

TRƯỜNG ĐẠI HỌC BÀ RỊA – VŨNG TÀU
VIỆN CNTT - ĐIỆN - ĐIỆN TỬ

--- 📖 ---



BARIA VUNGTAU
UNIVERSITY
CAP SAINT JACQUES

BÁO CÁO
MÔN HỌC LẬP TRÌNH .NET
ĐỀ TÀI: QUẢN LÝ NHÀ HÀNG

Giảng viên hướng dẫn: Th.S Lê Thị Vĩnh Thanh
Nhóm sinh viên thực hiện : Nguyễn Xuân Bắc
Lê Ngọc Châu

BÀ RỊA - VŨNG TÀU, NĂM 2018

LỜI NÓI ĐẦU

Ngày nay công nghệ thông tin đã có những bước phát triển mạnh mẽ theo cả chiều sâu và chiều rộng. Máy tính điện tử không còn là một thứ phương tiện quý hiếm mà đang ngày càng trở thành một công cụ làm việc và giải trí thông dụng của con người không chỉ ở công sở mà ngay cả trong gia đình.

Đứng trước vai trò của thông tin hoạt động cạnh tranh gay gắt, các tổ chức và các doanh nghiệp đều tìm mọi biện pháp để xây dựng và hoàn thiện hệ thống thông tin của mình nhằm tin học hóa các hoạt động tác vụ của đơn vị.

Hiện nay các công ty tin học hàng đầu thế giới không ngừng đầu tư và cải thiện các giải pháp cũng như các sản phẩm nhằm giúp cho các công ty, doanh nghiệp thực hiện nhanh, chính xác các công việc bằng chương trình. Thông qua các sản phẩm và công nghệ này, chúng ta dễ nhận ra tầm quan trọng và tính tất yếu của chương trình hỗ trợ cho công việc. Với những thao tác đơn giản chương trình sẽ giúp bạn tất cả những gì mà mình cần mà không phải mất nhiều thời gian.

Để tiếp cận và đóng góp đầy mạnh sự phổ biến của thương mại điện tử ở Việt Nam, chúng em đã tìm hiểu và xây dựng một chương trình “Quản lý nhà hàng”.

Với sự hướng dẫn tận tình của cô Lê Thị Vĩnh Thanh đã giúp chúng em hoàn thành cuốn báo cáo môn học Lập trình .Net này. Tuy đã cố gắng hết sức tìm hiểu, phân tích thiết kế và xây dựng hệ thống nhưng chắc chắn không tránh khỏi được những thiếu sót. Chúng em rất mong được sự thông cảm và góp ý của các quý thầy cô. Chúng em xin chân thành cảm ơn.

Xin chân thành cảm ơn Ban giám hiệu cùng toàn thể quý thầy cô, đặc biệt là quý thầy cô Viện CNTT - Điện - Điện Tử Trường Đại Học Bà Rịa

Vũng Tàu, những người đã truyền đạt cho chúng em nhiều kiến thức quý báu trong học tập.

Xin trân trọng cảm ơn!

Tp. Vũng Tàu, ngày 27 tháng 11 năm 2018

Nhóm sinh viên thực hiện

Nguyễn Xuân Bắc

Lê Ngọc Châu

TÓM TẮT NỘI DUNG

Với mức độ phức tạp và quy mô ứng dụng, cộng thêm vấn đề thời gian cho nên đề tài “Quản lý nhà hàng – C#” của chúng em chỉ dừng lại ở mức độ tìm hiểu ngôn ngữ lập trình C# và áp dụng xây dựng thực nghiệm chương trình quản lý nhà hàng.

Chương trình với mục đích giải quyết các vấn đề về quản lý, bán hàng,... cho những người quản lý nhà hàng. Giao diện người dùng dễ sử dụng, hỗ trợ nhanh chóng nên rút ngắn được rất nhiều thời gian trong công việc.

Do còn một số hạn chế nên chương trình chỉ mới dừng lại ở những chức năng cơ bản như là quản lý bán hàng, quản lý thông tin nhân viên, thêm xóa sửa món ăn, sao lưu, khôi phục dữ liệu, thống kê chi tiết... Trong tương lai, chúng em sẽ phát triển theo hướng thương mại điện tử.

Chương trình sử dụng 2 framework để hỗ trợ giao diện:

- DevExpress: <https://www.devexpress.com/>
- Bunifu: <https://bunifuframework.com/>

MỤC LỤC

Mục lục

LỜI NÓI ĐẦU	1
TÓM TẮT NỘI DUNG	3
MỤC LỤC.....	4
CHƯƠNG I – GIỚI THIỆU ĐỀ TÀI.....	6
1. Giới thiệu đề tài.....	6
2. Phân tích.....	6
CHƯƠNG VI – CƠ SỞ DỮ LIỆU	8
1. Sơ đồ quan hệ	8
2. Chi tiết các lớp.....	8
CHƯƠNG II – PHÂN TÍCH VÀ THIẾT KẾ HỆ THỐNG	11
1. Xác định yêu cầu bài toán.....	11
1.1. Mô tả bài toán	11
1.2. Xác định và phân tích các giá trị nghiệp vụ	11
1.3. Xác định yêu cầu hệ thống.....	12
2. Phân tích hệ thống.....	13
2.1. Xác định các tác nhân hệ thống	13
2.2. Các biểu đồ	14
2.3. Đặc tả các ca sử dụng hệ thống.....	20
CHƯƠNG III – CÁC CHỨC NĂNG CỦA CHƯƠNG TRÌNH.....	27
I – Các chức năng chính.....	27
1. Quản lý bán hàng	27
2. Quản lý sản phẩm	28
3. Quản lý nhân viên	31
4. Quản lý thống kê.....	34
II – Các Form giải quyết chức năng.....	42
1. Form đăng nhập, đăng ký	42
2. Form thông tin người dùng.....	49
3. Form thay đổi mật khẩu.....	51

III – Các chức năng phụ.....	54
1. Sao lưu dữ liệu	54
2. Khôi phục dữ liệu	54
3. Chức năng khóa màn hình	55
4. Thay đổi giao diện chương trình.....	56
5. Mã hóa mật khẩu	57
KẾT LUẬN	58
TÀI LIỆU THAM KHẢO.....	59

CHƯƠNG I – GIỚI THIỆU ĐỀ TÀI

Đề tài: “*Xây dựng phần mềm Quản lý nhà hàng – C#*”

1. Giới thiệu đề tài

Với nhu cầu giải quyết vấn đề buôn bán cũng như để dễ dàng quản lý nhà hàng, từ đó xuất hiện các chương trình quản lý nhà hàng.

Đối tượng sử dụng chương trình có thể là các chủ nhà hàng, sử dụng để quản lý công việc của mình một cách dễ dàng hơn.

2. Phân tích

Đây là một phần mềm cho phép chủ nhà hàng thực hiện nghiệp vụ bán hàng, quản lý được thông tin của nhân viên, món ăn, bàn ăn và doanh thu của nhà hàng.

Các chức năng của chương trình:

- Đăng nhập, đăng ký tài khoản.
- Xem, chỉnh sửa thông tin tài khoản.
- Thay đổi mật khẩu.
- Sao lưu dữ liệu.
- Phục hồi dữ liệu.
- Khóa màn hình.
- Thay đổi giao diện ứng dụng.
- Quản lý bán hàng.
- Thanh toán và in hóa đơn.
- Thêm, xóa sửa các loại sản phẩm.
- Sửa thông tin, khóa, mở khóa tài khoản nhân viên.
- Xem thống kê chi tiết doanh thu và hóa đơn.

Ngoài ra, do hạn chế về mặt thời gian, một số chức năng đang trong giai đoạn phát triển:

- Nhập hàng.
- Kiểm tra tồn kho.
- Hóa đơn, phiếu nhập.
- License Key.

Hệ thống có 2 phần:

Thứ nhất: Dành cho nhân viên:

Nhân viên chỉ có quyền:

- Đăng nhập, đăng ký tài khoản.
- Xem, chỉnh sửa thông tin tài khoản.
- Thay đổi mật khẩu.
- Khóa màn hình.
- Thay đổi giao diện ứng dụng.
- Quản lý bán hàng.
- Thanh toán và in hóa đơn.
- Thêm, xóa sửa các loại sản phẩm.

Thứ hai: Dành cho chủ nhà hàng

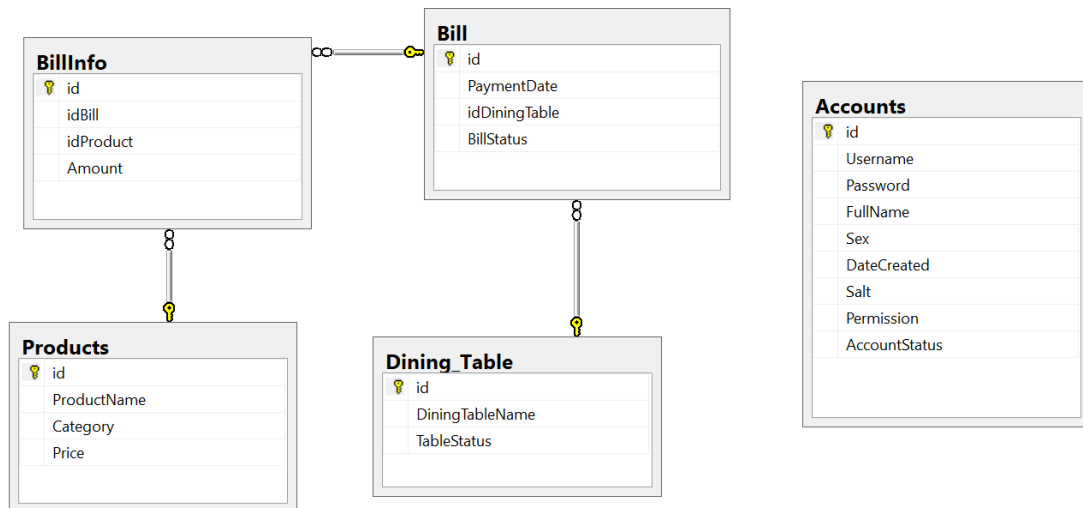
Chủ nhà hàng sẽ quản lý được tất cả quyền mà nhân viên có và được sử dụng thêm một số chức năng:

Các chức năng:

- Sao lưu dữ liệu.
- Phục hồi dữ liệu.
- Sửa thông tin, khóa, mở khóa tài khoản nhân viên.
- Xem thống kê chi tiết doanh thu và hóa đơn.

CHƯƠNG VI – CƠ SỞ DỮ LIỆU

1. Sơ đồ quan hệ



2. Chi tiết các lớp

a. dbo.Accounts

Column Name	Data Types	Constraints
id	INT	IDENTITY, PRIMARY KEY
Username	VARCHAR(255)	NOT NULL, UNIQUE
Password	VARCHAR (255)	NOT NULL
FullName	NVARCHAR (255)	NOT NULL
Sex	NVARCHAR(10)	NOT NULL
DateCreated	DATETIME	DEFAULT CURRENT_TIMESTAMP
Salt	VARCHAR(10)	NOT NULL
Permission	NVARCHAR(10)	NOT NULL

AccountStatus	NVARCHAR(20)	NOT NULL
---------------	--------------	----------

b. dbo.Dining_Table

Column Name	Data Types	Constraints
id	INT	IDENTITY, PRIMARY KEY
DiningTableName	NVARCHAR(10)	NOT NULL, UNIQUE
TableStatus	NVARCHAR (10)	NOT NULL

c. dbo.Bill

Column Name	Data Types	Constraints
id	INT	IDENTITY, PRIMARY KEY
PaymentDate	DATETIME	DEFAULT '1753-01-01'
idDiningTable	INT	NOT NULL, FOREIGN KEY
BillStatus	NVARCHAR(20)	DEFAULT N'Chưa thanh toán'

d. dbo.BillInfo

Column Name	Data Types	Constraints
id	INT	IDENTITY, PRIMARY KEY
idBill	NVARCHAR(255)	NOT NULL, UNIQUE, FOREIGN KEY
idProduct	NVARCHAR (255)	NOT NULL, FOREIGN KEY

Amount	INT	DEFAULT 0
--------	-----	-----------

e. dbo.Products

Column Name	Data Types	Constraints
id	INT	IDENTITY, PRIMARY KEY
ProductName	NVARCHAR(255)	NOT NULL, UNIQUE
Category	NVARCHAR (255)	NOT NULL
Price	FLOAT	DEFAULT 0

* Tất cả các bảng đều có một khóa chính là id kiểu INT và tự động tăng.

Username, DiningTableName, ProductName

CHƯƠNG II – PHÂN TÍCH VÀ THIẾT KẾ HỆ THỐNG

1. Xác định yêu cầu bài toán

1.1. Mô tả bài toán

a. Đặt bàn

- Nhà hàng cho phép khách hàng sử dụng một hệ thống quản lý nhà hàng để đặt bàn ăn.
- Khách hàng có thể gọi điện đặt bàn, liên hệ đặt bàn trực tiếp tại quầy lễ tân hoặc thông qua Website.
- Khi đặt bàn khách hàng phải trả tiền đặt cọc trước, nhân viên lễ tân tiếp nhận thông tin cá nhân, yêu cầu khách hàng (vị trí bàn, đồ ăn, ngày ăn) và lưu thông tin vào hệ thống.

b. Thanh toán

- Khách hàng yêu cầu thanh toán với nhân viên và nhân viên báo cáo lại với lễ tân.
- Nhân viên in hóa đơn và cập nhật vào hệ thống.

1.2. Xác định và phân tích các giá trị nghiệp vụ

a. Mang lại giá trị nghiệp vụ

- Tăng khả năng xử lý: thông tin được xử lý một cách tự động, có thể xử lý đồng thời và cho kết quả nhanh chóng, chính xác.
- Thu thập được thông tin về các khách hàng một cách tự động, không phải mất công nhập lại thông tin.
- Đáp ứng yêu cầu nghiệp vụ một cách tin cậy, chính xác, an toàn, bí mật.

b. Mang lại giá trị sử dụng

- Khách hàng có thể nhanh chóng tìm ra các thông tin về đồ ăn, giá tiền, địa điểm,... mà mình muốn đặt.
- Chủ nhà hàng có thể dựa vào hệ thống để kiểm tra doanh thu của nhà hàng, quản lý hàng hóa của nhà hàng.
- Rút ngắn thời gian, thay vì phải đến trực tiếp thì khách hàng có thể sử dụng hệ thống để tìm hiểu thông tin cần thiết hoặc quản lý có thể kiểm soát được thông tin về nhân viên của mình.

c. Mang lại giá trị kinh tế

- Giảm chi phí hoạt động: nhờ có hệ thống quản lý nhà hàng, thông tin được xử lý tự động, không phải mất nhiều giấy tờ nên số lượng người tham gia hoạt động thanh toán giảm, từ đó giảm chi phí hoạt động.
- Khách hàng có thể tiết kiệm thời gian và công sức khi sử dụng hệ thống quản lý nhà hàng.

