

[Description](#)

[Intended User](#)

[Features](#)

[User Interface Mocks](#)

[Screen 1](#)

[Screen 2](#)

[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.](#)

[Describe how you will implement Google Play Services.](#)

[Next Steps: Required Tasks](#)

[Task 1: Project Setup](#)

[Task 2: Implement UI for Each Activity and Fragment](#)

[Task 3: Setup API key](#)

[Task 4: Create helpers](#)

[Task 5: Database and other functionality](#)

[Task 6: Testing](#)

**GitHub Username:** amardeepkumar

## Movie Diary

### Description

If you are a movie freak and you want to save the details of movies which you like and which gave you memory with your friends or family then "Movie Diary" is for you. It helps you to browse and make a watchlist. You can search movies and watch trailers. It allows you to add movie to favourite list. You can mark your favourite movie as "Seen details" or save the details for watch later.

### Intended User

This app is for all the movie lovers

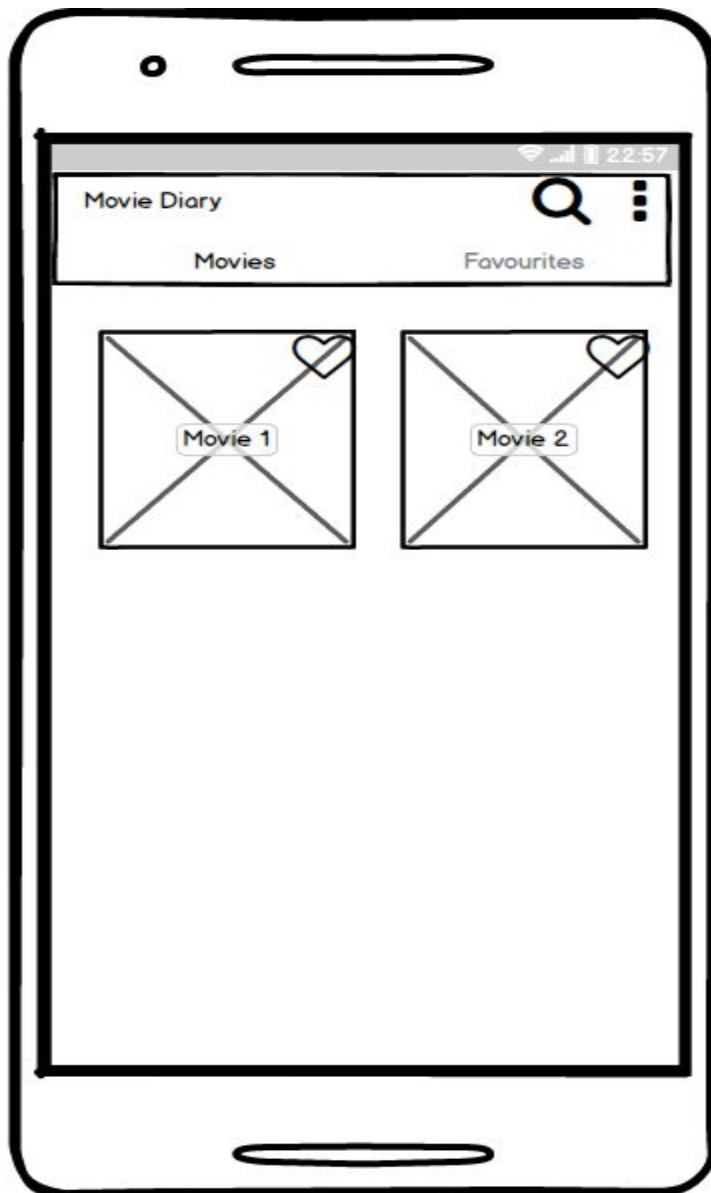
### Features

- Search movie

- Sort movie list based on popularity, title, release date and rating
- Make a favourite list with personalized details like Theatre name, Location, Friends name, Date
- Check the movie details with trailers, reviews and release date.

