GitHub Username: MahmoudMohsenSoliman

SeeMoV

Description

SeeMoV presents popular, top-rated, upcoming and now-playing movies. Also, you can store your favorite movies and retrieve it later.

Intended User

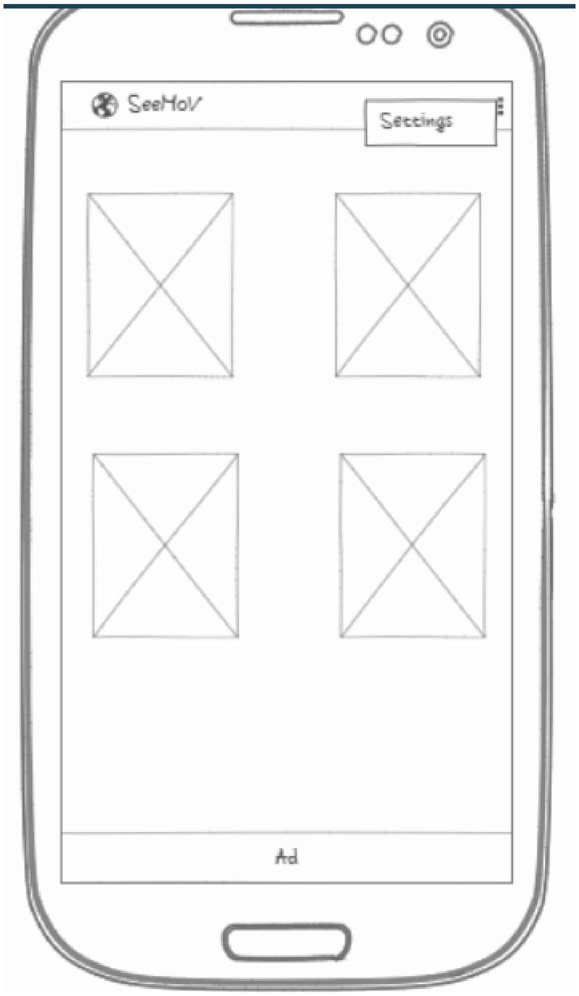
SeeMoV is intended to the movie audience.

Features

- Saves favorite movies
- Displays the favorite movies on the widget
- Fetches movies from TMDB
- Uses Ads
- Navigates between upcoming ,popular ,top-rated and now-playing movies

