Kathmandu University

Department of Computer Science and Engineering

Dhulikhel, Kavre

A Project Report
on
"MovieMaestro"

[Code No: ENGG 102]

(For partial fulfillment of I Year/II Semester in Computer Engineering)

Submitted by

Aditya Pokharel (32)
Abhiyan Regmi (42)
Sameer Kumar Sah (43)
Parikchit Sen (45)

Submitted to

Mr. Dhiraj Shrestha

Department of Computer Science and Engineering

Submission Date: 25th August, 2024

Bona fide Certificate

This project work on

“MovieMaestro”

“Aditya Pokharel” (32)

“Abhiyan Regmi” (42)

“Sameer Kumar Sah” (43)

“Parikchit Sen” (45)

who carried out the project work under my supervision.

Project Supervisor

_____

Mr. Suman Shrestha

Visiting Lecturer

Department of Computer Science and Engineering

Date: 25th August, 2024

Acknowledgement

We would like to express our deepest gratitude to our supervisor, Mr. Suman Shrestha, for his unwavering support, guidance, and mentorship throughout the course of this project. His insightful feedback and constant encouragement have been essential in the successful completion of this work.

His expertise and attention to detail have greatly enhanced our understanding of the subject, and we are truly fortunate for the opportunity to work under his supervision. This project would not have been possible without his invaluable contributions and commitment to our academic growth.

Abstract

MovieMaestro is a movie recommendation system that aims to make the movie selection process easier and more user friendly. This project's main goal is to improve the user experience in finding and picking movies by suggesting them to the user. The application was built using C++, along with Qt framework for constructing the user interface. To efficiently store and retrieve the data about users as well as movies, SQLite3 was used as its database management system. This desktop application will recommend movies depending on preferred genres, average & personalized ratings, date of release and how they have been interacting with the movies.

In conclusion, MovieMaestro provides a personalized and efficient movie recommendation facility which drastically improves one's ability to find films that match their preferences. Regular updating of database with new movie information would be helpful in giving better user experience.

Keywords: *MovieMaestro, C++, Qt framework, SQLite3, Database Management System*

Table of Contents

List of Figures

Acronyms/Abbreviations

3NF	Third Normal Form
ASCII	American Standard Code for Information Interchange
GPU	Graphics Processing Unit
RAM	Random Access Memory
SQL	Structured Query Language
SSD	Solid State Drive

Chapter 1 Introduction

Countless movies on the internet can be overwhelming for viewers to choose from in this digital age. However, with technology coming into play, this has changed with the existence of recommendation systems. These systems use algorithms to examine through and predict the user's preferences thereby helping users to find films they are likely to enjoy. The creation of efficient recommendations is even more important considering that today there are streaming services where people can access an array of films at once.

Our project, MovieMaestro, aims to simplify and improve the movie selection process by providing a better user-friendly experience. It is a desktop application targeted at enhancing the user experience with personalized recommendations of movies based on preferred genres, average & personalized ratings, date of release and how they have been interacting with the movies. This project's main goal is to improve the user experience in finding and picking movies by suggesting them to the user. The application was built using C++, along with Qt framework for constructing the user interface and SQLite3 was used for database management system. It intends to make movie selection an enjoyable and hassle-free experience, ultimately transforming how users engage with digital entertainment.

In our project, all the information of the movies stored in our database was collected from a movie streaming site called Sflix[1]. The movies were added manually as samples through the admin panel.

1.1 Background

Recently, tremendous progress has been registered in the development of such systems, with major changes already witnessed, particularly in the case of the machine learning algorithms, collaborative filtering and content-based filtering techniques being the focus. Netflix and Hulu are among platforms that mainly rely on the deployment of complicated algorithms to provide the right insights and services of entertainment, thereby & boosting customer satisfaction.

MovieMaestro is an application whose main purpose is to provide an efficient and enjoyable platform for users to find films that fit their preferences. The easy framework and usability of MovieMaestro points to the fact it will be a valuable tool for users who want to get personal movie suggestions. As it is a simple tool that is expected to deliver quality content to the users based on preferred genres, average & personalized ratings, date of release and how they have been interacting with the movies which ultimately leads to the feeling of satisfaction and platform lightness.

The downside of MovieMaestro is that it does not have any advanced or complex algorithms in the recommendation process. Absence of these sophisticated procedures might make this system fall behind other advanced systems. Therefore, the recommendations given are not accurate enough and it raises disconnection aspect.

