ALBERT-LUDWIGS-
UNIVERSITÄT FREIBURG

Department of Computer Science
Chair of Computer Architecture

ARCape

Master Project

Android Based Augmented Reality Hint System For Smart Escape Rooms

Project of: Sachin Shibu Dominic
Matriculation number: 5164585

Submission on: 21.2.2022

Supervisor: Dr. Philipp M. Scholl

Acknowledgement

I would like to thank my supervisor, Dr. Philipp M. Scholl for his timely guidance and active feedback during the project. His insight into how to develop the the project and ideas were a valuable resource.

Also a huge shout out to the Sceneform Open Source Community for maintaining the repository after it was deprecated by Google.

Abstract

The aim of this project was to develop a system which could be used to provide hints to the participants of the escape room without any external assistance. It had to be done in such a manner that the participant would be aware of the existence of this hint system and it would not go unnoticed which was an issue with the previous hint system using amazon echo. In order to do this an android application was developed, which on scanning an image related to the puzzle and in it's vicinity, projects an AR object which represents the internal state of the puzzle, whether it's not activated yet, activated or solved. In case the puzzle is activated, the AR object displayed will be that of the hint.

The application was developed using Kotlin in Android Studio and uses Google's augmented reality platform ARCore to identify the images and place an AR object on the identified image.

Table of Contents

List of Figures

List of Tables

List of Abbreviations

AR - Augmented Reality

API - Application Programming Interface

AI - Artificial Intelligence

SDK - Software Development Kit

IoT - Internet of Things

MQTT - Message Queue Telemetry Transport

m2m - Machine to Machine

IDE - Integrated Development Environment

OS - Operating System

JSON - JavaScript Object Notation

UI - User Interface

UX - User Experience

Chapter 1

Introduction

1.1 Aims

The main objective of this project was to develop a hint system to help the players in the escape room solve the puzzles. In order to develop this system, it was decided to make an android application through which the users could understand the internal state of a puzzle and get relevant hints. For this, it was decided to place AR objects using an application which depicts the state of a puzzle and also show the hints when the puzzle is playable/activated.

1.2 Motivation

In the current iteration of the escape room, if a player gets stuck, it is sometimes not possible to make any progress or understand what needs to be done next without external intervention. This affects the enjoyment factor and also requires a person who is familiar with the puzzles to be present around when the puzzles are being solved. Additionally, in the previous iterations, there was an Amazon Echo with Alexa which could be asked for hints in game (as an AI), however, the majority of the players failed to even realize the existence of this system.

Therefore, it was decided to develop a system which would enable the players to understand the internal state of a puzzle, get hints and proceed through the game environment without the need of any assistance.

1.3 Project Overview

For this project, an android application was developed which can be used to scan an image in the vicinity of the puzzle. The application would augment this image and and place an AR object based on the internal state of the puzzle:

(i) **Not activated state:** In this state, the puzzle has not been activated and is not playable. The application will place an AR object with the text "Not Activated" to highlight the state of the puzzle.

(ii) **Activated state:** In this state, the puzzle has been activated and is playable. The application does the following:

   (a) **Tap to Show Hint:** The AR projection displays "Tap to show hint", in case the player doesn't want to use the hint to solve the puzzle.

   (b) **Hint:** When the user taps the screen, the application places an object with the hint text. In case of multiple hints, when the next hint is active, the application goes back to (a) and this repeats until all the hints have been obtained.

(iii) **Solved state:** When the puzzle has been completely solved it reaches the solved state. The AR object now displays the text"solved!!".

In addition to this, the application also starts the escape room and initiates the power failure scenario of the escape room, so that this does not need to be done externally.

1.4 Report Overview

The next chapter will go through the different technologies and software used for the development of this project.

Chapter 3, will go through the implementation and the individual activities of the project along with a description of each puzzle.

Finally, the conclusion chapter will discuss the results of the project, known issues, future improvements and the learning outcomes, along with the link to the source code of the project.

Chapter 2

Literature Survey

This section will talk about the technologies and terms that were required to be known to successfully undertake and complete this project.

2.1 Augmented Reality (AR)

AR is defined as a technology that superimposes a computer-generated image on a user's view of the real world, thus providing a composite view[13]. It enhances the real physical world through the use of digital visual elements, sound, or other sensory stimuli delivered through technology.[8]

Figure 2.1.1: Several Examples of Augmented Reality[12]

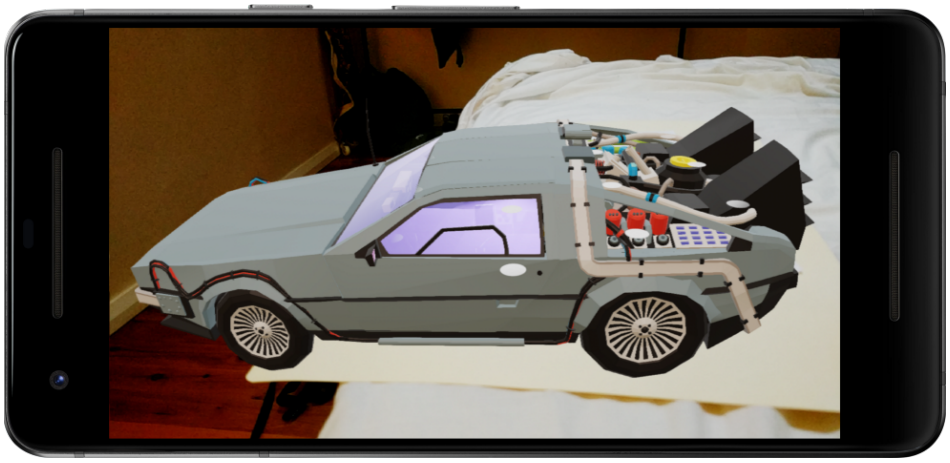
An image showing several examples of AR being used to place furniture around a room to simulate how it would fit in before making the actual purchase

2.2 Augmented Images

The word augment, by definition means to add information to something. So in theory, we can define augmented images as the process of adding information to an image. Though the use of AR technology, it is possible to do this by detecting an image and showing relevant information on the device being used for this detection.

To understand this better, a demonstration of an augmented image technology is shown in the figure 2.2.1.

((a)) Before augmenting the image



((b)) After augmenting the image

Figure 2.2.1: Demonstration of Augmented Image[3]

Subfigure ((a)) depicts an image being seen through the camera before it has been augmented. Once the image has been augmented(to place a 3D model of a car), when the image is scanned the model is rendered on top of the image as shown in subfigure ((b))

2.3 Internet of Things (IoT)

The Internet of things defines the collection of devices around us which are connected to the internet, and can transmit data among themselves and communicate with each other. It can range from something as simple as a kid's toy to a collection of sensors and actuators controlling a smart home. The escape room is built on an IoT platform and communicates via MQTT. The MQTT messaging platform will be discussed in section 2.8.

Figure 2.3.1: Depiction of an IoT network[11]

The image visualizes how a smartphone is connected to user's home and the other devices around them. This connection can be used to control the connected devices remotely and make it easier to monitor these devices.

2.4 Android Development

In this section, the necessary aspects of Android Development required for this project will be discussed.

2.4.1 Android SDK

Android applications are primarily developed using the Android Software Development Kit which is provided by Google. The relevant libraries necessary for development are already included in the SDK. Along with that it also

consists of a debugger and an android virtual device manager for simulating apps on a virtual platform during development.

In addition to these, the Android SDK also provides the arcoreimg tool[6]. This tool is used to evaluate if an image is suitable for an augmented images application and also helps to create the augmented images database (will be discussed in section 2.5.3).

2.4.2 Android Studio

Android Studio is the official platform provided by Google to develop apps for it's Android operating system using the Android SDK. The IDE for Android studio is built on top of JetBrain's Intellij IDEA.

Android Studio provides the functionality to develop the backend of a project and also the ability to design the frontend for the corresponding backend developed.

In addition to this, Android Studio also offers several features which are useful for development of an application like auto-complete, version control systems and the ability to detect errors in code during development.

