How to Use this Template

1. Make a copy [File → Make a copy...]
2. Rename this file: "**Capstone_Stage1**"
3. Replace the text in green

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**"

GitHub Username: nkt1001

Alarmiko

Description

Alarmiko is an incredible app that helps you never miss the bus stop or the train station. Also the app will be very helpful to don't forget to visit a shop, dry-cleaning or tavern on your way back home!

Intended User

Tired students who always sleep in the public transport, travelers who visually don't know the exact place of required station, forgetful people who always forget to buy food home.

Features

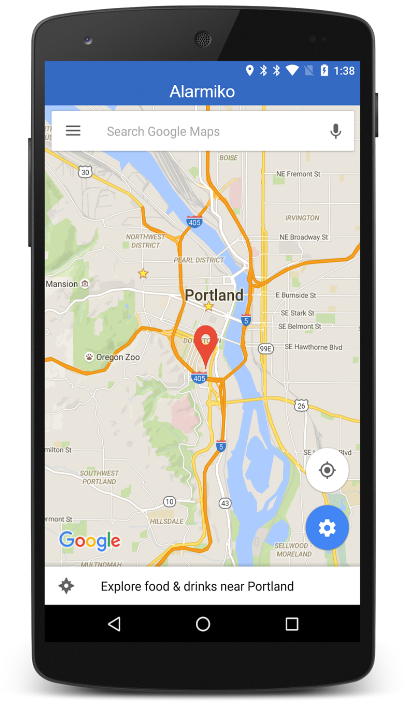
List the main features of your app. For example:

- Saves information
- Google maps
- Alarm

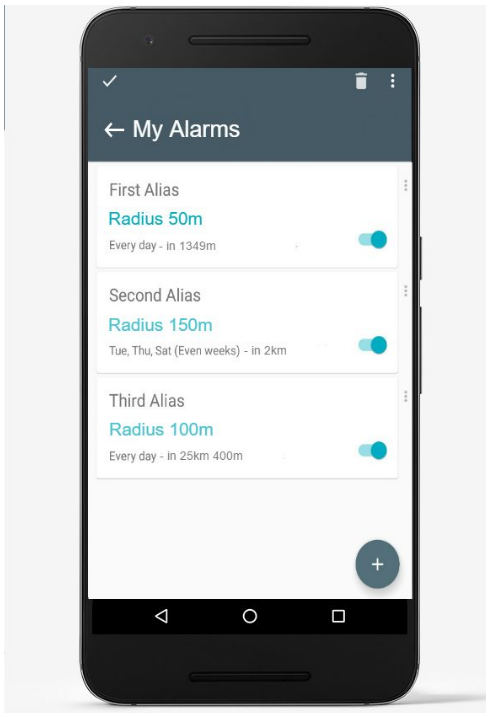
User Interface Mocks

These can be created by hand (take a photo of your drawings and insert them in this flow), or using a program like Photoshop or Balsamiq.

Screen 1



Screen 2



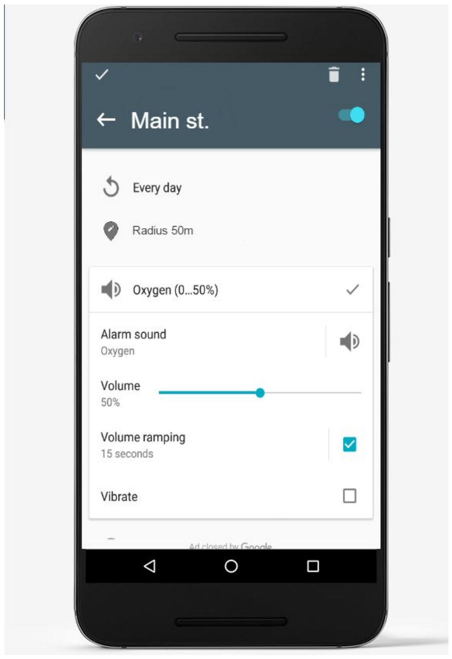
My Alarms

First Alias
Radius 50m
Every day - in 1349m

Second Alias
Radius 150m
Tue, Thu, Sat (Even weeks) - in 2km

Third Alias
Radius 100m
Every day - in 25km 400m

Screen 3



Main st.

Every day

Radius 50m

Oxygen (0...50%)

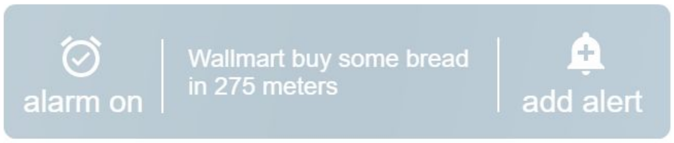
Alarm sound
Oxygen

Volume
50%

Volume ramping
15 seconds

Vibrate

Widget



Key Considerations

How will your app handle data persistence?

SQLite Database will save all user's alarms. Content provider with loaders manage SQLite database.

Describe any corner cases in the UX.

User always gets to the screen 1 if the app was closed. Screen 1 is the parent screen for Screen 2 and Screen 2 is the parent for Screen 3. After hitting back button user getting to the parent screen

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

Material design library to implement material style, stetho for database debugging, canary leak to find memory leaks, butter knife for fast view binding, fabric will be tracking my app growing and collecting errors, GCM for sending push notification over my server

Describe how you will implement Google Play Services.

Google Maps with Google places will be implemented on the map screen. Google location is implemented in background service. Admob is used for app monetization

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.

You may want to list the subtasks. For example:

- Configure libraries
- Create google console account
- Create admob account
- Add permissions and create permission activity
- Create git repository
- Connect to git
- Configure git ignore
- Configure fabric

If it helps, imagine you are describing these tasks to a friend who wants to follow along and build this app with you.

Task 2: Implement UI for Each Activity and Fragment

- Build UI for MapActivity
- Build UI for AlarmListActivity
- Build UI for AlarmDetailActivity
- Build UI for AlarmListFragment
- Build UI for AlarmDetailFragment
- Build layout for AlarmListItem
- Build widget layout

Task 3: Implement google play services

- Add Google Maps fragment
- Add Google Places picker
- Put small banner to the bottom of Activities
- Place Interstitial ads between activity transitions

Task 3: Implement database interface

- Create database table contract and db helper
- Create and configure ContentProvider

Task 4: **Create MapActivity**

- Create Google maps activity
- Add Google place picker
- Implement
- Set custom marker
- Implement LoaderManager interface

Task 5: **Create AlarmListActivity**

- Create layout for activity
- Create layout for fragment with RecyclerView
- Create AlarmListItem layout
- Implement LoaderManager interface in fragment
- Create RecyclerViewAdapter
- Implement distance displaying to all items from current location

Task 6: **Create AlarmDetailActivity**

- Create layout for activity
- Create layout for fragment with RecyclerView
- Implement LoaderManager interface in fragment

Task 7: **Create AlarmSerivice**

- Connect to google api client
- Load all alarms with LoaderManager
- Implement tracking user location
- Matching current location with alarm location
- Create notification with desired configurations when locations are matched

Task 8: **Create Wigdet**

- Create layout for widget
- Add widget to manifest
- Create app wigdet subclass with my widget

Task 9: Implement GCM

- Get google json from google console
- Add information to manifest
- Implement IntentService subclass sending tokenId to endpoint
- Configure notification