|  |
| --- |
| FPT POLYTECHNIC |
| **DỰ ÁN 1** |
| NGÀNH CÔNG NGHỆ THÔNG TIN (ỨNG DỤNG PHẦN MỀM) |



|  |
| --- |
| HÀ NỘI 2018 |

|  |
| --- |
| GIẢNG VIÊN HƯỚNG DẪN: NGUYỄN TRUNG KIÊN |
| SINH VIÊN THỰC HIỆN |
| 1. Phạm Huỳnh Quang Minh (PC04603), Trưởng nhóm 2. Lê Sỹ Sơn (PC04620) 3. Võ Long Hải (PC04485) 4. Tô Vũ Linh (PC04612) 5. Nguyễn Minh Tiến (PC04492) |

MỤC LỤC

[1 Phân tích 2](#_Toc520132211)

[1.1 Hiện trạng 2](#_Toc520132212)

[1.2 Yêu cầu hệ thống 2](#_Toc520132213)

[1.3 Use case 2](#_Toc520132214)

[2 Thiết kế 3](#_Toc520132215)

[2.1 Mô hình triển khai 12](#_Toc520132216)

[2.2 Thiết kế CSDL 13](#_Toc520132217)

[2.2.1 Sơ đồ quan hệ thực thể 13](#_Toc520132218)

[2.2.2 Thiết kế chi tiết các thực thể 13](#_Toc520132219)

[2.3 Thiết kế giao diện 13](#_Toc520132220)

[2.3.1 Sơ đồ tổ chức giao diện 13](#_Toc520132221)

[2.3.2 Thiết kế giao diện cho các chức năng nghiệp vụ 13](#_Toc520132222)

[3 Thực hiện viết mã 13](#_Toc520132223)

[3.1 Viết mã tạo CSDL 13](#_Toc520132224)

[3.1.1 Tạo CSDL 13](#_Toc520132225)

[3.1.2 SQL truy vấn và thao tác 13](#_Toc520132226)

[3.1.3 Các thủ tục lưu tổng hợp thống kê 13](#_Toc520132227)

[3.2 Lập trình JDBC 13](#_Toc520132228)

[3.2.1 Lớp hỗ trợ 13](#_Toc520132229)

[3.2.2 Model class - Các lớp mô tả dữ liệu 13](#_Toc520132230)

[3.2.3 DAO Class - Các lớp truy xuất dữ liệu 14](#_Toc520132231)

[3.3 Viết mã cho ứng dụng 14](#_Toc520132232)

[3.3.1 Xử lý Form X 14](#_Toc520132233)

[3.3.2 Xử lý Form Y 14](#_Toc520132234)

[4 Kiểm thử 14](#_Toc520132235)

[4.1 Kiểm thử form X 14](#_Toc520132236)

[4.2 Kiểm thử form Y 14](#_Toc520132237)

[5 Đóng gói và triển khai 14](#_Toc520132238)

[5.1 Hướng dẫn chuyển đổi jar thành exe 14](#_Toc520132239)

[5.2 Hướng dẫn cài đặt triển khai 14](#_Toc520132240)

[5.3 Hướng dẫn sử dụng phần mềm 14](#_Toc520132241)

# Phân tích

## Hiện trạng

Giày là mặt hàng thuộc lĩnh vực thời trang và được sử dụng thường xuyên trong cuộc sống hàng ngày. Giày có nhiều chủng loại, nhiều chất liệu, nhiều xuất xứ khác nhau… tạo nên một thị trường giày thời trang đa dạng và đầy màu sắc. Đó cũng là lí do khiến công việc quản lý những cửa hàng giày trở nên khó khăn. Có thể kể đến như:

  + Mất kiểm soát về hàng hóa thực tế trong kho, luôn trong trạng thái mông lung, không rõ ràng về số lượng hàng hóa tồn thực tế.

  + Lượng công việc quá lớn khiến nhà quản lý luôn cảm thấy thiếu một bộ phận nhân sự làm việc có hiệu quả, từ đó dẫn đến tâm lý lo lắng về đội ngũ nhân sự, bán hàng của mình.

  + Vì lo sợ không quản lý được nên kế hoạch mở rộng sản xuất, kinh doanh khó lòng thực hiện được.

  + Chính vì lượng công việc quá nhiều nên thời gian thư giãn, dành cho cho gia đình, cho bản thân không có, làm ảnh hưởng đến sức khỏe thậm chí là khủng hoảng về tinh thần.

 + Một số những nhà quản lý hầu như đều không có nghiệp vụ về kế toán nên đôi khi các bản báo cáo được gửi lên đều gây ra cảm giác khó hiểu, khó kiểm tra sai sót. Thông thường, việc kiểm tra báo cáo kinh doanh đột xuất, không theo kỳ thường làm các doanh nghiệp cảm thấy khó khăn.

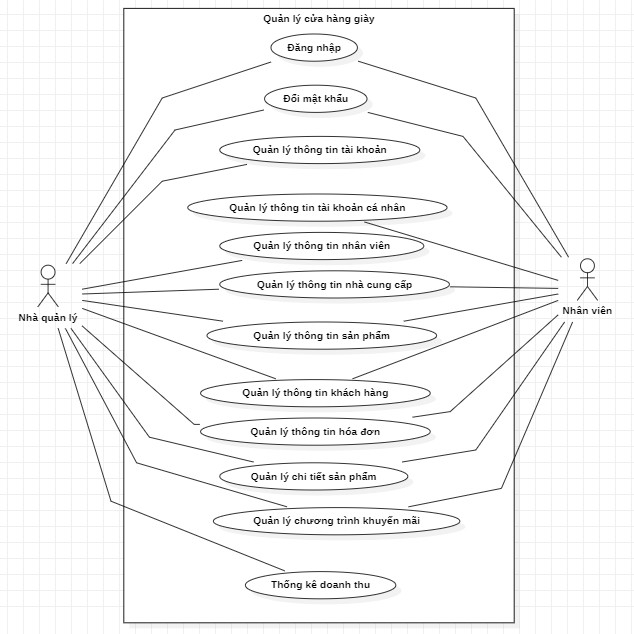
      + Phần mềm của nhóm 4 ra đời nhằm giải quyết được những khó khăn mà các nhà quản lý cửa hàng giày gặp phải đồng thơi giúp nâng cao chất lượng và hiệu quả quản lý.

Yêu cầu hệ thống

* Yêu cầu chức năng nghiệp vụ
* Quản lý thông tin nhân viên.
* Quản lý tài khoản
* Quản lý tài khoản cá nhân
* Quản lý thông tin nhà cung cấp.
* Quản lý thông tin sản phẩm.
* Quản lý chi tiết sản phẩm
* Quản lý thông tin khách hàng.
* Quản lý Hóa đơn.
* Quản lý chi tiết hóa đơn
* Quản lý chương trình khuyến mãi
* Thống kê doanh thu theo tháng, năm.
* Yêu cầu về bảo mật
* Tất cả mọi thành viên phải đăng nhập mới sử dụng được phần mềm.
* Quản lý được phép thực hiện tất cả.
* Nhân viên không được phép xóa và không được xem thông tin về doanh thu.
* Yêu cầu về công nghệ
* Ứng dụng phải được thực với công nghệ Swing và JDBC.
* JDK tối thiểu 1.8.
* Hệ quản trị CSDL SQL Server 2008 trở lên.

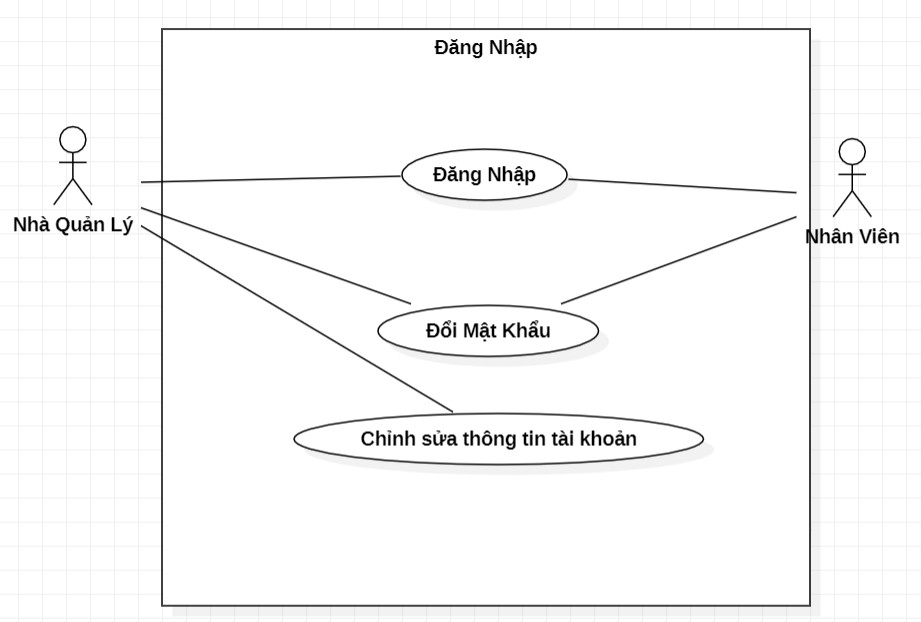
## Use case

### . Sơ đồ tổng quát



### Sơ đồ chi tiết

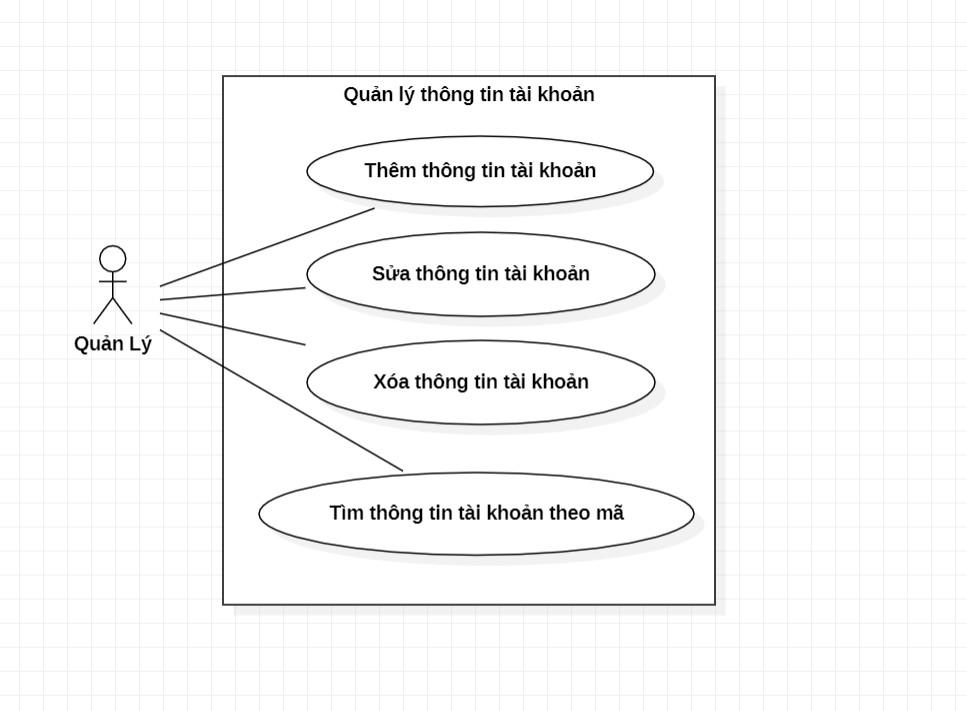
* Đăng nhập



* Quản lý nhân viên



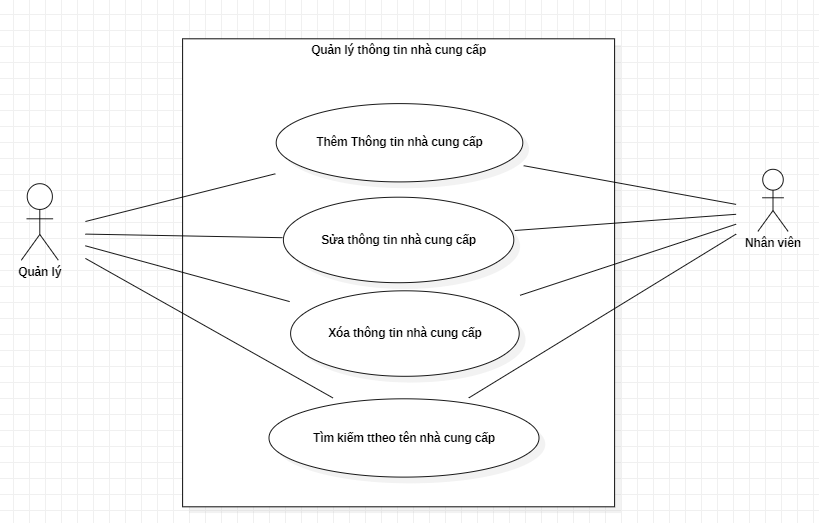
* Quản lý tài khoản



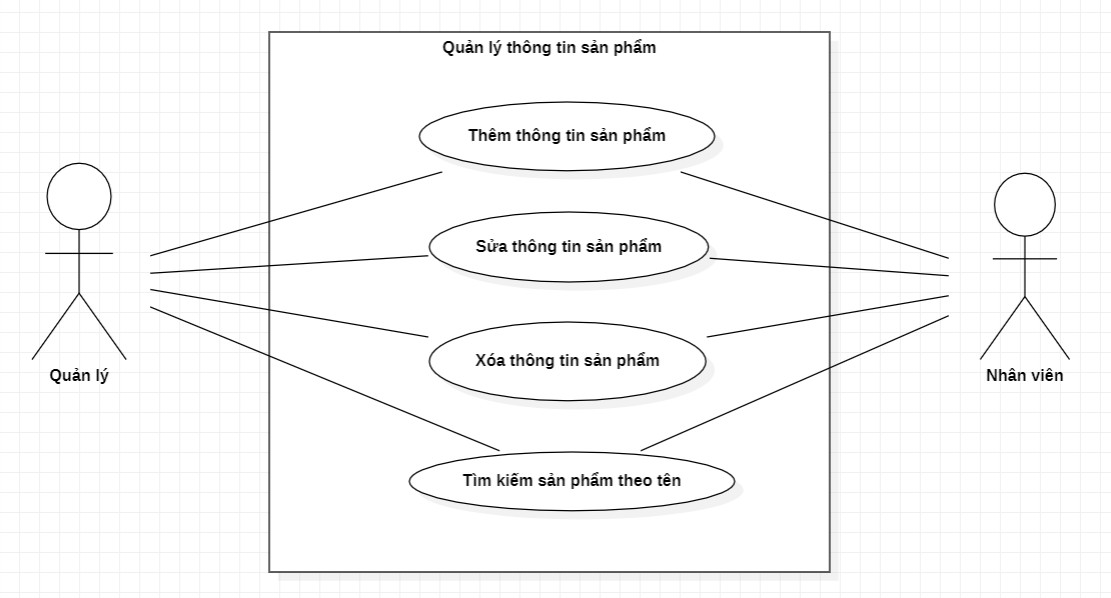
* Quản lý tài khoản cá nhân



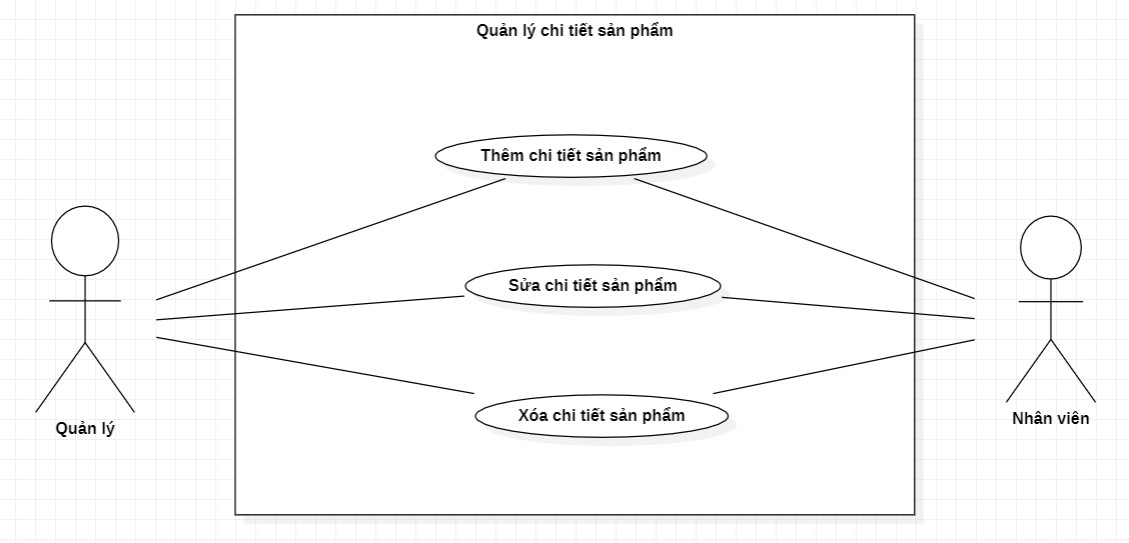
* Quản lý nhà cung cấp



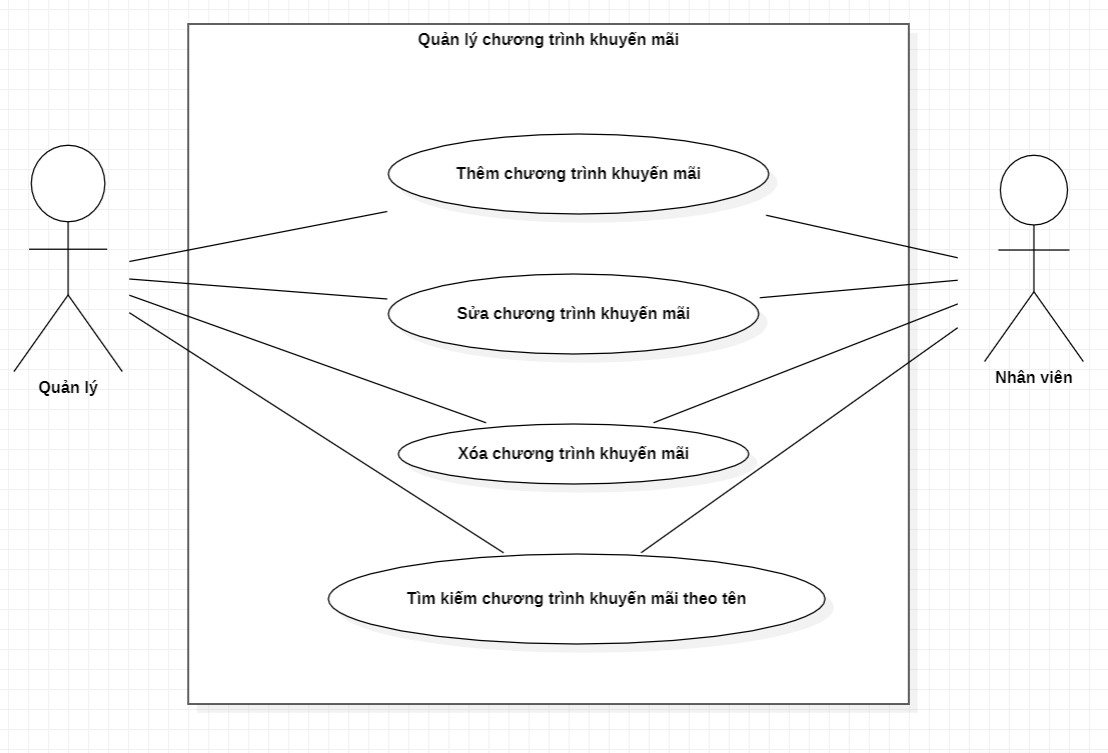
* Quản lý sản phẩm



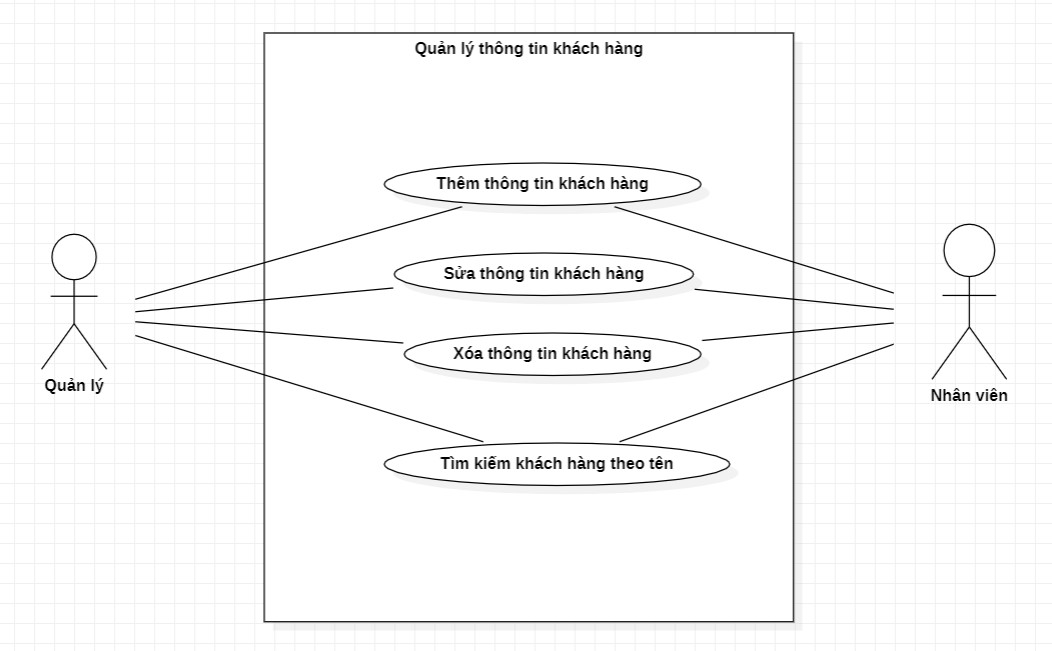
* Quản lý chi tiết sản phẩm



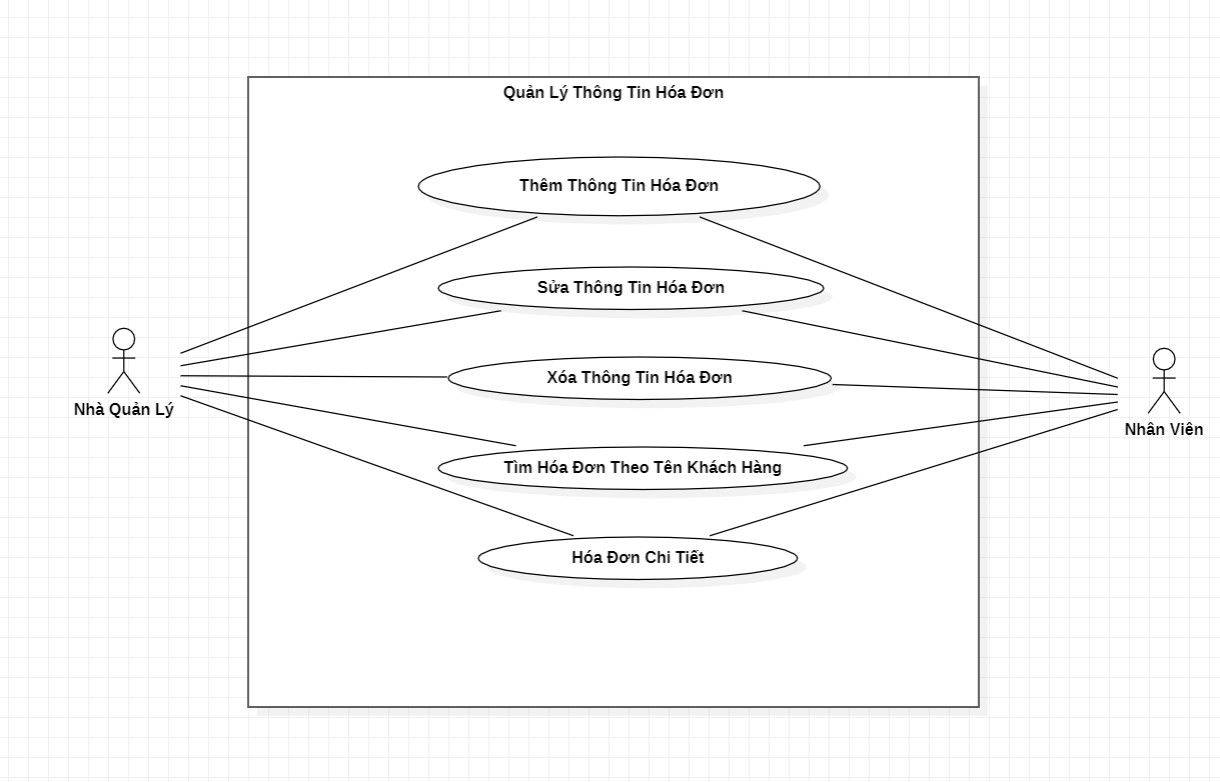
* Quản lý khuyến mãi



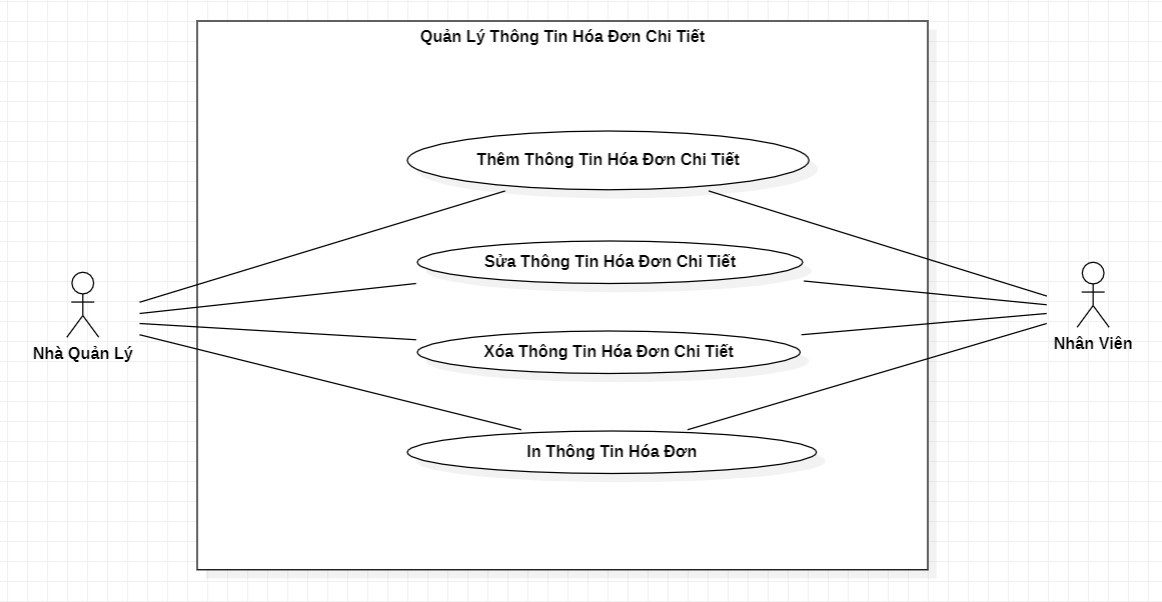
* Quản lý khách hàng



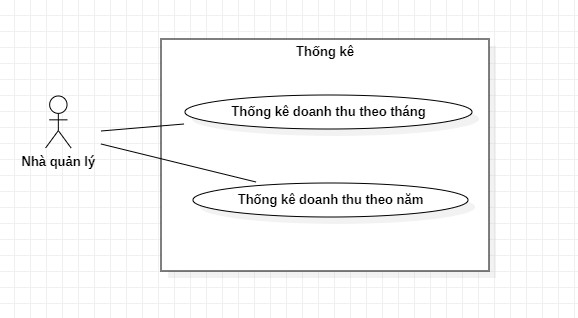
* Quản lý hóa đơn



* Quản lý hóa đơn chi tiết

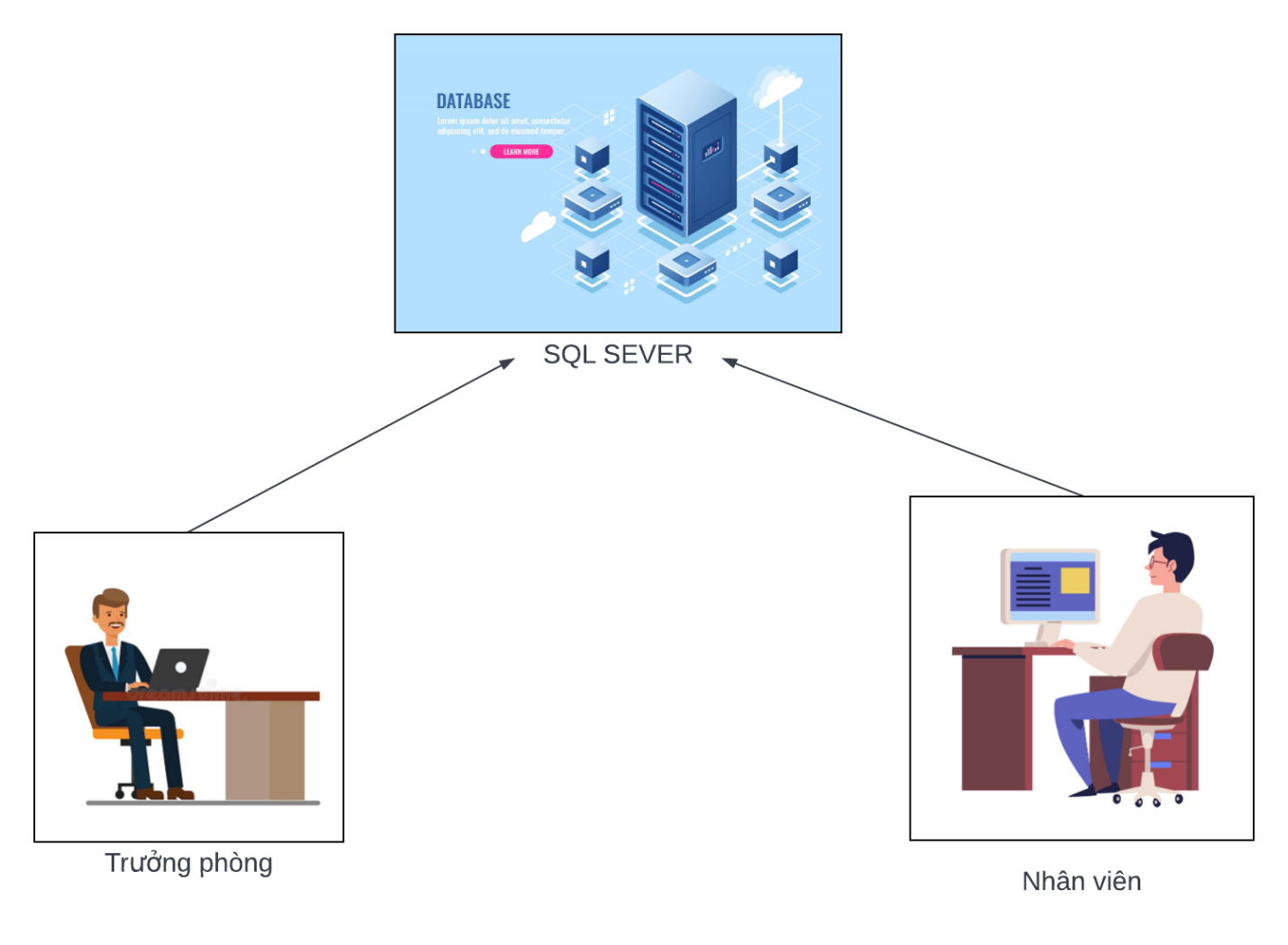


* Quản lý Thống kê



# Thiết kế

## Mô hình triển khai

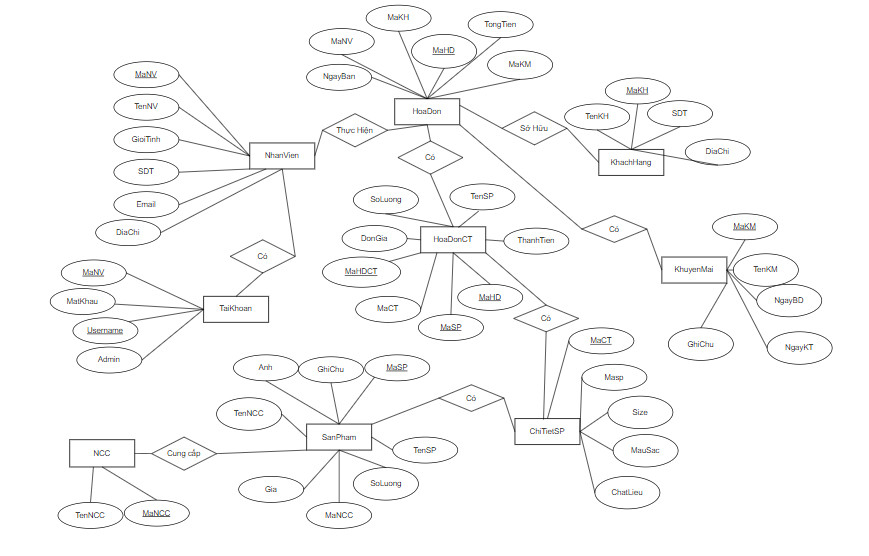


Quản lý

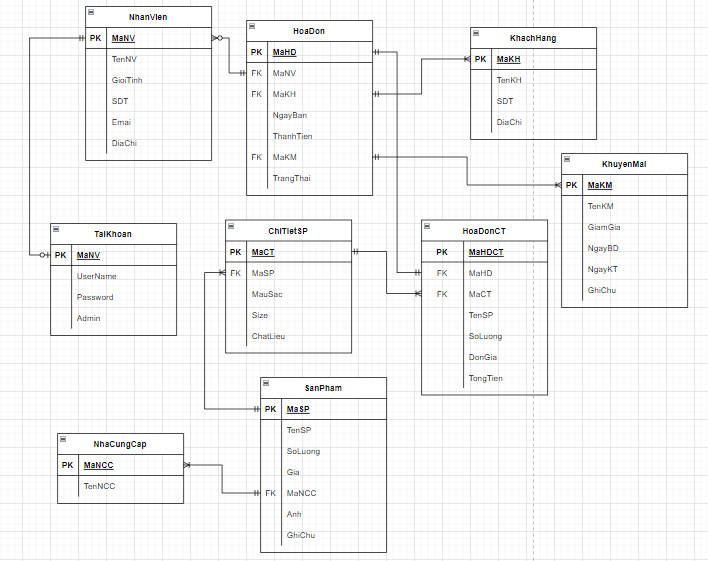
## Thiết kế CSDL

### Sơ đồ quan hệ thực thể

#### ERD Diagram level 1



#### ERD Diagram level 2



### Thiết kế chi tiết các thực thể

#### Nhân viên

|  |  |  |  |
| --- | --- | --- | --- |
| Tên cột | Kiểu | Ràng buộc | Ghi chú |
| MaNV | NVARCHAR (50) | PK, NOT NULL | Mã nhân viên |
| TenNV | NVARCHAR (50) | NOT NULL | Tên nhân viên |
| GioiTinh | BIT | DEFAULT 0 | Giới tính |
| SDT | NVARCHAR (50) | NOT NULL | Số điện thoại nhân viên |
| Email | NVARCHAR (50) | NOT NULL | Email nhân viên |
| Hinh | NVARCHAR (50) | NOT NULL | Hình ảnh nhân viên |
| DiaChi | NVARCHAR (255) | NOT NULL | Địa chỉ nhân viên |

