

**Title: Online Movie Ticket Booking System**

**A CAPSTONE PROJECT REPORT**

**Submitted by**

**<J. Harsha Vardhan Reddy> (192210422)**

**CSE Department.**

In Partial Fulfillment for the completion of the course

CSA0912-PROGRAMMING IN JAVA FOR ACCESSING DATABASE

**SEPTEMBER -2024**



SAVEETHA INSTITUTE OF MEDICAL AND TECHNICAL SCIENCES

CHENNAI - 602105

TAMIL NADU, INDIA



# **BONAFIDE CERTIFICATE**

This is to certify that the project report entitled **Online Movie Ticket Booking System** submitted by **J. Harsha Vardhan Reddy (192210422)** to Saveetha School of Engineering, Saveetha Institute of Medical and Technical Sciences, Chennai, is a record of bonafide work carried out by him/her under my guidance. The project fulfills the requirements as per the regulations of this institution and in my appraisal meets the required standards for submission.

Dr.K.Jayasakthi Velmurugan

**COURSE FACULTY**

*Department of Deep Learning.*

*Saveetha School of Engineering,*

*SIMATS, Chennai - 602105* **ACKNOWLEDGEMENT**

This project work would not have been possible without the contribution of many people. It gives me immense pleasure to express my profound gratitude to our Honorable Chancellor **Dr. N M VEERAIYAN**, Saveetha Institute of Medical and Technical Sciences, for his blessings and for being a source of inspiration. I sincerely thank our Director of Academics **Dr. DEEPAK NALLASWAMY,** SIMATS, for his visionary thoughts and support. I am indebted to extend my gratitude to our Director **Dr. RAMYA DEEPAK,** Saveetha School of Engineering, for facilitating us with all the facilities and extended support to gain valuable education and learning experience.

I register my special thanks to **Dr. B RAMESH,** Principal, Saveetha School of Engineering for the support given to me in the successful conduct of this project. I wish to express my sincere gratitude to my Course faculty **Dr.K.Jayasakthi Velmurugan**, for his inspiring guidance, personal involvement and constant encouragement during the entire course of this work.

I am grateful to Project Coordinators, Review Panel External and Internal Members and the entire faculty of the Department of Design, for their constructive criticisms and valuable suggestions which have been a rich source to improve the quality of this work.

**INDEX**

|  |  |
| --- | --- |
| ABSTRACT | 4 |
| INTRODUCTION | 5 |
| ARCHITECTURE DIAGRAM | 6 |
| FLOWCHART | 7 |
| UML DIAGRAM | 8 |
| CLASS DIAGRAM | 8 |
| CODE IMPLEMENTATION | 9-14 |
| OUTPUT SCREENSHOT | 14-15 |
| CONCLUSION | 16 |
| REFERENCES | 17 |

**1.ABSTRAC****T**

Online Movie Ticket Booking System is a comprehensive web application designed to simplify the movie ticket booking process for users while providing robust management tools for administrators. The system allows customers to browse a variety of movies, select their desired showtimes, and book tickets easily. Features include user registration, account management, secure payment integration, and a user-friendly interface that enhances the overall booking experience. For administrators, the system provides features to manage movie listings, track bookings, generate sales reports, and manage user accounts efficiently. The application uses AWS services such as RDS for database management, S3 for media storage, and Cognito for user authentication to ensure a scalable and secure environment. The responsive design caters to both desktop and mobile users, making it accessible to everyone. The system not only simplifies the ticket booking process but also enriches the user experience, thereby increasing engagement and satisfaction of cinema visits.

The application is integrated with Amazon Web Services (AWS) to support the backend using RDS for database management, S3 for media storage, and Cognito for user authentication. This cloud infrastructure provides increased security and scalability, ensuring a reliable experience for users. The online movie ticket booking system aims to revolutionize the way audiences interact with movies by providing a seamless and enjoyable movie-watching experience.

Furthermore, the system is designed to be mobile-responsive, accommodating users on various devices and screen sizes, thus expanding its reach. By combining modern technologies with user-centric design, the Online Movie Ticket Booking System aims to revolutionize how moviegoers purchase tickets, ultimately leading to increased customer satisfaction and enhanced operational efficiency for cinemas.

**2. INTRODUCTION**

In today's fast-paced digital world, convenience and efficiency are paramount, especially in the entertainment industry. Online movie ticket booking systems are innovative solutions designed to meet the changing needs of movie lovers and cinema operators. As online transactions continue to grow in popularity, this system allows users to easily purchase movie tickets from the comfort of their own home or on the go, eliminating the traditional hassle of in-person bookings.

