

**TRƯỜNG ĐẠI HỌC**

**KHOA 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 KẾ TOÁN**

|  |  |
| --- | --- |
| Tên chương trình: QUẢN LÝ BÁN HÀNG CỦA SIÊU THỊ | Mã chương trình:  QLBHCST |
| Tên module: Lập trình kế toán | Mã tài liệu:  LT4 |
| Tên người làm: | Mã người làm: N1 |
| Thời hạn: Ngày bắt đầu: 10/04/2024 | Ngày kết thúc:  14/05/2024 |
| Ngày nộp: 29/05/2024 | |
| Phiên bản/ File: LT4-240526.docx | |

**NGƯỜI LÀM KÝ XÁC NHẬN**

**MỤC LỤC**

I. GIAO DIỆN 3

1. FormDangNhap 3

2. FormKeToan 17

II.THỰC THỂ 39

1. SalaryReport 39

2. SalesReceiptDetail 39

III. CONTOLLER 41

1. SalaryReportController 41

2. SalesReceiptDetailController 49

IV. CONNECT 51

MyConnect 51

V. KẾT QUẢ 52

1. Đăng nhập 52

2. Sắp xếp doanh thu 52

3. Hiển thị tổng doanh thu 53

4. Xóa doanh thu 53

# 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. FormKeToan

/\*  
 \* 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.connect.MyConnection;  
import org.example.controllers.SalaryReportController;  
import org.example.controllers.SalesReceiptController;  
import org.example.controllers.UserController;  
  
import javax.swing.\*;  
import javax.swing.table.DefaultTableModel;  
import javax.swing.table.TableRowSorter;  
import java.awt.event.ActionEvent;  
import java.awt.event.KeyEvent;  
import java.sql.\*;  
import java.time.LocalDate;  
import java.util.ArrayList;  
import java.util.Comparator;  
import java.util.List;  
  
*/\*\*  
 \*  
 \* @author   
 \*/*public class FormKeToan extends javax.swing.JFrame {  
 private DefaultTableModel tableModel;  
 private SalaryReportController salaryReportController;  
 private int loggedInUserID;  
 private int employeeID; // Thêm biến để lưu trữ EmployeeID  
 */\*\*  
 \* Creates new form FormKeToan  
 \*/* public FormKeToan(int loggedInUserID) {  
 this.loggedInUserID = loggedInUserID;  
 UserController userController = new UserController();  
 employeeID = userController.getEmployeeIDByUserID(loggedInUserID);  
 initComponents();  
 showBaoCaoLuong();  
 showBaoCaoDoanhThu();  
 addEscapeKeyBinding();  
 System.*out*.println("Đang Quản Lý kế toán 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*(  
 FormKeToan.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*) {  
 // Xử lý đăng xuất  
 System.*out*.println("Bộ phận kế toán đã đăng xuất");  
 btnDangXuatActionPerformed(null);  
 } else if (response == JOptionPane.*NO\_OPTION*) {  
 System.*out*.println("Đã thoát chương trình");  
 // 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 showBaoCaoLuong() {  
 salaryReportController = new SalaryReportController();  
 tableModel = salaryReportController.getAllSalaryReport();  
// TableBaoCaoLuong.setModel(tableModel); // Sửa đổi ở đây  
 }  
  
 private void showBaoCaoDoanhThu() {  
 salaryReportController = new SalaryReportController();  
 tableModel = salaryReportController.getAllDoanhThu();  
 TableBaoCaoDoanhThu.setModel(tableModel);  
 }  
  
 */\*\*  
 \* 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();  
 Main = new javax.swing.JTabbedPane();  
 jPanel3 = new javax.swing.JPanel();  
 jScrollPane2 = new javax.swing.JScrollPane();  
 TableBaoCaoDoanhThu = new javax.swing.JTable();  
 btnSapXepTheoSoLuongDaBan = new javax.swing.JButton();  
 btnSapXepTheoGia = new javax.swing.JButton();  
 btnSapXepTheoTenSanPham = new javax.swing.JButton();  
 ComboBoxTongDoanhThu = new javax.swing.JComboBox<>();  
 lbDoanhThu = new javax.swing.JLabel();  
 btnXoa = new javax.swing.JButton();  
 btnDangXuat = new javax.swing.JButton();  
  
 setDefaultCloseOperation(javax.swing.WindowConstants.*EXIT\_ON\_CLOSE*);  
  
 jLabel1.setFont(new java.awt.Font("Arial", 1, 36)); // NOI18N  
 jLabel1.setText("Kế Toán");  
  
 TableBaoCaoDoanhThu.setFont(new java.awt.Font("Arial", 0, 18)); // NOI18N  
 TableBaoCaoDoanhThu.setModel(new javax.swing.table.DefaultTableModel(  
 new Object [][] {  
 {},  
 {},  
 {},  
 {}  
 },  
 new String [] {  
  
 }  
 ));  
 TableBaoCaoDoanhThu.setRowHeight(40);  
 TableBaoCaoDoanhThu.setSelectionBackground(new java.awt.Color(232, 57, 95));  
 TableBaoCaoDoanhThu.setShowHorizontalLines(true);  
 TableBaoCaoDoanhThu.addMouseListener(new java.awt.event.MouseAdapter() {  
 public void mouseClicked(java.awt.event.MouseEvent evt) {  
 TableBaoCaoDoanhThuMouseClicked(evt);  
 }  
 });  
 jScrollPane2.setViewportView(TableBaoCaoDoanhThu);  
  
 btnSapXepTheoSoLuongDaBan.setText("Sắp Xếp Theo Số Lượng Đã Bán");  
 btnSapXepTheoSoLuongDaBan.addActionListener(new java.awt.event.ActionListener() {  
 public void actionPerformed(java.awt.event.ActionEvent evt) {  
 btnSapXepTheoSoLuongDaBanActionPerformed(evt);  
 }  
 });  
  
 btnSapXepTheoGia.setText("Sắp Xếp Theo Giá");  
 btnSapXepTheoGia.addActionListener(new java.awt.event.ActionListener() {  
 public void actionPerformed(java.awt.event.ActionEvent evt) {  
 btnSapXepTheoGiaActionPerformed(evt);  
 }  
 });  
  
 btnSapXepTheoTenSanPham.setText("Sắp Xếp Theo Tên Sản phẩm");  
 btnSapXepTheoTenSanPham.addActionListener(new java.awt.event.ActionListener() {  
 public void actionPerformed(java.awt.event.ActionEvent evt) {  
 btnSapXepTheoTenSanPhamActionPerformed(evt);  
 }  
 });  
  
