Software Requirements Specification

for

<Movie App Recommender System>

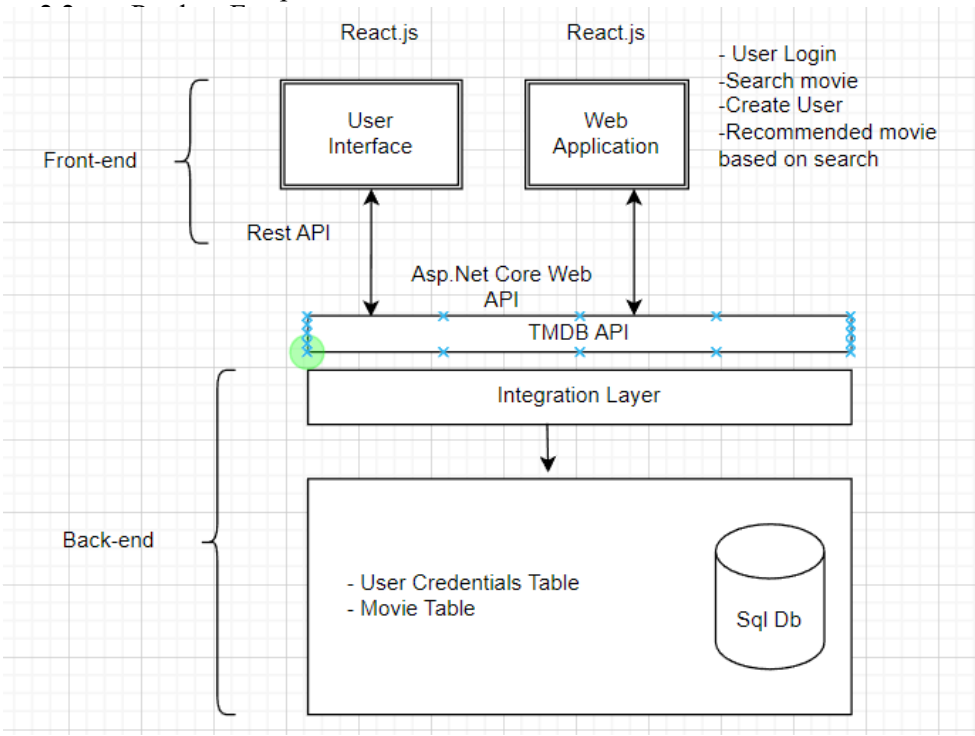
Version 1.0 approved

Prepared by <Ahmad Zulikmal Bin Zulkifli>

<Collabera>

<29th January 2023>

Table of Contents



Revision History

Name	Date	Reason For Changes	Version
Ahmad Zulikmal	29/1/2023		1.0

1. Introduction

1.1 Purpose

The product whose software requirements are specified in this document is a movie recommendation system. The revision or release number of the product is not specified in this statement.

The scope of the product that is covered by this SRS includes the following:

- Personalized movie recommendations to users based on their viewing history and preferences.
- A recommendation algorithm that utilizes machine learning to improve its accuracy and relevance.
- A user-friendly interface for browsing and selecting recommended movies.
- Integration with a database of movies and user data for recommendation generation and storage.
- Any other features, functionalities, and non-functional requirements that are defined in this SRS.

It is important to note that this SRS describes the entire system, including any subsystems that are necessary for the functioning of the movie recommendation system. If the system is part of a larger product and the SRS only describes the requirements of the movie recommendation system and not the entire product, this should be explicitly stated in the SRS.

1.2 Intended Audience and Reading Suggestions

The document is intended for several different types of readers, including:

- Developers: The developers will need to understand the technical requirements of the system in order to implement it. They will be interested in the functional and non-functional requirements, as well as the design constraints and assumptions.
- Project Managers: Project managers will need to understand the overall scope of the project and the timelines for delivery. They will be interested in the project overview, the functional and non-functional requirements, as well as the project schedule.
- Marketing Staff: Marketing staff will need to understand the system's features and capabilities in order to promote it to users. They will be interested in the functional requirements, as well as the user interface and usability requirements.