A Snapshot of the Android Studio IDE is shown in figure 2.4.1

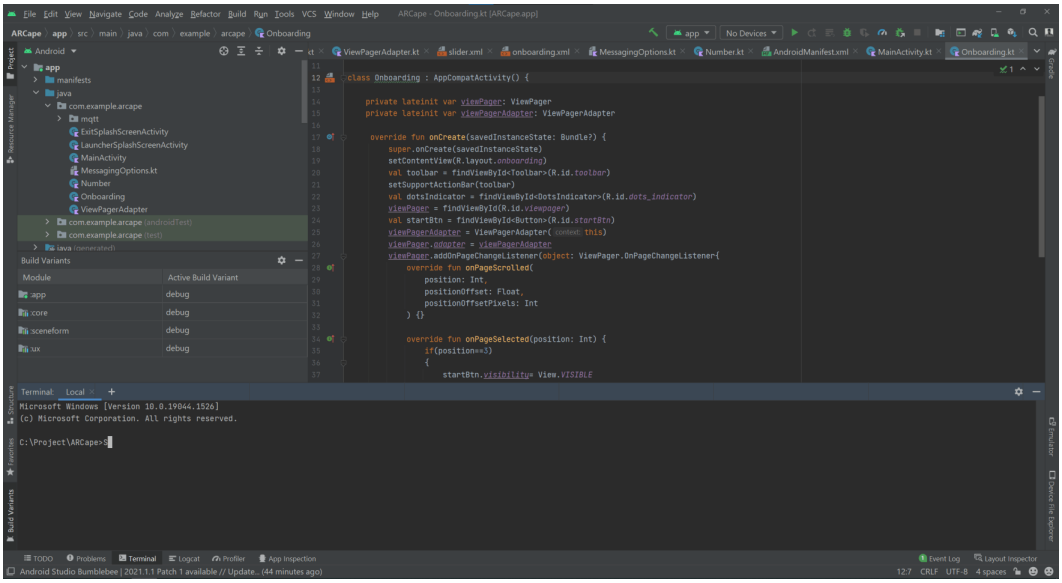


Figure 2.4.1: Android Studio Interface

A snapshot of Android Studio showing the Editor, Project Pane (which shows the files and directories) and the inbuilt terminal window.

2.4.3 Programming Languages Used

Android Studio offers support to write the backend of an android application in Kotlin and Java. These languages are inter-operable and can be used together without any issues. For this project, Kotlin was the language of choice as it was more intuitive to understand, program with and easier for a beginner in android development.

For the frontend and UI/UX development, the layout of the app is designed using XML. Additionally the general configuration of the application (build settings/ packages) is also configured using XML.

Figure 2.4.2 shows a sample from the development of the frontend for the ARCape application.

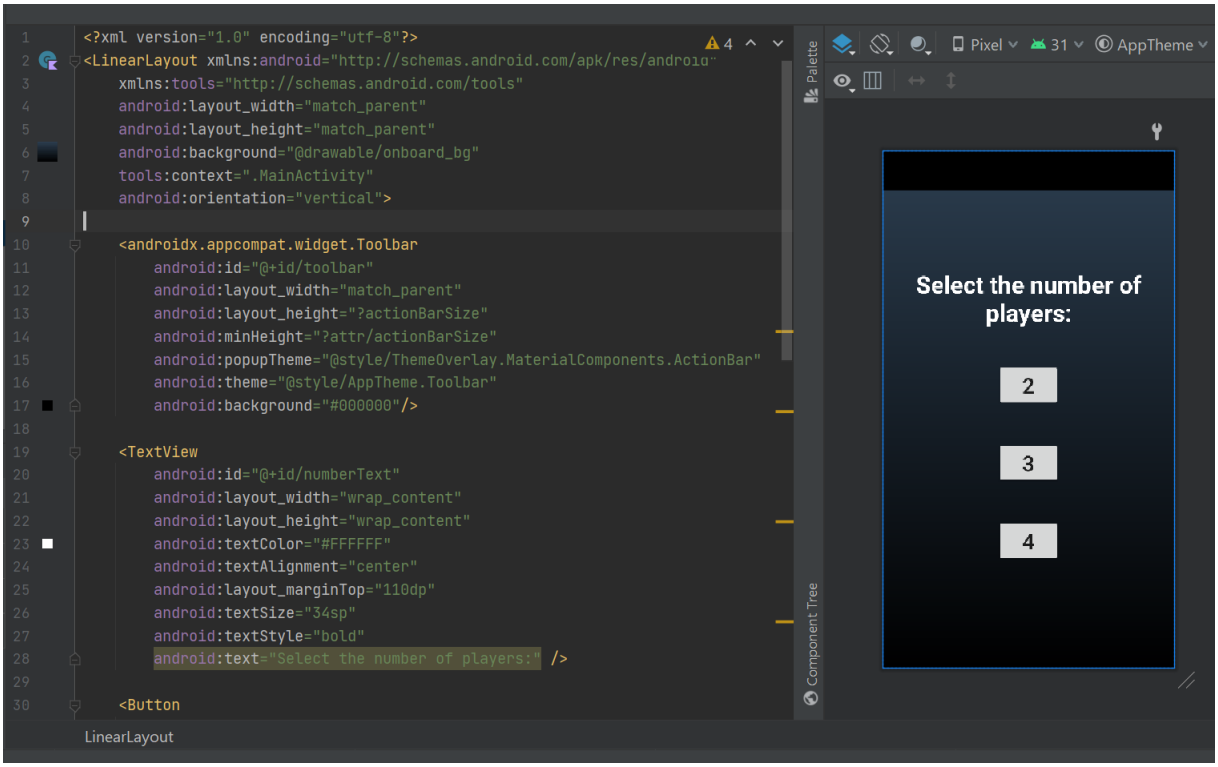


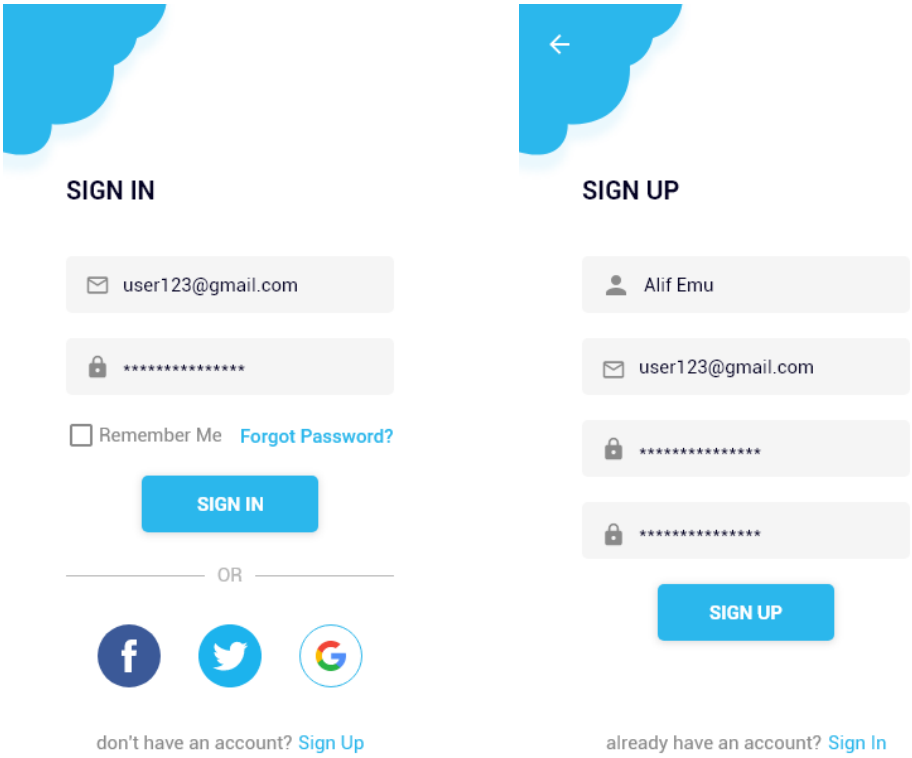
Figure 2.4.2: UI Development in XML

A snapshot of Android Studio showing the development of the Number activity of ARCape application using XML and visualization of the results side-by-side in the IDE.

2.4.4 Android Activity

In android development, activity refers to the screen which is provided to the user for interaction. An android application may consist of multiple activities.

To understand this better lets take the example of the signing up process of an application shown in figure 2.4.3. The figure depicts the the Sign-In and Sign-Up process of an application which requires the user to have an account on the application's platform.



((a)) Sign-In activity **((b))** Sign-Up activity

Figure 2.4.3: Android activities example[1]

If the user already has an account on this application, on opening the app they see the Sign-In activity depicted by ((a)). If the user does not have an account, they can click the *Sign Up* link on the bottom. On doing so, the application switches to the Sign-Up activity depicted by ((b))

2.5 ARCore

ARCore is the platform provided by Google to develop and build applications for AR experiences[7]. ARCore provides a set of API's which enables a smartphone to understand it's environment and use this understanding to provide more information to the user or generate data.

2.5.1 ARCore Session

ARCore session is the most important element of ARCore. It provides the application the ability to access the device's camera, understand it's orientation and enables the application to make use of the ARCore API's.

2.5.2 Augmented Images API

The Augmented Images API provided by ARCore enables a smartphone to detect and augment a 2D image by scanning it. Upon being provided with an image, ARCore utilizes a computer vision algorithm and extracts gray-scale information of the image and stores it in an Augmented Image Database. When an app with Augmented Images API, is in use, ARCore scans for these features on 2D surfaces and and if these features are present it can augment the image based on what is specified through the application.

2.5.3 Image Database

For an ARCore session to recognize an image in it's environment and augment it with the Augmented Images API, it needs to be provided with a set of images. There are two way's to provide theses images to the session:

(i) **Add images to database at runtime:** In this method, the images are added to the database and created at runtime through the application. This is good for testing and development, however, a release version of an application will be slowed down by this method.

