TRƯỜNG ĐẠI HỌC AN GIANG
KHOA CÔNG NGHỆ THÔNG TIN

BÁO CÁO ĐỒ ÁN

Chuyên đề Python (COS525)

XÂY DỰNG ỨNG DỤNG QUẢN LÝ CỬA HÀNG XE MÁY VỚI PYTHON, TKINTER VÀ MYSQL

Giảng viên hướng dẫn: Ths. Nguyễn Ngọc Minh

Sinh viên thực hiện: DTH235690.Dương Nguyễn Ngọc Linh.DH24TH2_NhómTH02_TổTH01
DTH235706.Đỗ Thị Kim Ngọc.DH24TH2_NhómTH02_TổTH01

AN GIANG, 09-2025

BẢNG PHÂN CHIA CÔNG VIỆC		
Họ và Tên	**Tỉ lệ %**	**Công việc phụ trách**
Dương Nguyễn Ngọc Linh	50%	Viết báo cóa word, xây dựng cơ sở dữ liệu MySQL, kiểm thử.
Đỗ Thị Kim Ngọc	50%	Thiết kế giao diện toàn hệ thống, lập trình các form trong Python.

Mục Lục

I. Phần 1: Giới thiệu đề tài 2

II. Phần 2: Mục tiêu và phạm vi 3

1. Mục tiêu 3

2. Phạm vi 4

III. Phần 3: Phân tích hệ thống 4

1. Khách thể sử dụng 4

2. Các chức năng chính của hệ thống 4

IV. Phần 4: Thiết kế cơ sở dữ liệu 7

1. Sơ đồ bảng 7

2. Trigger tự tạo mã 9

V. Phần 5: Thiết kế giao diện (GUI) 11

VI. Phần 6: Hiện thực chương trình (CODE CHÍNH) 15

1. Kết nối CSDL – database.py 15

2. CRUD Xe máy – form_XeMay.py 16

3. CRUD Khách hàng – form_KhachHang.py 22

4. CRUD Nhân viên – form_NhanVien.py 26

5. CRUD Hóa đơn – form_HoaDon.py 32

6. CRUD Chi tiết hóa đơn – form_CTHoaDon.py 40

7. CRUD Lịch sử giá – form_LichSuGia.py 47

8. Đăng nhập phân quyền – form_login.py 49

9. CRUD file chính– form_Main.py 52

VII. Phần 7: Kết luận 55

VIII. Phần 8: Hướng phát triển 55

IX. Phần 9: Tài liệu tham khảo 55

1. Kết luận và hướng dẫn phát triển 56

VIẾT BÁO CÁO ĐỒ ÁN MÔN PYTHON

I. Phần 1: Giới thiệu đề tài

- Trong thời đại công nghệ số, việc quản lý bằng phương pháp thủ công như dùng giấy tờ hoặc file Excel đã bộc lộ nhiều hạn chế. Đặc biệt đối với các cửa hàng xe máy – nơi có số lượng sản phẩm lớn, nhiều loại xe, nhiều hóa đơn mỗi ngày – thì việc quản lý thủ công dễ dẫn đến sai sót, thất lạc thông tin, không thể thống kê nhanh và mất nhiều thời gian.
- Xuất phát từ thực tế đó, nhóm nghiên cứu quyết định thực hiện đề tài **Xây dựng ứng dụng quản lý cửa hàng xe máy bằng Python và MySQL**. Ứng dụng hỗ trợ quản lý xe, khách hàng, nhân viên, hóa đơn và chi tiết hóa đơn, giúp chủ cửa hàng theo dõi hoạt động kinh doanh thuận tiện hơn. Mọi thao tác CRUD (Create – Read – Update – Delete) đều được hỗ trợ.
- Đề tài được xây dựng bằng:
    + **Python**: xử lý nghiệp vụ và giao diện Tkinter
    + **Tkinter**: tạo giao diện người dùng
    + **MySQL**: lưu trữ dữ liệu
    + **mysql-connector-python**: kết nối Python – MySQL

II. Phần 2: Mục tiêu và phạm vi

1. Mục tiêu

- Xây dựng ứng dụng quản lý cửa hàng xe máy có giao diện thân thiện, dễ sử dụng.
- Thực hiện đầy đủ các chức năng CRUD cho:
    + Xe máy
    + Khách hàng
    + Nhân viên

+ Hóa đơn

+ Chi tiết hóa đơn

- Xây dựng hệ thống đăng nhập phân quyền cơ bản.
- Sử dụng Trigger MySQL để tự động tạo mã Xe, Khách hàng, Hóa đơn.
- Đảm bảo dữ liệu được lưu trữ chính xác trong MySQL.
- Tự động tạo mã xe, mã khách hàng, mã nhân viên, mã hóa đơn thông qua Trigger.

2. Phạm vi

- Xây dựng ứng dụng chạy trên máy tính Windows.
- Không phát triển mobile/web.
- Chỉ tập trung vào quản lý nội bộ (không có bán hàng trực tuyến).
- Không áp dụng mã hóa nâng cao hoặc bảo mật tài khoản chuyên sâu.
- Không áp dụng mã hóa mật khẩu nâng cao (mật khẩu plaintext).

III. Phần 3: Phân tích hệ thống

1. Khách thể sử dụng

- Nhân viên bán hàng:
  + Lập hóa đơn
  + Xem danh sách xe còn hàng
  + Tìm kiếm xe và khách hàng
  + Không được phép sửa nhân viên hoặc xóa dữ liệu nhạy cảm
- Quản lý / chủ cửa hàng:
  + Quản lý tất cả chức năng
  + Chỉnh sửa dữ liệu nhân viên
  + Xem các báo cáo đơn giản

2. Các chức năng chính của hệ thống

a. Module Đăng nhập (form_login.py)

- Nhập tài khoản / mật khẩu
- Kiểm tra thông tin trong bảng TaiKhoan
- Phân quyền role: Admin / Nhân viên
- Nếu đúng → mở giao diện chính
- Nếu sai → thông báo lỗi

b. Module Xe máy (form_XeMay.py)

Chức năng:

- Thêm tạm xe mới
- Lưu danh sách thêm mới vào MySQL
- Cập nhật thông tin xe
- Xóa xe
- Tìm kiếm theo tên xe, hãng xe, màu
- Theo dõi số lượng tồn
- Tự động sinh mã XExxxx bằng Trigger MySQL

Các trường thông tin xe:

- MaXe (readonly – tự sinh)
- TenXe
- HangXe
- MauXe
- GiaXe
- SoLuong

Cấu trúc giao diện (dựa theo mã):

- Ô input nhập xe
- Khu tìm kiếm
- TreeView hiển thị danh sách
- Nhóm nút chức năng:
    + Thêm tạm
    + Lưu
    + Cập nhật

+ Xóa
+ Hủy
+ Quay lại

c. Module Khách hàng (form_KhachHang.py)

Chức năng:

- Tìm kiếm theo tên, SDT, email, địa chỉ
- Thêm khách hàng
- Lưu
- Sửa khách hàng
- Xóa khách hàng
- Quay lại

Trường dữ liệu:

- MaKH (auto – readonly)
- TenKH
- DiaChi
- SDT
- Email

Điểm nổi bật:

- Dùng Auto (DB) làm placeholder cho mã tự động
- Hỗ trợ tìm kiếm đa tiêu chí
- Có temp_data để thêm nhiều dòng trước khi lưu

d. Module Nhân viên (form_NhanVien.py)

Chức năng:

- Thêm / Sửa / Xóa nhân viên
- Tìm kiếm nhân viên

Trường dữu liệu:

- MaNV
- HoTen
- GioiTinh

- NgaySinh
- SDT
- DiaChi
- ChucVu
- Luong

e. Module Hóa đơn (form_HoaDon.py)

Chức năng:

- Tạo hóa đơn mới
- Lấy mã khách hàng thông qua combobox hoặc input
- Tự động xác định mã nhân viên theo người đăng nhập
- Tạo hóa đơn → mở form Chi tiết hóa đơn
- Tìm kiếm hóa đơn
- Xóa hóa đơn

Trường dữ liệu:

- MaHD
- NgayBan
- MaKH
- MaNV
- TongTien (tính từ CTHD)
- GhiChu

f. Module Chi tiết hóa đơn (form_CTHoaDon.py)

Chức năng:

- Chọn xe bán
- Nhập số lượng
- Tự nhân giá
- Trừ số lượng tồn kho
- Lưu chi tiết hóa đơn
- Tải lại chi tiết hóa đơn

g. Module Lịch sử giá (form_LichSuGia.py)

- Lưu lịch sử thay đổi giá xe
- Hỗ trợ cập nhật giá mới
- Tự động ghi vào bảng LichSuGia
- Hiển thị bảng lịch sử theo Xe

IV. Phần 4: Thiết kế cơ sở dữ liệu

1. Sơ đồ bảng

Bảng 1: Bảng XeMay

MaXe	TenXe	HangXe	MauXe	GiaXe	SoLuong	CreatedAt
XE0001	Vision 2024	Honda	Đỏ	40000000.00	10	2025-11-18 20:27:34
XE0002	Sirius RC	Yamaha	Đen	25000000.00	15	2025-11-18 20:27:34
XE0003	Air Blade 160	Honda	Trắng	30000000.00	8	2025-11-18 20:27:34
XE0004	Exciter 155	Yamaha	Xanh	60000000.00	12	2025-11-18 20:27:34
XE0005	Future fi125	Honda	Nâu	39000000.00	5	2025-11-18 22:44:47
NULL	NULL	NULL	NULL	NULL	NULL	NULL

Bảng 2: Bảng KhachHang

MaKH	TenKH	DiaChi	SDT	Email	CreatedAt
KH0001	Phạm Huỳnh Trúc Anh	Long Xuyên	0911222333	anhpham@gmail.com	2025-11-18 20:28:12
KH0002	Bùi Thị Mén	An Giang	0909888777	maibui@gmail.com	2025-11-18 20:28:12
KH0003	Trần Trí Tường	Đồng Tháp	0977665544	tuongtran@gmail.com	2025-11-18 20:28:12
KH0004	Trần Thị Trúc Giang	Cần Thơ	0989797556	giangtran@gmail.com	2025-11-18 20:28:12
KH0005	Huỳnh Quốc Khánh	Cần Thơ	0353930698	khanhhuynh@gmail.com	2025-11-18 20:28:12
KH0006	Nguyễn Tấn Lộc	TP HCM	0983600990	locnguyen@gmail.com	2025-11-18 20:28:12
NULL	NULL	NULL	NULL	NULL	NULL

Bảng 3: Bảng NhanVien

MaNV	HoTen	GioiTinh	NgaySinh	SDT	DiaChi	ChucVu	Luong	CreatedAt
NV0001	Dương Hồng Ngọc	Nữ	1996-05-12	0912345678	An Giang	Quản lý	15000000.00	2025-11-18 20:27:57
NV0002	Huỳnh Phú Quý	Nam	2000-08-20	0987654321	Cần Thơ	Nhân viên bán hàng	9000000.00	2025-11-18 20:27:57
NV0003	Dương Hồng Quý	Nữ	1994-08-01	0901112233	Long Xuyên	Kế toán	10000000.00	2025-11-18 20:27:57
NV0004	Lý Thị Thu Ngọc	Nữ	1996-05-27	0398736567	Đồng Tháp	Nhân viên bán hàng	9000000.00	2025-11-18 20:27:57
NV0005	Trương Chí Khan	Nam	2004-09-07	0354255699	Kiêng Giang	Nhân viên bán hàng	9000000.00	2025-11-18 20:27:57
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

