Coursework Report - Movie Seeker

Andrea Silvestro Ortino
40270115@live.napier.ac.uk
Edinburgh Napier University - SET08114

Abstract

A movie application using basic concepts for the coursework of the Mobile Development Module at Edinburgh Napier University. Movie Seeker lets the user show popular and most rated movies, as well as creating a favourite list. It is mainly used with a network but maintains some offline functionalities because favourites movies are also stored locally through a SQLite database. The data content is provided by TMDB which offers its API key by registering to their website.

1 Introduction

This is an android project for the coursework of Mobile Development at Edinburgh Napier University. I was supposed to develop an app from scratch without any particular restrictions and with the goal of going further the treated topics during the lectures.

I have chosen to do a Movie Seeker application with the intent of going deeper in the knowledge of using an API service to retrieve data and store it also locally when necessary.

There are many apps that use TMDB [1] as API service and I wanted to develop one as well, mainly for my own needs.

I have followed the guidelines of the workbook of the module [2], as well as Android Recipe [3], Android Developers [4], and Tutorials Point [5].

Movie Seeker takes mainly inspiration for what concerns functionalities and UI from 'Cinemapp' [6] that can be found in the Google market store. I adapted it to my own competence breaking it down in too simpler concepts to implement. In addition, I have also found very useful the developer page of TMDB [7] which has a lot of ideas and ways to implement the multitude of possibilities achievable with its search tool and database information.

My app was supposed to show the users Popular and Most Voted movies, and to let them check and uncheck their most favourite movies retrieving this information both online and offline. Another feature that I wanted to implement was a way to show information about a singular movie by clicking on it in the gridview along with random search with input filters.

2 Software Design

Movie seeker has a clean and simple interface. It is divided into different activities, but I have decided for an easier user's experience to use fragment activities inside containers. This allows to process and show different parts of the app at the same time, without the necessity to click or to open new activities.

The main element of the fragment is a gridview contained in a scrollview. The gridview is made of poster's image which are clickable to show movies' information.

While the users interact with the main screen, some processes are done in the background using AsyncTask's class extension, and Listener's object and methods for catching user's inputs either on the gridview or on the settings.

Finally, the app shows a different number of box images based on the screen's size, in order to accommodate both tablets and smartphones; each box maintains a light design with the use of Picasso to always re-size the images' height by 1.5 of the width[8][9].

3 Software Implementation

The MainActivity of Movie Seeker has the only purposes of checking for screen dimension, inflating the menu, and being a container for the MovieFragment activity. Indeed, it is in the fragment that takes place most of the processes of the app.

Starting with the layout, it can be seen from the fig.1 that the main element is a Scrollview containing a gridview that is inflated with boxes of movies' images.

Each box is clickable and will open the MoviesDetailsFragment activity (inside a container activity "MovieDetails") showing the main information of the selected movies (fig.2).

This is achieved by using the TMDB API [7] queries with an open connection URL that returns data in JSON format which are then parsed into strings for retrieving the movie's path URL of the posters and the movie information. These elements are stored in different Arraylists divided between the type of information for either favourite movies or not. Bool variables are used to check the type of queries based on the Settings preferences "sortby" with the possibility of searching by popularity, votes, and preferred movies (fig.3).

Favourite movies are stored, retrieved, and updated using SQLite database which is set up and created in the StoreDB

Figure 1: **Main Screen** - Showing GridView and ScrollView

[10] class which extends the class ContentProvider. Movies are added to the DB only if are checked as favourites.

Users' favourites are registered in an ArrayList of Boolean (MovieFragment Class) with the method "RegisterFavorites" whose elements are all set to false at first and changed only when a movie is present in the titleListF ArrayList. The method is then called on "onCreateView" if the "sortby" is not based on favourites, so it is possible for the user to check whether a popular movie is already on its favourites list.

The elements of the lists are put as extras into the MovieDetails activity so individual components can be retrieved by the MovieDetailsFragment's OnCreateView method and the view inflated with data.

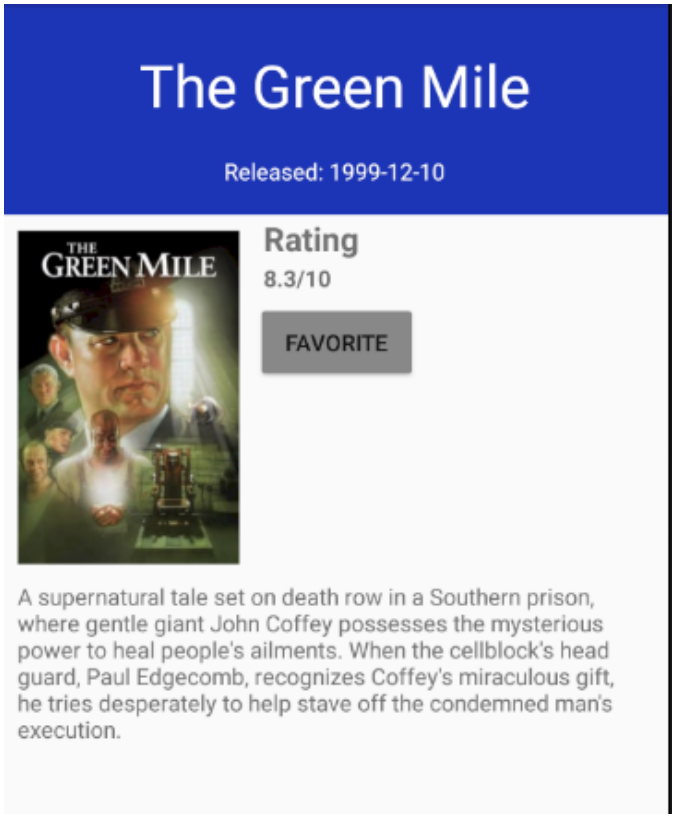

Figure 2: **Movie Details** - Showing Movie information

4 Critical Evaluation

I managed to implement most of the features that I decided my app should have. However, I did not manage the time at my disposal at best for the final additions, and the random search along with the implementation of more search filters had been cut out for now.