 ComboBoxTongDoanhThu.setModel(new javax.swing.DefaultComboBoxModel<>(new String[] { "Tổng Doanh Thu Siêu Thị", "Tổng Doanh Thu Theo Ngày Hiện Tại", "Tổng Doanh Thu Theo Tháng Hiện Tại", "Tổng Doanh Thu Theo Năm Hiện Tại" }));  
 ComboBoxTongDoanhThu.addActionListener(new java.awt.event.ActionListener() {  
 public void actionPerformed(java.awt.event.ActionEvent evt) {  
 ComboBoxTongDoanhThuActionPerformed(evt);  
 }  
 });  
  
 lbDoanhThu.setText("DoanhThu");  
  
 btnXoa.setText("Xóa");  
 btnXoa.addActionListener(new java.awt.event.ActionListener() {  
 public void actionPerformed(java.awt.event.ActionEvent evt) {  
 btnXoaActionPerformed(evt);  
 }  
 });  
  
 javax.swing.GroupLayout jPanel3Layout = new javax.swing.GroupLayout(jPanel3);  
 jPanel3.setLayout(jPanel3Layout);  
 jPanel3Layout.setHorizontalGroup(  
 jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.*LEADING*)  
 .addGroup(jPanel3Layout.createSequentialGroup()  
 .addContainerGap()  
 .addGroup(jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.*LEADING*)  
 .addComponent(jScrollPane2, javax.swing.GroupLayout.Alignment.*TRAILING*)  
 .addGroup(jPanel3Layout.createSequentialGroup()  
 .addGroup(jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.*TRAILING*, false)  
 .addComponent(btnXoa, javax.swing.GroupLayout.Alignment.*LEADING*, javax.swing.GroupLayout.*DEFAULT\_SIZE*, javax.swing.GroupLayout.*DEFAULT\_SIZE*, Short.*MAX\_VALUE*)  
 .addComponent(btnSapXepTheoTenSanPham, javax.swing.GroupLayout.*DEFAULT\_SIZE*, javax.swing.GroupLayout.*DEFAULT\_SIZE*, Short.*MAX\_VALUE*)  
 .addComponent(btnSapXepTheoGia, javax.swing.GroupLayout.Alignment.*LEADING*, javax.swing.GroupLayout.*DEFAULT\_SIZE*, javax.swing.GroupLayout.*DEFAULT\_SIZE*, Short.*MAX\_VALUE*)  
 .addComponent(btnSapXepTheoSoLuongDaBan, javax.swing.GroupLayout.Alignment.*LEADING*, javax.swing.GroupLayout.*DEFAULT\_SIZE*, javax.swing.GroupLayout.*DEFAULT\_SIZE*, Short.*MAX\_VALUE*))  
 .addGap(121, 121, 121)  
 .addGroup(jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.*LEADING*)  
 .addGroup(jPanel3Layout.createSequentialGroup()  
 .addComponent(ComboBoxTongDoanhThu, javax.swing.GroupLayout.*PREFERRED\_SIZE*, javax.swing.GroupLayout.*DEFAULT\_SIZE*, javax.swing.GroupLayout.*PREFERRED\_SIZE*)  
 .addGap(0, 0, Short.*MAX\_VALUE*))  
 .addComponent(lbDoanhThu, javax.swing.GroupLayout.*DEFAULT\_SIZE*, 561, Short.*MAX\_VALUE*))))  
 .addContainerGap())  
 );  
 jPanel3Layout.setVerticalGroup(  
 jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.*LEADING*)  
 .addGroup(jPanel3Layout.createSequentialGroup()  
 .addContainerGap()  
 .addComponent(jScrollPane2, javax.swing.GroupLayout.*PREFERRED\_SIZE*, 402, javax.swing.GroupLayout.*PREFERRED\_SIZE*)  
 .addGap(18, 18, 18)  
 .addGroup(jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.*BASELINE*)  
 .addComponent(btnSapXepTheoSoLuongDaBan)  
 .addComponent(ComboBoxTongDoanhThu, javax.swing.GroupLayout.*PREFERRED\_SIZE*, javax.swing.GroupLayout.*DEFAULT\_SIZE*, javax.swing.GroupLayout.*PREFERRED\_SIZE*))  
 .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.*RELATED*)  
 .addGroup(jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.*LEADING*)  
 .addGroup(jPanel3Layout.createSequentialGroup()  
 .addComponent(btnSapXepTheoGia)  
 .addGap(18, 18, 18)  
 .addComponent(btnSapXepTheoTenSanPham))  
 .addComponent(lbDoanhThu, javax.swing.GroupLayout.*PREFERRED\_SIZE*, 51, javax.swing.GroupLayout.*PREFERRED\_SIZE*))  
 .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.*UNRELATED*)  
 .addComponent(btnXoa)  
 .addContainerGap(42, Short.*MAX\_VALUE*))  
 );  
  
 Main.addTab("Báo cáo doanh thu", jPanel3);  
  