- Users: Users will want to understand how the system works and what it can do for them. They will be interested in the functional requirements, as well as the user interface and usability requirements.

- Testers: Testers will need to understand the requirements of the system in order to create test cases and test plans. They will be interested in the functional and non-functional requirements, as well as the design constraints and assumptions.

- Documentation Writers: Documentation writers will need to understand the requirements of the system in order to create user manuals and other documentation. They will be interested in the functional and non-functional requirements, as well as the user interface and usability requirements.

The SRS contains several sections that are organized in a logical sequence. It begins with an overview of the project, including the project scope and objectives. It then goes on to describe the functional and non-functional requirements of the system. Next, it covers the design constraints and assumptions, as well as the user interface and usability requirements. Finally, it includes the project schedule and budget.

Suggested sequence for reading the document:
1. Project Overview
2. Functional and Non-functional Requirements
3. Design Constraints and Assumptions
4. User Interface and Usability Requirements
5. Project Schedule and Budget

Depending on their role, readers should start by reading the sections most pertinent to them and then proceed to the rest of the document as needed.

1.3 Project Scope

The software being specified is a movie recommendation system. Its purpose is to provide personalized movie recommendations to users based on their viewing history and preferences. The benefits of this system include improved user engagement and satisfaction, as well as increased revenue from targeted movie recommendations. The main objectives and goals of the system are to provide accurate and relevant movie recommendations, as well as to continuously improve the recommendation algorithm through machine learning. The system aligns with corporate goals and business strategies by leveraging data and technology to enhance the user experience and drive revenue growth. A separate vision and scope document should provide more details on the long-term strategic product vision for the movie recommendation system.

2. Overall Description

2.1 Product Perspective

The context and origin of the product being specified in this SRS is that it is a new, self-contained product for providing personalized movie recommendations to users. The movie recommendation system is not a follow-on member of a product family or a replacement for certain existing systems.

The movie recommendation system is a standalone product that utilizes machine learning algorithms to provide personalized movie recommendations to users based on their viewing history and preferences. The system is designed to be integrated with a database of movies and user data, which it uses to generate and store recommendations.

A simple diagram that shows the major components of the overall system, subsystem interconnections, and external interfaces can be helpful to understand how the movie recommendation system fits into the larger context.

A simple diagram of the overall system could show the following components:

- A user interface for browsing and selecting recommended movies.
- A recommendation algorithm that utilizes machine learning to generate personalized recommendations.
- A database of movies and user data for recommendation generation and storage.
- External interfaces, such as integration with streaming platforms or other movie databases.

The user interface component of the system is connected to the recommendation algorithm component, which in turn is connected to the database component. The external interfaces are also connected to the recommendation algorithm component and database component.

It's important to note that this is a simple representation of the system and the actual system may be more complex and may have more components and interfaces.

2.2 Product Features

The major features of the movie recommendation system include:

- Personalized movie recommendations based on user viewing history and preferences
- Movie browsing and search functionality
- Integration with external streaming platforms and movie databases
- User account management and preferences settings
- Reporting and analytics on system usage and performance

A top level data flow diagram or a class diagram can be effective in illustrating the major groups of related requirements and how they relate in the system.

A simple data flow diagram of the system could show the following components:

- User Interface: The user interacts with the system through a web or mobile interface to browse and select recommended movies, search for movies, manage their account and preferences settings.
- Recommendation Algorithm: The system uses machine learning algorithms to analyze the user's viewing history and preferences to generate personalized recommendations.
- Database: The system uses a database to store information about movies, user data, and recommended movies.
- External Interface: The system integrates with external streaming platforms and movie databases to retrieve information about movies and user data.

