**EVENT REGISTRATION SYSTEM**

**A MINI PROJECT REPORT**

**Submitted by**

**ABIRAMI K                                                      230701008**

**ABHINAYA LAKSHMI S                              230701004**

**in partial fulfillment of the award of the degree**

**of**

**BACHELOR OF ENGINEERING**

**IN**

**COMPUTER SCIENCE AND ENGINEERING**

****

**RAJALAKSHMI ENGINEERING COLLEGE, CHENNAI**

**An Autonomous Institute**

**CHENNAI**

**DECEMBER 2024**

**BONAFIDE CERTIFICATE**

Certified that this project is the bonafide work of **“ ABIRAMI K,**

**S. ABHINAYA LAKSHMI ”**  who carried out the project work under my supervision.

Submitted for the practical examination held on \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

SIGNATURE

Ms.THEEPA SAI

Rajalakshmi Engineering College Rajalakshmi Engineering College  (Autonomous), (Autonomous),

Thandalam,Chennai-602105 Thandalam,Chennai-602105

**INTERNAL EXAMINER EXTERNAL EXAMINER**

**ABSTRACT**

The "Event Registration System" is a Java-based desktop application designed to streamline the process of registering for events. Built with Java Swing for a user-friendly interface and JDBC for database connectivity, this application enables users to input personal details, select from various event options, and manage their registration data efficiently.

Core functionalities include user registration, modification, deletion, and a reset feature to clear input fields, allowing users to manage their information seamlessly. The event dropdown offers options such as Debate, Shark Tank, and Hackathon, enhancing usability by providing clear choices.

The application securely connects to a MySQL database through JDBC, facilitating efficient storage, updating, and retrieval of user data. By leveraging Java’s object-oriented principles, this project exemplifies how Java technologies can be applied to create reliable and practical solutions for event management.

The "Event Registration System" showcases Java’s capability in building structured, responsive, and scalable applications tailored to real-world needs.

**TABLE OF CONTENTS**

1. **Introduction**  
   1.1 Introduction  
   1.2 Implementation  
   1.3 Scope of the Project  
   1.4 Application Features
2. **System Specification**  
   2.1 Hardware Specification  
   2.2 Software Specification
3. **Sample Cod**e  
   3.1 Registration Form  
   3.2 Add Event Form  
   3.3 Admin Page  
   3.4 Login Page
4. **Snapshots**  
   4.1 Login page  
   4.2 Admin dashboard  
   4.3 Adding event  
   4.4 Event registration - USER  
   4.5 Modifying event  
   4.6 Cancelling event
5. **Conclusion**
6. **References**

**INTRODUCTION**

**1.1 INTRODUCTION**

The Event Registration System is a Java-based application designed to streamline the process of registering for events. Users can enter personal details, select an event, and manage their registration data through a user-friendly interface. The application allows for easy modification and deletion of user data, making it versatile for event management needs.

**1.2 IMPLEMENTATION**

This project is implemented using Java Swing for the graphical user interface and MySQL for the backend database, connected via JDBC. The Swing framework provides a structured GUI, while MySQL allows secure data storage and retrieval**.**

**1.3 SCOPE OF THE PROJECT**

The application is intended for event organizers who need a simple and efficient way to manage event registrations. It supports user registration, modification, deletion, and form reset functions, and provides users with predefined event options. By centralizing event registration data, it saves organizers time and improves data management.

**1.4 APPLICATION FEATURES**

* User registration and login functionality
* User profile management with data modification and deletion options
* Event selection with multiple predefined options
* Reset button to clear all input fields

**SYSTEM SPECIFICATIONS**

**2.1 HARDWARE SPECIFICATIONS**

* PROCESSOR : Intel i5 or equivalent
* MEMORY : 4GB RAM (minimum)
* STORAGE : 500 GB of free space

**2.2 SOFTWARE SPECIFICATIONS**

* PROGRAMMING LANGUAGE : Java, MySQL
* FRONT-END : Java Swing
* BACK-END : MySQL Database
* OPERATING SYSTEM : Windows 10

