**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 TIVI  
VỚI PYTHON, TKINTER VÀ MYSQL**

**Giảng viên hướng dẫn: Ths. Nguyễn Ngọc Minh**

**Sinh viên thực hiện:**

**DTH235704 - Thái Vĩnh Nghi - Lớp DH24TH2\_NhómTH2\_TổTH2**

**DTH235680 - Trần Đăng Khoa - Lớp DH24TH2\_NhómTH2\_TổTH2**

**AN GIANG, 10-2025**

**Mục Lục**

**[I. GIỚI THIỆU CHUNG 2](#_Toc7624)**

[1. Đặt vấn đề 2](#_Toc12404)

[2. Mục tiêu đề tài 2](#_Toc14829)

[3. Công cụ sử dụng 2](#_Toc7354)

**[II. PHÂN TÍCH & THIẾT KẾ HỆ THỐNG 2](#_Toc27545)**

**[III. CÀI ĐẶT CHƯƠNG TRÌNH 4](#_Toc31320)**

*[Hình 1 : Database QL cưả hàng tivi](#_Toc30838)* [4](#_Toc30838)

*[Hình 2 : Giao diện login](#_Toc27068)* [5](#_Toc27068)

*[Hình 3 : Tab quản lí ti vi](#_Toc21243)* [5](#_Toc21243)

*[Hình 5 : Tab quản lí khách hàng](#_Toc8249)* [6](#_Toc8249)

*[Hình 5 : Tab quản lí tồn kho](#_Toc7427)* [7](#_Toc7427)

*[Hình 6 : Tab quản lí hoá đơn](#_Toc15046)* [7](#_Toc15046)

*[Hình 7 : Tab bán hàng](#_Toc8199)* [8](#_Toc8199)

**[IV. DEMO CODE QUẢN LÍ CỬA HÀNG TIVI 9](#_Toc983)**

**[V. KẾT QUẢ ĐẠT ĐƯỢC 57](#_Toc28705)**

**[VI. HƯỚNG PHÁT TRIỂN 57](#_Toc11458)**

**[VII. KẾT LUẬN 58](#_Toc22605)**

**[VIII. NGUỒN THAM KHẢO 58](#_Toc16276)**

# I. GIỚI THIỆU CHUNG

## 1. Đặt vấn đề

Cửa hàng kinh doanh Tivi cần một hệ thống quản lý hiệu quả để theo dõi sản phẩm, khách hàng, nhân viên và tồn kho. Ứng dụng được xây dựng nhằm giảm thiểu sai sót và tăng năng suất quản lý.

## 2. Mục tiêu đề tài

- Quản lý thông tin sản phẩm Tivi (tên, hãng sản xuất, kích thước, giá, số lượng, bảo hành).  
- Quản lý nhân viên, khách hàng và tồn kho.  
- Lưu trữ và cập nhật dữ liệu trong MySQL thông qua Python.

## 3. Công cụ sử dụng

- Python   
- Tkinter GUI  
- MySQL Workbench và Visual Studio Code  
- Tkcalendar

# II. PHÂN TÍCH & THIẾT KẾ HỆ THỐNG

Hệ thống bao gồm 6 bảng dữ liệu chính:

- Bảng tivi: (mã, tên, hãng SX, kích thước, giá, số lượng, bảo hành).

- Bảng nhanvien: (mã NV, họ tên, chức vụ, lương).

- Bảng khachhang: (mã KH, họ tên, sdt, địa chỉ).

- Bảng tonkho: (mã tồn, mã tivi, ngày cập nhật, số lượng tồn, ghi chú).

- Bảng hoadon: (mã hoá đơn, mã nhân viên, mã khách hàng, ngày lập, tổng tiền).

- Bảng bán hàng

Các bảng liên kết với nhau qua khóa chính và khóa ngoại, trong đó:

- Bảng tồn kho có khóa ngoại trỏ đến bảng tivi.

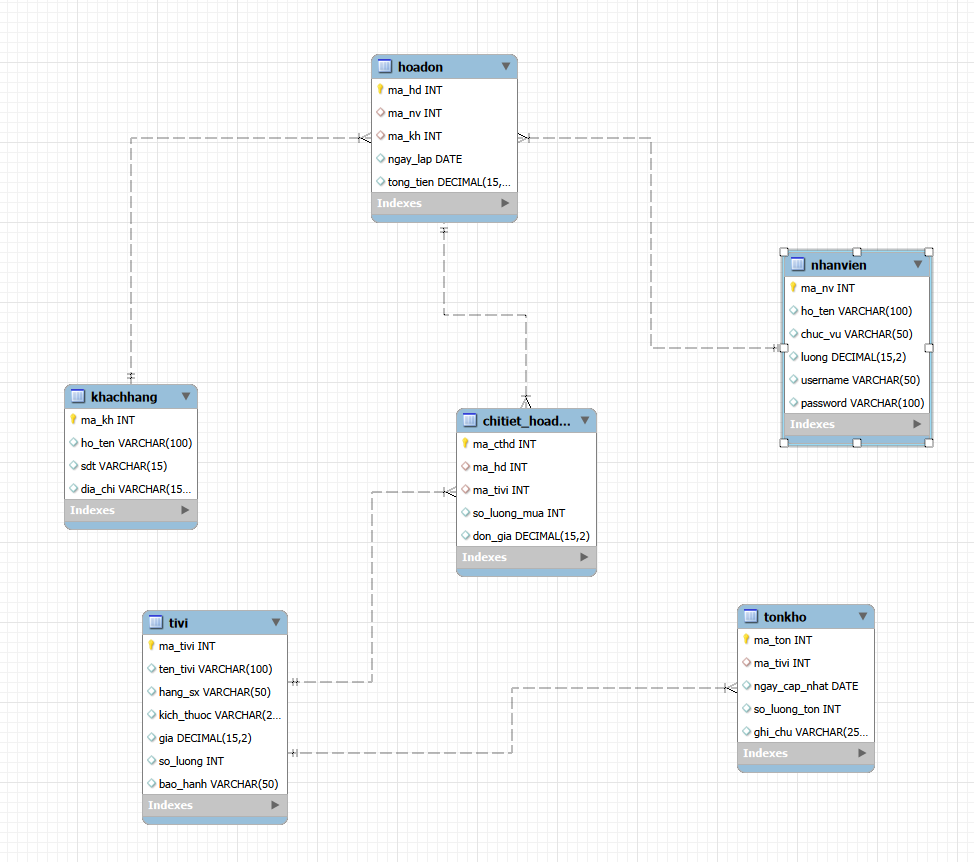
- Bảng nhân viên, khách hàng có khóa ngoại trỏ đến bảng hoá đơn.

- Bảng hoá đơn, tivi có khóa ngoại trỏ đến bảng bán hàng.

# III. CÀI ĐẶT CHƯƠNG TRÌNH

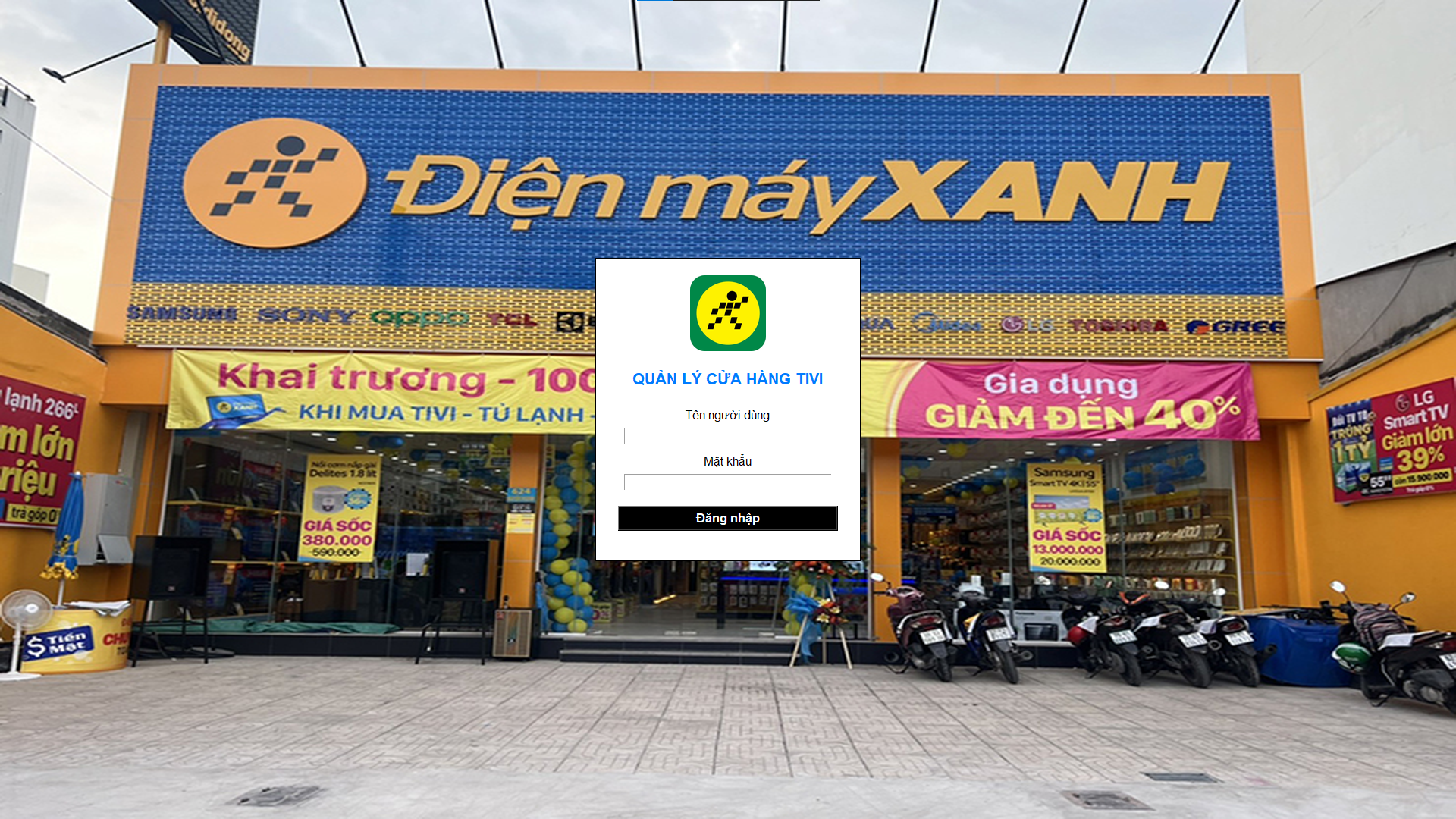
Ứng dụng sử dụng giao diện Tkinter gồm 6 tab chính: Tivi, Nhân viên, Khách hàng và Tồn kho

- Cơ sở dữ liệu :



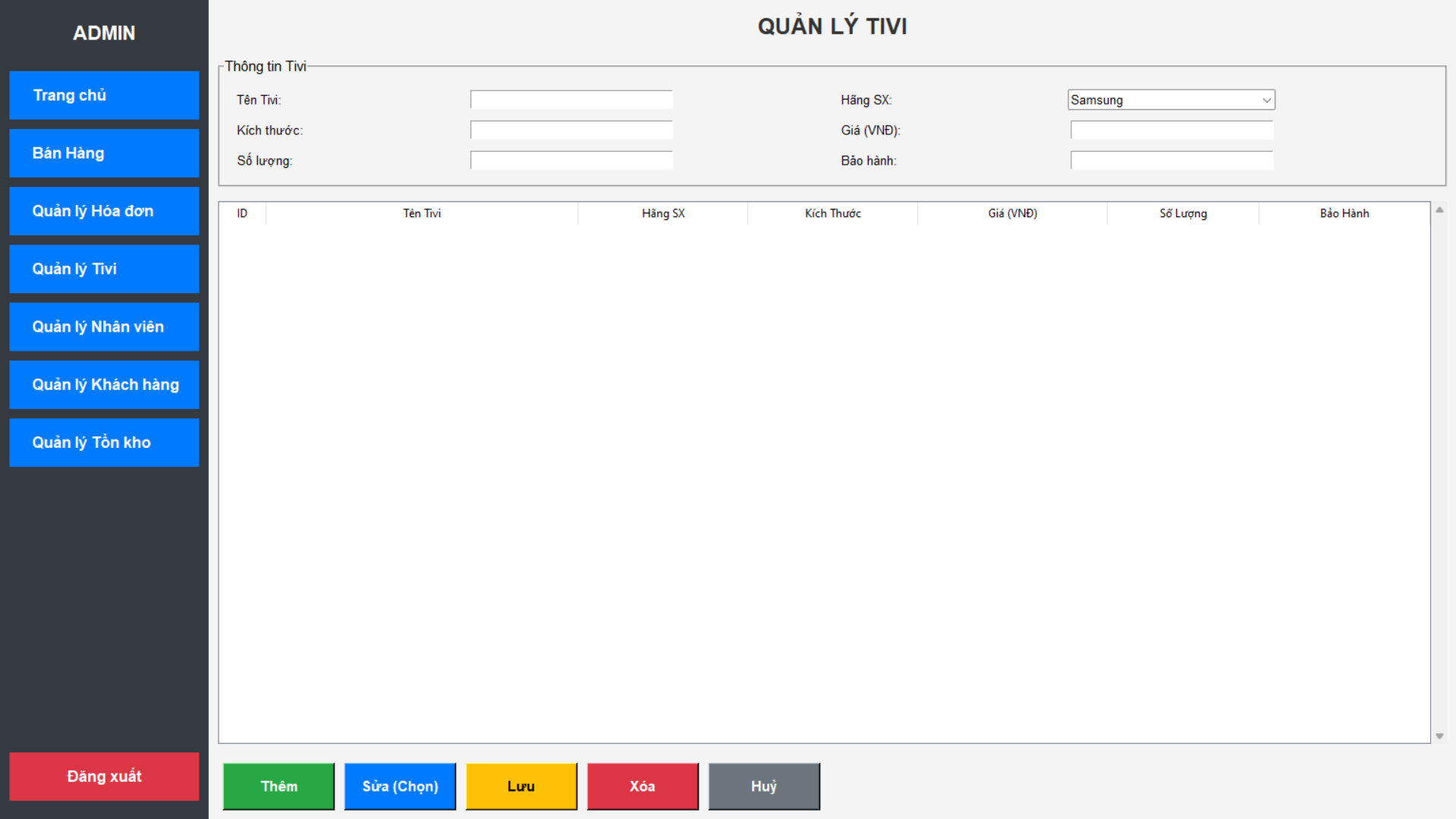
*Hình 1 : Database QL cưả hàng tivi*

- Giao diện đăng nhập :