In this diagram, the user's input flows into the recommendation algorithm, which generates personalized recommendations based on user data stored in the database. The recommendations are then displayed to the user via the user interface. The external interface allows the system to retrieve information from external streaming platforms and movie databases, which is then used to generate recommendations and store in the database.

It's important to note that this is a simple representation of the system and the actual system may be more complex and may have more components and interfaces.

2.3 User Classes and Characteristics

The various user classes that are anticipated to use the movie recommendation system include:

- Casual Users: These users are individuals who use the system infrequently, mostly to browse recommended movies and watch them on external streaming platforms. They have limited technical expertise and may have little to no experience with similar systems.
- Regular Users: These users are individuals who use the system frequently, mostly to browse recommended movies, watch them on external streaming platforms, and maintain their account and preferences settings. They have some technical expertise and may have experience with similar systems.
- Power Users: These users are individuals who use the system frequently and have a deep understanding of the system's capabilities. They use the system to browse recommended movies, watch them on external streaming platforms, maintain their account and preferences settings, and access the reporting and analytics features of the system. They have a high level of technical expertise and may have experience with similar systems.

It is important to note that the above user classes are not mutually exclusive and a user may fall into multiple categories.

The favored user class would be the Regular Users as they are the most frequent users of the system and have the most need for the system's capabilities. They are also the most likely to generate the most revenue for the system.

The Casual Users and Power Users will also be important user classes to satisfy, but their requirements may not be as critical as the Regular Users.

It's also important to take into account that the system should be designed to be easy to use for all user classes, with clear and intuitive navigation, appropriate feedback, and error messages.

For example, the system should provide clear instructions and guidance for Casual Users who may not be familiar with the system, while also providing advanced features and capabilities for Power Users who have a deep understanding of the system.

2.4 Operating Environment

The operating environment for the movie recommendation system built using ASP.NET and React.js can be described as follows:

- Hardware platform: The system can operate on a wide range of hardware platforms, including desktops, laptops, servers, or cloud-based infrastructure.
- Operating system: The system can run on various operating systems such as Windows, Linux, or macOS.
- Operating system versions: The specific versions of the operating system compatible with the system may be specified.
- Other software components: The movie recommendation system may be required to coexist with other software components or applications such as databases, web servers, or caching systems.
- Compatibility requirements: The system must be designed to work seamlessly with other software components and applications, without interfering with their performance or stability.

By specifying the operating environment in the SRS, the development team can ensure that the system is compatible with the customer's existing infrastructure and technology stack, and can be deployed and operated smoothly in the customer's environment.

2.5 Design and Implementation Constraints

Design and Implementation Constraints for the movie recommendation system built using ASP.NET (C#) and React.js can include:

- Corporate or regulatory policies: Developers may have to abide by specific policies set by the company or regulations, such as data privacy laws.
- Hardware limitations: The system may have limitations such as processing power, memory, and storage that may affect the performance of the recommendation system.
- Interfaces to other applications: Developers may have to consider the integration with existing applications and services that the recommendation system will be using.
- Specific technologies, tools, and databases: The recommendation system has to be built using specific technologies, tools, and databases as specified by the customer.
- Parallel operations: The system may need to perform multiple tasks simultaneously and efficiently.
- Language requirements: Developers must ensure that the code is written in a language that is supported by the customer's organization.
- Communications protocols: The recommendation system must use specific communication protocols for data exchange and must be compatible with the customer's existing systems.
- Security considerations: The system must be secure and protect user data. This includes protecting against unauthorized access, data breaches, and other security threats.
- Design and programming standards: The developers must follow specific design conventions and programming standards set by the customer's organization to maintain the code's quality and consistency.

These constraints will help guide the development team in making design and implementation decisions, ensuring that the system meets the customer's requirements and expectations and can be maintained by the customer's organization.

