TRƯỜNG ĐẠI HỌC AN GIANG

KHOA CÔNG NGHỆ THÔNG TIN

BÁO CÁO ĐỒ ÁN

Chuyên đề Python (COS525)

XÂY DỰNG ỨNG DỤNG QUẢN LÝ GIÁO VIÊN TRƯỜNG PHỔ THÔNG VỚI PYTHON, TKINTER VÀ MYSQL

Giảng viên hướng dẫn: Ths. Nguyễn Ngọc Minh

Sinh viên thực hiện:

DTH235701.Nguyễn Thị Trà My.DH24TH2_Nhóm02_TổTH1

DTH235727.La Thanh Pats.DH24TH2_Nhóm02_TổTH1

AN GIANG, THÁNG 11 NĂM 2025

➢ **Thành viên tham gia và phân công công việc:**

Họ và tên	MSSV	Lớp	Nhiệm vụ	Tỉ lệ tham gia
Nguyễn Thị Trà My	DTH235701	DH24TH2	Code, word,SQL	100%
La Thanh Pats	DTH235727	DH24TH2	Code, word, SQL	100%

➢ **Phân công công việc chi tiết:**

-Nguyễn Thị Trà My:

+SQL: tạo cơ sở dữ liệu

+Word: Chụp giao diện, chụp code, chụp SQL.

+Code: kết nối SQL, thêm xóa sửa, form giáo viên, lớp, môn học, xuất file excel.

-La Thanh Pats:

+SQL: tạo liên kết bảng.

+Word: Gõ nội dung.

+Code: thiết kế giao diện, tìm giáo viên theo mã, form đăng nhập, form menu.

1.ĐẶT VẤN ĐỀ

1.1 Lý do chọn đề tài

Trong bối cảnh chuyển đổi số hiện nay, công tác quản lý trong các trường học đang dần được tin học hóa nhằm nâng cao hiệu quả, tiết kiệm thời gian và hạn chế sai sót. Tuy nhiên, tại nhiều trường phổ thông, việc quản lý thông tin giáo viên vẫn còn thực hiện thủ công bằng sổ sách hoặc file Excel rời rạc. Điều này gây ra nhiều khó khăn khi cập nhật dữ liệu, tra cứu hoặc thống kê thông tin. Vì vậy, việc xây dựng phần mềm **"Quản lý giáo viên phổ thông"** là cần thiết, nhằm hỗ trợ bộ phận quản lý nhà trường lưu trữ, tra cứu và thống kê thông tin giáo viên một cách nhanh chóng, chính xác và khoa học.

1.2 Mục tiêu của đề tài

Đề tài được thực hiện nhằm: Xây dựng phần mềm hỗ trợ quản lý thông tin giáo viên trong trường phổ thông. Ứng dụng ngôn ngữ **Python** và thư viện **Tkinter** để thiết kế giao diện người dùng trực quan. Sử dụng hệ quản trị cơ sở dữ liệu **MySQL** để lưu trữ và xử lý dữ liệu. Cung cấp các chức năng cơ bản: thêm, sửa, xóa, tìm kiếm, phân công giảng dạy.

1.3 Ý nghĩa của đề tài

Giúp người dùng (cán bộ quản lý) dễ dàng theo dõi, cập nhật và tìm kiếm thông tin giáo viên. Đồng thời, giúp sinh viên lập trình củng cố kỹ năng thực hành Python, Tkinter và MySQL. Góp phần thúc đẩy việc ứng dụng công nghệ thông tin vào lĩnh vực giáo dục phổ thông.

2.TỔNG QUAN VÀ CƠ SỞ LÝ THUYẾT

2.1 Lịch sử giải quyết vấn đề

Công tác quản lý giáo viên trong các trường phổ thông từ trước đến nay chủ yếu được thực hiện thủ công bằng sổ sách, hồ sơ giấy hoặc bảng tính Excel. Mặc dù các phương pháp này có ưu điểm là đơn giản, dễ thực hiện, nhưng lại tồn tại nhiều hạn chế như: Dữ liệu phân tán, khó đồng bộ. Việc tra cứu, thống kê, cập nhật mất nhiều thời gian. Khả năng xảy ra sai sót cao khi nhập hoặc sửa thông tin.

Trong những năm gần đây, cùng với sự phát triển của công nghệ thông tin, đã xuất hiện nhiều phần mềm quản lý trong lĩnh vực giáo dục như: Phần mềm quản lý học sinh, giáo viên do Bộ GD&ĐT triển khai. Các hệ thống quản lý trường học điện tử **(School Management System)** được phát triển bằng các nền tảng web hoặc desktop.

Tuy nhiên, hầu hết các phần mềm trên thường quá phức tạp so với nhu cầu quản lý nội bộ của một trường phổ thông, hoặc không phù hợp với môi trường học tập – thực hành của sinh viên.

Vì vậy, đề tài **"Quản lý giáo viên phổ thông"** được xây dựng với mục tiêu: Thiết kế một ứng dụng đơn giản, dễ sử dụng. Phù hợp với quy mô nhỏ như một trường phổ thông hoặc lớp học mô phỏng. Giúp sinh viên rèn luyện kỹ năng lập trình **Python**, thiết kế giao diện **Tkinter**, và thao tác với cơ sở dữ liệu **MySQL**.

2.2 Phạm vi

Đề tài tập trung vào việc xây dựng phần mềm quản lý thông tin giáo viên trong trường phổ thông, với các chức năng chính: Quản lý hồ sơ giáo viên (thêm, sửa, xóa, tìm kiếm). Quản lý môn học, lớp học. Phân công giảng dạy giữa giáo viên – môn học – lớp học. Quản lý dữ liệu thông qua cơ sở dữ liệu MySQL.

Phần mềm được phát triển trên nền tảng ngôn ngữ Python, với giao diện đồ họa sử dụng thư viện Tkinter, và cơ sở dữ liệu được xây dựng trên MySQL.

Phạm vi chưa bao gồm: Quản lý học sinh, điểm số hoặc kết quả giảng dạy. Kết nối mạng hoặc chia sẻ dữ liệu trực tuyến. Phân quyền người dùng hoặc bảo mật nâng cao.

Tuy nhiên, hệ thống được thiết kế mở để có thể mở rộng trong tương lai khi cần thiết.

2.3 Tổng quan về ngôn ngữ lập trình Python

Python là ngôn ngữ lập trình bậc cao, dễ học, dễ đọc và có cú pháp đơn giản. Một số ưu điểm của Python: Có **thư viện phong phú** hỗ trợ phát triển ứng dụng

nhanh chóng. **Tương thích đa nền tảng** (Windows, macOS, Linux). Có thể kết hợp với **MySQL**, **Tkinter**, … để xây dựng phần mềm hoàn chỉnh.

Trong đề tài này, Python được sử dụng để:

- Xây dựng giao diện ứng dụng bằng **Tkinter**.
- Kết nối với **cơ sở dữ liệu MySQL** để thực hiện các thao tác thêm, sửa, xóa, tìm kiếm.

2.4 Thư viện Tkinter

Tkinter là thư viện giao diện đồ họa (GUI) được tích hợp sẵn trong Python, cho phép tạo ra các cửa sổ ứng dụng, nút bấm, nhãn, ô nhập liệu, danh sách,… Ưu điểm: Dễ sử dụng, không cần cài đặt thêm. Có thể thiết kế các form quản lý trực quan, thân thiện người dùng. Trong đồ án này, Tkinter được dùng để xây dựng các **form chức năng** như: Quản lý giáo viên; quản lý môn học; menu chính lựa chọn chức năng.

2.5 Hệ quản trị cơ sở dữ liệu MySQL

MySQL là hệ quản trị cơ sở dữ liệu quan hệ mã nguồn mở phổ biến nhất hiện nay.
MySQL hỗ trợ ngôn ngữ **SQL (Structured Query Language)** để truy vấn, quản lý và bảo mật dữ liệu.

