Movie Watchlist: Project technical Report

Ricardo Monteiro (1240842), Gonçalo Matos (1241270)

Abstract. This report documents the development of the android app Movie Watchlist. Throughout the report, the whole process of making the application will be clearly evidenced, as well as what came out of the work put into it.

Keywords: API, android, java, app, movie watchlist, login.

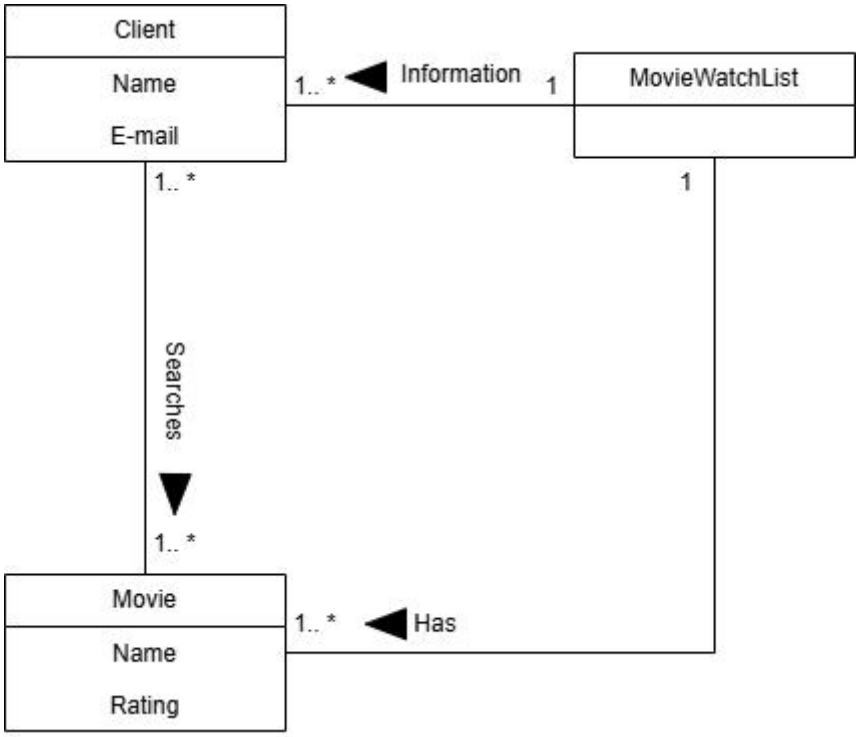
1 Context

As part of the software and mobile development course, we were asked to make an android project using java programming language.

Our project, named Movie Watchlist, consists of searching for movies, through an API. There, the user can add the movie to a watchlist, where it will be stored until the user decides to remove it. There is also a rating system, and a top 10 movies list, based on the rating given to them. This is useful, as it allows an organized list for people to remember and add movies they want to watch, know after they watched what they thought of them, and the ranking of the best movies. We implemented a register and login system, where the account will be stored for future use, even if you leave the app. That way, it is possible to have various users, and each one can have his own taste.

2 Analysis

2.1 Domain Model



The app requires a login, which is through knowing the Client's email, and password. The client searches for the movie (or movies), which has a name and rating. The movie is part of the application.

2.2 Non-functional Requirements

2.2.1 Usability

- Graphical Interface

2.2.2 Reliability

- No

2.2.3 Performance

- Must occupy a low memory space.
- Mustn’t use a lot of phone charge.