(ii) **Use a Preloaded Image Database:** In this method a preloaded database is used. This database is created using the arcoreimg tool. It's significantly faster than creating a database at runtime and is the method of choice for an application in it's release version. The ARCape application makes use of this method.

2.5.3.1 Image Quality Requirements for Augmented Images

In order to successfully recognize and augment an image, the Augmented Images API has certain criterion's[5]:

(i) Color information is not used for detection, as ARcore extracts gray-scale information from the images for detection. Both color and equivalent grayscale images can be used as reference images.

(ii) Images with heavy compression is not recommended as this interferes with feature extraction.

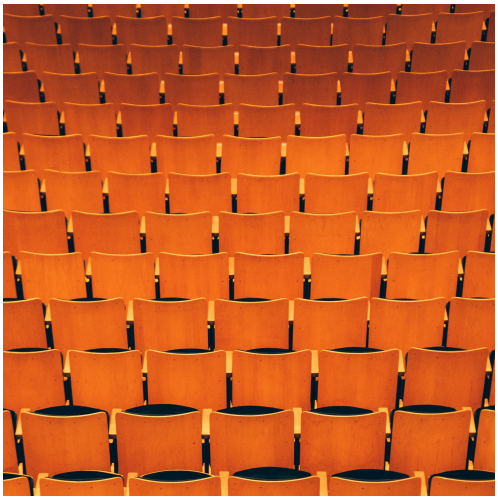
(iii) Avoid images with that contain a large number of geometric features, or very few features (e.g. barcodes, QR codes, logos and other line art) as this will result in poor detection and tracking performance.

(iv) Avoid images with repeating patterns as this also can causes issues with detection and tracking.

(v) Use the arcoreimg tool included in the ARCore SDK to get a quality score between 0 and 100 for each image. A quality score of at least 75 is recommended.

The figure 2.5.1 illustrates the concept of quality score for better understanding.

All the images used in this project have been evaluated with the arcoreimg tool to have a quality score greater than 95.



((a)) Score: 20 [16]

((b)) Score: 95 [15]

Figure 2.5.1: Sample images with quality score

The quality score of the above images were evaluated with the arcoreimg tool. The image ((a)) has repetitive patterns and makes it hard to distinguish features and hence has a lower quality score whereas image ((b)) has better contrasting features which can be recognized easily, and therefore, results in a higher quality score.

2.6 Sceneform SDK

By itself, ARCore cannot import and render 3D models. This is facilitated though the use of Sceneform SDK, which enables the application to import, render and add a 3D model to a scene without the need of OpenGL.

Sceneform was deprecated by Google and made open source in 2020[4]. However, Sceneform Maintained, a community maintained fork of the original Sceneform SDK is available and has the same functionality of Sceneform[2].

2.7 Blender

Blender is a free and open source 3D model creation suite. For this project all the 3D models used were generated in blender. Figure 2.7.1 shows a snapshot of the blender interface.

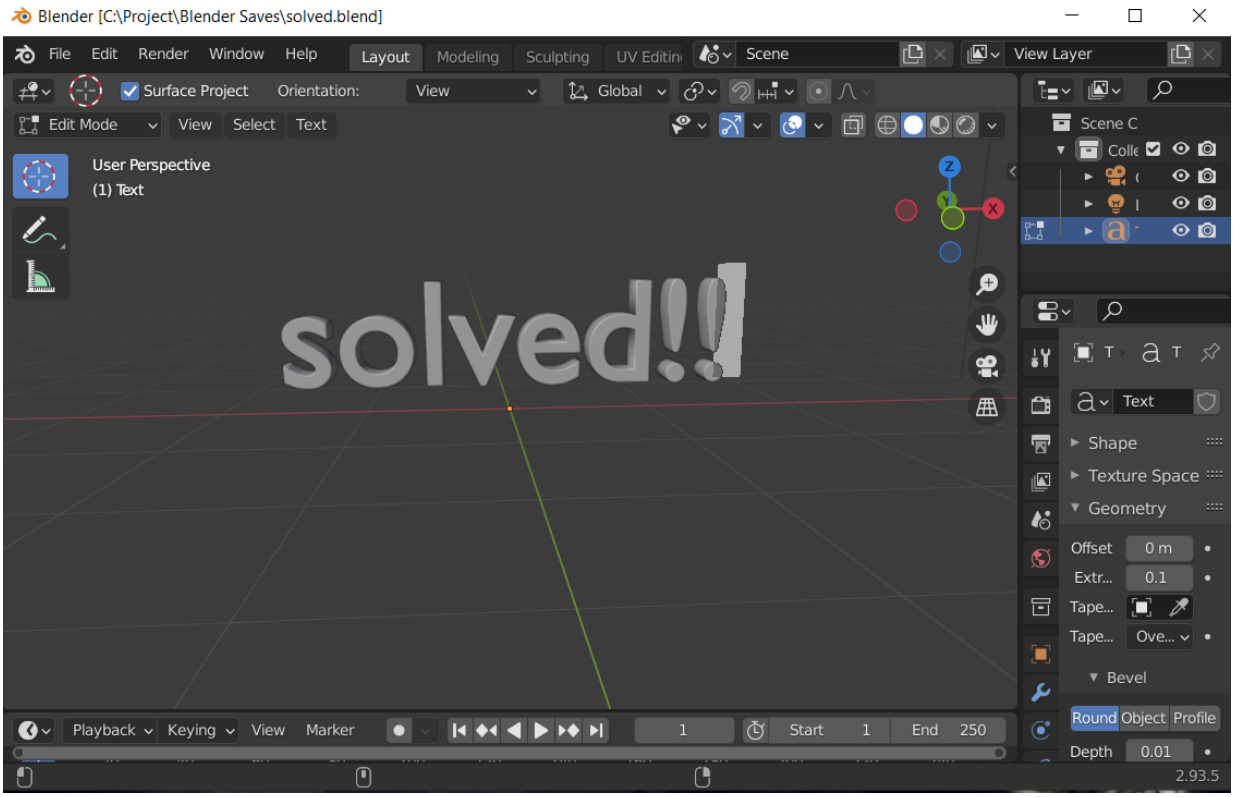


Figure 2.7.1: Blender Interface

A snapshot of blender showing the the development of one of the models used in the ARCape application

2.8 MQTT

MQTT is a lightweight messaging protocol suitable for applications which require low code overhead and have very low bandwidth available[14]. It is primarily used for m2m communication and for IoT applications. It runs on top of a TCP/IP stack and communicates using a publish/subscribe model.

The different puzzles and environment objects in the escape room communicate using the MQTT protocol.

2.8.1 The publish/subscribe model

MQTT makes use of a publish/subscribe model to communicate. There are two types of system in MQTT communication, broker and client. The clients do not directly communicate with each other instead they connect to the broker. The clients can publish messages to the broker, subscribe to the broker for messages or do both. The broker receives messages from the clients and send it to the clients who are subscribed.[14]

Figure 2.8.1 shows a simple visualization of how the clients and the broker interact with each other.

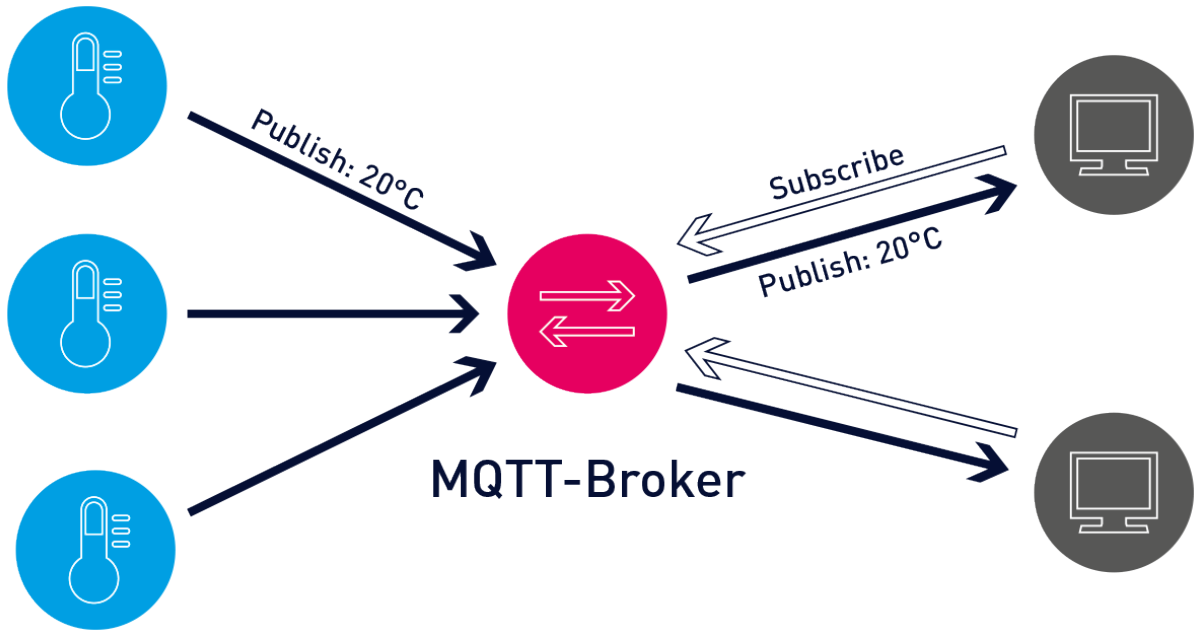


Figure 2.8.1: Visualization of MQTT Protocol[14]

The figure depicts a sensor sending a temperature data to the broker, which then sends it forward to a computer which has subscribed for that data.

