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

Description
Intended User
Features
User Interface Mocks
    Screen 1-2
    Screen 3-4
    Screen 5
    Screen 6
    Screen 7
    Screen 8
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.
        Describe how you will implement Google Play Services.
Next Steps: Required Tasks
    Task 1: Project Setup
    Task 2: Implement UI for Each Activity and Fragment
    Task 3: Implement First Run Case
    Task 4: Implement Settings Activity
    Task 5: Setup Firebase, Firestore and Google Auth
    Task 6: Text-to-speech configuration
    Task 7: Google Ads integration
    Task 8: Setup Paging Library logic to list more items
    Task 9: Setup Room database with Pagination library
    Task 10: Prepare AppWidget to list favorite countries
    Task 11: Integrate ViewModel and ViewModelFactory

GitHub Username: nuhkoca

Trippo

Description

Trippo is your smart travel guide - it’s free and works offline, too! You can skim through tours, locations, POIs, city walks, articles and so on. This guide will save you from a serious headache!

Intended User

Especially for travelers or for those who love travelling!

Features

List the main features of your app. For example:

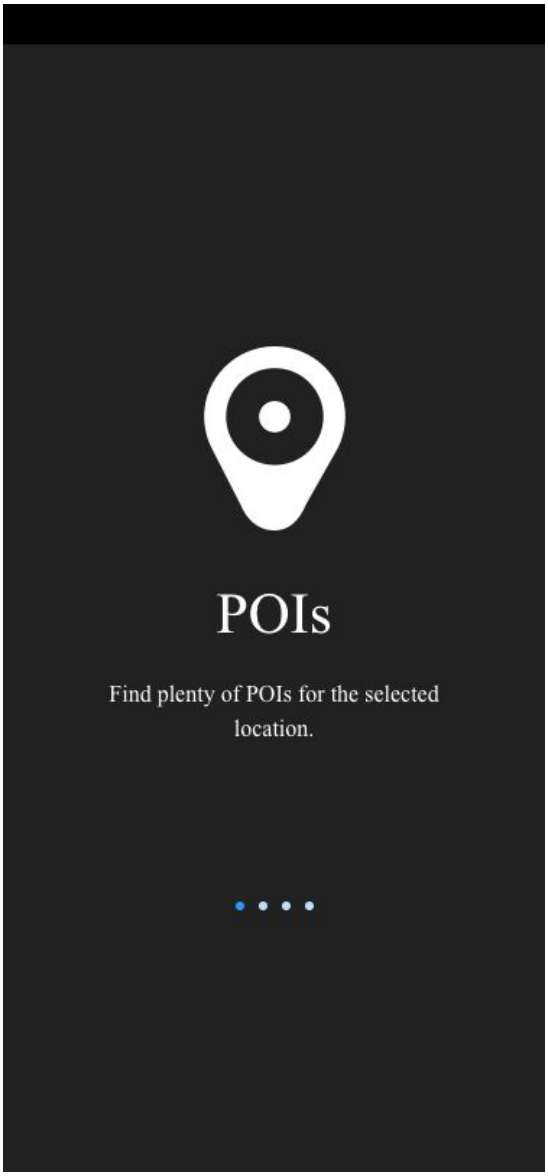
- Shows offline data when there is no internet(favorite countries)
- Sends regular notifications about landmarks, places etc
- Search online with preferences
- Search fetched data by Search button
- View restaurants, hotels, night clubs, etc of a city.

