

HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG
KHOA CÔNG NGHỆ THÔNG TIN



BÀI TẬP LỚN LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG

Đề tài: Xây dựng ứng dụng quản lý quán cà phê

LỚP 01 – NHÓM 13

Tên thành viên	Mã số sinh viên
Lê Duy Quyết	B21DCCN642
Lê Ngọc Thảo Vân	B21DCCN786
Phạm Văn Tiến	B21DCCN708
Cao Hồng Đức	B21DCCN234
Nguyễn Thái Minh	B21DCCN090

Hà Nội, 2023

MỤC LỤC

I. GIỚI THIỆU.....	2
A. PHÁT BIỂU BÀI TOÁN VÀ CÁC CHỨC NĂNG DỰ ĐỊNH PHÁT TRIỂN	2
1. Bài toán	2
2. Các chức năng ứng dụng.....	2
B. KỸ THUẬT, CÔNG NGHỆ SỬ DỤNG.....	3
1. Phương pháp.....	3
2. Ngôn ngữ sử dụng.....	3
3. Công cụ.....	3
II. NỘI DUNG CHÍNH.....	4
A. XÂY DỰNG CƠ SỞ DỮ LIỆU	4
B. THIẾT KẾ CÁC PACKAGE CỦA ỨNG DỤNG	5
1. Package Util	5
2. Package Obj.....	5
3. Package DAO và DAO.impl	6
4. Package GUI.....	15
5. Package team13.quanlyquancaphe	51
III. TÀI LIỆU THAM KHẢO.....	51

I. Giới thiệu

A. Phát biểu bài toán và các chức năng dự định phát triển

1. Bài toán

Xây dựng một ứng dụng quản lý quán cà phê nhằm tối ưu hóa quá trình quản lý, giúp chủ quán và nhân viên thực hiện công việc một cách hiệu quả và thuận lợi. Ứng dụng này sẽ cung cấp các chức năng cơ bản để quản lý các hoạt động hàng ngày của quán cà phê.

2. Các chức năng ứng dụng

Hệ thống đăng nhập

- Cho phép nhân viên và quản lý đăng nhập vào hệ thống

Trang chủ

Danh mục

- Quản lý danh sách danh mục trong quán cà phê
- Cho phép thêm mới, sửa đổi, xóa, tìm kiếm các danh mục đối với quản lý, nhân viên chỉ có thể tìm kiếm.

Sản phẩm

- Quản lý thông tin chi tiết về sản phẩm: tên sản phẩm, thuộc danh mục nào, giá và trạng thái.
- Cho phép thêm mới, sửa đổi, xóa, tìm kiếm sản phẩm đối với quản lý, nhân viên chỉ có thể tìm kiếm.

Khu vực & Bàn ăn

- Cho phép quản lý các khu vực và bàn ăn trong quán cà phê đối với quản lý.
- Cho phép theo dõi trạng thái và đặt bàn đối với quản lý và nhân viên.

Hóa đơn

- Quản lý hóa đơn bán hàng
- Cho phép xem, xóa, tìm kiếm và xuất hóa đơn đối với quản lý, nhân viên không có chức năng xóa.

Người dùng

- Cho phép quản lý thông tin về người dùng đối với quản lý

B. Kỹ thuật, công nghệ sử dụng

1. Phương pháp

- Sử dụng các kiến thức về lập trình hướng đối tượng (OOP)
- Sử dụng Java Design Pattern: DAO Pattern

Ý tưởng là thay vì có logic giao tiếp trực tiếp với cơ sở dữ liệu, hệ thống file, dịch vụ web hoặc bất kỳ cơ chế lưu trữ nào mà ứng dụng cần sử dụng, chúng ta sẽ để logic này sẽ giao tiếp qua lớp trung gian DAO. Lớp DAO này sau đó giao tiếp với hệ thống lưu trữ, hệ quản trị CSDL như thực hiện các công việc liên quan đến lưu trữ và truy vấn dữ liệu (tìm kiếm, thêm, xóa, sửa,...).



2. Ngôn ngữ sử dụng

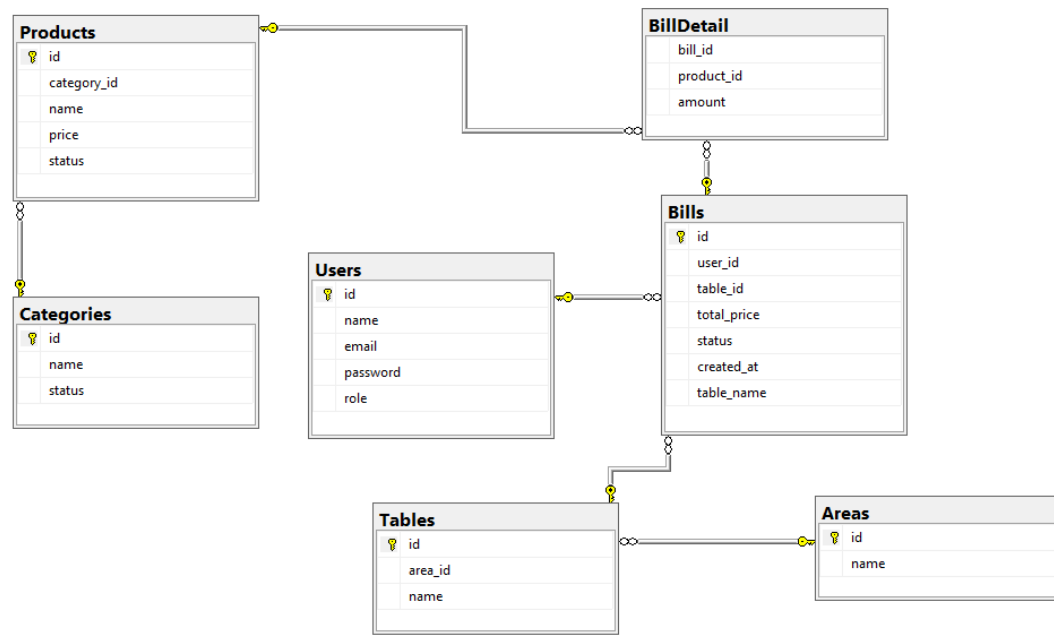
- Java và SQL

3. Công cụ

- Apache Netbeans IDE.
- SQL Server và SQL Server Management Studio (SSMS).
- Thư viện Java AWT, Swing và API JDBC.

II. Nội dung chính

A. Xây dựng cơ sở dữ liệu

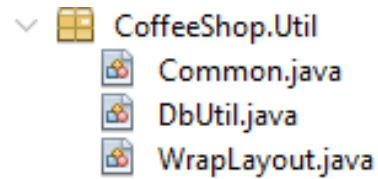


- Products bao gồm: id (khóa chính), category_id (khóa ngoại tham chiếu đến bảng Categories), name, price và status.
- Categories bao gồm: id (khóa chính), name và status.
- BillDetail bao gồm: bill_id (khóa chính), product_id (khóa ngoại tham chiếu đến bảng Products) và amount.
- Bills bao gồm: id (khóa chính), user_id(khóa ngoại tham chiếu đến bảng Users), table_id(khóa ngoại tham chiếu đến bảng Tables), total_price, status, creat_at và table_name.
- Users bao gồm: id (khóa chính), name, email, password và role.
- Tables bao gồm: id (khóa chính), area_id((khóa ngoại tham chiếu đến bảng Areas) và name.
- Areas bao gồm: id (khóa chính) và name.

B. Thiết kế các package của ứng dụng

1. Package Util

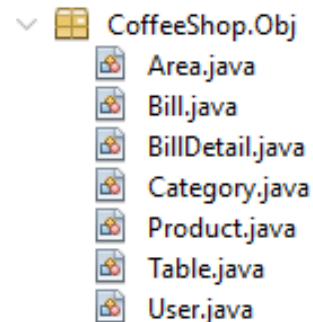
- Common: Class cung cấp 2 phương thức: Kiểm tra 1 đối tượng (String, List, Set hoặc Map) có null hoặc rỗng không và kiểm tra một chuỗi có thể chuyển thành số nguyên không.
- DbUtil: Thiết lập kết nối đến cơ sở dữ liệu SQL Server trong Java sử dụng JDBC, cung cấp phương thức getConnection() trả về một đối tượng Connection đến cơ sở dữ liệu.
- WrapLayout: Là 1 lớp kế thừa của FlowLayout trong Java Swing được thiết kế để hỗ trợ việc xếp các thành phần (components) trong GUI theo chiều ngang và hỗ trợ xuống dòng khi không đủ không gian để đặt tất cả thành phần trong 1 dòng duy nhất.



2. Package Obj

Định nghĩa các đối tượng trong ứng dụng và thuộc tính, phương thức tương ứng:

- Area: có 2 thuộc tính mã khu vực và tên khu vực, có hàm tạo, các phương thức get/ set, toString.
- Bill: có các thuộc tính: mã hoá đơn, mã người dùng, mã bàn, tổng tiền hoá đơn, trạng thái thanh toán, thời gian tạo, tên người dùng và tên bàn; có hàm tạo, các phương thức get/ set.
- BillDetail: có các thuộc tính: mã hoá đơn, mã hàng, tên hàng, số lượng, đơn giá của mỗi loại hàng; có hàm tạo, các phương thức get/ set.
- Category: có các thuộc tính: mã danh mục, tên danh mục và trạng thái hoạt động; có hàm tạo, các phương thức get/ set, toString.
- Product: có các thuộc tính: mã hàng hoá, tên hàng, đơn giá, trạng thái hoạt động, mã danh mục và tên danh mục; có hàm tạo, các phương thức get/ set.
- Table: có các thuộc tính: mã bàn, mã khu vực và tên bàn; có hàm tạo, các phương thức get/ set, toString.

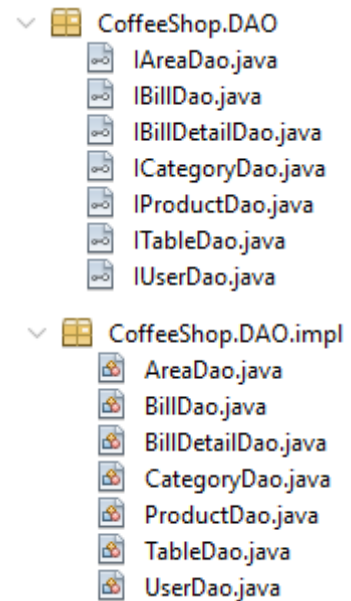


- User: có các thuộc tính: mã, tên, email, mật khẩu và vai trò (Admin/ Nhân viên) của người dùng; có hàm tạo, các phương thức get/ set.

3. Package DAO và DAO.impl

Package DAO gồm các interfaces để định nghĩa các phương thức trừu tượng triển khai truy cập dữ liệu cho từng đối tượng trong ứng dụng.