2.8.2 MQTT Topics

MQTT messages are not directly published to the broker and not all messages are sent to every subscribed client. Instead the messages are published to specific topics on the broker. The broker then sends these messages to the clients who are subscribed to those particular topics.

The specification of a topic looks somewhat like a file directory structure. For example let's look at a topic that might be used for monitoring the temperature of a furnace on the second floor of a factory. It can be specified as: `factory/secondFloor/furnace/temperature`

Chapter 3

Application Design and Implementation

This section will first give a general overview of the application design and the different steps the application goes though from the time it starts to the time it finishes.

Then, the individual activities of the application will be discussed, along with the functions that those activities are performing, the MQTT messages they publish, the topics they are subscribed to and the additional features that are implemented in those activities.

In the final section of this chapter, the puzzles in the escape room will be discussed, specifically the topics assigned to each puzzle, the hints they show and the messages they react to.

3.1 General Overview of the Application

The different steps the application goes through from launch to finish is depicted in the flow chart shown in the figure 3.1.1

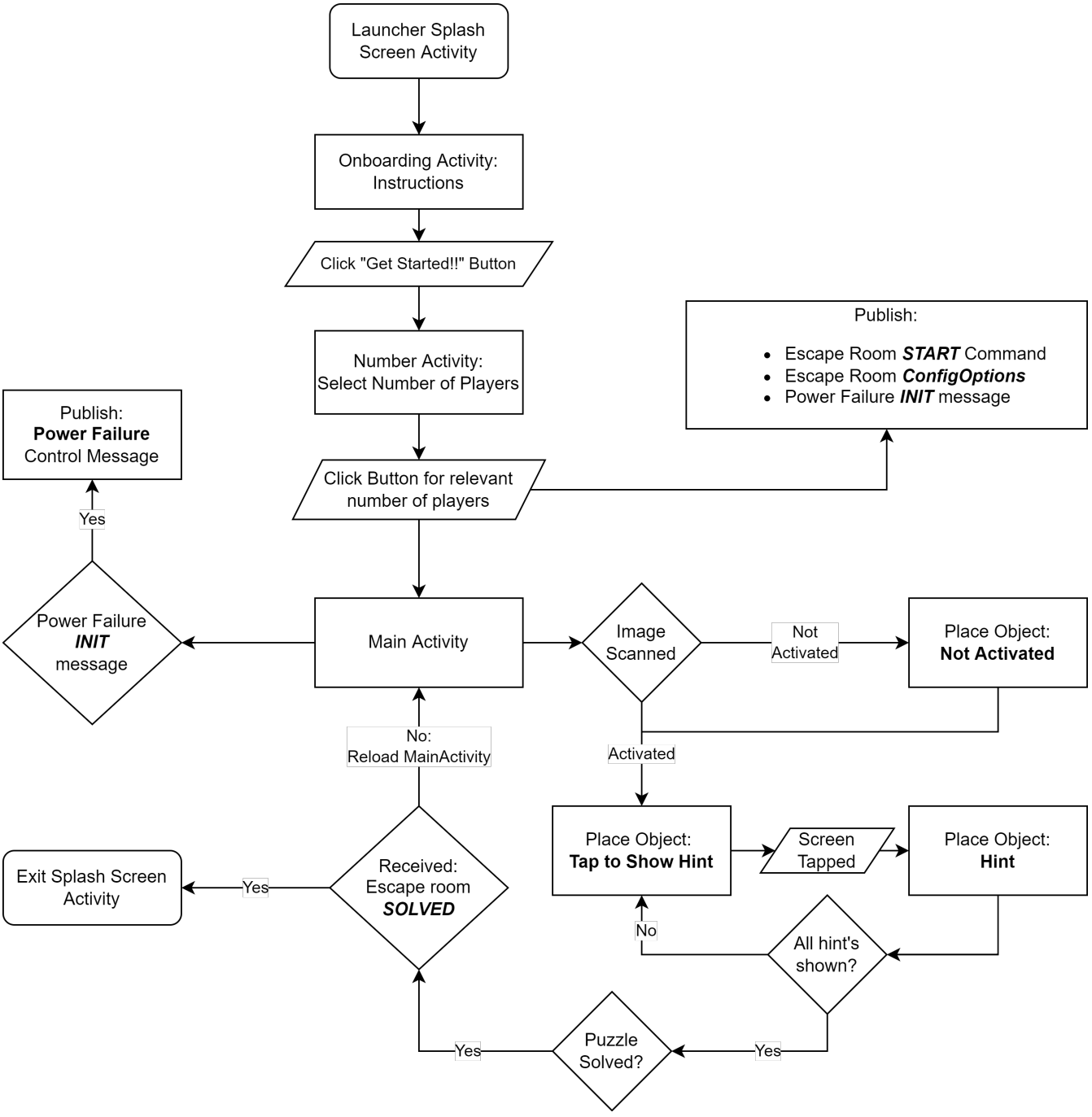


Figure 3.1.1: Flow chart describing the app design

The flow chart describes the activities and the processes that the application goes through and performs from the point when the app is opened to time it exits.

The application starts with the launcher screen (Launcher Splash Screen Activity) and goes to the Onboarding activity. Here the player sees the general instructions on how to use the app, along with an introduction message. On the last page of the Onboarding activity, there is a *GET STARTED!!* button. When this button is clicked, the app moves on to the Number activity. The Number activity facilitates the selection of the number of players participating in the escape room. When the number of players is selected by clicking the relevant button, MQTT messages are published to the escape room control topics to start the escape room along with the escape room configuration. This button click also switches the app to the Main Activity.

The Main Activity contains all the 3D Models for the puzzles, their hints and is also subscribed to the puzzle topics and the escape room control topic. Here the application goes through the different states of the puzzle depending on the messages being received from the MQTT broker. When one puzzle is solved, the Main Activity is reloaded for the next puzzle. This repeats until we reach the final puzzle. Once the final puzzle is solved, the application receives a message from the escape room control topic and the last activity is launched. The last activity, the Exit Splash Screen Activity, is just a basic screen with a congratulatory message. This screen is displayed for a short time and then the application exits.

