Want to fly

COMP 3717 Term Report



By: Ethan Lee, Arunab Singh, Dani Jin, Harry He

Report Date: Nov 27, 2022

Contents

[I. Project Description 4](#_Toc120457315)

[1. Project Overview 4](#_Toc120457316)

[2. The Purpose of the Project 4](#_Toc120457317)

[3. Product Scenarios 4](#_Toc120457318)

[II. Requirements 5](#_Toc120457319)

[4. Functional Requirements 5](#_Toc120457320)

[Requirement 1: Use Case 3 5](#_Toc120457321)

[Requirement 2: Use Case 1 5](#_Toc120457322)

[Requirement 3: Use Case 2 5](#_Toc120457323)

[Requirement 4: Use Case 4 6](#_Toc120457324)

[Requirement 5: Use Case 6 6](#_Toc120457325)

[Requirement 6: Use Case 5 6](#_Toc120457326)

[5. Data Requirements 7](#_Toc120457327)

[III. Design 7](#_Toc120457328)

[6. System Design 7](#_Toc120457329)

[6a. Class Diagrams 7](#_Toc120457330)

[6b. Subsystem Decomposition 7](#_Toc120457331)

[6c. Data Dictionary 8](#_Toc120457332)

[7. User Interface 8](#_Toc120457333)

[IV. Implementation 8](#_Toc120457334)

[8. Progress Documentation 8](#_Toc120457335)

[9. Meeting Documentation 9](#_Toc120457336)

[V. Test Plans 11](#_Toc120457337)

[10. Features to Test 11](#_Toc120457338)

[11. Pass/Fail Criteria 11](#_Toc120457339)

[12. Testing Cases 12](#_Toc120457340)

[VI. Project Issues 13](#_Toc120457341)

[13. Open Issues 13](#_Toc120457342)

[14. Risks 13](#_Toc120457343)

[VII. Glossary 13](#_Toc120457344)

[VIII. References / Bibliography 13](#_Toc120457345)

**List of Figures**

[Figure 1: Onboarding Screen 7](#_Toc120471345)

[Figure 2: Home Screen 7](#_Toc120471346)

[Figure 3: Flight Information Screen (top) 8](#_Toc120471347)

[Figure 4: Flight Information Screen (bottom) 8](#_Toc120471348)

**List of Tables**

[Table 1: Progression Outline 7](#_Toc120469691)

[Table 2: Weekly Discussion Notes 8](#_Toc120469692)

[Table 3: Test Cases 11](#_Toc120469693)

# 

# I. Project Description

## 1. Project Overview

Want to Fly is a travel companion application that delivers flight delay predictions to your phone before you arrive at the airport. Our application takes historical flight data as well as real time weather data to predict the likelihood that a given flight is likely to be delayed, cancelled or On-Time.

## 2. The Purpose of the Project

Everyday people travel via airlines be it for business or pleasure. However, given their importance, any changes in schedule can vastly change the itinerary of unwary travellers. When airlines have to make changes to flights, travellers often only find out this information once they get to the airport. Even if they are compensated for the flight, the rest of their plans may not be as flexible.

This lack of information is what motivated the creation of Want to Fly. Want to Fly looks to take publicly available information and project them onto the flight status of your flight. We all know that bad weather is a leading cause of delays and cancellation of flights but they can be hard to interpret yourself. Want to Fly looks to solve that issue through predictive analytics, giving the traveller all the information, they need before arriving at the airport terminal.

Want to Fly looks to give users critical information by parsing publicly available data. The main goal is to take weather data and historic flight data to predict the likelihood that a flight might be delayed or cancelled. Users can search within the app for their scheduled flight and the app will give them additional information that the airline company may ignore.

## 3. Product Scenarios

Use case 1: A user can search by their flight number

Use case 2: A user can search for flight routes through departure and arrival cities.

Use case 3: Given a flight route, the app will show relevant weather information

Use case 4: Given a flight route, the app will show the likelihood of delays and cancellation

Use case 5: Given a city, show relevant travel information polled from twitter.

Use case 6: The search bar should help the user fight their flight through an Auto complete.

# II. Requirements

## 4. Functional Requirements

### Requirement 1: Use Case 3

Description: Given a input city, the app should show the relevant weather information

Rationale: Since weather is the primary predictor of flight cancellation, the user should also be aware of the data. Weather information can also be used in pre-travel preparations as well.

