

**BỘ GIÁO DỤC VÀ ĐÀO TẠO
TRƯỜNG ĐẠI HỌC SƯ PHẠM TP. HỒ CHÍ MINH
KHOA CÔNG NGHỆ THÔNG TIN**

ĐỀ TÀI

**TÌM HIỂU VỀ FRAMEWORK PYTHON DJANGO VÀ
XÂY DỰNG WEB TODOLIST CƠ BẢN**

Học phần: COMP103101 – CÔNG NGHỆ WEB

**BỘ GIÁO DỤC VÀ ĐÀO TẠO
Thành phố Hồ Chí Minh, ngày 24 tháng 11 năm 2021**

TRƯỜNG ĐẠI HỌC SƯ PHẠM TP. HỒ CHÍ MINH
KHOA CÔNG NGHỆ THÔNG TIN

ĐỀ TÀI

**TÌM HIỂU VỀ FRAMEWORK PYTHON DJANGO VÀ
XÂY DỰNG WEB TODOLIST CƠ BẢN**

Học phần: COMP103101 – CÔNG NGHỆ WEB

Nhóm sinh viên thực hiện:

1. Lê Thanh Thoại – 4501104227
2. Huỳnh Anh Dự – 4501104041
3. Nguyễn Văn Giàu – 4501104061
4. Trương Đình Anh - 4501104011

Giảng viên hướng dẫn: ThS. Lương Trần Hy Hiến

Thành phố Hồ Chí Minh, ngày 17 tháng 9 năm 2021

MỤC LỤC

MỞ ĐẦU.....	4
CHƯƠNG 1. TÌM HIỂU VỀ PYTHON DJANGO.....	5
1. Giới thiệu về Python	5
2. Python Django là gì?	6
3. Cài đặt Python Django	7
3.1. Cài đặt	7
3.2. Tạo project.....	7
3.3. Cài đặt môi trường	9
3.4. Chạy server.....	10
3.5. Tạo web app	11
3.6. Model	11
3.7. Hệ thống admin	11
3.8. View và templates	13
CHƯƠNG 2. XÂY DỰNG WEB TODOLIST CƠ BẢN	15
CHƯƠNG 3. KẾT LUẬN	23
TÀI LIỆU THAM KHẢO.....	24

KẾT CẤU TIỂU LUẬN
NHIỆM VỤ THÀNH VIÊN NHÓM

Họ tên	Công việc
Lê Thanh Thoại	Code, làm báo cáo
Huỳnh Anh Dự	Code, tìm hiểu
Nguyễn Văn Giàu	Kiểm thử, làm báo cáo
Trương Đình Anh	Kiểm thử, làm trình chiếu

DANH MỤC CÁC HÌNH VẼ

Hình 1-1: Cài đặt python Django	7
Hình 1-2: Kiểm tra version django.....	7
Hình 1-3: Tạo project	8
Hình 1-4: Cấu trúc project.....	8
Hình 1-5: Cài đặt môi trường ảo	9
<i>Hình 1-6: Đã activate môi trường ảo</i>	<i>9</i>
Hình 1-7: Khởi chạy server	10
Hình 1-8: Chạy server thành công.....	10
Hình 1-9: Cấu trúc app	11
Hình 1-10: Giao diện đăng nhập Django administration	12
Hình 1-11: Tạo tài khoản thành công	12
Hình 1-12: Đăng nhập thành công	13
Hình 2-1: Đăng kí task trên Django admin	15
Hình 2-2: Task đã được đăng ký thành công	15
Hình 2-3: User	16
Hình 2-4: Phân quyền User	16
Hình 2-5: Chỉnh sửa Task của admin	17
Hình 2-6: Code chi tiết thêm, sửa todolist.....	17
Hình 2-7: Giao diện & code thêm công việc	18
Hình 2-8: Code giao diện chính	18
Hình 2-9: Giao diện home todolis	19
Hình 2-10: Code đăng nhập.....	19
Hình 2-11: Code đăng ký	20
Hình 2-12: Code xóa việc.....	20
Hình 2-13: Url patterns.....	20
Hình 2-14: Giao diện chính	21
Hình 2-15: Giao diện đăng nhập và đăng ký	21
Hình 2-16: Giao diện Thêm, Sửa, Xóa việc	22
Hình 2-17: Giao diện footer thông tin nhóm.....	22

MỞ ĐẦU

1. Lý do chọn đề tài

Django là một trong những Web Framework phổ biến nhất được viết bằng Python, cung cấp nhiều tính năng cho việc phát triển web về bảo mật, database access, session, routing, localization ... Nhóm chúng em đã lựa chọn đề tài “Tìm hiểu Framework Python Django” để nghiên cứu, tìm hiểu đề tài giữa kì môn học *Công nghệ web*.

2. Mục tiêu và nhiệm vụ nghiên cứu

- Tìm hiểu lý thuyết về Python Django.
- Nghiên cứu cách cài đặt và chạy thử các chương trình.
- Xây dựng cấu trúc, các thành phần chính trong Python Django.
- Xây dựng website Todolist cơ bản.

3. Đối tượng và phạm vi nghiên cứu

- Đối tượng nghiên cứu: Python Django, Website Todolist cơ bản.
- Phạm vi nghiên cứu: Tìm hiểu khái quát về React, nắm được các định nghĩa, khái niệm về Django, cách render, import trong Django... Về website Todolist cơ bản, người dùng có thể thao tác thêm, sửa, xóa, đăng kí, đăng nhập, tìm kiếm việc.

4. Phương pháp nghiên cứu

- Sử dụng các nguồn link youtube, blogs, eBook để nghiên cứu và tìm ra hướng giải quyết vấn đề.

5. Kết cấu của đề tài

Nội dung bài tiểu luận được xây dựng gồm các phần sau:

- **Chương 1: Tìm hiểu về Python Django**
- **Chương 2: Xây dựng Website Todolist cơ bản**
- **Chương 3: Kết luận**

CHƯƠNG 1. TÌM HIỂU VỀ PYTHON DJANGO

1. Giới thiệu về Python

Python đã được hình thành vào cuối những năm 1980 và được bắt đầu thực hiện vào tháng 12/1989 bởi Guido van Rossum tại CWI tại Hà Lan như là người kế thừa của ngôn ngữ ABC (tự lấy cảm hứng từ SETL) có khả năng xử lý ngoại lệ và giao tiếp với hệ điều hành Amoeba. Van Rossum là tác giả chính của Python, và vai trò trung tâm của ông tiếp tục trong việc quyết định hướng phát triển của Python được phản ánh trong tiêu đề mà cộng đồng Python dành cho ông “Độc tài nhân từ cho cuộc sống” (benevolent dictator for life)(BDFL).

Python 2.0 được phát hành vào ngày 16/10/2000, với nhiều tính năng chính mới bao gồm một bộ dọn rác đầy đủ và hỗ trợ Unicode. Với phiên bản này, quá trình phát triển đã được thay đổi và trở thành minh bạch hơn và được cộng đồng ủng hộ.

Python 3.0 (còn được gọi là Python 3000 hoặc Py3k), một bản phát hành lớn, không tương thích ngược, được phát hành vào ngày 03/12/2008 sau một thời gian dài thử nghiệm. Nhiều trong số các tính năng chính của nó đã được điều chỉnh để tương thích ngược với Python 2.6 và 2.7. Các tính năng và triết lý phát triển Python là 1 ngôn ngữ lập trình đa hình: lập trình hướng đối tượng và hướng cấu trúc được hỗ trợ đầy đủ, và có 1 số tính năng của ngôn ngữ hỗ trợ lập trình theo chức năng và lập trình hướng khía cạnh (Aspect-oriented programming). Nhiều mô hình khác được hỗ trợ bằng việc sử dụng các phần mở rộng, bao gồm thiết kế theo hợp đồng (design by contract) và lập trình luận lý. Các trang như Mozilla, Reddit, Instagram và PBS đều được viết bằng Python.

