

ỦY BAN NHÂN DÂN THÀNH PHỐ HỒ CHÍ MINH

TRƯỜNG ĐẠI HỌC SÀI GÒN
KHOA CÔNG NGHỆ THÔNG TIN



BÁO CÁO PROJECT 2

**MÔN HỌC: XÂY DỰNG PHẦN MỀM THEO
MÔ HÌNH PHÂN LỚP**

TÊN ĐỀ TÀI: PHẦN MỀM QUẢN LÝ BÁN HÀNG

Người thực hiện báo cáo

GIẢNG VIÊN ĐÁNH GIÁ: ThS.

TP. HCM, tháng 11/2022

MỤC LỤC

Chương I: BẢNG PHÂN CÔNG NHIỆM VỤ	1
Chương II: GIỚI THIỆU ĐỀ TÀI.....	2
1. Giới thiệu đề tài:	2
Chương III: CÁC CHỨC NĂNG QUẢN LÝ QUAN TRỌNG	3
Chức năng quản lý khách hàng	3
Sơ đồ class.....	3
1.1. Xử lý 1: Hiển thị danh sách khách hàng.....	3
1.2. Xử lý 2: Thêm khách hàng	4
1.3. Xử lý 3: Cập nhật khách hàng	6
1.4. Xử lý 4: Xoá khách hàng	8
1.5. Xử lý 5: Tìm kiếm khách hàng	9
2. Chức năng quản lý sản phẩm.....	11
2.1. Sơ đồ class	11
2.2. Xử lý 1: Hiển thị danh sách sản phẩm.....	11
2.3. Xử lý 2: Thêm Sản Phẩm	13
2.4. Xử lý 3: Cập nhật sản phẩm	14
2.5. Xử lý 4: Xóa sản phẩm	16
2.6. Xử lý 5: Tìm kiếm sản phẩm	17
3. Chức năng quản lý hóa đơn	19
3.1. Xử lý 1: Tạo hóa đơn bán hàng	19
3.2. Xử lý 2: Hiển thị danh sách hóa đơn	22
3.3. Xử lý 3: Xem chi tiết hóa đơn	23
3.4. Xử lý 4: Tìm kiếm hóa đơn	25
3.5. Xử lý 5: Xóa hóa đơn	26
4. Chức năng quản lý thống kê.....	28
4.1. Sơ đồ class	28
4.2. Xử lý 1: Hiển thị số lượng sản phẩm.....	28
4.3. Xử lý 2: Hiển thị số lượng Loại.....	30
4.4. Xử lý 3: Hiển thị số lượng đơn hàng	31
4.5. Xử lý 4: Hiển thị tổng số doanh thu	32
4.6. Xử lý 5: Hiển thị biểu đồ doanh thu	34
5. Chức năng quản lý loại	36
5.1. Sơ đồ class	36
5.2. Xử lý 1: Hiển thị danh sách loại	36
5.3. Xử lý 2: Thêm Loại	38

5.4. Xử lý 3: Sửa Loại	40
5.5. Xử lý 4: Xoá Loại.....	41
5.6. Xử lý 5: Tìm kiếm loại	43

**Chương IV: SOURCE CODE KẾT NỐI CSDL, HƯỚNG DẪN CÀI ĐẶT
CHƯƠNG TRÌNH VÀ LINK CHÚA SOURCE CODE ĐỒ ÁN 45**

Chương I: BẢNG PHÂN CÔNG NHIỆM VỤ

Tên người thực hiện	Công việc thực hiện
Nguyễn Đức Minh Trung	Thực hiện tạo vẽ sơ đồ tuần tự Xây dựng chức năng quản lý sản phẩm, loại sản phẩm
Hồ Tấn Thuận	Thực hiện tạo vẽ sơ đồ tuần tự Xây dựng chức năng quản lý hoá đơn
Trần Kim Phú	Thực hiện tạo vẽ sơ đồ tuần tự Xây dựng chức năng quản lý khách hàng
Minh Hiếu Calan Tog	Thực hiện tạo vẽ sơ đồ tuần tự Xây dựng chức năng quản lý loại, thống kê
Võ Hoàng Quỳnh Như	Thực hiện tạo vẽ sơ đồ tuần tự Xây dựng chức năng quản lý bán hàng

Chương II: GIỚI THIỆU ĐỀ TÀI

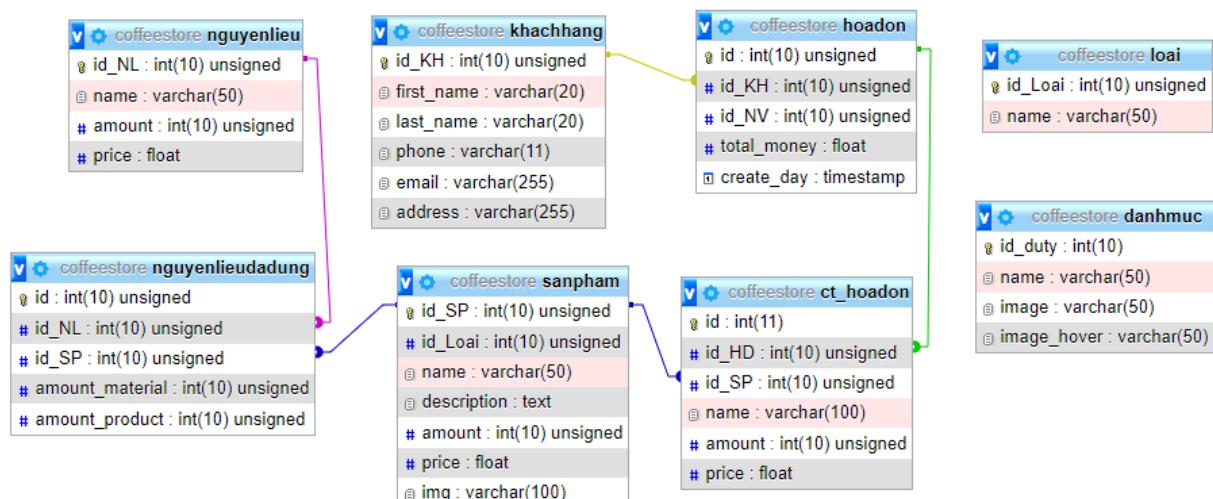
1. Giới thiệu đề tài:

Tên đề tài: Phần mềm quản lý bán café.

Mô tả: Ứng dụng phần mềm vào việc quản lí bán café ở các quán cafe là một nhu cầu tất yếu nhằm nâng cao hiệu quả quản lý tạo sự hài lòng cho khách hàng khi mua sản phẩm nhanh chóng tiện lợi. Do vấn đề đặt ra là khi bán hàng số lượng khách quá đông, số lượng các sản phẩm quá nhiều, nhu cầu nhập xuất và quản lí nguyên vật liệu nếu bằng sổ sách sẽ tốn nhiều công sức và không đảm bảo độ chính xác. Việc thống kê doanh thu bán hàng gấp nhiều khó khăn do lượng sổ sách quá lớn. Thế nên việc dùng phần mềm trong quản lí sẽ giúp đỡ rất nhiều cho người quản lý và nhân viên chỉ bằng vài thao tác xử lý trên hệ thống sẽ kiểm soát được thực trạng kinh doanh của quán.

Chính vì những lí do đó, nhóm chúng em đã triển khai một phần mềm quản hỗ trợ quản lí bán cửa hàng bán café.

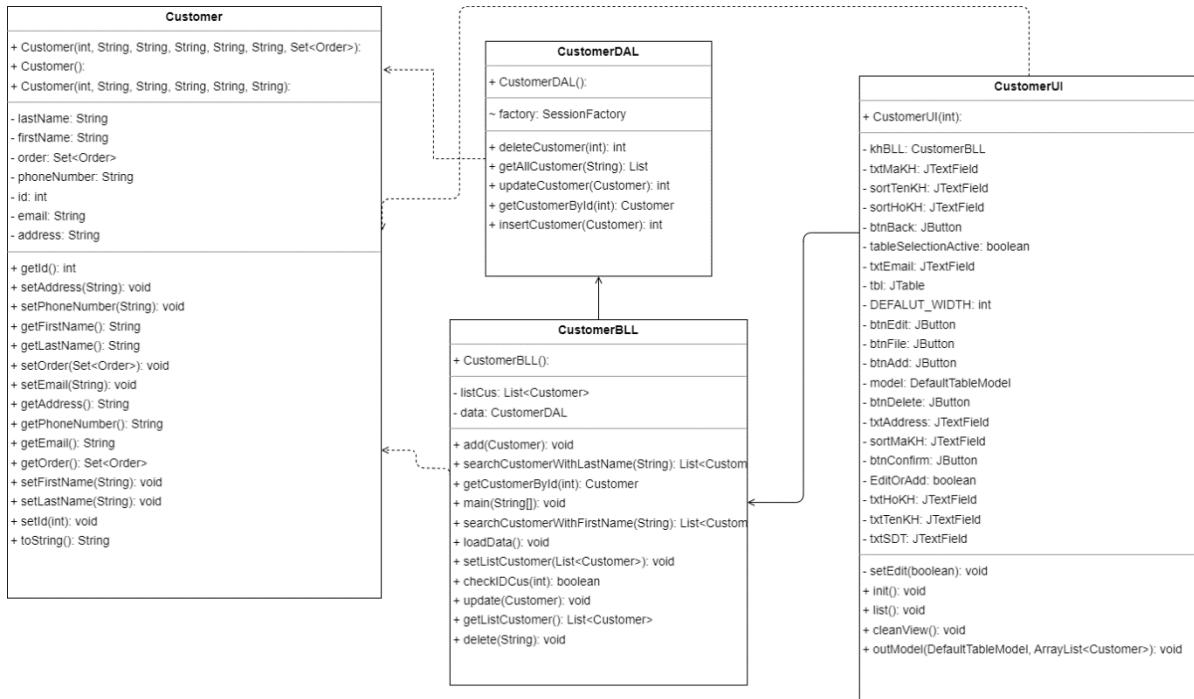
2. Mô hình CSDL mức cài đặt:



Chương III: CÁC CHỨC NĂNG QUẢN LÝ QUAN TRỌNG

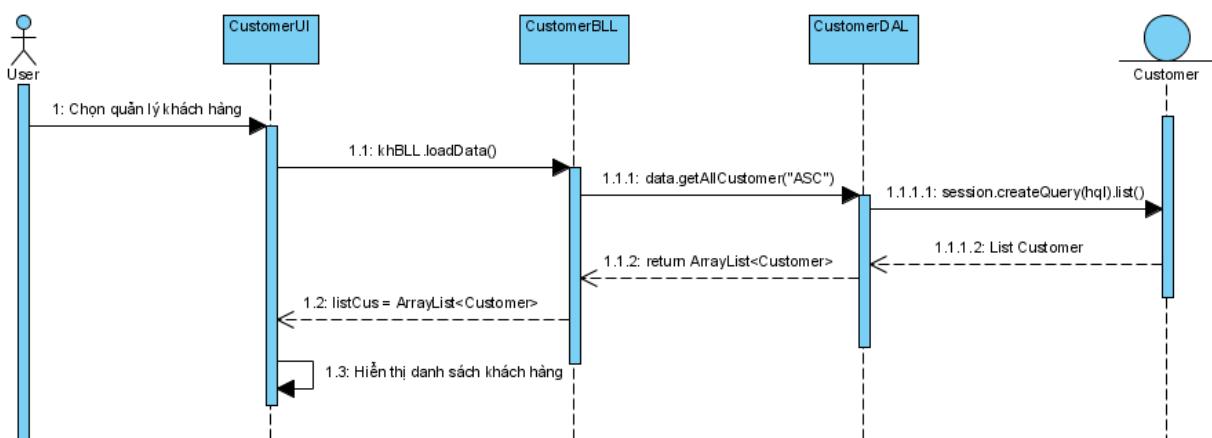
Chức năng quản lý khách hàng

Sơ đồ class



1.1. Xử lý 1: Hiển thị danh sách khách hàng

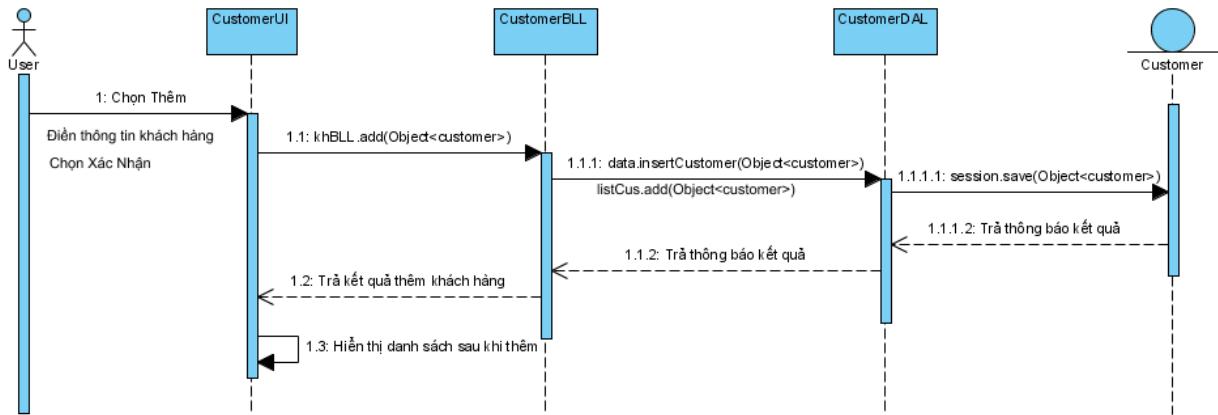
Sơ đồ tuần tự:



DAL	<pre> public List getAllCustomer(String orderby) { Session session = factory.openSession(); List listCustomer = null; Transaction tx = null; try { tx = session.beginTransaction(); String hql = "FROM Customer ORDER BY id " + orderby; listCustomer = session.createQuery(hql).list(); tx.commit(); } catch (HibernateException e) { if (tx != null) tx.rollback(); e.printStackTrace(); } finally { session.close(); } return listCustomer; } </pre>
BLL	<pre> public void loadData() { if (listCus == null) { listCus = new ArrayList<Customer>(); } listCus = data.getAllCustomer(orderby: "ASC"); } </pre>
UI	<pre> public void list() // Chép ArrayList lên table { try { if (khBLL.getListCustomer() == null) { khBLL.loadData(); } ArrayList<Customer> c = (ArrayList<Customer>) khBLL.getListCustomer(); model.setRowCount(0); outModel(model, c); } catch (Exception e) { JOptionPane.showMessageDialog(null, "Không Thể Lấy Thông Tin Danh Sách Khách Hàng"); } } public void outModel(DefaultTableModel model, ArrayList<Customer> nv) // Xuất ra Table từ ArrayList { Vector data; model.setRowCount(0); for (Customer n : nv) { data = new Vector(); data.add(n.getId()); // index table = 0 data.add(n.getFirstName()); // 1 data.add(n.getLastName()); // 2 data.add(n.getPhoneNumber()); // 3 data.add(n.getAddress()); // 4 data.add(n.getEmail()); // 5 ý dòng xét model.addRow(data); } tbl.setModel(model); } </pre>

1.2. Xử lý 2: Thêm khách hàng

Sơ đồ tuân tự



DAL

```

public int insertCustomer(Customer customer) {
    Session session = factory.openSession();
    int result = 1;
    Transaction tx = null;
    try {
        tx = session.beginTransaction();
        session.save(customer);
        tx.commit();
    } catch (HibernateException e) {
        if (tx != null) tx.rollback();

        e.printStackTrace();
        return 0;
    } finally {
        session.close();
    }
    return result;
}
  
```

BLL

```

public void add(Customer c) {
    try {
        data.insertCustomer(c);
        listCus.add(c);
    } catch (Exception e) {
        e.printStackTrace();
    }
}
  
