WEST UNIVERSITY OF TIMIŞOARA

FACULTY OF MATHEMATICS AND COMPUTER SCIENCE

BACHELOR: Computer Science in English

BACHELOR THESIS

SUPERVISOR:
Lect. Dr. Liviu Mafteiu-Scai

GRADUATE:
Sorin-Ionuţ Rosalim

TIMIŞOARA

2023

WEST UNIVERSITY OF TIMIŞOARA

FACULTY OF MATHEMATICS AND COMPUTER SCIENCE

BACHELOR: Computer Science in English

Development of a Mobile Augmented Reality (MAR) application as an interactive tool

SUPERVISOR:
Lect. Dr. Liviu Mafteiu-Scai

GRADUATE:
Sorin-Ionuţ Rosalim

TIMIŞOARA

2023

Abstract

In today's world, technology is everywhere; we use it to communicate, to learn and to discover new things in order to improve our lives. This paper presents a novel approach to the problem of augmenting the real world with digital content for entertaining or didactic purposes. We propose a new method to access digital content using Quick Response (QR) codes, present a proof of concept of our approach, and discuss the potential of Augmented Reality (AR) as an interactive tool for learning and discovering new things. This thesis theme is the development of an Android mobile application called RealityEnhance. The application aims to augment the real world with digital content through the user's smartphone. RealityEnhance gives its users the power of an enhanced perception of their surroundings to discover and interact with digital content.

În lumea de astăzi, tehnologia este peste tot; o folosim pentru a comunica, pentru a învăța și pentru a descoperi lucruri noi pentru a ne îmbunătăți viața. Această lucrare prezintă o abordare nouă a problemei îmbunătăţirii lumii reale cu conținut digital în scopuri de divertisment sau didactice. Propunem o nouă metodă de accesare a conținutului digital folosind coduri QR, prezentăm o dovadă de concept a abordării noastre și discutăm potențialul realităţii augumentate ca instrument interactiv de învățare și descoperire de lucruri noi. Urmărim dezvoltarea unei aplicații mobile pentru Android numită RealityEnhance. Aplicația își propune să îmbunatăţească lumea reală cu conținut digital prin intermediul smartphone-ului utilizatorului. RealityEnhance oferă utilizatorilor săi puterea unei percepții îmbunătățite asupra mediului înconjurător pentru a descoperi și a interacționa cu conținut digital.

TABLE OF CONTENTS

List of Figures

List of Abbreviations

Chapter 1

Introduction

1.1 Project Theme

Augmented Reality is a technology combining the virtual and physical worlds, allowing digital content to be added on top of the physical world. AR relies on device sensors to detect the users' surroundings, position and orientation. QR codes are a type of barcode; they are predominantly used to store information such as text, links, and images and are easy to share.

By combining these two technologies, we can create a new way of accessing digital content and interact with newly accessed information in the real world. This technology gives users the power of enhanced perception while accessing digital content that can be used for entertainment or didactic purposes.

1.2 Theme Relevance

The topic of AR is relevant because it can be implemented and used in a multitude of domains. This technology is already used in the following domains:

- 3-dimensional (3D) printing - AR is used to visualise the 3D model of the object that will be printed.
- Architecture - ARis used to visualise the 3D model of a building or a house.
- Automotive - AR is used for navigation and to display map information.
- Publicity - AR is used to lunch novel marketing campaigns and interact with digital content.

Augmented reality is an interactive experience that recently became more accessible to the world with the help of the smartphone. Most smartphones have the necessary hardware to run AR applications. This AR applications give their users a new perspective on the digital content that they are accessing.

QR codes are two-dimensional matrix barcodes that are easily scanned by a smartphone camera. They were first used for tracking parts in vehicle manufacturing. Nowadays, they are used in a wide variety of domains, such as:

- Virtual stores - QR codes are used to display information about the products.
- Online payments - QR codes are used to store payment information.
- WI-FI access - QR codes are used to store WI-FI access information.
- Augmented reality - QR codes are used in some AR applications to determine the positions of objects in 3D space.

1.3 Project Aim

The AR topic was chosen because this technology has many applications in different fields. The immersive experiences of AR allow people to do interactive activities.

The primary objective is to develop a highly functional mobile application that utilises AR and allows users to scan QR codes. Then using the available models in the Augmented Reality (AR) mode to move, resize, rotate and interact with them.

The application is developed for the Android platform. To accomplish this, Android Studio integrated development environment (IDE) was utilized alongside Google's ARCore Software Development Kit (SDK), which provides powerful augmented reality capabilities. In order to store the 3D models, we make use of the Google Buckets service, which offers a reliable and easy-to-use storage solution. This application has a user-friendly interface that is both simple and intuitive, ensuring that users can easily navigate and interact with all the features.

One of the key highlights of this application is its interactive functionality for some 3D models. Users can trigger engaging animations or explore different states of the model by simply touching a button. This innovative approach offers users a fresh way to access digital content and actively participate in the learning and discovery process. It opens up exciting new possibilities for users to create, learn and explore.

The application will undergo rigorous testing on multiple Android devices to ensure a seamless experience for all users. This will help identify and address any compatibility issues that may arise. Once the application has been thoroughly tested, it will be published on the Google Play Store, making it available to a broad audience.

1.4 Author's contribution

I have utilized my knowledge and expertise to develop a mobile application that incorporates the Google ARCore SDK, a QR code scanner and the Google Buckets service. Through the integration of these technologies, I have created a novel approach to accessing and interacting with digital content. This application empowers users with an augmented perception of their surroundings, enabling them to engage with digital content using their smartphones.

The application is able to load the interaction module specific for each 3D model. The interaction module will be a simple animation or a different state of the 3D model. For example, if the 3D model is a LEGO car, the interaction module will be a simple guide for assembling the LEGO car. If the model is related to a geometrical shape, related to the school geometry curriculum, the interaction module will display a different perspective of the 3D model; for example, if the 3D model is a cube, the interaction module will display only the cube's edges.

Chapter 2

Related Work

2.1 Related writings

Deeper Learning With QR Codes and Augmented Reality: A Scannable Solution for Your Classroom

In the realm of education, the integration of technology has revolutionized teaching and learning processes. Drawing inspiration from Monica Burns' book "Deeper Learning With QR Codes and Augmented Reality: A Scannable Solution for Your Classroom", exploring how AR and QR codes can enhance learning experiences and engage students in interactive educational content.

Augmented Reality (AR) offers a unique opportunity to create immersive and interactive learning environments. By overlaying digital content onto real-world objects, AR provides students with an interactive tool for learning. As Monica Burns highlights, AR can be likened to a "window to interactive learning" [3]. Students can unlock a wealth of three-dimensional models, virtual objects, and multimedia resources through Quick Response (QR) codes. By scanning these QR codes using smartphones or tablets, students' surroundings are enhanced with digital content.

Educators worldwide have embraced AR as a powerful tool for instructional design. Teachers can seamlessly integrate AR into their lessons by leveraging AR apps and printing corresponding trigger images. For instance, using an Augmented Reality (AR), students can explore the intricacies of the human body by scanning large posters of anatomical models. As Burns emphasizes, using AR captivates students' attention and provides a unique opportunity to interact with and deepen their understanding of content [3].

Unleashing digital resources alongside AR, QR codes has become an efficient way to connect physical and digital resources. By scanning these codes using mobile devices, students gain instant access to a wealth of online content. With a simple scan, QR codes can link students to websites, videos, interactive quizzes, and other digital resources, expanding their learning beyond the confines of traditional materials.

QR codes have become a versatile tool in the classroom, empowering teachers to

deliver digital content seamlessly. Educators can create QR codes to supplement lessons, provide additional resources, or foster independent research. Students can embark on self-directed learning journeys by incorporating QR codes into lesson plans, exploring personalized content, and engaging in collaborative projects. The ease of use and immediate access to digital resources make QR codes invaluable in modern classrooms.

Integrating AR and QR codes in education holds tremendous potential for transforming learning experiences. Educators can foster engagement, deepen understanding, and inspire lifelong learners by immersing students in interactive virtual worlds and providing access to vast digital resources. As we embrace Monica Burns' notion of "tasks before apps," we must approach these technologies with clear pedagogical objectives [3]. By strategically integrating AR and QR codes, we can create meaningful and impactful learning experiences that prepare students for success in a technology-driven world.

2.2 Related apps

Snapchat

Regarding AR applications, Snapchat [10] stands out as a platform that pioneers the use of AR and revolutionises how we create, explore, and play with digital content. Harnessing the power of their AR module and Snap AR [9] enables users worldwide to scan their surroundings and discover valuable information. With just a tap on the Camera screen, the AR Bar unfolds.

What sets Snapchat AR apart is its commitment to accessibility. The introduction of the Web Lens Builder and Lens Studio tools has democratised the creation of AR Lenses, making it easier for aspiring creators to bring their visions to life. Moreover, Snapchat has embraced the incorporation of real-world physics and real-time data integrations, elevating the realism and immersion of their AR experiences to new heights.

