MOVIE TICKET BOOKING RESERVATION SYSTEM

CS23333 – Object Oriented Programming Using JAVA Project Report

Submitted by

DHAKSHINYA P.I - 231001032

HARSHITHA.S - 231001061

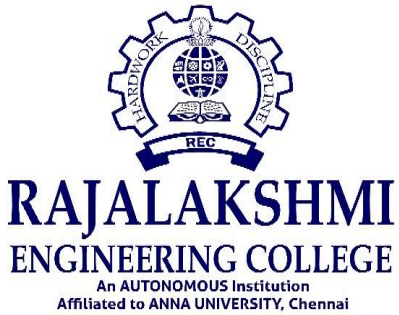
`

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 **DHAKSHINYA(231001032)**,**HARSHITHA(231001061)** 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:

1.1 Abstract

The Movie Reservation System is a GUI-based application developed in Java, integrated with a database to provide a seamless user experience for booking movie tickets. The system allows users to register or log in, select preferred dates and timings, view available movies, and reserve tickets. The process also includes payment integration through mobile numbers, ensuring secure and convenient transactions. This project aims to digitize and streamline the movie-ticketing process, offering a user-friendly interface for both users and administrators. The system enhances efficiency by reducing manual errors, improving accessibility, and enabling a smoother reservation experience.

1.2 Introduction

In today's fast-paced world, technology has transformed various sectors, including the entertainment industry. Movie ticket booking, which was once a manual and time-consuming process, has now evolved into an efficient online system. This project, "Movie Reservation System," combines Java-based GUI and database connectivity to develop a reliable platform for booking movie tickets.

The system is designed to cater to users by providing functionalities such as user login, movie schedule selection, availability checking, ticket reservation, and secure payment processing. It integrates modern programming techniques with database management to ensure data integrity and user satisfaction. This project not only simplifies the ticket booking process but also promotes digital transactions, aligning with the global trend toward cashless economies.

1.3 Purpose

The primary purpose of the Movie Reservation System is to create a digital platform for managing movie ticket reservations efficiently. This system aims to:

- Provide a user-friendly interface for booking movie tickets.
- Automate the process of selecting movie schedules and managing reservations.
- Reduce errors and inefficiencies associated with manual ticket booking.
- Enable secure and hassle-free payment through mobile numbers.
- Maintain a centralized database for storing user details, movie schedules, and transaction records.

1.4 Scope

The scope of the Movie Reservation System extends to various functionalities designed to meet the needs of both users and administrators. These include:

User Registration and Login: Secure authentication mechanism for new and existing users.
Movie Schedule Selection: Option to select preferred dates and timings from the available schedule.
Real-time Movie Availability: Display of currently available movies along with show details.
Ticket Reservation: Efficient ticket booking with seat selection capabilities.
Payment Processing: Integration with mobile payment systems for secure transactions.
Database Management: Centralized storage of user profiles, booking records, and movie schedules to ensure data consistency and reliability.

The system is scalable and can be extended to include additional features such as loyalty programs, discount offers, and multi-language support. It is intended to be deployed in cinemas, online platforms, or as a mobile application.

1.5 Software Requirement Specification

Introduction:

The Movie Reservation System is a GUI-based software application developed using Java and MySQL to streamline the process of booking movie tickets. It allows users to log in, select showtimes, view available movies, and reserve tickets with ease.

Document Purpose:

This document outlines the software requirements for the development and implementation of the Movie Reservation System. It serves as a detailed reference for the development team, stakeholders, and end-users, ensuring that all functional and non-functional requirements are clearly defined and met.

Product Scope:

The Movie Reservation System is a GUI-based application developed using Java and MySQL, integrated with Visual Studio Code for development. The system enables users to log in, view available movies, select schedules, reserve tickets, make payments through mobile numbers, and receive payment confirmations. The system simplifies ticket booking processes, reduces manual intervention, and enhances user convenience.

References and Acknowledgments:

- Oracle Java Documentation: https://docs.oracle.com
- MySQL Documentation: https://dev.mysql.com

Product Perspective:

The Movie Reservation System operates as a standalone application with a centralized MySQL database. The system's Java-based GUI is developed using Swing/JavaFX, while database operations are handled via JDBC. The software integrates seamlessly with payment gateways to provide a secure booking and payment experience.

Product Functionality:

Key functionalities of the system include:

- User registration and login.
- Movie schedule selection by day and time.
- Display of available movies with details.
- Ticket booking with seat selection.
- Secure payment through mobile-based methods.
- Automatic generation of booking confirmations.

Users and Characteristics:

- End-users:
    - Moviegoers looking to book tickets conveniently.
- Administrators:
    - Manage movie schedules, availability, and user data.

