

Augmented Reality app that educates the user about the lunar surface

By

Jonathan Roddy

Student ID: K00212118

Course: LC 233 Software Development

Supervisor: Carol Rainsford

Date: January 2021

**Table of Contents**

[1 Introduction 3](#_Toc62830186)

[2 Software Chapter: Design 4](#_Toc62830187)

[2.1 Introduction 4](#_Toc62830188)

[2.2 Requirements 5](#_Toc62830189)

[2.2.1 Project Requirements 5](#_Toc62830190)

[2.2.2 Non-Functional Requirements 6](#_Toc62830191)

[2.2.3 Supportability List of Actors 6](#_Toc62830192)

[2.2.4 List of Use Cases 6](#_Toc62830193)

[2.3 Android Design 8](#_Toc62830194)

[2.3.1 Material design 8](#_Toc62830195)

[2.3.2 Fragments 9](#_Toc62830196)

[2.4 Diagrams 10](#_Toc62830197)

[2.4.1 Use Case Diagram 11](#_Toc62830198)

[2.4.2 Use Case Documentation 13](#_Toc62830199)

[2.4.2.1 Use case 1. 13](#_Toc62830200)

[2.4.2.2 Use case 2. 14](#_Toc62830201)

[2.4.2.3 Use case 3. 15](#_Toc62830202)

[2.4.3 Sequence Diagram 16](#_Toc62830203)

[2.4.3.1 Dashboard 16](#_Toc62830204)

[2.4.3.2 Augmented Reality Moon Model 0](#_Toc62830205)

[2.4.3.3 0](#_Toc62830206)

[2.4.3.4 Moon Phase 19](#_Toc62830207)

[2.4.4 Class Diagram 20](#_Toc62830208)

[2.4.5 Flowchart 22](#_Toc62830209)

[2.4.6 Application Wireframe 24](#_Toc62830210)

[2.5 Conclusion 28](#_Toc62830211)

[3 Software Chapter: Implementation 29](#_Toc62830212)

[3.1 Introduction 29](#_Toc62830213)

[3.2 Project Management 30](#_Toc62830214)

[3.2.1 Management tools 30](#_Toc62830215)

[3.2.2 OneNote 30](#_Toc62830216)

[3.2.3 GitHub 30](#_Toc62830217)

[3.2.4 Task list 31](#_Toc62830218)

[3.3 Learning obstacles 34](#_Toc62830219)

[3.3.1 Certifications 35](#_Toc62830220)

[3.4 Development Process 37](#_Toc62830221)

[3.4.1 AR Model Creation 37](#_Toc62830222)

[3.4.1.1 Blender 37](#_Toc62830223)

[3.4.2 Database implementation 37](#_Toc62830224)

[3.4.3 Android Development 37](#_Toc62830225)

[3.4.4 Text-to-Speech Implementation 37](#_Toc62830226)

[3.5 System Structure 37](#_Toc62830227)

[3.6 Completed System 38](#_Toc62830228)

[3.7 Conclusion 38](#_Toc62830229)

[4 Bibliography 39](#_Toc62830230)

**Table of Figures**

[**Figure 1:** Software Development Lifecycle (Kwon, 2019) 5](#_Toc62830231)

[**Figure 2:** Android Lifecycle (Masters, 2013) 8](#_Toc62830232)

[**Figure 3:** Android Fragments (mit, 2021) 9](#_Toc62830233)

[**Figure 4:** Use Case Diagram 12](#_Toc62830234)

[**Figure 5:** Sequence Diagram Dashboard 16](#_Toc62830235)

[**Figure 6:** AR Moon Model 0](#_Toc62830236)

[**Figure 7:** Sequence Diagram Moon Phase 19](#_Toc62830237)

[**Figure 8:** Class Diagram 21](#_Toc62830238)

[**Figure 9:** Application Flowchart 23](#_Toc62830239)

[**Figure 10:** Splash Screen, Home page and Settings 25](https://studentlit.sharepoint.com/sites/SD4-FYP-Jonathan/Shared%20Documents/Jonathan/Design%20&%20Implementation/K00212118_JonathanRoddy_DesignImplementation%20.docx#_Toc62830240)

[**Figure 11:** Augmented Reality Page 26](#_Toc62830241)

[**Figure 12:** Discover Page 27](#_Toc62830242)

[**Figure 13:** Moon Phase Page 28](#_Toc62830243)

[**Figure 14:** LinkedIn Learning Cert 35](#_Toc62830244)

[**Figure 15:** Android Development UI with Java 36](#_Toc62830245)

[**Figure 16:** UX for Non-Designers 37](#_Toc62830246)

**Table of Tables**

[**Table 1:** Task Management List 33](#_Toc62825149)

# Introduction

In this paper, I aim to continue in the development of my final year project by moving on to the design and implementation sections. By having done both my literature review and my research paper I am no equipped with tackling this stage. To work on this project while adhering to software development's best practices, I will be developing under the agile methodology. Agile is currently the best way to develop a successful project without the worries of code scrap of code redundancy. It states that both the customer and developers have constant communication with one another and are updated frequently. Agile is an iterative approach so during the design and implementation drawings and ideas might change and that is to be expected. What developers used to utilize was a waterfall, but this led to many issues in the development as things would be designed and implemented without the thought of updates. From being able to develop a group project in the third year I was able to utilize the spiral waterfall method which I will use some parts of this as well. In my work placement, I was taught how to life SCRUM with the aid of agile and this will benefit this project's lifecycle a lot. SCRUM is a business lifecycle were a team meet up daily/ twice a week to talk about any issues that one might have so that the whole team is included in one another’s task. They can offer insight of help if any issues arise that they can tackle together. I will use this to write down any issues I have in the development stage so that I can see what tasks or issues came to be so that I can later investigate them further. I will record each action that comes up during this section so that I can have a compressive archive of the project’s lifecycle.

For this project, I am playing the role of customer and the developer so I can easily iron out any issues or faults that must be addressed. Since coming up with this project I have been thinking about the look and feel a lot as I knew this would take a considerate amount of time to perfect. This project is a limited resource project so I know that I may not have everything complete in time, but I will try my best to meet my set timelines. Knowing what I have gathered already I will aim to develop further on the requirements, diagrams like use case, class diagram sequence diagrams, and more.

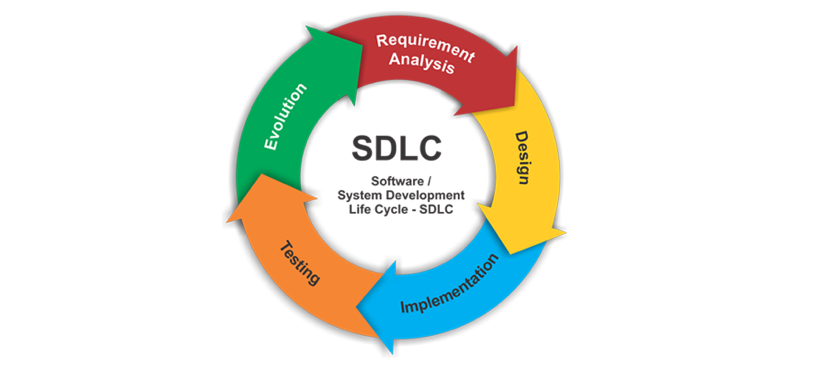
# Software Chapter: Design

## Introduction

This chapter will document my journey in continuing further into my project’s lifecycle. So far, I have talked about Augmented Reality and Text-to-Speech in my literature review, I have talked about the technologies that I will need for this project as well as the overall theme as well. I now wish to talk about the design of the project. I have gathered some good resources ahead of implementation so this gives me a good chance in completing what I sought out to accomplish. As previously mentioned, I completed a group project in my third year which enabled me to work as a team to accomplish a project within a timescale. I was able to use design elements that I had learned in the previous years to better grow my knowledge and familiarisation with them. This form of exposure granted me the best practice for my final year project as I intend to use those diagrams here.