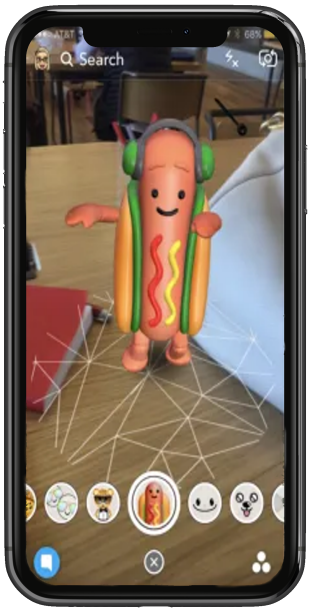


Figure 2.1: Snapchat AR

Snapchat AR brings a fantastic array of special Lenses, each presenting a unique and captivating visual journey. With a plethora of playful animations and informative overlays, the creative possibilities are boundless. Moreover, it boasts a diverse collection of filters designed to enhance the user's appearance or transport them into an array of different characters. What's more, these filters are regularly updated, guaranteeing users a constant stream of exciting options.

However, it's important to acknowledge that Snapchat AR does have its limitations. A consistent internet connection is imper-

ative to browse for new lenses and filters, which may prove challenging in certain circumstances. Additionally, while Snapchat AR offers a compelling experience, it needs to improve in terms of functionality when compared to more specialised AR applications.

Pokemon Go

In mobile gaming, Pokemon Go has become a global sensation by leveraging the power of AR. The game's AR+[8] mode takes the concept of AR to new heights by seamlessly merging the Pokemon universe with the player's real-world surroundings. Through this innovative approach, Pokemon come alive and appear anchored to the user's environment, providing exciting opportunities for an interactive gameplay experience.

In AR+ mode, Pokemon Go brings an added layer of realism to the gameplay experience. Players can approach, walk toward, or even move around these digital creatures by fixing Pokemon to specific points in the real world, as shown in Figure 2.2. The Pokemon in AR+ mode possess an uncanny awareness of the player's proximity and movement, requiring strategic and cautious approaches to maximise successful encounters.

Figure 2.2: PokemonGO

In contrast, the AR mode in Pokemon Go detaches Pokemon from the real-world environment, lacking the anchor points and awareness of the player's location and movement found in AR+ mode. The enhanced AR features in Pokemon offer players a captivating and dynamic experience. As they explore their surroundings, Pokemon seemingly inhabit their world, creating magical moments for excellent throws and captivating photos. The fusion of AR with the beloved Pokemon franchise has captivated millions of players worldwide.

Like Snapchat AR, Pokemon Go relies on a consistent internet connection to enable the AR features. This requirement may pose challenges in situations where a stable connection is unavailable. Additionally, the GPS needs to be enabled to play Pokemon Go, which may drain the battery life of mobile devices. Despite these limitations, Pokemon Go has become a global phenomenon, demonstrating the power of AR in mobile gaming.

Chapter 3

Application description

3.1 Architecture

RealityEnhance was developed using Android Studio Flamingo [5], which is an IDE for Android development. The application was developed using Java[11], which is a general-purpose programming language.

For the AR functionality, the application uses the ARCore software development kit (SDK). For the QR code scanner, the application uses a library called code-scanner [2], and for storing the models, the application uses Google Bucket Storage. As shown in Figure 3.1, the architecture of the application consists of Android Package Kit file format (APK) that runs on the user's smartphone, which communicates with Google Cloud Storage.

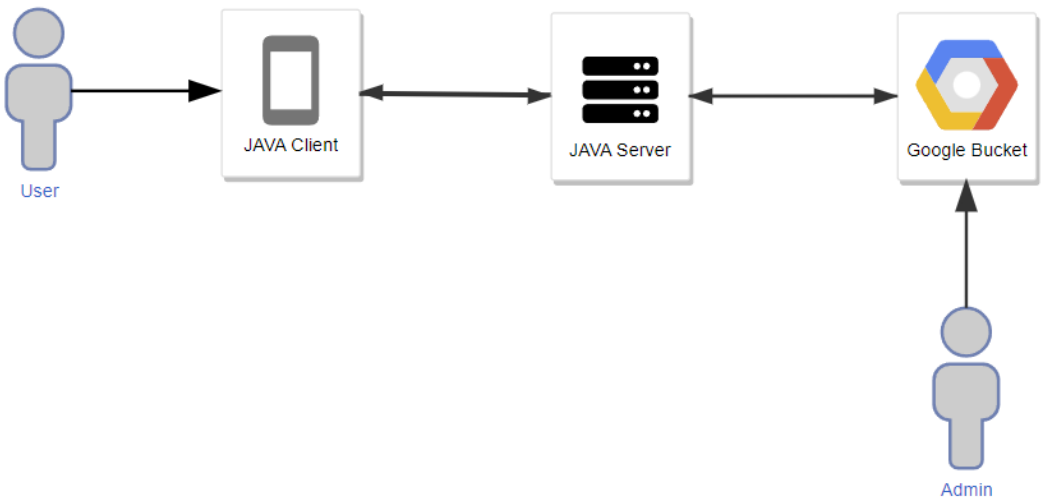


Figure 3.1: Architecture

As shown in Figure 3.2, the file structure of the application consists of the following main folders: java, res and assets. Inside the *java* directory are the main classes of the application (MainActivity, QRActivity, LibraryActivity). Inside the res directory are the resources of the application (icons, layouts, strings). Each layout represents a screen of the application (an Activity). Inside the assets directory are the 3D models, their images and the tutorial for some of the models. The assets directory is read-only at runtime, so to remove models from the application or add new models, the application will copy the files from assets to the local smartphone storage.

As shown in Figure 3.3, the *build.gradle* file contains the dependencies of the application. Also at the bottom of the figure is the method to generate a *.sfb* file from a *.obj* file. Sceneform supports 3D assets in the following formats:

1. OBJ
2. glTF (animations not supported)
3. FBX, with or without animations.

The models can be animated and also they can have textures and custom material[7].

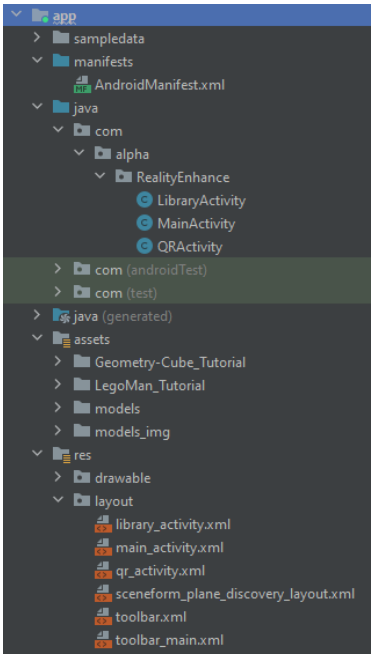


Figure 3.2: File Structure

```
dependencies {
    // ARCore (Google Play Services for AR) library.
    implementation 'com.google.ar:core:1.37.0'
    implementation 'com.google.ar.sceneform.ux:sceneform-ux:1.17.1'
    implementation 'com.google.ar.sceneform:core:1.17.1'
    implementation 'de.javagl:obj:0.2.1'
    // QR code-scanner library
    implementation 'com.github.yuriy-budiyev:code-scanner:2.3.2'
    implementation 'androidx.appcompat:appcompat:1.6.1'
    implementation 'com.google.android.material:material:1.9.0'
    implementation 'androidx.fragment:fragment:1.5.7'
    // Request - okhttp library
    implementation 'com.squareup.okhttp3:okhttp:4.9.1'
    androidTestImplementation 'org.testng:testng:6.9.6'
    testImplementation 'junit:junit:4.13.2'
}

// Generate .sfb file from .obj file
//sceneform.asset('sampledata/Painting.obj',
//        'default',
//        'sample/Painting.sfa',
//        'src/main/assets/models/Painting')
//
//sceneform.asset('sampledata/Geometry-Cube-Edge.obj',
//        'default',
//        'sample/Geometry-Cube-Edge.sfa',
//        'src/main/assets/Geometry-Cube_Tutorial/1')
```

Figure 3.3: build.gradle file

3.2 Diagrams

Class Diagram

As shown in Figure 3.4, the class diagram of the application consists of the following main classes: MainActivity, QRActivity and LibraryActivity.

The MainActivity class is the main class of the application. It is the first class called when the application is started, and it is responsible for the AR mode. It contains methods dedicated to interacting with the 3D models (moving the model, rotating the model, scaling the model).

The QRActivity class is responsible for the QR code reader. It has a method called when a QR code is scanned and verifies if it is valid. It will check if the model is already imported. If the model is already imported, it will load it from the local storage; it will also request the Google Bucket Storage. Lastly, the LibraryActivity class is responsible for the list of models and the loading of the models.