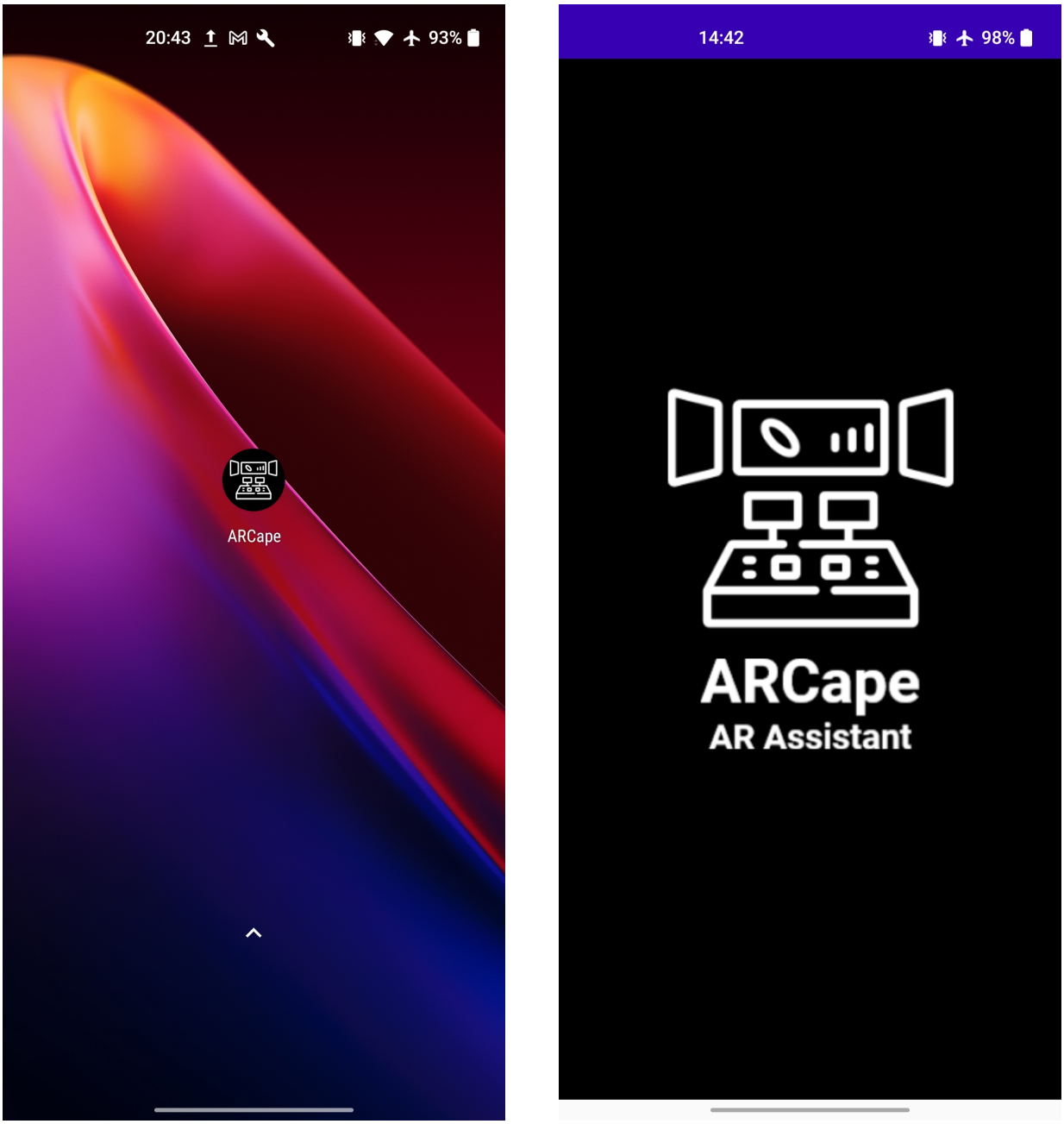
3.2 Android Activities

In this section, the functionality of each android activity of the application and what role they perform will be discussed.

3.2.1 Launcher Splash Screen Activity

The Launcher Splash Screen Activity, is a very basic activity and the first activity which is called when the user opens the application. This activity displays an enlarged version of the app icon with the additional text "**ARCape** AR Assistant" below it. It appears for a duration of 2.5 seconds before switching over to the Onboarding activity.

The view of the application in the home screen (the icon) and the Launcher Splash Screen Activity is shown in Figure 3.2.1



((a)) View in App Drawer

((b)) Launcher Screen

Figure 3.2.1: Launcher Screen from Android Home

Figure ((a)) shows the ARCape application icon in the home screen. When this icon is clicked, the app launches and Figure((b)) shows the Launcher Splash Screen Activity.

3.2.2 Onboarding Activity

The Onboarding activity introduces the application to the user and gives a brief set of instructions. It consists of swipeable pages which were developed

using a viewpager adapter and leads to a *GET STARTED!!* button on the last page. When clicked, the *GET STARTED!!* button switches the application to the Number activity. Along with the viewpager, there is also a dot indicator to indicate which page of the instructions the user is currently on.

The Onboarding activity defines the layout of the activity and the processes to be executed when an action is taken, for example when a button is clicked. The data which is displayed on the Onboarding activity and the swipeable pages are defined though the viewpager adapter. This will be discussed in a subsection 3.2.2.1.

The Onboarding activity consists of four pages, which are shown in figures 3.2.2 and 3.2.3.

3.2.2.1 Viewpager

The Onboarding activity has been developed using the viewpager adaptor, which is the android class that allows users to flip through pages of data in an android application.

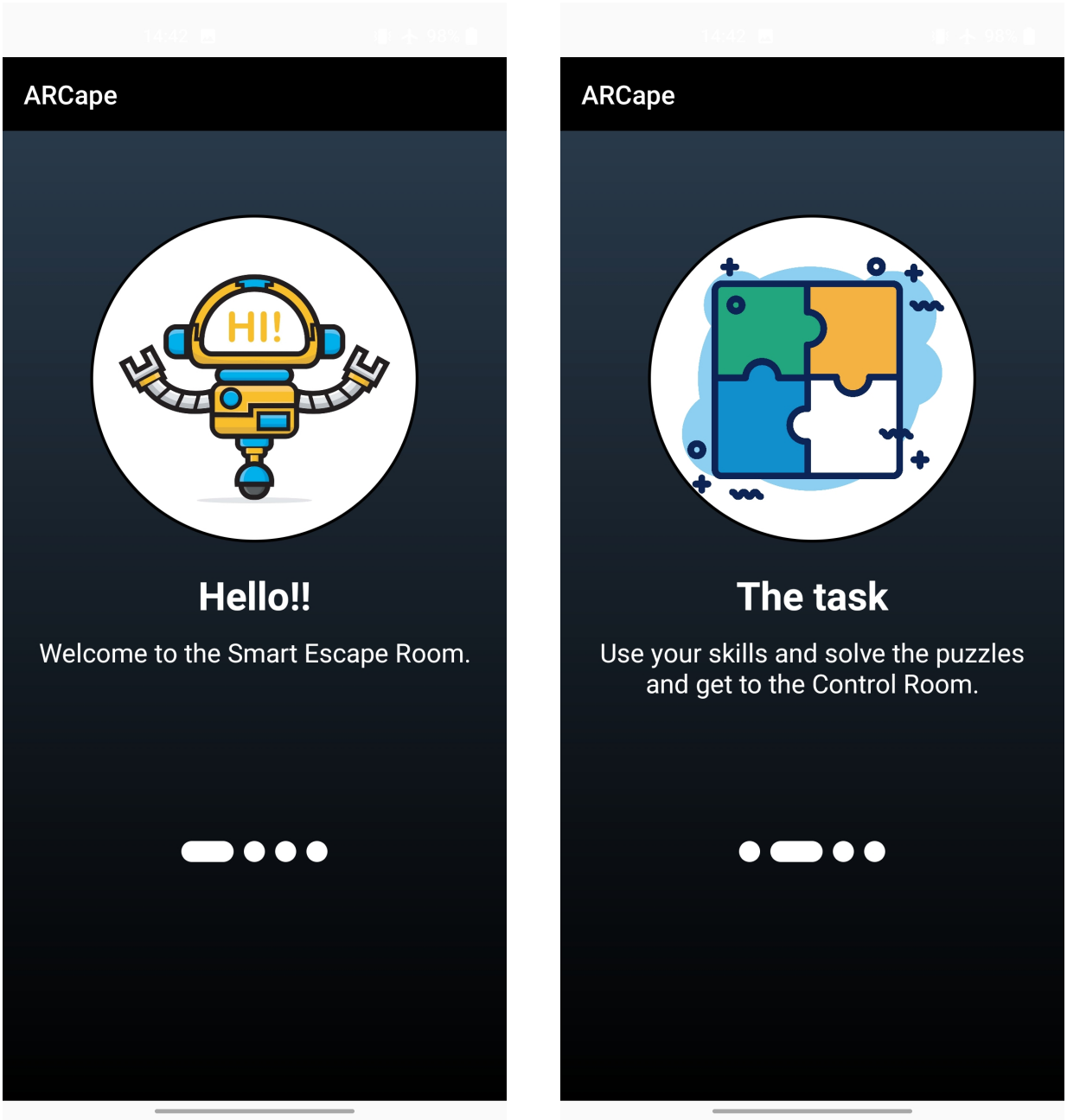
The Onboarding activity only defines the layout of that particular activity and the modes for interaction. The viewpager describes the data that goes on each page. In the case of ARCape application, it defines, the title of page, the accompanying image and the description which goes under the image. Each of these is defined in an array which is passed onto the Onboarding activity and displayed as the user swipes through the pages. The number of pages is controlled by the size of the arrays defined for the individual properties.

3.2.2.2 Dots Indicator

The dots indicator used in the Onboarding activity is a useful visual cue for the user to quickly understand which page of the viewpager they are currently seeing and how many pages there are in total.

