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



|  |
| --- |
| THÀNH PHỐ HỒ CHÍ MINH 2022 |

|  |
| --- |
| GIẢNG VIÊN HƯỚNG DẪN: LÊ VĂN PHỤNG |
| SINH VIÊN THỰC HIỆN |
| 1. Đào Hồng Đăng \_ PS17490\_Trưởng nhóm 2. Mai Thụy Ngọc Hân\_PS17521 |

MỤC LỤC

[1 Phân tích 2](#_Toc101366723)

[1.1 Hiện trạng 2](#_Toc101366724)

[1.2 Yêu cầu hệ thống 2](#_Toc101366725)

[1.3 Use case 3](#_Toc101366726)

[2 Thiết kế 5](#_Toc101366727)

[2.1 Mô hình triển khai 5](#_Toc101366728)

[2.2 Thiết kế CSDL 5](#_Toc101366729)

[2.2.1 Sơ đồ quan hệ thực thể 5](#_Toc101366730)

[2.2.2 Thiết kế chi tiết các thực thể 6](#_Toc101366731)

[2.3 Thiết kế giao diện 9](#_Toc101366732)

[2.3.1 Sơ đồ tổ chức giao diện 9](#_Toc101366733)

[2.3.2 Thiết kế giao diện cho các chức năng nghiệp vụ 9](#_Toc101366734)

[3 Thực hiện viết mã 24](#_Toc101366735)

[3.1 Viết mã tạo CSDL 24](#_Toc101366736)

[3.1.1 Tạo CSDL 24](#_Toc101366737)

[3.1.2 SQL truy vấn và thao tác 25](#_Toc101366738)

[3.1.3 Các thủ tục lưu tổng hợp thống kê 27](#_Toc101366739)

[3.2 Lập trình JDBC 28](#_Toc101366740)

[3.2.1 Entity Class- Các lớp truy xuất dữ liệu 29](#_Toc101366741)

[3.2.2 DAO Class - Các lớp truy xuất dữ liệu 45](#_Toc101366742)

[3.3 Viết mã cho ứng dụng 72](#_Toc101366743)

[3.3.1 Đăng nhập (DangNhap) 72](#_Toc101366744)

[3.3.2 Màn hình chào (Loading) 73](#_Toc101366745)

[3.3.3 Giao diện( QL, NV) 73](#_Toc101366746)

[3.3.4 Bán hàng (BanHang) 74](#_Toc101366747)

[3.3.5 Quản lý nhân viên (QuanLyNhanVien) 78](#_Toc101366748)

[3.3.6 Quản lý tài khoản (TK) 82](#_Toc101366749)

[3.3.7 Quản lý sản phẩm(QuanLySanPham) 85](#_Toc101366750)

[3.3.8 Quản lý khách hàng (KH) 89](#_Toc101366751)

[3.3.9 Quản lý hóa đơn(QuanLyHoaDon) 92](#_Toc101366752)

[3.3.10 Thống kê doanh thu(ThongKeDoanhThu) 95](#_Toc101366753)

[4 Kiểm thử 99](#_Toc101366754)

[4.1 Kiểm thử form Đăng nhập 99](#_Toc101366755)

[4.2 Kiểm thử form nhân viên 99](#_Toc101366756)

[4.3 Kiểm thử form khách hàng 99](#_Toc101366757)

[4.4 Kiểm thử form sản phẩm 99](#_Toc101366758)

[4.5 Kiểm thử form tài khoản 99](#_Toc101366759)

# Phân tích

## Hiện trạng

Hiện nay, công nghệ thông tin được xem là một ngành mũi nhọn của quốc gia, đặc biệt là các nước đang phát triển, tiến hành công nghiệp hóa hiện đại hóa như nước ta. Sự bùng nổ thông tin và sự phát triển mạnh mẽ của công nghệ kỹ thuật số, muốn phát triển thì phải áp dụng tin học hóa vào tất cả các ngành các lĩnh vực. Cùng với sự phát triển nhanh chóng về phần cứng máy tính, các phần mềm càng trở nên đa dạng, phong phú, hoàn thiện hơn và hỗ trợ hiệu quả cho con người. Các phần mềm hiện nay ngày càng hỗ trợ cho người dùng thuận tiện sử dụng, thời gian xử lý nhanh chóng, và một số nghiệp vụ được tự động hóa cao. Do vậy mà trong việc phát triển phần mềm, sự đòi hỏi không chỉ là sự chính xác, xử lý được nhiều nghiệp vụ thực tế mà còn phải đáp ứng yêu cầu khác như về tốc độ, giao diện thân thiện, mô hình hóa được thực tế vào máy tính để người dùng sử dụng tiện lợi, quen thuộc, tính tương thích cao, bảo mật cao, … .Các phần mềm giúp tiết kiệm một lượng lớn thời gian, công sức của con người, tăng độ chính xác và hiệu quả trong công việc. Cụ thể, việc quản lý quán cà phê. Nếu không có sự hỗ trợ của tin học, việc quản lý phải cần khá nhiều người, chia thành nhiều khâu mới có thể quản lý quán cà phê như: Quản lý nhân viên, quản lý hóa đơn thanh toán,… Các công việc này đòi hỏi nhiều thời gian và công sức, mà sự chính xác và hiệu quả không cao, nếu làm bằng thủ công không mang tính tự động. Một số nghiệp vụ như tra cứu, thống kê và hiệu chỉnh thông tin khá vất vả. Ngoài ra còn có một số khó khăn về việc lưu trữ khá đồ sộ, dễ bị thất lạc, tốn kém,…. Trong khi đó các nghiệp vụ này có thể tin học hóa một cách đơn giản hơn.

## Yêu cầu hệ thống

Phần mềm Quản Lý Quán Coffee của nhóm em mong muốn:

* Yêu cầu chức năng nghiệp vụ:
* Quản lý khách hàng
* Quản lý nhân viên
* Quản lý menu
* Quản lý danh mục
* Quản lý hóa đơn
* Thống kê doanh thu
* Yêu cầu về bảo mật:
* Tất cả mọi thành viên phải đăng nhập mới sử dụng được phần mềm
* Admin được phép thực hiện tất cả các chức năng
* Nhân viên không được phép quản lý tài khoản, thông tin nhân viên và xóa dữ liệu trong các chức năng quản lý
* Yêu cầu về môi trường công nghệ:
* Phần mềm phải phù hợp với môi trường hiện tại
* Hệ quản trị CSDL SQL Server 2008 trở lên

## Use case

Use case là sơ đồ tổng quan về mặt chức năng và phân vai trò người sử dụng. Dựa vào yêu cầu hệ thống của khách hàng, chúng ta có thể phác thảo sơ đồ use case như sau:

Diagram

Description automatically generated

* Chi tiết các chức năng:
* Mỗi chức năng quản lý bao gồm các chức năng con:
* Xem: xem tất cả và chi tiết một mục
* Thêm: thêm mới vào cơ sở dữ liệu
* Xóa: xóa khỏi cơ sở dữ liệu
* Sửa: cập nhật dữ liệu hiện có
* Tìm kiếm: tìm kiếm theo điều kiện
* Làm mới: làm mới form để thêm dữ liệu
* Các chức năng thao tác dữ liệu(thêm, sửa, xóa) cần được kiểm lỗi hợp lý với dữ liệu
* Chức năng tổng hợp thống kê số liệu:
* Thống kê hóa đơn có cấu trúc:
* Số lượng hóa đơn
* Tổng doanh thu
* Chức năng đăng nhập: Đăng nhập để sử dụng phần mềm, bắt buộc phải đăng nhập
* Chi tiết về yêu cầu bảo mật:
* Tất cả phải đăng nhập mới được sử dụng các chức năng trong phần mềm
* Admin: được phép sử dụng các chức năng
* Nhân viên: Không được phép quản lý tài khoản, thông tin nhân viên
* Thông tin các thực thể:
* Nhân viên:
* Mã nhân viên
* Họ và tên
* Phụ cấp
* Chức vụ
* Lương cơ bản
* Khách hàng:
* Mã khách hàng
* Tên khách hàng
* Địa chỉ
* Sản phẩm:
* Mã sản phẩm
* Tên sản phẩm
* Đơn giá
* Số lượng
* Hóa đơn:
* Mã hóa đơn
* Mã nhân viên
* Số điện thoại
* Tên khách hàng
* Ngày lập hóa đơn
* Địa chỉ
* Chi tiết hóa đơn:
* Mã hóa đơn
* Mã sản phẩm
* Số lượng bán
* Thống kê:
* Ngày lập
* Tổng doanh thu
* Tài khoản:
* Mã nhân viên
* Usename
* Pass
* Quyền truy cập

# Thiết kế

## Mô hình triển khai

Ứng dụng phần mềm được xây dựng để phục vụ cho nhiều người dùng nhưng cơ sở dữ liệu thì lưu trữ tập trung

Diagram

Description automatically generated

* Cần một máy cài SQL Server 2008+. Máy này cần hệ điều hành window xp trở lên
* Các máy nhân viên phòng đào tạo cài phần mềm edusys. Các máy nhân viên cần JDK 1.8+ với hệ điều hành bất kỳ

## Thiết kế CSDL

### Sơ đồ quan hệ thực thể

Trong hệ thống đã được mô tả trên chúng ta thấy có các thực thể sau

* Hoạt động bán : cung cấp thông tin sản phẩm đã bán
* Khách hàng : cung cấp thông tin khách hàng
* Nhân viên: là người quản lý các sản phẩm được bán

Sau khi phân tích chúng ta có thể thấy các thực thể có mối quan hệ với nhau như sơ đồ SQL sau:

Diagram

Description automatically generated

### Thiết kế chi tiết các thực thể

**NHÂN VIÊN**

|  |  |  |  |
| --- | --- | --- | --- |
| **Column Name** | **Data type** | **Allow null** |  |
| MaNV | nchar(20) | NOT NULL | Mã nhân viên |
| TenNV | nvarchar(30) | NOT NULL | Tên nhân viên |
| chucVu | nvarchar(20) | NOT NULL | Chức vụ |
| LuongCoBan | float | NOT NULL | Lương cơ bản |
| PhuCap | float | NOT NULL | Phụ cấp |

**KHÁCH HÀNG**

|  |  |  |  |
| --- | --- | --- | --- |
| **Column Name** | **Data type** | **Allow null** |  |
| MaKH | nchar(20) | NOT NULL | Mã khách hàng |
| TenKH | nvarchar(30) | NOT NULL | Tên khách hàng |
| DiaChi | nvarchar(50) | NOT NULL | Địa chỉ |

**TÀI KHOẢN**

|  |  |  |  |
| --- | --- | --- | --- |
| **Column Name** | **Data type** | **Allow null** |  |
| MaNV | nchar(20) | NOT NULL | Mã nhân viên |
| username | nchar(20) | NOT NULL | Tên đăng nhập |
| pass | nchar(20) | NOT NULL | pass |
| quyenTruyCap | Nvarchar(15) | NOT NULL | Quyền truy cập |

**SẢN PHẨM**

|  |  |  |  |
| --- | --- | --- | --- |
| **Column Name** | **Data type** | **Allow null** |  |
| MaSP | nchar(20) | NOT NULL | Mã sản phẩm |
| TenSP | nvarchar(30) | NOT NULL | Tên sản phẩm |
| DonGia | float | NOT NULL | Đơn giá |
| soLuong | Int | NOT NULL | Số lượng |

**HÓA ĐƠN**

|  |  |  |  |
| --- | --- | --- | --- |
| **Column Name** | **Data type** | **Allow null** |  |
| MaHD | nchar(15) | NOT NULL | Mã hóa đơn |
| MaNV | nchar(20) | NOT NULL | Mã nhân viên |
| SoDT | nchar(15) | NOT NULL | Số điện thoại |
| TenKH | nvarchar(50) | NOT NULL | Tên khách hàng |
| NgayLap | Date | NOT NULL | Ngày lập |
| DiaChi | nvarchar(30) | NOT NULL | Địa chỉ |

**CHI TIẾT HÓA ĐƠN**

|  |  |  |  |
| --- | --- | --- | --- |
| **Column Name** | **Data type** | **Allow null** |  |
| MaHD | nchar(15) | NOT NULL | Mã hóa đơn |
| maSP | nchar(20) | NOT NULL | Mã sản phẩm |
| SoLuongBan | int | NOT NULL | Số lượng bán |

**THỐNG KÊ**

|  |  |  |  |
| --- | --- | --- | --- |
| **Column Name** | **Data type** | **Allow null** |  |
| ngayLap | date | NOT NULL | Ngày lập |
| TongDoanhThu | float | NOT NULL | Tổng doanh thu |

## Thiết kế giao diện

### Sơ đồ tổ chức giao diện

Giao diện chính là một cửa sổ chứa menu chính và toolbar. Thông qua đó để đi đến các giao diện thành viên để thực hiện các chức năng trong hệ thống.

Theo yêu cầu thì mọi nhân viên phải đăng nhập trước khi sử dụng ứng dụng nên form đăng nhập xuất hiện trước để yêu cầu đăng nhập.

Ngoài ra mỗi ứng dụng trong thời gian khởi động cần có một màn hình chào cùng với thanh tiến trình để người có cảm giác ứng dụng đang khởi động.

Diagram

Description automatically generated

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

Trong phần thiết kế giao diện các bạn nên sử dụng gói hình ảnh được cung cấp sẵn trên tài nguyên học tập của LMS. Nếu các bạn không muốn sử dụng gói hình ảnh này thì bạn có thể chọn gói hình ảnh phù hợp của các bạn.

