Software Requirements Specification

for

Capstone Project

Prepared by Muhammad Syahmi bin Mohd Radzuan

.Net Full Stack Crimson Logic Training

30/1/2023

Table of Contents

Contents

1 Introduction

1.1. Purpose

The purpose of this document is to specify the requirements for the Movie Website project that allows users to register, login, and interact with a database to perform CRUD functions for movies and users if they are an admin.

1.2. Scope

The scope of this web application includes the following functionalities:

- User registration and login
- Admin CRUD operations for movies
- Admin addition of new users
- User movie ratings and recommendation system

1.3. Definitions, Acronyms, and Abbreviations

- API = Application Programming Interface
- CRUD = Create, Read, Update and Delete

1.4. References

N/A

2. System Overview

2.1. System Architecture

The web application will be developed using React for the front-end and ASP.Net Core and Entity Framework for the back-end in the form of an API.

2.2. User Types

There are two types of users in the system:

- Admin
- Regular User

3. User Requirements

3.1. User Registration

- The user must provide their name, email, and password to register.
- The user must confirm their email address to complete the registration process.
- A registered user can log in to the system using their email and password.

3.2. Login

- The user must enter their email and password to log in.
- A user can log in as either an Admin or a Regular User.
- The user will be redirected to the appropriate dashboard based on their user type.

3.3. Admin Dashboard

The Admin Dashboard allows Admins to perform CRUD operations on the movie database:

- Admins can add, edit, and delete movies in the database.

Admins can also add new users through the Admin Dashboard.

3.4. Regular User Dashboard

- The Regular User Dashboard allows Regular Users to view a list of available movies in the database and give them ratings.
- Regular Users can also view the ratings they have given to movies.
- When a Regular User clicks on a movie, the website will give recommendations to the user of other similar movies based on their ratings using a Cosine Similarity matrix.

4. Technical Requirements

4.1. Front-end

- The front-end of the web application will be developed using React.
- The front-end will use React Router to manage navigation between different pages.

4.2. Back-end

- The back-end of the web application will be developed using ASP.Net Core and Entity Framework.
- The back-end will provide an API for the front-end to access and manipulate the movie database.
- The back-end will implement user authentication and authorization.
- The back-end will also implement the recommendation system based on the Cosine Similarity matrix.

5. Functional Requirements

5.1. User Registration

- The system must provide a form for the user to enter their name, email, and password.
- The system must validate the email address provided by the user and ensure it is unique.
- The system must send a confirmation email to the user to confirm their email address.
- The system must store the user's name, email, and encrypted password in the database.

5.2. Login

- The system must provide a form for the user to enter their email and password.
- The system must validate the email and password provided by the user against the values stored in the database.
- The system must redirect the user to the appropriate dashboard based on their user type.

5.3. Admin Dashboard

- The system must provide a user interface for admins to perform CRUD operations on the movie database.
- The system must allow Admins to add new movies to the database by providing a form to enter the Title, posterId, genre, and description of the movie.
- The system must allow Admins to edit existing movies in the database by providing a form to modify the attributes of the movie.
- The system must allow Admins to delete movies from the database.
- The system must allow Admins to add new users to the system by providing a form to enter the name, email, and password of the user.

5.4. Regular User Dashboard

- The system must provide a user interface for Regular Users to view a list of available movies in the database.
- The system must store the ratings given by Regular Users in the database.
- The system must provide recommendations to Regular Users of other similar movies based on their ratings using a Cosine Similarity matrix.

6. Non-functional Requirements

6.1. Performance

- The system must provide fast response times for user actions such as logging in, movie search, and movie ratings.
- The system must be able to handle a large number of users and movie entries without affecting performance.

6.2. Security

- The system must implement secure user authentication and authorization.
- The system must store sensitive user data such as passwords in encrypted form in the database.
- The system must prevent unauthorized access to the movie database and user data.

6.3. Usability

- The system must be user-friendly and provide a simple and intuitive interface for users to interact with.
- The system must provide clear and concise error messages in case of invalid user inputs.
- The system must be accessible and usable by users with disabilities.

6.4. Scalability

- The system must be designed in such a way that it can be easily scaled to accommodate future growth.

7. Conclusion

This SRS document specifies the requirements for a web application that allows users to register, login, and interact with a movie database. The front-end will be developed using React and the back-end using ASP.Net Core and Entity Framework as an API. The functional requirements include user registration, login, Admin Dashboard, and Regular User Dashboard. The non-functional requirements include performance, security, usability, and scalability.