**TRƯỜNG ĐẠI HỌC AN GIANG**

**KHOA CÔNG NGHỆ THÔNG TIN**

**A yellow circle with a green triangle and white text

AI-generated content may be incorrect.**

**BÁO CÁO ĐỒ ÁN**

**Chuyên đề lập trình.Net**

**XÂY DỰNG ỨNG DỤNG QUẢN LÝ XE VÀ LÁI XE**

Giảng viên hướng dẫn: Ths. Nguyễn Ngọc Minh

Sinh viên thực hiện:

DTH235675.Nguyễn Tuấn Khanh.DH24TH2\_Nhóm TH2\_Tổ TH1

DTH235717.Bùi Thành Nhơn.DH24TH2\_Nhóm TH2\_Tổ TH1

**AN GIANG, 11 - 2025**

# **LỜI NÓI ĐẦU**

Trong nền kinh tế thị trường hội nhập và phát triển như hiện nay, hoạt động vận tải đóng vai trò là "huyết mạch" then chốt, kết nối mọi mắt xích của chuỗi cung ứng, từ sản xuất, phân phối đến tiêu dùng. Sự vận hành trơn tru, hiệu quả và an toàn của hệ thống này phụ thuộc trực tiếp vào hai yếu" tố cốt lõi và không thể tách rời: phương tiện (xe) và con người (lái xe). Chính vì vậy, chủ đề "Quản lý xe và lái xe" không chỉ là một nghiệp vụ hậu cần đơn thuần mà đã trở thành một bài toán quản trị chiến lược, ảnh hưởng trực tiếp đến chi phí, hiệu suất cạnh tranh và sự phát triển bền vững của mọi tổ chức, doanh nghiệp. Trên thực tế, công tác quản lý này bao gồm hai mảng nghiệp vụ phức tạp và liên quan mật thiết đến nhau. Về phương tiện (Quản lý xe), đây là tài sản có giá trị lớn, đòi hỏi một quy trình quản lý chặt chẽ từ khi mua sắm, đăng ký, đăng kiểm, bảo trì, bảo dưỡng định kỳ, cấp phát nhiên liệu, giám sát hành trình, cho đến khi thanh lý. Việc quản lý xe hiệu quả giúp tối ưu hóa chi phí vận hành (nhiên liệu, sửa chữa), kéo dài tuổi thọ phương tiện và đảm bảo xe luôn ở trạng thái kỹ thuật tốt nhất, sẵn sàng thực hiện nhiệm vụ. Về con người (Quản lý lái xe), đây là yếu tố then chốt quyết định đến sự an toàn của mỗi hành trình và hình ảnh của tổ chức. Lái xe là đối tượng lao động đặc thù, thường xuyên làm việc độc lập bên ngoài, chịu nhiều áp lực về thời gian, sức khỏe và phải đối mặt với vô số tình huống rủi ro trên đường. Do đó, công tác quản lý lái xe đòi hỏi sự toàn diện từ tuyển dụng, đào tạo nghiệp vụ, phổ biến pháp luật an toàn giao thông, giám sát ý thức tuân thủ, kiểm tra sức khỏe, đến việc xây dựng các chế độ đãi ngộ hợp lý để giữ chân nhân tài.

**MỤC LỤC**

[LỜI NÓI ĐẦU I](#_Toc215001261)

[MỤC LỤC II](#_Toc215001263)

[DANH SÁCH HÌNH ẢNH III](#_Toc215001262)

[CHƯƠNG 1. MỤC ĐÍCH YÊU CẦU ĐỒ ÁN 4](#_Toc215001263)

[**1.1 Đặt vấn đề 4**](#_Toc215001264)

[**1.2 Mục tiêu đồ án 4**](#_Toc215001273)

[**1.3 Yêu cầu của hệ thống 5**](#_Toc215001274)

[1.3.1 Yêu cầu chức năng 5](#_Toc215001275)

[1.3.2 Yêu cầu phi chức năng 6](#_Toc215001276)

[​CHƯƠNG 2. PHÂN TÍCH THIẾT KẾ HỆ THỐNG 7](#_Toc215001277)

[**2.1 Phân tích hệ thống 7**](#_Toc215001278)

[2.1.1 Xác định các tác nhân 7](#_Toc215001279)

[**2.2 Thiết kế cơ sở dữ liệu 7**](#_Toc215001284)

[**2.3 Xây dựng giao diện người dùng 10**](#_Toc215001285)

[CHƯƠNG 3: GIAO DIỆN CHƯƠNG TRÌNH 11](#_Toc215001286)

[**3.1 Form đăng nhập 11**](#_Toc215001287)

[**3.2 Form giao diện menu quản lý xe và lái xe 11**](#_Toc215001288)

[**3.3 Hướng dẫn cài đặt và code ứng dụng 15**](#_Toc215001289)

[3.3.1 Chuẩn bị cơ sở dữ liệu 15](#_Toc215001290)

[3.3.2 Viết code ứng dụng giao diện 17](#_Toc215001291)

[TÀI LIỆU THAM KHẢO 107](#_Toc215001292)

# 

# **DANH SÁCH HÌNH ẢNH**

[Hình 1: Bảng bảo dưỡng 7](#_Toc215001263)

[Hình 2: Bảng lịch đăng kiểm 7](#_Toc215001263)

[Hình 3: Bảng phân công 8](#_Toc215001263)

[Hình 4: Bảng tài xế 8](#_Toc215001263)

[Hình 5: Bảng vi phạm 8](#_Toc215001263)

[Hình 6: Bảng bảo xe 9](#_Toc215001263)

[Hình 7: Sơ đồ ERD 9](#_Toc215001263)

[Hình 8: Giao diện đăng nhập 11](#_Toc215001263)

[Hình 9: Giao diện menu quản lý 12](#_Toc215001263)

[Hình 10: Giao diện quản lý xe 12](#_Toc215001263)

[Hình 11: Giao diện quản lý tài xế 13](#_Toc215001263)

[Hình 12: Giao diện quản lý phân công 13](#_Toc215001263)

[Hình 13: Giao diện bảo dưỡng xe 14](#_Toc215001263)

[Hình 14: Giao diện vi phạm sự cố 14](#_Toc215001263)

[Hình 15: Giao diện lịch đăng kiểm 15](#_Toc215001263)

# **CHƯƠNG 1. MỤC ĐÍCH YÊU CẦU ĐỒ ÁN**

## **1.1 Đặt vấn đề**

# ​- Ngày nay, cùng với sự phát triển của kinh tế, nhu cầu vận tải, giao nhận hàng hóa và di chuyển của các doanh nghiệp ngày càng tăng cao. Hầu hết các doanh nghiệp (từ sản xuất, thương mại dịch vụ đến logistics) đều sở hữu hoặc thuê một đội xe nhất định để phục vụ hoạt động kinh doanh.

# ​- Tuy nhiên, thực trạng quản lý đội xe và tài xế tại nhiều doanh nghiệp, đặc biệt là các doanh nghiệp vừa và nhỏ, vẫn còn rất thủ công:

# + Quản lý bằng sổ sách, Excel: Mọi thông tin về xe (lý lịch, bảo dưỡng, sửa chữa, đăng kiểm) và lái xe (bằng lái, hợp đồng, lịch trình) được ghi chép trên giấy tờ hoặc các file Excel phân tán.

# ​ + Sai sót và thất thoát dữ liệu: Việc nhập liệu thủ công dễ dẫn đến nhầm lẫn, sai sót. Dữ liệu không nhất quán, khó tra cứu và dễ bị mất mát.

# ​ + Khó khăn trong điều phối: Khi cần điều xe, người quản lý (điều phối viên) mất rất nhiều thời gian để xác định xe nào đang rảnh, lái xe nào sẵn sàng, xe nào phù hợp với loại hàng hóa... Điều này thường dẫn đến việc điều phối không tối ưu, xe chạy rỗng hoặc xe/lái xe phải chờ đợi.

# ​ + Không kiểm soát được chi phí: Các chi phí vận hành như nhiên liệu, cầu đường, sửa chữa... thường chỉ được ghi nhận thủ công qua hóa đơn, chứng từ. Rất khó để thống kê chính xác chi phí cho từng xe, chi phí trên mỗi km, hay phát hiện các bất thường (ví dụ: xe hỏng vặt nhiều, xe tiêu thụ nhiên liệu cao bất thường).

# ​ + Bị động trong quản lý pháp lý: Việc theo dõi thời hạn đăng kiểm, bảo hiểm của hàng chục chiếc xe hay thời hạn bằng lái của tài xế rất dễ bị bỏ sót, dẫn đến rủi ro pháp lý và phạt vi phạm.

# ​ + Mất thời gian làm báo cáo: Cuối tháng, bộ phận quản lý và kế toán tốn rất nhiều công sức để tổng hợp, đối chiếu sổ sách và làm báo cáo thống kê hiệu suất, chi phí.

## **1.2 Mục tiêu đồ án**

​- Mục tiêu tổng quát: Tin học hóa và tập trung hóa toàn bộ quy trình nghiệp vụ liên quan đến quản lý, vận hành đội xe và tài xế của một tổ chức.

- ​Mục tiêu cụ thể:

​ + Quản lý minh bạch: Cung cấp một cơ sở dữ liệu (CSDL) tập trung để lưu trữ toàn bộ hồ sơ, lý lịch của phương tiện (thông số kỹ thuật, giấy tờ pháp lý, lịch sử sửa chữa) và lái xe (thông tin cá nhân, bằng lái, lịch sử vi phạm).

​ + Tối ưu hóa điều phối: Hỗ trợ điều phối viên ra quyết định nhanh chóng và chính xác bằng cách cung cấp thông tin về trạng thái (sẵn sàng, đang chạy, bảo dưỡng) của cả xe và lái xe.

​ + Kiểm soát chi phí: Ghi nhận, theo dõi và thống kê chi tiết mọi chi phí phát sinh (nhiên liệu, bảo dưỡng, sửa chữa, cầu đường...). Tính toán tự động các chỉ số quan trọng (ví dụ: chi phí/km, mức tiêu hao nhiên liệu trung bình).

​ + Nâng cao tính tuân thủ & An toàn: Tự động cảnh báo các thời hạn quan trọng sắp hết hạn (đăng kiểm, bảo hiểm, bằng lái) và hỗ trợ ghi nhận các sự cố, vi phạm để có biện pháp xử lý kịp thời.

​ + Tự động hóa báo cáo: Cung cấp các báo cáo thống kê đa chiều (theo xe, theo lái xe, theo thời gian) một cách nhanh chóng, trực quan (dạng bảng, biểu đồ), giúp lãnh đạo nắm bắt tình hình và ra quyết định.

## **1.3 Yêu cầu của hệ thống**

### **1.3.1 Yêu cầu chức năng**

Hệ thống phải cung cấp các nhóm chức năng chính (modules) sau:

- Module Quản lý Lái xe:

+ Thêm/Sửa/Xóa hồ sơ lái xe (họ tên, SĐT, địa chỉ, file scan CCCD, hợp đồng...).

+ Quản lý thông tin bằng lái (hạng, số bằng, ngày cấp, ngày hết hạn).

+ Theo dõi và tự động cảnh báo khi bằng lái sắp hết hạn (ví dụ: trước 30 ngày).

+ Cập nhật trạng thái lái xe (sẵn sàng, đang chạy, nghỉ phép).

- Module Quản lý Xe (Phương tiện):

+ Thêm/Sửa/Xóa hồ sơ xe (biển số, loại xe, hãng, tải trọng, định mức nhiên liệu, file scan giấy tờ xe...).

+ Quản lý thông tin pháp lý (ngày đăng kiểm, ngày hết hạn bảo hiểm TNDS).

+ Theo dõi và tự động cảnh báo khi giấy tờ xe sắp hết hạn.

+ Cập nhật trạng thái xe (sẵn sàng, đang chạy, đang bảo dưỡng, hỏng hóc)

- ​Module Quản lý Vận hành (Điều phối):

+ Tạo "Lệnh điều xe" ( chuyến đi mới): Ghi nhận điểm đi, điểm đến, thời gian, loại hàng...

+ Phân công xe và lái xe cho Lệnh điều xe (Hệ thống chỉ hiển thị danh sách xe/lái xe "sẵn sàng" và "phù hợp" với yêu cầu).

+ Theo dõi trạng thái chuyến đi (chưa bắt đầu, đang chạy, đã hoàn thành, đã hủy).

+ In Lệnh điều xe (phiếu lộ trình).

### **1.3.2 Yêu cầu phi chức năng**

- Giao diện người dùng (UI/UX): Giao diện phải thân thiện, rõ ràng, logic. Các thao tác nghiệp vụ chính (như điều xe, nhập chi phí) phải dễ dàng thực hiện. (Nếu là Web App thì phải có thiết kế responsive để xem được trên điện thoại).

- ​Hiệu suất (Performance): Hệ thống phải phản hồi nhanh, thời gian tải trang và truy vấn dữ liệu (ví dụ: tìm kiếm xe) phải dưới 2 giây.

- Bảo mật (Security): Phải có cơ chế đăng nhập, mật khẩu phải được mã hóa. Phân quyền chặt chẽ, đảm bảo người dùng chỉ thấy và thao tác được các chức năng trong phạm vi của mình.

