**Install & Setup Django**

**Make sure you have Python 3.7 :** <https://www.python.org/downloads/release/python-376/>

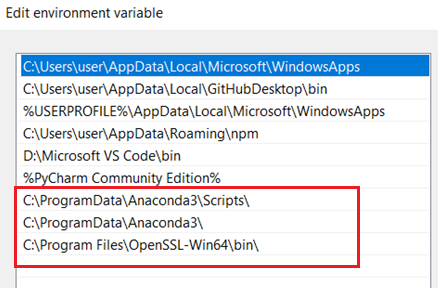
**Install VS Code or PyCharm**

**Install Anaconda for Python 3.7 :** <https://www.anaconda.com/distribution/>

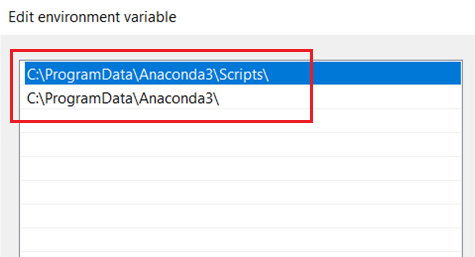
**Install Win64 OpenSSL v1.1.1d** **:** <https://slproweb.com/products/Win32OpenSSL.html>

**Add to Environment Variable:**

* Existing variable “Path” :



* New variable name “conda” :

****

**In terminal run following Commands :-**

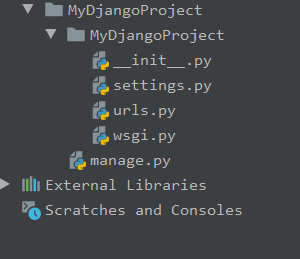
* conda create --name MyDjangoEnv python=3.7
* conda info --envs
* conda activate MyDjangoEnv
* conda install django

**Create Django Project**

**In terminal run following Commands :-**

* django-admin startproject MyDjangoProject
* cd MyDjangoProject
* python manage.py runserver

Visit <http://127.0.0.1:8000/> in Chrome

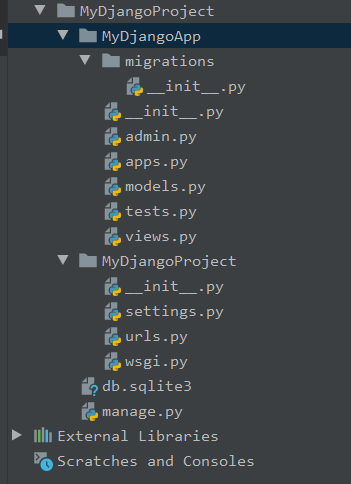


|  |  |
| --- | --- |
| **\_\_init\_\_.py** | Blank python script. Allows directory to be used as a package |
| **settings.py** | Store our project settings |
| **urls.py** | URLs pattern of all the different pages of our web application |
| **wsgi.py** | Web Server Gateway Interface. Helps in deploying apps to production |
| **manage.py** | Associates with many commands as we build our web application |

