# TRƯỜNG CAO ĐẲNG THỰC HÀNH FPT



**BÁO CÁO**

**DỰ ÁN**

**PHẦN MỀM QUẢN LÝ NHÀ HÀNG**

**RESTAURANT MANAGEMENT SOFTWARE**

**Người hướng dẫn: nguyễn đỗ anh khoa**

**Thành viên thực hiện:**

**Đoàn Trương Gia Bảo – PS11020**

**Trần Minh Hào – PS11072**

**Trần Tuấn Kiệt – PS11044**

**Võ Thị Cẩm Nguyệt – PS11038**

***Thành phố Hồ Chí Minh – 2020***

**LỜI CẢM ƠN**

*Đề tài* ***“Phần mềm quản lý quán ăn”*** *là nội dung nhóm chúng tôi chọn để nghiên cứu và làm đồ án khi theo học chuyên ngành Ứng dụng phần mềm tại Trường Cao đẳng Thực hành FPT (FPT Polytechnic) Cơ sở TP.HCM.*

*Để hoàn thành quá trình nghiên cứu và hoàn thiện đề tài này, lời đầu tiên chúng tôi xin chân thành cảm ơn sâu sắc đến* ***Thầy Nguyễn Đỗ Anh Khoa*** *thuộc Khoa Công nghệ thông tin. Thầy đã trực tiếp chỉ bảo và hướng dẫn nhóm trong suốt quá trình nghiên cứu để chúng tôi hoàn thiện đề tài này.*

*Nhân dịp này, chúng tôi cũng xin cảm ơn Khoa Công nghệ thông tin Cao đẳng Thực hành FPT (FPT Polytechnic), lãnh đạo và các anh chị đang công tác tại trường đã tạo điều kiện và thời gian cho chúng tôi trong suốt quá trình nghiên cứu.*

*Cuối cùng xin cảm ơn các thành viên của nhóm* ***The Dream Maker*** *đã đồng lòng, đoàn kết thực hiện dự án này.*

*Thành phố Hồ Chí Minh, tháng 08 năm 2020*

***The Dream Maker***

**MỤC LỤC**

[TRƯỜNG CAO ĐẲNG THỰC HÀNH FPT 1](#_Toc48556927)

[LỜI CẢM ƠN 2](#_Toc48556928)

[LỜI MỞ ĐẦU 6](#_Toc48556929)

[1 CHƯƠNG I – GIỚI THIỆU ĐỀ TÀI 8](#_Toc48556930)

[1.1 Tìm hiểu “Xây dựng Phần mềm quản lý nhà hàng” 8](#_Toc48556931)

[1.2 Nền tảng công nghệ 8](#_Toc48556932)

[1.3 Công cụ lập trình và công nghệ được sử dụng 8](#_Toc48556933)

[1.4 Giới thiệu ngôn ngữ lập trình Java 9](#_Toc48556934)

[1.5 Giới thiệu hệ quản trị cơ sở dữ liệu Microsoft SQL Server 9](#_Toc48556935)

[1.5.1 Java Database Connectivity (JDBC) 9](#_Toc48556936)

[2 CHƯƠNG II – PHÂN TÍCH 10](#_Toc48556937)

[2.1 Hiện trạng thực tế 10](#_Toc48556938)

[2.2 Yêu cầu hệ thống 10](#_Toc48556939)

[2.3 Use Case 11](#_Toc48556940)

[3 CHƯƠNG III THIẾT KẾ 14](#_Toc48556941)

[3.1 Mô hình triển khai 14](#_Toc48556942)

[3.2 Thiết kế CSDL 14](#_Toc48556943)

[3.2.1 Sơ đồ quan hệ thực thể 14](#_Toc48556944)

[3.2.1.1 ERD Diagram level 1 15](#_Toc48556945)

[3.2.1.2 ERD Diagram level 2 16](#_Toc48556946)

[3.2.2 Thiết kế chi tiết các thực thể 16](#_Toc48556947)

[3.2.2.1 Nhân viên 16](#_Toc48556948)

[3.2.2.2 Loại thẻ khách hàng 17](#_Toc48556949)

[3.2.2.3 Thẻ khách hàng 17](#_Toc48556950)

[3.2.2.4 Loại sản phẩm 18](#_Toc48556951)

[3.2.2.5 Sản phẩm 18](#_Toc48556952)

[3.2.2.6 Đơn hàng 18](#_Toc48556953)

[3.2.2.7 Chi tiết đơn hàng 19](#_Toc48556954)

[3.3 Thiết kế giao diện 19](#_Toc48556955)

[3.3.1 Sơ đồ tổ chức giao diện 19](#_Toc48556956)

[3.3.1.1 Đăng nhập 21](#_Toc48556957)

[3.3.1.2 Giao diện chính 23](#_Toc48556958)

[3.3.1.3 Khách hàng 24](#_Toc48556959)

[3.3.1.4 Nhân viên 26](#_Toc48556960)

[3.3.1.5 Sản phẩm 29](#_Toc48556961)

[3.3.1.6 Hóa đơn 30](#_Toc48556962)

[3.3.1.7 Thống kê 33](#_Toc48556963)

[4 CHƯƠNG IV THỰC HIỆN VIẾT MÃ 37](#_Toc48556964)

[4.1 Viết mã tạo CSDL 37](#_Toc48556965)

[4.1.1 Tạo CSDL 37](#_Toc48556966)

[4.1.2 SQL truy vấn và thao tác bảng 38](#_Toc48556967)

[4.1.2.1 LOAITHEKHACHHANG 38](#_Toc48556968)

[4.1.2.2 LOAITHEKHACHHANG 39](#_Toc48556969)

[4.1.2.3 THEKHACHHANG 39](#_Toc48556970)

[4.1.2.4 LOAISANPHAM 39](#_Toc48556971)

[4.1.2.5 SANPHAM 40](#_Toc48556972)

[4.1.2.6 DONHANG 40](#_Toc48556973)

[4.1.2.7 CHITIETDONHANG 40](#_Toc48556974)

[4.1.3 Các thủ tục lưu tổng hợp thống kê 40](#_Toc48556975)

[4.1.3.1 Thống kê doanh thu theo ngày 40](#_Toc48556976)

[4.1.3.2 Thống kê doanh thu theo tháng 41](#_Toc48556977)

[4.1.3.3 Thống kê doanh thu theo năm 41](#_Toc48556978)

[4.2 Lập trình JDBC 41](#_Toc48556979)

[4.2.1 Lớp hỗ trợ 42](#_Toc48556980)

[4.2.1.1 Lớp tiện ích DateHelper 42](#_Toc48556981)

[4.2.1.2 Lớp tiện ích JdbcHelper 43](#_Toc48556982)

[4.2.1.3 Lớp tiện ích DateConvert 44](#_Toc48556983)

[4.2.1.4 Lớp tiện ích QRHelper 47](#_Toc48556984)

[4.2.1.5 Lớp tiện ích TichDiem 48](#_Toc48556985)

[4.2.2 Model class – Các lớp mô tả dữ liệu 48](#_Toc48556986)

[4.2.2.1 NHANVIEN 48](#_Toc48556987)

[4.2.2.2 LOAITHEKH 52](#_Toc48556988)

[4.2.2.3 THEKHACHHANG 53](#_Toc48556989)

[4.2.2.4 SANPHAM 55](#_Toc48556990)

[4.2.3 DAO Class – Các lớp truy xuất dữ liệu 57](#_Toc48556991)

[4.2.3.1 NhanVienDAO 58](#_Toc48556992)

[4.2.3.2 KhachHangDAO 61](#_Toc48556993)

[4.2.3.3 LoaiSanPhamDAO 63](#_Toc48556994)

[4.2.3.4 SanPhamDAO 65](#_Toc48556995)

[4.2.3.5 DonHangDAO 68](#_Toc48556996)

[4.2.3.6 ChiTietDonHangDAO 69](#_Toc48556997)

[4.2.3.7 ThongKeDAO 71](#_Toc48556998)

[4.3 Viết mã cho ứng dụng 76](#_Toc48556999)

[4.3.1 Các lớp tiện ích 76](#_Toc48557000)

[4.3.1.1 ShareHelper 76](#_Toc48557001)

[4.3.1.2 DialogHelper 78](#_Toc48557002)

[4.3.1.3 SQLConnect 79](#_Toc48557003)

[4.3.2 Viết mã cho các chức năng 79](#_Toc48557004)

[4.3.2.1 Màn hình chào (ChaoJDialog) 79](#_Toc48557005)

[4.3.2.2 Form đăng nhập (SignInFrame) 80](#_Toc48557006)

[4.3.2.3 Giao diện chính 82](#_Toc48557007)

[4.3.2.4 Nhân viên 90](#_Toc48557008)

[4.3.2.5 Khách hàng 99](#_Toc48557009)

[4.3.2.6 Sản phẩm 108](#_Toc48557010)

[4.3.2.7 Hóa đơn 115](#_Toc48557011)

[4.3.2.8 Thống kê 127](#_Toc48557012)

[5 CHƯƠNG V KIỂM THỬ 135](#_Toc48557013)

[5.1 Kiểm thử form nhân viên 135](#_Toc48557014)

[5.2 Kiểm thử form sản phẩm 135](#_Toc48557015)

[5.3 Kiểm thử form khách hàng 135](#_Toc48557016)

[6 CHƯƠNG VI – TỔNG KẾT VÀ HƯỚNG PHÁT TRIỂN 136](#_Toc48557017)

[6.1 Tổng kết 136](#_Toc48557018)

[6.2 Hướng phát triển 136](#_Toc48557019)

**LỜI MỞ ĐẦU**

Vấn đề lớn nhất khi kinh doanh mà các nhà hàng gặp phải là chi phí quản lý vận hành. Thất thoát kho, mất tiền cũng là do không kiểm soát chặt chẽ việc kinh doanh. **Vậy vấn đề nằm ở đâu?** Chính từ việc quản lý thủ công. Thực tế những phương pháp trên chỉ mang lại chữ “tiện” nhưng không hề mang lại cái “lợi”. Dù là với quy mô lớn, nhỏ, kinh doanh đơn lẻ hay vận hành đa chi nhánh, **phần mềm quản lý nhà hàng** trực tuyến đều có thể đáp ứng được. Được thiết kế các gói tính năng dành riêng cho các mô hình quán ăn, cùng với đó là sự tiện lợi cũng như dễ dàng sử dụng để quán ăn, nhà hàng nào cũng có thể sử dụng được.

Được xây dựng từ chính nhu cầu quản lý của những bộ phận cơ bản trong một nhà hàng, chúng tôi tự tin mình hoàn toàn có thể đáp ứng tốt mọi yêu cầu của bạn về một phần mềm quản lý nhà hàng. Từ nghiệp vụ đặt chỗ, order, thanh toán.

Thay vì việc ghi chép thủ công, hóa đơn tay dễ gây nhầm lẫn, sai sót, nhân viên gian lận nay với phần mềm quản lý nhà hàng thời gian phục vụ được rút gọn, vừa tránh được nguy cơ thất thoát từ việc thu ngân, thanh toán của nhân viên. Mặt khác, khách hàng cảm thấy hài lòng, thích thú với sự chuyên nghiệp mà mình nhận được, điều này sẽ khiến nhà hàng của bạn ghi điểm nhiều hơn.

Thông thường công việc quản lý tại các nhà hàng hiện nay đòi hỏi sự có mặt trực tiếp từ quản lý. Mặt khác, việc cập nhật doanh thu, lỗ lãi từ hoạt động kinh doanh thường chỉ có thể tổng hợp vào cuối mỗi tháng hoặc cuối tuần. Tuy nhiên việc này khá mất thời gian và không dễ dàng gì tổng hợp trong một sớm một chiều. Điều này cản trở khá nhiều, đặc biệt là báo cho bạn tình trạng quán đang ra sao, có thất thoát hay hao hụt ở khâu nào hay không, những lỗ hổng nằm ở bộ phận gì. Điều đó phần mềm quản lý nhà hàng của chúng tôi tích hợp thêm tính năng quản lý từ xa để mọi công việc đều nằm trong sự kiểm soát của chủ nhà hàng.

Dự án “Phần mềm quản lý nhà hàng” được chọn làm đề tài cho dự án 1 của nhóm với mong muốn góp phần nhằm nâng cao chất lượng quản lý của nhà hàng cũng như trải nghiệm của khách hàng, là một địa chỉ đáng tin cậy cho người sử dụng. Đề tài được nghiên cứu dựa vào những tài liệu và thông tin thu thập được và tham khảo tài liệu của nhiều hoạt động danh mục khác để phát triển một hệ thống trợ giúp quản lý danh mục theo phương pháp hướng cấu trúc. Công cụ và môi trường để phát triển hệ thống là hệ quản trị cơ sở dữ liệu Microsoft SQL Server và ngôn ngữ lập trình Java. Chương trình được phát triển sẽ hỗ trợ rất tốt việc quản lý nhà hàng trực tuyến, đem đến sự tiện lợi, tiết kiệm được nhiều công sức và nguồn nhân lực, đáp ứng tốt nhất yêu cầu của khách hàng.

# CHƯƠNG I – GIỚI THIỆU ĐỀ TÀI

## Tìm hiểu “Xây dựng Phần mềm quản lý nhà hàng”

* Hầu hết tại các nhà hàng vào những giờ cao điểm số lượng khách vào quán thường rất đông, khách hàng vào hàng quán ăn uống thường là khách lẻ hoặc đi theo từng tốp nhỏ. Vì thế mà mô hình nhà hàng thường khó kiểm soát. Để có thể giải quyết được những điều đó, hệ thống phần mềm quản lý nhà hàng đã ra đời, đáp ứng nhu cầu quản lý của các chủ cửa hàng.
* Nhờ những tính năng ưu việt của phần mềm quản lý nhà hàng việc phục vụ khách hàng trở nên đơn giản hơn bao giờ hết. Ngay sau khi đầu bếp chuẩn bị xong đồ ăn, nhân viên phục vụ chỉ cần đưa đồ ăn ra cho khách hàng theo đúng bàn yêu cầu mà không cần mất công chạy đi chạy lại hỏi xem khách nào dùng món gì để đưa đồ ăn cho đúng.
* Với phần mềm này, ngay sau khi khách hàng gọi món, nhân viên phục vụ có thể có thể sử dụng hóa đơn tính tạm được tạo trên phần mềm. Sau khi khách gọi thanh toán, nhân viên chỉ việc đưa hóa đơn tính tạm cho khách hàng đối chiếu và thanh toán. Việc tạo hóa đơn ngay trên phần mềm quản lý nhà hàng theo phiếu gọi món sẽ giúp nhân viên không bỏ sót khách hàng chưa thanh toán. Thêm vào đó, phần mềm tạo hóa đơn tính tiền chính xác sẽ không xảy ra trường hợp tính thiếu hoặc thừa tiền, gây tâm lý ức chế cho khách hàng hoặc làm thất thoát cho quán ăn. Đó là lợi ích tuyệt vời của phần mềm quản lý nhà hàng trực tuyến.
* Dựa trên nhu cầu thực tế, nhóm chúng tôi lựa chọn đề tài “Phần mềm quản lý quán ăn”.

## Nền tảng công nghệ

* Ứng dụng quản lý: Sử dụng Java Swing, Java Object Oriented Programming, Java Data Access Object.

## Công cụ lập trình và công nghệ được sử dụng

* Hệ quản trị cơ sở dữ liệu: Microsoft SQL Server 2012.
* Công cụ lập trình: NetBeans.
* Ngôn ngữ lập trình: Java.
* Xây dựng giao diện front-end: Java Swing
* Xử lý back-end: Java

## Giới thiệu ngôn ngữ lập trình Java

Java là một trong những ngôn ngữ lập trình hướng đối tượng. Nó được sử dụng trong phát triển phần mềm, trang web, game hay ứng dụng trên các thiết bị di động.

Java được khởi đầu bởi James Gosling và bạn đồng nghiệp ở Sun MicroSystem năm 1991. Ban đầu Java được tạo ra nhằm mục đích viết phần mềm cho các sản phẩm gia dụng, và có tên là Oak. Java được phát hành năm 1994, đến năm 2010 được Oracle mua lại từ Sun MicroSystem.

Java được tạo ra với tiêu chí “Viết (code) một lần, thực thi khắp nơi” (Write Once, Run Anywhere – WORA). Chương trình phần mềm viết bằng Java có thể chạy trên mọi nền tảng (platform) khác nhau thông qua một môi trường thực thi với điều kiện có môi trường thực thi thích hợp hỗ trợ nền tảng đó .

## Giới thiệu hệ quản trị cơ sở dữ liệu Microsoft SQL Server

SQL Server là một hệ quản trị cơ sở dữ liệu quan hệ (Relational Database Management System (RDBMS) ) sử dụng câu lệnh SQL (Transact-SQL) để trao đổi dữ liệu giữa máy Client và máy cài SQL Server. Một RDBMS bao gồm databases, database engine và các ứng dụng dùng để quản lý dữ liệu và các bộ phận khác nhau trong RDBMS.

SQL Server được tối ưu để có thể chạy trên môi trường cơ sở dữ liệu rất lớn (Very Large Database Environment) lên đến Tera-Byte và có thể phục vụ cùng lúc cho hàng ngàn user. SQL Server có thể kết hợp “ăn ý” với các server khác như Microsoft Internet Information Server (IIS), E-Commerce Server, Proxy Server….

### Java Database Connectivity (JDBC)

Java Database Connectivity (JDBC) là một API được thiết kế dành cho ngôn ngữ lập trình Java hỗ trợ Java trong việc truy cập Cơ Sở Dữ Liệu (CSDL). Nó gồm có những phương thức thực hiện truy vấn và cập nhật CSDL gián tiếp qua Java. Nền tảng Java 2, Standard Edition, version 1.4 (J2SE) bao gồm JDBC 3.0 API mở rộng khả năng truy cập CSDL trong môi trường máy ảo Java.

# CHƯƠNG II – PHÂN TÍCH

## Hiện trạng thực tế

Kinh doanh ẩm thực luôn là lĩnh vực đầu tư "hái ra tiền" bởi nhu cầu ăn uống là vô cùng thiết yếu. Nếu quán có nhiều món ăn ngon, đặc sắc, chắc chắn sẽ có được lượng khách hàng đông đảo. Dù chỉ là một quán ăn nhỏ, doanh thu vẫn có thể đạt mức ổn định nếu bạn có phương pháp marketing và quản lý tối ưu. Một trong những giải pháp quản lý hiệu quả nhất cho các nhà hàng chính là áp dụng phần mềm quản lý bán hàng để nắm bắt được chính xác sự biến động doanh thu, món ăn nào bán chạy, chương trình khuyến mãi nào hiệu quả. Từ đó, bạn sẽ dễ dàng phân tích và đưa ra hướng đi đúng đắn trong tương lai cho nhà hàng của mình.

## Yêu cầu hệ thống

Phần mềm được viết ra với mong muốn xây dựng một phần mềm để giải quyết khó khăn trên.

* Yêu cầu chức năng nghiệp vụ
* Quản lý nhân viên
* Quản lý các loại thẻ khách hàng
* Quản lý khách hàng
* Quản lý các loại sản phẩm
* Quản lý sản phẩm
* Quản lý các đơn hàng
* Thống kê doanh thu các đơn hàng theo ngày, tháng, năm.
* Yêu cầu về bảo mật
* Tất cả mọi thành viên phải đăng nhập mới được sử dụng phần mềm
* Admin được phép thực hiện tất cả các chức năng
* Nhân viên của nhà hàng không được phép thêm, xóa, sửa dữ liệu cũng như không được xem thông tin về doanh thu.
* Yêu cầu về môi trường công nghệ
* Ứng dụng phải được thực hiện với công nghệ Swing và JDBC chạy trên mọi hệ điều hành với môi trường JDK tối thiểu 1.8
* 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:

A picture containing text

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 theo mã
* Sửa: Cập nhật dữ liệu đang xem
* Tìm kiếm: Tìm kiếm theo điều kiện
* Điều hướng: Di chuyển đến dữ liệu của bảng ghi chi tiết trước và sau
* 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 doanh thu từng ngày, tháng, năm phải có cấu trúc:
* Ngày/tháng/năm
* Tổng tiền
* Thống kê tính phổ biến của từng sản phẩm
* Tên sản phẩm
* Số lượng
* Chức năng đăng nhập và đăng xuất
* Đăng nhập: Để sử dụng phần mềm, bắt buộc phải đăng nhập
* Đăng xuất: Đăng nhập lại với người dùng khác
* 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 tất cả
* Nhân viên:
* Không được phép thực hiện các chức năng thêm, xóa, sửa trong các chức năng quản lý
* Không được phép xem chức năng thống kê doanh thu
* THÔNG TIN CÁC THỰC THỂ
* Nhân viên
* Mã nhân viên
* Mật khẩu
* Tên nhân viên
* Ngày sinh
* Giới tính
* Số điện thoại
* Email
* Loại
* Hình
* Chức vụ
* Địa chỉ
* Loại thẻ khách hàng
* Mã loại thẻ khách hàng
* Tên loại
* Thẻ khách hàng
* Mã khách hàng
* Tên khách hàng
* Số điện thoại
* Email
* Tích điểm
* Mã loại thẻ khách hàng
* Loại sản phẩm
* Mã loại sản phẩm
* Tên loại
* Sản phẩm
* Mã sản phẩm
* Mã loại sản phẩm
* Tên sản phẩm
* Số lượng
* Đơn giá
* Hình
* Ghi chú
* Đơn hàng
* Mã đơn hàng
* Mã khách hàng
* Tên khách hàng
* Ngày bán
* Tổng tiền
* Mã nhân viên
* Chi tiết đơn hàng
* Mã đơn hàng
* Mã sản phẩm
* Tên sản phẩm
* Số lượng
* Đơn giá
* Giảm giá
* Thành tiền
* Mô tả

# CHƯƠNG III THIẾT KẾ

## Mô hình triển khai

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

Cần một máy cài SQL Server 2008+. Máy này cần hệ điều hành Window XP trở lên.

Máy của hệ thống cửa hàng cần cài phần mềm Ressys. Các máy cần JDK 1.8+ với hệ điều hành bất kỳ.

## Thiết kế CSDL

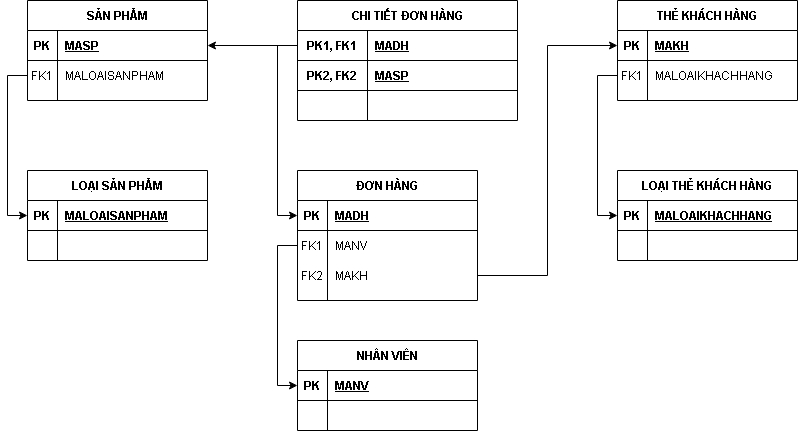
### Sơ đồ quan hệ thực thể

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

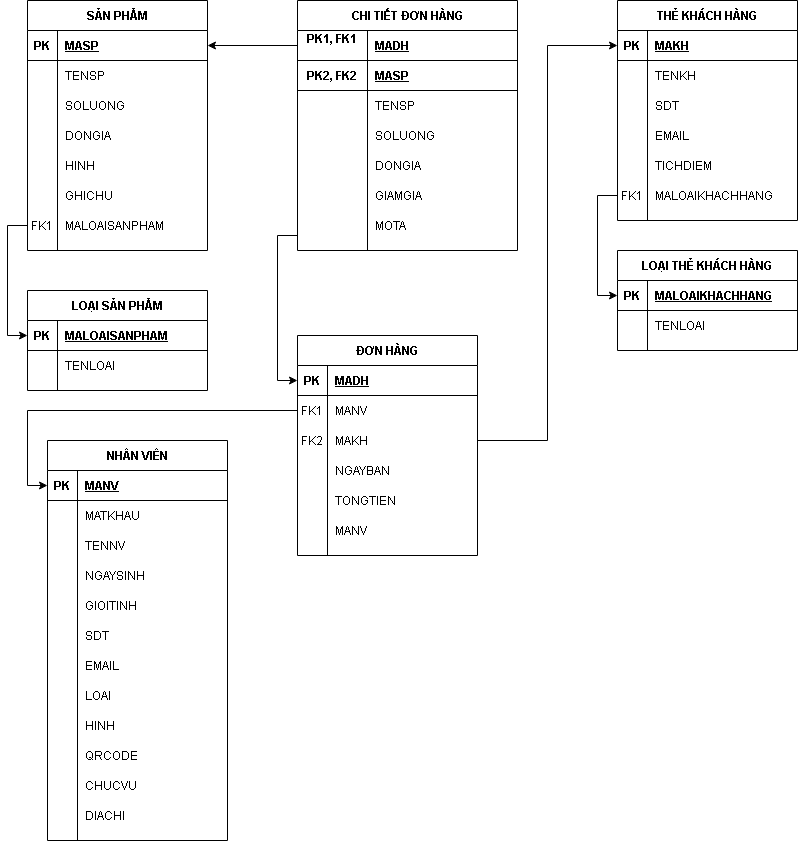
* Nhân viên: Quản lý thông tin của nhân viên.
* Loại thẻ khách hàng: Quản lý các loại thẻ khách hàng.
* Thẻ khách hàng: Quản lý thông tin của khách hàng. Dựa vào hóa đơn mà khách hàng được tích điểm. Từ đó, dựa vào số điểm mà phân loại từng loại thẻ khách hàng.
* Loại sản phẩm: Phân loại từng sản phẩm. Mỗi loại sản phẩm có nhiều sản phẩm khác nhau.
* Sản phẩm: Quản lý các sản phẩm có trong cửa hàng.
* Đơn hàng: Quản lý các đơn hàng được bán trong ngày. Một đơn hàng gồm nhiều đơn hàng chi tiết.
* Chi tiết đơn hàng: Quản lý các đơn hàng chi tiết.