Bảng 4: Bảng HoaDon

	MaHD	NgayBan	MaKH	MaNV	TongTien	GhiChu	CreatedAt
▶	HD0001	2025-11-02	KH0002	NV0002	52000000.00	NULL	2025-11-18 20:28:19
	HD0002	2025-11-01	KH0001	NV0001	34000000.00	NULL	2025-11-18 20:28:19
*	NULL	NULL	NULL	NULL	NULL	NULL	NULL

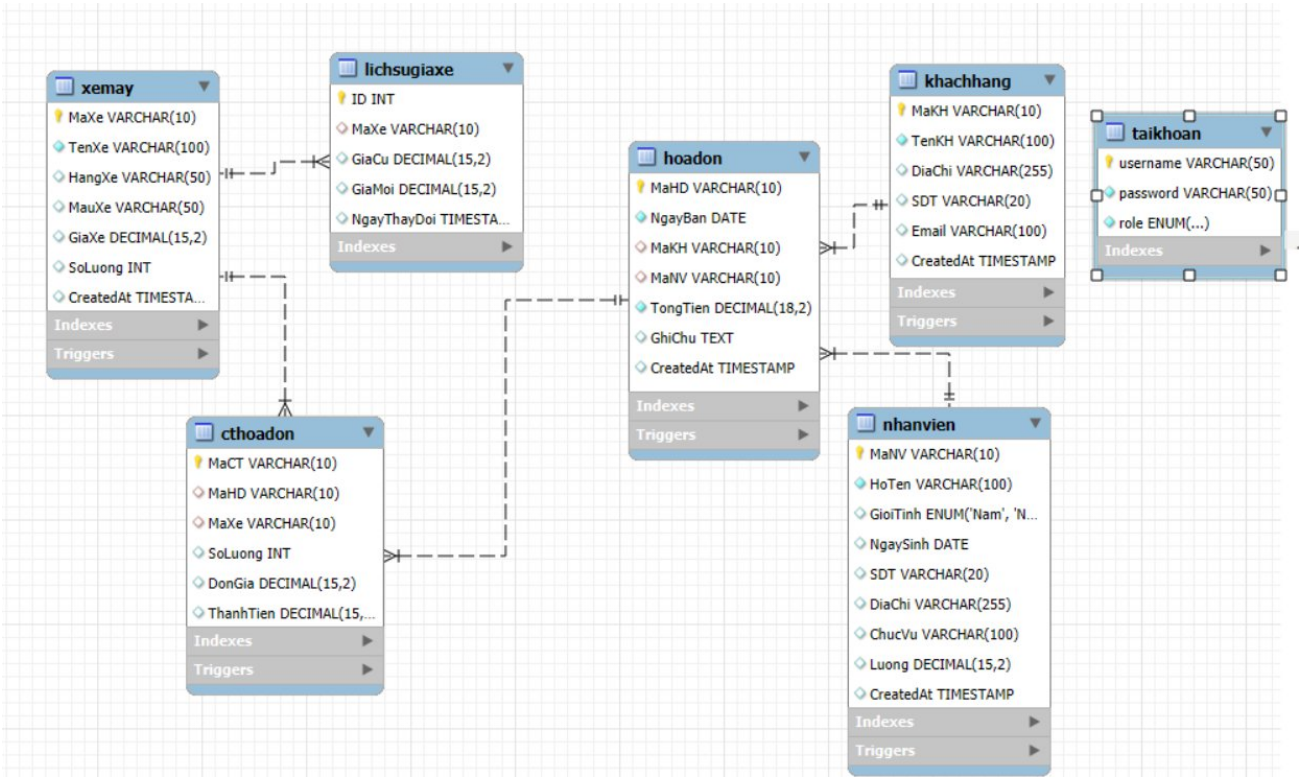
Bảng 5: Bảng CTHoaDon

	MaCT	MaHD	MaXe	SoLuong	DonGia	ThanhTien
▶	CT0001	HD0001	XE0001	1	34000000.00	34000000.00
	CT0002	HD0002	XE0003	1	52000000.00	52000000.00
*	NULL	NULL	NULL	NULL	NULL	NULL

Bảng 6: Bảng TaiKhoan

	username	password	role
▶	admin	0909	Admin
	nhanvien	0000	Nhan_Vien
	NULL	NULL	NULL

Relasionship



2. Trigger tự tạo mã

- form_XeMay.py

```
DELIMITER $$
CREATE TRIGGER tg_XeMay_before_insert
BEFORE INSERT ON XeMay
FOR EACH ROW
BEGIN
  DECLARE max_id INT;

  SELECT COALESCE(MAX(CAST(SUBSTRING(MaXe, 3) AS UNSIGNED)), 0) + 1
  INTO max_id
  FROM XeMay;

  SET NEW.MaXe = CONCAT('XE', LPAD(max_id, 4, '0'));
END$$
DELIMITER ;
```

- form_NhanVien.py

```
DELIMITER $$
CREATE TRIGGER tg_NhanVien_before_insert
BEFORE INSERT ON NhanVien
FOR EACH ROW
BEGIN
  DECLARE max_id INT;

  SELECT COALESCE(MAX(CAST(SUBSTRING(MaNV, 3) AS UNSIGNED)), 0) + 1
  INTO max_id
  FROM NhanVien;

  SET NEW.MaNV = CONCAT('NV', LPAD(max_id, 4, '0'));
END$$
DELIMITER ;
```

- form_KhachHang.py

```
DELIMITER $$
CREATE TRIGGER tg_KhachHang_before_insert
BEFORE INSERT ON KhachHang
FOR EACH ROW
BEGIN
  DECLARE max_id INT;

  SELECT COALESCE(MAX(CAST(SUBSTRING(MaKH, 3) AS UNSIGNED)), 0) + 1
  INTO max_id
  FROM KhachHang;

  SET NEW.MaKH = CONCAT('KH', LPAD(max_id, 4, '0'));
```

```
        END$$
DELIMITER ;
```

- form_HoaDon.py

```
        DELIMITER $$
        CREATE TRIGGER tg_HoaDon_before_insert
        BEFORE INSERT ON HoaDon
        FOR EACH ROW
        BEGIN
          DECLARE max_id INT;

          SELECT COALESCE(MAX(CAST(SUBSTRING(MaHD, 3) AS UNSIGNED)), 0) + 1
          INTO max_id
          FROM HoaDon;

          SET NEW.MaHD = CONCAT(HD, LPAD(max_id, 4, '0'));
        END$$
DELIMITER ;
```

- form_CTHoaDon.py

```
        DELIMITER $$
        CREATE TRIGGER tg_CTHoaDon_before_insert
        BEFORE INSERT ON CTHoaDon
        FOR EACH ROW
        BEGIN
          DECLARE max_id INT;

          SELECT COALESCE(MAX(CAST(SUBSTRING(MaHD, 3) AS UNSIGNED)), 0) + 1
          INTO max_id
          FROM CTHoaDon;

          SET NEW.MaCT = CONCAT(CT, LPAD(max_id, 4, '0'));
        END$$
DELIMITER ;
```

V. Phần 5: Thiết kế giao diện (GUI)

Có các nút mở form:

- Giao diện form_XeMay.py:

- Giao diện form_NhanVien.py

- Giao diện form_KhachHang.py

- Giao diện form_HoaDon.py

- Giao diện form_CTHoaDon.py

- Giao diện form_login.py

VI. Phần 6: Hiện thực chương trình (CODE CHÍNH)

1. Kết nối CSDL – database.py

```
import mysql.connector
from mysql.connector import Error
def get_connection():
    try:
        connection = mysql.connector.connect(
            host='localhost',
            user='root',
            password='ngoclinh1707@',
            database='QLCuaHangXeMay'
        )
        if connection.is_connected():
            return connection
    except Error as e:
        print("✖ Lỗi kết nối MySQL:", e)
        return None
def close_connection(conn):
    """
    Đóng kết nối nếu nó đang mở.
    Hàm này đã được thêm vào để khắc phục lỗi Import
    """
    if conn and conn.is_connected():
        conn.close()
def connect_and_get_cursor():
    """
    Hàm hỗ trợ: Kết nối database và trả về (kết nối, cursor).
    Dùng cho các form con cần thao tác CSDL.
    """
    conn = get_connection()
    if conn:
        # Sử dụng dictionary=True để cursor trả về dữ liệu dưới dạng dictionary (có tên
cột)
        cursor = conn.cursor(dictionary=True)
        return conn, cursor
    return None, None
```