**Create Django App**

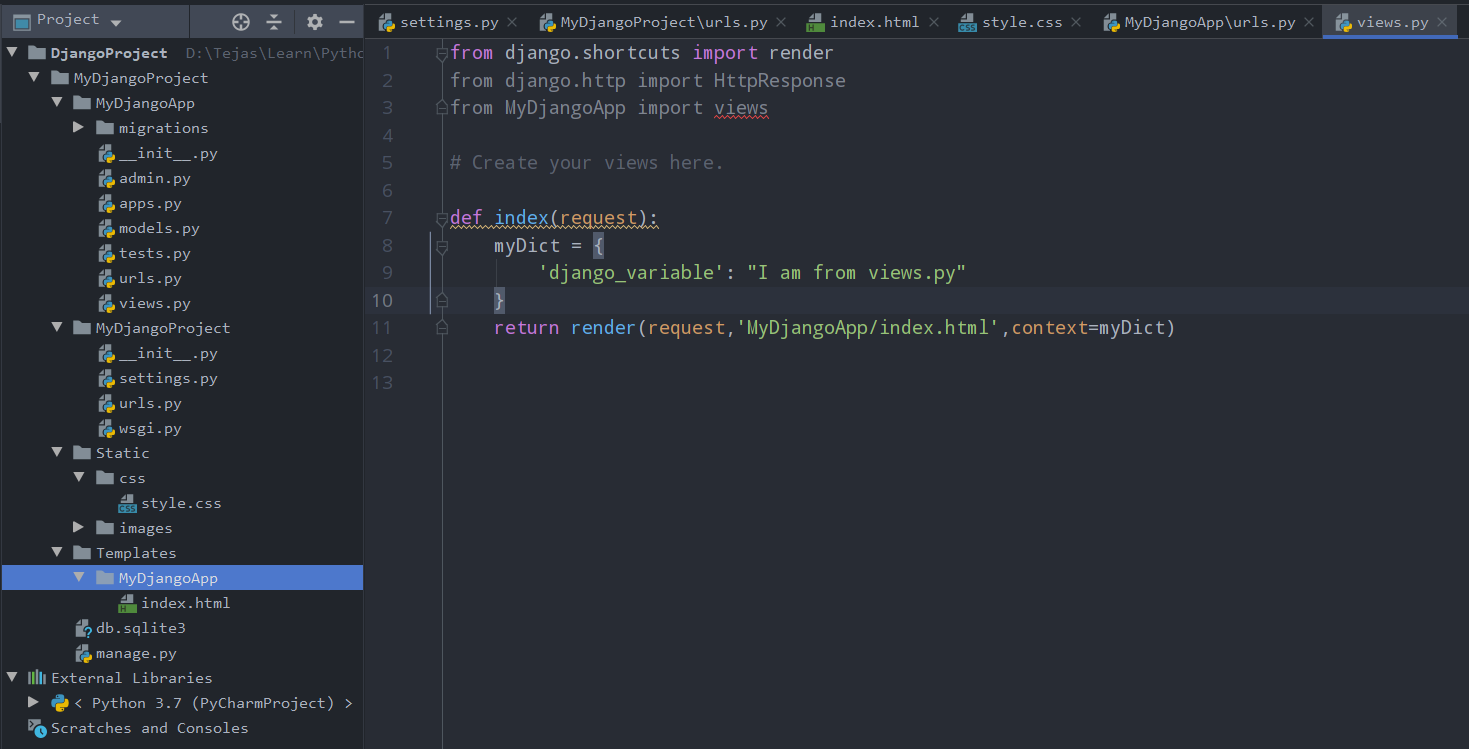
**In terminal run following Commands :-**