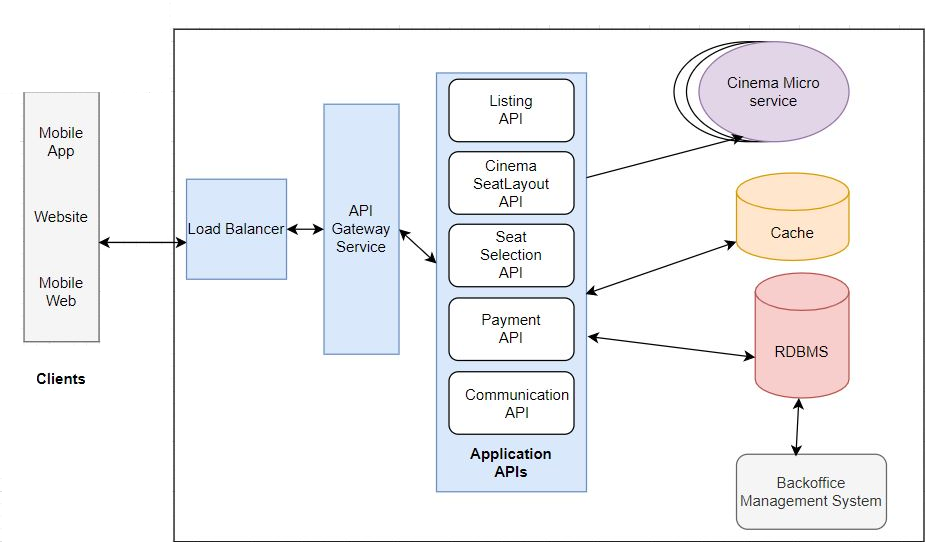
The system provides a user-friendly interface that allows customers to view a wide range of movies currently showing in various theaters. Users can easily browse movie listings, watch trailers, read synopses, check showtimes, and make informed viewing decisions. This level of accessibility not only improves the user experience, but also increases cinema attendance by making the booking process more engaging.

From the cinema operator's perspective, the system provides reliable management. Admins can manage movie listings including showtimes, pricing, availability updates, etc. The ability to generate real-time reports on ticket sales and user activity provides valuable information to help develop marketing strategies and make operational decisions. Moreover, online movie ticket booking systems are designed to evolve and adapt to changing market dynamics, and using cloud technologies such as AWS ensures that the system can handle fluctuations in user traffic without compromising performance. This scalability is crucial during peak periods, such as blockbuster movie releases or holiday weekends, when demand for tickets increases.

The system is also designed with mobile responsiveness in mind, recognizing the increasing number of users accessing services via smartphones and tablets. This mobile-friendly approach improves the overall user experience and makes it easier for customers to book tickets anytime, anywhere.

In summary, online movie ticket booking is revolutionizing the way movie tickets are purchased and managed, meeting the needs of both consumers and cinema operators. Combining user-centric design and cutting-edge technology, the system aims to simplify the ticket booking process, increase interest in cinemas' services and ultimately enhance the overall movie-going experience.

