

LỜI CẢM ƠN

Kính gửi thầy Nguyễn Ngọc Minh, giáo viên môn Chuyên đề Python.

Chúng em xin chân thành cảm ơn thầy đã hướng dẫn và hỗ trợ chúng em trong quá trình thực hiện báo cáo về phần mềm quản lý cửa hàng bán TV. Nhờ có sự chỉ bảo tận tình và những ý kiến quý báu của thầy, chúng em đã có thể hoàn thiện báo cáo một cách tốt nhất.

Trong quá trình nghiên cứu, chúng em đã học hỏi được rất nhiều kiến thức bổ ích về cách thức hoạt động của phần mềm quản lý, cũng như những yếu tố cần thiết để xây dựng một hệ thống hiệu quả cho cửa hàng bán TV. Sự động viên và khích lệ của thầy đã giúp chúng em vượt qua những khó khăn và thách thức trong quá trình thực hiện.

Chúng em hy vọng rằng báo cáo này sẽ không chỉ đáp ứng được yêu cầu học tập mà còn có thể mang lại giá trị thực tiễn cho những ai đang quan tâm đến việc quản lý cửa hàng bán TV. Một lần nữa, chúng em xin chân thành cảm ơn thầy và mong nhận được sự góp ý từ thầy để em có thể cải thiện hơn nữa trong tương lai.

MỤC LỤC

I. TỔNG QUÁT.....	1
1. Đặt vấn đề.....	1
1.1. Tính cần thiết của đề tài.....	1
1.2. Mục tiêu nghiên cứu.....	1
2. Tổng quan và cơ sở lý thuyết.....	1
2.1. Đặt vấn đề tổng quan.....	1
2.2. Lịch sử giải quyết vấn đề.....	1
2.3. Phạm vi của đề tài.....	1
2.4. Phương pháp nghiên cứu.....	2
2.5. Cơ sở lý thuyết.....	2
2.6. Thiết kế cơ sở dữ liệu MySQL.....	3
2.7 Viết mã PyThon kết nối cơ sở dữ liệu.....	3
II. Kết quả đạt được và Demo Code.....	4
1. Thiết kế giao diện quản lý cửa hàng TV.....	4
2. Cài đặt môi trường.....	5
3. Chuẩn bị CSDL MySQL.....	6
4. Viết code ứng dụng với Tkinter + MySQL.....	8

I. TỔNG QUÁT

1. Đặt vấn đề

1.1. Tính cần thiết của đề tài

Trong bối cảnh kinh tế hiện nay, nhu cầu sử dụng các thiết bị điện tử, đặc biệt là tivi, ngày càng tăng cao. Các cửa hàng kinh doanh mặt hàng này phải quản lý nhiều thông tin như sản phẩm, nhà cung cấp, khách hàng, hóa đơn, doanh thu,... Nếu quản lý thủ công bằng sổ sách hoặc file Excel sẽ dễ xảy ra sai sót, mất dữ liệu, khó tra cứu và tổng hợp báo cáo.

Do đó, việc xây dựng một phần mềm quản lý cửa hàng bán TV là cần thiết nhằm:

- Tự động hóa quá trình quản lý sản phẩm.
- Tiết kiệm thời gian, nâng cao hiệu quả công việc.
- Hỗ trợ người quản lý nắm bắt tình hình kinh doanh nhanh chóng, chính xác.

1.2. Mục tiêu nghiên cứu

- Xây dựng phần mềm quản lý cửa hàng bán TV có giao diện thân thiện, dễ sử dụng.
- Quản lý thông tin sản phẩm, khách hàng, hóa đơn một cách hệ thống.
- Thực hiện được các chức năng cơ bản: thêm, sửa, xóa, tìm kiếm, thống kê.
- Ứng dụng được ngôn ngữ Python, thư viện Tkinter và hệ quản trị cơ sở dữ liệu MySQL trong phát triển phần mềm thực tế.

2. Tổng quan và cơ sở lý thuyết

2.1. Đặt vấn đề tổng quan

Hiện nay có nhiều cửa hàng đã ứng dụng công nghệ thông tin trong quản lý, tuy nhiên đa số hệ thống có chi phí cao, phức tạp hoặc yêu cầu hạ tầng mạng. Vì vậy, việc xây dựng một phần mềm quản lý đơn giản, hoạt động độc lập trên máy tính cá nhân là cần thiết và phù hợp với các cửa hàng quy mô nhỏ.

2.2. Lịch sử giải quyết vấn đề

Trên thực tế, đã có nhiều phần mềm quản lý bán hàng điện tử như KiotViet, Sapo,... nhưng đa số là phần mềm thương mại, đóng gói sẵn, ít linh hoạt.

Đề tài này hướng đến tự xây dựng một hệ thống nhỏ gọn, có thể tùy chỉnh theo nhu cầu thực tế của cửa hàng, phục vụ cho mục đích học tập và thực hành.

2.3. Phạm vi của đề tài

- **Đối tượng quản lý:** sản phẩm (tivi), khách hàng, hóa đơn bán hàng.

- **Phạm vi dữ liệu:** chỉ thực hiện các nghiệp vụ cơ bản của cửa hàng bán lẻ TV.
- **Không bao gồm:** quản lý nhập hàng, công nợ, hoặc kết nối với thiết bị phần cứng như máy in hóa đơn thực tế.