* python manage.py startapp MyDjangoApp

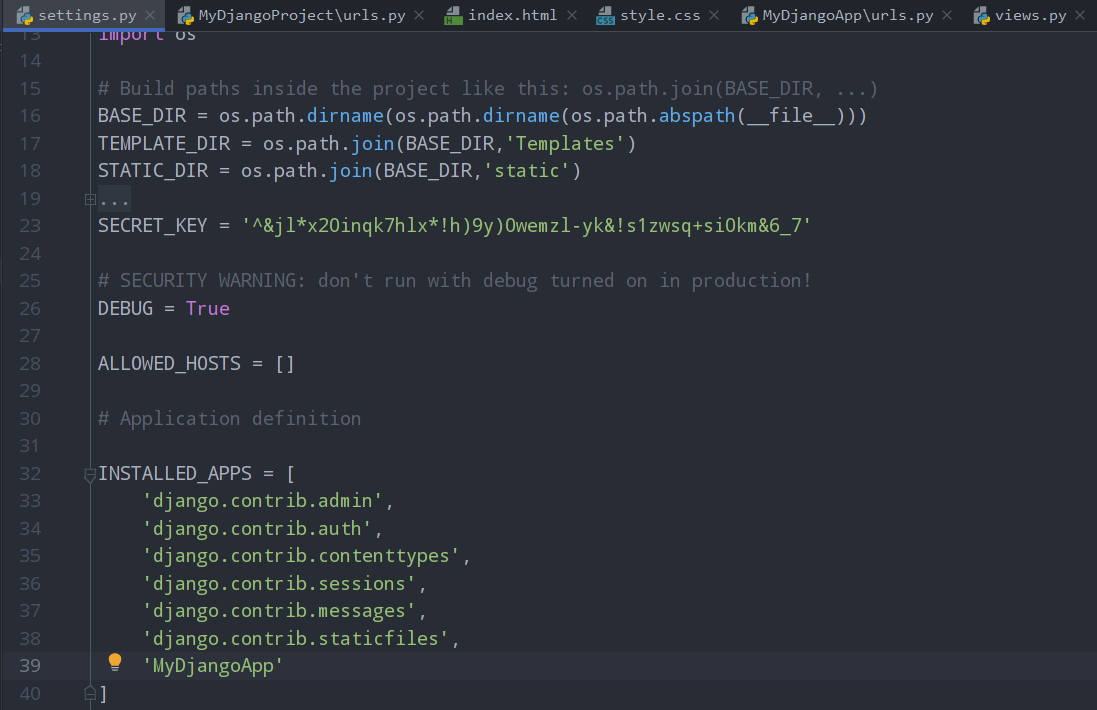


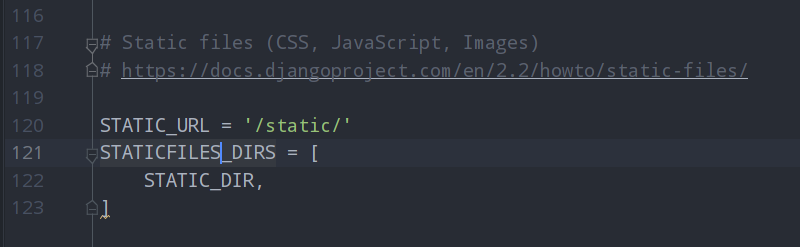
|  |  |
| --- | --- |
| **\_\_init\_\_.py** | Blank python script. Allows directory to be used as a package |
| **admin.py** | Register all our models here, which Django will use later with interface |
| **app.py** | Contains application specific configurations |
| **models.py** | Contains the application’s data models |
| **views.py** | Contains functions that handles requests and responses |
| **migrations** | Contains database specific information as it related to the models |

