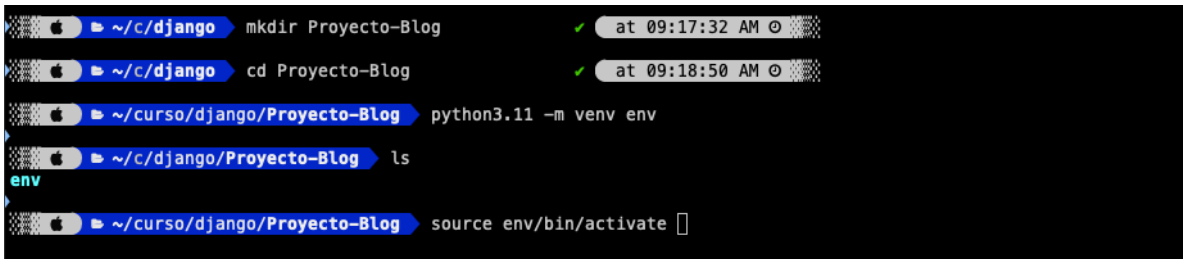
PROYECTO BLOG

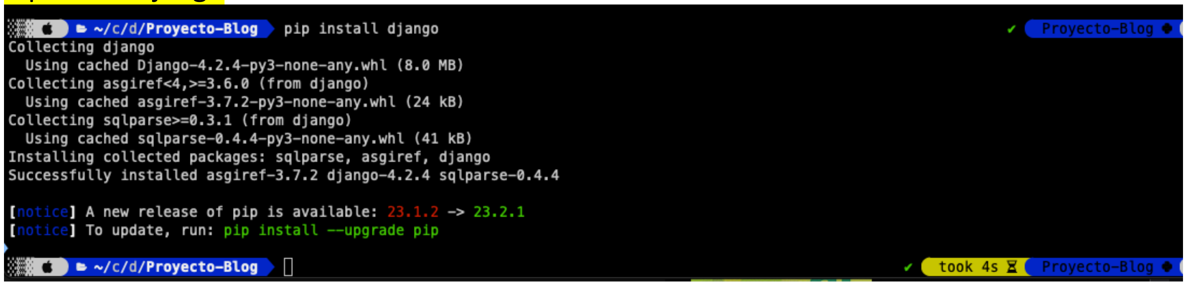
I. CREAR CARPETA Proyecto-block y entorno virtual y activarlo.



```
~/c/django  mkdir Proyecto-Blog                      ✔  at 09:17:32 AM
~/c/django  cd Proyecto-Blog                         ✔  at 09:18:50 AM
~/curso/django/Proyecto-Blog  python3.11 -m venv env
~/c/django/Proyecto-Blog  ls
env
~/curso/django/Proyecto-Blog  source env/bin/activate
```

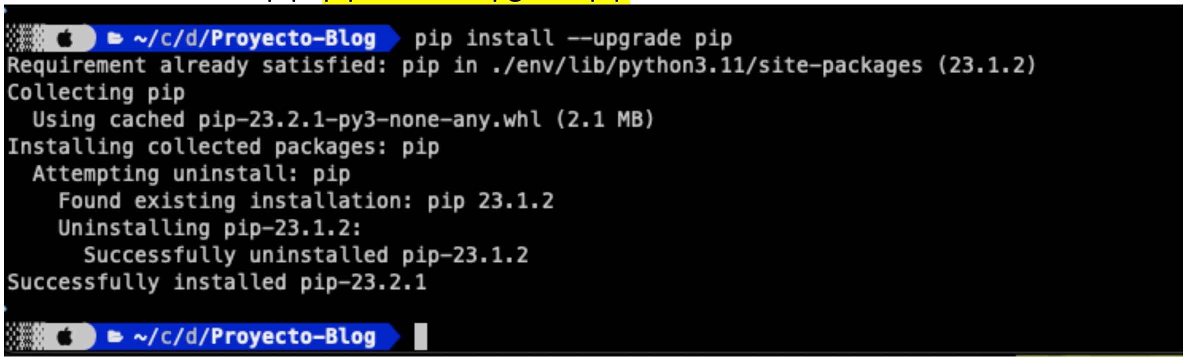
II. Installar django (*recuerde debe tener activado el enterno virtual)

Pip install django



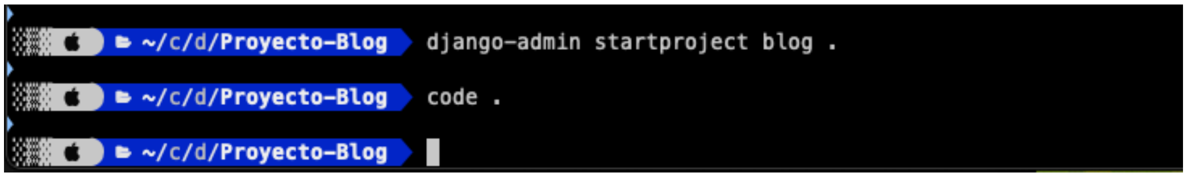
```
~/c/d/Proyecto-Blog  pip install django                                  ✔  Proyecto-Blog
Collecting django
  Using cached Django-4.2.4-py3-none-any.whl (8.0 MB)
Collecting asgiref<4,>=3.6.0 (from django)
  Using cached asgiref-3.7.2-py3-none-any.whl (24 kB)
Collecting sqlparse>=0.3.1 (from django)
  Using cached sqlparse-0.4.4-py3-none-any.whl (41 kB)
Installing collected packages: sqlparse, asgiref, django
Successfully installed asgiref-3.7.2 django-4.2.4 sqlparse-0.4.4

[notice] A new release of pip is available: 23.1.2 -> 23.2.1
[notice] To update, run: pip install --upgrade pip
~/c/d/Proyecto-Blog                                        ✔  took 4s  Proyecto-Blog
```

III. Actualizar el pip: pip install --upgrade pip



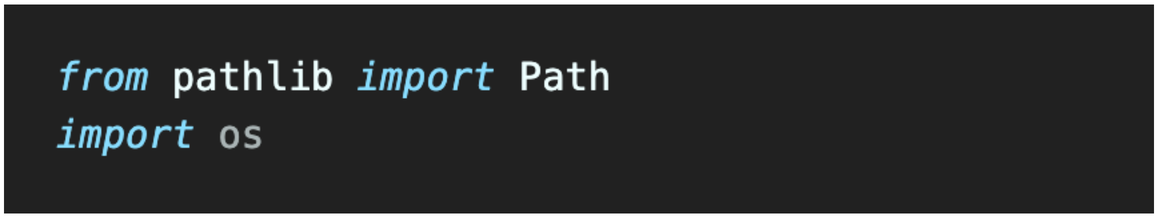
```
~/c/d/Proyecto-Blog  pip install --upgrade pip
Requirement already satisfied: pip in ./env/lib/python3.11/site-packages (23.1.2)
Collecting pip
  Using cached pip-23.2.1-py3-none-any.whl (2.1 MB)
Installing collected packages: pip
  Attempting uninstall: pip
    Found existing installation: pip 23.1.2
    Uninstalling pip-23.1.2:
      Successfully uninstalled pip-23.1.2
Successfully installed pip-23.2.1
~/c/d/Proyecto-Blog
```

IV. CREAR EL PROYECTO LLAMARLO BLOG. Abrirlo en el visual code .



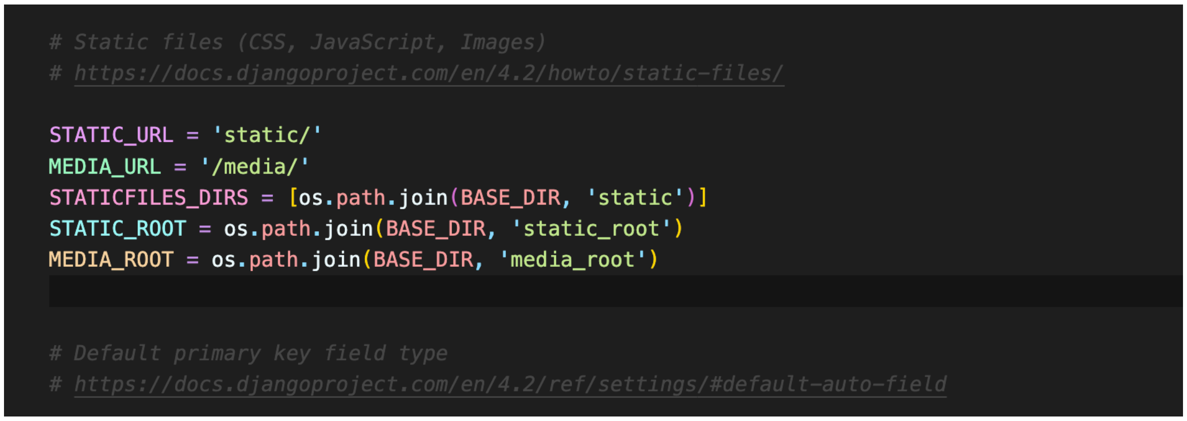
```
~/c/d/Proyecto-Blog  django-admin startproject blog .
~/c/d/Proyecto-Blog  code .
~/c/d/Proyecto-Blog
```

V. Configurar el proyecto preparando archivos y direcciones en el settings.py importamos el os y preparamos los archvios estaticos



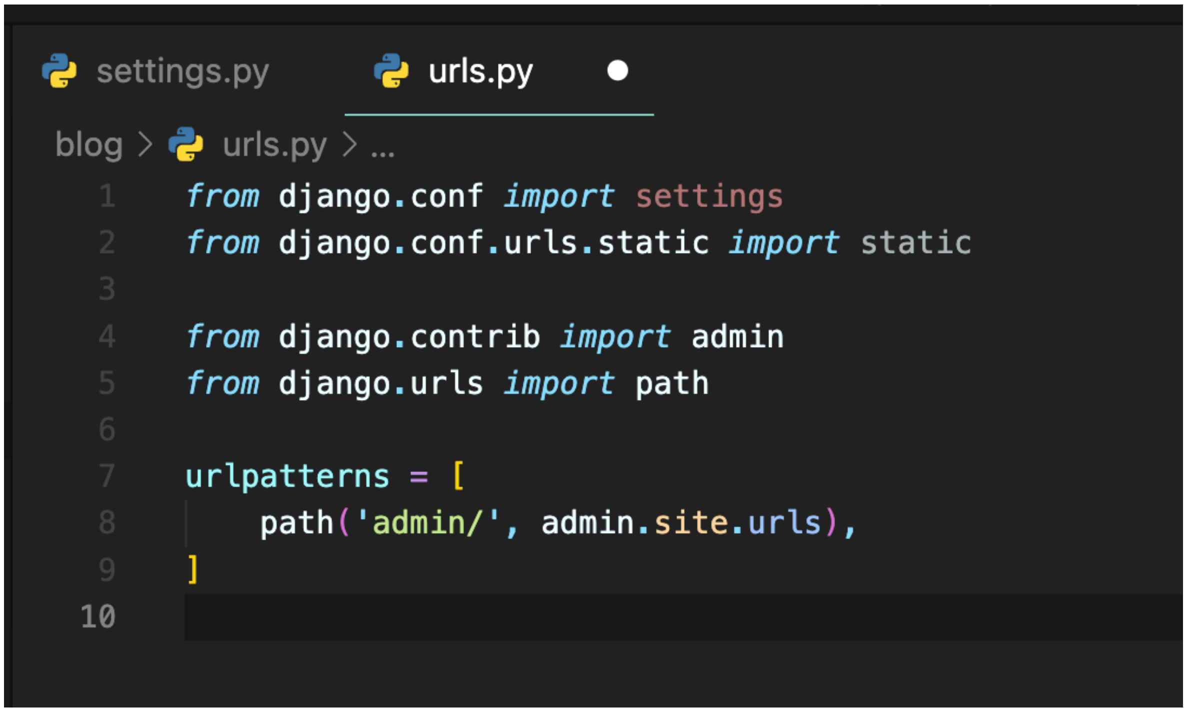
```python
from pathlib import Path
import os
```

VI. Configuramos el lugar de nuestros archivos siempre en settings.py estas configuraciones son para los archivos estaticos.