Fit Criterion: The weather for the given city is displayed on screen through an API call

Priority: High

Supporting Materials: City Name, Weather API

History: Nov 25, 2022

### Requirement 2: Use Case 1

Description: The user can search for their flight through the flight number.

Rationale: This is the primary way a user can search for the relevant flight to their destination

Fit Criterion: Given a flight number, the app should show the correct flight information and predictions.

Priority: High

Supporting Materials: Flight Number, Weather API, Flight info API

History: Nov 19, 2022

### Requirement 3: Use Case 2

Description: The user can search for their flight through the arrival and departure city.

Rationale: Since flight numbers can be difficult to remember, the app should be able to search for the flight routes by city names.

Fit Criterion: Given an arrival and departure city, the app should show possible flight routes that the user can choose from as their flight number.

Priority: Medium

Supporting Materials: Flight Number, Weather API, Flight info API

History: Nov 19, 2022

### Requirement 4: Use Case 4

Description: Given a flight number, display the likelihood of flight delays and cancellations

Rationale: The primary purpose of our app. This should use our prediction models to give the user a cancellation score.

Fit Criterion: Given a flight number and using historic data and weather data, show a cancellation score that reflects the likelihood of delays and cancellations.

Priority: high

Supporting Materials: Flight Number, Weather API, Flight info API

History: Nov 24, 2022

### Requirement 5: Use Case 6

Description: The search bar should give the user suggestions on their intended input as they enter information.

Rationale: Help the user fight their flights faster and help prevent frustrating input errors.

Fit Criterion: As the user enters each letter of their search, the app should continually update suggestions based on what is input.

Priority: low

Supporting Materials: Advanced data structure, database on possible inputs.

History: Nov 19, 2022

### Requirement 6: Use Case 5

Description: On the flight information screen, show the user relevant tweets regarding the location and airline

Rationale: Since this app only considers historical flight data and weather, there may be other reasons why a flight might be delayed. Twitter as a medium can be searched as news snippets are often posted in real time.

Fit Criterion: On the flight information page, there should be a cycling feed of relevant information from twitter about the flight/city.

Priority: low

Supporting Materials: Twitter API

History: Nov 24, 2022

## 5. Data Requirements

For this project, we will require many publicly available API’s and databases:

Real time weather information

Historic Flight data

Twitter API

Real time weather information is used both as predictor data as well as updating the user with relevant information on their arrival and destination cities.

Historic Flight data is used to both help search for flight routes as well as for our application to assess an airline’s timeliness and likelihood of cancellations.

Twitter API is used to give the user more relevant information about their arrival and destination cities. As a user prepared to travel, they should be informed of local news such as traffic information or travel related new articles.

# III. Design

## 6. System Design

### 6a. Class Diagrams

Content

This section describes the types of objects and their relationships, including association, inheritance and aggregation.

Example

Refer to lecture notes about Requirement Engineering.

### 6b. Subsystem Decomposition

Content

This section decomposes the system into cohesive, well-defined subsystems.

Example

### 6c. Data Dictionary

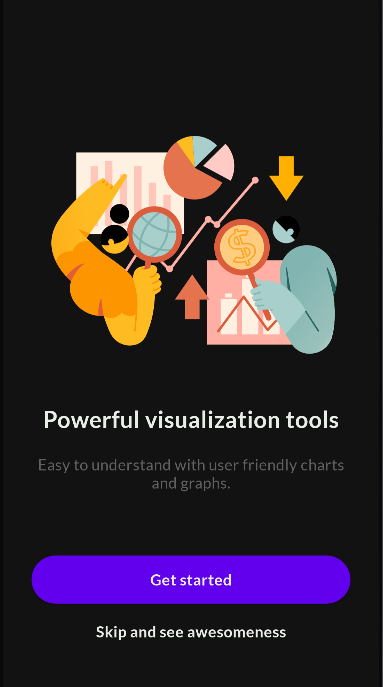
This section contains a collection of names, definitions, and attributes about data elements that are being used or captured in our software system.

Data dictionary is used to avoid data inconsistencies across a project and make data easier to analyse by enforcing the use of data standards.