Views.py

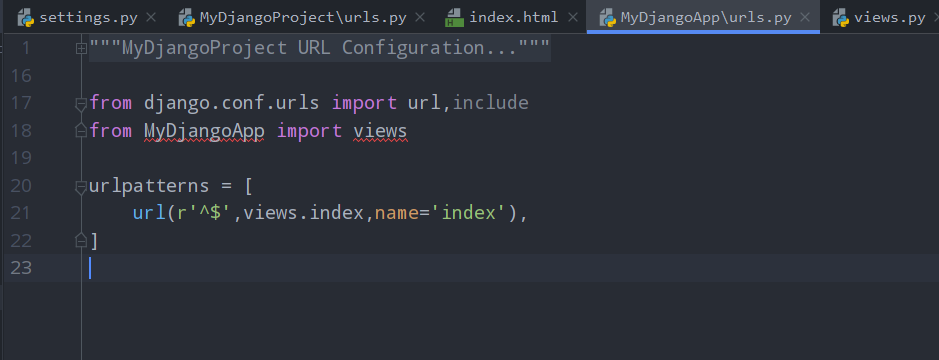


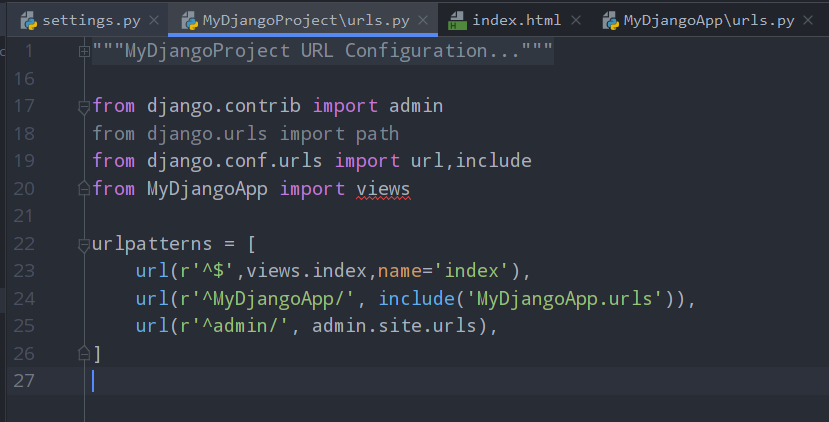
settings.py



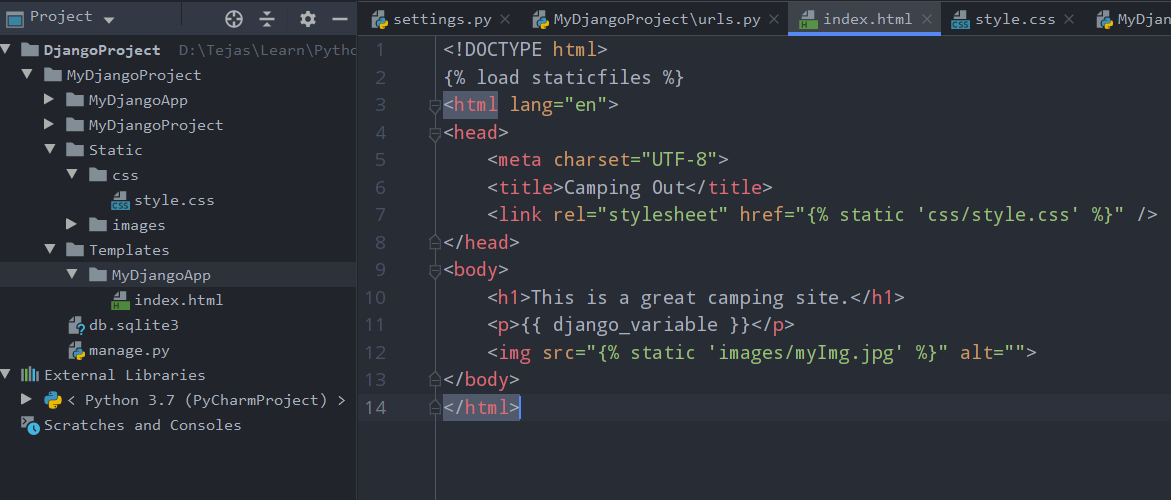


urls.py

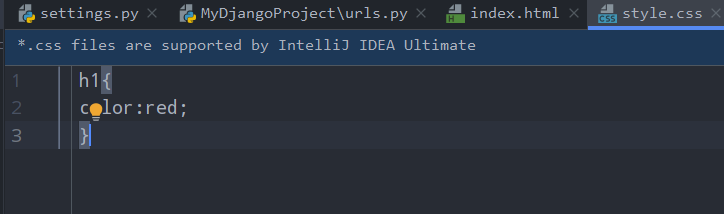




index.html



style.css

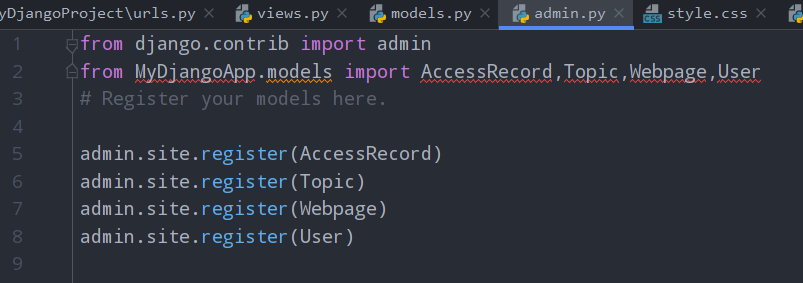


**Create Models**

models.py

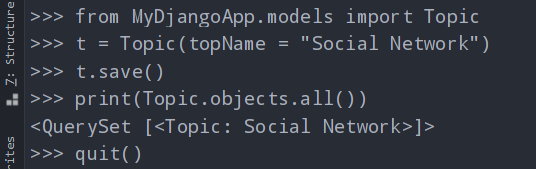
from django.db import models  
  
  
class Topic(models.Model):  
 topName = models.CharField(max\_length=264,unique=True)  
  