Lưu ý quan trọng:

* Các bạn phải đặt tên đúng như hướng dẫn vì các tên này ảnh hưởng đến việc viết mã sau này. Nếu các bạn đặt tên khác thì phải chỉnh mã sau này theo tên mà các bạn đã đặt.
* Trong lập trình giao diện người ta thường qui ước tiếp đầu ngữ của tên các thành phần giao diện như sau:

|  |  |  |
| --- | --- | --- |
| Control | Prelix | Mô tả |
| Jlabel | lbl | Nhãn |
| Jbutton | Btn | Nút |
| JTextField,JTextArea,JPassword,JTextPane | txt | Các loại ô nhập |
| JRadioButton | rdo | Hộp chọn |
| JCheckBox | chk | Hộp kiểm |
| ButtonGroup | bgr | Nhóm các button |
| JComboBox | cbo | Hộp chọn xổ xuống |
| JPanel | pnl | Khung chứa |
| JTabbedPane | tab | Khung chứa bảng |
| JTable | tbl | Bảng |
| … |  |  |

#### Thiết kế màn hình chào

Graphical user interface, application, website

Description automatically generated

#### Thiết kế màn Form đăng nhập

Graphical user interface, application

Description automatically generated

Các thành phần giao diện

|  |  |  |  |
| --- | --- | --- | --- |
|  | Componel | Kiểu | Thuộc tính |
| 1 | DangNhap | JFrame |  |
| 2 | JLabel | JLabel | Text: Tên đăng Nhập |
| 3 | txtTenDangNhap | JTextField |  |
| 4 | JLabel | JLabel | Text: Mật khẩu |
| 5 | txtMatKhau | JPassword |  |
| 6 | JLabel | JLabel | Text: Quyền truy cập |
| 7 | cbQuyenTruyCap | JComboBox |  |
| 8 | chkHienMK | JCheckBox | Text: Hiện mật khẩu |
| 9 | btnDangNhap | JButton | Text:Đăng nhập |
| 10 | btnThoat | JButton | Text:Thoát |

#### Thiết kế giao diện chính

Graphical user interface, application, Teams

Description automatically generated

Graphical user interface, application, Teams

Description automatically generated

#### Thiết kế Form quản lý bán hàng

Graphical user interface, application

Description automatically generated

|  |  |  |  |
| --- | --- | --- | --- |
|  | Componel | Kiểu | Thuộc tính |
| 1 | BanHang | JFrame |  |
| 2 | JLabel11 | JLabel | Text: mã hóa đơn |
| 3 | txtMaHoaDon | JTextField |  |
| 4 | Jlabel3 | JLabel | Text: Mã nhân viên |
| 5 | txtMaNV | JTextField |  |
| 6 | JLabel5 | JLabel | Text: Ngày lập |
| 7 | dateNgayLap | JDateChooser |  |
| 8 | JLabel3 | JLabel | Text: Tên khách hàng |
| 9 | txtTenKhachHang | JTextField |  |
| 10 | JLabel2 | JLabel | Text: Số điện thoại |
| 11 | txtSoDT | JTextField |  |
| 12 | JLabel4 | JLabel | Text: Địa chỉ |
| 13 | txtDiaChi | JTextField |  |
| 14 | JLabel13 | JLabel | Text: Mã sản phẩm |
| 15 | txtMaSP | JTextField |  |
| 16 | JLabel7 | JLabel | Text: tên sản phẩm |
| 17 | cbTenSP | JComboBox |  |
| 18 | JLabel8 | JLabel | Text: Đơn giá |
| 19 | txtDonGia | JTextField |  |
| 20 | JLabel9 | JLabel | Text: Số lượng bán |
| 21 | txtSoLuongBan | JTextField |  |
| 22 | JLabel6 | JLabel | Text: Mã hóa đơn |
| 23 | cbmaHD | JComboBox |  |
| 24 | JLabel12 | JLabel | Text: Tổng tiền hóa đơn |
| 25 | txtTongTien | JTextField |  |
| 26 | tblDanhSach | JTable |  |
| 27 | btnReset | JButton | Text: Reset |
| 28 | btnThem | JButton | Text: Thêm hàng |
| 29 | btnSapXep | Jbutton | Text: Sắp xếp theo giá |
| 30 | btnXoa | JButton | Text: Xóa hàng |
| 31 | btnThemHoaDon | JButton | Text: Thêm hóa đơn |

#### Thiết kế Form quản lý nhân viên

Graphical user interface

Description automatically generated

|  |  |  |  |
| --- | --- | --- | --- |
|  | Componel | Kiểu | Thuộc tính |
| 1 | QuanLyNhanVien | JPanel |  |
| 2 | JLabel6 | JLabel | Text: Mã nhân viên |
| 3 | txtMaNV | JTextField |  |
| 4 | JLabel7 | JLabel | Text: Tên nhân viên |
| 5 | txtTenNV | JTextField |  |
| 6 | JLabel8 | JLabel | Text: Lương cơ bản |
| 7 | txtLuongCoBan | JTextField |  |
| 8 | JLabel9 | JLabel | Text: phụ cấp |
| 9 | txtPhuCap | JTextField |  |
| 10 | JLabel10 | JLabel | Text: Chức vụ |
| 11 | txtChucVu | JTextField |  |
| 12 | tblDanhSach | JTable |  |
| 13 | btnReset | JButton | Text: Reset |
| 14 | btnThem | JButton | Text: Thêm |
| 15 | btnTimKiem | Jbutton | Text: Tìm kiếm theo tên |
| 16 | btnXoa | JButton | Text: Xóa |
| 17 | btnSua | JButton | Text: Sửa |

#### Thiết kế Form quản lý khách hàng

Graphical user interface, application

Description automatically generated

|  |  |  |  |
| --- | --- | --- | --- |
|  | Componel | Kiểu | Thuộc tính |
| 1 | KH | JPanel |  |
| 2 | JLabel6 | JLabel | Text: Mã khách hàng |
| 3 | txtMaKH | JTextField |  |
| 4 | JLabel8 | JLabel | Text: tên khách hàng |
| 5 | txtTenKH | JTextField |  |
| 6 | JLabel9 | JLabel | Text: địa chỉ |
| 7 | txtDiaChi | JTextField |  |
| 8 | tblDanhSach | JTable |  |
| 9 | btnReset | JButton | Text: Reset |
| 10 | btnThem | JButton | Text: Thêm |
| 11 | btnTimKiem | Jbutton | Text: Tìm kiếm |
| 12 | btnXoa | JButton | Text: Xóa |
| 13 | btnSua | JButton | Text: Sửa |

#### Thiết kế Form quản lý tài khoản

Graphical user interface, application

Description automatically generated

|  |  |  |  |
| --- | --- | --- | --- |
|  | Componel | Kiểu | Thuộc tính |
| 1 | TK | JPanel |  |
| 2 | JLabel6 | JLabel | Text: mã nhân viên |
| 3 | cbMaNV | JComboBox |  |
| 4 | JLabel7 | JLabel | Text: quyền truy cập |
| 5 | cbQuyenTruyCap | JComboBox |  |
| 6 | JLabel8 | JLabel | Text: tên đăng nhập |
| 7 | txtTenDangNhap | JTextField |  |
| 8 | JLabel9 | JLabel | Text: mật khẩu |
| 9 | txtMatKhau | JTextField |  |
| 10 | tblDanhSach | JTable |  |
| 11 | btnReset | JButton | Text: Reset |
| 12 | btnThem | JButton | Text: Thêm |
| 13 | btnSapXep | Jbutton | Text: Sắp xếp theo tên |
| 14 | btnXoa | JButton | Text: Xóa |
| 15 | btnSua | JButton | Text: Sửa |

#### Thiết kế Form quản lý sản phẩm

Graphical user interface, application

Description automatically generated

|  |  |  |  |
| --- | --- | --- | --- |
|  | Componel | Kiểu | Thuộc tính |
| 1 | QuanLySanPham | JPanel |  |
| 2 | JLabel6 | JLabel | Text: mã sản phẩm |
| 3 | txtMasanPham | JTextField |  |
| 4 | JLabel7 | JLabel | Text: tên sản phẩm |
| 5 | txtTenSanPham | JTextField |  |
| 6 | JLabel8 | JLabel | Text: đơn giá |
| 7 | txtDonGia | JTextField |  |
| 8 | JLabel9 | JLabel | Text: số lượng |
| 9 | txtSoLuong | JTextField |  |
| 10 | tblDanhSach | JTable |  |
| 11 | btnReset | JButton | Text: Reset |
| 12 | btnThem | JButton | Text: Thêm |
| 13 | btnTimKiem | Jbutton | Text: Tìm kiếm theo tên |
| 14 | btnXoa | JButton | Text: Xóa |
| 15 | btnSua | JButton | Text: Sửa |

#### Thiết kế Form quản lý hóa đơn

Graphical user interface, text, application, email

Description automatically generated

|  |  |  |  |
| --- | --- | --- | --- |
|  | Componel | Kiểu | Thuộc tính |
| 1 | QuanLyHoaDon | JFrame |  |
| 2 | btnTimKiem | JButton | Text: Tìm kiếm theo mã |
| 3 | btnSapXep | JButton | Text: Sắp xếp theo ngày |
| 4 | btnXoa | JButton | Text: xóa |
| 5 | tblDanhsach | JTable |  |
| 6 | lblTongTien | JLabel | Text: Tổng tiền hóa đơn |
| 7 | lblSoLuongHD | JLabel | Text: số lượng hóa đơn |

#### Thiết kế Form thống kê doanh thu

Graphical user interface

Description automatically generated

|  |  |  |  |
| --- | --- | --- | --- |
|  | Componel | Kiểu | Thuộc tính |
| 1 | ThongKeDoanhThu | JPanel |  |
| 2 | daetNgayThongKe | JDateChoose |  |
| 3 | btnReset | JButton | Text: reset |
| 4 | btnSapXep | JButton | Text: sắp xếp theo thành tiền |
| 5 | btnInThongKe | JButton | Text: In danh sách |
| 6 | btnThongKeNgay | JButton | Text: thống kê theo ngày |
| 7 | tblDanhsach | JTable |  |
| 8 | lblSoLuongHD | Jlabel | Text: số lượng hóa đơn |
| 9 | lblTongDoanhThu | JLabel | Text: tổng doanh thu |

# Thực hiện viết mã

## Viết mã tạo CSDL

### Tạo CSDL

create database BTL\_QuanLyBanHang

go

use BTL\_QuanLyBanHang

go

create table NhanVien(

maNV nchar(20) primary key not null,

tenNV nvarchar(30),

chucVu nvarchar(20),

luongCoBan float,

phuCap float

)

create table TaiKhoan(

maNV nchar(20) constraint FK\_maNV foreign key (maNV) references NhanVien(maNV),

username nchar(20) not null,

pass nchar(30) not null,

quyenTruyCap nvarchar(15) not null

)

create table SanPham(

maSP nchar(20) primary key not null,

tenSP nvarchar(40),

donGia float,

soLuong int

)

create table KhachHang(

maKH nchar(20) primary key not null,

tenKH nvarchar(30),

diaChi nvarchar(50)

)

create table HoaDon(

maHD nchar(15) primary key not null,

maNV nchar(20),

soDT nchar(15),

tenKH nvarchar(50),

ngayLap date,

diaChi nvarchar(30)

)

create table ChiTietHoaDon(

maHD nchar(15) not null,

maSP nchar(20) not null,

soLuongBan int,

primary key(maHD, maSP)

)

create table ThongKe(

ngayLap date,

tongDoanhThu float

)

### SQL truy vấn và thao tác

#### NhanVien

|  |  |
| --- | --- |
|  |  |
| **Hành động** | **Câu lệnh SQL** |
| Thêm mới | insert into NhanVien values(?,?,?,?,?) |
|  |  |
| Cập nhật theo mã | update NhanVien set tenNV = ?, chucVu = ?, luongCoBan = ?, phuCap = ? where maNV = ? |
|  |  |
| Xóa theo mã | delete from NhanVien where maNV = ? |
|  |  |
| Truy vấn tất cả | SELECT \* FROM NhanVien |
|  |  |
| Truy vấn theo tên | SELECT \* FROM NhanVien where tenNV like ? |
|  |  |

#### KhachHang

|  |  |
| --- | --- |
|  |  |
| **Hành động** | **Câu lệnh SQL** |
| Thêm mới | insert into KhachHang values(?,?,?) |
|  |  |
| Cập nhật theo mã | update KhachHang set tenKH = ?, diaChi = ? where maKH = ? |
|  |  |
| Xóa theo mã | delete from KhachHang where maKH = ? |
|  |  |
| Truy vấn tất cả | select \* from KhachHang |
|  |  |
| Truy vấn theo tên | select \* from KhachHang where tenKH like ? |
|  |  |

#### SanPham

|  |  |
| --- | --- |
|  |  |
| **Hành động** | **Câu lệnh SQL** |
| Thêm mới | insert into SanPham values(?,?,?,?) |
|  |  |
| Cập nhật theo mã | update SanPham set tenSP = ?, donGia = ?, soLuong = ? where maSP = ? |
|  |  |
| Xóa theo mã | delete from SanPham where maSP = ? |
|  |  |
| Truy vấn tất cả | SELECT \* FROM SanPham |
|  |  |
| Truy vấn theo tên | select \* from SanPham where tenSP like ? |
|  |  |

#### TaiKhoan

|  |  |
| --- | --- |
|  |  |
| **Hành động** | **Câu lệnh SQL** |
| Thêm mới | insert into TaiKhoan values(?,?,?,?) |
|  |  |
| Cập nhật theo mã | update TaiKhoan set username = ?, pass = ?, quyenTruyCap = ? where maNV = ? |
|  |  |
| Xóa theo mã | delete from TaiKhoan where maNV = ? |
|  |  |
| Truy vấn tất cả | SELECT \* FROM TaiKhoan |
|  |  |
| Truy vấn theo tên | select \* from TaiKhoan order by username |
|  |  |

### Các thủ tục lưu tổng hợp thống kê

#### ChiTietHoaDon

|  |
| --- |
| select ChiTietHoaDon.maHD, ngayLap, maNV, tenKH, sum(soLuongBan \* donGia) as 'thanhTien'  from ChiTietHoaDon inner join SanPham  on SanPham.maSP = ChiTietHoaDon.maSP  inner join HoaDon on ChiTietHoaDon.maHD = HoaDon.maHD  where maNV = ?  group by ChiTietHoaDon.maHD,ngayLap, maNV, tenKH |
| select ChiTietHoaDon.maHD, ngayLap, maNV, tenKH, sum(soLuongBan \* donGia) as 'thanhTien'  from ChiTietHoaDon inner join SanPham on SanPham.maSP = ChiTietHoaDon.maSP inner join HoaDon  on ChiTietHoaDon.maHD = HoaDon.maHD  where ChiTietHoaDon.maHD = ? and maNV = ?  group by ChiTietHoaDon.maHD,ngayLap, maNV, tenKH |

