UNIVERSIDAD POLITÉCNICA DE MADRID

ESCUELA TÉCNICA SUPERIOR
DE INGENIEROS DE TELECOMUNICACIÓN

GRADO EN INGENIERÍA DE TECNOLOGÍAS Y SERVICIOS DE TELECOMUNICACIÓN

TRABAJO FIN DE GRADO

DEVELOPMENT OF AN AUGMENTED REALITY ANDROID APPLICATION FOR INTERACTING WITH SMART OBJECTS IN A SMART OFFICE USING THE TECHNOLOGY GOOGLE ARCORE

PABLO ALBERTO GARCÍA BENEDICTO

ENERO 2020

TRABAJO DE FIN DE GRADO

Título:	Desarrollo de una Aplicación Android en Realidad Aumentada para Interacturar con Objetos Inteligentes en una Oficina Inteligente empleando la tecnología Google ARCore
Título (inglés):	Development of an Augmented Reality Application for Interacting with Smart Objects in a Smart Office using the technology Android ARCore
Autor:	Pablo Alberto García Benedicto
Tutor:	Carlos Ángel Iglesias Fernández
Departamento:	Departamento de Ingeniería de Sistemas Telemáticos

MIEMBROS DEL TRIBUNAL CALIFICADOR

Presidente:

Vocal:

Secretario:

Suplente:

FECHA DE LECTURA:

CALIFICACIÓN:

UNIVERSIDAD POLITÉCNICA DE MADRID

ESCUELA TÉCNICA SUPERIOR DE
INGENIEROS DE TELECOMUNICACIÓN

Departamento de Ingeniería de Sistemas Telemáticos
Grupo de Sistemas Inteligentes

TRABAJO FIN DE GRADO

DEVELOPMENT OF AN AUGMENTED REALITY ANDROID APPLICATION FOR INTERACTING WITH SMART OBJECTS IN A SMART OFFICE USING THE TECHNOLOGY GOOGLE ARCORE

PABLO ALBERTO GARCÍA BENEDICTO

ENERO 2020

Resumen

El objetivo principal de este trabajo de fin de titulación es la implementación de una aplicación Android para teléfonos móviles en la cual modelos de realidad aumentada permitan interactuar con distintos objetos inteligentes presentes en una oficina inteligente, cuyas pruebas se han realizado en la del Grupo de Sistemas Inteligentes (GSI) de la Escuela Técnica Superior de Ingenieros de Telecomunicación de Madrid (ETSIT). Mediante la utilización de la aplicación, los usuarios podrán crear un usuario mediante el cual se autenticarán permitiéndoles emplear la cámara de su teléfono para reconocer imágenes en dos dimensiones o superficies planas presentes en la smart office que generen automáticamente o con la acción de una pulsación modelos de realidad virtual en tres dimensiones con los que interactuar para, por ejemplo, abrir la puerta o apagar y encender las luces entre otras funcionalidades.

Se ha considerado de especial interés el hecho de que los usuarios requieran de autenticación para poder acceder a la aplicación. Esto es así ya que esta está diseñada específicamente para los integrantes del GSI, de tal forma que se pueda llevar un control de acceso y evitar que personas ajenas puedan modificar el estado de los objetos inteligentes. Para llevar a cabo el proceso de registro e inicio de sesión se ha creado una base de datos mediante Google Firebase de tal forma que el método de login tenga lugar a través de email y contraseña. Gracias a Android Studio, entorno de desarrollo empleado para llevar a cabo la aplicación, se permite la integración de la base de datos en la aplicación.

La realidad virtual que engloba el proyecto se lleva a cabo mediante la tecnología ARCore, plataforma de Google que permite, gracias a diferentes APIs, que el teléfono reconozca las superficies del entorno mediante la cámara e interactuar con él. En este caso, la técnica empleada es la de imágenes aumentadas: a través de la cámara del dispositivo móvil empleado se reconocerán imágenes en dos dimensiones colocadas en zonas específicas situadas cerca de los objetos inteligentes con los que se posibilita la interacción. Al reconocer dichas imágenes se generan modelos de realidad aumentada que muestran las distintas opciones con las que cuenta cada objeto inteligente y permiten modificar el estado de los mismos.

Palabras clave: Aplicación móvil, Android, Android Studio, ARCore, imágenes aumentadas.

Abstract

The main aim of this project is to implement an Android mobile application in which augmented reality allows users to interact with several intelligent objects that can be found in a smart office, being tested in the Intelligent Systems Group (GSI). By using the app, the GSI members will be able to create an account so as to log in and use the camera of their own mobile phones to recognize two dimensional pictures. Once this has been done, three dimensional models will be displayed letting the user choose between different options to modify the intelligent objects state, such as opening the laboratory door or turning on or off the lights among other functionalities.

Authentication in order to access the application has been considered as a priority. The main reason is that the access to intelligent objects should be restricted to registered members to prevent any security flaw. A Firebase database has been created to allow the process of registration and logging in with email and password as credentials. Android Studio, the environment used to develop the application, makes the database integration feasible.

The augmented reality related to this project is carried out according to ARCore technology, a Google platform which allows Android mobile phones to recognize surfaces present in a near distance and interact with them. According to this case, augmented images is the technology implemented: the camera will detect 2D images located in certain zones near the objects setted to interact with. Recognizing one of this images will lead the app to show a 3D model which includes the different options the object offers and letting the user interact with it.

Keywords: Mobile application, Android, ARCore, Firebase, Android Studio, augmented image

Agradecimientos

Dedicado a mi familia, amigos y Elena, sin cuyo apoyo no habría sido posible llegar hasta aquí. Agradecerle también a mi tutor del trabajo, Carlos Ángel Iglesias Fernández, por orientarme y estar atento siempre cuando lo he necesitado.

¡Muchas gracias a todos!

Contents

List of Figures

CHAPTER 1

Introduction

1.1 Context

Augmented reality idea has evolved a long way from being considered as science-fiction related to being a science-based concept. At first, the most evident downside that this technology came across were costs, which tended to be substantial. In contrast, augmented reality can now be available on any average mobile gadget with no cost at all nowadays.

"The limitation comes from the main interest we have in our daily life, which is not directed toward some virtual world, but rather toward the real world surrounding us" [1]. As a concept, augmented reality is considered to be attractive, but is it really necessary? Classical electronic devices are isolated to their physical environment and just need a pile of instructions given by a user and electrical power. In addition, the boundless amount of information available online is generally disconnected from the real world, and that is accurately the gap AR is aiming to suture [2].

Several appliance examples can be currently found with their apps available or being under development in nearly every sector this days, including architecture, commerce, tourism, education or medicine among others. The premise all AR apps share is being able to recognize the surrounding physical world through the sensor interface of an electronic device

and add the virtual information required on each case, overlapping and combining it with the real data in such a way that interaction is made possible. In other words, "augmented reality can overlay computer-generated information on views of the real world, amplifying human perception and cognition in remarkable new ways" [1] enhancing machine-men communication.

1.2 Project goals

The main aim of this project is to create a scalable Android application which shows an augmented reality environment to interact with the intelligent objects located in a smart office. The interaction itself consists in portraying virtual floating buttons which can result in different actions, depicting information virtually or playing augmented reality videos.

Among the principal objectives this project has reached are the following:

- Deep research on Google ARCore, the technology chosen and based to develop augmented reality in Android Studio.
- Design and implementation of an Android application capable to use ARCore to accomplish the goals previously mentioned.
- Two augmented reality techniques implemented: surface recognition and augmented images.
- Three different runtime stages: user register and login, navigation through the app and augmented reality interaction.
- Interface based on Google Firebase for register and login purposes.
- Intelligent objects dually protected: WiFi password and authentication.
- Different techniques research to improve the Android application appearance.

1.3 Structure of this document

In this section, a brief overview of the chapters included in this document is provided. The project is divided into the following sections:

Chapter 1 Project introduction in which context, motivation, main goals and document structure are defined.

Chapter 2 Enabling technologies to this project are explained.

Chapter 3 A requirement analysis is made, in which the use cases and principal actors of the system are introduced supported on UML schemes.

Chapter 4 The architecture of the project is described, including the design phase and implementation details.

Chapter 5 The use cases and how the actors interact with augmented reality are described in depth for the smart office environment.

Chapter 6 Closing conclusions after completing the project are given together with a brief future perspective.

CHAPTER 2

Enabling Technologies

2.1 Introduction

In this chapter, a full description of the technologies implemented and the programming environment is made. Starting off by analyzing what augmented reality is and avoiding confusion with other similar techniques, a clear definition of its usages is given. Secondly, it is explained how Google ARCore is implemented in Android Studio. Vectary, the platform used to create some of the 3D models. Finally, the register and login method through Google Firebase is presented.

2.2 Augmented reality

Augmented reality (AR) [15], sometimes confused with virtual reality, is the term used to describe the group of technologies that allows a user to visualize part of its environment throughout an electronic device that adds graphic information. The tangible physical objects located in the real world combine themselves with the virtual ones, creating an augmented reality in real time.

Several **differences** are found between augmented reality, virtual reality (VR) and a third term that combines both of them: mixed reality (MR).

- **Augmented reality** uses our own world as foundation. A camera and a screen let us interact with the augmented objects that are projected on a real environment whilst **virtual reality** creates a completely new world in which we are immersed.
- **Mixed reality** [15] is a hybrid between augmented reality and virtual reality that allows to create dimensions in which real and virtual people and objects can interact.
- Another significant difference to consider are the objects these three similar technologies use to manage the realities created. Virtual and mixed realities need a specific object such as VR glasses or headset, while augmented reality only requires an application downloaded on a phone or tablet.

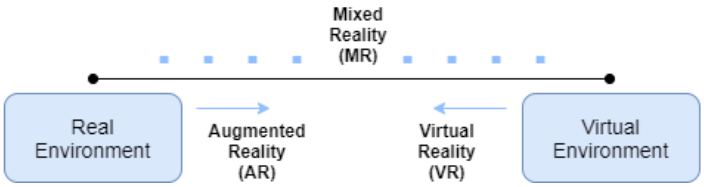

Figure 2.1: AR, VR, and MR schema [20]

Keeping that in mind, we will focus on augmented reality, the technology chosen to develop this project. What makes it being one of the principal innovation areas an have a rocketing potential are its characteristics:

- By using augmented reality we can interact with the real world by utilizing non physically existing objects.
- A three dimensional pattern is used. Information is shown with perspective making the user aware of how the augmented object changes depending on factors such as position and height.
- It depends on the context, as the virtual information we see is strictly linked to it.
- Augmented reality provides real time interaction. Any action, change or answer carried out by the user will immediately have repercussion.

2.2.1 Augmented reality types

There are several types of augmented reality depending on the final purpose and the real world components that are involved. Among them, the following can be distinguished:

- **Images**: Any image can be used to deploy an augmented three dimensional objects. Formerly, simple images such as QR codes were used as markers, whilst now the technology has lead the way to recognize more complex pictures such as a piece of art or a company logo, making the experience even more natural.
- **Spaces**: Augmented reality allows a device to recognize any surface, room or space and memorize the position and physical capacity of the surrounding environment. By using this information, it is possible to generate three dimensional maps where augmented reality objects can be placed.
- **Places**: It is possible to create points of interest (POI) to visualize augmented reality content in certain places by using the specific coordinates where they are located. Among the uses this could be applied for are tourism, culture or finding cars or flats for rent.

