EMPLOYEE

package cms.Employee;  
  
import cms.LoginPage.LoginForm;  
  
import javax.swing.\*;  
import javax.swing.table.DefaultTableModel;  
import java.awt.\*;  
import java.sql.\*;  
  
public class EmployeeUI extends JFrame {  
  
 private String employeeName;  
 private String empUsername;  
 private int employeeId;  
 private JLabel employeeLabel;  
 private JTable courierTable;  
 private JScrollPane tableScrollPane;  
 private JButton updateButton;  
 private JButton logoutButton;  
  
 public EmployeeUI(int employeeId) { // ✅ Receiving employeeId as int  
 this. EmployeeId = employeeId; // ✅ Correct assignment  
 initializeUI();  
 loadEmployeeDetails(); // ✅ Loading employee details correctly  
 loadCourierTable(); // ✅ Loading assigned courier data correctly  
 }  
  
  
 private void initializeUI() {  
 setTitle("Employee Panel - Courier Management System");  
 setSize(800, 600);  
 setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);  
 setLayout(new BorderLayout());  
  
 JPanel employeePanel = new JPanel();  
 employeePanel.setLayout(new BoxLayout(employeePanel, BoxLayout.Y\_AXIS));  
  
 employeeLabel = new JLabel("Loading employee details...");  
 employeePanel.add(employeeLabel);  
 add(employeePanel, BorderLayout.WEST);  
  
 courierTable = new JTable();  
 tableScrollPane = new JScrollPane(courierTable);  
 add(tableScrollPane, BorderLayout.CENTER);  
  
 JPanel bottomPanel = new JPanel(new FlowLayout(FlowLayout.CENTER));  
 updateButton = new JButton("Update Status");  
 logoutButton = new JButton("Logout");  
  
 bottomPanel.add(updateButton);  
 bottomPanel.add(logoutButton);  
 add(bottomPanel, BorderLayout.SOUTH);  
  
 updateButton.addActionListener(e -> updateSelectedCourierStatus());  
  
 logoutButton.addActionListener(e -> {  
 int confirm = JOptionPane.showConfirmDialog(null, "Are you sure you want to log out?", "Logout Confirmation", JOptionPane.YES\_NO\_OPTION);  
 if (confirm == JOptionPane.YES\_OPTION) {  
 JOptionPane.showMessageDialog(null, "Logging out...");  
 SwingUtilities.invokeLater(() -> new LoginForm().setVisible(true));  
 dispose();  
 }  
 });  
  
 setVisible(true);  
 }  
  
 private void loadEmployeeDetails() {  
 System.out.println("Employee ID received in loadEmployeeDetails: " + employeeId);  
 try (Connection connection = DriverManager.getConnection("jdbc:mysql://localhost:3306/cms", "root", "My134340Database")) {  
 System.out.println("Database connection successful in loadEmployeeDetails.");  
  
 String query = "SELECT emp\_name, emp\_username, contact FROM employee\_table WHERE eid = ?";  
 System.out.println("Executing query: " + query);  
  
 PreparedStatement preparedStatement = connection.prepareStatement(query);  
 System.out.println("Employee ID passed to query: " + employeeId);  
 preparedStatement.setInt(1, employeeId);  
  
 ResultSet resultSet = preparedStatement.executeQuery();  
  
 if (resultSet.next()) {  
 employeeName = resultSet.getString("emp\_name");  
 String email = resultSet.getString("emp\_username");  
 String phone = resultSet.getString("contact");  
  
 SwingUtilities.invokeLater(() -> { // added this  
 if (employeeName != null && email != null && phone != null) { // added this  
 employeeLabel.setText("<html><b>Employee: </b>" + employeeName + "<br>" +  
 "<b>Username: </b>" + email + "<br>" +  
 "<b>Phone: </b>" + phone + "</html>");  
 employeeLabel.revalidate();  
 employeeLabel.repaint();  
 } else {  
 employeeLabel.setText("Employee details not available.");  
 }  
 });  
  
 System.out.println("Employee Details Loaded:");  
 System.out.println("Name: " + employeeName);  
 System.out.println("Username: " + email);  
 System.out.println("Phone: " + phone);  
 } else {  
 JOptionPane.showMessageDialog(this, "Employee details not found!");  
 System.out.println("No employee details found for eid: " + employeeId);  
 }  
 } catch (SQLException ex) {  
 ex.printStackTrace();  
 JOptionPane.showMessageDialog(this, "Error loading employee details: " + ex.getMessage());  
 } catch (Exception ex) {  
 ex.printStackTrace();  
 JOptionPane.showMessageDialog(this, "An unexpected error occurred: " + ex.getMessage());  
 }  
 }  
 private void loadCourierTable() {  
 SwingUtilities.invokeLater(() -> {  
 try (Connection connection = DriverManager.getConnection("jdbc:mysql://localhost:3306/cms", "root", "My134340Database")) {  
 System.out.println("Database connection successful in loadCourierTable.");  
  
 String query = "SELECT \* FROM courier\_table WHERE assigned\_eid = ?";  
 System.out.println("Executing query: " + query);  
  
 PreparedStatement preparedStatement = connection.prepareStatement(query);  
 System.out.println("Employee ID used for courier query: " + employeeId);  
 preparedStatement.setInt(1, employeeId);  
  
 ResultSet resultSet = preparedStatement.executeQuery();  
  
 DefaultTableModel model = new DefaultTableModel();  
 ResultSetMetaData metaData = resultSet.getMetaData();  
 int columnCount = metaData.getColumnCount();  
  
 for (int i = 1; i <= columnCount; i++) {  
 model.addColumn(metaData.getColumnName(i));  
 }  
  
 while (resultSet.next()) {  
 Object[] row = new Object[columnCount]; // Corrected declaration  
 for (int i = 1; i <= columnCount; i++) {  
 row[i - 1] = resultSet.getObject(i);  
 }  
 model.addRow(row);  
 }  
  
 courierTable.setModel(model);  
 System.out.println("Courier table loaded.");  
  
