Capstone Project

GitHub Username: zawlynn

Soul Cast

Description

Soul Cast is a new podcast player for people who love to listen to podcast audio. The user can be able to bookmark podcast and download and listen if a phone doesn't have internet connection. The user can be able to search by genre and name.

Intended User

The people who loves to listen to podcast audio.

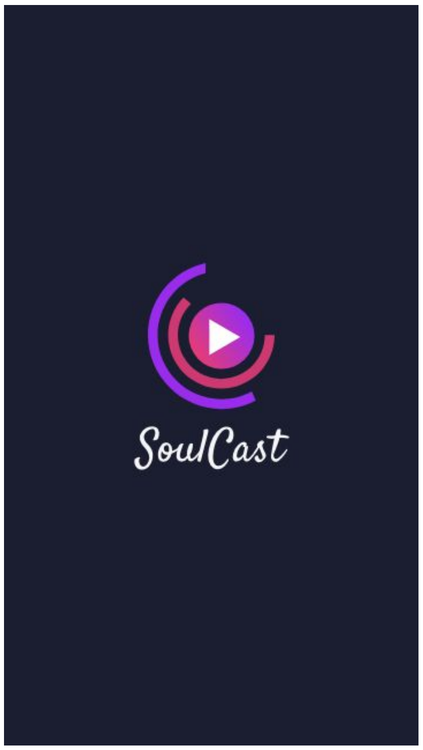
Features

List the main features of your app. For example:

- Show latest podcast
- Bookmark podcast
- Download episode
- Offline support
- Search by Genre and text

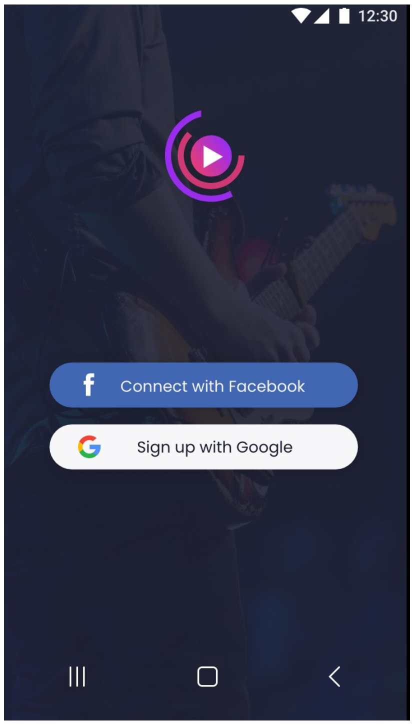
User Interface Mocks

Splash Screen



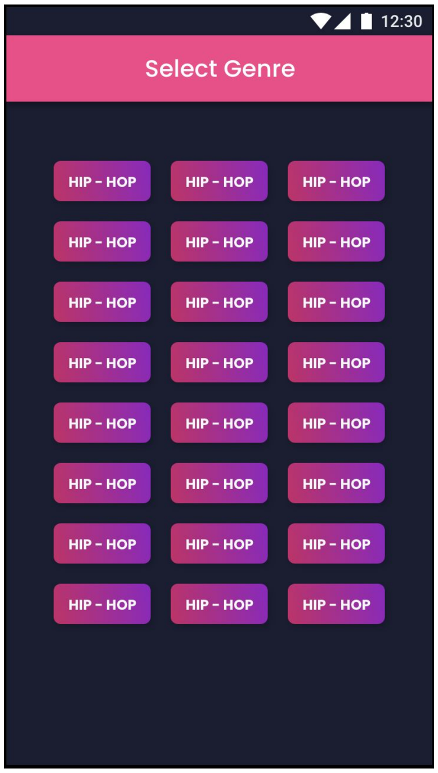
This screen for checking user login session and get data from api. If the user already logged in, app will navigate to home screen otherwise will navigate to login screen

