****

ONLINE TICKET MANAGEMENT SYSTEM

A MINI PROJECT REPORT

|  |  |  |
| --- | --- | --- |
| **Submitted** | **by** |  |
| **Rishikesan** | |  | **231801137** |
| **Sanjay ramkumar** |  | **231801153** |
| **sanjay** |  | **231801150** |
|  | by |  |

**CS23332 DATABASE MANAGEMENT SYSTEM**

**Department of Artificial Intelligence and Data Science**

**Rajalakshmi Engineering College, Thandalam**

2024-2025

**BONAFIDE CERTIFICATE**

Certified that this project report “**ONLINE TICKET MANAGEMENT SYSTEM**” is the bonafide work of **“SANYAY RAMKUMAR (231801153),RISHIKESAN (2318011137), SANJAY (231801150),”**

who carried out the project work under my supervision.

**Submitted for the Practical Examination held on**

**SIGNATURE SIGNATURE**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Dr. GNANASEKAR J M** | **Dr. MANORANJINI J** | **Dr.GNANASEKAR J M** | **Dr. MANORANJINI J** | **Dr.MANORANJINI J** |
| **Head of the Department, Artificial intelligence**  **and data Science,Rajalakshmi Engineering College (Autonomous),Chennai-602105** | **Assoc.Professor, Artificial Intelligence and Data**  **Science, Rajalakshmi Engineering College (Autonomous), Thandalam, Chennai-602105** | |  | | --- | | **Head of the Department, Artificial intelligence**  **and data Science,Rajalakshmi Engineering College (Autonomous),Chennai-602105** | | **Professor, Artificial Intelligence and Data**  **Science, Rajalakshmi Engineering College (Autonomous), Thandalam, Chennai-602105** | **Associate Professor, Artificial Intelligence and Data Science, Rajalakshmi Engineering College,(Autonomous), Thandalam, Chennai-602105** |

**INTERNAL EXAMINER EXTERNAL EXAMINER**

**S.NO CHAPTER**

**1. INTRODUCTION**

1.1 GENERAL

1.2 OBJECTIVES

1.3 SCOPE

**2. SYSTEM OVERVIEW**

2. SYSTEM ARCHITECTURE

2.2 MODULES OVERVIEW

2.3 USER ROLES AND ACCESS LEVELS

**3. SURVEY OF TECHNOLOGIES**

3.1 SOFTWARE AND TOOLS USED

3.2 PROGRAMMING LANGUAGES

3.3 FRAMEWORKS AND LIBRARIES

**4. REQUIREMENTS AND ANALYSIS**

4.1 FUNCTIONAL REQUIREMENTS

4.2 NON-Functional Requirements

4.3 HARDWARE AND SOFTWARE REQUIREMENTS

4.4 ARCHITECTURE DIAGRAM

4.5 ER DIAGRAM

**5. SYSTEM DESIGN**

5.1DATABASE DESIGN AND TABLES

5.2 UI DESIGN OVERVIEW

5.3 WORKFLOW AND PROCESS DIAGRAMS

**6. IMPLEMENTATION**

6.1 CODE STRUCTURE AND ORGANIZATION

6.2 KEY MODULES AND THEIR FUNCTIONS

6.3 CHALLENGES AND SOLUTIONS

**7. TESTING AND VALIDATION**

7.1 TESTING STRATEGIES

7.2 TEST CASES AND RESULTS

7.3 BUG FIXES AND IMPROVEMENTS

**8. RESULTS AND DISCUSSION**

8.1 SUMMARY OF FEATURES

8.2 USER EXPERIENCE FEEDBACK

8.3 POTENTIAL IMPROVEMENTS

9 CONCLUSION

**10. REFERENCES**

**ABSTRACT**

The Online Movie Ticket Reservation System is a web-based platform designed to streamline the process of booking tickets for movies, enhancing convenience and efficiency for users. In this system, users can log in to the website using their registered credentials, which include a unique username and password. This authentication process ensures secure access to the platform. Once logged in, users can browse the list of currently available movies, choose their desired movie, and proceed with the ticket booking process.

The platform provides detailed information about each movie, such as its title, genre, language, duration, and show timings. Users can select a suitable showtime from the options provided and specify the number of tickets they wish to book. The system also displays information about the theaters where the selected movie is running, allowing users to choose their preferred theater based on location or availability.

After selecting the movie, showtime, and theater, the user is guided through a streamlined booking process. The system checks seat availability in real-time, ensuring that users can only select seats that are currently vacant. Once the seat selection is finalized, the system calculates the total ticket cost and generates a detailed summary of the booking. This summary includes information such as the movie name, theater, showtime, number of tickets, seat numbers, and total cost.

Users can then proceed to confirm their booking by completing the payment process. The system offers a secure payment gateway, supporting various payment methods such as credit/debit cards and digital wallets. After successful payment, the system generates a digital ticket, which can be printed or presented at the respective ticket counter for validation.

This project eliminates the need for users to visit physical ticket counters and stand in long queues. It also reduces manual errors and provides a convenient, time-saving solution for moviegoers. The system ensures a smooth and hassle-free user experience while maintaining the security and accuracy of the booking process. The platform is designed to handle high user traffic efficiently and provides administrators with tools to manage movie schedules, theaters, and booking data.

In conclusion, the Online Movie Ticket Reservation System not only simplifies the ticket booking process but also enhances the overall moviegoing experience for users, providing them with a modern, digital alternative to traditional ticketing methods.

**1. INTRODUCTION**

**1.1 GENERAL**

The Online Movie Ticket Booking System is a web-based platform designed to streamline and automate the process of booking movie tickets. This system provides users with an easy-to-use interface where they can search for movies, view show timings, and book tickets from the comfort of their homes or on-the-go. For administrators, the system offers comprehensive tools to manage movie listings, user accounts, and booking records, as well as the ability to generate reports for analysis and decision-making. By using this system, theaters can efficiently handle the operational tasks of managing shows, movie data, and customer bookings, ensuring a seamless experience for both users and administrators. The primary aim of the system is to replace the traditional manual booking methods with a more efficient, time-saving, and secure digital process.

The system is designed to be scalable, secure, and user-friendly, making it accessible to a wide range of users. The system can handle multiple user requests simultaneously, ensuring that it can support large numbers of customers, especially during peak hours like weekends and holidays.

**1.2 OBJECTIVES**

**The objectives of the Online Movie Ticket Booking System are as follows:**

* Simplify the Ticket Booking Process: The system aims to provide a simplified process for users to book tickets for their desired movies, select seats, and make payments securely online. Users should be able to book tickets with just a few clicks.
* Real-time Movie and Showtime Availability: The system should provide up-to-date movie schedules and availability in real-time, ensuring users can book tickets only for available showtimes and seats. This minimizes issues related to double-booking and unavailable tickets.
* Efficient Management for Administrators: The system allows administrators to manage movies, showtimes, user accounts, and bookings efficiently. The admin panel provides tools for updating movie listings, adding or removing movies, and managing showtimes.
* Report Generation for Analysis: Admin users can generate various reports, such as booking statistics, revenue generation, and user activity, which can help theater management in decision-making and performance analysis.
* Enhanced User Experience: The user interface should be intuitive, easy to navigate, and visually appealing. It should provide users with a hassle-free experience from the moment they land on the website to when they confirm their ticket booking.
* Secure Online Transactions: Ensure that all financial transactions, such as payment for movie tickets, are secure and encrypted, providing a safe experience for users.
* Flexibility for Multiple User Roles: The system is designed to handle different user roles such as Admin, User, and Guest, each with specific access levels, providing a controlled and secure environment.

