MOVIE TICKET RESERVATION SYSTEM

CS23333 – Object Oriented Programming Using Java

Submitted by

ASMITHA SREE J A – 231001020

HARINI K - 231001054

Of

BACHELOR OF TECHNOLOGY

In

INFORMATION TECHNOLOGY

DEPARTMENT OF INFORMATION TECHNOLOGY

RAJALAKSHMI ENGINEERING COLLEGE

(AnAutonomousInstitution)

NOVEMBER 2024

RAJALAKSHMI ENGINEERING COLLEGE

BONAFIDE CERTIFICATE

Certified that this project titled **“MOVIE TICKET RESERVATION SYSTEM”** is the bonafide work of “**ASMITHA SREE J A (231001020), HARINI K (231001054)**” who carried out the project work under mysupervision. Certified further that to the best of my knowledge the work reported herein does not form part of any other thesis of dissertation on the basis of which a degree or award was conferred on an earlier occasion on this or any other candidate.

SIGNATURE	**SIGNATURE**
Dr.P.Valarmathie	**Mrs.Usha S**
HEAD OF THE DEPARTMENT	**COURSE INCHARGE**
Information Technology	Information Technology
Rajalakshmi Engineering College,	Rajalakshmi Engineering College

Submitted to Project Viva – Voce Examination held on . . . .

INTERNAL EXAMINAR EXTERNAL EXAMINAR

ABSTRACT

The primary objective of the JDBC-powered Movie 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. This system ensures data integrity and consistency through JDBC transactions, establishing a secure connection with a relational database for real-time updates on seat availability and reservation status. Theuser-friendly interface and scalability make it an effective solution for optimizing the movie reservation process in various cinema environment This system also benefits theater operators by providing tools for efficient management of schedules, bookings, and revenue tracking. The platform supports features like dynamic pricing, promotional offers, and customer engagement through notifications and reminders. Its robust backend ensures scalability to handle high traffic during peak times, while user-friendly policies for ticket cancellations and refunds ensure customer satisfaction. Overall, the Movie Reservation System simplifies the movie-going experience, making it more accessible and enjoyable for all stakeholders.

TABLE OF CONTENTS

CHAPTER NO	TITLE	PAGE NO
	ABSTRACT	i
	LIST OF FIGURES	iii
	LIST OF TABLES	iv
1	**INTRODUCTION**	1
	1.1 PROBLEM STATEMENT	1
	1.2 OBJECTIVEOFTHE PROJECT	2
	1.3 ORGANIZATION OF THE REPORT	2
2	**SYSTEM DESIGN**	3
	2.1 OVERVIEW	3
	2.2 LIST OF MODULES	3
	2.3 UML MODELING	4
	2.3.1 USE CASE DIAGRAM OVERVIEW	4
	2.3.2 ENTITY-RELATIONSHIP DIAGRAM	6
	2.3.3 DATA FLOW DIAGRAM	7
	2.4 SYSTEM SPECIFICATION	9
	2.4.1 FUNCTIONAL REQUIREMENTS	10
3	**DESIGN**	11
	3.1 DESIGN	11
	3.2 DATABASE DESIGN	16
	3.3 IMPLEMENTATION(CODE)	18
4	**CONCLUSION**	23
	REFERENCES	24

LIST OF FIGURES

FIGURE NO	FIGURE NAME	PAGE NUMBER
2.1	USE CASE DIAGRAM OF MOVIE RESERVATION SYSTEM	5
2.2	ENTITY RELATIONSHIP DIAGRAM OF MOVIE RESERVATION	6
2.3	DATA FLOW DIAGRAM OF MOVIE RESERVATION SYSTEM	8
3.1	LOGIN PAGE	11
3.2	SIGN UP PAGE	11
3.3	BOOKINGS PAGE	12
3.4	PAYMENT DETAILS PAGE	12
3.5	RECEIPT PAGE	13

LIST OF TABLES

TABLE NO	TABLE NAME	PAGE NUMBER
3.1	USER TABLES	16
3.2	MOVIE TABLES	17
3.3	BOOKING TABLES	18

CHAPTER 1
INTRODUCTION

1.1.PROBLEM STATEMENT

The Movie Reservation System is a cutting-edge software solution aimed at transforming the conventional manual processes associated with booking and managing movie tickets. This innovative system introduces automation to streamline operations, minimizing time consumption and costs. Administered through a secure authentication code, the system ensures that only authorized personnel can access and manage movie listings, seat allocations, and other essential functionalities. By seamlessly integrating technology into the movie reservation process, this system offers a more efficient and user-friendly experience, marking a notable departure from traditional methods.

1. Simplify the process of reserving movie tickets.
2. Enhance the cinema experience by providing users with movie information and easy booking options.
3. Enable cinema administrators to manage reservations, movies, and show times effectively.
4. Store and analyze data for business in sight sand decision-making..
5. Time-Saving: Significantly reduces the time spent on manual ticket booking.
6. Error-Free Booking: Minimizes human errors in seat allocation and payment processing.
7. Enhanced Customer Experience: Offers features like movie trailers, reviews, and seat previews for informed decision-making.

Revenue Optimization for Theaters: Simplifies management of bookings, pricing strategies, and promotional offers.

1.2OBJECTIVE OF THIS PROJECT

The envisioned Movie Reservation System aims to seamlessly interact with administrators and effectively fulfill all proposed functionalities. Built on Java (JDBC) with a MYSQL database, the system ensures efficient management of movie details, reducing response time for user queries. This project addresses the complexities associated with manual movie reservation processes, storing comprehensive information about movies, show times, and seat availability. Emphasizing features like verification, validation, security,and user-friendliness, the system strives to optimize the movie reservation experience, offering a scope that encompasses the entire spectrum of cinema management.It provides a user-friendly interface for interactive seat selection and integrates secure payment gateways for smooth transactions. By automating the ticketing process, the system reduces manual errors, saves time, and enhances the overall customer experience. Additionally, it supports theaters in managing schedules, tracking revenue, and implementing promotional strategies, making it a comprehensive solution for modernizing the movie-going experience.

1.3.ORGANIZATION OF THE REPORT

CHAPTER 1 – INTRODUCTION
CHAPTER 2 – SYSTEM DESIGN
CHAPTER 3 – IMPLEMENTATION
CHAPTER 4 – CONCLUSION

CHAPTER 2

SYSTEMDESIGN

2.1. OVERVIEW

The system design of the Movie Reservation System is based on a three-tier architecture, including the presentation, application, and data layers. The front-end is responsible for user interactions through web or mobile applications, offering features like user registration, login, and profile management. The back-end handles core functionalities such as movie and seat management, enabling users to view movie details, select showtimes, and reserve seats in real-time.

