

**TRƯỜNG ĐẠI HỌC**

**VIỆN CÔNG NGHỆ THÔNG TIN**

**BÁO CÁO BÀI TẬP LỚN**

**CÔNG NGHỆ PHẦN MỀM**

**ĐỀ TÀI:**

**QUẢN LÝ BÁN HÀNG CỦA SIÊU THỊ**

|  |  |
| --- | --- |
| Giảng viên hướng dẫn: |  |
| Sinh viên thực hiện: |  |
| Thành viên nhóm: |  |
| Lớp: |  |

Hà Nội, năm 2024

**LẬP TRÌNH QUẢN LÝ HÀNG**

|  |  |
| --- | --- |
| Tên chương trình: QUẢN LÝ BÁN HÀNG CỦA SIÊU THỊ | Mã chương trình:  QLBHCST |
| Tên module: Lập trình quản lý hàng | Mã tài liệu:  LT3 |
| Tên người làm: | Mã người làm: N4 |
| Thời hạn: Ngày bắt đầu: 10/04/2024 | Ngày kết thúc:  14/05/2024 |
| Ngày nộp: 29/05/2024 | |
| Phiên bản/ File: LT3-240529.docx | |

**NGƯỜI LÀM KÝ XÁC NHẬN**

**MỤC LỤC**

I. GIAO DIỆN 3

1. FormDangNhap 3

2. FormQuanLyHang 17

II.THỰC THỂ 62

1. Product 62

2. Supplier 62

III. CONTOLLER 64

1. ProductController 64

2. SupplierController 81

IV. CONNECT 88

MyConnect 88

V. KẾT QUẢ 89

1. Đăng nhập 89

2. Quản lý nhà cung cấp 89

3. Quản lý sản phẩm 90

# I. GIAO DIỆN

## 1. FormDangNhap

1. /\*  
    \* Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license  
    \* Click nbfs://nbhost/SystemFileSystem/Templates/GUIForms/JFrame.java to edit this template  
    \*/  
   package org.example.views;  
     
   import org.example.controllers.UserController;  
   import org.example.models.Role;  
     
   import javax.swing.\*;  
   import java.awt.\*;  
   import java.awt.event.ActionEvent;  
   import java.awt.event.KeyAdapter;  
   import java.awt.event.KeyEvent;  
   import java.awt.event.KeyListener;  
   import java.util.List;  
     
   */\*\*  
    \*  
    \* @author   
    \*/*public class FormDangNhap extends javax.swing.JFrame {  
    private UserController userController; // Thêm biến để truy cập vào controller  
    private final int loggedInUserID; // Declare as final  
     
    public int getLoggedInUserID() {  
    return loggedInUserID;  
    }  
     
    */\*\*  
    \* Creates new form FormDangNhap  
    \*/* public FormDangNhap() {  
    initComponents();  
    userController = new UserController(); // Khởi tạo UserController  
    loggedInUserID = 0; // Initialize the final variable  
    loadRoles(); // Load danh sách quyền khi form được tạo  
    addKeyListeners();  
    addEscapeKeyBinding();  
    }  
     
    private void addEscapeKeyBinding() {  
    String escapeAction = "escapeAction";  
    KeyStroke escapeKeyStroke = KeyStroke.*getKeyStroke*(KeyEvent.*VK\_ESCAPE*, 0, false);  
     
    Action escapeActionHandler = new AbstractAction() {  
    @Override  
    public void actionPerformed(ActionEvent e) {  
    Object[] options = {"Thoát", "Không"};  
    int response = JOptionPane.*showOptionDialog*(  
    FormDangNhap.this,  
    "Bạn có muốn thoát chương trình?",  
    "Xác nhận",  
    JOptionPane.*YES\_NO\_OPTION*,  
    JOptionPane.*QUESTION\_MESSAGE*,  
    null,  
    options,  
    options[0]  
    );  
    if (response == JOptionPane.*NO\_OPTION*) {  
    System.*out*.println("không có gì");  
    } else if (response == JOptionPane.*YES\_OPTION*) {  
    // Thoát chương trình  
    System.*out*.println("đã thoát chương trình đăng nhập");  
    System.*exit*(0);  
    }  
    }  
    };  
     
    JRootPane rootPane = this.getRootPane();  
    rootPane.getInputMap(JComponent.*WHEN\_IN\_FOCUSED\_WINDOW*).put(escapeKeyStroke, escapeAction);  
    rootPane.getActionMap().put(escapeAction, escapeActionHandler);  
    }  
     
     
    private void addKeyListeners() {  
    KeyListener keyListener = new KeyAdapter() {  
    @Override  
    public void keyPressed(KeyEvent e) {  
    int keyCode = e.getKeyCode();  
    if (keyCode == KeyEvent.*VK\_ENTER*) {  
    btnDangNhap.doClick(); // Giả lập sự kiện nhấn nút Đăng Nhập  
    }  
   // else if (keyCode == KeyEvent.VK\_ESCAPE) {  
   // dispose(); // Đóng form đăng nhập  
   // }  
    }  
    };  
     
    tbUsername.addKeyListener(keyListener);  
    tbPassword.addKeyListener(keyListener);  
    cbRole.addKeyListener(keyListener);  
    btnDangNhap.addKeyListener(keyListener);  
    }  
    private void loadRoles() {  
    List<Role> roles = userController.getAllRoles();  
    for (Role role : roles) {  
    cbRole.addItem(role.getRoleName());  
    }  
    }  
    */\*\*  
    \* This method is called from within the constructor to initialize the form.  
    \* WARNING: Do NOT modify this code. The content of this method is always  
    \* regenerated by the Form Editor.  
    \*/* @SuppressWarnings("unchecked")  
    // <editor-fold defaultstate="collapsed" desc="Generated Code">//GEN-BEGIN:initComponents  
    private void initComponents() {  
     
    jPanel1 = new javax.swing.JPanel();  
    btnDangNhap = new javax.swing.JButton();  
    jLabel4 = new javax.swing.JLabel();  
    jLabel1 = new javax.swing.JLabel();  
    jLabel2 = new javax.swing.JLabel();  
    jLabel3 = new javax.swing.JLabel();  
    cbRole = new javax.swing.JComboBox<>();  
    tbUsername = new javax.swing.JTextField();  
    tbPassword = new javax.swing.JTextField();  
     
    setDefaultCloseOperation(javax.swing.WindowConstants.*EXIT\_ON\_CLOSE*);  
    setBackground(new java.awt.Color(102, 153, 255));  
     
    jPanel1.setBackground(new java.awt.Color(204, 204, 255));  
     
    btnDangNhap.setBackground(new java.awt.Color(102, 102, 255));  
    btnDangNhap.setFont(new java.awt.Font("Segoe UI", 1, 24)); // NOI18N  
    btnDangNhap.setForeground(new java.awt.Color(255, 255, 255));  
    btnDangNhap.setText("Đăng nhập");  
    btnDangNhap.addActionListener(new java.awt.event.ActionListener() {  
    public void actionPerformed(java.awt.event.ActionEvent evt) {  
    btnDangNhapActionPerformed(evt);  
    }  
    });  
     
    jLabel4.setFont(new java.awt.Font("Segoe UI", 1, 36)); // NOI18N  
    jLabel4.setHorizontalAlignment(javax.swing.SwingConstants.*CENTER*);  
    jLabel4.setText("Đăng Nhập");  
     
    jLabel1.setFont(new java.awt.Font("Arial", 1, 24)); // NOI18N  
    jLabel1.setText("Tên dăng nhập");  
     
    jLabel2.setFont(new java.awt.Font("Arial", 1, 24)); // NOI18N  
    jLabel2.setText("Mật khẩu");  
     
    jLabel3.setFont(new java.awt.Font("Arial", 1, 24)); // NOI18N  
    jLabel3.setText("Quyền");  
     
    cbRole.addActionListener(new java.awt.event.ActionListener() {  
    public void actionPerformed(java.awt.event.ActionEvent evt) {  
    cbRoleActionPerformed(evt);  
    }  
    });  
     
    tbUsername.addActionListener(new java.awt.event.ActionListener() {  
    public void actionPerformed(java.awt.event.ActionEvent evt) {  
    tbUsernameActionPerformed(evt);  
    }  
    });  
     
    tbPassword.addActionListener(new java.awt.event.ActionListener() {  
    public void actionPerformed(java.awt.event.ActionEvent evt) {  
    tbPasswordActionPerformed(evt);  
    }  
    });  
     
    javax.swing.GroupLayout jPanel1Layout = new javax.swing.GroupLayout(jPanel1);  
    jPanel1.setLayout(jPanel1Layout);  
    jPanel1Layout.setHorizontalGroup(  
    jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.*LEADING*)  
    .addGroup(jPanel1Layout.createSequentialGroup()  
    .addContainerGap()  
    .addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.*LEADING*)  
    .addGroup(jPanel1Layout.createSequentialGroup()  
    .addComponent(jLabel1)  
    .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.*UNRELATED*)  
    .addComponent(tbUsername))  
    .addComponent(jLabel4, javax.swing.GroupLayout.Alignment.*TRAILING*, javax.swing.GroupLayout.*DEFAULT\_SIZE*, javax.swing.GroupLayout.*DEFAULT\_SIZE*, Short.*MAX\_VALUE*)  
    .addGroup(jPanel1Layout.createSequentialGroup()  
    .addGap(62, 62, 62)  
    .addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.*LEADING*)  
    .addGroup(jPanel1Layout.createSequentialGroup()  
    .addGap(31, 31, 31)  
    .addComponent(jLabel3)  
    .addGap(18, 18, 18)  
    .addComponent(cbRole, 0, 454, Short.*MAX\_VALUE*))  
    .addGroup(javax.swing.GroupLayout.Alignment.*TRAILING*, jPanel1Layout.createSequentialGroup()  
    .addComponent(jLabel2)  
    .addGap(18, 18, 18)  
    .addComponent(tbPassword))))  
    .addGroup(javax.swing.GroupLayout.Alignment.*TRAILING*, jPanel1Layout.createSequentialGroup()  
    .addGap(0, 0, Short.*MAX\_VALUE*)  
    .addComponent(btnDangNhap, javax.swing.GroupLayout.*PREFERRED\_SIZE*, 177, javax.swing.GroupLayout.*PREFERRED\_SIZE*)))  
    .addContainerGap())  
    );  
    jPanel1Layout.setVerticalGroup(  
    jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.*LEADING*)  
    .addGroup(javax.swing.GroupLayout.Alignment.*TRAILING*, jPanel1Layout.createSequentialGroup()  
    .addContainerGap()  
    .addComponent(jLabel4, javax.swing.GroupLayout.*PREFERRED\_SIZE*, 74, javax.swing.GroupLayout.*PREFERRED\_SIZE*)  
    .addGap(66, 66, 66)  
    .addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.*LEADING*)  
    .addComponent(jLabel1)  
    .addGroup(jPanel1Layout.createSequentialGroup()  
    .addGap(3, 3, 3)  
    .addComponent(tbUsername, javax.swing.GroupLayout.*PREFERRED\_SIZE*, 34, javax.swing.GroupLayout.*PREFERRED\_SIZE*)))  
    .addGap(12, 12, 12)  
    .addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.*BASELINE*)  
    .addComponent(jLabel2)  
    .addComponent(tbPassword, javax.swing.GroupLayout.*PREFERRED\_SIZE*, 34, javax.swing.GroupLayout.*PREFERRED\_SIZE*))  
    .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.*UNRELATED*)  
    .addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.*BASELINE*)  
    .addComponent(cbRole, javax.swing.GroupLayout.*PREFERRED\_SIZE*, 34, javax.swing.GroupLayout.*PREFERRED\_SIZE*)  
    .addComponent(jLabel3))  
    .addGap(86, 86, 86)  
    .addComponent(btnDangNhap, javax.swing.GroupLayout.*PREFERRED\_SIZE*, 48, javax.swing.GroupLayout.*PREFERRED\_SIZE*)  
    .addContainerGap())  
    );  
     
    javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());  
    getContentPane().setLayout(layout);  
    layout.setHorizontalGroup(  
    layout.createParallelGroup(javax.swing.GroupLayout.Alignment.*LEADING*)  
    .addGroup(layout.createSequentialGroup()  
    .addContainerGap()  
    .addComponent(jPanel1, javax.swing.GroupLayout.*PREFERRED\_SIZE*, javax.swing.GroupLayout.*DEFAULT\_SIZE*, javax.swing.GroupLayout.*PREFERRED\_SIZE*)  
    .addContainerGap(javax.swing.GroupLayout.*DEFAULT\_SIZE*, Short.*MAX\_VALUE*))  
    );  
    layout.setVerticalGroup(  
    layout.createParallelGroup(javax.swing.GroupLayout.Alignment.*LEADING*)  
    .addGroup(layout.createSequentialGroup()  
    .addContainerGap()  
    .addComponent(jPanel1, javax.swing.GroupLayout.*DEFAULT\_SIZE*, javax.swing.GroupLayout.*DEFAULT\_SIZE*, Short.*MAX\_VALUE*)  
    .addContainerGap())  
    );  
     
    pack();  
    }// </editor-fold>//GEN-END:initComponents  
     
    private void tbUsernameActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event\_tbUsernameActionPerformed  
    // *TODO add your handling code here:* }//GEN-LAST:event\_tbUsernameActionPerformed  
     
    private void tbPasswordActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event\_tbPasswordActionPerformed  
    // *TODO add your handling code here:* }//GEN-LAST:event\_tbPasswordActionPerformed  
     
    private void cbRoleActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event\_cbRoleActionPerformed  
    // *TODO add your handling code here:* }//GEN-LAST:event\_cbRoleActionPerformed  
     
    private void btnDangNhapActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event\_btnDangNhapActionPerformed  
    // *TODO add your handling code here:* String username = tbUsername.getText();  
    String password = tbPassword.getText();  
    String selectedRole = cbRole.getSelectedItem().toString();  
     
    // Attempt login  
    boolean loggedIn = userController.loginWithRole(username, password, selectedRole);  
     
    if (loggedIn) {  
    // Retrieve the user's ID after successful login  
    int userID = userController.getUserID(username);  
     
