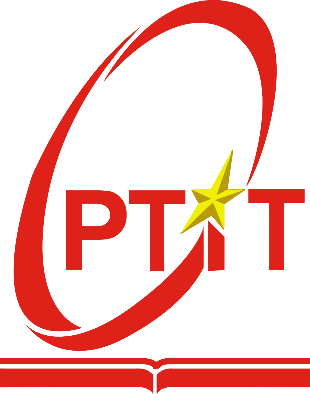
**BỘ THÔNG TIN VÀ TRUYỀN THÔNG**

**HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG**

**KHOA CÔNG NGHỆ THÔNG TIN 2**

-----------------------------



**BÁO CÁO CUỐI KỲ**

**IOT VÀ ỨNG DỤNG**

***Đề tài:***

**“TRIỂN KHAI DJANGO FRAMEWORK”**

**Giảng viên hướng dẫn :** **ThS. ĐÀM MINH LỊNH**

**Sinh viên thực hiện** **:**  **NGUYỄN THỊ TUYẾT NHI – N20DCCN050**

**TP.HCM, 1/2024**

**NHẬN XÉT CỦA GIẢNG VIÊN HƯỚNG DẪN**

........................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................

Thành phố Hồ Chí Minh, năm 2024

Giảng viên hướng dẫn

**LỜI CẢM ƠN**

Đề tài “Triển khai Django Framework” là nội dung em được phân công để nghiên cứu và làm báo cáo cuối kỳ môn IoT và Ứng dụng.

Để hoàn thành quá trình nghiên cứu và hoàn thiện báo cáo này, em xin chân thành cảm ơn sâu sắc đến Thầy Đàm Minh Lịnh. Thầy đã chỉ dạy và hướng dẫn em trong quá trình nghiên cứu để hoàn thiện báo cáo này.

Mặc dù đã cố gắng hoàn thành báo cáo với tất cả sự nỗ lực nhưng chắc chắn không tránh khỏi những thiếu sót, do thời gian tìm hiểu và trình độ hiểu biết còn hạn hẹp. Em rất mong được sự tận tình chỉ bảo thêm của quý thầy cô.

Thành phố Hồ Chí Minh, năm 2024

Sinh viên thực hiện

**Nguyễn Thị Tuyết Nhi**

**MỤC LỤC**

[CHƯƠNG 1: TỔNG QUAN 5](#_Toc155297603)

[1.1 Sự cấp thiết 5](#_Toc155297604)

[1.2 Giới thiệu về Django 5](#_Toc155297605)

[1.3 Kết luận 7](#_Toc155297606)

[CHƯƠNG 2: BẮT ĐẦU PHÁT TRIỂN VỚI DJANGO 8](#_Toc155297607)

[2.1 Tạo môi trường ảo 8](#_Toc155297608)

[2.2 Cài đặt Django 8](#_Toc155297609)

[2.3 Tạo dự án Django 8](#_Toc155297610)

[2.4 Chạy dự án Django 9](#_Toc155297611)

[2.5 Tạo ứng dụng Django 9](#_Toc155297612)

[2.6 Cấu hình cơ sở dữ liệu SQLite với Django 10](#_Toc155297613)

[2.7 Xây dựng Model 11](#_Toc155297614)

[2.8 Xây dựng View 13](#_Toc155297615)

[2.9 Xây dựng mẫu URL 15](#_Toc155297616)

[2.10 Xây dựng Template 16](#_Toc155297617)

[2.11 Trang quản trị Django 18](#_Toc155297618)

[CHƯƠNG 3: TRIỂN KHAI THỰC NGHIỆM 20](#_Toc155297619)

[3.1 Môi trường 20](#_Toc155297620)

[3.1.1 Giới thiệu PyCharm 20](#_Toc155297621)

[3.1.2 Giới thiệu ngôn ngữ Python 21](#_Toc155297622)

[3.1.3 Giới thiệu SQLite 22](#_Toc155297623)

[3.2 Demo 22](#_Toc155297624)

[CHƯƠNG 4: TỔNG KẾT 25](#_Toc155297625)

[4.1 Ưu điểm 25](#_Toc155297626)

[4.2 Hạn chế 25](#_Toc155297627)

[4.3 Hướng phát triển 25](#_Toc155297628)

[TÀI LIỆU THAM KHẢO 26](#_Toc155297629)

# **CHƯƠNG 1: TỔNG QUAN**

## **1.1 Sự cấp thiết**

Trong những năm gần đây, công nghệ thông tin không ngừng phát triển một cách mạnh mẽ và hiện đại. Sự ra đời của công nghệ thông tin làm phong phú bộ mặt xã hội, đời sống con người được nâng cao rõ rệt, đóng góp to lớn cho sự phát triển của nhân loại. Việc xây dựng website để phục vụ cho nhu cầu của con người ngày càng tăng cao. Để xây dựng được một website cách nhanh chóng và dễ dàng cần có sự hỗ trợ từ các công cụ mạnh mẽ. Django Framework là một sự hỗ trợ mạnh mẽ và linh hoạt để giúp làm điều đó.

## **1.2 Giới thiệu về Django**

Django là một framework mã nguồn mở dựa trên Python cho phép người dùng nhanh chóng tạo ra các ứng dụng web hiệu quả. Đặc điểm nổi bật của Django trong Python chính là tính chất linh hoạt, độ tin cậy cao và khả năng mở rộng lớn. [1]

Django sở hữu các bảng admin hay còn gọi là admin panel để giúp cho lập trình viên quản trị dễ dàng hơn so với các framework khác như Laravel hay Yii, ngoài ra Django còn có một số tính năng đặc biệt khác như:

* Có web server riêng
* Sử dụng cấu trục US MVC
* Có thư viện HTTP
* Có ORM
* Bao gồm Batteries
* Sở hữu Python Unit Test Framework

Ngoài ra, Django còn cung cấp giao diện CRUD (create, read, update, delete) được sử dụng để mô tả các lệnh cơ sở dữ liệu cơ bản, tạo điều kiện thuận lợi cho việc xem, thay đổi và tìm kiếm các thông tin.

Django sử dụng mô hình MVT (Model-View-Template). Về phương thức hoạt động sẽ giống với mô hình MVC, trong đó V (View) sẽ tương đương với C (Controller), T (Template) sẽ tương đương với V (View) ở các framework khác. Cấu trúc của một dự án Django bao gồm các thành phần sau [1]:

* **Mô hình (Model)**: Đây là thành phần chịu trách nhiệm cho việc xử lý dữ liệu. Nó đại diện cho các đối tượng trong ứng dụng và quản lý truy cập vào cơ sở dữ liệu.
* Đóng vai trò định nghĩa dữ liệu trong cơ sở dữ liệu.
* Xử lý các tương tác với cơ sở dữ liệu.
* Trong ứng dụng web, dữ liệu có thể được lưu trữ trong cơ sở dữ liệu quan hệ, cơ sở dữ liệu phi quan hệ, tệp XML.
* Django model là một lớp Python phác thảo các biến và phương thức cho một loại dữ liệu cụ thể.
* **Giao diện người dùng (View)**: Đây là thành phần chịu trách nhiệm cho việc hiển thị dữ liệu cho người dùng. Nó xử lý các yêu cầu từ người dùng và trả về các trang web hoặc API.
* Liên kết model với template.
* Là nơi viết code để tạo ra các trang web.
* Xác định dữ liệu nào sẽ được hiển thị trong template, lấy dữ liệu từ cơ sở dữ liệu và chuyển dữ liệu đến template.
* Trích xuất dữ liệu được đăng trong một form của template và chèn vào cơ sở dữ liệu.
* Mỗi trang trong trang web có một View riêng và về cơ bản là một hàm Python trong tệp view.py
* View cũng có thể thực hiện các tác vụ bổ sung như xác thực, gửi email, ...
* **Mẫu (Template)**: Đây là thành phần chịu trách nhiệm cho việc hiển thị dữ liệu theo cách định dạng. Nó sử dụng các biểu thức để hiển thị dữ liệu và tạo ra các trang web tĩnh.
* Template chỉ đơn giản là một trang HTML có thêm một số placeholders.
* Ngôn ngữ template của Django có thể được sử dụng để tạo nhiều dạng tệp văn bản khác nhau (XML, email, CSS, JavaScript, CSV).
* Django template cho phép tách phần trình bày dữ liệu khỏi logic liên quan đến dữ liệu thực tế sử dụng placeholder.
* Template nhận ngữ cảnh từ View và trình bày cơ sở dữ liệu trong các biến ngữ cảnh trong placeholder.
* **Cơ sở dữ liệu (Database)**: Đây là thành phần chịu trách nhiệm cho việc lưu trữ dữ liệu của ứng dụng.
* Django bao gồm một trình ánh xạ quan hệ đối tượng, một hệ thống tạo khuôn mẫu web và bộ điều phối URL dựa trên biểu thức chính quy.
* Hầu hết các ứng dụng web đều có phần lớp truy cập dữ liệu của cơ sở dữ liệu.
* Các nhà phát triển có nhiều lựa chọn cơ sở dữ liệu có thể được sử dụng cho các ứng dụng web bao gồm cả cơ sở dữ liệu quan hệ và không quan hệ.
* Django cung cấp một API phù hợp cho các lớp truy cập dữ liệu của cơ sở dữ liệu, do đó cho phép tự do lựa chọn cơ sở dữ liệu.
* Django hỗ trợ nhiều công cụ cơ sở dữ liệu quan hệ khác nhau bao gồm MySQL, PostgreSQL, Oracle và SQLite3.
* Django hỗ trợ cơ sở dữ liệu không quan hệ như MongoDB, MongoDB có thể được thêm bằng cách cài đặt các công cụ bổ sung.
* Với thiết kế ứng dụng web kết hợp với khả năng mạnh mẽ của Python và hệ sinh thái python => Django phù hợp nhất cho nhiều ứng dụng IoT.

**Web application framework** là một khuôn khổ phần mềm bao gồm các dịch vụ website, tài nguyên web hoặc có thể là API web (tập hợp các giải pháp, gói) được thiết kế với mục đích hỗ trợ các lập trình viên trong việc xây dựng và phát triển các ứng dụng web.

Mỗi dự án có nhiều ứng dụng và mỗi ứng dụng có thể là một phần của nhiều dự án.

Ứng dụng là nơi viết mã tạo nên chức năng của trang web.

## **1.3 Kết luận**

Việc sử dụng Django Framework để phát triển ứng dụng website là cần thiết vì các lý do sau:

* Tính năng đầy đủ: Django cung cấp một loạt các tính năng và thành phần sẵn có để xây dựng các trang web phức tạp. Nó bao gồm các thành phần quản lý người dùng, quản lý phiên, xử lý biểu mẫu, quản lý cơ sở dữ liệu và nhiều hơn nữa. Điều này giúp giảm thời gian và công sức cần thiết để triển khai các tính năng chung trong một ứng dụng web.
* Quản lý cơ sở dữ liệu dễ dàng: Django có hỗ trợ tích hợp cho nhiều hệ quản trị cơ sở dữ liệu, bao gồm SQLite, MySQL, PostgreSQL và Oracle. Nó cung cấp một lớp trừu tượng ORM (Object-Relational Mapping) mạnh mẽ cho phép tương tác với cơ sở dữ liệu một cách trực quan và dễ dàng.
* Bảo mật cao: Django tích hợp sẵn các tính năng bảo mật như xác thực người dùng, quản lý phiên, bảo vệ chống tấn công CSRF (Cross-Site Request Forgery) và kiểm tra dữ liệu đầu vào. Điều này giúp bảo vệ ứng dụng web khỏi các lỗ hổng bảo mật phổ biến và cung cấp một môi trường an toàn cho người dùng.
* Quản lý URL và định tuyến linh hoạt: Django cung cấp một hệ thống định tuyến mạnh mẽ cho phép bạn quản lý các URL trong ứng dụng web dễ dàng. Bằng cách định tuyến các yêu cầu HTTP đến các hàm xử lý tương ứng, Django giúp bạn tổ chức mã nguồn một cách rõ ràng và dễ bảo trì.
* Cộng đồng phát triển lớn: Django có một cộng đồng phát triển rộng lớn, với nhiều tài liệu, ví dụ và gói mở rộng sẵn có. Dễ dàng tìm thấy sự hỗ trợ và giải quyết các vấn đề trong quá trình phát triển.
* Khả năng mở rộng: Với kiến trúc linh hoạt, Django cho phép mở rộng ứng dụng web dễ dàng khi nhu cầu mở rộng phát sinh. Có thể thêm các thành phần tùy chỉnh, phát triển các ứng dụng con và tích hợp các dịch vụ bên thứ ba.

Tóm lại, Django là một framework phát triển ứng dụng web mạnh mẽ và linh hoạt. Việc sử dụng Django giúp giảm thời gian và công sức cần thiết để xây dựng các tính năng chung, cung cấp bảo mật, quản lý cơ sở dữ liệu, quản lý URL và định tuyến, và có một cộng đồng phát triển lớn để hỗ trợ.

# **CHƯƠNG 2: BẮT ĐẦU PHÁT TRIỂN VỚI DJANGO**

## **2.1 Tạo môi trường ảo**

Để kiểm tra xem hệ thống đã cài đặt Python chưa, hãy chạy lệnh này trong dấu nhắc lệnh: **py --version**

Nếu Python đã cài đặt, sẽ nhận được kết quả với số phiên bản, ví dụ như thế này: **Python 3.9.13**

Để cài đặt Django, phải sử dụng trình quản lý gói như PIP, được bao gồm trong Python từ phiên bản 3.4. Để kiểm tra xem hệ thống đã cài đặt PIP chưa, hãy chạy lệnh này trong dấu nhắc lệnh: **py -m pip --version**

Nếu PIP đã cài đặt, sẽ nhận được kết quả với số phiên bản: **pip 22.0.4 from C:\Users\hello\AppData\Local\Programs\Python\Python39\lib\site-packages\pip (python 3.9)**

Nên tạo môi trường ảo cho từng dự án Django vì cần cài đặt các gói khác nhau với các phiên bản khác nhau. Nhập nội dung sau vào dấu nhắc lệnh, điều hướng đến nơi muốn tạo dự án: **py -m venv django\_project**

Kích hoạt môi trường, bằng cách gõ lệnh này: **django\_project\Scripts\activate.bat**

Khi môi trường được kích hoạt, sẽ thấy kết quả này trong dấu nhắc lệnh:

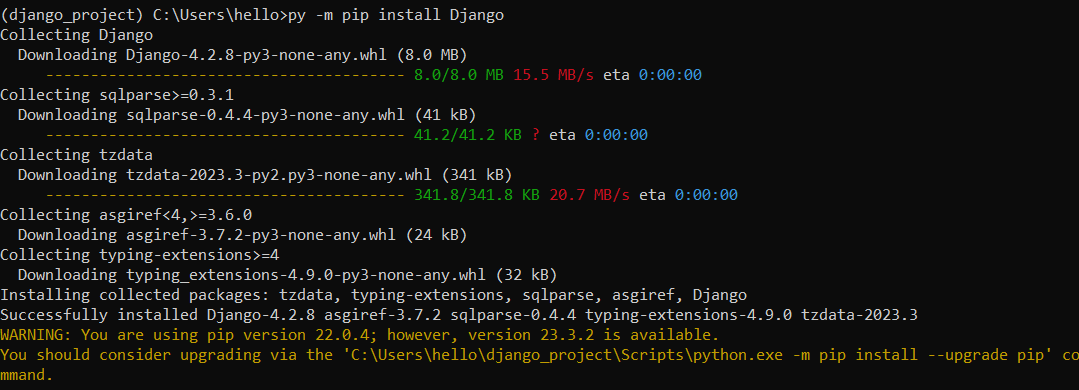
**(django\_project) C:\Users\hello>**

* Lưu ý: Phải kích hoạt môi trường ảo mỗi khi mở dấu nhắc lệnh để làm việc trên dự án.

