MOVIE TICKET BOOKING SYSTEM

A PROJECT REPORT

Submitted by

KETHAAR ESHWAR A (2303811710421079)

in partial fulfillment of requirements for the award of the course

CGB1201 - JAVA PROGRAMMING

In

COMPUTER SCIENCE AND ENGINEERING

K. RAMAKRISHNAN COLLEGE OF TECHNOLOGY

(An Autonomous Institution, affiliated to Anna University Chennai and Approved by AICTE, New Delhi)

SAMAYAPURAM – 621 112

NOVEMBER- 2024

K. RAMAKRISHNAN COLLEGE OF TECHNOLOGY (AUTONOMOUS)

SAMAYAPURAM – 621 112

BONAFIDE CERTIFICATE

Certified that this project report on **"MOVIE TICKET BOOKING SYSTEM"** is the bonafide work of **KETHAAR ESHWAR A (2303811710421079)** who carried out the project work during the academic year 2024 - 2025 under my supervision.

CGB1201-JAVA PROGRAMMING
Dr.A.DELPHIN CAROLINA RANI, M.E.,Ph.D.,
HEAD OF THE DEPARTMENT
PROFESSOR

SIGNATURE

Dr.A.Delphin Carolina Rani, M.E.,Ph.D.,

HEAD OF THE DEPARTMENT

PROFESSOR

Department of CSE

K.Ramakrishnan College of Technology (Autonomous)

Samayapuram–621112.

CGB1201-JAVA PROGRAMMING
Mrs.K.VALLI PRIYADHARSHINI, M.E.,(Ph.D.,)
SUPERVISOR
ASSISTANTP ROFESSOR

SIGNATURE

Mrs.K.Valli Priyadharshini, M.E.,(Ph.D.,),

SUPERVISOR

ASSISTANT PROFESSOR

Department of CSE

K.Ramakrishnan College of Technology (Autonomous)

Samayapuram–621112.

Submitted for the viva-voce examination held on 03/12/2024

CGB1201-JAVA PROGRAMMING
Mr.MAHARAJAMANNAN A, M.E.,
INTERNAL EXAMINER
ASSISTANT PROFESSOR

INTERNAL EXAMINER

CGB1201-JAVA PROGRAMMING
Mrs.R.ARUNA PRIYA, M.E.,
EXTERNAL EXAMINER
ASSISTANT PROFESSOR
8104-DSEC, PERAMBALUR.

EXTERNAL EXAMINER

DECLARATION

I declare that the project report on **"MOVIE TICKET BOOKING SYSTEM"** is the result of original work done by us and best of our knowledge, similar work has not been submitted to **"ANNA UNIVERSITY CHENNAI"** for the requirement of Degree of **BACHELOR OF ENGINEERING**. This project report is submitted on the partial fulfilment of the requirement of the completion of the course **CGB1201 - JAVA PROGRAMMING.**

.

Signature

KETHAAR ESHWAR A

Place: Samayapuram

Date: 03/12/2024

ACKNOWLEDGEMENT

It is with great pride that I express our gratitude and in-debt to our institution "**K.Ramakrishnan College of Technology (Autonomous)**", for providing us with the opportunity to do this project.

I glad to credit honourable chairman **Dr. K. RAMAKRISHNAN**, **B.E.,** for having provided for the facilities during the course of our study in college.

I would like to express our sincere thanks to our beloved Executive Director **Dr. S. KUPPUSAMY, MBA, Ph.D.,** for forwarding to our project and offering adequate duration in completing our project.

I would like to thank **Dr. N. VASUDEVAN, M.Tech., Ph.D.,** Principal, who gave opportunity to frame the project the full satisfaction.

I whole heartily thanks to **Dr. A. DELPHIN CAROLINA RANI, M.E.,Ph.D.,** Head of the department, **COMPUTER SCIENCE AND ENGINEERING** for providing her encourage pursuing this project.

I express our deep expression and sincere gratitude to our project supervisor **Mrs. K. VALLI PRIYADHARSHINI, M.E., (Ph.D.,),** Department of **COMPUTER SCIENCE AND ENGINEERING,** for his incalculable suggestions, creativity, assistance and patience which motivated us to carry out this project.

I render our sincere thanks to Course Coordinator and other staff members for providing valuable information during the course.