1.3. Xác định yêu cầu hệ thống

- Hệ thống phải cung cấp cho nhân viên một danh sách các món ăn cùng với các thông tin liên quan(tên món , giá món ăn ,...) để dễ dàng với việc tìm kiếm và đặt món cho khách hàng.
- Hệ thống phải cho phép nhân viên đăng ký tài khoản và đăng nhập hệ thống dễ dàng.
- Sau khi đăng ký xong hệ thống xác nhận thông tin và xử lý thông tin phản hồi tới người dùng về việc đăng ký thành công hay không. Thông tin phản hồi nhanh chóng chính xác.
- Sau khi khách hàng đăng nhập, hệ thống phải hiển thị thông tin của người dùng.
- Khi nhân viên đặt bàn và món ăn, danh sách các bàn và món ăn đã được đặt sẽ lưu vào hệ thống.

2. Phân tích hệ thống

2.1. Xác định các tác nhân hệ thống

Dựa vào văn bản mô tả bài toán, ta xác định được các tác nhân của hệ thống như sau:

- + Tác nhân Nhân viên: sử dụng hệ thống để xem thông tin của mình, quản lý món ăn, bàn ăn... và đặt bàn, thanh toán cho khách hàng.
- + Tác nhân nhà quản lý: sử dụng hệ thống để quản lý thông tin khách hàng, quản lý thông tin món ăn, quản lý thông tin nhân viên...

Xác định các ca sử dụng

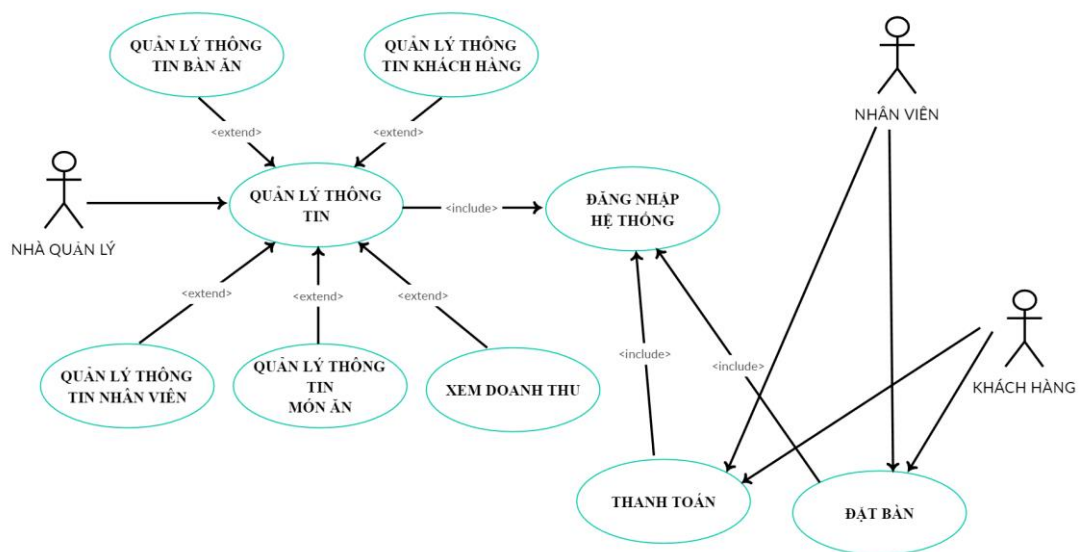
Dựa trên văn bản mô tả bài toán và việc phân tích để tìm ra các tác nhân, ta xác định được các ca sử dụng như sau:

- Đăng nhập, đăng ký hệ thống.
- Quản lý, lưu trữ thông tin khách hàng đặt bàn.
- Tra cứu thông tin món, bàn ăn.
- Quản lý thông tin nhân viên.
- Cập nhật, quản lý thông tin món ăn.
- Thanh toán hóa đơn.
- Quản lý về doanh thu.

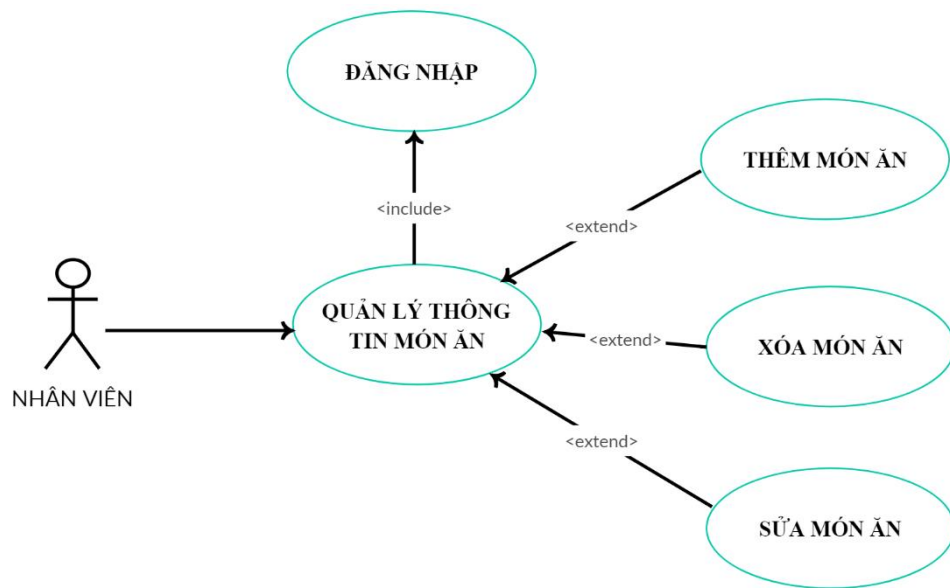
Các tác nhân	Các ca sử dụng
Nhân viên	Đăng nhập, đăng ký hệ thống, Quản lý thông tin món ăn, bàn ăn mà khách hàng đặt.
Nhà quản lý	Quản lý thông tin nhân viên, món ăn, doanh thu.