Package DAO.impl gồm các lớp triển khai chi tiết các phương thức của từng interface được định nghĩa trong DAO, các lớp này sẽ thao tác trực tiếp với cơ sở dữ liệu.



a) Interface IAreaDao và Class AreaDao

Tương tác với cơ sở dữ liệu về khu vực.

```
public interface IAreaDao {

    public int count();

    public List<Area> getAll();

    public Map<String, Object> create(Area area);

    public Area findByName(String name);

    public Map<String, Object> update(Area area);

    public Map<String, Object> delete(int id);

}
```

Kết nối AreaDao với CSDL

```
private Connection conn = null;
private CallableStatement cs = null;
private ResultSet rs = null;

public AreaDao(DbUtil dbUtil) {
    conn = dbUtil.getConnection();
}
```

Các phương thức:

```
public class AreaDao implements IAreaDao {

    private Connection conn = null;
    private CallableStatement cs = null;
    private ResultSet rs = null;

    public AreaDao(DbUtil dbUtil) {...3 lines }

    @Override
    public int count() {...19 lines }

    @Override
    public ArrayList<Area> getAll() {...25 lines }

    @Override
    public Map<String, Object> create(Area area) {...22 lines }

    @Override
    public Area findByName(String name) {...25 lines }

    @Override
    public Map<String, Object> update(Area area) {...23 lines }

    @Override
    public Map<String, Object> delete(int id) {...22 lines }

}
```

- Hàm count(): Đếm số khu vực
- Hàm getAll(): lấy dữ liệu của tất cả các khu vực trong database
- Hàm create(): Tạo mới khu vực có các thuộc tính tương đương với đối tượng area truyền vào.
- Hàm findByName(): Tìm khu vực bằng tên (người dùng nhập tên)
- Hàm update(): Cập nhật khu vực thành các thuộc tính tương ứng với đối tượng area truyền vào.
- Hàm delete(): Xóa khu vực dựa trên mã khu vực.

b) Interface *IBillDao* và Class *BillDao*

Tương tác với cơ sở dữ liệu về hoá đơn.

```
public interface IBillDao {

    public int count();

    public List<Bill> getAll(Bill bill);

    public Map<String, Object> create(Bill bill);

    public Map<String, Object> update(Bill bill);

    public Map<String, Object> delete(int id);

    public Bill getByTableId(Bill bill);

}
```

Khai báo các phương thức trong BillDao:

- Constructor BillDao: Phương thức khởi tạo lớp BillDao nhận vào DbUtil để kết nối với cơ sở dữ liệu

```
public class BillDao implements IBillDao {

    private Connection conn = null;
    private CallableStatement cs = null;
    private ResultSet rs = null;

    public BillDao(DbUtil dbUtil) {
        conn = dbUtil.getConnection();
    }

}
```

```

public class BillDao implements IBillDao {

    private Connection conn = null;
    private CallableStatement cs = null;
    private ResultSet rs = null;

    public BillDao(DbUtil dbUtil) {...3 lines }

    @Override
    public int count() {...19 lines }

    @Override
    public List<Bill> getAll(Bill bill, String dateStart, String dateEnd) {...64 lines }

    @Override
    public Map<String, Object> create(Bill bill) {...40 lines }

    @Override
    public Map<String, Object> update(Bill bill) {...26 lines }

    @Override
    public Map<String, Object> delete(int id) {...21 lines }

    @Override
    public Bill getByTableId(Bill bill) {...35 lines }

}

```

- Phương thức getAll(): Phương thức được sử dụng để lấy danh sách các hoá đơn từ cơ sở dữ liệu dựa trên các thuộc tính trùng với đối tượng bill được đưa vào và thời gian truy xuất.
- Phương thức getByTableId(): Phương thức dùng để đọc dữ liệu của một hóa đơn có table_id trùng với tham số được đưa vào

c) Interface IBillDetailDao và Class BillDetailDao

Tương tác với cơ sở dữ liệu về dữ liệu chi tiết của các hoá đơn.

```

public interface IBillDetailDao {

    public List<BillDetail> getAll(int bill_id);

    public Map<String, Object> create(BillDetail billDetail);

    public Map<String, Object> update(BillDetail billDetail);

    public Map<String, Object> delete(BillDetail billDetail);

}

```

Khai báo các phương thức trong BillDetailDao:

```
public class BillDetailDao implements IBillDetailDao {

    private Connection conn = null;
    private CallableStatement cs = null;
    private ResultSet rs = null;

    public BillDetailDao(DbUtil dbUtil) {...3 lines }

    @Override
    public ArrayList<BillDetail> getAll(int bill_id) {...29 lines }

    @Override
    public Map<String, Object> create(BillDetail billDetail) {...23 lines }

    @Override
    public Map<String, Object> update(BillDetail billDetail) {...23 lines }

    @Override
    public Map<String, Object> delete(BillDetail billDetail) {...23 lines }
}
```

- Phương thức getAll(): Phương thức được sử dụng để lấy dữ liệu chi tiết hoá đơn từ cơ sở dữ liệu dựa trên mã hoá đơn
- Phương thức create(): Phương thức được sử dụng để dữ liệu chi tiết vào 1 hoá đơn (ở đây là thêm sản phẩm vào hoá đơn đang đặt)
- Phương thức update(): Phương thức được sử dụng để cập nhật thông tin chi tiết của 1 hoá đơn (ở đây là thay đổi số lượng của 1 sản phẩm trong hoá đơn đang đặt)
- Phương thức delete(): Phương thức được sử dụng để xóa thông tin chi tiết của 1 hoá đơn (ở đây là xóa 1 sản phẩm trong hoá đơn đang đặt)

d) Interface ICategoryDao và Class CategoryDao

Tương tác với cơ sở dữ liệu về danh mục các sản phẩm của quán cà phê.

```
package CoffeeShop.DAO;

import CoffeeShop.Obj.Category;

import java.util.List;
import java.util.Map;

public interface ICategoryDao {

    public int count();

    public List<Category> getAll(Category category);

    public Map<String, Object> create(Category category);

    public Map<String, Object> update(Category category);

    public Map<String, Object> delete(int id);
}
```

Các phương thức trong CategoryDao:

```
public class CategoryDao implements ICategoryDao {

    private Connection conn = null;
    private CallableStatement cs = null;
    private ResultSet rs = null;

    public CategoryDao(DbUtil dbUtil) {...3 lines }

    @Override
    public int count() {...19 lines }

    @Override
    public List<Category> getAll(Category category) {...36 lines }

    @Override
    public Map<String, Object> create(Category category) {...23 lines }

    @Override
    public Map<String, Object> update(Category category) {...24 lines }

    @Override
    public Map<String, Object> delete(int id) {...22 lines }
}
```

e) Interface IProductDao và Class ProductDao

Tương tác với cơ sở dữ liệu về sản phẩm.

```
public interface IProductDao {

    public int count();

    public List<Product> getAll(Product product, Integer fromPrice, Integer toPrice);

    public Map<String, Object> create(Product product);

    public Map<String, Object> update(Product product);

    public Map<String, Object> delete(int id);

}
```

Các phương thức trong ProductDao:

```
public class ProductDao implements IProductDao {

    private Connection conn = null;
    private CallableStatement cs = null;
    private ResultSet rs = null;

    public ProductDao(DbUtil dbUtil) {...3 lines }

    @Override
    public int count() {...19 lines }

    @Override
    public List<Product> getAll(Product product, Integer fromPrice, Integer toPrice) {...53 lines }

    @Override
    public Map<String, Object> create(Product product) {...25 lines }

    @Override
    public Map<String, Object> update(Product product) {...26 lines }

    @Override
    public Map<String, Object> delete(int id) {...22 lines }

}
```

Hàm getAll(): Lấy sản phẩm có trong danh mục với điều kiện lấy tương ứng các tham số truyền vào (thuộc tính của product và khoảng giá).

f) Interface ITableDao và Class TableDao

Tương tác với cơ sở dữ liệu về bàn trong quán.

```
package CoffeeShop.DAO;

import CoffeeShop.Obj.Table;

import java.util.List;
import java.util.Map;

public interface ITableDao {

    public int count();

    public List<Table> getAll(Table table);

    public Table findByName(String name);

    public Map<String, Object> create(Table table);

    public Map<String, Object> update(Table table);

    public Map<String, Object> delete(int id);
}
```

Các phương thức trong TableDao:

```
public class TableDao implements ITableDao {

    private Connection conn = null;
    private CallableStatement cs = null;
    private ResultSet rs = null;

    public TableDao(DbUtil dbUtil) {...3 lines }

    @Override
    public int count() {...19 lines }

    @Override
    public ArrayList<Table> getAll(Table table) {...42 lines }

    @Override
    public Map<String, Object> create(Table table) {...23 lines }

    @Override
    public Map<String, Object> update(Table table) {...25 lines }

    @Override
    public Map<String, Object> delete(int id) {...22 lines }

    @Override
    public Table findByName(String name) {...32 lines }
}
```

g) Interface IUserDao và Class UserDao

Tương tác với cơ sở dữ liệu về người dùng (gồm nhân viên và quản lý (admin)).

```
package CoffeeShop.DAO;

import CoffeeShop.Obj.User;
import java.util.List;
import java.util.Map;

public interface IUserDao {
    public int count();
    public List<User> getAll(User user);
    public Map<String, Object> create(User user);
    public Map<String, Object> update(User user);
    public Map<String, Object> delete(int id);
    public User auth(String email, String password);
}

package CoffeeShop.DAO.impl;

import CoffeeShop.DAO.IUserDao;
import CoffeeShop.Obj.User;
import CoffeeShop.Util.*;

import java.sql.*;
import java.util.ArrayList;
import java.util.HashMap;
import java.util.List;
import java.util.Map;

public class UserDao implements IUserDao {
    private Connection conn = null;
    private CallableStatement cs = null;
    private ResultSet rs = null;

    public UserDao(DbUtil dbUtil) {
        conn = dbUtil.getConnection();
    }

    @Override
    public int count() { ...19 lines }

    @Override
    public List<User> getAll(User user) { ...44 lines }

    @Override
    public Map<String, Object> create(User user) { ...25 lines }

    @Override
    public Map<String, Object> update(User user) { ...26 lines }

    @Override
    public Map<String, Object> delete(int id) { ...22 lines }

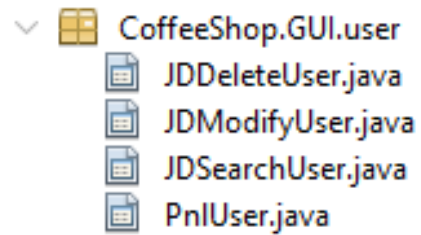
    @Override
    public User auth(String email, String password) { ...29 lines }
}
```

- auth(String email, String password): Kiểm tra đăng nhập bằng cách gọi procedure “sp_checkUser”. Trả về một đối tượng User nếu đăng nhập thành công (có email và password trong cơ sở dữ liệu), ngược lại trả về null.

4. Package GUI

a) Package GUI.user

Gồm 1 Jpanel PnlUser và 3 Jdialog JDDeleteUser, JDModifyUser và JDSearchUser định nghĩa giao diện và xử lý tương tác của người dùng.



(1) JDDeleteUser

Chức năng: Định nghĩa một cửa sổ dialog cho việc xác nhận và xóa một người dùng, bao gồm các phương thức như:

```
package CoffeeShop.GUI.user;

import CoffeeShop.DAO.impl.UserDao;
import CoffeeShop.Obj.User;
import CoffeeShop.Util.DbUtil;

import javax.swing.*;
import java.util.Map;

public class JDDeleteUser extends javax.swing.JDialog {
    private CallbackUserDelete callback;
    private User user;
    private UserDao userDao;

    interface CallbackUserDelete {
        public void actionUserDelete();
    }

    public JDDeleteUser(java.awt.Frame parent, boolean modal, DbUtil dbUtil, CallbackUserDelete callback, User user) {...10 lines }

    @SuppressWarnings("unchecked")
    Generated Code

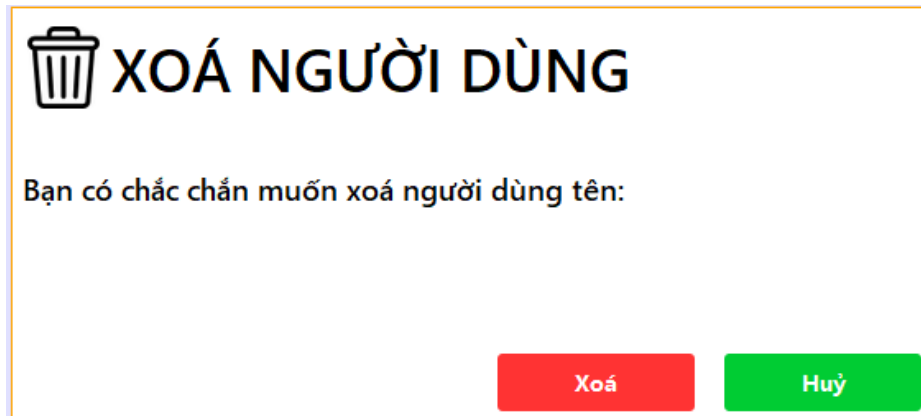
    private void btnDeleteActionPerformed(java.awt.event.ActionEvent evt) {...12 lines }

    private void btnCancelActionPerformed(java.awt.event.ActionEvent evt) {...3 lines }

    // Variables declaration - do not modify
    private javax.swing.JButton btnCancel;
    private javax.swing.JButton btnDelete;
    private javax.swing.JLabel jLabel1;
    private javax.swing.JPanel jPanel1;
    private javax.swing.JLabel lblConfirm;
    // End of variables declaration
}
```

- Constructor JDDeleteUser: Nhận một Frame cha, giá trị boolean cho biết cửa sổ có là modal hay không, một đối tượng DbUtil để tạo kết nối đến cơ sở dữ liệu, một callback để xử lý sự kiện sau khi xóa người dùng, và thông tin về người dùng cần xóa.
- btnDeleteActionPerformed: Xử lý sự kiện khi nút "Xóa" được nhấn. Gọi phương thức delete UserDao để thực hiện xóa người dùng. Hiển thị thông báo và gọi callback nếu xóa thành công.
- btnCancelActionPerformed: Xử lý sự kiện khi nút "Huỷ" được nhấn. Thực hiện đóng cửa sổ dialog.