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

#### ERD Diagram level 1



#### ERD Diagram level 2



### Thiết kế chi tiết các thực thể

#### Nhân viên

Bảng NHANVIEN lưu thông tin nhân viên của nhà hàng có cấu trúc như sau

|  |  |  |  |
| --- | --- | --- | --- |
| **Tên cột** | **Kiểu** | **Ràng buộc** | **Ghi chú** |
| MANV | VARCHAR(10) | PK, NOT NULL | Mã nhân viên |
| MATKHAU | VARCHAR(50) | NOT NULL | Mật khẩu |
| TENNV | NVARCHAR(50) | NOT NULL | Tên nhân viên |
| NGAYSINH | DATE | NOT NULL | Ngày sinh |
| GIOITINH | BIT | NOT NULL | Giới tính |
| SDT | VARCHAR(20) | NOT NULL | Số điện thoại |
| EMAIL | VARCHAR(50) | NOT NULL | Email |
| LOAI | BIT | DEFAULT 0, NOT NULL | Loại nhân viên. Part time hoặc full time |
| HINH | VARCHAR(200) | NULL | Hình |
| QRCODE | VARCHAR(200) | NOT NULL | Mã QR dùng để đăng nhập |
| CHUCVU | BIT | DEFAULT 0, NOT NULL | Chức vụ. Admin hoặc nhân viên |
| DIACHI | NVARCHAR(255) | NOT NULL | Địa chỉ |

#### Loại thẻ khách hàng

Bảng LOAITHEKHACHHANG lưu thông tin của từng loại thẻ khách hàng có cấu trúc như sau

|  |  |  |  |
| --- | --- | --- | --- |
| **Tên cột** | **Kiểu** | **Ràng buộc** | **Ghi chú** |
| MALOAITHEKHACHHANG | INT | PK, IDENTITY | Mã loại thẻ khách hàng |
| TENLOAI | NVARCHAR(50) | NOT NULL | Tên loại thẻ khách hàng |

#### Thẻ khách hàng

Bảng THEKHACHHANG lưu thông tin của từng khách hàng có cấu trúc như sau

|  |  |  |  |
| --- | --- | --- | --- |
| **Tên cột** | **Kiểu** | **Ràng buộc** | **Ghi chú** |
| MAKH | INT | PK, IDENTITY | Mã khách hàng |
| TENKH | NVARCHAR(50) | NOT NULL | Tên khách hàng |
| SDT | VARCHAR(20) | NULL | Số điện thoại |
| EMAIL | VARCHAR(255) | NULL | Email |
| TICHDIEM | INT | NULL, DEFAULT 0 | Tích điểm |
| MALOAIKHACHHANG | INT | FK, NOT NULL | Mã loại thẻ khách hàng |

#### Loại sản phẩm

Bảng LOAISANPHAM lưu thông tin của từng loại sản phẩm có cấu trúc như sau

|  |  |  |  |
| --- | --- | --- | --- |
| **Tên cột** | **Kiểu** | **Ràng buộc** | **Ghi chú** |
| MALOAISANPHAM | NCHAR(5) | PK, NOT NULL | Mã loại sản phẩm |
| TENLOAI | NVARCHAR(50) | NOT NULL | Tên loại sản phẩm |

#### Sản phẩm

Bảng SANPHAM lưu thông tin của từng sản phẩm có cấu trúc như sau

|  |  |  |  |
| --- | --- | --- | --- |
| **Tên cột** | **Kiểu** | **Ràng buộc** | **Ghi chú** |
| MASP | INT | PK, IDENTITY | Mã sản phẩm |
| MALOAISANPHAM | NCHAR(5) | FK, NOT NULL | Mã loại sản phẩm |
| TENSP | NVARCHAR(50) | NOT NULL | Tên sản phẩm |
| SOLUONG | INT | DEFAULT 0, NOT NULL | Số lượng |
| DONGIA | INT | DEFEULT 0, NOT NULL | Đơn giá |
| HINH | VARCHAR(200) | VARCHAR(20) | Hình |
| GHICHU | NVARCHAR(255) | NVARCHAR(255), NULL | Ghi chú |

#### Đơn hàng

Bảng DONHANG lưu thông tin của từng đơn hàng có cấu trúc như sau

|  |  |  |  |
| --- | --- | --- | --- |
| **Tên cột** | **Kiểu** | **Ràng buộc** | **Ghi chú** |
| MADH | VARCHAR(10) | PK, NOT NULL | Mã đơn hàng |
| MAKH | INT | NULL | Mã khách hàng |
| TENKH | NVARCHAR(50) | NULL | Tên khách hàng |
| NGAYBAN | DATE | NOT NULL | Ngày bán |
| TONGTIEN | INT | NOT NULL | Tổng tiền |
| MANV | VARCHAR(10) | FK, NOT NULL | Mã nhân viên |

#### Chi tiết đơn hàng

Bảng CHITIETDONHANG lưu thông tin của chi tiết từng đơn hàng có cấu trúc như sau

|  |  |  |  |
| --- | --- | --- | --- |
| **Tên cột** | **Kiểu** | **Ràng buộc** | **Ghi chú** |
| MACTDH | INT | PK, IDENTITY, NOT NULL | Mã chi tiết đơn hàng |
| MADH | VARCHAR(100) | NOT NULL | Mã đơn hàng |
| MASP | INT | PK, FK, NOT NULL | Mã sản phẩm |
| TENSP | NVARCHAR(50) | NOT NULL | Tên sản phẩm |
| SOLUONG | INT | NOT NULL | Số lượng |
| DONGIA | INT | NOT NULL | Đơn giá |
| GIAMGIA | INT | NOT NULL | Giảm giá |
| THANHTIEN | INT | NOT NULL | Thành tiền |
| MOTA | NVARCHAR(255) | NULL | Mô tả |

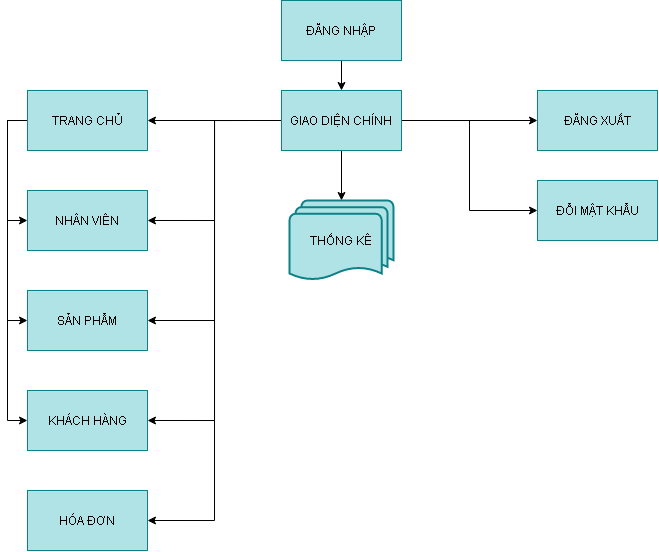
## Thiết kế giao diện

### 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.

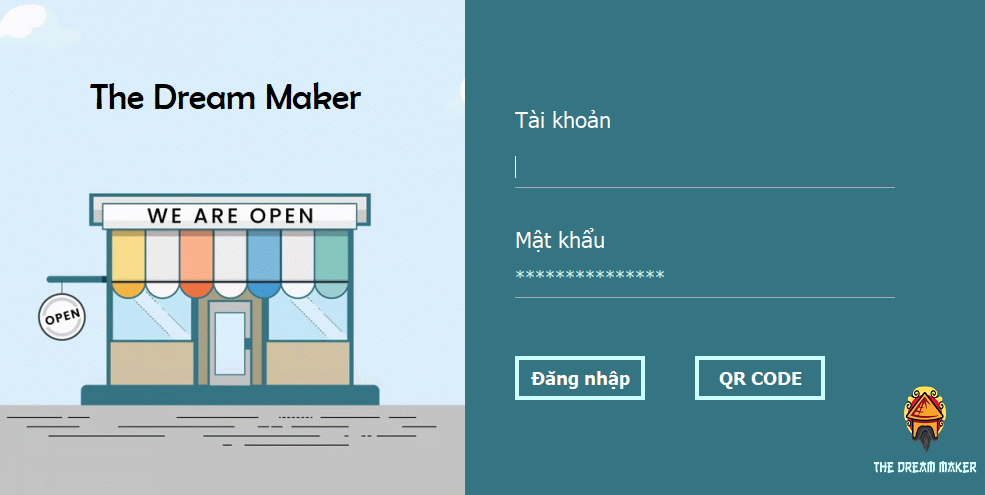
Theo yêu cầu thì mọi nhân viên phải đăng nhập trước khi sử dụng ứng dụng nên form đăng nhập xuất hiện trước để yêu cầu đăng nhập.

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

Thiết kế giao diện cho các chức năng nghiệp vụ

|  |  |  |
| --- | --- | --- |
| **Control** | **Prefix** | **Mô tả** |
| JLabel | lbl | Nhãn |
| JButton | btn | Nút |
| JTextField, JPasswordField | txt | Các loại ô nhập |
| JRadioButton | rdo | Hộp chọn |
| JCombobox | cbo | Hộp chọn đổ xuống |
| JPanel | pnl | Khung chứa |
| JTabbedPane | tab |  |
| JTable | tbl | Bảng |
| JSeparator |  |  |
| JFileChooser |  |  |
| JDateChooser |  |  |

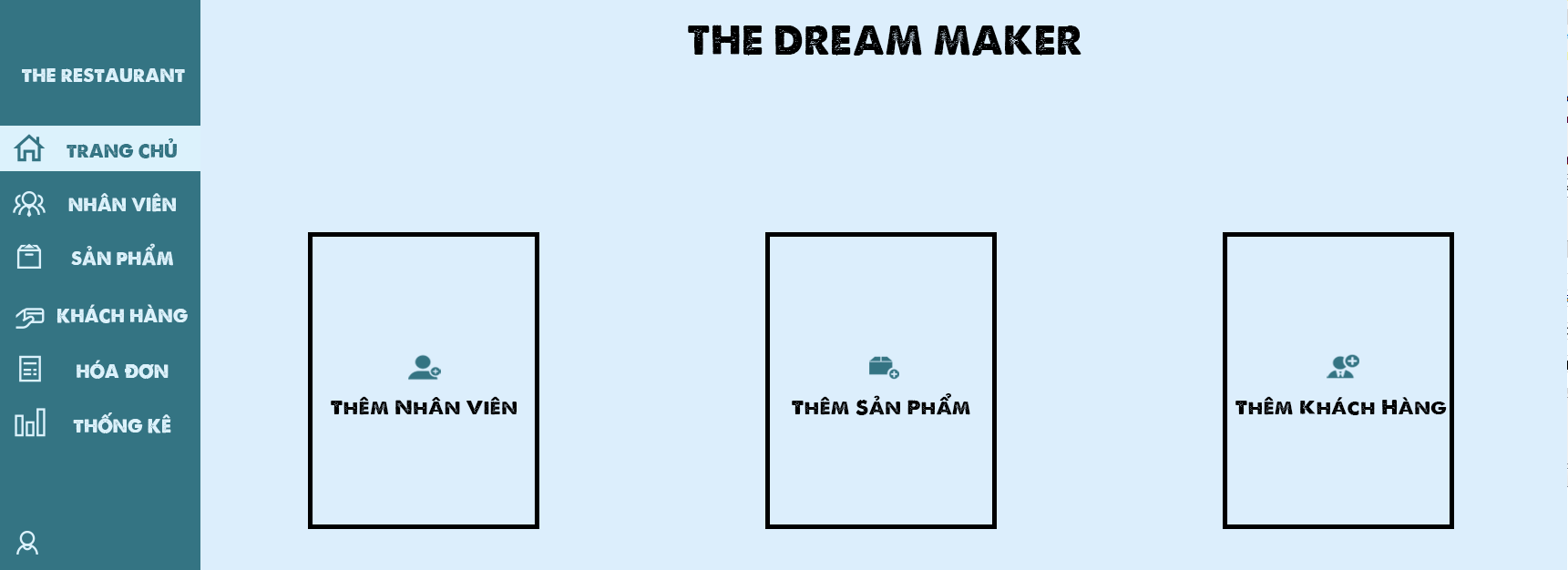
#### Đăng nhập



Các thành phần giao diện

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Componel** | **Kiểu** | **Thuộc tính** |
| 1 | SignInFrame | JFrame | Layout: AbsoluteLayout |
| 2 | pnlBgr | JPanel | Layout: AbsoluteLayout |
| 3 | lblSlogan | JLabel | Text: THE DREAM MAKER  Font: Berlin Sans FB 36 Plain  Foreground: [0,0,0] |
| 4 | lblBgrLeft | JLabel | Icon: bgrleft.gif |
| 5 | lblBgrRight | JLabel | Icon: bgrright.png |
| 6 | lblTaiKhoan | JLabel | Text: Tài khoản  Font: Tahoma 22 Plain  Foreground: [255,255,255] |
| 7 | txtNhapTaiKhoan | JTextField | Font: Tahoma 18 Plain  Foreground: [204,255,255]  Background: [52,116,131]  Opaque: Null  Border: No Border |
| 8 | jspTaiKhoan | Jseparator |  |
| 9 | lblMatKhau | JLable | Text: Mật khẩu  Font: Tahoma 22 Plain  Foreground: [255,255,255] |
| 10 | txtMatKhau | JPasswordField | Font: Tahoma 18 Plain  Foreground: [204,255,255]  Background: [52,116,131]  Opaque: Null  Border: No Border |
| 11 | jspMatKhau | JSeparator |  |
| 12 | pnlDangNhap | JPanel | Background: [52,116,131]  Border: No Border |
| 13 | lblDangNhap | JLabel | Text: Đăng nhập  Font: Tahoma 18 Bold  Background: [102,51,0]  Foreground: [255,255,255]  Border: Line Border  Color: [204,255,255] |
| 14 | lblQRCode | JLabel | Text: QR CODE  Font: Tahoma 18 Bold  Background: [102,51,0]  Foreground: [255,255,255]  Border: Line Border  Color: [204,255,255] |
| 15 | lblLogo | JLabel | Icon: logo.png |

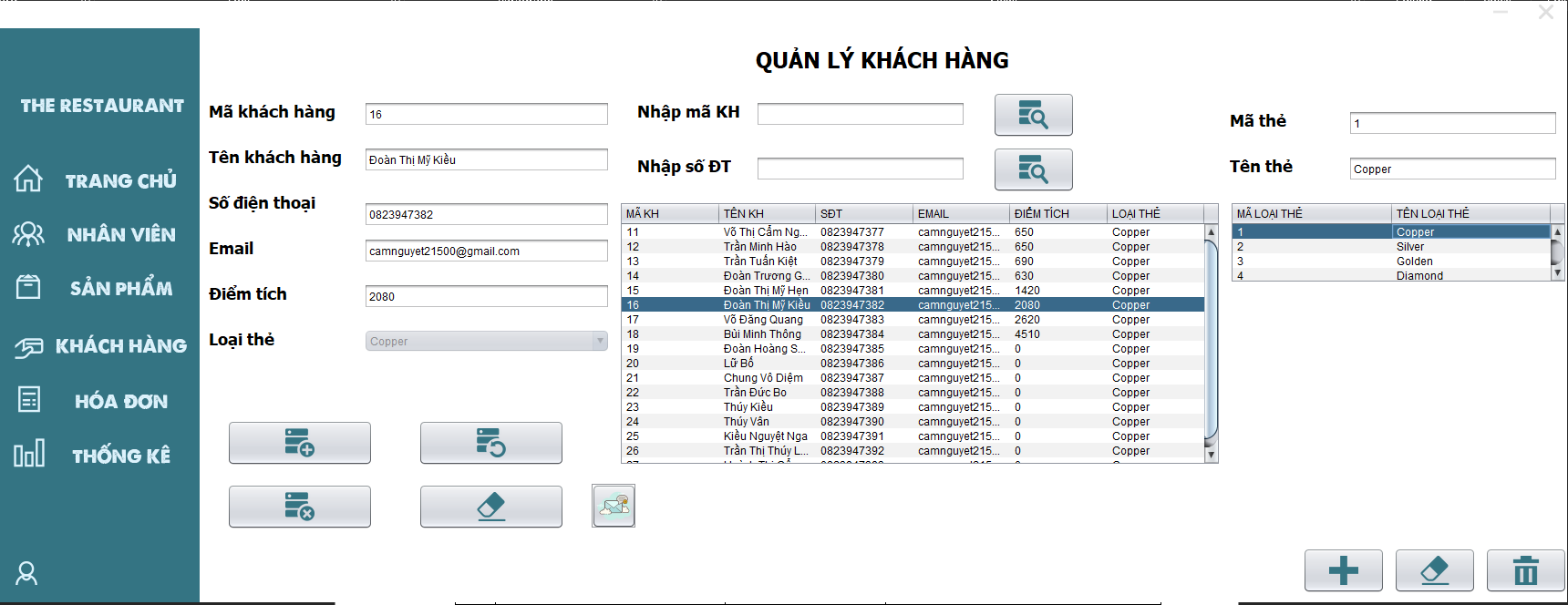
#### Giao diện chính



Các thành phần giao diện

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Componel** | **Kiểu** | **Thuộc tính** |
| 1 | MainFrame | JFrame | Layout: AbsoluteLayout |
| 2 | pnlHeader | JPanel | Layout: AbsoluteLayout |
| 3 | btnMinimize | JButton | Background: [255,255,255]  Icon: Minus.png |
| 4 | btnExit | JButton | Background: [255,255,255]  Icon: Close.png |
| 5 | lblTrangChu | JLabel |  |
| 6 | lblNhanVien | JLabel |  |
| 7 | lblSanPham | JLabel |  |
| 8 | lblKhachHang | JLabel |  |
| 9 | lblHoaDon | JLabel |  |
| 10 | lblThongKe | JLabel |  |
| 11 | lblUser | JLabel |  |
| 12 | lblLogout | JLabel | Icon: logout.png |
| 13 | lblPanel\_Main | JLabel | Icon: panel\_main.png |
| 14 | Desktop | JDesktopPane |  |
| 15 | lblThemNV | JLabel |  |
| 16 | lblThemSP | JLabel |  |
| 17 | lnlThemKH | JLabel |  |
| 18 | lblBackground | JLabel | Icon: bg.png |

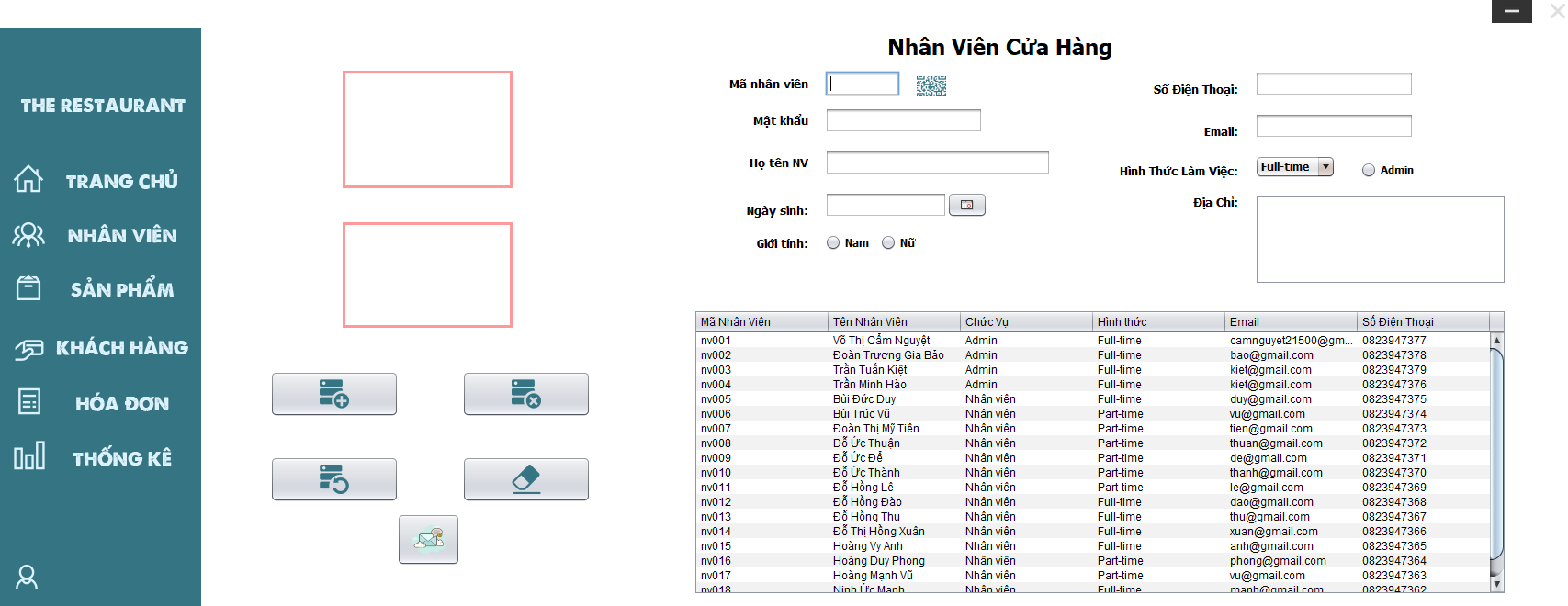
#### Khách hàng



Các thành phần giao diện

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Componel** | **Kiểu** | **Thuộc tính** |
| 1 | KhachHangJInternalFrame | JInternalFrame |  |
| 2 | pnlMain | JPanel | Layout: AbsoluteLayout |
| 3 | lblTieuDe | JLabel | Text: Quản Lý Khách Hàng  Font: Tahoma 24 Bold  Foreground: [0,0,0] |
| 4 | lblMaKH | JLable | Text: Mã khách hàng  Font: Tahoma 18 Bold  Foreground: [0,0,0] |
| 5 | lblTenKH | JLabel | Text: Tên khách hàng  Font: Tahoma 18 Bold  Foreground: [0,0,0] |
| 6 | lblSDT | JLabel | Text: Số điện thoại  Font: Tahoma 18 Bold  Foreground: [0,0,0] |
| 7 | lblEmail | JLabel | Text: Email  Font: Tahoma 18 Bold  Foreground: [0,0,0] |
| 8 | lblDiemTich | JLabel | Text: Điểm tích  Font: Tahoma 18 Bold  Foreground: [0,0,0] |
| 9 | lblLoaiThe | JLabel | Text: Loại thẻ  Font: Tahoma 18 Bold  Foreground: [0,0,0] |
| 10 | txtMaKH | JTextField |  |
| 11 | txtTenKH | JTextField |  |
| 12 | txtSDT | JTextField |  |
| 13 | txtEmail | JTextField |  |
| 14 | txtDiemTich | JTextField |  |
| 15 | cboLoaiThe | JComboBox | Model:  Font: Tahoma 13 Bold  Foreground: [0,0,0] |
| 16 | lblTimMaKH | JLabel | Text: Nhập mã KH  Font: Tahoma 18 Bold  Foreground: [0,0,0] |
| 17 | lblNhapSDT | JLabel | Text: Nhập số ĐT  Font: Tahoma 18 Bold  Foreground: [0,0,0] |
| 18 | txtTimMaKH | JTextField |  |
| 19 | txtNhapSoDT | JTextField |  |
| 20 | lblMaThe | JLabel | Text: Mã thẻ  Font: Tahoma 18 Bold  Foreground: [0,0,0] |
| 21 | lblTenThe | JLabel | Text: Tên thẻ  Font: Tahoma 18 Bold  Foreground: [0,0,0] |
| 22 | txtMaThe | JTextField |  |
| 23 | txtTenThe | JTextField |  |
| 24 | tblGridView | JTable |  |
| 25 | tblLoaiThe | JTable |  |
| 26 | btnThem | JButton | Icon |
| 27 | btnSua | JButton | Icon |
| 28 | btnXoa | JButton | Icon |
| 29 | btnLamMoiKH | JButton | Icon |
| 30 | btnTimMa | JButton | Icon |
| 31 | btnTimSDT | JButton | Icon |
| 32 | btnSendMail | JButton | Icon |
| 33 | btnThemLoaiThe | JButton | Icon |
| 34 | btnLamMoiLT | JButton | Icon |
| 35 | btnXoaLoaiThe | JButton | Icon |

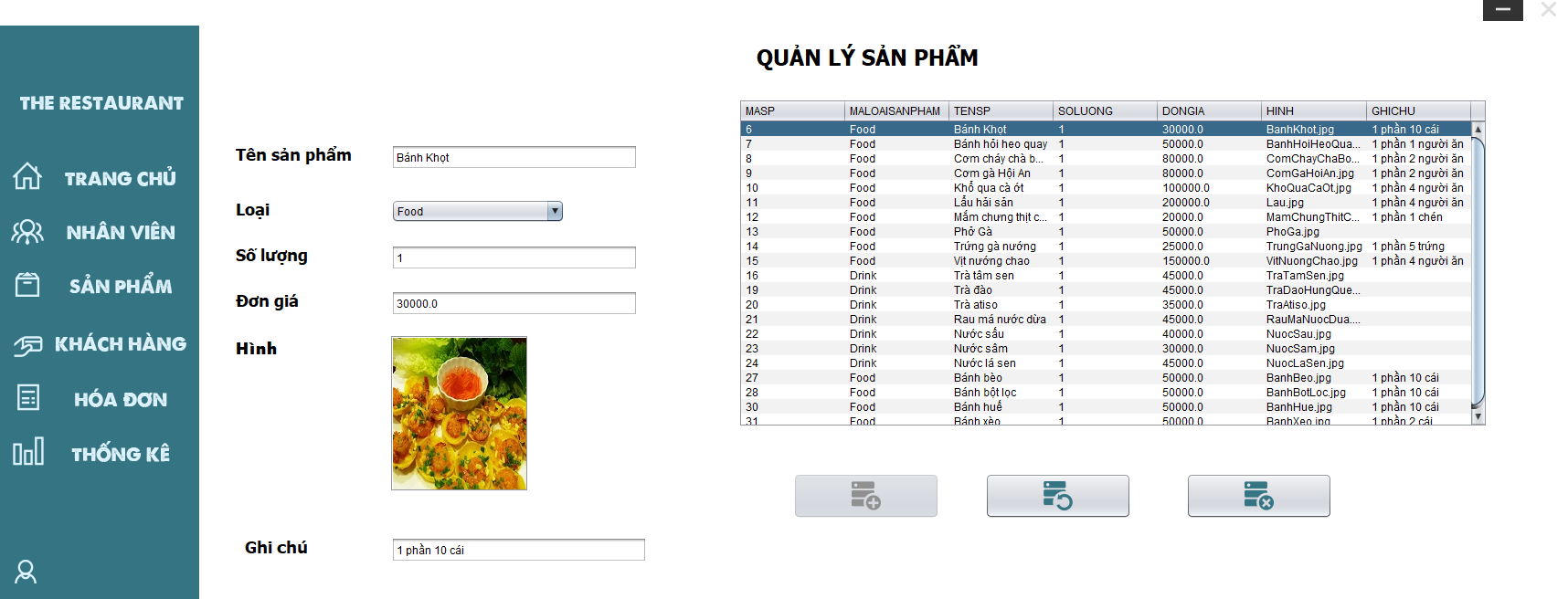
#### Nhân viên



Các thành phần giao diện

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Componel** | **Kiểu** | **Thuộc tính** |
| 1 | NhanVienJInternalFrame | JInternalFrame |  |
| 2 | pnlBgr | JPanel | Layout: AbsoluteLayout |
| 3 | pnlBgrRight | JPanel | Background: [255,255,255] |
| 4 | lblQRCode | JLabel | Border: LineBorder  Color: [255,153,153] |
| 5 | lblImg | JLabel | Border: LineBorder  Color: [255,153,153] |
| 6 | lblMaNV | JLabel | Text: Mã nhân viên  Font: Tahoma 13 Bold  Foreground: [0,0,0] |
| 7 | txtMaNV | JTextField |  |
| 8 | lblMatKhau | JLabel | Text: Mật khẩu  Font: Tahoma 13 Bold  Foreground: [0,0,0] |
| 9 | txtPass | JPasswordField |  |
| 10 | lblHoTenNV | JLabel | Text: Họ tên NV  Font: Tahoma 13 Bold  Foreground: [0,0,0] |
| 11 | txtName | JTextField |  |
| 12 | lblBirthDay | JLabel | Text: Ngày sinh  Font: Tahoma 13 Bold  Foreground: [0,0,0] |
| 13 | jdcDate | JDateChooser |  |
| 14 | lblSex | JLabel | Text: Giới tính  Font: Tahoma 13 Bold  Foreground: [0,0,0] |
| 15 | rdoNam | JRadioButton | Text: Nam  Font: Tahoma 13 Bold  Foreground: [0,0,0] |
| 16 | rdoNu | JRadioButton | Text: Nữ  Font: Tahoma 13 Bold  Foreground: [0,0,0] |
| 17 | lblSoDienThoai | JLabel | Text: Số điện thoại  Font: Tahoma 13 Bold  Foreground: [0,0,0] |
| 18 | txtSDT | JTextField |  |
| 19 | lblEmailNV | JLabel | Text: Email  Font: Tahoma 13 Bold  Foreground: [0,0,0] |
| 20 | txtEmail | JTextField |  |
| 21 | lblHinhThuc | JLabel | Text: Hình thức làm việc  Font: Tahoma 13 Bold  Foreground: [0,0,0] |
| 22 | cbbLoai | JComboBox | Model: Full time, Part time  Font: Tahoma 13 Bold  Foreground: [0,0,0] |
| 23 | rdoAdmin | JRadioButton | Text: Admin  Font: Tahoma 13 Bold  Foreground: [0,0,0] |
| 24 | lblDiaChi | JLabel | Text: Địa chỉ  Font: Tahoma 13 Bold  Foreground: [0,0,0] |
| 25 | txtDiaChi | JTextArea |  |
| 26 | tblNhanVien | JTable |  |
| 27 | btnTao | JButton | Icon |
| 28 | btnXoa | JButton | Icon |
| 29 | btnSua | JButton | Icon |
| 30 | btnSendGmail | JButton | Icon |
| 31 | btnMoi | JButton | Icon |