2.6 User Documentation

Here are the common user documentation components for a movie recommender system:

1. User manual: A comprehensive document explaining the features and functions of the system, including installation and setup instructions, usage guidelines and tips, and troubleshooting information.
2. On-line help: An interactive help system that provides context-sensitive information about the system, accessible from within the software interface.
3. Quick start guide: A brief document providing an overview of the system and step-by-step instructions for getting started with it.
4. Tutorials: Step-by-step, multimedia guides that demonstrate how to use the system and perform common tasks.

The user documentation may be delivered in formats such as PDF, HTML, and CHM (compiled HTML help), depending on the delivery method (print, web, or software). There is no specific standard for user documentation delivery formats for movie recommender systems.

2.7 Assumptions and Dependencies

Here are some assumed factors that could affect the requirements stated in the SRS for a movie recommender system:

1. Third-party or commercial components: The use of third-party libraries, APIs, or tools for specific functionalities could affect the performance, security, and scalability of the system.
2. Development environment: The availability of specific tools, hardware, or software needed to develop the system could impact the timeline, cost, and quality of the project.
3. User data sources: The availability, quality, and format of user data sources such as movie ratings, reviews, and genre information could affect the accuracy and relevance of the recommendations.
4. User preferences: The way in which user preferences are collected, stored, and used to generate recommendations could impact the system's performance and accuracy.
5. Privacy and security: The privacy and security of user data and recommendations could be affected by external factors such as data breaches or changes in regulations.
6. Interoperability: The compatibility of the system with other systems, devices, or platforms could impact its adoption and usage.
7. External dependencies: The project may have dependencies on external factors such as the reliability of internet connectivity, the performance of cloud-based services, or the availability of APIs.

3. System Features

System features refer to the distinct and individual components of a software system that together make up the system's overall functionality. These features are designed to perform specific tasks or provide specific services within the system, and often interact with other features to deliver the desired results. Examples of system features include user management, movie database,

recommendation engine, user interaction, reporting and analytics, security and privacy, and user notifications, as described in the previous answer.

3.1 Movie Recommender System Functional Requirements

1. User Management:
    a. User registration and login
    b. Profile management (edit/update personal information)
2. Movie Database:
    a. Movie information storage and retrieval (title, release date, genre, cast, etc.)
    b. Movie search functionality (by title, genre, release date, etc.)
3. Recommendation Engine:
    a. Personalized movie recommendations based on user preferences and viewing history
    b. Ability to provide different recommendation types (e.g. popular, recent, etc.)
4. User Interaction:
    a. Ability to rate and review movies
    b. Option to add movies to a watchlist or save favorite movies
5. Reporting and Analytics:
    a. Generate user behavior reports (most viewed movies, watchlist, etc.)
    b. Trend analysis on movie ratings and recommendations.
6. Security and Privacy:
    a. Secure user data management (hashed passwords, etc.)
    b. Compliance with data privacy regulations
7. User Notifications:
    a. Ability to send notifications to users (e.g. new movie releases, recommendations, etc.)
    b. Option for users to control notification preferences.

4. External Interface Requirements

4.1 User Interfaces

Movie Recommender System User Interface Characteristics:

1. Screen Layout:
    a. Consistent design across all screens with a clear navigation menu
    b. Large movie cover images and clear font sizes
    c. User profile section with easy access to personal information and preferences
    d. Clear and concise information on movie details (title, release date, genre, cast, etc.)
2. GUI Standards:
    a. Adherence to established design standards (e.g. Material Design, etc.)
    b. Use of recognizable icons for actions (e.g. add to watchlist, save, etc.)
    c. Responsive design for optimal viewing on different devices (desktop, tablet, mobile)
3. Buttons and Functions:
    a. Standard buttons for actions such as "search", "filter", "sort", "rate", "review", etc.
    b. Help button with access to user manual or support resources
    c. Option to sign up/login from every screen
4. Keyboard Shortcuts:

   a. Option for users to navigate the application using keyboard shortcuts
   b. Shortcuts for commonly used actions (e.g. search, filter, sort, etc.)
5. Error Message Display:
   a. Clear and concise error messages displayed in case of any errors or invalid input
   b. Option for users to report errors or request support
6. Components with User Interface:
   a. User management (registration, login, profile management)
   b. Movie database (search, movie information display)
   c. Recommendation engine (personalized recommendations)
   d. User interaction (rating, reviewing, watchlist)
   e. Reporting and analytics (user behavior reports, trend analysis)
   f. Security and privacy (data management, privacy policy)
   g. User notifications (notifications, notification preferences)

Note: The detailed design of the user interface should be documented in a separate user interface specification to ensure consistency in design and functionality.

4.2 Hardware Interfaces

Movie Recommender System Hardware Interface Characteristics:

1. Supported Devices:
   a. Desktop computers
   b. Laptops
   c. Tablets
   d. Smartphones
   e. Smart TVs
2. Data and Control Interactions:
   a. Data transfer between the software and hardware components (e.g. movie database, user information)
   b. Remote control of hardware components (e.g. smart TV) from the software
   c. Interaction with hardware components for multimedia playback (e.g. sound, video)
3. Communication Protocols:
   a. HTTP and HTTPS for data transfer and communication with hardware components
   b. JSON or XML for data exchange between software and hardware components
   c. Streaming protocols such as RTMP, HLS, or MPEG-DASH for multimedia playback

Note: The hardware interface specifications should include detailed information on the supported devices, data and control interactions, and communication protocols to ensure compatibility and seamless operation between the software and hardware components.

4.3 Software Interfaces

Movie Recommender System Software Component Connections:

1. Databases:
   a. Connection to movie database (e.g. IMDb, TMDB, etc.) for movie information and metadata
   b. User information database for storing user profiles and preferences

   c. Connection to movie ratings database (e.g. Rotten Tomatoes, Metacritic, etc.) for movie ratings
2. Operating Systems:
   a. Support for major operating systems (Windows, MacOS, Linux)
   b. Support for mobile operating systems (iOS, Android)
3. Tools and Libraries:
   a. Integration with machine learning libraries (e.g. TensorFlow, PyTorch, etc.) for recommendation engine
   b. Integration with data visualization tools (e.g. Matplotlib, Plotly, etc.) for reporting and analytics
   c. Integration with database management systems (e.g. MySQL, PostgreSQL, etc.) for data management
4. Integrated Commercial Components:
   a. Integration with streaming services (e.g. Netflix, Amazon Prime, etc.) for movie playback
   b. Integration with payment gateways (e.g. Stripe, PayPal, etc.) for subscription management
5. Data In and Out:
   a. Incoming: movie information, movie ratings, user information, user preferences, payment information
   b. Outgoing: personalized movie recommendations, movie playback, user reports and analytics, user notifications
6. Services Needed and Communications:
   a. API for communication between software components (e.g. movie database API, recommendation engine API, etc.)
   b. RESTful API for data transfer between components
   c. Real-time communication for movie playback and user interaction
7. Data Sharing:
   a. User information and preferences shared across software components for personalized recommendations
   b. Movie information shared across software components for movie details and metadata
   c. User ratings shared across software components for recommendation engine

Note: The detailed specifications for the software component connections should include the specific version of the software components, the nature of communications, and any implementation constraints to ensure seamless operation and data sharing between components.

4.4 Communications Interfaces

Communication interfaces in Software Requirement Specification (SRS) refers to the methods and protocols used for communication between different components and systems. It defines the input/output and data exchange requirements between the system and other systems, as well as the communication between different parts of the system itself. Below are communication standards that will be used:

- E-mail: The system may require sending notifications and updates to users via e-mail. SMTP protocol can be used for sending e-mails.
- Web browser: The system may be accessed through a web browser using React.js for the front-end. The back-end can be built using ASP.NET with C#.

