## Project-Based Evaluation

Project Report

Semester-V (Batch-2023)

**Movie Booking Website**

A red and white sign

Description automatically generated with low confidence

**Supervised By: Submitted By:**

Mr. Rahul Aman, 2310990275(G4)

Aryan Singla, 2310990304(G4)

Deepanshu Jindal, 2310990321(G4)

Gopesh Goyal, 2310990337(G4)

**Department of Computer Science and Engineering**

**Chitkara University Institute of Engineering & Technology,**

**Chitkara University, Punjab**

**ABSTRACT**

In today’s digital age, convenience and efficiency are essential in everyday activities, including entertainment. Traditional methods of purchasing movie tickets often involve long queues, time-consuming processes, and limited access to real-time information. To address these challenges, we have developed a **Movie Booking Website** using the MERN stack, which offers a seamless and user-friendly platform for moviegoers. This web application enables users to browse currently showing movies, view detailed information such as timings, posters, and descriptions, and securely book tickets online.

The system is designed with a focus on **usability, accessibility, and efficiency**. Users can select preferred show timings and seats using an interactive seat selection interface, ensuring a personalized experience. Additionally, a booking history feature allows users to track their past and upcoming reservations conveniently. On the administrative side, the platform provides tools for managing movies, including adding, editing, or deleting entries, as well as uploading posters and providing detailed descriptions.

The backend of the application is powered by **Node.js and Express.js**, which handle server-side operations and API management, while **MongoDB** serves as the database for storing movie details, user information, and booking records. The frontend is built using **React.js**, offering a responsive and dynamic interface that adapts to multiple devices.

This project demonstrates the potential of modern web technologies to enhance user experience and streamline traditional processes. By reducing wait times, ensuring real-time availability, and providing an intuitive interface, the Movie Booking Website significantly improves the process of ticket booking. Furthermore, the system is scalable and can be expanded with features such as online payments, personalized recommendations, multi-city support, and mobile applications in the future. Overall, this project illustrates how technology can transform routine tasks into efficient, enjoyable experiences.

### TABLE OF CONTENTS

1. Introduction
   1. Background
   2. Objectives
   3. Significance
   4. Scope of Project
2. Problem Definition and Requirements
   1. Problem Statement
   2. Software Requirements
   3. Hardware Requirements
3. Proposed Design
   1. System Architecture Overview
   2. Methodology
   3. Class and Functional Responsibility
   4. File Structure Description
   5. Data Flow and Processing Logic
4. Results
   1. Screenshots
5. References

**1. INTRODUCTION**

**1.1 Background**

With the rapid growth of the entertainment industry and the increasing number of moviegoers, the demand for efficient and convenient movie ticket booking systems has risen significantly. Traditionally, booking a movie ticket required physically visiting a theater, standing in long queues, and relying on limited information about show timings and seat availability. This process was often time-consuming, inconvenient, and prone to errors, such as double bookings or unavailability of preferred seats. Additionally, manually managing movie listings, schedules, and bookings was cumbersome for theater administrators, leading to inefficiencies in overall operations.

The emergence of digital technology and the internet has revolutionized how people access services, including entertainment. Online platforms have become a preferred solution for simplifying daily tasks, and movie ticket booking is no exception. By leveraging web technologies, it is now possible to provide a seamless, real-time, and user-friendly platform for both customers and administrators. A well-designed online movie booking system not only reduces waiting time and improves customer satisfaction but also enhances operational efficiency for theater management.

The **Movie Booking Website** project addresses these challenges by providing a comprehensive platform built using the **MERN stack (MongoDB, Express.js, React.js, Node.js)**. The system allows users to explore currently showing movies, select preferred show timings, choose seats through an interactive seat selection interface, and securely book tickets online. Users can also maintain a history of their bookings for future reference. From the administrative perspective, the platform enables easy management of movie listings, including adding, editing, or deleting movies, uploading posters, and updating descriptions.