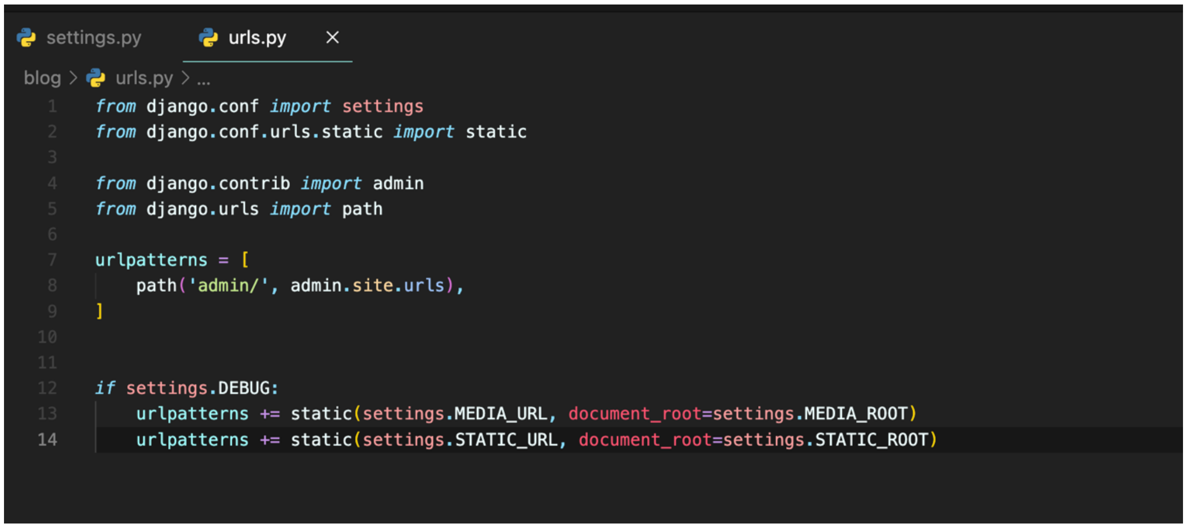
```python
# Static files (CSS, JavaScript, Images)
# https://docs.djangoproject.com/en/4.2/howto/static-files/

STATIC_URL = 'static/'
MEDIA_URL = '/media/'
STATICFILES_DIRS = [os.path.join(BASE_DIR, 'static')]
STATIC_ROOT = os.path.join(BASE_DIR, 'static_root')
MEDIA_ROOT = os.path.join(BASE_DIR, 'media_root')


# Default primary key field type
# https://docs.djangoproject.com/en/4.2/ref/settings/#default-auto-field
```

VII. VAMOS AL ARCHIVO URLS.PY del proyecto blog importamos los archivos settings y statics



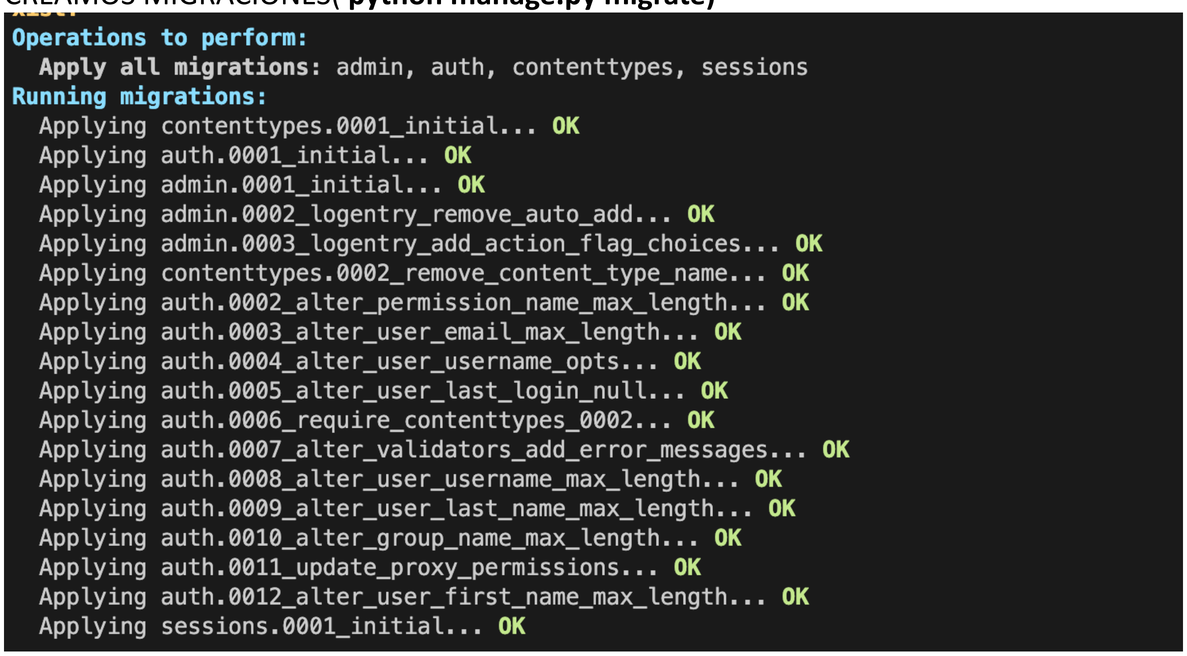
```python
from django.conf import settings
from django.conf.urls.static import static

from django.contrib import admin
from django.urls import path

urlpatterns = [
    path('admin/', admin.site.urls),
]

```

AGREGAMOS UN CONDICIONAL TIPO DEPURADOR O debug archivo final del urls.py del blog se deberia ver de esta manera.



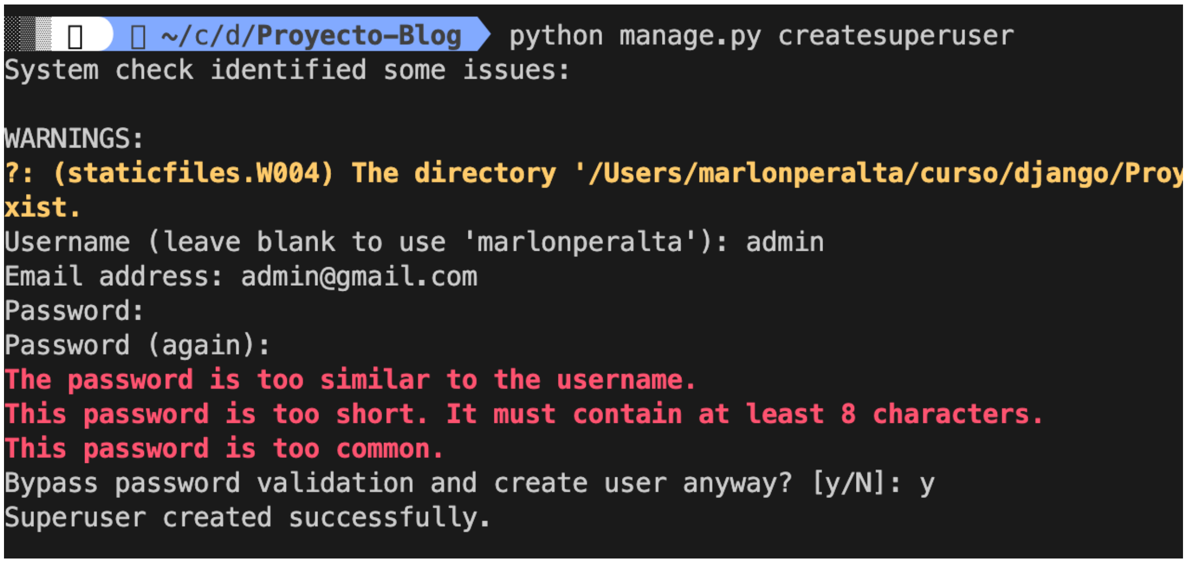
```python
from django.conf import settings
from django.conf.urls.static import static

from django.contrib import admin
from django.urls import path

urlpatterns = [
    path('admin/', admin.site.urls),
]


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

CREAMOS MIGRACIONES(**python manage.py migrate)**

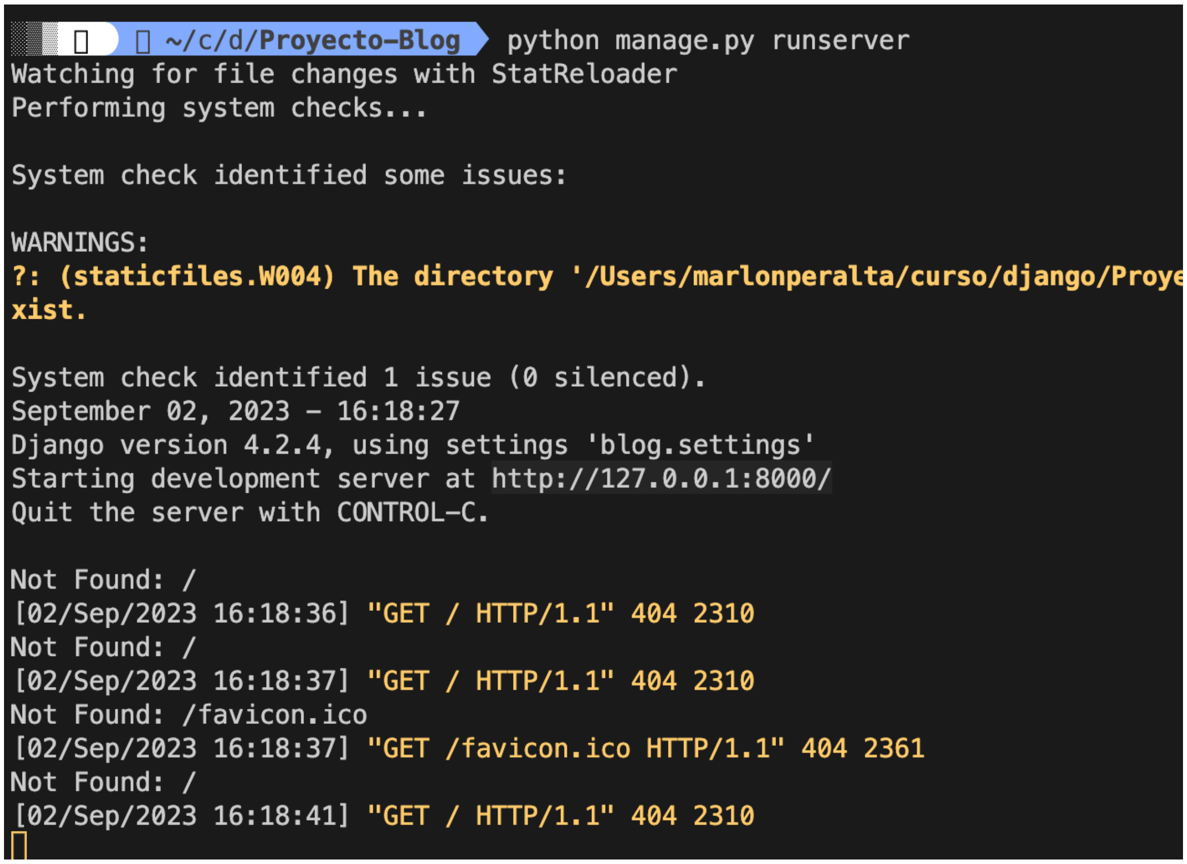


```
Operations to perform:
  Apply all migrations: admin, auth, contenttypes, sessions
Running migrations:
  Applying contenttypes.0001_initial... OK
  Applying auth.0001_initial... OK
  Applying admin.0001_initial... OK
  Applying admin.0002_logentry_remove_auto_add... OK
  Applying admin.0003_logentry_add_action_flag_choices... OK
  Applying contenttypes.0002_remove_content_type_name... OK
  Applying auth.0002_alter_permission_name_max_length... OK
  Applying auth.0003_alter_user_email_max_length... OK
  Applying auth.0004_alter_user_username_opts... OK
  Applying auth.0005_alter_user_last_login_null... OK
  Applying auth.0006_require_contenttypes_0002... OK
  Applying auth.0007_alter_validators_add_error_messages... OK
  Applying auth.0008_alter_user_username_max_length... OK
  Applying auth.0009_alter_user_last_name_max_length... OK
  Applying auth.0010_alter_group_name_max_length... OK
  Applying auth.0011_update_proxy_permissions... OK
  Applying auth.0012_alter_user_first_name_max_length... OK
  Applying sessions.0001_initial... OK
```

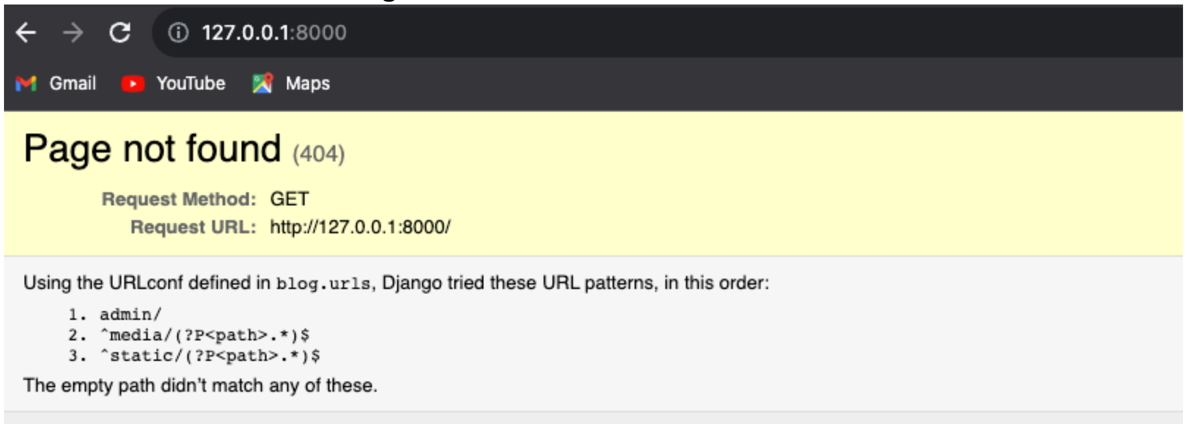
CREAMOS UN SUPER USUARIO:


```
 ~/c/d/Proyecto-Blog  python manage.py createsuperuser
System check identified some issues:

WARNINGS:
?: (staticfiles.W004) The directory '/Users/marlonperalta/curso/django/Proy
xist.
Username (leave blank to use 'marlonperalta'): admin
Email address: admin@gmail.com
Password:
Password (again):
The password is too similar to the username.
This password is too short. It must contain at least 8 characters.
This password is too common.
Bypass password validation and create user anyway? [y/N]: y
Superuser created successfully.
```

CORRAMOS NUESTRO SERVIDOR..