```

UI

```

btnConfirm.addMouseListener(new MouseAdapter() {
    public void mouseClicked(MouseEvent e) {
        txtTenKH.requestFocus();
        int i;
        if (EditOrAdd) //Thêm khách hàng
        {
            String sdt = txtSDT.getText();
            //validate SDT
            Pattern pattern = Pattern.compile("^\\d{10,11}$");
            Matcher m = pattern.matcher(sdt); //so sánh
            if (!m.matches())
                JOptionPane.showConfirmDialog(null, "\"Số điện thoại không hợp lệ!! Vui lòng nhập 10 hoặc 11 số !!!\"");
            return;
        }
        for (int j = 0; j < khBILL.getListCustomer().size(); j++) {
            if (khBILL.getListCustomer().get(j).getPhoneNumber().equals(sdt))
                JOptionPane.showConfirmDialog(null, "\"Số điện thoại đã tồn tại, vui lòng nhập số khác !!!\"");
            return;
        }
        i = JOptionPane.showConfirmDialog(null, "Xác nhận thêm khách hàng", "", JOptionPane.YES_NO_OPTION);
        if (i == 0)
            //Lấy dữ liệu từ TextField
            String hoKH = txtHoKH.getText();
            String tenKH = txtTenKH.getText();
            String dienThoai = txtSDT.getText();
            String diaachi = txtAddress.getText();
            String email = txtEmail.getText();
        }
    }
}
  
```

```

        if (hoKH.equals("") || tenKH.equals("") || dienThoai.equals("") || diachi.equals("") || email.equals("")) {
            JOptionPane.showConfirmDialog(null, "\"Vui lòng nhập đầy đủ thông tin !!!\"");
            return;
        }
        //Upload khách hàng lên DAO và BUS
        Customer c = new Customer();
        c.setFirstName(hoKH);
        c.setLastName(tenKH);
        c.setPhoneNumber(dienThoai);
        c.setAddress(diachi);
        c.setEmail(email);
        khBLL.add(c);
        outModel.setModel((ArrayList<Customer>) khBLL.getListCustomer());
        JOptionPane.showConfirmDialog(null, "\"Thêm khách hàng thành công !!!\"");
        cleanView();
    }

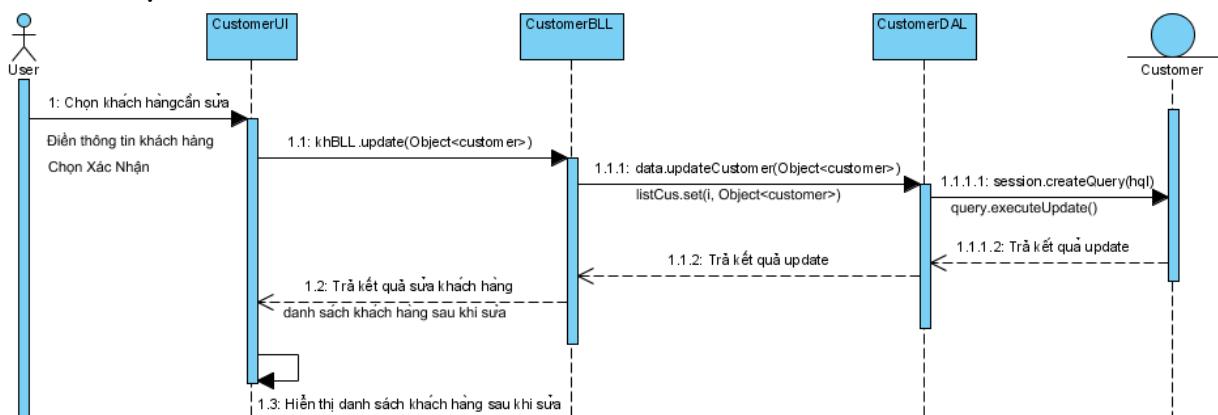
    public void cleanView() //Xóa trống các TextField
    {
        txtMaKH.setEditable(false);

        txtMaKH.setText("");
        txtHoKH.setText("");
        txtTenKH.setText("");
        txtSDT.setText("");
        txtAddress.setText("");
        txtEmail.setText("");
    }
}

```

1.3. Xử lý 3: Cập nhật khách hàng

Sơ đồ tuần tự



DAL	<pre> public int updateCustomer(Customer c) { Session session = factory.openSession(); int result = 0; Transaction tx = null; try { tx = session.beginTransaction(); String hql = "UPDATE Customer set firstName=:firstName, lastName=:lastName, phoneNumber=:phoneNumber" + " WHERE id = :id"; Query query = session.createQuery(hql); query.setParameter("id", c.getId()); query.setParameter("firstName", c.getFirstName()); query.setParameter("lastName", c.getLastName()); query.setParameter("phoneNumber", c.getPhoneNumber()); result = query.executeUpdate(); System.out.println("Rows affected: " + result); session.update(query); tx.commit(); } catch (HibernateException e) { if (tx != null) { tx.rollback(); } e.printStackTrace(); } finally { session.close(); } return result; } </pre>
BLL	<pre> public void update(Customer c) { for (int i = 0; i < listCus.size(); i++) { if (listCus.get(i).getId() == c.getId()) { try { data.updateCustomer(c); listCus.set(i, c); } catch (Exception e) { e.printStackTrace(); } } } } </pre>
UI	<pre> } else // Edit khách hàng { String sdt = txtSDT.getText(); //validate SDT Pattern pattern = Pattern.compile("^\\d{10,11}\$"); Matcher m = pattern.matcher(sdt); //so sánh if (!m.matches()) { JOptionPane.showConfirmDialog(null, "\"Số điện thoại không hợp lệ!! Vui lòng nhập 10 hoặc 11 số !!!\""); return; } for (int j = 0; j < khBLL.getListCustomer().size(); j++) { if (khBLL.getListCustomer().get(j).getPhoneNumber().equals(sdt) && khBLL.getListCustomer().get(j).getId() != Integer.parseInt(txtMaKH.getText())) { JOptionPane.showConfirmDialog(null, "\"Số điện thoại đã tồn tại, vui lòng nhập số khác !!!\""); return; } } i = JOptionPane.showConfirmDialog(null, "Xác nhận sửa Khách hàng", "", JOptionPane.YES_NO_OPTION); if (i == 0) { int maKH = Integer.parseInt(txtMaKH.getText()); String hoKH = txtHoKH.getText(); String tenKH = txtTenKH.getText(); String dienThoai = txtSDT.getText(); String diaChi = txtAddress.getText(); String email = txtEmail.getText(); if (hoKH.equals("") tenKH.equals("") dienThoai.equals("") diaChi.equals("") email.equals("")) { JOptionPane.showConfirmDialog(null, "Vui lòng nhập đầy đủ thông tin !!!"); } } } </pre>

```

Customer c = new Customer();
c.setId(maKH);
c.setFirstName(hoKH);
c.setLastName(tenKH);
c.setPhoneNumber(dienThoai);
c.setAddress(diachi);
c.setEmail(email);
KhBLL.update(c);
outModel(model, (ArrayList<Customer>) khBLL.getListCustomer());// Load lại table
JOptionPane.showConfirmDialog(null, "Sửa thông tin khách hàng thành công");

}

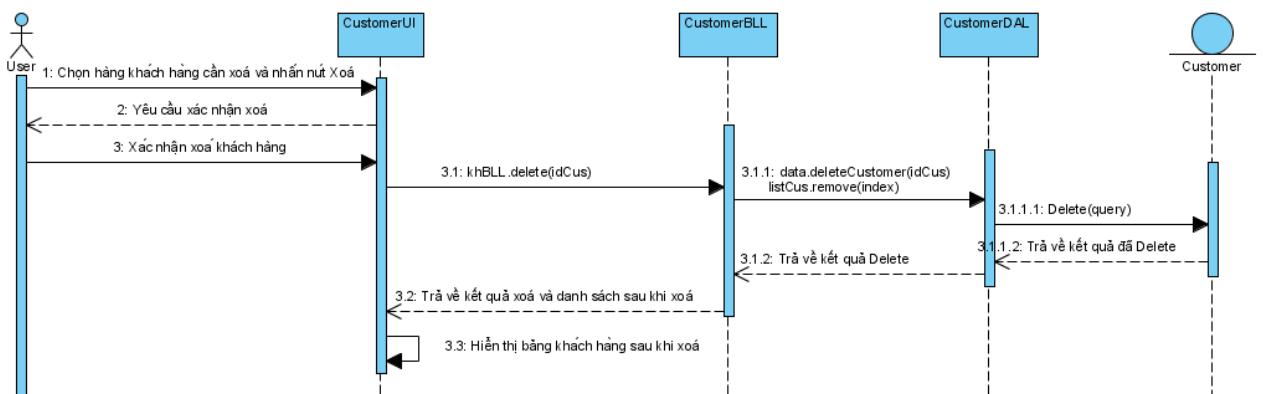
public void cleanView() //Xóa trống các TextField
{
    txtMaKH.setEditable(false);

    txtMaKH.setText("");
    txtHoKH.setText("");
    txtTenKH.setText("");
    txtSDT.setText("");
    txtAddress.setText("");
    txtEmail.setText("");
}

```

1.4. Xử lý 4: Xoá khách hàng

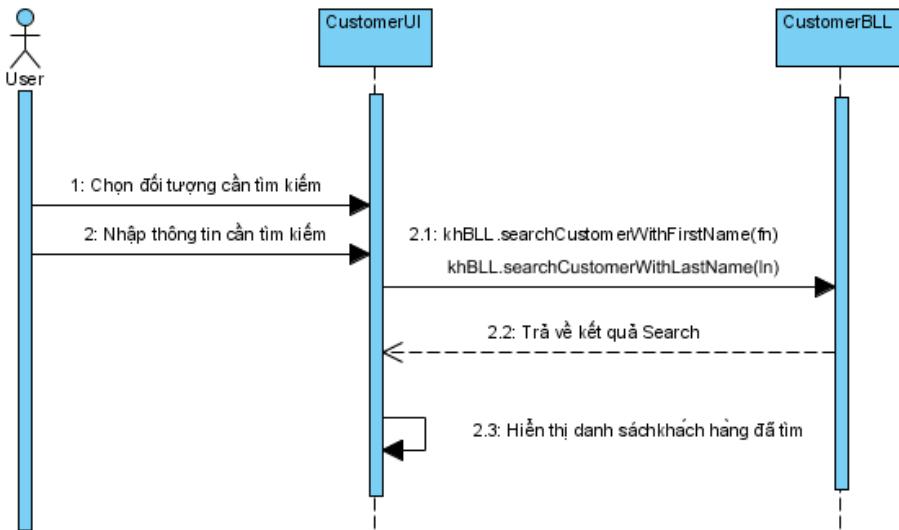
Sơ đồ tuần tự



DAL	<pre> public int deleteCustomer(int id) { Session session = factory.openSession(); int result = 0; Transaction tx = null; try { tx = session.beginTransaction(); String hql = "DELETE FROM Customer WHERE id = :id"; Query query = session.createQuery(hql); query.setParameter("id", id); result = query.executeUpdate(); // System.out.println("Rows affected: " + result); tx.commit(); } catch (HibernateException e) { if (tx != null) { tx.rollback(); } e.printStackTrace(); } finally { session.close(); } return result; } </pre>
BLL	<pre> public void delete(String id) { int idCus = Integer.parseInt(id); for (int i = 0; i < listCus.size(); i++) { if (listCus.get(i).getId() == idCus) { try { data.deleteCustomer(idCus); listCus.remove(i); } catch (Exception e) { e.printStackTrace(); } } } } </pre>
UI	<pre> btnDelete.addMouseListener(new MouseAdapter() { public void mouseClicked(MouseEvent e) { if (txtMaKH.getText().equals("")) { JOptionPane.showMessageDialog(null, "\"Vui lòng chọn khách hàng cần xóa !!!\""); return; } int i = JOptionPane.showConfirmDialog(null, "Xác nhận xóa", "Thông Báo Xác Nhận", JOptionPane.YES_NO_OPTION); if (i == 0) { khBLL.delete(txtMaKH.getText()); JOptionPane.showMessageDialog(null, "Xóa tài khoản thành công !!!"); cleanView(); tbl.clearSelection(); outModel(model, (ArrayList<Customer>) khBLL.getListCustomer()); } } }); </pre>

1.5. Xử lý 5: Tìm kiếm khách hàng

Sơ đồ tuần tự



BLL

```

public List<Customer> searchCustomerWithFirstName(String fn) {
    List<Customer> search = null;
    for (Customer ps : listCus) {
        if (ps.getFirstName().trim().toLowerCase().contains(fn.trim().toLowerCase())) {
            search.add(ps);
        }
    }
    return search;
}

public List<Customer> searchCustomerWithLastName(String ln) {
    List<Customer> search = null;
    for (Customer ps : listCus) {
        if (ps.getLastName().trim().toLowerCase().contains(ln.trim().toLowerCase())) {
            search.add(ps);
        }
    }
    return search;
}
  
```

UI

```

txtSearch.getDocument().addDocumentListener(new DocumentListener() {
    @Override
    public void insertUpdate(DocumentEvent e) {
        String text = txtSearch.getText();
        int choice = cmbChoice.getSelectedIndex(); // lấy index sort

        if (text.trim().length() == 0) {
            rowSorter.setRowFilter(null);
        } else {
            rowSorter.setRowFilter(RowFilter.regexFilter(regex: "(?i)" + text + "", choice));
        }
    }

    @Override
    public void removeUpdate(DocumentEvent e) {
        String text = txtSearch.getText();
        int choice = cmbChoice.getSelectedIndex();

        if (text.trim().length() == 0) {
            rowSorter.setRowFilter(null);
        } else {
            rowSorter.setRowFilter(RowFilter.regexFilter(regex: "(?i)" + text + "", choice));
        }
    }
})
  
```

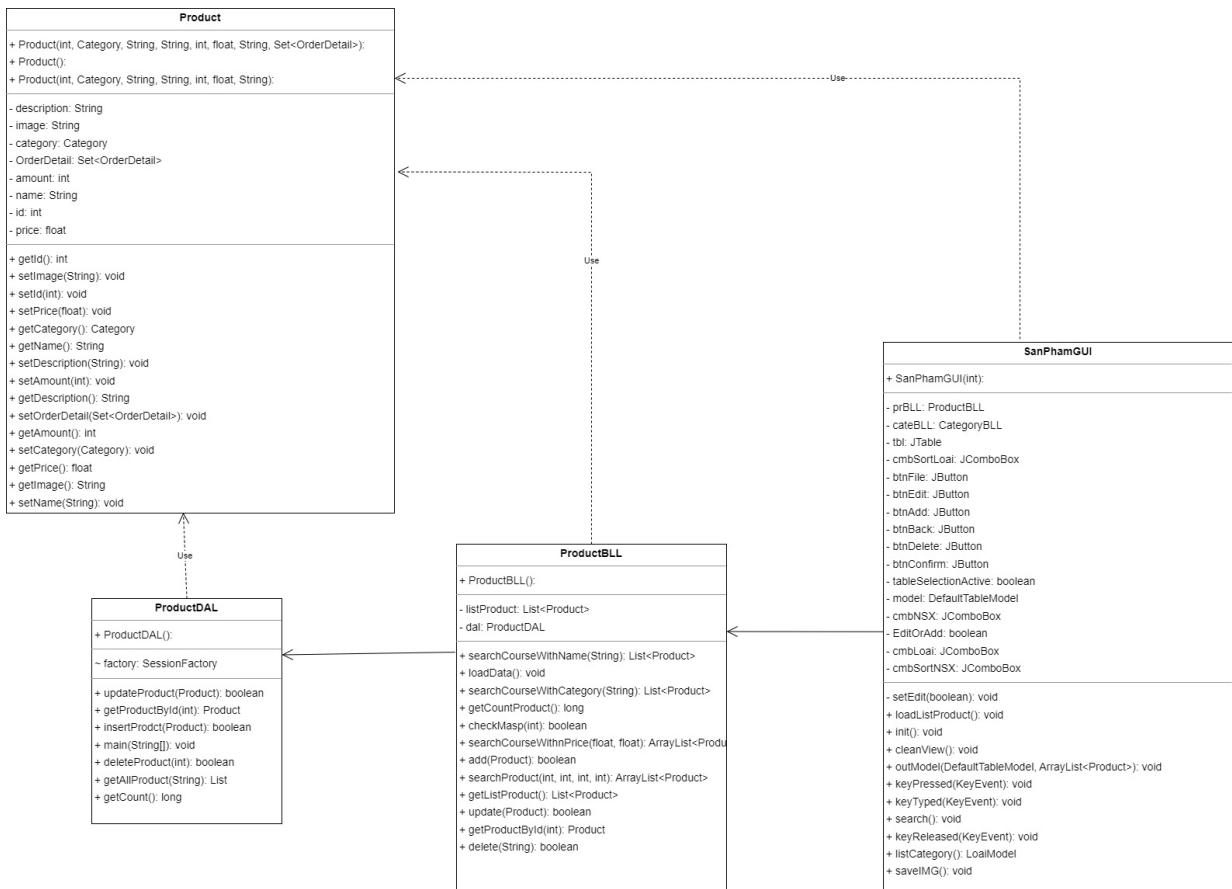
```

//PHẦN CHỌN SEARCH
JComboBox cmbChoice = new JComboBox();
cmbChoice.setFont(new Font("Segoe UI", Font.PLAIN, size: 14));
cmbChoice.addItem("Mã KH");//cmbchoice index = 0
cmbChoice.addItem("TÊN KH");//1
cmbChoice.addItem("HỌ KH");//2
cmbChoice.addItem("SĐT");//3
cmbChoice.setBounds(new Rectangle(x: 0, y: 0, width: 120, height: 30));
cmbChoice.setEditable(false);

