CA400 Airport Assistant

Technical Manual

Name: Shauna Moran

Student Number: 15381166

Supervisor: Ray Walshe

Abstract:

Airport Assistant is an Android Application to help users, primarily those who suffer from anxiety, with dealing with passing through an airport. According to travelweekly.co.uk a study conducted by CPP found that one third of people find a working week to be less stressful that taking a flight and a quarter say it would be less stressful to move house than to take a flight. The purpose of this application is to take the stress out of air travel and make it more accessible to people who avoid travelling due to fear of the panic it will cause. Airport Assistant works by, instead of overloading users with information about the airport, bringing them through the process step by step, completing checks, giving advice and providing important information. It is almost like someone holding the user's hand and being their assistant throughout the journey.

The primary feature of the application is the AR (Augmented Reality) functionality for measuring luggage sizes. This will help ease the worries of users who are concerned that their hand luggage may not meet the requirements of there airline before the leave home. Similarly, features such as security tips and walk times to gates will help make the entire process less stressful.

0. Table of Contents

1. Introduction

1.1. Overview

Airport Assistant is an Android Application to help users, primarily those who suffer from anxiety, with dealing with passing through an airport. This technical manual will outline a number of aspects of this project including research completed as part of the project, information about the development environment, a number of diagrams including system architecture, component model, data flow diagrams, state machines and sequence diagrams, sample code, problems faced and their resolutions, future work and finally installation and configuration of Airport Assistant.

1.2. Glossary

- **AR**
  - Augmented reality (AR) is an interactive experience of a real-world environment where the objects that reside in the real-world are "augmented" by computer-generated perceptual information.
- **Accessible**
  - Easy to approach, reach, enter, speak with, or use.
- **Integration**
  - The act of combining or adding parts to make a unified whole.
- **Duty Free**
  - Duty-free shops (or stores) are retail outlets that are exempt from the payment of certain local or national taxes and duties, on the requirement that the goods sold will be sold to travelers who will take them out of the country.
- **Check-in**
  - The check-in process at airports enables passengers to check in luggage onto a plane.
- **Airport Security**
  - Airport security refers to the techniques and methods used in an attempt to protect passengers, staff, aircraft, and airport property from accidental/malicious harm, crime, and other threats.

2. Research

2.1. Research Carried Out

At the beginning of my project I completed a vast amount of research anxiety associated with the airport. This allowed me to develop a further understanding of the topic at hand and establish whether there was a need for such an application. I decided to complete a multi vocal search to cover a wide range of possible sources. I completed these searches on Google Scholar and standard Google Search.

I looked into airport and travel anxiety. I found a number of articles relating to the topic but was primarily interested in the following

- An analysis of the airport experience from an air traveler perspective *(Wattanacharoensil et al., 2017)*
- Anxiety and health problems related to air travel *(B McIntosh et al., 1999)*
- Traveler anxiety and enjoyment: The effect of airport environment on traveler's emotions *(Bogicevic et al., 2016)*
- The anxiety condition that turns travelling into a nightmare *(Home, 2019)*
- Airport Anxiety: 12 Life-Changing Tips On How to Beat It! *(That Anxious Traveller, 2019)*

Each of these sources all appeared to have one common theme, people experience anxiousness in the airport as they feel a lack of control. I hope my application will help overcome this by keeping users informed and allowing them to feel in control throughout the process.

I was also surprised to see that blogs such as thatanxioustraveller.com recommend utilising mobile applications to help with airport anxiety. All of this information solidified my idea that such an application was necessary.

- **Resources**
    - B McIntosh, I & Swanson, Vivien & Power, Kevin & Raeside, F & Dempster, C. (1999). Anxiety and health problems related to air travel. Journal of travel medicine. 5. 198-204.
    - Bogicevic, V., Yang, W., Cobanoglu, C., Bilgihan, A. and Bujisic, M. (2016). Traveler anxiety and enjoyment: The effect of airport environment on traveler's emotions. Journal of Air Transport Management, 57, pp.122-129.
    - Home, W. (2019). 'I'm never going on holiday again': The anxiety condition that turns travelling into a nightmare. [online] woman&home. Available at: https://www.womanandhome.com/travel/airport-anxiety-fear-of-flying-62184/ [Accessed 14 Apr. 2019].

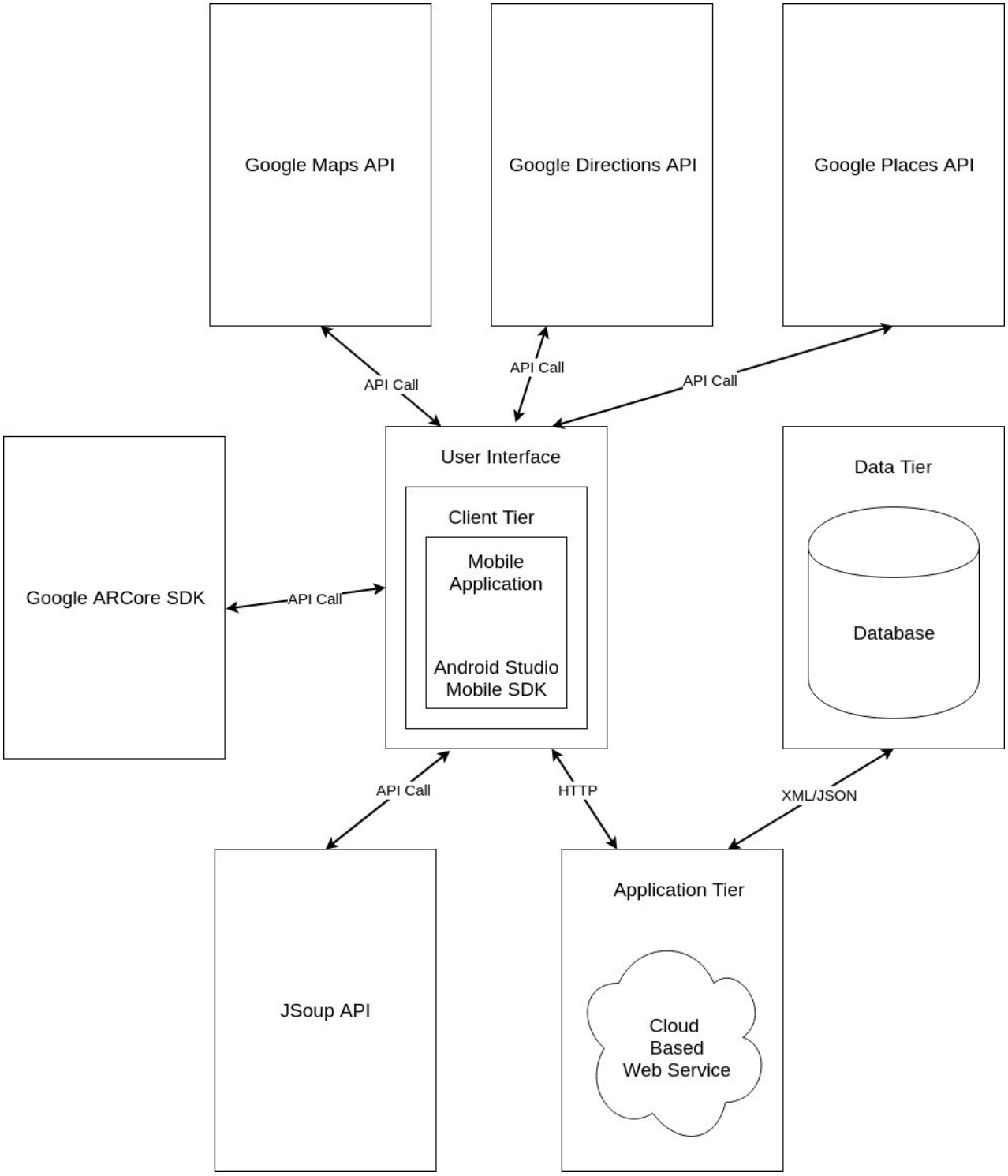
- That Anxious Traveller. (2019). Airport Anxiety: 12 Life-Changing Tips On How to Beat It! - That Anxious Traveller. [online] Available at: https://thatanxioustraveller.com/airport-anxiety-tips-how-beat-it/ [Accessed 14 Apr. 2019].
- Wattanacharoensil, W., Schuckert, M., Graham, A. and Dean, A. (2017). An analysis of the airport experience from an air traveler perspective. Journal of Hospitality and Tourism Management, 32, pp.124-135.