#### Tài khoản

|  |  |  |  |
| --- | --- | --- | --- |
| Tên cột | Kiểu | Ràng buộc | Ghi chú |
| MaNV | NVARCHAR (50) | PK, NOT NULL | Mã nhân viên |
| Username | NVARCHAR (50) | PK, NOT NULL | Tên đăng nhập |
| Pass | NVARCHAR (50) | NOT NULL | Mật khẩu |
| role | bit | NOT NULL | Quyền |

#### Nhà cung cấp

|  |  |  |  |
| --- | --- | --- | --- |
| Tên cột | Kiểu | Ràng buộc | Ghi chú |
| MaNCC | NVARCHAR (50) | PK, NOT NULL | Mã nhà cung cấp |
| TenNCC | NVARCHAR (50) | NOT NULL | Tên nhà cung cấp |

#### Khách hàng

|  |  |  |  |
| --- | --- | --- | --- |
| Tên cột | Kiểu | Ràng buộc | Ghi chú |
| MaKH | NVARCHAR (50) | PK, NOT NULL | Mã khách hàng |
| TenKH | NVARCHAR (50) | NOT NULL | Tên khách hàng |
| SDT | NVARCHAR (50) | NOT NULL | Số điện thoại |
| Diachi | NVARCHAR (255) | NOT NULL | Địa chỉ khách hàng |

#### Khuyến mãi

|  |  |  |  |
| --- | --- | --- | --- |
| Tên cột | Kiểu | Ràng buộc | Ghi chú |
| MaKM | NVARCHAR (50) | PK, NOT NULL | Mã khuyến mãi |
| TenKM | NVARCHAR(255) | NOT NULL | Tên khuyến mãi |
| GiamGia | FLOAT | NOT NULL | Giảm giá |
| NgayBD | DATE | NOT NULL | Ngày bắt đầu |
| NgayKT | DATE | NOT NULL | Ngày kết thúc |
| GhiChu | NVARCHAR(255) | NULL | Ghi chú |

#### Sản phẩm

|  |  |  |  |
| --- | --- | --- | --- |
| Tên cột | Kiểu | Ràng buộc | Ghi chú |
| MaSP | NVARCHAR (50) | PK, NOT NULL | Mã sản phẩm |
| MaNCC | NVARCHAR (50) | FK,NOT NULL | Mã nhà cung cấp |
| TenSP | NVARCHAR (50) | NOT NULL | Tên sản phẩm |
| SoLuong | INT | NOT NULL | Số lượng |
| Anh | NVARCHAR (50) | NOT NULL | Ảnh |
| GhiChu | NVARCHAR (50) | NULL | Ghi chú |

#### Chi tiết sản phẩm

|  |  |  |  |
| --- | --- | --- | --- |
| Tên cột | Kiểu | Ràng buộc | Ghi chú |
| MaCT | NVARCHAR (50) | PK, NOT NULL | Mã chi tiết sản phẩm |
| MaSP | NVARCHAR (50) | FK,NOT NULL | Mã sản phẩm |
| Size | FLOAT | NOT NULL | Size giày(Cỡ giày) |
| MauSac | NVARCHAR (50) | NOT NULL | Màu sắc |
| ChatLieu | NVARCHAR (50) | NOT NULL | Chất liệu |
| Giá | FLOAT | NOT NULL | Giá |

#### Hóa đơn

|  |  |  |  |
| --- | --- | --- | --- |
| Tên cột | Kiểu | Ràng buộc | Ghi chú |
| MaHD | NVARCHAR (50) | PK | Mã Hóa Đơn |
| MaNV | NVARCHAR (50) | NOT NULL | Mã Nhân Viên |
| MaKH | NVARCHAR (50) | NOT NULL | Mã Khách Hàng |
| NgayBan | DATE | NOT NULL | Ngày Bán |
| TongTien | FLOAT | NOT NULL | Tổng Tiền |
| MaKM | NVARCHAR(50) | NOT NULL | Mã Khuyến Mãi |
| TrangThai | BIT | NOT NULL | Trạng Thái |

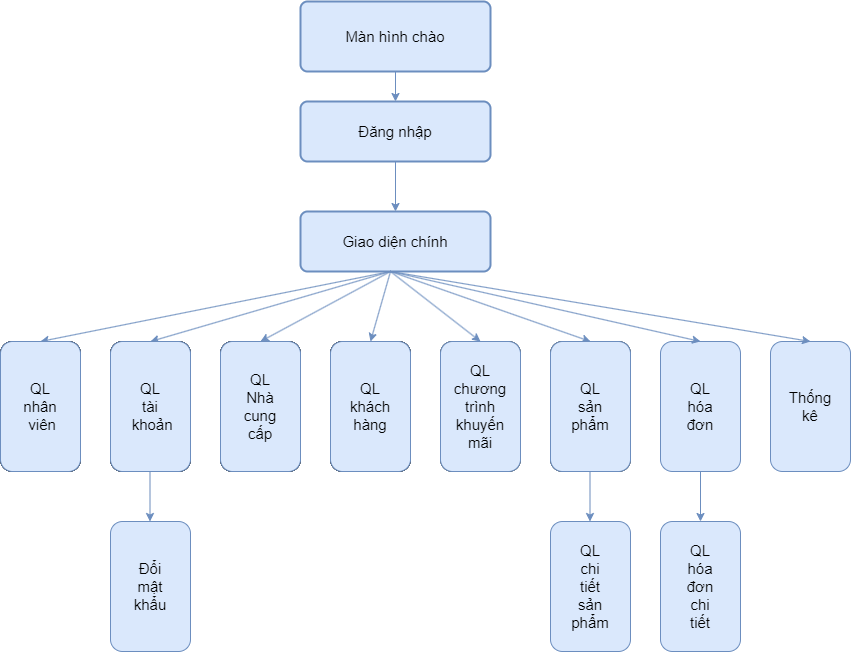
#### Hóa đơn chi tiết

|  |  |  |  |
| --- | --- | --- | --- |
| Tên cột | Kiểu | Ràng buộc | Ghi chú |
| MaHD | NVARCHAR (50) | PK | Mã Hóa Đơn |
| MaCT | NVARCHAR (50) | NOT NULL | Mã hóa đơn chi tiết |
| TenSP | NVARCHAR (50) | NOT NULL | Tên sản phẩm |
| SoLuong | INT | NOT NULL | Số lượng |
| Gia | FLOAT | NOT NULL | Giá |
| TongTien | FLOAT | NOT NULL | Tổng Tiền |

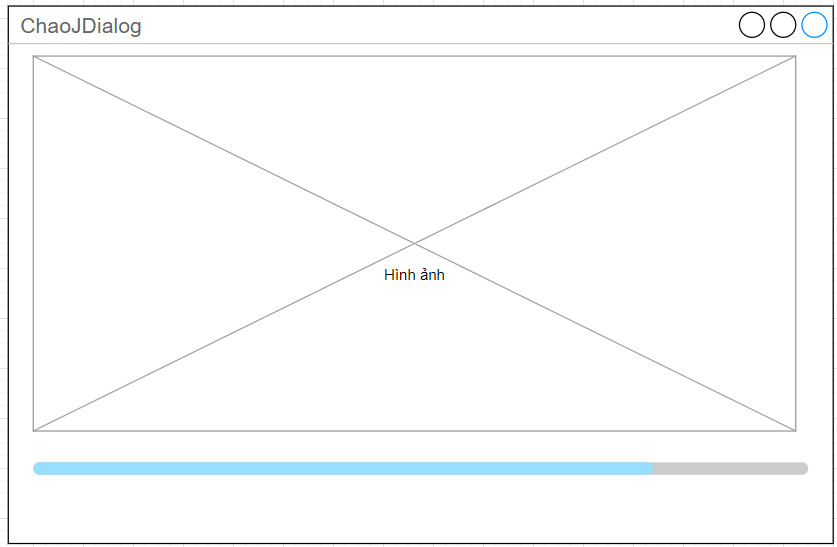
## Thiết kế giao diện

### Phác thảo giao diện

#### Sơ đồ tổ chức giao diện



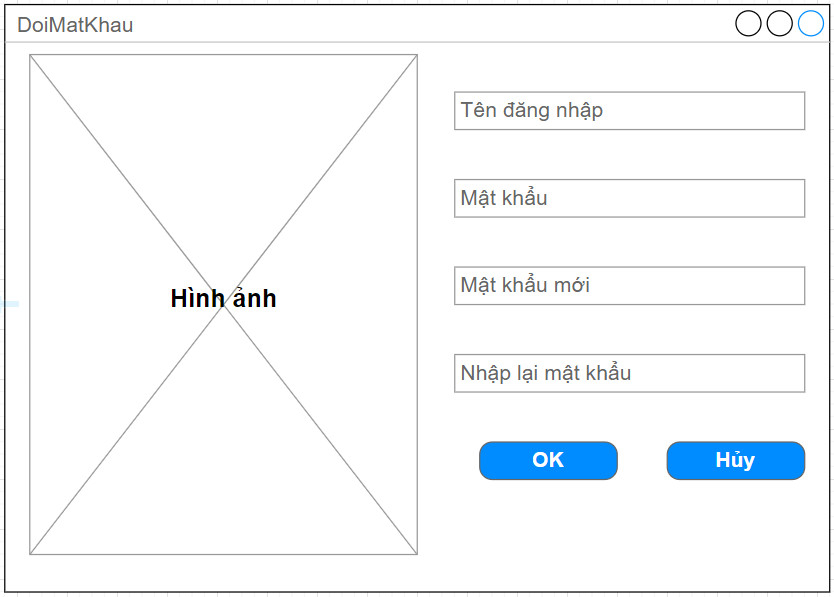
#### Cửa sổ chào



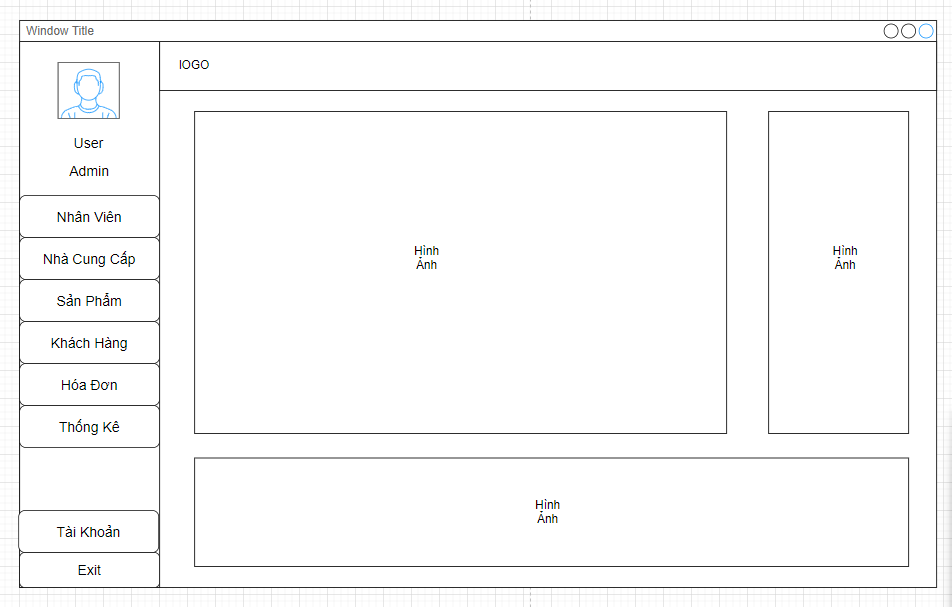
#### Đăng nhập



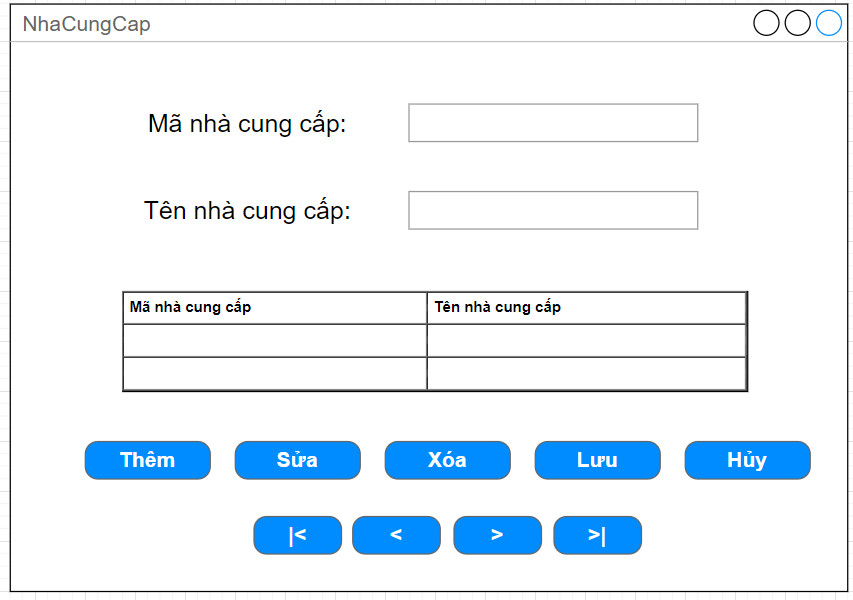
#### Đổi mật khẩu



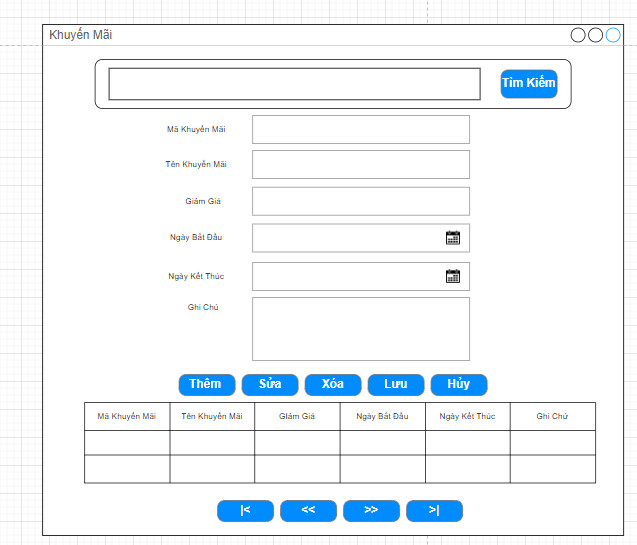
#### Cửa sổ chính (Sửa)