- Network server communications protocols: The system may communicate with other servers or APIs (e.g. IMDb, TMDB) to gather movie data. REST or GraphQL can be used for API communication.
- Electronic forms: The system may require electronic forms for user input, such as rating a movie or providing feedback. These forms can be implemented in React.js.
- Message formatting: JSON or XML can be used for message formatting.
- Communication standards: HTTP or HTTPS can be used as communication standards.
- Communication security and encryption: SSL/TLS encryption can be used for secure communication and to protect sensitive user data.
- Data transfer rate: The rate of data transfer will depend on the network speed and system architecture.
- Synchronization mechanisms: The system may require data synchronization mechanisms to ensure that all movie data is up-to-date. Delta syncing or periodic full syncing can be used. This can be implemented using C# and SQL Server for database management.

5. Other Nonfunctional Requirements

5.1 Performance Requirements

These performance requirements should be used as guidelines for the developers to make suitable design choices and ensure that the system meets the desired level of performance. The requirements may be subject to change based on the specific requirements of the system. The performance requirements:

- Response time: The system should have a fast response time, with page loading time not exceeding 2 seconds on a fast network connection. This is important to ensure a good user experience.
- Recommendation generation time: The time taken to generate recommendations should not exceed 5 seconds, even with a large number of users and movie data. This is to ensure that recommendations are generated in a timely manner.
- Scalability: The system should be scalable to accommodate an increasing number of users and movie data. It should be able to handle at least 10,000 concurrent users without any performance degradation.
- Load balancing: The system should be designed for load balancing to distribute the load evenly across multiple servers and prevent a single server from becoming a bottleneck.
- Data storage: The system should be able to store and retrieve a large amount of movie data efficiently. The database should be optimized for fast read and write operations.
- API response time: The time taken to retrieve movie data from external APIs should not exceed 2 seconds. This is to ensure that the system can retrieve data in a timely manner.
- Error handling: The system should be designed to handle errors and exceptions gracefully, without affecting the overall performance of the system.
- Memory usage: The system should be optimized for efficient memory usage, with no memory leaks or excessive usage.
- Real-time systems: If the system requires real-time functionality, such as live streaming, specify the timing relationships and requirements for real-time communication. For example, the system should be able to handle live streaming with a maximum latency of 2 seconds.

5.2 Safety Requirements

Safety requirements typically include requirements related to data protection and privacy, error handling, and compliance with relevant laws and regulations. These requirements are essential to ensure the safe and secure use of the system and to prevent potential harm or damage that may result from the use of the product. They are usually specified in the SRS document and are used by developers to guide the design and implementation of the system to meet these requirements:

- Data protection: The system should be designed to protect sensitive user data, such as passwords and personal information, from unauthorized access and potential harm. SSL/TLS encryption can be used for secure communication and to protect sensitive data.
- User privacy: The system should be designed to respect user privacy and not collect or store any data without the user's consent. The system should also provide a clear and concise privacy policy that outlines how user data is collected, stored, and used.
- External policies: The system should comply with relevant data protection and privacy laws, such as the General Data Protection Regulation (GDPR) in the European Union.
- Safety certifications: If applicable, the system should meet any safety certifications required by relevant regulatory bodies.
- Error handling: The system should be designed to handle errors and exceptions gracefully, without causing any harm or damage to the system or users.

These requirements are important to ensure the safety and security of the system and to protect users from potential harm or damage that may result from the use of the product. The developers should take these requirements into consideration when designing the system and implement appropriate safeguards to meet these requirements.

5.3 Security Requirements

Security requirements in a Software Requirements Specification (SRS) are a set of non-functional requirements that outline the measures that must be taken to ensure the security and privacy of the system and to protect sensitive data and user information. Security requirements typically include requirements related to data protection, user authentication, privacy policies, compliance with laws and regulations, and security certifications. These requirements are essential to prevent unauthorized access, theft, or misuse of sensitive data and to ensure the secure use of the system. They are usually specified in the SRS document and are used by developers to guide the design and implementation of the system to meet these requirements.