Ngôn ngữ lập trình Python được dùng vào các mục đích:

- Phát triển web (trên máy chủ)
- Phát triển phần mềm
- Tính toán một cách khoa học
- Lên kịch bản cho hệ thống

Tại Sao Nên Học Lập Trình Python?

- Python hỗ trợ nhiều nền tảng khác nhau (Windows, Mac, Linux, Raspberry Pi, etc).
- Python có cú pháp đơn giản, dễ đọc hiểu và rất gần gũi với tiếng Anh.
- Cú pháp của Python giúp lập trình viên sử dụng ít dòng code để lập trình cùng một thuật toán hơn so với các ngôn ngữ lập trình khác.
- Python sử dụng trình thông dịch để thực thi các dòng code. Do đó, những dòng code có thể được thực thi ngay lập tức mà không cần biên dịch toàn bộ chương trình. Như vậy giúp chúng ta kiểm tra code nhanh hơn.

Python cũng hỗ trợ hàm, thủ tục, hay kể cả lập trình hướng đối tượng.

Để viết mã nguồn Python, ta có thể sử dụng bất kỳ một trình soạn thảo nào, kể cả những trình soạn thảo đơn giản nhất như NotePad. Tuy nhiên, để phát triển các ứng dụng một cách hiệu quả hơn, ta nên sử dụng một IDE, để có thể tiết kiệm thời gian và công sức viết code. ở đây chúng ta sử dụng một trong những IDE thông dụng nhất để lập trình ứng dụng Python, đó là PyCharm IDE, VS Code.

2. Python Django là gì?

Django là một web framework khá nổi tiếng được viết hoàn toàn bằng ngôn ngữ Python. Nó là một framework với đầy đủ các thư viện, module hỗ trợ các web-developer. Mục tiêu chính của Django là đơn giản hóa việc tạo các website phức tạp có sử dụng cơ sở dữ liệu.

Django tập trung vào tính năng “có thể tái sử dụng” và “có thể tự chạy”, tính năng phát triển nhanh, không làm lại những gì đã làm. Django được thiết kế với triết lý làm sao để các lập trình viên đưa các ý tưởng trở thành một sản phẩm nhanh nhất có thể. Với sự kết hợp hoàn hảo đó chúng ta hoàn toàn có thể xây dựng một website bán hàng hay quản lý hàng hóa với độ chi tiết và chính xác cao.

Một số ưu điểm khi dùng Django là:

- **Nhanh:** Django được thiết kế với triết lý làm sao để các lập trình viên đưa các ý tưởng trở thành một sản phẩm nhanh nhất có thể.

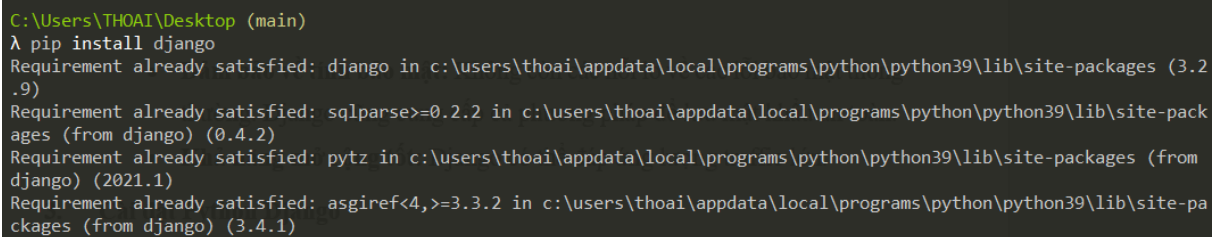
- **Có đầy đủ các thư viện/module cần thiết:** Django có sẵn các thư viện về user authentication, content admin, site maps...
- **Đảm bảo về tính bảo mật:** Không còn các nỗi lo về các lỗi bảo mật thông thường. Django cũng cung cấp cả phương pháp để lưu mật khẩu an toàn.
- **Khả năng mở rộng tốt:** Django có thể đáp ứng lượng traffic lớn.

3. Cài đặt Python Django

3.1. Cài đặt

Django là framework giúp cho việc xây dựng các website và phát triển ứng dụng web một cách dễ dàng nhanh chóng hơn, và ít code hơn. Đầu tiên chúng ta truy cập vào trang chủ của django <https://www.djangoproject.com/download/> để tìm hiểu và download. Đầu tiên chúng ta tiến hành cài đặt Django về bằng cách gõ lệnh sau:

```
pip install django
```

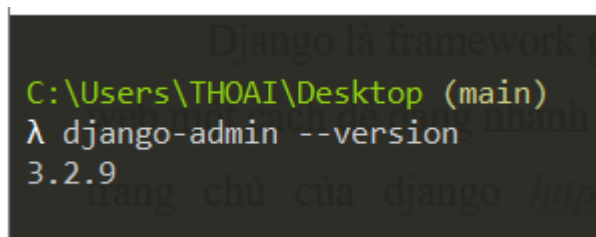


```
C:\Users\THOAI\Desktop (main)
λ pip install django
Requirement already satisfied: django in c:\users\thoai\appdata\local\programs\python\python39\lib\site-packages (3.2.9)
Requirement already satisfied: sqlparse<=0.2.2 in c:\users\thoai\appdata\local\programs\python\python39\lib\site-packages (from django) (0.4.2)
Requirement already satisfied: pytz in c:\users\thoai\appdata\local\programs\python\python39\lib\site-packages (from django) (2021.1)
Requirement already satisfied: asgiref<4,>=3.3.2 in c:\users\thoai\appdata\local\programs\python\python39\lib\site-packages (from django) (3.4.1)
```

Hình 1-1: Cài đặt python Django

Kiểm tra đã cài đặt hay chưa bằng cách gõ lệnh sau:

```
django-admin --version
```



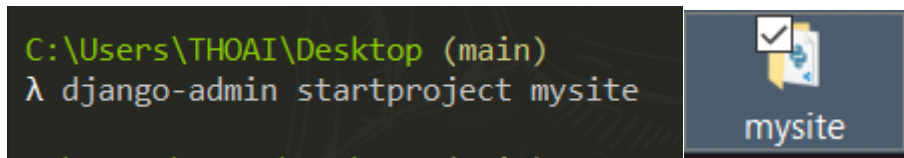
```
C:\Users\THOAI\Desktop (main)
λ django-admin --version
3.2.9
```

Hình 1-2: Kiểm tra version django

3.2. Tạo project

Bật cửa sổ CMD trên máy tính và gõ lệnh **django-admin startproject mysite**. Trong đó **django-admin startproject** là lệnh để tạo project, mysite

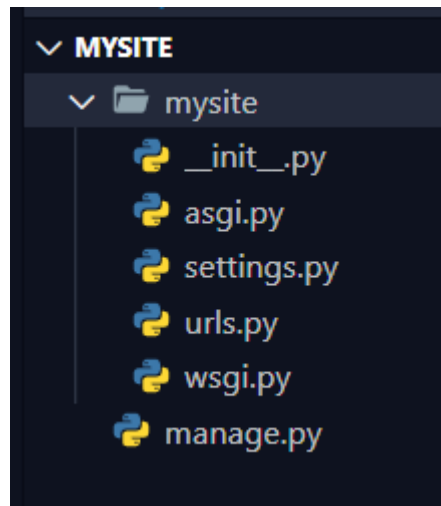
là tên của project Chúng ta cũng có thể sử dụng môi trường lập trình VSCode để làm việc.



