Table of Contents

1. Introduction

1.1 Purpose

The purpose of this Website is to provide a platform for users to save their watched movies, and their watchlists. It also provides a movie recommendation system based on the user's watch history.

1.2 Document Conventions

This Software Requirements Specification (SRS) follows the following standards and typographical conventions:

- **Font**: The default font used throughout the document is Arial.
- **Highlighting**: Important terms or concepts are highlighted using bold or italic formatting.
- **Priority Inheritance**: It is assumed that the priorities assigned to higher-level requirements are inherited by detailed requirements unless explicitly stated otherwise.

1.3 Intended Audience and Reading Suggestions

This document is intended for the developers, testers, and project managers involved in the development of the Movie Recommendation System. It is also intended for the stakeholders who are interested in understanding the requirements of the system.

1.4 Project Scope

The Movie Recommendation System is a web-based application that allows users to save their watched movies, and watchlists. It also provides a movie recommendation system based on the user's watch history. The system will have the following features:

- **User Registration**: Users can create an account on the platform by providing their user name and password.
- **User Authentication**: Users can log in to the platform using their user name and password.
- **Movie Database**: The system will have a database of movies that users can search and add to their watched list or watchlist.
- **Watched List**: Users can add movies to their watched list and mark them as watched.
- **Watchlist**: Users can add movies to their watchlist to keep track of the movies they want to watch in the future.
- **Recommendation System**: The system will recommend movies to users based on their watch history.

1.5 References

- User Stories

2. Overall Description

2.1 Product Perspective

The Movie Recommendation System is a standalone web application that interacts with users through a web interface. It will have a backend server that handles user requests, stores user data, and provides movie recommendations. The system will interact with a movie database to fetch movie information.

2.2 Product Features

The Movie Recommendation System will have the following features: - User Registration - IMDB Movie Database - Watchlist - Recommendation System

2.3 User Classes and Characteristics

The system will have the following user classes: - **User**: A user who has created an account on the platform. - **Movie**: A movie that is stored in the system's database.

2.4 Operating Environment

The Movie Recommendation System will be a web-based application that can be accessed through a modern web browser. The system will be hosted on a web server that supports the required technologies.

2.5 Design and Implementation Constraints

The Movie Recommendation System will be implemented using the following technologies: - **Frontend**: HTML, CSS, JavaScript - **Backend**: Python Django - **Database**: SQLite

2.6 User Documentation

The Movie Recommendation System will have user documentation that explains how to use the system, create an account, add movies to the watched list, and get movie recommendations.

2.7 Assumptions and Dependencies

The Movie Recommendation System assumes that users have a basic understanding of how to use a web application and are familiar with concepts like creating an account, and adding items to a list. The system depends on the availability of the IMDB movie database for fetching movie information.

