

Proyecto Final: Entrega módulo 6 Django

“Desarrollo completo de un blog con Django”

Nicolás Jofré Andrade - Desarrollo Full-stack Python

04-2024

Objetivo: Crear un blog utilizando Django que incluya desde la creación del proyecto hasta la administración de usuarios y grupos en el sitio administrativo.

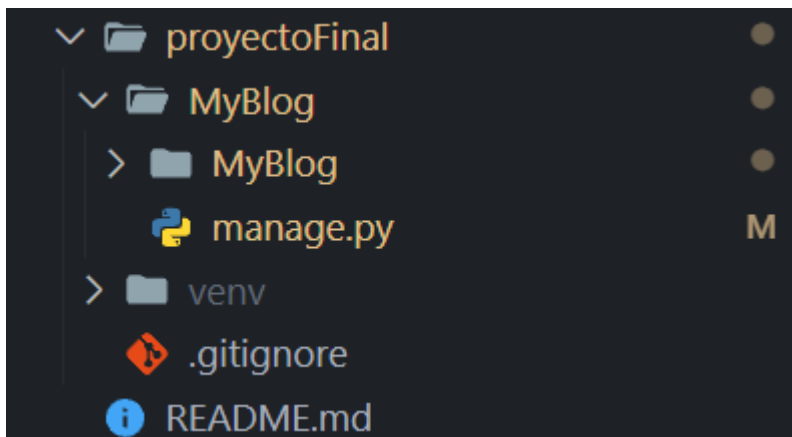
1. Introducción a Django:
 - a. Instalar Django en un entorno virtual

```
TERMINAL

> pip install django
• Collecting django
  Obtaining dependency information for django from https://files.pythonhost
y3-none-any.whl.metadata
  Using cached Django-5.0.4-py3-none-any.whl.metadata (4.1 kB)
```

2. Creación del proyecto django
 - a. Crea un nuevo proyecto llamado ‘MyBlog’

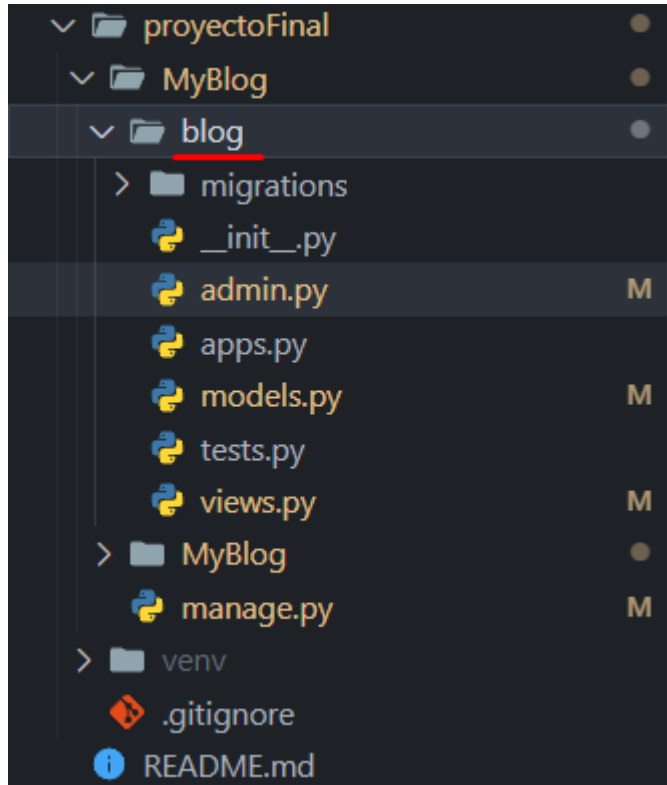
```
> django-admin startproject MyBlog
• (venv) njofre ~\Desktop\dev\portafolio-py-mod6\proyectoFinal main ≡*~5 -14 591ms
```



3. Despliegue de páginas con contenido dinámico:

a. Crea una aplicación llamada 'Blog'

```
> python manage.py startapp blog
• (venv) njofre ~\Desktop\dev\portafolio-py-mod6\proyectoFinal\MyBlog main ≡*~8 -7 621ms
```



4. Forms en Django:

a. Crea un formulario para agregar comentarios en las publicaciones.

```
forms.py ×
1  from django import forms
2  from .models import Comment
3
4  class CommentForm(forms.ModelForm):
5      class Meta:
6          model = Comment
7          fields = ['name', 'email', 'body']
```

- b. Vincula el formulario con una vista y muestra los comentarios en las publicaciones.

```
11 @login_required
12 def article_detail(request, article_id):
13     article = get_object_or_404(Article, id=article_id)
14     comments = article.comments.all() # Obtiene todos los comentarios asociados a este artículo
15     if request.method == 'POST':
16         form = CommentForm(request.POST)
17         if form.is_valid():
18             comment = form.save(commit=False)
19             comment.article = article
20             comment.save()
21             return redirect('article_detail', article_id=article_id)
22     else:
23         form = CommentForm()
24     return render(request, 'article_detail.html', {'article': article, 'comments': comments, 'form': form})
25
26 class CustomLoginView(LoginView):
27     template_name = 'login.html'
```

5. Autenticación y autorización:

- a. Implementa la autenticación para permitir a los usuarios comentar en el blog.

```
103 LOGIN_REDIRECT_URL = 'home'
104 LOGOUT_REDIRECT_URL = 'home'
```

- b. Limita el acceso a ciertas vistas solo a usuarios autenticados utilizando mixins.

```
11 @login_required
12 def article_detail(request, article_id):
13     article = get_object_or_404(Article, id=article_id)
14     comments = article.comments.all() # Obtiene todos los comentarios asociados a este artículo
15     if request.method == 'POST':
16         form = CommentForm(request.POST)
17         if form.is_valid():
18             comment = form.save(commit=False)
19             comment.article = article
20             comment.save()
21             return redirect('article_detail', article_id=article_id)
22     else:
23         form = CommentForm()
24     return render(request, 'article_detail.html', {'article': article, 'comments': comments, 'form': form})
25
```

6. El sitio administrativo de Django

- a. Crea un superusuario para gestionar el blog desde el sitio administrativo

```
> python manage.py createsuperuser
● Username: njofre
Error: That username is already taken.
Username: admin
Email address: a@a.cl
Password:
Password (again):
The password is too similar to the username.
This password is too common.
Bypass password validation and create user anyway? [y/N]: y
Superuser created successfully.
```