 def \_\_str\_\_(self):  
 return self.topName  
  
  
class Webpage(models.Model):  
 topic = models.ForeignKey(Topic, on\_delete=True)  
 name = models.CharField(max\_length=264,unique=True)  
 url = models.URLField(unique=True)  
  
 def \_\_str\_\_(self):  
 return self.name  
  
  
class AccessRecord(models.Model):  
 name = models.ForeignKey(Webpage, on\_delete=True)  
 date = models.DateField()  
  
 def \_\_str\_\_(self):  
 return str(self.date)  
  
  
class User(models.Model):  
 fname = models.CharField(max\_length=264)  
 lname = models.CharField(max\_length=264)  
 email = models.EmailField(max\_length=264,unique=True)  
  
 def \_\_str\_\_(self):  
 return str(self.fname) + " " + str(self.lname)

admin.py

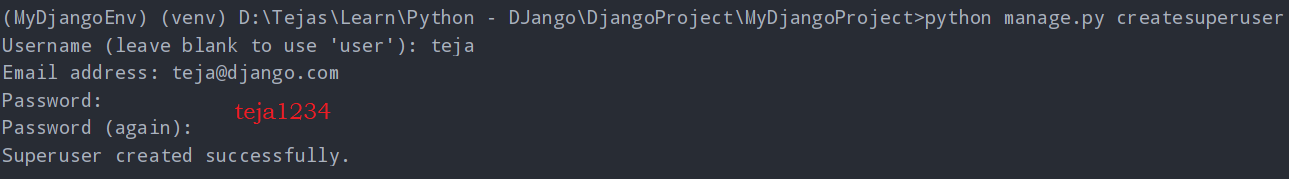


**In PyCharm terminal run following Commands :-**

* conda activate MyDjangoEnv
* python manage.py startapp MyDjangoApp
* python manage.py migrate
* python manage.py makemigrations MyDjangoApp
* python manage.py migrate
* python manage.py shell



* python manage.py createsuperuser



* python manage.py runserver

Visit <http://127.0.0.1:8000/>admin in Chrome

**Model – Template - View**

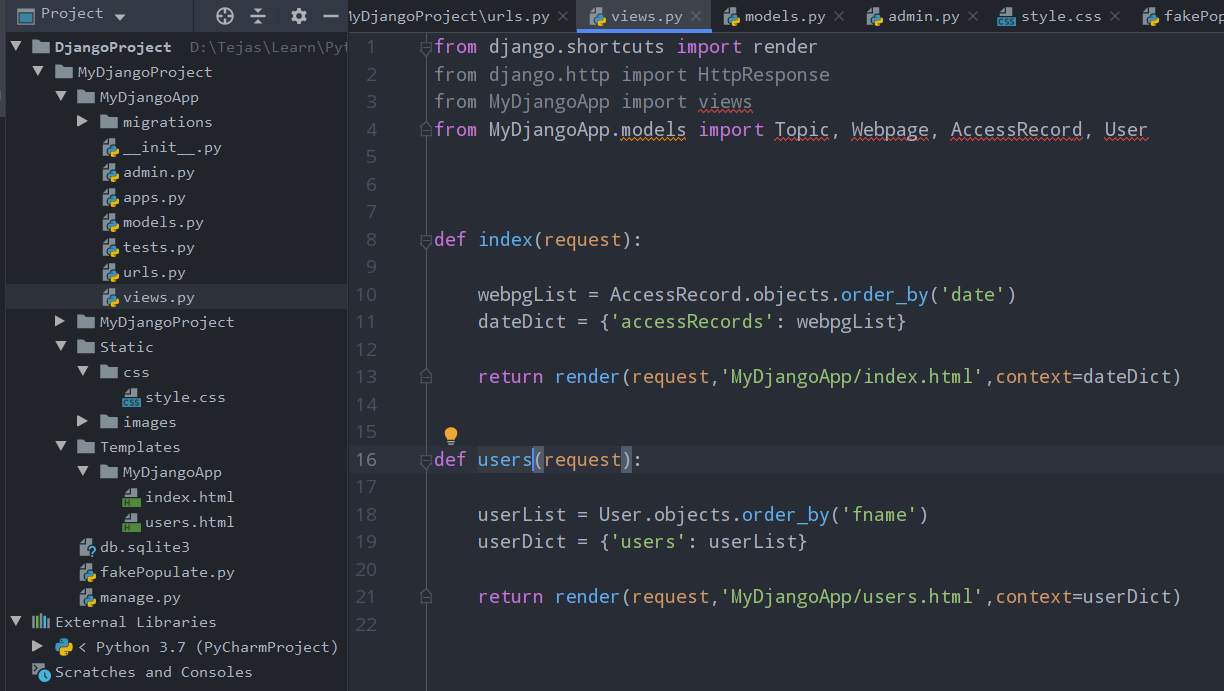
Steps to achieving the goal of serving dynamic content to a user based off the connection of the models, views, and templates:

1. In the views.py file we import any models that we will need to use.
2. Use the view to query the model for data that we will need
3. Pass results from the model to the template
4. Edit the template so that it is ready to accept and display the data from the model.
5. Map a URL to the view

**In PyCharm terminal run following Commands :-**

* pip install faker
* set DJANGO\_SETTINGS\_MODULE=MyDjangoProject.settings

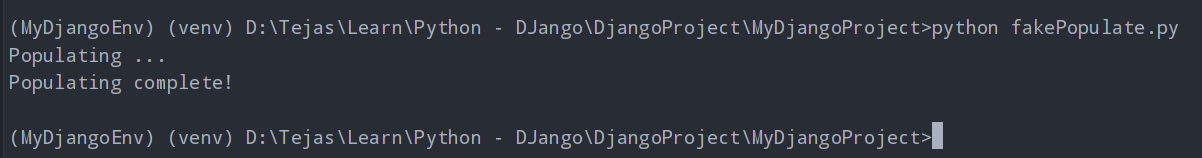
views.py



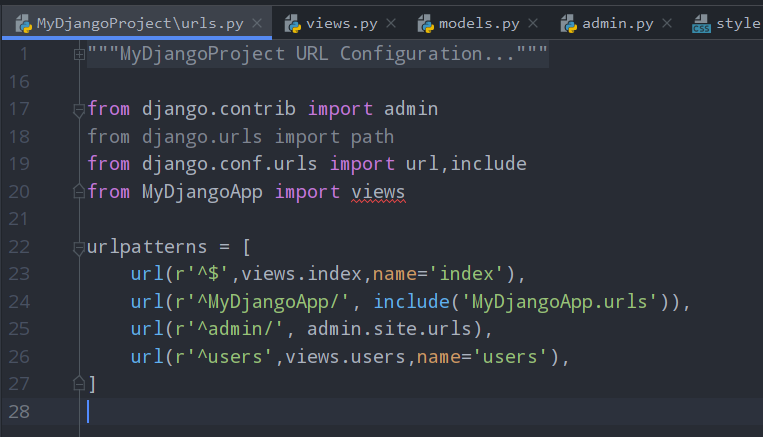
fakePopulate.py

import os  
os.environ.setdefault('DJANGO\_SETTINGS\_MODULE','MyDjangoProject.settings')  
  
import django  
django.setup()  
  
import random  
from MyDjangoApp.models import AccessRecord,Webpage,Topic,User  
from faker import Faker  
  
fakegen = Faker()  
topics = ['Search', 'Social', 'MarketPlace', 'News', 'Games']  
  
def addTopic():  
 t = Topic.objects.get\_or\_create(topName=random.choice(topics))[0]  
 t.save()  
 return t  
  
def populate(N=5):  
  
 for entry in range(N):  
  
 topic = addTopic()  
 fakeURL = fakegen.url()  
 fakeDate = fakegen.date()  
 fakeName = fakegen.name()  
 fakeEmail = fakegen.email()  
  
 webpg = Webpage.objects.get\_or\_create(topic=topic, url=fakeURL, name=fakeName)[0]  
  
 accRec = AccessRecord.objects.get\_or\_create(name=webpg, date=fakeDate)[0]  
 usr = User.objects.get\_or\_create(fname=fakeName.split()[0], lname=fakeName.split()[1], email=fakeEmail)[0]  
  
if \_\_name\_\_ == '\_\_main\_\_':  
 print('Populating ...')  
 populate(20)  
 print('Populating complete!')