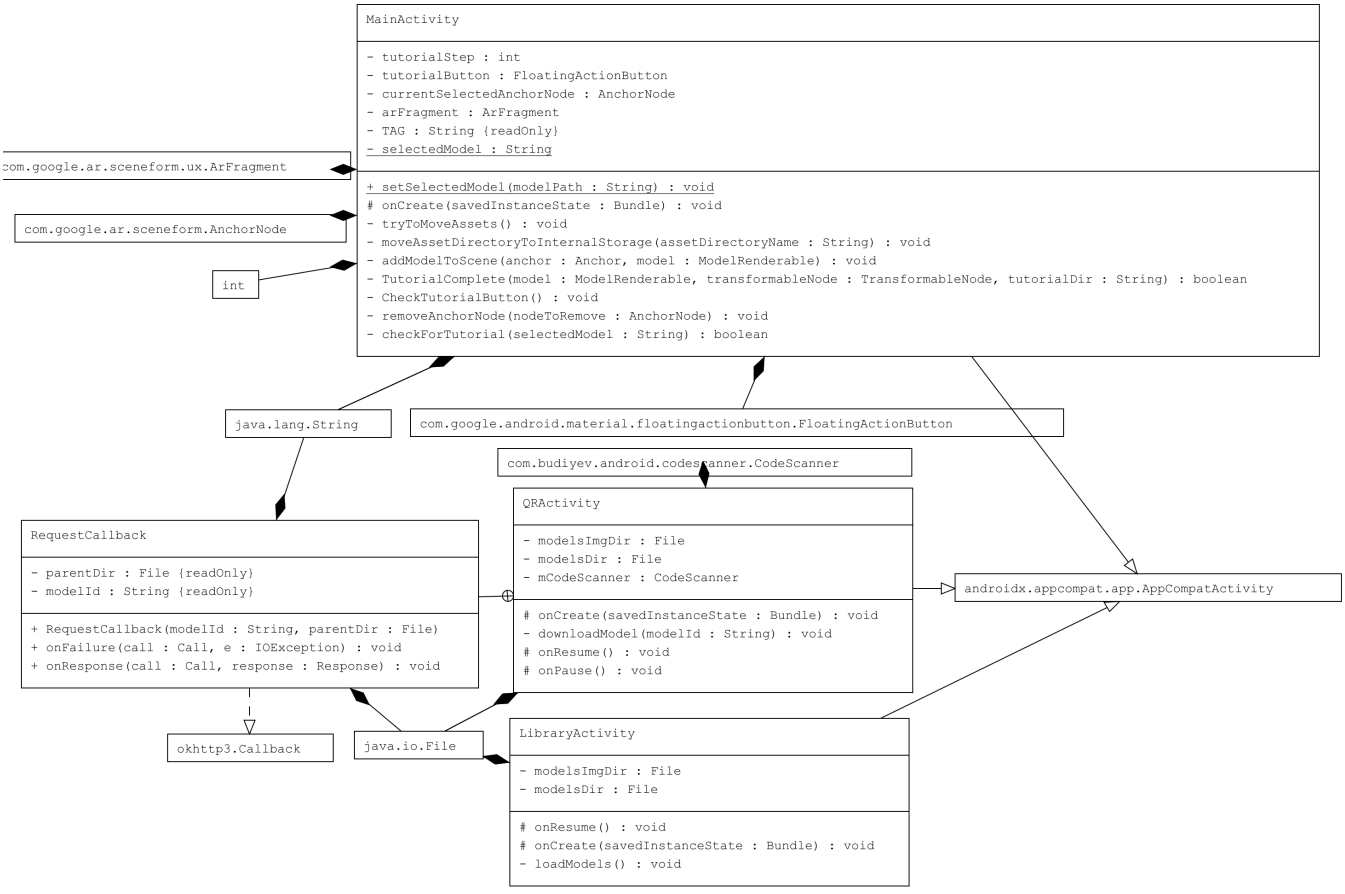


Figure 3.4: Class Diagram

Use Case Diagram

By examining the Use Case Diagram in Figure 3.5, we gain insights into the functionalities available to users within the application. RealityEnhance allows users to load or import a new model by scanning a QR code. Additionally, the application empowers users to engage with the 3D models in AR mode, granting them a dynamic and immersive experience. Furthermore, users can conveniently access a library containing recently used models.

The application's ability to interact with 3D models is a critical feature. It allows users to explore the models from various perspectives and manipulate their size, orientation, and position through simple gestures. Moreover, users are free to remove any selected model that has already been placed within the AR scene, providing them with flexibility and control over their augmented reality experience.

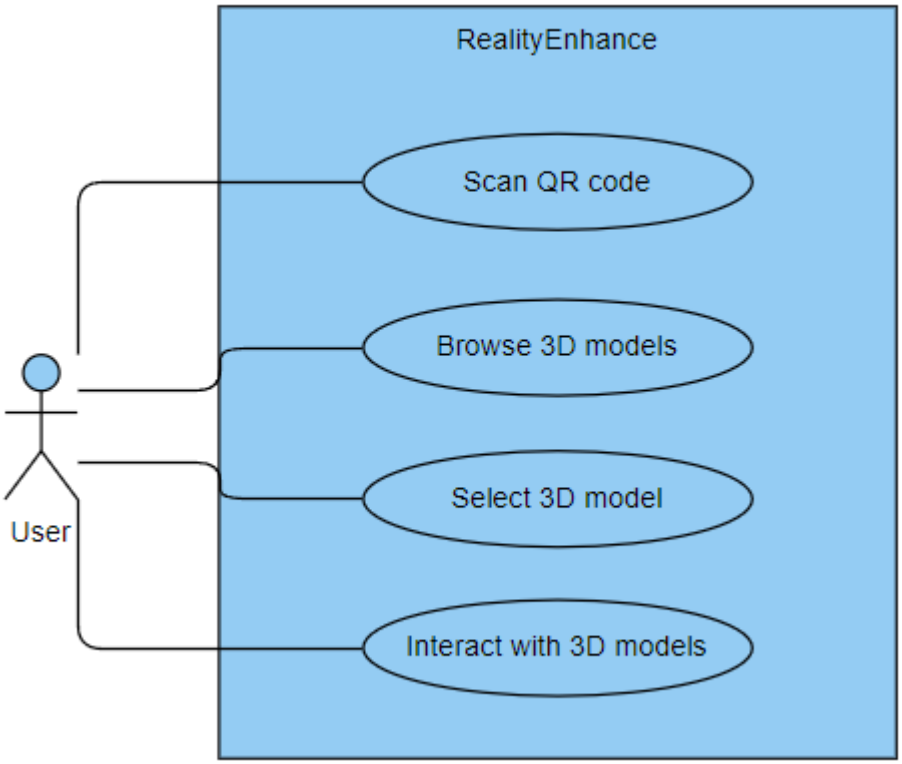


Figure 3.5: Use Case Diagram

Sequence Diagram

The Sequence Diagram in Figure 3.6 shows the interaction between the user and the application. When the user starts the application, a prompt will pop up and request permission to use the camera. After that, the AR module will begin to scan the user's surroundings in order to map the environment. For importing a new model, RealityEnhance uses a QR code scanner that will check if the code is valid.