Functional Requirements

- User Management:

  Allow users to register and log in securely.
- Movie Selection:

  Display movies based on the selected date and time.
- Ticket Booking:

  Facilitate seat selection and confirm booking.
- Payment Processing:

  Support mobile number-based payment methods.
- Database Management:

  Store user data, movie schedules, bookings, and payment records.

Non-Functional Requirements**:**

- Performance:

  Ensure fast response times for database queries.
- Scalability:

  Support an increasing number of users and bookings.
- Security:

  Protect user data using encrypted database storage.
- Reliability:

  Ensure 99.9% uptime for the application.
- Usability:

  Provide an intuitive and accessible user interface.

2.SYSTEM FLOW DIAGRAM

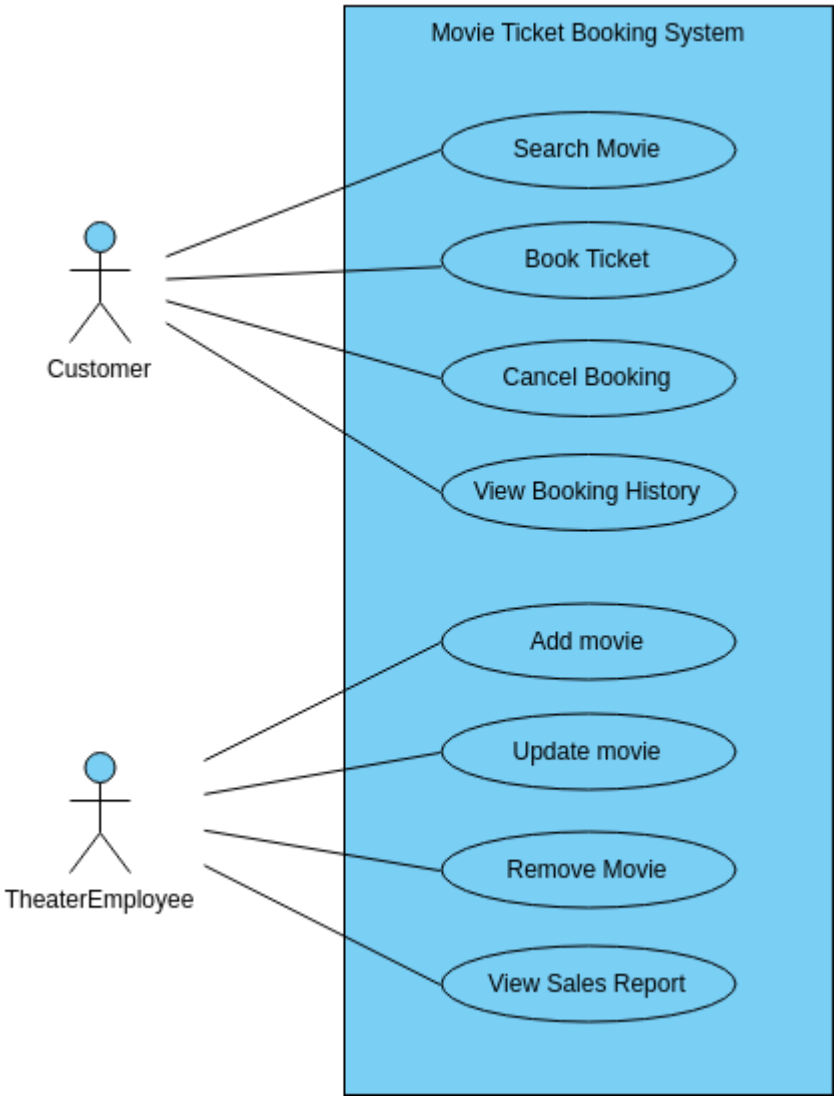


Figure 2.1.1 Use Case Diagram

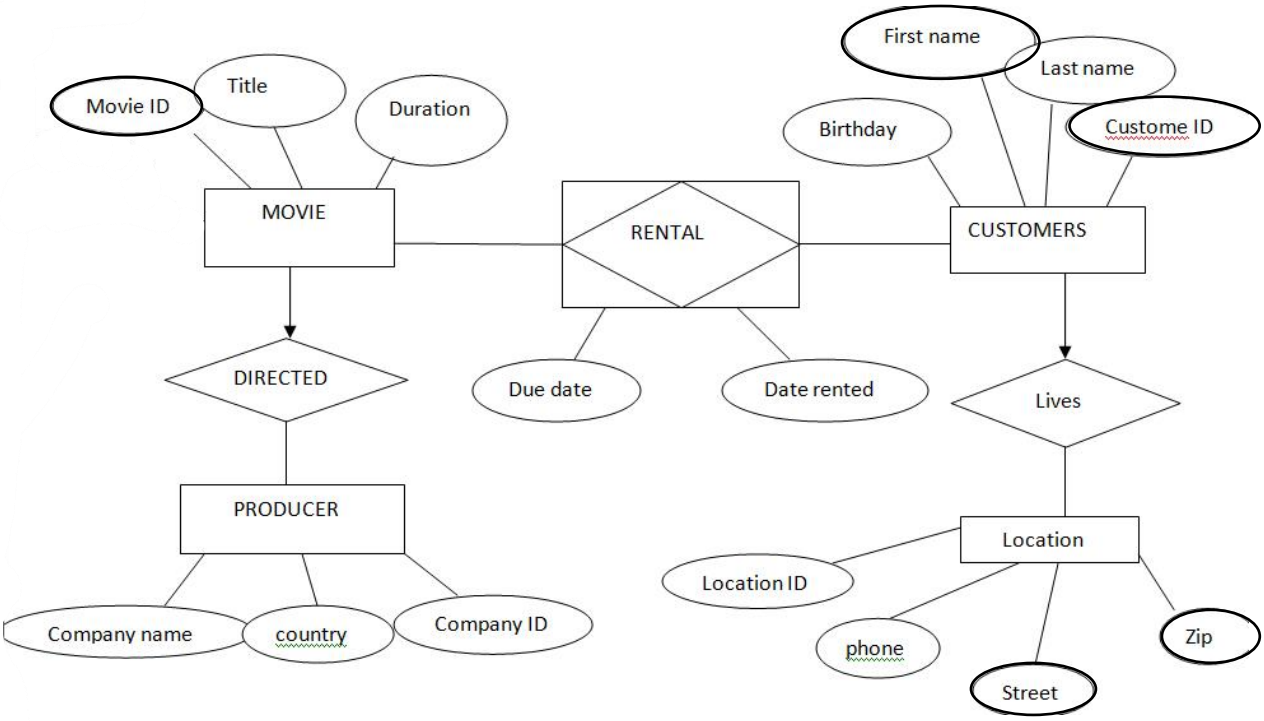


Figure 2.2.1 Entity Relation Diagram

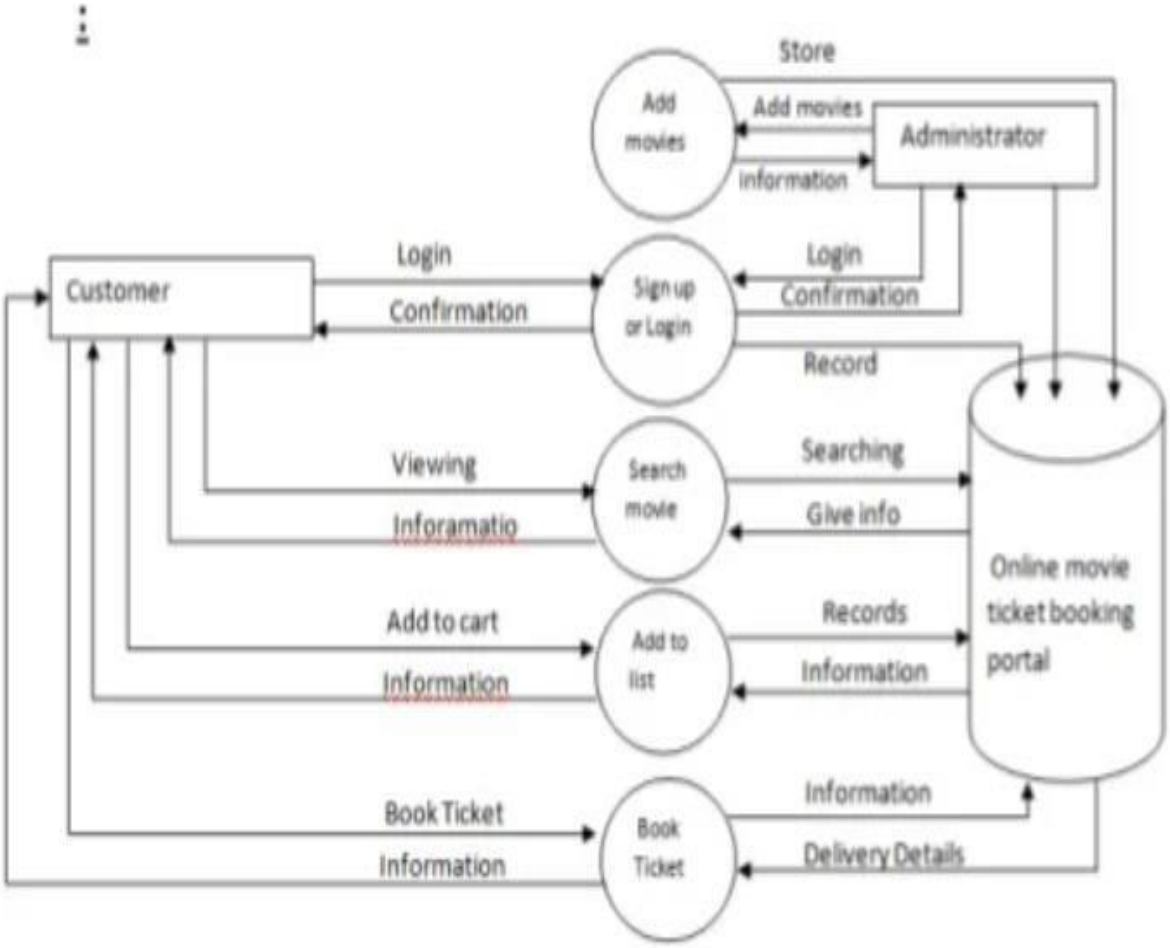


Figure 2.3.1 Data Flow Diagram

3.Module Description:

1. User Authentication Module

- Purpose: To allow users to securely log in and register.
- Features:
    - User registration with necessary details (e.g., name, email, password).
    - User login with validation against stored credentials in the database.
    - Password encryption for secure storage.
    - Error handling for failed login attempts.

2. Movie Scheduling and Selection Module

- Purpose: To enable users to choose a preferred date, time, and view available movies.
- Features:
    - Calendar-based date selection.
    - Dropdown or list selection for movie showtimes.
    - Display of movies filtered by the selected date and time.
    - Integration with the database to fetch real-time availability.

3. Ticket Reservation Module

- Purpose: To allow users to reserve tickets for their selected movie and showtime.
- Features:
    - Seat selection (optional if seating is specified).
    - Real-time updates on seat or ticket availability.
    - Storage of reservation details in the database.

4. Payment Module

- Purpose: To facilitate secure payment for ticket reservations.
- Features:

- Payment through mobile number integration (e.g., wallet or carrier-based payment methods).
- Validation of payment details and confirmation of transaction.
- Generation of a payment receipt upon successful payment.

5. Confirmation and Exit Module

- Purpose: To confirm the reservation and allow the user to exit the system.
- Features:
  - Display of booking confirmation, including movie details, date, time, and seat information.
  - Option to print or save the booking receipt.
  - Smooth exit from the system, saving user session details.