```
 ~/c/d/Proyecto-Blog  python manage.py runserver
Watching for file changes with StatReloader
Performing system checks...

System check identified some issues:

WARNINGS:
?: (staticfiles.W004) The directory '/Users/marlonperalta/curso/django/Proye
xist.

System check identified 1 issue (0 silenced).
September 02, 2023 - 16:18:27
Django version 4.2.4, using settings 'blog.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CONTROL-C.

Not Found: /
[02/Sep/2023 16:18:36] "GET / HTTP/1.1" 404 2310
Not Found: /
[02/Sep/2023 16:18:37] "GET / HTTP/1.1" 404 2310
Not Found: /favicon.ico
[02/Sep/2023 16:18:37] "GET /favicon.ico HTTP/1.1" 404 2361
Not Found: /
[02/Sep/2023 16:18:41] "GET / HTTP/1.1" 404 2310
```

NO SE ASUSTEN AL INICIO PRESENTARA EL SIGUIENTE ERROR, RECUERDEN NO HEMOS CONFIGURADO ARCHIVOS. Sigamos…



Page not found (404)

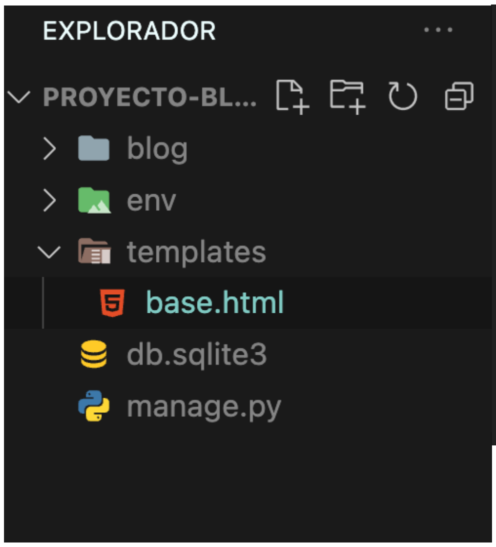
Request Method: GET
Request URL: http://127.0.0.1:8000/

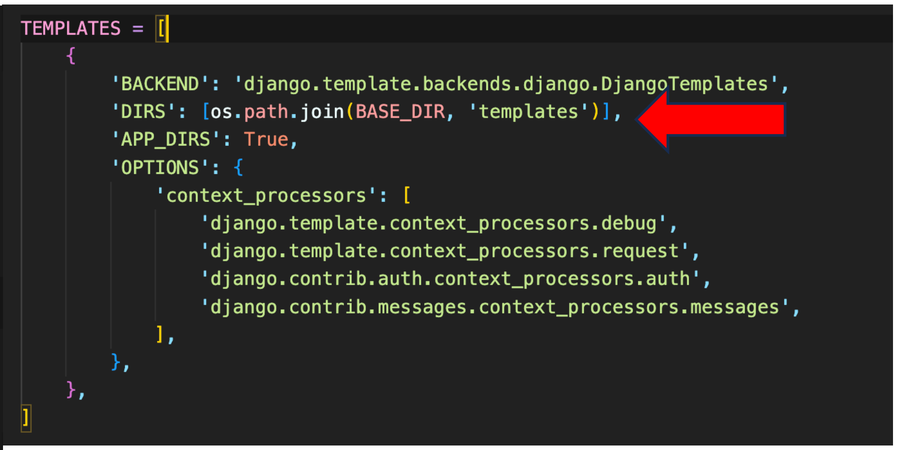
Using the URLconf defined in `blog.urls`, Django tried these URL patterns, in this order:

```
1. admin/
2. ^media/(?P<path>.*)$
3. ^static/(?P<path>.*)$
```

The empty path didn't match any of these.

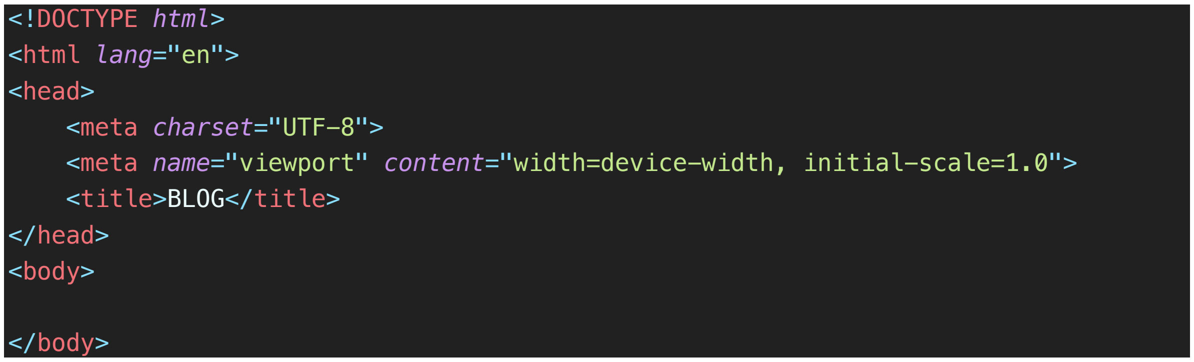
INDIQUEMOS DONDE SE ENCUENTRA NUESTRO DIRECTORIO TEMPLATES.
VAMOS a settings.py por lo tanto debemos primero crear la carpeta en el proyecto fuera de blog. Y un archivo llamado base.htmlbase.html





```
TEMPLATES = [
    {
        'BACKEND': 'django.template.backends.django.DjangoTemplates',
        'DIRS': [os.path.join(BASE_DIR, 'templates')],
        'APP_DIRS': True,
        'OPTIONS': {
            'context_processors': [
                'django.template.context_processors.debug',
                'django.template.context_processors.request',
                'django.contrib.auth.context_processors.auth',
                'django.contrib.messages.context_processors.messages',
            ],
        },
    },
]
```

INSTALLAR BASE DEL DISENO USAREMOS BUSTRAT.
AGREMOS LO SIGUIENTE EN NUESTRO ARCHIVO BASE.HTML



```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>BLOG</title>
</head>
<body>

</body>
```

```
</html>
```

Agregamos nuestro archivo CSS de bootstrap

```
<link rel="stylesheet"
href="https://cdn.jsdelivr.net/npm/bootstrap@4.6.2/dist/css/bootstrap.min
.css" integrity="sha384-
xOolHFLEh07PJGoPkLv1IbcEPTNtaed2xpHsD9ESMhqIYd0nLMwNLD69Npy4HI+N"
crossorigin="anonymous">
```

Nuestro archivo final quedaria de la siguiente manera

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>BLOG</title>
    <link rel="stylesheet"
href="https://cdn.jsdelivr.net/npm/bootstrap@4.6.2/dist/css/bootstrap.min.css"
integrity="sha384-
xOolHFLEh07PJGoPkLv1IbcEPTNtaed2xpHsD9ESMhqIYd0nLMwNLD69Npy4HI+N"
crossorigin="anonymous">
</head>
<body>

</body>
</html>
```

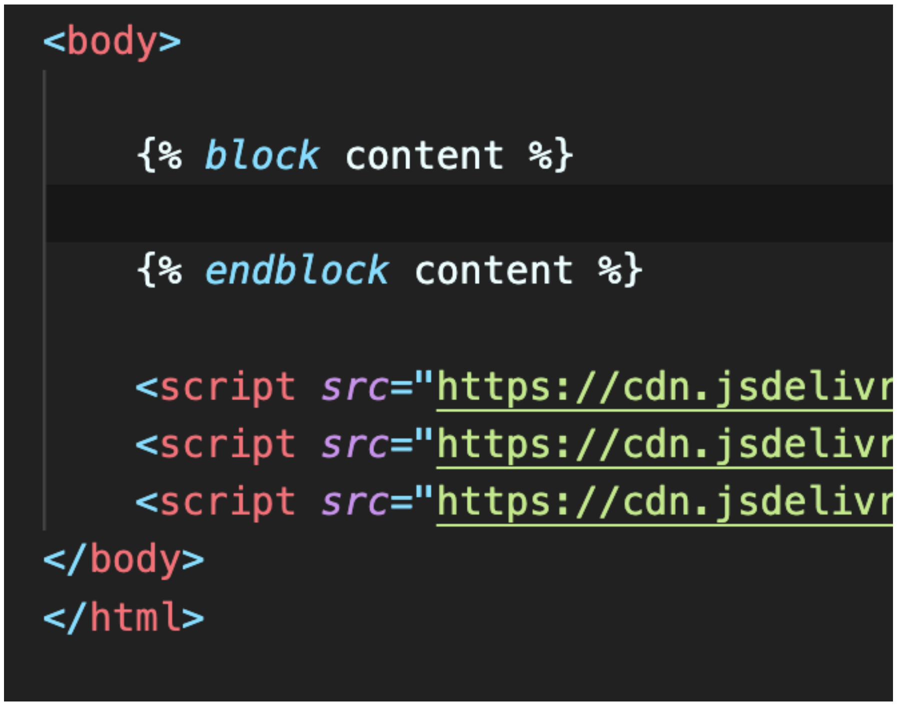
Agremos tambien los siguientes archivos javascript. En el bady de nuestro archivo html.

```
<script
src="https://cdn.jsdelivr.net/npm/jquery@3.5.1/dist/jquery.slim.min.js"
integrity="sha384-
DfXdz2htPH0lsSSs5nCTpuj/zy4C+OGpamoFVy38MVBnE+IbbVYUew+OrCXaRkfj"
crossorigin="anonymous"></script>
<script
src="https://cdn.jsdelivr.net/npm/popper.js@1.16.1/dist/umd/popper.min.js
" integrity="sha384-
9/reFTGAW83EW2RDu2S0VKaIzap3H66lZH81PoYlFhbGU+6BZp6G7niu735Sk7lN"
crossorigin="anonymous"></script>
<script
src="https://cdn.jsdelivr.net/npm/bootstrap@4.6.2/dist/js/bootstrap.min.j
s" integrity="sha384-
+sLIOodYLS7CIrQpBjl+C7nPvqq+FbNUBDunl/OZv93DB7Ln/533i8e/mZXLi/P+"
crossorigin="anonymous"></script>
```

EL FINAL DE NUESTRO ARCHIVO QUEDARIA ASI.

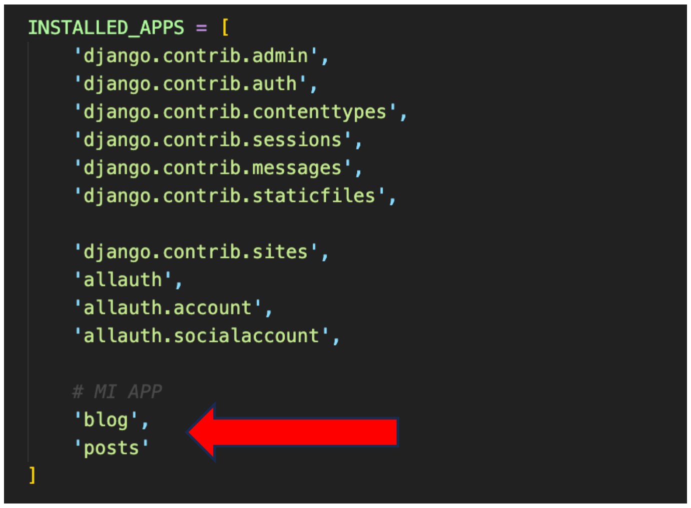
```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>BLOG</title>
    <link rel="stylesheet"
href="https://cdn.jsdelivr.net/npm/bootstrap@4.6.2/dist/css/bootstrap.min.css"
integrity="sha384-
xOolHFLEh07PJGoPkLv1IbcEPTNtaed2xpHsD9ESMhqIYd0nLMwNLD69Npy4HI+N"
crossorigin="anonymous">
</head>
<body>


    <script
src="https://cdn.jsdelivr.net/npm/jquery@3.5.1/dist/jquery.slim.min.js"
integrity="sha384-
DfXdz2htPH0lsSSs5nCTpuj/zy4C+OGpamoFVy38MVBnE+IbbVYUew+OrCXaRkfj"
crossorigin="anonymous"></script>
    <script
src="https://cdn.jsdelivr.net/npm/popper.js@1.16.1/dist/umd/popper.min.js"
integrity="sha384-
9/reFTGAW83EW2RDu2S0VKaIzap3H66lZH81PoYlFhbGU+6BZp6G7niu735Sk7lN"
crossorigin="anonymous"></script>
    <script
src="https://cdn.jsdelivr.net/npm/bootstrap@4.6.2/dist/js/bootstrap.min.js"
integrity="sha384-
+sLIOodYLS7CIrQpBjl+C7nPvqq+FbNUBDunl/OZv93DB7Ln/533i8e/mZXLi/P+"
crossorigin="anonymous"></script>
</body>
</html>
```

Hasta el momento solo estamos usando archivos css, y js, para implementar el diseño ahora debemos de configurar nuestros block content en el mismo archivo donde ubicariamos nuestro contenid.