3. System Architecture

3.1. Development Environment

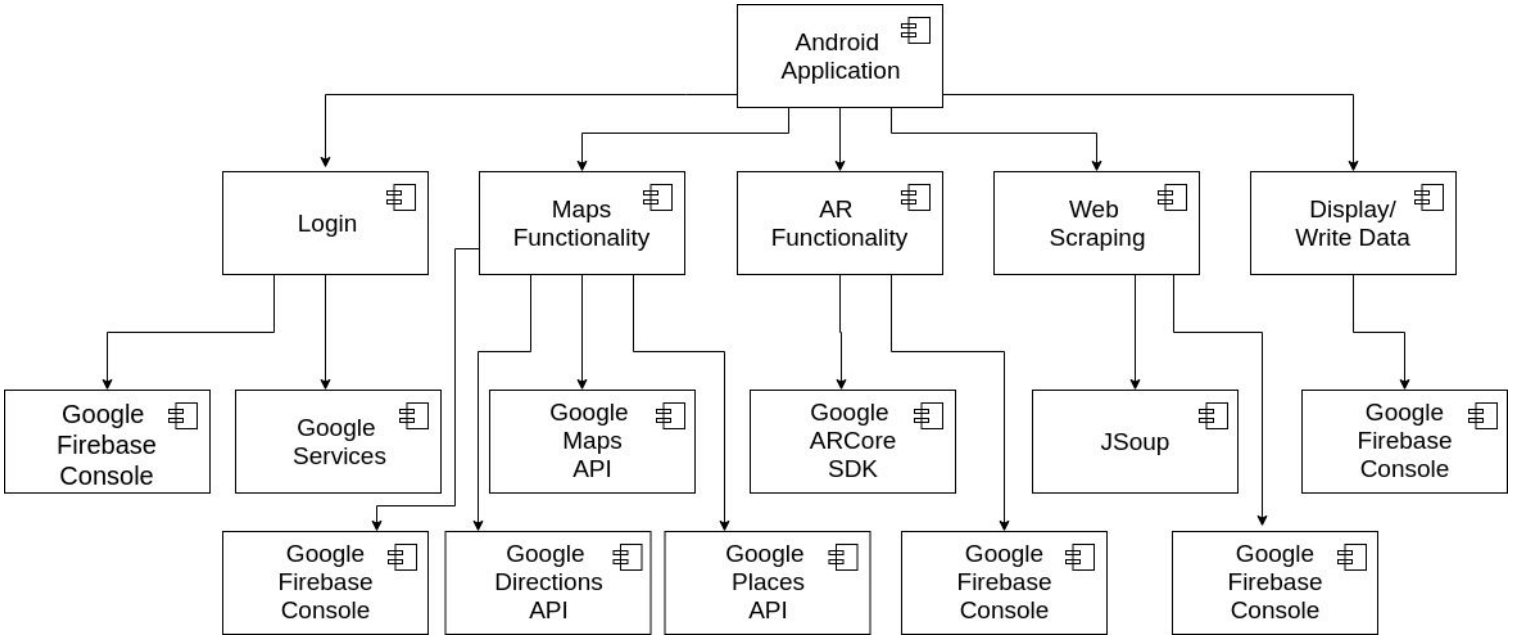
Airport Assistant was developed using Intelliji Ultimate version 2018.3.4. The application targets API level 27. This is due to the Google Sceneform requiring this API level. Devices must operate on this API to be allowed to utilise the emerging AR technology which requires the most up to date Android version to achieve it's groundbreaking functionality. It would not be feasible to develop this project on a lower API level if a user wished to utilise this functionality. This application consists of 31 classes and 29 XML files.