Giao diện:




(2) *JDModifyUser*

Chức năng: định nghĩa một cửa sổ dialog cho phép thêm mới hoặc sửa đổi thông tin người dùng, bao gồm các phương thức như:

- Constructor *JDModifyUser*: Thiết lập giao diện và điều chỉnh hiển thị dựa trên việc có đối tượng user hay không, nếu có thì tức đây là sửa đổi thông tin người dùng, nếu là null sẽ là thêm mới người dùng.
- *loadingData*: Nạp dữ liệu người dùng vào trường dữ liệu nếu đây là sửa đổi thông tin.
- *btnModifyActionPerformed*: Xử lý sự kiện khi người dùng nhấn “Thêm mới” hoặc “Sửa đổi”. Kiểm tra các dữ liệu đầu vào (tên, email, password) và báo lỗi nếu có. Nếu dữ liệu hợp lệ, tiến hành tạo 1 đối tượng User mới và thực hiện *create()* hoặc *update()* thông qua *UserDao*. Cuối cùng là hiển thị thông báo kết quả và gọi callback để cập nhật lại.

Giao diện:


THÊM MỚI NGƯỜI DÙNG

Tên người dùng

Không được để trống

Email


Không được để trống

Mật khẩu

Không được để trống

Quyền

Thêm mới


SỬA ĐỔI NGƯỜI DÙNG

Tên người dùng

Email

Mật khẩu


Quyền

Sửa đổi

(3) *JDSearchUser*

Chức năng: định nghĩa một cửa sổ dialog cho phép tìm kiếm thông tin người dùng dựa trên tên, email và quyền.

Giao diện:


TÌM KIẾM NGƯỜI DÙNG

Tên người dùng

Email

Email bạn nhập không đúng định dạng

Quyền

Tìm kiếm

(4) *PnlUser*

Chức năng: Jpanel hiển thị và quản lý thông tin người dùng, bao gồm các phương thức như:

```

public class PnlUser extends javax.swing.JPanel implements JDialogModifyUser.CallbackUserModify, JDialogDeleteUser.CallbackUserDelete, JDialogSearchUser.CallbackUserSearch {
    private Frame parent;
    private List<User> users = new ArrayList<>();
    private User user;
    private User currentUser;
    private UserDao userDao;
    private DbUtil dbUtil;

    public PnlUser(Frame parent, DbUtil dbUtil, User currentUser) {...8 lines }

    public void loading(User newUser) {...23 lines }

    /** This method is called from within the constructor to initialize the form ...5 lines */
    @SuppressWarnings("unchecked")
    Generated Code

    private void lblAddMouseClicked(java.awt.event.MouseEvent evt) {...4 lines }

    private void lblUpdateMouseClicked(java.awt.event.MouseEvent evt) {...6 lines }

    private void lblSearchMouseClicked(java.awt.event.MouseEvent evt) {...4 lines }

    private void lblDeleteMouseClicked(java.awt.event.MouseEvent evt) {...10 lines }

    private void lblRefreshMouseClicked(java.awt.event.MouseEvent evt) {
        loading(newUser: null);
    }

    // Variables declaration - do not modify
    private javax.swing.JPanel jPanel1;
    private javax.swing.JPanel jPanel2;
    private javax.swing.JPanel jPanel3;
    private javax.swing.JScrollPane jScrollPane1;
    private javax.swing.JTable jTable1;
    private javax.swing.JLabel lblAdd;
    private javax.swing.JLabel lblDelete;
    private javax.swing.JLabel lblRefresh;
    private javax.swing.JLabel lblSearch;
    private javax.swing.JLabel lblUpdate;
    private javax.swing.JTable tblUser;
    // End of variables declaration

    @Override
    public void actionUserModify() {...3 lines }

    @Override
    public void actionUserDelete() {...3 lines }

    @Override
    public void actionUserSearch(User user) {...3 lines }
}

```

- Constructor: Khởi tạo một đối tượng PnlUser làm giao diện hiển thị danh sách người dùng. Gọi hàm loading(null) để hiển thị danh sách người dùng khi panel được khởi tạo.
- loading: Hiển thị danh sách người dùng sử dụng getAll() trong UserDao. Tham số truyền vào newUser nếu khác null sẽ hiển thị kết quả tìm kiếm.
- Các phương thức lblAddMouseClicked(), lblUpdateMouseClicked(), lblSearchMouseClicked() và lblDeleteMouseClicked(): Xử lý sự kiện khi người dùng nhấn thêm, cập nhật, sửa và xóa. Gọi đến các Jdialog tương ứng để thực hiện.
- actionUserModify(), actionUserDelete(), actionUserSearch(): Ghi đè các phương thức callback từ các dialog (JDialogModifyUser, JDialogDeleteUser, JDialogSearchUser) để cập nhật lại danh sách người dùng sau khi thực hiện thêm mới, sửa đổi, xóa, hoặc tìm kiếm.

b) Package GUI.home

(1) Dashboard

Quản lý giao diện và chuyển hướng giữa các chức năng của ứng dụng quản lý quán cà phê.

```
public final class Dashboard extends javax.swing.JFrame implements JDLogin.CallbackLogin {
    private User user;
    private JDLogin jdLogin;
    private DbUtil dbUtil;
```

Khai báo các phương thức:

```
public Dashboard(DbUtil dbUtil) {...7 lines }

public void loadUser(User user) {...18 lines }

/** This method is called from within the constructor to initialize the form ...5 lines */
@SuppressWarnings("unchecked")
Generated Code

private void btnCategoryActionPerformed(java.awt.event.ActionEvent evt) {...15 lines }

private void btnProductActionPerformed(java.awt.event.ActionEvent evt) {...15 lines }

private void btnBillActionPerformed(java.awt.event.ActionEvent evt) {...15 lines }

private void btnDashboardActionPerformed(java.awt.event.ActionEvent evt) {...15 lines }

private void btnAreaActionPerformed(java.awt.event.ActionEvent evt) {...15 lines }

private void btnUserActionPerformed(java.awt.event.ActionEvent evt) {...15 lines }

private void btnLogOutActionPerformed(java.awt.event.ActionEvent evt) {...4 lines }

// Variables declaration - do not modify
private javax.swing.JButton btnArea;
private javax.swing.JButton btnBill;
private javax.swing.JButton btnCategory;
private javax.swing.JButton btnDashboard;
private static javax.swing.JButton btnLogOut;
private javax.swing.JButton btnProduct;
private javax.swing.JButton btnUser;
private javax.swing.JLabel jLabel16;
private javax.swing.JLabel lblArea;
private javax.swing.JLabel lblBill;
private javax.swing.JLabel lblCategory;
private javax.swing.JLabel lblDashboard;
private javax.swing.JLabel lblEmail;
private javax.swing.JLabel lblName;
private javax.swing.JLabel lblProduct;
private javax.swing.JLabel lblUser;
private javax.swing.JPanel pnlBody;
private javax.swing.JPanel pnlMenu;
// End of variables declaration

@Override
public void actionLogin(User user) {...3 lines }
```

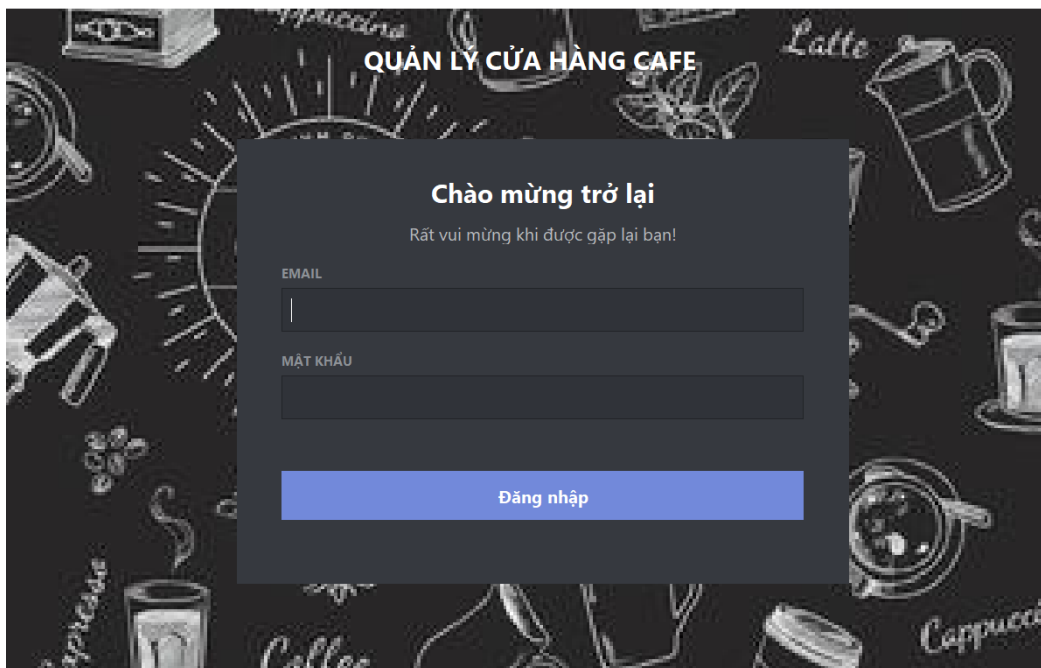
- Phương thức `loadUser()`: Phương thức này dùng để cập nhật thông tin người dùng khi đăng nhập thành công
- Phương thức `btnCategoryActionPerformed()`: Phương thức dùng để hiển thị danh mục trên giao diện và thay đổi màu nền của các nhãn để đánh dấu rằng người dùng đang xem “Danh mục”.
- Phương thức `btProductActionPerformed()`: Phương thức dùng để hiển thị danh sách sản phẩm trên giao diện và thay đổi màu nền của các nhãn để đánh dấu rằng người dùng đang xem “Sản phẩm”.
- Phương thức `btnBillActionPerformed()`: Phương thức dùng để hiển thị hóa đơn trên giao diện và thay đổi màu nền của các nhãn để đánh dấu rằng người dùng đang xem “Hóa đơn”.
- Phương thức `btnDashboardActionPerformed()`: Phương thức dùng để hiển thị trang chủ trên giao diện và thay đổi màu nền của các nhãn để đánh dấu rằng người dùng đang xem “Trang chủ”.
- Phương thức `btnAreaActionPerformed()`: Phương thức dùng để hiển thị nội dung quản lý khu vực & bàn ăn và thay đổi màu nền của các nhãn để đánh dấu rằng người dùng đang xem “Khu vực & bàn ăn”.
- Phương thức `btnLogOutActionPerformed()`: ẩn cửa sổ chính của ứng dụng và hiển thị cửa sổ đăng nhập khi nhấn vào “Đăng xuất”.
- Phương thức `btnUserActionPerformed()`: Phương thức dùng để hiển thị trang quản lý người dùng và thay đổi màu nền của các nhãn để đánh dấu rằng người dùng đang xem mục “Người dùng”.

