

# **1. Introduction**

## **1.1 Purpose**

The purpose of this document is to define the requirements the provide users with a seamless experience for browsing, searching, and watching movie trailers. This document will serve as a guide for the development team to understand the functionalities and constraints of the system.

## **1.2 Product Scope**

- Users can create accounts, log in, and manage their profiles.
- Users can browse movies by genre, release date, popularity, and search for specific movies by title, actor, or director.
- Detailed information about each movie, including synopsis, cast, crew, trailers, and user reviews.
- Users can write reviews and rate movies.
- The application will suggest movies based on user preferences and viewing history.
- Users can create and manage a list of movies they want to watch.
- An admin interface to manage movies, user accounts, and reviews.

## **1.3 References**

[1] TMDB API for movie data: <https://developer.themoviedb.org/>

## **2. Overall Description**

### **2.1 Product Perspective**

The Movie Web Application is a web-based platform that allows users to discover, explore, and manage movies effortlessly. It integrates with external APIs (such as TMDb) to provide real-time movie data, including trending movies, search functionality, trailers, ratings, and descriptions. The platform offers user authentication, a favorites list, and a responsive design for seamless access across devices. It ensures an engaging and intuitive experience with features like personalized movie collections.

### **2.2 Product Functions**

The main functionalities include:

- Browse trending and categorized movies.
- Search for movies by name.
- View movie details (description, trailer, rating).
- Save movies to a favorites list.
- Secure authentication and user management.

### **2.3 User Classes and Characteristics**

- Guest Users: Can browse movies but cannot save favorites
- Registered Users
  - Customer can log in to the system.
  - User can sign up – User should be able to create an account using email, Google, or Facebook.
  - User wants to select the payment package.
  - Password Reset – The user can password recover.
  - Profile Management -The user can manage profile details.
  - User can video playback – Play, Pause, Resume and Control Volume.
  - User can search the movies.
  - User can rate the movies.
  - User can play the free trial.
  - User can save the movies.
  - User can read the movie details.
  - User can new releases and movie details.

- Admin Users
  - Admin can add, edit, and delete the movies.
  - They can categorize the movies.
  - They can add the movie details.
  - Admin panel can watch user's history and recommendation.
  - They can watch the user ratings and reviews, likes and dislikes.
  - They can do content moderation.
  - User profile /Accounts management.
  - User data protection.

## **2.4 Operating Environment**

### **Hardware Requirements**

Windows 11

Processor- Intel Dual core or above

Processor Speed – 1.0GHz or above

RAM – 1GB RAM or above

Hard Disk Space – 40GB above

### **Software Requirements**

Frontend - React

Backend - Node.js

database design -MongoDB

API: TMDB API for movie data

### **Internet Connection:**

Requires a stable internet connection for browsing and interaction.

### **Supported Browsers:**

Works seamlessly on popular web browsers like Chrome, Firefox, Safari, and Edge.

## **2.5 Design and Implementation Constraints**

- User authentication must follow secure password encryption
- Scalability and Performance
- User Experience (UX) Constraints
- Integration Constraints
- Regulatory Compliance

## 2.6 Assumptions and Dependencies

### Internet Connectivity

- A stable internet connection is required for fetching movie data, trailers, and authentication services.

### API Availability

- **TMDB API** must be operational and accessible to fetch movie details, trending lists, and search results.

### Browser Compatibility

- The application assumes users are accessing it through modern web browsers (Chrome, Firefox, Edge, Safari) that support modern UI features.

### Device Compatibility

- The platform is designed for desktop devices with responsive design support.

### User Account Management

- Users are responsible for maintaining their login credentials and handling password security and authentication flow.

### Third-Party Integrations

- Any future third-party services (e.g., payment gateways for premium features) must be compatible with the existing tech stack.

## 3. Specific Requirements

### 3.1 Functional Requirements

**FR1:** Users can create an account on the platform by providing required information such as name, email address, and password.

**FR2:** Registered users can log in to their accounts using their email address and password.

**FR3:** Implement secure authentication mechanisms to protect user accounts from unauthorized access. **FR4:** Each user has a profile page to view and manage their personal information, profile picture, and account settings.

