Software Requirements Specification

for

Emotion Based Movie Recommender

Version 1.0 approved

Prepared by
Dhruv Mishra
Ripu Daman Singh Bankawat
Sanyam Kabra
Shreekar
Tavishi Srivastava
Ekta Saini

Computer Science Department
IIT- Jodhpur

29th January 2025

Table of Contents

Revision History

Name	Date	Reason For Changes	Version

1. Introduction

1.1 Purpose

The Emotion-Based Movie Recommender is a web application designed to suggest movies to users based on their emotions and preferences. This system utilizes machine learning to analyze user inputs, such as responses to a questionnaire or text prompts, and recommends movies accordingly. By eliminating the need for users to manually search for films, the recommender system enhances the entertainment experience by providing personalized movie suggestions. This document outlines the software requirements for the first version (1.0) of the system.

1.2 Document Conventions

This document follows standard IEEE formatting for Software Requirements Specification (SRS) documents. Headings and subheadings are numbered for easy reference, and key terms are in bold for emphasis. Functional and non-functional requirements are specified clearly using bullet points and structured sections.

1.3 Intended Audience and Reading Suggestions

This document is intended for multiple stakeholders, including:

- **Developers**: To understand the system architecture, constraints, and requirements for implementation.
- **Project Managers:** To ensure alignment with project goals and deliverables.
- **Testers:** To verify the correctness and completeness of the system functionalities.
- **End-users and Clients:** To gain insights into system capabilities and expected functionalities.

Readers should begin with the introduction and product overview sections before delving into the technical details in later sections.

1.4 Product Scope

The Emotion-Based Movie Recommender aims to provide users with personalized movie suggestions based on their mood and preferences. The system supports two main recommendation methods:

- A **questionnaire-based approach**, where users answer questions about their mood and preferences.
- A **text-based input**, where users describe their feelings or expectations, and AI extracts key insights to recommend suitable films.

The system maintains a user-specific recommendation history, ensuring improved recommendations over time. This recommender system aligns with modern trends in AI-driven personalization and enhances user engagement in digital entertainment.

1.5 References

This SRS references the following resources:

- Relevant IEEE and software engineering standards.
- API documentation for third-party emotion recognition and movie databases.
- Flask framework and associated Python libraries for web development.
- UI/UX guidelines for web-based recommendation systems.

2. Overall Description

2.1 Product Perspective

The emotion based recommender system is a new, self-contained product that aims to use artificial intelligence to avoid the task of manually searching for movies to watch by instead using artificial intelligence to recommend a list of movies instead.

2.2 Product Functions

The major functions of the product are :-

- *Recommending movies based on input to a series of questions such as genre preference, mood preference etc.*
- *Recommending movies based on prompt entered by user detailing the type of movie the user wants to watch.*
- *Displaying link to the recommended movies so that user can directly reach the trailers of the recommended movies.*
- *Maintaining a database of previously recommended movies unique to each user.*

2.3 User Classes and Characteristics

The various user classes that may find this product to be useful are:-

- **General Users (End-users):**

  Description:- Individuals seeking to use the product to get movie recommendations. The main target user class for this product.

  Frequency of use :- Regular, expected to be twice a week per user on average

  Technical expertise :- Basic, no advanced skills required

  Functions used :-

1. Emotion input through quizlet/ text prompt
2. Viewing recommended movies
3. Viewing trailer of recommended movies

- **Developers :**

  Description :- Software engineers responsible for system development and maintainence of software.

  Frequency of use :- Regular during development and maintainence phases

  Technical Expertise:- Expert-level; proficient in programming, software architecture, and debugging.

  Functions Used:

  1. Developing new features
  2. Fixing bugs and optimizing performance
  3. Integrating new technologies

2.4 Operating Environment

The Emotion based movie recommender will be developed as a web=application using the FLASK framework . The operating environment includes the following components :-

- Hardware platforms :-
  1. standard web servers
  2. User devices :- laptops, desktops, mobile phones etc.
- Operating systems :-
  1. Server Side :- Linux(Debian)
  2. Client – side :- Cross Platform (Windows, MacOS, linux etc.)
- Browser Compatibility :- all modern browsers like Chrome , Edge , Mozilla Firefox etc.