Ưu điểm: Hoạt động ổn định, tốc độ nhanh. Dễ tích hợp với Python thông qua thư viện mysql-connector-python. Dữ liệu dễ sao lưu, chia sẻ và triển khai.

2.6 Phương pháp nghiên cứu

Trong quá trình thực hiện đề tài, các phương pháp nghiên cứu sau được áp dụng:

Phương pháp nghiên cứu lý thuyết: Tìm hiểu các tài liệu về quản lý thông tin trong trường học. Nghiên cứu cú pháp và cách sử dụng thư viện Tkinter trong Python. Nghiên cứu cấu trúc và cách thao tác cơ sở dữ liệu MySQL.

Phương pháp phân tích hệ thống: Phân tích yêu cầu chức năng của người dùng (cán bộ quản lý giáo viên). Xác định các bảng dữ liệu cần thiết và mối quan hệ giữa chúng. Thiết kế sơ đồ chức năng (DFD) và sơ đồ cơ sở dữ liệu (ERD).

Phương pháp thực nghiệm: Lập trình, kiểm thử và đánh giá kết quả trên môi trường thực tế. Thực hiện các thao tác CRUD (Create, Read, Update, Delete) để kiểm tra tính đúng đắn của hệ thống.

Phương pháp so sánh – đối chiếu: So sánh ưu điểm, hạn chế của hệ thống so với cách quản lý thủ công hoặc phần mềm có sẵn

3. KẾT QUẢ ĐẠT ĐƯỢC VÀ DEMO CODE

A. THIẾT KẾ GIAO DIỆN QUẢN LÝ GIÁO VIÊN

B. CÁC BƯỚC XÂY DỰNG ỨNG DỤNG

Để xây dựng ứng dụng quản lý sản phẩm bằng Python, Tkinter và MySQL, cần cài đặt các thư viện cần thiết, thiết kế cơ sở dữ liệu MySQL để lưu trữ thông tin sản phẩm, sử dụng Tkinter để tạo giao diện người dùng, và kết hợp Python với thư viện mysql.connector để thực hiện các thao tác như thêm, xóa, sửa, và hiển thị danh sách Giáo Viên.

1. Cài đặt các thư viện cần thiết:

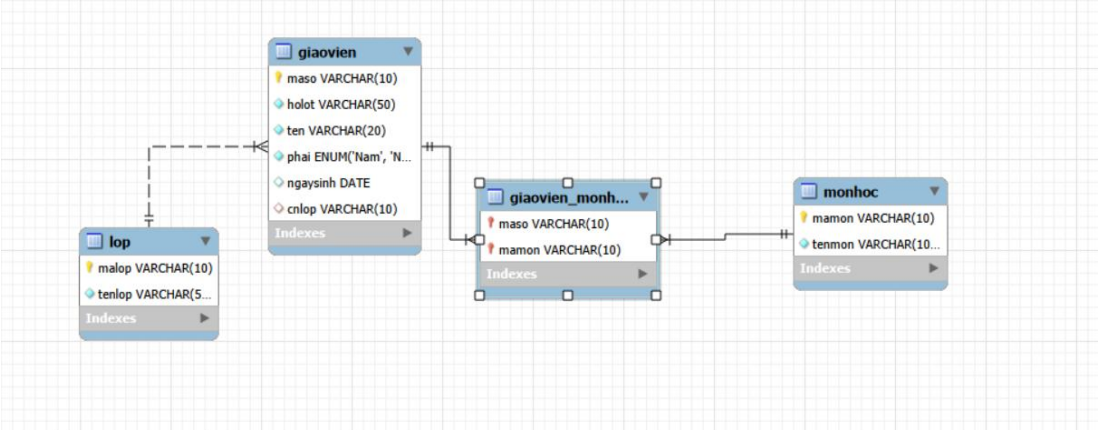
Python: Đảm bảo bạn đã cài đặt Python trên máy tính của mình.

MySQL: Cài đặt máy chủ MySQL và tạo một cơ sở dữ liệu cho ứng dụng của bạn.

mysql-connector-python: Cài đặt thư viện này để kết nối Python với cơ sở dữ liệu MySQL (pip install mysql-connector-python)

2. Thiết kế cơ sở dữ liệu MySQL

Tạo một bảng trong cơ sở dữ liệu MySQL để lưu trữ thông tin Nhân Viên như sau:



3. Xây dựng giao diện người dùng (GUI) bằng Tkinter:

Sử dụng thư viện Tkinter để tạo các cửa sổ, trường nhập liệu, nút bấm và bảng hiển thị thông tin Giáo viên.

Thiết kế giao diện cho phép người dùng thực hiện các chức năng sau:

- Thêm thông tin Giáo viên mới.
- Lưu thông tin Giáo viên
- Sửa thông tin Giáo viên.
- Hủy thông tin Giáo viên.
- Xóa thông tin Giáo viên.
- Tìm kiếm mã Giáo viên
- Xem danh sách tất cả Giáo viên,
- Xuất excel.

4. Viết mã Python kết nối cơ sở dữ liệu:

Sử dụng mysql.connector để thiết lập kết nối với cơ sở dữ liệu MySQL

Viết các hàm:

- Thêm Giáo viên: Lấy dữ liệu từ các trường nhập liệu trên giao diện và chèn vào bảng Danh sách Giáo viên .
- Lưu thông tin Giáo viên.
- Sửa thông tin Giáo viên: Sửa đổi thông tin Nhân viên trong cơ sở dữ liệu dựa trên magv.
- Hủy thông tin Giáo viên.
- Xóa thông tin Giáo viên: Xóa Giáo viên khỏi cơ sở dữ liệu dựa trên magv

- Xem danh sách tất cả Giáo viên: Truy vấn cơ sở dữ liệu để lấy tất cả Giáo viên và hiển thị chúng trong bảng trên giao diện.
- Tìm kiếm thông tin giáo viên theo magv, nhập magv trong ô tìm kiếm, hiển thị thông tin giáo viên cần tìm.
- Xuất excel: khi nhấn nút "xuất excel" người dùng địa chỉ lưu file excel và lưu, khi đó file danh sách giáo viên sẽ được lưu và thông báo đã lưu thành công.

5. Xây dựng giao diện Tkinter

Liên kết các nút bấm trên giao diện Tkinter với các hàm Python tương ứng để xử lý dữ liệu và tương tác với cơ sở dữ liệu.

Các Chức Năng:

- Thêm Giáo viên: Khi người dùng nhập tên, giá, số lượng và nhấn nút "Thêm", mã Python sẽ lấy các giá trị này và thêm một bản ghi mới vào bảng giaovien trong MySQL.
- Xóa Giáo viên: Khi chọn chức năng xóa thực hiện người dùng phải xác nhận có muốn xóa hay không, khi người dùng chọn có sẽ xóa thông tin dòng được chọn.
- Sửa: Cập nhật dòng được chọn thông báo sửa các thay đổi.
- Lưu: Lưu các thay đổi vào cơ sở dữ liệu, thông báo đã lưu.
- Xem Giáo viên: Khi người dùng mở cửa sổ ứng dụng, mã Python sẽ truy vấn cơ sở dữ liệu, lấy tất thông Giáo viên hiển thị chúng dưới dạng một danh sách hoặc bảng trên giao diện Tkinter.
- Tìm kiếm: sau khi nhập magv, nhấn nút "Tìm", mã Python sẽ lấy các thông tin của giáo viên mà người dùng cần tìm và hiển thị ra màn hình.

