REMOVAPP

PROJECT REPORT

FURKAN ATAKUL

CONTENTS

OBJECTIVE OF THE PROJECT AND PROJECT STRUCTURE

Project Description and Objectives

Removapp takes its name from the abbreviation of the initials of Movie Review Application. It is basically a film interpretation and rating application and also provides movie details to the user. As part of this project, I retrieved movie data from external sources using the TMDb API. I created an interface for GUI users that allows operations such as searching for movies, listing movies, and accessing movie details. Users can comment and rate the movies they want, and this data is saved in the database. Thanks to the application, users can easily access the information of the movie they want and save their comments and ratings. In short, the aim is to make life easier.

Project Structure

I used a Model-View-Controller (MVC) like structure to organize the project. I divided the project into three main components.

Database package (Controller-like): Establishes the database connection and performs operations like adding and updating films using the TMDb API. This package manages the business logic of the application and communicates with the database. It typically handles the tasks of the controller.

Helpers package (Model-like): Stores real-time user information, movie details, star ratings, and other data. This package represents the model part of the application. It handles the storage, processing, and application of business logic rules.

Main package (View-like): It's the part that bridges the user interface with the application. It creates the interface where the user interacts with the application. This includes graphical elements, buttons, text fields, etc., that the user sees.

USED TECHNOLOGIES

My project was developed using the Java programming language. Among the libraries used are standard Java libraries such as java.awt and javax.swing for creating GUI components. Additionally, the java.sql package was used for database operations, and during the testing phase, Istanbul Technical University's MySQL database was utilized. The java.awt.event and java.net packages were employed for event handling and network operations. The javax.swing.table package was used for managing Swing table components. For data reading operations, the java.io package was used, and the com.fasterxml.jackson.databind package was used for handling JSON data. Finally, the javax.imageio package was used for processing image files. The TMDb API was used to fetch movie data.

FUNCTIONALITY OF THE APPLICATION

Database Connection (DataBase package): Contains the DBConnect class to manage database operations and the methods responsible for performing connection operations. The DBConnect class establishes a connection to the database and retrieves movie data from the TMDB API, then inserts or updates this data in the database. The main method initializes the project, retrieves movie data from the TMDB API after establishing a database connection, and updates it if it exists in the database or inserts it if it doesn't. JSON data from the API is processed using the Jackson library when retrieving movie data. This class enables interface classes to allow users to view and update movie data.

Helper Classes (Helpers package): Contains helper classes used in the project. These classes are typically used to facilitate repetitive tasks.

The User class is used to provide a personalized interface to the user. This class typically stores user information and provides functions specific to the user.

The Film class provides basic information for the MovieInterface class to operate. This class stores the basic properties of films and is used by MovieInterface to retrieve movie data.

The StarRating and StarRating2 classes are custom components that allow users to rate a movie. StarRating enables users to rate a movie, while StarRating2 displays the ratings given by users.

User Interface (Main Package): Contains the main method that initializes the project and calls the MovieInterface class.

The LoginInterface class provides an interface for users to log in. This interface receives credentials such as username and password and uses this information for the authentication process, as shown in Figure 1. In order for users to log in, it checks the user information registered in the database and performs the login process. It can also enable users to register, as shown in Figure 2. Therefore, the LoginInterface class handles user authentication and management operations and provides the necessary interface to perform these operations.