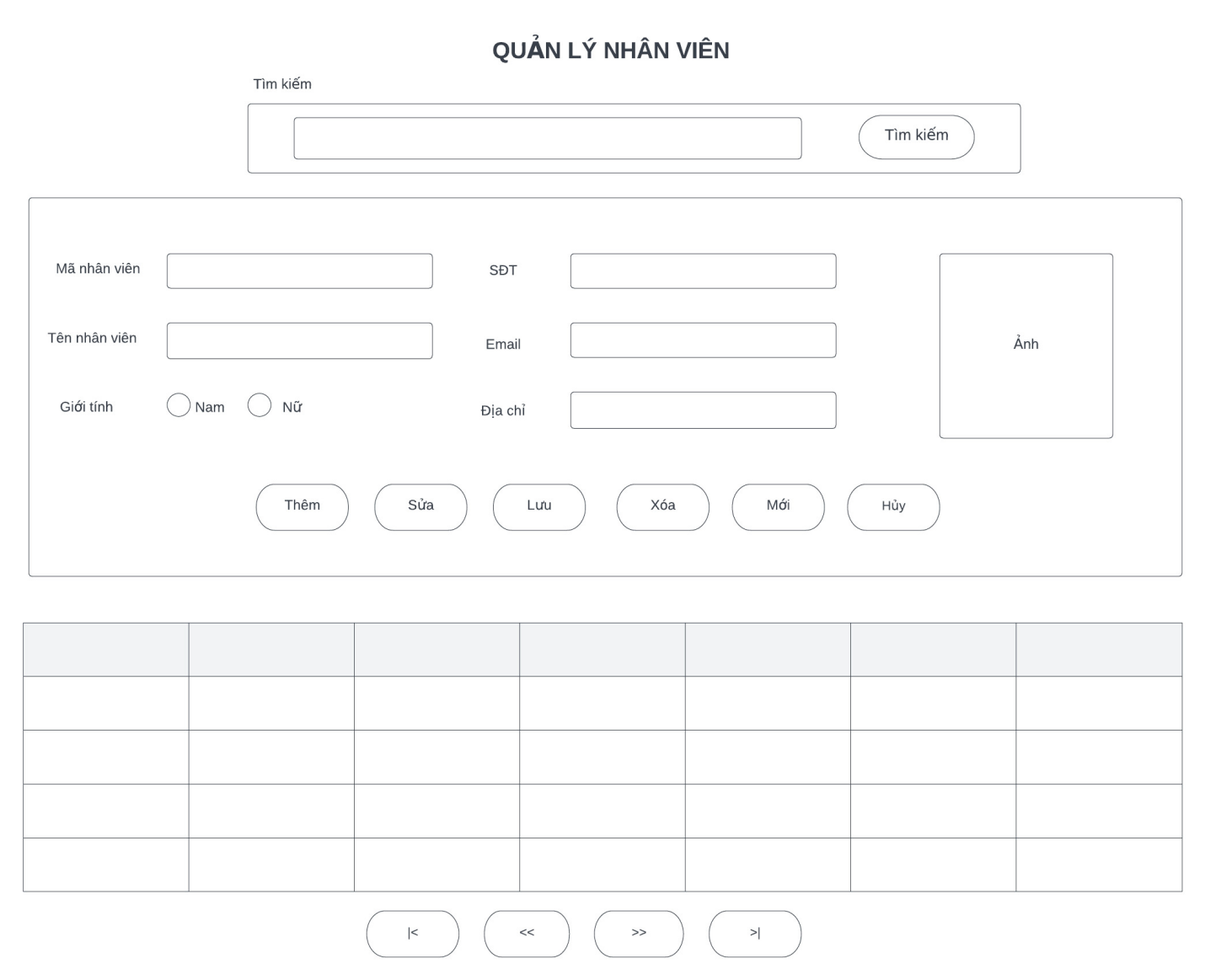
#### Nhà cung cấp



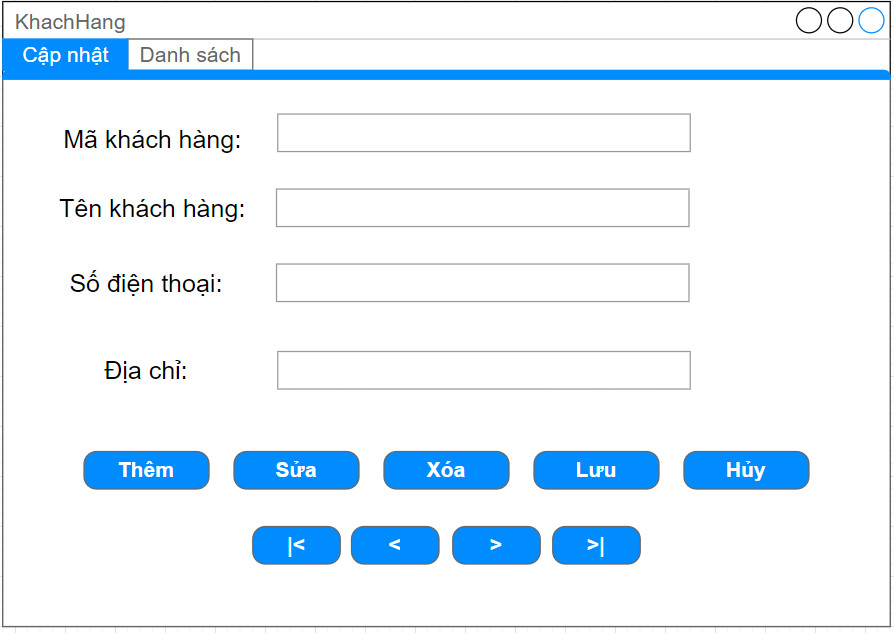
#### Chương trình khuyến mãi

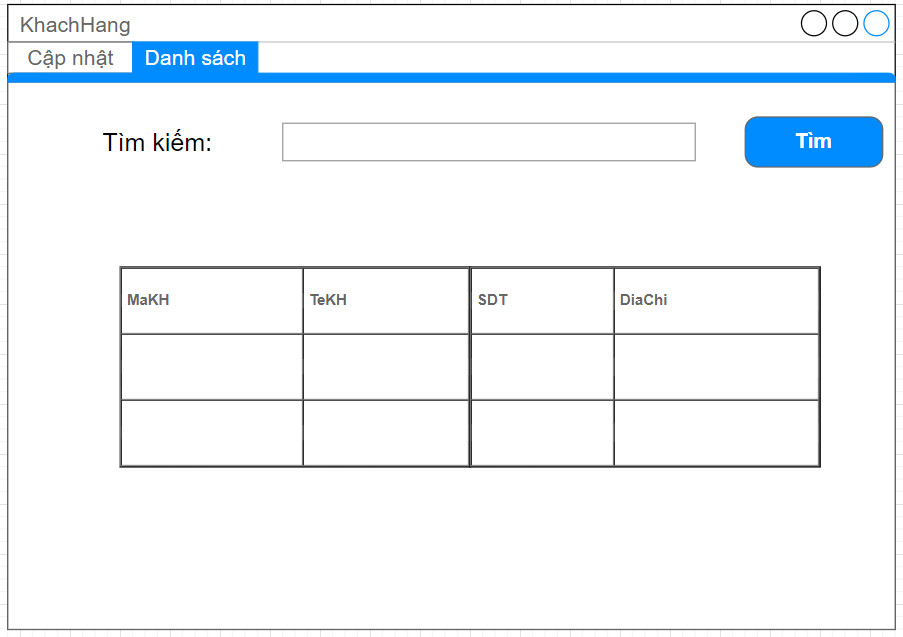


#### Nhân viên

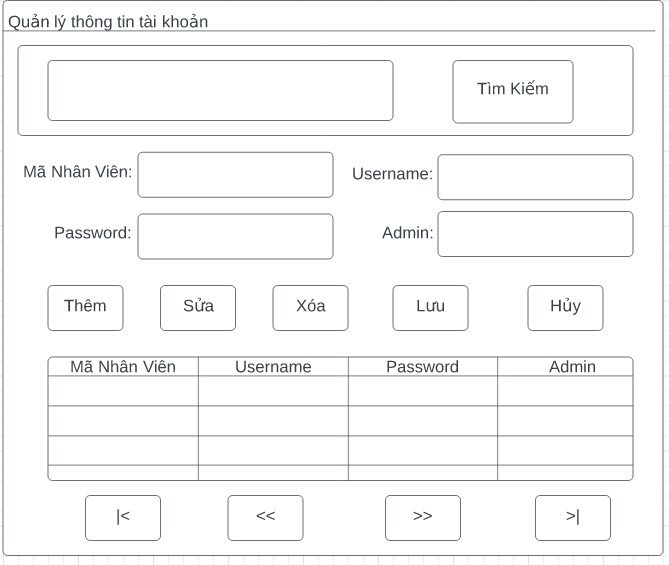


#### Khách hàng

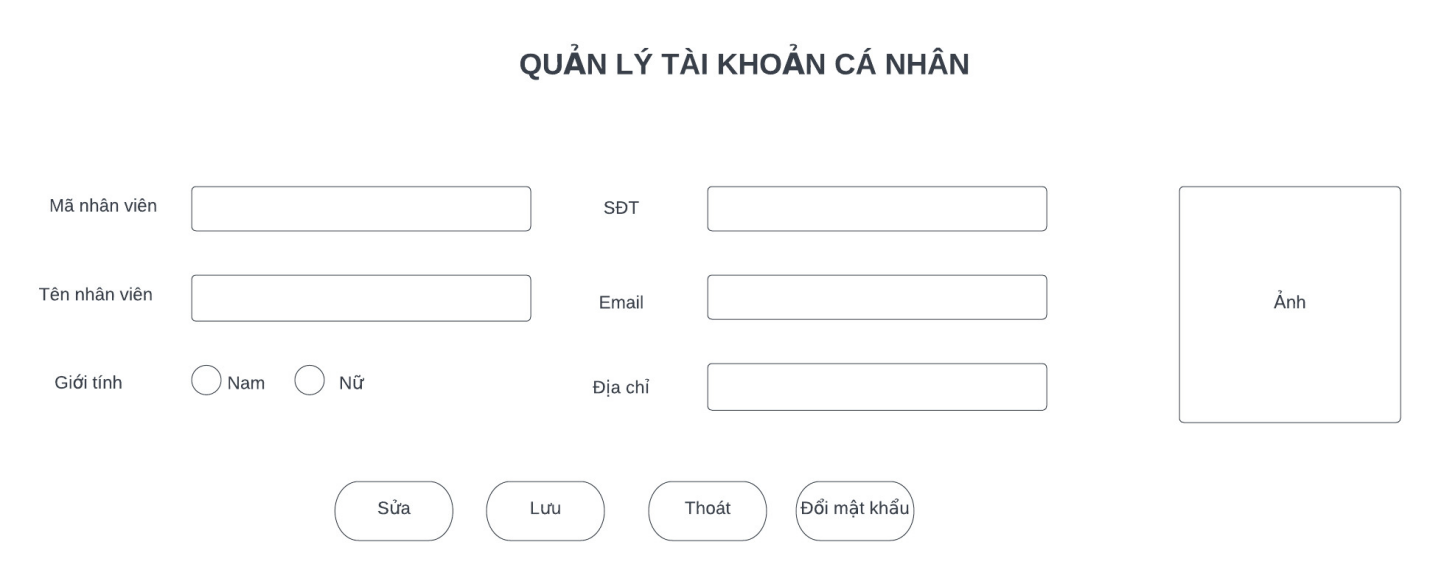




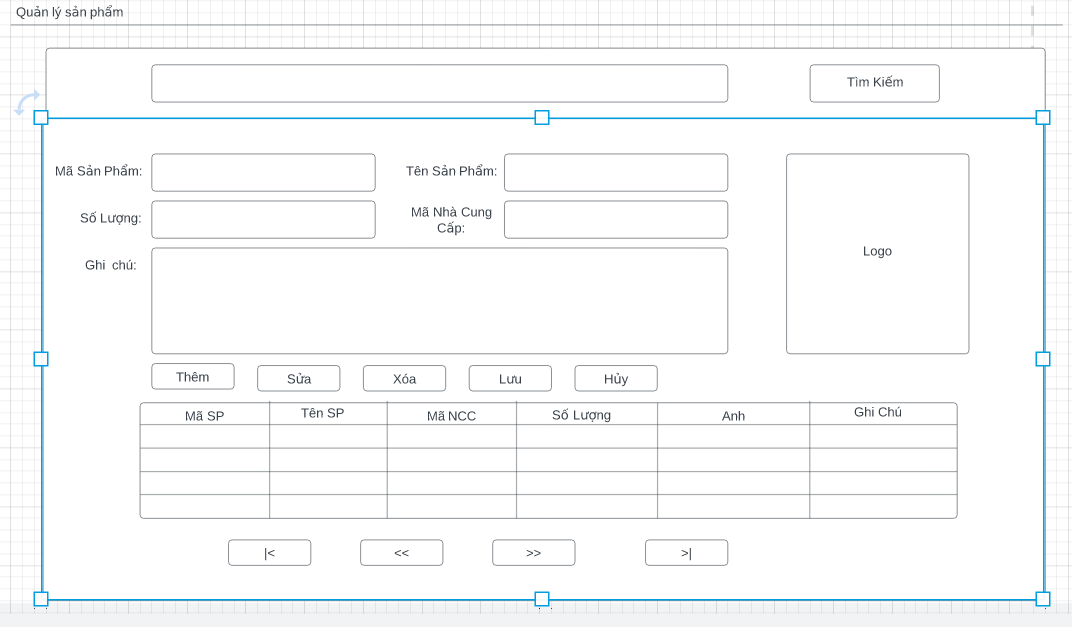
#### Tài khoản



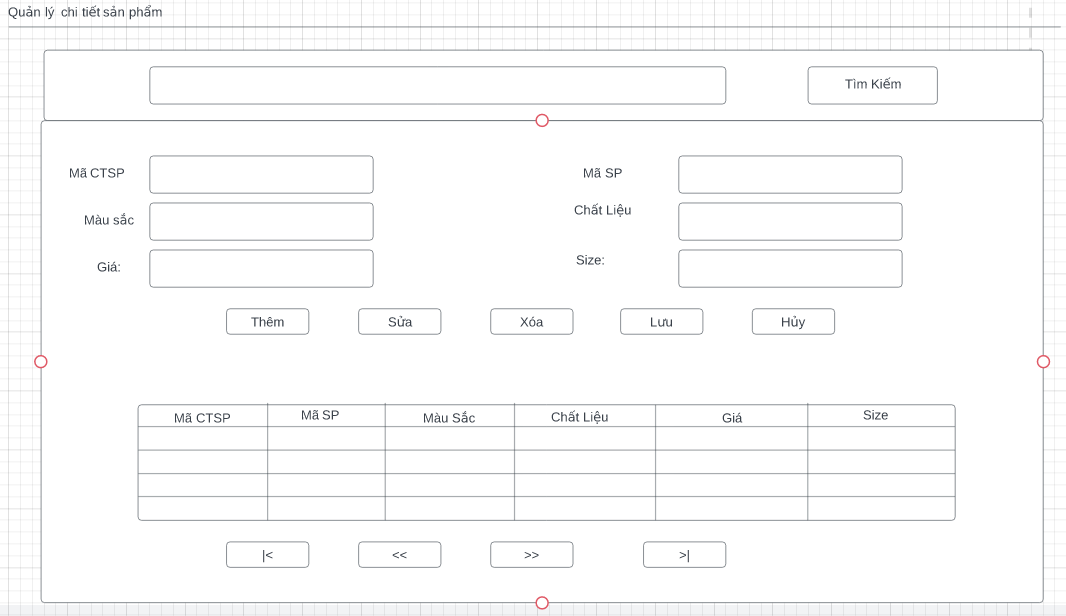
#### Tài khoản cá nhân



#### Sản phẩm



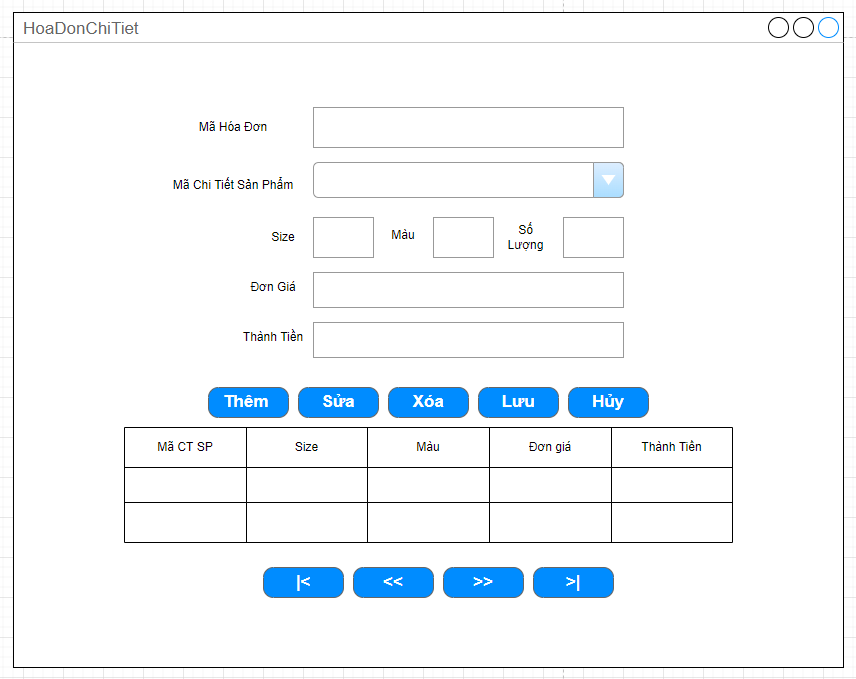
#### Chi tiết sản phẩm



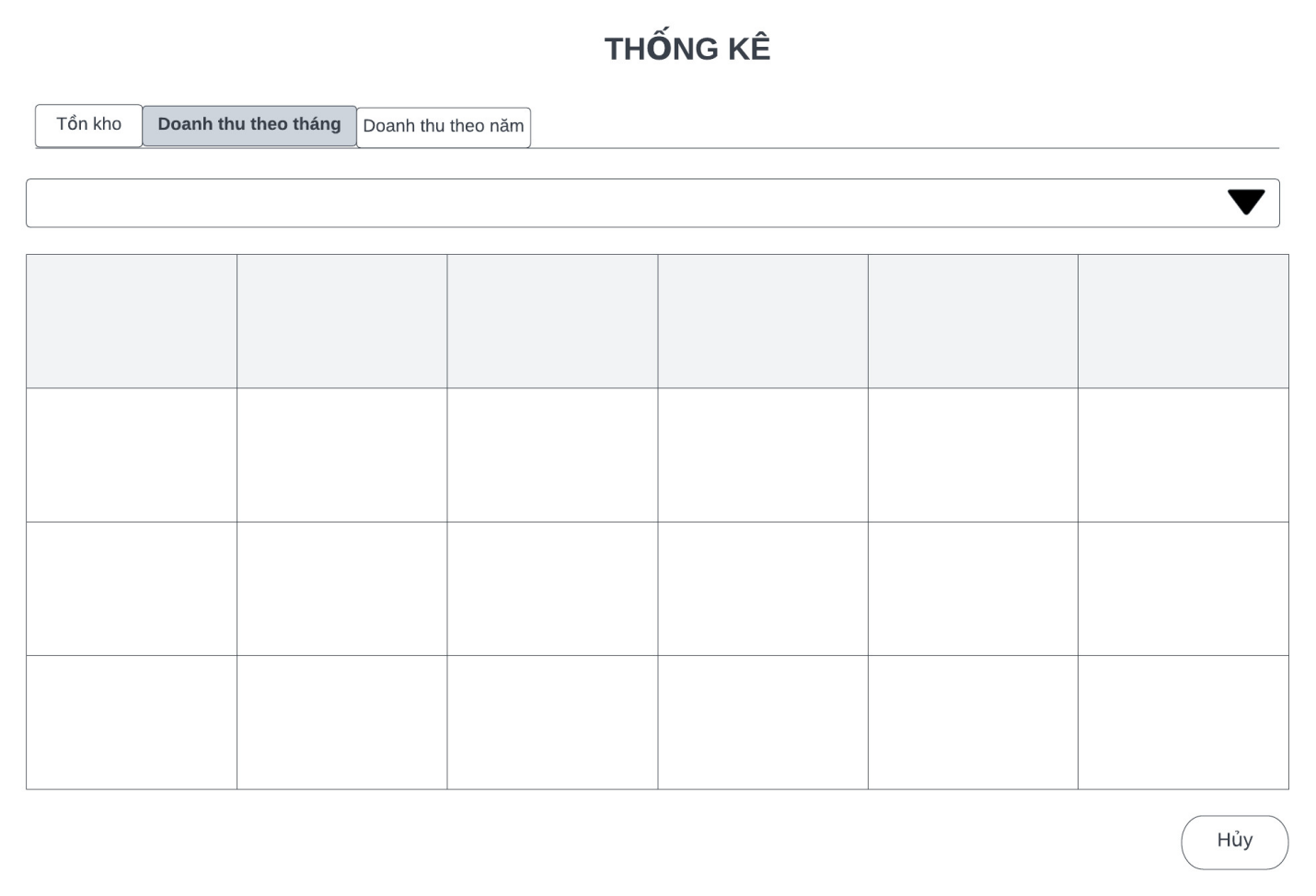
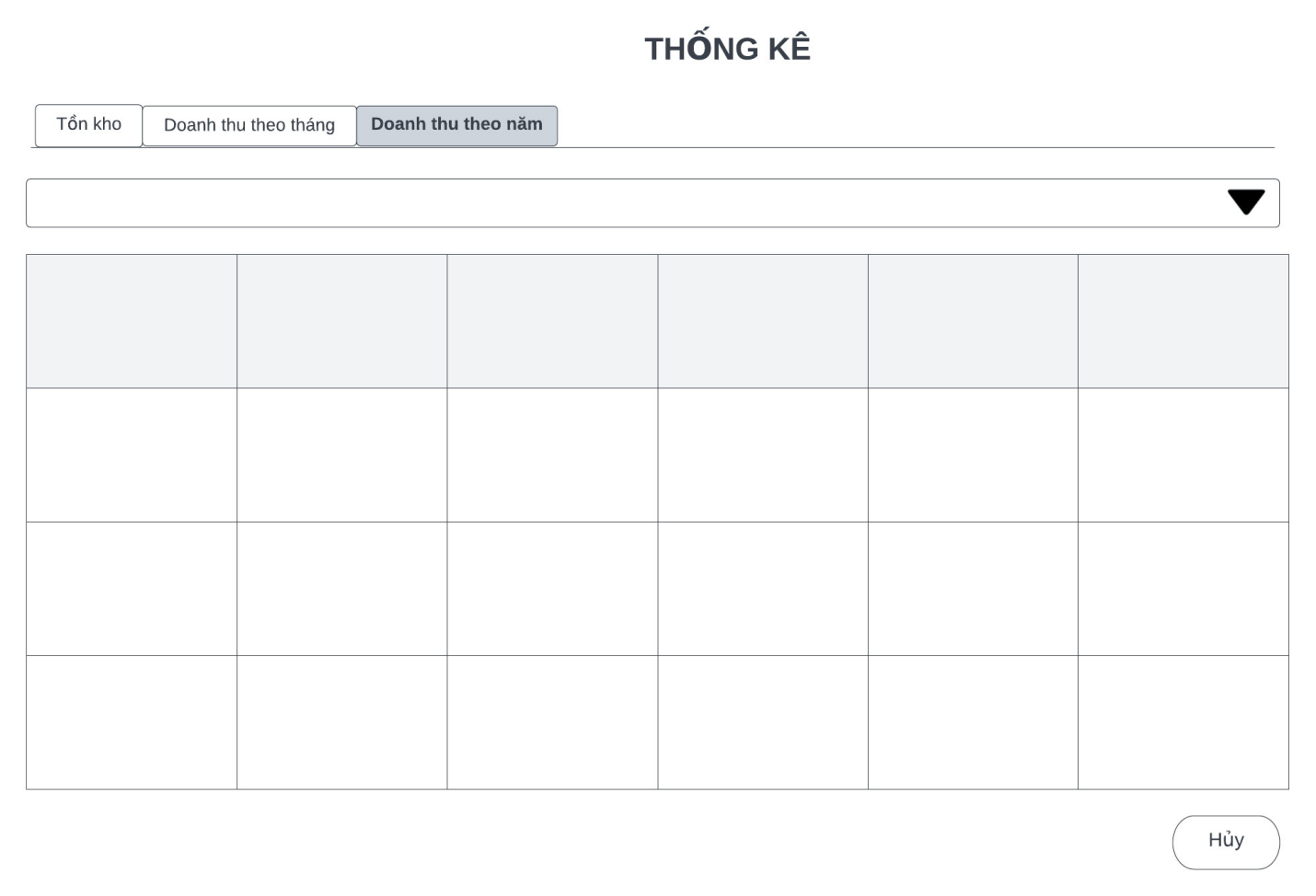
#### Hóa đơn



#### Hóa đơn chi tiết (sửa)

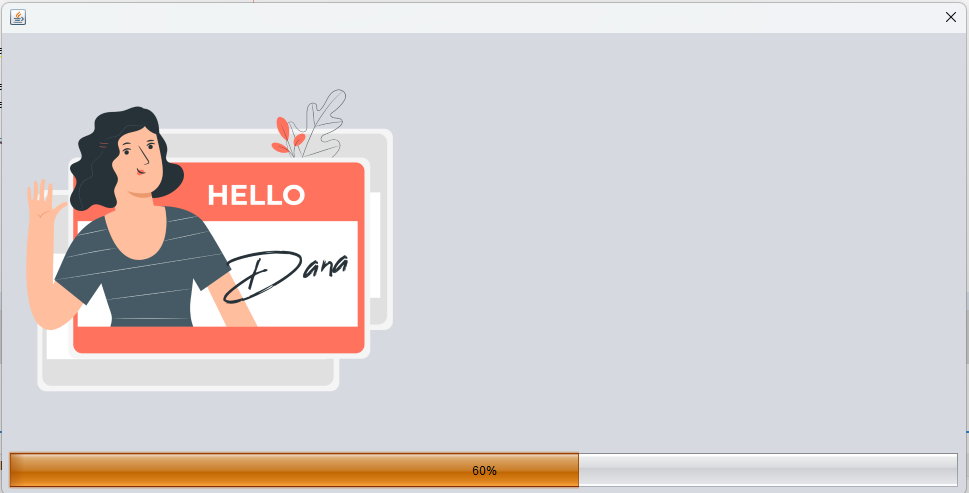


#### Thống kê

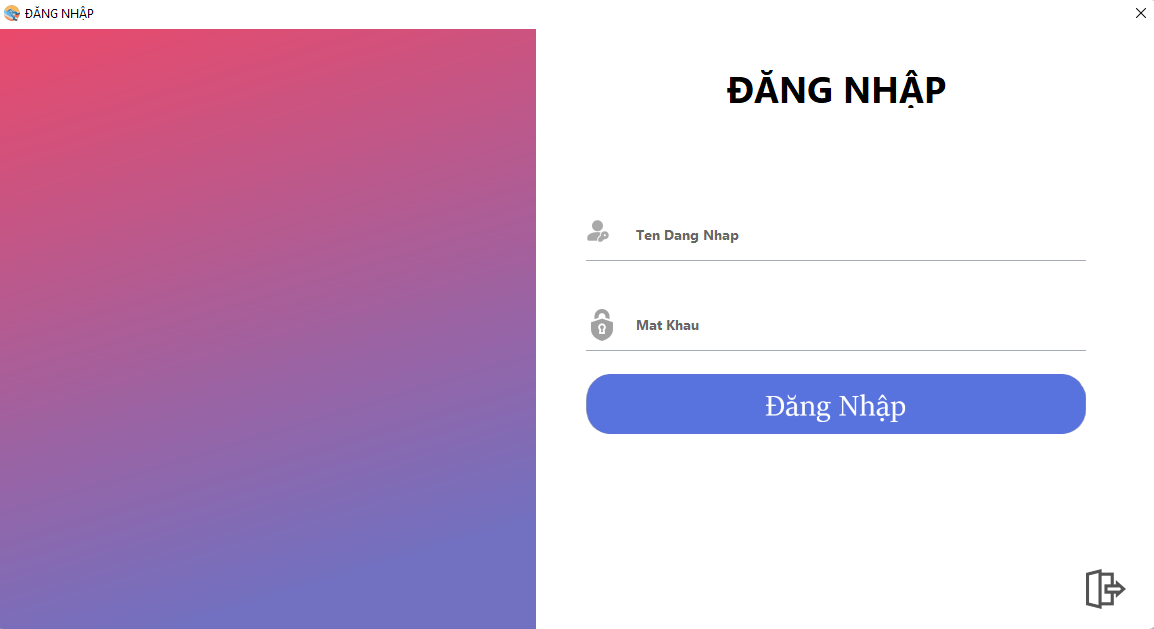
### Thiết kế giao diện cho các chức năng nghiệp vụ

#### Thiết kế Form Chào (Minh)



|  |  |  |  |
| --- | --- | --- | --- |
|  | Component | Kiểu | Thuộc tính |
| 1 | ChaoJDialog | Panel | defaultCloseOperation:DISPONSE |
| 2 | prbChao | Progress Bar | Text: 0% |
| 3 | lblAnh | Label |  |

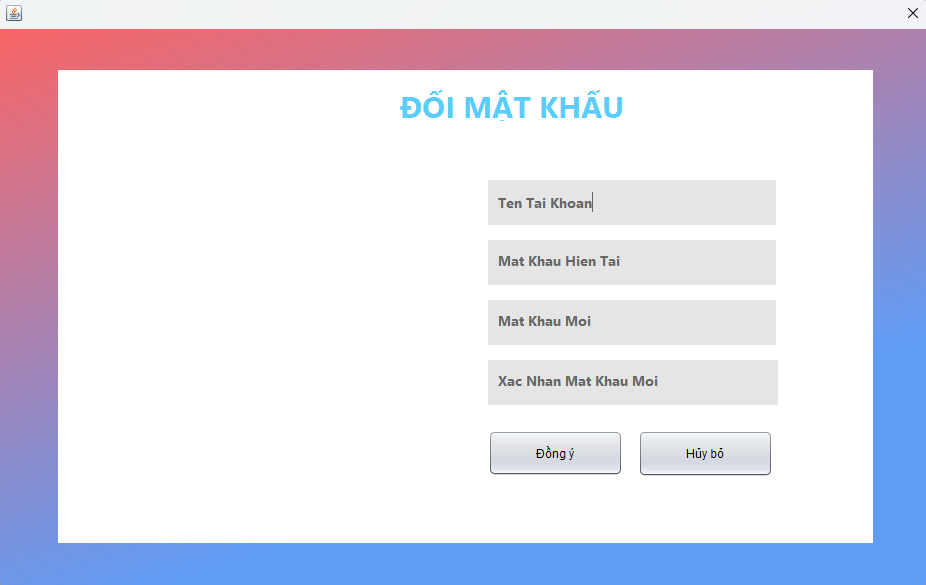
#### Thiết kế Form đăng nhập



|  |  |  |  |
| --- | --- | --- | --- |
|  | Component | Kiểu | Thuộc tính |
| 1 | Dangnhap | Panel | Title: Đăng Nhập  defaultCloseOperation:DISPONSE |
| 2 | lblTitle | JLabel | Text: Đăng Nhập |
| 3 | lblTenDangNhap | Label | Text: Tên Đăng Nhập |
| 4 | txtTenDangNhap | JTextPane |  |
| 5 | lblMatKhau | Label | Text: Mật Khẩu |
| 6 | txtMatKhau | JTextPane |  |
| 7 | lblDangNhap | Label | Text: Đăng Nhập |
| 8 | btnDangNhap | JTextPane | Text: Đăng Nhập |
| 9 | lblOut | Label |  |

#### Thiết kế Form chính (Hải)

#### Thiết kế Form đổi mật khẩu



|  |  |  |  |
| --- | --- | --- | --- |
|  | Component | Kiểu | Thuộc tính |
| 1 | DoiMatKhau | Panel | Title: Đổi Mật Khẩu defaultCloseOperation:DISPONSE |
| 2 | lblTitle | JLabel | Text: Đổi Mật Khẩu |
| 3 | lblTenTaiKhoan | Label | Text: Tên Tài Khoản |
| 4 | txtTenTaiKhoan | JTextPane |  |
| 5 | lblMatKhau | Label | Text: Mật Khẩu Hiện Tại |
| 6 | txtMatKhau | JTextPane |  |
| 7 | lblMatKhau1 | Label | Text: Mật Khẩu Mới |
| 8 | txtMatKhauMoi | JTextPane |  |
| 9 | lblMatKhau2 | Label | Text: Xác Nhận Mật Khẩu Mới |
| 10 | txtMatKhau2 | JTextPane |  |
| 11 | btnDongY | Button | Text: Đồng Ý |
| 12 | btnHuyBo | Button | Text: Hủy Bỏ |

#### Thiết kế Form nhà cung cấp

|  |  |  |  |
| --- | --- | --- | --- |
|  | Component | Kiểu | Thuộc tính |
| 1 | NhaCungCap1 | Panel | Title: Nhà Cung Cấp defaultCloseOperation:DISPONSE |
| 2 | lblMaNCC | Label | Text: mã nhà cung cấp |
| 3 | lblTenNCC | Label | Text: Tên Nhà cung cấp |
| 4 | txtMaNCC | JTextPane |  |
| 5 | txtTenNCC | JTextPane |  |
| 6 | btnDongY | Button | Text: Đồng Ý |
| 7 | btnHuyBo | Button | Text: Hủy Bỏ |
| 8 | btnThem | Button | Text:Thêm |
| 9 | btnSua | Button | Text:Sửa |
| 10 | btnXoa | Button | Text:Xóa |
| 11 | btnLuu | Button | Text:Mới |
| 12 | btnHuy | Button | Text:Hủy |
| 13 | btnFirst | JButton | Text: |< |
| 14 | btnPrev | JButton | Text: << |
| 15 | btnNext | JButton | Text: >> |
| 16 | btnLast | JButton | Text: >| |
| 17 | tblNCC | JTable | Model: như hình |

#### Thiết kế Form chương trình khuyến mãi

|  |  |  |  |
| --- | --- | --- | --- |
|  | Component | Kiểu | Thuộc tính |
| 1 | KhuyenMai1 | Panel | Title: Khuyến mãi defaultCloseOperation:DISPONSE |
| 2 | txtTimKiem | JTextPane |  |
| 3 | btnTimKiem | Button | Text: Tìm kiếm |
| 4 | lblMaKM | Label | Text: Mã khuyến mãi |
| 5 | txtMaKM | JTextPane |  |
| 6 | lblNgayBD | Label | Text: Ngày bắt đầu |
| 7 | txtNgayBD | JTextPane |  |
| 8 | lblNgayKT | Label | Text: Ngày kết thúc |
| 9 | txtNgayKT | JTextPane |  |
| 10 | lblGiamGia | Label | Text: Giảm giá |
| 11 | txtGiamGia | JTextPane |  |
| 12 | lblTenKM | Label | Text: Tên khuyến mãi |
| 13 | txtTenKM | JTextPane |  |
| 14 | lblGhiChu | Label | Text: Ghi chú |
| 15 | txtGhiChu | JTextPane |  |
| 16 | btnThem | Button | Text:Thêm |
| 17 | btnSua | Button | Text:Sửa |
| 18 | btnXoa | Button | Text:Xóa |
| 19 | btnLuu | Button | Text:Mới |
| 20 | btnHuy | Button | Text:Hủy |
| 21 | btnFirst | JButton | Text: |< |
| 22 | btnPrev | JButton | Text: << |
| 23 | btnNext | JButton | Text: >> |
| 24 | btnLast | JButton | Text: >| |
| 25 | tblBang | JTable | Model: như hình |

#### Thiết kế Form Nhân viên

|  |  |  |  |
| --- | --- | --- | --- |
|  | Component | Kiểu | Thuộc tính |
| 1 | NhanVienJFrame | JFrame | Title: QUẢN LÝ NHÂN VIÊN defaultCloseOperation:DISPONSE |
| 2 | lblTitle | JLabel | Text: Quản Lý Nhân Viên |
| 3 | lblTimKiem | Label | Text: Tìm Kiếm |
| 4 | txtTimKiem | JTextPane |  |
| 5 | btnTimKiem | Button | Text: Tìm Kiếm |
| 6 | lblMaNhanVien | Label | Text: Mã Nhân Viên |
| 7 | txtMaNhanVien | JTextPane |  |
| 8 | lblTenNhanVien | Label | Text: Tên  Nhân Viên |
| 9 | txtTenNhanVien | JTextPane |  |
| 10 | lblGioiTinh | Label | Text: Giới Tính |
| 11 | rdoGioiTinh | Radio Button |  |
| 12 | lblSoDienThoai | Label | Text: Số Điện Thoại |
| 13 | txtSoDienThoai | JTextPane |  |
| 14 | lblDiaChi | Label | Text: Địa Chỉ |
| 15 | txtDiaChi | JTextPane |  |
| 16 | lblEmail | Label | Text: Email |
| 17 | txtEmail | JTextPane |  |
| 18 | lblHinh | JLabel | Text: Hình |
| 19 | btnThem | Button | Text:Thêm |
| 20 | btnSua | Button | Text:Sửa |
| 21 | btnXoa | Button | Text:Xóa |
| 22 | btnLuu | Button | Text:Mới |
| 23 | btnHuy | Button | Text:Hủy |
| 24 | btnFirst | JButton | Text: |< |
| 25 | btnPrev | JButton | Text: << |
| 26 | btnNext | JButton | Text: >> |
| 27 | btnLast | JButton | Text: >| |
| 28 | tblBang | JTable | Model: như hình   Height: 25 |

#### Thiết kế Form khách hàng

|  |  |  |  |
| --- | --- | --- | --- |
|  | Component | Kiểu | Thuộc tính |
| 1 | KhachHangJFrame | JFrame | Title: QUẢN LÝ KHÁCH HÀNG defaultCloseOperation:DISPONSE |
| 2 | lblTitle | JLabel | Text: Quản Lý Khách Hàng |
| 3 | lblTimKiem | Label | Text: Tìm Kiếm |
| 4 | txtTimKiem | JTextPane |  |
| 5 | btnTimKiem | Button | Text: Tìm Kiếm |
| 6 | lblMaKhachHang | Label | Text: Mã Khách Hàng |
| 7 | txtMaKhachHang | JTextPane |  |
| 8 | lblTenKhachHang | Label | Text: TênKhách Hàng |
| 9 | txtTenKhachHang | JTextPane |  |
| 10 | lblSoDienThoai | Label | Text: Số Điện Thoại |
| 11 | txtSoDienThoai | JTextPane |  |
| 12 | lblDiaChi | Label | Text: Địa Chỉ |
| 13 | txtDiaChi | JTextPane |  |
| 14 | btnThem | Button | Text:Thêm |
| 15 | btnSua | Button | Text:Sửa |
| 16 | btnXoa | Button | Text:Xóa |
| 17 | btnLuu | Button | Text:Mới |
| 18 | btnHuy | Button | Text:Hủy |
| 19 | btnFirst | JButton | Text: |< |
| 20 | btnPrev | JButton | Text: << |
| 21 | btnNext | JButton | Text: >> |
| 22 | btnLast | JButton | Text: >| |
| 23 | tblBang | JTable | Model: như hình   Height: 25 |

#### Thiết kế Form Tài khoản (Linh)

|  |  |  |  |
| --- | --- | --- | --- |
|  | Component | Kiểu | Thuộc tính |
| 1 | TaiKhoanJPanel | JPanel | Title: QUẢN LÝ TÀI KHOẢN  defaultCloseOperation:DISPONSE |
| 2 | lblTitle | JLabel | Text: Quản Lý Tài Khoản |
| 3 | lblTimKiem | Label | Text: Tìm Kiếm |
| 4 | txtTimKiem | JTextField |  |
| 5 | btnTimKiem | JButton | Text: Tìm Kiếm |
| 6 | lblMaNhanVien | JLabel | Text: Mã Nhân Viên |
| 7 | txtMaNhanVien | JJTextField |  |
| 8 | lblTenDangNhap | JLabel | Text: Tên đăng nhập |
| 9 | txtTenDangNhap | JTextfield |  |
| 10 | lblMatKhau | JLabel | Text: Mật khẩu |
| 11 | txtMatKhau | JTextfield |  |
| 12 | rdoNam | JRadio | Text: Nam |
| 13 | rdoNu | JRadio | Text: Nữ |
| 14 | btnThem | JButton | Text: Thêm |
| 15 | btnSua | JButton | Text: Sửa |
| 16 | btnHuy | JButton | Text: Hủy |
| 17 | btnLuu | JButton | Text: Lưu |
| 18 | btnFirst | JButton | Text: |< |
| 19 | btnPreV | JButton | Text: << |
| 20 | btnNext | JButton | Text: >> |
| 21 | btnLast | JButton | Text: >| |
| 22 | tblBang | JTable |  |

#### Thiết kế Form Tài khoản cá nhân

|  |  |  |  |
| --- | --- | --- | --- |
|  | Component | Kiểu | Thuộc tính |
| 1 | TaiKhoanCaNhanJPanel | JPanel | Title: QUẢN LÝ TÀI KHOẢN CÁ NHÂN defaultCloseOperation:DISPONSE |
| 2 | lblTitle | JLabel | Text: Quản Lý Tài Khoản Cá Nhân |
| 3 | lblMaNhanVien | Label | Text: Mã Nhân Viên |
| 4 | txtMaNhanVien | JTextPane |  |
| 5 | lblTenNhanVien | Label | Text: Tên  Nhân Viên |
| 6 | txtTenNhanVien | JTextPane |  |
| 7 | lblGioiTinh | Label | Text: Giới Tính |
| 8 | rdoGioiTinh | Radio Button |  |
| 9 | lblSoDienThoai | Label | Text: Số Điện Thoại |
| 10 | txtSoDienThoai | JTextPane |  |
| 11 | lblDiaChi | Label | Text: Địa Chỉ |
| 12 | txtDiaChi | JTextPane |  |
| 13 | lblEmail | Label | Text: Email |
| 14 | txtEmail | JTextPane |  |
| 15 | lblHinh | JLabel | Text: Hình |
| 16 | btnLuu | Button | Text:Lưu |
| 17 | btnSua | Button | Text:Sửa |
| 18 | btnThoat | Button | Text:Thoát |
| 19 | btnDoiMatKhau | Button | Text:Đổi mật khẩu |

#### Thiết kế Form sản phẩm (Tiến)

#### Thiết kế Form chi tiết sản phẩm (Tiến)

#### Thiết kế Form hóa đơn

|  |  |  |  |
| --- | --- | --- | --- |
|  | Component | Kiểu | Thuộc tính |
| 1 | HoaDon | JPanel |  |
| 2 | lblTitle | JLabel | Text:  QUẢN LÝ HÓA ĐƠN |
| 3 | lblTimKiem | Label | Text: Tìm Kiếm |
| 4 | txtTimKiem | JTextPane |  |
| 5 | btnTimKiem | Button | Text: Tìm Kiếm |
| 6 | tblHoaDon | JTable |  |
| 7 | lblMaHD | JLabe | Text: Mã Hóa đơn |
| 8 | txtMaHD | JTextField |  |
| 9 | lblMaNV | JLabe | Text: Mã nhân viên |
| 10 | cboMaNV | JComboBox |  |
| 11 | lblMaKH | JLabe | Text: Mã khách hàng |
| 12 | cboMaKH | JComboBox |  |
| 13 | lblMaKM | JLabe | Text: Mã khuyến mãi |
| 14 | cboMaKM | JComboBox |  |
| 15 | btnThemHD | JButton | Text:Thêm |
| 16 | btnSuaHD | JButton | Text:Sửa |
| 17 | btnXoaHD | JButton | Text:Xóa |
| 18 | btnLuuHD | JButton | Text:Mới |
| 19 | btnHuyHD | JButton | Text:Hủy |

#### Thiết kế Form hóa đơn chi tiết (Hải)

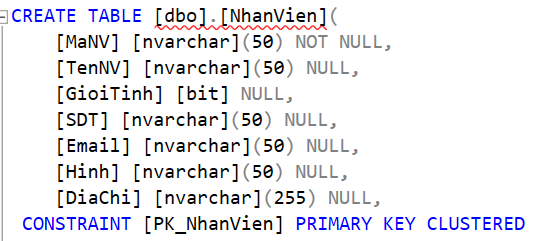
|  |  |  |  |
| --- | --- | --- | --- |
|  | Component | Kiểu | Thuộc tính |
| 1 | HoaDonChiTiet | JPanel |  |
| 2 | tblHoaDonCT | JTable |  |
| 3 | lblThongBao | JLabe | Text: Chưa Chọn Hóa Đơn |
| 4 | lblMaHD | JTextField |  |
| 5 | cboMaCT | JComboBox |  |
| 6 | lblTenSP | JLabel | Text: Tên sản phẩm |
| 7 | lblSoLuong | JLabe | Text: Số lượng |
| 8 | cboSoLuong | JComboBox |  |
| 9 | lblSize | JLabe | Text: Size |
| 10 | lblMau | JLabe | Text:Màu |
| 11 | lblDonGia | JLabe | Text:Đơn giá |
| 12 | txtDonGia | JTextField |  |
| 13 | lblThanhTien | JLabe | Text:Thành Tiền |
| 14 | txtDonGia | JTextField |  |
| 15 | btnThemHDCT | JButton | Text:Thêm |
| 16 | btnSuaHDCT | JButton | Text:Sửa |
| 17 | btnXoaHDCT | JButton | Text:Xóa |
| 18 | btnLuuHDCT | JButton | Text:Mới |
| 19 | btnHuyHDCT | JButton | Text:Hủy |

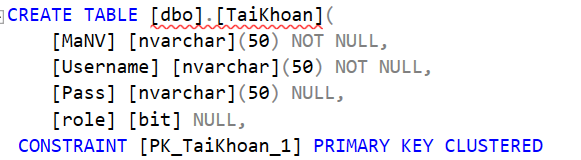
#### Thiết kế Form thống kê (Linh)

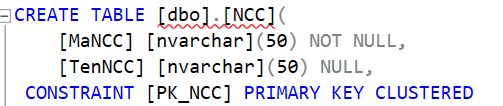
# Thực hiện viết mã

## Viết mã tạo CSDL

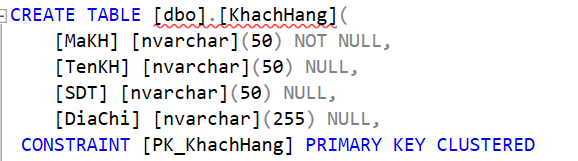
### Tạo CSDL

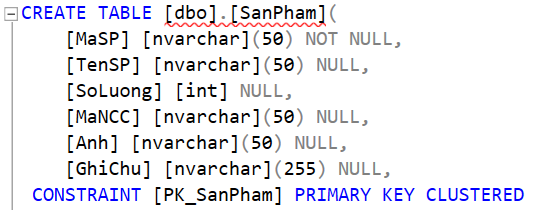






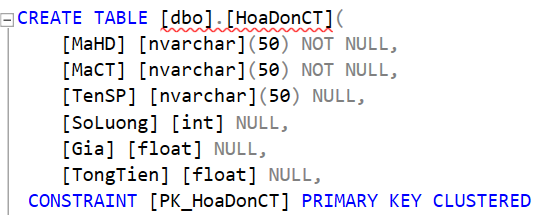












### SQL truy vấn và thao tác

#### Nhân viên

|  |  |
| --- | --- |
| **Hành Động** | **Câu Lệnh SQL** |
| **Thêm mới** | insert into NhanVien(MaNV,TenNV,GioiTinh,SDT,Email,Hinh,DiaChi)  Values(?,?,?,?,?,?,?) |
| **Cập Nhật theo mã** | UpdateNhanViensetTenNV=?,GioiTinh=?,SDT=?,Email=?,Hinh=?,DiaChi=? where MaNV = ? |
| **Xóa theo mã** | delete from NhanVien where MaNV = ? |
| **Truy vấn tất cả** | select \* from NhanVien |
| **Truy vấn theo mã** | select \* from NhanVien where MaNV=? |

#### Tài khoản

|  |  |
| --- | --- |
| Hành động | Câu lệnh SQL |
| Thêm mới | INSERT INTO TaiKhoan(MaNV, Username, Pass, Role)  VALUES ( ?,?,?,? ) |
| Cập nhật theo mã | UPDATE TaiKhoan SET Username = ? , Pass = ? , Role = ? WHERE MaNV = ? |
| Xóa theo mã | DELETE FROM TaiKhoan WHERE MaNV = ? |
| Truy vấn tất cả | SELECT \* FROM TaiKhoan |
| Truy vấn theo mã | SELECT \* FROM TaiKhoan WHERE MaNV = ? |

#### Nhà cung cấp

|  |  |
| --- | --- |
| Hành động | Câu lệnh SQL |
| Thêm mới | INSERT INTO NCC (MaNCC, TenNCC) VALUES ( ?, ? ) |
| Cập nhật theo mã | UPDATE NCC SET TenNCC = ? WHERE MaNCC = ? |
| Xóa theo mã | DELETE FROM NCC WHERE MaNCC = ? |
| Truy vấn tất cả | SELECT \* FROM NCC |
| Truy vấn theo mã | SELECT \* FROM NCC WHERE MaNCC = ? |

#### Khách hàng

|  |  |
| --- | --- |
| **Hành Động** | **Câu Lệnh SQL** |
| **Thêm mới** | insert into KhachHang (MaKH,TenKH,SDT,DiaChi)  Values(?,?,?,?) |
| **Cập Nhật theo mã** | update KhachHang set TenKH=?,SDT=?,DiaChi=? where MaKH = ? |
| **Xóa theo mã** | delete from KhachHang where MaKH = ? |
| **Truy vấn tất cả** | select \* from KhachHang |
| **Truy vấn theo mã** | select \* from KhachHang where MaKH=? |

#### Khuyến mãi

|  |  |
| --- | --- |
| Hành động | Câu lệnh SQL |
| Thêm mới | INSERT INTO KhuyenMai (MaKM, TenKM, GiamGia, NgayBD, NgayKT, GhiChu) VALUES (?,?,?,?,?,?) |
| Cập nhật theo mã | UPDATE KhuyenMai SET TenKM =?, GiamGia=?, NgayBD =?, NgayKT =?, GhiChu =? WHERE MaKM=? |
| Xóa theo mã | DELETE FROM KhuyenMai WHERE MaKM = ? |
| Truy vấn tất cả | SELECT \* FROM KhuyenMai |
| Truy vấn theo mã | SELECT \* FROM KhuyenMai WHERE MaKM = ? |

#### Sản phẩm

|  |  |
| --- | --- |
| Hành động | Câu lệnh SQL |
| Thêm mới | INSERT INTO SanPham(MaSP, TenSP, SoLuong, MaNCC, Anh, GhiChu) VALUES (?,?,?,?,?,?) |
| Cập nhật theo mã | UPDATE SanPham SET TenSP = ?, SoLuong= ?, MaNCC =?, Anh =?, GhiChu =? WHERE MaSP=? |
| Xóa theo mã | DELETE FROM SanPham WHERE MaSP = ? |
| Truy vấn tất cả | SELECT \* FROM SanPham |
| Truy vấn theo mã | SELECT \* FROM SanPham WHERE MaSP = ? |

#### Chi tiết sản phẩm

|  |  |
| --- | --- |
| Hành động | Câu lệnh SQL |
| Thêm mới | INSERT INTO ChiTietSP(MaCT, MaSP, Size, MauSac, ChatLieu, Gia) VALUES (?,?,?,?,?,?) |
| Cập nhật theo mã | UPDATE ChiTietSP SET MaSP =?, Size=?, MauSac =?, ChatLieu =?, Gia =? WHERE MaCT=? |
| Xóa theo mã | DELETE FROM ChiTietSP WHERE MaCT = ? |
| Truy vấn tất cả | SELECT \* FROM ChiTietSP |
| Truy vấn theo mã | SELECT \* FROM ChiTietSP WHERE MaCT = ? |

#### Hóa đơn

|  |  |
| --- | --- |
| Hành động | Câu lệnh SQL |
| Thêm mới | INSERT INTO HoaDon (MaHD, MaNV, MaKM, NgayBan, MaKH,ThanhTien,TrangThai)VALUES (?,?,?,?,?,?,?) |
| Cập nhật theo mã | UPDATE HoaDon SET MaNV =?, MaKM=?, NgayBan =?, MaKH =?, ThanhTien =?, TrangThai =? WHERE MaHD=? |
| Xóa theo mã | DELETE FROM HoaDon WHERE MaHD = ? |
| Truy vấn tất cả | SELECT \* FROM HoaDon |
| Truy vấn theo mã | SELECT \* FROM HoaDon WHERE MaHD = ? |

#### Hóa đơn chi tiết

|  |  |
| --- | --- |
| Hành động | Câu lệnh SQL |
| Thêm mới | INSERT INTO HoaDonCT (MaHD, MaCT, TenSP, SoLuong,Gia,TongTien)VALUES(?,?,?,?,?,?) |
| Cập nhật theo mã | UPDATE HoaDonCT SET MaCT =?, TenSP=?, SoLuong=?, Gia=?, TongTien=? WHERE MaHD=? |
| Xóa theo mã | DELETE FROM HoaDonCT WHERE MaHD = ? |
| Truy vấn tất cả | SELECT \* FROM HoaDonCT |
| Truy vấn theo mã | SELECT \* FROM HoaDonCT WHERE MaHD = ? |

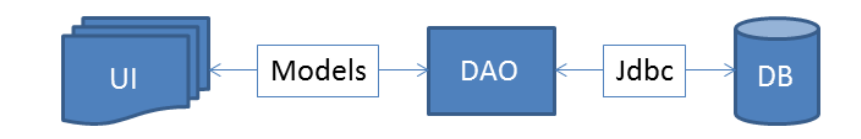
### Các thủ tục lưu tổng hợp thống kê

#### Procedure 1

#### Procedure 2

## Lập trình JDBC

Trong dự án này chúng ta chọn mô hình lập trình với CSDL như hình sau



Trong đó:

* UI: các thành phần giao diện
* DB: cơ sở dữ liệu
* Jdbc: là lớp tiện ích cung cấp các hàm tiện ích làm việc với CSDL thông qua câu lệnh sql hoặc lời gọi thủ tục lưu
* Model: là các lớp mô tả dữ liệu theo cấu trúc các bảng trong CSDL
* DAO: là các lớp thao tác và truy vấn dữ liệu. Nó có nhiệm vụ chuyển đổi Model sang SQL và ngược lại.