4.1 Design

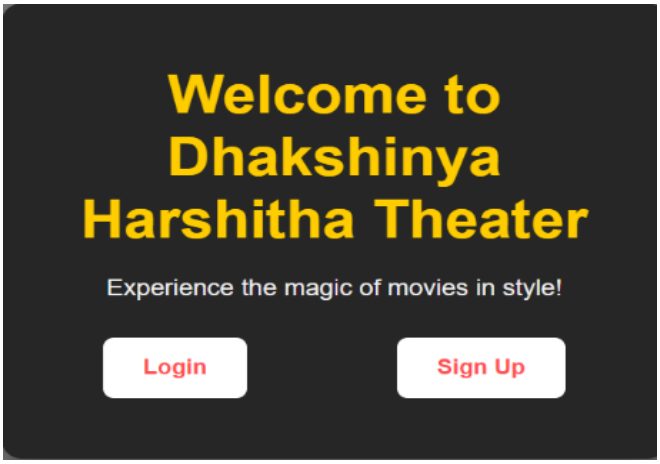


Figure 4.1.1 Home Page

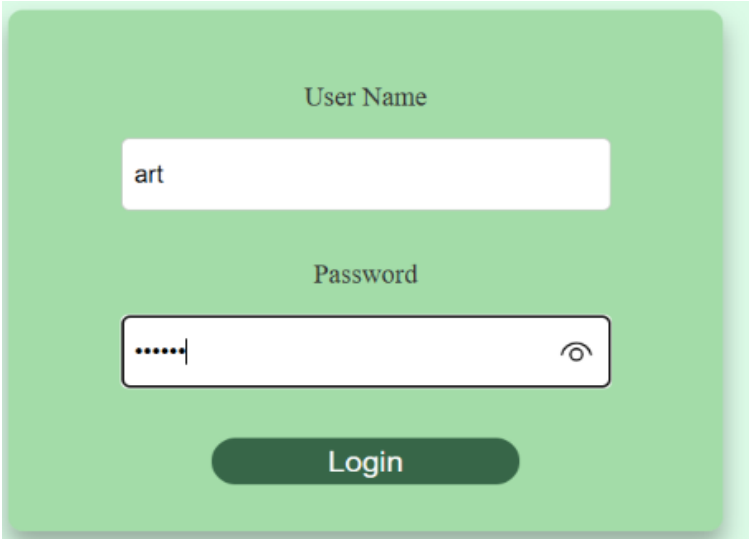


Figure 4.1.2 Login Page