**3. ARCHITECTURE DIAGRAM**



# 

**4.FLOW CHART**

**Booking Confirmation**

Payment Process

Select

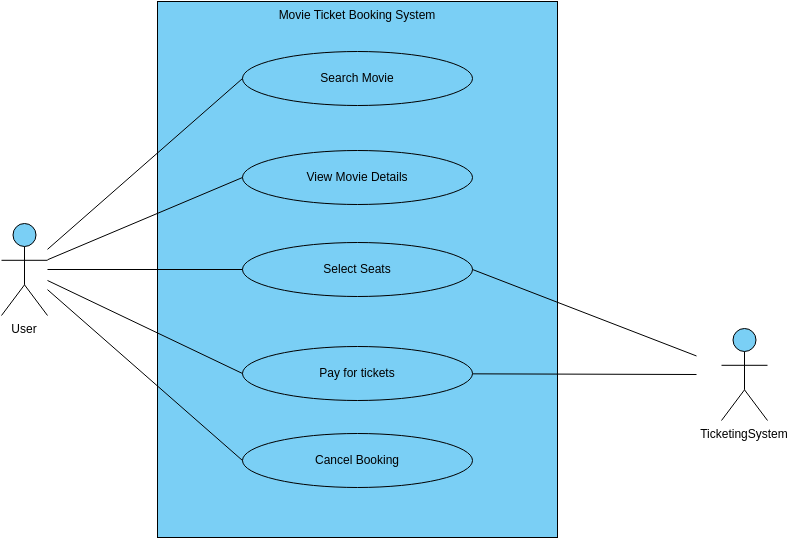
Movie

Select Movie

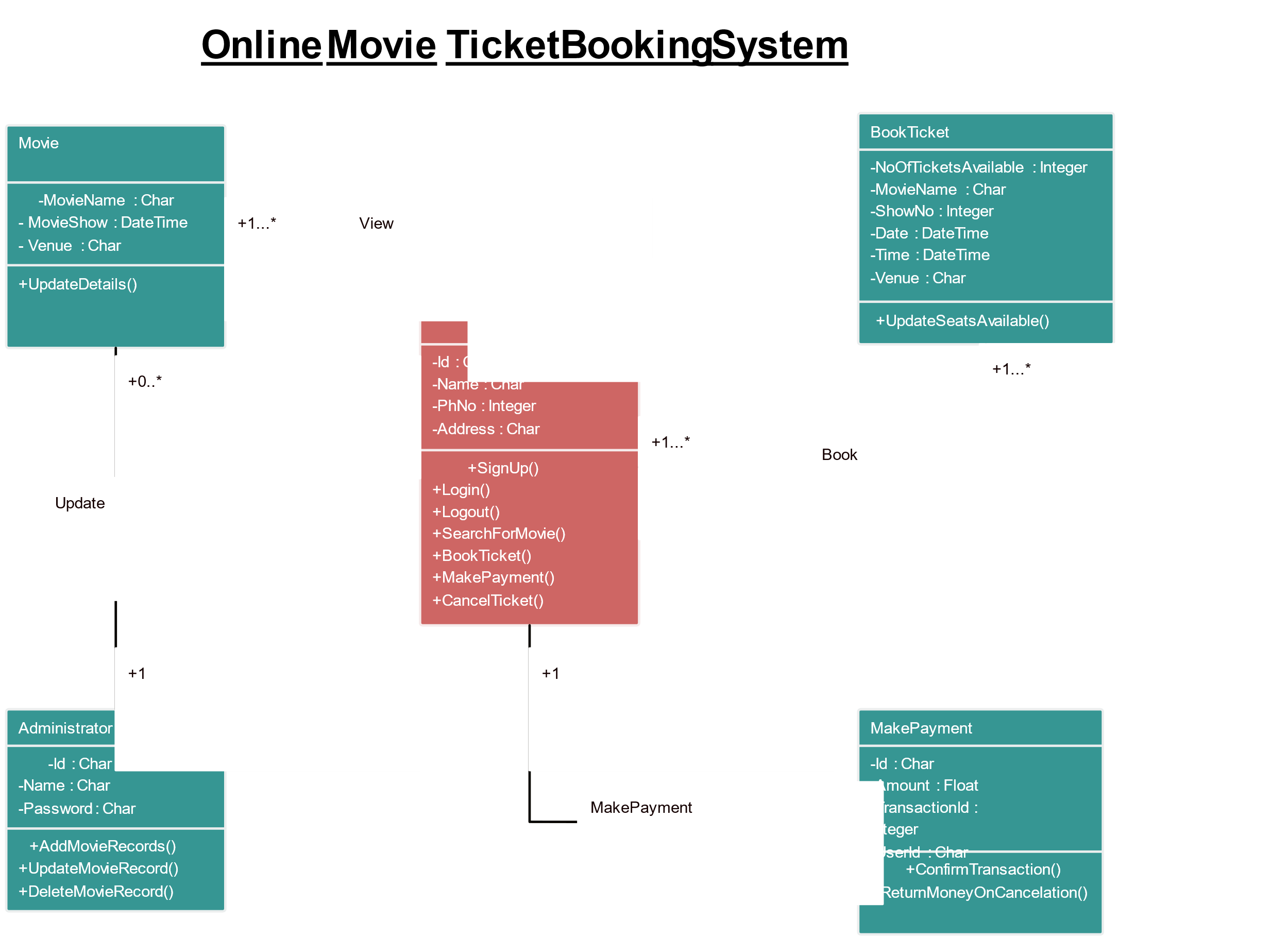
Choose Showtime

Select Seats

**5.UML DIAGRAM**



**6.CLASS DIAGRAM**

****

**7.Code Implementation**

**1. Login**

import java.sql.\*;

import java.security.MessageDigest;

import java.security.NoSuchAlgorithmException;

import java.util.Scanner;

public class LoginSystem {

private static final String DB\_URL = "jdbc:mysql://your-db-url:3306/your\_db\_name";

private static final String USER = "your\_db\_user";

private static final String PASSWORD = "your\_db\_password";

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

int choice;

do {

System.out.println("1. Register");

System.out.println("2. Login");

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

System.out.print("Enter your choice: ");

choice = scanner.nextInt();

scanner.nextLine();

switch (choice) {

case 1:

registerUser(scanner);

break;

case 2:

loginUser(scanner);

break;

case 3:

System.out.println("Exiting...");

break;

default:

System.out.println("Invalid choice. Please try again.");

}

} while (choice != 3);

scanner.close();

}

private static void registerUser(Scanner scanner) {

System.out.print("Enter username: ");

String username = scanner.nextLine();

System.out.print("Enter password: ");

String password = scanner.nextLine();

String hashedPassword = hashPassword(password);

try (Connection conn = DriverManager.getConnection(DB\_URL, USER, PASSWORD)) {

String sql = "INSERT INTO users (username, password) VALUES (?, ?)";

PreparedStatement pstmt = conn.prepareStatement(sql);

pstmt.setString(1, username);

pstmt.setString(2, hashedPassword);

pstmt.executeUpdate();

System.out.println("Registration successful!");

} catch (SQLException e) {

System.out.println("Error: " + e.getMessage());

}

}

private static void loginUser(Scanner scanner) {

System.out.print("Enter username: ");

String username = scanner.nextLine();

System.out.print("Enter password: ");

String password = scanner.nextLine();

String hashedPassword = hashPassword(password);

try (Connection conn = DriverManager.getConnection(DB\_URL, USER, PASSWORD)) {

String sql = "SELECT \* FROM users WHERE username = ? AND password = ?";

PreparedStatement pstmt = conn.prepareStatement(sql);

pstmt.setString(1, username);

pstmt.setString(2, hashedPassword);