    // Save the user ID in the session  
    userController.setUserIDInSession(userID);  
     
    // Update the loggedInUserID  
     
    JOptionPane.*showMessageDialog*(this, "Đăng nhập thành công với quyền: " + selectedRole);  
    System.*out*.println("Đăng nhập thành công với các thông tin");  
    // Redirect to the appropriate page based on the user's role  
    switch(selectedRole) {  
    case "bộ phận nhân sự":  
    FormNhanSu formNhanSu = new FormNhanSu();  
    formNhanSu.setVisible(true);  
    System.*out*.println("Chức Vụ: bộ phận nhân sự");  
    break;  
    case "bộ phận giao dịch":  
    FormGiaoDich formCungUng = new FormGiaoDich(userID);  
    formCungUng.setVisible(true);  
    System.*out*.println("Chức Vụ: bộ phận giao dịch");  
    break;  
    case "quản lý hàng":  
    FormQuanLyHang formQuanLyHang = new FormQuanLyHang(userID);  
    formQuanLyHang.setVisible(true);  
    System.*out*.println("Chức Vụ: quản lý hàng");  
    break;  
    case "kế toán":  
    FormKeToan formKeToan = new FormKeToan(userID);  
    formKeToan.setVisible(true);  
    System.*out*.println("Chức Vụ: kế toán");  
    break;  
    case "Admin":  
    FormAdmin formAdmin = new FormAdmin();  
    formAdmin.setVisible(true);  
    break;  
    default:  
    JOptionPane.*showMessageDialog*(this, "Không tìm thấy trang tương ứng cho quyền này");  
    }  
    System.*out*.println("UserName: " + username);  
    System.*out*.println("UserID: " + userID);  
     
    this.dispose(); // Close the login form  
    } else {  
    JOptionPane.*showMessageDialog*(this, "Đăng nhập thất bại");  
    }  
    }//GEN-LAST:event\_btnDangNhapActionPerformed  
     
    */\*\*  
    \* @param args the command line arguments  
    \*/* public static void main(String args[]) {  
    /\* Set the Nimbus look and feel \*/  
    //<editor-fold defaultstate="collapsed" desc=" Look and feel setting code (optional) ">  
    /\* If Nimbus (introduced in Java SE 6) is not available, stay with the default look and feel.  
    \* For details see http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html   
    \*/  
    try {  
    for (javax.swing.UIManager.LookAndFeelInfo info : javax.swing.UIManager.*getInstalledLookAndFeels*()) {  
    if ("Nimbus".equals(info.getName())) {  
    javax.swing.UIManager.*setLookAndFeel*(info.getClassName());  
    break;  
    }  
    }  
    } catch (ClassNotFoundException ex) {  
    java.util.logging.Logger.*getLogger*(FormDangNhap.class.getName()).log(java.util.logging.Level.*SEVERE*, null, ex);  
    } catch (InstantiationException ex) {  
    java.util.logging.Logger.*getLogger*(FormDangNhap.class.getName()).log(java.util.logging.Level.*SEVERE*, null, ex);  
    } catch (IllegalAccessException ex) {  
    java.util.logging.Logger.*getLogger*(FormDangNhap.class.getName()).log(java.util.logging.Level.*SEVERE*, null, ex);  
    } catch (javax.swing.UnsupportedLookAndFeelException ex) {  
    java.util.logging.Logger.*getLogger*(FormDangNhap.class.getName()).log(java.util.logging.Level.*SEVERE*, null, ex);  
    }  
    //</editor-fold>  
     
    /\* Create and display the form \*/  
    java.awt.EventQueue.*invokeLater*(new Runnable() {  
    public void run() {  
    new FormDangNhap().setVisible(true);  
    }  
    });  
    }  
     
    // Variables declaration - do not modify//GEN-BEGIN:variables  
    private javax.swing.JButton btnDangNhap;  
    private javax.swing.JComboBox<String> cbRole;  
    private javax.swing.JLabel jLabel1;  
    private javax.swing.JLabel jLabel2;  
    private javax.swing.JLabel jLabel3;  
    private javax.swing.JLabel jLabel4;  
    private javax.swing.JPanel jPanel1;  
    private javax.swing.JTextField tbPassword;  
    private javax.swing.JTextField tbUsername;  
    // End of variables declaration//GEN-END:variables  
   }

## 2. FormQuanLyHang

/\*  
 \* Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license  
 \* Click nbfs://nbhost/SystemFileSystem/Templates/GUIForms/JFrame.java to edit this template  
 \*/  
package org.example.views;  
  
import org.example.controllers.\*;  
import org.example.models.\*;  
import org.example.util.ImageRenderer;  
  
import javax.imageio.ImageIO;  
import javax.imageio.ImageReader;  
import javax.swing.\*;  
import javax.swing.table.DefaultTableCellRenderer;  
import javax.swing.table.DefaultTableModel;  
import javax.swing.table.TableColumn;  
import javax.swing.table.TableColumnModel;  
import java.awt.\*;  
import java.awt.event.ActionEvent;  
import java.awt.event.KeyAdapter;  
import java.awt.event.KeyEvent;  
import java.awt.event.KeyListener;  
import java.awt.image.BufferedImage;  
import java.io.File;  
import java.io.IOException;  
import java.net.URL;  
import java.nio.file.Files;  
import java.nio.file.StandardCopyOption;  
import java.util.List;  
import java.util.Random;  
  
*/\*\*  
 \* @author   
 \*/*// Định nghĩa class ImageRenderer  
  
  
  
public class FormQuanLyHang extends javax.swing.JFrame {  
 private DefaultTableModel tableModel;  
 private InventoryReceiptController inventoryReceiptController;  
 private SalesReceiptController salesReceiptController;  
 private ProductController productController;  
 private SupplierController supplierController;  
 private int loggedInUserID;  
  
 */\*\*  
 \* Creates new form FormQuanLyHang  
 \*/* public FormQuanLyHang(int loggedInUserID) {  
 initComponents();  
 this.loggedInUserID = loggedInUserID;  
  
 // Initialize controllers  
 inventoryReceiptController = new InventoryReceiptController();  
 salesReceiptController = new SalesReceiptController();  
 productController = new ProductController();  
 supplierController = new SupplierController();  
 // Load data  
  
 showSanPham();  
 showNhaCungCap();  
 loadSupplier();  
 addEscapeKeyBinding();  
 System.out.println("Đang Quản Lý Hàng Với UserID: " + loggedInUserID); // In giá trị loggedInUserID  
 }  
  
 private void addEscapeKeyBinding() {  
 String escapeAction = "escapeAction";  
 KeyStroke escapeKeyStroke = KeyStroke.getKeyStroke(KeyEvent.VK\_ESCAPE, 0, false);  
  
 Action escapeActionHandler = new AbstractAction() {  
 @Override  
 public void actionPerformed(ActionEvent e) {  
 Object[] options = {"Đăng xuất", "Thoát", "Không"};  
 int response = JOptionPane.showOptionDialog(  
 FormQuanLyHang.this,  
 "Bạn có muốn đăng xuất hay thoát chương trình?",  
 "Xác nhận",  
 JOptionPane.YES\_NO\_CANCEL\_OPTION,  
 JOptionPane.QUESTION\_MESSAGE,  
 null,  
 options,  
 options[0]  
 );  
  
 if (response == JOptionPane.YES\_OPTION) {  
 System.out.println("Bộ phận quản lý hàng đã đăng xuất");  
 // Xử lý đăng xuất  
 btnDangXuatActionPerformed(null);  
 } else if (response == JOptionPane.NO\_OPTION) {  
 // Thoát chương trình  
 System.out.println("Đã thoát chương trình");  
 System.exit(0);  
 }else if (response == JOptionPane.CANCEL\_OPTION) {  
 System.out.println("Không có gì");  
 }  
 }  
 };  
  
 JRootPane rootPane = this.getRootPane();  
 rootPane.getInputMap(JComponent.WHEN\_IN\_FOCUSED\_WINDOW).put(escapeKeyStroke, escapeAction);  
 rootPane.getActionMap().put(escapeAction, escapeActionHandler);  
 }  
  
 private void refreshTable() {  
 DefaultTableModel model = (DefaultTableModel) TableSanPham.getModel();  
 model.fireTableDataChanged();  
 updateImageColumn();  
 }  
  
  
 // Cập nhật hiển thị của ảnh trong bảng khi có thay đổi dữ liệu  
 public void updateImageColumn() {  
 TableColumnModel columnModel = TableSanPham.getColumnModel();  
 TableColumn imageColumn = columnModel.getColumn(11); // Assuming column index of the image column is 11  
 ImageRenderer imageRenderer = new ImageRenderer();  
 imageColumn.setCellRenderer( imageRenderer);  
 }  
  
  
 private void loadSupplier() {  
 SupplierController supplierController = new SupplierController(); // Create an instance of SupplierController  
 DefaultTableModel model = supplierController.getAllSuppliers(); // Call getAllSuppliers method  
  
 for (int i = 0; i < model.getRowCount(); i++) {  
 String supplierName = (String) model.getValueAt(i, 1); // Assuming SupplierName is at column index 1  
 ComboBoxNhaCungCap.addItem(supplierName); // Add supplier names to your combo box  
 }  
  
 // Select a specific supplier, for example, the first supplier in the list  
 if (model.getRowCount() > 0) {  
 selectSupplier((String) model.getValueAt(0, 1)); // Assuming SupplierName is at column index 1  
 }  
 }  
  
  
 private void selectSupplier(String supplierName) {  
 for (int i = 0; i < ComboBoxNhaCungCap.getItemCount(); i++) {  
 String item = (String) ComboBoxNhaCungCap.getItemAt(i);  
 if (item.equals(supplierName)) {  
 ComboBoxNhaCungCap.setSelectedItem(item);  
 break;  
 }  
 }  
 }  
  
  
 private void showNhaCungCap() {  
 supplierController = new SupplierController();  
 tableModel = supplierController.getAllSuppliers();  
 TableNhaCungCap.setModel(tableModel);  
  
 }  
  
 private void showSanPham() {  
 productController = new ProductController();  
 tableModel = productController.getAllProductDetails();  
 TableSanPham.setModel(tableModel);  
 ImageRenderer imageRenderer = new ImageRenderer();  
 TableSanPham.getColumnModel().getColumn(11).setCellRenderer(imageRenderer);  
  
 }  
  
   
  
  
   
  
  
 */\*\*  
 \* This method is called from within the constructor to initialize the form.  
 \* WARNING: Do NOT modify this code. The content of this method is always  
 \* regenerated by the Form Editor.  
 \*/* @SuppressWarnings("unchecked")  
 // <editor-fold defaultstate="collapsed" desc="Generated Code">//GEN-BEGIN:initComponents  
 private void initComponents() {  
  
 jPanel1 = new javax.swing.JPanel();  
 jLabel1 = new javax.swing.JLabel();  
 jTabbedPane1 = new javax.swing.JTabbedPane();  
 jPanel5 = new javax.swing.JPanel();  
 jScrollPane4 = new javax.swing.JScrollPane();  
 TableSanPham = new javax.swing.JTable();  
 btnThemSanPham = new javax.swing.JButton();  
 btnSuaSanPham = new javax.swing.JButton();  
 btnXoaSanPham = new javax.swing.JButton();  
 btnLamTrong = new javax.swing.JButton();  
 btnChonAnhSanPham = new javax.swing.JButton();  
 LabelAnhSanPham = new javax.swing.JLabel();  
 jPanel7 = new javax.swing.JPanel();  
 ComboBoxNhaCungCap = new javax.swing.JComboBox<>();  
 jLabel13 = new javax.swing.JLabel();  
 tbDonGia = new javax.swing.JTextField();  
 jLabel12 = new javax.swing.JLabel();  
 jLabel11 = new javax.swing.JLabel();  
 tbSoLuong = new javax.swing.JTextField();  
 jPanel8 = new javax.swing.JPanel();  
 tbTenSanPham = new javax.swing.JTextField();  
 jLabel9 = new javax.swing.JLabel();  
 tbDanhMuc = new javax.swing.JTextField();  
 jLabel10 = new javax.swing.JLabel();  
 jLabel8 = new javax.swing.JLabel();  
 tbMaSanPham = new javax.swing.JTextField();  
 jPanel6 = new javax.swing.JPanel();  
 jScrollPane5 = new javax.swing.JScrollPane();  
 TableNhaCungCap = new javax.swing.JTable();  
 jLabel2 = new javax.swing.JLabel();  
 jLabel3 = new javax.swing.JLabel();  
 jLabel4 = new javax.swing.JLabel();  
 jLabel5 = new javax.swing.JLabel();  
 jLabel6 = new javax.swing.JLabel();  
 jLabel7 = new javax.swing.JLabel();  
 tbSupplierID = new javax.swing.JTextField();  
 tbSupplierName = new javax.swing.JTextField();  
 tbContactInfo = new javax.swing.JTextField();  
 tbEmail = new javax.swing.JTextField();  
 tbPhone = new javax.swing.JTextField();  
 tbAddress = new javax.swing.JTextField();  
 btnThemNhaCungCap = new javax.swing.JButton();  
 btnSuaNhaCungCap = new javax.swing.JButton();  
 btnXoaNhaCungCap = new javax.swing.JButton();  
 btnLamRongNhaCungCap = new javax.swing.JButton();  
 btnDangXuat = new javax.swing.JButton();  
  
 setDefaultCloseOperation(javax.swing.WindowConstants.EXIT\_ON\_CLOSE);  
  
 jLabel1.setFont(new java.awt.Font("Segoe UI", 1, 36)); // NOI18N  
 jLabel1.setHorizontalAlignment(javax.swing.SwingConstants.LEFT);  
 jLabel1.setText("Quản Lý Hàng");  
  
 TableSanPham.setFont(new java.awt.Font("Arial", 1, 14)); // NOI18N  
 TableSanPham.setModel(new javax.swing.table.DefaultTableModel(  
 new Object [][] {  
 {},  
 {},  
 {},  
 {}  
 },  
 new String [] {  
  