(2) *JDLogin*

Hiển thị cửa sổ đăng nhập

ĐĂNG NHẬP TÀI KHOẢN

✕



```

public class JDLogin extends javax.swing.JDialog {

    private CallbackLogin callback;
    private Frame parent;
    private UserDao userDao;
    private DbUtil dbUtil;

    interface CallbackLogin {...3 lines }

    public JDLogin(java.awt.Frame parent, boolean modal, DbUtil dbUtil, CallbackLogin callback) {...26 lines }

    /** This method is called from within the constructor to initialize the form ...5 lines */
    @SuppressWarnings("unchecked")
    Generated Code

    private void txtEmailKeyPressed(java.awt.event.KeyEvent evt) {...5 lines }

    private void txtPasswordKeyPressed(java.awt.event.KeyEvent evt) {...5 lines }

    private void btnLoginActionPerformed(java.awt.event.ActionEvent evt) {...70 lines }

    private void formWindowClosed(java.awt.event.WindowEvent evt) {...8 lines }

    // Variables declaration - do not modify
    private javax.swing.JPanel boxLogin;
    private javax.swing.JButton btnLogin;
    private javax.swing.JLabel lblBackground;
    private javax.swing.JLabel lblEmail;
    private javax.swing.JLabel lblEmailError;
    private javax.swing.JLabel lblHeaderPrimary;
    private javax.swing.JLabel lblHeaderSecondary;
    private javax.swing.JLabel lblPassword;
    private javax.swing.JLabel lblPasswordError;
    private javax.swing.JLabel lblTitle;
    private javax.swing.JPanel panelBackground;
    private javax.swing.JPanel panelLogin;
    private javax.swing.JTextField txtEmail;
    private javax.swing.JPasswordField txtPassword;
    // End of variables declaration
}

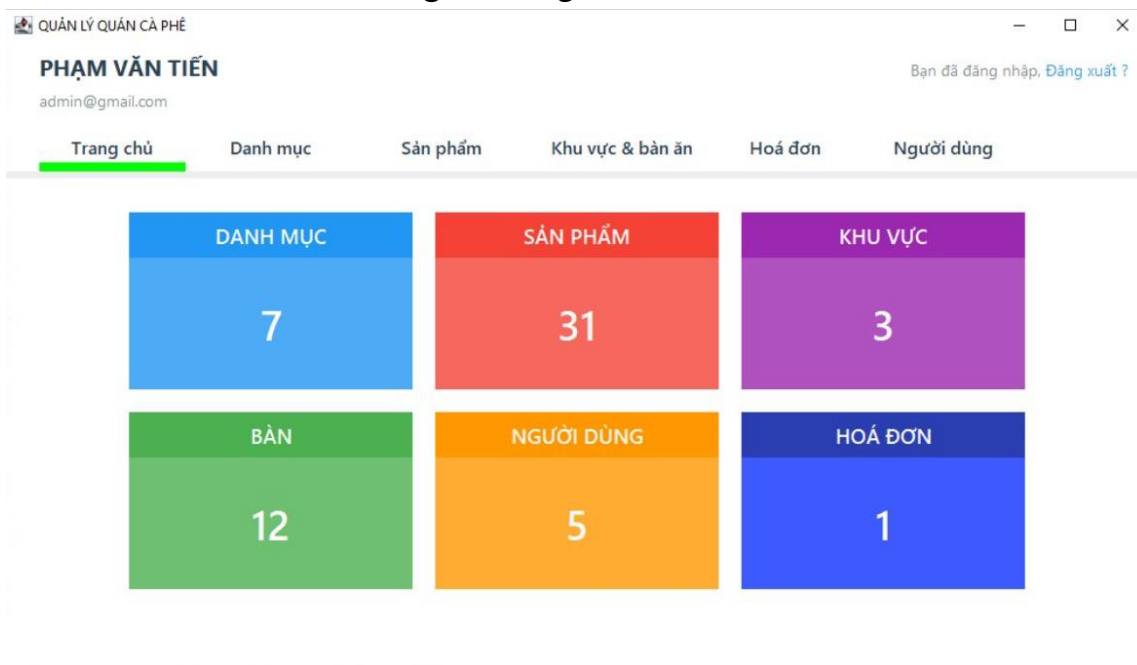
```

Khai báo các phương thức:

- Constructor JDLogin: phương thức khởi tạo của lớp JDLogin nhận các tham số parent, modal, dbUtil, callback để khởi tạo và đặt vị trí các thành phần giao diện.
- Phương thức txtEmailKeyPressed() và txtPasswordKeyPressed(): khi người dùng nhấn phím Enter khi đang nhập dữ liệu vào ô Email hoặc Mật khẩu thì sẽ thực hiện hành động đăng nhập.
- Phương thức btnLoginActionPerformed(): Lấy thông tin email và mật khẩu từ các trường văn bản và kiểm tra tính hợp lệ của chúng. Nếu thông tin không hợp lệ thì hiển thị thông báo lỗi và thay đổi màu sắc của khung văn bản. Nếu thông tin hợp lệ thì sử dụng “userDao” để xác thực thông tin đăng nhập. Nếu thông tin chính xác, thực hiện hành động đăng nhập thông qua “callback” và ẩn cửa sổ hiện tại, hiển thị cửa sổ chính và xóa thông tin đăng nhập đã nhập.
- Phương thức formWindowClosed(): khi đóng cửa sổ, phương thức này sẽ được gọi để đóng kết nối đến cơ sở dữ liệu và kết thúc chương trình.

(3) PnlHome

Hiển thị tổng quan số lượng về các mục khác nhau: Danh mục, Sản phẩm, Khu vực & bàn ăn, Hóa đơn, Người dùng



```

public class PnlHome extends javax.swing.JPanel {
    private Frame parent;
    private DbUtil dbUtil;

    private UserDao userDao;
    private CategoryDao categoryDao;
    private ProductDao productDao;
    private AreaDao areaDao;
    private TableDao tableDao;
    private BillDao billDao;

    public PnlHome(Frame parent, DbUtil dbUtil) {...14 lines }

    private void loading() {...8 lines }

    /** This method is called from within the constructor to initialize the form ...5 lines */
    @SuppressWarnings("unchecked")
    Generated Code

    // Variables declaration - do not modify
    private javax.swing.JPanel jPanel10;
    private javax.swing.JPanel jPanel12;
    private javax.swing.JPanel jPanel13;
    private javax.swing.JPanel jPanel15;
    private javax.swing.JPanel jPanel16;
    private javax.swing.JPanel jPanel18;
    private javax.swing.JPanel jPanel19;
    private javax.swing.JPanel jPanel2;
    private javax.swing.JPanel jPanel3;
    private javax.swing.JPanel jPanel4;
    private javax.swing.JPanel jPanel6;
    private javax.swing.JPanel jPanel7;
    private javax.swing.JPanel jPanel9;
    private javax.swing.JLabel lblArea;
    private javax.swing.JLabel lblBill;
    private javax.swing.JLabel lblCategory;
    private javax.swing.JLabel lblCountArea;
    private javax.swing.JLabel lblCountBill;
    private javax.swing.JLabel lblCountCategory;
    private javax.swing.JLabel lblCountProduct;
    private javax.swing.JLabel lblCountTable;
    private javax.swing.JLabel lblCountUser;
    private javax.swing.JLabel lblProduct;
    private javax.swing.JLabel lblTable;
    private javax.swing.JLabel lblUser;
    private javax.swing.JPanel pnlArea;
    private javax.swing.JPanel pnlBill;
    private javax.swing.JPanel pnlCategory;
    private javax.swing.JPanel pnlProduct;
    private javax.swing.JPanel pnlTable;
    private javax.swing.JPanel pnlUser;
    // End of variables declaration
}

```

Khai báo các phương thức:

- Constructor PnlHome: phương thức khởi tạo của lớp PnlHome nhận các tham số parent, dbUtil để khởi tạo và đặt vị trí các thành phần giao diện.
- Phương thức loading(): Phương thức có tác dụng cập nhật giao diện với thông tin về số lượng người dùng, danh mục, sản phẩm, khu vực & bàn, hóa đơn từ cơ sở dữ liệu.

c) Package GUI.area

Gồm 1 Jpanel PnlArea và 3 Jdialog JDDeleteArea, JDModifyArea định nghĩa giao diện và xử lý tương tác của người dùng đối với khu vực.

(1) *JDeleteArea*

XOÁ KHU VỰC

Bạn có chắc chắn muốn xoá

Xoá

Hủy

```
public class JDeleteArea extends javax.swing.JDialog {
    private CallbackAreaDelete callback;
    private Area area;
    private AreaDao areaDao;
    private DbUtil dbUtil;

    interface CallbackAreaDelete {...3 lines }

    public JDeleteArea(java.awt.Frame parent, boolean modal, DbUtil dbUtil, CallbackAreaDelete callback, Area area) {...11 lines }

    /** This method is called from within the constructor to initialize the form ...5 lines */
    @SuppressWarnings("unchecked")
    Generated Code

    private void btnDeleteActionPerformed(java.awt.event.ActionEvent evt) {...13 lines }

    private void btnCancelActionPerformed(java.awt.event.ActionEvent evt) {...3 lines }

    // Variables declaration - do not modify
    private javax.swing.JButton btnCancel;
    private javax.swing.JButton btnDelete;
    private javax.swing.JLabel jLabel1;
    private javax.swing.JPanel jPanel1;
    private javax.swing.JLabel lblConfirm;
    // End of variables declaration
}
```

(2) *JModifyArea*

THÊM MỚI KHU VỰC

Tên khu vực

Tên khu vực không được để trống

Thêm mới

```

public final class JDModifyArea extends javax.swing.JDialog {

    private CallbackAreaModify callback;
    private Area area;
    private DbUtil dbUtil;
    private AreaDao areaDao;

    public interface CallbackAreaModify {...3 lines }

    public JDModifyArea(java.awt.Frame parent, boolean modal, DbUtil dbUtil, CallbackAreaModify callback, Area area) {...18 lines }

    public void loadData() {...5 lines }

    /** This method is called from within the constructor to initialize the form ...5 lines */
    @SuppressWarnings("unchecked")
    Generated Code

    private void btnModifyActionPerformed(java.awt.event.ActionEvent evt) {...45 lines }

    private void txtNameKeyPressed(java.awt.event.KeyEvent evt) {...5 lines }

    // Variables declaration - do not modify
    private javax.swing.JButton btnModify;
    private javax.swing.ButtonGroup buttonGroup1;
    private javax.swing.JPanel jPanel1;
    private javax.swing.JLabel lblName;
    private javax.swing.JLabel lblNameError;
    private javax.swing.JLabel lblTitle;
    private javax.swing.JTextField txtName;
    // End of variables declaration
}

```

Các phương thức:

- Phương thức loadData(): Sửa đổi giao diện nếu là hành động “Sửa đổi khu vực” và lấy dữ liệu từ khu vực được người dùng chọn.
- Phương thức btnModifyActionPerformed(): Phương thức giải quyết sự kiện cho nút “Thêm mới” hoặc “Sửa đổi”.
- Phương thức txtNameKeyPressed(): làm phím Enter có chức năng như khi nhấn chuột vào nút “Thêm mới” hoặc “Sửa đổi”.

(3) PnlArea



Các phương thức:

- Phương thức PnlArea(): tạo lựa chọn làm mới, nếu là admin thì sẽ có thêm chức năng thêm mới, sửa đổi và xóa.
- Phương thức loading(): tải dữ liệu khu vực và in trong 1 JTabbedPane

```

public final class PnlArea extends javax.swing.JPanel implements JDMModifyArea.CallbackAreaModify, JDDDeleteArea.CallbackAreaDelete,
    JDMModifyTable.CallbackTableModify, JDDDeleteTable.CallbackTableDelete, JDTable.CallbackTableExit {
    private Frame parent;
    private JPanel self;
    private DbUtil dbUtil;

    private Area area = null;
    private Table table = null;
    private User user = null;
    private Bill bill = null;

    private List<Area> areas = new ArrayList<>();
    private List<Table> tables = new ArrayList<>();

    private AreaDao areaDao;
    private TableDao tableDao;
    private BillDao billDao;

    public PnlArea(Frame parent, DbUtil dbUtil, User user) {...19 lines }

    private void loading() {...23 lines }

    public void addTab(JTabbedPane tabbedPane, String title, Component tab) {...16 lines }

    public JComponent makeTextPanel() {...25 lines }

    public void makeTable(JComponent rootPanel, Table objTable) {...44 lines }

    /** This method is called from within the constructor to initialize the form ...5 lines */
    @SuppressWarnings("unchecked")
    Generated Code

    private void lblAddMouseClicked(java.awt.event.MouseEvent evt) {...4 lines }

    private void lblUpdateMouseClicked(java.awt.event.MouseEvent evt) {...6 lines }

    private void tabbedPaneMouseClicked(java.awt.event.MouseEvent evt) {...6 lines }

    private void lblDeleteMouseClicked(java.awt.event.MouseEvent evt) {...6 lines }

    private void lblRefreshMouseClicked(java.awt.event.MouseEvent evt) {...3 lines }

    private void jMenuItemDeleteActionPerformed(java.awt.event.ActionEvent evt) {...4 lines }

    private void jMenuItemEditActionPerformed(java.awt.event.ActionEvent evt) {...4 lines }