2.2. Các biểu đồ

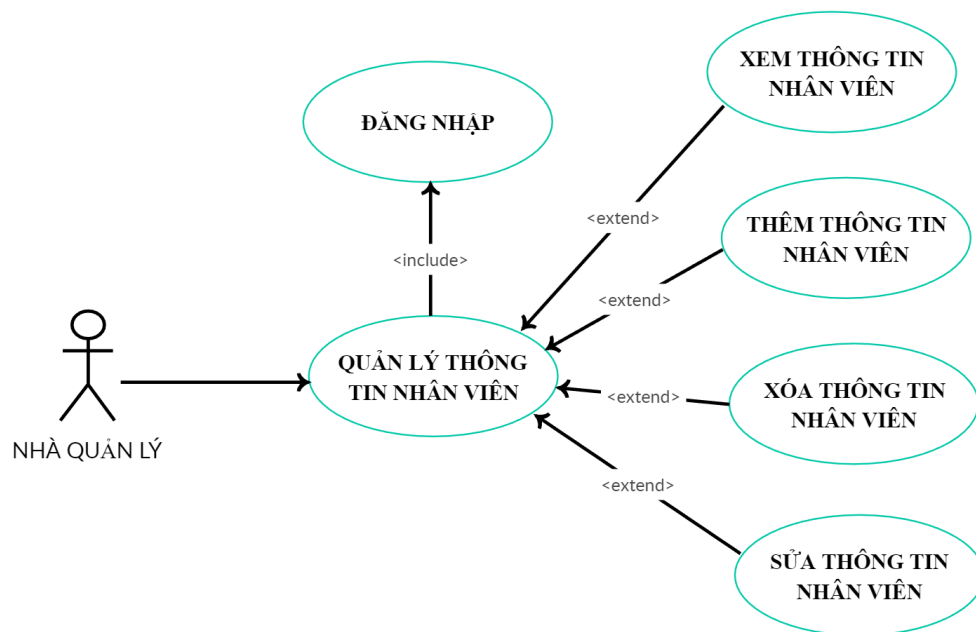
2.2.1. Biểu đồ ca sử dụng



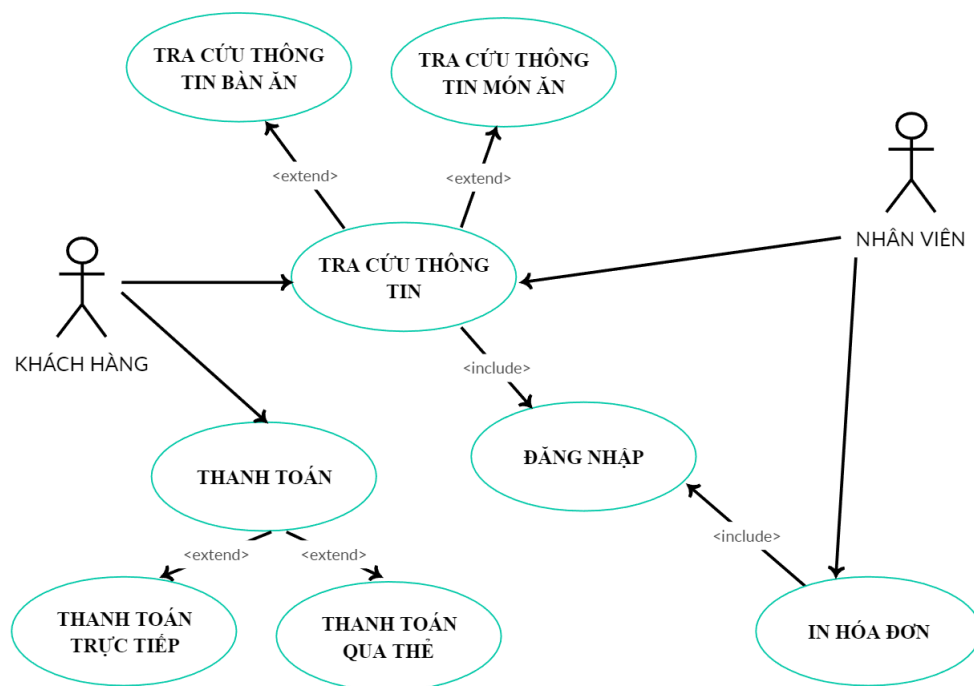
Biểu đồ ca sử dụng chính



Biểu đồ ca sử dụng chức năng quản lý món ăn

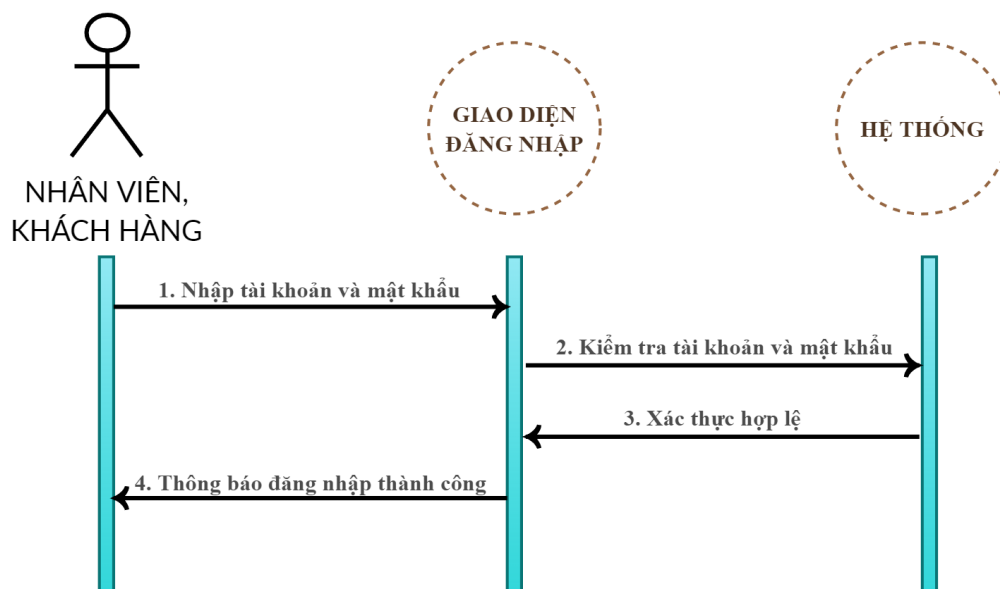


Biểu đồ ca sử dụng chức năng quản lý thông tin nhân viên

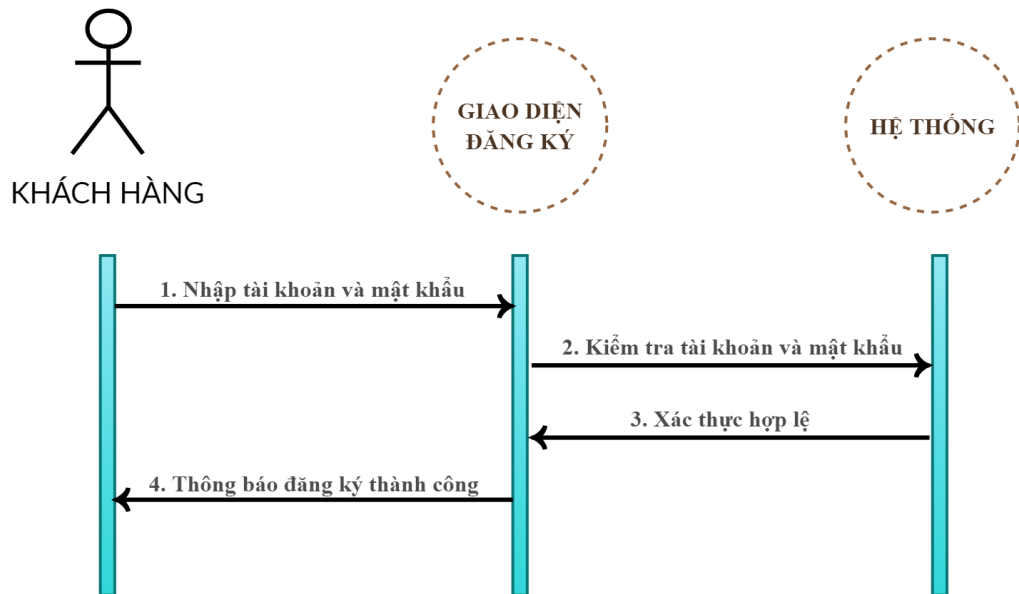


Biểu đồ ca sử dụng chức năng thanh toán

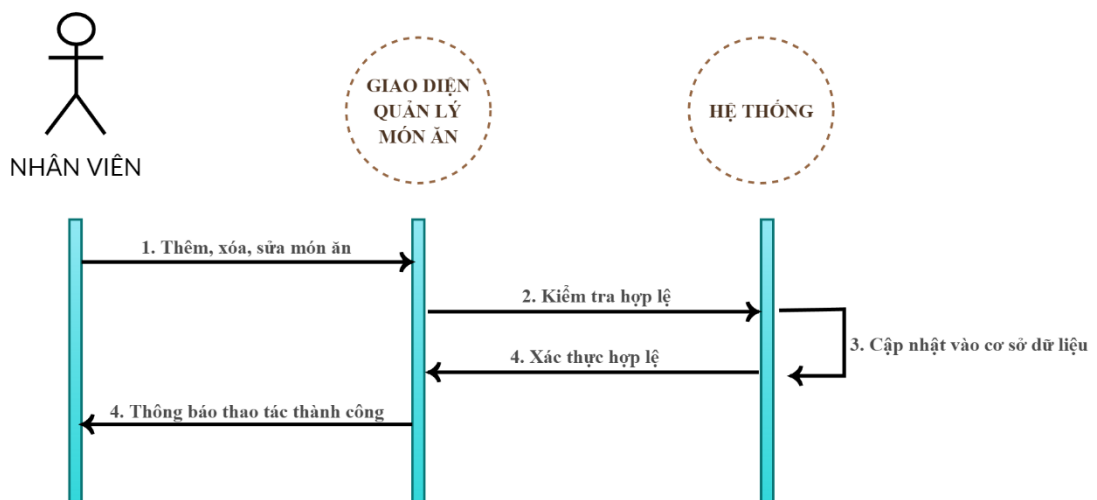
2.2.2. Biểu đồ tuần tự



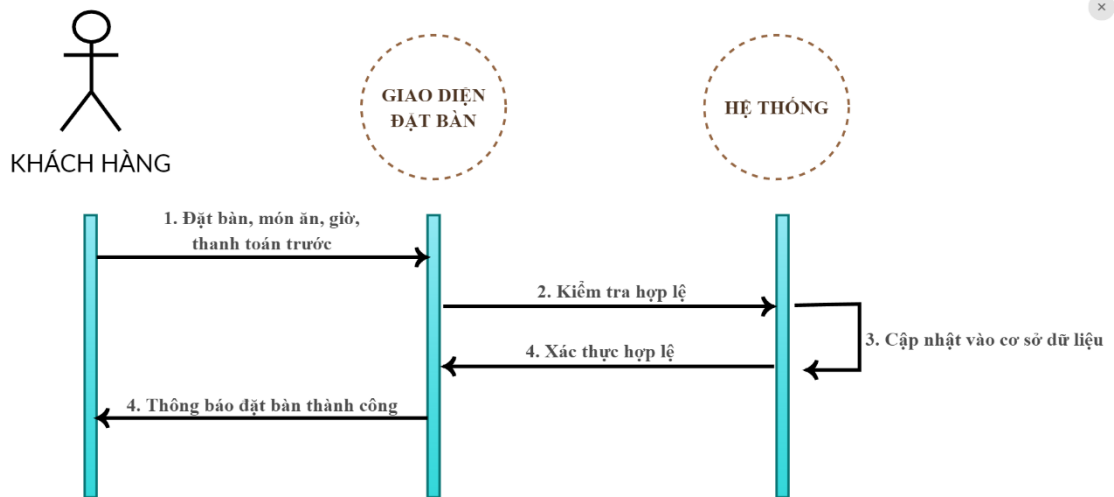
Biểu đồ tuần tự cho chức năng đăng nhập



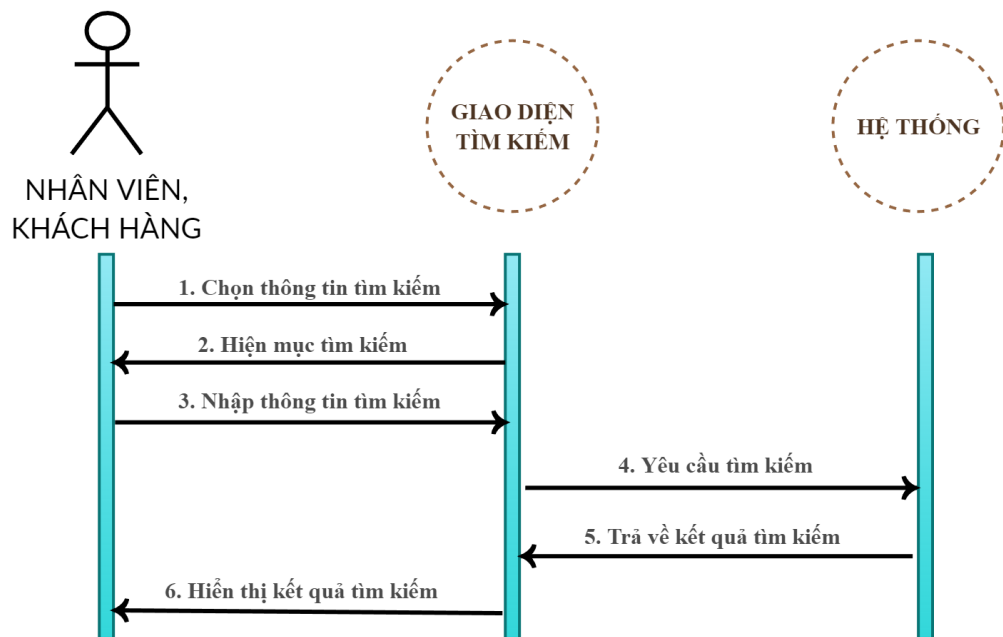
Biểu đồ tuần tự cho chức năng đăng ký



Biểu đồ tuần tự cho chức năng quản lý món ăn

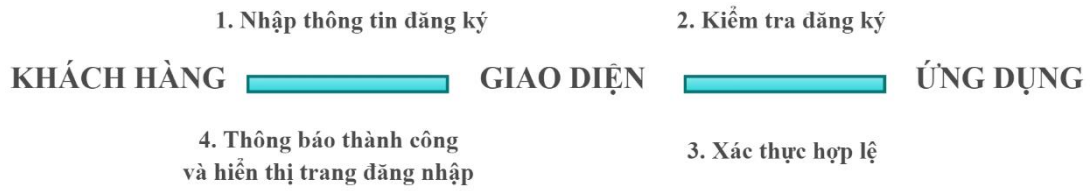


Biểu đồ tuần tự cho chức năng đặt bàn

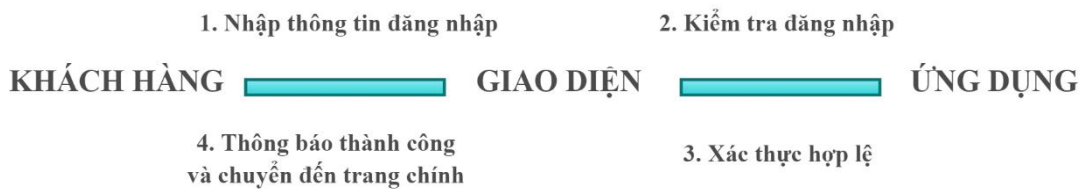


Biểu đồ tuần tự cho chức năng tìm kiếm

2.2.3. Biểu đồ hợp tác



Biểu đồ hợp tác cho chức năng đăng ký



Biểu đồ hợp tác cho chức năng đăng nhập



Biểu đồ hợp tác cho chức năng đặt bàn



Biểu đồ hợp tác cho chức năng quản lý món ăn

2.3. Đặc tả các ca sử dụng hệ thống

2.3.1. Ca sử dụng Đăng nhập hệ thống

Mô tả tóm tắt

- Tên ca sử dụng: Đăng nhập hệ thống
- Mục đích: Mô tả cách một người sử dụng đăng nhập vào hệ thống.
- Tác nhân: Khách hàng, nhân viên, nhà quản lý.

Các luồng sự kiện

Luồng sự kiện chính

- Ca sử dụng này bắt đầu khi tác nhân muốn đăng nhập vào hệ thống.
- Hệ thống yêu cầu tác nhân nhập tài khoản và mật khẩu đăng nhập.
- Tác nhân nhập tên đăng nhập và mật khẩu đăng nhập của mình.
- Hệ thống xác nhận tài khoản và mật khẩu đăng nhập có hợp lệ không, nếu không hợp lệ thì thực hiện luồng A1.
- Hệ thống lưu vào cơ sở dữ liệu thông tin đăng nhập.

Các luồng rẽ nhánh

- Luồng A1: Nhập sai tài khoản/mật khẩu đăng nhập

- Hệ thống hiển thị một thông báo lỗi.

- Người sử dụng có thể chọn hoặc là đăng nhập lại hoặc là huỷ bỏ đăng nhập, khi đó ca sử dụng kết thúc.

Điều kiện bắt đầu: Không.

Điều kiện kết thúc: Nếu việc đăng nhập thành công, người sử dụng sẽ đăng nhập được vào hệ thống.

2.3.2. Ca sử dụng Đăng nhập hệ thống

Mô tả tóm tắt

- Tên ca sử dụng: Đăng ký hệ thống
- Mục đích: Mô tả cách một người sử dụng đăng ký tài khoản trên hệ thống.
- Tác nhân: Khách hàng, nhân viên, nhà quản lý.

Các luồng sự kiện

Luồng sự kiện chính

- Ca sử dụng này bắt đầu khi tác nhân muốn đăng ký tài khoản trên hệ thống.
- Hệ thống yêu cầu tác nhân nhập họ tên, tên tài khoản, email và mật khẩu đăng nhập.
- Tác nhân nhập họ tên, tên tài khoản, email và mật khẩu đăng nhập của mình.
- Hệ thống xác nhận tên tài khoản, email có trùng với tài khoản đã có sẵn trong hệ thống hay không, nếu không hợp lệ thì thực hiện luồng A1.
- Hệ thống xác nhận mật khẩu nhập vào hay không, nếu không hợp lệ thì thực hiện luồng A2.

- Hệ thống lưu vào cơ sở dữ liệu thông tin đăng ký.

Các luồng rẽ nhánh

- Luồng A1: Nhập tên tài khoản/email đã đăng ký trên hệ thống
- Hệ thống hiển thị một thông báo lỗi.
 - Người sử dụng có thể chọn tên tài khoản khác, email khác hoặc là huỷ bỏ đăng ký, khi đó ca sử dụng kết thúc.
- Luồng A1: Nhập mật khẩu không hợp lệ (để trống hoặc ký tự đặc biệt...)
 - Người sử dụng phải nhập mật khẩu và mật khẩu không có ký tự đặc biệt hoặc là huỷ bỏ đăng ký, khi đó ca sử dụng kết thúc.

Điều kiện bắt đầu: Không.

Điều kiện kết thúc: Nếu việc đăng ký thành công, người sử dụng sẽ nhận được thông báo đăng ký thành công và được chuyển đến trang đăng nhập.

2.3.2. Ca sử dụng Đặt bàn, món ăn

Mô tả tóm tắt

- Tên ca sử dụng: Đặt bàn, món ăn
- Mục đích:
 - + Giúp cho khách hàng có thể đặt món ăn và bàn ăn mà mình muốn.
 - + Khách hàng chọn các bàn, món ăn có thể đặt hoặc có thể huỷ bỏ, thay đổi các món.
- Tác nhân: Khách hàng

Các luồng sự kiện

Luồng sự kiện chính

- Ca sử dụng này bắt đầu khi khách hàng muốn đặt bàn, món ăn hoặc thay đổi bàn, món ăn đã đặt.
 - Hệ thống cho phép khách hàng tùy chọn bàn ăn, món ăn.
 - Hệ thống yêu cầu nhập thông tin họ tên, ngày giờ đặt khi khách hàng đặt bàn ăn, món ăn.
 - Hệ thống hiển thị các lựa chọn: Thêm, Xóa, Xem, Thoát.
 - Hệ thống không cho phép khách hàng đặt bàn mà bàn đó đã có người đang sử dụng hoặc đang đặt.
- Nếu Khách hàng lựa chọn “Thêm món ăn” thì luồng sự kiện con A1 sẽ được thực hiện.
- Nếu Khách hàng lựa chọn “Xóa một món đã đặt ” thì luồng sự kiện con A2 sẽ được thực hiện.
- Nếu Khách hàng chọn “Đặt bàn” thì luồng sự kiện con A3 sẽ được thực hiện.

Các luồng rẽ nhánh

Luồng A1: Khách hàng lựa chọn thêm món ăn

- Hệ thống hiển thị danh sách thức đơn bao gồm tên món ăn, số lượng và đơn giá để khách hàng lựa chọn.
- Hệ thống tự động tính tổng cộng số tiền khi khách hàng chọn món ăn.

Luồng A2: Khách hàng lựa chọn xóa món ăn đã đặt.

- Hệ thống hiển thị ra danh sách những món ăn mà khách hàng đã đặt để khách hàng có thể xóa món.

- Hệ thống tự động cập nhật lại tổng số tiền món ăn và hiển thị cho khách hàng.

Luồng A3: Khách hàng lựa chọn đặt bàn.

- Khách hàng thực hiện thanh toán trả trước tiền cọc đặt bàn.

- Hệ thống hiển thị thông báo thành công cho khách hàng nếu khách hàng nhập thông tin đầy đủ, hợp lệ.

- Hệ thống sẽ lưu vào cơ sở dữ liệu thông tin khách đặt bàn, món ăn, giờ ăn và tổng số tiền.

Điều kiện bắt đầu: Khách hàng phải đăng nhập vào hệ thống để đặt bàn, đặt món ăn.

Điều kiện kết thúc: Nếu ca sử dụng được thực hiện thành công thì danh sách các món mà khách hàng đặt sẽ được lưu vào cơ sở dữ liệu. Khách hàng thay đổi hoặc xóa những món đã đặt, hệ thống sẽ tự động cập nhật vào cơ sở dữ liệu.

2.3.3. Ca sử dụng quản lý nhân viên

- Tên ca sử dụng: Quản lý nhân viên

- Tác nhân: Nhân viên, nhà quản lý

- Mô tả: Chức năng liên quan đến quá trình quản lý thông tin nhân viên, quản lý số ngày nghỉ.

Các luồng sự kiện

Luồng sự kiện chính

- Hệ thống cho phép nhà quản lý thêm, xóa, sửa, xem thông tin nhân viên và quản lý số ngày nghỉ của nhân viên.

- Nếu Nhà quản lý lựa chọn “Thêm nhân viên” thì luồng sự kiện con A1 sẽ được thực hiện.
- Nếu Nhà quản lý lựa chọn “Xóa nhân viên” thì luồng sự kiện con A2 sẽ được thực hiện.
- Nếu Nhà quản lý lựa chọn “Sửa thông tin nhân viên” thì luồng sự kiện con A3 sẽ được thực hiện.

Các luồng rẽ nhánh

Luồng A1: Nhà quản lý lựa chọn thêm nhân viên

- Hệ thống hiển thị một giao diện để nhà quản lý có thể nhập thông tin của nhân viên bao gồm họ tên, số điện thoại và loại nhân viên.
- Hệ thống hiển thị thông báo thành công nếu thao tác thành công.

Luồng A2: Nhà quản lý lựa chọn xóa nhân viên.

- Hệ thống hiển thị ra danh sách những nhân viên của nhà hàng và nhà quản lý có thể xóa nhân viên.
- Hệ thống hiển thị thông báo thành công nếu thao tác thành công.