**FR5:** Users can update their profile information, and change passwords.

**FR6:** The system shall allow users to browse movies categorized by trending, popular, and genres.

**FR7:** The system shall provide a search bar to find movies by name.

**FR8:** The system shall display movie details, including title, description, rating, and trailer.

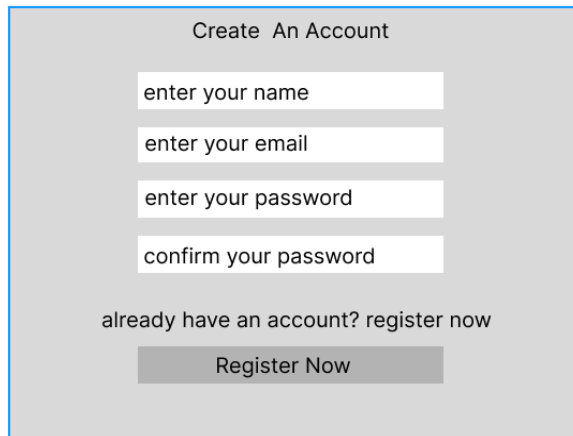
**FR9:** Registered users shall be able to add/remove movies from favorites.

**FR9:** The system shall allow users to reset their password via email.

## 3.2 External Interface Requirements

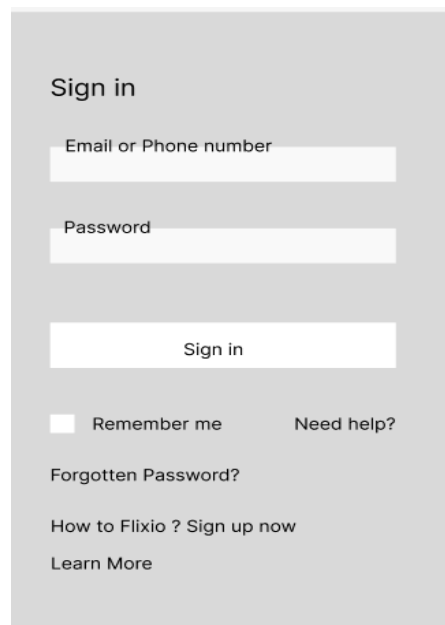
### 3.2.1 User Interfaces

- Register Form



A UI mockup of a registration form titled "Create An Account". It features four input fields stacked vertically: "enter your name", "enter your email", "enter your password", and "confirm your password". Below these fields is a link that says "already have an account? register now". At the bottom is a grey button labeled "Register Now".

- Login Form



A UI mockup of a login form titled "Sign in". It features two input fields: "Email or Phone number" and "Password". Below these is a white button labeled "Sign in". Under the button is a checkbox labeled "Remember me" and a link labeled "Need help?". At the bottom are two links: "Forgotten Password?" and "How to Flixio ? Sign up now", followed by a link labeled "Learn More".

### 3.2.2 Hardware Interfaces

- The application will interact with various user devices, including desktops, laptops.

### 3.2.3 Software Interfaces

- **Payment Gateway API:** Integration with payment processing services (e.g., PayPal) for subscription management.

## 3.3 Performance Requirements

- The website should load quickly to prevent user frustration and bounce rates.
- The website is fully responsive and performs well on various devices. The platform should be able to handle increased traffic and user activity without significant degradation in performance.
- Search functionality should return results quickly, even with a large number of listings.
- Ensure that reservation requests are processed promptly, availability is updated in real-time, and confirmation emails are sent out efficiently.

## 3.4 Security Requirements

**Authentication Methods -** User authentication via email/password and OAuth for social logins (e.g., Google, Facebook).

## 3.5 Software Quality Attributes

### Reliability:

- 4 The website should be dependable and operate consistently without unexpected failures.
- 5 Ensure high availability to prevent downtime and disruptions in service.
- 6 Implement robust error handling and recovery mechanisms to gracefully handle exceptions and errors.

### **Performance:**

- 7 Unit tests, integration tests, and automated tests shall be implemented for core functionalities.
- 8 The system shall follow maintainability metrics like cyclomatic complexity, code duplication, and test coverage.
- 9 The codebase should follow a **modular structure** for easy updates.

### **Scalability:**

- 10 The platform should be able to handle increasing loads and accommodate growing numbers of users and listings.

### **Usability:**

- 11 The UI should be **easy to navigate** for all users, including first-time visitors.
- 12 Ensure clear and concise user interfaces with logical workflows for searching, listing, and managing properties.
- 13 Provide helpful guidance and tooltips to assist users in completing tasks efficiently.