 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()  
 .addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.*LEADING*)  
 .addGroup(jPanel1Layout.createSequentialGroup()  
 .addContainerGap()  
 .addComponent(jLabel1, javax.swing.GroupLayout.*DEFAULT\_SIZE*, javax.swing.GroupLayout.*DEFAULT\_SIZE*, Short.*MAX\_VALUE*)  
 .addGap(18, 18, 18)  
 .addComponent(btnDangXuat))  
 .addComponent(Main))  
 .addContainerGap())  
 );  
 jPanel1Layout.setVerticalGroup(  
 jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.*LEADING*)  
 .addGroup(jPanel1Layout.createSequentialGroup()  
 .addGap(6, 6, 6)  
 .addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.*TRAILING*)  
 .addComponent(btnDangXuat)  
 .addComponent(jLabel1, javax.swing.GroupLayout.*PREFERRED\_SIZE*, 68, javax.swing.GroupLayout.*PREFERRED\_SIZE*))  
 .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.*RELATED*)  
 .addComponent(Main)  
 .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.*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.*DEFAULT\_SIZE*, javax.swing.GroupLayout.*DEFAULT\_SIZE*, Short.*MAX\_VALUE*)  
 .addContainerGap())  
 );  
  
 pack();  
 }// </editor-fold>//GEN-END:initComponents  
  
 private void btnSapXepTheoSoLuongDaBanActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event\_btnSapXepTheoSoLuongDaBanActionPerformed  
 // *TODO add your handling code here:* // Sắp xếp bảng theo cột "SoldQuantity" (số lượng đã bán)  
 DefaultTableModel model = (DefaultTableModel) TableBaoCaoDoanhThu.getModel();  
 TableRowSorter<DefaultTableModel> sorter = new TableRowSorter<>(model);  
 TableBaoCaoDoanhThu.setRowSorter(sorter);  
 List<RowSorter.SortKey> sortKeys = new ArrayList<>();  
 sortKeys.add(new RowSorter.SortKey(9, SortOrder.*DESCENDING*)); // 9 là chỉ số cột "SoldQuantity"  
 sorter.setSortKeys(sortKeys);  
 sorter.sort();  
 }//GEN-LAST:event\_btnSapXepTheoSoLuongDaBanActionPerformed  
  
 private void btnSapXepTheoGiaActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event\_btnSapXepTheoGiaActionPerformed  
 // Sắp xếp bảng theo cột "TotalAmount" (giá)  
 DefaultTableModel model = (DefaultTableModel) TableBaoCaoDoanhThu.getModel();  
 TableRowSorter<DefaultTableModel> sorter = new TableRowSorter<>(model);  
 TableBaoCaoDoanhThu.setRowSorter(sorter);  
 List<RowSorter.SortKey> sortKeys = new ArrayList<>();  
 sortKeys.add(new RowSorter.SortKey(2, SortOrder.*DESCENDING*)); // 2 là chỉ số cột "TotalAmount"  
 sorter.setSortKeys(sortKeys);  
 sorter.setComparator(2, new Comparator<Double>() {  
 @Override  
 public int compare(Double o1, Double o2) {  
 return o1.compareTo(o2);  
 }  
 });  
 sorter.sort();  
 }//GEN-LAST:event\_btnSapXepTheoGiaActionPerformed  
  
 private void btnSapXepTheoTenSanPhamActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event\_btnSapXepTheoTenSanPhamActionPerformed  
 // *TODO add your handling code here:* DefaultTableModel model = (DefaultTableModel) TableBaoCaoDoanhThu.getModel();  
 TableRowSorter<DefaultTableModel> sorter = new TableRowSorter<>(model);  
 TableBaoCaoDoanhThu.setRowSorter(sorter);  
 List<RowSorter.SortKey> sortKeys = new ArrayList<>();  
 sortKeys.add(new RowSorter.SortKey(8, SortOrder.*ASCENDING*)); // 8 là chỉ số cột "ProductName"  
 sorter.setSortKeys(sortKeys);  
 sorter.sort();  
 }//GEN-LAST:event\_btnSapXepTheoTenSanPhamActionPerformed  
  
 private void ComboBoxTongDoanhThuActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event\_ComboBoxTongDoanhThuActionPerformed  
 // *TODO: Xử lý sự kiện khi lựa chọn từ combobox* String selectedOption = (String) ComboBoxTongDoanhThu.getSelectedItem();  
 switch (selectedOption) {  
 case "Tổng Doanh Thu Siêu Thị":  
 hienThiTongDoanhThuCuaSieuThi();  
 break;  
 case "Tổng Doanh Thu Theo Ngày Hiện Tại":  
 hienThiTongDoanhThuTheoNgayHienTai();  
 break;  
 case "Tổng Doanh Thu Theo Tháng Hiện Tại":  
 hienThiTongDoanhThuTheoThangHienTai();  
 break;  
 case "Tổng Doanh Thu Theo Năm Hiện Tại":  
 hienThiTongDoanhThuTheoNamHienTai();  
 break;  
 default:  
 // Xử lý nếu không có lựa chọn phù hợp  
 break;  
 }  
 }//GEN-LAST:event\_ComboBoxTongDoanhThuActionPerformed  
  
 private void updateTotalRevenueLabel(String message) {  
 lbDoanhThu.setText(message);  
 }  
  
 private void hienThiTongDoanhThuCuaSieuThi() {  
 // Your existing code to fetch total revenue  
 try {  
 // Tạo truy vấn SQL để tính tổng doanh thu  
 String query = "SELECT SUM(TotalAmount) AS TongDoanhThu FROM SalesReceipts";  
  
 // Thực thi truy vấn  
 Statement statement = MyConnection.*getConnection*().createStatement();  
 ResultSet resultSet = statement.executeQuery(query);  
  
 // Kiểm tra nếu có kết quả  
 if (resultSet.next()) {  
 // Lấy tổng doanh thu từ kết quả  
 double tongDoanhThu = resultSet.getDouble("TongDoanhThu");  
 // Update the total revenue label  
 updateTotalRevenueLabel("Tổng doanh thu của siêu thị là: "+tongDoanhThu);  
  
 System.*out*.println("Tổng doanh thu của siêu thị là: "+tongDoanhThu);  
  
 }  
 } catch (SQLException e) {  
 e.printStackTrace();  
 }  
 }  
  
 private void hienThiTongDoanhThuTheoNgayHienTai() {  
 try {  
 // Lấy ngày hiện tại  
 LocalDate currentDate = LocalDate.*now*();  
  
 // Tạo truy vấn SQL để tính tổng doanh thu theo ngày hiện tại  
 String query = "SELECT SUM(TotalAmount) AS TongDoanhThu FROM SalesReceipts WHERE SaleDate = ?";  
  