Login Screen



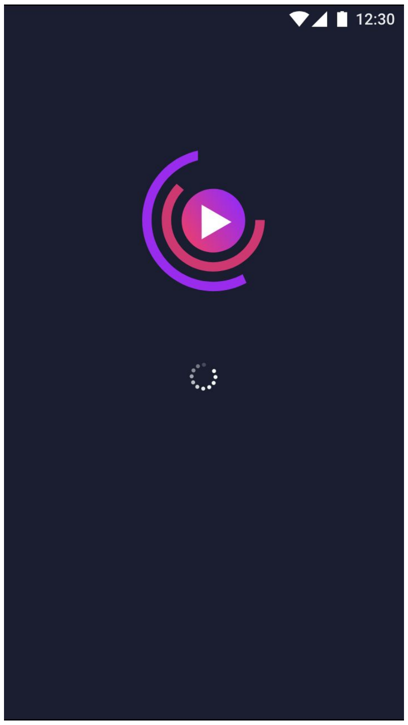
The login screen main function is getting user credential by facebook and google login api.

Genre Screen



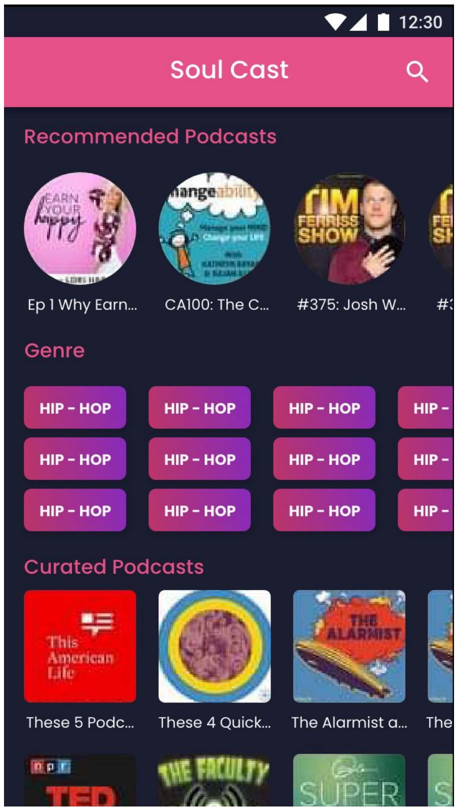
After login function is successful , the app will navigate to this screen, The user must choose at least 3 genres. After that the app retrieve podcast information by genres.

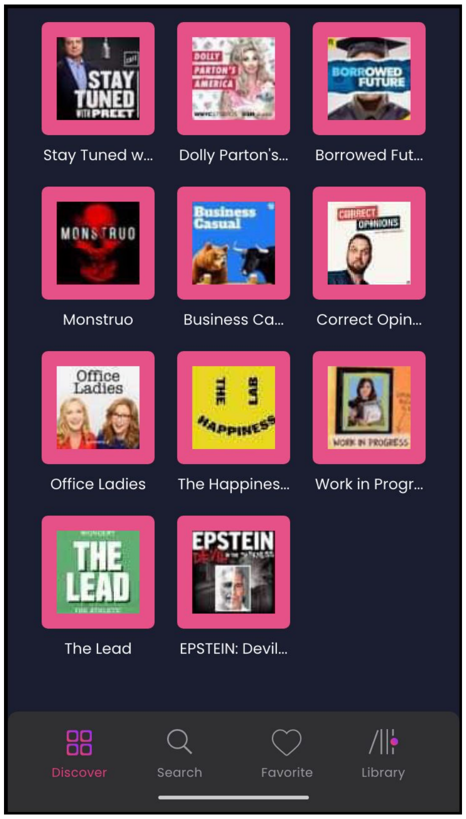
Onboarding Screen



The screen is getting latest podcasts information form api and save in database.