## **2.2 Cài đặt Django**

Django được cài đặt bằng pip, với lệnh này: **py -m pip install Django**

Điều này sẽ cho kết quả trông như thế này:



Kiểm tra phiên bản Django: **django-admin --version**

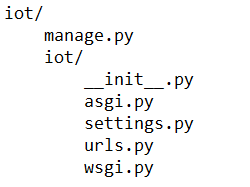
Nếu Django đã cài đặt, sẽ nhận được kết quả với số phiên bản: **4.2.8**

## **2.3 Tạo dự án Django**

Hãy điều hướng đến nơi trong hệ thống tệp muốn lưu trữ mã (trong môi trường ảo) và chạy lệnh này trong dấu nhắc lệnh với tên dự án là **iot**:

**(django\_project) C:\Users\hello>django-admin startproject iot**

Django tạo một thư mục **iot** trên máy tính, với nội dung sau:

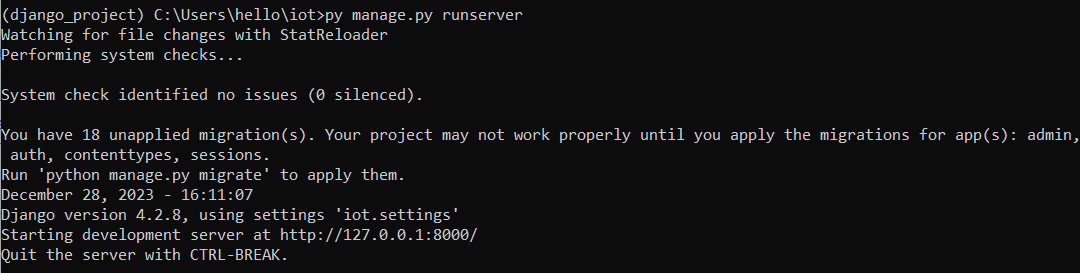


## **2.4 Chạy dự án Django**

Điều hướng đến thư mục **/iot** và thực hiện lệnh này trong dấu nhắc lệnh:

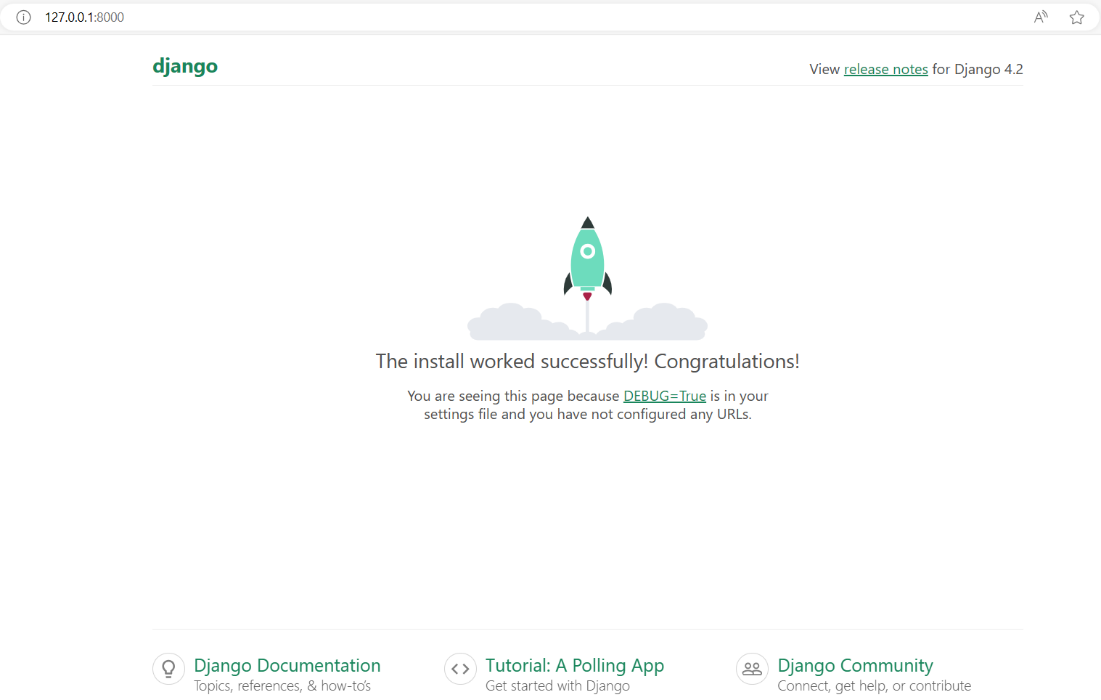
**py manage.py runserver**

Điều này sẽ tạo ra kết quả này:



Mở cửa sổ trình duyệt mới và nhập 127.0.0.1:8000 vào thanh địa chỉ.

Kết quả:



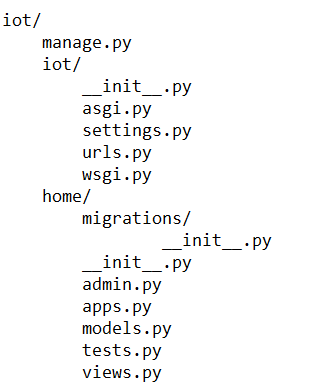
## **2.5 Tạo ứng dụng Django**

Ví dụ đặt tên cho ứng dụng là **home**.

Bắt đầu bằng cách điều hướng đến vị trí đã chọn nơi muốn lưu trữ ứng dụng, trong trường hợp này là thư mục **iot** và chạy lệnh này: **py manage.py startapp home**

Nếu máy chủ vẫn đang chạy và không thể viết lệnh, hãy nhấn [CTRL] [BREAK], hoặc [CTRL] [C] để dừng máy chủ và sẽ trở lại môi trường ảo.

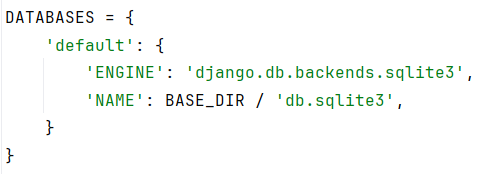
Django tạo một thư mục có tên **home** trong dự án, với nội dung như sau:



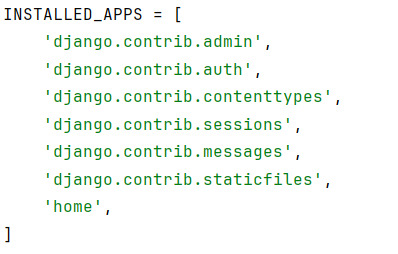
## **2.6 Cấu hình cơ sở dữ liệu SQLite với Django**

Mặc định, Django server sử dụng cơ sở dữ liệu Sqlite. Do vậy khi tạo một dự án web bằng Django, Sqlite sẽ được cài đặt sẵn. [2]

Xem thông tin cấu hình trong tập tin settings.py



Khai báo ứng dụng **home** vừa tạo trong tập tin settings.py



Tải và cài đặt phần mềm DB browser for SQLite để quan sát và thao tác với Sqlite: https://sqlitebrowser.org/. Đi đến dấu nhắc lệnh và đưa ra lệnh sqlite3, sẽ hiển thị kết quả như sau:

**C:\Users\hello>sqlite3**

**SQLite version 3.44.2 2023-11-24 11:41:44 (UTF-16 console I/O)**

**Enter ".help" for usage hints.**

**Connected to a transient in-memory database.**

**Use ".open FILENAME" to reopen on a persistent database.**

**sqlite>**

## **2.7 Xây dựng Model**

Khi tạo ứng dụng web, Django sẽ tạo sẵn tập tin model.py, chúng ta sẽ khai báo các model dữ liệu trong tập tin này. Khai báo model dữ liệu giống như khai báo các lớp trong lập trình hướng đối tượng. Ví dụ: tạo ra class Task để lưu các công việc [3].