 } catch (SQLException ex) {  
 ex.printStackTrace();  
 JOptionPane.showMessageDialog(this, "Error loading courier data: " + ex.getMessage());  
 } catch (Exception ex) {  
 ex.printStackTrace();  
 JOptionPane.showMessageDialog(this, "An unexpected error occurred: " + ex.getMessage());  
 }  
 });  
 }  
 private void updateSelectedCourierStatus() {  
 int selectedRow = courierTable.getSelectedRow();  
 if (selectedRow == -1) {  
 JOptionPane.showMessageDialog(this, "Please select a row to update.");  
 return;  
 }  
  
 String newStatus = JOptionPane.showInputDialog(this, "Enter new status:");  
 if (newStatus == null || newStatus.trim().isEmpty()) {  
 JOptionPane.showMessageDialog(this, "Status cannot be empty.");  
 return;  
 }  
  
 int cid = (int) courierTable.getValueAt(selectedRow, 0);  
 updateCourierStatus(cid, newStatus);  
 }  
  
 private void updateCourierStatus(int cid, String newStatus) {  
 try (Connection connection = DriverManager.getConnection("jdbc:mysql://localhost:3306/cms", "root", "My134340Database")) {  
 System.out.println("Database connection successful in updateCourierStatus.");  
  
 String updateQuery = "UPDATE courier\_table SET status = ? WHERE cid = ?";  
 System.out.println("Executing query: " + updateQuery);  
  
 PreparedStatement preparedStatement = connection.prepareStatement(updateQuery);  
 preparedStatement.setString(1, newStatus);  
 preparedStatement.setInt(2, cid);  
  
 int rowsAffected = preparedStatement.executeUpdate();  
 if (rowsAffected > 0) {  
 JOptionPane.showMessageDialog(this, "Status updated successfully.");  
 loadCourierTable();  
 } else {  
 JOptionPane.showMessageDialog(this, "No rows affected. Please try again.");  
 }  
 } catch (SQLException ex) {  
 ex.printStackTrace();  
 JOptionPane.showMessageDialog(this, "Error updating status: " + ex.getMessage());  
 } catch (Exception ex) {  
 ex.printStackTrace();  
 JOptionPane.showMessageDialog(this, "An unexpected error occurred: " + ex.getMessage());  
 }  
 }  
  
 public static void main(String[] args) {  
  
 int employeeId = 203;  
 SwingUtilities.invokeLater(() -> new EmployeeUI(employeeId).setVisible(true));  
 }  
}

ADMIN

package cms.Admin;  
  
import cms.LoginPage.LoginForm;  
  
import javax.swing.\*;  
import javax.swing.table.DefaultTableModel;  
import java.awt.\*;  
import java.awt.event.ActionEvent;  
import java.awt.event.ActionListener;  
import java.sql.\*;  
  
public class AdminUI extends JFrame {  
  
 private String adminName;  
 private String adminEmail;  
 private String adminPhone;  
 private JButton employeeButton;  
 private JButton courierButton;  
 private JButton customerButton;  
 private JLabel adminLabel;  
 private JTable dataTable;  
 private JScrollPane tableScrollPane;  
 private JButton logoutButton;  
 private JButton updateButton;  
  
 // Variable to track the selected table name  
 private String currentTableName;  
  
 // Updated Constructor to Pass Admin Details  
 public AdminUI(String adminName) {  
 this.adminName = adminName;  
 this.adminEmail = adminEmail;  
 this.adminPhone = adminPhone;  
 initializeUI();  
 loadAdminDetails(); // Load admin details dynamically  
 }  
  
 private void initializeUI() {  
 setTitle("Admin Panel - Courier Management System");  
 setSize(800, 600);  
 setDefaultCloseOperation(JFrame.*EXIT\_ON\_CLOSE*);  
 setLayout(new BorderLayout());  
  
 // Top Panel for Admin Info  
 JPanel topPanel = new JPanel(new GridLayout(3, 1));  
 adminLabel = new JLabel();  
 loadAdminDetails(); // Load and display admin info  
 topPanel.add(adminLabel);  
 add(topPanel, BorderLayout.*NORTH*);  
  
 // Left Panel with Buttons  
 JPanel buttonPanel = new JPanel();  
 buttonPanel.setLayout(new BoxLayout(buttonPanel, BoxLayout.*Y\_AXIS*));  
  
 employeeButton = new JButton("Employees");  
 courierButton = new JButton("Couriers");  
 customerButton = new JButton("Customers");  
  
 employeeButton.setAlignmentX(Component.*CENTER\_ALIGNMENT*);  
 courierButton.setAlignmentX(Component.*CENTER\_ALIGNMENT*);  
 customerButton.setAlignmentX(Component.*CENTER\_ALIGNMENT*);  
  
 buttonPanel.add(employeeButton);  
 buttonPanel.add(Box.*createVerticalStrut*(10));  
 buttonPanel.add(courierButton);  
 buttonPanel.add(Box.*createVerticalStrut*(10));  
 buttonPanel.add(customerButton);  
  
 add(buttonPanel, BorderLayout.*WEST*);  
  
 // Center Panel for Data Table  
 dataTable = new JTable();  
 tableScrollPane = new JScrollPane(dataTable);  
 add(tableScrollPane, BorderLayout.*CENTER*);  
  
 // Bottom Panel for Update & Logout Buttons  
 JPanel bottomPanel = new JPanel(new FlowLayout(FlowLayout.*CENTER*));  
 updateButton = new JButton("Update");  
 logoutButton = new JButton("Logout");  
  
 bottomPanel.add(updateButton);  
 bottomPanel.add(logoutButton);  
  
 add(bottomPanel, BorderLayout.*SOUTH*);  
  
