Python django

BY: JUNIELL JUSTINE EARL M. CORONADO

Setting up the environment

- First install the latest version of the python and Django. You can easily install them using pip install command. And then the next thing is that you have to install the pipenv so that you can put Django in your project directory.

Setting up the environment

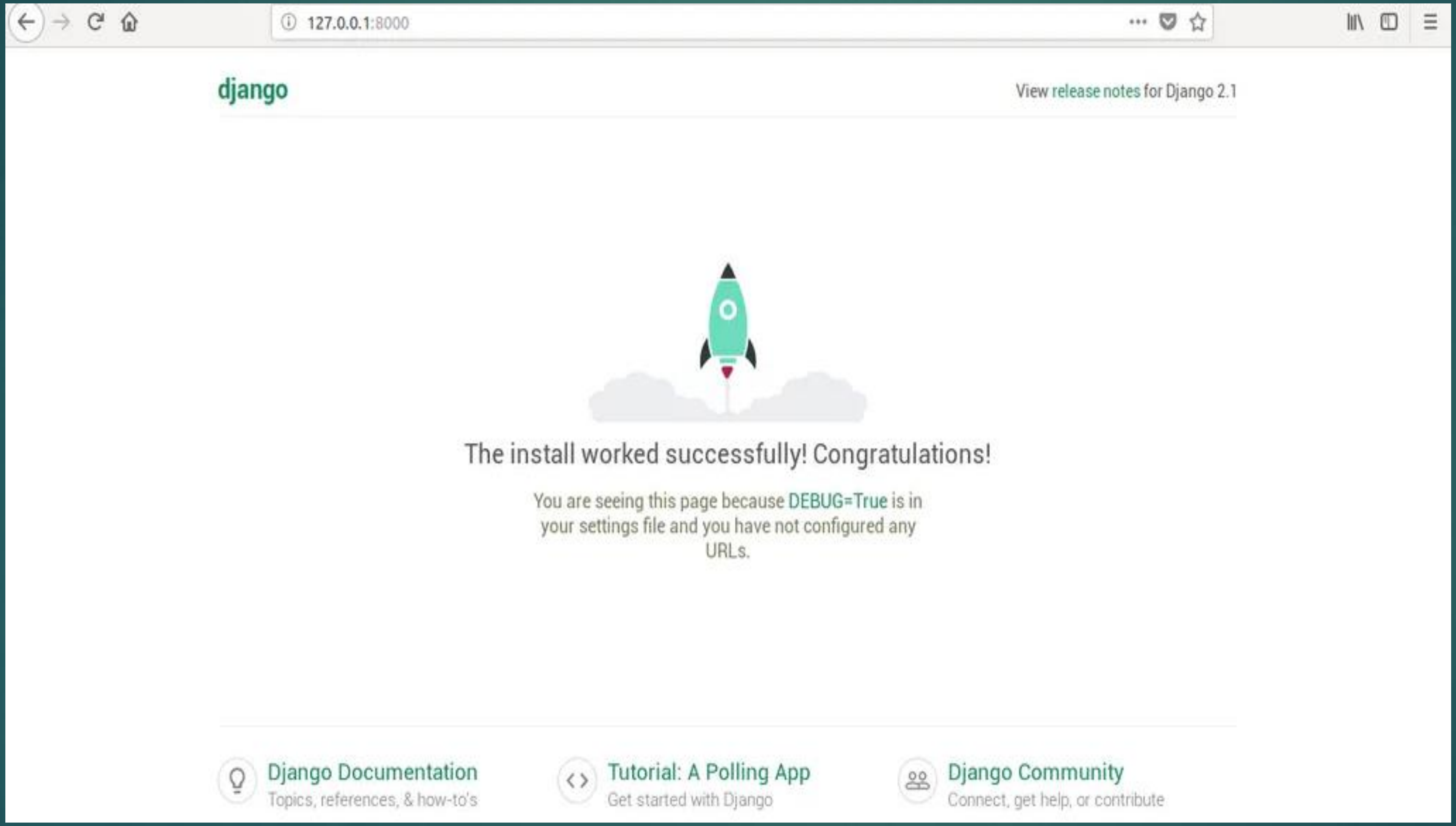
- Open the terminal and navigate to a designated directory (eg. test site).
- In the terminal type mkdir “folder name” (eg. Mkdir site).
- And install Django type and type Python manage.py startproject django_project
- to avoid to many same directory just type django-admin startproject file name and add a dot to the last part of the command .
- And also we want to add an app folder type Python manage.py startapp blog
- And after that we have to run our server so first we have to type Python manage.py runserver.
- And look for an http address and copy it and run it on your browser.
- Note: carefull with the url it will not run if you put s on the https:// unless you modify if but do not worry it is just for beginner.

- For pro: you can change the port to what you like just type python manage.py runserver 5000.

You can set it if you are using the same local ports like viewing html on a browser or etc.

As the previous process is completed and working the server is functioning

and showing a website with something like this. And if it does let begin the creation


127.0.0.1:8000

django

View release notes for Django 2.1

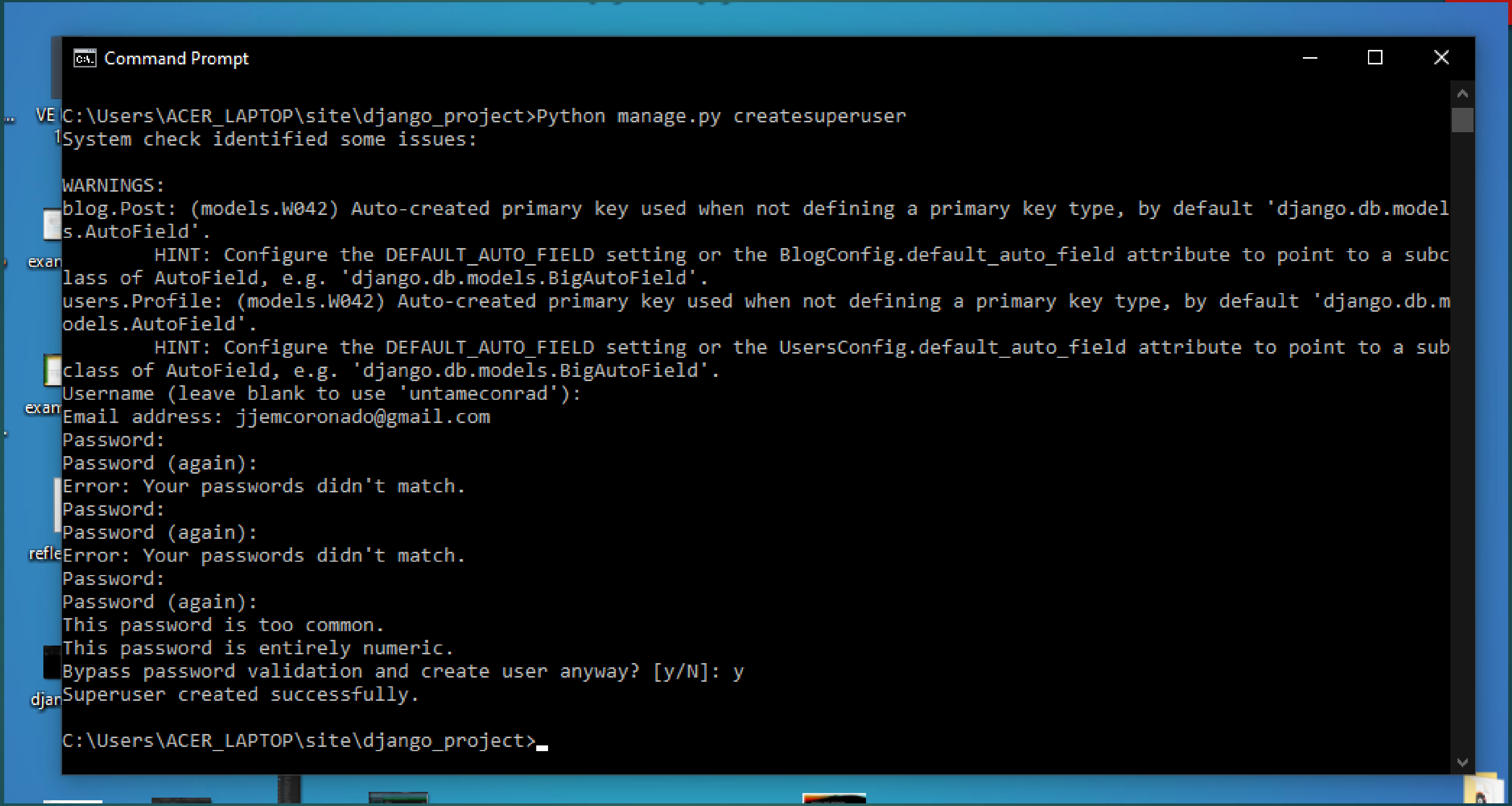
The install worked successfully! Congratulations!

You are seeing this page because DEBUG=True is in your settings file and you have not configured any URLs.

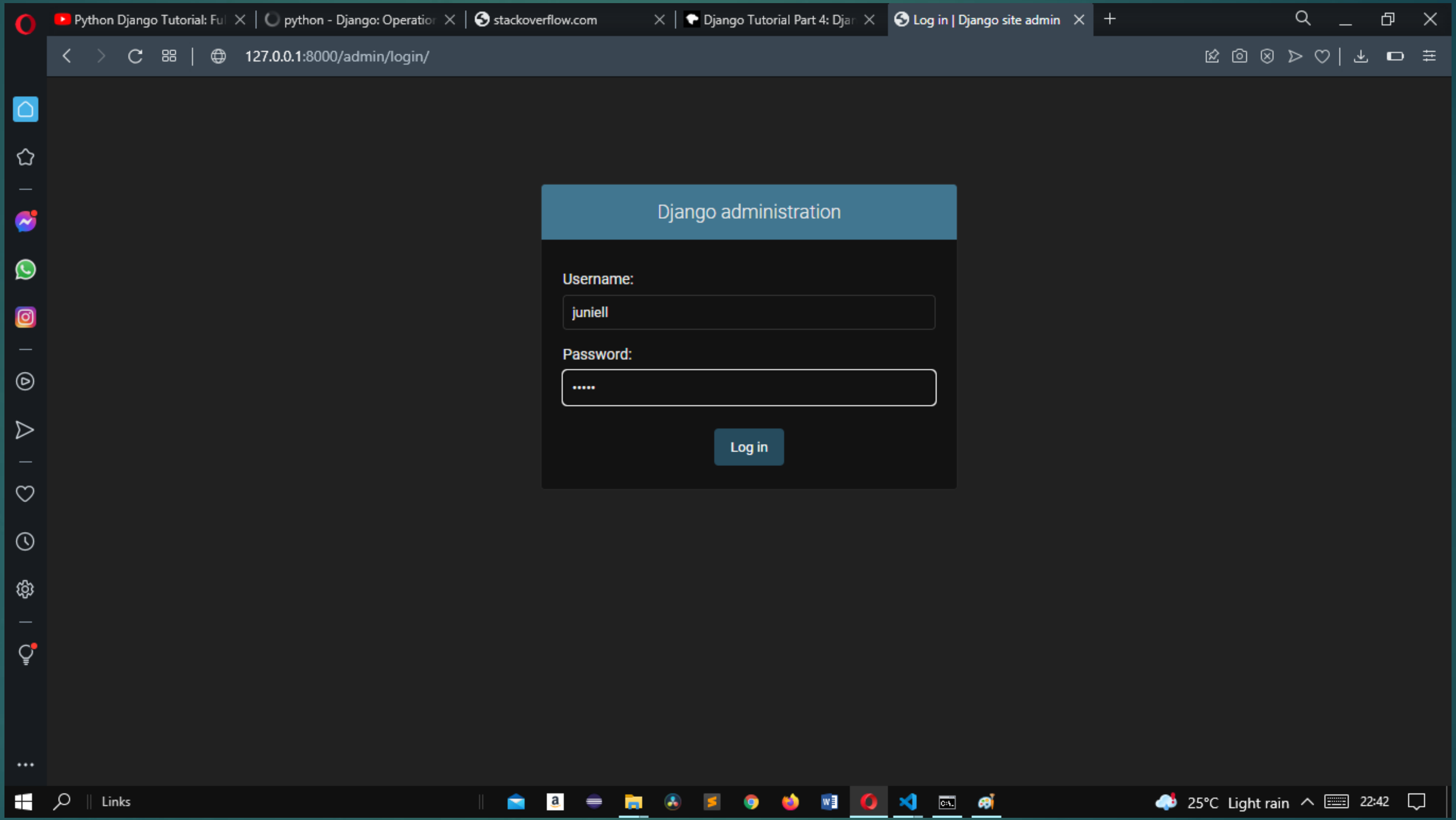
Django Documentation
Topics, references, & how-to's

