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: hendercine

Sunday Assembly

Description

Live Better, Help Often and Wonder More!

An app for the Sunday Assembly community. Intended to help immerse assemblers into the Assembly with the program schedule, song lyrics and speaker profiles. It also enables assemblers to connect with each other and Sunday Assembly organizers in between assemblies, with notifications about service and social events.

Intended User

Anyone who attends, has thought of attending or might be interested in attending a Sunday Assembly event. This first version of the app will focus only on Sunday Assembly Los Angeles but is looking toward expanding its scope for other locales in the future.

Features

List the main features of your app. For example:

- Saves user information
- Takes pictures
- Displays event calendars
- Displays Event Schedule
- Displays song lyrics
- Sends push notifications about events
- Allows users to chat with other users
- Links to Sunday Assembly (Los Angeles) social media

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.

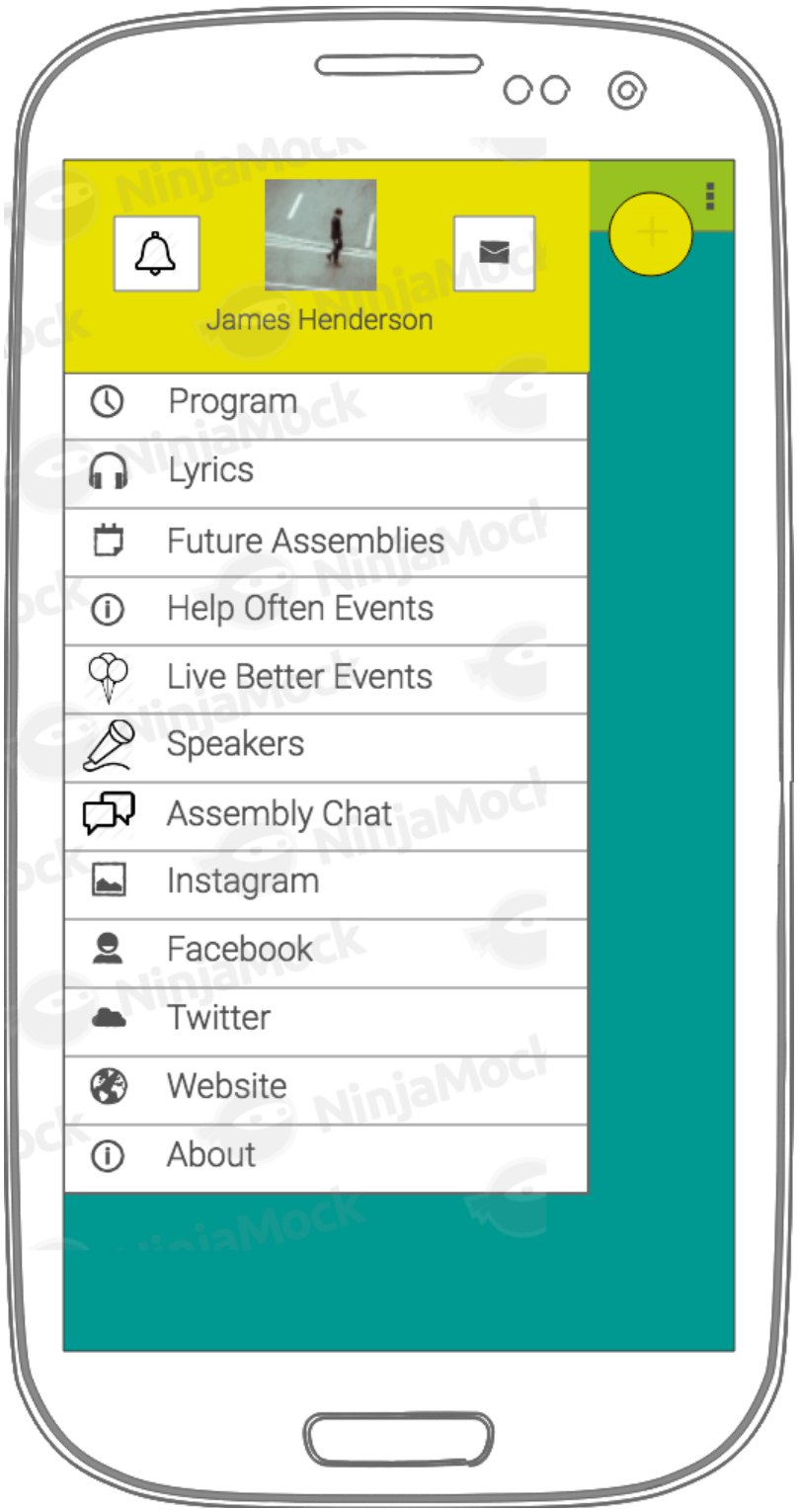
Screen 1

Splash page with the Sunday Assembly Logo to display on App startup.