#### ThongKeDoanhThu

|  |
| --- |
| select ChiTietHoaDon.maHD, HoaDon.maNV,tenNV, tenKH, ngayLap, sum(soLuongBan \* donGia) as 'thanhTien'  from ChiTietHoaDon inner join SanPham  on SanPham.maSP = ChiTietHoaDon.maSP inner join HoaDon  on ChiTietHoaDon.maHD = HoaDon.maHD, NhanVien  where NhanVien.maNV = HoaDon.maNV  group by ChiTietHoaDon.maHD, HoaDon.maNV, tenNV, tenKH, ngayLap |
| select ChiTietHoaDon.maHD, HoaDon.maNV, tenNV, tenKH, ngayLap, sum(soLuongBan \* donGia) as 'thanhTien'  from ChiTietHoaDon inner join SanPham  on SanPham.maSP = ChiTietHoaDon.maSP inner join HoaDon  on ChiTietHoaDon.maHD = HoaDon.maHD, NhanVien  where NhanVien.maNV = HoaDon.maNV and ngayLap = ?  group by ChiTietHoaDon.maHD, HoaDon.maNV, tenNV, tenKH, ngayLap |
| select ChiTietHoaDon.maHD, HoaDon.maNV, tenNV, tenKH, ngayLap, sum(soLuongBan \* donGia) as 'thanhTien'  from ChiTietHoaDon inner join SanPham  on SanPham.maSP = ChiTietHoaDon.maSP inner join HoaDon  on ChiTietHoaDon.maHD = HoaDon.maHD, NhanVien  where NhanVien.maNV = HoaDon.maNV  group by ChiTietHoaDon.maHD, HoaDon.maNV, tenNV, tenKH, ngayLap order by thanhTien |
| select sum(soLuongBan\*donGia) as 'tongTien'  from SanPham inner join ChiTietHoaDon on SanPham.maSP = ChiTietHoaDon.maSP |

## Lập trình JDBC

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

Graphical user interface, diagram

Description automatically generated

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.*

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

#### NhanVien

package entity;

/\*\*

\*

\* @author Admin

\*/

public class NhanVien {

private String maNV;

private String tenNV;

private String chucVu;

private float luongCoBan;

private float phuCap;

public NhanVien() {

}

public NhanVien(String maNV, String tenNV, String chucVu, float luongCoBan, float phuCap) {

this.maNV = maNV;

this.tenNV = tenNV;

this.chucVu = chucVu;

this.luongCoBan = luongCoBan;

this.phuCap = phuCap;

}

public NhanVien(String maNV) {

this.maNV = maNV;

}

public NhanVien(String maNV,String tenNV) {

this.tenNV = tenNV;

this.maNV = maNV;

}

public String getMaNV() {

return maNV;

}

public void setMaNV(String maNV) {

this.maNV = maNV;

}

public String getTenNV() {

return tenNV;

}

public void setTenNV(String tenNV) {

this.tenNV = tenNV;

}

public String getChucVu() {

return chucVu;

}

public void setChucVu(String chucVu) {

this.chucVu = chucVu;

}

public float getLuongCoBan() {

return luongCoBan;

}

public void setLuongCoBan(float luongCoBan) {

this.luongCoBan = luongCoBan;

}

public float getPhuCap() {

return phuCap;

}

public void setPhuCap(float phuCap) {

this.phuCap = phuCap;

}

public float tinhLuong(float luongCoBan, float phuCap){

return luongCoBan + phuCap;

}

}

#### KhachHang

package entity;

/\*\*

\*

\* @author Admin

\*/

public class KhachHang {

private String maKH;

private String tenKH;

private String diaChi;

public KhachHang() {

}

public String getMaKH() {

return maKH;

}

public String getTenKH() {

return tenKH;

}

public String getDiaChi() {

return diaChi;

}

public KhachHang(String maKH, String tenKH, String diaChi) {

this.maKH = maKH;

this.tenKH = tenKH;

this.diaChi = diaChi;

}

public void setMaKH(String maKH) {

this.maKH = maKH;

}

public void setTenKH(String tenKH) {

this.tenKH = tenKH;

}

public void setDiaChi(String diaChi) {

this.diaChi = diaChi;

}

}

#### SanPham

package entity;

/\*\*

\*

\* @author Ho Nam

\*/

public class SanPham {

protected String maSP;

protected String tenSP;

protected float donGia;

protected int soLuong;

public SanPham() {

}

public SanPham(String maSP, String tenSP, float donGia, int soLuong) {

this.maSP = maSP;

this.tenSP = tenSP;

this.donGia = donGia;

this.soLuong = soLuong;

}

public String getMaSP() {

return maSP;

}

public void setMaSP(String maSP) {

this.maSP = maSP;

}

public String getTenSP() {

return tenSP;

}

public void setTenSP(String tenSP) {

this.tenSP = tenSP;

}

public float getDonGia() {

return donGia;

}

public void setDonGia(float donGia) {

this.donGia = donGia;

}

public int getSoLuong() {

return soLuong;

}

public void setSoLuong(int soLuong) {

this.soLuong = soLuong;

}

}

#### TaiKhoan

package entity;

/\*\*

\*

\* @author Ho Nam

\*/

public class TaiKhoan {

private String maNV;

private String user;

private String password;

private String quyenTruyCap;

public TaiKhoan() {

}

public TaiKhoan(String maNV, String user, String password, String quyenTruyCap) {

this.maNV = maNV;

this.user = user;

this.password = password;

this.quyenTruyCap = quyenTruyCap;

}

public String getMaNV() {

return maNV;

}

public void setMaNV(String maNV) {

this.maNV = maNV;

}

public String getUser() {

return user;

}

public void setUser(String user) {

this.user = user;

}

public String getPassword() {

return password;

}

public void setPassword(String password) {

this.password = password;

}

public String getQuyenTruyCap() {

return quyenTruyCap;

}

public void setQuyenTruyCap(String quyenTruyCap) {

this.quyenTruyCap = quyenTruyCap;

}

}

#### HoaDon

package entity;

/\*\*

\*

\* @author Ho Nam

\*/

public class HoaDon extends NhanVien {

private String maHD;

private String soDT;

private String tenKH;

private String diaChi;

private String ngayLap;

private String tongTien;

public HoaDon() {

}

public HoaDon(String maHD, String maNV, String ngayLap, String soDT, String tenKH, String diaChi) {

super(maNV);

this.soDT = soDT;

this.tenKH = tenKH;

this.diaChi = diaChi;

this.maHD = maHD;

this.ngayLap = ngayLap;

}

public HoaDon(String maNV, String ngayLap, String tenKH){

super(maNV);

this.tenKH = tenKH;

this.ngayLap = ngayLap;

}

public HoaDon(String maNV, String tenNV, String tenKH, String ngayLap) {

super(maNV, tenNV);

this.tenKH = tenKH;

this.ngayLap = ngayLap;

}

public String getMaHD() {

return maHD;

}

public void setMaHD(String maHD) {

this.maHD = maHD;

}

public String getSoDT() {

return soDT;

}

public void setSoDT(String soDT) {

this.soDT = soDT;

}

public String getTenKH() {

return tenKH;

}

public void setTenKH(String tenKH) {

this.tenKH = tenKH;

}

public String getDiaChi() {

return diaChi;

}

public void setDiaChi(String diaChi) {

this.diaChi = diaChi;

}

public String getNgayLap() {

return ngayLap;

}

public void setNgayLap(String ngayLap) {

this.ngayLap = ngayLap;

}

public String getTongTien() {

return tongTien;

}

public void setTongTien(String tongTien) {

this.tongTien = tongTien;

}

}

#### ChiTietHoaDon

package entity;

/\*\*

\*

\* @author Ho Nam

\*/

public class ChiTietHoaDon extends HoaDon{

protected String maHD;

protected String maSP;

protected int soLuongBan;

protected float thanhTien;

public ChiTietHoaDon() {

}

public ChiTietHoaDon(String maHD, String maNV, String tenNV, String tenKH, String ngayLap, float thanhTien){

super(maNV, tenNV, tenKH, ngayLap);

this.maHD = maHD;

this.thanhTien = thanhTien;

}

public ChiTietHoaDon(String maHD, String maSP, int soLuongBan, float thanhTien) {

this.maHD = maHD;

this.maSP = maSP;

this.soLuongBan = soLuongBan;

this.thanhTien = thanhTien;

}

public ChiTietHoaDon(String maHD, String maSP, int soLuongBan) {

this.maHD = maHD;

this.maSP = maSP;

this.soLuongBan = soLuongBan;

}

public ChiTietHoaDon(String maHD, String ngayLap, String maNV,String tenKH, float thanhTien){

super(maNV, ngayLap,tenKH);

this.maHD = maHD;

this.thanhTien = thanhTien;

}

public String getMaHD() {

return maHD;

}

public void setMaHD(String maHD) {

this.maHD = maHD;

}

public String getMaSP() {

return maSP;

}

public void setMaSP(String maSP) {

this.maSP = maSP;

}

public int getSoLuongBan() {

return soLuongBan;

}

public void setSoLuongBan(int soLuongBan) {

this.soLuongBan = soLuongBan;

}

public float getThanhTien() {

return thanhTien;

}

public void setThanhTien(float thanhTien) {

this.thanhTien = thanhTien;

}

}

#### ThongKe

package entity;

/\*\*

\*

\* @author Ho Nam

\*/

public class ThongKe {

private float thanhTien;

public ThongKe() {

}

public float getThanhTien() {

return thanhTien;

}

public void setThanhTien(float thanhTien) {

this.thanhTien = thanhTien;

}

}

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

DAO (Data Access Object) là các lớp giúp giúp làm việc với các bảng trong CSDL. Mỗi lớp được định nghĩa một cách đầy đủ nhất về các thao tác và truy vấn dữ liệu sẽ được sử dụng trong ứng dụng. DAO sau đó sẽ được sử dụng trong các điều khiển sự kiện của giao diện để làm việc với CSDL một cách thuận tiện.

Trong phạm vi ứng dụng này chúng ta cần xây dựng mỗi DAO chứa tối thiểu các hành động sau:

#### NhanVienDAO

|  |
| --- |
| package DAO;  import Connect.JDBCConnection;  import entity.NhanVien;  import java.sql.Connection;  import java.sql.PreparedStatement;  import java.sql.ResultSet;  import java.sql.SQLException;  import java.util.ArrayList;  import java.util.List;  /\*\*  \*  \* @author Ho Nam  \*/  public class NhanVienDao {  public static List<NhanVien> getAll() {  List<NhanVien> nvList = new ArrayList<>();  Connection conn = null;  PreparedStatement pre = null;  ResultSet rs = null;  try {  conn = JDBCConnection.getConnection();  String sql = "select \* from NhanVien";  pre = conn.prepareStatement(sql);  rs = pre.executeQuery();  while (rs.next()) {  NhanVien nv = new NhanVien(rs.getString("maNV"), rs.getString("tenNV"), rs.getString("chucVu"), rs.getFloat("luongCoBan"), rs.getFloat("phuCap"));  nvList.add(nv);  }  } catch (SQLException ex) {  System.out.println("Lỗi: " + ex);  } finally {  if (pre != null) {  try {  pre.close();  } catch (SQLException ex) {  System.out.println("Lỗi: " + ex);  }  }  if (conn != null) {  try {  conn.close();  } catch (SQLException ex) {  System.out.println("Lỗi: " + ex);  }  }  }  return nvList;  }  public static void insert(NhanVien nv) {  Connection conn = null;  PreparedStatement pre = null;  try {  conn = JDBCConnection.getConnection();  String sql = "insert into NhanVien values(?,?,?,?,?)";  pre = conn.prepareStatement(sql);  pre.setString(1, nv.getMaNV());  pre.setString(2, nv.getTenNV());  pre.setString(3, nv.getChucVu());  pre.setFloat(4, nv.getLuongCoBan());  pre.setFloat(5, nv.getPhuCap());  pre.executeUpdate();  } catch (SQLException ex) {  System.out.println("Lỗi: " + ex);  } finally {  if (pre != null) {  try {  pre.close();  } catch (SQLException ex) {  System.out.println("Lỗi: " + ex);  }  }  if (conn != null) {  try {  conn.close();  } catch (SQLException ex) {  System.out.println("Lỗi: " + ex);  }  }  }  }  public static void update(NhanVien nv) {  Connection conn = null;  PreparedStatement pre = null;  try {  conn = JDBCConnection.getConnection();  String sql = "update NhanVien set tenNV = ?, chucVu = ?, luongCoBan = ?, phuCap = ? where maNV = ?";  pre = conn.prepareStatement(sql);  pre.setString(1, nv.getTenNV());  pre.setString(2, nv.getChucVu());  pre.setFloat(3, nv.getLuongCoBan());  pre.setFloat(4, nv.getPhuCap());  pre.setString(5, nv.getMaNV());  pre.executeUpdate();  } catch (SQLException ex) {  System.out.println("Lỗi: " + ex);  } finally {  if (pre != null) {  try {  pre.close();  } catch (SQLException ex) {  System.out.println("Lỗi: " + ex);  }  }  if (conn != null) {  try {  conn.close();  } catch (SQLException ex) {  System.out.println("Lỗi: " + ex);  }  }  }  }  public static void delete(String maNV) {  Connection conn = null;  PreparedStatement pre = null;  try {  conn = JDBCConnection.getConnection();  String sql = "delete from NhanVien where maNV = ?";  pre = conn.prepareStatement(sql);  pre.setString(1, maNV);  pre.executeUpdate();  } catch (SQLException ex) {  System.out.println("Lỗi: " + ex);  } finally {  if (pre != null) {  try {  pre.close();  } catch (SQLException ex) {  System.out.println("Lỗi: " + ex);  }  }  if (conn != null) {  try {  conn.close();  } catch (SQLException ex) {  System.out.println("Lỗi: " + ex);  }  }  }  }  public static List<NhanVien> findByTenNV(String tenNV) {  List<NhanVien> nvList = new ArrayList<>();  Connection conn = null;  PreparedStatement pre = null;  ResultSet rs = null;  try {  conn = JDBCConnection.getConnection();  String sql = "select \* from NhanVien where tenNV like ?";  pre = conn.prepareStatement(sql);  pre.setString(1, "%"+tenNV+"%");  rs = pre.executeQuery();  while (rs.next()) {  NhanVien sp = new NhanVien(rs.getString("maNV"), rs.getString("tenNV"), rs.getString("chucVu"), rs.getFloat("luongCoBan"), rs.getFloat("phuCap"));  nvList.add(sp);  }  } catch (SQLException ex) {  System.out.println("Lỗi: " + ex);  } finally {  if (pre != null) {  try {  pre.close();  } catch (SQLException ex) {  System.out.println("Lỗi: " + ex);  }  }  if (conn != null) {  try {  conn.close();  } catch (SQLException ex) {  System.out.println("Lỗi: " + ex);  }  }  }  return nvList;  }  } |