Tutorial: A Polling App
Get started with Django

Django Community
Connect, get help, or contribute



```
C:\Users\ACER_LAPTOP\site\django_project>Python manage.py createsuperuser
System check identified some issues:

WARNINGS:
blog.Post: (models.W042) Auto-created primary key used when not defining a primary key type, by default 'django.db.model
s.AutoField'.
        HINT: Configure the DEFAULT_AUTO_FIELD setting or the BlogConfig.default_auto_field attribute to point to a subc
lass of AutoField, e.g. 'django.db.models.BigAutoField'.
users.Profile: (models.W042) Auto-created primary key used when not defining a primary key type, by default 'django.db.m
odels.AutoField'.
        HINT: Configure the DEFAULT_AUTO_FIELD setting or the UsersConfig.default_auto_field attribute to point to a sub
class of AutoField, e.g. 'django.db.models.BigAutoField'.
Username (leave blank to use 'untameconrad'):
Email address: jjemcoronado@gmail.com
Password:
Password (again):
Error: Your passwords didn't match.
Password:
Password (again):
Error: Your passwords didn't match.
Password:
Password (again):
This password is too common.
This password is entirely numeric.
Bypass password validation and create user anyway? [y/N]: y
Superuser created successfully.

C:\Users\ACER_LAPTOP\site\django_project>
```



Python Django Tutorial: Fu | python - Django: Operatio | stackoverflow.com | Django Tutorial Part 4: Dja | Log in | Django site admin

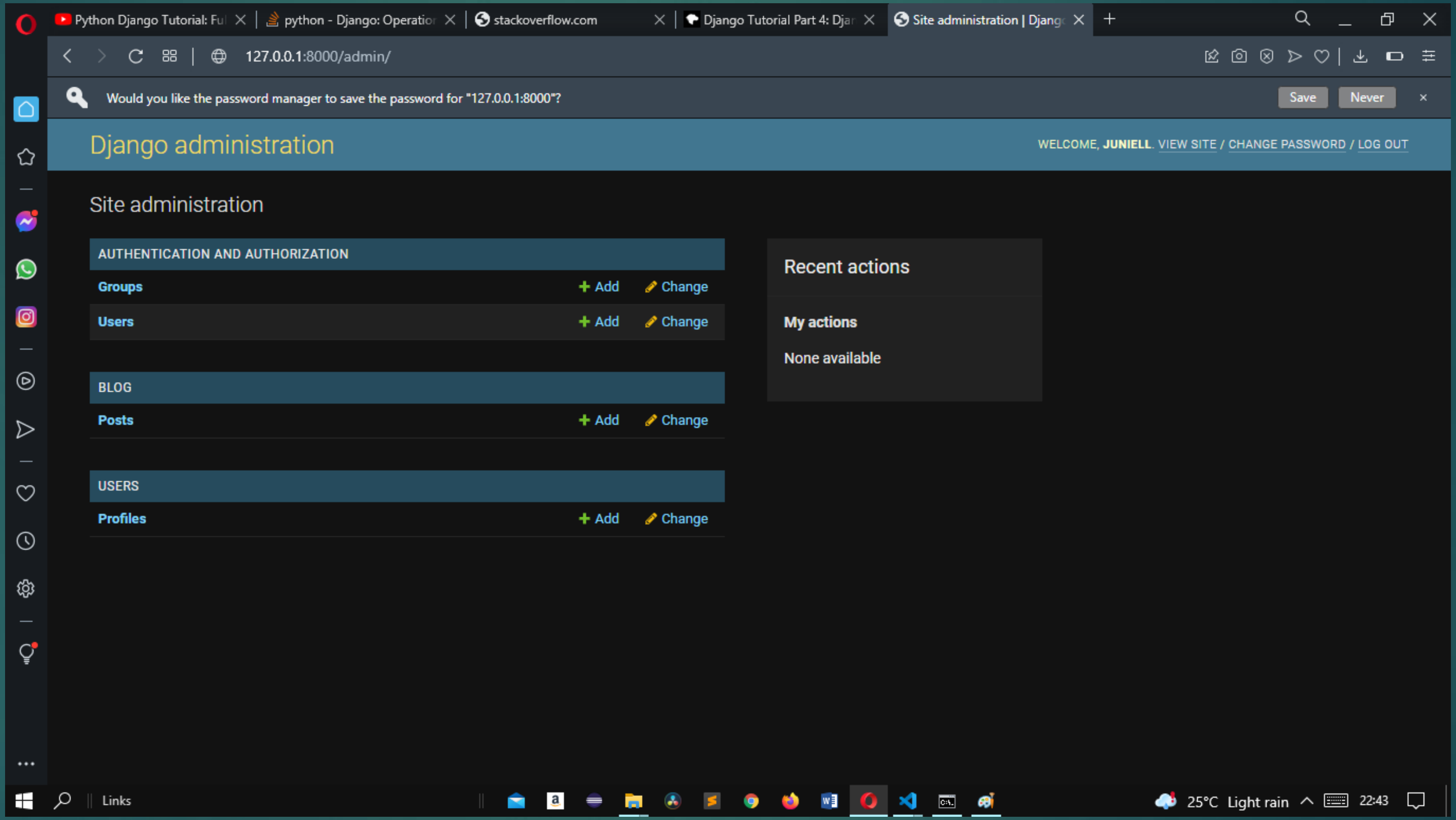
127.0.0.1:8000/admin/login/

Django administration

Username: juniell

Password: •••••

Log in



Python Django Tutorial: Fu
python - Django: Operatio
stackoverflow.com
Django Tutorial Part 4: Dja
Site administration | Djang
127.0.0.1:8000/admin/
Save
Never
Links
25°C Light rain
22:43

Would you like the password manager to save the password for "127.0.0.1:8000"?