Screen 2

Example of the Program screen. Most of the app's screens will be list layouts or text for reading info, like the about page or the lyrics.

Screen 3

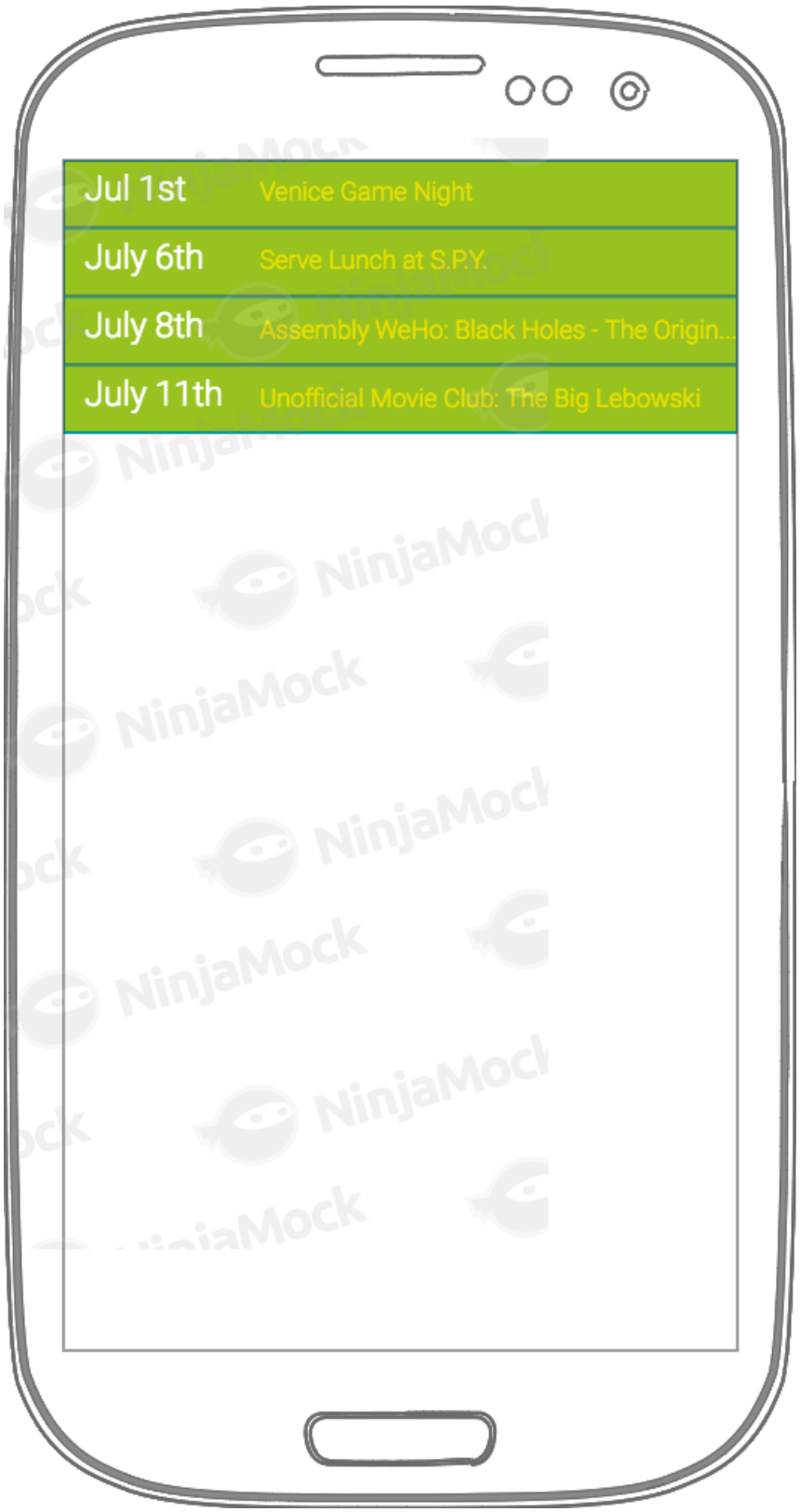


Example of the Navigation Drawer. This will be the hub of the app. Links to events and social media should use pending intents to display them in either a browser or the appropriate app.

Screen 4

Example of Tablet Mode. Navigation drawer becomes the sidebar and we go into a dual pane layout.

Screen 5



Widget Mock.