Luồng A3: Nhà quản lý lựa chọn sửa nhân viên.

- Hệ thống hiển thị thông tin nhân viên cũ của nhà hàng và nhà quản lý có thể chỉnh sửa thông tin.
- Hệ thống hiển thị thông báo thành công nếu thao tác thành công.

Điều kiện bắt đầu: nhà quản lý đăng nhập vào hệ thống .

Điều kiện kết thúc: thông tin nhân viên được cập nhật và lưu trữ vào hệ thống.

2.3.4. Ca sử dụng liên hệ

- Tên ca sử dụng: Liên hệ
- Tác nhân: Nhân viên, nhà quản lý, khách hàng
- Mô tả: Chức năng cho phép người dùng liên hệ với nhà hàng thông qua hệ thống website.

Các luồng sự kiện

Luồng sự kiện chính

- Khách hàng liên hệ trực tiếp với nhà hàng thông qua website.
- Hệ thống yêu cầu khách hàng nhập họ tên, email, và nội dung liên hệ.
- Khách hàng nhập họ tên, email, nội dung liên hệ. Nếu đầy đủ và hợp lệ thì luồng sự kiện A1 được thực hiện.

Luồng A1: Khách hàng gửi liên hệ.

- Sau khi khách hàng gửi liên hệ, hệ thống sẽ hiển thị thông báo đã gửi thành công đến khách hàng và thông tin liên hệ của khách hàng sẽ được gửi tới email của nhà hàng.
- Nhân viên hỗ trợ của nhà hàng sẽ có nhiệm vụ đọc và trả lời, tư vấn cho khách hàng về nội dung liên hệ của họ.

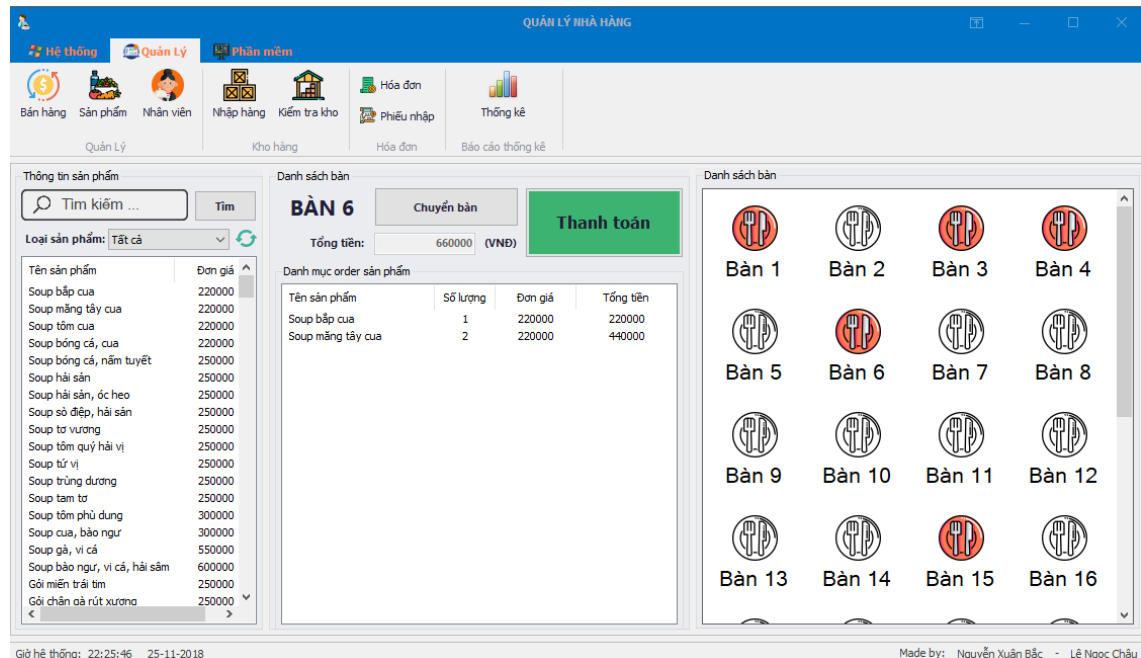
Điều kiện bắt đầu: Không.

Điều kiện kết thúc: thông tin liên hệ của khách hàng được phản hồi sớm nhất.

CHƯƠNG III – CÁC CHỨC NĂNG CỦA CHƯƠNG TRÌNH

I – Các chức năng chính

1. Quản lý bán hàng



Form bán hàng gồm các chức năng:

- Tìm kiếm, thêm món ăn vào bàn.
- Thêm bàn, xóa bàn.
- Hiện thị bàn trống, hiện thị bàn có khách.
- Chuyển bàn.
- Thanh toán biên lai.

* Khi click button thanh toán, một report hiện ra giúp người dùng có thể in hóa đơn.

2. Quản lý sản phẩm

Form quản lý sản phẩm gồm các chức năng:

- Hiện thị danh sách tên sản phẩm, các loại sản phẩm và đơn giá.
- Thêm, xóa sửa thông tin sản phẩm.

Code xử lý:

```
private void XuLyButton(bool b)
{
    btnProductAdd.Enabled = btnProductDelete.Enabled =
    btnProductEdit.Enabled = b;
    btnProductSave.Enabled = btnProductCancel.Enabled =
    groupBoxProductEdit.Enabled = !b;

    categoryTextEdit.Visible = false;
    btnProduct2.Visible = false;
    cbxCategory2.Visible = true;
    btnProduct1.Visible = true;
}

private void btnProductPrevious_Click(object sender, EventArgs e)
{
    this.productsBindingSource.MovePrevious();
}

private void btnProductNext_Click(object sender, EventArgs e)
{
    this.productsBindingSource.MoveNext();
}
```

```

private void btnProductAdd_Click(object sender, EventArgs e)
{
    XuLyButton(false);
    productsGridControl.Enabled = false;
    productNameTextEdit.Text = string.Empty;
    productNameTextEdit.Focus();
    cbxCategory2.SelectedIndex = 0;
    priceSpinEdit.Value = 0;
    categoryTextEdit.Text = cbxCategory2.Text;
}
private void btnProductEdit_Click(object sender, EventArgs e)
{
    XuLyButton(false);
    productsGridControl.Enabled = false;
    productNameTextEdit.Focus();
    CheckEdit = true;
}
private void btnProductSave_Click(object sender, EventArgs e)
{
    if (priceSpinEdit.Value < 0)
    {
        MessageBox.Show("Đơn giá không hợp lệ !", "Thông báo",
        MessageBoxButtons.OK, MessageBoxIcon.Error);
        return;
    }
    if (productNameTextEdit.Text == string.Empty || categoryTextEdit.Text
    == string.Empty)
    {
        MessageBox.Show("Thông tin chỉnh sửa không được trống !", "Thông
        báo", MessageBoxButtons.OK, MessageBoxIcon.Error);
        return;
    }
    if (CheckEdit)
    {
        try
        {
            productsTableAdapter.UpdateQuery(categoryTextEdit.Text,
            (double)priceSpinEdit.Value, productNameTextEdit.Text,
            Convert.ToInt32(textEdit1.Text));
        }
        catch (Exception ex)
        {
            if (ex.Message.Contains("UNIQUE KEY"))
            {
                MessageBox.Show("Tên sản phẩm đã tồn tại !", "Thông báo",
                MessageBoxButtons.OK, MessageBoxIcon.Error);
            }
            else
            {
                MessageBox.Show(ex.Message);
            }
        }
    }

    CheckEdit = false;

    productsTableAdapter.Fill(this.dataSetProducts.Products);
    productsGridControl.Enabled = true;
    XuLyButton(true);
    return;
}

```

```

        try
        {
            productsTableAdapter.InsertQuery(productNameTextEdit.Text,
categoryTextEdit.Text, (double)priceSpinEdit.Value);
        }
        catch (Exception ex)
        {
            if (ex.Message.Contains("UNIQUE KEY"))
            {
                MessageBox.Show("Tên sản phẩm đã tồn tại !", "Thông báo",
MessageBoxButtons.OK, MessageBoxIcon.Error);
            }
            else
            {
                MessageBox.Show(ex.Message);
            }
        }

        XuLyButton(true);
        productsTableAdapter.Fill(this.dataSetProducts.Products);
        productsGridControl.Enabled = true;
    }

    private void btnProductCancel_Click(object sender, EventArgs e)
    {
        this.productsBindingSource.CancelEdit();
        CheckEdit = false;
        productsGridControl.Enabled = true;
        XuLyButton(true);
    }

    private void btnProductDelete_Click(object sender, EventArgs e)
    {
        productsGridControl.Enabled = false;
        if (MessageBox.Show("Bạn có chắc muốn xóa sản phẩm này ?\nLưu ý: Sản
phẩm sẽ bị xóa trong hóa đơn", "Thông báo", MessageBoxButtons.OKCancel,
MessageBoxIcon.Question) == DialogResult.OK)
        {
            try
            {
                cProducts.DeleteProduct(Convert.ToInt32(textEdit1.Text));
            }
            catch (Exception ex)
            {
                MessageBox.Show("Lỗi " + ex.Message, "Thông báo",
MessageBoxButtons.OK, MessageBoxIcon.Error);
            }
        }
        productsTableAdapter.Fill(this.dataSetProducts.Products);
        productsGridControl.Enabled = true;
        LoadProducts();
    }
}

```

3. Quản lý nhân viên

ID	Tên tài khoản	Họ tên	Giới tính	Ngày tạo	Quyền	Trạng thái
1	admin	Admin	Nam	24-11-2018 21:03:03	Admin	Hoạt động
2	xuanbac	Nguyễn Xuân Bắc	Nam	24-11-2018 21:03:03	Admin	Hoạt động
3	thutrinh	Nguyễn Thị Thu Trinh	Nữ	24-11-2018 21:03:03	Admin	Hoạt động
4	chauht	Lê Ngọc Châu	Nam	24-11-2018 21:03:03	Admin	Hoạt động
5	thodz	Nguyễn Văn Thọ	Nam	24-11-2018 21:03:03	Admin	Hoạt động
6	ngatran	Trần Thị Nga	Nữ	24-11-2018 21:03:03	User	Hoạt động
7	mylinh	Nguyễn Thị Mỹ Linh	Nữ	24-11-2018 21:03:03	User	Hoạt động
8	huyen	Huyền	Nữ	24-11-2018 21:03:03	User	Hoạt động
9	thaibinh	Trần Thị Thái Bình	Nữ	24-11-2018 21:03:03	User	Hoạt động
10	ngocmai	Nguyễn Ngọc Mai	Nữ	24-11-2018 21:03:03	User	Hoạt động
11	thanhnhân	Nguyễn Thị Thanh Nhân	Nữ	24-11-2018 21:03:03	User	Hoạt động
12	thuhien	Nguyễn Thị Thu Hiền	Nữ	24-11-2018 21:03:03	User	Hoạt động
13	ngodbich	Vũ Ngọc Bích	Nữ	24-11-2018 21:03:03	User	Hoạt động
14	hoangyen	Một ngôn tình :)	Nữ	24-11-2018 21:03:03	User	Hoạt động
15	chuchu	Hạ Linh <3	Nữ	24-11-2018 21:03:03	User	Hoạt động

Form quản lý nhân viên gồm các chức năng:

- Hiển thị danh sách tên tài khoản, họ tên, giới tính, ngày tạo, quyền và trạng thái.
- Sửa thông tin tài khoản nhân viên.
- Khóa và mở khóa tài khoản.

Code xử lý:

```
private void XuLyButton2(bool b)
{
    btnEmployeesAdd.Enabled = btnEmployeesDelete.Enabled =
    btnEmployeesEdit.Enabled = b;
    btnEmployeesSave.Enabled = btnEmployeesCancel.Enabled =
    groupBoxEmployeesEdit.Enabled = !b;
}

private void btnEmployeesBlocked_Click(object sender, EventArgs e)
{
    accountsGridControl.Enabled = false;
    if (MessageBox.Show("Bạn có chắc muốn khóa tài khoản này ?", "Thông
báo", MessageBoxButtons.OKCancel, MessageBoxIcon.Question) == DialogResult.OK)
    {
        if (idTextEdit.Text == idAcc.ToString())
        {
            MessageBox.Show("Không thể khóa tài khoản chính bạn !", "Thông
báo", MessageBoxButtons.OK, MessageBoxIcon.Error);
        }
    }
}
```



```

        else if (textEdit2.Text == "Admin")
        {
            MessageBox.Show("Không thể khóa tài khoản Admin !", "Thông báo", MessageBoxButtons.OK, MessageBoxIcon.Error);
        }
        else
        {
            DataProvider.ExecuteNonQuery("UPDATE      dbo.Accounts      SET
            dbo.Accounts.AccountStatus = N'Đã bị khóa' WHERE  dbo.Accounts.id = N'" +
            idTextEdit.Text + "'");

            this.employeesTableAdapter.Fill(this.dataSetEmployees.Employees);
            MessageBox.Show("Khóa tài khoản thành công !", "Thông báo",
            MessageBoxButtons.OK, MessageBoxIcon.Information);
        }
    }
    accountsGridControl.Enabled = true;
}

private void btnEmployeesEdit_Click(object sender, EventArgs e)
{
    CheckEdit2 = true;
    XuLyButton2(false);
    accountsGridControl.Enabled = false;
    fullNameTextEdit.Focus();
}

private void btnEmployeesUnblocked_Click(object sender, EventArgs e)
{
    accountsGridControl.Enabled = false;
    if (MessageBox.Show("Bạn có chắc muốn mở khóa tài khoản này ?", "Thông báo", MessageBoxButtons.OKCancel, MessageBoxIcon.Question) == DialogResult.OK)
    {
        DataProvider.ExecuteNonQuery("UPDATE      dbo.Accounts      SET
        dbo.Accounts.AccountStatus = N'Hoạt động' WHERE  dbo.Accounts.id = N'" +
        idTextEdit.Text + "'");
        this.employeesTableAdapter.Fill(this.dataSetEmployees.Employees);
        MessageBox.Show("Mở khóa tài khoản thành công !", "Thông báo",
        MessageBoxButtons.OK, MessageBoxIcon.Information);
    }
    accountsGridControl.Enabled = true;
}

private void btnEmployeesSave_Click(object sender, EventArgs e)
{
    if (fullNameTextEdit.Text == string.Empty || sexTextEdit.Text ==
    string.Empty)
    {
        MessageBox.Show("Thông tin chỉnh sửa không được trống !", "Thông báo", MessageBoxButtons.OK, MessageBoxIcon.Error);
        return;
    }
    if (CheckEdit2)
    {
        try
        {
            string sex = string.Empty;
            if (radioButton1.Checked)
            {
                sex = "Nam";
            }
        }
    }
}

