**Practical 11**

# **Generate SRS Document as per given format.**

# **Topic: Movie Booking Management System**

**1. Introduction**

**1.1 Purpose**

This document details the functional and non-functional requirements for the Movie Booking Management System. It serves as a comprehensive guide for developers, testers, project managers, and stakeholders by:

* Defining the system’s functionalities, performance standards, and interface expectations.
* Establishing a framework for design, implementation, and validation.
* Ensuring a shared understanding of the project scope and objectives throughout the Software Development Life Cycle (SDLC).

**1.2 Need of the Application**

The digital transformation in the cinema industry necessitates an efficient and user-friendly ticket booking system. The Movie Booking Management System addresses these needs by:

* **Enhancing User Experience:** Customers can search for movies, view real-time seat availability, and book tickets online, reducing the reliance on manual processes.
* **Optimizing Operations:** Theatre staff and administrators can manage movie listings, schedules, and bookings with greater accuracy and efficiency.
* **Driving Business Growth:** A streamlined digital ticketing process attracts a larger audience and opens opportunities for targeted promotions and data-driven decision-making.
* **Providing Secure Transactions:** Robust security measures are integrated to protect user data and ensure safe online payment processing.

**1.3 Scope**

The scope of the system includes the following features and modules:

* **User Interactions:**
  + Registration, authentication, and profile management for customers.
  + Movie browsing and filtering with real-time seat selection.
* **Booking and Payment:**
  + Secure, seamless ticket booking integrated with external payment gateways.
  + Automated generation and delivery of booking confirmations and tickets.
* **Administrative Functions:**
  + Management of movie listings, showtimes, and user accounts.
  + Generation of comprehensive reports on bookings, revenue, and theatre operations.
* **System Integration:**
  + Compatibility across various devices and platforms.
  + Easy integration with third-party services such as payment providers.

**1.4 Platform Specifications**

**1.4.1 Software Specification**

* **Frontend:**
  + **Technologies:** HTML5, CSS3, JavaScript, and modern frameworks (React or Angular) to create a dynamic and responsive user interface.
  + **Features:** Responsive design, user-friendly layouts, and interactive components that comply with accessibility standards.
* **Backend:**
  + **Technologies:** Server-side implementation using Python (Django/Flask) or Node.js, utilizing RESTful API architecture.
  + **Functions:** Handling business logic, user authentication, data processing, and transaction management.
* **Database:**
  + **Systems:** Relational database management systems (e.g., MySQL or PostgreSQL) with a normalized schema to ensure data integrity.
* **Security & Integration:**
  + **Standards:** Secure authentication (e.g., JWT), encryption protocols (AES-256) for sensitive data, and integration with external payment gateways such as Stripe or PayPal.

**1.4.2 Hardware Specification**

* **Server Infrastructure:**
  + **Processing:** Multi-core processors to support high-concurrency transactions.
  + **Memory & Storage:** Minimum of 8GB RAM and high-speed storage solutions (SSDs, RAID configurations) to ensure rapid data access and scalability.
* **Client Devices:**
  + Modern desktop computers, tablets, and smartphones with updated web browsers.
* **Network:**
  + A reliable, secure network infrastructure that guarantees fast and uninterrupted data transmission between clients and servers.

**2. System Requirement Analysis**

**2.1 Product Perspective**

The Movie Booking Management System is a modular, web-based application designed to operate both as a stand-alone solution and as an integrated part of larger theatre management systems. Its core components include:

* **User Interface Modules:** Customized portals for customers, administrators, and theatre managers.
* **Backend Services:** Processes for handling data management, transaction processing, and secure communications.
* **Data Integration:** Seamless connectivity with external systems, such as payment gateways and analytics services, ensuring a cohesive ecosystem.

**2.2 System Interfaces**

**2.2.1 Economic Feasibility**

* **Cost Reduction:** Automation reduces the need for manual ticketing processes, thereby lowering labor costs and minimizing errors.
* **Revenue Opportunities:** Enhanced customer experience and operational efficiency lead to increased ticket sales and potential for upselling promotions.
* **Investment Return:** The initial development and integration costs are justified by long-term savings in operational costs and improved service delivery.

**2.2.2 Technical Feasibility**

* **Proven Technologies:** Utilizing established platforms such as Python/Node.js and MySQL/PostgreSQL ensures a reliable and maintainable system.
* **Scalability:** The use of cloud hosting solutions (AWS, Azure, or Google Cloud) enables the system to scale dynamically based on user demand.
* **Security:** Implementation of industry-standard security practices minimizes the risk of data breaches and unauthorized access.

