Table of Contents



**GitHub Username**: Jacko1972

# Aviate

## Description

An App to provide access to important and the latest Aviation weather information issued by Aerodrome/Heliports (AD/HPs) globally.

## Intended User

Anyone who requires or wants access to Aviation Weather reports from aviation enthusiasts to Commercial pilots.

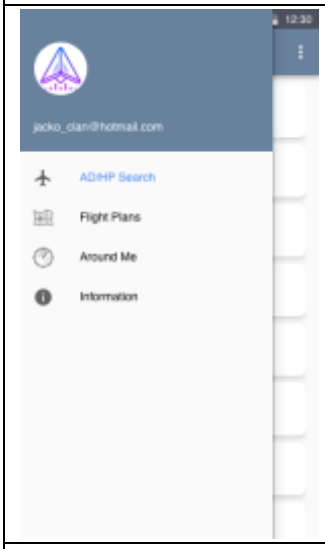
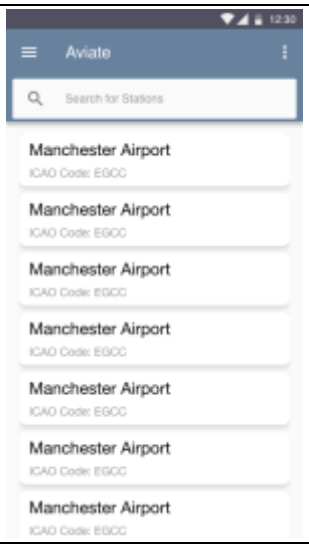
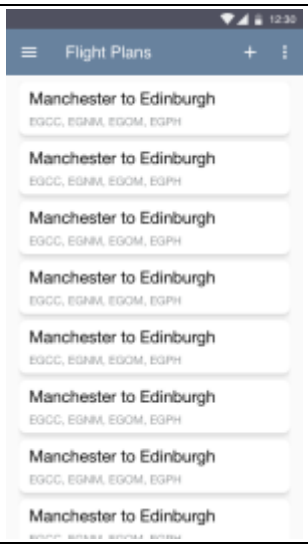
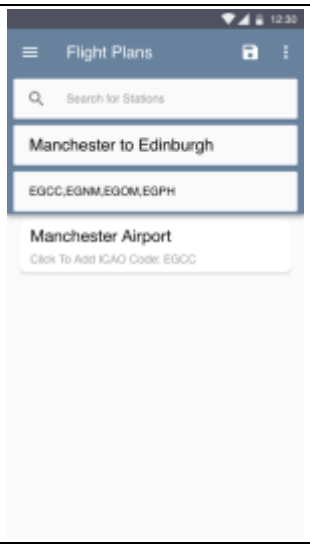
## Features

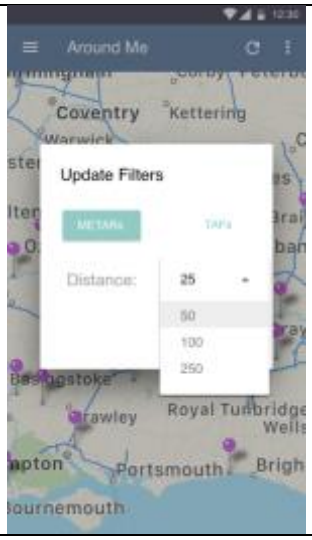
Fit Me App has the following features:

- Provides access to current weather information by AD/HP.
- Displays a list of flight the user can create and save on the device so one click can download all relevant information on the flight path.
- Using Google Maps the user can display the information on a Map around their current location.
- Provides a service to download the information for one AD/HP provided by the user and display it in a widget.

# User Interface Mocks (Phone)

Created with the Adobe Creative Cloud program, Experience Design (Beta).

| Navigation Drawer | AD/HP<br>Search Screen | Flight Plan List | Flight Plan Builder |
|---|---|---|---|
|  |  |  |  |
| A variety of screens to cover the UI layouts used in the Application. | | | |

| Around Me Map | Around Me Filter Dialog | Widget |
|---|---|---|
|  |  |  |
| A variety of screens to cover the UI layouts used in the Application. | | |

# User Interface Mocks (Tablet)

UI layouts for Tablets will follow Material Design specifications following the Phone layouts above. Predominantly the App will be used on a phone, due to ease of stowing a phone whilst flying.

# Key Considerations

## How will your app handle data persistence?

- A Content Provider will hold a persistent record of the Flight Plans and the stored data will be used to provide the user with quick access to weather information along the flight path.
- A Loader will be used to fill a list view of Flight Plans.

## Describe any corner cases in the UX.

- A Navigation Drawer will provide access to the different functions the App provides.
- When required an icon will appear on the top right of the screen to provide extra navigation options.

## Describe any libraries you'll be using and share your reasoning for including them.

- Android Design Support Libraries – Material based design and other fantastic features.
- Retrofit2 – Used to download data from the data source.
- Schematic – Creates a Content Provider backed by an SQL database.

## Google/Firebase Services

- Google Play Services for Location based information
- Google Maps API for display of Maps and locations of Weather information.
- Google Firebase Analytics and Crash Reporting

# Next Steps: Required Tasks

## Task 1: Project Setup

- Create project in Android Studio
- Create Classes

## Task 2: Google APIs setup

- Create Project in Google Developer Console
- Create Google Maps API Credentials
- Import Project in Firebase Developer Console
- Setup Firebase Analytics for use in Project

## Task 3: Implement UI for Each Activity and Fragment

- Build a UI for Aerodrome/Heliport Search and a UI for displaying selected AD/HP information.
- Build a UI for Flight Path listing and creation.
- Build a UI for displaying Weather Information around the user's current location, provide filters for the two different types of information and allow to set a distance from the location of the user.
- Build a UI for Instructions for User

## Task 3: Implement App Logic and Libraries/APIs

- Implement Firebase Analytics and Crash Reporting
- Implement Google Maps integration

- Create Content Provider for storing Flight Plans
- Create Loader to provide a list of stored Flight Plans
- Generate access to ‘fake’ Weather Information source, as this is a development app, data will be downloaded via web interface that will be turned off once project is completed.

## Task 4: Create Widget

- Build a widget that displays the latest downloaded information from the service built into the App.

---

**Submission Instructions**

1. After you’ve completed all the sections, download this document as a PDF [ File → Download as PDF ]
2. Create a new GitHub repo for the capstone. Name it “**Capstone Project**”
3. Add this document to your repo. Make sure it’s named “**Capstone_Stage1.pdf**”