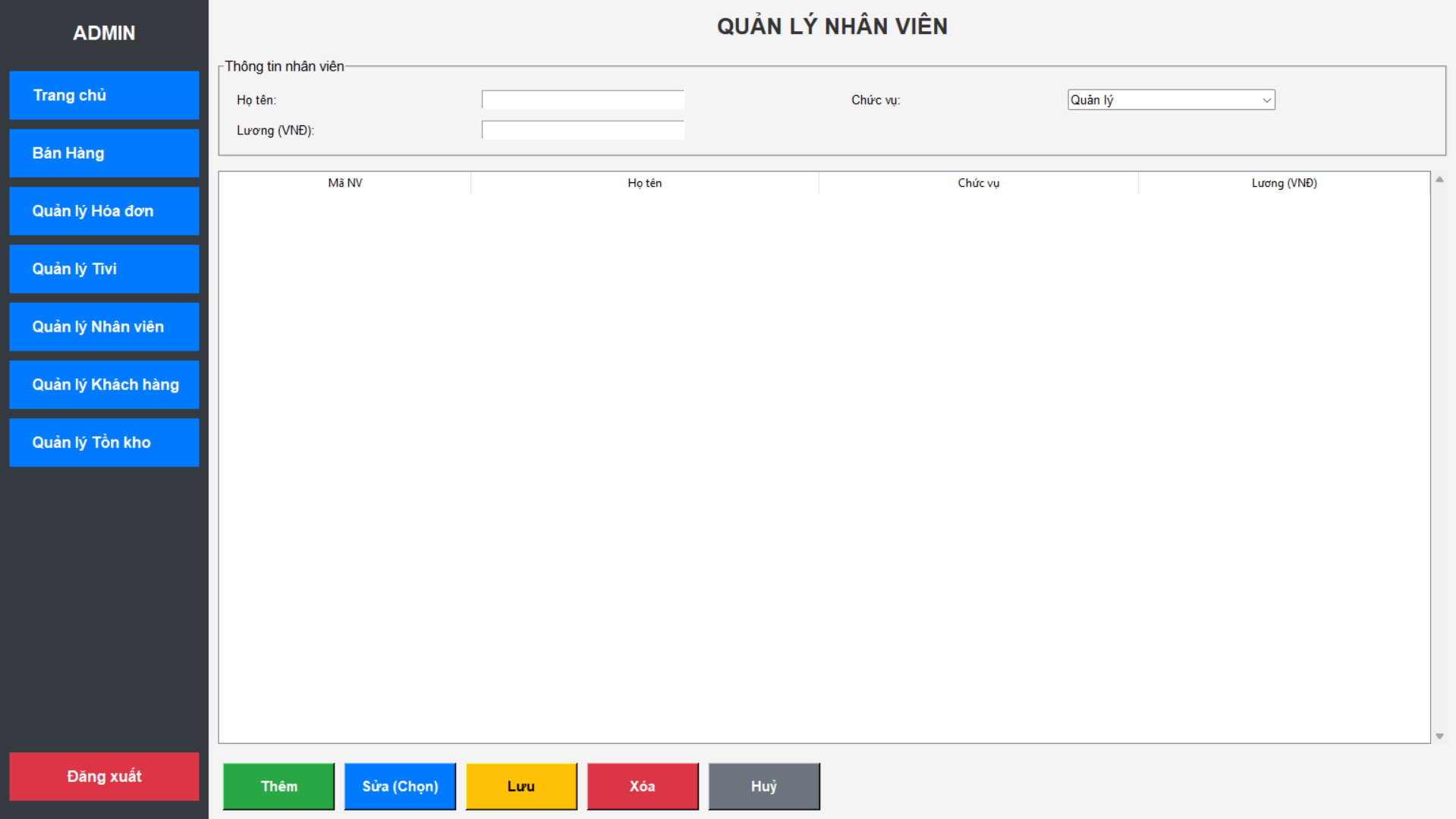
*Hình 2 : Giao diện login*

- Tab Quản lý Tivi: Cho phép thêm, sửa, xóa, lưu dữ liệu tivi; có thêm trường Bảo hành.



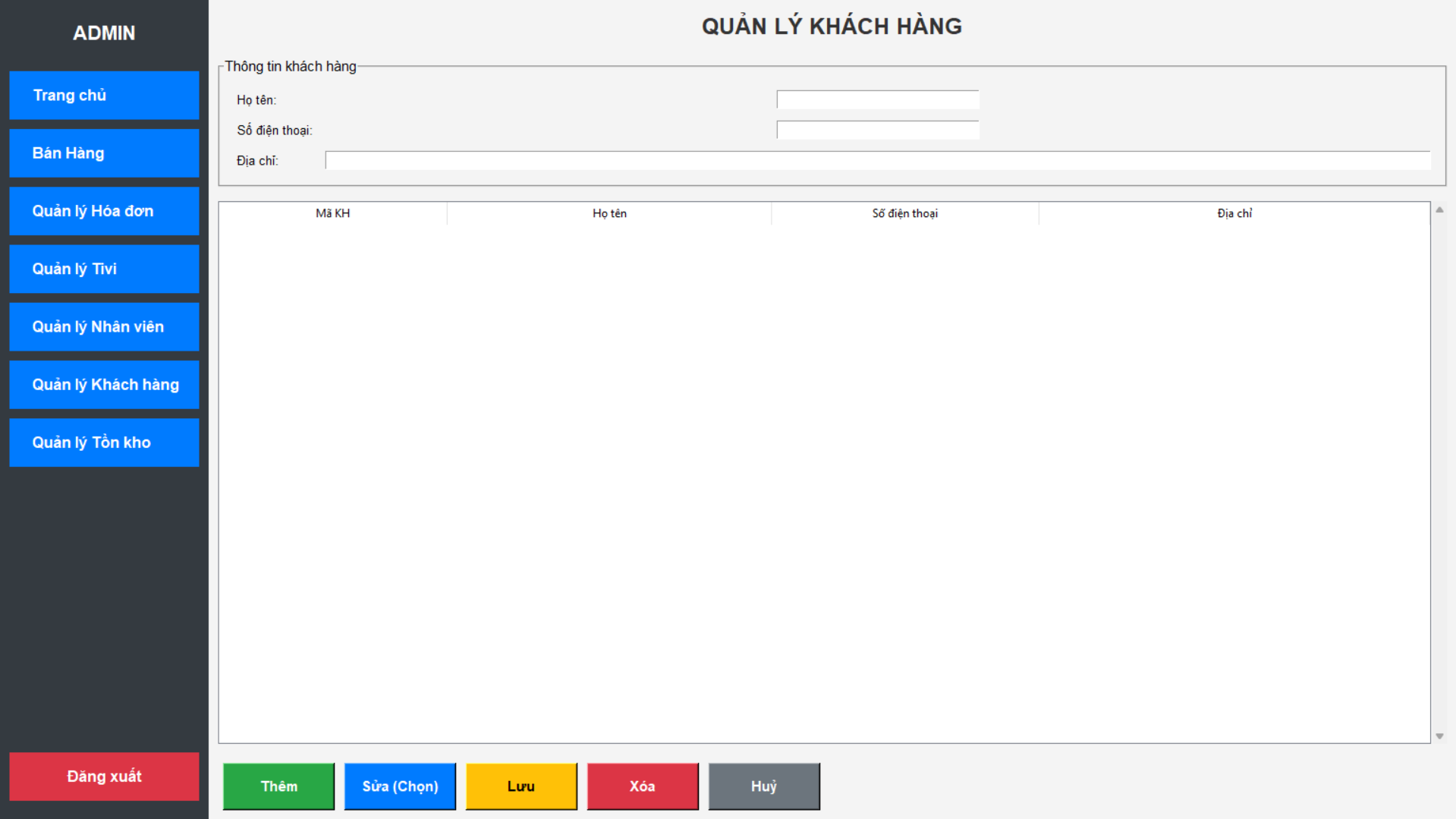
*Hình 3 : Tab quản lí ti vi*

- Tab Quản lý Nhân viên: Hiển thị mã NV, cho phép sửa thông tin, lương, chức vụ.



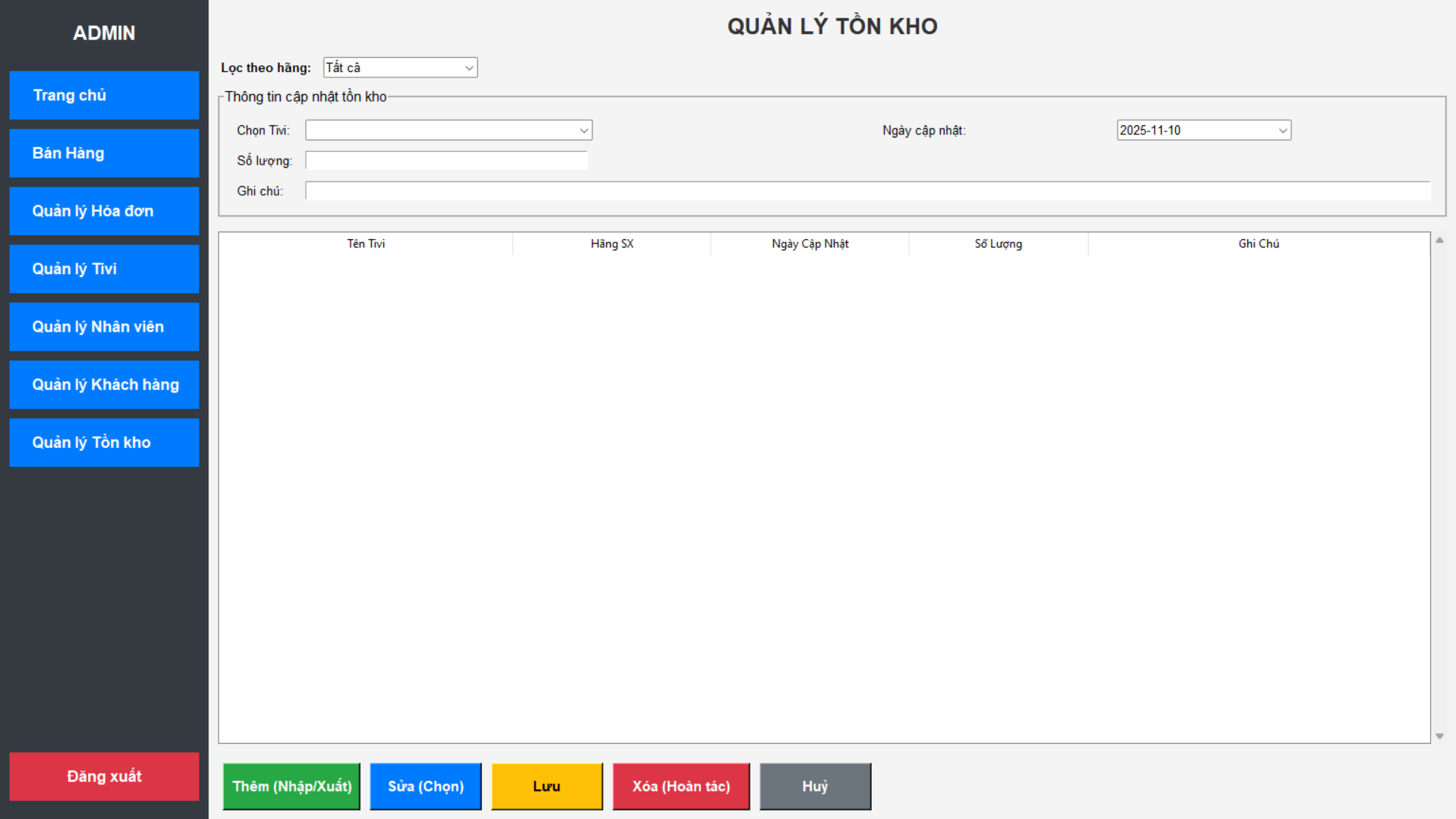
*Hình 4 : Tab quản lí nhân viên*

- Tab Quản lý Khách hàng: Lưu thông tin khách hàng, số điện thoại, địa chỉ.



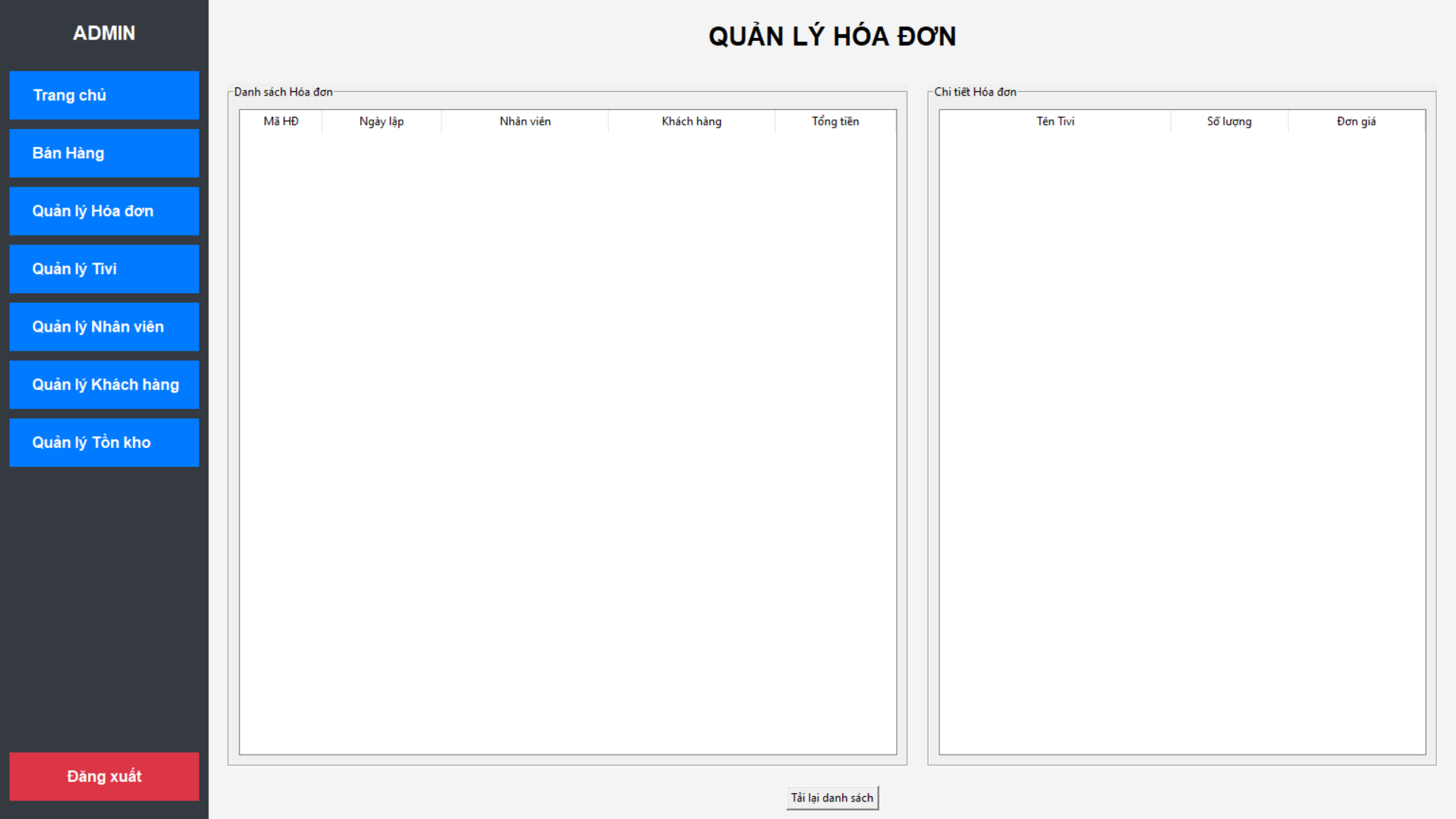
*Hình 5 : Tab quản lí khách hàng*

- Tab Quản lý Tồn kho: Quản lý số lượng tồn, liên kết với bảng tivi theo mã tivi.