- ​Tính sẵn sàng (Availability): Hệ thống phải hoạt động ổn định 24/7 (nếu triển khai online), hạn chế tối đa thời gian chết (downtime).

- Khả năng mở rộng (Scalability): Thiết kế CSDL và kiến trúc phần mềm phải cho phép dễ dàng thêm mới các chức năng hoặc mở rộng để quản lý số lượng xe/lái xe lớn hơn trong tương lai.

# ​**CHƯƠNG 2. PHÂN TÍCH THIẾT KẾ HỆ THỐNG**

## **2.1 Phân tích hệ thống**

### **2.1.1 Xác định các tác nhân**

### - Quản trị viên (Admin): Là người có quyền cao nhất, chịu trách nhiệm cấu hình hệ thống, quản lý tài khoản và phân quyền cho người dùng khác.

### ​- Quản lý / Điều phối viên (Manager/Dispatcher): Là người dùng chính, thực hiện các nghiệp vụ hằng ngày như quản lý hồ sơ xe, lái xe, tạo lệnh điều xe, theo dõi bảo dưỡng và xem báo cáo.

### ​- Nhân viên Kế toán (Accountant): Là người dùng chịu trách nhiệm về mảng chi phí, nhập liệu chi phí, duyệt chi và xem các báo cáo tài chính liên quan đến vận hành xe.

### - Lái xe (Driver): Nếu hệ thống có Mobile App, lái xe là một tác nhân. Họ có thể xem lịch trình được phân công, cập nhật trạng thái chuyến đi (bắt đầu/kết thúc), và tự nhập liệu chi phí (như đổ xăng) qua app.

## **2.2 Thiết kế cơ sở dữ liệu**

- Bảng bảo dưỡng: Lưu thông tin về bảo dưỡng.

A screen shot of a computer code

AI-generated content may be incorrect.

# *Hình 1: Bảng bảo dưỡng*

- Bảng đăng kiểm: Lưu thông tin đăng kiểm.

A screen shot of a computer

AI-generated content may be incorrect.

# *Hình 2: Bảng đăng kiểm*

- Bảng Phân công: Lưu thông tin những xe đã được phân công.

A screen shot of a computer

AI-generated content may be incorrect.

Hình 3: Bảng phân công

- Bảng tài xế: Lưu thông tin của tài xế.

A screen shot of a computer

AI-generated content may be incorrect.

Hình 4: Bảng tài xế

- Bảng vi phạm: Lưu thông các xe vi phạm.

A computer screen with text

AI-generated content may be incorrect.

Hình 5: Bảng vi phạm

- Bảng xe: Lưu thông tin xe và tình trạng của xe.

A screen shot of a computer code

AI-generated content may be incorrect.

Hình 6: Bảng xe

- Sơ đồ ERM

**A computer screen shot of a computer

AI-generated content may be incorrect.**

# 

*Hình 7: Sơ đồ ERD*

## **2.3 Xây dựng giao diện người dùng**

- Các thành phần của giao diện:

Menu chuyển trang ( cho phép người dùng di chuyển giữa các trang khác nhau):

- Quản lý xe

- Quản lý tài xế

- Quản lý phân công

- Quản lý vi phạm

- Bảo dưỡng

- Đăng kiểm

- Các nút chức năng: Thêm, sửa, xóa, lưu, hủy, thoát, reset.

- Bảng hiển thị dữ liệu: Hiển thị danh sách tài xế, xe, phân công, vi phạm, lịch đăng kiểm, bảo dưỡng được lấy từ cơ sở dữ liệu. Khi người dùng nhấn chọn 1 nút chức năng bất kì, dữ liệu sẽ được điền vào các ô nhập để chỉnh sửa hoặc xóa.

# **CHƯƠNG 3: GIAO DIỆN CHƯƠNG TRÌNH**

## **3.1 Form đăng nhập**

**-** Đăng nhập tài khoản:Người dùng cần nhập Tên đăng nhập (Username) và Mật khẩu (Password). Hệ thống sẽ tiến hành xác thực dữ liệu này với Cơ sở dữ liệu. Sau khi đăng nhập thành công, hệ thống sẽ căn cứ vào vai trò của tài khoản để hiển thị các chức năng và quyền thao tác tương ứng.

A screenshot of a computer

AI-generated content may be incorrect.

*Hình 8: Giao diện đăng nhập*

## **3.2 Form giao diện menu quản lý xe và lái xe**

- Trang chủ menu chính: Hiển thị các thông tin tổng quan cho người quản lý, ví dụ: số xe/lái xe sẵn sàng, các cảnh báo (xe sắp hết hạn đăng kiểm, lái xe sắp hết hạn bằng lái).

A screenshot of a computer

AI-generated content may be incorrect.

*Hình 9: Giao diện menu quản lý*

*-* Danh sách quản lý xe: Hiển thị danh sách các xe trong hệ thống, cho phép tìm kiếm, lọc theo trạng thái và thực hiện các thao tác Thêm, Sửa, Xóa hồ sơ xe.

A screenshot of a computer

AI-generated content may be incorrect.

*Hình 10: Giao diện quản lý xe*

- Danh sách quản lý tài xế: Lưu trữ thông tin cá nhân cơ bản của từng tài xế, quản lý các yếu tố quan trọng liên quan đến nghiệp vụ lái xe như bằng lái và cung cấp cơ sở dữ liệu cho các chức năng liên quan khác như Phân công và Quản lý Vi phạm.

A screenshot of a computer

AI-generated content may be incorrect.

*Hình 11: Giao diện quản lý tài xế*

- Quản lý phân công: Dùng để điều phối viên, hiển thị danh sách các chuyến đi, trạng thái, ngày được phân công và thời địa điểm được phân công.

A screenshot of a computer

AI-generated content may be incorrect.

*Hình 12: Giao diện quản lý phân công*

- Bảo dưỡng xe: Để theo dõi toàn bộ lịch sử bảo trì, bảo dưỡng định kỳ và sửa chữa đột xuất của từng phương tiện. Điều này giúp đảm bảo xe luôn trong tình trạng hoạt động tốt nhất, kéo dài tuổi thọ phương tiện và quản lý chi phí vận hành một cách hiệu quả.

A screenshot of a computer

AI-generated content may be incorrect.

*Hình 13: Giao diện bảo dưỡng xe*

- Vi phạm sự cố: Cho phép người dùng nhập đầy đủ thông tin về một sự cố hoặc vi phạm mới, chỉnh sửa thông tin vi phạm đã có, và theo dõi trạng thái xử lý qua lưới dữ liệu.

A screenshot of a computer

AI-generated content may be incorrect.

*Hình 14: Giao diện vi phạm sự cố*

- Danh sách lịch đăng kiểm: Được thiết kế để theo dõi và cảnh báo các mốc thời gian pháp lý quan trọng của phương tiện, đảm bảo xe luôn tuân thủ quy định pháp luật.

A screenshot of a computer

AI-generated content may be incorrect.

Hình 15: Giao diện lịch đăng kiểm

## **3.3 Hướng dẫn cài đặt và code ứng dụng**

### **3.3.1 Chuẩn bị cơ sở dữ liệu**

– Trong cửa sổ  Solution Explorer, nháy phải chuột lên tên ứng dụng, chọn Add ->New Item… Chọn Data -> Service-based Database (hoặc SQL Database trong Visual Studio Net 2005).

– Đặt tên cơ sở dữ liệu: QLXVLX.mdf

– Tạo các bảng: Trong cửa sổ Server Explorer, chọn cơ sở dữ liệu, nháy phải lên Tables, chọn Add New Table…

A screen shot of a computer code

AI-generated content may be incorrect.

A screen shot of a computer

AI-generated content may be incorrect.

**A screen shot of a computer

AI-generated content may be incorrect.**

A screen shot of a computer

AI-generated content may be incorrect.

A computer screen with text

AI-generated content may be incorrect.

A screen shot of a computer code

AI-generated content may be incorrect.

### **3.3.2 Viết code ứng dụng giao diện**

**- Form đăng nhập**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace QLXevaLaiXe

{

public partial class DangNhap : Form

{

public DangNhap()

{

InitializeComponent();

txtMatKhau.PasswordChar = '\*';

this.AcceptButton = btnDangNhap;

}

private void label2\_Click(object sender, EventArgs e)

{

}

private void DangNhap\_Load(object sender, EventArgs e)

{

}

private void btnThoat\_Click(object sender, EventArgs e)

{

Application.Exit();

}

private void btnDangNhap\_Click(object sender, EventArgs e)

{

string tenDangNhap = txtTaiKhoan.Text;

string matKhau = txtMatKhau.Text;

if (KiemTraDangNhap(tenDangNhap, matKhau))

{

MenuChinh frmMain = new MenuChinh();

frmMain.FormClosed += (s, args) => this.Close();

frmMain.Show();

this.Hide();

}

else

{

MessageBox.Show("Sai tên đăng nhập hoặc mật khẩu!", "Lỗi Đăng nhập", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

private bool KiemTraDangNhap(string user, string pass)

{

if ((user == "Bùi Thành Nhơn" && pass == "12345") || (user == "NTK" && pass == "12345"))

{

return true;

}

return false;

}

**- Form lịch đăng kiểm**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Data.SqlClient;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace QLXevaLaiXe

{

public partial class LichDangKiemcs : Form

{

SqlConnection conn = new SqlConnection(@"Data Source=(LocalDB)\MSSQLLocalDB;AttachDbFilename=C:\Users\ADMIN\Downloads\QLXevaLaiXe\QLXevaLaiXe\QLXVLX.mdf;Integrated Security=True");

SqlDataAdapter da;

DataTable dt;

// Giả định tên DataGridView là dgrdsDangKy

void Load\_data()

{

// Lấy dữ liệu từ bảng DangKy

da = new SqlDataAdapter("select \* from DangKiem", conn);

dt = new DataTable();

da.Fill(dt);

dgrDanhSachDangKiem.DataSource = dt;

}

// Hàm tải danh sách Mã xe từ bảng Xe

private void LoadMaXe()

{

// Giả định tên ComboBox là cboMaXe

cboMaXe.Items.Clear();

string sql = "SELECT maxe FROM Xe";

SqlCommand cmd = new SqlCommand(sql, conn);

SqlDataReader reader = null;

try

{

conn.Open();

reader = cmd.ExecuteReader();

while (reader.Read())

{

// Thêm từng Mã xe vào ComboBox

cboMaXe.Items.Add(reader["maxe"].ToString());

}

}

catch (Exception ex)

{

MessageBox.Show("Lỗi khi tải danh sách Mã xe: " + ex.Message, "Lỗi", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

finally

{

if (reader != null)

{

reader.Close();

}

if (conn.State == ConnectionState.Open)

{

conn.Close();

}

}

}

private string currentAction = "DEFAULT";

// Cập nhật controls cho bảng DangKy

private void SetInputControlsEnabled(bool isEnabled)

{

txtMaDangKy.Enabled = isEnabled; // madk

cboMaXe.Enabled = isEnabled; // maxe

dtpNgayDangKiem.Enabled = isEnabled; // ngaydk

dtpNgayHetHan.Enabled = isEnabled; // ngayhethan

txtGhiChu.Enabled = isEnabled; // ghichu

}

private void SetButtonStates(bool isEditing)

{

// isEditing = TRUE khi đang ở trạng thái Thêm/Sửa (Lưu/Hủy được bật)

// isEditing = FALSE khi ở trạng thái Mặc định

btnThem.Enabled = !isEditing;

btnSua.Enabled = !isEditing;

btnXoa.Enabled = !isEditing;

btnTimKiem.Enabled = !isEditing;

btnLuu.Enabled = isEditing;

btnHuy.Enabled = isEditing;

btnReset.Enabled = !isEditing;

}

private void ClearInputFields()

{

txtMaDangKy.Clear();

cboMaXe.SelectedIndex = -1;

dtpNgayDangKiem.Value = DateTime.Now;

dtpNgayHetHan.Value = DateTime.Now.AddYears(1); // Mặc định là 1 năm

txtGhiChu.Clear();

}

public LichDangKiemcs()

{

InitializeComponent();

// Khóa controls và đặt trạng thái nút khi form khởi tạo

SetInputControlsEnabled(false);

SetButtonStates(false);

txtMaDangKy.Enabled = true;

}

private void LichDangKiemcs\_Load(object sender, EventArgs e)

{

Load\_data();

LoadMaXe();

}

private void cboXe\_SelectedIndexChanged(object sender, EventArgs e)

{

}

private void btnThem\_Click(object sender, EventArgs e)

{

currentAction = "THEM";

ClearInputFields();

SetInputControlsEnabled(true);

SetButtonStates(true);

txtMaDangKy.Focus();

}

private void btnSua\_Click(object sender, EventArgs e)

{

if (dgrDanhSachDangKiem.SelectedRows.Count == 0 && string.IsNullOrEmpty(txtMaDangKy.Text))

{

MessageBox.Show("Vui lòng chọn một bản ghi hoặc nhập Mã đăng ký để sửa.", "Thông báo", MessageBoxButtons.OK, MessageBoxIcon.Warning);

return;

}

currentAction = "SUA";

SetInputControlsEnabled(true);

SetButtonStates(true);