Despite the faults, it is a simple and efficient platform which still provides valuable recommendations based on user preferences and available data.

1.2 Objectives

- To create a simple and reliable desktop application using the Qt framework that enhances the user experience by making movie discovery and selection process easy and enjoyable
- To recommend movies that meet the expectations of different users based on their individual taste preferences such as genres, personalized & average ratings and date of release
- To efficiently store and retrieve user data and movie information, ensuring quick and reliable performance

1.3 Motivation and Significance

Several times, users find it hard to get new movies that are in line with their taste. It is this problem which motivated us to come up with the project called MovieMaestro. Its aim is to ensure a great user experience and offer personalized recommendations. On another note, it keeps simplicity and ease of access at its center while still giving dependable suggestions. This will help make the user's life easier by saving his or her time and simplifying the process of finding enjoyable movies.

Chapter 2 Related Works

2.1 MovieLens

MovieLens[2] is a research site run by GroupLens Research at the University of Minnesota. It uses collaborative filtering technology to make recommendations of movies that users might enjoy, and to help you avoid the ones that you won't. Based on the user's movie ratings, it generates personalized predictions for movies users haven't seen yet.

2.2 FindMoviez

FindMoviez[3] is a movie recommendation system based on a combination of two recommendation algorithms, implemented in a web application. The three sub-data sets (i.e. ratings, users, and metadata) of the famous Movielens data set have been used. A combination of item–item collaborative filtering and genre based using the average weighted rating method has been used. These algorithms have been modified in a way where the user is always recommended movies, and even if one of the above algorithms fails, the other comes into play making this product more reliable for the user.

Chapter 3 Design and Implementation

The development of the MovieMaestro application has involved a structured and systematic approach that involves various stages such as analysis, design and implementation. It explains the order in which activities were done during the course of the project.

- **Research & Planning**

  The research and planning phase involved identifying the core features of MovieMaestro, and selecting an appropriate technology (C++, Qt, and SQLite3) for its development.

- **Set Up & Work Distribution**

  In this phase, the project environment was set up, including configuring version control system, development tools and setting up the database. Work distribution was organized, assigning specific tasks such as front-end design, backend development, and algorithm implementation to team members.

- **Frontend & Backend Development**

  Front-end development concentrated on creating user-friendly interfaces using the Qt framework with a view for delivering flawless interactions. Back-end development consisted in integrating the recommendation algorithms into the SQLite3 database while managing data storage and retrieval processes.

- **Testing**

  A basic check was conducted to determine if major functionalities of MovieMaestro still worked or not. This primarily constituted of ensuring that essential characteristics like user interactions, movie recommendations and data handling were okay.

3.1 System Requirement Specifications

3.1.1 Software Specifications

The following are the software specifications that were used for successful creation and operation of MovieMaestro:

- **Operating System:**
  Windows: The application was developed and tested in windows.

- **Development Environment:**
  QT Creator: Development was carried out using Qt Creator, a powerful integrated development environment specifically tailored for C++ and Qt applications.

- **Database:**
  SQLite3: SQLite3 was the chosen database management system, selected for its lightweight, serverless nature, making it an ideal choice for a desktop application like MovieMaestro.

- **Programming Languages:**
  C++: The core logic and user interface components of MovieMaestro were developed using C++, leveraging its performance benefits.
  SQL: SQL was used extensively for creating and managing the database, ensuring efficient storage, retrieval, and manipulation of data.

- **Other tools:**
  Git: Git was used for version control, enabling collaborative development and efficient management of code changes.

3.1.2 Hardware Specifications

The hardware specifications for the successful development of MovieMaestro are:

- **Processor:** Intel Core i5 12500H
- **Graphics:** Integrated GPU
- **Memory:** 8 GB RAM
- **Storage:** 512 GB SSD

Chapter 4 Working Mechanism

The working mechanism of MovieMaestro is explained below along with some screenshots are:

4.1 Admin Panel

4.1.1 Admin Credentials Verification

Admin starts by entering their username and password after which he/she is prompted to complete a two-factor authentication that involves entering a 4-digit code. If the admin forgets their password or the two-factor authentication code during the login process, they can reset it by answering two predefined security questions. This provides a secure method to regain access while maintaining account protection.