By integrating frontend and backend technologies, the system ensures a smooth flow of information, real-time updates, and secure handling of user data. This project demonstrates how modern web technologies can enhance user experience, improve accessibility, and streamline operations in the entertainment industry. Ultimately, the Movie Booking Website provides a **convenient, efficient, and scalable solution** for movie ticket booking in the digital era.

**1.2 Objectives**

The main objectives of the Movie Booking Website project are as follows:

1. **Simplify Ticket Booking:** To provide users with an easy and convenient way to book movie tickets online, eliminating the need to stand in long queues at theaters.
2. **Real-Time Seat Availability:** To allow users to select seats based on real-time availability, preventing double bookings and ensuring a smooth booking experience.
3. **Manage Movie Listings:** To enable administrators to efficiently add, edit, or delete movies, including uploading posters and updating movie descriptions.
4. **Maintain Booking History:** To allow users to view their past and upcoming bookings for easy reference and better planning.
5. **Enhance User Experience:** To design a responsive and user-friendly interface that works seamlessly across multiple devices, such as desktops, tablets, and mobile phones.
6. **Secure Data Handling:** To ensure secure management of user information, bookings, and administrative operations.
7. **Scalable System:** To develop a platform that can be expanded in the future to include features like online payments, personalized recommendations, multi-city support, and mobile applications.

The project aims to combine modern web technologies with practical user requirements to create an efficient, reliable, and enjoyable movie ticket booking system.

**1.3 Significance**

The Movie Booking Website holds significant value in today’s fast-paced digital world by offering a convenient, time-saving, and user-friendly solution for moviegoers. It streamlines the ticket booking process, reduces waiting times, and provides real-time seat availability, enhancing the overall movie-going experience. Additionally, it assists administrators in efficiently managing movie listings, schedules, and bookings, ensuring smooth theater operations. By combining modern web technologies with practical usability, the system demonstrates how digital solutions can improve both customer satisfaction and operational efficiency.

**1. User Convenience**

* Book tickets anytime, anywhere.
* Choose preferred show timings and seats.
* Avoid long queues at theaters.

**2. Real-Time Operations**

* View real-time seat availability.
* Reduce errors and double bookings.
* Track booking history for planning.

**3. Administrative Efficiency**

* Add, edit, or delete movies easily.
* Upload posters and update movie details.
* Manage theater schedules efficiently.

**4. Enhanced User Experience**

* Intuitive and responsive interface.
* Accessible on desktops, tablets, and mobile devices.
* Streamlined navigation and booking process.

**5. Future Readiness**

* Supports potential online payment integration.
* Can include recommendations and promotions.
* Scalable to multiple cities and theaters.

**1.4. Scope of the Project**

The Movie Booking Website aims to provide a scalable and user-friendly platform that streamlines the process of booking movie tickets online. It allows users to browse current movies, select preferred show timings, choose seats, and maintain a history of bookings. The system also empowers administrators to manage movie listings, posters, and descriptions efficiently. With the potential for future enhancements such as online payments, personalized recommendations, and multi-city support, this project has a wide scope to improve the movie-going experience and adapt to evolving user needs.

**1. User Convenience**

* Online ticket booking anytime and anywhere.
* Seat selection and showtime preferences.
* Access to past booking history.

**2. Administrative Control**

* Add, edit, or delete movie listings.
* Upload posters and provide descriptions.
* Manage theater schedules and data efficiently.

**3. Technology & Scalability**

* Built on MERN stack for modern web capabilities.
* Easily extendable to include mobile apps or additional features.
* Supports integration with payment gateways and recommendation systems.

**4. Future Enhancements**

* Personalized recommendations based on user behavior.
* Multi-city and multi-theater support.
* Offers, discounts, and loyalty program integration.

