsarlos permiten irse a otra ventana. Para ello estos botones disponen de un script en los cuales se llama a una función de la librería de Unity cuya función es esa. Esta función se llama "SceneManager.LoadScene()" y una implementación sencilla puede ser la siguiente:

```
public class VolverScanear : MonoBehaviour
{
    public void Menu()
    {
        SceneManager.LoadScene("Scanning");
    }
}
```

Como se puede apreciar esta función acepta un parámetro. Este parámetro puede ser tanto un numero entero (int) que este dentro del rango del numero de escenas que tenga la aplicación o una cadena de caracteres (string) cuyo valor sea el nombre de alguna de las escenas existentes. La forma de buscar según una string es más lenta y es recomendable usar los números enteros, pero el tener esta aplicación nada más que tres escenas, este tiempo de retardo es casi imperceptible y no se pierden las referencias a las escenas si se mueven de orden de jerarquía (cambiaria su valor de numero entero, ya que van todas numeradas).
Se puede observar también que esta función esta dentro de otra función llamada "Menu()", esta función es simplemente la función que será llamada al ser pulsado el botón, por eso tiene el prefijo de publica, para que pueda ser llamada por un componente externo. Unity tiene incorporado una funcionalidad que permite conectar componentes entre si arrastrando un componente a otro. Los botones en particular vienen con una función llamada "onClick()" que se ejecuta cada vez que se presiona el botón. Todos los scripts que se añadan a esta función serán ejecutados al ser presionado el botón. Un ejemplo de esto esta en la figura 15, en la que se puede ver como la función "VolverScanear.Menu()" (es decir la función Menu() del script VolverScanear) será llamada al ser presionado el botón.

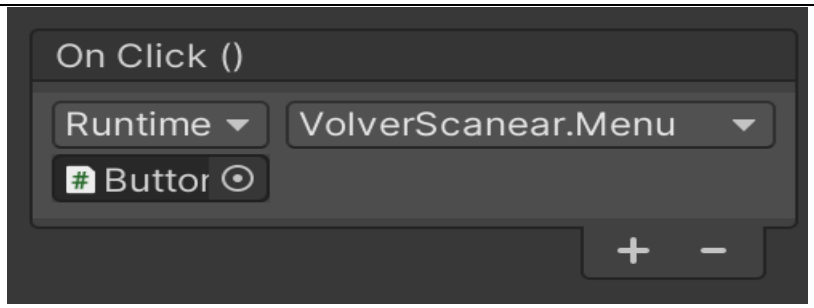


Figura 15 - Función Onclick() de un Botón

5.2 IMPLEMENTACIÓN

5.2.1 ESCENA DE CREACIÓN DE PLANTILLA

Esta escena tiene la función de crear una ficha para una nueva maquina a partir de la plantilla dispuesta en la escena. Esta plantilla consiste en el nombre de la maquina, la información que el usuario crea conveniente de esta, un enlace que pueda ser de interés, una imagen objetivo que luego será usada para identificar la maquina, esta puede ser elegida a través de la galería del dispositivo o haciendo una foto con la cámara del móvil. En ultimó lugar, esta escena se encarga de comunicar con la API de Vuforia para generar esa nueva ficha en la base datos online.
Dicho esto, hay también diversas funcionalidades en cara a mejorar la experiencia del usuario, y se irán comentando todas ellas a continuación.

Para poder escribir el nombre, contenido y enlace con el teclado del móvil se hace uso de "inputfields", componentes que vienen de serie con Unity y que permiten acceder al teclado del dispositivo y escribir con él. Adicionalmente, para mejorar la experiencia del usuario e informarle del propósito de cada inputfield, se hace uso también de "placeholders" que son un texto ficticio que aparecen cuando no hay nada escrito y describe al usuario lo que se espera que escriba en ese espacio.

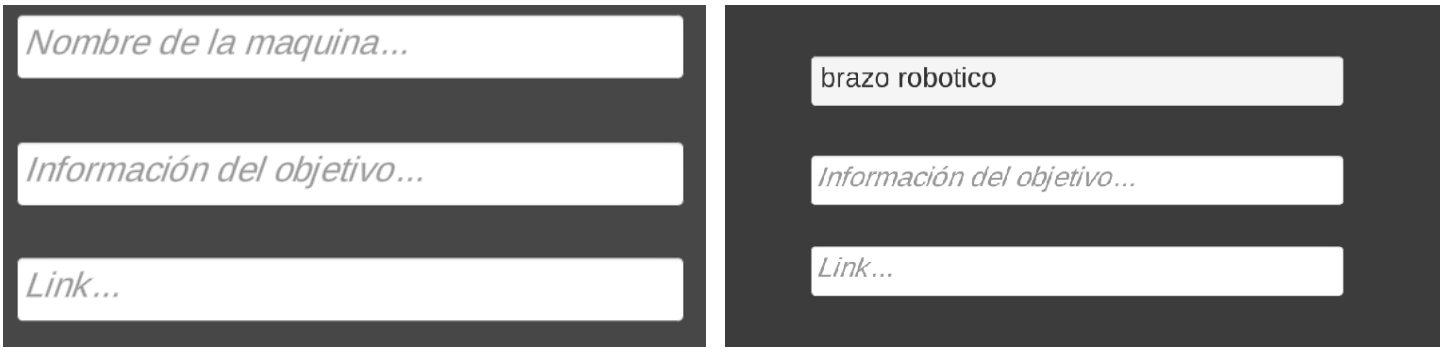


Figura 16 - Ejemplo de uso de placeholders

Para acceder a los placeholders es necesario desagrupar el inputfield, desagrupar el Text Area y acceder al componente llamado placeholder en la que habrá que cambiar el valor del texto al que se quiera ver. Se puede ver esta jerarquía en la figura 17.

