

**TRƯỜNG ĐẠI HỌC AN GIANG
KHOA CÔNG NGHỆ THÔNG TIN**



BÁO CÁO ĐỒ ÁN

Chuyên đề Python (COS525)

**XÂY DỰNG ỨNG DỤNG QUẢN LÝ XE VÀ LÁI
XE VỚI PYTHON, TKINTER VÀ MYSQL**

Giảng viên hướng dẫn: Ths. Nguyễn Ngọc Minhli

**Sinh viên thực hiện: DPM215547.Nguyễn Thị Thanh
Xuân.DH22PM_Nhóm2_01**

AN GIANG, 09-2025

Lời nói đầu

Trong bối cảnh nền kinh tế hội nhập và nhu cầu phát triển mạnh mẽ của ngành dịch vụ logistics và vận tải, việc quản lý đội xe và nhân sự lái xe một cách khoa học, chính xác đã trở thành yếu tố sống còn, quyết định trực tiếp đến hiệu quả kinh doanh và năng lực cạnh tranh của các doanh nghiệp.

Hiện nay, nhiều công ty vận tải vừa và nhỏ vẫn đang phải đối mặt với thách thức trong việc quản lý thông tin rời rạc. Việc theo dõi hồ sơ phương tiện (như biển số, loại xe, lịch bảo trì), hồ sơ lái xe (bằng cấp, kinh nghiệm, lịch làm việc) và đặc biệt là quá trình phân công nhiệm vụ thủ công, tốn nhiều thời gian và dễ xảy ra sai sót. Điều này không chỉ làm giảm năng suất mà còn gây khó khăn trong việc ra quyết định điều hành kịp thời.

Nhận thức được tính cấp thiết của việc tự động hóa và số hóa quy trình quản lý, đề tài "**Xây dựng Ứng dụng Quản lý Xe và Lái Xe với Python, Tkinter và MySQL**" được thực hiện. Mục tiêu chính của đề án là xây dựng một hệ thống phần mềm desktop tập trung, cung cấp các chức năng cơ bản nhưng thiết yếu như thêm mới, chỉnh sửa, tra cứu và báo cáo về thông tin xe và lái xe, từ đó tối ưu hóa công tác quản lý vận hành.

Báo cáo đề án này sẽ trình bày chi tiết toàn bộ quá trình nghiên cứu, từ việc phân tích yêu cầu nghiệp vụ thực tế, thiết kế cấu trúc cơ sở dữ liệu, cho đến việc triển khai code và kiểm thử các chức năng chính của ứng dụng. Hy vọng sản phẩm này sẽ là một công cụ hỗ trợ đắc lực, góp phần hiện đại hóa quy trình quản lý vận tải.

Xin chân thành cảm ơn Thầy Ths. Nguyễn Ngọc Minh đã tận tình chỉ bảo, hỗ trợ em hoàn thành đề án này.

Mục lục

CHƯƠNG 1 :TỔNG QUAN VỀ ĐỀ TÀI	1
1.1 Giới thiệu chung về đề tài	1
1.2 Lý do chọn đề tài	1
1.3 Mục tiêu nghiên cứu đề tài	2
1.4 Phạm vi nghiên cứu	2
1.5 Phương pháp nghiên cứu	3
CHƯƠNG 2 : CỐ SỞ LÝ THUYẾT	3
2.1 Tổng quan về Python	3
2.1.1 Giới thiệu	3
2.1.2 Ứng dụng chung	4
2.1.3 Đặc điểm cốt lõi	4
2.1.4 Ứng dụng trong Đồ án	5
2.2 Thư viện Tkinter (Thành phần giao diện người dùng)	5
2.2.1 Khái niệm	5
2.2.2 Đặc điểm Lập trình	6
2.2.3 Các thành phần cơ bản	6
2.2.4 Vai trò cụ thể của các Widget trong Đồ án	6
2.3 Hệ quản trị cơ sở dữ liệu (MySQL Workbench) Liên kết Python	7
2.3.1 Giới thiệu chung và Cấu trúc Dữ liệu Quan hệ	7
2.3.2 Module kết nối Python và MySQL (Connector)	8
2.4 Môi trường và công cụ phát triển	8
CHƯƠNG 3:PHÂN TÍCH VÀ THIẾT KẾ HỆ THỐNG	9
3.1 Phân tích Yêu cầu	9
3.1.1 Xác định các Tác nhân (Actor) và Phạm vi Hệ thống	9
3.1.2 Phân tích Yêu cầu Chức năng	9
3.1.3 Phân tích Yêu cầu Phi Chức năng	10
3.2 Thiết kế Cơ sở Dữ liệu	10
CHƯƠNG 4 : GIAO DIỆN HỆ THỐNG VÀ CHỨC NĂNG	11
4.1 Giao diện Ứng dụng và Trình diễn Chức năng	11
4.1.1 Giao diện Chính (Main Interface)	11
4.1.2 Trình diễn Chức năng “Quản lý Hồ sơ Xe”	13
4.1.3 Trình diễn Chức năng Quản lý Hồ sơ Lái xe	19
4.1.4 . Trình diễn Chức năng Quản lý Phòng Ban	28
4.1.5 Trình diễn Chức năng Quản lý Chuyển đi	38
4.1.6 Trình diễn Chức năng Quản lý Thống kê	40
CHƯƠNG 6 :KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN	42
6.1 Đánh giá Kết quả Đạt được	43
6.2 Hướng phát triển trong tương lai	44
TÀI LIỆU THAM KHẢO	45

Danh Mục Hình Ảnh

Hình1 : Tổng Quan Python	4
HÌNH 2:CƠ SỞ DỮ LIỆU	8
HÌNH 3: CƠ SỞ DỮ LIỆU ĐỒ ÁN	11
HÌNH 4 : ĐĂNG NHẬP HỆ THỐNG	11
HÌNH 5 : GIAO DIỆN HỆ THỐNG	13
HÌNH 6 : GIAO DIỆN QUẢN LÝ XE	15
HÌNH 7 : GIAO DIỆN THÊM DỮ LIỆU THÀNH CÔNG	16
HÌNH 8 : GIAO DIỆN QUẢN LÝ THÔNG TIN (NHÂN SỰ) LÁI XE	28
HÌNH 9 : DANH SÁCH PHÒNG BAN + CHỨC VỤ	28
HÌNH 10 : GIAO DIỆN THÊM DỮ LIỆU PB + CV	30
HÌNH 11 : GIAO DIỆN QUẢN LÝ CHUYỂN ĐI	39
HÌNH 12 : GIAO DIỆN THÔNG KÊ CHUYỂN ĐI	40
Hình 13 : XUẤT FILE EXCEL	42
Hình 14 : GIA DIỆN XUẤT FILE THÀNH CÔNG	42

CHƯƠNG 1 :TỔNG QUAN VỀ ĐỀ TÀI

1.1 Giới thiệu chung về đề tài

Trong bối cảnh phát triển nhanh chóng của ngành vận tải , việc quản lý đội xe và đội ngũ lái xe đã trở thành yếu tố then chốt quyết định hiệu quả hoạt động của doanh nghiệp. Hiện tại, nhiều đơn vị vẫn đang sử dụng phương pháp quản lý thủ công, dẫn đến tình trạng thông tin rời rạc, khó kiểm soát lịch sử phương tiện, hồ sơ lái xe và quá trình phân công nhiệm vụ. Điều này làm gia tăng chi phí vận hành và giảm khả năng ra quyết định nhanh chóng. Nhằm giải quyết những thách thức trên, đề tài này tập trung Xây dựng Ứng dụng Quản lý Xe và Lái Xe là một giải pháp phần mềm desktop. Ứng dụng cung cấp các chức năng cốt lõi như quản lý hồ sơ xe, quản lý hồ sơ lái xe, theo dõi tình trạng sử dụng và phân công cơ bản. Hệ thống được phát triển sử dụng ngôn ngữ lập trình Python, thư viện giao diện Tkinter và hệ quản trị cơ sở dữ liệu MySQL, đảm bảo tính ổn định, dễ sử dụng và khả năng lưu trữ dữ liệu an toàn, góp phần hiện đại hóa quy trình quản lý vận tải.

1.2 Lý do chọn đề tài

- **Tính cấp thiết trong quản lý vận tải thực tế**

Tình trạng quản lý thủ công bằng sổ sách, bảng tính Excel còn phổ biến, dẫn đến dữ liệu không đồng nhất, khó khăn trong việc tổng hợp và tra cứu thông tin nhanh chóng. Nhu cầu về một công cụ tự động hóa để kiểm soát hồ sơ xe (chuyến đi khởi hành ,thông tin xe), hồ sơ lái xe (bảng cấp, kinh nghiệm) và tình trạng phân công là rất lớn, nhằm tối ưu hóa chi phí và tăng cường an toàn.

- **Khả năng ứng dụng kiến thức và công nghệ**

Đề tài là cơ hội tốt để áp dụng các kiến thức đã học về lập trình Python, đặc biệt là việc xây dựng giao diện người dùng (GUI) với Tkinter và kỹ năng quản trị cơ sở dữ liệu với MySQL. Việc sử dụng Python/Tkinter cung cấp một môi trường phát triển nhanh chóng, dễ bảo trì, đồng thời rèn luyện kỹ năng xây dựng ứng dụng độc lập trên nền tảng desktop.

- **Ý nghĩa thực tiễn**

Kết quả của đồ án là một sản phẩm thực tế có thể ứng dụng trực tiếp tại các doanh nghiệp vận tải quy mô vừa và nhỏ, giúp họ chuyển đổi số hóa quy trình quản lý ban đầu. Đề tài giúp người thực hiện nắm vững toàn bộ quy trình

phát triển một hệ thống thông tin, từ phân tích yêu cầu, thiết kế CSDL đến lập trình và kiểm thử.

1.3 Mục tiêu nghiên cứu đề tài

Xây dựng thành công một hệ thống quản lý thông tin tập trung, ứng dụng công nghệ Python (Tkinter) và MySQL, nhằm tự động hóa quy trình quản lý đội xe và hồ sơ lái xe trong lĩnh vực vận tải. Đóng góp một công cụ đơn giản, hiệu quả vào việc nâng cao hiệu suất làm việc và giảm thiểu sai sót do quản lý thủ công. Phân tích chi tiết quy trình nghiệp vụ quản lý xe, lái xe và thiết kế mô hình cơ sở dữ liệu hoàn chỉnh, đáp ứng được nhu cầu lưu trữ dữ liệu đa dạng:

- Phát triển GUI: Xây dựng giao diện người dùng (GUI) thân thiện, trực quan bằng thư viện Tkinter, đảm bảo người dùng có thể thao tác dễ dàng trên môi trường desktop.
- Chức năng Quản lý: Triển khai đầy đủ các chức năng quản lý CRUD (Create, Read, Update, Delete) cho hai đối tượng chính: Hồ sơ Xe và Hồ sơ Lái xe.
- Chức năng Phân công: Phát triển chức năng gán xe cho lái xe và ngược lại, cho phép theo dõi tình trạng của phương tiện và nhân sự.
- Báo cáo: Xây dựng chức năng tra cứu và xuất báo cáo thống kê đơn giản (ví dụ: lọc tin của các chuyến đi, thông tin xe, danh sách lái xe đã được phân công).

1.4 Phạm vi nghiên cứu

Về chức năng tập trung vào ba nhóm chức năng cốt lõi: Quản lý hồ sơ xe (CRUD), Quản lý hồ sơ lái xe (CRUD), và chức năng Phân công cơ bản (Gán xe cho lái xe theo thời gian).

Ứng dụng không bao gồm các chức năng mở rộng như:

❖ *Phạm vi về công nghệ*

1. Ứng dụng được phát triển trên nền tảng Desktop.
2. Ngôn ngữ lập trình: Python.
3. Thư viện xây dựng giao diện: Tkinter.
4. Hệ quản trị Cơ sở dữ liệu: MySQL.
5. Môi trường hoạt động: Windows/Linux/macOS (tùy thuộc vào cấu hình môi trường Python).

❖ *Các quy trình nghiệp vụ quản lý*

Nghiên cứu và phân tích quy trình nghiệp vụ cơ bản tại các đơn vị vận tải: Quy trình nhập, xuất, sửa thông tin xe và lái xe ,chuyển đi khởi hành .Quy trình phân công nhiệm vụ. Quy trình kiểm tra trạng thái hoạt động.

❖ **Đối tượng kỹ thuật**

Cấu trúc cơ sở dữ liệu (database schema) được xây dựng trên MySQL để lưu trữ dữ liệu.Các thuật toán và logic nghiệp vụ được lập trình bằng Python. Giao diện người dùng (GUI) được thiết kế và triển khai bằng thư viện Tkinter.

1.5 Phương pháp nghiên cứu

•Thu thập tài liệu:

Tìm hiểu yêu cầu nghiệp vụ quản lý vận tải, tham khảo quy trình quản lý xe – lái xe – chuyển đi thông qua tài liệu tham khảo, giáo trình và các hệ thống mẫu.

• Phương pháp phân tích hệ thống:

Phân tích chức năng, dữ liệu và luồng xử lý của bài toán; xác định các bảng dữ liệu cần thiết (xe, lái xe, phòng ban, chuyển đi,...); xây dựng sơ đồ quan hệ và nghiệp vụ.

