

[Description](#)

[Intended User](#)

[Features](#)

[User Interface Mocks](#)

[Screen 1](#)

[Screen 2](#)

[Screen 3](#)

[Screen 4](#)

[Screen 5](#)

[Screen 6](#)

[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.](#)

[Next Steps: Required Tasks](#)

[Task 1: Build Startup Screen](#)

[Task 2: API Calls to Last.FM](#)

[Task 3: Search Input Screen](#)

[Task 4: Location / Map API \(Google\)](#)

[Task 5: Content Provider](#)

[Task 6: Widget](#)

[Task 7: Tablet Integration](#)

[Task 8: AdMob Integration](#)

**GitHub Username :Puttaraj**

## MyLastFM

### Description

Keep your music collection at your fingertips using a new Last.FM Playlist assistant. Browse through the Top Music Charts (including your country's' Charts) in search of something fresh. Search for your favorite tracks by Tag, Artist, Track name, or (Artist/Album). Add individual tracks to the Playlist saved on your device for convenient viewing. Includes Tablet support and a Launcher Widget to quickly access your Playlist.

### Intended User

This application is intended for music fans who'd like to have Music search capabilities at their fingertips. It can also assist them in finding new bands/kinds of music they may not have known about otherwise.

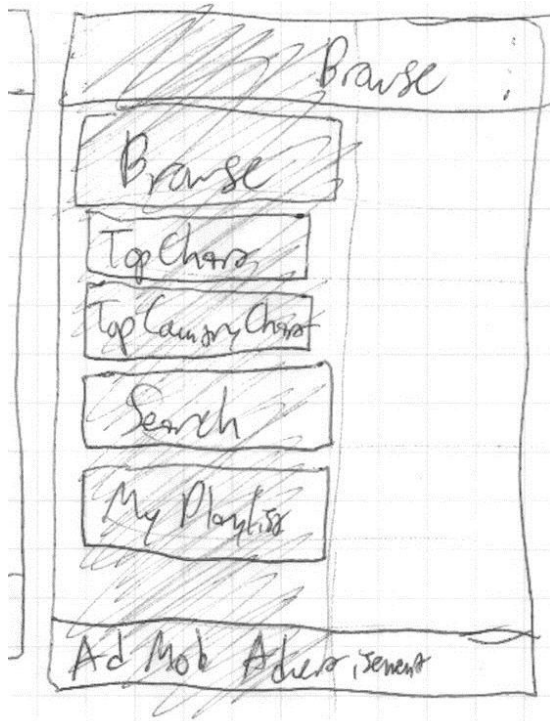
## Features

- Clean, ease-of-use interface
- Browse Tracks on the Top Charts
- Using your device's location, browse tracks on the Top Charts in your country
- Search for tracks in a variety of ways (Tag, Artist, Track name, (Artist/Album) combination)
- Convenient storage of user-populated playlist accessible within seconds
- Detailed Track activities
- Tablet Layout support (Landscape)
- Launcher Widget displaying Track last entered into Playlist
- Full accessibility features

## User Interface Mocks

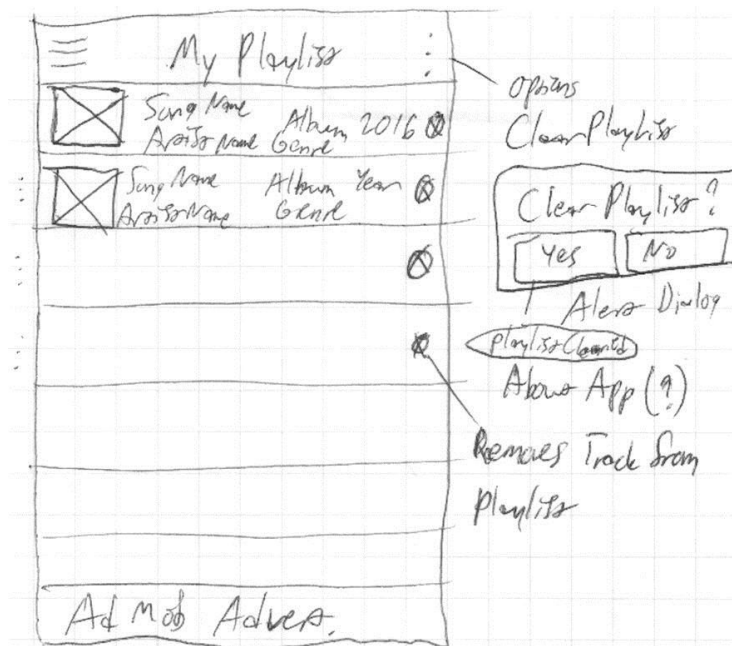
Word of caution: I am terrible at making drawn UI interfaces. Please know I tried my best to draw these out.