Figura 17 - Como acceder al placeholder de un inputfield

Una vez la información introducida por el usuario, el próximo paso consiste en elegir la imagen objetivo. Se han diseñado dos formas de seleccionar la imagen objetivo deseada, una es a través de la galería del dispositivo y otra haciendo una foto con la cámara del teléfono. La forma sugerida de hacerlo es seleccionando una foto de la galería, ya que la foto de una cámara puede ser borrosa y es necesaria una muy buena imagen para que el reconocimiento se haga sin problemas. Para la selección de la foto a través de la galería del dispositivo existe un botón en el que pone “select image”. Al presionar este botón, el dispositivo te permite abrir la galería y te da la opción de seleccionar una imagen. El acceso a la galería del dispositivo esta gestionado por una extensión externa a Unity llamada NativeGalley que con una simple llamada con una función permite acceder a la galería. Una vez elegida la imagen, se guarda en el script la ubicación del archivo seleccionado.

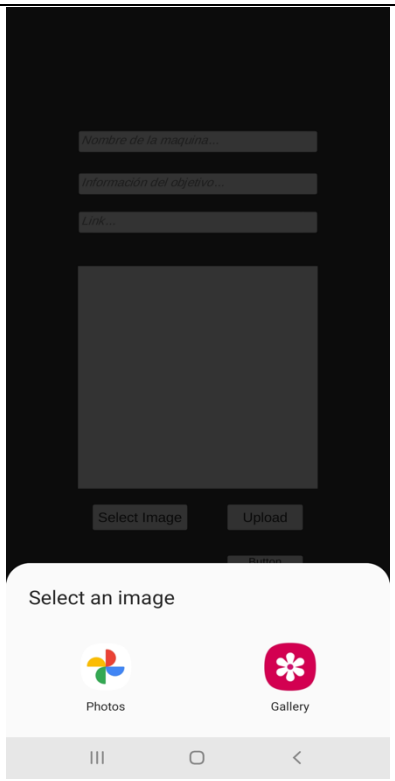


Figura 18 - Ejemplo de acceso a galería del dispositivo

La segunda forma de obtener la imagen es a través de la cámara de fotos del dispositivo, para ello se usa una extensión externa que permite acceder a la cámara de fotos. Al hacer la foto se guarda la textura de la foto almacenada en una ubicación del móvil.

Finalmente, el proceso de creación de la plantilla termina al presionarse el botón de upload. Al presionarse este botón, lo primero que se verifica es que todos los datos necesarios y obligatorios para rellenar la ficha de la maquina existen y son validos. Para ello se conectan entre si todos los componentes que se usan para rellenar información, ya se los inputfiels o la imagen objetivo, y se conectan con el botón upload. Este verifica que el valor de todos los parámetros son validos, es decir que existe un nombre de la maquina, y que la imagen objetivo ha sido elegida. Estos son los dos únicos parámetros obligatorios a la hora de crear la ficha en la base de datos de Vuforia. En el caso de que uno de estos no estuviese completo, un mensaje de alerta seria enviado, y un mensaje aparecería en la pantalla avisando de que

es lo que falla. Como se puede ver en la figura 19, existe un mensaje para la falta de nombre y uno para la falta de imagen.