I wish to express our special thanks to the officials and Lab Technicians of our departments who rendered their help during the period of the work progress.

VISION OF THE INSTITUTION

To serve the society by offering top-notch technical education on par with global standards

MISSION OF THE INSTITUTION

- Be a center of excellence for technical education in emerging technologies by exceeding the needs of the industry and society.
- Be an institute with world class research facilities
- Be an institute nurturing talent and enhancing the competency of students to transform them as all-round personality respecting moral and ethical values

VISION OF DEPARTMENT

To be a center of eminence in creating competent software professionals with research and innovative skills.

MISSION OF DEPARTMENT

M1: Industry Specific: To nurture students in working with various hardware and software platforms inclined with the best practices of industry.

M2: Research: To prepare students for research-oriented activities.

M3: Society: To empower students with the required skills to solve complex technological problems of society.

PROGRAM EDUCATIONAL OBJECTIVES

1. PEO1: Domain Knowledge

To produce graduates who have strong foundation of knowledge and skills in the field of Computer Science and Engineering.

2. PEO2: Employability Skills and Research

To produce graduates who are employable in industries/public sector/research organizations or work as an entrepreneur.

3. PEO3: Ethics and Values

To develop leadership skills and ethically collaborate with society to tackle real-world challenges.

PROGRAM SPECIFIC OUTCOMES (PSOs)

PSO 1: Domain Knowledge

To analyze, design and develop computing solutions by applying foundational concepts of Computer Science and Engineering.

PSO 2: Quality Software

To apply software engineering principles and practices for developing quality software for scientific and business applications.

PSO 3: Innovation Ideas

To adapt to emerging Information and Communication Technologies (ICT) to innovate ideas and solutions to existing/novel problems

PROGRAM OUTCOMES (POs)

Engineering students will be able to:

1. **Engineering knowledge:** Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.
2. **Problem analysis:** Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences
3. **Design/development of solutions:** Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations
4. **Conduct investigations of complex problems:** Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions

5. **Modern tool usage:** Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations

6. **The engineer and society:** Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice

7. **Environment and sustainability:** Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development

8. **Ethics:** Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.

9. **Individual and team work:** Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.

10. **Communication:** Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.

11. **Project management and finance:** Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.

12. **Life-long learning:** Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

ABSTRACT

The Movie Ticket Booking System is a Java- based application designed to simulate a movie ticket booking experience using a graphical user interface (GUI). The system is structured into three primary panels, each catering to a different set of user roles. The Main panel serves as the entry point, offering options to navigate to either the User or Admin panel using buttons. The User panel allows individuals to book movie tickets by filling out a form with their personal details, such as name, mobile number, the number of tickets, and payment amount. The system validates the input, ensuring that the mobile number is exactly 10 digits and that the payment made by the user is sufficient for the selected number of tickets.The Admin panel, accessible only after a successful login with credentials, offers administrative functionalities. Admins can update movie information, including the movie name, ticket price, and show timings, which are reflected in the system. Admins can also view all booking records, including user details and the movies they've booked, as well as receive real-time notifications for new bookings made by users.

ABSTRACT WITH POs AND PSOs MAPPING

CO 5 : BUILD JAVA APPLICATIONS FOR SOLVING REAL-TIME PROBLEMS.

ABSTRACT	POs MAPPED	PSOs MAPPED
The Movie Ticket Booking System is a Java-based GUI application designed to facilitate movie ticket booking for users and management for administrators. It features three primary panels: Main, User, and Admin, with navigation handled via a CardLayout. Users can select a movie, input personal details, and book tickets by ensuring valid inputs, such as a 10-digit mobile number and sufficient payment. Administrators, upon successful login, can update movie details, view booking history, and manage notifications of new bookings. The system validates user input and provides real-time feedback, ensuring a seamless experience. Additionally, the Movie Class encapsulates movie data, and the interface is designed for easy navigation, making the system efficient for both users and admins.	**PO1 -3** **PO2 -3** **PO3 -3** **PO5 -3** **PO6 -3** **PO8 -3** **PO9 -3** **PO10 -3** **PO11-3** **PO12 -3**	**PSO1 -3** **PSO2 -3** **PSO3 -3**