## Screen 1



Shows initial Track Playlist (empty) (Mistakenly named Browse: Should be "My Playlist")  
 Using a NavigationDrawer for app navigation  
 AdMob advertisement at the bottom to avoid being in the way

## Screen 2



Playlist partially populated

Each row is an individual Track

The not-shown images are track-supplied from the API

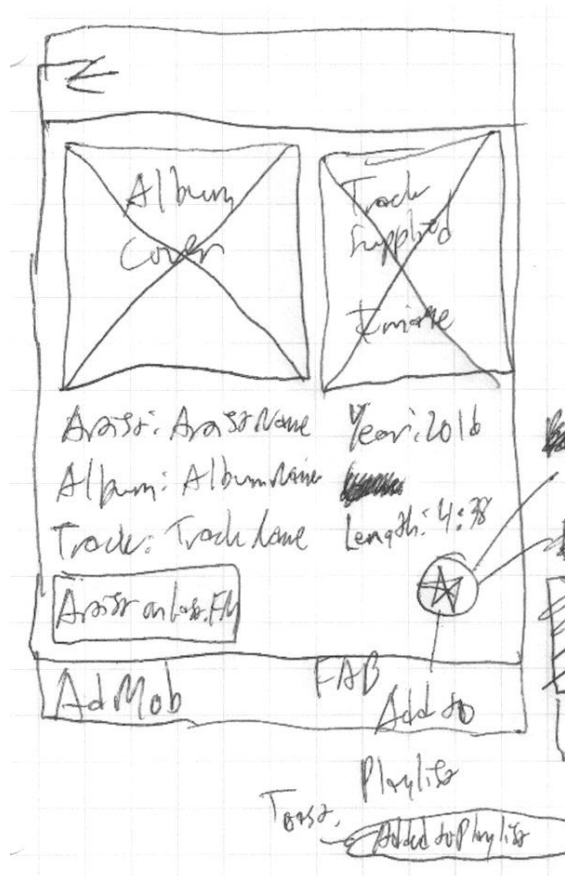
Album artwork for the album which the track comes is acquired upon track selection

Each track can be removed from the playlist here

Upon removal, list item should disappear with the other items readjusting (going to look at how that's possible)

Options Menu including Clear Playlist (with alert dialog) and About App Screen (Simple)

### Screen 3



Detail Track Activity

Shows both Track-supplied image and Album Cover acquired upon track selection

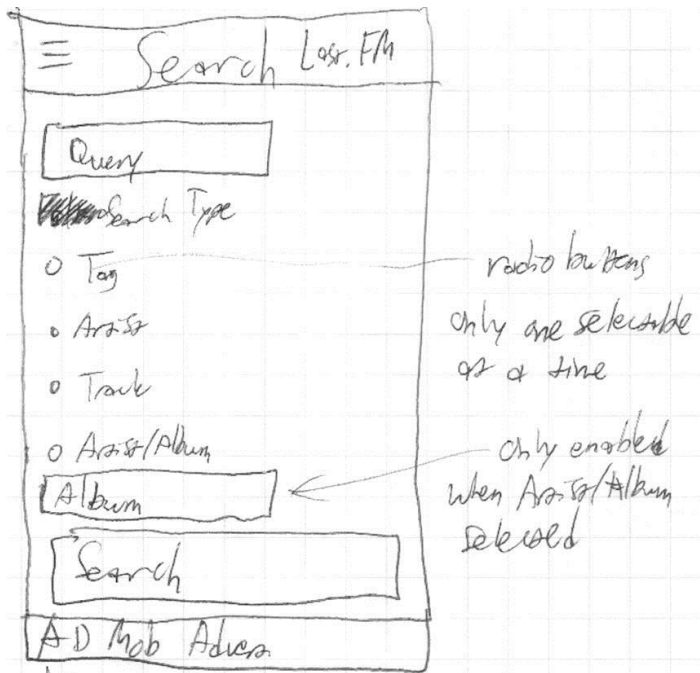
Additional details about track itself

"Artist on Last.FM" button opens URL in browser to artists' page.

FAB for Add to Playlist (acknowledgement)

(Possibly Turn FAB into Remove from Playlist, if not voided from the Playlist action)

