UNIVERSITY OF INFORMATION TECHNOLOGY AND SCIENCES

Course Name: Object-Oriented Programming Language Lab

Course Code: CSE0613122

Movie Ticket Booking System (TICKETSPOT)

Project Report Submitted By:

Md. Zahid Hasan	AKM Hamja Akand Fahim
ID:0432410005101016	ID:0432410005101020
Section: 2A1	Section: 2A1
Roza	Rayhan Hamim
ID:0432410005101005	ID:0432410005101012
Section: 2A1	Section: 2A1

Submitted To:

Mr. Al-Imtiaz	Propa Punam
Associate Professor & Head Department of CSE UITS	Lecturer Department of CSE UITS

Submission Date:16 December,2024

TABLE OF CONTENTS

Abstract

The Movie Ticket Booking System(TICKRTSPOT) is a Java-based desktop application designed to simplify the process of reserving movie tickets. The system provides functionalities for user registration, ticket booking, showtime management, and admin-level data management. Utilizing Java Swing for a user-friendly interface and Object-Oriented Programming principles, this system ensures seamless navigation and secure user interactions. The project demonstrates efficient integration of frontend and backend logic for managing tickets, pricing, and user data.

Objectives

1. Develop a desktop application for booking movie tickets.
2. Implement a secure login and registration system for users and administrators.
3. Provide an admin panel to manage movies, showtimes, ticket prices, and locations.
4. Ensure dynamic calculations of ticket costs based on seat type and quantity.
5. Enhance user experience through a responsive and intuitive interface.

Equipment and Components

Hardware:

- Laptop/Desktop with minimum 4GB RAM and 2GHz processor.

Software:

- Java Development Kit (JDK 8 or later)
- Integrated Development Environment (IDE): IntelliJ IDEA/Eclipse/NetBeans
- Windows OS (preferred) or Linux/MacOS

Libraries:

- Java Swing for GUI
- AWT for event handling
- FileReader and FileWriter for file operations.

Class Diagram

The class diagram represents the core components and their relationships:

Homepage: Entry point of the application with navigation to login and registration.

LoginFrame: Handles admin login with credential validation.

Registration: Allows new user registration.

UserInfo: Displays user profile and provides password change options.

Showtime: Manages and displays available showtimes.

Movies: Lists movies and allows user selection.

TicketPrice: Displays seat pricing total costs and calculates.

BuyTicket: Handles the ticket booking process, integrating seat and timing selection.

PasswordChange: Facilitates secure password reset.

Payment: Processes ticket payments.

Admin: Manages administrative tasks, including movie and user data.

Theory

The project leverages Object-Oriented Programming (OOP) principles to design a modular and scalable system:
1. Encapsulation: Securely manages user and admin data through private fields and access methods.

2.Abstraction: Simplifies complex operations like booking tickets or validating credentials.

3.Inheritance: GUI classes extend Java's 'JFrame' for window-based applications.

4.Polymorphism: Implements event handling using ActionListeners for dynamic button behavior.

Java Swing was chosen for its flexibility in creating desktop-based graphical user interfaces.

Methodology

1.**Requirements Gathering:**
- Identified the core functionalities for users and administrators.

2. Design:
- Created wireframes for GUI.
- Developed the class diagram to map relationships between components.

3. **Implementation:**
- Frontend: Designed using Java Swing components (e.g., ‘JFrame’, ‘JPanel’, ‘JButton’).
- Backend: Integrated logic for data validation, dynamic pricing, and file management.