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 1-2

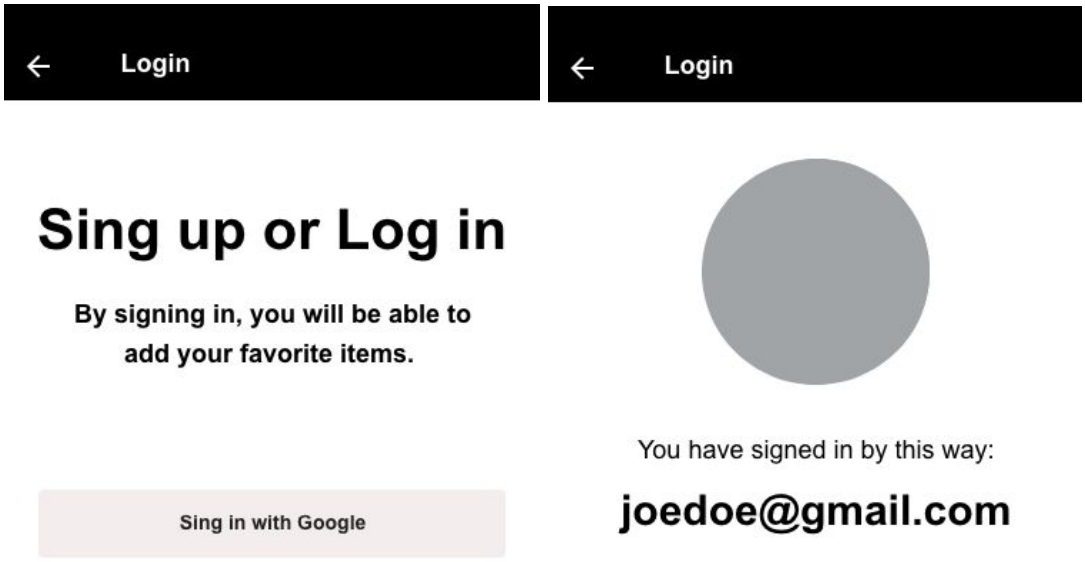
1.Splash Screen



2.Onboarding Screen

At first, these two screen will welcome users. If the app runs for the first time, Onboarding screen will appear otherwise Splash Screen will appear for each run. At present, each app in the market has such screen and I wouldn't develop my app without these ones!

Screen 3 - 4



Skip Now

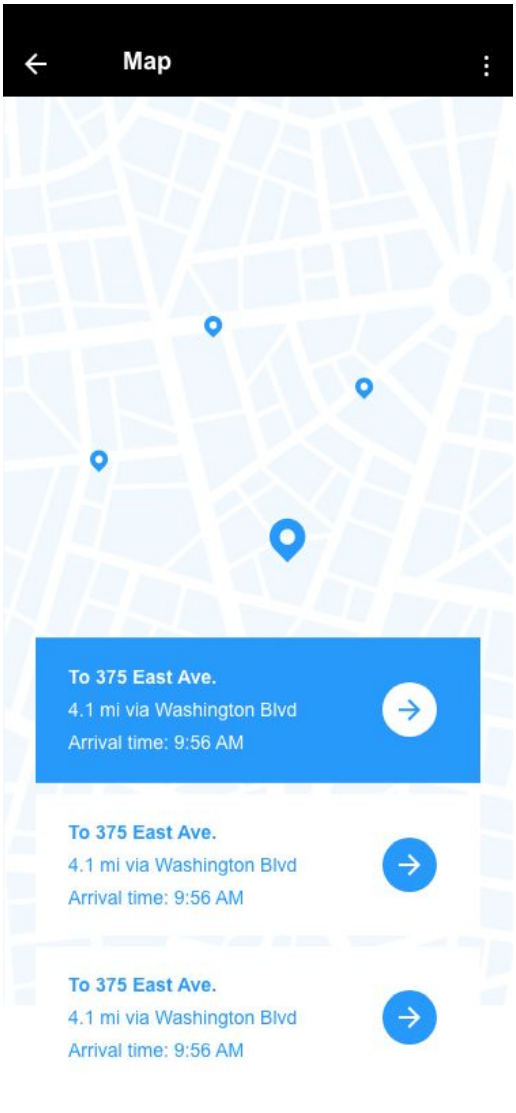
1.Sign-in Screen

Log out

2.Logout Screen

To allow only authorised users to add their favorite items or delete some files for application, they should sing-in. They can log-out whenever they want, as well.

