MOVIE TICKET BOOKING RESERVATION SYSTEM

CS2333 – Object Oriented Programming Using JAVA Project Report

Submitted by

ARUNACHALAM A - 231001019

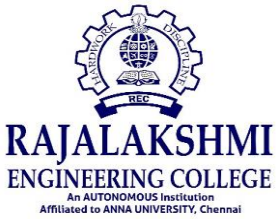
ARUN V - 231001018

Of

BACHELOR OF TECHNOLOGY

In

INFORMATION TECHNOLOGY



DEPARTMENT OF INFORMATION TECHNOLOGY

RAJALAKSHMI ENGINEERING COLLEGE

NOVEMBER-2024

BONAFIDE CERTIFICATE

Certified that this project titled "Movie Reservation System" is the bonafide work of **ARUNACHALAM (231001019)**, **ARUN (231001018)** who carried out the project work under my supervision

SIGNATURE
Dr.P.Valarmathie
HEAD OF THE DEPARTMENT

Department of Information Technology
Rajalakshmi Engineering College

SIGNATURE
Mrs.Usha S
COURSE INCHARGE
Assistant Professor(S.G)

Department of Information Technology
Rajalakshmi Engineering College

This project is submitted for CS23333 – Object Oriented Programming Using JAVA held on _____

INTERNAL EXAMINER **EXTERNAL EXAMINER**

Table of Contents:

CHAPTER NO.	TITLE	PAGE NO.
1	**1.1 ABSTRACT**	**6**
	1.2 INTRODUCTION	**6**
	1.3 PURPOSE	**6**
	1.4 SCOPE OF PROJECT	**7**
	1.5 SOFTWARE REQUIREMENT SPECIFICATION	**7**
2	**SYSTEM FLOW DIAGRAM**	**13**
	2.1 USE CASE DIAGRAM	**13**
	2.ENTITY RELATIONSHIP DIAGRAM	**14**
	2.3 DATA FLOW DIAGRAM	**15**

3	**MODULE DESCRIPTION**	**16**
4	**DESIGN AND IMPLEMENTATION**	**17**
	4.1 DESIGN	**17**
	4.2 DATABASE DESIGN	**21**
	4.3 IMPLEMENTATIONS	**22**
5	**CONCLUSION**	**28**
6	**REFERENCE**	**28**

LIST OF FIGURES

Figure Numbers	Figure Captions	Pg.no
2.1.1	Use Case Diagram	13
2.2.1	Entity Relationship Diagram	14
2.3.1	Data Flow Diagram	15
4.1.1	Login Page	17
4.1.2	Home Page	17
4.1.3	Movie Booking Steps	18
4.1.4	Details Of Booked Movie	18
4.1.5	Details Of Movie Booked	19
4.1.6	Payment	19
4.1.7	Reservation Status	20

LIST OF TABLES

Table Number	Table Caption	Pg.no
4.2.1	Movie Booking Table	22

1.1Abstract:

The primary objective of the JDBC-powered Movie Ticket Booking Reservation System in Java is to provide a reliable and efficient platform for users to seamlessly browse, select seats, and reserve movie tickets, all while leveraging the power of Java's database connectivity capabilities.

1.2 Introduction:

The Movie Ticket Booking Reservation System is an advanced software application designed to revolutionize the traditional manual methods of booking and managing movie tickets. This state-of-the-art system introduces automation to enhance efficiency, significantly reducing time and operational costs. Secured by a robust authentication process, it ensures that only authorized users can access and manage key features such as movie schedules, seat assignments, and other critical functions. By leveraging technology, this system provides a streamlined, user-friendly approach to movie reservations, offering a modern and efficient alternative to conventional practices.

1.3 Purpose:

The goal of this project is to develop a reliable and user-friendly movie ticket reservation system that serves both cinema-goers and administrators. The system is designed to:

- Streamline the process of booking movie tickets.
- Enhance the movie-going experience by providing users with comprehensive movie details and simplified booking options.
- Empower cinema administrators to efficiently manage movie schedules, reservations, and showtimes.
- Store and analyze data to provide valuable business insights and support informed decision-making.

1.4 Scope of the Project:

The proposed Movie Ticket Booking Reservation System is designed to efficiently engage with administrators and deliver all intended functionalities. Developed using Java (JDBC) and powered by a MySQL database, the system ensures seamless management of movie details while minimizing response times for user inquiries. By addressing the challenges of manual reservation processes, it provides a comprehensive solution for storing and managing information about movies, showtimes, and seat availability. With a focus on verification, validation, security, and user-friendliness, this system aims to enhance the movie reservation experience and support a full range of cinema management activities.

1.5 Software Requirement Specification:

Introduction:

The Movie Reservation System is a streamlined solution for handling all elements of movie ticket booking, including managing movie details, show schedules, and seat assignments. It replaces traditional manual processes with automation, boosting efficiency and simplifying cinema management.

Document Purpose:

This Software Requirements Specification (SRS) document provides a detailed outline of the Movie Reservation System's requirements. It includes design considerations, architectural structure, and technical specifications, ensuring smooth implementation and continued software maintenance.

Product Scope:

The Movie Reservation System is developed to modernize and replace legacy paper-based booking methods. It enhances efficiency by offering a unified platform for managing cinema

operations. The system is highly adaptable, allowing modifications to movie information and ensuring an optimized, user-friendly booking process.

MRS - Movie Reservation System
SRS - Software Requirements Specification.

References and Acknowledgement:

[1] https://www.javatpoint.com/java-awt
[2] https://www.javatpoint.com/java-swing
[3] https://www.geeksforgeeks.org/introduction-to-jdbc/

The Movie Reservation System enables authorized personnel to efficiently manage movie ticket records, offering a streamlined reservation process for cinemas globally. This system simplifies daily operations across various entertainment venues.