2.5 Design and Implementation Constraints

The development of the **Emotion-Based Movie Recommender System** will face the following constraints:

- **Technological Constraints:**

  1. Use of FLASK as primary web framework
  2. Use of python as backend programming language
  3. HTML , CSS and Javascript for frontend development
  4. Compatibility with modern browsers
- **Hardware Constraints:**

1. Performance may vary based on user-device capabilities like internet connectivity
2. Memory limitations from server-side due to limited memory available to host the server

- **Security Considerations:**
  1. Secure handling of personal data, especially emotion-related and behavioural data.
  2. Implementation of HTTP protocols for data transmission.

2.6 User Documentation

The user will get user manual in form of a webpage bundled with the main website.

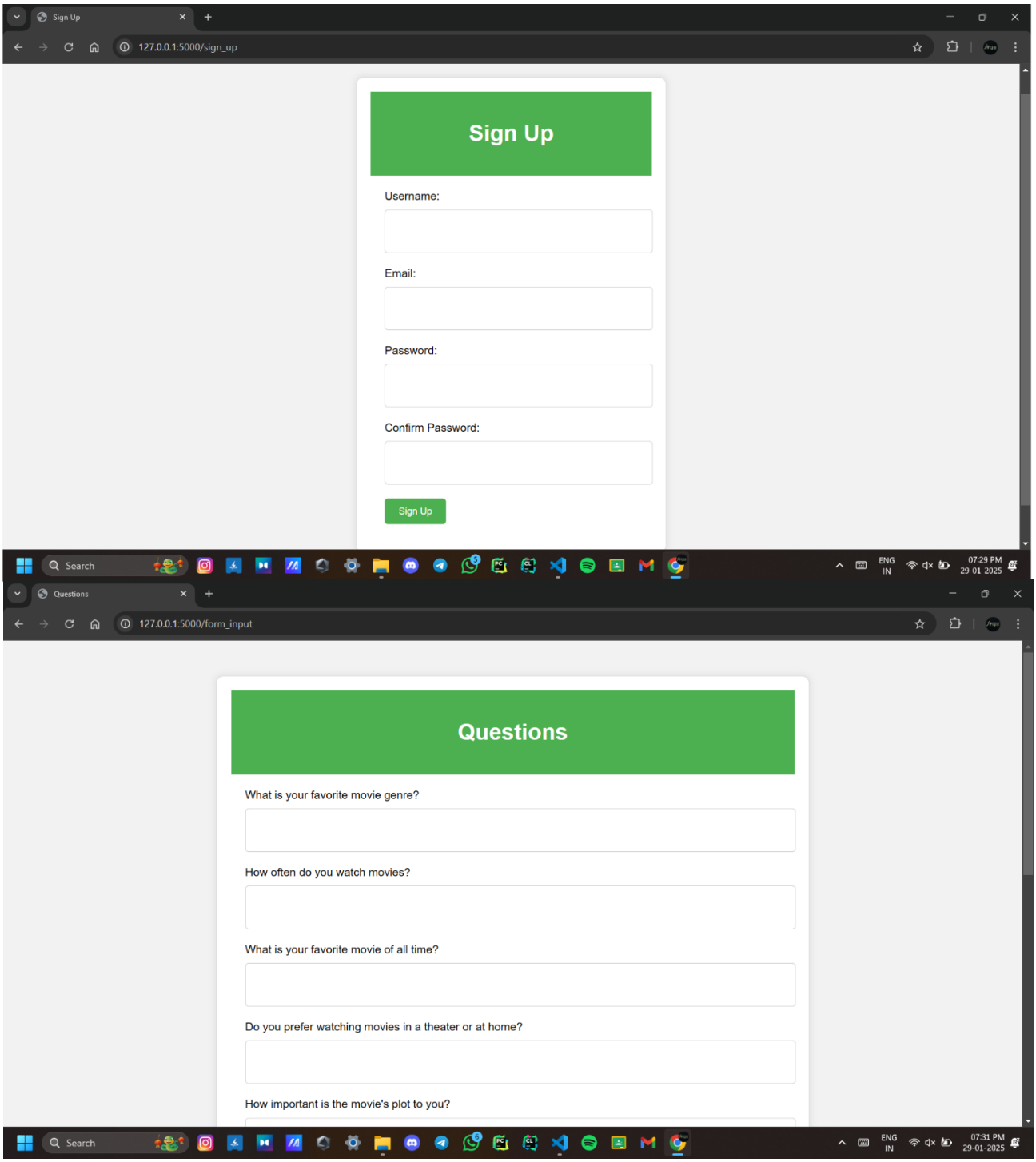
2.7 Assumptions and Dependencies

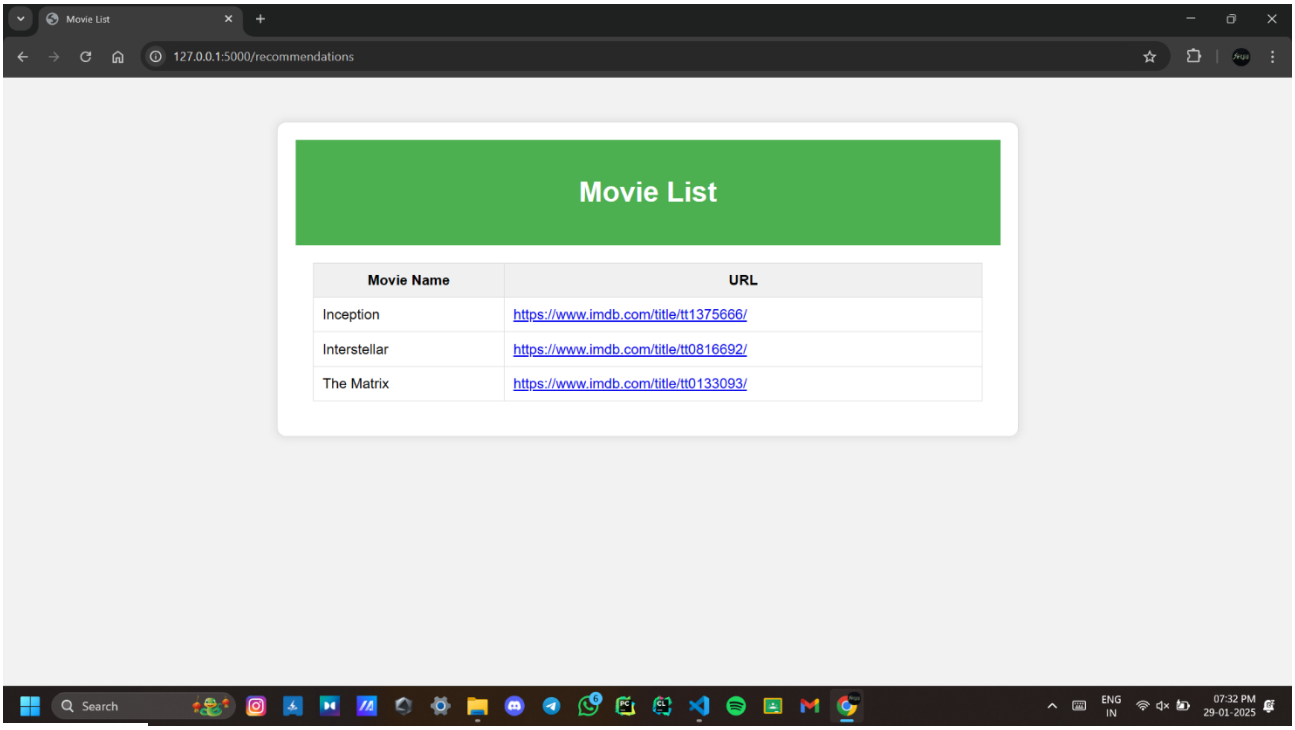
- Assumptions :
  1. Users will have stable internet access to interact with the web application.
  2. Third-party APIs for emotion recognition and movie databases will be reliable and well-documented
  3. Users will access the system using up-to-date web browsers that support modern web technologies.
- Dependencies :-
  1. Dependency on third-party APIs for emotion analysis and movie recommendations.
  2. Reliance on the Flask framework and associated Python libraries for web development.

