**Movies Review & Information System**

**Muhammad Nouman Zafar**

**A00314863**



B.Eng. Software Engineering

Technological University of the Shannon: Midlands Midwest

# Introduction

My comprehensive movie review and information system is here to help film enthusiasts dive deep into the world of cinema. The purpose of this platform is to provide users with in-depth knowledge about their favourite films, as well as an plenty of related content and functions.

There is an extensive database of movies waiting to be explored when user enter my website. A wealth of information can be accessed with just a few clicks, including the title, genre, release date, description, overall rating, and user ratings. User can find the information they're looking for quickly using the intuitive user interface.

A dedicated page that serves as a treasure trove of movie-related content is available when user click on a specific movie. A comprehensive list of individuals involved in the making of the film can be found here. Each person is linked to a page where user can explore all the movies associated with them.

This platform goes beyond mere information display. User can get a glimpse of the movie by watching related clips, such as trailers. User can also post their thoughts and opinions in the comments section, creating a lively community of movie fans.

A Toolkit page offers a range of advanced features for those who want to take their experience to the next level. User can alter the data and contribute to the ever-growing database by adding new movies and people or linking people to specific movies.

This movie review and information system is ultimate destination for all things cinema, no matter what kind of movie user like.

The technical project documentation serves as a comprehensive guide to understanding the architecture, functionality, and implementation details of my movie review and information system application. From the initial conceptualization to the deployment phase, each aspect of development process is documented herein, offering insights into the design choices, challenges encountered, and solutions adapted during the phases like brain storming phase, planning, implementation.

Key components of the Movie Review and Information Application are divided into three different sections and each section serves a very important role to complete the application, and those three components are

* Front End
* Back End
* Storage

# Research

The preliminary research for the Movie Review and Information Application focused on identifying existing movie review platforms, assessing user requirements, and assessing potential technologies and frameworks. Like TMBD, IMDB etc.

My initial plan was to integrate with existing movie databases to retrieve comprehensive movie information. The platforms provided me with APIs to access movie titles, genres, release dates, descriptions, and ratings. However, I encountered difficulties with the usage guidelines and restrictions associated with these APIs, prompting me to investigate other options. At the same time, I wanted to build my own API using backend services, if I had used the external APIs then there would be no need of backend or Database, so My final decision was to build everything myself because this way every component of the project is controlled by me.

After evaluating several options, including external APIs and open data sources, I decided to build my own database of movie information. Flexibility in data management and the desire to avoid potential restrictions imposed by third-party APIs were factors that influenced this decision. Even though I am built my own database but I ran that database on AWS EC2 console instance with the help of Ubuntu virtual machine. For data storage, I explored many options like relational and NoSQL databases. In the end I decided to use MySQL database which is easier to integrate with the back and front-end technologies.

I considered several options for both the front-end and back-end components of the application. The front end was evaluated for their ability to provide a dynamic and responsive user interface. It was ultimately chosen for its popularity, extensive community support, and robust ecosystem of libraries and tools. As front-end technology first, I wanted to use Angular but after the extensive research I realised that it is way harder to handle and there are so many things needs to be sorted before using Angular so eventually I decided to use React with the help of Node.js for my front-end user interface.

I evaluated different programming languages and frameworks for building APIs and managing data storage on the back end. While alternatives like Java, Python with Django and Node.js with Express were considered, I ultimately settled on Spring-boot due to its robust features for both API development and data storage management with the help of Speedment. Overall, its combination of flexibility, performance, and community support made it the optimal choice for my project’s backend development needs.

## Technologies

Movie Review and Information application is divided into three different sections

* Front End
* Back End
* Storage

### Front-End Technologies:

The technologies used to build the front-end user interface are:

* React
* React Router DOM
* Axios
* React Player
* HTML & CSS
* JavaScript
* Node.js
* Local Storage
* VS Code
* npm

**React:**

* React is a JavaScript library for building user interface. It allows developers to create reusable UI components and efficiently manage the state of the application.
* React’s component-based architecture encourages a modular approach to building applications, making it easier to maintain and scale projects.
* JSX is used to write React components, allowing developers to write HTML-like syntax within JavaScript code.