2.2.2 Augmented reality advantages

Augmented reality brings many advantages when is applied to different sectors:

- Time optimization is allowed in workers routines due to the useful and accurate information displayed.
- A new communication channel is opened for users, who interact in a direct and instant way with the virtual objects placed in a real environment. This fact makes the users have an immersive sensation and can be applied in advertisements to turn them more effective.
- In a near future, hands free interaction with elements such as augmented reality glasses will make it possible to manage the virtual world with no physical intervention.

These are just some of the many uses that augmented reality can implement. For years to come, the development of this technology will make it possible to create more useful applications in order to simplify our lives.

2.3 Google ARCore

ARCore [3] is the platform that Google provides to build augmented reality experiences for Android and IOS devices. It makes your phone able to recognize its environment, understand the world and work with the information provided by using different APIs. Some of them available across Android and IOS enabling shared augmented reality experiences.

One of its main competitors, Apple's ARKit, shares many similarities. It is an algorithmically different framework or development environment which offers a very close final product to the one ARCore provides. Basically, both of them offer SLAM (Simultaneous Location and Mapping). SLAM offers the possibility to perform an instant environment scanning and mapping, locating where the device is in real time. In other words, by utilizing any device compatible with a camera available and enough processor speed, it can measure where the device is exactly placed taking the floor, ceiling and any kind of flat obstacles as reference. SLAM just let the device know its position and sets a base to place the relevant augmented information in the real environment. As the mapping is done, knowing the position of the camera and the obstacles located nearby lets add augmented objects such as 3D models, images or videos at any time using the environment mapped as support and fixing them on the locations where they are placed, also known as augmented reality anchors. This makes the object seem to be located in the same spot as the user changes the camera perspective.

2.3.1 Fundamental concepts

To integrate the augmented reality objects in the mapping already described virtually recognizing the real world, ARCore uses three key capabilities [3]:

- ***Motion tracking:*** the devices used are able to scan, understand and track its position related to real world measures. While the device is moving around the world, ARCore executes a process called concurrent odometry and mapping (COM). To clarify, "odometry is a measurement method from motion sensor or rotation sensor to estimate change in position over time and is used to stimate a position from a starting location. This method is sensitive to errors due to the integration of velocity measurements over time to give position estimates. Rapid and accurate data collection, instrument calibration, and processing are required in most cases for odometry to be used effectively" [17]. COM and SLAM result in the phone understanding where is placed. That way, ARCore detects through the camera of the device distinct ***feature***

points which are used to determine a change in location.

- ***Environmental understanding:*** the device used is allowed to recognize the location and size of all the vertical or horizontal flat surfaces and obstacles nearby and determine their bounds.

- ***Light estimation:*** the device used can distinguish a wide range of light conditions on its environment thanks to ARCore. This makes it possible to adjust the augmented reality objects brightness and saturation depending on actual ilumination providing a more relistic sensation.

- ***User interaction:*** the methot used by ARCore to get the smartphone screen coordinates (x,y) on which the user taps is hit testing. It consists in projecting a ray through the camera of the device to determine the position in the real world with which and object or surface intersects with the ray.

2.3.1.1 Anchors and trackables

The visual information obtained with the feature points helps to stimate the ***pose*** of the device, or in other words, its position and orientation over time related to the surrounding environment. These poses can change as time passes due to ARCore recognizing better the surroundings through the camera. The planes metioned before and the points obtained by user interaction are special objects known as ***trackables***, and as its own name refers, their position is being tracked constantly.

To create an augmented reality object and fix it on a trackable, an ***anchor*** must be set. Anchors act as immovable points wich can be established on any trackable or augmented image, as it is described in following sections.

2.3.1.2 Augmented images

Augmented images [3] is one of the main techniques used in this project among surface recognition to develop the augmented reality experience. Augmented images consist in a Google ARCore's feature that allows an application to recognize a two dimensional image, mobile or still, present in the real world. These key images can be a simple QR code or even any poster or billboard that the developer considers and is valid enough for the criteria of the tool described in appendix C. The images can be added to the application creating a database with which the the app gets lighter and the compilation time is diminished, or

individually. Once they are created, the app is able to recognize them, their bounds and pose.

2.4 Android Studio

Android Studio is the framework chosen to develop the augmented images application. Considered as the official platform for Android, it replaced Eclipse as former IDE for Android applications development. Based on JetBrains IntelliJ IDEA, it has been published with no monetary cost through Apache 2.0 license. It is available for Linux, macOS and Microsoft Windows and designed specifically for Android applications development.

2.4.1 Requirements

To start creating an ARCore application with Android Studio [4] and debugging it after completion, a list of prerequisites should be fulfilled:

- Android Studio 3.1 version or higher with Android SDK Platform 7.0 (API level 24) must be installed.
- Android mobile phones emulator or your own device should be prepared to try the app. Android augmented reality applications can be played either on an emulator or directly on a connected mobile phone. Both of the must have the latest version of ARCore application installed and updated. Otherwise, the augmented reality app won't run correctly and ARCore should be installed for free through Play Store. If the emulator option is chosen, the version required should be 27.2.9 or later. OpenGL ES 3.0 or higher must be supported and enabled in the emulator as well.

Android Studio builds the project into a debuggable APK, installs the APK on the device selected and runs the application on it. However, if ARCore application is not already available, installed or updated, an option can be selected to install it from Google Play Store.

2.4.2 Enabling ARCore in Android Studio

As mentioned before, ARCore is the key technology and the one chosen to make augmented reality possible in our app. In order to enable ARCore in an Android Studio project, the next steps should be followed:

- Android manifest document must include AR required or AR optional entries.
- Dependencies should be added to the project build dependencies (build gradle, project and module).
- Camera permission should be granted so that the application is able to access the camera available on the device.
- Runtime checks are recommended to verify if ARCore is correctly installed and updated as well as camera permission granted.

2.4.3 Android Manifest

The manifest document (AndroidManifest.xml) provides relevant information for the Android system about the app under development. The former information should be regarded so that the system is able to run the app. Among its several purposes, the manifest document manages the following:

- Names the application Java package, which is used as unitary identifier to be recognized by the app.

```
<manifest xmlns:android="http://schemas.android.com/apk/res/
android"
    package="com.tfg.pablo.arswitch">
```

- App components such as activities or services are presented. By describing the tag of each component, a reference to the class they implement is given with Intent messages in case of having them. An Intent could be defined as and abstract description of an operation to launch other activity and send information if required.

```
<activity
      android:name=".Index"
      android:label="@string/title_activity_index"
      android:theme="@style/AppTheme.NoActionBar">
</activity>
```

- Includes AR optional or AR required entries. In the application implemented, only the compulsory options are defined. The main differences between optional and required AR entries are the android:required and android:value options as well as the minimum SDK version necessary: at least 14 for optional and 24 for required.

  If an application is AR required, Android Play Store will only show the option to download it on devices compatible with ARCore. What's more, devices with no ARCore

application downloaded will automatically install it. Nevertheless, further runtime checks should be performed to verify ARCore has not been uninstalled afterwards or is not up to date.

```
<uses-feature android:name="android.hardware.camera.ar"
android:required="true"/>

<meta-data android:name="com.google.ar.core" android:value="required"/>
```

- Access to the camera is permitted, as using it is considered as the base of augmented reality. There is no chance to experience AR without looking through a camera to interact in the different ways augmented reality offers.

```
<uses-permission android:name="android.permission.CAMERA"/>
```

- The fact that some of the use cases required internet connection to develop the actions established, such as accessing the different GSI social networks or making HTTP POST instances to vary the intelligent objects parameters, made it compulsory to allow internet permission by modifying the manifest document.

```
<uses-permission android:name="android.permission.INTERNET" />
```

2.4.4 Dependencies

As the principal aim is to perform ARCore in Android Studio, some dependencies should be added to the project as well. Android Studio compilation system allows including binary files or external libraries as dependencies. These dependencies can be located in the file system or in an external repository. The augmented reality project described contains the following:

- Project's build grade must include Google's Maven repository:

```
repositories {
    google()
    jcenter()
}
```

- In order to add Google's sceneform plugin to perform augmented reality, its dependency should also be added to project's build gradle:

```
dependencies {
    ...
    classpath 'com.google.ar.sceneform:plugin:1.7.0'
}
```

- App's build gradle must include Google's ARCore latest library as dependency:

```
dependencies {
    ...
    implementation 'com.google.ar:core:1.2.0'
}
```

2.4.5 Runtime checks

Android manifest document makes clear that android AR is required with the statements mentioned. However, apart from describing the requirements in manifest.xml, runtime verifications should be performed so that no problems are encountered while running the app.

Regarding the code, the majority of runtime checks have been added to onResume method in ARMainActivity. If the AR session has not been initialized yet, runtime verifications are performed starting off by checking if the camera available on the device has permission and showing messages if ARCore is not installed, updated or not supported due to the SDK version the device presents.

Apart from that, the method onRequestPermissionsResult should also be included to inform that camera permissions are needed to run the application in case of not having them accepted, or just launching permission settings.

2.5 Philips Hue

Philips Hue is a section of intelligent products, more precisely light bulbs, led stripes and a broad range of different lamps which are sold as a wireless personal illumination system. As described on its website, they allow a simple light control and create the most suitable ambiance for each situation. Philips Hue works with different intelligent devices such as Amazon Echo, Google Home or Apple Kit among others.

In this project, the exact product used is known as "Portable Lamp, Go" [5], which is a luminous hemisphere made out of a white synthetic material that can work as an independent light or as part of an intelligent illumination system thanks to Hue Bridge. Its integrated battery makes it possible for the light to be portable with an autonomy of 2,5 hours in full-brightness mode or up to 18 hours when used in candle-effect mode. Other characteristics considered to be mentioned are its 12 volts LED-type, 43 watts power or an estimated 20,000 hours lifetime.

Figure 2.2: Philips Hue Portable Lamp

2.6 Vectary

Vectary [6] is a 3D design tool used to create some of the 3D models the augmented reality application comprises. It is based on a drag and drop 3D modeling tool. In other words, different shapes and forms can be dragged into the modeling area to form the desired model. Each of the pieces added can be modified in multiple ways by modifying its position, size,

color, material or reflectivity among many other options.

To complement the application use cases, a 3D GSI logo and signboards for some doors have been designed:

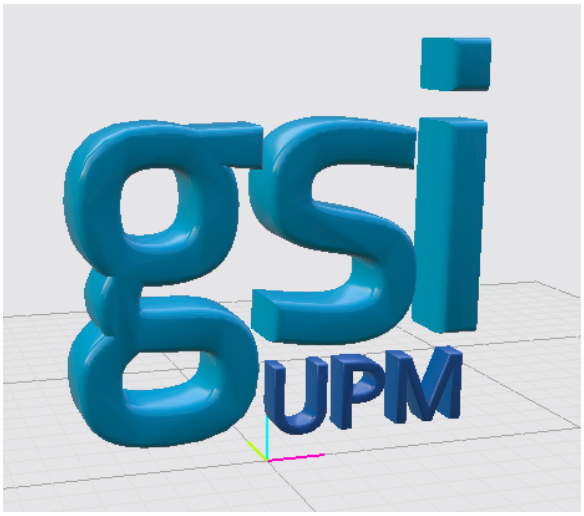

Figure 2.3: GSI 3D logo

2.6.1 Importing 3D models into Android Studio

In order to add these 3D models to the Android Studio project, a sample data folder has been created. Any 3D model to appear in the application should be inside this folder. There, each of the objects should have the next files to be imported correctly:

- OBJ file in which the shape and size is vectorially defined.
- PNG file where all the rendering information such as color and texture is defined.
- MTL file that links the rendering information to de object itself.

