GitHub Username: mikepalarz

JammyJamz

Description

This app is intended to allow users to share what current music they're listening to with a newsfeed-style interface. End users will also see what their friends are currently listening to (perhaps out of pure curiosity or for inspiration to check out new artists) and share music interests with friends.

Intended User

The app is intended for music enthusiasts or for anyone who is comfortable with sharing their musical interests. It is meant to capture those moments in a user's day when they're listening to a particular song and can't help but think to themselves "Man, this song is awesome! I've **gotta** share it with my friends!". Substitute "awesome" with "a jammy jam" if the song is totally off the hook.

The focus wouldn't be strictly on individual songs/tracks. Users will also be able to share their albums or artists if they feel it is more appropriate for their current musical mood. For example, an end user could express that they're really enjoying *The Wall* by Pink Floyd as a whole, instead of a track. Another instance could be that the user in fact simply enjoys listening to Pink Floyd instead of a particular song or album. The key concept is that the user can express their musical interests in 3 different ways.

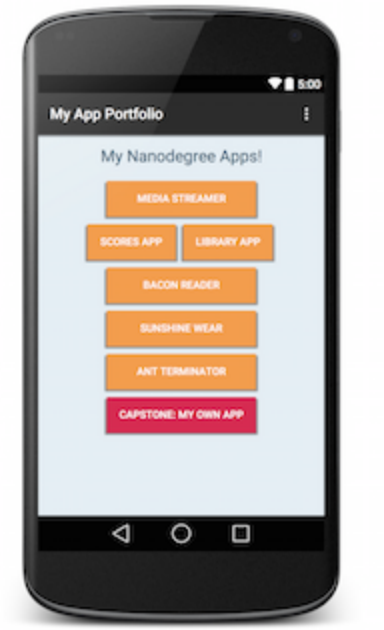
Features

Main features of the app:

- News feed interface which shows all registered users' current musical interests (tracks, albums, and/or artists).
- A search bar which allows users to find tracks/albums/artists that they're interested in that they would like to express their interests about to all other JammyJamz users.
- Previews of tracks/albums/artists.
    - If the user has the Spotify app installed, they'll be taken directly to that particular track/album/artist within Spotify.
    - If they do not have Spotify installed, then they'll be taken to the Spotify website for that track/album/artist that has 30s previews.
- Allows users to follow each other so that they can see other users' latest posts and share music with them.
- Allow users to share music with each other if they feel like they know a particular person that would really enjoy a track/album/artist.
    - Users will receive a notification when a friend shares a musical interest with them.

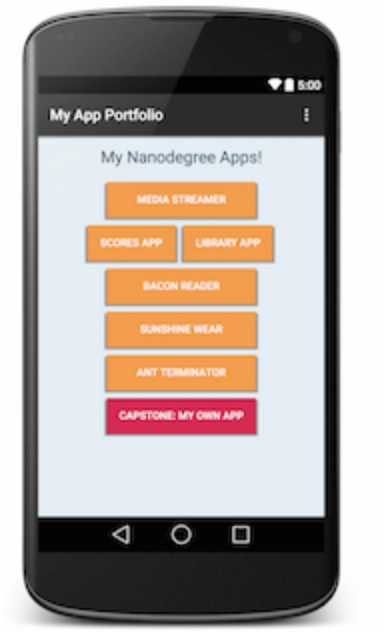
User Interface Mocks

Screen 1



Replace the above image with your own mock [click on the above image, then navigate to Insert → Image…]
Provide descriptive text for each screen

Screen 2



Replace the above image with your own mock [click on the above image, then navigate to Insert → Image…]
Provide descriptive text for each screen

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

Key Considerations

How will your app handle data persistence?

The app will use Firebase Realtime Database for data persistence. This is critical for the news feed as well as allowing users to follow each other and share music with one another.

Describe any edge or corner cases in the UX.

The fact that I'm using the Spotify Web API can be considered an edge case as a whole. I'm assuming/expecting users to be comfortable with using Spotify, even if they have another, more preferred music streaming service. I did look into other popular music streaming services, such as Google Play Music, Amazon Prime Music, and Apple Music. However, they either did not have a RESTful API or they didn't offer the features that aligned with what I wanted to do for my project (at least based on my research).

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

I plan on using the following libraries:

- Spotify Web API - Searching for tracks, albums, and artists; it will also be used to allow users to either listen to or preview music.
- Picasso - image loading and caching
- Butterknife - bindings XML views to View objects within Java
- Firebase - Realtime Database for the news feed, Authentication for user sign-in
- Timber - better control over error logging
- Espresso - UI testing

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

I will be using Firebase heavily for this project, especially the Realtime Database.

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:

- Configure libraries
- Something else

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:

- Build UI for MainActivity
- Build UI for something else

Task 3: Your Next Task

Describe the next task. For example, “Implement Google Play Services,” or “Handle Error Cases,” or “Create Build Variant.”

Describe the next task. List the subtasks. For example:

- Create layout
- Something else

Task 4: Your Next Task

Describe the next task. List the subtasks. For example:

- Create layout
- Something else

Task 5: Your Next Task

Describe the next task. List the subtasks. For example:

- Create layout
- Something else

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**"