**2.2.3 Behavioral Feasibility**

* **User-Centric Design:** The system is designed with intuitive interfaces that cater to both technical and non-technical users.
* **Performance & Reliability:** Rigorous testing and a robust architectural design ensure high availability and responsiveness, even under heavy load.
* **Adaptability:** A modular architecture facilitates future enhancements and the integration of additional functionalities without significant restructuring.

**3. System Analysis**

**3.1 Entity-Relationship (ER) Diagram**

The ER Diagram provides a visual representation of the system’s database structure. It defines key entities such as Users, Movies, Theatres, Showtimes, Bookings, and Payments, along with their attributes and relationships. The purpose of this diagram is to ensure data integrity, support database normalization, and serve as a blueprint for how information is organized and interrelated within the system. It is essential for designing a robust and efficient database that underpins the application.

A computer screen shot of a diagram

Description automatically generated

**Fig 1: E-R Diagram**

**3.2 Data Flow Diagram (DFD)**

The DFD illustrates the flow of data through the system. It identifies external entities (e.g., customers and administrators), internal processes (such as registration, movie search, booking, and payment processing), and data stores (like customer and booking databases). The diagram’s purpose is to provide a clear understanding of how data is collected, processed, stored, and delivered, highlighting the interaction between various system components. This clarity helps in identifying potential bottlenecks and ensures that all necessary data interactions are efficiently managed.

A screen shot of a movie ticket system

Description automatically generated

Fig 2: Level 0 DFD Diagram

A diagram of a movie project

Description automatically generated

Fig 3: Level 1 DFD Diagram

A screenshot of a computer screen

Description automatically generated

Fig 4: Level 2 DFD Diagram

**3.3 Use Case Diagram**

The Use Case Diagram outlines the interactions between system users (actors) and the system functionalities. It lists key use cases such as booking tickets, searching for movies, managing schedules, and processing payments. The diagram defines system boundaries and clarifies the roles and expectations of each user type. Its purpose is to serve as a communication tool, ensuring that all stakeholders understand the operational scope and that all functional requirements are addressed during development.

A screenshot of a diagram

Description automatically generated

Fig 5: Use Case Diagram

**3.4 Activity Diagram**

The Activity Diagram maps out the sequential flow of actions within a process—specifically, the movie ticket booking workflow. It depicts each step from the initial movie search, through seat selection and payment processing, to the final booking confirmation. The diagram identifies decision points, parallel processes, and overall workflow logic. Its purpose is to provide clarity on process execution, facilitate workflow optimization, and help developers understand how different activities interact within the system.

A diagram of a flowchart

Description automatically generated

Fig 6: Activity Diagram

**3.5 Sequence Diagram**

The Sequence Diagram focuses on the order and timing of interactions among system components during a transaction. It details the message exchanges between the customer interface, application server, database, and external payment gateway. The purpose of this diagram is to ensure that all system interactions occur in the correct sequence, verify that dependencies are managed properly, and aid developers in implementing precise control flows and error handling routines.



Fig 7: Sequence Diagram

**3.6 State Transition Diagram**

The State Transition Diagram illustrates the lifecycle of a booking by defining various states such as "Initiated," "Payment Pending," "Confirmed," and "Cancelled." It also specifies the events that trigger transitions between these states. The purpose of this diagram is to manage dynamic system behavior, ensure that state changes are handled consistently, and provide a clear mechanism for tracking the status of a booking throughout its lifecycle.

A diagram of a company

AI-generated content may be incorrect.

Fig 8: State Transition Diagram

**3.7 Class Diagram**

The Class Diagram provides an object-oriented view of the system, outlining the key classes (e.g., User, Movie, Theatre, Showtime, Booking, Payment), their attributes, and methods. It details the relationships and interactions between these classes. The purpose of the class diagram is to serve as a blueprint for the system architecture, guide developers in writing modular and reusable code, and ensure that the object model accurately reflects the system’s functional requirements.

Fig 9: Class Diagram

**4. Conclusion**

This SRS document offers an in-depth framework for the Movie Booking Management System. It covers the complete scope of functional and non-functional requirements, system analysis, and architectural design. Detailed explanations for each diagram clarify their purpose and role in the system, ensuring that when the actual images are inserted, they will complement the documented requirements effectively. This living document is intended to evolve with the project, serving as a reliable reference for developers, testers, and stakeholders throughout the development lifecycle.

**5. References**

[1] Zozi, Online booking system

URL: https://www.zozi.com/advance/blog/the-advantages-and-disadvantages-of-online-

booking-systems. (2016).

[2] Wikipedia, Computer reservations system.

URL: https://en.wikipedia.org/wiki/Computer\_reservations\_system. (2016).