 // Thực thi truy vấn với tham số là ngày hiện tại  
 try (Connection conn = MyConnection.*getConnection*();  
 PreparedStatement stmt = conn.prepareStatement(query)) {  
 stmt.setDate(1, java.sql.Date.*valueOf*(currentDate));  
 ResultSet resultSet = stmt.executeQuery();  
  
 // Kiểm tra nếu có kết quả  
 if (resultSet.next()) {  
 // Lấy tổng doanh thu từ kết quả  
 double tongDoanhThu = resultSet.getDouble("TongDoanhThu");  
  
 // Tạo bảng để hiển thị tổng doanh thu  
 DefaultTableModel model = new DefaultTableModel(  
 new String[]{"Tổng Doanh Thu"}, 0);  
 model.addRow(new Object[]{tongDoanhThu});  
 System.*out*.println("Tổng doanh thu của ngày hiện tại là: "+tongDoanhThu);  
 updateTotalRevenueLabel("Tổng doanh thu của ngày hiện tại là: "+tongDoanhThu);  
  
 // Hiển thị bảng  
 TableBaoCaoDoanhThu.setModel(tableModel);  
  
 }  
 }  
 } catch (SQLException e) {  
 e.printStackTrace();  
 }  
 }  
  
  
 private void hienThiTongDoanhThuTheoThangHienTai() {  
 try {  
 // Lấy tháng và năm hiện tại  
 LocalDate currentDate = LocalDate.*now*();  
 int currentMonth = currentDate.getMonthValue();  
 int currentYear = currentDate.getYear();  
  
 // Tạo truy vấn SQL để tính tổng doanh thu theo tháng và năm hiện tại  
 String query = "SELECT SUM(TotalAmount) AS TongDoanhThu FROM SalesReceipts WHERE MONTH(SaleDate) = ? AND YEAR(SaleDate) = ?";  
  
 // Thực thi truy vấn với tham số là tháng và năm hiện tại  
 try (Connection conn = MyConnection.*getConnection*();  
 PreparedStatement stmt = conn.prepareStatement(query)) {  
 stmt.setInt(1, currentMonth);  
 stmt.setInt(2, currentYear);  
 ResultSet resultSet = stmt.executeQuery();  
  
 // Kiểm tra nếu có kết quả  
 if (resultSet.next()) {  
 // Lấy tổng doanh thu từ kết quả  
 double tongDoanhThu = resultSet.getDouble("TongDoanhThu");  
  
 // Tạo bảng để hiển thị tổng doanh thu  
 DefaultTableModel model = new DefaultTableModel(  
 new String[]{"Tổng Doanh Thu"}, 0);  
 model.addRow(new Object[]{tongDoanhThu});  
 System.*out*.println("Tổng doanh thu của tháng hiện tại là: "+tongDoanhThu);  
 updateTotalRevenueLabel("Tổng doanh thu của tháng hiện tại là: "+tongDoanhThu);  
 // Hiển thị bảng  
 TableBaoCaoDoanhThu.setModel(tableModel);  
  
 }  
 }  
 } catch (SQLException e) {  
 e.printStackTrace();  
 }  
 }  
  
  
 private void hienThiTongDoanhThuTheoNamHienTai() {  
 try {  
 // Lấy năm hiện tại  
 int currentYear = LocalDate.*now*().getYear();  
  
 // Tạo truy vấn SQL để tính tổng doanh thu theo năm hiện tại  
 String query = "SELECT SUM(TotalAmount) AS TongDoanhThu FROM SalesReceipts WHERE YEAR(SaleDate) = ?";  
  
 // Thực thi truy vấn với tham số năm hiện tại  
 try (Connection conn = MyConnection.*getConnection*();  
 PreparedStatement stmt = conn.prepareStatement(query)) {  
 stmt.setInt(1, currentYear);  
 ResultSet resultSet = stmt.executeQuery();  
  
 // Kiểm tra nếu có kết quả  
 if (resultSet.next()) {  
 // Lấy tổng doanh thu từ kết quả  
 double tongDoanhThu = resultSet.getDouble("TongDoanhThu");  
  
 // Tạo bảng để hiển thị tổng doanh thu  
 DefaultTableModel model = new DefaultTableModel(  
 new String[]{"Tổng Doanh Thu"}, 0);  
 model.addRow(new Object[]{tongDoanhThu});  
 System.*out*.println("Tổng doanh thu của năm hiện tại là: "+tongDoanhThu);  
 // Hiển thị bảng  
 updateTotalRevenueLabel("Tổng doanh thu của năm hiện tại là: "+tongDoanhThu);  
  
 TableBaoCaoDoanhThu.setModel(tableModel);  
 }  
 }  
 } catch (SQLException e) {  
 e.printStackTrace();  
 }  
 }  
  
 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 TableBaoCaoDoanhThuMouseClicked(java.awt.event.MouseEvent evt) {//GEN-FIRST:event\_TableBaoCaoDoanhThuMouseClicked  
 // *TODO add your handling code here:* int selectedRow = TableBaoCaoDoanhThu.getSelectedRow();  
 if (selectedRow != -1) {  
 // Lấy giá trị SalesReceiptID từ hàng được chọn  
 int salesReceiptID = (int) TableBaoCaoDoanhThu.getValueAt(selectedRow, 0);  
  
 // Hiển thị hộp thoại xác nhận  
 int confirm = JOptionPane.*showConfirmDialog*(this, "Are you sure you want to delete this sales report?", "Confirm Delete", JOptionPane.*YES\_NO\_OPTION*);  
  
 if (confirm == JOptionPane.*YES\_OPTION*) {  
 // Gọi phương thức xóa báo cáo doanh thu từ controller  
 SalaryReportController controller = new SalaryReportController();  
 boolean success = controller.deleteSalesReport(salesReceiptID);  
  
 if (success) {  
 // Hiển thị thông báo xóa thành công  
 JOptionPane.*showMessageDialog*(this, "Sales report deleted successfully");  
  