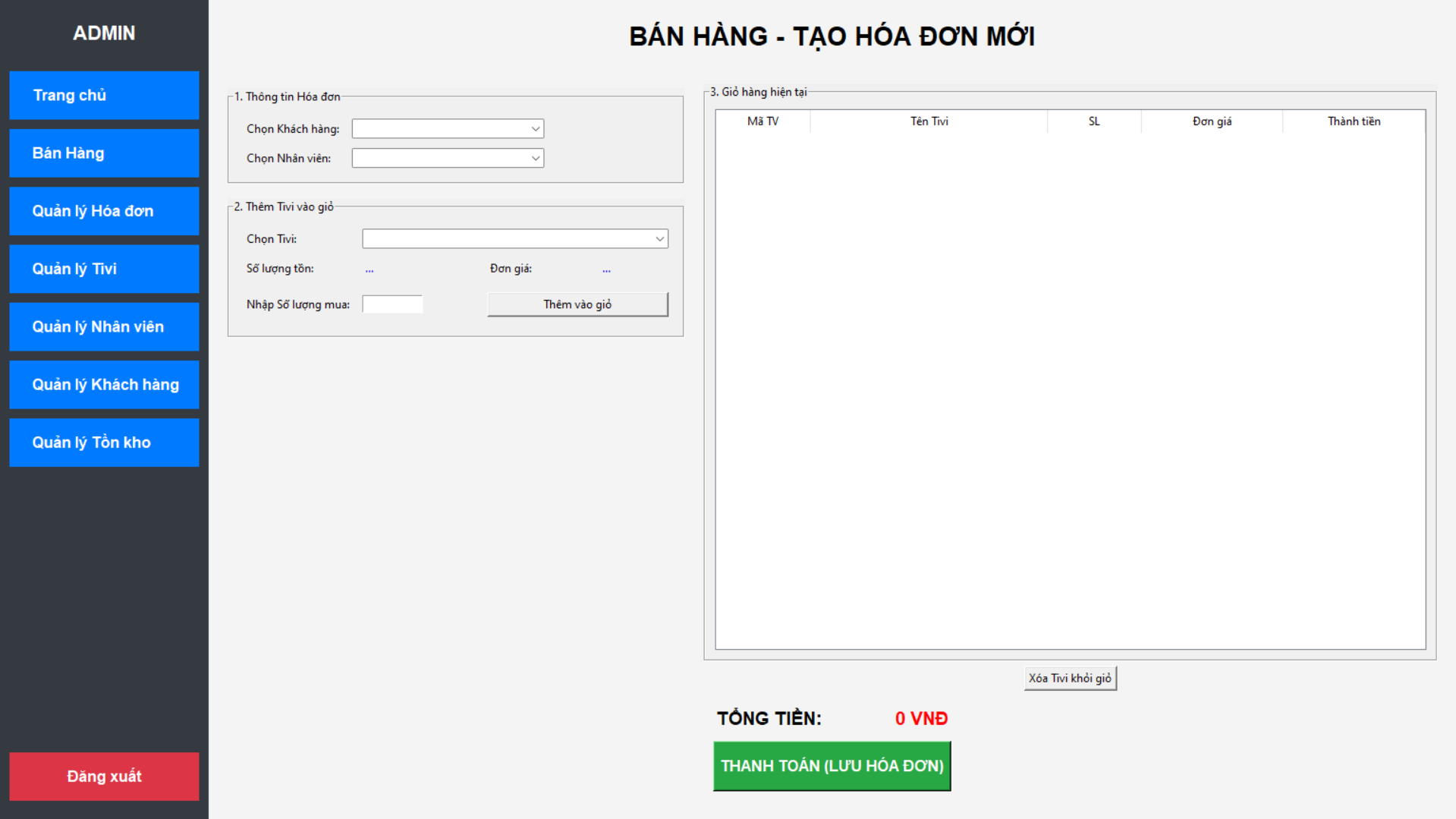
*Hình 5 : Tab quản lí tồn kho*

- Tab Quản lý hoá đơn : bao gồm những hoá đơn đã được thanh toán



*Hình 6 : Tab quản lí hoá đơn*

- Tab bán hàng : bao gồm những thông tin đã nhập từ các bảng, dùng để thanh toán và xuất hoá đơn



*Hình 7 : Tab bán hàng*

Chương trình được chạy trực tiếp trong Visual Studio Code, kết nối cơ sở dữ liệu qua MySQL Connector.

# IV. DEMO CODE QUẢN LÍ CỬA HÀNG TIVI

TAB MAIN ĐỂ CHẠY TẤT CẢ CÁC TAB VÀ GIAO DIỆN LOGIN

import tkinter as tk

from tkinter import ttk, messagebox

from PIL import Image, ImageTk  # cài: python -m pip install pillow

import os

from db import connect\_db

# Import các tab chức năng

import tivi\_tab

import nhanvien\_tab

import khachhang\_tab

import tonkho\_tab

import banhang\_tab

import hoadon\_tab

class App(tk.Tk):

    def \_\_init\_\_(self):

        super().\_\_init\_\_()

        # --- CÀI ĐẶT MÀU SẮC (Cho Giao diện chính) ---

        self.sidebar\_bg = "#343a40"

        self.button\_bg = "#007bff"

        self.fg\_color = "white"

        self.main\_bg = "#f4f4f4"

        # Tạo giao diện đăng nhập đầu tiên

        self.\_create\_login\_ui()

    def \_create\_login\_ui(self):

        ## Tạo giao diện đăng nhập

        self.title("Đăng nhập - Quản lý cửa hàng tivi")

        self.attributes('-fullscreen', True) # Fullscreen

        self.resizable(False, False)

        # Đường dẫn cơ sở để tải ảnh

        base\_path = os.path.dirname(\_\_file\_\_)

        # --- Ảnh nền ---

        try:

            bg\_img\_path = os.path.join(base\_path, "anhgiaodien", "dienmayxanh\_background.jpg")

            bg\_image = Image.open(bg\_img\_path)

            # Lấy kích thước màn hình để resize ảnh nền

            screen\_width = self.winfo\_screenwidth()

            screen\_height = self.winfo\_screenheight()

            bg\_image = bg\_image.resize((screen\_width, screen\_height), Image.LANCZOS)

            self.bg\_photo = ImageTk.PhotoImage(bg\_image) # Lưu lại tham chiếu

            self.bg\_label = tk.Label(self, image=self.bg\_photo)

            self.bg\_label.place(x=0, y=0, relwidth=1, relheight=1)

        except Exception as e:

            print(f"Lỗi tải ảnh nền: {e}. Dùng nền trơn.")

            self.config(bg="#F0F0F0")

        # --- 2. Khung đăng nhập ---

        self.login\_frame = tk.Frame(self, bg="white", relief="solid", bd=1)

        self.login\_frame.place(relx=0.5, rely=0.5, anchor="center", width=350, height=400)

        # --- 3. Logo ---

        try:

            logo\_img\_path = os.path.join(base\_path, "anhgiaodien", "logodienmayxanh.jpg")

            logo\_image = Image.open(logo\_img\_path)

            logo\_image = logo\_image.resize((100, 100), Image.LANCZOS)

            self.logo\_photo = ImageTk.PhotoImage(logo\_image) # Lưu lại tham chiếu

            logo\_label = tk.Label(self.login\_frame, image=self.logo\_photo, bg="white")

            logo\_label.pack(pady=(20, 10))

        except Exception as e:

            print(f"Lỗi tải logo: {e}")

            logo\_label = tk.Label(self.login\_frame, text="Logo", bg="white")

            logo\_label.pack(pady=(20, 10))

        # --- Tiêu đề  ---

        title\_label = tk.Label(self.login\_frame, text="QUẢN LÝ CỬA HÀNG TIVI",

                               font=("Arial", 16, "bold"), bg="white", fg="#007bff")

        title\_label.pack(pady=10)

        # --- Form nhập liệu ---

        user\_label = tk.Label(self.login\_frame, text="Tên người dùng", font=("Arial", 12), bg="white")

        user\_label.pack(pady=(10, 5))

        # Lưu entry vào 'self' để \_check\_login có thể truy cập

        self.entry\_user = tk.Entry(self.login\_frame, font=("Arial", 12), width=30)

        self.entry\_user.pack()

        pass\_label = tk.Label(self.login\_frame, text="Mật khẩu", font=("Arial", 12), bg="white")

        pass\_label.pack(pady=(10, 5))

        self.entry\_pass = tk.Entry(self.login\_frame, font=("Arial", 12), width=30, show="\*")

        self.entry\_pass.pack()

        # --- Nút Đăng nhập ---

        login\_button = tk.Button(self.login\_frame, text="Đăng nhập", font=("Arial", 12, "bold"),

                                 width=28, command=self.\_check\_login, bg="#000000", fg="white")

        login\_button.pack(pady=20)

        # Gán phím Enter để đăng nhập

        self.bind('<Return>', self.\_check\_login\_event)

    def \_check\_login\_event(self, event):

        # Xử lý sự kiện nhấn Enter

        self.\_check\_login()

    def \_check\_login(self):

        # Lấy thông tin đăng nhập

        username = self.entry\_user.get()

        password = self.entry\_pass.get()

        conn = None

        try:

            conn = connect\_db()

            cursor = conn.cursor(dictionary=True)

            # Kiểm tra trong bảng 'nhanvien'

            query = "SELECT ho\_ten FROM nhanvien WHERE username = %s AND password = %s"

            cursor.execute(query, (username, password))

            user = cursor.fetchone()

            if user:

                # ĐĂNG NHẬP THÀNH CÔNG

                messagebox.showinfo("Thành công", f"Chào mừng {user['ho\_ten']}!")

                # Xóa UI đăng nhập và xây dựng Giao diện chính

                self.\_build\_main\_app()

            else:

                messagebox.showerror("Lỗi Đăng Nhập", "Tên người dùng hoặc mật khẩu không đúng!")

        except Exception as e:

            messagebox.showerror("Lỗi CSDL", f"Không thể kiểm tra đăng nhập.\nLỗi: {e}\nĐảm bảo CSDL đang chạy và bảng 'nhanvien' có cột 'username', 'password'.")

        finally:

            if conn:

                conn.close()

    def \_build\_main\_app(self):

        #  Xóa giao diện đăng nhập

        self.unbind('<Return>') # Hủy gán phím Enter

        if hasattr(self, 'bg\_label'): # Xóa ảnh nền

            self.bg\_label.destroy()

        self.login\_frame.destroy() # Xóa khung trắng

        #  Cấu hình lại cửa sổ cho Giao diện chính

        self.title("Hệ Thống Quản Lý Cửa Hàng Tivi NK")

        self.attributes('-fullscreen', True)

        self.resizable(True, True)

        self.minsize(900, 600)

        #  Vẽ Giao diện chính với sidebar và khung chính

        self.\_create\_sidebar()

        self.\_create\_main\_frame()

        # Hiển thị trang chủ mặc định

        self.show\_trangchu\_view()

    def \_create\_sidebar(self):

        # Tạo khung sidebar bên trái

        self.sidebar\_frame = tk.Frame(self, bg=self.sidebar\_bg, width=220)

        self.sidebar\_frame.pack(side="left", fill="y")

        self.sidebar\_frame.pack\_propagate(False)

        tk.Label(self.sidebar\_frame, text="Chào sếp ạ!",

                 font=("Arial", 16, "bold"),

                 bg=self.sidebar\_bg, fg=self.fg\_color).pack(pady=20, padx=10)

        # Các nút điều hướng

        buttons\_info = [

            ("Trang chủ", self.show\_trangchu\_view),

            ("Bán Hàng", self.show\_banhang\_view),

            ("Quản lý Hóa đơn", self.show\_hoadon\_view),

            ("Quản lý Tivi", self.show\_tivi\_view),

            ("Quản lý Nhân viên", self.show\_nhanvien\_view),

            ("Quản lý Khách hàng", self.show\_khachhang\_view),

            ("Quản lý Tồn kho", self.show\_tonkho\_view),

        ]

        for text, command in buttons\_info:

            btn = tk.Button(

                self.sidebar\_frame, text=text, bg=self.button\_bg, fg=self.fg\_color,

                font=("Arial", 12, "bold"), relief="flat", anchor="w",

                padx=20, pady=10, command=command

            )

            btn.pack(fill="x", pady=5, padx=10)

        # Nút Đăng xuất

        logout\_btn = tk.Button(

            self.sidebar\_frame, text="Đăng xuất", bg="#dc3545", fg=self.fg\_color,

            font=("Arial", 12, "bold"), relief="flat", anchor="center",

            padx=20, pady=10, command=self.on\_logout

        )

        logout\_btn.pack(side="bottom", fill="x", pady=20, padx=10)

    def \_create\_main\_frame(self):

        # Tạo khung chính bên phải để hiển thị nội dung

        self.main\_frame = tk.Frame(self, bg=self.main\_bg)

        self.main\_frame.pack(side="right", fill="both", expand=True)

    def clear\_main\_frame(self):

        # Xóa hết widget trong khung chính

        for widget in self.main\_frame.winfo\_children():

            widget.destroy()

    # --- CÁC HÀM CHUYỂN ĐỔI VIEW ---

    def show\_trangchu\_view(self):

        self.clear\_main\_frame()

        tk.Label(self.main\_frame, text="TRANG CHỦ",

                 font=("Arial", 40, "bold"), bg=self.main\_bg,

                 fg="#333").pack(expand=True)

        tk.Label(self.main\_frame, text="Chào mừng đến với hệ thống quản lý cửa hàng tivi NK!",

                 font=("Arial", 30), bg=self.main\_bg).pack(expand=True)

    def show\_banhang\_view(self):

        self.clear\_main\_frame()

        banhang\_tab.create\_view(self.main\_frame)

    def show\_hoadon\_view(self):

        self.clear\_main\_frame()

        hoadon\_tab.create\_view(self.main\_frame)

    def show\_tivi\_view(self):

        self.clear\_main\_frame()

        tivi\_tab.create\_view(self.main\_frame)

    def show\_nhanvien\_view(self):

        self.clear\_main\_frame()

        nhanvien\_tab.create\_view(self.main\_frame)

    def show\_khachhang\_view(self):

        self.clear\_main\_frame()

        khachhang\_tab.create\_view(self.main\_frame)

    def show\_tonkho\_view(self):

        self.clear\_main\_frame()

        tonkho\_tab.create\_view(self.main\_frame)

    def on\_logout(self):

        # Xử lý đăng xuất

        if messagebox.askyesno("Đăng xuất", "Bạn có chắc muốn đăng xuất?"):

            self.destroy()

#  ĐIỂM KHỞI CHẠY CHÍNH CỦA ỨNG DỤNG

if \_\_name\_\_ == "\_\_main\_\_":

    # Khởi tạo App.

    app = App()

    app.mainloop()

TAB QUẢN LÍ TIVI

import tkinter as tk

from tkinter import ttk, messagebox

from db import connect\_db

def create\_view(parent\_frame):

    # --- Màu sắc ---

    bg\_color = "#f4f4f4" # Màu nền chính

    btn\_add\_bg = "#28a745"   # Màu nút Thêm

    btn\_edit\_bg = "#007bff"  # Màu nút Sửa

    btn\_save\_bg = "#ffc107"  # Màu nút Lưu

    btn\_del\_bg = "#dc3545"   # Màu nút Xóa

    btn\_clear\_bg = "#6c757d" # Màu nút Huỷ

    btn\_fg\_white = "white" # Màu chữ trắng

    btn\_fg\_black = "black"    # Màu chữ đen

    # --- Tiêu đề ---

    tk.Label(parent\_frame, text="QUẢN LÝ TIVI TRƯNG BÀY",

             font=("Arial", 18, "bold"),

             bg=bg\_color, fg="#333").pack(pady=10)

    # --- Khung nhập dữ liệu ---

    frm\_input = tk.LabelFrame(parent\_frame, text="Thông tin Tivi",

                                  padx=10, pady=10, bg=bg\_color,

                                  font=("Arial", 11))

    frm\_input.pack(padx=10, pady=5, fill="x")