## Screen 4



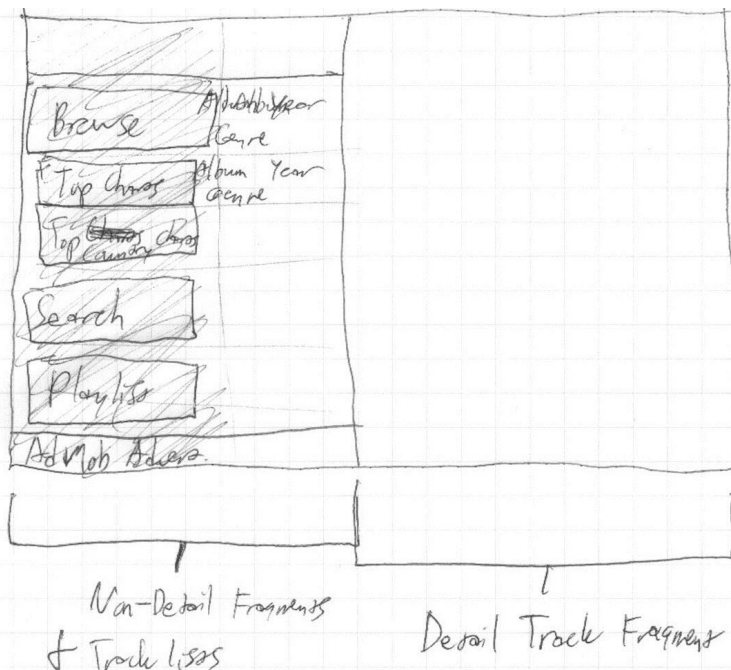
Illustrates the Search Fragment

Query + Search Type denoted by Radio buttoned items

Artist/Album includes secondary search field (not null)

Search results in Track List

## Screen 5



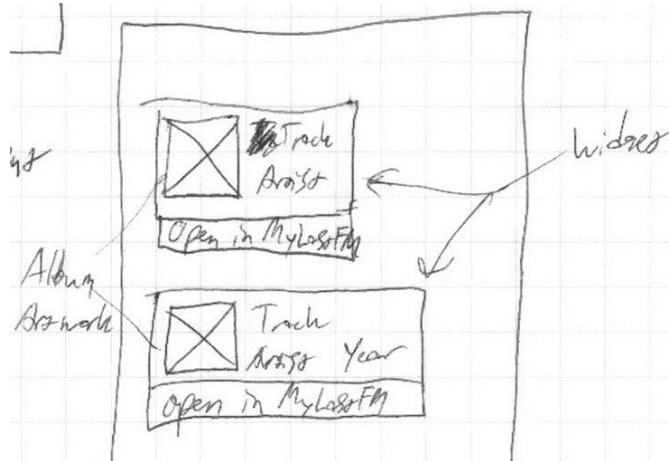
## Tablet Layout Diagram

SplitScreen (twoPane)

Search / Navigation / Track lists on left

Track Details on Right

## Screen 6



## Simple Widget Concept

Displays last-added track to the Playlist

OnClick, opens My Playlist in the application

## Key Considerations

### How will your app handle data persistence?

Largely, the data for the lists of tracks are generated via API calls to Last.FM API. However, this data is not kept around, and can be quickly replaced. The user-populated playlist must be kept beyond closing the app. Thus, a Content Provider will be implemented to store Track objects in a device-stored database. These tracks can be easily mapped to a RecyclerView adapter to display lists of tracks with ease.

### Describe any corner cases in the UX.

Navigation between the lists of tracks and the Search Windows could potentially have back issues. To help alleviate this issue, I decided to use a Navigation Drawer similar to the one in Alexandria to swap out Fragments in the Main Activity. This keeps the UI easy to navigate in the MainActivity Fragments.

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

Originally, I thought Gson would assist with handling of API JSON results. However, the results had to be extracted carefully from the JSON string, so it couldn't be applied as easily. The tracks will each have both an image tied to the track itself as well as the cover of the Album. Because the API results return URLs to the images themselves, background loading of the images were necessary. I'll be using Picasso to help manage that imagery data, as well as handling lack of connection.

## Next Steps: Required Tasks

### Task 1: Build Startup Screen

- Start on empty/populated Playlist screen
- NavigationDrawer
- Dysfunctional options to be implemented later
- Accessibility

### Task 2: API Calls to Last.FM

- JSON parsing
- ArrayList of Track objects
- Population to RecyclerView Adapter
- Reusability to be attempted for all results

### Task 3: Search Input Screen

- Validate Input (length cap / null string)
- Search options affect API query + resulting list
- Connect API search results to RecyclerView Adapter created in Task 2

### Task 4: Location / Map API (Google)

- For use with Top Country Chart tracks
- Uses device location coordinates in Maps API call to acquire Country name

### Task 5: Content Provider

- Detail Track Activity Add Button
- Playlist population of Track objects
- Playlist = list of tracks stored via Content Provider

### Task 6: Widget

- Last-entered Playlist Track displayed
- Click to enter Playlist

### Task 7: Tablet Integration

- Develop UI for tablet (Landscape only) use
- SplitScreen (twoPane)
- Left: Search / Track List
- Right: Detail Track

### Task 8: AdMob Integration

- Add non-invasive advertisements
- Bar at bottom of screen, largely out of user's way

---

### Submission Instructions

1. After you've completed all the sections, download this document as a PDF [ File → Download as PDF ]
2. Create a new GitHub repo for the capstone. Name it "**Capstone Project**"
3. Add this document to your repo. Make sure it's named "**Capstone\_Stage1.pdf**"