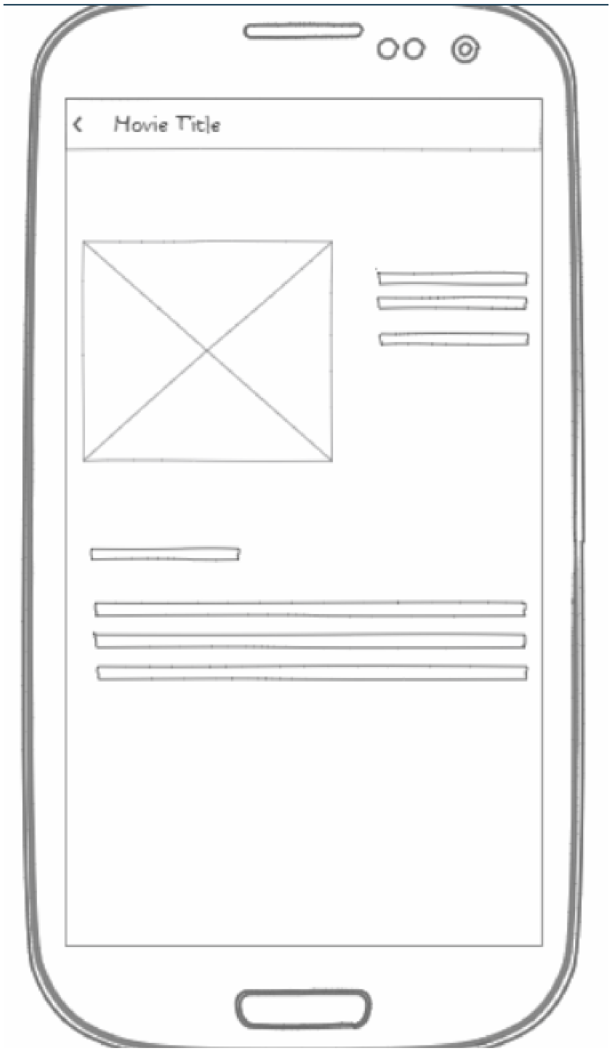
User Interface Mocks

Screen 1



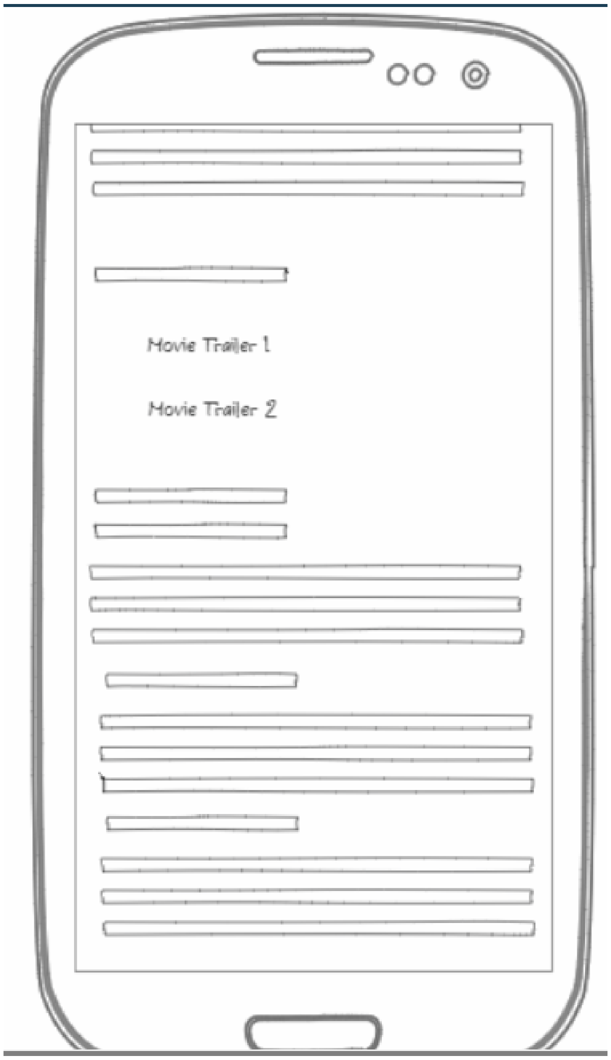
Main activity to present the selected type of movies. After pressing the settings menu,
It goes to the settings activity.

Screen 2



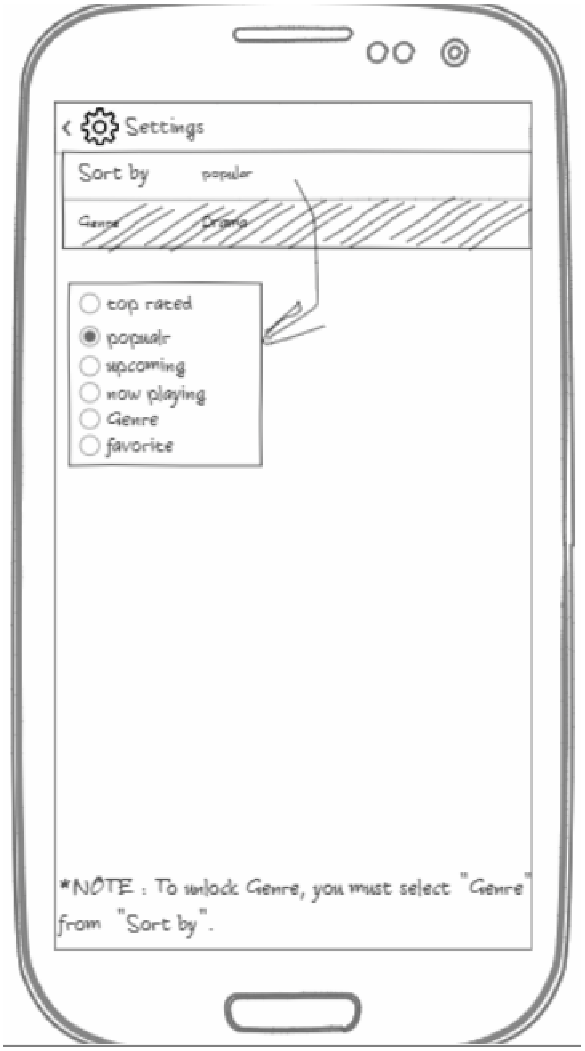
After clicking a movie, display the movie detail activity which have the Details of the movie, trailer and reviews.

Screen 3



After Scrolling the screen, this is what will be.