**React Router DOM:**

* React Router DOM is library that provide routing capabilities for React applications.
* It allows developers to define different routes within the application based on URL, enabling navigation between different components or pages of the application.
* Features like dynamic route matching and nested routing make it powerful for building complex single-page applications.

**Axios:**

* Axios is a popular JavaScript library for making HTTP requests from he browser or from the backend of the whole application.
* It provides a simple and intuitive API for performing asynchronous operations like fetching data from server or posting data to an API endpoint.
* Axios supports features like request and response interception, automatic transformation of JSON data, and handling errors.

**React Player:**

* React Player is a React component specially designed for playing media files such as videos and audio.
* It abstracts away the complexities of implementing media playback in a web application, providing a simple interface for embedding media content.
* React Player supports various media sourced, including URLs, files, and streaming services like YouTube.

**HTML & CSS:**

* HTML (Hypertext Mark-up Language) is used for creating web pages and applications. It provides the structure and content on webpage.
* CSS (Cascading Style Sheets) is used to style HTML elements, controlling their appearance, layout, and presentation on the screen.

**JavaScript:**

* JavaScript is a programming language of the web, used for adding interactivity and dynamic behaviour to the web pages and applications.
* JavaScript features like arrow functions, restructuring assignment, template literals, and async/await are commonly used in React development for writing clean and concise code.

**Node.js:**

* Node.js is a JavaScript runtime environment that allows developers to run JavaScript code outside of a web browser.
* In this application Node.js is used in order to use its component npm

**Local Storage:**

* Local Storage is a web browser feature that allows web applications data to store locally on the user’s device.
* It provides simple key-value storage mechanism and can store large amounts of data composed to cookies.
* In my Application Local Storage is used to store the information of a logged in user like email, likely for authentication purpose.

**Visual Studio Code:**

* VS Code is a free and open source code editor developed by Microsoft.
* It provides a rich set of features for coding, debugging, and version control which makes I a popular choice among the developers.
* VS Code offers built-in support for JavaScript, JSX, HTML, CSS and various other programming languages and file formats commonly used in web application development.

**npm (Node package manager):**

* npm is the default package manager for Node.js, the JavaScript runtime environment.
* It allows developers to install, share and manage dependencies for their projects.
* With npm, developers can easily integrate third-party libraries, frameworks, and tools into their applications.
* npm also provides a command-line interface for package installation, version management, and script execution.

### Back End Technologies:

The technologies used to build the back-end are:

* Spring Boot
* Spring Web
* Spring Framework Annotations
* Spring Response Entity
* CORS
* Java Streams
* Java Lambda Expression
* Java Generics
* Java Collections Framework
* JPA
* Dependency Injection
* Speedment
* Maven
* Eclipse

**Spring Boot:** Spring boot is a popular Java framework for building web applications. It has tools for building RESTful APIs, handling HTTP requests, and handling dependencies.

**Spring Web:** Spring Web is a component of the Spring framework and offers tools for building web applications, including RESTful services.

**Spring Framework Annotations:** Annotations like @RestController, @RequestMapping, @GetMapping, @PostMapping, and @RequestBody are used to define REST endpoints and handle HTTP requests.

**Spring Response Entity:** This class is used to represent the entire HTTP response. It allows you to control the HTTP response status, headers, and body.

**CORS:** Spring Cross-Origin Resource Sharing @CrossOrigin annotation allows cross-origin requests from any origin. This is useful when the client-side code is served from a different domain.

**Java Streams:** Java Streams are used for processing collections of data in a functional style. They allow you to perform aggregate operations on collections, such as filtering, mapping, and reducing.