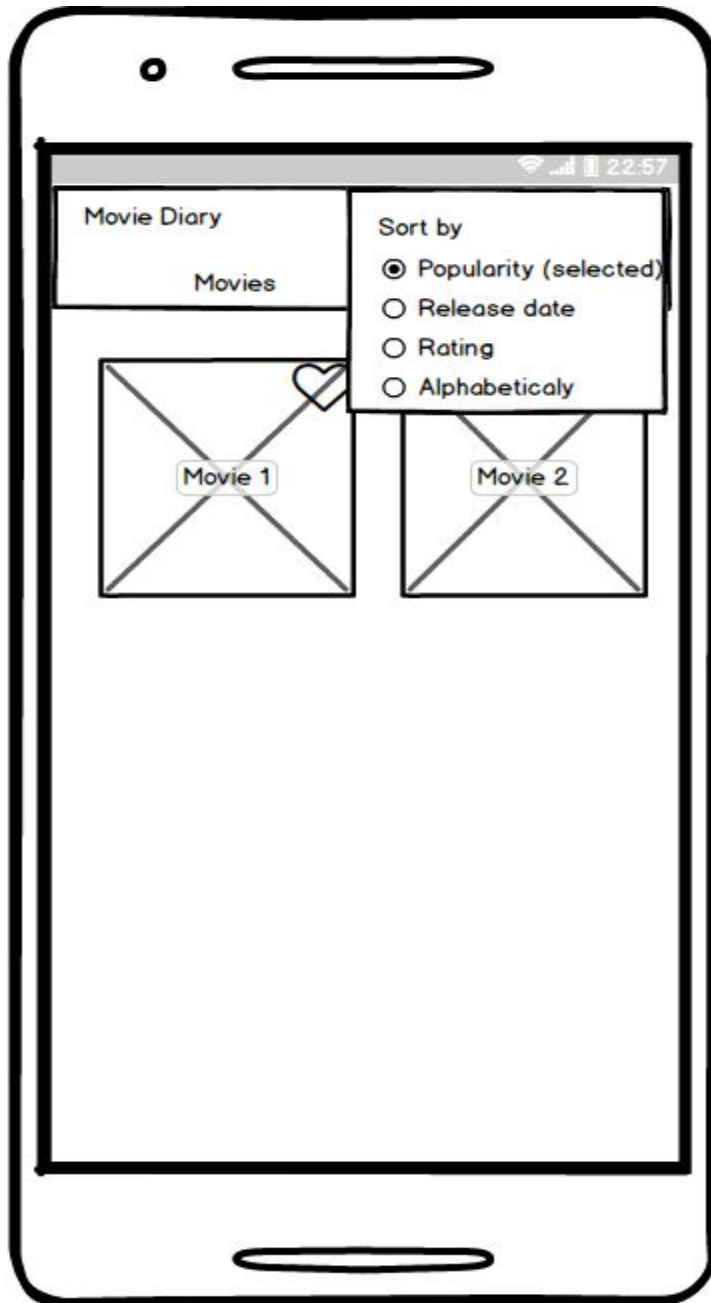
## User Interface Mocks

### Screen 1



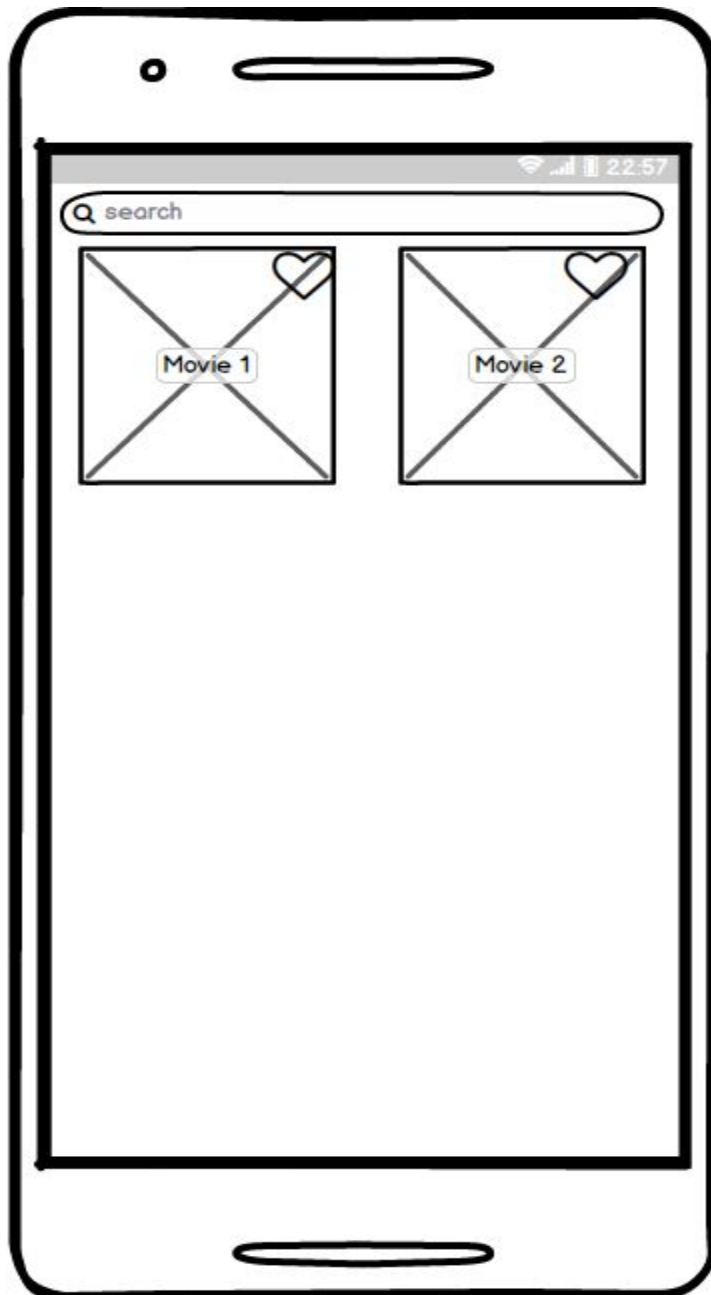
This is home page will show the movie list based on user selection in sort order.