    # Entry ẩn để chứa ma\_tivi khi chọn

    tk.Label(frm\_input, text="Tên Tivi:", bg=bg\_color, font=("Arial", 10)).grid(row=0, column=0, padx=5, pady=5, sticky="w")

    entry\_ten = tk.Entry(frm\_input, width=30, font=("Arial", 10))

    entry\_ten.grid(row=0, column=1, padx=5, pady=5)

    tk.Label(frm\_input, text="Hãng SX:", bg=bg\_color, font=("Arial", 10)).grid(row=0, column=2, padx=5, pady=5, sticky="w")

    combo\_hang = ttk.Combobox(frm\_input, width=28, state="readonly", font=("Arial", 10))

    combo\_hang['values'] = ("Samsung", "LG", "Sony", "TCL", "Panasonic", "Sharp", "Asanzo")

    combo\_hang.current(0)

    combo\_hang.grid(row=0, column=3, padx=5, pady=5)

    tk.Label(frm\_input, text="Kích thước:", bg=bg\_color, font=("Arial", 10)).grid(row=1, column=0, padx=5, pady=5, sticky="w")

    entry\_kichthuoc = tk.Entry(frm\_input, width=30, font=("Arial", 10))

    entry\_kichthuoc.grid(row=1, column=1, padx=5, pady=5)

    tk.Label(frm\_input, text="Giá (VNĐ):", bg=bg\_color, font=("Arial", 10)).grid(row=1, column=2, padx=5, pady=5, sticky="w")

    entry\_gia = tk.Entry(frm\_input, width=30, font=("Arial", 10))

    entry\_gia.grid(row=1, column=3, padx=5, pady=5)

    tk.Label(frm\_input, text="Số lượng:", bg=bg\_color, font=("Arial", 10)).grid(row=2, column=0, padx=5, pady=5, sticky="w")

    entry\_sl = tk.Entry(frm\_input, width=30, font=("Arial", 10))

    entry\_sl.grid(row=2, column=1, padx=5, pady=5)

    tk.Label(frm\_input, text="Bảo hành:", bg=bg\_color, font=("Arial", 10)).grid(row=2, column=2, padx=5, pady=5, sticky="w")

    entry\_bh = tk.Entry(frm\_input, width=30, font=("Arial", 10))

    entry\_bh.grid(row=2, column=3, padx=5, pady=5)

    # Căn chỉnh 2 cột trong grid

    frm\_input.grid\_columnconfigure(1, weight=1)

    frm\_input.grid\_columnconfigure(3, weight=1)

    # --- Khung nút bấm ---

    frame\_btn\_bottom = tk.Frame(parent\_frame, bg=bg\_color)

    frame\_btn\_bottom.pack(side="bottom", fill="x", padx=10, pady=10)

    # --- Khung Treeview ---

    frame\_tree = tk.Frame(parent\_frame)

    frame\_tree.pack(padx=10, pady=10, fill="both", expand=True)

    # --- Treeview ---

    cols = ("ma\_tivi", "ten\_tivi", "hang\_sx", "kich\_thuoc", "gia", "so\_luong", "bao\_hanh")

    tree = ttk.Treeview(frame\_tree, columns=cols, show="headings", height=10)

    # Định nghĩa tiêu đề và độ rộng cột

    tree.heading("ma\_tivi", text="ID")

    tree.heading("ten\_tivi", text="Tên Tivi")

    tree.heading("hang\_sx", text="Hãng SX")

    tree.heading("kich\_thuoc", text="Kích Thước")

    tree.heading("gia", text="Giá (VNĐ)")

    tree.heading("so\_luong", text="Số Lượng")

    tree.heading("bao\_hanh", text="Bảo Hành")

    # Định nghĩa độ rộng cột

    tree.column("ma\_tivi", width=50, stretch=tk.NO, anchor="center")

    tree.column("ten\_tivi", width=250,anchor="center")

    tree.column("hang\_sx", width=100, anchor="center")

    tree.column("kich\_thuoc", width=100, anchor="center")

    tree.column("gia", width=120, anchor="center")

    tree.column("so\_luong", width=80, anchor="center")

    tree.column("bao\_hanh", width=100, anchor="center")

    # --- Thanh cuộn dọc ---

    scroll\_y = ttk.Scrollbar(frame\_tree, orient="vertical", command=tree.yview)

    tree.configure(yscrollcommand=scroll\_y.set)

    scroll\_y.pack(side="right", fill="y")

    tree.pack(side="left", fill="both", expand=True)

    # --- Các hàm xử lý ---

    def load\_data():

        # Tải dữ liệu từ CSDL lên Treeview

        tree.delete(\*tree.get\_children())

        try:

            conn = connect\_db()

            cur = conn.cursor()

            cur.execute("""

                SELECT ma\_tivi, ten\_tivi, hang\_sx, kich\_thuoc, gia, so\_luong, bao\_hanh

                FROM tivi

                ORDER BY ma\_tivi ASC

            """)

            for row in cur.fetchall():

                # Định dạng số tiền và số lượng

                formatted\_row = (

                    row[0], row[1], row[2], row[3],

                    f"{row[4]:,.0f}",  # Định dạng tiền tệ

                    f"{row[5]:,}",     # Định dạng số lượng

                    row[6]

                )

                tree.insert("", tk.END, values=formatted\_row)

            conn.close()

        except Exception as e:

            messagebox.showerror("Lỗi", f"Không thể tải dữ liệu: {e}")

    def \_clear\_entries():

        # Xóa nội dung các trường nhập liệu

        entry\_ten.delete(0, tk.END)

        entry\_kichthuoc.delete(0, tk.END)

        entry\_gia.delete(0, tk.END)

        entry\_sl.delete(0, tk.END)

        entry\_bh.delete(0, tk.END)

        combo\_hang.current(0)

    def clear\_form():

        # Xóa form và bỏ chọn trên Treeview

        \_clear\_entries()

        if tree.selection():

            tree.selection\_remove(tree.selection())

    def on\_tree\_select(event=None):

        """Hàm "Chọn": được gọi khi click vào Treeview hoặc nút Sửa"""

        sel = tree.selection()

        if not sel:

            if event is None:

                messagebox.showwarning("Chưa chọn", "Vui lòng chọn Tivi trên bảng!")

            return

        \_clear\_entries()

        # Lấy dữ liệu Tivi đã chọn

        item = tree.item(sel[0])

        values = item["values"]

        # Điền dữ liệu vào form

        entry\_ten.insert(0, values[1])

        combo\_hang.set(values[2])

        entry\_kichthuoc.insert(0, values[3])

        # Xóa dấu phẩy khi điền lại form

        entry\_gia.insert(0, str(values[4]).replace(",", ""))

        entry\_sl.insert(0, str(values[5]).replace(",", ""))

        entry\_bh.insert(0, values[6])

    def sua():

        # Gọi hàm chọn khi nhấn nút Sửa

        on\_tree\_select(event=None)

    def get\_form\_data():

        # Lấy và xác thực dữ liệu từ form

        ten\_tivi = entry\_ten.get().strip()

        hang\_sx = combo\_hang.get().strip()

        kich\_thuoc = entry\_kichthuoc.get().strip()

        # Xử lý số có dấu phẩy

        gia\_str = entry\_gia.get().strip().replace(",", "")

        so\_luong\_str = entry\_sl.get().strip().replace(",", "")

        bao\_hanh = entry\_bh.get().strip()

        if not ten\_tivi or not hang\_sx or not kich\_thuoc or not gia\_str or not so\_luong\_str:

            messagebox.showwarning("Thiếu dữ liệu", "Vui lòng nhập đầy đủ thông tin (trừ Bảo hành)!")

            return None

        try:

            gia = float(gia\_str)

            so\_luong = int(so\_luong\_str)

            if gia < 0 or so\_luong < 0:

                raise ValueError("Giá và số lượng phải là số không âm")

        except ValueError as e:

            messagebox.showerror("Lỗi dữ liệu", f"Giá và Số lượng phải là số hợp lệ! \n{e}")

            return None

        return (ten\_tivi, hang\_sx, kich\_thuoc, gia, so\_luong, bao\_hanh)

    def them():

        # Thêm Tivi mới

        data = get\_form\_data()

        if data is None:

            return

        conn = connect\_db()

        cur = conn.cursor()

        try:

            cur.execute("""

                INSERT INTO tivi (ten\_tivi, hang\_sx, kich\_thuoc, gia, so\_luong, bao\_hanh)

                VALUES (%s, %s, %s, %s, %s, %s)

            """, data)

            conn.commit()

            messagebox.showinfo("Thành công", "Đã thêm Tivi mới!")

        except Exception as e:

            conn.rollback()

            messagebox.showerror("Lỗi SQL", str(e))

        finally:

            conn.close()

        load\_data()

        clear\_form()

    def luu():

        # Lưu thay đổi cho Tivi đã chọn

        sel = tree.selection()

        if not sel:

            messagebox.showwarning("Chưa chọn", "Vui lòng chọn Tivi để sửa!")

            return

        ma\_tivi\_can\_sua = tree.item(sel[0])["values"][0]

        data = get\_form\_data()

        if data is None:

            return

        full\_data = data + (ma\_tivi\_can\_sua,) # Thêm mã Tivi vào cuối tuple

        conn = connect\_db()

        cur = conn.cursor()

        try:

            cur.execute("""

                UPDATE tivi

                SET ten\_tivi=%s, hang\_sx=%s, kich\_thuoc=%s, gia=%s, so\_luong=%s, bao\_hanh=%s

                WHERE ma\_tivi=%s

            """, full\_data)

            conn.commit()

            messagebox.showinfo("Thành công", "Đã cập nhật thông tin Tivi!")

        except Exception as e:

            conn.rollback()

            messagebox.showerror("Lỗi SQL", str(e))

        finally:

            conn.close()

        load\_data()

        clear\_form()

    def xoa():

        # Xóa Tivi đã chọn (với xử lý lỗi Khóa ngoại)

        sel = tree.selection()

        if not sel:

            messagebox.showwarning("Chưa chọn", "Vui lòng chọn Tivi để xóa!")

            return

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

        if not messagebox.askyesno("Xác nhận", f"Bạn có chắc muốn xóa Tivi (ID: {ma})?"):

            return

        conn = connect\_db()

        cur = conn.cursor()

        try:

            cur.execute("DELETE FROM tivi WHERE ma\_tivi=%s", (ma,))

            conn.commit()

            messagebox.showinfo("Thành công", "Đã xóa Tivi!")

        except Exception as e:

            conn.rollback()

            # --- BẮT LỖI KHÓA NGOẠI ---

            if e.args and e.args[0] == 1451:

                messagebox.showerror(

                    "Không thể xóa",

                    "Không thể xóa Tivi này vì đã có lịch sử bán hàng (tồn tại trong hóa đơn).\n\n"

                    "Bạn chỉ có thể xóa các Tivi chưa từng được bán."

                )

            else:

                # Báo các lỗi SQL khác

                messagebox.showerror("Lỗi SQL", f"Không thể xóa: {e}")

            # --- KẾT THÚC THAY ĐỔI ---

        finally:

            if conn:

                conn.close()

        load\_data()

        clear\_form()

    # Gán sự kiện click vào Treeview

    tree.bind("<<TreeviewSelect>>", on\_tree\_select)

    # --- Các nút bấm  ---

    btn\_them = tk.Button(frame\_btn\_bottom, text="Thêm", command=them,

                         width=12, bg=btn\_add\_bg, fg=btn\_fg\_white,

                         font=("Arial", 11, "bold"), height=2)

    btn\_them.pack(side="left", padx=5)

    btn\_sua = tk.Button(frame\_btn\_bottom, text="Sửa (Chọn)", command=sua,

                         width=12, bg=btn\_edit\_bg, fg=btn\_fg\_white,

                         font=("Arial", 11, "bold"), height=2)

    btn\_sua.pack(side="left", padx=5)

    btn\_luu = tk.Button(frame\_btn\_bottom, text="Lưu", command=luu,

                         width=12, bg=btn\_save\_bg, fg=btn\_fg\_black,

                         font=("Arial", 11, "bold"), height=2)

    btn\_luu.pack(side="left", padx=5)

    btn\_xoa = tk.Button(frame\_btn\_bottom, text="Xóa", command=xoa,

                         width=12, bg=btn\_del\_bg, fg=btn\_fg\_white,

                         font=("Arial", 11, "bold"), height=2)

    btn\_xoa.pack(side="left", padx=5)

    btn\_clear = tk.Button(frame\_btn\_bottom, text="Huỷ", command=clear\_form,

                          width=12, bg=btn\_clear\_bg, fg=btn\_fg\_white,

                          font=("Arial", 11, "bold"), height=2)

    btn\_clear.pack(side="left", padx=5)

    # Tải dữ liệu ban đầu

    load\_data()

    clear\_form()

TAB QUẢN LÍ NHÂN VIÊN

import tkinter as tk

from tkinter import ttk, messagebox

from db import connect\_db

def create\_view(parent\_frame):

    # --- Màu sắc ---

    bg\_color = "#f4f4f4"

    btn\_add\_bg = "#28a745"

    btn\_edit\_bg = "#007bff"

    btn\_save\_bg = "#ffc107"

    btn\_del\_bg = "#dc3545"

    btn\_clear\_bg = "#6c757d"

    btn\_fg\_white = "white"

    btn\_fg\_black = "black"

    # --- Tiêu đề ---

    tk.Label(parent\_frame, text="QUẢN LÝ NHÂN VIÊN",

             font=("Arial", 18, "bold"),

             bg=bg\_color, fg="#333").pack(pady=10)

    # --- Khung nhập liệu ---

    frm\_input = tk.LabelFrame(parent\_frame, text="Thông tin nhân viên",

                              padx=10, pady=10, bg=bg\_color,

                              font=("Arial", 11))

    frm\_input.pack(padx=10, pady=5, fill="x")

    entry\_ma = tk.Entry(frm\_input)

    # --- Hàng 1 ---

    tk.Label(frm\_input, text="Họ tên:", bg=bg\_color, font=("Arial", 10)).grid(row=0, column=0, padx=5, pady=5, sticky="w")

    entry\_ten = tk.Entry(frm\_input, width=30, font=("Arial", 10))

    entry\_ten.grid(row=0, column=1, padx=5, pady=5)

    tk.Label(frm\_input, text="Chức vụ:", bg=bg\_color, font=("Arial", 10)).grid(row=0, column=2, padx=5, pady=5, sticky="w")

    combo\_chucvu = ttk.Combobox(frm\_input, width=28, state="readonly", font=("Arial", 10),

        values=["Quản lý", "Thu ngân", "Bán hàng", "Kỹ thuật viên", "Bảo vệ", "Lễ tân", "Dọn vệ sinh"])

    combo\_chucvu.grid(row=0, column=3, padx=5, pady=5)

    combo\_chucvu.current(0)

    # --- Hàng 2 ---

    tk.Label(frm\_input, text="Lương (VNĐ):", bg=bg\_color, font=("Arial", 10)).grid(row=1, column=0, padx=5, pady=5, sticky="w")

    entry\_luong = tk.Entry(frm\_input, width=30, font=("Arial", 10))

    entry\_luong.grid(row=1, column=1, padx=5, pady=5)

    frm\_input.grid\_columnconfigure(1, weight=1)

    frm\_input.grid\_columnconfigure(3, weight=1)

    # --- Khung nút ở dưới cùng ---

    frame\_btn\_bottom = tk.Frame(parent\_frame, bg=bg\_color)

    frame\_btn\_bottom.pack(side="bottom", fill="x", padx=10, pady=10)

