MOVIE RESERVATION SYSTEM

A PROJECT REPORT

Submitted by

ARCHANA A - 231001016

BHAGHYA LAKSHMI M - 231001025

in partial fulfilment for the award of the degree

Of

BACHELOR OF TECHNOLOGY

In

INFORMATION TECHNOLOGY

DEPARTMENT OF INFORMATION TECHNOLOGY

RAJALAKSHMI ENGINEERING COLLEGE

NOVEMBER 2024

BONAFIDE CERTIFICATE

Certified that this project report titled "**MOVIE RESERVATION SYSTEM** " is the bonafide work of **ARCHANA A (231001016), BHAGHYA LAKSHMI M (231001025)** who carried out the project work under my supervision. Certified further that to the best of my knowledge the work reported herein does not form part of any other thesis or 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**
Department Of Information Technology Rajalakshmi Engineering College	Department Of Information Technology Rajalakshmi Engineering College

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

INTERNAL EXAMINAR EXTERNAL EXAMINAR

ABSTRACT

The Movie Reservation System is a comprehensive solution designed to simplify movie ticket bookings for a theatre with two halls, offering features such as user authentication, movie selection, interactive seat reservation, and secure payment handling. It ensures real-time data consistency and efficient management of concurrent user requests, reducing issues like double bookings and scheduling conflicts. Users can securely log in or register, browse available movies with detailed show timings and ticket prices, and dynamically select seats based on availability. Secure and fast payment processing ensures a hassle-free experience, with instant booking confirmations provided upon successful transactions. By automating key operations, the system minimizes manual workload for theatre staff, enabling administrators to efficiently manage movie schedules, monitor bookings, and maintain accurate records through a centralized platform. This all-digital approach bridges the gap between customer expectations and theatre management requirements, delivering a scalable, reliable, and user-friendly solution for modern movie ticket reservations.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

1.1 PROBLEM STATEMENT

Traditional movie ticket booking systems, such as in-person counters or phone reservations, are outdated and inefficient, causing inconvenience for both customers and theatre administrators. Customers face long queues, the risk of overbooking, and limited visibility into seat availability and movie schedules. Additionally, these systems often lack modern payment options, resulting in a time-consuming and unsatisfactory booking experience.

Theatre administrators struggle with managing bookings manually, leading to errors in seat assignments, scheduling conflicts, and difficulties in handling payment records. During peak demand, the absence of automation makes it challenging to scale operations efficiently, leading to revenue loss and customer dissatisfaction.

While some online systems offer partial solutions, they often fail to provide real-time updates, intuitive interfaces, or secure payment integration. Users also lack a seamless experience, from registration and login to seat reservation and payment completion.

The proposed **Movie Reservation System** aims to address these issues by providing an integrated, user-friendly platform for ticket booking. It will include features such as secure user authentication, real-time seat tracking, movie selection, and payment processing through gateways like Google Pay. By automating the process, the system will enhance user satisfaction and streamline theatre operations, ensuring accuracy, efficiency, and scalability.

1.2 OBJECTIVE OF THE PROJECT

This project focuses on developing a movie reservation system for a single theater with two halls, each screening the same movie across multiple time slots throughout the week. The system includes the following key features:

- **User Registration and Login:** Secure user authentication to ensure data privacy.
- **Movie Selection and Seat Availability:** Users can view movie schedules, choose time slots, and check available seats.
- **Booking and Payment:** Users can reserve seats and complete payments through an integrated payment gateway like Google Pay.
- **Database Management:** The system uses SQL to manage user data, movie schedules, and booking information.

  The project will be implemented using Java for backend processing, JDBC for database interaction, and HTML, CSS, and JavaScript for frontend development. The system is designed to be scalable, flexible, and adaptable to future enhancements, such as additional features or payment gateways.