3.2.2.3 Get Started!! Button

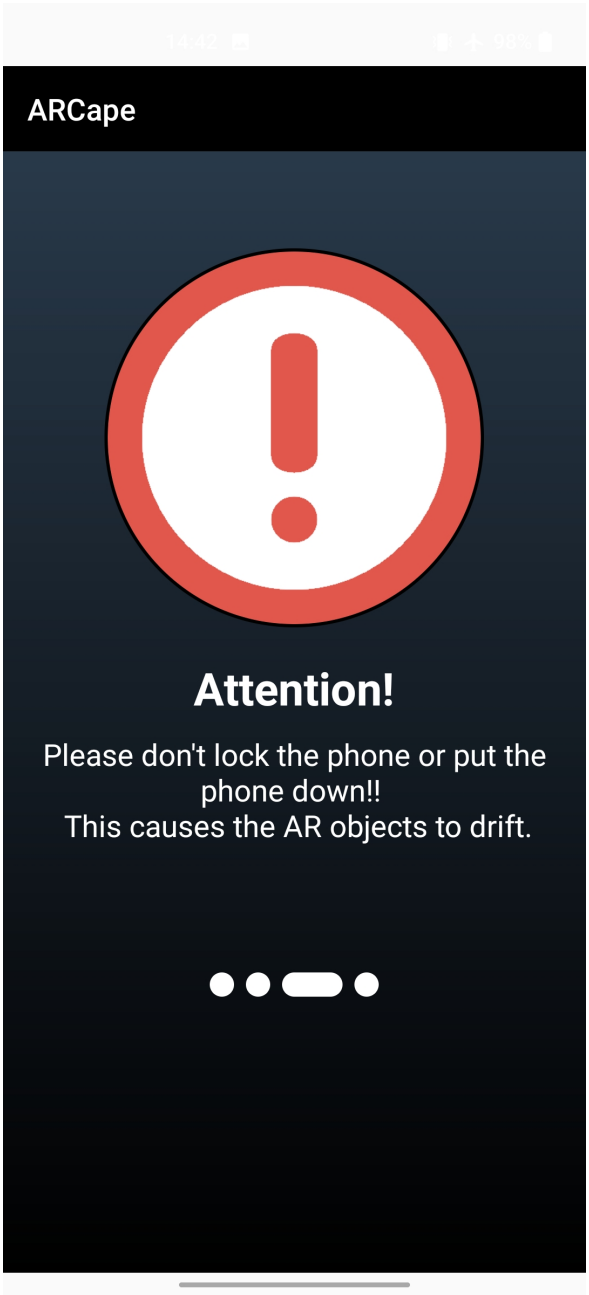
The *GET STARTED!!* button is a generic button which when clicked redirects to the Number activity



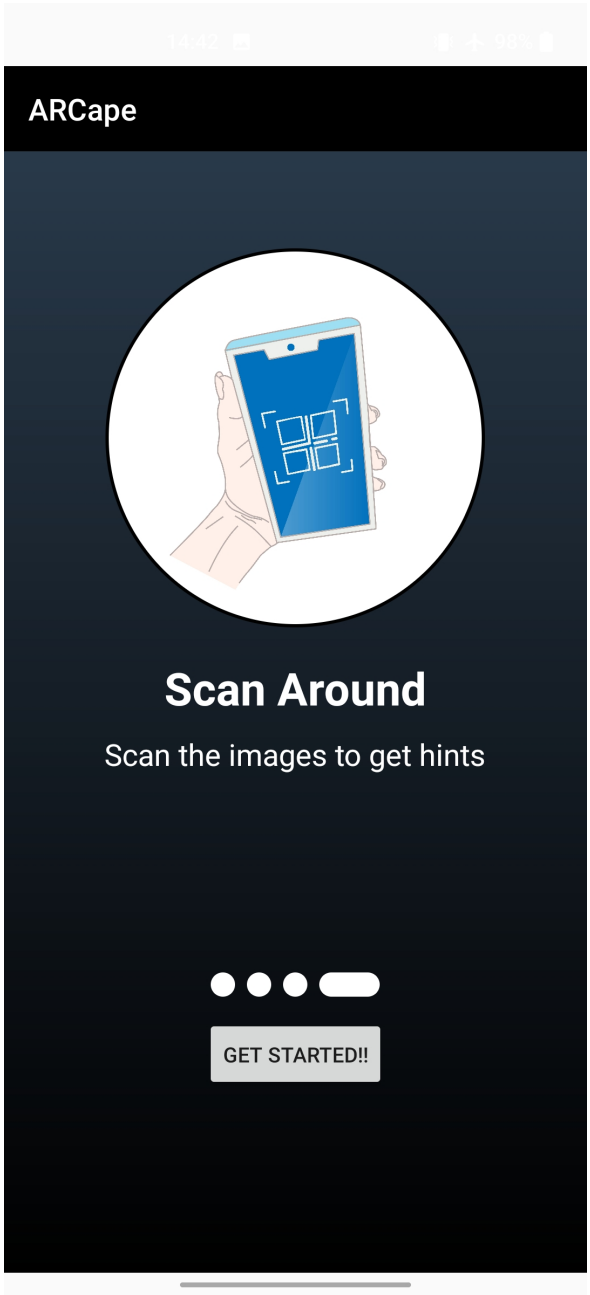
((a)) Page 1: Hello

((b)) Page 2: The Task Page

Figure 3.2.2: Page 1 & 2 of Onboarding activity
Figure ((a)) shows the Welcome message
Figure ((b)) shows the task to be performed in the Escape room



((a)) Page 3: Attention



((b)) Page 4: Scan Around & Get Started

Figure 3.2.3: Page 3 & 4 of Onboarding activity

Figure ((a)) shows an instruction to be followed while playing the room. The relevance of this message will be discussed in the conclusion chapter.

Figure((b)) shows the Scan Around page. Clicking the *GET STARTED!!* button switches the application to the Number activity.

3.2.3 Number Activity

The Number activity lets the user input the number of players participating in the escape room. It is shown in the figure 3.2.4

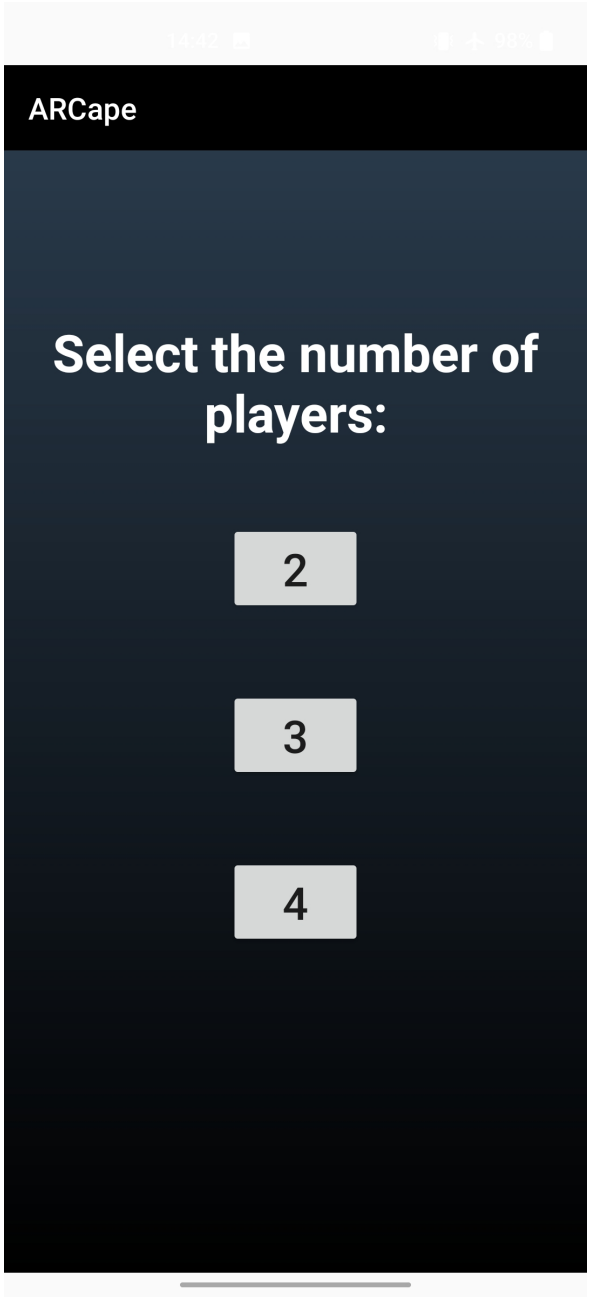


Figure 3.2.4: The Number Activity

The user can select the number of players participating in the escape room. On clicking a button, the START command for the escape room is published along with the escape room configuration options.

The activity instructs the user to select the total number of participants in the escape room. The layout of this activity further consists of three clickable buttons with the numbers 2,3 and 4.

The following actions take place as soon as one of the buttons is clicked:

(i) **Configuration Options:** A JSON message is published to the topic `op/gameOptions`, which looks like so:

```
{
"participants": $participants,
"duration": $duration,
"skipTo": ""
}
```

The purpose of this message is to configure the escape room and set the time according to the number of participants, based on the button clicked. The different combinations with the variable values for the same is described in the table 3.2.1.

Button	$participants	$duration
2	2	90 minutes
3	3	75 minutes
4	4	60 minutes

Table 3.2.1: Description of the different configuration settings for the escape room

This message is retained on the broker. The option `"skipTo"` is from the legacy code which is used to skip to a certain point in the escape room.

(ii) **START message:** The text `START` is published to the topic `op/gameControl`. This is the command which is used for triggering the start of the escape room. Also, this message is retained on the broker.

(iii) **Puzzle Topic Reset:** The text `Not Activated` is published to all the puzzle topics. this is done to reset the puzzle topics to the *not activated state* before the game play begins.

(iv) **Power Failure INIT:** The message `INIT` is published to the topic `env/powerFail`. This command is used by the Main Activity to ini-

tiate the power failure in the escape room and will be discussed further in section 3.2.4.1. This message is also retained on the broker.