**1.3 SCOPE**

The scope of the Online Movie Ticket Booking System includes both functional and non-functional requirements that define its capabilities and limitations:

* User Management:  
  The system provides functionality for users to register, login, and manage their profiles. Registered users can view their booking history and update their account details. The login system is secure, ensuring that users' personal information is protected.
* Movie Management:  
  The admin can add, update, and delete movies from the system. Movie details, such as title, genre, language, and duration, are stored in the database. This allows the system to display accurate information for users to make informed decisions while booking tickets.
* Showtime Management:  
  The system allows the admin to schedule movie showtimes. This includes setting specific times for each movie across multiple days and managing the availability of seats. The system will automatically adjust the available seat count as bookings are made.
* Booking and Payment System:  
  Users can browse the list of available movies, select their preferred showtime, choose their seats, and proceed with the payment process. The system should offer multiple payment options, including credit/debit cards, net banking, or other online payment gateways.
* Admin Panel:  
  The system includes an admin panel for managing all aspects of the system. This includes user management, movie management, showtime scheduling, and the ability to view reports on bookings, users, and revenue.
* Report Generation:  
  The admin can generate various reports, such as total bookings, total revenue, and movie popularity. These reports are useful for decision-making and assessing the performance of the system.
* Security Features:  
  The system ensures that all user data, including login credentials and payment details, are encrypted and securely stored. Security features also include validation checks to prevent unauthorized access and data breaches.
* Mobile and Web Compatibility:  
  The system is designed to be responsive, ensuring that it works well on both desktop and mobile devices. This flexibility enables users to book tickets anytime, anywhere, on a variety of devices.
* Limitations:  
  The system is currently designed for a single theater chain or a single location, though scalability could be addressed in future versions. It also does not support live streaming or offline bookings.

This system aims to replace traditional manual booking methods and streamline the entire ticketing process, from booking to management, thereby improving operational efficiency and customer satisfaction.

**2. SYSTEM OVERVIEW**

**2.1 SYSTEM ARCHITECTURE**

**1. Presentation Layer (Client Layer)**

The Presentation Layer is responsible for the user interface. It interacts with the users, collects their inputs (such as movie selection, showtime, and seats), and displays information like available movies, showtimes, and booking confirmation. This layer is built using HTML, CSS, and JavaScript to ensure a user-friendly experience and is designed to be responsive, ensuring compatibility with various devices such as desktops and smartphones.

**2. Business Logic Layer (Application Layer)**

The Business Logic Layer processes the data received from the presentation layer. It validates the user inputs (e.g., booking requests, seat availability), handles the booking logic, calculates the total cost, and manages other business rules like seat allocation. This layer ensures the system's rules are applied correctly, such as preventing users from booking more tickets than available or selecting a sold-out show.

**3. Data Layer (Database Layer)**

The Data Layer is responsible for storing and managing all the data in the system. This includes user information, movie details, booking records, and showtimes. It interacts with a relational database (e.g., MySQL) to fetch, update, or delete data based on the operations performed in the business logic layer. The data layer ensures data integrity and security, protecting sensitive information like user credentials and payment details**.**

**2.2 MODULES OVERVIEW**

1. User Login Module

The User Login Module allows users to securely log in to the system with their credentials. The module is essential for ensuring only authorized users can access their account information, book tickets, and manage bookings. Key functionalities include:

* Login Form: The user enters their registered email address and password.
* Authentication: The system validates the login credentials against the database.
* Session Management: Upon successful login, the user’s session is created, allowing them to access various features, such as booking tickets and viewing their past bookings.

**2. New Registration Module**

The New Registration Module allows new users to create an account on the system. This is a one-time process that enables users to securely log in and make bookings. Key features of this module include:

* Registration Form: The user fills in details such as name, email address, password, and contact information.
* **Email Validation: Ensures the email provided by the user is valid and not already in use.**
* Password Security: Users must create a secure password, and the system may include password strength guidelines (e.g., minimum length, special characters).
* Account Confirmation: After registration, the user receives a confirmation email to verify their account.

**3. Selection of Movie Time, Date, Theater, and Ticket Details**

This module allows users to select their preferred movie, showtime, theater, and the number of tickets. It ensures that users can easily choose from available movies and make informed decisions. Features of this module include:

* Movie Selection: Users can browse and select movies available for booking.
* Showtime and Date: After selecting a movie, users can choose from available showtimes and dates.
* Theater Selection: Displays available theaters showing the selected movie at the chosen time.
* Ticket Details: Users specify the number of tickets they wish to book, and the system checks seat availability.

This module ensures that users have a smooth and informed movie selection and ticket booking experience.

**4. Book Ticket**

Once the movie, showtime, theater, and tickets are selected, the Book Ticket Module handles the final steps of the booking process. It includes:

* Seat Allocation: Users choose their preferred seats from the available options in the theater.
* Booking Confirmation: The system confirms the seat selection and calculates the total ticket price.
* Payment Processing: Users proceed to the payment stage to complete the booking**.**
* Ticket Generation: After successful payment, the system generates a digital ticket with all relevant details (movie, theater, time, seats, cost).

The Book Ticket Module ensures that users can reserve their tickets efficiently and securely.

**5. Cancel Ticket**

The Cancel Ticket Module allows users to cancel their previously booked tickets. This functionality is important for users who can no longer attend the movie or need to make changes to their plans. Key features include:

* View Bookings: Users can view their upcoming bookings and select the tickets they want to cancel.
* Cancellation Confirmation: The system confirms the cancellation request and checks if any cancellation fees apply.
* Refund Process: If applicable, the system processes any refunds according to the cancellation policy.
* Ticket Update: Once canceled, the seats are made available again for other users.

This moduleprovides users with flexibility, allowing them to modify their plans if needed**.**

**6. Card Selection**

The Card Selection Module is responsible for handling the payment process by selecting the method of payment. It typically includes:

* Choose Payment Method: Users can choose from various payment options, such as credit/debit cards, e-wallets, or online banking.
* Card Type Selection: The user selects the card type (Visa, MasterCard, etc.) for the transaction.

This module ensures that users can conveniently select their preferred payment method during the ticket booking process.

**7. Card Details**

The Card Details Module captures sensitive payment information for processing the payment. Key aspects include:

* Card Number: Users enter their credit or debit card number.
* Expiration Date: Users provide the expiration date of the card.
* CVV: Users enter the card’s CVV (Card Verification Value) for security.
* Billing Information: Users provide billing details such as name and address associated with the card.