#### KhachHangDAO

|  |
| --- |
| package DAO;  import Connect.JDBCConnection;  import entity.KhachHang;  import java.sql.Connection;  import java.sql.PreparedStatement;  import java.sql.ResultSet;  import java.sql.SQLException;  import java.util.ArrayList;  import java.util.List;  /\*\*  \*  \* @author Admin  \*/  public class KhachHangDao {  public static List<KhachHang> getAll() {  List<KhachHang> khList = new ArrayList<>();  Connection conn = null;  PreparedStatement pre = null;  ResultSet rs = null;  try {  conn = JDBCConnection.getConnection();  String sql = "select \* from KhachHang";  pre = conn.prepareStatement(sql);  rs = pre.executeQuery();  while (rs.next()) {  KhachHang kh = new KhachHang(rs.getString("maKH"), rs.getString("tenKH"), rs.getString("diaChi"));  khList.add(kh);  }  } catch (SQLException ex) {  System.out.println("Lỗi: " + ex);  } finally {  if (pre != null) {  try {  pre.close();  } catch (SQLException ex) {  System.out.println("Lỗi: " + ex);  }  }  if (conn != null) {  try {  conn.close();  } catch (SQLException ex) {  System.out.println("Lỗi: " + ex);  }  }  }  return khList;  }  public static void insert(KhachHang kh) {  Connection conn = null;  PreparedStatement pre = null;  try {  conn = JDBCConnection.getConnection();  String sql = "insert into KhachHang values(?,?,?)";  pre = conn.prepareStatement(sql);  pre.setString(1, kh.getMaKH());  pre.setString(2, kh.getTenKH());  pre.setString(3, kh.getDiaChi());  pre.executeUpdate();  } catch (SQLException ex) {  System.out.println("Lỗi: " + ex);  } finally {  if (pre != null) {  try {  pre.close();  } catch (SQLException ex) {  System.out.println("Lỗi: " + ex);  }  }  if (conn != null) {  try {  conn.close();  } catch (SQLException ex) {  System.out.println("Lỗi: " + ex);  }  }  }  }  public static void update(KhachHang kh) {  Connection conn = null;  PreparedStatement pre = null;  try {  conn = JDBCConnection.getConnection();  String sql = "update KhachHang set tenKH = ?, diaChi = ? where maKH = ?";  pre = conn.prepareStatement(sql);  pre.setString(1, kh.getTenKH());  pre.setString(2, kh.getDiaChi());  pre.setString(3, kh.getMaKH());  pre.executeUpdate();  } catch (SQLException ex) {  System.out.println("Lỗi: " + ex);  } finally {  if (pre != null) {  try {  pre.close();  } catch (SQLException ex) {  System.out.println("Lỗi: " + ex);  }  }  if (conn != null) {  try {  conn.close();  } catch (SQLException ex) {  System.out.println("Lỗi: " + ex);  }  }  }  }  public static void delete(String maNV) {  Connection conn = null;  PreparedStatement pre = null;  try {  conn = JDBCConnection.getConnection();  String sql = "delete from KhachHang where maKH = ?";  pre = conn.prepareStatement(sql);  pre.setString(1, maNV);  pre.executeUpdate();  } catch (SQLException ex) {  System.out.println("Lỗi: " + ex);  } finally {  if (pre != null) {  try {  pre.close();  } catch (SQLException ex) {  System.out.println("Lỗi: " + ex);  }  }  if (conn != null) {  try {  conn.close();  } catch (SQLException ex) {  System.out.println("Lỗi: " + ex);  }  }  }  }  public static List<KhachHang> findByTenKH(String tenKH) {  List<KhachHang> nvList = new ArrayList<>();  Connection conn = null;  PreparedStatement pre = null;  ResultSet rs = null;  try {  conn = JDBCConnection.getConnection();  String sql = "select \* from KhachHang where tenKH like ?";  pre = conn.prepareStatement(sql);  pre.setString(1, "%"+tenKH+"%");  rs = pre.executeQuery();  while (rs.next()) {  KhachHang sp = new KhachHang(rs.getString("maKH"), rs.getString("tenKH"), rs.getString("diaChi"));  nvList.add(sp);  }  } catch (SQLException ex) {  System.out.println("Lỗi: " + ex);  } finally {  if (pre != null) {  try {  pre.close();  } catch (SQLException ex) {  System.out.println("Lỗi: " + ex);  }  }  if (conn != null) {  try {  conn.close();  } catch (SQLException ex) {  System.out.println("Lỗi: " + ex);  }  }  }  return nvList;  }  } |

#### SanPhamDAO

|  |
| --- |
| package DAO;  import Connect.JDBCConnection;  import entity.SanPham;  import java.sql.Connection;  import java.sql.PreparedStatement;  import java.sql.ResultSet;  import java.sql.SQLException;  import java.util.ArrayList;  import java.util.List;  /\*\*  \*  \* @author Ho Nam  \*/  public class SanPhamDao {  public static List<SanPham> getAll() {  List<SanPham> spList = new ArrayList<>();  Connection conn = null;  PreparedStatement pre = null;  ResultSet rs = null;  try {  conn = JDBCConnection.getConnection();  String sql = "select \* from SanPham";  pre = conn.prepareStatement(sql);  rs = pre.executeQuery();  while (rs.next()) {  SanPham sp = new SanPham(rs.getString("maSP"), rs.getString("tenSP"), rs.getFloat("donGia"), rs.getInt("soLuong"));  spList.add(sp);  }  } catch (SQLException ex) {  System.out.println("Lỗi: " + ex);  } finally {  if (pre != null) {  try {  pre.close();  } catch (SQLException ex) {  System.out.println("Lỗi: " + ex);  }  }  if (conn != null) {  try {  conn.close();  } catch (SQLException ex) {  System.out.println("Lỗi: " + ex);  }  }  }  return spList;  }    public static void getSoLuong(){    }  public static void insert(SanPham sp) {  Connection conn = null;  PreparedStatement pre = null;  try {  conn = JDBCConnection.getConnection();  String sql = "insert into SanPham values(?,?,?,?)";  pre = conn.prepareStatement(sql);  pre.setString(1, sp.getMaSP());  pre.setString(2, sp.getTenSP());  pre.setFloat(3, sp.getDonGia());  pre.setInt(4, sp.getSoLuong());    pre.executeUpdate();    } catch (SQLException ex) {  System.out.println("Lỗi: " + ex);  } finally {  if (pre != null) {  try {  pre.close();  } catch (SQLException ex) {  System.out.println("Lỗi: " + ex);  }  }  if (conn != null) {  try {  conn.close();  } catch (SQLException ex) {  System.out.println("Lỗi: " + ex);  }  }  }  }    public static void update(SanPham sp) {  Connection conn = null;  PreparedStatement pre = null;  try {  conn = JDBCConnection.getConnection();  String sql = "update SanPham set tenSP = ?, donGia = ?, soLuong = ? where maSP = ?";  pre = conn.prepareStatement(sql);  pre.setString(1, sp.getTenSP());  pre.setFloat(2, sp.getDonGia());  pre.setInt(3, sp.getSoLuong());  pre.setString(4, sp.getMaSP());    pre.executeUpdate();    } catch (SQLException ex) {  System.out.println("Lỗi: " + ex);  } finally {  if (pre != null) {  try {  pre.close();  } catch (SQLException ex) {  System.out.println("Lỗi: " + ex);  }  }  if (conn != null) {  try {  conn.close();  } catch (SQLException ex) {  System.out.println("Lỗi: " + ex);  }  }  }  }    public static void delete(String maSP) {  Connection conn = null;  PreparedStatement pre = null;  try {  conn = JDBCConnection.getConnection();  String sql = "delete from SanPham where maSP = ?";  pre = conn.prepareStatement(sql);  pre.setString(1, maSP);    pre.executeUpdate();    } catch (SQLException ex) {  System.out.println("Lỗi: " + ex);  } finally {  if (pre != null) {  try {  pre.close();  } catch (SQLException ex) {  System.out.println("Lỗi: " + ex);  }  }  if (conn != null) {  try {  conn.close();  } catch (SQLException ex) {  System.out.println("Lỗi: " + ex);  }  }  }  }    public static List<SanPham> findByTenSP(String tenSP) {  List<SanPham> spList = new ArrayList<>();  Connection conn = null;  PreparedStatement pre = null;  ResultSet rs = null;  try {  conn = JDBCConnection.getConnection();  String sql = "select \* from SanPham where tenSP like ?";  pre = conn.prepareStatement(sql);  pre.setString(1, "%"+tenSP +"%");  rs = pre.executeQuery();  while (rs.next()) {  SanPham sp = new SanPham(rs.getString("maSP"), rs.getString("tenSP"), rs.getFloat("donGia"), rs.getInt("soLuong"));  spList.add(sp);  }  } catch (SQLException ex) {  System.out.println("Lỗi: " + ex);  } finally {  if (pre != null) {  try {  pre.close();  } catch (SQLException ex) {  System.out.println("Lỗi: " + ex);  }  }  if (conn != null) {  try {  conn.close();  } catch (SQLException ex) {  System.out.println("Lỗi: " + ex);  }  }  }  return spList;  }  } |

#### TaiKhoanDAO

|  |
| --- |
| package DAO;  import Connect.JDBCConnection;  import entity.TaiKhoan;  import java.sql.Connection;  import java.sql.PreparedStatement;  import java.sql.ResultSet;  import java.sql.SQLException;  import java.util.ArrayList;  import java.util.List;  import java.util.logging.Level;  import java.util.logging.Logger;  /\*\*  \*  \* @author Ho Nam  \*/  public class TaiKhoanDao {  public static List<TaiKhoan> getAll() {  List<TaiKhoan> taiKhoans = new ArrayList<>();  Connection conn = null;  PreparedStatement pre = null;  ResultSet rs = null;  String sql = "select \* from TaiKhoan";  try {  conn = JDBCConnection.getConnection();  pre = conn.prepareStatement(sql);  rs = pre.executeQuery();  while (rs.next()) {  TaiKhoan tk = new TaiKhoan(rs.getString("maNV"), rs.getString("username"), rs.getString("pass"), rs.getString("quyenTruyCap"));  taiKhoans.add(tk);  }  } catch (SQLException ex) {  System.out.println("Lỗi: " + ex);  } finally {  if (pre != null) {  try {  pre.close();  } catch (SQLException ex) {  Logger.getLogger(TaiKhoanDao.class.getName()).log(Level.SEVERE, null, ex);  }  }  if (conn != null) {  try {  conn.close();  } catch (SQLException ex) {  Logger.getLogger(TaiKhoanDao.class.getName()).log(Level.SEVERE, null, ex);  }  }  }  return taiKhoans;  }  public static void insert(TaiKhoan tk) {  Connection conn = null;  PreparedStatement pre = null;  try {  conn = JDBCConnection.getConnection();  String sql = "insert into TaiKhoan values(?,?,?,?)";  pre = conn.prepareStatement(sql);  pre.setString(1, tk.getMaNV());  pre.setString(2, tk.getUser());  pre.setString(3, tk.getPassword());  pre.setString(4, tk.getQuyenTruyCap());  pre.executeUpdate();  } catch (SQLException ex) {  System.out.println("Lỗi: " + ex);  } finally {  if (pre != null) {  try {  pre.close();  } catch (SQLException ex) {  System.out.println("Lỗi: " + ex);  }  }  if (conn != null) {  try {  conn.close();  } catch (SQLException ex) {  System.out.println("Lỗi: " + ex);  }  }  }  }  public static void update(TaiKhoan tk) {  Connection conn = null;  PreparedStatement pre = null;  try {  conn = JDBCConnection.getConnection();  String sql = "update TaiKhoan set username = ?, pass = ?, quyenTruyCap = ? where maNV = ?";  pre = conn.prepareStatement(sql);  pre.setString(1, tk.getUser());  pre.setString(2, tk.getPassword());  pre.setString(3, tk.getQuyenTruyCap());  pre.setString(4, tk.getMaNV());  pre.executeUpdate();  } catch (SQLException ex) {  System.out.println("Lỗi: " + ex);  } finally {  if (pre != null) {  try {  pre.close();  } catch (SQLException ex) {  System.out.println("Lỗi: " + ex);  }  }  if (conn != null) {  try {  conn.close();  } catch (SQLException ex) {  System.out.println("Lỗi: " + ex);  }  }  }  }  public static void delete(String maNV) {  Connection conn = null;  PreparedStatement pre = null;  try {  conn = JDBCConnection.getConnection();  String sql = "delete from TaiKhoan where maNV = ?";  pre = conn.prepareStatement(sql);  pre.setString(1, maNV);  pre.executeUpdate();  } catch (SQLException ex) {  System.out.println("Lỗi: " + ex);  } finally {  if (pre != null) {  try {  pre.close();  } catch (SQLException ex) {  System.out.println("Lỗi: " + ex);  }  }  if (conn != null) {  try {  conn.close();  } catch (SQLException ex) {  System.out.println("Lỗi: " + ex);  }  }  }  }  public static List<TaiKhoan> sortBytenNV() {  List<TaiKhoan> taiKhoans = new ArrayList<>();  Connection conn = null;  PreparedStatement pre = null;  ResultSet rs = null;  try {  conn = JDBCConnection.getConnection();  String sql = "select \* from TaiKhoan order by username";  pre = conn.prepareStatement(sql);  rs = pre.executeQuery();  while(rs.next()){  TaiKhoan tk = new TaiKhoan(rs.getString("maNV"), rs.getString("username"), rs.getString("pass"), rs.getString("quyenTruyCap"));  taiKhoans.add(tk);  }  } catch (SQLException ex) {  System.out.println("Lỗi: " + ex);  } finally {  if (pre != null) {  try {  pre.close();  } catch (SQLException ex) {  System.out.println("Lỗi: " + ex);  }  }  if (conn != null) {  try {  conn.close();  } catch (SQLException ex) {  System.out.println("Lỗi: " + ex);  }  }  }  return taiKhoans;  }  } |