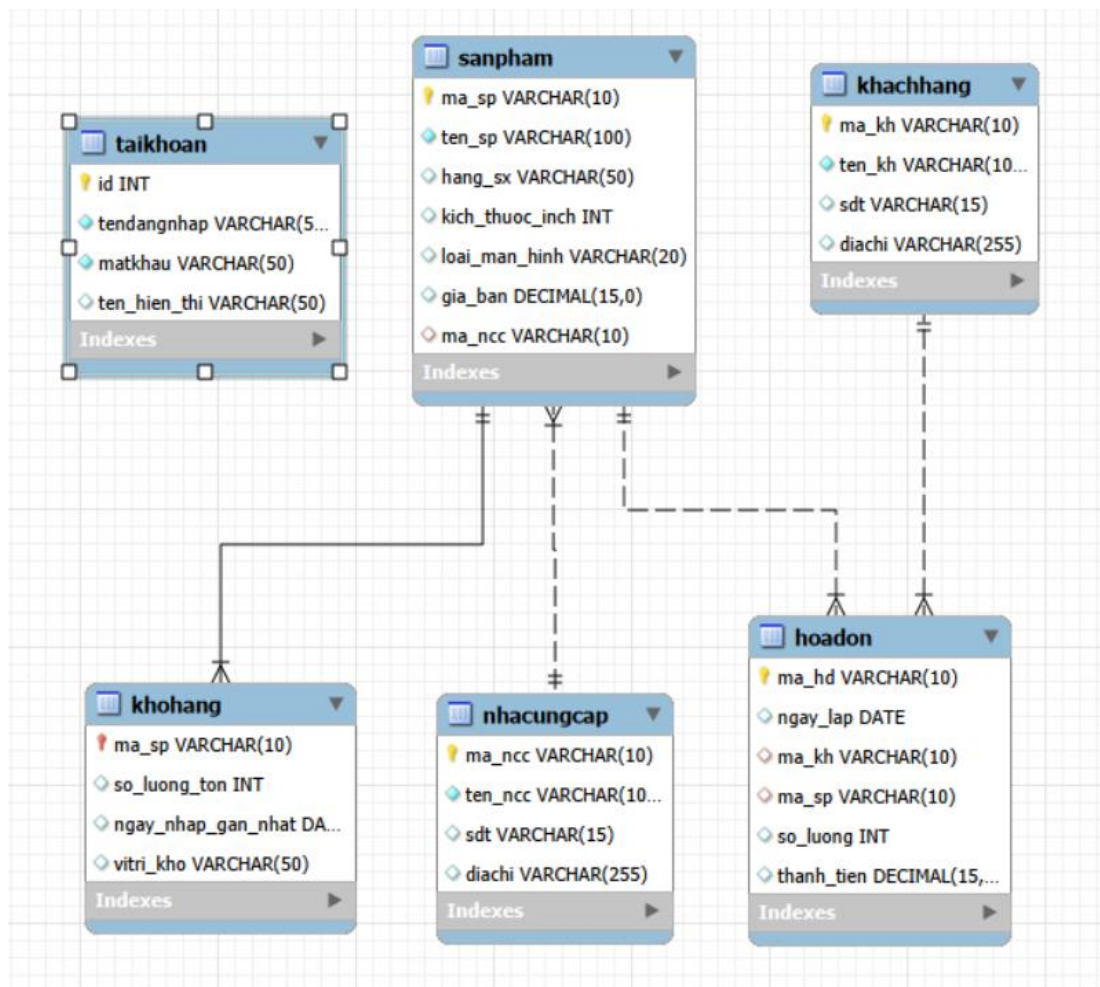
2.4. Phương pháp nghiên cứu

- Đề tài được thực hiện dựa trên phương pháp phân tích và thiết kế hệ thống thông tin kết hợp với lập trình hướng đối tượng.
- Người thực hiện tiến hành khảo sát yêu cầu thực tế, xác định các chức năng cần có của hệ thống, từ đó xây dựng sơ đồ cơ sở dữ liệu, thiết kế giao diện người dùng và lập trình các chức năng xử lý bằng Python kết hợp thư viện Tkinter.
- Việc kết nối giữa chương trình và cơ sở dữ liệu được thực hiện thông qua thư viện mysql-connector-python, giúp truyền tải dữ liệu hai chiều giữa giao diện và MySQL.

2.5. Cơ sở lý thuyết

- **Python:** Là ngôn ngữ lập trình thông dịch, có cấu trúc rõ ràng, cú pháp ngắn gọn, được ứng dụng rộng rãi trong các lĩnh vực như khoa học dữ liệu, trí tuệ nhân tạo và phát triển ứng dụng.
- **Tkinter:** Là thư viện giao diện đồ họa tích hợp trong Python, hỗ trợ tạo các cửa sổ, nhãn, ô nhập liệu, nút lệnh và bảng dữ liệu, giúp người dùng dễ dàng thao tác với chương trình.
- **MySQL:** Là hệ quản trị cơ sở dữ liệu quan hệ (RDBMS) phổ biến, được sử dụng để lưu trữ, truy xuất và quản lý thông tin một cách có tổ chức.
- **mysql-connector-python:** Là thư viện trung gian giúp Python kết nối và tương tác trực tiếp với MySQL, thực hiện các thao tác như thêm, sửa, xóa, tìm kiếm và truy vấn dữ liệu một cách hiệu quả.

2.6. Thiết kế cơ sở dữ liệu MySQL.

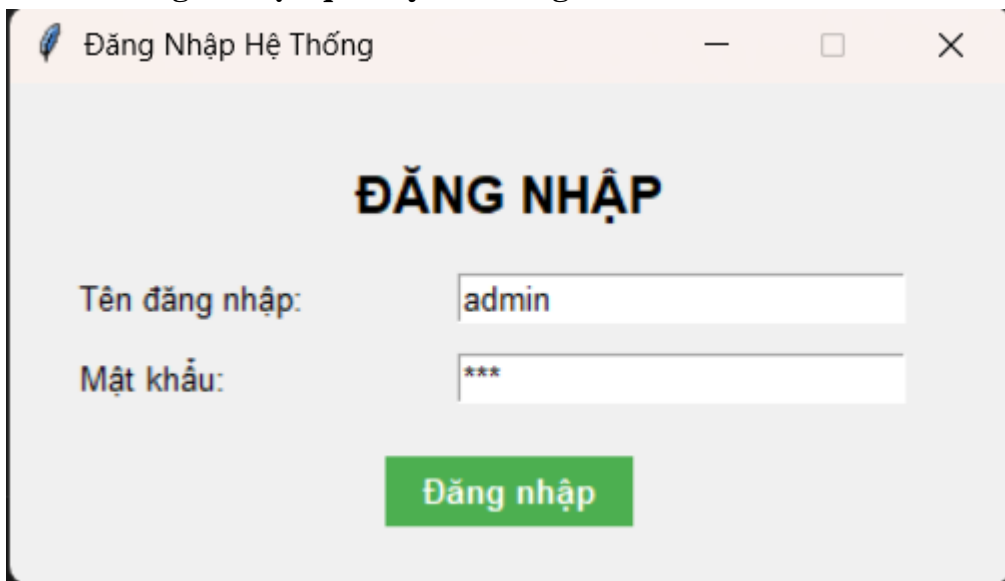


2.7 Viết mã PyThon kết nối cơ sở dữ liệu.

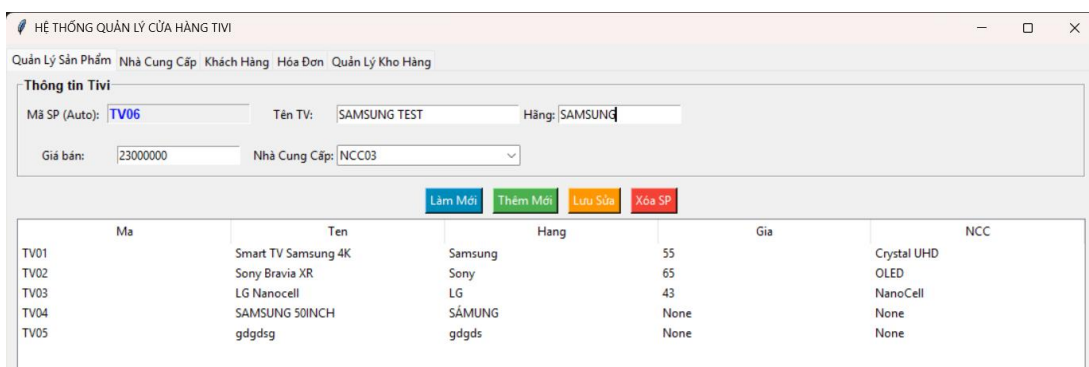
```
14 # --- HÀM KẾT NỐI CƠ SỞ DỮ LIỆU (CSDL) ---
15 def get_connection():
16     """
17     Tạo và trả về một đối tượng kết nối đến CSDL MySQL.
18     Thông tin kết nối (host, user, password, database) được hardcoded.
19     Hiển thị hộp thoại lỗi nếu kết nối thất bại.
20     """
21     try:
22         # Thử kết nối với thông tin cung cấp
23         return mysql.connector.connect(
24             host="localhost",      # Địa chỉ máy chủ CSDL
25             user="root",           # Tên người dùng CSDL
26             password="tuanthanh7", # Mật khẩu
27             database="quanlytivi"  # Tên CSDL
28         )
29     except Exception as e:
30         # Nếu có lỗi xảy ra (ví dụ: sai mật khẩu, CSDL không tồn tại), hiển thị thông báo
31         messagebox.showerror("Lỗi kết nối CSDL", f"Không thể kết nối đến MySQL: {e}")
32         return None # Trả về None nếu kết nối thất bại
```