2.2.4 Supportability

- Android platform
- Minimal android API 16

2.2.5 Design Constraints

- No

2.2.6 Implementation Constraints

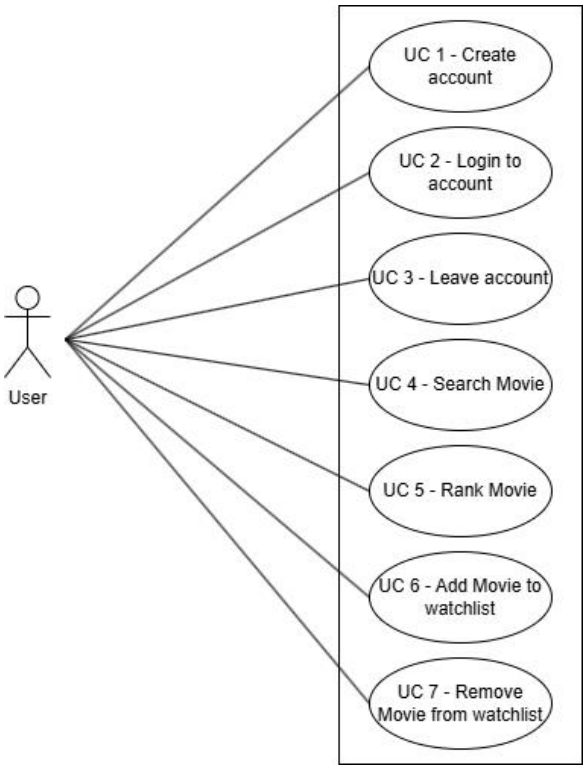
- Java language

2.2.7 Interface Constraints

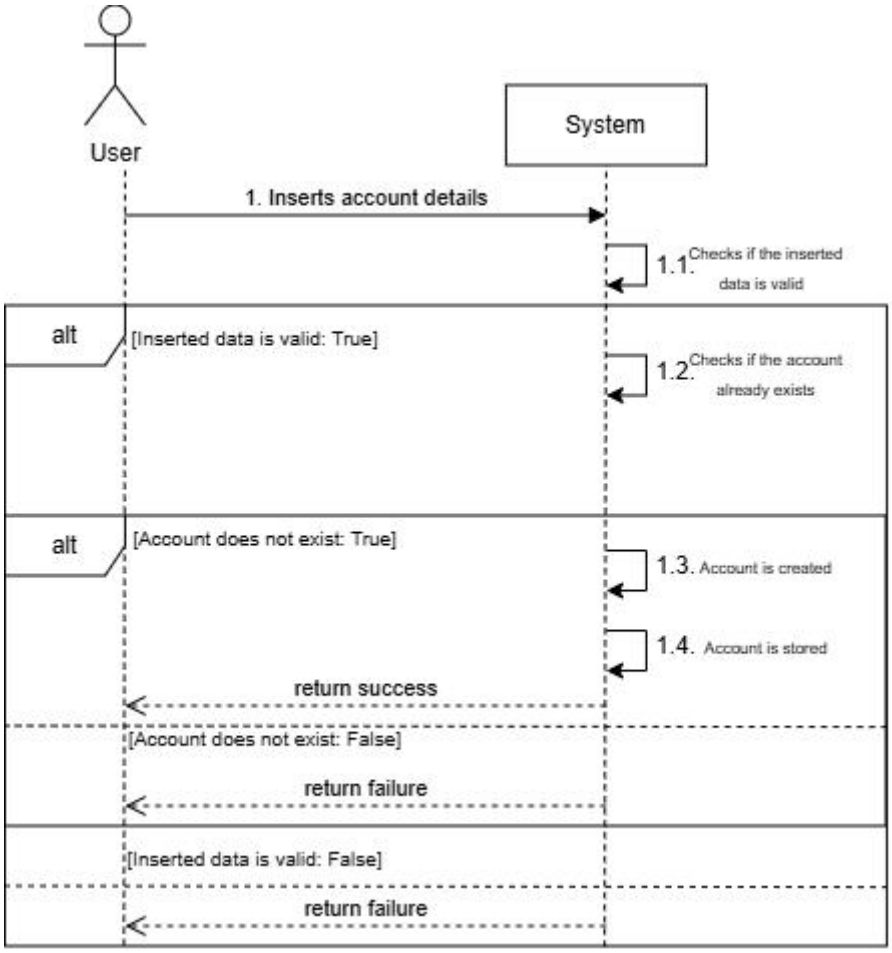
- Android Graphical Interface

2.2 Functional Requirements

- Functionalities

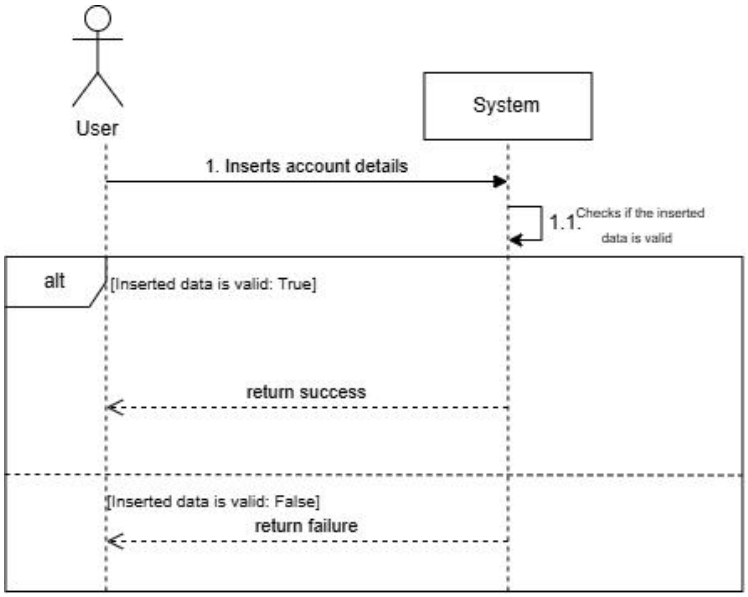


- **UC 1 – Create account:**



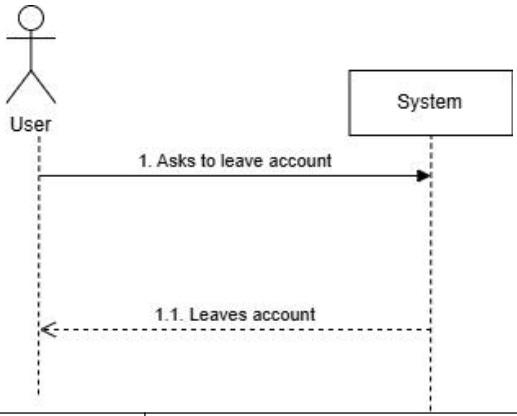
Description	Add a client to the system.
Pre-condition	There are no pre-conditions associated with this use case.
Post-condition	The account is created and stored.
Basic Path	1. The user enters the account data. 2. The system checks if the data is valid. 3. The system checks if the account already exists. 4. The account is created. 5. The account is stored.
Alternative Path	• Invalid data The system returns failure. • Account already exists. The system returns failure.

- **UC 2 – Login to account:**



Description	Login to an account.
Pre-condition	The account must already exist.
Post-condition	The user accesses the account.
Basic Path	1. The user enters the account data. 2. The system checks if the inserted data is valid. 3. The system opens the account.
Alternative Path	• Invalid data The system returns failure.