• Phương pháp thiết kế phần mềm:

Thiết kế giao diện các tab chức năng bằng Tkinter, xây dựng cấu trúc cơ sở dữ liệu MySQL, thiết kế quy trình thêm – sửa – xóa – thống kê dữ liệu.

• Phương pháp lập trình và hiện thực hóa:

Sử dụng Python kết hợp MySQL Connector để xây dựng hệ thống; lập trình chức năng theo từng module: quản lý xe, lái xe, phòng ban, chuyển đi, thống kê.

• Phương pháp kiểm thử:

Thực hiện kiểm thử chức năng (thêm/sửa/xóa), kiểm thử giao diện, kiểm thử dữ liệu và kiểm thử kết nối DB; khắc phục lỗi phát sinh để đảm bảo hệ thống hoạt động ổn định.

• Phương pháp đánh giá:

Đánh giá hệ thống dựa trên tiêu chí đúng yêu cầu, dễ sử dụng, xử lý nhanh và ổn định; cải tiến giao diện và chức năng dựa trên kết quả kiểm thử.

CHƯƠNG 2 : CỐ SỞ LÝ THUYẾT

2.1 Tổng quan về Python

2.1.1 Giới thiệu

Python là ngôn ngữ lập trình đa năng, cấp cao, được tạo ra bởi lập trình viên người Hà Lan Guido van Rossum và phát hành lần đầu vào năm 1991. Triết lý cốt lõi của Python được tóm tắt trong "Zen of Python", trong đó nhấn mạnh mạnh mẽ vào tính dễ đọc (Readability counts) và sự đơn giản (Simple is better than complex) của mã nguồn



Hình1 : Tổng Quan Python

Python là ngôn ngữ lập trình thông dịch (Interpreted), đa mô hình (Multi-paradigm), hỗ trợ Lập trình hướng đối tượng (OOP), Lập trình hướng cấu trúc (Structured Programming) và Lập trình chức năng (Functional Programming).

2.1.2 Ứng dụng chung

Do tính linh hoạt và thư viện phong phú, Python được ứng dụng rộng rãi trong nhiều lĩnh vực:

Phát triển Web: Sử dụng các framework mạnh mẽ như Django và Flask để xây dựng các ứng dụng web quy mô lớn và nhỏ.

Khoa học Dữ liệu và Học máy (Data Science/ML): Đây là lĩnh vực mà Python chiếm ưu thế tuyệt đối nhờ các thư viện như NumPy, Pandas (xử lý dữ liệu), Matplotlib (trực quan hóa), và TensorFlow/PyTorch (học máy).

Tự động hóa và Scripting: Được sử dụng để viết các script tự động hóa các tác vụ quản trị hệ thống, xử lý file, thu thập dữ liệu (Web Scraping) và thực hiện kiểm thử phần mềm.

Phát triển Ứng dụng Desktop (Desktop Application Development): Thông qua các bộ công cụ GUI như Tkinter, PyQt, Python là lựa chọn hiệu quả cho việc phát triển các ứng dụng độc lập trên máy tính cá nhân.

2.1.3 Đặc điểm cốt lõi

Ngôn ngữ thông dịch (Interpreted Language): Mã Python được thực thi từng dòng bởi trình thông dịch (Python Interpreter). Điều này giúp rút ngắn chu trình phát triển (Development Cycle) và tăng tốc độ gỡ lỗi (Debugging) so với ngôn ngữ biên dịch.

Kiểu dữ liệu động (Dynamically Typed): Kiểu dữ liệu của biến được xác định tự động trong thời gian chạy (Runtime) dựa trên giá trị được gán, không cần khai báo kiểu từ trước. Ví dụ: `x = 10` (Integer) và sau đó `x = "hello"` (String) là hợp lệ.

Lập trình Hướng đối tượng (Object-Oriented Programming - OOP): Python hỗ trợ đầy đủ các tính năng OOP như Class, Object, Kế thừa, Đóng gói và Đa hình. Việc này cho phép cấu trúc hóa mã nguồn của ứng dụng quản lý thành các đối tượng độc lập (như đối tượng Xe, đối tượng Lái Xe) với các thuộc tính (dữ liệu) và phương thức (hàm xử lý) rõ ràng, tăng khả năng bảo trì và tái sử dụng.

Quản lý bộ nhớ tự động (Automatic Memory Management): Python sử dụng cơ chế thu gom rác (Garbage Collection), tự động theo dõi và giải phóng bộ nhớ của các đối tượng không còn được chương trình sử dụng, giúp lập trình viên tránh được các lỗi liên quan đến quản lý bộ nhớ thủ công.

2.1.4 Ứng dụng trong Đồ án

Python là lựa chọn tối ưu cho đồ án này vì:

Xây dựng Logic Nghiệp vụ Tập trung: Toàn bộ logic nghiệp vụ phức tạp của ứng dụng (kiểm tra tính hợp lệ dữ liệu, logic phân công, các hàm CRUD) được viết bằng Python, tận dụng cú pháp rõ ràng và khả năng xử lý dữ liệu mạnh mẽ.

Tích hợp CSDL hiệu quả: Python đóng vai trò là lớp giữa giao diện Tkinter và CSDL MySQL, sử dụng các Connector để thực thi các câu lệnh SQL. Điều này đảm bảo dữ liệu từ giao diện được gửi đi và nhận về một cách an toàn và chính xác.

Hệ sinh thái Tkinter: Việc Tkinter là thư viện GUI chuẩn và tích hợp sẵn của Python giúp đơn giản hóa quá trình cài đặt và đóng gói ứng dụng cuối cùng.

2.2 Thư viện Tkinter (Thành phần giao diện người dùng)

2.2.1 Khái niệm

Tkinter là bộ công cụ GUI tiêu chuẩn, được xây dựng trên nền tảng của bộ công cụ Tcl/Tk. Tkinter không phải là một thư viện độc lập mà là một lớp bao bọc (Wrapper) cung cấp giao diện lập trình hướng đối tượng (OOP) để gọi các hàm của Tcl/Tk.

Vai trò: Cung cấp các thành phần đồ họa (Widget) để xây dựng ứng dụng desktop độc lập, tương thích với các hệ điều hành phổ biến (Windows, Linux, macOS) mà không yêu cầu cài đặt thêm các gói bên ngoài.

2.2.2 Đặc điểm Lập trình

Nguyên tắc hoạt động: Ứng dụng Tkinter hoạt động theo mô hình lập trình hướng sự kiện. Thay vì chạy tuần tự, chương trình sẽ chờ đợi các sự kiện (Events) được tạo ra bởi người dùng (như nhấn chuột, nhập phím, di chuyển cửa sổ).

Vòng lặp sự kiện (Main Loop): Mọi ứng dụng Tkinter đều cần một vòng lặp chính (mainloop()) để liên tục lắng nghe và xử lý các sự kiện này. Khi một sự kiện xảy ra, nó sẽ kích hoạt các hàm xử lý (Callback Functions) đã được gán sẵn. Ví dụ, sự kiện click vào nút "Thêm" sẽ gọi hàm `them_ban_ghi()` của Python.

2.2.3 Các thành phần cơ bản

Widgets (Thành phần giao diện): Là các đối tượng trực quan cơ bản mà người dùng tương tác. Mỗi Widget đều có các thuộc tính (ví dụ: màu sắc, kích thước, văn bản) và phương thức riêng.

Geometry Managers (Bộ quản lý bố cục): Dùng để sắp xếp và quản lý vị trí của các Widget bên trong cửa sổ ứng dụng. Tkinter cung cấp ba bộ quản lý chính:

- ❖ `pack()`: Sắp xếp các Widget theo khối (block) một cách đơn giản (trên, dưới, trái, phải).
- ❖ `grid()`: Sắp xếp các Widget theo cấu trúc lưới (hàng và cột), phù hợp cho các form nhập liệu phức tạp, đảm bảo tính căn chỉnh đồng đều giữa các trường dữ liệu của hồ sơ xe/lái xe.
- ❖ `place()`: Đặt Widget ở vị trí tuyệt đối (theo tọa độ X, Y), ít được khuyến nghị vì không linh hoạt khi thay đổi kích thước cửa sổ.

2.2.4 Vai trò cụ thể của các Widget trong Đồ án

➤ **Label (nhãn):**

Mục đích: Hiển thị tên trường dữ liệu (ví dụ: "Biển số xe", "Loại xe", "Họ tên Lái xe, v.v").

➤ **Button (Nút bấm):**

Mục đích: Kích hoạt các thao tác CRUD. Các Button như "Thêm", "Sửa", "Xóa", "Tìm kiếm", "Lưu", "Hủy", "Thoát", "Lọc dữ liệu thông tin", "Xuất file Excel" sẽ được liên kết trực tiếp với các hàm Python xử lý CSDL.

➤ **Entry/Text (Ô nhập liệu):**

Mục đích: Thu thập dữ liệu đầu vào từ người dùng (ví dụ: nhập tải trọng xe, nhập sdt ,v.v.v).

➤ **Treeview (Bảng hiển thị dữ liệu):**

Mục đích: Là Widget quan trọng nhất cho chức năng đọc (Read) dữ liệu. Treeview được sử dụng để:

Hiển thị toàn bộ danh sách Xe hoặc Lái xe dưới dạng bảng với các cột được định nghĩa rõ ràng.

Cung cấp giao diện để người dùng chọn một hàng dữ liệu (bản ghi), sau đó kích hoạt các thao tác Sửa hoặc Xóa bản ghi đó.

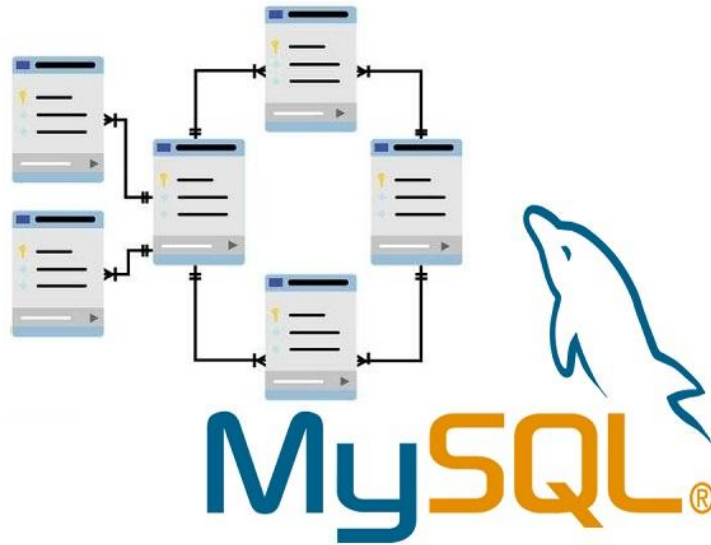
2.3 Hệ quản trị cơ sở dữ liệu (MySQL Workbench) Liên kết Python

2.3.1 Giới thiệu chung và Cấu trúc Dữ liệu Quan hệ

MySQL là một hệ quản trị cơ sở dữ liệu quan hệ (Relational Database Management System - RDBMS) mã nguồn mở hàng đầu, đóng vai trò là kho lưu trữ dữ liệu tập trung, được quản lý thông qua Ngôn ngữ truy vấn có cấu trúc (SQL).

Mô hình Quan hệ (Relational Model): Dữ liệu được tổ chức dưới dạng các bảng (Table) với mối quan hệ rõ ràng. Các bảng trong đồ án (Xe, LaiXe, PhanCong) được thiết kế để đảm bảo tính toàn vẹn thông qua:

- Khóa chính (Primary Key): Đảm bảo tính duy nhất của mỗi bản ghi.
- Khóa ngoại (Foreign Key): Thiết lập liên kết giữa các bảng, duy trì sự nhất quán dữ liệu (Data Consistency) khi thực hiện các thao tác phân công.
- Lý do lựa chọn: MySQL cung cấp sự cân bằng tối ưu giữa Hiệu suất (tốc độ truy vấn nhanh), Độ tin cậy và Chi phí (miễn phí), rất phù hợp cho một ứng dụng desktop độc lập quy mô nhỏ.



HÌNH 2: CƠ SỞ DỮ LIỆU

2.3.2 Module kết nối Python và MySQL (Connector)

Vai trò của Connector: Module `mysql-connector-python` (hoặc `PyMySQL`) đóng vai trò là "cầu nối" (Interface) giữa tầng giao diện người dùng (Tkinter) và tầng dữ liệu (MySQL Server).

Quy trình tương tác: Code Python sử dụng Connector để: (1) Thiết lập kết nối an toàn đến CSDL. (2) Thực thi các chuỗi lệnh SQL (CRUD, JOIN) được sinh ra từ các thao tác trên giao diện. (3) Xử lý giao dịch bằng `commit()` để xác nhận thay đổi hoặc `rollback()` để đảm bảo tính nguyên tử của dữ liệu.

2.4 Môi trường và công cụ phát triển

Sử dụng VS Code hoặc PyCharm: Các IDE này là nền tảng không thể thiếu cho việc phát triển.