Theo mô hình này thì lập trình giao diện làm việc với CSDL thông qua các lớp model. Model như là một gói dữ liệu được sử dụng để giao tiếp giữa giao diện và các thành phần DAO. DAO mới thực sự là lập trình CSDL nó làm nhiệm vụ nhận model từ giao diện, chuyển đổi sang SQL và sử dụng Jdbc để thực hiện các thao tác dữ liệu. Ngược lại thì truy vấn dữ liệu từ CSDL, chuyển đổi thành model để hiển thị lên các thành phần giao diện.

Việc tách thành các phần riêng biết giúp quản lý dự án tốt hơn, dễ bảo trì nâng cấp hơn.

### Lớp hỗ trợ

Các lớp hỗ trợ chứa các phương thức tiện ích giúp chúng ta thực hiện các xử lý trong ứng dụng đơn giản hơn, ngắn gọn hơn, dễ hiểu hơn. Trong phần này chúng ta xây dựng 5 lớp như sau:

#### Lớp tiện ích Xdate

|  |
| --- |
| public class XDate {      static SimpleDateFormat DATE\_FORMATER = new SimpleDateFormat("dd-MM-yyyy");        public static Date toDate(String date, String pattern) {          try {              DATE\_FORMATER.applyPattern(pattern);              return DATE\_FORMATER.parse(date);          } catch (ParseException ex) {              throw new RuntimeException(ex);          }      }        public static String toString(Date date, String pattern) {              DATE\_FORMATER.applyPattern(pattern);              return DATE\_FORMATER.format(date);          }        public static Date now() {              return new Date();          }        public static Date addDays(Date date, long days) {              date.setTime(date.getTime() + days \* 24 \* 60 \* 60 \* 1000);              return date;          }  } |

#### Lớp tiện ích JdbcHelper

|  |
| --- |
| public class JdbcHelper {      private static String driver = "com.microsoft.sqlserver.jdbc.SQLServerDriver";      private static String dburl = "jdbc:sqlserver://localhost:1433;databaseName=QL\_CuaHangGiay;encrypt=false;trustServerCertificate=false;";      private static String user = "sa";      private static String pass = "123";        static {          try {              Class.forName(driver);          } catch (ClassNotFoundException e) {              throw new RuntimeException(e);          }      }        public static PreparedStatement getprepareStatement(String sql, Object... args) throws SQLException {          Connection connection = DriverManager.getConnection(dburl, user, pass);          PreparedStatement pstmt = null;          if (sql.trim().startsWith("{")) {              pstmt = connection.prepareCall(sql);          } else {              pstmt = connection.prepareStatement(sql);          }          for (int i = 0; i < args.length; i++) {              pstmt.setObject(i + 1, args[i]);          }          return pstmt;      }        public static void executeUpdate(String sql, Object... args) {          try {              PreparedStatement stmt = getprepareStatement(sql, args);              try {                  stmt.executeUpdate();              } finally {                  stmt.getConnection().close();              }          } catch (SQLException e) {              throw new RuntimeException(e);          }      }        public static ResultSet executeQuery(String sql, Object... args) {          try {              PreparedStatement stmt = getprepareStatement(sql, args);              return stmt.executeQuery();          } catch (SQLException e) {              throw new RuntimeException(e);          }      }        public static Object value(String sql, Object... args){          try {              ResultSet rs = executeQuery(sql, args);              if(rs.next()){                  return rs.getObject(0);              }              rs.getStatement().getConnection().close();          } catch (Exception e) {              throw new RuntimeException(e);          }          return null;      }  } |

#### Lớp tiện ích XImage

|  |
| --- |
| public class XImage {  public static Image getAppIcon() {  URL url = XImage.class.getResource("/com/nhom4/icon/logoo.png");  return new ImageIcon(url).getImage();  }  public static boolean save(File src) {  File dst = new File("/src/com/nhom4/icon/logoo.png", src.getName());  if (!dst.getParentFile().exists()) {  dst.getParentFile().mkdirs();  }  try {  Path from = Paths.get(src.getAbsolutePath());  Path to = Paths.get(dst.getAbsolutePath());  Files.copy(from, to, StandardCopyOption.REPLACE\_EXISTING);  return true;  } catch (Exception e) {  return false;  }  }  public static ImageIcon read(String fileName){  File path = new File("/src/com/nhom4/icon/logoo.png",fileName);  return new ImageIcon(path.getAbsolutePath());  }  } |

#### Lớp tiện ích MsgBox

|  |
| --- |
| public class MsgBox {      public static void alert(Component parent, String message) {          JOptionPane.showMessageDialog(parent, message, "Shop giày ABC", JOptionPane.INFORMATION\_MESSAGE);      }        public static boolean confirm(Component parent, String message) {          int result = JOptionPane.showConfirmDialog(parent, message, "Shop giày ABC", JOptionPane.YES\_NO\_OPTION, JOptionPane.QUESTION\_MESSAGE);          return result == JOptionPane.YES\_OPTION;      }        public static String prompt(Component parent, String message) {          return JOptionPane.showInputDialog(parent, message, "Shop giày ABC", JOptionPane.INFORMATION\_MESSAGE);      } |

#### Lớp tiện ích Auth

|  |
| --- |
| public class Auth{      public static TaiKhoan user = null;        public static void clear(){          Auth.user = null;      }        public static boolean  islogin(){          return Auth.user !=null;      }        public static boolean isManager(){          return Auth.islogin() && user.getRole();      }        public static String isMaNV(){          return user.getMaNV();      }  } |

### Model class - Các lớp mô tả dữ liệu

#### Nhân viên

|  |
| --- |
| package com.nhom4.entity;  public class NhanVien {      private String maNV;      private String tenNV;      private boolean gioiTinh = true;      private String SDT;      private String email;      private String hinh;      private String diaChi;      @Override          public String toString() {              return this.tenNV;          }  Constructor  getters/setters     } |

#### Tài khoản

|  |
| --- |
| public class TaiKhoan {      private String maNV;      private String username;      private String pass;      private boolean role = false;        @Override          public String toString() {              return this.username;  Constructor  getters/setters  }  } |

#### Nhà cung cấp

|  |
| --- |
| package com.nhom4.entity;  public class NhaCungCap {          private String maNCC;          private String tenNCC;            @Override          public String toString() {              return this.tenNCC;          }      Constructor      getters/setters  } |

#### Khách hàng

|  |
| --- |
| package com.nhom4.entity;  public class KhachHang {      private String maKH;      private String tenKH;      private String SDT;      private String diaChi;        @Override          public String toString() {              return this.tenKH;          }      Constructor  getters/setters  } |

#### Sản phẩm

|  |
| --- |
| public class SanPham {      private String maSP;      private String tenSP;      private int soLuong;      private String maNCC;      private String anh;      private String ghiChu;        @Override          public String toString() {              return this.tenSP;          }      contructors      getters/setters  } |

#### Chi tiết sản phẩm

|  |
| --- |
| public class ChiTietSanPham {      private String maCT;      private String maSP;      private float size;      private String mauSac;      private String chatLieu;      private float gia;        @Override          public String toString() {              return this.maCT;          }      contructors      getters/setters  } |

#### Hóa đơn

|  |
| --- |
| public class HoaDon {      private String maHD;      private String maNV;      private String maKM;      private Date ngayBan;      private String maKH;      private float thanhTien;      private boolean trangThai;        @Override          public String toString() {              return this.maHD;          }  contructors      getters/setters  } |

#### Hóa đơn chi tiết

|  |
| --- |
| public class HoaDonChiTiet {      private String maHDCT;      private String maHD;      private String maCT;      private String tenSP;      private int soLuong;      private float gia;      private float tongTien;        @Override          public String toString() {              return this.tenSP;          }  contructors      getters/setters  } |

#### Khuyến mãi

|  |
| --- |
| package com.nhom4.entity;  import java.util.Date;  public class KhuyenMai {      private String maKM;      private String tenKM;      private float giamGia;      private Date ngayBD;      private Date ngayKT;      private String ghiChu;        @Override          public String toString() {              return this.tenKM;          }      contructors      getters/setters  } |

### DAO Class - Các lớp truy xuất dữ liệu

#### Lớp MainDAO

|  |
| --- |
| package com.nhom4.dao;  import java.util.List;  public abstract class MainDAO <Enttity, Key> {      public abstract void insert (Enttity entity);      public abstract void update (Enttity entity);      public abstract void delete (Key id);      public abstract List<Enttity> selectAll();      public abstract Enttity selectById(Key id);      public abstract List<Enttity> selectBySql(String sql, Object...args);  } |

#### Lớp DAO làm việc với Bảng Nhân viên

|  |
| --- |
| package com.nhom4.dao;  import com.nhom4.entity.NhanVien;  import com.nhom4.utils.JdbcHelper;  import java.util.ArrayList;  import java.util.List;  import java.sql.\*;  public class NhanVienDAO extends MainDAO <NhanVien, String> {      final String INSERT\_SQL = "insert into NhanVien(MaNV,TenNV,GioiTinh,SDT,Email,Hinh,DiaChi) Values(?,?,?,?,?,?,?)";      final String UPDATE\_SQL = "Update NhanVien set TenNV=?,GioiTinh=?,SDT=?,Email=?,Hinh=?,DiaChi=? where MaNV = ?";      final String DELETE\_SQL = "delete from NhanVien where MaNV = ?";      final String SELECT\_ALL\_SQL = "select \* from NhanVien";      final String SELECT\_By\_Id\_SQL = "select \* from NhanVien where MaNV=?";      @Override      public void insert(NhanVien entity) {          JdbcHelper.executeUpdate(INSERT\_SQL, entity.getMaNV(), entity.getTenNV(),entity.getGioiTinh(), entity.getSDT(), entity.getEmail(), entity.getHinh(), entity.getDiaChi());      }      @Override      public void update(NhanVien entity) {          JdbcHelper.executeUpdate(UPDATE\_SQL, entity.getTenNV(),entity.getGioiTinh(), entity.getSDT(), entity.getEmail(), entity.getHinh(), entity.getDiaChi(), entity.getMaNV());      }      @Override      public void delete(String id) {          JdbcHelper.executeUpdate(DELETE\_SQL, id);      }      @Override      public List<NhanVien> selectAll() {          return selectBySql(SELECT\_ALL\_SQL);      }      @Override      public NhanVien selectById(String id) {          List<NhanVien> list = selectBySql(SELECT\_By\_Id\_SQL, id);          if(list.isEmpty()){              return null;          }          return list.get(0);      }      @Override      public List<NhanVien> selectBySql(String sql, Object... args) {          List<NhanVien> list = new ArrayList<NhanVien>();          try {              ResultSet rs = JdbcHelper.executeQuery(sql, args);              while (rs.next()) {                 NhanVien entity = new NhanVien();                 entity.setMaNV(rs.getString("MaNV"));                 entity.setTenNV(rs.getString("TenNV"));                 entity.setGioiTinh(rs.getBoolean("GioiTinh"));                 entity.setSDT(rs.getString("SDT"));                 entity.setEmail(rs.getString("Email"));                 entity.setHinh(rs.getString("Hinh"));                 entity.setDiaChi(rs.getString("DiaChi"));                 list.add(entity);              }          } catch (Exception e) {              throw new RuntimeException(e);          }          return list;      }  } |

#### Lớp DAO làm việc với Bảng Tài khoản

|  |
| --- |
| package com.nhom4.dao;  import com.nhom4.entity.NhanVien;  import com.nhom4.entity.TaiKhoan;  import com.nhom4.utils.JdbcHelper;  import java.sql.ResultSet;  import java.util.AbstractList;  import java.util.ArrayList;  import java.util.List;  public class TaiKhoanDAO extends MainDAO<TaiKhoan, String> {      final String INSERT\_SQL = "INSERT INTO TaiKhoan(MaNV, Username, Pass, Role) VALUES ( ?,?,?,? ) ";      final String UPDATE\_SQL = "UPDATE TaiKhoan SET Username = ? , Pass = ? , Role =  ? WHERE MaNV = ?";      final String DELETE\_SQL = "DELETE FROM TaiKhoan WHERE MaNV = ?";      final String SELECT\_ALL\_SQL = "SELECT \* FROM TaiKhoan";      final String SELECT\_By\_Id\_SQL = "SELECT \* FROM TaiKhoan WHERE Username = ?";      @Override      public void insert(TaiKhoan entity) {          JdbcHelper.executeUpdate(INSERT\_SQL, entity.getMaNV(), entity.getUsername(),entity.getPass(), entity.getRole());      }      @Override      public void update(TaiKhoan entity) {          JdbcHelper.executeUpdate(UPDATE\_SQL, entity.getUsername(),entity.getPass(), entity.getRole(), entity.getMaNV());      }      @Override      public void delete(String id) {          JdbcHelper.executeUpdate(DELETE\_SQL, id);      }      @Override      public List<TaiKhoan> selectAll() {          return selectBySql(SELECT\_ALL\_SQL);      }      @Override      public TaiKhoan selectById(String id) {          List<TaiKhoan> list = selectBySql(SELECT\_By\_Id\_SQL, id);          if(list.isEmpty()){              return null;          }          return list.get(0);      }        public List<TaiKhoan> selectByKeyword(String user) {          String sql = "SELECT \* FROM TaiKhoan WHERE Username like ?";          return this.selectBySql(sql, "%"+user+"%");      }      @Override      public List<TaiKhoan> selectBySql(String sql, Object... args) {          List<TaiKhoan> list = new ArrayList<TaiKhoan>();          try {              ResultSet rs = JdbcHelper.executeQuery(sql, args);              while (rs.next()) {                 TaiKhoan entity = new TaiKhoan();                 entity.setMaNV(rs.getString("MaNV"));                 entity.setUsername(rs.getString("Username"));                 entity.setPass(rs.getString("Pass"));                 entity.setRole(rs.getBoolean("role"));                 list.add(entity);              }          } catch (Exception e) {              throw new RuntimeException(e);          }          return list;      }  } |

#### Lớp DAO làm việc với Bảng Nhà cung cấp

|  |
| --- |
| package com.nhom4.dao;  import com.nhom4.dao.MainDAO;  import com.nhom4.entity.NhaCungCap;  import com.nhom4.utils.JdbcHelper;  import java.sql.ResultSet;  import java.util.ArrayList;  import java.util.List;  public class NhaCungCapDAO extends MainDAO<NhaCungCap, String>{          final String INSERT\_SQL = "INSERT INTO NCC (MaNCC, TenNCC) VALUES (?,?)";          final String UPDATE\_SQL = "UPDATE NCC SET TenNCC = ? WHERE MaNCC = ?";          final String DELETE\_SQL = "DELETE FROM NCC WHERE MaNCC = ?";          final String SELECT\_ALL\_SQL = "SELECT \* FROM NCC";          final String SELECT\_By\_Id\_SQL = "SELECT \* FROM NCC WHERE MaNCC = ?";        @Override      public void insert(NhaCungCap entity) {          JdbcHelper.executeUpdate(INSERT\_SQL, entity.getMaNCC(), entity.getTenNCC());      }      @Override      public void update(NhaCungCap entity) {          JdbcHelper.executeUpdate(UPDATE\_SQL, entity.getTenNCC(), entity.getMaNCC());      }      @Override      public void delete(String id) {          JdbcHelper.executeUpdate(DELETE\_SQL, id);      }      @Override      public List<NhaCungCap> selectAll() {          return selectBySql(SELECT\_ALL\_SQL);      }      @Override      public NhaCungCap selectById(String id) {          List<NhaCungCap> list = selectBySql(SELECT\_By\_Id\_SQL, id);          if(list.isEmpty()){              return null;          }          return list.get(0);      }      @Override      public List<NhaCungCap> selectBySql(String sql, Object... args) {          List<NhaCungCap> list = new ArrayList<NhaCungCap>();          try {              ResultSet rs = JdbcHelper.executeQuery(sql, args);              while (rs.next()) {                 NhaCungCap entity = new NhaCungCap();                 entity.setMaNCC(rs.getString("MaNCC"));                 entity.setTenNCC(rs.getString("TenNCC"));                 list.add(entity);              }          } catch (Exception e) {              throw new RuntimeException(e);          }          return list;      }  } |

#### Lớp DAO làm việc với Bảng Khách hàng

|  |
| --- |
| package com.nhom4.dao;  import com.nhom4.entity.KhachHang;  import com.nhom4.utils.JdbcHelper;  import java.util.ArrayList;  import java.util.List;  import java.sql.ResultSet;  public class KhachHangDAO extends MainDAO<KhachHang, String> {      final String INSERT\_SQL = "insert into KhachHang (MaKH,TenKH,SDT,DiaChi) Values(?,?,?,?)";      final String UPDATE\_SQL = "update KhachHang set TenKH=?,SDT=?,DiaChi=? where MaKH = ?";      final String DELETE\_SQL = "delete from KhachHang where MaKH = ?";      final String SELECT\_ALL\_SQL = "select \* from KhachHang";      final String SELECT\_By\_Id\_SQL = "select \* from KhachHang where MaKH=?";        @Override      public void insert(KhachHang entity) {          JdbcHelper.executeUpdate(INSERT\_SQL, entity.getMaKH(), entity.getTenKH(),entity.getSDT(), entity.getDiaChi());      }      @Override      public void update(KhachHang entity) {          JdbcHelper.executeUpdate(UPDATE\_SQL, entity.getMaKH(), entity.getTenKH(),entity.getSDT(), entity.getDiaChi());      }      @Override      public void delete(String id) {          JdbcHelper.executeUpdate(DELETE\_SQL, id);      }      @Override      public List<KhachHang> selectAll() {          return selectBySql(SELECT\_ALL\_SQL);      }      @Override      public KhachHang selectById(String id) {          List<KhachHang> list = selectBySql(SELECT\_By\_Id\_SQL, id);          if(list.isEmpty()){              return null;          }          return list.get(0);      }      @Override      public List<KhachHang> selectBySql(String sql, Object... args) {          List<KhachHang> list = new ArrayList<KhachHang>();          try {              ResultSet rs = JdbcHelper.executeQuery(sql, args);              while (rs.next()) {                 KhachHang entity = new KhachHang();                 entity.setMaKH(rs.getString("MaKH"));                 entity.setTenKH(rs.getString("TenKH"));                 entity.setSDT(rs.getString("SDT"));                 entity.setDiaChi(rs.getString("DiaChi"));                 list.add(entity);              }          } catch (Exception e) {              throw new RuntimeException(e);          }          return list;      }  } |

#### Lớp DAO làm việc với Bảng Sản phẩm

|  |
| --- |
| package com.nhom4.dao;  import com.nhom4.entity.SanPham;  import com.nhom4.utils.JdbcHelper;  import java.sql.ResultSet;  import java.util.ArrayList;  import java.util.List;  public class SanPhamDAO extends MainDAO<SanPham, String>  {      final String INSERT\_SQL = "INSERT INTO SanPham(MaSP, TenSP, SoLuong, MaNCC, Anh, GhiChu) VALUES (?,?,?,?,?,?)";      final String UPDATE\_SQL = "UPDATE SanPham SET TenSP = ?, SoLuong= ?, MaNCC =?, Anh =?, GhiChu =? WHERE MaSP=?";      final String DELETE\_SQL = "DELETE FROM SanPham WHERE MaSP = ?";      final String SELECT\_ALL\_SQL = "SELECT \* FROM SanPham";      final String SELECT\_By\_Id\_SQL = "SELECT \* FROM SanPham WHERE MaSP = ?";      @Override      public void insert(SanPham entity) {          JdbcHelper.executeUpdate(INSERT\_SQL, entity.getMaSP(), entity.getTenSP(),entity.getSoLuong(), entity.getMaNCC(), entity.getAnh(), entity.getGhiChu());      }      @Override      public void update(SanPham entity) {          JdbcHelper.executeUpdate(UPDATE\_SQL, entity.getTenSP(),entity.getSoLuong(), entity.getMaNCC(), entity.getAnh(), entity.getGhiChu(), entity.getMaSP());      }      @Override      public void delete(String id) {          JdbcHelper.executeUpdate(DELETE\_SQL, id);      }      @Override      public List<SanPham> selectAll() {          return selectBySql(SELECT\_ALL\_SQL);      }      @Override      public SanPham selectById(String id) {          List<SanPham> list = selectBySql(SELECT\_By\_Id\_SQL, id);          if(list.isEmpty()){              return null;          }          return list.get(0);      }      @Override      public List<SanPham> selectBySql(String sql, Object... args) {          List<SanPham> list = new ArrayList<SanPham>();          try {              ResultSet rs = JdbcHelper.executeQuery(sql, args);              while (rs.next()) {                 SanPham entity = new SanPham();                 entity.setMaSP(rs.getString("MaSP"));                 entity.setTenSP(rs.getString("TenSP"));                 entity.setSoLuong(rs.getInt("SoLuong"));                 entity.setMaNCC(rs.getString("MaNCC"));                 entity.setAnh(rs.getString("Anh"));                 entity.setGhiChu(rs.getString("GhiChu"));                 list.add(entity);              }          } catch (Exception e) {              throw new RuntimeException(e);          }          return list;      }  } |

#### Lớp DAO làm việc với Bảng Chi tiết sản phẩm

|  |
| --- |
| package com.nhom4.dao;  import com.nhom4.entity.ChiTietSanPham;  import com.nhom4.utils.JdbcHelper;  import java.util.List;  import java.util.ArrayList;  import java.sql.ResultSet;  public class ChiTietSanPhamDAO extends MainDAO<ChiTietSanPham, String> {          final String INSERT\_SQL = "INSERT INTO ChiTietSP(MaCT, MaSP, Size, MauSac, ChatLieu, Gia) VALUES (?,?,?,?,?,?)";          final String UPDATE\_SQL = "UPDATE ChiTietSP SET MaSP =?, Size=?, MauSac =?, ChatLieu =?, Gia =? WHERE MaCT=?";          final String DELETE\_SQL = "DELETE FROM ChiTietSP WHERE MaCT = ?";          final String SELECT\_ALL\_SQL = "SELECT \* FROM ChiTietSP";          final String SELECT\_By\_Id\_SQL = "SELECT \* FROM ChiTietSP WHERE MaCT = ?";          final String SELECT\_By\_Id\_SQLL = "SELECT \* FROM ChiTietSP WHERE MaSP = ?";      @Override      public void insert(ChiTietSanPham entity) {          JdbcHelper.executeUpdate(INSERT\_SQL, entity.getMaCT(), entity.getMaSP(), entity.getSize(), entity.getMauSac(), entity.getChatLieu(), entity.getGia());      }      @Override      public void update(ChiTietSanPham entity) {          JdbcHelper.executeUpdate(UPDATE\_SQL, entity.getMaSP(), entity.getSize(), entity.getMauSac(), entity.getChatLieu(), entity.getGia(), entity.getMaCT());      }      @Override      public void delete(String id) {          JdbcHelper.executeUpdate(DELETE\_SQL, id);      }      @Override      public List<ChiTietSanPham> selectAll() {          return selectBySql(SELECT\_ALL\_SQL);      }      @Override      public ChiTietSanPham selectById(String id) {          List<ChiTietSanPham> list = selectBySql(SELECT\_By\_Id\_SQL, id);          if(list.isEmpty()){              return null;          }          return list.get(0);      }      public ChiTietSanPham selectByIdd(String id) {          List<ChiTietSanPham> list = selectBySql(SELECT\_By\_Id\_SQLL, id);          if(list.isEmpty()){              return null;          }          return list.get(0);      }        public List<ChiTietSanPham> selectByKeyword(String key) {          String sql = "SELECT \* FROM ChiTietSP WHERE MaCT like ?";          return this.selectBySql(sql, "%"+key+"%");      }        @Override      public List<ChiTietSanPham> selectBySql(String sql, Object... args) {          List<ChiTietSanPham> list = new ArrayList<ChiTietSanPham>();          try {              ResultSet rs = JdbcHelper.executeQuery(sql, args);              while (rs.next()) {                 ChiTietSanPham entity = new ChiTietSanPham();                 entity.setMaCT(rs.getString("MaCT"));                 entity.setMaSP(rs.getString("MaSP"));                 entity.setSize(rs.getFloat("Size"));                 entity.setMauSac(rs.getString("MauSac"));                 entity.setChatLieu(rs.getString("ChatLieu"));                 entity.setGia(rs.getFloat("Gia"));                 list.add(entity);              }          } catch (Exception e) {              throw new RuntimeException(e);          }          return list;      }  } |

#### Lớp DAO làm việc với Bảng Hóa đơn

|  |
| --- |
| package com.nhom4.dao;  import com.nhom4.entity.HoaDon;  import com.nhom4.utils.JdbcHelper;  import java.sql.ResultSet;  import java.sql.SQLException;  import java.util.ArrayList;  import java.util.List;  public class HoaDonDAO extends MainDAO<HoaDon, String> {          final String INSERT\_SQL = "INSERT INTO HoaDon (MaHD, MaNV, MaKM, NgayBan, MaKH,ThanhTien,TrangThai)VALUES (?,?,?,?,?,?,?)";          final String UPDATE\_SQL = "UPDATE HoaDon SET MaNV =?, MaKM=?, NgayBan =?, MaKH =?, ThanhTien =?, TrangThai =? WHERE MaHD=?";          final String DELETE\_SQL = "DELETE FROM HoaDon WHERE MaHD = ?";          final String SELECT\_ALL\_SQL = "SELECT \* FROM HoaDon";          final String SELECT\_By\_Id\_SQL = "SELECT \* FROM HoaDon WHERE MaHD = ?";      @Override      public void insert(HoaDon entity) {          JdbcHelper.executeUpdate(INSERT\_SQL, entity.getMaHD(), entity.getMaNV(), entity.getMaKM(), entity.getNgayBan(), entity.getMaKH(), entity.getThanhTien(), entity.isTrangThai());      }      @Override      public void update(HoaDon entity) {          JdbcHelper.executeUpdate(UPDATE\_SQL, entity.getMaNV(), entity.getMaKM(), entity.getNgayBan(), entity.getMaKH(), entity.getThanhTien(), entity.isTrangThai(), entity.getMaHD());      }      @Override      public void delete(String id) {          JdbcHelper.executeUpdate(DELETE\_SQL, id);      }      @Override      public List<HoaDon> selectAll() {          return selectBySql(SELECT\_ALL\_SQL);      }      @Override      public HoaDon selectById(String id) {          List<HoaDon> list = selectBySql(SELECT\_By\_Id\_SQL, id);          if(list.isEmpty()){              return null;          }          return list.get(0);      }      @Override      public List<HoaDon> selectBySql(String sql, Object... args) {          List<HoaDon> list = new ArrayList<HoaDon>();          try {              ResultSet rs = JdbcHelper.executeQuery(sql, args);              while (rs.next()) {                 HoaDon entity = new HoaDon();                 entity.setMaHD(rs.getString("MaHD"));                 entity.setMaNV(rs.getString("MaNV"));                 entity.setMaKM(rs.getString("MaKM"));                 entity.setNgayBan(rs.getDate("NgayBan"));                 entity.setMaKH(rs.getString("MaKH"));                 entity.setThanhTien(rs.getFloat("ThanhTien"));                 entity.setTrangThai(rs.getBoolean("TrangThai"));                 list.add(entity);              }          } catch (Exception e) {              throw new RuntimeException(e);          }          return list;      }      public List<String> selectMaNV(){          String sql = "Select maNV from NhanVien";          List<String> list = new ArrayList<>();          try {              ResultSet rs = JdbcHelper.executeQuery(sql);              while (rs.next()) {                  list.add(rs.getString(1));              }              rs.getStatement().getConnection().close();              return list;          } catch (SQLException e) {              throw new RuntimeException(e);          }      }      public List<String> selectMaKH(){          String sql = "Select maKH from KhachHang";          List<String> list = new ArrayList<>();          try {              ResultSet rs = JdbcHelper.executeQuery(sql);              while (rs.next()) {                  list.add(rs.getString(1));              }              rs.getStatement().getConnection().close();              return list;          } catch (SQLException e) {              throw new RuntimeException(e);          }      }      public List<String> selectMaKM(){          String sql = "Select maKM from KhuyenMai where GETDATE() <= NgayKT;";          List<String> list = new ArrayList<>();          try {              ResultSet rs = JdbcHelper.executeQuery(sql);              while (rs.next()) {                  list.add(rs.getString(1));              }              rs.getStatement().getConnection().close();              return list;          } catch (SQLException e) {              throw new RuntimeException(e);          }      }      public void updateNgayBan(String id){          String UPDATE\_Date = "UPDATE HOADON SET NGAYBAN=GETDATE() WHERE MaHD=?";          JdbcHelper.executeUpdate(UPDATE\_Date, id);      }      public void updateThanhTien(String id){          String Update\_ThanhTien ="{CALL sp\_TongTien(?)}";          JdbcHelper.executeUpdate(Update\_ThanhTien, id);      }  } |