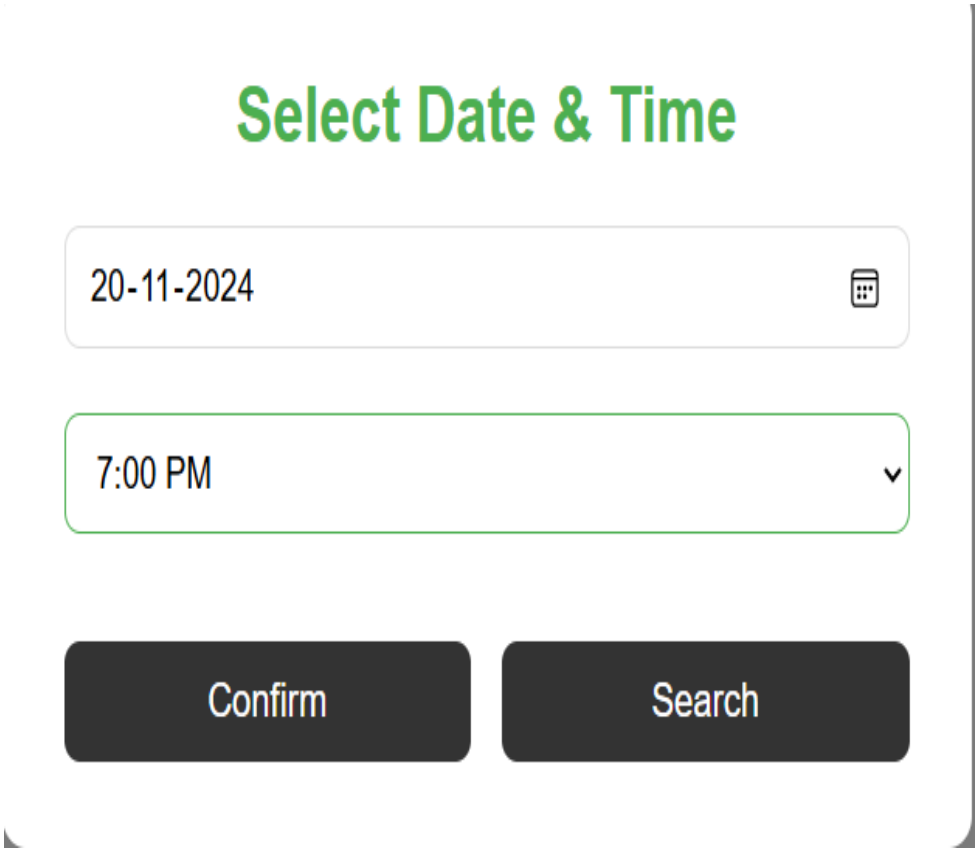


Figure 4.1.3 Date and Time Selection

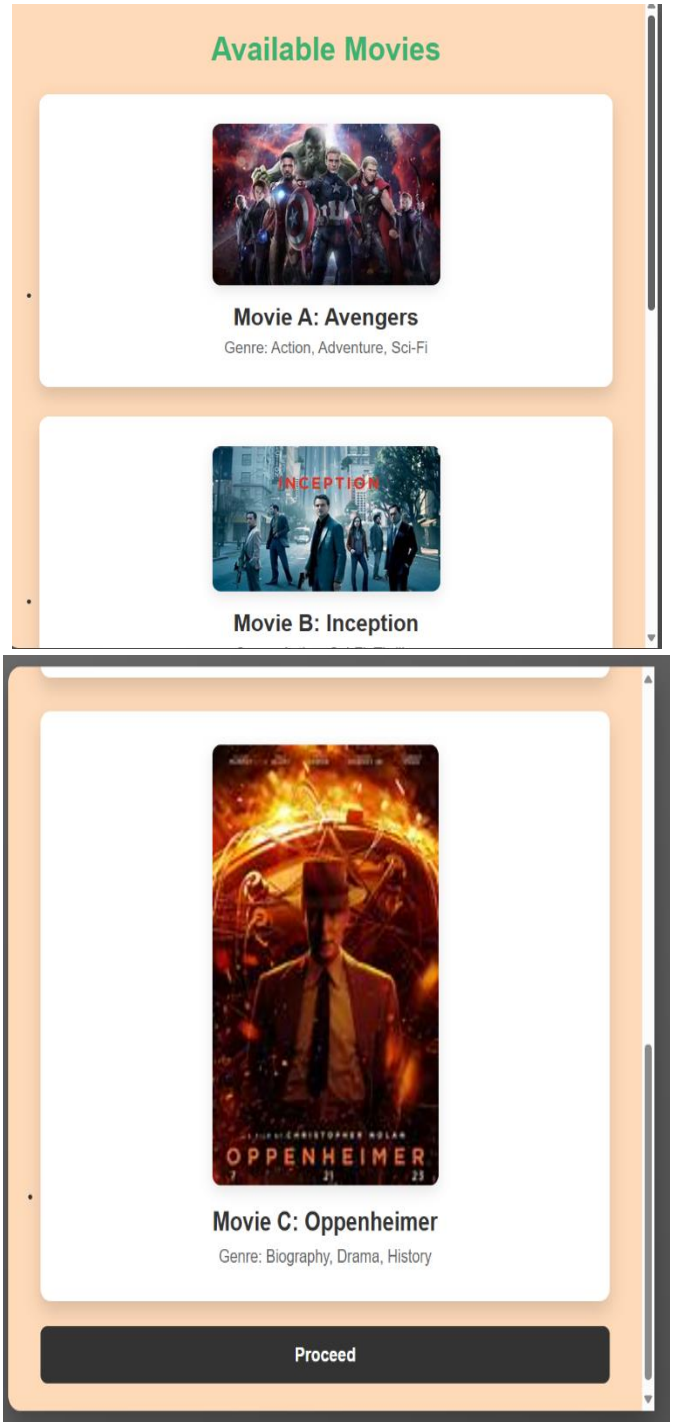