C. HƯỚNG DẪN CÀI ĐẶT & CODE ỨNG DỤNG

Ứng dụng sẽ có:

- Giao diện đăng nhập
- Giao diện menu: quản lý giáo viên, quản lý lớp, quản lý môn học. Chi tiết như sau:
    - Giao diện nhập thông tin giáo viên (Mã số, Họ tên, Tên, Giới tính, Ngày sinh, Môn giảng dạy, Chủ nhiệm).
    - Bảng danh sách Giáo .
    - Chức năng CRUD (Thêm, Sửa, Xóa, Lưu, Hủy).
    - Dữ liệu lưu trực tiếp vào MySQL Database (không mất khi tắt ứng dụng).
    - Giao diện Tkinter thiết kế chuẩn như hình minh họa ở trên.

- Giao diện quản lý môn học và lớp học tương tự nhưng chỉ thông tin chỉ có mã môn, tên môn và mã lớp, tên lớp.

1.Cài đặt môi trường

Trước tiên cài đặt thư viện cần thiết:

```
pip install mysql-connector-python tkcalendar
```

Giải thích:

- mysql-connector-python: dùng để kết nối Python với MySQL.
- tkcalendar: để chọn ngày sinh bằng DateEntry.

2. Chuẩn bị CSDL MySQL

Tạo database và bảng bằng MySQL Workbench 8.0 CE

```
CREATE DATABASE IF NOT EXISTS qlgv;
USE qlgv;
-- Bảng môn học
CREATE TABLE IF NOT EXISTS monhoc (
    mamon VARCHAR(10) PRIMARY KEY,
    tenmon VARCHAR(100) NOT NULL
);
-- Bảng lớp
CREATE TABLE IF NOT EXISTS lop (
    malop VARCHAR(10) PRIMARY KEY,
    tenlop VARCHAR(50) NOT NULL
);
--Bảng giáo viên
CREATE TABLE IF NOT EXISTS giaovien (
    maso VARCHAR(10) PRIMARY KEY,
    holot VARCHAR(50) NOT NULL,
    ten VARCHAR(20) NOT NULL,
    phai ENUM('Nam','Nữ') NOT NULL,
    ngaysinh DATE,
```

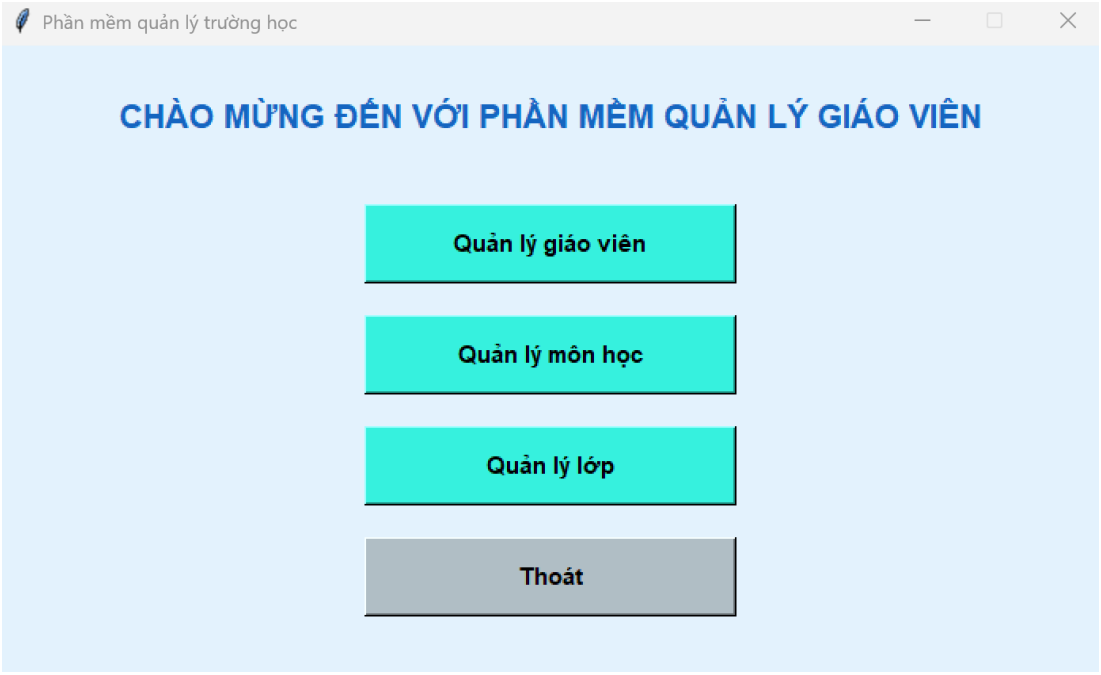
```
    cnlop VARCHAR(10),
    FOREIGN KEY (cnlop) REFERENCES lop(malop) ON DELETE SET NULL
);



-- Bảng liên kết giáo viên - môn học
CREATE TABLE IF NOT EXISTS giaovien_monhoc (
    maso VARCHAR(10),
    mamon VARCHAR(10),
    PRIMARY KEY (maso, mamon),
    FOREIGN KEY (maso) REFERENCES giaovien(maso) ON DELETE CASCADE,
    FOREIGN KEY (mamon) REFERENCES monhoc(mamon) ON DELETE CASCADE
);
```

3. Viết code ứng dụng với Tkinter + MySQL

- Căn giữa:

```
def center_window(win, w=700,
h=500):
    ws = win.winfo_screenwidth()
    hs = win.winfo_screenheight()
    x = (ws // 2) - (w // 2)
    y = (hs // 2) - (h // 2)
    win.geometry(f'{w}x{h}+{x}+{y}')
    win.resizable(False, False)
```

- Form đăng nhập và cửa sổ chính



```
import tkinter as tk
from tkinter import messagebox
from teacher_ui import open_teacher_management
from subject_ui import open_subject_management
from class_ui import open_class_management
from utils import center_window

def check_login():
    user = entry_user.get()
    pw = entry_pass.get()
    if user == "gv" and pw == "2025":
        login_win.destroy()
        open_main_menu()
    else:
        messagebox.showerror("Lỗi", "Sai tên đăng nhập hoặc mật khẩu!")

def open_main_menu():
    global root
    root = tk.Tk()
    root.title("Phần mềm quản lý trường học")
    center_window(root, 700, 400)
    root.configure(bg="#E3F2FD")

    tk.Label(root, text="CHÀO MỪNG ĐẾN VỚI PHẦN MỀM QUẢN LÝ GIÁO VIÊN",
             font=("Arial", 16, "bold"), bg="#E3F2FD",
fg="#1565C0").pack(pady=30)

    def make_button(text, cmd, bg="#36F1DE"):
        return tk.Button(root, text=text, width=25, height=2,
                         command=lambda: cmd(root, open_main_menu),
                         bg=bg, fg="black", font=("Arial", 11, "bold"),
                         relief="raised", cursor="hand2",
activebackground="#4DB6AC")

    make_button("Quản lý giáo viên",
open_teacher_management).pack(pady=10)
    make_button("Quản lý môn học",
open_subject_management).pack(pady=10)
    make_button("Quản lý lớp", open_class_management).pack(pady=10)
    tk.Button(root, text="Thoát", width=25, height=2,
command=root.destroy,
              bg="#B0BEC5", fg="black", font=("Arial", 11,
"bold")).pack(pady=10)

   # root.mainloop()

# ===== Đăng nhập =====
login_win = tk.Tk()
login_win.title("Đăng nhập hệ thống")
center_window(login_win, 700, 350)
login_win.configure(bg="#E3F2FD")
```

```
lbl_title = tk.Label(login_win, text="ĐĂNG NHẬP HỆ THỐNG",
                     font=("Arial", 20, "bold"), bg="#E3F2FD",
fg="#1565C0")
lbl_title.pack(pady=(20, 10))

frame_login = tk.Frame(login_win, bg="#FFFFFF", padx=20, pady=20, bd=2,
relief="groove")
frame_login.pack(pady=10)

tk.Label(frame_login, text="Tên đăng nhập:", bg="#FFFFFF", fg="#0D47A1",
         font=("Arial", 12)).grid(row=0, column=0, pady=8, sticky="e")
tk.Label(frame_login, text="Mật khẩu:", bg="#FFFFFF", fg="#0D47A1",
         font=("Arial", 12)).grid(row=1, column=0, pady=8, sticky="e")

entry_user = tk.Entry(frame_login, width=30, font=("Arial", 12))
entry_user.grid(row=0, column=1, pady=8)
entry_pass = tk.Entry(frame_login, show="*", width=30, font=("Arial",
12))
entry_pass.grid(row=1, column=1, pady=8)

frame_buttons = tk.Frame(login_win, bg="#E3F2FD")
frame_buttons.pack(pady=20)

btn_login = tk.Button(frame_buttons, text="Đăng nhập", width=14,
height=1,
                      bg="#36F1DE", fg="black",
activebackground="#4DB6AC",
                      font=("Arial", 11, "bold"), cursor="hand2",
relief="raised",
                      command=check_login)
btn_login.pack(side="left", padx=12)

btn_exit = tk.Button(frame_buttons, text="Thoát", width=14, height=1,
                     bg="#B0BEC5", fg="black",
activebackground="#CFD8DC",
                     font=("Arial", 11, "bold"), cursor="hand2",
relief="raised",
                     command=login_win.quit)
btn_exit.pack(side="left", padx=12)

login_win.mainloop()
```