II. Kết quả đạt được và Demo Code

1. Thiết kế giao diện quản lý cửa hàng TV.

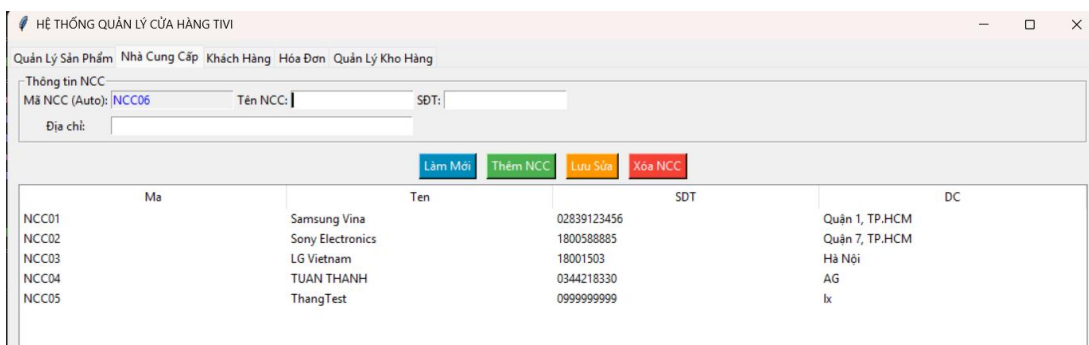


FORM ĐĂNG NHẬP



Ma	Ten	Hãng	Giá	NCC
TV01	Smart TV Samsung 4K	Samsung	55	Crystal UHD
TV02	Sony Bravia XR	Sony	65	OLED
TV03	LG Nanocell	LG	43	NanoCell
TV04	SAMSUNG 50INCH	SAMSUNG	None	None
TV05	gdgdg	gdgds	None	None

TAB QUẢN LÝ SẢN PHẨM



Ma	Ten	SDT	DC
NCC01	Samsung Vina	02839123456	Quận 1, TP.HCM
NCC02	Sony Electronics	1800588885	Quận 7, TP.HCM
NCC03	LG Vietnam	18001503	Hà Nội
NCC04	TUAN THANH	0344218330	AG
NCC05	ThangTest	0999999999	lx

TAB NHÀ CUNG CẤP

HỆ THỐNG QUẢN LÝ CỬA HÀNG TIVI

Quản Lý Sản Phẩm | Nhà Cung Cấp | **Khách Hàng** | Hóa Đơn | Quản Lý Kho Hàng

Thông tin Khách hàng

Mã KH (Auto): KH04 Tên KH: SĐT: Địa chỉ:

Làm Mới Thêm KH Lưu Sửa Xóa KH

	Ma	Ten	SĐT	DC
KH01		Nguyễn Văn A	0909123456	Cần Thơ
KH02		Trần Thị B	0912345678	An Giang
KH03		TUAN THANH	0344218330	LX

TAB KHÁCH HÀNG

HỆ THỐNG QUẢN LÝ CỬA HÀNG TIVI

Quản Lý Sản Phẩm | Nhà Cung Cấp | Khách Hàng | **Hóa Đơn** | Quản Lý Kho Hàng

Lập Hóa Đơn

Mã HD (Auto): HD03 Ngày lập: 2025-12-01 Khách hàng: Sản phẩm: Số lượng: Tổng tiền: (Tự tính khi thêm)

Làm Mới **Lập Hóa Đơn** Xóa HD

	MaHD	Ngày	MaKH	MaSP	SL	
HD01	2025-12-01	KH01	TV02	1	35900000	
HD02	2025-12-01	KH02	TV02	4	143600000	

TAB HÓA ĐƠN

HỆ THỐNG QUẢN LÝ CỬA HÀNG TIVI

Quản Lý Sản Phẩm | Nhà Cung Cấp | Khách Hàng | Hóa Đơn | **Quản Lý Kho Hàng**

Thông tin Kho Hàng

Mã SP: Số Lượng: Vị Trí:

Làm Mới **Nhập Kho** Xuất Kho Lưu Sửa Xóa Khỏi Kho

Mã SP	Tên Sản Phẩm	Số Lượng Tồn	Vị Trí Kho	Ngày Cập Nhật
TV01	Smart TV Samsung 4K	10	Kệ A	2025-10-01
TV02	Sony Bravia XR	5	Kệ B	2025-10-05
TV03	LG Nanocell	20	Kệ A	2025-10-10
TV04	SAMSUNG 50INCH	20	A	2025-12-01

TAB QUẢN LÝ KHO HÀNG

2. Cài đặt môi trường.

- Trước tiên cài đặt các thư viện cần thiết:

- *pip install tk*
- *pip install tkcalendar*
- *pip install mysql-connector-python*

- **Mysql-connector-python:**

Dùng để **kết nối Python với MySQL.**

- Kết nối đến database MySQL.

- Gửi câu lệnh SQL (SELECT, INSERT, UPDATE, DELETE).
- Nhận dữ liệu trả về từ MySQL.

- Tkcalendar:

- Dùng để tạo widget **DateEntry** trong Tkinter, giúp **chọn ngày sinh bằng lịch** thay vì nhập thủ công.
- Điều này hạn chế lỗi sai ngày/tháng/năm và giao diện thân thiện hơn.

- Tk / Tkinter:

- Tạo cửa sổ ứng dụng.
 - Tạo các nút bấm, ô nhập, bảng, menu.
 - Kết hợp với tkcalendar để tạo giao diện có chọn ngày.
- (Lưu ý: Tkinter không cài bằng pip, vì nó đi kèm Python.)

3. Chuẩn bị CSDL MySQL.

```
CREATE DATABASE quanlytivi CHARACTER SET utf8mb4 COLLATE utf8mb4_unicode_ci;
```

```
USE quanlytivi;
```

-- 2. Bảng TÀI KHOẢN (Dùng cho Form Đăng nhập)

```
CREATE TABLE taikhoan (
```

```
    id INT AUTO_INCREMENT PRIMARY KEY,
```

```
    tendangnhap VARCHAR(50) NOT NULL UNIQUE,
```

```
    matkhau VARCHAR(50) NOT NULL, -- Trong đồ án môn học có thể lưu text thường cho đơn giản
```

```
    ten_hien_thi VARCHAR(50)
```

```
);
```

-- 3. Bảng NHÀ CUNG CẤP (Dùng cho Tab Nhà cung cấp)

```
CREATE TABLE nhacungcap (
```

```
    ma_ncc VARCHAR(10) PRIMARY KEY,
```

```
    ten_ncc VARCHAR(100) NOT NULL,
```

```
    sdt VARCHAR(15),
```

```
    diachi VARCHAR(255)
```

```
);
```

-- 4. Bảng SẢN PHẨM (Dùng cho Tab Sản phẩm - Lưu thông tin Tivi)

```
CREATE TABLE sanpham (
```

```
    ma_sp VARCHAR(10) PRIMARY KEY,
```



```

ten_sp VARCHAR(100) NOT NULL,
hang_sx VARCHAR(50),    -- Ví dụ: Sony, Samsung, LG
kich_thuoc_inch INT,    -- Ví dụ: 55, 65, 75
loai_man_hinh VARCHAR(20),-- Ví dụ: OLED, QLED, LED
gia_ban DECIMAL(15, 0),
ma_ncc VARCHAR(10),
FOREIGN KEY (ma_ncc) REFERENCES nhacungcap(ma_ncc) ON
DELETE SET NULL
);

```

-- 5. Bảng KHO HÀNG (Dùng cho Tab Kho hàng - Quản lý số lượng tồn)

```

CREATE TABLE khohang (
    ma_sp VARCHAR(10) PRIMARY KEY,
    so_luong_ton INT DEFAULT 0,
    ngay_nhap_gan_nhat DATE,
    vitri_kho VARCHAR(50),    -- Ví dụ: Kệ A1, Kho 2
    FOREIGN KEY (ma_sp) REFERENCES sanpham(ma_sp) ON DELETE
    CASCADE
);

```

-- 6. Tạo bảng KHÁCH HÀNG

```

CREATE TABLE khachhang (
    ma_kh VARCHAR(10) PRIMARY KEY,
    ten_kh VARCHAR(100) NOT NULL,
    sdt VARCHAR(15),
    diachi VARCHAR(255)
);

```

-- 7. Tạo bảng HÓA ĐƠN

```

CREATE TABLE hoadon (
    ma_hd VARCHAR(10) PRIMARY KEY,
    ngay_lap DATE,
    ma_kh VARCHAR(10),
    ma_sp VARCHAR(10),
    so_luong INT,
    thanh_tien DECIMAL(15,0),

```

```

    FOREIGN KEY (ma_kh) REFERENCES khachhang(ma_kh) ON DELETE
    SET NULL,
    FOREIGN KEY (ma_sp) REFERENCES sanpham(ma_sp) ON DELETE
    SET NULL
);

```

4. Viết code ứng dụng với Tkinter + MySQL.

--- HÀM TẠO MÃ (ID) TỰ ĐỘNG ---

```
def generate_new_id(table, col_name, prefix):
```

```
    """
```

Tạo một mã mới cho một bảng dựa trên mã lớn nhất hiện có.

