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: jayc1299

New York Travel Guide

Description

New York Travel guide is a simple tourist attraction app that aims to help guide you around some of the highlights of New York city. It provides a list and a map of some of the key sites to visit around New York City.

User's can read reviews of the attractions and even add new reviews of their own. The app allows you to add favourite attractions so they can be quickly found.

The app will be written entirely in Java.

Intended User

This app is for anyone visiting NYC. It will help people find the key sites to visit and a central place to keep track of the things they need to know.

Features

- List of key NYC attractions
- Map view of NYC attractions
- Ability to add favourites to be saved for later
- Ability to share an attraction
- Ability to rate favourite locations

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.

https://www.ninjamock.com/s/8QBSXTx

Screen 1 - Attraction list

This is the main page of the app. This shows a list of attractions with an image, a title and a short extract of the description.

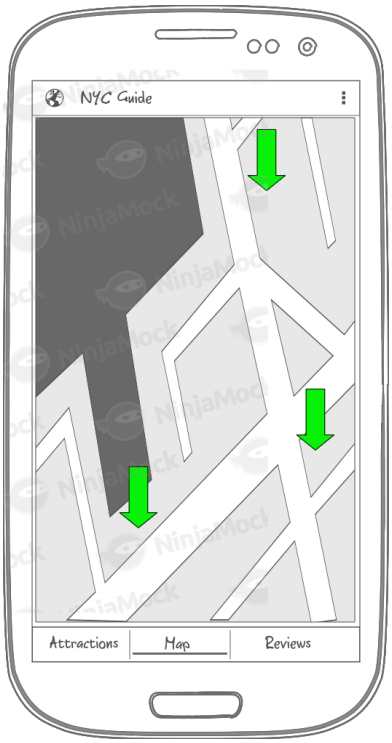
The bottom of the page shows a three tabs:

- Attractions
- Maps
- Reviews

Clicking an attraction links to (screen 4) the attraction details page in a new Activity.

The top right kebab menu has only settings available. Clicking settings opens the setting page.

Screen 2 - Map



This page lists the attractions on a map of new york.

Clicking a pin opens a popup window. Clicking the window opens the attraction detail page (screen 4).

The map is scrollable and centers on New York.

The map will show the user's current location if the user wants it to. Configurable in settings and if location permission is granted.

Screen 3 - Reviews

The reviews screen lists all reviews for a specific attraction. The desired attraction is selected in the spinner at the top of the screen.

There is a FAB plus button which will allow a user to add a new Review.

Each review in the list shows stars, title and a short extract of the content. Title and content will be truncated to fit.

Screen 4 - Settings

This is the settings page where the user can configure various settings and options in use around the app.

Clicking back closes the settings page.

Settings is accessed from the kebab menu on all main tabbed pages.

Settings include:

- Show pins on map
- Show user location on map
- Show only favourites

Screen 5 - Attraction Detail

This page shows the detail of the attraction.

At the top is a photo of the attraction, this will scroll out of the frame as the content is scrolled up.

The page has a heart which will allow the user to toggle favourite on and off.
The page has a map button which will link to the map page and the map pin will be located.
The page has a read reviews button which will link to the main tabbed activity opening the reviews tab. When clicking this link the current attraction will be auto selected in the spinner.

The page also has a set of five stars which are highlighted according to the review score for this attraction.

The detail page will also allow you to share an attraction on social media.

Screen 6 - Attraction List Favourites

This page lists all attractions but filters to show only favourite attractions.

As with the normal attractions page it shows images, titles and short descriptions. On clicking it opens the attraction detail page.

Screen 8 - Review Detail

This is a review detail page. It will list the full details of the review. Including the title and content of the review.

It will show the number of stars the user gave the attraction.

It will show the full title the user gave.

The full user description will also be visible. The page will scroll if the content requires it.

Screen 9 - Add Review

This is what happens if you click the add review FAB button in the review list. It features five stars to rate the attraction. It also features a title and a multi line text input for the content.

On pressing the submit button the review will be saved and the page will close.

These input fields will all use TextInputLayout. The details input will be multiline.