 // Action Listeners for Buttons  
 employeeButton.addActionListener(new ActionListener() {  
 @Override  
 public void actionPerformed(ActionEvent e) {  
 currentTableName = "employee\_table";  
 loadTableData(currentTableName);  
 }  
 });  
  
 courierButton.addActionListener(new ActionListener() {  
 @Override  
 public void actionPerformed(ActionEvent e) {  
 currentTableName = "courier\_table";  
 loadTableData(currentTableName);  
 }  
 });  
  
 customerButton.addActionListener(new ActionListener() {  
 @Override  
 public void actionPerformed(ActionEvent e) {  
 currentTableName = "customer\_table";  
 loadTableData(currentTableName);  
 }  
 });  
  
 updateButton.addActionListener(new ActionListener() {  
 @Override  
 public void actionPerformed(ActionEvent e) {  
 updateSelectedTable();  
 }  
 });  
  
 // Updated Logout Button to Redirect to Login Page  
 logoutButton.addActionListener(new ActionListener() {  
 @Override  
 public void actionPerformed(ActionEvent e) {  
 int confirm = JOptionPane.*showConfirmDialog*(null, "Are you sure you want to log out?", "Logout Confirmation", JOptionPane.*YES\_NO\_OPTION*);  
 if (confirm == JOptionPane.*YES\_OPTION*) {  
 JOptionPane.*showMessageDialog*(null, "Logging out...");  
  
 // Open LoginForm and close the AdminUI  
 SwingUtilities.*invokeLater*(() -> {  
 new LoginForm().setVisible(true); // Open Login Page  
 });  
 dispose(); // Close current window  
 }  
 }  
 });  
  
 setVisible(true);  
 }  
  
 // Updated to Display Admin Details  
 private void loadAdminDetails() {  
 adminLabel.setText("<html><b>Admin Info:</b><br>Name: " + adminName + "<br>Email: " + adminEmail + "<br>Phone: " + adminPhone + "</html>");  
 }  
  
 // Load Table Data from Database  
 private void loadTableData(String tableName) {  
 SwingUtilities.*invokeLater*(() -> {  
 try (Connection connection = DriverManager.*getConnection*("jdbc:mysql://localhost:3306/cms", "root", "My134340Database");  
 Statement statement = connection.createStatement()) {  
  
 String query = "SELECT \* FROM " + tableName;  
  
 try (ResultSet resultSet = statement.executeQuery(query)) {  
 DefaultTableModel model = new DefaultTableModel();  
 ResultSetMetaData metaData = resultSet.getMetaData();  
 int columnCount = metaData.getColumnCount();  
  
 // Add Column Names to Table  
 for (int i = 1; i <= columnCount; i++) {  
 model.addColumn(metaData.getColumnName(i));  
 }  
  
 // Add Data Rows to Table  
 while (resultSet.next()) {  
 Object[] row = new Object[columnCount];  
 for (int i = 1; i <= columnCount; i++) {  
 row[i - 1] = resultSet.getObject(i);  
 }  
 model.addRow(row);  
 }  
 dataTable.setModel(model);  
 }  
  
 } catch (SQLException ex) {  
 ex.printStackTrace();  
 JOptionPane.*showMessageDialog*(this, "Error loading data: " + ex.getMessage());  
 }  
 });  
 }  
  
 // Update Assigned Courier or Employee  
 private void updateSelectedTable() {  
 if (dataTable.getSelectedRow() == -1) {  
 JOptionPane.*showMessageDialog*(this, "Please select a row to update.");  
 return;  
 }  
  
 if (currentTableName != null) {  
 if (currentTableName.equals("courier\_table")) {  
 updateAssignedEid();  
 } else if (currentTableName.equals("employee\_table")) {  
 updateAssignedCourier();  
 } else {  
 JOptionPane.*showMessageDialog*(this, "Updates are not allowed for this table.");  
 }  
 } else {  
 JOptionPane.*showMessageDialog*(this, "No table selected.");  
 }  
 }  
  
 // Update Assigned Eid in courier\_table  
 private void updateAssignedEid() {  
 int selectedRow = dataTable.getSelectedRow();  
 DefaultTableModel model = (DefaultTableModel) dataTable.getModel();  
  
 int cid = (int) model.getValueAt(selectedRow, 0); // CID  
 String newAssignedEidString = JOptionPane.*showInputDialog*(this, "Enter new assigned Eid:");  
  
 if (newAssignedEidString == null || newAssignedEidString.trim().isEmpty()) {  
 JOptionPane.*showMessageDialog*(this, "Assigned eid cannot be empty.");  
 return;  
 }  
  
 try {  
 int newAssignedEid = Integer.*parseInt*(newAssignedEidString);  
  
 try (Connection connection = DriverManager.*getConnection*("jdbc:mysql://localhost:3306/cms", "root", "My134340Database");  
 PreparedStatement preparedStatement = connection.prepareStatement("UPDATE courier\_table SET assigned\_eid = ? WHERE cid = ?")) {  
  
 preparedStatement.setInt(1, newAssignedEid);  
 preparedStatement.setInt(2, cid);  
  
 int rowsAffected = preparedStatement.executeUpdate();  
  
 if (rowsAffected > 0) {  
 JOptionPane.*showMessageDialog*(this, "Assigned eid updated successfully.");  
 loadTableData("courier\_table");  
 } else {  
 JOptionPane.*showMessageDialog*(this, "Update failed. No rows updated.");  
 }  
 }  
 } catch (NumberFormatException e) {  
 JOptionPane.*showMessageDialog*(this, "Invalid input. Please enter a valid integer for assigned eid.");  
 } catch (SQLException ex) {  
 ex.printStackTrace();  
 JOptionPane.*showMessageDialog*(this, "Error updating assigned eid: " + ex.getMessage());  
 }  
 }  
  