Chức năng nâng cao: IDE không chỉ đơn thuần là trình soạn thảo văn bản. Nó cung cấp các công cụ tích hợp sâu như:

- **Gỡ lỗi (Debugging):** Cho phép đặt các điểm dừng (Breakpoints) để theo dõi luồng thực thi của chương trình và kiểm tra giá trị của biến tại thời điểm cụ thể, giúp sửa lỗi logic nghiệp vụ nhanh chóng.
 - **Quản lý dự án:** Hỗ trợ quản lý cấu trúc thư mục, các file code (ví dụ: `main.py`, `database_handler.py`, `gui_module.py`) và môi trường ảo (Virtual Environment), giữ cho môi trường làm việc sạch sẽ và độc lập.
- 2.4.2 Công cụ Phát triển và Quản lý CSDL (MySQL Workbench)

MySQL Workbench là công cụ giao diện người dùng đồ họa (GUI) chính thức, được thiết kế chuyên biệt để hỗ trợ toàn bộ vòng đời phát triển của

MySQL từ thiết kế mô hình dữ liệu, phát triển SQL cho đến quản trị cơ sở dữ liệu.

Vai trò cốt lõi trong Quy trình Phát triển Đồ án:

➤ **Giai đoạn Thiết kế Cấu trúc (ERD Modeling):**

Đóng vai trò là công cụ mô hình hóa trực quan, cho phép xây dựng Sơ đồ Thực thể Liên kết (ERD) chi tiết. Tính năng này hỗ trợ định nghĩa các bảng, khóa chính, khóa ngoại và các ràng buộc dữ liệu một cách logic trước khi triển khai, đảm bảo tính toàn vẹn và nhất quán của CSDL. Workbench tự động sinh ra mã SQL CREATE TABLE từ mô hình này.

➤ **Giai đoạn Phát triển và Tối ưu hóa Truy vấn (SQL Development):**

Cung cấp một giao diện console chuyên nghiệp để biên soạn, kiểm thử và tối ưu hóa các câu lệnh SQL. Đây là nơi xây dựng các truy vấn CRUD (INSERT, SELECT, UPDATE, DELETE) và các câu lệnh JOIN phức tạp cần thiết cho các chức năng báo cáo của ứng dụng Python, đảm bảo hiệu suất truy vấn cao.

➤ **Giai đoạn Kiểm thử và Quản trị Dữ liệu (Administration):**

Hỗ trợ trực tiếp cho quá trình kiểm thử tích hợp (Integration Testing). Người phát triển sử dụng Workbench để:

- (1) Kiểm tra kết quả trực tiếp của các thao tác dữ liệu được thực hiện thông qua giao diện Tkinter.
- (2) Thực hiện các tác vụ quản trị như sao lưu, phục hồi dữ liệu ban đầu, và giám sát hiệu suất hoạt động của CSDL.

CHƯƠNG 3: PHÂN TÍCH VÀ THIẾT KẾ HỆ THỐNG

3.1 Phân tích Yêu cầu

3.1.1 Xác định các Tác nhân (Actor) và Phạm vi Hệ thống

- ❖ Tác nhân chính (Primary Actor): Nhân viên Quản lý Vận tải/Admin Hệ thống (người sử dụng ứng dụng Tkinter để thực hiện CRUD và Phân công).
- ❖ Tác nhân phụ (Secondary Actor): Hệ quản trị CSDL MySQL (đóng vai trò lưu trữ và xử lý dữ liệu).
- ❖ Phạm vi hệ thống: Xác định ranh giới chức năng của ứng dụng theo mục tiêu đã đề ra.

3.1.2 Phân tích Yêu cầu Chức năng

1. Nhóm chức năng Quản lý Xe: CRUD (Thêm, Sửa, Xóa, Xem) hồ sơ nhân sự lái xe, bao gồm thông tin: Biển số, Loại xe, Sức chứa, Loại xe, Tình Trạng , v.v.
2. Nhóm chức năng Quản lý Lái xe: CRUD hồ sơ lái xe, bao gồm: Họ tên, Bằng cấp, bộ phận phòng ban và chức vụ , Kinh nghiệm, v.v.
3. Nhóm chức năng Chuyển đi : Gán xe cho lái xe và ngược lại, mã của chuyến đi đó , theo dõi trạng thái (đang hoạt động/đang rảnh), ngày đi ,ngày về ,đích đến.
4. Nhóm chức năng Báo cáo & Thống kê: Tra cứu, lọc dữ liệu (ví dụ: xe sắp hết hạn đăng kiểm) và xuất dữ liệu.

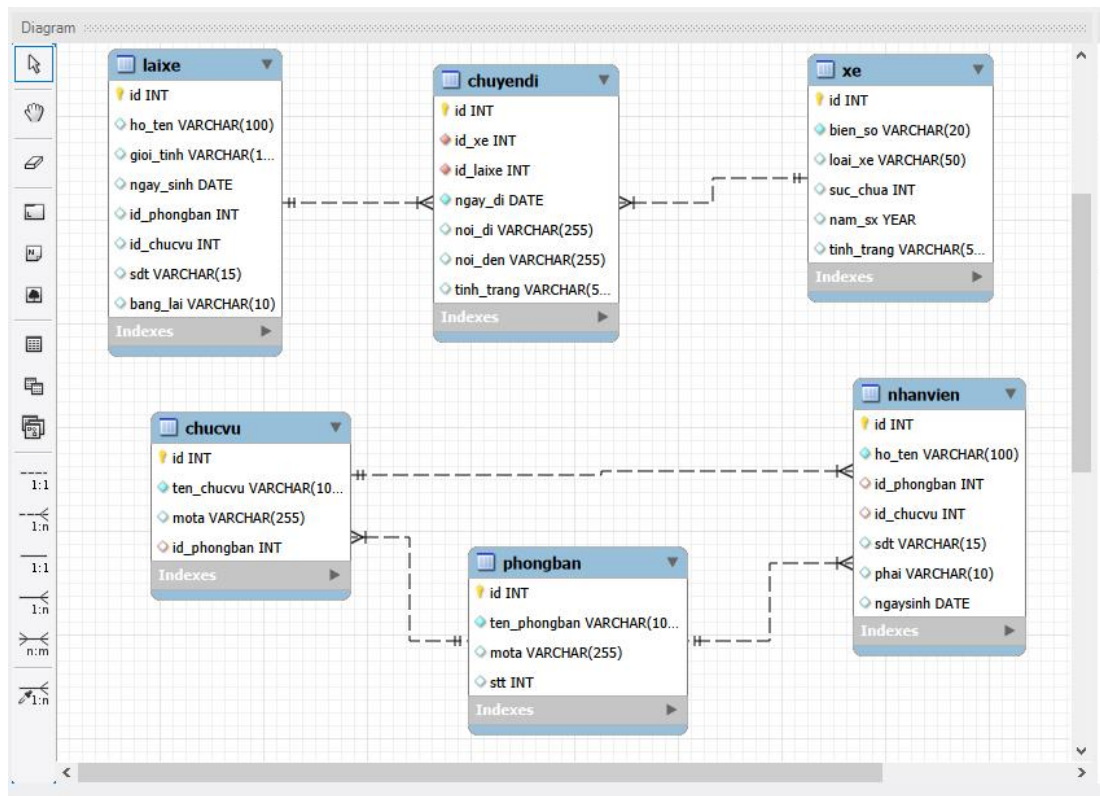
3.1.3 Phân tích Yêu cầu Phi Chức năng

Khả năng sử dụng (Usability): Giao diện trực quan, dễ thao tác, thời gian phản hồi nhanh.

Hiệu suất (Performance): Tốc độ truy vấn CSDL phải nhanh (dưới 1-2 giây) đối với số lượng bản ghi dưới 1000.

Bảo mật (Security): Cơ chế xác thực kết nối CSDL (username/password) và đảm bảo tính toàn vẹn dữ liệu (sử dụng khóa chính/khóa ngoại).

3.2 Thiết kế Cơ sở Dữ liệu



HÌNH 3: CƠ SỞ DỮ LIỆU ĐỒ ÁN

CHƯƠNG 4 : GIAO DIỆN HỆ THỐNG VÀ CHỨC NĂNG

4.1 Giao diện Ứng dụng và Trình diễn Chức năng

4.1.1 Giao diện Chính (Main Interface)

Đăng nhập hệ thống

ĐĂNG NHẬP HỆ THỐNG

Tên đăng nhập: admin

Mật khẩu: ***

Đăng nhập

HÌNH 4 : ĐĂNG NHẬP HỆ THỐNG

```
# =====
# FORM ĐĂNG NHẬP
# =====
def open_main_app():
    login_window.destroy()
    root.deiconify() # Hiện cửa sổ chính

def check_login():
    user = username_entry.get()
    pw = password_entry.get()
```

```
    if user == "admin" and pw == "123":
        messagebox.showinfo
        open_main_app()
    else:
        messagebox.showerror("Sai thông tin", "Tên đăng nhập hoặc mật khẩu không đúng!")
```

```
login_window = tk.Toplevel()
login_window.title("Đăng nhập hệ thống")
login_window.geometry("350x220")
login_window.resizable(False, False)
```

```
# Canh giữa form đăng nhập
sw = login_window.winfo_screenwidth()
sh = login_window.winfo_screenheight()
x = (sw - 350) // 2
y = (sh - 220) // 2
login_window.geometry(f"350x220+{x}+{y}")
```

```
tk.Label(login_window, text="ĐĂNG NHẬP HỆ THỐNG", font=("Arial", 14, "bold")).pack(pady=10)
```

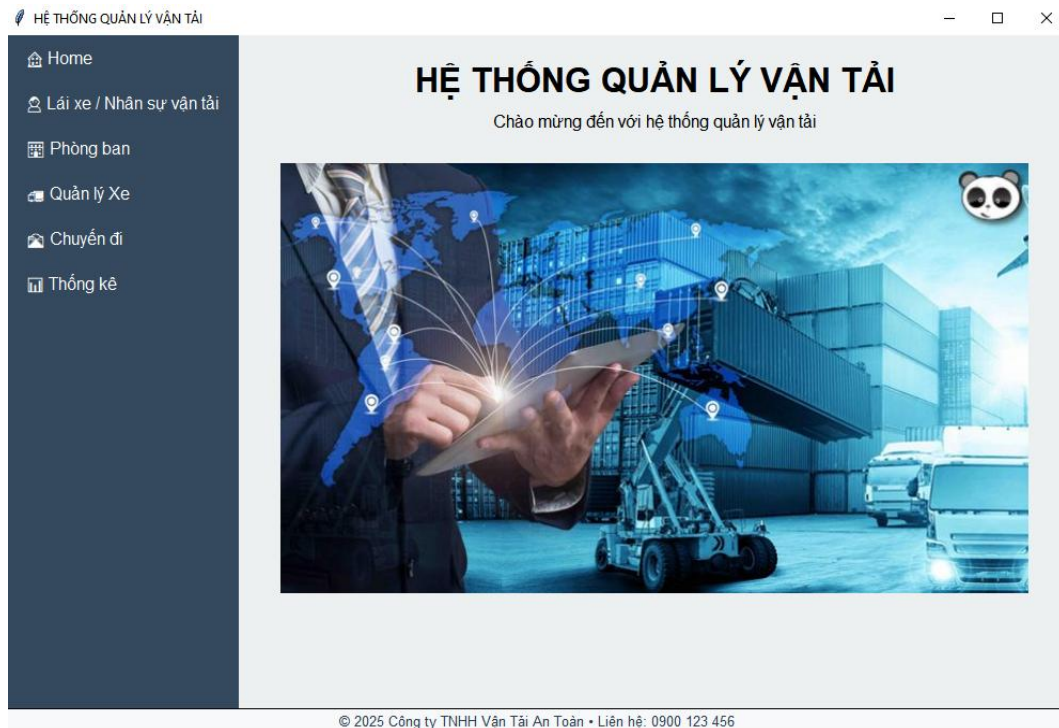
```
frame_login = tk.Frame(login_window)
frame_login.pack(pady=10)
```

```
tk.Label(frame_login, text="Tên đăng nhập:").grid(row=0, column=0)
username_entry = tk.Entry(frame_login, width=25)
username_entry.grid(row=0, column=1, pady=5)
```

```
tk.Label(frame_login, text="Mật khẩu:").grid(row=1, column=0)
password_entry = tk.Entry(frame_login, show="*", width=25)
password_entry.grid(row=1, column=1, pady=5)
```

```
tk.Button(login_window, text="Đăng nhập", width=12, bg="#3498db",
fg="white",
command=check_login).pack(pady=10)
```

Thiết kế tổng thể, bao gồm thanh menu hoặc các nút điều hướng chính (Chức năng Quản lý Xe, Quản lý Lái Xe/nhân sự, Chuyển đi ,thống kê).



HÌNH 5 : GIAO DIỆN HỆ THỐNG

4.1.2 Trình diễn Chức năng “Quản lý Hồ sơ Xe”

Quản lý Xe cho phép hệ thống lưu trữ, cập nhật và tra cứu thông tin các phương tiện được sử dụng trong hoạt động vận tải của đơn vị. Màn hình quản lý được thiết kế trực quan trên giao diện Tkinter, gồm hai phần chính: khung nhập liệu và bảng danh sách xe.

Khung nhập liệu cho phép người dùng nhập thông tin chi tiết của từng xe, bao gồm:

- Biển số xe: định danh duy nhất của phương tiện.
- Loại xe: mô tả chủng loại xe (tải nhỏ, tải lớn, container...).
- Sức chứa: trọng tải hoặc số chỗ ngồi tùy theo loại phương tiện.
- Năm sản xuất: thông tin phục vụ quản lý tuổi đời xe.
- Tình trạng xe: mô tả tình trạng hiện tại của phương tiện (tốt, cần bảo dưỡng...).