Lớp Task kế thừa từ lớp django.db.models.Model: **class Task(models.Model)**

Mỗi thuộc tính trong model kế thừa từ lớp Field, đây là lớp ảo trong Django, lớp này có nhiều lớp kế thừa khác đại diện cho các kiểu dữ liệu. Ví dụ: CharField đại diện cho kiểu dữ liệu char, TextField đại diện cho kiểu dữ liệu text, DateTimeField đại diện cho kiểu dữ liệu thời gian.

Sau khi đã tạo các class trong models.py, chạy lệnh **makemigrations home** để báo cho Django cập nhật các nội dung vừa được khai báo trong models.py:

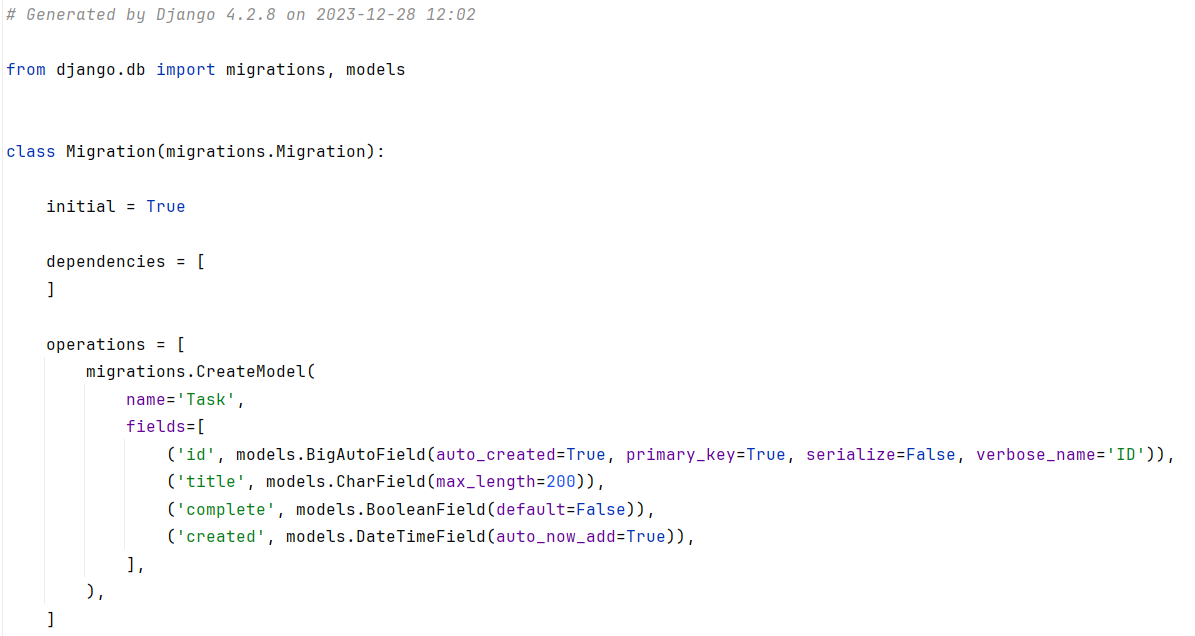
**(django\_project) C:\Users\hello\iot>py manage.py makemigrations home**

**Migrations for 'home':**

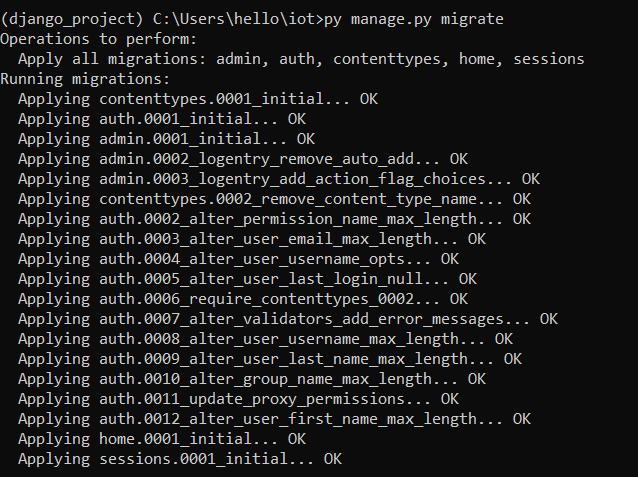
**home\migrations\0001\_initial.py**

**- Create model Task**

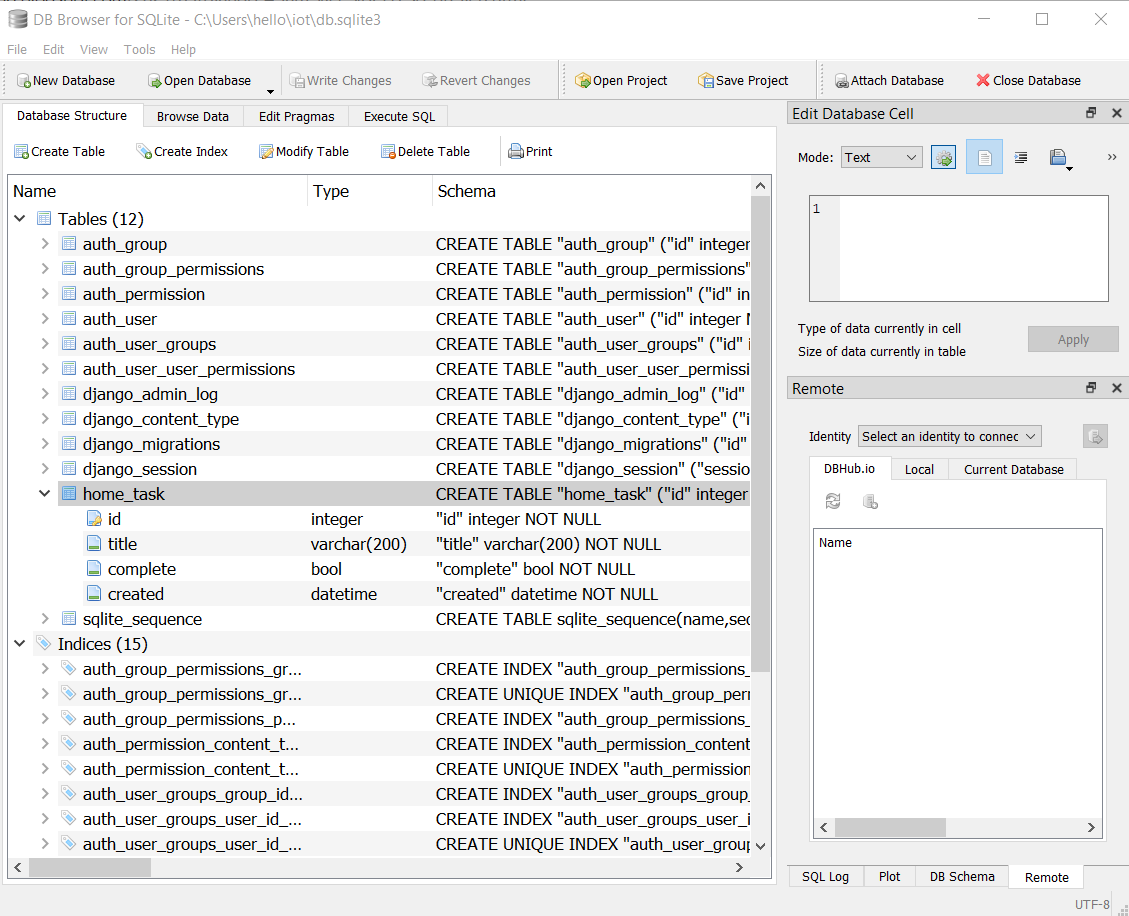
Model Task đã được tạo ra. Django sẽ tạo ra tập tin 0001\_initial.py để lưu sự thay đổi về model. Mở tập tin home\migrations\0001\_initial.py để xem kết quả:



Để cập nhật lại cơ sở dữ liệu, nghĩa là lưu và thực thi các thay đổi từ model vào Sqlite, sử dụng lệnh **migrate**.



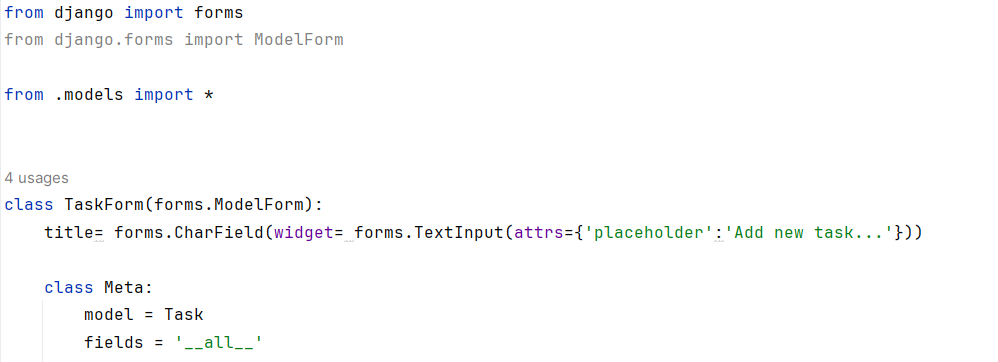
Mở lại DB browser, vào menu File/Open Database…, tìm và mở tập tin db.sqlite3. sẽ thấy xuất hiện bảng **home\_task**.



Vậy là đã tạo được bảng để chứa dữ liệu, từ đây có thể thực hiện các thao tác thêm, sửa, xóa các dòng tin trong bảng dữ liệu [2].

## **2.8 Xây dựng View**

Sau đó, tiếp tục triển khai phần views của ứng dụng. Như đã nói ban đầu, view của Django quyết định các dữ liệu được trả về và điều hướng các request http của ứng dụng.







Sau khi đã có model và view, việc còn lại cần làm là phải tạo route(url) và template cho ứng dụng để người dùng có thể truy cập vào ứng dụng. [3]

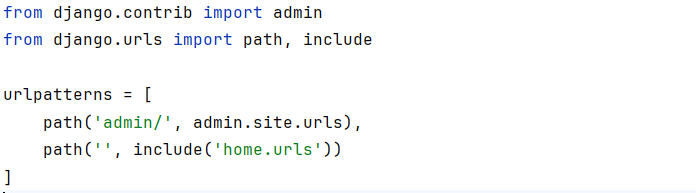
## **2.9 Xây dựng mẫu URL**

Mẫu URL là một cách ánh xạ URL tới các View sẽ xử lý các yêu cầu URL.

URLs do người dùng yêu cầu phải khớp với mẫu URL.

View tương ứng với mẫu phải khớp với URL được sử dụng để xử lý yêu cầu.

Trước tiên cần điều hướng để những request đến url mặc định '/' từ **iot\iot\urls.py** sẽ được đưa đến **iot\home\urls.py** thông qua việc include **path('', include('home.urls'))**

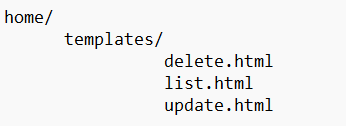




Để người dùng có thể request đến đúng tác vụ mong muốn thì sẽ cần thiết lập các url tương ứng với các functions đã tạo ở file views. (Ví dụ: khi người dùng truy cập vào địa chỉ <http://127.0.0.1:8000/> thì hàm index sẽ được gọi)

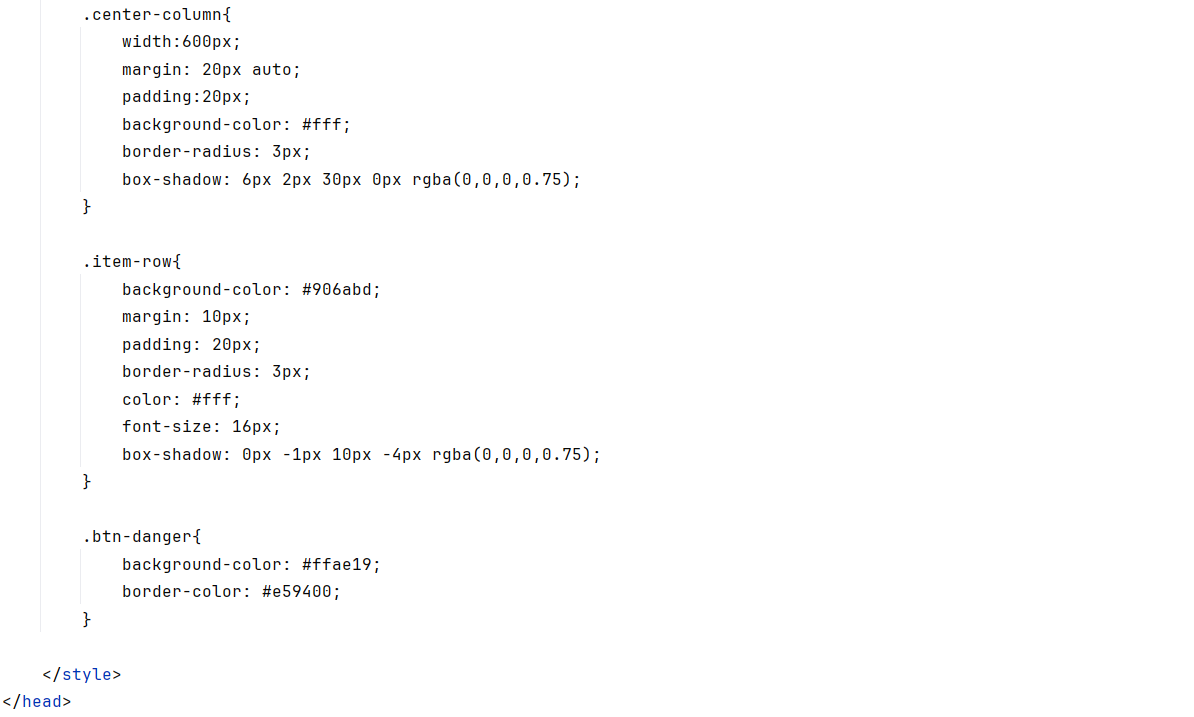
## **2.10 Xây dựng Template**

Tạo template, trước tiên cần tạo các file có cấu trúc thư mục như sau [3]:



list.html







delete.html



update.html



Vì tính bảo mật nên tất cả các form trong django đều sẽ cần có **csrf\_token.**

## **2.11 Trang quản trị Django**

Django cung cấp một hệ thống quản trị cho phép quản lý trang web mà không cần viết thêm mã.