```

- Phương thức addTab(): thêm 1 tab mới (khu vực) vào JTabbedPane (chứa nhiều khu vực)
- Phương thức makeTextPanel(): Tạo và trả về một JPanel khu vực được sử dụng để chứa nội dung của mỗi tab trong JTabbedPane (chứa 1 danh sách khu vực)
- Phương thức makeTable(): Tạo và hiển thị một bàn trong giao diện người dùng, được đặt trong một JPanel, thông tin bàn và hóa đơn sẽ hiển thị khi người dùng click vào giao diện bàn
- Phương thức lblAddMouseClicked(): Thêm sự kiện trong nút “Thêm mới” để thêm khu vực
- Phương thức lblUpdateMouseClicked(): Thêm sự kiện trong nút “Sửa đổi” để sửa đổi khu vực

- Phương thức `tabbedPaneMouseClicked ()`: Thêm sự kiện cho các nút trong thanh `TabbedPane` để thay đổi lựa chọn các khu vực đang có
- Phương thức `lblDeleteMouseClicked()`: Thêm sự kiện trong nút “Xóa” để xóa khu vực
- Phương thức `lblRefreshMouseClicked ()`: Thêm sự kiện trong nút “Làm mới” để hiển thị lại toàn bộ dữ liệu ban đầu
- Phương thức `jMenuItemDeleteActionPerformed ()`: Thêm sự kiện xóa bản trong 1 khu vực
- Phương thức `jMenuItemEditActionPerformed ()`: Thêm sự kiện cập nhật bản trong 1 khu vực

d) Package GUI.bill

(1) *ExportBill*

Xuất file dữ liệu bill dưới dạng file văn bản

```

Bill_2023-11-28_15-40-24.txt - Notepad
File Edit Format View Help
*****Hoá đơn thanh toán*****
Mã hoá đơn: 13
Giờ đặt bàn: 28/11/2023 10:41:00
Nhân viên: Phạm Văn Tiến
*****
    Tên hàng          Đơn giá      Số lượng    Thành tiền
    Bạc xỉu           35.000        52         1.820.000
    Espresso          30.000        41         1.230.000
    Chanh leo          49.000        51         2.499.000
*****
    Tổng thành tiền:                                5.549.000
*****Cảm ơn quý khách*****
  
```

Khai báo các thành phần thuộc tính và phương thức:

```

public final class ExportBill {
    private DbUtil dbUtil;

    private Bill bill = null;

    private List<BillDetail> billDetails = null;

    private BillDao billDao = null;
    private BillDetailDao billDetailDao = null;

    private static int columnWidthName = 25;
    private static int columnWidthNumber = 15;

    public ExportBill(Frame parent, boolean modal, DbUtil dbUtil, Bill bill) throws ParseException {...51 lines }

    private String createFileName() {...5 lines }

    private static String printRow(String[] rowData) {...15 lines }

    private static String padString(String str, int length) {...7 lines }
}
  
```

Khai báo các phương thức:


- Constructor ExportBill: Khởi tạo lớp ExportBill nhận các tham số parent, modal, dbUtil, bill để khởi tạo các thuộc tính và thực hiện các thao tác xuất hoá đơn dựa trên dữ liệu và tùy chọn người dùng
- Phương thức createFileName(): Tạo tên File dựa trên thời gian hiện tại
- Phương thức printRow(String[] rowData): Phương thức trả về một dòng trong bảng hóa đơn để xuất ra file với các cột được căn chỉnh
- Phương thức padString(String str, int length): Phương thức này dùng để thêm ký tự vào chuỗi để đạt được độ dài length

(2) JDBill

Hiển thị thông tin chi tiết hóa đơn

THÔNG TIN CHI TIẾT HOÁ ĐƠN

×



THÔNG TIN CHI TIẾT HOÁ ĐƠN

Mã hoá đơn

1|

Thời gian

2023-10-26 07:37:00.0

Tên bàn

Bàn 2

Tổng tiền

540.000

ID	Tên sản phẩm	Đơn giá	Số lượng	Thành tiền
5	Latte	45.000	12	540.000

Khai báo các thành phần thuộc tính và phương thức:

```
public final class JDBill extends javax.swing.JDialog {
    private DbUtil dbUtil;

    private Bill bill = null;

    private List<BillDetail> billDetails = null;

    private BillDao billDao = null;
    private BillDetailDao billDetailDao = null;

    public JDBill(Frame parent, boolean modal, DbUtil dbUtil, Bill bill) {...29 lines }

    private void loading() {...18 lines }

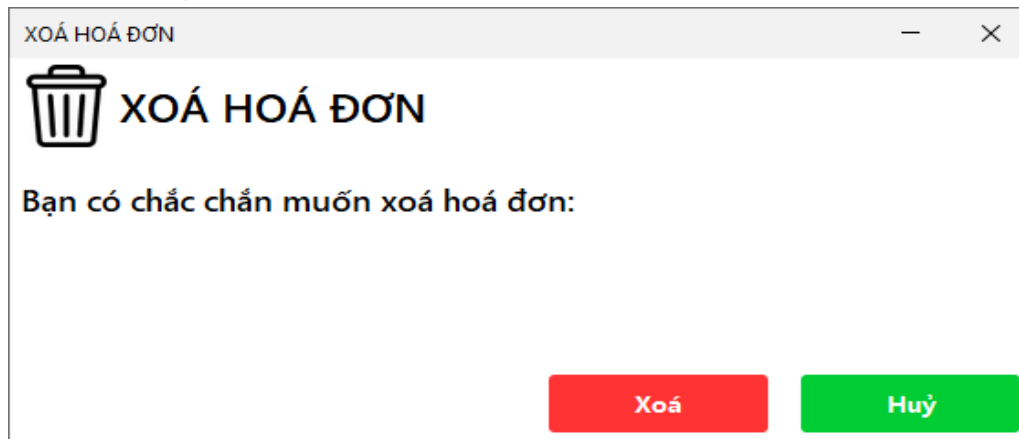
    /** This method is called from within the constructor to initialize the form ...5 lines */
    @SuppressWarnings("unchecked")
    Generated Code

    // Variables declaration - do not modify
    private javax.swing.JPanel jPanel1;
    private javax.swing.JScrollPane jScrollPane1;
    private javax.swing.JLabel lblBillId;
    private javax.swing.JLabel lblBillTime;
    private javax.swing.JLabel lblBillTotalPrice;
    private javax.swing.JLabel lblNameTable;
    private javax.swing.JLabel lblTitle;
    private javax.swing.JTable tblBillDetail;
    private javax.swing.JTextField txtBillId;
    private javax.swing.JTextField txtBillTime;
    private javax.swing.JTextField txtBillTotalPrice;
    private javax.swing.JTextField txtNameTable;
    // End of variables declaration
}
```

- Constructor JDBill: Phương thức khởi tạo của lớp JDBill nhận các tham số parent, modal, dbUtil, callback, bill để khởi tạo và đặt vị trí các thành phần giao diện
- Phương thức loading(): Phương thức này dùng để hiển thị dữ liệu chi tiết của hóa đơn từ cơ sở dữ liệu và hiển thị lên bảng tblBillDetail

(3) *JDeleteBill*

Hiện thị bảng xác nhận xóa hóa đơn



Khai báo các thành phần thuộc tính và phương thức:

```

public class JDeleteBill extends javax.swing.JDialog {

    private CallbackBillDelete callback;
    private DbUtil dbUtil;
    private Bill bill = null;
    private BillDao billDao;

    public interface CallbackBillDelete {...3 lines }

    public JDeleteBill(Frame parent, boolean modal, DbUtil dbUtil, CallbackBillDelete callback, Bill bill) {...11 lines }

    /** This method is called from within the constructor to initialize the form ...5 lines */
    @SuppressWarnings("unchecked")
    Generated Code

    private void btnDeleteActionPerformed(java.awt.event.ActionEvent evt) {...12 lines }

    private void btnCancelActionPerformed(java.awt.event.ActionEvent evt) {...3 lines }

    // Variables declaration - do not modify
    private javax.swing.JButton btnCancel;
    private javax.swing.JButton btnDelete;
    private javax.swing.JLabel jLabel1;
    private javax.swing.JPanel jPanel1;
    private javax.swing.JLabel lblConfirm;
    // End of variables declaration
}

```


(4) JDSearchBill

Hiện thị bảng tìm kiếm hóa đơn

**TÌM KIẾM HOÁ ĐƠN**

Mã nhân viên

Mã nhân viên là một số nguyên dương

Thời gian

đến

Nhập đúng định dạng yyyy-mm-dd

Nhập đúng định dạng yyyy-mm-dd

Tên bàn

Trạng thái

Tìm kiếm

Khai báo các thành phần thuộc tính và phương thức:

```
public final class JDSearchBill extends javax.swing.JDialog {

    private CallbackBillSearch callback;
    private Bill bill;
    private DbUtil dbUtil;

    interface CallbackBillSearch {...3 lines }

    public JDSearchBill(java.awt.Frame parent, boolean modal, DbUtil dbUtil, CallbackBillSearch callback) {...31 lines }

    private static boolean isValidDate (String dateStr) {...11 lines }

    /** This method is called from within the constructor to initialize the form ...5 lines */
    @SuppressWarnings("unchecked")
    Generated Code

    private void btnSearchActionPerformed(java.awt.event.ActionEvent evt) {...69 lines }

    // Variables declaration - do not modify
    private javax.swing.JButton btnSearch;
    private javax.swing.JComboBox<String> cboStatus;
    private javax.swing.JLabel jLabel1;
    private javax.swing.JLabel jLabel3;
    private javax.swing.JLabel jLabel6;
    private javax.swing.JPanel jPanel1;
    private javax.swing.JLabel lblIdError;
    private javax.swing.JLabel lblTableName;
    private javax.swing.JLabel lblTime;
    private javax.swing.JLabel lblTimeError;
    private javax.swing.JLabel lblTimeError1;
    private javax.swing.JLabel lblUserId;
    private javax.swing.JTextField txtTableName;
    private javax.swing.JTextField txtTimeEnd;
    private javax.swing.JTextField txtTimeStart;
    private javax.swing.JTextField txtUserId;
    // End of variables declaration
}
```

- Constructor JDSearchBill: Phương thức khởi tạo của lớp JDSearchBill nhận các tham số parent, modal, dbUtil, callback để khởi tạo và đặt vị trí các thành phần giao diện
- Phương thức btnSearchActionPerformed(): Phương thức dùng để lấy thông tin người dùng nhập vào từ các ô, kiểm tra tính hợp lệ của từng trường nhập vào và hiển thị danh sách các Bill thỏa mãn nếu nhập vào hợp lệ.

(5) PnlBill

Hiển thị danh sách các hóa đơn được tạo và các nút truy cập tiện ích liên quan tới hóa đơn

			
Tim kiếm	Xoá	Làm mới	Xuất hoá đơn

ID	Nhân viên	Bàn	Tổng tiền	Trạng thái	Thời gian
2	Phạm Văn Tiến	Bàn 5	3.700.000	Đã thanh toán	2023-11-27 21:51:00.0
13	Phạm Văn Tiến	Bàn 4	5.549.000	Đã thanh toán	2023-11-28 10:41:00.0
17	Phạm Văn Tiến	Bàn 3	1.980.000	Đã thanh toán	2023-11-28 10:56:00.0
18	Phạm Văn Tiến	Bàn 5	2.790.000	Đã thanh toán	2023-11-28 10:56:00.0
19	Phạm Văn Tiến	Bàn 13	945.000	Đã thanh toán	2023-11-28 10:56:00.0

Tổng tiền thu về	14.964.000
------------------	------------

Khai báo các thành phần thuộc tính và phương thức:

- Constructor PnlBill: Phương thức khởi tạo của lớp PnlBill nhận các tham số parent, dbUtil, user để khởi tạo và đặt vị trí các thành phần giao diện
- Phương thức loading(): Phương thức dùng để biểu diễn danh sách các bill theo tham số được truyền vào và tính toán tổng tiền thu về của các hoá đơn hiển thị trên bảng.
- Phương thức tblBillMouseClicked(): Hiển thị 1 popupMenu sau khi người dùng chuột phải vào 1 hàng trên bảng danh sách hoá đơn, cho phép người dùng chọn “Xem” để xem chi tiết hoá đơn và “Thanh toán” nếu như hoá đơn này chưa được thanh toán.
- Phương thức actionBillDelete(): Dùng để hiển thị lại toàn bộ danh sách các bill bằng việc gọi tới hàm loading() với tham số null sau khi thực hiện xóa 1 hoá đơn.