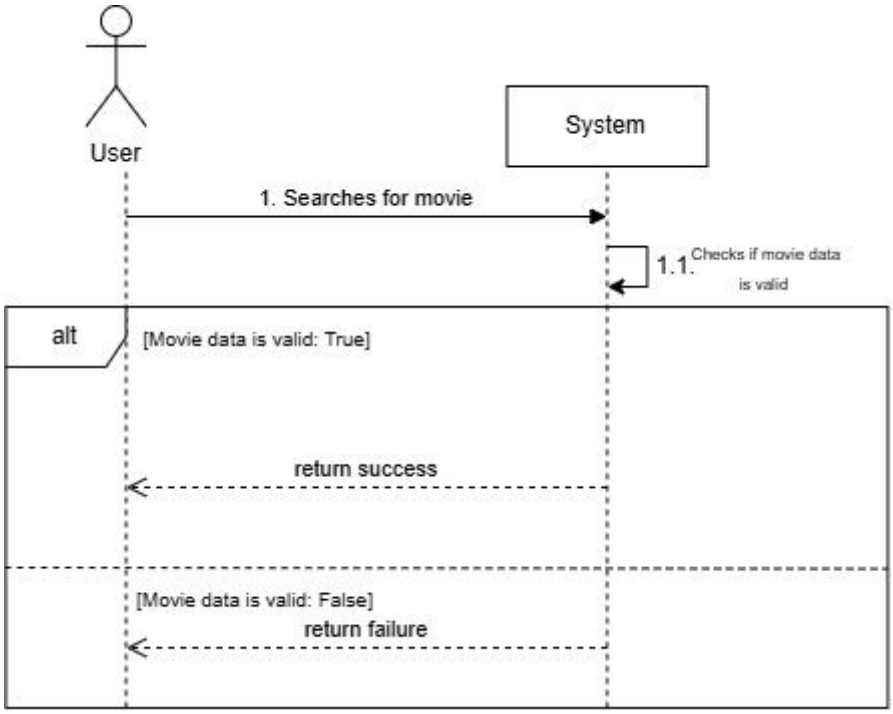
- **UC 3 – Leave account:**



Description	Logout from the account.
Pre-condition	The account must already exist.

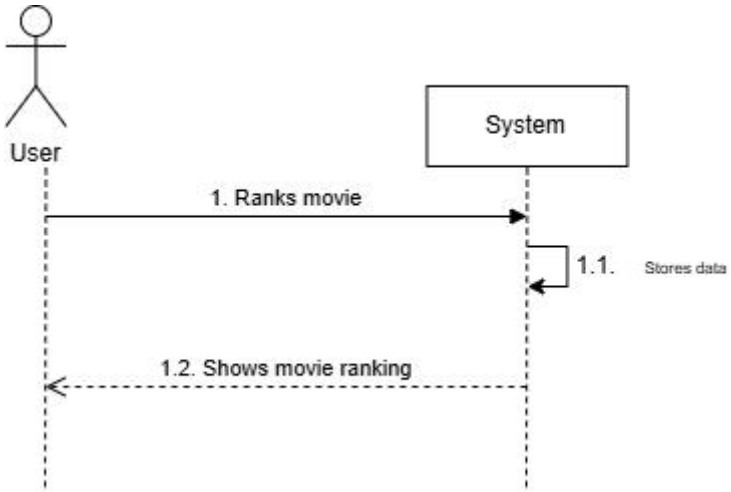
	The account must be logged in.
Post-condition	The user leaves the account.
Basic Path	1. The user asks to leave the account. 2. The system leaves the account

- **UC 4 – Search Movie:**



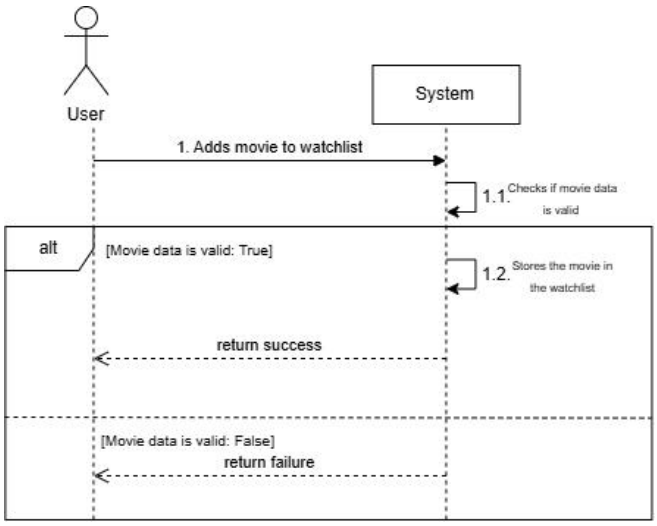
Description	The movies are shown to the user by the system.
Pre-condition	The user must be logged in to an account.
Post-condition	The options of available movies searched are shown.
Basic Path	1. The user enters the movie name. 2. The system checks if movie data is valid. 3. The system shows list of available movies.
Alternative Path	• Invalid movie data The system returns failure.