```

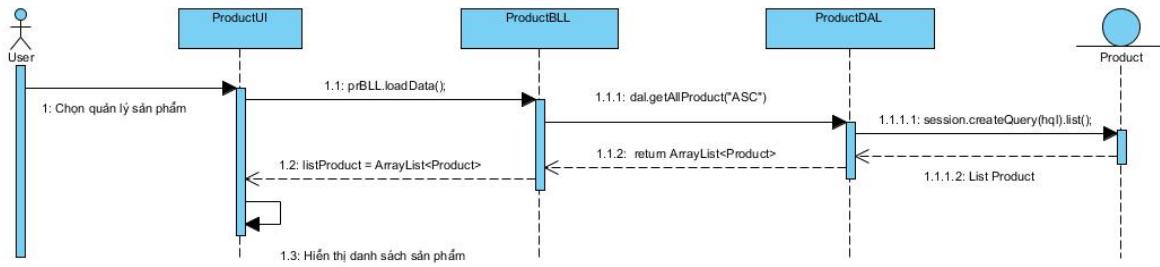
2. Chức năng quản lý sản phẩm

2.1. Sơ đồ class



2.2. Xử lý 1: Hiển thị danh sách sản phẩm

Sơ đồ tuần tự



DAL	<pre> public List getAllProduct(String orderby){ Session session = factory.openSession(); List listProduct = null; Transaction tx = null; try { tx = session.beginTransaction(); String hql = "FROM Product ORDER BY id "+orderby; listProduct = session.createQuery(hql).list(); tx.commit(); } catch (HibernateException e) { if (tx != null) tx.rollback(); e.printStackTrace(); } finally { session.close(); } return listProduct; } </pre>
BLL	<pre> public void loadData() { if (listProduct == null) { listProduct = new ArrayList<Product>(); } listProduct = dal.getAllProduct(orderby: "ASC"); } </pre>
UI	<pre> public void loadListProduct() // Chép ArrayList lên table { try { if (spBUS.getListProduct() == null) { spBUS.loadData(); } ArrayList<Product> sp = (ArrayList<Product>) spBUS.getListProduct(); model.setRowCount(0); outModel(model, sp); } catch (Exception e) { JOptionPane.showMessageDialog(parentComponent: null, message: "Không Thể Lấy Thông Tin Danh Sách Sản Phẩm"); } } </pre>

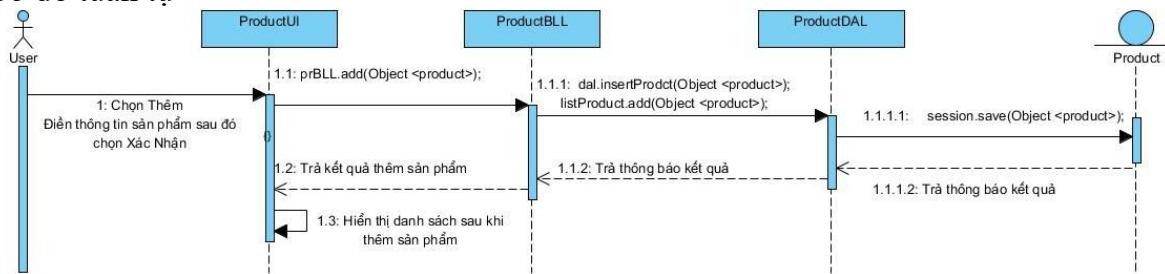
```

public void outModel(DefaultTableModel model, ArrayList<Product> sp) // Xuất ra Table từ ArrayList
{
    Vector data;
    model.setRowCount(0);
    for (Product s : sp) {
        data = new Vector();
        data.add(s.getId());
        data.add(s.getName());
        data.add(s.getPrice());
        data.add(s.getCategory().getId());
        data.add(s.getDescription());
        data.add(s.getImage());
        model.addRow(data);
    }
    tbl.setModel(model);
}

```

2.3. Xử lý 2: Thêm Sản Phẩm

Sơ đồ tuần tự



DAL

```

public int insertProdct(Product product){
    Session session = factory.openSession();
    int result = 1;
    Transaction tx = null;
    try {
        tx = session.beginTransaction();
        session.save(product);
        tx.commit();
    } catch (HibernateException e) {
        if (tx != null) tx.rollback();

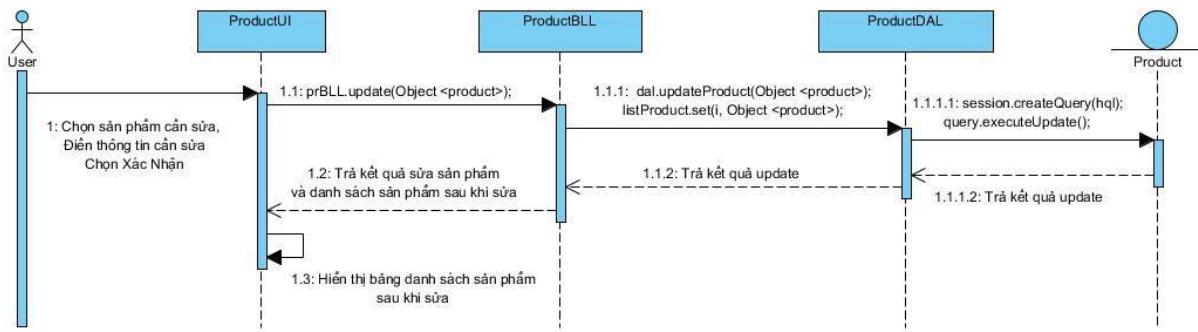
        e.printStackTrace();
        return 0;
    } finally {
        session.close();
    }
    return result;
}

```

BLL	<pre> public void add(Product spDTO) { try { dal.insertProduct(spDTO); listProduct.add(spDTO); } catch (Exception e) { e.printStackTrace(); } } </pre>
UI	<pre> btnConfirm.addMouseListener((MouseListener) mouseClicked(e) → { txtTenSP.requestFocus(); int i; if (EditOrAdd) //Thêm Sản Phẩm { i = JOptionPane.showConfirmDialog(parentComponent: null, message: "Xác nhận thêm sản phẩm", title: "", JOptionPane.YES_NO_OPTION); if (i == 0) { //Lấy dữ liệu từ TextField String tenSP = txtTenSP.getText(); float gia = txtGia.getText().equals("") ? 0 : Float.parseFloat(txtGia.getText()); String mota = txtMT.getText(); Category loai = (Category) cmbLoai.getSelectedItem(); //gán loại tạm // Category loai=new Category(1,"Bánh Đàn"); int maLoai = loai.getId(); String IMG = imgName; //Upload sản phẩm lên DAO và BUS if (tenSP.equals("")) gia == 0 mota.equals("") IMG.equals("") maLoai == 0) { JOptionPane.showMessageDialog(parentComponent: null, message: "Bạn chưa nhập đủ thông tin để thêm sản phẩm !!!"); return; } Product sp = new Product(); sp.setCategory(loai); sp.setName(tenSP); sp.setDescription(mota); sp.setAmount(10); sp.setPrice(gia); sp.setImage(IMG); spBUS.add(sp); outModel(model, (ArrayList<Product>) spBUS.getListProduct()); // Load lại table saveIMG(); // Lưu hình ảnh JOptionPane.showMessageDialog(parentComponent: null, message: "Thêm sản phẩm thành công !!!"); cleanView(); } } } public void cleanView() //Xóa trắng các TextField { txtId.setEditable(false); txtId.setText(""); txtTenSP.setText(""); txtGia.setText(""); txtMT.setText(""); img.setIcon(null); img.setText("Image"); cmbLoai.setSelectedIndex(0); imgName = "null"; } </pre>

2.4. Xử lý 3: Cập nhật sản phẩm

Sơ đồ tuần tự



DAL	<pre> public int updateProduct(Product product){ Session session = factory.openSession(); int result = 0; Transaction tx = null; try { tx = session.beginTransaction(); String hql = "UPDATE Product set category=:category, name = :name, description = :description," + " price = :price, amount=:amount,image=:image WHERE id = :id"; Query query = session.createQuery(hql); query.setParameter(name: "name", product.getName()); query.setParameter(name: "id", product.getId()); query.setParameter(name: "category", product.getCategory()); query.setParameter(name: "description", product.getDescription()); query.setParameter(name: "price", product.getPrice()); query.setParameter(name: "amount", product.getAmount()); query.setParameter(name: "image", product.getImage()); System.out.println(hql); result = query.executeUpdate(); System.out.println("Rows affected: " + result); tx.commit(); } catch (HibernateException e) { if (tx != null) tx.rollback(); e.printStackTrace(); } finally {session.close();} return result; } </pre>
BLL	<pre> public void update(Product spDTO) { for (int i = 0; i < listProduct.size(); i++) { if (listProduct.get(i).getId() == spDTO.getId()) { try { dal.updateProduct(spDTO); listProduct.set(i, spDTO); } catch (Exception e) { e.printStackTrace(); } } } } </pre>

UI

```
i = JOptionPane.showConfirmDialog( parentComponent: null, message: "Xác nhận sửa sản phẩm", title: "", JOptionPane.YES_NO_OPTION);
if (i == 0) {
    //Lấy dữ liệu từ TextField
    int maSP = Integer.parseInt(txtId.getText());
    String tenSP = txtTenSP.getText();
    float gia = Float.parseFloat(txtGia.getText());
    String mota = txtMT.getText();

    Category loai = (Category) cmbLoai.getSelectedItem();
    /// Category loai=new Category(3,"Cà Phê Pha Máy");
    int maLoai = loai.getId();

    String IMG = imgName;
    if (tenSP.equals("") || gia == 0 || IMG.equals("") || maLoai == 0) {
        JOptionPane.showMessageDialog( parentComponent: null, message: "Bạn chưa nhập đủ thông tin để sửa sản phẩm");
        return;
    }

    //Upload sản phẩm lên DAO và BUS
    Product sp = new Product();
    sp.setCategory(loai);
    sp.setName(tenSP);
    sp.setDescription(mota);
    sp.setAmount(10);
    sp.setPrice(gia);
    sp.setImage(IMG);
    sp.setId(maSP);
    spBUS.update(sp);
    outModel(model, (ArrayList<Product>) spBUS.getListProduct()); // Load lại table
    saveIMG(); // Lưu hình ảnh
    JOptionPane.showMessageDialog( parentComponent: null, message: "Sửa sản phẩm thành công");
}

});

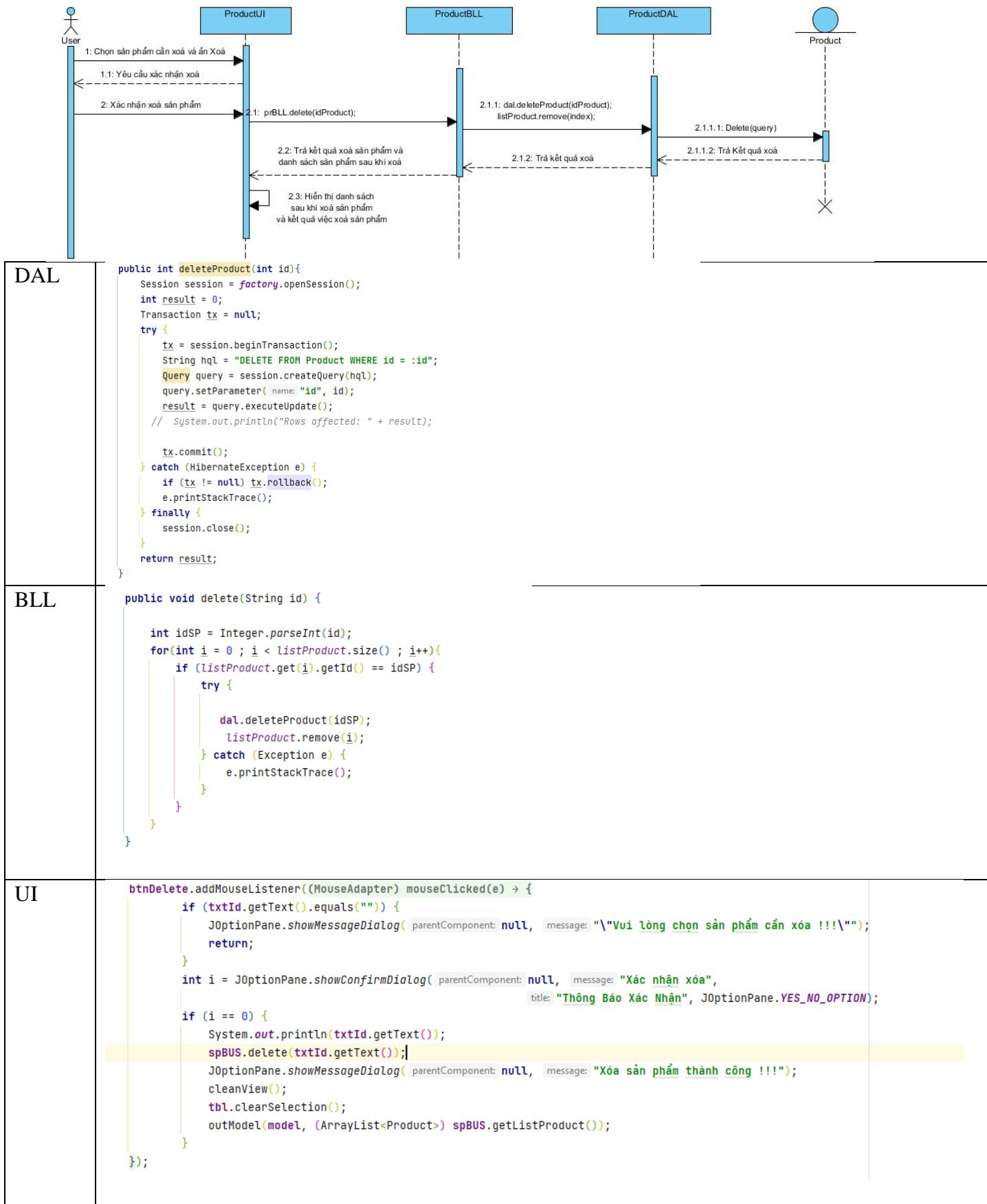
public void cleanView() //Xóa trắng các TextField
{
    txtId.setEditable(false);

    txtId.setText("");
    txtTenSP.setText("");
    txtGia.setText("");
    txtMT.setText("");

    img.setIcon(null);
    img.setText("Image");
    cmbLoai.setSelectedIndex(0);
    imgName = "null";
}
```