- Phương thức `actionBillSearch()`: Dùng để hiển thị danh sách các bill thỏa mãn bill được truyền vào và theo khoảng thời gian bằng hàm `loading()`

```

public class PnlBill extends javax.swing.JPanel implements JDSearchBill.CallbackBillSearch, JDDeleteBill.CallbackBillDelete {
    private Frame parent;
    private DbUtil dbUtil;
    private Bill bill;
    private List<Bill> bills = new ArrayList<>();
    private BillDao billDao;
    private List<BillDetail> billDetails = null;
    private BillDetailDao billDetailDao = null;

    public PnlBill(Frame parent, DbUtil dbUtil, User user) {...13 lines }
    private void loading(Bill newBill, String dateStart, String dateEnd) {...28 lines }

    /** This method is called from within the constructor to initialize the form ...5 lines */
    @SuppressWarnings("unchecked")
    Generated Code

    private void lblSearchMouseClicked(java.awt.event.MouseEvent evt) {...4 lines }
    private void lblDeleteMouseClicked(java.awt.event.MouseEvent evt) {...6 lines }
    private void lblRefreshMouseClicked(java.awt.event.MouseEvent evt) {...3 lines }
    private void lblExportMouseClicked(java.awt.event.MouseEvent evt) {...11 lines }
    private void tblBillMouseClicked(java.awt.event.MouseEvent evt) {...7 lines }
    private void jMenuItemCheckoutActionPerformed(java.awt.event.ActionEvent evt) {...17 lines }
    private void jMenuItemShowActionPerformed(java.awt.event.ActionEvent evt) {...6 lines }

    // Variables declaration - do not modify
    private javax.swing.JMenuItem jMenuItemCheckout;
    private javax.swing.JMenuItem jMenuItemShow;
    private javax.swing.JPanel jPanel1;
    private javax.swing.JPanel jPanel2;
    private javax.swing.JPanel jPanel3;
    private javax.swing.JPanel jPanel4;
    private javax.swing.JPopupMenu jPopupMenu1;
    private javax.swing.JScrollPane jScrollPane1;
    private javax.swing.JLabel lblDelete;
    private javax.swing.JLabel lblExport;
    private javax.swing.JLabel lblRefresh;
    private javax.swing.JLabel lblSearch;
    private javax.swing.JLabel lblTotalBill;
    private javax.swing.JTable tblBill;
    private javax.swing.JTextField txtTotalBill;
    // End of variables declaration

    @Override
    public void actionBillDelete() {...3 lines }

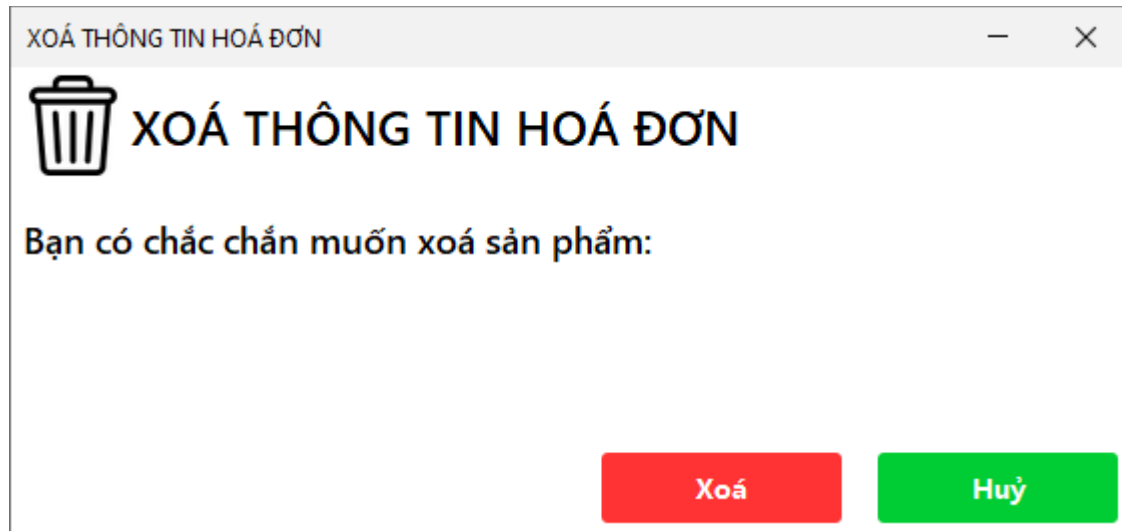
    @Override
    public void actionBillSearch(Bill bill, String dateStart, String dateEnd) {...3 lines }
}

```

e) Package GUI.billDetail

(1) *JDDeleteBillDetail*

Hiển thị bảng xác nhận xóa thông tin chi tiết trong hoá đơn (ở đây là xóa 1 sản phẩm trong hoá đơn đang đặt)



Khai báo các thành phần thuộc tính và phương thức:

```
public class JDDeleteBillDetail extends javax.swing.JDialog {
    private CallbackBillDetailDelete callback;
    private DbUtil dbUtil;
    private BillDetail billDetail = null;
    private BillDetailDao billDetailDao;

    public interface CallbackBillDetailDelete {...3 lines }

    public JDDeleteBillDetail(JDialog parent, boolean modal, DbUtil dbUtil, CallbackBillDetailDelete callback, BillDetail billDetail) {...11 lines }

    /** This method is called from within the constructor to initialize the form ...5 lines */
    @SuppressWarnings("unchecked")
    Generated Code

    private void btnDeleteActionPerformed(java.awt.event.ActionEvent evt) {...12 lines }

    private void btnCancelActionPerformed(java.awt.event.ActionEvent evt) {...3 lines }

    // Variables declaration - do not modify
    private javax.swing.JButton btnCancel;
    private javax.swing.JButton btnDelete;
    private javax.swing.JLabel jLabel1;
    private javax.swing.JPanel jPanel1;
    private javax.swing.JLabel lblConfirm;
    // End of variables declaration
}
```

Khai báo các phương thức:

- Constructor JDDeleteBillDetail: Phương thức khởi tạo của lớp JDDeleteBill nhận các tham số parent, modal, dbUtil, callback, billDetail để khởi tạo và đặt vị trí các thành phần giao diện
- Phương thức btnDeleteActionPerformed(): Phương thức này dùng để xóa một chi tiết hóa đơn khi người dùng ấn vào nút xóa và hiển thị thông báo xóa thành công
- Phương thức btnCancelActionPerformed(): Phương thức dùng hàm dispose() để đóng cửa sổ khi người dùng nhấn nút hủy

(2) JDModifyBillDetail

Hiện thị bảng xác nhận cập nhật chi tiết thông tin trong hoá đơn (ở đây là sửa lại số lượng của 1 sản phẩm trong hoá đơn đang đặt)

The screenshot shows a Java Swing window titled "CẬP NHẬT THÔNG TIN HOÁ ĐƠN". Inside the window, there is a logo of a folder with a document icon and the text "SỬA THÔNG TIN HOÁ ĐƠN". Below this, there are three input fields: "Tên sản phẩm", "Đơn giá", and "Số lượng". The "Số lượng" field has a red error message "Không được để trống" below it. A green button labeled "Cập nhật" is located at the bottom right of the window.

Khai báo các thành phần thuộc tính và phương thức:

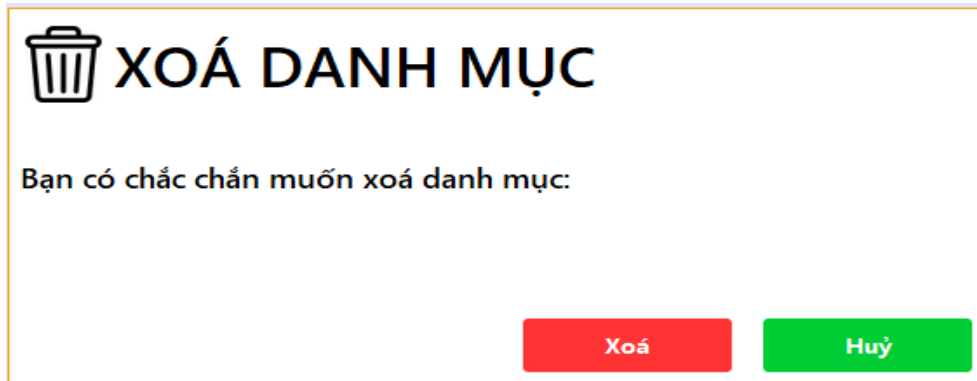
- Constructor JDModifyBillDetail: Phương thức khởi tạo của lớp JDDeleteBill nhận các tham số parent, modal, dbUtil, callback, billDetail để khởi tạo và đặt vị trí các thành phần giao diện
- Phương thức btnModifyActionPerformed(): Phương thức này dùng để thay đổi một chi tiết hóa đơn khi người dùng ấn vào nút xóa và hiển thị thông báo xóa thành công

f) Package GUI.category

Gồm 1 Jpanel PnlCategory và 3 Jdialog JDDeleteCategory, JDModifyCategory và JDSearchCategory định nghĩa giao diện và xử lý tương tác của người dùng đối với danh mục.

(1) JDDeleteCategory

Định nghĩa một cửa sổ dialog cho việc xác nhận và xoá một danh mục



```
public class JDDeleteUser extends javax.swing.JDialog {

    private CallbackUserDelete callback;
    private User user;
    private UserDao userDao;

    interface CallbackUserDelete {
        public void actionUserDelete();
    }

    public JDDeleteUser(java.awt.Frame parent, boolean modal, DbUtil dbUtil) {
        super(parent, modal, true);

        /** This method is called from within the constructor to initialize the content of the dialog
         * @SuppressWarnings("unchecked")
         */
        Generated Code

        private void btnDeleteActionPerformed(java.awt.event.ActionEvent evt) {
            // TODO add your handling code here:
        }

        private void btnCancelActionPerformed(java.awt.event.ActionEvent evt) {
            // TODO add your handling code here:
        }

        // Variables declaration - do not modify
        private javax.swing.JButton btnCancel;
        private javax.swing.JButton btnDelete;
        private javax.swing.JLabel jLabel1;
        private javax.swing.JPanel jPanel1;
        private javax.swing.JLabel lblConfirm;
        // End of variables declaration
    }
}
```

(2) JDModifyCategory

Định nghĩa một cửa sổ dialog cho phép thêm mới hoặc sửa đổi thông tin danh mục

```

public final class JDMModifyCategory extends javax.swing.JDialog {

    private CallbackModify callback;
    private Category category;
    private DbUtil dbUtil;
    private CategoryDao categoryDao;

    interface CallbackModify {...3 lines }

    public JDMModifyCategory(java.awt.Frame parent, boolean modal, DbUtil dbUtil) {
        super(parent, modal, true);

        loadData();
    }

    /** This method is called from within the constructor to initialize the content of the dialog
    @SuppressWarnings("unchecked")
    Generated Code

    private void btnModifyActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
    }

    private void txtNameKeyPressed(java.awt.event.KeyEvent evt) {...5 lines }

    // Variables declaration - do not modify
    private javax.swing.JButton btnModify;
    private javax.swing.ButtonGroup buttonGroup1;
    private javax.swing.JPanel jPanel1;
    private javax.swing.JLabel lblName;
    private javax.swing.JLabel lblNameError;
    private javax.swing.JLabel lblStatus;
    private javax.swing.JLabel lblTitle;
    private javax.swing.JRadioButton rdoActive;

```

- Constructor JDMModifyCategory: Thiết lập giao diện và điều chỉnh hiển thị dựa trên việc có đối tượng category hay không, nếu có thì tức đây là sửa đổi thông tin danh mục, nếu là null sẽ là thêm mới danh mục
- loadData: Nạp dữ liệu danh mục vào trường dữ liệu và sửa lại giao diện JDialog nếu đây là sửa đổi thông tin.
- btnModifyActionPerformed: Xử lý sự kiện khi người dùng nhấn “Thêm mới” hoặc “Sửa đổi”. Kiểm tra các dữ liệu đầu vào (tên) và báo lỗi nếu có. Nếu dữ liệu hợp lệ, tiến hành tạo 1 đối tượng danh mục mới và thực hiện create() hoặc update() thông qua CategoryDao. Cuối cùng là hiển thị thông báo kết quả và gọi callback để cập nhật lại.