**Java Lambda Expression:** Lambda expressions are used in stream operations to concisely define inline functions.

**Java Generics:** Generics are used throughout the code to provide type safety and reusability.

**Java Collections Framework:** The List and Map interfaces from the Java Collections Framework are used to store and manipulate collections of data.

**Java Persistence API:** It is used for data persistence as the back-end application interacts with database e.g. (MovieManager, MoviePeopleManager etc.).

**Dependency Injection:** Spring boot’s dependency injection is used in the constructor of the Controller class to inject instance of Manager classes.

**Speedment:** It is a java object relational mapping tool that allows developers to create java applications with high-performance database access layer. Speedment generates java code based on an existing database schema, providing type-safe APIs for interacting with the database.

**Maven:** Maven is a build automation tool primarily used for Java projects. It is primarily used for building Java projects. Managing a project's dependencies, compiling source code, and packaging the application into distributable forms like JAR files are all streamlined by it. Maven takes a declarative approach, where developers specify project configurations, dependencies, and build steps in an XML file called pom.xml (Project Object Model) The directory structure in Maven makes it easier for developers to work together and share code. Overall, Maven helps streamline the development process by automating repetitive tasks and enforcing best practices in Java project management. It also helps enforce best practices in Java project management.

**Eclipse:** Eclipse is an integrated development environment (IDE) primarily used for Java development, although it supports various programming languages through plugins. It provides developers with a comprehensive set of tools for writing, debugging, and deploying software applications.

### Storage Technologies:

Technologies used to store the data are:

* MySQL Database
* AWS EC2
* Security Groups
* PuTTY

**MySQL Database:** MySQL is an open-source relational database management system. It's the core technology for storing and managing structured data.

**AWS EC2:** Elastic Compute Cloud provides scalable computing capacity in the cloud. You launch virtual servers, known as instances, which can run various operating systems and applications, including MySQL.

**Security Groups:** Security groups act as virtual firewalls for your EC2 instances to control inbound and outbound traffic. You can configure security groups to allow only necessary traffic to reach your MySQL server.

**PuTTY:** PuTTY is a popular open-source SSH client. It allows you to securely connect to your EC2 instance over SSH (Secure Shell) protocol. With PuTTY, you can remotely access the command-line interface of your EC2 instance, where you can execute commands to manage your MySQL server, such as querying databases, configuring settings, or running maintenance tasks.

## Application Domain

The movie review and information system application domain encompass various aspects related to the world of cinema, catering to the needs and interests of movie enthusiasts, critics, and casual viewers alike.

The application domain is broken down into different sections like:

* Movie Information
* User Interaction
* Movie Search
* Cinematic content
* Industry Insights
* Recommendations
* Data Management
* Accessibility
* Community Building

**Movie Information:** The core of the domain revolves around providing comprehensive information about movies. This includes information such as title, genre, release date, synopsis, cast and crew, ratings, trailers, and related media content.

**User Interaction:** User interaction is facilitated by features like comments, evaluations, and evaluations in this application. Users can share their thoughts, gain insights, and discuss their favourite films. This creates a lively community of movie enthusiasts within the platform.

**Movie Search:** The application domain incorporates a robust movie search feature, allowing users to find specific movies easily. Users can search by title or keyword. This feature enhances usability and efficiency by enabling users to discover relevant movies quickly.

**Cinematic content:** Feature films, documentaries, short films, and series are covered by the domain. It caters to diverse preferences, covering various genres, languages, and cultural backgrounds.

**Industry Insights:** Information about filmmakers, actors, producers and other professionals involved in the movie-making process is provided by this application.

**Recommendations:** This application domain provides the recommendation based on users selected movie genre. It will display all the movies with the same genre.

**Data Management:** This application manages vast amounts of data related to movies, users, comments, ratings, and interactions. Database management systems are used to organize, retrieve and update information efficiently.