Table 1: Data Dictionary

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Filed Name | Data Type | Filed Length | Constraint | Description |
| flight\_number | Int | 10 | Primary key | Flight number |
| departure\_delay | Int | 5 | / | Number of delayed flights |
| arrival\_airport | Varchar2 | 30 | Not null | Arrival airport name |
| airline\_name | Varchar2 | 30 | Not null | Airline name corresponding to the flight number |
| flight\_status | Varchar | 10 | Not null | Flight status, either cancelled or other status |
| windspeed | Double | 8 | Not null | Windspeed of the destination city |
| temperature | Double | 8 | Not null | Temperature of the destination city |
| lon | Double | 8 | Not null | Longitude of the destination city |
| lat | Double | 8 | Not null | Latitude of the destination city |

## 7. User Interface

As users enter our app for the first time, they are greeted by a helpful onboarding screen that gives a brief overview of the app’s capabilities.

The user can swipe left and right to see the various pages of the onboarding.

Figure 1: Onboarding Screen

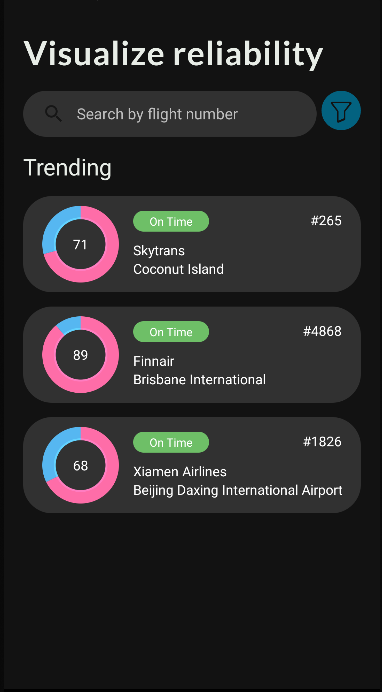
Next the new user and returning users will land on the home screen where different information will be presented depending on the user’s interactions. By default, the app will display a search bar at the top of the screen, and a list of trending flights. Users that have already used the app will see a section containing their most recent searches above the trending flight section.

Figure 2: Home Screen

Within the search bar, there are options to filter the results based on “on-time”, “delayed” and “canceled” which will exclude certain flights that have specific tags

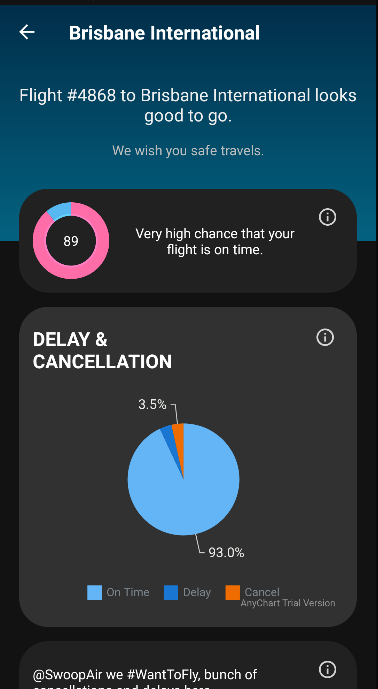
Once the user finds the flight they are looking for, clicking into the flight will bring up a flight details page. Here the user can find detailed information about their flight and destination cities. The user will first see a section that gives the WTF score for the current flight. Following the score, the user can find a detailed breakdown on the flight status.

Figure 3: Flight Information Screen (top)

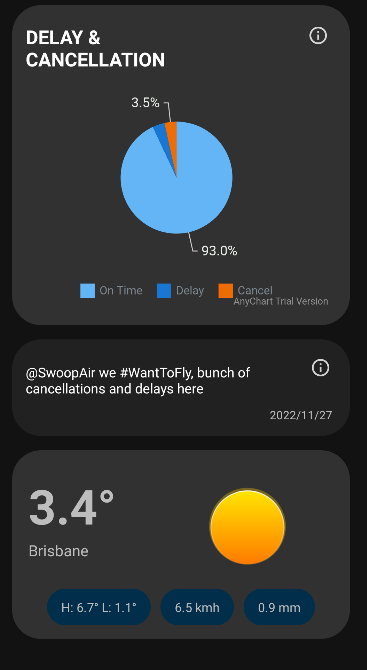
Under the detailed flight information, the user can find a Twitter ticker. The ticker shows relevant tweets about the destination city compiled in real time. Lastly, the flight information page displays the weather information of the destination city. Here the user can find the current temperature, daily temperature range, current windspeed, and precipitation.