 // Refresh bảng báo cáo doanh thu  
 DefaultTableModel model = controller.getAllDoanhThu();  
 TableBaoCaoDoanhThu.setModel(model);  
 } else {  
 // Hiển thị thông báo lỗi  
 JOptionPane.*showMessageDialog*(this, "Failed to delete sales report", "Error", JOptionPane.*ERROR\_MESSAGE*);  
 }  
 }  
 } else {  
 JOptionPane.*showMessageDialog*(this, "Please select a sales report to delete");  
 }  
 }//GEN-LAST:event\_TableBaoCaoDoanhThuMouseClicked  
  
 private void btnXoaActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event\_btnXoaActionPerformed  
 // *TODO add your handling code here:* // Lấy chỉ mục hàng đã chọn trong bảng  
 int selectedRow = TableBaoCaoDoanhThu.getSelectedRow();  
  
 // Kiểm tra xem hàng đã chọn có hợp lệ không  
 if (selectedRow != -1) {  
 // Lấy giá trị của cột SalesReceiptID từ hàng đã chọn  
 int salesReceiptID = (int) TableBaoCaoDoanhThu.getValueAt(selectedRow, 0);  
  
 // Xác nhận xóa  
 int confirm = JOptionPane.*showConfirmDialog*(this, "Are you sure you want to delete this sales report?", "Confirm Delete", JOptionPane.*YES\_NO\_OPTION*);  
  
 if (confirm == JOptionPane.*YES\_OPTION*) {  
 // Gọi phương thức xóa báo cáo doanh thu từ controller  
 boolean success = salaryReportController.deleteSalesReport(salesReceiptID);  
  
 if (success) {  
 // Hiển thị thông báo thành công  
 JOptionPane.*showMessageDialog*(this, "Sales report deleted successfully");  
  
 // Cập nhật bảng báo cáo doanh thu  
 showBaoCaoDoanhThu();  
 } else {  
 // Hiển thị thông báo lỗi nếu xóa không thành công  
 JOptionPane.*showMessageDialog*(this, "Failed to delete sales report", "Error", JOptionPane.*ERROR\_MESSAGE*);  
 }  
 }  
 } else {  
 // Hiển thị thông báo yêu cầu chọn hàng để xóa  
 JOptionPane.*showMessageDialog*(this, "Please select a sales report to delete");  
 }  
 }//GEN-LAST:event\_btnXoaActionPerformed  
  
 */\*\*  
 \* @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*(FormKeToan.class.getName()).log(java.util.logging.Level.*SEVERE*, null, ex);  
 } catch (InstantiationException ex) {  
 java.util.logging.Logger.*getLogger*(FormKeToan.class.getName()).log(java.util.logging.Level.*SEVERE*, null, ex);  
 } catch (IllegalAccessException ex) {  
 java.util.logging.Logger.*getLogger*(FormKeToan.class.getName()).log(java.util.logging.Level.*SEVERE*, null, ex);  
 } catch (javax.swing.UnsupportedLookAndFeelException ex) {  
 java.util.logging.Logger.*getLogger*(FormKeToan.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 FormKeToan(loggedInUserID[0]).setVisible(true);  
 System.*out*.println(loggedInUserID[0]);  
 }  
 });  
 }  
  
 // Variables declaration - do not modify//GEN-BEGIN:variables  
 private javax.swing.JComboBox<String> ComboBoxTongDoanhThu;  
 private javax.swing.JTabbedPane Main;  
 private javax.swing.JTable TableBaoCaoDoanhThu;  
 private javax.swing.JButton btnDangXuat;  
 private javax.swing.JButton btnSapXepTheoGia;  
 private javax.swing.JButton btnSapXepTheoSoLuongDaBan;  
 private javax.swing.JButton btnSapXepTheoTenSanPham;  
 private javax.swing.JButton btnXoa;  
 private javax.swing.JLabel jLabel1;  
 private javax.swing.JPanel jPanel1;  
 private javax.swing.JPanel jPanel3;  
 private javax.swing.JScrollPane jScrollPane2;  
 private javax.swing.JLabel lbDoanhThu;  
 // End of variables declaration//GEN-END:variables  
}

# II.THỰC THỂ

## 1. SalaryReport

package org.example.models;  
  
  
import lombok.Data;  
  
import java.sql.Date;  
@Data  
public class SalaryReport {  
 private int reportID;  
 private int employeeID;  
 private Date reportDate;  
 private String period;  
 private double totalSalary;  
 private int createdBy;  
 private int updatedBy;  
 private Date updatedAt;  
  
 // Constructors, getters, and setters  
}

## 2. SalesReceiptDetail

package org.example.models;  
  
import lombok.Data;  
  
import java.util.Date;  
@Data  
public class SalesReceiptDetail {  
 private int detailID;  
 private int salesReceiptID;  
 private int productID;  
 private int quantity;  
 private double unitPrice;  
 private double totalPrice;  
 private int createdBy;  
 private int updatedBy;  
 private Date updatedAt;  
  
 // Constructors, getters, and setters  
}

# III. CONTOLLER

## 1. SalaryReportController

package org.example.controllers;  
  
import org.example.connect.MyConnection;  
import org.example.models.SalaryReport;  
  
import javax.swing.table.DefaultTableModel;  
import java.sql.\*;  
import java.sql.Date;  
import java.util.\*;  
  
public class SalaryReportController {  
 private Connection connection;  
  
  
 public SalaryReportController() {  
 this.connection = MyConnection.*getConnection*();  
 }  
  
 public int getTotalQuantitySoldByDate(Date startDate, Date endDate) {  
 String query = "SELECT SUM(Quantity) AS TotalQuantity FROM SalesReceiptDetails INNER JOIN SalesReceipts ON SalesReceiptDetails.SalesReceiptID = SalesReceipts.SalesReceiptID WHERE SaleDate BETWEEN ? AND ?";  
 try (Connection conn = MyConnection.*getConnection*();  
 PreparedStatement stmt = conn.prepareStatement(query)) {  
 stmt.setDate(1, new java.sql.Date(startDate.getTime()));  
 stmt.setDate(2, new java.sql.Date(endDate.getTime()));  
 try (ResultSet rs = stmt.executeQuery()) {  
 if (rs.next()) {  
 return rs.getInt("TotalQuantity");  
 }  
 }  
 } catch (SQLException e) {  
 e.printStackTrace();  
 }  
 return 0;  
 }  
 public double getTotalRevenueByDate(Date startDate, Date endDate) {  
 String query = "SELECT SUM(TotalAmount) AS TotalRevenue FROM SalesReceipts WHERE SaleDate BETWEEN ? AND ?";  
 try (Connection conn = MyConnection.*getConnection*();  
 PreparedStatement stmt = conn.prepareStatement(query)) {  
 stmt.setDate(1, new java.sql.Date(startDate.getTime()));  
 stmt.setDate(2, new java.sql.Date(endDate.getTime()));  
 try (ResultSet rs = stmt.executeQuery()) {  
 if (rs.next()) {  
 return rs.getDouble("TotalRevenue");  
 }  
 }  
 } catch (SQLException e) {  
 e.printStackTrace();  
 }  
 return 0.0;  
 }  
  