 // Update Assigned Courier in employee\_table  
 private void updateAssignedCourier() {  
 int selectedRow = dataTable.getSelectedRow();  
 DefaultTableModel model = (DefaultTableModel) dataTable.getModel();  
  
 int employeeId = (int) model.getValueAt(selectedRow, 0);  
 String newAssignedCourier = JOptionPane.*showInputDialog*(this, "Enter new assigned courier:");  
  
 if (newAssignedCourier != null && !newAssignedCourier.isEmpty()) {  
 try (Connection connection = DriverManager.*getConnection*("jdbc:mysql://localhost:3306/cms", "root", "My134340Database");  
 PreparedStatement preparedStatement = connection.prepareStatement("UPDATE employee\_table SET assigned\_courier = ? WHERE eid = ?")) {  
  
 preparedStatement.setString(1, newAssignedCourier);  
 preparedStatement.setInt(2, employeeId);  
  
 int rowsAffected = preparedStatement.executeUpdate();  
  
 if (rowsAffected > 0) {  
 JOptionPane.*showMessageDialog*(this, "Assigned courier updated successfully.");  
 loadTableData("employee\_table");  
 } else {  
 JOptionPane.*showMessageDialog*(this, "Update failed. No rows updated.");  
 }  
 } catch (SQLException ex) {  
 ex.printStackTrace();  
 JOptionPane.*showMessageDialog*(this, "Error updating assigned courier: " + ex.getMessage());  
 }  
 } else {  
 JOptionPane.*showMessageDialog*(this, "Assigned courier cannot be empty.");  
 }  
 }  
  
 public static void main(String[] args) {  
 SwingUtilities.*invokeLater*(() -> {  
 // Dummy Data for Testing  
 AdminUI adminUI = new AdminUI( "John Doe");  
 adminUI.setVisible(true);  
 });  
 }  
}

CUSTOMER

package cms.Customer;  
  
import cms.LoginPage.LoginForm;  
  
import javax.swing.\*;  
import javax.swing.table.DefaultTableModel;  
import java.awt.\*;  
import java.sql.\*;  
  
public class CustomerUI extends JFrame {  
  
 private String custUsername;  
 private int customerId;  
 private JLabel customerDetailsLabel;  
 private JTable courierTable;  
 private DefaultTableModel tableModel;  
 private JButton addCourierButton;  
 private JButton logoutButton;  
  
 public CustomerUI(String custUsername) {  
 this.custUsername = custUsername;  
 initializeUI();  
 loadCustomerDetails();  
 loadCourierData();  
 }  
  
 private void initializeUI() {  
 setTitle("Customer Panel - Courier Management System");  
 setSize(800, 600);  
 setDefaultCloseOperation(JFrame.*EXIT\_ON\_CLOSE*);  
 setLayout(new BorderLayout(10, 10));  
  
 // Panel for customer details  
 JPanel customerDetailsPanel = new JPanel(new FlowLayout(FlowLayout.*LEFT*));  
 customerDetailsLabel = new JLabel("Loading customer details...");  
 customerDetailsPanel.add(customerDetailsLabel);  
 add(customerDetailsPanel, BorderLayout.*NORTH*); // Add customer details panel to the top  
  
 // Table for courier data  
 tableModel = new DefaultTableModel();  
 courierTable = new JTable(tableModel);  
 JScrollPane tableScrollPane = new JScrollPane(courierTable);  
 add(tableScrollPane, BorderLayout.*CENTER*); // Add table to the center  
  
 // Panel for buttons  
 JPanel buttonPanel = new JPanel(new FlowLayout(FlowLayout.*CENTER*));  
 addCourierButton = new JButton("Add Courier");  
 addCourierButton.addActionListener(e -> openAddCourierForm());  
 buttonPanel.add(addCourierButton);  
  
 logoutButton = new JButton("Logout");  
 logoutButton.addActionListener(e -> logout());  
 buttonPanel.add(logoutButton);  
  
 add(buttonPanel, BorderLayout.*SOUTH*); // Add button panel to the bottom  
  
 revalidate();  
 repaint();  
  
 setVisible(true);  
 }  
  
 private void loadCustomerDetails() {  
 try (Connection connection = DriverManager.*getConnection*("jdbc:mysql://localhost:3306/cms", "root", "My134340Database")) {  
 String query = "SELECT \* FROM customer\_table WHERE cust\_username = ?";  
 PreparedStatement preparedStatement = connection.prepareStatement(query);  
 preparedStatement.setString(1, custUsername);  
 ResultSet resultSet = preparedStatement.executeQuery();  
  
 if (resultSet.next()) {  
 customerId = resultSet.getInt("cust\_id");  
 String customerName = resultSet.getString("cust\_name");  
 String email = resultSet.getString("cust\_username");  
 String phone = resultSet.getString("contact");  
 String address = resultSet.getString("address");  
  
 // Update the label to show customer details (Corrected)  
 customerDetailsLabel.setText("<html><b>Customer Name: </b>" + customerName + "<br>" +  
 "<b>Username: </b>" + email + "<br>" +  
 "<b>Contact: </b>" + phone + "<br>" + // Corrected: Single Contact Label  
 "<b>Address: </b>" + address + "</html>"); // Corrected: Added Address Label  
  
 System.*out*.println("Customer found! ID: " + customerId + ", Name: " + customerName);  
 } else {  
 JOptionPane.*showMessageDialog*(this, "Customer details not found!");  
 System.*out*.println("Customer details not found for username: " + custUsername);  
 }  
  
 } catch (SQLException ex) {  
 ex.printStackTrace();  
 JOptionPane.*showMessageDialog*(this, "Error loading customer details: " + ex.getMessage());  
 System.*err*.println("SQL Exception in loadCustomerDetails: " + ex.getMessage());  
 } catch (Exception generalException) {  
 generalException.printStackTrace();  
 JOptionPane.*showMessageDialog*(this, "An unexpected error occurred: " + generalException.getMessage());  
 System.*err*.println("General Exception in loadCustomerDetails: " + generalException.getMessage());  
 }  
 }  
  
 private void loadCourierData() {  
 SwingUtilities.*invokeLater*(() -> {  
 try (Connection connection = DriverManager.*getConnection*("jdbc:mysql://localhost:3306/cms", "root", "My134340Database")) {  
 String query = "SELECT sender\_name, receiver\_name, pickup\_add, destination, weight, status FROM courier\_table WHERE cust\_id = ?";  
 PreparedStatement preparedStatement = connection.prepareStatement(query);  
 preparedStatement.setInt(1, customerId);  
 ResultSet resultSet = preparedStatement.executeQuery();  
  