Django administration

WELCOME, JUNIELL. VIEW SITE / CHANGE PASSWORD / LOG OUT

Site administration

AUTHENTICATION AND AUTHORIZATION

Groups	+ Add	Change
Users	+ Add	Change

BLOG

Posts	+ Add	Change

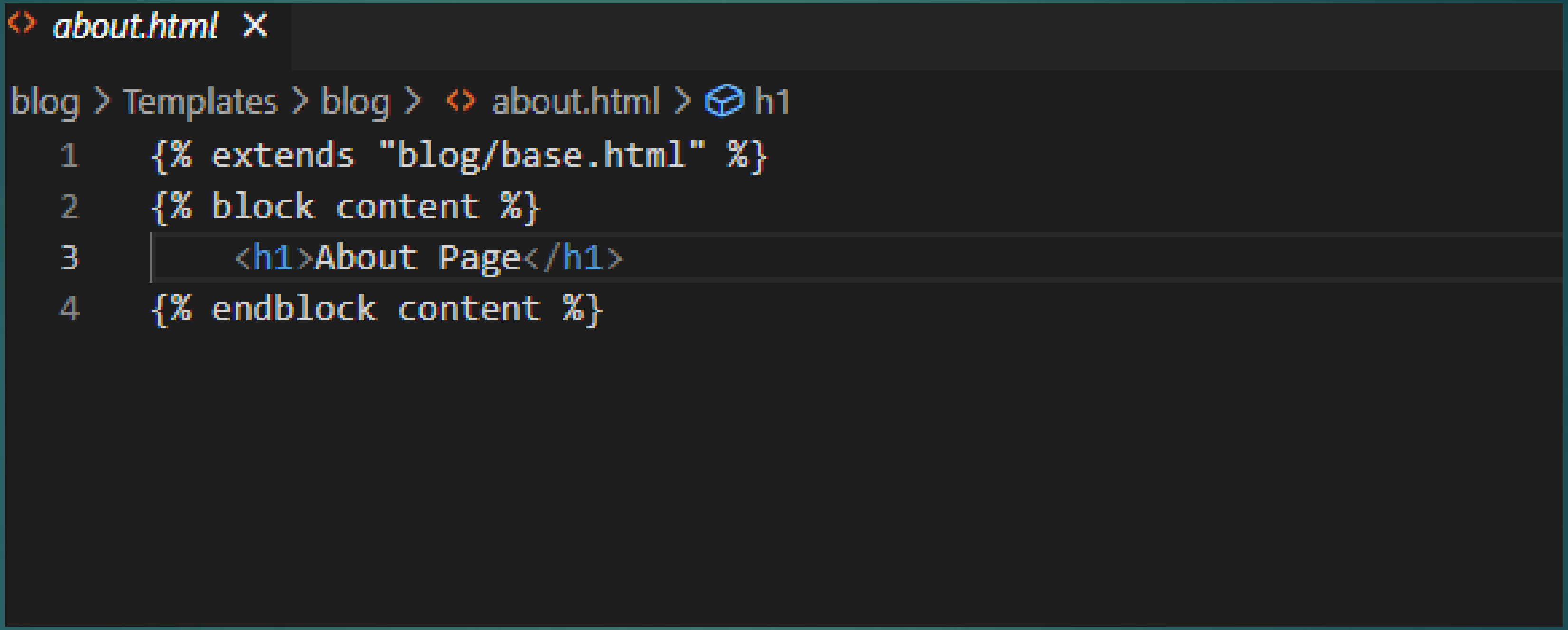
USERS

Profiles	+ Add	Change

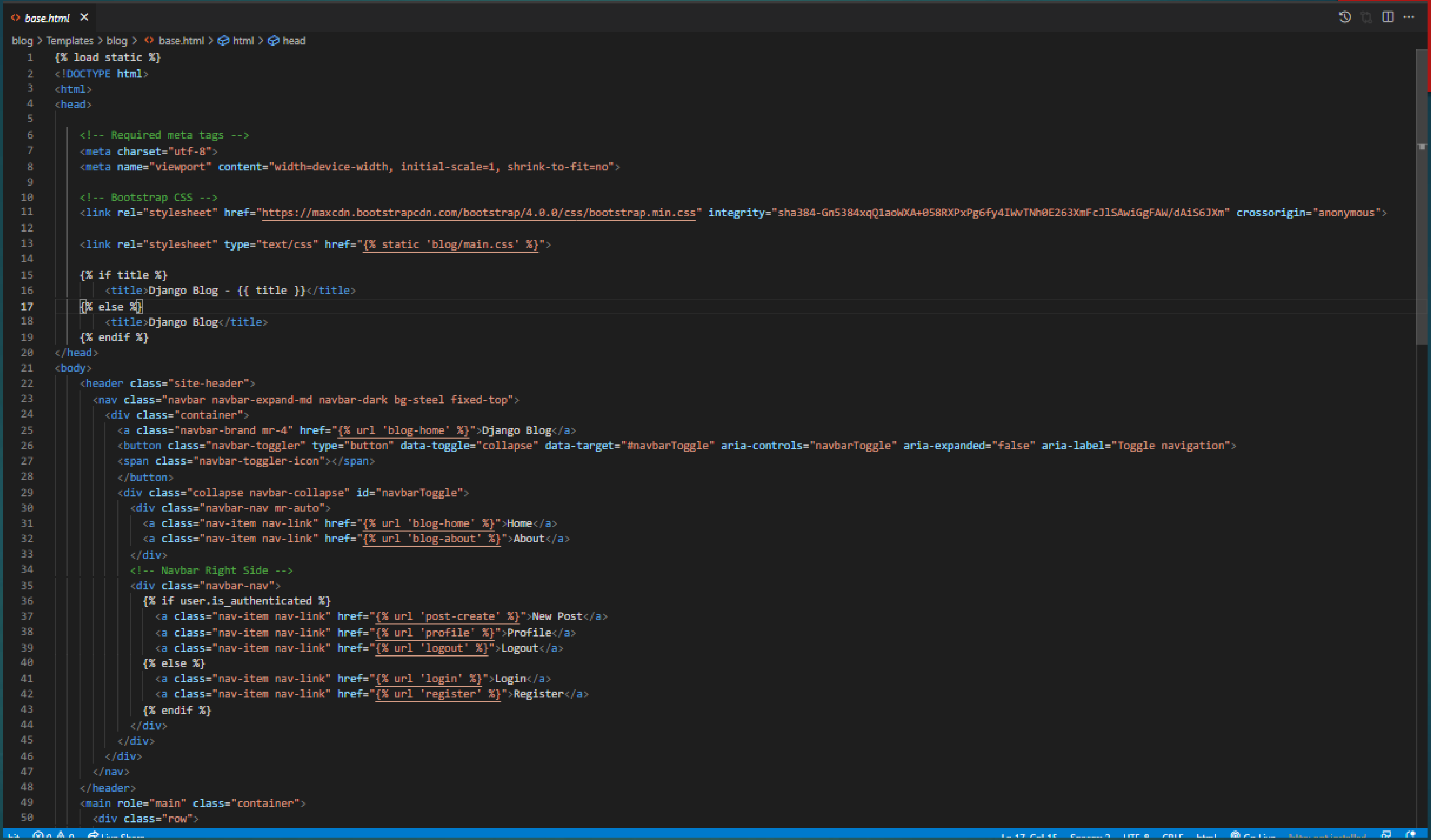
Recent actions

My actions

None available

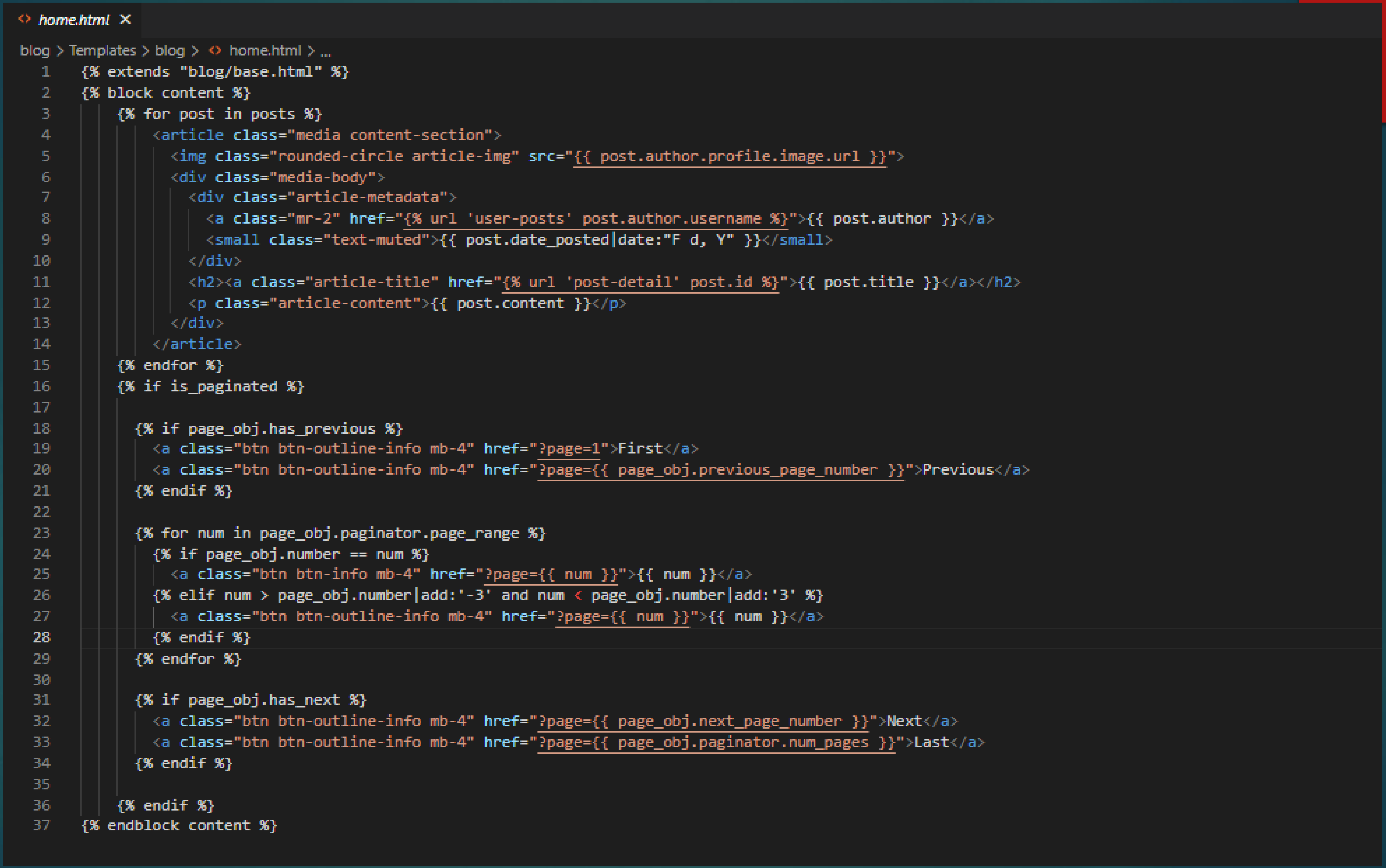


```
{% extends "blog/base.html" %}
{% block content %}
    <h1>About Page</h1>
{% endblock content %}
```

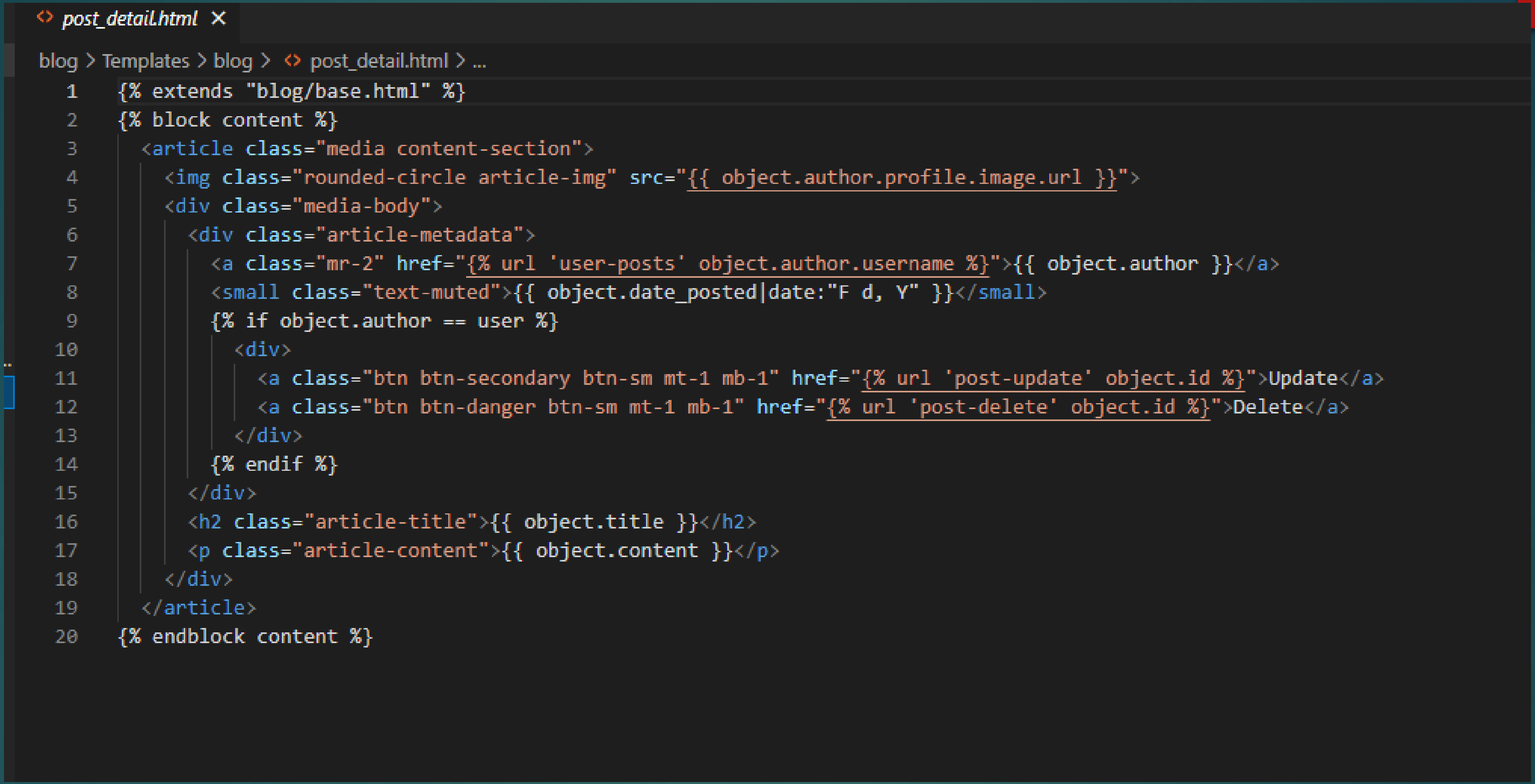

base.html

blog > Templates > blog > base.html > html > head

```html
{% load static %}
<!DOCTYPE html>
<html>
<head>

    <!-- Required meta tags -->
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">

    <!-- Bootstrap CSS -->
    <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/css/bootstrap.min.css" integrity="sha384-Gn5384xqQ1aoWXA+058RXPxPg6fy4IWvTNh0E263XmFcJlSAwiGgFAW/dAiS6JXm" crossorigin="anonymous">

    <link rel="stylesheet" type="text/css" href="{% static 'blog/main.css' %}">

    {% if title %}
        <title>Django Blog - {{ title }}</title>
    {% else %}
        <title>Django Blog</title>
    {% endif %}
</head>
<body>
    <header class="site-header">
      <nav class="navbar navbar-expand-md navbar-dark bg-steel fixed-top">
        <div class="container">
          <a class="navbar-brand mr-4" href="{% url 'blog-home' %}">Django Blog</a>
          <button class="navbar-toggler" type="button" data-toggle="collapse" data-target="#navbarToggle" aria-controls="navbarToggle" aria-expanded="false" aria-label="Toggle navigation">
          <span class="navbar-toggler-icon"></span>
          </button>
          <div class="collapse navbar-collapse" id="navbarToggle">
            <div class="navbar-nav mr-auto">
              <a class="nav-item nav-link" href="{% url 'blog-home' %}">Home</a>
              <a class="nav-item nav-link" href="{% url 'blog-about' %}">About</a>
            </div>
            <!-- Navbar Right Side -->
            <div class="navbar-nav">
              {% if user.is_authenticated %}
                <a class="nav-item nav-link" href="{% url 'post-create' %}">New Post</a>
                <a class="nav-item nav-link" href="{% url 'profile' %}">Profile</a>
                <a class="nav-item nav-link" href="{% url 'logout' %}">Logout</a>
              {% else %}
                <a class="nav-item nav-link" href="{% url 'login' %}">Login</a>
                <a class="nav-item nav-link" href="{% url 'register' %}">Register</a>
              {% endif %}
            </div>
          </div>
        </div>
      </nav>
    </header>
    <main role="main" class="container">
      <div class="row">
```