// Mã đăng ký (khóa chính) không cho sửa

txtMaDangKy.Enabled = false;

cboMaXe.Focus();

}

private void btnXoa\_Click(object sender, EventArgs e)

{

if (string.IsNullOrEmpty(txtMaDangKy.Text))

{

MessageBox.Show("Vui lòng chọn một bản ghi đăng ký để xóa.", "Thông báo", MessageBoxButtons.OK, MessageBoxIcon.Warning);

return;

}

if (MessageBox.Show("Bạn có chắc chắn muốn xóa bản ghi đăng ký " + txtMaDangKy.Text + " không?", "Xác nhận xóa", MessageBoxButtons.YesNo, MessageBoxIcon.Question) == DialogResult.No)

{

return;

}

string sql = "DELETE FROM DangKiem WHERE madk=@madk";

SqlCommand cmd = new SqlCommand(sql, conn);

cmd.Parameters.AddWithValue("@madk", txtMaDangKy.Text);

try

{

conn.Open();

int rowsAffected = cmd.ExecuteNonQuery();

if (rowsAffected > 0)

{

MessageBox.Show("Đã xóa bản ghi đăng ký thành công!", "Thông báo", MessageBoxButtons.OK, MessageBoxIcon.Information);

}

else

{

MessageBox.Show("Không tìm thấy Mã đăng ký để xóa.", "Lỗi", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

catch (Exception ex)

{

MessageBox.Show("Lỗi khi xóa bản ghi: " + ex.Message, "Lỗi", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

finally

{

conn.Close();

Load\_data();

ClearInputFields();

SetInputControlsEnabled(false);

SetButtonStates(false);

}

}

private void btnTimKiem\_Click(object sender, EventArgs e)

{

if (string.IsNullOrEmpty(txtMaDangKy.Text) && string.IsNullOrEmpty(cboMaXe.Text))

{

MessageBox.Show("Vui lòng nhập Mã đăng ký hoặc chọn Mã xe để tìm kiếm.", "Thông báo", MessageBoxButtons.OK, MessageBoxIcon.Warning);

return;

}

try

{

string sql = "SELECT \* FROM DangKiem WHERE madk LIKE @madk OR maxe LIKE @maxe";

SqlCommand cmd = new SqlCommand(sql, conn);

cmd.Parameters.AddWithValue("@madk", "%" + txtMaDangKy.Text + "%");

cmd.Parameters.AddWithValue("@maxe", "%" + cboMaXe.Text + "%"); // Tìm kiếm theo ComboBox

SqlDataAdapter daTK = new SqlDataAdapter(cmd);

DataTable dtTK = new DataTable();

daTK.Fill(dtTK);

dgrDanhSachDangKiem.DataSource = dtTK;

if (dtTK.Rows.Count == 0)

{

MessageBox.Show("Không tìm thấy bản ghi nào khớp với tiêu chí tìm kiếm.", "Kết quả tìm kiếm", MessageBoxButtons.OK, MessageBoxIcon.Information);

}

}

catch (Exception ex)

{

MessageBox.Show("Lỗi khi tìm kiếm: " + ex.Message, "Lỗi", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

finally

{

currentAction = "DEFAULT";

SetInputControlsEnabled(false);

SetButtonStates(false);

txtMaDangKy.Enabled=true;

}

}

private void btnReset\_Click(object sender, EventArgs e)

{

Load\_data();

LoadMaXe();

ClearInputFields();

currentAction = "DEFAULT";

SetInputControlsEnabled(false);

SetButtonStates(false);

txtMaDangKy.Enabled = true;

}

private void btnThoat\_Click(object sender, EventArgs e)

{

if (MessageBox.Show("Bạn có chắc muốn thoát không?", "Xác nhận",

MessageBoxButtons.YesNo, MessageBoxIcon.Question) == DialogResult.Yes)

{

this.Close();

}

}

private void btnHuy\_Click(object sender, EventArgs e)

{

ClearInputFields();

currentAction = "DEFAULT";

SetInputControlsEnabled(false);

SetButtonStates(false);

txtMaDangKy.Enabled = true;

Load\_data();

}

private void btnLuu\_Click(object sender, EventArgs e)

{

if (string.IsNullOrEmpty(txtMaDangKy.Text) || string.IsNullOrEmpty(cboMaXe.Text))

{

MessageBox.Show("Mã đăng ký và Mã xe không được để trống.", "Lỗi nhập liệu", MessageBoxButtons.OK, MessageBoxIcon.Error);

return;

}

if (dtpNgayDangKiem.Value > dtpNgayHetHan.Value)

{

MessageBox.Show("Ngày hết hạn không được nhỏ hơn Ngày đăng ký.", "Lỗi nhập liệu", MessageBoxButtons.OK, MessageBoxIcon.Error);

return;

}

// 2. Chuẩn bị câu lệnh SQL

string sql = "";

if (currentAction == "THEM")

{

sql = "INSERT INTO DangKiem (madk, maxe, ngaydk, ngayhethan, ghichu) "

+ "VALUES (@madk, @maxe, @ngaydk, @ngayhethan, @ghichu)";

}

else if (currentAction == "SUA")

{

// Cập nhật tất cả các trường ngoại trừ madk

sql = "UPDATE DangKiem SET maxe = @maxe, ngaydk = @ngaydk, ngayhethan = @ngayhethan, " +

"ghichu = @ghichu WHERE madk = @madk";

}

else

{

MessageBox.Show("Không thể thực hiện Lưu. Vui lòng nhấn Thêm hoặc Sửa trước.", "Lỗi", MessageBoxButtons.OK, MessageBoxIcon.Warning);

return;

}

SqlCommand cmd = new SqlCommand(sql, conn);

// 3. Gán tham số

cmd.Parameters.AddWithValue("@madk", txtMaDangKy.Text);

cmd.Parameters.AddWithValue("@maxe", cboMaXe.Text);

cmd.Parameters.AddWithValue("@ngaydk", dtpNgayDangKiem.Value.Date);

cmd.Parameters.AddWithValue("@ngayhethan", dtpNgayHetHan.Value.Date);

// Cột ghichu là nvarchar nên cần đảm bảo dữ liệu Unicode được truyền đúng

cmd.Parameters.AddWithValue("@ghichu", txtGhiChu.Text);

// 4. Thực thi

try

{

conn.Open();

int rowsAffected = cmd.ExecuteNonQuery();

if (rowsAffected > 0)

{

if (currentAction == "THEM")

MessageBox.Show("Thêm bản ghi đăng ký thành công!", "Thông báo", MessageBoxButtons.OK, MessageBoxIcon.Information);

else if (currentAction == "SUA")

MessageBox.Show("Cập nhật thông tin đăng ký thành công!", "Thông báo", MessageBoxButtons.OK, MessageBoxIcon.Information);

}

else

{

MessageBox.Show("Không có bản ghi nào được thay đổi.", "Lỗi", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

catch (Exception ex)

{

MessageBox.Show("Lỗi khi lưu dữ liệu: " + ex.Message, "Lỗi", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

finally

{

conn.Close();

Load\_data();

// Trở về trạng thái mặc định

currentAction = "DEFAULT";

SetInputControlsEnabled(false);

SetButtonStates(false);

ClearInputFields();

}

}

private void dgrDanhSachDangKiem\_CellContentClick(object sender, DataGridViewCellEventArgs e)

{

int i = e.RowIndex;

if (i >= 0 && i < dgrDanhSachDangKiem.Rows.Count - 1)

{

// Giả định tên cột trong DataGridView khớp với tên cột trong CSDL

txtMaDangKy.Text = dgrDanhSachDangKiem.Rows[i].Cells["madk"].Value.ToString();

cboMaXe.Text = dgrDanhSachDangKiem.Rows[i].Cells["maxe"].Value.ToString(); // Lấy giá trị cho ComboBox

if (dgrDanhSachDangKiem.Rows[i].Cells["ngaydk"].Value != DBNull.Value)

dtpNgayDangKiem.Value = Convert.ToDateTime(dgrDanhSachDangKiem.Rows[i].Cells["ngaydk"].Value);

if (dgrDanhSachDangKiem.Rows[i].Cells["ngayhethan"].Value != DBNull.Value)

dtpNgayHetHan.Value = Convert.ToDateTime(dgrDanhSachDangKiem.Rows[i].Cells["ngayhethan"].Value);

txtGhiChu.Text = dgrDanhSachDangKiem.Rows[i].Cells["ghichu"].Value.ToString();

}

SetInputControlsEnabled(false);

SetButtonStates(false);

}

}

}

**- Form menu chính**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace QLXevaLaiXe

{

public partial class MenuChinh : Form

{

public MenuChinh()

{

InitializeComponent();

}

private void btnx\_Click(object sender, EventArgs e)

{

Xe f1 = new Xe();

f1.ShowDialog();

}

private void btntx\_Click(object sender, EventArgs e)

{

Taixe f2 = new Taixe();

f2.ShowDialog();

}

private void btnpc\_Click(object sender, EventArgs e)

{

Phancong f3 = new Phancong();

f3.ShowDialog();

}

private void btnThoat\_Click(object sender, EventArgs e)

{

Application.Exit();

}

private void MenuChinh\_Load(object sender, EventArgs e)

{

}

private void button2\_Click(object sender, EventArgs e)

{

QuanLyViPham f1 = new QuanLyViPham();

f1.ShowDialog();

}

private void button3\_Click(object sender, EventArgs e)

{

QuanLyBaoDuong f2 = new QuanLyBaoDuong();

f2.ShowDialog();

}

private void Dangkiem\_Click(object sender, EventArgs e)

{

LichDangKiemcs f3 = new LichDangKiemcs();

f3.ShowDialog();

}

}

}

**- Form phân công**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using System.Data.SqlClient;

namespace QLXevaLaiXe

{

public partial class Phancong : Form

{

string connectionString = "Data Source=(LocalDB)\\MSSQLLocalDB;AttachDbFilename=C:\\Users\\ADMIN\\Downloads\\QLXevaLaiXe\\QLXevaLaiXe\\QLXVLX.mdf;Integrated Security=True";

SqlConnection conn;

SqlDataAdapter da;

DataTable dt;

private string currentAction = "DEFAULT";

private void SetButtonStates(bool isEditing)

{

// isEditing = TRUE khi đang ở trạng thái Thêm/Sửa (Lưu/Hủy được bật)

// isEditing = FALSE khi ở trạng thái Mặc định (Thêm/Sửa/Xóa được bật)

btnThem.Enabled = !isEditing;

btnSua.Enabled = !isEditing;

btnXoa.Enabled = !isEditing;

btnTimKiem.Enabled = !isEditing;

btnLuu.Enabled = isEditing;

btnHuy.Enabled = isEditing;

// btnReset thường chỉ có ý nghĩa khi đang nhập, nhưng ở đây ta dùng btnHuy

btnReset.Enabled = !isEditing;

}

private void ClearInputFields()

{

txtMaPhanCong.Clear();

txtMaXe.Clear();

txtMaSo.Clear(); // maso (Mã tài xế)

dtpNgayPhanCong.Value = DateTime.Now; // ngayphancong

txtDiaDiemDen.Clear();

txtGioDi.Clear();

txtGioDen.Clear();

// Xử lý trạng thái chuyến (trangthaichuyen)

rdoDaDi.Checked = false;

rdoChuaDi.Checked = false;

}

private void SetInputControlsEnabled(bool isEnabled)

{

txtMaPhanCong.Enabled = isEnabled; // mapc

txtMaXe.Enabled = isEnabled; // maxe

txtMaSo.Enabled = isEnabled; // maso (Mã tài xế)

dtpNgayPhanCong.Enabled = isEnabled; // ngayphancong

rdoDaDi.Enabled = isEnabled; // trangthaichuyen

rdoChuaDi.Enabled = isEnabled; // trangthaichuyen

txtDiaDiemDen.Enabled = isEnabled; // diadiemden

txtGioDi.Enabled = isEnabled; // giodi

txtGioDen.Enabled = isEnabled; // gioden

}

public Phancong()

{

InitializeComponent();

conn = new SqlConnection(connectionString);

SetInputControlsEnabled(false);

SetButtonStates(false);

txtMaSo.Enabled = true;

txtMaPhanCong.Enabled = true;

txtMaXe.Enabled = true;

}

// ==================== HÀM LOAD DỮ LIỆU ====================

private void LoadData()

{

try

{

da = new SqlDataAdapter("SELECT \* FROM Phancong", conn);

dt = new DataTable();

da.Fill(dt);

dgrdanhsach.DataSource = dt;

}

catch (Exception ex)

{

MessageBox.Show("Lỗi khi tải dữ liệu: " + ex.Message);

}

}

// ==================== HÀM XÓA TRẮNG Ô NHẬP ====================

private void ClearInputs()

{

txtMaSo.Clear();

txtMaPhanCong.Clear();

txtMaXe.Clear();

txtDiaDiemDen.Clear();

txtGioDi.Clear();

txtGioDen.Clear();

rdoChuaDi.Checked = false;

rdoDaDi.Checked = false;

dtpNgayPhanCong.Value = DateTime.Now;

}

// ==================== LOAD FORM ====================

private void Phancong\_Load(object sender, EventArgs e)

{

LoadData();