Ví dụ: Nếu mã lớn nhất là 'TV05', hàm sẽ trả về 'TV06'.

- table: Tên bảng cần tạo mã (vd: 'sanpham').

- col_name: Tên cột chứa mã (vd: 'ma_sp').

- prefix: Tiền tố của mã (vd: 'TV').

```
    """
```

```
    conn = get_connection()
```

```
    if not conn: return f"{prefix}01" # Nếu không kết nối được, trả về mã mặc
    định
```

```
    cur = conn.cursor()
```

```
    cur.execute(f"SELECT {col_name} FROM {table}") # Lấy tất cả mã từ
    bảng
```

```
    data = cur.fetchall()
```

```
    conn.close()
```

```
    max_num = 0
```

```
    for row in data:
```

```
        code = row[0] # Lấy mã từ mỗi hàng
```

```
        if code and code.startswith(prefix): # Nếu mã tồn tại và có đúng tiền tố
```

```
            try:
```

```
                # Tách phần số ra khỏi mã và chuyển thành kiểu integer
```

```
                num_part = int(code[len(prefix):])
```

```

        if num_part > max_num: # Tìm số lớn nhất
            max_num = num_part
    except ValueError:
        # Bỏ qua nếu phần sau tiền tố không phải là số
        pass

    # Trả về mã mới bằng cách lấy số lớn nhất + 1, và định dạng nó với 2 chữ số (vd: 01, 08, 12)
    return f"{prefix}{max_num + 1:02d}"

# PHẦN 3: GIAO DIỆN NGƯỜI DÙNG (UI)
# --- HÀM TẠO CỬA SỔ ĐĂNG NHẬP ---
def tao_giao_dien_dang_nhap():
    """
    Tạo và hiển thị cửa sổ đăng nhập.
    Đây là cửa sổ đầu tiên của ứng dụng.
    """

    login_window = tk.Tk() # Tạo cửa sổ gốc (root window)
    login_window.title("Đăng Nhập Hệ Thống")
    login_window.geometry("400x200")

    login_window.resizable(False, False) # Không cho phép thay đổi kích thước cửa sổ

    # --- Canh giữa cửa sổ ---
    window_width = 400
    window_height = 200

    screen_width = login_window.winfo_screenwidth() # Lấy chiều rộng màn hình
    screen_height = login_window.winfo_screenheight() # Lấy chiều cao màn hình

    # Tính toán tọa độ để cửa sổ nằm giữa
    center_x = int(screen_width / 2 - window_width / 2)
    center_y = int(screen_height / 2 - window_height / 2)

    login_window.geometry(f'{window_width}x{window_height}+{center_x}+{center_y}')

```

```

# --- Tạo các widget trong cửa sổ ---

main_frame = tk.Frame(login_window, padx=20, pady=20) # Frame chính để chứa các widget khác

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


# Tiêu đề

tk.Label(main_frame, text="ĐĂNG NHẬP", font=("Arial", 16, "bold")).grid(row=0, column=0, columnspan=2, pady=10)


# Nhãn và ô nhập tên đăng nhập

tk.Label(main_frame, text="Tên đăng nhập:", font=("Arial", 10)).grid(row=1, column=0, sticky="w", padx=5, pady=5)

username_entry = tk.Entry(main_frame, font=("Arial", 10), width=25)

username_entry.grid(row=1, column=1, padx=5, pady=5)


# Nhãn và ô nhập mật khẩu

tk.Label(main_frame, text="Mật khẩu:", font=("Arial", 10)).grid(row=2, column=0, sticky="w", padx=5, pady=5)

password_entry = tk.Entry(main_frame, show="*", font=("Arial", 10), width=25) # show="*" để ẩn mật khẩu

password_entry.grid(row=2, column=1, padx=5, pady=5)


username_entry.focus_set() # Tự động focus vào ô tên đăng nhập khi mở cửa sổ


# --- HÀM XỬ LÝ LOGIC ĐĂNG NHẬP ---

def xu_ly_dang_nhap(event=None):

    """

    Kiểm tra tên đăng nhập và mật khẩu.

    Nếu đúng, mở giao diện chính. Nếu sai, hiển thị lỗi.

    """

    username = username_entry.get()

    password = password_entry.get()

```

```

# Kiểm tra thông tin đăng nhập (hiện tại đang hardcoded)
if username == "admin" and password == "123":
    login_window.withdraw() # Ẩn cửa sổ đăng nhập đi
    mo_giao_dien_chinh(login_window) # Mở giao diện chính và truyền tham chiếu của cửa sổ login
else:
    messagebox.showerror("Lỗi Đăng Nhập", "Tên đăng nhập hoặc mật khẩu không đúng.")

login_window.bind('<Return>', xu_ly_dang_nhap) # Cho phép nhấn phím Enter để đăng nhập

# Nút đăng nhập
login_button = tk.Button(main_frame, text="Đăng nhập",
    command=xu_ly_dang_nhap, font=("Arial", 10, "bold"), bg="#4CAF50", fg="white", relief="flat", padx=10)
login_button.grid(row=3, column=0, columnspan=2, pady=15)

# Cấu hình grid để các cột co giãn đều
main_frame.grid_columnconfigure(0, weight=1)
main_frame.grid_columnconfigure(1, weight=1)

login_window.mainloop() # Bắt đầu vòng lặp sự kiện cho cửa sổ đăng nhập

# --- HÀM TẠO CỬA SỔ QUẢN LÝ CHÍNH ---
def mo_giao_dien_chinh(login_window):
    """
    Tạo và hiển thị giao diện quản lý chính của ứng dụng sau khi đăng nhập thành công.
    - login_window: Tham chiếu đến cửa sổ đăng nhập để có thể đóng hoàn toàn ứng dụng.
    """
    # Hàm xử lý việc đóng cửa sổ chính
    def on_closing():

```

```
"""Khi đóng cửa sổ này, hủy luôn cửa sổ đăng nhập đã bị ẩn để kết thúc ứng dụng."""
```

```
login_window.destroy()
```

```
root = tk.Toplevel() # Tạo một cửa sổ phụ (Toplevel) thay vì cửa sổ gốc mới
```

```
root.protocol("WM_DELETE_WINDOW", on_closing) # Gán hàm on_closing cho sự kiện đóng cửa sổ
```

```
root.title("HỆ THỐNG QUẢN LÝ CỬA HÀNG TIVI")
```

```
root.geometry("1100x700")
```

```
# Tạo Notebook (hệ thống các tab)
```

```
notebook = ttk.Notebook(root)
```

```
notebook.pack(pady=5, expand=True, fill="both")
```

```
# TAB 1: QUẢN LÝ SẢN PHẨM
```

```
tab_sp = tk.Frame(notebook)
```

```
notebook.add(tab_sp, text="Quản Lý Sản Phẩm")
```

```
# --- Khung nhập liệu thông tin sản phẩm ---
```

```
fr_input_sp = tk.LabelFrame(tab_sp, text="Thông tin Tivi", font=("Arial", 10, "bold"))
```