Once logged in, admins have the ability to change their login credentials. This process requires the admin to enter the current password and two-factor authentication once again for extra security.

All the login credentials for admins are stored in the SQLite3 database using an obfuscation method. In this method, each letter of the text is first converted to its ASCII value. Custom calculations are then applied to these values, which are subsequently converted to hexadecimal format. The processed values are combined and stored in the database, adding an extra layer of security.

4.1.2 Data Input

Once logged in, the admin can navigate to the movie addition section of the admin panel. This interface allows the admin to input and manage movie data. The admin enters key details about the movie, including the title, genre, director, cast, release date, duration, a brief description and a movie poster. Once all the

necessary information is provided, the movie data is stored in the SQLite3 database.

The interface also provides options for updating or deleting existing movies. Admin can select any movie from the given list or search the movie by its title and can modify any aspect of the movie's details. Once the necessary updates are made, the admin saves the changes, which are immediately reflected in the SQLite3 database. While deleting a movie, the admin is prompted to confirm the action. Upon confirmation, the movie is permanently removed from the SQLite3 database.

4.1.3 Data Storage and Management

The SQLite3 database which is created to store movies is organized into several tables, each designed to store specific types of information efficiently. Movies Table stores core information about each movie, including the title, duration, release date, description, and poster. The table also includes a rating column to capture average user ratings. Genres Table, Directors Table & Casts Table stores genre, director & cast information, with each genre, director & cast having a unique genre ID, director ID & cast ID respectively.

In order to establish many-to-many relationships between movies, genres, directors, and cast members, linking tables are created. These tables ensure that each movie can be associated with multiple genres, directors, and cast members, and vice versa. Additionally, a table links movies with users, storing individual movie ratings provided by each user which is later utilized by the recommendation algorithm to generate personalized movie suggestions.

The database design follows the principles of normalization, specifically up to 3NF. Each piece of data is stored only once, reducing redundancy and ensuring data integrity. The primary table stores individual entities (movies, genres,

directors, and cast members), while linking tables manage the relationships between these entities. This approach minimizes data duplication, simplifies updates, and ensures that the database remains consistent and organized.

4.2 User Panel

4.2.1 User Interaction and Data Input

Users interact with MovieMaestro by providing a username, password and profile picture to either create a new profile or login to an existing one. Their preferred genres are also specified during the process which is used as the foundation for personalized recommendations. Password, preferred genres & profile picture can also be changed later. Once logged in, MovieMaestro utilizes session handling to keep them logged in until they log out manually.

4.2.2 Data Storage and Management

The credentials of the users are stored in SQLite3 database to efficiently store and manage the login process. To enhance data security, MovieMaestro employs the Hill cipher technique for encrypting the passwords of the user. This method uses linear algebra and matrix operations to encrypt and store data in the database.

4.2.3 Movie Selection and Display

Once logged in, the homepage of the MovieMaestro is displayed using the recommendation algorithm. The movies are organized in three different tabs. They are For You, Top Rated and Latest. Each tab displays the movies according to their specified recommendation algorithm. In each tab, users are allowed to select any movie & view its details and also rate it according to their preferences. The rating of the movie for each user is stored in the database which is used to refine and enhance future recommendations, ensuring that the suggestions remain relevant and personalized. Additionally, users have the ability to search for any

movie by its title using the search bar, making it easier to quickly find and explore specific films.

4.2.4 Recommendation Algorithms

Three different algorithms have been implemented for displaying movies in For You, Top Rated & Latest tab. MovieMaestro uses filtering based recommendation algorithm and sorts the movies according to various priorities for each tab.

For Top Rated tab, the movies are sorted on the basis of average rating of all users. For Latest Tab, the movies with the genres preferred by the users are filtered out after which they are sorted on the basis of the date of release. In For You tab, the movies with the genres preferred by the users are filtered out first after which the filtered movies are sorted on the basis of average user rating. Additionally if a user rates any movie with rating greater or equal to 4, other movies having the same director/s or cast/s will be added to the homepage. Furthermore if a user rates 3 or more movies having common genres with rating greater or equal to 4, other movies with the same genres will also be recommended even if the users haven't selected that genre as their preferred genre.