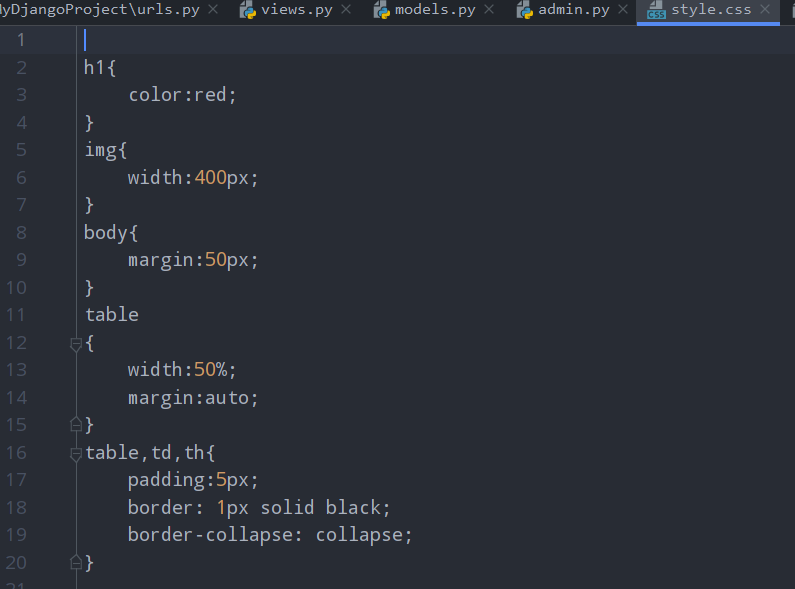
* python fakePopulate.py



urls.py



style.css



index.html

<!DOCTYPE html>

{% load staticfiles %}

<html lang="en">

<head>

    <meta charset="UTF-8">

    <title>Camping Out</title>

    <link rel="stylesheet" href="{% static 'css/style.css' %}" />

</head>

<body>

    <div>

        <h1>This is a great camping site.</h1>

        <img src="{% static 'images/myImg.jpg' %}" alt="">

    </div>

    <div class="accRecords">

        <h1>Access records:-</h1>

        {% if accessRecords %}

        <table>

            <thead>

                <th>Site Name</th>

                <th>Data Accessed</th>

            </thead>

            {% for acc in accessRecords %}

            <tr>

                <td>{{ acc.name }}</td>

                <td>{{ acc.date }}</td>

            </tr>

            {% endfor %}

        </table>

        {% else %}

        <p>No Access Records found!</p>

        {% endif %}

    </div>

</body>

</html>

users.html

<!DOCTYPE html>

{% load staticfiles %}

<html lang="en">