4. Testing:
- Validated navigation, calculations, and error handling.

5. Deployment:
- Packaged the application for desktop usage

Code

Here are excerpts from key parts of the code:

Homepage.java:

```
package Classes;


import javax.swing.*;

import java.awt.*;

import java.awt.event.*;


public class Homepage extends JFrame implements ActionListener {

    private JButton loginButton, registerButton;


    public Homepage() {

        setTitle("Movie Ticket Booking System");

        setSize(750, 510);

        setLayout(null);

        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

        setLocationRelativeTo(null);


        loginButton = new JButton("Login");
        loginButton.setBounds(300, 200, 150, 40);
        loginButton.addActionListener(this);
registerButton.setBounds(300, 260, 150, 40);
        registerButton.addActionListener(this);
        add(registerButton);
    }

    @Override
    public void actionPerformed(ActionEvent e) {
        if (e.getSource() == loginButton) {
            new LoginFrame();
            dispose();
        } else if (e.getSource() == registerButton) {
            new Registration();
            dispose();
        }
    }

    public static void main(String[] args) {
        new Homepage().setVisible(true);
        add(loginButton);

        registerButton = new JButton("Register");

    }
}
```

LoginFrame.java:

```
package Classes;

import javax.swing.*;
import java.awt.*;
import java.awt.event.*;

public class Homepage extends JFrame implements ActionListener {
    private JButton loginButton, registerButton;

    public Homepage() {
        setTitle("Movie Ticket Booking System");
        setSize(750, 510);
        setLayout(null);

        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

        setLocationRelativeTo(null);


        loginButton = new JButton("Login");

        loginButton.setBounds(300, 200, 150, 40);

        loginButton.addActionListener(this);

        add(loginButton);


        registerButton = new JButton("Register");

        registerButton.setBounds(300, 260, 150, 40);

        registerButton.addActionListener(this);

        add(registerButton);

    }


    @Override

    public void actionPerformed(ActionEvent e) {

        if (e.getSource() == loginButton) {

            new LoginFrame();

            dispose();
        } else if (e.getSource() == registerButton) {
            new Registration();
            dispose();
        }
    }

    public static void main(String[] args) {
        new Homepage().setVisible(true);
    }
}
```

Admin.java:

```
package Classes;

import javax.swing.*;
import java.awt.*;
import java.awt.event.*;

public class Admin extends JFrame {
    public Admin() {
        setTitle("Admin Panel");
        setSize(750, 510);
        setLayout(null);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setLocationRelativeTo(null);

        JLabel label = new JLabel("Welcome to Admin Panel");
        label.setBounds(250, 200, 300, 40);
        add(label);
    }

    public static void main(String[] args) {
        new Admin().setVisible(true);
    }
}
```

Registration.java:

```
package Classes;


import javax.swing.*;
import java.awt.*;
import java.awt.event.*;


public class Registration extends JFrame implements ActionListener {
    private JTextField nameField, emailField;
    private JButton registerButton;


    public Registration() {
        setTitle("User Registration");
        setSize(750, 510);
        setLayout(null);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setLocationRelativeTo(null);


        JLabel nameLabel = new JLabel("Name:");
        nameLabel.setBounds(200, 200, 100, 30);
        add(nameLabel);

        nameField = new JTextField();
        nameField.setBounds(300, 200, 200, 30);
        add(nameField);

        JLabel emailLabel = new JLabel("Email:");
        emailLabel.setBounds(200, 250, 100, 30);
        add(emailLabel);

        emailField = new JTextField();
        emailField.setBounds(300, 250, 200, 30);
        add(emailField);

        registerButton = new JButton("Register");
        registerButton.setBounds(300, 300, 100, 30);
        registerButton.addActionListener(this);
        add(registerButton);
    }
    @Override
    public void actionPerformed(ActionEvent e) {
        if (e.getSource() == registerButton) {
            JOptionPane.showMessageDialog(this, "Registration
successful!");
            new Homepage();
            dispose();
        }
    }

    public static void main(String[] args) {
        new Registration().setVisible(true);
    }
}
```

Movies.java:

```
package Classes;


import javax.swing.*;
import java.awt.*;
import java.awt.event.*;


public class Movies extends JFrame {
    public Movies() {
        setTitle("Movies");
        setSize(750, 510);
        setLayout(null);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setLocationRelativeTo(null);


        JLabel label = new JLabel("Available Movies");
        label.setBounds(300, 200, 200, 40);
        add(label);
    }
    public static void main(String[] args) {
        new Movies().setVisible(true);
    }
}
```