- Form Quản lý giáo viên


```
import tkinter as tk
from tkinter import ttk, messagebox, filedialog
from tkcalendar import DateEntry
from utils import center_window
import mysql.connector
import re
import pandas as pd

def connect_db():
    return mysql.connector.connect(
        host="localhost",
        user="root",
        password="1234",
        database="qlgv"
    )

def open_teacher_management(root, open_main_menu):
    root.destroy()
    win = tk.Tk()
    win.title("Quản lý giáo viên")
    center_window(win, 1000, 600)
    win.configure(bg="#E3F2FD")
    win.resizable(False, False)

    # ===== Tiêu đề =====
    tk.Label(win, text="QUẢN LÝ GIÁO VIÊN THPT",
             font=("Arial", 18, "bold"), bg="#E3F2FD",
fg="#1565C0").pack(pady=15)
```

```
    # ===== Frame thông tin =====
    frame_info = tk.LabelFrame(win, text="Thông tin giáo viên",
bg="#FFFFFF", fg="#0D47A1",
                               font=("Arial", 12, "bold"), padx=15,
pady=10)
    frame_info.pack(padx=15, pady=10, fill="x")

    # Entry và Combobox/Listbox
    entry_maso = tk.Entry(frame_info, width=12)
    entry_holot = tk.Entry(frame_info, width=25)
    entry_ten = tk.Entry(frame_info, width=15)
    date_entry = DateEntry(frame_info, width=12, date_pattern="yyyy-
mm-dd", selectbackground="#1976D2",
    selectforeground="white")
    gender_var = tk.StringVar(value="Nam")
    gvcn_var = tk.StringVar(value="Không")
    cnlop_combo = ttk.Combobox(frame_info, width=15)
    list_monhoc = tk.Listbox(frame_info, selectmode="multiple",
width=30, height=4, exportselection=False)

    # ===== Hàm bật/tắt CN lớp =====
    def toggle_cnlop():
        if gvcn_var.get() == "Có":
            cnlop_combo.config(state="readonly")
        else:
            cnlop_combo.set("")
            cnlop_combo.config(state="disabled")

    # ===== Gắn nhãn và grid =====
    # Hàng 0
    tk.Label(frame_info, text="Mã số:", bg="#FFFFFF").grid(row=0,
column=0, padx=5, pady=5, sticky="w")
    entry_maso.grid(row=0, column=1, padx=5, pady=5, sticky="w")
    tk.Label(frame_info, text="Môn giảng dạy:",
bg="#FFFFFF").grid(row=0, column=4, padx=5, pady=5, sticky="w")
    tk.Label(frame_info, text="GVCN", bg="#FFFFFF").grid(row=0,
column=2, padx=5, pady=5, sticky="w")
    tk.Radiobutton(frame_info, text="Có", variable=gvcn_var,
value="Có", bg="#FFFFFF", command=toggle_cnlop).grid(row=0, column=3,
sticky="w")
    tk.Radiobutton(frame_info, text="Không", variable=gvcn_var,
value="Không", bg="#FFFFFF", command=toggle_cnlop).grid(row=0,
column=3, padx=60, sticky="w")
    tk.Label(frame_info, text="Chủ nhiệm lớp",
bg="#FFFFFF").grid(row=0, column=5, padx=5, pady=5, sticky="w")
    cnlop_combo.grid(row=0, column=6, padx=5, pady=5, sticky="w")

    # Hàng 1
```

```
    list_monhoc.grid(row=1, column=4, rowspan=2, padx=5, pady=5,
sticky="w")
    tk.Label(frame_info, text="Họ lót:", bg="#FFFFFF").grid(row=1,
column=0, padx=5, pady=5, sticky="w")
    entry_holot.grid(row=1, column=1, padx=5, pady=5, sticky="w")
    tk.Label(frame_info, text="Tên:", bg="#FFFFFF").grid(row=1,
column=2, padx=5, pady=5, sticky="w")
    entry_ten.grid(row=1, column=3, padx=5, pady=5, sticky="w")
    tk.Label(frame_info, text="Tìm kiếm:", bg="#FFFFFF").grid(row=1,
column=5, padx=5, pady=5, sticky="w")
    entry_search = tk.Entry(frame_info, width=15)
    entry_search.grid(row=1, column=6, padx=5, pady=5, sticky="w")

    # Hàng 2
    tk.Label(frame_info, text="Phái:", bg="#FFFFFF").grid(row=2,
column=0, padx=5, pady=5, sticky="w")
    tk.Radiobutton(frame_info, text="Nam", variable=gender_var,
value="Nam", bg="#FFFFFF").grid(row=2, column=1, sticky="w")
    tk.Radiobutton(frame_info, text="Nữ", variable=gender_var,
value="Nữ", bg="#FFFFFF").grid(row=2, column=1, padx=60, sticky="w")
    tk.Label(frame_info, text="Ngày sinh:", bg="#FFFFFF").grid(row=2,
column=2, padx=5, pady=5, sticky="w")
    date_entry.grid(row=2, column=3, padx=5, pady=5, sticky="w")

    toggle_cnlop()

    # ===== Load môn và lớp =====
    def load_monhoc():
        conn = connect_db()
        cur = conn.cursor()
        cur.execute("SELECT mamon, tenmon FROM monhoc ORDER BY
mamon")
        rows = cur.fetchall()
        list_monhoc.delete(0, tk.END)
        global mon_dict
        mon_dict = {tenmon: mamon for mamon, tenmon in rows}
        for tenmon in mon_dict.keys():
            list_monhoc.insert(tk.END, tenmon)
        conn.close()

    def load_lop():
        conn = connect_db()
        cur = conn.cursor()
        cur.execute("SELECT malop FROM lop")
        cnlop_combo['values'] = [row[0] for row in cur.fetchall()]
        conn.close()

    load_monhoc()
    load_lop()
```

```
    # ===== Frame danh sách =====
    frame_list = tk.LabelFrame(win, text="Danh sách giáo viên",
bg="#FFFFFF", fg="#0D47A1",
                               font=("Arial", 12, "bold"), padx=5,
pady=5)
    frame_list.pack(padx=15, pady=10, fill="both", expand=True)

    columns =
("maso","holot","ten","phai","ngaysinh","monhoc","cnlop")
    tree = ttk.Treeview(frame_list, columns=columns, show="headings")
    for col in columns:
        tree.heading(col, text=col.capitalize())
        tree.column(col, width=120, anchor="center")
    tree.pack(fill="both", expand=True)

    tree.heading("maso", text="Mã giáo viên")
    tree.heading("holot", text="Họ lót")
    tree.heading("ten", text="Tên")
    tree.heading("phai", text="Phái")
    tree.heading("ngaysinh", text="Ngày sinh")
    tree.heading("monhoc", text="Môn học")
    tree.heading("cnlop", text="Chủ nhiệm lớp")


    # ===== Frame nút =====
    frame_btn = tk.Frame(win, bg="#E3F2FD")
    frame_btn.pack(pady=10)

    def make_btn(text, cmd=None, bg="#36F1DE"):
        return tk.Button(frame_btn, text=text, width=10, bg=bg,
fg="black",
                         font=("Arial", 10, "bold"), cursor="hand2",
                         relief="raised", activebackground="#4DB6AC",
command=cmd)

    # ===== Hàm CRUD =====
    def clear_input():
        entry_maso.delete(0, tk.END)
        entry_holot.delete(0, tk.END)
        entry_ten.delete(0, tk.END)
        gender_var.set("Nam")
        date_entry.set_date("2000-01-01")
        gvcn_var.set("Không")
        cnlop_combo.set("")
        cnlop_combo.config(state="disabled")
        list_monhoc.selection_clear(0, tk.END)

    def load_data():
```

```
        for item in tree.get_children():
            tree.delete(item)
        conn = connect_db()
        cur = conn.cursor()
        cur.execute("""
            SELECT g.maso, g.holot, g.ten, g.phai, g.ngaysinh,
                   GROUP_CONCAT(m.tenmon SEPARATOR ', ') AS monhoc,
                   g.cnlop
            FROM giaovien g
            LEFT JOIN giaovien_monhoc gm ON g.maso = gm.maso
            LEFT JOIN monhoc m ON gm.mamon = m.mamon
            GROUP BY g.maso
            ORDER BY g.maso
        """)
        for row in cur.fetchall():
            tree.insert("", tk.END, values=row)
        conn.close()
    def tim_kiem():
        maso = entry_search.get().strip()
        if not maso:
            messagebox.showwarning("Thiếu dữ liệu", "Hãy nhập mã giáo
viên để tìm!")
            return

        conn = connect_db()
        cur = conn.cursor()
        cur.execute("""
            SELECT g.maso, g.holot, g.ten, g.phai, g.ngaysinh,
                GROUP_CONCAT(m.tenmon SEPARATOR ', ') AS monhoc,
                g.cnlop
            FROM giaovien g
            LEFT JOIN giaovien_monhoc gm ON g.maso = gm.maso
            LEFT JOIN monhoc m ON gm.mamon = m.mamon
            WHERE g.maso = %s
            GROUP BY g.maso
        """, (maso,))
        row = cur.fetchone()
        conn.close()

        for item in tree.get_children():
            tree.delete(item)

        if row:
            tree.insert("", tk.END, values=row)
        else:
            messagebox.showerror("Không tìm thấy", f"Không có giáo
viên mã: {maso}")
```

```
    def them():
        maso = entry_maso.get().strip()
        holot = entry_holot.get().strip()
        ten = entry_ten.get().strip()
        if not maso or not holot or not ten:
            messagebox.showwarning("Thiếu dữ liệu","Nhập đầy đủ thông
tin!")
            return
        if not maso.startswith("GV"):
            messagebox.showwarning("Lỗi","Mã giáo viên phải bắt đầu
GV")
            return
        if not re.fullmatch(r"[A-Za-zÀ-ỹ\s]+", holot):
            messagebox.showwarning("Lỗi","Họ lót chỉ chứa chữ và
khoảng trắng")
            return
        if not re.fullmatch(r"[A-Za-zÀ-ỹ\s]+", ten):
            messagebox.showwarning("Lỗi","Tên chỉ chứa chữ và khoảng
trắng")
            return

        phai = gender_var.get()
        ngaysinh = date_entry.get()
        cnlop = cnlop_combo.get() if gvcn_var.get()=="Có" else None
        selected_monhoc = [mon_dict[list_monhoc.get(i)] for i in
list_monhoc.curselection()]

        conn = connect_db()
        cur = conn.cursor()

        cur.execute("SELECT * FROM giaovien WHERE maso=%s", (maso,))
        if cur.fetchone():
            messagebox.showerror("Lỗi", f"Mã giáo viên {maso} đã tồn
tại!")
            conn.close()
            return
        if cnlop:
            cur.execute("SELECT COUNT(*) FROM giaovien WHERE
cnlop=%s", (cnlop,))
            if cur.fetchone()[0] > 0:
                messagebox.showwarning("Lỗi", f"Lớp {cnlop} đã có
giáo viên chủ nhiệm")
                conn.close()
                return
        try:
            cur.execute("INSERT INTO giaovien(maso, holot, ten, phai,
ngaysinh, cnlop) VALUES (%s,%s,%s,%s,%s,%s)",
                        (maso, holot, ten, phai, ngaysinh, cnlop))
            for mamon in selected_monhoc:
```