}

// ==================== NÚT THÊM ====================

private void btnThem\_Click(object sender, EventArgs e)

{

// 1.Chuyển sang trạng thái Thêm

currentAction = "THEM";

// 2. Xóa trắng các trường nhập liệu

ClearInputFields();

// 3. Bật các trường nhập liệu

SetInputControlsEnabled(true);

// 4. Bật nút Lưu/Hủy và Tắt các nút khác

SetButtonStates(true);

// 5. Đặt con trỏ vào Mã phân công

txtMaPhanCong.Focus();

// Đảm bảo tất cả các trường chính được phép nhập

txtMaPhanCong.Enabled = true;

txtMaSo.Enabled = true;

txtMaXe.Enabled = true;

}

// ==================== NÚT SỬA ====================

private void btnSua\_Click(object sender, EventArgs e)

{

if (dgrdanhsach.SelectedRows.Count == 0 && txtMaPhanCong.Text == "")

{

// Cập nhật thông báo

MessageBox.Show("Vui lòng chọn một bản \*\*Phân công\*\* từ danh sách hoặc nhập Mã \*\*Phân công (mapc)\*\* để sửa.", "Thông báo", MessageBoxButtons.OK, MessageBoxIcon.Warning);

return;

}

// 2. Chuyển sang trạng thái chỉnh sửa

currentAction = "SUA";

SetInputControlsEnabled(true);

// Gọi hàm thiết lập trạng thái nút (nếu có)

SetButtonStates(true);

txtMaPhanCong.Enabled = false;

// Nếu không cho phép sửa:

txtMaSo.Enabled = false;

txtMaXe.Enabled = false;

txtDiaDiemDen.Focus();

}

// ==================== NÚT XÓA ====================

private void btnXoa\_Click(object sender, EventArgs e)

{

if (string.IsNullOrWhiteSpace(txtMaSo.Text))

{

MessageBox.Show("Vui lòng nhập Mã số phân công cần xóa!");

return;

}

if (MessageBox.Show("Bạn có chắc muốn xóa phân công này?",

"Xác nhận xóa", MessageBoxButtons.YesNo, MessageBoxIcon.Warning) == DialogResult.Yes)

{

try

{

string sql = "DELETE FROM Phancong WHERE maso = @maso";

SqlCommand cmd = new SqlCommand(sql, conn);

cmd.Parameters.AddWithValue("@maso", txtMaSo.Text.Trim());

conn.Open();

cmd.ExecuteNonQuery();

conn.Close();

LoadData();

ClearInputs();

MessageBox.Show("🗑️ Xóa phân công thành công!");

}

catch (Exception ex)

{

MessageBox.Show("❌ Lỗi khi xóa: " + ex.Message);

if (conn.State == ConnectionState.Open) conn.Close();

}

}

}

// ==================== NÚT TÌM KIẾM ====================

private void btnTimKiem\_Click(object sender, EventArgs e)

{

try

{

string keyword = txtMaSo.Text.Trim();

if (string.IsNullOrWhiteSpace(keyword))

{

MessageBox.Show("Vui lòng nhập Mã số hoặc Mã phân công để tìm!");

return;

}

string sql = @"SELECT \* FROM Phancong

WHERE maso LIKE @keyword OR maphancong LIKE @keyword OR diadiemden LIKE @keyword";

SqlDataAdapter da = new SqlDataAdapter(sql, conn);

da.SelectCommand.Parameters.AddWithValue("@keyword", "%" + keyword + "%");

DataTable dtSearch = new DataTable();

da.Fill(dtSearch);

dgrdanhsach.DataSource = dtSearch;

if (dtSearch.Rows.Count == 0)

MessageBox.Show("Không tìm thấy phân công nào phù hợp.");

}

catch (Exception ex)

{

MessageBox.Show("❌ Lỗi khi tìm kiếm: " + ex.Message);

}

finally

{

// 1. Đảm bảo trạng thái form không phải là THÊM/SỬA

currentAction = "DEFAULT";

SetButtonStates(false);

SetInputControlsEnabled(false); // Khóa tất cả các controls khác

// 2. Mở lại ô Mã xe để tiếp tục nhập từ khóa tìm kiếm

txtMaSo.Enabled = true;

txtMaPhanCong.Enabled = true;

txtMaXe.Enabled = true;

// 3. Đặt con trỏ chuột vào ô Mã xe

txtMaSo.Focus();

// 4. Xóa trống ô Biển số để chuẩn bị cho tìm kiếm mới

txtMaXe.Clear();

txtMaPhanCong.Clear();

}

}

// ==================== NÚT RESET, HỦY, THOÁT ====================

private void btnReset\_Click(object sender, EventArgs e)

{

LoadData();

ClearInputFields();

currentAction = "DEFAULT";

SetInputControlsEnabled(false);

SetButtonStates(false);

txtMaSo.Enabled = true;

txtMaPhanCong.Enabled = true;

txtMaXe.Enabled = true;

}

private void btnHuy\_Click(object sender, EventArgs e)

{

// Giữ nguyên: Xóa trắng các trường

ClearInputFields();

// Bổ sung: Trở về trạng thái mặc định

currentAction = "DEFAULT";

SetInputControlsEnabled(false);

SetButtonStates(false);

txtMaSo.Enabled = true;

txtMaPhanCong.Enabled = true;

txtMaXe.Enabled = true;

}

private void btnThoat\_Click(object sender, EventArgs e)

{

if (MessageBox.Show("Bạn có chắc muốn thoát không?", "Xác nhận",

MessageBoxButtons.YesNo, MessageBoxIcon.Question) == DialogResult.Yes)

{

this.Close();

}

}

private void dgrdstaixe\_CellContentClick(object sender, DataGridViewCellEventArgs e)

{

// Đảm bảo có ít nhất một hàng được chọn trong DataGridView

// int i = e.RowIndex; // Biến 'i' đã được khai báo sẵn từ sự kiện DataGridView

int i = dgrdanhsach.CurrentRow.Index;

if (i >= 0)

{

// Lấy dữ liệu từ dòng có chỉ mục 'i' của lưới dgrPhancong (hoặc dgrdanhsach)

// 1. Gán các trường Text

txtMaPhanCong.Text = dgrdanhsach.Rows[i].Cells["mapc"].Value.ToString();

txtMaXe.Text = dgrdanhsach.Rows[i].Cells["maxe"].Value.ToString();

txtMaSo.Text = dgrdanhsach.Rows[i].Cells["maso"].Value.ToString(); // maso (Mã tài xế)

txtDiaDiemDen.Text = dgrdanhsach.Rows[i].Cells["diadiemden"].Value.ToString();

txtGioDi.Text = dgrdanhsach.Rows[i].Cells["giodi"].Value.ToString();

txtGioDen.Text = dgrdanhsach.Rows[i].Cells["gioden"].Value.ToString();

// 2. Xử lý trường Ngày Phân công (DateTimePicker)

if (dgrdanhsach.Rows[i].Cells["ngayphancong"].Value != DBNull.Value)

{

// Chuyển đổi giá trị thành DateTime

dtpNgayPhanCong.Value = DateTime.Parse(dgrdanhsach.Rows[i].Cells["ngayphancong"].Value.ToString());

}

// 3. Xử lý trường Trạng Thái Chuyến (Radio Button)

// Tên cột: trangthaichuyen

string trangthai = dgrdanhsach.Rows[i].Cells["trangthaichuyen"].Value.ToString().Trim().ToLower();

// Dùng .ToLower() để so sánh không phân biệt chữ hoa/thường

// Kiểm tra giá trị Trạng Thái được lưu trong CSDL

if (trangthai == "đã đi")

{

rdoDaDi.Checked = true;

rdoChuaDi.Checked = false;

}

else // Giả định trường hợp còn lại là "chưa đi"

{

rdoDaDi.Checked = false;

rdoChuaDi.Checked = true;

}

}

currentAction = "DEFAULT";

SetInputControlsEnabled(false);

SetButtonStates(false);

}

private void button1\_Click(object sender, EventArgs e)

{

if (string.IsNullOrWhiteSpace(txtMaPhanCong.Text) || string.IsNullOrWhiteSpace(txtMaSo.Text) || string.IsNullOrWhiteSpace(txtMaXe.Text) || string.IsNullOrWhiteSpace(txtGioDi.Text))

{

MessageBox.Show("Mã phân công, Mã tài xế, Mã xe và Giờ đi không được để trống!", "Lỗi nhập liệu", MessageBoxButtons.OK, MessageBoxIcon.Error);

return;

}

string sql = "";

string successMessage = "";

if (currentAction == "THEM")

{

// Câu lệnh INSERT

sql = @"INSERT INTO Phancong (mapc, maso, maxe, ngayphancong, diadiemden, giodi, gioden, trangthaichuyen)

VALUES (@mapc, @maso, @maxe, @ngayphancong, @diadiemden, @giodi, @gioden, @trangthaichuyen)";

successMessage = "✅ Thêm phân công thành công!";

}

else if (currentAction == "SUA")

{

// Câu lệnh UPDATE

// \*\*Lưu ý: Tôi dùng mapc (Mã phân công) làm WHERE để UPDATE, vì mapc thường là PK.\*\*

// Nếu bạn dùng maso, logic UPDATE sẽ bị sai nếu 1 maso có nhiều mapc.

sql = @"UPDATE Phancong

SET maso=@maso, maxe=@maxe, ngayphancong=@ngayphancong,

diadiemden=@diadiemden, giodi=@giodi, gioden=@gioden, trangthaichuyen=@trangthaichuyen

WHERE mapc=@mapc";

successMessage = "✅ Cập nhật phân công thành công!";

}

else

{

MessageBox.Show("Vui lòng nhấn Thêm hoặc Sửa trước khi Lưu.", "Cảnh báo", MessageBoxButtons.OK, MessageBoxIcon.Warning);

return;

}

try

{

SqlCommand cmd = new SqlCommand(sql, conn);

// Gán tham số

cmd.Parameters.AddWithValue("@mapc", txtMaPhanCong.Text.Trim());

cmd.Parameters.AddWithValue("@maso", txtMaSo.Text.Trim());

cmd.Parameters.AddWithValue("@maxe", txtMaXe.Text.Trim());

cmd.Parameters.AddWithValue("@ngayphancong", dtpNgayPhanCong.Value.Date);

cmd.Parameters.AddWithValue("@diadiemden", txtDiaDiemDen.Text.Trim());

cmd.Parameters.AddWithValue("@giodi", txtGioDi.Text.Trim());

cmd.Parameters.AddWithValue("@gioden", txtGioDen.Text.Trim());

cmd.Parameters.AddWithValue("@trangthaichuyen", rdoDaDi.Checked ? "Đã đi" : "Chưa đi");

conn.Open();

int rowsAffected = cmd.ExecuteNonQuery();

conn.Close();

if (rowsAffected > 0)

{

MessageBox.Show(successMessage);

}

else

{

MessageBox.Show("Không có bản ghi nào được thay đổi. Vui lòng kiểm tra Mã phân công.", "Lỗi", MessageBoxButtons.OK, MessageBoxIcon.Warning);

}

}

catch (Exception ex)

{

MessageBox.Show("❌ Lỗi khi lưu dữ liệu: " + ex.Message);

}

finally

{

if (conn.State == ConnectionState.Open) conn.Close();

LoadData();

currentAction = "DEFAULT";

SetInputControlsEnabled(false); // Khóa controls

SetButtonStates(false); // Bật nút Thêm/Sửa/Xóa

ClearInputFields(); // Xóa trắng trường nhập liệu

}

}

}

}

**- Form quản lý bảo dưỡng**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Data.SqlClient;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace QLXevaLaiXe

{

public partial class QuanLyBaoDuong : Form

{

SqlConnection conn = new SqlConnection(@"Data Source=(LocalDB)\MSSQLLocalDB;AttachDbFilename=C:\Users\ADMIN\Downloads\QLXevaLaiXe\QLXevaLaiXe\QLXVLX.mdf;Integrated Security=True");

SqlDataAdapter da;

DataTable dt;

// Giả định tên DataGridView là dgrdsbaoduong

void Load\_data()

{

// Lấy dữ liệu từ bảng Baoduong

da = new SqlDataAdapter("select \* from Baoduong", conn);

dt = new DataTable();

da.Fill(dt);

dgrbd.DataSource = dt;

}

private string currentAction = "DEFAULT";

private void SetInputControlsEnabled(bool isEnabled)

{

txtMaBaoTri.Enabled = isEnabled; // Ma bao tri (mabt)

cboMaXe.Enabled = isEnabled; // Ma xe (maxe)

cboLoaiDichVu.Enabled = isEnabled; // Dich vu (dichvu)

dtpNgayBatDau.Enabled = isEnabled; // Ngay bat dau (ngaybd)

dtpNgayKetThuc.Enabled = isEnabled; // Ngay ket thuc (ngaykt)

txtChiPhi.Enabled = isEnabled; // Chi phi (chiphi)

txtGhiChu.Enabled = isEnabled; // Ghi chu (ghichu)

}

private void SetButtonStates(bool isEditing)

{

// isEditing = TRUE khi đang ở trạng thái Thêm/Sửa (Lưu/Hủy được bật)