 ResultSetMetaData metaData = resultSet.getMetaData();  
 int columnCount = metaData.getColumnCount();  
  
 tableModel.setColumnCount(0);  
 for (int i = 1; i <= columnCount; i++) {  
 tableModel.addColumn(metaData.getColumnName(i));  
 }  
  
 tableModel.setRowCount(0);  
 while (resultSet.next()) {  
 Object[] row = new Object[columnCount];  
 for (int i = 1; i <= columnCount; i++) {  
 row[i - 1] = resultSet.getObject(i);  
 }  
 tableModel.addRow(row);  
 }  
  
 tableModel.fireTableDataChanged();  
 System.*out*.println("Courier data loaded successfully!");  
  
 } catch (SQLException ex) {  
 ex.printStackTrace();  
 JOptionPane.*showMessageDialog*(this, "Error loading courier data: " + ex.getMessage());  
 }  
 });  
 }  
  
 private void openAddCourierForm() {  
 JTextField senderNameField = new JTextField(20);  
 JTextField receiverNameField = new JTextField(20);  
 JTextField pickupAddressField = new JTextField(20);  
 JTextField destinationField = new JTextField(20);  
 JTextField weightField = new JTextField(20);  
  
 JPanel panel = new JPanel(new GridLayout(5, 2, 10, 10));  
 panel.add(new JLabel("Sender Name:"));  
 panel.add(senderNameField);  
 panel.add(new JLabel("Receiver Name:"));  
 panel.add(receiverNameField);  
 panel.add(new JLabel("Pickup Address:"));  
 panel.add(pickupAddressField);  
 panel.add(new JLabel("Destination:"));  
 panel.add(destinationField);  
 panel.add(new JLabel("Weight (kg):"));  
 panel.add(weightField);  
  
 int result = JOptionPane.*showConfirmDialog*(null, panel, "Add New Courier", JOptionPane.*OK\_CANCEL\_OPTION*);  
 if (result == JOptionPane.*OK\_OPTION*) {  
 String senderName = senderNameField.getText().trim();  
 String receiverName = receiverNameField.getText().trim();  
 String pickupAdd = pickupAddressField.getText().trim();  
 String destination = destinationField.getText().trim();  
 String weightStr = weightField.getText().trim();  
  
 if (senderName.isEmpty() || receiverName.isEmpty() || pickupAdd.isEmpty() || destination.isEmpty() || weightStr.isEmpty()) {  
 JOptionPane.*showMessageDialog*(this, "Please fill all fields.");  
 return;  
 }  
  
 try {  
 double weight = Double.*parseDouble*(weightStr);  
 addCourierToDatabase(senderName, receiverName, pickupAdd, destination, weight);  
 } catch (NumberFormatException e) {  
 JOptionPane.*showMessageDialog*(this, "Weight must be a valid number.");  
 }  
 }  
 }  
  
 private void addCourierToDatabase(String senderName, String receiverName, String pickupAdd, String destination, double weight) {  
 try (Connection connection = DriverManager.*getConnection*("jdbc:mysql://localhost:3306/cms", "root", "My134340Database")) {  
  
 String insertQuery = "INSERT INTO courier\_table (sender\_name, receiver\_name, pickup\_add, destination, weight, status, cust\_id) VALUES (?, ?, ?, ?, ?, ?, ?)";  
 PreparedStatement preparedStatement = connection.prepareStatement(insertQuery);  
 preparedStatement.setString(1, senderName);  
 preparedStatement.setString(2, receiverName);  
 preparedStatement.setString(3, pickupAdd);  
 preparedStatement.setString(4, destination);  
 preparedStatement.setDouble(5, weight);  
 preparedStatement.setString(6, "Pending");  
 preparedStatement.setInt(7, customerId);  
  
 int rowsAffected = preparedStatement.executeUpdate();  
 if (rowsAffected > 0) {  
 JOptionPane.*showMessageDialog*(this, "Courier added successfully!");  
 loadCourierData();  
 } else {  
 JOptionPane.*showMessageDialog*(this, "Failed to add courier. Please try again.");  
 }  
  
 } catch (SQLException ex) {  
 ex.printStackTrace();  
 JOptionPane.*showMessageDialog*(this, "Error adding courier: " + ex.getMessage());  
 }  
 }  
  
 private void logout() {  
 int confirm = JOptionPane.*showConfirmDialog*(this, "Are you sure you want to log out?", "Logout Confirmation", JOptionPane.*YES\_NO\_OPTION*);  
 if (confirm == JOptionPane.*YES\_OPTION*) {  
 JOptionPane.*showMessageDialog*(this, "Logging out...");  
 SwingUtilities.*invokeLater*(() -> new LoginForm().setVisible(true));  
 dispose();  
 }  
 }  
 public static void main(String[] args) {  
 SwingUtilities.*invokeLater*(() -> new CustomerUI("krystal123@gmail.com")); // Replace with a valid username from your database  
 }  
}

Login

package cms.LoginPage;  
  
import cms.RegisterPage.RegistrationForm;  
import cms.Admin.AdminUI;  
import cms.Employee.EmployeeUI;  
import cms.Customer.CustomerUI;  
  
import javax.swing.\*;  
import java.awt.\*;  
import java.awt.event.ActionEvent;  
import java.awt.event.ActionListener;  
import java.sql.\*;  
  
public class LoginForm extends JFrame {  
 private JTextField emailField;  
 private JPasswordField passwordField;  
 private JComboBox<String> userTypeComboBox;  
 private JLabel registerLabel;  
  
 public LoginForm() {  
 setTitle("Courier Management System - Login");  
 setSize(500, 350);  
 setDefaultCloseOperation(JFrame.*EXIT\_ON\_CLOSE*);  
 setLocationRelativeTo(null);  
 setLayout(new GridBagLayout());  
  