Các trường dữ liệu được kiểm tra hợp lệ trước khi lưu để hạn chế lỗi nhập liệu.

```
# -----  
# FORM NHẬP LIỆU
```

```

# -----
form_frame_xe = tk.LabelFrame(tab_xe, text="Thông tin xe", bg="#ecf0f1",
font=("Arial", 12, "bold"))
form_frame_xe.pack(fill="x", padx=20, pady=5)

tk.Label(form_frame_xe, text="Biển số:", bg="#ecf0f1").grid(row=0,
column=0, padx=10, pady=8, sticky="e")
entry_bienso = tk.Entry(form_frame_xe, width=25)
entry_bienso.grid(row=0, column=1, padx=10, pady=8)

tk.Label(form_frame_xe, text="Loại xe:", bg="#ecf0f1").grid(row=0,
column=2, padx=10, pady=8, sticky="e")
entry_loaixex = tk.Entry(form_frame_xe, width=25)
entry_loaixex.grid(row=0, column=3, padx=10, pady=8)

tk.Label(form_frame_xe, text="Sức chứa:", bg="#ecf0f1").grid(row=1,
column=0, padx=10, pady=8, sticky="e")
entry_succhua = tk.Entry(form_frame_xe, width=25)
entry_succhua.grid(row=1, column=1, padx=10, pady=8)

tk.Label(form_frame_xe, text="Năm SX:", bg="#ecf0f1").grid(row=1,
column=2, padx=10, pady=8, sticky="e")
entry_namsx = tk.Entry(form_frame_xe, width=25)
entry_namsx.grid(row=1, column=3, padx=10, pady=8)

tk.Label(form_frame_xe, text="Tình trạng:", bg="#ecf0f1").grid(row=2,
column=0, padx=10, pady=8, sticky="e")
entry_tinhtrang = tk.Entry(form_frame_xe, width=25)
entry_tinhtrang.grid(row=2, column=1, padx=10, pady=8)

```

HỆ THỐNG QUẢN LÝ VẬN TẢI

QUẢN LÝ XE

Thông tin xe

Biển số: Loại xe:

Sức chứa: Năm SX:

Tình trạng:

Thêm **Sửa** **Lưu** **Xóa** **Hủy** **Thoát**

STT	Biển số	Loại xe	Sức chứa	Năm SX	Tình trạng
1	51A-12345	Xe tải	30	2020	Tốt
2	51B-54321	Xe khách	55	2019	Tốt

HÌNH 6 : GIAO DIỆN QUẢN LÝ XE

QUẢN LÝ XE

Thông tin xe

Biển số: Loại xe:

Sức chứa: Năm SX:

Tình trạng:

Thêm **Sửa** **Lưu** **Xóa** **Hủy** **Thoát**

STT	Biển số	Loại xe	Sức chứa	Năm SX	Tình trạng
1	51A-12345	Xe tải	30	2020	Tốt
2	51B-54321	Xe khách	55	2019	Tốt

Thành công

Thêm xe thành công!

OK

QUẢN LÝ XE

Thông tin xe

Biển số:

Loại xe:

Sức chứa:

Năm SX:

Tình trạng:

Thêm
Sửa
Lưu
Xóa
Hủy
Thoát

STT	Biển số	Loại xe	Sức chứa	Năm SX	Tình trạng
1	51A-12345	Xe tải	30	2020	Tốt
2	51B-54321	Xe khách	55	2019	Tốt
3	67C1-6789	Xe HonDa	50	2024	Bảo dưỡng

HÌNH 7 : GIAO DIỆN THÊM DỮ LIỆU THÀNH CÔNG

```
# -----
# CRUD FUNCTIONS
# -----

def load_xe():
    tree_xe.delete(*tree_xe.get_children())
    cursor = conn.cursor()
    cursor.execute("SELECT bien_so, loai_xe, suc_chua, nam_sx,
    tinh_trang FROM xe ORDER BY id ASC")
    rows = cursor.fetchall()
    for stt, row in enumerate(rows, start=1):
        tree_xe.insert("", tk.END, values=(stt, row[0], row[1], row[2],
    row[3], row[4]))

def add_xe():
    bien_so = entry_bienso.get()
    loai_xe = entry_loaixxe.get()
    suc_chua = entry_succhua.get()
    nam_sx = entry_namsx.get()
    tinh_trang = entry_tinhtrang.get() or "Tốt"
```

```
    if not bien_so:
        messagebox.showwarning("Cảnh báo", "Biển số không được để
    trống!")
    return
```

```
    try:
        suc_chua_val = int(suc_chua) if suc_chua else 0
    except ValueError:
        messagebox.showerror("Lỗi", "Sức chứa phải là số nguyên!")
```

```
return
```

```
try:
    nam_sx_val = int(nam_sx) if nam_sx else None
except ValueError:
    messagebox.showerror("Lỗi", "Năm SX phải là số hợp lệ!")
    return
```

```
cursor = conn.cursor()
sql = "INSERT INTO xe (bien_so, loai_xe, suc_chua, nam_sx,
    tinh_trang) VALUES (%s,%s,%s,%s,%s)"
cursor.execute(sql, (bien_so, loai_xe, suc_chua_val, nam_sx_val,
    tinh_trang))
conn.commit()
```

```
messagebox.showinfo("Thành công", "Thêm xe thành công!")
clear_form_xe()
load_xe()
```

```
def edit_xe_mode():
    global mode
    selected = tree_xe.focus()
    if not selected:
        messagebox.showwarning("Cảnh báo", "Chọn xe để sửa!")
        return
    mode = "edit"
    data = tree_xe.item(selected)["values"]
    entry_bienso.delete(0, tk.END)
    entry_bienso.insert(0, data[1])
    entry_loaixe.delete(0, tk.END)
    entry_loaixe.insert(0, data[2])
    entry_succhua.delete(0, tk.END)
    entry_succhua.insert(0, data[3])
    entry_namsx.delete(0, tk.END)
    entry_namsx.insert(0, data[4])
    entry_tinhtrang.delete(0, tk.END)
    entry_tinhtrang.insert(0, data[5])
```

```
def save_xe():
    global mode
    if mode != "edit":
        messagebox.showinfo("Info", "Chỉ dùng để cập nhật sửa!")
        return
```

```
selected = tree_xe.focus()
if not selected:
    messagebox.showwarning("Cảnh báo", "Chọn xe để sửa!")
    return
```

```
bien_so = entry_bienso.get()
```

```
loai_xe = entry_loaixex.get()
suc_chua = entry_succhua.get()
nam_sx = entry_namsx.get()
tinh_trang = entry_tinhtrang.get() or "Tốt"
```

```
try:
    suc_chua_val = int(suc_chua) if suc_chua else 0
except ValueError:
    messagebox.showerror("Lỗi", "Sức chứa phải là số nguyên!")
    return
```

```
try:
    nam_sx_val = int(nam_sx) if nam_sx else None
except ValueError:
    messagebox.showerror("Lỗi", "Năm SX phải là số hợp lệ!")
    return
```

```
stt = tree_xe.item(selected)["values"][0]
# Lấy ID thực từ DB để UPDATE
cursor = conn.cursor()
cursor.execute("SELECT id FROM xe ORDER BY id ASC LIMIT %s,1",
(stt-1,))
xe_id = cursor.fetchone()[0]
```

```
sql = "UPDATE xe SET bien_so=%s, loai_xe=%s, suc_chua=%s, nam_sx=%s,
tinh_trang=%s WHERE id=%s"
cursor.execute(sql, (bien_so, loai_xe, suc_chua_val, nam_sx_val,
tinh_trang, xe_id))
conn.commit()
messagebox.showinfo("Thành công", "Cập nhật xe thành công!")
clear_form_xe()
load_xe()
mode = None
```

```
def delete_xe():
    selected = tree_xe.focus()
    if not selected:
        messagebox.showwarning("Cảnh báo", "Chọn xe để xóa!")
        return
```

```
stt = tree_xe.item(selected)["values"][0]
cursor = conn.cursor()
cursor.execute("SELECT id FROM xe ORDER BY id ASC LIMIT %s,1",
(stt-1,))
xe_id = cursor.fetchone()[0]
```

```
if messagebox.askyesno("Xác nhận", "Bạn có chắc muốn xóa xe này?"):
    cursor.execute("DELETE FROM xe WHERE id=%s", (xe_id,))
    conn.commit()
    messagebox.showinfo("Thành công", "Xóa xe thành công!")
```

```

        clear_form_xe()
        load_xe()

def exit_app():
    if messagebox.askyesno("Xác nhận", "Bạn có chắc muốn thoát chương trình?"):
        root.destroy() # đóng hẳn cửa sổ Tkinter

def exit_tab():
    root.destroy()

```

Chức năng CRUD:

Tab Quản lý Xe hỗ trợ đầy đủ các chức năng xử lý dữ liệu:

- Thêm xe:
Người dùng nhập thông tin xe rồi lưu vào cơ sở dữ liệu MySQL. Hệ thống kiểm tra ràng buộc như không để trống biển số, sức chứa phải là số nguyên, năm sản xuất phải hợp lệ.
- Sửa xe:
Cho phép chọn một dòng trên bảng, nạp dữ liệu lên form để chỉnh sửa. Khi lưu, hệ thống xác định chính xác ID xe trong MySQL để cập nhật đúng bản ghi.
- Xóa xe:
Người dùng có thể xóa một xe khỏi hệ thống. Trước khi xóa, chương trình hiển thị hộp thoại xác nhận nhằm tránh thao tác nhầm.
- Hủy nhập:
Xóa trắng toàn bộ thông tin đang nhập trên form.
- Thoát:
Đóng tab/quay về giao diện chính.

Chức năng Quản lý Xe

Dễ dàng theo dõi và cập nhật thông tin đội xe. Hạn chế sai sót nhờ kiểm tra dữ liệu nhập. Lưu trữ tập trung trên MySQL, thuận tiện cho mở rộng hoặc tích hợp các chức năng khác như phân công chuyển đi, quản lý bảo dưỡng...

4.1.3 Trình diễn Chức năng Quản lý Hồ sơ Lái xe

- Thêm hồ sơ lái xe

Nhập thông tin vào form: họ tên, giới tính, ngày sinh, phòng ban, chức vụ, SĐT và bằng lái.

Phòng ban và chức vụ được lấy tự động từ hai bảng tương ứng.
Khi nhấn nút Thêm, hệ thống:

Kiểm tra thông tin bắt buộc. Tự động tìm ID của phòng ban/chức vụ. Thực hiện câu lệnh INSERT INTO laixe.

Cập nhật bảng Treeview.

➤ **Sửa thông tin**

Người dùng chọn dòng cần sửa trong Treeview → dữ liệu được đổ ngược vào form.

ID bị khóa (không cho chỉnh sửa). Nhấn Lưu để cập nhật. Câu lệnh UPDATE được thực thi và Treeview được refresh.

➤ **Xóa hồ sơ**

Người dùng chọn lái xe cần xóa và nhấn Xóa. Hệ thống hiển thị hộp thoại xác nhận. Thực hiện câu lệnh DELETE FROM laixe.

➤ **Hủy / Làm mới form**

Nút Hủy xóa toàn bộ dữ liệu nhập và đưa form về mặc định. Giao diện này giúp người dùng tạo – chỉnh sửa – quản lý hồ sơ lái xe một cách trực quan và thuận tiện.

Liên kết dữ liệu với Phòng ban & Chức vụ. Form nhập liệu luôn cập nhật danh sách phòng ban/chức vụ mới nhất mỗi khi tab được mở.

Nếu người dùng thêm hoặc xóa phòng ban/chức vụ ở tab khác thì combobox trong tab NV/LX cũng tự được làm mới.