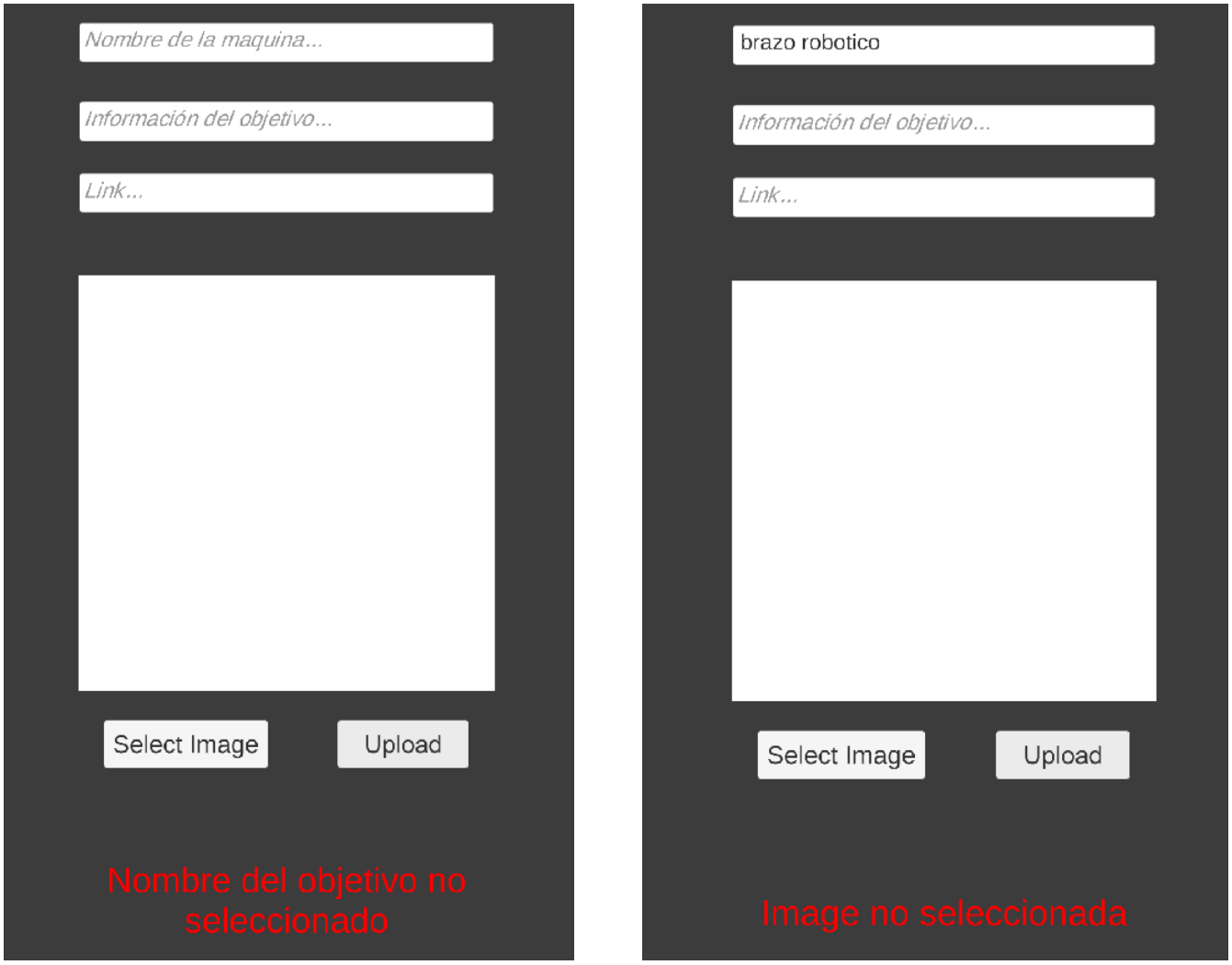


Figura 19 - Mensajes de error debido a información introducida incompleta

Si todos los datos son validos, la aplicación procederá a comunicarse a través de la API de Vuforia con la base de datos online. Esto puede dar lugar a errores. Todos estos errores son visibles en la figura 20, como por ejemplo el que ya exista el nombre introducido, o que la imagen no sea valida debido al tipo o al tamaño. Toda esta realimentación de información sobre el éxito de la operación esta recogida en otros mensajes visibles para el usuario, para

que sepa en todo momento, si la operación se ha realizado de forma correcta. En la figura 20, se puede ver todos los resultados posibles que puede tener esta operación.

result_code	HTTP status code	Description
Success	OK (200)	Transaction succeeded
TargetCreated	Created (201)	Target created (target POST response)
AuthenticationFailure	Authentication failure (401)	Signature authentication failed
RequestTimeTooSkewed	Forbidden (403)	Request timestamp outside allowed range
TargetNameExist	Forbidden (403)	The corresponding target name already exists (target POST/PUT response)
RequestQuotaReached	Forbidden (403)	The maximum number of API calls for this database has been reached.
TargetStatusProcessing	Forbidden (403)	The target is in the processing state and cannot be updated.
TargetStatusNotSuccess	Forbidden (403)	The request could not be completed because the target is not in the success state.
TargetQuotaReached	Forbidden (403)	The maximum number of targets for this database has been reached.
ProjectSuspended	Forbidden (403)	The request could not be completed because this database has been suspended.
ProjectInactive	Forbidden (403)	The request could not be completed because this database is inactive.
ProjectHasNoApiAccess	Forbidden (403)	The request could not be completed because this database is not allowed to make API requests.
UnknownTarget	Not Found (404)	The specified target ID does not exist (target PUT/GET/DELETE response)
BadImage	Unprocessable Entity (422)	Image corrupted or format not supported (target POST/PUT response)
ImageTooLarge	Unprocessable Entity (422)	Target metadata size exceeds maximum limit (target POST/PUT response)
MetadataTooLarge	Unprocessable Entity (422)	Image size exceeds maximum limit (target POST/PUT response)
DateRangeError	Unprocessable Entity (422)	Start date is after the end date
Fail	Unprocessable Entity (422)	The request was invalid and could not be processed. Check the request headers and fields.
Fail	Internal Server Error (500)	The server encountered an internal error; please retry the request

Figura 20 - Posibles resultados de la operación de subir la ficha de la maquina a la base de datos [23]

Y en la siguiente figura podemos observar unos ejemplos de como funciona esa realimentación en la aplicación:

Figura 21 - Mensajes obtenidos de la operación de subir la ficha de la maquina a la base de datos

5.2.2 ESCENA DE DETECCIÓN DE MÁQUINAS

Esta escena tiene la función de reconocer las imágenes objetivo que se encuentran en la base de datos online. Para ello es necesario conectarse a esta base datos y tener un componente de objetivo y de cámara de realidad virtual presenta en la jerarquía de la escena. Como se puede ver en la figura 22, en la jerarquía esta un objeto llamado ARCamera y Image Target. El objeto Image Target es un objeto que según como se coloque respecto a la cámara AR se vera el objetivo reconocido de cierta forma (en caso de añadir tu mismo figuras 3D a la escena). Esto hay que tenerlo en cuenta ya que si la cámara esta mal posicionada, no se podrá hacer correctamente la detección del objeto.

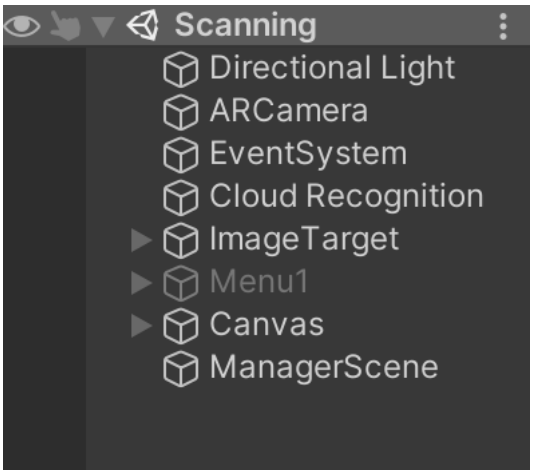


Figura 22 - Jerarquía de la escena de detección

Se puede ver también como en la imagen anterior hay un objeto llamado Cloud Recognition. Este objeto se encarga de conectarse con la base de datos online de Vuforia. Tiene en uno de sus componentes, las llaves de acceso necesarias para poder conectarse con la correcta base de datos y que no pueda haber intrusos en ella en caso de que la información que se maneje sea sensitiva. Como se puede ver en la figura 23, hay unos parámetros llamados "Access Key" y "Secret Key", que son las llaves únicas que permitan comunicarnos con nuestra base de datos online.