Product Perspective:

Built on a client/server model, the system is optimized for the Microsoft Windows operating system. The front end is developed using Java AWT and Swing, while MySQL server powers the backend for robust and efficient data management.

Product Functionality:

- Admin Register: Allows new administrators to register on the platform.
- Admin Login: Provides secure login functionality for existing administrators.
- Add Movie: Enables administrators to add details of new movies.
- View Movie: Facilitates viewing and updating of existing movie information.
- Delete Movie: Permits removal of movies from the system database.
- Add Reservation: Supports the creation of new user reservations.
- Update Reservation: Allows users to view and modify their existing reservations.

- Remove Admin: Enables deletion of specific administrator accounts.

User and Characteristics:

Qualification: Users should have at least basic educational qualifications, such as matriculation, and be comfortable with English.
Experience: Familiarity with the university registration process is advantageous. Technical Experience: Users are expected to have elementary knowledge of computers for optimal system interaction.

Operating Environment:

Hardware Requirements:

- Processor: Any Processor over i3
- Operating System: Windows 8, 10, 11
- Processor Speed: 2.0 GHz
- RAM: 4GB
- Hard Disk: 500GB

Software Requirements:

- Database: MySQL
- Frontend: Java (SWING, AWT)
- Technology: Java (JDBC)

Constraints:

- System access limited to administrators.
- Delete operation restricted to administrators without additional checks for simplicity.
- Administrators must exercise caution during deletion to maintain data consistency.

Assumptions and Dependencies:

- System administrators create and confidentially communicate login IDs and passwords to users. Specific Requirements:

User Interface:

The Movie Reservation System provides user-friendly, simplified interfaces for:

a) Admin Register: Registering new administrators.

b) Admin Login: Logging in existing administrators.

c) View Movie: Viewing and updating existing movie information.

d) Delete Movie: Deleting existing movies.

e) Add Reservation: Creating new reservations for users.

f) Update Reservation: Viewing and modifying existing reservations.

g) Remove Admin: Deleting specific administrator accounts.

Hardware Interface:

- Screen resolution of at least 640 x 480 or above.
- Compatible with any version of Windows 8, 10, 11.

Software Interface:

a) MS-Windows Operating System

b) Java AWT and SWING for designing the front end

c) MySQL for the backend

d) Platform: Java Language

e) Integrated Development Environment (IDE): Netbeans

Functional Requirements:

I. Log in Module (LM):

- Users (admins) access the Login Module.
- LM supports user login with a username and password.
- Passwords are masked for security.
- Successful login verification by the database administrator is required for access.

II. Registered Users Module (RUM):

- After successful login, users (admins) can navigate through the application. - Users can view detailed information about movies, showtimes, and reservations.
- Users can update and maintain movie details, including modifying showtimes and seat allocations.

III. Administrator Module (AM)

- Upon successful login, the system displays administrative functions.
- Functions include adding and updating movie details.
- The "Add" function allows administrators to input new movie details and remove unused entries.
- The "Update" function enables administrators to modify existing movie details in the database.
- All add, update, or delete requests trigger the AM module to communicate with the Server Module (SM) for necessary database changes.

IV. Server Module (SM):

- SM acts as an intermediary between various modules and the database (DB).
- Receives requests from different modules and formats pages for display.
- Validates and executes requests received from other modules.
- Handles communication with the database, ensuring data consistency and integrity, especially regarding movie details, showtimes, and reservations.

Non-functional Requirements:

Performance:

The system must handle real-time reservation requests efficiently, ensuring a response time of less than 2 seconds for seat selection and confirmation. - Safety-critical failures, such as payment processing errors, must be addressed instantly to ensure a smooth user experience.

Reliability:

The system is safety-critical; in case of abnormal operation or downtime, immediate measures should be taken to resolve the issue and restore normal functionality.

Availability:

Under normal operating conditions, user requests for movie reservations, including seat selection and payment, should be processed within 2 seconds to maintain a seamless booking experience.

Immediate feedback on reservation status and confirmation should be communicated to users to enhance the overall booking process.

Security:

A robust security mechanism must be in place on the server side to prevent unauthorized access, safeguard user payment information, and ensure the integrity of the reservation system.

User privacy, including personal details, must be securely stored and managed to maintain confidentiality.

Maintainability:

Design documents outlining software and database maintenance procedures must be available to facilitate regular updates and modifications to the movie reservation system.

Administrative access should be provided for proper maintenance at both the front end and back end, ensuring the system's long-term functionality and adaptability.

2.System Flow Diagrams:

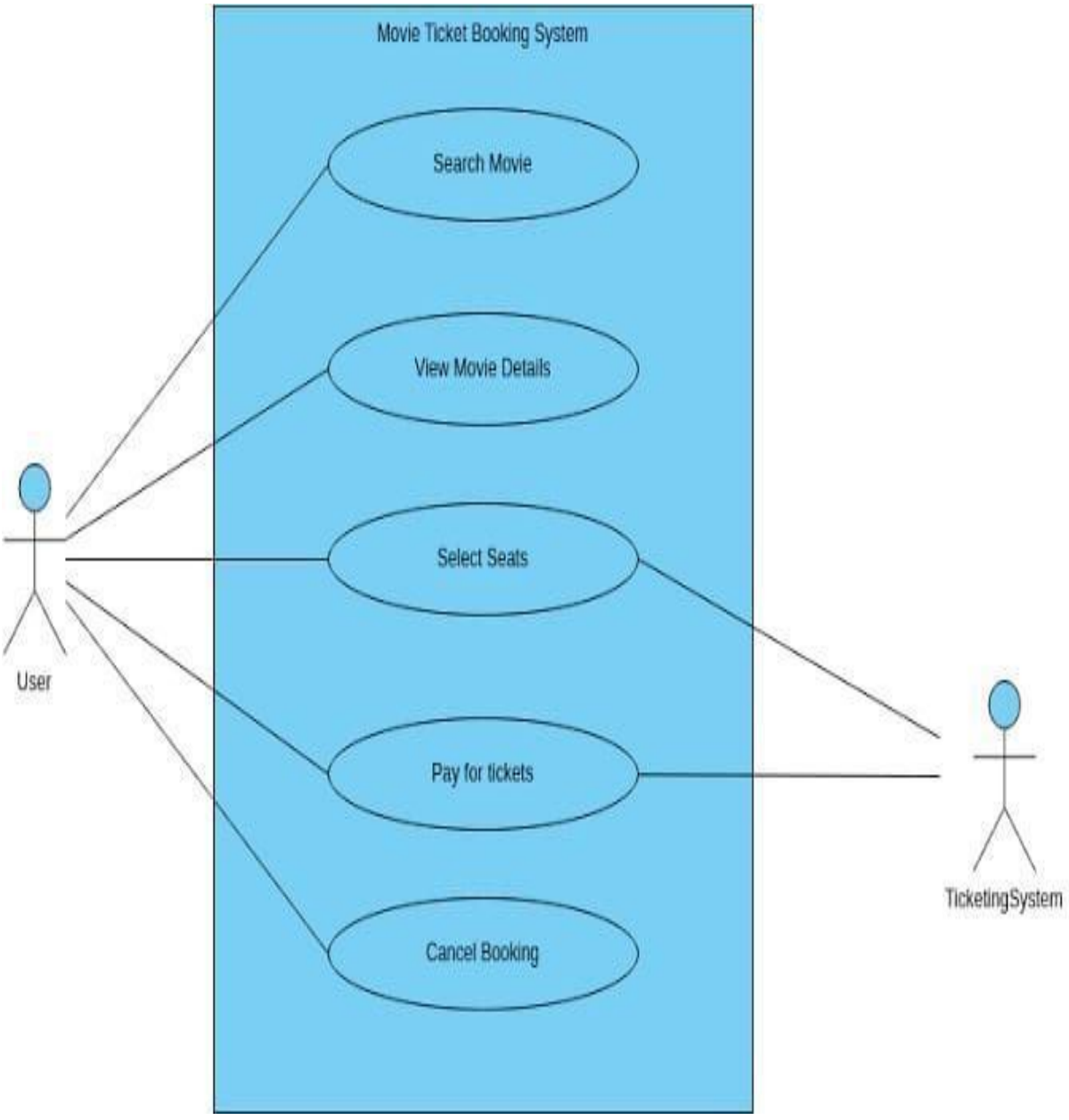


Figure 2.1.1 Use Case Diagram

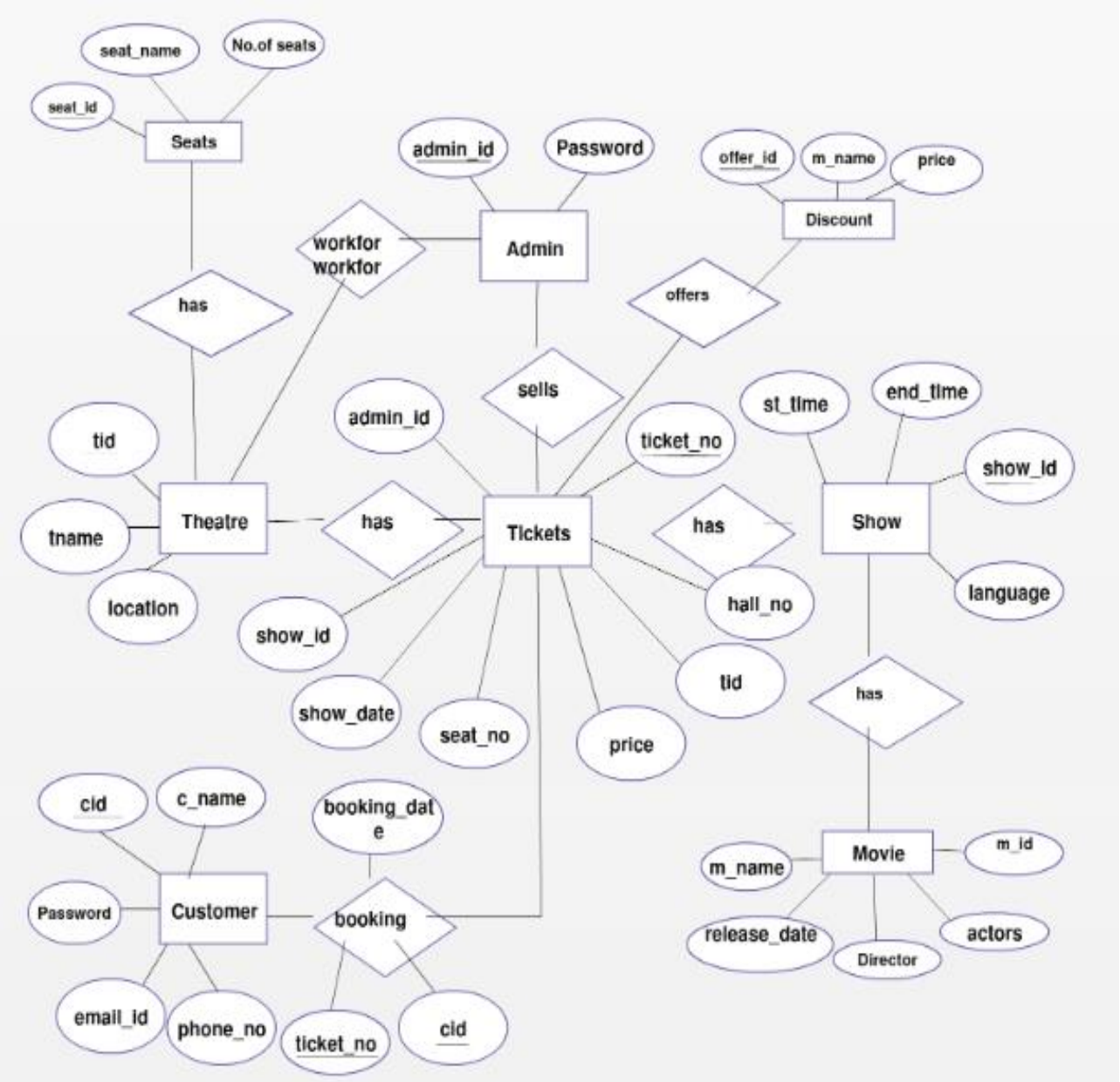


Figure 2.2.1 Entity Relationship Diagram

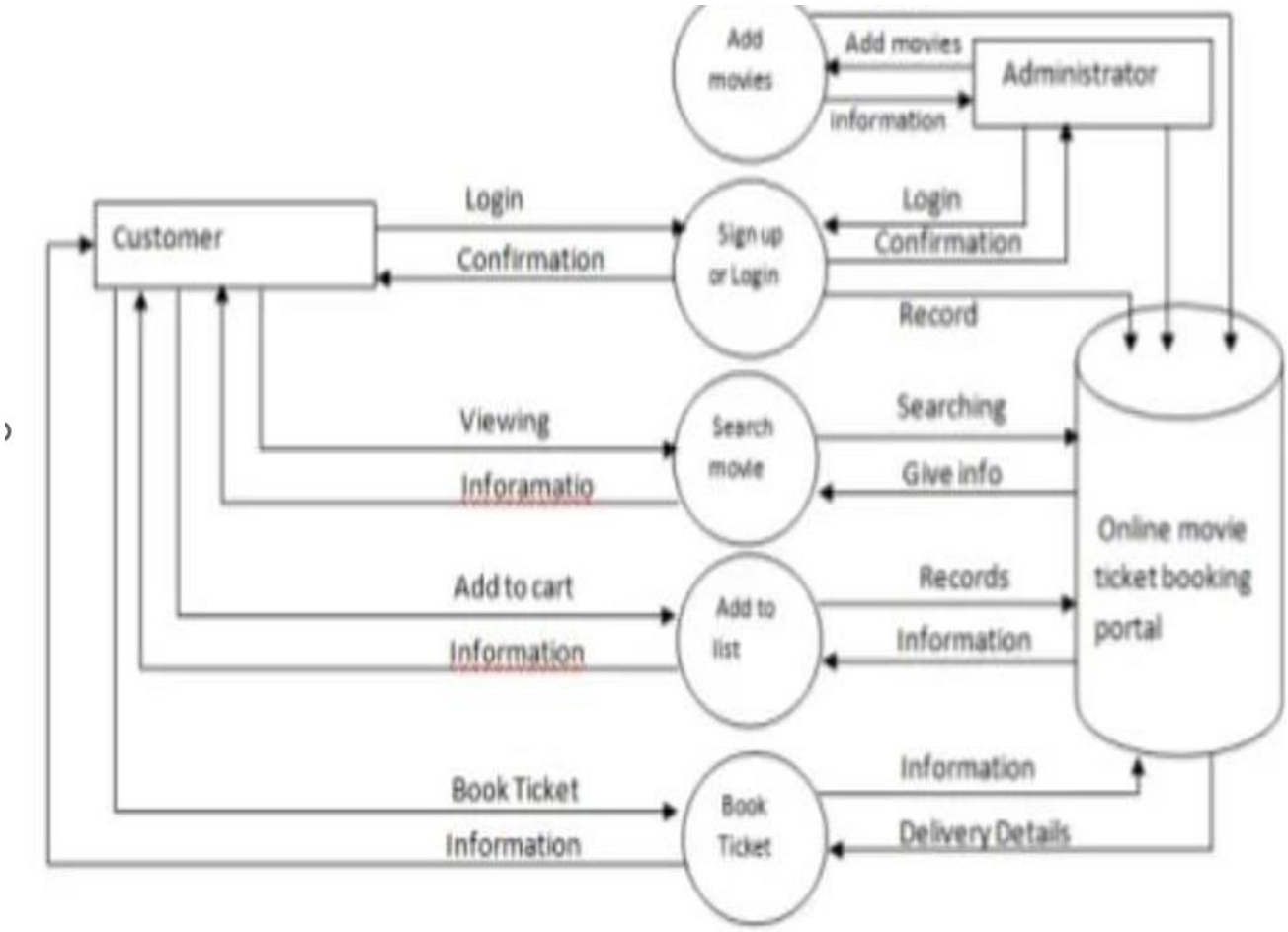


Figure 2.3.1 Data Flow Diagram

3. Module description:

1. Register:

   Admin can register with the username and password for the registration

2. Login:

   Admin can log in with their username and password.

3. After Login:

   - Add Movie:

     In this section, admin can add details about new movies, including the title, genre release date, and other relevant information.

   - View Movie:

     Admin can view and update movie details such as the title, no of seats and release date.

   - Delete Movie:

     Admin can delete movie details, removing movies from the system.

   - Add Reservation:

     Admin can add reservation details, including the movie, date, time, and number of seats.

   - Update Reservation:

     Admin can update reservation details, such as changing the date, time, or the number of reserved seats.