    # --- Khung Treeview ---

    frame\_tree = tk.Frame(parent\_frame)

    frame\_tree.pack(padx=10, pady=10, fill="both", expand=True)

    cols = ("ma\_nv", "ho\_ten", "chuc\_vu", "luong")

    tree = ttk.Treeview(frame\_tree, columns=cols, show="headings", height=12)

    tree.heading("ma\_nv", text="Mã NV")

    tree.heading("ho\_ten", text="Họ tên")

    tree.heading("chuc\_vu", text="Chức vụ")

    tree.heading("luong", text="Lương (VNĐ)")

    tree.column("ma\_nv", width=80, anchor="center")

    tree.column("ho\_ten", width=180, anchor="center")

    tree.column("chuc\_vu", width=150, anchor="center")

    tree.column("luong", width=120, anchor="center")

    scroll\_y = ttk.Scrollbar(frame\_tree, orient="vertical", command=tree.yview)

    tree.configure(yscrollcommand=scroll\_y.set)

    scroll\_y.pack(side="right", fill="y")

    tree.pack(side="left", fill="both", expand=True)

    # --- Các hàm xử lý ---

    def load\_data():

        for i in tree.get\_children():

            tree.delete(i)

        try:

            conn = connect\_db()

            cur = conn.cursor()

            cur.execute("SELECT ma\_nv, ho\_ten, chuc\_vu, luong FROM nhanvien ORDER BY ma\_nv ASC")

            for row in cur.fetchall():

                tree.insert("", tk.END, values=row)

            conn.close()

        except Exception as e:

            messagebox.showerror("Lỗi", f"Không thể tải dữ liệu: {e}")

    def clear\_form():

        entry\_ma.delete(0, tk.END)

        entry\_ten.delete(0, tk.END)

        entry\_luong.delete(0, tk.END)

        combo\_chucvu.current(0)

        if tree.selection():

            tree.selection\_remove(tree.selection())

    def them():

        ho\_ten = entry\_ten.get().strip()

        chuc\_vu = combo\_chucvu.get().strip()

        luong\_str = entry\_luong.get().strip()

        if not ho\_ten or not chuc\_vu or not luong\_str:

            messagebox.showwarning("Thiếu dữ liệu", "Vui lòng nhập đầy đủ thông tin!")

            return

        try:

            luong = float(luong\_str)

            if luong < 0: raise ValueError("Lương phải là số dương")

        except ValueError as e:

            messagebox.showerror("Lỗi dữ liệu", f"Lương phải là số hợp lệ! \n{e}")

            return

        conn = connect\_db()

        cur = conn.cursor()

        try:

            cur.execute(

                "INSERT INTO nhanvien (ho\_ten, chuc\_vu, luong) VALUES (%s,%s,%s)",

                (ho\_ten, chuc\_vu, luong)

            )

            conn.commit()

            messagebox.showinfo("Thành công", "Đã thêm nhân viên mới!")

            clear\_form()

            load\_data()

        except Exception as e:

            conn.rollback()

            messagebox.showerror("Lỗi", f"Không thể thêm dữ liệu: {e}")

        finally:

            conn.close()

    def xoa():

        selected = tree.selection()

        if not selected:

            messagebox.showwarning("Chưa chọn", "Hãy chọn nhân viên để xóa!")

            return

        ma\_nv = tree.item(selected[0])["values"][0]

        if not messagebox.askyesno("Xác nhận", f"Bạn có chắc muốn xóa nhân viên mã {ma\_nv}?"):

            return

        conn = connect\_db()

        cur = conn.cursor()

        try:

            cur.execute("DELETE FROM nhanvien WHERE ma\_nv=%s", (ma\_nv,))

            conn.commit()

            messagebox.showinfo("Thành công", "Đã xóa nhân viên!")

        except Exception as e:

            conn.rollback()

            messagebox.showerror("Lỗi", f"Không thể xóa: {e}")

        finally:

            conn.close()

        clear\_form()

        load\_data()

    def sua(event=None):

        selected = tree.selection()

        if not selected:

            return

        entry\_ten.delete(0, tk.END)

        entry\_luong.delete(0, tk.END)

        vals = tree.item(selected[0])["values"]

        entry\_ma.delete(0, tk.END)

        entry\_ma.insert(0, vals[0])

        entry\_ten.insert(0, vals[1])

        combo\_chucvu.set(vals[2])

        entry\_luong.insert(0, str(vals[3]))

    def luu():

        ma\_nv = entry\_ma.get().strip()

        ho\_ten = entry\_ten.get().strip()

        chuc\_vu = combo\_chucvu.get().strip()

        luong\_str = entry\_luong.get().strip()

        if not ma\_nv:

            messagebox.showwarning("Thiếu mã", "Hãy nhấn 'Sửa' (nhấp vào bảng) để chọn nhân viên trước!")

            return

        if not ho\_ten or not chuc\_vu or not luong\_str:

            messagebox.showwarning("Thiếu dữ liệu", "Vui lòng nhập đầy đủ thông tin!")

            return

        try:

            luong = float(luong\_str)

            if luong < 0: raise ValueError("Lương phải là số dương")

        except ValueError as e:

            messagebox.showerror("Lỗi dữ liệu", f"Lương phải là số hợp lệ! \n{e}")

            return

        conn = connect\_db()

        cur = conn.cursor()

        try:

            cur.execute("""

                UPDATE nhanvien SET ho\_ten=%s, chuc\_vu=%s, luong=%s

                WHERE ma\_nv=%s

            """, (ho\_ten, chuc\_vu, luong, ma\_nv))

            conn.commit()

            messagebox.showinfo("Thành công", "Đã lưu thay đổi!")

            clear\_form()

            load\_data()

        except Exception as e:

            conn.rollback()

            messagebox.showerror("Lỗi", f"Không thể cập nhật dữ liệu: {e}")

        finally:

            conn.close()

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

    # --- Thêm nút vào frame\_btn\_bottom  ---

    btn\_them = tk.Button(frame\_btn\_bottom, text="Thêm", command=them,

                         width=12, bg=btn\_add\_bg, fg=btn\_fg\_white,

                         font=("Arial", 11, "bold"), height=2)

    btn\_them.pack(side="left", padx=5)

    btn\_sua = tk.Button(frame\_btn\_bottom, text="Sửa (Chọn)", command=lambda: sua(None),

                        width=12, bg=btn\_edit\_bg, fg=btn\_fg\_white,

                        font=("Arial", 11, "bold"), height=2)

    btn\_sua.pack(side="left", padx=5)

    btn\_luu = tk.Button(frame\_btn\_bottom, text="Lưu", command=luu,

                        width=12, bg=btn\_save\_bg, fg=btn\_fg\_black,

                        font=("Arial", 11, "bold"), height=2)

    btn\_luu.pack(side="left", padx=5)

    btn\_xoa = tk.Button(frame\_btn\_bottom, text="Xóa", command=xoa,

                        width=12, bg=btn\_del\_bg, fg=btn\_fg\_white,

                        font=("Arial", 11, "bold"), height=2)

    btn\_xoa.pack(side="left", padx=5)

    btn\_clear = tk.Button(frame\_btn\_bottom, text="Xóa form", command=clear\_form,

                          width=12, bg=btn\_clear\_bg, fg=btn\_fg\_white,

                          font=("Arial", 11, "bold"), height=2)

    btn\_clear.pack(side="right", padx=5)

    load\_data()

TAB QUẢN LÍ KHÁCH HÀNG

import tkinter as tk

from tkinter import ttk, messagebox

from db import connect\_db

def create\_view(parent\_frame):

    # --- Màu sắc ---

    bg\_color = "#f4f4f4" # Màu nền khung chính

    btn\_add\_bg = "#28a745" # Màu nút Thêm

    btn\_edit\_bg = "#007bff" # Màu nút Sửa

    btn\_save\_bg = "#ffc107" # Màu nút Lưu

    btn\_del\_bg = "#dc3545" # Màu nút Xóa

    btn\_clear\_bg = "#6c757d" # Màu nút Huỷ

    btn\_fg\_white = "white" # Màu chữ nút trắng

    btn\_fg\_black = "black" # Màu chữ nút đen

    # --- Tiêu đề ---

    tk.Label(parent\_frame, text="QUẢN LÝ KHÁCH HÀNG",

             font=("Arial", 18, "bold"),

             bg=bg\_color, fg="#333").pack(pady=10)

    # --- Khung nhập liệu ---

    frm\_input = tk.LabelFrame(parent\_frame, text="Thông tin khách hàng",

                                  padx=10, pady=10, bg=bg\_color,

                                  font=("Arial", 11))

    frm\_input.pack(padx=10, pady=5, fill="x")

    # Entry ẩn để chứa ma\_kh khi chọn

    entry\_ma = tk.Entry(frm\_input)

    tk.Label(frm\_input, text="Họ tên:", bg=bg\_color, font=("Arial", 10)).grid(row=0, column=0, padx=5, pady=5, sticky="w")

    entry\_ten = tk.Entry(frm\_input, width=30, font=("Arial", 10))

    entry\_ten.grid(row=0, column=1, padx=5, pady=5)

    tk.Label(frm\_input, text="Số điện thoại:", bg=bg\_color, font=("Arial", 10)).grid(row=1, column=0, padx=5, pady=5, sticky="w")

    entry\_sdt = tk.Entry(frm\_input, width=30, font=("Arial", 10))

    entry\_sdt.grid(row=1, column=1, padx=5, pady=5)

    tk.Label(frm\_input, text="Địa chỉ:", bg=bg\_color, font=("Arial", 10)).grid(row=2, column=0, padx=5, pady=5, sticky="w")

    entry\_diachi = tk.Entry(frm\_input, width=60, font=("Arial", 10))

    entry\_diachi.grid(row=2, column=1, padx=5, pady=5, columnspan=3, sticky="we")

    frm\_input.grid\_columnconfigure(1, weight=1)

    # --- Khung nút ở dưới cùng ---

    frame\_btn\_bottom = tk.Frame(parent\_frame, bg=bg\_color)

    frame\_btn\_bottom.pack(side="bottom", fill="x", padx=10, pady=10)

    # --- Khung Treeview ---

    frame\_tree = tk.Frame(parent\_frame)

    frame\_tree.pack(padx=10, pady=10, fill="both", expand=True)

    # --- Treeview ---

    columns = ("ma\_kh", "ho\_ten", "sdt", "dia\_chi")

    tree = ttk.Treeview(frame\_tree, columns=columns, show="headings", height=10)

    # Định nghĩa cột

    tree.heading("ma\_kh", text="Mã KH")

    tree.heading("ho\_ten", text="Họ tên")

    tree.heading("sdt", text="Số điện thoại")

    tree.heading("dia\_chi", text="Địa chỉ")

    # Định nghĩa độ rộng cột

    tree.column("ma\_kh", width=80, anchor="center")

    tree.column("ho\_ten", width=180, anchor="center")

    tree.column("sdt", width=120, anchor="center")

    tree.column("dia\_chi", width=250, anchor="center")

    # Thêm thanh cuộn dọc

    scroll\_y = ttk.Scrollbar(frame\_tree, orient="vertical", command=tree.yview)

    tree.configure(yscrollcommand=scroll\_y.set)

    scroll\_y.pack(side="right", fill="y")

    tree.pack(side="left", fill="both", expand=True)

    # --- Các hàm xử lý ---

    def load\_data():

        # Tải dữ liệu từ CSDL vào Treeview

        tree.delete(\*tree.get\_children())

        try:

            conn = connect\_db()

            cur = conn.cursor()

            cur.execute("SELECT ma\_kh, ho\_ten, sdt, dia\_chi FROM khachhang ORDER BY ma\_kh ASC")

            for row in cur.fetchall():

                tree.insert("", tk.END, values=row)

            conn.close()

        except Exception as e:

            messagebox.showerror("Lỗi", f"Không thể tải dữ liệu: {e}")

    def \_clear\_entries():

        # Xóa nội dung các trường nhập liệu

        entry\_ma.delete(0, tk.END)

        entry\_ten.delete(0, tk.END)

        entry\_sdt.delete(0, tk.END)

        entry\_diachi.delete(0, tk.END)

    def clear\_form():

        # Xóa form và bỏ chọn trên Treeview

        \_clear\_entries()

        if tree.selection():

            tree.selection\_remove(tree.selection())

    def get\_form\_data():

        #   Lấy và xác thực dữ liệu từ form

        ho\_ten = entry\_ten.get().strip()

        sdt = entry\_sdt.get().strip()

        dia\_chi = entry\_diachi.get().strip()

        if not ho\_ten or not sdt or not dia\_chi:

            messagebox.showwarning("Thiếu dữ liệu", "Vui lòng nhập đầy đủ thông tin!")

            return None

        return (ho\_ten, sdt, dia\_chi)

    def them():

        #   Thêm khách hàng mới

        data = get\_form\_data()

        if data is None:

            return

        conn = connect\_db()

        cur = conn.cursor()

        try:

            cur.execute("""

                INSERT INTO khachhang (ho\_ten, sdt, dia\_chi)

                VALUES (%s, %s, %s)

            """, data)

            conn.commit()

            messagebox.showinfo("Thành công", "Đã thêm khách hàng mới!")

            clear\_form()

            load\_data()

        except Exception as e:

            conn.rollback()

            messagebox.showerror("Lỗi", f"Không thể thêm dữ liệu: {e}")

        finally:

            conn.close()

    def xoa():

        # Xóa khách hàng (với xử lý lỗi Khóa ngoại)

        sel = tree.selection()

        if not sel:

            messagebox.showwarning("Chưa chọn", "Chọn khách hàng để xóa!")

            return

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

        if not messagebox.askyesno("Xác nhận", f"Bạn có chắc muốn xóa khách hàng mã {ma}?"):

            return

        conn = connect\_db()

        cur = conn.cursor()

        try:

            cur.execute("DELETE FROM khachhang WHERE ma\_kh=%s", (ma,))

            conn.commit()

            messagebox.showinfo("Thành công", "Đã xóa khách hàng!")

        except Exception as e:

            conn.rollback()

            # --- BẮT LỖI KHÓA NGOẠI ---

            if e.args and e.args[0] == 1451:

                messagebox.showerror(

                    "Không thể xóa",

                    "Không thể xóa khách hàng này vì đã có lịch sử mua hàng (tồn tại trong hóa đơn).\n\n"

                    "Bạn chỉ có thể xóa khách hàng chưa từng mua hàng."