Hình 1-3: Tạo project

Và một folder tên mysite đã được tạo ra, chứa các thành phần căn bản trong đó cho các Developer.

Lệnh startproject sẽ tạo một thư mục có tên là mysite, cấu trúc bên trong thư mục sẽ gồm các file, mỗi file sẽ để làm một số công việc khác nhau như chạy server hay cài đặt cấu hình server.



Hình 1-4: Cấu trúc project

Những file trong này đều có 1 chức năng riêng và cụ thể nó sẽ như sau:

- **__init__.py** là 1 file rỗng chỉ định việc cái đường dẫn folder này sẽ được xem như là 1 Python package.
- **settings.py** là file chứa các settings của project. Trong file này chứa các setting cơ bản như DEBUG, ALLOWED_HOSTS, INSTALLED_APP, DATABASES, ...
- **urls.py** là file khai báo các URL của project (kiểu như routing, với địa chỉ nào thì sẽ thực thi hàm nào).
- **wsgi.py** là file dùng deploy project lên server.

- **manage.py** là file để tạo app, migrate...

3.3. Cài đặt môi trường

Virtual Environment dịch nôm na là môi trường ảo. Cũng giống như máy ảo (Virtual Machine), Virtual Environment thiết lập một môi trường ảo, cho phép bạn nghịch ngợm lung tung với các packages của Python mà không làm ảnh hưởng đến những packages đã được cài đặt sẵn trên Python. Ví dụ bạn muốn thử nghiệm với Django 1.8 trong khi trên hệ thống đang cài đặt Django 1.4 LTS. Cũng giống như việc bạn dùng Virtual Machine để thử nghiệm phiên bản Chrome beta mới nhất mà không muốn làm ảnh hưởng đến phiên bản đang có trên máy.

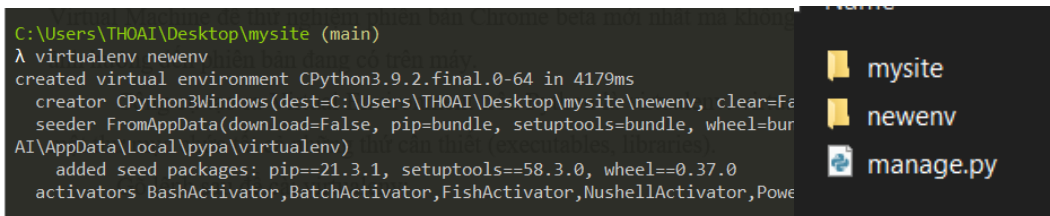
Công cụ tạo ra Virtual Environment trên Python là virtualenv. virtualenv tạo ra một thư mục chứa tất cả những thứ cần thiết (executables, libraries).

Gõ lệnh sau để cài virtualenv:

```
pip install virtualenv
```

Tại thư mục mystie chúng ta mở Command và gõ lệnh sau để cài đặt môi trường ảo cho mystie:

```
virtualenv newenv
```

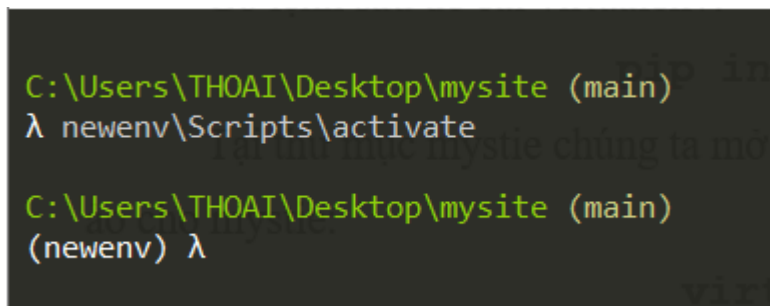


```
C:\Users\THOAI\Desktop\mysite (main)
λ virtualenv newenv
created virtual environment CPython3.9.2.final.0-64 in 4179ms
  creator CPython3Windows(dest=C:\Users\THOAI\Desktop\mysite\newenv, clear=False,
  seeder FromAppData(download=False, pip=bundle, setuptools=bundle, wheel=bur
  AI\AppData\Local\pypa\virtualenv)
    added seed packages: pip==21.3.1, setuptools==58.3.0, wheel==0.37.0
    activators BashActivator,BatchActivator,FishActivator,NushellActivator,Powe
```

Hình 1-5: Cài đặt môi trường ảo

Để activate môi trường ảo, ta gõ lệnh sau:

```
newenv\Scripts\activate
```



```
C:\Users\THOAI\Desktop\mysite (main)
λ newenv\Scripts\activate

C:\Users\THOAI\Desktop\mysite (main)
(newenv) λ
```

Hình 1-6: Đã activate môi trường ảo

3.4. Chạy server

Trong cửa sổ command đã khởi chạy môi trường ảo chúng ta chạy sever bằng file manage.py với tham số là runserver. Ta gõ lệnh sau:

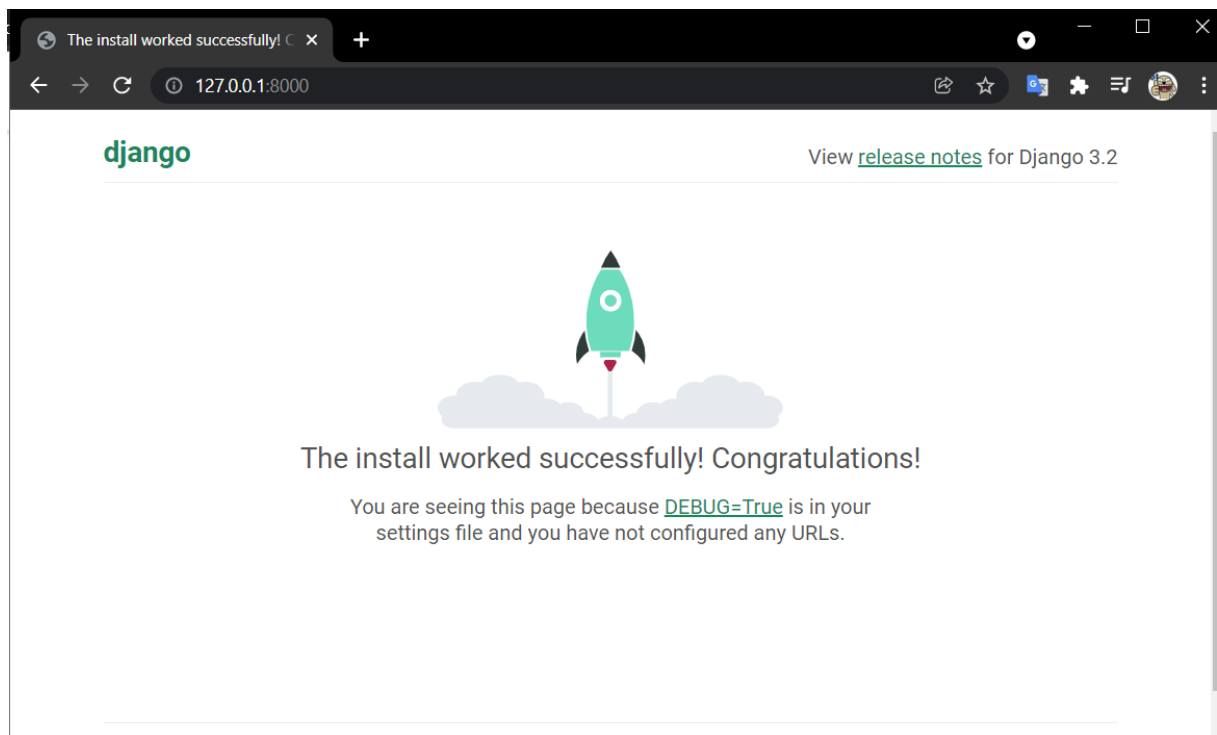
```
python manage.py runserver
```

```
C:\Users\THOAI\Desktop\tesst\mysite (main -> origin)
(newenv) λ python manage.py runserver
Watching for file changes with StatReloader
Performing system checks...

System check identified no issues (0 silenced).
November 25, 2021 - 10:24:33
Django version 3.2.9, using settings 'mysite.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CTRL-BREAK.
```

Hình 1-7: Khởi chạy server

Sau đó chúng ta truy cập vào đường dẫn <http://127.0.0.1:8000/> hoặc <http://localhost:8000/> để xem chúng ta đã chạy server thành công chưa.