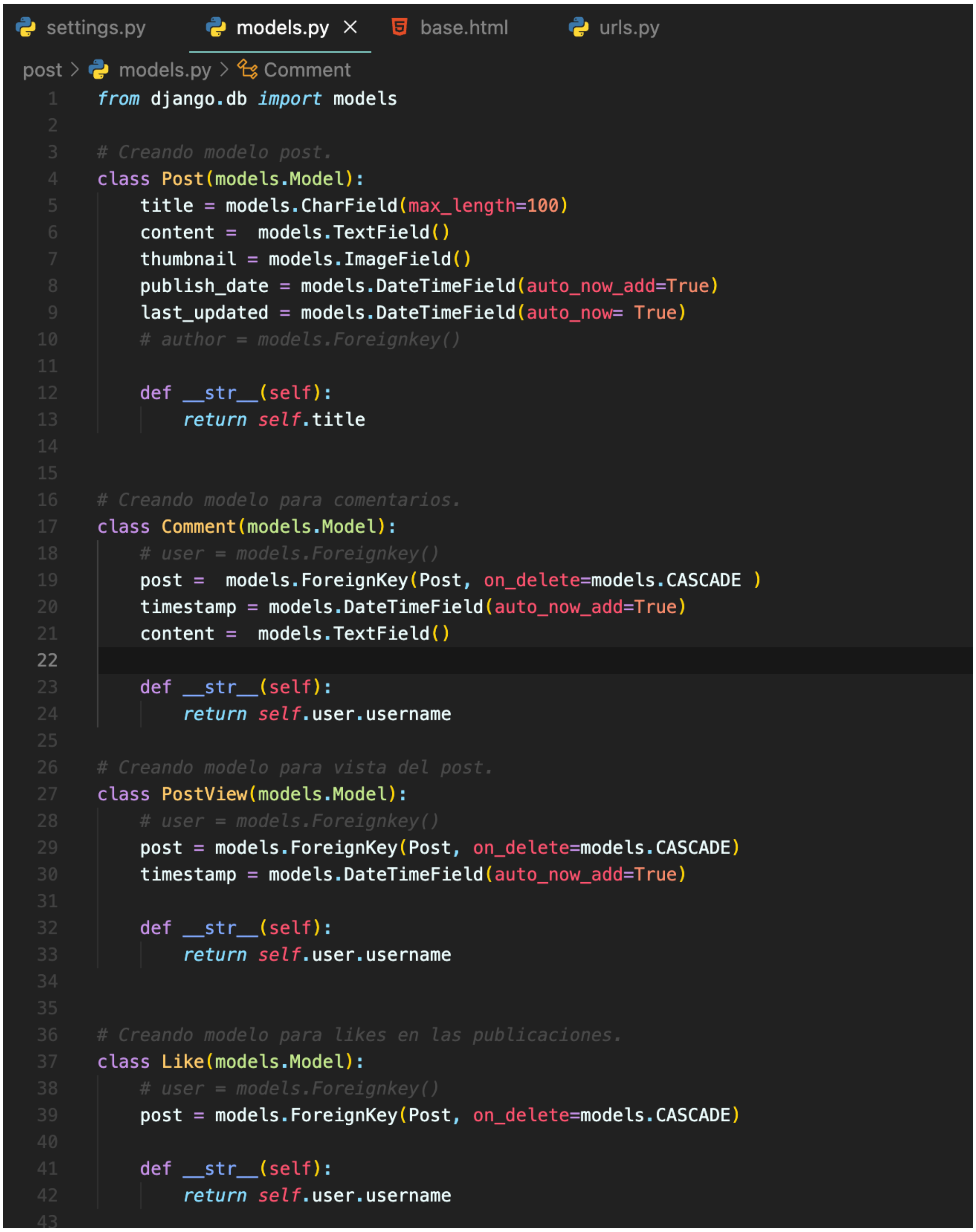
```
<body>

    {% block content %}

    {% endblock content %}

    <script src="https://cdn.jsdelivr
    <script src="https://cdn.jsdelivr
    <script src="https://cdn.jsdelivr
</body>
</html>
```

AHORA DEFINIREMOS NUESTRA PRIMERA APP. Llamada post.
python manage.py startapp posts, agregarla al archive settings.py



```
INSTALLED_APPS = [
    'django.contrib.admin',
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',

    'django.contrib.sites',
    'allauth',
    'allauth.account',
    'allauth.socialaccount',

    # MI APP
    'blog',
    'posts'
]
```

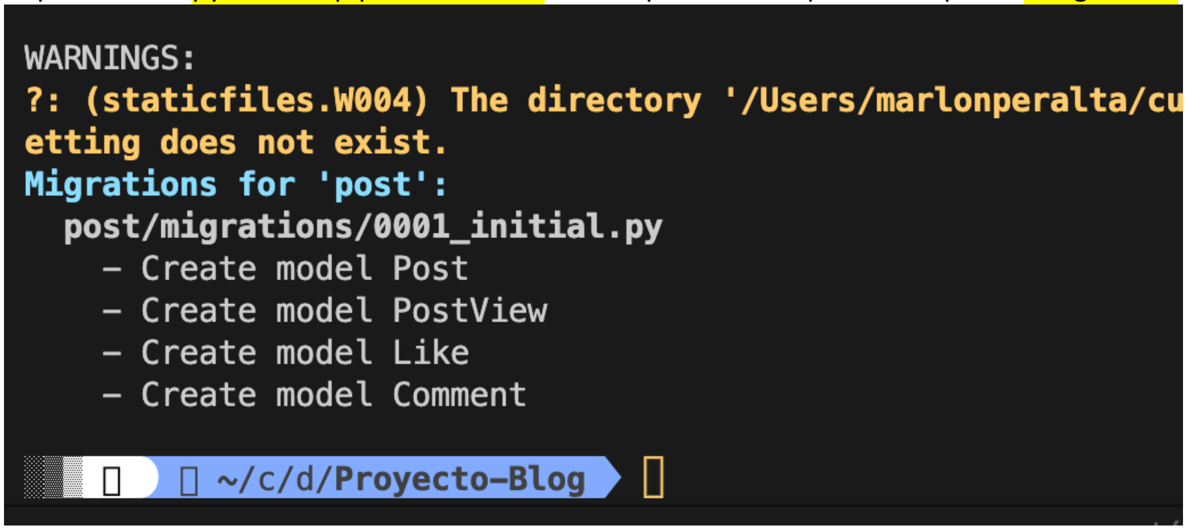
CREAMOS NUESTROS MODELOS EN models.py de la aplicación post:



```python
from django.db import models

# Creando modelo post.
class Post(models.Model):
    title = models.CharField(max_length=100)
    content =  models.TextField()
    thumbnail = models.ImageField()
    publish_date = models.DateTimeField(auto_now_add=True)
    last_updated = models.DateTimeField(auto_now= True)
    # author = models.Foreignkey()

    def __str__(self):
        return self.title


# Creando modelo para comentarios.
class Comment(models.Model):
    # user = models.Foreignkey()
    post =  models.ForeignKey(Post, on_delete=models.CASCADE )
    timestamp = models.DateTimeField(auto_now_add=True)
    content =  models.TextField()

    def __str__(self):
        return self.user.username

# Creando modelo para vista del post.
class PostView(models.Model):
    # user = models.Foreignkey()
    post = models.ForeignKey(Post, on_delete=models.CASCADE)
    timestamp = models.DateTimeField(auto_now_add=True)

    def __str__(self):
        return self.user.username


# Creando modelo para likes en las publicaciones.
class Like(models.Model):
    # user = models.Foreignkey()
    post = models.ForeignKey(Post, on_delete=models.CASCADE)

    def __str__(self):
        return self.user.username
```

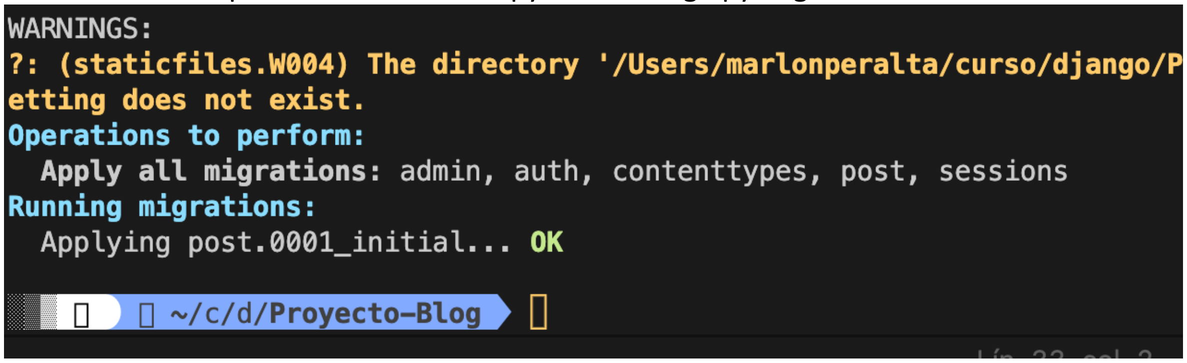
CREAMOS NUESTRAS MIGRACIONES..
Recuerden python manage.py makemigrations

Si al crear las migraciones les da error por los tipos de campos deben instalar algunas dependencias python -m pip install Pillow esta dependecia es por el campo de Imagenfield



```
WARNINGS:
?: (staticfiles.W004) The directory '/Users/marlonperalta/cu
etting does not exist.
Migrations for 'post':
  post/migrations/0001_initial.py
    - Create model Post
    - Create model PostView
    - Create model Like
    - Create model Comment

 ~/c/d/Proyecto-Blog
```

Posteriormente aplicamos el comando python manage.py migrate



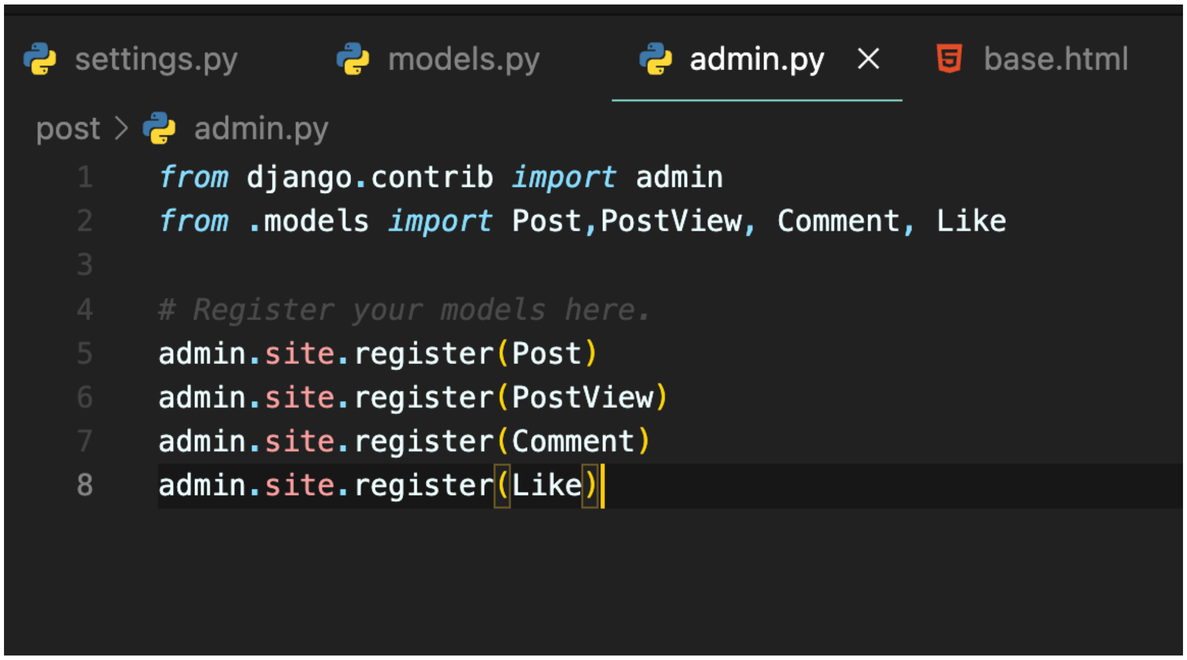
```
WARNINGS:
?: (staticfiles.W004) The directory '/Users/marlonperalta/curso/django/P
etting does not exist.
Operations to perform:
  Apply all migrations: admin, auth, contenttypes, post, sessions
Running migrations:
  Applying post.0001_initial... OK

 ~/c/d/Proyecto-Blog
```