 }  
 ));  
 TableSanPham.setAlignmentX(2.0F);  
 TableSanPham.setAlignmentY(2.0F);  
 TableSanPham.setCursor(new java.awt.Cursor(java.awt.Cursor.DEFAULT\_CURSOR));  
 TableSanPham.setDoubleBuffered(true);  
 TableSanPham.setRowHeight(50);  
 TableSanPham.setSelectionBackground(new java.awt.Color(232, 57, 95));  
 TableSanPham.setShowGrid(false);  
 TableSanPham.setShowHorizontalLines(true);  
 TableSanPham.addMouseListener(new java.awt.event.MouseAdapter() {  
 public void mouseClicked(java.awt.event.MouseEvent evt) {  
 TableSanPhamMouseClicked(evt);  
 }  
 });  
 jScrollPane4.setViewportView(TableSanPham);  
  
 btnThemSanPham.setBackground(new java.awt.Color(108, 117, 125));  
 btnThemSanPham.setFont(new java.awt.Font("Arial", 0, 18)); // NOI18N  
 btnThemSanPham.setForeground(new java.awt.Color(255, 255, 255));  
 btnThemSanPham.setText("Thêm");  
 btnThemSanPham.addActionListener(new java.awt.event.ActionListener() {  
 public void actionPerformed(java.awt.event.ActionEvent evt) {  
 btnThemSanPhamActionPerformed(evt);  
 }  
 });  
  
 btnSuaSanPham.setBackground(new java.awt.Color(0, 123, 255));  
 btnSuaSanPham.setFont(new java.awt.Font("Arial", 0, 18)); // NOI18N  
 btnSuaSanPham.setForeground(new java.awt.Color(255, 255, 255));  
 btnSuaSanPham.setText("Sửa");  
 btnSuaSanPham.addActionListener(new java.awt.event.ActionListener() {  
 public void actionPerformed(java.awt.event.ActionEvent evt) {  
 btnSuaSanPhamActionPerformed(evt);  
 }  
 });  
  
 btnXoaSanPham.setBackground(new java.awt.Color(220, 53, 69));  
 btnXoaSanPham.setFont(new java.awt.Font("Arial", 0, 18)); // NOI18N  
 btnXoaSanPham.setForeground(new java.awt.Color(255, 255, 255));  
 btnXoaSanPham.setText("Xóa");  
 btnXoaSanPham.addActionListener(new java.awt.event.ActionListener() {  
 public void actionPerformed(java.awt.event.ActionEvent evt) {  
 btnXoaSanPhamActionPerformed(evt);  
 }  
 });  
  
 btnLamTrong.setBackground(new java.awt.Color(0, 204, 204));  
 btnLamTrong.setFont(new java.awt.Font("Arial", 0, 18)); // NOI18N  
 btnLamTrong.setForeground(new java.awt.Color(255, 255, 255));  
 btnLamTrong.setText("Làm Trỗng");  
 btnLamTrong.addActionListener(new java.awt.event.ActionListener() {  
 public void actionPerformed(java.awt.event.ActionEvent evt) {  
 btnLamTrongActionPerformed(evt);  
 }  
 });  
  
 btnChonAnhSanPham.setBackground(new java.awt.Color(102, 153, 255));  
 btnChonAnhSanPham.setText("Chọn ảnh");  
 btnChonAnhSanPham.addActionListener(new java.awt.event.ActionListener() {  
 public void actionPerformed(java.awt.event.ActionEvent evt) {  
 btnChonAnhSanPhamActionPerformed(evt);  
 }  
 });  
  
 LabelAnhSanPham.setText("Ảnh Sản Phẩm");  
  
 jLabel13.setText("Nhà cung cấp");  
  
 tbDonGia.addActionListener(new java.awt.event.ActionListener() {  
 public void actionPerformed(java.awt.event.ActionEvent evt) {  
 tbDonGiaActionPerformed(evt);  
 }  
 });  
  
 jLabel12.setText("Đơn giá");  
  
 jLabel11.setText("Số lượng");  
  
 javax.swing.GroupLayout jPanel7Layout = new javax.swing.GroupLayout(jPanel7);  
 jPanel7.setLayout(jPanel7Layout);  
 jPanel7Layout.setHorizontalGroup(  
 jPanel7Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)  
 .addGroup(jPanel7Layout.createSequentialGroup()  
 .addContainerGap()  
 .addGroup(jPanel7Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)  
 .addGroup(jPanel7Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)  
 .addGroup(jPanel7Layout.createSequentialGroup()  
 .addGap(39, 39, 39)  
 .addComponent(jLabel12))  
 .addComponent(jLabel11, javax.swing.GroupLayout.Alignment.TRAILING))  
 .addComponent(jLabel13))  
 .addGap(18, 18, 18)  
 .addGroup(jPanel7Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)  
 .addComponent(tbDonGia)  
 .addComponent(tbSoLuong)  
 .addComponent(ComboBoxNhaCungCap, 0, 132, Short.MAX\_VALUE))  
 .addContainerGap())  
 );  
 jPanel7Layout.setVerticalGroup(  
 jPanel7Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)  
 .addGroup(jPanel7Layout.createSequentialGroup()  
 .addContainerGap()  
 .addGroup(jPanel7Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)  
 .addComponent(tbSoLuong, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)  
 .addComponent(jLabel11))  
 .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)  
 .addGroup(jPanel7Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)  
 .addComponent(tbDonGia, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)  
 .addComponent(jLabel12))  
 .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, 23, Short.MAX\_VALUE)  
 .addGroup(jPanel7Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)  
 .addComponent(jLabel13)  
 .addComponent(ComboBoxNhaCungCap, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE))  
 .addGap(41, 41, 41))  
 );  
  
 tbTenSanPham.addActionListener(new java.awt.event.ActionListener() {  
 public void actionPerformed(java.awt.event.ActionEvent evt) {  
 tbTenSanPhamActionPerformed(evt);  
 }  
 });  
  
 jLabel9.setText("Tên sản phẩm");  
  
 jLabel10.setText("Danh mục");  
  
 jLabel8.setText("Mã sản phẩm");  
  
 javax.swing.GroupLayout jPanel8Layout = new javax.swing.GroupLayout(jPanel8);  
 jPanel8.setLayout(jPanel8Layout);  
 jPanel8Layout.setHorizontalGroup(  
 jPanel8Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)  
 .addGroup(jPanel8Layout.createSequentialGroup()  
 .addContainerGap()  
 .addGroup(jPanel8Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING, false)  
 .addGroup(jPanel8Layout.createSequentialGroup()  
 .addComponent(jLabel9)  
 .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)  
 .addComponent(tbTenSanPham, javax.swing.GroupLayout.PREFERRED\_SIZE, 205, javax.swing.GroupLayout.PREFERRED\_SIZE))  
 .addGroup(jPanel8Layout.createSequentialGroup()  
 .addComponent(jLabel10)  
 .addGap(18, 18, 18)  
 .addComponent(tbDanhMuc))  
 .addGroup(jPanel8Layout.createSequentialGroup()  
 .addComponent(jLabel8)  
 .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)  
 .addComponent(tbMaSanPham)))  
 .addContainerGap(12, Short.MAX\_VALUE))  
 );  
 jPanel8Layout.setVerticalGroup(  
 jPanel8Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)  
 .addGroup(javax.swing.GroupLayout.Alignment.TRAILING, jPanel8Layout.createSequentialGroup()  
 .addGap(20, 20, 20)  
 .addGroup(jPanel8Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)  
 .addComponent(jLabel8)  
 .addComponent(tbMaSanPham, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE))  
 .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, 26, Short.MAX\_VALUE)  
 .addGroup(jPanel8Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)  
 .addComponent(jLabel10)  
 .addComponent(tbDanhMuc, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE))  
 .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)  
 .addGroup(jPanel8Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)  
 .addComponent(jLabel9)  
 .addComponent(tbTenSanPham, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE))  
 .addContainerGap())  
 );  
  
 javax.swing.GroupLayout jPanel5Layout = new javax.swing.GroupLayout(jPanel5);  
 jPanel5.setLayout(jPanel5Layout);  
 jPanel5Layout.setHorizontalGroup(  
 jPanel5Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)  
 .addGroup(jPanel5Layout.createSequentialGroup()  
 .addContainerGap()  
 .addGroup(jPanel5Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)  
 .addComponent(jScrollPane4, javax.swing.GroupLayout.Alignment.TRAILING)  
 .addGroup(javax.swing.GroupLayout.Alignment.TRAILING, jPanel5Layout.createSequentialGroup()  
 .addGroup(jPanel5Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)  
 .addGroup(jPanel5Layout.createSequentialGroup()  
 .addComponent(jPanel8, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)  
 .addGap(18, 18, 18)  
 .addComponent(jPanel7, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE))  
 .addGroup(jPanel5Layout.createSequentialGroup()  
 .addGroup(jPanel5Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)  
 .addGroup(jPanel5Layout.createSequentialGroup()  
 .addComponent(btnThemSanPham)  
 .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)  
 .addComponent(btnSuaSanPham)  
 .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)  
 .addComponent(btnXoaSanPham)  
 .addGap(18, 18, 18)  
 .addComponent(btnLamTrong))  
 .addComponent(btnChonAnhSanPham, javax.swing.GroupLayout.PREFERRED\_SIZE, 106, javax.swing.GroupLayout.PREFERRED\_SIZE))  
 .addGap(0, 0, Short.MAX\_VALUE)))  
 .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)  
 .addComponent(LabelAnhSanPham, javax.swing.GroupLayout.PREFERRED\_SIZE, 250, javax.swing.GroupLayout.PREFERRED\_SIZE)))  
 .addContainerGap())  
 );  
 jPanel5Layout.setVerticalGroup(  
 jPanel5Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)  
 .addGroup(jPanel5Layout.createSequentialGroup()  
 .addContainerGap()  
 .addComponent(jScrollPane4, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)  
 .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)  
 .addGroup(jPanel5Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)  
 .addComponent(LabelAnhSanPham, javax.swing.GroupLayout.PREFERRED\_SIZE, 250, javax.swing.GroupLayout.PREFERRED\_SIZE)  
 .addGroup(jPanel5Layout.createSequentialGroup()  
 .addGroup(jPanel5Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)  
 .addComponent(jPanel8, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)  
 .addComponent(jPanel7, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE))  
 .addGap(18, 18, 18)  
 .addComponent(btnChonAnhSanPham, javax.swing.GroupLayout.PREFERRED\_SIZE, 46, javax.swing.GroupLayout.PREFERRED\_SIZE)  
 .addGap(18, 18, 18)  
 .addGroup(jPanel5Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)  
 .addComponent(btnThemSanPham)  
 .addComponent(btnSuaSanPham)  
 .addComponent(btnXoaSanPham)  
 .addComponent(btnLamTrong))))  
 .addContainerGap(javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE))  
 );  
  
 jTabbedPane1.addTab("Quản lý sản phẩm", jPanel5);  
  
 TableNhaCungCap.setFont(new java.awt.Font("Arial", 1, 14)); // NOI18N  
 TableNhaCungCap.setModel(new javax.swing.table.DefaultTableModel(  
 new Object [][] {  
 {},  
 {},  
 {},  
 {}  
 },  
 new String [] {  
  
 }  
 ));  
 TableNhaCungCap.setRowHeight(50);  
 TableNhaCungCap.setSelectionBackground(new java.awt.Color(232, 57, 95));  
 TableNhaCungCap.setShowHorizontalLines(true);  
 TableNhaCungCap.addMouseListener(new java.awt.event.MouseAdapter() {  
 public void mouseClicked(java.awt.event.MouseEvent evt) {  
 TableNhaCungCapMouseClicked(evt);  
 }  
 });  
 jScrollPane5.setViewportView(TableNhaCungCap);  
  
 jLabel2.setText("SupplierID");  
  
 jLabel3.setText("SupplierName");  
  
 jLabel4.setText("ContactInfo");  
  
 jLabel5.setText("Email");  
  
 jLabel6.setText("Phone");  
  
 jLabel7.setText("Address");  
  
 tbSupplierID.addActionListener(new java.awt.event.ActionListener() {  
 public void actionPerformed(java.awt.event.ActionEvent evt) {  
 tbSupplierIDActionPerformed(evt);  
 }  
 });  
  
 tbSupplierName.addActionListener(new java.awt.event.ActionListener() {  
 public void actionPerformed(java.awt.event.ActionEvent evt) {  
 tbSupplierNameActionPerformed(evt);  
 }  
 });  
  
 btnThemNhaCungCap.setBackground(new java.awt.Color(108, 117, 125));  
 btnThemNhaCungCap.setFont(new java.awt.Font("Arial", 0, 18)); // NOI18N  
 btnThemNhaCungCap.setForeground(new java.awt.Color(255, 255, 255));  
 btnThemNhaCungCap.setText("Thêm");  
 btnThemNhaCungCap.addActionListener(new java.awt.event.ActionListener() {  
 public void actionPerformed(java.awt.event.ActionEvent evt) {  
 btnThemNhaCungCapActionPerformed(evt);  
 }  
 });  
  
 btnSuaNhaCungCap.setBackground(new java.awt.Color(0, 123, 255));  
 btnSuaNhaCungCap.setFont(new java.awt.Font("Arial", 0, 18)); // NOI18N  
 btnSuaNhaCungCap.setForeground(new java.awt.Color(255, 255, 255));  
 btnSuaNhaCungCap.setText("Sửa");  
 btnSuaNhaCungCap.addActionListener(new java.awt.event.ActionListener() {  
 public void actionPerformed(java.awt.event.ActionEvent evt) {  
 btnSuaNhaCungCapActionPerformed(evt);  
 }  
 });  
  
 btnXoaNhaCungCap.setBackground(new java.awt.Color(220, 53, 69));  
 btnXoaNhaCungCap.setFont(new java.awt.Font("Arial", 0, 18)); // NOI18N  
 btnXoaNhaCungCap.setForeground(new java.awt.Color(255, 255, 255));  
 btnXoaNhaCungCap.setText("Xóa");  
 btnXoaNhaCungCap.addActionListener(new java.awt.event.ActionListener() {  
 public void actionPerformed(java.awt.event.ActionEvent evt) {  
 btnXoaNhaCungCapActionPerformed(evt);  
 }  
 });  
  