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

```
tk.Label(fr_input_sp, text="Mã SP (Auto):").grid(row=0, column=0, padx=5, pady=10)
```

```
e_ma_sp = tk.Entry(fr_input_sp, state='readonly', fg="blue", font=("Arial", 10, "bold"))
```

```
e_ma_sp.grid(row=0, column=1)
```

```
tk.Label(fr_input_sp, text="Tên TV:").grid(row=0, column=2)
```

```
e_ten_sp = tk.Entry(fr_input_sp, width=30); e_ten_sp.grid(row=0, column=3)
```

```
tk.Label(fr_input_sp, text="Hãng:").grid(row=0, column=4)
```

```

e_hang = tk.Entry(fr_input_sp); e_hang.grid(row=0, column=5)

tk.Label(fr_input_sp, text="Giá bán:").grid(row=1, column=0, pady=10)
e_gia = tk.Entry(fr_input_sp); e_gia.grid(row=1, column=1)

tk.Label(fr_input_sp, text="Nhà Cung Cấp:").grid(row=1, column=2)
cb_mancc_sp = ttk.Combobox(fr_input_sp, width=27, state="readonly") #
Combobox để chọn NCC
cb_mancc_sp.grid(row=1, column=3)

# --- Các hàm xử lý cho Tab Sản phẩm ---
def load_sp():
    """Tải lại toàn bộ dữ liệu sản phẩm từ CSDL và hiển thị lên Treeview."""
    for i in tree_sp.get_children(): tree_sp.delete(i) # Xóa dữ liệu cũ trên Treeview
    conn = get_connection()
    if conn:
        cur = conn.cursor()
        cur.execute("SELECT * FROM sanpham") # Lấy tất cả sản phẩm
        for row in cur.fetchall(): tree_sp.insert("", tk.END, values=row)

    # Tải danh sách mã NCC cho Combobox
    cur.execute("SELECT ma_ncc FROM nhacungcap")
    cb_mancc_sp['values'] = [r[0] for r in cur.fetchall()]
    conn.close()

    reset_form_sp() # Reset form sau khi tải

def reset_form_sp():
    """Xóa trắng các ô nhập liệu và tạo mã sản phẩm mới."""
    e_ma_sp.config(state='normal'); e_ma_sp.delete(0, tk.END)
    e_ma_sp.insert(0, generate_new_id("sanpham", "ma_sp", "TV")) # Tạo mã mới
    e_ma_sp.config(state='readonly')

```

```

        e_ten_sp.delete(0, tk.END); e_hang.delete(0, tk.END); e_gia.delete(0,
tk.END); cb_mancc_sp.set("")

def them_sp():
    """Thêm một sản phẩm mới vào CSDL."""
    sql = "INSERT INTO sanpham VALUES (%s, %s, %s, %s, %s)"
    val = (e_ma_sp.get(), e_ten_sp.get(), e_hang.get(), e_gia.get(),
cb_mancc_sp.get())
    if execute_query(sql, val, "Đã thêm sản phẩm thành công!"): load_sp() #
Tải lại dữ liệu nếu thành công

def sua_sp():
    """Cập nhật thông tin một sản phẩm đã có."""
    sql = "UPDATE sanpham SET ten_sp=%s, hang_sx=%s, gia_ban=%s,
ma_ncc=%s WHERE ma_sp=%s"
    val = (e_ten_sp.get(), e_hang.get(), e_gia.get(), cb_mancc_sp.get(),
e_ma_sp.get())
    if execute_query(sql, val, "Đã cập nhật sản phẩm thành công!"): load_sp()

def xoa_sp():
    """Xóa một sản phẩm khỏi CSDL."""
    if messagebox.askyesno("Xác Nhận Xóa", "Bạn có chắc chắn muốn xóa
sản phẩm này không?"):
        if execute_query("DELETE FROM sanpham WHERE ma_sp=%s",
(e_ma_sp.get(),), "Đã xóa sản phẩm!"): load_sp()

def on_click_sp(event):
    """Sự kiện khi người dùng click vào một dòng trên Treeview."""
    sel = tree_sp.selection() # Lấy dòng được chọn
    if sel:
        val = tree_sp.item(sel[0], 'values') # Lấy giá trị của dòng đó
        # Điền thông tin của dòng được chọn lên các ô nhập liệu
        e_ma_sp.config(state='normal'); e_ma_sp.delete(0, tk.END);
e_ma_sp.insert(0, val[0]); e_ma_sp.config(state='readonly')
        e_ten_sp.delete(0, tk.END); e_ten_sp.insert(0, val[1])
        e_hang.delete(0, tk.END); e_hang.insert(0, val[2])

```



```
e_gia.delete(0, tk.END); e_gia.insert(0, str(val[3]))
cb_mancc_sp.set(str(val[4]))
```

--- Khung chứa các nút bấm ---

```
fr_btn_sp = tk.Frame(tab_sp); fr_btn_sp.pack(pady=5)

tk.Button(fr_btn_sp, text="Làm Mới", bg="#008CBA", fg="white",
command=reset_form_sp).pack(side="left", padx=5)

tk.Button(fr_btn_sp, text="Thêm Mới", bg="#4CAF50", fg="white",
command=them_sp).pack(side="left", padx=5)

tk.Button(fr_btn_sp, text="Lưu Sửa", bg="#ff9800", fg="white",
command=sua_sp).pack(side="left", padx=5)

tk.Button(fr_btn_sp, text="Xóa SP", bg="#f44336", fg="white",
command=xoa_sp).pack(side="left", padx=5)
```

--- Bảng hiển thị dữ liệu (Treeview) ---

```
tree_sp = ttk.Treeview(tab_sp, columns=("Ma", "Ten", "Hang", "Gia",
"NCC"), show="headings", height=10)

for c in ("Ma", "Ten", "Hang", "Gia", "NCC"): tree_sp.heading(c, text=c);
tree_sp.column(c, width=100)

tree_sp.pack(fill="both", expand=True, padx=10)

tree_sp.bind("<<TreeviewSelect>>", on_click_sp) # Gán sự kiện click vào
bảng
```

TAB 2: NHÀ CUNG CẤP

```
tab_ncc = tk.Frame(notebook)
notebook.add(tab_ncc, text="Nhà Cung Cấp")

fr_ncc = tk.LabelFrame(tab_ncc, text="Thông tin NCC")
fr_ncc.pack(fill="x", padx=10, pady=5)

tk.Label(fr_ncc, text="Mã NCC (Auto):").grid(row=0, column=0)
e_mncc = tk.Entry(fr_ncc, state='readonly', fg="blue"); e_mncc.grid(row=0,
column=1)

tk.Label(fr_ncc, text="Tên NCC:").grid(row=0, column=2)
e_tncc = tk.Entry(fr_ncc); e_tncc.grid(row=0, column=3)
```

```

tk.Label(fr_ncc, text="SĐT:").grid(row=0, column=4)
e_sdt_ncc = tk.Entry(fr_ncc); e_sdt_ncc.grid(row=0, column=5)

tk.Label(fr_ncc, text="Địa chỉ:").grid(row=1, column=0, pady=5)
e_dc_ncc = tk.Entry(fr_ncc, width=50); e_dc_ncc.grid(row=1, column=1,
columnspan=5, sticky="w")

def load_ncc():
    """Tải dữ liệu nhà cung cấp."""
    for i in tree_ncc.get_children(): tree_ncc.delete(i)
    conn = get_connection()
    if conn:
        cur = conn.cursor()
        cur.execute("SELECT * FROM nhacungcap")
        for row in cur.fetchall(): tree_ncc.insert("", tk.END, values=row)
        conn.close()
    reset_form_ncc()

def reset_form_ncc():
    """Làm mới form nhà cung cấp."""
    e_mncc.config(state='normal'); e_mncc.delete(0, tk.END)
    e_mncc.insert(0, generate_new_id("nhacungcap", "ma_ncc", "NCC"))
    e_mncc.config(state='readonly')
    e_tncc.delete(0, tk.END); e_sdt_ncc.delete(0, tk.END); e_dc_ncc.delete(0,
tk.END)

def them_ncc():
    """Thêm nhà cung cấp mới."""
    if execute_query("INSERT INTO nhacungcap VALUES (%s, %s, %s,
%s)",
                    (e_mncc.get(), e_tncc.get(), e_sdt_ncc.get(), e_dc_ncc.get()),
                    "Thêm NCC thành công!"): load_ncc()

def sua_ncc():