➤ **Tóm tắt hoạt động**

Chức năng Quản lý Hồ sơ Lái xe đảm bảo:

```
# =====  
# TAB NHÂN VIÊN / LÁI XE  
# =====  
tab_nv = tk.Frame(content_frame, bg="#ecf0f1")  
pages["NhanVien"] = tab_nv  
  
tk.Label(tab_nv, text="QUẢN LÝ NHÂN VIÊN / LÁI XE",  
         font=("Arial", 18, "bold"), bg="#ecf0f1").pack(pady=10)
```

```
# -----  
# Notebook bên trong tab NV  
# -----  
nv_notebook = ttk.Notebook(tab_nv)  
nv_notebook.pack(fill="both", expand=True, padx=20, pady=5)
```



```
# -----  
# TAB 1: Danh sách nhân viên  
# -----
```

```
tab_list = tk.Frame(nv_notebook, bg="#f7f9fc")  
nv_notebook.add(tab_list, text=" NV/LX")
```

```
columns =  
("id", "name", "gender", "dob", "phongban", "chucvu", "phone", "license")  
tree_list = ttk.Treeview(tab_list, columns=columns, show="headings",  
height=20)
```

```
col_text = ["Mã NV/LX", "Họ và tên", "Giới tính", "Ngày sinh", "Phòng  
ban", "Chức vụ", "SĐT", "Bằng lái"]  
col_width = [80, 120, 80, 80, 100, 80, 80, 80]
```

```
for col, txt, w in zip(columns, col_text, col_width):  
    tree_list.heading(col, text=txt)  
    tree_list.column(col, width=w, anchor="center")
```

```
sb = ttk.Scrollbar(tab_list, orient="vertical", command=tree_list.yview)  
tree_list.configure(yscrollcommand=sb.set)  
tree_list.pack(side="left", fill="both", expand=True)  
sb.pack(side="right", fill="y")
```

```
# ----- Style Treeview đẹp -----  
style = ttk.Style()  
style.configure("Treeview", rowheight=22, font=("Arial", 10))  
style.map('Treeview', background=[('selected', '#2980b9')],  
foreground=[('selected', 'white')])  
tree_list.tag_configure('oddrow', background="#f0f3f7")  
tree_list.tag_configure('evenrow', background="ffffff")
```

```
# =====  
# TAB 2: Thêm/Sửa/Xóa  
# =====
```

```
tab_form = tk.Frame(nv_notebook, bg="#ecf0f1")  
nv_notebook.add(tab_form, text="Nhập / Sửa / Xóa")
```

```
# ----- Form nhập liệu -----  
form_frame = tk.LabelFrame(tab_form, text="Thông tin nhân viên / lái  
xe",  
bg="#ecf0f1", font=("Arial", 12, "bold"))  
form_frame.pack(fill="x", pady=10, padx=10)
```

```
tk.Label(form_frame, text="Mã NV/LX:", bg="#ecf0f1").grid(row=0,  
column=0, padx=10, pady=8, sticky="e")  
entry_id = tk.Entry(form_frame, width=25)  
entry_id.grid(row=0, column=1, padx=10, pady=8)
```

```
tk.Label(form_frame, text="Họ tên:", bg="#ecf0f1").grid(row=0, column=2,
padx=10, pady=8, sticky="e")
entry_name = tk.Entry(form_frame, width=25)
entry_name.grid(row=0, column=3, padx=10, pady=8)
```

```
gender_var = tk.StringVar(value="Nam")
tk.Label(form_frame, text="Giới tính:", bg="#ecf0f1").grid(row=1,
column=0, padx=10, pady=8, sticky="e")
gender_frame = tk.Frame(form_frame, bg="#ecf0f1")
gender_frame.grid(row=1, column=1, padx=10, pady=8, sticky="w")
tk.Radiobutton(gender_frame, text="Nam", variable=gender_var,
value="Nam", bg="#ecf0f1").pack(side="left")
tk.Radiobutton(gender_frame, text="Nữ", variable=gender_var, value="Nữ",
bg="#ecf0f1").pack(side="left")
```

```
tk.Label(form_frame, text="Ngày sinh:", bg="#ecf0f1").grid(row=1,
column=2, padx=10, pady=8, sticky="e")
entry_dob = DateEntry(form_frame, width=23, background='darkblue',
foreground='white', borderwidth=2, date_pattern="yyyy-mm-dd")
entry_dob.grid(row=1, column=3, padx=10, pady=8)
```

```
tk.Label(form_frame, text="Phòng ban:", bg="#ecf0f1").grid(row=2,
column=0, padx=10, pady=8, sticky="e")
cb_phongban = ttk.Combobox(form_frame, width=22, state="readonly")
cb_phongban.grid(row=2, column=1, padx=10, pady=8)
```

```
tk.Label(form_frame, text="Chức vụ:", bg="#ecf0f1").grid(row=2,
column=2, padx=10, pady=8, sticky="e")
cb_chucvu = ttk.Combobox(form_frame, width=22, state="readonly")
cb_chucvu.grid(row=2, column=3, padx=10, pady=8)
```

```
tk.Label(form_frame, text="SĐT:", bg="#ecf0f1").grid(row=3, column=0,
padx=10, pady=8, sticky="e")
entry_phone = tk.Entry(form_frame, width=25)
entry_phone.grid(row=3, column=1, padx=10, pady=8)
```

```
tk.Label(form_frame, text="Bằng lái:", bg="#ecf0f1").grid(row=3,
column=2, padx=10, pady=8, sticky="e")
cb_license = ttk.Combobox(form_frame,
values=["A1", "A2", "B1", "B2", "C", "D", "E", "FC", "FE"], width=25,
state="readonly")
cb_license.grid(row=3, column=3, padx=10, pady=8)
# =====
# HÀM CRUD
# =====
def clear_form():
    entry_id.config(state="normal")
    entry_id.delete(0, tk.END)
    entry_name.delete(0, tk.END)
```

```

gender_var.set("Nam")
entry_dob.set_date(date.today())
cb_phongban.set("")
cb_chucvu.set("")
entry_phone.delete(0, tk.END)
cb_license.set("")
global selected_id
selected_id = None

```

```

def load_nhanvien():
    tree_nv.delete(*tree_nv.get_children())
    cur = conn.cursor()
    cur.execute("""
        SELECT nv.id, nv.ho_ten, nv.gioi_tinh, nv.ngay_sinh,
               pb.ten_phongban, cv.ten_chucvu,
               nv.sdt, nv.bang_lai
        FROM laixe nv
        LEFT JOIN phongban pb ON nv.id_phongban = pb.id
        LEFT JOIN chucvu cv ON nv.id_chucvu = cv.id
    """)
    for row in cur.fetchall():
        tree_nv.insert("", "end", values=row)
    cur.close()

```

```

def add_nv():
    name = entry_name.get().strip()
    phongban = cb_phongban.get()
    chucvu = cb_chucvu.get()
    if not name or not phongban or not chucvu:
        return messagebox.showwarning("Chú ý", "Không để trống Tên /
Phòng ban / Chức vụ")

```

```

nv_id = entry_id.get().strip() or None
gender = gender_var.get()
dob = entry_dob.get_date().strftime("%Y-%m-%d")
phone = entry_phone.get().strip()
license_val = cb_license.get()

```

```

cur = conn.cursor()
cur.execute("SELECT id FROM phongban WHERE ten_phongban=%s",
(phongban,))
pb_id = cur.fetchone()[0]

```

```

cur.execute("SELECT id FROM chucvu WHERE ten_chucvu=%s", (chucvu,))
cv_id = cur.fetchone()[0]

```

```

cur.execute("""

```

```

        INSERT INTO laixe (id, ho_ten, gioi_tinh, ngay_sinh,
id_phongban, id_chucvu, sdt, bang_lai)
        VALUES (%s,%s,%s,%s,%s,%s,%s,%s)
        """, (nv_id, name, gender, dob, pb_id, cv_id, phone, license_val))
    conn.commit()
    cur.close()

```

```

load_nhanvien()
clear_form()

```

```

def edit_nv():
    global selected_id
    sel = tree_nv.selection()
    if not sel:
        return messagebox.showwarning("Chú ý", "Chọn nhân viên để sửa!")

```

```

values = tree_nv.item(sel)["values"]
selected_id = values[0]

```

```

entry_id.config(state="disabled")
entry_id.delete(0, tk.END)
entry_id.insert(0, values[0])
entry_name.delete(0, tk.END)
entry_name.insert(0, values[1])
gender_var.set(values[2])
entry_dob.set_date(values[3])
cb_phongban.set(values[4])
cb_chucvu.set(values[5])
entry_phone.delete(0, tk.END)
entry_phone.insert(0, values[6])
cb_license.set(values[7])

```

```

def save_nv():
    global selected_id
    if not selected_id:
        return messagebox.showwarning("Chú ý", "Bạn chưa chọn nhân viên để lưu!")

```

```

name = entry_name.get().strip()
phongban = cb_phongban.get()
chucvu = cb_chucvu.get()
gender = gender_var.get()
dob = entry_dob.get_date().strftime("%Y-%m-%d")
phone = entry_phone.get().strip()
license_val = cb_license.get()

```

```

cur = conn.cursor()
cur.execute("SELECT id FROM phongban WHERE ten_phongban=%s",
(phongban,))

```

```
pb_id = cur.fetchone()[0]
```

```
cur.execute("SELECT id FROM chucvu WHERE ten_chucvu=%s", (chucvu,))  
cv_id = cur.fetchone()[0]
```

```
cur.execute("""  
    UPDATE laixe  
    SET ho_ten=%s, gioi_tinh=%s, ngay_sinh=%s, id_phongban=%s,  
id_chucvu=%s, sdt=%s, bang_lai=%s  
    WHERE id=%s  
    """, (name, gender, dob, pb_id, cv_id, phone, license_val,  
selected_id))  
conn.commit()  
cur.close()
```

```
load_nhanvien()  
clear_form()
```

```
def delete_nv():  
    sel = tree_nv.selection()  
    if not sel:  
        return messagebox.showwarning("Chú ý", "Chọn nhân viên để xóa!")
```

```
nv_id = tree_nv.item(sel)["values"][0]
```

```
if not messagebox.askyesno("Xác nhận", f"Bạn chắc chắn xóa  
{nv_id}?"):  
    return
```

```
cur = conn.cursor()  
cur.execute("DELETE FROM laixe WHERE id=%s", (nv_id,))  
conn.commit()  
cur.close()
```

```
load_nhanvien()  
clear_form()  
# Frame chứa nút, full chiều ngang  
btn_frame = tk.Frame(tab_form, bg="#ecf0f1")  
btn_frame.pack(fill="x", pady=10, padx=10)
```

```
# Frame con để căn giữa các nút  
center_frame = tk.Frame(btn_frame, bg="#ecf0f1")  
center_frame.pack(anchor="center") # anchor center
```

```
# Tạo các nút trong frame con  
buttons = [  
    ("Thêm", "#1abc9c", add_nv),  
    ("Sửa", "#3498db", edit_nv),  
    ("Xóa", "#e74c3c", delete_nv),  
    ("Lưu", "#f39c12", save_nv),
```

```

        ("Hủy", "#95a5a6", clear_form),
        ("Thoát", "#7f8c8d", lambda: show_page("Home"))
    ]

```

```

for text, color, cmd in buttons:
    tk.Button(center_frame, text=text, bg=color, fg="white", width=12,
command=cmd).pack(side="left", padx=5)

```

```

def load_pb_to_combobox():
    """Cập nhật danh sách Phòng ban và Chức vụ cho combobox của tab
NV"""

```

```

    try:
        cur = conn.cursor()
        # Phòng ban
        cur.execute("SELECT ten_phongban FROM phongban ORDER BY
ten_phongban")
        pb_list = [r[0] for r in cur.fetchall()]
        cb_phongban['values'] = pb_list

```

```

        # Chức vụ
        cur.execute("SELECT ten_chucvu FROM chucvu ORDER BY ten_chucvu")
        cv_list = [r[0] for r in cur.fetchall()]
        cb_chucvu['values'] = cv_list

```

```

        cur.close()
    except Exception as e:
        messagebox.showerror("Lỗi", f"Không tải được danh sách PB/CV:
{e}")

```

```

def refresh_cb_nv():
    cur = conn.cursor()
    cur.execute("SELECT ten_phongban FROM phongban ORDER BY
ten_phongban")
    cb_phongban['values'] = [r[0] for r in cur.fetchall()]

```

```

    cur.execute("SELECT ten_chucvu FROM chucvu ORDER BY ten_chucvu")
    cb_chucvu['values'] = [r[0] for r in cur.fetchall()]
    cur.close()
load_pb_to_combobox()

```

```

# ----- Treeview Frame -----
tree_frame = tk.Frame(tab_form, bg="#ecf0f1")
tree_frame.pack(fill="both", expand=True, pady=10, padx=10)

```

```

columns =
("id", "name", "gender", "dob", "phongban", "chucvu", "phone", "license")
tree_nv = ttk.Treeview(tree_frame, columns=columns, show="headings",
height=12)

```

```
col_text = ["Mã NV/LX", "Họ và tên", "Giới tính", "Ngày sinh", "Phòng ban", "Chức vụ", "SĐT", "Bằng lái"]
col_width = [80, 120, 80, 80, 100, 80, 80, 80]
```

```
for col, txt, w in zip(columns, col_text, col_width):
    tree_nv.heading(col, text=txt)
    tree_nv.column(col, width=w, anchor="center")
```

```
# Scrollbars
scroll_y = ttk.Scrollbar(tree_frame, orient="vertical",
    command=tree_nv.yview)
scroll_x = ttk.Scrollbar(tree_frame, orient="horizontal",
    command=tree_nv.xview)
tree_nv.configure(yscrollcommand=scroll_y.set,
    xscrollcommand=scroll_x.set)
```

```
tree_nv.pack(side="left", fill="both", expand=True)
scroll_y.pack(side="right", fill="y")
scroll_x.pack(side="bottom", fill="x")
```

```
# =====
# BIẾN TRẠNG THÁI
# =====
selected_id = None
```

```
# =====
# KHỞI TẠO TAB NV
# =====
refresh_cb_nv() # load PB/CV cho combobox
load_nhanvien() # load dữ liệu Treeview
clear_form() # reset form
```

Nhóm chức năng	Mô tả
Xem	Hiển thị toàn bộ danh sách lái xe trong Treeview.
Thêm	Nhập hồ sơ mới và lưu vào CSDL.
Sửa	Cập nhật thông tin lái xe đã chọn.
Xóa	Loại bỏ hồ sơ khỏi hệ thống sau khi xác nhận.
Tự động hóa	Tải lại dữ liệu, liên kết bảng PB/CV, làm mới form.

