Real Time Movie Ticket Booking Application
"Moviegoer"

Learner Details

- **Name: Shrayanth S**
- **Enrollment Number: SU625MR011**
- **Batch / Class: June 2025 MERN**
- **Assignment: Software Requirement Specification documentation for "Real Time Movie Ticket Booking Application Using React.js"**
- **Date of Submission: 18/08/2025**

---

1. Introduction

1.1 Purpose

The purpose of the Real Time Movie Ticket Booking Application is to provide users with a convenient and interactive platform to browse movies, view details, select cities, choose seats, and book tickets online. It eliminates the need for manual booking and enhances the moviegoing experience with a seamless, user-friendly interface.

1.2 Scope

The application will serve as a real-time ticket booking system built using React.js. It will provide the following features:
• User registration and login functionality.
• Movie catalog with trailers, cast details, and reviews.
• City selection for localized availability.
• Seat selection with availability indication.
• Checkout process with Terms & Conditions acceptance.
• Booking confirmation summary.

Future versions may include payment integration, admin dashboards, and real-time seat availability from a backend system.

1.3 Definitions, Acronyms, Abbreviations

• React.js: A JavaScript library for building user interfaces.
• React Router: A routing library for managing navigation in React apps.

• State: Data variables managed within React components using hooks.
• Backend/Database (Future Scope): Stores persistent data such as users, movies, and bookings.

2. Overall Description

2.1 Product Perspective

The Movie Ticket Booking Application is a standalone React application focusing on frontend functionality. It allows users to interact with movies, seats, and booking flows using client-side state management. In the future, it can be integrated with a backend for authentication, payments, and real-time seat availability.

2.2 Product Features

• Homepage with movie listings and navigation.
• Movie details page with trailer, cast, and reviews.
• City and seat selection for ticket booking.
• Checkout with Terms & Conditions acceptance.
• Booking confirmation summary.
• Future scope: Admin dashboard, payment integration, and backend support.

2.3 User Classes and Characteristics

• Guest Users: Can browse movies but must log in to book tickets.
• Registered Users: Can log in, book tickets, and confirm seats.
• Admin (Future Scope): Manage movies, bookings, and users.

2.4 Operating Environment

• Frontend: React.js, React Router, HTML, CSS.
• Browser Support: Chrome, Edge, Safari, Firefox.
• Future Backend (optional): Node.js, Express, MongoDB.

3. System Features

3.1 Movie List

Displays available movies with title, rating, and links to details.

3.2 Movie Details

Shows trailer, cast, reviews, and option to book tickets.

3.3 Booking

City and seat selection process with navigation to checkout.

3.4 Checkout

Displays booking summary and requires Terms & Conditions acceptance.

3.5 Confirmation

Shows confirmation message with movie, city, and seats booked.

4. Data Flow Diagram

4.1 Context Level (Level 0 DFD)

At the highest level, the system allows a User to interact with the Movie Ticket Booking Application. The user provides login/signup data, selects movies, cities, and seats, and receives booking confirmation.

Entities:

- User – Provides inputs (login, signup, movie selection, seat selection).
- Application (React Frontend) – Processes inputs, updates UI and state.
- (Future) Backend/Database – Stores users, movies, and booking data.

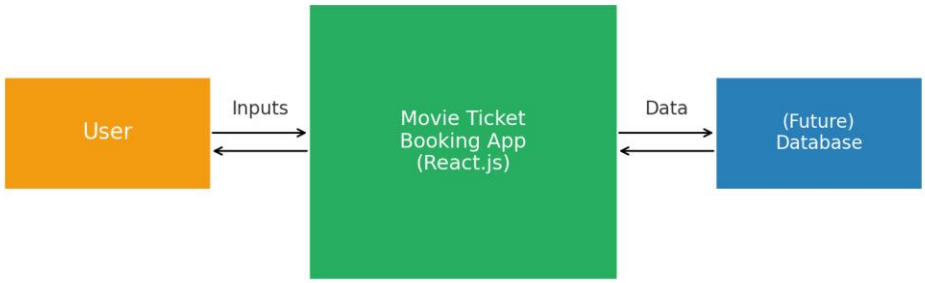

Figure 1: Context Level (Level 0 DFD)

4.2 Level 1 DFD (Frontend Flow)

- **User Authentication**
  Input: Email, Password → Stored in React state (users[] and loggedInUser).
  Output: Login success/failure message.

