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: Pasupatirajesh (www.github.com/pasupatirajesh)

WorldOfAT

Description

The name of the app is World of AT. This app is a prototype to showcase some of the aspects of Assistive Technology and its use in helping people with disabilities. The app displays a lot of images of AT solutions that are sourced from Getty Images.

The App also gives a brief overview of what AT is and the UIC Assistive Technology Unit's Website and Personnel. The app has a mock built in job search feature. The app is purely to showcase implementation of various concepts I learnt in the Nano-degree program.

Intended User

Any one who wishes to know about Assistive Technology

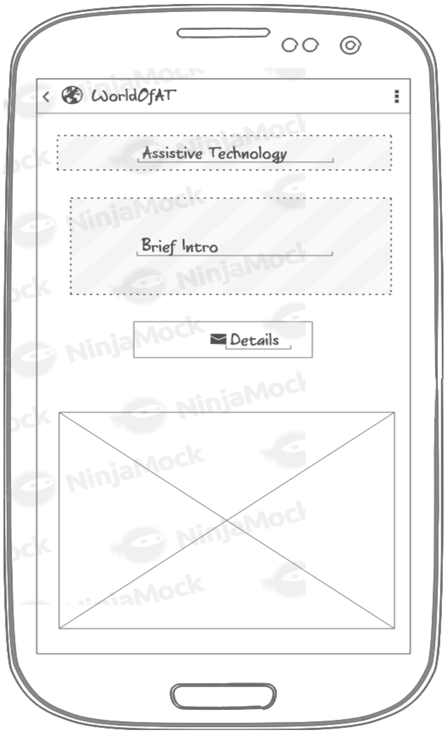
Features

List the main features of your app. For example:

- Information about AT, Displaying custom AT solutions sourced from Getty Images
- Custom Job search focussed on AT jobs (Curated list of jobs - As I couldn't get access to proprietary data from job listing services)
- Supports PIP

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

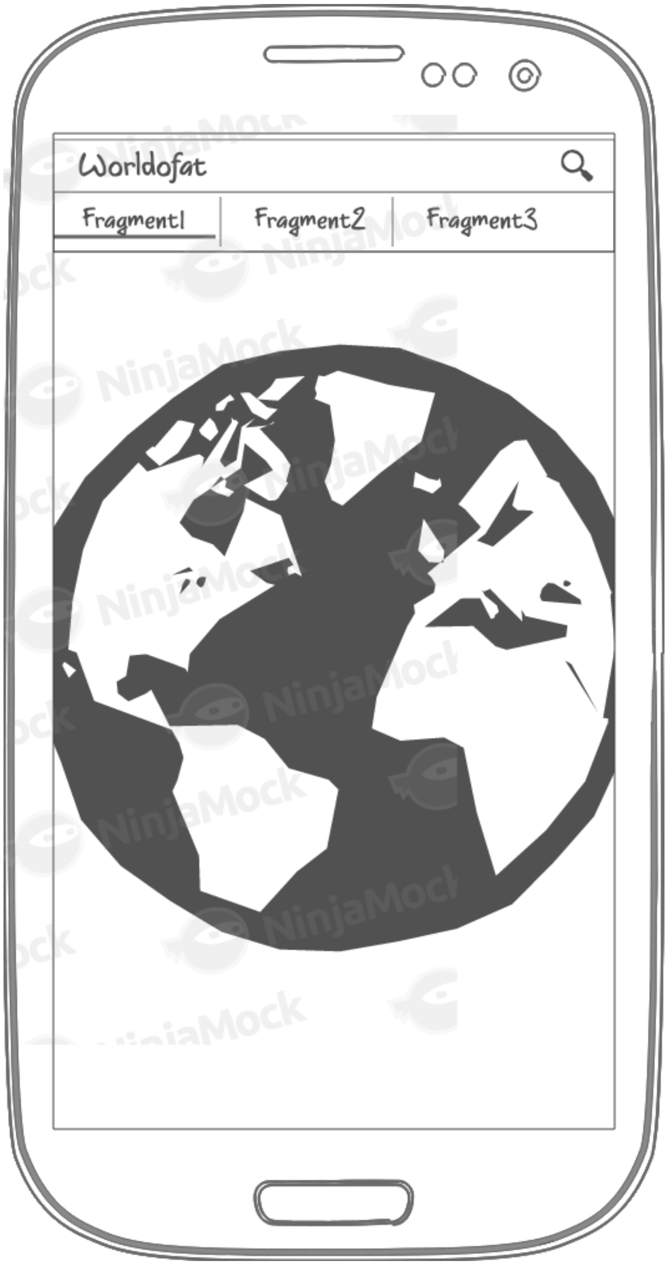
Screen 1 is the welcome screen that hosts a Navigation Drawer (Implemented as proof of concept, do not expect anything more than a couple of Toast messages)
The main screen has a TextView which displays the text "Assistive Technology". Following this is another text view that displays a brief definition for what AT stands for. This is succeeded by a