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

1.1 Objective

The Movie Ticket Booking System is a Java-based graphical user interface application designed to streamline the movie ticket booking process. It features separate panels for users and administrators, ensuring a tailored experience for each. Users can browse available movies, book tickets by providing their details, and receive instant feedback on booking status or payment discrepancies. Administrators can securely log in to manage movie listings, update ticket prices and showtimes, and view booking history or user notifications. The system incorporates input validation, intuitive navigation using a CardLayout, and visually distinct themes for enhanced usability, making it an efficient tool for managing movie ticket operations.The system leverages a user-friendly design with clearly defined components, such as text fields, buttons, and dropdown menus, to ensure ease of use for both users and administrators. Input validation mechanisms, such as checking mobile numbers and payment amounts, enhance reliability, while real-time feedback ensures a smooth booking experience.

1.2 Overview

The Movie Ticket Booking System is a Java-based application in for designed to simplify movie ticket booking and management using a graphical user interface built with AWT and Swing. It provides two distinct modules: User Panel and Admin Panel. The User Panel allows users to view available movies, input personal details, select movies, book tickets, and receive feedback on payment and booking status. The Admin Panel, secured with login authentication, enables administrators to manage movie listings, update ticket details, and view booking histories and notifications. With features like input validation, real-time feedback, and intuitive navigation through a CardLayout, the system ensures an efficient and user-friendly experience for both customers and administrators.

1.3 Java Programming Concepts

1. Encapsulation: The Movie class encapsulates movie details, such as name, price, and timing, using private fields and public getter and setter methods. This ensures controlled access to movie data and maintains data integrity within the system.

2. Inheritance:The MovieTicketBookingSystem class extends the Frame class, inheriting methods and properties to create a GUI window. This allows the integration of GUI components such as buttons, text fields, and panels, along with event-handling capabilities.

3. Abstraction: The system provides a high-level interface for users and administrators, hiding the complexity of operations like payment validation, booking history updates, and admin authentication. Users interact with simple GUI components without needing to know the underlying logic.

4. Object-Oriented Design:The program utilizes classes like Movie, MovieTicketBookingSystem, and separate panels for user and admin functionality, adhering to the principles of modular and object-oriented design. This separation of concerns makes the system more organized and maintainable.

5. Collection Framework:The ArrayList is used to manage dynamic collections such as the list of movies (movies), user booking histories (bookingHistory), and admin notifications (adminNotifications), providing flexibility and scalability for managing data.

6. Event Handling:The system implements ActionListener to handle various user actions, such as button clicks for booking tickets, updating movie details, and switching between panels. This demonstrates Java's event-driven programming capabilities.

