Capstone Project: Movies On Demand

GitHub Username: **ngaruko**

Movies On Demand

Description

With modern advances in technology, the good old "brick-and-mortar" DVD rental store does not make any sense. I mean, who needs to drive a mile to the store, spends an hour walking up and down the store aisles looking for a movie that he could get from the comfort of his home. for a fraction of a price and in better quality than the often scratched DVDs we get from the DVD store. This app helps movie lovers, not only browse what is available on the market and in theatres, but also to watch trailers for free and full movies for a fraction of the DVD cost either on subscription basis or on a pay-as-you-use model.

Intended User

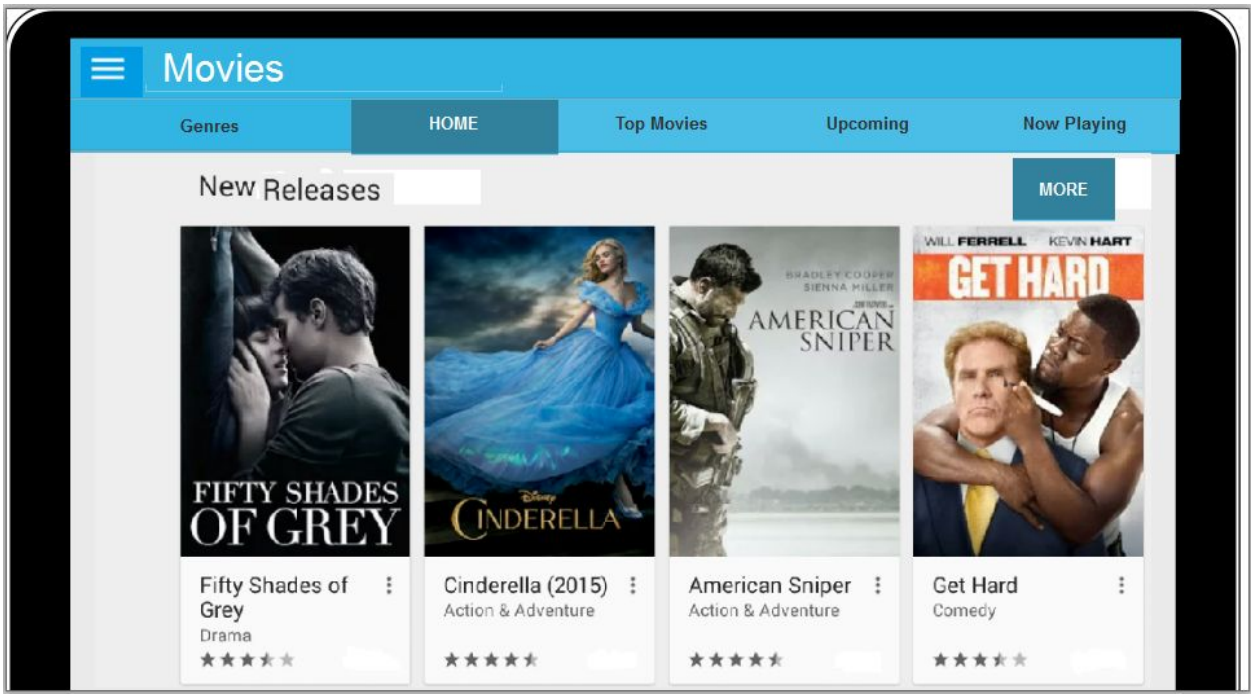
This app is for the typical "modern citizen" who needs family entertainment at their fingertips.

Features

- Downloads movie details
- categorises movies by genres
- Lets users view trailers from the internet
- Allows subscribed users to watch movies or download