3. System Features

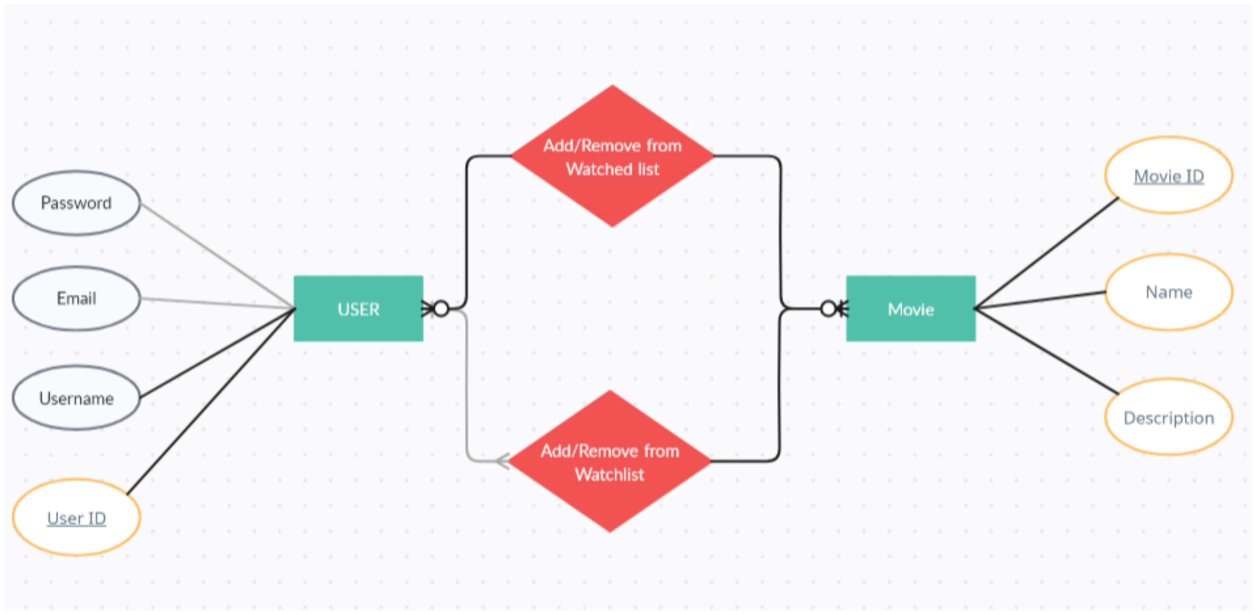


Figure 1: Overall System ERD

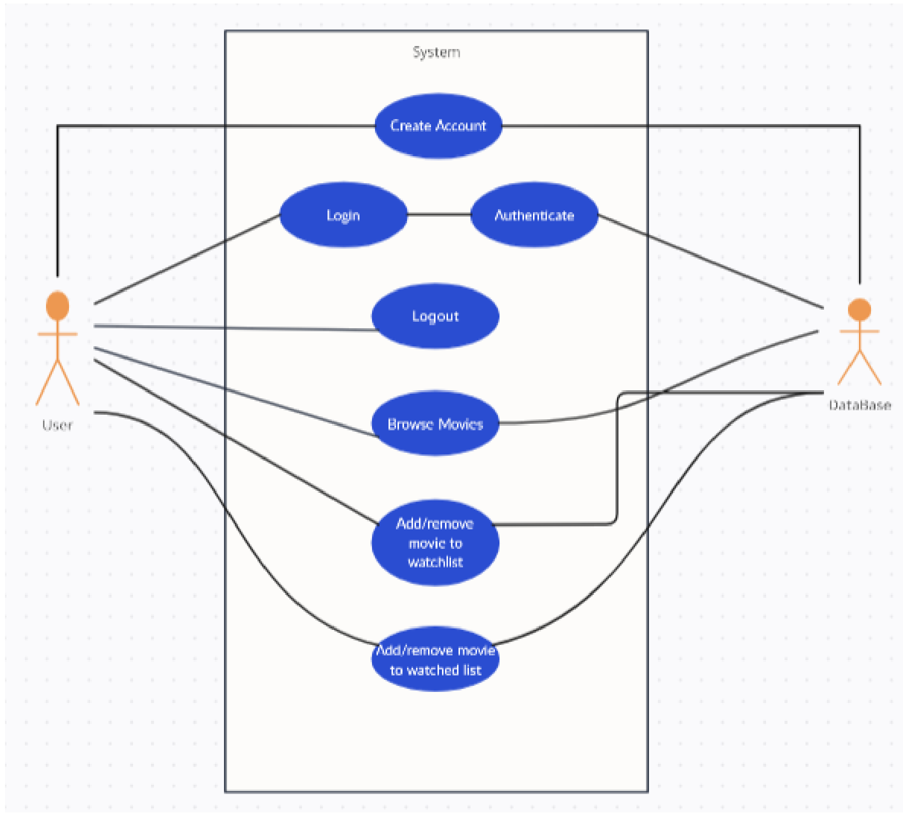


Figure 2: Overall System Usecase Diagram

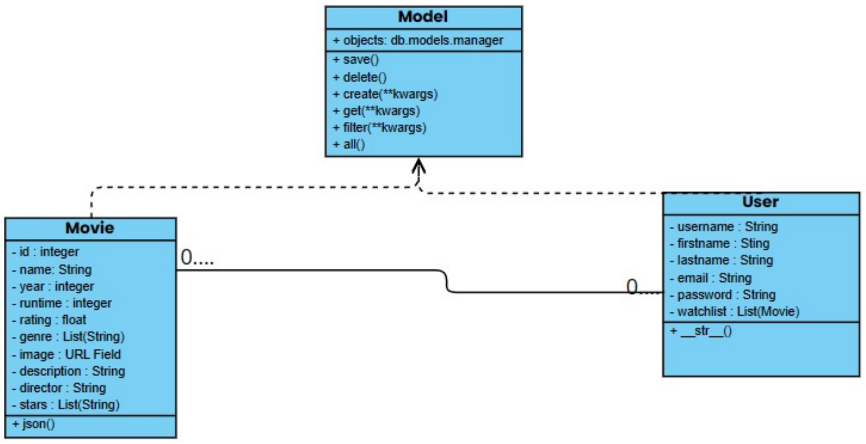
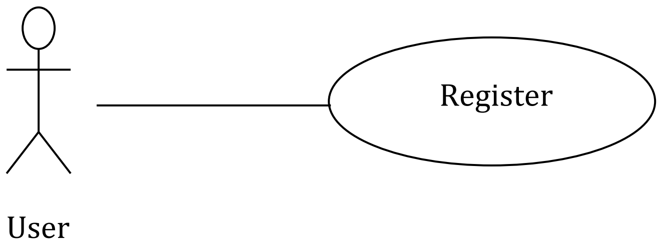