**Accessibility:** The domain emphasizes accessibility, ensuring that users can easily navigate the platform, search for specific movies, and access relevant content. User-friendly interfaces, intuitive navigation menus, and robust search functionalities enhance usability.

**Community Building:** A sense of community is built by providing features for social interaction, collaboration, and knowledge sharing through the usage of feature of posting comments for any selected movie.

By addressing these key aspects of movie watching experience, this application creates a comprehensive and engaging environment for users to explore, interact and appreciate the art of cinema.

# Requirements

The purpose of the Movie Review and Information System is to provide users with a comprehensive platform to explore and interact with a vast database of movies, covering various genres, release dates, and ratings. The application aims to help users discover, analyse, and discuss their favourite films, while also providing insights into the movie-making process and fostering a sense of community among movie enthusiasts.

There are two type of requirements:

* Functional Requirements
  + Movie Exploration Functionality
  + Display Movie Information
  + User Interaction Features
  + Advance Functionality
* Non-Functional Requirements
  + Performance
  + Usability
  + Security

## Functional Requirements

**Movie Exploration Functionality:**

* **Search by Title:** User should be able to search by movie title.
* **Browse by Category:** user should be able to browse movies by category such as popular, top-rated, recently released.

**Display Movie Information:**

* **List View:** Display search results or browse categories in a list format, showing relevant details such as title, genre, release date, and thumbnail image.
* **Detailed View:** Allow users to access detailed information about a selected movie, including synopsis, cast and crew, ratings, trailers, and related media content.

**User Interaction Feature:**

* **Comments Section:** Enable users to post comments and discuss their thoughts and opinions on specific movies.
* **Rating:** Allow users to rate movies and write reviews to share their feedback with the community.

**Advanced Functionality:**

* **Toolkit Page:** Offer advanced features for users who want to contribute to the database, such as adding new movies, linking individuals to specific movies, and updating movie information.
* **Registration:** Allow new users to register themselves, when they register their information stores in database and whenever they try to log in their provided details will match the details stored in database.

## Non-Functional Requirements

**Performance:**

* **Response Time:** Ensure that movie information is retrieved and displayed promptly, with minimal latency.
* **Scalability:** Design the system to handle a large number of concurrent users and a growing database of movies without compromising performance.

**Usability:**

* **Intuitive Interface:** Design a user-friendly interface with intuitive navigation and clear organization of movie information.
* **Accessibility:** Ensure that the application is accessible to all users.

**Security:**

* **Data Privacy:** Implement measures to protect user data, including encryption of sensitive information and secure authentication mechanisms.
* **Secure Communication:** Use HTTPS protocol to encrypt data transmitted between the client and server, ensuring privacy and integrity.

## Technology Stack

**Frontend:** Utilize React.js for building the frontend user interface, providing a responsive and interactive user experience.

**Backend:** Implement Spring Boot for the backend application, handling API requests, data retrieval, and database management.

**Database:** Utilize MySQL for storing and managing movie information, user data, and interaction records.

## Constraints

**Data Storage:** Ensure that sufficient storage capacity is available to store the growing database of movies and user interactions.

**AWS EC2:** Utilize AWS EC2 for hosting the application and database, ensuring scalability and reliability of the infrastructure.

## Assumptions

**Internet Connectivity:** It is assumed that users will have a stable internet connection to access the application and retrieve movie data.

**Modern Devices:** The application is designed to be compatible with modern web browsers and devices, including desktops, laptops, tablets, and smartphones.

## Dependencies

**Third-Party Libraries:** Identify and manage dependencies on third-party libraries and frameworks used in the development of the application, including those managed via npm for frontend and Maven for backend.

**Custom APIs:** Acknowledge the dependency on custom APIs developed for fetching movie data, managing user interactions, and integrating additional features.

**AWS Services:** Recognize the dependency on AWS services such as EC2 for hosting the application and database management.

# Architecture