A secure payment module is integrated to process transactions, and an automated notification system sends booking confirmations and reminders. The database forms the data layer, storing crucial information like user profiles, movie schedules, seat availability, and transaction records. Admin functionalities allow theater management to update schedules, manage bookings, track revenue, and run promotional campaigns. The system ensures seamless integration of all components to provide a reliable, secure, and user-friendly platform for booking movie tickets.

2.2.LIST OF MODULES

1. User Module
2. Movie Management Module
3. Seat Reservation Module
4. Payment Module
5. Theater Management Module
6. Notification Module

2.3. UML MODELING:

2.3.1. USE CASE DIAGRAM OVERVIEW:

A use case diagram for a Movie Reservation System illustrates the interactions between the system's users (actors) and its primary functionalities (use cases).

1. User/Customer: Books movie tickets and manages personal information.

2. Admin/Theater Manager: Manages movies, schedules, and seats.

3. Payment Gateway: Processes payments securely.

Key Use Cases for Users Include:

1. User Registration and Login:

Create an account or log in to access personalized features.

2. Browse Movies and Showtimes:

View movie details, schedules, trailers, and reviews.

3. Search Movies:

Search for movies by title, genre, language, or release date.

4. Seat Selection:

Choose preferred seats using an interactive layout.

5. Book Tickets:

Reserve tickets for selected movies and showtimes.

6. Make Payments:

Complete transactions securely using various payment methods.

8. Cancel Booking:

Cancel reserved tickets and request refunds, if applicable.

Key Use Cases for Admin include:

1. Manage Movie Listings:

Add, update, or remove movies from the system, including details like showtimes, genres, and descriptions.

2. Schedule Showtimes:

Define and modify movie schedules for different theaters.

3. Manage Seat Layouts:

Configure seating arrangements, including premium and regular seat classifications.

4. Monitor Bookings:

Track real-time ticket bookings and seat occupancy for all shows.

5. Revenue Tracking and Reporting:

Generate detailed reports on earnings, bookings, and occupancy rates.

6. Handle User Accounts:

Manage user accounts, including resolving issues or banning users for violations.

7. Manage Promotions and Discounts:

Create and apply promotional offers, discount codes,

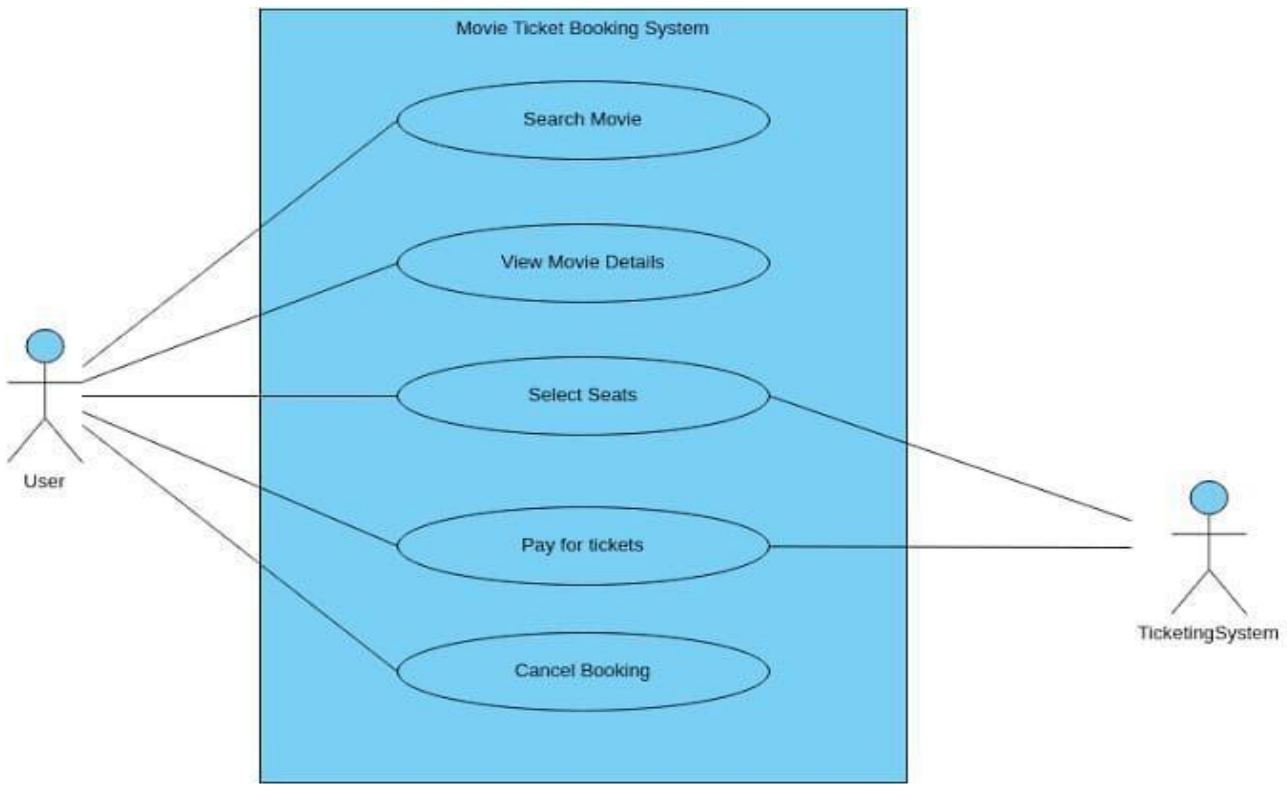


Figure 2.1. Use Case Diagram Of Movie Reservation System

2.3.2. ENTITY RELATIONSHIP DIAGRAM:

The Entity Relationship Diagram (ERD) for the Movie Reservation System illustrates the relationships between various entities in the system. Key entities include User, Movie, Showtime, Seat, Booking, and Payment. The User entity stores user details and is linked to Booking and Review. The Movie entity contains information about each movie, and is connected to Showtime and Review. Showtime entities represent the screening schedules, tied to both Movie and Theater. Payment records are associated with each Booking, ensuring secure transaction management. This ERD outlines the relationships between entities, ensuring a well-structured data model that facilitates smooth operations in the movie reservation process