 btnLamRongNhaCungCap.setBackground(new java.awt.Color(0, 204, 204));  
 btnLamRongNhaCungCap.setFont(new java.awt.Font("Arial", 0, 18)); // NOI18N  
 btnLamRongNhaCungCap.setForeground(new java.awt.Color(255, 255, 255));  
 btnLamRongNhaCungCap.setText("Làm trỗng");  
 btnLamRongNhaCungCap.addActionListener(new java.awt.event.ActionListener() {  
 public void actionPerformed(java.awt.event.ActionEvent evt) {  
 btnLamRongNhaCungCapActionPerformed(evt);  
 }  
 });  
  
 javax.swing.GroupLayout jPanel6Layout = new javax.swing.GroupLayout(jPanel6);  
 jPanel6.setLayout(jPanel6Layout);  
 jPanel6Layout.setHorizontalGroup(  
 jPanel6Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)  
 .addGroup(jPanel6Layout.createSequentialGroup()  
 .addContainerGap()  
 .addGroup(jPanel6Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)  
 .addComponent(jScrollPane5, javax.swing.GroupLayout.DEFAULT\_SIZE, 825, Short.MAX\_VALUE)  
 .addGroup(jPanel6Layout.createSequentialGroup()  
 .addGroup(jPanel6Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)  
 .addComponent(jLabel2)  
 .addComponent(jLabel3)  
 .addGroup(jPanel6Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING)  
 .addComponent(btnThemNhaCungCap)  
 .addComponent(jLabel4)))  
 .addGroup(jPanel6Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)  
 .addGroup(jPanel6Layout.createSequentialGroup()  
 .addGroup(jPanel6Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING)  
 .addGroup(jPanel6Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)  
 .addComponent(tbSupplierID, javax.swing.GroupLayout.PREFERRED\_SIZE, 154, javax.swing.GroupLayout.PREFERRED\_SIZE)  
 .addComponent(tbSupplierName, javax.swing.GroupLayout.Alignment.TRAILING, javax.swing.GroupLayout.PREFERRED\_SIZE, 151, javax.swing.GroupLayout.PREFERRED\_SIZE))  
 .addComponent(tbContactInfo, javax.swing.GroupLayout.PREFERRED\_SIZE, 151, javax.swing.GroupLayout.PREFERRED\_SIZE))  
 .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)  
 .addGroup(jPanel6Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)  
 .addComponent(jLabel7)  
 .addComponent(jLabel6)  
 .addComponent(jLabel5))  
 .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)  
 .addGroup(jPanel6Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)  
 .addComponent(tbAddress)  
 .addComponent(tbPhone, javax.swing.GroupLayout.Alignment.TRAILING)  
 .addComponent(tbEmail))  
 .addContainerGap())  
 .addGroup(jPanel6Layout.createSequentialGroup()  
 .addGap(6, 6, 6)  
 .addComponent(btnSuaNhaCungCap)  
 .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)  
 .addComponent(btnXoaNhaCungCap)  
 .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)  
 .addComponent(btnLamRongNhaCungCap)  
 .addContainerGap(javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE))))))  
 );  
 jPanel6Layout.setVerticalGroup(  
 jPanel6Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)  
 .addGroup(jPanel6Layout.createSequentialGroup()  
 .addContainerGap()  
 .addComponent(jScrollPane5, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)  
 .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)  
 .addGroup(jPanel6Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)  
 .addComponent(jLabel7)  
 .addGroup(jPanel6Layout.createSequentialGroup()  
 .addComponent(tbAddress, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)  
 .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)  
 .addGroup(jPanel6Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)  
 .addComponent(tbPhone, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)  
 .addComponent(jLabel6))  
 .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)  
 .addGroup(jPanel6Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)  
 .addComponent(tbEmail, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)  
 .addComponent(jLabel5)  
 .addComponent(tbContactInfo, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)))  
 .addGroup(jPanel6Layout.createSequentialGroup()  
 .addGroup(jPanel6Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)  
 .addGroup(jPanel6Layout.createSequentialGroup()  
 .addGap(3, 3, 3)  
 .addComponent(tbSupplierID, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)  
 .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)  
 .addGroup(jPanel6Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)  
 .addComponent(tbSupplierName, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)  
 .addComponent(jLabel3)))  
 .addComponent(jLabel2))  
 .addGap(18, 18, 18)  
 .addComponent(jLabel4)))  
 .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)  
 .addGroup(jPanel6Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)  
 .addGroup(jPanel6Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)  
 .addComponent(btnXoaNhaCungCap)  
 .addComponent(btnLamRongNhaCungCap))  
 .addGroup(jPanel6Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)  
 .addComponent(btnThemNhaCungCap)  
 .addComponent(btnSuaNhaCungCap)))  
 .addContainerGap(125, Short.MAX\_VALUE))  
 );  
  
 jTabbedPane1.addTab("Nhà cung cấp", jPanel6);  
  
 btnDangXuat.setBackground(new java.awt.Color(220, 53, 69));  
 btnDangXuat.setFont(new java.awt.Font("Arial", 0, 18)); // NOI18N  
 btnDangXuat.setForeground(new java.awt.Color(255, 255, 255));  
 btnDangXuat.setText("Đăng Xuất");  
 btnDangXuat.addActionListener(new java.awt.event.ActionListener() {  
 public void actionPerformed(java.awt.event.ActionEvent evt) {  
 btnDangXuatActionPerformed(evt);  
 }  
 });  
  
 javax.swing.GroupLayout jPanel1Layout = new javax.swing.GroupLayout(jPanel1);  
 jPanel1.setLayout(jPanel1Layout);  
 jPanel1Layout.setHorizontalGroup(  
 jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)  
 .addGroup(jPanel1Layout.createSequentialGroup()  
 .addContainerGap()  
 .addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)  
 .addGroup(jPanel1Layout.createSequentialGroup()  
 .addComponent(jLabel1, javax.swing.GroupLayout.PREFERRED\_SIZE, 591, javax.swing.GroupLayout.PREFERRED\_SIZE)  
 .addGap(41, 41, 41)  
 .addComponent(btnDangXuat)  
 .addGap(0, 0, Short.MAX\_VALUE))  
 .addComponent(jTabbedPane1))  
 .addContainerGap())  
 );  
 jPanel1Layout.setVerticalGroup(  
 jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)  
 .addGroup(jPanel1Layout.createSequentialGroup()  
 .addContainerGap()  
 .addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING)  
 .addComponent(jLabel1, javax.swing.GroupLayout.PREFERRED\_SIZE, 55, javax.swing.GroupLayout.PREFERRED\_SIZE)  
 .addComponent(btnDangXuat))  
 .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)  
 .addComponent(jTabbedPane1)  
 .addContainerGap())  
 );  
  
 javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());  
 getContentPane().setLayout(layout);  
 layout.setHorizontalGroup(  
 layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)  
 .addGroup(javax.swing.GroupLayout.Alignment.TRAILING, layout.createSequentialGroup()  
 .addContainerGap()  
 .addComponent(jPanel1, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)  
 .addContainerGap())  
 );  
 layout.setVerticalGroup(  
 layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)  
 .addGroup(layout.createSequentialGroup()  
 .addContainerGap()  
 .addComponent(jPanel1, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)  
 .addContainerGap(89, Short.MAX\_VALUE))  
 );  
  
 pack();  
 }// </editor-fold>//GEN-END:initComponents  
  
 private void btnDangXuatActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event\_btnDangXuatActionPerformed  
 // TODO add your handling code here:  
 FormDangNhap loginForm = new FormDangNhap(); // Assuming LoginForm is your login screen  
 loginForm.setVisible(true);  
 this.dispose(); // Close the current window  
 }//GEN-LAST:event\_btnDangXuatActionPerformed  
  
 private void btnLamRongNhaCungCapActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event\_btnLamRongNhaCungCapActionPerformed  
 // TODO add your handling code here:  
 // Xóa nội dung của tất cả các trường nhập  
 tbSupplierID.setText("");  
 tbSupplierName.setText("");  
 tbContactInfo.setText("");  
 tbAddress.setText("");  
 tbPhone.setText("");  
 tbEmail.setText("");  
 }//GEN-LAST:event\_btnLamRongNhaCungCapActionPerformed  
  
 private void btnXoaNhaCungCapActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event\_btnXoaNhaCungCapActionPerformed  
 // TODO add your handling code here:  
 try {  
 int supplierID = Integer.parseInt(tbSupplierID.getText());  
  
 // Gọi phương thức deleteSupplier từ SupplierController  
 SupplierController supplierController = new SupplierController();  
 boolean success = supplierController.deleteSupplier(supplierID);  
  
 if (success) {  
 JOptionPane.showMessageDialog(this, "Xóa nhà cung cấp thành công!");  
 // Cập nhật lại bảng dữ liệu  
  
 showNhaCungCap();  
 loadSupplier();  
 } else {  
 JOptionPane.showMessageDialog(this, "Không thể xóa nhà cung cấp vì nhà cung cấp có sản phẩm liên quan !");  
 }  
 } catch (NumberFormatException e) {  
 JOptionPane.showMessageDialog(this, "Vui lòng nhập đúng định dạng dữ liệu cho SupplierID.");  
 } catch (Exception e) {  
 e.printStackTrace();  
 JOptionPane.showMessageDialog(this, "Đã xảy ra lỗi khi xóa nhà cung cấp.");  
 }  
   
 showSanPham();  
 showNhaCungCap();  
 loadSupplier();  
 }//GEN-LAST:event\_btnXoaNhaCungCapActionPerformed  
  
 private void btnSuaNhaCungCapActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event\_btnSuaNhaCungCapActionPerformed  
 String supplierName = tbSupplierName.getText();  
 String contactInfo = tbContactInfo.getText();  
 String address = tbAddress.getText();  
 String phone = tbPhone.getText();  
 String email = tbEmail.getText();  
  
 // Get the selected supplier ID  
 int supplierID = getSelectedSupplierID();  
  
 // Validate the supplier ID  
 if (supplierID == -1) {  
 JOptionPane.showMessageDialog(this, "Vui lòng chọn nhà cung cấp cần sửa.", "Error", JOptionPane.ERROR\_MESSAGE);  
 return;  
 }  
  
 // Sử dụng giá trị loggedInUserID đã được lưu trong lớp FormDangNhap  
 int userID = loggedInUserID;  
  
 if (userID == 0) {  
 JOptionPane.showMessageDialog(this, "Vui lòng đăng nhập trước khi thực hiện sửa nhà cung cấp.");  
 return;  
 }  
  
 // Get the current user ID  
 int updatedBy = userID;  
  
 // Tạo đối tượng Supplier với thông tin cập nhật  
 Supplier supplier = new Supplier();  
 supplier.setSupplierID(supplierID); // Set the supplier ID  
 supplier.setSupplierName(supplierName);  
 supplier.setContactInfo(contactInfo);  
 supplier.setAddress(address);  
 supplier.setPhone(phone);  
 supplier.setEmail(email);  
 supplier.setUpdatedBy(updatedBy);  
  
 // Gọi phương thức updateSupplier từ SupplierController  
 SupplierController supplierController = new SupplierController();  
 boolean success = supplierController.updateSupplier(supplier);  
  
 // Provide feedback to the user  
 if (success) {  
 JOptionPane.showMessageDialog(this, "Cập nhật nhà cung cấp thành công!");  
  
 showNhaCungCap();  
 loadSupplier();  
 } else {  
 JOptionPane.showMessageDialog(this, "Cập nhật nhà cung cấp thất bại!");  
 }  
 }//GEN-LAST:event\_btnSuaNhaCungCapActionPerformed  
  
 private int getSelectedSupplierID() {  
 int selectedRow = TableNhaCungCap.getSelectedRow(); // Assuming supplierTable is your JTable  
 if (selectedRow == -1) {  
 return -1; // No row is selected  
 }  
 // Assuming the supplier ID is in the first column  
 return (int) TableNhaCungCap.getValueAt(selectedRow, 0); // Adjust the column index as needed  
 }  
  
 private void btnThemNhaCungCapActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event\_btnThemNhaCungCapActionPerformed  
 // Lấy dữ liệu từ các trường nhập  
 String supplierName = tbSupplierName.getText();  
 String contactInfo = tbContactInfo.getText();  
 String address = tbAddress.getText();  
 String phone = tbPhone.getText();  
 String email = tbEmail.getText();  
  
 // Sử dụng giá trị loggedInUserID đã được lưu trong lớp FormDangNhap  
 int userID = loggedInUserID;  
  
 if (userID == 0) {  
 JOptionPane.showMessageDialog(this, "Vui lòng đăng nhập trước khi thực hiện thêm sản phẩm.");  
 return;  
 }  
  
 // Get the current user ID  
 int createdBy = userID;  
 int updatedBy = userID;  
  
 // Tạo đối tượng Supplier  
 Supplier supplier = new Supplier();  
 supplier.setSupplierName(supplierName);  
 supplier.setContactInfo(contactInfo);  
 supplier.setAddress(address);  
 supplier.setPhone(phone);  
 supplier.setEmail(email);  
 supplier.setCreatedBy(createdBy);  
 supplier.setUpdatedBy(updatedBy);  
  
 // Gọi phương thức addSupplier từ SupplierController  
 SupplierController supplierController = new SupplierController();  
 boolean success = supplierController.addSupplier(supplier);  
  
 if (success) {  
 JOptionPane.showMessageDialog(this, "Thêm nhà cung cấp thành công!");  
 showSanPham();  
 showNhaCungCap();  
 loadSupplier();  
 } else {  
 JOptionPane.showMessageDialog(this, "Thêm nhà cung cấp thất bại!");  
 }  
 }//GEN-LAST:event\_btnThemNhaCungCapActionPerformed  
  
 private void tbSupplierNameActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event\_tbSupplierNameActionPerformed  
 // TODO add your handling code here:  
 }//GEN-LAST:event\_tbSupplierNameActionPerformed  
  
 private void tbSupplierIDActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event\_tbSupplierIDActionPerformed  
 // TODO add your handling code here:  
 }//GEN-LAST:event\_tbSupplierIDActionPerformed  
  
 private void TableNhaCungCapMouseClicked(java.awt.event.MouseEvent evt) {//GEN-FIRST:event\_TableNhaCungCapMouseClicked  
 // TODO add your handling code here:  
 // Lấy chỉ số hàng được chọn  
 int row = TableNhaCungCap.getSelectedRow();  
 // Lấy dữ liệu từ các ô trong hàng được chọn  
 String supplierID = TableNhaCungCap.getValueAt(row, 0).toString();  
 String supplierName = TableNhaCungCap.getValueAt(row, 1).toString();  
 String contactInfo = TableNhaCungCap.getValueAt(row, 2).toString();  
 String address = TableNhaCungCap.getValueAt(row, 3).toString();  
 String phone = TableNhaCungCap.getValueAt(row, 4).toString();  
 String email = TableNhaCungCap.getValueAt(row, 5).toString();  
  