```

```

        else
        {
            sex = "Nữ";
        }
        DataProvider.ExecuteNonQuery("UPDATE      dbo.Accounts      SET
dbo.Accounts.FullName = N'" + fullNameTextEdit.Text + "', dbo.Accounts.Sex = N'" +
sex + "' WHERE dbo.Accounts.id = '" + idTextEdit.Text + "'");

this.employeesTableAdapter.Fill(this.dataSetEmployees.Employees);
    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.Message);
    }
}
CheckEdit2 = false;
XuLyButton2(true);
accountsGridControl.Enabled = true;
}

private void btnEmployeesCancel_Click(object sender, EventArgs e)
{
    this.employeesBindingSource.CancelEdit();
    CheckEdit2 = false;
    accountsGridControl.Enabled = true;
    XuLyButton2(true);
}

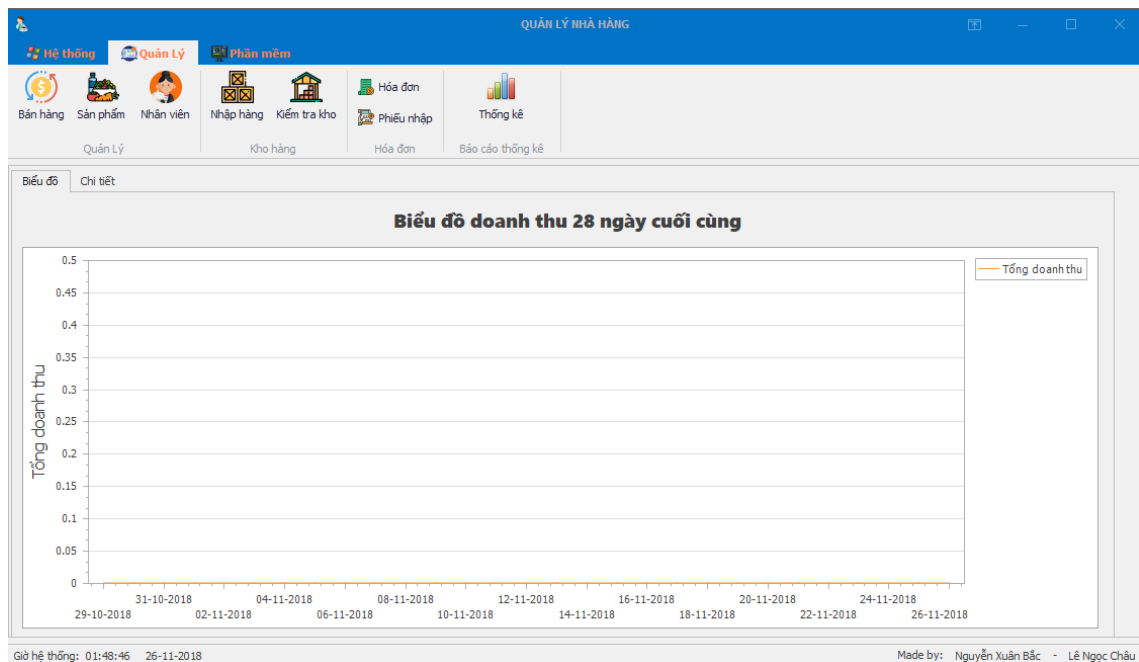
private void btnEmployeesPrevious_Click(object sender, EventArgs e)
{
    this.employeesBindingSource.MovePrevious();
}

private void btnEmployeesNext_Click(object sender, EventArgs e)
{
    this.employeesBindingSource.MoveNext();
}

private void sexTextEdit_EditValueChanged(object sender, EventArgs e)
{
    if (sexTextEdit.Text == "Nam")
    {
        radioButton1.Checked = true;
    }
    else
    {
        radioButton2.Checked = true;
    }
}

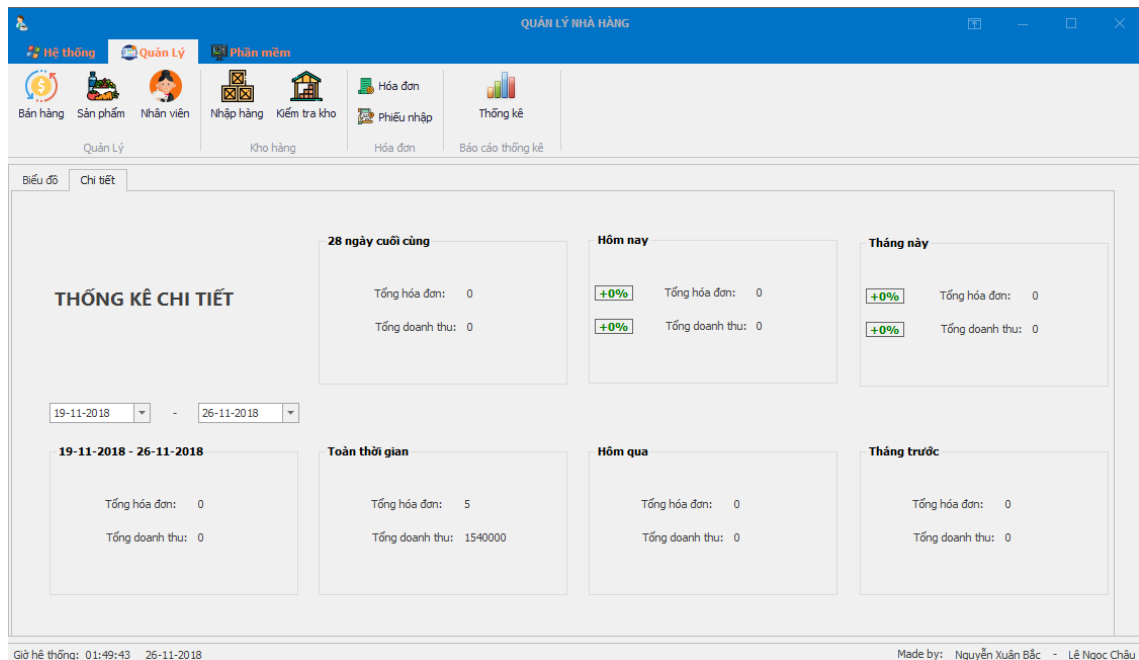
```

4. Quản lý thống kê



Form quản lý thống kê gồm 2 phần:

- Phần Biểu đồ hiển thị doanh thu 28 ngày cuối cùng của việc bán hàng theo dạng biểu đồ đường.



- Phần Chi tiết hiển thị tổng doanh thu, tổng hóa đơn 28 ngày cuối cùng, toàn thời gian, hôm nay, hôm qua, tháng này, tháng trước và hiển thị theo ngày được chọn.

Ngoài ra thống kê Hôm nay và Tháng này sẽ hiện tỷ lệ phần trăm tăng hay giảm so với Hôm qua, Tháng trước.

Code xử lý:

```
private void dateEdit1_EditValueChanged(object sender, EventArgs e)
{
    lblBillCustom.Text = cAnalytics.BillCustom(dateEdit1.Text,
dateEdit2.Text);
    lblRevenueCustom.Text = cAnalytics.RevenueCustom(dateEdit1.Text,
dateEdit2.Text);
    groupBox15.Text = dateEdit1.Text + " - " + dateEdit2.Text;
}

private void dateEdit2_EditValueChanged(object sender, EventArgs e)
{
    lblBillCustom.Text = cAnalytics.BillCustom(dateEdit1.Text,
dateEdit2.Text);
    lblRevenueCustom.Text = cAnalytics.RevenueCustom(dateEdit1.Text,
dateEdit2.Text);
    groupBox15.Text = dateEdit1.Text + " - " + dateEdit2.Text;
}

private void barBtnAnalytics_ItemClick(object sender,
DevExpress.XtraBars.ItemClickEventArgs e)
{
    if (Perminssion != "Admin")
    {
        MessageBox.Show("Bạn không có quyền truy cập !", "Thông báo",
MessageBoxButtons.OK, MessageBoxIcon.Error);
        return;
    }
    pnl0.Visible = true;
    pnl1.Visible = true;
    pnl2.Visible = true;
    pnl3.Visible = true;
    pnl3.BringToFront();

    dateEdit1.Text = DateTime.Now.AddDays(-7).ToString("dd-MM-yyyy");
    dateEdit2.Text = DateTime.Now.ToString("dd-MM-yyyy");
    groupBox15.Text = dateEdit1.Text + " - " + dateEdit2.Text;

    lblBillCustom.Text = cAnalytics.BillCustom(dateEdit1.Text,
dateEdit2.Text);
    lblRevenueCustom.Text = cAnalytics.RevenueCustom(dateEdit1.Text,
dateEdit2.Text);

    lblBillLast28days.Text = cAnalytics.BillLast28days();
    lblRevenueLast28days.Text = cAnalytics.RevenueLast28days();

    lblBillLifetime.Text = cAnalytics.BillLifetime();
    lblRevenueLifetime.Text = cAnalytics.RevenueLifetime();

    lblBillToday.Text = cAnalytics.BillToday();
    lblRevenueToday.Text = cAnalytics.RevenueToday();

    lblBillYesterday.Text = cAnalytics.BillYesterday();
    lblRevenueYesterday.Text = cAnalytics.RevenueYesterday();
}
```

```

lblBillThisMonth.Text = cAnalytics.BillThisMonth();
lblRevenueThisMonth.Text = cAnalytics.RevenueThisMonth();

lblBillLastMonth.Text = cAnalytics.BillLastMonth();
lblRevenueLastMonth.Text = cAnalytics.RevenueLastMonth();

if (Convert.ToDouble(cAnalytics.BillToday()) >=
Convert.ToDouble(cAnalytics.BillYesterday()))
{
    lblPercentBillToday.ForeColor = Color.Green;
    if (Convert.ToDouble(cAnalytics.BillYesterday()) == 0)
    {
        lblPercentBillToday.Text = "+" +
Convert.ToString(Convert.ToDouble(cAnalytics.BillToday()) * 100) + "%";
    }
    else
    {
        lblPercentBillToday.Text = "+" +
Convert.ToString(Math.Round((Convert.ToDouble(cAnalytics.BillToday())
Convert.ToDouble(cAnalytics.BillYesterday()))
Convert.ToDouble(cAnalytics.BillYesterday()) * 100, 1)) + "%";
    }
}
else
{
    lblPercentBillToday.ForeColor = Color.Red;
    lblPercentBillToday.Text = "- " +
Convert.ToString(Math.Round((Convert.ToDouble(cAnalytics.BillYesterday())
Convert.ToDouble(cAnalytics.BillToday()))
Convert.ToDouble(cAnalytics.BillToday()) * 100, 1)) + "%";
}

if (Convert.ToDouble(cAnalytics.RevenueToday()) >=
Convert.ToDouble(cAnalytics.RevenueYesterday()))
{
    lblPercentRevenueToday.ForeColor = Color.Green;
    if (Convert.ToDouble(cAnalytics.RevenueYesterday()) == 0)
    {
        lblPercentRevenueToday.Text = "+" +
Convert.ToString(Convert.ToDouble(cAnalytics.RevenueToday()) * 100) + "%";
    }
    else
    {
        lblPercentRevenueToday.Text = "+" +
Convert.ToString(Math.Round((Convert.ToDouble(cAnalytics.RevenueToday())
Convert.ToDouble(cAnalytics.RevenueYesterday()))
Convert.ToDouble(cAnalytics.RevenueYesterday()) * 100, 1)) + "%";
    }
}
else
{
    lblPercentRevenueToday.ForeColor = Color.Red;
    lblPercentRevenueToday.Text = "- " +
Convert.ToString(Math.Round((Convert.ToDouble(cAnalytics.RevenueYesterday())
Convert.ToDouble(cAnalytics.RevenueToday()))
Convert.ToDouble(cAnalytics.RevenueToday()) * 100, 1)) + "%";
}

//

```

```

        if (Convert.ToDouble(cAnalytics.BillThisMonth()) >=
Convert.ToDouble(cAnalytics.BillLastMonth()))
        {
            lblPercentBillThisMonth.ForeColor = Color.Green;
            if (Convert.ToDouble(cAnalytics.BillLastMonth()) == 0)
            {
                lblPercentBillThisMonth.Text = "+" +
Convert.ToString(Convert.ToDouble(cAnalytics.BillThisMonth()) * 100) + "%";
            }
            else
            {
                lblPercentBillThisMonth.Text = "+" +
Convert.ToString(Math.Round((Convert.ToDouble(cAnalytics.BillThisMonth())
Convert.ToDouble(cAnalytics.BillLastMonth()))
Convert.ToDouble(cAnalytics.BillLastMonth()) * 100, 1)) + "%";
            }
        }
        else
        {
            lblPercentBillThisMonth.ForeColor = Color.Red;
            lblPercentBillThisMonth.Text = "- " +
Convert.ToString(Math.Round((Convert.ToDouble(cAnalytics.BillLastMonth())
Convert.ToDouble(cAnalytics.BillThisMonth()))
Convert.ToDouble(cAnalytics.BillThisMonth()) * 100, 1)) + "%";
        }

        if (Convert.ToDouble(cAnalytics.RevenueThisMonth()) >=
Convert.ToDouble(cAnalytics.RevenueLastMonth()))
        {
            lblPercentRevenueThisMonth.ForeColor = Color.Green;
            if (Convert.ToDouble(cAnalytics.RevenueYesterday()) == 0)
            {
                lblPercentRevenueThisMonth.Text = "+" +
Convert.ToString(Convert.ToDouble(cAnalytics.RevenueThisMonth()) * 100) + "%";
            }
            else
            {
                lblPercentRevenueThisMonth.Text = "+" +
Convert.ToString(Math.Round((Convert.ToDouble(cAnalytics.RevenueThisMonth())
Convert.ToDouble(cAnalytics.RevenueLastMonth()))
Convert.ToDouble(cAnalytics.RevenueLastMonth()) * 100, 1)) + "%";
            }
        }
        else
        {
            lblPercentRevenueThisMonth.ForeColor = Color.Red;
            lblPercentRevenueThisMonth.Text = "- " +
Convert.ToString(Math.Round((Convert.ToDouble(cAnalytics.RevenueLastMonth())
Convert.ToDouble(cAnalytics.RevenueThisMonth()))
Convert.ToDouble(cAnalytics.RevenueThisMonth()) * 100, 1)) + "%";
        }

        //Chart
        string total = string.Empty;
        for (int i = -28; i <= 0; i++)
        {
            foreach (DataRow dr in DataProvider.ExecuteQuery("SELECT SUM(Price
* Amount) AS [Total] FROM dbo.Products, dbo.BillInfo, dbo.Bill WHERE

```

```

dbo.Products.id = dbo.BillInfo.idProduct AND dbo.Bill.id = dbo.BillInfo.idBill AND
FORMAT(PaymentDate, 'yyyy-MM-dd') = '' + DateTime.Now.AddDays(i).ToString("yyyy-
MM-dd") + "").Rows)
    {
        total = dr["Total"].ToString();
    }
    if (total == string.Empty)
    {
        chartControl.Series["Tổng doanh
thu"].Points.AddPoint(DateTime.Now.AddDays(i).ToString("dd-MM-yyyy"), 0);
    }
    else
    {
        chartControl.Series["Tổng doanh
thu"].Points.AddPoint(DateTime.Now.AddDays(i).ToString("dd-MM-yyyy"),
Convert.ToDouble(total));
    }
}
}