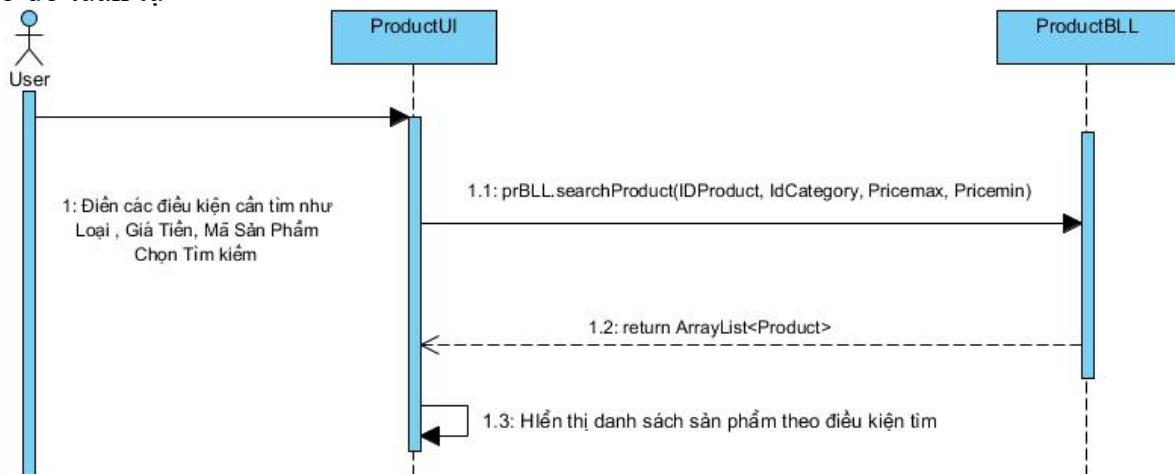
2.5. Xử lý 4: Xóa sản phẩm

Sơ đồ tuần tự



2.6. Xử lý 5: Tìm kiếm sản phẩm

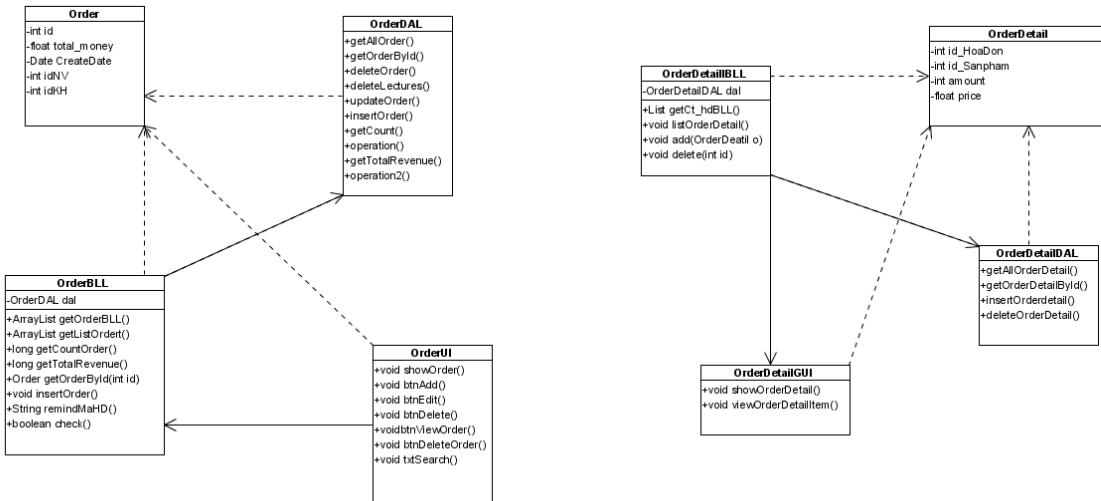
Sơ đồ tuần tự



BLL	<pre> public ArrayList<Product> searchProduct(int masp, int maloai, int max, int min) { ArrayList<Product> search = new ArrayList<>(); for (Product sp : listProduct) { Product spTemp = null; if (masp == 0 && maloai == 0) { spTemp = sp; } else if (masp == 0) { if (sp.getCategory().getId() == maloai) { spTemp = sp; } } else if (maloai == 0) { if (sp.getId() == masp) { spTemp = sp; } } if (spTemp != null && spTemp.getPrice() >= min && spTemp.getPrice() <= max) { search.add(spTemp); } } return search; } </pre>
UI	<pre> public void search() { int masp = sortMaSP.getText().equals("") ? 0 : Integer.parseInt(sortMaSP.getText()); int maloai = 0; if (cmbSortLoai.getSelectedIndex() != 0) { Category loai = (Category) cmbSortLoai.getSelectedItem(); maloai = loai.getId(); System.out.println(maloai); } int max = txtMaxPrice.getText().equals("") ? 99999999 : Integer.parseInt(txtMaxPrice.getText()); int min = txtMinPrice.getText().equals("") ? 0 : Integer.parseInt(txtMinPrice.getText()); outModel(model, spBUS.searchProduct(masp, maloai, max, min)); } </pre>

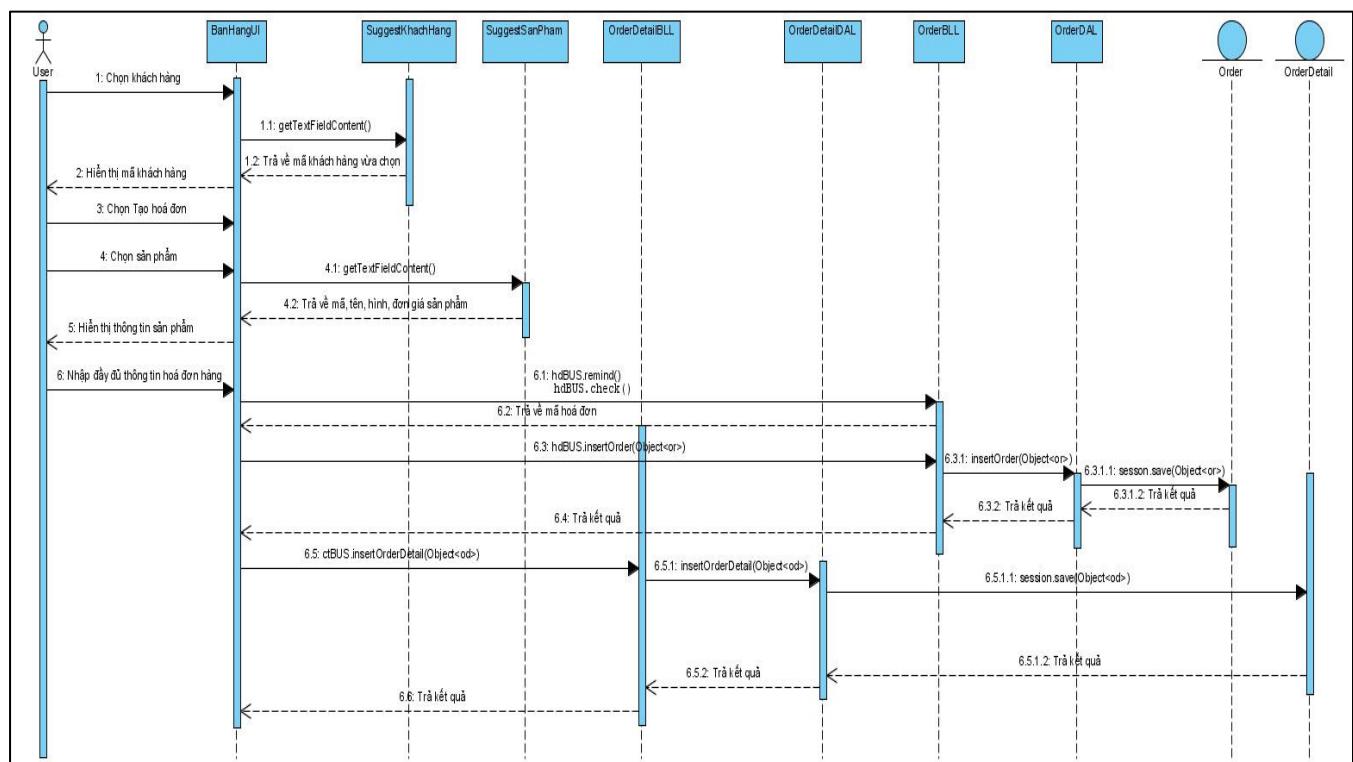
3. Chức năng quản lý hóa đơn

Sơ đồ class



3.1. Xử lý 1: Tạo hóa đơn bán hàng

Sơ đồ tuần tự:

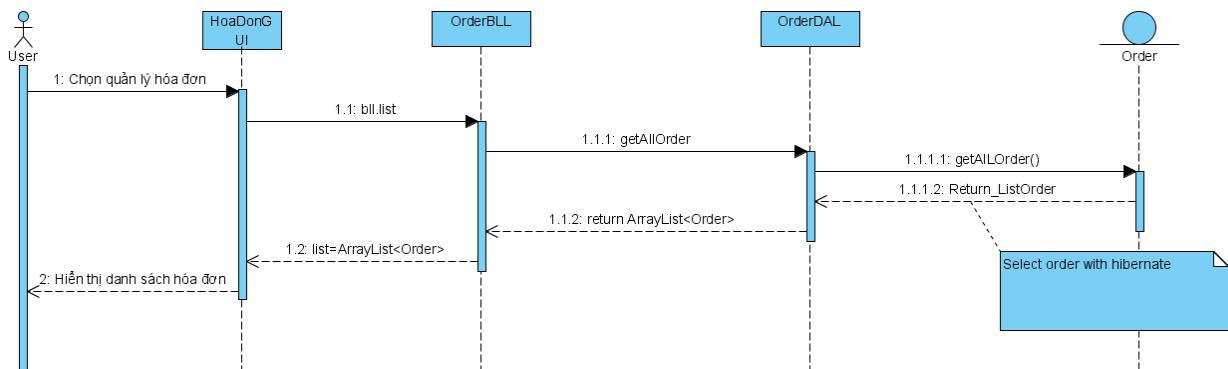


DAL	<ul style="list-style-type: none"> ■ OrderDal: <pre> 119 public int insertOrder(Order order){ 120 Session session = factory.openSession(); 121 int result = 1; 122 Transaction tx = null; 123 try { 124 tx = session.beginTransaction(); 125 session.save(o:order); 126 tx.commit(); 127 } catch (HibernateException e) { 128 if (tx != null) tx.rollback(); 129 e.printStackTrace(); 130 return 0; 131 } finally { 132 session.close(); 133 } 134 return result; 135 } </pre> <ul style="list-style-type: none"> ■ OrderDetailDal: <pre> public int insertOrderdetail(OrderDetail orderDetail){ Session session = factory.openSession(); int result = 1; Transaction tx = null; try { tx = session.beginTransaction(); session.save(orderDetail); tx.commit(); } catch (HibernateException e) { if (tx != null) tx.rollback(); e.printStackTrace(); return 0; } finally { session.close(); } return result; } </pre>
BLL	<ul style="list-style-type: none"> ■ OrderBLL: <pre> 84 public void insertOrder(Order hd) { 85 try { 86 hdBLL.add(e:hd); 87 dal.insertOrder(order:hd); 88 } 89 catch (Exception e){ 90 e.printStackTrace(); 91 } 92 } 93 } 94 </pre> <ul style="list-style-type: none"> ■ OrderDetailBLL:

	<pre> public void insertOrderDetail(OrderDetail cthdDTO) { try { orderDetailDAL.insertOrderdetail(cthdDTO); orderDetailsBLL.add(cthdDTO); } catch (Exception ex) { ex.printStackTrace(); } } </pre>
UI	<ul style="list-style-type: none"> ▪ Tạo Order: <pre> 526 if (e.getSource().equals(obj:btnNewHD)) // Tạo hóa đơn { Date date = Timestamp.valueOf(dateTime:LocalDateTime.now()); if (txtMaHD.getText().isEmpty()) { JOptionPane.showMessageDialog(parentComponent: null, message: "Vui lòng nhập mã hóa đơn", title: "Thông báo", messageType: 0); txtMaHD.requestFocus(); return; } else if (hdBUS.check(maHD:txtMaHD.getText())) { JOptionPane.showMessageDialog(parentComponent: null, message: "Mã hóa đơn đã tồn tại", title: "Thông báo", messageType: 2); txtMaHD.requestFocus(); txtMaHD.setText(ti:hdBUS.remindMaHD()); return; } if (txtMaNV.getText().isEmpty()) { txtMaNV.setText(t:"1"); } txtNgayHD.setText(t:date.toString()); reset(flag:false); flag = false; txtMaSP.requestFocus(); ----- Khởi tạo hóa đơn ----- int maHD = Integer.parseInt(s:txtMaHD.getText().trim()); int maKH = Integer.parseInt(s:txtMaKH.getText().trim()); int maNV = Integer.parseInt(s:txtMaNV.getText().trim()); Timestamp stamp = Timestamp.valueOf(s:txtNgayHD.getText()); Date ngayHD = new Date(date:stamp.getTime()); float tongTien = Float.parseFloat(s:txtTongTien.getText()); Customer customer = khBUS.getCustomerById(id:maKH); List<OrderDetail> orderDetail = ctBUS.get Ct_hdBLL(); Order hd = new Order(id:maHD, totalPrice:tongTien, createdDate:ngayHD, id_Staff:maNV, customer, orderDetail); hdBUS.insertOrder(hd); } </pre> <ul style="list-style-type: none"> ▪ Tạo Các OrderDetail: <p>Sau khi tạo hóa đơn người dùng chọn sản phẩm số lượng để thêm vào các chi tiết đơn hàng và ấn nút Xác Nhận .</p> <pre> if (e.getSource().equals(btnConfirm)) //Xác nhận { if (dsct.isEmpty()) { JOptionPane.showMessageDialog(parentComponent: null, message: "Vui lòng chọn mã sản phẩm", title: "Thông báo", messageType: 0); return; } for (OrderDetail ct : dsct) { System.out.println(ct); ctBUS.insertOrderDetail(ct); } JOptionPane.showMessageDialog(parentComponent: null, message: "Thêm hóa đơn thành công !!!", title: "Thành công", JOptionPane.INFORMATION_MESSAGE); flag = true; reset(flag: true); } </pre>

3.2. Xử lý 2: Hiển thị danh sách hóa đơn

Sơ đồ tuần tự



DAL	<pre> public class OrderDAL { static final SessionFactory factory = HibernateUtils.getSessionFactory(); public List getAllOrder(String orderby) { Session session = factory.openSession(); List listOrder = null; Transaction tx = null; try { tx = session.beginTransaction(); String hql = "FROM Order ORDER BY id " + orderby; listOrder = session.createQuery(string:hql).list(); tx.commit(); } catch (HibernateException e) { if (tx != null) tx.rollback(); e.printStackTrace(); } finally { session.close(); } return listOrder; } } </pre>
BLL	<pre> 24 public void list() { 25 try{ 26 hdBLL = new ArrayList<>(); 27 hdBLL = dal.getAllOrder(orderby: "DESC"); 28 29 } catch (Exception e){ 30 e.printStackTrace(); 31 } 32 } 33 </pre>