3.2. System Architecture Diagram

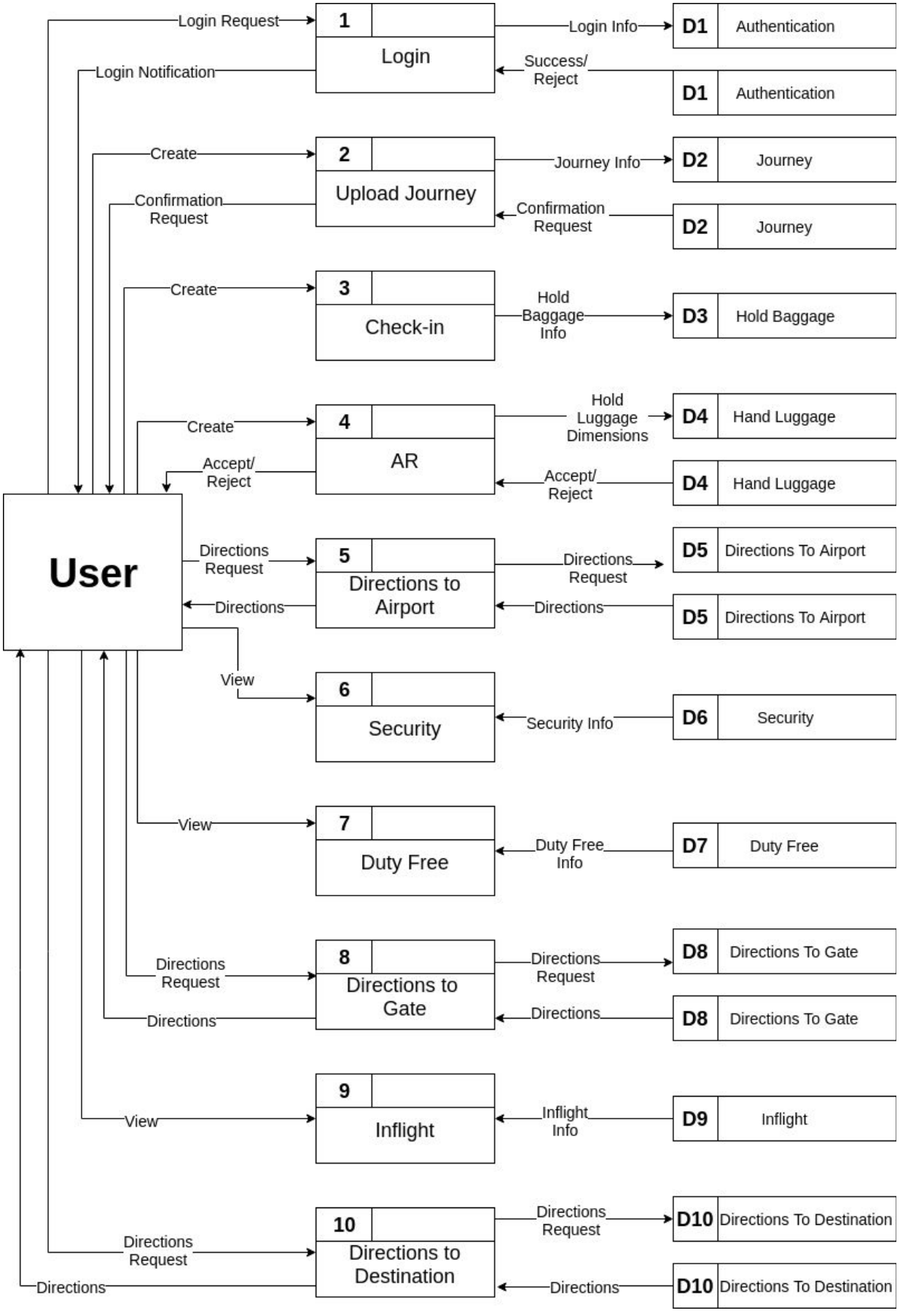


4. High Level Design

4.1. Component Model

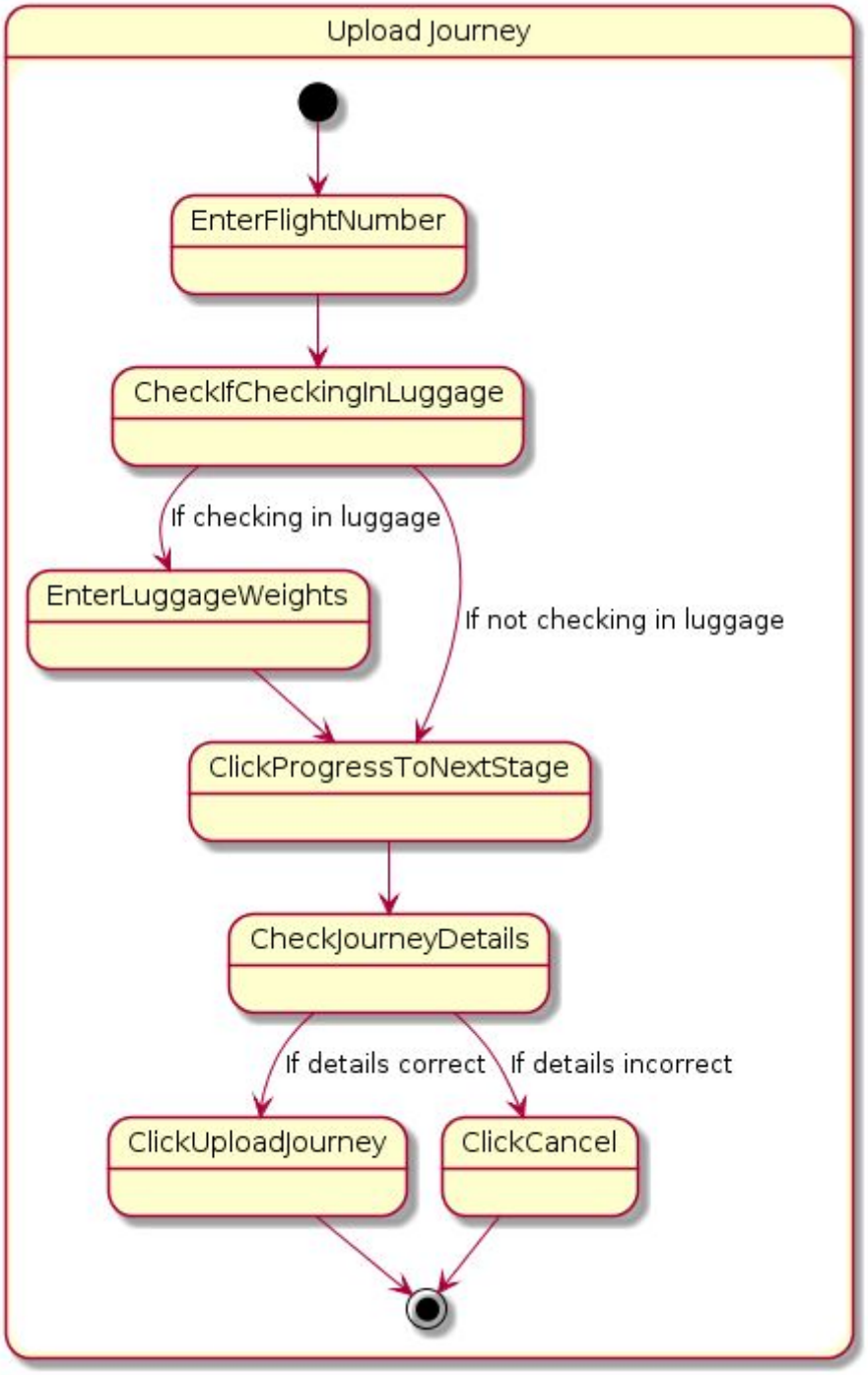


4.2. Data Flow Diagram

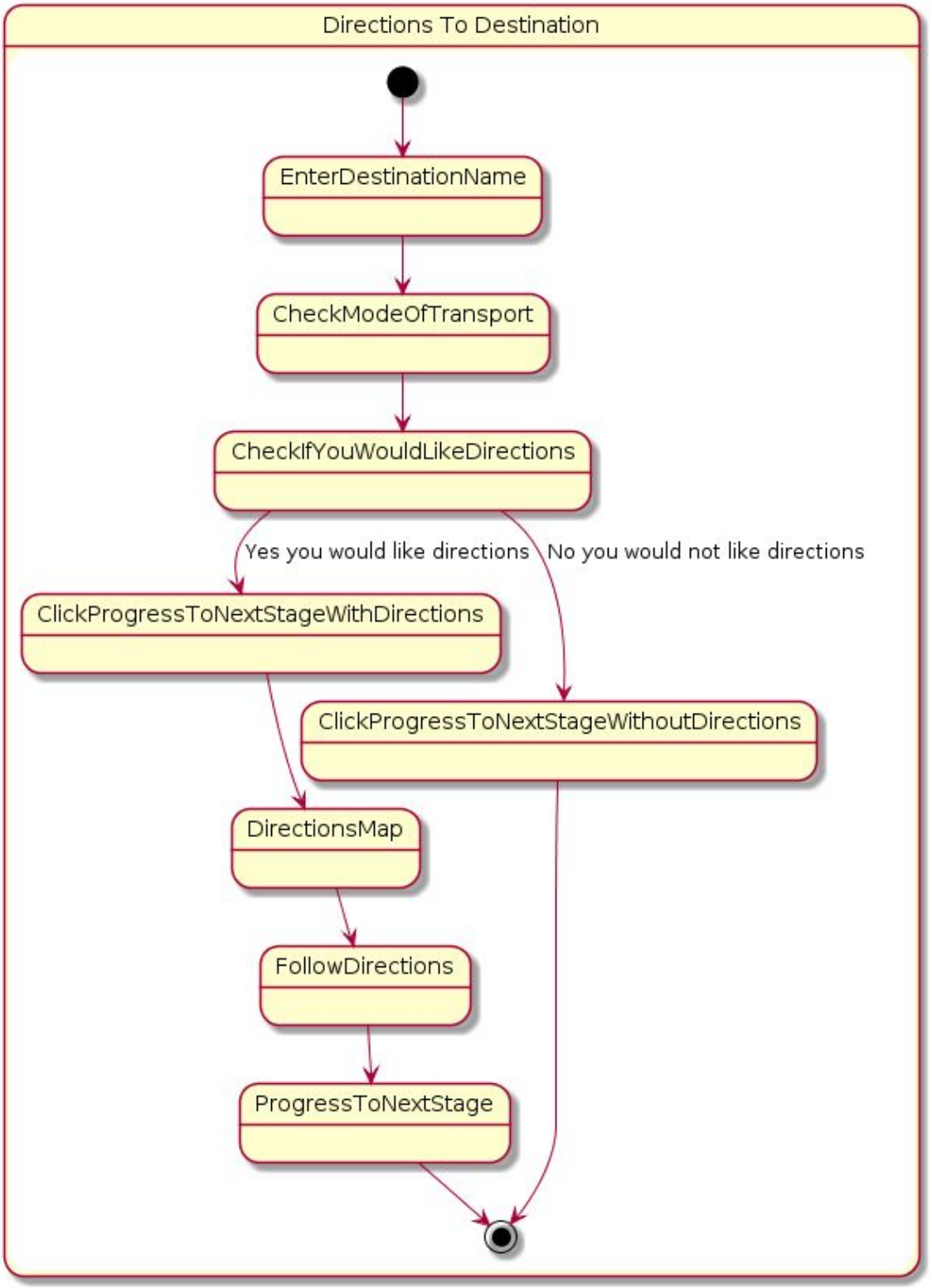


4.3. State Machine

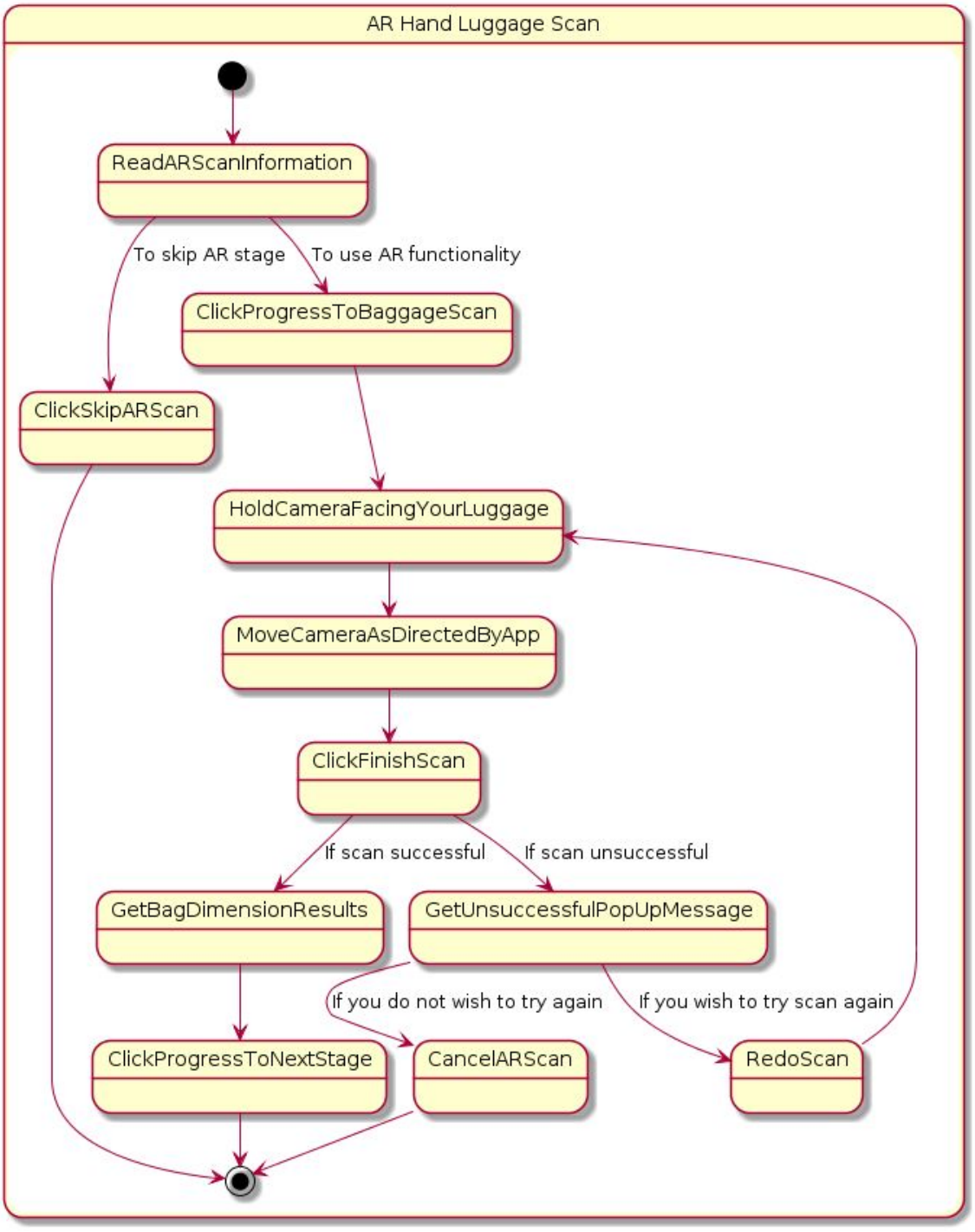
Upload Journey



Directions To Destination

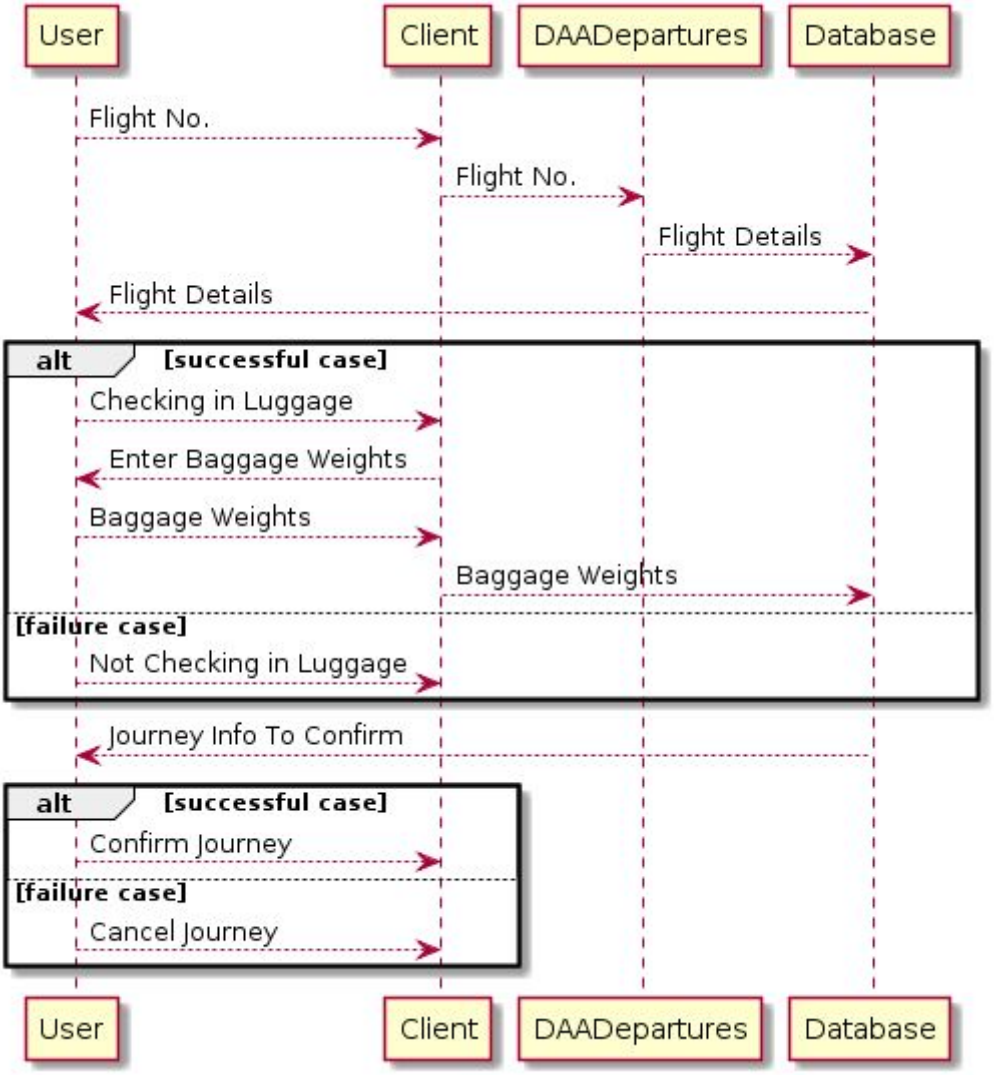


AR Hand Luggage Scan

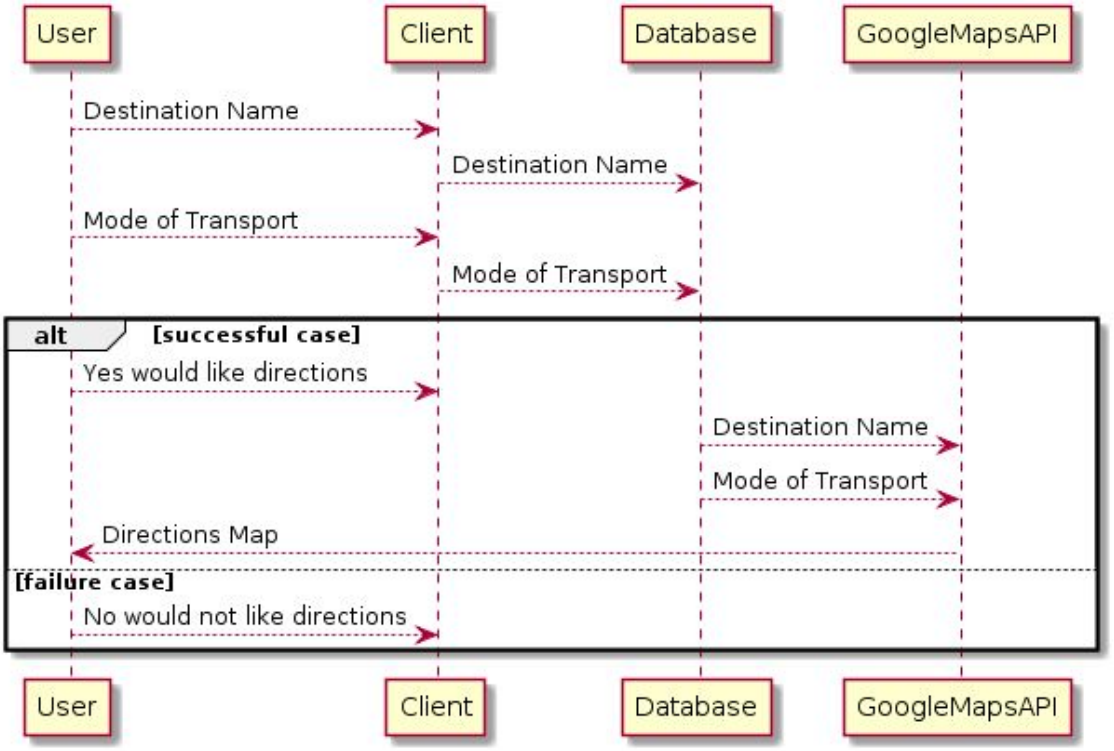


4.4. Sequence Diagram

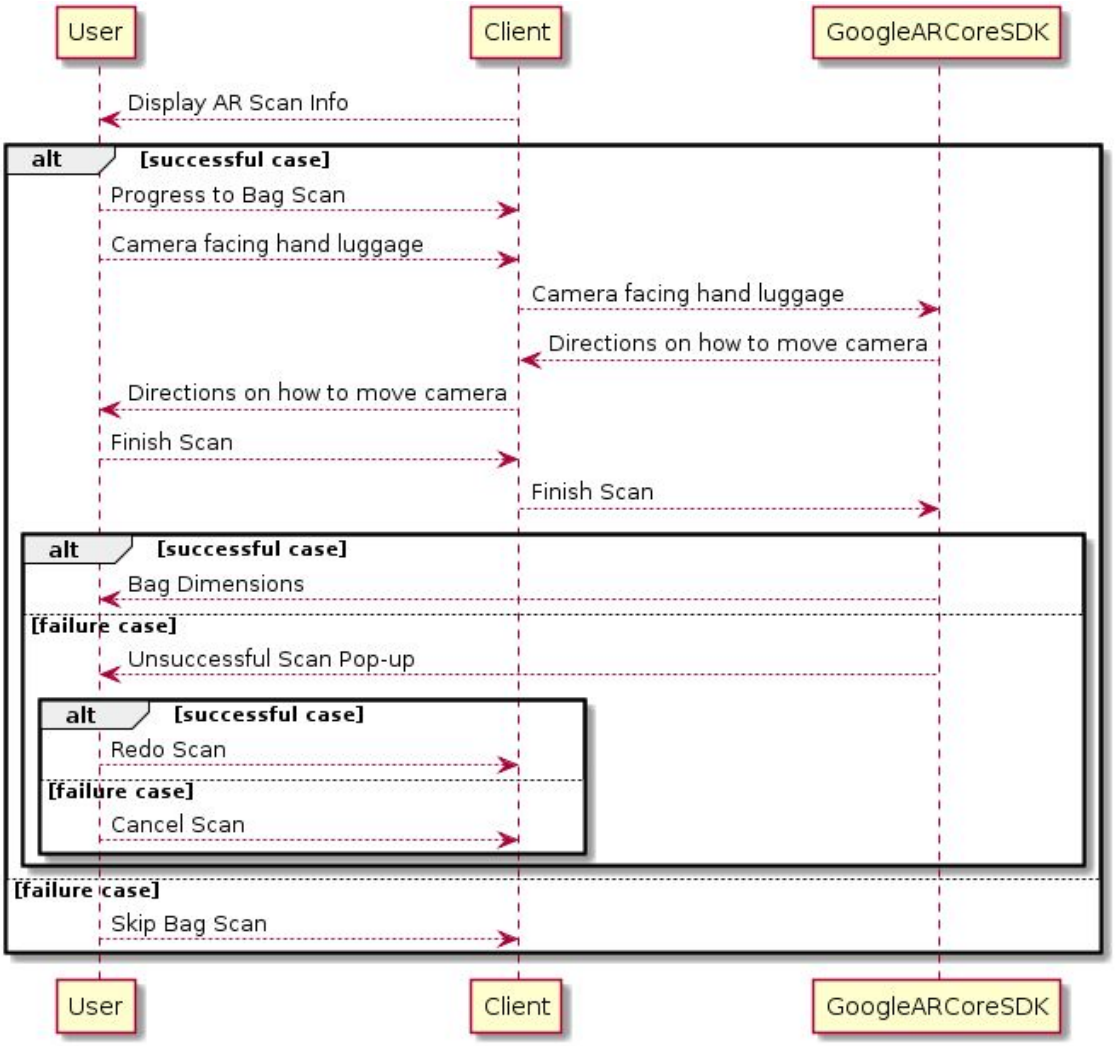
Upload Journey



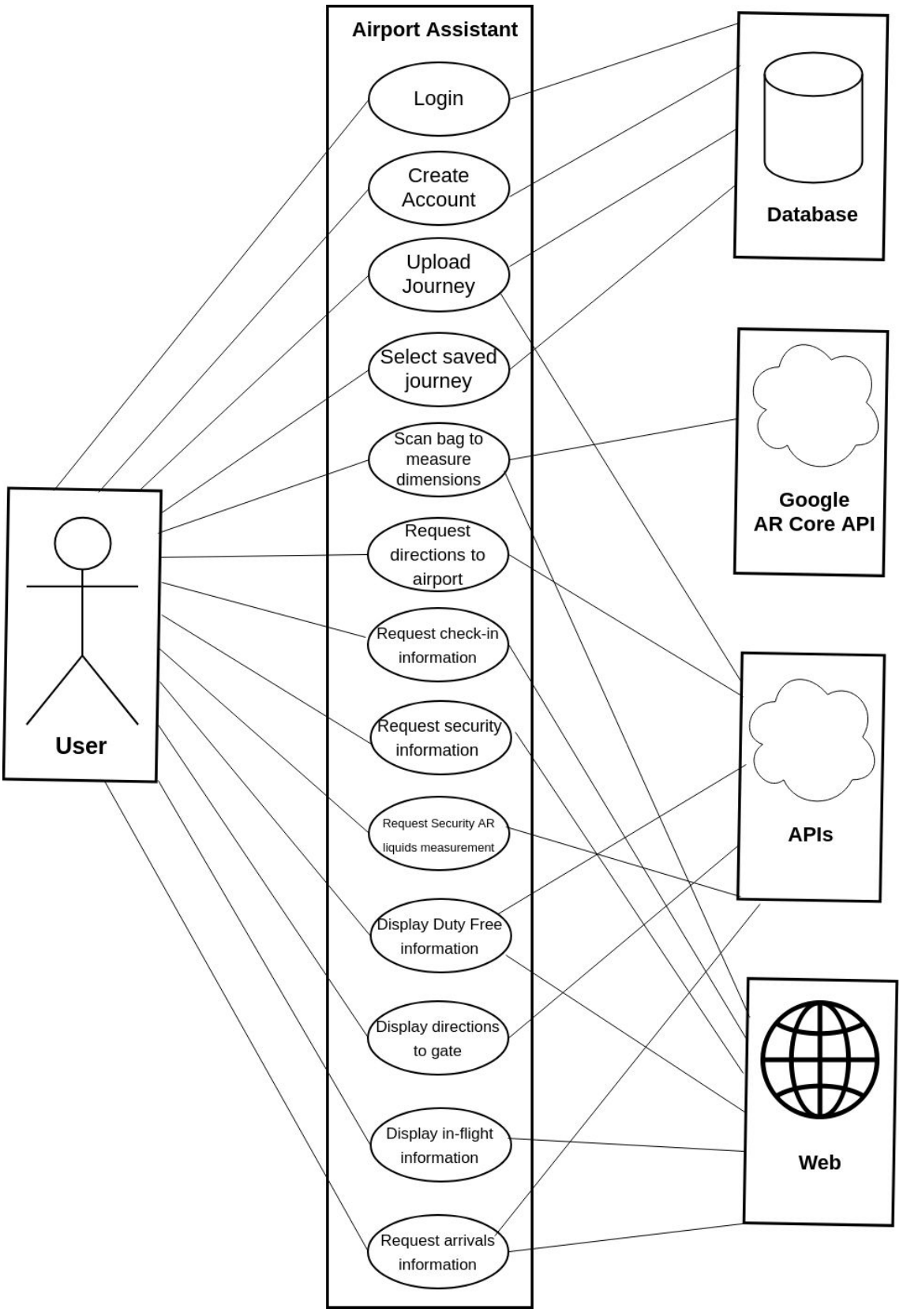
Directions To Destination



AR Scan Info


User