4. Design And Implementation :

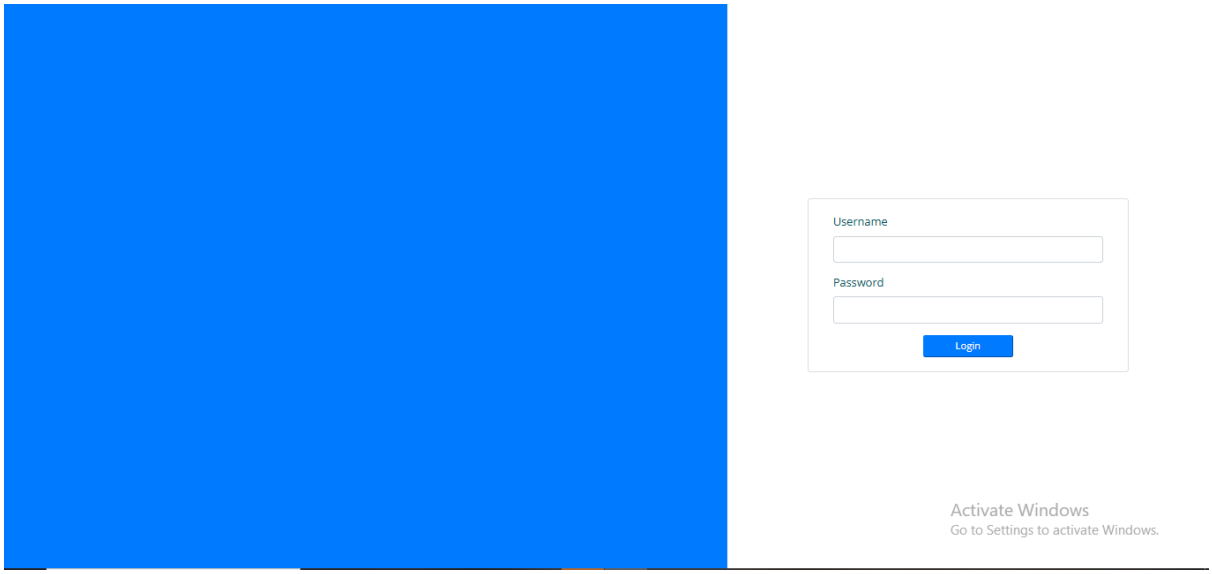


Figure 4.1.1 Login Page

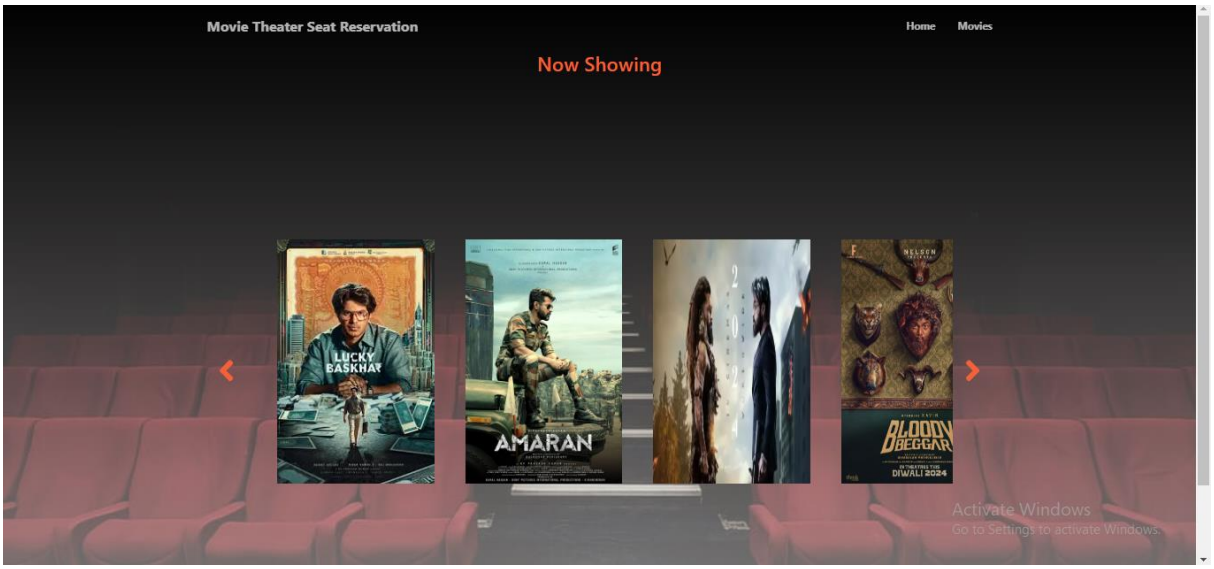


Figure 4.1.2 Home Page

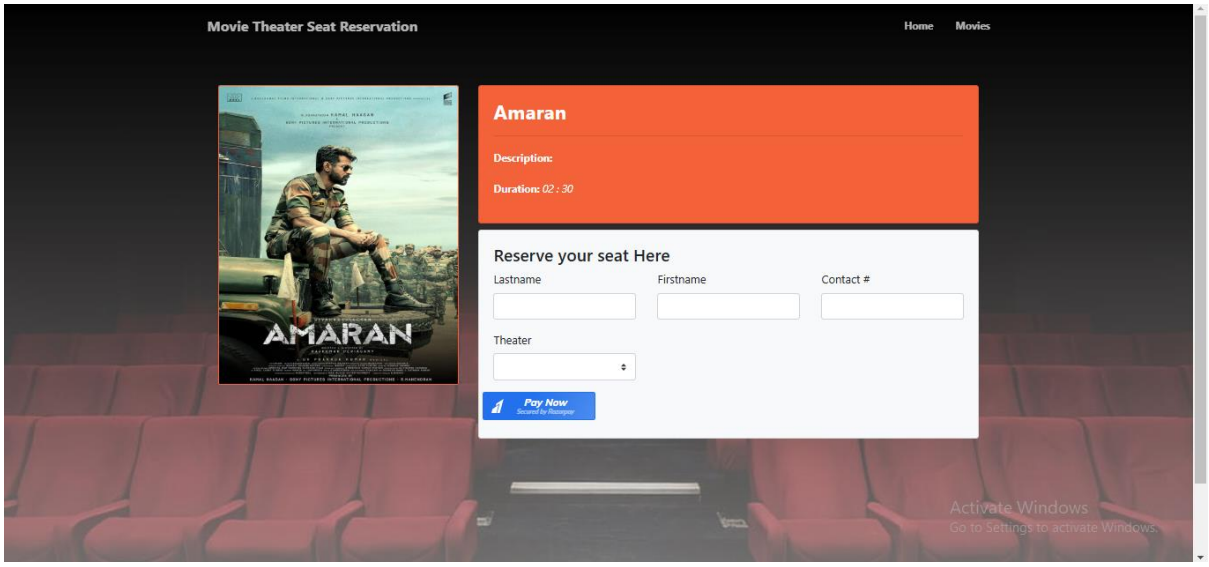


Figure 4.1.3 Movie Booking Steps

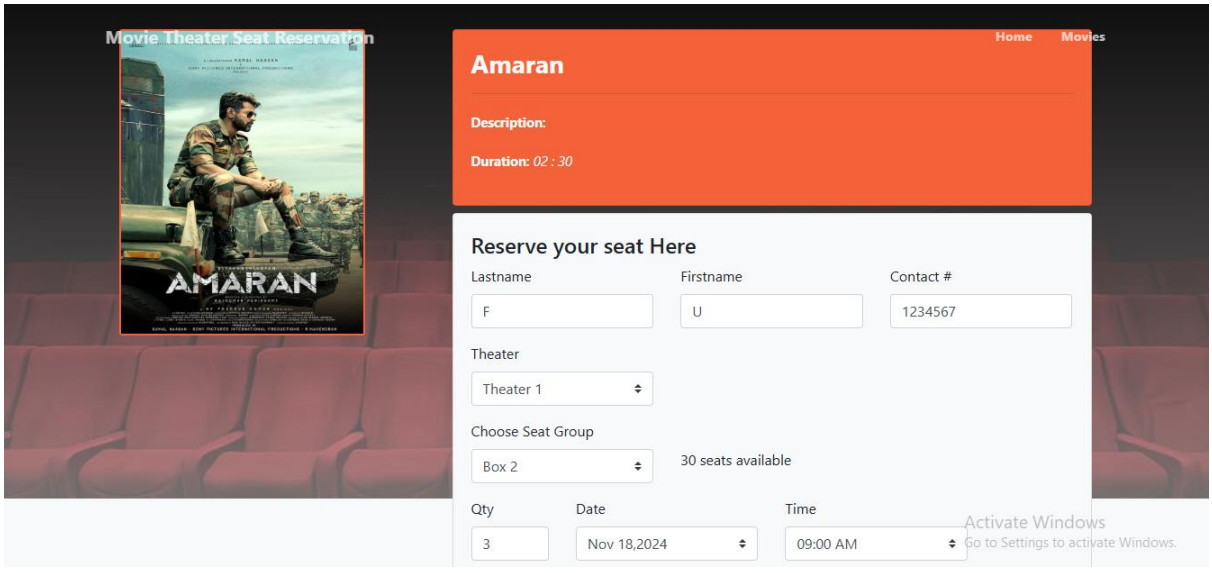


Figure 4.1.4 Details of booked Movie

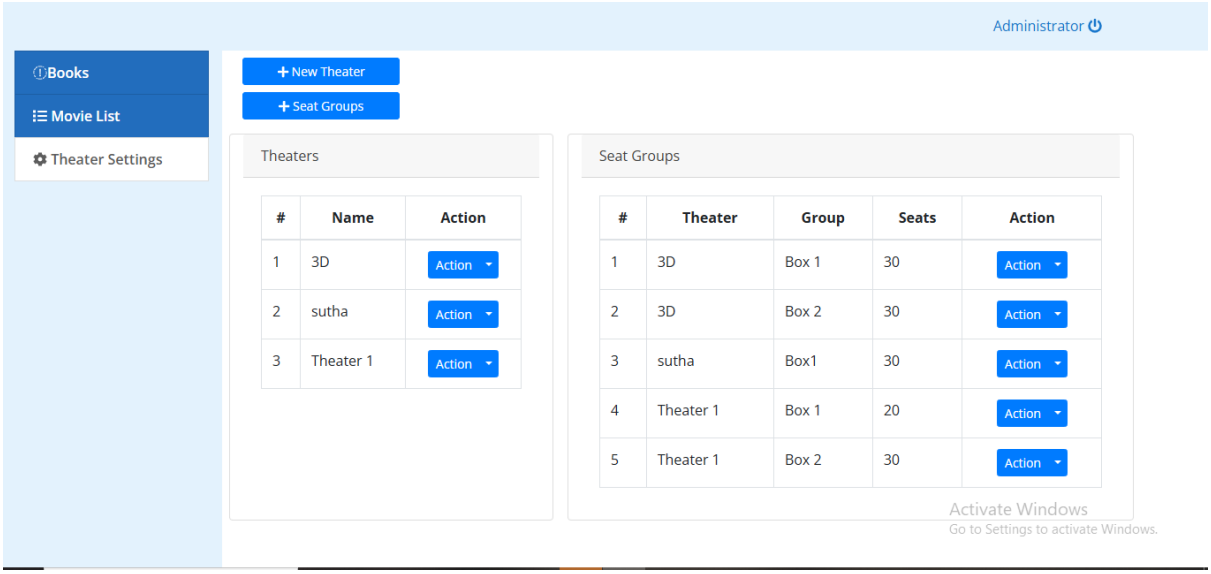


Figure 4.1.5 Details of Movie Booked

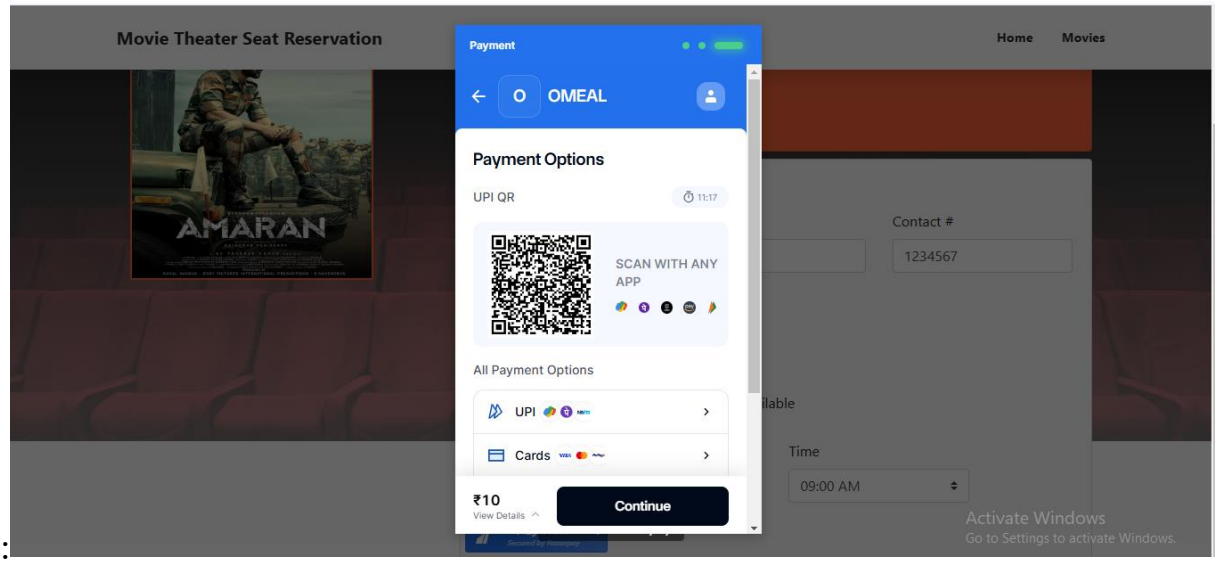


Figure 4.1.6 Payment

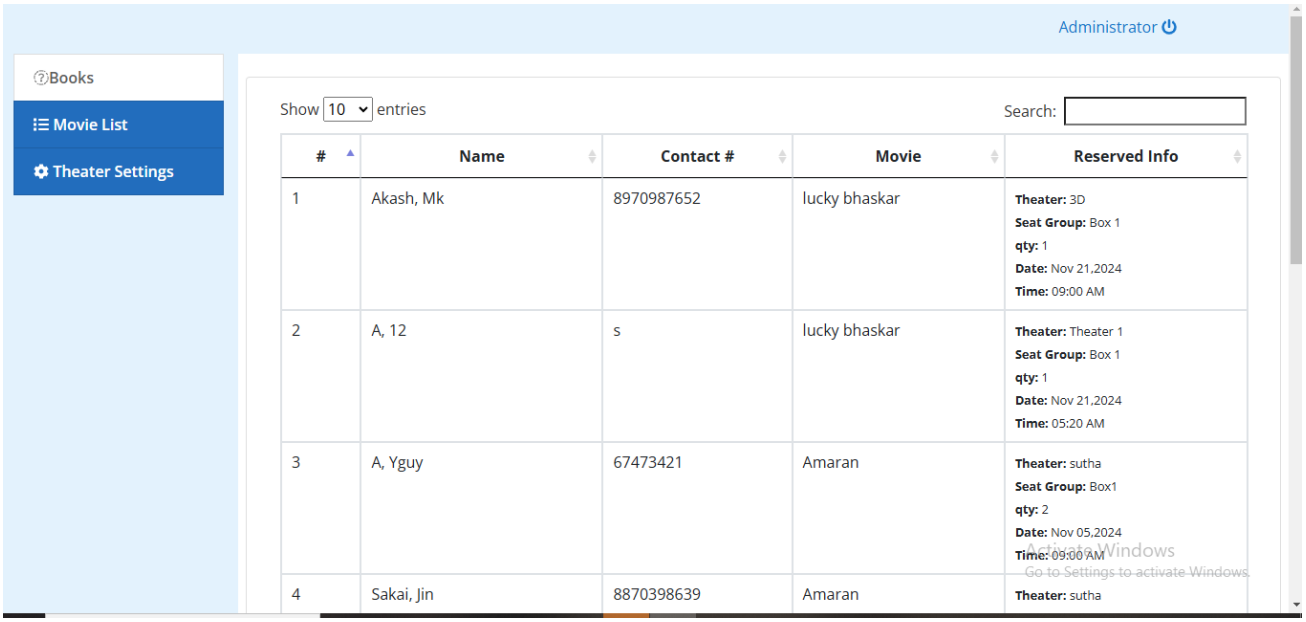


Figure 4.1.7 Reservation Status

4.2 Database Design:

The database design is a crucial part of the system design process, where the data elements and structures identified during the analysis stage are organized for efficient storage and retrieval. A database is a collection of related data, stored with minimal redundancy, to serve multiple users quickly and effectively. The primary goal is to ensure that database access is easy, fast, cost-effective, and flexible. Data relationships are established, and unnecessary items are removed. Normalization is performed to achieve internal consistency, minimize redundancy, and ensure maximum stability. This process helps to reduce the storage requirements, avoid data inconsistencies, and optimize updates. MySQL has been selected as the database management system for developing the necessary databases.

MySQL is a widely-used, open-source relational database management system known for its reliability, scalability, and performance. It is the go-to choice for many web applications, particularly those using the LAMP (Linux, Apache, MySQL, PHP) stack. MySQL enables developers to manage large datasets efficiently, offering features like fast query processing and strong data security. Its structured query language (SQL) makes data retrieval and manipulation intuitive. MySQL's cross-platform compatibility and seamless integration with various programming languages and frameworks make it ideal for diverse applications, from small-scale websites to enterprise-level databases.