2. CRUD Xe máy – form_XeMay.py

```
import tkinter as tk
from tkinter import ttk, messagebox
# ===================== STYLE BUTTON  =====================
def style_button(btn, bg="#2196F3", fg="white"):
    btn.config(bg=bg,fg=fg,activebackground=bg, activeforeground=fg,relief="flat",bd=1)
def open_xe_form(root=None, conn=None):
    cursor = conn.cursor() if conn else None
```

```
    if root is not None:
        root.withdraw()

    win = tk.Toplevel(root) if root else tk.Tk()
    win.title("Quản lý Xe máy")
    win.geometry("1100x650")
    win.configure(bg="#AEEBFF")
    # ================== TIÊU ĐỀ ==================
    tk.Label(
        win, text="QUẢN LÝ XE MÁY",font=("Arial", 20, "bold"),fg="#003399",
bg="#AEEBFF").pack(pady=10)
    temp_data = []

        # ================== TÌM KIẾM ==================#
    search_frame = tk.Frame(win, bg="#AEEBFF")
    search_frame.pack(pady=5)

    tk.Label(search_frame, text="🔍 Tìm kiếm:", font=("Arial", 10,
"bold"),bg="#AEEBFF").grid(row=0, column=0, padx=5)

    search_var = tk.StringVar()
    tk.Entry(search_frame, textvariable=search_var, width=60).grid(row=0, column=1,
padx=5)
    # ---- NÚT TÌM  ----
    tk.Button(search_frame, text="Tìm", width=10, bg="#2196F3", fg="black",
command=lambda: load_data(search_var.get())).grid(row=0, column=2, padx=5)
    # ---- NÚT TẢI LẠI  ----
    tk.Button(search_frame, text="Tải lại", width=10,bg="#9E9E9E",
fg="black",command=lambda: (search_var.set(""), load_data())).grid(row=0, column=3,
padx=5)
    # ================== BẢNG XE ==================
    frame_ds = tk.LabelFrame( win, text="Danh sách xe máy", padx=10, pady=10,
font=("Arial", 11, "bold"), fg="#003366", bg="#AEEBFF")
    frame_ds.pack(padx=10, pady=10, fill="both", expand=True)

    scroll = tk.Scrollbar(frame_ds)
    scroll.pack(side=tk.RIGHT, fill=tk.Y)

    columns = ["MaXe", "TenXe", "HangXe", "MauXe", "GiaXe", "SoLuong", "CreatedAt"]
    tree = ttk.Treeview(frame_ds, columns=columns, show="headings",
yscrollcommand=scroll.set, height=12)
    for c in columns:
        tree.heading(c, text=c)
        tree.column(c, width=150)
    tree.pack(fill="both", expand=True)
```

```
    scroll.config(command=tree.yview)
    # ================== FORM XE ==================
    frame_form = tk.LabelFrame( win, text="Thông tin xe máy",padx=10, pady=10,
font=("Arial", 10, "bold"),bg="#AEEBFF")
    frame_form.pack(pady=10, padx=10, fill="x")
    entries = {}
    fields = ["MaXe", "TenXe", "HangXe", "MauXe", "GiaXe", "SoLuong"]
    for i, f in enumerate(fields):
        tk.Label(frame_form, text=f + ":", font=("Arial", 10), bg="#AEEBFF").grid(row=i // 3,
column=(i % 3) * 2, sticky="w", padx=6, pady=4)
        e = tk.Entry(frame_form, width=25)
        e.grid(row=i // 3, column=(i % 3) * 2 + 1, padx=6, pady=4)
        entries[f] = e
    # ====== MaXe readonly ======
    def set_auto_maxe():
        entries["MaXe"].config(state="normal")
        entries["MaXe"].delete(0, tk.END)
        entries["MaXe"].insert(0, "Auto (DB)")
        entries["MaXe"].config(state="readonly")
    set_auto_maxe()
    # ================== CLEAR FORM ==================
    def clear_form():
        for c in entries:
            entries[c].config(state="normal")
            entries[c].delete(0, tk.END)
        set_auto_maxe()
    # ================== LOAD DATA ==================
    def load_data(search=None):
        tree.delete(*tree.get_children())
        temp_data.clear()
        try:
            if search:
                like = f"%{search}%"
                cursor.execute("""
                    SELECT MaXe, TenXe, HangXe, MauXe, GiaXe, SoLuong, CreatedAt
                    FROM XeMay
                    WHERE MaXe LIKE %s OR TenXe LIKE %s OR HangXe LIKE %s OR MauXe
LIKE %s
                       OR CAST(GiaXe AS CHAR) LIKE %s OR CAST(SoLuong AS CHAR) LIKE %s
                """, (like, like, like, like, like, like))
            else:
                cursor.execute("""
                    SELECT MaXe, TenXe, HangXe, MauXe, GiaXe, SoLuong, CreatedAt
                    FROM XeMay ORDER BY MaXe ASC
                """)
```

```
            for row in cursor.fetchall():
                tree.insert("", tk.END, values=row)
            clear_form()
        except Exception as e:
            messagebox.showerror("Lỗi", f"Lỗi tải dữ liệu: {e}")
    # ================== THÊM TẠM ==================
    def add_temp():
        vals = {
            "MaXe": "",
            "TenXe": entries["TenXe"].get().strip(),
            "HangXe": entries["HangXe"].get().strip(),
            "MauXe": entries["MauXe"].get().strip(),
            "GiaXe": entries["GiaXe"].get().strip(),
            "SoLuong": entries["SoLuong"].get().strip()
        }
        if not vals["TenXe"]:
            messagebox.showwarning("Thiếu", "Tên xe không được để trống!")
            return
        temp_data.append(vals)
        tree.insert("", tk.END,
                    values=["", vals["TenXe"], vals["HangXe"], vals["MauXe"], vals["GiaXe"],
vals["SoLuong"], ""])
        clear_form()
    # ================== CẬP NHẬT XE ==================
    def update_xe():
        sel = tree.selection()
        if not sel:
            messagebox.showwarning("Chọn dòng", "Chọn xe cần cập nhật!")
            return
        maxe = entries["MaXe"].get()
        if maxe == "Auto (DB)":
            messagebox.showwarning("Lỗi", "Không thể cập nhật bản ghi chưa có trong
DB!")
            return
        try:
            cursor.execute("""
                UPDATE XeMay
                SET TenXe=%s, HangXe=%s, MauXe=%s, GiaXe=%s, SoLuong=%s
                WHERE MaXe=%s
            """, (
                entries["TenXe"].get(),
                entries["HangXe"].get(),
                entries["MauXe"].get(),
                entries["GiaXe"].get(),
                entries["SoLuong"].get(),
```

```
            maxe
        ))
        conn.commit()
        messagebox.showinfo("Thành công", f"Xe {maxe} đã được cập nhật!")
        load_data()
    except Exception as e:
        messagebox.showerror("Lỗi", f"Lỗi cập nhật: {e}")
# ================== LƯU DB ==================
def save_all():
    if not temp_data:
        messagebox.showinfo("Thông báo", "Không có dữ liệu để lưu.")
        return
    try:
        for xe in temp_data:
            cursor.execute("""
                INSERT INTO XeMay (TenXe, HangXe, MauXe, GiaXe, SoLuong)
                VALUES (%s,%s,%s,%s,%s)
            """, (xe["TenXe"], xe["HangXe"], xe["MauXe"], xe["GiaXe"], xe["SoLuong"]))

        conn.commit()
        temp_data.clear()
        load_data()
        messagebox.showinfo("OK", "Đã lưu xe mới!")

    except Exception as e:
        messagebox.showerror("Lỗi", f"Lỗi lưu: {e}")
# ================== XÓA XE ==================
def delete_xe():
    sel = tree.selection()
    if not sel:
        messagebox.showwarning("Chọn dòng", "Chọn xe cần xóa!")
        return
    maxe = tree.item(sel[0])["values"][0]
    if not messagebox.askyesno("Xác nhận", f"Xóa xe mã {maxe}?"):
        return
    try:
        cursor.execute("DELETE FROM XeMay WHERE MaXe=%s", (maxe,))
        conn.commit()
    except:
        pass
    tree.delete(sel[0])
    clear_form()
# ================== CHỌN DÒNG ==================
def on_select(event):
    sel = tree.selection()
```

```
        if not sel:
            return
        vals = tree.item(sel[0])["values"]
        for i, c in enumerate(columns[:-1]):
            entries[c].config(state="normal")
            entries[c].delete(0, tk.END)
            entries[c].insert(0, vals[i])
            if c == "MaXe":
                entries[c].config(state="readonly")
    tree.bind("<<TreeviewSelect>>", on_select)
    # ================== NÚT CHỨC NĂNG ==================
    frame_btn = tk.Frame(win, bg="#AEEBFF")
    frame_btn.pack(pady=10)

    tk.Button(frame_btn, text="Thêm tạm", width=12,
command=add_temp,bg="#2196F3", fg="black").grid(row=0, column=0, padx=6)
    tk.Button(frame_btn, text="Lưu", width=12, command=save_all,bg="#4CAF50",
fg="black").grid(row=0, column=1, padx=6)
    tk.Button(frame_btn, text="Cập nhật", width=12,
command=update_xe,bg="#FFC107",fg="black").grid(row=0, column=2, padx=6)
    tk.Button(frame_btn, text="Xóa", width=12, command=delete_xe, bg="#f44336",
fg="black").grid(row=0, column=3, padx=6)
    tk.Button(frame_btn, text="Hủy", width=12, command=clear_form,bg="#9E9E9E",
fg="black").grid(row=0, column=4, padx=6)
    def on_back():
        win.destroy()
        if root:
            root.deiconify()
    tk.Button(frame_btn, text="Quay lại", width=12, command=on_back,bg="#2196F3",
fg="black").grid(row=0, column=5, padx=6)
    win.protocol("WM_DELETE_WINDOW", on_back)
    load_data()
# === Chạy riêng ===
if __name__ == "__main__":
    from database import get_connection
    conn = get_connection()
    open_xe_form(None, conn)
    tk.mainloop()
```