 // Hiển thị thông tin trong các ô nhập liệu hoặc cửa sổ thông tin chi tiết  
 // Ví dụ:  
 tbSupplierID.setText(supplierID);  
 tbSupplierName.setText(supplierName);  
 tbContactInfo.setText(contactInfo);  
 tbAddress.setText(address);  
 tbPhone.setText(phone);  
 tbEmail.setText(email);  
 }//GEN-LAST:event\_TableNhaCungCapMouseClicked  
  
 private void tbTenSanPhamActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event\_tbTenSanPhamActionPerformed  
 // TODO add your handling code here:  
 }//GEN-LAST:event\_tbTenSanPhamActionPerformed  
  
 private void tbDonGiaActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event\_tbDonGiaActionPerformed  
 // TODO add your handling code here:  
 }//GEN-LAST:event\_tbDonGiaActionPerformed  
  
 private void btnChonAnhSanPhamActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event\_btnChonAnhSanPhamActionPerformed  
 // Mở hộp thoại để chọn tệp hình ảnh  
 JFileChooser fileChooser = new JFileChooser();  
 fileChooser.setDialogTitle("Chọn Ảnh Sản Phẩm"); // Thiết lập tiêu đề cho hộp thoại  
 fileChooser.setFileSelectionMode(JFileChooser.FILES\_ONLY); // Chỉ cho phép chọn tệp  
  
 // Hiển thị hộp thoại chọn tệp  
 int result = fileChooser.showOpenDialog(this);  
  
 // Kiểm tra xem người dùng đã chọn tệp hay chưa  
 if (result == JFileChooser.APPROVE\_OPTION) {  
 // Lấy đường dẫn đến tệp hình ảnh đã chọn  
 File selectedFile = fileChooser.getSelectedFile();  
 String fileName = selectedFile.getName();  
  
 // Kiểm tra xem tên tệp có trùng lặp không  
 ProductController productController = new ProductController();  
 boolean fileNameExists = productController.checkFileNameExists(fileName);  
  
 if (fileNameExists) {  
 String newFileName = fileName;  
 String[] fileParts = fileName.split("\\.");  
 int counter = 1;  
  
 while (fileNameExists) {  
 newFileName = fileParts[0] + "\_" + counter + "." + fileParts[1];  
 fileNameExists = productController.checkFileNameExists(newFileName);  
 counter++;  
 }  
  
 // Hiển thị tên tệp mới trên nhãn  
 LabelAnhSanPham.setText(newFileName);  
  
 // Cập nhật lại đường dẫn đến tệp đã chọn  
 String imagePath = selectedFile.getAbsolutePath();  
 File newSelectedFile = new File("src/main/resources/images/" + newFileName);  
  
 try {  
 Files.copy(selectedFile.toPath(), newSelectedFile.toPath(), StandardCopyOption.REPLACE\_EXISTING);  
 } catch (IOException e) {  
 e.printStackTrace();  
 JOptionPane.showMessageDialog(this, "Đã xảy ra lỗi khi sao chép tệp ảnh: " + e.getMessage());  
 }  
 } else {  
 LabelAnhSanPham.setText(fileName);  
  
 try {  
 File destDir = new File("src/main/resources/images");  
 if (!destDir.exists()) {  
 destDir.mkdirs();  
 }  
  
 File destFile = new File(destDir, fileName);  
 Files.copy(selectedFile.toPath(), destFile.toPath(), StandardCopyOption.REPLACE\_EXISTING);  
 } catch (IOException e) {  
 e.printStackTrace();  
 JOptionPane.showMessageDialog(this, "Đã xảy ra lỗi khi sao chép tệp ảnh: " + e.getMessage());  
 }  
 }  
  
 // Gọi phương thức refreshTable để cập nhật lại bảng và hình ảnh  
 }  
 }//GEN-LAST:event\_btnChonAnhSanPhamActionPerformed  
  
 private void btnLamTrongActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event\_btnLamTrongActionPerformed  
 // TODO add your handling code here:  
 // Xóa nội dung trong các trường nhập liệu  
 tbTenSanPham.setText("");  
 tbDanhMuc.setText("");  
 tbSoLuong.setText("");  
 tbDonGia.setText("");  
 ComboBoxNhaCungCap.setSelectedIndex(0);  
 LabelAnhSanPham.setText("");  
 }//GEN-LAST:event\_btnLamTrongActionPerformed  
  
 private void btnXoaSanPhamActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event\_btnXoaSanPhamActionPerformed  
 // TODO add your handling code here:  
 try {  
 int productID = Integer.parseInt(tbMaSanPham.getText());  
  
 // Gọi phương thức deleteProduct từ ProductController  
 ProductController productController = new ProductController();  
 boolean success = productController.deleteProduct(productID);  
  
 if (success) {  
 JOptionPane.showMessageDialog(this, "Xóa sản phẩm thành công!");  
 // Cập nhật lại bảng dữ liệu  
  
 showSanPham();  
 refreshTable();  
 } else {  
  
 JOptionPane.showMessageDialog(this, "Xóa sản phẩm thất bại!");  
 }  
 } catch (NumberFormatException e) {  
 JOptionPane.showMessageDialog(this, "Vui lòng nhập đúng định dạng dữ liệu cho ProductID.");  
 } catch (Exception e) {  
 e.printStackTrace();  
 JOptionPane.showMessageDialog(this, "Đã xảy ra lỗi khi xóa sản phẩm.");  
 }  
 }//GEN-LAST:event\_btnXoaSanPhamActionPerformed  
  
 private void btnSuaSanPhamActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event\_btnSuaSanPhamActionPerformed  
 // Lấy ID sản phẩm cần sửa từ bảng  
 int selectedRow = TableSanPham.getSelectedRow();  
 if (selectedRow == -1) {  
 JOptionPane.showMessageDialog(this, "Vui lòng chọn một sản phẩm để sửa.");  
 return;  
 }  
 int productID = (int) TableSanPham.getValueAt(selectedRow, 0); // Assuming productID is in the first column  
  
 // Lấy thông tin mới của sản phẩm từ các trường nhập liệu  
 String productName = tbTenSanPham.getText();  
 String category = tbDanhMuc.getText();  
 int stockQuantity = Integer.parseInt(tbSoLuong.getText());  
 double unitPrice = Double.parseDouble(tbDonGia.getText());  
 String supplierName = (String) ComboBoxNhaCungCap.getSelectedItem();  
 String imageUrl = LabelAnhSanPham.getText();  
  
 // Lấy ID của nhà cung cấp từ tên nhà cung cấp  
 SupplierController supplierController = new SupplierController();  
 Supplier supplier = supplierController.getSupplierByName(supplierName);  
 if (supplier == null) {  
 JOptionPane.showMessageDialog(this, "Không tìm thấy thông tin về nhà cung cấp.");  
 return;  
 }  
 int supplierID = supplier.getSupplierID();  
  
 // Tạo đối tượng ProductImage từ đường dẫn của hình ảnh  
 ProductImage productImage = new ProductImage();  
 productImage.setImageUrl(imageUrl);  
  
 // Tạo đối tượng Product với thông tin mới  
 Product product = new Product();  
 product.setProductID(productID);  
 product.setProductName(productName);  
 product.setCategory(category);  
 product.setStockQuantity(stockQuantity);  
 product.setUnitPrice(unitPrice);  
  
 // Sử dụng ID người dùng đã đăng nhập  
 int userID = loggedInUserID;  
 if (userID == 0) {  
 JOptionPane.showMessageDialog(this, "Vui lòng đăng nhập trước khi sửa sản phẩm.");  
 return;  
 }  
  
 // Gọi phương thức updateProduct từ ProductController  
 ProductController productController = new ProductController();  
 boolean success = productController.updateProduct(product, productImage, supplierID, userID);  
  
 // Hiển thị thông báo tương ứng  
 if (success) {  
 JOptionPane.showMessageDialog(this, "Sản phẩm đã được cập nhật thành công!");  
 // Cập nhật lại dữ liệu trong bảng  
 showSanPham();  
 refreshTable();  
 } else {  
 JOptionPane.showMessageDialog(this, "Đã xảy ra lỗi khi cập nhật sản phẩm!");  
 }  
 }//GEN-LAST:event\_btnSuaSanPhamActionPerformed  
  
 private void btnThemSanPhamActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event\_btnThemSanPhamActionPerformed  
 // Lấy thông tin về sản phẩm từ các trường nhập liệu trên giao diện  
 String productName = tbTenSanPham.getText();  
 String category = tbDanhMuc.getText();  
 int stockQuantity = Integer.parseInt(tbSoLuong.getText());  
 double unitPrice = Double.parseDouble(tbDonGia.getText());  
  
 // Lấy thông tin về nhà cung cấp từ combobox hoặc trường nhập liệu tương ứng  
 String supplierName = (String) ComboBoxNhaCungCap.getSelectedItem(); // Lấy tên nhà cung cấp từ combobox  
  
 // Lấy thông tin về hình ảnh sản phẩm từ LabelAnhSanPham  
 String imageUrl = LabelAnhSanPham.getText();  
  
 // Lấy ID của nhà cung cấp từ tên nhà cung cấp  
 SupplierController supplierController = new SupplierController();  
 Supplier supplier = supplierController.getSupplierByName(supplierName);  
 if (supplier == null) {  
 JOptionPane.showMessageDialog(this, "Không tìm thấy thông tin về nhà cung cấp.");  
 return;  
 }  
 int supplierID = supplier.getSupplierID();  
  
 // Tạo một đối tượng ProductImage từ đường dẫn của hình ảnh  
 ProductImage productImage = new ProductImage();  
 productImage.setImageUrl(imageUrl);  
  
 // Tạo một đối tượng Product từ thông tin vừa lấy được  
 Product product = new Product();  
 product.setProductName(productName);  
 product.setCategory(category);  
 product.setStockQuantity(stockQuantity);  
 product.setUnitPrice(unitPrice);  
  
 // Sử dụng giá trị loggedInUserID đã được lưu trong lớp FormDangNhap  
 int userID = loggedInUserID;  
  
 if (userID == 0) {  
 JOptionPane.showMessageDialog(this, "Vui lòng đăng nhập trước khi thực hiện thêm sản phẩm." + loggedInUserID);  
 return;  
 }  
  
 // Gọi phương thức addProduct trong ProductController để thêm sản phẩm và hình ảnh vào cơ sở dữ liệu  
 ProductController productController = new ProductController();  
 boolean success = productController.addProduct(product, productImage, supplierID, userID);  
  
 // Kiểm tra xem việc thêm sản phẩm có thành công không và hiển thị thông báo tương ứng  
 if (success) {  
 JOptionPane.showMessageDialog(this, "Sản phẩm đã được thêm thành công!");  
  
 showSanPham();  
 refreshTable();  
 } else {  
 JOptionPane.showMessageDialog(this, "Đã xảy ra lỗi khi thêm sản phẩm!");  
 }  
 }//GEN-LAST:event\_btnThemSanPhamActionPerformed  
  
 private void TableSanPhamMouseClicked(java.awt.event.MouseEvent evt) {//GEN-FIRST:event\_TableSanPhamMouseClicked  
 int selectedRow = TableSanPham.getSelectedRow();  
 if (selectedRow >= 0) {  
 DefaultTableModel model = (DefaultTableModel) TableSanPham.getModel();  
 String fileName = (String) model.getValueAt(selectedRow, 11); // Lấy giá trị của cột ImageUrl  
  
 // Lấy thông tin từ dòng được chọn  
 int productID = (int) model.getValueAt(selectedRow, 0);  
 String productName = (String) model.getValueAt(selectedRow, 1);  
 String category = (String) model.getValueAt(selectedRow, 2);  
 int stockQuantity = (int) model.getValueAt(selectedRow, 3);  
 double unitPrice = (double) model.getValueAt(selectedRow, 4);  
 String supplierName = (String) model.getValueAt(selectedRow, 5);  
  
 // Hiển thị thông tin sản phẩm trên giao diện  
 tbMaSanPham.setText(String.valueOf(productID));  
 tbTenSanPham.setText(productName);  
 tbDanhMuc.setText(category);  
 tbSoLuong.setText(String.valueOf(stockQuantity));  
 tbDonGia.setText(String.valueOf(unitPrice));  
 ComboBoxNhaCungCap.setSelectedItem(supplierName);  
  
 String imagePath = "src/main/resources/images/" + fileName;  
 File imgFile1 = new File(imagePath);  
 if (imgFile1.exists()) {  
 ImageIcon imageIcon = new ImageIcon(imagePath);  
 Image image = imageIcon.getImage().getScaledInstance(250, 250, Image.SCALE\_SMOOTH);  
 LabelAnhSanPham.setIcon(new ImageIcon(image));  
 LabelAnhSanPham.setText(fileName);  
 } else {  
 LabelAnhSanPham.setIcon(null);  
 LabelAnhSanPham.setText("Ảnh không tồn tại");  
 }  
 }  
 }//GEN-LAST:event\_TableSanPhamMouseClicked  
  
 */\*\*  
 \* @param args the command line arguments  
 \*/* public static void main(String args[]) {  
 final int[] loggedInUserID = {0}; // Khởi tạo với một giá trị mặc định  
  