Key Considerations

How will your app handle data persistence?

The app will use Firebase Realtime Database to handle Data persistence.

Describe any edge or corner cases in the UX.

Everything will be accessible to the user via the side Navigation Drawer and there really shouldn't be any edge cases for that reason. The back button will navigate up to the most recent activity or fragment or else exit the app.
Screen orientation changes will be handled with savedInstanceState and the Realtime Database should handle caching of data when the device is offline.

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

Definite Implementation

Android Support Library - To use Android's amazing built-in features like Appbar, Collapsing Toolbar, etc.
Firebase UI Auth - To handle the sign-in flow.
Glide - To manage loading and caching of images.
Butterknife - To attach views and reduce boilerplate.
Timber - To simplify logging.
Espresso - To handle UI testing.

Hope to Implement (time/patience provided)

Parceler - To handle POJOs and reduce boilerplate.
RxJava & RxAndroid - To handle asynchronous tasks and thread management.
Retrofit - To handle HTTP requests and parse any JSON that may be available from the Sunday Assembly host (NationBuilder) webserver.
Dagger2 - To implement dependency injection and lay the framework for an MVP architecture. I have completed a tutorial that implements this architecture and it follows much the same UI convention as my planned app.

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

The app will implement Firebase Realtime Database, as mentioned above, as well as Firebase Authentication for user sign-in and the chat feature of the app. The app will also use Firebase Cloud Messaging to send notification campaigns about events to users with the app installed.

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

Steps to setup and/or configure this project (**Note: Project will be written solely in the Java Programming Language):

- ❏ Create GitHub Repo called SundayAssembly
- ❏ Make sure Android Studio and Gradle are up to date
- ❏ Set project VCS to git, make initial commit on master and push to origin
- ❏ Configure App/build.gradle
    - ❏ Set compileSdkVersion to 27
    - ❏ buildToolsVersion to 27.0.3
    - ❏ minSdkVersion to 19
    - ❏ targetSdkVersion to 27
    - ❏ Configure libraries:
        - ❏ Suppport-v4:
          implementation "com.android.support:support-v4:$androidSupportVersion"
        - ❏ Suppport-v13:
          implementation "com.android.support:support-v13:$androidSupportVersion"
        - ❏ Suppport-v7:
          implementation
          "com.android.support:appcompat-v7:$androidSupportVersion"
        - ❏ Pallette:
          implementation "com.android.support:palette-v7:$androidSupportVersion"
        - ❏ RecyclerView:
          implementation
          "com.android.support:recyclerview-v7:$androidSupportVersion"
        - ❏ Cardview:
          implementation "com.android.support:cardview-v7:$androidSupportVersion"
        - ❏ Suppport-design:
          implementation "com.android.support:design:$androidSupportVersion"
            - ❏ Add ext {androidSupportVersion = “27.1.1”} under repositories{} in allProjects{}
        - ❏ ConstraintLayout:
          implementation 'com.android.support.constraint:constraint-layout:1.1.0'
        - ❏ Glide:
          implementation 'com.github.bumptech.glide:glide:4.7.1'
          annotationProcessor 'com.github.bumptech.glide:compiler:4.7.1'

- // Google Services
  implementation 'com.google.gms:google-services:3.1.2'
- Firebase:
  implementation 'com.google.firebase:firebase-core:15.0.2'
  implementation 'com.google.firebase:firebase-database:15.0.1'
  implementation 'com.google.firebase:firebase-auth:15.1.0'
  implementation 'com.google.firebase:firebase-messaging:15.0.2'
    - Add to bottom of app/build,gradle:
      apply plugin: 'com.google.gms.google-services'
- Firebase UI Auth:
  implementation 'com.firebaseui:firebase-ui-auth:3.3.1'
- Parceler:
  implementation 'org.parceler:parceler-api:1.1.9'
  annotationProcessor 'org.parceler:parceler:1.1.6'
- Gson:
  implementation 'com.google.code.gson:gson:2.8.2'
- Retrofit:
  implementation 'com.squareup.retrofit2:retrofit:2.3.0'
  implementation 'com.squareup.retrofit2:converter-gson:2.3.0'
  implementation 'com.squareup.retrofit2:adapter-rxjava:2.3.0'
- RxAndroid:
  implementation 'io.reactivex.rxjava2:rxandroid:2.0.1'
- RxJava:
  implementation 'io.reactivex.rxjava2:rxjava:2.1.6'
- RxNetwork:
  implementation 'io.andref:Rx.Network:1.0.1'
- Butterknife:
  implementation 'com.jakewharton:butterknife:8.8.1'
  annotationProcessor 'com.jakewharton:butterknife-compiler:8.8.1'
- Timber:
  implementation 'com.jakewharton.timber:timber:4.6.0'
- // Espresso and Testing Dependencies:
  androidTestImplementation('com.android.support.test.espresso:espresso-cor
  e:2.2.2') {
  exclude group: 'com.android.support', module: 'support-annotations'
  }
  androidTestImplementation('com.android.support.test:runner:0.5') {
  exclude group: 'com.android.support', module: 'support-annotations'
  }
  androidTestImplementation('com.android.support.test.espresso:espresso-con
  trib:2.2') {
  // Necessary to avoid version conflicts
  exclude group: 'com.android.support', module: 'appcompat'

```
exclude group: 'com.android.support', module: 'support-v4'
exclude group: 'com.android.support', module: 'support-annotations'
exclude module: 'recyclerview-v7'
 }
 androidTestImplementation
'com.android.support.test.espresso:espresso-idling-resource:3.0.1'
 implementation
'com.android.support.test.espresso:espresso-idling-resource:3.0.1'
```