```

Class cAnalytics:

```

public class cAnalytics
{
    public static string BillLast28days()
    {
        int a = (int)DataProvider.ExecuteScalar("SELECT COUNT(id) FROM
dbo.Bill WHERE FORMAT(PaymentDate, 'yyyy-MM-dd') > '' + DateTime.Now.AddDays(-
28).ToString("yyyy-MM-dd") + "");
        return a.ToString();
    }

    public static string RevenueLast28days()
    {
        string total = string.Empty;

        foreach (DataRow dr in DataProvider.ExecuteQuery("SELECT SUM(Price *
Amount) AS [Total] FROM dbo.Products, dbo.BillInfo, dbo.Bill WHERE dbo.Products.id
= dbo.BillInfo.idProduct AND dbo.Bill.id = dbo.BillInfo.idBill AND
FORMAT(PaymentDate, 'yyyy-MM-dd') > '' + DateTime.Now.AddDays(-28).ToString("yyyy-
MM-dd") + "").Rows)
        {
            total = dr["Total"].ToString();
        }

        if (total == string.Empty)
        {
            return "0";
        }
        return total;
    }

    public static string BillLifetime()
    {
        int a = (int)DataProvider.ExecuteScalar("SELECT COUNT(id) FROM
dbo.Bill");
        return a.ToString();
    }
}

```

```

public static string RevenueLifetime()
{
    string total = string.Empty;

    foreach (DataRow dr in DataProvider.ExecuteQuery("SELECT SUM(Price *
Amount) AS [Total] FROM dbo.Products, dbo.BillInfo, dbo.Bill WHERE dbo.Products.id
= dbo.BillInfo.idProduct AND dbo.Bill.id = dbo.BillInfo.idBill").Rows)
    {
        total = dr["Total"].ToString();
    }

    if (total == string.Empty)
    {
        return "0";
    }
    return total;
}

public static string BillToday()
{
    int a = (int)DataProvider.ExecuteScalar("SELECT COUNT(id) FROM
dbo.Bill WHERE FORMAT(PaymentDate, 'yyyy-MM-dd') = '' +
DateTime.Now.ToString('yyyy-MM-dd') + '');
    return a.ToString();
}

public static string RevenueToday()
{
    string total = string.Empty;

    foreach (DataRow dr in DataProvider.ExecuteQuery("SELECT SUM(Price *
Amount) AS [Total] FROM dbo.Products, dbo.BillInfo, dbo.Bill WHERE dbo.Products.id
= dbo.BillInfo.idProduct AND dbo.Bill.id = dbo.BillInfo.idBill AND
FORMAT(PaymentDate, 'yyyy-MM-dd') = '' + DateTime.Now.ToString('yyyy-MM-dd') +
'').Rows)
    {
        total = dr["Total"].ToString();
    }

    if (total == string.Empty)
    {
        return "0";
    }
    return total;
}

public static string BillYesterday()
{
    int a = (int)DataProvider.ExecuteScalar("SELECT COUNT(id) FROM
dbo.Bill WHERE FORMAT(PaymentDate, 'yyyy-MM-dd') = '' + DateTime.Now.AddDays(-
1).ToString('yyyy-MM-dd') + '');
    return a.ToString();
}

public static string RevenueYesterday()
{
    string total = string.Empty;

    foreach (DataRow dr in DataProvider.ExecuteQuery("SELECT SUM(Price *
Amount) AS [Total] FROM dbo.Products, dbo.BillInfo, dbo.Bill WHERE dbo.Products.id

```



```

=     dbo.BillInfo.idProduct    AND     dbo.Bill.id     =     dbo.BillInfo.idBill    AND
FORMAT(PaymentDate, 'yyyy-MM-dd') = '' + DateTime.Now.AddDays(-1).ToString("yyyy-
MM-dd") + "").Rows)
    {
        total = dr["Total"].ToString();
    }

    if (total == string.Empty)
    {
        return "0";
    }
    return total;
}

public static string BillThisMonth()
{
    int a = (int)DataProvider.ExecuteScalar("SELECT COUNT(id) FROM
dbo.Bill WHERE FORMAT(PaymentDate, 'yyyy-MM') = '' + DateTime.Now.ToString("yyyy-
MM") + "");
    return a.ToString();
}

public static string RevenueThisMonth()
{
    string total = string.Empty;

    foreach (DataRow dr in DataProvider.ExecuteQuery("SELECT SUM(Price *
Amount) AS [Total] FROM dbo.Products, dbo.BillInfo, dbo.Bill WHERE dbo.Products.id
=     dbo.BillInfo.idProduct    AND     dbo.Bill.id     =     dbo.BillInfo.idBill    AND
FORMAT(PaymentDate, 'yyyy-MM') = '' + DateTime.Now.ToString("yyyy-MM") + "").Rows)
    {
        total = dr["Total"].ToString();
    }

    if (total == string.Empty)
    {
        return "0";
    }
    return total;
}

public static string BillLastMonth()
{
    int a = (int)DataProvider.ExecuteScalar("SELECT COUNT(id) FROM
dbo.Bill WHERE FORMAT(PaymentDate, 'yyyy-MM') = '' + DateTime.Now.AddMonths(-
1).ToString("yyyy-MM") + "");
    return a.ToString();
}

public static string RevenueLastMonth()
{
    string total = string.Empty;

    foreach (DataRow dr in DataProvider.ExecuteQuery("SELECT SUM(Price *
Amount) AS [Total] FROM dbo.Products, dbo.BillInfo, dbo.Bill WHERE dbo.Products.id
=     dbo.BillInfo.idProduct    AND     dbo.Bill.id     =     dbo.BillInfo.idBill    AND
FORMAT(PaymentDate, 'yyyy-MM') = '' + DateTime.Now.AddMonths(-1).ToString("yyyy-
MM") + "").Rows)
    {
        total = dr["Total"].ToString();
    }
}

```

```

        if (total == string.Empty)
        {
            return "0";
        }
        return total;
    }

    public static string BillCustom(string date1, string date2)
    {
        int a = (int)DataProvider.ExecuteScalar("SELECT COUNT(id) FROM
        dbo.Bill WHERE FORMAT(PaymentDate, 'yyyy-MM') >= ' " + date1 + " ' AND
        FORMAT(PaymentDate, 'yyyy-MM') <= ' " + date2 + "'");
        return a.ToString();
    }

    public static string RevenueCustom(string date1, string date2)
    {
        string total = string.Empty;

        foreach (DataRow dr in DataProvider.ExecuteQuery("SELECT SUM(Price *
        Amount) AS [Total] FROM dbo.Products, dbo.BillInfo, dbo.Bill WHERE dbo.Products.id
        = dbo.BillInfo.idProduct AND dbo.Bill.id = dbo.BillInfo.idBill AND
        FORMAT(PaymentDate, 'yyyy-MM') >= ' " + date1 + " ' AND FORMAT(PaymentDate, 'yyyy-
        MM') <= ' " + date2 + "'").Rows)
        {
            total = dr["Total"].ToString();
        }

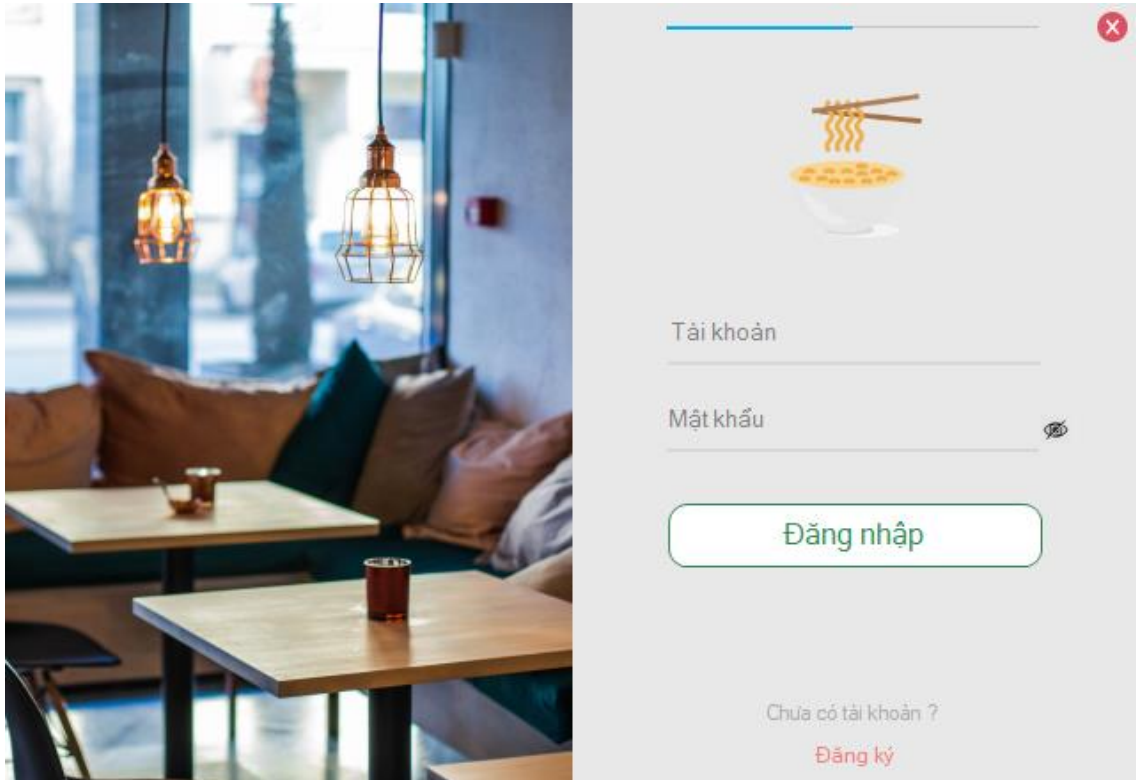
        if (total == string.Empty)
        {
            return "0";
        }
        return total;
    }
}

```

II – Các Form giải quyết chức năng

1. Form đăng nhập, đăng ký

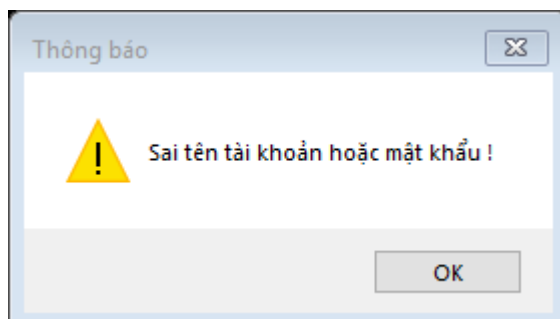
Form đăng nhập:



Giao diện form đăng nhập bao gồm một Panel control chứa ảnh ở bên trái, bên phải phần đăng nhập bao gồm hai Textbox tài khoản và mật khẩu, một Button đăng nhập.

Ngoài ra còn có thêm Button (hình con mắt) giúp hiện và ẩn mật khẩu.

* Khi người dùng đăng nhập vào hệ thống với tài khoản hoặc mật khẩu không đúng sẽ hiển thị thông báo lỗi.



* Nếu đăng nhập thành công, chương trình sẽ mở một form mới.

Class DataProvider

```
public class DataProvider
{
    private static string connectionSTR = @"Data Source=.;Initial
Catalog=QuanLyNhaHang;Integrated Security=True;Connect Timeout=10";

    public static string dbName()
    {
        SqlConnection connection = new SqlConnection(connectionSTR);
        return connection.Database.ToString();
    }

    public static DataTable ExecuteQuery(string query, object[] parameter =
null)
    {
        DataTable data = new DataTable();
        try
        {
            using (SqlConnection connection = new
SqlConnection(connectionSTR))
            {
                connection.Open();

                SqlCommand command = new SqlCommand(query, connection);

                if (parameter != null)
                {
                    string[] listPara = query.Split(' ');
                    int i = 0;
                    foreach (string item in listPara)
                    {
                        if (item.Contains('@'))
                        {
                            command.Parameters.AddWithValue(item,
parameter[i]);
                            i++;
                        }
                    }
                }

                SqlDataAdapter adapter = new SqlDataAdapter(command);

                adapter.Fill(data);

                connection.Close();
            }
        }
        catch (SqlException ex)
        {
            MessageBox.Show("Lỗi: " + ex.Message, "Thông báo",
MessageBoxButtons.OK, MessageBoxIcon.Error);
            return null;
        }

        return data;
    }
}
```

```

}

public static int ExecuteNonQuery(string query, object[] parameter = null)
{
    int data = 0;

    using (SqlConnection connection = new SqlConnection(connectionSTR))
    {
        connection.Open();

        SqlCommand command = new SqlCommand(query, connection);

        if (parameter != null)
        {
            string[] listPara = query.Split(' ');
            int i = 0;
            foreach (string item in listPara)
            {
                if (item.Contains('@'))
                {
                    command.Parameters.AddWithValue(item, parameter[i]);
                    i++;
                }
            }
        }

        data = command.ExecuteNonQuery();

        connection.Close();
    }

    return data;
}

public static object ExecuteScalar(string query, object[] parameter = null)
{
    object data = 0;

    using (SqlConnection connection = new SqlConnection(connectionSTR))
    {
        connection.Open();

        SqlCommand command = new SqlCommand(query, connection);

        if (parameter != null)
        {
            string[] listPara = query.Split(' ');
            int i = 0;
            foreach (string item in listPara)
            {
                if (item.Contains('@'))
                {
                    command.Parameters.AddWithValue(item, parameter[i]);
                    i++;
                }
            }
        }

        data = command.ExecuteScalar();

        connection.Close();
    }
}

```

```

    }
    return data;
}
}

```

Code xử lý khi bấm button đăng nhập:

```

private void btnLogin_Click(object sender, EventArgs e)
{
    progress.Visible = true;
    new Thread(() => {
        string salt = string.Empty;
        if (DataProvider.ExecuteQuery("SELECT Salt FROM dbo.Accounts WHERE
dbo.Accounts.Username = '" + tbxLoginUsername.Text + "'" ) != null)
        {
            foreach (DataRow d in DataProvider.ExecuteQuery("SELECT
dbo.Accounts.Salt FROM dbo.Accounts WHERE dbo.Accounts.Username = '" +
tbxLoginUsername.Text + "'" ).Rows)
            {
                salt = d["Salt"].ToString();
            }
        }
        else
        {
            MessageBox.Show("Sai tên tài khoản hoặc mật khẩu !", "Thông
báo", MessageBoxButtons.OK, MessageBoxIcon.Warning);
            return;
        }

        string encryptPassword = Encrypt("skagre" + tbxLoginPassword.Text
+ salt);

        if (DataProvider.ExecuteQuery("SELECT dbo.Accounts.id FROM
dbo.Accounts WHERE dbo.Accounts.Username = '" + tbxLoginUsername.Text + "' AND
dbo.Accounts.Password = '" + encryptPassword + "'" ).Rows.Count > 0)
        {
            string idAccount = string.Empty;
            string permission = string.Empty;
            string accountStatus = string.Empty;
            foreach (DataRow dr in DataProvider.ExecuteQuery("SELECT
dbo.Accounts.id, dbo.Accounts.Permission, dbo.Accounts.AccountStatus FROM
dbo.Accounts WHERE dbo.Accounts.Username = '" + tbxLoginUsername.Text + "' AND
dbo.Accounts.Password = '" + encryptPassword + "'" ).Rows)
            {
                idAccount = dr["id"].ToString();
                permission = dr["Permission"].ToString();
                accountStatus = dr["AccountStatus"].ToString();
            }

            if (accountStatus != "Hoạt động")
            {
                MessageBox.Show("Tài khoản của bạn đã bị khóa", "Thông
báo", MessageBoxButtons.OK, MessageBoxIcon.Warning);
                return;
            }
            fMain f = new fMain();
            f.idAcc = int.Parse(idAccount);
            f.Perminssion = permission;
        }
    });
}