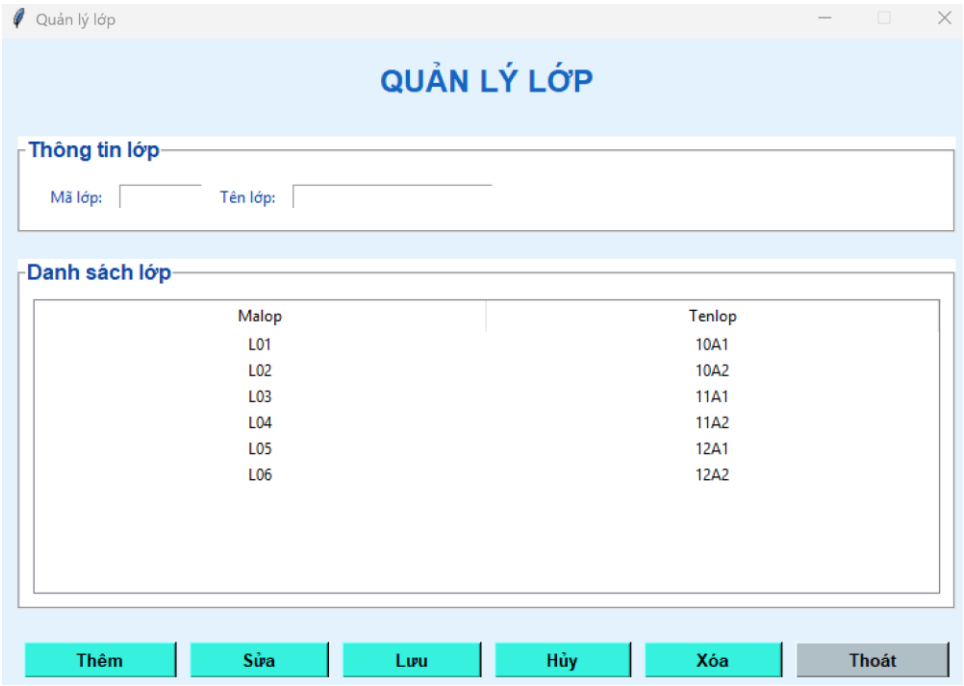
```
                cur.execute("INSERT INTO giaovien_monhoc(maso, mamon)
VALUES (%s,%s)", (maso, mamon))
            conn.commit()
            load_data()
            clear_input()
        except Exception as e:
            messagebox.showerror("Lỗi", str(e))
        conn.close()

    def xoa():
        selected = tree.selection()
        if not selected:
            messagebox.showwarning("Chưa chọn","Hãy chọn giáo viên để
xoá!")
            return

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

        if not messagebox.askyesno("Xác nhận", f"Bạn có chắc muốn xoá
giáo viên {maso}?"):
            return

        conn = connect_db()
        cur = conn.cursor()

        # Kiểm tra giáo viên là chủ nhiệm lớp
        cur.execute("SELECT COUNT(*) FROM lop WHERE malop IN (SELECT
cnlop FROM giaovien WHERE maso=%s)", (maso,))
        if cur.fetchone()[0] > 0:
            messagebox.showwarning("Lỗi", f"Giáo viên {maso} đang là
chủ nhiệm lớp, không thể xoá!")
            conn.close()
            return

        try:
            cur.execute("DELETE FROM giaovien WHERE maso=%s",
(maso,))
            conn.commit()
            tree.delete(selected)
        except Exception as e:
            messagebox.showerror("Lỗi", str(e))
        conn.close()



    def on_select(event):
        selected = tree.selection()
        if not selected: return
        values = tree.item(selected)["values"]
```

```
        entry_maso.delete(0, tk.END); entry_maso.insert(0, values[0])
        entry_holot.delete(0, tk.END); entry_holot.insert(0,
values[1])
        entry_ten.delete(0, tk.END); entry_ten.insert(0, values[2])
        gender_var.set(values[3])
        date_entry.set_date(values[4])
        cnlop_combo.set(values[6] if values[6] else "")
        if values[6]:
            gvcn_var.set("Có")
            cnlop_combo.config(state="readonly")
        else:
            gvcn_var.set("Không")
            cnlop_combo.config(state="disabled")
        monchon = values[5].split(", ") if values[5] else []
        list_monhoc.selection_clear(0, tk.END)
        for i, tenmon in enumerate(list_monhoc.get(0, tk.END)):
            if tenmon in monchon:
                list_monhoc.selection_set(i)

    def sua():
        selected = tree.selection()
        if not selected:
            messagebox.showwarning("Chưa chọn","Hãy chọn giáo viên để
sửa!")
            return
        messagebox.showinfo("Sửa","Chỉnh sửa thông tin rồi nhấn
LƯU.")

    def luu():
        maso = entry_maso.get().strip()
        holot = entry_holot.get().strip()
        ten = entry_ten.get().strip()
        if not re.fullmatch(r"[A-Za-zÀ-ỹ\s]+", holot) or not
re.fullmatch(r"[A-Za-zÀ-ỹ\s]+", ten):
            messagebox.showwarning("Lỗi","Họ tên chỉ chứa chữ và
khoảng trắng")
            return

        phai = gender_var.get()
        ngaysinh = date_entry.get()
        cnlop = cnlop_combo.get() if gvcn_var.get()=="Có" else None
        selected_monhoc = [mon_dict[list_monhoc.get(i)] for i in
list_monhoc.curselection()]

        conn = connect_db()
        cur = conn.cursor()
        if cnlop:
            cur.execute("SELECT COUNT(*) FROM giaovien WHERE cnlop=%s
AND maso<>%s", (cnlop, maso))
```

```
            if cur.fetchone()[0] > 0:
                messagebox.showwarning("Lỗi", f"Lớp {cnlop} đã có
giáo viên chủ nhiệm")
                conn.close()
                return
        try:
            cur.execute("UPDATE giaovien SET holot=%s, ten=%s,
phai=%s, ngaysinh=%s, cnlop=%s WHERE maso=%s",
                        (holot, ten, phai, ngaysinh, cnlop, maso))
            cur.execute("DELETE FROM giaovien_monhoc WHERE maso=%s",
(maso,))
            for mamon in selected_monhoc:
                cur.execute("INSERT INTO giaovien_monhoc(maso, mamon)
VALUES (%s,%s)", (maso, mamon))
            conn.commit()
            load_data()
            clear_input()
        except Exception as e:
            messagebox.showerror("Lỗi", str(e))
        conn.close()

    def huy():
        clear_input()

    def export_to_excel():
        conn = connect_db()
        query = """
            SELECT g.maso, g.holot, g.ten, g.phai, g.ngaysinh,
                GROUP_CONCAT(m.tenmon SEPARATOR ', ') AS monhoc,
                g.cnlop
            FROM giaovien g
            LEFT JOIN giaovien_monhoc gm ON g.maso = gm.maso
            LEFT JOIN monhoc m ON gm.mamon = m.mamon
            GROUP BY g.maso
            ORDER BY g.maso
        """
        df = pd.read_sql(query, conn)
        conn.close()

        # Mở hộp thoại chọn vị trí lưu file
        file_path =
filedialog.asksaveasfilename(defaultextension=".xlsx",
                                                filetypes=[("Excel
files", "*.xlsx")],
                                                title="Lưu file
Excel")
        if file_path:
            df.to_excel(file_path, index=False)
```

```
            messagebox.showinfo("Thành công", "Xuất dữ liệu ra Excel
thành công!")


    # ===== Tạo nút =====
    buttons = [("Tìm", tim_kiem), ("Thêm", them), ("Sửa", sua),
("Lưu", luu),
           ("Hủy", huy), ("Xóa", xoa),
           ("Thoát", lambda: (win.destroy(), open_main_menu())),
("Xuất Excel", export_to_excel)]
    for idx, (text, cmd) in enumerate(buttons):
        if text == "Thoát":
            color = "#B0BEC5"
        elif text == "Tìm":
            color = "#6CF2E5"
        elif text == "Xuất Excel":
            color = "#4CAF50"
        else:
            color = "#36F1DE"
        make_btn(text, cmd, bg=color).grid(row=0, column=idx, padx=5,
pady=5)

    tree.bind("<<TreeviewSelect>>", on_select)
    load_data()
    win.mainloop()
```