#### Lớp DAO làm việc với Bảng Chi tiết hóa đơn

|  |
| --- |
| package com.nhom4.dao;  import com.nhom4.entity.HoaDonChiTiet;  import java.sql.ResultSet;  import java.util.List;  import com.nhom4.utils.JdbcHelper;  import java.sql.SQLException;  import java.util.ArrayList;  public class HoaDonChiTietDAO extends MainDAO<HoaDonChiTiet, String> {      final String SET\_IDENTITY = "SET IDENTITY\_INSERT HoaDonCT ON";      final String INSERT\_SQL = "INSERT INTO HoaDonCT (MaHD, MaCT, TenSP, SoLuong,Gia,TongTien)VALUES(?,?,?,?,?,?)";      final String UPDATE\_SQL = "UPDATE HoaDonCT SET MaHD=?, MaCT =?, TenSP=?, SoLuong=?, Gia=?, TongTien=? WHERE MaHDCT=?";      final String DELETE\_SQL = "DELETE FROM HoaDonCT WHERE MaHDCT = ?";      final String SELECT\_ALL\_SQL = "SELECT \* FROM HoaDonCT ";      final String SELECT\_By\_Id\_SQL = "SELECT \* FROM HoaDonCT WHERE MaHDCT = ?";      @Override      public void insert(HoaDonChiTiet entity) {          JdbcHelper.executeUpdate(INSERT\_SQL, entity.getMaHD(), entity.getMaCT(), entity.getTenSP(), entity.getSoLuong(), entity.getGia(), entity.getTongTien());      }      @Override      public void update(HoaDonChiTiet entity) {          JdbcHelper.executeUpdate(UPDATE\_SQL, entity.getMaHD(), entity.getMaCT(), entity.getTenSP(), entity.getSoLuong(), entity.getGia(), entity.getTongTien(), entity.getMaHDCT());      }      @Override      public void delete(String id) {          JdbcHelper.executeUpdate(DELETE\_SQL, id);      }      @Override      public List<HoaDonChiTiet> selectAll() {          return selectBySql(SELECT\_ALL\_SQL);      }      @Override      public HoaDonChiTiet selectById(String id) {          List<HoaDonChiTiet> list = selectBySql(SELECT\_By\_Id\_SQL, id);          if (list.isEmpty()) {              return null;          }          return list.get(0);      }      public void deletebByMaHD(String id){          String DELETE= "DELETE FROM HoaDonCT WHERE MaHD = ?";          JdbcHelper.executeUpdate(DELETE, id);      }      @Override      public List<HoaDonChiTiet> selectBySql(String sql, Object... args) {          List<HoaDonChiTiet> list = new ArrayList<HoaDonChiTiet>();          try {              ResultSet rs = JdbcHelper.executeQuery(sql, args);              while (rs.next()) {                  HoaDonChiTiet entity = new HoaDonChiTiet();                  entity.setMaHDCT(rs.getInt("MaHDCT"));                  entity.setMaHD(rs.getString("MaHD"));                  entity.setMaCT(rs.getString("MaCT"));                  entity.setTenSP(rs.getString("TenSP"));                  entity.setSoLuong(rs.getInt("SoLuong"));                  entity.setGia(rs.getFloat("Gia"));                  entity.setTongTien(rs.getFloat("TongTien"));                  list.add(entity);              }          } catch (Exception e) {              throw new RuntimeException(e);          }          return list;      }       public List<HoaDonChiTiet> selectByMaHD(String key) {          String sql = "SELECT \* FROM HoaDonCT WHERE MaHD like ?";          return this.selectBySql(sql, "%"+key+"%");      }      public List<String> selectMaCT(){          String sql = "Select maCT from ChiTietSP";          List<String> list = new ArrayList<>();          try {              ResultSet rs = JdbcHelper.executeQuery(sql);              while (rs.next()) {                  list.add(rs.getString(1));              }              rs.getStatement().getConnection().close();              return list;          } catch (SQLException e) {              throw new RuntimeException(e);          }      }  } |

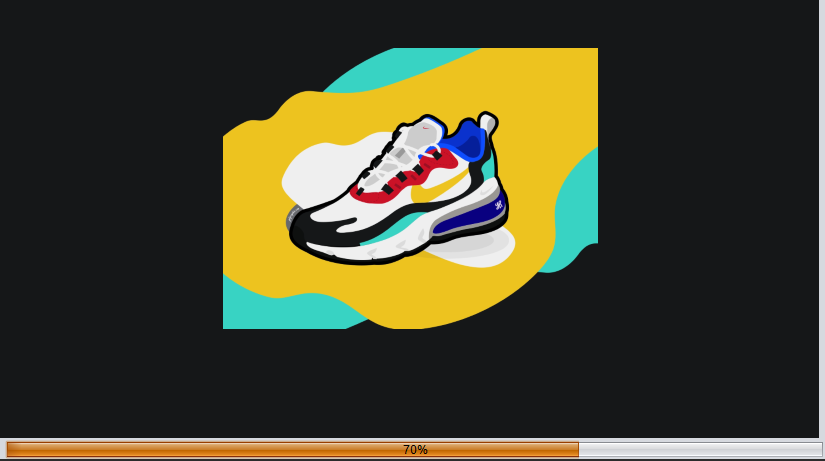
#### Lớp DAO làm việc với Bảng Khuyến mãi

|  |
| --- |
| package com.nhom4.dao;  import com.nhom4.entity.KhuyenMai;  import com.nhom4.utils.JdbcHelper;  import java.sql.ResultSet;  import java.util.ArrayList;  import java.util.Date;  import java.util.List;  public class KhuyenMaiDAO extends MainDAO<KhuyenMai, String>{      final String INSERT\_SQL = "INSERT INTO KhuyenMai (MaKM, TenKM, GiamGia, NgayBD, NgayKT, GhiChu) VALUES (?,?,?,?,?,?)";      final String UPDATE\_SQL = "UPDATE KhuyenMai SET TenKM =?, GiamGia=?, NgayBD =?, NgayKT =?, GhiChu =? WHERE MaKM=?";      final String DELETE\_SQL = "DELETE FROM KhuyenMai WHERE MaKM = ?";      final String SELECT\_ALL\_SQL = "SELECT \* FROM KhuyenMai";      final String SELECT\_By\_Id\_SQL = "SELECT \* FROM KhuyenMai WHERE MaKM = ?";        @Override      public void insert(KhuyenMai entity) {          JdbcHelper.executeUpdate(INSERT\_SQL, entity.getMaKM(), entity.getTenKM(),entity.getGiamGia(),new java.sql.Date(entity.getNgayBD().getTime()), new java.sql.Date(entity.getNgayKT().getTime()), entity.getGhiChu());      }      @Override      public void update(KhuyenMai entity) {          JdbcHelper.executeUpdate(UPDATE\_SQL, entity.getTenKM(),entity.getGiamGia(), new java.sql.Date(entity.getNgayBD().getTime()), new java.sql.Date(entity.getNgayKT().getTime()), entity.getGhiChu(), entity.getMaKM());      }      @Override      public void delete(String id) {          JdbcHelper.executeUpdate(DELETE\_SQL, id);      }      @Override      public List<KhuyenMai> selectAll() {          return selectBySql(SELECT\_ALL\_SQL);      }      @Override      public KhuyenMai selectById(String id) {          List<KhuyenMai> list = selectBySql(SELECT\_By\_Id\_SQL, id);          if(list.isEmpty()){              return null;          }          return list.get(0);      }        public List<KhuyenMai> selectByKeyword(String key) {          String sql = "SELECT \* FROM KhuyenMai WHERE TenKM like ?";          return this.selectBySql(sql, "%"+key+"%");      }      @Override      public List<KhuyenMai> selectBySql(String sql, Object... args) {          List<KhuyenMai> list = new ArrayList<KhuyenMai>();          try {              ResultSet rs = JdbcHelper.executeQuery(sql, args);              while (rs.next()) {                 KhuyenMai entity = new KhuyenMai();                 entity.setMaKM(rs.getString("MaKM"));                 entity.setTenKM(rs.getString("TenKM"));                 entity.setGiamGia(rs.getFloat("GiamGia"));                 java.sql.Date ngayBD = rs.getDate("NgayBD");                 java.sql.Date ngayKT = rs.getDate("NgayKT");                 entity.setNgayBD(new Date(ngayBD.getTime()));                 entity.setNgayKT(new Date(ngayKT.getTime()));                 entity.setGhiChu(rs.getString("GhiChu"));                 list.add(entity);              }          } catch (Exception e) {              throw new RuntimeException(e);          }          return list;      }  } |

#### Lớp DAO làm việc với bảng thống kê

## Viết mã cho ứng dụng

### Xử lý Form Chào



Mô tả hoạt động

|  |  |
| --- | --- |
| Sự kiện | Công việc phải làm |
| Constructor | init(); |

Hướng dẫn cài đặt mã nguồn

|  |
| --- |
| private void init() {          this.setLocationRelativeTo(null);          new Timer(3, new ActionListener() {              @Override              public void actionPerformed(ActionEvent e) {                  int value = prbChao.getValue();                  if(value < prbChao.getMaximum()){                      prbChao.setValue(value + 100);                  } else {                      ChaoJDialog.this.dispose();                  }              }          }).start();      } |

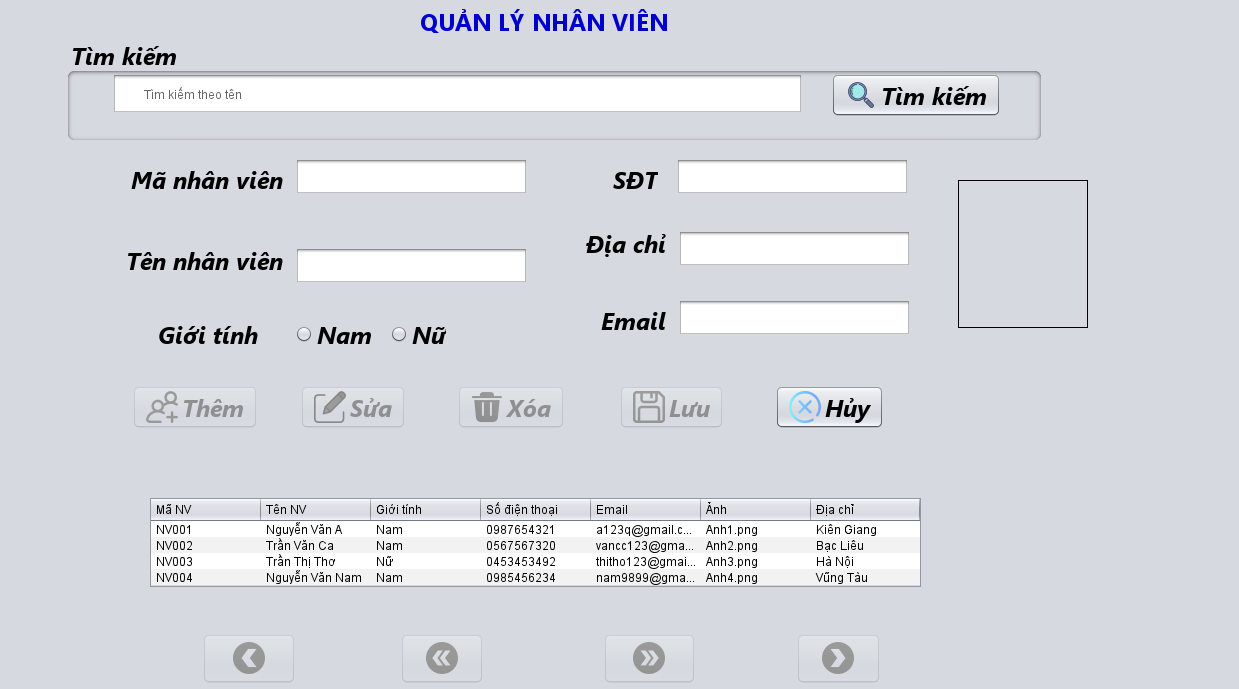
### Xử lý Form Đăng nhập

|  |
| --- |
| TaiKhoanDAO tkDAO = new TaiKhoanDAO();   public DangNhap(java.awt.Frame parent, boolean modal) {          super(parent, modal);          initComponents();          init();        }  class jPanelGradient extends JPanel {          protected void paintComponent(Graphics g) {              Graphics2D g2d = (Graphics2D) g;              int width = getWidth();              int height = getHeight();              Color color1 = new Color(234, 74, 107);              Color color2 = new Color(114, 113, 193);              GradientPaint gp = new GradientPaint(0, 0, color1, 180, height, color2);              g2d.setPaint(gp);              g2d.fillRect(0, 0, width, height);          }      }  void dangNhap() {          String manv = txtMaNV.getText();          String matkhau = new String(txtMatKhau.getPassword());          TaiKhoan nv = tkDAO.selectById(manv);          if (nv == null) {              MsgBox.alert(this, "Tên đăng nhập không chính xác");          } else {              if (!nv.getPass().equals(matkhau)) {                  MsgBox.alert(this, "Mật khẩu không chính xác");              } else {                  //new loading().setVisible(true);                  Auth.user = nv;                  MsgBox.alert(this, "Đăng nhập thành công!");                  this.dispose();              }          }      }      void ketThuc() {          if (MsgBox.confirm(this, "Bạn có chắc muốn kết thúc ứng dụng?")) {              System.exit(0);          }      } |

### Xử lý Form Đổi mật khẩu

|  |
| --- |
| TaiKhoanDAO tkDAO = new TaiKhoanDAO();  public DoiMatKhau(java.awt.Frame parent, boolean modal) {          super(parent, modal);          initComponents();          init();      }  class jPanelGradient extends JPanel {          protected void paintComponent(Graphics g) {              Graphics2D g2d = (Graphics2D) g;              int width = getWidth();              int height = getHeight();              Color color1 = new Color(249, 102, 102);              Color color2 = new Color(95, 157, 247);              GradientPaint gp = new GradientPaint(0, 0, color1, 180, height, color2);              g2d.setPaint(gp);              g2d.fillRect(0, 0, width, height);          }      }   private void init() {          this.setLocationRelativeTo(null);      }   private void doiMatKhau(){          String manv = txtMaNV.getText();          String matKhau = new String(txtMatKhau.getPassword());          String matKhau1 = new String(txtMatKhau1.getPassword());          String matKhau2 = new String(txtMatKhau2.getPassword());            if(!manv.equalsIgnoreCase(Auth.user.getUsername())){              MsgBox.alert(this, "Tên đăng nhập chưa chính xác");          } else if(!matKhau.equalsIgnoreCase(Auth.user.getPass())){              MsgBox.alert(this, "Mật khẩu chưa chính xác");          } else if(!matKhau2.equalsIgnoreCase(matKhau1)){              MsgBox.alert(this, "Xác nhận mật khẩu sai");          } else {              Auth.user.setPass(matKhau1);              tkDAO.update(Auth.user);              MsgBox.alert(this, "Đổi mật khẩu thành công");          }      }        private void huyBo(){          this.dispose();      } |

### Xử lý Form Nhân viên



Mô tả hoạt động

|  |  |
| --- | --- |
| Sự Kiện | Công việc phải làm |
| Constructor | init(); |
| tblnhanVienMouseClicked | if (evt.getClickCount() == 1) {  this.row = tblnhanVien.getSelectedRow();  this.edit();  } |
| btnThem | them = 1;  btnSua.setEnabled(false);  btnXoa.setEnabled(false);  btnLuu.setEnabled(true);  txtmaNhanVien.setText("");  txttenNhanVien.setText("");  rdoGioiTinhNam.isSelected();  txtsoDienSo.setText("");  txtdiaChi.setText("");  txtEmail.setText("");  ImageIcon icon = new ImageIcon(".//src//com//nhom4//icon//user.png");  lblHinh.setIcon(icon);  txtON(); |
| btnSua | them = 2;  btnThem.setEnabled(false);  btnXoa.setEnabled(false);  btnLuu.setEnabled(true);  txtON();  txtmaNhanVien.setEditable(false); |
| btnXoa | delete(); |
| btnLuu | if (check() == true) {  if (them == 1) {  insert();  return;  }  if (them == 2) {  update();  return;  }  } |
| btnHuy | updateStatus();  clearForm();  btnLuu.setEnabled(false); |
| btnFirst | first(); |
| btnPre | prev(); |
| btnNext | next(); |
| btnLast | last(); |
| txtTimKiem | if (txttimKiem.getText().equals("")) {  lblTimKiem.setVisible(true);  } else  lblTimKiem.setVisible(false); |
| btnTimKiem | this.TimKiem(); |
| lblHinh | this.selectImage(); |

|  |
| --- |
| NhanVienDAO dao = new NhanVienDAO();      int row = -1;      int them = 0;      JFileChooser fileChooser = new JFileChooser();      String path = ".//src//com//nhom4//icon//";  private void init() {           this.row = -1;          fillTable();          this.updateStatus();          btnLuu.setEnabled(false);      }  public void fillTable() {          DefaultTableModel model = (DefaultTableModel) tblnhanVien.getModel();          model.setRowCount(0);          try {              List<NhanVien> list = dao.selectAll(); // Đọc dữ liệu từ CSDL              for (NhanVien nv : list) {                  Object[] data = {                      nv.getMaNV(),                      nv.getTenNV(),                      nv.getGioiTinh()?"Nam":"Nữ",                      nv.getSDT(),                      nv.getEmail(),                      nv.getHinh(),                      nv.getDiaChi(),                  };                  model.addRow(data);              }          } catch (Exception e) {              MsgBox.alert(this, "Lỗi truy vấn dữ liệu!");          }      }  public void insert() {                NhanVien nv = getForm();              try {                  dao.insert(nv);                  this.fillTable();                  this.clearForm();                  MsgBox.alert(this, "Thêm mới thành công");                  updateStatus();                  them = 0;                  btnLuu.setEnabled(false);              } catch (Exception e) {                  MsgBox.alert(this, "Thêm mới thất bại");                  updateStatus();                  them = 0;                    btnLuu.setEnabled(false);              }      }   public void update() {          NhanVien nv = getForm();              try {                  dao.update(nv);                  this.fillTable();                  MsgBox.alert(this, "Cập nhật thành công");                  updateStatus();                  them = 0;                  btnLuu.setEnabled(false);              } catch (Exception e) {                  MsgBox.alert(this, "Cập nhật thất bại");                  updateStatus();                   them = 0;                  btnLuu.setEnabled(false);              }      }        public void delete() {              String nv = txtmaNhanVien.getText();              if (MsgBox.confirm(this, "Bạn thực sự muốn xóa nhân viên này")) {                  try {                      dao.delete(nv);                      this.fillTable();                      this.clearForm();                      MsgBox.alert(this, "Xóa thành công");                  } catch (Exception e) {                      MsgBox.alert(this, "Xóa thất bại");                  }              }      }        void selectImage() {          if (fileChooser.showOpenDialog(this) == JFileChooser.APPROVE\_OPTION) {              File file = fileChooser.getSelectedFile();              if (XImage.save(file)) {                  // Hiển thị hình lên form                  lblHinh.setIcon(XImage.read(file.getName()));                  lblHinh.setToolTipText(file.getName());              }          }      }      public void clearForm() {          NhanVien nv = new NhanVien();          this.setForm(nv);          this.row = -1;          this.updateStatus();      }        public void setForm(NhanVien nv) {          txtmaNhanVien.setText(nv.getMaNV());          txttenNhanVien.setText(nv.getTenNV());          rdoGioiTinhNam.setSelected(nv.getGioiTinh());           rdoGioiTinhNu.setSelected(!nv.getGioiTinh());          txtsoDienSo.setText(nv.getSDT());          txtEmail.setText(nv.getEmail());          lblHinh.setToolTipText(nv.getHinh());          if (nv.getHinh()!= null) {              lblHinh.setIcon(XImage.read(nv.getHinh()));          }          txtdiaChi.setText(nv.getDiaChi());      }        NhanVien getForm() {          NhanVien nv = new NhanVien();          nv.setMaNV(txtmaNhanVien.getText());          nv.setTenNV(txttenNhanVien.getText());          nv.setGioiTinh(rdoGioiTinhNam.isSelected());          nv.setSDT(txtsoDienSo.getText());          nv.setEmail(txtEmail.getText());          nv.setHinh(lblHinh.getToolTipText());          nv.setDiaChi(txtdiaChi.getText());            return nv;      }        public void updateStatus() {          boolean edit = (this.row >= 0);          boolean first = (this.row == 0);          boolean last = (this.row == tblnhanVien.getRowCount() - 1);  //        Trạng thái form          btnThem.setEnabled(edit);          btnSua.setEnabled(edit);          btnXoa.setEnabled(edit);  // Trạng thái điều hướng          btnFirst.setEnabled(edit && !first);          btnPrev.setEnabled(edit && !first);          btnNext.setEnabled(edit && !last);          btnLast.setEnabled(edit && !last);      }        public void edit() {          String manv = (String) tblnhanVien.getValueAt(this.row, 0);          NhanVien nv = dao.selectById(manv);          this.setForm(nv);          this.updateStatus();      }        public void first() {          this.row = 0;          tblnhanVien.setRowSelectionInterval(row, row);          this.edit();      }        public void prev() {          if (this.row > 0) {              this.row--;               tblnhanVien.setRowSelectionInterval(row, row);              this.edit();          }      }        public void next() {          if (this.row < tblnhanVien.getRowCount() - 1) {              this.row++;               tblnhanVien.setRowSelectionInterval(row, row);              this.edit();          }      }        public void last() {          this.row = tblnhanVien.getRowCount() - 1;           tblnhanVien.setRowSelectionInterval(row, row);          this.edit();      }        public void txtOFF(){          txtmaNhanVien.setEditable(false);          txttenNhanVien.setEditable(false);          rdoGioiTinhNam.setEnabled(false);           txtsoDienSo.setEditable(false);           txtdiaChi.setEditable(false);           txtEmail.setEditable(false);  //         lblHinh.setEditable(false);          }      public void txtON(){         txtmaNhanVien.setEditable(true);          txttenNhanVien.setEditable(true);          rdoGioiTinhNam.setEnabled(true);           txtsoDienSo.setEditable(true);           txtdiaChi.setEditable(true);           txtEmail.setEditable(true);  //         lblHinh.setEditable(true);      }        public boolean check(){   if (txtmaNhanVien.getText().equals("") || txtmaNhanVien.getText().length() < 5 || txtmaNhanVien.getText().length() > 6) {              MsgBox.alert(this, "Vui lòng nhập mã nhân viên từ 5---->6 kí tự");              txtmaNhanVien.requestFocus();              return false;          } else if (txttenNhanVien.getText().length() == 0) {              MsgBox.alert(this, "Tên nhân viên không được bỏ trống!!!");              txttenNhanVien.requestFocus();              return false;          } else if (txtsoDienSo.getText().equals("") || txtsoDienSo.getText().length() < 9 || txtsoDienSo.getText().length() > 10) {              MsgBox.alert(this, "Vui lòng nhập số điện thoại từ 9---->10 kí tự");              txtsoDienSo.requestFocus();              return false;          } else if (txtdiaChi.getText().length() == 0) {              MsgBox.alert(this, "Địa chỉ không được bỏ trống!!!");              txtdiaChi.requestFocus();              return false;          }else if (txtEmail.getText().length() == 0) {              MsgBox.alert(this, "Email không được bỏ trống!!!");              txtEmail.requestFocus();              return false;          }else if (lblHinh.getIcon() == null) {              MsgBox.alert(this, "Bạn chưa chọn hình! click vào khu vực hình để chọn");              return false;          }          return true;      }       public void TimKiem() {   DefaultTableModel model = (DefaultTableModel) tblnhanVien.getModel();          model.setRowCount(0);          try {              String nvv = txttimKiem.getText();              List<NhanVien> list = dao.selectByKeyword(nvv);              for (NhanVien nv : list) {                  Object[] data = {                      nv.getMaNV(),                      nv.getTenNV(),                      nv.getGioiTinh(),                      nv.getSDT(),                      nv.getDiaChi(),                      nv.getEmail(),                      nv.getHinh(),                  };                  model.addRow(data);              }          } catch (Exception e) {              MsgBox.alert(this, e.getMessage());          }          this.clearForm();          this.row = -1;          updateStatus();      } |

### Xử lý Form Tài khoản



Mô tả hoạt động

|  |  |
| --- | --- |
| Sự Kiện | Công việc phải làm |
| Constructor | this.setColumns();  this.init();  txtOFF(); |
| tblBangMouseClicked | if (evt.getClickCount() == 1) {  this.row = tblBang.getSelectedRow();  this.edit();  setForm(row);  } |
| btnThem | them = 1;  btnSua.setEnabled(false);  btnXoa.setEnabled(false);  btnLuu.setEnabled(true);  txtMaNV.setText("");  txtTenDN.setText("");  txtMatKhau.setText("");  show();  txtON(); |
| btnSua | them = 2;  btnThem.setEnabled(false);  btnXoa.setEnabled(false);  btnLuu.setEnabled(true);  // hide();  txtON();  txtMaNV.setEditable(false); |
| btnXoa | this.delete(); |
| btnLuu | if(them == 1 ){  insert();  return;  }  if(them == 2){  update();  return;  } |
| btnHuy | TaiKhoan tk = new TaiKhoan();  this.setForm1(tk);  this.row = -1;  this.updateStatus(); |
| btnFirst | this.first(); |
| btnPre | this.prev(); |
| btnNext | this.next(); |
| btnLast | this.last(); |
| txtTimKiem | if (txtTimKiem.getText().equals("")) {  lblTimKiem.setVisible(true);  } else  lblTimKiem.setVisible(false); |
| btnTimKiem | this.TimKiem(); |

Hướng dẫn cài đặt mã nguồn

|  |
| --- |
| DefaultTableModel model;  int checklap = 0;  ArrayList<TaiKhoan> listTK = new ArrayList<>();  TaiKhoanDAO tkDAO = new TaiKhoanDAO();  int row = -1;  int them = 0;  public void init() {  this.row = -1;  this.fillTable();  this.updateStatus();  btnLuu.setEnabled(false);  }  public void setColumns() {  String[] row = new String[]{"Mã nhân viên", "Tên đăng nhập", "Mật khẩu", "Vai trò"};  model = new DefaultTableModel();  model.setColumnIdentifiers(row);  tblBang.setModel(model);  }  public void fillTable() {  DefaultTableModel model = (DefaultTableModel) tblBang.getModel();  model.setRowCount(0);  try {  List<TaiKhoan> list = tkDAO.selectAll(); // Đọc dữ liệu từ CSDL  for (TaiKhoan tk : list) {  Object[] data = {  tk.getMaNV(),  tk.getUsername(),  tk.getPass(),  tk.getRole() ? "Quản lý" : "Nhân viên",};  model.addRow(data);  listTK.add(tk);  }  } catch (Exception e) {  MsgBox.alert(this, "Lỗi truy vấn dữ liệu!");  }  }  public void insert() {  if (check()) {  TaiKhoan tk = getForm();  try {  tkDAO.insert(tk);  this.fillTable();  // this.clearForm();  MsgBox.alert(this, "Thêm mới thành công");  updateStatus();  them = 0;  btnLuu.setEnabled(false);  } catch (Exception e) {  MsgBox.alert(this, "Thêm mới thất bại");  updateStatus();  them = 0;  btnLuu.setEnabled(false);  }  }  }  public void update() {  if (check()) {  TaiKhoan tk = getForm();  try {  tkDAO.update(tk);  this.fillTable();  MsgBox.alert(this, "Cập nhật thành công");  updateStatus();  them = 0;  btnLuu.setEnabled(false);  } catch (Exception e) {  MsgBox.alert(this, "Cập nhật thất bại");  updateStatus();  them = 0;  btnLuu.setEnabled(false);  }  }  }  public void delete() {  if (!Auth.isManager()) {  MsgBox.alert(this, "Bạn không có quyền xóa nhân viên");  } else {  String manv = txtMaNV.getText();  if (manv.equals(Auth.user.getMaNV())) {  MsgBox.alert(this, "Bạn không được xóa chính bạn");  } else if (MsgBox.confirm(this, "Bạn thực sự muốn xóa nhân viên này")) {  try {  tkDAO.delete(manv);  this.clearForm();  this.fillTable();  this.clearForm();  MsgBox.alert(this, "Xóa thành công");  } catch (Exception e) {  MsgBox.alert(this, "Xóa thất bại");  }  }  }  }  public void first() {  this.row = 0;  tblBang.setRowSelectionInterval(row, row);  this.edit();  this.setForm(row);  }  public void prev() {  if (this.row > 0) {  this.row--;  tblBang.setRowSelectionInterval(row, row);  this.edit();  this.setForm(row);  }  }  public void next() {  if (this.row < tblBang.getRowCount() - 1) {  this.row++;  tblBang.setRowSelectionInterval(row, row);  this.edit();  this.setForm(row);  }  }  public void last() {  this.row = tblBang.getRowCount() - 1;  tblBang.setRowSelectionInterval(row, row);  this.edit();  this.setForm(row);  }  public boolean check() {  String patternUser = "^[a-zA-Z0-9\_-]{6,15}$";  String patternPass = "^[a-zA-Z0-9]{6,15}$";  for (int i = 0; i < listTK.size(); i++) {  if (listTK.get(i).getMaNV().equalsIgnoreCase(txtMaNV.getText())) {  checklap = 1;  }  }  if (txtMaNV.getText().equals("")) {  MsgBox.alert(this, "Không được để trống mã nhân viên");  txtMaNV.requestFocus();  return false;  } else if (them == 1 && checklap == 1) {  MsgBox.alert(this, "Mã nhân viên đã tồn tại. Vui lòng nhập mã mới");  checklap = 0;  return false;  } else if (txtTenDN.getText().equals("")) {  MsgBox.alert(this, "Không được để trống tên đăng nhập");  txtTenDN.requestFocus();  return false;  } else if (!txtTenDN.getText().matches(patternUser) || txtTenDN.getText().length() < 6 || txtTenDN.getText().length() > 15) {  MsgBox.alert(this, "Tên đăng nhập không chứa kí tự đặc biệt (@, #, %,..) và phải từ 6 đến 15 kí tự");  txtTenDN.requestFocus();  return false;  } else if (txtMatKhau.getText().equals("")) {  MsgBox.alert(this, "Không được để trống mật khẩu");  txtMatKhau.requestFocus();  return false;  } else if (!txtMatKhau.getText().matches(patternPass) || txtMatKhau.getText().length() < 6 || txtMatKhau.getText().length() > 15) {  MsgBox.alert(this, "Mật khẩu không chứa kí tự đặc biệt (@, #, %,..) và phải từ 6 đến 15 kí tự");  txtMatKhau.requestFocus();  return false;  } else if (!rdoNhanVien.isSelected() && !rdoQuanLy.isSelected()) {  MsgBox.alert(this, "Bạn chưa chọn chức vụ");  return false;  } else if (them == 1 && Auth.isManager() && rdoQuanLy.isSelected()) {  MsgBox.alert(this, "Không được chọn vai trò là quản lý");  return false;  }  return true;  }  public void setForm(int index) {  txtMaNV.setText(tblBang.getValueAt(index, 0).toString());  txtTenDN.setText(tblBang.getValueAt(index, 1).toString());  txtMatKhau.setText(tblBang.getValueAt(index, 2).toString());  if (tblBang.getValueAt(index, 3).toString().equals("Nhân viên")) {  rdoNhanVien.setSelected(true);  } else {  rdoQuanLy.setSelected(true);  }  }  public void setForm1(TaiKhoan tk) {  txtMaNV.setText(tk.getMaNV());  txtTenDN.setText(tk.getUsername());  txtMatKhau.setText(tk.getPass());  rdoQuanLy.setSelected(tk.getRole());  rdoNhanVien.setSelected(!tk.getRole());  }  TaiKhoan getForm() {  TaiKhoan tk = new TaiKhoan();  tk.setMaNV(txtMaNV.getText());  tk.setUsername(txtTenDN.getText());  tk.setPass(String.valueOf(txtMatKhau.getText()));  tk.setRole(rdoQuanLy.isSelected());  return tk;  }  public void clearForm() {  TaiKhoan tk = new TaiKhoan();  this.setForm(0);  this.row = -1;  this.updateStatus();  }  public void updateStatus() {  boolean edit = (this.row >= 0);  boolean first = (this.row == 0);  boolean last = (this.row == tblBang.getRowCount() - 1);  // Trạng thái form  btnThem.setEnabled(edit);  btnSua.setEnabled(edit);  btnXoa.setEnabled(edit);  // Trạng thái điều hướng  btnFirst.setEnabled(edit && !first);  btnPre.setEnabled(edit && !first);  btnNext.setEnabled(edit && !last);  btnLast.setEnabled(edit && !last);  txtOFF();  }  public void edit() {  String manv = (String) tblBang.getValueAt(row, 0);  TaiKhoan tk = tkDAO.selectById(manv);  int i = tblBang.getSelectedRow();  this.setForm(i);  this.updateStatus();  }  public void txtOFF() {  txtMaNV.setEditable(false);  txtTenDN.setEditable(false);  txtMatKhau.setEditable(false);  // rdoQuanLy.setSelected(true);  rdoQuanLy.setEnabled(false);  rdoNhanVien.setEnabled(false);  }  public void txtON() {  txtMaNV.setEditable(true);  txtTenDN.setEditable(true);  txtMatKhau.setEditable(true);  }  public void hide() {  }  public void show() {  rdoQuanLy.setEnabled(true);  rdoNhanVien.setEnabled(true);  } |

### Xử lý Form Tài khoản cá nhân



Mô tả hoạt động

|  |  |
| --- | --- |
| Sự kiện | Công việc phải làm |
| Constructor | this.init();  txtMaNV.setEditable(false);  fill();  btnLuu.setEnabled(false);  txtOFF(); |
| btnSua | them = 2;  btnLuu.setEnabled(true);  txtON(); |
| btnLuu | if(them == 2){  update();  txtOFF();  return;  } |
| btnDoiMK | DoiMatKhau dialog = new DoiMatKhau(new javax.swing.JFrame(), true);  dialog.setVisible(true); |
| lblHinhMouseClicked | this.selectImage(); |