I must say that compared to the applications that are in the market, Movie Seeker is really basic both for the graphical interface and user's interaction, with not as many functions and personalisation as them.

This is due again to the time at my disposal. For this project, I have spent most of the time learning how to develop and implement the main concepts of the app, but I had not enough experience to deliver a competitive app. It does not show movies by genre, as I wanted or statistics of the user's favourites and watched movies, but all of these missing details will be a start point for future developments.

5 Personal Evaluation

I had barely any experience in mobile development and I was not into applications before this project, but I was particularly surprised how it was the whole learning experience. Even though I had to do a lot of research for my lack of knowledge, I have found a new world full of possibilities and adaptability.

Android Studio is a simple platform that let you jump straight on programming without worrying too much especially for the

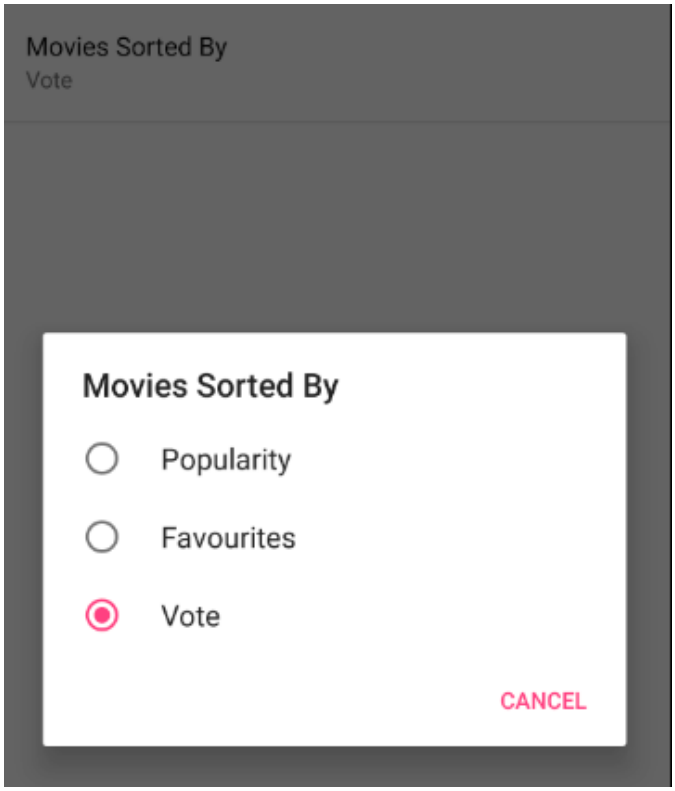

Figure 3: **Settings** - Sorting Options

interface because of the design studio that let you compose your UI with only the mouse. However, I have soon realised how limited that approach was, and I studied from the web about XML and Android in general. I have learned about activities and their life cycle, services, fragments, preferences, and databases by reading articles, websites, books, tutorials, etc., and most of these elements, even though looked unfamiliar for me at first, they became clearer in time with practice.

The most difficult challenge was the introduction of a database in my code. It was one of the topic not covered by the module and proposed to be one of the personal research. I decided to use SQLite and I have found on the web many tutorials and reusable code which I discovered they followed a pretty standard pattern. I have used part of the code explained on TutorialsPoint[11], and Proft.me [10] which had what I needed with some little touches to adapt it to my needs.

Finally, I can say that I am pretty satisfied with the outcome, even though it does not look like a product ready to be put on the market, but with much potential to be a good application with the appropriate and advanced implementations.

References

[1] "The movie db." URL: https://www.themoviedb.org/.

[2] D. S. Wells, *Notes And Workbook*. APress, 2011.

[3] J. Friesen and D. Smith, *Android Recipe*.

[4] "Android developers." URL: https://developer.android.com/.

[5] "Tutorials point." URL: https://www.tutorialspoint.com.

[6] P. Lzx, "Cinemapp," Dec. 2017. URL: https://play.google.com/store/apps/details?id=com.lzx.cinemapp.

[7] "The movie db api." URL: https://developers.themoviedb.org/3/getting-started/introduction.

[8] "Picasso." URL: http://square.github.io/picasso/.

[9] "Stack overflow use of picasso." URL: https://stackoverflow.com/questions/21889735/resize-image-to-full-width-and-variable-height-with-picasso.

[10] "Tutorial content provider." URL: http://en.proft.me/2017/03/12/android-contentprovider-tutorial/.

[11] "Tutorial on the use of sqlite." URL: https://www.tutorialspoint.com/android/android$_sqlite_database.htm$.