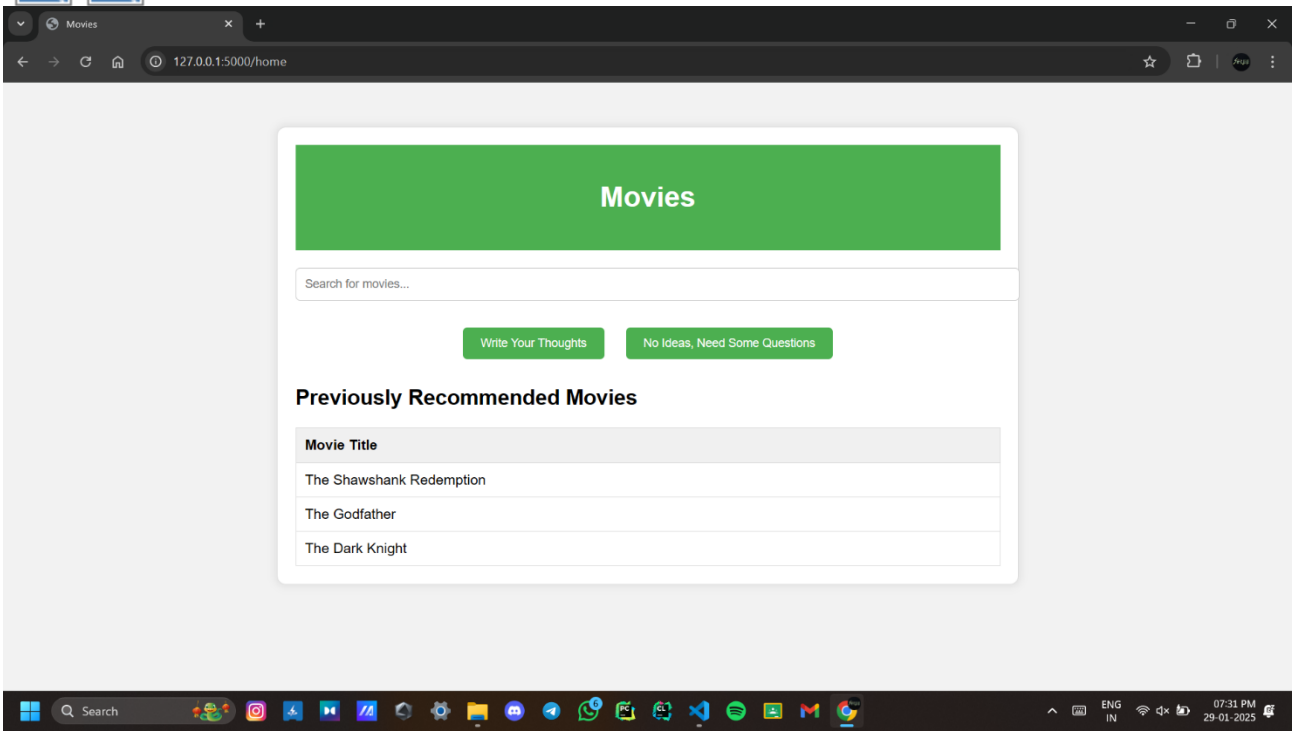
These assumptions and dependencies are critical for the development of this software to an acceptable status.

3. External Interface Requirements

3.1 User Interfaces







Requirements: UI must be easy to understand for new users. UI should look clean and minimal

Landing Page: A welcoming page with options to start the mood assessment.

Mood Assessment Interface: A sequence of questions presented via forms or multiple-choice selections. Users may also input text responses.

Recommendation Display Page: A dynamically generated page displaying movie recommendations based on the user's mood, including descriptions, trailers, and ratings.

Navigation and Controls:
Standard buttons: "Next", "Back", "Submit", "Retry"

Help and guidance tooltips:
Consistent error message formatting and validation messages for invalid inputs

Dark and light mode support

3.2 Hardware Interfaces

Client-side requirements:
A device with an internet connection (desktop, laptop, tablet, or smartphone) and capable of running a chromium based web browser

Server-side requirements:
A cloud-based or dedicated server for hosting the web application and handling backend processing

GPU-accelerated computation for the ML model (if applicable for real-time processing)

A database server for storing user data and movie recommendations (SQL/Mongo DB)

Communication protocol:
http protocol to communicate between server and frontend

3.3 Software Interfaces

Frontend Technologies*: HTML, CSS, JavaScript*

Backend Technologies*: Python (Flask) , (express.js?)*

Database*: MySQL/MongoDB for structured data storage*

Machine Learning Model*:*
Developed in Python using TensorFlow/PyTorch/Scikit-learn
Hosted on the backend server or a cloud-based inference engine (e.g., AWS Lambda, Google Cloud AI)

APIs and External Services*:*
OpenAI/GPT API or similar for NLP-based mood analysis
TMDB API or IMDb API for fetching movie details
Speech-to-text API (Google Speech-to-Text, Mozilla DeepSpeech) for voice input processing(optional)

Operating System Compatibility*:*
The application will run on any OS with a modern web browser (Windows, macOS, Linux, Android, iOS)

3.4 Communications Interfaces

***Web Communication**:*

HTTP/HTTPS for frontend-backend communication

***External API Calls**:*

Secure API calls to fetch movie details, perform NLP-based mood analysis, and process speech input

***Data Transfer and Storage**:*

JSON format for API request/response data exchange
SQL database to store User data.