Client
GoogleARCoreSDK
Display AR Scan Info
alt
[successful case]
Progress to Bag Scan
Camera facing hand luggage
Camera facing hand luggage
Directions on how to move camera
Directions on how to move camera
Finish Scan
Finish Scan
alt
[successful case]
Bag Dimensions
[failure case]
Unsuccessful Scan Pop-up
alt
[successful case]
Redo Scan
[failure case]
Cancel Scan
[failure case]
Skip Bag Scan
User
Client
GoogleARCoreSDK

4.5. Use Case Diagram



5. Implementation

5.1. Sample Code

5.1.1. AR Functionality

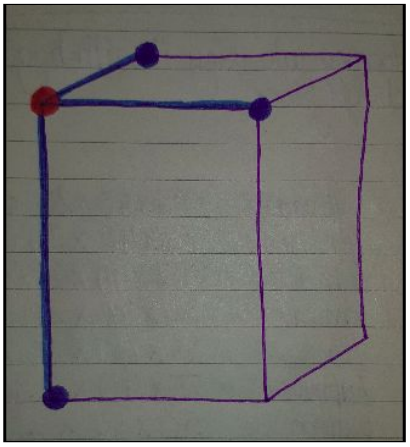
ARUtil.java

```
public class ARUtil {

    public ARUtil() { }

    public double getCubeVolume(double d1, double d2, double d3) {
        return d1 * d2 * d3;
    }
    public double getCylinderVolume(double d1, double d2) {
        double radius = d1/2;
        double radiussquared = Math.pow(radius, 2);
        double result = (3.14159*radiussquared*d2);
        Log.d("CylinderDimensions", "cylinder" + result);
        return (3.14159*radiussquared*d2);
    }
    public boolean passFailResult(double volume){
        if (volume <= 200) {
            return true;
        } else {
            return false;
        }
    }
    public boolean luggageResult(double userd1, double userd2, double
userd3, double airlined1, double airlined2, double airlined3){
         if(userd1 <= airlined1 && userd2 <= airlined2 && userd3 <=
airlined3) {
            return true;
        }
        else{
            return false;
        }
    }
}
```

I decided to implement a ARUtil file as part of the development of the AR functionality of my application. Each of the three different AR files call this file for calculations involved in their functionality. Here we can see the get cylinder and get cuboid volume calculations, luggage result calculations and finally the pass/fail result functionality.