At the beginning of this project, I was just starting my final year at Limerick Institute of Technology and knew what this project was going to entail. I was lucky enough to be able to do the mobile application elective which I knew would help me a lot for this project. I went from having no prior experience with designing and developing an android application to now having completed that module. I will use the knowledge I learned in that module to my advantage as I know have had a chance to use Android studio and Google Firebase for a separate project I needed to complete. I learned about UX and UI design and how to best utilize a device for all its components like camera, screen, and notifications.

As Mentioned before this project is being developed under the agile methodology so I expect updates and iterations of one design until the final project is completed. I will use the feedback I receive from my supervisor as well as my research to be able to draw out a successful plan. As an inspiring Software Developer, I have learned that following in the Software Development Lifecycle (SDLC) is the best way in which I can complete any project. This lifecycle consists of five stages that we must follow for the best possible outcome. As seen in Figure 1 below it is made from these five stages are Requirements, Design, Implementation, Testing, and Evolution. This diagram is a great representation of the SDLC as arrows are connecting each one into the next. This means that once we are at the evolution stage we are not finished; we continue again to make sure all aspects have been looked at for further improvements.

**Figure 1:** Software Development Lifecycle (Kwon, 2019)

## Requirements

Like any software project lifecycle, the requirements stage is vital as that is the starting block to which the project can begin. Any requirement can be given by the client at the early stages but can come at any stage of the project’s lifecycle. If requirements are skipped or not properly looked after, then that may lead to customer dissatisfaction as well as a poorly design application. I will write down the requirements that this project needs to have so that I can pull use cases from those. I will also write down non-functional requirements what was not specified. These non-functional requirements relate to aspects of the application that the customer has not asked for, but we know as developers that they need to be included. This thing can be security, scalability, privacy which the customer would have thought about in the early stage.

### Project Requirements

* An android application running on the latest version of Android.
* Free to download the application.
* Utilize edutainment soul for the user to learn all aspects of the Moon.
  + Actuate images and facts about the Moon.
  + Have a Moon phases section with inputted dates from the user.
* Implements Augmented Reality to help engage with the user.
  + Have an accurate model of the Moon available.
  + Have markers on the moon showing interesting pointers and locations.
* Implements the library Text-to-Speech to aid in the immersion.
* Applies best UI and UX approaches
* Implements new Android Development features
* Available for all screen sizes under the given Android API
* Have a section where the user can scroll through facts, videos, and images.

### Non-Functional Requirements

* Security
* Scalability
* Ease of use
* Reliability
* Privacy
* Resilience
* Platform compatibility
* Performance / response time
* Testability
* Usability

### Supportability List of Actors

* User
* Google Firebase

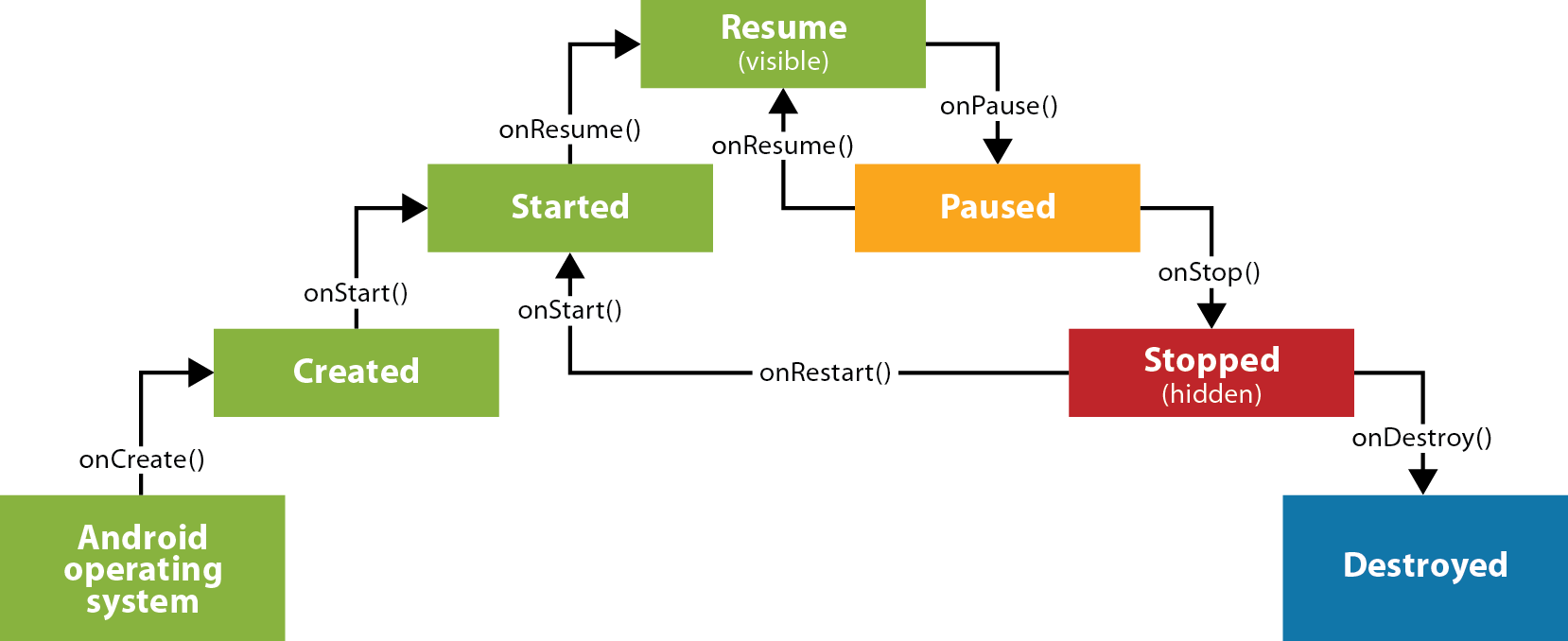
### List of Use Cases

* Install the android application.
* Connecting to Wifi
* Not connected to Wifi
* Check the Phone version for compatibility.
* The phone is not compatible with ARCore.
* The phone is compatible with ARCore.
* Check android screen size.
* Log exceptions
* View Application home page dashboard
* Use the links on the dashboard to open AR screens.
* Ask the user for access permission on the rear camera.
* Connect to Firebase database
* Database connection not found.
* The database sends the Moon model to the system.
* The database sends the Earth model to the system.
* The database sends Moon Portal to the system.
* Render Augmented Reality Moon Model
* Render Augmented Reality Earth Model
* Render Augmented Reality Portal Model
* View Augmented Reality Moon Model
* View Augmented Reality Earth Model
* View Augmented Reality Portal Model
* The user interacts with the Moon model.
* A user interacts with the Earth model.
* The user interacts with the Portal model.
* Display pens, markers, and names on augmented models.
* Use Text-to-Speech to call out text on each model.
* View Discovery page
* Retrieve video links from the Firebase database.
* Retrieve data from the Firebase database.
* Use Text-to-Speech on retrieved data.
* View Moon phase page
* Calculate Moon Phase by a given date.
* Update Moon phase screen

## Android Design

When designing this application, I need to think about what I have learned during my mobile development elective so that I can carry over the knowledge here. If I were not able to do that elective I would not be as confident in the design process. I learned about the importance of material design, android architecture, fragments, Google APIs.