1.3 ORGANIZATION OF THE REPORT

CHAPTER 1 - INTRODUCTION

CHAPTER 2 - SYSTEM DESIGN

CHAPTER 3 - IMPLEMENTATION

CHAPTER 4 – CONCLUSION

CHAPTER -2

SYSTEM FLOW DIAGRAMS

2.1 USE CASE DIAGRAM

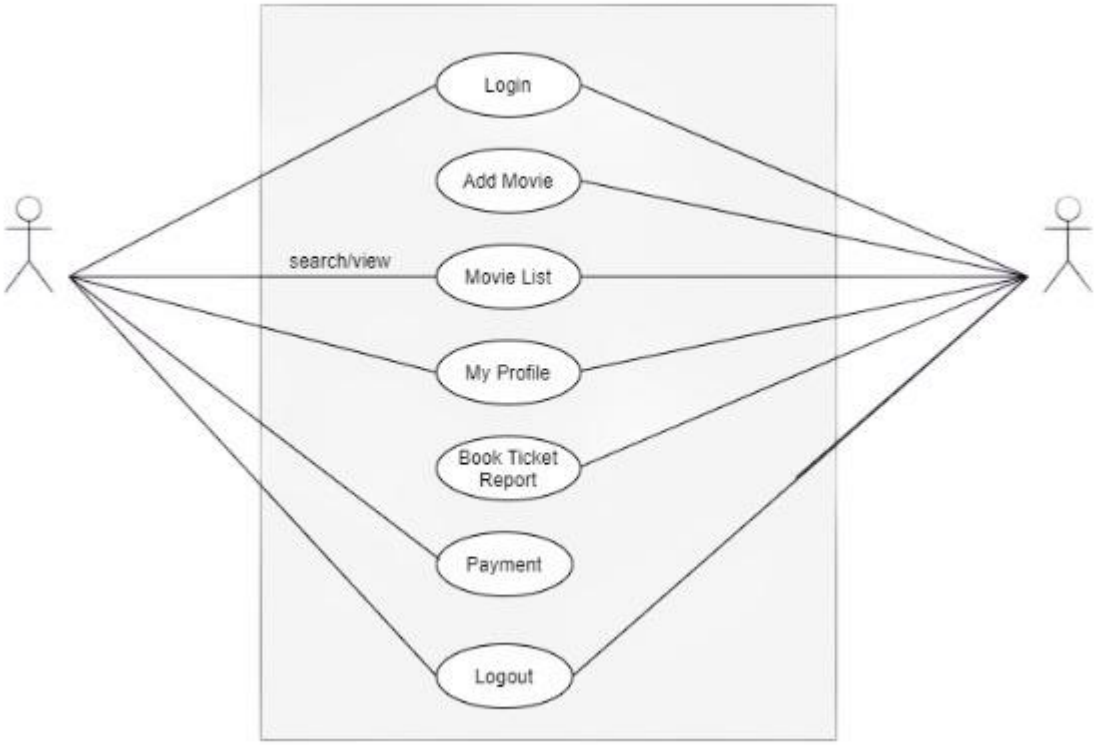

Figure 2.1. Use Case Diagram

2.2 ENTITY RELATIONSHIP DIAGRAM

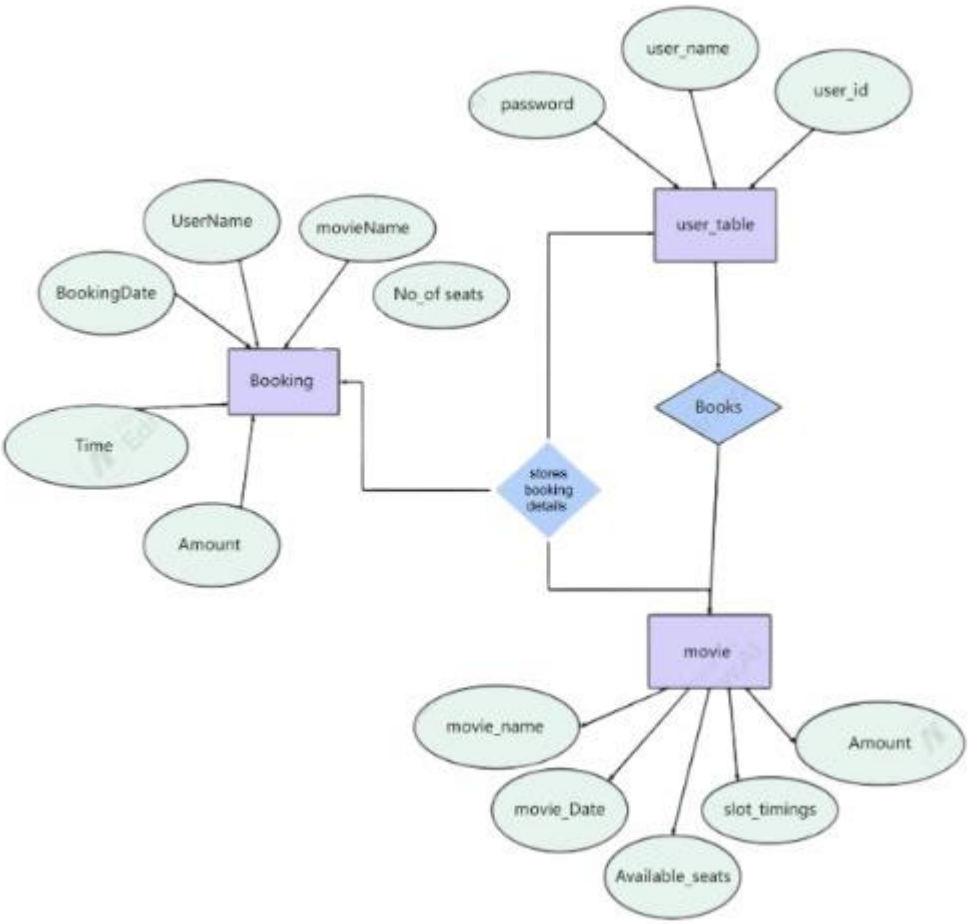

Figure 2.2. Entity -Relationship Diagram

2.3 DATA FLOW DIAGRAM

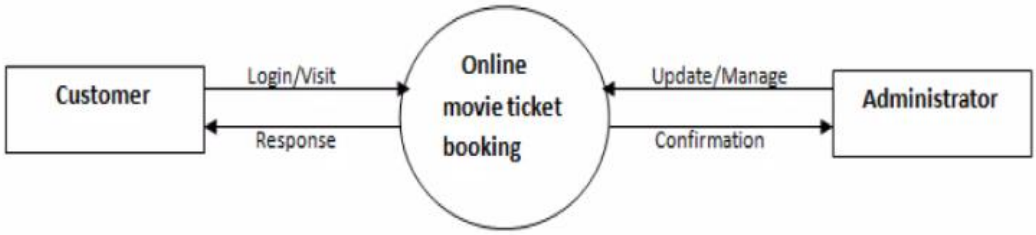

Figure 2.3. Data Flow Diagram

2.4. SYSTEM SPECIFICATION

The Movie Reservation System is a streamlined software solution for managing movie ticket bookings. Developed using JDBC, Java, SQL Plus (Oracle), HTML, CSS, and JavaScript, it ensures efficient functionality, secure data handling, and an intuitive user experience. Below are the specifications:

2.4.1 SOFTWARE REQUIREMENTS

1. **Programming Language:** Java (backend development).
2. **Database**: SQL Plus (Oracle) for storing user data, movie schedules, and bookings.
3. **Frontend Technologies:**
    - HTML5: Structure and layout.
    - CSS3: Styling and design.
    - JavaScript: Interactivity and dynamic content.
4. **Middleware:** JDBC for database connectivity.
5. **Browser Support:** Compatibility with modern web browsers**.**
6. **Operating System**: Platform-independent (Windows, macOS, Linux).

2.4.2 FUNCTIONAL REQUIREMENTS

1. **User Authentication:** Secure login and registration system.
2. **Movie Selection:** Display movies, time slots, and dates dynamically.
3. **Seat Reservation**: Real-time seat availability and selection.
4. **Payment Processing:** Calculate amount and integrate secure payment options.
5. **Booking Confirmation:** Generate and display booking details post-payment.

2.4.3 ADDITIONAL FEATURES

1. **Interactive Seat Layout**: Real-time updates for reserved and available seats.
2. **Responsive Design:** Frontend optimized for desktop and mobile devices.
3. **Scalability:** Designed to accommodate additional features or halls in the future.
4. **User Feedback:** Confirmation notifications upon successful booking.