This module ensures secure payment data collection and adheres to payment security standards (e.g., PCI-DSS).

**8. Generating Tickets**

After the payment process is completed successfully, the Generating Tickets Module issues the final booking confirmation and generates the user's tickets. Features of this module include:

* Ticket Details: The system generates a digital ticket that includes important details such as the movie name, showtime, theater, seat numbers, and total cost.
* QR Code/Barcode: The ticket may include a QR code or barcode for easy verification at the theater.
* Email Confirmation: The system sends the ticket details to the user’s email address as a confirmation and for easy access.
* Ticket History: Users can view their past tickets from their profile page.

This module completes the booking process by providing the user with a formal, verifiable ticket for the movie.

**2.3 USER ROLES AND ACCESS LEVELS**

**Admin**: Full access to all modules, including movie management, user management, and report generation.

**User**: Can view movies, book tickets, and manage their booking history.

**Guest**: Can browse movies and showtimes, but must register or log in to book tickets.

**3. SURVEY OF TECHNOLOGIES**

**3.1 SOFTWARE AND TOOLS USED**

**NetBeans IDE**: Used for coding the Java-based backend.

**MySQL**: Used for the database management system.

**Apache Tomcat**: For running the web application in a server environment.

**3.2 PROGRAMMING LANGUAGES**

**Java**: For the backend logic, handling movie showtimes, user registration, and booking functionality.

**SQL**: For database interaction, querying data about movies, users, and bookings.

**3.3 FRAMEWORKS AND LIBRARIES**

**JDBC**: Java Database Connectivity for integrating the database with the Java application.

**JSP/Servlets**: For creating dynamic web pages.

**4. REQUIREMENTS AND ANALYSIS**

**4.1 FUNCTIONAL REQUIREMENTS**

User registration and login system.

Ability to browse and search for movies.

Display available showtimes and seating options.

Make bookings and generate a confirmation.

Admin functionality for generating reports.

**4.2 NON-FUNCTIONAL REQUIREMENTS**

**Performance**: The system should support multiple users concurrently.

**Security**: User data should be protected, and the system should ensure secure transactions.

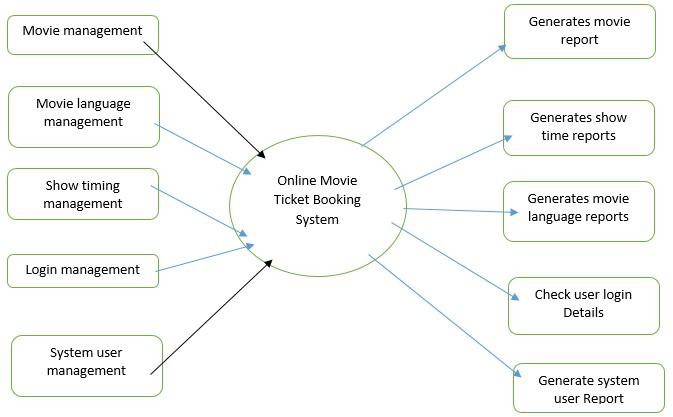
**Usability**: The system must be user-friendly and intuitive.

**4.3 HARDWARE AND SOFTWARE REQUIREMENTS**

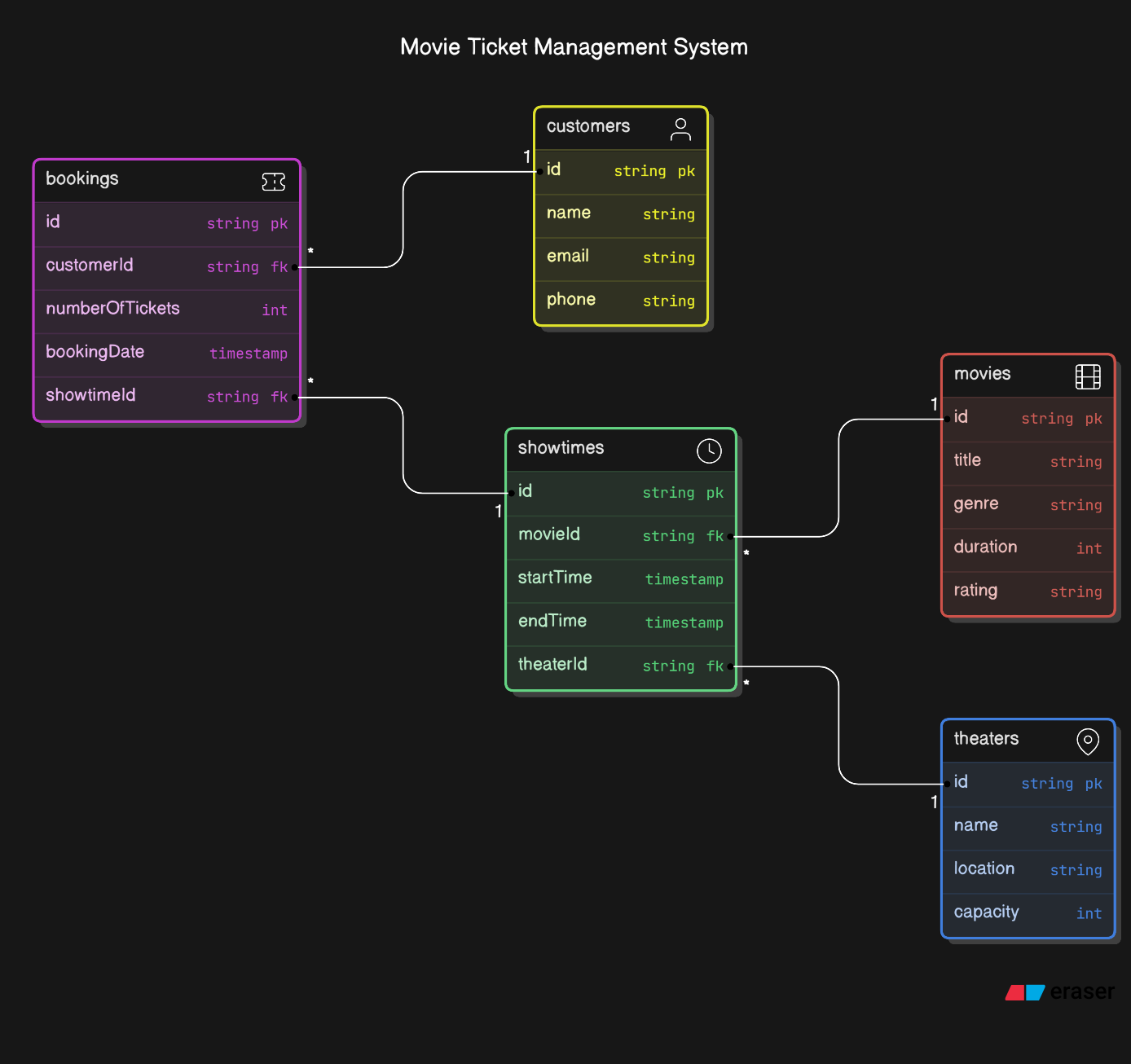
**Hardware**: Processor - Intel Core i3, 4GB RAM, 50GB Free Storage.