Figure 4.1.4 Movie Selection

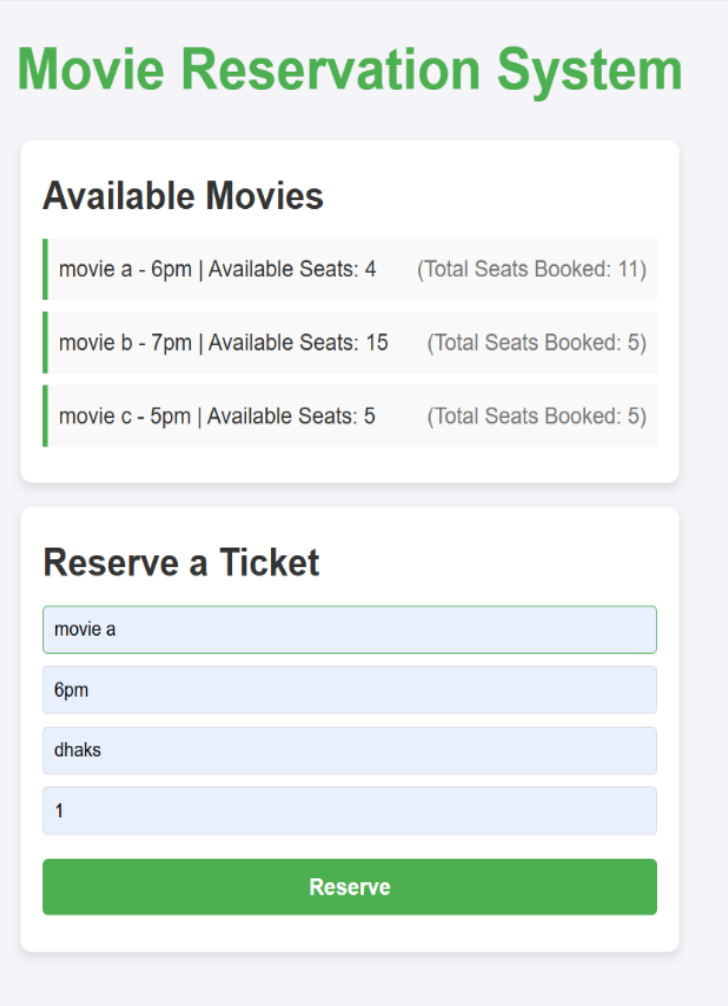
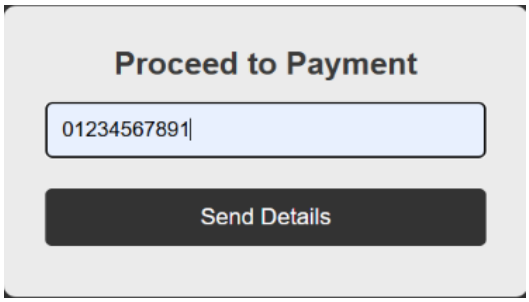
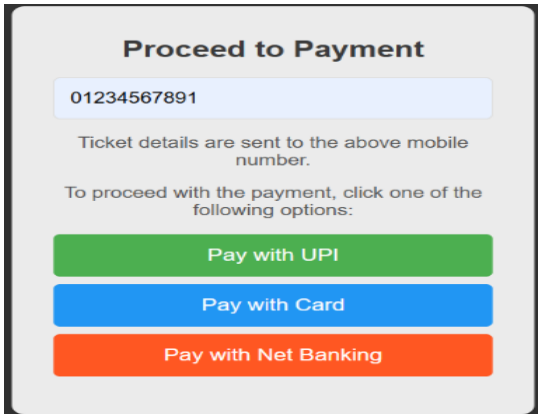