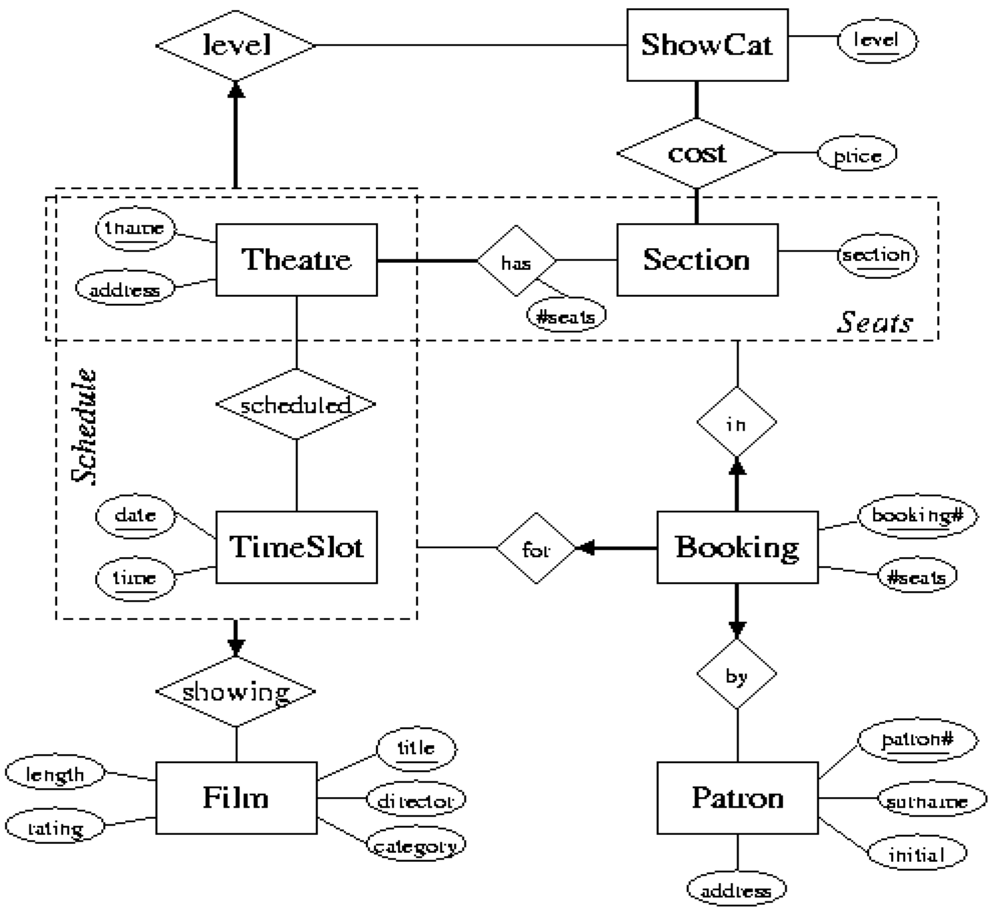


Figure 2.2. Entity Relationship Diagram Of Movie Reservation System

2.3.3. DATAFLOWDIAGRAM:

A Data Flow Diagram (DFD) is a graphical representation that illustrates the flow of data within a system. It shows how data is input, processed, stored, and output at various stages, helping to understand system functions, processes, and the interaction between different system components. DFDs are used during system analysis and design to identify inefficiencies, simplify complex systems, and communicate the system's functionality to both technical and non-technical stakeholders.

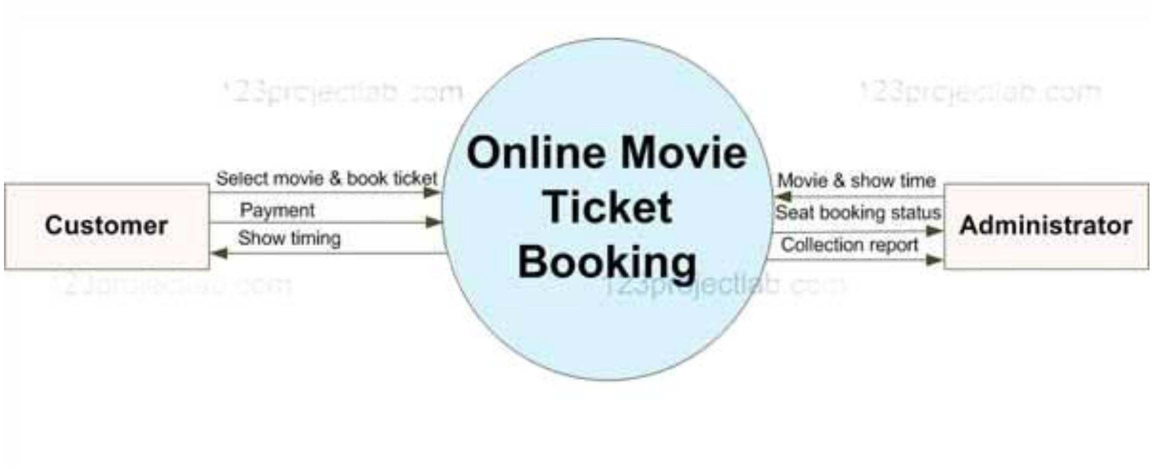


Figure 2.3. Data Flow Diagram of Movie Reservation System

1. Level 0 Context Diagram

Definition:

The Level 0 DFD, also called the Context Diagram, represents the system as a single process with inputs and outputs. It provides a high-level overview of the system and its interaction with external entities, such as users or other systems.

Purpose:

To show the overall boundaries of the system and its relationships with external entities, highlighting how the system interacts with external sources of data.

Components:

One central process representing the entire system

External entities sources or destinations of data, e.g., users, payment systems

Data flows between the system and external entities.

2. Level 2 and Beyond DFD

Definition:

Level 2 DFD (and subsequent levels) break down the processes from Level 1 into even smaller sub-processes, providing greater detail about the system's functionality. These levels are used to show how specific processes are further divided into more detailed tasks.

Purpose:

To provide an in-depth view of how specific processes are executed and how data flows between even more detailed sub-processes.

Components:

Further decomposed processes showing more specific operations.

Detailed data stores that capture more specific types of data.

Refined data flows between processes and data stores.

3. Level 1 DFD

Definition: The Level 1 DFD expands upon the context diagram by breaking the central system process from Level 0 into several high-level sub-processes. It provides a more detailed view of the system’s internal operations and shows how data flows between different processes within the system.

Purpose: To further define the system processes, breaking down the major tasks into smaller sub-processes, without delving into too much detail.

Components:

Multiple processes representing different functions within the system.Data stores that hold information used by the processes (e.g., user data, booking records).External entities that interact with the system (e.g., users, payment gateway).Data flows that show how data moves between processes, data stores, and external entities.

2.4. SYSTEM SPEFICITION

1. Frontend Specification

Login Screen: Username, password, login button, sign-up button.
Admin Panel: View/manage bookings, add/delete movies/theaters, user management.
User Booking Screen: Movie selection, theater selection, date/time selection, seat count, book button.
Sign-Up Screen: Username, password, confirm password, role selection, submit button.
Error Handling: Display error/success messages for login, sign-up, and booking.
Technology: Java Swing for GUI, MySQL for database.

2. Backend Specification

User Authentication: Login, sign-up, validate credentials.
Booking Logic: Movie selection, theater selection, availability check, booking confirmation.
Database Queries: Insert, update, delete operations for users, movies, theaters, and bookings.
Data Validation: Username/password matching, password strength, role validation.
Error Handling: Handle invalid login, database connection issues.