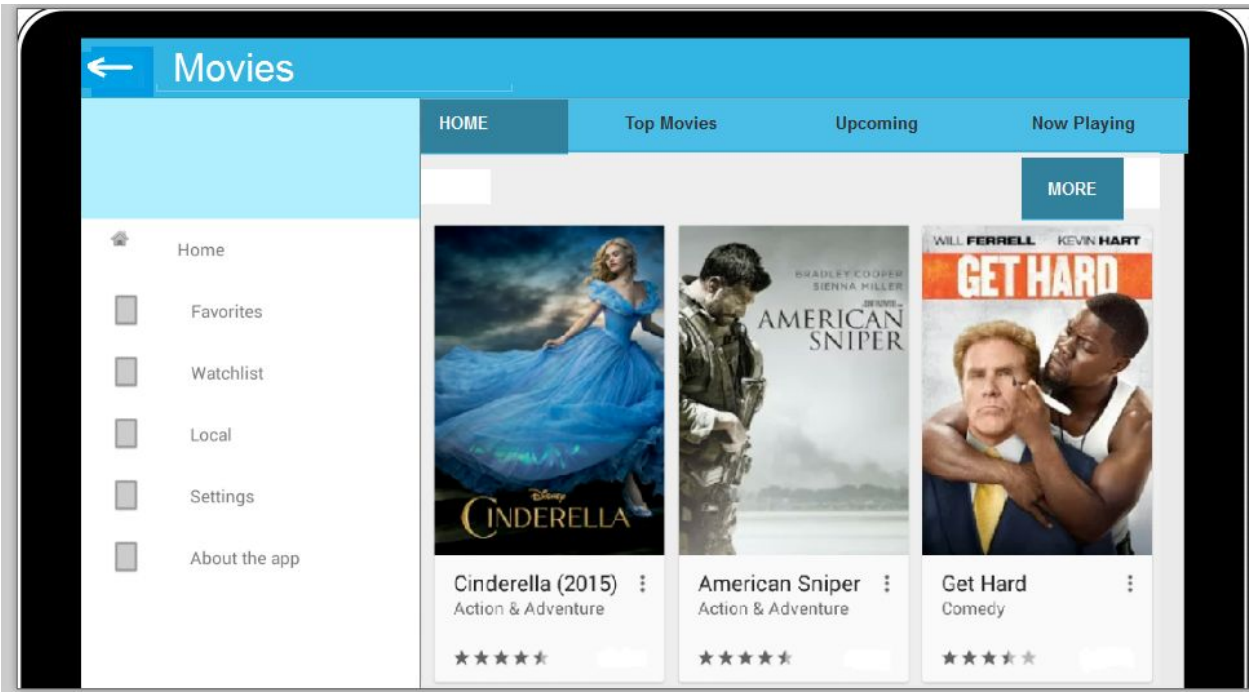
User Interface Mocks

Home Screen: Movies



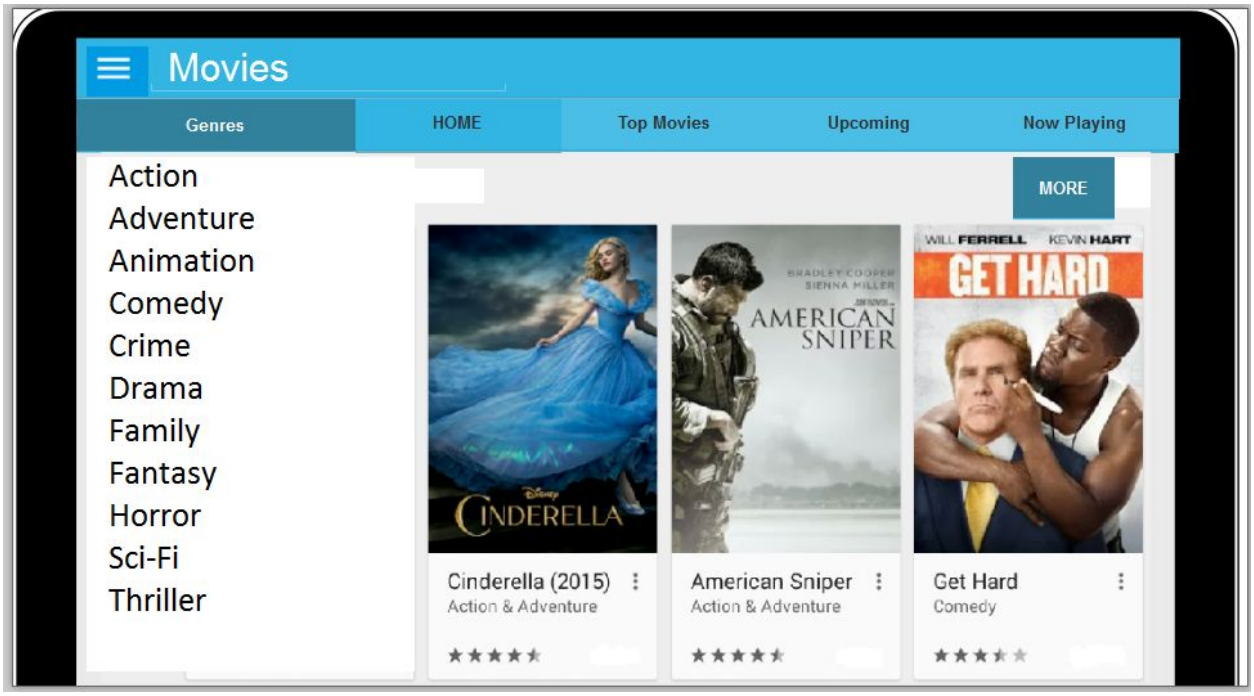
This is the home page, it shows a few sample movies in each category with link to more movies. The first screen displays samples movies in all categories and provides tabs for top movies, upcoming and now playing.