ResultSet rs = pstmt.executeQuery();

if (rs.next()) {

System.out.println("Login successful! Welcome " + username);

} else {

System.out.println("Invalid username or password.");

}

} catch (SQLException e) {

System.out.println("Error: " + e.getMessage());

}

}

private static String hashPassword(String password) {

try {

MessageDigest md = MessageDigest.getInstance("SHA-256");

byte[] hash = md.digest(password.getBytes());

StringBuilder hexString = new StringBuilder();

for (byte b : hash) {

String hex = Integer.toHexString(0xff & b);

if (hex.length() == 1) hexString.append('0');

hexString.append(hex);

}

return hexString.toString();

} catch (NoSuchAlgorithmException e) {

throw new RuntimeException(e);

}

}

}

**2.Customer Dashboard**

import java.util.Scanner;

import java.util.HashMap;

import java.util.Map;

public class CustomerDashboard extends JFrame {

private Map<String, String> userDatabase;

private JPanel loginPanel, dashboardPanel;

private JTextField usernameField;

private JPasswordField passwordField;

private JComboBox<String> movieDropdown;

private JComboBox<String> showtimeDropdown;

private JButton bookTicketButton;

private JButton makePaymentButton;

private JTextField cardNumberField;

private JTextField nameOnCardField;

private JTextField expiryDateField;

private JTextField cvvField;

public CustomerDashboard() {

userDatabase = new HashMap<>();

userDatabase.put("customer1", "password1");

userDatabase.put("customer2", "password2");

setTitle("Online Movie Ticket Booking System");

setSize(500, 600);

setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

setLayout(new CardLayout());

loginPanel = new JPanel();

loginPanel.setLayout(new GridLayout(3, 2));

usernameField = new JTextField();

passwordField = new JPasswordField();

JButton loginButton = new JButton("Login");

loginPanel.add(new JLabel("Username:"));

loginPanel.add(usernameField);

loginPanel.add(new JLabel("Password:"));

loginPanel.add(passwordField);

loginPanel.add(loginButton);

add(loginPanel);

dashboardPanel = new JPanel();

dashboardPanel.setLayout(new GridLayout(6, 2));

movieDropdown = new JComboBox<>(new String[] { "Movie A", "Movie B", "Movie C" });

showtimeDropdown = new JComboBox<>(new String[] { "10:00 AM", "1:00 PM", "4:00 PM", "7:00 PM" });

bookTicketButton = new JButton("Book Ticket");

makePaymentButton = new JButton("Make Payment");

dashboardPanel.add(new JLabel("Select Movie:"));

dashboardPanel.add(movieDropdown);

dashboardPanel.add(new JLabel("Select Showtime:"));

dashboardPanel.add(showtimeDropdown);

dashboardPanel.add(new JLabel());

dashboardPanel.add(bookTicketButton);

dashboardPanel.add(new JLabel());

dashboardPanel.add(makePaymentButton);

dashboardPanel.add(new JLabel("Card Number:"));

loginButton.addActionListener(e -> authenticateUser());

bookTicketButton.addActionListener(e -> bookTicket());

makePaymentButton.addActionListener(e -> processPayment());

add(loginPanel, "Login");

add(dashboardPanel, "Dashboard");

setVisible(true);

}

private void authenticateUser() {

String username = usernameField.getText();

String password = new String(passwordField.getPassword());

if (userDatabase.containsKey(username) && userDatabase.get(username).equals(password)) {

switchToDashboard();

} else {

JOptionPane.showMessageDialog(this, "Invalid username or password!");

}

}

private void switchToDashboard() {

loginPanel.setVisible(false);

dashboardPanel.setVisible(true);

setContentPane(dashboardPanel);

validate();

repaint();

}

private void bookTicket() {

String selectedMovie = (String) movieDropdown.getSelectedItem();

String selectedShowtime = (String) showtimeDropdown.getSelectedItem();

JOptionPane.showMessageDialog(this, "Ticket booked for " + selectedMovie + " at " + selectedShowtime + ". Proceed to payment.");

}

private void processPayment() {

String cardNumber = cardNumberField.getText();

String nameOnCard = nameOnCardField.getText();

String expiryDate = expiryDateField.getText();

String cvv = cvvField.getText();

if (cardNumber.isEmpty() || nameOnCard.isEmpty() || expiryDate.isEmpty() || cvv.isEmpty()) {

JOptionPane.showMessageDialog(this, "Please fill in all payment details!");

} else {

JOptionPane.showMessageDialog(this, "Payment successful! Thank you for booking.");

}

}

public static void main(String[] args) {

new CustomerDashboard();

}

}

**3.Payment**

import java.util.Scanner;

class Payment {

private String cardHolderName;

private String cardNumber;

private String expiryDate;

private String cvv;

private double amount;

public Payment(String cardHolderName, String cardNumber, String expiryDate, String cvv, double amount) {

this.cardHolderName = cardHolderName;

this.cardNumber = cardNumber;

this.expiryDate = expiryDate;

this.cvv = cvv;

this.amount = amount;

