package PROJECT;

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.\*;

import java.sql.\*;

import java.text.ParseException;

import java.text.SimpleDateFormat;

public class PROJECT implements ActionListener {

private JFrame frame;

private JTextField nameField, reasonField, dateField, daysField;

private JButton submitButton;

private final String ADMIN\_PASSWORD = "admin123"; // Set your admin password here

public PROJECT() {

// Show the home page

showHomePage();

}

private void showHomePage() {

JFrame homeFrame = new JFrame("Student Outpass System");

homeFrame.setLayout(new FlowLayout());

homeFrame.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

homeFrame.setSize(300, 200);

JButton studentButton = new JButton("Student");

studentButton.addActionListener(e -> {

homeFrame.dispose();

showStudentForm();

});

JButton adminButton = new JButton("Admin");

adminButton.addActionListener(e -> {

homeFrame.dispose();

showAdminLogin(); // Show admin login instead of directly opening the interface

});

homeFrame.add(studentButton);

homeFrame.add(adminButton);

homeFrame.setLocationRelativeTo(null); // Center on screen

homeFrame.setVisible(true);

}

private void showAdminLogin() {

// Show a dialog for admin password

JPasswordField passwordField = new JPasswordField();

int option = JOptionPane.showConfirmDialog(null, passwordField, "Enter Admin Password", JOptionPane.OK\_CANCEL\_OPTION, JOptionPane.PLAIN\_MESSAGE);

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

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

if (password.equals(ADMIN\_PASSWORD)) {

showAdminInterface();

} else {

JOptionPane.showMessageDialog(null, "Invalid password!", "Error", JOptionPane.ERROR\_MESSAGE);

showHomePage(); // Go back to home page if password is incorrect

}

} else {

showHomePage(); // Go back to home page if canceled

}

}

private void showStudentForm() {

// Create the frame for student request submission

frame = new JFrame("Student Outpass System");

frame.setLayout(new BorderLayout());

frame.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

frame.getContentPane().setBackground(new Color(220, 240, 255)); // Light blue background

// Create a panel for inputs

JPanel inputPanel = new JPanel(new GridBagLayout());

inputPanel.setBackground(new Color(255, 255, 255)); // White background

inputPanel.setBorder(BorderFactory.createEmptyBorder(20, 20, 20, 20));

GridBagConstraints gbc = new GridBagConstraints();

gbc.fill = GridBagConstraints.HORIZONTAL;

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

// Create input fields

gbc.gridx = 0;

gbc.gridy = 0;

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

nameField = new JTextField(15);

gbc.gridx = 1;

inputPanel.add(nameField, gbc);

gbc.gridx = 0;

gbc.gridy = 1;

inputPanel.add(new JLabel("Reason:"), gbc);

reasonField = new JTextField(15);

gbc.gridx = 1;

inputPanel.add(reasonField, gbc);

gbc.gridx = 0;

gbc.gridy = 2;

inputPanel.add(new JLabel("Date (YYYY-MM-DD):"), gbc);

dateField = new JTextField(15);

gbc.gridx = 1;

inputPanel.add(dateField, gbc);

gbc.gridx = 0;

gbc.gridy = 3;

inputPanel.add(new JLabel("Number of Days:"), gbc);

daysField = new JTextField(15);

gbc.gridx = 1;

inputPanel.add(daysField, gbc);

// Submit button

submitButton = new JButton("Submit");

submitButton.setBackground(new Color(100, 200, 100)); // Green background

submitButton.setForeground(Color.WHITE); // White text

submitButton.addActionListener(this);

gbc.gridx = 0;

gbc.gridy = 4;

gbc.gridwidth = 2;

inputPanel.add(submitButton, gbc);

// Add input panel to frame

frame.add(inputPanel, BorderLayout.CENTER);

// Set frame properties

frame.setPreferredSize(new Dimension(400, 300));

frame.pack();

frame.setLocationRelativeTo(null); // Center on screen

frame.setVisible(true);

}

private void showAdminInterface() {

JFrame adminFrame = new JFrame("Admin Interface");

adminFrame.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

adminFrame.setSize(400, 300);

JTextArea textArea = new JTextArea();

textArea.setEditable(false);

JScrollPane scrollPane = new JScrollPane(textArea);

JButton refreshButton = new JButton("Refresh Requests");

refreshButton.addActionListener(e -> {

textArea.setText(getRequests());

});

// Load initial requests

textArea.setText(getRequests());

// Button to accept a request

JButton acceptButton = new JButton("Accept Request");

acceptButton.addActionListener(e -> {

String selectedRequest = getSelectedRequest(textArea);

if (selectedRequest != null) {

updateRequestStatus(selectedRequest, "Accepted");

refreshButton.doClick(); // Refresh the list after accepting

}

});

// Button to deny a request

JButton denyButton = new JButton("Deny Request");

denyButton.addActionListener(e -> {

String selectedRequest = getSelectedRequest(textArea);

if (selectedRequest != null) {

updateRequestStatus(selectedRequest, "Denied");

refreshButton.doClick(); // Refresh the list after denying

}

});

JPanel buttonPanel = new JPanel();

buttonPanel.add(acceptButton);

buttonPanel.add(denyButton);

adminFrame.add(scrollPane, BorderLayout.CENTER);

adminFrame.add(buttonPanel, BorderLayout.SOUTH);

adminFrame.add(refreshButton, BorderLayout.NORTH);

adminFrame.setLocationRelativeTo(null); // Center on screen

adminFrame.setVisible(true);