 // Thử lấy ID của người dùng đã đăng nhập từ FormDangNhap  
 try {  
 FormDangNhap form = new FormDangNhap();  
 loggedInUserID[0] = form.getLoggedInUserID(); // Gọi phương thức không tĩnh trên một thể hiện  
 } catch (Exception e) {  
 // Xử lý trường hợp người dùng chưa đăng nhập  
 System.out.println("Người dùng chưa đăng nhập. Vui lòng đăng nhập trước.");  
 // Có thể chuyển hướng người dùng đến form đăng nhập hoặc xử lý theo logic của ứng dụng  
 return; // Kết thúc chương trình hoặc thực hiện bất kỳ hành động cần thiết khác  
 }  
  
 // Kiểm tra giá trị của loggedInUserID trước khi sử dụng  
 if (loggedInUserID[0] == 0) {  
 System.out.println("Người dùng chưa đăng nhập. Vui lòng đăng nhập trước.");  
 return;  
 }  
 /\* Set the Nimbus look and feel \*/  
 //<editor-fold defaultstate="collapsed" desc=" Look and feel setting code (optional) ">  
 /\* If Nimbus (introduced in Java SE 6) is not available, stay with the default look and feel.  
 \* For details see http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html  
 \*/  
 try {  
 for (javax.swing.UIManager.LookAndFeelInfo info : javax.swing.UIManager.getInstalledLookAndFeels()) {  
 if ("Nimbus".equals(info.getName())) {  
 javax.swing.UIManager.setLookAndFeel(info.getClassName());  
 break;  
 }  
 }  
 } catch (ClassNotFoundException ex) {  
 java.util.logging.Logger.getLogger(FormQuanLyHang.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);  
 } catch (InstantiationException ex) {  
 java.util.logging.Logger.getLogger(FormQuanLyHang.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);  
 } catch (IllegalAccessException ex) {  
 java.util.logging.Logger.getLogger(FormQuanLyHang.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);  
 } catch (javax.swing.UnsupportedLookAndFeelException ex) {  
 java.util.logging.Logger.getLogger(FormQuanLyHang.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);  
 }  
 //</editor-fold>  
  
 /\* Create and display the form \*/  
 java.awt.EventQueue.invokeLater(new Runnable() {  
 public void run() {  
 new FormQuanLyHang(loggedInUserID[0]).setVisible(true);  
 System.out.println(loggedInUserID[0]);  
 }  
 });  
 }  
  
 // Variables declaration - do not modify//GEN-BEGIN:variables  
 private javax.swing.JComboBox<String> ComboBoxNhaCungCap;  
 private javax.swing.JLabel LabelAnhSanPham;  
 private javax.swing.JTable TableNhaCungCap;  
 private javax.swing.JTable TableSanPham;  
 private javax.swing.JButton btnChonAnhSanPham;  
 private javax.swing.JButton btnDangXuat;  
 private javax.swing.JButton btnLamRongNhaCungCap;  
 private javax.swing.JButton btnLamTrong;  
 private javax.swing.JButton btnSuaNhaCungCap;  
 private javax.swing.JButton btnSuaSanPham;  
 private javax.swing.JButton btnThemNhaCungCap;  
 private javax.swing.JButton btnThemSanPham;  
 private javax.swing.JButton btnXoaNhaCungCap;  
 private javax.swing.JButton btnXoaSanPham;  
 private javax.swing.JLabel jLabel1;  
 private javax.swing.JLabel jLabel10;  
 private javax.swing.JLabel jLabel11;  
 private javax.swing.JLabel jLabel12;  
 private javax.swing.JLabel jLabel13;  
 private javax.swing.JLabel jLabel2;  
 private javax.swing.JLabel jLabel3;  
 private javax.swing.JLabel jLabel4;  
 private javax.swing.JLabel jLabel5;  
 private javax.swing.JLabel jLabel6;  
 private javax.swing.JLabel jLabel7;  
 private javax.swing.JLabel jLabel8;  
 private javax.swing.JLabel jLabel9;  
 private javax.swing.JPanel jPanel1;  
 private javax.swing.JPanel jPanel5;  
 private javax.swing.JPanel jPanel6;  
 private javax.swing.JPanel jPanel7;  
 private javax.swing.JPanel jPanel8;  
 private javax.swing.JScrollPane jScrollPane4;  
 private javax.swing.JScrollPane jScrollPane5;  
 private javax.swing.JTabbedPane jTabbedPane1;  
 private javax.swing.JTextField tbAddress;  
 private javax.swing.JTextField tbContactInfo;  
 private javax.swing.JTextField tbDanhMuc;  
 private javax.swing.JTextField tbDonGia;  
 private javax.swing.JTextField tbEmail;  
 private javax.swing.JTextField tbMaSanPham;  
 private javax.swing.JTextField tbPhone;  
 private javax.swing.JTextField tbSoLuong;  
 private javax.swing.JTextField tbSupplierID;  
 private javax.swing.JTextField tbSupplierName;  
 private javax.swing.JTextField tbTenSanPham;  
 // End of variables declaration//GEN-END:variables  
}

# II.THỰC THỂ

## 1. Product

package org.example.models;  
  
import lombok.Data;  
  
import java.util.Date;  
@Data  
public class Product {  
 private int productID;  
 private String productName;  
 private String category;  
 private int stockQuantity;  
 private double unitPrice;  
 private int supplierID;  
 private int createdBy;  
 private int updatedBy;  
 private Date updatedAt;  
  
 // Constructors, getters, and setters  
}

## 2. Supplier

package org.example.models;  
  
import lombok.Data;  
  
import java.util.Date;  
@Data  
public class Supplier {  
 private int supplierID;  
 private String supplierName;  
 private String contactInfo;  
 private String address;  
 private String phone;  
 private String email;  
 private int createdBy;  
 private int updatedBy;  
 private Date updatedAt;  
  
 // Constructors, getters, and setters  
}

# III. CONTOLLER

## 1. ProductController

package org.example.controllers;  
  
import org.example.connect.MyConnection;  
import org.example.models.\*;  
  
import javax.swing.\*;  
import javax.swing.table.DefaultTableModel;  
import java.awt.\*;  
import java.math.BigDecimal;  
import java.sql.\*;  
import java.util.ArrayList;  
import java.util.Date;  
import java.util.List;  
  
public class ProductController extends Component {  
  
 private Connection connection;  
  
 public ProductController() {  
 // Initialize database connection  
 this.connection = MyConnection.*getConnection*();  
 }  
  
 public DefaultTableModel getAllProducts() {  
 String[] columnNames = {"Mã sản phẩm", "Tên sản phẩm", "Danh mục", "Số lượng tồn kho", "Đơn giá", "Mã nhà cung cấp", "Tạo bởi", "Cập nhật bởi", "Thời gian cập nhật"};  
 DefaultTableModel model = new DefaultTableModel(columnNames, 0);  
  
 String sql = "SELECT \* FROM Products";  
  
 try (Connection conn = MyConnection.*getConnection*();  
 PreparedStatement preparedStatement = conn.prepareStatement(sql);  
 ResultSet resultSet = preparedStatement.executeQuery()) {  
  
 while (resultSet.next()) {  
 int productID = resultSet.getInt("ProductID");  
 String productName = resultSet.getString("ProductName");  
 String category = resultSet.getString("Category");  
 int stockQuantity = resultSet.getInt("StockQuantity");  
 double unitPrice = resultSet.getDouble("UnitPrice");  
 int supplierID = resultSet.getInt("SupplierID");  
 int createdBy = resultSet.getInt("CreatedBy");  
 int updatedBy = resultSet.getInt("UpdatedBy");  
 Date updatedAt = resultSet.getTimestamp("UpdatedAt");  
  
 model.addRow(new Object[]{productID, productName, category, stockQuantity, unitPrice, supplierID, createdBy, updatedBy, updatedAt});  
 }  
  
 } catch (SQLException e) {  
 e.printStackTrace();  
 }  
  
 return model;  
 }  
  
  
 public DefaultTableModel getAllProductDetails() {  
 String[] columnNames = {"Mã sản phẩm", "Tên sản phẩm", "Danh mục", "Số lượng tồn kho", "Đơn giá", "Nhà cung cấp", "Tạo bởi", "Người tạo", "Cập nhật bởi", "Người cập nhật", "Thời gian cập nhật", "Hình ảnh"};  
 DefaultTableModel model = new DefaultTableModel(columnNames, 0);  
  
 String sql = "SELECT " +  
 "p.ProductID AS 'Mã sản phẩm', " +  
 "p.ProductName AS 'Tên sản phẩm', " +  
 "p.Category AS 'Danh mục', " +  
 "p.StockQuantity AS 'Số lượng tồn kho', " +  
 "p.UnitPrice AS 'Đơn giá', " +  
 "s.SupplierName AS 'Nhà cung cấp', " +  
 "p.CreatedBy AS 'Tạo bởi', " +  
 "uc.Username AS 'Người tạo', " +  
 "p.UpdatedBy AS 'Cập nhật bởi', " +  
 "uu.Username AS 'Người cập nhật', " +  
 "p.UpdatedAt AS 'Thời gian cập nhật', " +  
 "pi.ImageUrl AS 'Hình ảnh' " +  
 "FROM Products p " +  
 "LEFT JOIN Suppliers s ON p.SupplierID = s.SupplierID " +  
 "LEFT JOIN Users uc ON p.CreatedBy = uc.UserID " +  
 "LEFT JOIN Users uu ON p.UpdatedBy = uu.UserID " +  
 "LEFT JOIN ProductImages pi ON p.ProductID = pi.ProductID";  
  
 try (Connection conn = MyConnection.*getConnection*();  
 PreparedStatement preparedStatement = conn.prepareStatement(sql);  
 ResultSet resultSet = preparedStatement.executeQuery()) {  
  
 while (resultSet.next()) {  
 int productID = resultSet.getInt("Mã sản phẩm");  
 String productName = resultSet.getString("Tên sản phẩm");  
 String category = resultSet.getString("Danh mục");  
 int stockQuantity = resultSet.getInt("Số lượng tồn kho");  
 double unitPrice = resultSet.getDouble("Đơn giá");  
 String supplierName = resultSet.getString("Nhà cung cấp");  
 int createdBy = resultSet.getInt("Tạo bởi");  
 String creatorUsername = resultSet.getString("Người tạo");  
 int updatedBy = resultSet.getInt("Cập nhật bởi");  
 String updaterUsername = resultSet.getString("Người cập nhật");  
 java.util.Date updatedAt = resultSet.getTimestamp("Thời gian cập nhật");  
 String imageUrl = resultSet.getString("Hình ảnh");  
  
 model.addRow(new Object[]{productID, productName, category, stockQuantity, unitPrice, supplierName, createdBy, creatorUsername, updatedBy, updaterUsername, updatedAt, imageUrl});  
 }  
  
 } catch (SQLException e) {  
 e.printStackTrace();  
 }  
  
 return model;  
 }  
  
  
 public boolean addProduct(Product product, ProductImage productImage, int supplierID, int userID) {  
 String insertProductQuery = "INSERT INTO Products (ProductName, Category, StockQuantity, UnitPrice, SupplierID, CreatedBy, UpdatedBy, UpdatedAt) " +  
 "VALUES (?, ?, ?, ?, ?, ?, ?, ?)";  
  
 String insertProductImageQuery = "INSERT INTO ProductImages (ProductID, ImageUrl, CreatedAt) VALUES (?, ?, ?)";  
  
 try {  
 // Thêm sản phẩm  
 PreparedStatement insertProductStmt = connection.prepareStatement(insertProductQuery, Statement.*RETURN\_GENERATED\_KEYS*);  
 insertProductStmt.setString(1, product.getProductName());  
 insertProductStmt.setString(2, product.getCategory());  
 insertProductStmt.setInt(3, product.getStockQuantity());  
 insertProductStmt.setDouble(4, product.getUnitPrice());  
 insertProductStmt.setInt(5, supplierID);  
 insertProductStmt.setInt(6, userID); // Sử dụng userID cho trường CreatedBy  
 insertProductStmt.setInt(7, userID); // Sử dụng userID cho trường UpdatedBy  
 insertProductStmt.setTimestamp(8, new Timestamp(new Date().getTime()));  
 insertProductStmt.executeUpdate();  
  
 // Lấy ID của sản phẩm vừa thêm  
 ResultSet generatedKeys = insertProductStmt.getGeneratedKeys();  
 int productID = 0;  
 if (generatedKeys.next()) {  
 productID = generatedKeys.getInt(1);  
 } else {  
 throw new SQLException("Creating product failed, no ID obtained.");  
 }  
  
 // Thêm hình ảnh sản phẩm  
 PreparedStatement insertProductImageStmt = connection.prepareStatement(insertProductImageQuery);  
 insertProductImageStmt.setInt(1, productID);  
 insertProductImageStmt.setString(2, productImage.getImageUrl());  
 insertProductImageStmt.setTimestamp(3, new Timestamp(new Date().getTime()));  
 insertProductImageStmt.executeUpdate();  
  
 return true;  
 } catch (SQLException e) {  
 e.printStackTrace();  
 return false;  
 }  
 }  
  
  
 public boolean giaoDichSanPham(SalesReceiptDetail salesReceiptDetail, String customerName, String paymentMethod, int employeeID, int userID) {  
 String insertSalesReceiptSQL = "INSERT INTO SalesReceipts (EmployeeID, SaleDate, TotalAmount, CustomerName, PaymentMethod, CreatedBy, UpdatedBy) VALUES (?, ?, ?, ?, ?, ?, ?)";  
 String insertSalesReceiptDetailSQL = "INSERT INTO SalesReceiptDetails (SalesReceiptID, ProductID, Quantity, UnitPrice, TotalPrice, CreatedBy, UpdatedBy) VALUES (?, ?, ?, ?, ?, ?, ?)";  
 String updateProductStockSQL = "UPDATE Products SET StockQuantity = StockQuantity - ? WHERE ProductID = ?";  
  
 try (Connection conn = MyConnection.*getConnection*();  
 PreparedStatement insertSalesReceiptStmt = conn.prepareStatement(insertSalesReceiptSQL, Statement.*RETURN\_GENERATED\_KEYS*);  
 PreparedStatement insertSalesReceiptDetailStmt = conn.prepareStatement(insertSalesReceiptDetailSQL);  
 PreparedStatement updateStmt = conn.prepareStatement(updateProductStockSQL)) {  
  