Once every necessary file is located in the sample data folder, the 3D object can be brought to life by importing the 3D sceneform asset making a right click on the .obj file. A sceneform A file and sceneform B file output path will be asked, selecting the sample data folder for the .sfa file and /res/raw folder for the .sfb file. After importing the sceneform asset, it is ready to be used.

2.7 Google Firebase

As some of the use cases this application comprises involve actions that the smart office members should be able to make such as opening the door or switching on and off certain lights, a login and register system has been considered essential. After analyzing several options, Google Firebase turned out to be the most suitable for the purpose aimed.

Google Firebase [7] offers different features like a real time database, authentication, hosting, storage, cloud functions, Android tests laboratory, performance monitoring, failures report, remote configuration or AdMob among many other options. However, knowing all its benefits, the augmented reality application will only use its authentication services.

By making our users pass through an authentication process not only the app gets safer but a useful analytics panel can be accessed. Apart from checking a graphic with the amount of users connected each day, a table containing all the users along with the date in which they created their accounts, the last time they connected and their coded UID is shown.

Furthermore, users must be connected to GSI WiFi to perform certain JSON post actions that modify intelligent objects state. The password to connect this WiFi is confidential and only GSI members have access to it, so a dual protection alongside Google Firebase authentication is carried.

2.7.1 Linking Google Firebase to the Android Studio project

In contemplation of adding the beneficial qualities Google Firebase offers to the augmented reality Android Studio project, a series of actions have been taken:

- Two new classes (Login and Register) and their layouts (activity login and activity register) have been created to shelter the authentication formulary.
- A FirebaseAuth object instance must be declared and then initialized in the method onCreate.

```
private FirebaseAuth firebaseAuth;
…
firebaseAuth = FirebaseAuth.getInstance();
```

- App build gradle must have Google Firebase among its dependencies. The version recommended at the moment of implementing the authentication is 16.0.1:

```
dependencies {
    …
    implementation 'com.google.firebase:firebase-auth:16.0.1'
    implementation 'com.google.firebase:firebase-core:16.0.1'
}
```

- Manifest document should include references to the new classes created. As the login class has been selected, its activity label must define the action as 'MAIN' and category as 'LAUNCHER'

```
<activity android:name=".User.login">
    <intent-filter>
        <action android:name="android.intent.action.MAIN" />

        <category android:name="android.intent.category.LAUNCHER" />
    </intent-filter>
</activity>
```

- Functionality to log in is added in method userLogin. If the login process is successful a positive message will be prompted and the name of the user (email without any letter after @ included) will be transferred to the index activity, where it will be shown to the user as it follows: "User: . . . ". In case of the login process encountering an error, a "Non existing user" message will be shown.

```
// Iniciar sesion usuario
firebaseAuth.signInWithEmailAndPassword(email, password)
        .addOnCompleteListener(this, new OnCompleteListener<AuthResult>() {
            @Override
            public void onComplete(@NonNull Task<AuthResult> task) {
                if (task.isSuccessful()) {
                    Toast.makeText(login.this, "Inicio de sesión correcto",
                            Toast.LENGTH_SHORT).show();

                    // Buscar la posicion en la que esta @ en el string email
                    int pos = email.indexOf("@");
                    String userSinEmail = email.substring(0, pos);

                    // Enlazar con la activity Index y pasar info de usuario
                    Intent intent = new Intent(getApplication(),Index.class);
                    intent.putExtra(Index.user, userSinEmail);
                    startActivity(intent);

                } else {
                    Toast.makeText(login.this, "Usuario no existente",
                            Toast.LENGTH_SHORT).show();
                }
                progressDialog.dismiss();
            }
        });
```

- Functionality to register is added in method userRegister. A user will satisfactorily register ("Successful registration") if the registration email is not already taken. In

case of the email being already chosen, there will be a collision exception and a "User already exists" message will be prompted. Other error cases such as blank or incorrect inputs are also considered, showing their respective "Register error" messages.

```
// Registrar nuevo usuario
firebaseAuth.createUserWithEmailAndPassword(email, password)
        .addOnCompleteListener(this, new OnCompleteListener<AuthResult>() {
            @Override
            public void onComplete(@NonNull Task<AuthResult> task) {
                if (task.isSuccessful()) {
                    Toast.makeText(login.this, "Registro con éxito",
                            Toast.LENGTH_SHORT).show();
                } else {
                    if(task.getException() instanceof
FirebaseAuthUserCollisionException){
                        Toast.makeText(login.this, "Usuario ya existente",
                                Toast.LENGTH_SHORT).show();
                    } else {
                        Toast.makeText(login.this, "Error en el registro",
                                Toast.LENGTH_SHORT).show();
                    }
                }
                progressDialog.dismiss();
            }
        });
```

CHAPTER 3

Requirement Analysis

3.1 Introduction

In this chapter, the requirement analysis is described by introducing an UML (Unified Modeling Language) use cases model. UML [8] is considered to be a graphic language to visualize, build and document a software system. It addresses the system issues from different perspectives, more precisely, the static aspects (structure) and dynamic aspects (behavior and interactions).

3.2 Use cases

Use cases model the system behavior due to external actors, and represent what the system will do for for the benefit of the actors. The following sections depict the use cases considered in this project, giving information about the uses of the application and an extended list of requisites. First, the principal and secondary actors involved are described, followed by an UML [8] schema where its role and how they interrelate is made clear.

3.2.1 System actors

Identifying system actors is one of the first steps to take in use case analysis. Evaluating the application that is going to be developed and its specific purposes should imply the deduction of who or what interacts with the system. "An actor specifies a role played by a user or any other system that interacts with the subject" [9] and can represent human, external hardware or other subjects roles. Considering this, the system actors taken into account in this project are the following:

- ***User:*** It is considered to be the system's main actor and could be described as any person using the mobile application by using its smartphone. The user is directly in touch with everything related **app**, from authentication to the augmented reality experience, and will be responsible of whichever state modification -made throughout smartphone interaction- of the smart objects located in the smart office executed.

- ***Google ARCore [3]:*** Regarded as a secondary actor, its doubtless importance is given because it is responsible of surfaces recognition and mapping, key images recognition, augmented reality nodes creation along with their respective anchors and definitely, AR objects displaying which is the main objective in which this project is focused.

- ***Google Firebase [7]:*** This secondary actor has not been considered in the UML schema for the reasons exposed in section 3.2.2.1, but its importance, as it is the software related to the authentication process, makes it compulsory to quote it.

- ***GSI Server:*** Proxy in charge of managing post requests related with the door system, changing its parameters when the 'open door' augmented reality button is pressed.

- ***Philips Hue Server:*** Proxy in charge of managing post requests related with the Philips Hue Go Lamp, changing its parameters depending on the interaction.

- ***Social Media:*** This group of secondary actors is formed by applications used for social interactions such as Youtube, social networks like Twitter and mailing platforms like Gmail, Outlook or others.

- ***Smartphone:*** Principal actor that gives a user the chance to interact with the application through its interface and experience the augmented reality through the camera device.

3.2.2 Use cases

Once the system actors have been presented, the UML schema described in figure 3.1 summarizes how they interrelate and the use cases involved.

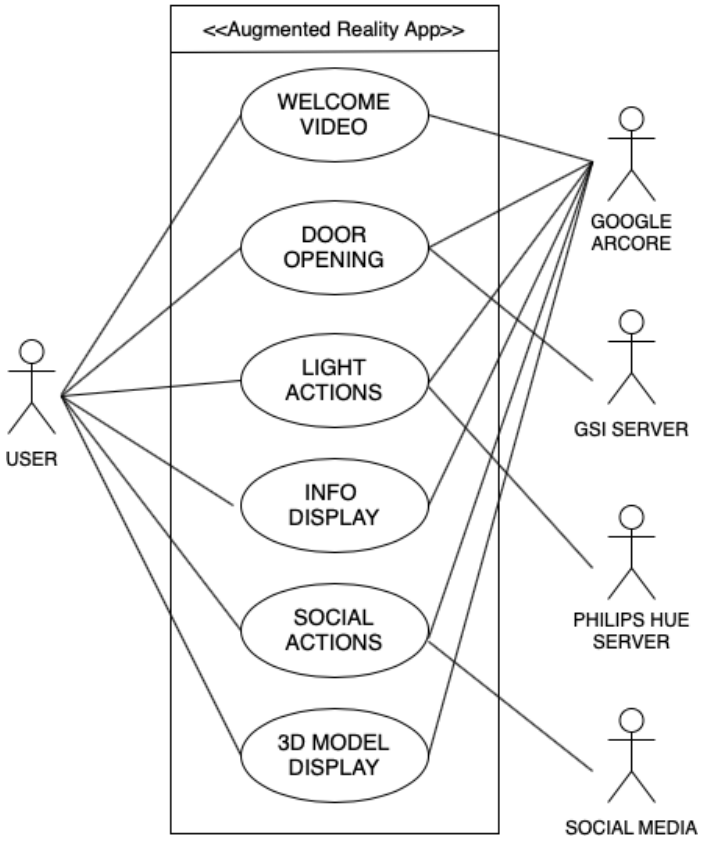

Figure 3.1: Use cases UML schema

To clarify how every use case is managed, an UML sequence diagram [8] is stated for each of them in the following sections. This sort of diagrams represent a vertical lifeline for each object, representing its existence during a period of time, and a control focus which can be recognized as the narrow vertical rectangles that represent the duration of an object's action.

3.2.2.1 Authentication

An authentication process —consisting of the user introducing its personal data (user and password) and Google Firebase validating the access if the user had been registered before— is the key point from where the app starts. The user who arrives at the smart office and opens the app has to face it first, writing the required credentials, submitting them, logging

in if they are validated and finally gaining access to the application.

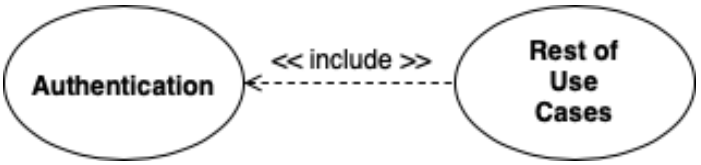

Figure 3.2: Authentication use case relation with other use cases

Authentication is a compulsory first step required by the rest of the use cases, but has not been considered in the UML use cases schema due to the lack of interest compared to the actual issue that the mobile application is involved, augmented reality. However, as it is a use case by itself it must be taken into account and, as figure 3.2 describes, included by the rest of the use cases because it is not optional.

The user interacting with the app is considered to be the principal actor and Google Firebase, software responsible of the authentication process, the secondary.