**SAMPLE CODE**

**3.1 REGISTRATION FORM**

package swingex;

import java.awt.\*;

import javax.swing.\*;

import javax.swing.border.EmptyBorder;

import java.awt.event.ActionEvent;

import java.sql.\*;

public class RegistrationForm extends JFrame {

    private static final long serialVersionUID = 1L;

    private JPanel contentPane;

    private JTextField txtName, txtAddress, txtAge, txtmb, txtEmail, txtUsername;

    private JPasswordField txtPassword;

    private ButtonGroup genderGroup;

    private static JComboBox<String> eventDropdown;

    public static void main(String[] args) {

        EventQueue.invokeLater(() -> {

            try {

                RegistrationForm frame = new RegistrationForm();

                frame.setVisible(true);

            } catch (Exception e) {

                e.printStackTrace();

            }

        });

    }

    public RegistrationForm() {

        setIconImage(Toolkit.getDefaultToolkit().getImage("D:\\Downloads\\user.png"));

        setTitle("Event Registration Form");

        setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

        setExtendedState(JFrame.MAXIMIZED\_BOTH); // Make the frame full screen

        // Use GridBagLayout for centered alignment

        contentPane = new JPanel(new GridBagLayout());

        contentPane.setBackground(new Color(245, 245, 220)); // Beige background

        contentPane.setBorder(new EmptyBorder(10, 10, 10, 10));

        setContentPane(contentPane);

        GridBagConstraints gbc = new GridBagConstraints();

        gbc.insets = new Insets(5, 5, 5, 5);

        gbc.anchor = GridBagConstraints.CENTER;

        gbc.fill = GridBagConstraints.HORIZONTAL;

        // Heading Label

        JLabel headingLabel = new JLabel("User Registration Form");

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

        headingLabel.setForeground(Color.BLACK);

        gbc.gridx = 0;

        gbc.gridy = 0;

        gbc.gridwidth = 2;

        gbc.insets = new Insets(20, 5, 20, 5);

        contentPane.add(headingLabel, gbc);

        // Reset grid width and insets for the rest of the fields

        gbc.gridwidth = 1;

        gbc.insets = new Insets(5, 5, 5, 5);

        // Add form labels and fields

        gbc.gridx = 0;

        gbc.gridy++;

        contentPane.add(new JLabel("Name:"), gbc);

        txtName = new JTextField(20);

        gbc.gridx = 1;

        contentPane.add(txtName, gbc);

        gbc.gridx = 0;

        gbc.gridy++;

        contentPane.add(new JLabel("Address:"), gbc);

        txtAddress = new JTextField(20);

        gbc.gridx = 1;

        contentPane.add(txtAddress, gbc);

        gbc.gridx = 0;

        gbc.gridy++;

        contentPane.add(new JLabel("Gender:"), gbc);

        JPanel genderPanel = new JPanel(new FlowLayout());

        JRadioButton rbFemale = new JRadioButton("Female");

  JRadioButton rbMale = new JRadioButton("Male");

        genderPanel.add(rbFemale);

        genderPanel.add(rbMale);

        genderGroup = new ButtonGroup();

        genderGroup.add(rbFemale);

        genderGroup.add(rbMale);

        gbc.gridx = 1;

        contentPane.add(genderPanel, gbc);

        gbc.gridx = 0;

        gbc.gridy++;

        contentPane.add(new JLabel("Age:"), gbc);

        txtAge = new JTextField(20);

        gbc.gridx = 1;

        contentPane.add(txtAge, gbc);

        gbc.gridx = 0;

        gbc.gridy++;

        contentPane.add(new JLabel("Mobile No.:"), gbc);

        txtmb = new JTextField(20);

        gbc.gridx = 1;

        contentPane.add(txtmb, gbc);

        gbc.gridx = 0;

        gbc.gridy++;

        contentPane.add(new JLabel("Email:"), gbc);

        txtEmail = new JTextField(20);

        gbc.gridx = 1;

        contentPane.add(txtEmail, gbc);

        gbc.gridx = 0;

        gbc.gridy++;

        contentPane.add(new JLabel("Select Event:"), gbc);

        eventDropdown = new JComboBox<>();

        gbc.gridx = 1;

        contentPane.add(eventDropdown, gbc);

        gbc.gridx = 0;

        gbc.gridy++;

        contentPane.add(new JLabel("Username:"), gbc);

        txtUsername = new JTextField(20);

        gbc.gridx = 1;

        contentPane.add(txtUsername, gbc);

        gbc.gridx = 0;

        gbc.gridy++;

        contentPane.add(new JLabel("Password:"), gbc);

        txtPassword = new JPasswordField(20);

        gbc.gridx = 1;

        contentPane.add(txtPassword, gbc);