**2. PROBLEM DEFINITION AND REQUIREMENTS**

**2.1 Problem Statement**

In traditional movie ticket booking systems, users face multiple challenges that reduce convenience and efficiency. Long queues at theaters, limited access to real-time seat availability, and dependence on manual booking methods often lead to frustration and wasted time. For example, a popular show may be fully booked before users can reach the counter, or users may end up choosing undesired seats due to lack of proper seat selection options. Additionally, tracking past bookings or planning for future shows becomes cumbersome without a centralized system.

On the administrative side, manually managing movie listings, schedules, and booking records is time-consuming and prone to errors. The lack of automation makes it difficult to provide accurate and up-to-date information to customers, affecting overall theater management and customer satisfaction.

The **Movie Booking Website** addresses these problems by providing a secure, efficient, and user-friendly online platform. Users can browse movies, select show timings, choose seats using an interactive interface, and maintain a booking history. Administrators can easily add, edit, or delete movies, upload posters, and update descriptions. By bridging the gap between users and theater operations, this system enhances the movie-going experience and streamlines the booking process in a digital, time-conscious era.

**2.2 Software Requirements**

The Movie Booking Website is built using modern web technologies and requires the following software components for development and deployment:

**1. Frontend:**

* **React.js –** To create a dynamic and responsive user interface.
* **Bootstrap / CSS –** For styling and designing a modern and user-friendly interface.
* **Node Package Manager (NPM) –** To manage frontend dependencies and packages.

**2. Backend:**

* **Node.js –** For server-side scripting and handling HTTP requests.
* **Express.js –** For building RESTful APIs and managing server routing efficiently.
* **Nodemon –** To automatically restart the server during development.

**3. Database:**

* **MongoDB –** To store movie details, user information, and booking records.
* **MongoDB Compass –** For managing and visualizing the database.

**4. Tools and Others:**

* **Visual Studio Code (VS Code) –** Recommended IDE for development.
* **Postman –** To test API endpoints during development.
* **Git/GitHub –** For version control and collaboration.
* **Web Browser (Chrome, Firefox, Edge) –** To run and test the web application.

These software requirements ensure smooth development, testing, and deployment of the Movie Booking Website, enabling efficient user interactions and administrative operations.

* 1. **Hardware Requirements**

The Movie Booking Website does not require high-end hardware and can run efficiently on standard personal computers and servers. The following specifications are recommended for smooth development, testing, and deployment:

**1. Client-Side (User Requirements):**

* **Processor:** Intel Core i3 or equivalent
* **RAM:** 4 GB or higher
* **Storage:** At least 20 GB free disk space
* **Display:** 1024×768 resolution or higher
* **Internet Connection:** Broadband or stable internet for online access

**2. Server-Side (Development/Hosting Requirements):**

* **Processor:** Intel Core i5 or equivalent
* **RAM:** 8 GB or higher
* **Storage:** 100 GB or higher, preferably SSD for faster performance
* **Operating System:** Windows 10/11, Linux (Ubuntu, CentOS), or macOS
* **Database Storage:** MongoDB server installed with sufficient space for movie and booking records
* **Network:** Stable internet connection for API access, deployment, and testing

These hardware requirements ensure that the Movie Booking Website can perform efficiently for both users and administrators, providing a smooth, responsive, and reliable experience.

**3. PROPOSED DESIGN**

**3.1 System Architecture Overview**

The Movie Booking Website is designed using the **MERN stack (MongoDB, Express.js, React.js, Node.js)**, which allows seamless interaction between the frontend, backend, and database. The system follows a **three-tier architecture** comprising the Presentation Layer, Application Layer, and Database Layer.

**1. Presentation Layer (Frontend):**

* Built with **React.js**, this layer handles the user interface and user interactions.
* Users can browse movies, view details, select show timings and seats, and book tickets.
* Provides responsive design to ensure accessibility on desktops, tablets, and mobile devices.