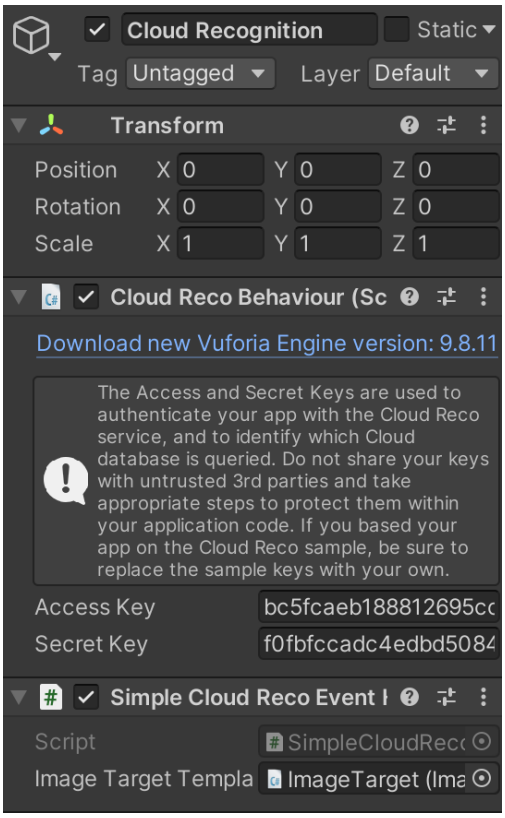


Figura 23 - Componente de Cloud Recognition

Una vez se ha identificado la imagen objetivo se abre una serie de menús que muestran el nombre del objetivo detectado. Se sabe que objetivo ha sido identificado porque Vuforia envía un mensaje al identificar un objetivo que contiene un paquete de información en formato JSON con todo el contenido que nosotros pusimos al crear la ficha de la maquina, por lo tanto, podemos acceder al nombre de dicho objetivo y mostrarlo. Si queremos obtener más información sobre dicho objetivo, simplemente basta con presionar el botón “abrir más información” que aparece cuando un objetivo ha sido detectado.

5.2.3 ESCENA DE VISUALIZACIÓN DE INFORMACIÓN

Esta escena permite visualizar la información que fue introducida por el usuario al crear la ficha. Es una escena muy simple compuesta de pocas partes. Una de ellas es el nombre del objetivo (el de la maquina) seguido por un cuadro que permite albergar un texto grande que pueda contener toda la información necesaria sobre la maquina. Ya sea sobre como mantenerla, que especificaciones de uso tiene o lo que el usuario desee conveniente.
Tenemos la opción de abrir un enlace. Este enlace redirige a una pagina que el usuario considero aportaba información útil y adicional sobre la maquina. Para poder abrir un enlace en Unity se puede usar la función "Application.OpenUrl()" que viene ya integrada. Para evitar problemas de errores de compilación, ponemos esta función dentro de una condición que solo permite ejecutarla si la url existe. El siguiente código es un ejemplo de como se puede escribir lo antes mencionado:

```
public class AbrirLink : MonoBehaviour
{
    public TemplateManager template;
    private string url;

    public void Open()
    {
        url = template.link;
        if(url != null && url != "")
        {
            Application.OpenURL(url);
        }
    }
}
```

El resultado es que se abre una pestaña que permite elegir si queremos redirigirnos a la pagina del enlace como se puede ver la figura 24.

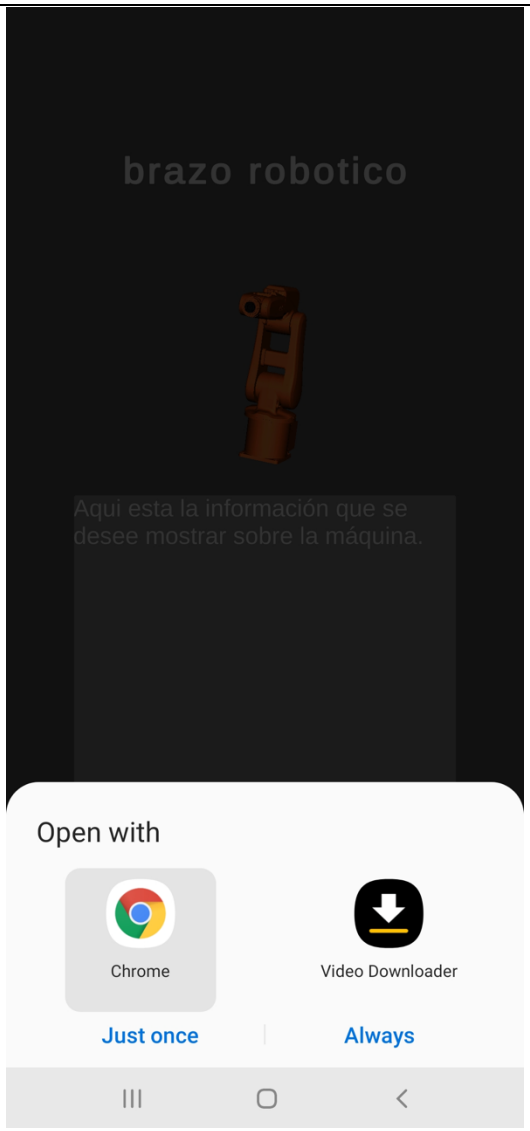


Figura 24 - Ejemplo de acceso al enlace

5.2.4 CREACIÓN DE LA APLICACIÓN

Una vez estén el diseño y la implementación de la aplicación terminada, es hora de crear el archivo que contenga toda la información de la aplicación. Para ello, vamos a usar una funcionalidad de Unity que permite crear archivos comprimidos de nuestras aplicaciones para diferentes tipos de plataformas. Si nos dirigimos a "File / Build Settings" nos entraremos con la ventana que sale en la figura 25. En ella podemos observar arriba las

diferentes escenas de nuestra aplicación con su jerarquía a la derecha (numero entero correspondiente). Es importante tener esto en cuenta al ser la escena de jerarquía 0, la escena con la que la aplicación se abrirá. Más abajo a la izquierda, se pueden ver las diferentes plataformas en las que Unity puede crear la aplicación. En el caso de este trabajo, solo se construirá la aplicación para la plataforma de Android. Al darle a "build", se empieza a generar el archivo comprimido, que tendrá un formato ".apk", comentado anteriormente en el apartado 2.4, que es el formato de todas las aplicaciones de Android. Por lo tanto, el proceso de crear la aplicación es sencillo debido a que Unity te ofrece las herramientas posibles para ello. Si no fuese el caso habría que indagar sobre como se genera un archivo apk y hacerlo manualmente.

