GitHub Username: evindj

travlers

Description

Travlers gives you the power to browser among a myriad of flights each cheaper than others and choose whatever suits your budget. A full description of your itinerary will be provided as well as a glance of few attractions you may want to visit during your stay wherever you are going.

Intended User

Travlers(). is primary intended user.

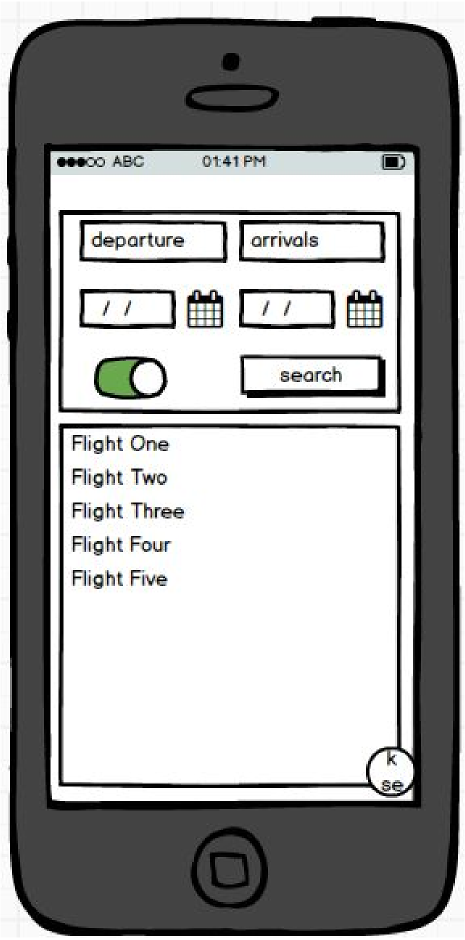
Features

List the main features of your app. For example:

- query and display the list of flight tickets
- sort the list of of flight tickets and display from the cheapest to the most expensive.
- display points of interest on a map activity with markers.

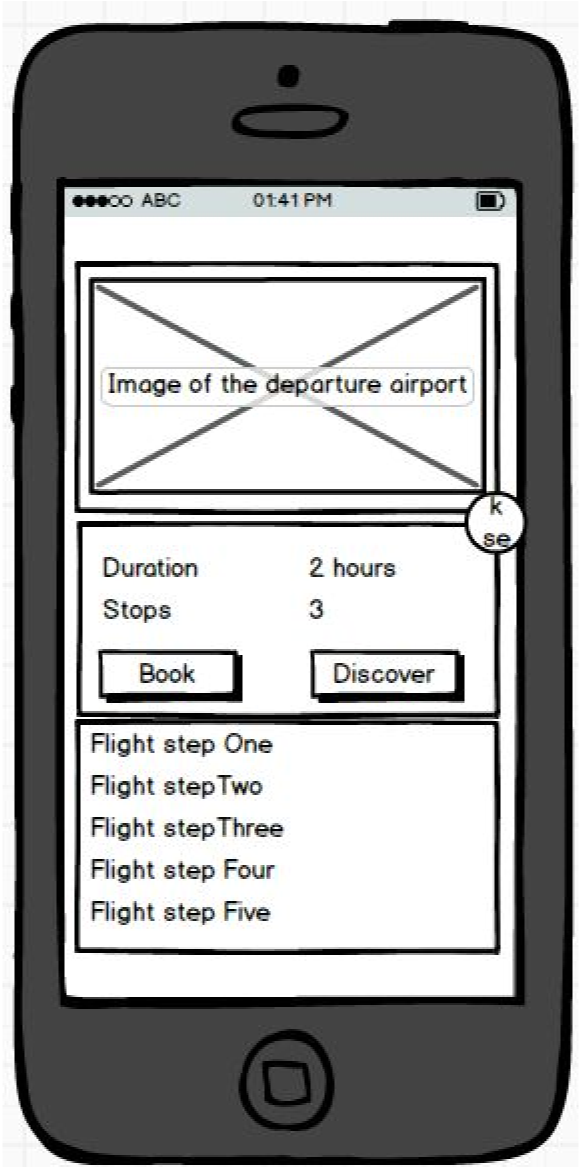
User Interface Mocks

Screen 1



This screen will have two separated boxes.
The top container will display the form to search for flights based on user's inputs . The container at the bottom of the page will display a default list of flight that will be requested based on random criterias or an empty view if the user is not connected to the internet.
The fab button will be used to launch the search based on previously saved criterias that will be stored in sharedpreferences.