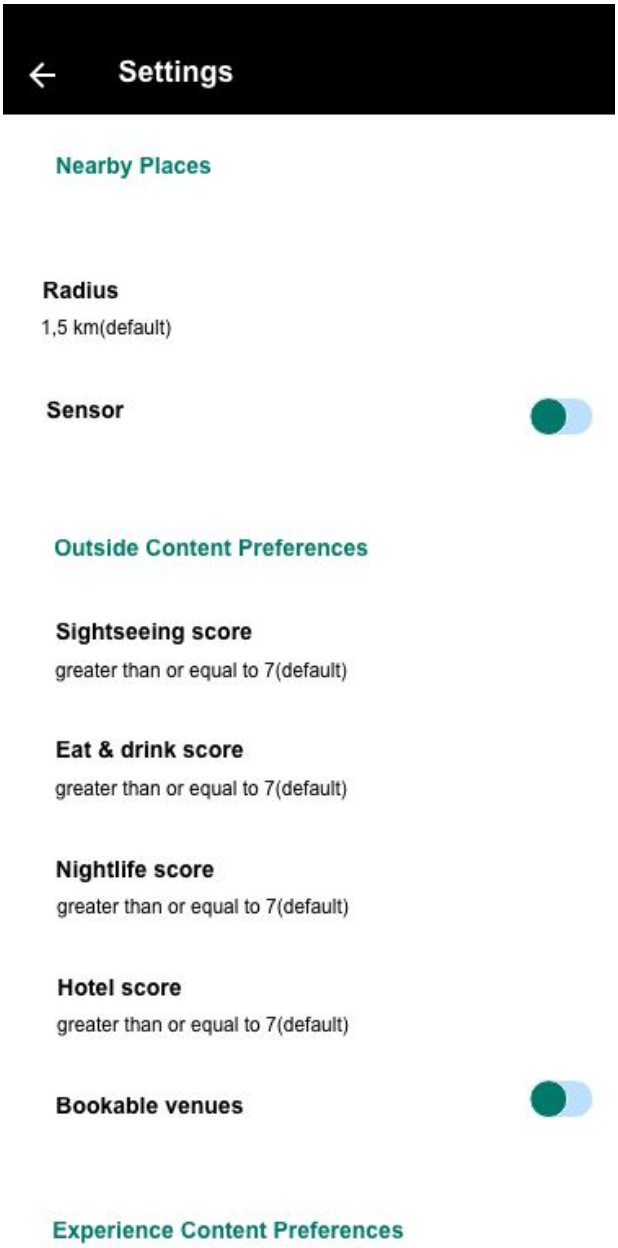
Screen 5



5.Map Screen **5.Nearby Places Screen**

Map Activity allows users to find their current location or nearby places around them. Users also can change radius for nearby places in SettingsActivity.