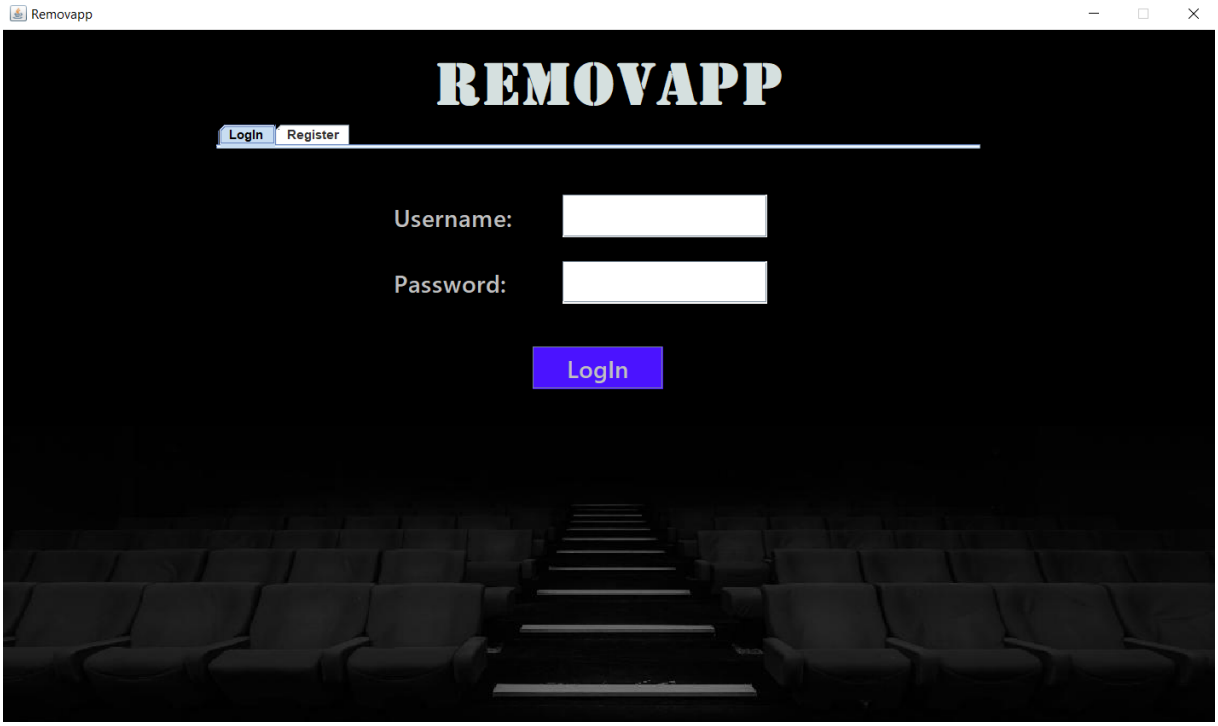

Figure 1 : LoginInterface - Login Section

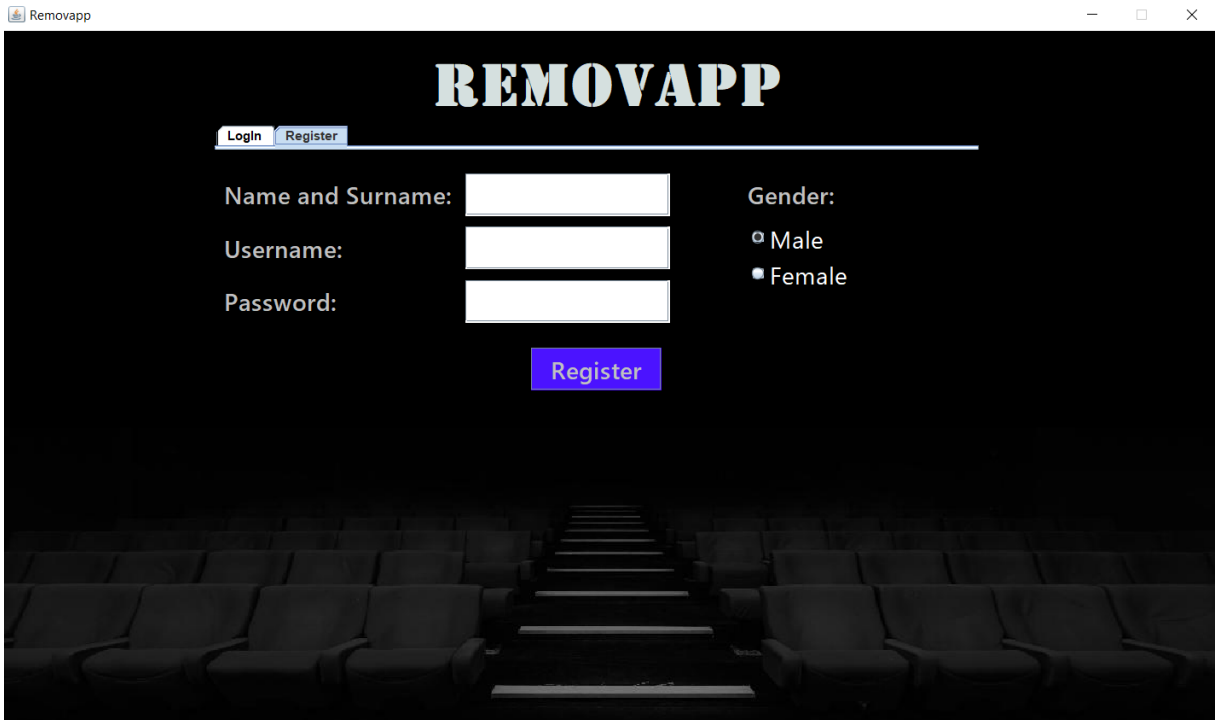

Figure 2 : LoginInterface - Register Section

The ListInterface class provides an interface for displaying the movie list. This interface lists the available movies to the user, displaying the title, genre, poster of each movie, as shown in Figure 3. It offers the user functions such as browsing the movie list, viewing movie details, selecting specific movies, searching for movies and logging out. In this way, users can view the movie list and select the movie they want or log out of their account.