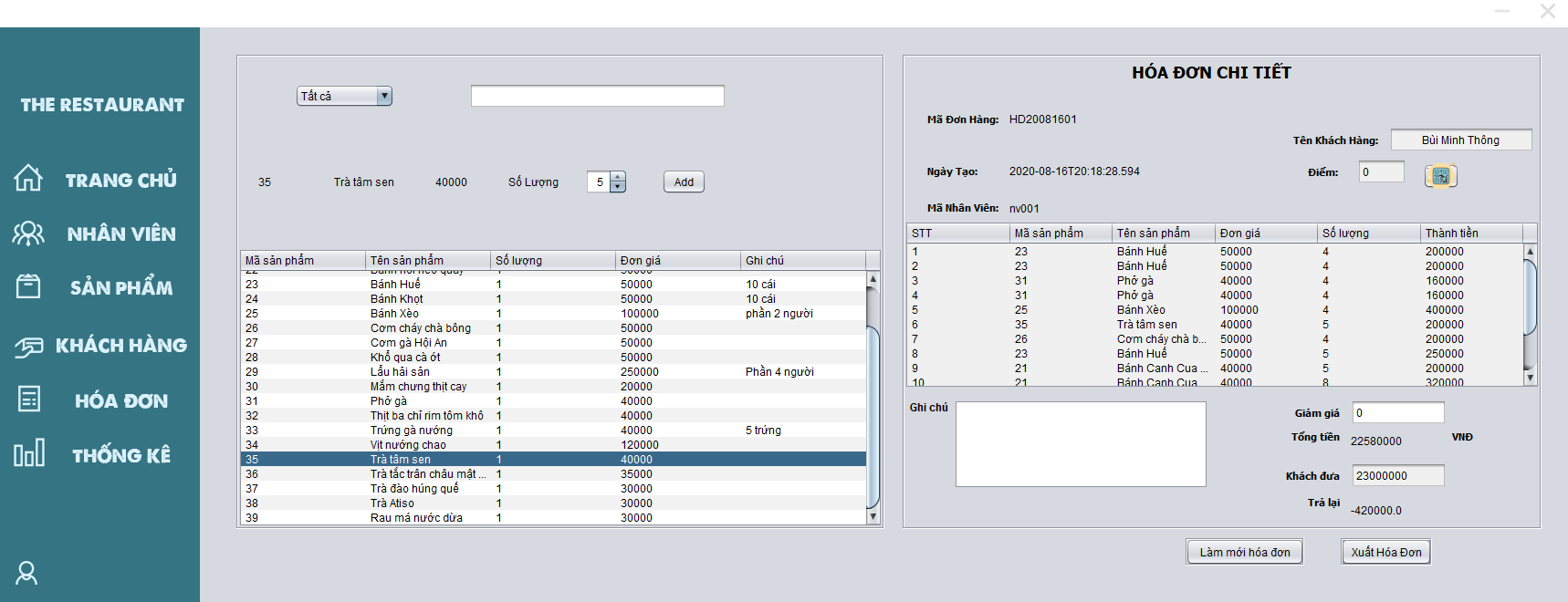
#### Sản phẩm



Các thành phần giao diện

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Componel** | **Kiểu** | **Thuộc tính** |
| 1 | SanPhamJInternalFrame | JInternalFrame |  |
| 2 | pnlBgr | JPanel | Layout: AbsoluteLayout |
| 3 | lblTieuDe | JLabel |  |
| 4 | lblTenSP | JPanel | Text: Tên sản phẩm  Font: Tahoma 13 Bold  Foreground: [0,0,0] |
| 5 | txtTenSP | JTextField |  |
| 6 | lblLoai | JLabel | Text: Loại  Font: Tahoma 13 Bold  Foreground: [0,0,0] |
| 7 | cboLoai | JCombobox |  |
| 8 | lblSoLuong | JLabel | Text: Số lượng  Font: Tahoma 13 Bold  Foreground: [0,0,0] |
| 9 | txtSoLuong | JTextField |  |
| 10 | lblDonGia | JLabel | Text: Đơn giá  Font: Tahoma 13 Bold  Foreground: [0,0,0] |
| 11 | txtDonGia | JTextField |  |
| 12 | lblHinh | JLabel | Text: Hình  Font: Tahoma 13 Bold  Foreground: [0,0,0] |
| 13 | lblHinhSP | JLabel |  |
| 14 | lblGhiChu | JLabel | Text: Ghi chú  Font: Tahoma 13 Bold  Foreground: [0,0,0] |
| 15 | txtGhiChu | JTextField |  |
| 16 | tblGridView | JTable |  |
| 17 | btnThem | JButton | Icon |
| 18 | btnSua | JButton | Icon |
| 19 | btnXoa | JButton | Icon |

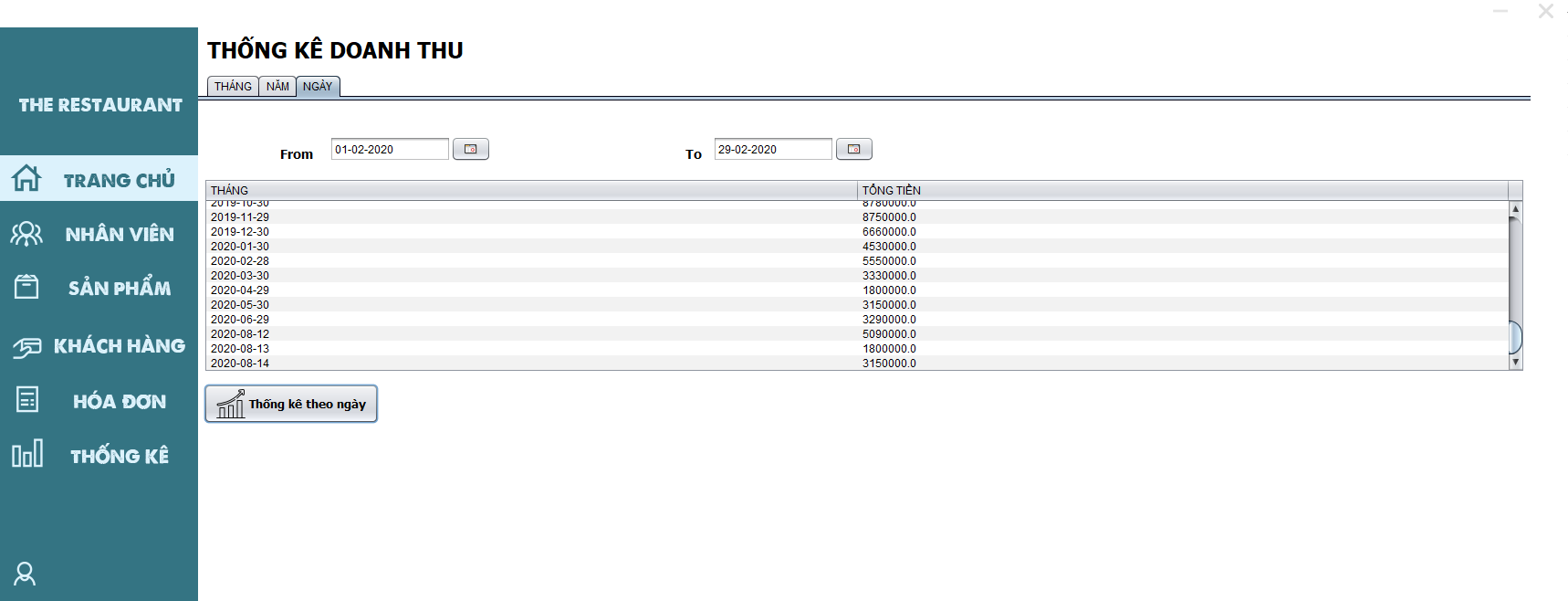
#### Hóa đơn



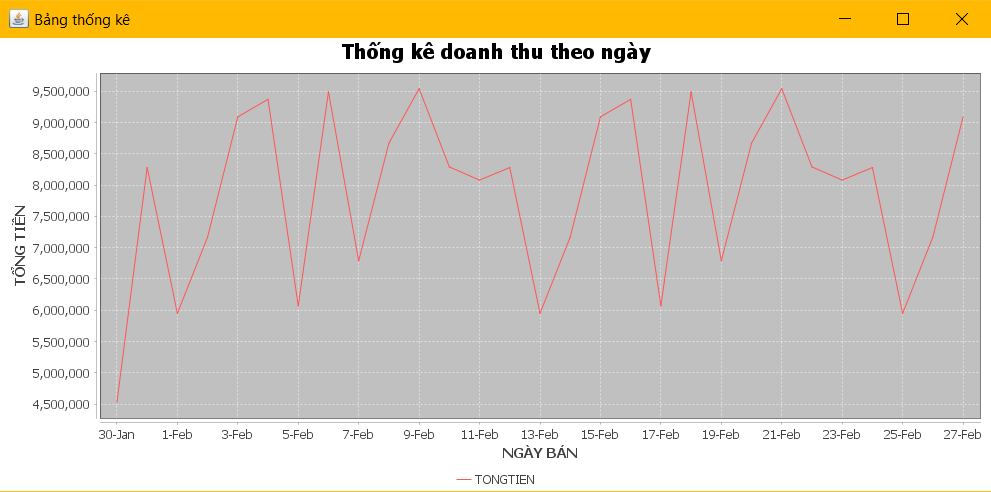
Các thành phần giao diện

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Componel** | **Kiểu** | **Thuộc tính** |
| 1 | HoaDonJInternalFrame | JInternalFrame |  |
| 2 | pnlMain | JPanel | Layout: AbsoluteLayout |
| 3 | pnlDonGiaTam | JPanel |  |
| 4 | cboLoai | JComboBox |  |
| 5 | txtSearch | JTextField |  |
| 6 | lblMaMonAn | JLabel | Text: Mã món ăn  Font: Tahoma 13 Bold  Foreground: [0,0,0] |
| 7 | lblTenMonAn | JLabel | Text: Tên Món Ăn  Font: Tahoma 13 Bold  Foreground: [0,0,0] |
| 8 | lblDonGia | JLabel | Text: Đơn giá  Font: Tahoma 13 Bold  Foreground: [0,0,0] |
| 9 | lblSoLuong | JLabel | Text: Số lượng  Font: Tahoma 13 Bold  Foreground: [0,0,0] |
| 10 | spnSoLuong | JSpinner |  |
| 11 | btnThem | JButton | Text: Add |
| 12 | pnlHDCT | JPanel |  |
| 13 | lblTitle | JLabel | Text: Hóa đơn chi tiết  Font: Tahoma 18 Bold  Foreground: [0,0,0] |
| 14 | lblMaDonHang | JLabel | Text: Mã đơn hàng  Font: Tahoma 13 Bold  Foreground: [0,0,0] |
| 15 | lblMaDH | JLabel | Font: Tahoma 13 Bold  Foreground: [0,0,0] |
| 16 | lblNgayTaoDH | JLabel | Text: Ngày tạo  Font: Tahoma 13 Bold  Foreground: [0,0,0] |
| 17 | lblNgayTao | JLabel | Font: Tahoma 13 Bold  Foreground: [0,0,0] |
| 18 | lblMaNVDH | JLabel | Text: Mã nhân viên  Font: Tahoma 13 Bold  Foreground: [0,0,0] |
| 19 | lblMaNV | JLabel | Font: Tahoma 13 Bold  Foreground: [0,0,0] |
| 20 | lblTenKHDH | JLabel | Text: Tên khách hàng  Font: Tahoma 13 Bold  Foreground: [0,0,0] |
| 21 | lblTenKH | JLabel | Font: Tahoma 13 Bold  Foreground: [0,0,0] |
| 22 | lblDiemDH | JLabel | Text: Điểm  Font: Tahoma 13 Bold  Foreground: [0,0,0] |
| 23 | lblDiem | JLabel | Font: Tahoma 13 Bold  Foreground: [0,0,0] |
| 24 | btnQRKH | JButton | Icon |
| 25 | tblHoaDon | JTable |  |
| 26 | lblGhiChu | JLabel | Text: Ghi chú  Font: Tahoma 13 Bold  Foreground: [0,0,0] |
| 27 | txaGhiChu | JTextArea | Font: Tahoma 13 Bold  Foreground: [0,0,0] |
| 28 | lblGiamGia | JLabel | Text: Giảm giá  Font: Tahoma 13 Bold  Foreground: [0,0,0] |
| 29 | txtGiamGia | JTextField |  |
| 30 | lblTongTienDH | JLabel | Text: Tổng tiền  Font: Tahoma 13 Bold  Foreground: [0,0,0] |
| 31 | lblTongTien | JLabel | Font: Tahoma 13 Bold  Foreground: [0,0,0] |
| 32 | lblDonVi | JLabel | Text: VNĐ  Font: Tahoma 13 Bold  Foreground: [0,0,0] |
| 33 | lblKhachDua | JLabel | Text: Khách đưa  Font: Tahoma 13 Bold  Foreground: [0,0,0] |
| 34 | txtKhachDua | JTextField | Font: Tahoma 13 Bold  Foreground: [0,0,0] |
| 35 | lblTraLai | JLabel | Text: Trả lại  Font: Tahoma 13 Bold  Foreground: [0,0,0] |
| 36 | txtTraLai | JTextField |  |
| 37 | btnXuatHD | JButton | Text: Xuất hóa đơn |
| 38 | btnLamMoiHD | JButton | Text: Làm mới hóa đơn |

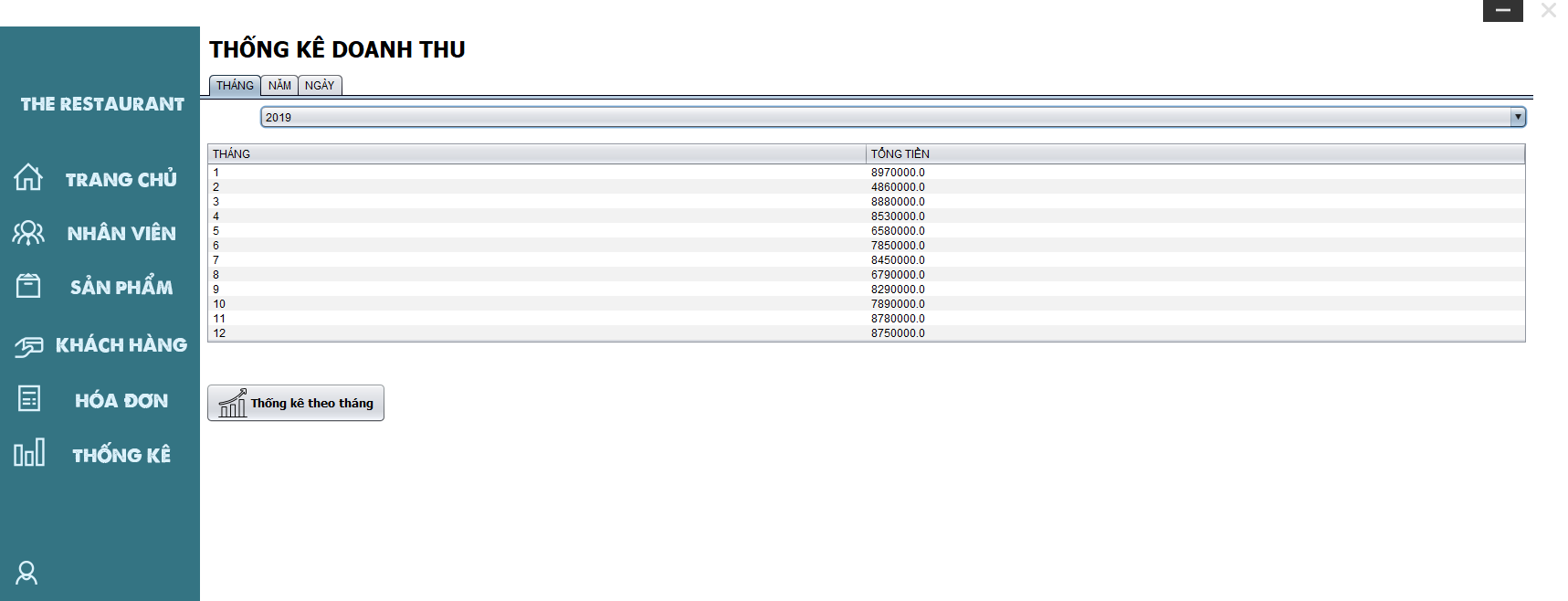
#### Thống kê



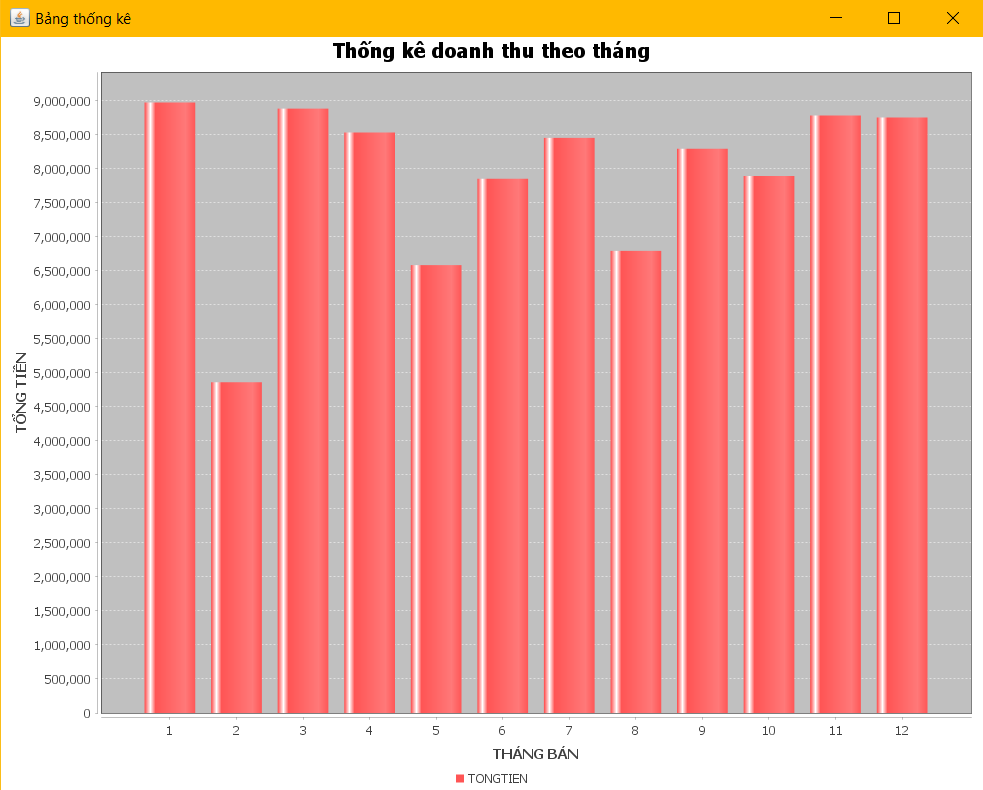
*Thống kê doanh thu theo ngày*



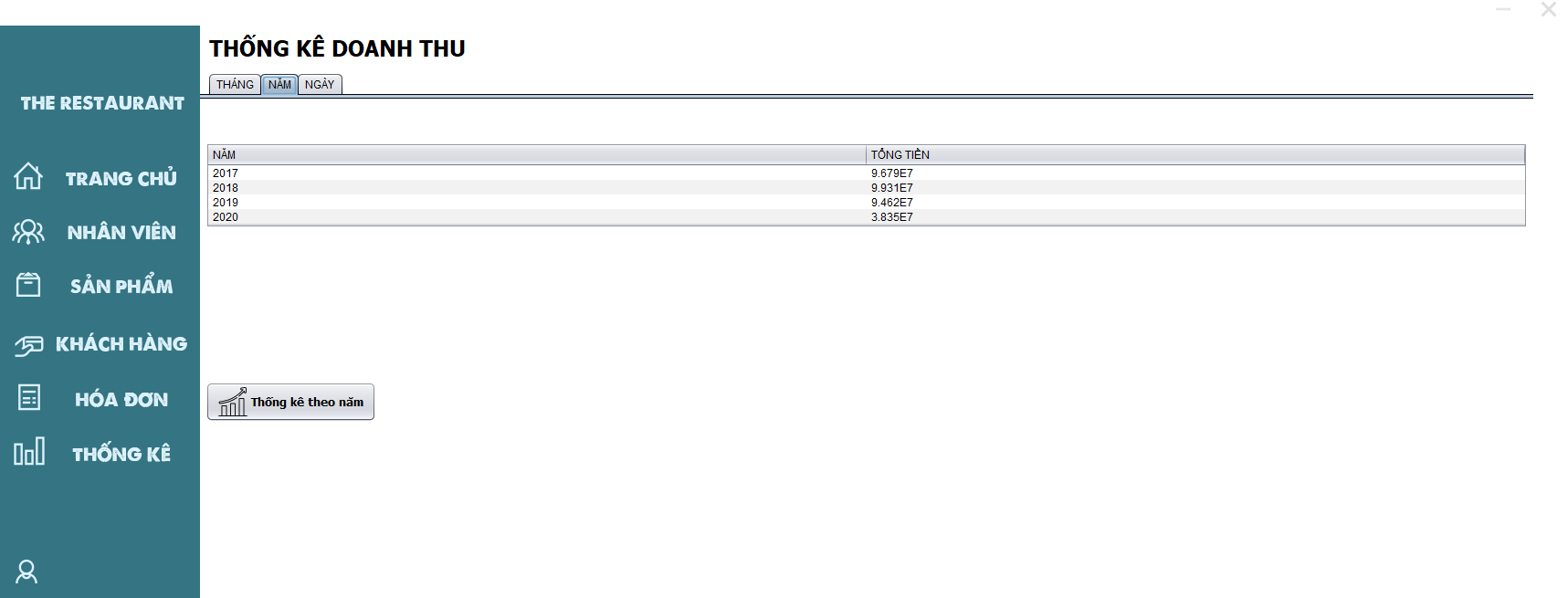
*Biểu đồ cột thể hiện doanh thu ngày*



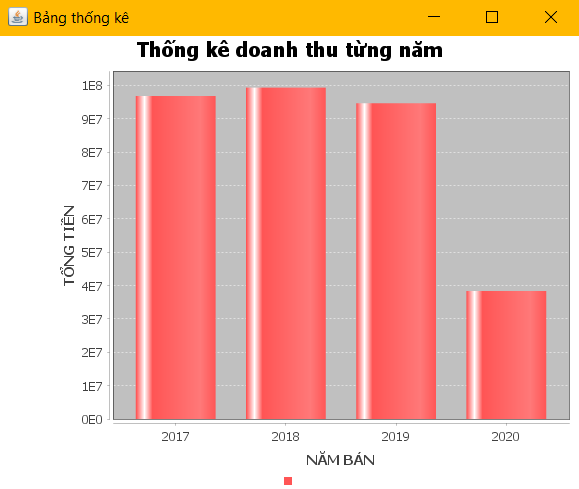
*Thống kê doanh thu theo tháng*



*Biểu đồ cột thể hiện doanh thu tháng*



*Thống kê doanh thu năm*



*Biểu đồ cột thể hiện doanh thu năm*

Các thành phần giao diện

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Componel** | **Kiểu** | **Thuộc tính** |
| 1 | ThongKeJInternalFrame | JInternalFrame |  |
| 2 | pnlMain | JPanel | Layout: AbsoluteLayout |
| 3 | lblTitle | JLabel | Text: Thống kê doanh thu  Font: Tahoma 24 Bold  Foreground: [0,0,0] |
| 4 | Tabs | JTabbedPane |  |
| 5 | pnlThang | JPanel |  |
| 6 | tblTKDTThang | JTable |  |
| 7 | btnTKThang | JButton | Text: Thống kê theo tháng |
| 8 | cboThang | JComboBox |  |
| 9 | lblThang | JLabel | Text: Tháng  Font: Tahoma 13 Plain  Foreground: [255,255,255] |
| 10 | pnlNam | JPanel |  |
| 11 | tblTKDTNam | JTable |  |
| 12 | btnTKNam | JButton | Text: Thống kê theo năm |
| 13 | pnlDoanhThu | JPanel |  |
| 14 | pnlNgay | JPanel |  |
| 15 | tblTKDTNgay | JTable |  |
| 16 | btnTKNgay | JButton | Text: Thống kê theo ngày |
| 17 | jdcTo | JDateChooser |  |
| 21 | jdcFrom | JDateChooser |  |
| 22 | lblFrom | JLabel | Text: From |
| 23 | lblTo | JButton | Text: To |

# CHƯƠNG IV THỰC HIỆN VIẾT MÃ

## Viết mã tạo CSDL

### Tạo CSDL

CREATE DATABASE Project\_1\_test1

GO

USE Project\_1\_test1

GO

CREATE TABLE NHANVIEN(

MANV VARCHAR(10) NOT NULL PRIMARY KEY, -- MÃ NHÂN VIÊN

MATKHAU VARCHAR(50) NOT NULL, -- MẬT KHẨU

TENNV NVARCHAR(50) NOT NULL, -- TÊN NHÂN VIÊN

NGAYSINH DATE NOT NULL, -- NGÀY SINH

GIOITINH BIT NOT NULL, -- GIỚI TÍNH

SDT VARCHAR(20) NOT NULL, -- SỐ ĐIỆN THOẠI

EMAIL VARCHAR(50) NOT NULL, -- EMAIL

LOAI BIT NOT NULL DEFAULT 0, -- LOẠI PARTTIME OR FULLTIME

HINH VARCHAR(200), -- HÌNH

QRCODE VARCHAR(200) NOT NULL, -- QUICK RESPONSE CODE

CHUCVU BIT NOT NULL DEFAULT 0, -- ADMIN

DIACHI NVARCHAR(255) NOT NULL, -- ĐỊA CHỈ

);