Figure 2 below shows the basic lifecycle for an android application. When developing any android application this needs to be included in the early stage as each stage alters the application in some way. For example, if the user moves off from the application you don’t want the app to instantiate itself instead of continuing where the user left off. I will be considering the lifecycle.



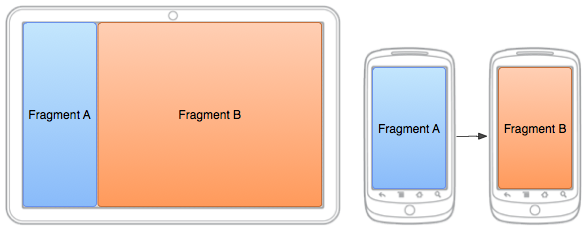
**Figure 2:** Android Lifecycle (Masters, 2013)

### Material design

Material design was introduced by Google in 2014 as a design language to be used in android development. It is a comprehensive guide to aid in the visual, interaction, and motion design across its many platforms. It is made up of design best practices, common conventions, and visuals to help build expressive apps that feel professional on any device (Google, 2021). Since Android 7, they have added more features to the existing development that help with material design. They added a new theme to style all the UI widgets, they added more complex widget views like lists and cards, and new APIs for custom shadows and animations.

The addition of the new themes allows the application to be forced into whatever colour scheme the developers want. The most common themes are light and dark, and they can change the texts, fonts, and buttons accordingly. It is best practice to add at least one UX patterns in your design as they add to the overall implementation. You can promote your UI’s main action by adding a Floating Action Button (FAB). Usually, they are located on the bottom right of the screen and appear to be hovering over the screen. This is pulled off but having shadows and shaders underneath promoting a sense of depth. Another common UX is implementing a Navigation Drawer which helps the user swap from screen to screen more effectively. Whenever possible it is advised to use predefined material icons on the navigation menus and buttons. These icons are small vector images that describe what the action is, i.e., the back button has a left arrow, email has a letter. RecyclerView and CardView allow the developers to add a card layout to their application that automatically changes depending on its input. You only need to create on CardView to any scale and when you call it inside the RecyclerView you can change the layout to suit the needs. I plan on utilizing most of these in my design as it will make the end application feel more professional as well as more pleasant to look at.

### Fragments

Fragments have been around since Android 3 and they offer a lot to developers. Before they were introduced developers had to start a new activity whenever they wanted to move onto another screen. With the need for more complex applications, this approach was causing issues. What developers can do now is combine one or more fragments to build a single activity or even reuse fragments across multiple activities. Developers can no update different portions of their UI on the screen that corresponds to what the user has navigated onto without updating the entire screen. With every growing difference in Android devices with all their types of screen sizes ranging from small smartphones to large tablets, fragments allow more flexibility. On smaller screened devices we can have one fragment loaded doing one action while on a tablet we can have two fragments side by side doing the same actions as before.

**Figure 3:** Android Fragments (mit, 2021)

## Diagrams

It is important to have diagrams of all types designed and ready before implementation as this gives a general representation of how the software system will look. This allows the developers and the clients to share ideas and iron out any issues that they might have before committing to the code. I have created and generate the below diagrams for this project, and I will be describing each diagram. They are as follows:

1. Use Case Diagram
2. Sequence Diagrams
   1. Dashboard
   2. Augmented Reality Moon
   3. Moon Phase
3. Class Diagram
4. Flowchart
5. Application Wireframes
   1. Splash Screen, Home Page, and Settings
   2. Augmented Reality Page
   3. Discover Page
   4. Moon Phase Page

### Use Case Diagram

A UML Use Case diagram can be described as the simplest way that we can represent the user’s interaction with a given system. It is a high-level diagram meaning that it does not go into the fine detail that the system is built upon. These diagrams are created right after the requirements have been successfully gathered. The users of a system are represented as ‘Actors’ while interactions are annotated as ‘novels’. The arrowed lines represent a connection between the actors and the use cases.

Figure 4 below is the use case diagram of the application. We can see two actors, User, and Firebase. The user is acting as the physical user who would be interacting with the system while the Firebase is acting as the cloud database. I have added the Firebase as an actor as it is responsible for sending out the models to the user in real-time. If there is an update to be made on the models, the database will be updated and not the application allowing for a lightweight system. The user can pick four routes that each use case is connected to. The use cases are included to one another in a way where they are connected to another via an ‘include’ statement.

I have created the allocated use case documentation as well below the diagram. Use Case documentation is great as it allows the developers and clients to talk through the entire process on interaction from start to finish. If anything is missing, then developers will go back onto their requirements and begin again.



**Figure 4:** Use Case Diagram

### Use Case Documentation

#### Use case 1.

**Use Case Name:** Launch the application dashboard.

**Participating Actor:**  User

**Entry Conditions:**

1. The user has an Android smartphone in hand.
2. The user has the application already installed on their phone.

**The flow of Events:**

1. The user locates the application icon on their smartphone and clicks to open it.
2. The application presents the user with a splash screen containing the logo, title, and slogan.
3. The application checks to see if the user has been connected to the internet.
   1. **If:** The application is not connected, the application sends a message to the user asking to connect to the nearest Wi-Fi router.
4. The system builds the navigation menus, application fragments, and connection to the Firebase database.
5. The user is greeted with the home page on the dashboard.

**Exit Conditions:**

1. The user has been successfully presented with the application's dashboard.

**Special Conditions:**

1. The android device is higher than the Android 9 Pie (API level 28).

#### Use case 2.

**Use Case Name:** Launch the augmented reality, Moon.

**Participating Actor:**  User

**Entry Conditions:**

1. The user has an Android smartphone in hand.
2. The user has the application already installed on their phone.
3. The android smartphone is connected to the internet.
4. Google Firebase database is stable and operational.

**The flow of Events:**

1. The user opens the application and successfully views the dashboard.
2. They click on the AR navbar button to open the augmented reality screen.
3. The system responds by displaying the AR fragment that houses 3 icon buttons, Earth, Moon, and Portal.
   1. **If** the user clicks on the Earth button, the system will contact Firebase for the ‘Earth’ model stored online.
   2. **If** the user clicks on the Moon button, the system will contact Firebase for the ‘Moon’ model stored online.
   3. **Else if** the user clicks on the Portal button, the system will contact Firebase for the ‘Portal’ model stored online.
4. Once the system confirms that the user has clicked the Moon button, it generates a local copy of the model from Firebase.
5. The system displays the fragmented screen to the user.
   1. **If** it is the user's first time opening this screen, the system will require the user to allow permissions to use the camera.
6. On confirmation, the system uses the rear camera to show an image in real-time with a button overlay on the bottom of the screen.
7. The system calls ARCore library to translate the environment using SLAM. This gives the system variable information about its surroundings.
8. The user clicks the button, they will be able to drop the model onto the surface.
9. A 3D model that represents the moon is displayed to the user for their interaction.

**Exit Conditions:**

1. The user has been successfully launched and interacted with the augmented reality Moon.

**Special Conditions:**

1. The android device is higher than the Android 9 Pie (API level 28).

#### Use case 3.

**Use Case Name:** Check the moon phases on a given date.

**Participating Actor:**  User

**Entry Conditions:**

1. The user has an Android smartphone in hand.
2. The user has the application already installed on their phone.

**The flow of Events:**

1. The user opens the application and successfully views the dashboard.
2. They click on the Moon Phases navbar button to open the Moon Phases screen.
3. The system responds by displaying the user the picture of the moon, some text asking the user to choose a date, and a button.
4. If the user clicks the button, the DateDiaglog window appears with today’s day asset to default.
5. When the user clicks on the 28th of January, the system response by updating the image with the correct Moon phases for that night.
6. The system will update the text on the screen to show the user how much of the moon is illuminated as well as the name of the Moon cycle.
7. The user can see that on the 28th of January that it will be a Full Moon, and the lamination is 100%

**Exit Conditions:**

1. The user can see that there is a full moon on the 28th of January.

### Sequence Diagram

UML Sequence Diagrams are another form of interaction diagram that software developers use while creating an application. These types of diagrams help aid in the development as they show the interaction between components relative to time (Visual Paradigm, 2021). Using the vertical axis on the diagram, one can see the order in which a message is sent back and forward between components. The solid black arrows facing the right represent a message call while the dotted arrows to the left represent its response. In a sequence diagram, every message call must have a response call. These diagrams usually are high-level diagrams as they do not go into the finer details of a system or its subsystem. When a component is not needed anymore in the diagram there will be an ‘X’ at the end of their lifeline to show that the lifeline will be terminated. If the X is not present, then the component stays active within the system.