// isEditing = FALSE khi ở trạng thái Mặc định (Thêm/Sửa/Xóa được bật)

btnThem.Enabled = !isEditing;

btnSua.Enabled = !isEditing;

btnXoa.Enabled = !isEditing;

btnTimKiem.Enabled = !isEditing;

btnLuu.Enabled = isEditing;

btnHuy.Enabled = isEditing;

btnReset.Enabled = !isEditing;

}

private void ClearInputFields()

{

txtMaBaoTri.Clear();

cboLoaiDichVu.SelectedIndex = -1;

cboMaXe.SelectedIndex = -1;

dtpNgayBatDau.Value = DateTime.Now;

dtpNgayKetThuc.Value = DateTime.Now;

txtChiPhi.Clear();

txtGhiChu.Clear();

}

private void LoadMaXe()

{

// Giả định tên ComboBox mới là cboMaXe

cboMaXe.Items.Clear();

string sql = "SELECT maxe FROM Xe";

SqlCommand cmd = new SqlCommand(sql, conn);

SqlDataReader reader = null;

try

{

conn.Open();

reader = cmd.ExecuteReader();

while (reader.Read())

{

// Thêm từng Mã xe vào ComboBox

cboMaXe.Items.Add(reader["maxe"].ToString());

}

}

catch (Exception ex)

{

MessageBox.Show("Lỗi khi tải danh sách Mã xe: " + ex.Message, "Lỗi", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

finally

{

if (reader != null)

{

reader.Close();

}

if (conn.State == ConnectionState.Open)

{

conn.Close();

}

}

}

public QuanLyBaoDuong()

{

InitializeComponent();

SetInputControlsEnabled(false);

SetButtonStates(false);

txtMaBaoTri.Enabled = true;

}

private void dgvBaoDuong\_CellContentClick(object sender, DataGridViewCellEventArgs e)

{

int i = e.RowIndex;

if (i >= 0 && i < dgrbd.Rows.Count - 1) // Trừ hàng trống cuối cùng

{

// Giả định tên cột trong DataGridView khớp với tên cột trong CSDL

txtMaBaoTri.Text = dgrbd.Rows[i].Cells["mabt"].Value.ToString();

// Cập nhật: Gán giá trị Mã xe từ DB vào ComboBox

cboMaXe.Text = dgrbd.Rows[i].Cells["maxe"].Value.ToString();

cboLoaiDichVu.Text = dgrbd.Rows[i].Cells["loaidichvu"].Value.ToString();

// Chuyển đổi ngày tháng

if (dgrbd.Rows[i].Cells["ngaybatdau"].Value != DBNull.Value)

dtpNgayBatDau.Value = Convert.ToDateTime(dgrbd.Rows[i].Cells["ngaybatdau"].Value);

if (dgrbd.Rows[i].Cells["ngayketthuc"].Value != DBNull.Value)

dtpNgayKetThuc.Value = Convert.ToDateTime(dgrbd.Rows[i].Cells["ngayketthuc"].Value);

txtChiPhi.Text = dgrbd.Rows[i].Cells["chiphi"].Value.ToString();

txtGhiChu.Text = dgrbd.Rows[i].Cells["ghichu"].Value.ToString();

}

// Khóa controls sau khi click để người dùng phải bấm Sửa nếu muốn thay đổi

SetInputControlsEnabled(false);

SetButtonStates(false);

}

private void QuanLyBaoDuong\_Load(object sender, EventArgs e)

{

Load\_data();

LoadMaXe();

// Khởi tạo ComboBox LoaiDichVu (ví dụ)

cboLoaiDichVu.Items.Clear();

cboLoaiDichVu.Items.Add("Bảo dưỡng định kỳ");

cboLoaiDichVu.Items.Add("Sửa chữa động cơ");

cboLoaiDichVu.Items.Add("Thay lốp");

cboLoaiDichVu.Items.Add("Kiểm tra phanh");

cboLoaiDichVu.Items.Add("Bảo hiểm");

}

private void btnThem\_Click(object sender, EventArgs e)

{

currentAction = "THEM";

ClearInputFields();

SetInputControlsEnabled(true);

SetButtonStates(true);

txtMaBaoTri.Focus();

}

private void btnSua\_Click(object sender, EventArgs e)

{

if (dgrbd.SelectedRows.Count == 0 && txtMaBaoTri.Text == "")

{

MessageBox.Show("Vui lòng chọn một bản bảo dưỡng từ danh sách hoặc nhập Mã bảo trì để sửa.", "Thông báo", MessageBoxButtons.OK, MessageBoxIcon.Warning);

return;

}

currentAction = "SUA";

SetInputControlsEnabled(true);

SetButtonStates(true);

// Mã bảo trì thường là khóa chính, không cho sửa

txtMaBaoTri.Enabled = false;

}

private void btnXoa\_Click(object sender, EventArgs e)

{

if (string.IsNullOrEmpty(txtMaBaoTri.Text))

{

MessageBox.Show("Vui lòng chọn một bản ghi bảo dưỡng để xóa.", "Thông báo", MessageBoxButtons.OK, MessageBoxIcon.Warning);

return;

}

if (MessageBox.Show("Bạn có chắc chắn muốn xóa bản ghi bảo dưỡng " + txtMaBaoTri.Text + " không?", "Xác nhận xóa", MessageBoxButtons.YesNo, MessageBoxIcon.Question) == DialogResult.No)

{

return;

}

string sql = "DELETE FROM BaoDuong WHERE mabt=@mabt";

SqlCommand cmd = new SqlCommand(sql, conn);

cmd.Parameters.AddWithValue("@mabt", txtMaBaoTri.Text);

try

{

conn.Open();

int rowsAffected = cmd.ExecuteNonQuery();

if (rowsAffected > 0)

{

MessageBox.Show("Đã xóa bản ghi bảo dưỡng thành công!", "Thông báo", MessageBoxButtons.OK, MessageBoxIcon.Information);

}

else

{

MessageBox.Show("Không tìm thấy Mã bảo trì để xóa.", "Lỗi", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

catch (Exception ex)

{

MessageBox.Show("Lỗi khi xóa bản ghi: " + ex.Message, "Lỗi", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

finally

{

conn.Close();

Load\_data();

ClearInputFields();

}

}

private void btnTimKiem\_Click(object sender, EventArgs e)

{

if (string.IsNullOrEmpty(txtMaBaoTri.Text) && string.IsNullOrEmpty(cboMaXe.Text))

{

MessageBox.Show("Vui lòng nhập Mã bảo trì để tìm kiếm.", "Thông báo", MessageBoxButtons.OK, MessageBoxIcon.Warning);

return;

}

try

{

string sql = "SELECT \* FROM BaoDuong WHERE mabt LIKE @mabt";

SqlCommand cmd = new SqlCommand(sql, conn);

cmd.Parameters.AddWithValue("@mabt", "%" + txtMaBaoTri.Text + "%");

SqlDataAdapter daTK = new SqlDataAdapter(cmd);

DataTable dtTK = new DataTable();

daTK.Fill(dtTK);

dgrbd.DataSource = dtTK;

if (dtTK.Rows.Count == 0)

{

MessageBox.Show("Không tìm thấy bản ghi bảo dưỡng nào với Mã bảo trì đã nhập.", "Kết quả tìm kiếm", MessageBoxButtons.OK, MessageBoxIcon.Information);

}

}

catch (Exception ex)

{

MessageBox.Show("Lỗi khi tìm kiếm: " + ex.Message, "Lỗi", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

finally

{

currentAction = "DEFAULT";

SetInputControlsEnabled(false);

SetButtonStates(false);

txtMaBaoTri.Enabled = true;

}

}

private void btnLuu\_Click(object sender, EventArgs e)

{

// 1. Kiểm tra ràng buộc

if (string.IsNullOrEmpty(txtMaBaoTri.Text) || string.IsNullOrEmpty(cboMaXe.Text) || string.IsNullOrEmpty(cboLoaiDichVu.Text)) // Đã đổi txtMaXeBD thành cboMaXe

{

MessageBox.Show("Mã bảo trì, Mã xe và Loại dịch vụ không được để trống.", "Lỗi nhập liệu", MessageBoxButtons.OK, MessageBoxIcon.Error);

return;

}

// Kiểm tra ngày tháng

if (dtpNgayBatDau.Value > dtpNgayKetThuc.Value)

{

MessageBox.Show("Ngày kết thúc không được nhỏ hơn Ngày bắt đầu.", "Lỗi nhập liệu", MessageBoxButtons.OK, MessageBoxIcon.Error);

return;

}

// Kiểm tra Chi phí phải là số

decimal chiPhi;

if (!decimal.TryParse(txtChiPhi.Text, out chiPhi))

{

MessageBox.Show("Chi phí phải là số.", "Lỗi nhập liệu", MessageBoxButtons.OK, MessageBoxIcon.Error);

return;

}

// 2. Chuẩn bị câu lệnh SQL

string sql = "";

if (currentAction == "THEM")

{

sql = "INSERT INTO BaoDuong (mabt, maxe, loaidichvu, ngaybatdau, ngayketthuc, chiphi, ghichu) "

+ "VALUES (@mabt, @maxe, @loaidichvu, @ngaybatdau, @ngayketthuc, @chiphi, @ghichu)";

}

else if (currentAction == "SUA")

{

sql = "UPDATE BaoDuong SET maxe = @maxe, loaidichvu = @loaidichvu, ngaybatdau = @ngaybatdau, " +

"ngayketthuc = @ngayketthuc, chiphi = @chiphi, ghichu = @ghichu WHERE mabt = @mabt";

}

else

{

MessageBox.Show("Không thể thực hiện Lưu. Vui lòng nhấn Thêm hoặc Sửa trước.", "Lỗi", MessageBoxButtons.OK, MessageBoxIcon.Warning);

return;

}

SqlCommand cmd = new SqlCommand(sql, conn);

// 3. Gán tham số

cmd.Parameters.AddWithValue("@mabt", txtMaBaoTri.Text);

cmd.Parameters.AddWithValue("@maxe", cboMaXe.Text);

cmd.Parameters.AddWithValue("@loaidichvu", cboLoaiDichVu.Text);

cmd.Parameters.AddWithValue("@ngaybatdau", dtpNgayBatDau.Value.Date); // Lấy phần ngày

cmd.Parameters.AddWithValue("@ngayketthuc", dtpNgayKetThuc.Value.Date); // Lấy phần ngày

cmd.Parameters.AddWithValue("@chiphi", chiPhi); // Sử dụng giá trị đã parse

cmd.Parameters.AddWithValue("@ghichu", txtGhiChu.Text);

// 4. Thực thi

try

{

conn.Open();

int rowsAffected = cmd.ExecuteNonQuery();

if (rowsAffected > 0)

{

if (currentAction == "THEM")

MessageBox.Show("Thêm bản ghi bảo dưỡng thành công!", "Thông báo", MessageBoxButtons.OK, MessageBoxIcon.Information);

else if (currentAction == "SUA")

MessageBox.Show("Cập nhật thông tin bảo dưỡng thành công!", "Thông báo", MessageBoxButtons.OK, MessageBoxIcon.Information);

}

else

{

MessageBox.Show("Không có bản ghi nào được thay đổi. Kiểm tra Mã bảo trì.", "Lỗi", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

catch (Exception ex)

{

MessageBox.Show("Lỗi khi lưu dữ liệu: " + ex.Message, "Lỗi", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

finally

{

conn.Close();

Load\_data();

// Trở về trạng thái mặc định sau khi lưu thành công

currentAction = "DEFAULT";

SetInputControlsEnabled(false);

SetButtonStates(false);

ClearInputFields();

}

}

private void btnHuy\_Click(object sender, EventArgs e)

{

ClearInputFields();

currentAction = "DEFAULT";

SetInputControlsEnabled(false);

SetButtonStates(false);

txtMaBaoTri.Enabled = true;

Load\_data(); // Tải lại dữ liệu gốc

}

private void btnReset\_Click(object sender, EventArgs e)

{

Load\_data();

ClearInputFields();

currentAction = "DEFAULT";

SetInputControlsEnabled(false);

SetButtonStates(false);

txtMaBaoTri.Enabled = true;

}

private void btnThoat\_Click(object sender, EventArgs e)

{

if (MessageBox.Show("Bạn có chắc muốn thoát không?", "Xác nhận",

MessageBoxButtons.YesNo, MessageBoxIcon.Question) == DialogResult.Yes)

{

this.Close();

}

}

}

}

**- Form Quản lý vi phạm**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Data.SqlClient;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace QLXevaLaiXe

{

public partial class QuanLyViPham : Form

{

SqlConnection conn = new SqlConnection(@"Data Source=(LocalDB)\MSSQLLocalDB;AttachDbFilename=C:\Users\ADMIN\Downloads\QLXevaLaiXe\QLXevaLaiXe\QLXVLX.mdf;Integrated Security=True");

SqlDataAdapter da;

DataTable dt;

private string currentAction = "DEFAULT";

public QuanLyViPham()

{

InitializeComponent();

SetInputControlsEnabled(false);

SetButtonStates(false);

txtMaViPham.Enabled = true;

}

private void LoadData()

{

try

{

da = new SqlDataAdapter("SELECT \* FROM Vipham", conn);

dt = new DataTable();

da.Fill(dt);