**2. Application Layer (Backend):**

* **Node.js** and **Express.js** form the backend, which processes client requests, manages API endpoints, and communicates with the database.
* Handles authentication, seat selection, booking processing, and administrative functions like adding or updating movies.
* Ensures secure data handling and real-time response to user actions.

**3. Database Layer:**

* **MongoDB** stores all data, including user information, movie details, and booking records.
* Enables fast retrieval and updating of records while maintaining data integrity.

**Data Flow:**

1. Users send requests through the frontend.
2. Backend APIs process the requests, perform validations, and interact with the database.
3. Database sends the required data back to the backend, which then updates the frontend interface.

This architecture ensures **scalability, maintainability, and efficient performance**, providing a seamless and interactive movie booking experience.

### 3.2 Methodology

The development of the Movie Booking Website follows a **structured and systematic methodology** to ensure efficiency, usability, and reliability. The project employs the **MERN stack**, combining MongoDB, Express.js, React.js, and Node.js to create a full-stack web application.

**1. Requirement Analysis:**

* Understanding user needs, including ticket booking, seat selection, and booking history.
* Identifying administrative requirements for managing movies, posters, and show timings.
* Studying existing systems to recognize limitations and areas for improvement.

**2. System Design:**

* Designing the **frontend** using React.js for a responsive and user-friendly interface.
* Structuring the **backend** with Node.js and Express.js to handle requests, API endpoints, and server-side logic.
* Creating the **database schema** in MongoDB to store users, movies, and booking data efficiently.

**3. Implementation:**

* Developing frontend components for browsing movies, selecting seats, and booking tickets.
* Integrating backend APIs to handle authentication, booking, and administrative functions.
* Connecting the frontend and backend to ensure smooth data flow.

**4. Testing and Validation:**

* Conducting functional testing to verify that all features work as expected.
* Performing usability testing to ensure a seamless user experience.
* Debugging and optimizing performance for responsiveness and efficiency.

**5. Deployment:**

* Hosting the application on a suitable platform and ensuring accessibility via web browsers.
* Maintaining the system for future updates and scalability.

This methodology ensures that the Movie Booking Website is **efficient, secure, and scalable**, providing a reliable platform for both users and administrators.

**3.3 Classes and Functional Responsibilities**

The Movie Booking Website is structured using a modular approach where different classes/components handle specific functionalities. The key classes and their responsibilities are as follows:

**1. User Class:**

* **Responsibilities:**
  + Register and log in to the system.
  + Browse movies and view details such as posters, timings, and descriptions.
  + Select show timings and preferred seats.
  + Book tickets and view booking history.
  + Update personal information.

**2. Movie Class:**

* **Responsibilities:**
  + Store movie details including title, genre, duration, show timings, poster URL, and description.
  + Provide movie data to the frontend for display.
  + Support search and filtering operations for easy navigation.

**3. Booking Class:**

* **Responsibilities:**
  + Handle ticket booking requests.
  + Manage selected seats and ensure real-time seat availability.
  + Record booking details including user, movie, timing, seats, and payment status.
  + Generate booking confirmation for users.

**4. Admin Class:**

* **Responsibilities:**
  + Add, edit, or delete movie listings.
  + Upload posters and provide detailed descriptions.
  + Manage show timings and theater schedules.
  + Maintain overall system data integrity and monitor bookings.

**5. Database Handler Class (or DB Interface):**

* **Responsibilities:**
  + Establish connection with MongoDB.
  + Perform CRUD operations on Users, Movies, and Bookings collections.
  + Ensure secure data storage and retrieval.

This class structure ensures **modularity, maintainability, and scalability**, making the system easy to manage and extend with additional features in the future.