}

private String getSelectedRequest(JTextArea textArea) {

// Logic to get the selected request (e.g., by highlighting)

// For simplicity, we ask the admin for the name of the student

String input = JOptionPane.showInputDialog("Enter the name of the student to accept/deny:");

return input; // You should enhance this to find the actual request from the list

}

private String getRequests() {

StringBuilder requests = new StringBuilder();

try (Connection conn = DriverManager.getConnection("jdbc:mysql://localhost:3306/students\_outpass", "root", "@Shiva777");

Statement stmt = conn.createStatement();

ResultSet rs = stmt.executeQuery("SELECT \* FROM outpass\_requests")) {

while (rs.next()) {

String name = rs.getString("student\_name");

String reason = rs.getString("reason");

String date = rs.getDate("outpass\_date").toString();

int days = rs.getInt("num\_of\_days");

String status = rs.getString("status");

requests.append(String.format("Name: %s, Reason: %s, Date: %s, Days: %d, Status: %s\n",

name, reason, date, days, status));

}

if (requests.length() == 0) {

requests.append("No requests found.");

}

} catch (SQLException ex) {

ex.printStackTrace();

requests.append("Error fetching requests: " + ex.getMessage());

}

return requests.toString();

}

private void updateRequestStatus(String studentName, String newStatus) {

try (Connection conn = DriverManager.getConnection("jdbc:mysql://localhost:3306/students\_outpass", "root", "@Shiva777");

PreparedStatement stmt = conn.prepareStatement(

"UPDATE outpass\_requests SET status = ? WHERE student\_name = ? AND status = 'Pending'")) {

stmt.setString(1, newStatus);

stmt.setString(2, studentName);

stmt.executeUpdate();

JOptionPane.showMessageDialog(null, "Request " + newStatus + " successfully for " + studentName);

} catch (SQLException ex) {

ex.printStackTrace();

JOptionPane.showMessageDialog(null, "Error updating request status: " + ex.getMessage(), "Database Error", JOptionPane.ERROR\_MESSAGE);

}

}

@Override

public void actionPerformed(ActionEvent e) {

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

String name = nameField.getText().trim();

String reason = reasonField.getText().trim();

String date = dateField.getText().trim();

String days = daysField.getText().trim();

// Input validation

if (name.isEmpty() || reason.isEmpty() || date.isEmpty() || days.isEmpty()) {

JOptionPane.showMessageDialog(frame, "All fields must be filled out.", "Input Error", JOptionPane.ERROR\_MESSAGE);

return;

}

// Validate name length

if (name.length() > 50) {

JOptionPane.showMessageDialog(frame, "Name cannot exceed 50 characters.", "Input Error", JOptionPane.ERROR\_MESSAGE);

return;

}

// Validate reason length

if (reason.length() > 200) {

JOptionPane.showMessageDialog(frame, "Reason cannot exceed 200 characters.", "Input Error", JOptionPane.ERROR\_MESSAGE);

return;

}

// Validate date format

SimpleDateFormat dateFormat = new SimpleDateFormat("yyyy-MM-dd");

dateFormat.setLenient(false);

try {

dateFormat.parse(date);

} catch (ParseException ex) {

JOptionPane.showMessageDialog(frame, "Date must be in the format YYYY-MM-DD.", "Input Error", JOptionPane.ERROR\_MESSAGE);

return;

}

// Validate days as integer and positive value

int numDays;

try {

numDays = Integer.parseInt(days);

if (numDays <= 0 || numDays > 30) { // Assuming a max of 30 days for outpass

JOptionPane.showMessageDialog(frame, "Number of days must be between 1 and 30.", "Input Error", JOptionPane.ERROR\_MESSAGE);

return;

}

} catch (NumberFormatException ex) {

JOptionPane.showMessageDialog(frame, "Number of days must be a valid positive integer.", "Input Error", JOptionPane.ERROR\_MESSAGE);

return;

}

// Insert data into MySQL

saveOutpassRequest(name, reason, date, String.valueOf(numDays));

// Display a pop-up message with the request details

String message = String.format("Outpass Request Submitted:\n\nName: %s\nReason: %s\nDate: %s\nNumber of Days: %s",

name, reason, date, numDays);

JOptionPane.showMessageDialog(frame, message, "Submission Successful", JOptionPane.INFORMATION\_MESSAGE);

// Return to the home page

frame.dispose(); // Close the current student form

showHomePage(); // Show the home page again

}

}

private void saveOutpassRequest(String studentName, String reason, String date, String days) {

try (Connection conn = DriverManager.getConnection("jdbc:mysql://localhost:3306/students\_outpass", "root", "@Shiva777");

PreparedStatement stmt = conn.prepareStatement(

"INSERT INTO outpass\_requests (student\_name, reason, outpass\_date, num\_of\_days, status) VALUES (?, ?, ?, ?, 'Pending')")) {

stmt.setString(1, studentName);

stmt.setString(2, reason);

stmt.setDate(3, Date.valueOf(date));

stmt.setInt(4, Integer.parseInt(days));

stmt.executeUpdate();

System.out.println("Outpass request saved successfully.");

} catch (SQLException ex) {

ex.printStackTrace();

}

}

public static void main(String[] args) {

SwingUtilities.invokeLater(PROJECT::new);

}

}