3. Database Specification:

Users Table: user_id (PK), username (unique), password, is_admin (boolean), deleted_at.
Theaters Table: theater_id (PK), theater_name, deleted_at.
Movies Table: movie_id (PK), movie_name, deleted_at.
Movie_Theater Table: movie_id (FK), theater_id (FK) — many-to-many relation.
Bookings Table: booking_id (PK), user_id (FK), movie_id (FK), booking_date, show_time, num_tickets, deleted_at.
Relationships: Foreign keys for user, movie, theater links; soft delete with deleted_at.

Technology Stack

Frontend: Java Swing.
Backend: Java, MySQL for database interaction.
Database: MySQL.

2.4.1.FUNCTIONAL REQUIREMENTS:

1. User Registration and Authentication:

Users must be able to register and log in to the system using their email or social media account
The system should allow for secure password recovery and support role-based access

2. Movie and Showtime Management:
Admins should be able to add, update, or remove movie listings, along with relevant information such as movie details, cast, genre, ratings, and images.
Admins can manage showtimes, including selecting dates, times, and theaters, while users can view available movies and select their preferred showtime.

3. Seat Selection and Booking:
Users should be able to select their preferred seats for a particular movie and showtime.
The system must ensure real-time seat availability, updating automatically when a seat is booked.

4. Payment Processing:
Users should be able to complete bookings by paying through integrated payment gateways
The system must provide secure payment processing, generate invoices, and send booking confirmation notifications via email or SMS.

5. Order History and Management:

Users should be able to view their booking history, including past and upcoming bookings.
Users should have the option to cancel or modify their bookings, subject to the theater’s cancellation policy.

CHAPTER 3

3.1 DESIGN:

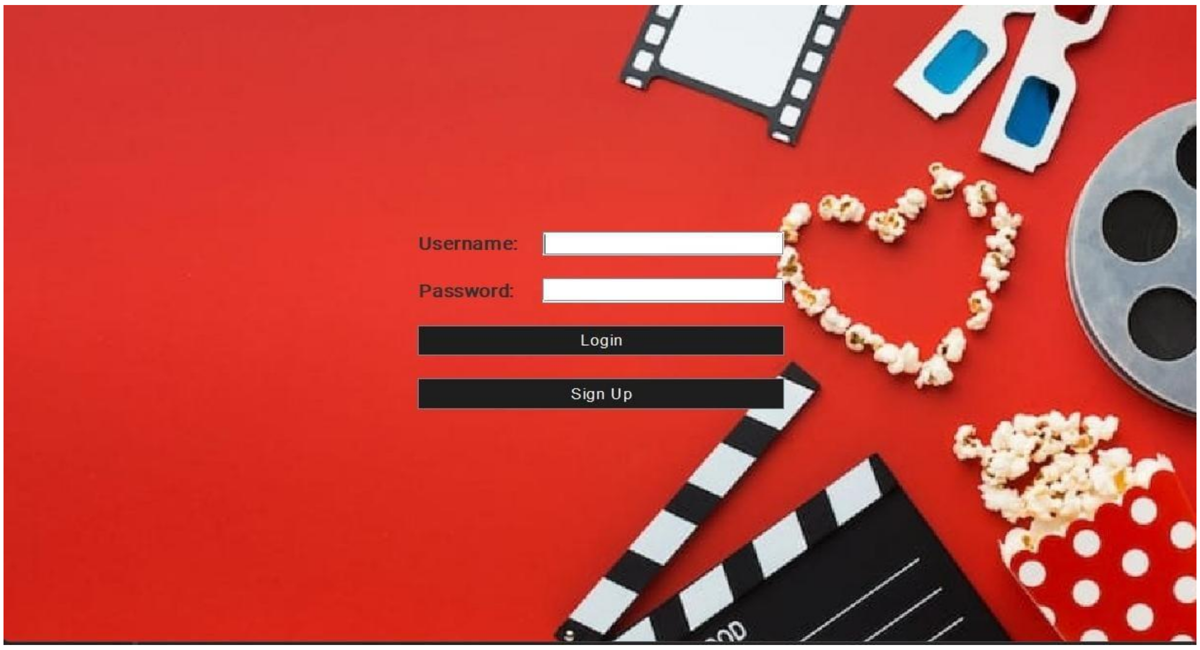
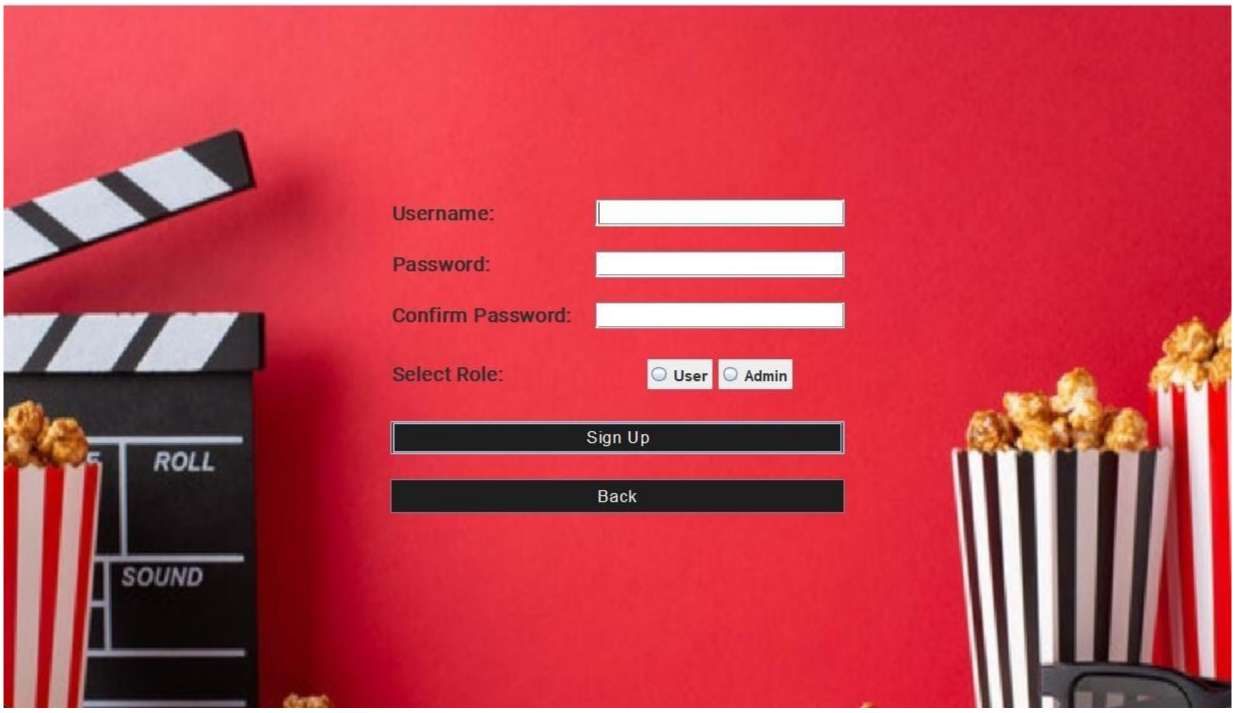


Figure3.1 Login/Signup Page



.