- **UC 5 – Rank Movie:**



Description	The user ranks a movie.
Pre-condition	The user must be logged in to an account.
Post-condition	The ranking is saved.
Basic Path	1. The user ranks the movie. 2. The system saves the ranking. 3. The system shows the ranking.

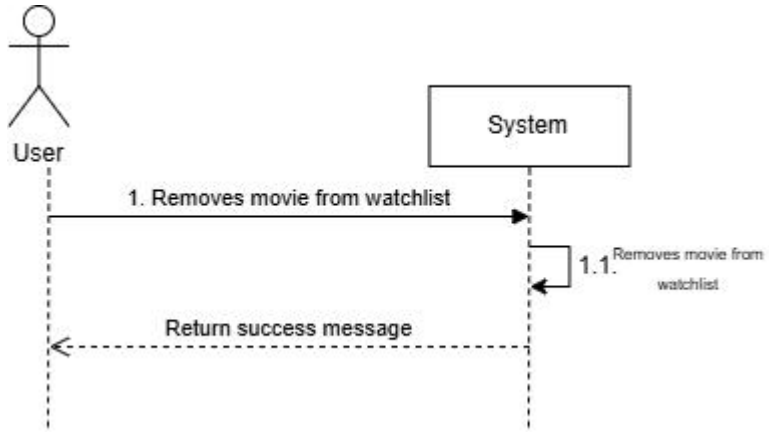
- **UC 6 – Add movie to watchlist:**



Description	The user adds a movie of choice to the watchlist

Pre-condition	The user must be logged in to an account. The user must have searched for the movie.
Post-condition	The movie is added to the watchlist.
Basic Path	1. The user adds movie to the watchlist. 2. The system checks if the movie data is valid. 3. The system stores the movie in the watchlist.
Alternative Path	• Invalid movie data The system returns failure.

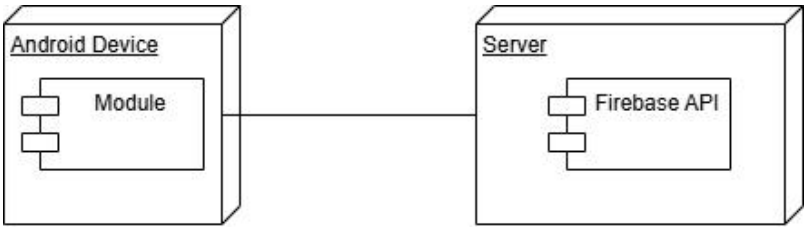
- **UC 7 – Remove movie from watchlist:**



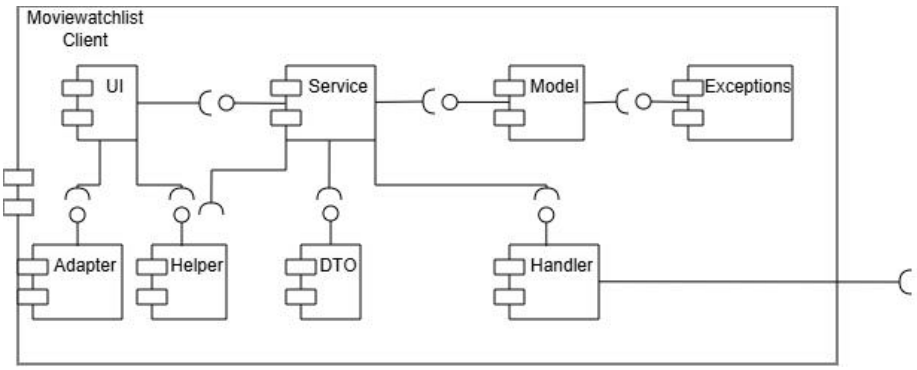
Description	The user removes a movie from the watchlist.
Pre-condition	The user must be logged in to an account. The movie must be on the watchlist.
Post-condition	The movie is removed from the watchlist.
Basic Path	1. The user removes a movie from the watchlist. 2. The system removes the movie from the watchlist. 3. The system returns a “movie deleted from watchlist” message.