<head>

    <meta charset="UTF-8">

    <title>Users</title>

    <link rel="stylesheet" href="{% static 'css/style.css' %}" />

</head>

<body>

    <div class="Users">

        <h1>All Users:-</h1>

        {% if users %}

        <ol>

            {% for user in users %}

            <li>User list : </li>

            <ul>

                <li>First Name: {{ user.fname }}</li>

                <li>Last Name: {{ user.lname }}</li>

                <li>Email ID: {{ user.email }}</li>

            </ul>

            {% endfor %}

        </ol>

        {% else %}

        <p>No Users found!</p>

        {% endif %}

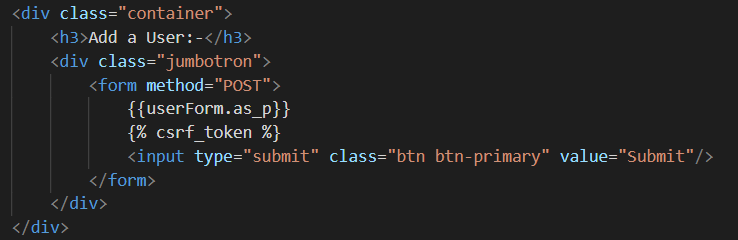
    </div>

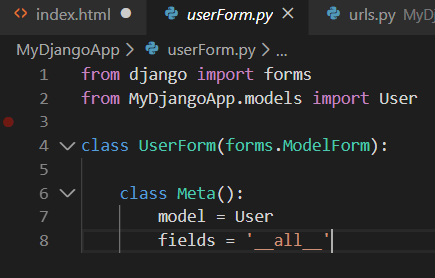
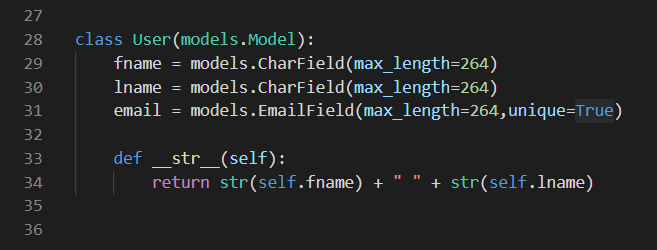
</body>

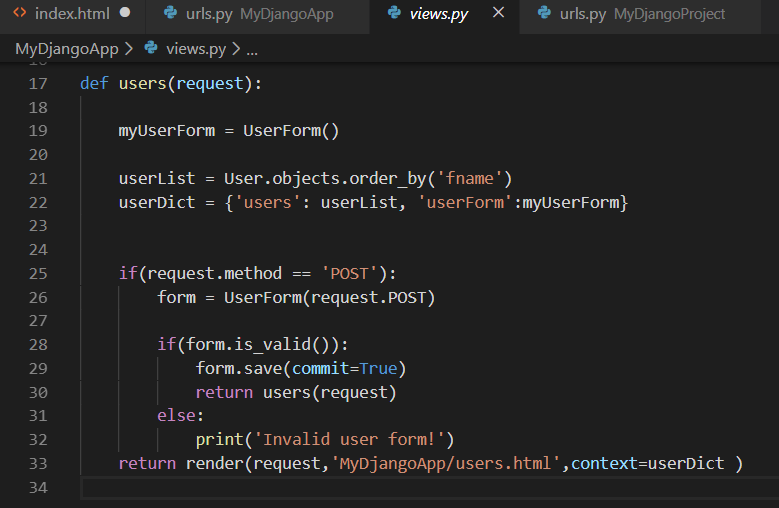
</html>

* python manage.py runserver
* Visit <http://127.0.0.1:8000/>users in Chrome

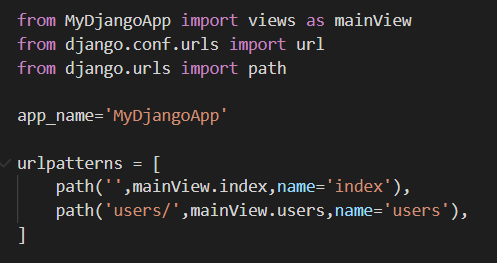
**User form**

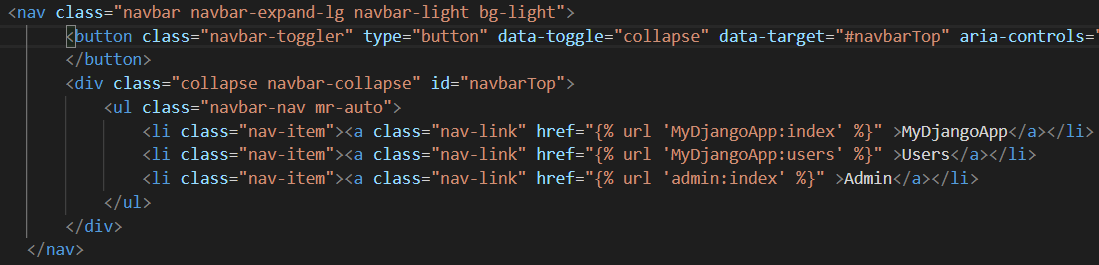




**Relative URLs**





**HTML Template inheritance**



**Custom filters**