Figure 4: Flight Information Screen (bottom)

# IV. Implementation

## 8. Progress Documentation

Table 2: Progression Outline

|  |  |  |
| --- | --- | --- |
| **Dates:** | **Progress** | **Contributions** |
| Week 2: | Project discussion  Conceptualization | Everyone: planning and discussion |
| Week 3: | UI design  Project Skeleton  API Research | Ethan: UI design, Project Skeleton  Arunab: UI design, Project Skeleton  Dani: API research  Harry: API research |
| Week 4: | API Research | Everyone: API research |
| Week 5: | Onboarding Screen  Splash Screen  API Feasibility | Ethan: Onboarding/Splash Screen  Arunab: Onboarding/Splash Screen  Dani: Weather API feasibility  Harry: Historic Flight database |
| Week 6: | Flight Details page | Ethan: Flight Details Page  Arunab: Flight Details Page |
| Midterm Week: | Break for Exam Season | Ethan:  Arunab:  Dani:  Harry: |
| Week 8: | Search page  Predictive Model initial exploration | Ethan: Search page  Arunab: Search page  Dani: PM initial exploration  Harry: PM initial exploration |
| Week 9: | Search Functionality  Model development  Report and Presentation Planning | Ethan: Search Functionality  Arunab: Search Functionality  Dani: Model development  Harry: Report Initial planning |
| Week 10: | API integration  Model Integration  Report Writing | Ethan: API integration  Arunab: API integration  Dani: Model Integration  Harry: Report Writing |
| Week 11: | App Finalisation  Report Finalisation  Presentation Finalisation | Everyone: Project finalisation |

## 9. Meeting Documentation

Table 3: Weekly Discussion Notes

|  |  |  |
| --- | --- | --- |
| **Date:** | **Time:** | **Content:** |
| Sept 19, 2022 | 12:30PM | Topic: General Discussion:  UI design resources  Meeting structures  GitHub repository  Activities discussion  User flow diagrams  API and Database research  Colour Scheme  Results: This meeting was to facilitate the start of the investigation. We listed the perceived requirements for the project and started researching the required resources. |
| Sept 30, 2022 | 1:30PM | Topic: Resources Finalization  Initial Conceptualization of UI layouts  API Research results  Usable API/Databases/PA model requirements  Results: Ethan presented on the initial conceptualization of the app UI and everyone agreed with the visual direction. We received a reply from the US DoT which pointed us in the direction of a historical flight data database that we can use to create our PA model |
| Oct 7, 2022 | 1:30PM | Topic: App main screens  Onboarding Screen  Weather API investigation  PA model discussion  Coding Requirements  Results: Ethan demonstrated a set of Onboarding screens that he created. We still need to finalise the graphics still but the implementation was accepted.  We decided on moving forward with the Weather API inclusion in our MVP. Reevaluated the needs for the PA model. |
| Oct 14, 2022 | 1:30PM | Topic: Progress check, short discussion due to midterms coming up  Current progress on app development and research.  Meeting ended early for the sake of midterms  Result: Only updates on the App was presented, no group decision making occurred. |
| Oct 28, 2022 | 1:30PM | Topic: Search Screens and PA model investigation  Discussed requirements for the search bar functionality  Finalise the requirements for the PA model  Result: For the search bar, we decided on the need to include both “search by flight number” and “search by city” functionality, with autocomplete being a nice to have if time allows.  We decided that the PA model should factor historic flight data as well as current weather data. |
| Nov 4, 2022 | 1:30PM | Topic App development update, API and Model integration, End of term report planning  Presented the progress of the app development  Began API integration for key features  Model integration along with API  App report preparations  Result: Weather API integrated into the app; the usages of weather data is still being discussed. PA model requirements were updated to include wind speed and direction. Work on information gathering for term report to begin the following week. |
| Nov 11, 2022 | 1:30PM | Topic: Progress report, short meeting due to holiday |
| Nov 18, 2022 | 1:30PM | Topic: App finalisation, Report drafting  Identified remaining features that still need to be implemented.  Temporary fix for Weather API call limit  Report information accumulation and starting initial draft  Results: App was considered 90% complete, last minutes touches still need to be made  Hard coded weather data is being used currently to conserve the limited number of API calls we can make.  Final report initial draft started. Revisiting project mission and MVP. |

# V. Test Plans

## 10. Features to Test

The following main features are scheduled for testing:

1. Onboarding screen flow
   1. Search screen
   2. search autocompletes
   3. Display recent searches
   4. Display Trending flights
   5. Sorting bottom sheet fragment
   6. Donut chart displaying flight rating
2. Flight details screen
   1. Donut chart displaying flight rating
   2. Pie chart displaying chance of on time, delay, cancellation
   3. Related Twitter news feed
   4. Weather information

## 11. Pass/Fail Criteria

1. Onboarding screen flow: On-boarding screen should scroll as expected and images appear as implemented
2. Search screen:
   1. As the user enters the third letter in their query, the autocomplete should display suggestions that are updated as more inputs are added
   2. Search history section should display a list of recent searches in reverse chronological order
   3. Trending flights section should show and expected number of popular flights for the user to select
   4. Bottom sheet fragment should be sorted in and expected manner
   5. Flight rating donut should display the appropriate colours and ratios based on the model output.
3. Flight Details Screen:
   1. Flight rating donut should display the appropriate colours and ratios based on the model output.
   2. Flight info pie chart should display the proper values from the model predictions
   3. The twitter news feed should cycle through a predetermined number of current tweets based on location
   4. The weather information should display the current weather information from the API (or hard coded weather data)

## 12. Testing Cases

The following tables contain the list of test cases for each of our features:

Table 4: Test Cases

|  |  |  |
| --- | --- | --- |
| Scenario | Name | Test Case |
| 1 | Onboarding Screen flow | cycling through the onboarding screen back and forth |
| 2a | Search Autocomplete | Enter a city name one character at a time. |
|  |  | Enter a nonsensical word and visually inspect suggestions |
| 2b | Search History | Conduct a series of queries in the search bar |
| 2c | Trending Flights | Visually inspect that there are 3 views presented |
| 2d | Bottom Fragments | Perform a search, open the sorting bottom sheet fragment, and then sort by each possibility. |
| 2e | Flight Rating Donut | Enter a known flight number and visually inspect the flight rating |
| 3a | Flight Rating Donut | Enter a known flight number, tap into the flight information page and visually inspect the flight rating |
| 3b | Flight Info Pie Chart | Enter a known flight number, tap into the flight information page and visually inspect the pie chart |
| 3c | Twitter News Feed | Enter a known flight number, tap into the flight information page and visually inspect the Twitter feed. |
| 3d | Weather Widget | Enter a known flight number, tap into the flight information page and visually inspect the weather widget |

# VI. Project Issues

## 13. Open Issues

Currently, our major issue concerns with Flight API (API) call limits. Without subscribing to their service, we have only a limited number of Flight API calls which severely restricts feature testing. We replaced the live data with hard coded data for the purpose of development and until a better solution is found for live weather data.

Issue with retrieving weather data of destination location:

In the initial stages, our team decided to use the open-mateo API for weather data, since it didn’t require any API keys and we could make HTTP GET requests very easily. Later, we ran into an issue, the issue was that the API only returned weather data for location coordinates (longitude and latitude) and we only had access to the destination airport name to get the weather data.

We navigated this problem by using the free Geocoding API provided by OpenWeatherMap.org. We first got the coordinates of the destination airport by integrating our Geocoding API and then querying the open-mateo API for the weather details.

## 14. Risks

Our app currently does not save persistent data to a database. This will increase the number of inputs from the user which can be seen as tedious. The was deemed unimportant for the purpose of an MVP.

The app currently does not have any tracking/user data analytic functionalities. There is no way for the app to serve tailored content.

The app currently lacks device location services which would reduce user inputs for search purposes. This was deemed unimportant for the purposes of an MVP

Given the time restraints placed on this project, there was no user/focus group testing to gather usage data. We developed with what we think the user needs but lacks the time and funding to conduct proper user surveys.

Along with the same concerns on timing, our app lacks extensive QA/QC testing. While bugs found during developments are fixed, we do not have the resources to test the robustness of our interfaces and functionalities.

Given the scope of this project, this service is only being released for the Android environment. While this is the only requirement for the MVP, we lack access to the iOS user base.

# VII. Glossary

The glossary defines terms that may not be familiar to all readers.

# VIII. References / Bibliography

This section describes the documents and other sources from which information was gathered.

EazeGraph Android by Paul Roehr

https://github.com/paulroehr/EazeGraph

AnyChart Android

https://github.com/AnyChart/AnyChart-Android