  
 public Map<String, Double> getRevenueByProduct() {  
 Map<String, Double> revenueMap = new HashMap<>();  
 String query = "SELECT ProductID, SUM(TotalPrice) AS TotalRevenue FROM SalesReceiptDetails GROUP BY ProductID";  
 try (Connection conn = MyConnection.*getConnection*();  
 PreparedStatement stmt = conn.prepareStatement(query);  
 ResultSet rs = stmt.executeQuery()) {  
 while (rs.next()) {  
 int productID = rs.getInt("ProductID");  
 double totalRevenue = rs.getDouble("TotalRevenue");  
 revenueMap.put("Product ID: " + productID, totalRevenue);  
 }  
 } catch (SQLException e) {  
 e.printStackTrace();  
 }  
 return revenueMap;  
 }  
  
  
 public DefaultTableModel getAllSalaryReport() {  
 String query = "SELECT sr.ReportID, sr.EmployeeID, sr.ReportDate, sr.Period, sr.TotalSalary, e.FirstName, e.LastName, COUNT(a.AttendanceID) AS AttendanceCount " +  
 "FROM SalaryReports sr " +  
 "INNER JOIN Employees e ON sr.EmployeeID = e.EmployeeID " +  
 "INNER JOIN Attendance a ON sr.EmployeeID = a.EmployeeID " +  
 "GROUP BY sr.ReportID, sr.EmployeeID, sr.ReportDate, sr.Period, sr.TotalSalary, e.FirstName, e.LastName";  
  
 try (Statement statement = connection.createStatement();  
 ResultSet resultSet = statement.executeQuery(query)) {  
  
 DefaultTableModel model = new DefaultTableModel(  
 new String[]{"ReportID", "EmployeeID", "ReportDate", "Period", "TotalSalary", "FirstName", "LastName", "AttendanceCount"}, 0);  
  
 while (resultSet.next()) {  
 int reportID = resultSet.getInt("ReportID");  
 int employeeID = resultSet.getInt("EmployeeID");  
 Date reportDate = resultSet.getDate("ReportDate");  
 String period = resultSet.getString("Period");  
 double totalSalary = resultSet.getDouble("TotalSalary");  
 String firstName = resultSet.getString("FirstName");  
 String lastName = resultSet.getString("LastName");  
 int attendanceCount = resultSet.getInt("AttendanceCount");  
  
 model.addRow(new Object[]{reportID, employeeID, reportDate, period, totalSalary, firstName, lastName, attendanceCount});  
 }  
  
 return model;  
 } catch (SQLException e) {  
 e.printStackTrace();  
 return null;  
 }  
 }  
  
  
 // báo cáo doanh thu  
  
 public DefaultTableModel getAllDoanhThu() {  
 String query = "SELECT sr.SalesReceiptID, sr.SaleDate, sr.TotalAmount, sr.CustomerName, sr.PaymentMethod, e.FirstName, e.LastName, srd.ProductID, p.ProductName, SUM(srd.Quantity) AS SoldQuantity " +  
 "FROM SalesReceipts sr " +  
 "INNER JOIN Employees e ON sr.EmployeeID = e.EmployeeID " +  
 "INNER JOIN SalesReceiptDetails srd ON sr.SalesReceiptID = srd.SalesReceiptID " +  
 "INNER JOIN Products p ON srd.ProductID = p.ProductID " +  
 "GROUP BY sr.SalesReceiptID, sr.SaleDate, sr.TotalAmount, sr.CustomerName, sr.PaymentMethod, e.FirstName, e.LastName, srd.ProductID, p.ProductName";  
  
 try (Statement statement = connection.createStatement();  
 ResultSet resultSet = statement.executeQuery(query)) {  
  
 DefaultTableModel model = new DefaultTableModel(  
 new String[]{"SalesReceiptID", "SaleDate", "TotalAmount", "CustomerName", "PaymentMethod", "FirstName", "LastName", "ProductID", "ProductName", "SoldQuantity"}, 0);  
  
 while (resultSet.next()) {  
 int salesReceiptID = resultSet.getInt("SalesReceiptID");  
 Date saleDate = resultSet.getDate("SaleDate");  
 double totalAmount = resultSet.getDouble("TotalAmount");  
 String customerName = resultSet.getString("CustomerName");  
 String paymentMethod = resultSet.getString("PaymentMethod");  
 String firstName = resultSet.getString("FirstName");  
 String lastName = resultSet.getString("LastName");  
 int productID = resultSet.getInt("ProductID");  
 String productName = resultSet.getString("ProductName");  
 int soldQuantity = resultSet.getInt("SoldQuantity");  
  
 model.addRow(new Object[]{salesReceiptID, saleDate, totalAmount, customerName, paymentMethod, firstName, lastName, productID, productName, soldQuantity});  
 }  
  
 return model;  
 } catch (SQLException e) {  
 e.printStackTrace();  
 return null;  
 }  
 }  
  
  
  
  
  