3.2.4 Main Activity

The Main Activity is the core of the ARCape application and performs the function of recognizing an image using the Augmented Images API, getting the internal puzzle state by subscribing to the puzzle topic associated with the image and augmenting it by placing a 3D rendered object on it using Sceneform.

The activity starts by configuring an ARCore session with the preloaded image database containing the images associated with the puzzles. This ARCore session gives the application the access to the camera of the android device being used. Through the camera, the application looks for the features which are present in the images (from the image database) in the device environment and tries to recognize them. As soon as an image is recognized, the internal state of the puzzle is received though an MQTT message obtained by subscribing to the relevant puzzle topic. Once this MQTT message is obtained, an AR object is then placed on the image depicting the internal state of the puzzle.

As described before, there are three internal states a puzzle can have. Let's look at the example of puzzle 4 and go through each of these states and understand what happens in each state till the puzzle is solved and how the Main Activity facilitates this.

The Main Activity is subscribed to all the puzzle topics, but since we are looking at puzzle 4, lets look at the relevant topic for puzzle 4 i.e. `game/puzzle4` and see what are the messages available on the topic during game play:

(i) **Not activated state:** The puzzle is not activated hence not playable. The message on the puzzle topic during this time is `Not Activated`. When the image is scanned during this time the AR object projected is the text "Not Activated" as shown in figure 3.2.5

Figure 3.2.5: Not Activated State

When the image has been scanned and the puzzle has not been activated yet, the AR object displayed is the text "Not Activated".

(ii) **Activated state:** When the puzzle has been activated, the message `Activated` or the puzzle specific activated message is published to the puzzle topic `game/puzzle4`. If the image had already been scanned, then in this state the AR object is changed to "Tap to show hint", or if the image is scanned for the first time, this AR object will be

projected. When the user taps on the screen, the AR object is changed to the relevant hint for the puzzle which in this case is “push down red button”. Figure 3.2.6 illustrates this.

((a)) “Tap to show hint” **((b))** “Hint”

Figure 3.2.6: Activated State

Figure ((a)) shows the AR object “Tap to show hint”. When the screen is tapped by the user in this situation, the AR object is changed to the hint.

Figure((b)) shows the hint for Puzzle 4 after the user has tapped on the screen.

(iii) **Solved state:** When the puzzle has been solved, the message `SOLVED` is published immediately on the topic `game/puzzle4`. On receiving this message the Main Activity reloads itself, this is done in order to clear up the memory and dispose of the rendered models which are no longer useful for the next set of puzzles. When the image is scanned in this state, the AR Object "solved!!" is projected as shown in figure 3.2.7. This object is removed five seconds after the projection as it is not relevant to the game play anymore.

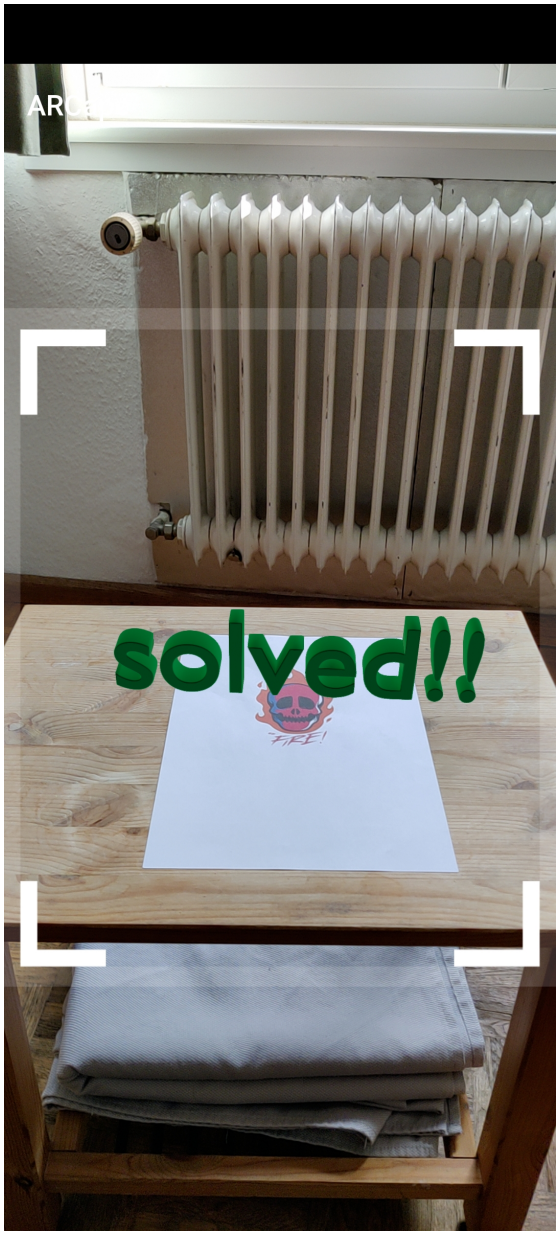


Figure 3.2.7: Solved State
When the image is scanned, the object "solved!!" is projected and removed after five seconds.

3.2.4.1 Initiating Power Failure

When the Main Activity is loaded for the first time, it subscribes to the topic `env/PowerFail` to see if the payload is `INIT`. If this condition evaluates to true, after five seconds the following JSON message will be published to the topic `env/powerFail`:

```
{
    "method": "status",
    "state": "solved"
}
```

This message is retained on the broker. When this message is received by the broker, it switches off the lights in the escape room and simulates a power failure scenario. When the main activity is reloaded again, this message won't be published anymore as now the retained message has changed.

3.2.4.2 Tap to touch listener

The tap to touch listener is used to evaluate the user tap on the screen in the *Activated* state of the puzzle. This function listens for user taps on the screen and when a tap is recorded and the AR object spawned is "Tap to show hint", the AR object is changed to the hint.

3.2.4.3 Vibration Manager

Whenever a new hint is available within the escape room, the device vibrates. This is facilitated by the Vibration Manager, which triggers a vibration when a new hint is available.

3.2.5 Exit Splash Screen Activity

When the escape room is solved completely within the time limit, we get the message `SOLVED` on the topic `op/gameControl`. The Main Activity is subscribed to this topic, and the moment we receive the `SOLVED` message, the Main Activity exits and launches the Exit Splash Screen Activity. The Exit Splash Screen Activity displays a message congratulating the user for solving the escape room. The layout for this activity is shown in figure 3.2.8.

Figure 3.2.8: The Number Activity

The congratulatory message shown to the user on successfully completing the escape room puzzles within the time limit.

3.3 Puzzles

This section will discuss the puzzle topics and the messages that are published to the puzzles in order to trigger the hint along with the image associated with the puzzle, which is to be scanned to see the state of the puzzle/hints.

3.3.1 Puzzle 1

Puzzle 1 had three parts so, and after discussions with the team, it was decided that there would be three hints for this puzzle. All the states of the puzzle including the three hints would be triggered by publishing messages to the topic `game/puzzle1`.

The image associated with Puzzle 1 is a pixelated cube. This image was chosen as the central element in the puzzle is a cube. The image is shown in figure 3.3.1

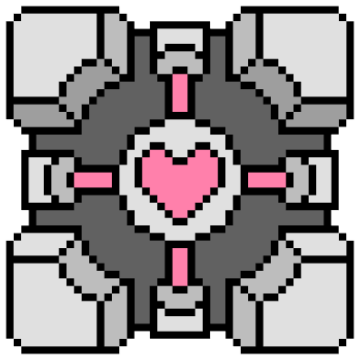


Figure 3.3.1: Puzzle 1: Cube Image[19]
The internal state of Puzzle 1 and hints can be obtained by scanning this image.

The messages that need to be published and the text displayed through the AR object along with the puzzle state is described in the table 3.3.1 below:

Puzzle State	**Message Published**	**Text Displayed**
Not Activated	Not Activated	Not Activated
Activated	Hint1	Tap to show hint
		On tap: find panels
Activated	Hint2	Tap to show hint
		On tap: get correct sequence
Activated	Hint3	Tap to show hint
		On tap: complete path
Solved	Solved	solved!!

Table 3.3.1: Description of puzzle state, messages to be published and text displayed on AR object for Puzzle 1

3.3.2 Puzzle 2

Puzzle 2 had just one hint, which would be triggered on activating the puzzle itself. All the states of the puzzle and the hint would be triggered by publishing messages to the topic `game/puzzle2`.

The image associated with Puzzle 2 is an animated clock. This image was chosen based on the recommendation made by the team for a picture of a clock or watch. The image is shown in figure 3.3.2

Figure 3.3.2: Puzzle 2: Animated Clock[9]
The internal state of Puzzle 2 and hints can be obtained by scanning this image.

The messages that need to be published and the text displayed through the AR object along with the puzzle state is described in the table 3.3.2 below:

Puzzle State	**Message Published**	**Text Displayed**
Not Activated	Not Activated	Not Activated
Activated	Activated	Tap to show hint
		On tap: ecsape 100111
Solved	Solved	solved!!