UI

```

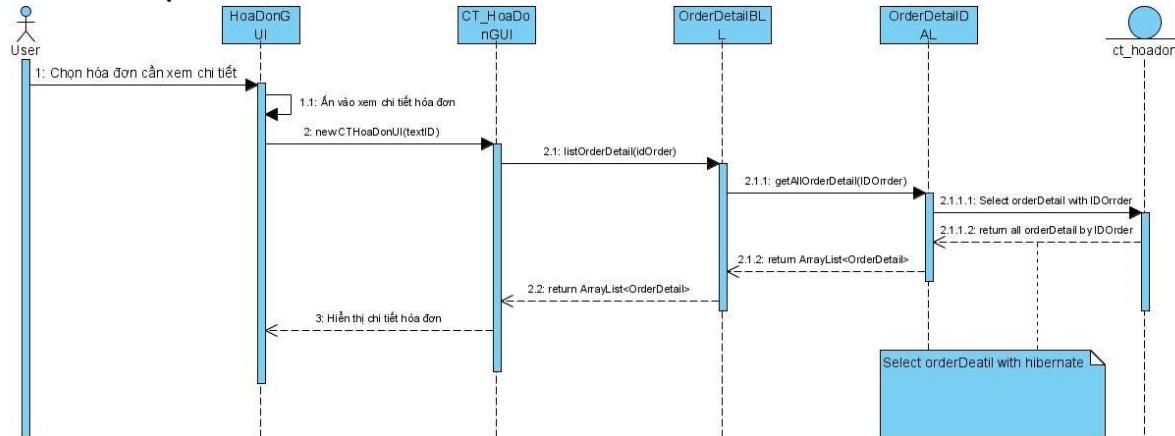
public void outModel(DefaultTableModel model, ArrayList<Order> hd) { //xuat tu arraylist len table
    Vector data;
    model.setRowCount( rowCount: 0 );
    for (Order h : hd) {
        data = new Vector();
        data.add( e: h.getId());
        data.add( e: h.getCustomer().getId());
        data.add( e: h.getId_Staff());
        data.add( e: h.getCreatedDate());
        data.add( e: h.getTotalPrice());
        model.addRow( rowData: data);
    }
    tbl.setModel( dataModel: model );
}

public void list() // Chép ArrayList lên table
{
    if (orderBLL.getOrderBLL() == null) {
        orderBLL.list();
    }
    ArrayList<Order> hd = (ArrayList<Order>) orderBLL.getOrderBLL();
    model.setRowCount( rowCount: 0 );
    outModel(model, hd);
}
}

```

3.3. Xử lý 3: Xem chi tiết hóa đơn

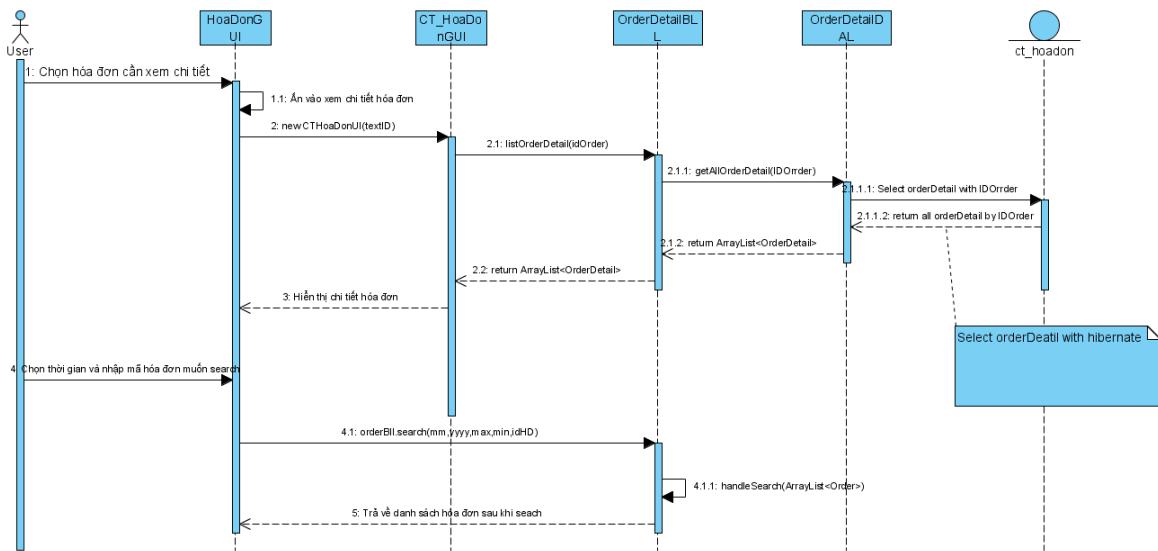
Sơ đồ tuần tự



DAL	<pre> public class OrderDetailDAL { static final SessionFactory factory = HibernateUtils.getSessionFactory(); public List getAllOrderDetail(int orderID) { Session session = factory.openSession(); List listOrderDetail = null; Transaction tx = null; try { tx = session.beginTransaction(); String hql = "FROM OrderDetail WHERE order='"+orderID; listOrderDetail = session.createQuery(string:hql).list(); tx.commit(); } catch (HibernateException e) { if (tx != null) tx.rollback(); e.printStackTrace(); } finally { session.close(); } return listOrderDetail; } } </pre>
BLL	<pre> public final class OrderDetailBLL { 6 usages private List<OrderDetail> orderDetailsBLL; 2 usages public OrderDetailBLL() { orderDetailsBLL = null; } 3 usages OrderDetailDAL cthdDAO = new OrderDetailDAL(); 4 usages public OrderDetailBLL(int i1) { listOrderDetail(i1); } 6 usages public List<OrderDetail> getCt_hdBLL() { return orderDetailsBLL; } 3 usages public void listOrderDetail(int orderID) { orderDetailsBLL = cthdDAO.getAllOrderDetail(orderID); } } </pre>
UI	<pre> public void outModel(DefaultTableModel model, ArrayList<OrderDetail> hd) { //xuat tu arraylist len table Vector data; model.setRowCount(rowCount:0); for (OrderDetail h : hd) { data = new Vector(); data.add(e:h.getId()); data.add(e:h.getProduct().getName()); data.add(e:h.getAmount()); data.add(e:h.getPrice()); data.add(e:h.getOrder()); model.addRow(rowData:data); } tbl.setModel(dataModel:model); } public void list() // Chép ArrayList lên table { if (ctBLL.getCt_hdBLL() == null) { ctBLL.listOrderDetail(orderID: Integer.parseInt(s:mahd)); } ArrayList<OrderDetail> cthd = (ArrayList<OrderDetail>) ctBLL.getCt_hdBLL(); model.setRowCount(rowCount:0); outModel(model, hd:cthd); } </pre>

3.4. Xử lý 4: Tìm kiếm hóa đơn

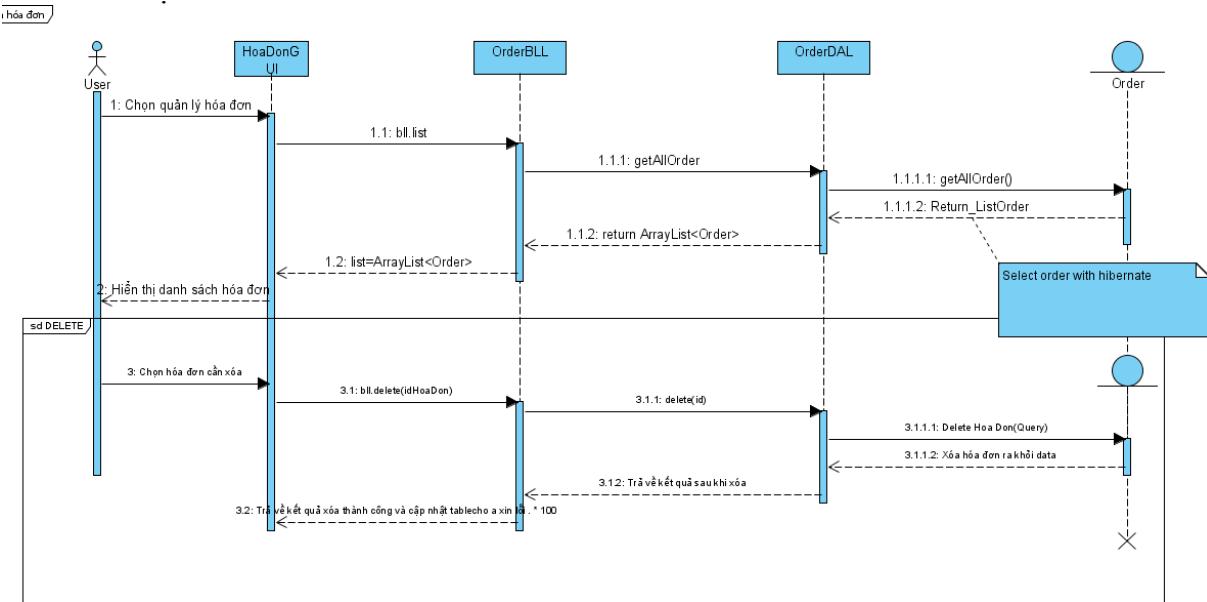
So đồ tuần tự



BLL	<pre> public ArrayList<Order> search(int mm, int yyyy, double max, double min, int maHD) { int mm1 = 0, mm2 = 12; int yyyy1 = 0, yyyy2 = Calendar.getInstance().get(Calendar.YEAR); if (mm != -1) { mm1 = mm; mm2 = mm; } if (yyyy != 0) { yyyy1 = yyyy; yyyy2 = yyyy; } for (Order hd : hdBL) { Date date = hd.getCreatedDate(); Calendar calendar = Calendar.getInstance(); calendar.setTimeInMillis(date.getTime()); int month = calendar.get(Calendar.MONTH); int year = calendar.get(Calendar.YEAR); if (hd.getTotalPrice() >= min && hd.getTotalPrice() <= max && (month >= mm1 && month <= mm2) && (year >= yyyy1 && year <= yyyy2)) { if (maHD != 0 && hd.getId() == maHD) { search.clear(); search.add(hd); break; } if (maHD != 0 && hd.getId() != maHD) { search.clear(); } search.add(hd); } } return search; } </pre>
UI	<pre> public void search() { int maHD = txtMaHDSearch.getText().equals("anObject: """) ? 0 : Integer.parseInt(txtMaHDSearch.getText()); int mm = monthChoice.getSelectedIndex() - 1; int yyyy; try { yyyy = Integer.parseInt(yearChoice.getSelectedItem()); } catch (NumberFormatException ex) { yyyy = 0; } double max = txtMaxPrice.getText().equals("anObject: """) ? 99999999 : Double.parseDouble(txtMaxPrice.getText()); double min = txtMinPrice.getText().equals("anObject: """) ? 0 : Double.parseDouble(txtMinPrice.getText()); outModel(model, hd:orderBL.search(mm, yyyy, max, min, maHD)); } </pre>

3.5. Xử lý 5: Xóa hóa đơn

Sơ đồ tuần tự



DAL

- OrderDal:

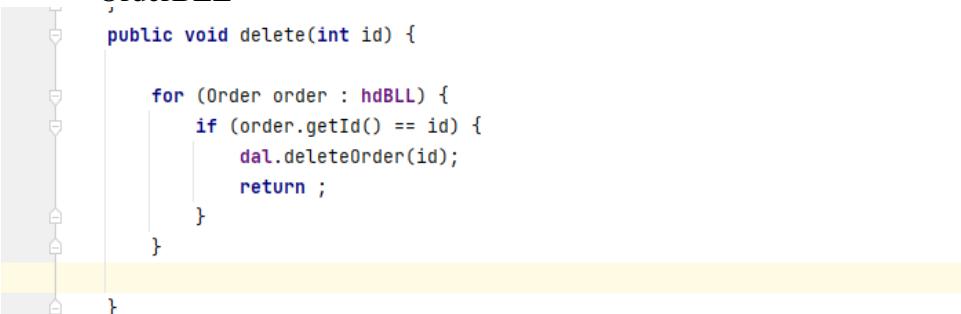
```

2 usages
public int deleteOrder(int id) {
    Session session = factory.openSession();
    int result = 0;
    Transaction tx = null;
    try {
        tx = session.beginTransaction();
        String hql = "DELETE FROM Order WHERE id = :id";
        Query query = session.createQuery(hql);
        query.setParameter( <id>, id);
        result = query.executeUpdate();
        // System.out.println("Rows affected: " + result);

        tx.commit();
    } catch (HibernateException e) {
        if (tx != null)
            tx.rollback();
        e.printStackTrace();
    } finally {
        session.close();
    }
    return result;
}

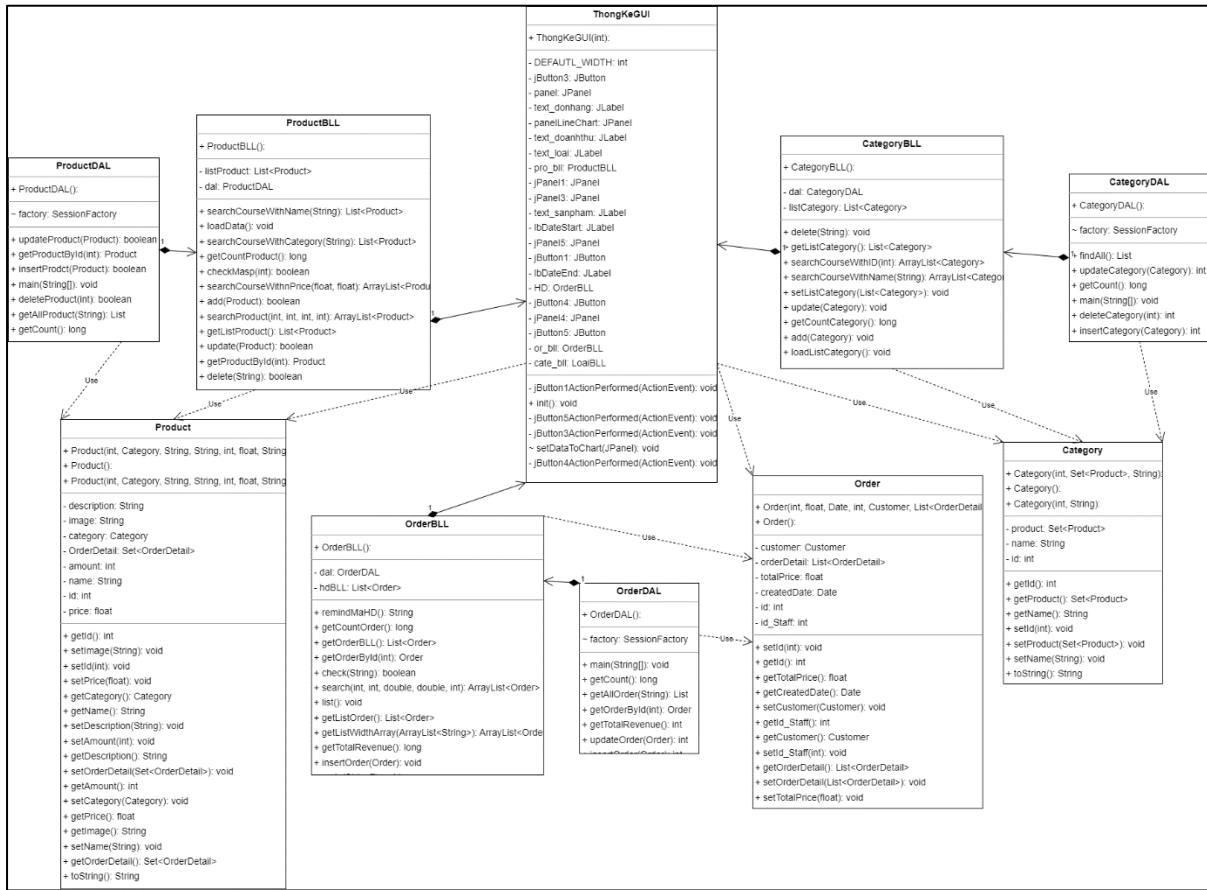
```