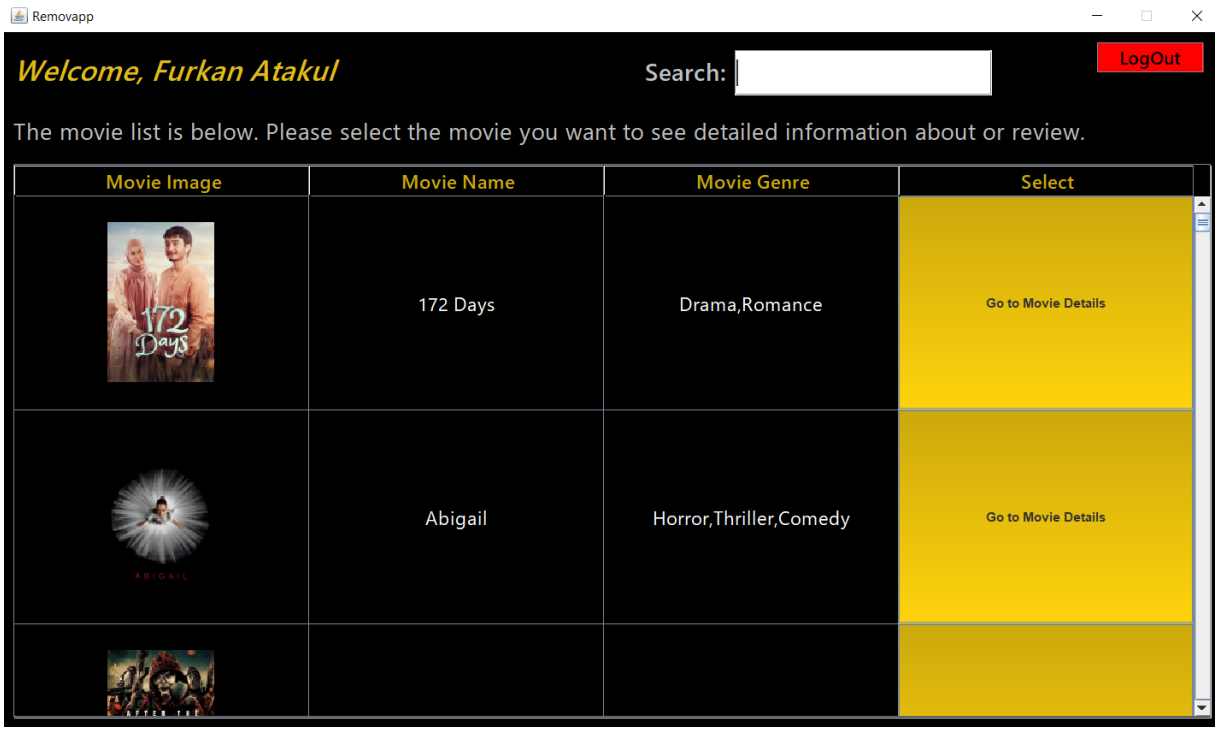

Figure 3 : ListInterface

The MovieInterface class uses the Movie class to access relevant movie data and presents it on the screen, including general information such as the movie poster, actors and overview, as shown in Figure 4. It also includes a "Back Page" button to return to the ListInterface. Users can also choose to leave comments and delete existing comments, as shown in Figure 5. Reviews appear at the bottom of the page along with their star ratings. Based on these star ratings, an average rating is calculated and displayed on the screen.