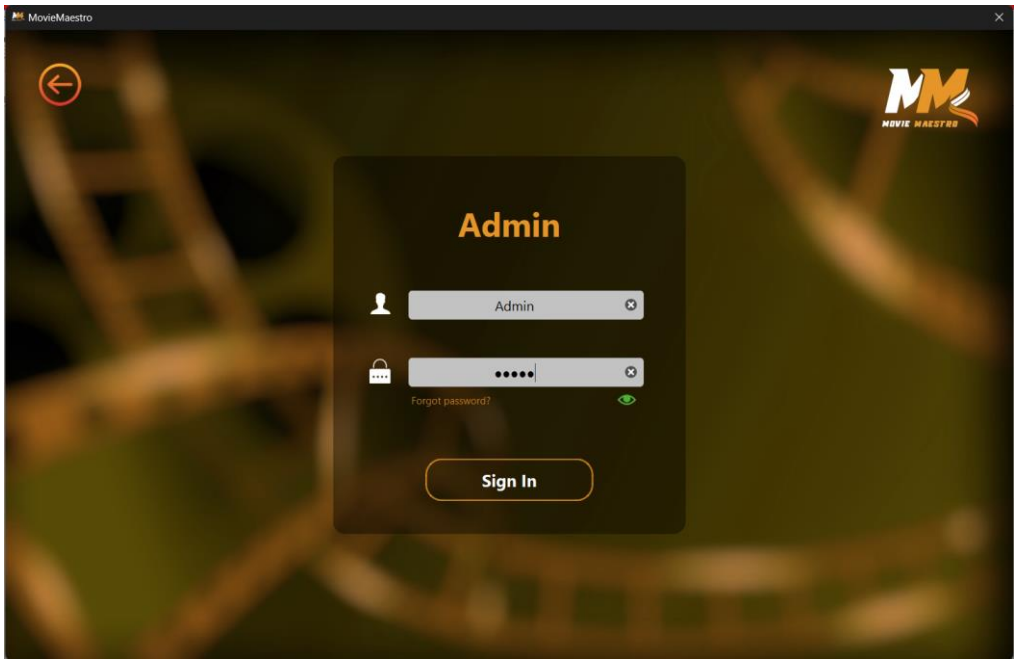

Figure 4.1: Admin Credentials Verification

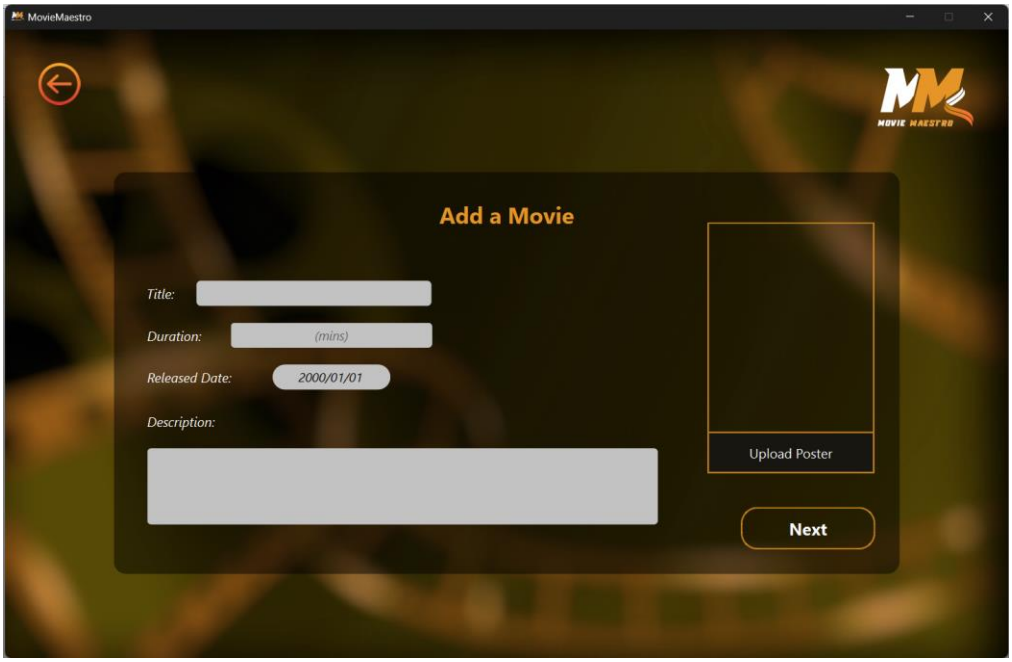

Figure 4.2: Adding Movie by Admin

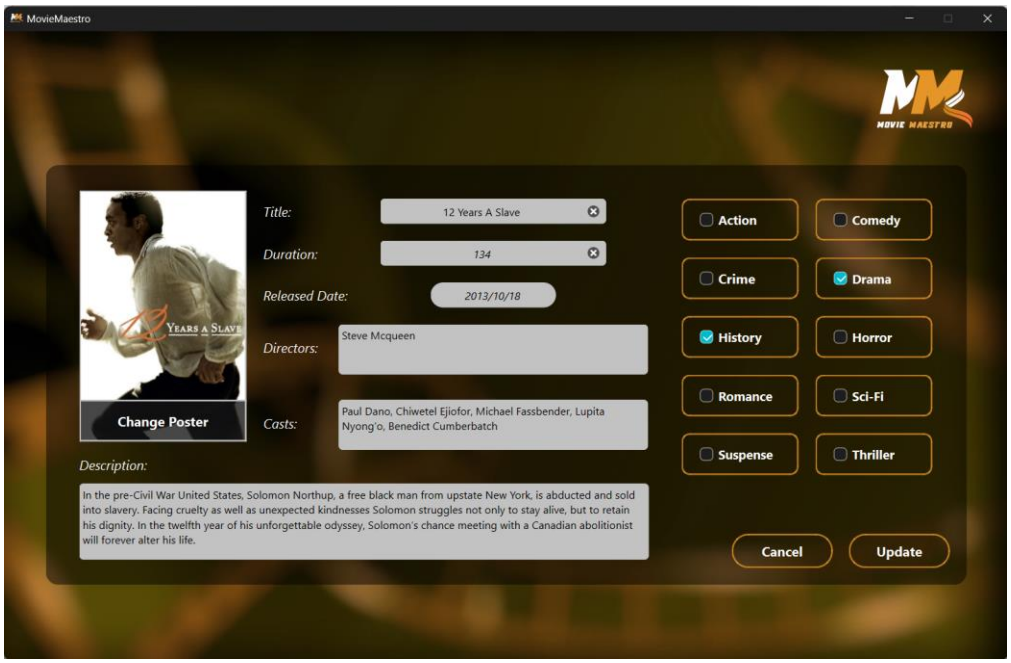