Navigation drawer



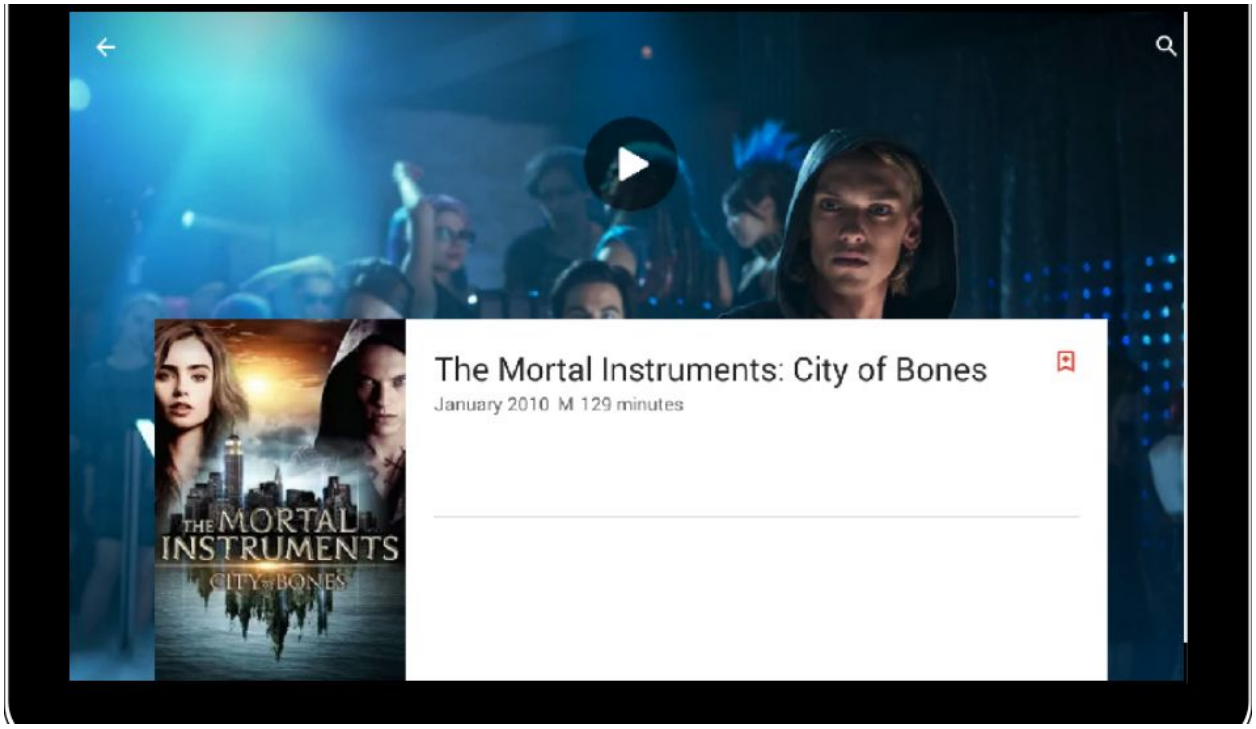
There is also a navigation drawer that allows the user to go to the home screen and gives them more option for a personalised experience

Genres



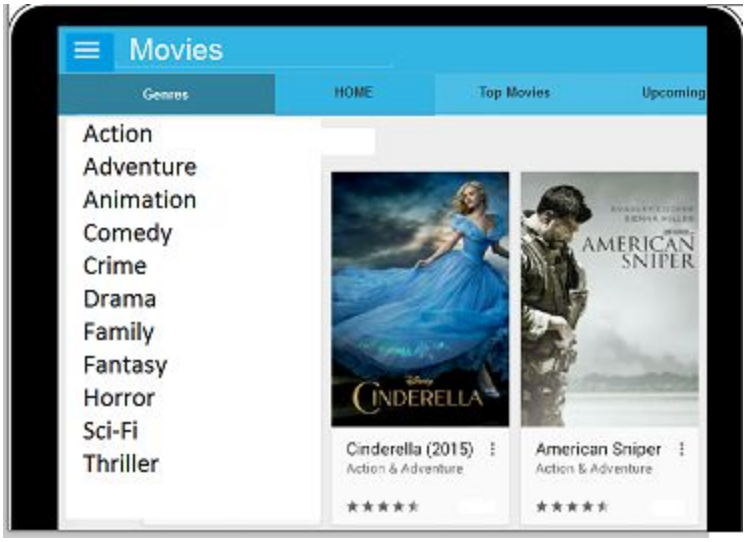
The user can select movies of different genres from action to drama to Sci-Fi.