}

public boolean validatePaymentDetails() {

if (cardNumber.length() == 16 && cvv.length() == 3) {

System.out.println("Payment details validated.");

return true;

} else {

System.out.println("Invalid payment details.");

return false;

}

}

public boolean processPayment() {

if (validatePaymentDetails()) {

System.out.println("Payment of $" + amount + " has been processed successfully for " + cardHolderName + ".");

return true;

} else {

System.out.println("Payment failed. Please check the card details and try again.");

return false;

}

}

}

**4.Confirmation**

import java.util.\*;

public class ConfirmationPage extends JFrame {

private String username;

private String movieName;

private String showtime;

private int tickets;

public ConfirmationPage(String username, String movieName, String showtime, int tickets) {

this.username = username;

this.movieName = movieName;

this.showtime = showtime;

this.tickets = tickets;

initComponents();

}

private void initComponents() {

JLabel confirmationLabel = new JLabel("Booking Confirmation");

confirmationLabel.setFont(new Font("Arial", Font.BOLD, 24));

JLabel usernameLabel = new JLabel("Username: " + username);

JLabel movieLabel = new JLabel("Movie: " + movieName);

JLabel showtimeLabel = new JLabel("Showtime: " + showtime);

JLabel ticketsLabel = new JLabel("Tickets: " + tickets);

JButton printTicketButton = new JButton("Print Ticket");

printTicketButton.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

}

});

JButton backButton = new JButton("Back to Dashboard");

backButton.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

}

});

JPanel panel = new JPanel();

panel.setLayout(new BoxLayout(panel, BoxLayout.Y\_AXIS));

panel.add(confirmationLabel);

panel.add(usernameLabel);

panel.add(movieLabel);

panel.add(showtimeLabel);

panel.add(ticketsLabel);

panel.add(printTicketButton);

panel.add(backButton);

add(panel);

setSize(400, 300);

setDefaultCloseOperation(EXIT\_ON\_CLOSE);

setVisible(true);

}

public static void main(String[] args) {

String username = "johnDoe";

String movieName = "Avengers";

String showtime = "10:00 AM";

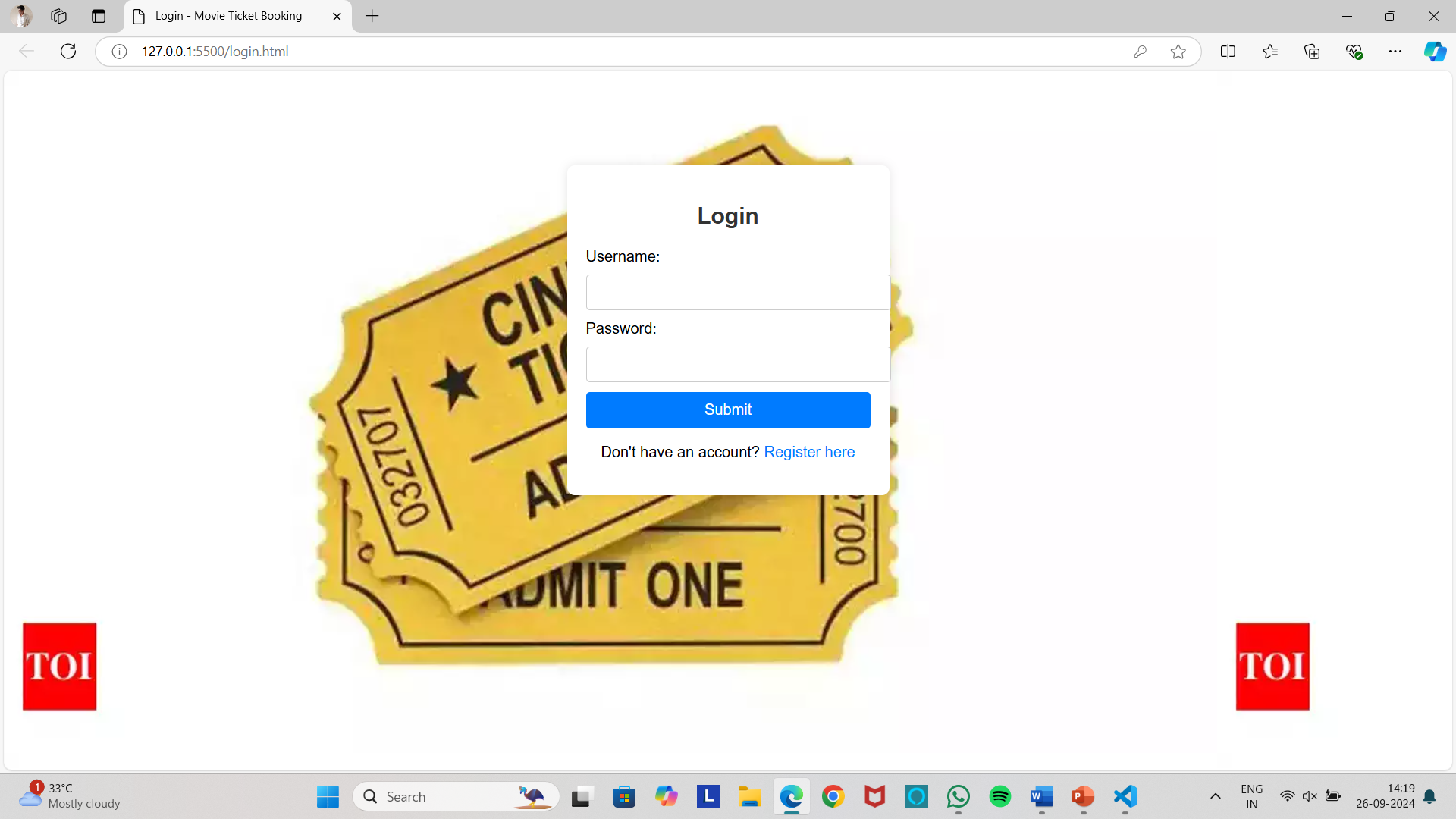
int tickets = 2;