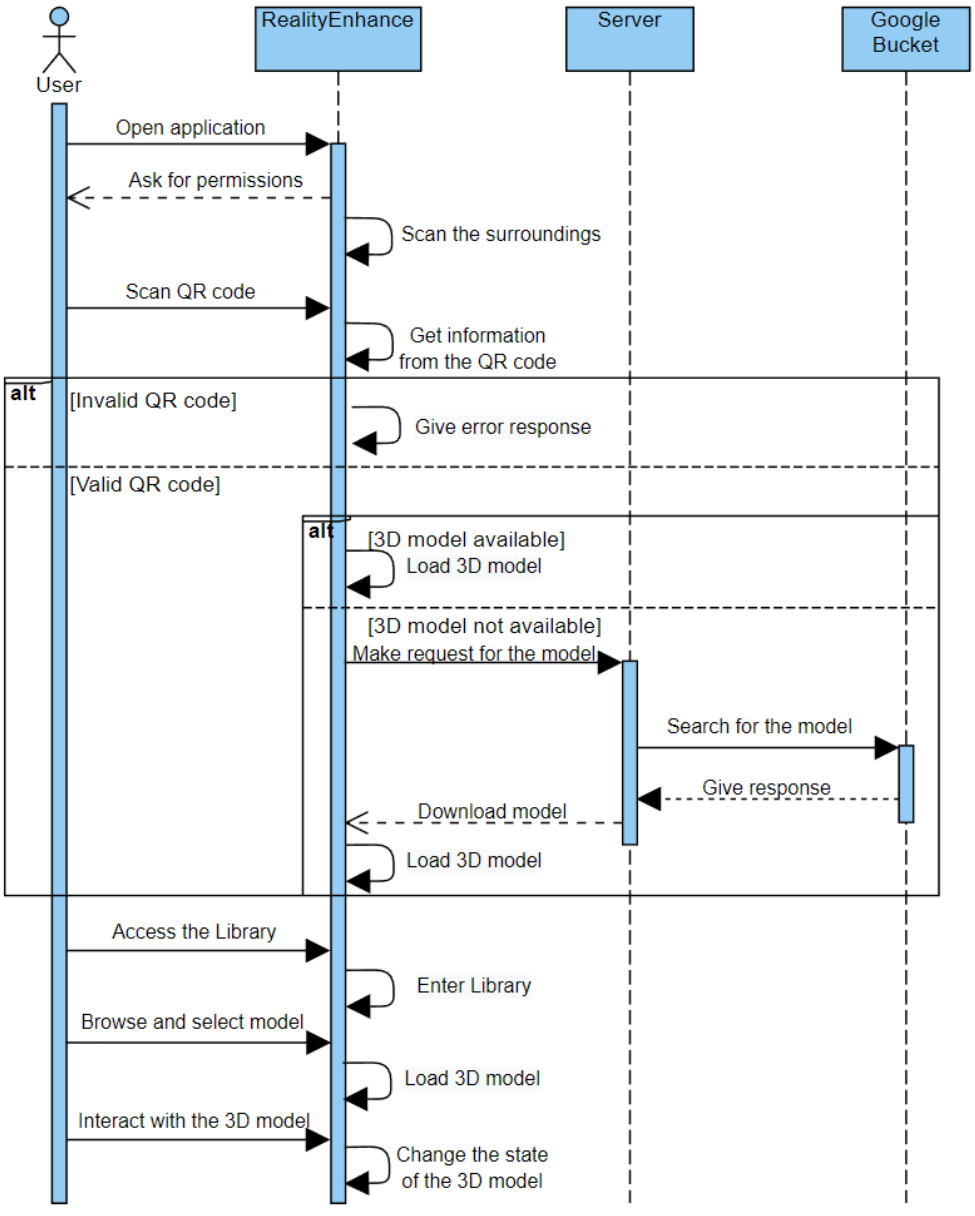


Figure 3.6: Sequence diagram

If the code fails the validation process, the user is notified and asked to scan another code. If the code is correct, the application will check to see if the model is already in local storage and then load it. If the model is not in the application's local storage, it will send a request to Google Bucket Storage to retrieve all of the model's information (model file, textures, model image, model guide/interactive guide). When the request is completed, a response will be sent back with all of the model's information. The model will then be saved locally and loaded into the AR scene.

The user can also browse the Library of already imported models. It also has access to the QR scanner interface and can search for the desired model that is wanted to be loaded in the augmented reality scene. In the AR mode, the user is able to place a multitude of models at the same time. These models can be rotated, resized and moved. The interaction between the user and the model bridges the virtual-physical gap and helps build a sense of immersion.

3.3 Publishing RealityEnhance

Introduction to App Distribution Platforms

Developers rely on app distribution platforms to release their applications. The platform they choose greatly impacts their application success. Among the top platforms, Google Play Store and Apple App Store stand out as the go-to choices. These platforms hold immense significance in terms of reaching a vast audience and making an application widely accessible.

Google Play Store

Google Play Store is the official app store for Android devices[6], offering users a diverse range of applications across various categories. This distribution platform was selected for publishing RealityEnhance because the application was developed for Android devices.

Before publishing an application on Google Play Store, the developers must create a Google Play Console account. The Google Play Console serves as a powerful platform for developers to share and distribute their applications to Android users worldwide. Not only does it provide a convenient way to reach a vast audience, but it also equips developers with a range of tools to expand their user base and generate revenue.

Accessing the Google Play Console is a breeze since it is a web-based platform, as shown in Figure 3.7. However, before publishing anything on the Google Play Store, it is crucial to ensure that the application is properly prepared for release. Taking the necessary steps to prepare the application will help guarantee a smooth and successful launch on the platform. The developer must have a specific file structure and then create a signed bundle file. After the bundle file is created and validated by Google Play Console, the application can be published.

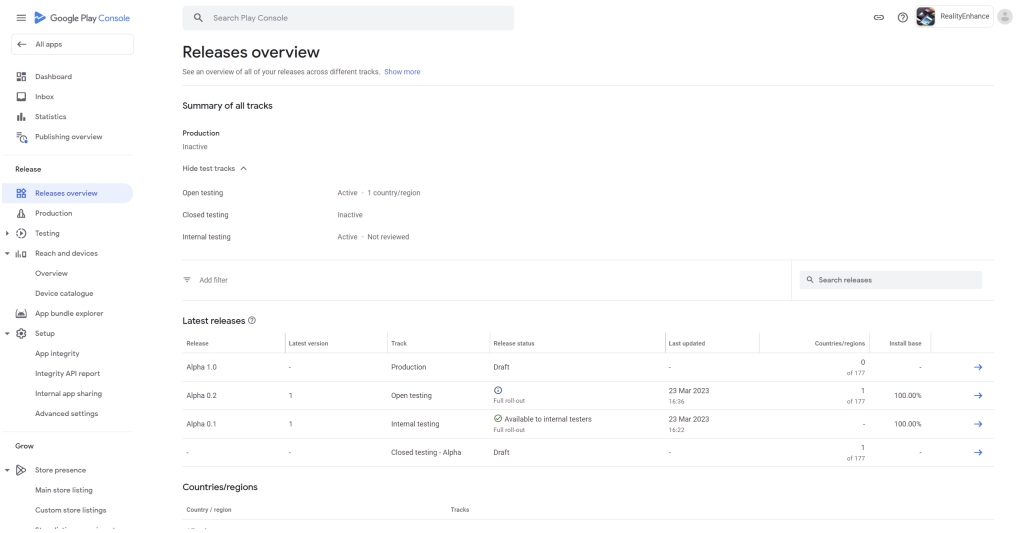


Figure 3.7: Google Play Console

Chapter 4

Implementation of the application features

4.1 QR code-scanner

The only way to import a new model is by scanning a QR code. The QR code contains the ID of the model. The application will check if the model is already in the local models' directory. If it is, it will load it from there, as shown in Figure 4.1. If it is not, it will proceed to download it from Google Bucket Storage.

Then the application will handle the response from Google Bucket Storage. If the response is successful, the model will be saved in the local models' directory and loaded in the AR scene. If the response is not successful, the application will display an error message, as seen in Figure 4.2.

```
public void onResponse(@NonNull Call call, @NonNull Response response) throws IOException {
    if (!response.isSuccessful()) { //Handle fail response
        String error = String.format("No model with ID: %s", modelId);
        runOnUiThread(() -> {
            Toast.makeText( context: QRActivity.this, error, Toast.LENGTH_SHORT).show();
            mCodeScanner.startPreview();
        });
    }
    if (response.isSuccessful()) { //Handle success response
        // Save the downloaded file to the assets directory
        ResponseBody responseBody = response.body();
        if (responseBody != null) {
            File file = new File(parentDir, modelId);
            InputStream inputStream = responseBody.byteStream();
            FileOutputStream fileOutputStream = new FileOutputStream(file);
            byte[] buffer = new byte[8192]; // Buffer size
            int bytesRead;
            while ((bytesRead = inputStream.read(buffer)) != -1) {
                fileOutputStream.write(buffer,  off: 0, bytesRead);
            }
```

Figure 4.2: onResponse method

```
mCodeScanner.setDecodeCallback(result -> runOnUiThread(() -> {
    String modelId = result.getText();
    for (File f : Objects.requireNonNull(modelsDir.listFiles())) {
        // IF the model is available, load it
        if (f.getName().equals(modelId)) {
            MainActivity.setSelectedModel(f.getAbsolutePath());
            finish();
            return;
        }
    }
    downloadModel(modelId);
```

(a) loadModelQR

```
private void downloadModel(String modelId) {
    // Download a model using OkHttp
    OkHttpClient client = new OkHttpClient();
    Request modelRequest = new Request.Builder()
            .url("https://storage.googleapis.com/reality-enhance-bucket/models/" + modelId)
            .build();
    Request modelImgRequest = new Request.Builder()
            .url("https://storage.googleapis.com/reality-enhance-bucket/models_img/" + modelId)
            .build();
    Request modelTutorialRequest = new Request.Builder()
            .url("https://storage.googleapis.com/reality-enhance-bucket/models_tutorial/" + modelId)
            .build();

    client.newCall(modelRequest).enqueue(new RequestCallback(modelId, modelsDir));
    client.newCall(modelImgRequest).enqueue(new RequestCallback(modelId, modelsImgDir));
    client.newCall(modelTutorialRequest).enqueue(new RequestCallback(modelId, modelsTutorialDir));
}
```