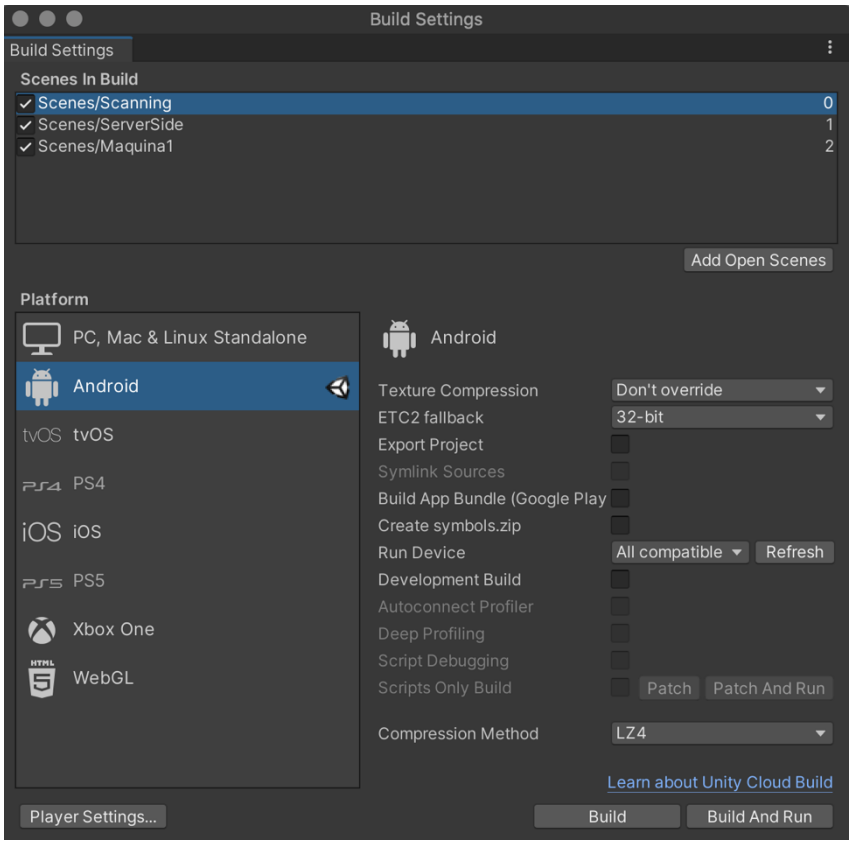


Figura 25 - Menu de configuración de la Build

Capítulo 6. ANÁLISIS DE RESULTADOS

El objetivo de este proyecto era conseguir una aplicación que permitiese el acceso de forma útil y sencilla a información sobre las máquinas de la minifabrica. Los resultados obtenidos son buenos y responden a las expectativas. A la hora de usar la aplicación, añadir una maquina es un proceso simple y la aplicación esta dotada de elementos que permiten al usuario saber si se ha producido un error durante este proceso. Al usar el reconocimiento de objetivos, la aplicación te proporciona información sobre el estado del reconocimiento, ya sea mientras escanea o ha obtenido un resultado. Esto permite que el usuario este en control en todo momento y no añada dificultades ya que esta aplicación esta diseñada principalmente para ayudar y se quiere evitar ante todo que cause problemas adicionales.

Uno de los mayores logros es la versatilidad de la aplicación, que ha sido diseñada de forma que se pueda adaptar a cualquier maquina, al ser el usuario que introduce toda la información que el considera útil en la plantilla. El único inconveniente es que el tamaño máximo de la información que se puede añadir es de 2mb que para la mayoría de las aplicaciones es más que suficiente, pero en algunos casos se puede quedar corto. Para ello se ha implementado una forma de vincular un enlace que puede redirigir a una pagina creada por el usuario que contenga toda la información necesaria.

Cabe destacar también que el reconocimiento del objetivo es bueno si este tiene un buen valor en rating (nota dada por Vuforia que identifica lo bien que se hará el reconocimiento posterior), por lo que es importante tener buenos objetivos para que el reconocimiento se produzca de manera fluida. Si el objetivo que se desea tiene mala nota, habrá que comprometerse a cambiarlo ya que permanecer con el mismo podría suponer un mal funcionamiento de la aplicación.

Capítulo 7. CONCLUSIONES Y TRABAJOS FUTUROS

En este proyecto se ha conseguido crear una aplicación funcional que permite a alumnos de ICAI interactuar de forma interactiva con la minifabrica de ICAI. Para ello se ha diseñado una interfaz que permite crear fichas técnicas independientes para cada maquina y una forma rápida y eficaz de acceder a ellas a través de reconocimiento visual. Para crear la ficha técnica y elegir la imagen objetivo, se ha implementado funciones que permiten el acceso a la cámara del dispositivo y la galería de fotos.