- form Quản lý lớp



```
import tkinter as tk
from tkinter import ttk, messagebox
from utils import center_window
import mysql.connector
import re

def connect_db():
    return mysql.connector.connect(
        host="localhost",
        user="root",
        password="1234",
        database="qlgv"
    )

def open_class_management(root, open_main_menu):
    root.destroy()
    win = tk.Tk()
    win.title("Quản lý lớp")
    center_window(win, 750, 500)
    win.configure(bg="#E3F2FD")
    win.resizable(False, False)

    tk.Label(win, text="QUẢN LÝ LỚP", font=("Arial", 18, "bold"),
             bg="#E3F2FD", fg="#1565C0").pack(pady=15)

    # ===== Frame thông tin =====
    frame_info = tk.LabelFrame(win, text="Thông tin lớp",
bg="#FFFFFF", fg="#0D47A1",
                               font=("Arial", 12, "bold"), padx=15,
pady=10)
    frame_info.pack(padx=15, pady=10, fill="x")

    tk.Label(frame_info, text="Mã lớp:", bg="#FFFFFF",
fg="#0D47A1").grid(row=0, column=0, padx=5, pady=5, sticky="w")
    entry_malop = tk.Entry(frame_info, width=10)
    entry_malop.grid(row=0, column=1, padx=5, pady=5, sticky="w")

    tk.Label(frame_info, text="Tên lớp:", bg="#FFFFFF",
fg="#0D47A1").grid(row=0, column=2, padx=5, pady=5, sticky="w")
    entry_tenlop = tk.Entry(frame_info, width=25)
    entry_tenlop.grid(row=0, column=3, padx=5, pady=5, sticky="w")

    # ===== Frame danh sách =====
    frame_list = tk.LabelFrame(win, text="Danh sách lớp",
bg="#FFFFFF", fg="#0D47A1",
                               font=("Arial", 12, "bold"), padx=5,
pady=5)
    frame_list.pack(padx=15, pady=10, fill="both", expand=True)
```