MySQL offers several advantages over other databases like PostgreSQL and Oracle. It is lightweight, making it faster and more efficient for web- based applications. MySQL's ease of use and quick setup process make it accessible for beginners, while its scalability supports large-scale systems. Compared to Oracle, MySQL is more cost-effective, particularly for startups and smaller enterprises. Additionally, MySQL's broad community support provides extensive resources and documentation. Its performance is particularly optimized for read-heavy workloads, which is ideal for applications requiring fast data retrieval, making it a preferred choice for many businesses.

	Table	Action	Rows	Type	Collation	Size
☐	books	Browse Structure Search Insert Empty Drop	15	InnoDB	utf8mb4_general_ci	16.0 KiB
☐	movies	Browse Structure Search Insert Empty Drop	4	InnoDB	utf8mb4_general_ci	16.0 KiB
☐	theater	Browse Structure Search Insert Empty Drop	3	InnoDB	utf8mb4_general_ci	16.0 KiB
☐	theater_settings	Browse Structure Search Insert Empty Drop	6	InnoDB	utf8mb4_general_ci	16.0 KiB
☐	users	Browse Structure Search Insert Empty Drop	1	InnoDB	utf8mb4_general_ci	16.0 KiB
	5 tables	Sum	29	InnoDB	utf8mb4_general_ci	80.0 KiB

Table 4.2.1 Movie Booking Table

4.3 IMPLEMENTATIONS (CODE):

```
import javax.swing.*;

import java.awt.*;

import java.awt.event.*;

import java.sql.*;


public class MovieReservationSystem {

    private JFrame frame;

    private JComboBox<String> movieDropdown;

    private JTextField customerNameField;

    private JTextField seatNumberField;

    private JButton reserveButton;

    private JTextArea reservationStatus;


    // Database credentials

    private static final String DB_URL = "jdbc:mysql://localhost:3306/theater_db";

    private static final String USER = "root";

    private static final String PASSWORD = "password";


    public MovieReservationSystem() {

        // Setup the GUI

        frame = new JFrame("Movie Reservation System");

        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
```

```
frame.setSize(500, 400);

JPanel panel = new JPanel();
panel.setLayout(new GridLayout(6, 2, 10, 10));

panel.add(new JLabel("Select Movie:"));
movieDropdown = new JComboBox<>();
loadMovies(); // Populate dropdown with movies from the database
panel.add(movieDropdown);

panel.add(new JLabel("Customer Name:"));
customerNameField = new JTextField();
panel.add(customerNameField);

panel.add(new JLabel("Seat Number:"));
seatNumberField = new JTextField();
panel.add(seatNumberField);

reserveButton = new JButton("Reserve Seat");
reserveButton.addActionListener(e -> reserveSeat());
panel.add(reserveButton);

reservationStatus = new JTextArea();
reservationStatus.setEditable(false);
frame.add(panel, BorderLayout.CENTER);
frame.add(new JScrollPane(reservationStatus), BorderLayout.SOUTH);
```

```
    frame.setVisible(true);
}

private void loadMovies() {
    try (Connection conn = DriverManager.getConnection(DB_URL, USER, PASSWORD);
         Statement stmt = conn.createStatement();
         ResultSet rs = stmt.executeQuery("SELECT title FROM movies")) {

        while (rs.next()) {
            movieDropdown.addItem(rs.getString("title"));
        }
    } catch (SQLException e) {
        JOptionPane.showMessageDialog(frame, "Error loading movies: " + e.getMessage());
    }
}

private void reserveSeat() {
    String selectedMovie = (String) movieDropdown.getSelectedItem();
    String customerName = customerNameField.getText();
    String seatNumber = seatNumberField.getText();

    if (selectedMovie == null || customerName.isEmpty() || seatNumber.isEmpty()) {
        reservationStatus.setText("Please fill out all fields.");
        return;
    }

    try (Connection conn = DriverManager.getConnection(DB_URL, USER, PASSWORD)) {
```

```
            // Get the movie ID
            PreparedStatement getMovieIdStmt = conn.prepareStatement("SELECT id FROM movies WHERE title = ?");
            getMovieIdStmt.setString(1, selectedMovie);
            ResultSet rs = getMovieIdStmt.executeQuery();

            if (rs.next()) {
                int movieId = rs.getInt("id");

                // Insert reservation
                PreparedStatement reserveStmt = conn.prepareStatement(
                        "INSERT INTO reservations (movie_id, seat_number, customer_name) VALUES (?, ?, ?)");
                reserveStmt.setInt(1, movieId);
                reserveStmt.setString(2, seatNumber);
                reserveStmt.setString(3, customerName);
                reserveStmt.executeUpdate();

                reservationStatus.setText("Reservation successful for " + customerName + "!");
            } else {
                reservationStatus.setText("Movie not found.");
            }
        } catch (SQLException e) {
            reservationStatus.setText("Error: " + e.getMessage());
        }
    }
```

```
    public static void main(String[] args) {
        // Load JDBC driver
        try {
            Class.forName("com.mysql.cj.jdbc.Driver");
        } catch (ClassNotFoundException e) {
            JOptionPane.showMessageDialog(null, "JDBC Driver not found: " + e.getMessage());
            return;
        }

        SwingUtilities.invokeLater(MovieReservationSystem::new);
    }
}
```

5.Conclusion:

The Movie Ticket Booking Reservation System project, developed under expert supervision, reflects a thorough and systematic design and implementation process. Prioritizing user-centric features such as adding, viewing movies, and managing reservations, the system ensures an effortless and efficient experience for users. Enhanced security protocols, especially in the "Remove Admin" module, emphasize the system's dedication to safeguarding data and ensuring operational integrity. This project is a comprehensive solution tailored to meet current demands while remaining adaptable for future upgrades and enhancements.

6.Reference links:

[1] https://www.javatpoint.com/java-awt
[2] https://www.javatpoint.com/java-swing
[3] https://www.geeksforgeeks.org/introduction-to-jdbc/