Hướng dẫn cài đặt mã nguồn

|  |
| --- |
| NhanVienDAO nvDao = new NhanVienDAO();  ArrayList<NhanVien> list = new ArrayList<>();  int checkLap = 0;  int them =0;  int row =-1;  JFileChooser fileChooser = new JFileChooser(".//src//com//nhom4//hinh");  public void init(){  String maNv = Auth.isMaNV();  txtMaNV.setText(maNv);  }  public void getForm(NhanVien nv){  try {  txtTenNV.setText(nv.getTenNV());  rdoNam.setSelected(nv.getGioiTinh());  rdoNu.setSelected(!nv.getGioiTinh());  txtSDT.setText(nv.getSDT());  txtDiaChi.setText(nv.getDiaChi());  txtEmail.setText(nv.getEmail());  if (nv.getHinh()!= null) {  lblHinh.setIcon(XImage.read(nv.getHinh()));  }  } catch (Exception e) {  }  }  public void fill(){  String maNV = txtMaNV.getText();  NhanVien nv = nvDao.selectById(maNV);  this.getForm(nv);  }  public void update() {  if(check()){  NhanVien nv = getForm2();  try {  nvDao.update(nv);  MsgBox.alert(this, "Cập nhật thành công");  them = 0;  btnLuu.setEnabled(false);  } catch (Exception e) {  MsgBox.alert(this, "Cập nhật thất bại");  them = 0;  btnLuu.setEnabled(false);  }  }  }  public void clearForm() {  NhanVien nv = new NhanVien();  this.setForm(nv);  this.row = -1;  }  public void selectImage() {  if (fileChooser.showOpenDialog(this) == JFileChooser.APPROVE\_OPTION) {  File file = fileChooser.getSelectedFile();  if (XImage.save(file)) {  // Hiển thị hình lên form  lblHinh.setIcon(XImage.read(file.getName()));  lblHinh.setToolTipText(file.getName());  }  }  }  public void setForm(NhanVien nv) {  txtMaNV.setText(nv.getMaNV());  txtTenNV.setText(nv.getTenNV());  rdoNam.setSelected(nv.getGioiTinh());  rdoNu.setSelected(!nv.getGioiTinh());  txtSDT.setText(nv.getSDT());  txtEmail.setText(nv.getEmail());  lblHinh.setToolTipText(nv.getHinh());  if (nv.getHinh()!= null) {  lblHinh.setIcon(XImage.read(nv.getHinh()));  }  txtDiaChi.setText(nv.getDiaChi());  }  NhanVien getForm2() {  String maNV = txtMaNV.getText();  NhanVien nhanVien = nvDao.selectById(maNV);  NhanVien nv = new NhanVien();  nv.setMaNV(txtMaNV.getText());  nv.setTenNV(txtTenNV.getText());  nv.setGioiTinh(rdoNam.isSelected());  nv.setSDT(txtSDT.getText());  nv.setEmail(txtEmail.getText());  if (lblHinh.getToolTipText()!=null) {  nv.setHinh(lblHinh.getToolTipText());  }else{  nv.setHinh(nhanVien.getHinh());  }  System.out.println(lblHinh.getToolTipText());  nv.setDiaChi(txtDiaChi.getText());  return nv;  }  public void txtOFF(){  txtTenNV.setEditable(false);  rdoNam.setEnabled(false);  txtSDT.setEditable(false);  txtDiaChi.setEditable(false);  txtEmail.setEditable(false);  rdoNu.setEnabled(false);  }  public void txtON(){  txtTenNV.setEditable(true);  rdoNam.setEnabled(true);  txtSDT.setEditable(true);  txtDiaChi.setEditable(true);  txtEmail.setEditable(true);  rdoNu.setEnabled(true);  }  private static final String EMAIL\_PATTERN  = "^[\_A-Za-z0-9-\\+]+(\\.[\_A-Za-z0-9-]+)\*@"  + "[A-Za-z0-9-]+(\\.[A-Za-z0-9]+)\*(\\.[A-Za-z]{2,})$";  public static boolean verifyEmail(String email) {  if (email == null) {  return false;  }  return email.matches(EMAIL\_PATTERN);  }  public boolean check(){  boolean checkSDT = true;  try {  Float.parseFloat(txtSDT.getText());  } catch (NumberFormatException e1) {  checkSDT = false;  }  for (int i = 0; i < list.size(); i++) {  if (list.get(i).getMaNV().equalsIgnoreCase(txtMaNV.getText())) {  checkLap = 1;  }  }  if (txtMaNV.getText().equals("") || txtMaNV.getText().length() < 5 || txtMaNV.getText().length() > 6) {  MsgBox.alert(this, "Vui lòng nhập mã nhân viên từ 5---->6 kí tự");  txtMaNV.requestFocus();  return false;  } else if (them == 1 && checkLap == 1) {  MsgBox.alert(this, "Mã nhân viên đã tồn tại. Vui lòng nhập mã mới");  checkLap = 0;  return false;  } else if (txtTenNV.getText().length() == 0) {  MsgBox.alert(this, "Tên nhân viên không được bỏ trống!!!");  txtTenNV.requestFocus();  return false;  } else if (checkSDT == false) {  MsgBox.alert(this, "Vui lòng nhập số");  txtSDT.requestFocus();  return false;  } else if (txtSDT.getText().equals("") || txtSDT.getText().length() < 9 || txtSDT.getText().length() > 10) {  MsgBox.alert(this, "Vui lòng nhập số điện thoại từ 9---->10 kí tự");  txtSDT.requestFocus();  return false;  } else if (txtDiaChi.getText().length() == 0) {  MsgBox.alert(this, "Địa chỉ không được bỏ trống!!!");  txtDiaChi.requestFocus();  return false;  }else if (txtEmail.getText().length() == 0) {  MsgBox.alert(this, "Email không được bỏ trống!!!");  txtEmail.requestFocus();  return false;  }else if (verifyEmail(txtEmail.getText()) == false) {  MsgBox.alert(this, "Định dạng email bạn nhập không chính xác");  txtEmail.requestFocus();  return false;  }  return true;} |

### Xử lý Form Nhà cung cấp



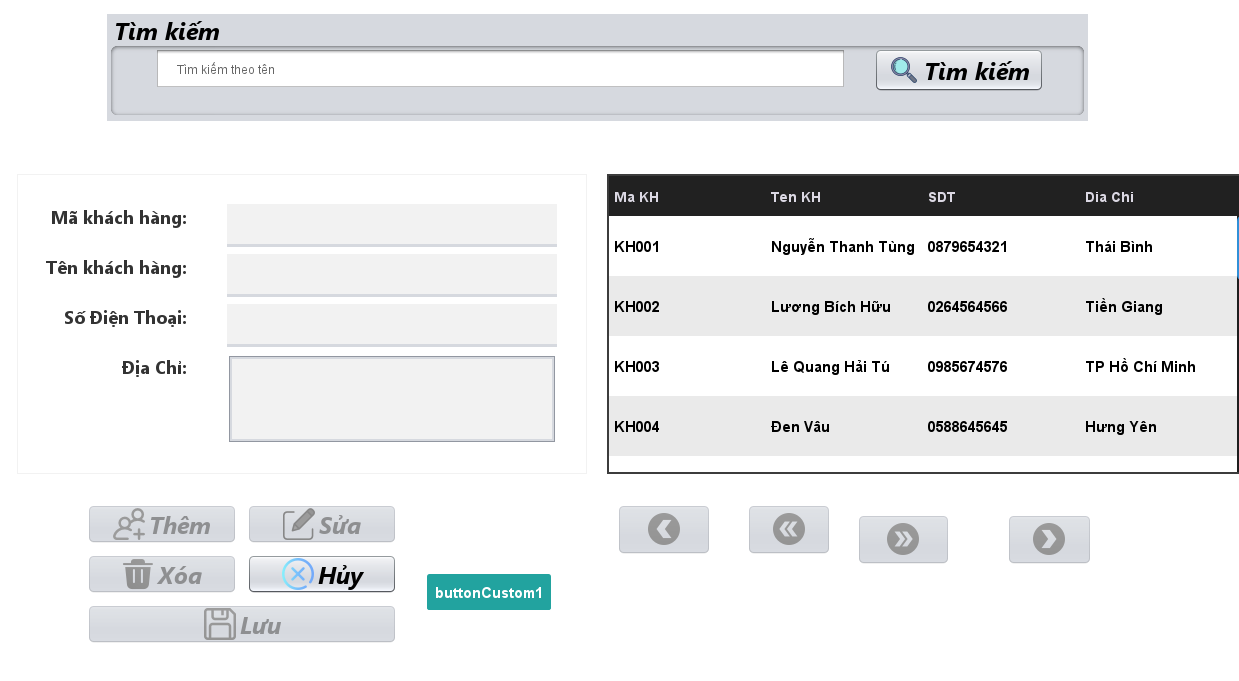
Mô tả hoạt động

|  |  |
| --- | --- |
| Sự kiện | Công việc phải làm |
| Constructor | init(); |
| tblNCCMouseClicked | if (evt.getClickCount() == 1) {  this.row = tblNCC.getSelectedRow();  this.edit();  tblNCC.setSelectionBackground(new Color(0, 156, 222));  } |
| btnThem | them = 1;  btnSua.setEnabled(false);  btnXoa.setEnabled(false);  btnLuu.setEnabled(true);  txtMaNCC.setText("");  txtTenNCC.setText("");  txtON(); |
| btnSua | them = 2;  btnThem.setEnabled(false);  btnXoa.setEnabled(false);  btnLuu.setEnabled(true);  txtON();  txtMaNCC.setEditable(false); |
| btnXoa | delete(); |
| btnLuu | if (check() == true) {  if (them == 1) {  insert();  return;  }  if (them == 2) {  update();  return;  }  } |
| btnHuy | updateStatus();  clearForm();  btnLuu.setEnabled(false); |
| btnFirst | first(); |
| btnPre | prev(); |
| btnNext | next(); |
| btnLast | last(); |

Hướng dẫn cài đặt mã nguồn

|  |
| --- |
| NhaCungCapDAO dao = new NhaCungCapDAO();  int row = 0;  int them = 0;  int checklap = 0;  ArrayList<NhaCungCap> listNCC = new ArrayList<>();  private void init() {  this.row = -1;  fillTable();  this.updateStatus();  btnLuu.setEnabled(false);  }  public void fillTable() {  DefaultTableModel model = (DefaultTableModel) tblNCC.getModel();  model.setRowCount(0);  listNCC.clear();  try {  List<NhaCungCap> list = dao.selectAll();  for (NhaCungCap nv : list) {  Object[] data = {  nv.getMaNCC(),  nv.getTenNCC()  };  listNCC.add(nv);  model.addRow(data);  }  } catch (Exception e) {  MsgBox.alert(this, "Lỗi truy vấn dữ liệu!");  }  }  public void insert() {  NhaCungCap ncc = getForm();  try {  dao.insert(ncc);  this.fillTable();  this.clearForm();  MsgBox.alert(this, "Thêm mới thành công");  updateStatus();  them = 0;  btnLuu.setEnabled(false);  } catch (Exception e) {  MsgBox.alert(this, "Thêm mới thất bại");  updateStatus();  them = 0;  btnLuu.setEnabled(false);  }  }  public void update() {  NhaCungCap nv = getForm();  try {  dao.update(nv);  this.fillTable();  MsgBox.alert(this, "Cập nhật thành công");  updateStatus();  them = 0;  btnLuu.setEnabled(false);  } catch (Exception e) {  MsgBox.alert(this, "Cập nhật thất bại");  updateStatus();  them = 0;  btnLuu.setEnabled(false);  }  }  public void delete() {  String ncc = txtMaNCC.getText();  if (MsgBox.confirm(this, "Bạn thực sự muốn xóa nhà cung cấp này")) {  try {  dao.hide(ncc);  //dao.delete(ncc);  this.fillTable();  this.clearForm();  MsgBox.alert(this, "Xóa thành công");  } catch (Exception e) {  MsgBox.alert(this, "Xóa thất bại");  }  }  }  public void clearForm() {  NhaCungCap ncc = new NhaCungCap();  this.setForm(ncc);  this.row = -1;  this.updateStatus();  }  public void setForm(NhaCungCap ncc) {  txtMaNCC.setText(ncc.getMaNCC());  txtTenNCC.setText(ncc.getTenNCC());  }  NhaCungCap getForm() {  NhaCungCap ncc = new NhaCungCap();  ncc.setMaNCC(txtMaNCC.getText());  ncc.setTenNCC(txtTenNCC.getText());  ncc.setXoa(true);  return ncc;  }  public void updateStatus() {  boolean edit = (this.row >= 0);  boolean first = (this.row == 0);  boolean last = (this.row == tblNCC.getRowCount() - 1);  // Trạng thái form  btnThem.setEnabled(edit);  btnSua.setEnabled(edit);  btnXoa.setEnabled(edit);  // Trạng thái điều hướng  btnFirst.setEnabled(edit && !first);  btnPre.setEnabled(edit && !first);  btnNext.setEnabled(edit && !last);  btnLast.setEnabled(edit && !last);  txtOFF();  }  public void edit() {  String mancc = (String) tblNCC.getValueAt(this.row, 0);  NhaCungCap ncc = dao.selectById(mancc);  this.setForm(ncc);  this.updateStatus();  }  public void Dislay(int i) {  NhaCungCap v = listNCC.get(i);  txtMaNCC.setText(v.getMaNCC());  txtTenNCC.setText(v.getTenNCC());  }  public void first() {  try {  this.row = 0;  tblNCC.setSelectionBackground(new Color(0, 156, 222));  tblNCC.setRowSelectionInterval(row, row);  this.edit();  Dislay(row);  } catch (Exception e) {  return;  }  }  public void prev() {  try {  if (this.row > 0) {  this.row--;  this.edit();  }  tblNCC.setSelectionBackground(new Color(0, 156, 222));  tblNCC.setRowSelectionInterval(row, row);  Dislay(row);  } catch (Exception e) {  if (row < 0) {  return;  }  }  }  public void next() {  try {  if (this.row < tblNCC.getRowCount() - 1) {  this.row++;  this.edit();  }  tblNCC.setSelectionBackground(new Color(0, 156, 222));  tblNCC.setRowSelectionInterval(row, row);  Dislay(row);  } catch (Exception e) {  if (this.row == tblNCC.getRowCount() - 1) {  return;  }  }  }  public void last() {  try {  this.row = tblNCC.getRowCount() - 1;  this.edit();  tblNCC.setSelectionBackground(new Color(0, 156, 222));  tblNCC.setRowSelectionInterval(row, row);  Dislay(row);  } catch (Exception e) {  return;  }  }  public void txtOFF() {  txtMaNCC.setEditable(false);  txtTenNCC.setEditable(false);  }  public void txtON() {  txtMaNCC.setEditable(true);  txtTenNCC.setEditable(true);  }  public boolean check() {  for (int i = 0; i < listNCC.size(); i++) {  if (listNCC.get(i).getMaNCC().equalsIgnoreCase(txtMaNCC.getText())) {  checklap = 1;  }  }  if (txtMaNCC.getText().equals("")) {  MsgBox.alert(this, "Không được để trống mã nhà cung cấp");  txtMaNCC.requestFocus();  return false;  } else if (txtMaNCC.getText().length() > 0 && txtMaNCC.getText().length() < 3) {  MsgBox.alert(this, "Mã nhà cung cấp phải nhập ít nhất 3 ký tự");  txtMaNCC.requestFocus();  return false;  } else if (them == 1 && checklap == 1) {  MsgBox.alert(this, "Mã nhà cung cấp đã tồn tại. Vui lòng nhập mã mới");  checklap = 0;  return false;  } else if (txtTenNCC.getText().equals("")) {  MsgBox.alert(this, "Không được để trống tên nhà cung cấp");  txtTenNCC.requestFocus();  return false;  }  return true;  } |

### Xử lý Form Khách hàng



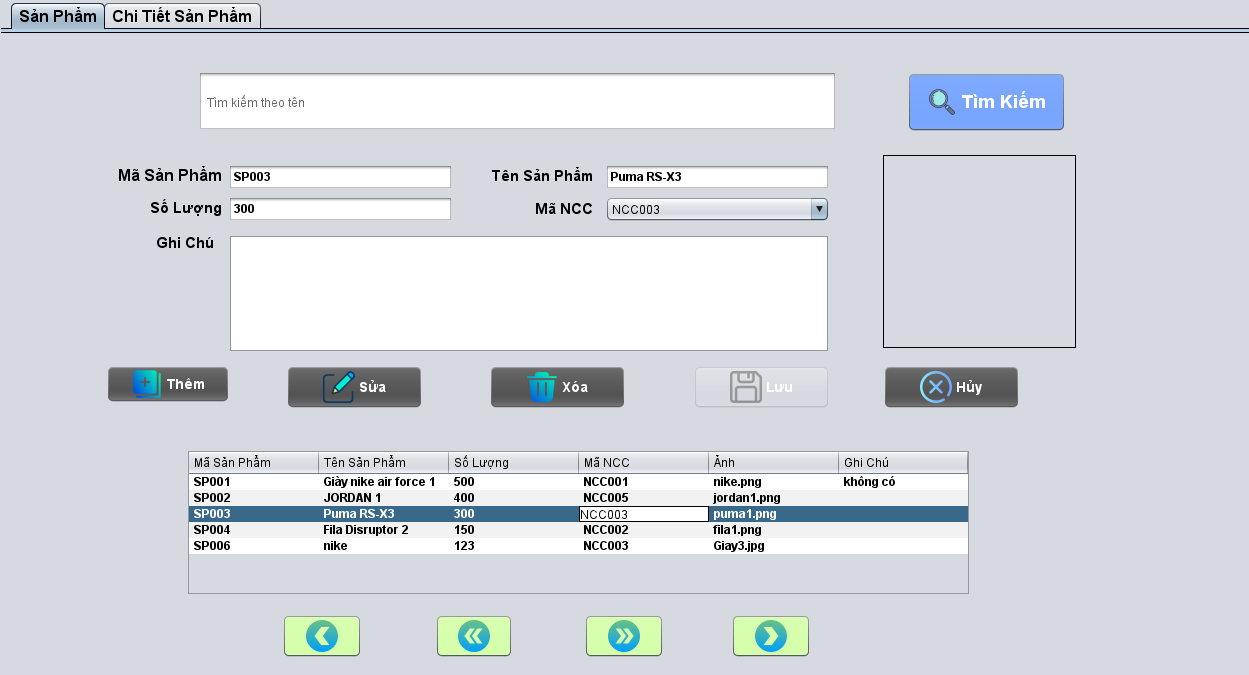
Mô tả hoạt động

|  |  |
| --- | --- |
| Sự kiện | Công việc phải làm |
| Constructor | init();  fillTable(); |
| tblkhachHangMouseClicked | if (evt.getClickCount() == 1) {  this.row = tblkhachHang.getSelectedRow();  this.edit();  } |
| btnThem | them = 1;  btnSua.setEnabled(false);  btnXoa.setEnabled(false);  btnLuu.setEnabled(true);  txtmaKhachHang.setText("");  txttenKhachHang.setText("");  txtsoDienThoai.setText("");  txtdiaChi.setText("");  txtON(); |
| btnSua | them = 2;  btnThem.setEnabled(false);  btnXoa.setEnabled(false);  btnLuu.setEnabled(true);  fillTable();  txtON();  txtmaKhachHang.setEditable(false); |
| btnXoa | delete(); |
| btnLuu | if (check() == true) {  if (them == 1) {  insert();  return;  }  if (them == 2) {  update();  return;  }  } |
| btnHuy | updateStatus();  clearForm();  btnLuu.setEnabled(false); |
| btnFirst | first(); |
| btnPre | prev(); |
| btnNext | next(); |
| btnLast | last(); |
| txtTimKiem | if (txttimKiem.getText().equals("")) {  lblTimKiem.setVisible(true);  } else  lblTimKiem.setVisible(false); |
| btnTimKiem | this.TimKiem(); |

Hướng dẫn cài đặt mã nguồn

|  |
| --- |
| KhachHangDAO dao = new KhachHangDAO();  int row = -1;  int them = 0;  int checklap = 0;  ArrayList<KhachHang> listKH = new ArrayList<>();  private void init() {  this.row = -1;  fillTable();  this.updateStatus();  btnLuu.setEnabled(false);  }  public void fillTable() {  DefaultTableModel model = (DefaultTableModel) tblkhachHang.getModel();  model.setRowCount(0);  try {  List<KhachHang> list = dao.selectAll(); // Đọc dữ liệu từ CSDL  for (KhachHang kh : list) {  Object[] data = {  kh.getMaKH(),  kh.getTenKH(),  kh.getSDT(),  kh.getDiaChi(),};  listKH.add(kh);  model.addRow(data);  }  } catch (Exception e) {  MsgBox.alert(this, "Lỗi truy vấn dữ liệu!");  }  }  public void insert() {  KhachHang kh = getForm();  try {  dao.insert(kh);  this.fillTable();  this.clearForm();  MsgBox.alert(this, "Thêm mới thành công");  updateStatus();  them = 0;  btnLuu.setEnabled(true);  } catch (Exception e) {  MsgBox.alert(this, "Thêm mới thất bại");  updateStatus();  them = 0;  btnLuu.setEnabled(true);  }  }  public void update() {  KhachHang kh = getForm();  try {  dao.update(kh);  this.fillTable();  MsgBox.alert(this, "Cập nhật thành công");  them = 0;  btnLuu.setEnabled(true);  } catch (Exception e) {  MsgBox.alert(this, "Cập nhật thất bại");  updateStatus();  them = 0;  }  }  public void delete() {  String kh = txtmaKhachHang.getText();  if (MsgBox.confirm(this, "Bạn thực sự muốn xóa khách hàng này")) {  try {  dao.hide(kh);  this.fillTable();  this.clearForm();  MsgBox.alert(this, "Xóa thành công");  } catch (Exception e) {  MsgBox.alert(this, "Xóa thất bại");  }  }  }  public void clearForm() {  KhachHang kh = new KhachHang();  this.setForm(kh);  this.row = -1;  this.updateStatus();  }  txtmaKhachHang.setText(kh.getMaKH());  txttenKhachHang.setText(kh.getTenKH());  txtsoDienThoai.setText(kh.getSDT());  txtdiaChi.setText(kh.getDiaChi());  }  KhachHang getForm() {  KhachHang kh = new KhachHang();  kh.setMaKH(txtmaKhachHang.getText());  kh.setTenKH(txttenKhachHang.getText());  kh.setSDT(txtsoDienThoai.getText());  kh.setDiaChi(txtdiaChi.getText());  kh.setXoa(true);  return kh;  }  public void updateStatus() {  boolean edit = (this.row >= 0);  boolean first = (this.row == 0);  boolean last = (this.row == tblkhachHang.getRowCount() - 1);  // Trạng thái form  btnThem.setEnabled(edit);  btnSua.setEnabled(edit);  btnXoa.setEnabled(edit);  // Trạng thái điều hướng  btnFirst.setEnabled(edit && !first);  btnPrev.setEnabled(edit && !first);  btnNext.setEnabled(edit && !last);  btnLast.setEnabled(edit && !last);  txtOFF();  }  public void edit() {  String makh = (String) tblkhachHang.getValueAt(this.row, 0);  KhachHang kh = dao.selectById(makh);  this.setForm(kh);  this.updateStatus();  }  public void first() {  this.row = 0;  tblkhachHang.setRowSelectionInterval(row, row);  this.edit();  }  public void prev() {  if (this.row > 0) {  this.row--;  tblkhachHang.setRowSelectionInterval(row, row);  this.edit();  }  }  public void next() {  if (this.row < tblkhachHang.getRowCount() - 1) {  this.row++;  tblkhachHang.setRowSelectionInterval(row, row);  this.edit();  }  }  public void last() {  this.row = tblkhachHang.getRowCount() - 1;  tblkhachHang.setRowSelectionInterval(row, row);  this.edit();  }  public void txtOFF() {  txtmaKhachHang.setEditable(false);  txttenKhachHang.setEditable(false);  txtsoDienThoai.setEditable(false);  txtdiaChi.setEditable(false);  txtmaKhachHang.setFocusable(false);  txttenKhachHang.setFocusable(false);  txtsoDienThoai.setFocusable(false);  txtdiaChi.setFocusable(false);  }  public void txtON() {  txtmaKhachHang.setEditable(true);  txttenKhachHang.setEditable(true);  txtsoDienThoai.setEditable(true);  txtdiaChi.setEditable(true);  txtmaKhachHang.setFocusable(true);  txttenKhachHang.setFocusable(true);  txtsoDienThoai.setFocusable(true);  txtdiaChi.setFocusable(true);  }  public boolean check() {  boolean checkSDT = true;  try {  Float.parseFloat(txtsoDienThoai.getText());  } catch (NumberFormatException e1) {  checkSDT = false;  }  for (int i = 0; i < listKH.size(); i++) {  if (listKH.get(i).getMaKH().equalsIgnoreCase(txtmaKhachHang.getText())) {  checklap = 1;  }  }  if (txtmaKhachHang.getText().equals("") || txtmaKhachHang.getText().length() < 5 || txtmaKhachHang.getText().length() > 6) {  MsgBox.alert(this, "Vui lòng nhập mã khách hàng từ 5---->6 kí tự");  txtmaKhachHang.requestFocus();  return false;  } else if (them == 1 && checklap == 1) {  MsgBox.alert(this, "Mã khách hàng đã tồn tại. Vui lòng nhập mã mới");  checklap = 0;  return false;  } else if (txttenKhachHang.getText().length() == 0) {  MsgBox.alert(this, "Tên khách hàng không được bỏ trống!!!");  txttenKhachHang.requestFocus();  return false;  } else if (checkSDT == false) {  MsgBox.alert(this, "Vui lòng nhập số");  txtsoDienThoai.requestFocus();  return false;  } else if (txtsoDienThoai.getText().equals("") || txtsoDienThoai.getText().length() < 9 || txtsoDienThoai.getText().length() > 10) {  MsgBox.alert(this, "Vui lòng nhập số điện thoại từ 9---->10 kí tự");  txtsoDienThoai.requestFocus();  return false;  } else if (txtdiaChi.getText().length() == 0) {  MsgBox.alert(this, "Địa chỉ không được bỏ trống!!!");  txtdiaChi.requestFocus();  return false;  }  return true;  }  public void TimKiem() {  DefaultTableModel model = (DefaultTableModel) tblkhachHang.getModel();  model.setRowCount(0);  try {  String khh = txttimKiem.getText();  List<KhachHang> list = dao.selectByKeyword(khh);  for (KhachHang kh : list) {  Object[] data = {  kh.getMaKH(),  kh.getTenKH(),  kh.getSDT(),  kh.getDiaChi(),};  model.addRow(data);  }  } catch (Exception e) {  MsgBox.alert(this, e.getMessage());  }  this.clearForm();  this.row = -1;  updateStatus();  } |

### Xử lý Form Sản phẩm



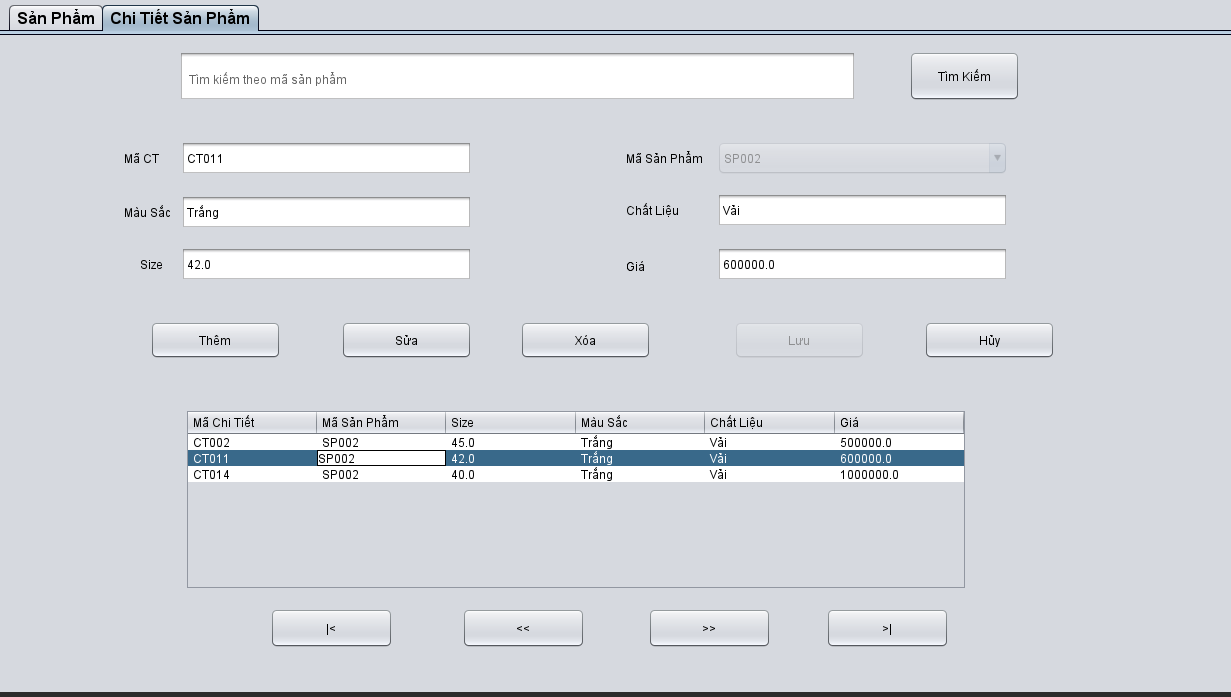
Mô tả hoạt động

|  |  |
| --- | --- |
| Sự kiện | Công việc phải làm |
| Constructor | Init(); |
| tblSanPhamMousePressed | if (evt.getClickCount() == 1) {  this.row = tblSanPham.getSelectedRow();  fillCboMaNCC();  this.edit();  }  if (evt.getClickCount() == 2) {  tabs.setSelectedComponent(jPanel2);  fillTable2();  } |
| btnThem | them = 1;  btnUpdate.setEnabled(false);  btnXoa.setEnabled(false);  btnLuu.setEnabled(true);  txtMaSP.setText("");  txtTenSP.setText("");  cboMaNCC.setSelectedIndex(0);  txtSoLuong.setText("");  txtGhiChu.setText("");  ImageIcon icon = new ImageIcon(".//src//com//nhom4//icon//user.png");  lblAnh.setIcon(icon = null);  txtON(); |
| btnUpdate | them = 2;  btnThem.setEnabled(false);  btnXoa.setEnabled(false);  btnLuu.setEnabled(true);  txtON();  txtMaSP.setEditable(false); |
| btnLuu | if (kiemTra() == true) {  if (them == 1) {  insert();  return;  }  if (them == 2) {  update();  return;  }  } |
| btnXoa | delete(); |
| btnHuy | updateStatus();  clearForm();  ImageIcon icon = new ImageIcon(".//src//com//nhom4//icon//user.png");  lblAnh.setIcon(icon = null);  btnLuu.setEnabled(false);  btnThem.setEnabled(true); |
| btnFirst | this.row = 0;  tblSanPham.setRowSelectionInterval(row, row);  this.edit(); |
| btnPre | if (this.row > 0) {  this.row--;  tblSanPham.setRowSelectionInterval(row, row);  this.edit();  } |
| BtnNext | if (this.row < tblSanPham.getRowCount() - 1) {  this.row++;  tblSanPham.setRowSelectionInterval(row, row);  this.edit();  } |
| btnLast | this.row = tblSanPham.getRowCount() - 1;  tblSanPham.setRowSelectionInterval(row, row);  this.edit(); |
| txtTimKiem | if (txtTimKiem.getText().equals("")) {  lblTimKiem.setVisible(true);  } else {  lblTimKiem.setVisible(false);  } |
| btnTimKiem | TimKiem(); |
| lblAnhMouseClicked | this.selectImage(); |