➤ Giao diện demo

QUẢN LÝ NHÂN VIÊN / LÁI XE

NV/LX
Nhập / Sửa / Xóa

Thông tin nhân viên / lái xe

Mã NV/LX:

Họ tên:

Giới tính: ☒ Nam ☐ Nữ

Ngày sinh:

Phòng ban:

Chức vụ:

SĐT:

Bằng lái:

Thêm
Sửa
Xóa
Lưu
Hủy
Thoát

Mã NV/LX	Họ và tên	Giới tính	Ngày sinh	Phòng ban	Chức vụ	SĐT	Bằng lái
1	Tài xế A	Nam	1990-01-01	None	None	0909000000	B2
2	Tài xế B	Nam	1992-05-10	None	None	0909000001	B2

HÌNH 8 : GIAO DIỆN QUẢN LÝ THÔNG TIN (NHÂN SỰ) LÁI XE

4.1.4 . Trình diễn Chức năng Quản lý Phòng Ban

Chức năng và Vai trò của Tab Phòng Ban

Tab Phòng Ban & Chức Vụ được xây dựng nhằm hỗ trợ quản lý toàn bộ thông tin liên quan đến phòng ban và chức vụ trong hệ thống quản lý vận tải. Đây là một phân hệ quan trọng giúp tổ chức bộ máy nhân sự rõ ràng, hỗ trợ cho việc phân công tài xế – phân bổ xe – và các nghiệp vụ liên quan. Phân hệ cung cấp 2 nhóm chức năng chính: Xem danh sách phòng ban và chức vụ

HỆ THỐNG QUẢN LÝ VẬN TẢI
— □ ×

Home

Lái xe / Nhân sự vận tải

Phòng ban

Quản lý Xe

Chuyển đi

Thống kê

Danh sách PB + CV
Quản lý PB

STT	Mã P	Tên PB	Mô tả PB	Tên CV	Mô tả CV
1	25	Phòng Quản Lý	Quản lý đăng ký xe, bảo dưỡng, hồ sơ xe	Trưởng Phòngg	Quản lý tổng thể phòng ban
2	26	Kỹ Thuật	xử lý kỹ thuật, giảm thiểu sự cố khẩn cấp	Quản lý	theo dõi tiến độ của phòng b;

HÌNH 9 : DANH SÁCH PHÒNG BAN + CHỨC VỤ

➤ Giao diện Sub-tab 1: Danh Sách PB + CV

```
# =====
# Sub-tab 1: Danh sách PB + CV
# =====
```



```

tab_pb_list = tk.Frame(sub_tab_control, bg="#ecf0f1")
sub_tab_control.add(tab_pb_list, text="Danh sách PB + CV")

tree1_frame = tk.Frame(tab_pb_list)
tree1_frame.pack(fill="both", expand=True, padx=20, pady=20)

```

```

columns1 = ("stt", "id", "ten_pb", "mota_pb", "ten_cv", "mota_cv")
tree_pb_all = ttk.Treeview(tree1_frame, columns=columns1,
show="headings", height=20)

```

```

headers = ["STT", "Mã PB", "Tên PB", "Mô tả PB", "Tên CV", "Mô tả CV"]
widths = [50, 60, 150, 200, 150, 200]

```

```

for col, head, w in zip(columns1, headers, widths):
    tree_pb_all.heading(col, text=head)
    tree_pb_all.column(col, width=w, anchor="center")

```

```

tree_pb_all.pack(fill="both", expand=True)
scroll_y1 = ttk.Scrollbar(tree1_frame, orient="vertical",
command=tree_pb_all.yview)
tree_pb_all.configure(yscrollcommand=scroll_y1.set)
scroll_y1.pack(side="right", fill="y")

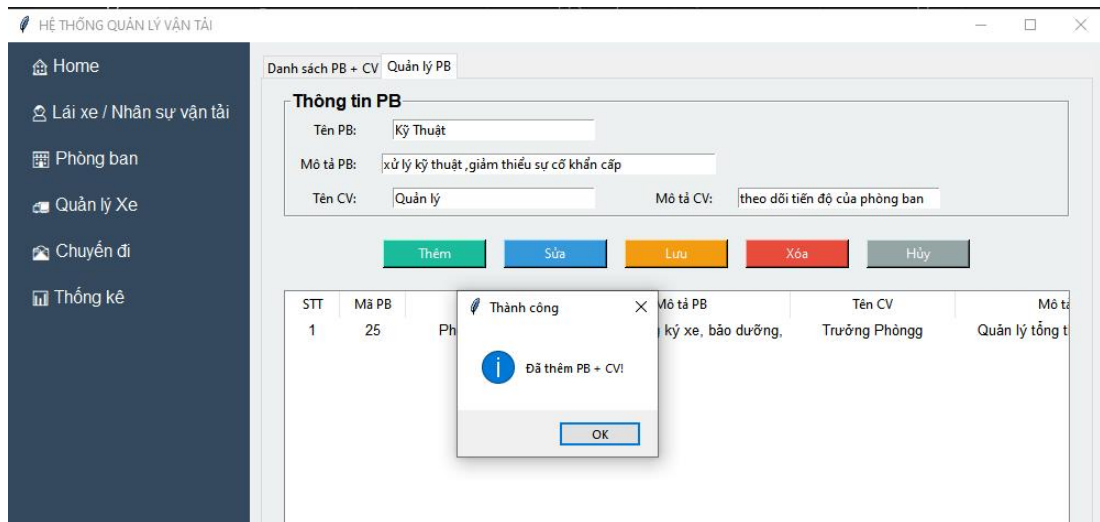
```

=> Sub-tab này hiển thị toàn bộ phòng ban và chức vụ dưới dạng bảng (TreeView), cho phép người dùng:

- Xem STT, Mã phòng ban, Tên phòng ban, Mô tả, Tên chức vụ, Mô tả chức vụ
- Dữ liệu được lấy từ bảng phongban và chucvu, kết hợp bằng lệnh LEFT JOIN, đảm bảo hiển thị đầy đủ ngay cả khi phòng ban chưa có chức vụ.

Giúp người dùng quan sát nhanh cấu trúc tổ chức trong doanh nghiệp.

➤ Quản lý (CRUD) phòng ban và chức vụ



HÌNH 10 : GIAO DIỆN THÊM DỮ LIỆU PB + CV

```

Code

# =====
# RESET FORM
# =====
def reset_form():
    global pb_mode, selected_pb_id
    entry_pb_name.delete(0, tk.END)
    entry_pb_desc.delete(0, tk.END)
    entry_cv_name.delete(0, tk.END)
    entry_cv_desc.delete(0, tk.END)
    pb_mode = ""
    selected_pb_id = None

# =====
# LOAD TREEVIEW
# =====
def load_tree(tab="all"):
    cur = conn.cursor()
    cur.execute("""
        SELECT pb.id, pb.ten_phongban, pb.mota,
               cv.ten_chucvu, cv.mota
        FROM phongban pb
        LEFT JOIN chucvu cv ON pb.id = cv.id_phongban
        ORDER BY pb.id
    """)
    rows = cur.fetchall()
    cur.close()

target_tree = tree_pb_all if tab == "all" else tree_pb_crud
target_tree.delete(*target_tree.get_children())

```

```

        for stt, r in enumerate(rows, start=1):
            target_tree.insert("", "end", values=(stt, r[0], r[1], r[2],
r[3], r[4]))

```

```

# =====
# CRUD PB + CV
# =====
def add_pb():
    tenpb = entry_pb_name.get().strip()
    mota = entry_pb_desc.get().strip()
    tencv = entry_cv_name.get().strip()
    motacv = entry_cv_desc.get().strip()
    if not tenpb:
        messagebox.showerror("Lỗi", "Tên PB không được để trống!")
    return

```

```

    cur = conn.cursor()
    cur.execute("INSERT INTO phongban (ten_phongban, mota) VALUES
(%s, %s)", (tenpb, mota))
    conn.commit()
    pb_id = cur.lastrowid
    if tencv:
        cur.execute("INSERT INTO chucvu (ten_chucvu, mota, id_phongban)
VALUES (%s,%s,%s)",
                    (tencv, motacv, pb_id))
        conn.commit()
    messagebox.showinfo("Thành công", "Đã thêm PB + CV!")
    reset_form()
    load_tree(tab="crud")
    load_tree(tab="all")
    load_pb_to_combobox() # Cập nhật combobox NV/LX

```

```

def edit_pb():
    global pb_mode, selected_pb_id
    selected = tree_pb_crud.focus()
    if not selected:
        messagebox.showwarning("Chú ý", "Chọn dòng để sửa!")
        return
    values = tree_pb_crud.item(selected, "values")
    selected_pb_id = values[1]
    pb_mode = "edit"

```

```

entry_pb_name.delete(0, tk.END)
entry_pb_desc.delete(0, tk.END)
entry_cv_name.delete(0, tk.END)
entry_cv_desc.delete(0, tk.END)

```

```

entry_pb_name.insert(0, values[2])
entry_pb_desc.insert(0, values[3])

```

```
entry_cv_name.insert(0, values[4])
entry_cv_desc.insert(0, values[5])
```

```
def save_pb():
    global pb_mode, selected_pb_id
    if pb_mode != "edit" or not selected_pb_id:
        messagebox.showwarning("Chú ý", "Chọn dòng để sửa trước khi lưu!")
    return
```

```
tenpb = entry_pb_name.get().strip()
mota = entry_pb_desc.get().strip()
tencv = entry_cv_name.get().strip()
motacv = entry_cv_desc.get().strip()
if not tenpb:
    messagebox.showerror("Lỗi", "Tên PB không được để trống!")
return
```

```
cur = conn.cursor()
cur.execute("UPDATE phongban SET ten_phongban=%s, mota=%s WHERE id=%s", (tenpb, mota, selected_pb_id))
cur.execute("SELECT id FROM chucvu WHERE id_phongban=%s", (selected_pb_id,))
res = cur.fetchone()
if tencv:
    if res:
        cur.execute("UPDATE chucvu SET ten_chucvu=%s, mota=%s WHERE id=%s", (tencv, motacv, res[0]))
    else:
        cur.execute("INSERT INTO chucvu (ten_chucvu, mota, id_phongban) VALUES (%s,%s,%s)",
                    (tencv, motacv, selected_pb_id))
conn.commit()
messagebox.showinfo("Thành công", "Đã cập nhật PB + CV!")
reset_form()
load_tree(tab="crud")
load_tree(tab="all")
load_pb_to_combobox() # Cập nhật combobox NV/LX
```

```
def delete_pb():
    selected = tree_pb_crud.focus()
    if not selected:
        messagebox.showwarning("Chú ý", "Chọn dòng để xóa!")
        return
    values = tree_pb_crud.item(selected, "values")
    id_pb = values[1]
    if not messagebox.askyesno("Xác nhận", f"Xóa PB {id_pb}?"):
        return
    cur = conn.cursor()
```

```

cur.execute("DELETE FROM chucvu WHERE id_phongban=%s", (id_pb,))
cur.execute("DELETE FROM phongban WHERE id=%s", (id_pb,))
conn.commit()
messagebox.showinfo("Thành công", "Đã xóa PB!")
reset_form()
load_tree(tab="crud")
load_tree(tab="all")
load_pb_to_combobox() # Cập nhật combobox NV/LX

```

```

# =====
# Double-click chọn dòng CRUD
# =====
def on_crud_double_click(event):
    selected = tree_pb_crud.focus()
    if not selected:
        return
    values = tree_pb_crud.item(selected, "values")
    entry_pb_name.delete(0, tk.END)
    entry_pb_desc.delete(0, tk.END)
    entry_cv_name.delete(0, tk.END)
    entry_cv_desc.delete(0, tk.END)

```

```

entry_pb_name.insert(0, values[2])
entry_pb_desc.insert(0, values[3])
entry_cv_name.insert(0, values[4])
entry_cv_desc.insert(0, values[5])

```

```

global pb_mode, selected_pb_id
pb_mode = "edit"
selected_pb_id = values[1]

```

```

tree_pb_crud.bind("<Double-1>", on_crud_double_click)

```

```

# =====
# Cập nhật PB + CV cho combobox NV/LX
# =====
def load_pb_to_combobox():
    try:
        cur = conn.cursor()
        cur.execute("SELECT ten_phongban FROM phongban ORDER BY
ten_phongban")
        pb_list = [r[0] for r in cur.fetchall()]
        try:
            cb_phongban['values'] = pb_list
        except NameError:
            pass
        cur.execute("SELECT ten_chucvu FROM chucvu ORDER BY ten_chucvu")
        cv_list = [r[0] for r in cur.fetchall()]
        try:

```

```

        cb_chucvu['values'] = cv_list
    except NameError:
        pass
    cur.close()
except Exception as e:
    messagebox.showerror("Lỗi", f"Không tải được danh sách PB/CV: {e}")
# =====
# RESET FORM
# =====
def reset_form():
    global pb_mode, selected_pb_id
    entry_pb_name.delete(0, tk.END)
    entry_pb_desc.delete(0, tk.END)
    entry_cv_name.delete(0, tk.END)
    entry_cv_desc.delete(0, tk.END)
    pb_mode = ""
    selected_pb_id = None

# =====
# LOAD TREEVIEW
# =====
def load_tree(tab="all"):
    cur = conn.cursor()
    cur.execute("""
        SELECT pb.id, pb.ten_phongban, pb.mota,
               cv.ten_chucvu, cv.mota
        FROM phongban pb
        LEFT JOIN chucvu cv ON pb.id = cv.id_phongban
        ORDER BY pb.id
    """)
    rows = cur.fetchall()
    cur.close()

```

```

target_tree = tree_pb_all if tab == "all" else tree_pb_crud
target_tree.delete(*target_tree.get_children())