        // Buttons in a separate panel

        JPanel buttonPanel = new JPanel(new FlowLayout(FlowLayout.CENTER, 10, 10));

        JButton btnRegister = new JButton("Register");

        btnRegister.addActionListener(e -> {

            if (validateFields()) {

                registerUser(rbFemale, rbMale);

            }

        });

        buttonPanel.add(btnRegister);

        JButton btnReset = new JButton("Reset");

        btnReset.addActionListener(e -> resetFields());

        buttonPanel.add(btnReset);

        JButton btnLogout = new JButton("Logout");

        btnLogout.addActionListener(e -> {

            dispose(); // Close the current window

            new Login().setVisible(true); // Open Login page

        });

        buttonPanel.add(btnLogout);

        gbc.gridx = 0;

        gbc.gridy++;

        gbc.gridwidth = 2;

        contentPane.add(buttonPanel, gbc);

        // Set initial visibility for specific buttons

        btnRegister.setVisible(true);

        btnReset.setVisible(true);

        loadEventsIntoDropdown(); // Load events into the eventDropdown

    }

    private boolean validateFields() {

        if (txtName.getText().isEmpty() || txtAddress.getText().isEmpty() ||

                txtAge.getText().isEmpty() || txtmb.getText().isEmpty() ||

                txtEmail.getText().isEmpty() || txtUsername.getText().isEmpty() ||

                txtPassword.getPassword().length == 0 || genderGroup.getSelection() == null) {

            JOptionPane.showMessageDialog(this, "Please fill in all fields", "Validation Error", JOptionPane.ERROR\_MESSAGE);

            return false;

        }

        try {

            Integer.parseInt(txtAge.getText());

            Long.parseLong(txtmb.getText());

        } catch (NumberFormatException e) {

            JOptionPane.showMessageDialog(this, "Please enter valid numbers for Age and Mobile", "Input Error", JOptionPane.ERROR\_MESSAGE);

            return false;

        }

        return true;

    }

    public static void loadEventsIntoDropdown() {

        try {

            Connection conn = createConnection();

            String sql = "SELECT event\_name FROM event\_details";

            PreparedStatement statement = conn.prepareStatement(sql);

            ResultSet resultSet = statement.executeQuery();

            eventDropdown.removeAllItems();

            while (resultSet.next()) {

                String eventName = resultSet.getString("event\_name");

                eventDropdown.addItem(eventName);

            }

            conn.close();

        } catch (Exception e) {

            e.printStackTrace();

        }

    }

    private void registerUser(JRadioButton rbFemale, JRadioButton rbMale) {

        try (Connection con = createConnection()) {

            String query = "INSERT INTO registration VALUES(?, ?, ?, ?, ?, ?, ?, ?)";

            PreparedStatement ps = con.prepareStatement(query);

            ps.setString(1, txtName.getText());

            ps.setString(2, txtAddress.getText());

            ps.setString(3, rbFemale.isSelected() ? "Female" : "Male");

            ps.setInt(4, Integer.parseInt(txtAge.getText()));

            ps.setString(5, txtmb.getText());

            ps.setString(6, txtEmail.getText());

            ps.setString(7, txtUsername.getText());

            ps.setString(8, new String(txtPassword.getPassword()));

            ps.executeUpdate();

            JOptionPane.showMessageDialog(this, "Registered successfully!");

        } catch (SQLException | ClassNotFoundException e) {

            e.printStackTrace();

        }

        loadEventsIntoDropdown();

    }

    private void resetFields() {

        txtName.setText("");

        txtAddress.setText("");

        txtAge.setText("");

        txtmb.setText("");

        txtEmail.setText("");

        txtUsername.setText("");

        txtPassword.setText("");

        genderGroup.clearSelection();

    }

    public static Connection createConnection() throws SQLException, ClassNotFoundException {

        Class.forName("com.mysql.cj.jdbc.Driver");

        return DriverManager.getConnection("jdbc:mysql://127.0.0.1:3308/users", "root", "Abi");

    }

}

**3.2 ADD EVENT FORM**

package swingex;

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.sql.\*;

public class AddEventForm extends JFrame {

    private JPanel contentPane;

    private JTextField txtEventName, txtEventDate, txtTicketPrice;

public AddEventForm() {

        // Set up the frame properties

        setTitle("Add Event");

        setDefaultCloseOperation(JFrame.DISPOSE\_ON\_CLOSE);