Figure 3.2 Sign up Page

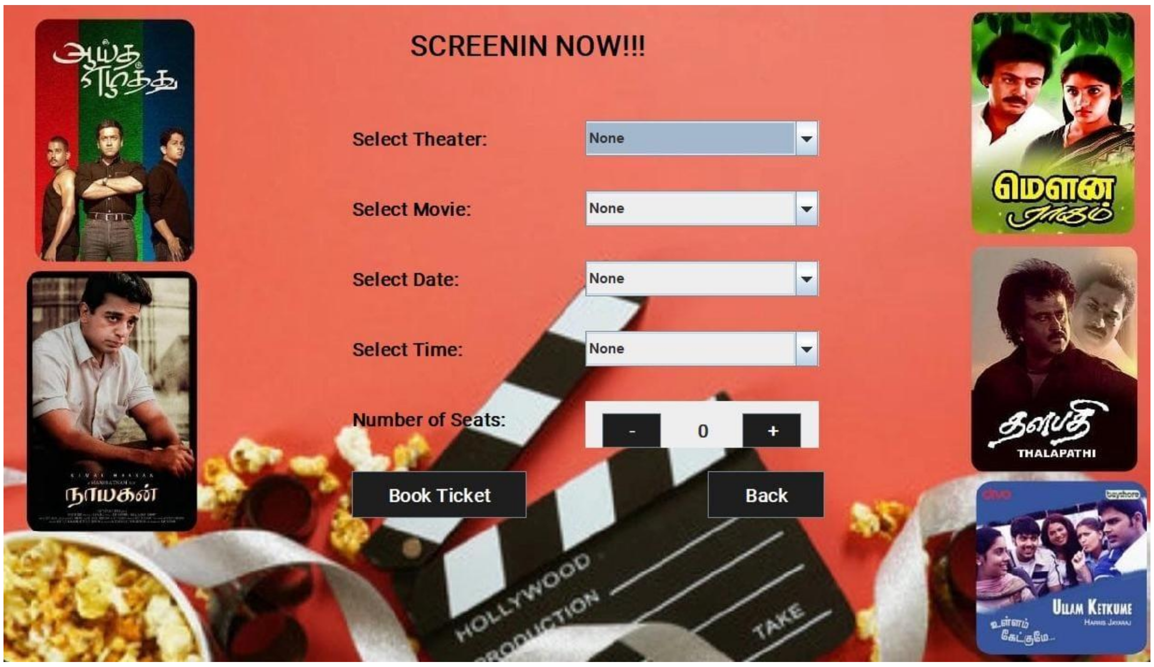
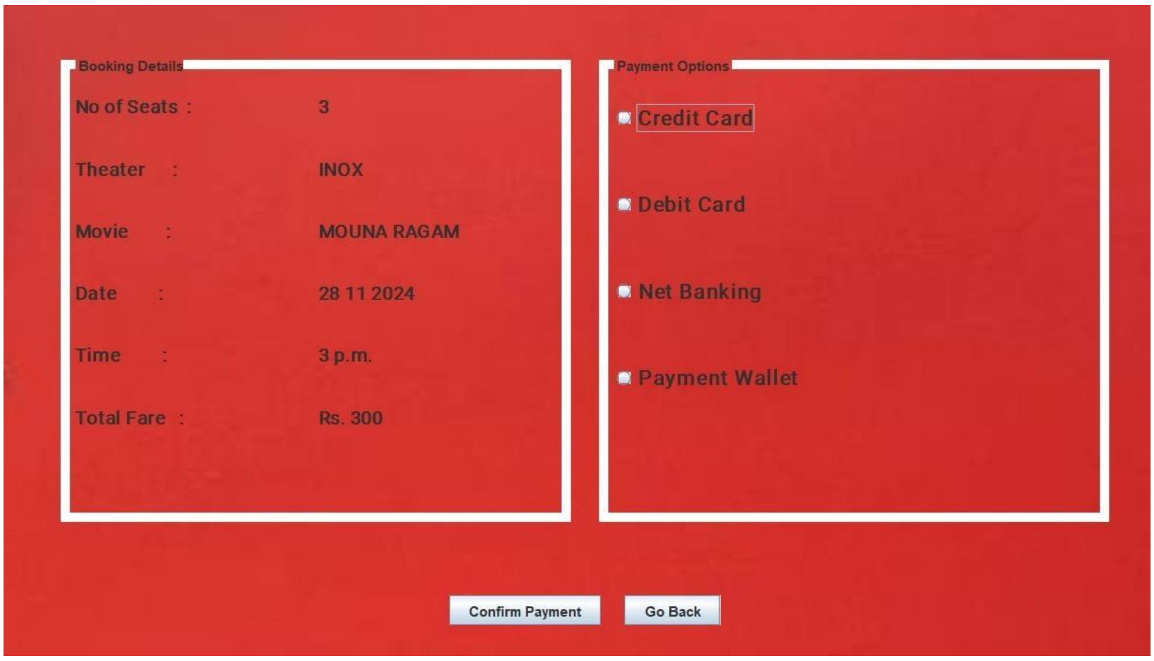