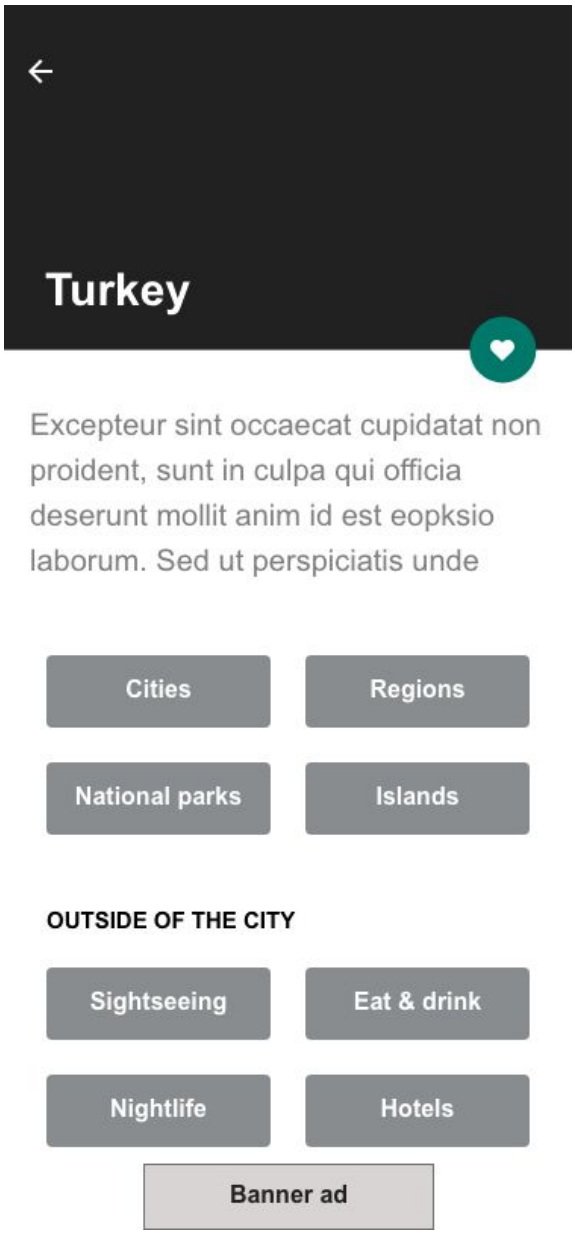
Screen 6



6.Settings Screen

In Settings Activity, there are plenty of settings for different screens. For instance, user can select different radius to have a wider area for nearby places or they can specify beginning scores for POI items or switch on/off notifications.

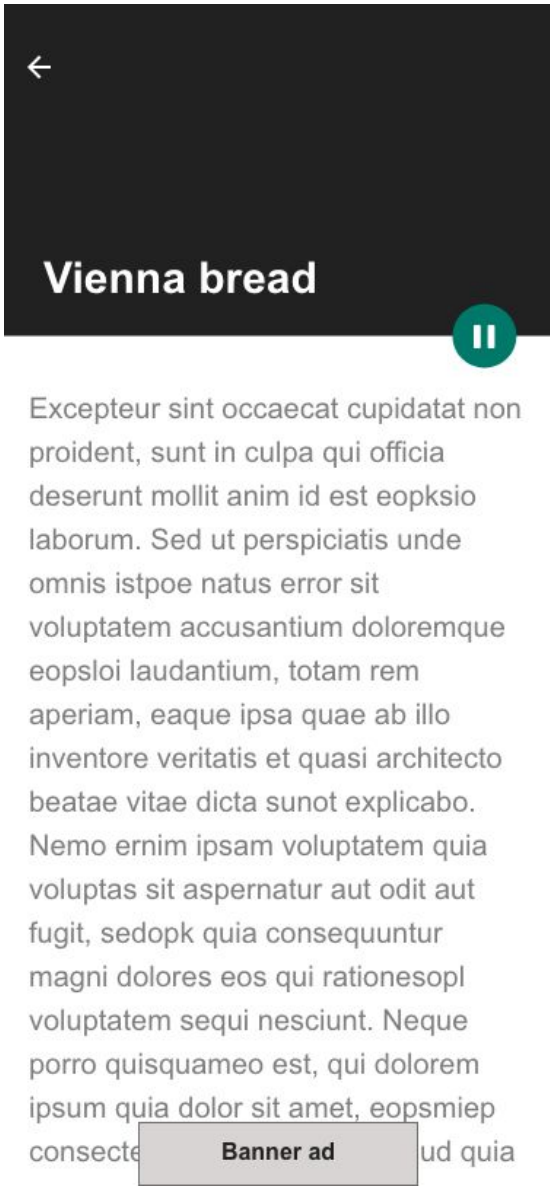
Screen 7



7.Country Detail Screen

In Country Detail Activity, user can view different kind of contents that selected country has like its cities, islands or sightseeing, hotels or tours, day trips and so on. In this activity, there is a banner ad at the bottom of the page, too. When users click on favorite button, country gets added to database. On the second touch, it is deleted from database.

Screen 8



8.Article Detail Screen

In Article Detail Activity, users can view country exclusive backgrounds or practicalities like food, culture, celebrations, visa information, electricity plugs or groceries. When users clicks on a result of any kind counted above screen pops up. Since texts place there are too long in terms of reading, this activity has its feature that it can ready the entire article for users. When you click on play button it starts reading and second touch stops it.

Key Considerations

How will your app handle data persistence?

This app uses Room component to store favorite countries. Even if there is no active connection, users can view this list and access to country detail page.

Describe any edge or corner cases in the UX.

This app has a Bottom Sheet for may map activity. Users can search for supermarkets, cafes, gas stations, ATMs and groceries by expanding this sheet.
In addition, this app benefits from Text-to-speech feature that allows app to speak out articles for backgrounds or practicalities of counties. If user clicks on Floating Action Button, app speaks out or clicks on it again, app keeps quiet. With the back button, app keeps quiet again.

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

Glide for handling images - the best one, no need to explain why :)
Retrofit and **RxJava** for API fetch because these two propose developer a very effective data consuming.