**3.4 File Structure Description**

**backend**/

│── **models**/

│ ├── User.js

│ ├── Movie.js

│ └── Booking.js

│

│── **routes**/

│ ├── auth.js

│ ├── movies.js

│ └── bookings.js

│

│── **controllers**/

│ ├── authController.js

│ ├── movieController.js

│ └── bookingController.js

│

│── server.js

│── .env

│── package.json

**frontend**/

│── src/

│ ├── **components**/

│ │ ├── MovieCard.jsx

│ │ ├── SeatSelector.jsx

│ │ └── Navbar.jsx

│ │

│ ├── **pages**/

│ │ ├── Home.jsx

│ │ ├── MovieDetails.jsx

│ │ ├── BookingHistory.jsx

│ │ └── AdminDashboard.jsx

│ │

│ ├── App.jsx

│ └── index.js

│

│── package.json

**3.5 Data Flow and Processing Logic**

The Movie Booking Website follows a **structured data flow** to ensure seamless interactions between the user, frontend, backend, and database. The processing logic is designed to handle user requests efficiently while maintaining data integrity and real-time updates.

**1. User Interaction:**

* Users interact with the **frontend React.js interface** to browse movies, view show timings, select seats, and book tickets.
* Input data, such as login credentials or seat selection, is validated on the client side for immediate feedback.

**2. Backend Processing:**

* User requests are sent to the **Node.js and Express.js backend** via RESTful APIs.
* The backend processes the request by performing necessary validations (e.g., seat availability, user authentication).
* Business logic ensures that operations like booking, canceling, or updating user profiles are executed accurately.

**3. Database Interaction:**

* Backend communicates with **MongoDB** to fetch, update, or store data related to users, movies, and bookings.
* Real-time updates prevent double bookings and ensure that seat availability is always current.

**4. Response to User:**

* After processing, the backend sends a response to the frontend, confirming successful bookings or displaying relevant error messages.
* The frontend updates the user interface dynamically, reflecting changes in booking status, seat selection, or movie listings.