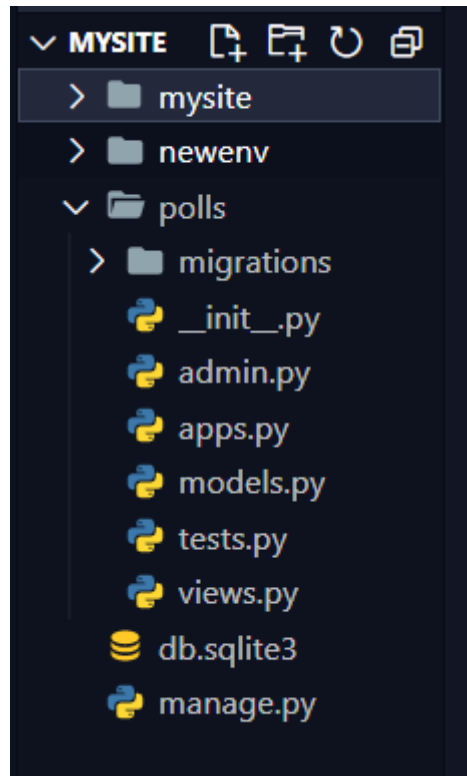


Hình 1-8: Chạy server thành công

3.5. Tạo web app

Web App là một project bao gồm nhiều app, trong đó mỗi app thực hiện một công việc riêng biệt và thư mục này chứa các file chuẩn của một ứng dụng web Django.

Tạo web app bằng lệnh `"python manage.py startapp polls"` Trong đó `"python manage.py startapp"` là lệnh để tạo web app. `"polls"` là tên app. Một thư mục với tên polls sẽ được tạo ra và có cấu trúc như sau:



Hình 1-9: Cấu trúc app

3.6. Model

Model là nguồn định nghĩa thông tin từ cơ sở dữ liệu. Nó bao gồm các lớp và các thuộc tính của dữ liệu. Thông thường, mỗi một class model tương ứng với một bảng cơ sở dữ liệu. Mỗi thuộc tính trong class sẽ tương ứng với một trường cơ sở dữ liệu.

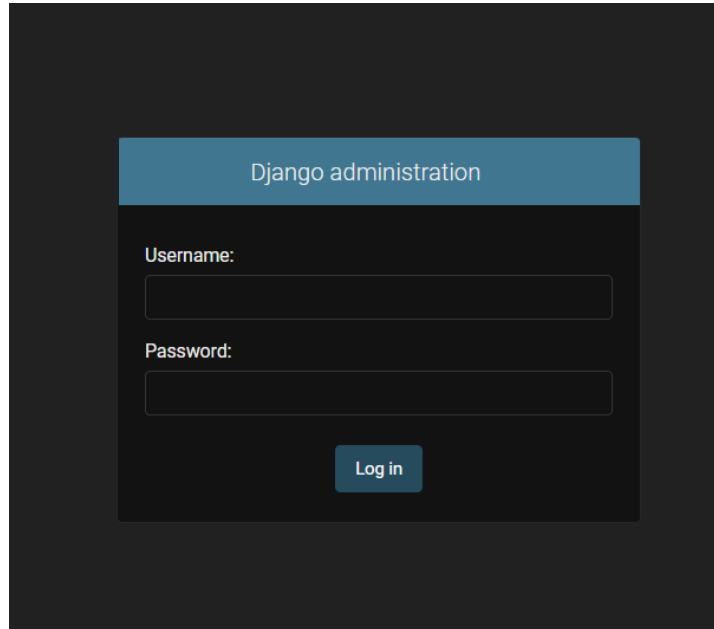
Model có thể được đồng bộ với cơ sở dữ liệu bằng lệnh:

```
python manage.py syncdb
```

3.7. Hệ thống admin

Khi viết một ứng dụng nào đó, chẳng hạn như website bán hàng, blog, web tin tức, diễn đàn...v.v. ngoài các trang hiển thị thông tin thì chúng ta còn phải xây dựng

một trang nữa là trang admin để quản lý mọi thứ, trong đó lại bao gồm nhiều trang nhỏ hơn như thêm, sửa, xóa bài viết, cài đặt trang web. Và Django cung cấp sẵn một trang admin cho riêng chúng ta bằng cách truy cập vào đường dẫn <http://127.0.0.1:8000/admin>



Hình 1-10: Giao diện đăng nhập Django administration

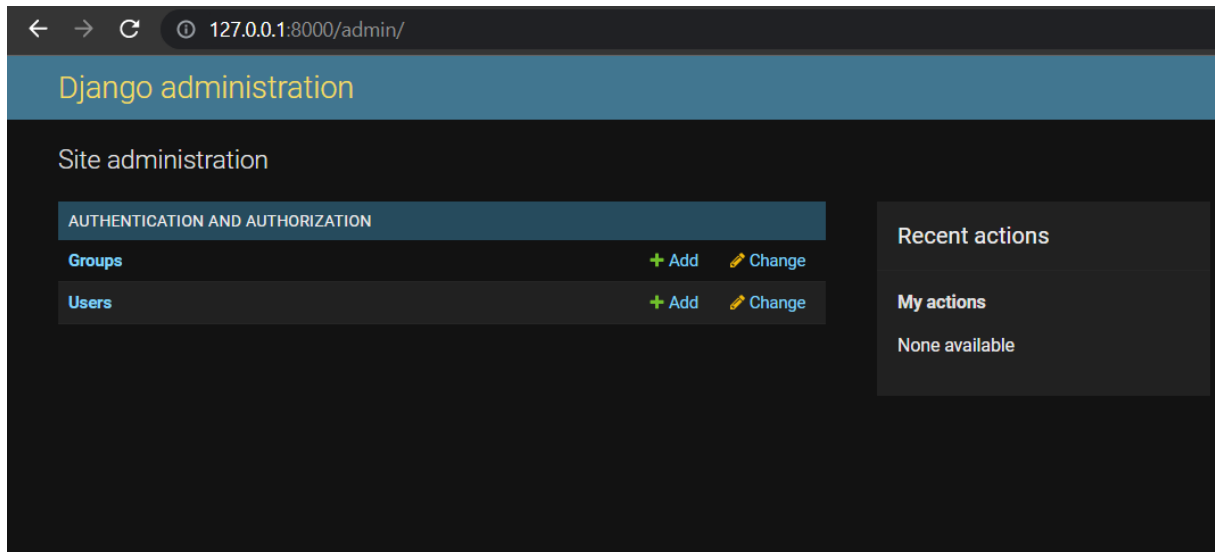
Để có tài khoản chúng ta dùng lệnh:

```
python manage.py createsuperuser
```

và tiến hành tạo tài khoản sau đó đăng nhập.

```
C:\Users\THOAI\Desktop\tesst\mysite (main)
(newenv) λ python manage.py createsuperuser
Username (leave blank to use 'thoai'): thoait
Email address: thoait.cnthongtin@gmail.com
Password:
Password (again):
This password is too short. It must contain at least 8 characters.
This password is too common.
This password is entirely numeric.
Bypass password validation and create user anyway? [y/N]: y
Superuser created successfully.
```

Hình 1-11: Tạo tài khoản thành công



Hình 1-12: Đăng nhập thành công

Giao diện admin mặc định của Django rất đơn giản, bạn có thể thực hiện thêm, sửa, xóa hoặc phân quyền hạn cho các user một cách dễ dàng.