## Screen 2



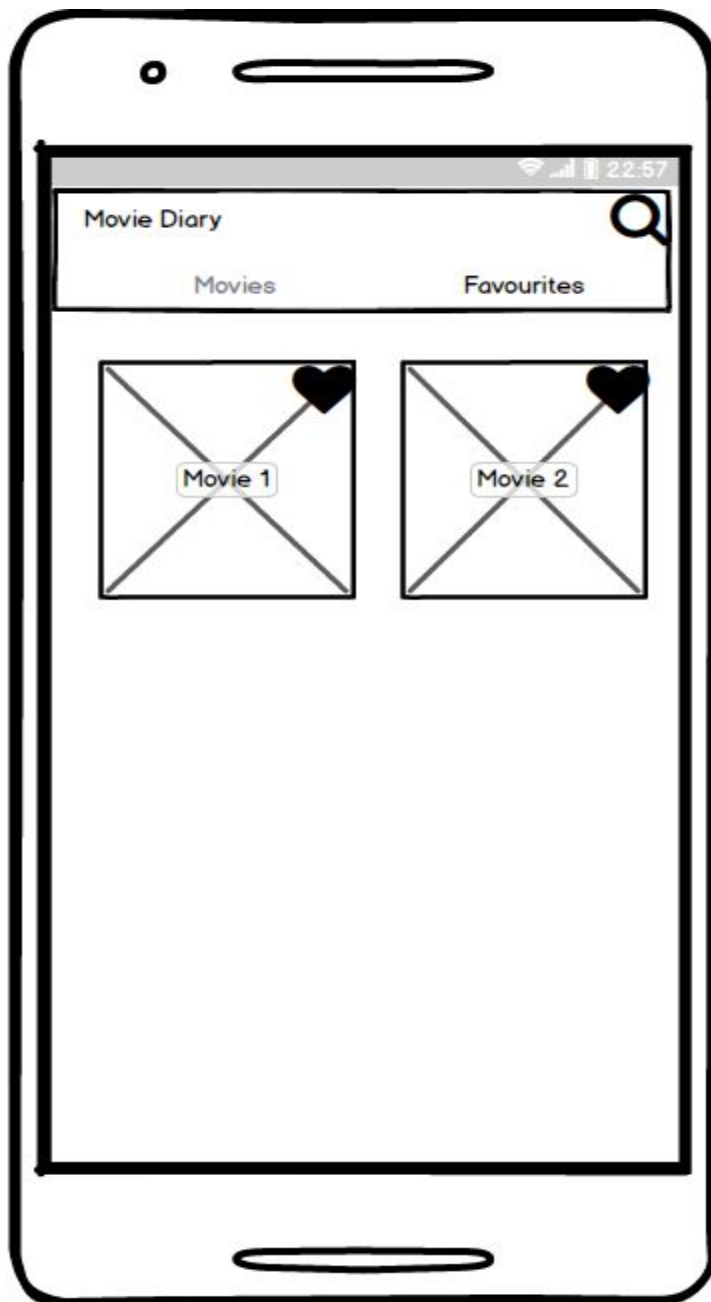
Sort options for the user

### Screen 3



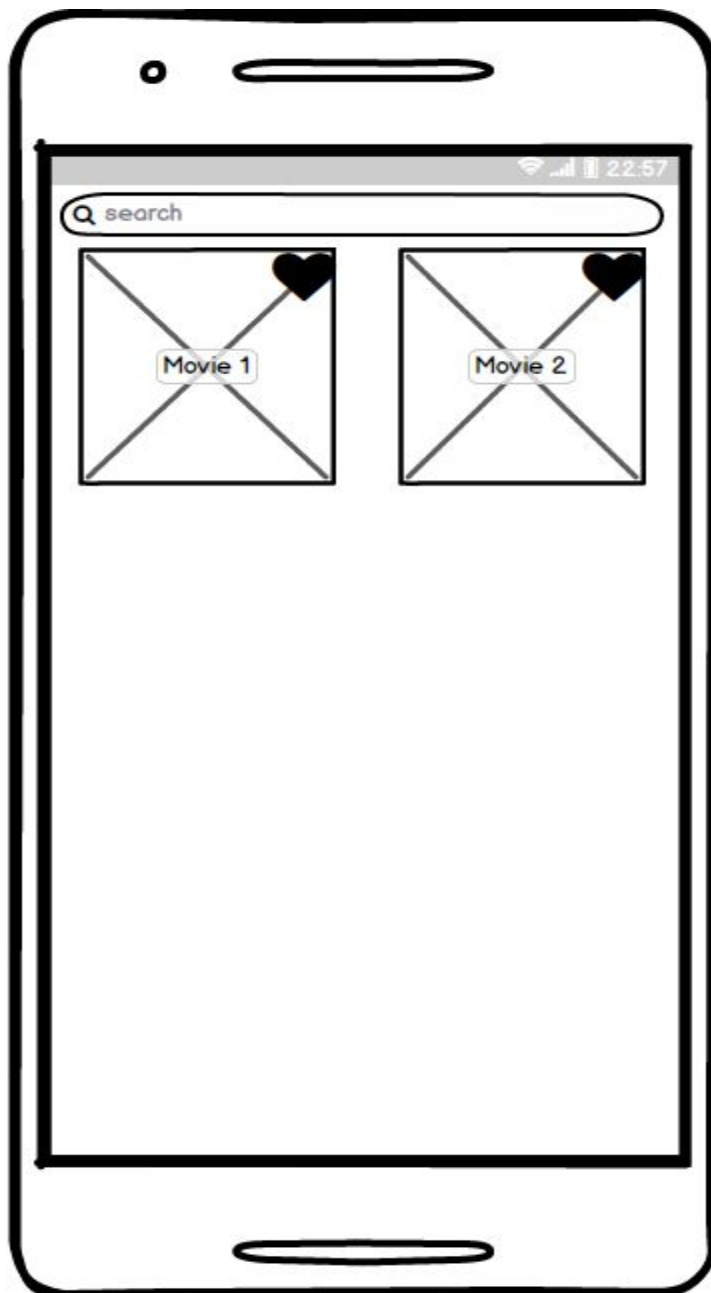
Search any movie by name.