Figure 3.3 Bookings Page

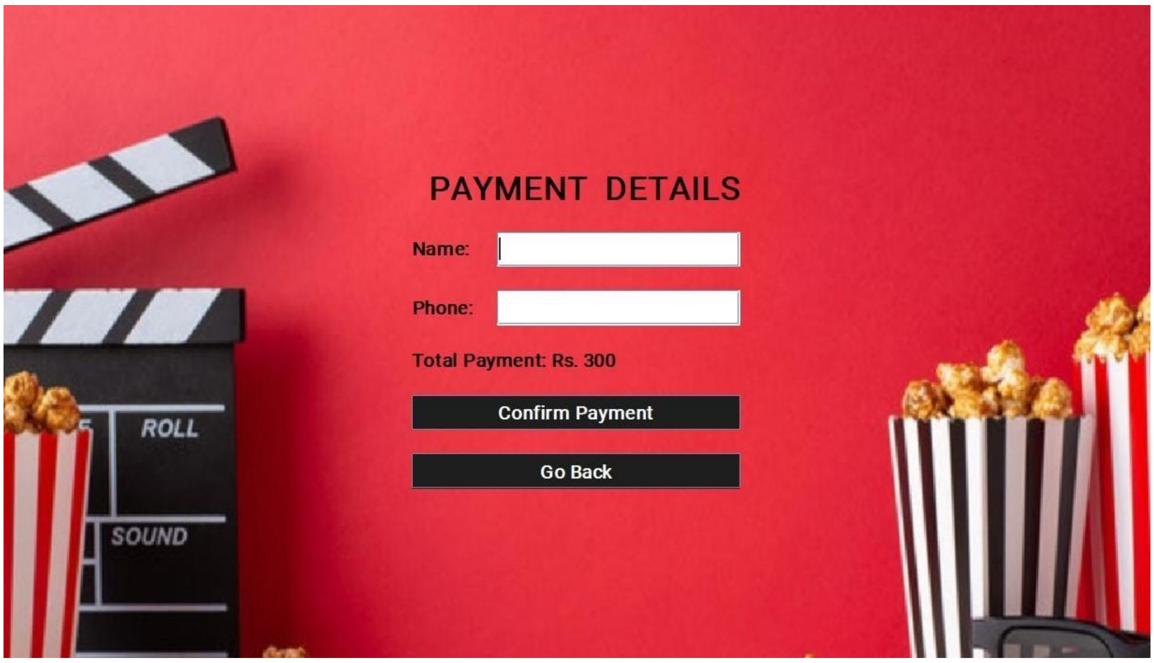


Booking Details

No of Seats :	3
Theater :	INOX
Movie :	MOUNA RAGAM
Date :	28 11 2024
Time :	3 p.m.
Total Fare :	Rs. 300

Payment Options

Figure 3.4 Booking details/payment option Page

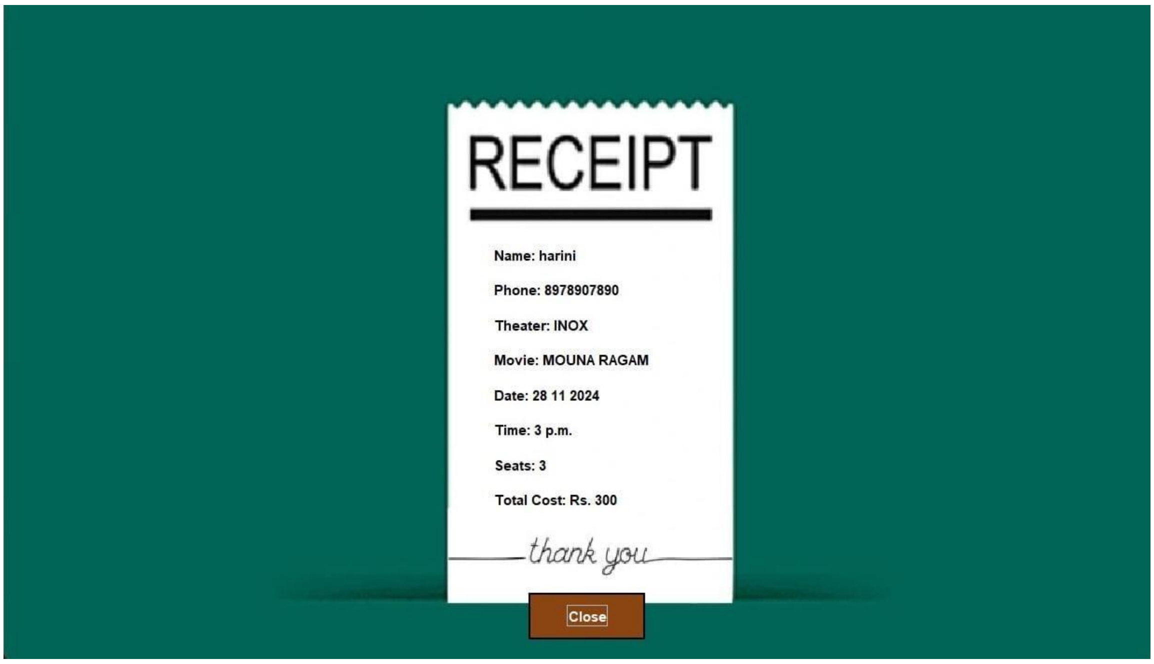

PAYMENT DETAILS

Name:

Phone:

Total Payment: Rs. 300

Figure 3.5 Payment Details Page


RECEIPT

Name: harini

Phone: 8978907890

Theater: INOX

Movie: MOUNA RAGAM

Date: 28 11 2024

Time: 3 p.m.

Seats: 3

Total Cost: Rs. 300

Figure 3.6 Reciept

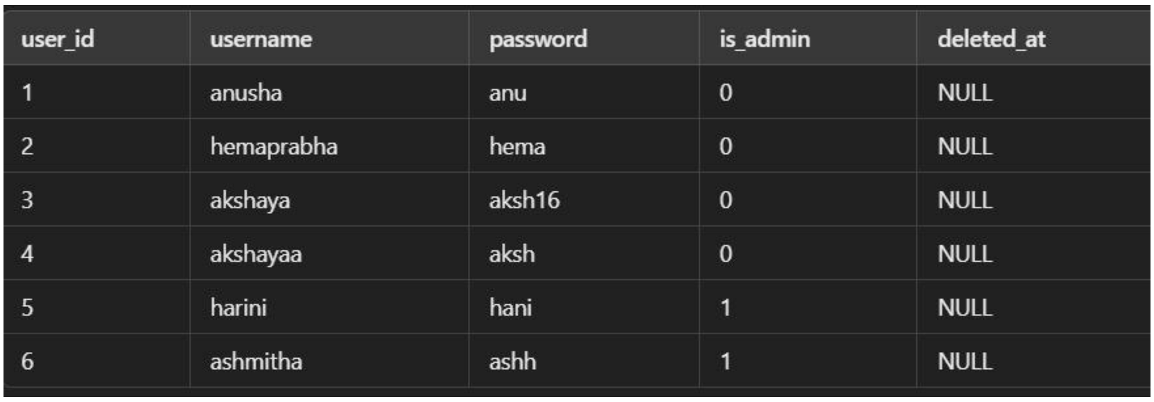
DATABASE DESIGN:

1.Users Table:

The **Users Table** stores user credentials and admin status for the cinema booking system.