3 Design

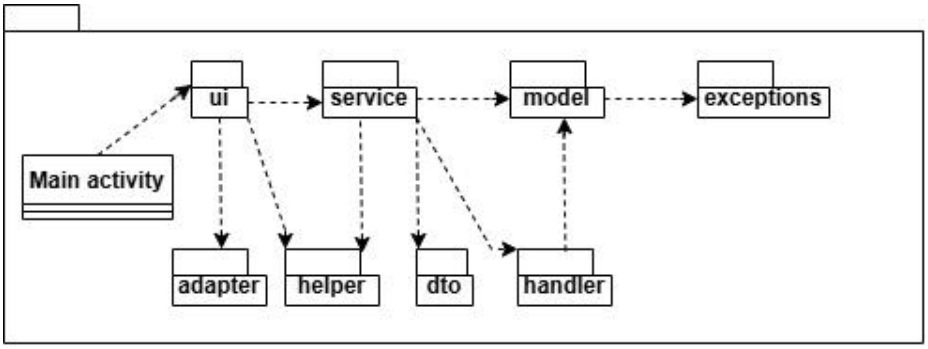
3.2.1 Physical architecture



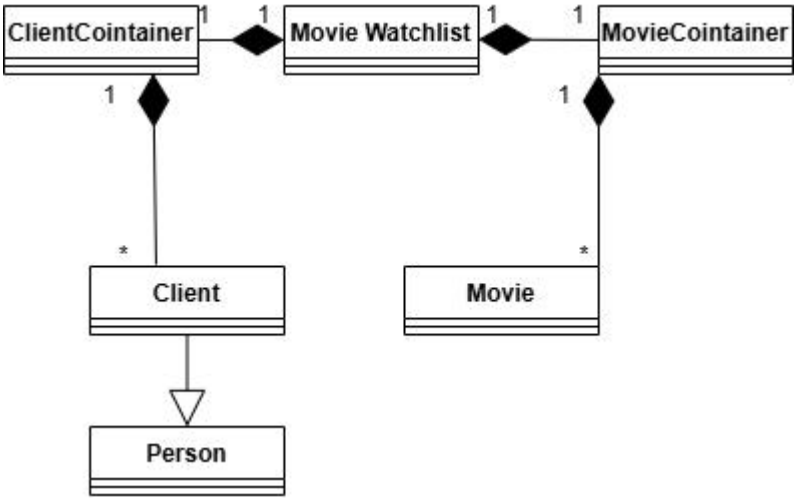
3.2.2 Logical architecture



3.2.3 Code organization

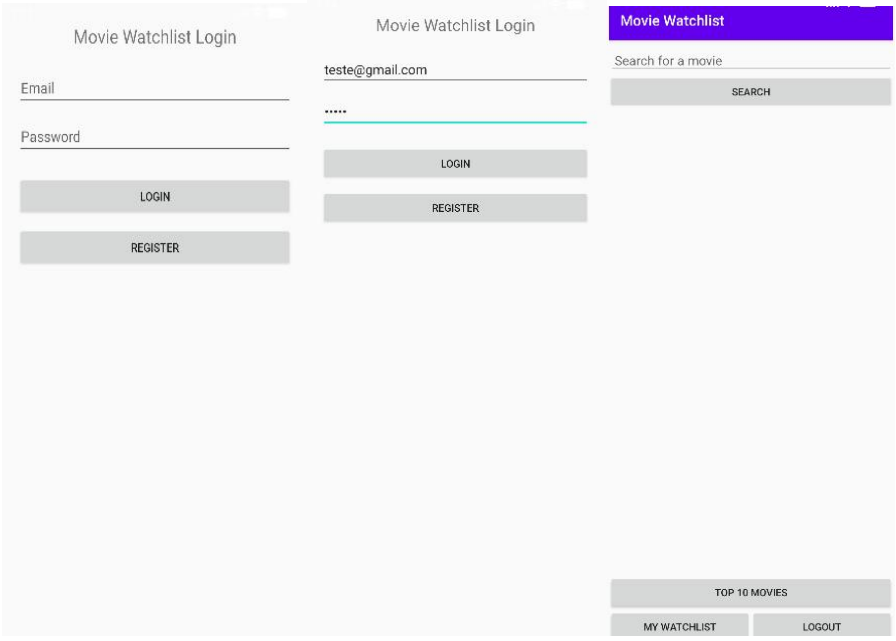


3.2.4 Class diagram



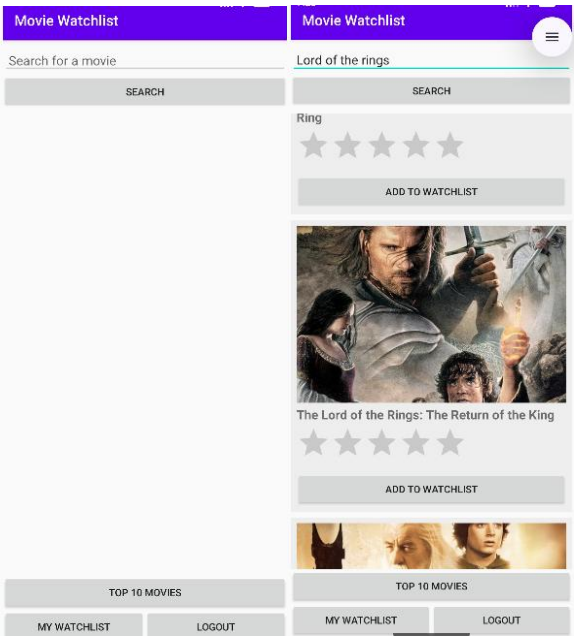
3.2.5 User Interface

- **Create an account and login:**



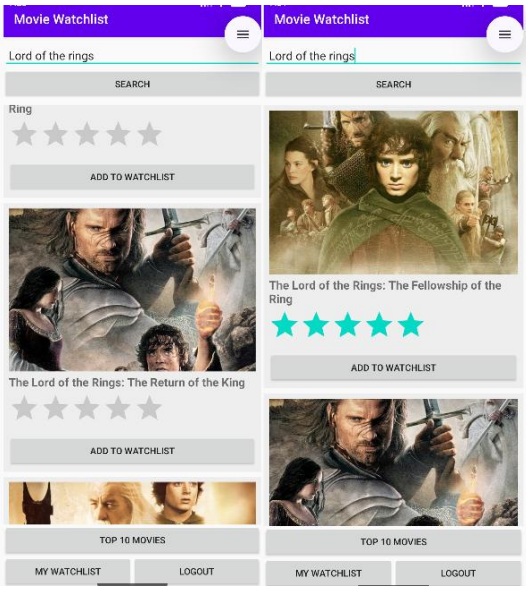
By entering the right credentials, we can either login or creat an account, which will take us to the main menu, which allows us to either logout, search for movies, check the watchlist or the top 10 movies.

- **Search movie:**



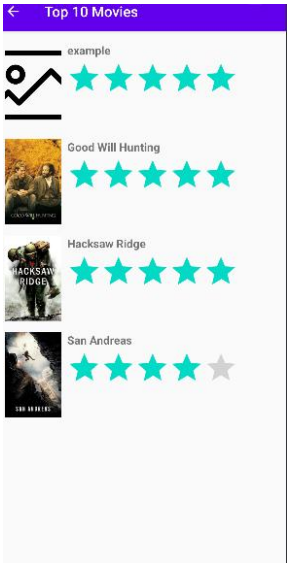
By searching for the movie we want to find, all the options that relate to the searched name appear. Here we can rate the movie and add it to the watchlist.