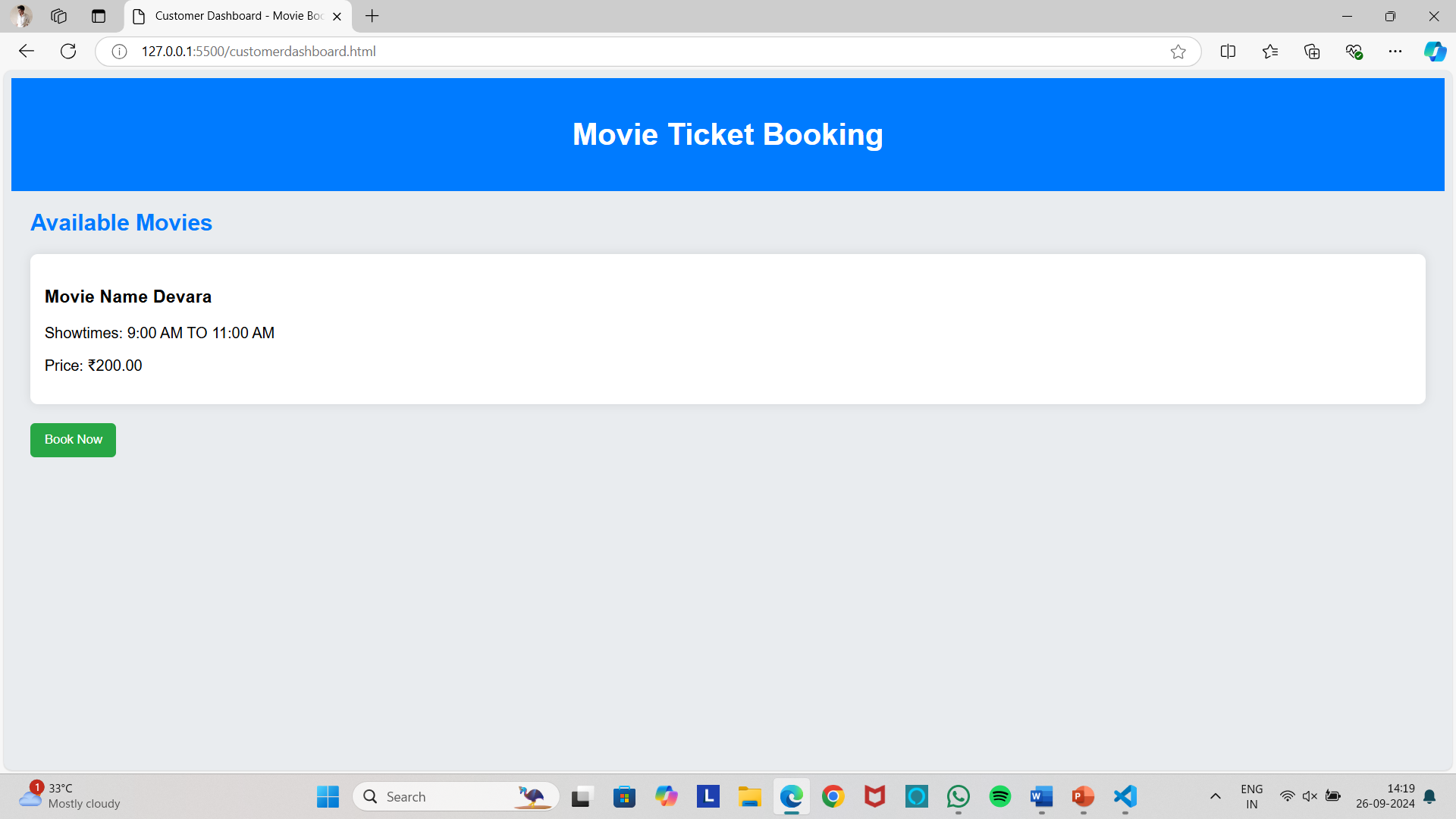
ConfirmationPage confirmationPage = new ConfirmationPage(username, movieName, showtime, tickets);