```

```

        """Sửa thông tin nhà cung cấp."""
        if execute_query("UPDATE nhacungcap SET ten_ncc=%s, sdt=%s,
        diachi=%s WHERE ma_ncc=%s",
            (e_tncc.get(), e_sdt_ncc.get(), e_dc_ncc.get(), e_mncc.get()),
            "Sửa NCC thành công!"): load_ncc()

def xoa_ncc():
    """Xóa nhà cung cấp."""
    if messagebox.askyesno("Xác Nhận Xóa", "Bạn có chắc muốn xóa nhà
    cung cấp này?"):
        if execute_query("DELETE FROM nhacungcap WHERE ma_ncc=%s",
            (e_mncc.get(),), "Đã xóa NCC!"): load_ncc()

def on_click_ncc(event):
    """Sự kiện click vào bảng NCC."""
    sel = tree_ncc.selection()
    if sel:
        val = tree_ncc.item(sel[0], 'values')
        e_mncc.config(state='normal'); e_mncc.delete(0, tk.END);
        e_mncc.insert(0, val[0]); e_mncc.config(state='readonly')
        e_tncc.delete(0, tk.END); e_tncc.insert(0, val[1])
        e_sdt_ncc.delete(0, tk.END); e_sdt_ncc.insert(0, val[2])
        e_dc_ncc.delete(0, tk.END); e_dc_ncc.insert(0, val[3])

fr_btn_ncc = tk.Frame(tab_ncc); fr_btn_ncc.pack(pady=5)
tk.Button(fr_btn_ncc, text="Làm Mới", bg="#008CBA", fg="white",
command=reset_form_ncc).pack(side="left", padx=5)
tk.Button(fr_btn_ncc, text="Thêm NCC", bg="#4CAF50", fg="white",
command=them_ncc).pack(side="left", padx=5)
tk.Button(fr_btn_ncc, text="Lưu Sửa", bg="#ff9800", fg="white",
command=sua_ncc).pack(side="left", padx=5)
tk.Button(fr_btn_ncc, text="Xóa NCC", bg="#f44336", fg="white",
command=xoa_ncc).pack(side="left", padx=5)

tree_ncc = ttk.Treeview(tab_ncc, columns=("Ma", "Ten", "SDT", "DC"),
show="headings", height=8)

```

```

for c in ("Ma", "Ten", "SDT", "DC"): tree_ncc.heading(c, text=c)
tree_ncc.pack(fill="both", expand=True, padx=10)
tree_ncc.bind("<<TreeviewSelect>>", on_click_ncc)

# TAB 3: KHÁCH HÀNG
tab_kh = tk.Frame(notebook)
notebook.add(tab_kh, text="Khách Hàng")

fr_kh = tk.LabelFrame(tab_kh, text="Thông tin Khách hàng")
fr_kh.pack(fill="x", padx=10, pady=5)

tk.Label(fr_kh, text="Mã KH (Auto):").grid(row=0, column=0)
e_makh = tk.Entry(fr_kh, state='readonly', fg="blue"); e_makh.grid(row=0,
column=1)

tk.Label(fr_kh, text="Tên KH:").grid(row=0, column=2)
e_tenkh = tk.Entry(fr_kh); e_tenkh.grid(row=0, column=3)

tk.Label(fr_kh, text="SDT:").grid(row=0, column=4)
e_sdt_kh = tk.Entry(fr_kh); e_sdt_kh.grid(row=0, column=5)

tk.Label(fr_kh, text="Địa chỉ:").grid(row=1, column=0, pady=5)
e_dc_kh = tk.Entry(fr_kh, width=50); e_dc_kh.grid(row=1, column=1,
columnspan=5, sticky="w")

def load_kh():
    """Tải dữ liệu khách hàng."""
    for i in tree_kh.get_children(): tree_kh.delete(i)
    conn = get_connection()
    if conn:
        cur = conn.cursor()
        cur.execute("SELECT * FROM khachhang")
        for row in cur.fetchall(): tree_kh.insert("", tk.END, values=row)
        conn.close()

```

```

reset_form_kh()

def reset_form_kh():
    """Làm mới form khách hàng."""
    e_makh.config(state='normal'); e_makh.delete(0, tk.END)
    e_makh.insert(0, generate_new_id("khachhang", "ma_kh", "KH"))
    e_makh.config(state='readonly')
    e_tenkh.delete(0, tk.END); e_sdt_kh.delete(0, tk.END); e_dc_kh.delete(0, tk.END)

def them_kh():
    """Thêm khách hàng mới."""
    if execute_query("INSERT INTO khachhang VALUES (%s, %s, %s, %s)",
                    (e_makh.get(), e_tenkh.get(), e_sdt_kh.get(), e_dc_kh.get()),
                    "Thêm KH thành công!"): load_kh()

def sua_kh():
    """Sửa thông tin khách hàng."""
    if execute_query("UPDATE khachhang SET ten_kh=%s, sdt=%s,
                    diachi=%s WHERE ma_kh=%s",
                    (e_tenkh.get(), e_sdt_kh.get(), e_dc_kh.get(), e_makh.get()),
                    "Sửa KH thành công!"): load_kh()

def xoa_kh():
    """Xóa khách hàng."""
    if messagebox.askyesno("Xác Nhận Xóa", "Bạn có chắc muốn xóa khách hàng này?"):
        if execute_query("DELETE FROM khachhang WHERE ma_kh=%s",
                        (e_makh.get(),), "Đã xóa khách hàng!"): load_kh()

def on_click_kh(event):
    """Sự kiện click vào bảng khách hàng."""
    sel = tree_kh.selection()
    if sel:
        val = tree_kh.item(sel[0], 'values')

```

```

        e_makh.config(state='normal'); e_makh.delete(0, tk.END);
e_makh.insert(0, val[0]); e_makh.config(state='readonly')
        e_tenkh.delete(0, tk.END); e_tenkh.insert(0, val[1])
        e_sdt_kh.delete(0, tk.END); e_sdt_kh.insert(0, val[2])
        e_dc_kh.delete(0, tk.END); e_dc_kh.insert(0, val[3])

fr_btn_kh = tk.Frame(tab_kh); fr_btn_kh.pack(pady=5)
        tk.Button(fr_btn_kh, text="Làm Mới", bg="#008CBA", fg="white",
command=reset_form_kh).pack(side="left", padx=5)
        tk.Button(fr_btn_kh, text="Thêm KH", bg="#4CAF50", fg="white",
command=them_kh).pack(side="left", padx=5)
        tk.Button(fr_btn_kh, text="Lưu Sửa", bg="#ff9800", fg="white",
command=sua_kh).pack(side="left", padx=5)
        tk.Button(fr_btn_kh, text="Xóa KH", bg="#f44336", fg="white",
command=xoa_kh).pack(side="left", padx=5)

tree_kh = ttk.Treeview(tab_kh, columns=("Ma", "Ten", "SDT", "DC"),
show="headings", height=8)
for c in ("Ma", "Ten", "SDT", "DC"): tree_kh.heading(c, text=c)
tree_kh.pack(fill="both", expand=True, padx=10)
tree_kh.bind("<<TreeviewSelect>>", on_click_kh)

# TAB 4: HÓA ĐƠN
tab_hd = tk.Frame(notebook)
notebook.add(tab_hd, text="Hóa Đơn")

fr_hd = tk.LabelFrame(tab_hd, text="Lập Hóa Đơn")
fr_hd.pack(fill="x", padx=10, pady=5)

tk.Label(fr_hd, text="Mã HD (Auto):").grid(row=0, column=0)
e_mahd = tk.Entry(fr_hd, state='readonly', fg="blue"); e_mahd.grid(row=0,
column=1)

tk.Label(fr_hd, text="Ngày lập:").grid(row=0, column=2)
        e_ngayhd = DateEntry(fr_hd, date_pattern='yyyy-MM-dd');
e_ngayhd.grid(row=0, column=3)