Attributes of the Users Table:

- **user_id (INT)**: Unique identifier for each user.
- **username (VARCHAR(50))**: User's login name.
- **password (VARCHAR(50))**: User's login password.
- **is_admin (BOOLEAN)**: Indicates if the user is an admin (TRUE or FALSE).
- **deleted_at (DATETIME)**: Tracks soft deletions (if the account is deleted).



user_id	username	password	is_admin	deleted_at
1	anusha	anu	0	NULL
2	hemaprabha	hema	0	NULL
3	akshaya	aksh16	0	NULL
4	akshayaa	aksh	0	NULL
5	harini	hani	1	NULL
6	ashmitha	ashh	1	NULL

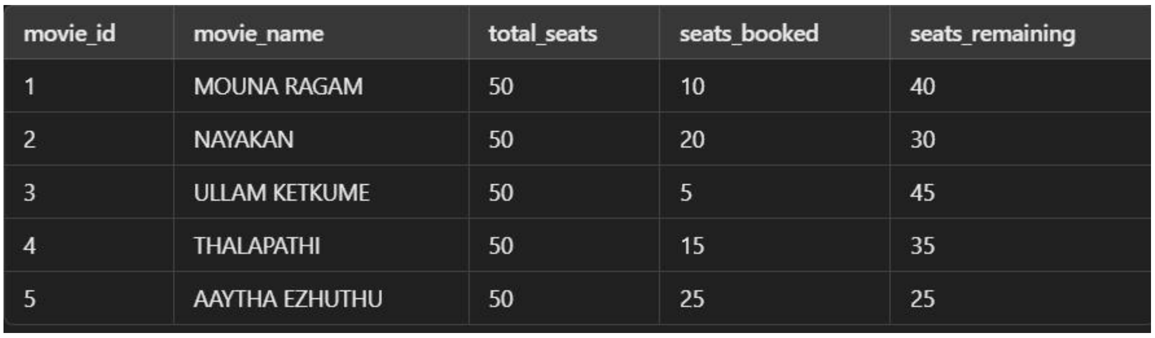
Table 3.1 User Table

2.Movies Table:

The **Movies Table** stores details of movies available for booking, including total seats, booked seats, and remaining seats.

Attributes of the Movies Table:

- **movie_id (INT)**: Unique identifier for each movie.
- **movie_name (VARCHAR(100))**: The name of the movie.
- **total_seats (INT)**: The total number of seats available for each movie.
- **seats_booked (INT)**: The number of seats already booked for the movie.
- **seats_remaining (INT)**: The number of seats remaining for booking, calculated as total_seats - seats_booked.



movie_id	movie_name	total_seats	seats_booked	seats_remaining
1	MOUNA RAGAM	50	10	40
2	NAYAKAN	50	20	30
3	ULLAM KETKUME	50	5	45
4	THALAPATHI	50	15	35
5	AAYTHA EZHUTHU	50	25	25

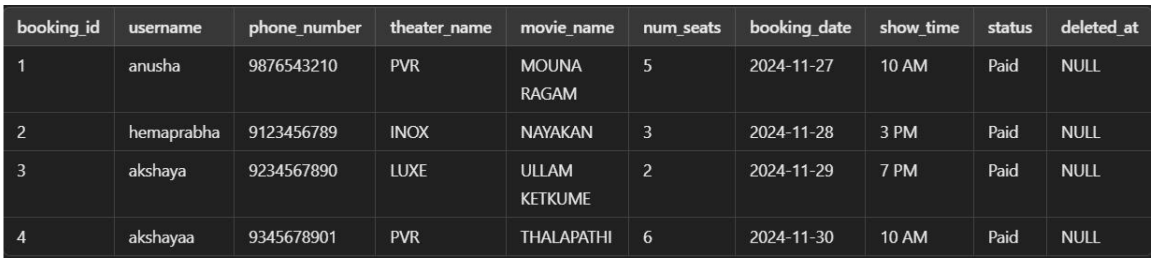
Table 3.2 Movie table

3.Bookings Table:

The **Bookings Table** stores the details of movie bookings made by users, including their contact information, theater, movie, number of seats, booking date, show time, and payment status.

Attributes of the Bookings Table:

- **booking_id (INT)**: Unique identifier for each booking.
- **username (VARCHAR(50))**: User's name making the booking.
- **phone_number (VARCHAR(10))**: User's 10-digit phone number (unique).
- **theater_name (VARCHAR(50))**: Name of the theater where the movie is being shown (PVR, INOX, LUXE).
- **movie_name (VARCHAR(100))**: The movie name booked by the user.
- **num_seats (INT)**: Number of seats booked.
- **booking_date (DATE)**: The date on which the booking is made.
- **show_time (VARCHAR(10))**: The show time for the movie (10 AM, 3 PM, 7 PM).
- **status (VARCHAR(10))**: The booking status (defaults to 'Paid').
- **deleted_at (DATETIME)**: Soft delete flag (tracks when a booking is deleted).



booking_id	username	phone_number	theater_name	movie_name	num_seats	booking_date	show_time	status	deleted_at
1	anusha	9876543210	PVR	MOUNA RAGAM	5	2024-11-27	10 AM	Paid	NULL
2	hemaprabha	9123456789	INOX	NAYAKAN	3	2024-11-28	3 PM	Paid	NULL
3	akshaya	9234567890	LUXE	ULLAM KETKUME	2	2024-11-29	7 PM	Paid	NULL
4	akshayaa	9345678901	PVR	THALAPATHI	6	2024-11-30	10 AM	Paid	NULL

Table 3.3 Bookings Table

3.3 IMPLEMENTATION (CODE):

1. login page

This part handles the login action by validating the user credentials and determining whether the user is an admin or a regular user.

```
loginButton.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        String username = usernameField.getText();
        String password = new String(passwordField.getPassword());

        // Asynchronously check user credentials
        SwingWorker<Integer, Void> loginTask = new SwingWorker<Integer, Void>() {
            @Override
            protected Integer doInBackground() throws Exception {
                return checkUserCredentials(username, password);   // Validates credentials in the
background
            }
@Override
            protected void done() {
                try {
                    int userType = get();  // Retrieve result (admin or regular user)
                    if (userType == 1) {
                        // Redirect to admin page
                    } else if (userType == 0) {
                        // Redirect to regular user page
                    } else {
                        JOptionPane.showMessageDialog(null, "Invalid username or password!", "Login
Failed", JOptionPane.ERROR_MESSAGE);
                    }
                } catch (Exception ex) {
                    JOptionPane.showMessageDialog(null, "Error occurred during login: " +
ex.getMessage(), "Error", JOptionPane.ERROR_MESSAGE);
                }
            }
        };
        loginTask.execute();  // Executes the login task in the background
}
});
```

2. Signup page

This code implements a sign-up page where users can register by providing their username, password, and role (User/Admin). It validates the inputs and stores the user information in the database.

```
// Handle user registration and store in the database
private void registerUser(String username, String password, String role) {
    try (Connection connection = 
DriverManager.getConnection("jdbc:mysql://localhost:3306/cinema_booking", "root", "")) {
        String query = "INSERT INTO users (username, password, is_admin) VALUES (?, ?, ?)";
        try (PreparedStatement statement = connection.prepareStatement(query)) {
            statement.setString(1, username);
            statement.setString(2, password);
            statement.setBoolean(3, "Admin".equals(role));  // Set is_admin based on the role
            int rowsAffected = statement.executeUpdate();
            if (rowsAffected > 0) {
                JOptionPane.showMessageDialog(null, "Registration successful!");
            }
        }
    } catch (SQLException e) {
        e.printStackTrace();
    }
}
```