Nếu chúng ta có tài khoản admin của django thì chúng ta có thể được quyền thêm, xóa người dùng, phân quyền người dùng cho các tài khoản.

Với quyền hạn admin chúng ta có thể cấp phép, phân quyền cho những user trong quyền hạn nhất định. Như ở đây chúng ta chỉ cho người dùng có quyền được xem câu hỏi và câu trả lời của cuộc thăm dò ý kiến, tức là người dùng này không được quyền bình chọn cũng như thêm sửa xóa.

3.8. View và templates

**View*

Trong Django thì một View là một hàm / phương thức làm một công việc cụ thể nào đó, một view thường đi kèm với một Template.

Trong Django, một trang web được tạo ra bởi các hàm View, Django sẽ chọn View nào tùy thuộc vào URL mà chúng ta đã thiết lập. Có thể bạn đã từng thấy những đường dẫn URL nhìn rất “không đẹp mắt” như:

[“ME2/Sites/dirmod.asp?sid=&type=gen&mod=Core+Pages&gid=A6CD4967199A42D9B65B1B”](#) do website tự tạo ra, Django cho phép chúng ta tạo những đường dẫn dễ nhìn hơn. Để từ một đường dẫn URL đến một View thì Django sử dụng khái niệm URLConf, đây là một module Python của Django làm nhiệm vụ phân tích đường

dẫn và chuyển đến một hàm View nhất định. Chúng ta sẽ viết thêm một số hàm View trong file `views.py`

Tiếp theo chúng ta tạo thêm các url dẫn đến từng view này ở trong file `urls.py`

Khi gõ địa chỉ lên thanh URL của trình duyệt, Django sẽ đọc biến `urlpatterns` trong file `mysite.urls` vì mặc định file này được trỏ tới trong biến `ROOT_URLCONF` trong file **`mysite/settings.py`**, các đối tượng url sẽ được đọc dần dần từ trên xuống dưới cho đến khi có một đường dẫn vừa khít với URL mà bạn nhập vào.

****Templates***

Templates là một layout được thiết kế các khung web có sẵn, ta chỉ cần thêm nội dung chính của nó vào, và nhờ các template ta mới tiết kiệm thời gian trong việc phát triển website.

Trình duyệt chỉ hiểu code HTML chứ không hiểu code Python, để có thể sử dụng code Python thì Django cung cấp cho chúng ta các thẻ template, thẻ template bắt đầu và kết thúc bằng cặp kí tự `{% %}` hoặc `{{ }}`, Các câu lệnh Python nằm trong cặp dấu `{% %}`, còn các biến thì nằm trong cặp `{{ }}`. Ngôn ngữ Template của Django được thiết kế với mục đích chính là hỗ trợ những người đã từng làm việc với HTML, do đó nếu đã từng học HTML thì sẽ không quá khó khăn để làm quen với Template.

****Đặt namespace cho URL***

Khi dùng đến URL động thì lại phát sinh một vấn đề nữa, mặc định thì Django tự động tìm các file template bên trong thư mục `template`, vậy thì giả sử khi chúng ta có thêm nhiều ứng dụng khác ngoài `polls`, chẳng hạn như một ứng dụng blog, trong đó cũng có hàm `view detail()`, và hàm view này cũng sử dụng một template tên là `detail.html`, vậy thì khi đó Django sẽ gán template của ứng dụng `polls` vào `view detail()` của ứng dụng blog, như thế sẽ báo lỗi vì ứng dụng blog sẽ không có các biến giống như `polls`.

Để giải quyết vấn đề này, chúng ta sẽ đặt namespace cho các biến url, chúng cũng giống như một cách gộp nhóm những thứ giống nhau lại với nhau. Để đặt tên namespace cho các đối tượng url thì chúng ta chỉ cần đặt giá trị cho biến `app_name` trong file `urls.py` là được.

CHƯƠNG 2. XÂY DỰNG WEB TODOLIST CƠ BẢN

1. Xây dựng khung

Website bao gồm các chức năng: đăng nhập, đăng ký, thêm, sửa, xóa, tìm kiếm công việc chưa làm, đã làm.

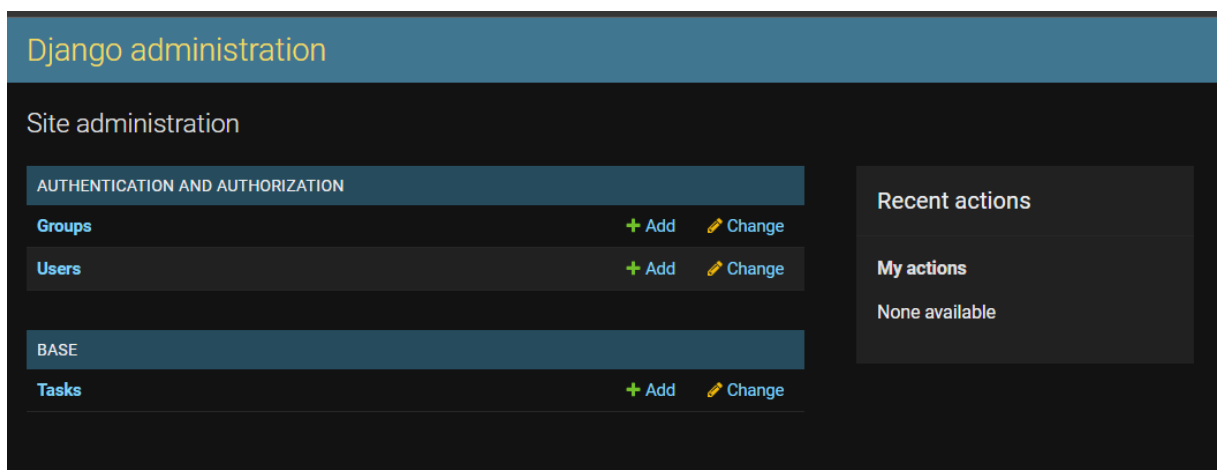
2. Hệ thống admin

Để đăng kí các bảng (hay các mô hình) với trang admin thì chúng ta cần dùng phương thức `admin.site.register()` trong file `admin.py` mà Django đã tạo cho chúng ta.

```
base > admin.py
1  from django.contrib import admin
2  from .models import Task
3
4  admin.site.register(Task)
```