```

```

tk.Label(fr_hd, text="Khách hàng:").grid(row=1, column=0, pady=5)
    cb_kh_hd = ttk.Combobox(fr_hd, width=25, state="readonly");
cb_kh_hd.grid(row=1, column=1)

tk.Label(fr_hd, text="Sản phẩm:").grid(row=1, column=2)
    cb_sp_hd = ttk.Combobox(fr_hd, width=25, state="readonly");
cb_sp_hd.grid(row=1, column=3)

tk.Label(fr_hd, text="Số lượng:").grid(row=2, column=0, pady=5)
e_sl_hd = tk.Entry(fr_hd); e_sl_hd.grid(row=2, column=1)

tk.Label(fr_hd, text="Tổng tiền:").grid(row=2, column=2)
    e_tongtien = tk.Entry(fr_hd, state='readonly'); e_tongtien.grid(row=2,
column=3)
    tk.Label(fr_hd, text="(Tự tính khi thêm)").grid(row=2, column=4,
sticky="w")

def load_hd():
    """Tải dữ liệu hóa đơn và các combobox liên quan."""
    for i in tree_hd.get_children(): tree_hd.delete(i)
    conn = get_connection()
    if conn:
        cur = conn.cursor()
        cur.execute("SELECT * FROM hoadon")
        for row in cur.fetchall(): tree_hd.insert("", tk.END, values=row)

# Tải danh sách khách hàng cho combobox
cur.execute("SELECT ma_kh, ten_kh FROM khachhang")
cb_kh_hd['values'] = [f"{r[0]} - {r[1]}" for r in cur.fetchall()]

# Tải danh sách sản phẩm cho combobox
cur.execute("SELECT ma_sp, ten_sp, gia_ban FROM sanpham")
cb_sp_hd['values'] = [f"{r[0]} - {r[1]} ({r[2]})" for r in cur.fetchall()]

```

```

        conn.close()
    reset_form_hd()

def reset_form_hd():
    """Làm mới form hóa đơn."""
    e_mahd.config(state='normal'); e_mahd.delete(0, tk.END)
    e_mahd.insert(0, generate_new_id("hoadon", "ma_hd", "HD"))
    e_mahd.config(state='readonly')
    e_sl_hd.delete(0, tk.END); e_tongtien.config(state='normal');
    e_tongtien.delete(0, tk.END); e_tongtien.config(state='readonly')
    cb_kh_hd.set(""); cb_sp_hd.set("")

def them_hd():
    """Thêm hóa đơn mới, tự động tính tổng tiền."""
    try:
        sl = int(e_sl_hd.get())
        msp_full = cb_sp_hd.get()
        ma_sp = msp_full.split(" - ")[0]

        # Lấy giá bán của sản phẩm từ CSDL để tính tiền
        conn = get_connection()
        cur = conn.cursor()
        cur.execute("SELECT gia_ban FROM sanpham WHERE ma_sp=%s",
(ma_sp,))
        gia = cur.fetchone()[0]
        conn.close()

        thanh_tien = gia * sl
        ma_kh = cb_kh_hd.get().split(" - ")[0]

        sql = "INSERT INTO hoadon VALUES (%s, %s, %s, %s, %s, %s)"
        val = (e_mahd.get(), e_ngayhd.get(), ma_kh, ma_sp, sl, thanh_tien)
        if execute_query(sql, val, f"Đã lập hóa đơn! Tổng tiền: {thanh_tien:,.0f} VND"): load_hd()

```



```

except Exception as e:
    messagebox.showerror("Lỗi", "Vui lòng chọn sản phẩm, khách hàng và
nhập số lượng hợp lệ!")

def xoa_hd():
    """Xóa một hóa đơn."""
    if messagebox.askyesno("Xác Nhận Xóa", "Bạn có chắc muốn xóa hóa
đơn này?"):
        if execute_query("DELETE FROM hoaddon WHERE ma_hd=%s",
(e_mahd.get(),), "Đã xóa hóa đơn!"): load_hd()

def on_click_hd(event):
    """Sự kiện click vào bảng hóa đơn."""
    sel = tree_hd.selection()
    if sel:
        val = tree_hd.item(sel[0], 'values')
        e_mahd.config(state='normal'); e_mahd.delete(0, tk.END);
e_mahd.insert(0, val[0]); e_mahd.config(state='readonly')
        e_ngayhd.set_date(val[1])
        # Set combo (hiện tại chỉ hiển thị mã, muốn hiển thị cả tên và mã
cần xử lý phức tạp hơn)
        cb_kh_hd.set(val[2])
        cb_sp_hd.set(val[3])
        e_sl_hd.delete(0, tk.END); e_sl_hd.insert(0, val[4])
        e_tongtien.config(state='normal'); e_tongtien.delete(0, tk.END);
e_tongtien.insert(0, str(val[5])); e_tongtien.config(state='readonly')

fr_btn_hd = tk.Frame(tab_hd); fr_btn_hd.pack(pady=5)
tk.Button(fr_btn_hd, text="Làm Mới", bg="#008CBA", fg="white",
command=reset_form_hd).pack(side="left", padx=5)
tk.Button(fr_btn_hd, text="Lập Hóa Đơn", bg="#4CAF50", fg="white",
command=them_hd).pack(side="left", padx=5)
tk.Button(fr_btn_hd, text="Xóa HD", bg="#f44336", fg="white",
command=xoa_hd).pack(side="left", padx=5)

```

```

tree_hd = ttk.Treeview(tab_hd, columns=("MaHD", "Ngày", "MaKH",
"MaSP", "SL", "TongTien"), show="headings", height=8)
for c in ("MaHD", "Ngày", "MaKH", "MaSP", "SL", "TongTien"):
tree_hd.heading(c, text=c)
tree_hd.pack(fill="both", expand=True, padx=10)
tree_hd.bind("<<TreeviewSelect>>", on_click_hd)

```

TAB 5: KHO HÀNG

```

tab_kho = tk.Frame(notebook)
notebook.add(tab_kho, text="Quản Lý Kho Hàng")

```

--- Khung Nhập Liệu ---

```

fr_kho = tk.LabelFrame(tab_kho, text="Thông tin Kho Hàng", font=("Arial",
10, "bold"))

```

```

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

```

```

tk.Label(fr_kho, text="Mã SP:").grid(row=0, column=0, padx=5, pady=10)

```

```

cb_msp_kho = ttk.Combobox(fr_kho, width=22, state="readonly")

```

```

cb_msp_kho.grid(row=0, column=1)

```

```

tk.Label(fr_kho, text="Số Lượng:").grid(row=0, column=2)

```

```

e_sl_kho = tk.Entry(fr_kho); e_sl_kho.grid(row=0, column=3)

```

```

tk.Label(fr_kho, text="Vị Trí:").grid(row=0, column=4)

```

```

e_vitri = tk.Entry(fr_kho); e_vitri.grid(row=0, column=5)

```

--- Các hàm xử lý cho Tab Kho Hàng ---

```

def load_kho():

```

```

    """Tải lại dữ liệu kho hàng và cập nhật các combobox."""