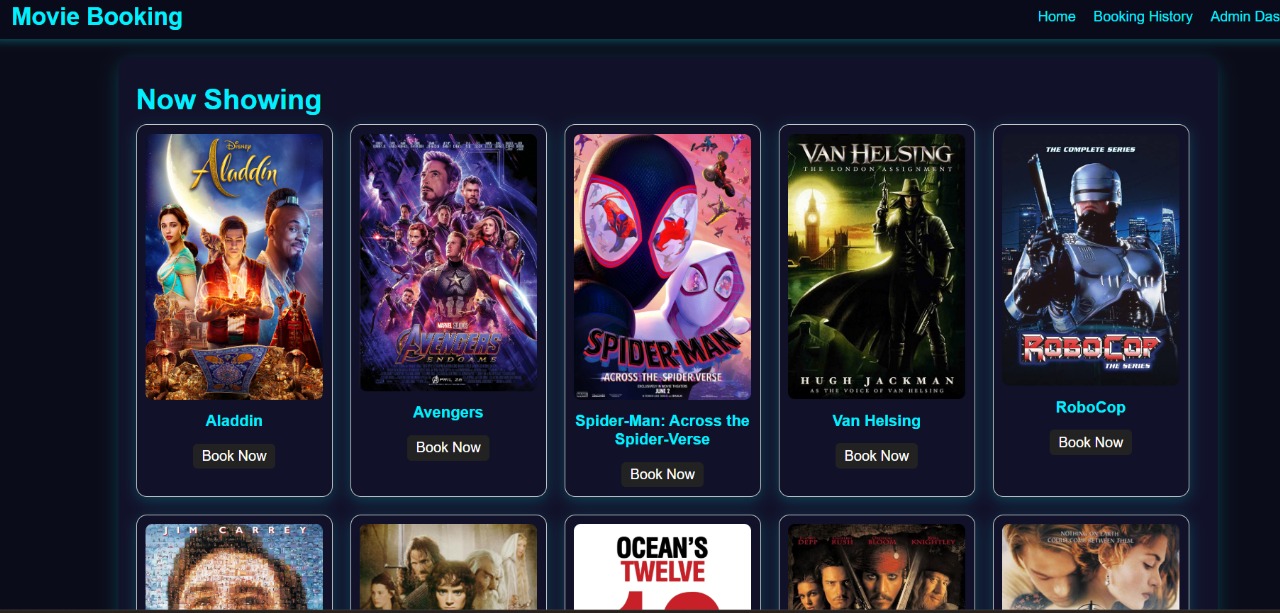
**5. Administrative Processing:**

* Admin actions, such as adding or editing movies, are handled similarly: validated in the backend, stored in the database, and reflected immediately on the frontend.

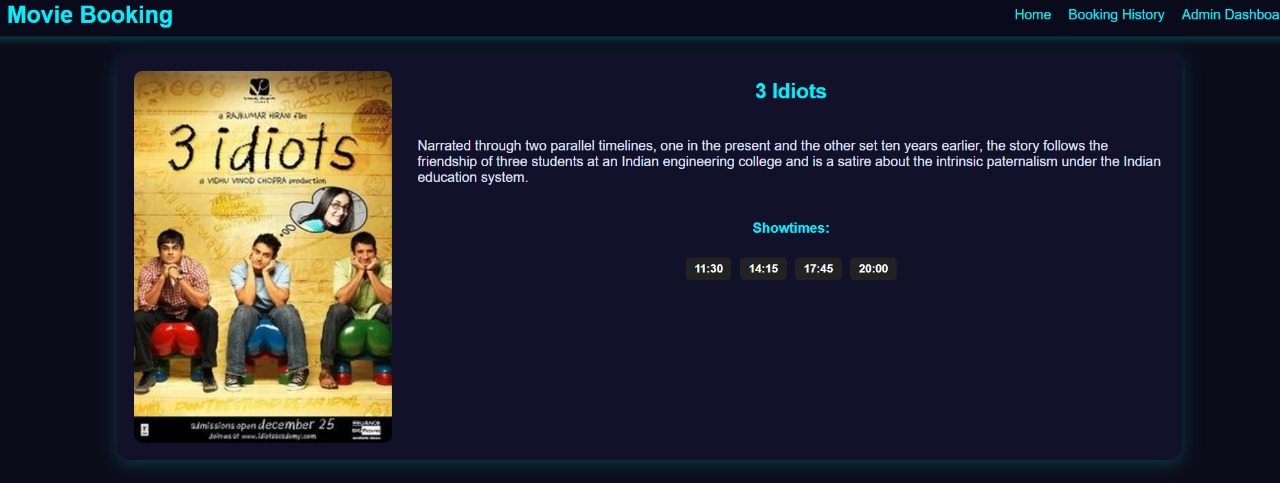
This **data flow and processing logic** ensures smooth, secure, and efficient operations, providing users with a reliable movie booking experience while enabling administrators to manage the system effectively.