#### Screen 4



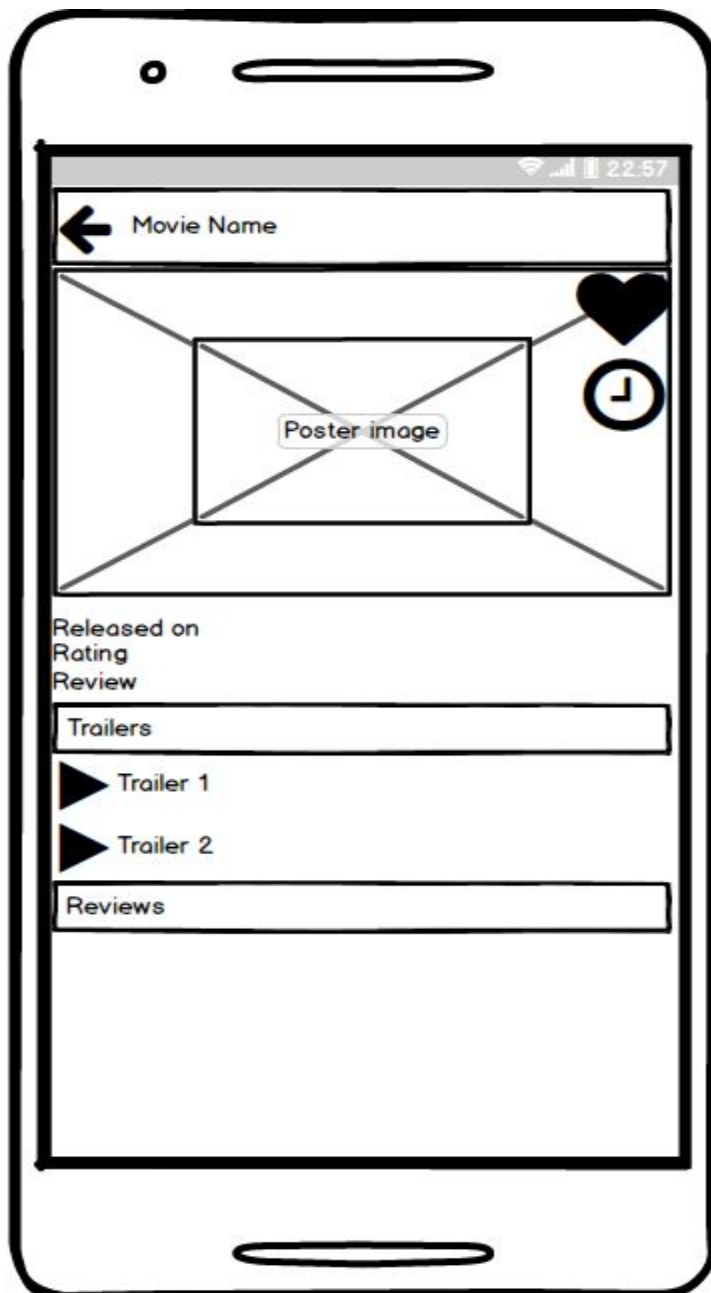
Favourite tab selected

## Screen 5



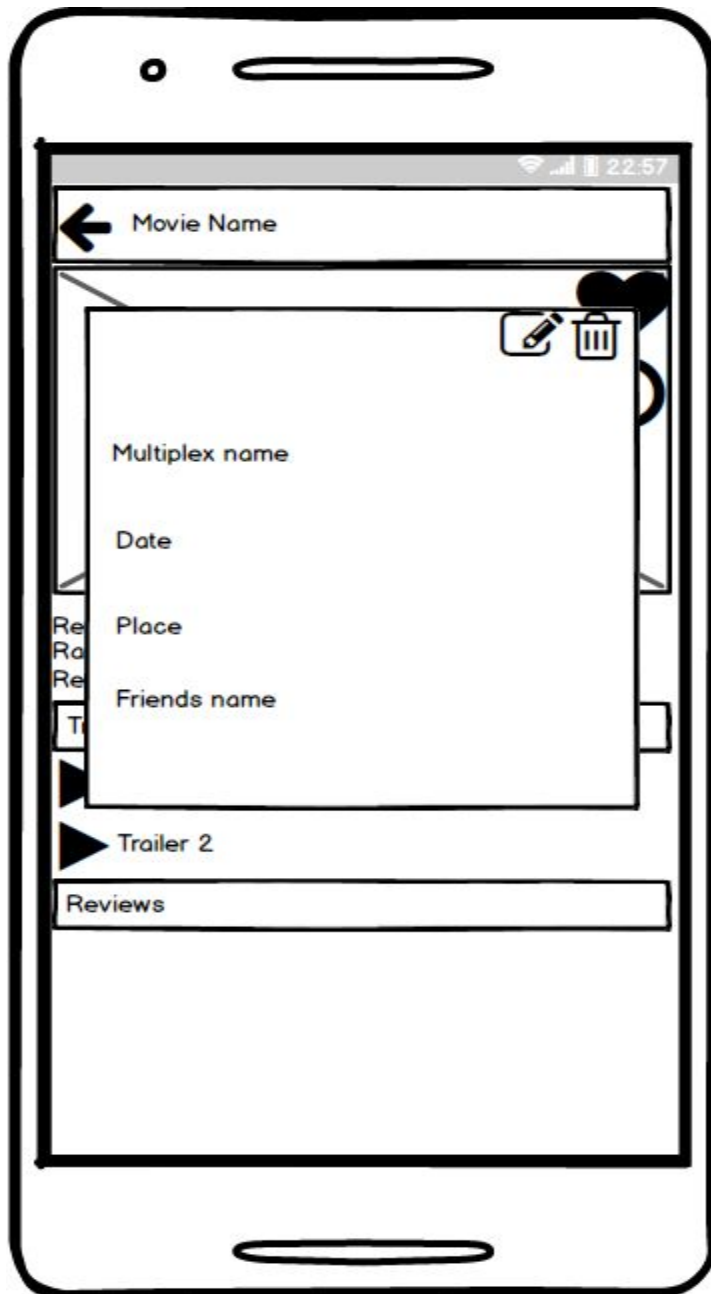
Local search for favourite movie list. User can search by name.

## Screen 6



Movie details to show all the information related to movie.