// Giả định tên DataGridView là dgrdsVipham

dgrViPham.DataSource = dt;

}

catch (Exception ex)

{

MessageBox.Show("Lỗi khi tải dữ liệu: " + ex.Message);

}

}

private void LoadMaSoTaiXe()

{

cboMaTaiXe.Items.Clear();

string sql = "SELECT maso FROM Taixe";

LoadComboBoxData(sql, cboMaTaiXe, "maso");

}

private void LoadMaXe()

{

cboMaXe.Items.Clear();

string sql = "SELECT maxe FROM Xe";

LoadComboBoxData(sql, cboMaXe, "maxe");

}

private void LoadLoaiViPham()

{

// Các loại vi phạm có thể là giá trị cố định hoặc lấy từ bảng ThamSo/CauHinh nếu có

cboLoaiViPham.Items.Clear();

cboLoaiViPham.Items.Add("Vượt đèn đỏ");

cboLoaiViPham.Items.Add("Chạy quá tốc độ");

cboLoaiViPham.Items.Add("Đi sai làn");

cboLoaiViPham.Items.Add("Không đội mũ bảo hiểm");

// cboLoaiViPham.Items.Add("Vi phạm khác");

}

// Hàm hỗ trợ tải dữ liệu chung từ DB vào ComboBox

private void LoadComboBoxData(string sql, ComboBox cbo, string columnName)

{

SqlDataReader reader = null;

try

{

SqlCommand cmd = new SqlCommand(sql, conn);

if (conn.State != ConnectionState.Open) conn.Open();

reader = cmd.ExecuteReader();

while (reader.Read())

{

cbo.Items.Add(reader[columnName].ToString());

}

}

catch (Exception ex)

{

MessageBox.Show($"Lỗi khi tải dữ liệu cho {columnName}: " + ex.Message, "Lỗi", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

finally

{

if (reader != null) reader.Close();

if (conn.State == ConnectionState.Open) conn.Close();

}

}

// ==========================================================

// QUẢN LÝ GIAO DIỆN

// ==========================================================

private void SetInputControlsEnabled(bool isEnabled)

{

txtMaViPham.Enabled = isEnabled;

cboMaTaiXe.Enabled = isEnabled;

cboMaXe.Enabled = isEnabled;

cboLoaiViPham.Enabled = isEnabled;

dtpNgayXayRa.Enabled = isEnabled;

cboTrangThai.Enabled = isEnabled;

txtMucPhat.Enabled = isEnabled;

}

private void SetButtonStates(bool isEditing)

{

btnThem.Enabled = !isEditing;

btnSua.Enabled = !isEditing;

btnXoa.Enabled = !isEditing;

btnTimKiem.Enabled = !isEditing;

btnLuu.Enabled = isEditing;

btnHuy.Enabled = isEditing;

btnReset.Enabled = !isEditing;

}

private void ClearInputFields()

{

txtMaViPham.Clear();

cboMaTaiXe.SelectedIndex = -1;

cboMaXe.SelectedIndex = -1;

cboLoaiViPham.SelectedIndex = -1;

dtpNgayXayRa.Value = DateTime.Now;

cboTrangThai.SelectedIndex = -1;

txtMucPhat.Clear();

}

private void QuanLyViPham\_Load(object sender, EventArgs e)

{

LoadData();

// Tải danh sách cho các ComboBox

LoadMaSoTaiXe();

LoadMaXe();

LoadLoaiViPham();

// Khởi tạo ComboBox Trạng thái

cboTrangThai.Items.Clear();

cboTrangThai.Items.Add("Chưa xử lý");

cboTrangThai.Items.Add("Đã xử lý");

// Khóa controls khi load form

SetInputControlsEnabled(false);

SetButtonStates(false);

txtMaViPham.Enabled = true;

}

// =========================================

// 3. THÊM

// =========================================

private void btnThem\_Click(object sender, EventArgs e)

{

currentAction = "THEM";

ClearInputFields();

SetInputControlsEnabled(true);

SetButtonStates(true);

txtMaViPham.Focus();

txtMaViPham.Enabled = true;

}

// =========================================

// 4. SỬA

// =========================================

private void btnSua\_Click(object sender, EventArgs e)

{

if (dgrViPham.SelectedRows.Count == 0 && string.IsNullOrEmpty(txtMaViPham.Text))

{

MessageBox.Show("Vui lòng chọn một bản ghi vi phạm hoặc nhập Mã vi phạm để sửa.", "Thông báo", MessageBoxButtons.OK, MessageBoxIcon.Warning);

return;

}

currentAction = "SUA";

SetInputControlsEnabled(true);

SetButtonStates(true);

// Mã vi phạm (khóa chính) không cho sửa

txtMaViPham.Enabled = false;

// Mã tài xế và Mã xe thường cũng không cho sửa

cboMaTaiXe.Enabled = false;

cboMaXe.Enabled = false;

cboLoaiViPham.Focus();

}

// =========================================

// 5. XÓA

// =========================================

private void btnXoa\_Click(object sender, EventArgs e)

{

if (string.IsNullOrWhiteSpace(txtMaViPham.Text))

{

MessageBox.Show("Vui lòng nhập Mã vi phạm cần xóa!");

return;

}

if (MessageBox.Show($"Bạn có chắc muốn xóa bản ghi vi phạm {txtMaViPham.Text}?",

"Xác nhận xóa", MessageBoxButtons.YesNo, MessageBoxIcon.Warning) == DialogResult.Yes)

{

string sql = "DELETE FROM Vipham WHERE mavipham = @mavipham";

SqlCommand cmd = new SqlCommand(sql, conn);

cmd.Parameters.AddWithValue("@mavipham", txtMaViPham.Text.Trim());

try

{

conn.Open();

int rowsAffected = cmd.ExecuteNonQuery();

if (rowsAffected > 0)

{

MessageBox.Show("🗑️ Xóa bản ghi vi phạm thành công!");

}

else

{

MessageBox.Show("Không tìm thấy Mã vi phạm để xóa.");

}

}

catch (Exception ex)

{

MessageBox.Show("❌ Lỗi khi xóa: " + ex.Message);

}

finally

{

if (conn.State == ConnectionState.Open) conn.Close();

LoadData();

ClearInputFields();

}

}

}

// =========================================

// 6. TÌM KIẾM

// =========================================

private void btnTimKiem\_Click(object sender, EventArgs e)

{

try

{

// Tìm kiếm theo Mã vi phạm, Mã tài xế hoặc Mã xe

string keyword = txtMaViPham.Text.Trim();

if (string.IsNullOrWhiteSpace(keyword))

{

MessageBox.Show("Vui lòng nhập Mã vi phạm để tìm!");

return;

}

string sql = @"SELECT \* FROM ViPham

WHERE mavipham LIKE @keyword";

SqlDataAdapter daSearch = new SqlDataAdapter(sql, conn);

daSearch.SelectCommand.Parameters.AddWithValue("@keyword", "%" + keyword + "%");

DataTable dtSearch = new DataTable();

daSearch.Fill(dtSearch);

dgrViPham.DataSource = dtSearch;

if (dtSearch.Rows.Count == 0)

MessageBox.Show("Không tìm thấy bản ghi vi phạm nào phù hợp.");

}

catch (Exception ex)

{

MessageBox.Show("❌ Lỗi khi tìm kiếm: " + ex.Message);

}

finally

{

currentAction = "DEFAULT";

SetInputControlsEnabled(false);

SetButtonStates(false);

txtMaViPham.Enabled = true;

}

}

// =========================================

// 7. CLICK DÒNG → ĐỔ DỮ LIỆU LÊN Ô NHẬP

// =========================================

private void dgvDanhSach\_CellClick(object sender, DataGridViewCellEventArgs e)

{

if (e.RowIndex < 0) return;

DataGridViewRow r = dgrViPham.Rows[e.RowIndex];

txtMaViPham.Text = r.Cells["maVP"].Value.ToString();

cboMaTaiXe.Text = r.Cells["maTX"].Value.ToString();

cboMaXe.Text = r.Cells["maXe"].Value.ToString();

cboLoaiViPham.Text = r.Cells["loaiVP"].Value.ToString();

dtpNgayXayRa.Value = Convert.ToDateTime(r.Cells["ngay"].Value);

cboTrangThai.Text = r.Cells["trangthai"].Value.ToString();

txtMucPhat.Text = r.Cells["mucphat"].Value.ToString();

}

// =========================================

// 8. RESET

// =========================================

private void btnReset\_Click(object sender, EventArgs e)

{

// Tải lại dữ liệu và reset trạng thái

LoadData();

ClearInputFields();

currentAction = "DEFAULT";

SetInputControlsEnabled(false);

SetButtonStates(false);

txtMaViPham.Enabled = true;

}

private void btnThoat\_Click(object sender, EventArgs e)

{

if (MessageBox.Show("Bạn có chắc muốn thoát không?", "Xác nhận",

MessageBoxButtons.YesNo, MessageBoxIcon.Question) == DialogResult.Yes)

{

this.Close();

}

}

private void dgrDanhSachViPham\_CellContentClick(object sender, DataGridViewCellEventArgs e)

{

int i = e.RowIndex;

if (i >= 0 && i < dgrViPham.Rows.Count)

{

txtMaViPham.Text = dgrViPham.Rows[i].Cells["mavipham"].Value.ToString();

cboMaTaiXe.Text = dgrViPham.Rows[i].Cells["maso"].Value.ToString();

cboMaXe.Text = dgrViPham.Rows[i].Cells["maxe"].Value.ToString();

cboLoaiViPham.Text = dgrViPham.Rows[i].Cells["loaivipham"].Value.ToString();

cboTrangThai.Text = dgrViPham.Rows[i].Cells["trangthai"].Value.ToString();

txtMucPhat.Text = dgrViPham.Rows[i].Cells["mucphat"].Value.ToString();

// Xử lý Ngày Xảy Ra (DateTimePicker)

if (dgrViPham.Rows[i].Cells["ngayxayra"].Value != DBNull.Value)

{

dtpNgayXayRa.Value = DateTime.Parse(dgrViPham.Rows[i].Cells["ngayxayra"].Value.ToString());

}

}

currentAction = "DEFAULT";

SetInputControlsEnabled(false);

SetButtonStates(false);

// Đảm bảo Mã vi phạm luôn bị vô hiệu hóa khi ở trạng thái Mặc định/Click

txtMaViPham.Enabled = false;

}

private void btnHuy\_Click(object sender, EventArgs e)

{

ClearInputFields();

currentAction = "DEFAULT";

SetInputControlsEnabled(false);

SetButtonStates(false);

txtMaViPham.Enabled = true;

LoadData(); // Tải lại dữ liệu gốc

}

private void btnLuu\_Click(object sender, EventArgs e)

{

// 1. KIỂM TRA RÀNG BUỘC

if (string.IsNullOrEmpty(txtMaViPham.Text) || string.IsNullOrEmpty(cboMaTaiXe.Text) || string.IsNullOrEmpty(cboMaXe.Text) || string.IsNullOrEmpty(cboLoaiViPham.Text))

{

MessageBox.Show("Mã vi phạm, Mã tài xế, Mã xe và Loại vi phạm không được để trống.", "Lỗi nhập liệu", MessageBoxButtons.OK, MessageBoxIcon.Error);

return;

}

if (!decimal.TryParse(txtMucPhat.Text, out \_))

{

MessageBox.Show("Mức phạt phải là số.", "Lỗi nhập liệu", MessageBoxButtons.OK, MessageBoxIcon.Error);

return;

}

// 2. CHUẨN BỊ SQL

string sql = "";

string successMessage = "";

if (currentAction == "THEM")

{

sql = @"INSERT INTO ViPham (mavipham, maso, maxe, loaivipham, ngayxayra, trangthai, mucphat)

VALUES (@mavipham, @maso, @maxe, @loaivipham, @ngayxayra, @trangthai, @mucphat)";

successMessage = "✅ Thêm bản ghi vi phạm thành công!";

}

else if (currentAction == "SUA")

{

// Sử dụng mavipham làm khóa chính để UPDATE

sql = @"UPDATE ViPham

SET loaivipham=@loaivipham, ngayxayra=@ngayxayra, trangthai=@trangthai, mucphat=@mucphat

WHERE mavipham=@mavipham";

successMessage = "✅ Cập nhật thông tin vi phạm thành công!";

}

else

{

MessageBox.Show("Vui lòng nhấn Thêm hoặc Sửa trước khi Lưu.", "Cảnh báo", MessageBoxButtons.OK, MessageBoxIcon.Warning);

return;

}

try

{

SqlCommand cmd = new SqlCommand(sql, conn);

// Gán tham số chung

cmd.Parameters.AddWithValue("@mavipham", txtMaViPham.Text.Trim());

cmd.Parameters.AddWithValue("@maso", cboMaTaiXe.Text.Trim());

cmd.Parameters.AddWithValue("@maxe", cboMaXe.Text.Trim());

cmd.Parameters.AddWithValue("@loaivipham", cboLoaiViPham.Text.Trim());

cmd.Parameters.AddWithValue("@ngayxayra", dtpNgayXayRa.Value.Date);

cmd.Parameters.AddWithValue("@trangthai", cboTrangThai.Text.Trim());

cmd.Parameters.AddWithValue("@mucphat", decimal.Parse(txtMucPhat.Text)); // Đảm bảo lưu đúng kiểu số

conn.Open();

int rowsAffected = cmd.ExecuteNonQuery();

if (rowsAffected > 0)