Figure 3: Overall System UML Class Diagram

3.1 User Registration



Description & Priority

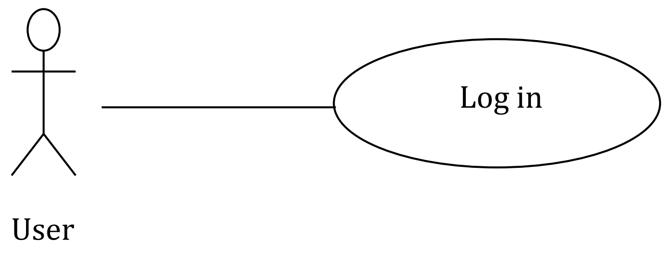
Users can create an account on the platform by providing their user name and password. This feature is of high priority as it is required for users to access the system.

Use Case #:	- MV-REG
Use Case name:	- Register
Purpose:	- To create new account.
Primary actor:	- Customer
Secondary actor:	- None
Input parameters:	- User name, Password

Output parameters:	- None
Precondition:	- Input parameters must be in appropriate format. - Both input parameters must be given.
Post-condition:	- If successful, a confirmation is sent to the user for successful Account creation.
Successful scenario:	1. User navigates to the registration page. 2. User enters their user name and password. 3. User clicks on the “Register” button. 4. System validates the user name and password. 5. System creates a new user account and redirects the user to the login page.
Exceptions:	- One or both parameters are missing. - One or both parameters are in incorrect format.
Functional requirements	- The system shall validate the user name format. - The system shall validate the password length.

3.2 User Login & Authentication

Description & Priority



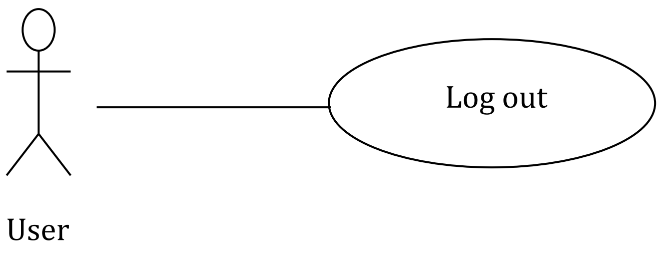
Users can log in to the platform using their user name and password. This feature is of high priority as it is required for users to access their account.

Use Case #:	- MV-L1
Use Case name:	- Login
Purpose:	- To login into an account.
Primary actor:	- Customer
Secondary actor:	- None
Input parameters:	- User name, Password
Output parameters:	- None
Precondition:	- Both Input parameters must be correct. - Both input parameters must be given.
Post-condition:	- If successful, a confirmation is sent to the user for successful login.
Successful scenario:	1. User navigates to the login page. 2. User enters their user name and password. 3. User clicks on the “Login” button. 4. System validates the user name

	and password. 5. System redirects the user to the home page.
Exceptions:	- One or both parameters are missing. - One or both parameters are in incorrect.
Functional requirements	- The system shall validate the user name and password. - The system shall authenticate the user.