```

```

    for i in tree_kho.get_children(): tree_kho.delete(i)

```

```

    conn = get_connection()

```

```

    if conn:

```

```

        cur = conn.cursor()

```

Lấy dữ liệu từ bảng khohang, JOIN với sanpham để lấy tên SP cho dễ nhìn

```
cur.execute("""
                SELECT k.ma_sp, s.ten_sp, k.so_luong_ton, k.vitri_kho,
k.ngay_nhap_gan_nhat
                FROM khohang k
                JOIN sanpham s ON k.ma_sp = s.ma_sp
            """)
for row in cur.fetchall(): tree_kho.insert("", tk.END, values=row)
```

Tải danh sách sản phẩm cho combobox

```
cur.execute("SELECT ma_sp FROM sanpham")
cb_msp_kho['values'] = [r[0] for r in cur.fetchall()]
conn.close()
reset_form_kho()
```

def reset_form_kho():

```
    """Làm mới các ô nhập liệu trong tab kho."""
    cb_msp_kho.set("")
    e_sl_kho.delete(0, tk.END)
    e_vitri.delete(0, tk.END)
```

def on_click_kho(event):

```
    """Sự kiện khi click vào một dòng trong Treeview kho hàng."""
    sel = tree_kho.selection()
    if sel:
        val = tree_kho.item(sel[0], 'values')
        cb_msp_kho.set(val[0])
        e_sl_kho.delete(0, tk.END); e_sl_kho.insert(0, val[2])
        e_vitri.delete(0, tk.END); e_vitri.insert(0, val[3])
```

def nhap_xuat_kho(action):

```
    """
```

Hàm xử lý chung cho việc nhập và xuất kho.

```

- action: 'nhap' hoặc 'xuat'.
"""

ma_sp = cb_msp_kho.get()
vitri = e_vitri.get()
try:
    so_luong = int(e_sl_kho.get())
    if so_luong <= 0:
        messagebox.showwarning("Cảnh báo", "Số lượng phải là số dương.")
        return
except ValueError:
    messagebox.showerror("Lỗi", "Số lượng phải là một con số.")
    return

if not ma_sp:
    messagebox.showwarning("Cảnh báo", "Vui lòng chọn một sản phẩm.")
    return

conn = get_connection()
if not conn: return

cur = conn.cursor()
# Kiểm tra xem sản phẩm đã tồn tại trong kho chưa
cur.execute("SELECT so_luong_ton FROM khohang WHERE ma_sp = %s", (ma_sp,))
result = cur.fetchone()

if action == 'nhap':
    if result: # Nếu sản phẩm đã có trong kho -> CẬP NHẬT
        so_luong_moi = result[0] + so_luong
        sql = "UPDATE khohang SET so_luong_ton = %s, vitri_kho = %s, ngay_nhap_gan_nhat = CURDATE() WHERE ma_sp = %s"
        params = (so_luong_moi, vitri, ma_sp)
        msg = "Nhập kho thành công!"
    else: # Nếu sản phẩm chưa có -> THÊM MỚI

```

```

        sql = "INSERT INTO khohang (ma_sp, so_luong_ton, vitri_kho,
ngay_nhap_gan_nhat) VALUES (%s, %s, %s, CURDATE())"
        params = (ma_sp, so_luong, vitri)
        msg = "Đã thêm sản phẩm vào kho!"

elif action == 'xuat':
    if not result: # Nếu sản phẩm không có trong kho
        messagebox.showerror("Lỗi", "Sản phẩm này không có trong kho để
xuất.")
        conn.close()
        return

    so_luong_hien_tai = result[0]
    if so_luong > so_luong_hien_tai: # Kiểm tra số lượng xuất
        messagebox.showerror("Lỗi", f"Không thể xuất {so_luong} sản phẩm,
chỉ còn {so_luong_hien_tai} trong kho.")
        conn.close()
        return

    so_luong_moi = so_luong_hien_tai - so_luong
    sql = "UPDATE khohang SET so_luong_ton = %s, ngay_nhap_gan_nhat
= CURDATE() WHERE ma_sp = %s"
    params = (so_luong_moi, ma_sp)
    msg = "Xuất kho thành công!"

    conn.close() # Đóng kết nối cũ
    if execute_query(sql, params, msg):
        load_kho()

def sua_kho():
    """Cập nhật lại thông tin số lượng và vị trí cho một sản phẩm trong kho."""
    ma_sp = cb_msp_kho.get()
    vitri = e_vitri.get()
    try:

```

```

so_luong = int(e_sl_kho.get())
if so_luong < 0:
    messagebox.showwarning("Cảnh báo", "Số lượng không được là số âm.")
    return
except ValueError:
    messagebox.showerror("Lỗi", "Số lượng phải là một con số.")
    return

if not ma_sp:
    messagebox.showwarning("Cảnh báo", "Vui lòng chọn một sản phẩm.")
    return

sql = "UPDATE khohang SET so_luong_ton = %s, vitri_kho = %s, ngay_nhap_gan_nhat = CURDATE() WHERE ma_sp = %s"
params = (so_luong, vitri, ma_sp)
if execute_query(sql, params, "Cập nhật kho thành công!"):
    load_kho()

def xoa_kho():
    """Xóa một sản phẩm ra khỏi kho."""
    ma_sp = cb_msp_kho.get()
    if not ma_sp:
        messagebox.showwarning("Cảnh báo", "Vui lòng chọn sản phẩm để xóa.")
        return

    if messagebox.askyesno("Xác Nhận Xóa", f"Bạn có chắc muốn xóa sản phẩm '{ma_sp}' ra khỏi kho?"):
        if execute_query("DELETE FROM khohang WHERE ma_sp = %s", (ma_sp,), "Đã xóa sản phẩm khỏi kho!"):
            load_kho()

# --- Khung chứa các nút bấm ---
fr_btn_kho = tk.Frame(tab_kho); fr_btn_kho.pack(pady=10)

```

```

tk.Button(fr_btn_kho, text="Làm Mới", bg="#008CBA", fg="white",
command=reset_form_kho).pack(side="left", padx=5)

tk.Button(fr_btn_kho, text="Nhập Kho", bg="#4CAF50", fg="white",
command=lambda: nhap_xuat_kho('nhap')).pack(side="left", padx=5)

tk.Button(fr_btn_kho, text="Xuất Kho", bg="#f44336", fg="white",
command=lambda: nhap_xuat_kho('xuat')).pack(side="left", padx=5)

tk.Button(fr_btn_kho, text="Lưu Sửa", bg="#ff9800", fg="white",
command=sua_kho).pack(side="left", padx=5)

tk.Button(fr_btn_kho, text="Xóa Kho", bg="#A0A0A0", fg="white",
command=xoa_kho).pack(side="left", padx=5)

```

--- Bảng hiển thị dữ liệu (Treeview) ---

```

tree_kho_cols = ("MaSP", "TenSP", "SL", "ViTri", "NgayNhap")

tree_kho = ttk.Treeview(tab_kho, columns=tree_kho_cols,
show="headings")

```

```

tree_kho.heading("MaSP", text="Mã SP")
tree_kho.heading("TenSP", text="Tên Sản Phẩm")
tree_kho.heading("SL", text="Số Lượng Tồn")
tree_kho.heading("ViTri", text="Vị Trí Kho")
tree_kho.heading("NgayNhap", text="Ngày Cập Nhật")

```

```

tree_kho.column("MaSP", width=80, anchor="center")
tree_kho.column("TenSP", width=250)
tree_kho.column("SL", width=100, anchor="center")
tree_kho.column("ViTri", width=120, anchor="center")
tree_kho.column("NgayNhap", width=120, anchor="center")

```

```

tree_kho.pack(fill="both", expand=True, padx=10, pady=5)
tree_kho.bind("<<TreeviewSelect>>", on_click_kho)

```

--- Tải dữ liệu lần đầu cho tất cả các tab ---

```

load_sp()
load_ncc()
load_kh()

```

```
load_hd()
```

```
load_kho()
```

PHẦN 4: ĐIỂM KHỞI CHẠY ỨNG DỤNG

```
if __name__ == "__main__":
```

```
    # Khi chương trình được chạy, gọi hàm để tạo cửa sổ đăng nhập đầu tiên
```

```
    tao_giao_dien_dang_nhap()
```