(b) downloadModel

Figure 4.1: QR code-scanner methods

4.2 Load the starting models

When the user starts the application, the method from Figure 4.3 will be called.

```
private void tryToMoveAssets() {
    File internalStorageDir = getFilesDir();
    File modelsDir = new File(internalStorageDir, child: "models");
    File modelImgsDir = new File(internalStorageDir, child: "models_img");
    if (modelsDir.exists() && modelsDir.isDirectory() && modelImgsDir.exists() && modelImgsDir.isDirectory()) {
        return;
    }
    moveAssetDirectoryToInternalStorage( assetDirectoryName: "models");
    moveAssetDirectoryToInternalStorage( assetDirectoryName: "models_img");
    try {
        String[] assetDirectories = getAssets().list( path: "");
        if (assetDirectories != null) {
            for (String assetDirectory : assetDirectories) {
                if (assetDirectory.endsWith("_Tutorial")) {
                    moveAssetDirectoryToInternalStorage(assetDirectory);
                }
            }
        }
```

```
private void moveAssetDirectoryToInternalStorage(String assetDirectoryName) {
    try {
        // Get the list of files in the asset directory
        String[] fileList = getAssets().list(assetDirectoryName);
        // Create a directory in the internal storage
        File internalDirectory = new File(getFilesDir(), assetDirectoryName);
        if (!internalDirectory.exists()) {...}
        // Iterate through the files in the asset directory
        for (String fileName : fileList) {...}
        // The directory and its contents have been moved to the internal storage
    } catch (IOException e) {
        e.printStackTrace();
    }
}
```

Figure 4.3: Move files from assets to local storage

It will load the models from the assets directory to the local smartphone storage because the assets directory is read-only at runtime, and the application must store all the newly imported models at the same location. The method will check if the models are already in the local storage. If they are, it will load them from there. If they are not, it will copy them from the assets directory to the local storage.

4.3 Models' library

In the Library Activity, the models will be displayed in two columns. This is done at the start of the activity, in the method from Figure 4.4. It will go through all the files from the local models' directory and it will create a new button for each file.

The buttons will have as background the image of the model. If the model does not have a background image, the button will have as background the name of the model. When the user clicks on a button, the model will be loaded in the AR scene.

```
private void loadModels() {
    GridLayout gridLayout = findViewById(R.id.gridLayout);
    gridLayout.setColumnCount(2);
    File[] fileList = modelsDir.listFiles();
    int numRows = (int) Math.ceil((double) Objects.requireNonNull(fileList).length / 2);
    gridLayout.setRowCount(numRows);

    int i = 0;
    for (File file : fileList) {
        Button button = new Button( context: this);
        button.setOnClickListener(view -> {
            MainActivity.setSelectedModel(file.getAbsolutePath());
            Toast.makeText( context: this,  text: "Selected model: " + file.getName(), Toast.LENGTH_SHORT).show();
        });

        File imgFile = new File(modelsImgDir, file.getName());
        try {
            FileInputStream stream = new FileInputStream(imgFile);
            Drawable drawable = Drawable.createFromStream(stream,  srcName: null);
            button.setBackground(drawable);
            button.setText("");
            stream.close();
        } catch (IOException e) {
            button.setText(file.getName());
        }

        GridLayout.LayoutParams params = new GridLayout.LayoutParams();
        params.width = 0;
        params.height = ViewGroup.LayoutParams.WRAP_CONTENT;
        params.columnSpec = GridLayout.spec( start: i % 2,  weight: 1f);
        params.rowSpec = GridLayout.spec( start: i / 2,  weight: 1f);
        params.setMargins( left: 8,  top: 4,  right: 8,  bottom: 4);
        button.setLayoutParams(params);
        gridLayout.addView(button);

        i++;
    }
}
```

Figure 4.4: Load models in the library

4.4 Interaction with the models

The user can interact with the models in the AR scene by using the buttons on the screen and using smart gestures on the model. As seen in Figure 4.5, when the user taps on the screen, the addModelToScene method will be called. It will check if the model has an interactive mode; if it does, it will activate the button responsible for the interaction with the model. If the model does not have an interactive mode, it will display a message to the user. Each time the user presses the interaction button, the model will change its state.

```
private void addModelToScene(Anchor anchor, ModelRenderable model) {
    // Create the anchor node
    AnchorNode anchorNode = new AnchorNode(anchor);
    // Create the transformable node
    TransformableNode transformableNode = new TransformableNode(arFragment.getTransformationSystem());
    transformableNode.setRenderable(model);
    // Add the node to the scene
    arFragment.getArSceneView().getScene().addChild(anchorNode);
    // Set the min and max scales of the ScaleController.
    transformableNode.getScaleController().setMinScale(0.4f);
    transformableNode.getScaleController().setMaxScale(1.2f);
    // Set the local scale of the node BEFORE setting its parent
    transformableNode.setLocalScale(new Vector3( v: 0.55f,  v1: 0.55f,  v2: 0.55f));
    transformableNode.setParent(anchorNode);
    currentSelectedAnchorNode = anchorNode;
    CheckTutorialButton();
    // Select the renderer node
    transformableNode.select();
    transformableNode.setOnTapListener((hitTestResult, motionEvent) -> currentSelectedAnchorNode = anchorNode);
```

Figure 4.5: addModelToScene method

If a model is selected and the user wants to remove that model from the scene, he can do that by pressing the remove button. This will call the removeAnchorNode method, as seen in Figure 4.6. It will remove the model from the scene, and it will set the *selectedModel* variable to null.

```
private void removeAnchorNode(AnchorNode nodeToRemove) {
    //Remove an anchor node
    if (nodeToRemove != null) {
        arFragment.getArSceneView().getScene().removeChild(nodeToRemove);
        Objects.requireNonNull(nodeToRemove.getAnchor()).detach();
        nodeToRemove.setParent(null);
        Toast.makeText( context: MainActivity.this,  text: "Test Delete - markAnchorNode removed", Toast.LENGTH_SHORT).show();
        tutorialStep = 0;
        tutorialButton.setVisibility(View.GONE);
        Log.d(TAG, String.format("removeAnchorNode: %d", tutorialStep));
    } else {
        Toast.makeText( context: MainActivity.this,  text: "Delete - no node selected! Touch a node to select it.", Toast.LENGTH_SHORT).show();
    }
}
```

Figure 4.6: removeAnchorNode method

Chapter 5

User Manual

5.1 Requirements

The application needs some hardware, software and permission requirements in order to run and to be used. The hardware requirements are:

1. CPU/GPU SoC (System on chip) that support OpenGL ES 3.0 or later
2. Touch-Screen with a size of at least 4.7 inches
3. Camera supporting auto-focus, at least 720p resolution, 30fps and 60 degrees horizontal field of view
4. Internet antenna

The software requirements are:

1. Android 7.0 or newer
2. Google Play Services for AR
3. Google Play Store application
4. RealityEnhance application

More information about the hardware and software requirements can be found on the official ARCore website [4].
The permission requirements are:

1. Camera permission
2. Internet permission
3. Storage permission

5.2 Graphical overview of RealityEnhance

We will now go through some of the application's graphical user interfaces. The following should summarise the fundamental flows present in the application and

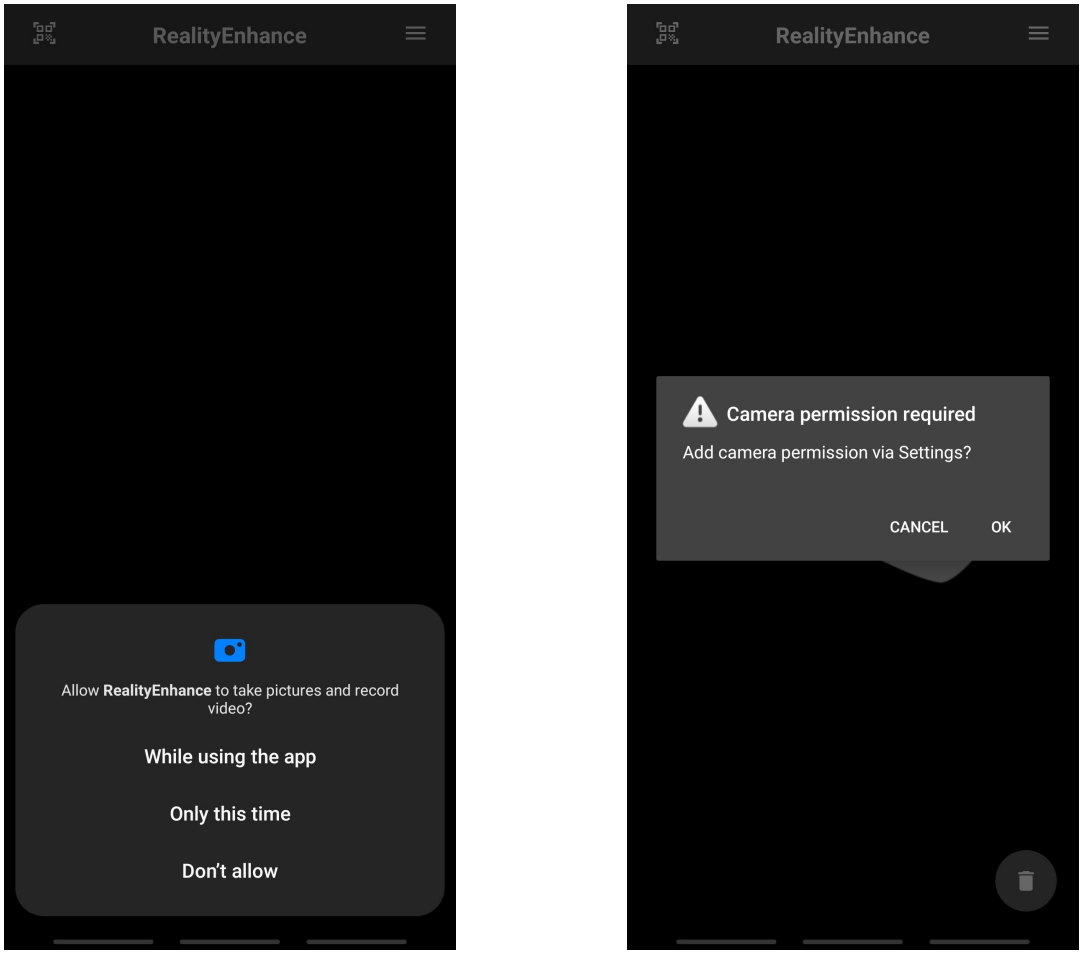
accommodate the user concerning the interface. The following figures are screenshots that were taken inside the application. All the presented interfaces are fully functional and are implemented in the application.