#### HoaDonDAO

|  |
| --- |
| package DAO;  import Connect.JDBCConnection;  import entity.HoaDon;  import java.sql.Connection;  import java.sql.PreparedStatement;  import java.sql.ResultSet;  import java.sql.SQLException;  /\*\*  \*  \* @author Ho Nam  \*/  public class HoaDonDao {    static Connection conn = null;  static PreparedStatement pre = null;    public static void insertHD(HoaDon sp) {  try {  conn = JDBCConnection.getConnection();  String sql = "insert into HoaDon values(?,?,?,?,?,?)";  pre = conn.prepareStatement(sql);  pre.setString(1, sp.getMaHD());  pre.setString(2, sp.getMaNV());  pre.setString(3, sp.getSoDT());  pre.setString(4, sp.getTenKH());  pre.setString(5, sp.getNgayLap());  pre.setString(6, sp.getDiaChi());  pre.executeUpdate();  } catch (SQLException ex) {  System.out.println("Lỗi: " + ex);  } finally {  if (pre != null) {  try {  pre.close();  } catch (SQLException ex) {  System.out.println("Lỗi: " + ex);  }  }  if (conn != null) {  try {  conn.close();  } catch (SQLException ex) {  System.out.println("Lỗi: " + ex);  }  }  }  }  public static void deleteHD(String maHD) {  try {  conn = JDBCConnection.getConnection();  String sql = "delete from HoaDon where maHD = ?";  pre = conn.prepareStatement(sql);  pre.setString(1, maHD);  pre.executeUpdate();  } catch (SQLException ex) {  System.out.println("Lỗi: " + ex);  } finally {  if (pre != null) {  try {  pre.close();  } catch (SQLException ex) {  System.out.println("Lỗi: " + ex);  }  }  if (conn != null) {  try {  conn.close();  } catch (SQLException ex) {  System.out.println("Lỗi: " + ex);  }  }  }  }  } |

#### ChiTietHoaDonDAO

|  |
| --- |
| package DAO;  import Connect.JDBCConnection;  import entity.ChiTietHoaDon;  import java.sql.Connection;  import java.sql.PreparedStatement;  import java.sql.ResultSet;  import java.sql.SQLException;  import java.util.ArrayList;  import java.util.List;  import java.util.logging.Level;  import java.util.logging.Logger;  /\*\*  \*  \* @author Ho Nam  \*/  public class ChiTietHoaDonDao {  public static List<ChiTietHoaDon> getTTHD(String maNV) {  List<ChiTietHoaDon> cthds = new ArrayList<>();  Connection conn = null;  PreparedStatement pre = null;  ResultSet rs = null;  String sql = "select ChiTietHoaDon.maHD, ngayLap, maNV, tenKH, sum(soLuongBan \* donGia) as 'thanhTien'\n"  + "from ChiTietHoaDon inner join SanPham on SanPham.maSP = ChiTietHoaDon.maSP\n"  + "inner join HoaDon on ChiTietHoaDon.maHD = HoaDon.maHD\n"  + "where maNV = ?\n"  + "group by ChiTietHoaDon.maHD,ngayLap, maNV, tenKH";  try {  conn = JDBCConnection.getConnection();  pre = conn.prepareStatement(sql);  pre.setString(1, maNV);  rs = pre.executeQuery();  while (rs.next()) {  ChiTietHoaDon ct = new ChiTietHoaDon(rs.getString("maHD"), rs.getString("ngayLap"), rs.getString("maNV"),rs.getString("tenKH"), rs.getFloat("thanhTien"));  cthds.add(ct);  }  } catch (SQLException ex) {  System.out.println("Lỗi: " + ex);  } finally {  if (pre != null) {  try {  pre.close();  } catch (SQLException ex) {  Logger.getLogger(ChiTietHoaDonDao.class.getName()).log(Level.SEVERE, null, ex);  }  }  if (conn != null) {  try {  conn.close();  } catch (SQLException ex) {  Logger.getLogger(ChiTietHoaDonDao.class.getName()).log(Level.SEVERE, null, ex);  }  }  }  return cthds;  }  public static List<ChiTietHoaDon> findByMaHD(String maHD, String maNV) {  List<ChiTietHoaDon> cthds = new ArrayList<>();  Connection conn = null;  PreparedStatement pre = null;  ResultSet rs = null;  String sql = "select ChiTietHoaDon.maHD, ngayLap, maNV, tenKH, sum(soLuongBan \* donGia) as 'thanhTien'\n" +  "from ChiTietHoaDon inner join SanPham on SanPham.maSP = ChiTietHoaDon.maSP inner join HoaDon on ChiTietHoaDon.maHD = HoaDon.maHD\n" +  "where ChiTietHoaDon.maHD = ? and maNV = ?\n" +  "group by ChiTietHoaDon.maHD,ngayLap, maNV, tenKH";  try {  conn = JDBCConnection.getConnection();  pre = conn.prepareStatement(sql);  pre.setString(1, maHD);  pre.setString(2, maNV);  rs = pre.executeQuery();  while (rs.next()) {  ChiTietHoaDon ct = new ChiTietHoaDon(rs.getString("maHD"), rs.getString("ngayLap"), rs.getString("maNV"),rs.getString("tenKH"), rs.getFloat("thanhTien"));  cthds.add(ct);  }  } catch (SQLException ex) {  System.out.println("Lỗi: " + ex);  } finally {  if (pre != null) {  try {  pre.close();  } catch (SQLException ex) {  Logger.getLogger(ChiTietHoaDonDao.class.getName()).log(Level.SEVERE, null, ex);  }  }  if (conn != null) {  try {  conn.close();  } catch (SQLException ex) {  Logger.getLogger(ChiTietHoaDonDao.class.getName()).log(Level.SEVERE, null, ex);  }  }  }  return cthds;  }  public static void insertCTHD(ChiTietHoaDon cthd) {  Connection conn = null;  PreparedStatement pre = null;  try {  conn = JDBCConnection.getConnection();  String sql = "insert into ChiTietHoaDon values(?,?,?)";  pre = conn.prepareStatement(sql);  pre.setString(1, cthd.getMaHD());  pre.setString(2, cthd.getMaSP());  pre.setInt(3, cthd.getSoLuongBan());  pre.executeUpdate();  } catch (SQLException ex) {  System.out.println("Lỗi: " + ex);  } finally {  if (pre != null) {  try {  pre.close();  } catch (SQLException ex) {  System.out.println("Lỗi: " + ex);  }  }  if (conn != null) {  try {  conn.close();  } catch (SQLException ex) {  System.out.println("Lỗi: " + ex);  }  }  }  }    public static void deleteCTHD(String maHD) {  Connection conn = null;  PreparedStatement pre = null;  try {  conn = JDBCConnection.getConnection();  String sql = "delete from ChiTietHoaDon where maHD = ?";  pre = conn.prepareStatement(sql);  pre.setString(1, maHD);  pre.executeUpdate();  } catch (SQLException ex) {  System.out.println("Lỗi: " + ex);  } finally {  if (pre != null) {  try {  pre.close();  } catch (SQLException ex) {  System.out.println("Lỗi: " + ex);  }  }  if (conn != null) {  try {  conn.close();  } catch (SQLException ex) {  System.out.println("Lỗi: " + ex);  }  }  }  }    public static List<ChiTietHoaDon> sortByDate(String maNV) {  List<ChiTietHoaDon> cthds = new ArrayList<>();  Connection conn = null;  PreparedStatement pre = null;  ResultSet rs = null;  try {  String sql = "select ChiTietHoaDon.maHD, ngayLap, maNV, tenKH, sum(soLuongBan \* donGia) as 'thanhTien'\n" +  "from ChiTietHoaDon inner join SanPham on SanPham.maSP = ChiTietHoaDon.maSP inner join HoaDon on ChiTietHoaDon.maHD = HoaDon.maHD\n" +  "where maNV = ?\n" +  "group by ChiTietHoaDon.maHD,ngayLap, maNV, tenKH\n" +  "order by ngayLap";  conn = JDBCConnection.getConnection();  pre = conn.prepareStatement(sql);  pre.setString(1, maNV);  rs = pre.executeQuery();  while (rs.next()) {  ChiTietHoaDon ct = new ChiTietHoaDon(rs.getString("maHD"), rs.getString("ngayLap"), rs.getString("maNV"),rs.getString("tenKH"), rs.getFloat("thanhTien"));  cthds.add(ct);  }  } catch (SQLException ex) {  System.out.println("Lỗi: " + ex);  } finally {  if (pre != null) {  try {  pre.close();  } catch (SQLException ex) {  Logger.getLogger(ChiTietHoaDonDao.class.getName()).log(Level.SEVERE, null, ex);  }  }  if (conn != null) {  try {  conn.close();  } catch (SQLException ex) {  Logger.getLogger(ChiTietHoaDonDao.class.getName()).log(Level.SEVERE, null, ex);  }  }  }  return cthds;  }  } |

#### ThongKeDoanhThuDAO

|  |
| --- |
| package DAO;  import Connect.JDBCConnection;  import entity.ChiTietHoaDon;  import entity.ThongKe;  import java.sql.Connection;  import java.sql.PreparedStatement;  import java.sql.ResultSet;  import java.sql.SQLException;  import java.util.ArrayList;  import java.util.List;  import java.util.logging.Level;  import java.util.logging.Logger;  /\*\*  \*  \* @author Ho Nam  \*/  public class ThongKeDoanhSoDao {  public static List<ChiTietHoaDon> getData(){  List<ChiTietHoaDon> cthdList = new ArrayList<>();  Connection conn = null;  PreparedStatement pre = null;  ResultSet rs = null;  try {  String sql = "select ChiTietHoaDon.maHD, HoaDon.maNV,tenNV, tenKH, ngayLap, sum(soLuongBan \* donGia) as 'thanhTien'\n" +  "from ChiTietHoaDon inner join SanPham on SanPham.maSP = ChiTietHoaDon.maSP inner join HoaDon on ChiTietHoaDon.maHD = HoaDon.maHD, NhanVien\n" +  "where NhanVien.maNV = HoaDon.maNV\n" +  "group by ChiTietHoaDon.maHD, HoaDon.maNV, tenNV, tenKH, ngayLap";  conn = JDBCConnection.getConnection();  pre = conn.prepareStatement(sql);  rs = pre.executeQuery();  while (rs.next()) {  ChiTietHoaDon ct = new ChiTietHoaDon(rs.getString("maHD"), rs.getString("maNV"),rs.getString("tenNV"), rs.getString("tenKH"), rs.getString("ngayLap"), rs.getFloat("thanhTien"));  cthdList.add(ct);  }  } catch (SQLException ex) {  System.out.println("Lỗi: " + ex);  } finally {  if (pre != null) {  try {  pre.close();  } catch (SQLException ex) {  Logger.getLogger(ChiTietHoaDonDao.class.getName()).log(Level.SEVERE, null, ex);  }  }  if (conn != null) {  try {  conn.close();  } catch (SQLException ex) {  Logger.getLogger(ChiTietHoaDonDao.class.getName()).log(Level.SEVERE, null, ex);  }  }  }  return cthdList;  }    public static List<ChiTietHoaDon> findByDate(String ngayLap){  List<ChiTietHoaDon> cthdList = new ArrayList<>();  Connection conn = null;  PreparedStatement pre = null;  ResultSet rs = null;  try {  String sql = "select ChiTietHoaDon.maHD, HoaDon.maNV, tenNV, tenKH, ngayLap, sum(soLuongBan \* donGia) as 'thanhTien'\n" +  "from ChiTietHoaDon inner join SanPham on SanPham.maSP = ChiTietHoaDon.maSP inner join HoaDon on ChiTietHoaDon.maHD = HoaDon.maHD, NhanVien\n" +  "where NhanVien.maNV = HoaDon.maNV and ngayLap = ?\n" +  "group by ChiTietHoaDon.maHD, HoaDon.maNV, tenNV, tenKH, ngayLap";  conn = JDBCConnection.getConnection();  pre = conn.prepareStatement(sql);  pre.setString(1, ngayLap);  rs = pre.executeQuery();  while (rs.next()) {  ChiTietHoaDon ct = new ChiTietHoaDon(rs.getString("maHD"), rs.getString("maNV"),rs.getString("tenNV"), rs.getString("tenKH"), rs.getString("ngayLap"), rs.getFloat("thanhTien"));  cthdList.add(ct);  }  } catch (SQLException ex) {  System.out.println("Lỗi: " + ex);  } finally {  if (pre != null) {  try {  pre.close();  } catch (SQLException ex) {  Logger.getLogger(ChiTietHoaDonDao.class.getName()).log(Level.SEVERE, null, ex);  }  }  if (conn != null) {  try {  conn.close();  } catch (SQLException ex) {  Logger.getLogger(ChiTietHoaDonDao.class.getName()).log(Level.SEVERE, null, ex);  }  }  }  return cthdList;  }    public static List<ChiTietHoaDon> sortByTT() {  List<ChiTietHoaDon> cthds = new ArrayList<>();  Connection conn = null;  PreparedStatement pre = null;  ResultSet rs = null;  try {  String sql = "select ChiTietHoaDon.maHD, HoaDon.maNV, tenNV, tenKH, ngayLap, sum(soLuongBan \* donGia) as 'thanhTien'\n" +  "from ChiTietHoaDon inner join SanPham on SanPham.maSP = ChiTietHoaDon.maSP inner join HoaDon on ChiTietHoaDon.maHD = HoaDon.maHD, NhanVien\n" +  "where NhanVien.maNV = HoaDon.maNV\n" +  "group by ChiTietHoaDon.maHD, HoaDon.maNV, tenNV, tenKH, ngayLap\n" +  "order by thanhTien";  conn = JDBCConnection.getConnection();  pre = conn.prepareStatement(sql);  rs = pre.executeQuery();  while (rs.next()) {  ChiTietHoaDon ct = new ChiTietHoaDon(rs.getString("maHD"), rs.getString("maNV"),rs.getString("tenNV"), rs.getString("tenKH"), rs.getString("ngayLap"), rs.getFloat("thanhTien"));  cthds.add(ct);  }  } catch (SQLException ex) {  System.out.println("Lỗi: " + ex);  } finally {  if (pre != null) {  try {  pre.close();  } catch (SQLException ex) {  Logger.getLogger(ChiTietHoaDonDao.class.getName()).log(Level.SEVERE, null, ex);  }  }  if (conn != null) {  try {  conn.close();  } catch (SQLException ex) {  Logger.getLogger(ChiTietHoaDonDao.class.getName()).log(Level.SEVERE, null, ex);  }  }  }  return cthds;  }    public static List<ThongKe> getTongTien() {  List<ThongKe> tkList = new ArrayList<>();    Connection conn = null;  PreparedStatement pre = null;  ResultSet rs = null;  try {  conn = JDBCConnection.getConnection();  String sql = "select sum(soLuongBan\*donGia) as 'tongTien'\n"  + "from SanPham inner join ChiTietHoaDon on SanPham.maSP = ChiTietHoaDon.maSP";  pre = conn.prepareStatement(sql);  rs = pre.executeQuery();  while (rs.next()) {  ThongKe tk = new ThongKe();  tk.setThanhTien(Float.parseFloat(rs.getString("tongTien")));  tkList.add(tk);  }  } catch (SQLException ex) {  System.out.println("Lỗi: " + ex);  } finally {  if (pre != null) {  try {  pre.close();  } catch (SQLException ex) {  System.out.println("Lỗi: " + ex);  }  }  if (conn != null) {  try {  conn.close();  } catch (SQLException ex) {  System.out.println("Lỗi: " + ex);  }  }  }  return tkList;  }  } |