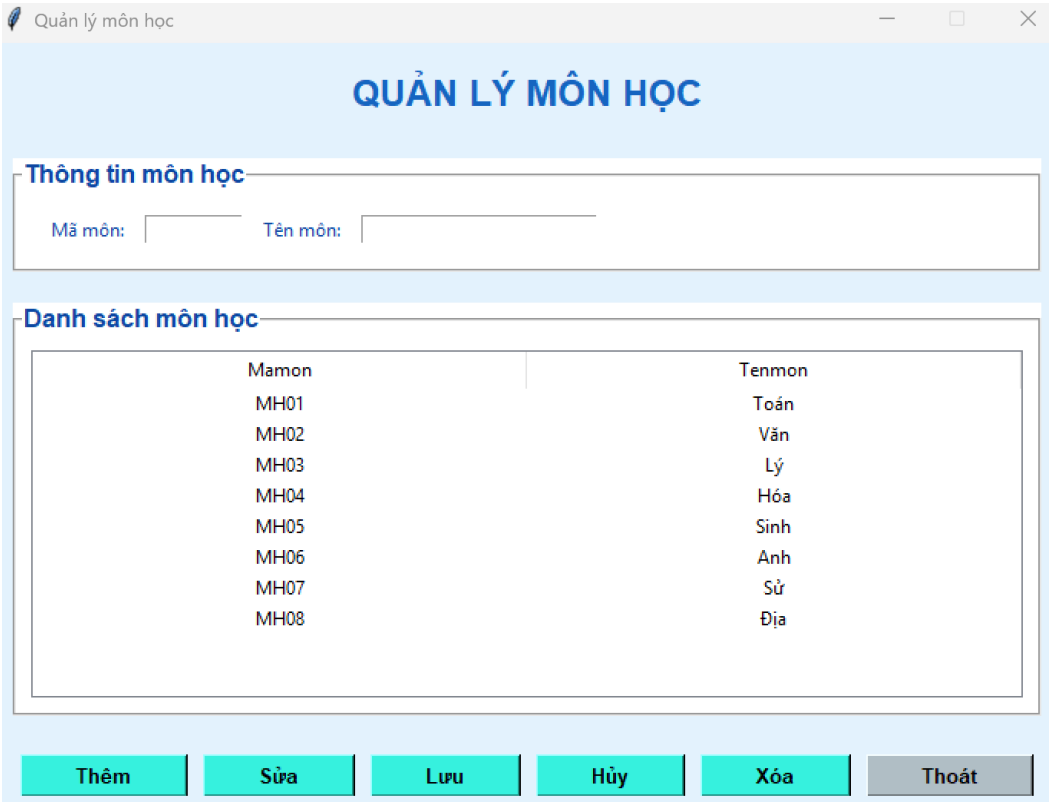
```
    columns = ("malop", "tenlop")
    tree = ttk.Treeview(frame_list, columns=columns, show="headings",
height=10)
    for col in columns:
        tree.heading(col, text=col.capitalize())
        tree.column(col, width=150, anchor="center")
    tree.pack(padx=5, pady=5, fill="both", expand=True)

    # ===== Frame nút =====
    frame_btn = tk.Frame(win, bg="#E3F2FD")
    frame_btn.pack(padx=15, pady=10, fill="x")
    for i in range(6):
        frame_btn.columnconfigure(i, weight=1)

    def make_btn(text, cmd=None, col=0, bg="#36F1DE"):
        btn = tk.Button(frame_btn, text=text, bg=bg, fg="black",
                        font=("Arial", 10, "bold"), cursor="hand2",
                        relief="raised", activebackground="#4DB6AC",
command=cmd)
        btn.grid(row=0, column=col, padx=5, pady=5, sticky="ew")
        return btn

    # ===== Hàm CRUD =====
    def clear_input():
        entry_malop.delete(0, tk.END)
        entry_tenlop.delete(0, tk.END)

    def load_data():
        for item in tree.get_children():
            tree.delete(item)
        conn = connect_db()
        cur = conn.cursor()
        cur.execute("SELECT * FROM lop ORDER BY malop")
        for row in cur.fetchall():
            tree.insert("", tk.END, values=row)
        conn.close()

    def them():
        malop = entry_malop.get().strip()
        tenlop = entry_tenlop.get().strip()

        # Kiểm tra dữ liệu trống
        if not malop or not tenlop:
            messagebox.showwarning("Thiếu dữ liệu", "Nhập đầy đủ
thông tin!")
            return

        # Kiểm tra mã lớp
        if not malop.startswith("L"):
```

```
            messagebox.showwarning("Lỗi", "Mã lớp phải bắt đầu bằng
L")
            return

        # Kiểm tra tên lớp hợp lệ
        if re.search(r'\d{3,}', tenlop):
            messagebox.showwarning("Lỗi", "Tên lớp không hợp lệ")
            return

        conn = connect_db()
        cur = conn.cursor()

        # ---------- Kiểm tra mã lớp trùng ----------
        cur.execute("SELECT * FROM lop WHERE malop=%s", (malop,))
        if cur.fetchone():
            messagebox.showerror("Lỗi", f"Mã lớp {malop} đã tồn
tại!")
            conn.close()
            return

        # ---------- Kiểm tra tên lớp trùng ----------
        cur.execute("SELECT * FROM lop WHERE tenlop=%s", (tenlop,))
        if cur.fetchone():
            messagebox.showerror("Lỗi", f"Tên lớp '{tenlop}' đã tồn
tại!")
            conn.close()
            return

        # Thêm dữ liệu nếu hợp lệ
        try:
            cur.execute("INSERT INTO lop(malop, tenlop) VALUES
(%s,%s)", (malop, tenlop))
            conn.commit()
            load_data()
            clear_input()
        except Exception as e:
            messagebox.showerror("Lỗi", str(e))
        conn.close()

    def xoa():
        selected = tree.selection()
        if not selected:
            messagebox.showwarning("Chưa chọn", "Hãy chọn lớp để
xoá!")
            return

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

```
        if not messagebox.askyesno("Xác nhận", f"Bạn có chắc muốn xoá
lớp {malop}?"):
            return

        conn = connect_db()
        cur = conn.cursor()

        # Kiểm tra lớp có giáo viên chủ nhiệm không
        cur.execute("SELECT COUNT(*) FROM giaovien WHERE cnlop=%s",
(malop,))
        if cur.fetchone()[0] > 0:
            messagebox.showwarning("Lỗi", f"Lớp {malop} đang có giáo
viên chủ nhiệm, không thể xoá!")
            conn.close()
            return

        try:
            cur.execute("DELETE FROM lop WHERE malop=%s", (malop,))
            conn.commit()
            tree.delete(selected)
        except Exception as e:
            messagebox.showerror("Lỗi", str(e))
        conn.close()



    def sua():
        selected = tree.selection()
        if not selected:
            messagebox.showwarning("Chưa chọn", "Chọn lớp để sửa!")
            return
        messagebox.showinfo("Sửa", "Chỉnh sửa thông tin rồi nhấn
Lưu.")

    def luu():
        malop = entry_malop.get().strip()
        tenlop = entry_tenlop.get().strip()
        conn = connect_db()
        cur = conn.cursor()
        try:
            cur.execute("UPDATE lop SET tenlop=%s WHERE malop=%s",
(tenlop, malop))
            conn.commit()
            load_data()
            clear_input()
        except Exception as e:
            messagebox.showerror("Lỗi", str(e))
        conn.close()
```

```
    def huy():
        clear_input()

    def on_select(event):
        selected = tree.selection()
        if not selected: return
        values = tree.item(selected)["values"]
        entry_malop.delete(0, tk.END); entry_malop.insert(0,
values[0])
        entry_tenlop.delete(0, tk.END); entry_tenlop.insert(0,
values[1])