Para trabajos futuros se pueden aportar algunas mejoras y extensiones a la aplicación. En primer lugar, se puede buscar la compatibilidad con todos los sistemas operativos ya la aplicación actual tiene extensiones y funcionalidades solo disponibles en los dispositivos Android. Aunque como se comento el apartado 4.4 sobre la estimación económica, sacar una aplicación al sistema iOS es de pago y se puede valorar su utilidad.
En segundo lugar, se puede buscar una forma de conectar la aplicación a los brazos robóticos de la minifabrica para poder acceder a información sobre su posición en el espacio en ese mismo instante, es decir, poder acceder a información dinámica del sistema. Esto podría abrir muchos caminos sobre lo que se puede hacer con la aplicación. Tener información en tiempo real del estado de la maquina permitiría a los alumnos tener una mejor comprensión de lo que esta ocurriendo y, en caso de fallo o de una respuesta inesperada, poder encontrar una solución de forma más directa.

Capítulo 8. BIBLIOGRAFÍA

[1] Onac.org.co, "Infraestructura de la Calidad 4.0". https://onac.org.co/blog-de-ulrich/infraestructura-de-la-calidad-4-0/
Última visita: 24/08/2021

[2] Desarollolibre.net, "Realidad Aumentada con Vuforia". Enero 2014
https://www.desarrollolibre.net/blog/android/realidad-aumentada-con-vuforia
Última visita: 24/08/2021

[3] Vuforia.com, "VuMark". https://library.vuforia.com/features/objects/vumark.html
Última visita: 24/08/2021

[4] Vuforia.com, "AddTarget".
https://developer.vuforia.com/targetmanager/project/targets?projectId=f6e3077dc57e4faea2b47d8a003f8109&av=false
Última visita: 24/08/2021

[5] Vuforia.com, "Best Practices for Designing and Developing Image-Based Targets".
https://library.vuforia.com/features/images/image-targets/best-practices-for-designing-and-developing-image-based-targets.html
Última visita: 24/08/2021

[6] Unity.com, "What platforms are supported by Unity". https://support.unity.com/hc/en-us/articles/206336795-What-platforms-are-supported-by-Unity-
Última visita: 24/08/2021

[7] Classpert.com, "Top 15: los lenguajes de programación más usados en 2021". Mayo 2021. https://es.classpert.com/blog/lenguajes-de-programacion-mas-usados
Última visita: 24/08/2021

[8] Unity3d.com, "Usando Components".
https://docs.unity3d.com/es/2019.4/Manual/UsingComponents.html
Última visita: 24/08/2021

[9] Platzi.com, "Qué es Unity - Todo sobre el popular motor de videojuegos". Enero 2021.
https://platzi.com/blog/que-es-unity-motor-videojuegos/
Última visita: 24/08/2021

[10] Xatakandroid.com, "API: qué es y para qué sirve". Agosto 2019.
https://www.xataka.com/basics/api-que-sirve
Última visita: 24/08/2021

[11] Hevodata.com, "Data Architecture Explained: A Comprehensive Guide". Junio 2021.
https://hevodata.com/learn/everything-about-data-architecture-explained/#c1
Última visita: 24/08/2021

[12] Xatakandroid.com, "Qué es un APK de Android, cómo se instala y diferencias con las apps normales". Julio 2019. https://www.xatakandroid.com/aplicaciones-android/que-apk-android-como-se-instala-diferencias-apps-normales
Última visita: 24/08/2021

[13] Silicon.es. "De aquí a 2023, el gasto en realidad aumentada y virtual se multiplicará por 10", Junio 2019. https://www.silicon.es/de-aqui-a-2023-el-gasto-en-realidad-aumentada-y-virtual-se-multiplicara-por- 10-2398135
Última visita: 24/08/2021

[14] BBVA.com. "Los siete usos de la realidad aumentada que ya están aquí", https://www.bbva.com/es/siete-usos-realidad-aumentada-ya-estan-aqui/
Última visita: 24/08/2021

[15] Businessofapps.com, "Pokémon Go Revenue and Usage Statistics (2021)". Mayo 2021.
https://www.businessofapps.com/data/pokemon-go-statistics/
Última visita: 24/08/2021

[16] Xataka.com, "Google Lens: qué es, cómo instalarlo y todo lo que puedes hacer con ella". Junio 2019. https://www.xataka.com/basics/google-lens-que-como-instalarlo-todo-que-puedes-hacer-ella
Última visita: 24/08/2021

[17] Neosentec.com, "Realidad aumentada en la industria 4.0".
https://www.neosentec.com/realidad-aumentada-en-la-industria-4-0/
Última visita: 24/08/2021

[18] UN.org. "Objetivo 9: Construir infraestructuras resilientes, promover la industrialización sostenible y fomentar la innovación",
https://www.un.org/sustainabledevelopment/es/infrastructure/
Última visita: 24/08/2021

[19] UN.org. "Objetivo 7: Garantizar el acceso a una energía asequible, segura, sostenible y moderna",
https://www.un.org/sustainabledevelopment/es/energy/
Última visita: 24/08/2021

[20] Developer.apple.com, "How much does Apple charge for free apps hosted on the app store?". https://developer.apple.com/forums/thread/76587
Última visita: 24/08/2021

[21] Ptc.com, "Precios de Vuforia Engine: el plan perfecto para crear su visión". https://www.ptc.com/es/products/vuforia/vuforia-engine/pricing
Última visita: 24/08/2021

[22] Store.unity.com, "Elige el plan adecuado para ti". https://store.unity.com/es/compare-plans
Última visita: 24/08/2021

[23] Vuforia.com, "How To Use the Vuforia Web Services API".
https://library.vuforia.com/articles/Solution/How-To-Use-the-Vuforia-Web-Services-API.html
Última visita: 24/08/2021