  
 // Hàm để lấy báo cáo doanh thu theo ngày  
 public DefaultTableModel getDailySalesReport(Date date) {  
 String query = "SELECT SaleDate, SUM(TotalAmount) AS TotalSales " +  
 "FROM SalesReceipts " +  
 "WHERE SaleDate = '" + date + "' " +  
 "GROUP BY SaleDate";  
  
 return executeQueryAndGenerateTableModel(query);  
 }  
  
 // Hàm để lấy báo cáo doanh thu theo tháng  
 public DefaultTableModel getMonthlySalesReport(int month, int year) {  
 String query = "SELECT YEAR(SaleDate) AS SaleYear, MONTH(SaleDate) AS SaleMonth, SUM(TotalAmount) AS TotalSales " +  
 "FROM SalesReceipts " +  
 "WHERE MONTH(SaleDate) = " + month + " AND YEAR(SaleDate) = " + year + " " +  
 "GROUP BY YEAR(SaleDate), MONTH(SaleDate)";  
  
 return executeQueryAndGenerateTableModel(query);  
 }  
  
 // Hàm để lấy báo cáo doanh thu theo năm  
 public DefaultTableModel getYearlySalesReport(int year) {  
 String query = "SELECT YEAR(SaleDate) AS SaleYear, SUM(TotalAmount) AS TotalSales " +  
 "FROM SalesReceipts " +  
 "WHERE YEAR(SaleDate) = " + year + " " +  
 "GROUP BY YEAR(SaleDate)";  
  
 return executeQueryAndGenerateTableModel(query);  
 }  
  
 // Hàm thực hiện truy vấn và tạo DefaultTableModel từ kết quả truy vấn  
 private DefaultTableModel executeQueryAndGenerateTableModel(String query) {  
 try (Statement statement = connection.createStatement();  
 ResultSet resultSet = statement.executeQuery(query)) {  
  
 DefaultTableModel model = new DefaultTableModel(  
 new String[]{"SaleDate", "TotalSales"}, 0);  
  
 while (resultSet.next()) {  
 Date saleDate = resultSet.getDate("SaleDate");  
 double totalSales = resultSet.getDouble("TotalSales");  
  
 model.addRow(new Object[]{saleDate, totalSales});  
 }  
  
 return model;  
 } catch (SQLException e) {  
 e.printStackTrace();  
 return null;  
 }  
 }  
  
 public boolean deleteSalesReport(int salesReceiptID) {  
 String deleteDetailsQuery = "DELETE FROM SalesReceiptDetails WHERE SalesReceiptID = ?";  
 String deleteReportQuery = "DELETE FROM SalesReceipts WHERE SalesReceiptID = ?";  
  
 try (Connection conn = MyConnection.*getConnection*();  
 PreparedStatement deleteDetailsStmt = conn.prepareStatement(deleteDetailsQuery);  
 PreparedStatement deleteReportStmt = conn.prepareStatement(deleteReportQuery)) {  
  
 // Xóa tất cả các chi tiết báo cáo doanh thu liên quan  
 deleteDetailsStmt.setInt(1, salesReceiptID);  
 deleteDetailsStmt.executeUpdate();  
  
 // Sau đó xóa báo cáo doanh thu  
 deleteReportStmt.setInt(1, salesReceiptID);  
 int rowsAffected = deleteReportStmt.executeUpdate();  
  
 return rowsAffected > 0;  
 } catch (SQLException e) {  
 e.printStackTrace();  
 return false;  
 }  
 }  
  
  
  
}

## 2. SalesReceiptDetailController

package org.example.controllers;  
  
import org.example.connect.MyConnection;  
import org.example.models.SalesReceiptDetail;  
  
import javax.swing.table.DefaultTableModel;  
import java.sql.Connection;  
import java.sql.PreparedStatement;  
import java.sql.ResultSet;  
import java.util.List;  
import java.util.Optional;  
  
public class SalesReceiptDetailController {  
  
 private Connection connection;  
  
 public SalesReceiptDetailController() {  
 // Initialize database connection  
 this.connection = MyConnection.getConnection();  
 }  
  
  
 public DefaultTableModel getAllSalesReceiptDetail() {  
 String[] columnNames = {"DetailID", "SalesReceiptID", "ProductName", "Quantity", "UnitPrice", "TotalPrice"};  
 DefaultTableModel model = new DefaultTableModel(columnNames, 0);  
  
 String sql = "SELECT SRD.DetailID, SR.SalesReceiptID, P.ProductName, SRD.Quantity, SRD.UnitPrice, SRD.TotalPrice " +  
 "FROM SalesReceiptDetails SRD " +  
 "INNER JOIN SalesReceipts SR ON SRD.SalesReceiptID = SR.SalesReceiptID " +  
 "INNER JOIN Products P ON SRD.ProductID = P.ProductID";  
  
 try (Connection conn = MyConnection.getConnection();  
 PreparedStatement preparedStatement = conn.prepareStatement(sql);  
 ResultSet resultSet = preparedStatement.executeQuery()) {  
  
 while (resultSet.next()) {  
 Object[] rowData = {  
 resultSet.getInt("DetailID"),  
 resultSet.getInt("SalesReceiptID"),  
 resultSet.getString("ProductName"),  
 resultSet.getInt("Quantity"),  
 resultSet.getDouble("UnitPrice"),  
 resultSet.getDouble("TotalPrice")  
 };  
 model.addRow(rowData);  
 }  
 } catch (Exception e) {  
 e.printStackTrace();  
 }  
  