Figure 4.3: Updating Movie by Admin

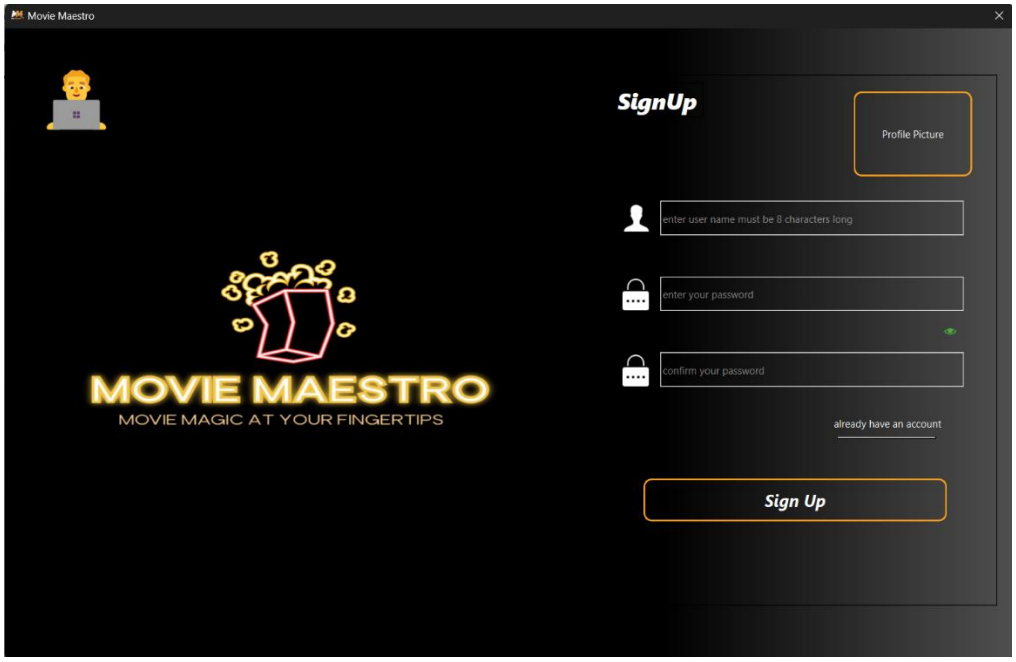

Figure 4.4: User Sign up Panel

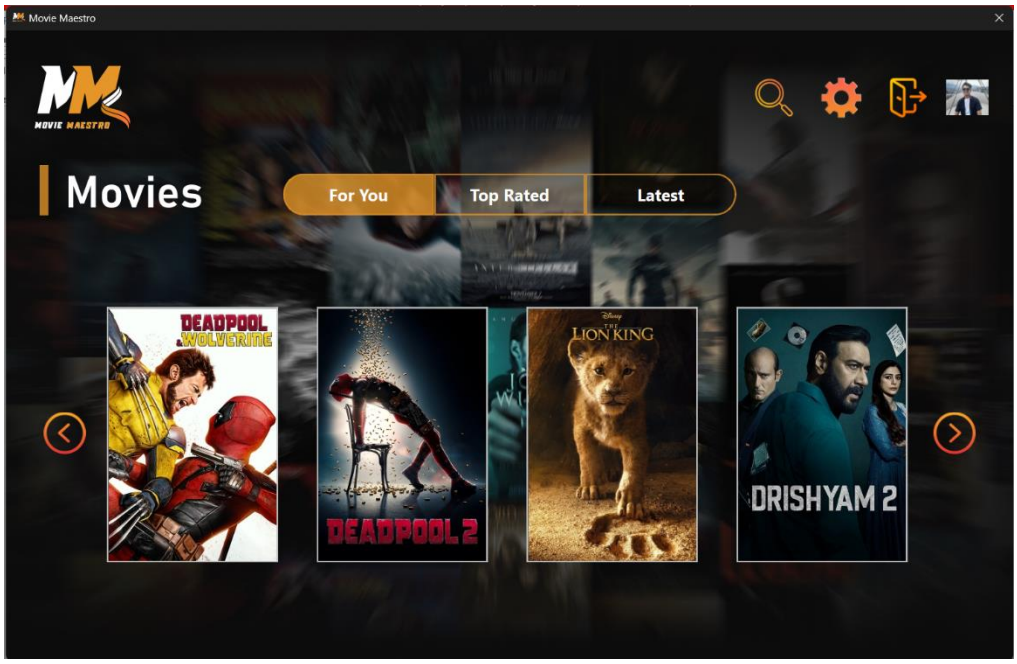

Figure 4.5: Movie Recommendation to User