 JLabel emailLabel = new JLabel("Email/Username:");  
 emailLabel.setFont(new Font("Arial", Font.*BOLD*, 16));  
  
 emailField = new JTextField();  
 emailField.setPreferredSize(new Dimension(250, 35));  
 emailField.setFont(new Font("Arial", Font.*PLAIN*, 16));  
  
 JLabel passwordLabel = new JLabel("Password:");  
 passwordLabel.setFont(new Font("Arial", Font.*BOLD*, 16));  
  
 passwordField = new JPasswordField();  
 passwordField.setPreferredSize(new Dimension(250, 35));  
 passwordField.setFont(new Font("Arial", Font.*PLAIN*, 16));  
  
 JLabel userTypeLabel = new JLabel("User Type:");  
 userTypeLabel.setFont(new Font("Arial", Font.*BOLD*, 16));  
  
 userTypeComboBox = new JComboBox<>(new String[]{"Select User Type", "Admin", "Employee", "Customer"});  
 userTypeComboBox.setPreferredSize(new Dimension(250, 35));  
 userTypeComboBox.setFont(new Font("Arial", Font.*PLAIN*, 16));  
  
 JButton loginButton = new JButton("Login");  
 loginButton.setPreferredSize(new Dimension(150, 40));  
 loginButton.setBackground(new Color(59, 89, 182));  
 loginButton.setForeground(Color.*WHITE*);  
 loginButton.setFont(new Font("Arial", Font.*BOLD*, 16));  
 loginButton.addActionListener(new LoginAction());  
  
 registerLabel = new JLabel("<html><u>Register</u></html>");  
 registerLabel.setForeground(Color.*BLUE*);  
 registerLabel.setCursor(Cursor.*getPredefinedCursor*(Cursor.*HAND\_CURSOR*));  
 registerLabel.addMouseListener(new java.awt.event.MouseAdapter() {  
 public void mouseClicked(java.awt.event.MouseEvent evt) {  
 openRegisterPage();  
 }  
 });  
  
 // Layout setup  
 GridBagConstraints gbc = new GridBagConstraints();  
 gbc.gridx = 0;  
 gbc.gridy = 0;  
 gbc.insets = new Insets(10, 10, 10, 10);  
 gbc.anchor = GridBagConstraints.*WEST*;  
  
 add(emailLabel, gbc);  
  
 gbc.gridx = 1;  
 gbc.fill = GridBagConstraints.*HORIZONTAL*;  
 add(emailField, gbc);  
  
 gbc.gridx = 0;  
 gbc.gridy = 1;  
 add(passwordLabel, gbc);  
  
 gbc.gridx = 1;  
 add(passwordField, gbc);  
  
 gbc.gridx = 0;  
 gbc.gridy = 2;  
 add(userTypeLabel, gbc);  
  
 gbc.gridx = 1;  
 add(userTypeComboBox, gbc);  
  
 gbc.gridx = 1;  
 gbc.gridy = 3;  
 add(loginButton, gbc);  
  
 gbc.gridx = 1;  
 gbc.gridy = 4;  
 add(registerLabel, gbc);  
 }  
  
 // Open registration page  
 private void openRegisterPage() {  
 RegistrationForm registerPage = new RegistrationForm();  
 registerPage.setVisible(true);  
 this.setVisible(false);  
 }  
  
 // Login action to authenticate and navigate  
 private class LoginAction implements ActionListener {  
 @Override  
 public void actionPerformed(ActionEvent e) {  
 String email = emailField.getText().trim().toLowerCase();  
 String password = new String(passwordField.getPassword());  
 String userType = (String) userTypeComboBox.getSelectedItem();  
  
 if (userType.equals("Select User Type")) {  
 JOptionPane.*showMessageDialog*(null, "Please select a user type.");  
 return;  
 }  
  
 String tableName, usernameColumn, passwordColumn, nameColumn, idColumn;  
  
 switch (userType.toLowerCase()) {  
 case "admin":  
 tableName = "admin";  
 usernameColumn = "admin\_username";  
 passwordColumn = "admin\_password";  
 nameColumn = "admin\_name";  
 idColumn = "admin\_id";  
 break;  
 case "employee":  
 tableName = "employee\_table";  
 usernameColumn = "emp\_username";  
 passwordColumn = "emp\_password";  
 nameColumn = "emp\_name";  
 idColumn = "eid";  
 break;  
 case "customer":  
 tableName = "customer\_table";  
 usernameColumn = "cust\_username";  
 passwordColumn = "cust\_password";  
 nameColumn = "cust\_name";  
 idColumn = "cust\_id";  
 break;  
 default:  
 JOptionPane.*showMessageDialog*(null, "Invalid user type.");  
 return;  
 }  
  
 try (Connection conn = DriverManager.*getConnection*("jdbc:mysql://localhost:3306/cms", "root", "My134340Database")) {  
 String query = "SELECT \* FROM " + tableName + " WHERE LOWER(" + usernameColumn + ") = ? AND " + passwordColumn + " = ?";  
 System.*out*.println("Executing query: " + query);  
 try (PreparedStatement stmt = conn.prepareStatement(query)) {  
 stmt.setString(1, email);  
 stmt.setString(2, password);  
 ResultSet rs = stmt.executeQuery();  
  
 if (rs.next()) {  
 System.*out*.println("Result Set Data:");  
 for (int i = 1; i <= rs.getMetaData().getColumnCount(); i++) {  
 System.*out*.println(rs.getMetaData().getColumnName(i) + ": " + rs.getString(i));  
 }  
 String name = rs.getString(nameColumn);  
 int userId = rs.getInt(idColumn);  
 System.*out*.println("User ID (eid) retrieved from database: " + userId);  
  