Hệ thống “quản trị” đọc Django Model và cung cấp giao diện có thể được sử dụng để thêm nội dung vào trang web.

Trang quản trị Django được kích hoạt bằng cách thêm django.contrib.admin và django.contrib.admindocs vào phần INSTALLED\_APP trong tệp setting.py

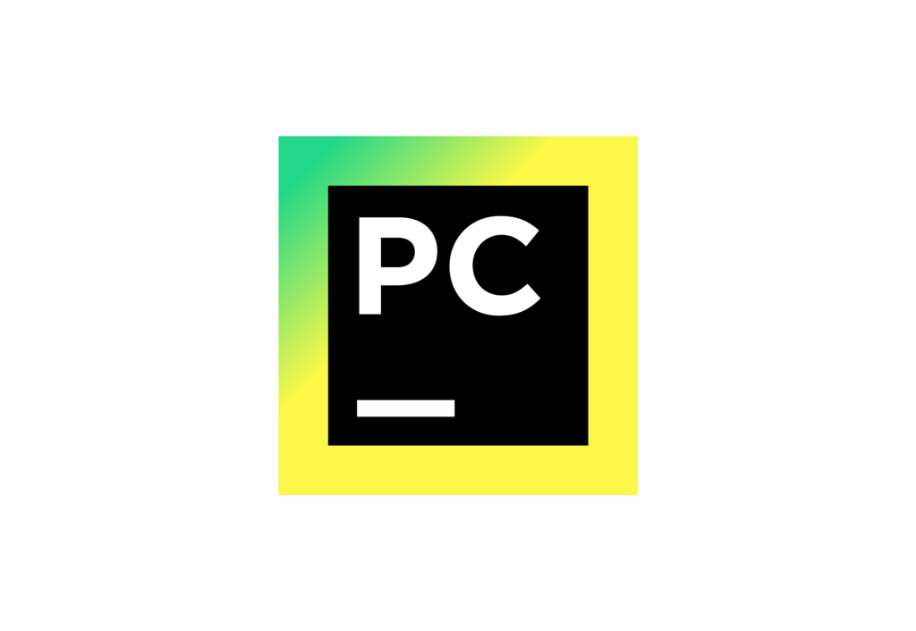
Trang quản trị yêu cầu định nghĩa mẫu URL trong urls.py

Để xác định các mô hình ứng dụng có thể chỉnh sửa trong giao diện quản trị, một tệp mới có tên admin.py sẽ được tạo.

# **CHƯƠNG 3: TRIỂN KHAI THỰC NGHIỆM**

## **3.1 Môi trường**

### **3.1.1 Giới thiệu PyCharm**



PyCharm là một môi trường phát triển tích hợp (IDE) dành riêng cho việc phát triển ứng dụng Python. Được phát triển bởi JetBrains, PyCharm cung cấp nhiều tính năng mạnh mẽ và công cụ hữu ích để giúp lập trình viên Python tạo ra các dự án phức tạp và hiệu quả. [4]

Dưới đây là một số đặc điểm và tính năng chính của PyCharm:

* Môi trường phát triển toàn diện: PyCharm cung cấp một môi trường phát triển toàn diện cho Python, bao gồm trình soạn thảo mã, gỡ lỗi, và khả năng chạy và kiểm tra mã. Hỗ trợ cả Python 2 và Python 3, cho phép phát triển ứng dụng trên nền tảng Python phổ biến.
* Giao diện người dùng thân thiện: PyCharm có một giao diện người dùng thân thiện và dễ sử dụng. Cung cấp các công cụ giúp tăng năng suất, bao gồm gợi ý mã, hoàn thành tự động, điều hướng nhanh, và tìm kiếm thông minh. Điều này giúp giảm thời gian và công sức cần thiết để viết mã.
* Hỗ trợ quản lý dự án: PyCharm cho phép quản lý dự án Python một cách dễ dàng. Có thể tạo, mở và tổ chức các tệp tin và thư mục trong dự án. Nó cũng tích hợp với hệ thống kiểm soát phiên bản như Git và SVN, cho phép làm việc với phiên bản mã nguồn quản lý.
* Gỡ lỗi và kiểm tra mã: PyCharm cung cấp một bộ công cụ mạnh mẽ để gỡ lỗi và kiểm tra mã Python. Có thể đặt các điểm dừng (breakpoint) để theo dõi và kiểm tra các biến và giá trị trong quá trình thực thi mã. Nó cũng cung cấp các công cụ phân tích mã để phát hiện lỗi cú pháp và lỗi logic trong mã.
* Hỗ trợ đa nền tảng: PyCharm là một IDE đa nền tảng, hỗ trợ Windows, macOS và Linux. Điều này cho phép làm việc trên bất kỳ hệ điều hành nào.
* Cộng đồng và tài liệu phong phú: PyCharm có một cộng đồng người dùng lớn và nhiều tài liệu hữu ích. Có thể tìm thấy hỗ trợ, câu trả lời và ví dụ từ cộng đồng này, giúp giải quyết các vấn đề trong quá trình phát triển.

### **3.1.2 Giới thiệu ngôn ngữ Python**



Python là một ngôn ngữ lập trình cao cấp, đa nền tảng và có mã nguồn mở. Được tạo ra bởi Guido van Rossum và được ra mắt lần đầu vào năm 1991. Python được thiết kế với cấu trúc rõ ràng, dễ đọc và dễ hiểu, giúp người lập trình tập trung vào giải quyết vấn đề thay vì lo lắng về cú pháp phức tạp [5]. Và vì thế nó được sử dụng rộng rãi.

Dưới đây là một số đặc điểm chính của Python:

* Cú pháp đơn giản: Python có cú pháp đơn giản, gần gũi với ngôn ngữ tự nhiên. Điều này làm cho nó dễ học và dễ đọc. Sử dụng các dòng mã ngắn, có thể thể hiện ý tưởng một cách rõ ràng và dễ hiểu.
* Đa năng: Python là một ngôn ngữ đa mục đích, cho phép phát triển các loại ứng dụng phong phú, từ ứng dụng web, phân tích dữ liệu, trí tuệ nhân tạo, đến lập trình hệ thống và nhiều hơn nữa. Nó cung cấp nhiều thư viện và framework mạnh mẽ để hỗ trợ các lĩnh vực khác nhau.
* Cộng đồng phát triển mạnh mẽ: Python có một cộng đồng phát triển lớn và nhiều tài liệu phong phú. Có thể tìm thấy hàng ngàn thư viện, framework và công cụ hữu ích từ cộng đồng này. Cộng đồng Python cũng rất hỗ trợ và thân thiện, giúp giải quyết các vấn đề và tìm kiếm sự khuyến nghị.
* Hỗ trợ đa nền tảng: Python có thể chạy trên nhiều hệ điều hành như Windows, macOS và Linux. Điều này cho phép phát triển ứng dụng trên nền tảng ưa thích và triển khai trên nhiều môi trường khác nhau.
* Tích hợp dễ dàng: Python có thể dễ dàng tích hợp với các ngôn ngữ khác như C/C++, Java và .NET. Điều này cho phép sử dụng các thư viện và mã nguồn mở có sẵn trong các ngôn ngữ khác và mở rộng khả năng của ứng dụng.
* Tính bảo mật cao: Python cung cấp các công cụ và thư viện mạnh mẽ để phát triển các ứng dụng bảo mật. Với các tính năng như quản lý bộ nhớ tự động, kiểm tra kiểu dữ liệu và xử lý ngoại lệ, Python giúp giảm thiểu các lỗ hổng bảo mật phổ biến.