3. CRUD Khách hàng – form_KhachHang.py

```
import tkinter as tk
from tkinter import ttk, messagebox
```

```
from database import get_connection
def open_kh_form(root=None, conn=None):
    cursor = conn.cursor() if conn else None
    if root is not None:
        root.withdraw()
    win = tk.Toplevel(root) if root is not None else tk.Tk()
    win.title("Quản lý Khách hàng")
    win.geometry("1100x650")
    win.configure(bg="#AEEBFF")   # ✨ Nền xanh giống form Xe/NV
    tk.Label(win, text="QUẢN LÝ KHÁCH HÀNG",font=("Arial", 20, "bold"),fg="#003399",
bg="#AEEBFF").pack(pady=10)
    # ================ TÌM KIẾM ================
    frame_search = tk.Frame(win, bg="#AEEBFF")
    frame_search.pack(pady=5)
    tk.Label(frame_search, text="🔍 Tìm kiếm:", font=("Arial", 10,
"bold"),bg="#AEEBFF").grid(row=0, column=0, padx=5)
    search_var = tk.StringVar()
    tk.Entry(frame_search, textvariable=search_var, width=60).grid(row=0, column=1,
padx=5)
    tk.Button(frame_search, text="Tìm", bg="#2196F3", fg="black",
width=10,command=lambda: load_data(search_var.get())).grid(row=0, column=2,
padx=5)
    tk.Button(frame_search, text="Tải lại", bg="#9E9E9E", fg="black",
width=10,command=lambda: (search_var.set(""), load_data())).grid(row=0, column=3,
padx=5)
    # ================ BẢNG KHÁCH HÀNG ================
    frame_ds = tk.LabelFrame(win, text="Danh sách khách hàng",padx=10,
pady=10,font=("Arial", 11, "bold"),fg="#003366", bg="#AEEBFF")
    frame_ds.pack(padx=10, pady=10, fill="both", expand=True)
    scroll = tk.Scrollbar(frame_ds)
    scroll.pack(side=tk.RIGHT, fill=tk.Y)
    columns = ["MaKH", "TenKH", "DiaChi", "SDT", "Email"]
    tree = ttk.Treeview(frame_ds, columns=columns, show="headings",
yscrollcommand=scroll.set, height=12)
    for c in columns:
        tree.heading(c, text=c)
        tree.column(c, width=180)
    tree.pack(fill="both", expand=True)
    scroll.config(command=tree.yview)
    # ================ FORM NHẬP ================
    frame_form = tk.LabelFrame(
        win, text="Thông tin khách hàng",padx=10, pady=10,font=("Arial", 10, "bold"),
bg="#AEEBFF")
    frame_form.pack(pady=10, padx=10, fill="x")
    entries = {}
```

```
    fields = ["MaKH", "TenKH", "DiaChi", "SDT", "Email"]
    for i, f in enumerate(fields):
        tk.Label(frame_form, text=f + ":", font=("Arial", 10),bg="#AEEBFF").grid(row=i // 3,
column=(i % 3) * 2,sticky="w", padx=6, pady=4)
        e = tk.Entry(frame_form, width=25)
        e.grid(row=i // 3, column=(i % 3) * 2 + 1, padx=6, pady=4)
        entries[f] = e
    # ====== MaKH readonly ======
    def set_auto_makh():
        entries["MaKH"].config(state="normal")
        entries["MaKH"].delete(0, tk.END)
        entries["MaKH"].insert(0, "Auto (DB)")
        entries["MaKH"].config(state="readonly")
    set_auto_makh()
    temp_data = []
    # ================ CLEAR FORM ==================
    def clear_form():
        for c in entries:
            entries[c].config(state="normal")
            entries[c].delete(0, tk.END)
        set_auto_makh()
    # ================ LOAD DATA ==================
    def load_data(search=None):
        tree.delete(*tree.get_children())
        temp_data.clear()
        try:
            if search:
                like = f"%{search}%"
                cursor.execute("""
                    SELECT MaKH, TenKH, DiaChi, SDT, Email
                    FROM KhachHang
                    WHERE MaKH LIKE %s OR TenKH LIKE %s
                      OR DiaChi LIKE %s OR SDT LIKE %s OR Email LIKE %s
                """, (like, like, like, like, like))
            else:
                cursor.execute("SELECT MaKH, TenKH, DiaChi, SDT, Email FROM KhachHang
ORDER BY MaKH ASC")

            for row in cursor.fetchall():
                tree.insert("", tk.END, values=row)
            clear_form()
        except Exception as e:
            messagebox.showerror("Lỗi", f"Lỗi tải dữ liệu: {e}")
    # ================ THÊM TẠM ==================
    def add_temp():
```

```
        vals = {
            "MaKH": "",
            "TenKH": entries["TenKH"].get().strip(),
            "DiaChi": entries["DiaChi"].get().strip(),
            "SDT": entries["SDT"].get().strip(),
            "Email": entries["Email"].get().strip()
        }
        if not vals["TenKH"]:
            messagebox.showwarning("Thiếu", "Tên khách hàng không được để trống!")
            return
        temp_data.append(vals)
        tree.insert("", tk.END, values=["", vals["TenKH"], vals["DiaChi"], vals["SDT"],
vals["Email"]])
        clear_form()
    # ============== CẬP NHẬT KH ==================
    def update_kh():
        sel = tree.selection()
        if not sel:
            messagebox.showwarning("Chọn dòng", "Chọn khách hàng cần cập nhật!")
            return
        makh = entries["MaKH"].get()
        if makh == "Auto (DB)":
            messagebox.showwarning("Lỗi", "Không thể cập nhật bản ghi chưa có trong
database!")
            return
        try:
            cursor.execute("""
                UPDATE KhachHang
                SET TenKH=%s, DiaChi=%s, SDT=%s, Email=%s
                WHERE MaKH=%s
            """, (
                entries["TenKH"].get(),
                entries["DiaChi"].get(),
                entries["SDT"].get(),
                entries["Email"].get(),
                makh
            ))
            conn.commit()
            messagebox.showinfo("Thành công", f"Đã cập nhật khách hàng {makh}!")
            load_data()
        except Exception as e:
            messagebox.showerror("Lỗi", f"Lỗi cập nhật: {e}")
    # ============== LƯU VÀO DB ================
    def save_all():
        if not temp_data:
```

```
        messagebox.showinfo("Thông báo", "Không có dữ liệu để lưu.")
        return
    try:
        for kh in temp_data:
            cursor.execute("""
                INSERT INTO KhachHang (TenKH, DiaChi, SDT, Email)
                VALUES (%s,%s,%s,%s)
            """, (kh["TenKH"], kh["DiaChi"], kh["SDT"], kh["Email"]))
        conn.commit()
        temp_data.clear()
        load_data()
        messagebox.showinfo("OK", "Đã lưu khách hàng mới!")
    except Exception as e:
        messagebox.showerror("Lỗi", f"Lỗi lưu dữ liệu: {e}")
# =============== XÓA KH ====================
def delete_kh():
    sel = tree.selection()
    if not sel:
        messagebox.showwarning("Chọn dòng", "Chọn khách hàng để xóa!")
        return
    makh = tree.item(sel[0])["values"][0]
    if not messagebox.askyesno("Xác nhận", f"Xóa khách hàng {makh}?"):
        return
    try:
        cursor.execute("DELETE FROM KhachHang WHERE MaKH=%s", (makh,))
        conn.commit()
    except:
        pass
    tree.delete(sel[0])
    clear_form()
# =============== CHỌN DÒNG ==================
def on_select(event):
    sel = tree.selection()
    if not sel:
        return
    vals = tree.item(sel[0])["values"]
    for i, c in enumerate(columns):
        entries[c].config(state="normal")
        entries[c].delete(0, tk.END)
        entries[c].insert(0, vals[i])
        if c == "MaKH":
            entries[c].config(state="readonly")
tree.bind("<<TreeviewSelect>>", on_select)
# =============== NÚT CHỨC NĂNG ==============
frame_btn = tk.Frame(win, bg="#AEEBFF")
```

```
    frame_btn.pack(pady=10)

    tk.Button(frame_btn, text="Thêm tạm", width=12, command=add_temp,
bg="#2196F3", fg="black").grid(row=0, column=0, padx=6)
    tk.Button(frame_btn, text="Lưu", width=12, command=save_all,bg="#4CAF50",
fg="black").grid(row=0, column=1, padx=6)
    tk.Button(frame_btn, text="Cập nhật", width=12,
command=update_kh,bg="#FFC107", fg="black").grid(row=0, column=2, padx=6)
    tk.Button(frame_btn, text="Xóa", width=12, command=delete_kh,bg="#f44336",
fg="black").grid(row=0, column=3, padx=6)
    tk.Button(frame_btn, text="Hủy", width=12, command=clear_form, bg="#9E9E9E",
fg="black").grid(row=0, column=4, padx=6)
    def on_back():
        win.destroy()
        if root is not None:
            root.deiconify()
    tk.Button(frame_btn, text="Quay lại", width=12, command=on_back,bg="#2196F3",
fg="black").grid(row=0, column=5, padx=6)
    win.protocol("WM_DELETE_WINDOW", on_back)
    load_data()
# === Chạy riêng ==
if __name__ == "__main__":
    conn = get_connection()
    open_kh_form(None, conn)
    tk.mainloop()
```

4. CRUD Nhân viên – form_NhanVien.py

```
import tkinter as tk
from tkinter import ttk, messagebox
def open_nv_form(root=None, conn=None):
    cursor = conn.cursor() if conn else None
    if root is not None:
        root.withdraw()
    win = tk.Toplevel(root) if root is not None else tk.Tk()
    win.title("Quản lý Nhân Viên")
    win.geometry("1100x650")
    win.configure(bg="#AEEBFF")
    # ===== TIÊU ĐỀ =====
    tk.Label( win,text="QUẢN LÝ NHÂN VIÊN",font=("Arial", 20,
"bold"),fg="#003399",bg="#AEEBFF").pack(pady=10)
    temp_data = []
    # ==================== TÌM KIẾM ====================
    search_frame = tk.Frame(win, bg="#AEEBFF")
    search_frame.pack(pady=5)
```

```
    tk.Label(search_frame, text="🔍 Tìm kiếm:", font=("Arial", 11, "bold"),
bg="#AEEBFF").grid(row=0, column=0, padx=5)
    search_var = tk.StringVar()
    tk.Entry(search_frame, textvariable=search_var, width=60).grid(row=0, column=1,
padx=5)
    # ---- Nút Tìm ----
    tk.Button(search_frame, text="Tìm", width=10,bg="#2196F3",
fg="black",command=lambda: load_data(search_var.get())).grid(row=0, column=2,
padx=5)
    # ---- Nút Tải lại ----
    tk.Button(search_frame, text="Tải lại", width=10,bg="#9E9E9E",
fg="black",command=lambda: (search_var.set(""), load_data())).grid(row=0, column=3,
padx=5)
    # ====================== DANH SÁCH NHÂN VIÊN ======================
    frame_ds = tk.LabelFrame(
        win,
        text="Danh sách nhân viên",
        padx=10, pady=10,
        font=("Arial", 11, "bold"),
        fg="#003366",
        bg="#AEEBFF"
    )
    frame_ds.pack(padx=10, pady=10, fill="both", expand=True)
    scroll = tk.Scrollbar(frame_ds)
    scroll.pack(side=tk.RIGHT, fill=tk.Y)
    columns = ["MaNV", "HoTen", "GioiTinh", "NgaySinh", "SDT", "DiaChi", "ChucVu",
"Luong", "CreatedAt"]
    tree = ttk.Treeview(frame_ds, columns=columns, show="headings",
yscrollcommand=scroll.set, height=12)
    for c in columns:
        tree.heading(c, text=c)
        tree.column(c, width=130)
    tree.pack(fill="both", expand=True)
    scroll.config(command=tree.yview)
    # ====================== FORM NHẬP ======================
    form = tk.LabelFrame(
        win,
        text="Thông tin nhân viên",
        padx=10, pady=10,
        font=("Arial", 10, "bold"),
        bg="#AEEBFF"
    )
    form.pack(pady=10, padx=10, fill="x")
    entries = {}
    fields = ["MaNV", "HoTen", "GioiTinh", "NgaySinh", "SDT", "DiaChi", "ChucVu", "Luong"]
```

```
    for i, c in enumerate(fields):
        tk.Label(form, text=c + ":", font=("Arial", 10), bg="#AEEBFF").grid(
            row=i // 3, column=(i % 3) * 2, sticky="w", padx=6, pady=4
        )
        e = tk.Entry(form, width=25)
        e.grid(row=i // 3, column=(i % 3) * 2 + 1, padx=6, pady=4)
        entries[c] = e
    # ===== AUTO MÃ NV =====
    def set_auto_manv():
        entries["MaNV"].config(state="normal")
        entries["MaNV"].delete(0, tk.END)
        entries["MaNV"].insert(0, "Auto (DB)")
        entries["MaNV"].config(state="readonly")
    set_auto_manv()
    # ===== CLEAR FORM =====
    def clear_form():
        for c in entries:
            entries[c].config(state="normal")
            entries[c].delete(0, tk.END)
        set_auto_manv()
    # ==================== LOAD DATA ====================
    def load_data(search=None):
        tree.delete(*tree.get_children())
        temp_data.clear()
        try:
            if search:
                like = f"%{search}%"
                cursor.execute("""
                    SELECT MaNV, HoTen, GioiTinh, NgaySinh, SDT, DiaChi, ChucVu, Luong,
CreatedAt
                    FROM NhanVien
                    WHERE MaNV LIKE %s
                       OR HoTen LIKE %s
                       OR GioiTinh LIKE %s
                       OR SDT LIKE %s
                       OR DiaChi LIKE %s
                       OR ChucVu LIKE %s
                """, (like, like, like, like, like, like))
            else:
                cursor.execute("""
                    SELECT MaNV, HoTen, GioiTinh, NgaySinh, SDT, DiaChi, ChucVu, Luong,
CreatedAt
                    FROM NhanVien ORDER BY MaNV ASC
                """)
            for row in cursor.fetchall():
```

```
            tree.insert("", tk.END, values=row)
        clear_form()
    except Exception as e:
        messagebox.showerror("Lỗi", f"Lỗi load dữ liệu: {e}")
# ==================== THÊM TẠM ====================
def add_temp():
    vals = {c: entries[c].get().strip() for c in entries}
    if not vals["HoTen"]:
        messagebox.showwarning("Thiếu", "Họ tên không được để trống!")
        return
    temp_data.append(vals)
    tree.insert("", tk.END, values=[
        "", vals["HoTen"], vals["GioiTinh"], vals["NgaySinh"],
        vals["SDT"], vals["DiaChi"], vals["ChucVu"], vals["Luong"], ""
    ])
    clear_form()
# ==================== CẬP NHẬT ====================
def update_nv():
    sel = tree.selection()
    if not sel:
        messagebox.showwarning("Chú ý", "Chọn nhân viên cần cập nhật!")
        return
    manv = entries["MaNV"].get()
    if manv == "Auto (DB)":
        messagebox.showwarning("Lỗi", "Không thể cập nhật bản ghi chưa có trong
DB!")
        return
    hoten = entries["HoTen"].get()
    gt = entries["GioiTinh"].get()
    ns = entries["NgaySinh"].get()
    sdt = entries["SDT"].get()
    dc = entries["DiaChi"].get()
    cv = entries["ChucVu"].get()
    luong = entries["Luong"].get()
    try:
        cursor.execute("""
            UPDATE NhanVien
            SET HoTen=%s, GioiTinh=%s, NgaySinh=%s, SDT=%s, DiaChi=%s, ChucVu=%s,
Luong=%s
            WHERE MaNV=%s
        """, (hoten, gt, ns, sdt, dc, cv, luong, manv))
        conn.commit()
        messagebox.showinfo("Thành công", f"Đã cập nhật nhân viên {manv}!")
        load_data()
    except Exception as e:
```

```
            messagebox.showerror("Lỗi", f"Lỗi cập nhật: {e}")
    # ===================== LƯU DB =====================
    def save_all():
        if not temp_data:
            messagebox.showinfo("Thông báo", "Không có dữ liệu để lưu.")
            return
        try:
            for nv in temp_data:
                cursor.execute("""
                    INSERT INTO NhanVien (HoTen, GioiTinh, NgaySinh, SDT, DiaChi, ChucVu,
Luong)
                    VALUES (%s,%s,%s,%s,%s,%s,%s)
                """, (nv["HoTen"], nv["GioiTinh"], nv["NgaySinh"],
                      nv["SDT"], nv["DiaChi"], nv["ChucVu"], nv["Luong"]))
            conn.commit()
            temp_data.clear()
            load_data()
            messagebox.showinfo("OK", "Đã lưu dữ liệu mới!")
        except Exception as e:
            messagebox.showerror("Lỗi", f"Lỗi lưu dữ liệu: {e}")
    # ===================== XÓA =====================
    def delete_nv():
        sel = tree.selection()
        if not sel:
            messagebox.showwarning("Chọn dòng", "Phải chọn nhân viên để xóa!")
            return

        manv = tree.item(sel[0])["values"][0]

        if not messagebox.askyesno("Xác nhận", f"Xóa nhân viên {manv}?"):
            return

        try:
            cursor.execute("DELETE FROM NhanVien WHERE MaNV=%s", (manv,))
            conn.commit()
        except:
            pass

        tree.delete(sel[0])
        clear_form()
    # ===================== CLICK BẢNG =====================
    def on_select(event):
        sel = tree.selection()
        if not sel:
            return
```

```
        vals = tree.item(sel[0])["values"]
        for i, c in enumerate(columns[:-1]):
            entries[c].config(state="normal")
            entries[c].delete(0, tk.END)
            entries[c].insert(0, vals[i])
            if c == "MaNV":
                entries[c].config(state="readonly")
    tree.bind("<<TreeviewSelect>>", on_select)
    # ==================== BUTTON BAR ====================
    btn_frame = tk.Frame(win, bg="#AEEBFF")
    btn_frame.pack(pady=10)
    # Thêm tạm
    tk.Button(btn_frame, text="Thêm tạm", width=12, command=add_temp,
              bg="#2196F3", fg="black").grid(row=0, column=0, padx=6)
    # Lưu
    tk.Button(btn_frame, text="Lưu", width=12, command=save_all,
              bg="#4CAF50", fg="black").grid(row=0, column=1, padx=6)
    # Cập nhật
    tk.Button(btn_frame, text="Cập nhật", width=12, command=update_nv,
              bg="#FFC107", fg="black").grid(row=0, column=2, padx=6)
    # Xóa
    tk.Button(btn_frame, text="Xóa", width=12, command=delete_nv,
              bg="#f44336", fg="black").grid(row=0, column=3, padx=6)
    # Hủy
    tk.Button(btn_frame, text="Hủy", width=12, command=clear_form,
              bg="#9E9E9E", fg="black").grid(row=0, column=4, padx=6)
    # Quay lại
    def on_back():
        win.destroy()
        if root is not None:
            root.deiconify()
    tk.Button(btn_frame, text="Quay lại", width=12, command=on_back,
              bg="#2196F3", fg="black").grid(row=0, column=5, padx=6)
    win.protocol("WM_DELETE_WINDOW", on_back)
    load_data()
# === Chạy riêng ===
if __name__ == "__main__":
    from database import get_connection
    conn = get_connection()
    open_nv_form(None, conn)
    tk.mainloop()
```