***Error Handling and Logging**:*

Logging mechanisms for debugging and monitoring API requests/responses

4. System Features

4.1 About Page

4.1.1 Description

The software's opening page is the "About" page, which gives the user a brief idea as to what the software is all about, what facilities the user can avail from the software and a short **"How to Operate"** manual for the user.

This feature is important as it eases the process of operating the software for the user. It will prove to be a solution to a large number of user queries/doubts regarding the inside features of the software and their functionalities.

4.1.2 Functional Requirements

This page does not require the user to perform any action. It is just for the user to read and get a hang of the overall working of the software. This would help the user decide whether he/she wants to further use the software. Accordingly, the user can choose the Sign-up/Login or Back options.

4.2 Sign Up/Login

4.2.1 Description

The "About" Page is followed by the Sign-up/Login Page. If the user is visiting the website for the first time, they will have to sign up with their details. If the user has already signed up once on the website and is revisiting it, they will only have to Login with their username and password.

4.2.2 Stimulus/Response Sequences

As soon as the user enters his/her details in the appropriate fields, they are logged into the system and are shown the Home Page of the software.

4.2.3 Functional Requirements

Req.1. If the user is not already a member, and he/she clicks on the Login Button and enters some details, then, an error message should be displayed stating that the user should sign up first and then login.

Req.2. Once the user has signed up for the software, a check should be available which should check that the exact same user details are not used for a new sign up. If the same user details are used for a new sign up, an error message should be popped up saying that the user already exists.

Req.3. For a user who has signed up, on choosing the Login option and entering the correct details, he should be directed to the Home page of the software. If any of the user credentials do not match with what he/she had signed up with, an error message regarding the same should be displayed.

4.3 Question/Answers Page

4.3.1 Description

On the Home Page, the user will be able to choose from one of the two options: Question/Answers based search OR Prompt based search. The Question/Answers feature comprises some built-in questions which will be asked to the user and user will have to select an answer from one of the options in the drop-down. Based on the user's answers, apt movies will be recommended to him/her.

4.3.2 Stimulus/Response Sequences

1) Questions will be asked to the user in order to identify his/her mood and their likes/dislikes and preferred genre of movies.

2) On the basis of the user's answers to the questions asked, the software will search for a movie which best suits the mood of the user at that moment.

3) Three such movies will then be recommended to the user and a link to the trailer of the movies will also be provided.

4) The recommended movies will be stored as history for the user, which would help in searching for appropriate movies for that particular user in future.

4.3.3 Functional Requirements

Req.1. Answers to all questions are mandatory. If a field is left unanswered, an error message should be displayed pointing out the field which is empty.

Req.2. If the software is able to find suitable movies, they should be displayed on the screen, else, a message stating "No Matching Movies Found!" should be displayed.

4.4 Prompts Input Page

4.4.1 Description

The second option on the Home Page is the Prompt-Based search, wherein, the user will be free to write a sentence/paragraph expressing his/her feelings, mood and likes/dislikes in their own words and the software would then make a search for the best movie matches and recommend them to the user.

4.4.2 Stimulus/Response Sequences

The user would write down their feelings in the prompt box and this should stimulate the software to find out key words which would help it to identify the user's mood and search for movies based on it.

4.4.3 Functional Requirements

Req.1. If the user enters something which does not provide the software enough information that it is able to figure out the user's mood, a request message asking for more specific prompt should be shown to the user.

Req.2. If the software is able to identify the mood and preferred genre of the user and finds out suitable movies, they should be recommended to the user. If mood is identified but no perfect movie matches are found, an error message stating "No Matching Movies Found!" should be displayed.

4.5 Movie Recommendation Page

4.5.1 Description

This is the final page where three movies (with the links of their trailers) matching the user's mood will be displayed. The movies displayed here as recommendations will be stored in the user's "Recommendation History" which will help in future movie recommendations of that particular user.