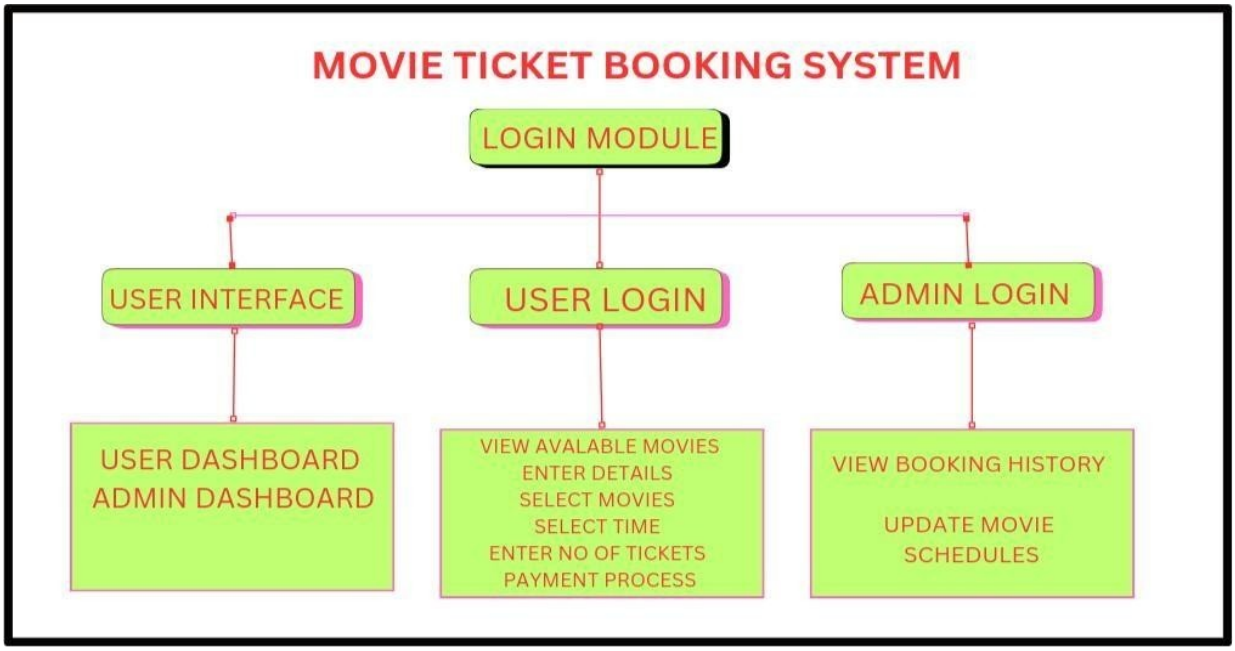
CHAPTER 2

PROJECT METHODOLOGY

2.1 Proposed Work

The proposed Movie Ticket Booking System is a Java-based application designed to simplify and enhance the process of booking movie tickets. The system will feature a user module allowing customers to browse available movies, book tickets, and receive real-time feedback on payment and booking status. An admin module will enable administrators to securely manage movie details, update ticket prices, and view booking histories and notifications. Core Java concepts like encapsulation, inheritance, polymorphism, and collections will be utilized to ensure a robust and modular design. The application will also leverage Java's AWT and Swing libraries to provide a user-friendly graphical interface for seamless interaction.

2.2 Block Diagram



CHAPTER 3
MODULE DESCRIPTION

3.1 User Module

Users allows to view and book tickets for available movies. Each movie is displayed with essential details, including the name, showtime, and ticket price. Users can select a movie, specify the number of tickets they wish to book, and enter their payment details. The module ensures input validation, such as verifying valid mobile numbers and checking sufficient payment for the selected tickets. After booking, the system provides feedback on the success or failure of the transaction, including payment status and booking confirmation.

3.2 Admin Module

The admin module provides tools for administrators to manage the movie listings. Administrators can add, update, or remove movies, set ticket prices, and modify showtimes. The module also includes a secure authentication process to prevent unauthorized access. Admins can view a list of all bookings made by users, with booking details such as the user's name, movie booked, number of tickets, and total payment. This ensures efficient management of movie data and customer bookings.

3.3 Booking history and notification Module

In this module maintains a record of all user bookings and sends notifications to the admin whenever a new booking is made. It stores booking details, including the user's name, mobile number, movie, ticket count, and payment amount, and provides a way to track the success or failure of the transactions. Admins receive notifications about new bookings and can view detailed reports through the admin panel.

3.4 Payment handling Module

Payment handling module handles the payment process during ticket bookings. It ensures the user provides the correct payment amount based on the number of tickets and movie prices. It also handles payment validation by checking whether the entered amount is sufficient for the booking. In case of an insufficient amount, the system notifies the user and requests the correct payment.

3.5 Data persistence Module

Data persistence system ensures that all user bookings and movie information are saved between application runs. It uses file handling mechanisms to serialize movie data and booking records to a file, ensuring data persistence. This module ensures that data is preserved even after the application is closed or restarted, enabling continuity and security of user and movie data.

CHAPTER 4

CONCLUSION & FUTURE SCOPE

4.1 CONCLUSION

The Movie Ticket Booking System successfully integrates core Java programming concepts to create a functional and user-friendly application. By utilizing encapsulation, inheritance, polymorphism, and event handling, the system efficiently manages movie bookings for users and movie data for administrators. The system's modular design, including user and admin modules, payment handling, and data persistence, ensures smooth and secure operations. The use of a graphical user interface (GUI) enhances the user experience, making navigation intuitive for both customers and administrators. Overall, the project demonstrates the power of object-oriented programming principles in building a real-world application, providing a seamless movie ticket booking experience while maintaining data integrity and security.