CREATE TABLE LOAITHEKHACHHANG(

MALOAIKHACHHANG INT identity(1,1) NOT NULL PRIMARY KEY, -- MÃ LOẠI

TENLOAI NVARCHAR(50) NOT NULL -- TÊN LOẠI

)

CREATE TABLE THEKHACHHANG(

MAKH INT IDENTITY(1,1) NOT NULL PRIMARY KEY, -- MÃ KHÁCH HÀNG

TENKH NVARCHAR(50), -- TÊN KHÁCH HÀNG

SDT VARCHAR(20), -- SỐ ĐIỆN THOẠI

EMAIL VARCHAR(255), -- EMAIL

TICHDIEM INT DEFAULT 0, -- TÍCH ĐIỂM

MALOAIKHACHHANG INT NOT NULL, -- MÃ LOẠI KHÁCH HÀNG

CONSTRAINT FK\_LOAITHEKHACHHANG FOREIGN KEY (MALOAIKHACHHANG) REFERENCES LOAITHEKHACHHANG(MALOAIKHACHHANG)

);

CREATE TABLE LOAISANPHAM(

MALOAISANPHAM nchar(5) NOT NULL PRIMARY KEY, --MÃ LOẠI SẢN PHẨM

TENLOAI NVARCHAR(50) NOT NULL, -- TÊN LOẠI

UNIQUE(TENLOAI),

)

CREATE TABLE SANPHAM(

MASP int IDENTITY(1,1) NOT NULL, -- MÃ SẢN PHẨM

MALOAISANPHAM nchar(5) NOT NULL, --MÃ LOẠI SẢN PHẨM

TENSP NVARCHAR(50) NOT NULL, -- TÊN SẢN PHẨM

SOLUONG INT NOT NULL DEFAULT 0, -- SỐ LƯỢNG

DONGIA FLOAT NOT NULL DEFAULT 0, -- ĐƠN GIÁ

HINH VARCHAR(20) NOT NULL, -- HÌNH SẢN PHẨM

GHICHU NVARCHAR(255), -- GHI CHÚ

PRIMARY KEY (MASP),

CHECK (SOLUONG > 0 AND DONGIA >= 0), -- KIỂM TRA SỐ LƯỢNG ĐƠN GIÁ > 0

CONSTRAINT FK\_LOAISANPHAM FOREIGN KEY (MALOAISANPHAM) REFERENCES LOAISANPHAM(MALOAISANPHAM) ON UPDATE CASCADE

);

CREATE TABLE DONHANG(

MADH VARCHAR(10) NOT NULL PRIMARY KEY, -- MÃ ĐƠN HÀNG

MAKH INT,-- MÃ KHÁCH HÀNG

TENKH NVARCHAR(50),-- TÊN KHÁCH HÀNG

NGAYBAN DATE NOT NULL, -- NGÀY GIAO DỊCH ĐƠN HÀNG

TONGTIEN FLOAT NOT NULL, -- TỔNG TIỀN KHÁCH PHẢI TRẢ

MANV VARCHAR(10) NOT NULL, -- NHÂN VIÊN THANH TOÁN HÓA ĐƠN

CONSTRAINT FK\_NHANVIEN\_BANHANG FOREIGN KEY (MANV) REFERENCES NHANVIEN(MANV),

);

CREATE TABLE CHITIETDONHANG(

MADH VARCHAR(10) NOT NULL,--MÃ ĐƠN HÀNG

MASP int NOT NULL,--MÃ SẢN PHẨM

TENSP NVARCHAR(50) NOT NULL, -- TÊN SẢN PHẨM

SOLUONG INT NOT NULL, -- SỐ LƯỢNG

DONGIA FLOAT NOT NULL, -- ĐƠN GIÁ

GIAMGIA FLOAT NOT NULL, -- GIẢM GIÁ

THANHTIEN FLOAT NOT NULL, -- THÀNH TIỀN (VD 1 MÓN ORDER 2L)

MOTA NVARCHAR(255), -- MÔ TẢ

CONSTRAINT PK\_CHITIETDONHANG PRIMARY KEY (MADH, MASP),

CONSTRAINT FK\_DONHANG\_CTDH FOREIGN KEY (MADH) REFERENCES DONHANG(MADH),

CONSTRAINT FK\_SANPHAM\_CTDH FOREIGN KEY (MASP) REFERENCES SANPHAM(MASP)

);

### SQL truy vấn và thao tác bảng

#### LOAITHEKHACHHANG

|  |  |
| --- | --- |
| **Hành động** | **Câu lệnh SQL** |
| Thêm mới | INSERT INTO NHANVIEN (MANV, MATKHAU, TENNV, NGAYSINH, GIOITINH, SDT, EMAIL, LOAI, HINH, QRCODE, CHUCVU, DIACHI)  VALUES (?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?) |
| Cập nhật theo mã | UPDATE NHANVIEN SET MATKHAU =?, TENNV=?, NGAYSINH=?, GIOITINH=?, SDT=?, EMAIL=?, LOAI=?, HINH=?, QRCODE=?, CHUCVU=?, DIACHI=? WHERE MANV=? |
| Xóa theo mã | DELETE FROM NHANVIEN WHERE MANV=? |
| Truy vấn tất cả | SELECT \* FROM NHANVIEN |
| Truy vấn theo mã | SELECT \* FROM NHANVIEN WHERE MANV=? |

#### LOAITHEKHACHHANG

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

#### THEKHACHHANG

|  |  |
| --- | --- |
| **Hành động** | **Câu lệnh SQL** |
| Thêm mới | INSERT INTO THEKHACHHANG (TENKH, SDT, EMAIL, TICHDIEM, MALOAIKHACHANG) VALUES (?, ?, ?, ?, ?) |
| Cập nhật theo mã | UPDATE THEKHACHHANG SET TENKH=?, SDT=?, EMAIL=?, TICHDIEM=?, MALOAIKHACHANG=? WHERE MAKH=? |
| Xóa theo mã | DELETE FROM THEKHACHHANG WHERE MAKH=? |
| Truy vấn tất cả | SELECT \* FROM THEKHACHHANG |
| Truy vấn theo mã | SELECT \* FROM THEKHACHHANG WHERE MAKH=? |

#### LOAISANPHAM

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

#### SANPHAM

|  |  |
| --- | --- |
| **Hành động** | **Câu lệnh SQL** |
| Thêm mới | INSERT INTO SANPHAM (MALOAISANPHAM, TENSP, SOLUONG, DONGIA, HINH, GHICHU) VALUES (?, ?, ?, ?, ?, ?, ?) |
| Cập nhật theo mã | UPDATE SANPHAM SET MALOAISANPHAM=?, TENSP=?, SOLUONG=?, DONGIA=?, HINH=?, GHICHU=? WHERE MASP=? |
| Xóa theo mã | DELETE FROM SANPHAM WHERE MASP=? |
| Truy vấn tất cả | SELECT \* FROM SANPHAM |
| Truy vấn theo mã | SELECT \* FROM SANPHAM WHERE MASP=? |

#### DONHANG

|  |  |
| --- | --- |
| **Hành động** | **Câu lệnh SQL** |
| Thêm mới | INSERT INTO DONHANG (MADH, MAKH, TENKH, NGAYBAN, TONGTIEN, MANV) VALUES (?, ?, ?, ?, ?, ?) |
| Cập nhật theo mã | UPDATE DONHANG SET MAKH=?, TENKH=?, NGAYBAN=?, TONGTIEN=?, MANV=? WHERE MADH=? |
| Xóa theo mã | DELETE FROM DONHANG WHERE MADH=? |
| Truy vấn tất cả | SELECT \* FROM DONHANG |
| Truy vấn theo mã | SELECT \* FROM DONHANG WHERE MADH=? |

#### CHITIETDONHANG

|  |  |
| --- | --- |
| **Hành động** | **Câu lệnh SQL** |
| Thêm mới | INSERT INTO CHITIETDONHANG (MADH, MASP, TENSP, SOLUONG, DONGIA, GIAMGIA, THANHTIEN) VALUES (?, ?, ?, ?, ?, ?, ?) |
| Cập nhật theo mã | UPDATE CHITIETDONHANG SET TENSP=?, SOLUONG=?, DONGIA=?, GIAMGIA=?, THANHTIEN=? WHERE MADH=? AND MASP=? |
| Xóa theo mã | DELETE FROM CHITIETDONHANG WHERE MADH=? AND MASP=? |
| Truy vấn tất cả | SELECT \* FROM CHITIETDONHANG |
| Truy vấn theo mã | SELECT \* FROM CHITIETDONHANG WHERE MADH=? AND MASP=? |

### Các thủ tục lưu tổng hợp thống kê

#### Thống kê doanh thu theo ngày

CREATE PROC SP\_THONGKEDOANHTHUNGAY

AS BEGIN

SELECT NGAYBAN, SUM(TONGTIEN) TONGTIEN

FROM DONHANG

GROUP BY NGAYBAN

END

#### Thống kê doanh thu theo tháng

CREATE PROC SP\_THONGKEDOANHTHUTHANG(@YEAR INT)

AS BEGIN

SELECT MONTH(NGAYBAN) THANG, SUM(TONGTIEN) TONGTIEN

FROM DONHANG

WHERE YEAR(NGAYBAN) = @YEAR

GROUP BY MONTH(NGAYBAN)

END

#### Thống kê doanh thu theo năm

CREATE PROC SP\_THONGKEDOANHTHUNAM

AS BEGIN

SELECT YEAR(NGAYBAN) THANG, SUM(TONGTIEN) TONGTIEN

FROM DONHANG

GROUP BY YEAR(NGAYBAN)

END

## Lập trình JDBC

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

A close up of a sign

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 từ CSDL, chuyển đổi thành model để hiển thị lên các thành phần giao diện.

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

### Lớp hỗ trợ

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

#### Lớp tiện ích DateHelper

Chứa các phương thức chuyển đổi thời gian và chuỗi giúp lập trình sau này khi xử lý thời gian dễ dàng hơn.

|  |
| --- |
| public class DateHelper {  static final SimpleDateFormat DATE\_FORMATER = new SimpleDateFormat("yyyy/MM/dd");  public static Date toDate(String date, String...pattern){  try {  if(pattern.length > 0){  DATE\_FORMATER.applyPattern(pattern[0]);  }  if(date == null){  return DateHelper.now();  }  return DATE\_FORMATER.parse(date);  } catch (ParseException e) {  throw new RuntimeException(e);  }  }    public static String toString(Date date, String...pattern){  if(pattern.length > 0){  DATE\_FORMATER.applyPattern(pattern[0]);  }  if(date == null){  date = DateHelper.now();  }  return DATE\_FORMATER.format(date);  }  public static Date now(){  return new Date();  }  public static Date addDays(Date date,int days){  date.setTime(date.getTime() + days\*24\*60\*60\*1000);  return date;  }  public static Date add(int days){  Date now = DateHelper.now();  now.setTime(now.getTime() + days\*24\*60\*60\*1000);  return now;  }  } |

#### Lớp tiện ích JdbcHelper

|  |
| --- |
| public class JdbcHelper {  private static String driver = "com.microsoft.sqlserver.jdbc.SQLServerDriver";  private static String db\_url = "jdbc:sqlserver://localhost:1433;databaseName=Project\_1\_test1;";  private static String username = "sa";  private static String password = "123456";  static {  try {  Class.forName(driver);  } catch (Exception e) {  throw new RuntimeException(e);  }  }  public static PreparedStatement preparedStatement(String sql, Object... args) throws SQLException {  Connection conn = DriverManager.getConnection(db\_url, username, password);  PreparedStatement ps = null;  if (sql.trim().startsWith("{")) {  ps = conn.prepareCall(sql);  } else {  ps = conn.prepareStatement(sql);  }  for (int i = 0; i < args.length; i++) {  ps.setObject(i + 1, args[i]);  }  return ps;  }  public static void executeUpdate(String sql, Object... args) {  try {  PreparedStatement st = preparedStatement(sql, args);  try {  st.executeUpdate();  } finally {  st.getConnection().close();  }  } catch (SQLException e) {  throw new RuntimeException(e);  }  }  public static ResultSet executeQuery(String sql, Object... args) {  try {  PreparedStatement ps = preparedStatement(sql, args);  return ps.executeQuery();  } catch (SQLException e) {  throw new RuntimeException(e);  }  }  } |

#### Lớp tiện ích DateConvert

|  |
| --- |
| public class DateConvert {    // Sử dụng LocalDate để convert ra String, Date... và ngược lại.    /\* Chuyển từ java.util.Date sang kiểu LocalDate(không giờ, không vùng)  @Truyền vào tham số là lớp util.Date (lấy từ JDateChooser  \*/  public static LocalDate DatetoLocal(java.util.Date date){  return date.toInstant().atZone(ZoneId.systemDefault()).toLocalDate();  }    /\*  Chuyển từ java.sql.đate sang kiểu LocalDate  @Truyền vào tham số sql.Date  \*/  public static LocalDate sqlDatetoLocal(java.sql.Date date){  return date.toInstant().atZone(ZoneId.systemDefault()).toLocalDate();  }  /\*  Chuyển từ LocalDate sang sql.Date  @Truyền vào tham số là LocalDate  \*/  public static java.sql.Date LocaltoSqlDate(LocalDate date){  java.sql.Date sqlDate = Date.valueOf(date);  return sqlDate;  }  /\*  Chuyển từ LocalDate sang util.Date  @Truyền vào tham số là LocalDate  \*/  public static java.util.Date LocaltoDate(LocalDate date){  java.util.Date utilDate= Date.from(date.atStartOfDay(ZoneId.systemDefault()).toInstant());  return utilDate;  }  /\*  Chuyển từ LocalDate sang String  @Truyền vào tham số là LocalDate  Fill lên các textfield  \*/  public static String toString(LocalDate date,String...pattern){  SimpleDateFormat formatDate = new SimpleDateFormat("dd/MM/yyyy");  if(pattern.length>0){  formatDate.applyPattern(pattern[0]);  return formatDate.format(date);  }  if(date == null){  date = LocalDate.now();  }  return formatDate.format(date);    }  /\*  Chuyển từ String sang LocalDate  @Truyền vào tham số là chuỗi  \*/  public static LocalDate toLocalDate(String date,String...pattern){  DateTimeFormatter format = DateTimeFormatter.ofPattern("dd/MM/yyyy");  if(pattern != null){  format = DateTimeFormatter.ofPattern(pattern[0]);  }  return (LocalDate) LocalDate.parse(date,format);  }  } |

#### Lớp tiện ích QRHelper

|  |
| --- |
| public class QRHelper {  public static byte[] getQRCodeImage(String text, int width, int height) {  try {  QRCodeWriter qrCodeWriter = new QRCodeWriter();    BitMatrix bitMatrix = qrCodeWriter.encode(text, BarcodeFormat.QR\_CODE, width, height);  ByteArrayOutputStream byteArrayOutputStream = new ByteArrayOutputStream();  MatrixToImageWriter.writeToStream(bitMatrix, "png", byteArrayOutputStream);      return byteArrayOutputStream.toByteArray();  } catch (Exception e) {  return null;  }  }    public static File getQRImage(String text, int width, int height) {  try {  QRCodeWriter qrCodeWriter = new QRCodeWriter();  File file = new File("image.png");  BitMatrix bitMatrix = qrCodeWriter.encode(text, BarcodeFormat.QR\_CODE, width, height);    MatrixToImageWriter.writeToFile(bitMatrix, "png", file);  System.out.println(file.getName());  return file.getAbsoluteFile();  } catch (Exception e) {  return null;  }  }    public static byte[] getBarCodeImage(String text, int width, int height) {  try {  Hashtable<EncodeHintType, ErrorCorrectionLevel> hintMap = new Hashtable<>();  hintMap.put(EncodeHintType.ERROR\_CORRECTION, ErrorCorrectionLevel.L);  Writer writer = new Code128Writer();  BitMatrix bitMatrix = writer.encode(text, BarcodeFormat.CODE\_128, width, height);  ByteArrayOutputStream byteArrayOutputStream = new ByteArrayOutputStream();  MatrixToImageWriter.writeToStream(bitMatrix, "png", byteArrayOutputStream);  return byteArrayOutputStream.toByteArray();  } catch (Exception e) {  return null;  }  }    public void readQRcode(){    }  } |

#### Lớp tiện ích TichDiem

|  |
| --- |
| public class TichDiem {    public static void main(String[] args) throws SQLException{  DonHang dh = new DonHang();  HoaDonDAO dao = new HoaDonDAO();    dh = dao.FindbyID("ABCD");  System.out.println(dh.getTenKH());  }  } |

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

Để tránh các sai sót về kiểu dữ liệu khi làm việc với CSDL, chúng ta cần phải xây dựng các lớp mô tả dữ liệu với kiểu phù hợp. Điều này giúp làm việc với CSDL chặt chẽ hơn, dễ hơn.

#### NHANVIEN

|  |
| --- |
| public String getDiaChi() {  return DiaChi;  }  public void setDiaChi(String DiaChi) {  this.DiaChi = DiaChi;  }  public boolean getLoai() {  return Loai;  }  public void setLoai(boolean Loai) {  this.Loai = Loai;  }    public boolean getController() {  return Controller;  }  public void setController(boolean Controller) {  this.Controller = Controller;  }    @Override  public String toString(){  return this.TenNV;  }    public NhanVien(){    }  public String getMaNV() {  return MaNV;  }  public void setMaNV(String MaNV) {  this.MaNV = MaNV;  }  public String getPass() {  return Pass;  }  public void setPass(String Pass) {  this.Pass = Pass;  }  public String getTenNV() {  return TenNV;  }  public void setTenNV(String TenNV) {  this.TenNV = TenNV;  }  public LocalDate getNgaySinh() {  return NgaySinh;  }  public void setNgaySinh(LocalDate NgaySinh) {  this.NgaySinh = NgaySinh;  }  public boolean getGioitinh() {  return Gioitinh;  }  public void setGioitinh(boolean Gioitinh) {  this.Gioitinh = Gioitinh;  }  public String getSDT() {  return SDT;  }  public void setSDT(String SDT) {  this.SDT = SDT;  }  public String getEmail() {  return Email;  }  public void setEmail(String Email) {  this.Email = Email;  }  public String getIMG() {  return IMG;  }  public void setIMG(String IMG) {  this.IMG = IMG;  }  public String getQRCode() {  return QRCode;  }  public void setQRCode(String QRCode) {  this.QRCode = QRCode;  }  } |

#### LOAITHEKH

|  |
| --- |
| public class LoaiTheKH {  public int MaLoaiKhachHang;  public String TenLoai;  public LoaiTheKH(int MaLoaiKhachHang, String TenLoai) {  this.MaLoaiKhachHang = MaLoaiKhachHang;  this.TenLoai = TenLoai;  }  public LoaiTheKH() {  }  @Override  public String toString() {  return this.TenLoai;  }  public int getMaLoaiKhachHang() {  return MaLoaiKhachHang;  }  public void setMaLoaiKhachHang(int MaLoaiKhachHang) {  this.MaLoaiKhachHang = MaLoaiKhachHang;  }  public String getTenLoai() {  return TenLoai;  }  public void setTenLoai(String TenLoai) {  this.TenLoai = TenLoai;  }    } |

#### THEKHACHHANG

|  |
| --- |
| public class TheKhachHang {  public int MaKH;  public String TenKH;  public String SDT;  public String Email;  public int DiemTich;  public int MaLoaiThe;  public String TenLoai;  @Override  public String toString() {  return this.TenKH;  }  public int getMaKH() {  return MaKH;  }  public void setMaKH(int MaKH) {  this.MaKH = MaKH;  }  public String getTenKH() {  return TenKH;  }  public void setTenKH(String TenKH) {  this.TenKH = TenKH;  }  public String getSDT() {  return SDT;  }  public void setSDT(String SDT) {  this.SDT = SDT;  }  public String getEmail() {  return Email;  }  public void setEmail(String Email) {  this.Email = Email;  }  public int getDiemTich() {  return DiemTich;  }  public void setDiemTich(int DiemTich) {  this.DiemTich = DiemTich;  }  public int getMaLoaiThe() {  return MaLoaiThe;  }  public void setMaLoaiThe(int MaLoaiThe) {  this.MaLoaiThe = MaLoaiThe;  }  public String getTenLoai() {  return TenLoai;  }  public void setTenLoai(String TenLoai) {  this.TenLoai = TenLoai;  }  } |

#### SANPHAM

|  |
| --- |
| public class SanPham {  private int MASP;  private String MALOAISANPHAM;  private String TENSP;  private int SOLUONG;  private double DONGIA;  private String HINH;  private String GHICHU;  public SanPham() {  }  public SanPham(int MASP, String TENSP, String MALOAISANPHAM, int SOLUONG, double DONGIA, String HINH, String GHICHU) {  this.MASP = MASP;  this.TENSP = TENSP;  this.MALOAISANPHAM = MALOAISANPHAM;  this.SOLUONG = SOLUONG;  this.DONGIA = DONGIA;  this.HINH = HINH;  this.GHICHU = GHICHU;  }  public int getMASP() {  return MASP;  }  public void setMASP(int MASP) {  this.MASP = MASP;  }  public String getMALOAISANPHAM() {  return MALOAISANPHAM;  }  public void setMALOAISANPHAM(String MALOAISANPHAM) {  this.MALOAISANPHAM = MALOAISANPHAM;  }  public String getTENSP() {  return TENSP;  }  public void setTENSP(String TENSP) {  this.TENSP = TENSP;  }  public int getSOLUONG() {  return SOLUONG;  }  public void setSOLUONG(int SOLUONG) {  this.SOLUONG = SOLUONG;  }  public double getDONGIA() {  return DONGIA;  }  public void setDONGIA(double DONGIA) {  this.DONGIA = DONGIA;  }  public String getHINH() {  return HINH;  }  public void setHINH(String HINH) {  this.HINH = HINH;  }  public String getGHICHU() {  return GHICHU;  }  public void setGHICHU(String GHICHU) {  this.GHICHU = GHICHU;  }  } |

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

DAO (Data Access Object) là các lớp 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:

|  |
| --- |
| /\*\* \* Thêm mới thực thể vào CSDL \* @param entity là thực thể chứa thông tin bản ghi mới \*/ public void insert(Entity entity){}  /\*\* \* Cập nhật thực thể vào CSDL \* @param entity là thực thể chứa thông tin bản ghi cần cập nhật \*/ public void update(Entity entity){} /\*\* \* Xóa bản ghi khỏi CSDL \* @param id là mã của bản ghi cần xóa \*/ public void delete(String id){} /\*\* \* Truy vấn tất cả các các thực thể \* @return danh sách các thực thể \*/ public List<Entity> select(){} /\*\* \* Truy vấn thực thể theo mã \* @param id là mã của bản ghi được truy vấn \* @return thực thể chứa thông tin của bản ghi \*/ public Entity findById(String id){} |

Sau đây là các lớp DAO cụ thể

#### NhanVienDAO

|  |
| --- |
| public class NhanVienDAO {  private ResultSet rs;    // Thêm Nhân Viên  public void insertNV(NhanVien model){  String sql = "insert into NHANVIEN(MANV,TENNV,MATKHAU,GIOITINH,NGAYSINH,SDT,EMAIL, DIACHI,LOAI,CHUCVU,HINH,QRCODE) "  + "values(?,?,?,?,?,?,?,?,?,?,?,?)";  JdbcHelper.executeUpdate(sql,model.getMaNV(),model.getTenNV(),model.getPass(),model.getGioitinh(),DateConvert.LocaltoSqlDate(model.getNgaySinh()),  model.getSDT(),model.getEmail(),model.getDiaChi(),model.getLoai(),model.getController(),model.getIMG(),model.getQRCode());  }    // Update Nhân Viên  public void updateNV(NhanVien model){  String sql = "update NHANVIEN set TENNV=?, MATKHAU=?, NGAYSINH=?, GIOITINH=?, SDT=?, EMAIL=?, LOAI=?, HINH=?, QRCODE=?, CHUCVU=? where MANV=?";  JdbcHelper.executeUpdate(sql, model.getTenNV(), model.getPass(),DateConvert.LocaltoSqlDate( model.getNgaySinh()), model.getGioitinh(), model.getSDT(), model.getEmail(), model.getLoai(), model.getIMG(), model.getQRCode(), model.getController(), model.getMaNV());  }    // Update Mã nhân viên  public void updateMaNV(NhanVien model){  String sql = " update NHANVIEN set MANV=? where TENNV like %N'?'";  JdbcHelper.executeUpdate(sql, model.getMaNV(),model.getTenNV());  }    // Xóa nhân viên  public void DelNV(String MaNV){  String sql = "delete from NHANVIEN where MANV =?";  JdbcHelper.executeUpdate(sql, MaNV);    }  // Lấy tất cả Nhân viên có trong DB  public List<NhanVien> select(){  String sql = "select \* from NHANVIEN";  return select(sql);  }    // Tìm nhân viên qua MaNV  public NhanVien FindbyID(String manv){  String sql = "select \* from NHANVIEN where MANV=?";  List<NhanVien> list = select(sql,manv);  return list.size()>0 ?list.get(0):null;  }    //  private List<NhanVien> select(String sql, Object...args){  List<NhanVien> list = new ArrayList<>();  try{  rs=null;  try{  rs = JdbcHelper.executeQuery(sql, args);  while(rs.next()){  NhanVien model = readFromResult(rs);  list.add(model);  }  }finally{  rs.getStatement().getConnection().close();  }  }catch(Exception e){  System.out.println("Lỗi phần Đọc nhân viên "+e);  }  return list;  }    // Đọc nhân viên  private NhanVien readFromResult(ResultSet result) throws SQLException{  this.rs = result;  NhanVien model = new NhanVien();  model.setMaNV(rs.getString("MANV"));  model.setTenNV(rs.getString("TENNV"));  model.setPass(rs.getString("MATKHAU"));  model.setGioitinh(rs.getBoolean("GIOITINH"));  model.setNgaySinh(rs.getDate("NGAYSINH").toLocalDate());  model.setSDT(rs.getString("SDT"));  model.setEmail(rs.getString("EMAIL"));  model.setDiaChi(rs.getString("DIACHI"));  model.setLoai(rs.getBoolean("LOAI"));  model.setController(rs.getBoolean("CHUCVU"));  model.setIMG(rs.getString("HINH"));  model.setQRCode(rs.getString("QRCODE"));    return model;  }  } |