ANEXO I

En el siguiente anexo se encuentran los códigos más importante s que componen este proyecto:

```
public class AddTarget : MonoBehaviour
{
    [Header("Image Target Settings")]
    public TMP_InputField targetNameField;
    public TMP_InputField metadataField;
    private string targetName;
    readonly float width = 1.0f;
    readonly bool active = true;
    private string metaData;
    public string imgPath = null;
    public event Action OnTargetNameNotFound;
    public event Action OnTargetPathNotFound;
    public event Action<string> OnUpload;

    public void CallPostTarget()
    {
        VuforiaTools uploadRequest =
gameObject.AddComponent<VuforiaTools>();
        TemplateMaquina template = new TemplateMaquina();
        template.nombre = targetNameField.text;
        template.contenido = metadataField.text;
        targetName = template.nombre;
        metaData = JsonUtility.ToJson(template);
        string metaDataUpload = null;
        if (metaData != null)
        {
            if (metaData == "")
            {
                metaData = null;
            }
            else
            {
                var plainTextBytes =
System.Text.Encoding.UTF8.GetBytes(metaData);
                metaDataUpload =
System.Convert.ToBase64String(plainTextBytes);
            }
        }

        if (targetName != null)
        {
            if (targetName == "")
            {
                targetName = null;
            }
        }
        if (imgPath != null)
        {
```

```
            if (imgPath == "")
            {
                imgPath = null;
            }
        }
        string returnString = "false";
        if (targetName != null && imgPath != null)
        {
            returnString = uploadRequest.UploadImageTarget(targetName,
width, imgPath, active, metaDataUpload);
        }
        else if (targetName == null)
        {
            OnTargetNameNotFound();
            return;
        }
        else if (imgPath == null)
        {
            OnTargetPathNotFound();
            return;
        }

        VuforiaAccountSummary message =
JsonUtility.FromJson<VuforiaAccountSummary>(returnString);
        OnUpload(message.result_code);
    }
}
```

```
public class Feedback : MonoBehaviour
{
    private AddTarget target;
    public TMP_Text alert;

    void Start()
    {
        target = FindObjectOfType<AddTarget>();
        target.OnTargetNameNotFound += AlertNameNotFound;
        target.OnTargetPathNotFound += AlertPathNotFound;
        target.OnUpload += AlertUpload;
    }


    void AlertNameNotFound()
    {
        alert.text = "Name of the target not set";
    }

    void AlertPathNotFound()
    {
        alert.text = "Image path not yet selected";
    }

    void AlertUpload(string m)
    {
        alert.text = m;
    }
}
```

```
public class ManagerScene : MonoBehaviour
{
    public static ManagerScene scanning;
    public string metadata = "";
    public CloudRecoBehaviour target;
    public string nombre = "";

    private void Awake()
    {
        if (scanning == null)
        {
            scanning = this;
            DontDestroyOnLoad(gameObject);
        }
        else
        {
            Destroy(gameObject);
        }
    }
    private void Start()
    {
        SimpleCloudRecoEventHandler parent =
target.GetComponentInParent<SimpleCloudRecoEventHandler>();
        parent.GiveMetadata += UpdateMetadata;
    }

    private void UpdateMetadata(TemplateMaquina aux)
    {
        metadata = aux.contenido;
        nombre = aux.nombre;
    }

}
```

```
public class SimpleCloudRecoEventHandler : MonoBehaviour
{
    private CloudRecoBehaviour mCloudRecoBehaviour;
    private bool mIsScanning = false;
    private string mTargetMetadata = "";
    public ImageTargetBehaviour ImageTargetTemplate;
    public event Action<TemplateMaquina> GiveMetadata;

    // Register cloud reco callbacks
    void Awake()
    {
        mCloudRecoBehaviour = GetComponent<CloudRecoBehaviour>();

mCloudRecoBehaviour.RegisterOnInitializedEventHandler(OnInitialized);
        mCloudRecoBehaviour.RegisterOnInitErrorEventHandler(OnInitError);

mCloudRecoBehaviour.RegisterOnUpdateErrorEventHandler(OnUpdateError);

mCloudRecoBehaviour.RegisterOnStateChangedEventHandler(OnStateChanged);

mCloudRecoBehaviour.RegisterOnNewSearchResultEventHandler(OnNewSearchResu
lt);
    }
    //Unregister cloud reco callbacks when the handler is destroyed
    void OnDestroy()
    {

mCloudRecoBehaviour.UnregisterOnInitializedEventHandler(OnInitialized);

mCloudRecoBehaviour.UnregisterOnInitErrorEventHandler(OnInitError);

mCloudRecoBehaviour.UnregisterOnUpdateErrorEventHandler(OnUpdateError);

mCloudRecoBehaviour.UnregisterOnStateChangedEventHandler(OnStateChanged);

mCloudRecoBehaviour.UnregisterOnNewSearchResultEventHandler(OnNewSearchRe
sult);
    }
    public void OnInitialized(TargetFinder targetFinder)
    {
        Debug.Log("Cloud Reco initialized");
    }
    public void OnInitError(TargetFinder.InitState initError)
    {
        Debug.Log("Cloud Reco init error " + initError.ToString());
    }
    public void OnUpdateError(TargetFinder.UpdateState updateError)
    {
        Debug.Log("Cloud Reco update error " + updateError.ToString());
    }

    public void OnStateChanged(bool scanning)
    {
        mIsScanning = scanning;
        if (scanning)
        {
```

```
            // clear all known trackables
            var tracker =
TrackerManager.Instance.GetTracker<ObjectTracker>();

tracker.GetTargetFinder<ImageTargetFinder>().ClearTrackables(false);
        }
    }

    // Here we handle a cloud target recognition event
    public void OnNewSearchResult(TargetFinder.TargetSearchResult
targetSearchResult)
    {
        TargetFinder.CloudRecoSearchResult cloudRecoSearchResult =
            (TargetFinder.CloudRecoSearchResult)targetSearchResult;
        // do something with the target metadata
        mTargetMetadata = cloudRecoSearchResult.MetaData;
        // stop the target finder (i.e. stop scanning the cloud)
        mCloudRecoBehaviour.CloudRecoEnabled = false;
        if (ImageTargetTemplate)