 // Thực hiện giao dịch trong một transaction  
 conn.setAutoCommit(false);  
  
 // Thêm thông tin phiếu bán hàng vào bảng SalesReceipts  
 insertSalesReceiptStmt.setInt(1, employeeID);  
 // Thay đổi SaleDate thành ngày hiện tại hoặc ngày của giao dịch  
 insertSalesReceiptStmt.setDate(2, new java.sql.Date(System.*currentTimeMillis*()));  
 // Thay đổi TotalAmount thành tổng giá tiền của chi tiết phiếu bán hàng  
 insertSalesReceiptStmt.setBigDecimal(3, BigDecimal.*valueOf*(salesReceiptDetail.getTotalPrice()));  
 insertSalesReceiptStmt.setString(4, customerName); // Sử dụng tên khách hàng từ tham số  
 insertSalesReceiptStmt.setString(5, paymentMethod); // Sử dụng phương thức thanh toán từ tham số  
 insertSalesReceiptStmt.setInt(6, userID);  
 insertSalesReceiptStmt.setInt(7, userID);  
 insertSalesReceiptStmt.executeUpdate();  
  
 // Lấy SalesReceiptID vừa được tạo  
 ResultSet generatedKeys = insertSalesReceiptStmt.getGeneratedKeys();  
 int salesReceiptID = -1;  
 if (generatedKeys.next()) {  
 salesReceiptID = generatedKeys.getInt(1);  
 }  
  
 // Thêm thông tin chi tiết phiếu bán hàng vào bảng SalesReceiptDetails  
 insertSalesReceiptDetailStmt.setInt(1, salesReceiptID);  
 insertSalesReceiptDetailStmt.setInt(2, salesReceiptDetail.getProductID());  
 insertSalesReceiptDetailStmt.setInt(3, salesReceiptDetail.getQuantity());  
 insertSalesReceiptDetailStmt.setBigDecimal(4, BigDecimal.*valueOf*(salesReceiptDetail.getUnitPrice()));  
 insertSalesReceiptDetailStmt.setBigDecimal(5, BigDecimal.*valueOf*(salesReceiptDetail.getTotalPrice()));  
 insertSalesReceiptDetailStmt.setInt(6, userID);  
 insertSalesReceiptDetailStmt.setInt(7, userID);  
 insertSalesReceiptDetailStmt.executeUpdate();  
  
 // Cập nhật số lượng sản phẩm trong bảng Products  
 updateStmt.setInt(1, salesReceiptDetail.getQuantity());  
 updateStmt.setInt(2, salesReceiptDetail.getProductID());  
 updateStmt.executeUpdate();  
  
 // Commit transaction  
 conn.commit();  
  
 return true; // Giao dịch thành công  
  
 } catch (SQLException e) {  
 // Rollback transaction nếu có lỗi  
 try {  
 connection.rollback();  
 } catch (SQLException rollbackEx) {  
 rollbackEx.printStackTrace();  
 }  
 e.printStackTrace();  
 return false; // Giao dịch không thành công  
 }  
 }  
  
  
 private int getInsertedSupplierID() throws SQLException {  
 // Lấy ID của nhà cung cấp vừa được thêm vào  
 String selectLastSupplierIDQuery = "SELECT LAST\_INSERT\_ID()";  
 PreparedStatement selectLastSupplierIDStmt = connection.prepareStatement(selectLastSupplierIDQuery);  
 ResultSet resultSet = selectLastSupplierIDStmt.executeQuery();  
 resultSet.next();  
 return resultSet.getInt(1);  
 }  
  
 private int getInsertedProductID() throws SQLException {  
 // Lấy ID của sản phẩm vừa được thêm vào  
 String selectLastProductIDQuery = "SELECT LAST\_INSERT\_ID()";  
 PreparedStatement selectLastProductIDStmt = connection.prepareStatement(selectLastProductIDQuery);  
 ResultSet resultSet = selectLastProductIDStmt.executeQuery();  
 resultSet.next();  
 return resultSet.getInt(1);  
 }  
  
 public boolean updateProduct(Product product, ProductImage productImage, int supplierID, int userID) {  
 String updateProductQuery = "UPDATE Products SET ProductName = ?, Category = ?, StockQuantity = ?, UnitPrice = ?, SupplierID = ?, UpdatedBy = ?, UpdatedAt = ? WHERE ProductID = ?";  
 String updateProductImageQuery = "UPDATE ProductImages SET ImageUrl = ? WHERE ProductID = ?";  
  
 try {  
 // Bắt đầu giao dịch  
 connection.setAutoCommit(false);  
  
 // Cập nhật thông tin sản phẩm  
 PreparedStatement updateProductStmt = connection.prepareStatement(updateProductQuery);  
 updateProductStmt.setString(1, product.getProductName());  
 updateProductStmt.setString(2, product.getCategory());  
 updateProductStmt.setInt(3, product.getStockQuantity());  
 updateProductStmt.setDouble(4, product.getUnitPrice());  
 updateProductStmt.setInt(5, supplierID);  
 updateProductStmt.setInt(6, userID);  
 updateProductStmt.setTimestamp(7, new Timestamp(new Date().getTime()));  
 updateProductStmt.setInt(8, product.getProductID());  
 updateProductStmt.executeUpdate();  
  
 // Cập nhật thông tin hình ảnh sản phẩm  
 PreparedStatement updateProductImageStmt = connection.prepareStatement(updateProductImageQuery);  
 updateProductImageStmt.setString(1, productImage.getImageUrl());  
 updateProductImageStmt.setInt(2, product.getProductID());  
 updateProductImageStmt.executeUpdate();  
  
 // Hoàn thành giao dịch  
 connection.commit();  
 return true;  
 } catch (SQLException e) {  
 try {  
 // Rollback nếu có lỗi  
 connection.rollback();  
 } catch (SQLException rollbackEx) {  
 rollbackEx.printStackTrace();  
 }  
 e.printStackTrace();  
 return false;  
 } finally {  
 try {  
 // Đặt lại trạng thái tự động commit  
 connection.setAutoCommit(true);  
 } catch (SQLException ex) {  
 ex.printStackTrace();  
 }  
 }  
 }  
  
  
 public boolean deleteProduct(int productID) {  
 String deleteProductImageQuery = "DELETE FROM ProductImages WHERE ProductID = ?";  
 String deletePricingQuery = "DELETE FROM Pricing WHERE ProductID = ?";  
 String deleteProductQuery = "DELETE FROM Products WHERE ProductID = ?";  
 String deleteSalesReceiptDetailsQuery = "DELETE FROM salesreceiptdetails WHERE ProductID = ?";  
  
 try {  
 // Begin transaction  
 connection.setAutoCommit(false);  
  
 // Delete product images  
 PreparedStatement deleteProductImageStmt = connection.prepareStatement(deleteProductImageQuery);  
 deleteProductImageStmt.setInt(1, productID);  
 deleteProductImageStmt.executeUpdate();  
  
 // Delete related pricing rows  
 PreparedStatement deletePricingStmt = connection.prepareStatement(deletePricingQuery);  
 deletePricingStmt.setInt(1, productID);  
 deletePricingStmt.executeUpdate();  
  
 // Delete related sales receipt details  
 PreparedStatement deleteSalesReceiptDetailsStmt = connection.prepareStatement(deleteSalesReceiptDetailsQuery);  
 deleteSalesReceiptDetailsStmt.setInt(1, productID);  
 deleteSalesReceiptDetailsStmt.executeUpdate();  
  
 // Delete product  
 PreparedStatement deleteProductStmt = connection.prepareStatement(deleteProductQuery);  
 deleteProductStmt.setInt(1, productID);  
 deleteProductStmt.executeUpdate();  
  
 // Commit transaction  
 connection.commit();  
 return true;  
 } catch (SQLException e) {  
 try {  
 // Rollback if there's an error  
 connection.rollback();  
 } catch (SQLException rollbackEx) {  
 rollbackEx.printStackTrace();  
 }  
  
 // Handle the specific SQLIntegrityConstraintViolationException  
 if (e instanceof SQLIntegrityConstraintViolationException) {  
 SQLIntegrityConstraintViolationException sqlEx = (SQLIntegrityConstraintViolationException) e;  
 if (sqlEx.getErrorCode() == 1451 || sqlEx.getErrorCode() == 1452) {  
 // Display a user-friendly error message for foreign key constraint violation  
 JOptionPane.*showMessageDialog*(null, "Không thể xóa sản phẩm vì có dữ liệu liên quan đến sản phẩm này.");  
 return false;  
 }  
 }  
  
 e.printStackTrace();  
 return false;  
 } finally {  
 try {  
 // Reset auto-commit mode  
 connection.setAutoCommit(true);  
 } catch (SQLException ex) {  
 ex.printStackTrace();  
 }  
 }  
 }  
  
  
 public boolean checkFileNameExists(String fileName) {  
 try {  
 // Prepare the SQL statement to check if the file name exists  
 String query = "SELECT COUNT(\*) AS count FROM ProductImages WHERE ImageUrl = ?";  
 try (PreparedStatement statement = connection.prepareStatement(query)) {  
 // Set the file name parameter in the prepared statement  
 statement.setString(1, fileName);  
  
 // Execute the query  
 try (ResultSet resultSet = statement.executeQuery()) {  
 // Check if any row exists with the given file name  
 if (resultSet.next()) {  
 int count = resultSet.getInt("count");  
 return count > 0;  
 }  
 }  
 }  
 } catch (SQLException e) {  
 e.printStackTrace();  
 }  
 return false;  
 }  
  
  
 public int getStockQuantity(int productID) {  
 String query = "SELECT StockQuantity FROM Products WHERE ProductID = ?";  
 try (PreparedStatement stmt = connection.prepareStatement(query)) {  
 stmt.setInt(1, productID);  
 try (ResultSet rs = stmt.executeQuery()) {  
 if (rs.next()) {  
 return rs.getInt("StockQuantity");  
 } else {  
 throw new SQLException("Product not found.");  
 }  
 }  
 } catch (SQLException e) {  
 e.printStackTrace();  
 return -1; // Trả về -1 nếu có lỗi xảy ra  
 }  
 }  
  
 public double getUnitPrice(int productID) throws SQLException {  
 String query = "SELECT UnitPrice FROM Products WHERE ProductID = ?";  
 try (PreparedStatement stmt = connection.prepareStatement(query)) {  
 stmt.setInt(1, productID);  
 try (ResultSet rs = stmt.executeQuery()) {  
 if (rs.next()) {  
 return rs.getDouble("UnitPrice");  
 } else {  
 throw new SQLException("Product not found.");  
 }  
 }  
 }  
 }  
  
 public DefaultTableModel getAllProductGiaoDichDetails() {  
 String[] columnNames = {"Mã sản phẩm", "Tên sản phẩm", "Danh mục", "Số lượng tồn kho", "Đơn giá", "Nhà cung cấp", "Tạo bởi", "Người tạo", "Cập nhật bởi", "Người cập nhật", "Thời gian cập nhật", "Hình ảnh"};  
 DefaultTableModel model = new DefaultTableModel(columnNames, 0);  
  
 String sql = "SELECT " +  
 "p.ProductID AS 'Mã sản phẩm', " +  
 "p.ProductName AS 'Tên sản phẩm', " +  
 "p.Category AS 'Danh mục', " +  
 "p.StockQuantity AS 'Số lượng tồn kho', " +  
 "p.UnitPrice AS 'Đơn giá', " +  
 "s.SupplierName AS 'Nhà cung cấp', " +  
 "p.CreatedBy AS 'Tạo bởi', " +  
 "uc.Username AS 'Người tạo', " +  
 "p.UpdatedBy AS 'Cập nhật bởi', " +  
 "uu.Username AS 'Người cập nhật', " +  
 "p.UpdatedAt AS 'Thời gian cập nhật', " +  
 "pi.ImageUrl AS 'Hình ảnh' " +  
 "FROM Products p " +  
 "LEFT JOIN Suppliers s ON p.SupplierID = s.SupplierID " +  
 "LEFT JOIN Users uc ON p.CreatedBy = uc.UserID " +  
 "LEFT JOIN Users uu ON p.UpdatedBy = uu.UserID " +  
 "LEFT JOIN ProductImages pi ON p.ProductID = pi.ProductID";  
  
 try (Connection conn = MyConnection.*getConnection*();  
 PreparedStatement preparedStatement = conn.prepareStatement(sql);  
 ResultSet resultSet = preparedStatement.executeQuery()) {  
  
 while (resultSet.next()) {  
 int productID = resultSet.getInt("Mã sản phẩm");  
 String productName = resultSet.getString("Tên sản phẩm");  
 String category = resultSet.getString("Danh mục");  
 int stockQuantity = resultSet.getInt("Số lượng tồn kho");  
 double unitPrice = resultSet.getDouble("Đơn giá");  
 String supplierName = resultSet.getString("Nhà cung cấp");  
 int createdBy = resultSet.getInt("Tạo bởi");  
 String creatorUsername = resultSet.getString("Người tạo");  
 int updatedBy = resultSet.getInt("Cập nhật bởi");  
 String updaterUsername = resultSet.getString("Người cập nhật");  
 java.util.Date updatedAt = resultSet.getTimestamp("Thời gian cập nhật");  
 String imageUrl = resultSet.getString("Hình ảnh");  
  
 model.addRow(new Object[]{productID, productName, category, stockQuantity, unitPrice, supplierName, createdBy, creatorUsername, updatedBy, updaterUsername, updatedAt, imageUrl});  
 }  
  