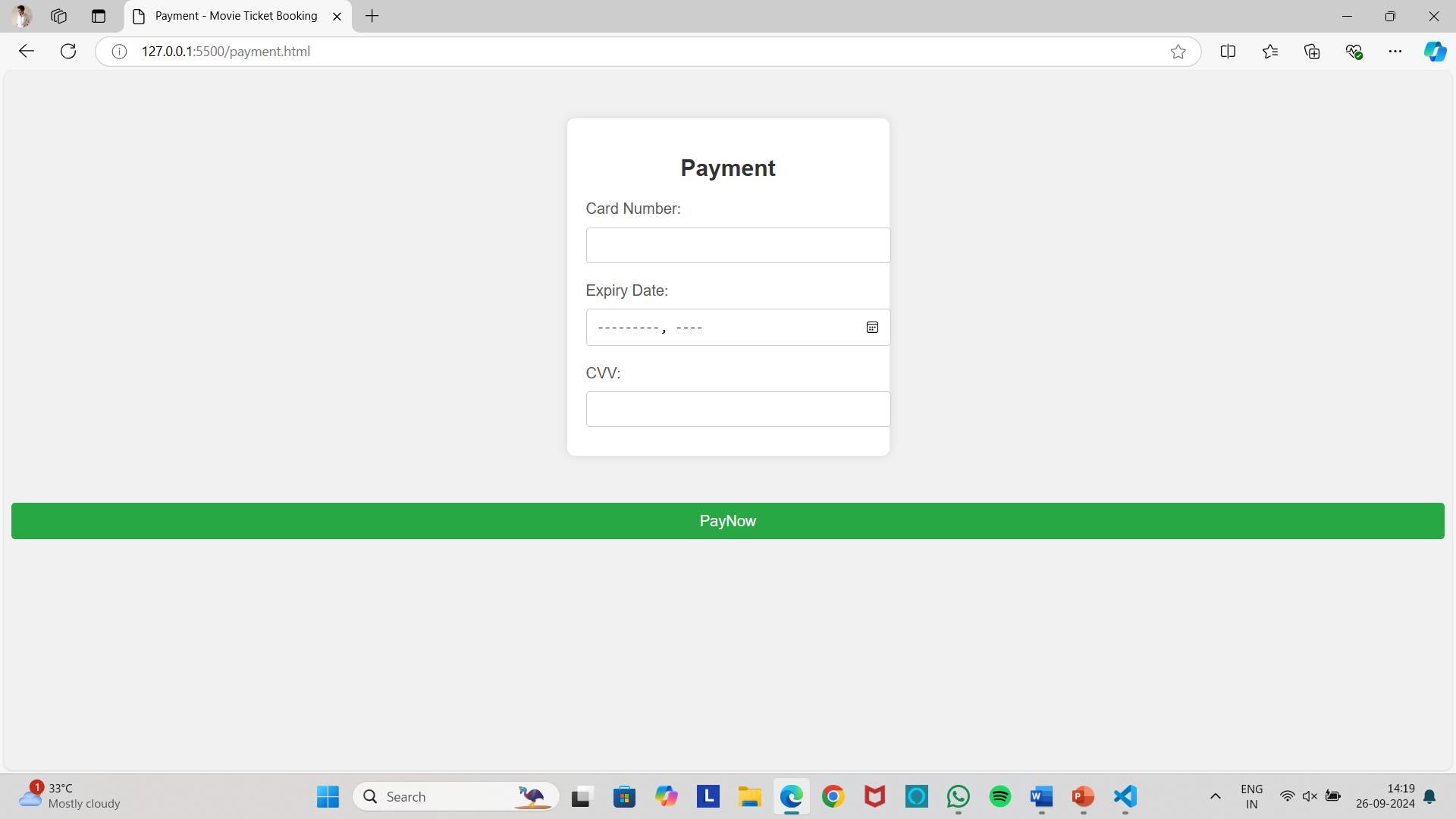
}

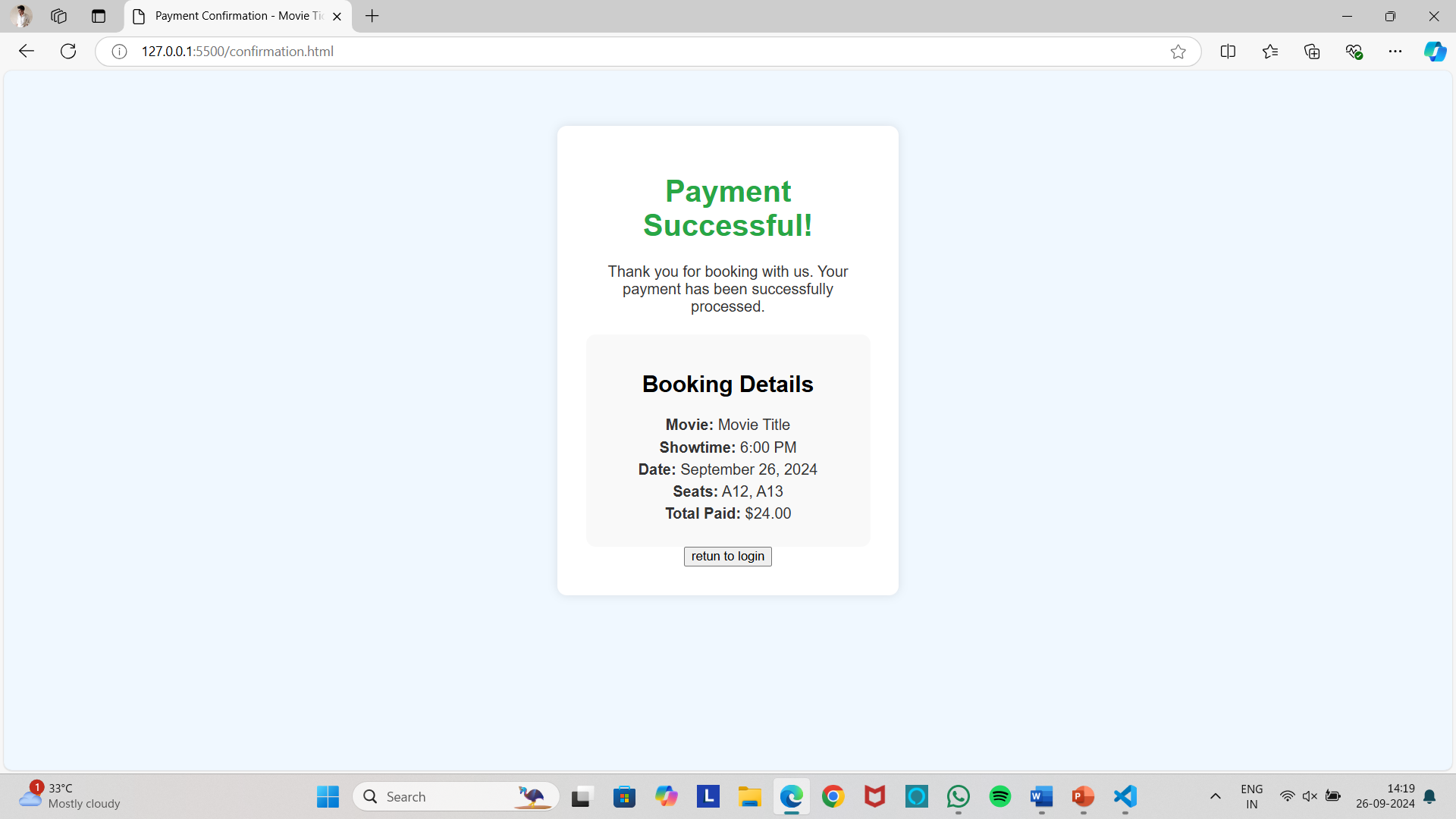
}

**8. OUTPUT SCREENSHOT**

****



****



**9. CONCLUSION**

The Online movie ticket booking system is a modern, efficient and convenient solution for purchasing movie tickets and managing cinema operations. The system enhances the entire cinema experience by allowing users to easily browse movies, choose showtimes, select seats and make secure payments. Integrating cloud technologies such as AWS provides the scalability, reliability, and security to meet customer and administrator needs during peak times.

For administrators, the system simplifies the process of managing movie listings, tracking reservations, and generating real-time reports, providing valuable insights into customer behavior and performance. Mobile responsiveness, cross-device compatibility, and robust backend infrastructure ensure that users can access your services anytime, anywhere.

**10. REFERENCES**

1. GitHub Repositories: Explore various open-source projects related to movie ticket booking systems on GitHub. [These repositories often include code, documentation, and implementation details1](https://github.com/topics/movie-ticket-booking-system).
2. [GeeksforGeeks: This article provides a comprehensive guide on designing a movie ticket booking system, including requirements, architecture, and implementation details2](https://www.geeksforgeeks.org/design-movie-ticket-booking-system-like-bookmyshow/).
3. [Filmgrail Blog: Learn about the essentials of an online cinema booking system, including features like browsing showtimes, seat selection, and payment processing3](https://filmgrail.com/blog/online-cinema-booking-system-essentials/).
4. BookMyShow: One of the most popular platforms for booking movie tickets in India. [It offers a user-friendly interface and integrates well with mobile apps4](https://filmgrail.com/blog/cinema-reservation-system-essentials/).
5. Research Papers: Look for academic papers on digital ticketing systems and their impact on user experience and business operations.
6. API Documentation: Check out APIs like TMDB (The Movie Database) for integrating movie data into your booking system.
7. Payment Gateway Integration: Learn about integrating payment gateways like PayPal, Stripe, or Razorpay for secure transactions.
8. User Authentication: Implement secure user authentication using OAuth or JWT (JSON Web Tokens).
9. Real-time Updates: Use WebSockets or similar technologies to provide real-time updates on seat availability.
10. UI/UX Design: Focus on creating an intuitive and responsive user interface, possibly using frameworks like React or Angular.
11. Database Management: Understand how to manage and optimize databases for storing movie schedules, user information, and booking details.
12. Security Best Practices: Learn about securing your application against common threats like SQL injection, cross-site scripting (XSS), and cross-site request forgery (CSRF).
13. Scalability: Explore techniques for scaling your application to handle high traffic, such as load balancing and microservices architecture.
14. Performance Optimization: Study methods to optimize the performance of your application, including caching strategies and efficient query handling.