```

```

for stt, r in enumerate(rows, start=1):
    target_tree.insert("", "end", values=(stt, r[0], r[1], r[2],
r[3], r[4]))

```

```

# =====
# CRUD PB + CV
# =====
def add_pb():
    tenpb = entry_pb_name.get().strip()
    mota = entry_pb_desc.get().strip()
    tencv = entry_cv_name.get().strip()
    motacv = entry_cv_desc.get().strip()

```

```

if not tenpb:
    messagebox.showerror("Lỗi", "Tên PB không được để trống!")
    return

```

```

cur = conn.cursor()
cur.execute("INSERT INTO phongban (ten_phongban, mota) VALUES
(%s, %s)", (tenpb, mota))
conn.commit()
pb_id = cur.lastrowid
if tencv:
    cur.execute("INSERT INTO chucvu (ten_chucvu, mota, id_phongban)
VALUES (%s,%s,%s)",
                (tencv, motacv, pb_id))
    conn.commit()
messagebox.showinfo("Thành công", "Đã thêm PB + CV!")
reset_form()
load_tree(tab="crud")
load_tree(tab="all")
load_pb_to_combobox() # Cập nhật combobox NV/LX

```

```

def edit_pb():
    global pb_mode, selected_pb_id
    selected = tree_pb_crud.focus()
    if not selected:
        messagebox.showwarning("Chú ý", "Chọn dòng để sửa!")
        return
    values = tree_pb_crud.item(selected, "values")
    selected_pb_id = values[1]
    pb_mode = "edit"

```

```

entry_pb_name.delete(0, tk.END)
entry_pb_desc.delete(0, tk.END)
entry_cv_name.delete(0, tk.END)
entry_cv_desc.delete(0, tk.END)

```

```

entry_pb_name.insert(0, values[2])
entry_pb_desc.insert(0, values[3])
entry_cv_name.insert(0, values[4])
entry_cv_desc.insert(0, values[5])

```

```

def save_pb():
    global pb_mode, selected_pb_id
    if pb_mode != "edit" or not selected_pb_id:
        messagebox.showwarning("Chú ý", "Chọn dòng để sửa trước khi
lưu!")
    return

```

```

tenpb = entry_pb_name.get().strip()
mota = entry_pb_desc.get().strip()

```

```

tencv = entry_cv_name.get().strip()
motacv = entry_cv_desc.get().strip()
if not tenpb:
    messagebox.showerror("Lỗi", "Tên PB không được để trống!")
    return

```

```

cur = conn.cursor()
cur.execute("UPDATE phongban SET ten_phongban=%s, mota=%s WHERE
id=%s", (tenpb, mota, selected_pb_id))
cur.execute("SELECT id FROM chucvu WHERE id_phongban=%s",
(selected_pb_id,))
res = cur.fetchone()
if tencv:
    if res:
        cur.execute("UPDATE chucvu SET ten_chucvu=%s, mota=%s WHERE
id=%s", (tencv, motacv, res[0]))
    else:
        cur.execute("INSERT INTO chucvu (ten_chucvu, mota,
id_phongban) VALUES (%s,%s,%s)",
(tencv, motacv, selected_pb_id))
conn.commit()
messagebox.showinfo("Thành công", "Đã cập nhật PB + CV!")
reset_form()
load_tree(tab="crud")
load_tree(tab="all")
load_pb_to_combobox() # Cập nhật combobox NV/LX

```

```

def delete_pb():
    selected = tree_pb_crud.focus()
    if not selected:
        messagebox.showwarning("Chú ý", "Chọn dòng để xóa!")
        return
    values = tree_pb_crud.item(selected, "values")
    id_pb = values[1]
    if not messagebox.askyesno("Xác nhận", f"Xóa PB {id_pb}?"):
        return
    cur = conn.cursor()
    cur.execute("DELETE FROM chucvu WHERE id_phongban=%s", (id_pb,))
    cur.execute("DELETE FROM phongban WHERE id=%s", (id_pb,))
    conn.commit()
    messagebox.showinfo("Thành công", "Đã xóa PB!")
    reset_form()
    load_tree(tab="crud")
    load_tree(tab="all")
    load_pb_to_combobox() # Cập nhật combobox NV/LX

```

```

# =====
# Double-click chọn dòng CRUD
# =====

```



```
def on_crud_double_click(event):
    selected = tree_pb_crud.focus()
    if not selected:
        return
    values = tree_pb_crud.item(selected, "values")
    entry_pb_name.delete(0, tk.END)
    entry_pb_desc.delete(0, tk.END)
    entry_cv_name.delete(0, tk.END)
    entry_cv_desc.delete(0, tk.END)
```

```
entry_pb_name.insert(0, values[2])
entry_pb_desc.insert(0, values[3])
entry_cv_name.insert(0, values[4])
entry_cv_desc.insert(0, values[5])
```

```
global pb_mode, selected_pb_id
pb_mode = "edit"
selected_pb_id = values[1]
```

```
tree_pb_crud.bind("<Double-1>", on_crud_double_click)
```

```
# =====
# Cập nhật PB + CV cho combobox NV/LX
# =====
def load_pb_to_combobox():
    try:
        cur = conn.cursor()
        cur.execute("SELECT ten_phongban FROM phongban ORDER BY
ten_phongban")
        pb_list = [r[0] for r in cur.fetchall()]
        try:
            cb_phongban['values'] = pb_list
        except NameError:
            pass
        cur.execute("SELECT ten_chucvu FROM chucvu ORDER BY ten_chucvu")
        cv_list = [r[0] for r in cur.fetchall()]
        try:
            cb_chucvu['values'] = cv_list
        except NameError:
            pass
        cur.close()
    except Exception as e:
        messagebox.showerror("Lỗi", f"Không tải được danh sách PB/CV:
{e}")
```

Các chức năng chính

➤ *Thêm phòng ban + chức vụ*

Người dùng nhập thông tin và chọn nút “Thêm”. Hệ thống sẽ:

Ghi dữ liệu vào bảng phongban
Nếu có nhập chức vụ → ghi thêm vào bảng chucvu

➤ **Sửa phòng ban + chức vụ**

Khi chọn một dòng, hệ thống nạp dữ liệu lên form để sửa. Sau đó người dùng nhấn “Lưu” để cập nhật CSDL.

➤ **Xóa phòng ban**

Hệ thống hỏi xác nhận và thực hiện:

Xóa các chức vụ thuộc phòng ban
Xóa phòng ban

Cập nhật lại dữ liệu giao diện và combobox ở tab Nhân viên – Lái xe

➤ **Hủy thao tác**

Xóa nội dung form, đưa phần mềm về trạng thái chờ.

Bảng dữ liệu CRUD

Hiển thị danh sách phòng ban + chức vụ riêng cho phần sửa/xóa. Khi double-click, thông tin tự động điền vào form để chỉnh sửa.

4.1.5 Trình diễn Chức năng Quản lý Chuyển đi

Chức năng Quản lý Chuyển đi cho phép người dùng theo dõi, thêm mới, chỉnh sửa, xóa và cập nhật trạng thái của các chuyến xe trong hệ thống. Giao diện được thiết kế trực quan, dễ thao tác, gồm ba phần chính: Form nhập liệu, Nhóm nút chức năng, và Bảng dữ liệu (TreeView).

QUẢN LÝ CHUYỂN ĐI

Thông tin chuyển đi

Mã chuyển đi: Xe:

Lái xe: Điểm xuất phát:

Điểm đến: Ngày đi:

Ngày về: Trạng thái:

Thêm mới **Lưu** **Sửa** **Xóa** **Hủy** **Thoát**

Mã chuyển đi	Xe	Lái xe	Điểm xuất phát	Điểm đến	Ngày đi	Ngày về
1	101 - Xe tải A	201 - Lái xe 1	An Giang	TP.HCM	2025-12-01	2025-12-02
2	102 - Xe tải B	202 - Lái xe 2	Cần Thơ	Hà Nội	2025-12-03	2025-12-05
3	103 - Xe tải C	203 - Lái xe 3	Hải Phòng	Đà Nẵng	2025-12-06	2025-12-07

QUẢN LÝ CHUYẾN ĐI

Thông tin chuyến đi

Mã chuyến đi:

Lái xe:

Điểm đến:

Ngày về:

Xe:

Điểm xuất phát:

Ngày đi:

Trạng thái:

Thêm mới
Lưu
Sửa
Xóa
Hủy
Thoát

Mã chuyến đi	Xe	Lái xe	Điểm xuất phát	Điểm đến	Ngày đi	Ngày về
1	101 - Xe tải A	201 - Lái xe 1	An Giang	TP.HCM	2025-12-01	2025-12-02
2	102 - Xe tải B	202 - Lái xe 2	Cần Thơ	Hà Nội	2025-12-03	2025-12-05
3	103 - Xe tải C	203 - Lái xe 3	Hải Phòng	Đà Nẵng	2025-12-06	2025-12-07

HÌNH 11 : GIAO DIỆN QUẢN LÝ CHUYẾN ĐI

Chức năng đã đáp ứng đầy đủ các thông tin cần thiết để theo dõi một chuyến xe, bao gồm: mã chuyến đi, xe thực hiện, lái xe phụ trách, điểm xuất phát, điểm đến, ngày đi – ngày về và trạng thái chuyến.

Giao diện được chia thành ba vùng rõ ràng: khu vực nhập liệu, khu vực chức năng và khu vực danh sách, giúp người dùng thao tác trực quan, không bị rối thông tin. Việc sử dụng DateEntry và Combobox giúp giảm thao tác nhập tay và tăng độ chính xác.

Form cho phép nhập mới dữ liệu hoặc tự động đổ dữ liệu vào mỗi khi người dùng chọn 1 dòng trong bảng để sửa.

Người dùng dễ dàng thêm – sửa – xóa – cập nhật chuyến đi.

Hệ thống hỗ trợ các thao tác quản lý cơ bản một cách đơn giản:

- Thêm mới: người dùng chỉ cần điền form rồi nhấn Lưu để tạo chuyến mới.
- Sửa: khi chọn một chuyến trên bảng, toàn bộ dữ liệu được tự động đổ về form để người dùng chỉnh sửa.
- Xóa: hệ thống kiểm tra và yêu cầu xác nhận trước khi xóa để tránh thao tác nhầm.
- Cập nhật trạng thái: cho phép thay đổi trạng thái chuyến (Chưa khởi hành, Đang đi, Hoàn thành, Hủy).

Nhờ các nút chức năng bố trí hợp lý và màu sắc phân biệt, thao tác trở nên nhanh và chính xác hơn.

Giảm thiểu lỗi nhập liệu và tránh sai sót thủ công
 Hệ thống hạn chế tối đa nhập nhầm hoặc thiếu thông tin nhờ:
 Combobox cho xe và lái xe giúp chọn nhanh thay vì nhập bằng tay.
 DateEntry đảm bảo ngày đi – ngày về luôn đúng định dạng.
 Kiểm tra các trường bắt buộc trước khi lưu.
 Trạng thái được định nghĩa sẵn theo danh sách chuẩn, không bị sai khác chữ viết.
 Điều này giúp giảm khả năng nhập sai dữ liệu, tránh thất thoát thông tin và đảm bảo dữ liệu thống nhất.

4.1.6 Trình diễn Chức năng Quản lý Thống kê

Chức năng Thống kê Chuyên đi cho phép người dùng xem, lọc và xuất báo cáo danh sách các chuyến đi theo thời gian.
 Tính năng này hỗ trợ nhà quản lý dễ dàng đánh giá tần suất hoạt động của đội xe, hiệu quả làm việc của tài xế và theo dõi các chuyến đi theo ngày, tháng hoặc khoảng thời gian tùy chọn.

❖ Giao diện hệ thống

THỐNG KÊ CHUYẾN ĐI						
Bộ lọc thống kê						
Từ ngày:	2025-12-01	Đến ngày:	2025-12-11	Lọc dữ liệu	Xuất Excel	
Mã chuyến	Biển số xe	Tên lái xe	Ngày đi	Nơi đi	Nơi đến	Tình trạng
37	51A-12345	Tài xế A	2025-11-01	TP.HCM	Hà Nội	Chưa khởi hành
39	51A-12345	Tài xế A	2025-11-01	TP.HCM	Hà Nội	Chưa khởi hành
41	51A-12345	Tài xế A	2025-11-01	TP.HCM	Hà Nội	Chưa khởi hành
43	51A-12345	Tài xế A	2025-11-01	TP.HCM	Hà Nội	Chưa khởi hành
45	51A-12345	Tài xế A	2025-11-01	TP.HCM	Hà Nội	Chưa khởi hành
47	51A-12345	Tài xế A	2025-11-01	TP.HCM	Hà Nội	Chưa khởi hành
49	51A-12345	Tài xế A	2025-11-01	TP.HCM	Hà Nội	Chưa khởi hành
51	51A-12345	Tài xế A	2025-11-01	TP.HCM	Hà Nội	Chưa khởi hành
53	51A-12345	Tài xế A	2025-11-01	TP.HCM	Hà Nội	Chưa khởi hành
55	51A-12345	Tài xế A	2025-11-01	TP.HCM	Hà Nội	Chưa khởi hành
57	51A-12345	Tài xế A	2025-11-01	TP.HCM	Hà Nội	Chưa khởi hành
38	51B-54321	Tài xế B	2025-11-05	Cần Thơ	Đà Nẵng	Hoàn thành
40	51B-54321	Tài xế B	2025-11-05	Cần Thơ	Đà Nẵng	Hoàn thành
42	51B-54321	Tài xế B	2025-11-05	Cần Thơ	Đà Nẵng	Hoàn thành

HÌNH 12 : GIAO DIỆN THÔNG KÊ CHUYẾN ĐI

❖ Chức năng

Giao diện của tab Thống kê được chia thành 3 phần chính:

Bộ lọc theo ngày

- Cho phép chọn **Từ ngày** và **Đến ngày** bằng DateEntry (lịch).