5. CRUD Hóa đơn – form_HoaDon.py

```
import tkinter as tk
```

```
from tkinter import ttk, messagebox
from database import get_connection
from form_KhachHang import open_kh_form
from form_CTHoaDon import open_cthd_form

def open_hd_form(root=None, conn=None):
    is_standalone = False
    if root is None:
        is_standalone = True
        root = tk.Tk()
        root.withdraw()
    if conn is None:
        conn = get_connection()
    cursor = conn.cursor()
    win = tk.Toplevel(root)
    win.title("Quản lý Hóa đơn")
    win.geometry("1000x600")
    win.configure(bg="#AEEBFF")   # ⭐ NỀN XANH
    if root is not None:
        root.withdraw()
    # ===== TIÊU ĐỀ =====
    tk.Label(
        win,
        text="QUẢN LÝ HÓA ĐƠN",
        font=("Arial", 19, "bold"),
        fg="#003399",
        bg="#AEEBFF"
    ).pack(pady=10)
    # ===== THANH TÌM KIẾM =====
    search_frame = tk.Frame(win, bg="#AEEBFF")
    search_frame.pack(pady=5)
    tk.Label(
        search_frame,
        text="🔍 Tìm kiếm:",
        font=("Arial", 11, "bold"),
        bg="#AEEBFF"
    ).grid(row=0, column=0, padx=5)
    search_var = tk.StringVar()
    tk.Entry(
        search_frame, textvariable=search_var,
        width=60, font=("Arial", 10)
    ).grid(row=0, column=1, padx=5)
    tk.Button(search_frame, text="Tìm",bg="#2196F3", fg="black",
width=10,command=lambda: load_data(search_var.get())).grid(row=0, column=2,
padx=5)
```

```
    tk.Button(search_frame, text="Tải lại",bg="#9E9E9E", fg="black",
width=10,command=lambda: (search_var.set(""), load_data())).grid(row=0, column=3,
padx=5)
    # ===== DANH SÁCH HÓA ĐƠN =====
    frame_dshd = tk.LabelFrame(
        win,
        text="Danh sách hóa đơn",
        padx=5, pady=1,
        font=("Arial", 11, "bold"),
        fg="#003366",
        bg="#AEEBFF"
    )
    frame_dshd.pack(padx=5, pady=1, fill="both", expand=True)
    scroll = tk.Scrollbar(frame_dshd)
    scroll.pack(side=tk.RIGHT, fill=tk.Y)
    columns = ["MaHD", "TenKH", "TenNV", "TongTien", "NgayBan"]
    tree = ttk.Treeview(
        frame_dshd,
        columns=columns,
        show="headings",
        yscrollcommand=scroll.set,
        height=10
    )
    for c in columns:
        tree.heading(c, text=c)
        tree.column(c, width=150)

    tree.pack(fill="both", expand=True)
    scroll.config(command=tree.yview)
    # ===== FORM =====
    form = tk.LabelFrame(
        win,
        text="Thông tin hóa đơn",
        padx=10, pady=10,
        font=("Arial", 10, "bold"),
        bg="#AEEBFF",
        fg="#003366"
    )
    form.pack(pady=10, padx=10, fill="x")
    entries = {}
    left_fields = ["MaHD", "NgayBan", "MaNV"]
    for i, c in enumerate(left_fields):
        tk.Label(
            form,
            text=c + ":",
```

```
        font=("Arial", 10),
        bg="#AEEBFF"
    ).grid(row=i, column=0, sticky="w", padx=6, pady=4)

    e = tk.Entry(form, width=30)
    e.grid(row=i, column=1, padx=6, pady=4)
    entries[c] = e

entries["MaHD"].config(state="readonly")
# --- Khách hàng ---
tk.Label(
    form,
    text="Khách hàng:",
    font=("Arial", 10),
    bg="#AEEBFF"
).grid(row=3, column=0, sticky="w", padx=6, pady=4)
kh_var = tk.StringVar()
cb_kh = ttk.Combobox(form, textvariable=kh_var, width=33)
cb_kh.grid(row=3, column=1, padx=6, pady=4)
entries["MaKH"] = cb_kh
tk.Button(
    form, text="🔍 Tìm khách", width=12,
    bg="#2196F3", fg="black",
    command=lambda: search_customer()
).grid(row=3, column=3, padx=5)
tk.Button(
    form, text="+ Thêm khách", width=12,
    bg="#4CAF50", fg="black",
    command=lambda: add_customer_and_reload()
).grid(row=3, column=4, padx=5)
# ===== TỔNG TIỀN (NHƯ CODE CŨ: CHỈ TẠO LABEL NHƯNG KHÔNG GRID) =====
tongtien_var = tk.StringVar()
lbl_tongtien = tk.Label(
    form,
    textvariable=tongtien_var,
    font=("Arial", 10),
    bg="#f0f0f0",
    width=28,
    anchor="w"
)
tk.Label(
    form, text="GhiChu:", font=("Arial", 10),
    bg="#AEEBFF"
).grid(row=5, column=0, sticky="w", padx=6, pady=4)
ghi = tk.Entry(form, width=30)
```

```
    ghi.grid(row=5, column=1, padx=6, pady=4)
    entries["GhiChu"] = ghi

    temp_data = []
    def next_mahd():
        entries["MaHD"].config(state="normal")
        entries["MaHD"].delete(0, tk.END)
        entries["MaHD"].insert(0, "Auto (DB)")
        entries["MaHD"].config(state="readonly")

    def load_customers():
        cursor.execute("SELECT MaKH, TenKH, SDT FROM KhachHang ORDER BY MaKH
ASC")
        rows = cursor.fetchall()
        cb_kh['values'] = [f"{r[0]} - {r[1]} ({r[2]})" for r in rows]

    def search_customer():
        kw = kh_var.get().strip()
        if not kw:
            return
        cursor.execute("""
            SELECT MaKH, TenKH, SDT FROM KhachHang
            WHERE TenKH LIKE %s OR SDT LIKE %s OR MaKH LIKE %s
        """, (f"%{kw}%", f"%{kw}%", f"%{kw}%"))
        rows = cursor.fetchall()
        if rows:
            cb_kh['values'] = [f"{r[0]} - {r[1]} ({r[2]})" for r in rows]
            cb_kh.current(0)

    def add_customer_and_reload():
        open_kh_form(win, conn)
        load_customers()

    def clear_form():
        for k, v in entries.items():
            if k == "MaKH":
                cb_kh.set("")
                continue
            if k == "TongTien":
                tongtien_var.set("")
                continue
            try:
                v.config(state="normal")
            except:
                pass
```

```
            if hasattr(v, "delete"):
                v.delete(0, tk.END)
            if k == "MaHD":
                v.config(state="readonly")
        next_mahd()

    def load_data(search=None):
        tree.delete(*tree.get_children())
        temp_data.clear()

        query = """
            SELECT hd.MaHD, kh.TenKH, nv.HoTen, hd.TongTien, hd.NgayBan
            FROM HoaDon hd
            JOIN KhachHang kh ON hd.MaKH = kh.MaKH
            JOIN NhanVien nv ON hd.MaNV = nv.MaNV
        """

        if search:
            like = f"%{search}%"
            query += " WHERE kh.TenKH LIKE %s OR nv.HoTen LIKE %s"
            cursor.execute(query + " ORDER BY hd.MaHD ASC", (like, like))
        else:
            cursor.execute(query + " ORDER BY hd.MaHD ASC")

        for row in cursor.fetchall():
            tree.insert("", tk.END, values=row)

        next_mahd()
        load_customers()

    def add_temp():
        vals = {}
        vals["NgayBan"] = entries["NgayBan"].get().strip()

        mk = entries["MaKH"].get()
        vals["MaKH"] = mk.split(" - ")[0] if " - " in mk else mk.strip()

        vals["MaNV"] = getattr(root, "current_user_id", None) or
entries["MaNV"].get().strip()
        vals["TongTien"] = "0"
        vals["GhiChu"] = entries["GhiChu"].get().strip()

        if not vals["NgayBan"] or not vals["MaKH"] or not vals["MaNV"]:
            messagebox.showwarning("Chú ý", "Vui lòng nhập đủ thông tin!")
            return
```

```
    temp_data.append(vals)
    messagebox.showinfo("Tạm", "Đã thêm hóa đơn.")
    clear_form()

def save_all():
    if not temp_data:
        return messagebox.showinfo("Thông báo", "Không có dữ liệu để lưu.")

    try:
        for hd in temp_data:
            cursor.execute("""
                INSERT INTO HoaDon (NgayBan, MaKH, MaNV, TongTien, GhiChu)
                VALUES (%s,%s,%s,%s,%s)
            """, (hd["NgayBan"], hd["MaKH"], hd["MaNV"], hd["TongTien"], hd["GhiChu"]))
            conn.commit()

            cursor.execute("""
                SELECT MaHD FROM HoaDon
                WHERE NgayBan=%s AND MaKH=%s AND MaNV=%s
                ORDER BY CreatedAt DESC LIMIT 1
            """, (hd["NgayBan"], hd["MaKH"], hd["MaNV"]))
            new_mahd = cursor.fetchone()[0]

            open_cthd_form(win, conn, new_mahd)

        temp_data.clear()
        load_data()
        messagebox.showinfo("OK", "Đã lưu!")

    except Exception as e:
        conn.rollback()
        messagebox.showerror("Lỗi", str(e))

def delete_all():
    sel = tree.selection()
    if not sel:
        return
    mahd = tree.item(sel[0])['values'][0]

    if not messagebox.askyesno("Xác nhận", f"Xóa HĐ {mahd}?"):
        return

    cursor.execute("DELETE FROM HoaDon WHERE MaHD=%s", (mahd,))
    conn.commit()
```

```
    tree.delete(sel[0])
    clear_form()

def open_selected_cthd():
    sel = tree.selection()
    if not sel:
        return
    mahd = tree.item(sel[0])['values'][0]
    open_cthd_form(win, conn, mahd)

def on_select(event):
    sel = tree.selection()
    if not sel:
        return

    mahd, tenkh, tennv, tong, ngayban = tree.item(sel[0])['values']

    entries["MaHD"].config(state="normal")
    entries["MaHD"].delete(0, tk.END)
    entries["MaHD"].insert(0, mahd)
    entries["MaHD"].config(state="readonly")

    entries["NgayBan"].delete(0, tk.END)
    entries["NgayBan"].insert(0, ngayban)

    tongtien_var.set(tong)

tree.bind("<<TreeviewSelect>>", on_select)

# ===== NÚT CHỨC NĂNG =====
btn_frame = tk.Frame(win, bg="#AEEBFF")
btn_frame.pack(pady=10)

tk.Button(btn_frame, text="Thêm tạm", width=12,
          bg="#2196F3", fg="black",
          command=add_temp).grid(row=0, column=0, padx=6)

tk.Button(btn_frame, text="Lưu", width=12,
          bg="#4CAF50", fg="black",
          command=save_all).grid(row=0, column=1, padx=6)

tk.Button(btn_frame, text="Xóa", width=12,
          bg="#f44336", fg="black",
          command=delete_all).grid(row=0, column=2, padx=6)
```

```
    tk.Button(btn_frame, text="Mở chi tiết HĐ", width=14,
              bg="#2196F3", fg="black",
              command=open_selected_cthd).grid(row=0, column=3, padx=6)

    tk.Button(btn_frame, text="Hủy", width=12,
              bg="#9E9E9E", fg="black",
              command=clear_form).grid(row=0, column=4, padx=6)

    tk.Button(btn_frame, text="Quay lại", width=12,
              bg="#2196F3", fg="black",
              command=lambda: on_back()).grid(row=0, column=5, padx=6)
    def on_back():
        win.destroy()
        if is_standalone:
            root.destroy()
        else:
            root.deiconify()
    load_data()
    if is_standalone:
        win.mainloop()
if __name__ == "__main__":
    open_hd_form()
```