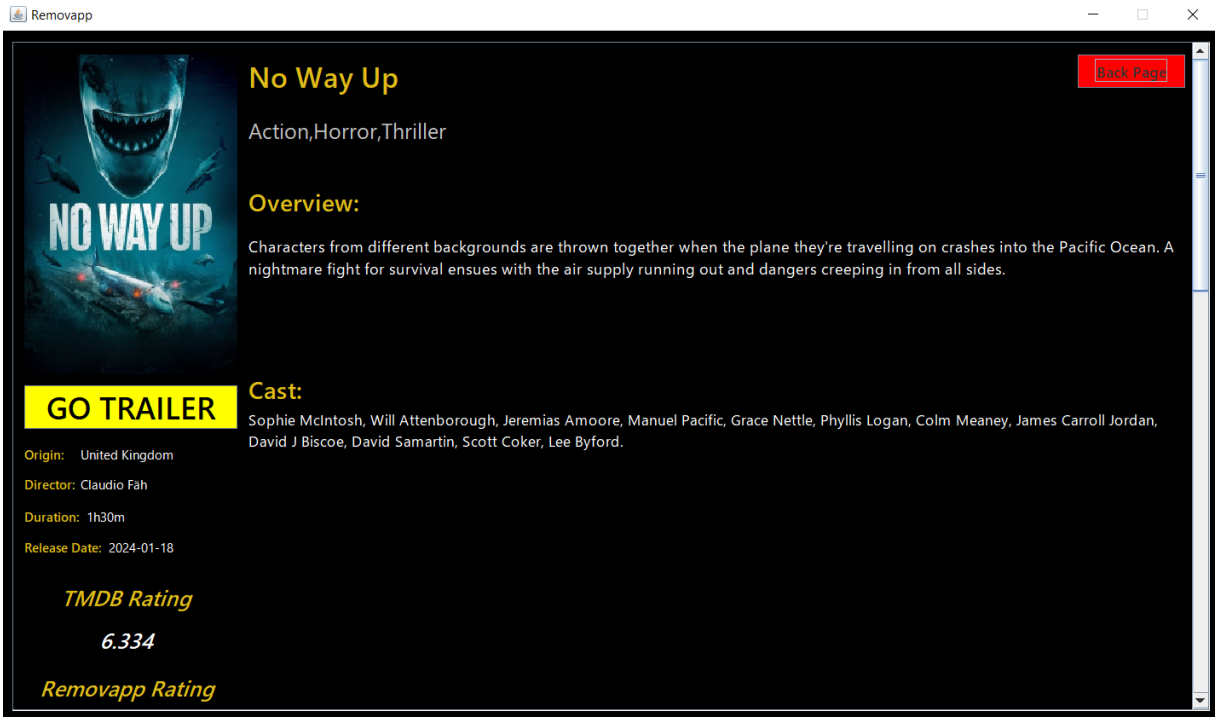

Figure 4 : MovieInterface - Movie Details

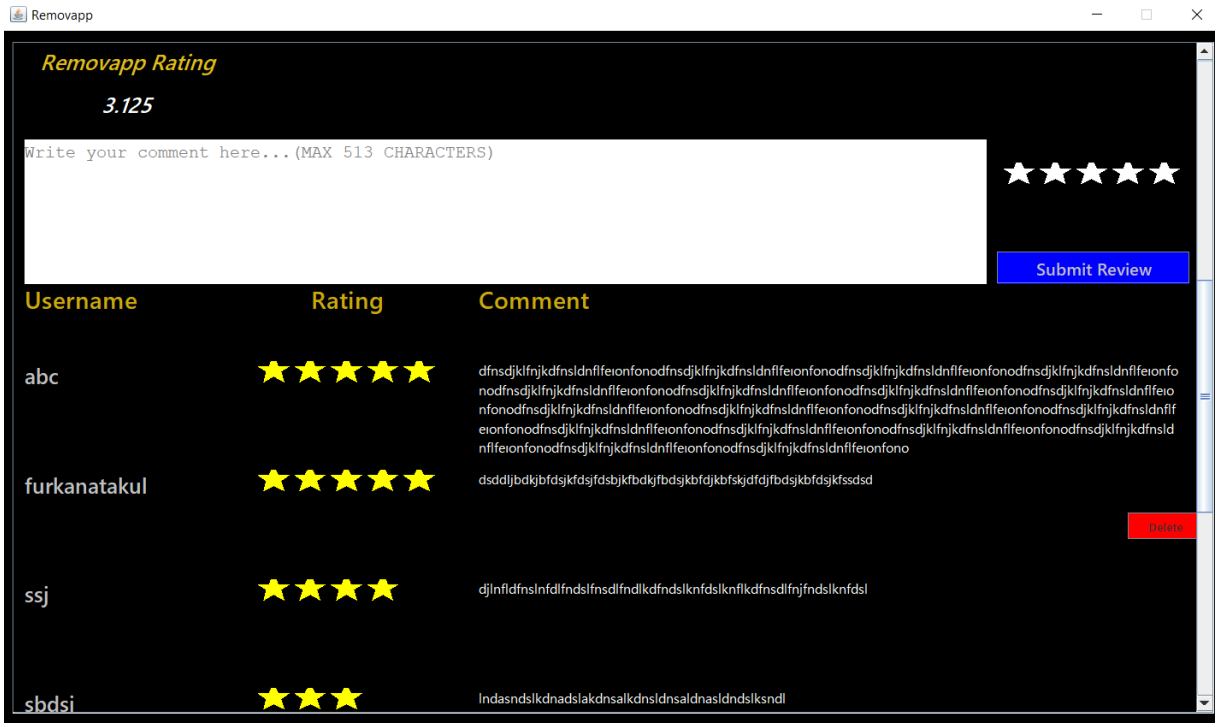

Figure 5 : MovieInterface - Ratings and Comments

CHALLENGES AND SOLUTIONS

During my development process, I encountered a number of significant challenges, from API integration to database management, from user interface design to error management.

API Integration and Data Retrieval: Initially, properly utilizing the API and processing incoming data was challenging. To overcome this challenge, I carefully reviewed the API documentation to understand how to retrieve and process the necessary data. Then, I ensured the processing of this data by populating appropriate classes.

Database Connection and Management: Establishing a connection to the database and managing data updates or additions when necessary was crucial. I securely connected to the database using tools like JDBC and performed database operations by preparing the necessary queries.

User Interface Design and Usability: It was important for the user interface to be user-friendly and easily understandable. At this point, I designed the user interface using the Swing library. Additionally, I continuously improved the interface by considering user feedback.

Error Handling and Debugging: Identifying and fixing encountered errors while writing code and testing the application was crucial. During the debugging process, I used logging operations to track and resolve errors. Furthermore, I anticipated potential error scenarios and displayed appropriate error messages to provide a better user experience.

To overcome these challenges, I adopted a patient and systematic approach, generating solutions by researching and gaining experience when necessary.

CONCLUSION

During the development of this project, a number of important steps were taken to create a movie database application. Various challenges were encountered, such as creating a movie database, TMDb API integration, user interface design and managing user interaction. However, these challenges have been successfully overcome and as a result, a platform has been developed where users can search, review and comment on movies. During the project, the basic capabilities of the Java programming language were utilized and code organization was achieved using a structure similar to MVC (Model-View-Controller). Additionally, the ability to deal with and resolve difficulties encountered during the project was improved. It is aimed to continuously develop and improve the project by considering future plans and developments such as the movie favorites system. As a result, this project development experience increased the ability to understand and apply different aspects of the software development process. The movie database application has made a significant contribution as a step towards providing a valuable experience to users.