                )

            else:

                messagebox.showerror("Lỗi SQL", f"Không thể xóa: {e}")

        finally:

            conn.close()

        clear\_form()

        load\_data()

    def on\_tree\_select(event=None):

        # Xử lý khi chọn một hàng trên Treeview

        sel = tree.selection()

        if not sel:

            if event is None: # Chỉ cảnh báo khi bấm nút "Sửa"

                messagebox.showwarning("Chưa chọn", "Vui lòng chọn khách hàng trên bảng!")

            return

        \_clear\_entries()

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

        entry\_ma.insert(0, vals[0])

        entry\_ten.insert(0, vals[1])

        entry\_sdt.insert(0, vals[2])

        entry\_diachi.insert(0, vals[3])

    def luu():

        # Lưu thay đổi cho khách hàng đã chọn

        ma\_kh = entry\_ma.get().strip()

        if not ma\_kh:

            messagebox.showwarning("Thiếu mã", "Hãy nhấn 'Sửa' (hoặc nhấp vào bảng) để chọn khách hàng trước!")

            return

        data = get\_form\_data()

        if data is None:

            return

        full\_data = data + (ma\_kh,) # (ho\_ten, sdt, dia\_chi, ma\_kh)

        conn = connect\_db()

        cur = conn.cursor()

        try:

            cur.execute("""

                UPDATE khachhang SET ho\_ten=%s, sdt=%s, dia\_chi=%s

                WHERE ma\_kh=%s

            """, full\_data)

            conn.commit()

            messagebox.showinfo("Thành công", "Đã lưu thay đổi!")

            clear\_form()

            load\_data()

        except Exception as e:

            conn.rollback()

            messagebox.showerror("Lỗi", f"Không thể cập nhật dữ liệu: {e}")

        finally:

            conn.close()

    # Gán sự kiện click vào Treeview

    tree.bind("<<TreeviewSelect>>", on\_tree\_select)

    # --- Thêm nút vào frame\_btn\_bottom ---

    btn\_them = tk.Button(frame\_btn\_bottom, text="Thêm", command=them,

                         width=12, bg=btn\_add\_bg, fg=btn\_fg\_white,

                         font=("Arial", 11, "bold"), height=2)

    btn\_them.pack(side="left", padx=5)

    btn\_sua = tk.Button(frame\_btn\_bottom, text="Sửa (Chọn)", command=lambda: on\_tree\_select(None),

                         width=12, bg=btn\_edit\_bg, fg=btn\_fg\_white,

                         font=("Arial", 11, "bold"), height=2)

    btn\_sua.pack(side="left", padx=5)

    btn\_luu = tk.Button(frame\_btn\_bottom, text="Lưu", command=luu,

                         width=12, bg=btn\_save\_bg, fg=btn\_fg\_black,

                         font=("Arial", 11, "bold"), height=2)

    btn\_luu.pack(side="left", padx=5)

    btn\_xoa = tk.Button(frame\_btn\_bottom, text="Xóa", command=xoa,

                         width=12, bg=btn\_del\_bg, fg=btn\_fg\_white,

                         font=("Arial", 11, "bold"), height=2)

    btn\_xoa.pack(side="left", padx=5)

    btn\_clear = tk.Button(frame\_btn\_bottom, text="Huỷ", command=clear\_form,

                          width=12, bg=btn\_clear\_bg, fg=btn\_fg\_white,

                          font=("Arial", 11, "bold"), height=2)

    btn\_clear.pack(side="left", padx=5)

    load\_data()

TAB QUẢN LÍ TỒN KHO

import tkinter as tk

from tkinter import ttk, messagebox

from tkcalendar import DateEntry  # Cần cài đặt: pip install tkcalendar

from db import connect\_db

import datetime

# Biến toàn cục để lưu danh sách Tivi (ma\_tivi, ten\_tivi, hang\_sx)

tivi\_list = []

def create\_view(parent\_frame):

    global tivi\_list

    # --- Màu sắc ---

    bg\_color = "#f4f4f4" # Màu nền khung chính

    btn\_add\_bg = "#28a745" # Màu nút Thêm

    btn\_edit\_bg = "#007bff" # Màu nút Sửa

    btn\_save\_bg = "#ffc107" # Màu nút Lưu

    btn\_del\_bg = "#dc3545" # Màu nút Xóa

    btn\_clear\_bg = "#6c757d" # Màu nút Huỷ

    btn\_fg\_white = "white" # Màu chữ nút trắng

    btn\_fg\_black = "black" # Màu chữ nút đen

    # --- Tiêu đề ---

    tk.Label(parent\_frame, text="QUẢN LÝ TỒN KHO",

             font=("Arial", 18, "bold"),

             bg=bg\_color, fg="#333").pack(pady=10)

    # --- Khung lọc ---

    frm\_filter = tk.Frame(parent\_frame, bg=bg\_color)

    frm\_filter.pack(padx=10, pady=5, fill="x")

    # --- Lọc theo hãng ---

    tk.Label(frm\_filter, text="Lọc theo hãng:", bg=bg\_color, font=("Arial", 10, "bold")).pack(side="left", padx=(0, 5))

    combo\_filter = ttk.Combobox(

        frm\_filter, state="readonly", font=("Arial", 10),

        values=["Tất cả", "Samsung", "LG", "Sony", "TCL", "Panasonic", "Sharp", "Asanzo"],

        width=20

    )

    combo\_filter.current(0)

    combo\_filter.pack(side="left", padx=5)

    # --- Khung nhập liệu ---

    frm\_input = tk.LabelFrame(parent\_frame, text="Thông tin cập nhật tồn kho",

                                  padx=10, pady=10, bg=bg\_color,

                                  font=("Arial", 11))

    frm\_input.pack(padx=10, pady=5, fill="x")

    tk.Label(frm\_input, text="Chọn Tivi:", bg=bg\_color, font=("Arial", 10)).grid(row=0, column=0, sticky="w", padx=5, pady=5)

    combo\_tivi = ttk.Combobox(frm\_input, state="readonly", width=40, font=("Arial", 10))

    combo\_tivi.grid(row=0, column=1, padx=5, pady=5, sticky="w")

    tk.Label(frm\_input, text="Ngày cập nhật:", bg=bg\_color, font=("Arial", 10)).grid(row=0, column=2, sticky="w", padx=5, pady=5)

    entry\_ngay = DateEntry(frm\_input, width=23, font=("Arial", 10), background="darkblue", foreground="white", date\_pattern="yyyy-mm-dd")

    entry\_ngay.grid(row=0, column=3, padx=5, pady=5)

    tk.Label(frm\_input, text="Số lượng:", bg=bg\_color, font=("Arial", 10)).grid(row=2, column=0, sticky="w", padx=5, pady=5)

    entry\_soluong = tk.Entry(frm\_input, width=42, font=("Arial", 10))

    entry\_soluong.grid(row=2, column=1, padx=5, pady=5, sticky="w")

    tk.Label(frm\_input, text="Ghi chú:", bg=bg\_color, font=("Arial", 10)).grid(row=3, column=0, sticky="w", padx=5, pady=5)

    entry\_note = tk.Entry(frm\_input, font=("Arial", 10))

    entry\_note.grid(row=3, column=1, columnspan=3, padx=5, pady=5, sticky="we")

    frm\_input.grid\_columnconfigure(1, weight=1)

    frm\_input.grid\_columnconfigure(3, weight=1)

    # --- Khung nút ở dưới cùng ---

    frame\_btn\_bottom = tk.Frame(parent\_frame, bg=bg\_color)

    frame\_btn\_bottom.pack(side="bottom", fill="x", padx=10, pady=10)

    # --- Khung Treeview ---

    frame\_tree = tk.Frame(parent\_frame)

    frame\_tree.pack(padx=10, pady=10, fill="both", expand=True)

    # --- Cập nhật cột Treeview ---

    columns = ( "ma\_ton", "ma\_tivi", "ten\_tivi", "hang\_sx", "ngay\_cap\_nhat", "so\_luong\_ton", "ghi\_chu")

    tree = ttk.Treeview(frame\_tree, columns=columns, show="headings", height=12)

    # Định nghĩa cột

    tree.heading("ma\_ton", text="ID Log")

    tree.heading("ma\_tivi", text="Mã TV")

    tree.heading("ten\_tivi", text="Tên Tivi")

    tree.heading("hang\_sx", text="Hãng SX")

    tree.heading("ngay\_cap\_nhat", text="Ngày Cập Nhật")

    tree.heading("so\_luong\_ton", text="Số Lượng")

    tree.heading("ghi\_chu", text="Ghi Chú")

    #   Định nghĩa độ rộng cột

    tree.column("ma\_ton", width=50, anchor="center")

    tree.column("ma\_tivi", width=50, anchor="center")

    tree.column("ten\_tivi", width=200, anchor="center")

    tree.column("hang\_sx", width=100, anchor="center")

    tree.column("ngay\_cap\_nhat", width=100, anchor="center")

    tree.column("so\_luong\_ton", width=80, anchor="center")

    tree.column("ghi\_chu", width=250, anchor="center")

    # Ẩn 2 cột ID

    tree.column("ma\_ton", width=0, stretch=tk.NO)

    tree.column("ma\_tivi", width=0, stretch=tk.NO)

    # Thêm thanh cuộn dọc

    scroll\_y = ttk.Scrollbar(frame\_tree, orient="vertical", command=tree.yview)

    tree.configure(yscrollcommand=scroll\_y.set)

    scroll\_y.pack(side="right", fill="y")

    tree.pack(side="left", fill="both", expand=True)

    # --- Các hàm xử lý  ---

    def load\_tivi\_combobox():

        # Tải danh sách Tivi từ CSDL vào Combobox

        global tivi\_list

        try:

            conn = connect\_db()

            cur = conn.cursor()

            cur.execute("SELECT ma\_tivi, ten\_tivi, hang\_sx FROM tivi ORDER BY ten\_tivi")

            tivi\_list = cur.fetchall()

            combo\_tivi['values'] = [f"{row[1]} (Hãng: {row[2]}, ID: {row[0]})" for row in tivi\_list]

            if tivi\_list:

                combo\_tivi.current(0)

        except Exception as e:

            messagebox.showerror("Lỗi", f"Không thể tải danh sách Tivi: {e}")

        finally:

            if conn:

                conn.close()

    def load\_data(event=None):

        # Tải dữ liệu từ CSDL vào Treeview

        tree.delete(\*tree.get\_children())

        try:

            conn = connect\_db()

            cur = conn.cursor()

            hang = combo\_filter.get()

            base\_query = """

                SELECT tk.ma\_ton, tk.ma\_tivi, tv.ten\_tivi, tv.hang\_sx,

                       tk.ngay\_cap\_nhat, tk.so\_luong\_ton, tk.ghi\_chu

                FROM tonkho tk

                JOIN tivi tv ON tk.ma\_tivi = tv.ma\_tivi

            """

            if hang == "Tất cả":

                cur.execute(base\_query + " ORDER BY tk.ngay\_cap\_nhat DESC, tk.ma\_ton DESC")

            else:

                cur.execute(base\_query + " WHERE tv.hang\_sx=%s ORDER BY tk.ngay\_cap\_nhat DESC, tk.ma\_ton DESC", (hang,))

            for row in cur.fetchall():

                so\_luong\_formatted = f"+{row[5]}" if row[5] > 0 else str(row[5])

                formatted\_row = (row[0], row[1], row[2], row[3], row[4], so\_luong\_formatted, row[6])

                tree.insert("", tk.END, values=formatted\_row)

        except Exception as e:

            messagebox.showerror("Lỗi", f"Không thể tải dữ liệu: {e}")

        finally:

            if conn:

                conn.close()

    def \_clear\_entries():

        # Xóa nội dung các trường nhập liệu

        if tivi\_list:

            combo\_tivi.current(0)

        entry\_soluong.delete(0, tk.END)

        entry\_note.delete(0, tk.END)

        entry\_ngay.set\_date(datetime.date.today())

    def clear\_form():

        # Xóa form và bỏ chọn trên Treeview

        \_clear\_entries()

        if tree.selection():

            tree.selection\_remove(tree.selection())

    def get\_form\_data():

        #   Lấy và xác thực dữ liệu từ form

        try:

            selected\_index = combo\_tivi.current()

            if selected\_index < 0:

                messagebox.showwarning("Thiếu dữ liệu", "Vui lòng chọn Tivi")

                return None

            ma\_tivi = tivi\_list[selected\_index][0]

        except IndexError:

            messagebox.showwarning("Thiếu dữ liệu", "Vui lòng chọn Tivi")

            return None

        ngay = entry\_ngay.get()

        so\_luong\_str = entry\_soluong.get().strip()

        note = entry\_note.get().strip()

        if not so\_luong\_str:

            messagebox.showwarning("Thiếu dữ liệu", "Vui lòng nhập Số lượng")

            return None

        try:

            so\_luong = int(so\_luong\_str)

            if so\_luong == 0:

                raise ValueError("Số lượng phải khác 0")

        except ValueError as e:

            messagebox.showerror("Lỗi dữ liệu", f"Số lượng phải là số nguyên khác 0!\n{e}")

            return None

        return (ma\_tivi, ngay, so\_luong, note)

    def them():

        #   Thêm bản ghi tồn kho mới

        data = get\_form\_data()

        if data is None:

            return

        # data = (ma\_tivi, ngay, so\_luong, note)

        conn = connect\_db()

        cur = conn.cursor()

        try:

            cur.execute("INSERT INTO tonkho (ma\_tivi, ngay\_cap\_nhat, so\_luong\_ton, ghi\_chu) VALUES (%s,%s,%s,%s)",

                        (data[0], data[1], data[2], data[3]))

            # --- Tự động cập nhật bảng tivi ---

            cur.execute("UPDATE tivi SET so\_luong = so\_luong + %s WHERE ma\_tivi = %s", (data[2], data[0]))

            conn.commit()

            messagebox.showinfo("Thành công", f"Đã thêm bản ghi tồn kho VÀ cập nhật số lượng Tivi ID: {data[0]}!")

        except Exception as e:

            conn.rollback()

            messagebox.showerror("Lỗi", str(e))

        finally:

            conn.close()

        load\_data()

        clear\_form()

    def xoa():

        #   Xóa bản ghi tồn kho (và hoàn tác số lượng trong kho Tivi)

        sel = tree.selection()

        if not sel:

            messagebox.showwarning("Chưa chọn", "Chọn bản ghi cần xóa!")

            return

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

        ma\_ton = values[0]

        ma\_tivi = values[1]

        try:

            so\_luong\_log = int(str(values[5]).replace("+", ""))

        except:

            so\_luong\_log = 0

        if not messagebox.askyesno("Xác nhận", f"Bạn có chắc muốn xóa bản ghi log {ma\_ton}?\n\nCẢNH BÁO: Thao tác này sẽ HOÀN TÁC thay đổi tồn kho (trừ lại {so\_luong\_log} cái cho Tivi ID {ma\_tivi})."):

            return

        conn = connect\_db()

        cur = conn.cursor()

        try:

            cur.execute("DELETE FROM tonkho WHERE ma\_ton=%s", (ma\_ton,))

            cur.execute("UPDATE tivi SET so\_luong = so\_luong - %s WHERE ma\_tivi = %s", (so\_luong\_log, ma\_tivi))

            conn.commit()

            messagebox.showinfo("Thành công", "Đã xóa bản ghi log VÀ hoàn tác số lượng trong kho Tivi!")

        except Exception as e:

            conn.rollback()

            messagebox.showerror("Lỗi", f"Không thể xóa: {e}")

        finally:

            conn.close()

        clear\_form()

        load\_data()

    def on\_tree\_select(event=None):

        # Xử lý khi chọn một hàng trên Treeview

        sel = tree.selection()

        if not sel:

            if event is None:

                messagebox.showwarning("Chưa chọn", "Vui lòng chọn bản ghi trên bảng!")

            return

        \_clear\_entries()

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

        ma\_tivi\_can\_chon = vals[1]

        for i, tivi in enumerate(tivi\_list):

            if tivi[0] == ma\_tivi\_can\_chon:

                combo\_tivi.current(i)

                break

        try:

            entry\_ngay.set\_date(vals[4])

        except:

            entry\_ngay.set\_date(None)

        so\_luong\_str = str(vals[5]).replace("+", "")

        entry\_soluong.insert(0, so\_luong\_str)

        entry\_note.insert(0, vals[6])

    def luu():

        #   Lưu thay đổi cho khách hàng đã chọn

        sel = tree.selection()

        if not sel:

            messagebox.showwarning("Chưa chọn", "Hãy nhấn 'Sửa' để chọn bản ghi cần lưu!")