5.2.1 Application permissions

The first thing any user will see when they launch the application on their smartphone is the loading page. A pop-up will be displayed asking the user for permission to take pictures or record video (to use the camera).

This page is a simple loading screen displayed while the application is loading. In the loading phase, the app will check if the user can run the app on their phone and if all the permission needed are met. If the user can run the app, the app will load the AR Activity. If the user does not give the required permissions, the app will display a message that the user cannot run the app on their phone and that permissions are required in order to use the app.

This prompt is displayed only once when the user launches the application for the first time.



(a) Image 1 (b) Image 2

Figure 5.1: Loading page

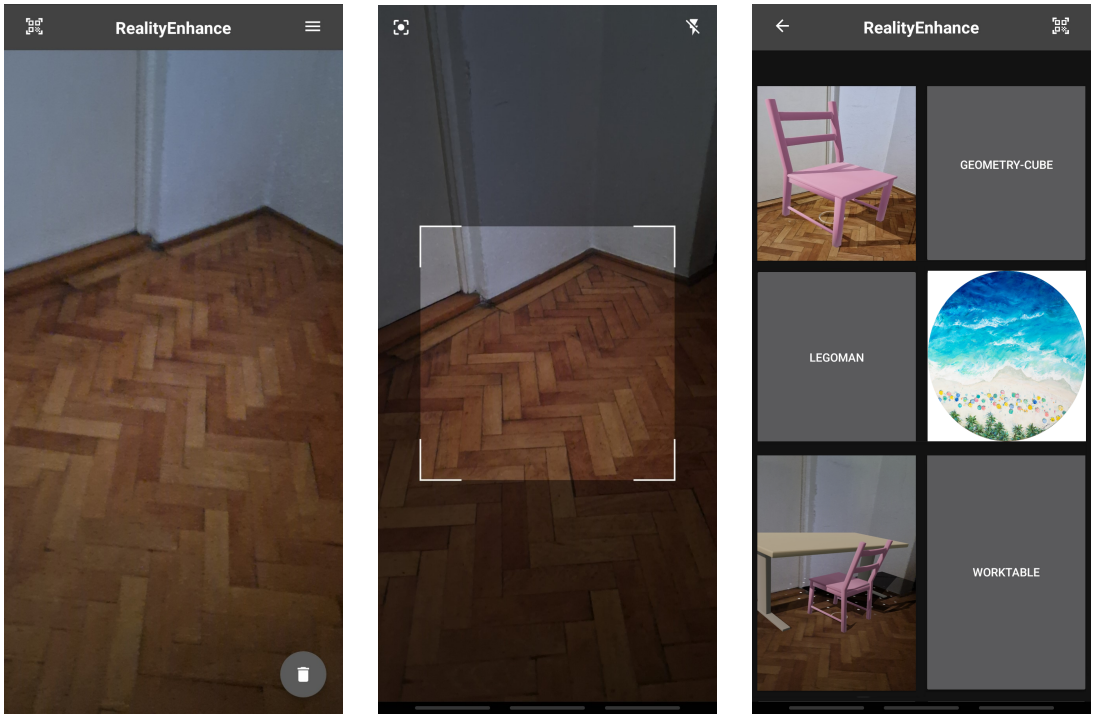
5.2.2 Navigation/Tool Bar

RealityEnhance uses a Navigation/Tool Bar to navigate between the different activities and to access the different features of the app. The application also uses the smartphone's back button (smart gesture from the bottom of the screen or from the sides) to navigate to the previous activity or exit the application. The Navigation/Tool Bar is displayed on the top of the screen. As you can see in Figure 5.2, the Navigation/Tool Bar has three different modes:

(a) AR Activity - can access the QR scanner and the Library

(b) QR Activity - can turn on the flashlight and can turn on the auto-focus

(c) Library Activity - can access the QR scanner and the AR Activity



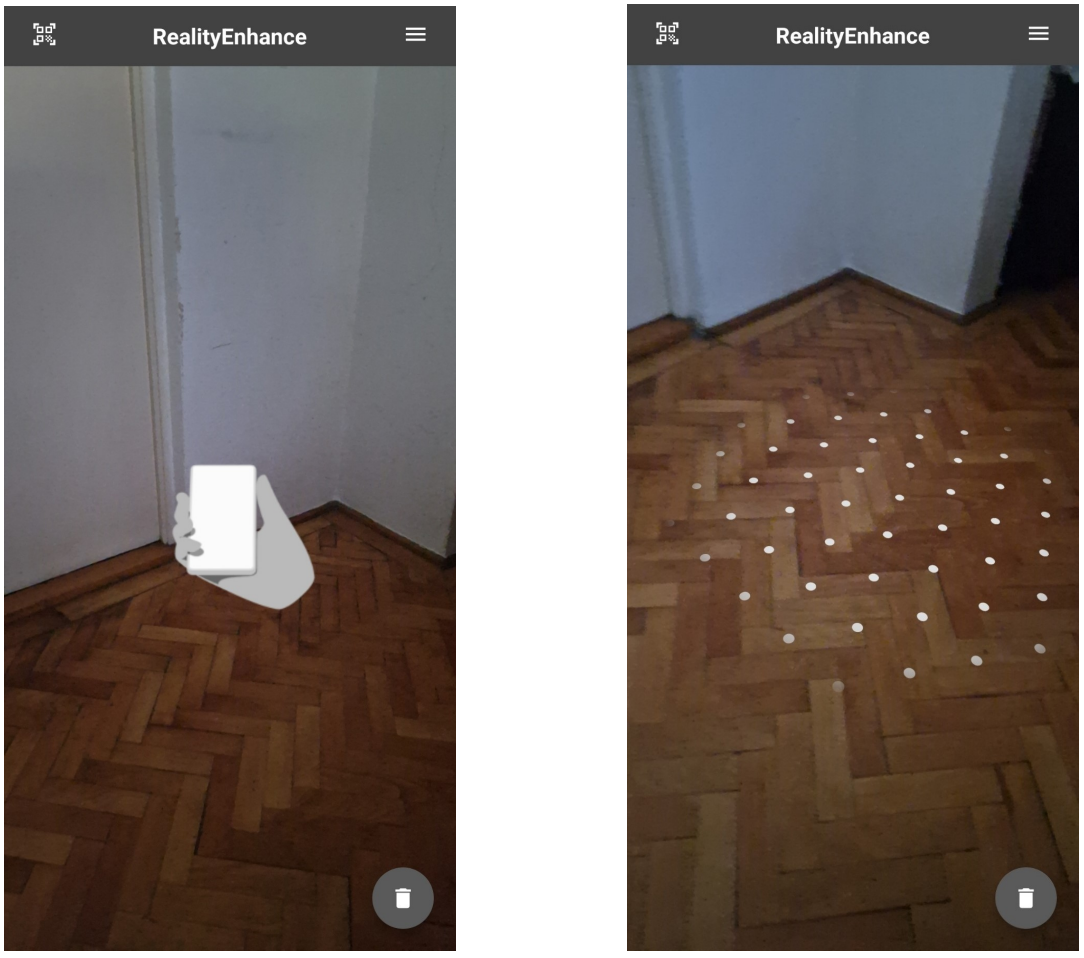
(a) AR Activity (b) QR Activity (c) Library Activity

Figure 5.2: Navigation and toolbar

5.2.3 AR Activity

The main "page" of the application is the AR activity. This activity is the one that will be displayed when the user launches the application after giving all the permissions. As you can see in Figure 5.3 picture *(a)*, an animation of a hand holding a phone will be displayed to indicate the application is scanning the environment and also to indicate to the user how to move the smartphone in order for the surroundings to be scanned.