Main Screen



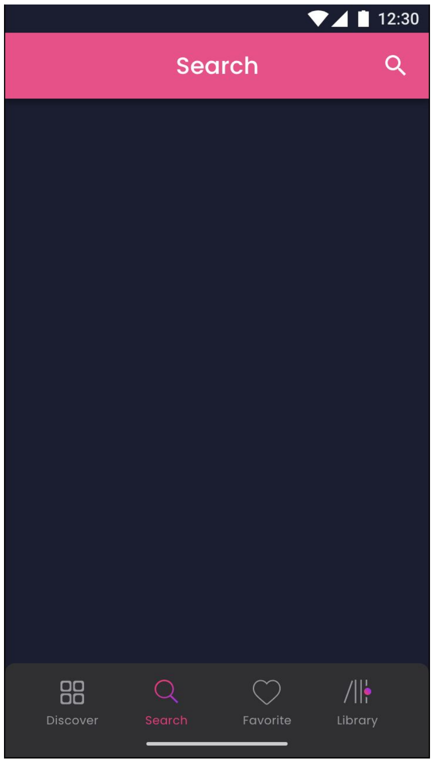


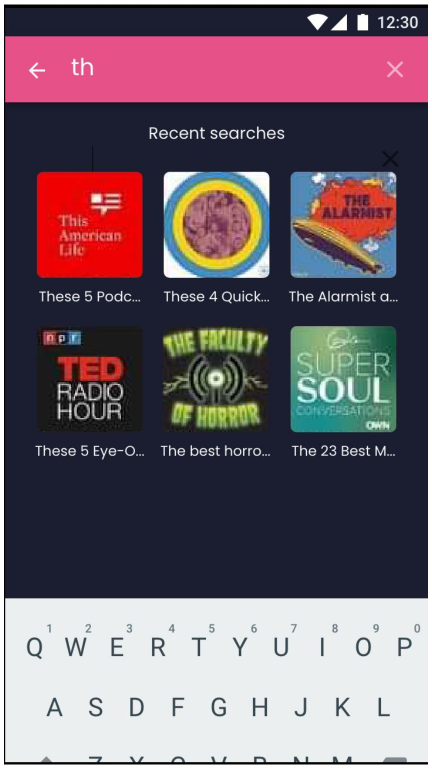
In the main screen the app has 4 main features.

1. New Feeds
2. Search Function
3. Favourite Podcast
4. Library

- New Feeds
  - The function will show up latest podcast, recommend podcast and the podcast which you may like

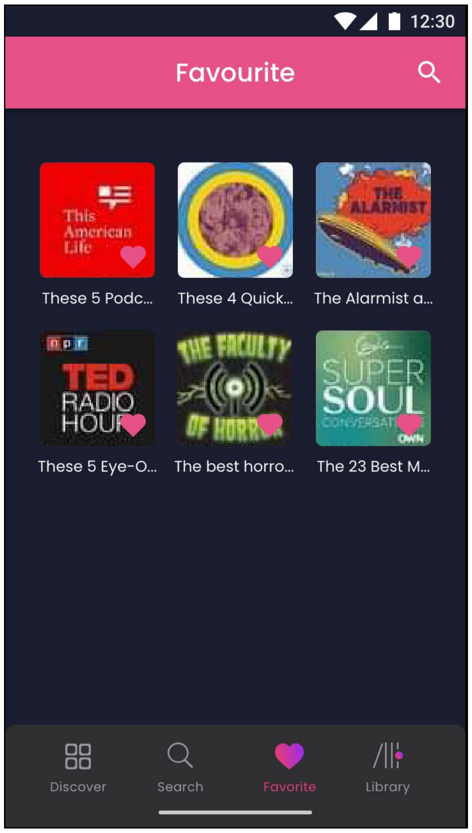
- Search Function





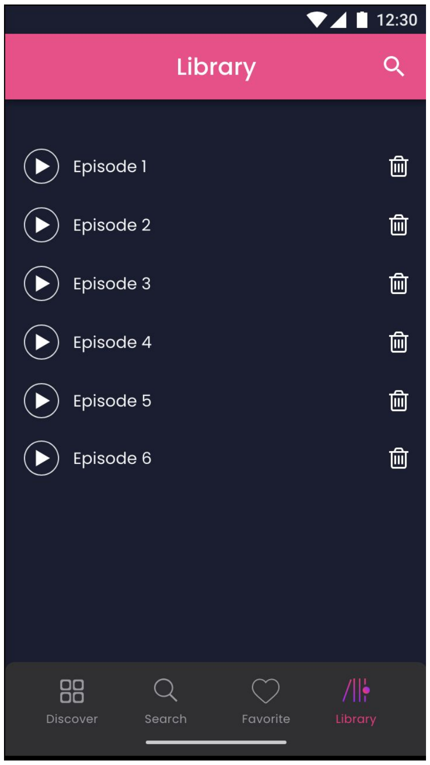
  - A user can be able to search podcast by name

