How to Use this Template

1. Make a copy [File → Make a copy...]
2. Rename this file: "**Capstone_Stage1**"
3. Replace the text in green

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**"

GitHub Username: sjsingh200893

Musical

Description

Musical in an audio playing app intended for users who don't want complicated music player feature and just want a simple music player.

Features:

- Simple and clean UI
- Lyrics for all the songs in your library
- Track info
- Artist info

Intended User

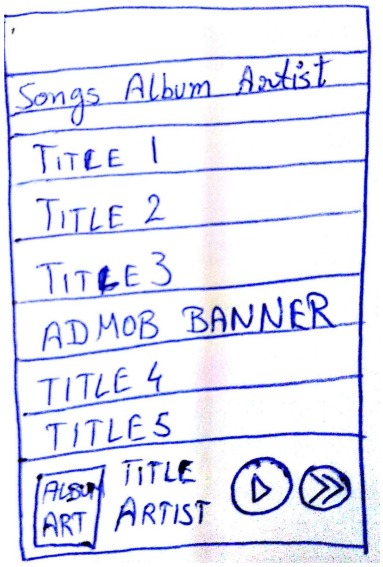
Music lovers who need a simple music app without much complexities

Features

- Retrieves locally stored music files
- Sorts the music library according to songs, albums and artists
- Uses a clean UI and easy UX for new users to understand

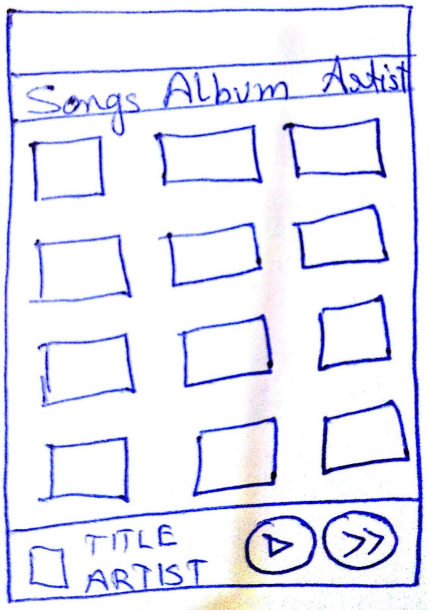
User Interface Mocks

Screen 1



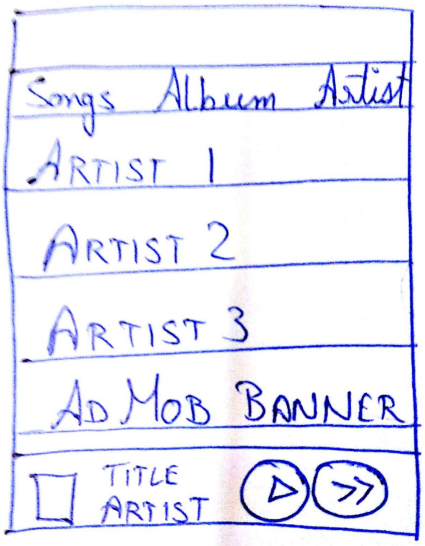
The startup screen has a ViewPager with pages of songs albums and artists.
The default will be songs. This page will contain a ListView of the songs with AdMob banners embedded in between. On the bottom will be a bar with play controls and track information.
Clicking on a song will cause it to play.

Screen 2



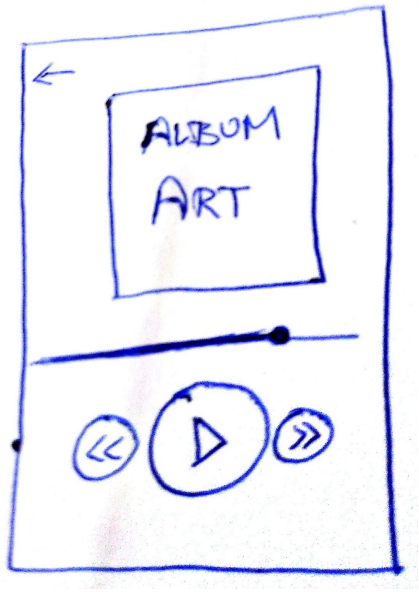
The album page will have a GridView of all the albums-arts clicking on which the DetailActivity with song lists will appear.

Screen 3



The artist page will contain info for the artists in a ListView. Clicking on an artist will open the DetailActivity for artist

Screen 4

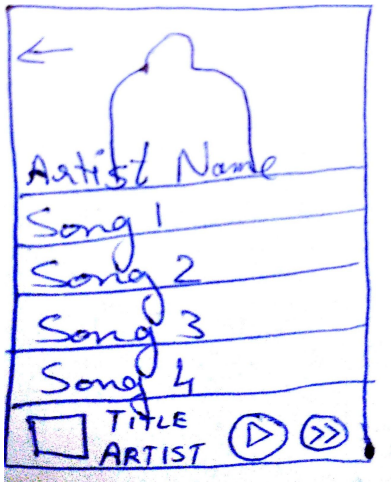


Now Playing Screen which will contain controls for playback

Screen 5

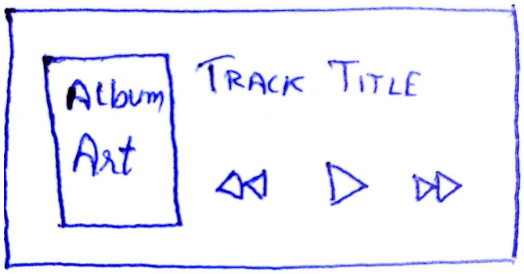
DetailActivity for album containing ListView of songs from the album

Screen 6



DetailActivity Screen for the artist having list of songs from that particular artist

Screen 7



The app Widget with playing controls, album art and title of the track

Key Considerations

How will your app handle data persistence?

The app will use the contentresolver to get the media files from the local storage of the device.

Describe any corner cases in the UX.

For improving the UX, the AdMob banners will be placed in between the list of songs to avoid any accidental interaction with the ads when not intended. Also this will not take any extra space on the screen which will help provide a clean UI

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

For image loading of the album art, Picasso will be used to cache the images and use a placeholder if there is no album art available

Describe how you will implement Google Play Services.

I will use the Google AdMobs for loading the ads which will be placed in the list of songs at every 4th position.

Next Steps: Required Tasks

This is the section where you can take the main features of your app (declared above) and decompose them into tangible technical tasks that you can complete incrementally until you have a finished app.

Task 1: Project Setup

- Configure Picasso
- Configure recyclerView dependency
- Configure dependency for Google AdMobs and Analytics
- Register for MusixMatch Lyrics API Key

Task 2: Implement UI for Each Activity and Fragment

- Build UI for NowPlaying activity
- Build UI for catalogue of songs,albums,artists

Task 3: Your Next Task

- Creating the adapter for song list and implement a view for admobs
- Creating an activity for lyrics on demand
- Creating the adapter for the gridview for albums
- Creating database content provider and loaders for storing the lyrics data locally

Task 4: Your Next Task

- Creating transitions for smooth switching between activities
- Working an AsyncTask for lyrics on demand
- Error checking for null lyrics

Task 5: Your Next Task

- Error handling for songs with improper names in the storage
- Placing placeholders for songs without album arts.
- Generate a paid release without AdMob
- Implement an app widget