**Software**: Java, MySQL, Apache Tomcat.

**4.4 ARCHITECTURE DIAGRAM**



**4.5 ER DIAGRAM**

****

**5. SYSTEM DESIGN**

**5.1 DATABASE DESIGN AND TABLES**

The system uses relational tables in MySQL to store data:

**Movies**: Stores movie details like title, genre, language, etc.

**Users**: Stores user details for registration and login.

**Bookings**: Stores booking data, including user information and showtime.

**5.2 UI DESIGN OVERVIEW**

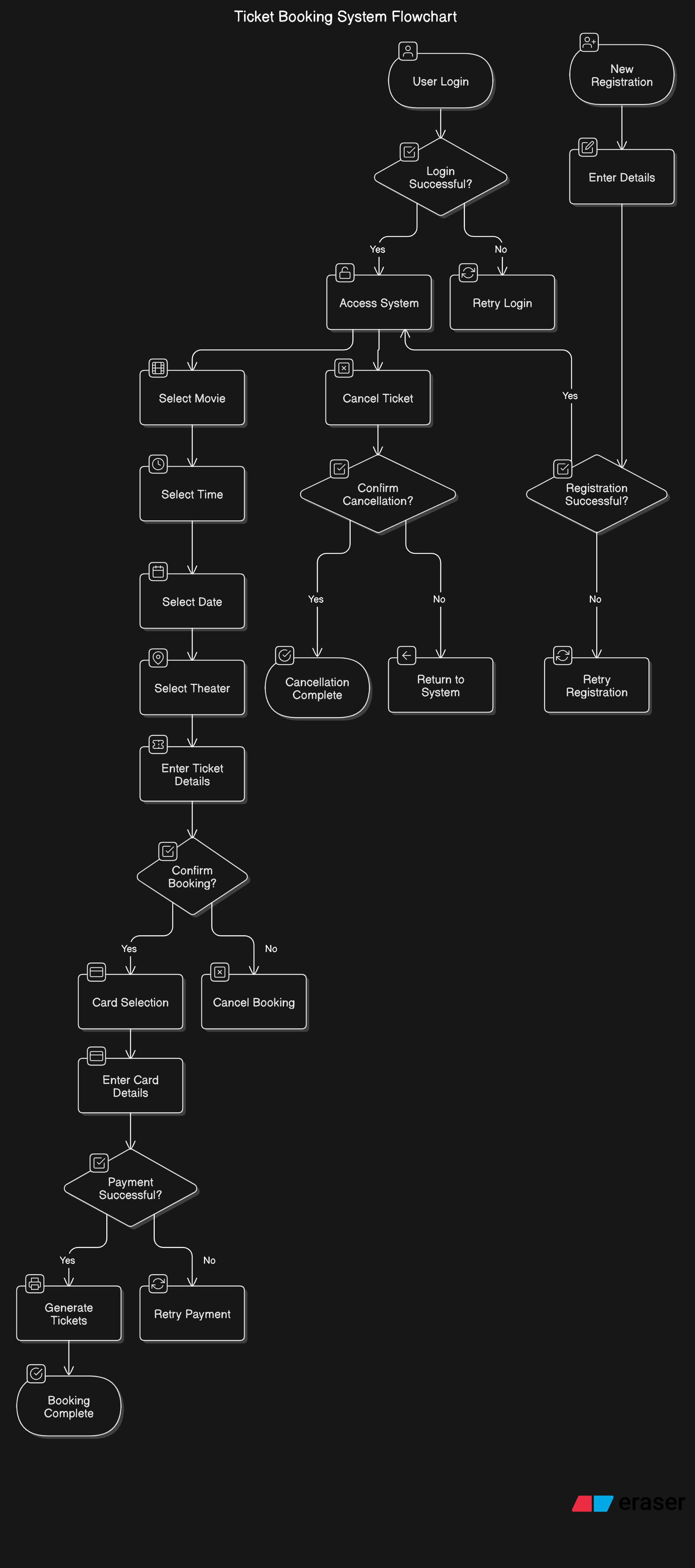
The interface is designed to be minimalistic and user-friendly:

Home page displays available movies.

A booking page allows users to select movies, showtimes, and seats.

Admin dashboard shows system status and reports.

**5.3 WORKFLOW AND PROCESS DIAGRAMS**



**6. IMPLEMENTATION**

**6.1 CODE STRUCTURE AND ORGANIZATION**

**1. Project Root Directory:**

* **This directory contains essential configuration files, setup scripts, and the main application files.**
* **Example files:**
  + **index.jsp: The main entry point of the application, displaying the homepage.**
  + **web.xml: Configuration file for servlet mappings and context parameters (if using Java servlets).**
  + **pom.xml (if using Maven): Manages dependencies and build configuration for the project.**
  + **application.properties: Contains configuration settings like database connection strings.**

**2. Controllers Folder:**

* **This folder contains Java files responsible for handling user requests, processing data, and rendering the appropriate views.**
* **Example files:**
  + UserController.java: Handles user login, registration, and profile management.
  + MovieController.java: Manages movie and showtime data, fetching available movies and managing user bookings.
  + BookingController.java: Handles the creation of new bookings, seat selection, and confirmation of ticket purchases.
  + PaymentController.java: Manages payment processing and validation.
  + AdminController.java: Admin functionality for managing users, movies, showtimes, and generating reports.

**3. Models Folder:**

* This folder contains Java files that represent the entities in the system and their relationships with each other, as well as database interactions.
* Example files:
  + User.java: Represents user data, including user credentials, booking history, and role (admin/user).
  + Movie.java: Represents movie details, such as title, genre, showtimes, and theater availability.
  + Showtime.java: Contains showtime data such as date, time, and theater.
  + Booking.java: Represents the booking entity, including seat selection, movie details, and user information.
  + Ticket.java: Represents the individual tickets, with details like seat number, movie, and booking status.
  + Payment.java: Manages the payment details for each ticket, such as the payment method, status, and amount.

**4. Views Folder:**

* This folder contains JSP or HTML files that define the user interface and render dynamic data sent from the controllers.
* Example files:
  + login.jsp: Displays the login form for users.
  + movieList.jsp: Displays the list of available movies for users to choose from.
  + bookingPage.jsp: Lets users choose showtimes, select seats, and confirm their booking.
  + paymentPage.jsp: Collects payment details and processes the payment.
  + confirmationPage.jsp: Displays the booking confirmation and ticket details**.**

**5. Static Folder:**

* This folder stores all the static files, including stylesheets, JavaScript, and images, that are used to enhance the user interface.
* Example files:
  + style.css: Defines the styling of the web pages.
  + scripts.js: Contains JavaScript functions for interactivity (e.g., form validation, seat selection).
  + logo.png: The logo image for the movie ticket booking website.
  + fonts/: Custom fonts used in the UI.

