Hướng dẫn code trang quản lý đơn hang

***I. Các bước chuẩn bị***

1. Tạo database

- Tải và cài đặt mysql <https://cdn.mysql.com//Downloads/MySQLInstaller/mysql-installer-community-8.0.32.0.msi>

- Kết nối với mysql trong python bằng thư viện mysql.connector (Nếu bị lỗi thì cài đặt thư viện mysql.connector bằng lệnh pip install mysql.connector trong cmd)

Text

Description automatically generated

- Sau khi cài đặt xong thì tạo database và tạo bảng trong mysql workbench

a. Tạo database

Graphical user interface, text, application, Word

Description automatically generated

- Cần chạy luôn đoạn code trên để tạo database trước khi tạo bảng

b. Tạo các bảng

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, text, application

Description automatically generated

***II. Hướng dẫn tổng quan về đoạn code***

Để kết nối với database, đoạn code này sử dụng một module được import từ file **db\_config.py** để lấy thông tin về cấu hình database. Sau đó, một đối tượng Flask được khởi tạo và các route được đăng ký để xử lý các yêu cầu từ người dùng.

Các route trong đoạn code bao gồm:

1. **/**: Đây là trang chính của ứng dụng, hiển thị danh sách đơn hàng. Các thông tin về đơn hàng được lấy từ database và được hiển thị lên trang web bằng cách sử dụng Jinja2 template.
2. **/home**: Đây là trang hiển thị danh sách sản phẩm. Các thông tin về sản phẩm được lấy từ database và hiển thị lên trang web bằng Jinja2 template.
3. **/add/<int:id>**: Đây là trang để thêm một sản phẩm vào đơn hàng. Khi người dùng click vào một sản phẩm, trang này sẽ hiển thị và cho phép người dùng nhập thông tin về khách hàng và số lượng sản phẩm. Khi người dùng submit thông tin, đoạn code sẽ thêm thông tin về khách hàng, đơn hàng và chi tiết đơn hàng vào database.
4. **/search/orders**: Đây là trang để tìm kiếm đơn hàng theo tên khách hàng, mã đơn hàng hoặc tên sản phẩm. Các thông tin được lấy từ database và hiển thị lên trang web bằng Jinja2 template.
5. **/search/products**: Đây là trang để tìm kiếm sản phẩm theo tên sản phẩm. Các thông tin được lấy từ database và hiển thị lên trang web bằng Jinja2 template.
6. **/update**: Đây là trang cập nhật đơn hang. Các thông tin hiển thị sẽ là thông tin của đơn hàng người dùng chọn để cập nhật, thông tin có thể cập nhật là: Tên khách hàng, số lượng sản phẩm, tổng số tiền
7. **/delete:** Xóa sản phẩm theo id sản phẩm khi người dùng chọn

Để thực hiện các thao tác với database, đoạn code này sử dụng một đối tượng **cursor** để thực hiện các câu lệnh SQL. Các câu lệnh SQL được sử dụng để truy vấn, thêm, sửa hoặc xóa các bản ghi trong database. Nếu thao tác với database thành công, đoạn code sẽ gọi phương thức **commit()** để lưu các thay đổi vào database. Nếu có lỗi xảy ra, phương thức **rollback()** sẽ được gọi để hủy bỏ các thay đổi.

Các chức năng của ứng dụng được định nghĩa bằng cách sử dụng decorators, ví dụ:

* **@app.route('/')**: Đây là decorator cho phép đăng ký URL cho hàm **index()**. Khi người dùng truy cập đường dẫn trang chủ '/', hàm **index()** được gọi để hiển thị danh sách đơn hàng lên trang web.
* **@app.route('/home')**: Đây là decorator cho phép đăng ký URL cho hàm **home()**. Khi người dùng truy cập đường dẫn '/home', hàm **home()** được gọi để hiển thị danh sách sản phẩm lên trang web.

***III. Hướng dẫn chi tiết về đoạn code***

* **from decimal import Decimal**: Đây là một module để làm việc với các số thập phân trong Python.
* **from flask import Flask, render\_template, request, redirect**: Import các module cần thiết để tạo ứng dụng Flask.
* **from db\_config import mydb, cursor**: Import cấu hình cơ sở dữ liệu từ module **db\_config**.
* **app = Flask(\_\_name\_\_)**: Tạo một đối tượng ứng dụng Flask.

Hàm **index()**:

* Sử dụng **cursor.execute()** để thực thi câu lệnh SQL để lấy dữ liệu đơn hàng và lưu vào biến **orders**.
* Trả về một template **index.html** để hiển thị dữ liệu.

Hàm **home()**:

* Sử dụng **cursor.execute()** để thực thi câu lệnh SQL để lấy dữ liệu sản phẩm và lưu vào biến **products**.
* Trả về một template **home.html** để hiển thị dữ liệu.

Hàm **add()**:

* Lấy dữ liệu sản phẩm từ bảng products với điều kiện là productID = id của sản phẩm được click.
* Kiểm tra xem dữ liệu được gửi lên là phương thức POST hay không và lấy dữ liệu gửi lên lưu vào các biến.
* Nếu dữ liệu nhập vào không đầy đủ, hiển thị thông báo lỗi và template **add.html**.
* Nếu dữ liệu nhập vào đầy đủ, bắt đầu transaction và thêm dữ liệu vào bảng customers, orders và orderDetails.
* Kết thúc transaction và quay lại trang đơn hàng (**/**).

Hàm **search\_orders()**:

* Lấy dữ liệu nhập vào từ ô input có tên là **search\_query**.
* Thực thi câu lệnh SQL để lấy ra các đơn hàng có tên khách hang hoặc tên sản phẩm hoặc mã đơn hàng

Hàm **search\_products()**:

* Lấy dữ liệu nhập vào từ ô input có tên là **search\_query**.
* Thực thi câu lệnh SQL để lấy ra các sản phẩm có tên tương ứng

Hàm update():

1. Sử dụng cursor.execute() để thực thi câu lệnh SQL để lấy thông tin sản phẩm cần chỉnh sửa dựa vào id sản phẩm được truyền vào.
2. Lấy thông tin sản phẩm từ kết quả truy vấn và gán vào biến product.
3. Kiểm tra nếu method là 'POST':
   * Sử dụng request.form để lấy các thông tin sản phẩm mới từ form chỉnh sửa.
   * Sử dụng câu lệnh SQL UPDATE để cập nhật thông tin sản phẩm mới vào cơ sở dữ liệu.
   * Chuyển hướng đến trang sản phẩm (detail) với id sản phẩm được truyền vào.
4. Trả về template update.html để hiển thị form chỉnh sửa sản phẩm, hiển thị các thông tin của sản phẩm được chọn để chỉnh sửa.

Hàm **delete()**:

* Nhận đầu vào là request và id của sản phẩm cần xóa
* Thực thi câu lệnh SQL để xóa các đơn hàng có id tương ứng