        // Maximize window but keep the title bar and window controls

        setExtendedState(JFrame.MAXIMIZED\_BOTH); // Maximize window

        setUndecorated(false); // Ensure window controls (close, minimize) are visible

        setResizable(true); // Allow resizing

        contentPane = new JPanel();

        setContentPane(contentPane);

        contentPane.setLayout(new GridBagLayout());

        contentPane.setBackground(new Color(245, 245, 220)); // Beige background

        GridBagConstraints gbc = new GridBagConstraints();

        gbc.insets = new Insets(5, 5, 5, 5); // Padding around components

        gbc.anchor = GridBagConstraints.WEST; // Align text to the left

        // Heading Label

        JLabel headingLabel = new JLabel("Add Event");

        headingLabel.setFont(new Font("Arial", Font.BOLD, 48)); // Larger font size

        headingLabel.setForeground(Color.BLACK);

        gbc.gridx = 0;

        gbc.gridy = 0;

        gbc.gridwidth = 2; // Make the heading span two columns

gbc.insets = new Insets(40, 5, 40, 5); // Adjust padding for heading

        contentPane.add(headingLabel, gbc);

        // Event Name Label and TextField

        gbc.gridwidth = 1;

        gbc.gridx = 0;

        gbc.gridy = 1;

        contentPane.add(new JLabel("Event Name:"), gbc);

        txtEventName = new JTextField(20);

        gbc.gridx = 1;

        contentPane.add(txtEventName, gbc);

        // Event Date Label and TextField

        gbc.gridx = 0;

        gbc.gridy = 2;

        contentPane.add(new JLabel("Event Date:"), gbc);

        txtEventDate = new JTextField(20);

        gbc.gridx = 1;

        contentPane.add(txtEventDate, gbc);

        // Ticket Price Label and TextField

        gbc.gridx = 0;

        gbc.gridy = 3;

        contentPane.add(new JLabel("Ticket Price:"), gbc);

        txtTicketPrice = new JTextField(20);

        gbc.gridx = 1;

        contentPane.add(txtTicketPrice, gbc);

        // Add Event Button

        JButton btnAddEvent = new JButton("Add Event");

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

        gbc.gridx = 0;

        gbc.gridy = 4;

        gbc.gridwidth = 1; // Set the grid width to 1

        contentPane.add(btnAddEvent, gbc);

        // Back Button

JButton btnBack = new JButton("Back");

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

        gbc.gridx = 1;

        contentPane.add(btnBack, gbc);

        // Action listener for adding event

        btnAddEvent.addActionListener(new ActionListener() {

            public void actionPerformed(ActionEvent e) {

                addEventToDatabase();

            }

        });

        // Action listener for going back to Admin Dashboard

        btnBack.addActionListener(new ActionListener() {

            public void actionPerformed(ActionEvent e) {

                // Close the current Add Event window

                dispose();

                // Open the Admin Dashboard window

                new AdminPage(); // Replace with actual admin dashboard class name

            }

        });

        setVisible(true); // Make the frame visible

    }

    private void addEventToDatabase() {

        String eventName = txtEventName.getText();

        String eventDate = txtEventDate.getText();

        String ticketPrice = txtTicketPrice.getText();

        if (eventName.isEmpty() || eventDate.isEmpty() || ticketPrice.isEmpty()) {

            JOptionPane.showMessageDialog(this, "All fields are required!");

            return;

        }

        try (Connection con = createConnection()) {

            String query = "INSERT INTO event\_details (event\_name, event\_date, ticket\_price) VALUES (?, ?, ?)";

            PreparedStatement ps = con.prepareStatement(query);

            ps.setString(1, eventName);

            ps.setString(2, eventDate);

            ps.setString(3, ticketPrice);

            ps.executeUpdate();

            // Show success message

            JOptionPane.showMessageDialog(this, "Event Added Successfully!");

            // Clear the text fields after adding the event

            txtEventName.setText("");

            txtEventDate.setText("");

            txtTicketPrice.setText("");

        } catch (SQLException | ClassNotFoundException e) {

            e.printStackTrace();

            JOptionPane.showMessageDialog(this, "Error adding event. Please try again.");

        }

    }

    private static Connection createConnection() throws SQLException, ClassNotFoundException {