Screen 4



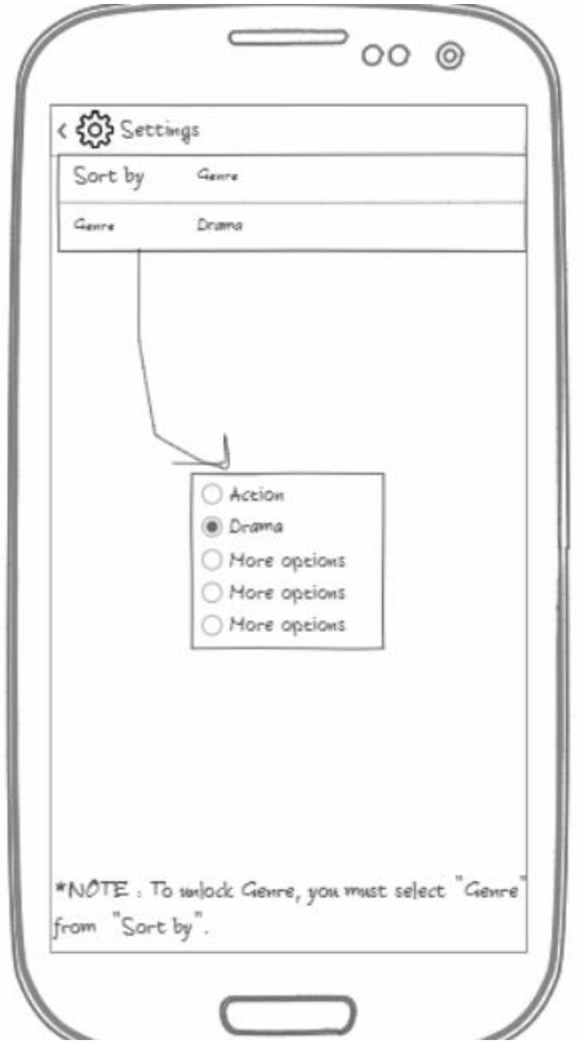
Sort by popular

Genre Drama

*NOTE : To unlock Genre, you must select "Genre" from "Sort by".

Settings activity controls what the main activity will show according to what is chosen in “Sort by” settings.

Screen 5



Sort by Genre

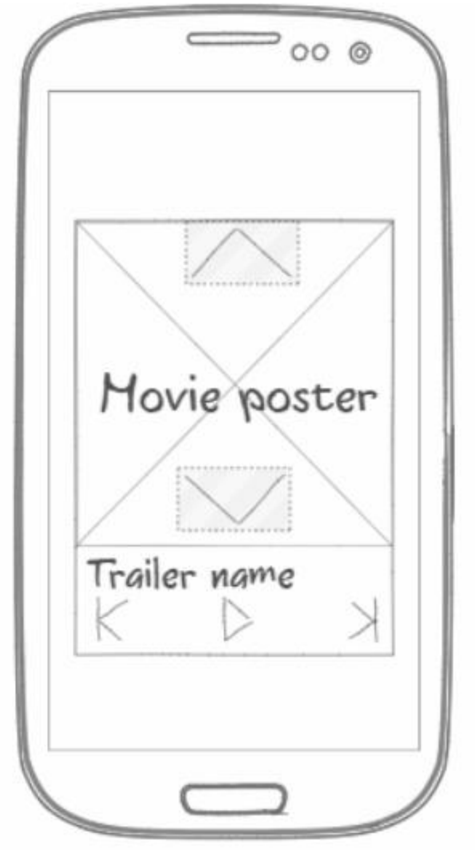
Genre Drama

○ Action
● Drama
○ More options
○ More options
○ More options

*NOTE : To unlock Genre, you must select "Genre" from "Sort by".

If the user selects “Genre” from “Sort by” settings, the “Genre” settings unlocks and the user can Choose which genre to be shown in the main activity.

Screen 6



The widget presents the favorite movies. The user can navigate through his favorite movies and he can change what trailer he wants to play.

Key Considerations

How will your app handle data persistence?

I will use Room persistence library.

Describe any edge or corner cases in the UX.

When the user rotates the device, the state will be saved using shared preference.

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

Library	Version
Picasso	2.71828
ButterKnife	10.2.0
Gson	2.8.5
Retrofit	2.6.1
Gson converter	2.6.1
Room	2.1.0
Google Mobile Ads	18.2.0
Firebase Analytics	17.2.0

Picasso to handle the loading and caching of images. ButterKnife for replacing findViewById calls. Room as a data persistence to store the favorite movies.

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

This app will be backed by the Listen API which is The MovieDB API. I will pull the JSON data from the API using Retrofit and Gson to populate results. The app will also use AdMob to display ads in the free build variant. The paid build variant will not display ads. In addition, the app will use Firebase Analytics to implement screen tracking. To retrieve from room database, I will use an AsyncTask to launch Room library's methods.

Next Steps: Required Tasks

Task 1: Project Setup

- Add libraries to the app level build gradle.
- Create activities for each screen.
- Write the app in Java.

Task 2: Implement UI for Each Activity and Fragment

List the subtasks. For example:

- Build UI for MainActivity.
- Build UI for movie details activity.

- Build UI for settings activity.

Task 3: Connect to the internet

- Create a class for Retrofit calls and add Gson converter factory.
- Create an Interface for the API Endpoints.
- Add “SerializedName” annotation of the Gson converter on Movie member variables each member variable annotated with the corresponding name of the JSON Object to store its value in that variable.
- Create a Call of Movie variable to enqueuer calls of the endpoint.
- Validate if the data is retrieved from the api correctly.

Task 4: Build Favorites Database

- Create entities for favorite movie entities.
- Create DAO for favorite entities.
- Create Database class that implements Room.
- Use LiveData and ViewModel to view favorite list.

Task 5: Implement Firebase services

- Create free and paid build variants.
- Use Admob to add ads to the free build variant.
- Use Firebase Analytics to implement screen tracking

Task 6: Build widget

- Create layout for the favorite movies.
- Create WidgetProvider and RemoteViewsServices classes.
- Build content provider to supply the list of the favorite movies to the widget.

Task 7: Implement accessibility support

- Implement content descriptions, navigation using d-pad, and other accessibility features
- All strings are kept in a strings.xml file

Task 8: Handle error cases

- Create tests using Espresso to make sure all UI connections are working correctly.
- Test AdMob ads