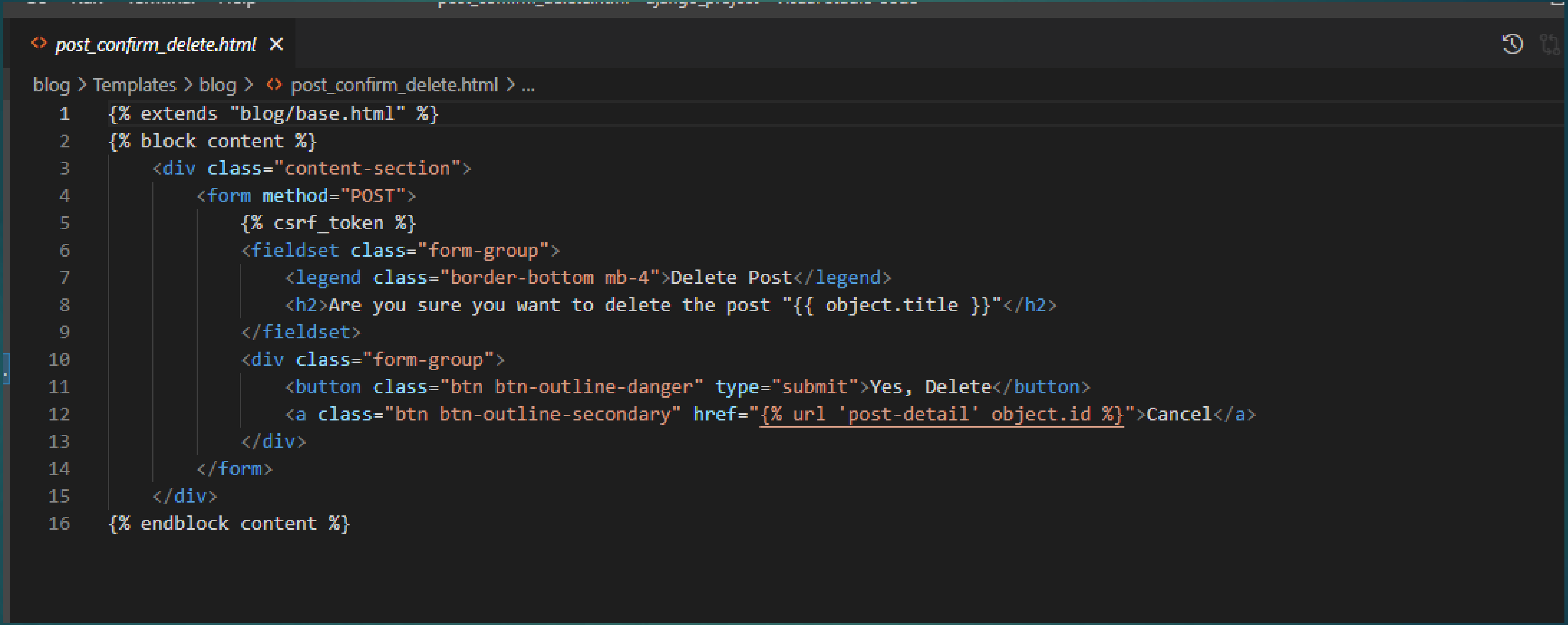


```
{% extends "blog/base.html" %}
{% block content %}
    {% for post in posts %}
        <article class="media content-section">
          <img class="rounded-circle article-img" src="{{ post.author.profile.image.url }}">
          <div class="media-body">
            <div class="article-metadata">
              <a class="mr-2" href="{% url 'user-posts' post.author.username %}">{{ post.author }}</a>
              <small class="text-muted">{{ post.date_posted|date:"F d, Y" }}</small>
            </div>
            <h2><a class="article-title" href="{% url 'post-detail' post.id %}">{{ post.title }}</a></h2>
            <p class="article-content">{{ post.content }}</p>
          </div>
        </article>
    {% endfor %}
    {% if is_paginated %}

      {% if page_obj.has_previous %}
        <a class="btn btn-outline-info mb-4" href="?page=1">First</a>
        <a class="btn btn-outline-info mb-4" href="?page={{ page_obj.previous_page_number }}">Previous</a>
      {% endif %}

      {% for num in page_obj.paginator.page_range %}
        {% if page_obj.number == num %}
          <a class="btn btn-info mb-4" href="?page={{ num }}">{{ num }}</a>
        {% elif num > page_obj.number|add:'-3' and num < page_obj.number|add:'3' %}
          <a class="btn btn-outline-info mb-4" href="?page={{ num }}">{{ num }}</a>
        {% endif %}
      {% endfor %}

      {% if page_obj.has_next %}
        <a class="btn btn-outline-info mb-4" href="?page={{ page_obj.next_page_number }}">Next</a>
        <a class="btn btn-outline-info mb-4" href="?page={{ page_obj.paginator.num_pages }}">Last</a>
      {% endif %}

    {% endif %}
{% endblock content %}
```

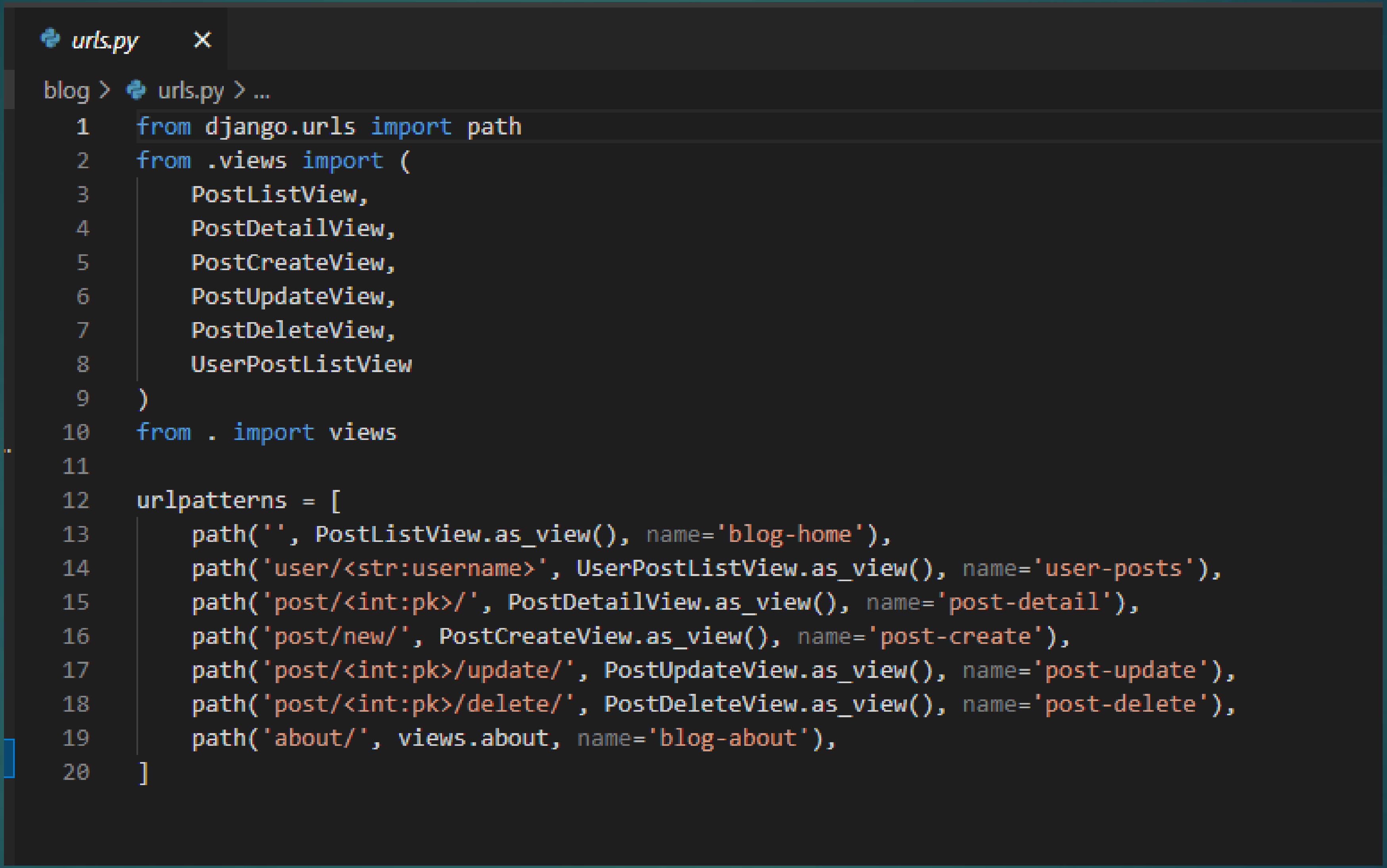

post_detail.html

blog > Templates > blog > post_detail.html > ...