A scanned surface will be represented by a mesh of dots, as shown in the Figure 5.3 picture *(b)*. A surface can be a floor, a table, a wall and many more. The mesh represents the available surfaces the user can place 3D models.



(a) Scanning animation

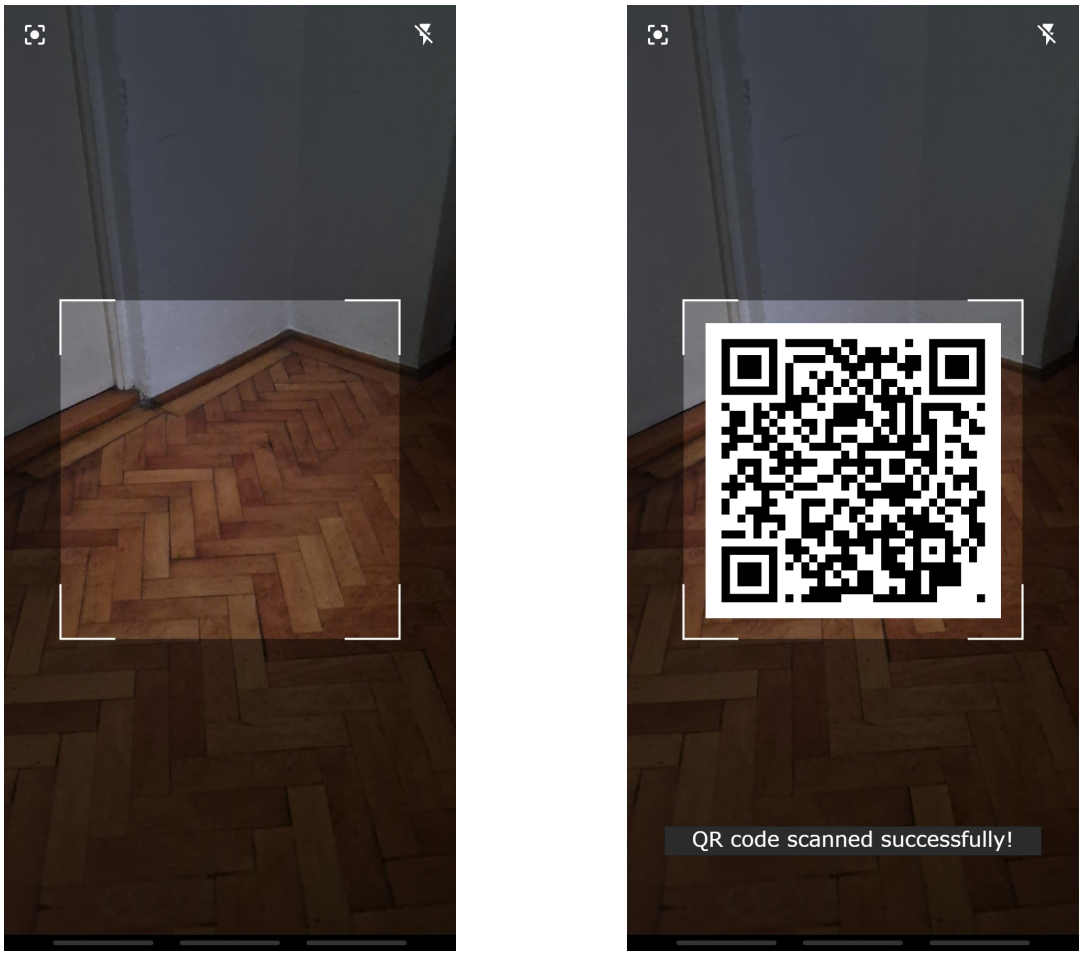
(b) Scanned enviroment

Figure 5.3: AR Activity

5.2.4 QR Activity

The only method to import a new model into the application is by scanning a QR code. In Figure 5.4, we can see the interface of the scanner. To scan a QR code, the user needs to point the camera to the QR code and fit the code in the square. The scanner will automatically scan the code and will load the model in the application. The user will then be redirected to the Augmented Reality (AR) Activity.

The user can also turn on the flashlight and the auto-focus by clicking on the buttons on the top of the screen. If the QR code is invalid, the user will be notified by a pop-up message, and the scanner will be restarted.



(a) QR Activity (b) Scanning QR code

Figure 5.4: QR Activity

5.2.5 Library Activity

RealityEnhance has, from the start, a couple of 3D models preloaded in the library. The user can access the library by clicking on the library button (the button is represented by three lines) in the Navigation/Tool Bar. As you can observe in Figure 5.5, the library has a grid listing of models that the user can choose from.

The user can click on a model to load it into the application. The user will then need to go back to the AR Activity to use the model. From the library, the user can also access the QR scanner by clicking on the QR button (the button is represented by a QR code). The list of models is scrollable, and the user can scroll through the list by swiping up or down on the screen.

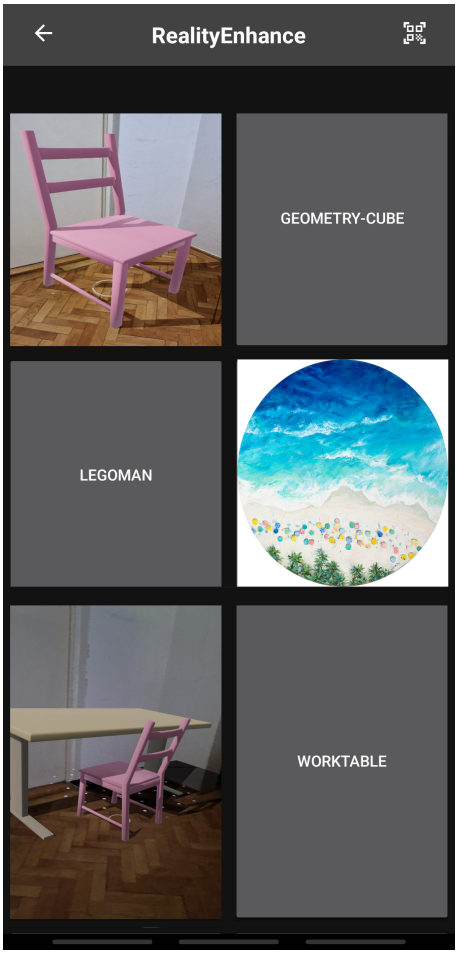


Figure 5.5: Library Activity

5.2.6 Placing a model

After the user has either scanned a model or selected one from the library, the user can place the model in the environment (on the scanned surface), as shown in Figure 5.6. The model's placement is done by simply touching the screen at the desired location.

Users are not limited to a single model. They can add multiple models to the scene, the same model or different models. The position of the already placed model will be maintained even if the user changes the activity or change the application they are using. In order to know which model is selected, the user can see a white circle under the model.

Figure 5.6: Placing a model

5.2.7 Using multiple models at the same time

The user will be able to use multiple models at the same time in the same scene. The model's hitboxes will be separated, and the user will be able to select a model by clicking on it, but the models will not interact with each other.

The only interaction between models is that they can cast shadows on one another. The user can see in Figure 5.7 an example of multiple models placed in the same scene and that the worktable is casting a shadow on the chair.

Figure 5.7: Removing a model

5.2.8 Removing a model

The user can remove the model by selecting it by clicking and then pressing the delete button. The delete button is represented by a trash can icon in the bottom right of the screen. The user can also remove a model by clicking and dragging the model on top of the delete button. The user can also remove all the models placed by holding the delete button for 5 seconds 5.8.