Revisen y analicen el archivo 0001_initial.py que se crea en migrations de la aplicación post

AGREGEMOS NUESTROS MODELOS AL admin.py del blog, recordemos que debemos importar primero.

```
from .models import Post,PostView, Comment, Like
```

luego registrarlos. Nuestro archive deberia de quedar de la liguiente manera.



```python
from django.contrib import admin
from .models import Post,PostView, Comment, Like

# Register your models here.
admin.site.register(Post)
admin.site.register(PostView)
admin.site.register(Comment)
admin.site.register(Like)
```

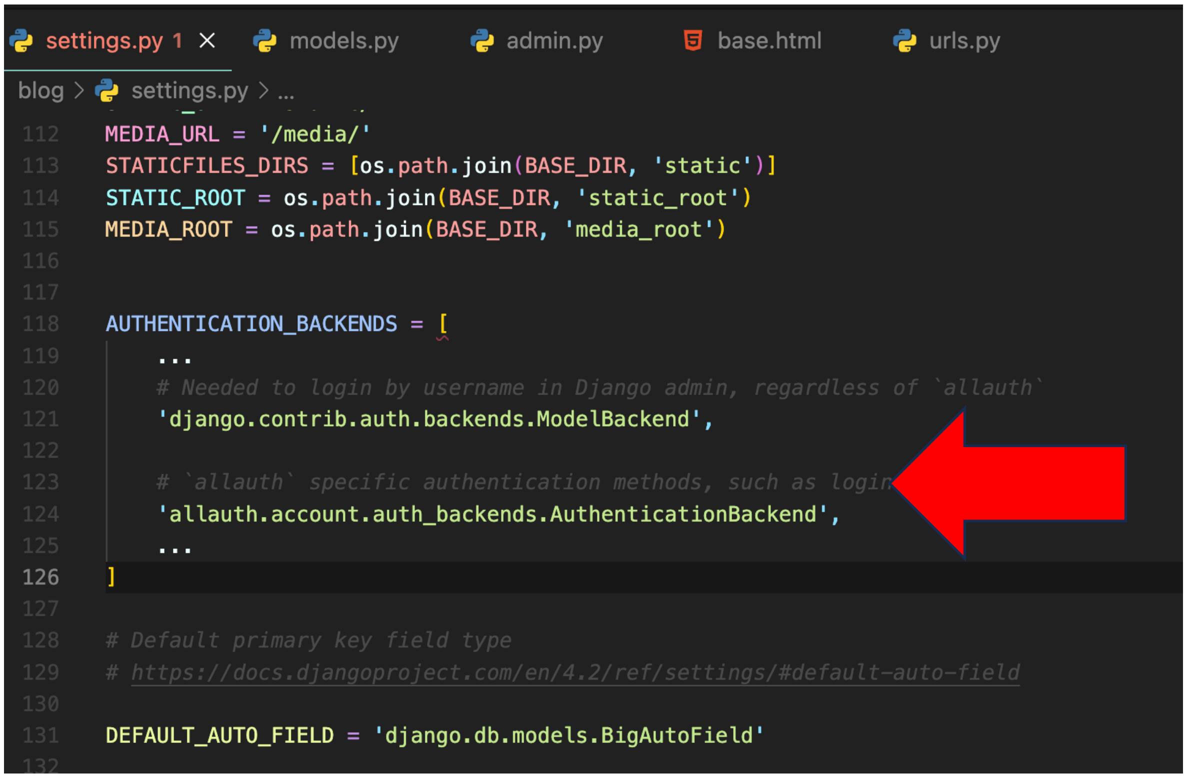
PARA PODER TRABAJAR CON AUTENTICACIONES DJANGO TIENE el django-allauth, que se utiliza como dependencias. Por tanto se usa pip install django-allauth

Una vez se instala debemos copiar los authentication backends. En el archivo settings.py del proyecto.

```
AUTHENTICATION_BACKENDS = [
    ...
    # Needed to login by username in Django admin, regardless of
`allauth`
    'django.contrib.auth.backends.ModelBackend',

    # `allauth` specific authentication methods, such as login by email
    'allauth.account.auth_backends.AuthenticationBackend',
    ...
]
```

Esta información la pueden encontrar en la siguiente dirección electronica https://django-allauth.readthedocs.io/en/latest/installation/quickstart.html#post-installation

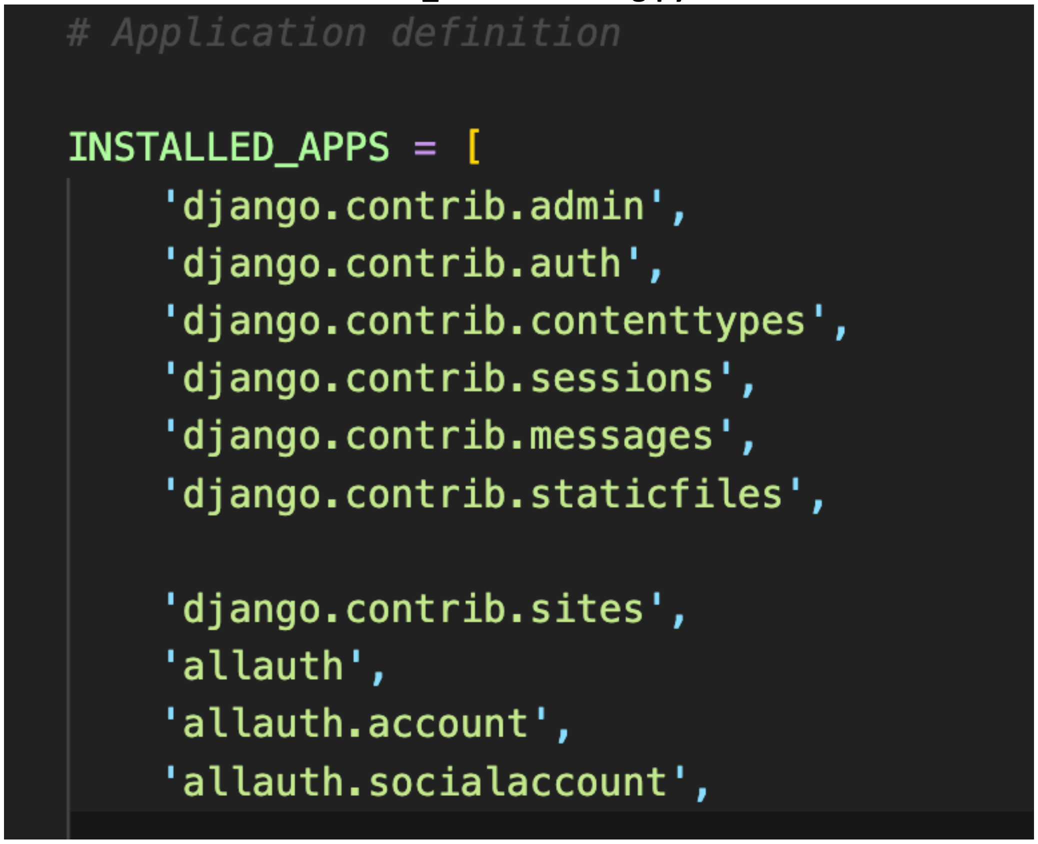
Nuestro archivo settings.py quedaria de la siguiente manera al final.


settings.py 1 × models.py admin.py base.html urls.py

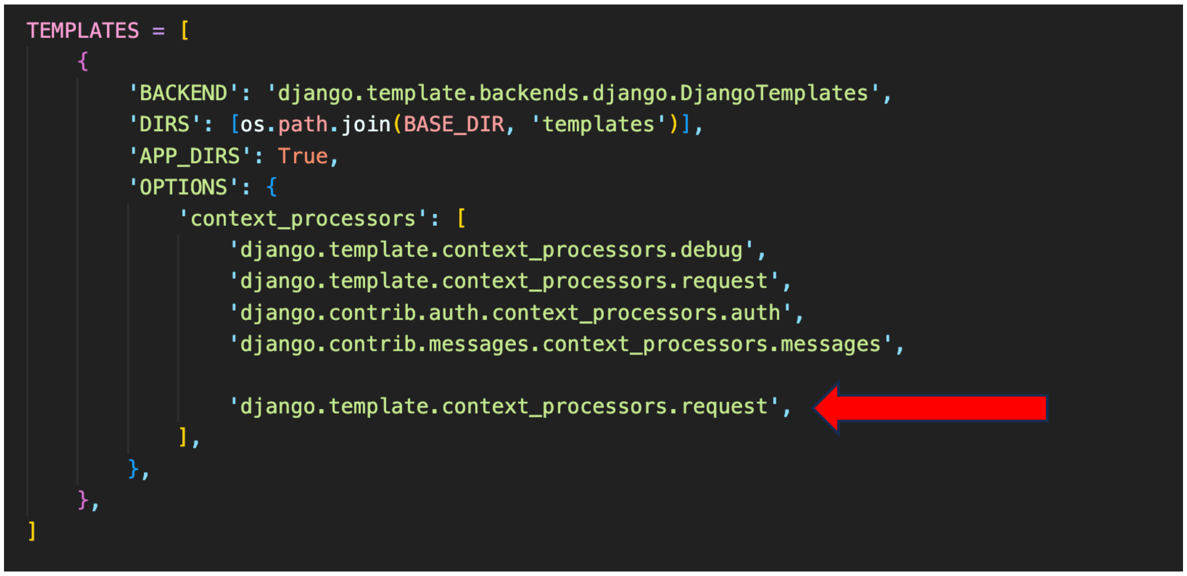
blog > settings.py > ...

```
MEDIA_URL = '/media/'
STATICFILES_DIRS = [os.path.join(BASE_DIR, 'static')]
STATIC_ROOT = os.path.join(BASE_DIR, 'static_root')
MEDIA_ROOT = os.path.join(BASE_DIR, 'media_root')


AUTHENTICATION_BACKENDS = [
    ...
    # Needed to login by username in Django admin, regardless of `allauth`
    'django.contrib.auth.backends.ModelBackend',

    # `allauth` specific authentication methods, such as login
    'allauth.account.auth_backends.AuthenticationBackend',
    ...
]

# Default primary key field type
# https://docs.djangoproject.com/en/4.2/ref/settings/#default-auto-field

DEFAULT_AUTO_FIELD = 'django.db.models.BigAutoField'
```

AGREGAR LOS APPS EN INSTALL_APPS DE setting.py


```
# Application definition

INSTALLED_APPS = [
    'django.contrib.admin',
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',

    'django.contrib.sites',
    'allauth',
    'allauth.account',
    'allauth.socialaccount',
```

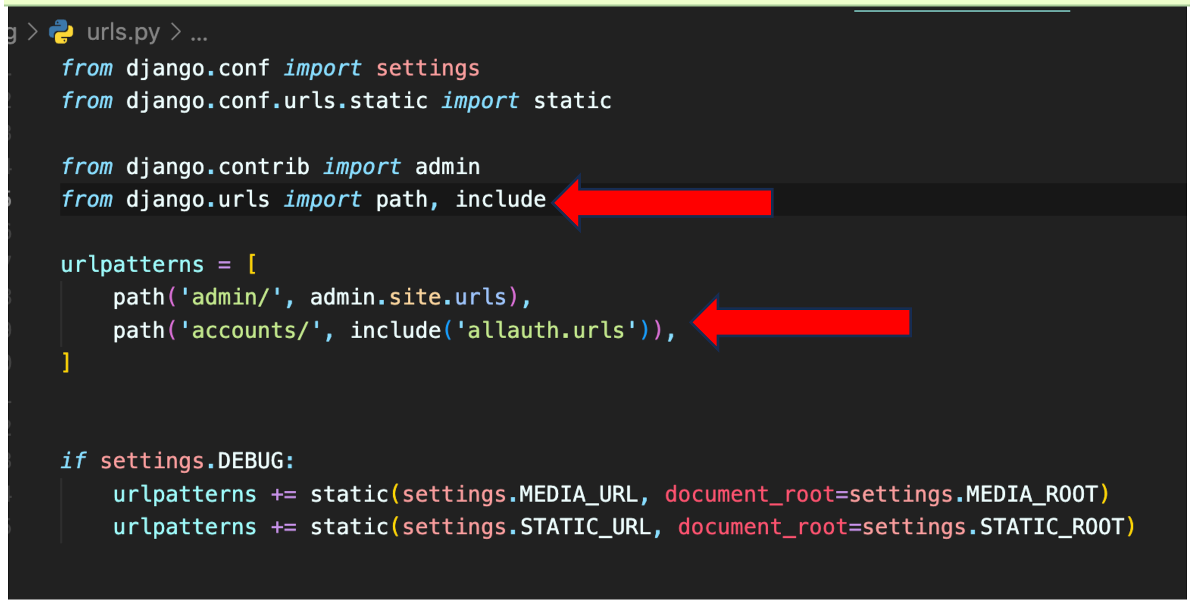
Y A LA CONFIGURACIÓN DEL TEMPLATE



```
TEMPLATES = [
    {
        'BACKEND': 'django.template.backends.django.DjangoTemplates',
        'DIRS': [os.path.join(BASE_DIR, 'templates')],
        'APP_DIRS': True,
        'OPTIONS': {
            'context_processors': [
                'django.template.context_processors.debug',
                'django.template.context_processors.request',
                'django.contrib.auth.context_processors.auth',
                'django.contrib.messages.context_processors.messages',

                'django.template.context_processors.request',
            ],
        },
    },
]
```

EN urls.py del blog agregar la siguiente path.

