**Watchlist App using Watchmode API**

***Outline***

Watchmode: https://api.watchmode.com/

The app’s primary purpose is to help users who subscribe to multiple streaming services to conveniently reference upcoming releases without having to visit multiple apps or find a website.

On opening the app, the user will see a list of the upcoming releases, and on which service the Title is available. The user will only see titles available on services in their region (user country identified by Android system).

A user can click on a Title in the list of upcoming releases and review film/programme details.

The app can be used as a Watchlist (‘My Watchlist’), if a Title appeals to the user, it can be stored in the app. The user can also remove Titles from their Watchlist once they have watched the title.

The list of upcoming releases will be downloaded from the watchmode API every day. Note that these are the US release dates but most services release original content on the same day in all territories.

API

This app will link to the Watchmode API. The API is a dataset that has registered which movies and programmes are available to be streamed from over 200 streaming services and in which region. It also holds detailed information about the content (including actors, directors, release dates etc). The API is organized around REST.

***Detail***

The user first lands on a launch fragment where two buttons are available; one will take the user to a list of upcoming releases and a second button will take them to their personal watchlist.

3 Features

1. Upcoming New Releases Fragment.

Titles are downloaded automatically for display in a vertical recycler view. The default GET request returns titles released today through 30 days.

The titles are saved down to a local Room database to a movie table. Each day the upcoming releases are refreshed and new incremental day’s worth of movies added.

A second API call retrieves streaming service provider information including region availability. The streaming services data is saved down to a Room database and is split out between Service and Region.

The streaming services are displayed across the bottom of the Upcoming Releases Fragment in a horizontal recycler. On opening the App, the Android system detects which country the user’s phone is currently and filters the streaming services accordingly (if the user disabled device location they can see all services).

The user is able to select which streaming services they are interested in, eg Prime and Netflix, this selection in the checkboxes will persist across Fragment navigation.

Option are available in a menu that can filter by date (see today’s releases, next seven days, all etc)

The user will only see a list of titles:

* Filtered by Timeframe (menu options)
* Filtered by Streaming Services available in the region (automatically determined by device location)
* Filtered Streaming Services selected by the user, checkboxes on the horizontal recycler.
* Recycler items shows : Title, Service, Type (movie/TV series), release date & whether the title has been marked to the user’s watchlist.

Cached data is available when there is no network availability.

1. Title Information. The user can click on one of the new releases in the recycler view and it will take them to the movie details screen which is accessed through an API call. These details are not saved down locally and are shown from the network (a toast will alert user that connectivity is not available).

The user will be taken to a fragment that shows them:

* Streaming Service logo
* Title poster image if available
* Year
* Critic Score
* Language
* Genre (chip group)
* Synopsis (scroll widget)
* Link to trailer if available
* Button to Add/Remove from watchlist

1. My Watchlist. This fragment shows a list of titles that have been marked by the user (in the title detail screen see 2.) as belonging to the watchlist. The ‘watchlist’ property on the movie/title object defines whether a title is a member of the watchlist. If the user clicks on the title a motion layout dialog invites them to remove the title from the watchlist, go to the details, or cancel the dialog. This fragment uses a shared view model with the Upcoming Releases fragment (see 1)

Navigation:

A nav graph enables users to navigate :

Launch screen directly to Upcoming Releases or My Watchlist

Upcoming Releases to My Watchlist directly

My Watchlist to Upcoming Releases directly

Network:

Retrofit

Moshi

Glide

Room database:

Movie\_table

Service\_table

Region\_table

Koin – is used for injections:

Share data source between viewmodels

Declare MoviesRepository and database

Workmanager – run download of upcoming releases while phone is charging and idle

**Permissions:**

Android.permission.INTERNET

Android.permission.ACCESS\_NETWORK\_STATE : allows app to access information about networks, for example is there a connection and is it Wi-Fi, mobile data

android.permission.ACCESS\_COARSE\_LOCATION

A diagram of a company

AI-generated content may be incorrect.

Milestones:

1. Design Document

UI layer

1. Base files – viewmodels and fragments, navigation command
2. Create main fragments, launch, upcoming releases and watchlist
3. Create shared view models and adapters

Data layer

1. Create data layer – repository, local data and nework data
2. Snag fixing
3. Submission

Images:

Launch

A screen shot of a phone

AI-generated content may be incorrect.

A screenshot of a phone

AI-generated content may be incorrect.

A screen shot of a phone

AI-generated content may be incorrect.

A screenshot of a phone

AI-generated content may be incorrect.

A screenshot of a phone

AI-generated content may be incorrect.

A screenshot of a phone

AI-generated content may be incorrect.

**Rubric Criteria:**

**Android UI/UX**

*Build a navigable interface consisting of multiple screens of functionality and data.*

Watchlist App: 4 screens/fragments including Launch, Upcoming Releases, My Watchlist and Title Details. Navigation controller is used for navigation. A bundle is used to transfer title information between the Upcoming Release Fragment/My Watchlist fragment to the Title Details fragment to enable the details to be called.

*Construct interfaces that adhere to Android standards and display appropriately on screens of different size and resolution.*

Constraint layouts are used for all layouts.

Recycler views used for data collections (Upcoming Releases (both vertical and horizontal) and My Watchlist).

*Res* files set up with Strings, colours, themes, drawable etc

*Animate UI components to better utilize screen real estate and create engaging content.*

Motion layout is used for the dialog action choice on the My Watchlist fragment. Dialog fades in over several seconds.

**Local and Network data**

*Connect to and consume data from a remote data source such as a RESTful API.*

Retrofit used to connect to watchmode API. Network container objects are converted to database objects (which in turn are converted to domain objects for UI display).

Dispatchers.IO thread is used for network calls.

Moshi used for parsing JSON responses.

*Load network resources, such as Bitmap Images, dynamically and on-demand.*

Glide is used for logos and posters with a placeholder backup.

*Store data locally on the device for use between application sessions and/or offline use.*

**Android system and hardware integration**

*Architect application functionality using MVVM.*

MVVM pattern is used including observers for live data.

Fragments control views with data binding.

View model holds logic

Data layer connected to view model via a repository under which sits local and network data.

Network, database and domain objects are used in relevant locations.

*Implement logic to handle and respond to hardware and system events that impact the Android Lifecycle.*

Filtering information is stored across the fragments eg streaming service checkboxes. Shared view model is used between Upcoming Releases and My Watchlist fragment.

*Utilize system hardware to provide the user with advanced functionality and features.*

Location permissions are requested on opening the Upcoming Releases fragments (although if a user refuses the app will simply show unfiltered data)