- Favourite Podcast



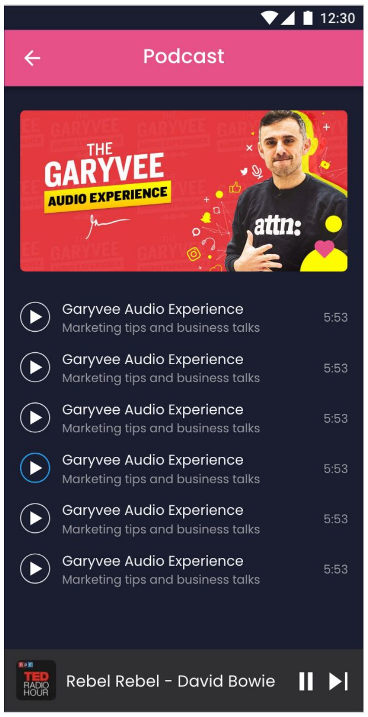
  - The function will show up the podcast that a user click by love icon in podcast

- Library



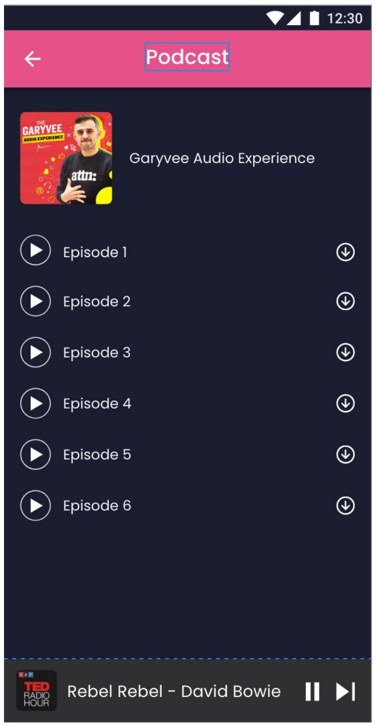
  - The screen will show up a episodes that a user downloaded, the can be able to listen offline

Podcast Detail Screen


12:30
Podcast
Garyvee Audio Experience
Marketing tips and business talks
5:53
Garyvee Audio Experience
Marketing tips and business talks
5:53
Garyvee Audio Experience
Marketing tips and business talks
5:53
Garyvee Audio Experience
Marketing tips and business talks
5:53
Garyvee Audio Experience
Marketing tips and business talks
5:53
Garyvee Audio Experience
Marketing tips and business talks
5:53
Rebel Rebel - David Bowie

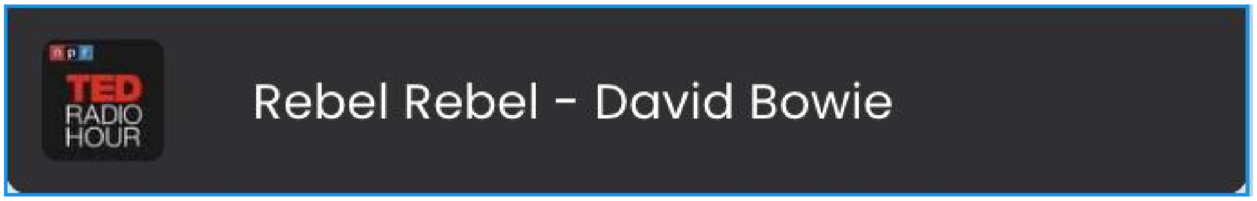
- The screen will show up click podcast icon on recyclerview in main screen.
- A user can set up favourite podcast by love icon. The app save podcast information and show up in favourite fragment.

Player Screen



- The screen will show up click episode icon on recyclerview in detail podcast screen.
- A user can download audio by download icon, app will download a file and save data in database. The data will show up in library fragment.

Music App Widget.



The widget will show the audio that user listening

Key Considerations

How will your app handle data persistence?

I will save user data in Sqlite database by using room persistence library. I will also use RxJava and Retrofit for getting podcasts information from api.

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

Glide to handle the loading and caching of images. I will use MVVM pattern and Android Clean Architecture for api function and save data in database. I will use dependency injection library to avoid boilerplate code. I will use Rxjava for reactive purposes.

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

I will use firebase auth sdk , facebook login api and google login api for save user data in cloud database.

What will happen when there is no internet connection?

I will create check network status function and call this function before api calling function. If there's no internet connection , I will show no internet connection message and get cache data from database and show up on UI.