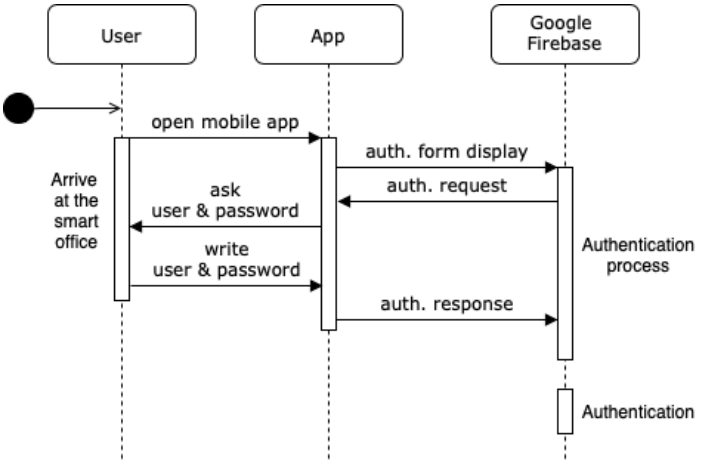

Figure 3.3: Authentication sequence schema

3.2.2.2 Welcome video

As described in figure 3.4, in this use case a principal actor takes part: the user of the app. When it comes to secondary actors, Google ARCore is the one involved in this use case to allow the augmented reality integration in the app.

After a correct process of authentication, the user is able to unfold a navigation drawer, which is the name given to a drop down lateral menu panel in Android, where a tutorial can be chosen as an option. By clicking it, the augmented reality experience is triggered, and the app is then able to recognize flat surfaces through the camera device of the smartphone

thanks to Google ARCore. The surfaces will be mapped and graphically shown to the user as a matrix of equally spaced dots. If the user presses anywhere on it, an AR node will be created and anchored displaying an augmented reality showing a welcoming video.

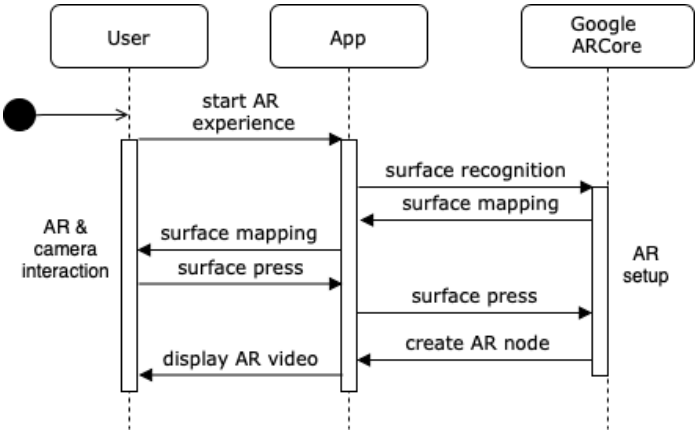

Figure 3.4: Welcome video UML sequence schema

3.2.2.3 Door Opening

The actors involved in this use case are the same as in the previous one: the user of the app as principal actor and Google ARCore as secondary actor, but there is an extra secondary actor, a proxy server.

The part of the authentication process remains the same, as it is a necessary and first step. After it, the user can start the augmented reality experience by choosing the option in the navigation drawer or clicking the bubble button in the main activity. Using the camera device of the mobile phone, the app helped by ARCore detects a key image placed outside the smart office next to the door, and an AR node anchored exactly on it will display an augmented reality button which shows the option "open door". If the user presses the button, an http post request is stated, and the smart office's door will be opened then.

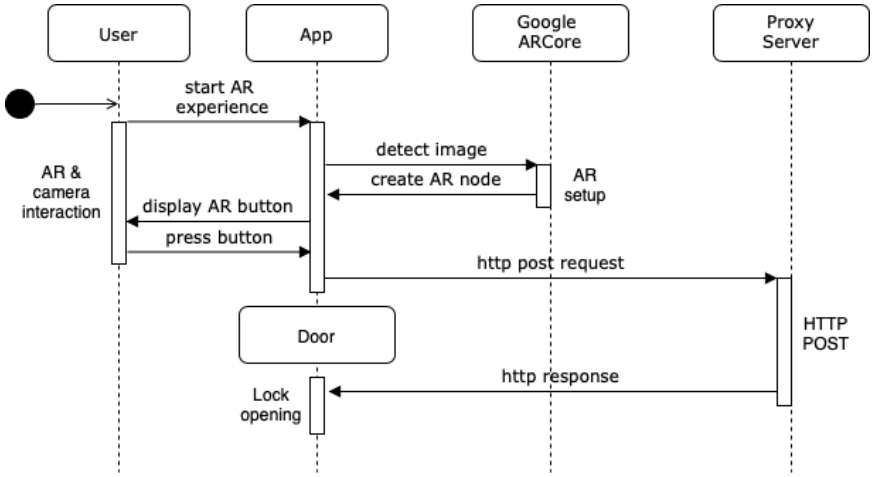

Figure 3.5: Door opening UML sequence schema

3.2.2.4 Light Actions

This use case working dynamic is pretty similar to the door opening use case. In fact, the system actors are exactly the same. After the process of authentication, the app used by the user is able to recognize a key image thanks to ARCore, that creates an anchored AR node on it, but not only one button is displayed this time as augmented reality object. Instead, a group of them consisting of two on/off buttons to switch the light on or off respectively and a row of rounded buttons of different colors to vary the light coloration by clicking on them. The user is able to click on any of them, triggering an http post request and changing the respective parameter turning out on a light action.

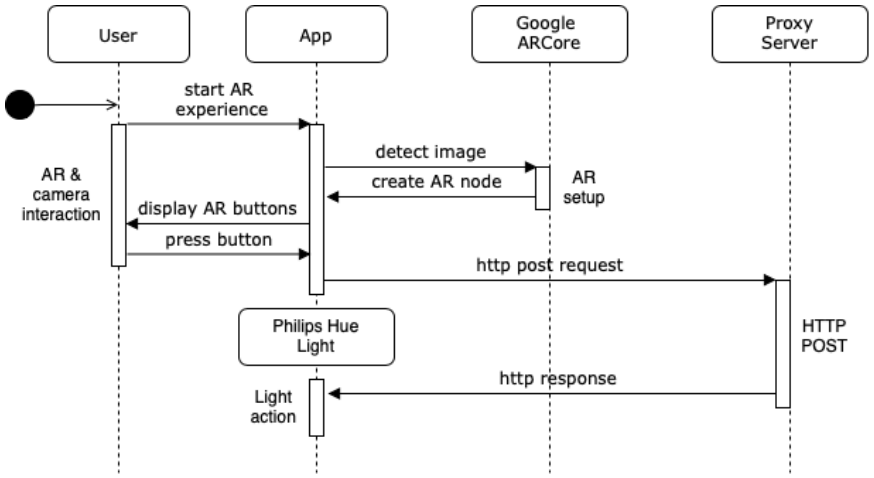

Figure 3.6: Light actions UML sequence schema

3.2.2.5 Info Display

The actors of this use case are the user of the application as principal actor and Google ARCore as secondary actor. Once the authentication process has correctly ended as the figure 3.5 depicts, the user can start the augmented reality experience and the app is able to detect a key image thanks to ARCore, that will create an AR node anchored on the image as in the previous cases, but this time displaying information in augmented reality related to the person occupying the position in the smart office where the key image detected was placed.

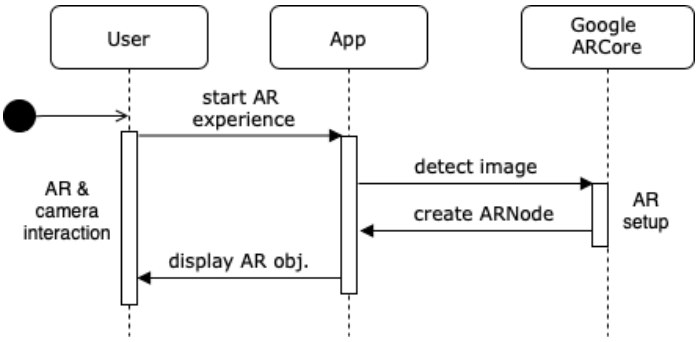

Figure 3.7: Info display UML sequence schema

3.2.2.6 Social Actions

The social actions use case shares the same actors as the light actions or door opening use cases, proxy server included. After a successful authentication process, the user can trigger the augmented reality experience, and the is be able to detect a key image placed in the smart office thanks to the camera device of the smartphone and Google ARCore. As in previous cases, an AR node is created and anchored on the image, displaying an aggregation of four buttons. The user is able to click on any of them, leading to a social network or email web view depending on the button chosen as described in the figure 3.8.

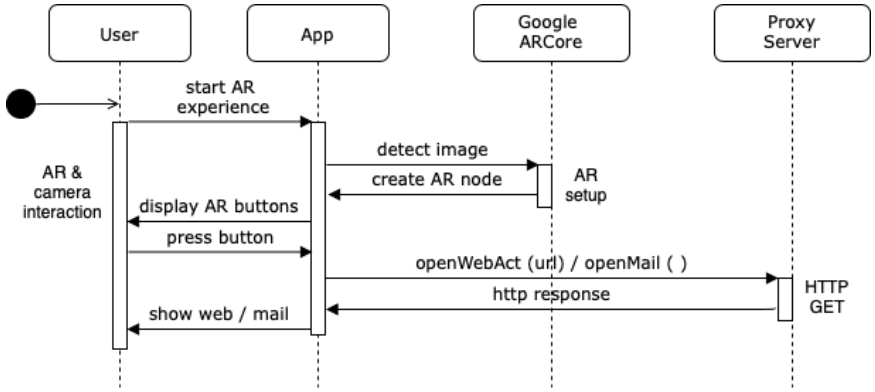

Figure 3.8: Social actions UML sequence schema

3.2.2.7 Model Display

This use case comprises the exact same actors and dynamic as the info use case display, except from the last step. After the authentication and image recognition has taken place, an AR node is created and anchored on the key image, but instead of an information display, a three dimensional augmented reality Intelligent Systems Group logo emerges from it.

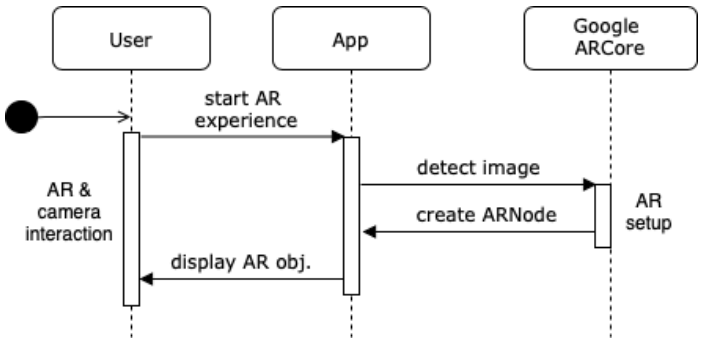

Figure 3.9: Model display UML sequence schema

3.2.3 Conclusion

Now that the seven main use cases that this augmented reality application comprises are explained alongside their similarities and differences, the functionalities have been well defined. In particular, we have identified a number of interesting scenarios where the interaction with smart objects can provide an enhanced experience. However, the future development lines of the application are open, as any new intelligent object brought to the smart office or any new functionality related to augmented reality could be implemented, keeping in mind that AR is a technology in blooming development.