- **Rank movie:**



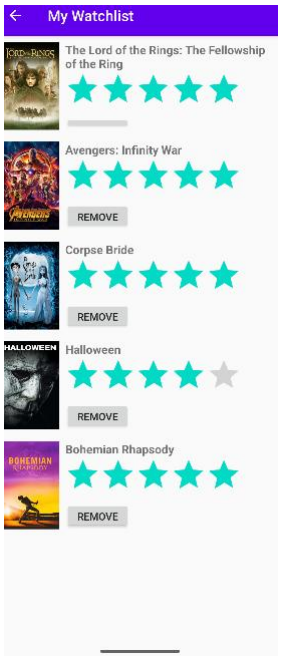
By swipping on the stars, we can give the movie any rating we'd like, and it will be stored.

- **Top 10 movies:**



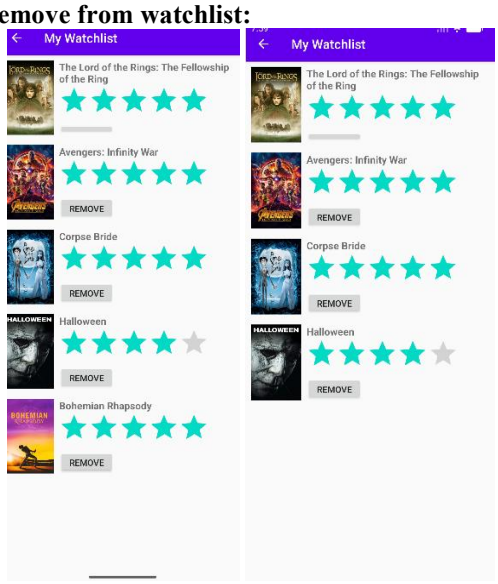
Here we can see an example of the top 10 movies list, based on the rating given to the movies by the users.

- **Watchlist:**



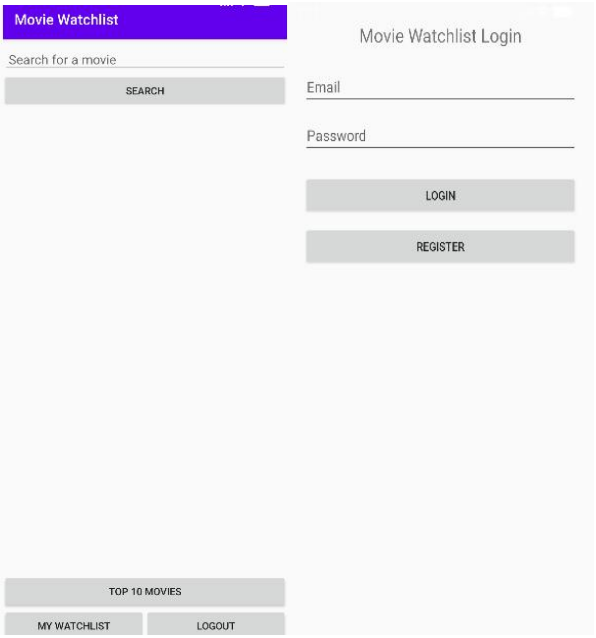
Here we can see all the movies added to the watchlist.

- **Remove from watchlist:**



By clicking on “remove”, on an movie we’d like to remove from the watchlist, the movie will me removed from the list. Here is an example of removing the movie “Bohemian Rhapsody” from the watchlist.

- **Exit account:**



To logout, we can simply click the logout button, and it takes us back to the login/register menu.

4 Conclusion

Overall, we feel like we accomplished the main goal of this project, since every functionality is working. We did have problems with code structure, having varied from what was ideal, but managed to pull through with an app we feel confident about.