Hướng dẫn cài đặt mã nguồn

|  |
| --- |
| SanPhamDAO spDao = new SanPhamDAO();  JFileChooser fileChooser = new JFileChooser(".//src//com//nhom4//hinh");  ArrayList<SanPham> splist = new ArrayList<>();  int row = -1;  int them = 0;  int checklap = 0;  Void init(){  initTable();  fillTable();  this.row = -1;  this.updateStatus();  btnThem.setEnabled(true);  btnLuu.setEnabled(false);  fillCboMaNCC();  }  private void initTable() {  DefaultTableModel model = (DefaultTableModel) tblSanPham.getModel();  String[] cols = new String[]{"Mã Sản Phẩm", "Tên Sản Phẩm", "Số Lượng", "Mã NCC", "Ảnh", "Ghi Chú"};  model.setColumnIdentifiers(cols);  }  public void fillTable() {  DefaultTableModel model = (DefaultTableModel) tblSanPham.getModel();  model.setRowCount(0);  splist.clear();  try {  List<SanPham> list = spDao.selectAll(); // Đọc dữ liệu từ CSDL  for (SanPham sp : list) {  Object[] data = {  sp.getMaSP(),  sp.getTenSP(),  sp.getSoLuong(),  sp.getMaNCC(),  sp.getAnh(),  sp.getGhiChu()  };  splist.add(sp);  model.addRow(data);  }  } catch (Exception e) {  }  }  public void updateStatus() {  boolean edit = (this.row >= 0);  boolean first = (this.row == 0);  boolean last = (this.row == tblSanPham.getRowCount() - 1);  // Trạng thái form  btnThem.setEnabled(edit);  btnUpdate.setEnabled(edit);  btnXoa.setEnabled(edit);  // Trạng thái điều hướng  btnFirst.setEnabled(edit && !first);  btnPre.setEnabled(edit && !first);  btnNext.setEnabled(edit && !last);  btnLast.setEnabled(edit && !last);  txtOFF();  }  private void fillCboMaNCC() {  DefaultComboBoxModel model = (DefaultComboBoxModel) cboMaNCC.getModel();  model.removeAllElements();  List<String> list = spDao.selectMaNCC();  for (String nv : list) {  model.addElement(nv);  }  }  public void setForm(SanPham sp) {  txtMaSP.setText(sp.getMaSP());  txtTenSP.setText(sp.getTenSP());  txtSoLuong.setText(String.valueOf(sp.getSoLuong()));  cboMaNCC.setSelectedItem(sp.getMaNCC());  txtGhiChu.setText(sp.getGhiChu());  lblAnh.setToolTipText(sp.getAnh());  if (sp.getAnh() != null) {  lblAnh.setIcon(XImage.read(sp.getAnh()));  lblAnh.setToolTipText(sp.getAnh());  }  }  SanPham getForm() {  SanPham sp = new SanPham();  sp.setMaSP(txtMaSP.getText());  sp.setTenSP(txtTenSP.getText());  sp.setSoLuong(Integer.valueOf(txtSoLuong.getText()));  sp.setMaNCC(cboMaNCC.getSelectedItem().toString());  sp.setGhiChu(txtGhiChu.getText());  sp.setAnh(lblAnh.getToolTipText());  sp.setXoa(true);  return sp;  }  public void edit() {  try {  String masp = (String) tblSanPham.getValueAt(this.row, 0);  SanPham sp = spDao.selectById(masp);  if (sp != null) {  this.setForm(sp);  this.updateStatus();  }  } catch (Exception e) {  }  }  void selectImage() {  if (fileChooser.showOpenDialog(this) == JFileChooser.APPROVE\_OPTION) {  File file = fileChooser.getSelectedFile();  if (XImage.save(file)) {  // Hiển thị hình lên form  lblAnh.setIcon(XImage.read(file.getName()));  lblAnh.setToolTipText(file.getName());  }  }  }  void clearForm() {  SanPham sp = new SanPham();  this.setForm(sp);  this.row = -1;  this.updateStatus();  }  void insert() {  SanPham sp = getForm();  try {  spDao.insert(sp);  this.fillTable();  MsgBox.alert(this, "Thêm mới thành công");  updateStatus();  this.clearForm();  them = 0;  btnLuu.setEnabled(false);  } catch (Exception e) {  MsgBox.alert(this, "Thêm mới thất bại");  System.out.println(e);  updateStatus();  them = 0;  btnLuu.setEnabled(false);  }  }  public void delete() {  if (MsgBox.confirm(this, "Bạn thực sự muốn xóa sản phẩm này")) {  String msp = txtMaSP.getText();  try {  spDao.hide(msp);  this.fillTable();  this.clearForm();  MsgBox.alert(this, "Xóa thành công");  } catch (Exception e) {  MsgBox.alert(this, "Xóa thất bại");  }  }  }  public void update() {  SanPham sp = getForm();  try {  spDao.update(sp);  this.fillTable();  MsgBox.alert(this, "Cập nhật thành công");  updateStatus();  them = 0;  btnLuu.setEnabled(false);  } catch (Exception e) {  MsgBox.alert(this, "Cập nhật thất bại");  updateStatus();  them = 0;  btnLuu.setEnabled(false);  }  }  boolean kiemTra() {  for (int i = 0; i < splist.size(); i++) {  if (splist.get(i).getMaSP().equalsIgnoreCase(txtMaSP.getText())) {  checklap = 1;  }  }  if (txtMaSP.getText().equals("") || txtMaSP.getText().length() < 5 || txtMaSP.getText().length() > 6) {  MsgBox.alert(this, "Vui lòng nhập mã sản phẩm từ 5---->6 kí tự");  txtMaSP.requestFocus();  return false;  } else if (them == 1 && checklap == 1) {  MsgBox.alert(this, "Mã sản phẩm đã tồn tại. Vui lòng nhập mã mới");  txtMaSP.requestFocus();  checklap = 0;  return false;  } else if (txtTenSP.getText().length() == 0) {  MsgBox.alert(this, "Tên Sản Phẩm không được bỏ trống!!!");  txtTenSP.requestFocus();  return false;  } else if (txtSoLuong.getText().equals("")) {  MsgBox.alert(this, "Vui lòng nhập số lượng");  txtSoLuong.requestFocus();  return false;  } else if (lblAnh.getIcon() == null) {  MsgBox.alert(this, "Bạn chưa chọn hình! Click vào khu vực hình để chọn");  return false;  } else if (!txtSoLuong.getText().equals("")) {  String sl = "";  String tv = "";  try {  if (!txtSoLuong.getText().equals("")) {  int soluong = Integer.parseInt(txtSoLuong.getText());  if (soluong <= 0) {  MsgBox.alert(this, "Số lượng phải > 0");  txtSoLuong.requestFocus();  return false;  } else {  return true;  }  }  } catch (Exception e) {  if (tv.equals(sl)) {  MsgBox.alert(this, "Số lượng là kiểu số nguyên");  txtSoLuong.requestFocus();  return false;  }  }  }  return true;  }  public void txtOFF() {  txtMaSP.setEditable(false);  txtTenSP.setEditable(false);  txtSoLuong.setEditable(false);  txtGhiChu.setEditable(false);  cboMaNCC.setEnabled(false);  }  public void txtON() {  txtMaSP.setEditable(true);  txtTenSP.setEditable(true);  txtSoLuong.setEditable(true);  txtGhiChu.setEditable(true);  cboMaNCC.setEnabled(true);  } |

### Xử lý Form Chi tiết sản phẩm



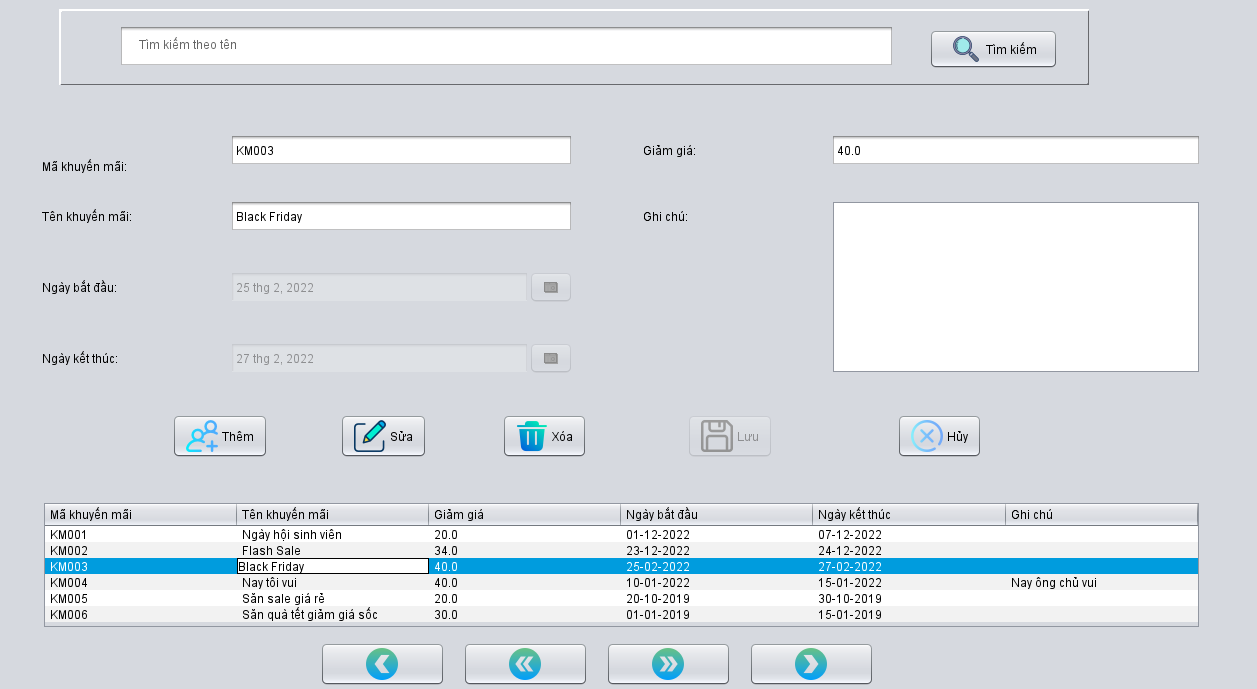
Mô tả hoạt động

|  |  |
| --- | --- |
| Sự kiện | Công việc phải làm |
| Constructor | Init(); |
| tblChiTietSpMousePressed | if (evt.getClickCount() == 1) {  this.row = tblChiTietSp.getSelectedRow();  fillCboMaSP();  this.edit2();  } |
| BtnReset | them = 1;  btnCapNhat.setEnabled(false);  btnDelete.setEnabled(false);  btnSave.setEnabled(true);  txtMaCT.setText("");  cboMaSP.setSelectedIndex(0);  txtChatLieu.setText("");  txtSize.setText("");  txtGia.setText("");  txtMauSac.setText("");  txtON2(); |
| btnCapNhat | them = 2;  btnReset.setEnabled(false);  btnDelete.setEnabled(false);  btnSave.setEnabled(true);  txtON2();  txtMaCT.setEditable(false);  cboMaSP.setEditable(false); |
| btnDelete | delete2(); |
| btnSave | if (kiemTra2() == true) {  if (them == 1) {  insert2();  return;  }  if (them == 2) {  update2();  return;  }  } |
| BtnHuyy | updateStatus2();  clearForm2();  btnSave.setEnabled(false);  btnReset.setEnabled(true); |
| btnDau | this.row = 0;  tblChiTietSp.setRowSelectionInterval(row, row);  this.edit2(); |
| btnToi | if (this.row < tblChiTietSp.getRowCount() - 1) {  this.row++;  tblChiTietSp.setRowSelectionInterval(row, row);  this.edit2();  } |
| btnLui | if (this.row > 0) {  this.row--;  tblChiTietSp.setRowSelectionInterval(row, row);  this.edit2();  } |
| btnCuoi | this.row = tblChiTietSp.getRowCount() - 1;  tblChiTietSp.setRowSelectionInterval(row, row);  this.edit2(); |
| txtTK | if (txtTK.getText().equals("")) {  lblTK.setVisible(true);  } else {  lblTK.setVisible(false);  } |
| btnTK | TimKiem2(); |

Hướng dẫn cài đặt mã nguồn

|  |
| --- |
| ArrayList<ChiTietSanPham> ctlist = new ArrayList<>();  ChiTietSanPhamDAO ctspDAO = new ChiTietSanPhamDAO();  int row = -1;  int them = 0;  int checklap = 0;  Void init(){  initTable();  this.row = -1;  this.updateStatus2();  btnReset.setEnabled(true);  btnSave.setEnabled(false);  fillCboMaSP();  initTable2();  }  private void fillCboMaSP() {  DefaultComboBoxModel model = (DefaultComboBoxModel) cboMaSP.getModel();  model.removeAllElements();  List<String> list = ctspDAO.selectMaSP();  for (String nv : list) {  model.addElement(nv);  }  }  private void initTable2() {  DefaultTableModel model = (DefaultTableModel) tblChiTietSp.getModel();  String[] cols = new String[]{"Mã Chi Tiết", "Mã Sản Phẩm", "Size", "Màu Sắc", "Chất Liệu", "Giá"};  model.setColumnIdentifiers(cols);  }  public void fillTable2() {  DefaultTableModel model = (DefaultTableModel) tblChiTietSp.getModel();  model.setRowCount(0);  ctlist.clear();  try {  String id = txtMaSP.getText();  List<ChiTietSanPham> list = ctspDAO.selectByKeyword(id); // Đọc dữ liệu từ CSDL  for (ChiTietSanPham ct : list) {  Object[] data = {  ct.getMaCT(),  ct.getMaSP(),  ct.getSize(),  ct.getMauSac(),  ct.getChatLieu(),  ct.getGia()  };  ctlist.add(ct);  model.addRow(data);  }  } catch (Exception e) {  }  }  public void setForm2(ChiTietSanPham ctsp) {  txtMaCT.setText(ctsp.getMaCT());  cboMaSP.setSelectedItem(ctsp.getMaSP());  txtSize.setText(String.valueOf(ctsp.getSize()));  txtMauSac.setText(ctsp.getMauSac());  txtChatLieu.setText(ctsp.getChatLieu());  txtGia.setText(String.valueOf(ctsp.getGia()));  }  ChiTietSanPham getForm2() {  ChiTietSanPham ctsp = new ChiTietSanPham();  ctsp.setMaCT(txtMaCT.getText());  ctsp.setMaSP(cboMaSP.getSelectedItem().toString());  ctsp.setSize(Float.valueOf(txtSize.getText()));  ctsp.setMauSac(txtMauSac.getText());  ctsp.setGia(Float.valueOf(txtGia.getText()));  ctsp.setChatLieu(txtChatLieu.getText());  return ctsp;  }  public void updateStatus2() {  boolean edit = (this.row >= 0);  boolean first = (this.row == 0);  boolean last = (this.row == tblChiTietSp.getRowCount() - 1);  // Trạng thái form  btnReset.setEnabled(edit);  btnCapNhat.setEnabled(edit);  btnDelete.setEnabled(edit);  // Trạng thái điều hướng  btnDau.setEnabled(edit && !first);  btnLui.setEnabled(edit && !first);  btnToi.setEnabled(edit && !last);  btnCuoi.setEnabled(edit && !last);  txtOFF2();  }  void clearForm2() {  ChiTietSanPham ctsp = new ChiTietSanPham();  this.setForm2(ctsp);  this.row = -1;  this.updateStatus2();  }  public void edit2() {  try {  String mact = (String) tblChiTietSp.getValueAt(this.row, 0);  ChiTietSanPham ctsp = ctspDAO.selectById(mact);  if (ctsp != null) {  this.setForm2(ctsp);  this.updateStatus2();  }  } catch (Exception e) {  }  }  void insert2() {  ChiTietSanPham ctsp = getForm2();  try {  ctspDAO.insert(ctsp);  this.fillTable2();  MsgBox.alert(this, "Thêm mới thành công");  updateStatus2();  them = 0;  btnSave.setEnabled(false);  } catch (Exception e) {  MsgBox.alert(this, "Thêm mới thất bại");  updateStatus2();  them = 0;  btnSave.setEnabled(false);  }  }  public void delete2() {  if (MsgBox.confirm(this, "Bạn thực sự muốn xóa chi tiết sản phẩm này")) {  String mct = txtMaCT.getText();  try {  ctspDAO.delete(mct);  this.fillTable2();  this.clearForm2();  MsgBox.alert(this, "Xóa thành công");  } catch (Exception e) {  MsgBox.alert(this, "Xóa thất bại");  }  }  }  public void update2() {  ChiTietSanPham ctsp = getForm2();  try {  ctspDAO.update(ctsp);  this.fillTable2();  MsgBox.alert(this, "Cập nhật thành công");  updateStatus2();  them = 0;  btnSave.setEnabled(false);  } catch (Exception e) {  MsgBox.alert(this, "Cập nhật thất bại");  updateStatus2();  them = 0;  btnSave.setEnabled(false);  }  }  boolean kiemTra2() {  for (int i = 0; i < ctlist.size(); i++) {  if (ctlist.get(i).getMaCT().equalsIgnoreCase(txtMaCT.getText())) {  checklap = 1;  }  }  if (txtMaCT.getText().equals("") || txtMaCT.getText().length() < 5 || txtMaCT.getText().length() > 6) {  MsgBox.alert(this, "Vui lòng nhập mã chi tiết sản phẩm từ 5---->6 kí tự");  txtMaCT.requestFocus();  return false;  } else if (them == 1 && checklap == 1) {  MsgBox.alert(this, "Mã chi tiết sản phẩm đã tồn tại. Vui lòng nhập mã mới");  txtMaCT.requestFocus();  checklap = 0;  return false;  } else if (txtMauSac.getText().equals("")) {  MsgBox.alert(this, "Vui lòng nhập màu sắc");  txtMauSac.requestFocus();  return false;  } else if (txtChatLieu.getText().equals("")) {  MsgBox.alert(this, "Vui lòng nhập chất liệu");  txtChatLieu.requestFocus();  return false;  } else if (txtSize.getText().equals("")) {  MsgBox.alert(this, "Vui lòng nhập size giày");  txtSize.requestFocus();  return false;  } else if (txtGia.getText().equals("")) {  MsgBox.alert(this, "Vui lòng nhập giá giày");  txtGia.requestFocus();  return false;  } else if (!txtSize.getText().equals("")) {  String size = "";  String tv = "";  try {  if (!txtSize.getText().equals("") || !txtGia.getText().equals("")) {  float siZe = Float.parseFloat(txtSize.getText());  float Gia = Float.parseFloat(txtGia.getText());  if (siZe <= 0) {  MsgBox.alert(this, "Size giày phải > 0");  txtSize.requestFocus();  return false;  } else if (Gia < 50000) {  MsgBox.alert(this, "Giá giày phải >= 50K");  txtGia.requestFocus();  return false;  } else {  return true;  }  }  } catch (Exception e) {  if (tv.equals(size)) {  MsgBox.alert(this, "Size giày và giá giày là kiểu số");  return false;  }  }  }  return true;  }  public void TimKiem2() {  DefaultTableModel model = (DefaultTableModel) tblChiTietSp.getModel();  model.setRowCount(0);  try {  String key = txtTK.getText();  List<ChiTietSanPham> list = ctspDAO.selectByKeyword(key);  for (ChiTietSanPham nh : list) {  Object[] data = {  nh.getMaCT(),  nh.getMaSP(),  nh.getMauSac(),  nh.getChatLieu(),  nh.getSize(),  nh.getGia(),};  model.addRow(data);  }  } catch (Exception e) {  MsgBox.alert(this, e.getMessage());  }  this.clearForm2();  this.row = -1;  updateStatus2();  }  public void txtOFF2() {  txtMaCT.setEditable(false);  cboMaSP.setEnabled(false);  txtChatLieu.setEditable(false);  txtMauSac.setEditable(false);  txtGia.setEditable(false);  txtSize.setEditable(false);  }  public void txtON2() {  txtMaCT.setEditable(true);  cboMaSP.setEnabled(true);  txtChatLieu.setEditable(true);  txtMauSac.setEditable(true);  txtGia.setEditable(true);  txtSize.setEditable(true);  } |

### Xử lý Form Khuyến mãi



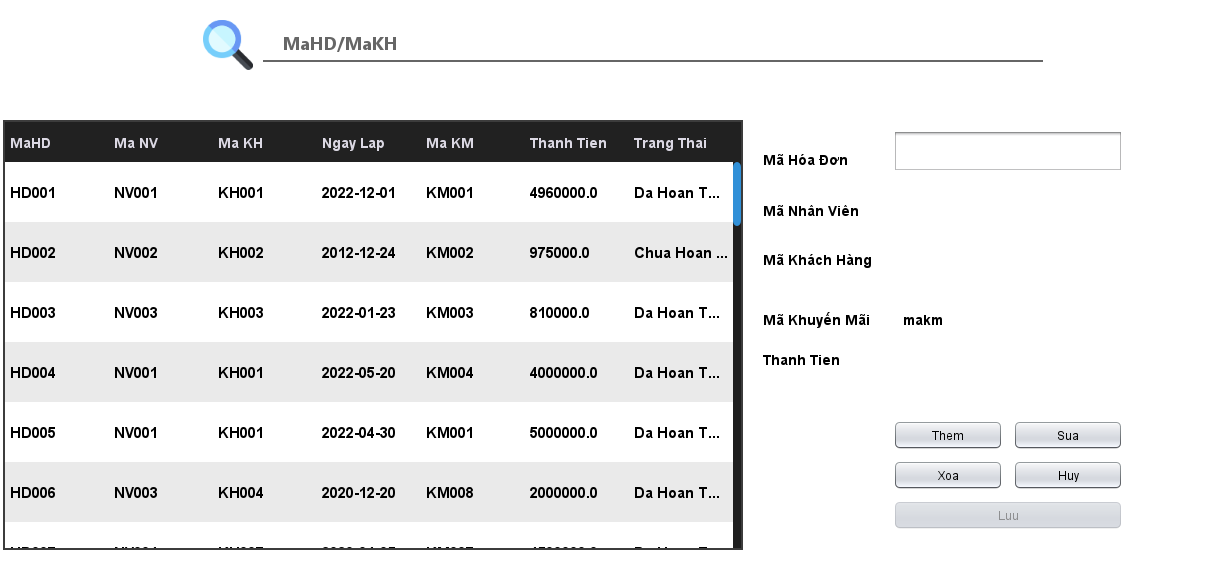
Mô tả công việc

|  |  |
| --- | --- |
| Sự kiện | Công việc phải làm |
| Constructor | init(); |
| tblKhuyenMaiMouseClicked | if (evt.getClickCount() == 1) {  this.row = tblKhuyenMai.getSelectedRow();  this.edit();  tblKhuyenMai.setSelectionBackground(new Color(0, 156, 222));  } |
| BtnThem | them = 1;  btnSua.setEnabled(false);  btnXoa.setEnabled(false);  btnLuu.setEnabled(true);  txtMaKM.setText("");  txtTenKM.setText("");  txtGiamGia.setText("");  txtGhiChu.setText("");  txtON(); |
| btnSua | them = 2;  btnThem.setEnabled(false);  btnXoa.setEnabled(false);  btnLuu.setEnabled(true);  txtON();  txtMaKM.setEditable(false); |
| btnXoa | if (checkXoa() == true) {  delete();  } |
| btnLuu | if (them == 1) {  if (check() == true) {  insert();  return;  }  }  if (them == 2) {  if (checkXoa()) {  update();  return;  }  } |
| btnHuy | updateStatus();  clearForm();  btnLuu.setEnabled(false); |
| btnFirst | first(); |
| btnPre | prev(); |
| btnNext | next(); |
| btnLast | last(); |
| txtTimKiem | if(txtTimKiem.getText().equals("")){  lblTimKiem.setVisible(true);  }  else  lblTimKiem.setVisible(false); |
| btnTimKiem | TimKiem(); |

Hướng dẫn cài đặt mã nguồn

|  |
| --- |
| Date now = new Date();  SimpleDateFormat formats = new SimpleDateFormat("dd-MM-yyyy");  ArrayList<KhuyenMai> listKM = new ArrayList<>();  KhuyenMaiDAO dao = new KhuyenMaiDAO();  int row = 0;  int them = 0;  int checklap = 0;  private void init() {  this.row = -1;  fillTable();  btnLuu.setEnabled(false);  this.updateStatus();  txtNgayBD.setLocale(Locale.forLanguageTag("vi"));  txtNgayKT.setLocale(Locale.forLanguageTag("vi"));  }  public void fillTable() {  DefaultTableModel model = (DefaultTableModel) tblKhuyenMai.getModel();  model.setRowCount(0);  try {  List<KhuyenMai> list = dao.selectAll(); // Đọc dữ liệu từ CSDL  for (KhuyenMai nv : list) {  Object[] data = {  nv.getMaKM(),  nv.getTenKM(),  nv.getGiamGia(),  formats.format(nv.getNgayBD()),  formats.format(nv.getNgayKT()),  nv.getGhiChu()  };  listKM.removeAll(listKM);  listKM.add(nv);  model.addRow(data);  }  } catch (Exception e) {  MsgBox.alert(this, "Lỗi truy vấn dữ liệu!");  }  }  public void insert() {  KhuyenMai ncc = getForm();  try {  dao.insert(ncc);  this.fillTable();  this.clearForm();  MsgBox.alert(this, "Thêm mới thành công");  updateStatus();  them = 0;  btnLuu.setEnabled(false);  } catch (Exception e) {  MsgBox.alert(this, "Thêm mới thất bại");  updateStatus();  them = 0;  btnLuu.setEnabled(false);  }  }  public void update() {  KhuyenMai nv = getForm();  try {  dao.update(nv);  this.fillTable();  MsgBox.alert(this, "Cập nhật thành công");  updateStatus();  them = 0;  btnLuu.setEnabled(false);  } catch (Exception e) {  MsgBox.alert(this, e.getMessage());  System.out.println(e);  e.getMessage();  updateStatus();  them = 0;  btnLuu.setEnabled(false);  }  }  public void delete() {  String km = txtMaKM.getText();  if (MsgBox.confirm(this, "Bạn thực sự muốn xóa chương trình khuyến mãi này")) {  try {  dao.hide(km);  this.fillTable();  this.clearForm();  MsgBox.alert(this, "Xóa thành công");  clearForm();  } catch (Exception e) {  MsgBox.alert(this, "Xóa thất bại");  }  }  }  public void clearForm() {  KhuyenMai km = new KhuyenMai();  this.setForm(km);  this.row = -1;  this.updateStatus();  }  public void TimKiem() {  DefaultTableModel model = (DefaultTableModel) tblKhuyenMai.getModel();  model.setRowCount(0);  try {  String key = txtTimKiem.getText();  List<KhuyenMai> list = dao.selectByKeyword(key);  for (KhuyenMai nh : list) {  Object[] data = {  nh.getMaKM(),  nh.getTenKM(),  nh.getGiamGia(),  formats.format(new java.sql.Date(nh.getNgayBD().getTime())),  formats.format(new java.sql.Date(nh.getNgayKT().getTime())),  nh.getGhiChu(),};  model.addRow(data);  }  } catch (Exception e) {  MsgBox.alert(this, e.getMessage());  }  this.clearForm();  this.row = -1;  updateStatus();  }  public void setForm(KhuyenMai km) {  try {  txtMaKM.setText(km.getMaKM());  txtTenKM.setText(km.getTenKM());  txtGiamGia.setText(String.valueOf(km.getGiamGia()));  txtNgayBD.setDate(km.getNgayBD());  txtNgayKT.setDate(km.getNgayKT());  txtGhiChu.setText(km.getGhiChu());  } catch (Exception e) {  MsgBox.alert(this, e.getMessage());  }  }  KhuyenMai getForm() {  KhuyenMai km = new KhuyenMai();  km.setMaKM(txtMaKM.getText());  km.setTenKM(txtTenKM.getText());  km.setGiamGia(Float.valueOf(txtGiamGia.getText()));  km.setNgayBD(txtNgayBD.getDate());  km.setNgayKT(txtNgayKT.getDate());  km.setGhiChu(txtGhiChu.getText());  km.setXoa(true);  return km;  }  public void updateStatus() {  boolean edit = (this.row >= 0);  boolean first = (this.row == 0);  boolean last = (this.row == tblKhuyenMai.getRowCount() - 1);  // Trạng thái form  btnThem.setEnabled(edit);  btnSua.setEnabled(edit);  btnXoa.setEnabled(edit);  // Trạng thái điều hướng  btnFirst.setEnabled(edit && !first);  btnPre.setEnabled(edit && !first);  btnNext.setEnabled(edit && !last);  btnLast.setEnabled(edit && !last);  txtOFF();  }  public void edit() {  String mancc = (String) tblKhuyenMai.getValueAt(this.row, 0);  KhuyenMai ncc = dao.selectById(mancc);  this.setForm(ncc);  this.updateStatus();  }  public void Dislay(int i) {  KhuyenMai v = listKM.get(i);  txtMaKM.setText(v.getMaKM());  txtTenKM.setText(v.getTenKM());  txtGiamGia.setText(String.valueOf(v.getGiamGia()));  txtNgayBD.setDate(v.getNgayBD());  txtNgayKT.setDate(v.getNgayKT());  txtGhiChu.setText(v.getGhiChu());  }  public void first() {  try {  this.row = 0;  tblKhuyenMai.setSelectionBackground(new Color(0, 156, 222));  tblKhuyenMai.setRowSelectionInterval(row, row);  this.edit();  Dislay(row);  } catch (Exception e) {  return;  }  }  public void prev() {  try {  if (this.row > 0) {  this.row--;  this.edit();  }  tblKhuyenMai.setSelectionBackground(new Color(0, 156, 222));  tblKhuyenMai.setRowSelectionInterval(row, row);  Dislay(row);  } catch (Exception e) {  if (row < 0) {  return;  }  }  }  public void next() {  try {  if (this.row < tblKhuyenMai.getRowCount() - 1) {  this.row++;  this.edit();  }  tblKhuyenMai.setSelectionBackground(new Color(0, 156, 222));  tblKhuyenMai.setRowSelectionInterval(row, row);  Dislay(row);  } catch (Exception e) {  if (this.row == tblKhuyenMai.getRowCount() - 1) {  return;  }  }  }  public void last() {  try {  this.row = tblKhuyenMai.getRowCount() - 1;  this.edit();  tblKhuyenMai.setSelectionBackground(new Color(0, 156, 222));  tblKhuyenMai.setRowSelectionInterval(row, row);  Dislay(row);  } catch (Exception e) {  return;  }  }  public void txtOFF() {  txtMaKM.setEditable(false);  txtTenKM.setEditable(false);  txtGiamGia.setEditable(false);  txtNgayBD.setEnabled(false);  txtNgayKT.setEnabled(false);  txtGhiChu.setEditable(false);  }  public void txtON() {  txtMaKM.setEditable(true);  txtTenKM.setEditable(true);  txtGiamGia.setEditable(true);  txtNgayBD.setEnabled(true);  txtNgayKT.setEnabled(true);  txtGhiChu.setEditable(true);  }  public boolean check() {  boolean checkGiamGia = true;  try {  Float.parseFloat(txtGiamGia.getText());  } catch (NumberFormatException e1) {  checkGiamGia = false;  }  Date date = null;  Date now = new Date();  SimpleDateFormat df = new SimpleDateFormat("dd-MM-yyyy");  SimpleDateFormat dfday = new SimpleDateFormat("dd");  SimpleDateFormat dfmonth = new SimpleDateFormat("MM");  SimpleDateFormat dfyear = new SimpleDateFormat("yyyy");  for (int i = 0; i < listKM.size(); i++) {  if (listKM.get(i).getMaKM().equalsIgnoreCase(txtMaKM.getText())) {  checklap = 1;  }  }  if (txtMaKM.getText().equals("")) {  MsgBox.alert(this, "Không được để trống mã chương trình khuyến mãi");  txtMaKM.requestFocus();  return false;  } else if (txtMaKM.getText().length() > 0 && txtMaKM.getText().length() < 3) {  MsgBox.alert(this, "Mã chương trình khuyến mãi phải nhập ít nhất 3 ký tự");  txtMaKM.requestFocus();  return false;  } else if (them == 1 && checklap == 1) {  MsgBox.alert(this, "Mã chương trình khuyến mãi đã tồn tại. Vui lòng nhập mã mới");  checklap = 0;  return false;  } else if (txtTenKM.getText().equals("")) {  MsgBox.alert(this, "Không được để trống tên chương trình khuyến mãi");  txtTenKM.requestFocus();  return false;  } else if (txtGiamGia.getText().equals("")) {  MsgBox.alert(this, "Không được để trống giá trị giảm giá (%)");  txtGiamGia.requestFocus();  return false;  } else if (checkGiamGia == false) {  MsgBox.alert(this, "Vui lòng nhập số");  txtGiamGia.requestFocus();  } else if (Float.parseFloat(txtGiamGia.getText()) > 100 || Float.parseFloat(txtGiamGia.getText()) < 0) {  MsgBox.alert(this, "Điểm phải từ 0 -- 10");  txtGiamGia.requestFocus();  } else if (Integer.parseInt(dfyear.format(txtNgayBD.getDate())) < Integer.parseInt(dfyear.format(now))) {  MsgBox.alert(this, "Thời gian bắt đầu phải lớn hơn thời gian hiện tại!(Năm)");  return false;  } else if (Integer.parseInt(dfyear.format(txtNgayBD.getDate())) == Integer.parseInt(dfyear.format(now))) {  if (Integer.parseInt(dfmonth.format(txtNgayBD.getDate())) < Integer.parseInt(dfmonth.format(now))) {  MsgBox.alert(this, "Thời gian bắt đầu phải lớn hơn thời gian hiện tại!(Tháng)");  return false;  }  if (Integer.parseInt(dfmonth.format(txtNgayBD.getDate())) == Integer.parseInt(dfmonth.format(now))) {  if (Integer.parseInt(dfday.format(txtNgayBD.getDate())) < Integer.parseInt(dfday.format(now))) {  MsgBox.alert(this, "Thời gian bắt đầu phải lớn hơn thời gian hiện tại!(Ngày)");  return false;  }  }  } else if (Integer.parseInt(dfyear.format(txtNgayKT.getDate())) < Integer.parseInt(dfyear.format(txtNgayBD.getDate()))) {  MsgBox.alert(this, "Thời gian kết thúc phải lớn hơn thời gian bắt đầu!-Năm");  return false;  } else if (Integer.parseInt(dfyear.format(txtNgayKT.getDate())) == Integer.parseInt(dfyear.format(txtNgayBD.getDate()))) {  if (Integer.parseInt(dfmonth.format(txtNgayKT.getDate())) < Integer.parseInt(dfmonth.format(txtNgayBD.getDate()))) {  MsgBox.alert(this, "Thời gian kết thúc phải lớn hơn thời gian bắt đầu!-Tháng");  return false;  }  if (Integer.parseInt(dfmonth.format(txtNgayKT.getDate())) == Integer.parseInt(dfmonth.format(txtNgayBD.getDate()))) {  if (Integer.parseInt(dfday.format(txtNgayKT.getDate())) < Integer.parseInt(dfday.format(txtNgayBD.getDate()))) {  MsgBox.alert(this, "Thời gian kết thúc phải lớn hơn thời gian bắt đầu!-Ngày");  return false;  }  }  }  return true;  }  public boolean checkXoa() {  Date date = null;  Date now = new Date();  SimpleDateFormat df = new SimpleDateFormat("dd-MM-yyyy");  SimpleDateFormat dfday = new SimpleDateFormat("dd");  SimpleDateFormat dfmonth = new SimpleDateFormat("MM");  SimpleDateFormat dfyear = new SimpleDateFormat("yyyy");  boolean checkGiamGia = true;  try {  Float.parseFloat(txtGiamGia.getText());  } catch (NumberFormatException e1) {  checkGiamGia = false;  }  if (txtTenKM.getText().equals("")) {  MsgBox.alert(this, "Không được để trống tên chương trình khuyến mãi");  txtTenKM.requestFocus();  return false;  } else if (txtGiamGia.getText().equals("")) {  MsgBox.alert(this, "Không được để trống giá trị giảm giá (%)");  txtGiamGia.requestFocus();  return false;  } else if (checkGiamGia == false) {  MsgBox.alert(this, "Vui lòng nhập số");  txtGiamGia.requestFocus();  return false;  } else if (Float.parseFloat(txtGiamGia.getText()) > 100 || Float.parseFloat(txtGiamGia.getText()) < 0) {  MsgBox.alert(this, "Giảm giá có giá trị từ 0 --> 100 (%)");  txtGiamGia.requestFocus();  return false;  } else if (Integer.parseInt(dfyear.format(txtNgayBD.getDate())) < Integer.parseInt(dfyear.format(now)) && Integer.parseInt(dfyear.format(now)) < Integer.parseInt(dfyear.format(txtNgayKT.getDate()))) {  MsgBox.alert(this, "Chương trình đang diễn ra không thể sửa hoặc xóa!");  return false;  } else if (Integer.parseInt(dfyear.format(txtNgayBD.getDate())) == Integer.parseInt(dfyear.format(now)) && Integer.parseInt(dfyear.format(now)) == Integer.parseInt(dfyear.format(txtNgayKT.getDate()))) {  if (Integer.parseInt(dfmonth.format(txtNgayBD.getDate())) < Integer.parseInt(dfmonth.format(now)) && Integer.parseInt(dfmonth.format(now)) < Integer.parseInt(dfmonth.format(txtNgayKT.getDate()))) {  MsgBox.alert(this, "Chương trình đang diễn ra không thể sửa hoặc xóa!");  return false;  }  if (Integer.parseInt(dfmonth.format(txtNgayBD.getDate())) == Integer.parseInt(dfmonth.format(now)) && Integer.parseInt(dfmonth.format(now)) == Integer.parseInt(dfmonth.format(txtNgayKT.getDate()))) {  if (Integer.parseInt(dfday.format(txtNgayBD.getDate())) < Integer.parseInt(dfday.format(now)) && Integer.parseInt(dfday.format(now)) < Integer.parseInt(dfday.format(txtNgayKT.getDate()))) {  MsgBox.alert(this, "Chương trình đang diễn ra không thể sửa hoặc xóa!");  return false;  }  }  }  return true;  } |