### **3.1.3 Giới thiệu SQLite**



SQLite là một hệ quản trị cơ sở dữ liệu hay còn gọi là hệ thống cơ sở dữ liệu quan hệ nhỏ gọn. SQLite là một thư viện phần mềm mà triển khai một SQL Database Engine truyền thống, không cần mô hình client-server nên rất nhỏ gọn. SQLite được sử dụng vào rất nhiều chương trình từ desktop đến mobile hay là website. Được tích hợp sẵn trong hầu hết các ngôn ngữ lập trình, bao gồm Python. [6]

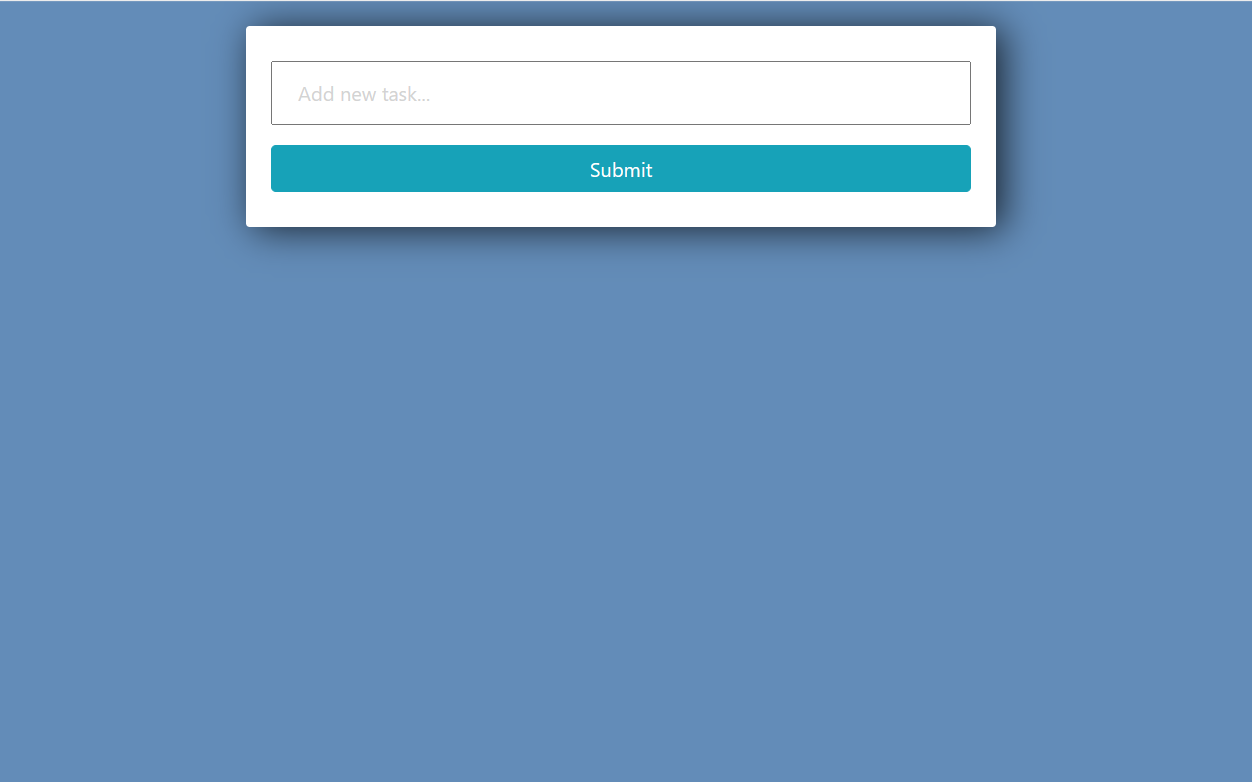
Dưới đây là một số điểm nổi bật về SQLite:

* Nhỏ gọn: SQLite là một thư viện nhỏ gọn và dễ dùng, không đòi hỏi cài đặt hoặc cấu hình phức tạp. Dữ liệu được lưu trữ trong một tệp đơn, giúp việc triển khai và quản lý cơ sở dữ liệu dễ dàng.
* Quản lý cơ sở dữ liệu quan hệ: SQLite hỗ trợ các tính năng quan hệ cơ bản như các bảng, cột, ràng buộc và khóa ngoại. Điều này cho phép bạn tổ chức dữ liệu theo cách quen thuộc và thực hiện các thao tác truy vấn SQL.
* Hỗ trợ giao dịch: SQLite hỗ trợ giao dịch ACID (Atomicity, Consistency, Isolation, Durability), cho phép bạn thực hiện các thao tác cùng nhau theo cách an toàn và đáng tin cậy. Nếu một giao dịch không thành công, dữ liệu sẽ được phục hồi về trạng thái ban đầu.
* Tích hợp sẵn: SQLite được tích hợp sẵn trong Python, không yêu cầu cài đặt thêm. Có thể sử dụng giao diện Python như sqlite3 để tương tác với cơ sở dữ liệu SQLite một cách dễ dàng.
* Độ tin cậy và hiệu suất: Mặc dù SQLite là một hệ quản trị cơ sở dữ liệu nhỏ gọn, nó vẫn đảm bảo độ tin cậy và hiệu suất cao. SQLite sử dụng cơ chế tập tin đọc/ghi trực tiếp, giúp giảm thiểu tải trọng mạng và tăng tốc độ truy cập dữ liệu.
* Đa nền tảng: SQLite hỗ trợ trên nhiều nền tảng, bao gồm Windows, macOS, Linux và các hệ điều hành nhúng. Điều này cho phép bạn phát triển ứng dụng sử dụng SQLite trên nhiều môi trường.

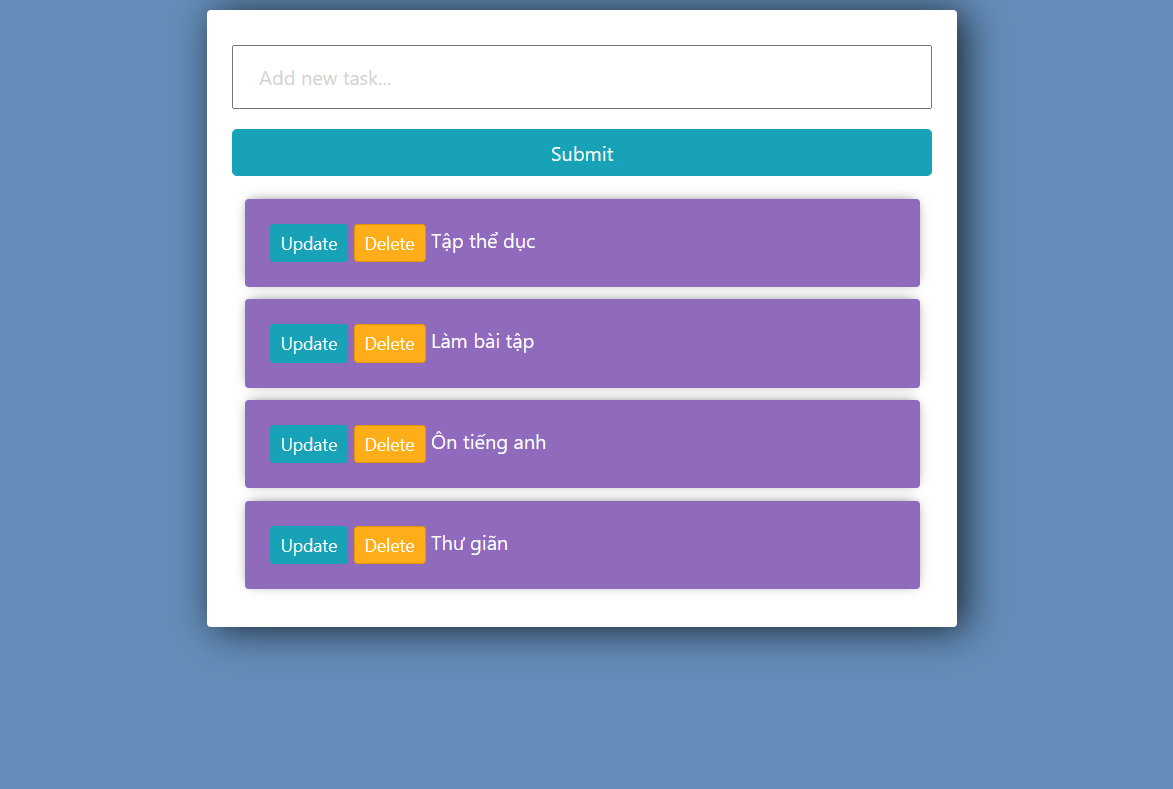
Với tính linh hoạt, hiệu suất và tính đáng tin cậy, SQLite là một lựa chọn phổ biến để lưu trữ và quản lý dữ liệu trong ứng dụng Python.

