How to Use this Template

1. Create a new document, and copy and paste the text from this template into your new document [Select All → Copy → Paste into new document]
2. Name your document file: "**Capstone_Stage1**"
3. Replace the text in green

GitHub Username: send2deb

tinru

Description

tinru is a travel app (the name means **T**ravel**IN**g, a**R**e yo**U**?), the app allows users to find interesting places of popular cities of the world. The app also has location awareness features which help user to find out the nearest restaurant or café or bar or petrol pump or airports, etc. The app also helps user to find out the cheapest flight option to reach the desired location.

Intended User

Travelers

Features

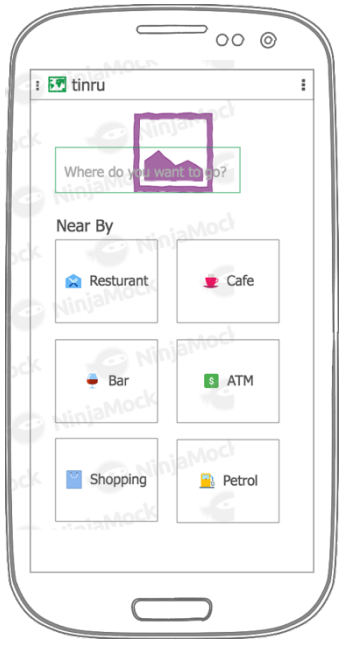
The main features of the app are below:

- Show points of interest of city / location
- Show nearest restaurant, café, bar, atm, petrol pump, etc
- Show cheapest flight option to reach a destination
- Uber availability (Aspirational / feature good to have – not covered in this document)
- Hotel Search (Aspirational / feature good to have – not covered in this document)

User Interface Mocks

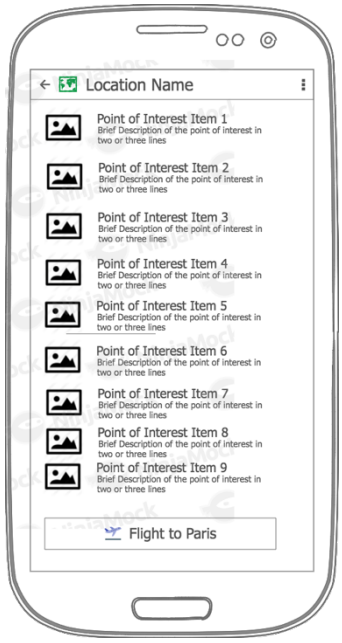
These can be created by hand (take a photo of your drawings and insert them in this flow), or using a program like Google Drawings, www.ninjamock.com, Paper by 53, Photoshop or Balsamiq.

Screen 1



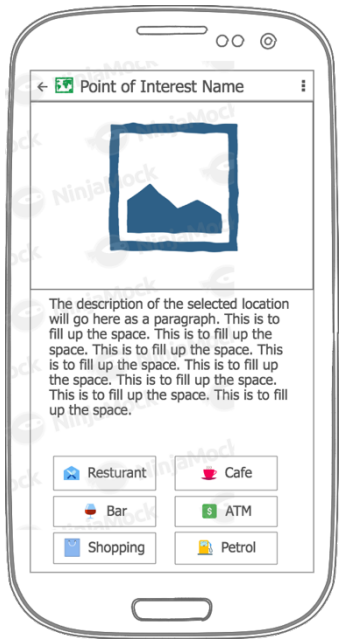
This is the Home Page of the app. There will a background image at the top and inside it a search box to enter a location name. Nearby attractions icons will be populated with relevant categories.

Screen 2



This screen will show the Points of Interest for given location as list. Clicking on the list item will open the following detail page. There will also be a button to display Flight options.

Screen 3

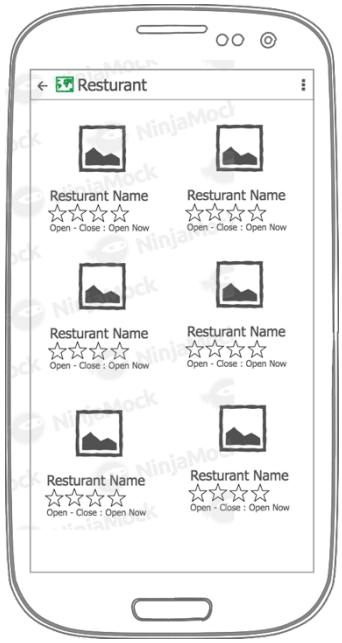


Display the details of the selected Point of Interest. It also shows the Near By categories of searched location.