### Xử lý Form Hóa đơn



Mô tả công việc

|  |  |
| --- | --- |
| Sự kiện | Công việc phải làm |
| Constructor | this.init();  cboMaNV.setVisible(false);  int i = 0; |
| tblhoadonMousePressed | if (evt.getClickCount() > 1) { jTabbedPane1.setSelectedComponent(jPanel2);  fillTableHDCT();  } |
| tblhoadonMouseClicked | int i = tblhoadon.getSelectedRow();  setForm(i);  fillCboMaCT();  setStatusHDCT(true);  fillTableHDCT(); |
| btnNew | buttonSaveHDCT = true;  buttonNewHDCT = true;  statusBtnHDCT(buttonSaveHDCT); |
| btnUpdate | int index = tblhoadon.getSelectedRow();  if (index < 0) {  MsgBox.alert(this, "Chua Chon Hoa Don");  } else {  buttonSave = true;  buttonUpdate = true;  statusBtn(buttonSave); cboMaNV.setSelectedItem(tblhoadon.getValueAt(index, 1).toString());  cboMaKH.setSelectedItem(tblhoadon.getValueAt(index, 2).toString()); cboMaKM.setSelectedItem(tblhoadon.getValueAt(index, 4).toString());  setStatus(true);  } |
| btnDelete | delete(); |
| btnCancel | cancel(); |
| btnSave | save(); |
| jLabel14 | timKiem(); |
| txtTimKiem | if (txtTimKiem.getText().equals("")) {  lblTimKiem.setVisible(true);  fillTable();  } else {  lblTimKiem.setVisible(false);  } |
| cboSLI | thanhTien(); |

Hướng dẫn cài đặt mã nguồn

|  |
| --- |
| HoaDonDAO hdDAO = new HoaDonDAO();  ChiTietSanPhamDAO spctDAO = new ChiTietSanPhamDAO();  HoaDonChiTietDAO hdctDAO = new HoaDonChiTietDAO();  SanPhamDAO spDAO = new SanPhamDAO();  NhanVienDAO nvDAO = new NhanVienDAO();  KhachHangDAO khDAO = new KhachHangDAO();  KhuyenMaiDAO kmDAO = new KhuyenMaiDAO();  Date now = new Date();  SimpleDateFormat ngay = new SimpleDateFormat("dd-MM-yyyy -- hh:mm:ss a");  String a = ngay.format(now);  String chuoi = "";  String patternPass = "^[a-zA-Z0-9]{6,15}$";  String tenNV;  String tenKH;  float khuyenMai;  private void init() {  fillCboNV();  fillCboKH();  fillCboKM();  initTable();  initTableHDCT();  fillTable();  fillTableHDCT();  fillCboMaCT();  btnSave.setEnabled(false);  btnSaveHDCT.setEnabled(false);  txtDonGia.setEditable(false);  txtThanhTien.setEditable(false);  setStatus(false);  setStatusHDCT(false);  thanhTien();  fill();  }  private void fillCboNV() {  DefaultComboBoxModel model = (DefaultComboBoxModel) cboMaNV.getModel();  model.removeAllElements();  List<String> list = hdDAO.selectMaNV();  for (String nv : list) {  model.addElement(nv);  }  }  private void fillCboKH() {  DefaultComboBoxModel model = (DefaultComboBoxModel) cboMaKH.getModel();  model.removeAllElements();  List<String> list = hdDAO.selectMaKH();  for (String nv : list) {  model.addElement(nv);  }  }  private void fillCboKM() {  DefaultComboBoxModel model = (DefaultComboBoxModel) cboMaKM.getModel();  model.removeAllElements();  List<String> list = hdDAO.selectMaKM();  for (String nv : list) {  model.addElement(nv);  }  }  private void initTable() {  DefaultTableModel model = (DefaultTableModel) tblhoadon.getModel();  String[] cols = new String[]{"MaHD", "Ma NV", "Ma KH", "Ngay Lap", "Ma KM", "Thanh Tien", "Trang Thai"};  model.setColumnIdentifiers(cols);  }  public void fillTable() {  DefaultTableModel model = (DefaultTableModel) tblhoadon.getModel();  model.setRowCount(0);  try {  List<HoaDon> list = hdDAO.selectAll(); // Đọc dữ liệu từ CSDL  for (HoaDon hd : list) {  String trangThai;  if (hd.isTrangThai()) {  trangThai = "Da Hoan Thanh";  } else {  trangThai = "Chua Hoan Thanh";  }  Object[] data = {  hd.getMaHD(),  hd.getMaNV(),  hd.getMaKH(),  hd.getNgayBan(),  hd.getMaKM(),  hd.getThanhTien(),  trangThai  };  listHD.add(hd);  model.addRow(data);  }  } catch (Exception e) {  }  }  boolean buttonNew = false;  boolean buttonUpdate = false;  boolean buttonSave = false;  boolean status = false;  private void statusBtn(boolean b) {  if (buttonNew == true && b == true) {  btnUpdate.setEnabled(false);  btnDelete.setEnabled(false);  btnSave.setEnabled(b);  } else if (buttonUpdate == true && b == true) {  btnNew.setEnabled(false);  btnDelete.setEnabled(false);  btnSave.setEnabled(b);  }  }  private void cancel() {  setStatus(false);  buttonSave = false;  buttonNew = false;  buttonSave = false;  btnNew.setEnabled(true);  btnUpdate.setEnabled(true);  btnSave.setEnabled(false);  btnDelete.setEnabled(true);  }  private void resetFormHD() {  int i = tblhoadon.getSelectedRow();  txtMaHD.setText("");  lblThanhTien.setText("");  cboMaKH.setSelectedIndex(0);  cboMaKM.setSelectedIndex(0);  cboMaNV.setSelectedIndex(0);  if (i >= 0) {  tblhoadon.removeRowSelectionInterval(i, i);  }  }  public void setStatus(boolean b) {  if (status == b) {  txtMaHD.setEditable(b);  cboMaKH.setVisible(b);  cboMaKM.setVisible(b);  cboMaNV.setVisible(b);  } else {  if (buttonNew == b) {  txtMaHD.setEditable(b);  } else if (buttonUpdate == true) {  txtMaHD.setEditable(false);  }  cboMaKH.setVisible(b);  cboMaKM.setVisible(b);  cboMaNV.setVisible(b);  }  }  HoaDon getForm() {  HoaDon hd = new HoaDon();  hd.setMaHD(txtMaHD.getText());  hd.setMaNV(String.valueOf(cboMaNV.getSelectedItem().toString()));  hd.setMaKH(String.valueOf(cboMaKH.getSelectedItem().toString()));  hd.setMaKM(String.valueOf(cboMaKM.getSelectedItem().toString()));  hd.setThanhTien(0);  hd.setTrangThai(false);  hd.setXoa(true);  return hd;  }  public void setForm(int index) {  txtMaHD.setText(tblhoadon.getValueAt(index, 0).toString());  lblMaHD.setText(tblhoadon.getValueAt(index, 0).toString());  lblMaNV.setText(tblhoadon.getValueAt(index, 1).toString());  lblMaKH.setText(tblhoadon.getValueAt(index, 2).toString());  lblMaKM.setText(tblhoadon.getValueAt(index, 4).toString());  lblThanhTien.setText(tblhoadon.getValueAt(index, 5).toString());  }  ArrayList<HoaDon> listHD = new ArrayList<>();  public boolean check() {  boolean check = true;  for (int i = 0; i < listHD.size(); i++) {  if (listHD.get(i).getMaHD().equalsIgnoreCase(txtMaHD.getText())) {  check = false;  }  }  if (txtMaHD.getText().equals("")) {  MsgBox.alert(this, "Khong Bo Trong MA HOA DON");  return false;  }  if (check == false) {  MsgBox.alert(this, "MA HOA DON");  return false;  }  return true;  }  private void insert() {  HoaDon hd = getForm();  hdDAO.insert(hd);  hdDAO.updateNgayBan(txtMaHD.getText());  this.fillTable();  }  public void delete() {  String mahd = txtMaHD.getText();  if (MsgBox.confirm(this, "Bạn thực sự muốn Hoa Don này")) {  try {  // hdctDAO.deletebByMaHD(mahd);  // hdDAO.delete(mahd);  hdDAO.hide(mahd);  MsgBox.alert(this, "Xóa thành công");  if (listHDCT.isEmpty()) {  hdDAO.updateTrangThai2(mahd);  }  this.fillTable();  } catch (Exception e) {  MsgBox.alert(this, "Xóa thất bại");  }  }  }  public void update() {  HoaDon hd = getForm();  try {  hdDAO.update(hd);  //hdDAO.updateNgayBan(txtMaHD.getText());  hdDAO.updateThanhTien(txtMaHD.getText());  this.fillTable();  MsgBox.alert(this, "Cập nhật thành công");  cancel();  } catch (Exception e) {  MsgBox.alert(this, "Cập nhật thất bại");  }  }  private void save() {  if (buttonNew == true && check()) {  insert();  MsgBox.alert(this, "them thanh cong");  cancel();  }  if (buttonUpdate == true) {  int index = tblhoadon.getSelectedRow();  update();  setForm(index);  tblhoadon.setRowSelectionInterval(index, index);  }  }  public void thanhTien() {  double sl = Double.parseDouble(cboSL.getSelectedItem().toString());  double gia = 0;  try {  gia = Double.parseDouble(txtDonGia.getText());  } catch (Exception e) {  }  txtThanhTien.setText(String.valueOf(sl \* gia));  }  boolean checkFind = true;  public void checkFind() {  for (int i = 0; i < listHD.size(); i++) {  if (listHD.get(i).getMaHD().equalsIgnoreCase(txtTimKiem.getText())) {  checkFind = true;  return;  }  }  for (int i = 0; i < listHD.size(); i++) {  if (listHD.get(i).getMaNV().equalsIgnoreCase(txtTimKiem.getText())) {  checkFind = false;  return;  }  }  }  public void timKiem() {  checkFind();  if (checkFind) {  find();  checkFind = true;  } else if (!checkFind) {  find2();  checkFind = true;  }  }  public void find() {  DefaultTableModel model = (DefaultTableModel) tblhoadon.getModel();  model.setRowCount(0);  try {  String key = txtTimKiem.getText();  List<HoaDon> list = hdDAO.selectByKeyword(key);  for (HoaDon hd : list) {  String trangThai;  if (hd.isTrangThai()) {  trangThai = "Da Hoan Thanh";  } else {  trangThai = "Chua Hoan Thanh";  }  Object[] data = {  hd.getMaHD(),  hd.getMaNV(),  hd.getMaKH(),  hd.getNgayBan(),  hd.getMaKM(),  hd.getThanhTien(),  trangThai  };  model.addRow(data);  }  } catch (Exception e) {  MsgBox.alert(this, e.getMessage());  }  }  public void find2() {  DefaultTableModel model = (DefaultTableModel) tblhoadon.getModel();  model.setRowCount(0);  try {  String key = txtTimKiem.getText();  List<HoaDon> list = hdDAO.selectByKeyword2(key);  for (HoaDon hd : list) {  String trangThai;  if (hd.isTrangThai()) {  trangThai = "Da Hoan Thanh";  } else {  trangThai = "Chua Hoan Thanh";  }  Object[] data = {  hd.getMaHD(),  hd.getMaNV(),  hd.getMaKH(),  hd.getNgayBan(),  hd.getMaKM(),  hd.getThanhTien(),  trangThai  };  model.addRow(data);  }  } catch (Exception e) {  MsgBox.alert(this, e.getMessage());  }  } |

### Xử lý Form Chi tiết hóa đơn



Mô tả công việc

|  |  |
| --- | --- |
| Sự kiện | Công việc phải làm |
| Constructor | this.init();  int i = 0; |
| tblHDCTaMouseClicked | int i = tblHDCTa.getSelectedRow();  setFormHDCT\_Click(i); |
| cboMaCT | fill();  thanhTien(); |
| cboSL | thanhTien(); |
| btnNewHDCT | buttonSaveHDCT = true;  buttonNewHDCT = true;  statusBtnHDCT(buttonSaveHDCT); |
| btnEditHDCT | buttonSaveHDCT = true;  buttonUpdateHDCT = true;  statusBtnHDCT(buttonSaveHDCT); |
| btnDeleteHDCT | int i = tblHDCTa.getSelectedRow();  if (i < 0) {  MsgBox.alert(this, "Chua chon san pham de xoa");  } else  deleteHDCT(); |
| btnSaveHDCT | saveHDCT(); |
| btnCancelHDCT | cancelHDCT(); |
| btnIN | xuatHoaDon(); |

Hướng dẫn cài đặt mã nguồn

|  |
| --- |
| HoaDonDAO hdDAO = new HoaDonDAO();  ChiTietSanPhamDAO spctDAO = new ChiTietSanPhamDAO();  HoaDonChiTietDAO hdctDAO = new HoaDonChiTietDAO();  SanPhamDAO spDAO = new SanPhamDAO();  NhanVienDAO nvDAO = new NhanVienDAO();  KhachHangDAO khDAO = new KhachHangDAO();  KhuyenMaiDAO kmDAO = new KhuyenMaiDAO();  Date now = new Date();  SimpleDateFormat ngay = new SimpleDateFormat("dd-MM-yyyy -- hh:mm:ss a");  String a = ngay.format(now);  String chuoi = "";  String patternPass = "^[a-zA-Z0-9]{6,15}$";  String tenNV;  String tenKH;  float khuyenMai;  private void init() {  fillCboNV();  fillCboKH();  fillCboKM();  initTable();  initTableHDCT();  fillTable();  fillTableHDCT();  fillCboMaCT();  btnSave.setEnabled(false);  btnSaveHDCT.setEnabled(false);  txtDonGia.setEditable(false);  txtThanhTien.setEditable(false);  setStatus(false);  setStatusHDCT(false);  thanhTien();  fill();  }  private void fillCboMaCT() {  DefaultComboBoxModel model = (DefaultComboBoxModel) cboMaCT.getModel();  model.removeAllElements();  List<String> list = hdctDAO.selectMaCT();  for (String nv : list) {  model.addElement(nv);  }  }  private void initTableHDCT() {  DefaultTableModel model = (DefaultTableModel) tblHDCTa.getModel();  String[] cols = new String[]{"#", "Ma CT", "Ten SP", "So Luong", "Don Gia", "Thanh Tien"};  model.setColumnIdentifiers(cols);  }  public void fillTableHDCT() {  DefaultTableModel model = (DefaultTableModel) tblHDCTa.getModel();  model.setRowCount(0);  listHDCT.removeAll(listHDCT);  try {  String id = lblMaHD.getText();  List<HoaDonChiTiet> list = hdctDAO.selectByMaHD(id);  for (HoaDonChiTiet hdct : list) {  Object[] data = {  hdct.getMaHDCT(),  hdct.getMaCT(),  hdct.getTenSP(),  hdct.getSoLuong(),  hdct.getGia(),  hdct.getTongTien()  };  listHDCT.add(hdct);  model.addRow(data);  }  } catch (Exception e) {  }  }  boolean buttonNewHDCT = false;  boolean buttonUpdateHDCT = false;  boolean buttonSaveHDCT = false;  boolean statusHDCT = false;  private void statusBtnHDCT(boolean b) {  if (buttonNewHDCT == true && b == true) {  btnEditHDCT.setEnabled(false);  btnDeleteHDCT.setEnabled(false);  btnSaveHDCT.setEnabled(b);  cboMaCT.setEnabled(true);  cboSL.setEnabled(true);  } else if (buttonUpdateHDCT == true && b == true) {  btnNewHDCT.setEnabled(false);  btnDeleteHDCT.setEnabled(false);  btnSaveHDCT.setEnabled(b);  cboMaCT.setEnabled(true);  cboSL.setEnabled(true);  }  }  private void cancelHDCT() {  setStatus(false);  buttonSaveHDCT = false;  buttonNewHDCT = false;  buttonSaveHDCT = false;  btnNewHDCT.setEnabled(true);  btnEditHDCT.setEnabled(true);  btnSaveHDCT.setEnabled(false);  btnDeleteHDCT.setEnabled(true);  cboMaCT.setEnabled(false);  cboSL.setEnabled(false);  }  public void setStatusHDCT(boolean b) {  int i = tblhoadon.getSelectedRow();  if (i < 0 || b == false) {  lblThongBao.setText("CHUA CHON HOA DON");  lblThongBao.setVisible(!b);  lblMaHD.setText("");  lblSize.setText("");  lblMau.setText("");  txtDonGia.setText("");  txtThanhTien.setText("");  //tblHDCTa.setEnabled(b);  cboMaCT.setEnabled(b);  cboSL.setEnabled(b);  btnEditHDCT.setEnabled(b);  btnDeleteHDCT.setEnabled(b);  btnSaveHDCT.setEnabled(b);  btnNewHDCT.setEnabled(b);  btnCancelHDCT.setEnabled(b);  return;  } else {  lblThongBao.setVisible(!b);  //tblHDCTa.setEnabled(!b);  cboMaCT.setEnabled(!b);  cboSL.setEnabled(!b);  //setFormHDCT\_Click(i);  btnEditHDCT.setEnabled(b);  btnDeleteHDCT.setEnabled(b);  btnSaveHDCT.setEnabled(b);  btnNewHDCT.setEnabled(b);  btnCancelHDCT.setEnabled(b);  fillTableHDCT();  }  }  HoaDonChiTiet getFormHDCT(int i) {  HoaDonChiTiet hdct = new HoaDonChiTiet();  if (buttonUpdateHDCT == true) {  hdct.setMaHDCT(Integer.parseInt(tblHDCTa.getValueAt(i, 0).toString()));  }  hdct.setMaHD(lblMaHD.getText());  hdct.setMaCT(String.valueOf(cboMaCT.getSelectedItem().toString()));  hdct.setSoLuong(Integer.parseInt(cboSL.getSelectedItem().toString()));  hdct.setTenSP(lblTenSP.getText());  hdct.setGia(Float.parseFloat(txtDonGia.getText()));  hdct.setTongTien(Float.parseFloat(txtThanhTien.getText()));  return hdct;  }  public void setFormHDCT\_Click(int index) {  cboMaCT.setSelectedItem(tblHDCTa.getValueAt(index, 1).toString());  lblTenSP.setText(tblHDCTa.getValueAt(index, 2).toString());  cboSL.setSelectedItem(tblHDCTa.getValueAt(index, 3).toString());  txtDonGia.setText(tblHDCTa.getValueAt(index, 4).toString());  txtThanhTien.setText(tblHDCTa.getValueAt(index, 5).toString());  }  ArrayList<HoaDonChiTiet> listHDCT = new ArrayList<>();  private void insertHDCT() {  HoaDonChiTiet hdct = getFormHDCT(0);  hdctDAO.insert(hdct);  hdDAO.updateThanhTien(lblMaHD.getText());  hdDAO.updateTrangThai1(lblMaHD.getText());  fillTable();  this.fillTableHDCT();  }  public void deleteHDCT() {  int i = tblHDCTa.getSelectedRow();  String mahdct = tblHDCTa.getValueAt(i, 0).toString();  if (MsgBox.confirm(this, "Bạn thực sự muốn sp này")) {  try {  hdctDAO.delete(mahdct);  hdDAO.updateThanhTien(lblMaHD.getText());  this.fillTable();  MsgBox.alert(this, "Xóa thành công");  fillTableHDCT();  } catch (Exception e) {  MsgBox.alert(this, "Xóa thất bại");  }  }  }  public void updateHDCT() {  int i = tblHDCTa.getSelectedRow();  HoaDonChiTiet hdct = getFormHDCT(i);  try {  hdctDAO.update(hdct);  fillTableHDCT();  MsgBox.alert(this, "Cập nhật thành công");  cancelHDCT();  } catch (Exception e) {  MsgBox.alert(this, "" + e);  }  }  private void saveHDCT() {  if (buttonNewHDCT == true) {  insertHDCT();  cancelHDCT();  MsgBox.alert(this, "them thanh cong");  } else if (buttonUpdateHDCT == true) {  int index = tblHDCTa.getSelectedRow();  updateHDCT();  //setFormHDCT\_Click(index);  tblHDCTa.setRowSelectionInterval(index, index);  }  }  public void setTenSP(SanPham sp) {  try {  lblTenSP.setText(sp.getTenSP());  } catch (Exception e) {  }  }  public void setFormHDCT(ChiTietSanPham ct) {  try {  lblSize.setText(String.valueOf(ct.getSize()));  lblMau.setText(ct.getMauSac());  txtDonGia.setText(String.valueOf(ct.getGia()));  } catch (Exception e) {  }  }  public void fill() {  String maCT = (String) cboMaCT.getSelectedItem();  ChiTietSanPham ct = spctDAO.selectById(maCT);  try {  String ten = ct.getMaSP();  SanPham sp = spDAO.selectById(ten);  this.setTenSP(sp);  } catch (Exception e) {  }  this.setFormHDCT(ct);  }  public void thanhTien() {  double sl = Double.parseDouble(cboSL.getSelectedItem().toString());  double gia = 0;  try {  gia = Double.parseDouble(txtDonGia.getText());  } catch (Exception e) {  }  txtThanhTien.setText(String.valueOf(sl \* gia));  }  public void xuatHoaDon() {  try {  getTen();  XSSFWorkbook wordkbook = new XSSFWorkbook();  XSSFSheet sheet = wordkbook.createSheet("HoaDon");  XSSFRow row = null;  Cell cell = null;  row = sheet.createRow(0);  cell = row.createCell(3, CellType.STRING);  cell.setCellValue("PHIẾU HÓA ĐƠN");  row = sheet.createRow(1);  cell = row.createCell(5, CellType.STRING);  cell.setCellValue("Thời gian lập phiếu: " + a);  row = sheet.createRow(2);  cell = row.createCell(1, CellType.STRING);  cell.setCellValue("Mã hóa đơn: " + lblMaHD.getText());  row = sheet.createRow(3);  cell = row.createCell(1, CellType.STRING);  cell.setCellValue("Tên nhân viên: " + tenNV);  row = sheet.createRow(4);  cell = row.createCell(1, CellType.STRING);  cell.setCellValue("Tên khách hàng: " + tenKH);  row = sheet.createRow(6);  cell = row.createCell(0, CellType.STRING);  cell.setCellValue("STT");  cell = row.createCell(1, CellType.STRING);  cell.setCellValue("Mã hóa đơn chi tiết");  cell = row.createCell(2, CellType.STRING);  cell.setCellValue("Mã hóa đơn");  cell = row.createCell(3, CellType.STRING);  cell.setCellValue("Mã chi tiết");  cell = row.createCell(4, CellType.STRING);  cell.setCellValue("Tên sản phẩm");  cell = row.createCell(5, CellType.STRING);  cell.setCellValue("Số lượng");  cell = row.createCell(6, CellType.STRING);  cell.setCellValue("Giá");  cell = row.createCell(7, CellType.STRING);  cell.setCellValue("Tổng tiền");  row = sheet.createRow(listHDCT.size() + 7);  cell = row.createCell(7, CellType.STRING);  cell.setCellValue("Khuyến mãi: " + khuyenMai + " (%)");  row = sheet.createRow(listHDCT.size() + 8);  cell = row.createCell(7, CellType.STRING);  cell.setCellValue("Thành tiền" + lblThanhTien.getText());  for (int i = 0; i < listHDCT.size(); i++) {  //Modelbook book =arr.get(i);  row = sheet.createRow(7 + i);  cell = row.createCell(0, CellType.NUMERIC);  cell.setCellValue(i + 1);  cell = row.createCell(1, CellType.STRING);  cell.setCellValue(listHDCT.get(i).getMaHDCT());  cell = row.createCell(2, CellType.STRING);  cell.setCellValue(listHDCT.get(i).getMaHD());  cell = row.createCell(3, CellType.STRING);  cell.setCellValue(listHDCT.get(i).getMaCT());  cell = row.createCell(4, CellType.STRING);  cell.setCellValue(listHDCT.get(i).getTenSP());  cell = row.createCell(5, CellType.STRING);  cell.setCellValue(listHDCT.get(i).getSoLuong());  cell = row.createCell(6, CellType.STRING);  cell.setCellValue(listHDCT.get(i).getGia());  cell = row.createCell(7, CellType.STRING);  cell.setCellValue(listHDCT.get(i).getTongTien());  }  chuoi = JOptionPane.showInputDialog("Nhập tên File: ");  if (chuoi == null || chuoi.equals("")) {  MsgBox.alert(this, "Vui lòng nhập tên File");  return;  } else if (!chuoi.matches(patternPass) || chuoi.length() < 6 || chuoi.length() > 15) {  MsgBox.alert(this, "Tên File không được chứa ký tự đặt biệt và tên File từ 6-->15 ký tự");  return;  }  try {  File f = new File("src//com//nhom4//File//" + chuoi + ".xlsx");  FileOutputStream fis = new FileOutputStream(f);  wordkbook.write(fis);  fis.close();  } catch (FileNotFoundException ex) {  ex.printStackTrace();  } catch (IOException ex) {  ex.printStackTrace();  }  MsgBox.alert(this, "In thành công \n File của bạn đang ở: \n..\\Nhom4\_QL\_ShopBanGiay\\src\\com\\nhom4\\File\\" + chuoi);  } catch (Exception ex) {  ex.printStackTrace();  MsgBox.alert(this, "Lỗi mở File");  }  } |

### Xử lý Form Thống kê



Mô tả công việc

|  |  |
| --- | --- |
| Sự Kiện | Công việc phải làm |
| Constructor | init(); |
| cboThang | this.fillTableKhachHang(); |
| cboNam | this.fillTableKhachHang(); |
| cboNamDT | this.fillTableDoanhThu(); |

Hướng dẫn cài đặt mã nguồn

|  |
| --- |
| ThongKeDAO tkDAO = new ThongKeDAO();  HoaDonDAO hdDAO = new HoaDonDAO();  KhachHangDAO khDAO = new KhachHangDAO();  private void init() {  this.fillComboBoxThang();  this.fillComboBoxNam();  this.fillTableKhachHang();  this.fillComboBoxNamDT();  this.fillTableDoanhThu();  this.selectTab(0);  if (!Auth.isManager()) {  tabs.remove(1);  }  }  public void selectTab(int index) {  tabs.setSelectedIndex(index);  }  private void fillComboBoxThang() {  DefaultComboBoxModel model = (DefaultComboBoxModel) cboThang.getModel();  model.removeAllElements();  List<Integer> list = tkDAO.selectMonths();  for (Integer month : list) {  model.addElement(month);  }  }  private void fillComboBoxNam() {  DefaultComboBoxModel model = (DefaultComboBoxModel) cboNam.getModel();  model.removeAllElements();  List<Integer> list = tkDAO.selectYears();  for (Integer year : list) {  model.addElement(year);  }  }  private void fillComboBoxNamDT() {  DefaultComboBoxModel model = (DefaultComboBoxModel) cboNamDT.getModel();  model.removeAllElements();  List<Integer> list = tkDAO.selectYears();  for (Integer year : list) {  model.addElement(year);  }  }  private void fillTableKhachHang() {  DefaultTableModel model = (DefaultTableModel) tblBang.getModel();  model.setRowCount(0);  int thang=0 ;  int nam=0;  try {  thang = Integer.parseInt( cboThang.getSelectedItem().toString());  nam = Integer.parseInt( cboNam.getSelectedItem().toString());  } catch (Exception e) {  }  List<Object[]> list = tkDAO.getKhachHang(thang, nam);  for (Object[] row : list) {  model.addRow(row);  }  }  private void fillTableDoanhThu() {  DefaultTableModel model = (DefaultTableModel) tblDoanhThu.getModel();  model.setRowCount(0);  int nam = (Integer) cboNamDT.getSelectedItem();  List<Object[]> list = tkDAO.getDoanhThu(nam);  for (Object[] row : list) {  model.addRow(row);  }  } |

# Kiểm thử

## Kiểm thử form nhân viên

|  |  |  |  |
| --- | --- | --- | --- |
| **Thuộc tính** | **Thao tác** | | |
| **Thêm mới** | **Cập nhật** | **Xóa** |
| *Mã nhân viên* | *Không để trống*  *Không được nhập trùng*  *Nhập 5 đến 6 ký tự* |  | *Không xóa chính mình* |
| *Tên nhân viên* | *Không để trống*  *Chỉ chứa alphabet và ký tự trắng* | | |
| *Giới tính* | *Không được để trống* | | |
| *Số điện thoại* | *Không được bỏ trống*  *Không được nhập chữ*  *Không được nhập trùng số điện thoại* | | |
| Địa chỉ | Không được bỏ trống | | |
| Email | Không được bỏ trống  Nhập đúng định dạng email | | |
| Hình | Không để trống | | |

## Kiểm thử form khách hàng

|  |  |  |
| --- | --- | --- |
| **Thuộc tính** | **Thao tác** | |
| **Thêm mới** | **Cập nhật** |
| Mã khách hàng | Không để trống  Không trùng  Nhập 5 đến 6 ký tự |  |
| Tên khách hàng | Không được để trống  Chỉ chứa alphabet và ký tự trắng | |
| Số điện thoại | Không được bỏ trống  Không được nhập chữ  Không được nhập trùng số điện thoại | |
| Địa chỉ | Không được bỏ trống | |

## Kiểm thử form nhà cung cấp

|  |  |  |  |
| --- | --- | --- | --- |
| **Thuộc tính** | **Thao tác** | | |
| **Thêm mới** | **Cập nhật** | **Xóa** |
| Mã nhà cung cấp | Không để trống  Không trùng  Nhập 3 đến 7 ký tự | Khóa mã nhà cung cấp không cho nhập | Không được xóa nhà cũng cấp đã cung cấp sản phẩm |
| Tên nhà cung cấp | Không được để trống | |

## Kiểm thử form sản phẩm

|  |  |  |
| --- | --- | --- |
| **Thuộc tính** | **Thao tác** | |
| **Thêm mới** | **Cập nhật** |
| Mã sản phẩm | Không để trống  Không trùng  Nhập 5 đến 6 ký tự |  |
| Tên sản phẩm | Không được để trống  Chỉ chứa alphabet và ký tự trắng | |
| Số lượng | Không được để trống | |
| Mã nhà cung cấp | Số dương (>=0)  Không được nhập chữ | |
| Ghi chú | … | |

## Kiểm thử form khuyến mãi

|  |  |  |  |
| --- | --- | --- | --- |
| **Thuộc tính** | **Thao tác** | | |
| **Thêm mới** | **Cập nhật** | **Xóa** |
| Mã khuyến mãi | Không để trống  Không trùng | Khóa mã khuyến mãi lại không cho nhập | Không thể xóa khi đã áp dụng cho một hóa đơn bất kỳ |
| Tên khuyến mãi | Không được để trống | |
| Ngày bắt đầu | Ngày bắt đầu phải lớn hơn thời gian hiện tại | Chương trình đang diễn ra thì không thể cập nhật |
| Ngày kết thúc | Thời gian kết thúc phải lớn hơn thời gian bắt đầu | Chương trình đang diễn ra thì không thể cập nhật |
| Giảm giá | Không được bỏ trống  Không được nhập chữ  Số dương (từ 0% đến 100%) | |
| Ghi chú | ….. | |

## Kiểm thử form tài khoản quản lý

|  |  |  |  |
| --- | --- | --- | --- |
| **Thuộc tính** | **Thao tác** | | |
| **Lưu** | **Cập nhật** | **Xóa** |
| Mã nhân viên | Không để trống  Không nhập trùng mã đã tồn tại | | Không được xóa chính mình |
| Tên đăng nhập | Không để trống  Không chứa kí tự đặc biệt, từ 6 đến 15 kí tự | |
| Mật khẩu | Không để trống  Không chứa kí tự đặc biệt, từ 6 đến 15 kí tự | |
| Vai trò | Chọn 1 vai trò  Không được chọn vai trò là quản lý | |

## Kiểm thử form tài khoản cá nhân

|  |  |
| --- | --- |
| **Thuộc tính** | **Thao tác** |
| **Lưu** |
| Tên nhân viên | Không để trống  Chỉ chứa alphabet và ký tự trắng |
| Giới tính | Chọn ít nhất 1 giới tính |
| Số điện thoại | Không được bỏ trống  Không được nhập chữ và nhập đủ 10 số  Không được nhập trùng số điện thoại |
| Địa chỉ | Không được bỏ trống |
| Email | Không được bỏ trống  Nhập đúng định dạng email |

# Đóng gói và triển khai

## Hướng dẫn chuyển đổi jar thành exe

✓ Sử dụng phần mềm EXE4J để chuyển đổi jar sang exe

✓ Sử dụng phần mềm InnoSetup để đóng gói

## Hướng dẫn cài đặt triển khai

✓Cài SQL Server 2008 trở lên

✓ Tạo CSDL QL\_ShopBanGiay bằng cách chạy file QL\_ShopBanGiay.sql

✓ Cài đặt JDK 1.8 trở lên

✓ Chạy file setup

## Hướng dẫn sử dụng phần mềm

✓ Đăng nhập với tài khoản username: nguyenvana và password: 1234

✓ Cấp các tài khoản có vai trò là Quản lý