**6. Services Folder:**

* This folder contains Java service classes that implement the business logic of the application, interacting with models and performing tasks such as booking tickets, processing payments, and managing users.
* Example files:
  + UserService.java: Provides methods for user registration, login authentication, and user management.
  + MovieService.java: Provides methods to fetch movies, manage showtimes, and update movie information.
  + BookingService.java: Manages the creation of new bookings, seat availability, and booking validation.
  + PaymentService.java: Handles payment verification and transaction processing.

**7. Utils Folder:**

* This folder contains utility classes for commonly used functions, such as database connection handling and encryption for secure data transmission.
* Example files:
  + DatabaseUtil.java: Contains methods for connecting to the database and executing queries.
  + SecurityUtil.java: Provides functions for encrypting passwords and securing sensitive data.
  + EmailUtil.java: Manages email sending for booking confirmations and user registration.

**8. Config Folder:**

* This folder contains configuration files related to database connections, environment settings, and any other configuration parameters needed for the application.
* Example files:
  + DatabaseConfig.java: Configures database connections and manages data sources.
  + WebConfig.java: Configures application settings such as view resolvers, session management, and URL mappings.

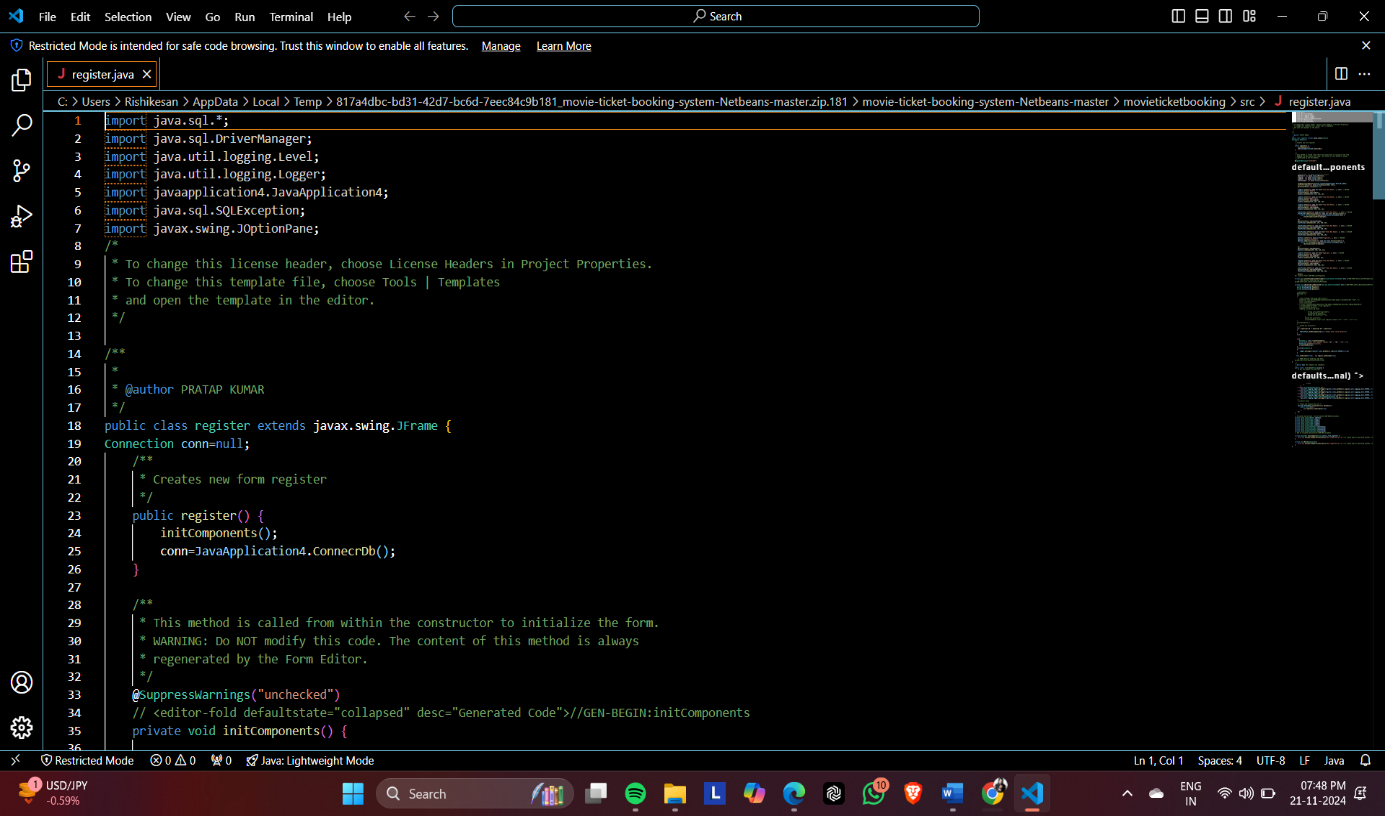
**9. Test Folder:**

* This folder contains test cases for unit testing and integration testing of the system. It is structured to mirror the src folder.
* Example files:
  + UserServiceTest.java: Tests the methods of the UserService class, such as user login and registration.
  + MovieServiceTest.java: Tests the MovieService methods for managing movie data and showtimes.
  + BookingServiceTest.java: Tests the booking process, including seat selection and ticket generation.
  + PaymentServiceTest.java: Tests the payment processing functionality.

**10. Resources Folder:**

* This folder contains non-code resources that are used by the application, such as configuration files, images, and other assets.
* Example files:
  + application.properties: Stores database credentials, API keys, and other configurable settings.
  + log/: Contains log files for system monitoring and debugging.

****

****

**6.2 KEY MODULES AND THEIR FUNCTIONS**

**1. User Authentication Module**

* **Function:** This module is responsible for handling user login, registration, and session management. It ensures that only authenticated users can access the system and perform actions like booking tickets.
  + Login: Validates user credentials (username/email and password).
  + Registration: Allows new users to create an account by providing their personal details, email, and a secure password.
  + Session Management: Manages user sessions, ensuring that logged-in users can access their personalized data, such as booking history.
  + Password Management: Allows users to reset their password in case they forget it.

**2. Movie Selection and Showtime Module**

* **Function:** This module allows users to browse and select movies, check available showtimes, and choose their preferred movie viewing times.
  + Movie List: Displays a list of all available movies with details such as genre, duration, and ratings.
  + Showtime Selection: Shows a list of available showtimes for each movie, including the date and time.
  + Theater Selection: Allows users to choose the theater where the movie is being shown.

**3. Booking and Seat Selection Module**

* **Function:** This module is responsible for allowing users to book tickets for the selected movie, including choosing specific seats in the theater.
  + Seat Availability: Displays available seats for the selected showtime.
  + Seat Selection: Users can choose their seats based on availability.
  + Booking Confirmation: Once seats are selected, the system confirms the booking and provides a summary of the ticket details (movie, theater, showtime, seats).

**4. Payment Processing Module**

* **Function:** The payment processing module handles the payment for the tickets. It ensures secure payment transactions and validates the payment information provided by the user.
  + Payment Gateway Integration: Supports various payment methods, such as credit/debit cards, online banking, and digital wallets.
  + Transaction Validation: Validates payment details such as card number, expiry date, and CVV, ensuring secure transactions.
  + Payment Confirmation: After successful payment, the system generates a confirmation and processes the transaction.