Screen 4

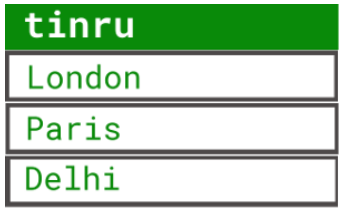
Displays the lowest Flight options from user location to searched location. The list is sorted on the price.

Screen 5



This page displays the content of selected Near By category in grid format. Clicking on this item will open the Google map.

Screen 6 (App Widget)



Add widget to the app to display last searched locations.

Key Considerations

How will your app handle data persistence?

The app will use RESTful api of amadeus travel innovation sandbox and Google places. All data will be retrieved real time in JSON format. Only the user's search location will be stored in database. Google's new Room data persistence will be used to store data.

Describe any edge or corner cases in the UX.

The app will use the external app Google map, on pressing the back button of Google map the user will be taken back to app.

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

Picasso to load any images.
Retrofit to consume RESTful apis.
Dagger2 for dependency injection.
Butter Knife for view binding.
Timber for logging.

Describe how you will implement Google Play Services or other external services.

Google Map Platform (https://cloud.google.com/maps-platform/)
The app will use Google Map platform for location awareness content. Refer this page (https://developers.google.com/places/android-sdk/start) to know as how to use places and this page (https://developers.google.com/maps/documentation/android-sdk/start) for map api.

Google AdMob(https://developers.google.com/admob/android/quick-start)
The app will use as the second Google Play Service api to show test add.

amadeus Travel Innovation Sandbox (https://sandbox.amadeus.com/)
This sandbox is provided by amadeus to test and prototype to use their largest repositories of travel data. They provide api data for various travel services but this app will use the Points of Interest and Flight search apis only. Refer the api documentation here https://sandbox.amadeus.com/api-catalog

Next Steps: Required Tasks

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

Task 1: Project Setup

Perform the following tasks to configure the project.

- Add Picasso library to app's build.gradle file, refer this page for details -> http://square.github.io/picasso/
- Add Dagger2 library to app's build.gradle file, refer this page for details -> https://google.github.io/dagger/android.html
- Add Retrofit to app's build.gradle file, refer this page for details -> https://github.com/square/retrofit
- Add Butter Knife to app's build.gradle file, refer this page for details -> http://jakewharton.github.io/butterknife/
- Add Timber to app's build.gradle file, refer this page for details -> https://github.com/JakeWharton/timber
- Use only the stable version of the libraries / dependencies

After adding and configuring all the libraries / dependencies, run Gradle sync to complete the setup / configuration.

App should also adhere to the following guidelines:

- App will use Java language for development
- App will use ONLY the stable versions of all libraries
- All String variables are defined in string.xml file
- All dimen values are defined in dimen.xml
- Enable RTL layout switching

App will be built using the Gradle version 4.4.
App will be built with Android Studio version 3.1.3

Task 2: Implement UI for Each Activity and Fragment

- Build UI for Home Page (i.e. Main Activity)
    - Add one editable text filed which will be used as search box
    - Show icons for location awareness content of device / user location (Refer the UI mock). Use image button to display icons.
    - Use a background image at the top
- Build UI for Points of Interest list for a city / location
    - Show Points of Interest as list
    - Show a button at the bottom of the page for Lowest fare Flight Search
- Build UI for Point of Interest Detail
    - Show Point of Interest image at the top
    - Show details of the Point of Interest
    - Show icons for location awareness content of searched location (Refer the UI mock). Use image button to display icons.
- Build UI for Lowest Fare Flight Search
    - Show lowest fare flight (from user current city/location to destination city / location) options as list
- Build UI for nearby content
    - Use cards to display nearby contents as grid
    - Navigate go to Google map when the item is clicked