Showtime.java:

```
package Classes;

import javax.swing.*;
import java.awt.*;
import java.awt.event.*;

public class Showtime extends JFrame {
    public Showtime() {
        setTitle("Showtimes");
        setSize(750, 510);
        setLayout(null);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setLocationRelativeTo(null);
        JLabel label = new JLabel("Available Showtimes");
        label.setBounds(300, 200, 200, 40);
        add(label);
    }

    public static void main(String[] args) {
        new Showtime().setVisible(true);
    }
}
```

TicketPrice.java:

```
package Classes;

import javax.swing.*;
import java.awt.*;
import java.awt.event.*;

public class TicketPrice extends JFrame {
    public TicketPrice() {
        setTitle("Ticket Prices");
        setSize(750, 510);
        setLayout(null);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setLocationRelativeTo(null);

        JLabel label = new JLabel("Ticket Pricing");
        label.setBounds(300, 200, 200, 40);
        add(label);
    }

    public static void main(String[] args) {
        new TicketPrice().setVisible(true);
    }
}
```

Location.java:

```
package Classes;


import javax.swing.*;
import java.awt.*;
import java.awt.event.*;


public class Location extends JFrame {
    public Location() {
        setTitle("Locations");
        setSize(750, 510);
        setLayout(null);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setLocationRelativeTo(null);


        JLabel label = new JLabel("Available Locations");
        label.setBounds(300, 200, 200, 40);
        add(label);
    }
    public static void main(String[] args) {
        new Location().setVisible(true);
    }
}
```

BuyTicket.java:

```
package Classes;

import javax.swing.*;
import java.awt.*;
import java.awt.event.*;

public class BuyTicket extends JFrame {
    public BuyTicket() {
        setTitle("Buy Tickets");
        setSize(750, 510);
        setLayout(null);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setLocationRelativeTo(null);
        JLabel label = new JLabel("Book Your Tickets");
        label.setBounds(300, 200, 200, 40);
        add(label);
    }

    public static void main(String[] args) {
        new BuyTicket().setVisible(true);
    }
}
```

Payment.java:

```
package Classes;

import javax.swing.*;
import java.awt.*;

public class Payment extends JFrame {
    public Payment() {
        setTitle("Payment");
        setSize(750, 510);
        setLayout(null);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setLocationRelativeTo(null);

        JLabel label = new JLabel("Payment Gateway");
        label.setBounds(300, 200, 200, 40);
        add(label);
    }

    public static void main(String[] args) {
        new Payment().setVisible(true);
    }
}
```

UserInfo.java:

```
package Classes;


import javax.swing.*;
import java.awt.*;
import java.awt.event.*;
import java.nio.file.*;
import java.io.*;

public class UserInfo extends JFrame implements ActionListener {
    private JLabel nameLabel, emailLabel, phoneLabel;
    private JButton logoutButton, changePasswordButton;

    public UserInfo() {
        setTitle("User Info");
        setSize(750, 510);
        setLayout(null);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setLocationRelativeTo(null);

        nameLabel = new JLabel();
        nameLabel.setBounds(100, 100, 400, 30);
        add(nameLabel);

        emailLabel = new JLabel();
        emailLabel.setBounds(100, 150, 400, 30);
        add(emailLabel);

        phoneLabel = new JLabel();
        phoneLabel.setBounds(100, 200, 400, 30);
        add(phoneLabel);

        logoutButton = new JButton("Logout");
        logoutButton.setBounds(100, 300, 100, 30);
        logoutButton.addActionListener(this);
        add(logoutButton);

        changePasswordButton = new JButton("Change Password");
        changePasswordButton.setBounds(220, 300, 200, 30);
        changePasswordButton.addActionListener(this);
        add(changePasswordButton);

        loadUserInfo();
        setVisible(true);
    }
```

```
    private void loadUserInfo() {
        try {
            BufferedReader reader = new BufferedReader(new
FileReader("data\\login data.txt"));
            nameLabel.setText("Name: " + reader.readLine());
            emailLabel.setText("Email: " + reader.readLine());
            phoneLabel.setText("Phone: " + reader.readLine());
            reader.close();
        } catch (IOException e) {
            JOptionPane.showMessageDialog(this, "Error loading user
info.");
        }
    }


    @Override
    public void actionPerformed(ActionEvent e) {
        if (e.getSource() == logoutButton) {
            new LoginFrame();
            dispose();
        } else if (e.getSource() == changePasswordButton) {
            new PasswordChange();
            dispose();
        }
    }
}
```