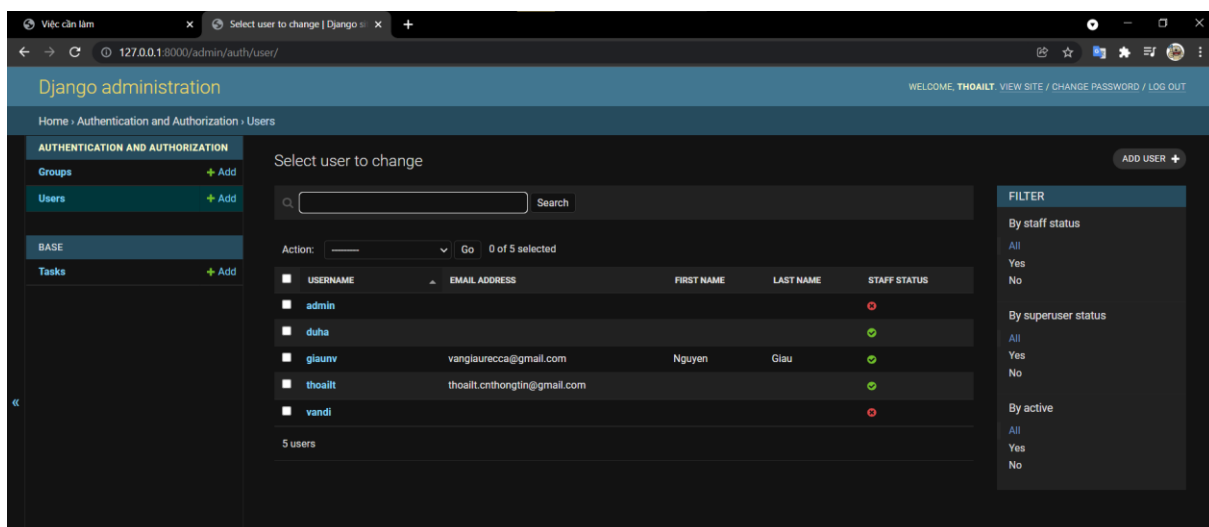
Hình 2-1: Đăng kí task trên Django admin

Sau khi đã đăng kí xong thì **Task** sẽ hiện ra trong giao diện admin. Giao diện admin mặc định của Django rất đơn giản, bạn có thể thực hiện thêm, sửa, xóa hoặc phân quyền hạn cho các user một cách dễ dàng.

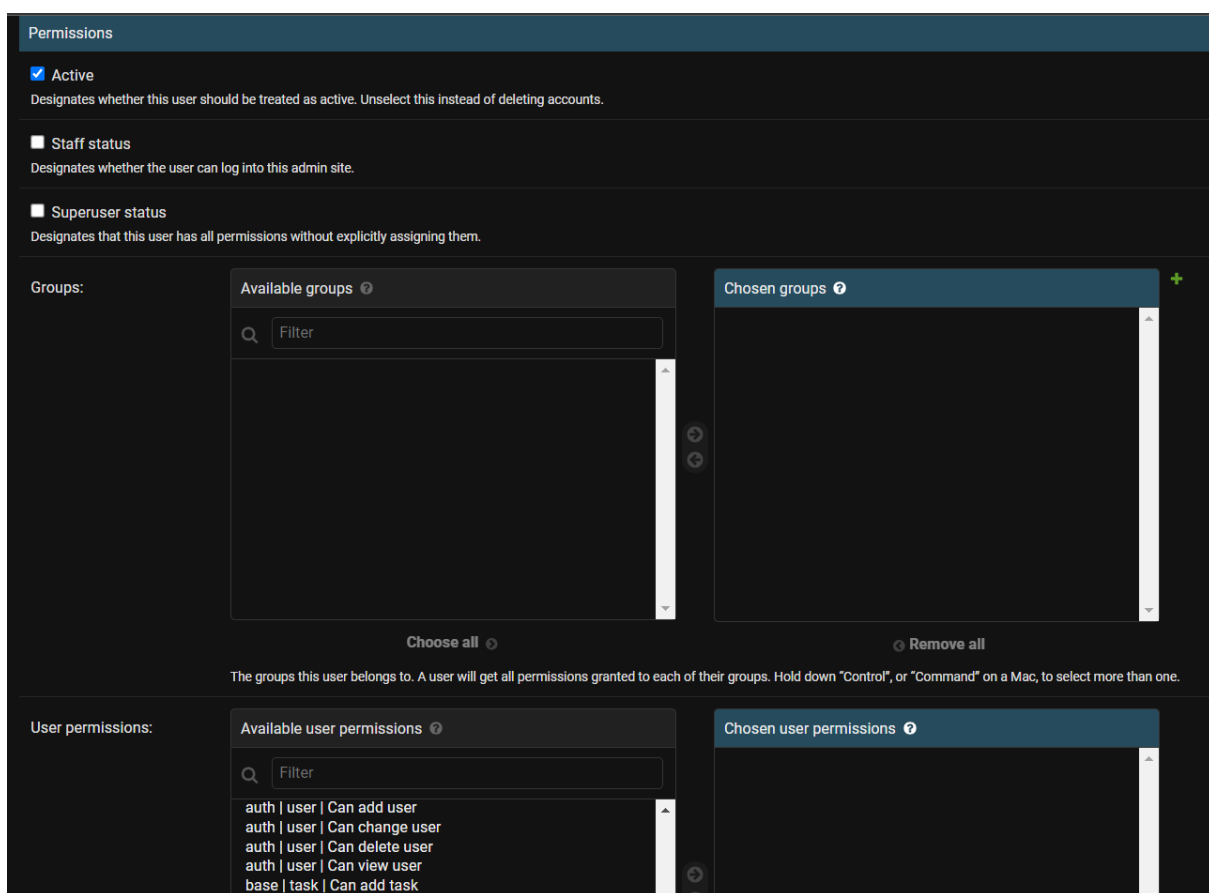


Hình 2-2: Task đã được đăng ký thành công

Nếu chúng ta có tài khoản admin của django thì chúng ta có thể được quyền thêm, xóa người dùng, phân quyền người dùng cho các tài khoản.

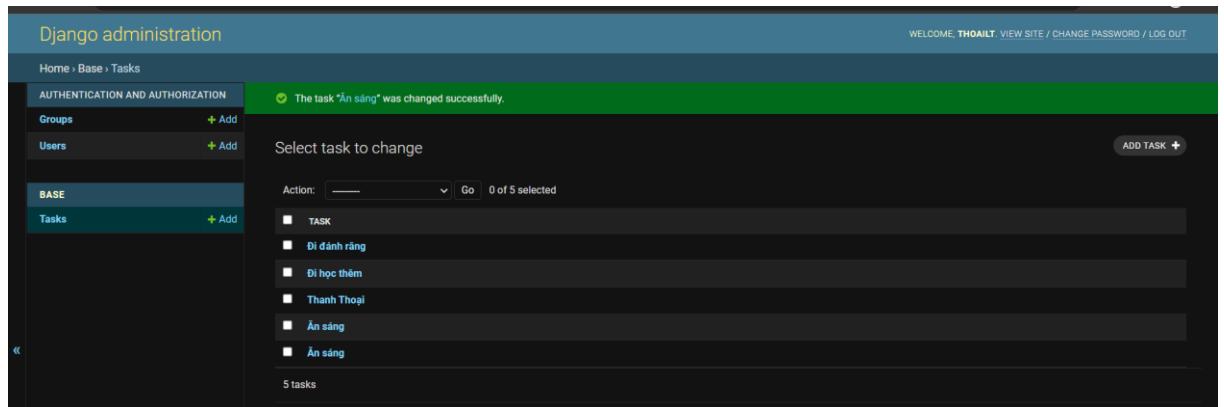


Hình 2-3: User



Hình 2-4: Phân quyền User

Với quyền hạn admin chúng ta có thể cấp phép, phân quyền cho những user trong quyền hạn nhất định. Như ở đây chúng ta chỉ cho người dùng có quyền được xem câu hỏi và câu trả lời của cuộc thăm dò ý kiến, tức là người dùng này không được quyền bình chọn cũng như thêm sửa xóa.