## Viết mã cho ứng dụng

### Đăng nhập (DangNhap)

|  |
| --- |
| public void xuly() {  String username = txtTenDangNhap.getText();  String pass = txtMatKhau.getText();  String option = cbQuyenTruyCap.getSelectedItem().toString();  if (username.isEmpty() || pass.isEmpty()) {  JOptionPane.showMessageDialog(rootPane, "Thông tin nhập không được trống", "Error", 1);  return;  }  Connection conn = null;  PreparedStatement pre = null;  ResultSet rs = null;  try {  conn = JDBCConnection.getConnection();  String sql = "Select \* from TaiKhoan inner join NhanVien on TaiKhoan.maNV = NhanVien.maNV where username = ? and pass = ? and quyenTruyCap = ?";  pre = conn.prepareStatement(sql);  pre.setString(1, username);  pre.setString(2, pass);  pre.setString(3, option);  rs = pre.executeQuery();  if (rs.next()) {  if (rs.getString("quyenTruyCap").equalsIgnoreCase("user")) {  NV uf = new NV();  uf.setVisible(true);  uf.setTenNV(rs.getString("tenNV"));  uf.setMaNV(rs.getString("maNV"));  this.dispose();  } else {  //new QL().setVisible(true);  QL ql = new QL();  ql.setVisible(true);  ql.setTenNV(rs.getString("tenNV"));  ql.setMaNV(rs.getString("maNV"));  this.dispose();  }  } else {  JOptionPane.showMessageDialog(rootPane, "Tên đăng nhập hoặc mật khẩu không đúng", "Error", 1);  }  } catch (Exception ex) {  System.out.println("Lỗi: " + ex);  } finally {  if (pre != null) {  try {  pre.close();  } catch (SQLException ex) {  Logger.getLogger(DangNhap.class.getName()).log(Level.SEVERE, null, ex);  }  }  if (conn != null) {  try {  conn.close();  } catch (SQLException ex) {  Logger.getLogger(DangNhap.class.getName()).log(Level.SEVERE, null, ex);  }  }  }  } |

### Màn hình chào (Loading)

|  |
| --- |
| private void init() {  setLocationRelativeTo(null);  setResizable(false);  new Timer(15, new ActionListener() {  @Override  public void actionPerformed(ActionEvent e) {  int value = pgbLoad.getValue();  if (value < pgbLoad.getMaximum()) {  pgbLoad.setValue(value + 1);  if (value == 50) {  jLabel2.setText("Đang tải tài nguyên...");  } else if (value == 80) {  jLabel2.setText("Tải thành công...");  }  } else {  Loading.this.dispose();  }  }  }).start();  } |

### Giao diện( QL, NV)

|  |
| --- |
| public QL() {  new Loading(this, true).setVisible(true);  initComponents();  init();  List<DanhMuc> listItem = new ArrayList<>();  listItem.add(new DanhMuc("TrangChu", jpnTrangchu, jlbTrangchu));  listItem.add(new DanhMuc("Menu", jpnMenu, jlbMenu));  listItem.add(new DanhMuc("NhanVien", jpnNhanVien, jlbNhanVien));  listItem.add(new DanhMuc("KhachHang", jpnKhachHang, jlbKhachHang));  //listItem.add(new DanhMuc("HoaDon", jpnHoaDon, jlbHoaDon));  listItem.add(new DanhMuc("ThongKe", jpnThongKe, jlbThongKe));  //listItem.add(new DanhMuc("DanhMuc", jpnDanhMuc, jlbDanhMuc));  listItem.add(new DanhMuc("TaiKhoan", jpnTaiKhoan, jlbTaiKhoan));  ChuyenManHinh controller = new ChuyenManHinh(jpnView);  controller.setView(jpnTrangchu, jlbTrangchu);  controller.setEvent(listItem);  } |

### Bán hàng (BanHang)

|  |
| --- |
| private void loadMaHD() {  Connection conn = null;  PreparedStatement pre = null;  ResultSet rs = null;  cbMaHD.removeAllItems();  try {  conn = JDBCConnection.getConnection();  String sql = "select \* from HoaDon where maNV = ?";  pre = conn.prepareStatement(sql);  pre.setString(1, txtMaNV.getText().trim());  rs = pre.executeQuery();  while (rs.next()) {  cbMaHD.addItem(rs.getString("maHD").trim());  }  } catch (SQLException ex) {  JOptionPane.showMessageDialog(rootPane, ex);  }  }  private int getSoLuong() {  int soluong = 0;  try {  conn = JDBCConnection.getConnection();  String sql = "select soLuong from SanPham where maSP = ?";  pre = conn.prepareStatement(sql);  pre.setString(1, txtMaSP.getText());  rs = pre.executeQuery();  while (rs.next()) {  soluong = rs.getInt("soLuong");  }  } catch (SQLException ex) {  System.out.println("Lỗi: " + ex);  } finally {  if (pre != null) {  try {  pre.close();  } catch (SQLException ex) {  System.out.println("Lỗi: " + ex);  }  }  if (conn != null) {  try {  conn.close();  } catch (SQLException ex) {  System.out.println("Lỗi: " + ex);  }  }  }  return soluong;  }  private void getSoluongCon() {  try {  conn = JDBCConnection.getConnection();  String sql = "select soLuong from SanPham where maSP = ?";  pre = conn.prepareStatement(sql);  pre.setString(1, txtMaSP.getText());  rs = pre.executeQuery();  while (rs.next()) {  lblSoLuongCon.setText("Số lượng còn: " + rs.getInt("soLuong"));  }  } catch (SQLException ex) {  System.out.println("Lỗi: " + ex);  } finally {  if (pre != null) {  try {  pre.close();  } catch (SQLException ex) {  System.out.println("Lỗi: " + ex);  }  }  if (conn != null) {  try {  conn.close();  } catch (SQLException ex) {  System.out.println("Lỗi: " + ex);  }  }  }  }  private void tongThanhTien() {  try {  conn = JDBCConnection.getConnection();  String sql = "select sum(soLuongBan \* donGia) as 'thanhTien'\n"  + "from ChiTietHoaDon inner join SanPham on SanPham.maSP = ChiTietHoaDon.maSP\n"  + "where maHD = ?";  pre = conn.prepareStatement(sql);  pre.setString(1, cbMaHD.getSelectedItem().toString().trim());  rs = pre.executeQuery();  while (rs.next()) {  txtTongTien.setText(String.valueOf(rs.getFloat("thanhTien")));  }  } catch (SQLException ex) {  JOptionPane.showMessageDialog(rootPane, ex);  } finally {  if (conn != null) {  try {  conn.close();  } catch (SQLException ex) {  Logger.getLogger(BanHang.class.getName()).log(Level.SEVERE, null, ex);  }  }  }  }  private void loadDataTable() {  try {  conn = JDBCConnection.getConnection();  String sql = "select ChiTietHoaDon.maSP, tenSP, soLuongBan, donGia, soLuongBan \* donGia as 'thanhTien'\n"  + "from ChiTietHoaDon inner join SanPham on SanPham.maSP = ChiTietHoaDon.maSP\n"  + "where maHD = ?";  pre = conn.prepareStatement(sql);  pre.setString(1, cbMaHD.getSelectedItem().toString().trim());  rs = pre.executeQuery();  tableModel.setRowCount(0);  while (rs.next()) {  Vector arr = new Vector();  arr.add(rs.getString("maSP"));  arr.add(rs.getString("tenSP"));  arr.add(rs.getInt("soLuongBan"));  arr.add(rs.getFloat("donGia"));  arr.add(rs.getString("thanhTien"));  tableModel.addRow(arr);  }  } catch (SQLException ex) {  JOptionPane.showMessageDialog(rootPane, ex);  } finally {  if (conn != null) {  try {  conn.close();  } catch (SQLException ex) {  Logger.getLogger(BanHang.class.getName()).log(Level.SEVERE, null, ex);  }  }  }  }  **private boolean checkMaHD() {**  **try {**  **conn = JDBCConnection.getConnection();**  **String sqlCheck = "SELECT maHD FROM HoaDon";**  **PreparedStatement pre = conn.prepareStatement(sqlCheck);**  **ResultSet rs = pre.executeQuery();**  **while (rs.next()) {**  **if (this.txtMaHoaDon.getText().trim().equalsIgnoreCase(rs.getString("maHD").toString().trim())) {**  **return false;**  **}**  **}**  **} catch (Exception ex) {**  **JOptionPane.showMessageDialog(rootPane, ex);**  **} finally {**  **if (conn != null) {**  **try {**  **conn.close();**  **} catch (SQLException ex) {**  **Logger.getLogger(BanHang.class.getName()).log(Level.SEVERE, null, ex);**  **}**  **}**  **}**  **return true;**  **}**  **private void LoadTenSP() {**  **cbTenSP.removeAllItems();**  **try {**  **conn = JDBCConnection.getConnection();**  **String sql = "select tenSP from SanPham";**  **pre = conn.prepareStatement(sql);**  **rs = pre.executeQuery();**  **while (rs.next()) {**  **cbTenSP.addItem(rs.getString("tenSP").trim());**  **}**  **} catch (SQLException ex) {**  **JOptionPane.showMessageDialog(rootPane, ex);**  **}**  **}** |

### Quản lý nhân viên (QuanLyNhanVien)

|  |
| --- |
| List<NhanVien> nvList = new ArrayList<>();  DefaultTableModel tableModel;  public void getData(){  nvList = NhanVienDao.getAll();  tableModel.setRowCount(0);  for (NhanVien nv : nvList) {  tableModel.addRow(new Object[]{  nv.getMaNV(),  nv.getTenNV(),  nv.getChucVu(),  nv.getLuongCoBan(),  nv.getPhuCap(),  nv.tinhLuong(nv.getLuongCoBan(), nv.getPhuCap())  });  }  }  private void btnThem1ActionPerformed(java.awt.event.ActionEvent evt) {  // TODO add your handling code here:  String maNV = txtMaNV.getText();  if(maNV.isEmpty()){  //JOptionPane.showMessageDialog(rootPane, "Mã nhân viên không để trống");  MsgBox.alert(this, "Mã nhân viên không để trống");  return;  }  String tenNV = txtTenNV.getText();  if(tenNV.isEmpty()){  //JOptionPane.showMessageDialog(rootPane, "Tên nhân viên không để trống");  MsgBox.alert(this, "Mã nhân viên không để trống");  return;  }  String chucVu = txtChucVu.getText();  if(chucVu.isEmpty()){  //JOptionPane.showMessageDialog(rootPane, "Chức vụ không để trống");  MsgBox.alert(this, "Mã nhân viên không để trống");  return;  }  try{  float luongCoBan = Float.parseFloat(txtLuongCoBan.getText());  float phuCap = Float.parseFloat(txtPhuCap.getText());  if(checkMaNV(txtMaNV.getText())){  NhanVien nv = new NhanVien(maNV, tenNV, chucVu, luongCoBan, phuCap);  NhanVienDao.insert(nv);  }else{  //JOptionPane.showMessageDialog(rootPane, "Mã nhân viên không được trùng");  MsgBox.alert(this, "Mã nhân viên không để trống");  }  }catch(NumberFormatException ex){  //JOptionPane.showMessageDialog(rootPane, "Nhập số và không để trống");  MsgBox.alert(this, "Mã nhân viên không để trống");  }catch(Exception ex){  //JOptionPane.showMessageDialog(rootPane, ex);  MsgBox.alert(this, "Mã nhân viên không để trống");  }  getData();  txtMaNV.setEnabled(true);  btnSua.setEnabled(false);  btnXoa.setEnabled(false);  }  private void btnResetActionPerformed(java.awt.event.ActionEvent evt) {  // TODO add your handling code here:  txtMaNV.setText("");  txtChucVu.setText("");  txtLuongCoBan.setText("");  txtPhuCap.setText("");  txtTenNV.setText("");  txtMaNV.setEnabled(true);  btnSua.setEnabled(false);  btnXoa.setEnabled(false);  }  private void btnSuaActionPerformed(java.awt.event.ActionEvent evt) {  // TODO add your handling code here:  String tenNV = txtTenNV.getText();  if(tenNV.isEmpty()){  //JOptionPane.showMessageDialog(rootPane, "Tên nhân viên không để trống");  MsgBox.alert(this, "Mã nhân viên không để trống");  return;  }  String chucVu = txtChucVu.getText();  if(chucVu.isEmpty()){  MsgBox.alert(this, "Chức vụ không để trống");  return;  }  try{  float luongCoBan = Float.parseFloat(txtLuongCoBan.getText());  float phuCap = Float.parseFloat(txtPhuCap.getText());  }catch(NumberFormatException ex){  MsgBox.alert(this, "Mã nhân viên không để trống");  }  int row = tblDanhSach.getSelectedRow();  if (row >= 0) {  NhanVien s = nvList.get(row);  s.setMaNV(txtMaNV.getText().trim());  s.setTenNV(txtTenNV.getText().trim());  s.setChucVu(txtChucVu.getText().trim());  s.setLuongCoBan(Float.parseFloat(txtLuongCoBan.getText()));  s.setPhuCap(Float.parseFloat(txtPhuCap.getText()));  NhanVienDao.update(s);  getData();  btnSua.setEnabled(false);  btnXoa.setEnabled(false);  }  }  private void btnTimKiemActionPerformed(java.awt.event.ActionEvent evt) {  // TODO add your handling code here:  try{  String tenNV = JOptionPane.showInputDialog(this, "Nhập tên nhân viên cần tìm");  if (tenNV.equals("") == false) {  nvList = NhanVienDao.findByTenNV(tenNV);  tableModel.setRowCount(0);  for (NhanVien nv : nvList) {  tableModel.addRow(new Object[]{  nv.getMaNV(),  nv.getTenNV(),  nv.getChucVu(),  nv.getLuongCoBan(),  nv.getPhuCap(),  nv.tinhLuong(nv.getLuongCoBan(), nv.getPhuCap())  });  }  } else {  getData();  }  }catch(Exception ex){  MsgBox.alert(this, "Mã nhân viên không để trống");  }  }  private void btnXoaActionPerformed(java.awt.event.ActionEvent evt) {  // TODO add your handling code here:  int row = tblDanhSach.getSelectedRow();  if (row >= 0) {  NhanVien nv = nvList.get(row);  boolean option = MsgBox.confirm(this, "Bạn có muốn xóa không");  if (option == true) {  TaiKhoanDao.delete(nv.getMaNV());  NhanVienDao.delete(nv.getMaNV());  }  btnSua.setEnabled(false);  btnXoa.setEnabled(false);  getData();  }  }  private boolean checkMaNV(String maNV){  for(NhanVien nv : nvList){  if(nv.getMaNV().trim().equalsIgnoreCase(maNV.trim())){  return false;  }  }  return true;  }  private void tblDanhSachMouseClicked(java.awt.event.MouseEvent evt) {  // TODO add your handling code here:  int row = tblDanhSach.getSelectedRow();  if (row >= 0) {  NhanVien nv = nvList.get(row);  btnSua.setEnabled(true);  btnXoa.setEnabled(true);  txtMaNV.setEnabled(false);  txtTenNV.setText(nv.getTenNV().trim());  txtMaNV.setText(nv.getMaNV().trim());  txtChucVu.setText(nv.getChucVu().trim());  txtLuongCoBan.setText(String.valueOf(nv.getLuongCoBan()));  txtPhuCap.setText(String.valueOf(nv.getPhuCap()));  }  } |