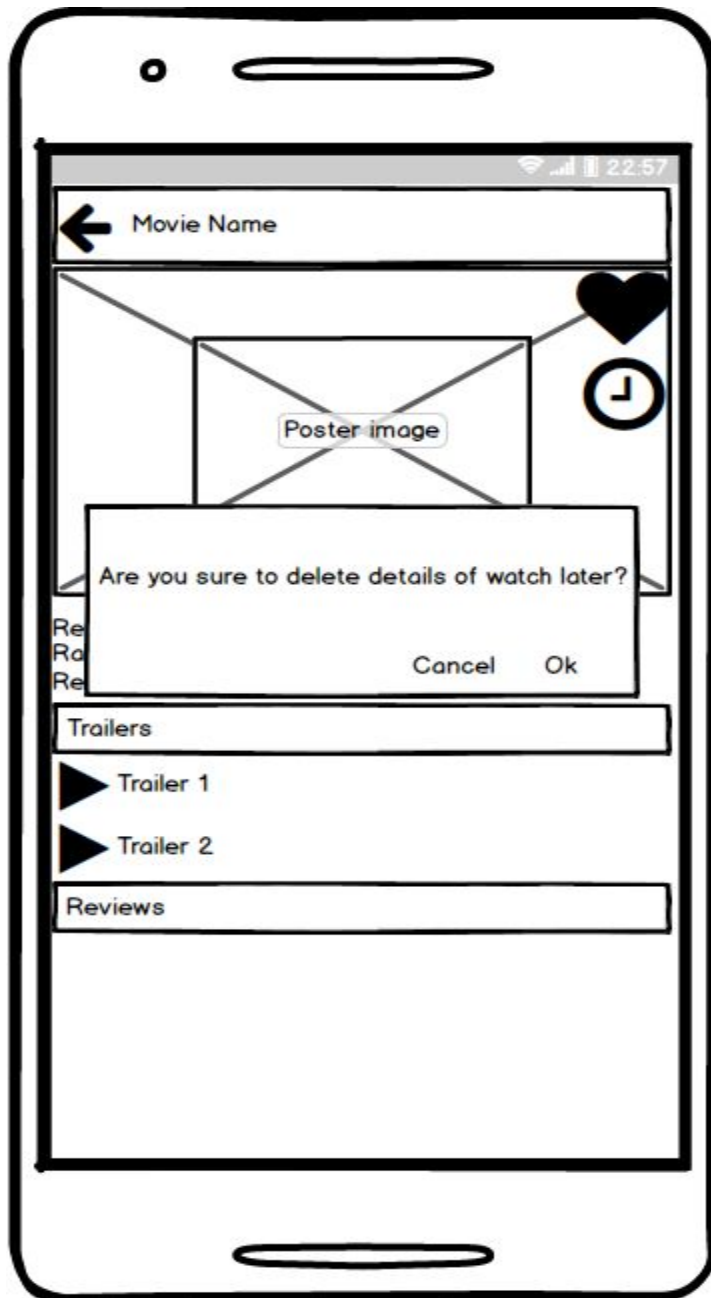
## Screen 7



Dialog screen to show the information of time clock click action.