        {
            // enable the new result with the same ImageTargetBehaviour:
            ObjectTracker tracker =
TrackerManager.Instance.GetTracker<ObjectTracker>();

tracker.GetTargetFinder<ImageTargetFinder>().EnableTracking(targetSearchR
esult, ImageTargetTemplate.gameObject);
        }
        if (mTargetMetadata != "")
        {
            TemplateMaquina template =
JsonUtility.FromJson<TemplateMaquina>(mTargetMetadata);
            GiveMetadata(template);
        }

    }

    void OnGUI()
    {
        // Display current 'scanning' status
        GUIStyle style = new GUIStyle(GUI.skin.button);
        style.fontSize = 50;
        GUI.Box(new Rect(200, 500, 350, 150), mIsScanning ? "Scanning" :
"Not scanning",style);
        // Display metadata of latest detected cloud-target
        TemplateMaquina template =
JsonUtility.FromJson<TemplateMaquina>(mTargetMetadata);
        if(template != null)
        {
            GUI.Box(new Rect(200, 900, 700, 150), "Metadata: " +
template.nombre, style);
        }
        // If not scanning, show button
        // so that user can restart cloud scanning
        if (!mIsScanning)
        {
```

```
            if (GUI.Button(new Rect(200, 1300, 500, 150), "Restart
Scanning", style))
            {
                // Restart TargetFinder
                mCloudRecoBehaviour.CloudRecoEnabled = true;
                mTargetMetadata = "";
            }
            if (GUI.Button(new Rect(200, 1600, 500, 150), "Abrir
Información", style))
            {
                SceneManager.LoadScene("Maquina1");
            }

        }
    }
}
```

```
public class OpenImage : MonoBehaviour
{
    public RawImage image;
    public AddTarget addTargetScript;

    public void OpenExplorer()
    {
        PickImage();
    }

    private void PickImage()
    {
        NativeGallery.Permission permission =
NativeGallery.GetImageFromGallery((path) =>
        {
            Debug.Log("Image path: " + path);
            if (path != null)
            {
                // Create Texture from selected image
                Texture2D texture = NativeGallery.LoadImageAtPath(path);
                if (texture == null)
                {
                    Debug.Log("Couldn't load texture from " + path);
                    return;
                }
                else
                {
                    image.texture = texture;
                    addTargetScript.imgPath = path;
                }

            }
        }, "Select an image");

        Debug.Log("Permission result: " + permission);
    }
}
```

```
public class VuforiaTools : MonoBehaviour
{
    public Texture2D texture;
    private VuforiaToolsConfiguration vtc;

    //Server keys are placed here
    private string access_key =
"16805b6ff47239731580cf792c2c3b709801e17c";
    private string secret_key =
"780932c596e74b3c3d9de4f0cae737abae9ba34f";
    //Address of Vuforia's server
    private string url = @"https://vws.vuforia.com";


    //Function to upload the image target
    public string UploadImageTarget(string targetName, float width,
string imagePath, bool active, string metaData)
    {
        return UploadTarget(targetName, width, imagePath, active,
metaData);
    }


    private string UploadTarget(string targetName, float width, string
imagePath, bool active, string metaData)
    {
        string requestPath = "/targets";
        string serviceURI = url + requestPath;
        string httpAction = "POST";
        string contentType = "application/json";

        byte[] imgBytes = File.ReadAllBytes(imagePath);
        string imageEncode64 = Convert.ToBase64String(imgBytes);
        JsonPost targetPost = new JsonPost(targetName, width,
imageEncode64, active, metaData);
        string requestBody = JsonUtility.ToJson(targetPost);

        UnityWebRequest webRequest = UnityWebRequest.Post(serviceURI,
UnityWebRequest.kHttpVerbPOST);
        UploadHandlerRaw MyUploadHandler = new
UploadHandlerRaw(System.Text.Encoding.UTF8.GetBytes(requestBody));
        webRequest.uploadHandler = MyUploadHandler;

        return VuforiaRequest(requestPath, httpAction, contentType,
requestBody, webRequest);
    }


    public string VuforiaRequest(string requestPath, string httpAction,
string contentType, string requestBody, UnityWebRequest unityWebRequest)
    {
        string serviceURI = url + requestPath;
        string date = string.Format("{0:r}",
DateTime.Now.ToUniversalTime());

        unityWebRequest.SetRequestHeader("host", url);
```

```
        unityWebRequest.SetRequestHeader("date", date);
        unityWebRequest.SetRequestHeader("content-type", contentType);

        MD5 md5 = MD5.Create();
        var contentMD5bytes =
md5.ComputeHash(System.Text.Encoding.ASCII.GetBytes(requestBody));
        System.Text.StringBuilder sb = new System.Text.StringBuilder();
        for (int i = 0; i < contentMD5bytes.Length; i++)
        {
            sb.Append(contentMD5bytes[i].ToString("x2"));
        }

        string contentMD5 = sb.ToString();

        string stringToSign = string.Format("{0}\n{1}\n{2}\n{3}\n{4}",
httpAction, contentMD5, contentType, date, requestPath);

        HMACSHA1 sha1 = new
HMACSHA1(System.Text.Encoding.ASCII.GetBytes(secret_key));
        byte[] sha1Bytes =
System.Text.Encoding.ASCII.GetBytes(stringToSign);
        MemoryStream stream = new MemoryStream(sha1Bytes);
        byte[] sha1Hash = sha1.ComputeHash(stream);
        string signature = System.Convert.ToBase64String(sha1Hash);
        unityWebRequest.SetRequestHeader("authorization",
string.Format("VWS {0}:{1}", access_key, signature));
        unityWebRequest.SendWebRequest();
        while (!unityWebRequest.isDone && !(unityWebRequest.result ==
UnityWebRequest.Result.ConnectionError)) { }

        //If request error, return fail
        if (unityWebRequest.error != null)
        {
            return unityWebRequest.downloadHandler.text;
        }
        else
        {
            if(httpAction == "DELETE")
            {
                return "Deleted";
            }
            return unityWebRequest.downloadHandler.text;
        }
    }
}
```