CHAPTER 4

Architecture

4.1 Introduction

In this chapter, the main purpose is to describe the design phase of the project and cover the implementation related to its architecture. Foremost, a schema of the architecture taken into account to develop the project is defined in the overview section, where the different elements that interact and how they interrelate are shown. After regarding these aspects, a deeper description of every individual module is given.

4.2 Overview

This section presents the global architecture of the project, distinguishing between the modules involved in the augmented reality application and how their relations are established. The different sections that are comprised in the application are described in the figure 4.1.

As described in the schema, the architecture is divided into three main modules: the augmented reality module, the server module and the client module; also considering the smart devices which are the final objective of the actions.

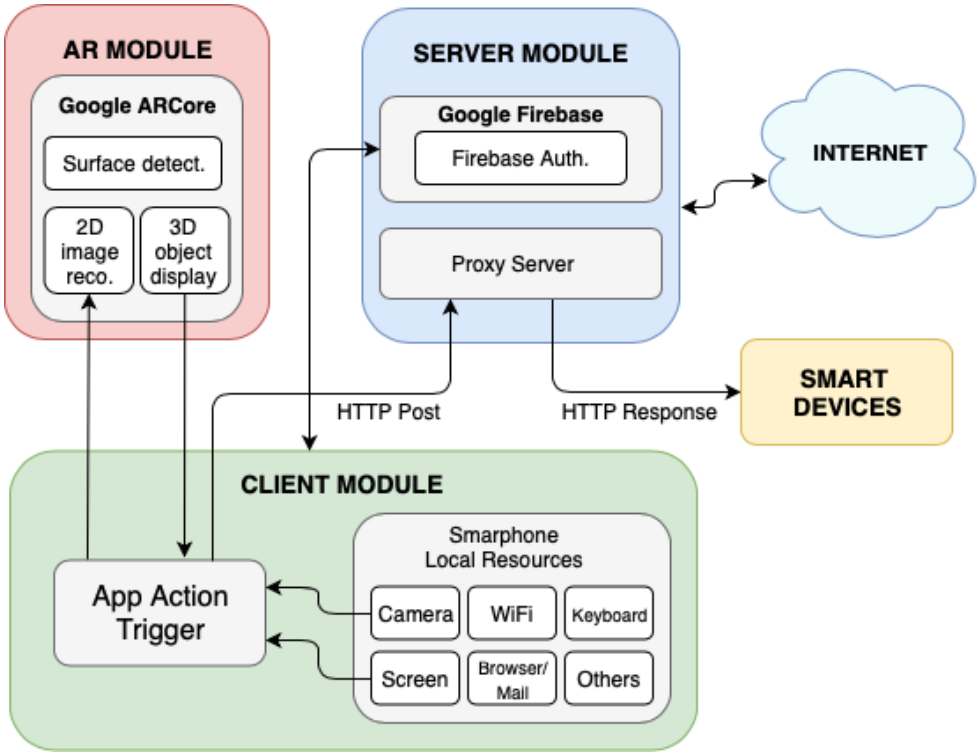

Figure 4.1: Architecture

- ***AR Module:*** This module manages the augmented reality involved in the project. It could be seen as the foundation on which the system is based. The AR module is composed of Google ARCore and interacts with the client module.

- ***Server Module:*** The server module includes Google Firebase and the server which recibes the HTTP POST requests from the client module.

- ***Client Module:*** The client module is based on the user. It is formed of the user's smartphone local resources and the android mobile application.

4.3 AR Module

As stated in the overview, the augmented reality module is the key component of the architecture that conforms the android application. It is responsible of giving the user the possibility of interacting with AR objects in the surrounding environment thanks to Google ARCore, the single component of this module.

4.3.1 Google ARCore

Google ARCore could be seen as the engine that enhances the main functionality of the application: augmented reality. It interelates with the client module when the user triggers an action through the android mobile application. Whilst the user interacts with its environment through the camera of a compatible smartphone, Google ARCore makes it possible to recognize surfaces or two-dimensional images, portray augmented reality objects through the camera of the user and display them on its screen as a response. To graphically describe the process in which this module takes part, the figure 4.2 has been elaborated showing the steps that occur after the user interacts with the application triggering the augmented reality experience.

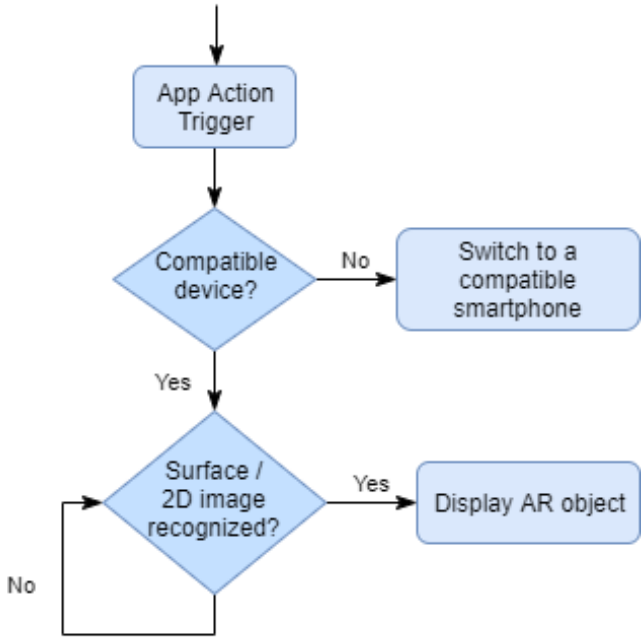

Figure 4.2: Decision diagram of the augmented reality related to the app

4.4 Server Module

4.4.1 Google Firebase

Google Firebase is the platform chosen to carry out the process of authentication required to secure the application from users that should not be allowed to modify the parameters of the smart objects. As its own name states, Firebase was acquired by Google in 2014

and offers several possibilities to enhance our apps such as data hosting and synchronizing, cloud storage or being a real time database among others. Considering all the options, I decided to use it to manage the authentication process, as the platform gives the chance to store the users alongside with their passwords and validate in a simple and safe way. To allow the application to use this system, Google Firebase must be integrated with it. Once it is paired with the app, the users can be created from the platform itself or through a registration view implemented in the application which asks just for a username and a password.

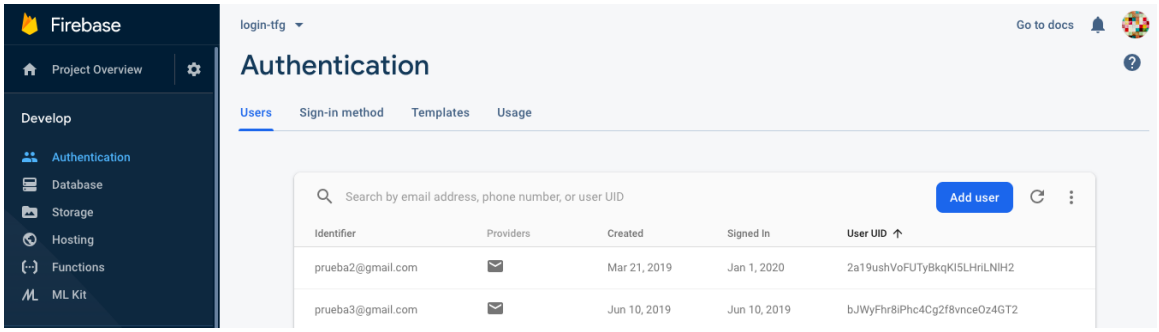

Figure 4.3: Google Firebase console

4.4.2 Proxy Server

The server module is also conformed of other component: the proxy server in charge of managing the HTTP POST [14]. As it is stated in chapter 3, two of the use cases need to interact with a server after triggering the action of pressing an augmented reality button by the user:

- **Light Actions Use Case:** different HTTP POST requests can be sent to the server depending on which button of the set is pressed. As described in the Philips Hue developer guide [16], once a new user has been created, the light parameters can be modified by just sending HTTP POST requests to the server changing the values required. The JSON of the light can be tested in the CLIP debugger tool introduced in the guide. To access the debugger tool, we must know the IP address of the bridge and browse a url with the following structure: "https://**[bridge ip address]**/debug/clip.html". In the resulting JSON described in figure 4.5, we can check the state of the parameters for the light "1", the one corresponding to the Philips Hue Go Lamp used for this project.

  As we can see, the relevant parameters for this use case are "on", which can be modified to true or false depending on the lamp being on or off, and "open door" adapting the

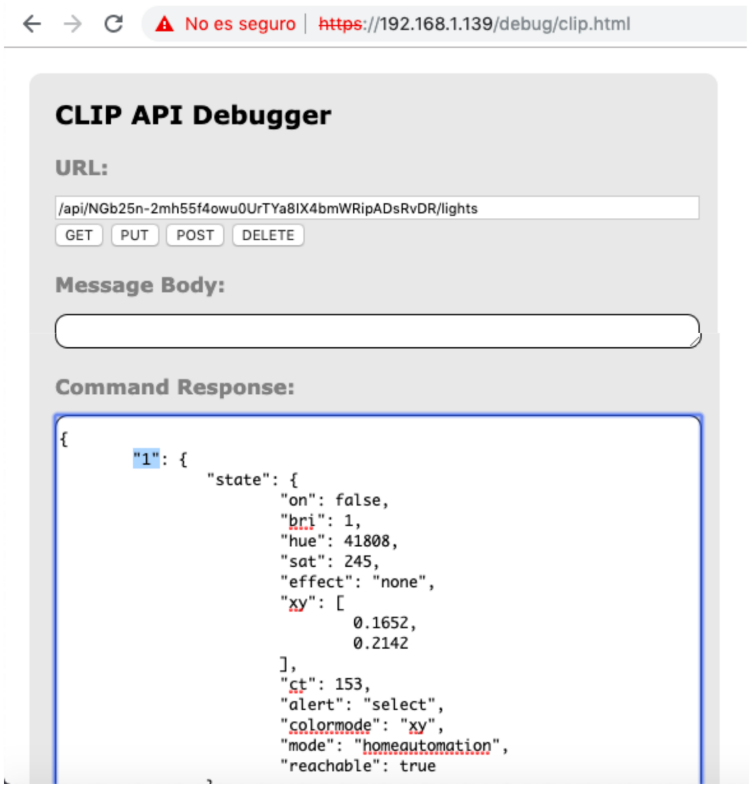

Figure 4.4: CLIP API Debugger tool and light JSON

format RGB, which can vary from 0 to 255 composing the different range of colors. For the set of buttons, the colors chosen are red (R=255, G=0, B=0), green (R=0, G=255, B=0), blue (R=0, G=0, B=255) and white (R=1, G=1, B=1).

```
ArrayList<Double> color_rojo = parseColors(255,0,0);
String json_rojo = "{\"xy\":"+ color_rojo +"}";
RequestBody body3 = RequestBody.create(json_rojo, MEDIA_TYPE);
```

- **Door Opening Use Case:** the HTTP POST request is triggered when the user presses the augmented reality button "open door". As explained in the previous case, the door parameters can be modified by sending the HTTP POST request to the server varying the aimed values to open the door. To achieve so, the parameter that should be modified in its respective JSON is the password "pass" which is sent along with its value which is not going to be stated in this chapter due to security reasons.