6. CRUD Chi tiết hóa đơn – form_CTHoaDon.py

```
import tkinter as tk
from tkinter import ttk, messagebox
from datetime import datetime
def number_to_vietnamese(n):
    units = ["", "một", "hai", "ba", "bốn", "năm", "sáu", "bảy", "tám", "chín"]
    tens = ["", "mười", "hai mươi", "ba mươi", "bốn mươi",
            "năm mươi", "sáu mươi", "bảy mươi", "tám mươi", "chín mươi"]
    thousands = ["", "nghìn", "triệu", "tỷ"]
    if n == 0:
        return "không"
    words = []
    i = 0
    while n > 0:
        part = n % 1000
        n //= 1000
        if part > 0:
            part_words = []
            hundred = part // 100
            ten = (part % 100) // 10
            unit = part % 10
```

```
            if hundred > 0:
                part_words.append(units[hundred] + " trăm")
            else:
                if (ten > 0 or unit > 0):
                    part_words.append("không trăm")

            if ten > 1:
                part_words.append(tens[ten])
            elif ten == 1:
                part_words.append("mười")
            else:
                if unit > 0 and hundred > 0:
                    part_words.append("lẻ")

            if unit > 0:
                if ten == 1 and unit == 5:
                    part_words.append("lăm")
                elif ten > 1 and unit == 1:
                    part_words.append("mốt")
                else:
                    part_words.append(units[unit])

            part_words.append(thousands[i])
            words.insert(0, " ".join(part_words))
        i += 1
    return " ".join(words).strip()
def open_cthd_form(root=None, conn=None, mahd=None):
    cursor = conn.cursor() if conn else None
    if root:
        root.withdraw()
    win = tk.Toplevel(root) if root else tk.Tk()
    win.title("Quản lý Chi Tiết Hóa Đơn")
    win.geometry("1250x700")
    win.configure(bg="#AEEBFF")  # ⭐ màu nền đồng bộ
    # ===== TIÊU ĐỀ =====
    tk.Label(win, text="QUẢN LÝ CHI TIẾT HÓA ĐƠN",
             font=("Arial", 19, "bold"),
             fg="#003399", bg="#AEEBFF").pack(pady=10)
    temp_data = []
    # ==================== TÌM KIẾM ====================
    search_frame = tk.Frame(win, bg="#AEEBFF")
    search_frame.pack(pady=5)
    tk.Label(search_frame, text="🔍 Tìm kiếm:", font=("Arial", 11,
"bold"),bg="#AEEBFF").grid(row=0, column=0, padx=5)
```

```
    search_var = tk.StringVar()
    tk.Entry(search_frame, textvariable=search_var, width=60).grid(row=0, column=1,
padx=5)
    tk.Button(search_frame, text="Tìm", bg="#2196F3",
fg="black",width=10,command=lambda: load_data(search_var.get())).grid(row=0,
column=2, padx=5)
    tk.Button(search_frame, text="Tải lại", bg="#9E9E9E",
fg="black",width=10,command=lambda: (search_var.set(""), load_data())).grid(row=0,
column=3, padx=5)
    # ==================== DANH SÁCH ====================
    frame_list = tk.LabelFrame(win, text="Danh sách chi tiết hóa đơn",
                               padx=10, pady=10,
                               font=("Arial", 11, "bold"),
                               fg="#003366", bg="#AEEBFF")
    frame_list.pack(fill="both", expand=True, padx=10, pady=10)
    scroll = tk.Scrollbar(frame_list)
    scroll.pack(side=tk.RIGHT, fill=tk.Y)
    columns = [
        "MaHD", "MaXe", "SoLuong", "DonGia", "ThanhTien",
        "TenKH", "TenNV", "NgayBan",
        "TenXe", "HangXe", "MauXe", "GhiChu"
    ]

    tree = ttk.Treeview(frame_list, columns=columns, show="headings",
                        yscrollcommand=scroll.set, height=12)
    for c in columns:
        tree.heading(c, text=c)
        tree.column(c, width=150)
    tree.pack(fill="both", expand=True)
    scroll.config(command=tree.yview)
    # ==================== FORM ====================
    form = tk.LabelFrame(win, text="Thông tin chi tiết hóa đơn",
                         padx=10, pady=10,
                         font=("Arial", 10, "bold"),
                         bg="#AEEBFF", fg="#003366")
    form.pack(fill="x", padx=10, pady=10)
    entries = {}
    fields = ["MaHD", "MaXe", "SoLuong", "DonGia", "ThanhTien"]
    for i, field in enumerate(fields):
        tk.Label(form, text=f"{field}:", font=("Arial", 10),
                 bg="#AEEBFF").grid(row=i // 3, column=(i % 3) * 2,
                                    padx=6, pady=4, sticky="w")
        e = tk.Entry(form, width=25)
        e.grid(row=i // 3, column=(i % 3) * 2 + 1, padx=6, pady=4)
        entries[field] = e
```

```
    entries["MaHD"].config(state="readonly")
    if mahd:
        entries["MaHD"].config(state="normal")
        entries["MaHD"].insert(0, mahd)
        entries["MaHD"].config(state="readonly")
    # ===== CLEAR FORM =====
    def clear_form():
        for f in entries:
            entries[f].config(state="normal")
            entries[f].delete(0, tk.END)

        if mahd:
            entries["MaHD"].insert(0, mahd)
            entries["MaHD"].config(state="readonly")
    # ==================== LOAD DATA ====================
    def load_data(search=None):
        tree.delete(*tree.get_children())
        temp_data.clear()
        if not cursor:
            return
        try:
            sql = """
                SELECT hd.MaHD, c.MaXe, c.SoLuong, c.DonGia, c.ThanhTien,
                       kh.TenKH, nv.HoTen, hd.NgayBan,
                       xm.TenXe, xm.HangXe, xm.MauXe, hd.GhiChu
                FROM CTHoaDon c
                JOIN HoaDon hd ON c.MaHD = hd.MaHD
                LEFT JOIN KhachHang kh ON hd.MaKH = kh.MaKH
                LEFT JOIN NhanVien nv ON hd.MaNV = nv.MaNV
                LEFT JOIN XeMay xm ON c.MaXe = xm.MaXe
            """
            params = ()

            if mahd:
                sql += " WHERE hd.MaHD=%s"
                params = (mahd,)
            if search:
                search = f"%{search}%"
                sql += " WHERE kh.TenKH LIKE %s OR xm.TenXe LIKE %s OR hd.MaHD LIKE %s"
                params = (search, search, search)
            sql += " ORDER BY hd.MaHD"

            cursor.execute(sql, params)
            for r in cursor.fetchall():
                tree.insert("", tk.END, values=r)
```

```
    except Exception as e:
        messagebox.showerror("Lỗi", str(e))
load_data()
# ==================== LẤY GIÁ XE ====================
def on_maxe_focus_out(event=None):
    mx = entries["MaXe"].get().strip()
    if not mx:
        return
    cursor.execute("SELECT GiaXe FROM XeMay WHERE MaXe=%s", (mx,))
    r = cursor.fetchone()

    entries["DonGia"].delete(0, tk.END)
    if r:
        entries["DonGia"].insert(0, r[0])

entries["MaXe"].bind("<FocusOut>", on_maxe_focus_out)
# ==================== THÊM TẠM ====================
def add_temp():
    vals = {f: entries[f].get().strip() for f in entries}

    if not vals["MaHD"] or not vals["MaXe"]:
        messagebox.showwarning("Thiếu", "Nhập đầy đủ Mã HD và Mã Xe")
        return
    try:
        sl = int(vals["SoLuong"])
        dg = float(vals["DonGia"])
        tt = sl * dg
    except:
        messagebox.showerror("Lỗi", "Số lượng hoặc đơn giá không hợp lệ!")
        return

    vals["ThanhTien"] = tt
    temp_data.append(vals)

    tree.insert("", tk.END, values=[
        vals["MaHD"], vals["MaXe"], vals["SoLuong"],
        vals["DonGia"], vals["ThanhTien"],
        "", "", "", "", "", "", ""
    ])
    clear_form()
# ==================== SAVE ====================
def save_all():
    if not temp_data:
        messagebox.showinfo("Thông báo", "Không có dữ liệu để lưu.")
```

```
            return
        try:
            for ct in temp_data:
                cursor.execute("""
                    INSERT INTO CTHoaDon (MaHD, MaXe, SoLuong, DonGia)
                    VALUES (%s,%s,%s,%s)
                """, (ct["MaHD"], ct["MaXe"], ct["SoLuong"], ct["DonGia"]))
                cursor.execute("""
                    UPDATE XeMay SET SoLuong = SoLuong - %s WHERE MaXe=%s
                """, (ct["SoLuong"], ct["MaXe"]))
                cursor.execute("""
                    UPDATE HoaDon
                    SET TongTien = (SELECT SUM(ThanhTien) FROM CTHoaDon WHERE
MaHD=%s)
                    WHERE MaHD=%s
                """, (ct["MaHD"], ct["MaHD"]))
            conn.commit()
            temp_data.clear()
            load_data()
            messagebox.showinfo("Thành công", "Đã lưu chi tiết hóa đơn!")
        except Exception as e:
            conn.rollback()
            messagebox.showerror("Lỗi", str(e))
    # ==================== CLICK TREEVIEW ====================
    def on_select(event):
        sel = tree.selection()
        if not sel:
            return
        row = tree.item(sel[0])["values"]

        for i, f in enumerate(fields):
            entries[f].delete(0, tk.END)
            entries[f].insert(0, row[i])
    tree.bind("<<TreeviewSelect>>", on_select)
    # ==================== IN HÓA ĐƠN ====================
    def build_invoice_text(mahd):
        cursor.execute("""
            SELECT hd.MaHD, hd.NgayBan, hd.TongTien, hd.GhiChu,
                   kh.TenKH, kh.SDT, kh.DiaChi,
                   nv.HoTen, nv.ChucVu
            FROM HoaDon hd
            LEFT JOIN KhachHang kh ON hd.MaKH = kh.MaKH
            LEFT JOIN NhanVien nv ON hd.MaNV = nv.MaNV
            WHERE hd.MaHD=%s
        """, (mahd,))
```

```
    hd = cursor.fetchone()
    cursor.execute("""
        SELECT x.MaXe, x.TenXe, x.HangXe, x.MauXe,
               c.SoLuong, c.DonGia, c.ThanhTien
        FROM CTHoaDon c
        JOIN XeMay x ON c.MaXe = x.MaXe
        WHERE c.MaHD=%s
    """, (mahd,))
    details = cursor.fetchall()
    lines = []
    lines.append("                    CỬA HÀNG XE MÁY ")
    lines.append("              ĐC: Long Xuyên – An Giang")
    lines.append("             SĐT: 0354299556 – MST: 0123456")
    lines.append("-" * 66)

    lines.append(f"MÃ HÓA ĐƠN : {hd[0]}")
    lines.append(f"NGÀY BÁN  : {hd[1]}")
    lines.append("")
    lines.append(f"KHÁCH HÀNG: {hd[4]}")
    lines.append(f"SĐT       : {hd[5]}")
    lines.append(f"ĐỊA CHỈ   : {hd[6]}")
    lines.append("")
    lines.append(f"NHÂN VIÊN : {hd[7]}")
    lines.append("-" * 66)
    lines.append(">> THÔNG TIN SẢN PHẨM")
    lines.append("-" * 66)

    for d in details:
        lines.append(f"• Mã xe     : {d[0]}")
        lines.append(f"  Tên xe    : {d[1]}")
        lines.append(f"  Hãng      : {d[2]}")
        lines.append(f"  Màu       : {d[3]}")
        lines.append(f"  SL        : {d[4]}")
        lines.append(f"  Thành tiền: {d[6]:,} VND")
        lines.append("")

    lines.append("-" * 66)
    lines.append(f"TỔNG TIỀN : {hd[2]:,} VND")
    lines.append(f"BẰNG CHỮ : {number_to_vietnamese(int(hd[2]))} đồng")
    lines.append(f"GHI CHÚ : {hd[3] if hd[3] else 'Không có'}")
    lines.append("-" * 66)
    lines.append("          CẢM ƠN QUÝ KHÁCH – HẸN GẶP LẠI")
    lines.append("")

    return "\n".join(lines)
```

```
    def print_invoice():
        sel = tree.selection()
        if not sel:
            messagebox.showwarning("Chọn dòng", "Chọn 1 dòng để in hóa đơn!")
            return

        mahd_print = tree.item(sel[0])["values"][0]

        preview = tk.Toplevel(win)
        preview.title("Xem trước khi in")
        preview.geometry("700x600")

        text_box = tk.Text(preview, font=("Consolas", 11))
        text_box.pack(fill="both", expand=True)
        text_box.insert("1.0", build_invoice_text(mahd_print))
        text_box.config(state="disabled")
    # ==================== BUTTONS ====================
    btn_frame = tk.Frame(win, bg="#AEEBFF")
    btn_frame.pack()
    tk.Button(btn_frame, text="Thêm tạm", command=add_temp, width=12,
              bg="#2196F3", fg="black").grid(row=0, column=0, padx=5)
    tk.Button(btn_frame, text="Lưu", command=save_all, width=12,
              bg="#4CAF50", fg="black").grid(row=0, column=1, padx=5)
    tk.Button(btn_frame, text="Xóa", width=12,
              bg="#f44336", fg="black").grid(row=0, column=2, padx=5)
    tk.Button(btn_frame, text="Hủy", command=clear_form, width=12,
              bg="#9E9E9E", fg="black").grid(row=0, column=3, padx=5)
    tk.Button(btn_frame, text="In HĐ", command=print_invoice, width=12,
              bg="#2196F3", fg="black").grid(row=0, column=4, padx=5)
    tk.Button(btn_frame, text="Quay lại", width=12, bg="#2196F3", fg="black",
              command=lambda: (win.destroy(), root.deiconify() if root else None)
              ).grid(row=0, column=5, padx=5)
    win.protocol("WM_DELETE_WINDOW",
                 lambda: (win.destroy(), root.deiconify() if root else None))
# ============== CHẠY FILE TRỰC TIẾP ==============
if __name__ == "__main__":
    from database import get_connection
    conn = get_connection()
    open_cthd_form(None, conn)
    tk.mainloop()
```