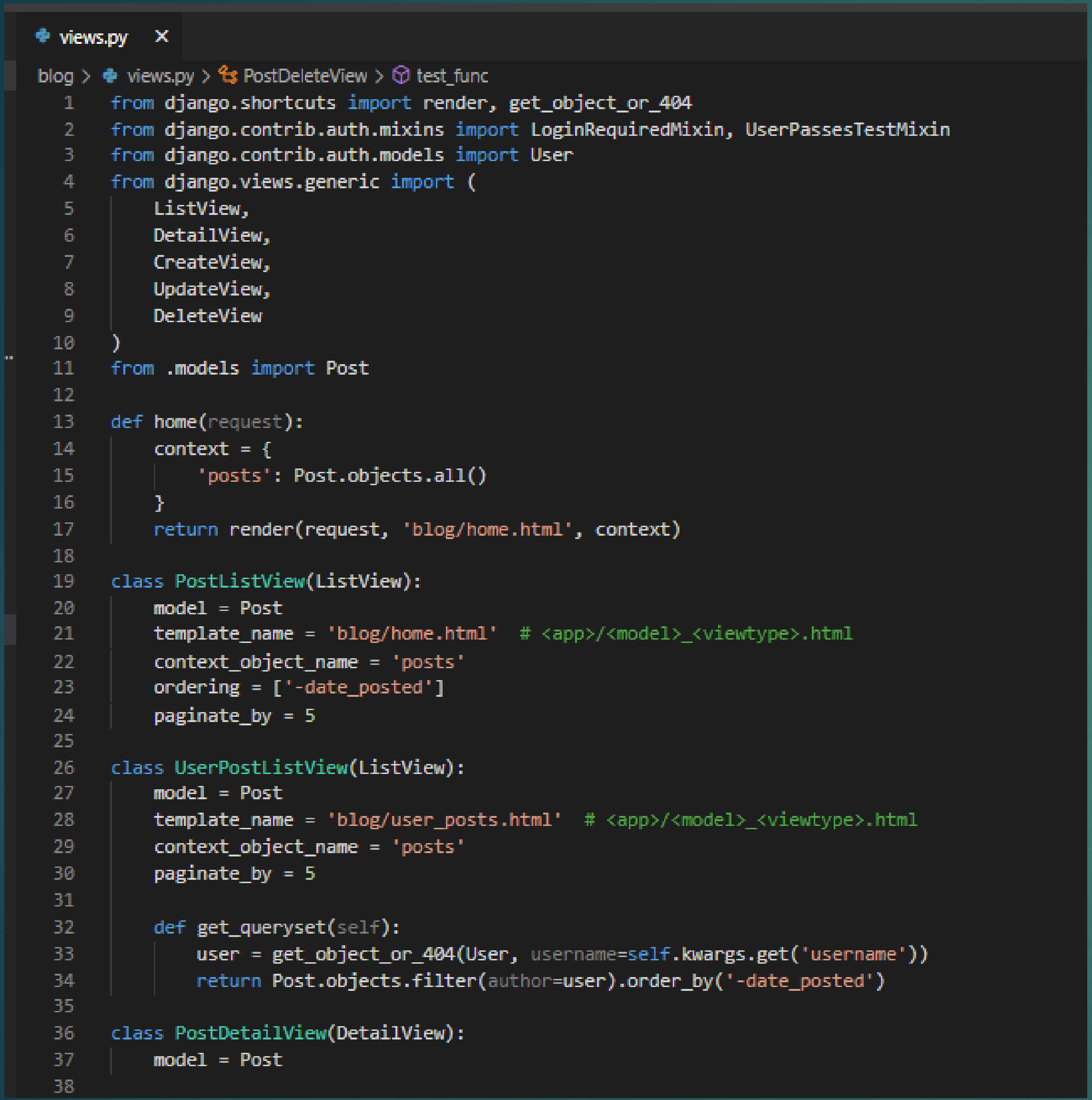
```
{% extends "blog/base.html" %}
{% block content %}
  <article class="media content-section">
    <img class="rounded-circle article-img" src="{{ object.author.profile.image.url }}">
    <div class="media-body">
      <div class="article-metadata">
        <a class="mr-2" href="{% url 'user-posts' object.author.username %}">{{ object.author }}</a>
        <small class="text-muted">{{ object.date_posted|date:"F d, Y" }}</small>
        {% if object.author == user %}
          <div>
            <a class="btn btn-secondary btn-sm mt-1 mb-1" href="{% url 'post-update' object.id %}">Update</a>
            <a class="btn btn-danger btn-sm mt-1 mb-1" href="{% url 'post-delete' object.id %}">Delete</a>
          </div>
        {% endif %}
      </div>
      <h2 class="article-title">{{ object.title }}</h2>
      <p class="article-content">{{ object.content }}</p>
    </div>
  </article>
{% endblock content %}
```



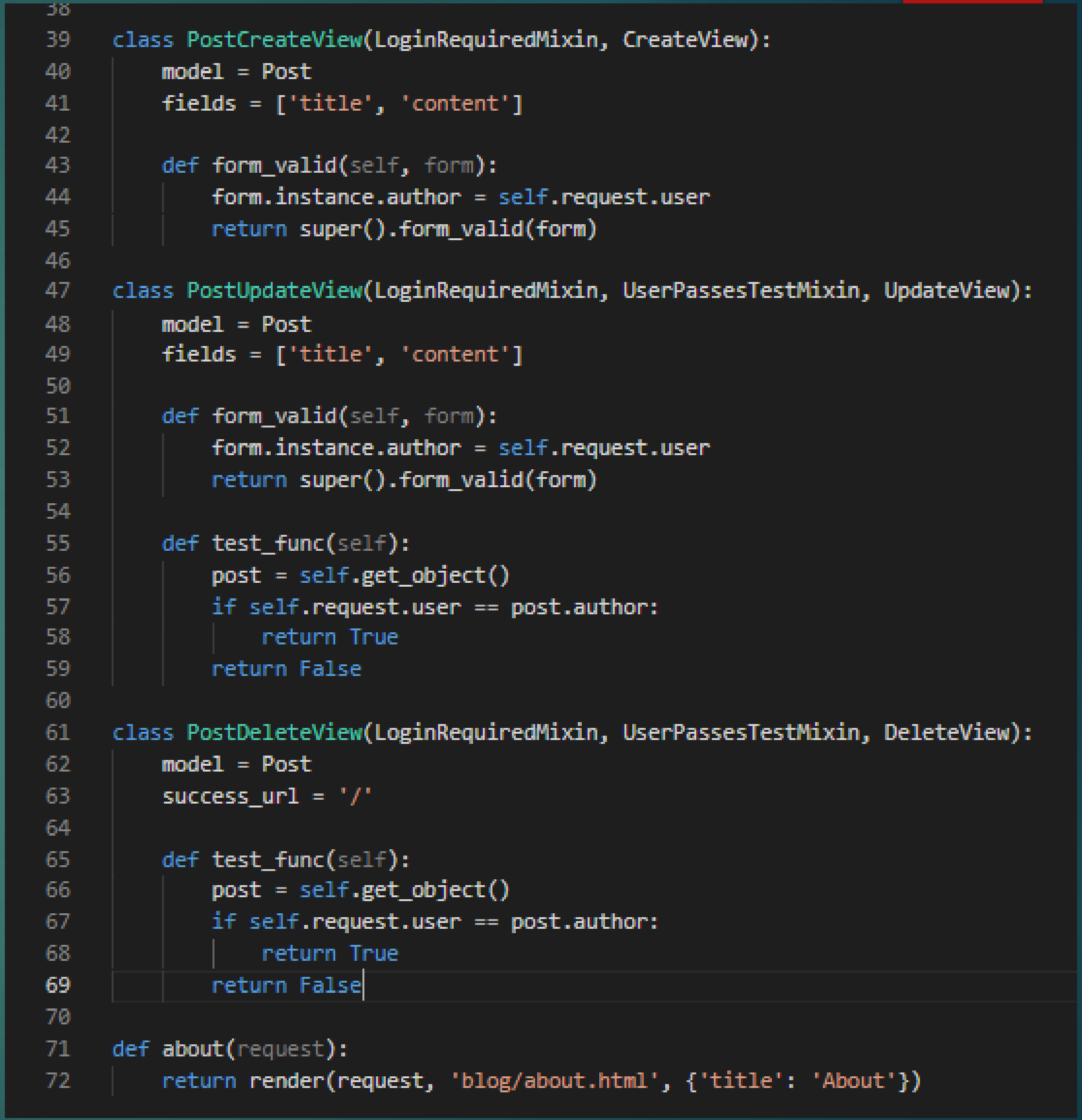
```
{% extends "blog/base.html" %}
{% block content %}
    <div class="content-section">
        <form method="POST">
            {% csrf_token %}
            <fieldset class="form-group">
                <legend class="border-bottom mb-4">Delete Post</legend>
                <h2>Are you sure you want to delete the post "{{ object.title }}"</h2>
            </fieldset>
            <div class="form-group">
                <button class="btn btn-outline-danger" type="submit">Yes, Delete</button>
                <a class="btn btn-outline-secondary" href="{% url 'post-detail' object.id %}">Cancel</a>
            </div>
        </form>
    </div>
{% endblock content %}
```



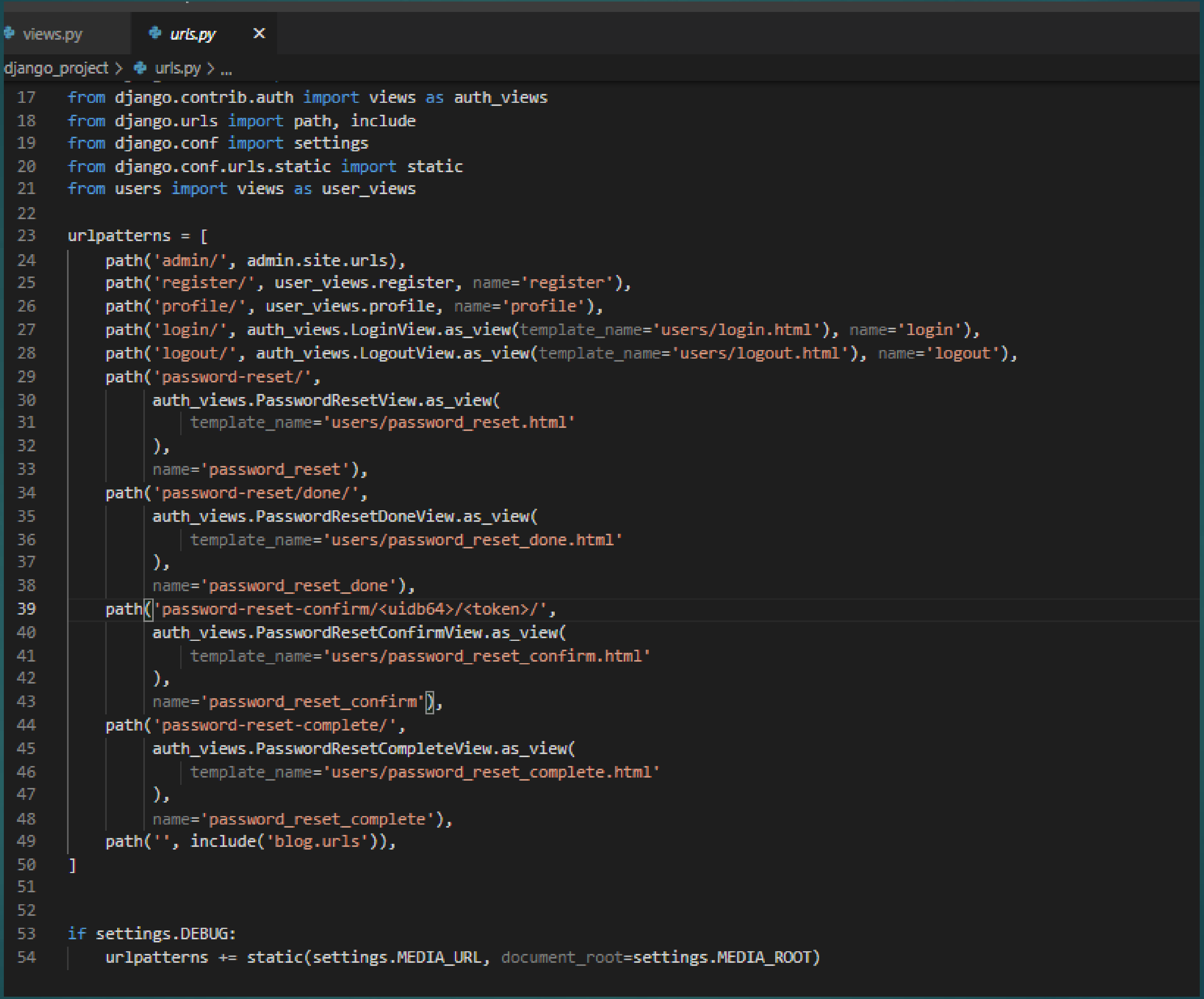
```python
from django.urls import path
from .views import (
    PostListView,
    PostDetailView,
    PostCreateView,
    PostUpdateView,
    PostDeleteView,
    UserPostListView
)
from . import views

urlpatterns = [
    path('', PostListView.as_view(), name='blog-home'),
    path('user/<str:username>', UserPostListView.as_view(), name='user-posts'),
    path('post/<int:pk>/', PostDetailView.as_view(), name='post-detail'),
    path('post/new/', PostCreateView.as_view(), name='post-create'),
    path('post/<int:pk>/update/', PostUpdateView.as_view(), name='post-update'),
    path('post/<int:pk>/delete/', PostDeleteView.as_view(), name='post-delete'),
    path('about/', views.about, name='blog-about'),
]
```



```python
from django.shortcuts import render, get_object_or_404
from django.contrib.auth.mixins import LoginRequiredMixin, UserPassesTestMixin
from django.contrib.auth.models import User
from django.views.generic import (
    ListView,
    DetailView,
    CreateView,
    UpdateView,
    DeleteView
)
from .models import Post

def home(request):
    context = {
        'posts': Post.objects.all()
    }
    return render(request, 'blog/home.html', context)

class PostListView(ListView):
    model = Post
    template_name = 'blog/home.html'  # <app>/<model>_<viewtype>.html
    context_object_name = 'posts'
    ordering = ['-date_posted']
    paginate_by = 5

class UserPostListView(ListView):
    model = Post
    template_name = 'blog/user_posts.html'  # <app>/<model>_<viewtype>.html
    context_object_name = 'posts'
    paginate_by = 5

    def get_queryset(self):
        user = get_object_or_404(User, username=self.kwargs.get('username'))
        return Post.objects.filter(author=user).order_by('-date_posted')

class PostDetailView(DetailView):
    model = Post