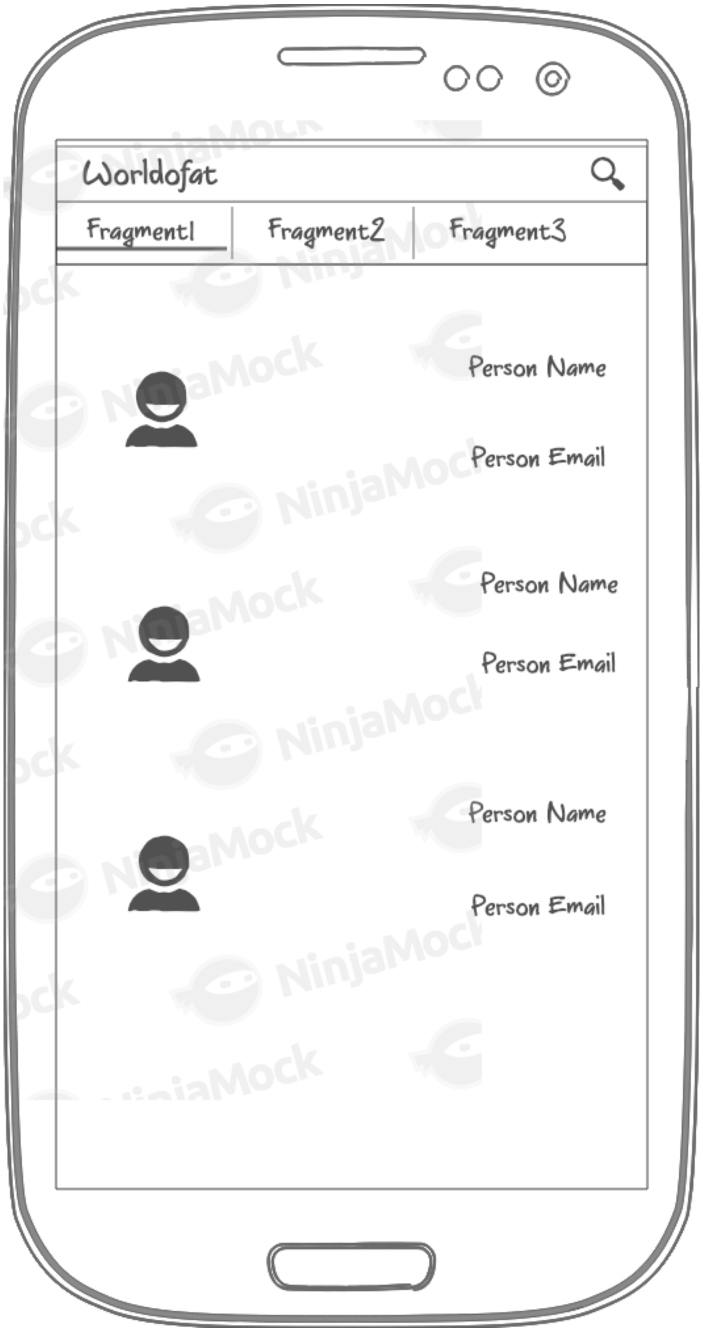
Button which displays the text “Details”. This button click takes the user to next screen in the activity.