For the uses cases, I have developed in the above section I have created their representation in their sequence diagram. I had to pull data from what I learned over my research stage to include into these diagrams. Such things as Android lifecycle as well as their requirements. I have learned a lot about the android lifecycle from the mobile development elective that I have completed last year.

#### Dashboard

**Figure 5:** Sequence Diagram Dashboard

In the above diagram, Figure 5, we see the sequence diagram for viewing the dashboard. For the user to be able to see the dashboard there is a series of calls that need to be done by the system. The system does these automatically without the user’s interaction like calling the AppCompatActivity, Builder, Navigation, and navigation. Once they are called and retrieved the system sends back the data to the user.

The diagram below, figure 6, is the sequence diagram for viewing the augmented reality moon. This diagram is more complex than the others due to there needing to be more components called and stored. From the actor's first message call there are all of the calls that the system needs to first call. From my research paper I learned that ARCore, the chosen Augmented reality API will only work on a set of few android devices. The system must check if the user’s phone is compatible as if not the system will crash and cause runnable problems. The system does a method call on itself called ‘checkIfSupportedDeviceOrFinish’ and this will pull the Android version into a double variable. If that does not match the compatible version, it will display so in a Toast message. A Toast is a pop-up dialog box that appears on the bottom of a screen that can display any data that is being passed into it. When the system is happy to continue it will move on and connect to the Firebase inside the fragment. A Fragment is a type of viewable screen inside the android development system. It allows a viewable container to be scaled to size or have a defined look on each screen size. The system needs to initialize a database connection and needs to pass the child ID model so that it can allocate a file in memory. The database sends the model data to the phone and once all the data is compiled it will send it over to the Throwable component. A Throwable component is a library inside ARCore that allows the movement, scale, and rotation of a static model within an application. The conversion needs to be done to this model so that the user may interact with it fully. Once the conversion is complete it will be sent back into the fragment to be displayed on the screen.

#### Augmented Reality Moon Model

Jonathan Roddy 27 Jan. 21 18

#### 

**Figure 6:** AR Moon Model

#### Moon Phase

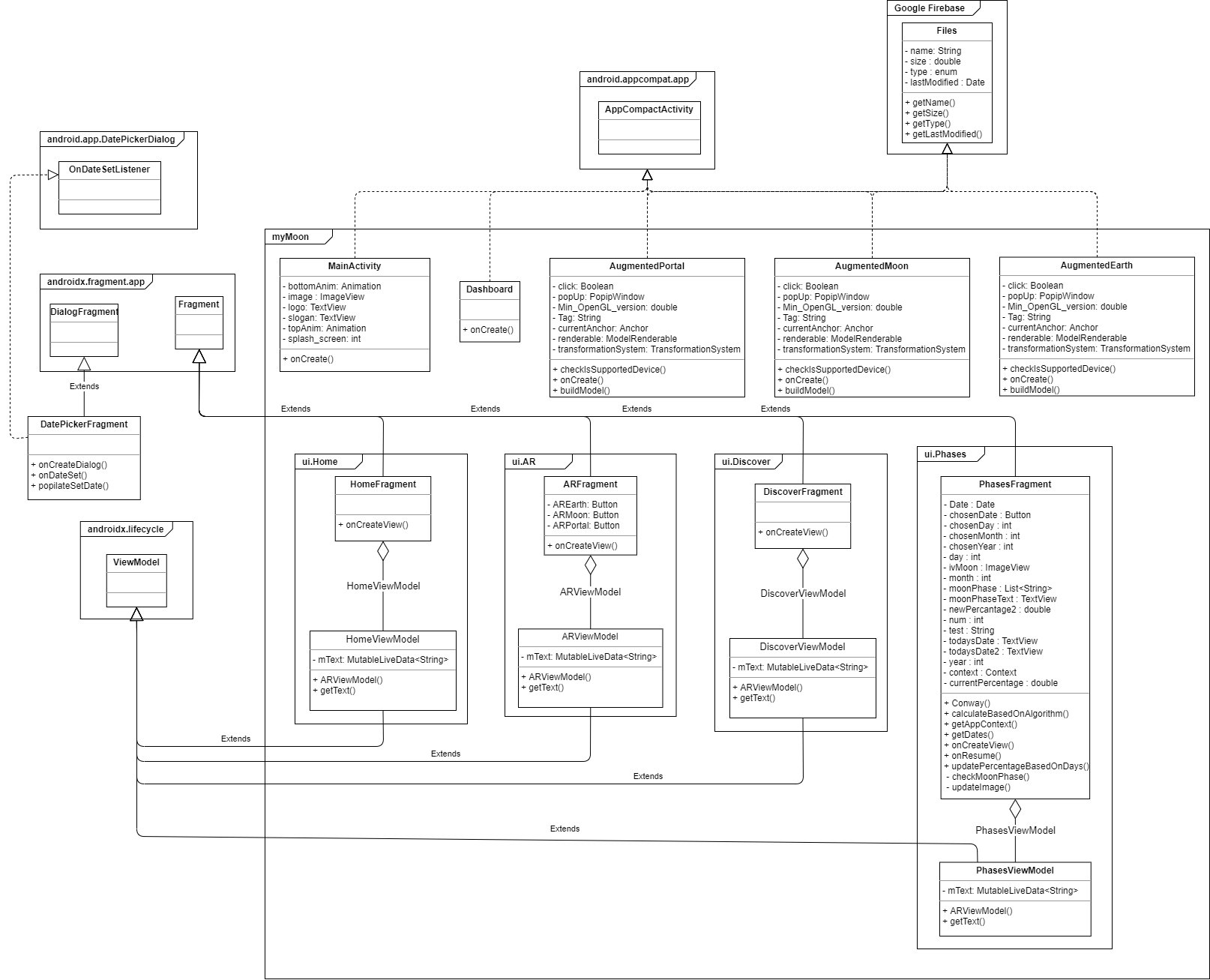
**Figure 7:** Sequence Diagram Moon Phase

In figure 7, we see the Moon phase sequence diagram. From my experience in mobile development to correctly pass data inside a navigation view the system must be separated into a view and model container. This allows the data to communicate loosely within the entire system. Once the user calls on the PhaseFragment the system begins by calling the ViewModelProvider. This action will allow the system to call the views to the user and scaling to the screen size. This use case calls for text views and button interaction so the system must find them before moving on. It finds the reference to each of them and stores a local connection with them. The actor will then click the button which will initiate the ‘OnClick’ method call. This message will pass the chosen date that the actor has selected into the DateDiaglog fragment. This DateDiaglog is a build-in asset inside the android development and its sole purpose is to display a calendar to the user. This Date is then sent back to the system for it to calculate the algorithm accordingly.