**4. RESULTS**

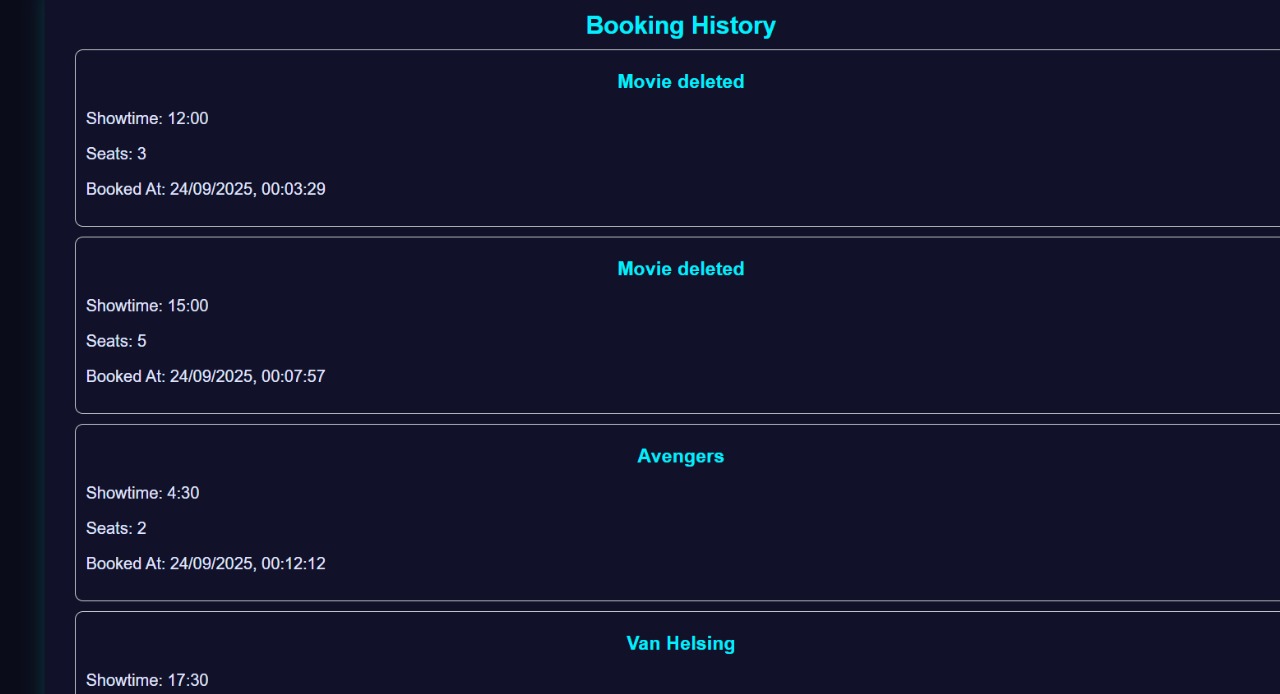
**4.1 Site Screenshots**



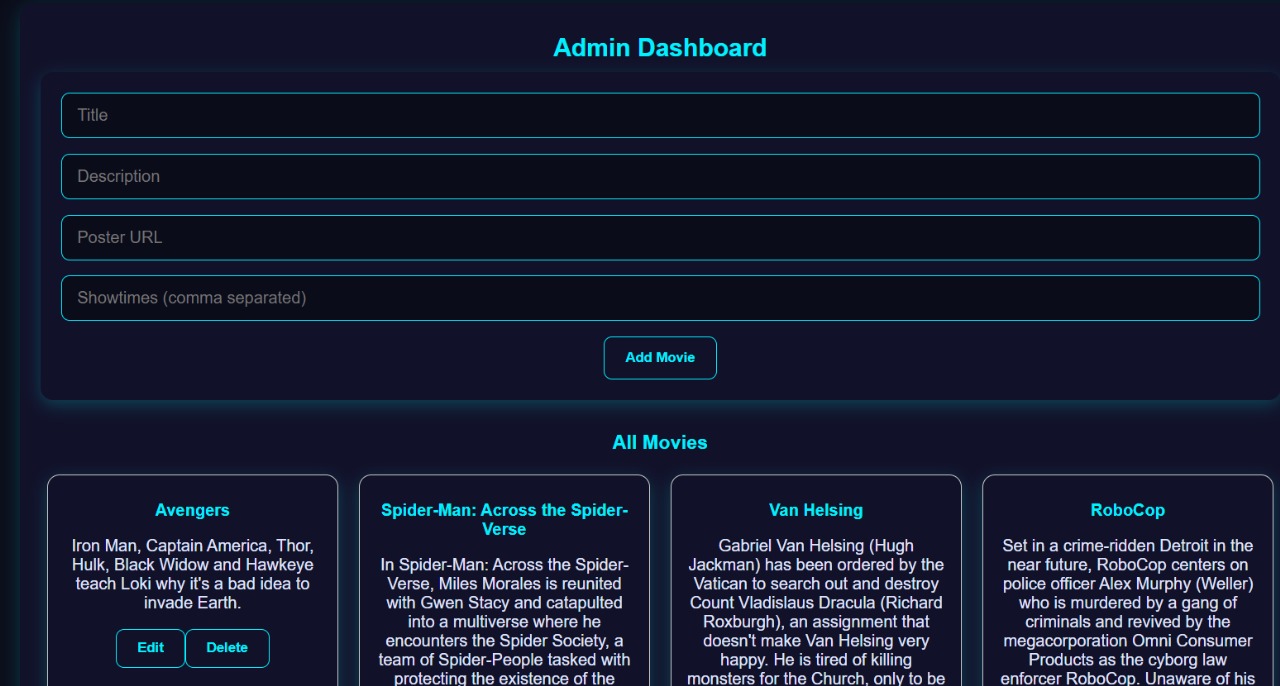
**Fig 1. Home Page**



**Fig 2. Movie Details**



**Fig 3. Booking History**

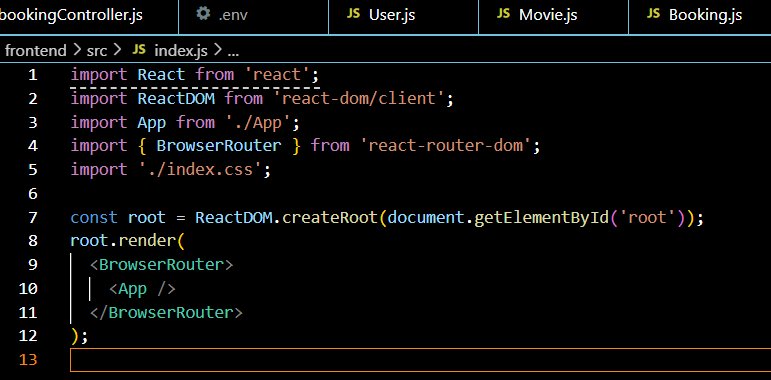


**Fig 4. Admin Dashboard**

**4.2 Code Screenshots**



**Fig 5. server.js (Backend)**



**Fig 6. index.js (Front-end)**

### 5. REFERENCES

1. Bass, L., Clements, P., & Kazman, R. (2013). *Software Architecture in Practice (3rd Edition)*. Addison-Wesley.
2. Sommerville, I. (2016). *Software Engineering (10th Edition)*. Pearson.
3. Pressman, R. S., & Maxim, B. R. (2020). *Software Engineering: A Practitioner’s Approach (9th Edition)*. McGraw-Hill.
4. MongoDB, Inc. (2024). *MongoDB Documentation*. Retrieved from <https://www.mongodb.com/docs/>
5. Node.js Foundation. (2024). *Node.js Documentation*. Retrieved from [https://nodReact.js.org/en/docs/](https://nodejs.org/en/docs/)
6. Express.js Team. (2024). *Express.js Guide*. Retrieved from <https://expressjs.com/>
7. REACT.JS Templating. (2024). *REACT.JS Official Documentation*. Retrieved from [https://React.js.co/](https://ejs.co/)
8. Redux Toolkit. (2024). *Redux Documentation*. Retrieved from <https://redux.js.org/>
9. Wazid, M., Das, A. K., Shetty, S., & Conti, M. (2019). “A Tutorial and Future Research for Secure Internet of Things: Authentication, Authorization, and Data Protection.” *IEEE Communications Surveys & Tutorials, 21*(4), 3449–3501.
10. Kumar, R., & Singh, P. (2021). “Design and Implementation of a Web-Based Student Management System.” *International Journal of Computer Applications, 183*(24), 12–18.
11. TutorialsPoint. (2024). *MERN Stack Tutorial*. Retrieved from https://www.tutorialspoint.com/mern\_stack/index.htm