Task 3: Setup libraries

Dagger2 Setup

- Configure Dagger2 by following the steps mentioned in this document https://google.github.io/dagger/users-guide
- The scholarship project 'Baking App' (https://github.com/send2deb/BakingApp) can be referred also as it uses Dagger2

Retrofit Setup

- Configure Retrofit by following the steps mentioned in this document https://square.github.io/retrofit/
- The scholarship project 'Baking App' (https://github.com/send2deb/BakingApp) can be referred also as it uses Retrofit
- Use the site http://www.jsonschema2pojo.org/ to create POJO from JSON for Retrofit

Create a custom Application and add all application level configuration at a single place. Refer the custom Application of 'Baking App' here https://github.com/send2deb/BakingApp/blob/master/app/src/main/java/com/debdroid/bakingapp/BakingApplication.java

Dependencies of libraries are as below (all stable versions):

```
//Retrofit related dependencies
implementation 'com.google.code.gson:gson:2.8.2'
implementation 'com.squareup.retrofit2:retrofit:2.4.0'
implementation 'com.squareup.retrofit2:converter-gson:2.4.0'
implementation 'com.squareup.okhttp3:logging-interceptor:3.8.0'

//Dependencies for Timber logging
implementation 'com.jakewharton.timber:timber:4.7.1'

//Dagger2 dependencies
implementation 'com.google.dagger:dagger:2.16'
// Annotation processor for dagger
annotationProcessor 'com.google.dagger:dagger-compiler:2.16'

//Picasso dependencies
implementation 'com.squareup.picasso:picasso:2.71828'

//Butterknife dependencies
implementation 'com.jakewharton:butterknife:8.8.1'
annotationProcessor 'com.jakewharton:butterknife-compiler:8.8.1'

//Facebook Stetho dependencies
implementation 'com.facebook.stetho:stetho:1.5.0'
implementation 'com.facebook.stetho:stetho-okhttp3:1.5.0'
```

Task 4: Create layouts and build modules

App will pull the data from apis on per request basis and all REST api call will be made asynchronously using AsynTask.

- Create all the layouts as per UI mocks
    - Use Coordinator layout wherever possible
    - Use Material design guidelines
    - Use RecyclerView and CardView for list and grid items respectively

- Home Screen Activity
    - Use Google Places autocomplete and place picker features for user search option and allow user to select a location
    - Use Card as grids to display icons for location awareness content
    - Store the user selected location to database

- Points of Interest List Activity
    - Fetch data from Google based on user selected location
    - Use the Geo location to fetch Points of Interest from amadeus api and show to user as list (use RecyclerView)
    - Show buttons at the bottom for Flight Search
    - On clicking the list item, open the Point of Interest detail Activity
    - On clicking Flight Search navigate to Flight Search List activity

- Points of Interest Detail Activity
    - Fetch data from amadeus api
    - Show the image and description of the Point of Interest
    - Show location awareness content categories at the bottom of the activity
    - On clicking location awareness content item, open Near By content list

- Lowest Fare Flight Activity
    - Fetch data from amadeus for lowest flight option between current location and destination location
    - Display the data as list to the user (use RecyclerView)

- Location awareness Near By Activity
    - Use Google Place api to fetch the data of the selected option and display as list (use RecyclerView and Card as single grid)
    - Allow user to navigate to Google map app

Task 5: Add Widget

- Add a Widget feature to the app
- The widget should show the last search locations of the user
- On clicking the widget the app should open and display the content for the location clicked on the Widget

Task 6: Accessibility Support

- Add content description for applicable components
- Add navigation using D-pad

Task 6: Error Handling

- Display meaningful messages if data cannot be displayed for any reason
- Display a message if the device is not connected to internet
- Handle all other errors and exit conditions / scenarios gracefully

Add as many tasks as you need to complete your app.

Submission Instructions

- After you've completed all the sections, download this document as a PDF [File → Download as PDF]
    - Make sure the PDF is named "**Capstone_Stage1.pdf**"
- Submit the PDF as a zip or in a GitHub project repo using the project submission portal

If using GitHub:

- Create a new GitHub repo for the capstone. Name it "**Capstone Project**"
- Add this document to your repo. Make sure it's named "**Capstone_Stage1.pdf**"