### Class Diagram

Class diagrams are very important for project development as it goes further into the details of the system. A good class diagram allows developers to do less code as its layouts the different classes that make up a system. Class diagrams are the only diagram that can represent object-orientated languages. They describe the attributes and operations that a class will undertake as well as their relationship with one another. Developers will take a lot of time to design a class diagram as the more information that is represented here results in better, cleaner code. The purpose of class diagrams in the design phases is to analyze the view of the application, describe the responsibilities of a system, the base for component and deployment diagrams, and finally to forward /reverse engineering. Each class is presented as a square with its name, attributes (variables), and operators (methods). Each attribute can be denoted as public, private, static, read-only, and protected while the operators can have the method call. The lines that connect one class to another is called an association and acts as a relationship between those linked classes. There is a wide array of associations and annotations that developers can add to their class diagrams, like a generalization, dependency, usage, aggregation, composite aggregation, and many more. There should always be a relation within classes in an object-oriented system (tutorialspoint, 2021).

Below in figure 8, I have created a package class diagram. This diagram has the classes that I will need for the system and it is presented inside each of their packages. The overall application is called ‘MyMoon’ and has 4 classes inside of it and 4 other packages. The packages and classes outside of this system are all needed here as they either represent parent classes or other classes are dependent on them. That means that those classes are needed for their functions and pass them down to their children. For example, we can see that MainActivity, Dashboard, AugmentedPortal, AugmentedMoon, AugmentedEarth are all dependant on android. appcompact. app class and GoogleFirebase so they are denoted with a dotted arrow facing up. The android. frament.app class as this is acting as a parent class to its children classes, HomeFragment, ARFragment, DiscoverFragment, and PhasesFragment as the children classes inherited methods from its parent. That is denoted with a solid black line facing up. We can see the inside each of the UI packages inside MyMoon, the ViewModel classes are connected to the fragment class via aggregation which means ‘has a’. So, we can say that HomeViewModel has a HomeFragment.

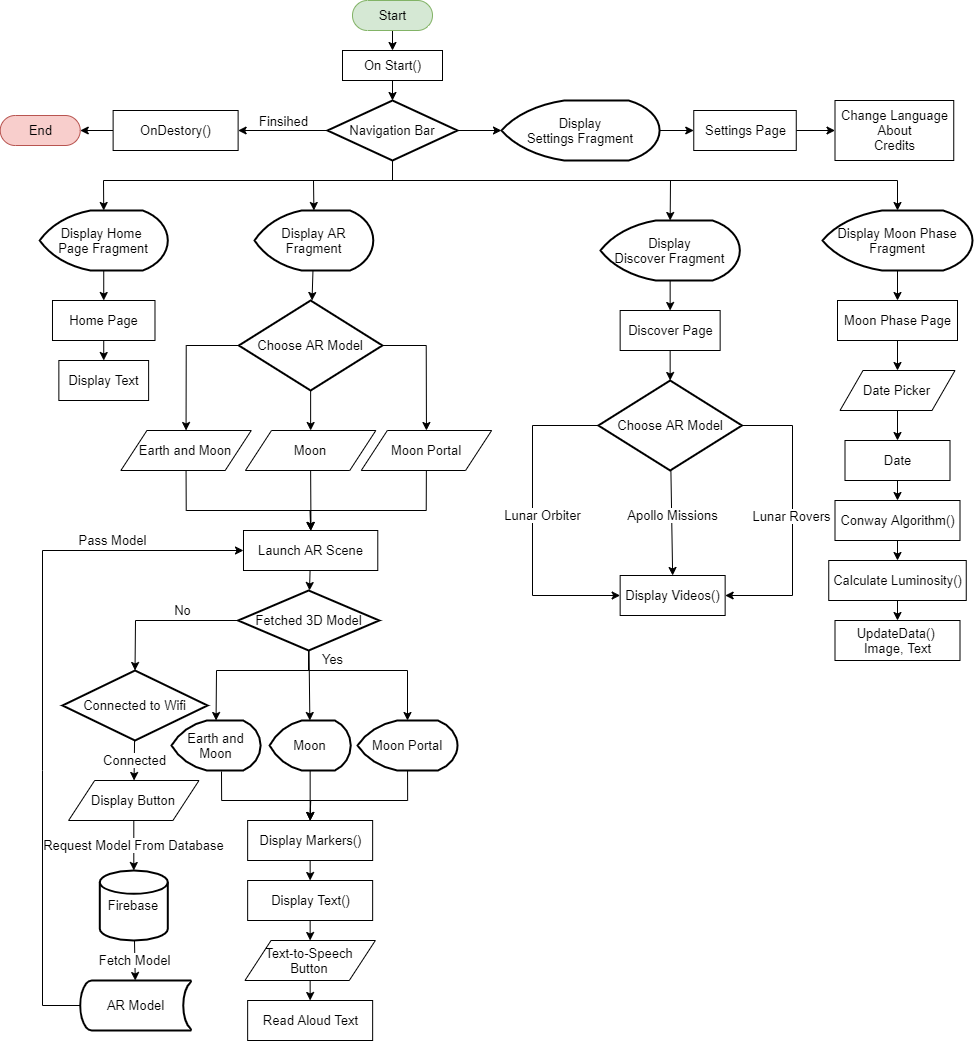


**Figure 8:** Class Diagram

### Flowchart

Flowcharts are a visual representation of the steps and flow of a given application. Each step in the process has its unique diagram shape ranging from circles, ovals, rectangles, diamonds, and more. It gives the presenters a flow of decisions that can be made within the application but having connected lines with directional arrows. Flowcharts were originally used by industrial engineers to structure work processes in their assemble manufacturing but today they are used in all areas of business.

I have used a flowchart to demonstrate the flow of events within this application down below in figure 9. We can see the most common actions that will be done in this diagram and the decisions a user can do while navigating. There should be no code in this diagram as this is a high-level diagram. Its detail is in the navigation of the application compared to the internal methods. The flowchart begins on the green icon ‘Start’ and ends on the red ‘End’ icon. Any rectangle represents a process that the application does on its own like ‘onStart()’ or ‘SettingsPage’. Any diamond shape represents a decision the user must do to continue like ‘Navigation Bar’ and ‘Choose AR Model’. The rectangles with one side as a point and the other as a curve represents a display process like ‘Display Settings Fragment’ and ‘Display Home Page’. Any rhombus shape represents an input of any type like ‘Moon’ and ‘Display Button’. The cylinder shape is a database so here it represents a database. Finally, the rectangle with a convex and concave curve on the side represents the data storage process. In my application, this is the process when the AR Model is being saved from the database.