4.5 Client Module

4.5.1 Smartphone Local Resources

The smartphone local resources also take part in the architecture of the project as a part of the client module. They are conformed of a mix of hardware and software resources present in the user smartphone that allows the user to run the app correctly.

Among the most remarkable resources are the camera, the WiFi connection, the keyboard, the screen and the browser or email. The camera is strictly necessary to allow the user see the environment through it on the screen. The WiFi connection is requested for the authentication process in which the keyboard is also required, but could be replaced by data connection, and is compulsory for the door opening use case and light actions use case in which the user must be connected to the WiFi of the smart office to accomplish the actions.

4.5.2 Android Mobile Application

As it has been made clear, the main way for the user to interact with the augmented reality experience in the smart office is by downloading the application APK in a compatible smartphone. This app is part of the client module, and could be chopped into the different activities it is composed of. There are six different activities, depicted in figure 4.5:

- ***Login Activity:*** First activity that emerges when the application is initialized. It allows the user to introduce its username and password to log in.
- ***Index Activity:*** Main activity of the application from which the augmented reality experience can be accessed pressing the floating button (AR+), unfold the lateral menu (navigation drawer) to navigate to the rest of the activities or log out in the upper right menu.
- ***AR Activity:*** Activity in which the user can experience the augmented reality through the camera, scanning 2D images to generate the AR objects.
- ***AR Video Activity:*** This activity lets the user experience augmented reality through the camera as well, but this time recognizing flat surfaces on which the augmented reality video can be portrayed by the action of a tap.
- ***Info Activities:*** Pair of activities which offer info about Google ARCore and AR in scroll mode.

- ***Web Activity:*** Activity that emerges after pressing one of the buttons of the social media in the social actions use case explained in section 3.2.2.5.

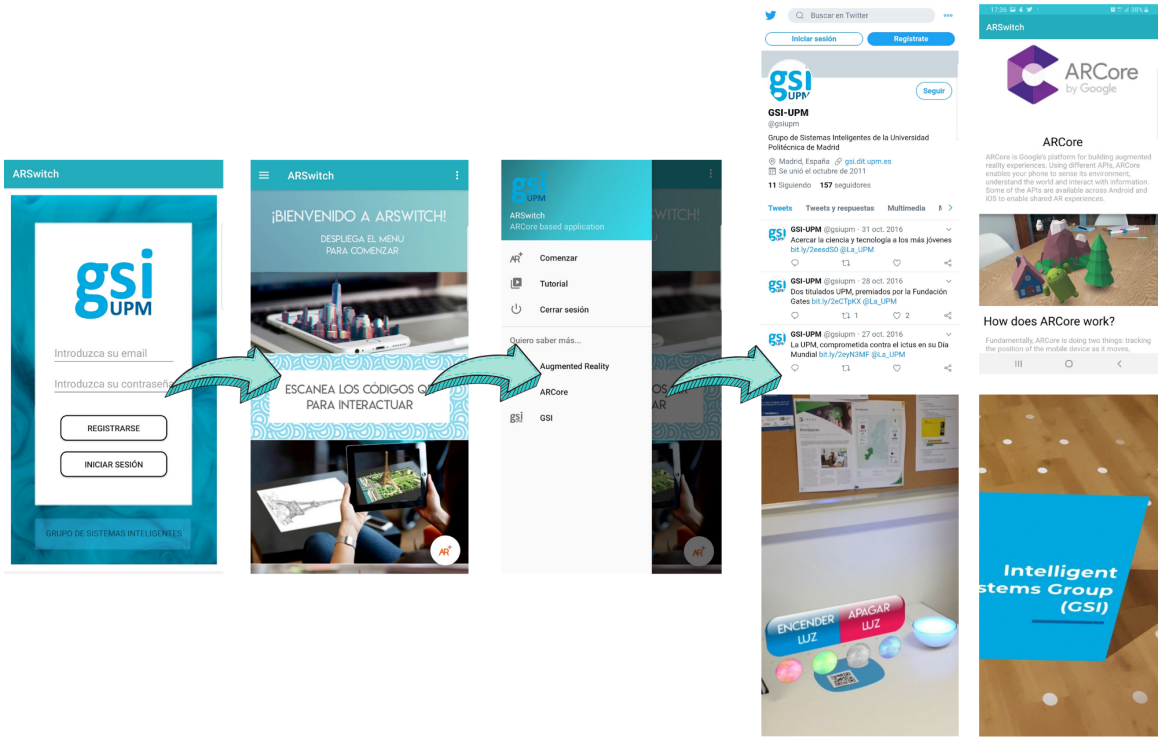

Figure 4.5: Android Application Activities

* The order of appearance from left to right and up to down of the views in figure 4.5 is the following: login activity, index activity, navigation drawer of the index activity, web activity, info activity, AR activity and AR video activity.

4.6 Smart devices

The smart devices are other of the main components of the architecture of this project. They are basically the gadgets present in the smart office and available at the moment of executing the project, that are set to interact with the augmented reality of the application. They are two: the smart office's door and the Philips Hue Go Lamp mentioned in section 2.5.

CHAPTER 5

Case study

5.1 Introduction

Due to the necessity to implement different use cases to execute the smart office required interactions, several augmented reality techniques have been used to define each of the use cases implemented in the Android application. Opening the door of the smart office, varying some of the light parameters, portraying an augmented reality video or an information display have been the use cases selected to carry out during this project development.

Apart from AR, other common characteristic the six main use cases share is the scenario where they take place which is a smart office and it has been tested in the one of the Intelligent Systems Group, in the Escuela Técnica Superior de Ingenieros de Telecomunicación of the Universidad Politécnica de Madrid.

5.2 Welcome video use case

The main aim of this use case is to portray an augmented reality video integrated in the environment of a smart office. It can be seen through the mobile phone of the user and

stared from different angles of the room. The video can play infinite options, from a video introducing a company and its different areas to a random Youtube entertainment video.

As part of the augmented reality experience offered through the use of the application, the user is able to select a 'Tutorial' option in the navigation drawer once the user name and password have been validated after the authentication process mentioned in the previous sections. Once this has been done, the augmented reality experience is triggered and a symbol of a smartphone handled in a waving hand invites the user to explore the environment throughout its camera.

By doing this, flat surfaces are recognized by the application due to ARCore integration using feature points. Surfaces without texture or plain coloured should be avoided in order to achieve a correct plane recognition conformed of an equally spaced dots matrix. If the user presses this matrix, a new AR node is created with the video selected anchored above it, as figure 5.1 shows.

Figure 5.1: Augmented reality video

The actors involved in this use case are the following:

- **User** Considered to be the principal actor of the project, it is in charge of using the application handling a smartphone, logging in and triggering the 'Tutorial' augmented reality experience. In this case, the app lets the user go though an authentication process and select the 'Tutorial' option in the navigation drawer. Once it is immersed in the AR experience, the user is able to recognize the surfaces in the smart office and tap anywhere on the to let the AR video start.

- **Google ARCore [3]** Secondary actor in charge of managing every aspect related to augmented reality in this use case, such as surface recognition, node creation and anchoring or AR video displaying.

5.2.1 Details of augmented reality implementation

In this use case, rendering a video in augmented reality is needed. The following aspects have to be taken in to account to achieve this goal:

- A **ModelRenderable** object [10] is declared in the app and serves as a base for the video to be played, rendering a three dimensional model ("videoscreen.sfb") by attaching it to a node. Namely, it could be seen as the surface in which the images are projected.

- To play the video, a **MediaPlayer** object is declared, assigning it the video chosen ("presentacion gsi.mp4") and avoiding it to be infinitely played as stated below.

```
private ModelRenderable videoRenderable;

@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.ar_video);

    ExternalTexture texture = new ExternalTexture();

    MediaPlayer mediaPlayer = MediaPlayer.create(this, R.raw.presentacion_gsi);
    mediaPlayer.setSurface(texture.getSurface());
    mediaPlayer.setLooping(false);

    ModelRenderable
            .builder()
            .setSource(this, R.raw.video_screen)
            .build()
            .thenAccept(modelRenderable -> {
                videoRenderable = modelRenderable;
                videoRenderable.getMaterial().setExternalTexture("videoTexture", texture);
                videoRenderable.getMaterial().setFloat4("keyColor", new Color(0.1f, 1.0f, 0.1f));
            });
```

- As the augmented reality video appears when the surface matrix is tapped by the use, a tap listener method call must be included also included.

```
ArFragment arFragment = (ArFragment)
getSupportFragmentManager().findFragmentById(R.id.fragment);

arFragment.setOnTapArPlaneListener((hitResult, plane, motionEvent) -> {
    AnchorNode anchorNode = new AnchorNode(hitResult.createAnchor());
```

- As depicted, an **ExternalTexture** object [11] is also used to create the surface to be displayed in the environment rendered by SceneForm [12], which is a high level scene API used to render realistic 3D scenes in augmented reality apps

```
if (!mediaPlayer.isPlaying()){
    mediaPlayer.start();

    texture.getSurfaceTexture().setOnFrameAvailableListener(surfaceTexture -> {
    anchorNode.setRenderable(videoRenderable);
    texture.getSurfaceTexture().setOnFrameAvailableListener(null);

    });
```

5.3 Door opening use case

In this case, the main objective is to create an augmented reality button that makes the user available to interact with the electronic system of a smart door's lock to open it. For this purpose, the AR button emerges using the augmented images technique and it can be actioned by tapping the screen of the smartphone just on it.

The Intelligent Group System (GSI) smart office has a door protected by a card authentication system consisting in approaching the card itself to a sensor placed next to the door. Apart from this method, the door can be opened by sending post requests with the password specified to the url 'http://mozart.gsi.dit.upm.es/'. The latter can be triggered by clicking an augmented reality button portrayed for this purpose with the message "open door".

The actors involved in this use case are the following:

- **User** In this use case, the user as a principal actor is able to interact with an augmented reality button in order to open the door as it is described in the picture above. The button emerges from a two dimensional key image, a QR code situated in a GSI panel. The app lets the user navigate into the augmented reality experience after the authentication process by clicking a rounded button in the main view or pressing the first option in the navigation drawer menu.
- **Google ARCore [3]** Secondary actor in charge of managing every aspect related to augmented reality. This time, the application will detect two dimensional images and will set them as anchored nodes for the model, in this case the augmented reality button.
- **Proxy Server** Secondary actor whose function is a attending the http post requests sent when the user clicks the AR button situated next to door.

Figure 5.2: GSI's door environment and AR button

5.3.1 Details of augmented reality implementation

As it has been described in the introduction, it is needed an augmented reality button to interact with the door. This button is shown by using the AR technique of augmented images. In other words, it consist in placing a 2D image, a QR code in this case, which is recognized by the camera of the Android device used and virtually scanned to generate a button floating above it.