```

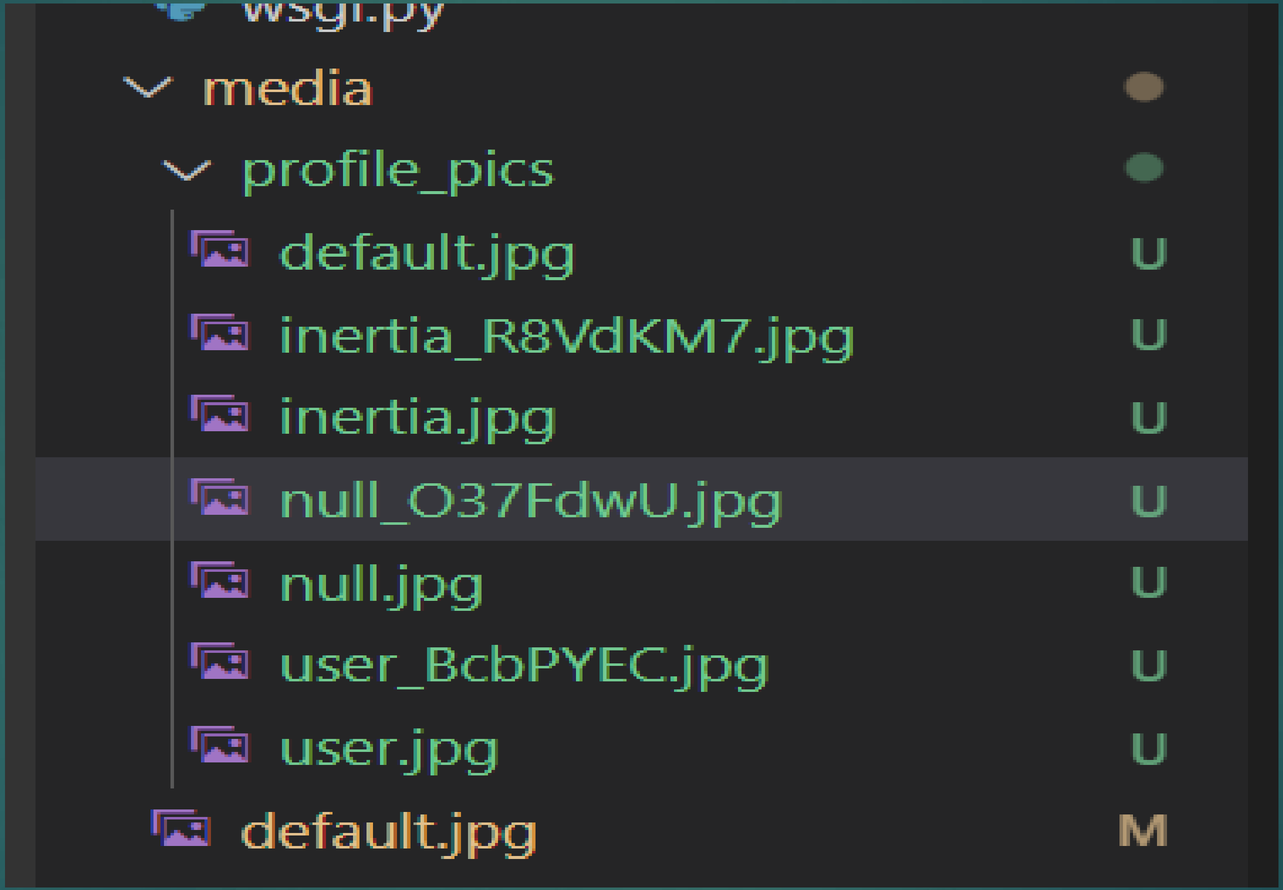


```python

class PostCreateView(LoginRequiredMixin, CreateView):
    model = Post
    fields = ['title', 'content']

    def form_valid(self, form):
        form.instance.author = self.request.user
        return super().form_valid(form)

class PostUpdateView(LoginRequiredMixin, UserPassesTestMixin, UpdateView):
    model = Post
    fields = ['title', 'content']

    def form_valid(self, form):
        form.instance.author = self.request.user
        return super().form_valid(form)

    def test_func(self):
        post = self.get_object()
        if self.request.user == post.author:
            return True
        return False

class PostDeleteView(LoginRequiredMixin, UserPassesTestMixin, DeleteView):
    model = Post
    success_url = '/'

    def test_func(self):
        post = self.get_object()
        if self.request.user == post.author:
            return True
        return False

def about(request):
    return render(request, 'blog/about.html', {'title': 'About'})
```



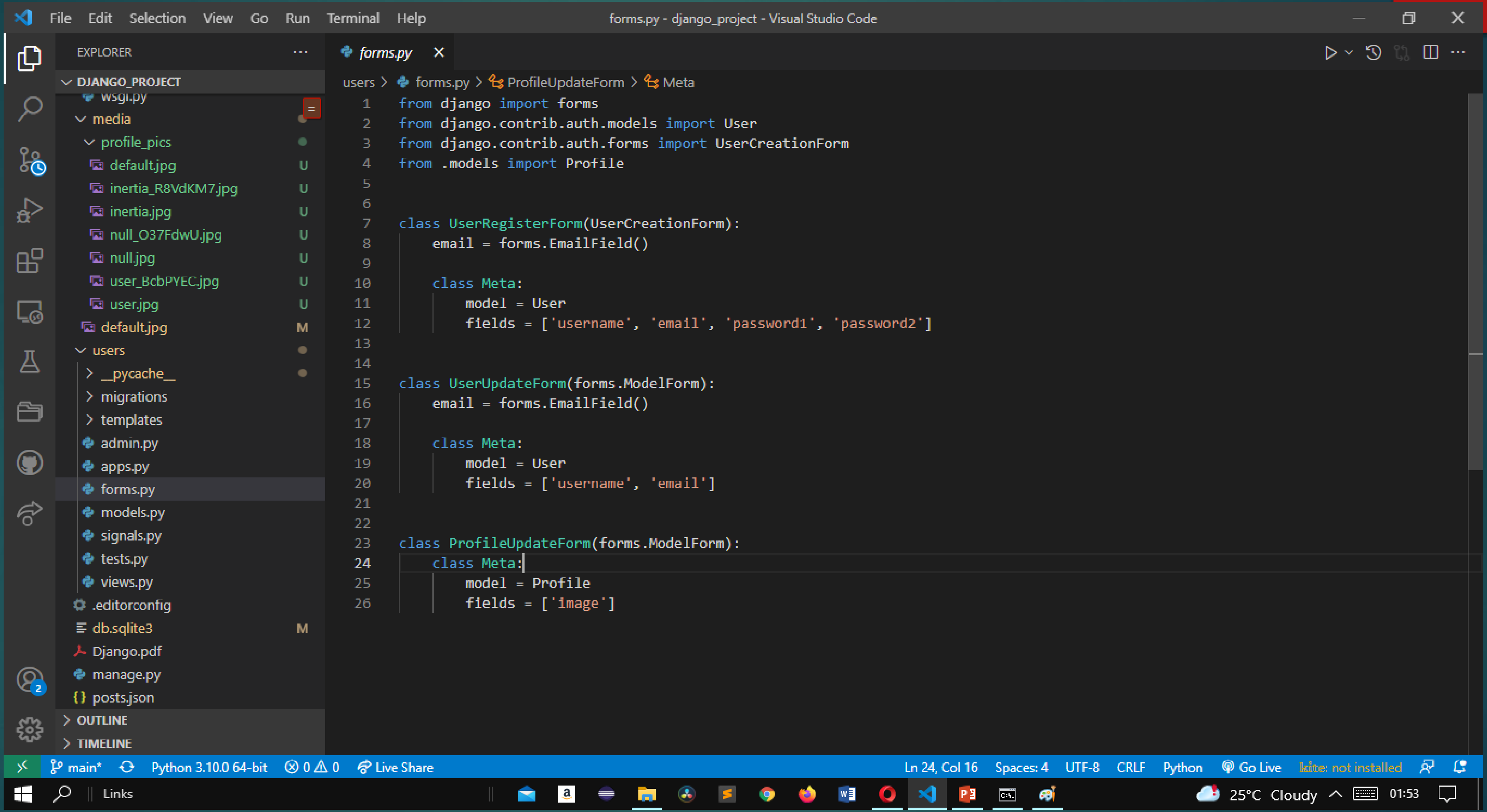
```python
from django.contrib.auth import views as auth_views
from django.urls import path, include
from django.conf import settings
from django.conf.urls.static import static
from users import views as user_views

urlpatterns = [
    path('admin/', admin.site.urls),
    path('register/', user_views.register, name='register'),
    path('profile/', user_views.profile, name='profile'),
    path('login/', auth_views.LoginView.as_view(template_name='users/login.html'), name='login'),
    path('logout/', auth_views.LogoutView.as_view(template_name='users/logout.html'), name='logout'),
    path('password-reset/',
         auth_views.PasswordResetView.as_view(
             template_name='users/password_reset.html'
         ),
         name='password_reset'),
    path('password-reset/done/',
         auth_views.PasswordResetDoneView.as_view(
             template_name='users/password_reset_done.html'
         ),
         name='password_reset_done'),
    path('password-reset-confirm/<uidb64>/<token>/',
         auth_views.PasswordResetConfirmView.as_view(
             template_name='users/password_reset_confirm.html'
         ),
         name='password_reset_confirm'),
    path('password-reset-complete/',
         auth_views.PasswordResetCompleteView.as_view(
             template_name='users/password_reset_complete.html'
         ),
         name='password_reset_complete'),
    path('', include('blog.urls')),
]


if settings.DEBUG:
    urlpatterns += static(settings.MEDIA_URL, document_root=settings.MEDIA_ROOT)
```

Create a folder named media to store some photos. This serves as profile to the users.