## **3.2 Demo**

Giao diện tạo nhiệm vụ: nhập tên nhiệm vụ, sau đó nhấn submit.



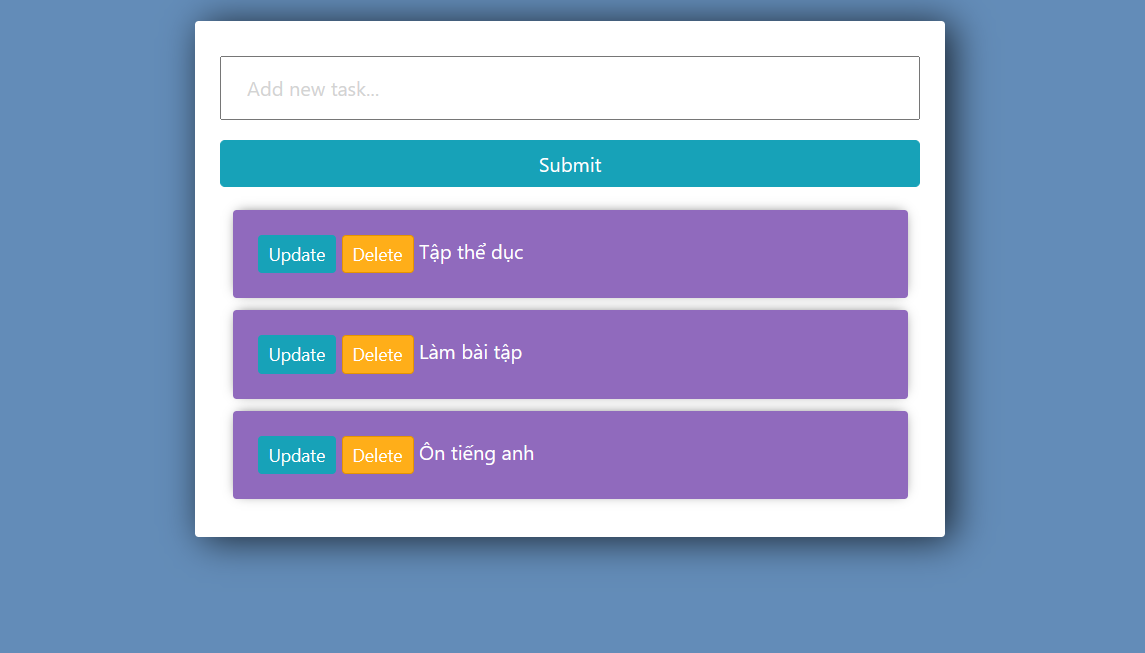
Danh sách các nhiệm vụ đã tạo:



Giao diện xóa nhiệm vụ đã tạo. Ví dụ: xóa “Thư giãn”.



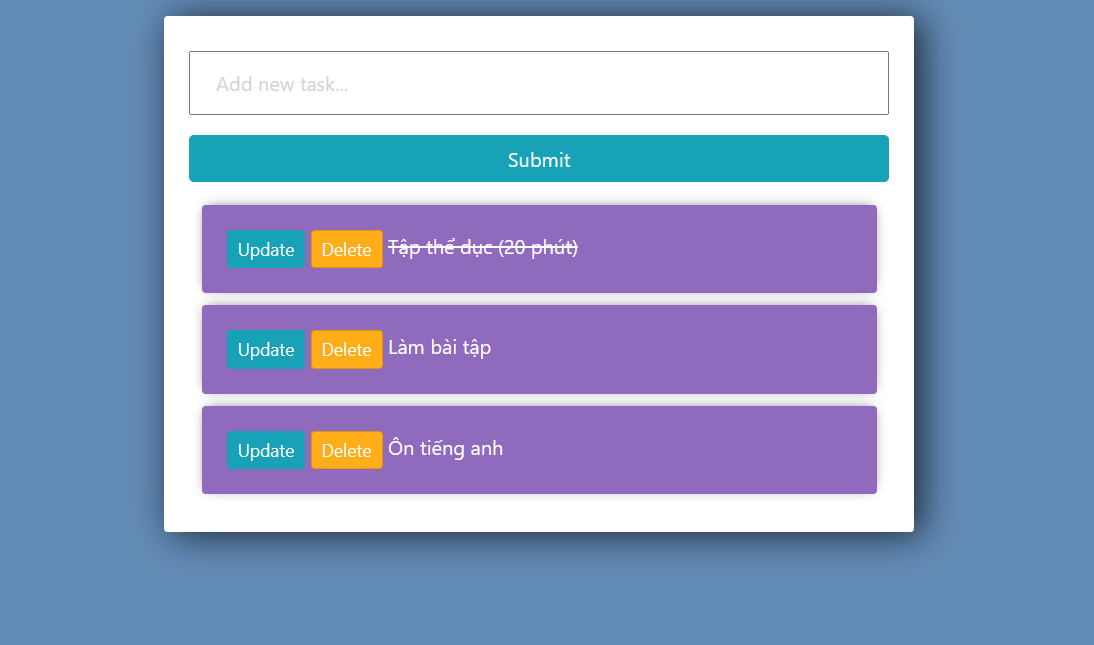
Giao diện sau khi đã xóa nhiệm vụ “Thư giãn”:



Giao diện cập nhật nhiệm vụ:



Giao diện sau khi đã cập nhật lại tên nhiệm vụ “Tập thể dục” và tình trạng đã hoàn thành.



# **CHƯƠNG 4: TỔNG KẾT**

## **4.1 Ưu điểm**

* Triển khai được một website sử dụng Django Framework.
* Trình bày các bước thực hiện chi tiết, rõ ràng.

## **4.2 Hạn chế**

* Giao diện còn đơn điệu, chưa đẹp mắt.
* Còn thiếu nhiều chức năng.

## **4.3 Hướng phát triển**

* Hoàn thiện website một cách chuyên nghiệp hơn về giao diện.
* Phát triển thêm nhiều chức năng.

# **TÀI LIỆU THAM KHẢO**

[1] W3schools, *Introduction to Django*, w3schools.com.

[2] Lê Gia Công, *Django (4) - Làm việc với cơ sở dữ liệu*, legiacong.blogspot.com, 4/5/2021.

[3] GMO-Z, *Xây dựng ứng dụng web sử dụng Django Framework*, vietnamlab.vn, 14/10/2022.

[4] GeeksforGeeks, *PyCharm Introduction – A Beginner’s Guide*, geeksforgeeks.org, 4/10/2023.

[5] GeeksforGeeks, *Introduction to Python*, geeksforgeeks.org, 14/8/2023.

[6] GeeksforGeeks, *Introduction to SQLite*, geeksforgeeks.org, 17/1/2023.