In this case, the button consists of a 2D impression created by designing a layout with a button on it with the message "open door", and rendering it by choosing the layout view and the context (its proper node) as it follows:

```
CompletableFuture<ViewRenderable> viewCompFuture =
        ViewRenderable.builder().setView(context, R.layout.ar_button2).build();
```

5.3.2 Post Request

In order to execute the door opening petition, an http post request is stated to the server hosted in the url 'http://mozart.gsi.dit.upm.es/'. To carry it out, the next requirements have been taken into account:

- The okhttp3 classes required for the post request must be imported. These imports could be summed up by just writing 'import okhttp3.*;'. However, as the classes to use are known, writing just the necessary ones makes the application lighter.

```
import okhttp3.Call;
import okhttp3.Callback;
import okhttp3.MediaType;
import okhttp3.MultipartBody;
import okhttp3.OkHttpClient;
import okhttp3.Request;
import okhttp3.RequestBody;
import okhttp3.Response;
```

- The client and url objects need to be set.

```
private final OkHttpClient client;
private String Url = "http://mozart.gsi.dit.upm.es/";
```

- The onClick( ) method has to contain the HTTP POST request logic. To make the post request, the request body (RequestBody) and request (Request) must be stated so that the client can make a call as it follows: client.newCall(request).enqueue(new Callback() ...

```
RequestBody body = new MultipartBody.Builder()
        .setType(MultipartBody.FORM)
        .addFormDataPart("pass", "*****")
        .build();

Request request = new Request.Builder()
        .post(body)
        .url(Url)
        .build();


client.newCall(request).enqueue(new Callback() { … }
```

* Note that the password has been censored ("*****") due to security reasons.

5.4 Light actions use case

The main aim of this use case is to control the smart light fixture of a smart office by interacting with AR. As several new gadgets related to this area have flooded the market relatively recently, from light bulbs to more complex illumination systems, this use case has been a priority due to its utility.

Aside from assigning augmented reality attributes to create a button to open the door, other set of buttons have been implemented to modify the attributes of a Philips intelligent lamp located in the smart office.

Figure 5.3: GSI's light environment and respective set of AR buttons

The actors involved in this use case are the following:

- **User** Principal actor of this use case which has a similar functionality as in the previous use case. The main difference is that this time the user is no only going to be able to click one single button, but a set of different buttons to turn on/off the light and change its coloration. The application lets the user navigate into the augmented reality experience after the authentication process by clicking a rounded button in the main view or pressing the first option in the navigation drawer menu.
- **Google ARCore [3]** Secondary actor in charge of managing every aspect related to augmented reality. In this case, the application detects two dimensional images and

will set them as anchored nodes for the model made of a set of buttons

- **Proxy Server** Secondary actor whose function is a attending the http post requests sent when the user clicks on one of the augmented reality buttons that appear next to the Philips Hue Light.

5.4.1 Details of augmented reality implementation

As depicted in the door opening case, the technology used is augmented images, which consist in placing a 2D image to be recognized by the camera of the Android device used and virtually scanned to generate a button floating above it.

This time, not only one button is be able to interact with. By scanning the QR code, a pair of buttons to switch on and off the light ('turn on lamp' and 'turn off lamp') along with other four buttons underneath to change its coloration appear.

5.4.2 Post request

In contradistinction to the door opening use case, to change the light parameters post request petition is stated to the server hosted in the url "http://192.168.1.191 /api/-naRLisQGuCawylPMxplFMwjX2PQBHAbcvfQAV9LS /lights/4/state". To carry it out, the next requirements have been taken into account:

- The okhttp3 classes required for the post request must be imported as in the previous use case.

- The client and url objects need to be set. This time, as the url has different attributes that can be changed, it is stated as it follows:

```
private final OkHttpClient client;
private String Ip = "192.168.1.139";
private String token = "NGb25n-2mh55f4owu0UrTYa8IX4bmWRipADsRvDR";
private String light = "1";
private String Url = "http://" + Ip + "/api/" + token + "/lights/"+light+"/state";
```

  Regarding the code, the attributes considered to build up the url are the IP direction where the own Philips light is located, a token to gain access to modify the parameters and the number of the light, which for this lamp is set to be 1.

- The media type should parse JSON, as button triggers a JSON modification when clicked.

```
private MediaType MEDIA_TYPE = MediaType.parse("application/json");
```

- In this case, as the augmented reality object is formed by various buttons allowing to turn on and off the light and changing its color, the onClick method will vary depending on the button clicked.

```
String json_encender = "{\"on\":true}";
RequestBody body = RequestBody.create(json_encender, MEDIA_TYPE);

Request request = new Request.Builder()
        .put(body)
        .url(Url)
        .build();

client.newCall(request).enqueue(new Callback() { … }
```

5.4.3 Information display use case

In this use case, the result desired was to create information displays that would be helpful to recognize positions and areas inside a smart office. As the one of the Intelligent Systems Group has different desks were students and staff work, we have considered to include information displays in augmented reality showing information about each of them. Due to confidential reasons, names and positions showcased are fictitious.

This use case offers many possibilities as well and other usages could have been implemented, such as signboards located in the door of any room or emerging augmented reality bubbles with intructions for the kettle usage near it.

The actors involved in this use case are the following:

- **User** In this case, the user is responsible of pointing through the camera of its smartphone to the different QR codes located on the desk of each person. The application lets the user navigate into the augmented reality experience after the authentication process by clicking a rounded button in the main view or pressing the first option in the navigation drawer menu.

- **Google ARCore [3]** Secondary actor in charge of managing every aspect related to augmented reality. This time, it detects an specific augmented image and will set the information display floating above it.

Figure 5.4: Example of an information display

5.4.4 Details of augmented reality implementation

The AR technique involved in this use case is augmented images, as in the two previous use cases. Scanning a QR code in the smart office makes the information display appear the same way the buttons do in the cases above.

5.4.5 Social actions use case

The main purpose of this use case is to create a social hub, in which a set of buttons corresponding to the social media or webs of a company are displayed. The user of the application is able to navigate through the different options by clicking in one button of the available. For this project, the actions chosen have been mainly social: Twitter, Youtube, email and GSI's website.

The actors involved in this use case are the same as in the light action or door opening use case: user as principal actor in charge of controlling the app and Google ARCore and a proxy server as secondary actors.

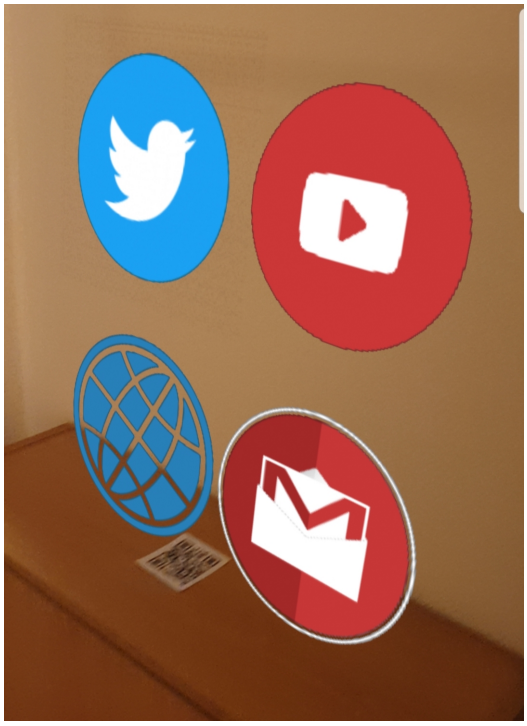

Figure 5.5: Social hub

5.4.6 Details of augmented reality implementation

The operating dynamics of the augmented reality related to this use case is exactly the same as the other use cases in which AR buttons are related. In this instance, the augmented reality object is formed of a set of buttons, not only one. Depending on the button tapped by the user during the AR experience, different web activities can be triggered as every button is associated to a different url.

5.4.7 3D Model display use case

Finally, as an extra use case, exploring the different possibilities that augmented reality with ARCore offers led to AR three dimensional models. This option makes infinite possibilities feasible as well, as any imaginable figure or model could be represented. For this project, as the application has been tested specifically in the smart office of the Intelligent Systems Group, the 3D model selected is the GSI logo shown in section 2.6.

The actors involved in this use case are the same as in the information display use case, but this time, as there is only one QR code to generate the GSI logo, the user must point precisely at it with the camera of the smartphone used.

5.4.8 Details of augmented reality implementation

This time, the augmented reality object that appears interacting with the augmented image is a three dimensional model, not an Android view such as in the previous cases. As described in section 2.6.1, to include the GSI 3D logo, three data files must be imported to the Android Studio project: '.obj', '.png' and '.mtl'. After importing the sceneform asset, the figure is available to be used in the resources raw folder of the project.

5.4.9 Conclusions

It has been made clear that AR is the cornerstone that sustains the whole project. As stated in this chapter, two different augmented reality techniques have been implemented for the use cases development:

- **Augmented images:** recognizing two dimensional key images from which augmented reality emerges. In this project, the 2D images used are QR codes for simplicity, but any picture could have been used, from a poster of the Intelligent Systems Group to a newspaper cover placed on a table.
- **Surfaces recognition and object placement:** any flat surface aimed with the smartphone's camera can be recognized and set as planes for the application. On them, augmented reality objects can be placed just with a single click.

CHAPTER 6

Conclusions and future work

In this chapter, the process carried out to develop the project and the main impressions are summarized. The conclusions extracted are explained along with the achieved goals to end up proposing some future work lines that could be followed.

6.1 Conclusions

In order to conclude this project, the main concepts this project comprises will be summed up. The main idea of the project was to use the Android Studio development ecosystem to implement an augmented reality (AR) application for Android devices using Google ARCore. The app should be able to interact with the smart objects available in the Intelligent Systems Group (GSI) smart office.

As augmented reality was a new sector for me to explore, a deep research on the possibilities available was done in order to start understanding how it could be implemented to reach the goals and bring to life visually catching interactions that actually added value to the automations already implemented. One of the first ideas was to give the chance to the GSI members to visually modify the kanban table available in the smart office. However, this option was finally discarded because Google ARCore did not have the resources to

implement what it was intended to do.

Finally, the idea of interacting with intelligent objects such as the door or a Philips intelligent light was proposed. As the majority of the work with augmented images was already done, it turned out into augmented reality buttons appearing from scanning 2D images next to the objects. This buttons gave the chance to interact with the intelligent objects, fulfilling the main goal of the project. Additionally, apart from the AR interactive buttons, other cases of augmented reality were implemented such as information displays emerging from 2D images, 3D floating models like the own GSI logo or AR videos used for portraying a tutorial or a GSI Youtube video by pressing on the augmented reality scene.

During the development of the project several problems have been encountered mainly due to the lack of information on developing augmented reality applications on Android available yet. The majority of resources found were excessively simple and not suitable for what it was intended. Besides, as different augmented reality techniques were used, combining each of their nodes or the different sceneforms brought lots of errors that could only be solved by empiric method, as barely no useful information to correct them was found online. This fact has been one of the toughest aspects to complete the project.

6.2 Achieved goals

As some of the achieved goals have been introduced in the previous section, they can be summed up as it follows:

Development of an Android application to interact with intelligent objects available in a smart office via augmented reality, such as the one of the Intelligent Systems Group (GSI).

Deep research on Google ARCore techniques for augmented reality Android applications.

Three different runtime stages: user register and login, navigation through the app and augmented reality interaction.

Interface based on Google Firebase for registration and login purposes.

Intelligent objects dually protected: wifi password and authentication.