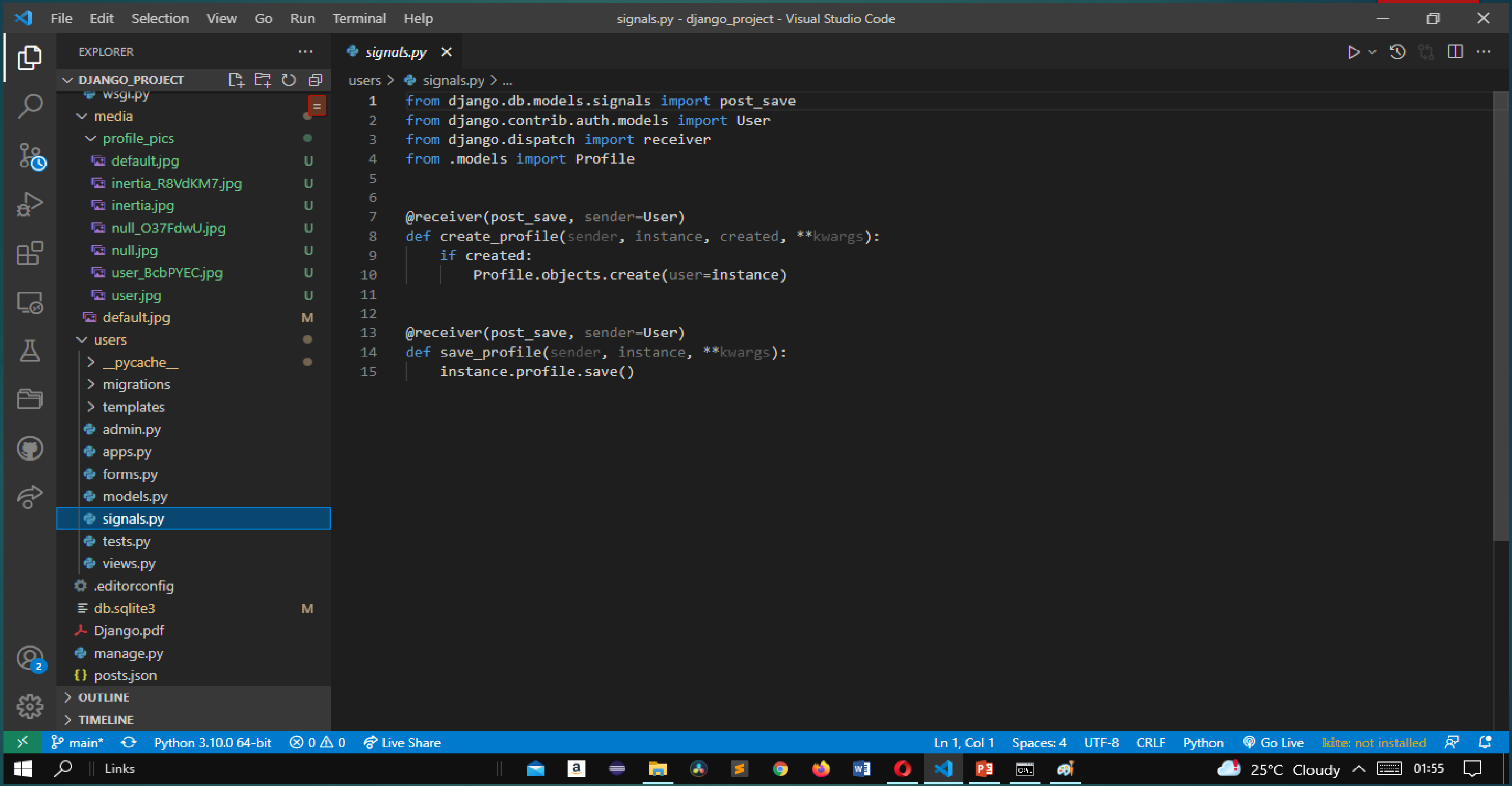


```
wsgi.py
media
  profile_pics
    default.jpg            U
    inertia_R8VdKM7.jpg    U
    inertia.jpg            U
    null_O37FdwU.jpg       U
    null.jpg               U
    user_BcbPYEC.jpg       U
    user.jpg               U
  default.jpg              M
```

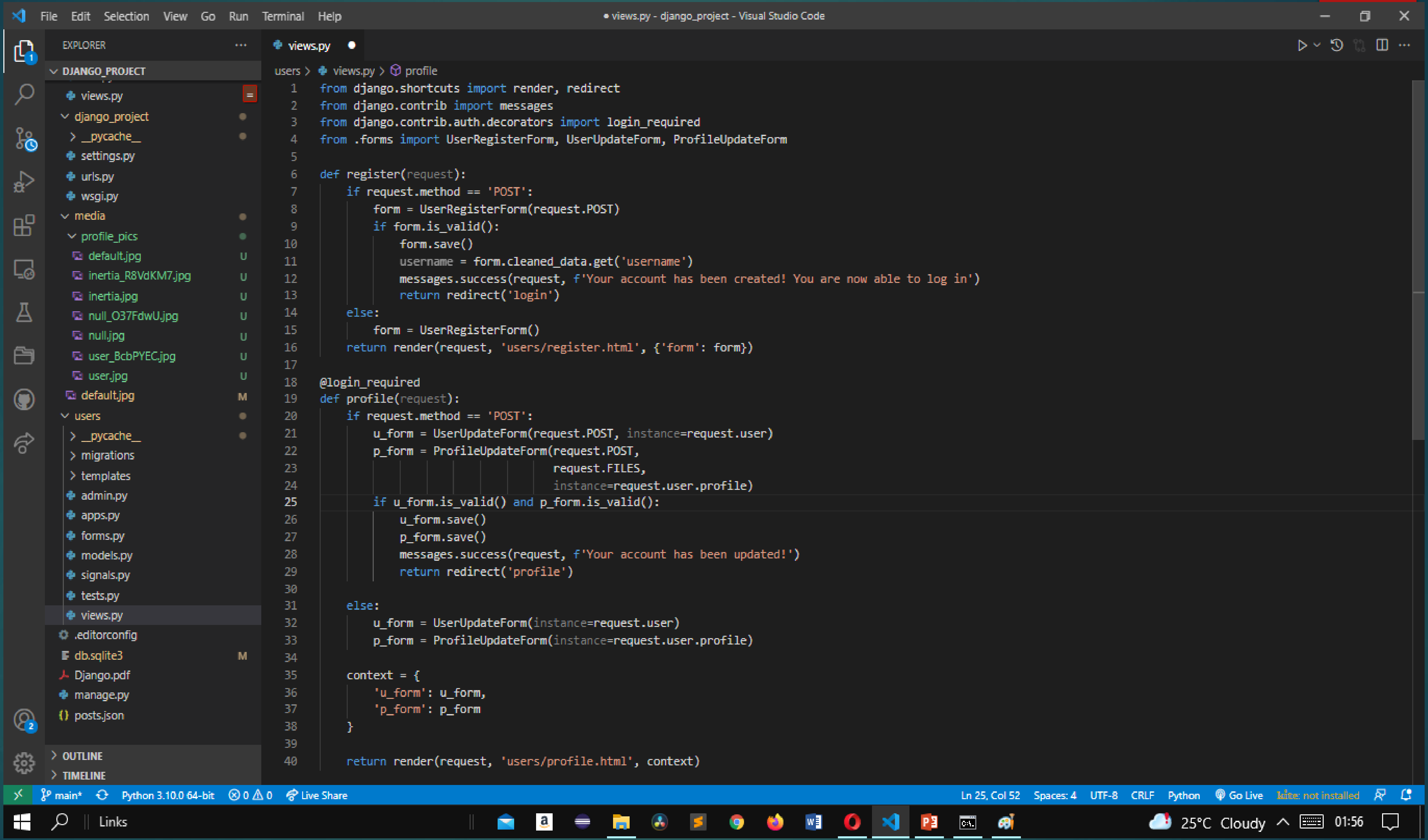
- Create another app folder and named it users.



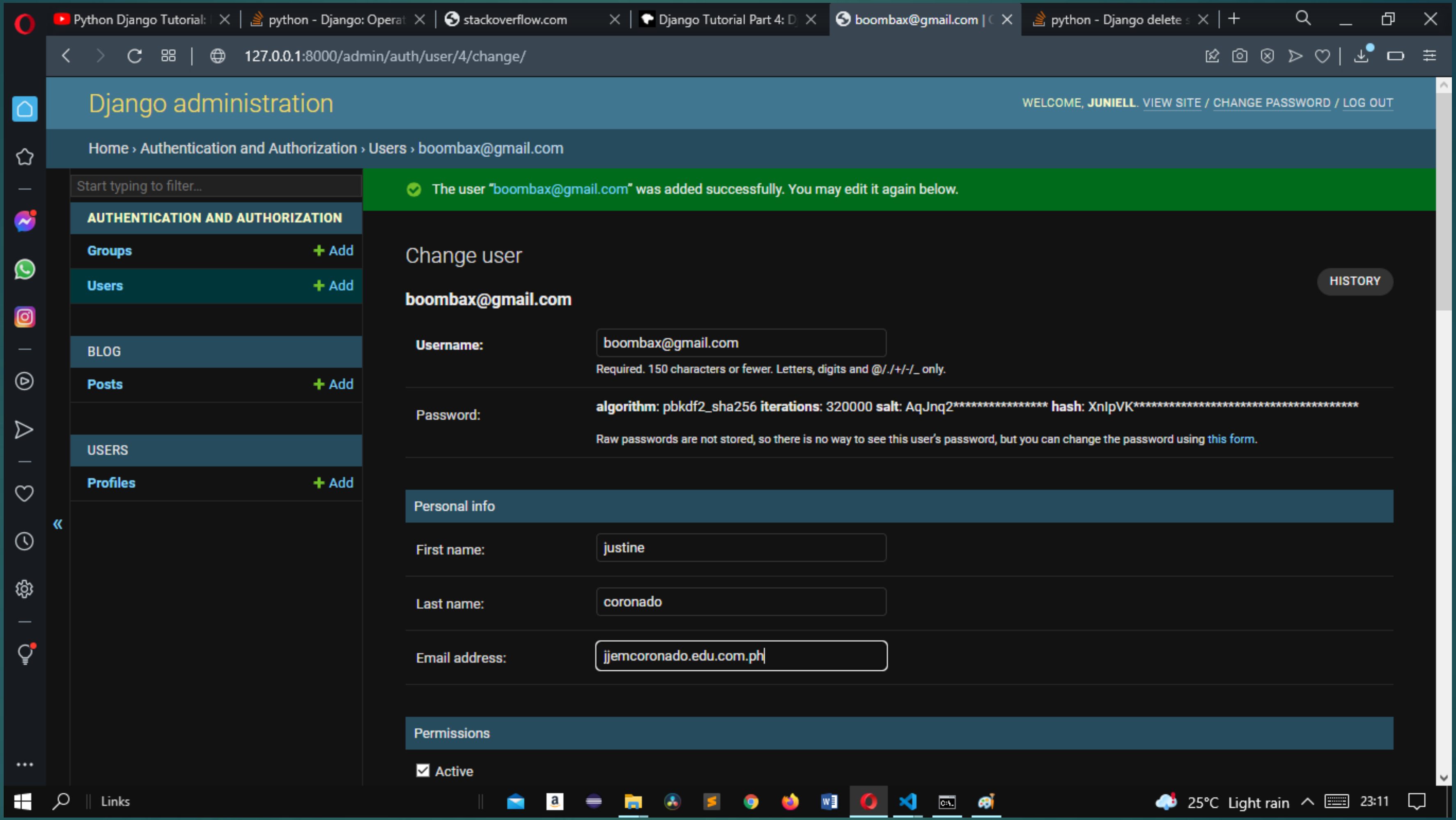
```python
from django import forms
from django.contrib.auth.models import User
from django.contrib.auth.forms import UserCreationForm
from .models import Profile


class UserRegisterForm(UserCreationForm):
    email = forms.EmailField()

    class Meta:
        model = User
        fields = ['username', 'email', 'password1', 'password2']


class UserUpdateForm(forms.ModelForm):
    email = forms.EmailField()

    class Meta:
        model = User
        fields = ['username', 'email']


class ProfileUpdateForm(forms.ModelForm):
    class Meta:
        model = Profile
        fields = ['image']
```



```python
from django.db.models.signals import post_save
from django.contrib.auth.models import User
from django.dispatch import receiver
from .models import Profile


@receiver(post_save, sender=User)
def create_profile(sender, instance, created, **kwargs):
    if created:
        Profile.objects.create(user=instance)


@receiver(post_save, sender=User)
def save_profile(sender, instance, **kwargs):
    instance.profile.save()
```



```python
from django.shortcuts import render, redirect
from django.contrib import messages
from django.contrib.auth.decorators import login_required
from .forms import UserRegisterForm, UserUpdateForm, ProfileUpdateForm

def register(request):
    if request.method == 'POST':
        form = UserRegisterForm(request.POST)
        if form.is_valid():
            form.save()
            username = form.cleaned_data.get('username')
            messages.success(request, f'Your account has been created! You are now able to log in')
            return redirect('login')
    else:
        form = UserRegisterForm()
    return render(request, 'users/register.html', {'form': form})

@login_required
def profile(request):
    if request.method == 'POST':
        u_form = UserUpdateForm(request.POST, instance=request.user)
        p_form = ProfileUpdateForm(request.POST,
                                   request.FILES,
                                   instance=request.user.profile)
        if u_form.is_valid() and p_form.is_valid():
            u_form.save()
            p_form.save()
            messages.success(request, f'Your account has been updated!')
            return redirect('profile')

    else:
        u_form = UserUpdateForm(instance=request.user)
        p_form = ProfileUpdateForm(instance=request.user.profile)

    context = {
        'u_form': u_form,
        'p_form': p_form
    }

    return render(request, 'users/profile.html', context)
```