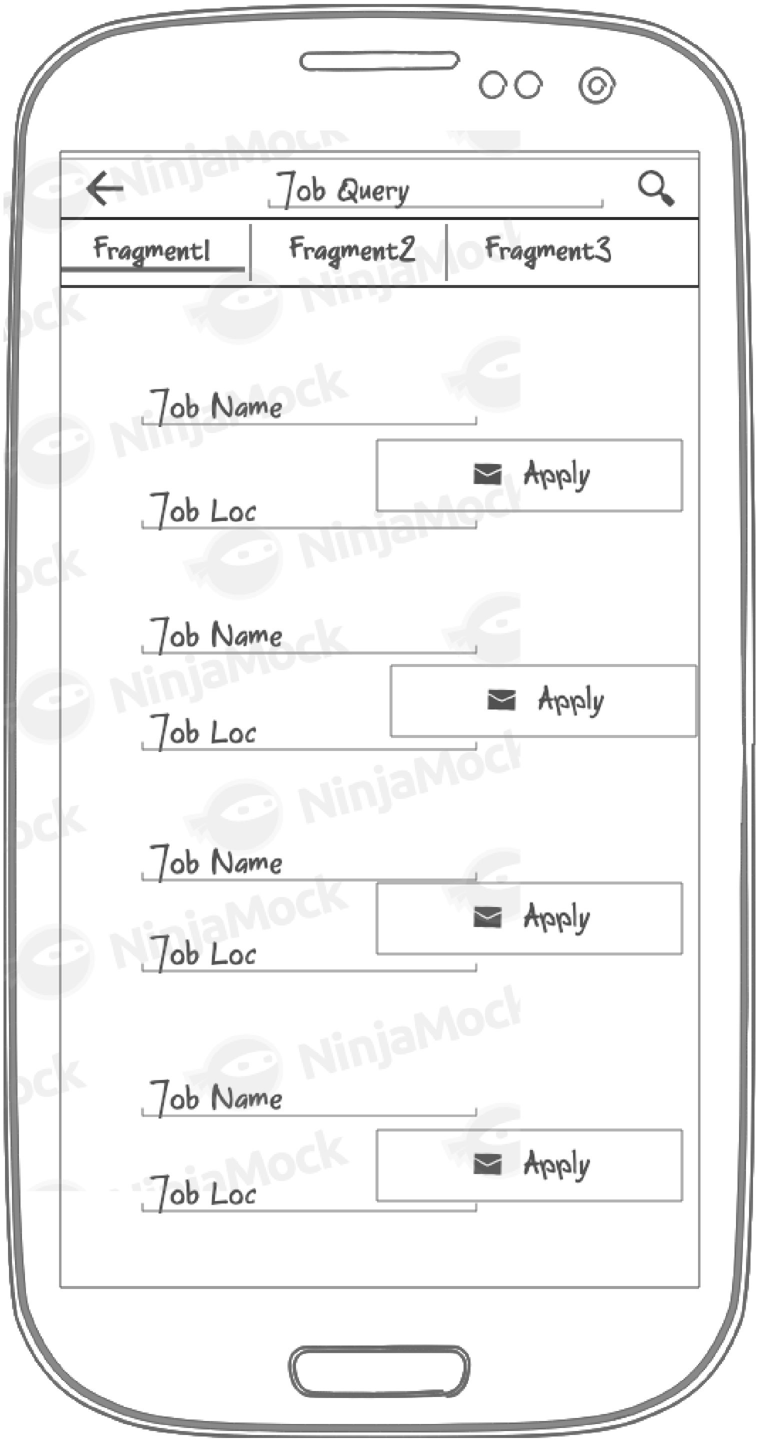
The Fragment View Pager below the “Details” buttons is used to fetch and display a running list of AT adaptations sourced from Getty Images

Screen 2

Screen 2 shows content for each of the Fragments down below and the widget on the main screen


Worldofat
Fragment1
Fragment2
Fragment3


Worldofat
Fragment1
Fragment2
Fragment3
Person Name
Person Email
Person Name
Person Email
Person Name
Person Email



Add as many screens as you need to portray your app’s UI flow.

Widget Screen:

Key Considerations

How will your app handle data persistence?

Describe how your app with handle data. (For example, will you build a Content Provider or use Firebase Realtime Database?)

The app has a job search feature which uses Firebase Realtime Database to process search queries and provide results

The app has personnel database which is a Sqlite Database to store and display that data
App keeps all strings in strings.xml & has RTL layout switching for all layouts
App provides a widget for providing relevant information on the home screen

Will use Android Work Manager to update Widget data periodically

Describe any edge or corner cases in the UX.

For example, how does the user return to a Now Playing screen in a media player if they hit the back button?

Have provided app bar navigation button to handle transition from different activity screens

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

I am using Picasso to load pictures sourced from Getty Images and to load pictures of placeholder personnel images

Use a third party random Java POJO object generator to generate random job data that can then be searched. (https://github.com/android-Infoedge/randomizer)

Retrofit to make asynchronous HTTP requests to fetch images from Getty Images

Parceler (https://github.com/johncarl81/parceler) to parcel model data objects
App makes use of Async Task for placing network calls through Retrofit

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

Implements Google Ad services to display banner ads in the Activity screen of the App

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

This project will be solely written in the **Java programming language** for app implementation purposes. This will use several stable releases of **Android studio (3.1).**, **Gradle (4.10.1)** & **Android build tools version (3.3.0)** and **target sdk version (28**) & **Google Play services (4.2.0)**
This application uses the **Android Jetpack libraries** and **third party libraries** as listed below in the dependencies section of the app module gradle file.

Start Android studio to create a empty activity such that it extends AppcompatActivity. Navigate over to the gradle files and open build.gradle and project.gradle. The build.gradle file would be the module level file in which dependencies for the project are to be added.