Screen 10 - Widget

This is a widget that is required for the app. It will show a list of favourite attractions. Clicking one will open the app at the attraction detail page.

Key Considerations

Language

The app will be written entirely in java programming language.

How will your app handle data persistence?

The app will use Firebase Realtime Database

Describe any edge or corner cases in the UX.

The three tabbed pages are the main pages and form the cornerstone of the navigation. All other pages will have a back button in the top left and will close on clicking putting the user back on the main tabbed activity.

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

I will need:

- Firebase Core (16.0.3)
- Firebase Realtime Database (16.0.1)
- Picasso for images (2.71828)
- Design library (27.1.1)
- CardView (27.1.1)
- Support library (27.1.1)
- AppCompat (27.1.1)
- RecyclerView (27.1.1)
- Google play services (15.0.1)
- Google play services maps (15.0.1)
- Google play services location (15.0.1)
- Android Studio 3.0 or greater
- Gradle (4.4)
- Gradle tools (3.1.4)
- Google services (4.0.1)

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

I will need to use Google Play Services to present the map page. This page will show pins for each attraction. It will also use Firebase for realtime database.

All attractions and all reviews will be stored in Firebase realtime database. The data structure will need to be carefully considered to ensure it can be read easily on the details page as well as other pages.

Resources

Wherever possible the app should make use of strings.xml, colors.xml & dimensions.xml. This will make the app easier to translate into other languages.

Accessibility

- The app will not support RTL at this time.
- All images will use a contentDescription for accessibility.
- All editText / TextInputLayout fields will use hints.
- Buttons should be large enough and clearly described.

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

- Create a new Android project in Android Studio. Include one new Activity which will be the main tab activity.
- Get rid of all the Constraint Layout nonsense from the main activity.
- Ensure gradle and the target sdk are all up to date.
- Add the required libraries
- Create Strings.xml, dimensions.xml and colors.xml files in the /res/values folder. You should put all strings, dimensions and colors used in this project in these folders.

Task 2: Implement UI for Each Activity and Fragment

- Implement a tabLayout in the main activity. Setup three separate tabs, Attractions, Maps and Reviews. Ensure they are all centered and share the bottom navigation bar appropriately.

- Create one fragment for each tab and allow clicking on the tab to swap between the three fragments in the top view of the activity. For now the Fragments can just show their title.
- Edit the attractions fragment and give it a recyclerview, adapter and an item layout. Use hard coded fake data to show the list of items. Add picasso library and use it to display images in the recyclerView.
- Add the maps library and set it up to show a map in the maps fragment. Do not add any pins at this point.
- Edit the reviews fragment and create a recyclerView and an adapter. Use mock reviews at this point. Create the spinner, preferably using dialog mode and give it some mock data. Create a listener for the spinner and update the adapter based on the spinner selection.
- Create a new activity for the attraction detail. Ensure in the manifest it is set to have a parent of the main activity. Implement the image and utilise CoordinatorLayout. Add the content and the favourites button. Add the map, reviews and stars to this page and ensure they are laid out well.
- Create the settings page using PreferenceActivity. Implement the desired settings and ensure they are saved in shared preferences.
- Add an FAB and a new review activity. Ensure all form elements use TextInputLayout for their input.

Task 3: Implement Firebase Realtime Database

We will use firebase for our backend work. This will involve implementing proper procedure for retrieving attractions and reviews from the firebase. We'll need to add firebase to the project and link the app. We'll also need to make sure the data is created in the firebase realtime database.

Once firebase has been setup implement listeners for the data and connect the data to the attractions list, the map pins and the reviews.

Task 4: Setup the Pins on the Map page

The map should show pins for every attraction the app shows. Each pin will be correctly located at the GPS for that attraction. Clicking on a pin will open a small popup window with the attraction name. Clicking on the popup will link to the attraction detail page.
Pins will be read from firebase.

Task 5: Create the widget

This widget will show a list of favourite attractions. Consideration needs to be given to the udacity section of widgets where list widgets are created. The list will show all favourite attractions. This list will need to be grabbed from firebase.

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**"