4.2 FUTURE SCOPE

The Movie Ticket Booking System has significant potential for future enhancement. Key developments could include integrating online payment gateways for seamless transactions, expanding the system to support multiple theaters and locations, and adding a seat reservation feature for a more personalized user experience. User profile management could offer personalized recommendations and booking histories, while the admin module could be upgraded to support role- based access for more efficient management. Furthermore, the implementation of a mobile app and the addition of analytics and reporting tools would improve accessibility and provide valuable insights for administrators. These advancements would make the system more scalable, user-friendly, and feature-rich, catering to a broader audience.

APPENDIX A

(SOURCE

CODE)

```
import java.awt.*;
import java.awt.event.*;
import java.util.ArrayList;
import javax.swing.*;
class MovieTicketBookingSystem extends Frame implements ActionListener
{ CardLayout card;
Panel mainPanel, userPanel, adminPanel;
 Button btnUser, btnAdmin, btnBackToMain, btnUserExit, btnAdminExit;
 Choice movieList;
   TextField userName, userMobile, ticketCount, paymentAmount;
    Button btnBook;
Label userMessage;
ArrayList<String> bookingHistory = new ArrayList<>();
  ArrayList<String> adminNotifications = new ArrayList<>();

      TextField movieName, ticketPrice, showTime;
   Button btnUpdate, btnViewBookings, btnViewNotifications;
   TextArea adminMessage;

   // Movie Data
   class Movie {
      String   name;
      double  price;
      String
      timing;

      Movie(String name, double price, String timing)
         { this.name = name;
         this.price = price;
         this.timing = timing;
      }

      @Override
      public String toString() {
         return name + " - " + timing + " ($" + price + ")";
      }
   }
```

```
ArrayList<Movie> movies = new ArrayList<>();

MovieTicketBookingSystem()
   { setTitle("Movie Ticket Booking
   System"); setSize(800, 600);
   setBackground(new Color(245, 245, 245)); // Light background color
   setLayout(new BorderLayout());

   // Sample Movies
   movies.add(new Movie("Avatar", 250, "10:00 AM"));
   movies.add(new Movie("Inception", 300, "1:00 PM"));
   movies.add(new Movie("Interstellar", 350, "5:00 PM"));

   // Main Panel
   mainPanel = new Panel();
   mainPanel.setBackground(new Color(173, 216, 230)); // Light blue
   mainPanel.setLayout(new FlowLayout());
   btnUser = new Button("User");
   btnUser.setBackground(new Color(135, 206, 250)); // Light Sky Blue
   btnUser.setForeground(Color.BLACK);
   btnAdmin = new Button("Admin");
   btnAdmin.setBackground(new Color(255, 182, 193)); // Light
   Pink btnAdmin.setForeground(Color.BLACK);
   mainPanel.add(btnUser);
   mainPanel.add(btnAdmin);

   // User Panel
   userPanel = new Panel();
   userPanel.setBackground(new Color(240, 255, 240)); // Light green
   userPanel.setLayout(new FlowLayout());
   movieList = new Choice();
   updateMovieList();
   userName = new TextField(15);
   userMobile = new TextField(10);
   ticketCount = new TextField(5);
   paymentAmount = new TextField(10);
   btnBook = new Button("Book Ticket");
   btnBook.setBackground(new Color(60, 179, 113)); // Medium Sea Green
   btnBook.setForeground(Color.WHITE);
   userMessage = new Label("Enter your details and book a movie ticket.");
   userMessage.setForeground(Color.RED); // Message in red
   btnUserExit = new Button("Exit");
   btnUserExit.setBackground(new Color(255, 69, 0)); // Red-
   Orange
```

```
        btnUserExit.setForeground(Color.WHITE);

        userPanel.add(new Label("Your Name:"));
        userPanel.add(userName);
        userPanel.add(new Label("Mobile Number (10
        digits):")); userPanel.add(userMobile);
        userPanel.add(new Label("Select Movie:"));
        userPanel.add(movieList);
        userPanel.add(new Label("No. of Tickets:"));
        userPanel.add(ticketCount);
        userPanel.add(new Label("Payment Amount:"));
        userPanel.add(paymentAmount);
        userPanel.add(btnBook);
        userPanel.add(userMessage);
        userPanel.add(btnUserExit);

        // Admin Panel
        adminPanel = new
        Panel();
        adminPanel.setBackground(new Color(255, 228, 196)); // Light peach
        adminPanel.setLayout(new FlowLayout());
        movieName = new TextField(15);
        ticketPrice = new TextField(5);
        showTime = new TextField(10);
        btnUpdate = new Button("Update Movie");
        btnUpdate.setBackground(new Color(0, 128, 128)); // Teal
        btnUpdate.setForeground(Color.WHITE);
        btnViewBookings = new Button("View Bookings");
Blue    btnViewBookings.setBackground(new Color(123, 104, 238)); // Medium Slate

        btnViewBookings.setForeground(Color.WHITE);
        btnViewNotifications = new Button("View Notifications");
        btnViewNotifications.setBackground(new Color(238, 232, 170)); // Pale
Goldenrod
        btnViewNotifications.setForeground(Color.BLACK);
        adminMessage = new TextArea(10, 50);
        btnAdminExit = new Button("Exit");
        btnAdminExit.setBackground(new Color(255, 69, 0)); // Red-Orange
        btnAdminExit.setForeground(Color.WHITE);

        adminPanel.add(new Label("Movie Name:"));
        adminPanel.add(movieName);
        adminPanel.add(new Label("Ticket Price:"));
        adminPanel.add(ticketPrice);
```

```
adminPanel.add(new Label("Show Time:"));
adminPanel.add(showTime);
adminPanel.add(btnUpdate);
adminPanel.add(btnViewBookings);
adminPanel.add(btnViewNotifications);
adminPanel.add(adminMessage);
adminPanel.add(btnAdminExit);

// Back to Main Button
btnBackToMain = new Button("Back");
btnBackToMain.setBackground(new Color(211, 211, 211)); // Light Gray
btnBackToMain.setForeground(Color.BLACK);
userPanel.add(btnBackToMain);
adminPanel.add(btnBackToMain);

// CardLayout to switch between panels
card = new CardLayout();
Panel cardPanel = new Panel();
cardPanel.setLayout(card);
cardPanel.add(mainPanel, "Main");
cardPanel.add(userPanel, "User");
cardPanel.add(adminPanel, "Admin");
add(cardPanel);

// Event Listeners
btnUser.addActionListener(e -> card.show(cardPanel, "User"));
btnAdmin.addActionListener(e -> {
   if (authenticateAdmin())
       { card.show(cardPanel,
       "Admin");
   }
});
btnBackToMain.addActionListener(e -> card.show(cardPanel, "Main"));
btnBook.addActionListener(this);
btnUpdate.addActionListener(this);
btnViewBookings.addActionListener(this);
btnViewNotifications.addActionListener(this);

// Exit Button Listeners
btnUserExit.addActionListener(e -> System.exit(0));
btnAdminExit.addActionListener(e -> System.exit(0));

// Close Action
addWindowListener(new WindowAdapter() {
```