Check to see that this file has the following lines

```
        “apply plugin ‘com.android.application’
```

Followed by

```
 android{
// lists the default implementation for the app
        compileSdkVersion
        defaultConfig {
        applicationId
        minSdkVersion
        targetSdkVersion
        versionCode
        versionName
        multiDexEnabled
        testInstrumentationRunner

buildTypes{
release {
    minifyEnabled true
    proguardFiles getDefaultProguardFile('proguard-android.txt'), 'proguard-rules.pro'
  }
}

 Dependencies{
//lists all dependencies for the ‘app’ module

api 'com.google.android.material:material:1.1.0-alpha02'
  implementation fileTree(include: ['*.jar'], dir: 'libs')
  implementation 'androidx.appcompat:appcompat:1.1.0-alpha01'
  implementation 'androidx.constraintlayout:constraintlayout:2.0.0-alpha3'
  implementation 'com.google.android.material:material:1.1.0-alpha02'
  implementation 'androidx.legacy:legacy-support-v4:1.0.0'
  implementation 'com.google.firebase:firebase-database:16.0.5'
  testImplementation 'junit:junit:4.12'
  implementation 'com.squareup.retrofit2:retrofit:2.3.0'
  implementation 'com.squareup.retrofit2:converter-gson:2.3.0'
  implementation 'com.google.code.gson:gson:2.8.2'
  implementation 'com.squareup.picasso:picasso:2.71828'
  implementation 'org.parceler:parceler-api:1.1.10'
  implementation 'androidx.cardview:cardview:1.0.0'
  implementation 'androidx.recyclerview:recyclerview:1.0.0'
  implementation 'jp.wasabeef:recyclerview-animators:2.3.0'
  implementation 'com.firebaseui:firebase-ui-database:4.3.1'
  implementation 'androidx.arch.core:core-runtime:2.0.1-alpha01'
  implementation 'androidx.arch.core:core-common:2.0.0'
  androidTestImplementation 'androidx.test:runner:1.1.1'
  androidTestImplementation 'androidx.test.espresso:espresso-core:3.1.1'
   implementation 'org.parceler:parceler-api:1.1.12'
  annotationProcessor 'org.parceler:parceler:1.1.12'
  implementation 'io.github.benas:random-beans:3.7.0'
  implementation 'com.infoedge:arandomizer:0.1-beta1'
  implementation 'com.rockerhieu:rv-adapter-states:1.2+'
  implementation 'com.google.firebase:firebase-ads:17.1.2'
  implementation 'androidx.multidex:multidex:2.0.1'
  implementation 'androidx.constraintlayout:constraintlayout:1.1.3'
  implementation 'androidx.viewpager:viewpager:1.0.0'
  implementation "android.arch.work:work-runtime:1.0.0-beta01"

}
```

Task 2: Implement UI for Each Activity and Fragment

UI for IntroActivity consists of a AdView, ToolBar with Navigation Drawer, TextViews, ViewPager Fragment with ImageViews, FAB to kick start image loading into the Viewpager Fragment and Button for Picture-in-Picture support.

Create the layout file that has a Toolbar, DrawerLayout and UI for Google Adview, two Textviews and a Button, Floating Action Button

Task 3: Discuss Second Activity creation and Navigation

Clicking on a Button in the IntroActivity launches the second activity which hosts a tabbed fragment adapter that hold three fragments that display information about Assistive Technology, Dummy Personnel data and a Job search Fragment that lets user search and display job data

Create the layout file for the Activity that hosts the tabbed Fragment. The UI consists of AppBarLayout , Toolbar, TabLayout, ViewPager & FrameLayout to load the Fragments it hosts

Task 4: Discuss Tabbed Fragment Creation

The three Fragments being hosted by the activity have three different layout files. Each Fragment displays unique information.

Fragment 1 has simple Webview
Fragment 2 has a Recycler View which hosts a CardView to display details of personnel
Fragment 3 has a Recycler View embedded with a searchview in the toolbar

All three Fragments to be hosted in a tabbed view pager layout

Task 5: Discuss Fragment Implementation

Hooking up the different fragments and Activities to download and display pertinent information.
FAB button in intro activity displays custom AT solutions/implementations sourced from Getty Images

Webview in Fragment 1 hosted in the second activity displays website information
RecyclerView in Fragment 2 displays Personnel data stored in SQLite Database
RecyclerView in Fragment 3 displays Job query results persisted and sourced from a Firebase Database

Task 6: Discuss App Widget Creation and Implementation

Create an App Widget to display App Icon to launch app & fire network call to fetch images from Getty Images.

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**"