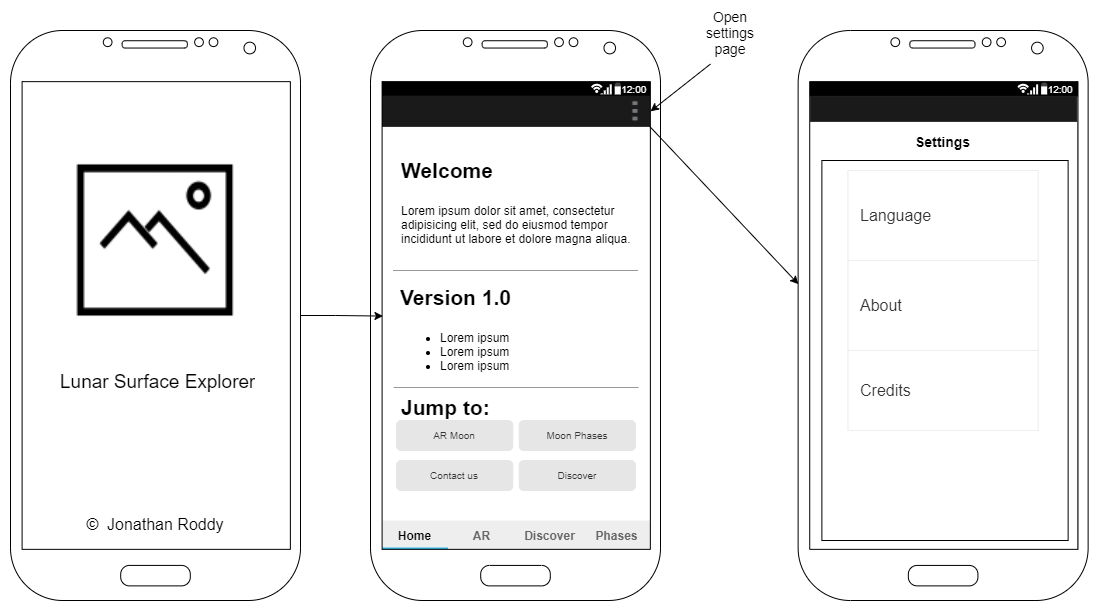


**Figure 9:** Application Flowchart

### Application Wireframe

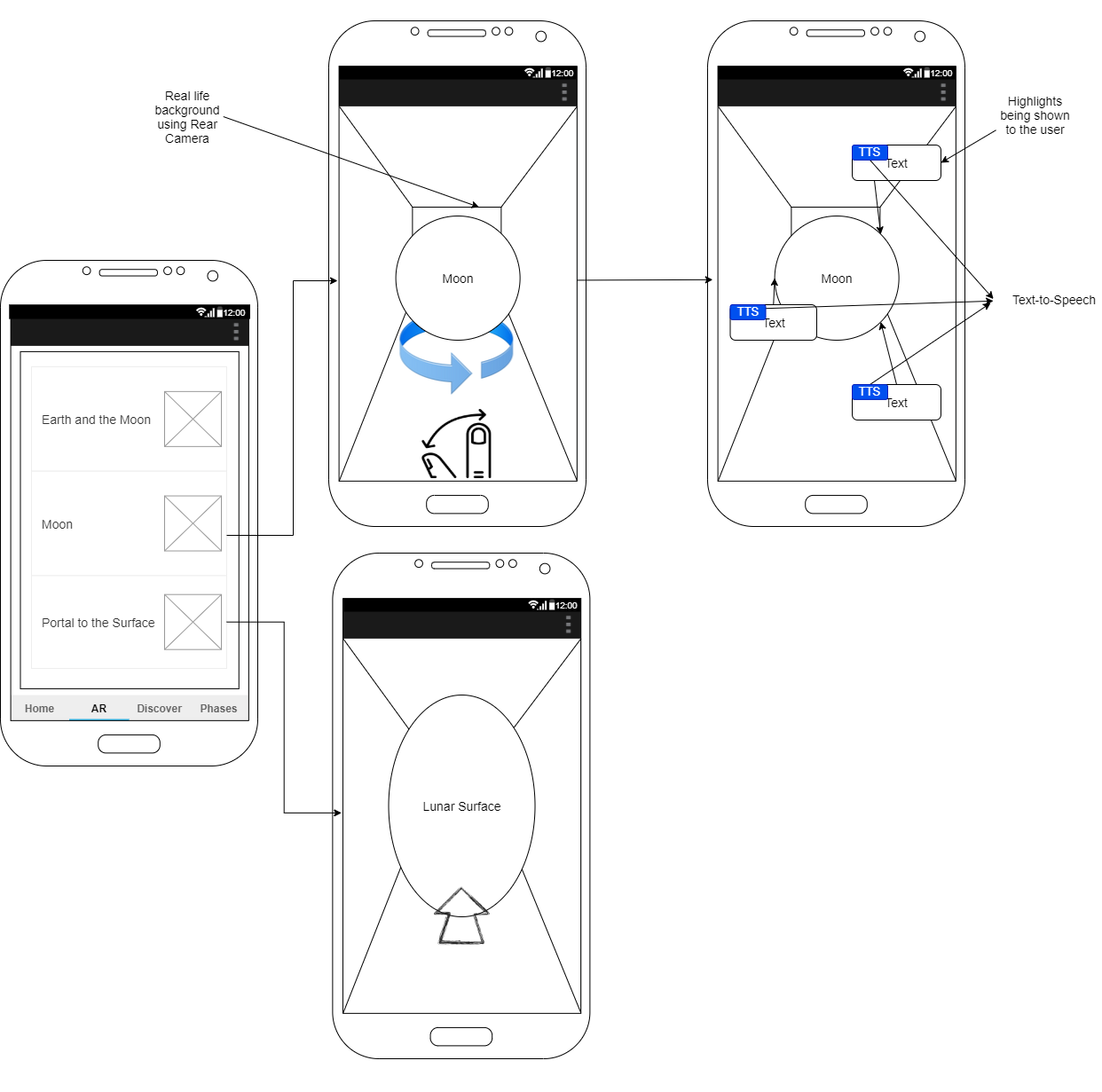
Wireframes are used as a visual aid when developing any web or mobile software application. They are used to show the User Experience (UX) of an application. It is a 2D representation outlining the different screens of a system with icons, text buttons, and more. It is known to be the earliest form of a prototype as with the wireframes you can follow along using arrows to point to the connection between each screen. Typically, wireframes are monochromatic, meaning black and white, but they can be developed with colors for a better visual representation. Using wireframes in the design stage helps the client to know the features of the proposed system and allows for their input before development can begin. This is great as it irons out any concerns that the client might have before spending time on concepts that the user might not have agreed on.

I have created a complete application wireframe outlining the different screens that my application will have. By having wireframes drawn out it helped me to visualize the application as I had complete control over how the application will look like. I had to use my knowledge of material design and UX to be able to generate the best outcome. I have connected different screens with arrows to show you the flow of events starting with figure 10 below.