CHAPTER 3

SYSTEM DESIGN

3.1 DESIGN

Figure 3.1.1 Home Page

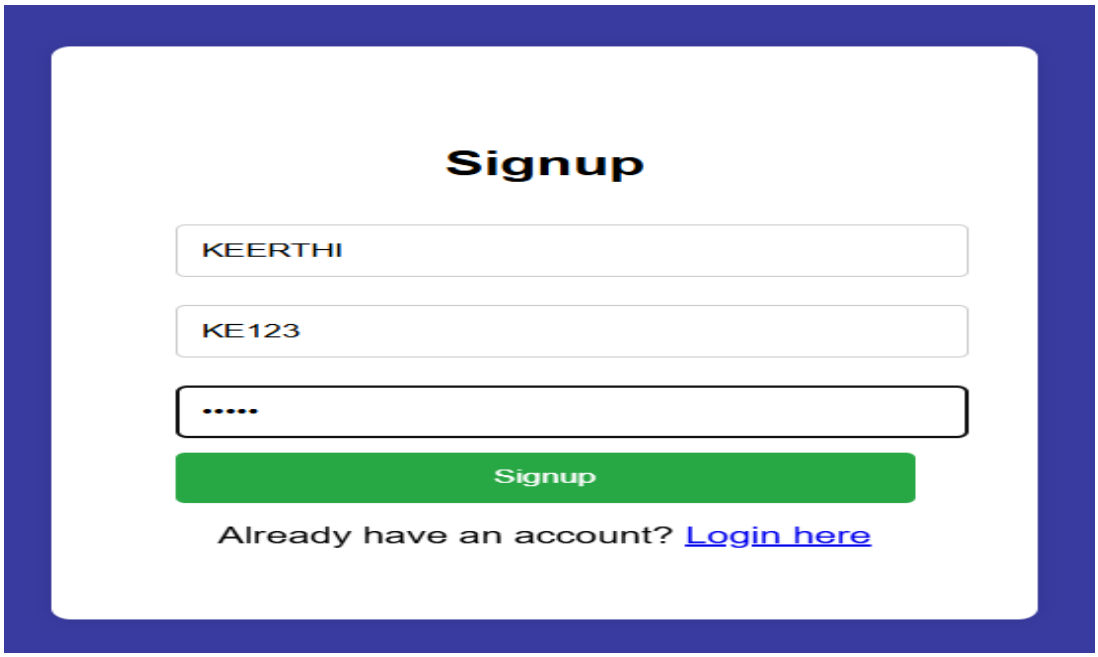

Figure 3.1.2 Signup Page

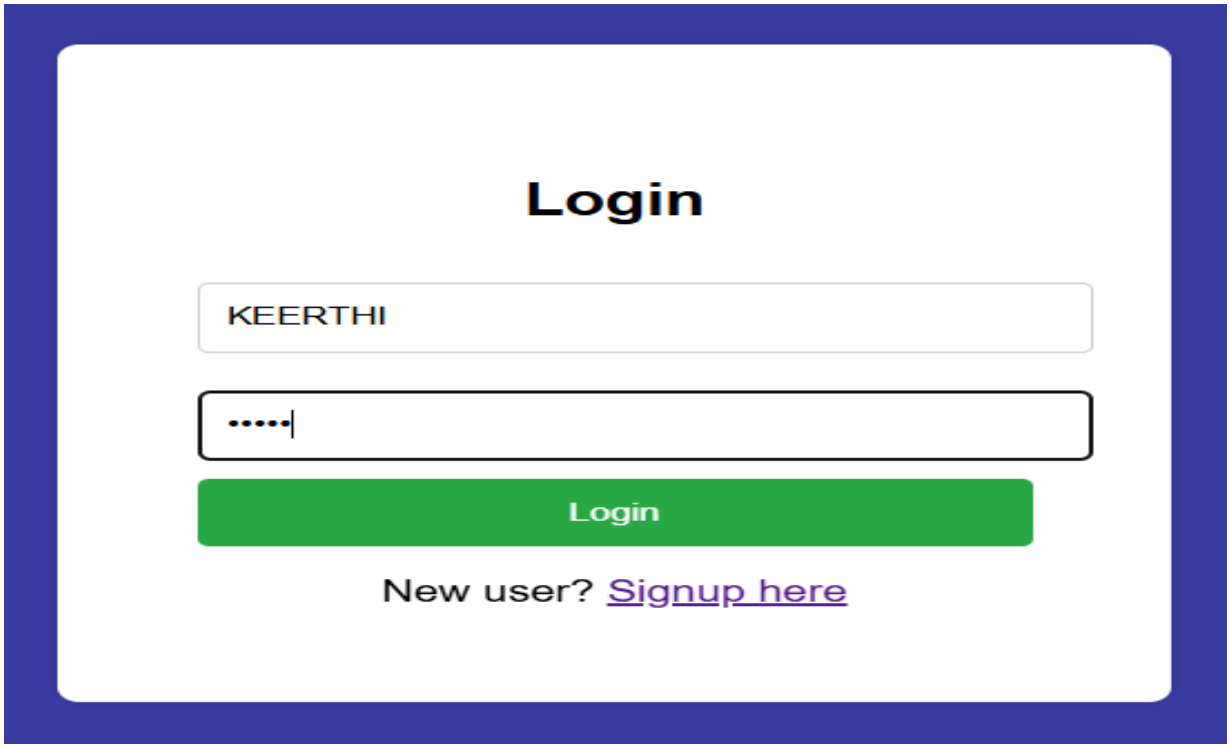

Figure 3.1.3 Login Page

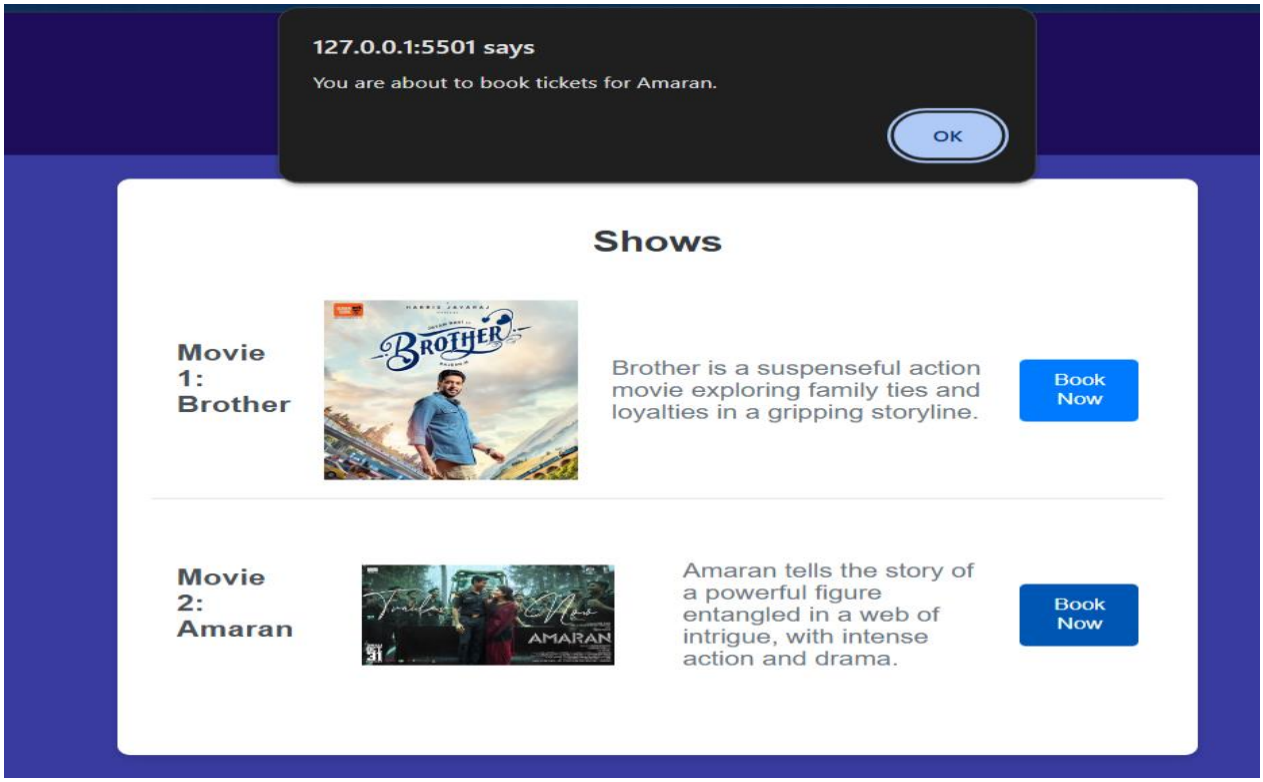

Figure 3.1.4 Movie Selection Page

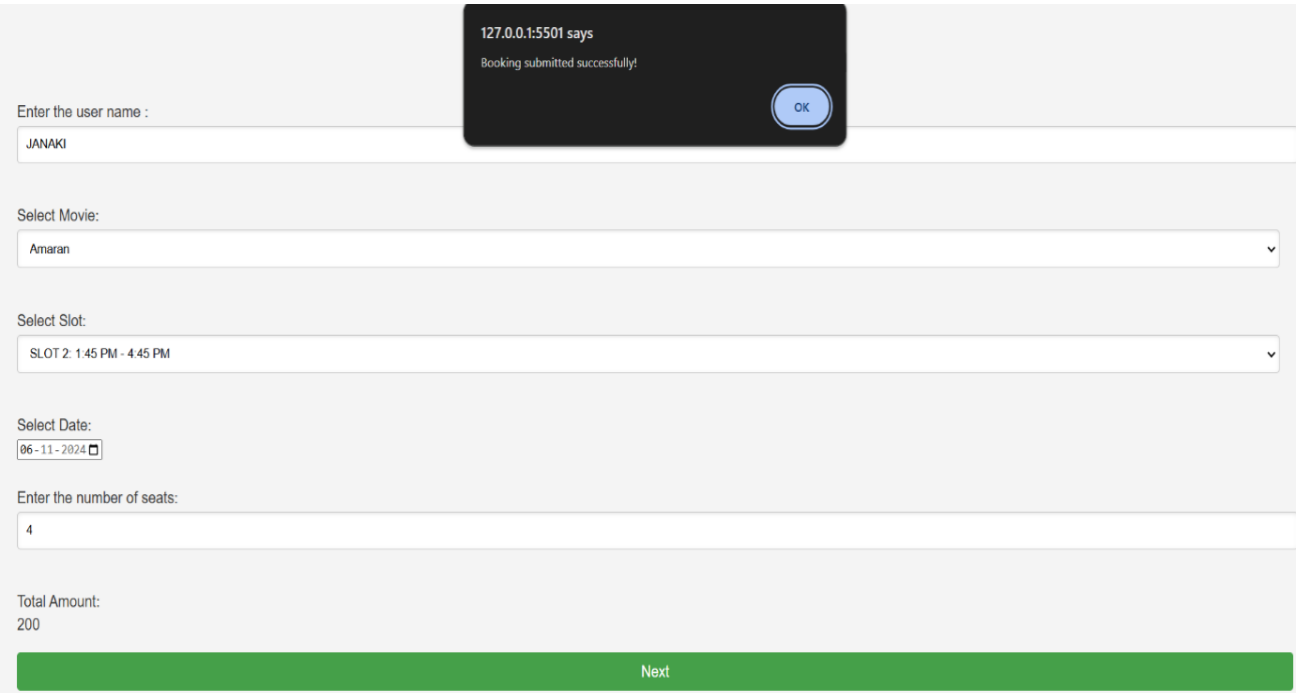

Figure 3.1.5 Slot Selection Page

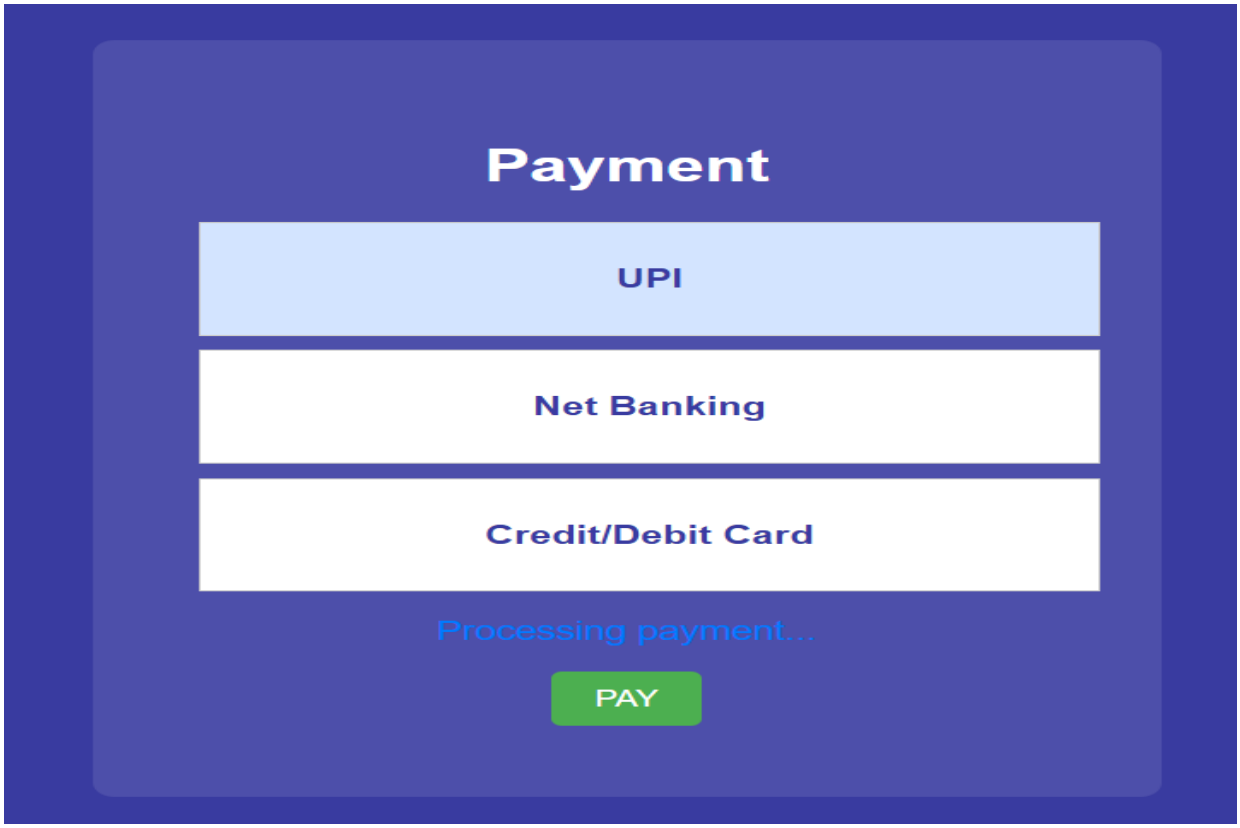

Figure 3.1.6 Payment Page

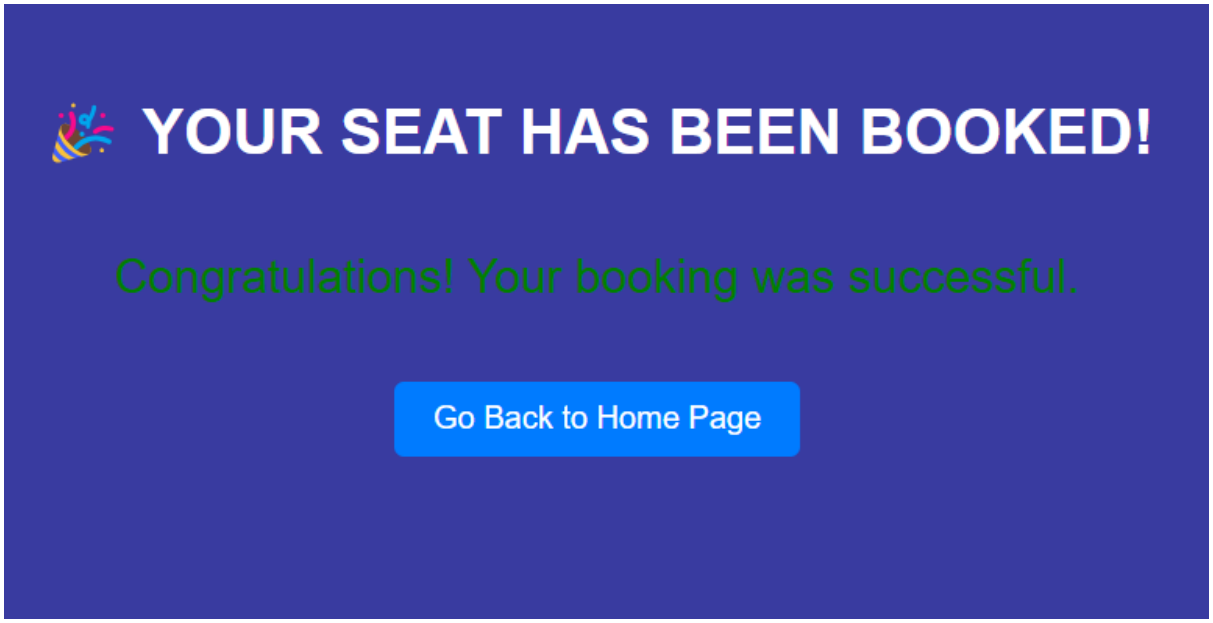

Figure 3.1.7 Payment Successful Page

3.2 DATABASE DESIGN

```
USER_
-----
USER_NAME
--------------------------------------------------------------------------------
PASSWORD
--------------------------------------------------------------------------------
JA123
JANAKI
JA123

KE123
KEERTHI
KE123

USER_
-----
USER_NAME
--------------------------------------------------------------------------------
PASSWORD
--------------------------------------------------------------------------------
```

Table 3.2.1 Users Table

```
 NOOFSEATS MOVIENAME                USERNAME                 BOOKINGDA