            return

        old\_vals = tree.item(sel[0])["values"]

        ma\_ton = old\_vals[0]

        old\_ma\_tivi = old\_vals[1]

        try:

            old\_so\_luong = int(str(old\_vals[5]).replace("+", ""))

        except:

            old\_so\_luong = 0

        new\_data = get\_form\_data()

        if new\_data is None:

            return

        # new\_data = (new\_ma\_tivi, new\_ngay, new\_so\_luong, new\_note)

        new\_ma\_tivi = new\_data[0]

        new\_ngay = new\_data[1]

        new\_so\_luong = new\_data[2]

        new\_note = new\_data[3]

        if not messagebox.askyesno("Xác nhận Cực kỳ Cẩn trọng!",

            "Bạn đang sửa một bản ghi LOG. Việc này sẽ:\n"

            "1. Hoàn tác số lượng CŨ (trừ {-old\_so\_luong} cho Tivi ID {old\_ma\_tivi})\n"

            "2. Cập nhật số lượng MỚI (cộng {new\_so\_luong} cho Tivi ID {new\_ma\_tivi})\n"

            "Bạn có chắc muốn tiếp tục?"):

            return

        conn = connect\_db()

        cur = conn.cursor()

        try:

            #  Hoàn tác số lượng CŨ

            cur.execute("UPDATE tivi SET so\_luong = so\_luong - %s WHERE ma\_tivi = %s", (old\_so\_luong, old\_ma\_tivi))

            #  Cập nhật bản ghi log

            cur.execute("""UPDATE tonkho

                           SET ma\_tivi=%s, ngay\_cap\_nhat=%s, so\_luong\_ton=%s, ghi\_chu=%s

                           WHERE ma\_ton=%s""",

                        (new\_ma\_tivi, new\_ngay, new\_so\_luong, new\_note, ma\_ton))

            #  Cập nhật số lượng MỚI

            cur.execute("UPDATE tivi SET so\_luong = so\_luong + %s WHERE ma\_tivi = %s", (new\_so\_luong, new\_ma\_tivi))

            conn.commit()

            messagebox.showinfo("Thành công", "Đã lưu thay đổi VÀ cập nhật cả hai kho Tivi!")

        except Exception as e:

            conn.rollback()

            messagebox.showerror("Lỗi", str(e))

        finally:

            conn.close()

        load\_data()

        clear\_form()

    # Kết nối sự kiện

    tree.bind("<<TreeviewSelect>>", on\_tree\_select)

    combo\_filter.bind("<<ComboboxSelected>>", load\_data)

    #--- Nút chức năng ở dưới cùng ---

    btn\_them = tk.Button(frame\_btn\_bottom, text="Thêm (Nhập/Xuất)", command=them,

                         width=15, bg=btn\_add\_bg, fg=btn\_fg\_white,

                         font=("Arial", 11, "bold"), height=2)

    btn\_them.pack(side="left", padx=5)

    btn\_sua = tk.Button(frame\_btn\_bottom, text="Sửa (Chọn)", command=lambda: on\_tree\_select(None),

                         width=12, bg=btn\_edit\_bg, fg=btn\_fg\_white,

                         font=("Arial", 11, "bold"), height=2)

    btn\_sua.pack(side="left", padx=5)

    btn\_luu = tk.Button(frame\_btn\_bottom, text="Lưu", command=luu,

                         width=12, bg=btn\_save\_bg, fg=btn\_fg\_black,

                         font=("Arial", 11, "bold"), height=2)

    btn\_luu.pack(side="left", padx=5)

    btn\_xoa = tk.Button(frame\_btn\_bottom, text="Xóa (Hoàn tác)", command=xoa,

                         width=15, bg=btn\_del\_bg, fg=btn\_fg\_white,

                         font=("Arial", 11, "bold"), height=2)

    btn\_xoa.pack(side="left", padx=5)

    btn\_clear = tk.Button(frame\_btn\_bottom, text="Huỷ", command=clear\_form,

                          width=12, bg=btn\_clear\_bg, fg=btn\_fg\_white,

                          font=("Arial", 11, "bold"), height=2)

    btn\_clear.pack(side="left", padx=5)

    # Tải dữ liệu ban đầu

    load\_tivi\_combobox() # Tải danh sách Tivi vào Combobox

    load\_data()          # Tải Log kho

    clear\_form()

TAB QUẢN LÍ HÓA ĐƠN

import tkinter as tk

from tkinter import ttk, messagebox

from db import connect\_db

def create\_view(parent\_frame): # Tạo giao diện quản lý hóa đơn

    frame = tk.Frame(parent\_frame, bg="#f4f4f4") # Màu nền khung chính

    frame.pack(fill="both", expand=True) # Đặt khung chính để chiếm toàn bộ không gian

    tk.Label(frame, text="QUẢN LÝ HÓA ĐƠN", font=("Arial", 20, "bold"), bg="#f4f4f4").pack(pady=20) # Tiêu đề

    # --- Frame chứa 2 TreeView ---

    main\_view\_frame = tk.Frame(frame, bg="#f4f4f4")

    main\_view\_frame.pack(fill="both", expand=True, padx=20, pady=10)

    # ---  Danh sách Hóa đơn  ---

    hoadon\_frame = tk.LabelFrame(main\_view\_frame, text="Danh sách Hóa đơn", padx=10, pady=10)

    hoadon\_frame.pack(side="left", fill="both", expand=True, padx=(0, 10))

    #-- TreeView Hóa đơn ---

    hoadon\_cols = ('ma\_hd', 'ngay\_lap', 'nhan\_vien', 'khach\_hang', 'tong\_tien')

    hoadon\_tree = ttk.Treeview(hoadon\_frame, columns=hoadon\_cols, show='headings', height=20)

    # Định nghĩa cột

    hoadon\_tree.heading('ma\_hd', text='Mã HĐ')

    hoadon\_tree.heading('ngay\_lap', text='Ngày lập')

    hoadon\_tree.heading('nhan\_vien', text='Nhân viên')

    hoadon\_tree.heading('khach\_hang', text='Khách hàng')

    hoadon\_tree.heading('tong\_tien', text='Tổng tiền')

    # Định nghĩa kích thước cột

    hoadon\_tree.column('ma\_hd', width=60, anchor="center")

    hoadon\_tree.column('ngay\_lap', width=100,anchor="center")

    hoadon\_tree.column('nhan\_vien', width=150,anchor="center")

    hoadon\_tree.column('khach\_hang', width=150,anchor="center")

    hoadon\_tree.column('tong\_tien', width=100, anchor="center")

    # Đặt TreeView vào frame

    hoadon\_tree.pack(fill="both", expand=True)

    # ---  Chi tiết Hóa đơn  ---

    chitiet\_frame = tk.LabelFrame(main\_view\_frame, text="Chi tiết Hóa đơn", padx=10, pady=10)

    chitiet\_frame.pack(side="right", fill="both", expand=True, padx=(10, 0))

    #-- TreeView Chi tiết Hóa đơn ---

    chitiet\_cols = ('ten\_tivi', 'so\_luong', 'don\_gia')

    chitiet\_tree = ttk.Treeview(chitiet\_frame, columns=chitiet\_cols, show='headings', height=20)

    # Định nghĩa cột

    chitiet\_tree.heading('ten\_tivi', text='Tên Tivi')

    chitiet\_tree.heading('so\_luong', text='Số lượng')

    chitiet\_tree.heading('don\_gia', text='Đơn giá')

    # Định nghĩa kích thước cột

    chitiet\_tree.column('ten\_tivi', width=200,anchor="center")

    chitiet\_tree.column('so\_luong', width=80, anchor="center")

    chitiet\_tree.column('don\_gia', width=100, anchor="center")

    # Đặt TreeView vào frame

    chitiet\_tree.pack(fill="both", expand=True)

    # --- Các hàm xử lý ---

    def load\_hoadon\_data():

        # Tải dữ liệu hóa đơn từ CSDL vào TreeView

        for item in hoadon\_tree.get\_children():

            hoadon\_tree.delete(item)

        try:

            conn = connect\_db()

            cursor = conn.cursor()

            query = """

                SELECT hd.ma\_hd, hd.ngay\_lap, nv.ho\_ten, kh.ho\_ten, hd.tong\_tien

                FROM hoadon hd

                JOIN nhanvien nv ON hd.ma\_nv = nv.ma\_nv

                JOIN khachhang kh ON hd.ma\_kh = kh.ma\_kh

                ORDER BY hd.ngay\_lap DESC, hd.ma\_hd DESC

            """

            cursor.execute(query)

            rows = cursor.fetchall()

            for row in rows:

                # Format lại tiền tệ và ngày tháng

                formatted\_row = (row[0], row[1], row[2], row[3], f"{row[4]:,.0f} VNĐ")

                hoadon\_tree.insert('', 'end', values=formatted\_row)

        except Exception as e:

            messagebox.showerror("Lỗi CSDL", f"Không thể tải danh sách hóa đơn: {e}")

        finally:

            if conn:

                conn.close()

    def show\_chitiet\_data(event):

        # Hiển thị chi tiết hóa đơn khi chọn một hóa đơn

        for item in chitiet\_tree.get\_children():

            chitiet\_tree.delete(item)

        selected\_item = hoadon\_tree.focus()

        if not selected\_item:

            return

        values = hoadon\_tree.item(selected\_item, 'values')

        ma\_hd = values[0] # Lấy mã hóa đơn từ cột đầu tiên

        try:

            conn = connect\_db()

            cursor = conn.cursor()

            # Dùng JOIN để lấy tên Tivi

            query = """

                SELECT tv.ten\_tivi, cthd.so\_luong\_mua, cthd.don\_gia

                FROM chitiet\_hoadon cthd

                JOIN tivi tv ON cthd.ma\_tivi = tv.ma\_tivi

                WHERE cthd.ma\_hd = %s

            """

            cursor.execute(query, (ma\_hd,))

            rows = cursor.fetchall()

            for row in rows:

                formatted\_row = (row[0], row[1], f"{row[2]:,.0f} VNĐ")

                chitiet\_tree.insert('', 'end', values=formatted\_row)

        except Exception as e:

            messagebox.showerror("Lỗi CSDL", f"Không thể tải chi tiết hóa đơn: {e}")

        finally:

            if conn:

                conn.close()

    # Gán sự kiện click (chọn) vào TreeView hóa đơn

    hoadon\_tree.bind('<<TreeviewSelect>>', show\_chitiet\_data)

    # Nút Tải lại dữ liệu

    refresh\_btn = tk.Button(frame, text="Tải lại danh sách", command=load\_hoadon\_data)

    refresh\_btn.pack(pady=10)

    # Tải dữ liệu lần đầu khi mở tab

    load\_hoadon\_data()

TAB BÁN HÀNG

import tkinter as tk

from tkinter import ttk, messagebox

from db import connect\_db

from datetime import date

current\_cart = [] # (ma\_tivi, ten\_tivi, so\_luong, don\_gia, thanh\_tien) # Giỏ hàng hiện tại

tivi\_list = [] # (ma\_tivi, ten\_tivi, don\_gia, so\_luong\_ton) # Danh sách Tivi còn hàng

customer\_list = [] # (ma\_kh, ho\_ten) # Danh sách Khách hàng

staff\_list = [] # (ma\_nv, ho\_ten) # Danh sách Nhân viên

def create\_view(parent\_frame):

    global current\_cart, tivi\_list, customer\_list, staff\_list

    # Reset giỏ hàng mỗi khi vào tab

    current\_cart = []

    # Tạo khung chính

    frame = tk.Frame(parent\_frame, bg="#f4f4f4")

    frame.pack(fill="both", expand=True)

    #--- Tiêu đề ---

    tk.Label(frame, text="BÁN HÀNG -  HÓA ĐƠN", font=("Arial", 20, "bold"), bg="#f4f4f4").pack(pady=20)

    # --- Khung chính chứa 2 cột ---

    main\_container = tk.Frame(frame, bg="#f4f4f4")

    main\_container.pack(fill="both", expand=True, padx=20, pady=10)

    # --- CỘT TRÁI: Thông tin chọn Khách hàng, Nhân viên, Tivi ---

    info\_frame = tk.Frame(main\_container, bg="#f4f4f4")

    info\_frame.pack(side="left", fill="y", padx=(0, 10))

    #  Chọn Khách hàng và Nhân viên

    customer\_frame = tk.LabelFrame(info\_frame, text="Thông tin Hóa đơn", padx=10, pady=10)

    customer\_frame.pack(fill="x", pady=5)

    # -- Combobox Khách hàng và Nhân viên ---

    tk.Label(customer\_frame, text="Chọn Khách hàng:").grid(row=0, column=0, sticky="w", padx=5, pady=5)

    customer\_combo = ttk.Combobox(customer\_frame, width=30, state="readonly")

    customer\_combo.grid(row=0, column=1, sticky="w", padx=5, pady=5)

    tk.Label(customer\_frame, text="Chọn Nhân viên:").grid(row=1, column=0, sticky="w", padx=5, pady=5)

    staff\_combo = ttk.Combobox(customer\_frame, width=30, state="readonly")

    staff\_combo.grid(row=1, column=1, sticky="w", padx=5, pady=5)

    #  Chọn Tivi và Số lượng

    product\_frame = tk.LabelFrame(info\_frame, text="Thêm Tivi vào giỏ", padx=10, pady=10)

    product\_frame.pack(fill="x", pady=10)

    #-- Combobox Tivi và nhập số lượng ---

    tk.Label(product\_frame, text="Chọn Tivi:").grid(row=0, column=0, sticky="w", padx=5, pady=5)

    tivi\_combo = ttk.Combobox(product\_frame, width=50, state="readonly")

    tivi\_combo.grid(row=0, column=1, columnspan=3, sticky="w", padx=5, pady=5)

    #-- Nhãn hiển thị số lượng tồn và đơn giá ---

    tk.Label(product\_frame, text="Số lượng tồn:").grid(row=1, column=0, sticky="w", padx=5, pady=5)

    ton\_kho\_label = tk.Label(product\_frame, text="...", fg="blue")

    ton\_kho\_label.grid(row=1, column=1, sticky="w", padx=5, pady=5)

    tk.Label(product\_frame, text="Đơn giá:").grid(row=1, column=2, sticky="w", padx=5, pady=5)

    don\_gia\_label = tk.Label(product\_frame, text="...", fg="blue")

    don\_gia\_label.grid(row=1, column=3, sticky="w", padx=5, pady=5)

    #-- Nhập số lượng mua ---

    tk.Label(product\_frame, text="Nhập Số lượng mua:").grid(row=2, column=0, sticky="w", padx=5, pady=5)

    so\_luong\_entry = tk.Entry(product\_frame, width=10)

    so\_luong\_entry.grid(row=2, column=1, sticky="w", padx=5, pady=5)

    # -- Nút Thêm vào giỏ ---

    add\_to\_cart\_btn = tk.Button(product\_frame, text="Thêm vào giỏ", command=lambda: add\_to\_cart(tivi\_combo, so\_luong\_entry, ton\_kho\_label, don\_gia\_label))

    add\_to\_cart\_btn.grid(row=2, column=2, columnspan=2, sticky="ew", padx=5, pady=10)

    # --- CỘT PHẢI: Giỏ hàng và Thanh toán ---

    cart\_frame\_container = tk.Frame(main\_container, bg="#f4f4f4")

    cart\_frame\_container.pack(side="right", fill="both", expand=True, padx=(10, 0))

    # Giỏ hàng

    cart\_frame = tk.LabelFrame(cart\_frame\_container, text="Giỏ hàng hiện tại", padx=10, pady=10)

    cart\_frame.pack(fill="both", expand=True)

    #-- TreeView Giỏ hàng ---

    cart\_cols = ('ma\_tivi', 'ten\_tivi', 'so\_luong', 'don\_gia', 'thanh\_tien')

    cart\_tree = ttk.Treeview(cart\_frame, columns=cart\_cols, show='headings', height=10)