```

```

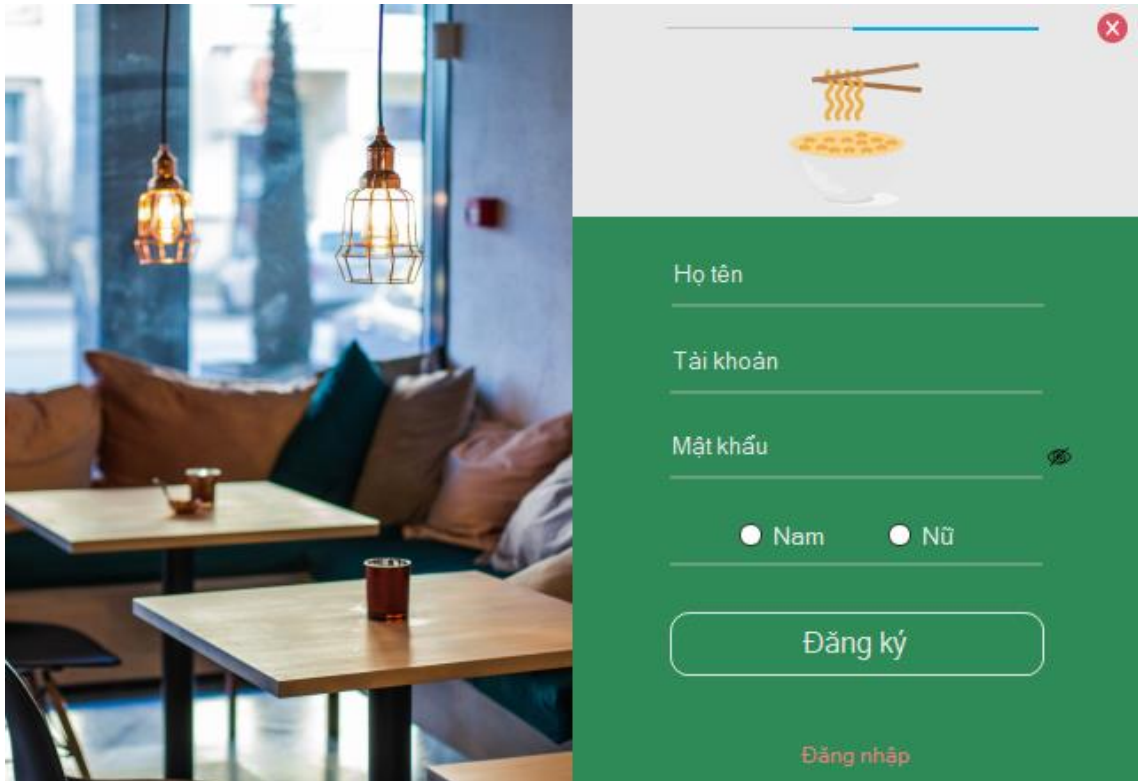
        progress.Visible = false;

        if (this.InvokeRequired)
        {
            this.BeginInvoke((MethodInvoker)delegate ()
            {
                this.Hide();
                f.ShowDialog();
                this.Show();
            });
        }
        else
        {
            this.Hide();
            f.ShowDialog();
            this.Show();
        }

        tbxLoginUsername.Text = "Tài khoản";
        tbxLoginPassword.Text = "Mật khẩu";
        tbxLoginPassword.PasswordChar = '\0';

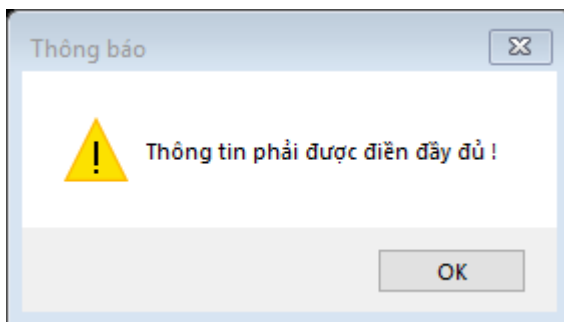
        this.ActiveControl = btnLogin;
    }
    else
    {
        MessageBox.Show("Sai tên tài khoản hoặc mật khẩu !", "Thông
báo", MessageBoxButtons.OK, MessageBoxIcon.Warning);
        progress.Visible = false;
    }
    }).Start();
}
private string Encrypt(string str)
{
    MD5 md5 = new MD5CryptoServiceProvider();
    md5.ComputeHash(ASCIIEncoding.ASCII.GetBytes(str));
    byte[] result = md5.Hash;
    StringBuilder strBuilder = new StringBuilder();
    for (int i = 0; i < result.Length; i++)
    {
        strBuilder.Append(result[i].ToString("x2"));
    }
    return strBuilder.ToString();
}

```

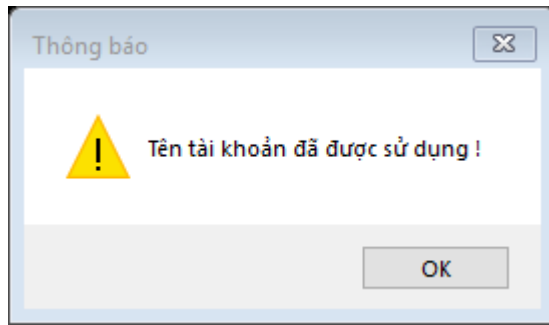


Giao diện form đăng ký cũng nằm trên cùng một form với form đăng nhập bao gồm ba Textbox họ tên, tài khoản và mật khẩu, radio button để chọn giới tính nam hay nữ, một Button đăng ký.

* Khi người dùng click vào button đăng ký, hệ thống sẽ kiểm tra tất cả thông tin ở Textbox đã nhập đầy đủ chưa, nếu chưa đầy đủ sẽ hiện thông báo lỗi.



* Nếu người dùng nhập đầy đủ tất cả thông tin ở Textbox nhưng tên tài khoản đã trùng với tài khoản có sẵn, chương trình sẽ hiện thông báo lỗi.



* Nếu không xảy ra lỗi, chương trình sẽ hiển thị thông báo đăng ký thành công.

Code xử lý khi bấm button đăng ký:

```
private void btnRegister_Click(object sender, EventArgs e)
{
    if (tbxRegisterFullName.Text == string.Empty ||
        tbxRegisterUsername.Text == string.Empty || tbxRegisterPassword.Text ==
        string.Empty || tbxRegisterFullName.Text == "Họ tên" || tbxRegisterUsername.Text
        == "Tài khoản" || tbxRegisterPassword.Text == "Mật khẩu" || rBtnMale.Checked ==
        false && rBtnFemale.Checked == false)
    {
        MessageBox.Show("Thông tin phải được điền đầy đủ !", "Thông báo",
        MessageBoxButtons.OK, MessageBoxIcon.Warning);
    }
    else
    {
        if (DataProvider.ExecuteQuery("SELECT dbo.Accounts.Username FROM
        dbo.Accounts WHERE dbo.Accounts.Username = '" + tbxRegisterUsername.Text +
        "'").Rows.Count > 0)
        {
            MessageBox.Show("Tên tài khoản đã được sử dụng !", "Thông báo",
            MessageBoxButtons.OK, MessageBoxIcon.Warning);
            return;
        }
        Random random = new Random();
        string salt = random.Next(100000, 999999).ToString();
        string sex = "Nam";
        sex = rBtnMale.Checked ? "Nam" : "Nữ";
        string encryptPassword = Encrypt("skagre" +
        tbxRegisterPassword.Text + salt);
        encryptPassword.Replace("'", "''");
        DataProvider.ExecuteQuery("INSERT INTO dbo.Accounts (Username,
        Password, FullName, Sex, Salt) VALUES ('" + tbxRegisterUsername.Text + "', '" +
        encryptPassword + "', N'" + tbxRegisterFullName.Text + "', N'" + sex + "', '" + salt
        + "'");
        MessageBox.Show("Đăng ký thành công !", "Thông báo",
        MessageBoxButtons.OK, MessageBoxIcon.Information);
        tbxRegisterFullName.Text = "Họ tên";
        tbxRegisterUsername.Text = "Tài khoản";
        tbxRegisterPassword.Text = "Mật khẩu";
        tbxRegisterPassword.PasswordChar = '\0';
        rBtnMale.Checked = false;
        rBtnFemale.Checked = false;
    }
}
```

2. Form thông tin người dùng

CHỈNH SỬA THÔNG TIN

ID: 1

Tên tài khoản: admin

Họ tên: Admin

Giới tính: ☒ Nam ☐ Nữ

Ngày tạo: 24-11-2018 21:03:03

Quyền: Admin

Trạng thái: Hoạt động

Lưu lại **Hủy bỏ**

Form thông tin người dùng hiển thị tất cả thông tin bao gồm:

- ID: Hiển thị mã tài khoản
- Tên tài khoản: Tên tài khoản mà người dùng đã đặt
- Họ tên: Họ tên của người dùng
- Giới tính: Giới tính nam, nữ
- Ngày tạo: Hiển thị ngày, giờ mà người dùng tạo tài khoản
- Quyền: Admin, user. Mặc định khi đăng ký là User
- Trạng thái: Hoạt động, đã bị khóa

Ở form này, người dùng chỉ có thể chỉnh sửa họ tên và giới tính.

Khi click button lưu lại, hệ thống sẽ hiển thị thông báo cập nhật thông tin thành công.

Code xử lý của fAccountInfo:

```
public fAccountInfo()
{
    InitializeComponent();
}
public int idAcc;

private void fAccountInfo_Load(object sender, EventArgs e)
{
```

```

        if (DataProvider.ExecuteQuery("SELECT dbo.Accounts.id,
        dbo.Accounts.Username, dbo.Accounts.FullName, dbo.Accounts.Sex,
        FORMAT(dbo.Accounts.DateCreated, 'dd-MM-yyyy HH:mm:ss') AS DateCreated,
        dbo.Accounts.Permission, dbo.Accounts.AccountStatus FROM dbo.Accounts WHERE id = "
        + idAcc) != null)
        {
            foreach (DataRow dr in DataProvider.ExecuteQuery("SELECT
        dbo.Accounts.id, dbo.Accounts.Username, dbo.Accounts.FullName, dbo.Accounts.Sex,
        FORMAT(dbo.Accounts.DateCreated, 'dd-MM-yyyy HH:mm:ss') AS DateCreated,
        dbo.Accounts.Permission, dbo.Accounts.AccountStatus FROM dbo.Accounts WHERE id = "
        + idAcc).Rows)
            {
                tbxID.Text = dr["id"].ToString();
                tbxUsername.Text = dr["Username"].ToString();
                tbxFullName.Text = dr["FullName"].ToString();
                if (dr["Sex"].ToString() == "Nam")
                {
                    rBtnMale.Checked = true;
                }
                else
                {
                    rBtnFemale.Checked = true;
                }
                tbxDayCreated.Text = dr["DateCreated"].ToString();
                tbxPermission.Text = dr["Permission"].ToString();
                tbxStatus.Text = dr["AccountStatus"].ToString();
            }
        }
    }

    private void btnOK_Click(object sender, EventArgs e)
    {
        string sex = string.Empty;
        if (rBtnMale.Checked == true)
        {
            sex = "Nam";
        }
        else
        {
            sex = "Nữ";
        }
        DataProvider.ExecuteQuery("UPDATE dbo.Accounts SET FullName = N'" +
        tbxFullName.Text + "', Sex = N'" + sex + "' WHERE id = " + idAcc);
        MessageBox.Show("Cập nhật thành công!", "Thông báo",
        MessageBoxButtons.OK, MessageBoxIcon.Information);

        this.Close();
    }

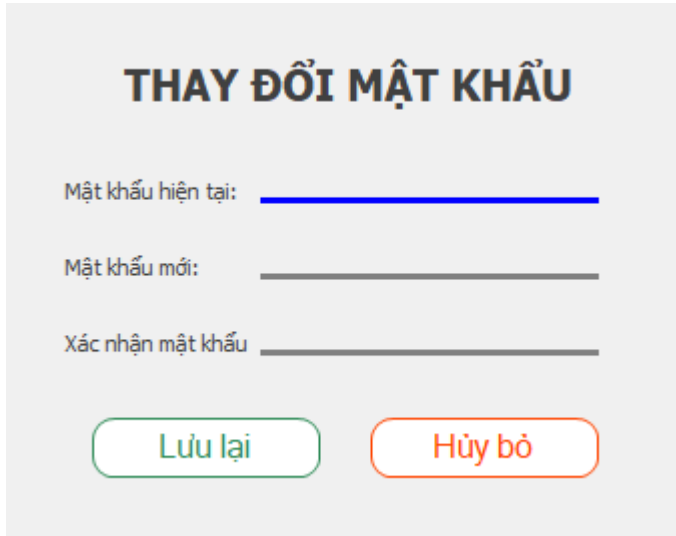
    private void btnCancel_Click(object sender, EventArgs e)
    {
        this.Close();
    }

    private void tbxFullName_KeyPress(object sender, KeyPressEventArgs e)
    {
        if (!Char.IsLetter(e.KeyChar) && (Keys)e.KeyChar != Keys.Back &&
        (Keys)e.KeyChar != Keys.Space)
        {
            e.Handled = true;
        }
    }

```

}

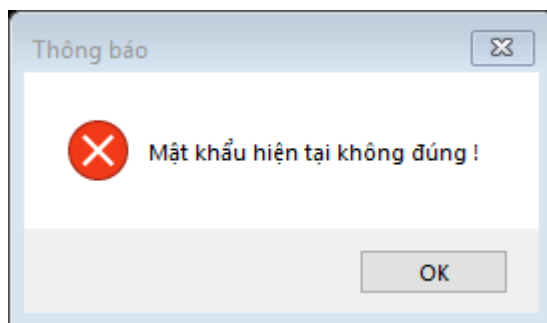
3. Form thay đổi mật khẩu



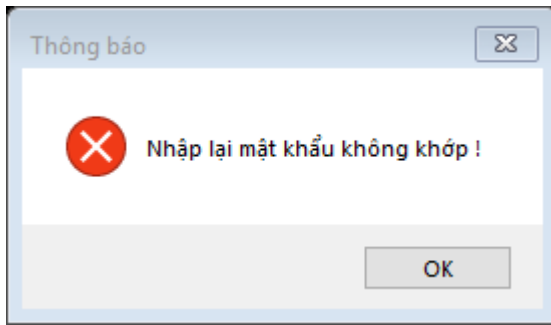
The image shows a web form titled "THAY ĐỔI MẬT KHẨU" (Change Password). It contains three text input fields: "Mật khẩu hiện tại:" (Current password), "Mật khẩu mới:" (New password), and "Xác nhận mật khẩu" (Confirm password). Below the fields are two buttons: "Lưu lại" (Save) in green and "Hủy bỏ" (Cancel) in orange.