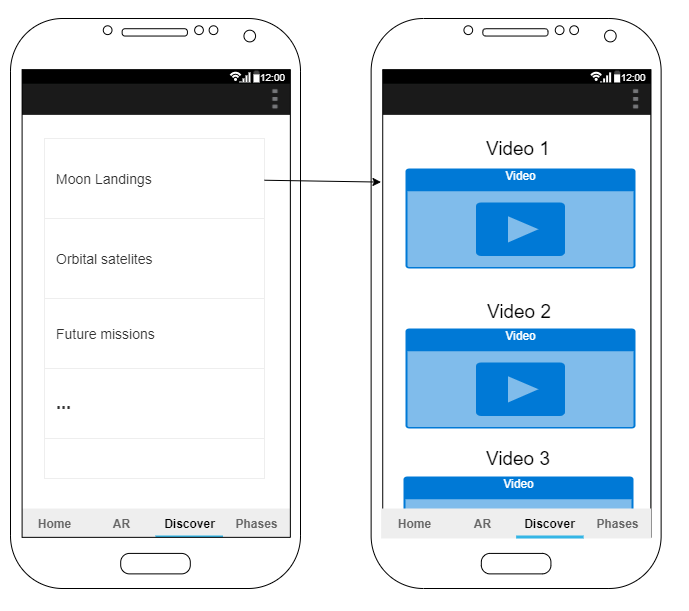


**Figure 10:** Splash Screen, Home page and Settings

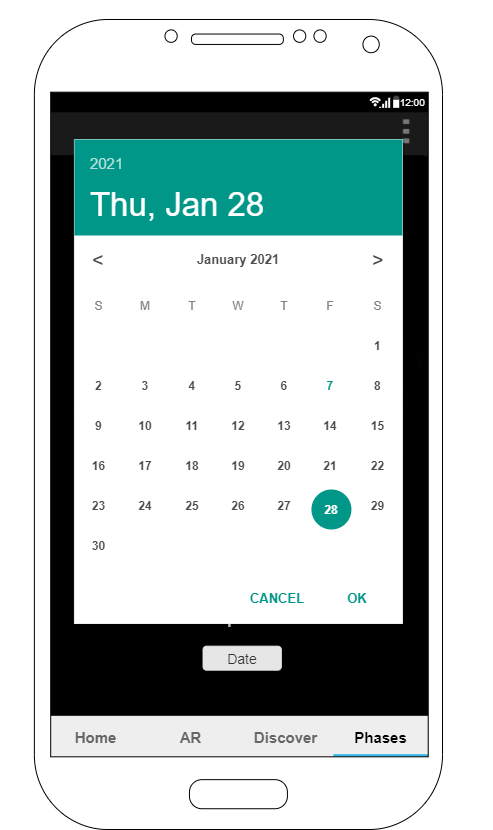
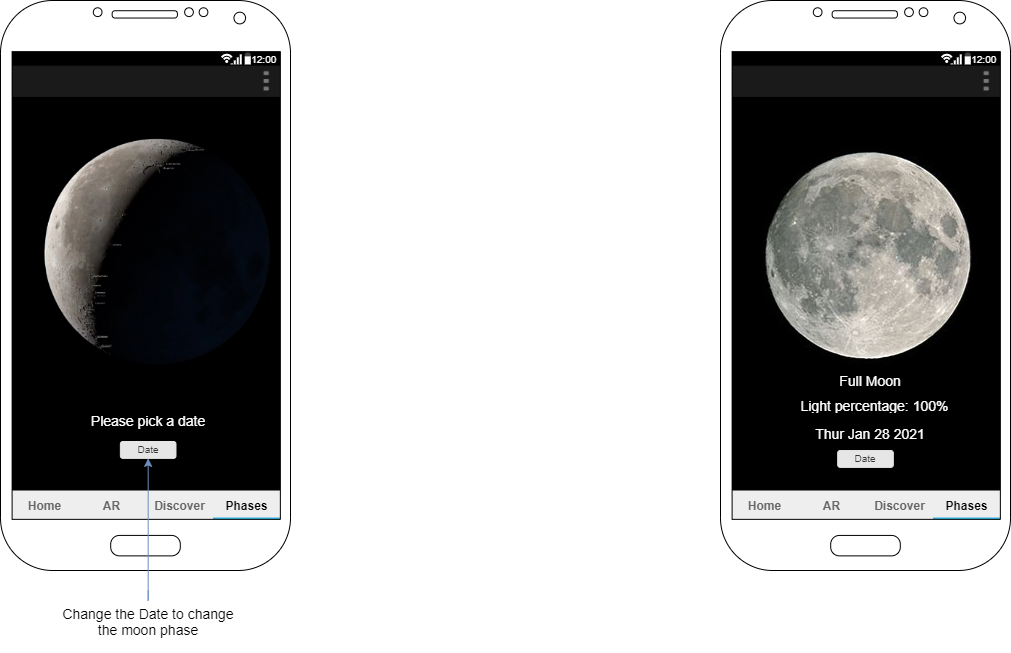
Figure 10 shows us what the splash screen, home page, and settings page will look like. A splash screen is a name given to the first screen that the user is presented with when launching an application. I will have an image of the Moon followed with some text underneath to greet the user when loaded. After the splash screen has concluded the user will be faced with the homepage screen. The home page will have a welcome section, version number, and any new updates that have been added as well as some links to some special parts of the application. On the top right of the application, you can see 3 dots, a menu icon that will be on each screen going forward. This menu icon will launch the settings page, giving the user the option to change language, about page, and the credit section. I have added a bottom navigation bar to help the user navigate between the different pages on the application. I have added this to adhere to the best practices of material design. If any of the buttons on the navigation bar is clicked the screen will be updated with the appropriate screen.

**Figure 11:** Augmented Reality Page

Above in figure 11, we can see the AR screen with its corresponding actions. On the AR screen, we can see 3 different buttons with an image of each of the actions. For either the ‘Earth’, ‘Moon’ and ’Portal’ the button will launch the same screen but will call the different model. This screen will be built using ARCore, scene form, and Android fragments. Building it as one page that will be updated reduces the code. From the above image, you can see a circle that represents the moon. The fingers underneath represent the user’s ability to move, twist and scale the model itself. This ability gives the user full control of them where they want the Moon to be placed. The Moon will have markers, pins, and text over it highlighting important aspects that the user has chosen to interact with. There will be a Text-to-Speech (TTS) option on the moon so the application will read out any of the text aloud. The Earth Model will also have the same option as the Moon screen. The Portal screen will allow the user to walk into the 3D model and it will transport them onto a virtual environment of the Moon’s surface.

**Figure 12:** Discover Page

In Figure 12 we can see the Discover page being represented to the user. This section and the next will be built using RecyclerView and CardView to again, adhere to the best practices when it comes to android design. I will have a list of options that will be shown to the user giving them choices like Moon landings, Orbital satellites, future missions, and more. Whichever one the user clicks a new page will be called and will populate information to the user in the form of videos or some text that can be read aloud to the user using TTS



**Figure 13:** Moon Phase Page

The last wireframe I have is the Moon phases screen. This screen will allow the user to pick any date that they want, and the application will show its corresponding moon phases. On initial load, the application asks the user to pick a date with a button underneath. A calendar will appear to the user to which they can input any date they so wise. After the application has retrieved the date it will update the image, and texts to the correct information. It will display the name of the Moon phases, the light percentage that it is currently, and the date again. The user can choose another date by clicking the date button.

## Conclusion

With these wireframes complete, I have finished the designing stage for the project. I have spent some valuable time coming up with concepts based on the research I have committed during the project’s lifecycle. The next stage is implementation, where I will start developing this project further. I will use all the diagrams I have created to help with the coding and aid me in getting this project complete on time.

# Software Chapter: Implementation

## Introduction

In this chapter, I will be talking about my implementation of this project of mine. At this stage, I have conducted a literature view, research paper, and a design chapter. I will everything I have to learn while undertaking those, with the knowledge I already must programmatically complete this section. I plan on testing each stage of the application and once I have my tests complete, I will be happy to move on to the next stage. I have developed a few android applications at this stage of the project thanks to my mobile development elective and my personal development. I feel a lot more comfortable with the software I have chosen and feel I have a good grasp of what needs to be done.

This project is a limited resource project meaning that I do not have an unlimited number of tools, assets, or time that I can use. I will need to plan out the remaining project timeline and keep track of my process. As of writing this project I am still currently in my final semester in my final year so I do not have all day and night to commit to this project so juggling my time will the hardest to juggle. Each day is vital from this day onwards to be able to complete this application on time. I need to divide some time each day to this project on either development or research into further development. I will talk further about the project management that I will undertake in section 3.2 below.

That said there is still a lot of steps that I do not know how to complete so I will need to get whatever resources that may be to better my understanding. I will need to improve my skill set so that those steps I am unfamiliar with can be overcome and tackled to the best of my ability. I will investigate these issues in the learning obstacles in section 3.3 below.

I will then talk about the development of the application ranging from the database, 3D model creation, and Android development.

## Project Management

In this section, I will be keeping a track of what tasks has been complete and what has not. I will form an action plan to which it will act as a structure I can follow. I will be talking about the different tools outside of the development cycle that I will be using to help me track the project. This section will allow me to be transparent on the lifecycle while also being accountable for each stage. This is a great opportunity to better my project management as it will be key for my future career.