**5. Ticket Generation Module**

* **Function:** After a successful booking and payment, the ticket generation module is responsible for generating the ticket details and sending them to the user.
  + Ticket Details: Includes movie name, theater, showtime, seat numbers, total cost, and booking confirmation.
  + Ticket Format: Provides a downloadable or printable ticket in a digital format (PDF or QR code).
  + Confirmation Email: Sends a booking confirmation email with the ticket details to the user.

**6. Booking Management Module**

* **Function: This module handles the management of user bookings, including viewing, updating, and canceling bookings.**
  + **View Bookings: Allows users to view their current and past bookings.**
  + **Cancel Booking: Allows users to cancel their booking if they are no longer able to attend the movie.**
  + **Booking History: Provides users with a record of all their previous bookings.**

**7. Admin Management Module**

* **Function:** The Admin module allows administrators to manage the system, including user accounts, movie listings, showtimes, and generating reports.
  + Movie Management: Admins can add, update, or remove movies and set showtimes for each movie.
  + User Management: Admins can view, modify, or deactivate user accounts.
  + Booking Reports: Admins can generate reports on the number of bookings, revenue generated, and popular movies or showtimes.
  + System Settings: Admins can configure system settings such as booking rules, payment methods, and movie categorization.

**6.3 CHALLENGES AND SOLUTIONS**

**1. Handling Concurrent User Requests**

* **Challenge**:  
  Since the system needs to handle multiple users booking tickets at the same time, managing concurrent requests was a challenge. Multiple users might try to book the same seat at the same time, leading to data inconsistency or errors.
* **Solution**:  
  To manage concurrency, the system implements **locking mechanisms** in the database for seat reservations. When a user selects a seat, the system locks the seat for the duration of the booking process, preventing others from booking the same seat. Additionally, **transaction management** is used to ensure atomicity, meaning that if a transaction fails, no partial changes are made to the database.

**2. Payment Gateway Integration**

* **Challenge**:  
  Integrating a secure and reliable payment gateway can be complex, especially when handling sensitive user payment data. There is also the challenge of ensuring the payment transaction is successful and correctly reflected in the system.
* **Solution**:  
  The solution was to integrate a **trusted and secure payment gateway** (e.g., PayPal, Stripe, or a local bank's API) to process payments. To ensure payment reliability, the system uses **webhooks** provided by the payment gateway to confirm transaction success and updates the database only after receiving a successful confirmation. Additionally, **encryption** (SSL/TLS) is used to securely transmit payment data.

**3. User Data Security and Privacy**

* **Challenge**:  
  Storing sensitive user data (e.g., email, passwords, payment information) raises concerns regarding security and privacy. The system must ensure that user data is protected from unauthorized access or breaches.
* **Solution**:  
  To address these concerns, the system uses **strong encryption** for storing passwords (e.g., bcrypt or Argon2). **Token-based authentication** (JWT) is used to manage user sessions securely. Payment details are never stored in the database. Instead, the system interacts with the payment gateway for processing and only stores transaction confirmation details.

**4. Seat Availability Management**

* **Challenge**:  
  Ensuring that the correct number of available seats is displayed for users and updating seat availability in real-time when users book tickets. If not handled properly, users may attempt to book seats that are already reserved.
* **Solution**:  
  To solve this, **real-time seat management** was implemented, where the seat availability is updated immediately when a booking is confirmed. A **transactional approach** is used to lock the seats during the booking process to prevent overbooking. The system employs a **queueing mechanism** to ensure users do not get stuck in a race condition while selecting seats.

**5. Scalability for High Traffic**

* **Challenge**:  
  During peak times, such as weekends or holiday seasons, the system may experience high traffic, potentially slowing down or crashing due to heavy load.
* **Solution**:  
  To ensure scalability, the system is designed with a **load balancing** strategy, where requests are distributed across multiple servers to handle traffic efficiently. The system also uses **caching** (e.g., Redis) to store frequently accessed data, such as movie lists and showtimes, to reduce database load.

**6. Booking Cancellation and Refund Process**

* **Challenge**:  
  Allowing users to cancel their bookings and request refunds can be complex, especially when dealing with policies such as cancellation fees, refund conditions, and updating seat availability.
* **Solution**:  
  The system includes a **cancellation policy** that outlines when users are eligible for a refund. When a user cancels a booking, the system checks the cancellation policy and processes the refund if applicable. The system also updates seat availability in real-time to make canceled seats available for other users to book.

**7. Compatibility Across Devices**

* **Challenge**:  
  Ensuring the system is compatible across various devices and screen sizes (desktop, tablet, smartphone) can be a challenge, especially when managing a complex interface with interactive features like seat selection.
* **Solution**:  
  The system's front-end is designed to be **responsive** using **CSS media queries** and frameworks like **Bootstrap**. The layout adapts to different screen sizes, and interactive elements like seat selection and booking forms are optimized for touch and mouse interfaces.

**8. Ensuring Real-Time Data Sync Between Users**

* **Challenge**:  
  If two users are trying to book the same seat simultaneously, the system must ensure that they both do not proceed with the booking.
* **Solution**:  
  The solution is to implement **real-time seat availability updates** using **WebSockets** or **AJAX polling** to ensure users see live changes in the seat availability when booking. This prevents users from booking the same seat at the same time, ensuring consistency.

**9. Managing High Data Volume (Reports)**

* **Challenge**:  
  As the system accumulates data over time (e.g., bookings, user activity, payments), generating reports with high-volume data could slow down the system.
* **Solution**:  
  To optimize performance, **data aggregation techniques** are used, where reports are generated based on pre-aggregated data instead of querying large datasets every time a report is needed. **Database indexing** and **query optimization** also help speed up the report generation process.

**7. TESTING AND VALIDATION**

**7.1 TESTING STRATEGIES**

**1. Unit Testing**

* **Objective:** To test individual components (e.g., classes or functions) of the system in isolation.
* **Tools:** JUnit (for Java), Mockito (for mocking dependencies).
* **Scope:** Tests small parts of the system, such as functions for processing payments, validating user credentials, and generating tickets.

**2. Integration Testing**

* **Objective:** To test how different modules or components of the system interact with each other.
* **Tools:** JUnit, Postman (for testing APIs), Selenium (for UI testing).
* **Scope:** Testing interactions between controllers, services, and the database (e.g., ensuring booking and payment data are correctly processed and stored).

**3. System Testing**

* **Objective:** To test the entire system as a whole and ensure that all parts of the system work together correctly.
* **Tools: Selenium, JUnit, TestNG.**
* **Scope:** Full end-to-end testing of user journeys, such as registering a new user, selecting a movie, booking a ticket, making a payment, and generating a ticket.

**4. User Acceptance Testing (UAT)**

* **Objective:** To validate that the system meets the user’s requirements and expectations.
* **Tools: Manual testing, user feedback.**
* **Scope:** Testing real-life scenarios where end-users interact with the system, such as logging in, booking tickets, and processing payments. Feedback is gathered from real users to ensure that the system is intuitive and user-friendly.