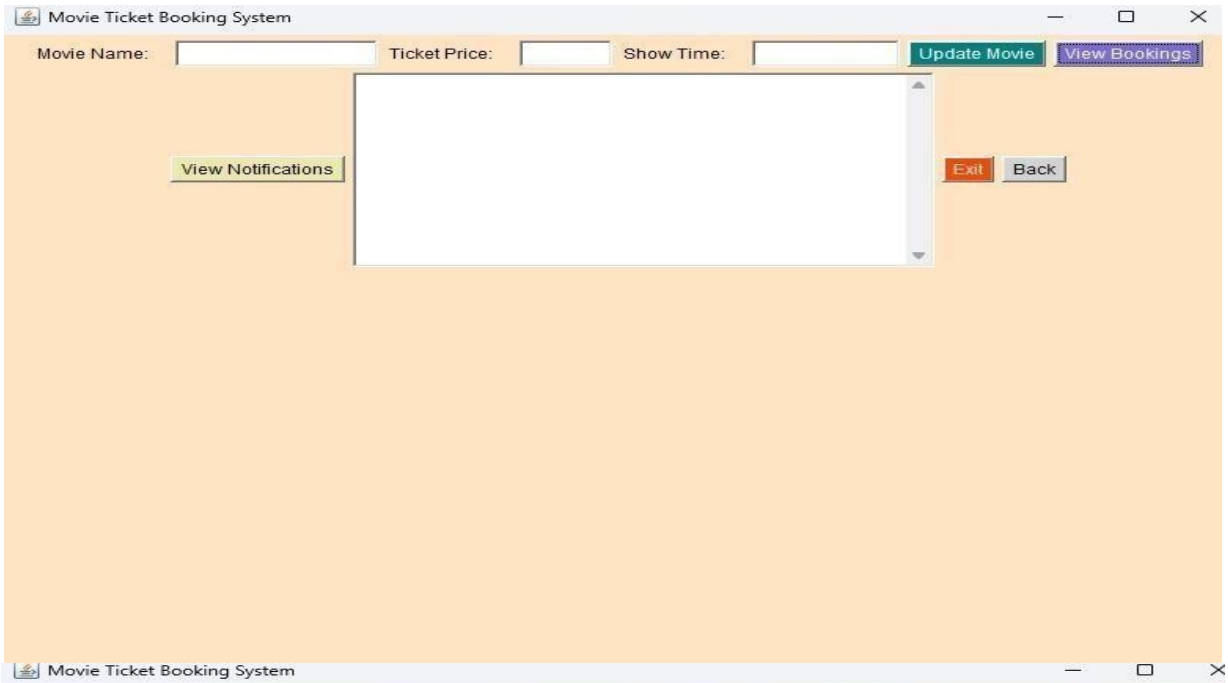
```
        public void windowClosing(WindowEvent e)
            { dispose();
        }
    });

    setVisible(true);
}

private boolean authenticateAdmin()
    { JTextField idField = new JTextField();
    JPasswordField passwordField = new
    JPasswordField(); Object[] message = {
        "Admin ID:", idField,
        "Password:", passwordField
    };
    int option = JOptionPane.showConfirmDialog(
        null, message, "Admin Login", JOptionPane.OK_CANCEL_OPTION);
    if (option == JOptionPane.OK_OPTION) {
        String id = idField.getText();
        String password = new String(passwordField.getPassword());
        return id.equals("admin@gmail.com") && password.equals("admin@123");
    }
    return false;
}

public void actionPerformed(ActionEvent e)
    { if (e.getSource() == btnBook)
        {try {
            String name = userName.getText().trim();
            String mobile = userMobile.getText().trim();
            if (!mobile.matches("\\d{10}")) {
                userMessage.setText("Invalid mobile number. Enter 10 digits.");
                return;
            }
            int tickets = Integer.parseInt(ticketCount.getText());
            double payment = Double.parseDouble(paymentAmount.getText());
            int selectedIndex = movieList.getSelectedIndex();
            Movie selectedMovie = movies.get(selectedIndex);
            double requiredPayment = tickets * selectedMovie.price;

            if (name.isEmpty())
                { userMessage.setText("Please enter your
                name.");
            } else if (payment >= requiredPayment) {
```