Timber for logging, the best logging library, completely sterilized from boilerplates.

Room Persistence Library - more effective and best way to store offline data.

Paging Library - To load more items for those have pagination web service. Just listing 20 items not enough. This is why apps need pagination and I used it.

Text-to-speech - To read articles. Articles are too long texts so that reading them takes time. This is because app reads it to you itself.

Firebase - Firestore and Google Auth - Firebase to send push notifications, Firestore to store some important data for app and Google Auth to allow users add their favorites by signing in.

ViewModel - ViewModelFactory - To have a live data over web service and to be against screen orientation

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

This app uses, Google Ads, Google Maps and Google Auth. The app has 2 ad style which are Banner and Interstitial. The app also shows nearby places by using Google Maps and it allows its users to sign in with Google Auth.

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

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:

- Find a good web service for the app
- Implement required dependencies

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

List the subtasks. For example:

- Implement UI for Splash Screen
- Implement Onboarding Activity for first run
- Implement Login Screen
- Implement Map Activity
- Implement Country List Activity
- Implement Favorites Activity
- Implement Country Detail Activity
- Implement Content Activity for cities, regions, islands and national parks
- Implement Content Activity for POIs of countries

- Implement Content Activity for tours, day trips, multi day tours, city walking and activities that each country has
- Implement Settings Activity

Task 3: Implement First Run Case

To detect if app runs for the first time, I need to introduce the app to users. This is why app runs Shared Preference in the background for that.

- Create **SharedPreferenceUtil** class
- Create the relevant method that return boolean for the first time run
- Show Onboarding Screen
- So it is first run, after Onboarding Screen, Login Screen is called. Users can skip it for next time

Task 4: Implement Settings Activity

This activity harbors plenty of settings for each activity like radius option for Map activity, or bookable option for POIs.

- Implement Preference dependency
- Create a Settings Activity with different options
- Implement **Preference Theme** to AppTheme.

Task 5: Setup Firebase, Firestore and Google Auth

To send push notifications, store important queries and allow users to sign in, this app runs different kind of logics in the background.

- Add the required dependencies
- Initialize **Firebase** to send push notifications
- Setup **Firestore** to keep token for push notifications
- Implement **Google Auth** to allow users to sign in or sign out.
- Don’t allow users to add their favorites unless they sign in.

Task 6: Text-to-speech configuration

In the Article Detail Activity, the app read articles since they are all too long to read. Along with this implementation, app will lend assistance to you.

- Create Article Detail Activity
- Add Floating Action Button to play or stop **TTS**
- Let the app speak out
- Finalize TTS if user goes back

Task 7: Google Ads integration

The app shows both banner and interstitial ads in different locations of the app.

- Add ad dependency
- Add **AdView** for banner ads
- Implement InterstitialAd for full-screen ad.

Task 8: Setup Paging Library logic to list more items

Since there are lots of items in responses, app needs Paging Library which is a part of Android Architecture Components.

- Add the necessary dependency
- Create required classes like **ItemKeyedDataSource** and **DataSourceFactory**
- Fetch items with pagination
- Customise RecyclerView according to that. For example show a **ProgressBar** when next items are being loaded.
- Extend ReyclerVewAdapter with **PagedListAdapter**.

Task 9: Setup Room database with Pagination Library

App uses Room to store favorite countries. In addition, as there are almost 225 countries, users can add all of them. To query 225 countries, Room also needs pagination.

- Add Room dependency

- Create entities, DAOs and database class
- List all favorite items by using PagedListAdapter and Room

Task 10: Prepare AppWidget to list favorite countries

App has an app widget that shows favorite countries in a short way. When users click on an item on the widget Country Detail Activity gets opened. When screen is empty, Main Activity gets opened.

- Create **AppWidget**
- Create **GridViewRemoteService**
- Shows favorite items on the widget
- Open relevant activity depending on action

Task 11: Integrate ViewModel and ViewModelFactory

The new Architecture Components are quite logical way for remote datum on Android. To stay up-to-date and to use recommended technologies, Trippo uses the latest technologies as always.

- Create **ViewModel** for each activity.
- Create **ViewModelFactory** to let ViewModels have their own constructors.

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**"