- OrderDetailDal:

	<pre> 105 public int deleteOrderDetail(int id){ 106 Session session = factory.openSession(); 107 int result = 0; 108 Transaction tx = null; 109 try { 110 tx = session.beginTransaction(); 111 String hql = "DELETE FROM OrderDetail WHERE order = :id"; 112 Query query = session.createQuery(string:hql); 113 query.setParameter(string:"id", o:id); 114 result = query.executeUpdate(); 115 // System.out.println("Rows affected: " + result); 116 117 tx.commit(); 118 } catch (HibernateException e) { 119 if (tx != null) tx.rollback(); 120 e.printStackTrace(); 121 } finally { 122 session.close(); 123 } 124 return result; 125 } </pre>
BLL	<p>■ OrderBLL</p>  <pre> public void delete(int id) { for (Order order : hdBLL) { if (order.getId() == id) { dal.deleteOrder(id); return ; } } } </pre> <p>■ OrderDetailBLL</p> <pre> public void delete(String id) { int idHD = Integer.parseInt(id); listOrderDetail(idHD); for (OrderDetail cthdDTO : orderDetailsBLL) { if (cthddTO.getOrder().getId() == idHD) { orderDetailsBLL.remove(cthdDTO); cthdDAO.deleteOrderDetail(idHD); return; } } } </pre>
UI	<pre> btnDelete.addMouseListener((MouseAdapter) mouseClicked(e) { if (txtMaHD.getText().equals("")) { JOptionPane.showMessageDialog(parentComponent: null, message: "Vui lòng chọn hóa đơn cần xóa !!!"); return; } int i = JOptionPane.showConfirmDialog(parentComponent: null, message: "Xác nhận xóa", title: "Alert", JOptionPane.YES_NO_OPTION); if (i == 0) { orderBLL.delete(txtMaHD.getText()); orderDetailBLL.delete(txtMaHD.getText()); JOptionPane.showMessageDialog(parentComponent: null, message: "Xóa hóa đơn thành công !!!"); cleanView(); tbl.clearSelection(); outModel(model, (ArrayList<Order>) orderBLL.getOrderBLL()); } }); </pre>

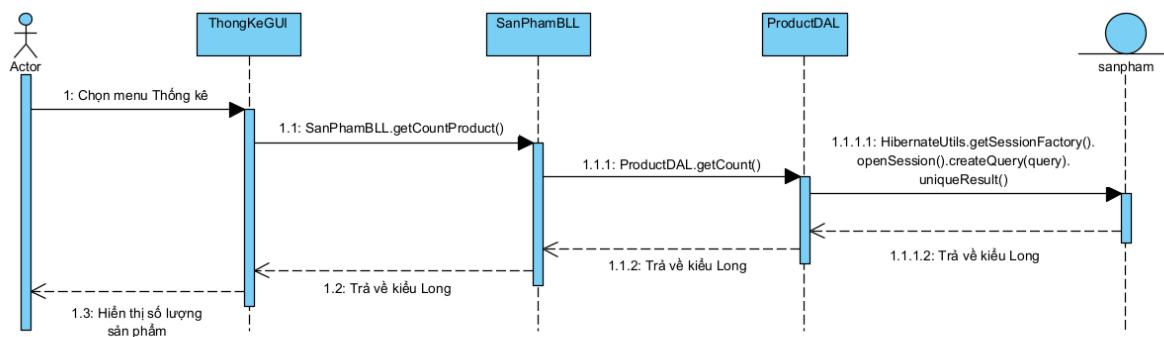
4. Chức năng quản lý thống kê

4.1. Sơ đồ class



4.2. Xử lý 1: Hiển thị số lượng sản phẩm

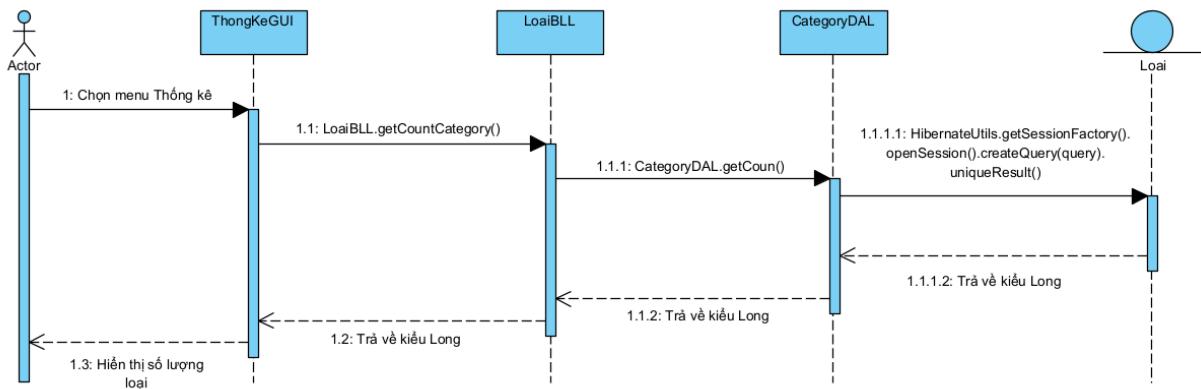
Sơ đồ tuần tự:



DAL	<pre> 125 public long getCount() { 126 Session session = factory.openSession(); 127 long amount = 0; 128 Transaction tx = null; 129 try { 130 tx = session.beginTransaction(); 131 Query query = session.createQuery("select count(*) from Product"); 132 amount = (long)query.uniqueResult(); 133 tx.commit(); 134 } catch (HibernateException e) { 135 if (tx != null) tx.rollback(); 136 e.printStackTrace(); 137 } finally { 138 session.close(); 139 } 140 } 141 }</pre>
BLL	<pre> 168 public long getCountProduct() { 169 try{ 170 return dal.getCount(); 171 } 172 catch (Exception e) { 173 e.printStackTrace(); 174 } 175 return 0; 176 } 177 }</pre>
UI	<pre> public class ThongKeGUI extends JPanel{ private int DEFAULT_WIDTH; private JLabel lbDateStart,lbDateEnd; private JPanel panel; private ProductBLL pro_bll=new ProductBLL(); private CategoryBLL cate_bll=new CategoryBLL(); private OrderBLL or_bll=new OrderBLL(); private OrderBLL HD=new OrderBLL(); public ThongKeGUI(int width) { this.DEFAULT_WIDTH = width; init(); text_loai.setText(String.valueOf(cate_bll.getCountCategory())); text_sanpham.setText(String.valueOf(pro_bll.getCountProduct())); text_donhang.setText(String.valueOf(or_bll.getCountOrder())); text_doanhthu.setText(String.valueOf(or_bll.getTotalRevenue())+" VND"); setDataToChart(panelLineChart); } void setDataToChart(JPanel jp){ List<Order> listOrder=(ArrayList<Order>) HD.getListOrder(); if(listOrder!=null){ DefaultCategoryDataset dataset = new DefaultCategoryDataset(); for(Order item : listOrder){ dataset.addValue(item.getTotalPrice(),"Total",item.getCreatedDate()); } //create chart JFreeChart chart = ChartFactory.createLineChart("THỐNG KÊ DOANH THU","Thời Gian","Doanh Thu (VND)",dataset); ChartPanel chpn=new ChartPanel(chart); chpn.setPreferredSize(new Dimension(jp.getWidth(),450)); jp.removeAll(); jp.setLayout(new CardLayout()); jp.add(chpn); jp.validate(); jp.repaint(); } } }</pre>

4.3. Xử lý 2: Hiển thị số lượng Loại

Sơ đồ tuần tự:



DAL	<pre> 109 public long getCount() { 110 Session session = factory.openSession(); 111 long amount = 0; 112 Transaction tx = null; 113 try { 114 tx = session.beginTransaction(); 115 Query query = session.createQuery("select count(*) from loai"); 116 amount = (long)query.uniqueResult(); 117 tx.commit(); 118 } catch (HibernateException e) { 119 if (tx != null) tx.rollback(); 120 e.printStackTrace(); 121 } finally { 122 session.close(); 123 } 124 return amount; 125 } </pre>
BLL	<pre> 55 public long getCountCategory() { 56 try{ 57 return dal.getCount(); 58 } 59 catch (Exception e) { 60 e.printStackTrace(); 61 } 62 return 0; 63 } </pre>

UI

```

public class ThongKeGUI extends JPanel{
    private int DEFAULT_WIDTH;
    private JLabel lbDateStart,lbDateEnd;
    private JPanel panel;
    private ProductBLL pro_bll=new ProductBLL();
    private CategoryBLL cate_bll=new CategoryBLL();
    private OrderBLL or_bll=new OrderBLL();
    private OrderBLL HD=new OrderBLL();

    public ThongKeGUI(int width) {
        this.DEFAULT_WIDTH = width;
        init();
        text_loai.setText(String.valueOf(cate_bll.getCountCategory()));
        text_sanpham.setText(String.valueOf(pro_bll.getCountProduct()));
        text_donhang.setText(String.valueOf(or_bll.getCountOrder()));
        text_doanhthu.setText(String.valueOf(or_bll.getTotalRevenue())+" VND");
        setDataToChart(panelLineChart);
    }

    void setDataToChart(JPanel jp){
        List<Order> listOrder=(ArrayList<Order>) HD.getListOrder();
        if(listOrder!=null){
            DefaultCategoryDataset dataset = new DefaultCategoryDataset();
            for(Order item : listOrder){
                dataset.addValue(item.getTotalPrice(),"Total",item.getCreatedDate());
            }
            //create chart
            JFreeChart chart = ChartFactory.createLineChart("THÔNG KÉ DOANH THU","Thời Gian","Doanh Thu (VND)",dataset);

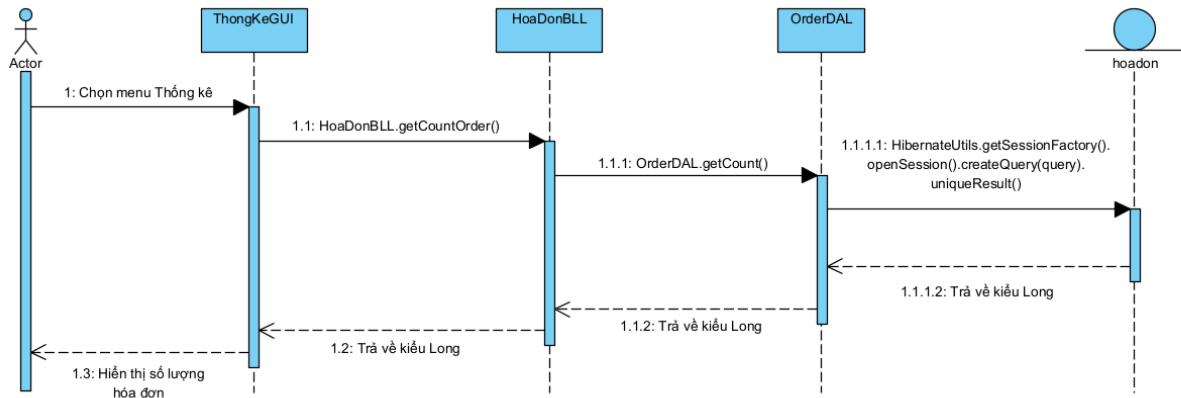
            ChartPanel chpn=new ChartPanel(chart);
            chpn.setPreferredSize(new Dimension(jp.getWidth(),450));

            jp.removeAll();
            jp.setLayout(new CardLayout());
            jp.add(chpn);
            jp.validate();
            jp.repaint();
        }
    }
}

```

4.4. Xử lý 3: Hiển thị số lượng đơn hàng

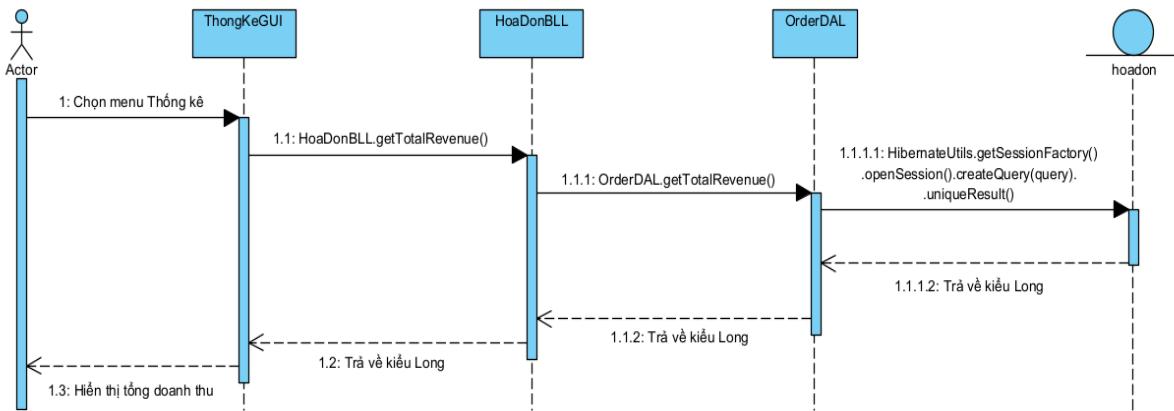
Sơ đồ tuần tự:



DAL	<pre> 47 48 public long getCount() { 49 Session session = factory.openSession(); 50 long amount = 0; 51 Transaction tx = null; 52 try { 53 tx = session.beginTransaction(); 54 Query query = session.createQuery("select count(*) from Order"); 55 amount = (long)query.uniqueResult(); 56 tx.commit(); 57 } catch (HibernateException e) { 58 if (tx != null) tx.rollback(); 59 e.printStackTrace(); 60 } finally { 61 session.close(); 62 } 63 } 64 }</pre>
BLL	<pre>] public long getCountOrder() { try { return dal.getCount(); } catch (Exception e) { e.printStackTrace(); } return 0; }</pre>
UI	<pre> public class ThongKeGUI extends JPanel{ private int DEFAULT_WIDTH; private JLabel lbDateStart,lbDateEnd; private JPanel panel; private ProductBLL pro_bll=new ProductBLL(); private CategoryBLL cate_bll=new CategoryBLL(); private OrderBLL or_bll=new OrderBLL(); private OrderBLL HD=new OrderBLL(); public ThongKeGUI(int width) { this.DEFAULT_WIDTH = width; init(); text_loai.setText(String.valueOf(cate_bll.getCountCategory())); text_sanpham.setText(String.valueOf(pro_bll.getCountProduct())); text_donhang.setText(String.valueOf(or_bll.getCountOrder())); text_doanhthu.setText(String.valueOf(or_bll.getTotalRevenue())+" VND"); setDataToChart(panelLineChart); } void setDataToChart(JPanel jp){ List<Order> listOrder=(ArrayList<Order>) HD.getListOrder(); if(listOrder!=null){ DefaultCategoryDataset dataset = new DefaultCategoryDataset(); for(Order item : listOrder){ dataset.addValue(item.getTotalPrice(),"Total",item.getCreatedDate()); } //create chart JFreeChart chart = ChartFactory.createLineChart("THỐNG KẾ DOANH THU","Thời Gian","Doanh Thu (VND)",dataset); ChartPanel chpn=new ChartPanel(chart); chpn.setPreferredSize(new Dimension(jp.getWidth(),450)); jp.removeAll(); jp.setLayout(new CardLayout()); jp.add(chpn); jp.validate(); jp.repaint(); } }</pre>

4.5. Xử lý 4: Hiển thị tổng số doanh thu

Sơ đồ tuân tự:

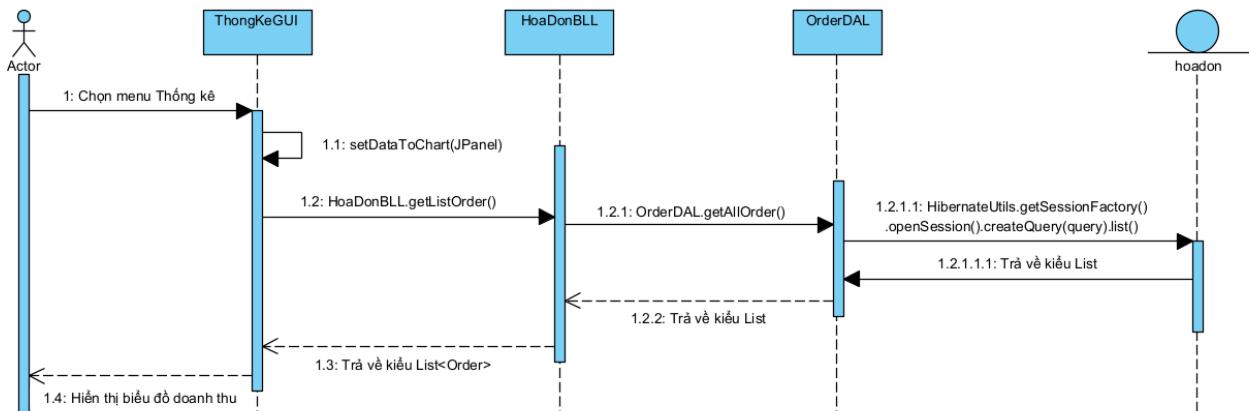


DAL	<pre> 65 66 [67 public int getTotalRevenue() { 68 Session session = factory.openSession(); 69 double total = 0; 70 Transaction tx = null; 71 try { 72 tx = session.beginTransaction(); 73 Query query = session.createQuery("SELECT sum(totalPrice) from Order"); 74 total = (double)query.uniqueResult(); 75 tx.commit(); 76 } catch (HibernateException e) { 77 if (tx != null) tx.rollback(); 78 e.printStackTrace(); 79 } finally { 80 session.close(); 81 } 82 } 83] </pre>
BLL	<pre> [] public long getTotalRevenue(){ try { return dal.getTotalRevenue(); } catch (Exception e) { e.printStackTrace(); } return 0; } </pre>

UI	<pre> public class ThongKeGUI extends JPanel{ private int DEFAULT_WIDTH; private JLabel lbDateStart,lbDateEnd; private JPanel panel; private ProductBLL pro_bll=new ProductBLL(); private CategoryBLL cate_bll=new CategoryBLL(); private OrderBLL or_bll=new OrderBLL(); private OrderBLL HD=new OrderBLL(); public ThongKeGUI(int width) { this.DEFAULT_WIDTH = width; init(); text_loai.setText(String.valueOf(cate_bll.getCountCategory())); text_sanpham.setText(String.valueOf(pro_bll.getCountProduct())); text_donhang.setText(String.valueOf(or_bll.getCountOrder())); text_doanhthu.setText(String.valueOf(or_bll.getTotalRevenue())+" VND"); setDataToChart(panelLineChart); } void setDataToChart(JPanel jp){ List<Order> listOrder=(ArrayList<Order>) HD.getListOrder(); if(listOrder!=null){ DefaultCategoryDataset dataset = new DefaultCategoryDataset(); for(Order item : listOrder){ dataset.addValue(item.getTotalPrice(),"Total",item.getCreatedDate()); } //create chart JFreeChart chart = ChartFactory.createLineChart("THÔNG KÉ DOANH THU","Thời Gian","Doanh Thu (VND)",dataset); ChartPanel chpn=new ChartPanel(chart); chpn.setPreferredSize(new Dimension(jp.getWidth(),450)); jp.removeAll(); jp.setLayout(new CardLayout()); jp.add(chpn); jp.validate(); jp.repaint(); } } } </pre>
----	--

4.6. Xử lý 5: Hiển thị biểu đồ doanh thu

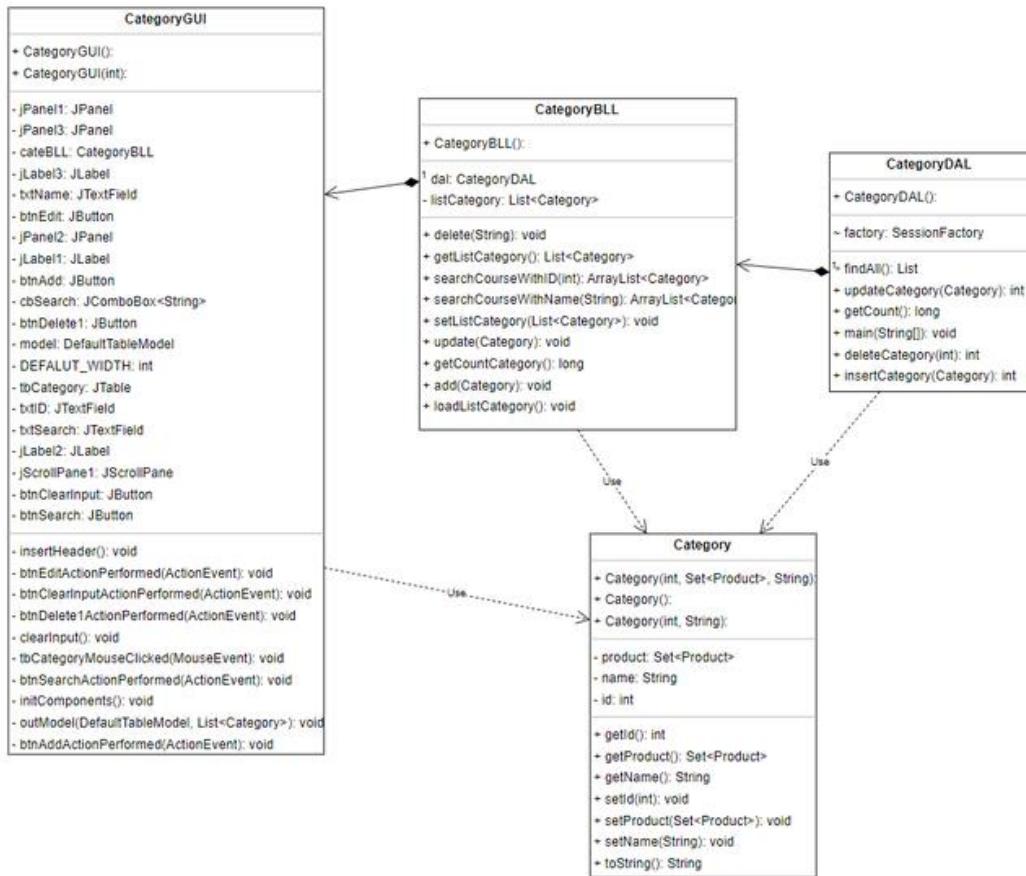
Sơ đồ tuần tự



DAL	<pre> 27 public List getAllOrder() { 28 Session session = factory.openSession(); 29 List listOrder = null; 30 Transaction tx = null; 31 try { 32 tx = session.beginTransaction(); 33 String hql = "FROM Order"; 34 listOrder = session.createQuery(hql).list(); 35 36 tx.commit(); 37 } catch (HibernateException e) { 38 if (tx != null) { 39 tx.rollback(); 39 } 39 e.printStackTrace(); 40 } finally { 41 session.close(); 42 } 43 return listOrder; 44 } 45 46 }</pre>
BLL	<pre>] public List<Order> getListOrder() { 1 2 try{ 3 return dal.getAllOrder(); 4 } 5 catch (Exception e) { 6 e.printStackTrace(); 7 } 8 return null; 9 } -</pre>
UI	<pre> public class ThongKeGUI extends JPanel{ private int DEFAULT_WIDTH; private JLabel lbDatestart,lbDateEnd; private JPanel panel; private ProductBLL pro_bll=new ProductBLL(); private CategoryBLL cate_bll=new CategoryBLL(); private OrderBLL or_bll=new OrderBLL(); private OrderBLL HD=new OrderBLL(); public ThongKeGUI(int width) { this.DEFAULT_WIDTH = width; init(); text_loai.setText(String.valueOf(cate_bll.getCountCategory())); text_sanpham.setText(String.valueOf(pro_bll.getCountProduct())); text_donhang.setText(String.valueOf(or_bll.getCountOrder())); text_doanhthu.setText(String.valueOf(or_bll.getTotalRevenue())+" VND"); setDataToChart(panelLineChart); } void setDataToChart(JPanel jp){ List<Order> listOrder=(ArrayList<Order>) HD.getListOrder(); if(listOrder!=null){ DefaultCategoryDataset dataset = new DefaultCategoryDataset(); for(Order item : listOrder){ dataset.addValue(item.getTotalPrice(),"Total",item.getCreatedDate()); } //create chart JFreeChart chart = ChartFactory.createLineChart("THỐNG KÊ DOANH THU","Thời Gian","Doanh Thu (VND)",dataset); ChartPanel chpn=new ChartPanel(chart); chpn.setPreferredSize(new Dimension(jp.getWidth(),450)); jp.removeAll(); jp.setLayout(new CardLayout()); jp.add(chpn); jp.validate(); jp.repaint(); } } }</pre>

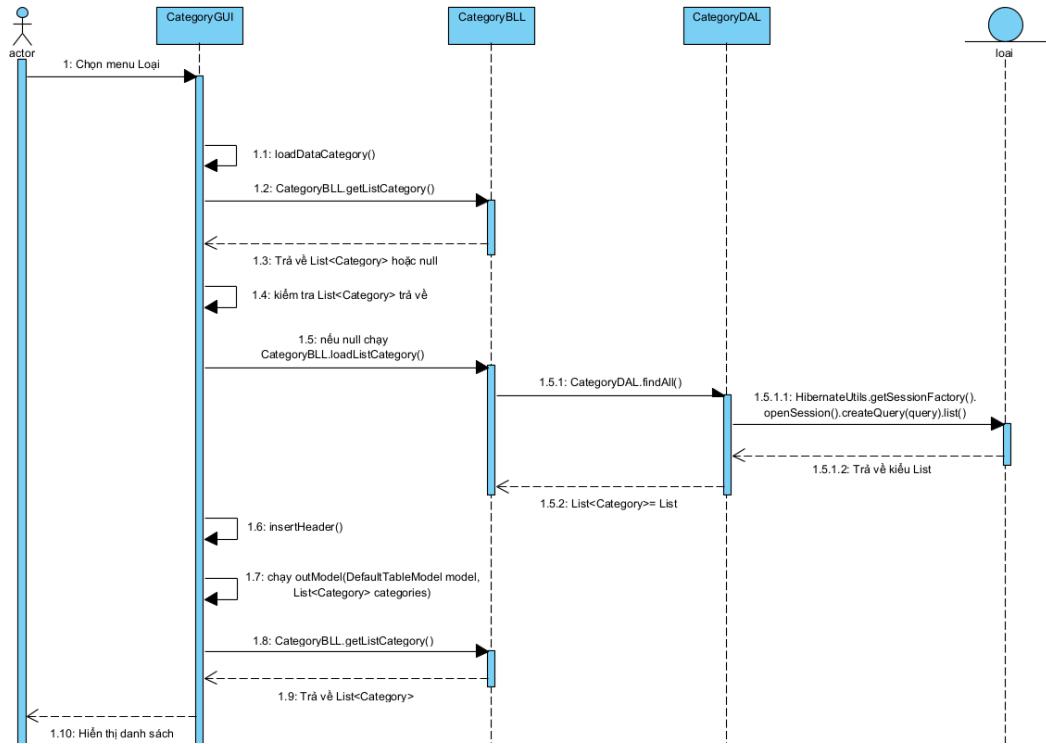
5. Chức năng quản lý loại

5.1. Sơ đồ class



5.2. Xử lý 1: Hiển thị danh sách loại

Sơ đồ tuần tự

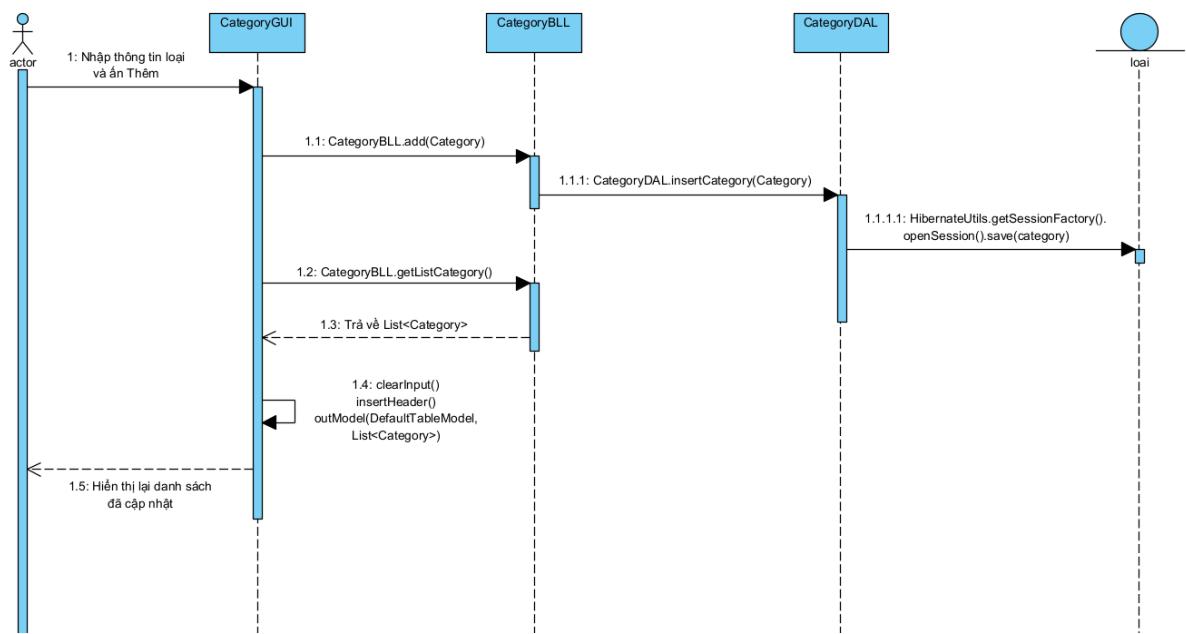


DAL	
	<pre> 17 public List findAll() { 18 Session session = factory.openSession(); 19 List listCategory = null; 20 Transaction tx = null; 21 try { 22 tx = session.beginTransaction(); 23 String hql = "FROM loai"; 24 listCategory = session.createQuery(hql).list(); 25 26 tx.commit(); 27 } catch (HibernateException e) { 28 if (tx != null) { 29 tx.rollback(); 30 } 31 e.printStackTrace(); 32 } finally { 33 session.close(); 34 } 35 return listCategory; 36 }</pre>

BLL	<pre> 29 public static List<Category> getListCategory() { 30 return listCategory; 31 } 32 33 public static void setListCategory(List<Category> listCategory) { 34 CategoryBLL.listCategory = listCategory; 35 } 36 37 public void loadListCategory() { 38 if (listCategory == null) { 39 listCategory = new ArrayList<Category>(); 40 } 41 listCategory=dal.findAll(); 42 } </pre>
UI	<pre> private void init() { loadDataCategory(); } private void loadDataCategory() { if (CategoryBLL.getListCategory() == null) cateBLL.loadListCategory(); insertHeader(); outModel(model, CategoryBLL.getListCategory()); } private void insertHeader() { Vector header = new Vector(); header.add("STT"); header.add("Mã Khoa Hoc"); model = new DefaultTableModel(header, 0); } private void outModel(DefaultTableModel model, List<Category> categories) // Xuất ra Table từ ArrayList { Vector data; model.setRowCount(0); for (Category cate : categories) { data = new Vector(); data.add(cate.getId()); data.add(cate.getName()); model.addRow(data); } tbCategory.setModel(model); } </pre>