THÊM MỚI DANH MỤC

Tên danh mục

Tên danh mục không được để trống

Trạng thái

☒ Hoạt động ☐ Không hoạt động

Thêm mới

CẬP NHẬT DANH MỤC

SỬA ĐỔI DANH MỤC

Tên danh mục

Trạng thái

☒ Hoạt động ☐ Không hoạt động

Sửa đổi

(3) *JDSearchCategory*

Định nghĩa một cửa sổ dialog cho tìm kiếm thông tin danh mục dựa trên tên danh mục, trạng thái

```
public final class JDSearchCategory extends javax.swing.JDialog {

    private CallbackSearch callback;
    private Category category;
    private DbUtil dbUtil;

    interface CallbackSearch {...3 lines }

    public JDSearchCategory(java.awt.Frame parent, boolean modal, DbUtil dbUtil, CallbackSearch callback) {...19 lines }

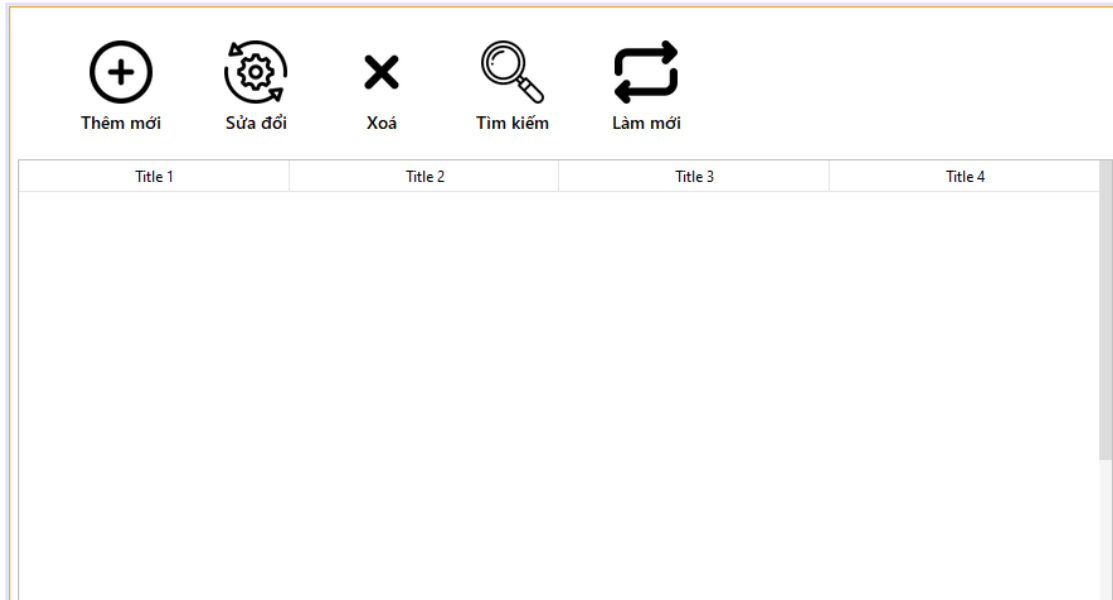
    /** This method is called from within the constructor to initialize the form ...5 lines */
    @SuppressWarnings("unchecked")
    Generated Code

    private void btnSearchActionPerformed(java.awt.event.ActionEvent evt) {...12 lines }

    // Variables declaration - do not modify
    private javax.swing.JButton btnSearch;
    private javax.swing.JComboBox<String> cboStatus;
    private javax.swing.JLabel jLabel1;
    private javax.swing.JLabel jLabel2;
    private javax.swing.JLabel jLabel3;
    private javax.swing.JPanel jPanel1;
    private javax.swing.JTextField txtName;
    // End of variables declaration
}
```

(4) *PnlCategory*

Jpanel hiển thị và quản lý thông tin category



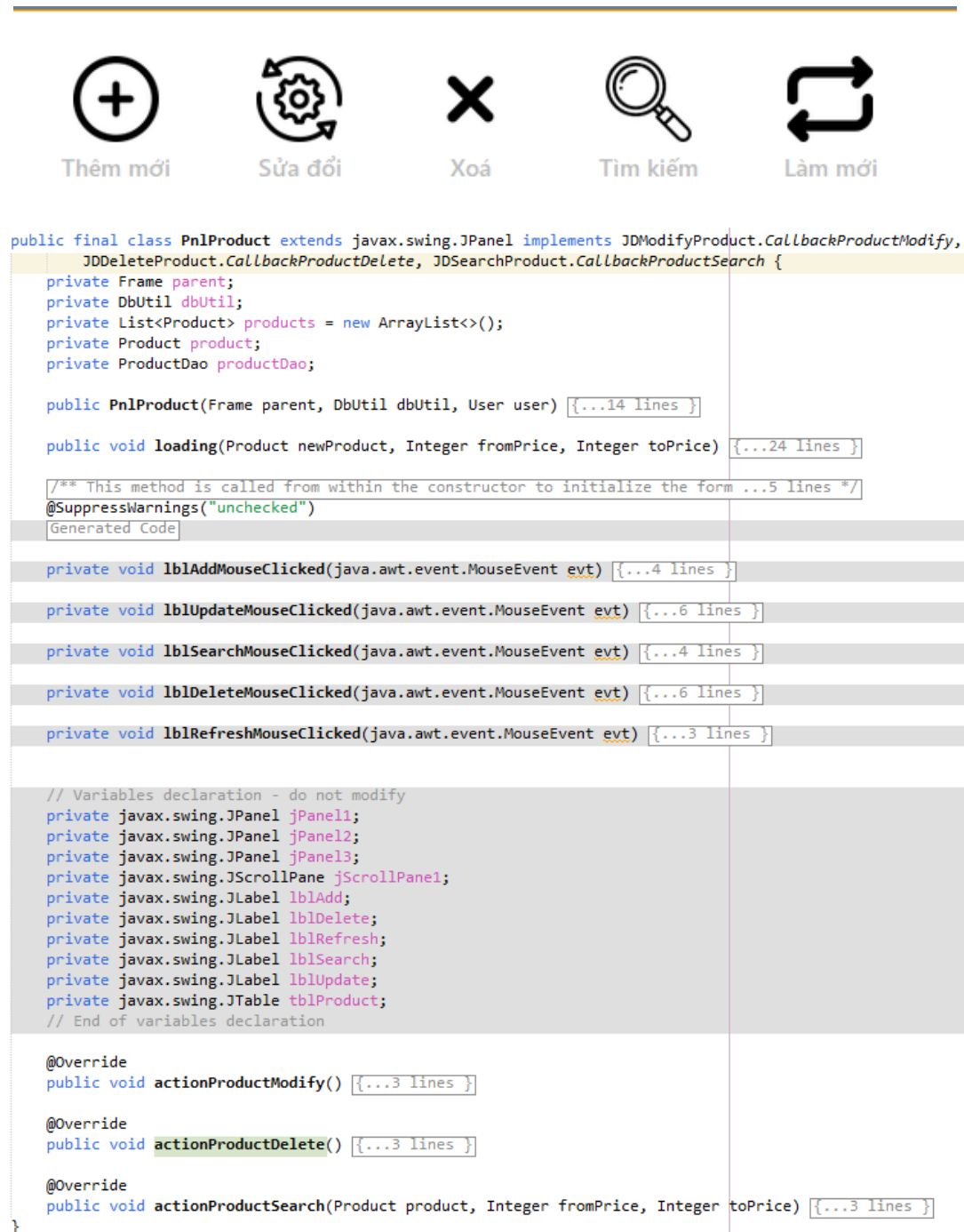
Các phương thức:

- Constructor: Khởi tạo một đối tượng PnlCategory làm giao diện hiển thị danh sách các category. Gọi hàm loading để hiển thị danh sách category khi panel được khởi tạo.
- loading: Hiển thị danh sách category sử dụng getAll() trong CategoryDao. Tham số truyền vào newCategory nếu khác null sẽ hiển thị kết quả tìm kiếm.
- Các phương thức lblAddMouseClicked, lblUpdateMouseClicked, lblSearchMouseClicked, lblDeleteMouseClicked và lblRefreshMouseClicked : Xử lý sự kiện khi người dùng nhấn thêm, cập nhật, tìm kiếm, xoá và làm mới. Gọi đến các Jdialog tương ứng để thực hiện.
- Các phương thức actionCategoryModify(), actionCategoryDelete(), actionCategorySearch(): Ghi đè các phương thức callback từ các dialog (JdModifyCategory, JdDeleteCategory, JdSearchCategory) để cập nhật lại danh sách category sau khi thực hiện thêm mới, sửa đổi, xoá, hoặc tìm kiếm.

g) Package GUI.product

Gồm 1 Jpanel PnlProduct và 3 Jdialog JDDeleteProduct, JDModifyProduct, JDSearchProduct định nghĩa giao diện và xử lý tương tác của người dùng.

(1) PnlProduct



(2) *JDeleteProduct*

XOÁ SẢN PHẨM

Bạn có chắc chắn muốn xóa

Xoá

Hủy

```
public class JDeleteProduct extends javax.swing.JDialog {
    private CallbackProductDelete callback;
    private Product product;
    private ProductDao productDao;
    private DbUtil dbUtil;

    interface CallbackProductDelete {...3 lines }

    public JDeleteProduct(java.awt.Frame parent, boolean modal, DbUtil dbUtil, CallbackProductDelete callback, Product product) {...10 lines }

    /** This method is called from within the constructor to initialize the form ...5 lines */
    @SuppressWarnings("unchecked")
    Generated Code

    private void btnDeleteActionPerformed(java.awt.event.ActionEvent evt) {...10 lines }

    private void btnCancelActionPerformed(java.awt.event.ActionEvent evt) {...3 lines }

    // Variables declaration - do not modify
    private javax.swing.JButton btnCancel;
    private javax.swing.JButton btnDelete;
    private javax.swing.JLabel jLabel1;
    private javax.swing.JPanel jPanel1;
    private javax.swing.JLabel lblConfirm;
    // End of variables declaration
}
```

Các phương thức:

- Phương thức `btnDeleteActionPerformed()`: Phương thức dùng để xóa một Product khỏi cơ sở dữ liệu khi người dùng ấn vào nút xóa trên cửa sổ `JDeleteProduct`, hiển thị thông báo xóa thành công hay không thành công sau khi hoàn thành.
- Phương thức `btnCancelActionPerformed()`: Phương thức đóng cửa sổ khi người dùng tương tác với nút “Hủy” bằng hàm `dispose()`

(3) *JDModifyProduct*

THÊM MỚI SẢN PHẨM

Tên sản phẩm

Không được để trống

Giá

Không được để trống

Danh mục

Trạng thái

☒ Hoạt động

☐ Không hoạt động

```

public final class JDModifyProduct extends javax.swing.JDialog {
    private Product product;
    private DbUtil dbUtil;
    private CallbackProductModify callback;
    private List<Category> categories = new ArrayList<>();
    private CategoryDao categoryDao;
    private ProductDao productDao;

    interface CallbackProductModify {...3 lines }

    public JDModifyProduct(java.awt.Frame parent, boolean modal, DbUtil dbUtil, CallbackProductModify callback, Product product) {...27 lines }

    public void loadCategory() {...11 lines }

    public void loadingData() {...9 lines }

    /** This method is called from within the constructor to initialize the form ...5 lines */
    @SuppressWarnings("unchecked")
    Generated Code

    private void btnModifyActionPerformed(java.awt.event.ActionEvent evt) {...94 lines }

    // Variables declaration - do not modify
    private javax.swing.JButton btnModify;
    private javax.swing.ButtonGroup buttonGroup1;
    private javax.swing.JComboBox<Category> cboCategory;
    private javax.swing.JPanel jPanel1;
    private javax.swing.JLabel lblCategory;
    private javax.swing.JLabel lblName;
    private javax.swing.JLabel lblNameError;
    private javax.swing.JLabel lblPrice;
    private javax.swing.JLabel lblPriceError;
    private javax.swing.JLabel lblStatus;
    private javax.swing.JLabel lblTitle;
    private javax.swing.JRadioButton rdoActive;
    private javax.swing.JRadioButton rdoNonActive;
    private javax.swing.JTextField txtName;
    private javax.swing.JTextField txtPrice;
    // End of variables declaration
}

```

- Phương thức `loadCategory()`: nạp dữ liệu các danh mục đang hoạt động vào 1 `DefaultComboBoxModel` cho người dùng lựa chọn.
- Phương thức `loadData()`: nạp dữ liệu `Product` đang được chọn vào các trường nhập nếu thực hiện sửa đổi sản phẩm.
- Phương thức `btnModifyActionPerformed()`: Phương thức dùng để tạo sự kiện cho nút “Thêm mới” hoặc “Sửa đổi”.