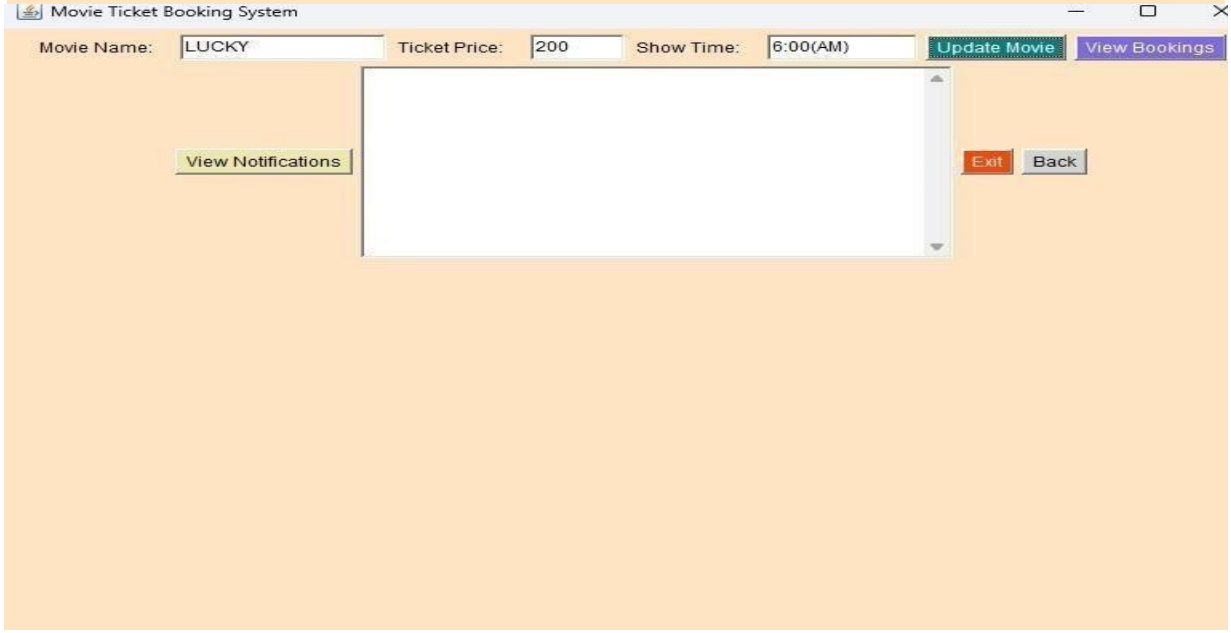
```
                String bookingDetails = "Name: " + name + ", Mobile: " + mobile +",
                                Movie: " + selectedMovie.name +
                                ", Tickets: " + tickets + ", Paid: " + payment;
                bookingHistory.add(bookingDetails);
                adminNotifications.add("New Booking: " + bookingDetails);
                userMessage.setText("Booking successful! Change: " + (payment -
requiredPayment));
            } else {
                userMessage.setText("Insufficient payment. Required: " +
requiredPayment);
            }
        } catch (NumberFormatException ex)
            { userMessage.setText("Invalid input. Please enter validnumbers.");
        }
    }
}

private void updateMovieList()
    { movieList.removeAll();
    for (Movie movie : movies)
        { movieList.add(movie.toString());
    }
}

public static void main(String[] args)
    { new MovieTicketBookingSystem();
}
}
```