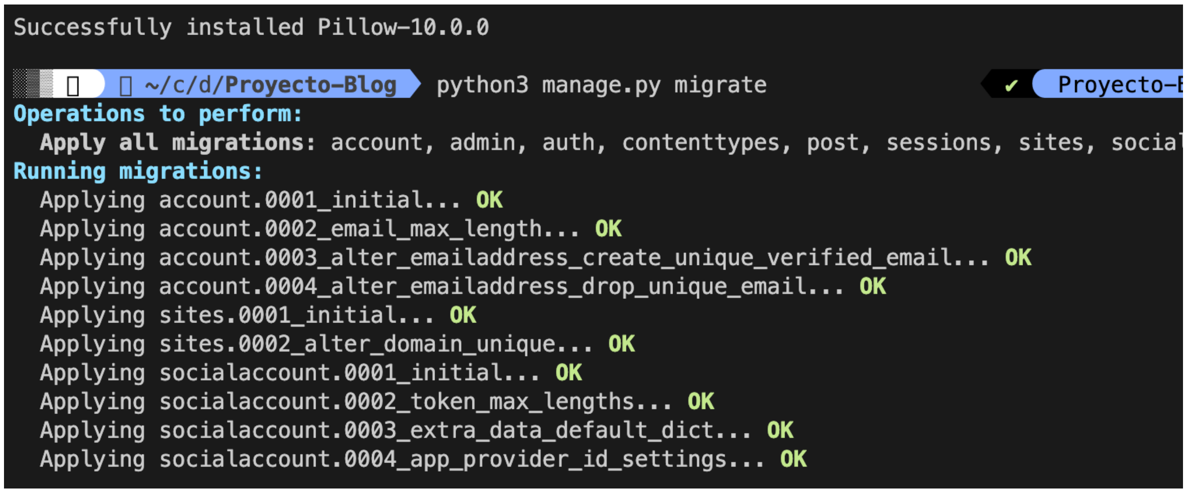
```
path('accounts/', include('allauth.urls')),
```



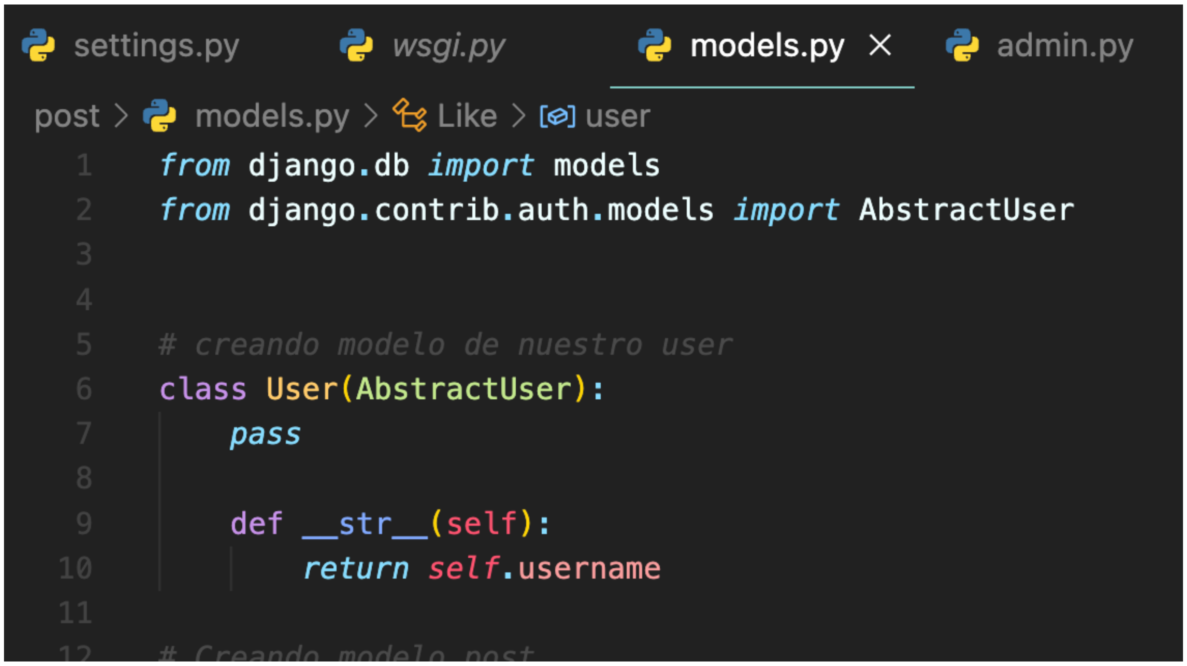
g > urls.py > ...

```
from django.conf import settings
from django.conf.urls.static import static

from django.contrib import admin
from django.urls import path, include

urlpatterns = [
    path('admin/', admin.site.urls),
    path('accounts/', include('allauth.urls')),
]


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

CORREMOS DE NUEVO LAS MIGRACIONES. Python manage.py migrate, RECUERDEN SI DA ERROR probar con python3 manage.py migrate

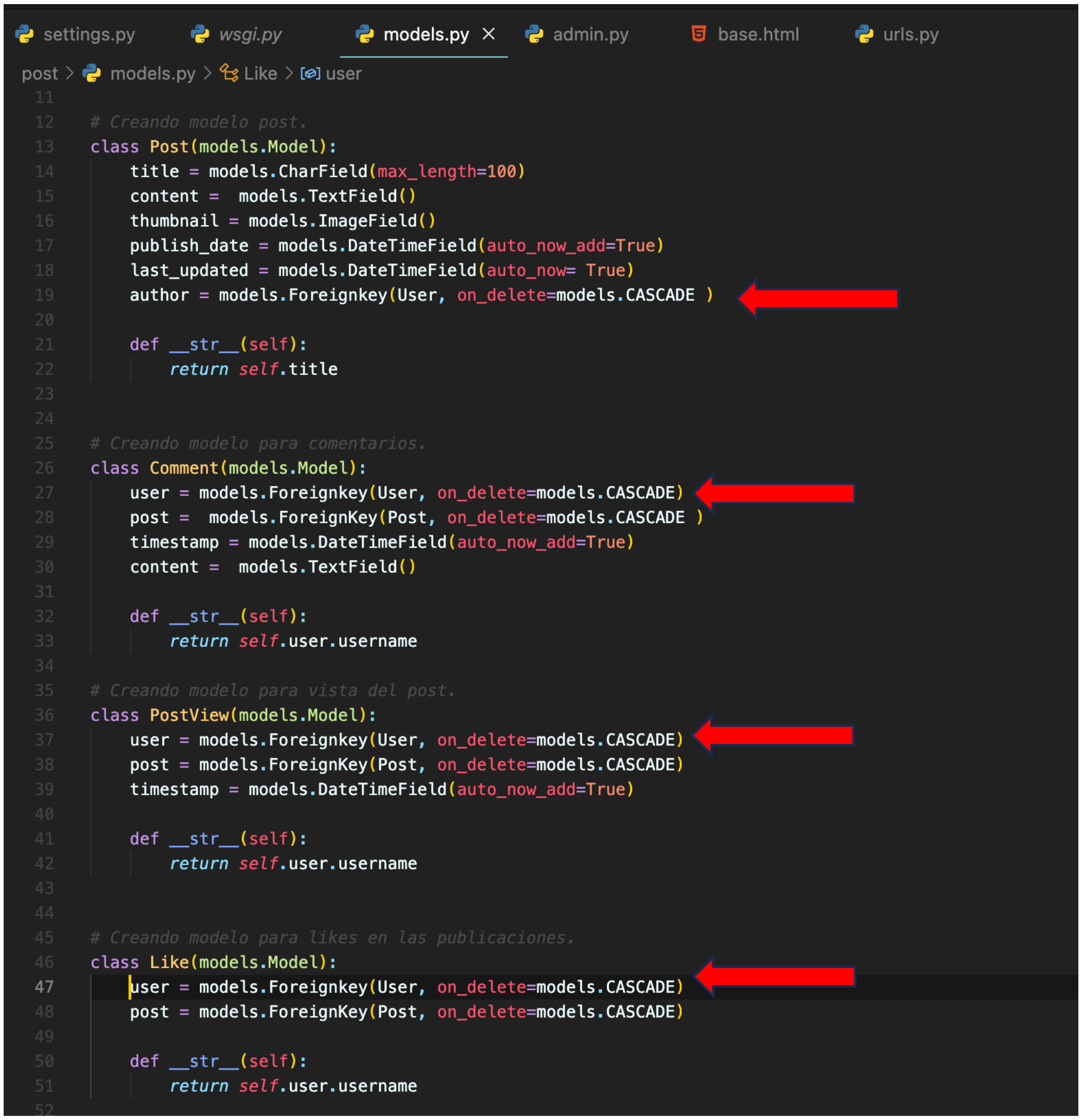

```
Successfully installed Pillow-10.0.0

 ~/c/d/Proyecto-Blog  python3 manage.py migrate                              ✔  Proyecto-B
Operations to perform:
  Apply all migrations: account, admin, auth, contenttypes, post, sessions, sites, socia
Running migrations:
  Applying account.0001_initial... OK
  Applying account.0002_email_max_length... OK
  Applying account.0003_alter_emailaddress_create_unique_verified_email... OK
  Applying account.0004_alter_emailaddress_drop_unique_email... OK
  Applying sites.0001_initial... OK
  Applying sites.0002_alter_domain_unique... OK
  Applying socialaccount.0001_initial... OK
  Applying socialaccount.0002_token_max_lengths... OK
  Applying socialaccount.0003_extra_data_default_dict... OK
  Applying socialaccount.0004_app_provider_id_settings... OK
```

CONFIGURAR USUARIO DE AUTENTICACIÓN EN APP de post
Vamos a models.py de la aplicación post.
Importamos el usuario base. Para ser heredado en user.



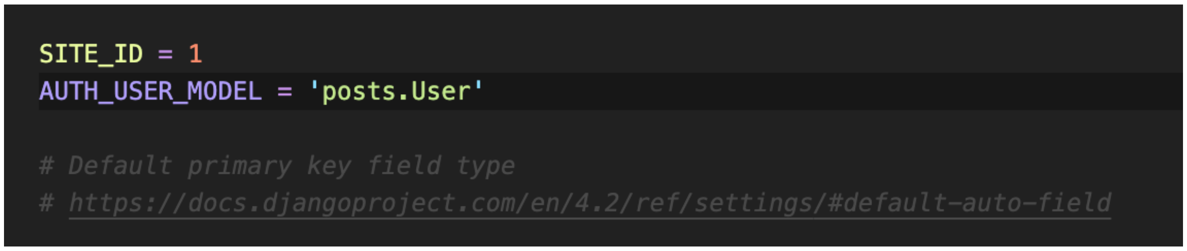
```python
from django.db import models
from django.contrib.auth.models import AbstractUser