ARScan.java - Nodes

```
if (baseNode == null) {
   baseNode = new AnchorNode(anchor);
   baseNode.setParent(arFragment.getArSceneView().getScene());
                    TransformableNode       model       =       new
TransformableNode(arFragment.getTransformationSystem());
   model.setParent(baseNode);
   model.setRenderable(modelRenderable);
   model.select();
   n++;
} else if (n < 4) {
   AnchorNode node = new AnchorNode(anchor);
   node.setParent(arFragment.getArSceneView().getScene());
   nodes.add(node);
                    TransformableNode       model       =       new
TransformableNode(arFragment.getTransformationSystem());
   model.setParent(node);
   model.setRenderable(secondaryModel);
   model.select();
```



The above file is responsible for setting the nodes on the AR Camera screen. There are two different types of nodes. An anchor node and three remaining base nodes. As seen in the diagram to left the user must place one anchor node and three base nodes. The anchor node is represented by a red ball, the three remaining base balls are blue. The reason for this is that the red balls acts as the center point. The distances to each of the three other balls is calculated from this central ball.

ARScan.java - Line

```
MaterialFactory.makeOpaqueWithColor(getApplicationContext(),      new
Color(0, 255, 244))
       .thenAccept(material -> {
           ModelRenderable mr = ShapeFactory.makeCube(
                   new Vector3(.01f, .01f, diff.length()),
                   Vector3.zero(), material);
           Float distance = diff.length();
           Float distcm = distance * 100;
           distlist.add(distcm);
           Log.d("lineBetweenPoints", "distance: " + diff.length());
```

```
            Log.d("distlist", "distanceList: " + distlist);
            Node n = new Node();
            n.setParent(node);
            n.setRenderable(mr);

n.setWorldPosition(Vector3.add(baseNode.getWorldPosition(),
node.getWorldPosition()).scaled(.5f));
            n.setWorldRotation(rotation);
        });
```

This snippet plots the line between each of these anchor nodes. It also creates a list of the distances between each of these points. This will later be used to check the dimensions of the hand luggage against the maximum hand luggage for each airline.

5.1.2. Maps Functionality

ArrivalsMap.java

```
getGeoContext();
DirectionsResult dr = DirectionsApi.newRequest(getGeoContext())
            .origin(new  com.google.maps.model.LatLng(origin.latitude,
origin.longitude))
                                                    .destination(new
com.google.maps.model.LatLng(destination.latitude,
destination.longitude))
        .mode(travelMode)
        .departureTime(now)
        .await();
if (dr != null) {
    addPolyline(dr);
    displayEndLocationTitle(dr);
}
return dr;
```

This file maps the directions between the origin and the destination. It does so by calling the directions api and mapping the polyline that connect the two points. I passed the origin, destination, travel mode and departure time into the api to calculate a route specific to the users needs.

5.1.3. Web Scraping

Security.java

```
protected String doInBackground(Void... params) {
   try {
                                          Document      document      =
Jsoup.connect("https://www.stuttgart-airport.com/security-wait-times
/").get();
       Element table = document.select("tbody").first();
       Element row = table.select("tr").first();
       if (row != null) {
           Element column = row.select("td").last();
           if (column != null) {
                   waitTime = Integer.parseInt(column.text().split("
")[0]);
           }
       }
}
```

Here I am scraping the security wait times in Stuttgart airport. As this information is not readily available from Dublin Airport I am scraping it from elsewhere. Here I am using Jsoup to scrape this information.

5.1.4. Firebase

Writing to Firebase

```
public void setWeightValues(){
   FirebaseUser user = mAuth.getCurrentUser();

   resultBookedWeight = editBookedWeight.getText().toString();
   resultActualWeight = editActualWeight.getText().toString();

   if (checkBagChoice.equals("Yes")) {

mUserRef.child(user.getUid()).child("bookedWeight").setValue(resultB
ookedWeight);

mUserRef.child(user.getUid()).child("actualWeight").setValue(resultA
ctualWeight);
   }
}
```

Here I am writing to the Firebase database. I am setting the weight values that the user sets in the set check in luggage screen. These values are added to the database user branch as bookedWeight and actualWeight.

Reading from Firebase

```
public void onDataChange(@NonNull DataSnapshot dataSnapshot) {
   data = dataSnapshot;
   DataSnapshot user = data.child("users").child(mAuth.getUid());
   if (user != null) {
                                       String     actualWeight     =
user.child("actualWeight").getValue(String.class);
                                       String     bookedWeight     =
user.child("bookedWeight").getValue(String.class);
   }
}
```

This snippet displays how I read data from the database. Here I am taking the values actualWeight and bookedWeight from the database.

5.2. Technologies Used

As this is an Android application the primary technology I worked with was Android. While developing on this platform I wrote my project in Java and XML.

I used a number of Google APIs including:

- Places API
- Directions API
- Geocoding API
- Distance Matrix API
- Geolocation API

I also used Google SDKs such as:

- Maps SDK for Android
- ARCore SDK

On top of this I used Google Sceneform project to implement the AR functionality of the application. The web scraping functionality was developing using JSoup. To test Airport Assistant I used Junit4 and Espresso.

6. Problems and Resolutions

- AR Sceneform Project Required API level
- AR beta version
- Google Pay for Passes API
- Dublin Airport Information not freely available

AR Sceneform Project Required API level

Problem

Unfortunately, the AR Sceneform project which is used for working with the Google ARCore SDK requires Android API level 27. I have a Samsung S8 which is currently at API level 25 and my laptop is unable to run an emulator. This means I must use a device with the latest form of android. As of March this has been released in Ireland on the Samsung S9 but not yet on the S8. It is expected to be released in the next month.

I decided to use the Google Sceneform project as I believe it's the best way to progress with my project as it assists with certain aspects of working with ARCore.

Solution

To combat this issue I spent some time running the application on a friend's phone until it was possible to do so on my own. This limited me to running my application at certain times. I spent other times working on testing and documentation as to not waste time.

AR beta version

Problem

I ran into a number of issues using the Google ARCore SDK. Due to how new of a technology AR is, ARCore is still in its beta version. This means that there are a number of bugs present in the software that are still being flattened out. Unfortunately, working with a software that is in beta means that one is bound to run into such issues.

Solution

I started working with this technology early in the development process so had the time to work through such problems.

Google Pay For Passes Issue

Problem

Google Pay for Passes did not work as first expected. I initially expected it to act like the Google Pay API which can be integrated into applications. Unfortunately, the

purpose of the API is to be built into applications, such as the Ryanair app, so that users can save their boarding pass to the Google Pay API and not for displaying boarding passes as I had first expected. This creates two issues. Firstly, I cannot display boarding passes and secondly, scanning boarding passes was how I intended to take in a user's flight information.

Solution

To resolve this issue I will ask a user the enter their flight number. I will then scrape the flight information from the Dublin Airport Departures site. This will mean that users will not have to enter all of their flight details which results in the application being easier to use. Unfortunately, I will not be able to display boarding passes in Airport Assistant.

Dublin Airport Information not freely available

Problem

Unfortunately, Dublin Airport does not have information such as gate numbers, check-in desks and security wait times available online. This caused difficulties as I wanted to direct users to their check-in gate, their departure gate and provide them with information on security wait times.