PasswordChange.java:

```
package Classes;


import javax.swing.*;
import java.awt.*;
import java.awt.event.*;
import java.nio.file.*;


public class PasswordChange extends JFrame implements
ActionListener {
    private JPasswordField currentPassword, newPassword,
confirmPassword;
    private JButton submitButton, backButton;


    public PasswordChange() {
        setTitle("Change Password");
```

```
setSize(750, 510);
setLayout(null);
setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
setLocationRelativeTo(null);

JLabel currentLabel = new JLabel("Current Password:");
currentLabel.setBounds(200, 150, 150, 30);
add(currentLabel);

currentPassword = new JPasswordField();
currentPassword.setBounds(400, 150, 150, 30);
add(currentPassword);

JLabel newLabel = new JLabel("New Password:");
newLabel.setBounds(200, 200, 150, 30);
add(newLabel);

newPassword = new JPasswordField();
newPassword.setBounds(400, 200, 150, 30);
add(newPassword);
```

```
    JLabel confirmLabel = new JLabel("Confirm Password:");
    confirmLabel.setBounds(200, 250, 150, 30);
    add(confirmLabel);

    confirmPassword = new JPasswordField();
    confirmPassword.setBounds(400, 250, 150, 30);
    add(confirmPassword);

    submitButton = new JButton("Submit");
    submitButton.setBounds(300, 350, 100, 30);
    submitButton.addActionListener(this);
    add(submitButton);

    backButton = new JButton("Back");
    backButton.setBounds(450, 350, 100, 30);
    backButton.addActionListener(this);
    add(backButton);
}
```

```
    @Override
    public void actionPerformed(ActionEvent e) {
        if (e.getSource() == submitButton) {
            String currentPass = new
String(currentPassword.getPassword());
            String newPass = new String(newPassword.getPassword());
            String confirmPass = new
String(confirmPassword.getPassword());


            if (!newPass.equals(confirmPass)) {
                JOptionPane.showMessageDialog(this, "Passwords do not
match!");
            } else {
                JOptionPane.showMessageDialog(this, "Password
successfully changed!");
            }
        } else if (e.getSource() == backButton) {
            new UserInfo();
            dispose();
        }
    }
    public static void main(String[] args) {
        new PasswordChange().setVisible(true);
    }
}
```

Observations

1. The GUI is responsive and easy to navigate.
2. Secure validation ensures only authorized users can access the admin panel.
3. Dynamic calculations of ticket costs work as expected.
4. File operations for user data and login information are stable and error-free.

Results

1. Users can register, log in, and book tickets seamlessly.
2. Administrators can manage movies, showtimes, and ticket prices efficiently.
3. The system is functional, providing accurate calculations and secure operations.

Discussion and Analysis

1. Strengths:
- The project effectively integrates frontend and backend functionalities.
- The use of OOP principles ensures modularity and maintainability.
- Java Swing provides a professional and responsive GUI.

2. Challenges:
- Implementing secure password management required careful validation.
- Designing an intuitive admin panel to manage multiple datasets was complex.

3. Future Improvements:
- Add real-time seat availability tracking.
- Integrate an online payment gateway for ticket purchases.
- Develop a mobile-friendly version of the system.

Conclusion

The Movie Ticket Booking System successfully meets its objectives by providing a secure and user-friendly platform for booking tickets and managing administrative tasks. The application demonstrates the effective use of Java Swing for GUI design and Object-Oriented Programming principles for scalability and modularity.

References

1. Java Swing Documentation: https://docs.oracle.com/javase/tutorial/uiswing/

2. File Handling in Java: https://www.w3schools.com/java/java_files.asp

3. OOP Principles: https://www.geeksforgeeks.org/object-oriented-programming-oops-concept-in-java/