#### KhachHangDAO

|  |
| --- |
| public class KhachHangDAO {  public void insert(KhachHang model) {  String sql = "INSERT INTO KHACHHANG (MAKH, TENKH, SDT, EMAIL, MOTA) VALUES (?, ?, ?, ?, ?,)";  JdbcHelper.executeUpdate(sql,  model.getMaKH(),  model.getTenKH(),  model.getSDT(),  model.getEmail(),  model.getMoTa());  }    public void update(KhachHang model) {  String sql = "UPDATE KHACHHANG SET TENKH = ?, SDT = ?, EMAIL = ?, MOTA = ? WHERE MAKH = ?";  JdbcHelper.executeUpdate(sql,  model.getTenKH(),  model.getSDT(),  model.getEmail(),  model.getMoTa(),  model.getMaKH());  }    public void delete(String MaKH) {  String sql = "DELETE FORM KHACHHANG WHERE MAKH";  JdbcHelper.executeUpdate(sql, MaKH);  }    public List<KhachHang>select() {  String sql = "SELECT \* FROM KHACHHANG";  return select(sql);  }    public KhachHang findByID(String makh) {  String sql = "SELECT \* FROM KHACHHANG WHERE MAKH = ?";  java.util.List<KhachHang> list = select(sql, makh);  return list.size() > 0 ? list.get(0) : null;  }    private List<KhachHang> select(String sql, Object...args){  List<KhachHang> list = new ArrayList<>();  try {  ResultSet rs = null;  try {  rs = JdbcHelper.executeQuery(sql, args);  while(rs.next()){  KhachHang model = readFromResultSet(rs);  list.add(model);  }  } finally {  rs.getStatement().getConnection().close();  }  } catch (Exception e) {  throw new RuntimeException(e);  }  return list;  }    private KhachHang readFromResultSet (ResultSet rs) throws SQLException {  KhachHang model = new KhachHang();  model.setMaKH(rs.getString("MAKH"));  model.setTenKH(rs.getString("TENKH"));  model.setSDT(rs.getString("SDT"));  model.setEmail(rs.getString("EMAIL"));  model.setMoTa(rs.getString("MOTA"));  return model;  }  } |

#### LoaiSanPhamDAO

|  |
| --- |
| public class LoaiSPDAO {  public void insert (LoaiSP model)  {  String sql = "INSERT INTO LOAISANPHAM(MALOAISANPHAM, TENLOAI) VALUES (?, ?)";  JdbcHelper.executeUpdate(sql,  model.getMALOAISANPHAM(),  model.getTENLOAI());  }    public void update(LoaiSP model)  {  String sql="UPDATE LOAISANPHAM SET TENLOAI=? WHERE MALOAISANPHAM=?";  JdbcHelper.executeUpdate(sql,  model.getTENLOAI(),  model.getMALOAISANPHAM());  }    public void delete(String MALOAISANPHAM)  {  String sql="DELETE FROM LOAISANPHAM WHERE MALOAISANPHAM=?";  JdbcHelper.executeUpdate(sql, MALOAISANPHAM);  }    public List<LoaiSP> select()  {  String sql="SELECT \* FROM LOAISANPHAM";  return select(sql);  }    public LoaiSP findById(String MALOAISANPHAM)  {  String sql="SELECT \* FROM LOAISANPHAM WHERE MALOAISANPHAM=?";  List<LoaiSP> list = select(sql, MALOAISANPHAM);  return list.size() > 0 ? list.get(0) : null;  }    private List<LoaiSP> select(String sql, Object...args)  {  List<LoaiSP> list=new ArrayList<>();  try {  ResultSet rs = null;  try  {  rs = JdbcHelper.executeQuery(sql, args);  while(rs.next())  {  LoaiSP model=readFromResultSet(rs);  list.add(model);  }  }  finally  {  rs.getStatement().getConnection().close();  }  }  catch (SQLException ex)  {  throw new RuntimeException(ex);  }  return list;  }  private LoaiSP readFromResultSet(ResultSet rs) throws SQLException  {  LoaiSP model=new LoaiSP();  model.setMALOAISANPHAM(rs.getString("MAlOAISANPHAM"));  model.setTENLOAI(rs.getString("TENLOAI"));  return model;  }  } |

#### SanPhamDAO

|  |
| --- |
| public class SanPhamDAO {  public void insert (SanPham model)  {  String sql = "INSERT INTO SANPHAM(MALOAISANPHAM, TENSP, SOLUONG, DONGIA, HINH, GHICHU) VALUES (?,?,?,?,?,?)";  JdbcHelper.executeUpdate(sql,  model.getMALOAISANPHAM(),  model.getTENSP(),  model.getSOLUONG(),  model.getDONGIA(),  model.getHINH(),  model.getGHICHU());  }    public void update(SanPham model)  {  String sql="UPDATE SANPHAM SET TENSP=?, MALOAISANPHAM=?, SOLUONG=?, DONGIA=?, HINH=?, GHICHU=? WHERE MASP=?";  JdbcHelper.executeUpdate(sql,  model.getTENSP(),  model.getMALOAISANPHAM(),  model.getSOLUONG(),  model.getDONGIA(),  model.getHINH(),  model.getGHICHU(),  model.getMASP());  }    public void delete(Integer MASP)  {  String sql="DELETE FROM SANPHAM WHERE MASP=?";  JdbcHelper.executeUpdate(sql, MASP);  }  public List<SanPham> select()  {  String sql="SELECT \* FROM SANPHAM";  return select(sql);  }    public SanPham findById(Integer masp)  {  String sql="SELECT \* FROM SANPHAM WHERE MASP=?";  List<SanPham> list = select(sql, masp);  return list.size() > 0 ? list.get(0) : null;  }    private List<SanPham> select(String sql, Object...args)  {  List<SanPham> list=new ArrayList<>();  try {  ResultSet rs = null;  try  {  rs = JdbcHelper.executeQuery(sql, args);  while(rs.next())  {  SanPham model=readFromResultSet(rs);  list.add(model);  }  }  finally  {  rs.getStatement().getConnection().close();  }  }  catch (SQLException ex)  {  throw new RuntimeException(ex);  }  return list;  }  private SanPham readFromResultSet(ResultSet rs) throws SQLException  {  SanPham model=new SanPham();  model.setMASP(rs.getInt("MASP"));  model.setMALOAISANPHAM(rs.getString("MALOAISANPHAM"));  model.setTENSP(rs.getString("TENSP"));  model.setSOLUONG(rs.getInt("SOLUONG"));  model.setDONGIA(rs.getDouble("DONGIA"));  model.setHINH(rs.getString("HINH"));  model.setGHICHU(rs.getString("GHICHU"));    return model;  }  } |

#### DonHangDAO

|  |
| --- |
| public class DonHangDAO {  public void insert(DonHang HD){  String sql = "insert into DONHANG(MADH,MAKH,TENKH,NGAYBAN,TONGTIEN,MANV) values (?,?,?,?,?,?)";  JdbcHelper.executeUpdate(sql,HD.getMaDH(),  HD.getMaKH(),  HD.getTenKH(),  java.sql.Date.valueOf(HD.getNgayBan()),  HD.getTongTien(),  HD.getMaNV());  }    public List<DonHang> select() {  String sql = "SELECT \* FROM DONHANG";  return select(sql);  }    public List<DonHang> selectFollowByMonthAndYear(int thang, int nam){  String sql = "select \* from DONHANG where MONTH(NGAYBAN) = ? and YEAR(NGAYBAN) = ?";  return select(sql);  }    public DonHang FindbyID(int MaDH){  String sql = " select \* from DONHANG where MADH = ?" ;  List<DonHang> list = select(sql,MaDH);  return list.size() > 0 ? list.get(0) : null;    }  private List<DonHang> select(String sql, Object...args){  List<DonHang> list = new ArrayList<>();  try {  ResultSet rs = null;  try {  rs = JdbcHelper.executeQuery(sql, args);  while(rs.next()){  DonHang model = readFromResultSet(rs);  list.add(model);  }  } finally {  rs.getStatement().getConnection().close();  }  } catch (Exception e) {  throw new RuntimeException(e);  }  return list;  }    private DonHang readFromResultSet (ResultSet rs) throws SQLException {  DonHang model = new DonHang();  model.setMaDH(rs.getString("MADH"));  model.setMaKH(rs.getInt("MAKH"));  model.setTenKH(rs.getString("TENKH"));  model.setNgayBan(rs.getDate("NGAYBAN").toLocalDate());  model.setTongTien(rs.getFloat("TONGTIEN"));  model.setMaNV(rs.getString("MANV"));  return model;  }  } |

#### ChiTietDonHangDAO

|  |
| --- |
| public class ChiTietDonHangDAO {    public void insert(ChiTietDonHang DHCT){  String sql = "insert into CHITIETDONHANG(MADH, MASP, TENSP, SOLUONG, DONGIA, GIAMGIA, THANHTIEN, MOTA) values (?,?,?,?,?,?,?,?)";  JdbcHelper.executeUpdate(sql, DHCT.getMaDH(),  DHCT.getMaSP(),  DHCT.getTenSP(),  DHCT.getSoLuong(),  DHCT.getDonGia(),  DHCT.getGiamGia(),  DHCT.getThanhTien(),  DHCT.getMoTa());  }    public List<ChiTietDonHang> select() {  String sql = "SELECT \* FROM CHITIETDONHANG";  return select(sql);  }  public ChiTietDonHang FindbyID(int MaDH){  String sql = " select \* from CHITIETDONHANG where MADH = ?" ;  List<ChiTietDonHang> list = select(sql,MaDH);  return list.size() > 0 ? list.get(0) : null;    }  private List<ChiTietDonHang> select(String sql, Object...args){  List<ChiTietDonHang> list = new ArrayList<>();  try {  ResultSet rs = null;  try {  rs = JdbcHelper.executeQuery(sql, args);  while(rs.next()){  ChiTietDonHang model = readFromResultSet(rs);  list.add(model);  }  } finally {  rs.getStatement().getConnection().close();  }  } catch (Exception e) {  throw new RuntimeException(e);  }  return list;  }    private ChiTietDonHang readFromResultSet (ResultSet rs) throws SQLException {  ChiTietDonHang model = new ChiTietDonHang();  model.setMaDH(rs.getString("MADH"));  model.setMaSP(rs.getInt("MASP"));  model.setTenSP(rs.getString("TENSP"));  model.setSoLuong(rs.getInt("SOLUONG"));  model.setDonGia(rs.getFloat("DONGIA"));  model.setGiamGia(rs.getFloat("GIAMGIA"));  model.setThanhTien(rs.getFloat("THANHTIEN"));  model.setMoTa(rs.getString("MOTA"));    return model;  }  } |

#### ThongKeDAO

Lớp DAO này chứa các phương thức gọi đến các thủ tịc lưu được xây dựng trước đây để thực hiện các chức năng tổng hợp thống kê trong ứng dụng.

|  |
| --- |
| public class ThongKeJInternalFrame extends javax.swing.JInternalFrame {  public ThongKeJInternalFrame() {  this.RemoveHeader();  initComponents();  this.setTitle("Thống kê");  fillComboBoxThang();  fillTableThongKeDoanhThuThang();  fillTableThongKeDoanhThuNam();  }  ThongKeDAO dao = new ThongKeDAO();  DonHangDAO dhdao = new DonHangDAO();  void fillComboBoxThang() {  DefaultComboBoxModel model = (DefaultComboBoxModel) cboThang.getModel();  model.removeAllElements();  List<DonHang> list = dhdao.select();  for (DonHang dh : list) {  int thang = dh.getNgayBan().getYear();  if (model.getIndexOf(thang) < 0) {  model.addElement(thang);  }  }  cboThang.setSelectedIndex(0);  }  void fillTableThongKeDoanhThuThang() {  DefaultTableModel model = (DefaultTableModel) tblTKDTThang.getModel();  model.setRowCount(0);  if (cboThang.getSelectedItem() == null) {  } else {  int thang = Integer.parseInt(cboThang.getSelectedItem().toString());  List<Object[]> list = dao.getThongKeDoanhThang(thang);  for (Object[] row : list) {  model.addRow(row);  }  }  }  void fillTableThongKeDoanhThuNam() {  DefaultTableModel model = (DefaultTableModel) tblTKDTNam.getModel();  model.setRowCount(0);  List<Object[]> list = dao.getThongKeDoanhNam();  for (Object[] row : list) {  model.addRow(row);  }  }  void fillTableThongKeDoanhThuNgay() {  DefaultTableModel model = (DefaultTableModel) tblTKDTNgay.getModel();  model.setRowCount(0);  List<Object[]> list = dao.getThongKeDoanhThuNgay();  for (Object[] row : list) {  model.addRow(row);  }  }  private void RemoveHeader() {  this.setBorder(javax.swing.BorderFactory.createEmptyBorder(0, 0, 0, 0));  BasicInternalFrameUI bi = (BasicInternalFrameUI) this.getUI();  bi.setNorthPane(null);  }  private void btnTKNamActionPerformed(java.awt.event.ActionEvent evt) {  try {  String query = "SELECT YEAR(NGAYBAN), SUM(TONGTIEN) \n"  + "FROM DONHANG\n"  + "GROUP BY YEAR(NGAYBAN)";  JDBCCategoryDataset dataset = new JDBCCategoryDataset(SQLConnect.ConnectDb(), query);  JFreeChart chart = ChartFactory.createBarChart("Thống kê doanh thu từng năm", "NĂM BÁN", "TỔNG TIỀN", dataset, PlotOrientation.VERTICAL, true, true, true);  BarRenderer renderer = null;  CategoryPlot plot = null;  ChartFrame frame = new ChartFrame("Bảng thống kê", chart);  frame.setSize(600, 500);  frame.setLocationRelativeTo(null);  frame.setVisible(true);  } catch (Exception e) {  }  }  private void cboThangActionPerformed(java.awt.event.ActionEvent evt) {  fillTableThongKeDoanhThuThang();  }  private void btnTKThangActionPerformed(java.awt.event.ActionEvent evt) {  try {  int thang = Integer.parseInt(cboThang.getSelectedItem().toString());  String query = "SELECT MONTH(NGAYBAN) THANG, SUM(TONGTIEN) TONGTIEN\n"  + "FROM DONHANG\n"  + "WHERE YEAR(NGAYBAN) = " + thang + "\n"  + "GROUP BY MONTH(NGAYBAN)";  JDBCCategoryDataset dataset = new JDBCCategoryDataset(SQLConnect.ConnectDb(), query);  JFreeChart chart = ChartFactory.createBarChart("Thống kê doanh thu theo tháng", "Tháng bán", "Tổng tiền", dataset, PlotOrientation.VERTICAL, true, true, true);  BarRenderer renderer = null;  CategoryPlot plot = null;  ChartFrame frame = new ChartFrame("Bảng thống kê", chart);  frame.setSize(1000, 800);  frame.setLocationRelativeTo(null);  frame.setVisible(true);  } catch (Exception e) {  }  }  private void btnTKThang1ActionPerformed(java.awt.event.ActionEvent evt) {  try {  SimpleDateFormat dateFrom1 = new SimpleDateFormat("yyyy-MM-dd");  String theDateFrom = dateFrom1.getDateInstance().format(jdcFrom.getDate());  SimpleDateFormat dateTo1 = new SimpleDateFormat("yyyy-MM-dd");  String theDateTo = dateTo1.getDateInstance().format(jdcTo.getDate());  String query = "SELECT NGAYBAN, SUM(TONGTIEN) AS TONGTIEN "  + "FROM DONHANG "  + "WHERE NGAYBAN BETWEEN '" + theDateFrom + "' AND '" + theDateTo + "' "  + "GROUP BY DONHANG.NGAYBAN";  JDBCXYDataset dataset = new JDBCXYDataset(DriverManager.getConnection("jdbc:sqlserver://localhost;DatabaseName=Project\_1\_test1", "sa", "123456"), query);  dataset.executeQuery(query);  SQLConnect.ConnectDb().close();  JFreeChart chart = ChartFactory.createTimeSeriesChart("Thống kê doanh thu theo tháng", "Ngay", "Tổng tiền", (XYDataset) dataset);  BarRenderer renderer = new BarRenderer();  ChartFrame frame = new ChartFrame("Bảng thống kê", chart);  frame.setSize(600, 500);  frame.setLocationRelativeTo(null);  frame.setVisible(true);  } catch (Exception e) {  e.printStackTrace();  }  }  private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {  SimpleDateFormat dateFrom1 = new SimpleDateFormat("yyyy-MM-dd");  String theDateFrom = dateFrom1.format(jdcFrom.getDate());  SimpleDateFormat dateTo1 = new SimpleDateFormat("yyyy-MM-dd");  String theDateTo = dateTo1.format(jdcTo.getDate());  LocalDate dateFrom = jdcFrom.getDate().toInstant().atZone(ZoneId.systemDefault()).toLocalDate();  java.sql.Date date1 = Date.valueOf(dateFrom);  LocalDate dateTo = jdcTo.getDate().toInstant().atZone(ZoneId.systemDefault()).toLocalDate();  java.sql.Date date2 = Date.valueOf(dateTo);  System.out.println(date1 + " " + date2);  }  } |

## Viết mã cho ứng dụng

### Các lớp tiện ích

Lập trình điều khiển hành vi tương tác của người dùng lên các giao diện của ứng dụng. Trước khi lập trình, chúng ta cần thực hiện 3 điều sau đây.

#### ShareHelper

Lớp ShareHelper cung cấp phương thức tiện ích quản lý người đăng nhập và làm việc với hình ảnh chia sẻ trong tất cả các form của ứng dụng

|  |
| --- |
| public class ShareHelper  {  /\*\*  \* Ảnh biểu tượng của ứng dụng, xuất hiện trên mọi cửa sổ  \*/  public static final Image APP\_ICON;    static  {  // Tải biểu tượng ứng dụng  String file = "/images/Window-New-Open-16.png";  APP\_ICON = new ImageIcon(ShareHelper.class.getResource(file)).getImage();  }  /\*\*  \* Sao chép file logo chuyên đề vào thư mục logo  \* @param file là đối tượng file ảnh  \* @return chép được hay không  \*/  public static boolean saveLogo(File file)  {  File dir = new File("logos");  // Tạo thư mục nếu chưa tồn tại  if(!dir.exists())  {  dir.mkdirs();  }  File newFile = new File(dir, file.getName());  try  {  // Copy vào thư mục logos (đè nếu đã tồn tại)  Path source = Paths.get(file.getAbsolutePath());  Path destination = Paths.get(newFile.getAbsolutePath());  Files.copy(source, destination, StandardCopyOption.REPLACE\_EXISTING);  return true;  }  catch (Exception ex)  {  return false;  }  }  /\*\*  \* Đọc hình ảnh logo chuyên đề  \* @param fileName là tên file logo  \* @return ảnh đọc được  \*/  public static ImageIcon readLogo(String fileName)  {  File path = new File("logos", fileName);  return new ImageIcon(path.getAbsolutePath());  }  /\*\*  \* Đối tượng này chứa thông tin người sử dụng sau khi đăng nhập  \*/  public static NhanVien USER = null;  /\*\*  \* Xóa thông tin của người sử dụng khi có yêu cầu đăng xuất  \*/  public static void logoff()  {  ShareHelper.USER = null;  }  /\*\*  \* Kiểm tra xem đăng nhập hay chưa  \* @return đăng nhập hay chưa  \*/  public static boolean authenticated()  {  return ShareHelper.USER != null;  }  } |

#### DialogHelper

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

#### SQLConnect

|  |
| --- |
| public class SQLConnect {  Connection conn=null;  public static Connection ConnectDb(){  try{  Connection conn = DriverManager.getConnection("jdbc:sqlserver://localhost;DatabaseName=Project\_1\_test1", "sa", "123456");  return conn;  }catch(Exception e){  JOptionPane.showMessageDialog(null, e);  return null;  }  }  } |

### Viết mã cho các chức năng

Trong phần này chúng ta tiến hành viết mã cho các giao diện đã thiết kế trước đây. Cụ thể chúng ta sẽ lần lượt viết mã thực hiện các chức năng cho các giao diện sau đây

#### Màn hình chào (ChaoJDialog)

#### Form đăng nhập (SignInFrame)



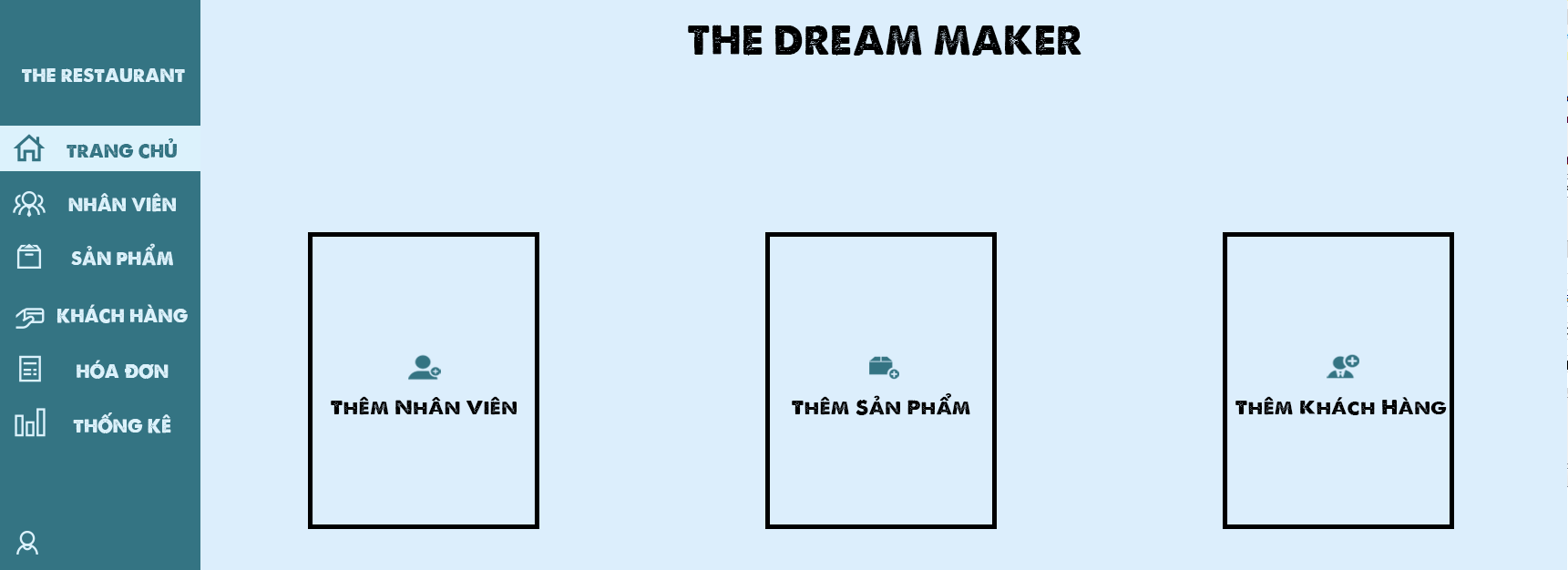
**MÔ TẢ HOẠT ĐỘNG**

|  |  |
| --- | --- |
| **Sự kiện** | **Công việc** |
| lblDangNhap | Login() |
| lblQRCode | openQRCode() |
| pnlDangNhap | setColor()  resetColor() |

**HƯỚNG DẪN CÀI ĐẶT MÃ NGUỒN**

|  |
| --- |
| NhanVienDAO dao = new NhanVienDAO();  int x;  int y;  public SignInFrame() {  initComponents();  setSize(1011, 550);  this.setLocationRelativeTo(null);  }    public void Login(){  String manv = txtNhapTaiKhoan.getText();  String matkhau = new String(txtMatKhau.getPassword());  try {  NhanVien nhanvien = dao.FindbyID(manv);  if(nhanvien!=null){  String matkhau2 = nhanvien.getPass();  if(matkhau2.equalsIgnoreCase(matkhau)){  ShareHelper.USER = nhanvien;    DialogHepler.alert(this, "Đăng nhập thành công!");  System.out.println(ShareHelper.USER);  new MainFrame().setVisible(true);  this.dispose();  }  else{  DialogHepler.alert(this, "Mật khẩu không đúng!");  }  }else{  DialogHepler.alert(this, "Tên tài khoản không đúng!");  }  } catch (Exception e) {  DialogHepler.alert(this, "Lỗi truy vấn dữ liệu");  }  }    public void openQRCode(){  new Menu().setVisible(true);  this.dispose();  }  public void setColor(JPanel p) {  p.setBackground(new Color(68, 154, 207));  }    public void resetColor(JPanel p1) {  p1.setBackground(new Color(52, 116, 131));  } |

#### Giao diện chính



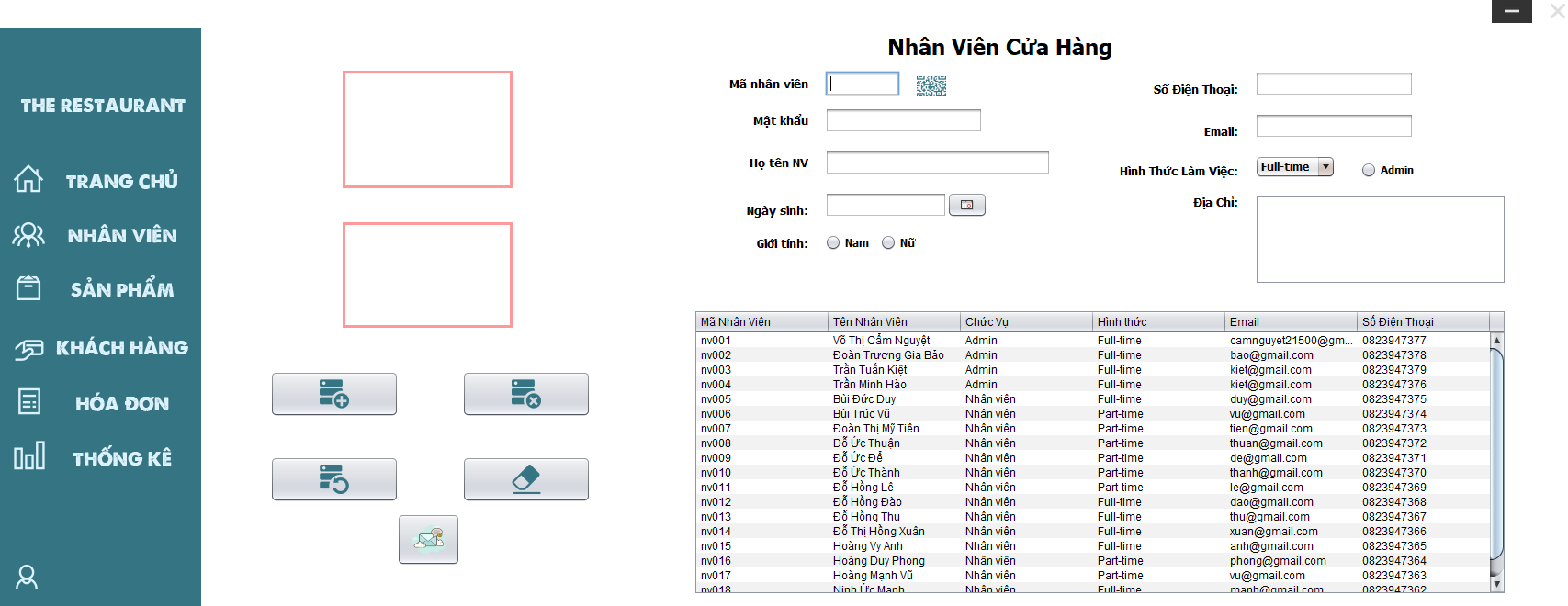
**MÔ TẢ HOẠT ĐỘNG**

|  |  |
| --- | --- |
| **Sự kiện** | **Công việc** |
| Constructor | lblLogout.setVisible(false); |
| DesktopAncestorAdded | DesktopAncestorAdded(); |
| lblTrangChuMouseEntered | TrangChuMouseEntered(); |
| lblTrangChuMouseExited | TrangChuMouseExited(); |
| lblTrangChuMouseClicked | TrangChuMouseClicked(); |
| lblNhanVienMouseEntered | NhanVienMouseEntered(); |
| lblNhanVienMouseExited | NhanVienMouseExited(); |
| lblNhanVienMouseClicked | NhanVienMouseClicked(); |
| lblSanPhamMouseEntered | SanPhamMouseEntered(); |
| lblSanPhamMouseExited | SanPhamMouseExited(); |
| lblSanPhamMouseClicked | SanPhamMouseClicked(); |
| lblKhachHangMouseEntered | KhachHangMouseEntered(); |
| lblKhachHangMouseExited | KhachHangMouseExited(); |
| lblKhachHangMouseClicked | KhachHangMouseClicked(); |
| lblHoaDonMouseEntered | HoaDonMouseEntered(); |
| lblHoaDonMouseExited | HoaDonMouseExited(); |
| lblHoaDonMouseClicked | HoaDonMouseClicked(); |
| lblThongKeMouseEntered | ThongKeMouseEntered(); |
| lblThongKeMouseExited | ThongKeMouseExited(); |
| lblThongKeMouseClicked | ThongKeMouseClicked(); |
| lblUserMouseClicked | UserMouseClicked(); |
| pnlHeaderMousePressed | x = evt.getX();  y = evt.getY(); |
| pnlHeaderMouseDragged | int x1 = evt.getXOnScreen();  int y1 = evt.getYOnScreen();  this.setLocation(x1 - x, y1 - y); |
| btnMinimizeMouseClicked | MinimizeMouseClicked(); |
| btnMinimizeMouseEntered | MinimizeMouseEntered(); |
| btnMinimizeMouseExited | MinimizeMouseExited(); |
| btnExitMouseClicked | ExitMouseClicked(); |
| btnExitMouseEntered | ExitMouseEntered(); |
| btnExitMouseExited | ExitMouseExited(); |
| lblDangXuatMouseEntered | DangXuatMouseEntered(); |
| lblDangXuatMouseExited | DangXuatMouseExited(); |
| lblDangXuatMouseClicked |  |
| lblDoiMKMouseEntered | DoiMKMouseEntered(); |
| lblDoiMKMouseExited | DoiMKMouseExited(); |
| lblDoiMKMouseClicked |  |