(4) *JDSearchProduct*



TÌM KIẾM SẢN PHẨM

Tên sản phẩm

Khoảng giá

đến

Danh mục

Trạng thái

Tim kiếm

```

public final class JDSearchProduct extends javax.swing.JDialog {

    private DbUtil dbUtil;
    private CallbackProductSearch callback;
    private Product product;
    private List<Category> categories = new ArrayList<>();
    private CategoryDao categoryDao;

    interface CallbackProductSearch {...3 lines }

    public JDSearchProduct(java.awt.Frame parent, boolean modal, DbUtil dbUtil, CallbackProductSearch callback) {...21 lines }

    public void loadCategory() {...10 lines }

    public void loadStatus() {...8 lines }

    /** This method is called from within the constructor to initialize the form ...5 lines */
    @SuppressWarnings("unchecked")
    Generated Code

    private void btnModifyActionPerformed(java.awt.event.ActionEvent evt) {...17 lines }

    // Variables declaration - do not modify
    private javax.swing.JButton btnModify;
    private javax.swing.JComboBox<Category> cboCategory;
    private javax.swing.JComboBox<String> cboStatus;
    private javax.swing.JLabel jLabel1;
    private javax.swing.JLabel jLabel2;
    private javax.swing.JLabel jLabel3;
    private javax.swing.JLabel jLabel4;
    private javax.swing.JLabel jLabel5;
    private javax.swing.JLabel jLabel6;
    private javax.swing.JPanel jPanel1;
    private javax.swing.JTextField txtFromPrice;
    private javax.swing.JTextField txtName;
    private javax.swing.JTextField txtToPrice;
    // End of variables declaration
}

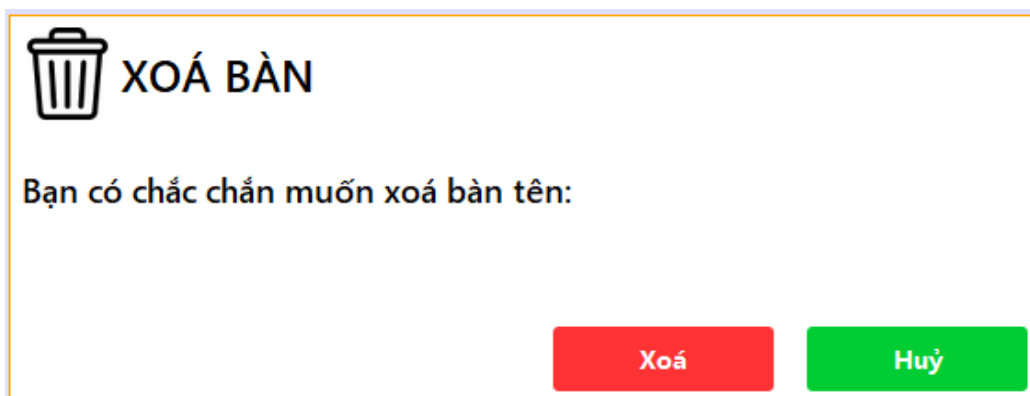
```

Các phương thức:

- Phương thức loadCategory(): tải danh sách Categories trong cơ sở dữ liệu, và hiển thị trong JComboBox, mặc định là “Không chọn”.
- Phương thức loadStatus(): thêm 3 lựa chọn "Không chọn", "Hoạt động" và "Không hoạt động" vào JComboBox
- Phương thức btnModifyActionPerformed(): tạo sự kiện click chuột tìm sản phẩm có trong cơ sở dữ liệu, tham số gồm tên, khoảng giá, tên danh mục và trạng thái hoạt động

h) Package GUI.table

(1) JDDeleteTable




```

public class JDDeleteTable extends javax.swing.JDialog {

    private CallbackTableDelete callback;
    private DbUtil dbUtil;
    private Table table = null;
    private TableDao tableDao;

    public interface CallbackTableDelete {...3 lines }

    public JDDeleteTable(java.awt.Frame parent, boolean modal, DbUtil db

    /**
     * This method is called from within the constructor to initialize t
     * WARNING: Do NOT modify this code. The content of this method is a
     * regenerated by the Form Editor.
     */
    @SuppressWarnings("unchecked")
    Generated Code


    private void btnDeleteActionPerformed(java.awt.event.ActionEvent evt

    private void btnCancelActionPerformed(java.awt.event.ActionEvent evt
        dispose();
    }

    // Variables declaration - do not modify
    private javax.swing.JButton btnCancel;
    private javax.swing.JButton btnDelete;
    private javax.swing.JLabel jLabel1;
    private javax.swing.JPanel jPanel1;
    private javax.swing.JLabel lblConfirm;

```

(2) JDModifyTable




THÊM MỚI BÀN

Tên bàn

Không được để trống

Khu vực

CẬP NHẬT BÀN



SỬA ĐỔI BÀN

Tên bàn

Khu vực

```
private CallbackTableModify callback;
private DbUtil dbUtil;

private Table table;
private Area area;
private List<Area> areas = new ArrayList<>();

private AreaDao areaDao;
private TableDao tableDao;

public interface CallbackTableModify {...3 lines }

public JDModifyTable(java.awt.Frame parent, boolean modal, DbUtil dbUtil, CallbackTableModify callback) {
    // ...
}

public void loadArea() {...8 lines }

/**
 * 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")
Generated Code

private void btnModifyActionPerformed(java.awt.event.ActionEvent evt) {...50 lines }

// Variables declaration - do not modify
private javax.swing.JButton btnModify;
private javax.swing.ButtonGroup buttonGroup1;
private javax.swing.JComboBox<Area> cboArea;
```

Các phương thức:

- loadArea(): Đưa dữ liệu khu vực vào JComboBox cho người dùng lựa chọn.
- btnModifyActionPerformed: Xử lý sự kiện khi nút "Sửa đổi" được nhấn. Hiện thị thông báo và gọi callback nếu sửa thành công.

(3) JDTable

THÔNG TIN CHI TIẾT BÀN

Tên sản phẩm: Bạc xỉu

Đơn giá: 35,000

Số lượng:

Thêm sản phẩm

Mã hoá đơn: 1

Tên bàn: Bàn 12

Thời gian: 2023-11-20 16:20:00.0

Tên nhân viên: Lê Văn

Hủy đơn

Tổng tiền: VND

ID	Tên sản phẩm	Giá	Tên danh mục
1	Bạc xỉu	35,000	Cà Phê Việt
2	Cà phê sữa tươi	35,000	Cà Phê Việt
3	Cà phê đen	29,000	Cà Phê Việt
4	Cà phê nâu	35,000	Cà Phê Việt
5	Latte	45,000	Cà Phê Tây
6	Cappuchino	45,000	Cà Phê Tây
7	Espresso	30,000	Cà Phê Tây
8	Sinh tố bơ	59,000	Sinh Tố
9	Sinh tố xoài	5,000	Sinh Tố
10	Trà cam quế	45,000	Trà
11	Trà đào chanh leo	45,000	Trà
12	Trà quất mật ong	45,000	Trà
13	Trà lip ton	25,000	Trà
14	Trà mạn	35,000	Trà
15	Cóc xanh (theo mùa)	55,000	Trái Cây
16	Chanh tươi	39,000	Trái Cây
17	Dưa hấu	49,000	Trái Cây
18	Chanh leo	49,000	Trái Cây
19	Cam tươi	65,000	Trái Cây
20	Ổi	45,000	Trái Cây
21	Xoài xanh	45,000	Trái Cây
22	Sữa chua dầm đá	35,000	Sữa Chua
23	Sữa chua ca cao	40,000	Sữa Chua
24	Sữa chua cà phê	40,000	Sữa Chua
25	Sữa chua trái cây	55,000	Sữa Chua
26	Trà chanh dây	25,000	Trà

ID	Tên sản phẩm	Đơn giá	Số lượng	Thành tiền
----	--------------	---------	----------	------------

```

public void loadingProduct() {...36 lines }

public void loadingBill() {...39 lines }

public void loadingBillDetail() {...41 lines }

/** This method is called from within the constructor to initialize the form ...5 lines */
@SuppressWarnings("unchecked")
Generated Code

private void btnBookActionPerformed(java.awt.event.ActionEvent evt) {...18 lines }

private void btnAddProductActionPerformed(java.awt.event.ActionEvent evt) {...66 lines }

private void btnCheckoutActionPerformed(java.awt.event.ActionEvent evt) {...18 lines }

private void menuItemEditActionPerformed(java.awt.event.ActionEvent evt) {...6 lines }

private void tblBillDetailMouseClicked(java.awt.event.MouseEvent evt) {...5 lines }

private void menuItemDeleteActionPerformed(java.awt.event.ActionEvent evt) {...6 lines }

private void formWindowClosing(java.awt.event.WindowEvent evt) {...4 lines }

private void btnDeleteBillActionPerformed(java.awt.event.ActionEvent evt) {...6 lines }

```

Các phương thức:

- Constructor JDTable: Nhận một Frame cha, giá trị boolean cho biết cửa sổ có là modal hay không, một đối tượng DbUtil để tạo kết nối đến cơ sở dữ liệu, một callback để xử lý sự kiện sau khi xóa người dùng, thông tin về tài khoản user đang sử dụng và tên bàn đang đặt.
- loadingProduct(): Thêm danh sách sản phẩm đang hoạt động cho người dùng lựa chọn vào Jtable.
- loadingBill(): Tạo thông tin hoá đơn khi ấn nút “Đặt bàn”.
- loadingBillDetail(): Hiển thị thông tin chi tiết sản phẩm trong hoá đơn người dùng đang đặt và cập nhật tổng giá trị hoá đơn.
- btnBookActionPerformed: Xử lý sự kiện khi nút "Đặt bàn" được nhấn.
- btnAddProductActionPerformed: Xử lý sự kiện khi nút "Thêm sản phẩm" được nhấn.
- btnCheckoutActionPerformed: Xử lý sự kiện khi nút "Thanh toán" được nhấn.
- menuItemEditActionPerformed và menuItemDeleteActionPerformed: Xử lý sự kiện khi người dùng chọn sản phẩm đã được đặt và tiến hành sửa đổi hoặc xóa nó khỏi hoá đơn.
- btnDeleteBillActionPerformed: Xử lý sự kiện khi nút "Hủy đơn" được nhấn.

5. Package team13.quanlyquancaphe

Chỉ có class Main để khởi chạy chương trình, tiến hành kết nối thử đến cơ sở dữ liệu, nếu thành công sẽ tạo và hiển thị giao diện chính (Dashboard) của ứng dụng.

```
public class QuanLyQuanCaPhe {

    public static void main(String[] args) {
        try {
            DbUtil dbUtil = new DbUtil();
            Connection conn = dbUtil.getConnection();

            if (Common.isNullOrEmpty(obj: conn)) {
                JOptionPane.showMessageDialog(parentComponent: null,
                    message: "Kết nối CSDL không thành công.", title: "Có lỗi xảy ra",
                    messageType: JOptionPane.ERROR_MESSAGE);
                System.exit(status: 0);
            } else {
                Dashboard dashboard = new Dashboard(dbUtil);
                dashboard.setVisible(b: true);
            }
        } catch (HeadlessException e) {
            e.printStackTrace();
            System.exit(status: 0);
        }
    }
}
```

III. Tài liệu tham khảo

Java Wrap Layout: <https://tips4java.wordpress.com/2008/11/06/wrap-layout/>

Mã nguồn tham khảo từ: <https://github.com/anhnmmt/java-coffee-shop-management/>