What will happen if there is no response from the API and how the app is going to handle it?

- We need to add an interceptor to the retrofit service (which will intercept all the api responses) and let the UI know if a request has resulted in error.

- We need to update the UI once an error occurs.
- if there's something wrong with api calling function , I will check a data in database and if a data in database , i will show up on UI.

How the app is going to show generic navigations between the activities.

- I will use Android Studio Navigation Editor for fragment and activity transaction. I create a navigation graph and make your whole app on a single activity on which fragments will be moved.

How will your app handle data persistence?

App will use a Content Provider and a Shared Preferences to maintain the local data.

Describe any corner cases in the UX.

Unstable or missed network connection: the application must not crash in that cases

Device orientation change: the application must handle all long- running operations correctly considering possible configuration changes

UI freezes: the application must not use the main thread for any resource consuming operations

Add that an application will be written solely in the Java Programming Language.

- **I will use java 1.8 version for this project and Android Studio 3.5**

How resources will be stored in the project including colors, strings, and themes.

- **I will be stored all color code in color.xml file in res/values folder**
- **I will be stored all strings in sting.xml file in res/values folder**
- **I will be stored all themes style in style.xml file in res/values folder**

Use of at least one of the following APIs - SyncAdapter/JobDispacter or IntentService or AsyncTask.

- **I will use IntentService for download mp3 file and store in folder.**

How your app supports accessibility.

For app supports accessibility

- All strings are kept in strings.xml file
- All font sizes , margin size are kept in dimebs.xml file
- All color code are kept in colors.xml file

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

Dagger 2: for dependency injections
Retrofit 2: for network API requests
Glide: for images loading
Firebase: for analytics and crash reports
RxJava2: for reactive programming
Room: for save data in sqlite
LifecycleExtension: for Viewmodel and livedata components
Gardle : 3.4.1
Android Studio 3.5

Java Version 1.8
AndroidX Library

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

1. Add dependencies in gradle (firebase, retrofit, glide, dagger2, room, rxjava)
2. Create Retrofit service interface class
3. Create Room Database class
4. Create NetworkBound and Resource class for clean architecture.
5. Create dependency injection module and component
6. Create ApiRepository and DatabaseRepository class handling data.

Task 2: Implement UI for Each Activity and Fragment

1. Build UI for Splash Screen
2. Build UI for Login Screen
3. Build UI for Genre Screen
4. Build UI for Onboarding Screen
5. Build UI for Main Activity
    a. Create New Feeds fragment
    b. Create Search Fragment
    c. Create Favourite Fragment
    d. Create Library Fragment

Task 3: Implement Api calling function

1. Configure Facebook login api
2. Configure Google login api

Task 4: Create Splash Screen

- Create layout
- Create a function for check login function

Task 5: Login Screen

- Create layout
- Create facebook and google login function
- Create get all genres from api and save in database function
    - Create Espresso api test function

Task 6: Genre Screen

- Create layout
- Create a function for retrieve genres from database
- Create Recyclerview adapter
- Create get all genres from api and save in database function

Task 7: Onboarding Screen

- Create layout
- Create a function for get podcast data genres from api and save in database

Task 8: Main Screen

- Create Main layout
  - Create 4 fragments as New Feed, Favourite , Search and Library
    - New Freed Fragment
      - Create function for get recommended episode information from api and show in recyclerview
      - Create genre recyclerview and retrieve data from database and show in Recyclerview
      - Create function for get curated podcasts information from api and show in Recyclerview
      - Create function for get popular podcast information from api and show in recyclerview
    - Search Fragment
      - Create search from api function by user input
      - Create Recyclerview for show up search data from api
    - Favourite Fragment
      - Create a function to get favourite data from database and show in recyclerview
      - Create a function for remove favourite function
    - Library Fragment
      - Create a function to get downloaded data from database and show in recyclerview.
      - Create a function for delete downloaded data function

Task 8: Podcast Detail screen

- Create layout
- Create a function for retrieve podcast information from api by podcast id
- Create Recyclerview and show up the data from api
- Create favourite function
- Add click item function to navigate to player

Task 8: Audio Player screen

- Create layout
- Create exo audio player
- Create download audio worker class

- Create save episode information function