        Class.forName("com.mysql.cj.jdbc.Driver");

        return DriverManager.getConnection("jdbc:mysql://127.0.0.1:3308/users", "root", "Abi");

    }

    public static void main(String[] args) {

        new AddEventForm(); // Open the Add Event form

    }

}

**3.3 ADMIN PAGE**

package swingex;

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.sql.\*;

public class AdminPage extends JFrame {

    private JPanel contentPane;

    public AdminPage() {

        setTitle("Admin Dashboard");

        setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

        setExtendedState(JFrame.MAXIMIZED\_BOTH); // Full screen

        contentPane = new JPanel();

        setContentPane(contentPane);

        contentPane.setLayout(null);

        contentPane.setBackground(new Color(245, 245, 220)); // Beige background

        // Heading Label

        JLabel headingLabel = new JLabel("Admin Dashboard");

        headingLabel.setFont(new Font("Arial", Font.BOLD, 32));

        headingLabel.setForeground(Color.BLACK);

        headingLabel.setBounds(350, 20, 300, 40);

        contentPane.add(headingLabel);

        // Add Event Button

        JButton btnAdd = new JButton("Add Event");

        btnAdd.setBounds(50, 100, 200, 40);

        btnAdd.setFont(new Font("Arial", Font.BOLD, 16));

        btnAdd.setBackground(Color.WHITE);

        contentPane.add(btnAdd);

        // Modify Event Button

        JButton btnModify = new JButton("Modify Event");

        btnModify.setBounds(50, 160, 200, 40);

        btnModify.setFont(new Font("Arial", Font.BOLD, 16));

        btnModify.setBackground(Color.WHITE);

        contentPane.add(btnModify);

        // Cancel Event Button

        JButton btnDelete = new JButton("Cancel Event");

        btnDelete.setBounds(50, 220, 200, 40);

        btnDelete.setFont(new Font("Arial", Font.BOLD, 16));

        btnDelete.setBackground(Color.WHITE);

        contentPane.add(btnDelete);

        // Remove View Registrations Button

        // Removed the View Registrations button

        // Add Event Form

        btnAdd.addActionListener(new ActionListener() {

            public void actionPerformed(ActionEvent e) {

                new AddEventForm(); // Open AddEventForm when Add Event button is clicked

            }

        });

        // Modify Event Form

        btnModify.addActionListener(new ActionListener() {

            public void actionPerformed(ActionEvent e) {

                modifyEvent(); // Modify event functionality

            }

        });

        // Delete Event functionality

        btnDelete.addActionListener(new ActionListener() {

            public void actionPerformed(ActionEvent e) {

                String eventName = JOptionPane.showInputDialog("Enter the event name to delete:");

                if (eventName != null) {

                    deleteEvent(eventName);

                }

            }

        });

    }

    public static void main(String[] args) {

        new AdminPage(); // Create and show the AdminPage window

    }

    private void deleteEvent(String eventName) {

        try (Connection conn = createConnection()) {

            String sql = "DELETE FROM event\_details WHERE event\_name = ?";

            PreparedStatement stmt = conn.prepareStatement(sql);

            stmt.setString(1, eventName);

            int rowsAffected = stmt.executeUpdate();

            if (rowsAffected > 0) {

                JOptionPane.showMessageDialog(this, "Event " + eventName + " deleted successfully.");

            } else {

                JOptionPane.showMessageDialog(this, "Event not found.");

            }

        } catch (SQLException | ClassNotFoundException ex) {

            ex.printStackTrace();

            JOptionPane.showMessageDialog(this, "Error deleting event.");

        }

    }

    private void modifyEvent() {

        // Ask for the event name to modify

        String eventName = JOptionPane.showInputDialog("Enter the event name to modify:");

        if (eventName != null && !eventName.isEmpty()) {

            // Show input fields to modify the event details

            JTextField txtEventDate = new JTextField(20);

            JTextField txtTicketPrice = new JTextField(20);

            Object[] message = {

                "Event Date:", txtEventDate,

                "Ticket Price:", txtTicketPrice

            };

            int option = JOptionPane.showConfirmDialog(this, message, "Modify Event - " + eventName, JOptionPane.OK\_CANCEL\_OPTION);

            if (option == JOptionPane.OK\_OPTION) {

                String newEventDate = txtEventDate.getText();