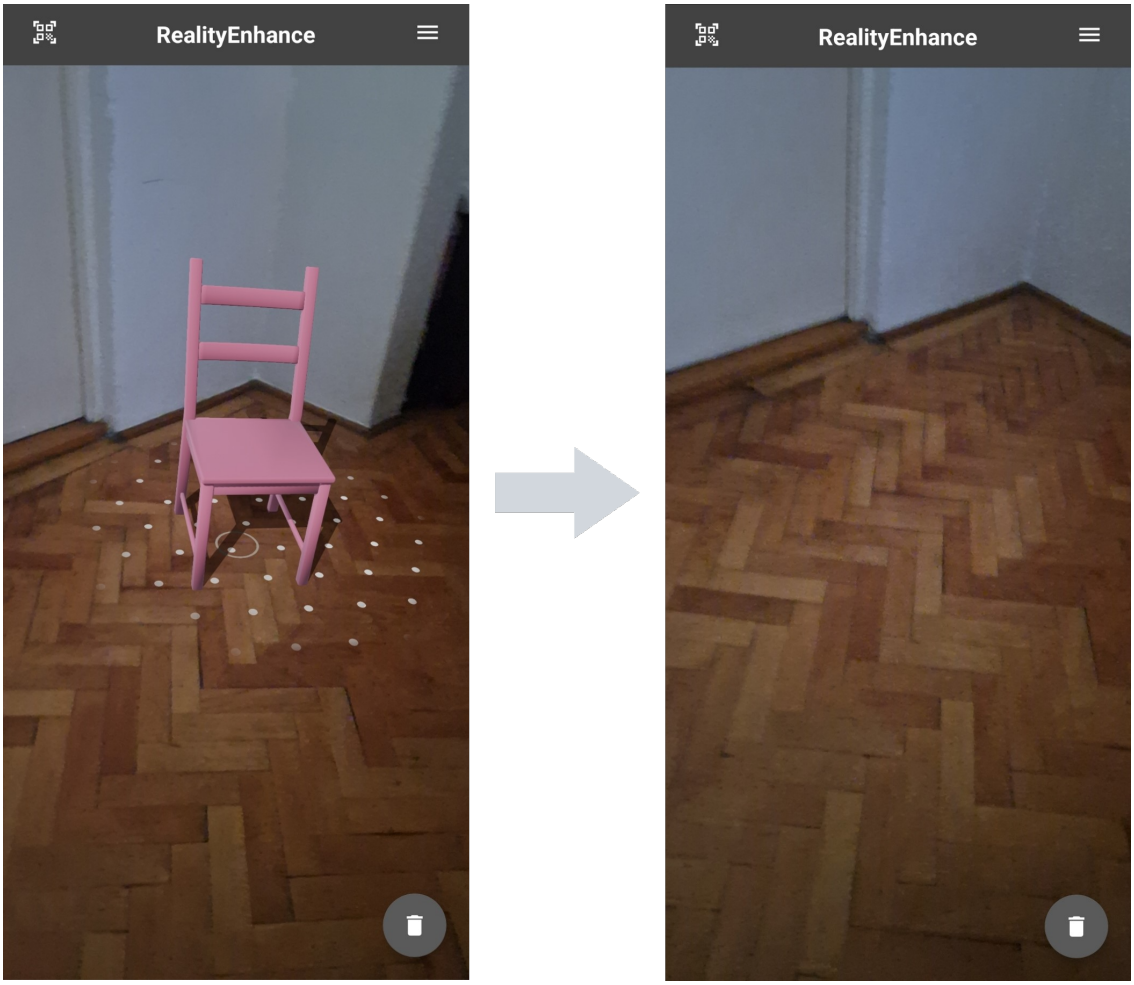


Figure 5.8: Removing a model

5.2.9 Resizing, moving and rotating a model

The user is able to change the position of a 3D model by holding the model and dragging it across the screen; it is also possible to move the model by holding it and moving the smartphone. The model can be resized by pinching the model with two fingers, and it can be rotated with the same two fingers. It can be seen in Figure 5.9 as an example of resizing and rotating a model. The gestures used to resize and rotate a model are the same gestures used to resize and rotate an image in the gallery application.

Figure 5.9: Resized and rotated a model

5.2.10 Interacting with a model

Some models will have a unique interaction with the user. Those models with access to the interactive feature are marked with a hat icon in the library, and when they are placed in the scene, the interaction button will appear at the bottom left of the screen.

As shown in Figure 5.10, after the user clicks on the interaction button, the model of a geometrical cube changes from a whole cube to just the edges being visible. This interaction is just an example of what can be done with the interactive feature. Another example of an interactive model is a model of a LEGO man that will have a tutorial for assembling the LEGO figure as the interactive part.

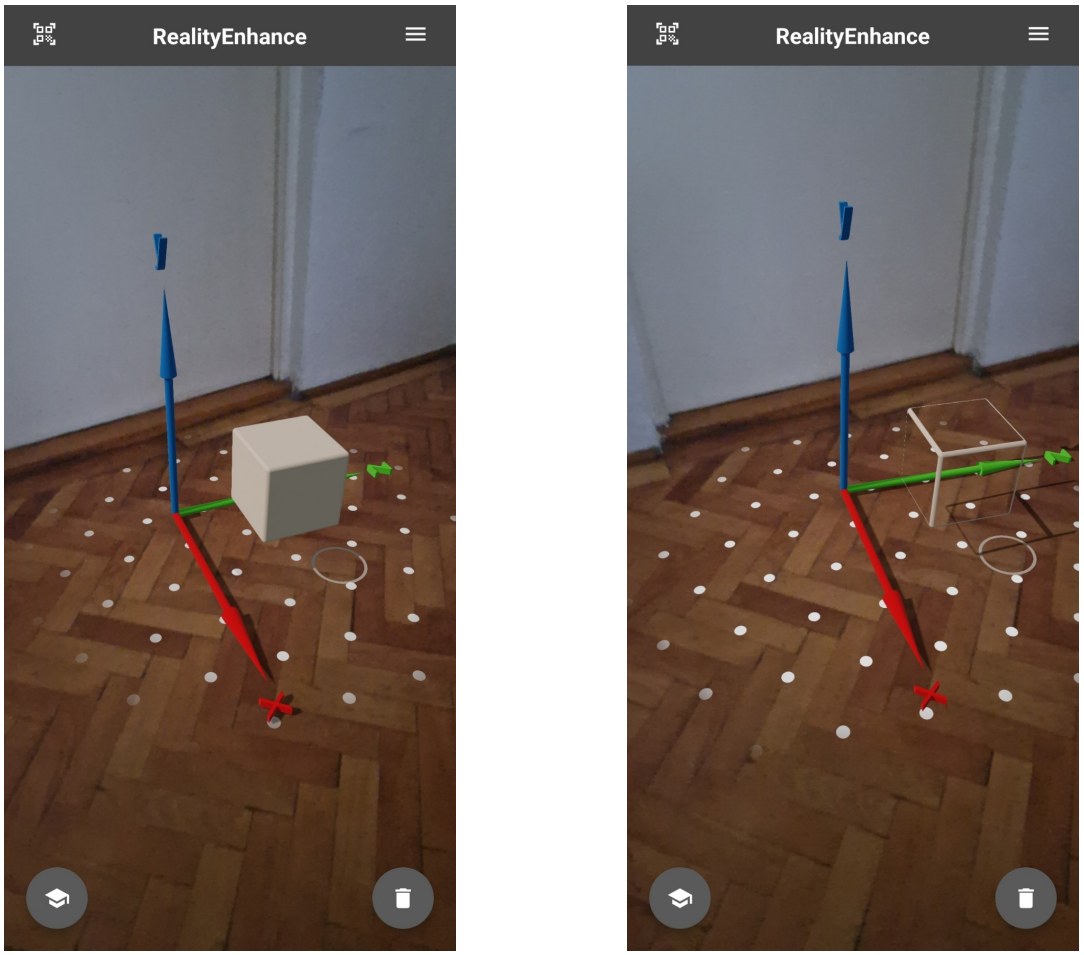


Figure 5.10: Interactive model

Chapter 6

Conclusion and Future work

6.1 Conclusion

To sum up the results of the thesis, the Android application RealityEnhance meets the goals that were set at the beginning of the project. The application gives the user the power of enhanced perception of their surroundings by placing 3D models in the real world. The application has a simple and intuitive user interface that allows it to have many applications in a wide range of domains. It also shows the potential of AR in the field of education. RealityEnhance can be used as a didactic tool for the teachers by allowing students to interact with the models in the AR mode.

The application will be available for Android devices and it will be free to download from the Google Play Store. The application will also be available for IOS devices in the future. All the application code is available on GitHub[1] and is open source.

6.2 Future work

In this section, we will present the future work that will be done to improve the application. A list of future improvements and features that will be added to the application:

(a) Store all the information about a model in a QR code

(b) Make the application available for IOS

(c) Develop a platform that allows users to upload models

Bibliography

[1] RealityEnhance. https://github.com/Sorin-Ionut-Rosalim/Bachelor-App. Last accessed 23 June 2023.

[2] Yuriy Budiyev. Code scanner library. https://github.com/yuriy-budiyev/code-scanner. Last accessed 23 June 2023.

[3] Monica Burns. *Deeper Learning With QR Codes and Augmented Reality: A Scannable Solution for Your Classroom*. Corwin, 2016.

[4] Google. Android ARCore supported devices. https://developers.google.com/ar/devices#google_play. Last accessed 18 June 2023.

[5] Google. Android Studio Flamigo. https://developer.android.com/studio/releases. Last accessed 23 June 2023.

[6] Google. Google play store. https://play.google.com/. Last accessed 23 June 2023.

[7] Google. Import and preview 3d assets. https://developers.google.com/sceneform/develop/import-assets. Last accessed 23 June 2023.

[8] Niantic Inc. Catching Pokémon in AR+ mode. https://niantic.helpshift.com/hc/en/6-pokemon-go/faq28-catching-pokemon-in-ar-mode/. Last accessed 21 June 2023.

[9] Snap Inc. Snap AR. https://ar.snap.com/. Last accessed 21 June 2023.

[10] Snap Inc. Snapchat. https://www.snapchat.com/, 2011. Last accessed 21 June 2023.

[11] Oracle James Gosling. Java SE 20. https://www.oracle.com/java/technologies/java-se-glance.html. Last accessed 23 June 2023.