### Management tools

I am going to be keeping a weekly diary of what I have done for this project so that I will not forget anything. From my time in college, I have learned many tools that I can utilize for this project to help me manage my project. I will be using OneNote to take any notes throughout the lifecycle as well as using GitHub to store my code repository. OneNote is a free Microsoft application that can be run on any device for free and can be saved online so that I can read and write from any device at any time. GitHub is a web application that developers use to store their code under their account. Using GitHub allows me to save my code online so that in any circumstances my project will be saved online for safety. I will be keeping a copy locally as well for further insurance.

### OneNote

*Updates to be added*

### GitHub

*Updates to be added*

### Tasklist

During my work placement in my third year, I was able to work for Kerry Group where I was apart of a team. I was able to contribute to many projects and learned a lot about project management and how to best track each task within a growing team. We initially used Planner to track our everyday tasks which allowed us to see what stage a given website was on. Each member was able to send comments or move them down the line if appropriate to do so. We soon moved over to the DevOps board which replaced Planner altogether as DevOps was integrated into Teams a lot better. Most of the websites were being managed on Azure DevOps so it was cutting out a few clicks. I was in charge of migrating this process over so I am quite comfortable using either Planner or Azure DevOps Boards. I am the only person working on this project, so I do not need to track other people's progress besides my own. For this reason, I have chosen not to utilize these great planners and will use an Excel spreadsheet to track my tasks. I am also comfortable in using Excel as I was trained in the advanced mechanics during my time there. I will be updating this Excel sheet each time a new task has been completed and if any new tasks arise. Please see table 1 below for the Excel task list.

|  |  |  |
| --- | --- | --- |
| **Task Name** | **Priority** | **Status** |
| Create Moon Model | High | Complete |
| Create Earth Model | High | Complete |
| Create Portal Model | High | Incomplete |
| Create Firebase Database | High | Complete |
| Create Android Studio Project | High | Complete |
| Create GitHub Repo | High | Complete |
| Connect Github Repo to Project | High | Complete |
| Create Splash Screen | High | Complete |
| Check phone for version compatibility | High | Complete |
| Check Screen Size | Medium | Incomplete |
| Create Logfile | Medium | Incomplete |
| Create Home page Screen | High | Complete |
| Create Welcome Section on Home Page | Low | Incomplete |
| Create Version Section on Home Page | Low | Incomplete |
| Create Jump-To Section on Home Page | Low | Incomplete |
| Create Navigation Menu | High | Complete |
| Create Settings Menu | High | Incomplete |
| Create Settings page Screen | High | Incomplete |
| Create Language section on Settings Page | Low | Incomplete |
| Create About section on Settings Page | Low | Incomplete |
| Create Credits section on Settings Page | Low | Incomplete |
| Create Augmented Reality page Screen | High | Complete |
| Ask for camera Permission | Medium | Complete |
| Create Earth Button Navigation on AR Page | High | Incomplete |
| Create Moon Button Navigation on AR Page | High | Incomplete |
| Create Portal Button Navigation on AR Page | High | Incomplete |
| Implement ARCorre API Library | High | Complete |
| Create ARCore fragment | High | Complete |
| Pull AR Models from Firebase DB | High | Complete |
| Display AR Model on the surface | High | Complete |
| Allow user to manipulate Model | High | Complete |
| Display Markers on AR Model | High | Incomplete |
| Display Text on AR Model | High | Incomplete |
| Display TTS on the Text | High | Incomplete |
| Create Discover page Screen | High | Complete |
| Create Recycler View on Discover Page | High | Incomplete |
| Create Card View on Discover Page | High | Incomplete |
| Pull discover data from Firebase DB | High | Incomplete |
| Populate Discover Page with different sections | High | Incomplete |
| Create Second Discover page | High | Incomplete |
| Create embed Video Streams | High | Incomplete |
| Create Text views | High | Incomplete |
| Implement TTS Library on the Text | High | Incomplete |
| Create MoonPhase page Screen | High | Complete |
| Enable user to manipulate a calendar for dates | High | Complete |
| Calculate Moon Phases by date | High | Complete |
| Update Moon Phases by date | High | Complete |

**Table 1:** Task Management List

## Learning obstacles

For this project to be complete, I needed to get educated on some skills that I did not have before. At the early stage of the project, I never made an android application, so I had a lot to do to get ready for its implementation. As mentioned, before I was lucky enough to do a semester of Mobile Development at Limerick Institute of Technology. This elective allowed me to learn so many things about the Android ecosystem as well as getting hands-on with the development process. I learned about material design, Android application lifecycle, connecting to a database online, the best practices to adhere to when developing a complex system. Without having this training, I would not have known as much as I do now which would have delayed the project further. I have used the knowledge I gained in the above chapter when it came to the design of the application, but I will be continuing to use the skills in the implementation. Since completing the elective I have created a few applications like a 2 player Tic-Tac-Toe, live weather application, a fully working recipe book application, and many more. We were using Android Studio throughout this learning, so I was able to get some time to understand and use this IDE. From my prior research, I knew that Android Studio was not the only IDE I could use for this project, so I went out and developed a few applications using Unity. I was able to get some time to use this IDE and see the options that it can give me under a free license. It was a good IDE for sure, but the limitations were on the licensing and publishing of the final application. I wanted to make a demo of Augmented Reality, so I started to learn how to use both IDEs for that purpose. There were many more steps in Unity to perform compared to Android Studio, but both were easy enough to use. I decided to use Android studio instead as it gave me more options while also being completely free.

It was not just the IDEs that I needed to learn and understand as this project calls for Augmented Reality 3D models. From doing research I could see that many people were offering their own skill sets to create a customer rendered model for a set price, but I knew this was not feasible. From my research paper, I concluded that Blender is the best option for me to be able to create, render and export any 3D models that I so needed. However, I have never used such complex modeling software before so I needed to find some resources online that I could use to educate myself. I was reading the documentation online as the Blender community is so massive that there is a lot of forum and websites to aid anyone in their journey.

When having one of my supervised meetings with my project supervisor, they advised me to use LinkedIn Learning. LinkedIn is a social media website used by professionals to interact with other organizations and professionals. They offer online courses for anyone in any area of expertise. These courses can range from only a few minutes to a good few hours long. I found some courses on there that I knew would be beneficial for me and this project. Below under the certification section, I have added the three courses I undertook for this project.

### Certifications

****

**Figure 14:** LinkedIn Learning Cert

**Figure 15:** Android Development UI with Java

**Figure 16:** UX for Non-Designers

## Development Process

### AR Model Creation

#### Blender

### Database implementation

### Android Development

### Text-to-Speech Implementation

## System Structure

## Completed System

## Conclusion

# Bibliography

Google. (2021, 01 27). *2021*. Retrieved from developer.android: https://developer.android.com/guide/topics/ui/look-and-feel

Kwon, J. (2019, 05 08). *What is Software Development Lifecycle (SDLC)?* Retrieved from blog.usejournal.com: https://blog.usejournal.com/what-is-software-development-lifecycle-sdlc-523fd09340a6

Masters, J. (2013, 05 10). *Writing a simple Android app: 2. Creating the main screen*. Retrieved from Geekily Interesting: https://geekilyinteresting.wordpress.com/2013/05/10/writing-a-simple-android-app-2-creating-the-main-screen/

mit. (2021, 01 27). *Building a Flexible UI*. Retrieved from stuff.mit.edu: https://stuff.mit.edu/afs/sipb/project/android/docs/training/basics/fragments/fragment-ui.html

tutorialspoint. (2021, 01 27). *UML - Class Diagram*. Retrieved from tutorialspoint: https://www.tutorialspoint.com/uml/uml\_class\_diagram.htm

Visual Paradigm. (2021, 01 27). *What is Sequence Diagram?* Retrieved from Visual Paradigm: https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-sequence-diagram/