                String newTicketPrice = txtTicketPrice.getText();

                if (newEventDate.isEmpty() || newTicketPrice.isEmpty()) {

                    JOptionPane.showMessageDialog(this, "Please fill in both fields.");

                    return;

                }

                // Update the event details in the database

                updateEvent(eventName, newEventDate, newTicketPrice);

            }

        } else {

            JOptionPane.showMessageDialog(this, "Event name cannot be empty.");

        }

    }

    private void updateEvent(String eventName, String newEventDate, String newTicketPrice) {

        try (Connection conn = createConnection()) {

            String sql = "UPDATE event\_details SET event\_date = ?, ticket\_price = ? WHERE event\_name = ?";

            PreparedStatement stmt = conn.prepareStatement(sql);

            stmt.setString(1, newEventDate);

            stmt.setString(2, newTicketPrice);

            stmt.setString(3, eventName);

            int rowsAffected = stmt.executeUpdate();

            if (rowsAffected > 0) {

                JOptionPane.showMessageDialog(this, "Event updated successfully!");

            } else {

                JOptionPane.showMessageDialog(this, "Event not found.");

            }

        } catch (SQLException | ClassNotFoundException ex) {

            ex.printStackTrace();

            JOptionPane.showMessageDialog(this, "Error updating event.");

        }

    }

    private Connection createConnection() throws SQLException, ClassNotFoundException {

        Class.forName("com.mysql.cj.jdbc.Driver");

        return DriverManager.getConnection("jdbc:mysql://127.0.0.1:3308/users", "root", "Abi");

    }

}

**3.4 LOGIN PAGE**

package swingex;

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

public class Login extends JFrame {

    private JPanel contentPane;

    private JTextField txtUsername;

    private JPasswordField txtPassword;

    public static void main(String[] args) {

        SwingUtilities.invokeLater(() -> {

            Login frame = new Login();

            frame.setVisible(true);

        });

    }

    public Login() {

        setTitle("Login Page");

        setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

        setExtendedState(JFrame.MAXIMIZED\_BOTH); // Full screen

        // Panel setup with beige background

        contentPane = new JPanel(new GridBagLayout());

        contentPane.setBackground(new Color(245, 245, 220)); // Beige background

        setContentPane(contentPane);

        // Setting layout constraints for centered alignment

        GridBagConstraints gbc = new GridBagConstraints();

        gbc.insets = new Insets(10, 10, 10, 10);

        gbc.fill = GridBagConstraints.HORIZONTAL;

        // Welcome Label

        JLabel lblWelcome = new JLabel("Welcome to EventHub!");

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

        lblWelcome.setForeground(Color.BLACK);

        gbc.gridx = 0;

        gbc.gridy = 0;

        gbc.gridwidth = 2;

        gbc.anchor = GridBagConstraints.CENTER;

        contentPane.add(lblWelcome, gbc);

        // Reset grid width for other elements

        gbc.gridwidth = 1;

        gbc.anchor = GridBagConstraints.WEST;

        // Username Label

        JLabel lblUsername = new JLabel("Username:");

        lblUsername.setForeground(Color.BLACK);

        lblUsername.setFont(new Font("Arial", Font.BOLD, 16));

        gbc.gridx = 0;

        gbc.gridy = 1;

        contentPane.add(lblUsername, gbc);

        // Username TextField in a white bordered panel

        txtUsername = new JTextField(20);

        txtUsername.setFont(new Font("Arial", Font.PLAIN, 14));

        JPanel usernamePanel = new JPanel();

        usernamePanel.setLayout(new BorderLayout());

        usernamePanel.setBackground(Color.WHITE);

        usernamePanel.setBorder(BorderFactory.createLineBorder(Color.BLACK));

        usernamePanel.add(txtUsername, BorderLayout.CENTER);

        gbc.gridx = 1;

        contentPane.add(usernamePanel, gbc);

        // Password Label

        JLabel lblPassword = new JLabel("Password:");

        lblPassword.setForeground(Color.BLACK);

        lblPassword.setFont(new Font("Arial", Font.BOLD, 16));

        gbc.gridx = 0;

        gbc.gridy = 2;

        contentPane.add(lblPassword, gbc);

        // Password Field in a white bordered panel

        txtPassword = new JPasswordField(20);

        txtPassword.setFont(new Font("Arial", Font.PLAIN, 14));