3.3 User Logout

Description & Priority

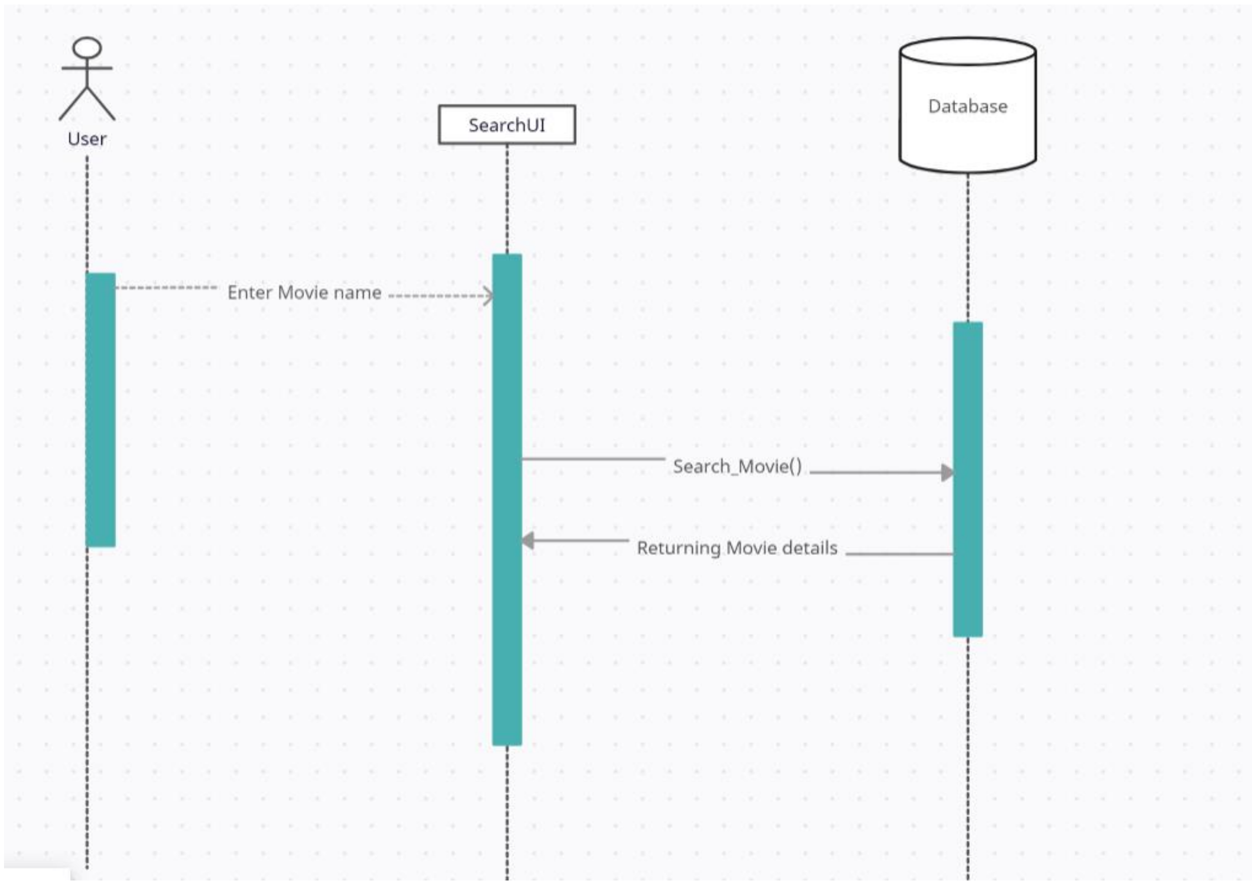


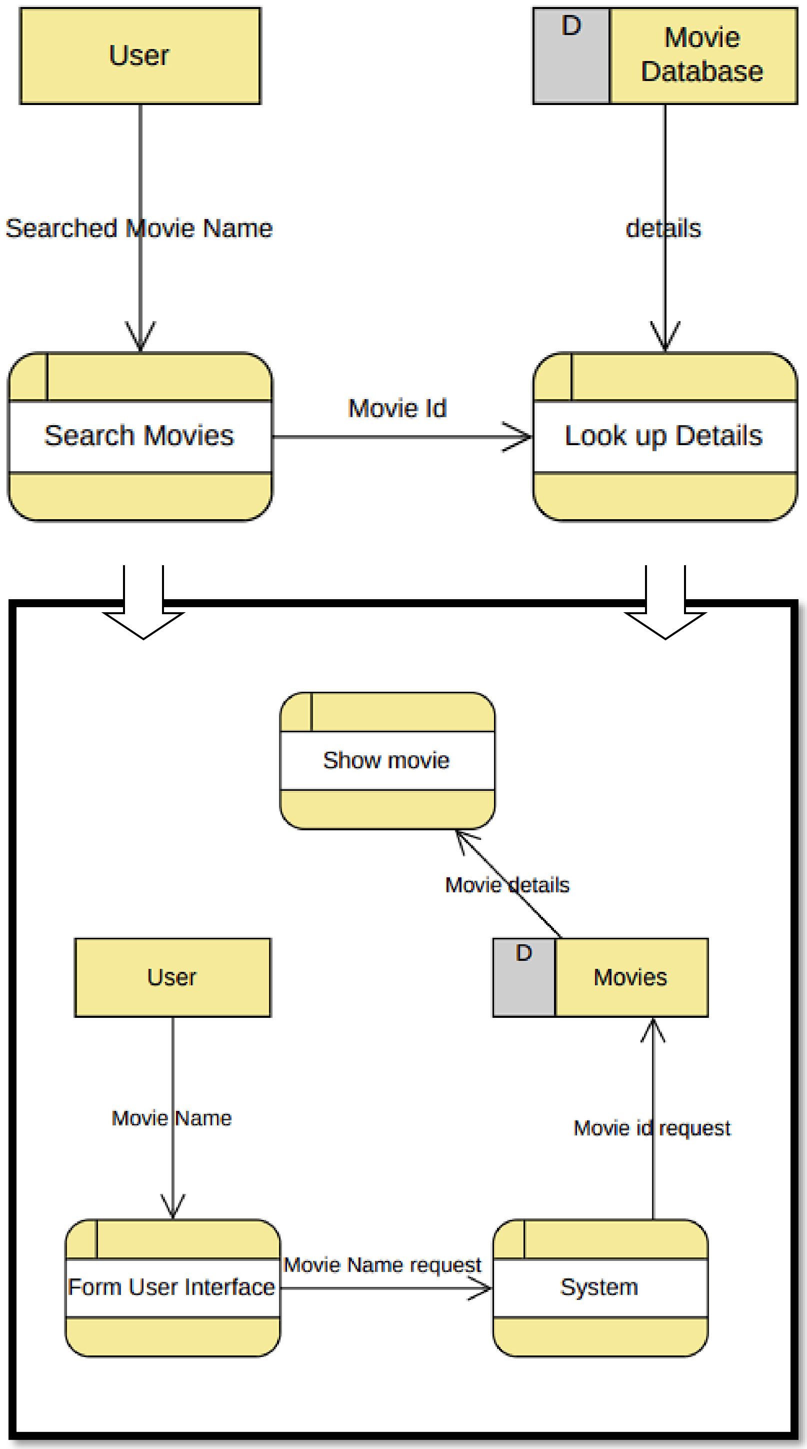
Users can log out from the platform. This feature is of high priority as it is required for users to access their account.

Use Case #:	- MV-L2
Use Case name:	- Logout
Purpose:	- To logout from an account.
Primary actor:	- Customer
Secondary actor:	- None
Input parameters:	- None
Output parameters:	- None

Precondition:	- User already logged in
Post-condition:	- If successful, a confirmation is sent to the user for successful logout.
Successful scenario:	1. User navigates to the home page. 2. User clicks on the "Logout" button. 3. System redirects the user to the login page.
Exceptions:	- User not logged in
Functional requirements	- None

3.4 Browse Movie Database




User
D
Movie Database
Searched Movie Name
details
Search Movies
Movie Id
Look up Details
Show movie
Movie details
User
D
Movies
Movie Name
Movie id request
Form User Interface
Movie Name request
System