Hình 2-5: Chỉnh sửa Task của admin

3. Các chức năng

Khởi tạo các giá trị của todolist: tài khoản tạo todolist, tiêu đề, mô tả, trạng thái.

```
base > models.py > ...
You, 17 hours ago | 1 author (You)
1 from django.db import models
2 from django.contrib.auth.models import User
3 # Create your models here.
4
5 class Task(models.Model):
6     user = models.ForeignKey(User, on_delete=models.CASCADE, null=True, blank=True)
7     title = models.CharField(max_length=200)
8     description = models.TextField(null=True, blank=True)
9     complete = models.BooleanField(default=False)
10    created = models.DateTimeField(auto_now_add=True)
11
12    def __str__(self):
13        return self.title
14
15    class Meta:
16        ordering = ['complete']
```

Hình 2-6: Code chi tiết thêm, sửa todolist

127.0.0.1:8000/task-create/ listing directory /

THÊM CÔNG VIỆC

User: thoait

Title:

Description:

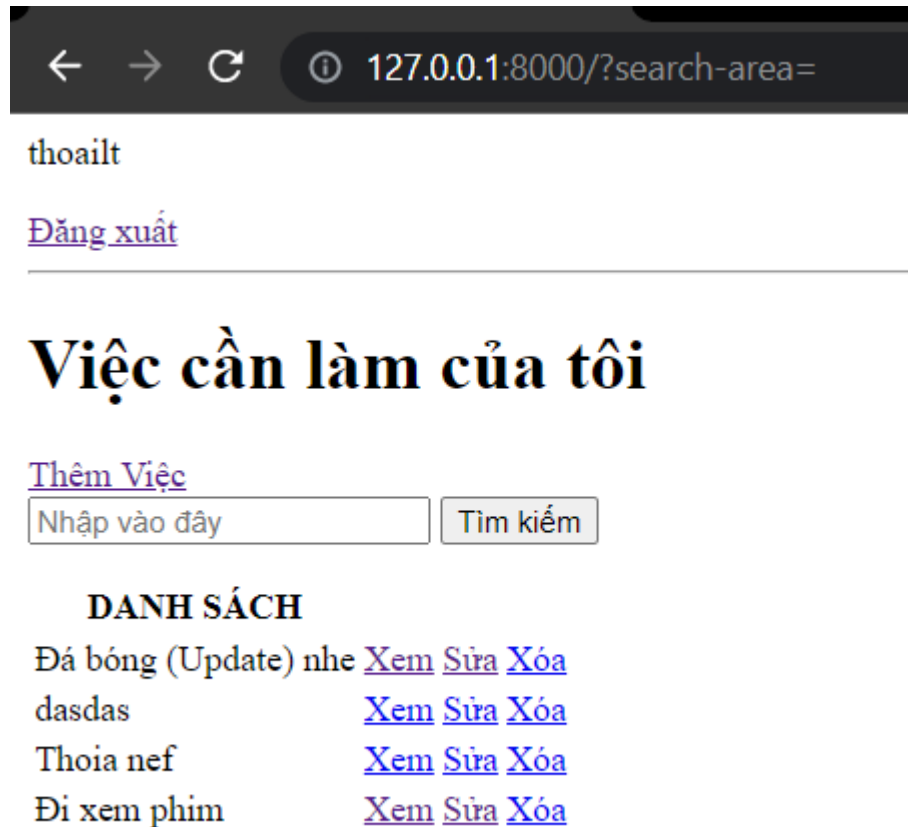
Complete: ☒

```
base > templates > base > task_form.html > form
1 <h3>THÊM CÔNG VIỆC</h3>
2
3 <form method="POST" action="">
4     {% csrf_token %}
5     {{form.as_p}}
6     <input type="submit" value="Thêm">
7 </form>
```

Hình 2-7: Giao diện & code thêm công việc

```
base > templates > base > task_list.html > form > input
1
2 {% if request.user.is_authenticated %}
3     <p>{{request.user}}</p>
4     <a href="{% url 'logout' %}">Đăng xuất</a>
5 {% else %}
6     <a href="{% url 'login' %}">Đăng nhập</a>
7 {% endif %}
8 <hr>
9
10 <h1>Việc cần làm của tôi {{color}}</h1>
11 <a href="{% url 'task-create' %}">Thêm Việc</a>
12
13 <form action="" method="GET">
14     <input type="text" name="search-area" placeholder="Nhập vào đây" value="{{search_input}}">
15     <input type="submit" value="Tìm kiếm">
16 </form>
17
18 <table>
19     <tr>
20         <th>DANH SÁCH</th>
21     </tr>
22     <tr>
23         <td>
24             {%for task in tasks %}
25             <tr>
26                 <td>{{task.title}}</td>
27                 <td><a href="{% url 'task' task.id %}">Xem</a></td>
28                 <td><a href="{% url 'task-update' task.id %}">Sửa</a></td>
29                 <td><a href="{% url 'task-delete' task.id %}">Xóa</a></td>
30             </tr>
31             {% empty %}
32                 <h3>Không có việc nào</h3>
33             {% endfor %}
34 </table>
```

Hình 2-8: Code giao diện chính



Hình 2-9: Giao diện home todolis

Ta sử dụng templates của Django để xây dựng giao diện đăng nhập thông qua html, Cung cấp các templates dựa trên các cú pháp thân thiện của template language, có thể mở rộng để tách riêng rẽ thiết kế, nội dung và mã code python. Nó là Django's template layer sử dụng template language giúp cân bằng giữa việc dễ sử dụng và công năng hiệu quả của nó. Nó được thiết kế sử dụng tiện lợi như làm việc với HTML và dễ làm quen kể cả với những người vốn dùng các template language khác như Smarty hay Cheetah Template.

```
> templates > base > login.html > p > a
<h1>Đăng nhập</h1>

<form action="" method="POST">
    {% csrf_token %}
    {{ form.as_p }}
    <input type="submit" value="Đăng nhập">
</form>

<p>Bạn chưa có tài khoản? <a href="{% url 'register' %}">Tạo tài khoản</a></p>
```

Hình 2-10: Code đăng nhập

```
base > templates > base > register.html > p
1 <h1>Đăng ký</h1>
2
3 <form method="POST">
4     {% csrf_token %}
5     {{ form.as_p }}
6     <input type="submit" value="Đăng ký">
7 </form>
8
9 <p>Bạn đã có tài khoản? <a href="">Đăng nhập</a></p>
```

Hình 2-11: Code đăng ký

```
base > templates > base > task_confirm_delete.html > form
1 <a href="{% url 'tasks' %}">Trở lại</a>
2 <form method="POST">
3     {% csrf_token %}
4     <p>Bạn có chắc muốn xóa việc này không? "{{task}}"</p>
5     <input type="submit" value="Xóa">
6 </form>
```

Hình 2-12: Code xóa việc

File main.html được kế thừa (extends) từ views.py để hiển thị các nội dung (phần động) mà chúng ta muốn hiển thị lên cũng như tương tác với nó.

```
urlpatterns = [
    path('login/', CustomLoginView.as_view(), name='login'),
    path('logout/', LogoutView.as_view(next_page='login'), name='logout'),
    path('register/', RegisterPage.as_view(), name='register'),
    path('', TaskList.as_view(), name='tasks'),
    path('task/<int:pk>/', TaskDetail.as_view(), name='task'),
    path('task-create/', TaskCreate.as_view(), name='task-create'),
    path('task-update/<int:pk>/', TaskUpdate.as_view(), name='task-update'),
    path('task-delete/<int:pk>/', DeleteView.as_view(), name='task-delete'),
]
```

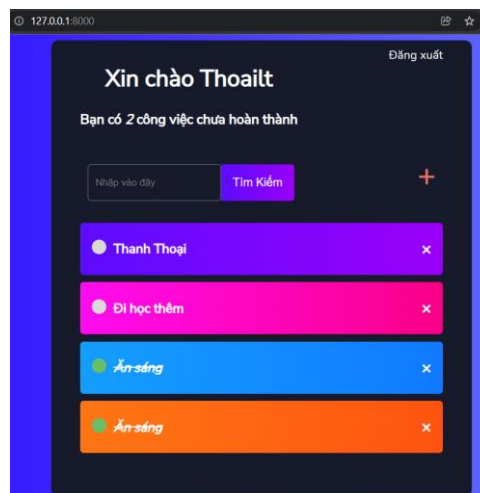
Hình 2-13: Url patterns

Các đường dẫn (url) được khai báo trong file `urls.py`, người dùng có thể thêm, xóa hoặc tùy chỉnh các url theo ý muốn của mình tại đây. URL được định nghĩa bằng đường dẫn của nó và hàm view mà nó ánh xạ đến.