Figure 4.1.5 Movie Reservation





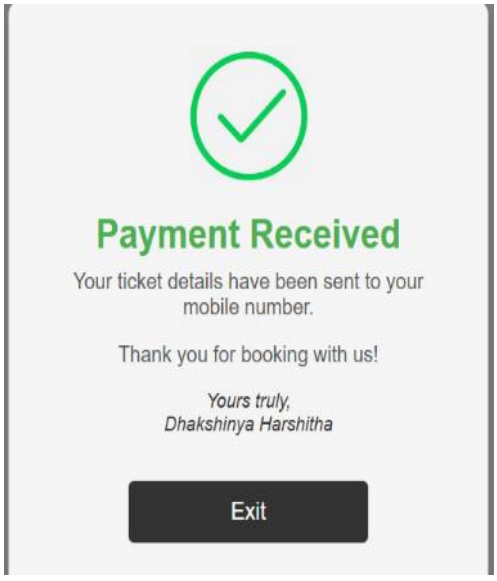


Figure 4.1,6 Payment Panel

4.2 Database Design for Movie Reservation System:

The database design for the Movie Reservation System is crucial for efficiently managing user data, movie information, show schedules, reservations, and payment transactions. The system is structured to ensure data integrity, maintain relationships between different entities, and allow seamless integration with the application's modules.

The database is designed with the following key principles in mind:

1. Normalization: To avoid redundancy and ensure data consistency.
2. Relational Integrity: To establish relationships between tables (such as linking users, movies, and reservations).
3. Scalability: To accommodate future expansions, such as adding new features like reviews, ratings, or loyalty programs.

Movie Reservation System which contains SQL Plus tables:

```
SQL> select * from movies;

MOVIE_NAME         MOVIE_TIME            PSEATS    USEATS
------------------ ------------------ --------- ---------
movie a            6pm                        3        12
movie b            7pm                       15         5
movie c            5pm                        5         5
```

Table 4.2.1 Database design

```
SQL> select * from reservations;

USERNAME             MOVIE_NAME          SEATS_RESERVED
-------------------- ------------------- --------------
dhaks                movie a                          2
dhaks                movie a                          2
dhaks                movie a                          1
dhaks                movie b                          2
dhaks                movie b                          1
dhaks                movie b                          1
dhaks                movie b                          1
dhaks                movie c                          1
dhaks                movie c                          1
dhaks                movie c                          1
dhaks                movie c                          1

USERNAME             MOVIE_NAME          SEATS_RESERVED
-------------------- ------------------- --------------
dhaks                movie c                          1
dhaks                movie a                          1
dhaks                movie a                          1
dhaks                movie a                          1
dhaks                movie a                          1
dhaks                movie a                          1
dhaks                movie a                          1
dhaks                movie a                          1

19 rows selected.
```