4.5.2 Stimulus/Response Sequences

As soon as the user enters and submits answers to the questions asked on the Question/Answers Page OR prompts on the Prompts Page, they are shown the Movie Recommendation Page

comprising three movie recommendations based on the user's current mood and preferred genre, likes, dislikes, etc.

4.5.3 **Functional Requirements**

Req.1. This page should show movie recommendations to the user if correct movie matches are found, else, display a sorry message saying "No Matching Movies Found!".

Req.2. On clicking the link to the trailer of a particular movie, the trailer of the movie should be played for the user.

5. Other Nonfunctional Requirements

5.1 Performance Requirements

The system must process and return movie recommendations within 2 seconds for 95% of queries under normal load (≤100 concurrent users).
Under heavy load (≥500 concurrent users), the response time should not exceed 5 seconds for 95% of queries. The web app must be able to scale horizontally to support at least 1000 concurrent users with minimal impact on response time. The web app should remain functional on mobile devices with a response time of <5 seconds on 3G connections.

5.2 Safety Requirements

The system must not store or share personally identifiable information (PII) without explicit user consent.
All user inputs, including prompts, should be anonymized and handled securely to prevent unauthorized access.
Data transmission must be encrypted using HTTPS (TLS 1.2 or higher) to prevent interception and unauthorized modifications.
The recommendation engine must not suggest movies that contain explicit, violent, or inappropriate content to underage users.

5.3 Security Requirements

Users must be authenticated before accessing personalized recommendations or saving preferences.
Authentication should support OAuth 2.0 or OpenID Connect for third-party logins (Google, Apple, etc.).
Password-based authentication (if used) must require
Third-party APIs must be integrated using secure authentication methods (API keys, OAuth, JWT tokens).

API access should be rate-limited to prevent abuse (e.g., max 1000 requests per hour per user). No sensitive user data (e.g., full names, email addresses) should be shared with external APIs unless explicitly required and consented by the user.

5.4 Software Quality Attributes

The system must maintain 99.9% uptime, allowing for a maximum of 8.76 hours of downtime per year.
Any planned maintenance should be scheduled during off-peak hours and communicated to users at least 24 hours in advance. The system should have automated monitoring and alerting for outages, with a response time of under 5 minutes for critical failures. The movie recommendation system must provide results that match at least 90% accuracy compared to expected genre, user preferences, or popularity trends. Data integrity checks should be implemented to prevent incorrect or duplicate movie suggestions.

5.5 Business Rules

General and registered user cannot access administrative features
General user cannot save preferences and recommendations
Administrator can generate system performance and analytics reports.
Developer can modify system configurations but require approval for major changes. Any movie metadata or media content displayed must be licensed or sourced from authorized providers.

6. Other Requirements

Here are some additional requirements not covered elsewhere in the SRS:

- **Database Requirements**: The system will utilize a scalable, secure relational database (e.g., PostgreSQL) to manage activity data, ensuring efficient querying and data integrity. It should support real-time updates, automated backups, and data encryption for security.

- **Legal Requirements**: The system must comply with data protection regulations, ensuring secure handling of personal and sensitive data. User data privacy and security protocols will be strictly enforced.

- **Reuse Objectives**: The system architecture should promote modularity, enabling components to be reused in similar educational projects. APIs developed should be well-documented for easy integration.

Appendix A: Glossary

- **API:** Application Programming Interface
- **GDPR:** General Data Protection Regulation
- **WCAG:** Web Content Accessibility Guidelines
- **TBD:** To Be Determined
- **UI:** User Interface
- **DB:** Database
- **HTTPS:** Hypertext Transfer Protocol Secure

Appendix B: Analysis Models

Data Flow Diagram (DFD): Illustrates the flow of data within the system, detailing how information is processed and stored.

Class Diagram: Defines the system's structure by showing system classes, attributes, methods, and the relationships among objects.

Entity-Relationship Diagram (ERD): Represents the data entities, their attributes, and relationships to support database design.

State-Transition Diagram: Describes the system states and transitions triggered by events, useful for understanding system behaviour.

Appendix C: To Be Determined List

1. Selection of the final database management system.
2. Confirmation of APIs to be integrated for automated research tracking.
3. Finalization of supported languages for internationalization.
4. Specification of data encryption standards.
5. Detailed accessibility compliance measures.