# creando modelo de nuestro user
class User(AbstractUser):
    pass

    def __str__(self):
        return self.username

# Creando modelo post
```

Asi mismo agregamos las relaciones de los usuarios en el resto de los modelos.



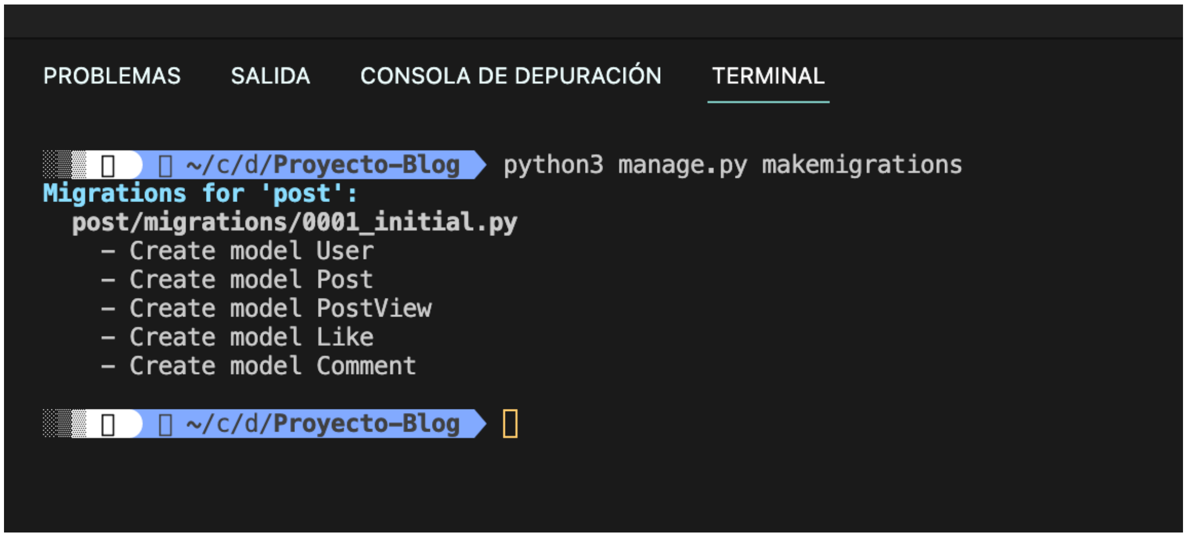
```python

# Creando modelo post.
class Post(models.Model):
    title = models.CharField(max_length=100)
    content =  models.TextField()
    thumbnail = models.ImageField()
    publish_date = models.DateTimeField(auto_now_add=True)
    last_updated = models.DateTimeField(auto_now= True)
    author = models.Foreignkey(User, on_delete=models.CASCADE )

    def __str__(self):
        return self.title


# Creando modelo para comentarios.
class Comment(models.Model):
    user = models.Foreignkey(User, on_delete=models.CASCADE)
    post =  models.ForeignKey(Post, on_delete=models.CASCADE )
    timestamp = models.DateTimeField(auto_now_add=True)
    content =  models.TextField()

    def __str__(self):
        return self.user.username

# Creando modelo para vista del post.
class PostView(models.Model):
    user = models.Foreignkey(User, on_delete=models.CASCADE)
    post = models.ForeignKey(Post, on_delete=models.CASCADE)
    timestamp = models.DateTimeField(auto_now_add=True)

    def __str__(self):
        return self.user.username


# Creando modelo para likes en las publicaciones.
class Like(models.Model):
    user = models.Foreignkey(User, on_delete=models.CASCADE)
    post = models.ForeignKey(Post, on_delete=models.CASCADE)

    def __str__(self):
        return self.user.username

```

ESPECIFICAMOS NUESTRO USER DE ALLAUTH EN settings.py



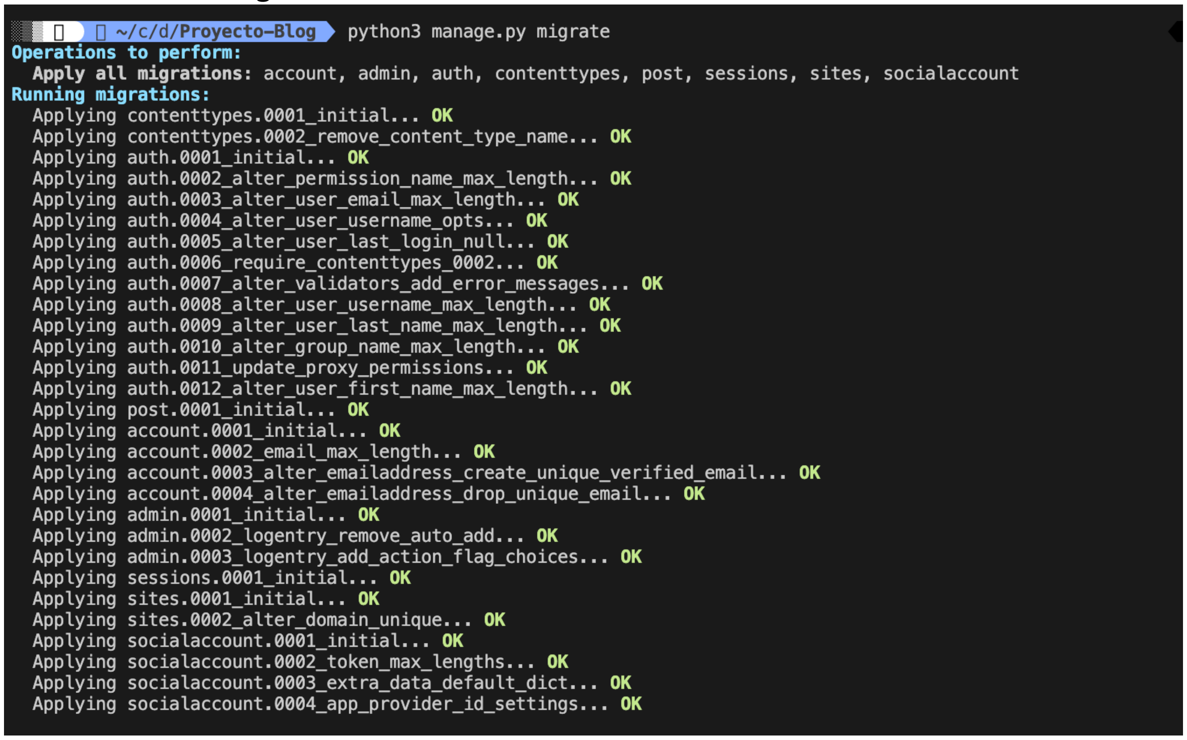
```python
SITE_ID = 1
AUTH_USER_MODEL = 'posts.User'

# Default primary key field type
# https://docs.djangoproject.com/en/4.2/ref/settings/#default-auto-field
```

VOLVEMOS HACER LAS MIGRACIONES, PRIMERAMENTE DEBES DE BORRAR EL ARCHIVO DE LAS PRIMERAS MIGRACIONES Y LA BASE DE DATOS creada de las migraciones anteriores.



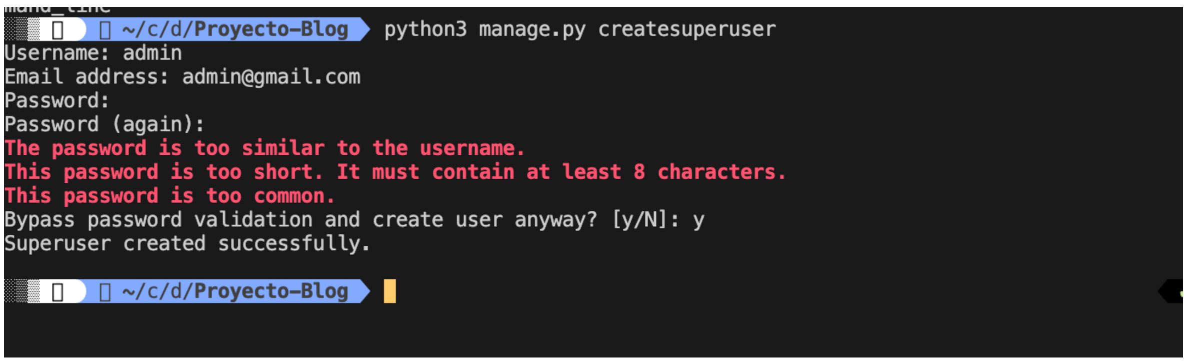
```
~/c/d/Proyecto-Blog  python3 manage.py makemigrations
Migrations for 'post':
  post/migrations/0001_initial.py
    - Create model User
    - Create model Post
    - Create model PostView
    - Create model Like
    - Create model Comment

~/c/d/Proyecto-Blog
```

Construimos las migraciones



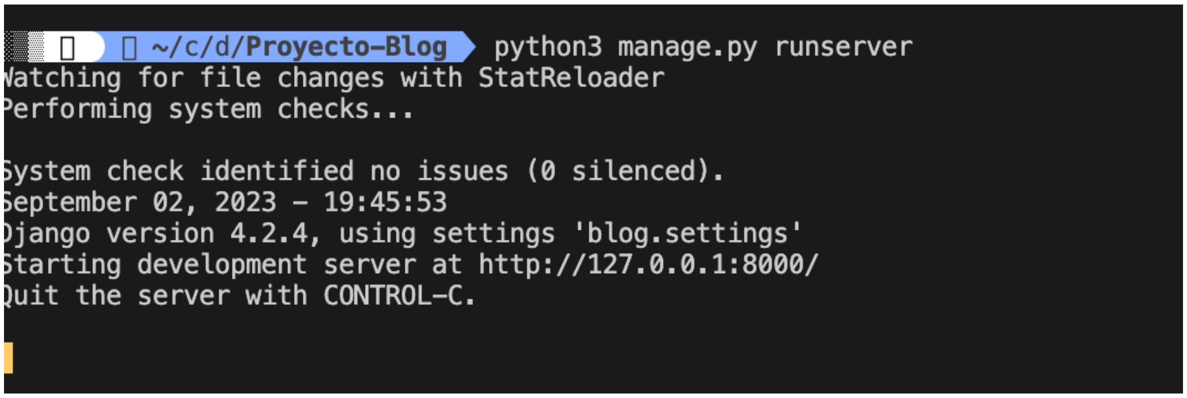
```
~/c/d/Proyecto-Blog  python3 manage.py migrate
Operations to perform:
  Apply all migrations: account, admin, auth, contenttypes, post, sessions, sites, socialaccount
Running migrations:
  Applying contenttypes.0001_initial... OK
  Applying contenttypes.0002_remove_content_type_name... OK
  Applying auth.0001_initial... OK
  Applying auth.0002_alter_permission_name_max_length... OK
  Applying auth.0003_alter_user_email_max_length... OK
  Applying auth.0004_alter_user_username_opts... OK
  Applying auth.0005_alter_user_last_login_null... OK
  Applying auth.0006_require_contenttypes_0002... OK
  Applying auth.0007_alter_validators_add_error_messages... OK
  Applying auth.0008_alter_user_username_max_length... OK
  Applying auth.0009_alter_user_last_name_max_length... OK
  Applying auth.0010_alter_group_name_max_length... OK
  Applying auth.0011_update_proxy_permissions... OK
  Applying auth.0012_alter_user_first_name_max_length... OK
  Applying post.0001_initial... OK
  Applying account.0001_initial... OK
  Applying account.0002_email_max_length... OK
  Applying account.0003_alter_emailaddress_create_unique_verified_email... OK
  Applying account.0004_alter_emailaddress_drop_unique_email... OK
  Applying admin.0001_initial... OK
  Applying admin.0002_logentry_remove_auto_add... OK
  Applying admin.0003_logentry_add_action_flag_choices... OK
  Applying sessions.0001_initial... OK
  Applying sites.0001_initial... OK
  Applying sites.0002_alter_domain_unique... OK
  Applying socialaccount.0001_initial... OK
  Applying socialaccount.0002_token_max_lengths... OK
  Applying socialaccount.0003_extra_data_default_dict... OK
  Applying socialaccount.0004_app_provider_id_settings... OK
```

CREAMOS DE NUEVO NUESTRO SUPERUSIARIO



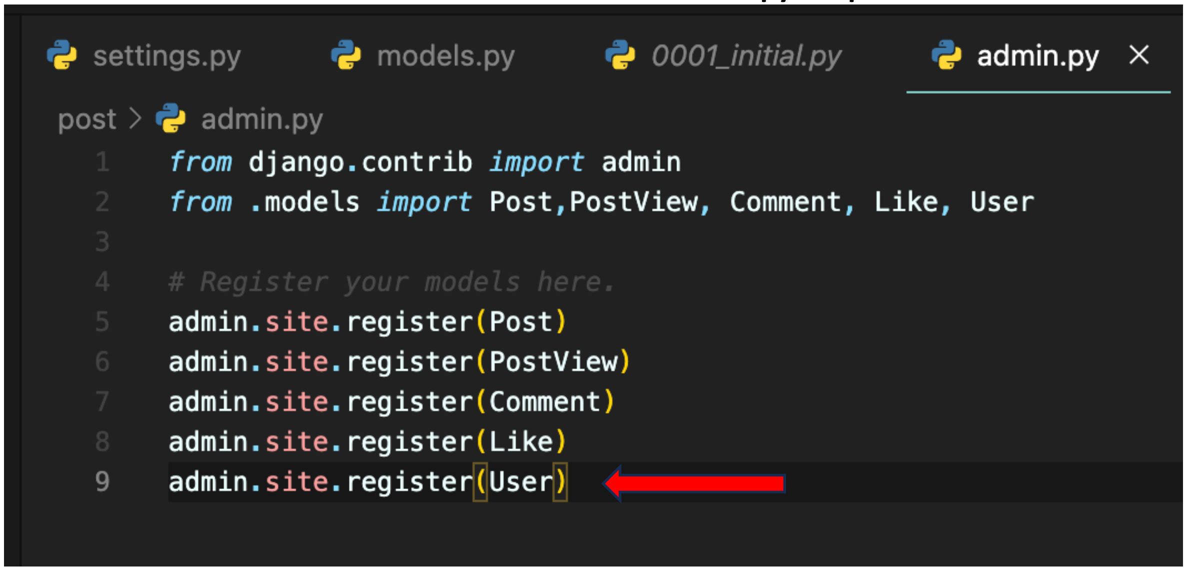
```
 ~/c/d/Proyecto-Blog  python3 manage.py createsuperuser
Username: admin
Email address: admin@gmail.com
Password:
Password (again):
The password is too similar to the username.
This password is too short. It must contain at least 8 characters.
This password is too common.
Bypass password validation and create user anyway? [y/N]: y
Superuser created successfully.