Solution

To combat this issue I ask users to check the check-in board to find their check-in gate. Users are also asked to enter their gate number which they are then directed to. Finally to combat the security wait time issue, I have scraped this information from Stuttgart Airport. As this application is a prototype I haved scraped this information to show how this information would be scraped from Dublin Airport if this info was available online.

7. Future Work

Future work for this application would be to delve further into the mental health aspect of the application. This would involve a button on every screen where users can access inspirational quotes or calming mechanisms including breathing exercises to help them move through the process. Users could mark if a message or exercise proved effective for them and the system could learn to recommend items that the user likes.

At the moment the prototype was built for Dublin Airport. In future the application would work for every airport in the world. This would be especially important in guiding users through airports which they are unfamiliar with and where surrounding signs may be through a foreign language.

Issues which are currently present in the application such as check-in desk numbers, gate numbers and security wait time information not being available for the application could all be added when they become available for Dublin Airport. Users could therefore only have to enter their flight number and will then have their check-in gate and departure gate numbers scraped by the application. They would be provided with push notifications when this information becomes available from the airport. This information would then be added to the journey information tab in the application.

Finally other aspects of the airport experience such as shopping in the The Loop Dublin Airport and picking items up on your way home could be integrated into the application. This could also be applied to boarding passes.

8. Installation and Configuration

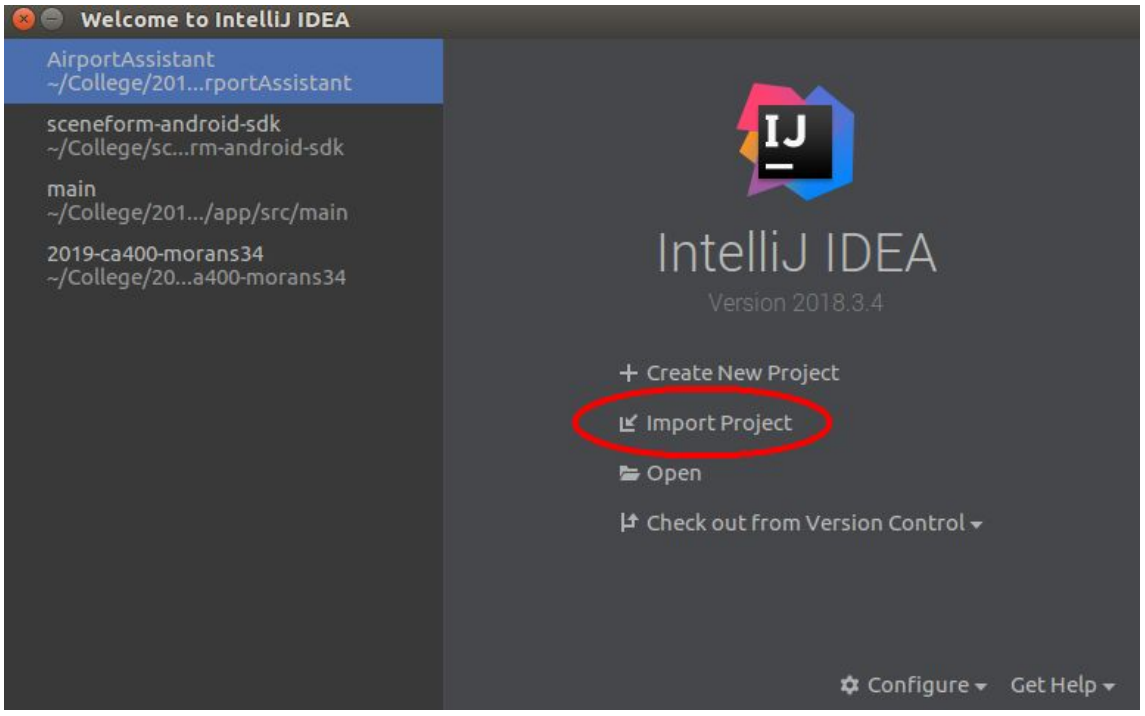
Required Software

- Intelliji Ultimate version 2018.3.4
- Java 8

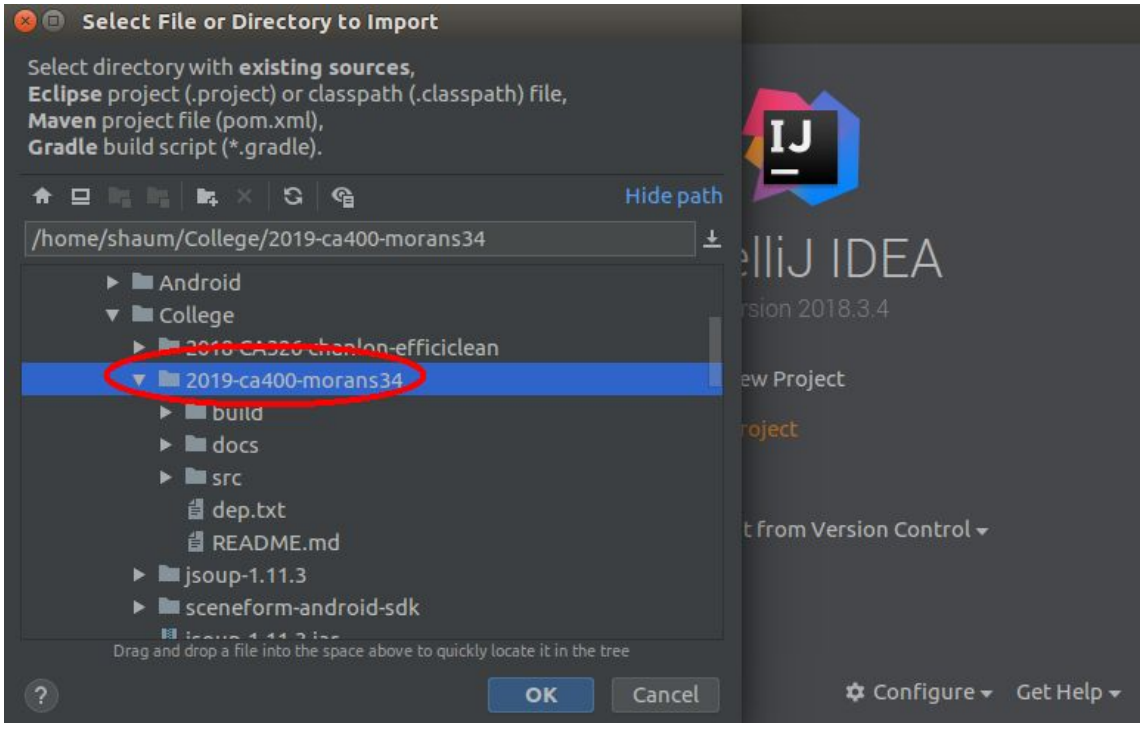
Instructions

Building the Android Project

- Create a folder where you would like the project to be stored
- Open a terminal
- Navigate to chosen folder
- Execute command: git clone https://gitlab.computing.dcu.ie/morans34/2019-ca400-morans34.git
- Open Intelliji
- Select Import Project as seen below



- Navigate to “C:\Users”YOUR CREATED DIRECTORY PATH” \2019-CA400-morans34”



- Click “Ok”
- The project will now build

Running the Application

- Connect your device to your laptop using a USB cable
- On the Toolbar at the top of the page click "Run"
- Select "Run App"
- Select your device
- Click "Ok"
- The application will now run