**HƯỚNG DẪN CÀI ĐẶT MÃ NGUỒN**

|  |
| --- |
| boolean check = true;  int x;  int y;  void UserMouseClicked() {  if (check) {  lblLogout.setVisible(true);  check = false;  } else {  lblLogout.setVisible(false);  check = true;  }  }  void TrangChuMouseEntered() {  lblPanel\_Main.setIcon(new javax.swing.ImageIcon(getClass().getResource("/hinh/panel\_hover1.png")));  }  void TrangChuMouseExited() {  lblPanel\_Main.setIcon(new javax.swing.ImageIcon(getClass().getResource("/hinh/panel\_main.png")));  }  void TrangChuMouseClicked() {  try {  TrangChuJInternalFrame tcjif = new TrangChuJInternalFrame();  Desktop.removeAll();  Desktop.add(tcjif);  tcjif.show();  } catch (Exception e) {  System.out.println(e.toString());  }  }  void NhanVienMouseEntered() {  lblPanel\_Main.setIcon(new javax.swing.ImageIcon(getClass().getResource("/hinh/panel\_hover2.png")));  }  void NhanVienMouseExited() {  lblPanel\_Main.setIcon(new javax.swing.ImageIcon(getClass().getResource("/hinh/panel\_main.png")));  }  void NhanVienMouseClicked() {  if (ShareHelper.USER.getController() == false) {  DialogHepler.alert(this, "Chỉ có Admin được phép truy cập!");  } else {  try {  NhanVienJInternalFrame nvjif = new NhanVienJInternalFrame();  Desktop.removeAll();  Desktop.add(nvjif);  nvjif.show();  } catch (Exception e) {  System.out.println(e.toString());  }  }  }  void SanPhamMouseEntered() {  lblPanel\_Main.setIcon(new javax.swing.ImageIcon(getClass().getResource("/hinh/panel\_hover3.png")));  }  void SanPhamMouseExited() {  lblPanel\_Main.setIcon(new javax.swing.ImageIcon(getClass().getResource("/hinh/panel\_main.png")));  }  void SanPhamMouseClicked() {  if (ShareHelper.USER.getController() == false) {  DialogHepler.alert(this, "Chỉ có Admin được phép truy cập!");  } else {  try {  SanPhamJInternalFrame spjif = new SanPhamJInternalFrame();  Desktop.removeAll();  Desktop.add(spjif);  spjif.show();  } catch (Exception e) {  System.out.println(e.toString());  }  }  }  void KhachHangMouseEntered() {  lblPanel\_Main.setIcon(new javax.swing.ImageIcon(getClass().getResource("/hinh/panel\_hover4.png")));  }  void KhachHangMouseExited() {  lblPanel\_Main.setIcon(new javax.swing.ImageIcon(getClass().getResource("/hinh/panel\_main.png")));  }  void KhachHangMouseClicked() {  if (ShareHelper.USER.getController() == false) {  DialogHepler.alert(this, "Chỉ có Admin được phép truy cập!");  } else {  try {  KhachHangJInternalFrame khjif = new KhachHangJInternalFrame();  Desktop.removeAll();  Desktop.add(khjif);  khjif.show();  } catch (Exception e) {  System.out.println(e.toString());  }  }  }  void HoaDonMouseEntered() {  lblPanel\_Main.setIcon(new javax.swing.ImageIcon(getClass().getResource("/hinh/panel\_hover5.png")));  }  void HoaDonMouseExited() {  lblPanel\_Main.setIcon(new javax.swing.ImageIcon(getClass().getResource("/hinh/panel\_main.png")));  }  void HoaDonMouseClicked() {  try {  HDJInternalFrame HDJInternalFrame = new HDJInternalFrame();  Desktop.removeAll();  Desktop.add(HDJInternalFrame);  HDJInternalFrame.show();  } catch (Exception e) {  System.out.println(e.toString());  }  }  void ThongKeMouseEntered() {  lblPanel\_Main.setIcon(new javax.swing.ImageIcon(getClass().getResource("/hinh/panel\_hover6.png")));  }  void ThongKeMouseExited() {  lblPanel\_Main.setIcon(new javax.swing.ImageIcon(getClass().getResource("/hinh/panel\_main.png")));  }  void ThongKeMouseClicked() {  try {  ThongKeJInternalFrame tkjif = new ThongKeJInternalFrame();  Desktop.removeAll();  Desktop.add(tkjif);  tkjif.show();  } catch (Exception e) {  System.out.println(e.toString());  }  }  void DesktopAncestorAdded() {  try {  TrangChuJInternalFrame tcjif = new TrangChuJInternalFrame();  Desktop.removeAll();  Desktop.add(tcjif);  tcjif.show();  } catch (Exception e) {  System.out.println(e.toString());  }  }  void MinimizeMouseClicked() {  this.setState(Frame.ICONIFIED);  }  void MinimizeMouseEntered() {  btnMinimize.setBackground(new Color(0, 0, 0));  }  void MinimizeMouseExited() {  btnMinimize.setBackground(new Color(255, 255, 255));  }  void ExitMouseClicked() {  if (ShareHelper.USER.getController() == false) {  DialogHepler.alert(this, "Chỉ có Admin được phép tắt ứng dụng!");  } else {  System.exit(0);  }  }  void ExitMouseEntered() {  btnExit.setBackground(new Color(232, 17, 35));  }  void ExitMouseExited() {  btnExit.setBackground(new Color(255, 255, 255));  }  void DangXuatMouseEntered() {  lblLogout.setIcon(new javax.swing.ImageIcon(getClass().getResource("/hinh/logout-doimatkhau\_hover1.png")));  }  void DangXuatMouseExited() {  lblLogout.setIcon(new javax.swing.ImageIcon(getClass().getResource("/hinh/logout-doimatkhau.png")));  }  void DoiMKMouseEntered() {  lblLogout.setIcon(new javax.swing.ImageIcon(getClass().getResource("/hinh/logout-doimatkhau-hover2.png")));  }  void DoiMKMouseExited() {  lblLogout.setIcon(new javax.swing.ImageIcon(getClass().getResource("/hinh/logout-doimatkhau.png")));  } |

#### Nhân viên



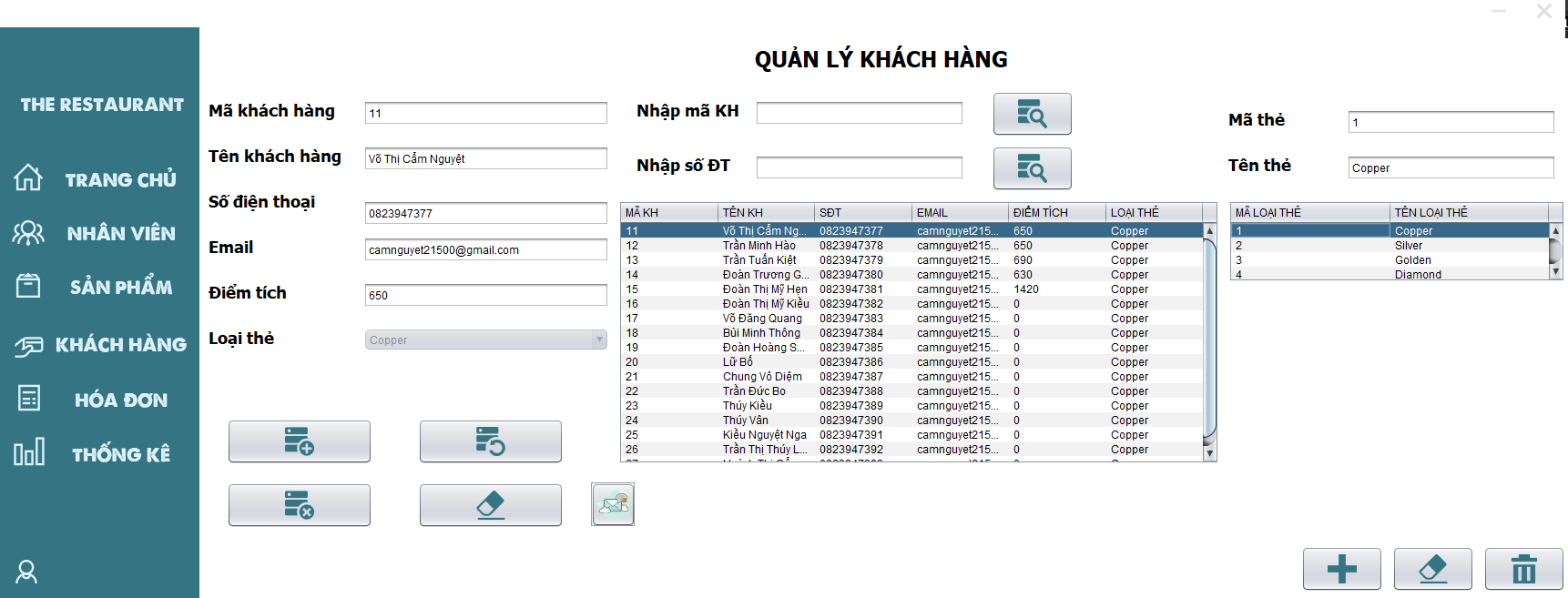
MÔ TẢ HOẠT ĐỘNG

|  |  |
| --- | --- |
| **Sự kiện** | **Công việc** |
| Constructor | RemoveHeader();  XuLi();  Load(); |
| tblNhanVienMouseClicked | this.Edit() |
| btnMoiActionPerformed | this.New() |
| btnSendGmailActionPerformed | sendEmailQRCode(); |
| lblImgMouseClicked | ResizeImage(); |

**HƯỚNG DẪN CÀI ĐẶT MÃ NGUỒN**

|  |
| --- |
| private String Path = "";  private String[] TieuDe = {"Mã Nhân Viên", "Tên Nhân Viên", "Chức Vụ", "Hình thức", "Email", "Số Điện Thoại"};  private DefaultTableModel table = new DefaultTableModel(TieuDe, 0);  private NhanVien nv = new NhanVien();  private NhanVienDAO DAO = new NhanVienDAO();  private List<NhanVien> list;  private int index = 0;  public NhanVienJInternalFrame() {  initComponents();  RemoveHeader();  XuLi();  Load();  }  private void XuLi() {  ButtonGroup a = new ButtonGroup();  a.add(rdoNu);  a.add(rdoNam);  }  private void RemoveHeader() {  this.setBorder(javax.swing.BorderFactory.createEmptyBorder(0, 0, 0, 0));  BasicInternalFrameUI bi = (BasicInternalFrameUI) this.getUI();  bi.setNorthPane(null);  }  private void Load() {  table.setRowCount(0);  try {  this.list = DAO.select();  for (NhanVien nv : list) {  Object[] row = {nv.getMaNV(), nv.getTenNV(), nv.getController() ? "Admin" : "Nhân viên", nv.getLoai() ? "Full-time" : "Part-time", nv.getEmail(), nv.getSDT()};  table.addRow(row);  tbNhanVien.setModel(table);  }  } catch (Exception e) {  System.out.println("Lỗi phần load dữ liệu " + e);  }  }  // fill Ngược  private void Edit() {  this.index = tbNhanVien.getSelectedRow();  try {  String MaNV = (String) tbNhanVien.getValueAt(index, 0);  NhanVien nv = DAO.FindbyID(MaNV);  System.out.println(nv.getMaNV());  System.out.println(nv.getController());  FillNguoc(nv);  } catch (Exception e) {  System.out.println("Lỗi phần fill ngược " + e);  }  }    private void New()  {  txtMaNV.setText("");  txtPass.setText("");  txtName.setText("");  txtSDT.setText("");  txtEmail.setText("");  txtDiaChi.setText("");  }  private void FillNguoc(NhanVien model) {  txtMaNV.setText(model.getMaNV());  txtPass.setText(model.getPass());  txtName.setText(model.getTenNV());  jdcDate.setDate(Date.from(model.getNgaySinh().atStartOfDay(ZoneId.systemDefault()).toInstant()));  if (model.getGioitinh()) {  rdoNam.setSelected(true);  } else {  rdoNu.setSelected(false);  }  txtSDT.setText(model.getSDT());  txtEmail.setText(model.getEmail());  if (nv.getLoai()) {  cbbLoai.setSelectedIndex(0);  } else {  cbbLoai.setSelectedIndex(1);  }  lbImg.setIcon(ResizeImage(model.getIMG()));  lblQRCode.setIcon(new ImageIcon(QRHelper.QRHelper.getQRCodeImage(model.getQRCode(), 150, 100)));  QRHelper.QRHelper.getQRImage(model.getQRCode(), 150, 100);  if (model.getController()) {  rbAmin.setSelected(true);  } else {  rbAmin.setSelected(false);  }  }  // Gửi Email mã QRcode cho nhân viên vừa đăng kí  private void sendEmailQRCode() throws AddressException, MessagingException, UnsupportedEncodingException {  Properties mailSeverPropeties;  Session getMailSession;  MimeMessage mailMessage;  // Khởi tạo Mail Server  mailSeverPropeties = System.getProperties();  mailSeverPropeties.put("mail.smtp.port", "587");  mailSeverPropeties.put("mail.smtp.auth", "true");  mailSeverPropeties.put("mail.smtp.starttls.enable", "true");  // Get Mail Session  getMailSession = Session.getDefaultInstance(mailSeverPropeties, null);  mailMessage = new MimeMessage(getMailSession);  mailMessage.addRecipient(Message.RecipientType.TO, new InternetAddress(txtEmail.getText())); // Thêm địa chỉ người nhận  // set Subject  mailMessage.setSubject("Nhà hàng The Restaurant gửi nhân viên mã QRCode");  // Tạo phần gửi gửi Message  BodyPart messagePart = new MimeBodyPart();  messagePart.setText("Nhà hàng The Restaurant gửi bạn " + txtName.getText() + " mã QRCode để đăng nhập hệ thống cũng như chấm công \n Lưu ý: Không được đưa mã này cho bất kì ai");  BodyPart filePart = new MimeBodyPart();  filePart.setFileName("QR CODE");  String qrz = txtMaNV.getText();  String qr = Base64.getMimeEncoder().encodeToString(qrz.getBytes("UTF-8"));  DataSource source = new FileDataSource(QRHelper.QRHelper.getQRImage(qr, 400, 400));  filePart.setDataHandler(new DataHandler(source));  // Gộp Message và File vào để gửi đi  Multipart multipart = new MimeMultipart();  multipart.addBodyPart(messagePart);  multipart.addBodyPart(filePart);  mailMessage.setContent(multipart);  // Gửi mail  Transport transpot = getMailSession.getTransport("smtp");  // truy cập mail  transpot.connect("smtp.gmail.com", "baubau2358@gmail.com", "giabao1998");  transpot.sendMessage(mailMessage, mailMessage.getAllRecipients());  transpot.close();  }  public ImageIcon ResizeImage(String ImagePath) {  ImageIcon IMG = new ImageIcon(ImagePath);  Image img = IMG.getImage();  Image newImg = img.getScaledInstance(lbImg.getWidth(), lbImg.getHeight(), Image.SCALE\_SMOOTH);  ImageIcon image = new ImageIcon(newImg);  return image;  }  public NhanVien getModel() {  nv.setMaNV(txtMaNV.getText());  nv.setPass(new String(txtPass.getPassword()));  nv.setTenNV(txtName.getText());  nv.setNgaySinh(DateConvert.DatetoLocal(jdcDate.getDate()));  if (rdoNam.isSelected()) {  nv.setGioitinh(true);  } else {  nv.setGioitinh(false);  }  nv.setSDT(txtSDT.getText());  nv.setEmail(txtEmail.getText());  if (cbbLoai.getSelectedItem().equals("Full-time")) {  nv.setLoai(true);  } else {  nv.setLoai(false);  }  nv.setIMG(Path);  nv.setQRCode(txtMaNV.getText());  if (rbAmin.isSelected()) {  nv.setController(true);  } else {  nv.setController(false);  }  nv.setDiaChi(txtDiaChi.getText());  return nv;  }  private void lbImgMouseClicked(java.awt.event.MouseEvent evt) {  try {  JFileChooser file = new JFileChooser();  int result = file.showOpenDialog(this);  if (result == JFileChooser.APPROVE\_OPTION) {  File seletedFile = file.getSelectedFile();  FileInputStream input = new FileInputStream(seletedFile);  String TenHinh = seletedFile.getName();  String chuoiLuu = Paths.get("").toAbsolutePath() + "/Img/" + TenHinh;  FileOutputStream output = new FileOutputStream(chuoiLuu);  System.out.println(chuoiLuu);  byte[] buffer = new byte[1024];  int len;  while ((len = input.read(buffer)) > 0) {  output.write(buffer, 0, len);  }  input.close();  output.close();  Path = Paths.get("").toAbsolutePath().toString() + "/Img/" + TenHinh;  lbImg.setIcon(ResizeImage(Path));  } else {  if (result == JFileChooser.CANCEL\_OPTION) {  JOptionPane.showMessageDialog(this, "Không có ảnh nào được chọn");  lbImg.setText("NONE");  }  }  } catch (Exception e) {  System.out.println(" " + e.toString());  }  }  private void tbNhanVienMouseClicked(java.awt.event.MouseEvent evt) {  if (evt.getClickCount() == 1) {  index = tbNhanVien.getSelectedRow();  this.Edit();  }  }  private void btTaoActionPerformed(java.awt.event.ActionEvent evt) {  try {  nv = getModel();  DAO.insertNV(nv);  this.Load();  JOptionPane.showMessageDialog(this, "Thêm thành công");  } catch (Exception e) {  System.out.println("Lỗi phần thêm nhân viên " + e);  }  }  private void btXoaActionPerformed(java.awt.event.ActionEvent evt) {  nv = getModel();  try {  int click = JOptionPane.showConfirmDialog(this, "Bạn có chắc chắn muốn xóa nhân viên này?");  if (click == JOptionPane.YES\_OPTION) {  DAO.DelNV(nv.getMaNV());  this.Load();  JOptionPane.showMessageDialog(this, "Đã xóa nhân viên");  } else {  JOptionPane.showMessageDialog(this, "Bạn rảnh ghê");  }  } catch (Exception e) {  System.out.println("Lỗi Xóa nhân viên " + e);  }  }  private void btSuaActionPerformed(java.awt.event.ActionEvent evt) {  try {  nv = getModel();  DAO.updateNV(nv);  Load();  DialogHepler.alert(this, "Sửa thông tin thành công!");  } catch (Exception e) {  e.printStackTrace();  DialogHepler.alert(this, "Sửa thông tin thất bại!");  }  }  private void btQRCodeMouseClicked(java.awt.event.MouseEvent evt) {  String Qr = txtMaNV.getText();  lblQRCode.setIcon(new ImageIcon(QRHelper.QRHelper.getQRCodeImage(Qr, 150, 100)));  }  private void btSendGmailActionPerformed(java.awt.event.ActionEvent evt) {  try {  sendEmailQRCode();  } catch (MessagingException ex) {  Logger.getLogger(NhanVienJInternalFrame.class.getName()).log(Level.SEVERE, null, ex);  } catch (UnsupportedEncodingException ex) {  Logger.getLogger(NhanVienJInternalFrame.class.getName()).log(Level.SEVERE, null, ex);  }  }  private void btnMoiActionPerformed(java.awt.event.ActionEvent evt) {  New();  } |

#### Khách hàng



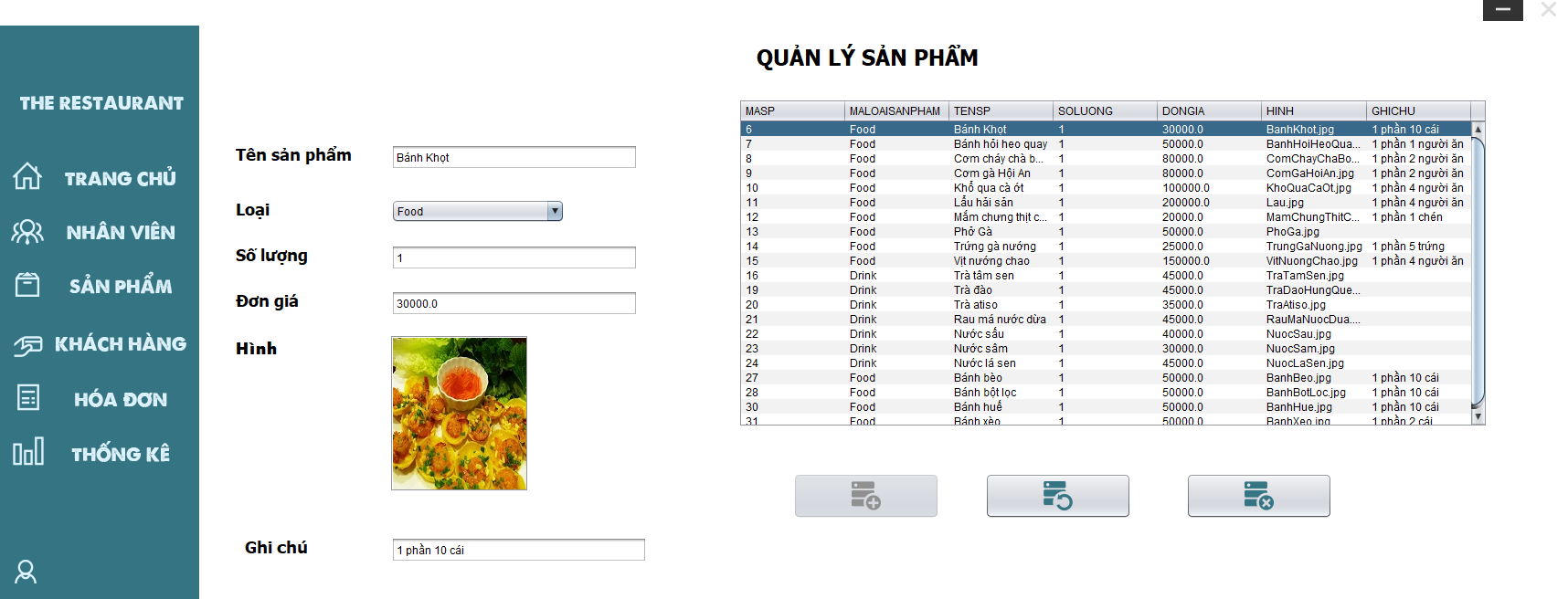
**MÔ TẢ HOẠT ĐỘNG**

|  |  |
| --- | --- |
| **Sự kiện** | **Công việc** |
| Constructor | this.RemoveHeader();  this.loadKH();  this.loadTheKH();  this.loadComboBox();  this.disableDiemLoaiThe(); |
| tblLoaiTheMouseClicked | LoaiTheMouseClicked(); |
| tblGridViewMouseClicked | GridViewMouseClicked(); |
| btnLamMoiKHActionPerformed | this.clear(); |
| btnXoaActionPerformed | this.delete(); |
| btnSuaActionPerformed | this.update(); |
| btnThemActionPerformed | this.insert(); |
| btnXoaLoaiTheActionPerformed | this.deleteLT(); |
| btnLamMoiLTActionPerformed | this.clearLT(); |
| btnThemLoaiTheActionPerformed | this.insertLT(); |
| btnTimSDTActionPerformed | this.timSDT(); |
| btnTimMaActionPerformed | this.timMaKH(); |
| btnSendMailActionPerformed | try {  sendEmailQRCode();  } catch (MessagingException ex) {  Logger.getLogger(NhanVienJInternalFrame.class.getName()).log(Level.SEVERE, null, ex);  } catch (UnsupportedEncodingException ex) {  Logger.getLogger(NhanVienJInternalFrame.class.getName()).log(Level.SEVERE, null, ex);  } |