Task 2: Implement UI for Each Activity and Fragment

- ❏ Build UI for MainActivity
  - ❏ Create res/layout/activity_main
- ❏ Build UI for MainFragment
  - ❏ Create res/layout.fragment_main
- ❏ Build UI for EventProgramFragment
  - ❏ Create res/layout/fragment_event_program
  - ❏ Implement RecyclerView
  - ❏ Build EventRecyclerViewAdapter
- ❏ Build UI for Navigation Drawer
  - ❏ Create res/layout/nav_drawer
  - ❏ Implement RecyclerView
  - ❏ Build NavDrawerRecyclerViewAdapter
- ❏ Build UI for AssemblyChatActivity
  - ❏ Create res/layout/activity_chat
  - ❏ Build ChatAdapter
- ❏ Build UI for
  - ❏ AboutFragment
    - ❏ Create layout xml
  - ❏ SpeakerFragment
    - ❏ Create layout xml
  - ❏ FutureAssembliesFragment
    - ❏ Create layout xml
  - ❏ LyricsFragment
    - ❏ Create layout xml
  - ❏ HelpOftenFragment
    - ❏ Create layout xml
  - ❏ LiveBetterFragment
    - ❏ Create layout xml
  - ❏ WebsiteFragment
    - ❏ Create layout xml

Task 3: Implement Firebase Realtime Database, Auth and UI Auth

- ❏ Create Firebase Project on Firebase Dashboard
- ❏ Add Firebase to App and Connect to Firebase
- ❏ Download google JSON
- ❏ Implement FirebaseDatabase
- ❏ Implement DatabaseReference
- ❏ Implement FirebaseAuth
- ❏ Set read/write rules for app
- ❏ Set read/write rules for chat
- ❏ Create User POJO
- ❏ Set up sign-in flow with method from FirebaseUIAuth Library

Task 4: Implement Notifications

- ❏ Add the following to the app manifest:

```
<service
      android:name=".MyFirebaseMessagingService">
    <intent-filter>
      <action android:name="com.google.firebase.MESSAGING_EVENT"/>
    </intent-filter>
</service>

<service
    android:name=".MyFirebaseInstanceIDService">
    <intent-filter>
      <action android:name="com.google.firebase.INSTANCE_ID_EVENT"/>
    </intent-filter>
</service>

<!-- Set custom default icon. This is used when no icon is set for
incoming notification messages. See README(https://goo.gl/l4GJaQ) for
more. -->

<meta-data
android:name="com.google.firebase.messaging.default_notification_icon"
android:resource="@drawable/ic_stat_ic_notification" />
<!-- Set color used with incoming notification messages. This is used
```

```
when no color is set for the incoming notification message. See
README(https://goo.gl/6BKBk7) for more. →
<meta-data
android:name="com.google.firebase.messaging.default_notification_color"
android:resource="@color/colorAccent" />

<meta-data
android:name="com.google.firebase.messaging.default_notification_channe
l_id"
android:value="@string/default_notification_channel_id"/>
```

- ❏ Retrieve the current token: `FirebaseInstanceId.getInstance().getToken()`
- ❏ Continue following the steps found here

Task 4: Polish the UI with Material Design

- ❏ Implement Appbar and CollapsingToolbar
- ❏ Implement Activity enter and exit transitions

Task 5: Build the Widget

- ❏ Create widget that displays the calendar of upcoming events
- ❏ Example

Task 6: Implement Remaining Accessibility Features

- ❏ Check for
  - ❏ Content Descriptions
  - ❏ Strings
  - ❏ D-pad navigation

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**"