Form thay đổi mật khẩu gồm ba Textbox mật khẩu hiện tại, mật khẩu mới, xác nhận mật khẩu.

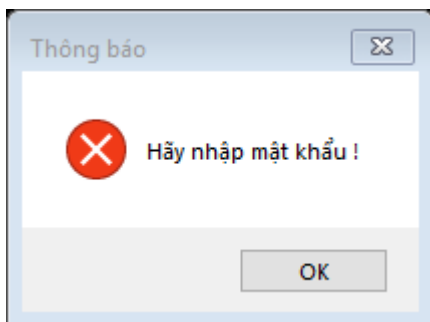
* Nếu khi click button lưu lại, mật khẩu hiện tại không đúng sẽ hiển thị thông báo lỗi.



* Nếu mật khẩu mới và xác nhận mật khẩu không giống nhau sẽ hiển thị thông báo lỗi.



* Nếu để trống một hay nhiều textbox, chương trình sẽ hiển thị thông báo lỗi.



* Khi không xảy ra lỗi, chương trình sẽ thông báo đổi mật khẩu thành công.

Code xử lý của fChangePassword:

```
public fChangePassword()
{
    InitializeComponent();
}
public int idAcc;

private void btnCancel_Click(object sender, EventArgs e)
{
    this.Close();
}

private void btnOK_Click(object sender, EventArgs e)
{
    if (tbxNewPassword.Text == string.Empty || tbxConfirmNewPassword.Text
    == string.Empty)
    {
        MessageBox.Show("Hãy nhập mật khẩu !", "Thông báo",
        MessageBoxButtons.OK, MessageBoxIcon.Error);
    }
    else if (tbxNewPassword.Text != tbxConfirmNewPassword.Text)
    {
        MessageBox.Show("Nhập lại mật khẩu không khớp !", "Thông báo",
        MessageBoxButtons.OK, MessageBoxIcon.Error);
    }
    else
    {
        string salt = string.Empty, pass = string.Empty;
        if (DataProvider.ExecuteQuery("SELECT dbo.Accounts.Password,
        dbo.Accounts.Salt FROM dbo.Accounts WHERE id = " + idAcc) != null)
        {

```

```

        foreach (DataRow d in DataProvider.ExecuteQuery("SELECT
dbo.Accounts.Password, dbo.Accounts.Salt FROM dbo.Accounts WHERE id = " +
idAcc).Rows)
        {
            salt = d["Salt"].ToString();
            pass = d["Password"].ToString();
        }
        string encryptPassword = Encrypt("skagre" +
tbxCurrentPassWord.Text + salt);
        if (encryptPassword != pass)
        {
            MessageBox.Show("Mật khẩu hiện tại không đúng !", "Thông báo",
MessageBoxButtons.OK, MessageBoxIcon.Error);
            tbxCurrentPassWord.Text = string.Empty;
            tbxNewPassword.Text = string.Empty;
            tbxConfirmNewPassword.Text = string.Empty;
            return;
        }
        DataProvider.ExecuteQuery("UPDATE dbo.Accounts SET Password = '" +
Encrypt("skagre" + tbxNewPassword.Text + salt) + "' WHERE id = " + idAcc);
        MessageBox.Show("Cập nhật thành công !", "Thông báo",
MessageBoxButtons.OK, MessageBoxIcon.Information);

        this.Close();
    }
}

private string Encrypt(string str)
{
    MD5 md5 = new MD5CryptoServiceProvider();
    md5.ComputeHash(ASCIIEncoding.ASCII.GetBytes(str));
    byte[] result = md5.Hash;
    StringBuilder strBuilder = new StringBuilder();
    for (int i = 0; i < result.Length; i++)
    {
        strBuilder.Append(result[i].ToString("x2"));
    }
    return strBuilder.ToString();
}

private void tbxCurrentPassWord_Enter(object sender, EventArgs e)
{
    tbxCurrentPassWord.IsPassword = true;
}

private void tbxNewPassword_Enter(object sender, EventArgs e)
{
    tbxNewPassword.IsPassword = true;
}

private void tbxConfirmNewPassword_Enter(object sender, EventArgs e)
{
    tbxConfirmNewPassword.IsPassword = true;
}

private void fChangePassword_Load(object sender, EventArgs e)
{
    this.ActiveControl = tbxCurrentPassWord;
}

```

III – Các chức năng phụ

1. Sao lưu dữ liệu

Khi click vào button Sao lưu dữ liệu tại tab Hệ thống, chương trình sẽ hiển thị một thông báo hỏi người dùng có muốn sao lưu dữ liệu hay không.

Nếu người dùng chọn OK, chương trình sẽ cho phép người dùng chọn nơi lưu và chương trình sẽ xuất ra một file .bak. File này có tác dụng để khôi phục dữ liệu.

Code xử lý sao lưu dữ liệu:

```
private void barBtnBackup_ItemClick(object sender, DevExpress.XtraBars.ItemClickEventArgs e)
{
    if (Perminssion != "Admin")
    {
        MessageBox.Show("Bạn không có quyền truy cập !", "Thông báo",
        MessageBoxButtons.OK, MessageBoxIcon.Error);
        return;
    }
    if (MessageBox.Show("Bạn có muốn sao lưu database ?", "Thông báo",
    MessageBoxButtons.OKCancel, MessageBoxIcon.Question) == DialogResult.OK)
    {
        FolderBrowserDialog folderDialog = new FolderBrowserDialog();
        if (folderDialog.ShowDialog() == DialogResult.OK)
        {
            if (DataProvider.ExecuteQuery("BACKUP DATABASE " +
            DataProvider.dbName() + " TO DISK = '" + folderDialog.SelectedPath + "\\backupDB"
            + "-" + DateTime.Now.ToString("dd-MM-yyyy--HH-mm-ss") + ".bak'") == null)
            {
                return;
            }
            MessageBox.Show("Backup Database thành công !", "Thông báo",
            MessageBoxButtons.OK, MessageBoxIcon.Information);
        }
    }
}
```

2. Khôi phục dữ liệu

Khi click vào button Phục hồi dữ liệu tại tab Hệ thống, chương trình sẽ hiển thị một thông báo hỏi người dùng có muốn phục hồi dữ liệu hay không.

Nếu người dùng chọn OK, chương trình sẽ cho phép người dùng chọn đến file .bak. Chương trình sẽ phục hồi dữ liệu từ file này.

Code xử lý phục hồi dữ liệu:

```
private void barBtnRestore_ItemClick(object sender, DevExpress.XtraBars.ItemClickEventArgs e)
{
    if (Perminssion != "Admin")
    {
        MessageBox.Show("Bạn không có quyền truy cập !", "Thông báo",
        MessageBoxButtons.OK, MessageBoxIcon.Error);
        return;
    }
    if (MessageBox.Show("Bạn có muốn phục hồi database ?", "Thông báo",
    MessageBoxButtons.OKCancel, MessageBoxIcon.Question) == DialogResult.OK)
    {
        OpenFileDialog openFileDialog = new OpenFileDialog();
        if (openFileDialog.ShowDialog() == DialogResult.OK)
        {
            if (DataProvider.ExecuteQuery("RESTORE DATABASE " +
            DataProvider.dbName() + " FROM DISK = '" + openFileDialog.FileName + "'") == null)
            {
                return;
            }
            MessageBox.Show("Phục hồi Database thành công !", "Thông báo",
            MessageBoxButtons.OK, MessageBoxIcon.Information);
        }
    }
}
```

```

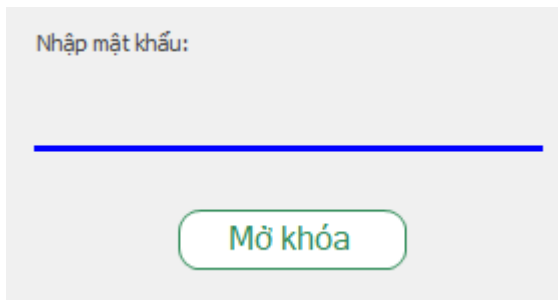
        if (Perminssion != "Admin")
        {
            MessageBox.Show("Bạn không có quyền truy cập !", "Thông báo",
                MessageBoxButtons.OK, MessageBoxIcon.Error);
            return;
        }
        if (MessageBox.Show("Bạn có muốn khôi phục database ?", "Thông báo",
            MessageBoxButtons.OKCancel, MessageBoxIcon.Question) == DialogResult.OK)
        {
            OpenFileDialog openFileDialog = new OpenFileDialog();
            openFileDialog.Filter = "SQL SERVER database backup files|*.bak";
            openFileDialog.Title = "Database restore";
            if (openFileDialog.ShowDialog() == DialogResult.OK)
            {
                DataProvider.ExecuteNonQuery("ALTER DATABASE [" +
                    DataProvider.dbName() + "] SET SINGLE_USER WITH ROLLBACK IMMEDIATE");
                DataProvider.ExecuteNonQuery("USE MASTER RESTORE DATABASE [" +
                    DataProvider.dbName() + "] FROM DISK='" + openFileDialog.FileName + "' WITH REPLACE;");
                DataProvider.ExecuteNonQuery("ALTER DATABASE [" +
                    DataProvider.dbName() + "] SET MULTI_USER");

                MessageBox.Show("Restore Database thành công !", "Thông báo",
                    MessageBoxButtons.OK, MessageBoxIcon.Information);
            }
        }
    }
}

```

3. Chức năng khóa màn hình

Đây là một chức năng phụ, để khóa màn hình tạm thời.



Với chức năng này, người dùng không thể thao tác bất kỳ chức năng nào khác cho tới khi mở khóa.

Để mở khóa, người dùng chỉ cần nhập đúng mật khẩu đăng nhập của mình.

Code xử lý của fLockScreen:

```

public fLockScreen()
{
    InitializeComponent();
}
public int idAcc;

```



```

private bool Check = false;

public bool check { get { return Check; } }

private void tbxInput_Enter(object sender, EventArgs e)
{
    tbxInput.IsPassword = true;
}

private void btnOK_Click(object sender, EventArgs e)
{
    string salt = string.Empty, pass = string.Empty;
    if (DataProvider.ExecuteQuery("SELECT dbo.Accounts.Salt FROM
dbo.Accounts WHERE id = " + idAcc) != null)
    {
        foreach (DataRow d in DataProvider.ExecuteQuery("SELECT
dbo.Accounts.Password, dbo.Accounts.Salt FROM dbo.Accounts WHERE id = " +
idAcc).Rows)
        {
            salt = d["Salt"].ToString();
            pass = d["Password"].ToString();
        }
    }
    string encryptPassword = Encrypt("skagre" + tbxInput.Text + salt);
    if (encryptPassword != pass)
    {
        MessageBox.Show("Mật khẩu không đúng!", "Thông báo",
        MessageBoxButtons.OK, MessageBoxIcon.Error);
        return;
    }
    Check = true;
    this.Close();
}

private string Encrypt(string str)
{
    MD5 md5 = new MD5CryptoServiceProvider();
    md5.ComputeHash(ASCIIEncoding.ASCII.GetBytes(str));
    byte[] result = md5.Hash;
    StringBuilder strBuilder = new StringBuilder();
    for (int i = 0; i < result.Length; i++)
    {
        strBuilder.Append(result[i].ToString("x2"));
    }
    return strBuilder.ToString();
}

private void fLockScreen_Load(object sender, EventArgs e)
{
    this.ActiveControl = tbxInput;
}

```

4. Thay đổi giao diện chương trình

Tại đây người dùng có thể chọn giao diện cho chương trình, có rất nhiều loại giao diện khác nhau, đẹp cho người dùng lựa chọn.

5. Mã hóa mật khẩu

Để đảm bảo tính bảo mật, mật khẩu người dùng được mã hóa md5 theo dạng:

“Chuỗi cố định” + Mật khẩu người dùng đặt + “Chuỗi random”

Trong đó:

- Chuỗi cố định nằm ở trong code, chuỗi này là một chuỗi bất kỳ do người viết chương trình đặt. VD: “skagre”, “xuanbac”,...

- Chuỗi random chuỗi này phát sinh random trong code và được lưu vào cơ sở dữ liệu. VD: “159357”, “123456”,...

- * Nếu mã hóa theo cách thông thường là chỉ mã hóa mật khẩu người dùng, không nối các chuỗi cố định hay random vào sẽ rất dễ để dịch ngược ra được mật khẩu.

- * Do đó cần phải mã hóa mật khẩu theo dạng này để bảo mật cao hơn, khó dịch ngược hơn nếu chẳng may bị hack cơ sở dữ liệu...

KẾT LUẬN

Đề tài “Quản lý nhà hàng – C#” cũng xuất phát từ thực tế ngày nay nhằm tạo một nền tảng cơ sở ban đầu, hỗ trợ thêm để thiết kế một chương trình cho công ty hoặc một chương trình hỗ trợ cá nhân.

Mặc dù đã có nhiều cố gắng, từ các kiến thức đã học và tìm hiểu thêm các kiến thức mới kết hợp tra cứu các tài liệu chuyên ngành. Nhưng đề tài “Quản lý nhà hàng – C#” do hạn chế về thời gian, khả năng và kinh nghiệm nên đề tài chỉ hoàn thiện ở những chức năng cơ bản và không thể tránh khỏi những thiếu sót.

Hướng nghiên cứu và phát triển

- Hoàn thiện và sửa các lỗi hiện có của chương trình.
- Hoàn thiện các chức năng như: Quên mật khẩu, Nhập hàng, Kiểm tra tồn kho, Hóa đơn, Phiếu nhập, Giảm giá... giúp người dùng có thể trải nghiệm chương trình tốt hơn.
- Xây dựng Website giúp người dùng dễ tương tác hơn.
- Xây dựng chức năng giúp người dùng đặt bàn online.
- ...

TÀI LIỆU THAM KHẢO

- [1] <http://stackoverflow.com>
- [2] <https://documentation.devexpress.com>
- [3] <https://bunifuframework.com>
- [4] <https://www.youtube.com>
- [5] <https://www.codeproject.com>