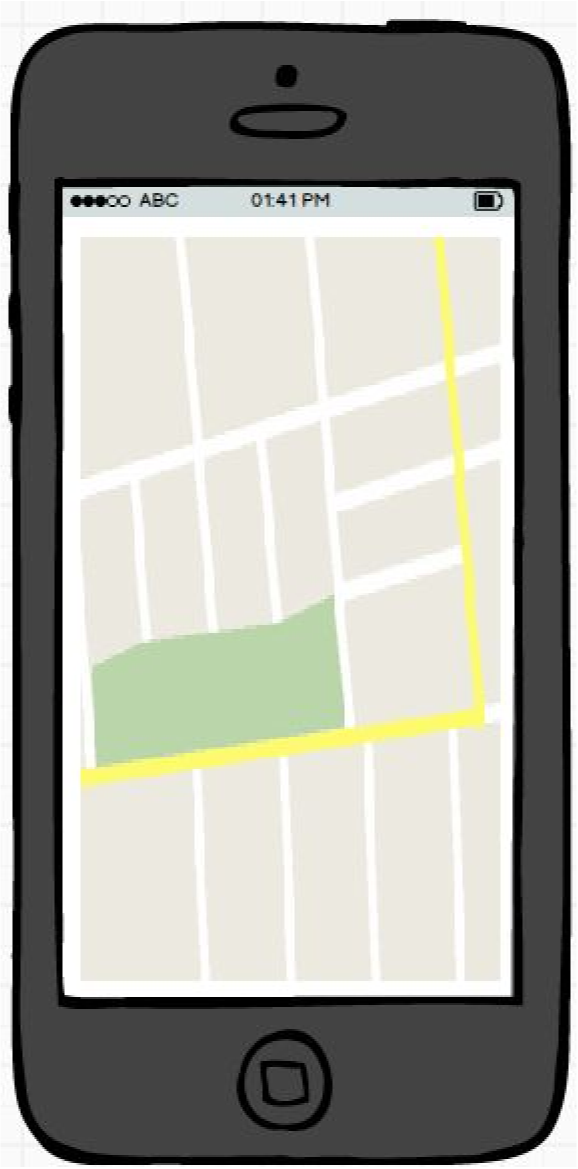
Screen 2



This screen will display the details of the selected flight. the top portion of the screen will display an image of the airport and as well as a label describing. the user will have the possibility to book the flight directly via the “book” button or to view some points of interests by clicking the “discover” button.

Finally, the itinerary of the flight will be be displayed at the bottom of the screen.

Screen 3



This screen will display the list of key point of interest on a map.

Key Considerations

How will your app handle data persistence?

I’ll build a content provider that queries data from other popular flights booking websites. Share preferences will be used to save the favorite user search. retrofit will be used to query data asynchronously.

Describe any corner cases in the UX.

EmptyViews will be used to accommodate the cases where the content providers return no data or when there is no internet connections.

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

Picasso will be used to handle loading and caching of images.

Next Steps: Required Tasks

This is the section where you can take the main features of your app (declared above) and decompose them into tangible technical tasks that you can complete incrementally until you have a finished app.

Task 1: Project Setup

Write out the steps you will take to setup and/or configure this project. See previous implementation guides for an example.

- Configure libraries
- Decide on data representations
- getting development keys from websites to query data from.

Task 2: Implement UI for Each Activity and Fragment

List the subtasks. For example:

- Build UI for MainActivity
- Build UI for the detail screens
- Build UI for the points of interests(this include implementing maps services and)

Task 3: Make the Application work for bigger screens.

- Create layout for bigger screens.

Task 4: Polish the Application

- include motion and animations.
- handling errors.

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**”