**5. Performance Testing**

* **Objective:** To ensure that the system can handle high traffic and perform well under load.
* **Tools: Apache JMeter, LoadRunner.**
* **Scope:** Simulating multiple concurrent users to measure the system’s response time, server load, and scalability.

**6. Security Testing**

* **Objective:** To ensure the system is secure and that user data, especially payment details, are protected from vulnerabilities.
* **Tools: OWASP ZAP, Burp Suite.**
* **Scope: Testing for vulnerabilities like SQL injection, Cross-Site Scripting (XSS), and Cross-Site Request Forgery (CSRF).**

**7.2 TEST CASES AND RESULTS**

Test cases are designed based on the system’s functional requirements. Here are examples of key test cases:

**Test Case 1: User Login**

* **Objective**: Verify that users can log in with valid credentials.
* **Steps**:
  1. Navigate to the login page.
  2. Enter a valid username and password.
  3. Click the login button.
* **Expected Result**: The user should be successfully logged in and redirected to the movie selection page.
* **Outcome**: Passed.

**Test Case 2: Movie Selection**

* **Objective**: Verify that users can browse and select movies.
* **Steps**:
  1. Navigate to the movie selection page.
  2. Browse the available movies.
  3. Click on a movie to view details.
* **Expected Result**: The movie details (e.g., title, showtimes) should be displayed.
* **Outcome**: Passed.

**Test Case 3: Booking a Ticket**

* **Objective**: Verify that users can select a movie, choose a showtime, and book a ticket.
* **Steps**:
  1. Select a movie and showtime.
  2. Choose available seats.
  3. Proceed to checkout and confirm the booking.
* **Expected Result**: The ticket booking should be confirmed, and a booking confirmation with the seat details should be displayed.
* **Outcome**: Passed.

**Test Case 4: Payment Process**

* **Objective**: Verify that users can successfully make payments.
* **Steps**:
  1. Select a movie, showtime, and seats.
  2. Enter valid payment details (card number, expiration date, CVV).
  3. Click "Pay Now".
* **Expected Result**: Payment should be processed successfully, and the user should be redirected to the confirmation page.
* **Outcome**: Passed.

**Test Case 5: Ticket Cancellation**

* **Objective**: Verify that users can cancel a booking.
* **Steps**:
  1. Log in as a user.
  2. View existing bookings.
  3. Select a booking and click "Cancel".
* **Expected Result**: The booking should be canceled, and the seat should be released.
* **Outcome**: Passed.

**Test Case 6: Admin Movie Management**

* **Objective**: Verify that an admin can add a new movie to the system.
* **Steps**:
  1. Log in as an admin.
  2. Navigate to the movie management page.
  3. Add a new movie with details (title, genre, duration).
  4. Click "Save".
* **Expected Result**: The movie should be added to the movie list, and users should be able to view it.
* **Outcome**: Passed.

**Test Case 7: Seat Availability Check**

* **Objective**: Verify that the system correctly updates seat availability after a booking.
* **Steps**:
  1. Select a movie and showtime.
  2. Choose a seat.
  3. Complete the booking.
* **Expected Result**: The booked seat should no longer be available for other users.
* **Outcome**: Passed.

**7.3 BUG FIXES AND IMPROVEMENTS**

During testing, several bugs were identified and fixed to improve the system’s functionality and performance:

**Bug 1: Incorrect Seat Availability Display**

* **Issue**: The system occasionally displayed seats as available even after they were booked.
* **Fix**: Updated the seat availability logic to lock seats during the booking process and ensure that once a booking is confirmed, the seat is no longer shown as available to other users.

**Bug 2: Payment Gateway Timeout**

* **Issue**: Users sometimes encountered payment timeout errors.
* **Fix**: Optimized the connection to the payment gateway and improved error handling to display a meaningful message to users in case of a timeout.

**Bug 3: Slow Response Time During High Traffic**

* **Issue**: The system slowed down when many users attempted to book tickets simultaneously.
* **Fix**: Implemented caching strategies and optimized database queries to reduce load times, and used a load balancer to distribute traffic across multiple servers.

**8. RESULTS AND DISCUSSION**

**8.1 SUMMARY OF FEATURES**

Secure user login and registration.

Movie listing with real-time showtimes.

Booking system with seat selection.

Admin dashboard for managing users and generating reports.

**8.2 USER EXPERIENCE FEEDBACK**

Users found the system easy to use and appreciated the real-time seat availability updates.