APPENDIX B
(SCREENSHOTS)


Movie Ticket Booking System
Your Name:
KETHAAR
Mobile Number (10 digits):
1234567890
Select Movie:
Avatar - 10:00 AM ($250.0)
No. of Tickets:
4
Payment Amount:
990
Book Ticket
Insufficient payment. Required: 1000.0
Exit


Movie Ticket Booking System
Your Name:
KETHAAR
Mobile Number (10 digits):
1234567890
Select Movie:
Avatar - 10:00 AM ($250.0)
No. of Tickets:
4
Payment Amount:
1090
Book Ticket
Booking successful! Change: 90.0
Exit


Movie Ticket Booking System
Movie Name:
Ticket Price:
Show Time:
Update Movie
View Bookings
View Notifications
Exit
Back


Movie Ticket Booking System
Movie Name:
LUCKY
Ticket Price:
200
Show Time:
6:00(AM)
Update Movie
View Bookings
View Notifications
Exit
Back

REFERENCES

Java Books:

1. "Java: The Complete Reference" by Herbert Schild

   Comprehensive coverage of core Java, including AWT and event handling.

2. "Head First Java" by Kathy Sierra and Bert Bates

   A beginner-friendly guide to Java programming with a focus on object-oriented concepts.

Websites:

1. Oracle Java Documentation
   https://docs.oracle.com/javase/tutorial/Official tutorials on AWT, Swing, and event handling in Java.

2. GeeksforGeeks
   https://www.geeksforgeeks.org/java/Tutorials and examples of Java programming, including GUI development.

YouTube Links:

1. Telusko
   https://www.youtube.com/c/TeluskoTutorials on Java programming, including GUI design using AWT and Swing.

2. CodeWithHarry
   https://www.youtube.com/c/CodeWithHarryBeginner to advanced Java programming tutorials.

Comprehensive coverage of core Java, including AWT and event handling.

3. "Head First Java" by Kathy Sierra and Bert Bates

A beginner-friendly guide to Java programming with a focus on object-oriented concepts.

Websites:

1. Oracle Java Documentation
   https://docs.oracle.com/javase/tutorial/Official tutorials on AWT, Swing, and event handling in Java.

2. GeeksforGeeks
   https://www.geeksforgeeks.org/java/Tutorials and examples of Java programming, including GUI development.

YouTube Links:

1. Telusko
   https://www.youtube.com/c/TeluskoTutorials on Java programming, including GUI design using AWT and Swing.

2. CodeWithHarry
   https://www.youtube.com/c/CodeWithHarryBeginner to advanced Java programming tutorial