Table 4.2.2 Database design

4.3 Implementation(CODE):

```
import java.sql.*;

import java.util.*;


public class MovieReservationApp {

    public static void main(String[] args) {

        Scanner scanner = new Scanner(System.in);

        Connection connection = null;



        try {

            // Establish connection to the database

            connection = DriverManager.getConnection("jdbc:oracle:thin:@localhost:1521:XE",
"system", "2306");

            System.out.println("Connected to the database successfully!");



            while (true) {

                System.out.println("1. List Movies");

                System.out.println("2. Reserve Ticket");

                System.out.println("3. Exit");

                System.out.print("Choose an option: ");

                int choice = scanner.nextInt();

                scanner.nextLine();  // Consume newline
```

```
            switch (choice) {
                case 1:
                    listMovies(connection);
                    break;
                case 2:
                    reserveTicket(connection, scanner);
                    break;
                case 3:
                    System.out.println("Exiting the system.");
                    return;
                 default:
                    System.out.println("Invalid option, please try again.");
            }
        }
    } catch (SQLException e) {
        System.out.println("Database connection failed: " + e.getMessage());
        e.printStackTrace();
    } finally {
        try {
            if (connection != null) {
                connection.close();
            }
```

```
            scanner.close();
        } catch (SQLException e) {
            System.out.println("Failed to close the database connection: " + e.getMessage());
        }
    }
}

// Method to list movies and their showtimes from the database
@SuppressWarnings("unused")
private static void listMovies(Connection connection) throws SQLException {
    String query = "SELECT ID, TITLE, SHOWTIME FROM movie";
    try (Statement stmt = connection.createStatement(); ResultSet rs = stmt.executeQuery(query)) {
        Map<String, List<String>> movieShowtimes = new HashMap<>();

        while (rs.next()) {
            String title = rs.getString("TITLE");
            String showtime = rs.getString("SHOWTIME");
            movieShowtimes.computeIfAbsent(title, k -> new ArrayList<>()).add(showtime);
        }

        if (movieShowtimes.isEmpty()) {
            System.out.println("No movies available.");
```

```
        } else {
            for (Map.Entry<String, List<String>> entry : movieShowtimes.entrySet()) {
                System.out.println("Movie: " + entry.getKey() + ", Showtimes: " + entry.getValue());
            }
        }
    }
}

// Method to reserve a ticket
private static void reserveTicket(Connection connection, Scanner scanner) throws SQLException {
    System.out.print("Enter movie title: ");
    String title = scanner.nextLine();
    System.out.print("Enter showtime: ");
    String showtime = scanner.nextLine();
    System.out.print("Enter your name: ");
    String customerName = scanner.nextLine();

    // Check if the movie and showtime exist
    if (!isShowtimeValid(connection, title, showtime)) {
        System.out.println("Invalid showtime for the selected movie.");
        return;
    }
```

```
        // Insert reservation into the database
        if (reserveTicketInDatabase(connection, title, showtime, customerName)) {
            System.out.println("Ticket reserved successfully.");
        } else {
            System.out.println("Reservation failed. The showtime might be already reserved.");
        }
    }

    // Check if the provided movie title and showtime exist
    private static boolean isShowtimeValid(Connection connection, String title, String showtime) throws SQLException {
        String query = "SELECT COUNT(*) FROM movie WHERE TITLE = ? AND SHOWTIME = ?";
        try (PreparedStatement pstmt = connection.prepareStatement(query)) {
            pstmt.setString(1, title);
            pstmt.setString(2, showtime);
            ResultSet rs = pstmt.executeQuery();
            if (rs.next()) {
                return rs.getInt(1) > 0;
            }
        }
        return false;
    }
```

```
    // Method to insert a reservation into the database
    private static boolean reserveTicketInDatabase(Connection connection, String title, String showtime, String customerName) throws SQLException {
        // Get ROOM_NUMBER (showtime ID) from the movie table
        String getRoomNumberQuery = "SELECT ID FROM movie WHERE TITLE = ? AND SHOWTIME = ?";
        try (PreparedStatement pstmt = connection.prepareStatement(getRoomNumberQuery)) {
            pstmt.setString(1, title);
            pstmt.setString(2, showtime);
            ResultSet rs = pstmt.executeQuery();
            if (rs.next()) {
                String roomNumber = rs.getString("ID");

                // Check if the customer has already reserved the same showtime
                String checkQuery = "SELECT COUNT(*) FROM reservations WHERE ROOM_NUMBER = ? AND CUSTOMER_NAME = ?";
                try (PreparedStatement checkStmt = connection.prepareStatement(checkQuery)) {
                    checkStmt.setString(1, roomNumber);
                    checkStmt.setString(2, customerName);
                    ResultSet checkRs = checkStmt.executeQuery();
                    if (checkRs.next() && checkRs.getInt(1) == 0) {
                        // Proceed with insertion if no existing reservation
```

```
            String insertQuery = "INSERT INTO reservations (ROOM_NUMBER, CUSTOMER_NAME, CHECK_IN_DATE, CHECK_OUT_DATE) VALUES (?, ?, ?, ?)";

            try (PreparedStatement insertStmt = connection.prepareStatement(insertQuery)) {

                insertStmt.setString(1, roomNumber);  // ROOM_NUMBER is the movie ID

                insertStmt.setString(2, customerName);

                insertStmt.setString(3, showtime);  // CHECK_IN_DATE is the showtime as string

                insertStmt.setString(4, showtime);  // CHECK_OUT_DATE is the same (can be ignored in this case)

                return insertStmt.executeUpdate() > 0;

              }

            }

          }

        }

      }

      return false;

   }

}
```

5.Conclusion:

The Movie Reservation System is a comprehensive, user-friendly solution designed to simplify the process of booking movie tickets. By utilizing Java for GUI development and MySQL for database management, it provides a seamless experience for users and administrators, enabling secure login, movie selection, ticket reservation, and mobile-based payment processing. The system's intuitive interface and efficient backend ensure fast, reliable handling of user requests and movie schedules, while the database design guarantees data consistency, scalability, and integrity. This project successfully enhances user convenience, reduces manual errors, and offers a modern, digital alternative to traditional ticket booking methods, with future scalability for additional features such as more payment options or loyalty programs. Ultimately, the system provides a solid foundation for modernizing ticketing processes in cinemas and entertainment venues.

6.REFERENCE :

https://code.visualstudio.com/docs

https://dev.mysql.com/doc/mysql-shell/8.0/en/

https://dev.mysql.com/doc/