- **Movie Browsing**
  Input: User clicks on "Movies" → Application fetches movie list from state.
  Output: Movie details (title, trailer, cast, reviews).

• **Booking Process**
Input: User selects city and seats.
State updates: selectedCity, selectedSeats.
Output: Redirects to checkout.

• **Checkout & Confirmation**
Input: User accepts Terms & Conditions.
State validation: Confirm acceptance.
Output: Booking confirmation message (summary of movie, city, seats).

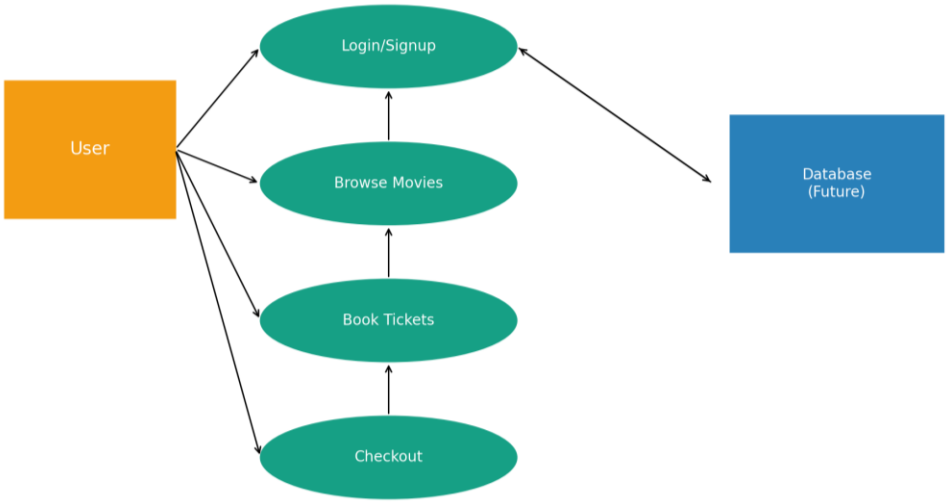

Figure 2: Level 1 DFD (Frontend Flow)

4. Non-Functional Requirements

- **Performance**:
  The movie list, details, and seat selection pages should load quickly (within 2–3 seconds) to ensure smooth user experience. Seat selection and booking confirmation should update instantly without page reloads (using React state).
- **Scalability**:
  The application should be designed to easily integrate with a backend in the future, allowing support for thousands of users booking tickets simultaneously and handling multiple cities, movies, and show timings.

- **Security**:
  Currently, user data (login credentials) is stored in React state for demo purposes. In future backend integration, secure authentication, encrypted storage, and HTTPS communication must be implemented to protect sensitive information.

- **Usability**:
  The system should have a clean and intuitive UI. Features like login, movie browsing, seat selection, and checkout must be easy to navigate for both casual users and frequent moviegoers. The design must be mobile-friendly to support bookings on smartphones.

- **Maintainability**:
  The codebase uses modular React components (Navbar, MovieList, MovieDetails, BookTickets, SeatSelection, Checkout, etc.), making it easy to debug, update, and extend with new features such as payment integration or admin dashboards.

- **Compatibility**:
  The application should run smoothly across popular browsers (Google Chrome, Mozilla Firefox, Microsoft Edge, Safari) and adapt well to different screen sizes, ensuring consistent experience across desktop, tablet, and mobile devices.

---

5. Conclusion

The Real Time Movie Ticket Booking Application using React.js provides a seamless solution for booking movie tickets online. It allows users to log in, browse available movies, select cities and seats, and confirm bookings in real time.

The system is designed with performance, usability, and scalability in mind. While the current version focuses on frontend functionality, it provides a strong foundation for future expansion with backend support, payment integration, and admin features. This ensures the application remains reliable, user-friendly, and adaptable as it grows, enhancing the overall moviegoing experience for users.