5.3. Xử lý 2: Thêm Loại

Sơ đồ tuần tự



DAL	<pre> 38 public int insertCategory(Category category) { 39 Session session = factory.openSession(); 40 int result = 1; 41 Transaction tx = null; 42 try { 43 tx = session.beginTransaction(); 44 session.save(category); 45 tx.commit(); 46 } catch (HibernateException e) { 47 if (tx != null) { 48 tx.rollback(); 49 } 50 51 e.printStackTrace(); 52 return 0; 53 } finally { 54 session.close(); 55 } 56 57 return result; } </pre>
BLL	<pre> 64 public void add(Category category) { 65 try { 66 dal.insertCategory(category); 67 listCategory.add(category); 68 } catch (Exception e) { 69 System.out.println(e.getMessage()); 70 } 71 } </pre>

UI

```

private void btnAddActionPerformed(java.awt.event.ActionEvent evt) {
    try {
        String id = txtID.getText();
        String name = txtName.getText();
        Category category = new Category();
        category.setName(name);
        cateBLL.add(category);
        clearInput();
        insertHeader();
        outModel(model, CategoryBLL.getListCategory());
    } catch (Exception e) {
        JOptionPane.showMessageDialog(this, "Không thể thêm loại sản phẩm",
            "Thông báo lỗi", JOptionPane.ERROR_MESSAGE);
    }
}

private void insertHeader() {
    Vector header = new Vector();
    header.add("STT");
    header.add("Mã Khoa Học");
    model = new DefaultTableModel(header, 0);
}

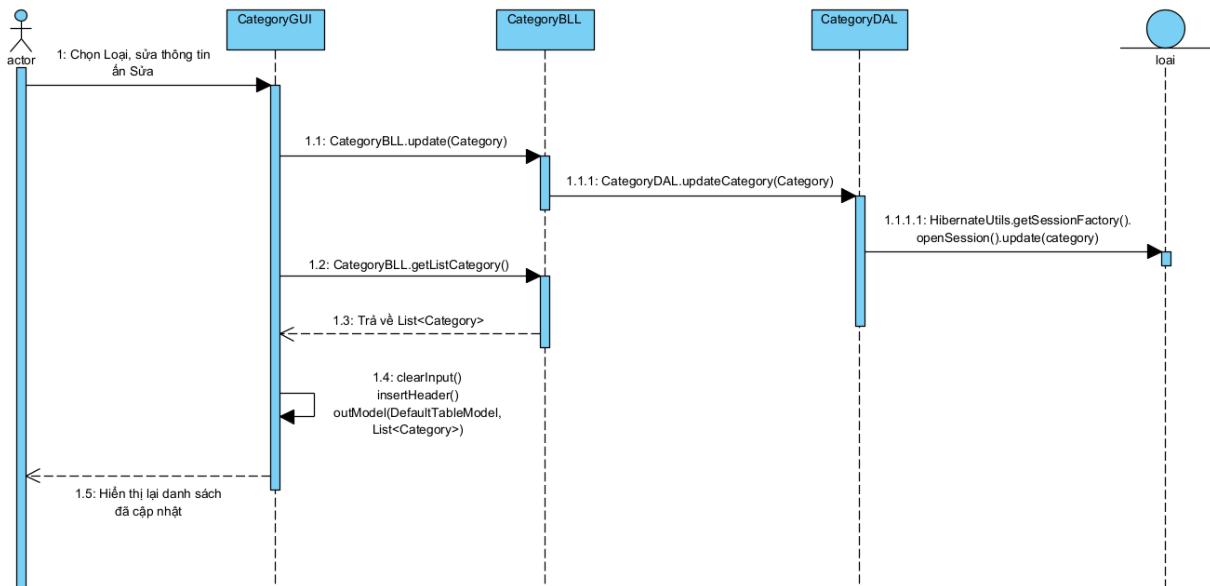
private void outModel(DefaultTableModel model, List<Category> categories) // Xuất ra Table từ ArrayList
{
    Vector data;
    model.setRowCount(0);
    for (Category cate : categories) {
        data = new Vector();
        data.add(cate.getId());
        data.add(cate.getName());
        model.addRow(data);
    }
    tbCategory.setModel(model);
}

private void clearInput() {
    txtName.setText("");
    txtID.setText("");
}

```

5.4. Xử lý 3: Sửa Loại

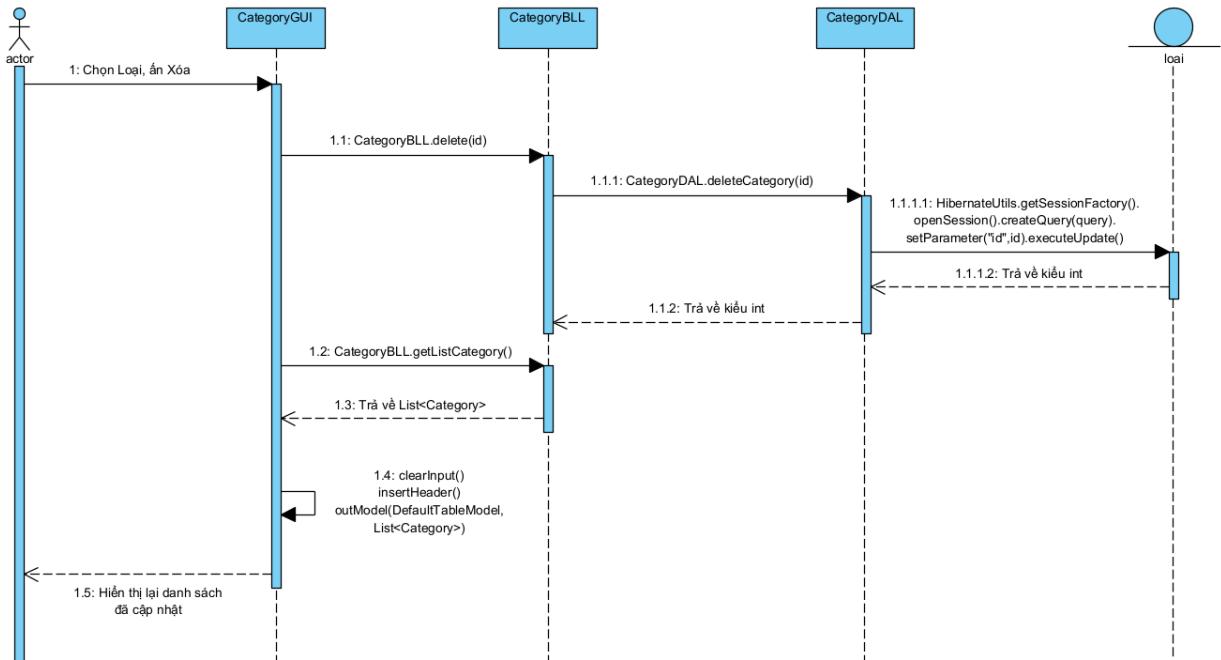
Sơ đồ tuần tự:



DAL	<pre> 59 public int updateCategory(Category category) { 60 Session session = factory.openSession(); 61 int result = 0; 62 Transaction tx = null; 63 try { 64 tx = session.beginTransaction(); 65 session.update(category); 66 tx.commit(); 67 } catch (HibernateException e) { 68 if (tx != null) { 69 tx.rollback(); 70 } 71 e.printStackTrace(); 72 } finally { 73 session.close(); 74 } 75 return result; 76 } </pre>
BLL	<pre> 88 public void update(Category ct) { 89 for (int i = 0; i < listCategory.size(); i++) { 90 if (listCategory.get(i).getId() == ct.getId()) { 91 try { 92 dal.updateCategory(ct); 93 listCategory.set(i, ct); 94 95 } catch (Exception e) { 96 e.printStackTrace(); 97 } 98 } 99 } 100 </pre>
UI	<pre> private void btnEditActionPerformed(java.awt.event.ActionEvent evt) { try { String id = txtID.getText(); String name = txtName.getText(); cateBLL.update(new Category(Integer.parseInt(id), name)); clearInput(); insertHeader(); outModel(model, CategoryBLL.getListCategory()); } catch (Exception e) { JOptionPane.showMessageDialog(this, "Không thể cập nhật loại sản phẩm", "Thông báo lỗi", JOptionPane.ERROR_MESSAGE); } } </pre>

5.5. Xử lý 4: Xoá Loại

Sơ đồ tuần tự:

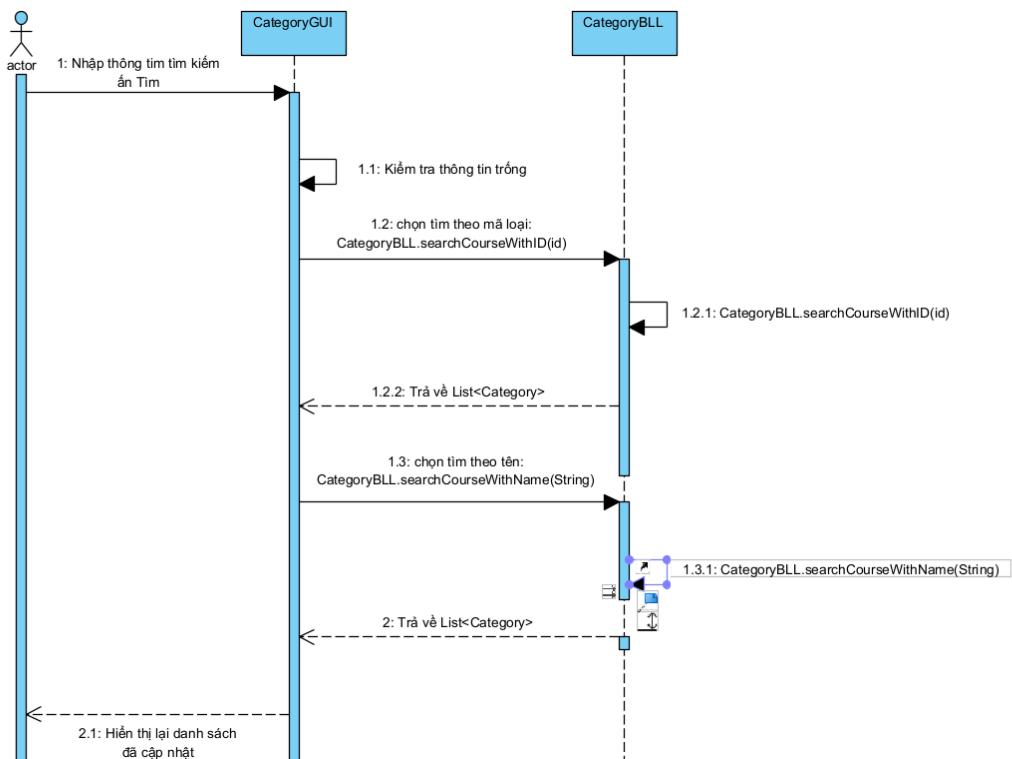


DAL	<pre> 78 public int deleteCategory(int id) { 79 Session session = factory.openSession(); 80 int result = 0; 81 Transaction tx = null; 82 try { 83 tx = session.beginTransaction(); 84 String hql = "DELETE FROM loai WHERE id= :id"; 85 Query query = session.createQuery(hql); 86 query.setParameter("id", id); 87 result = query.executeUpdate(); 88 // System.out.println("Rows affected: " + result); 89 90 tx.commit(); 91 } catch (HibernateException e) { 92 if (tx != null) { 93 tx.rollback(); 94 } 95 e.printStackTrace(); 96 } finally { 97 session.close(); 98 } 99 return result; 100 } </pre>
BLL	<pre> 73 public void delete(String id) { 74 75 int idCate = Integer.parseInt(id); 76 for(int i = 0 ; i < listCategory.size() ; i++){ 77 if (listCategory.get(i).getId() == idCate) { 78 try { 79 dal.deleteCategory(idCate); 80 listCategory.remove(i); 81 } catch (Exception e) { 82 e.printStackTrace(); 83 } 84 } 85 } 86 } </pre>

UI	<pre> 365 private void btnDeleteActionPerformed(java.awt.event.ActionEvent evt) { 366 int confirm = JOptionPane.showConfirmDialog(null, "Bạn Thực Sự Muốn Xóa Loại Sản Phẩm Này ?", 367 "Thông Báo", JOptionPane.YES_NO_OPTION); 368 if (confirm == 0) 369 try { 370 String id = txtID.getText(); 371 cateBLL.delete(id); 372 clearInput(); 373 insertHeader(); 374 outModel(model, CategoryBLL.getListCategory()); 375 } catch (Exception e) { 376 JOptionPane.showMessageDialog(this, "Không Thể Xóa Loại Sản Phẩm", 377 "Thông Báo Lỗi", JOptionPane.ERROR_MESSAGE); 378 } 379 } </pre>
----	--

5.6. Xử lý 5: Tìm kiếm loại

Sơ đồ tuần tự



BLL	<pre> 43 public ArrayList<Category> searchCourseWithID(int id) { 44 ArrayList<Category> search = new ArrayList<>(); 45 for (Category cs : listCategory) { 46 if (cs.getId() == id) { 47 search.add(cs); 48 } 49 } 50 return search; 51 } </pre>
-----	---

	<pre> 52 public ArrayList<Category> searchCourseWithName(String name) { 53 ArrayList<Category> search = new ArrayList<>(); 54 for (Category cs : listCategory) { 55 if (cs.getName().toLowerCase().contains(name.toLowerCase())) { 56 search.add(cs); 57 } 58 } 59 return search; 60 } </pre>
UI	<pre> 381 private void btnSearchActionPerformed(java.awt.event.ActionEvent evt) { 382 String searchType = cbSearch.getSelectedItem().toString(); 383 DefaultTableModel temp = new DefaultTableModel(); 384 ArrayList<Category> search = new ArrayList<>(); 385 Vector header = new Vector(); 386 header.add("Mã Loại"); 387 header.add("Tên Loại Sản Phẩm"); 388 try { 389 if (!txtSearch.getText().isEmpty()) { 390 String inputSearch = txtSearch.getText(); 391 switch (searchType) { 392 case "Mã Loại": 393 search= cateBLL.searchCourseWithID(Integer.parseInt(inputSearch)); 394 break; 395 case "Tên Loại": 396 search= cateBLL.searchCourseWithName(inputSearch); 397 break; 398 } 399 } else 400 JOptionPane.showMessageDialog(this, "Vui lòng nhập điều kiện tìm ", 401 "Thông Báo", JOptionPane.ERROR_MESSAGE); 402 } catch (Exception e) { 403 JOptionPane.showMessageDialog(this, "Không Thể Tìm Kiếm ", 404 "Thông Báo Lỗi", JOptionPane.ERROR_MESSAGE); 405 e.printStackTrace(); 406 } 407 if (search != null && search.size() > 0) { 408 if (temp.getRowCount() == 0) { 409 temp = new DefaultTableModel(header, 0); 410 } 411 for (int i = 0; i < search.size(); i++) { 412 Vector row = new Vector(); 413 row.add(search.get(i).getId()); 414 row.add(search.get(i).getName()); 415 temp.addRow(row); 416 } 417 } 418 } </pre>

Chương IV: SOURCE CODE KẾT NỐI CSDL, HƯỚNG DẪN CÀI ĐẶT CHƯƠNG TRÌNH VÀ LINK CHÚA SOURCE CODE ĐỒ ÁN

1. Hướng dẫn cài đặt chương trình

*** Các file source code nằm trong thư mục src/

1. Các thư viện mở rộng hỗ trợ nằm ở thư mục src/libs. (cần import đầy đủ các file .jar)
2. Tạo database “coffeestore” và import file “coffeestore.sql” trong folder “data” vào phpadmin trên XAMPP .
3. Import tất cả các thư viện trong thư mục /libs/ .
4. Mở IDE NetBeans (hoặc Eclipse) để import project .
5. Build project sau đó tiến hành Run application để sử dụng chương trình.

2. Link Chứa Source Code Đồ Án

1. Link source code: <https://github.com/minhtrung0110/manage-conffee-hibernate.git>