- Data protection: The system should be designed to protect sensitive user data, such as passwords and personal information, from unauthorized access and potential harm. SSL/TLS encryption can be used for secure communication and to protect sensitive data.
- User authentication: The system should provide user authentication mechanisms to ensure that only authorized users can access the system and their personal data.
- Privacy policy: The system should provide a clear and concise privacy policy that outlines how user data is collected, stored, and used.
- Compliance with laws and regulations: The system should comply with relevant data protection and privacy laws, such as the General Data Protection Regulation (GDPR) in the European Union.
- Security certifications: If applicable, the system should meet any security certifications required by relevant regulatory bodies.

These requirements are important to ensure the security and privacy of the system and to protect users and their data from potential harm or damage. The developers should take these requirements into consideration when designing the system and implement appropriate safeguards to meet these requirements.

5.4 Software Quality Attributes

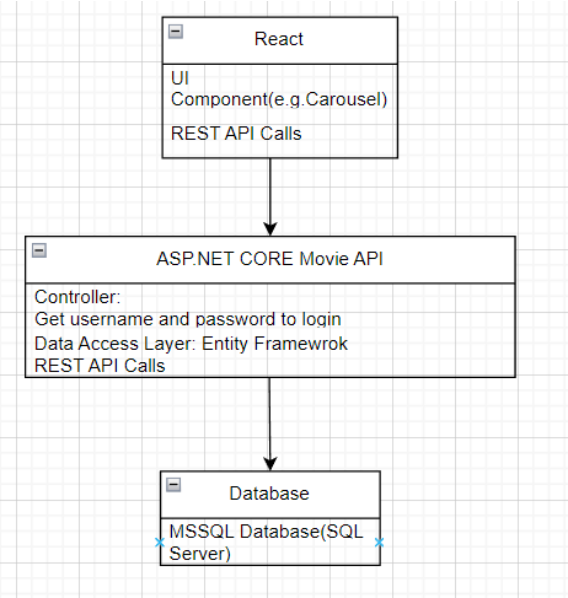
Software Quality Attributes (SQA) in a Software Requirements Specification (SRS) are a set of non-functional requirements that describe the desired quality characteristics of the software product. SQA describe the attributes of the software system that are important to its customers, users, or stakeholders, such as usability, reliability, performance, security, and maintainability. These quality attributes help to define the expected user experience and the desired level of performance of the software system. They are specified in the SRS and serve as a basis for the design, development, and testing of the software. The SQA are typically evaluated and tested during the development process to ensure that the software meets the specified quality requirements and provides the desired level of performance, usability, and security.

- Usability: The system should be user-friendly, intuitive, and easy to navigate, with a clear and simple interface. The system should also allow users to easily search for and find movies, save and retrieve favorite movies, and view movie recommendations.
- Reliability: The system should be reliable and available 24/7, with a high degree of uptime and minimal downtime. The system should also be able to handle large amounts of traffic and data with minimal latency and lag.
- Scalability: The system should be scalable to accommodate increasing amounts of data and traffic as the user base grows.
- Performance: The system should provide fast and responsive recommendations, with minimal latency and wait times. The system should also be able to handle high volumes of requests and data efficiently.
- Security and privacy: The system should provide a secure and private environment for user data, with strong user authentication and data protection mechanisms.
- Compliance with standards: The system should comply with relevant coding and design standards, such as OWASP, to ensure secure and reliable implementation.

These quality characteristics are important to ensure that the system meets the needs of its users and provides a positive user experience. The developers should prioritize these characteristics in their design and implementation decisions and measure their performance against these criteria to ensure that the system meets these requirements.

Appendix A: High-level and low-level diagram

Low-level diagram



High-level diagram