    # Định nghĩa cột

    cart\_tree.heading('ma\_tivi', text='Mã TV')

    cart\_tree.heading('ten\_tivi', text='Tên Tivi')

    cart\_tree.heading('so\_luong', text='SL')

    cart\_tree.heading('don\_gia', text='Đơn giá')

    cart\_tree.heading('thanh\_tien', text='Thành tiền')

    # Định nghĩa kích thước cột

    cart\_tree.column('ma\_tivi', width=50, anchor="center")

    cart\_tree.column('ten\_tivi', width=200,anchor="center")

    cart\_tree.column('so\_luong', width=50, anchor="center")

    cart\_tree.column('don\_gia', width=100, anchor="center")

    cart\_tree.column('thanh\_tien', width=100, anchor="center")

    # Đặt TreeView vào frame

    cart\_tree.pack(fill="both", expand=True, side="top")

    #  Nút Xóa khỏi giỏ

    remove\_item\_btn = tk.Button(cart\_frame\_container, text="Xóa Tivi khỏi giỏ", command=lambda: remove\_from\_cart(cart\_tree))

    remove\_item\_btn.pack(pady=5, side="top")

    # Thanh toán

    total\_frame = tk.Frame(cart\_frame\_container, bg="#f4f4f4")

    total\_frame.pack(fill="x", side="bottom", pady=10)

    #-- Nhãn Tổng tiền và Nút Thanh toán ---

    tk.Label(total\_frame, text="TỔNG TIỀN:", font=("Arial", 14, "bold"), bg="#f4f4f4").grid(row=0, column=0, sticky="w", padx=10)

    total\_label = tk.Label(total\_frame, text="0 VNĐ", font=("Arial", 14, "bold"), fg="red", bg="#f4f4f4")

    total\_label.grid(row=0, column=1, sticky="e", padx=10)

    checkout\_btn = tk.Button(total\_frame, text="THANH TOÁN (LƯU HÓA ĐƠN)", font=("Arial", 12, "bold"), bg="#28a745", fg="white",

                             command=lambda: checkout(customer\_combo, staff\_combo, cart\_tree, total\_label))

    checkout\_btn.grid(row=1, column=0, columnspan=2, sticky="ew", padx=10, pady=10, ipady=10)

    # --- CÁC HÀM XỬ LÝ ---

    def load\_combobox\_data():

        # Load dữ liệu cho các Combobox

        global tivi\_list, customer\_list, staff\_list

        try:

            conn = connect\_db()

            cursor = conn.cursor()

            # Load Khach hàng

            cursor.execute("SELECT ma\_kh, ho\_ten FROM khachhang ORDER BY ho\_ten")

            customer\_list = cursor.fetchall()

            customer\_combo['values'] = [f"{kh[1]} (ID: {kh[0]})" for kh in customer\_list]

            # Load Nhân viên

            cursor.execute("SELECT ma\_nv, ho\_ten FROM nhanvien ORDER BY ho\_ten")

            staff\_list = cursor.fetchall()

            staff\_combo['values'] = [f"{nv[1]} (ID: {nv[0]})" for nv in staff\_list]

            # Load Tivi (chỉ Tivi còn hàng)

            cursor.execute("SELECT ma\_tivi, ten\_tivi, gia, so\_luong FROM tivi WHERE so\_luong > 0 ORDER BY ten\_tivi")

            tivi\_list = cursor.fetchall()

            tivi\_combo['values'] = [f"{tv[1]} (Giá: {tv[2]:,.0f})" for tv in tivi\_list]

            if customer\_combo['values']: customer\_combo.current(0)

            if staff\_combo['values']: staff\_combo.current(0)

            if tivi\_combo['values']: tivi\_combo.current(0)

        except Exception as e:

            messagebox.showerror("Lỗi CSDL", f"Không thể tải dữ liệu: {e}")

        finally:

            if conn: conn.close()

    def on\_tivi\_select(event):

        # Cập nhật nhãn tồn kho và đơn giá khi chọn Tivi

        selected\_index = tivi\_combo.current()

        if selected\_index < 0: return

        selected\_tivi = tivi\_list[selected\_index]

        so\_luong\_ton = selected\_tivi[3]

        don\_gia = selected\_tivi[2]

        ton\_kho\_label.config(text=f"{so\_luong\_ton} cái")

        don\_gia\_label.config(text=f"{don\_gia:,.0f} VNĐ")

    def update\_cart\_tree(cart\_tree, total\_label):

        # Cập nhật TreeView giỏ hàng và tổng tiền

        # Xóa tất cả item cũ

        for item in cart\_tree.get\_children():

            cart\_tree.delete(item)

        tong\_tien = 0

        for item in current\_cart:

            # (ma\_tivi, ten\_tivi, so\_luong, don\_gia, thanh\_tien)

            formatted\_item = (item[0], item[1], item[2], f"{item[3]:,.0f}", f"{item[4]:,.0f}")

            cart\_tree.insert('', 'end', values=formatted\_item)

            tong\_tien += item[4]

        total\_label.config(text=f"{tong\_tien:,.0f} VNĐ")

    def add\_to\_cart(tivi\_combo, so\_luong\_entry, ton\_kho\_label, don\_gia\_label):

        # Thêm Tivi đã chọn vào giỏ hàng

        selected\_index = tivi\_combo.current()

        if selected\_index < 0:

            messagebox.showwarning("Chưa chọn", "Vui lòng chọn một Tivi")

            return

        try:

            so\_luong\_mua = int(so\_luong\_entry.get())

            if so\_luong\_mua <= 0:

                raise ValueError("Số lượng phải lớn hơn 0")

        except ValueError:

            messagebox.showwarning("Sai số lượng", "Vui lòng nhập số lượng mua hợp lệ")

            return

        selected\_tivi = tivi\_list[selected\_index]

        ma\_tivi = selected\_tivi[0]

        ten\_tivi = selected\_tivi[1]

        don\_gia = selected\_tivi[2]

        so\_luong\_ton = selected\_tivi[3]

        if so\_luong\_mua > so\_luong\_ton:

            messagebox.showwarning("Hết hàng", f"Số lượng tồn kho không đủ (Chỉ còn {so\_luong\_ton} cái)")

            return

        # Kiểm tra xem Tivi đã có trong giỏ chưa

        found = False

        for i, item in enumerate(current\_cart):

            if item[0] == ma\_tivi:

                # Cập nhật số lượng

                new\_sl = item[2] + so\_luong\_mua

                if new\_sl > so\_luong\_ton:

                    messagebox.showwarning("Hết hàng", f"Tổng số lượng mua ({new\_sl}) vượt quá tồn kho ({so\_luong\_ton})")

                    return

                thanh\_tien = new\_sl \* don\_gia

                current\_cart[i] = (ma\_tivi, ten\_tivi, new\_sl, don\_gia, thanh\_tien)

                found = True

                break

        if not found:

            # Thêm mới vào giỏ

            thanh\_tien = so\_luong\_mua \* don\_gia

            current\_cart.append((ma\_tivi, ten\_tivi, so\_luong\_mua, don\_gia, thanh\_tien))

        update\_cart\_tree(cart\_tree, total\_label)

        # Reset

        so\_luong\_entry.delete(0, 'end')

    def remove\_from\_cart(cart\_tree):

        # Xóa Tivi đã chọn khỏi giỏ hàng

        selected\_item = cart\_tree.focus()

        if not selected\_item:

            messagebox.showwarning("Chưa chọn", "Vui lòng chọn Tivi cần xóa khỏi giỏ")

            return

        values = cart\_tree.item(selected\_item, 'values')

        ma\_tivi = values[0]

        for i, item in enumerate(current\_cart):

            if item[0] == ma\_tivi:

                del current\_cart[i]

                break

        update\_cart\_tree(cart\_tree, total\_label)

    def checkout(customer\_combo, staff\_combo, cart\_tree, total\_label):

        # Thanh toán và lưu hóa đơn

        if not current\_cart:

            messagebox.showwarning("Giỏ hàng rỗng", "Vui lòng thêm sản phẩm vào giỏ")

            return

        try:

            # Lấy ma\_kh và ma\_nv từ combobox

            ma\_kh = customer\_list[customer\_combo.current()][0]

            ma\_nv = staff\_list[staff\_combo.current()][0]

        except IndexError:

            messagebox.showwarning("Thiếu thông tin", "Vui lòng chọn khách hàng và nhân viên")

            return

        if not messagebox.askyesno("Xác nhận", "Bạn có chắc muốn thanh toán và lưu hóa đơn này?"):

            return

        conn = None

        try:

            conn = connect\_db()

            cursor = conn.cursor()

            # --- Bắt đầu Transaction ---

            conn.start\_transaction()

            #  Tính tổng tiền từ giỏ hàng

            tong\_tien = sum(item[4] for item in current\_cart)

            #  Tạo Hóa đơn (hoadon)

            ngay\_lap = date.today()

            sql\_hoadon = "INSERT INTO hoadon (ma\_nv, ma\_kh, ngay\_lap, tong\_tien) VALUES (%s, %s, %s, %s)"

            cursor.execute(sql\_hoadon, (ma\_nv, ma\_kh, ngay\_lap, tong\_tien))

            # Lấy ID của hóa đơn vừa tạo

            ma\_hd\_moi = cursor.lastrowid

            #  Thêm các Chi tiết Hóa đơn (chitiet\_hoadon)

            sql\_chitiet = "INSERT INTO chitiet\_hoadon (ma\_hd, ma\_tivi, so\_luong\_mua, don\_gia) VALUES (%s, %s, %s, %s)"

            sql\_update\_tivi = "UPDATE tivi SET so\_luong = so\_luong - %s WHERE ma\_tivi = %s"

            for item in current\_cart:

                # item = (ma\_tivi, ten\_tivi, so\_luong, don\_gia, thanh\_tien)

                ma\_tivi = item[0]

                so\_luong\_mua = item[2]

                don\_gia = item[3]

                # Thêm vào chitiet\_hoadon

                cursor.execute(sql\_chitiet, (ma\_hd\_moi, ma\_tivi, so\_luong\_mua, don\_gia))

                # Cập nhật số lượng Tivi (tivi)

                cursor.execute(sql\_update\_tivi, (so\_luong\_mua, ma\_tivi))

            # --- Kết thúc Transaction (lưu) ---

            conn.commit()

            messagebox.showinfo("Thành công", f"Đã lưu hóa đơn (Mã HĐ: {ma\_hd\_moi}) thành công!")

            # Reset giỏ hàng và giao diện

            current\_cart.clear()

            update\_cart\_tree(cart\_tree, total\_label)

            load\_combobox\_data()

        except Exception as e:

            if conn:

                conn.rollback()

            messagebox.showerror("Lỗi Giao dịch", f"Không thể lưu hóa đơn: {e}")

        finally:

            if conn:

                conn.close()

    # --- Gán sự kiện và Tải dữ liệu ---

    tivi\_combo.bind('<<ComboboxSelected>>', on\_tivi\_select)

    load\_combobox\_data()

    # Kích hoạt sự kiện lần đầu

    on\_tivi\_select(None)

CODE CONNECT DATABASE MY SQL

import mysql.connector

def connect\_db():

    return mysql.connector.connect(

        host="Localhost",

        user="root",       # thay bằng tài khoản MySQL của bạn

        password="123456",       # thay bằng mật khẩu MySQL

        database="ql\_cuahangtv"

    )

# V. KẾT QUẢ ĐẠT ĐƯỢC

Hệ thống chạy ổn định, hỗ trợ đầy đủ các chức năng CRUD. Giao diện dễ sử dụng, dữ liệu được lưu trữ và cập nhật chính xác.

VI. HƯỚNG PHÁT TRIỂN

- Thêm tính năng đăng nhập, phân quyền.  
- Xuất báo cáo ra Excel hoặc PDF.  
- Tích hợp chức năng thống kê doanh thu.  
- Xây dựng phiên bản Web/Mobile.

# VII. KẾT LUẬN

Phần mềm Quản lý Cửa hàng Tivi là một ứng dụng nhỏ nhưng thực tế, giúp sinh viên rèn luyện kỹ năng lập trình giao diện, thao tác cơ sở dữ liệu và xây dựng hệ thống quản lý hoàn chỉnh.

Sinh viên có thể mở rộng hệ thống này thành phần mềm bán hàng hoặc quản lý kho chuyên nghiệp trong tương lai.

# VIII. NGUỒN THAM KHẢO

“Giáo trình Python cơ bản” – Phạm Văn Vinh & Lê Phương Nam.

Link: [https://okmindmap.com/map/file/53164/9097/Giao-trinh-python-co-ban-9642-2-10.pdf](https://okmindmap.com/map/file/53164/9097/Giao-trinh-python-co-ban-9642-2-10.pdf?utm_source=chatgpt.com" \t "_new) [okmindmap.com](https://okmindmap.com/map/file/53164/9097/Giao-trinh-python-co-ban-9642-2-10.pdf?utm_source=chatgpt.com" \t "_blank)

“Kết nối Python với MySQL” trên VietTuts.

Link: [https://www.viettuts.vn/python/ket-noi-python-voi-mysql](https://www.viettuts.vn/python/ket-noi-python-voi-mysql" \t "_new) [VietTuts](https://viettuts.vn/python/ket-noi-python-voi-mysql?utm_source=chatgpt.com" \t "_blank)

“Hướng dẫn Tkinter trong Python: Xây dựng ứng dụng GUI đơn giản”.

Link: [https://caodang.fpt.edu.vn/tin-tuc-poly/huong-dan-tkinter-trong-python-xay-dung-ung-dung-gui-don-gian.html](https://caodang.fpt.edu.vn/tin-tuc-poly/huong-dan-tkinter-trong-python-xay-dung-ung-dung-gui-don-gian.html?utm_source=chatgpt.com" \t "_new) [Cao Đẳng FPT](https://caodang.fpt.edu.vn/tin-tuc-poly/huong-dan-tkinter-trong-python-xay-dung-ung-dung-gui-don-gian.html?utm_source=chatgpt.com" \t "_blank)

“Lập trình ứng dụng với cơ sở dữ liệu MySQL trong Python” trên trang của Đại học Duy Tân.

Link: [https://kcntt.duytan.edu.vn/Home/ArticleDetail/vn/128/4894/lap-trinh-ung-dung-voi-co-so-du-lieu-mysql-trong-python](https://kcntt.duytan.edu.vn/Home/ArticleDetail/vn/128/4894/lap-trinh-ung-dung-voi-co-so-du-lieu-mysql-trong-python?utm_source=chatgpt.com" \t "_new) [Duy Tân IT](https://kcntt.duytan.edu.vn/Home/ArticleDetail/vn/128/4894/lap-trinh-ung-dung-voi-co-so-du-lieu-mysql-trong-python?utm_source=chatgpt.com" \t "_blank)

“Tài liệu: Sử dụng giao diện đồ họa Tkinter và thư viện xử lý ảnh… (PDF)”. Link: [https://docx.com.vn/tai-lieu/su-dung-giao-dien-do-hoa-tkinter-va-thu-vien-su-ly-anh-opencv2-de-viet-120824](https://docx.com.vn/tai-lieu/su-dung-giao-dien-do-hoa-tkinter-va-thu-vien-su-ly-anh-opencv2-de-viet-120824?utm_source=chatgpt.com" \t "_new) [Docx.com.vn](https://docx.com.vn/tai-lieu/su-dung-giao-dien-do-hoa-tkinter-va-thu-vien-su-ly-anh-opencv2-de-viet-120824?utm_source=chatgpt.com" \t "_blank)