 } catch (SQLException e) {  
 e.printStackTrace();  
 }  
  
 return model;  
 }  
  
 public boolean addSalesReceiptDetail(SalesReceiptDetail salesReceiptDetail) {  
 String insertSalesReceiptDetailQuery = "INSERT INTO SalesReceiptDetails (SalesReceiptID, ProductID, Quantity, UnitPrice, TotalPrice, CreatedBy, UpdatedBy) " +  
 "VALUES (?, ?, ?, ?, ?, ?, ?)";  
  
 try {  
 // Thực hiện thêm chi tiết giao dịch vào cơ sở dữ liệu  
 PreparedStatement insertSalesReceiptDetailStmt = connection.prepareStatement(insertSalesReceiptDetailQuery);  
 insertSalesReceiptDetailStmt.setInt(1, salesReceiptDetail.getSalesReceiptID());  
 insertSalesReceiptDetailStmt.setInt(2, salesReceiptDetail.getProductID());  
 insertSalesReceiptDetailStmt.setInt(3, salesReceiptDetail.getQuantity());  
 insertSalesReceiptDetailStmt.setDouble(4, salesReceiptDetail.getUnitPrice());  
 insertSalesReceiptDetailStmt.setDouble(5, salesReceiptDetail.getTotalPrice());  
 insertSalesReceiptDetailStmt.setInt(6, salesReceiptDetail.getCreatedBy());  
 insertSalesReceiptDetailStmt.setInt(7, salesReceiptDetail.getUpdatedBy());  
 insertSalesReceiptDetailStmt.executeUpdate();  
 return true;  
 } catch (SQLException e) {  
 e.printStackTrace();  
 return false;  
 }  
 }  
  
 public boolean updateStockQuantity(int productID, int newStockQuantity) {  
 String updateStockQuantityQuery = "UPDATE Products SET StockQuantity = ? WHERE ProductID = ?";  
  
 try {  
 // Thực hiện cập nhật số lượng hàng trong cơ sở dữ liệu  
 PreparedStatement updateStockQuantityStmt = connection.prepareStatement(updateStockQuantityQuery);  
 updateStockQuantityStmt.setInt(1, newStockQuantity);  
 updateStockQuantityStmt.setInt(2, productID);  
 updateStockQuantityStmt.executeUpdate();  
 return true;  
 } catch (SQLException e) {  
 e.printStackTrace();  
 return false;  
 }  
 }  
  
}

## 2. SupplierController

package org.example.controllers;  
  
import org.example.connect.MyConnection;  
import org.example.models.Supplier;  
  
import javax.swing.table.DefaultTableModel;  
import java.sql.Connection;  
import java.sql.PreparedStatement;  
import java.sql.ResultSet;  
import java.sql.SQLException;  
import java.util.List;  
import java.util.Optional;  
  
public class SupplierController {  
  
 private Connection connection;  
  
 public SupplierController() {  
 // Initialize database connection  
 this.connection = MyConnection.*getConnection*();  
 }  
  
 // Phương thức để lấy thông tin nhà cung cấp từ tên nhà cung cấp  
 public Supplier getSupplierByName(String supplierName) {  
 Supplier supplier = null;  
 String query = "SELECT \* FROM Suppliers WHERE SupplierName = ?";  
 try (PreparedStatement statement = connection.prepareStatement(query)) {  
 statement.setString(1, supplierName);  
 ResultSet resultSet = statement.executeQuery();  
 if (resultSet.next()) {  
 supplier = new Supplier();  
 supplier.setSupplierID(resultSet.getInt("supplierID"));  
 supplier.setSupplierName(resultSet.getString("supplierName"));  
 supplier.setContactInfo(resultSet.getString("contactInfo"));  
 supplier.setAddress(resultSet.getString("address"));  
 supplier.setPhone(resultSet.getString("phone"));  
 supplier.setEmail(resultSet.getString("email"));  
 supplier.setCreatedBy(resultSet.getInt("createdBy"));  
 supplier.setUpdatedBy(resultSet.getInt("updatedBy"));  
 supplier.setUpdatedAt(resultSet.getDate("updatedAt"));  
 }  
 } catch (SQLException e) {  
 e.printStackTrace();  
 }  
 return supplier;  
 }  
  
  
 public DefaultTableModel getAllSuppliers() {  
 String[] columnNames = {  
 "SupplierID", "SupplierName", "ContactInfo", "Address",  
 "Phone", "Email", "CreatedBy", "UpdatedBy", "UpdatedAt"  
 };  
 DefaultTableModel model = new DefaultTableModel(columnNames, 0);  
  
 String sql = "SELECT \* FROM Suppliers";  
  
 try (Connection conn = MyConnection.*getConnection*();  
 PreparedStatement preparedStatement = conn.prepareStatement(sql);  
 ResultSet resultSet = preparedStatement.executeQuery()) {  
  
 while (resultSet.next()) {  
 int supplierID = resultSet.getInt("SupplierID");  
 String supplierName = resultSet.getString("SupplierName");  
 String contactInfo = resultSet.getString("ContactInfo");  
 String address = resultSet.getString("Address");  
 String phone = resultSet.getString("Phone");  
 String email = resultSet.getString("Email");  
 int createdBy = resultSet.getInt("CreatedBy");  
 int updatedBy = resultSet.getInt("UpdatedBy");  
 java.sql.Timestamp updatedAt = resultSet.getTimestamp("UpdatedAt");  
  
 model.addRow(new Object[]{  
 supplierID, supplierName, contactInfo, address,  
 phone, email, createdBy, updatedBy, updatedAt  
 });  
 }  
 } catch (Exception e) {  
 e.printStackTrace();  
 }  
  
 return model;  
 }  
  
  
  
 public boolean addSupplier(Supplier supplier) {  
 String sql = "INSERT INTO Suppliers (SupplierName, ContactInfo, Address, Phone, Email, CreatedBy, UpdatedBy) VALUES (?, ?, ?, ?, ?, ?, ?)";  
 try (Connection conn = MyConnection.*getConnection*();  
 PreparedStatement preparedStatement = conn.prepareStatement(sql)) {  
 preparedStatement.setString(1, supplier.getSupplierName());  
 preparedStatement.setString(2, supplier.getContactInfo());  
 preparedStatement.setString(3, supplier.getAddress());  
 preparedStatement.setString(4, supplier.getPhone());  
 preparedStatement.setString(5, supplier.getEmail());  
 preparedStatement.setInt(6, supplier.getCreatedBy());  
 preparedStatement.setInt(7, supplier.getUpdatedBy());  
 return preparedStatement.executeUpdate() > 0;  
 } catch (Exception e) {  
 e.printStackTrace();  
 return false;  
 }  
 }  
  
 public boolean updateSupplier(Supplier supplier) {  
 String sql = "UPDATE Suppliers SET SupplierName = ?, ContactInfo = ?, Address = ?, Phone = ?, Email = ?, UpdatedBy = ? WHERE SupplierID = ?";  
 try (Connection conn = MyConnection.*getConnection*();  
 PreparedStatement preparedStatement = conn.prepareStatement(sql)) {  
 preparedStatement.setString(1, supplier.getSupplierName());  
 preparedStatement.setString(2, supplier.getContactInfo());  
 preparedStatement.setString(3, supplier.getAddress());  
 preparedStatement.setString(4, supplier.getPhone());  
 preparedStatement.setString(5, supplier.getEmail());  
 preparedStatement.setInt(6, supplier.getUpdatedBy());  
 preparedStatement.setInt(7, supplier.getSupplierID());  
 return preparedStatement.executeUpdate() > 0;  
 } catch (SQLException e) {  
 e.printStackTrace();  
 return false;  
 }  
 }  
  
  
 public boolean deleteSupplier(int supplierID) {  
 String countProductsSql = "SELECT COUNT(\*) FROM Products WHERE SupplierID = ?";  
 String deletePricingSql = "DELETE FROM Pricing WHERE ProductID IN (SELECT ProductID FROM Products WHERE SupplierID = ?)";  
 String deleteProductsSql = "DELETE FROM Products WHERE SupplierID = ?";  
 String deleteSupplierSql = "DELETE FROM Suppliers WHERE SupplierID = ?";  
 try (Connection conn = MyConnection.*getConnection*()) {  
 conn.setAutoCommit(false);  
 try (PreparedStatement countProductsStmt = conn.prepareStatement(countProductsSql);  
 PreparedStatement deletePricingStmt = conn.prepareStatement(deletePricingSql);  
 PreparedStatement deleteProductsStmt = conn.prepareStatement(deleteProductsSql);  
 PreparedStatement deleteSupplierStmt = conn.prepareStatement(deleteSupplierSql)) {  
  
 // Check if there are any products associated with the supplier ID  
 countProductsStmt.setInt(1, supplierID);  
 ResultSet resultSet = countProductsStmt.executeQuery();  
 resultSet.next();  
 int productCount = resultSet.getInt(1);  
 if (productCount > 0) {  
 // If there are products, throw an exception or handle it accordingly  
 throw new RuntimeException("Cannot delete supplier because it has associated products.");  
 }  
  
 deletePricingStmt.setInt(1, supplierID);  
 int deletedPricingRows = deletePricingStmt.executeUpdate();  
  
 deleteProductsStmt.setInt(1, supplierID);  
 int deletedProductRows = deleteProductsStmt.executeUpdate();  
  
 deleteSupplierStmt.setInt(1, supplierID);  
 boolean success = deleteSupplierStmt.executeUpdate() > 0;  
  
 conn.commit();  
 return success;  
 } catch (SQLException e) {  
 conn.rollback();  
 e.printStackTrace();  
 // You can log the exception or throw a custom exception here  
 throw new RuntimeException("Error deleting supplier: " + e.getMessage());  
 }  
 } catch (SQLException e) {  
 e.printStackTrace();  
 // You can log the exception or throw a custom exception here  
 throw new RuntimeException("Error connecting to the database: " + e.getMessage());  
 }  
 }  
  
  
  
  
}

# IV. CONNECT

## MyConnect

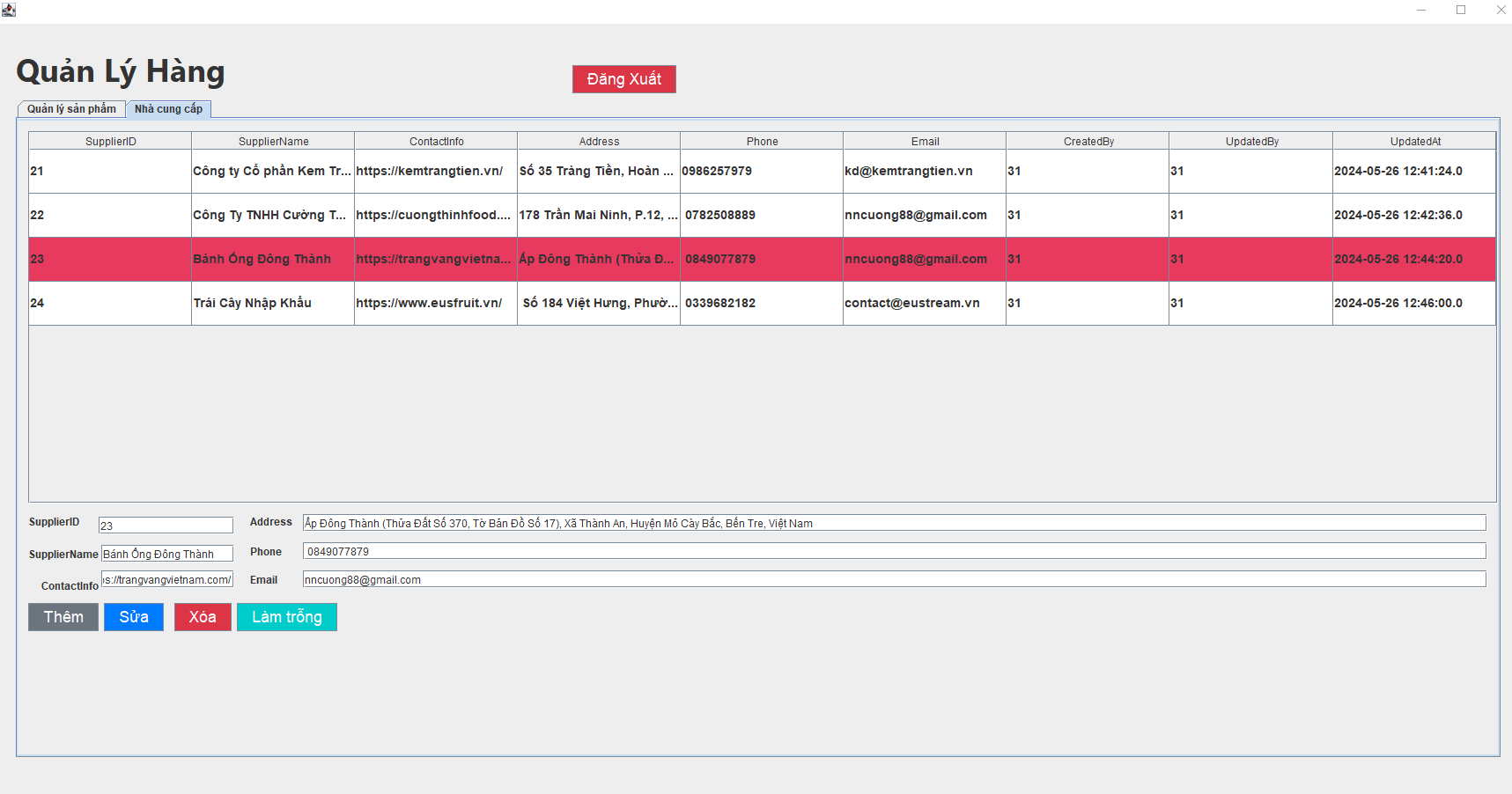
package org.example.connect;  
  
import java.sql.Connection;  
import java.sql.DriverManager;  
import java.sql.SQLException;  
  
public class MyConnection {  
 private static final String *dbURL* = "jdbc:mysql://localhost:3306/mart";  
 private static final String *userName* = "root";  
 private static final String *password* = "";  
  
 public static Connection getConnection() {  
 Connection conn = null;  
 try {  
 Class.*forName*("com.mysql.cj.jdbc.Driver");  
 conn = DriverManager.*getConnection*(*dbURL*, *userName*, *password*);  
 } catch (ClassNotFoundException | SQLException e) {  
 e.printStackTrace();  
 }  
 return conn;  
 }  
}

# V. KẾT QUẢ

## 1. Đăng nhập



## 2. Quản lý nhà cung cấp



## 3. Quản lý sản phẩm