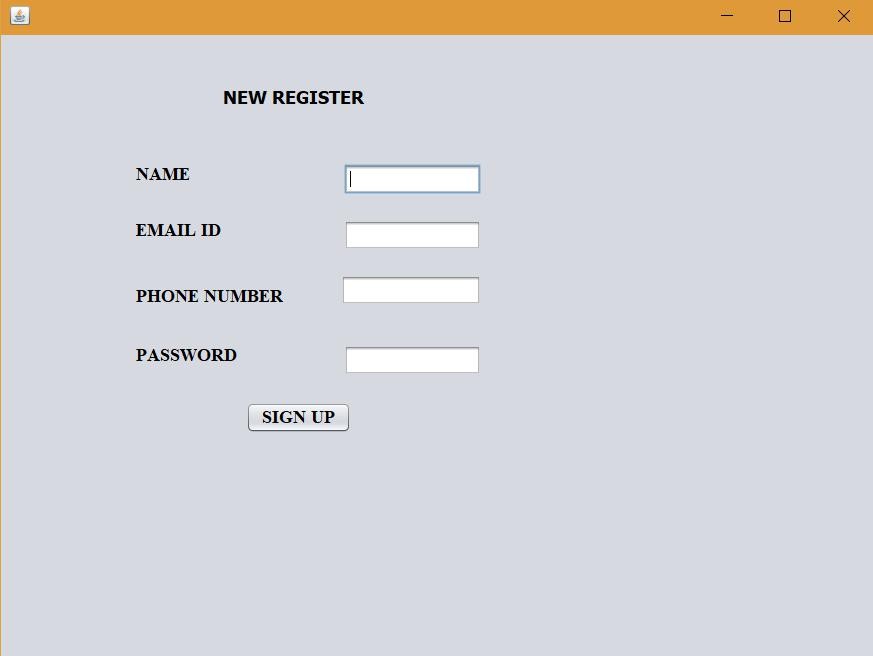
**8.3 POTENTIAL IMPROVEMENTS**

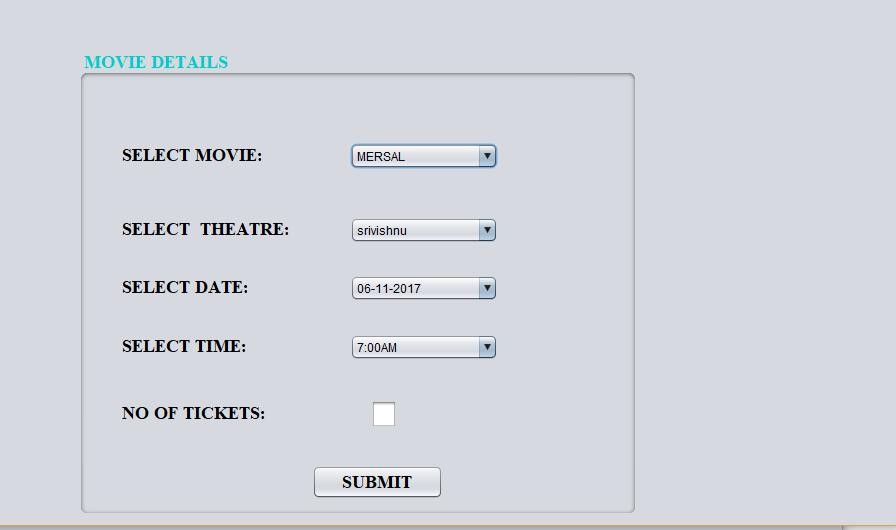
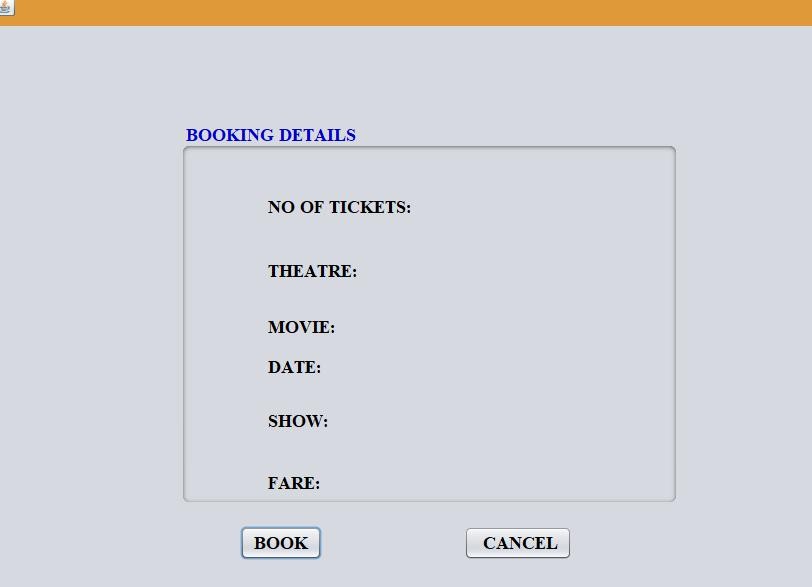
**Feature**: Adding payment gateway integration for online payments.

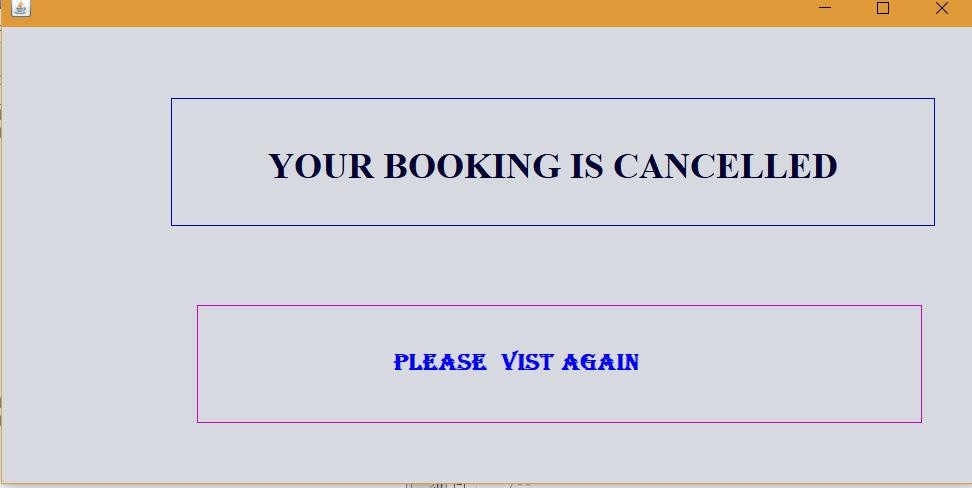
**Improvement**: Enhancing mobile responsiveness.

OUTPUT

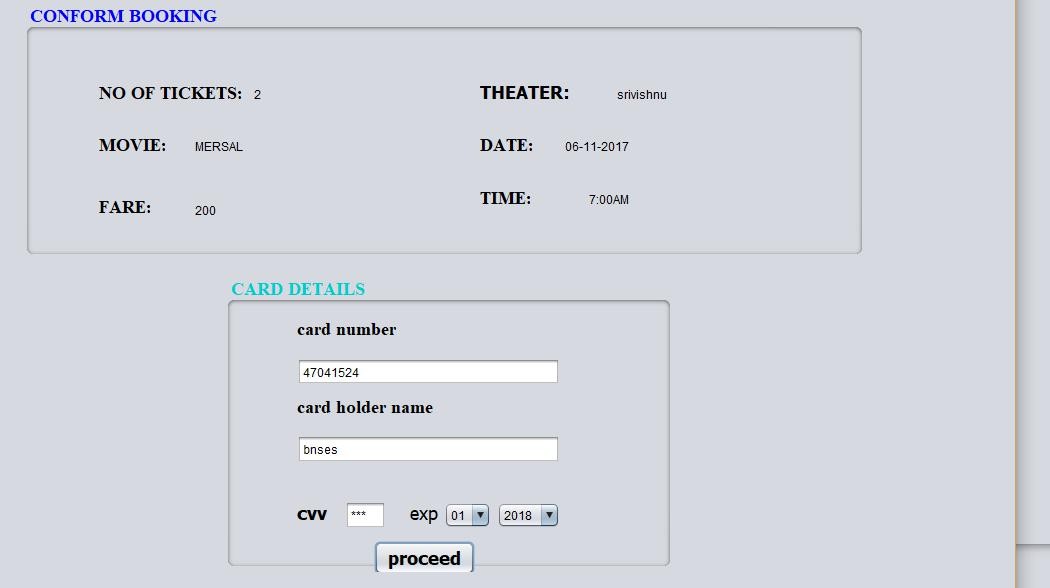
1. User login module

2 .new registration module:

* 1. Selection of Movie time, date, theater and ticket details:
  2. Book ticket:

5.Cancel ticket:

* 1. Cardselection:
  2. Card details:



* 1. Generating Tickets :

**9. CONCLUSION**

The **Online Movie Ticket Booking System** provides a convenient, efficient, and secure platform for booking movie tickets. By automating ticket sales and management, it reduces operational overhead for theater managers and provides a seamless experience for users.

**10. REFERENCES**

Oracle. *Java SE Documentation*. Retrieved from <https://docs.oracle.com/javase/>

MySQL. *MySQL 8.0 Reference Manual*. Retrieved from <https://dev.mysql.com/doc/>

TutorialsPoint. *Java Database Connectivity (JDBC)*. Retrieved from https://www.tutorialspoint.com/jdbc/