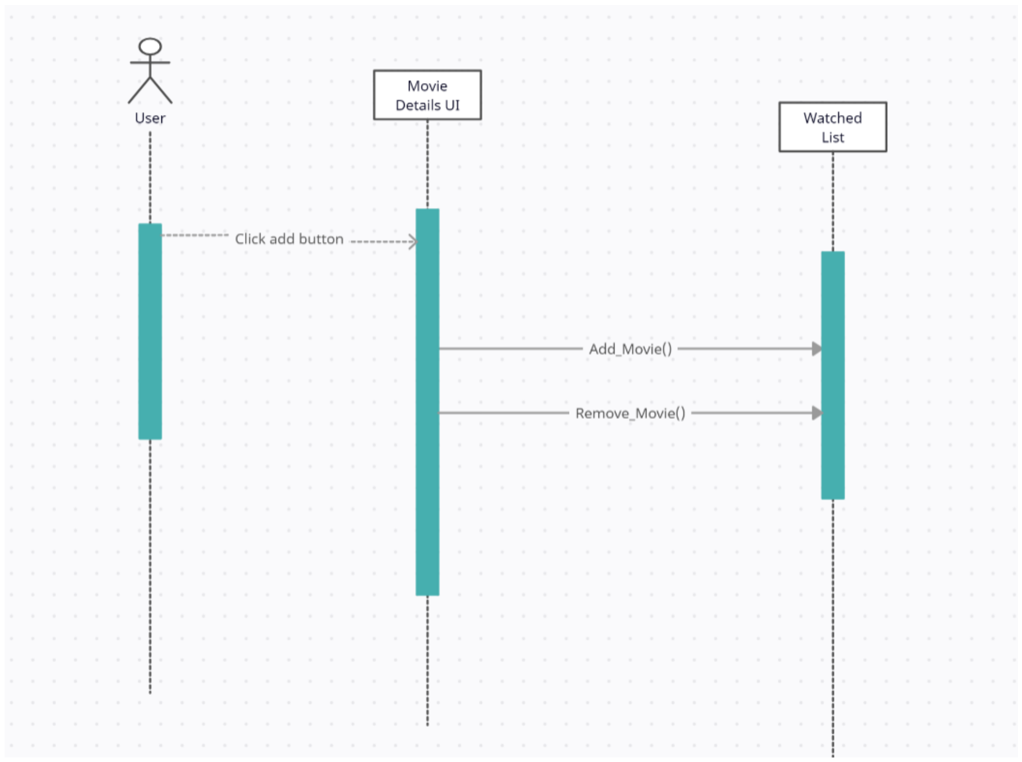
Description & Priority

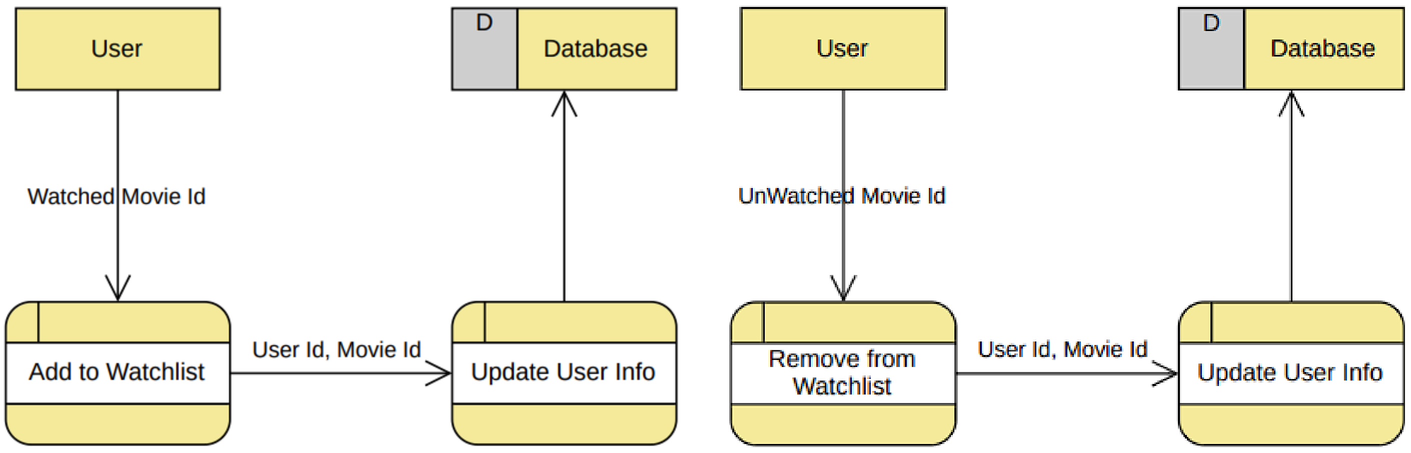
The system will have a database of movies that users can search and add to their watched list or watchlist. This feature is of high priority as it is required for users to interact with movies.

Use Case #:	- MV-MDB
Use Case name:	- Browse Movie Database
Purpose:	- To Search for Movies
Primary actor:	- Customer
Secondary actor:	- None
Input parameters:	- Movie name
Output parameters:	- Required Movie page - Related Movies
Precondition:	- User Logged in
Post-condition:	- If successful, the user will be sent to the required movie page
Successful scenario:	1. User navigates to the movie search page. 2. User enters a movie title in the search bar. 3. User clicks on the “Search” button. 4. System fetches movie information from the database. 5. System displays the movie information to the user.
Exceptions:	- Movie doesn’t exist in the database. - User not logged in

Functional requirements	- The system shall fetch movie information from the database. - The system shall display movie information to the user.