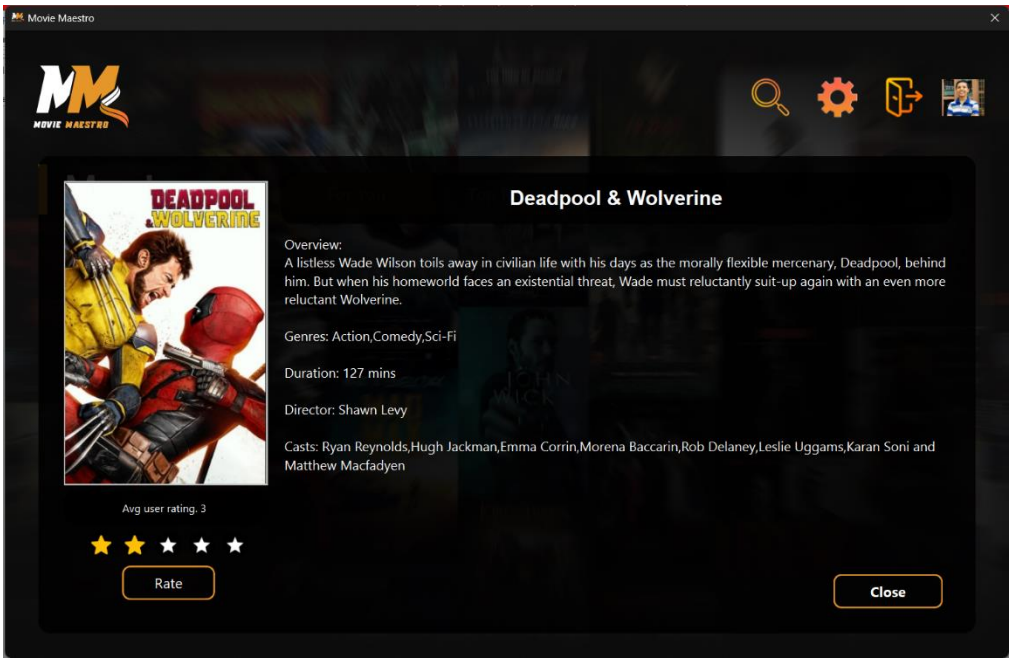

Figure 4.6: Movie Selection by User

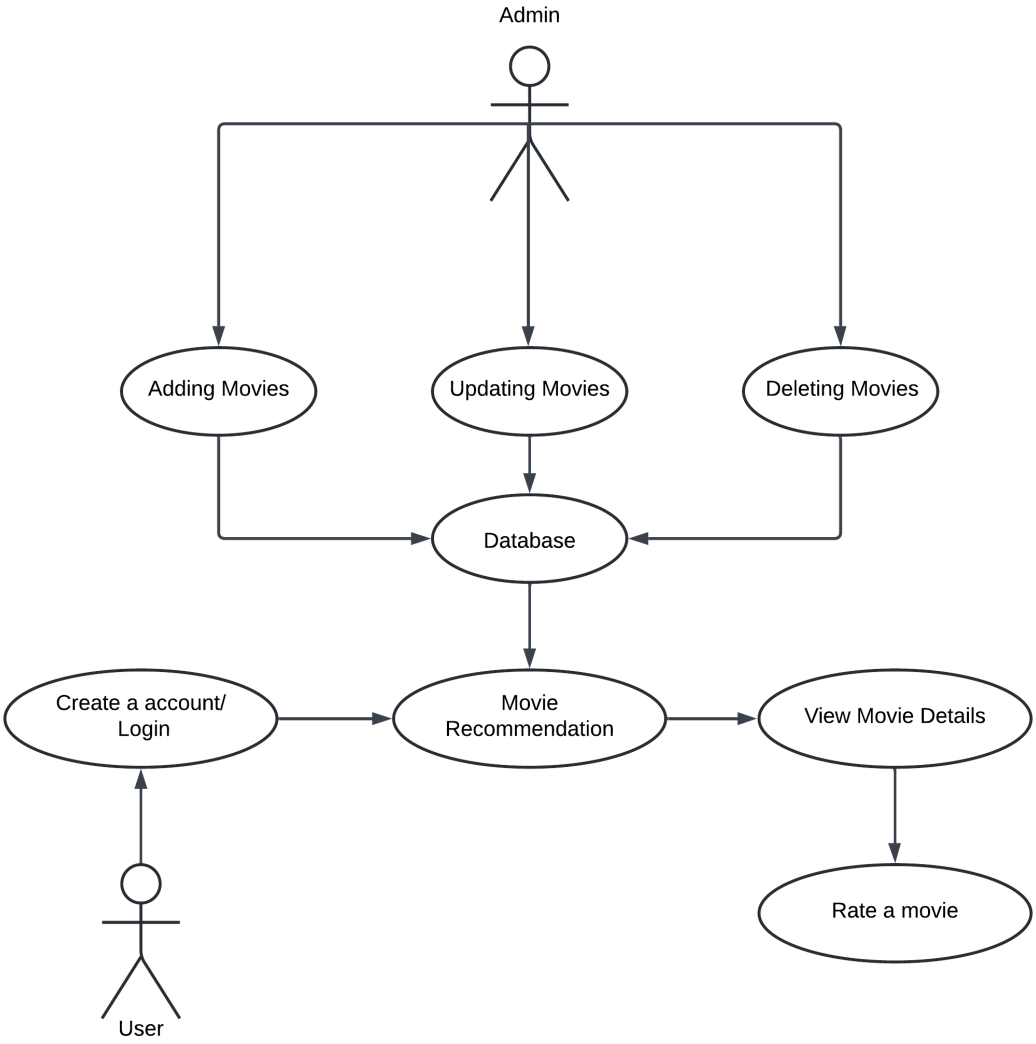

Figure 4. 7: Use-Case Diagram

Chapter 5 Discussion on the achievements

Throughout the development of MovieMaestro, several challenges were encountered due to lack of experience. The significant challenges were managing merge conflicts in Git and adapting to the Qt framework.