### Quản lý tài khoản (TK)

|  |
| --- |
| List<TaiKhoan> tkList = new ArrayList<>();  DefaultTableModel tableModel;  public void getData() {  tableModel.setRowCount(0);  tkList = TaiKhoanDao.getAll();  for (TaiKhoan tk : tkList) {  tableModel.addRow(new Object[]{  tk.getMaNV(),  tk.getUser(),  tk.getPassword(),  tk.getQuyenTruyCap()  });  }  }  private void loadMaNV() {  Connection conn = null;  PreparedStatement pre = null;  ResultSet rs = null;  cbMaNV.removeAllItems();  try {  conn = JDBCConnection.getConnection();  String sql = "select \* from NhanVien";  pre = conn.prepareStatement(sql);  rs = pre.executeQuery();  while (rs.next()) {  cbMaNV.addItem(rs.getString("maNV").trim());  }  } catch (SQLException ex) {  Logger.getLogger(TK.class.getName()).log(Level.SEVERE, null, ex);  }  }  private void btnThemActionPerformed(java.awt.event.ActionEvent evt) {  // TODO add your handling code here:  String maNV = cbMaNV.getSelectedItem().toString().trim();  String user = txtTenDangNhap.getText().trim();  if (user.isEmpty()) {  MsgBox.alert(this, "Tên đăng nhập không để trống");  return;  }  String password = txtMatKhau.getText().trim();  if (password.isEmpty()) {  MsgBox.alert(this, "Mật khẩu không để trống");  return;  }  String quyenTruyCap = cbQuyenTruyCap.getSelectedItem().toString().trim();  try {  if (checkMaNV()) {  TaiKhoan tk = new TaiKhoan(maNV, user, password, quyenTruyCap);  TaiKhoanDao.insert(tk);  } else {  MsgBox.alert(this, "Mã nhân viên đã tồn tại");  return;  }  } catch (Exception ex) {  MsgBox.alert(this, "Lỗi" +ex);  }  getData();  btnSua.setEnabled(false);  btnXoa.setEnabled(false);  }  private void btnResetActionPerformed(java.awt.event.ActionEvent evt) {  // TODO add your handling code here:  txtMatKhau.setText("");  txtTenDangNhap.setText("");  loadMaNV();  btnSua.setEnabled(false);  btnXoa.setEnabled(false);  }  private void btnSuaActionPerformed(java.awt.event.ActionEvent evt) {  // TODO add your handling code here:  String user = txtTenDangNhap.getText().trim();  if (user.isEmpty()) {  MsgBox.alert(this, "Tên đăng nhập không để trống");  return;  }  String password = txtMatKhau.getText().trim();  if (password.isEmpty()) {  MsgBox.alert(this, "Mật khẩu không để trống");  return;  }  int row = tblDanhSach.getSelectedRow();  if (row >= 0) {  TaiKhoan s = tkList.get(row);  s.setMaNV(cbMaNV.getSelectedItem().toString().trim());  s.setUser(txtTenDangNhap.getText().trim());  s.setPassword(txtMatKhau.getText().trim());  s.setQuyenTruyCap(cbQuyenTruyCap.getSelectedItem().toString().trim());  TaiKhoanDao.update(s);  btnSua.setEnabled(false);  btnXoa.setEnabled(false);  getData();  }  }  private void btnSapXepActionPerformed(java.awt.event.ActionEvent evt) {  // TODO add your handling code here:  tableModel.setRowCount(0);  tkList = TaiKhoanDao.sortBytenNV();  for (TaiKhoan tk : tkList) {  tableModel.addRow(new Object[]{  tk.getMaNV(),  tk.getUser(),  tk.getPassword(),  tk.getQuyenTruyCap()  });  }  }  private void btnXoaActionPerformed(java.awt.event.ActionEvent evt) {  // TODO add your handling code here:  int row = tblDanhSach.getSelectedRow();  if (row >= 0) {  TaiKhoan tk = tkList.get(row);  boolean option = MsgBox.confirm(this, "Mã sản phẩm không để trống");  if (option == true) {  TaiKhoanDao.delete(tk.getMaNV());  }  btnSua.setEnabled(false);  btnXoa.setEnabled(false);  getData();  }  }  private void tblDanhSachMouseClicked(java.awt.event.MouseEvent evt) {  // TODO add your handling code here:  int row = tblDanhSach.getSelectedRow();  if (row >= 0) {  TaiKhoan tk = tkList.get(row);  btnSua.setEnabled(true);  btnXoa.setEnabled(true);  cbMaNV.removeAllItems();  cbMaNV.addItem(tk.getMaNV().trim());  cbQuyenTruyCap.setSelectedItem(tk.getQuyenTruyCap().trim());  txtMatKhau.setText(tk.getPassword().trim());  txtTenDangNhap.setText(tk.getUser().trim());  }  }  private boolean checkMaNV() {  for (TaiKhoan tk : tkList) {  if (tk.getMaNV().trim().equalsIgnoreCase(cbMaNV.getSelectedItem().toString().trim())) {  return false;  }  }  return true;  } |

### Quản lý sản phẩm(QuanLySanPham)

|  |
| --- |
| List<SanPham> spList = new ArrayList<>();  DefaultTableModel tableModel;  private void getData() {  spList = SanPhamDao.getAll();  tableModel.setRowCount(0);  for (SanPham sp : spList) {  tableModel.addRow(new Object[]{  sp.getMaSP(),  sp.getTenSP(),  sp.getSoLuong(),  sp.getDonGia()  });  }  }  private boolean checkMaSP() {  for (SanPham sp : spList) {  if (sp.getMaSP().trim().equalsIgnoreCase(txtMaSanPham.getText().trim())) {  return false;  }  }  return true;  }  private void btnResetActionPerformed(java.awt.event.ActionEvent evt) {  // TODO add your handling code here:  txtTenSanPham.setText("");  txtMaSanPham.setText("");  txtSoLuong.setText("");  txtDonGia.setText("");  txtMaSanPham.setEnabled(true);  btnSua.setEnabled(false);  btnXoa.setEnabled(false);  getData();  }  private void btnThemActionPerformed(java.awt.event.ActionEvent evt) {  // TODO add your handling code here:    String maSP = txtMaSanPham.getText();  if (maSP.isEmpty()) {  //JOptionPane.showMessageDialog(rootPane, "Mã sản phẩm không để trống");  MsgBox.alert(this, "Mã sản phẩm không để trống");  return;  }  String tenSP = txtTenSanPham.getText();  if (tenSP.isEmpty()) {  //JOptionPane.showMessageDialog(rootPane, "Tên sản phẩm không để trống");  MsgBox.alert(this, "Tên sản phẩm không để trống");  return;  }  try {  float donGia = Float.parseFloat(txtDonGia.getText());  int soLuong = Integer.parseInt(txtSoLuong.getText());  if (checkMaSP()) {  SanPham sp = new SanPham(maSP, tenSP, donGia, soLuong);  SanPhamDao.insert(sp);  } else {  //JOptionPane.showMessageDialog(rootPane, "Mã sản phẩm không được trùng");  MsgBox.alert(this, "Mã sản phẩm không được trùng");  }  } catch (NumberFormatException ex) {  //JOptionPane.showMessageDialog(rootPane, "Yêu cầu nhập số và không để trống");  MsgBox.alert(this, "Yêu cầu nhập số và không để trống");  } catch (Exception ex) {  //JOptionPane.showMessageDialog(rootPane, "Lỗi: " + ex);  MsgBox.alert(this, "Lỗi: " + ex);  }  getData();  txtMaSanPham.setEnabled(true);  btnSua.setEnabled(false);  btnXoa.setEnabled(false);  }  private void btnXoaActionPerformed(java.awt.event.ActionEvent evt) {  // TODO add your handling code here:  try {  int row = tblDanhSach.getSelectedRow();  if (row >= 0) {  SanPham sp = spList.get(row);  //int option = JOptionPane.showConfirmDialog(rootPane, "Bạn có muốn xóa không");  boolean option = MsgBox.confirm(this, "Bạn có muốn xóa không");  if (option == true) {  if (checkMaSPXoa(txtMaSanPham.getText())) {  SanPhamDao.delete(sp.getMaSP());  } else {  //JOptionPane.showMessageDialog(rootPane, "Không xóa được vì thông tin sản phẩm đang tồn tại trong hóa đơn");    MsgBox.alert(this, "Không xóa được vì thông tin sản phẩm đang tồn tại trong hóa đơn");  return;  }  }  btnSua.setEnabled(false);  btnXoa.setEnabled(false);  getData();  }  } catch (Exception ex) {  //JOptionPane.showMessageDialog(rootPane, ex);  MsgBox.alert(this, "Lỗi: " + ex);  }  btnResetActionPerformed(evt);  }  private boolean checkMaSPXoa(String maSP) {  Connection conn = null;  PreparedStatement pre = null;  ResultSet rs = null;  try {  conn = JDBCConnection.getConnection();  String sql = "select maSP from ChiTietHoaDon";  pre = conn.prepareStatement(sql);  rs = pre.executeQuery();  while (rs.next()) {  if (maSP.trim().equalsIgnoreCase(rs.getString("maSP").toString().trim())) {  return false;  }  }  } catch (SQLException ex) {  //JOptionPane.showMessageDialog(rootPane, ex);  MsgBox.alert(this, "Lỗi: " + ex);  }  return true;  }  private void tblDanhSachMouseClicked(java.awt.event.MouseEvent evt) {  // TODO add your handling code here:  int row = tblDanhSach.getSelectedRow();  if (row >= 0) {  SanPham sp = spList.get(row);  btnSua.setEnabled(true);  btnXoa.setEnabled(true);  txtMaSanPham.setEnabled(false);  txtMaSanPham.setText(sp.getMaSP().trim());  txtTenSanPham.setText(sp.getTenSP().trim());  txtDonGia.setText(String.valueOf(sp.getDonGia()));  txtSoLuong.setText(String.valueOf(sp.getSoLuong()));  }  }  private void btnSuaActionPerformed(java.awt.event.ActionEvent evt) {  // TODO add your handling code here:  if (txtTenSanPham.getText().isEmpty()) {  //JOptionPane.showMessageDialog(rootPane, "Tên sản phẩm không để trống");  MsgBox.alert(this, "Tên sản phẩm không để trống");  return;  }  try {  Float.parseFloat(txtDonGia.getText());  Integer.parseInt(txtSoLuong.getText());  } catch (NumberFormatException ex) {  //JOptionPane.showMessageDialog(rootPane, "Yêu cầu nhập số và không để trống");  MsgBox.alert(this, "Yêu cầu nhập số và không để trống");  }  int row = tblDanhSach.getSelectedRow();  if (row >= 0) {  SanPham s = spList.get(row);  s.setMaSP(txtMaSanPham.getText());  s.setTenSP(txtTenSanPham.getText());  s.setDonGia(Float.parseFloat(txtDonGia.getText()));  s.setSoLuong(Integer.parseInt(txtSoLuong.getText()));  SanPhamDao.update(s);  btnSua.setEnabled(false);  btnXoa.setEnabled(false);  getData();  }  }  private void btnTimKiemActionPerformed(java.awt.event.ActionEvent evt) {  // TODO add your handling code here:  try {  String tenSP = JOptionPane.showInputDialog(this, "Nhập tên sản phẩm cần tìm");  if (tenSP.equals("") == false) {  spList = SanPhamDao.findByTenSP(tenSP);  tableModel.setRowCount(0);  for (SanPham sp : spList) {  tableModel.addRow(new Object[]{  sp.getMaSP(),  sp.getTenSP(),  sp.getSoLuong(),  sp.getDonGia()  });  }  } else {  getData();  }  } catch (Exception ex) {  //JOptionPane.showMessageDialog(rootPane, "Không tìm thấy!");  MsgBox.alert(this, "Không tìm thấy!");  }  } |