---------- ------------------------- ------------------------- ---------
TIME                              AMOUNT
------------------------------ ----------
         4 Amaran                   JANAKI                   06-NOV-24
1:45 PM - 4:45 PM                    200

         2 Amaran                   KEERTHI                  06-NOV-24
1:45 PM - 4:45 PM                    100
```

Table 3.2.2 Booking Table

```
SQL> select * from movie ;

MOVIE_NAME                                          MOVIE_DAT
--------------------------------------------------- ---------
SLOT_TIMINGS         AVAILABLE_SEATS     AMOUNT
-------------------- --------------- ----------
Brother                                             06-NOV-24
10:00 AM - 1:00 PM               50

Brother                                             06-NOV-24
1:45 PM - 4:45 PM                50

Brother                                             06-NOV-24
5:30 PM - 8:30 PM                50


MOVIE_NAME                                          MOVIE_DAT
--------------------------------------------------- ---------
SLOT_TIMINGS         AVAILABLE_SEATS     AMOUNT
-------------------- --------------- ----------
Brother                                             07-NOV-24
10:00 AM - 1:00 PM               50

Brother                                             07-NOV-24
1:45 PM - 4:45 PM                50

Brother                                             07-NOV-24
5:30 PM - 8:30 PM                50


MOVIE_NAME                                          MOVIE_DAT
--------------------------------------------------- ---------
SLOT_TIMINGS         AVAILABLE_SEATS     AMOUNT
-------------------- --------------- ----------
Amaran                                              06-NOV-24
10:00 AM - 1:00 PM               50

Amaran                                              06-NOV-24
1:45 PM - 4:45 PM                50
```

Figure 3.2.3 Movie Table

CHAPTER – 4

IMPLEMENTATION (CODE)

4.1.CONNECTION WITH SQL PLUS

```
public class SimpleHttpServer {
    private static final String URL = "jdbc:oracle:thin:@localhost:1521:XE";
    private static final String USERNAME = "system";
    private static final String PASSWORD = "it16";
    public static void main(String[] args) throws IOException {
        HttpServer server = HttpServer.create(new InetSocketAddress(8080), 0);
        server.createContext("/signup", SimpleHttpServer::handleSignup);
        server.createContext("/login", SimpleHttpServer::handleLogin);
        server.createContext("/bookSlot", SimpleHttpServer::handleBookSlot); // Added
booking handler
        server.setExecutor(null);
        server.start();
        System.out.println("Server started at http://localhost:8080");
    }}
```

4.2.BOOKING TABLE CREATION

```
public class BookingTableCreator {
   private static final String URL = "jdbc:oracle:thin:@localhost:1521:XE";
   private static final String USERNAME = "system";
   private static final String PASSWORD = "it16";
   public static void main(String[] args) {
      try (Connection connection = DriverManager.getConnection(URL, USERNAME,
PASSWORD);
          Statement statement = connection.createStatement()) {
         createBookingTable(statement);
      } catch (SQLException e) {
         e.printStackTrace();
      }
   }
```