 JOptionPane.*showMessageDialog*(null, "Welcome " + userType + ", " + name + "!");  
  
 switch (userType.toLowerCase()) {  
 case "admin":  
 new AdminUI(name);  
 break;  
 case "employee":  
 try {  
 SwingUtilities.*invokeLater*(() -> new EmployeeUI(userId).setVisible(true)); // Corrected line  
 } catch (Exception employeeUiException) {  
 employeeUiException.printStackTrace();  
 JOptionPane.*showMessageDialog*(null, "Error launching employee UI: " + employeeUiException.getMessage());  
 }  
 break;  
 case "customer":  
 try {  
 new CustomerUI(rs.getString("cust\_username"));  
 } catch (Exception customerUiException) {  
 customerUiException.printStackTrace();  
 JOptionPane.*showMessageDialog*(null, "Error launching customer UI: " + customerUiException.getMessage());  
 }  
 break;  
 }  
 dispose();  
 } else {  
 JOptionPane.*showMessageDialog*(null, "Invalid username or password.");  
 }  
 }  
 } catch (SQLException ex) {  
 ex.printStackTrace();  
 JOptionPane.*showMessageDialog*(null, "Database connection error: " + ex.getMessage());  
 } catch (Exception generalException) {  
 generalException.printStackTrace();  
 JOptionPane.*showMessageDialog*(null, "An unexpected error occurred: " + generalException.getMessage());  
 }  
 }  
 }  
  
 public static void main(String[] args) {  
 SwingUtilities.*invokeLater*(() -> new LoginForm().setVisible(true));  
 }  
}

REGISTER

package cms.RegisterPage;  
  
import cms.LoginPage.LoginForm;  
import javax.swing.\*;  
import java.awt.\*;  
import java.awt.event.ActionEvent;  
import java.awt.event.ActionListener;  
import java.sql.Connection;  
import java.sql.DriverManager;  
import java.sql.PreparedStatement;  
  
public class RegistrationForm extends JFrame {  
  
 private JTextField usernameField, nameField, contactField, addressField;  
 private JPasswordField passwordField;  
 private JComboBox<String> userTypeComboBox;  
 private LoginForm loginForm; // Reference to the LoginForm  
  
 // Constructor that accepts a LoginForm instance  
 public RegistrationForm(LoginForm loginForm) {  
 this.loginForm = loginForm; // Store the reference  
 initializeUI();  
 }  
  
 private void initializeUI() {  
 setTitle("Registration Form");  
 setSize(500, 500);  
 setDefaultCloseOperation(JFrame.*EXIT\_ON\_CLOSE*);  
 setLayout(new GridBagLayout());  
  
 GridBagConstraints gbc = new GridBagConstraints();  
 gbc.gridx = 0;  
 gbc.gridy = 0;  
 gbc.insets = new Insets(10, 10, 10, 10);  
 gbc.anchor = GridBagConstraints.*WEST*;  
  
 add(new JLabel("Username:"), gbc);  
  
 gbc.gridx = 1;  
 gbc.fill = GridBagConstraints.*HORIZONTAL*;  
 usernameField = new JTextField();  
 add(usernameField, gbc);  
  
 gbc.gridx = 0;  
 gbc.gridy = 1;  
 gbc.fill = GridBagConstraints.*NONE*;  
 add(new JLabel("Password:"), gbc);  
  
 gbc.gridx = 1;  
 gbc.fill = GridBagConstraints.*HORIZONTAL*;  
 passwordField = new JPasswordField();  
 add(passwordField, gbc);  
  
 gbc.gridx = 0;  
 gbc.gridy = 2;  
 add(new JLabel("Name:"), gbc);  
  
 gbc.gridx = 1;  
 gbc.fill = GridBagConstraints.*HORIZONTAL*;  
 nameField = new JTextField();  
 add(nameField, gbc);  
  
 gbc.gridx = 0;  
 gbc.gridy = 3;  
 add(new JLabel("Contact:"), gbc);  
  
 gbc.gridx = 1;  
 contactField = new JTextField();  
 add(contactField, gbc);  
  
 gbc.gridx = 0;  
 gbc.gridy = 4;  
 add(new JLabel("Address:"), gbc);  
  
 gbc.gridx = 1;  
 addressField = new JTextField();  
 add(addressField, gbc);  
  
 gbc.gridx = 0;  
 gbc.gridy = 5;  
 add(new JLabel("User Type:"), gbc);  
  
 gbc.gridx = 1;  
 userTypeComboBox = new JComboBox<>(new String[]{"Customer", "Employee"});  
 add(userTypeComboBox, gbc);  
  
 gbc.gridx = 0;  
 gbc.gridy = 6;  
 gbc.gridwidth = 2;  
 gbc.anchor = GridBagConstraints.*CENTER*;  
 JButton registerButton = new JButton("Register");  
 registerButton.addActionListener(new ActionListener() {  
 @Override  
 public void actionPerformed(ActionEvent e) {  
 registerUser();  
 }  
 });  
 add(registerButton, gbc);  
  
 // Add Login Link to Return to LoginForm  
 gbc.gridy = 7;  
 JLabel loginLinkLabel = new JLabel("<html><a href='#'>Already have an account? Login here.</a></html>");  
 loginLinkLabel.setCursor(new Cursor(Cursor.*HAND\_CURSOR*));  
 loginLinkLabel.addMouseListener(new java.awt.event.MouseAdapter() {  
 public void mouseClicked(java.awt.event.MouseEvent evt) {  
 openLoginPage();  
 }  
 });  
 add(loginLinkLabel, gbc);  
  