{

MessageBox.Show(successMessage);

}

else

{

MessageBox.Show("Không có bản ghi nào được thay đổi.", "Lỗi", MessageBoxButtons.OK, MessageBoxIcon.Warning);

}

}

catch (Exception ex)

{

MessageBox.Show("❌ Lỗi khi lưu dữ liệu: " + ex.Message, "Lỗi", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

finally

{

if (conn.State == ConnectionState.Open) conn.Close();

// Trở về trạng thái mặc định sau khi hoàn tất

LoadData();

currentAction = "DEFAULT";

SetInputControlsEnabled(false);

SetButtonStates(false);

ClearInputFields();

}

}

}

}

**- Form tài xế**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Data.SqlClient;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace QLXevaLaiXe

{

public partial class Taixe : Form

{

string connectionString = "Data Source=(LocalDB)\\MSSQLLocalDB;AttachDbFilename=C:\\Users\\ADMIN\\Downloads\\QLXevaLaiXe\\QLXevaLaiXe\\QLXVLX.mdf;Integrated Security=True";

SqlConnection conn;

SqlDataAdapter da;

DataTable dt;

private string currentAction = "DEFAULT";

private void SetButtonStates(bool isEditing)

{

// isEditing = TRUE khi đang ở trạng thái Thêm/Sửa (Lưu/Hủy được bật)

// isEditing = FALSE khi ở trạng thái Mặc định (Thêm/Sửa/Xóa được bật)

btnThem.Enabled = !isEditing;

btnSua.Enabled = !isEditing;

btnXoa.Enabled = !isEditing;

btnTimKiem.Enabled = !isEditing;

btnLuu.Enabled = isEditing;

btnHuy.Enabled = isEditing;

// btnReset thường chỉ có ý nghĩa khi đang nhập, nhưng ở đây ta dùng btnHuy

btnReset.Enabled = !isEditing;

}

private void ClearInputFields()

{

txtMaSo.Clear();

txtBanglai.Clear();

txtTen.Clear();

txtHoLot.Clear();

txtSDT.Clear();

radNam.Checked = false;

radNu.Checked = false;

dtpNgaySinh.Value = DateTime.Now;

}

private void SetInputControlsEnabled(bool isEnabled)

{

txtMaSo.Enabled = isEnabled;

txtBanglai.Enabled = isEnabled;

txtTen.Enabled = isEnabled;

txtHoLot.Enabled = isEnabled;

txtSDT.Enabled = isEnabled;

radNam.Enabled = isEnabled;

radNu.Enabled = isEnabled;

dtpNgaySinh.Enabled = isEnabled;

}

public Taixe()

{

InitializeComponent();

conn = new SqlConnection(connectionString);

SetInputControlsEnabled(false);

SetButtonStates(false);

txtMaSo.Enabled = true;

txtTen.Enabled = true;

txtHoLot.Enabled = true;

}

// Load dữ liệu chính

private void LoadData()

{

try

{

da = new SqlDataAdapter("SELECT \* FROM Taixe", conn);

dt = new DataTable();

da.Fill(dt);

dgrTaixe.DataSource = dt;

}

catch (Exception ex)

{

MessageBox.Show("Lỗi khi tải dữ liệu: " + ex.Message);

}

}

// Xóa trắng ô nhập

private void ClearInputs()

{

txtMaSo.Clear();

txtHoLot.Clear();

txtTen.Clear();

radNam.Checked = true;

dtpNgaySinh.Value = DateTime.Now;

txtSDT.Clear();

txtBanglai.Clear();

// Đảm bảo ô luôn mở

txtMaSo.ReadOnly = false;

txtHoLot.ReadOnly = false;

txtTen.ReadOnly = false;

txtSDT.ReadOnly = false;

txtBanglai.ReadOnly = false;

dtpNgaySinh.Enabled = true;

radNam.Enabled = true;

radNu.Enabled = true;

}

private void Taixe\_Load(object sender, EventArgs e)

{

LoadData();

}

// Vô hiệu hóa CellClick

private void dgrTaixe\_CellClick(object sender, DataGridViewCellEventArgs e)

{

// Không bắt buộc click để thao tác

}

// CÁC NÚT CHỨC NĂNG CHÍNH

// Nút THÊM

private void btnThem\_Click(object sender, EventArgs e)

{

currentAction = "THEM";

// 2. Xóa trắng các trường và bật nhập liệu

ClearInputFields();

SetInputControlsEnabled(true);

// 3. Bật nút Lưu/Hủy và tắt các nút khác

SetButtonStates(true);

// 4. Đặt con trỏ vào Mã số

txtMaSo.Focus();

// Đảm bảo Mã số được phép nhập khi thêm

txtMaSo.Enabled = true;

}

// Nút SỬA

private void btnSua\_Click(object sender, EventArgs e)

{

if (dgrTaixe.SelectedRows.Count == 0 && txtMaSo.Text == "")

{

MessageBox.Show("Vui lòng chọn một tài xế từ danh sách hoặc nhập Mã số để sửa.", "Thông báo", MessageBoxButtons.OK, MessageBoxIcon.Warning);

return;

}

// 2. Chuyển sang trạng thái chỉnh sửa

currentAction = "SUA";

SetInputControlsEnabled(true);

// Gọi hàm thiết lập trạng thái nút (nếu có)

SetButtonStates(true);

// Mã số (khóa chính) không cho sửa trong chế độ sửa.

txtMaSo.Enabled = false;

// Di chuyển con trỏ chuột đến trường đầu tiên có thể chỉnh sửa

txtHoLot.Focus();

}

// Nút XÓA

private void btnXoa\_Click(object sender, EventArgs e)

{

if (string.IsNullOrWhiteSpace(txtMaSo.Text))

{

MessageBox.Show("Vui lòng nhập Mã số tài xế cần xóa!");

return;

}

if (MessageBox.Show("Bạn có chắc muốn xóa tài xế này?",

"Xác nhận xóa", MessageBoxButtons.YesNo, MessageBoxIcon.Warning) == DialogResult.Yes)

{

try

{

string sql = "DELETE FROM Taixe WHERE maso = @maso";

SqlCommand cmd = new SqlCommand(sql, conn);

cmd.Parameters.AddWithValue("@maso", txtMaSo.Text.Trim());

conn.Open();

cmd.ExecuteNonQuery();

conn.Close();

LoadData();

ClearInputs();

MessageBox.Show("🗑️ Xóa tài xế thành công!");

}

catch (Exception ex)

{

MessageBox.Show("❌ Lỗi khi xóa: " + ex.Message);

if (conn.State == ConnectionState.Open) conn.Close();

}

}

}

// Nút TÌM

private void btnTim\_Click(object sender, EventArgs e)

{

try

{

string keyword = txtTen.Text.Trim(); // Có thể thay bằng txtMaSo.Text nếu bạn muốn tìm theo mã

if (string.IsNullOrWhiteSpace(keyword))

{

MessageBox.Show("Vui lòng nhập Tên hoặc Mã tài xế cần tìm!");

return;

}

string sql = @"SELECT \* FROM Taixe

WHERE ten LIKE @keyword OR maso LIKE @keyword OR holot LIKE @keyword";

SqlDataAdapter da = new SqlDataAdapter(sql, conn);

da.SelectCommand.Parameters.AddWithValue("@keyword", "%" + keyword + "%");

DataTable dtSearch = new DataTable();

da.Fill(dtSearch);

dgrTaixe.DataSource = dtSearch;

if (dtSearch.Rows.Count == 0)

{

MessageBox.Show("Không tìm thấy kết quả phù hợp.", "Thông báo", MessageBoxButtons.OK, MessageBoxIcon.Information);

}

}

catch (Exception ex)

{

MessageBox.Show("❌ Lỗi khi tìm kiếm: " + ex.Message);

}

finally

{

// 1. Đảm bảo trạng thái form không phải là THÊM/SỬA

currentAction = "DEFAULT";

SetButtonStates(false);

SetInputControlsEnabled(false); // Khóa tất cả các controls khác

// 2. Mở lại ô Mã xe để tiếp tục nhập từ khóa tìm kiếm

txtMaSo.Enabled = true;

txtTen.Enabled = true;

txtHoLot.Enabled = true;

// 3. Đặt con trỏ chuột vào ô Mã xe

txtMaSo.Focus();

// 4. Xóa trống ô Biển số để chuẩn bị cho tìm kiếm mới

txtHoLot.Clear();

txtTen.Clear();

}

}

// Nút THOÁT

private void btnThoat\_Click(object sender, EventArgs e)

{

if (MessageBox.Show("Bạn có chắc muốn thoát không?", "Xác nhận",

MessageBoxButtons.YesNo, MessageBoxIcon.Question) == DialogResult.Yes)

{

this.Close();

}

}

// NÚT HỦY

private void btnHuy\_Click(object sender, EventArgs e)

{

// Giữ nguyên: Xóa trắng các trường

ClearInputFields();

// Bổ sung: Trở về trạng thái mặc định

currentAction = "DEFAULT";

SetInputControlsEnabled(false);

SetButtonStates(false);

txtMaSo.Enabled = true;

txtTen.Enabled = true;

txtHoLot.Enabled = true;

}

// NÚT RESET

private void btnReset\_Click(object sender, EventArgs e)

{

LoadData();

ClearInputFields();

currentAction = "DEFAULT";

SetInputControlsEnabled(false);

SetButtonStates(false);

txtMaSo.Enabled = true;

txtTen.Enabled = true;

txtHoLot.Enabled = true;

}

private void button1\_Click(object sender, EventArgs e)

{

if (string.IsNullOrWhiteSpace(txtMaSo.Text) || string.IsNullOrWhiteSpace(txtTen.Text))

{

MessageBox.Show("Mã số và Tên tài xế không được để trống!", "Lỗi nhập liệu", MessageBoxButtons.OK, MessageBoxIcon.Error);

return;

}

string sql = "";

string successMessage = "";

if (currentAction == "THEM")

{

// Câu lệnh INSERT

sql = @"INSERT INTO Taixe (maso, holot, ten, phai, ngaysinh, sdt, banglai)

VALUES (@maso, @holot, @ten, @phai, @ngaysinh, @sdt, @banglai)";

successMessage = "✅ Thêm tài xế thành công!";

}

else if (currentAction == "SUA")

{

// Câu lệnh UPDATE

sql = @"UPDATE Taixe

SET holot = @holot, ten = @ten, phai = @phai,

ngaysinh = @ngaysinh, sdt = @sdt, banglai = @banglai

WHERE maso = @maso";

successMessage = "✅ Cập nhật thông tin tài xế thành công!";

}

else

{

MessageBox.Show("Không thể thực hiện Lưu. Vui lòng nhấn Thêm hoặc Sửa trước.", "Cảnh báo", MessageBoxButtons.OK, MessageBoxIcon.Warning);

return;

}

try

{

SqlCommand cmd = new SqlCommand(sql, conn);

// Gán tham số chung cho cả INSERT và UPDATE

cmd.Parameters.AddWithValue("@maso", txtMaSo.Text.Trim());

cmd.Parameters.AddWithValue("@holot", txtHoLot.Text.Trim());

cmd.Parameters.AddWithValue("@ten", txtTen.Text.Trim());

cmd.Parameters.AddWithValue("@phai", radNam.Checked ? "Nam" : "Nữ");

cmd.Parameters.AddWithValue("@ngaysinh", dtpNgaySinh.Value.Date);

cmd.Parameters.AddWithValue("@sdt", txtSDT.Text.Trim());

cmd.Parameters.AddWithValue("@banglai", txtBanglai.Text.Trim());

conn.Open();

int rowsAffected = cmd.ExecuteNonQuery();

conn.Close();

if (rowsAffected > 0)

{

MessageBox.Show(successMessage);

}

else

{

MessageBox.Show("Không có bản ghi nào được thay đổi. Vui lòng kiểm tra lại Mã số.", "Lỗi", MessageBoxButtons.OK, MessageBoxIcon.Warning);

}

}

catch (Exception ex)

{

MessageBox.Show("❌ Lỗi khi lưu dữ liệu: " + ex.Message);

}

finally

{

if (conn.State == ConnectionState.Open) conn.Close();

// Trở về trạng thái mặc định sau khi hoàn tất

LoadData();

currentAction = "DEFAULT";

SetInputControlsEnabled(false);

SetButtonStates(false);

ClearInputFields();

}

}

private void dgrTaixe\_CellContentClick(object sender, DataGridViewCellEventArgs e)

{

// 1. Kiểm tra xem có dòng nào được chọn không (tuyệt đối cần thiết)

int i = e.RowIndex;

if (i >= 0)

{

// Lấy dữ liệu từ dòng có chỉ mục 'i' của lưới dgrdtaixe

// 1. Gán các trường Text

txtMaSo.Text = dgrTaixe.Rows[i].Cells["maso"].Value.ToString();

txtHoLot.Text = dgrTaixe.Rows[i].Cells["holot"].Value.ToString();

txtTen.Text = dgrTaixe.Rows[i].Cells["ten"].Value.ToString();

txtSDT.Text = dgrTaixe.Rows[i].Cells["sdt"].Value.ToString();

txtBanglai.Text = dgrTaixe.Rows[i].Cells["banglai"].Value.ToString();

// 2. Xử lý trường Ngày Sinh (DateTimePicker)

if (dgrTaixe.Rows[i].Cells["ngaysinh"].Value != DBNull.Value)

{

dtpNgaySinh.Value = DateTime.Parse(dgrTaixe.Rows[i].Cells["ngaysinh"].Value.ToString());

}

// 3. Xử lý trường Phái (Radio Button)

string gioiTinh = dgrTaixe.Rows[i].Cells["phai"].Value.ToString().Trim().ToUpper();

// Kiểm tra giá trị Phái được lưu trong CSDL

if (gioiTinh == "NAM" || gioiTinh == "M")

{

radNam.Checked = true;

radNu.Checked = false;

}

else // Giả định là Nữ nếu không phải Nam (hoặc kiểm tra giá trị "NU", "F")

{

radNam.Checked = false;

radNu.Checked = true;

}

}

currentAction = "DEFAULT"; // Rất quan trọng!