6.3 Future work

As augmented reality is a whole new sector to discover and in constant change, multiple improvements could be made, no only by thinking about new ways to implement AR in the smart office but with the new possibilities that future developments of AR technology will bring. The augmented reality use cases described in the project could be more in case of not having a time limitation. However, future developments that could be taken into account are considered below:

- **Interaction with more smart objects**: apart from the two intelligent objects selected because of its availability at the moment of executing the project, more objects could be brought to augmented reality interaction by creating new nodes for the augmented images and their respective AR buttons.
- **Gamification**: a system in which the user earn points each time it connects or fulfills a series of milestones could be developed, so that the user with the highest amount of points after a certain period of time would get a reward.
- **Ingenuity is the limit**: augmented reality offers a wide range of possibilities once its performance has been understood. It is up to imagine new 3D models, videos or other uses suitable for any smart office.

APPENDIX A

Impact of this project

The principal aim of this project is to develop an augmented reality application to interact with the smart objects integrated in a smart office. This app is not only valid for the Intelligent Systems Group, but for any company that would like to integrate smart objects and control them via AR. In this section, the main impacts related to the project development and application utilization are described.

A.1 Social impact

The social impact could be seen as the footprint that this project has on its surrounding community. Augmented reality can cause a great impact in society, making daily tasks easier and enhancing virtual interaction with the environment.

Definitely, this project eases interaction with the smart objects present in the smart office, and gives extra visual information of the individuals sitting on each desk.

Moreover, any imaginable social interaction with the environment could be yet implemented as augmented reality establishes no limits apart from the smart objects available nowadays in the market.

A.2 Environmental impact

This section summarizes the environmental impact caused by this project. During its development, the use cases described do not cause any noticeable damage or amelioration in the environment state except from electrical consumption, as an excessive usage of the laptop and the mobile phone to interact with augmented reality has been carried out.

Furthermore, environmental friendly use cases could be implemented for any company interested in the area. Heating is one of the main pollution causatives in metropolitan areas, as an example. The application could implement a new use case in which a set of buttons could control a smart thermostat: increasing or decreasing a Celsius degree by tapping an augmented reality plus or minus button from our working desk.

With a responsible usage of this action, diminishing the thermostat temperature and keeping it at recommended levels would contribute to environment sustainability.

A.3 Economic impact

In this section, the economic effects that this project results in are summarized.

From a monetary point of view, a company could be benefited of having an augmented reality display of buttons on the desk of every employee. As a use case of this project describes, light could be controlled in order to reduce energy consumption and accordingly, economic expense.

Other use cases such as the thermostat augmented reality controller use case mentioned in the previous section could be also implemented to help companies diminish their economic expenses.

A.4 Ethical impact

The most relevant ethical problem is related to the personal information about the users, confidentiality must be assured. The Google Firebase authentication database used must be managed by a trustworthy person. This person can access the email of every user, the date when the account was created, the date when the user last signed in and the user UID or user identifier. As this information could be delicate, the European General Data Protection Regulation (GDPR) must be followed.

APPENDIX B

Economic budget

In this appendix, the most relevant costs for the project will be gathered and explained. As it has been considered, the budget spent can be divided into three groups: physical resources, human resources and licenses.

B.1 Physical resources

The project has been mainly developed by using a portable computer or laptop and an android mobile phone. The cost of the former is not difficult to determine, but as it devaluates and regarding the average market value, its price can be approximated to 900€. The latter has an approximated price of 650 €, so the total cost of the physical resources is 1550€.

The laptop used is a MacBook Air with the following specifications:

- Operating System: macOS Mojave, 10.14
- CPU: Intel Core i5-5350U
- RAM: 8 GB

- DISK: 128 GB

The mobile phone involved in testing the application is a Samsung Galaxy S9+ with the following specifications:

- Model: SM-G965F
- Operating System: Android Pie, 9.0
- RAM: 6 GB
- DISK: 64 GB

B.2 Human resources

Considering human resources, only a person has been involved in the project development. Supposing this person has no previous experience and no labor qualifications yet, its job has been estimated to be worth 8€ per hour. As the time dedicated for this project has been close to 300 hours, the amount dedicated to human resources in this case would be 2.400€.

However, if the person in charge of this project was considered to have already experience developing augmented reality applications and had already finished the degree, the average salary per year could be up to 25.000€.

B.3 Licenses

During the development of the project, every software utilized has had no cost at all except from Vectary, the web program used to create some of the 3D models comprised in the project. The cost for using its online platform is 9€ per month, an so is the total licenses cost, as it has been used only during the last month.

APPENDIX C

Augmented Images Database and Image Quality Tool

One of the main usages related to augmented reality in this project are augmented images. As described, this technique consist in recognizing two dimensional key images from which augmented reality objects emerge. Due to the existence of different use cases using this AR technique, several key images are needed to be recognized to trigger the actions.

C.1 Augmented Images Database

Key images can be added to the application in two different ways:

- **Individually** adding bitmaps to the database.

```
Bitmap augmentedImageBitmap = loadAugmentedImage();
. . .
augmentedImageDatabase = new AugmentedImageDatabase(session);
augmentedImageDatabase.addImage(image, augmentedImageBitmap);
. . .
config.setAugmentedImageDatabase(augmentedImageDatabase);
```

  As depicted above, the key images can be added directly to the 'augmentedImage-

Database' one by one thanks to the method 'addImage' depending the amount of key images needed. Once they are added to the database, the can be individually load using a loadAugmentedImage for each of them.

```
private Bitmap loadAugmentedImage() {
    try (InputStream is = getAssets().open(image + ".png")) {
        return BitmapFactory.decodeStream(is);
    } catch (IOException e) {
        Log.e(TAG, "IO exception loading augmented image bitmap.", e);
    }
    return null;
}
```

- **Using a pre-built augmented images database** [18] formed of every key image involved in the use cases. This augmented images database can contain over 1000 key images and can be created through the **arcoreimg tool** [19]. This option makes possible a shorter setup time and does not require the augmented images to be included in the APK, just the database. The JPEG or PNG valid images must be added to a directory for the database to be created with the following command: './arcoreimg build-db –input imagesdirectory=/path/to/images –output db-path=/path/to/myimages.imgdb'. As a result, an '.imgdb' file is obtained, containing the images that where located in the selected directory. This file can then be added to the Android Studio project and used as an individual object.

C.2 Checking Augmented Images Validity

When using the augmented images technique, it is important to notice that the pictures used should fulfill a series of requirements. ARCore arcoreimg tool helps developers to distinguish suitable images giving a range from 0 to 100 as a response of the following command: './arcoreimg eval-img –input imagepath=image.jpg'.

This range would be better depending on several aspects of an image: containing many characteristic feature points, avoiding repetitive features, image resolucion above 300x300 pixels or avoiding heavy compression for JPEG images among other aspects.

Finally, and as a suggestion, ARCore recommends using images with a quality greater or equal to 75 in order not to have trouble recognizing key images.

APPENDIX D

Acronyms and Abbreviations

- ***APK*** ——— Android Application Package, format
- ***APP*** ——— Application
- ***AR*** ——— Augmented Reality
- ***GSI*** ——— Intelligent Systems Group
- ***HTTP*** ———- Hypertext Transfer Protocol
- ***IP*** ——— Internet Protocol
- ***JPG*** ——— High quality images compression format
- ***JSON*** ——— JavaScript Object Notation
- ***OBJ*** ——— Geometry definition file, key in 3D graphics applications
- ***MP4*** ——— Multimedia file format which contains video and audio
- ***MR*** ——— Mixed Reality
- ***MTL*** ——— Material Library File, definition of a sequence of materials
- ***PNG*** ——— Portable Network Graphics, compressed graphic format

- ***QR*** ———— Quick Response Barcode
- ***RGB*** ———— Color composition in terms of light primary colors intensity
- ***UML*** ———— Unified Modeling Languaje
- ***URL*** ———— Uniform Resource Locator
- ***VR*** ———— Virtual Reality
- ***WIFI*** ———— Wireless Fidelity, internet connection

Bibliography

[1] Dieter Schmalstieg and Tobias Höllerer. *Augmented Reality, Principles and Practice*. Addison-Wesley, 2016.

[2] José María Ariso (Ed.). *Augmented Reality, Reflections on Its Contribution to Knowledge Formation. Accessed February 2019*. Gruyter, 2016.

[3] Google ARCore. ARCore: Overview an Fundamental Concepts. Accessed February 2019. "https://developers.google.com/ar/discover".

[4] Google ARCore. ARCore: Develop, Android, Enable ARCore. Accessed February 2019. "https://developers.google.com/ar/develop/java/enable-arcore".

[5] Philips Hue. Philips Hue Portable Go Lamp. Accessed December 2019. "https://www2.meethue.com/es-es/p/hue-white-and-color-ambiance-lampara-portatil-go-(ultimo-modelo)/7602031P7".

[6] Vectary. 3D Objects Online Design Tool. Accessed April 2019. "https://www.vectary.com".

[7] Google Firebase. Platform for Web and Mobile Apps Development. Accessed April 2019. "https://firebase.google.com".

[8] José María del Álamo Ramiro. *Asignatura ISST, Tema 3, Apartado 1, Modelado y Diseño de Sistemas*. Departamento de Ingeniería de Sistemas Telemáticos, 2019.

[9] Visual Paradigm. Use Case Analisys. Accessed December 2019. "https://www.visual-paradigm.com/guide/uml-unified-modeling-language/how-to-identify-actors/".

[10] Google ARCore. Model Renderable. Accessed April 2019. "https://developers.google.com/ar/reference/java/sceneform /reference/com/google/ar/sceneform/rendering/ModelRenderable".

[11] Google ARCore. External Texture. Accessed April 2019. "https://developers.google.com/ar/reference/java/sceneform /reference/com/google/ar/sceneform/rendering/ExternalTexture".

[12] Google ARCore. Sceneform. Accessed April 2019. "https://developers.google.com/ar/develop/java/sceneform".

[13] Enrique Sánchez Tolbaños. Design and Development of a Cognitive Computing Empowered Robot Assistant for Semantic Task Automation in Smart Places. Trabajo fin de grado, Universidad Politécnica de Madrid, ETSI Telecomunicación, June 2014.

[14] Visual Paradigm. HTTP POST Method. Accessed May 2019. “https://www.w3schools.com/tags/ref_httpmethods.asp”.

[15] Augment. Augmented Reality and Differences with Other Technologies. Accessed December 2019. “https://www.augment.com/blog/virtual-reality-vs-augmented-reality/”.

[16] Philips. Philips Hue Developer Guide. Accessed June 2019. “https://developers.meethue.com/develop/get-started-2/”.

[17] Resmana Lim Murtinyato Santoso Nemuel Daniel Pah Felix Pasila, Yusak Tanoto. *Proceedings of Second International Conference on Electrical Systems, Technology and Information.* Springer, 2015 (ICESTI 2015).

[18] Google ARCore. Augmented Images Database. Accessed March 2019. “https://developers.google.com/ar/reference/c/group/ augmented-image-database”.

[19] Google ARCore. Arcoreimg Tool. Accessed March 2019. “https://developers.google.com/ar/develop/c/augmented-images/arcoreimg”.

[20] Kishino F. (1994) Milgram, P. A Taxonomy of Mixed Reality Virtual Displays. IEICE Transactions on Information Systems, 77(12).