Selected Movie

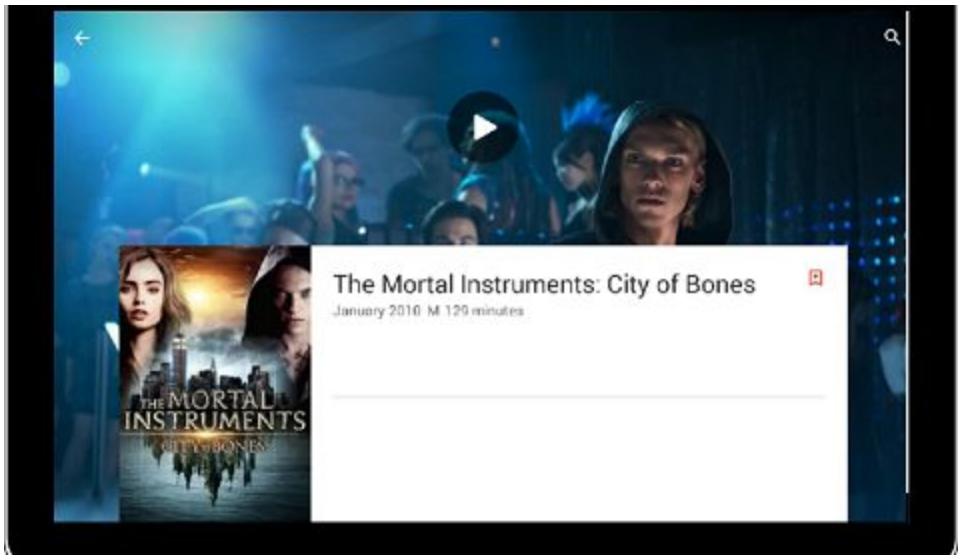


On movie selection, the user is presented with a full screen view with options to view trailer, view full movie or download...

Phone_Genres



Phone_Selected



Key Considerations

How will your app handle data persistence?

The uses a content provider and a sqlite database.

Describe any corner cases in the UX.

Upon launching the app, the app displays the Movies screen with “in-theaters” movies. Tabs allow for other movies’ option selection. To go to games, music and TV shows, the user clicks the app icon or swipes from left hand corner to open the navigation drawer.
When a user selects a movie, he is given options to watch it or download it (of course if their subscription allows for it).

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

The app uses Volley library to handle and cache images. This app deals with high-resolution compressed images and Volley handles these very well. Plus Volley does more than image loading, it is part of the app’s backend.

Next Steps: Required Tasks

Task 1: Project Setup

- search for (a) suitable api provider(s) for movies
- get required subscription
- design the application layout
- add required libraries

Task 2: Implement UI for Each Activity and Fragment

- Build UI for MainActivity

- Build UI for navigation drawer
- Build UI for fragments:
    - in theaters
    - top movies
    - coming soon
    - trailers
    - music
    - tv shows
    - games
- Build UI for widget
- Build UI for settings Activity
- Build UI for settings fragment

Task 3: Database Design

Design database model for the app:

- database contract
- database helper
- content provider

Task 4: Volley library

- set up networking with Volley
- get data from API providers (Download JSon feed)
- parse JSon data
- Volley error handling

Task 5: RecyclerView

Describe the next task. List the subtasks. For example:

- Set up adapter
- set up viewHolder
- Display data to RecyclerView

Task 6 :Transitions and animations

- recyclerview item animations
- appearance animations
- Swipe to refresh
- use android transition API

Task 7: Floating action buttons

- FABs for most common actions: sorting movies by release date or rating or just by title
- implement FABs java code

Task 8: Implement Google Play Services

- Admob
- Identity

Task 9: Accessibility

- content description
- TRL layout
- localisation

Task 10: add a widget for in-theaters movies nearest to user

- widget xml
- widget layout
- widget provider
- widget provider service
- widget configuration

Task 11:Add a job scheduler

- add a job scheduler service
- integrate the service with other components: db, views, Volley

Task 12: handle error cases

- Network errors
- Null pointers errors
- data format errors
- invalid data errors