3.5 Add/Remove Movie to Watched List





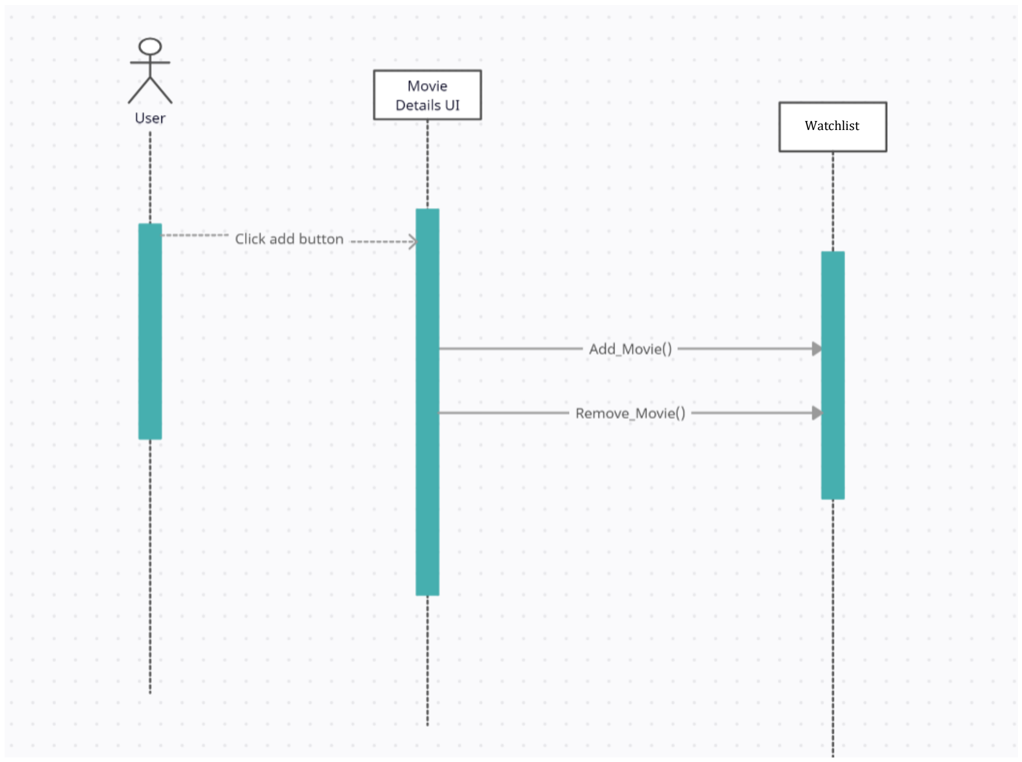
Description & Priority

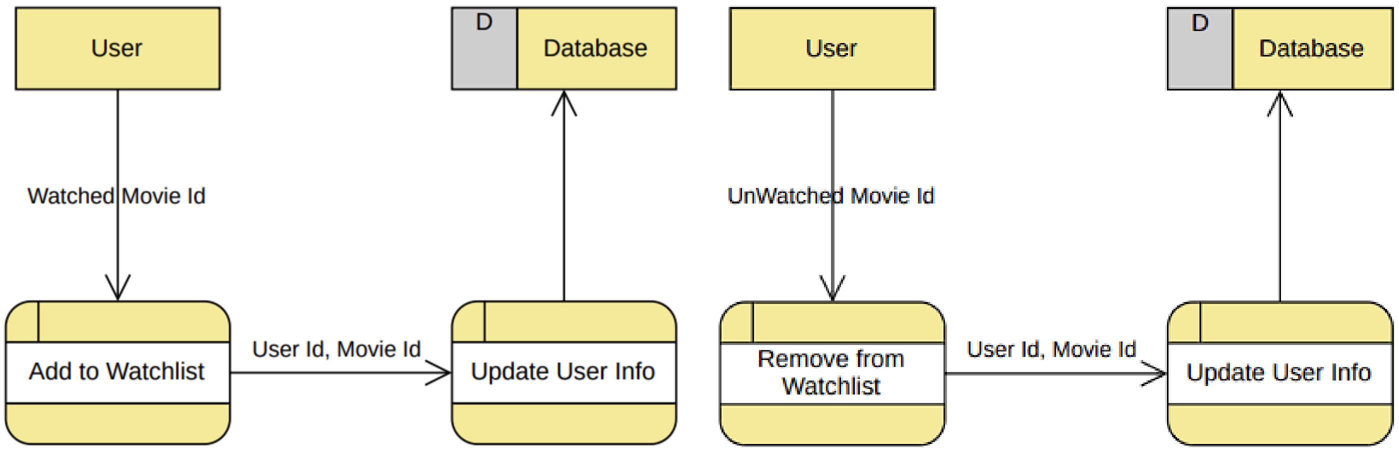
Users can add movies to their watched list and mark them as watched. This feature is of high priority as it is required for users to keep track of the movies they have watched.