### Quản lý khách hàng (KH)

|  |
| --- |
| List<KhachHang> khList = new ArrayList<>();  DefaultTableModel tableModel;  public void getData() {  tableModel.setRowCount(0);  khList = KhachHangDao.getAll();  for (KhachHang kh : khList) {  tableModel.addRow(new Object[]{  kh.getMaKH(),  kh.getTenKH(),  kh.getDiaChi()  });  }  }  private void loadMaKH() {  Connection conn = null;  PreparedStatement pre = null;  ResultSet rs = null;  cbMaKH.removeAllItems();  try {  conn = JDBCConnection.getConnection();  String sql = "select \* from KhachHang";  pre = conn.prepareStatement(sql);  rs = pre.executeQuery();  while (rs.next()) {  cbMaKH.addItem(rs.getString("maKH").trim());  }  } catch (SQLException ex) {  Logger.getLogger(TK.class.getName()).log(Level.SEVERE, null, ex);  }  }  private void btnThemActionPerformed(java.awt.event.ActionEvent evt) {  // TODO add your handling code here:  String maNV = cbMaKH.getSelectedItem().toString().trim();  String user = txtTenKhachHang.getText().trim();  if (user.isEmpty()) {  MsgBox.alert(this, "Tên đăng nhập không để trống");  return;  }  String password = txtDiaChi.getText().trim();  if (password.isEmpty()) {  MsgBox.alert(this, "Mật khẩu không để trống");  return;  }  getData();  btnSua.setEnabled(false);  btnXoa.setEnabled(false);  }  private void btnResetActionPerformed(java.awt.event.ActionEvent evt) {  // TODO add your handling code here:  txtDiaChi.setText("");  txtTenKhachHang.setText("");  loadMaKH();  btnSua.setEnabled(false);  btnXoa.setEnabled(false);  }  private void btnSuaActionPerformed(java.awt.event.ActionEvent evt) {  // TODO add your handling code here:  String user = txtTenKhachHang.getText().trim();  if (user.isEmpty()) {  MsgBox.alert(this, "Tên đăng nhập không để trống");  return;  }  String password = txtDiaChi.getText().trim();  if (password.isEmpty()) {  MsgBox.alert(this, "Mật khẩu không để trống");  return;  }  int row = tblDanhSach.getSelectedRow();  if (row >= 0) {  KhachHang s = khList.get(row);  s.setMaKH(cbMaKH.getSelectedItem().toString().trim());  s.setTenKH(txtTenKhachHang.getText().trim());  s.setDiaChi(txtDiaChi.getText().trim());  KhachHangDao.update(s);  btnSua.setEnabled(false);  btnXoa.setEnabled(false);  getData();  }  }  private void btnTimKiemActionPerformed(java.awt.event.ActionEvent evt) {  // TODO add your handling code here:  try{  String tenKH = JOptionPane.showInputDialog(this, "Nhập tên khách hàng cần tìm");  if (tenKH.equals("") == false) {  khList = KhachHangDao.findByTenKH(tenKH);  tableModel.setRowCount(0);  for (KhachHang kh : khList) {  tableModel.addRow(new Object[]{  kh.getMaKH(),  kh.getTenKH(),  kh.getDiaChi()  });  }  } else {  getData();  }  }catch(Exception ex){  //JOptionPane.showMessageDialog(rootPane, "Không tìm thấy!");  MsgBox.alert(this, "Mã nhân viên không để trống");  }  }  private void btnXoaActionPerformed(java.awt.event.ActionEvent evt) {  // TODO add your handling code here:  int row = tblDanhSach.getSelectedRow();  if (row >= 0) {  KhachHang tk = khList.get(row);  boolean option = MsgBox.confirm(this, "Mã sản phẩm không để trống");  if (option == true) {  KhachHangDao.delete(tk.getMaKH());  }  btnSua.setEnabled(false);  btnXoa.setEnabled(false);  getData();  }  }  private void tblDanhSachMouseClicked(java.awt.event.MouseEvent evt) {  // TODO add your handling code here:  int row = tblDanhSach.getSelectedRow();  if (row >= 0) {  KhachHang kh = khList.get(row);  btnSua.setEnabled(true);  btnXoa.setEnabled(true);  cbMaKH.removeAllItems();  cbMaKH.addItem(kh.getMaKH().trim());  cbMaKH.setSelectedItem(kh.getMaKH().trim());  txtDiaChi.setText(kh.getDiaChi().trim());  txtTenKhachHang.setText(kh.getTenKH().trim());  }  } |

### Quản lý hóa đơn(QuanLyHoaDon)

|  |
| --- |
| List<HoaDon> thdList = new ArrayList<>();  List<ChiTietHoaDon> cTHDList = new ArrayList<>();    DefaultTableModel tableModel;  private void getData(){  cTHDList = ChiTietHoaDonDao.getTTHD(this.getMaNV());  tableModel.setRowCount(0);  for (ChiTietHoaDon cthd : cTHDList) {  tableModel.addRow(new Object[]{  cthd.getMaHD(),  cthd.getNgayLap(),  cthd.getMaNV(),  cthd.getTenKH(),  cthd.getThanhTien()  });  }  }  private void loadSoLuongHD(){  int n = tableModel.getRowCount();  lblSoLuongHD.setText("Số lượng hóa đơn: " + String.valueOf(n));  }  private void loadTongTien(){  Connection conn = null;  PreparedStatement pre = null;  ResultSet rs = null;  try {  conn = JDBCConnection.getConnection();  String sql = "select sum(soLuongBan\*donGia) as 'tongTien' "  + "from SanPham inner join ChiTietHoaDon on SanPham.maSP = ChiTietHoaDon.maSP "  + "inner join HoaDon on ChiTietHoaDon.maHD = HoaDon.maHD "  + "where maNV = ?";  pre = conn.prepareStatement(sql);  pre.setString(1, getMaNV());  rs = pre.executeQuery();  while(rs.next()){  lblTongTien.setText("Tổng tiền hóa đơn: " + rs.getString("tongTien").trim());  }  } catch (SQLException ex) {  System.out.println("Lỗi: " + ex);  } finally {  if (pre != null) {  try {  pre.close();  } catch (SQLException ex) {  System.out.println("Lỗi: " + ex);  }  }  if (conn != null) {  try {  conn.close();  } catch (SQLException ex) {  System.out.println("Lỗi: " + ex);  }  }  }    }  private void btnXoaActionPerformed(java.awt.event.ActionEvent evt) {  // TODO add your handling code here:  row = tblDanhSach.getSelectedRow();  if(row >= 0){  ChiTietHoaDon cthd = cTHDList.get(row);  int option = JOptionPane.showConfirmDialog(rootPane, "Bạn có muốn xóa không");  if (option == 0) {  ChiTietHoaDonDao.deleteCTHD(cthd.getMaHD());  HoaDonDao.deleteHD(cthd.getMaHD());  }  btnXoa.setEnabled(false);  }  getData();  loadSoLuongHD();  loadTongTien();  }  int row;  private void tblDanhSachMouseClicked(java.awt.event.MouseEvent evt) {  // TODO add your handling code here:  row = tblDanhSach.getSelectedRow();  btnXoa.setEnabled(true);  }  private void btnTimKiemActionPerformed(java.awt.event.ActionEvent evt) {  // TODO add your handling code here:  try{  String MaHD = JOptionPane.showInputDialog(this, "Nhập mã hóa đơn cần tìm");  if (MaHD.equals("") == false) {  cTHDList = ChiTietHoaDonDao.findByMaHD(MaHD, getMaNV());  tableModel.setRowCount(0);  for (ChiTietHoaDon cthd : cTHDList) {  tableModel.addRow(new Object[]{  cthd.getMaHD(),  cthd.getNgayLap(),  cthd.getMaNV(),  cthd.getTenKH(),  cthd.getThanhTien()  });  lblTongTien.setText("Tổng tiền hóa đơn: " + String.valueOf(cthd.getThanhTien()));  }  loadSoLuongHD();  } else {  getData();  loadTongTien();  loadSoLuongHD();  }  }catch(Exception ex){  JOptionPane.showMessageDialog(rootPane, "Nhập không hợp lệ");  }    btnXoa.setEnabled(false);  }  private void btnSapXepActionPerformed(java.awt.event.ActionEvent evt) {  // TODO add your handling code here:  cTHDList = ChiTietHoaDonDao.sortByDate(getMaNV());  tableModel.setRowCount(0);  for (ChiTietHoaDon cthd : cTHDList) {  tableModel.addRow(new Object[]{  cthd.getMaHD(),  cthd.getNgayLap(),  cthd.getMaNV(),  cthd.getTenKH(),  cthd.getThanhTien()  });  }  loadSoLuongHD();  loadTongTien();  btnXoa.setEnabled(false);  }  private void formWindowOpened(java.awt.event.WindowEvent evt) {  // TODO add your handling code here:  getData();  loadTongTien();  loadSoLuongHD();  } |

### Thống kê doanh thu(ThongKeDoanhThu)

|  |
| --- |
| List<ChiTietHoaDon> cthdList = new ArrayList<>();    List<ThongKe> tkList = new ArrayList<>();    DefaultTableModel tableModel;  private void loadData() {  tableModel.setRowCount(0);  cthdList = ThongKeDoanhSoDao.getData();  for (ChiTietHoaDon cthd : cthdList) {  tableModel.addRow(new Object[]{  cthd.getMaHD(),  cthd.getMaNV(),  cthd.getTenNV(),  cthd.getTenKH(),  cthd.getNgayLap(),  cthd.getThanhTien()  });  }  }  private void loadSoLuongHD() {  int n = tableModel.getRowCount();  lblSoLuongHD.setText("Số lượng hóa đơn: " + String.valueOf(n));  }  private void loadTongTien() {  tkList = ThongKeDoanhSoDao.getTongTien();  for(ThongKe tk : tkList){  lblTongDanhThu.setText("Tổng doanh thu: " + tk.getThanhTien());  }  }  private void btnResetActionPerformed(java.awt.event.ActionEvent evt) {  // TODO add your handling code here:  dateNgayThongKe.setDate(null);  try {  loadData();  loadSoLuongHD();  loadTongTien();  } catch (Exception ex) {  MsgBox.alert(this, "Chưa có dữ liệu");  }  }  private void btnSapXepActionPerformed(java.awt.event.ActionEvent evt) {  // TODO add your handling code here:  try {  cthdList = ThongKeDoanhSoDao.sortByTT();  tableModel.setRowCount(0);  for (ChiTietHoaDon cthd : cthdList) {  tableModel.addRow(new Object[]{  cthd.getMaHD(),  cthd.getMaNV(),  cthd.getTenNV(),  cthd.getTenKH(),  cthd.getNgayLap(),  cthd.getThanhTien()  });  }  loadSoLuongHD();  loadTongTien();  } catch (Exception ex) {  MsgBox.alert(this, "Chưa có dữ liệu để sắp xếp");  }  }  private void btnInThongKeActionPerformed(java.awt.event.ActionEvent evt) {  FileOutputStream fileOutputStream = null;  BufferedOutputStream outputStream = null;  XSSFWorkbook workbook = null;  //Chọn đường dẫn để lưu file  JFileChooser excelFileChooser = new JFileChooser("C:\\Users\\dell\\Downloads\\Qly\_QuanCafe\\XuatThongKeFileExcel");  //Thay đổi tiêu đề  excelFileChooser.setDialogTitle("Save as");  //Chỉ lưu dưới dạng các tập tin này  FileNameExtensionFilter fnef = new FileNameExtensionFilter("EXCEL FILES", "xls", "xlsx", "xlsm");  excelFileChooser.setFileFilter(fnef);  int excelChoose = excelFileChooser.showSaveDialog(null);  Cell cell;  Row row;  //check nếu file đã được chọn  if (excelChoose == JFileChooser.APPROVE\_OPTION) {  try {  workbook = new XSSFWorkbook();  XSSFSheet excelSheet = workbook.createSheet("JTable Sheet");  int rowNum = 0;  row = excelSheet.createRow(rowNum++);  cell = row.createCell(0);  cell.setCellValue("THỐNG KÊ DANH SÁCH HOÁ ĐƠN");  row = excelSheet.createRow(rowNum++);  cell = row.createCell(0);  cell.setCellValue("MÃ HOÁ ĐƠN");  cell = row.createCell(1);  cell.setCellValue("MÃ NHÂN VIÊN");  cell = row.createCell(2);  cell.setCellValue("TÊN NHÂN VIÊN");  cell = row.createCell(3);  cell.setCellValue("TÊN KHÁCH HÀNG");  cell = row.createCell(4);  cell.setCellValue("NGÀY LẬP");  cell = row.createCell(5);  cell.setCellValue("THÀNH TIỀN");  for (int i = 0; i < tableModel.getRowCount(); i++) {  XSSFRow excelRow = excelSheet.createRow(rowNum++);  for (int j = 0; j < tableModel.getColumnCount(); j++) {  XSSFCell excelCell = excelRow.createCell(j);  excelCell.setCellValue(tableModel.getValueAt(i, j).toString());  }  }  fileOutputStream = new FileOutputStream(excelFileChooser.getSelectedFile() + ".xlsx");  outputStream = new BufferedOutputStream(fileOutputStream);  workbook.write(outputStream);  MsgBox.alert(this, "Xuất dữ liệu thành công");  } catch (FileNotFoundException ex) {  MsgBox.alert(this, "Đường dẫn file không hợp lệ");  } catch (IOException ex) {  MsgBox.alert(this, "Lỗi: " + ex);  } finally {  if (outputStream != null) {  try {  outputStream.close();  } catch (IOException ex) {  Logger.getLogger(ThongKeDoanhThu.class.getName()).log(Level.SEVERE, null, ex);  }  }  if (fileOutputStream != null) {  try {  fileOutputStream.close();  } catch (IOException ex) {  Logger.getLogger(ThongKeDoanhThu.class.getName()).log(Level.SEVERE, null, ex);  }  }  if (workbook != null) {  try {  workbook.close();  } catch (IOException ex) {  Logger.getLogger(ThongKeDoanhThu.class.getName()).log(Level.SEVERE, null, ex);  }  }  }  }  }  private void btnThongKeNgayActionPerformed(java.awt.event.ActionEvent evt) {  // TODO add your handling code here:  try {  SimpleDateFormat dateFormat = new SimpleDateFormat("MM/dd/yyyy");  String ngayLap = dateFormat.format(dateNgayThongKe.getDate());  int tongTT = 0;  if (dateNgayThongKe.getDate() != null || dateNgayThongKe.equals("") == false) {  cthdList = ThongKeDoanhSoDao.findByDate(ngayLap);  tableModel.setRowCount(0);  for (ChiTietHoaDon cthd : cthdList) {  tableModel.addRow(new Object[]{  cthd.getMaHD(),  cthd.getMaNV(),  cthd.getTenNV(),  cthd.getTenKH(),  cthd.getNgayLap(),  cthd.getThanhTien()  });  tongTT += cthd.getThanhTien();  lblTongDanhThu.setText("Tổng doanh thu: " + String.valueOf(tongTT));  }  loadSoLuongHD();  } else {  loadData();  loadTongTien();  loadSoLuongHD();  }  } catch (Exception ex) {  MsgBox.alert(this, "Ngày thống kê không để trống!");  }  } |

# Kiểm thử

## Kiểm thử form Đăng nhập

* Tên đăng nhập, mật khẩu phải không đươc trống

## Kiểm thử form nhân viên

* check trống: mã nhân viên, tên nhân viên và chức vụ
* check luongcoban, phucap: phải là số và không được để trống
* check mã nhân viên: trùng mã nhân viên
* chỉ admin được quyền quản lý form nhân viên

## Kiểm thử form khách hàng

* check trống: mã khách hàng, tên khách hàng và địa chỉ

## Kiểm thử form sản phẩm

* check trống: mã sản phẩm, tên sản phẩm, số lượng sản phẩm, đơn giá
* check số lượng và đơn giá: phải nhập số

## Kiểm thử form tài khoản

* check trống: mật khẩu, tên đăng nhập