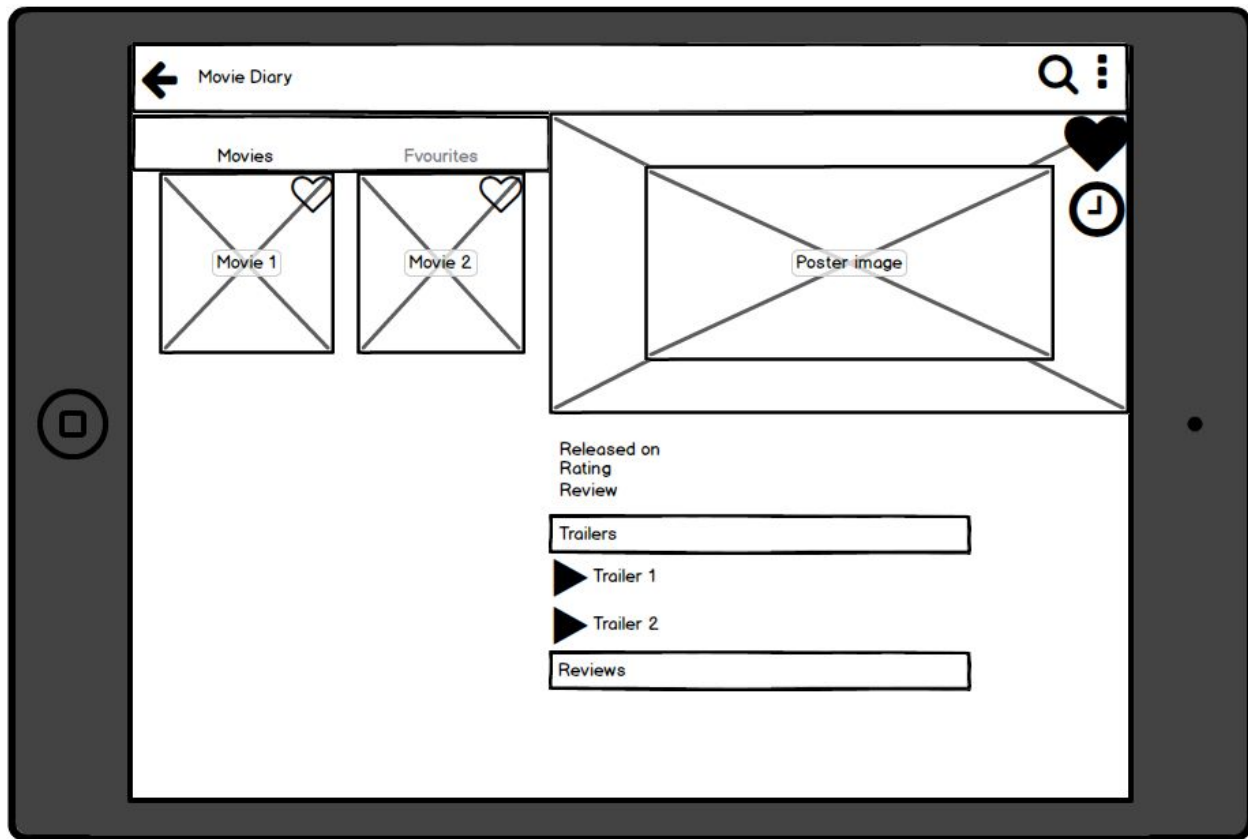


## Screen 8



Dialog to confirm from user before deleting the details of watch history.

## Tablet view



## Key Considerations

### How will your app handle data persistence?

App will use movie db API to show the details. Once downloaded the response will be saved in database using. View can query to this database using a content provider. User preference of sort by will be saved in shared preference.

### Describe any corner cases in the UX.

User can delete, edit or add the watch history in details screen. These data will be saved in database. Pagination will be supported in listing screen(Home screen). Offline support will be added.

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

Glide: For image loading

Retrofit2: For API call

Data Binding: To implement MVVM architecture

GSON library: to parse the response

Google support, appcompat and design library.

**Describe how you will implement Google Play Services.**

Google Analytics using GTM for tracking user action in app.

Google Add to show some ads.

## Next Steps: Required Tasks

### Task 1: Project Setup

- Configure libraries
- Add Base Activity, Base Fragment and Base Application
- Create MainActivity
- Add min and target sdk version

### Task 2: Implement UI for Each Activity and Fragment

- Build UI for MainActivity
- Build UI for Tablet layout
- Build UI for Movie details
- Build UI for watch history dialog

### Task 3: Setup API key

- Add API key to gradle.properties
- Use API key from build.gradle

### Task 4: Create helpers

- Create a Network Manager

- Create Preference Manager
- Create Databinding Util
- Create constants, utils and common methods

## **Task 5: Database and other functionality**

- Implement content provider and database manager
- Error handling
- Support accessibility
- Support RTL
- Widget for the app

## **Task 6: Testing**

- Unit test each screen with internet
- Unit test each screen without internet
- Test on tablet device
- Test for all error cases and validation
- Test for orientation
- Test for accessibility
- Test RTL support
- Test for any crashes