Use Case #:	- MV-MWdL
Use Case name:	- Add/Remove movie to watched list
Purpose:	- To mark Movies as watched
Primary actor:	- Customer
Secondary actor:	- None
Input parameters:	- Movie name, Movie id
Output parameters:	- None
Precondition:	- User Logged in - Movie Exists in the Database
Post-condition:	- If successful, the Watched list will be updated to contain the required movie
Successful scenario:	1. User navigates to the movie details page. 2. User clicks on the "Add to Watched List" button. 3. System adds the movie to the user's watched list. 4. User marks the movie as watched. 5. System updates the movie status in the watched list.
Exceptions:	- Movie doesn't exist in the database.

	- Movie already in watched list - User not logged in.
Functional requirements	- The system shall add movies to the watched list. - The system shall mark movies as watched

3.6 Add Movie to Watchlist





Description & Priority

Users can add movies to their watchlist to keep track of the movies they want to watch in the future. This feature is of high priority as it is required for users to keep track of the movies they want to watch.

Use Case #:	- MV-MWL
Use Case name:	- Add/Remove movie to watchlist
Purpose:	- To mark Movies as to be watched
Primary actor:	- Customer
Secondary actor:	- None
Input parameters:	- Movie name, Movie id
Output parameters:	- None
Precondition:	- User Logged in - Movie Exists in the Database
Post-condition:	- If successful, the Watchlist will be updated to contain the required movie
Successful scenario:	1. User navigates to the movie details page. 2. User clicks on the "Add to Watchlist" button. 3. System adds the movie to the user's watchlist.
Exceptions:	- Movie doesn't exist in the database. - Movie already on watchlist - User not logged in.

Functional requirements	- The system shall add movies to the watchlist. - The system shall mark movies as to be watched

3.7 Recommendation System

Description & Priority

The system will recommend movies to users based on their watch history. This feature is of high priority as it is required for users to discover new movies.

Stimulus/Response Sequences

1. User navigates to the recommendation page.
2. System fetches user watch history.
3. System generates movie recommendations based on user data.
4. System displays movie recommendations to the user.

Functional Requirements

- The system shall fetch user watch history.
- The system shall generate movie recommendations based on user data.
- The system shall display movie recommendations to the user.

4. External Interface Requirements

4.1 User Interfaces

The Movie Recommendation System will have the following user interfaces: - **Registration Page**: Allows users to create an account by providing their user name and password. - **Login Page**: Allows users to log in to the platform using their user name and password. - **Home Page**: Displays the user's watched list, watchlist, and movie recommendations. - **Movie Search Page**: Allows users to search for movies in the database. - **Movie Details Page**: Displays detailed information about a movie and allows users to add it to their watched list or watchlist. - **Recommendation Page**: Displays movie recommendations based on the user's watch history.

4.2 Hardware Interfaces

The Movie Recommendation System will be a web-based application that can be accessed through a modern web browser. It does not have any specific hardware requirements.

4.3 Software Interfaces

The Movie Recommendation System will interact with the following software components:

The data items or messages coming into the system include user registration information, user login credentials, movie search queries, movie details requests, and user watch history. The purpose of each is to enable user authentication, movie search and retrieval, user interaction with movies (watched list, watchlist), and generating movie recommendations.

The services needed include user registration, user authentication, movie search, movie details retrieval, and movie recommendation generation. The nature of communications will involve HTTP requests and responses between the frontend and backend components of the system.

Data that will be shared across software components includes user account information, movie data from the IMDB database, and user watch history. The specific implementation details of data sharing are not provided in the code, so you may need to consider appropriate data models and database queries to facilitate data sharing between components.

4.4 Communication Interfaces

The Movie Recommendation System will communicate with users through a web interface. The system will use HTTP requests and responses to interact with the frontend and backend components. The communication will be stateless and will follow RESTful principles.

5. Other Nonfunctional Requirements

5.1 Performance Requirements

The Movie Recommendation System shall respond to user requests within 2 seconds. The system shall be able to handle 100 concurrent users without performance degradation.

5.2 Safety Requirements

The Movie Recommendation System shall not store sensitive user information in plain text. The system shall use secure hashing algorithms to store user passwords.

5.3 Security Requirements

The Movie Recommendation System shall encrypt user passwords using a secure hashing algorithm. The system shall use HTTPS to secure communication between the frontend and backend components.

5.4 Software Quality Attributes

The Movie Recommendation System shall be user-friendly and intuitive to use. The system shall be responsive and provide real-time updates to users. The system shall be reliable and available 24/7.