    # ===== Tạo nút =====
    buttons = [
        ("Thêm", them), ("Sửa", sua), ("Lưu", luu),
        ("Hủy", huy), ("Xóa", xoa), ("Thoát", lambda: (win.destroy(),
open_main_menu()))
    ]
    for idx, (text, cmd) in enumerate(buttons):
        if text == "Thoát":
            make_btn(text, cmd, bg="#B0BEC5").grid(row=0, column=idx,
padx=5, pady=5)
        else:
            make_btn(text, cmd).grid(row=0, column=idx, padx=5,
pady=5)

    tree.bind("<<TreeviewSelect>>", on_select)
    load_data()
    win.mainloop()
```

- form Quản lý môn học




```
import tkinter as tk
from tkinter import ttk, messagebox
import mysql.connector
from utils import center_window
import re

def connect_db():
    return mysql.connector.connect(
        host="localhost",
        user="root",
        password="1234",
        database="qlgv"
    )

def open_subject_management(root, open_main_menu):
    root.destroy()
    win = tk.Tk()
    win.title("Quản lý môn học")
    center_window(win, 700, 500)
    win.configure(bg="#E3F2FD")
    win.resizable(False, False)

    tk.Label(win, text="QUẢN LÝ MÔN HỌC",
             font=("Arial", 18, "bold"), bg="#E3F2FD",
fg="#1565C0").pack(pady=15)
```

```
    # ===== Frame thông tin =====
    frame_info = tk.LabelFrame(win, text="Thông tin môn học",
bg="#FFFFFF", fg="#0D47A1",
                               font=("Arial", 12, "bold"), padx=15,
pady=10)
    frame_info.pack(padx=15, pady=10, fill="x")

    tk.Label(frame_info, text="Mã môn:", bg="#FFFFFF",
fg="#0D47A1").grid(row=0, column=0, padx=5, pady=5, sticky="w")
    entry_mamon = tk.Entry(frame_info, width=10)
    entry_mamon.grid(row=0, column=1, padx=5, pady=5, sticky="w")

    tk.Label(frame_info, text="Tên môn:", bg="#FFFFFF",
fg="#0D47A1").grid(row=0, column=2, padx=5, pady=5, sticky="w")
    entry_tenmon = tk.Entry(frame_info, width=25)
    entry_tenmon.grid(row=0, column=3, padx=5, pady=5, sticky="w")

    # ===== Frame danh sách =====
    frame_list = tk.LabelFrame(win, text="Danh sách môn học",
bg="#FFFFFF", fg="#0D47A1",
                               font=("Arial", 12, "bold"), padx=5,
pady=5)
    frame_list.pack(padx=15, pady=10, fill="both", expand=True)

    columns = ("mamon", "tenmon")
    tree = ttk.Treeview(frame_list, columns=columns, show="headings",
height=10)
    for col in columns:
        tree.heading(col, text=col.capitalize())
        tree.column(col, width=150, anchor="center")
    tree.pack(padx=5, pady=5, fill="both", expand=True)

    # ===== Frame nút =====
    frame_btn = tk.Frame(win, bg="#E3F2FD")
    frame_btn.pack(padx=15, pady=10, fill="x")
    for i in range(6):
        frame_btn.columnconfigure(i, weight=1)

    def make_btn(text, cmd=None, col=0, bg="#36F1DE"):
        btn = tk.Button(frame_btn, text=text, bg=bg, fg="black",
                        font=("Arial", 10, "bold"), cursor="hand2",
                        relief="raised", activebackground="#4DB6AC",
command=cmd)
        btn.grid(row=0, column=col, padx=5, pady=5, sticky="ew")
        return btn

    # ===== Hàm CRUD =====
    def clear_input():
```

```
        entry_mamon.delete(0, tk.END)
        entry_tenmon.delete(0, tk.END)

    def load_data():
        for item in tree.get_children():
            tree.delete(item)
        conn = connect_db()
        cur = conn.cursor()
        cur.execute("SELECT * FROM monhoc ORDER BY mamon")
        for row in cur.fetchall():
            tree.insert("", tk.END, values=row)
        conn.close()

    def them():
        mamon = entry_mamon.get().strip()
        tenmon = entry_tenmon.get().strip()
        if not mamon or not tenmon:
            messagebox.showwarning("Thiếu dữ liệu", "Nhập đầy đủ
thông tin!")
            return
        if not mamon.startswith("MH"):
            messagebox.showwarning("Lỗi", "Mã môn phải bắt đầu MH")
            return
        if re.search(r'\d', tenmon):
            messagebox.showwarning("Lỗi", "Tên môn không được chứa
số")
            return
        conn = connect_db()
        cur = conn.cursor()
        # ---------- Kiểm tra mã môn trùng ----------
        cur.execute("SELECT * FROM monhoc WHERE mamon=%s", (mamon,))
        if cur.fetchone():
            messagebox.showerror("Lỗi", f"Mã môn {mamon} đã tồn
tại!")
            conn.close()
            return

        # ---------- Kiểm tra tên môn trùng ----------
        cur.execute("SELECT * FROM monhoc WHERE tenmon=%s",
(tenmon,))
        if cur.fetchone():
            messagebox.showerror("Lỗi", f"Tên môn '{tenmon}' đã tồn
tại!")
            conn.close()
            return
        try:
            cur.execute("INSERT INTO monhoc(mamon, tenmon) VALUES
(%s,%s)", (mamon, tenmon))
            conn.commit()
```

```
            load_data()
            clear_input()
        except Exception as e:
            messagebox.showerror("Lỗi", str(e))
        conn.close()

    def xoa():
        selected = tree.selection()
        if not selected:
            messagebox.showwarning("Chưa chọn", "Hãy chọn môn học để
xoá!")
            return

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

        if not messagebox.askyesno("Xác nhận", f"Bạn có chắc muốn xoá
môn học {mamon}?"):
            return

        conn = connect_db()
        cur = conn.cursor()

        try:
            cur.execute("DELETE FROM monhoc WHERE mamon=%s",
(mamon,))
            conn.commit()
            tree.delete(selected)
        except Exception as e:
            messagebox.showerror("Lỗi", str(e))
        conn.close()


    def sua():
        selected = tree.selection()
        if not selected:
            messagebox.showwarning("Chưa chọn", "Chọn môn học để
sửa!")
            return
        messagebox.showinfo("Sửa", "Chỉnh sửa thông tin rồi nhấn
Lưu.")

    def luu():
        mamon = entry_mamon.get().strip()
        tenmon = entry_tenmon.get().strip()
        conn = connect_db()
        cur = conn.cursor()
        try:
            cur.execute("UPDATE monhoc SET tenmon=%s WHERE mamon=%s",
(tenmon, mamon))
```

```
            conn.commit()
            load_data()
            clear_input()
        except Exception as e:
            messagebox.showerror("Lỗi", str(e))
        conn.close()

    def huy():
        clear_input()

    def on_select(event):
        selected = tree.selection()
        if not selected: return
        values = tree.item(selected)["values"]
        entry_mamon.delete(0, tk.END); entry_mamon.insert(0,
values[0])
        entry_tenmon.delete(0, tk.END); entry_tenmon.insert(0,
values[1])

    # ===== Tạo nút =====
    buttons = [
        ("Thêm", them), ("Sửa", sua), ("Lưu", luu),
        ("Hủy", huy), ("Xóa", xoa), ("Thoát", lambda: (win.destroy(),
open_main_menu()))
    ]
    for idx, (text, cmd) in enumerate(buttons):
        make_btn(text, cmd, bg="#B0BEC5" if text=="Thoát" else
"#36F1DE").grid(row=0, column=idx, padx=5, pady=5)

    tree.bind("<<TreeviewSelect>>", on_select)
    load_data()
    win.mainloop()
```