**HƯỚNG DẪN CÀI ĐẶT MÃ NGUỒN**

|  |
| --- |
| int index = 0;  TheKhachHangDAO khdao = new TheKhachHangDAO();  LoaiTheKHDAO ltdao = new LoaiTheKHDAO();  public void loadKH() {  DefaultTableModel model = (DefaultTableModel) tblGridView.getModel();  model.setRowCount(0);  try {  List<TheKhachHang> list = khdao.select();  for (TheKhachHang kh : list) {  Object[] row = {  kh.getMaKH(),  kh.getTenKH(),  kh.getSDT(),  kh.getEmail(),  kh.getDiemTich(),  ltdao.findten(kh.getMaLoaiThe())  };  model.addRow(row);  }  } catch (Exception e) {  DialogHepler.alert(this, "Lỗi truy vấn dữ liệu Khách Hàng!");  }  }  public void timMaKH() {  DefaultTableModel model = (DefaultTableModel) tblGridView.getModel();  model.setRowCount(0);  try {  String MaKH = txtTimMaKH.getText();  List<TheKhachHang> list = khdao.selectByMaKH(MaKH);  for (TheKhachHang kh : list) {  Object[] row = {  kh.getMaKH(),  kh.getTenKH(),  kh.getSDT(),  kh.getEmail(),  kh.getDiemTich(),  kh.getMaLoaiThe()  };  model.addRow(row);  }  } catch (Exception e) {  DialogHepler.alert(this, "Không tìm được mã khách hàng!");  }  }  public void timSDT() {  DefaultTableModel model = (DefaultTableModel) tblGridView.getModel();  model.setRowCount(0);  try {  String SDT = txtNhapSoDT.getText();  List<TheKhachHang> list = khdao.selectBySDT(SDT);  for (TheKhachHang kh : list) {  Object[] row = {  kh.getMaKH(),  kh.getTenKH(),  kh.getSDT(),  kh.getEmail(),  kh.getDiemTich(),  kh.getMaLoaiThe()  };  model.addRow(row);  }  } catch (Exception e) {  DialogHepler.alert(this, "Không tìm được số điện thoại khách hàng!");  }  }  // Gửi email cho khách hàng  private void sendEmailQRCode() throws AddressException, MessagingException, UnsupportedEncodingException {  Properties mailSeverPropeties;  Session getMailSession;  MimeMessage mailMessage;  // Khởi tạo Mail Server  mailSeverPropeties = System.getProperties();  mailSeverPropeties.put("mail.smtp.port", "587");  mailSeverPropeties.put("mail.smtp.auth", "true");  mailSeverPropeties.put("mail.smtp.starttls.enable", "true");  // Get Mail Session  getMailSession = Session.getDefaultInstance(mailSeverPropeties, null);  mailMessage = new MimeMessage(getMailSession);  mailMessage.addRecipient(Message.RecipientType.TO, new InternetAddress(txtEmail.getText())); // Thêm địa chỉ người nhận  // set Subject  mailMessage.setSubject("Nhà hàng The Restaurant gửi nhân viên mã QRCode");  // Tạo phần gửi gửi Message  BodyPart messagePart = new MimeBodyPart();  messagePart.setText("Nhà hàng The Reastaurant gửi bạn " + txtTenKH.getText() + " mã QRCode để đăng nhập tích điểm và nhận ưu đãi");  BodyPart filePart = new MimeBodyPart();  filePart.setFileName("QR CODE");  String qrz = txtSDT.getText();  String qr = Base64.getMimeEncoder().encodeToString(qrz.getBytes("UTF-8"));  DataSource source = new FileDataSource(QRHelper.QRHelper.getQRImage(qr, 400, 400));  filePart.setDataHandler(new DataHandler(source));  // Gộp Message và File vào để gửi đi  Multipart multipart = new MimeMultipart();  multipart.addBodyPart(messagePart);  multipart.addBodyPart(filePart);  mailMessage.setContent(multipart);  // Gửi mail  Transport transpot = getMailSession.getTransport("smtp");  // truy cập mail  transpot.connect("smtp.gmail.com", "baubau2358@gmail.com", "giabao1998");  transpot.sendMessage(mailMessage, mailMessage.getAllRecipients());  transpot.close();  }  public void loadTheKH() {  DefaultTableModel model = (DefaultTableModel) tblLoaiThe.getModel();  model.setRowCount(0);  try {  List<LoaiTheKH> list = ltdao.select();  for (LoaiTheKH lt : list) {  Object[] row = {lt.getMaLoaiKhachHang(), lt.getTenLoai()};  model.addRow(row);  }  } catch (Exception e) {  DialogHepler.alert(this, "Lỗi truy vấn dữ liệu Loại Thẻ!");  }  }  public void loadComboBox() {  ResultSet rs = null;  try {  String sql = "select \* from LOAITHEKHACHHANG";  rs = JdbcHelper.executeQuery(sql);  while (rs.next()) {  String tenLoai = rs.getString("TENLOAI");  cboLoaiThe.addItem(tenLoai);  }  } catch (Exception e) {  }  }  public void insertLT() {  String model = txtTenThe.getText();  try {  ltdao.insert(model);  this.loadTheKH();  this.clearLT();  DialogHepler.alert(this, "Thêm thành công!");  } catch (Exception e) {  DialogHepler.alert(this, "Thêm thất bại!");  System.out.println(e.getMessage());  }  }  public void insert() {  TheKhachHang model = getModel();  try {  khdao.insert(model);  this.loadKH();  this.clear();  DialogHepler.alert(this, "Thêm thành công!");  } catch (Exception e) {  System.out.println(e.getMessage());  DialogHepler.alert(this, "Thêm thất bại!");  }  }  public void update() {  try {  TheKhachHang model = getModel();  khdao.update(model);  this.loadKH();  this.clear();  DialogHepler.alert(this, "Sửa thành công!");  } catch (Exception e) {  System.out.println(e.getMessage());  DialogHepler.alert(this, "Sửa thất bại!");  }  }  public void deleteLT() {  if (DialogHepler.confirm(this, "Bạn thực sự muốn xóa loại thẻ này!")) {  String malt = txtMaThe.getText();  try {  ltdao.Delete(malt);  this.loadTheKH();  this.clearLT();  DialogHepler.alert(this, "Xóa thành công!");  } catch (Exception e) {  DialogHepler.alert(this, "Xóa thất bại!");  }  }  }  public void delete() {  if (DialogHepler.confirm(this, "Bạn thực sự muốn xóa khách hàng này!")) {  String makh = txtMaKH.getText();  try {  khdao.delete(makh);  this.loadKH();  this.clear();  DialogHepler.alert(this, "Xóa thành công!");  } catch (Exception e) {  DialogHepler.alert(this, "Xóa thất bại!");  }  }  }  public void disableDiemLoaiThe() {  cboLoaiThe.setEnabled(false);  txtDiemTich.setEditable(false);  }  void LoaiTheMouseClicked() {  int \_index = tblLoaiThe.getSelectedRow();  String ma = tblLoaiThe.getValueAt(\_index, 0).toString();  LoaiTheKH ltkh = ltdao.findById(ma);  this.setModel(ltkh);  }    void GridViewMouseClicked() {  int \_index = tblGridView.getSelectedRow();  String ma = tblGridView.getValueAt(\_index, 0).toString();  TheKhachHang tkh = khdao.findById(ma);  this.setModel(tkh);  this.disableDiemLoaiThe();  }  public void clear() {  this.setModel(new TheKhachHang());  this.disableDiemLoaiThe();  cboLoaiThe.setSelectedIndex(0);  }  public void clearLT() {  this.setModel(new LoaiTheKH());  }  private void RemoveHeader() {  this.setBorder(javax.swing.BorderFactory.createEmptyBorder(0, 0, 0, 0));  BasicInternalFrameUI bi = (BasicInternalFrameUI) this.getUI();  bi.setNorthPane(null);  }  public void setModel(TheKhachHang model) {  txtMaKH.setText(Integer.toString(model.getMaKH()));  txtTenKH.setText(model.getTenKH());  txtSDT.setText(model.getSDT());  txtEmail.setText(model.getEmail());  txtDiemTich.setText(Integer.toString(model.getDiemTich()));  cboLoaiThe.setSelectedIndex(model.getMaLoaiThe() - 1);  }  public void setModel(LoaiTheKH model) {  txtMaThe.setText(Integer.toString(model.getMaLoaiKhachHang()));  txtTenThe.setText(model.getTenLoai());  }  TheKhachHang getModel() {  TheKhachHang model = new TheKhachHang();  model.setMaKH(Integer.parseInt(txtMaKH.getText()));  model.setTenKH(txtTenKH.getText());  model.setSDT(txtSDT.getText());  model.setEmail(txtEmail.getText());  model.setDiemTich(Integer.parseInt(txtDiemTich.getText()));  model.setMaLoaiThe(cboLoaiThe.getSelectedIndex() + 1);  return model;  }  LoaiTheKH getModelLT() {  LoaiTheKH model = new LoaiTheKH();  model.setMaLoaiKhachHang(Integer.parseInt(txtMaThe.getText()));  model.setTenLoai(txtTenThe.getText());  return model;  } |

#### Sản phẩm



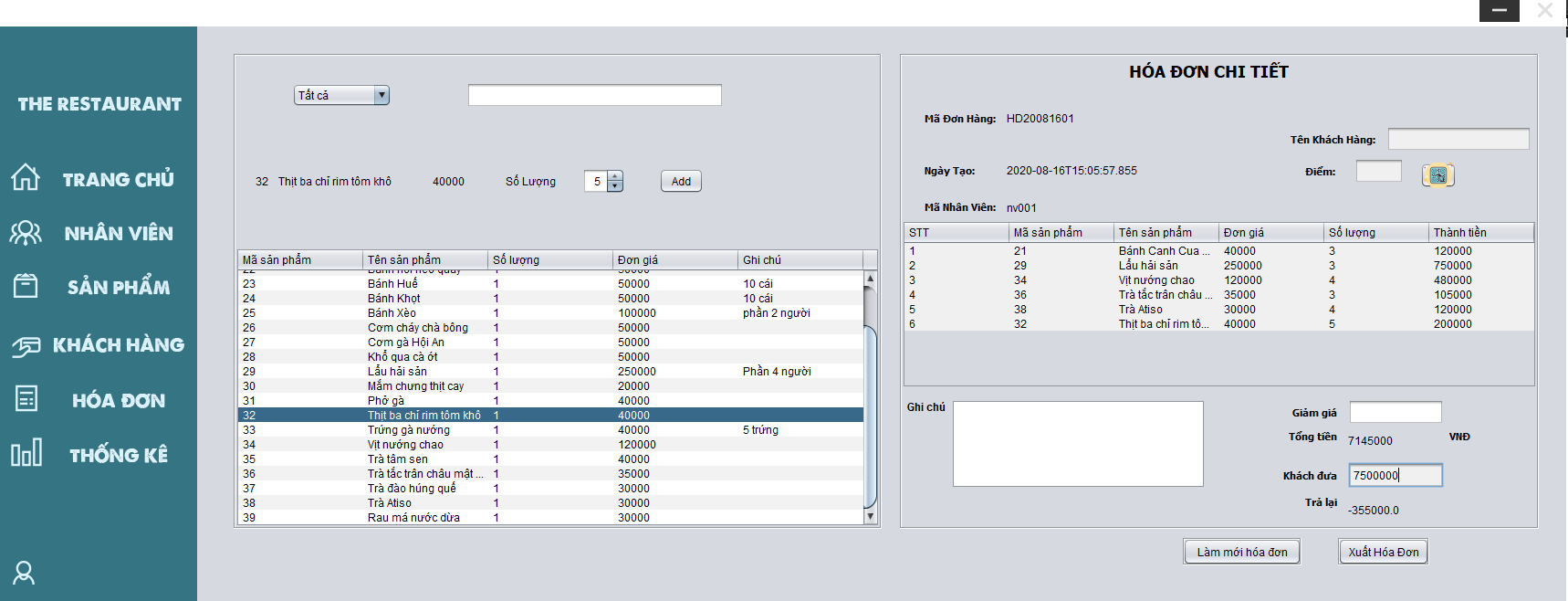
**MÔ TẢ HOẠT ĐỘNG**

|  |  |
| --- | --- |
| **Sự kiện** | **Công việc** |
| Constructor | this.RemoveHeader();  this.setTitle("Sản phẩm");  load();  fillComboBox();  clear();  this.setStatus(true);  selectComboBox();  tblGridView.setDefaultEditor(Object.class, null); |
| tblGridViewMouseClicked | if (evt.getClickCount() == 1) {  index = tblGridView.getSelectedRow();  this.edit();  } |
| btnThem | insert(); |
| btnSua | update(); |
| btnXoa | delete(); |
| lblHinhMouseClicked | this.selectImage(); |

**HƯỚNG DẪN CÀI ĐẶT MÃ NGUỒN**

|  |
| --- |
| public SanPhamJInternalFrame() {  initComponents();  this.RemoveHeader();  this.setTitle("Sản phẩm");  load();  fillComboBox();  clear();  this.setStatus(true);  selectComboBox();  tblGridView.setDefaultEditor(Object.class, null);  }  int index = 0;  SanPhamDAO dao = new SanPhamDAO();  LoaiSPDAO loaidao = new LoaiSPDAO();  void load() {  DefaultTableModel model = (DefaultTableModel) tblGridView.getModel();  model.setRowCount(0);  try {  List<SanPham> list = dao.select();  for (SanPham cd : list) {  Object[] row  = {  cd.getMASP(),  cd.getMALOAISANPHAM(),  cd.getTENSP(),  cd.getSOLUONG(),  cd.getDONGIA(),  cd.getHINH(),  cd.getGHICHU()  };  model.addRow(row);  }  } catch (Exception e) {  DialogHepler.alert(this, "Lỗi truy vấn dữ liệu!");  }  }  void insert() {  SanPham model = getModel();  try {  dao.insert(model);  this.load();  this.clear();  DialogHepler.alert(this, "Thêm mới thành công!");  } catch (Exception e) {  DialogHepler.alert(this, "Thêm mới thất bại!");  e.printStackTrace();  }  }  void update() {  SanPham model = getModel();  try {  dao.update(model);  this.load();  DialogHepler.alert(this, "Cập nhật thành công!");  } catch (Exception e) {  DialogHepler.alert(this, "Cập nhật thất bại!");  }  }  void delete() {  if (DialogHepler.confirm(this, "Bạn có muốn xóa hay không?")) {  Integer masp = Integer.valueOf(cboLoai.getToolTipText());  try {  dao.delete(masp);  this.load();  this.clear();  DialogHepler.alert(this, "Xóa thành công!");  } catch (Exception e) {  DialogHepler.alert(this, "Xóa thất bại!");  }  }  }  void clear() {  SanPham model = new SanPham();  LoaiSP loaisp = (LoaiSP) cboLoai.getSelectedItem();  model.setMALOAISANPHAM(loaisp.getMALOAISANPHAM());  this.setModel(model);  }  void edit() {  try {  Integer masp = (Integer) tblGridView.getValueAt(this.index, 0);  String loaisp = (String) tblGridView.getValueAt(this.index, 1);    List<LoaiSP> ds = loaidao.select();  int indextam = 0;    for (int i = 0; i < ds.size(); i++)  {  if (ds.get(i).getMALOAISANPHAM().equals(loaisp))  {  indextam = i;  break;  }  }  cboLoai.setSelectedIndex(indextam);  SanPham model = dao.findById(masp);  if (model != null) {  this.setModel(model);  this.setStatus(false);  }  } catch (Exception e) {  DialogHepler.alert(this, "Lỗi truy vấn dữ liệu!");  }  }  void setModel(SanPham model) {  cboLoai.setToolTipText(String.valueOf(model.getMASP()));  cboLoai.setSelectedItem(loaidao.findById(model.getMALOAISANPHAM()));  txtTenSP.setText(model.getTENSP());  txtSoLuong.setText(String.valueOf(model.getSOLUONG()));  txtDonGia.setText(String.valueOf(model.getDONGIA()));  lblHinh.setToolTipText(model.getHINH());  if (model.getHINH() != null) {  lblHinh.setIcon(ShareHelper.readLogo(model.getHINH()));  }  txtGhiChu.setText(model.getGHICHU());  }  SanPham getModel() {  SanPham model = new SanPham();  LoaiSP loaisp = (LoaiSP) cboLoai.getSelectedItem();  model.setMALOAISANPHAM(loaisp.getMALOAISANPHAM());  model.setTENSP(txtTenSP.getText());  model.setSOLUONG(Integer.valueOf(txtSoLuong.getText()));  model.setDONGIA(Integer.valueOf(txtDonGia.getText()));  model.setHINH(lblHinh.getToolTipText());  model.setGHICHU(txtGhiChu.getText());  model.setMASP(Integer.valueOf(cboLoai.getToolTipText()));  return model;  }  void setStatus(boolean insertable) {  btnThem.setEnabled(insertable);  btnSua.setEnabled(!insertable);  btnXoa.setEnabled(!insertable);  boolean first = this.index > 0;  boolean last = this.index < tblGridView.getRowCount() - 1;  }  void selectImage() {  if (jFileChooser1.showOpenDialog(this) == JFileChooser.APPROVE\_OPTION) {  File file = jFileChooser1.getSelectedFile();  if (ShareHelper.saveLogo(file)) {  // Hiển thị hình lên form  lblHinh.setIcon(ShareHelper.readLogo(file.getName()));  lblHinh.setToolTipText(file.getName());  }  }  }    void selectComboBox()  {  LoaiSP loaisp = (LoaiSP) cboLoai.getSelectedItem();  }    void fillComboBox()  {  DefaultComboBoxModel model = (DefaultComboBoxModel) cboLoai.getModel();  model.removeAllElements();  try {  List<LoaiSP> list = loaidao.select();  for (LoaiSP loai :list)  {  model.addElement(loai);  }  } catch (Exception e) {  DialogHepler.alert(this, "Lỗi truy vấn dữ liệu");  }  }  private void RemoveHeader() {  this.setBorder(javax.swing.BorderFactory.createEmptyBorder(0, 0, 0, 0));  BasicInternalFrameUI bi = (BasicInternalFrameUI) this.getUI();  bi.setNorthPane(null);  } |

#### Hóa đơn



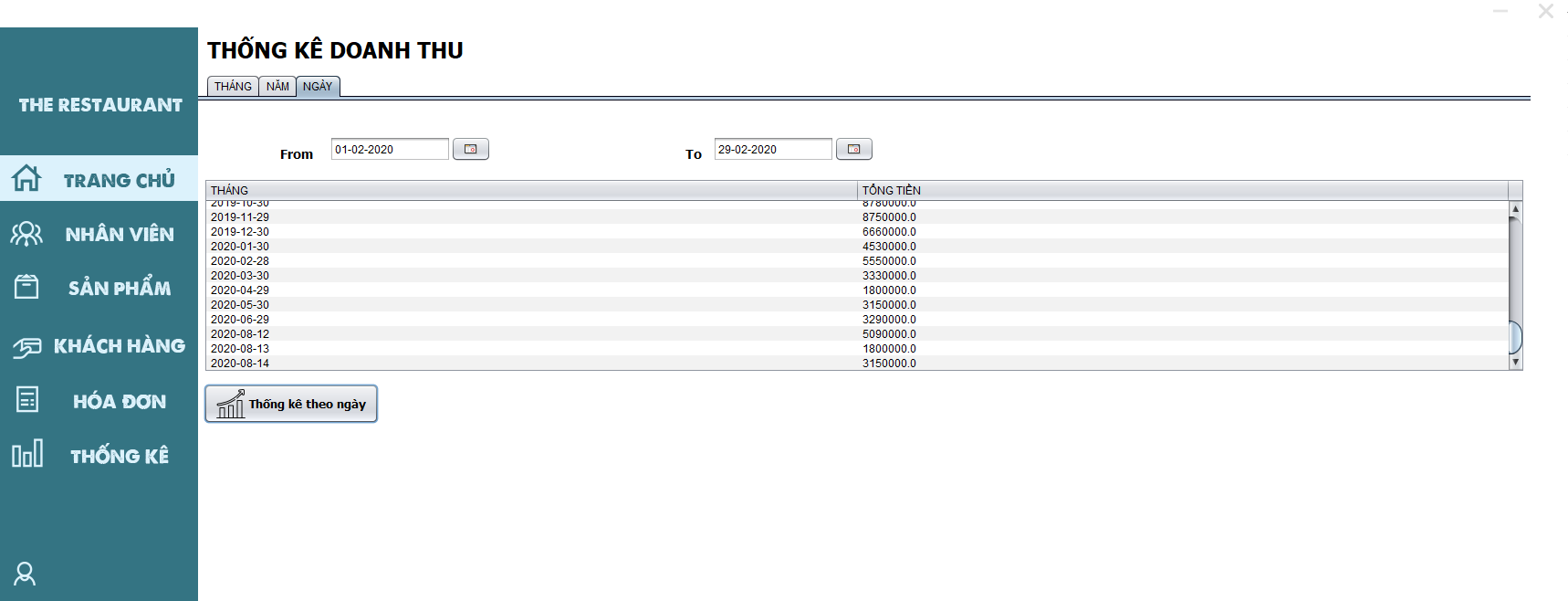
**MÔ TẢ HOẠT ĐỘNG**

|  |  |
| --- | --- |
| **Sự kiện** | **Công việc** |
| Constructor | this.RemoveHeader();  LoadDBSP();  fillTableFollowbyMaLoai();  filltoCBB();  lblNgayTao.setText(LocalDateTime.now().toString());  lblMaDH.setText(SoHD()); |
| btnAddActionPerformed | TongTien(); |
| tblSanPhamMouseClicked | NhapSoLuong(); |
| btnXuatHoaDonActionPerformed | ThanhToan();  New();  SoHD(); |
| btnLamMoiHDActionPerformed | New(); |