Django administration

Home › Authentication and Authorization › Users › boombax@gmail.com

The user “boombax@gmail.com” was added successfully. You may edit it again below.

Change user

boombax@gmail.com

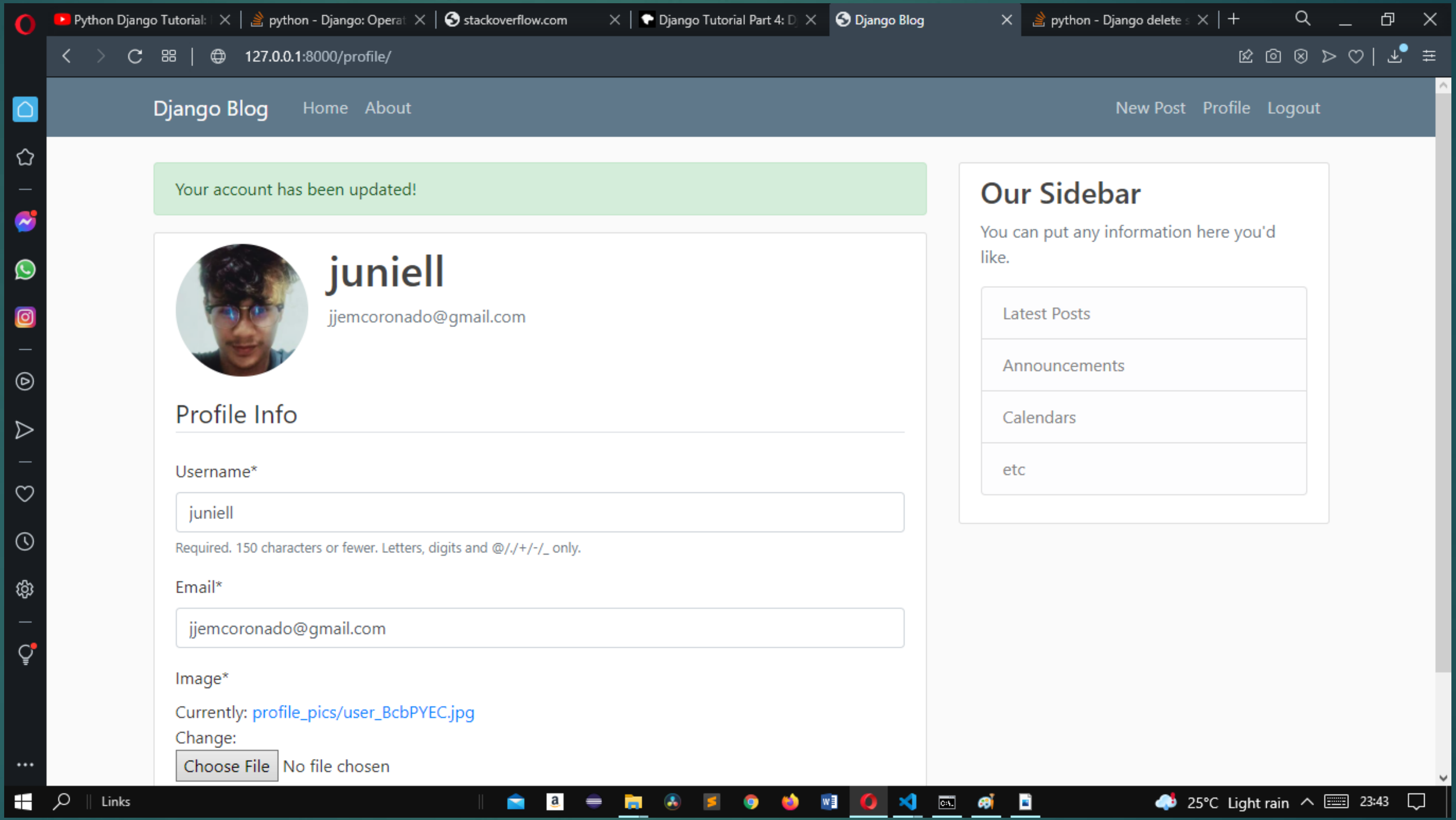
Username:	boombax@gmail.com Required. 150 characters or fewer. Letters, digits and @/./+/-/_ only.
Password:	**algorithm**: pbkdf2_sha256 **iterations**: 320000 **salt**: AqJnq2************** **hash**: XnIpVK************************************** Raw passwords are not stored, so there is no way to see this user’s password, but you can change the password using this form.

Personal info

First name:	justine
Last name:	coronado
Email address:	jjemcoronado.edu.com.ph

Permissions

☑ Active


Django Blog Home About New Post Profile Logout

Your account has been updated!

juniell
jjemcoronado@gmail.com

Profile Info

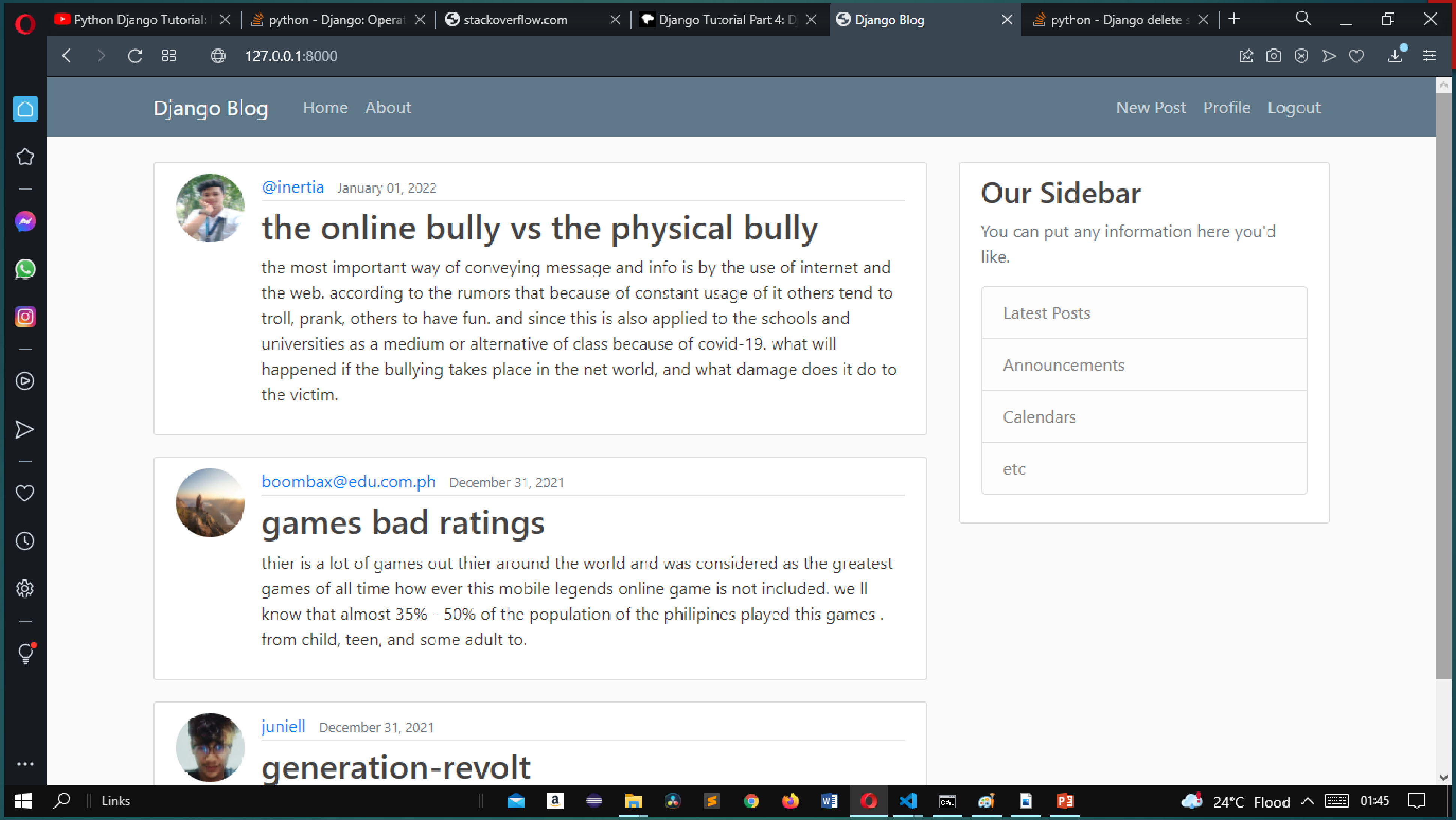
Username*
juniell
Required. 150 characters or fewer. Letters, digits and @/./+/-/_ only.

Email*
jjemcoronado@gmail.com

Image*
Currently: profile_pics/user_BcbPYEC.jpg
Change:
Choose File No file chosen

Our Sidebar
You can put any information here you'd like.

- Latest Posts
- Announcements
- Calendars
- etc



Django Blog Home About New Post Profile Logout

@inertia January 01, 2022

the online bully vs the physical bully

the most important way of conveying message and info is by the use of internet and the web. according to the rumors that because of constant usage of it others tend to troll, prank, others to have fun. and since this is also applied to the schools and universities as a medium or alternative of class because of covid-19. what will happened if the bullying takes place in the net world, and what damage does it do to the victim.

boombax@edu.com.ph December 31, 2021

games bad ratings

thier is a lot of games out thier around the world and was considered as the greatest games of all time how ever this mobile legends online game is not included. we ll know that almost 35% - 50% of the population of the philipines played this games . from child, teen, and some adult to.

juniell December 31, 2021

generation-revolt

Our Sidebar

You can put any information here you'd like.

- Latest Posts
- Announcements
- Calendars
- etc