        JPanel passwordPanel = new JPanel();

        passwordPanel.setLayout(new BorderLayout());

        passwordPanel.setBackground(Color.WHITE);

        passwordPanel.setBorder(BorderFactory.createLineBorder(Color.BLACK));

        passwordPanel.add(txtPassword, BorderLayout.CENTER);

        gbc.gridx = 1;

        contentPane.add(passwordPanel, gbc);

        // Login Button

        JButton btnLogin = new JButton("Login");

        btnLogin.setBackground(new Color(220, 220, 220));

        btnLogin.setForeground(Color.BLACK);

        btnLogin.setFont(new Font("Arial", Font.BOLD, 16));

        gbc.gridx = 0;

        gbc.gridy = 3;

        gbc.gridwidth = 2;

        gbc.anchor = GridBagConstraints.CENTER;

        contentPane.add(btnLogin, gbc);

        // Login Button Action

        btnLogin.addActionListener(new ActionListener() {

            public void actionPerformed(ActionEvent e) {

                String username = txtUsername.getText();

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

                if (username.equals("user") && password.equals("user")) {

                    dispose();

                    new RegistrationForm().setVisible(true);

                } else if (username.equals("admin") && password.equals("admin")) {

                    dispose();

                    new AdminPage().setVisible(true);

                } else {

                    JOptionPane.showMessageDialog(Login.this, "Invalid username or password", "Login Error", JOptionPane.ERROR\_MESSAGE);

                }

            }

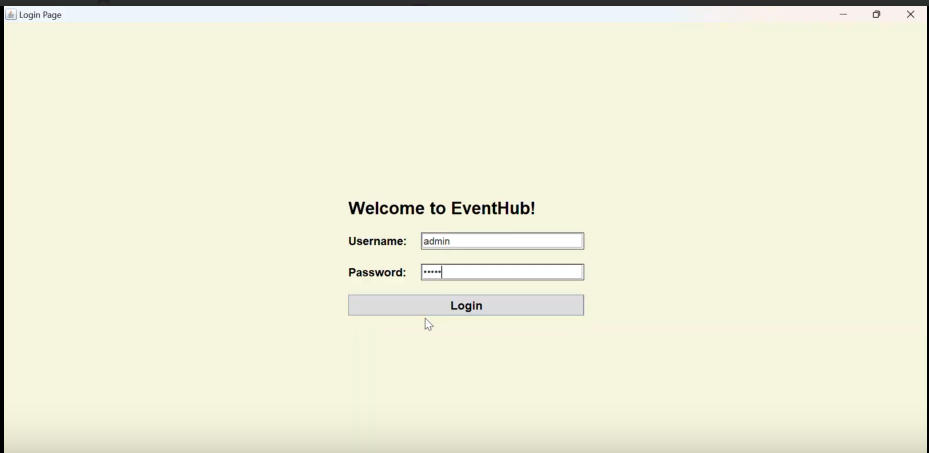
        });