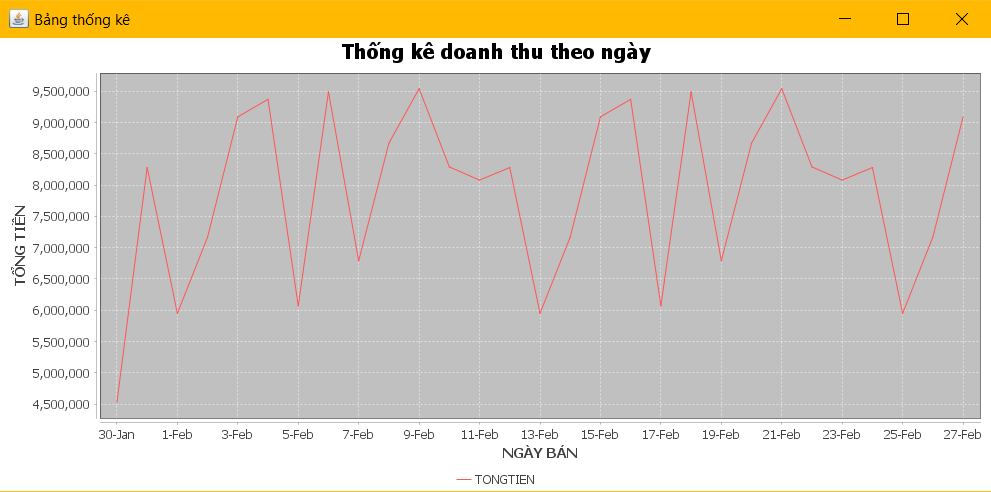
**HƯỚNG DẪN CÀI ĐẶT MÃ NGUỒN**

|  |
| --- |
| private String[] TieuDeSP = {"Mã sản phẩm", "Tên sản phẩm", "Số lượng", "Đơn giá", "Ghi chú"};  private String[] TieuDeHD = {"STT", "Mã sản phẩm", "Tên sản phẩm", "Đơn giá", "Số lượng", "Thành tiền"};  private DefaultTableModel table = new DefaultTableModel(TieuDeSP, 0);  private DefaultTableModel tableHD = new DefaultTableModel(TieuDeHD, 0);  private SanPhamDAO spDAO = new SanPhamDAO();  private HoaDonDAO dhDAO = new HoaDonDAO();  private HoaDonChiTietDAO hdctDAO = new HoaDonChiTietDAO();  private LoaiSPDAO loaiSPDAO = new LoaiSPDAO();  public static TheKhachHang khachHang = null;  private TheKhachHangDAO khDAO = new TheKhachHangDAO();  public HDJInternalFrame() {  initComponents();  this.RemoveHeader();  LoadDBSP();  fillTableFollowByMaLoai();  filltoCBB();  lbNgayTao.setText(LocalDateTime.now().toString());  lbMaDH.setText(SoHD());  }  private void RemoveHeader() {  this.setBorder(javax.swing.BorderFactory.createEmptyBorder(0, 0, 0, 0));  BasicInternalFrameUI bi = (BasicInternalFrameUI) this.getUI();  bi.setNorthPane(null);  }  // Tính tổng tiền của hóa đơn  int tongtien = 0;  private void TongTien() {  for (int index = 0; index < tbHoaDon.getRowCount(); index++) {  int thanhtien = Integer.parseInt(tbHoaDon.getValueAt(index, 5).toString());  tongtien += thanhtien;  }  String TongTien = String.valueOf(tongtien);  lbTongTien.setText(TongTien);  }  // Khách đưa và trả lại  private void tfKhachDuaKeyReleased(java.awt.event.KeyEvent evt) {  try {  Double Khachdua = Double.parseDouble(tfKhachDua.getText());  Double ThoiTien = tongtien - Khachdua;  String thoiTien = String.valueOf(ThoiTien);  lbTienThoi.setText(thoiTien);  } catch (Exception e) {  JOptionPane.showMessageDialog(this, "Nhập cho đàng hoàng vào" + e);  }  }  // Lọc theo Loại SP  private void fillTableFollowByMaLoai() {  table.setRowCount(0);  if (cbbLoai.getSelectedIndex() == 0) {  List<SanPham> list = spDAO.select();  for (SanPham cd : list) {  Object[] row  = {  cd.getMASP(),  cd.getTENSP(),  cd.getSOLUONG(),  cd.getDONGIA(),  cd.getGHICHU()  };  table.addRow(row);  tbSanPham.setModel(table);  }  }  if (cbbLoai.getSelectedItem() == null) {  } else {  try {  String maloai = String.valueOf(cbbLoai.getSelectedIndex());  List<SanPham> list = spDAO.getSanPhamByMaLoai2(maloai);  for (SanPham cd : list) {  Object[] row  = {  cd.getMASP(),  cd.getTENSP(),  cd.getSOLUONG(),  cd.getDONGIA(),  cd.getGHICHU()  };  table.addRow(row);  tbSanPham.setModel(table);  }  } catch (Exception e) {  e.printStackTrace();  }  }  }  // Đổ dữ liệu vào comboboxx  private void filltoCBB() {  DefaultComboBoxModel cbb = (DefaultComboBoxModel) cbbLoai.getModel();  List<LoaiSP> loai = loaiSPDAO.select();  for (LoaiSP sp : loai) {  cbb.addElement(sp);  }  }  private void btnQRKHActionPerformed(java.awt.event.ActionEvent evt) {  QRforBill a = new QRforBill();  a.setVisible(true);  }  // Show vào các label để nhập số lượng  private void NhapSoLuong() {  int index = tbSanPham.getSelectedRow();  String TenSP = (String) tbSanPham.getValueAt(index, 1);  String MaSP = tbSanPham.getValueAt(index, 0).toString();  Double DonGia = Double.parseDouble(tbSanPham.getValueAt(index, 3).toString());  String Dongia = String.valueOf(tbSanPham.getValueAt(index, 3));  jSpinner1.setValue(1);  lbMaMonAn.setText(MaSP);  lbTenMonAn.setText(TenSP);  lbDonGia.setText(Dongia);  }  private void tbSanPhamMouseClicked(java.awt.event.MouseEvent evt) {  if (evt.getClickCount() >= 1) {  NhapSoLuong();  }  }  // Giảm giá theo từng hóa đơn  public static void GiamGia(int DiemHienCo) {  if (khachHang != null) {  tfTenKH.setText(khachHang.getTenKH());  tfDiem.setText(String.valueOf(khachHang.getDiemTich()));  }  if (DiemHienCo < 1000) {  tfGiamGia.setText("0");  } else if (DiemHienCo >= 1000 && DiemHienCo < 2000) {  tfGiamGia.setText("5%");  int TongTien = (Integer.parseInt(lbTongTien.getText()) \* 5) / 100;  lbTongTien.setText(String.valueOf(TongTien));  } else if (DiemHienCo >= 2000 && DiemHienCo < 3000) {  tfGiamGia.setText("10%");  int TongTien2 = (Integer.parseInt(lbTongTien.getText()) \* 10) / 100;  lbTongTien.setText(String.valueOf(TongTien2));  } else if (DiemHienCo >= 3000 && DiemHienCo < 5000) {  tfGiamGia.setText("15%");  int TongTien3 = (Integer.parseInt(lbTongTien.getText()) \* 15) / 100;  lbTongTien.setText(String.valueOf(TongTien3));  } else if (DiemHienCo >= 5000) {  tfGiamGia.setText("30%");  int TongTien4 = (Integer.parseInt(lbTongTien.getText()) \* 30) / 100;  lbTongTien.setText(String.valueOf(TongTien4));  }  }  private void tfSearchKeyReleased(java.awt.event.KeyEvent evt) {  String tensp = tfSearch.getText();  String maloai = String.valueOf(cbbLoai.getSelectedIndex());  List<SanPham> listAll = spDAO.findSanPhamAllTK(tensp);  List<SanPham> list = spDAO.findSanPhamTK(tensp, maloai);  table.setRowCount(0);  if (cbbLoai.getSelectedIndex() == 0) {  for (SanPham cd : listAll) {  Object[] row  = {  cd.getMASP(),  cd.getTENSP(),  cd.getSOLUONG(),  cd.getDONGIA(),  cd.getGHICHU()  };  table.addRow(row);  tbSanPham.setModel(table);  }  } else {  for (SanPham cd : list) {  Object[] row  = {  cd.getMASP(),  cd.getTENSP(),  cd.getSOLUONG(),  cd.getDONGIA(),  cd.getGHICHU()  };  table.addRow(row);  tbSanPham.setModel(table);  }  }  }  // Thêm sản phẩm vào table Hóa đơn  int index = 1;  private void btAddActionPerformed(java.awt.event.ActionEvent evt) {  String MaSP = lbMaMonAn.getText();  String TenSP = lbTenMonAn.getText();  String Dongia = lbDonGia.getText();  int Soluong = Integer.parseInt(jSpinner1.getValue().toString());  String soLuong = String.valueOf(Soluong);  int DonGia = Integer.parseInt(Dongia);  int ThanhTien = DonGia \* Soluong;  String Thanhtien = String.valueOf(ThanhTien);  Object[] row = {index, MaSP, TenSP, DonGia, soLuong, ThanhTien};  tableHD.addRow(row);  tbHoaDon.setModel(tableHD);  index++;  TongTien();  }  // Thêm HD  private void ThanhToan() {  if (tbHoaDon.getRowCount() <= 0) {  JOptionPane.showMessageDialog(this, "Chưa có sản phẩm nào được chọn");  return;  }  DonHang dh = new DonHang();  dh.setMaDH(lbMaDH.getText());  dh.setMaNV(lbMaNV.getText());  dh.setNgayBan(LocalDate.now());  dh.setMaKH(32);  dh.setTenKH(tfTenKH.getText());  dh.setTongTien(Float.valueOf(lbTongTien.getText()));    // // tiến hành tích điểm  if (khachHang != null) {  TichDiem(Integer.valueOf(khachHang.getMaKH()), Integer.parseInt(lbTongTien.getText()));  }  // insert vào hóa đơn  dhDAO.insert(dh);  // Set MaHD  DonHang ab = dhDAO.FindbyID(dh.getMaDH());  dh.setMaDH(ab.getMaDH());  // insert vao hdct  ThemHDCT(ab.getMaDH());  System.out.println(ab.getMaDH());  // Xuất hóa đơn  XuatHoaDon(ab.getMaDH());  }  // add từ table vào CTHD  private void ThemHDCT(String MaHD) {  String Notes = tfGhiChu.getText();  float Giamgia = Float.valueOf(tfGiamGia.getText());  for (int i = 0; i < tbHoaDon.getRowCount(); i++) {  int Masp = Integer.parseInt(tbHoaDon.getValueAt(i, 1).toString());  String TenSP = String.valueOf(tbHoaDon.getValueAt(i, 2));  float DonGia = Float.parseFloat(tbHoaDon.getValueAt(i, 3).toString());  int Soluong = Integer.parseInt(tbHoaDon.getValueAt(i, 4).toString());  float Thanhtien = Float.parseFloat(tbHoaDon.getValueAt(i, 5).toString());  CTHD(MaHD, Masp, TenSP, Soluong, DonGia, Giamgia, Thanhtien, Notes);  }  }  // Tích điểm cho khách hàng  public void TichDiem(int Makh, int Money) {  int DiemHienCo = 0;  if (Money >= 500000) {  int Score = (Money / 50000) \* 10;  if (tfDiem.getText() != null) {  DiemHienCo = Integer.parseInt(tfDiem.getText());  DiemHienCo += Score;  khDAO.UpdateDiem(DiemHienCo, Makh);  }  }  }  // Số Hóa đơn  private DateTimeFormatter format = DateTimeFormatter.ofPattern("yyMMdd");  private LocalDate date = LocalDate.now();  private String datez = date.format(format);  private int STT = 0;  private String SoHD() {  String SoHoaDon = "HD" + datez;  STT++;  if (STT < 9) {  SoHoaDon = SoHoaDon + "0" + STT;  System.out.println(STT);  return SoHoaDon;  }  if (STT < 99) {  SoHoaDon = SoHoaDon + "0" + STT;  return SoHoaDon;  }  if (STT > 99) {  SoHoaDon = SoHoaDon + STT;  return SoHoaDon;  }  return SoHoaDon;  }  // Thêm chi tiết hóa đơn  private void CTHD(String MaHD, int masp, String TenSP, int soluong, float Dongia, float giamgia,  float thanhTien, String ghiChu) {  ChiTietDonHang ctdh = new ChiTietDonHang();  ctdh.setMaDH(MaHD);  ctdh.setMaSP(masp);  ctdh.setTenSP(TenSP);  ctdh.setSoLuong(soluong);  ctdh.setDonGia(Dongia);  ctdh.setGiamGia(giamgia);  ctdh.setThanhTien(thanhTien);  ctdh.setMoTa(ghiChu);  hdctDAO.insert(ctdh);  }  // Tạo hàm xuất hóa đơn  public void XuatHoaDon(String MADH) {  try {  Hashtable map = new Hashtable();  JasperReport report = JasperCompileManager.compileReport("src/frame/HoaDon.jrxml");  map.put("MADH", MADH);  JasperPrint p = JasperFillManager.fillReport(report, map, DriverManager.getConnection("jdbc:sqlserver://localhost:1433;databaseName = Project\_1\_test1;user = SA;password = 123456;"));  JasperViewer.viewReport(p, false);  JasperExportManager.exportReportToPdfFile(p, "HoaDon.pdf");  } catch (Exception ex) {  JOptionPane.showMessageDialog(this, "Xuất hóa đơn không thành công" + ex);  }  }  // Làm mới hóa đơn  private void New() {  tableHD.setRowCount(0);  tbHoaDon.setModel(tableHD);  lbTongTien.setText("0");  lbTienThoi.setText("0");  tfGhiChu.setText("");  lbNgayTao.setText(LocalDateTime.now().toString());  tfTenKH.setText("");  tfDiem.setText("0");  tfGiamGia.setText("0%");  tfKhachDua.setText("0");  }  private void btnXuatHoaDonActionPerformed(java.awt.event.ActionEvent evt) {  ThanhToan();  New();  lbMaDH.setText(SoHD());  }  private void btLamMoiHDActionPerformed(java.awt.event.ActionEvent evt) {  New();  }  private void lbMaNVAncestorAdded(javax.swing.event.AncestorEvent evt) {  lbMaNV.setText(ShareHelper.USER.getMaNV());  }  // Đổ dữ liệu lên bảng sản phẩm  private void LoadDBSP() {  table.setRowCount(0);  try {  List<SanPham> list = spDAO.select();  showtotable(list);  } catch (Exception ex) {  System.out.println("Lỗi đổ dữ liệu lên bảng sản phẩm" + ex);  }  }  // Đọc dữ liệu về Show lên bảng sản phầm  private void showtotable(List<SanPham> sp) {  for (SanPham sanPham : sp) {  Object[] row = {sanPham.getMASP(), sanPham.getTENSP(),  sanPham.getSOLUONG(), sanPham.getDONGIA(),  sanPham.getGHICHU()};  table.addRow(row);  tbSanPham.setModel(table);  }  } |

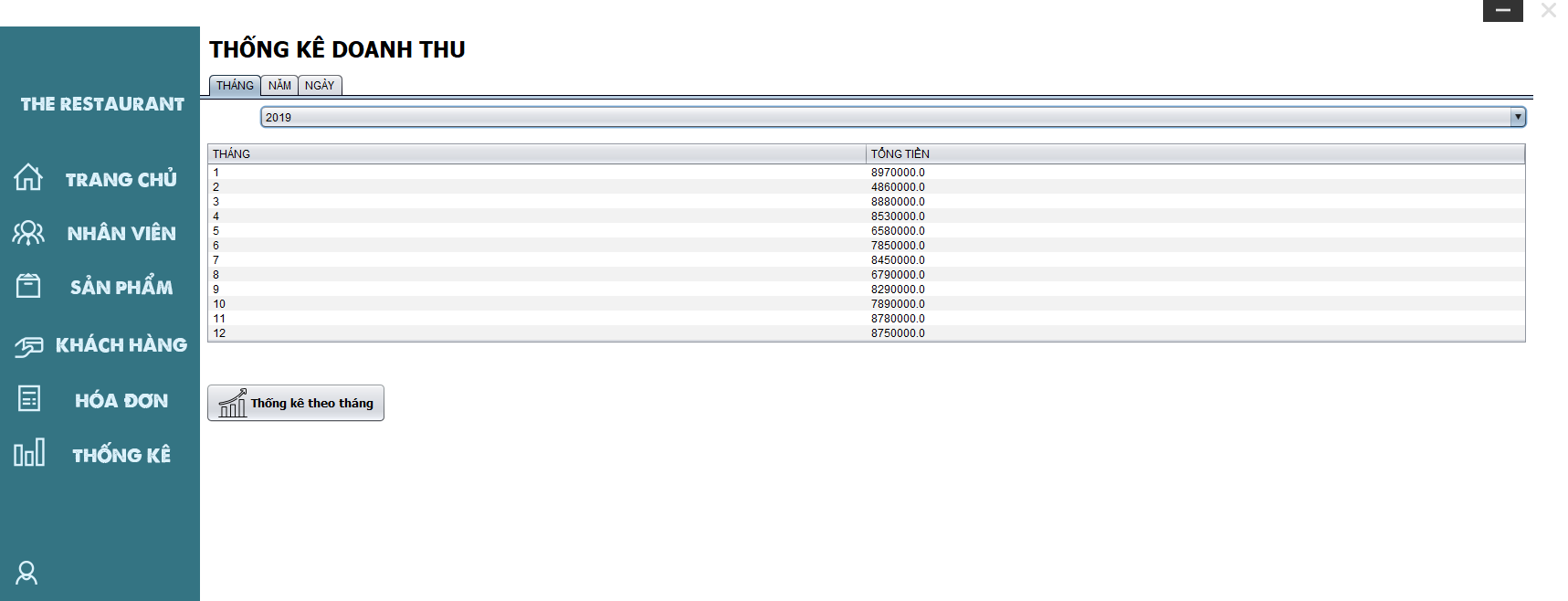
#### Thống kê



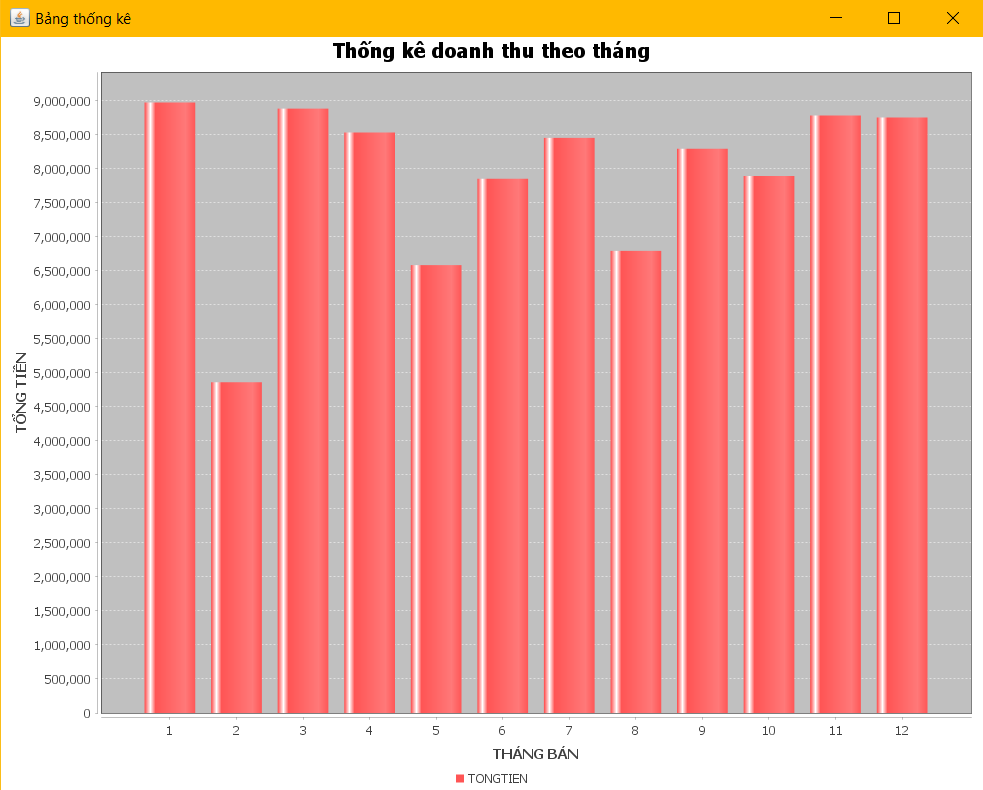
*Thống kê doanh thu theo ngày*



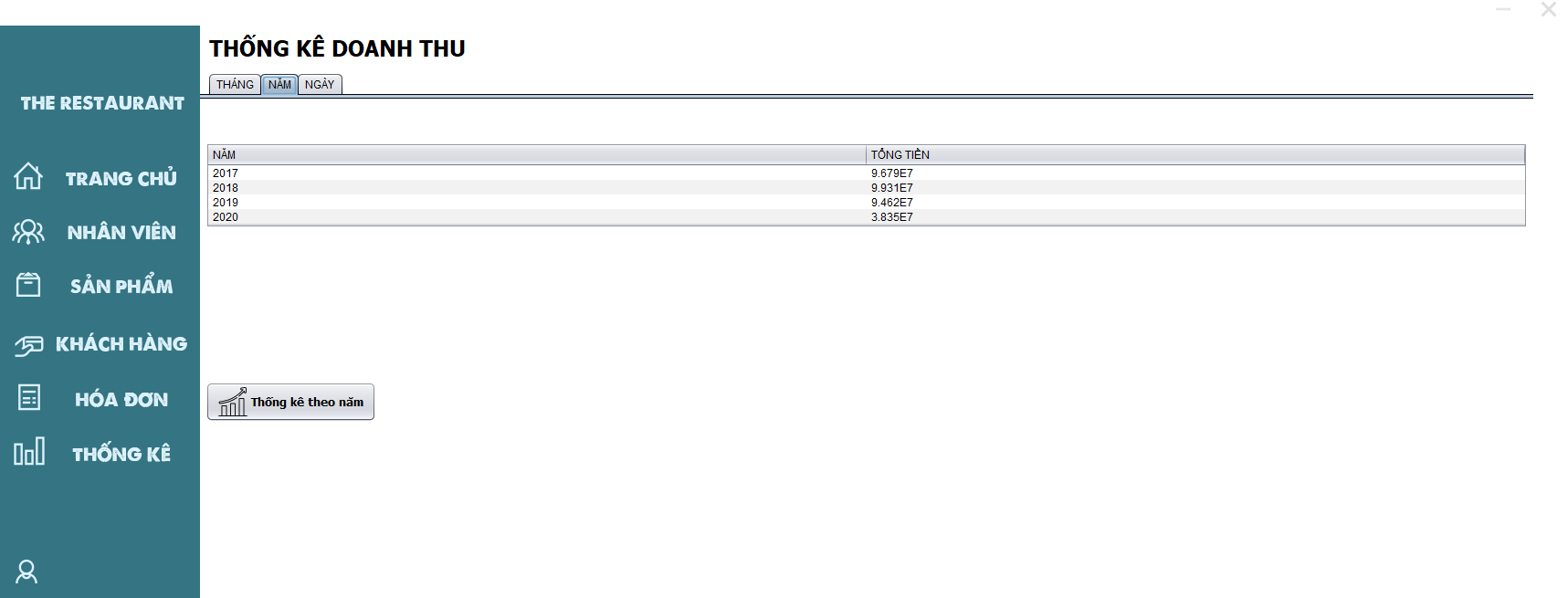
*Biểu đồ cột thể hiện doanh thu ngày*



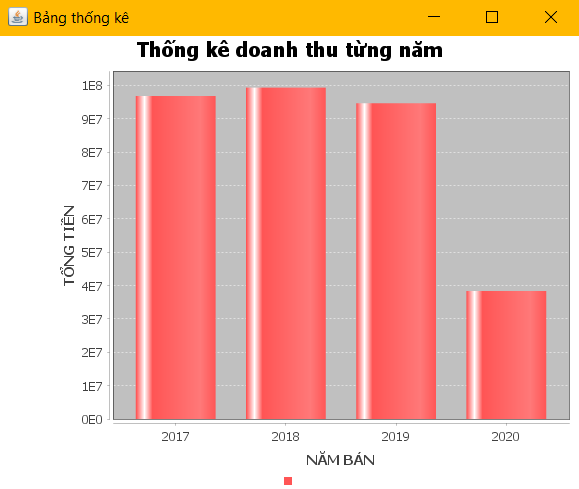
*Thống kê doanh thu theo tháng*



*Biểu đồ cột thể hiện doanh thu tháng*



*Thống kê doanh thu năm*



*Biểu đồ cột thể hiện doanh thu năm*

MÔ TẢ HOẠT ĐỘNG

|  |  |
| --- | --- |
| **Sự kiện** | **Công việc** |
| Constructor | This.RemoveHeader();  This.setTitle(“Thống kê);  fillComboBoxThang();  fillTableThongKeDoanhThuThang();  fillTableThongKeDoanhThuNam();  fillTableThongKeDoanhThuNgay(); |

HƯỚNG DẪN CÀI ĐẶT MÃ NGUỒN

|  |
| --- |
| public ThongKeJInternalFrame() {  this.RemoveHeader();  initComponents();  this.setTitle("Thống kê");  fillComboBoxThang();  fillTableThongKeDoanhThuThang();  fillTableThongKeDoanhThuNam();  fillTableThongKeDoanhThuNgay();  }  ThongKeDAO dao = new ThongKeDAO();  DonHangDAO dhdao = new DonHangDAO();  void fillComboBoxThang() {  DefaultComboBoxModel model = (DefaultComboBoxModel) cboThang.getModel();  model.removeAllElements();  List<DonHang> list = dhdao.select();  for (DonHang dh : list) {  int thang = dh.getNgayBan().getYear();  if (model.getIndexOf(thang) < 0) {  model.addElement(thang);  }  }  cboThang.setSelectedIndex(0);  }  void fillTableThongKeDoanhThuThang() {  DefaultTableModel model = (DefaultTableModel) tblTKDTThang.getModel();  model.setRowCount(0);  if (cboThang.getSelectedItem() == null) {  } else {  int thang = Integer.parseInt(cboThang.getSelectedItem().toString());  List<Object[]> list = dao.getThongKeDoanhThang(thang);  for (Object[] row : list) {  model.addRow(row);  }  }  }  void fillTableThongKeDoanhThuNam() {  DefaultTableModel model = (DefaultTableModel) tblTKDTNam.getModel();  model.setRowCount(0);  List<Object[]> list = dao.getThongKeDoanhNam();  for (Object[] row : list) {  model.addRow(row);  }  }  void fillTableThongKeDoanhThuNgay() {  DefaultTableModel model = (DefaultTableModel) tblTKDTNgay.getModel();  model.setRowCount(0);  List<Object[]> list = dao.getThongKeDoanhThuNgay();  for (Object[] row : list) {  model.addRow(row);  }  }  private void RemoveHeader() {  this.setBorder(javax.swing.BorderFactory.createEmptyBorder(0, 0, 0, 0));  BasicInternalFrameUI bi = (BasicInternalFrameUI) this.getUI();  bi.setNorthPane(null);  }  private void btnTKNamActionPerformed(java.awt.event.ActionEvent evt) {  try {  String query = "SELECT YEAR(NGAYBAN), SUM(TONGTIEN) \n"  + "FROM DONHANG\n"  + "GROUP BY YEAR(NGAYBAN)";  JDBCCategoryDataset dataset = new JDBCCategoryDataset(SQLConnect.ConnectDb(), query);  JFreeChart chart = ChartFactory.createBarChart("Thống kê doanh thu từng năm", "NĂM BÁN", "TỔNG TIỀN", dataset, PlotOrientation.VERTICAL, true, true, true);  BarRenderer renderer = null;  CategoryPlot plot = null;  ChartFrame frame = new ChartFrame("Bảng thống kê", chart);  frame.setSize(600, 500);  frame.setLocationRelativeTo(null);  frame.setVisible(true);  } catch (Exception e) {  }  }  private void cboThangActionPerformed(java.awt.event.ActionEvent evt) {  fillTableThongKeDoanhThuThang();  }  private void btnTKThangActionPerformed(java.awt.event.ActionEvent evt) {  try {  int thang = Integer.parseInt(cboThang.getSelectedItem().toString());  String query = "SELECT MONTH(NGAYBAN) THANG, SUM(TONGTIEN) TONGTIEN\n"  + "FROM DONHANG\n"  + "WHERE YEAR(NGAYBAN) = " + thang + "\n"  + "GROUP BY MONTH(NGAYBAN)";  JDBCCategoryDataset dataset = new JDBCCategoryDataset(SQLConnect.ConnectDb(), query);  JFreeChart chart = ChartFactory.createBarChart("Thống kê doanh thu theo tháng", "THÁNG BÁN", "TỔNG TIỀN", dataset, PlotOrientation.VERTICAL, true, true, true);  BarRenderer renderer = null;  CategoryPlot plot = null;  ChartFrame frame = new ChartFrame("Bảng thống kê", chart);  frame.setSize(1000, 800);  frame.setLocationRelativeTo(null);  frame.setVisible(true);  } catch (Exception e) {  }  }  private void btnTKThang1ActionPerformed(java.awt.event.ActionEvent evt) {  try {  SimpleDateFormat dateFrom1 = new SimpleDateFormat("yyyy-MM-dd");  String theDateFrom = dateFrom1.getDateInstance().format(jdcFrom.getDate());  SimpleDateFormat dateTo1 = new SimpleDateFormat("yyyy-MM-dd");  String theDateTo = dateTo1.getDateInstance().format(jdcTo.getDate());  String query = "SELECT NGAYBAN, SUM(TONGTIEN) AS TONGTIEN "  + "FROM DONHANG "  + "WHERE NGAYBAN BETWEEN '" + theDateFrom + "' AND '" + theDateTo + "' "  + "GROUP BY DONHANG.NGAYBAN";  JDBCXYDataset dataset = new JDBCXYDataset(DriverManager.getConnection("jdbc:sqlserver://localhost;DatabaseName=Project\_1\_test1", "sa", "123456"), query);  dataset.executeQuery(query);  SQLConnect.ConnectDb().close();  JFreeChart chart = ChartFactory.createTimeSeriesChart("Thống kê doanh thu theo ngày", "NGÀY BÁN", "TỔNG TIỀN", (XYDataset) dataset);  BarRenderer renderer = new BarRenderer();  ChartFrame frame = new ChartFrame("Bảng thống kê", chart);  frame.setSize(600, 500);  frame.setLocationRelativeTo(null);  frame.setVisible(true);  } catch (Exception e) {  e.printStackTrace();  }  } |

# CHƯƠNG V KIỂM THỬ

## Kiểm thử form nhân viên

|  |  |  |
| --- | --- | --- |
| **Thuộc tính** | **Thao tác** | |
| **Thêm mới** | **Cập nhật** |
| Mã nhân viên | Không trống, không trùng |  |
| Mật khẩu | Ít nhất 3 ký tự | |
| Họ tên | Không để trống | |
| Số điện thoại | Không để trống, đúng 10 ký tự và phải bắt đầu bằng các đầu số: 03, 04, 05, 07, 08, 09 | |
| Email | Không để trống, phải đúng định dạng “tenhopthu@tenmien” | |

## Kiểm thử form sản phẩm

|  |  |  |
| --- | --- | --- |
| **Thuộc tính** | **Thao tác** | |
| **Thêm mới** | **Cập nhật** |
| Tên sản phẩm | Không trống, không trùng | |
| Số lượng | Lớn hơn 0 | |
| Đơn giá | Lớn hơn 0 | |

## Kiểm thử form khách hàng

|  |  |  |
| --- | --- | --- |
| **Thuộc tính** | **Thao tác** | |
| **Thêm mới** | **Cập nhật** |
| Họ tên | Không để trống | |
| Số điện thoại | Không để trống, đúng 10 ký tự và phải bắt đầu bằng các đầu số: 03, 04, 05, 07, 08, 09 | |
| Email | Không để trống, phải đúng định dạng “tenhopthu@tenmien” | |

# CHƯƠNG VI – TỔNG KẾT VÀ HƯỚNG PHÁT TRIỂN

## Tổng kết

Mặc dù đã gặp không ít khó khăn nhưng với sự giúp đỡ của các thầy cô giáo, đặc biệt là thầy Nguyễn Đỗ Anh Khoa, nhóm em đã tìm hiểu thực tiễn hoạt động, giải quyết các vấn đề nghiệp vụ của phần mềm quản lý quán ăn và đã hoàn thành đồ án của mình theo đúng thời gian yêu cầu đề ra. Đồng thời đạt được những mục tiêu đã đặt ra:

**\* Kết quả đạt được:**

* Phân tích và thiết kế Database.
* Nắm được một ngôn ngữ lập trình.
* Xây dựng được cơ sở dữ liệu và giao diện cho chương trình bằng hệ quản trị cơ sở dữ liệu Microsoft SQL Server và Java.
* Quản lý được số lượng đơn hàng.
* Thống kê doanh thu theo khoảng thời gian được chọn một cách nhanh chóng và chính xác.
* Ứng dụng biểu đồ vào thống kê: để giao diện dễ nhìn hơn so với các số liệu.

**\* Hạn chế:**

* Thiếu kinh nghiệm trong việc phát triển dự án nên gặp nhiều vấn đề về cách tổ chức dự án và thời gian thực hiện.

## Hướng phát triển

* Xây dựng và hoàn thiện chức năng giám sát cửa hàng thông qua camera.
* Bảo mật cơ sở dữ liệu trên SQL Server: thay vì sử dụng cách thức phân quyền bằng bảng Login như trong đồ án này, chúng ta có thể chuyển sang phân quyền bằng cách: tạo các user trực tiếp trên cơ sở dữ liệu SQL và phân quyền cho các user này qua các Role của SQL. Khi đó, ta có thể quy định cụ thể người đăng nhập vào phần mềm với tên đăng nhập nào thì sẽ được giao quyền truy xuất những bảng nào, thậm chí là những trường dữ liệu nào trên database.