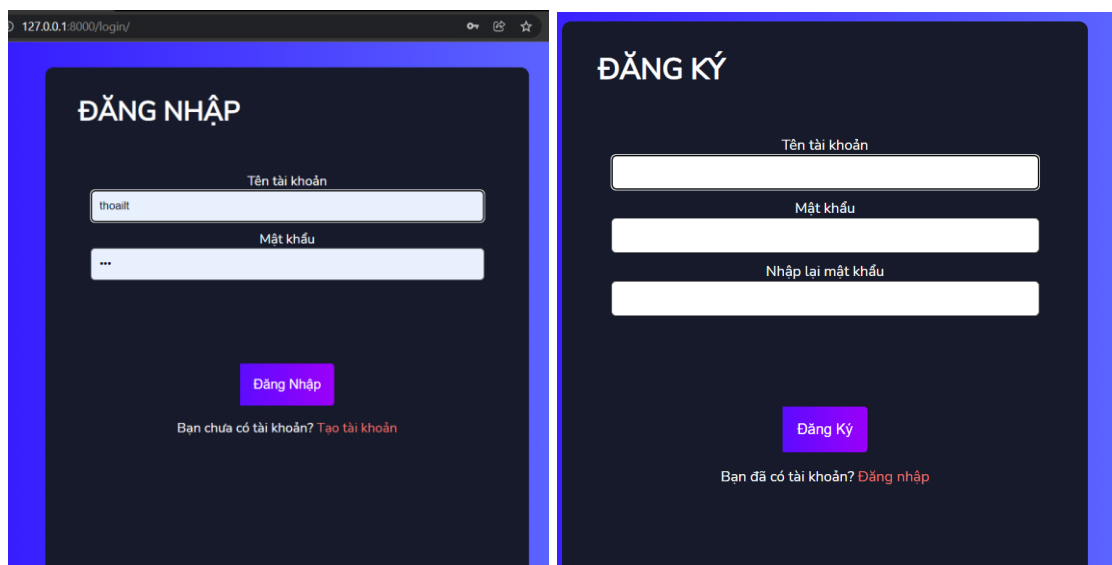
Trong trường hợp, ta muốn với mỗi url cùng tham số truyền sau sẽ trả về cho ta một kết quả khác mà không muốn viết lại nhiều lần định nghĩa url cho từng tham số, có một cách hữu hiệu giải quyết việc này đó là sử dụng URL động.

4. Sản phẩm

Sau khi đã hoàn thành load static xong thì chúng ta truy cập vào địa chỉ <http://127.0.0.1:8000/> để xem kết quả. Web site đã hiển thị được với một giao diện đẹp mắt hơn. Giao diện sau khi đã áp dụng thêm CSS:



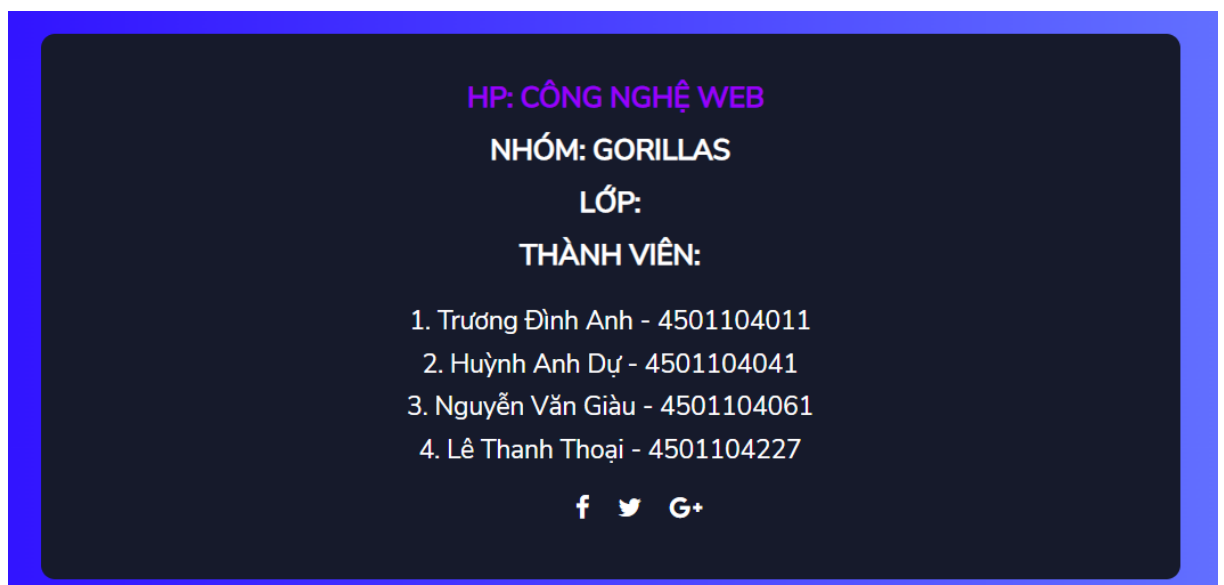
Hình 2-14: Giao diện chính



Hình 2-15: Giao diện đăng nhập và đăng ký



Hình 2-16: Giao diện Thêm, Sửa, Xóa việc



Hình 2-17: Giao diện footer thông tin nhóm

Link Github project: [https://github.com/Caption17/Todolist Django](https://github.com/Caption17/Todolist_Django)

CHƯƠNG 3. KẾT LUẬN

Sau thời gian tham khảo tìm tòi và dưới sự chỉ bảo của thầy hướng dẫn về bài tập đồ án Tìm hiểu framework Python Django và ứng dụng phát triển ứng dụng web với Django trong khoảng thời gian nhất định dành cho việc thực hiện đề tài, nên một số vấn đề vẫn chưa được hoàn chỉnh. Vì đề tài chỉ nằm ở mức tìm hiểu về cơ bản công nghệ web này nên sản phẩm demo của chúng em chỉ đơn giản và nằm ở mức tìm hiểu cơ bản.

Vì giới hạn thời gian và tập trung nhiều hơn cho việc thực hiện đề tài cuối kì, nên nhóm chúng em chưa tìm hiểu thật sự kĩ về framework này. Chúng em sẽ cố gắng tìm tòi, học hỏi nhiều hơn để có thêm kiến thức và trang bị thêm nhiều kĩ năng cũng như nâng cao, cải thiện kĩ năng lập trình, thiết kế trở nên tốt hơn.

Chúng em chân thành cảm ơn.

TÀI LIỆU THAM KHẢO

- [1] Tango With Django: A beginner's Guide to Web Development With Python / Django 1.9 by Leif Azzopardi & David Maxwell - 2017
- [2] <https://djangobook.com/>
- [3] <https://www.dennisivy.com/post/django-class-based-views/>
- [4] <https://docs.djangoproject.com/en/3.2/>

-HẾT-