    }

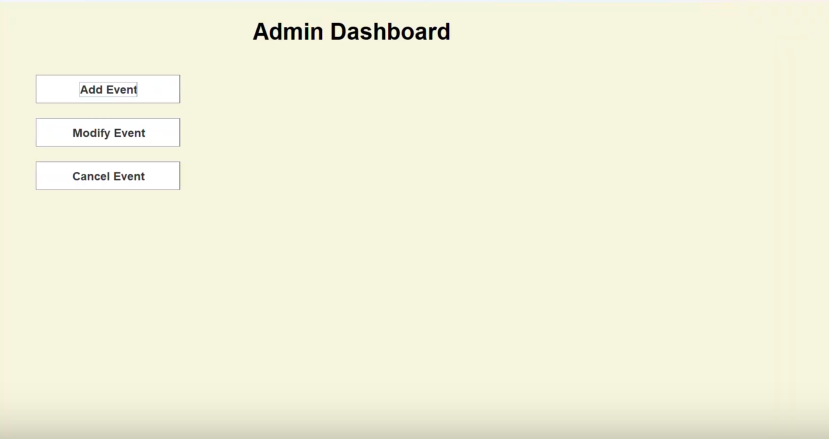
}

**SNAPSHOTS**

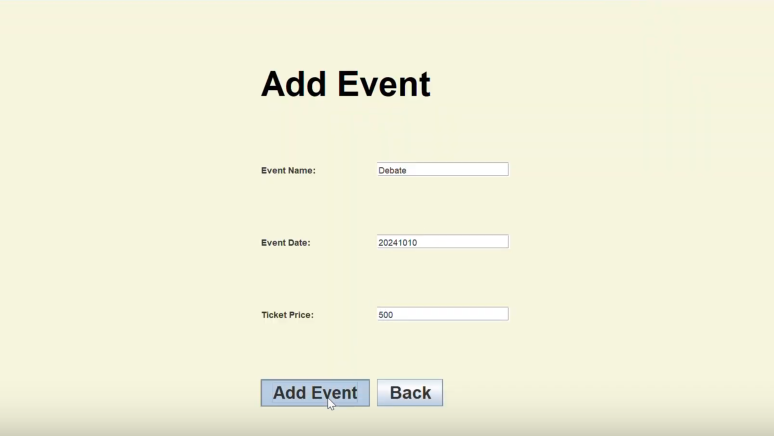
**LOGIN PAGE**

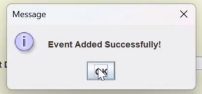
****

**ADMIN DASHBOARD**

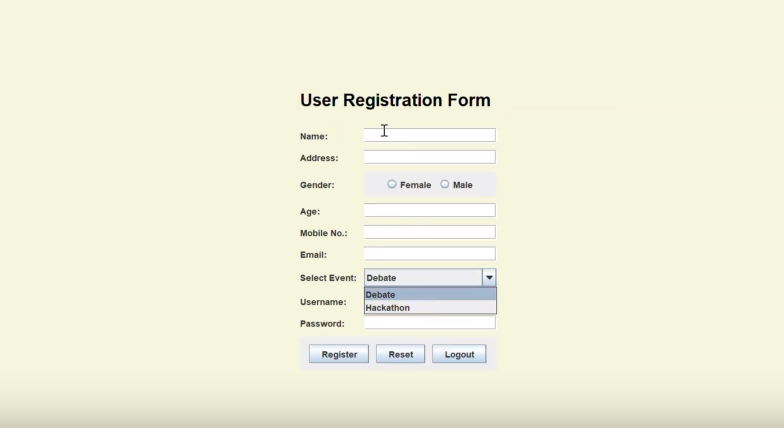
****

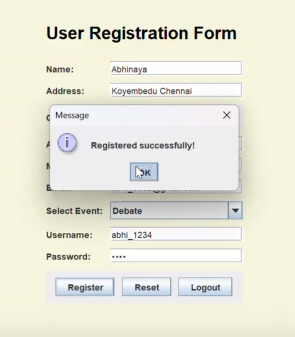
**ADD EVENT**

****

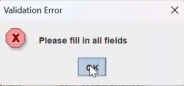


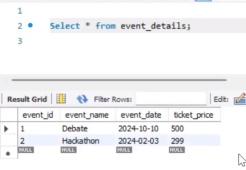
**EVENT REGISTRATION - USER**

****

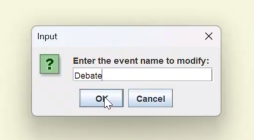


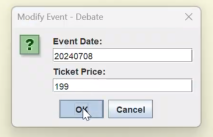


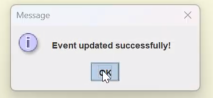


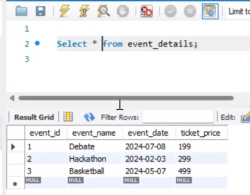


**MODIFY EVENT**

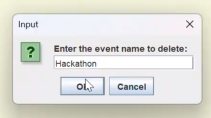


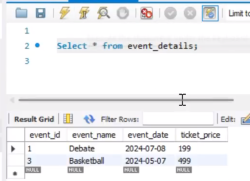






**CANCEL EVENT**





**CONCLUSION**

The Event Registration System provides an efficient solution for managing event registrations, utilizing Java Swing for the user interface and JDBC for database connectivity. It allows easy user registration, modification, and deletion, ensuring data accuracy and security. This project demonstrates the effectiveness of Java in building practical, user-friendly applications, making it a valuable tool for event organizers and a solid foundation for future enhancements.

**REFERENCES**

* <https://www.javacodegeeks.com/>
* <https://docs.oracle.com/javase/8/docs/technotes/guides/jdbc/>
* <https://dev.mysql.com/doc/>
* <https://docs.oracle.com/javase/8/docs/>