7. CRUD Lịch sử giá – form_LichSuGia.py

```
import tkinter as tk
```

```
from tkinter import ttk, messagebox

def open_lichsu_form(root=None, conn=None):
    cursor = conn.cursor()

    win = tk.Toplevel(root) if root else tk.Tk()
    win.title("Lịch sử thay đổi giá xe")
    win.geometry("900x520")
    win.configure(bg="#AEEBFF")  # 🌟 Nền xanh pastel

    # ====== TIÊU ĐỀ ======
    tk.Label(
        win,
        text="LỊCH SỬ THAY ĐỔI GIÁ XE",
        font=("Arial", 18, "bold"),
        fg="#003399",
        bg="#AEEBFF"
    ).pack(pady=10)

    # ====== FRAME SEARCH ======
    frame_search = tk.Frame(win, bg="#AEEBFF")
    frame_search.pack(pady=5)

    tk.Label(
        frame_search,
        text="Tìm theo Mã Xe hoặc Tên Xe:",
        font=("Arial", 11, "bold"),
        bg="#AEEBFF"
    ).grid(row=0, column=0)

    search_var = tk.StringVar()
    tk.Entry(frame_search, textvariable=search_var, width=40).grid(row=0, column=1,
padx=5)
    # ====== LOAD DATA ======
    def load_data(search=None):
        tree.delete(*tree.get_children())
        if search:
            like = f"%{search}%"
            cursor.execute("""
                SELECT ls.ID, ls.MaXe, xm.TenXe,
                       ls.GiaCu, ls.GiaMoi, ls.NgayThayDoi
                FROM LichSuGiaXe ls
                JOIN XeMay xm ON ls.MaXe = xm.MaXe
                WHERE ls.MaXe LIKE %s OR xm.TenXe LIKE %s
                ORDER BY ls.ID DESC
```

```
            """, (like, like))
        else:
            cursor.execute("""
                SELECT ls.ID, ls.MaXe, xm.TenXe,
                       ls.GiaCu, ls.GiaMoi, ls.NgayThayDoi
                FROM LichSuGiaXe ls
                JOIN XeMay xm ON ls.MaXe = xm.MaXe
                ORDER BY ls.ID DESC
            """)
        for row in cursor.fetchall():
            tree.insert("", tk.END, values=row)
    # ====== BUTTON SEARCH ======
    tk.Button(
        frame_search,
        text="Tìm",
        bg="#2196F3",
        fg="white",
        width=10,
        command=lambda: load_data(search_var.get())
    ).grid(row=0, column=2, padx=5)
    tk.Button(
        frame_search,
        text="Tải lại",
        bg="#9E9E9E",
        fg="white",
        width=10,
        command=lambda: (search_var.set(""), load_data())
    ).grid(row=0, column=3, padx=5)
    # ====== TABLE ======
    columns = ["ID", "MaXe", "TenXe", "GiaCu", "GiaMoi", "NgayThayDoi"]
    tree = ttk.Treeview(win, columns=columns, show="headings", height=10)
    for col in columns:
        tree.heading(col, text=col)
        tree.column(col, width=140)

    tree.pack(padx=19, pady=10)
    # ====== LOAD FIRST TIME ======
    load_data()
    win.mainloop()
if __name__ == "__main__":
    from database import get_connection
    conn = get_connection()
    open_lichsu_form(None, conn)
```