Since all the team members were new to Qt, understanding it and implementing its mechanisms required a significant investment of time and effort. With time, the team successfully adapted to Qt.

As multiple team members worked on different parts of the MovieMaestro project simultaneously, merge conflicts in Git became a recurring challenge. Through effective communication among team members, we were able to ensure that all changes were integrated smoothly without introducing bugs or malfunctions.

Despite these challenges, the creation of MovieMaestro was successful because of our dedication and coordination among the team members. The project's completion is a demonstration of our ability to overcome technical challenges.

5.1 Features:

During the development of MovieMaestro, several key features were implemented to enhance the user experience and improve the functionality of the system.

- **Personalized Recommendation:** One of the core features of MovieMaestro is its ability to provide personalized movie recommendations in three distinct tabs: "For You," "Top Rated," and "Latest" based on the preferred genres, average & personalized ratings, date of release and how they have been interacting with the movies.

- **Search Functionality:** The search bar feature allows users to quickly locate specific movies by entering the title.

- **User Profiles:** MovieMaestro allows users to create personalized profiles which is made secure using encryption technique. The implementation of session handling ensures that users remain logged in throughout their interaction without having to do repeated logins.

- **Administrative Controls:** The admin panel provides access to admin for managing the movie database, including the ability to add, update and delete movies. This feature is essential for maintaining the accuracy and relevance of the movie database, ensuring that users always have access to the latest and most accurate information.

Chapter 6 Conclusion and Recommendation

The development of MovieMaestro focused on delivering an application with personalized movie recommendations, having a well-designed frontend and a robust backend. The frontend development using Qt successfully provided a seamless and interactive experience for users. The integration of SQLite3 for data storage and the implementation of recommendation algorithms effectively support personalized movie suggestions based on preferred genres, average & personalized ratings, date of release and user interaction. Despite some hurdles that we faced during various development phases, we successfully created a personalized movie recommendation system.

6.1 Limitations

Several limitations were identified during the development of MovieMaestro:

- **Recommendation Algorithm:** The recommendation algorithms are relatively basic. More advanced techniques, such as collaborative filtering or machine learning-based recommendations, were not implemented due to lack of time and experience.

- **Testing:** The project was tested primarily for basic functionality. Comprehensive testing was not performed due to which there may be presence of undiscovered issues.

- **No Server:** The MovieMaestro application does not include a server for database management. As a result, the SQLite3 database is local to the device running the application, which means it cannot be accessed from other devices.

6.2 Future Enhancement

To improve MovieMaestro, several enhancements could be considered:

- **Advanced Algorithm:** Integrating more refined recommendation algorithms, such as machine learning models, could provide more reliable and accurate movie suggestions.

- **Enhanced Testing:** Advanced testing will help identify and address potential issues more effectively.

- **Database Server Implementation:** Introducing a server-based database can enable multi-device access allowing users to interact with MovieMaestro from different devices.

- **User Feedback Integration:** A feature to collect user feedback can help enhance recommendation algorithms and improve user satisfaction.

- **Additional Features:** New features such as movie trailers, comments and sharing options could enhance the application's functionality.

References

[1] Sflix. (n.d.). *Sflix - Watch movies online free*. Retrieved August 2, 2024, from https://sflix.to

[2] MovieLens. (n.d.). *About MovieLens*. Retrieved August 16, 2024, from https://movielens.org/info/about

[3] Padhi, A. K., Mohanty, A., & Sahoo, S. (2021). FindMoviez: A movie recommendation system. In *Lecture notes in networks and systems (pp. 49–57)*. Springer Singapore. https://doi.org/10.1007/978-981-33-6081-5_5

[4] Qt Group. (n.d). *Qt Documentation*. Retrieved June 14, 2024, from https://doc.qt.io/

[5] SQLite. (n.d). *SQLite Documentation.* Retrieved June 14, 2024, from https://www.sqlite.org/docs.html

APPENDIX

Gantt chart: The 12-week Gantt chart below outlines the project's schedule, displaying the planned start and end dates for each task. This chart provides a clear visual representation of the project timeline.

Task/Week	1	2	3	4	5	6	7	8	9	10	11	12
Research & Planning												
Set Up & Work Distribution												
Frontend & Backend Development												
Testing & Documentation												

Figure: Gantt chart