Table 3.3.2: Description of puzzle state, messages to be published and text displayed on AR object for Puzzle 2

3.3.3 Puzzle 3

This was the puzzle with the most number of parts and there were several different hint and messages published based on the part of the game which

was being played. Each hint was triggered when a part of this particular puzzle was completed. All the states of the puzzle and the hint's would be triggered by publishing messages to the topic `game/puzzle3`.

The image associated with Puzzle 3 is an animated radio. This image was chosen as Puzzle 3 revolved around a radio. The image is shown in figure 3.3.3

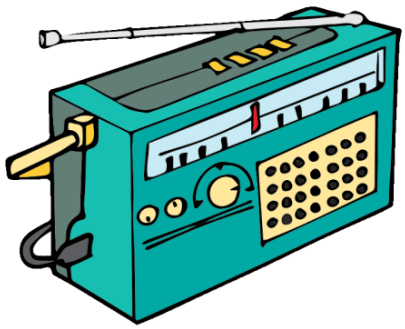


Figure 3.3.3: Puzzle 3: Animated Radio[10]
The internal state of Puzzle 3 and hints can be obtained by scanning this image.

The messages that need to be published and the text displayed through the AR object along with the puzzle state is described in table 3.3.3 below:

Puzzle State	**Message Published**	**Text Displayed**
Not Activated	Not Activated	Not Activated
Activated	antenna_activate_1	Tap to show hint
		On tap: fix radio
Activated	map_activate_1	Tap to show hint
		On tap: 3 locations
Activated	map_activate_2	Tap to show hint
		On tap: check time zones
Activated	touch_activate_1	Tap to show hint
		On tap: touch to make a word
Activated	touch_activate_2	Tap to show hint
		On tap: listen!!
Activated	touch_activate_3	Tap to show hint
		On tap: not automatic
Solved	Solved	solved!!

Table 3.3.3: Description of puzzle state, messages to be published and text displayed on AR object for Puzzle 3

3.3.4 Puzzle 4

Like Puzzle 2, Puzzle 4 has just one hint, triggered on activating the puzzle itself. All the states of the puzzle and the hint would be triggered by publishing messages to the topic `game/puzzle4`.

The image associated with Puzzle 4 is a skull on fire. This image signifies danger and was chosen because this puzzle is made up of warning symbols all over the escape room. The image is shown in figure 3.3.4

Figure 3.3.4: Puzzle 4: Skull on Fire[18]
The internal state of Puzzle 4 and hints can be obtained by scanning this image.

The messages that need to be published and the text displayed through the AR object along with the puzzle state is described in the table 3.3.4 below:

Puzzle State	Message Published	Text Displayed
Not Activated	Not Activated	Not Activated
Activated	Activated	Tap to show hint
		On tap: press down red button
Solved	Solved	solved!!

Table 3.3.4: Description of puzzle state, messages to be published and text displayed on AR object for Puzzle 4

3.3.5 Puzzle 5

Like Puzzle 2 and 4, Puzzle 5 also has just one hint, triggered by activating the puzzle itself. All the states of the puzzle and the hint would be triggered by publishing messages to the topic `game/puzzle5`.

The image associated with Puzzle 5 is a retro gamepad icon. This image fits in with this puzzle as a video game needs to be played to complete this puzzle. The image is shown in figure 3.3.5

Figure 3.3.5: Puzzle 4: Retro Gamepad Icon[17]
The internal state of Puzzle 5 and hints can be obtained by scanning this image.

The messages that need to be published and the text displayed through the AR object along with the puzzle state is described in the table 3.3.5 below:

Puzzle State	**Message Published**	**Text Displayed**
Not Activated	Not Activated	Not Activated
Activated	Activated	Tap to show hint
		On tap: play 2048: open lock
Solved	Solved	Solved

Table 3.3.5: Description of puzzle state, messages to be published and text displayed on AR object for Puzzle 5

Chapter 4

Results and Conclusion

This chapter concludes the report.

The main objective of this report was to develop a hint system which could help the participants of the escape room to progress through the puzzles and get suitable hints. The primary result of this project was the development of an android application which helps the participants of the escape room progress through the puzzles by providing suitable hints through the use of augmented images and AR projection overlays. Additionally, the application also starts the escape room and initiates the power failure in the room. The application is already deployed on the test device (OnePlus Nord) and functioning properly.

There are, however, some known issues. The projected AR objects sometimes tend to drift away and move a lot and this sometimes results in the object drifting and appearing as if it's sticking to the camera module of the device, if the phone is locked and moved or shaken really fast. This is a known issues and has been reported to Google several times over the past years as can be seen in Github Issue 508, Github Issue 624, Github Issue 492, Github Issue 528 and Github Issue 528. Most of these issues have been closed without a proper response. This is also the reason why the *Attention* page was added to the Onboarding activity.

Regarding future scope, the app can be easily be extended to implement a leader board based on the amount of time taken and the hints used by the participants to solve the escape room puzzles. This will also add an element of competition to the whole setup and the android application also provides

the platform to implement this. Additionally the AR object drifting issues can also be corrected, if a solution is given by Google in the future versions of ARCore.

From this project, I had several learning outcomes. I was completely new to android development, and over the course of this project I was able to learn all the basic aspects of Android application development, in addition to learning a new programming language, Kotlin. I, now, have the confidence to write and design small applications myself. I also have a new found respect due to the amount of time, energy and brainstorming it takes to find bugs, solve issues and implement feasible solutions. Additionally, it was the first time I had to write an entire report in LaTeX myself and this was a good learning opportunity for the same.

The source code for this project can be found at the following Github repository: ARCape or the mirrored repository in the UbiLab Repository

The Android package is present at : ARCape Release Version

Bibliography

[1] Emu. Sign in and sign up page android. https://www.uplabs.com/posts/sign-in-and-sign-up-page-android-b909ac81-0906-4dcc-8270-fef80bfef7c9, 2020. Last accessed 16th February 2022.

[2] Thomas Gorrise et. al. Sceneform maintained. https://github.com/SceneView/sceneform-android, 2020. Last accessed 20th February 2022.

[3] Calum Gathergood. Augmented Images with ARCore and Sceneform. https://proandroiddev.com/augmented-images-with-arcore-and-sceneform-4c3fe774b5bd, 2018. Last accessed 17th February 2022.

[4] Google. Sceneform overview. https://developers.google.com/sceneform/develop, 2020. Last accessed 10th February 2022.

[5] Google. Add dimension to images. https://developers.google.com/ar/develop/augmented-images, 2022. Last accessed 16th February 2022.

[6] Google. The arcoreimg tool. https://developers.google.com/ar/develop/augmented-images/arcoreimg, 2022. Last accessed 17th February 2022.

[7] Google. Overview of ARCore and supported development environments. https://developers.google.com/ar/develop, 2022. Last accessed 9th February 2022.

[8] Adam Hayes. Augmented reality. https://www.investopedia.com/terms/a/augmented-reality.asp, 2020. Last accessed 30th January 2022.

[9] jf staeulalia.pt. Pixel art retro alarm clock. https://www.shutterstock.com/image-illustration/pixel-art-retro-alarm-clock-8bit-1786206233, 2022. Last accessed 20th February 2022.

[10] Kaleb-Silva. Kostenlose radio cartoon cliparts. https://jf-staeulalia.pt/img/other/06/collection-radio-cartoon-cliparts.png, 2022. Last accessed 20th February 2022.

[11] Vladimir Kuskov. New trends in the world of IoT threats. https://securelist.com/new-trends-in-the-world-of-iot-threats/87991/, 2018. Last accessed 16th February 2022.

[12] Robert Huy Le. Several examples of AR applications. https://www.researchgate.net/figure/Several-examples-of-AR-applications_fig3_326216037, 2017. Last accessed 16th February 2022.

[13] Lexico. Augmented reality. https://www.lexico.com/definition/augmented_reality, 2022. Last accessed 30th January 2022.

[14] Paessler. IT explained: MQTT. https://www.paessler.com/it-explained/mqtt, 2022. Last accessed 9th February 2022.

[15] Rico Reutimann. Person holding grey glass ball. https://unsplash.com/photos/4ZDe5huKzjg, 2018. Last accessed 17th February 2022.

[16] Andre Seaman. Empty theatre. https://unsplash.com/photos/Y8ruVPHUSnc, 2018. Last accessed 17th February 2022.

[17] stockgiu. Pixelated retro gamepad. https://www.vecteezy.com/vector-art/650331-pixelated-retro-gamepad, 2018. Last accessed 20th February 2022.

[18] Lincung Studio. A read head skull on the blaze. https://www.vecteezy.com/free-vector/skull-fire, 2022. Last accessed 20th February 2022.

[19] Unknown. Weighted Companion Cube. https://www.pixilart.com/art/weighted-companion-cube-1cb9d4242c41a49, 2018. Last accessed 20th February 2022.