 return model;  
 }  
  
}

# IV. CONNECT

## MyConnect

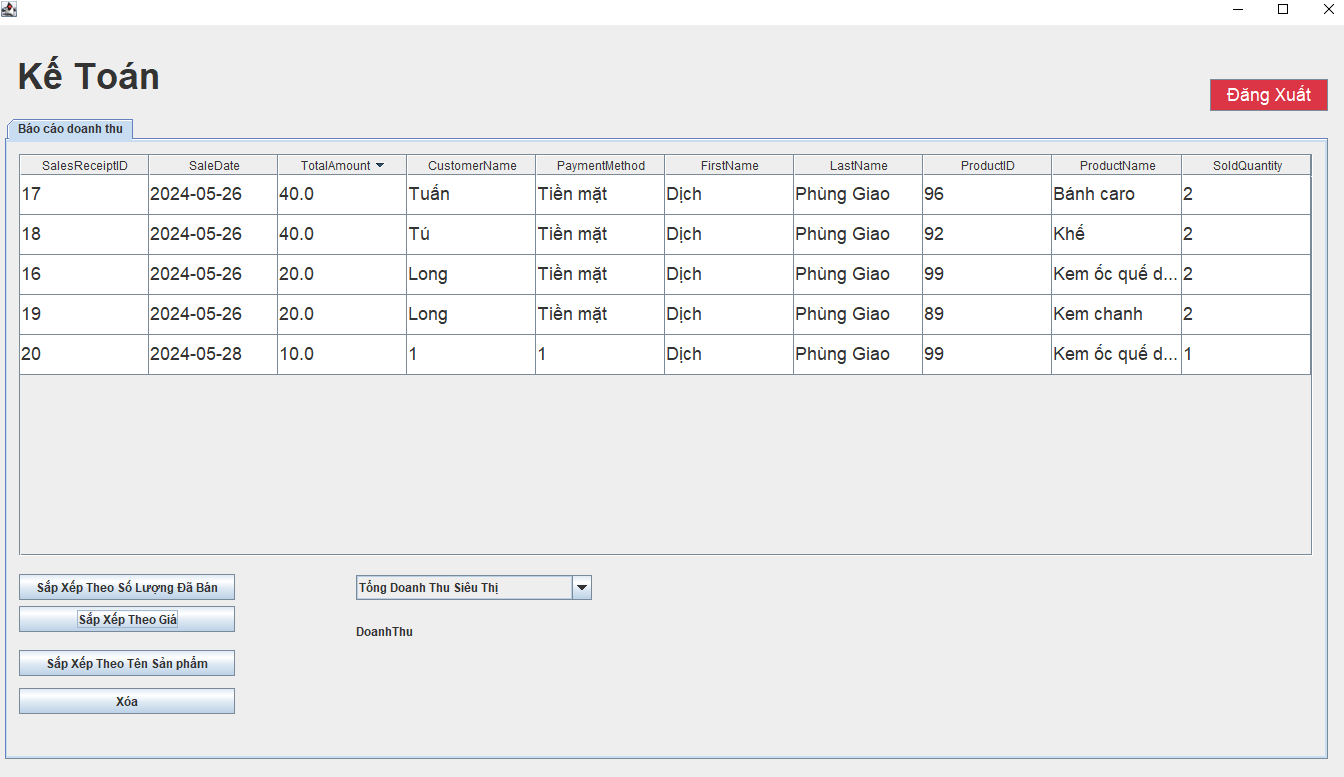
package org.example.connect;  
  
import java.sql.Connection;  
import java.sql.DriverManager;  
import java.sql.SQLException;  
  
public class MyConnection {  
 private static final String *dbURL* = "jdbc:mysql://localhost:3306/mart";  
 private static final String *userName* = "root";  
 private static 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

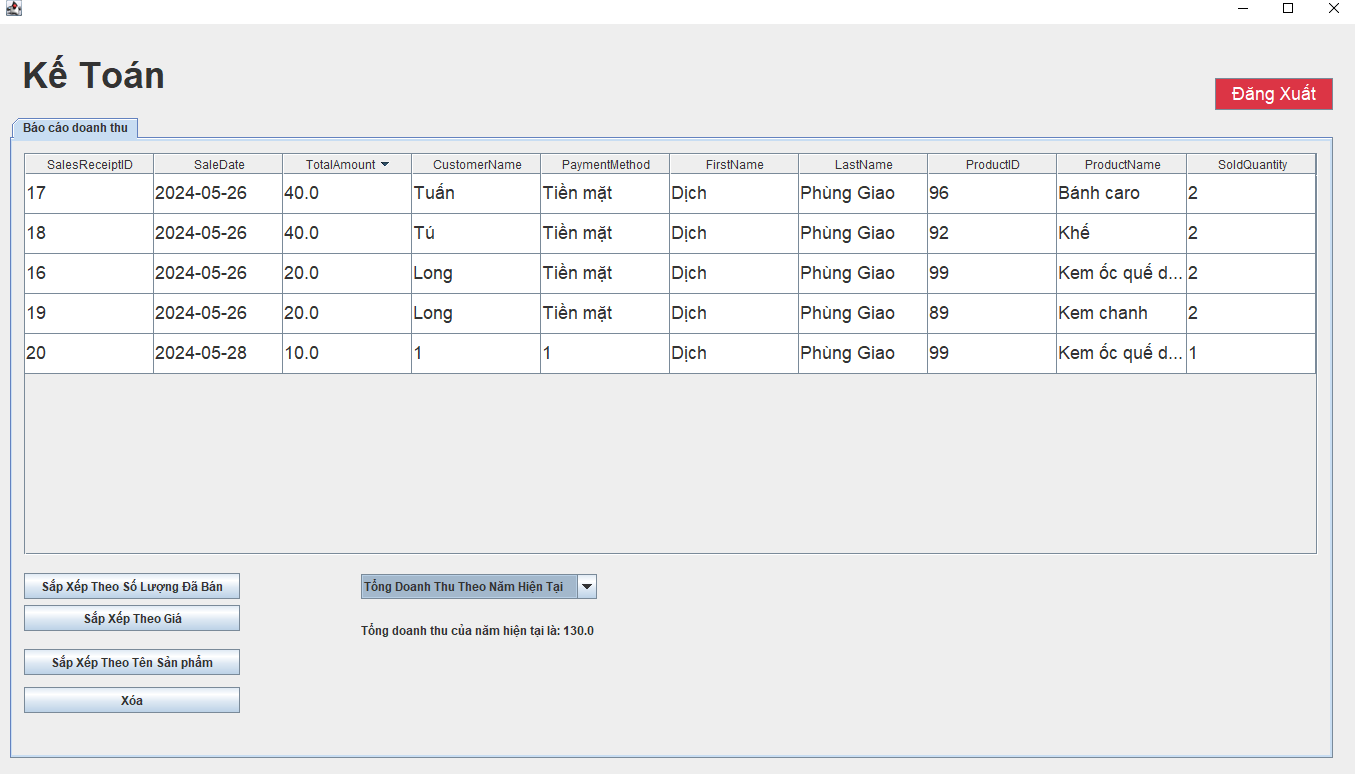
## 1. Đăng nhập



## 2. Sắp xếp doanh thu



## 3. Hiển thị tổng doanh thu



## 4. Xóa doanh thu