3.Booking page

The showMovieBookingPage() method creates a graphical user interface (GUI) for booking movie tickets. It allows users to select a theater, movie, date, time, and the number of seats.

```
// Book Button Action
bookButton.addActionListener(e -> {
    String theater = (String) theaterComboBox.getSelectedItem();
    String movie = (String) movieComboBox.getSelectedItem();
    String date = (String) dateComboBox.getSelectedItem();
    String time = (String) timeComboBox.getSelectedItem();
    int numSeats = Integer.parseInt(seatCountLabel.getText();
    // Validate user input
    if ("None".equals(theater) || "None".equals(movie) || "None".equals(date) || "None".equals(time)
|| numSeats == 0) {
        JOptionPane.showMessageDialog(bookingFrame, "Please fill in all fields!", "Error",
JOptionPane.ERROR_MESSAGE);
    } else {
        bookingFrame.dispose(); // Close the booking frame
        showConfirmationPage(theater, movie, date, time, numSeats); // Show confirmation page
    }
});
```

Booking Confirmation: This snippet handles the display of a confirmation page once a user has successfully booked their movie tickets. It confirms the details such as theater, movie, date, time, and the number of seats selected by the user.

```
private void showConfirmationPage(String theater, String movie, String date, String time, int
numSeats) {
    JFrame confirmationFrame = new JFrame("Booking Confirmation");
    confirmationFrame.setSize(400, 300);
    confirmationFrame.setDefaultCloseOperation(JFrame.DISPOSE_ON_CLOSE);
    confirmationFrame.setLocationRelativeTo(null);

    JPanel panel = new JPanel();
    panel.setLayout(new BoxLayout(panel, BoxLayout.Y_AXIS));
    panel.add(new JLabel("Booking Confirmation"));
    panel.add(new JLabel("Theater: " + theater));
    panel.add(new JLabel("Movie: " + movie));
    panel.add(new JLabel("Date: " + date));
    panel.add(new JLabel("Time: " + time));
    panel.add(new JLabel("Seats: " + numSeats));

    // Button to close or rebook
    JButton bookAgainButton = new JButton("Book Another Ticket");
    bookAgainButton.addActionListener(e -> confirmationFrame.dispose());
    panel.add(bookAgainButton);
    confirmationFrame.add(panel);
    confirmationFrame.setVisible(true);
}
```

4.Payment Form:

Displays fields for name, phone, and payment total.

```
private void showPaymentDetailsForm(String theater, String movie, String date, String time, int numSeats)
 {
    JFrame paymentFrame = new JFrame("Payment Details");
    paymentFrame.setSize(1000, 600);
    paymentFrame.setLocationRelativeTo(null);

    JPanel paymentPanel = new JPanel();
    JTextField nameField = new JTextField(15), phoneField = new JTextField(15);
    JButton confirmButton = new JButton("Confirm Payment");
    JButton goBackButton = new JButton("Go Back");

    confirmButton.addActionListener(e -> {
        if (nameField.getText().isEmpty() || phoneField.getText().isEmpty()) {
            JOptionPane.showMessageDialog(paymentFrame, "Fill all details!", "Error", JOptionPane.ERROR_MESSAGE);
        } else {
            showReceiptPage(theater, movie, date, time, numSeats, nameField.getText(), phoneField.getText());
            paymentFrame.dispose();
        }
    });

    goBackButton.addActionListener(e -> {
        paymentFrame.dispose();
        showConfirmationPage(theater, movie, date, time, numSeats);
    });

    paymentPanel.add(new JLabel("Name:"));
    paymentPanel.add(nameField);
    paymentPanel.add(new JLabel("Phone:"));
    paymentPanel.add(phoneField);
    paymentPanel.add(new JLabel("Total: Rs. " + (numSeats * 100)))
  paymentPanel.add(confirmButton);
    paymentPanel.add(goBackButton);

    paymentFrame.add(paymentPanel);
paymentFrame.setVisible(true);
}
```

4.Receipt page

The purpose of this code is to create a receipt page for a movie booking system. It displays the booking details such as the name, phone number, theater, movie, date, time, number of seats, and the total cost. It also includes a "Close" button to close the receipt window and navigate back to the login page.

```
private void showReceiptPage(String theater, String movie, String date, String time, int numSeats,
String name, String phone) {
    JFrame receiptFrame = new JFrame("Receipt");
    receiptFrame.setSize(1000, 600);
    receiptFrame.setLocationRelativeTo(null);

    // Background setup
    JLabel backgroundLabel = new JLabel(new ImageIcon(new ImageIcon("Images/bill.jpg")
        .getImage().getScaledInstance(1000, 600, Image.SCALE_SMOOTH)));
    receiptFrame.add(backgroundLabel);

    // Panel setup for details
    JPanel panel = new JPanel(null);
    panel.setOpaque(false);
    receiptFrame.add(panel);

    // Labels with details
    String[] details = {"Name: " + name, "Phone: " + phone, "Theater: " + theater,
                        "Movie: " + movie, "Date: " + date, "Time: " + time,
                        "Seats: " + numSeats, "Total: Rs. " + (numSeats * 100)};
    int yPos = 200;
    for (String text : details) {
        panel.add(new JLabel(text, JLabel.LEFT).setBounds(420, yPos, 500, 30));
        yPos += 30;
    }

    // Close button
    JButton closeButton = new JButton("Close");
    closeButton.setBounds(450, 500, 100, 40);
    closeButton.setBackground(new Color(139, 69, 19));
    closeButton.setForeground(Color.WHITE);
    closeButton.addActionListener(e -> {
        receiptFrame.dispose();
        initializeLoginForm();
    });
    panel.add(closeButton);

    // Show frame
    receiptFrame.setVisible(true);
}
```

CHAPTER 4

CONCLUSION

The movie reservation system is an innovative solution designed to simplify the process of booking movie tickets while enhancing the user experience and operational efficiency. This system ensures seamless integration of multiple components, including user management, movie scheduling, theater operations, and payment tracking. The robust database design facilitates real-time data management and retrieval, enabling users to view available showtimes, select seats, and make reservations effortlessly. For administrators, it offers streamlined operations for managing movies, showtimes, and customer interactions.Designed with a modular approach, the system is future-proof, enabling easy upgrades to incorporate advanced features such as loyalty programs, dynamic pricing, and multilingual support. By leveraging modern technology, the movie reservation system addresses the challenges of traditional booking methods, providing a reliable, user-friendly platform that meets the needs of a diverse audience. It is a practical, efficient, and scalable solution for the modern entertainment industry.

REFERENCES:

[1] https://dev.mysql.com/doc/connector-j/
[2] https://docs.oracle.com/javase/tutorial/jdbc/basics/
[3] https://www.javatpoint.com/java-swing
[4] https://www.w3schools.com/sql/
[5] https://www.geeksforgeeks.org/establishing-jdbc-connection-in-java/
[6] "Java Technology in the Design and Implementation of Web Applications," January 2012, Technies Technologies Education Management
[7] Research on "Java Web Framework based on OSGi," December 2011, Procedia Engineering