8. Đăng nhập phân quyền – form_login.py

```
import tkinter as tk
from tkinter import ttk, messagebox
# Cần đảm bảo file database.py có hàm get_connection() và close_connection()
from database import get_connection, close_connection
# Hàm được main.py gọi để mở cửa sổ đăng nhập.
# Tham số root_app là cửa sổ chính của ứng dụng.
def open_login_form(root_app):
    # Sử dụng root_app để tạo cửa sổ con (Toplevel)
    login_win = tk.Toplevel(root_app)
    login_win.title("Đăng nhập hệ thống")
    login_win.geometry("400x350")
    login_win.resizable(False, False)
    login_win.configure(bg="#B3E5FC")
    # RẤT QUAN TRỌNG: Bắt sự kiện đóng cửa sổ. Nếu đóng login, phải đóng luôn app.
    login_win.protocol("WM_DELETE_WINDOW", root_app.destroy)
    login_win.grab_set() # Buộc người dùng tương tác với cửa sổ này
    # ==== TIÊU ĐỀ (Giữ nguyên giao diện ) ====
    lbl_title = tk.Label(
        login_win,
        text="ĐĂNG NHẬP",
        font=("Arial", 22, "bold"),
        bg="#B3E5FC",
        fg="#0D47A1"
    )
    lbl_title.pack(pady=20)
    # ==== FORM FRAME ====
    frm = tk.Frame(login_win, bg="#B3E5FC")
    frm.pack(pady=10)
    # --- Tài khoản ---
    tk.Label(frm, text="Tài khoản:", bg="#B3E5FC", font=("Arial", 12)).grid(row=0,
column=0, sticky="w")
    entry_user = tk.Entry(frm, width=30)
    entry_user.grid(row=1, column=0, pady=5)
    # entry_user.insert(0, "admin") #
    # --- Mật khẩu ---
    tk.Label(frm, text="Mật khẩu:", bg="#B3E5FC", font=("Arial", 12)).grid(row=2,
column=0, sticky="w")
    entry_pass = tk.Entry(frm, width=30, show="*")
    entry_pass.grid(row=3, column=0, pady=5)
    # entry_pass.insert(0, "0909")
    # ==== CHECK LOGIN  ====
    def check_login():
        user = entry_user.get().strip()
        password = entry_pass.get().strip()
```

```
        if user == "" or password == "":
            messagebox.showwarning("Thiếu thông tin", "Vui lòng nhập đầy đủ Tài khoản và
Mật khẩu.")
            return

        conn = get_connection()
        if not conn: return

        cursor = None
        try:
            # SỬA LỖI LẤY DỮ LIỆU: Phải dùng dictionary=True để dễ lấy cột 'role'
            cursor = conn.cursor(dictionary=True)
            cursor.execute("SELECT role FROM TaiKhoan WHERE username=%s AND
password=%s", (user, password))
            result = cursor.fetchone()

            if result:
                messagebox.showinfo("Thành công", f"Đăng nhập thành công! Vai trò:
{result['role']}")

                # 1. Lưu vai trò
                root_app.role = result['role']

                # 2. Đóng cửa sổ login
                login_win.destroy()

                ### PHẦN SỬA LỖI HIỂN THỊ ###
                root_app.deiconify() # HIỆN LẠI CỬA SỔ CHÍNH ĐÃ BỊ ẨN
                root_app.lift()      # Đưa cửa sổ lên trên cùng
                root_app.focus_force() # Buộc cửa sổ lấy focus

                # 3. Áp dụng quyền
                if hasattr(root_app, 'apply_role_permissions'):
                    root_app.apply_role_permissions()
                ### KẾT THÚC PHẦN SỬA LỖI ###

            else:
                messagebox.showerror("Sai thông tin", "Tài khoản hoặc mật khẩu không
đúng.")

        except Exception as e:
            messagebox.showerror("Lỗi CSDL", f"Lỗi khi kiểm tra đăng nhập: {e}")
        finally:
            if cursor:
                cursor.close()
```

```
        close_connection(conn) # Đảm bảo đóng kết nối


    # ==== NÚT ĐĂNG NHẬP (Giữ nguyên giao diện của bạn) ====
    btn_login = tk.Button(
        login_win,
        text="Đăng nhập",
        width=15,
        height=2,
        bg="#64B5F6",
        fg="black",
        command=check_login
    )
    btn_login.pack(pady=15)

    # Enter = Đăng nhập
    login_win.bind("<Return>", lambda e: check_login())

    # Chờ cửa sổ login đóng lại trước khi mainloop tiếp tục
    root_app.wait_window(login_win)
```

9. CRUD file chính– form_Main.py

```
import tkinter as tk
from tkinter import messagebox
from database import get_connection
# các form con
from form_XeMay import open_xe_form
from form_NhanVien import open_nv_form
from form_KhachHang import open_kh_form
from form_HoaDon import open_hd_form
from form_CTHoaDon import open_cthd_form
from form_login import open_login_form

# ==================== TẠO CANVAS BO GÓC ====================
def _create_round_rect(self, x1, y1, x2, y2, radius=15, **kwargs):
    points = [
        x1 + radius, y1,
        x2 - radius, y1,
        x2, y1,
        x2, y1 + radius,
        x2, y2 - radius,
        x2, y2,
        x2 - radius, y2,
        x1 + radius, y2,
```

```
        x1, y2,
        x1, y2 - radius,
        x1, y1 + radius,
        x1, y1
    ]
    return self.create_polygon(points, smooth=True, **kwargs)
tk.Canvas.create_round_rect = _create_round_rect
# ==================== TẠO NÚT ĐẸP + HOVER ====================
def create_button(master, text, bg_color="#64B5F6", command=None):
    frame = tk.Frame(master, bg=master["bg"])
    frame.pack(pady=10)
    canvas = tk.Canvas(frame, width=300, height=50,
                       bg=master["bg"], highlightthickness=0)
    canvas.pack()
    rect = canvas.create_round_rect(2, 2, 298, 48, radius=12,
                                    fill=bg_color, outline="#1E88E5", width=2)
    lbl = tk.Label(canvas, text=text, font=("Arial", 12, "bold"),
                   bg=bg_color, fg="black")
    canvas.create_window(150, 25, window=lbl)
    # Hover effect
    def on_enter(e):
        canvas.itemconfig(rect, fill="#42A5F5")
        lbl.config(bg="#42A5F5")
    def on_leave(e):
        canvas.itemconfig(rect, fill=bg_color)
        lbl.config(bg=bg_color)
    canvas.bind("<Enter>", on_enter)
    canvas.bind("<Leave>", on_leave)
    lbl.bind("<Enter>", on_enter)
    lbl.bind("<Leave>", on_leave)
    if command:
        # Bắt sự kiện click vào Label
        lbl.bind("<Button-1>", lambda e: command())
        # Bắt sự kiện click vào Canvas (phòng trường hợp click vào viền)
        canvas.bind("<Button-1>", lambda e: command())
    return frame
# ==================== MAIN ====================
conn = get_connection()
if not conn:
    tk.messagebox.showerror("Lỗi", "Không kết nối được MySQL. Kiểm tra database.py")
    # Nếu không kết nối được DB, ứng dụng phải thoát ngay
    raise SystemExit("Không kết nối DB")
root = tk.Tk()
root.title("Quản Lý Cửa Hàng Xe Máy")
root.geometry("600x620")
```

```
root.configure(bg="#D6F4FF")
root.role = None # Biến lưu vai trò người dùng (Admin/Nhan_Vien)
# Ẩn menu chính để login trước
root.withdraw()
open_login_form(root)
# ==================== XỬ LÝ THOÁT ỨNG DỤNG ====================
def confirm_exit():
    # Sử dụng root.withdraw() để ẩn cửa sổ chính trong khi hỏi xác nhận, tránh lỗi focus
    root.withdraw()
    if messagebox.askyesno("Xác nhận", "Bạn chắc chắn muốn thoát?"):
        try:
            # Đảm bảo kết nối database được đóng trước khi thoát
            if conn:
                conn.close()
        except Exception:
            # Bỏ qua lỗi nếu conn đã bị đóng rồi
            pass
        root.destroy()
    else:
        # Nếu người dùng chọn No, hiển thị lại cửa sổ chính
        root.deiconify()
        root.lift()
        root.focus_force()
# DÒNG CODE QUAN TRỌNG ĐÃ THÊM/SỬA LỖI: Gắn hàm confirm_exit vào nút 'X'
root.protocol("WM_DELETE_WINDOW", confirm_exit)
# ==================== TIÊU ĐỀ ====================
tk.Label(root,
         text="QUẢN LÝ CỬA HÀNG",
         font=("Arial", 28, "bold"),
         fg="#0D47A1",
         bg="#D6F4FF"
         ).pack(pady=30)
# ==================== MENU CÁC NÚT ====================
# Truyền root và conn vào các form con
btn_xe = create_button(root, "QUẢN LÝ XE MÁY",command=lambda:
open_xe_form(root, conn))
btn_nv = create_button(root, "QUẢN LÝ NHÂN VIÊN",command=lambda:
open_nv_form(root, conn))
btn_kh = create_button(root, "QUẢN LÝ KHÁCH HÀNG",command=lambda:
open_kh_form(root, conn))
btn_hd = create_button(root, "QUẢN LÝ HÓA ĐƠN",command=lambda:
open_hd_form(root, conn))
btn_cthd = create_button(root, "QUẢN LÝ CHI TIẾT HÓA ĐƠN",command=lambda:
open_cthd_form(root, conn))
# ==================== NÚT THOÁT ====================
```

```
create_button(root, "THOÁT", bg_color="#E57373",
          command=confirm_exit) # Đảm bảo nút này gọi hàm confirm_exit
# ==================== ẨN CHỨC NĂNG THEO QUYỀN ====================
def apply_role_permissions():
    # Nếu nhân viên → ẩn mục quản lý nhân viên
    if root.role == "Nhan_Vien":
        btn_nv.pack_forget()
    # Thêm các logic ẩn/hiện khác nếu cần
root.apply_role_permissions = apply_role_permissions
# ==================== RUN ====================
root.mainloop()
```

VII. Phần 7: Kết luận

Ứng dụng đã hoàn thành đầy đủ chức năng theo yêu cầu đồ án. Giao diện đơn giản, dễ sử dụng, phù hợp với cửa hàng vừa và nhỏ. Dữ liệu được lưu vào MySQL nên đảm bảo ổn định và dễ mở rộng.

VIII. Phần 8: Hướng phát triển

- Thêm chức năng xuất hóa đơn PDF
- Tích hợp gửi email cho khách hàng
- Thêm biểu đồ thống kê doanh thu
- Chuyển sang giao diện đẹp hơn (ttkbootstrap hoặc customtkinter)

IX. Phần 9: Tài liệu tham khảo

[1] Python Documentation

Python Software Foundation. *Python Language Reference, version 3.x*.
Truy cập tại: https://docs.python.org

[2] Tkinter - Python GUI Library

Tkinter Reference Documentation.
Truy cập tại: https://tkdocs.com

[3] MySQL Documentation

Oracle Corporation. *MySQL 8.0 Reference Manual.*
Truy cập tại: https://dev.mysql.com/doc/

[4] mysql-connector-python

Document: *MySQL Connector/Python Developer Guide*.
Truy cập tại: https://dev.mysql.com/doc/connector-python/en/