SetInputControlsEnabled(false);

SetButtonStates(false);

txtMaSo.Enabled = false;

}

}

}

**- Form xe**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using System.Data.SqlClient;

namespace QLXevaLaiXe

{

public partial class Xe : Form

{

SqlConnection conn = new SqlConnection(@"Data Source=(LocalDB)\MSSQLLocalDB;AttachDbFilename=C:\Users\ADMIN\Downloads\QLXevaLaiXe\QLXevaLaiXe\QLXVLX.mdf;Integrated Security=True");

SqlDataAdapter da;

DataTable dt;

void Load\_data()

{

da= new SqlDataAdapter("select \* from Xe", conn);

dt= new DataTable();

da.Fill(dt);

dgrdsxe.DataSource = dt;

}

private string currentAction = "DEFAULT";

private void SetInputControlsEnabled(bool isEnabled)

{

txtMaXe.Enabled = isEnabled;

cboHangXe.Enabled = isEnabled;

txtBienSo.Enabled = isEnabled;

txtNamSX.Enabled = isEnabled;

cboTinhTrang.Enabled = isEnabled;

txtSoGhe.Enabled = isEnabled;

}

private void SetButtonStates(bool isEditing)

{

// isEditing = TRUE khi đang ở trạng thái Thêm/Sửa (Lưu/Hủy được bật)

// isEditing = FALSE khi ở trạng thái Mặc định (Thêm/Sửa/Xóa được bật)

btnThem.Enabled = !isEditing;

btnSua.Enabled = !isEditing;

btnXoa.Enabled = !isEditing;

btnTimKiem.Enabled = !isEditing;

btnLuu.Enabled = isEditing;

btnHuy.Enabled = isEditing;

// btnReset thường chỉ có ý nghĩa khi đang nhập, nhưng ở đây ta dùng btnHuy

btnReset.Enabled = !isEditing;

}

private void ClearInputFields()

{

txtMaXe.Clear();

cboHangXe.SelectedIndex = -1;

txtBienSo.Clear();

txtNamSX.Clear();

cboTinhTrang.SelectedIndex = -1;

txtSoGhe.Clear();

}

public Xe()

{

InitializeComponent();

SetInputControlsEnabled(false);

SetButtonStates(false);

txtMaXe.Enabled = true;

}

private void Xe\_Load(object sender, EventArgs e)

{

Load\_data();

cboTinhTrang.Items.Clear();

cboTinhTrang.Items.Add("Đang hoạt động");

cboTinhTrang.Items.Add("Cần bảo dưỡng");

cboTinhTrang.Items.Add("Hỏng");

cboTinhTrang.Items.Add("Đang bảo dưỡng");

cboHangXe.Items.Clear();

cboHangXe.Items.Add("Toyota");

cboHangXe.Items.Add("Ford");

cboHangXe.Items.Add("Hyundai");

cboHangXe.Items.Add("Kia");

cboHangXe.Items.Add("Mazda");

cboHangXe.Items.Add("Honda");

}

private void label3\_Click(object sender, EventArgs e)

{

}

private void btnThem\_Click(object sender, EventArgs e)

{

currentAction = "THEM";

// 2. Xóa trắng các trường nhập liệu

ClearInputFields();

// 3. Bật các trường nhập liệu

SetInputControlsEnabled(true);

// 4. Bật nút Lưu/Hủy và Tắt nút Thêm/Sửa/Xóa/Tìm kiếm

SetButtonStates(true);

// 5. Đặt con trỏ vào Mã xe

txtMaXe.Focus();

}

private void btnSua\_Click(object sender, EventArgs e)

{

if (dgrdsxe.SelectedRows.Count == 0 && txtMaXe.Text == "")

{

MessageBox.Show("Vui lòng chọn một xe từ danh sách hoặc nhập Mã xe để sửa.", "Thông báo", MessageBoxButtons.OK, MessageBoxIcon.Warning);

return;

}

// 2. Chuyển sang trạng thái chỉnh sửa

currentAction = "SUA";

SetInputControlsEnabled(true);

SetButtonStates(true);

// Mã xe thường là khóa chính, không cho sửa trong chế độ sửa.

txtMaXe.Enabled = false;

txtMaXe.Focus();

}

private void btnXoa\_Click(object sender, EventArgs e)

{

string sql = "DELETE FROM Xe WHERE maxe=@maxe";

SqlCommand cmd = new SqlCommand(sql, conn);

cmd.Parameters.AddWithValue("@maxe", txtMaXe.Text);

conn.Open();

cmd.ExecuteNonQuery();

conn.Close();

Load\_data();

MessageBox.Show("Đã xóa xe!");

}

private void btnTimKiem\_Click(object sender, EventArgs e)

{

if (string.IsNullOrWhiteSpace(txtMaXe.Text) && string.IsNullOrWhiteSpace(txtBienSo.Text))

{

MessageBox.Show("Vui lòng nhập Mã xe để tìm kiếm.", "Thông báo", MessageBoxButtons.OK, MessageBoxIcon.Warning);

return;

}

try

{

string sql = "SELECT \* FROM Xe WHERE maxe LIKE @maxe ";

SqlCommand cmd = new SqlCommand(sql, conn);

cmd.Parameters.AddWithValue("@maxe", "%" + txtMaXe.Text + "%");

SqlDataAdapter daTK = new SqlDataAdapter(cmd);

DataTable dtTK = new DataTable();

daTK.Fill(dtTK);

dgrdsxe.DataSource = dtTK;

// Thông báo nếu không tìm thấy

if (dtTK.Rows.Count == 0)

{

MessageBox.Show("Không tìm thấy kết quả phù hợp.", "Thông báo", MessageBoxButtons.OK, MessageBoxIcon.Information);

}

}

catch (Exception ex)

{

MessageBox.Show("Lỗi khi tìm kiếm: " + ex.Message, "Lỗi", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

finally

{

// 1. Đảm bảo trạng thái form không phải là THÊM/SỬA

currentAction = "DEFAULT";

SetButtonStates(false);

SetInputControlsEnabled(false); // Khóa tất cả các controls khác

// 2. Mở lại ô Mã xe để tiếp tục nhập từ khóa tìm kiếm

txtMaXe.Enabled = true;

// 3. Đặt con trỏ chuột vào ô Mã xe

txtMaXe.Focus();

// 4. Xóa trống ô Biển số để chuẩn bị cho tìm kiếm mới

txtBienSo.Clear();

}

}

private void btnThoat\_Click(object sender, EventArgs e)

{

if (MessageBox.Show("Bạn có chắc muốn thoát không?", "Xác nhận",

MessageBoxButtons.YesNo, MessageBoxIcon.Question) == DialogResult.Yes)

{

this.Close();

}

}

private void btnReset\_Click(object sender, EventArgs e)

{

Load\_data();

ClearInputFields();

currentAction = "DEFAULT";

SetInputControlsEnabled(false);

SetButtonStates(false);

txtMaXe.Enabled = true;

}

private void btnLuu\_Click(object sender, EventArgs e)

{

if (string.IsNullOrEmpty(txtMaXe.Text) || string.IsNullOrEmpty(cboHangXe.Text) || string.IsNullOrEmpty(txtBienSo.Text))

{

MessageBox.Show("Mã xe, Hãng xe và Biển số không được để trống.", "Lỗi nhập liệu", MessageBoxButtons.OK, MessageBoxIcon.Error);

return;

}

if (!int.TryParse(txtNamSX.Text, out \_) || !int.TryParse(txtSoGhe.Text, out \_))

{

MessageBox.Show("Năm sản xuất và Số ghế phải là số nguyên.", "Lỗi nhập liệu", MessageBoxButtons.OK, MessageBoxIcon.Error);

return;

}

string sql = "";

if (currentAction == "THEM")

{

// Sửa tên cột và tham số thành chữ thường

sql = "INSERT INTO Xe (maxe, hangxe, bienso, namsx, tinhtrang, soghe) "

+ "VALUES (@maxe, @hangxe, @bienso, @namsx, @tinhtrang, @soghe)";

}

else if (currentAction == "SUA")

{

// Sửa lỗi cú pháp và tên cột/tham số thành chữ thường

sql = "UPDATE Xe SET hangxe = @hangxe, bienso = @bienso, namsx = @namsx, " +

"tinhtrang=@tinhtrang, soghe=@soghe WHERE maxe=@maxe";

}

else // Trường hợp không xác định được hành động (chỉ để an toàn)

{

MessageBox.Show("Không thể thực hiện Lưu. Vui lòng nhấn Thêm hoặc Sửa trước.", "Lỗi", MessageBoxButtons.OK, MessageBoxIcon.Warning);

return;

}

SqlCommand cmd = new SqlCommand(sql, conn);

// Chuyển tên tham số thành chữ thường

cmd.Parameters.AddWithValue("@maxe", txtMaXe.Text);

cmd.Parameters.AddWithValue("@hangxe", cboHangXe.Text);

cmd.Parameters.AddWithValue("@bienso", txtBienSo.Text);

cmd.Parameters.AddWithValue("@namsx", int.Parse(txtNamSX.Text));

cmd.Parameters.AddWithValue("@tinhtrang", cboTinhTrang.Text);

cmd.Parameters.AddWithValue("@soghe", int.Parse(txtSoGhe.Text));

try

{

conn.Open();

int rowsAffected = cmd.ExecuteNonQuery();

if (rowsAffected > 0)

{

if (currentAction == "THEM")

MessageBox.Show("Thêm xe thành công!", "Thông báo", MessageBoxButtons.OK, MessageBoxIcon.Information);

else if (currentAction == "SUA")

MessageBox.Show("Cập nhật thông tin xe thành công!", "Thông báo", MessageBoxButtons.OK, MessageBoxIcon.Information);

}

}

catch (Exception ex)

{

MessageBox.Show("Lỗi khi lưu dữ liệu: " + ex.Message, "Lỗi", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

finally

{

conn.Close();

// Giả định Load\_data() tồn tại và tải lại DataGridView

Load\_data();

currentAction = "DEFAULT";

SetInputControlsEnabled(true); // Giả định hàm này tồn tại

SetButtonStates(false); // Giả định hàm này tồn tại

ClearInputFields(); // Giả định hàm này tồn tại

}

}

private void btnHuy\_Click(object sender, EventArgs e)

{

// Giữ nguyên: Xóa trắng các trường

ClearInputFields();

// Bổ sung: Trở về trạng thái mặc định

currentAction = "DEFAULT";

SetInputControlsEnabled(false);

SetButtonStates(false);

txtMaXe.Enabled = true;

}

private void dgrdsxe\_CellContentClick(object sender, DataGridViewCellEventArgs e)

{

int i = e.RowIndex;

if (i >= 0)

{

txtMaXe.Text = dgrdsxe.Rows[i].Cells["MaXe"].Value.ToString();

cboHangXe.Text = dgrdsxe.Rows[i].Cells["HangXe"].Value.ToString();

txtBienSo.Text = dgrdsxe.Rows[i].Cells["BienSo"].Value.ToString();

txtNamSX.Text = dgrdsxe.Rows[i].Cells["NamSX"].Value.ToString();

cboTinhTrang.Text = dgrdsxe.Rows[i].Cells["TinhTrang"].Value.ToString();

txtSoGhe.Text = dgrdsxe.Rows[i].Cells["SoGhe"].Value.ToString();

}

currentAction = "DEFAULT";

SetInputControlsEnabled(false);

SetButtonStates(false);

}

}

}

# **TÀI LIỆU THAM KHẢO**

1. Xây dựng hệ thống quản lý hồ sơ phương tiện và người lái – Tạp chí Khoa học & Công nghệ

2. Ứng dụng CNTT trong quản lý vận tải và điều phối phương tiện” – Cục Đường Bộ Việt Nam.

3. Tối ưu hóa quản lý xe và tài xế bằng công nghệ số – Tạp chí Logistics Việt Nam

4. Fleet Management System – Phân tích và thiết kế hệ thống – Học viện Công nghệ Bưu chính Viễn thông

5. Ứng dụng .NET xây dựng hệ thống quản lý xe và lái xe – Đại học Cần Thơ