Kacyn Fujii
Android Developer Nanodegree
Capstone Project Proposal
12 December 2015

- Description
- Intended User
- Features
- User Interface Mocks
  - Screen 1
  - Screen 2
- Key Considerations
    - How will your app handle data persistence?
    - Describe any corner cases in the UX.
    - Describe any libraries you'll be using and share your reasoning for including them.
- Next Steps: Required Tasks
  - Task 1: API Setup
  - Task 2: Implement UI for Each Activity and Fragment
  - Task 3: Database/Content Provider Setup
  - Task 4: Adding Google Maps/Location
  - Task 5: Customize Map Markers
  - Task 6: Implement Arrival Notification
  - Task 7: Handling Error Cases

GitHub Username: kacyn

Caltrain Plus

Description

The Caltrain Plus app will give users real-time schedule information and use location-based services to notify the when the user's destination is approaching. This will be useful for Caltrain riders who want updates on current train schedules before boarding the train. Caltrain riders will be able to relax, sleep, or work and the app will notify them when it's time to get off the train.

Intended User

This app is intended for Caltrain users who would like to get real-time Caltrain updates and useful notifications based on their current location.

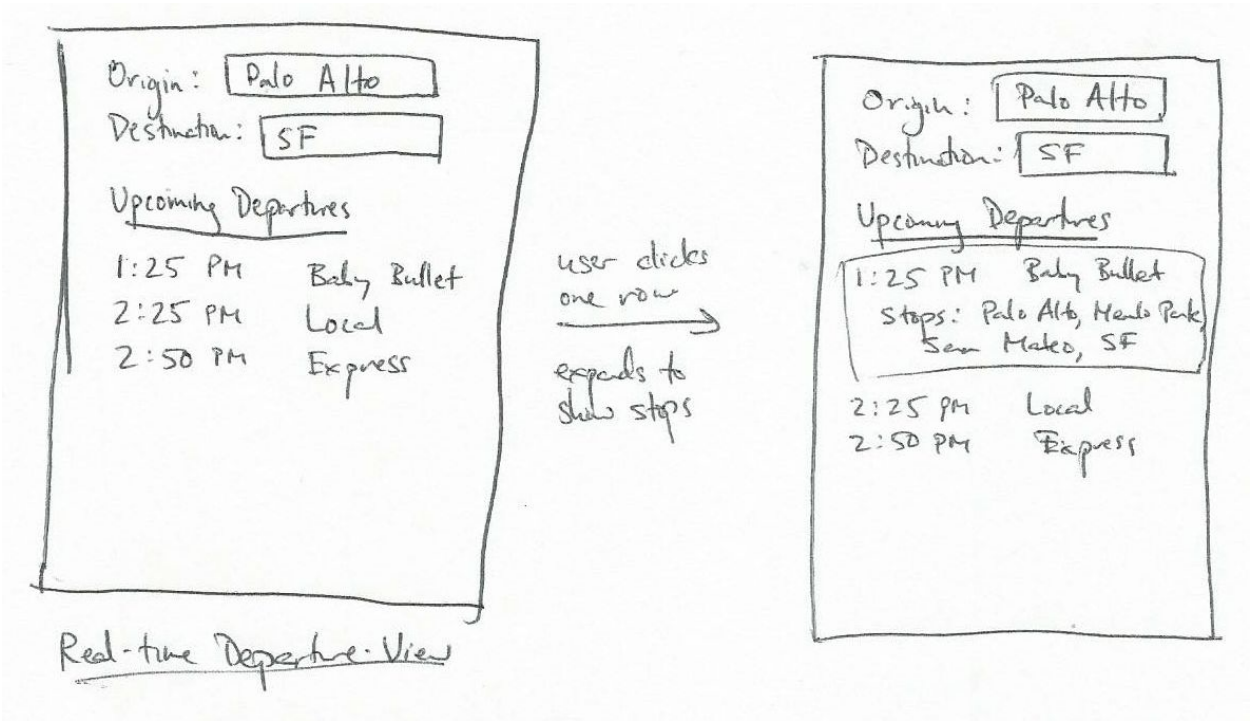
Features

The app will contain the following features:

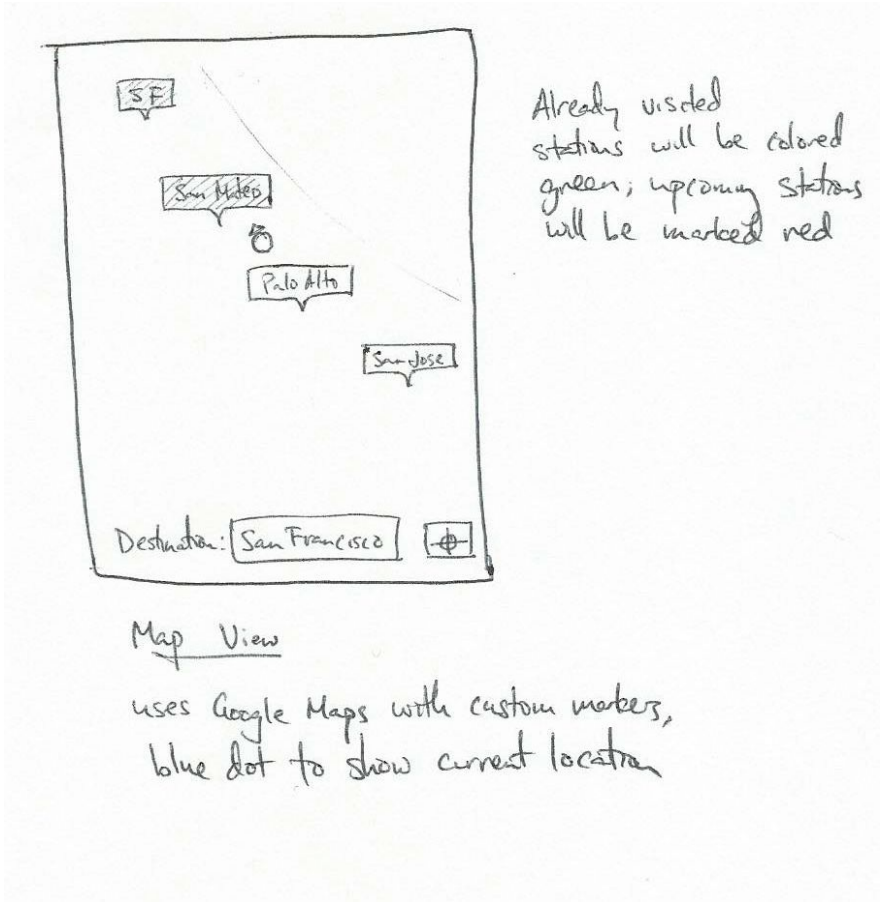
- Caltrain schedule information
- Map view showing stations and user's current location
- Notification when destination approaching

User Interface Mocks

Screen 1



Screen 2



Key Considerations

How will your app handle data persistence?

I will store information on each of the Caltrain stations using a SQLite database and access the data using a custom content provider. I will need to store information about what stops are included for each route (baby bullet, limited, local) as well as the stop name/codes for the API calls. I will likely also store timetable information and update this information when there has been an update to the real-time API.

Describe any corner cases in the UX.

If no real time information is available, I will display the normal train schedule and notify the user. The user will be able to switch between the map view and real-time departures view by swiping left or right.

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

I will use the Google Maps Android API Utility Library to make custom markers for each of the stations. The markers will contain the station name and a color indicating whether or not the user has already visited the station.

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: API Setup

Set up access 511 Real Time Transit API (http://assets.511.org/pdf/RTT%20API%20V2.0%20Reference.pdf)

Subtasks:

- Sign up for API token
- Configure OkHttp library to make calls to API
- Implement parsing of XML response
- Store relevant information in database

Task 2: Implement UI for Each Activity and Fragment

Build UI for activities/fragments

- Build UI layout for Map Activity
- Build UI layout for Upcoming Departures Activity

Task 3: Database/Content Provider Setup

Add use of database/content provider for efficient data management

Subtasks:

- Store information from API response into database
- Implement custom content provider to access data
- Add loaders to read data

Task 4: Adding Google Maps/Location

Add use of Google Maps/Location for map view and arrival notification

Subtasks:

- Get API key from Google Developer Console
- Add map view
- Add my location button

Task 5: Customize Map Markers

Customize map markers for each station

Subtasks:

- Modify markers to show:
    - Station name
    - Approximate arrival time at station
    - Update color if I've passed the station

Task 6: Implement Arrival Notification

Send notification when user near destination

Subtasks:

- Use geocaching to determine user proximity to destination
- Build notification and send when user close to station

Task 7: Handling Error Cases

Handle the following error cases:

- No network access
- Real time updates not received
- App persists information after state changes