- Cung cấp nút “**Lọc dữ liệu**” để lọc nhanh các chuyến nằm trong khoảng thời gian đã chọn.

Lọc dữ liệu theo khoảng thời gian

Người dùng chọn Từ ngày – Đến ngày, sau đó nhấn Lọc dữ liệu.
Hệ thống sẽ:

Kiểm tra dữ liệu hợp lệ.

Gửi câu lệnh SQL:

SELECT ...FROM chuyendiWHERE ngay_di BETWEEN ... AND ...

Cảnh báo nếu người dùng để trống ngày bắt đầu hoặc ngày kết thúc.

Bảng TreeView hiển thị kết quả

Danh sách các chuyến đi được hiển thị rõ ràng với các cột thông tin:

| Mã chuyến | Biển số xe | Tên lái xe | Ngày đi | Nơi đi | Nơi đến | Tình trạng |

TreeView hỗ trợ:

- Tự động cập nhật sau khi lọc.
- Có thanh cuộn dọc và ngang.
- Hiển thị nội dung rõ ràng, dễ theo dõi.

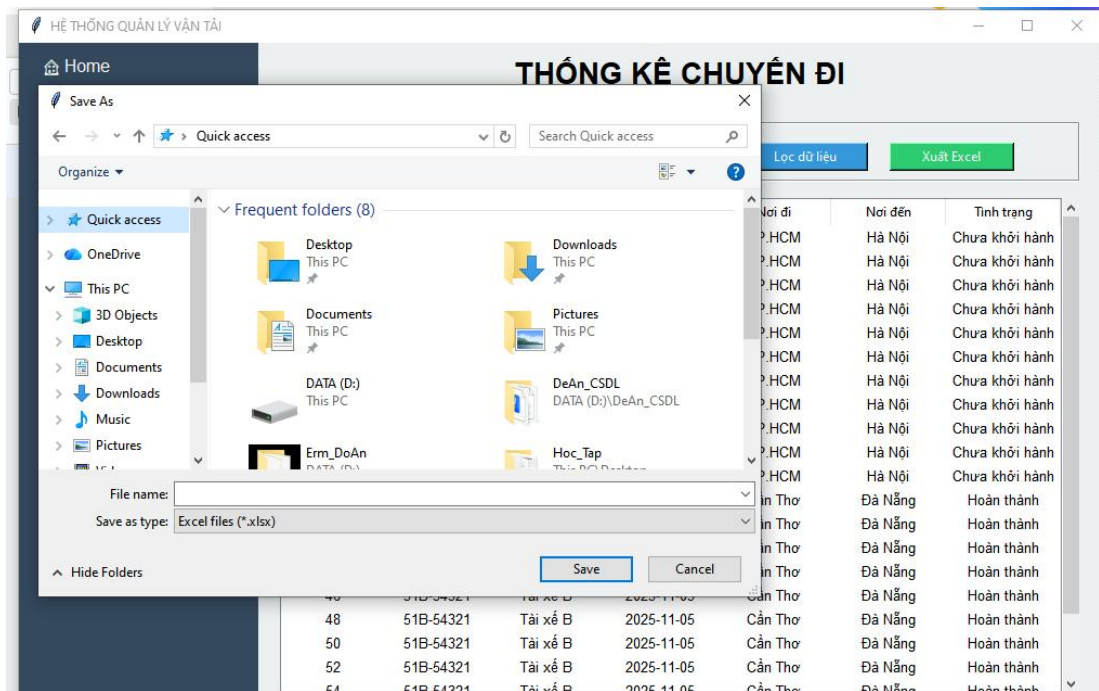
Chức năng xuất Excel

Người dùng có thể xuất dữ liệu thống kê ra file .xlsx để:

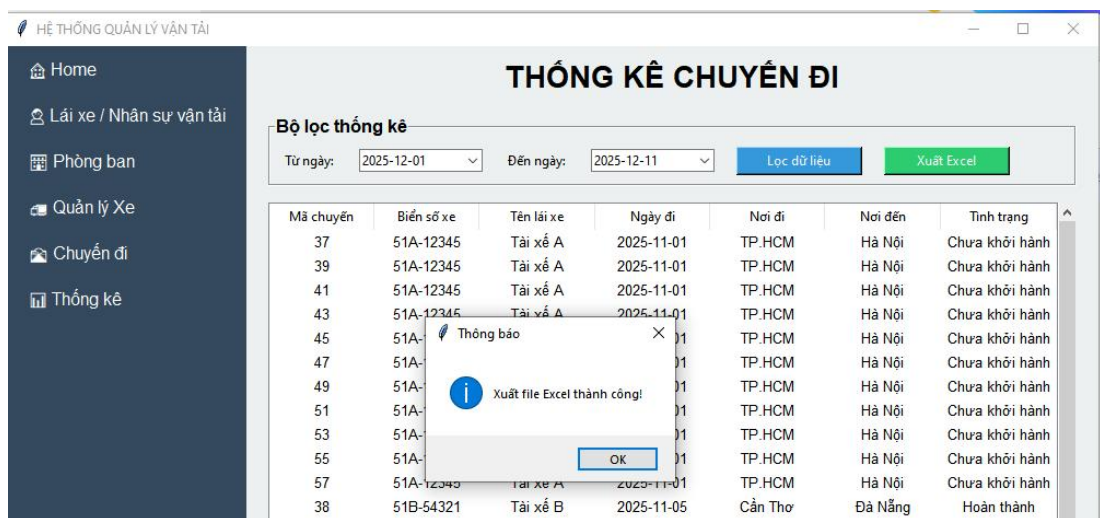
Báo cáo nội bộ.

- Lưu trữ.
- In ấn
- Phân tích thêm bằng Excel.

Hệ thống kiểm tra dữ liệu trước khi xuất để tránh file rỗng.



Hình 13 : XUẤT FILE EXCEL



Hình 14 : GIA DIỆN XUẤT FILE THÀNH CÔNG

❖ Kết quả đạt được

- Dễ dàng theo dõi chuyển đi theo thời gian → hỗ trợ quản lý, báo cáo
- Giảm thao tác thủ công, không cần lọc bằng tay.
- Xuất Excel tiện lợi, phù hợp cho doanh nghiệp. Giao diện trực quan, thao tác đơn giản.
- Tự động tải dữ liệu khi mở tab, giúp truy cập nhanh hơn.

CHƯƠNG 6 :KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

6.1 Đánh giá Kết quả Đạt được

Sau quá trình xây dựng và hoàn thiện hệ thống “Quản lý vận tải”, nhóm đã đạt được nhiều kết quả tích cực cả về mặt kỹ thuật lẫn nghiệp vụ. Cụ thể như sau:

Hoàn thiện các chức năng nghiệp vụ cốt lõi:

Hệ thống đã triển khai đầy đủ các chức năng cần thiết trong quản lý vận tải:

- ✓ Quản lý nhân viên – lái xe.
- ✓ Quản lý phòng ban – chức vụ.
- ✓ Quản lý xe và thông tin kỹ thuật của phương tiện.
- ✓ Quản lý chuyến đi, lịch trình, tình trạng hoạt động.
- ✓ Chức năng thống kê – báo cáo theo thời gian.

Hỗ trợ xuất dữ liệu Excel phục vụ báo cáo nhanh. Các chức năng hoạt động ổn định, đúng luồng nghiệp vụ mà doanh nghiệp đang sử dụng. Giao diện trực quan – dễ sử dụng.

Các tab chức năng được bố trí rõ ràng theo từng nhóm nghiệp vụ. TreeView hiển thị dữ liệu đầy đủ, dễ đọc. Các form nhập liệu thân thiện, hạn chế sai sót.

=> Điều này giúp người dùng dễ thao tác ngay cả khi không rành tin học.

Hệ thống đảm bảo tính chính xác và nhất quán dữ liệu:

- ✓ Dữ liệu được quản lý bằng MySQL, hạn chế mất mát.
- ✓ Các chức năng thêm – sửa – xóa được kiểm soát bằng kiểm tra hợp lệ.
- ✓ Tránh lỗi trùng khóa, thiếu thông tin hoặc nhập sai dữ liệu.
- ✓ Hệ thống đáp ứng tốt yêu cầu về an toàn dữ liệu trong doanh nghiệp vận tải.

Hỗ trợ công tác theo dõi – giám sát hiệu quả Chức năng thống kê chuyến đi theo ngày giúp:

- ✓ Nắm bắt nhanh số lượng chuyến trong từng giai đoạn.
- ✓ Đánh giá hiệu suất hoạt động của lái xe và phương tiện.

- ✓ Phục vụ công tác báo cáo, tổng hợp cuối tháng.
- ✓ Tính năng này giúp nâng cao hiệu quả quản lý thực tế.

Khả năng mở rộng và nâng cấp trong tương lai. Cấu trúc code theo dạng module và tách tab giúp:

- ✓ Dễ dàng bổ sung các chức năng mới.
- ✓ Dễ chỉnh sửa mà không ảnh hưởng toàn hệ thống.
- ✓ Có thể nâng cấp lên Web/App trong tương lai.
- ✓ Hệ thống có tính linh hoạt cao.

Kỹ năng của nhóm được nâng cao rõ rệt. Trong quá trình làm đồ án, em đã học được:

- ✓ Làm việc với MySQL, Python và Tkinter. Quản lý dự án, phân công công việc.
- ✓ Xử lý lỗi, đọc log, fix bug thực tế.
- ✓ Thiết kế giao diện ứng dụng. Hiểu nghiệp vụ vận tải thực tế. Đây là những kỹ năng quan trọng cho công việc sau này.

6.2 Hướng phát triển trong tương lai

Để chuyển đổi ứng dụng Quản lý Xe và Lái xe từ nền tảng desktop cơ bản (Python/Tkinter/MySQL) thành một hệ thống quản lý vận tải chuyên nghiệp, đa người dùng, và toàn diện, đề tài có thể phát triển theo ba hướng chiến lược sau:

Cải tiến Nền tảng Công nghệ

Chuyển đổi sang Web/Cloud: Nâng cấp từ Tkinter sang các Framework Web Python (Flask/Django) và triển khai trên Cloud để cho phép truy cập tập trung, đa thiết bị.

Tối ưu hóa CSDL & Dữ liệu: Cải thiện tốc độ truy vấn bằng cách thêm Chỉ mục (Indexes) và sử dụng CSDL NoSQL (ví dụ: Firestore) cho các dữ liệu phi cấu trúc (log, hồ sơ bằng lái).

Áp dụng API và Microservices: Tái cấu trúc logic nghiệp vụ thành các API RESTful để dễ dàng tích hợp với các hệ thống bên ngoài (kế toán, GPS).

Mở rộng Chức năng Nghiệp vụ Chuyên sâu

Quản lý Vận hành và Chi phí: Phát triển module theo dõi chi tiết Lịch sử Bảo trì và quản lý các loại Chi phí Vận hành (nhiên liệu, phí phạt), giúp tính toán TCO (Tổng chi phí sở hữu) của phương tiện.

Cảnh báo Tự động: Xây dựng cơ chế gửi thông báo (email/trong ứng dụng) khi xe/bằng lái sắp đến hạn bảo trì, đăng kiểm hoặc hết hạn.

Phân công Tối ưu & GPS: Phát triển thuật toán đề xuất phân công dựa trên tiêu chí nghiệp vụ và tích hợp API Bản đồ/GPS để tính toán tuyến đường và lưu lịch sử di chuyển.

Quản lý Tài chính: Tích hợp tính năng tính toán Lương thưởng cho lái xe dựa trên hiệu suất công việc.

Cải thiện Trải nghiệm & Bảo mật (UX/UI & Security)

Thiết kế Giao diện Nâng cao: Nâng cấp giao diện từ Tkinter lên CustomTkinter/PyQt để đạt tính thẩm mỹ và khả năng phản hồi tốt hơn. Trực quan hóa Dữ liệu: Sử dụng thư viện biểu đồ (Matplotlib/Plotly) để hiển thị báo cáo chi phí và tần suất sử dụng xe một cách sinh động.

Hệ thống Phân quyền & Bảo mật: Triển khai Hệ thống Phân quyền chi tiết (Admin, Nhân viên điều hành) và sử dụng cơ chế Mã hóa Mật khẩu Mạnh (Hashing) cùng giao thức SSL/TLS để bảo vệ dữ liệu.

TÀI LIỆU THAM KHẢO

Nguyễn Văn Vy, Phân tích và thiết kế hệ thống thông tin, NXB Thống Kê, 2020.

Trần Thị Thanh, Cơ sở dữ liệu – Lý thuyết và ứng dụng, NXB Khoa học & Kỹ thuật, 2018.

Giáo trình Nhập môn Công nghệ phần mềm, Trường Đại học An Giang – ĐHQG TP.HCM, 2022.

Giáo trình Hệ quản trị cơ sở dữ liệu, Trường Đại học An Giang, 2021.