4.3. METHOD TO CREATE THE BOOKING TABLE

```
   private static void createBookingTable(Statement statement) {
      try {
         String createTableSQL = "CREATE TABLE Booking ("
               + "NOOFSEATS int,"
               + "MOVIENAME VARCHAR2(25), "
               + "USERNAME VARCHAR2(25), "
               + "BOOKINGDATE date, "
```

```
                + "TIME VARCHAR(30),"
                + "AMOUNT INT"
            +")";
        statement.executeUpdate(createTableSQL);
        System.out.println("Table 'Booking' created successfully.");
    } catch (SQLException e) {
        if (e.getErrorCode() == 955) { // Error code 955 means table already exists in Oracle
            System.out.println("Table 'Booking' already exists.");
        } else {
            System.err.println("Error creating 'Booking' table: " + e.getMessage());
        }}}}
```

3.MOVIE TABLE CREATOR

```
// Method to create the Movie table
    private static void createMovieTable(Statement statement) {
        try {
        // SQL to create the Movie table
        String createTableSQL = "CREATE TABLE Movie ("
                + "MOVIE_NAME VARCHAR2(50), "
                + "MOVIE_DATE DATE, "
                + "SLOT_TIMINGS VARCHAR2(20), "
                + "AVAILABLE_SEATS NUMBER, "
                + "AMOUNT NUMBER"
                + ")";
```

```
            // Execute table creation
            statement.executeUpdate(createTableSQL);
            System.out.println("Table 'Movie' created successfully.");

        } catch (SQLException e) {
            // Check if the table already exists
            if (e.getErrorCode() == 955) { // Error code 955 means table already exists in Oracle
                System.out.println("Table 'Movie' already exists.");
            } else {
                System.err.println("Error creating 'Movie' table: " + e.getMessage());
            }
        }
    }

    // Method to insert data into the Movie table
    private static void insertMovieData(Connection connection, String movieName, String
movieDate, String slotTimings, int availableSeats) {
        String insertSQL = "INSERT INTO Movie (MOVIE_NAME, MOVIE_DATE,
SLOT_TIMINGS, AVAILABLE_SEATS) VALUES (?, TO_DATE(?, 'YYYY-MM-DD'), ?,
?)";
        try (PreparedStatement preparedStatement = connection.prepareStatement(insertSQL)) {
            preparedStatement.setString(1, movieName);
            preparedStatement.setString(2, movieDate);
            preparedStatement.setString(3, slotTimings);
```

```
            preparedStatement.setInt(4, availableSeats);
            preparedStatement.executeUpdate();
            System.out.println("Inserted data for movie: " + movieName + ", date: " + movieDate
+ ", slot: " + slotTimings);
        } catch (SQLException e) {
            System.err.println("Error inserting data for movie: " + movieName + " on date: " +
movieDate + " for slot: " + slotTimings);
            e.printStackTrace();
        }
    }
}
```

4.4.USER TABLE CREATOR

```
class UserTableCreator {
    private static final String URL = "jdbc:oracle:thin:@localhost:1521:XE"; // Database URL
    private static final String USERNAME = "system"; // Database username
    private static final String PASSWORD = "it16"; // Database password
    public static void main(String[] args) {
        try (Connection connection = DriverManager.getConnection(URL, USERNAME,
PASSWORD);
             Statement statement = connection.createStatement()) {
            createUserTable(statement); // Call the method to create the user table
        } catch (SQLException e) {
            e.printStackTrace(); // Print any SQL exceptions
        }
    }
```

```
 // Method to create the users_table
   private static void createUserTable(Statement statement) {
      try {
         String createTableSQL = "CREATE TABLE users_table (" +
               "user_id varchar(5)  PRIMARY KEY, " +
               "user_name VARCHAR2(100) NOT NULL, " +
               "password VARCHAR2(100) NOT NULL" +
               ")";

         statement.executeUpdate(createTableSQL); // Execute the SQL statement to create the table
         System.out.println("User table created successfully."); // Confirmation message

      } catch (SQLException e) {
         if (e.getErrorCode() == 955) { // Error code 955 means table already exists in Oracle
            System.out.println("User table already exists."); // Message for existing table
         } else {
            System.err.println("Error creating 'users_table': " + e.getMessage()); // Error message for other exceptions
         }
      }
   }
}
```

CONCLUSION

The Movie Reservation System is a user-friendly and secure application that streamlines movie ticket booking by integrating seamless user authentication, interactive seat reservation, and secure payment options like Google Pay. The system ensures real-time updates, responsive performance, and a smooth user experience. By automating processes, minimizing manual errors, and offering scalability for future growth, it addresses the challenges of traditional booking methods, enhancing customer satisfaction while supporting efficient theatre management.

REFERENCE

1.Movie Ticket Reservation System

https://1000projects.org/movie-ticket-booking-reservation-system-java-mysql-project-source-code-report.html

2. Movie Ticket Reservation System in Java (Full Project)

Movie Ticket Reservation System Java Project - YouTube

3. Java Swing Movie Ticket Booking System Project

Java Swing Movie Ticket Booking System - YouTube

4. Java Project - Online Movie Ticket Reservation System (with MySQL)

Online Movie Ticket Reservation System - YouTube