 // Force layout update  
 revalidate();  
 repaint();  
  
 setVisible(true);  
 }  
  
 private void openLoginPage() {  
 System.*out*.println("Opening LoginForm...");  
 loginForm.setVisible(true); // Show the LoginForm  
 dispose(); // Close the RegistrationForm  
 }  
  
 private void registerUser() {  
 String username = usernameField.getText().trim();  
 String password = new String(passwordField.getPassword()).trim();  
 String name = nameField.getText().trim();  
 String contact = contactField.getText().trim();  
 String address = addressField.getText().trim();  
 String userType = (String) userTypeComboBox.getSelectedItem();  
  
 if (username.isEmpty() || password.isEmpty() || name.isEmpty() || contact.isEmpty() || address.isEmpty()) {  
 JOptionPane.*showMessageDialog*(this, "Please fill all fields.");  
 return;  
 }  
  
 int confirm = JOptionPane.*showConfirmDialog*(this, "Are you sure you want to register?", "Confirm Registration", JOptionPane.*YES\_NO\_OPTION*);  
 if (confirm != JOptionPane.*YES\_OPTION*) {  
 return;  
 }  
  
 try (Connection conn = DriverManager.*getConnection*("jdbc:mysql://localhost:3306/cms", "root", "My134340Database")) {  
 String query;  
 if (userType.equalsIgnoreCase("customer")) {  
 query = "INSERT INTO customer\_table (cust\_name, cust\_username, cust\_password, contact, address) VALUES (?, ?, ?, ?, ?)";  
 } else {  
 query = "INSERT INTO employee\_table (emp\_name, emp\_username, emp\_password, contact, address) VALUES (?, ?, ?, ?, ?)";  
 }  
  
 try (PreparedStatement stmt = conn.prepareStatement(query)) {  
 stmt.setString(1, name);  
 stmt.setString(2, username);  
 stmt.setString(3, password);  
 stmt.setString(4, contact);  
 stmt.setString(5, address);  
  
 int rowsInserted = stmt.executeUpdate();  
 if (rowsInserted > 0) {  
 JOptionPane.*showMessageDialog*(this, "Registration successful! Please login.");  
 openLoginPage(); // Open LoginForm after successful registration  
 } else {  
 JOptionPane.*showMessageDialog*(this, "Error during registration.");  
 }  
 }  
 } catch (Exception ex) {  
 ex.printStackTrace();  
 JOptionPane.*showMessageDialog*(this, "Database error: " + ex.getMessage());  
 }  
 }  
  
 public static void main(String[] args) {  
 // Create an instance of LoginForm to pass to RegistrationForm  
 LoginForm loginForm = new LoginForm();  
 new RegistrationForm(loginForm);  
 }  
}

CMS

import cms.Admin.AdminUI;  
import cms.Employee.EmployeeUI;  
import cms.Customer.CustomerUI;  
  
import javax.swing.\*;  
import java.awt.\*;  
import java.io.File;  
import java.io.IOException;  
import javax.imageio.ImageIO;  
import java.awt.image.BufferedImage;  
import java.sql.\*;  
  
public class CMS extends JFrame {  
  
 public CMS() {  
 setTitle("Courier Management System");  
 setSize(800, 600);  
 setDefaultCloseOperation(JFrame.*EXIT\_ON\_CLOSE*);  
 setLocationRelativeTo(null);  
 setLayout(new BorderLayout());  
  
 // Load background image  
 BufferedImage backgroundImage = null;  
 try {  
 backgroundImage = ImageIO.*read*(new File("D:/bg1.jpg")); // Ensure correct path  
 } catch (IOException e) {  
 e.printStackTrace();  
 JOptionPane.*showMessageDialog*(this, "Background image not found!");  
 }  
  
 // Create a JLabel with the background image  
 JLabel backgroundLabel = new JLabel(new ImageIcon(backgroundImage)) {  
 @Override  
 protected void paintComponent(Graphics g) {  
 super.paintComponent(g);  
 g.drawImage(((ImageIcon) getIcon()).getImage(), 0, 0, getWidth(), getHeight(), this);  
 }  
 };  
 backgroundLabel.setLayout(new GridBagLayout()); // Use GridBagLayout for centering components  
  
 // User Selection Panel  
 JPanel userPanel = new JPanel(new FlowLayout(FlowLayout.*CENTER*));  
 JButton adminButton = new JButton("Admin");  
 JButton employeeButton = new JButton("Employee");  
 JButton customerButton = new JButton("Customer");  
  
 userPanel.add(adminButton);  
 userPanel.add(employeeButton);  
 userPanel.add(customerButton);  
  
 // Add user selection panel to the background label  
 GridBagConstraints gbc = new GridBagConstraints();  
 gbc.gridx = 0;  
 gbc.gridy = 0;  
 gbc.weightx = 1.0;  
 gbc.weighty = 1.0;  
 gbc.anchor = GridBagConstraints.*CENTER*;  
 backgroundLabel.add(userPanel, gbc);  
  
 // Action Listeners  
 adminButton.addActionListener(e -> launchAdmin());  
 employeeButton.addActionListener(e -> launchEmployee());  
 customerButton.addActionListener(e -> launchCustomer());  
  
 // Add background label to the frame  
 add(backgroundLabel);  
  
 setVisible(true);  
 }  
  
 private void launchAdmin() {  
 String adminName = JOptionPane.*showInputDialog*(this, "Enter Admin Name:");  
 if (adminName != null && !adminName.trim().isEmpty()) {  
 SwingUtilities.*invokeLater*(() -> new AdminUI(adminName).setVisible(true));  
 dispose();  
 }  
 }  
  
 private void launchEmployee() {  
 String employeeIdStr = JOptionPane.*showInputDialog*(this, "Enter Employee ID:");  
 if (employeeIdStr != null && !employeeIdStr.trim().isEmpty()) {  
 try {  
 int employeeId = Integer.*parseInt*(employeeIdStr);  
 SwingUtilities.*invokeLater*(() -> new EmployeeUI(employeeId).setVisible(true));  
 dispose();  
 } catch (NumberFormatException ex) {  
 JOptionPane.*showMessageDialog*(this, "Invalid Employee ID!");  
 }  
 }  
 }  
  
 private void launchCustomer() {  
 String customerUsername = JOptionPane.*showInputDialog*(this, "Enter Customer Username:");  
 if (customerUsername != null && !customerUsername.trim().isEmpty()) {  
 SwingUtilities.*invokeLater*(() -> new CustomerUI(customerUsername).setVisible(true));  
 dispose();  
 }  
 }  
  
 public static void main(String[] args) {  
 SwingUtilities.*invokeLater*(() -> new CMS());  
 }  
}