 ~/c/d/Proyecto-Blog 
```

CORREMOS EL SERVIDOR



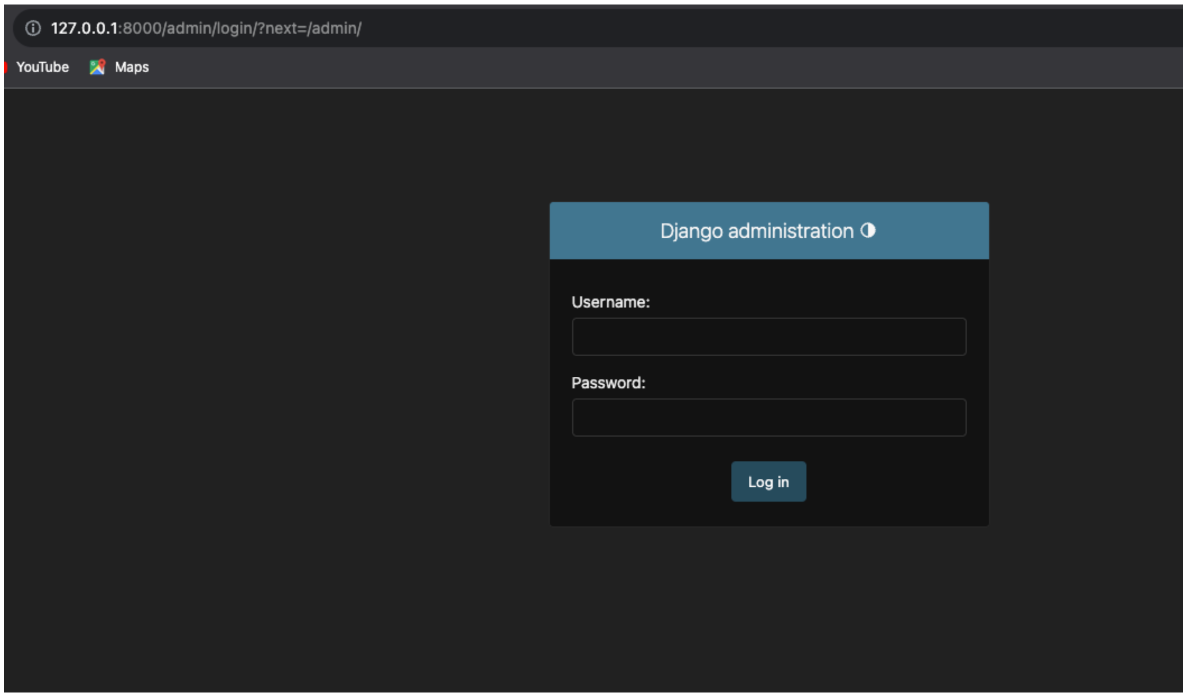
```
 ~/c/d/Proyecto-Blog  python3 manage.py runserver
Watching for file changes with StatReloader
Performing system checks...

System check identified no issues (0 silenced).
September 02, 2023 - 19:45:53
Django version 4.2.4, using settings 'blog.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CONTROL-C.
```

AGREGUEMOS NUESTRO MODELO DE USER EN EL admin.py del post

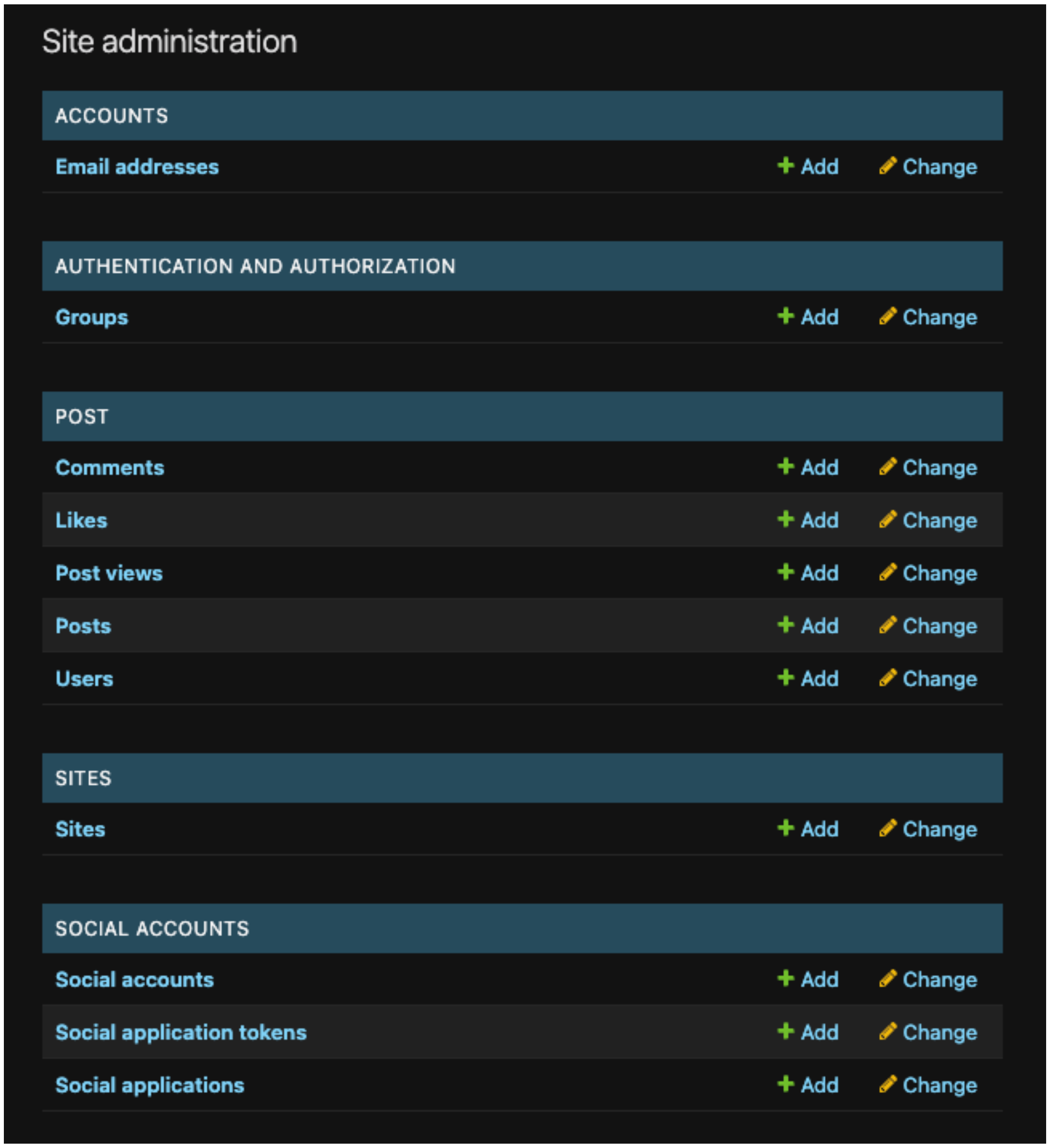


```python
from django.contrib import admin
from .models import Post,PostView, Comment, Like, User

# Register your models here.
admin.site.register(Post)
admin.site.register(PostView)
admin.site.register(Comment)
admin.site.register(Like)
admin.site.register(User)
```

Vamos al sitio en http://127.0.0.1:8000/admin



Ingresemos con nuestros datos del superusuario, si observas se tienen todos los modelos definidos y se pueden agregar desde el administrador.



Site administration

ACCOUNTS

Email addresses	+ Add	Change

AUTHENTICATION AND AUTHORIZATION

Groups	+ Add	Change

POST

Comments	+ Add	Change
Likes	+ Add	Change
Post views	+ Add	Change
Posts	+ Add	Change
Users	+ Add	Change

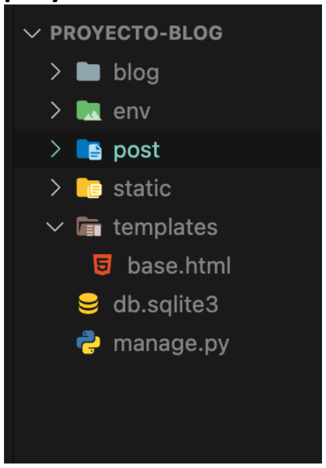
SITES

Sites	+ Add	Change

SOCIAL ACCOUNTS

Social accounts	+ Add	Change
Social application tokens	+ Add	Change
Social applications	+ Add	Change

Nota: tu proyecto en visual code debe de verse en este orden.

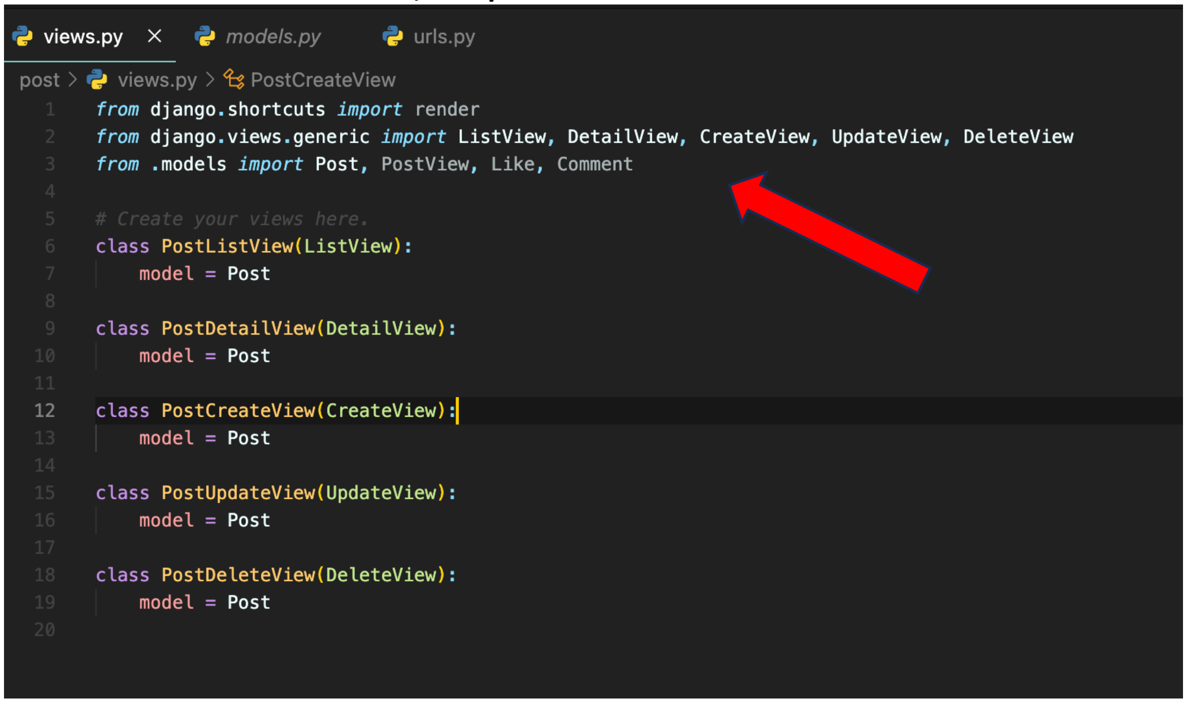


```
PROYECTO-BLOG
  > blog
  > env
  > post
  > static
  v templates
      base.html
    db.sqlite3
    manage.py
```

Si observas hay una carpeta que se llama static, si te dice algun error sobre STATICFILES_DIRS, DEBES DE CREAR ESA CARPETA.

CREANDO VISTAS….

1. **Vamos al archivo views.py de la aplicación post, importamos las vistas genericas y creamos nuestras vistas, e importamos los modelos**

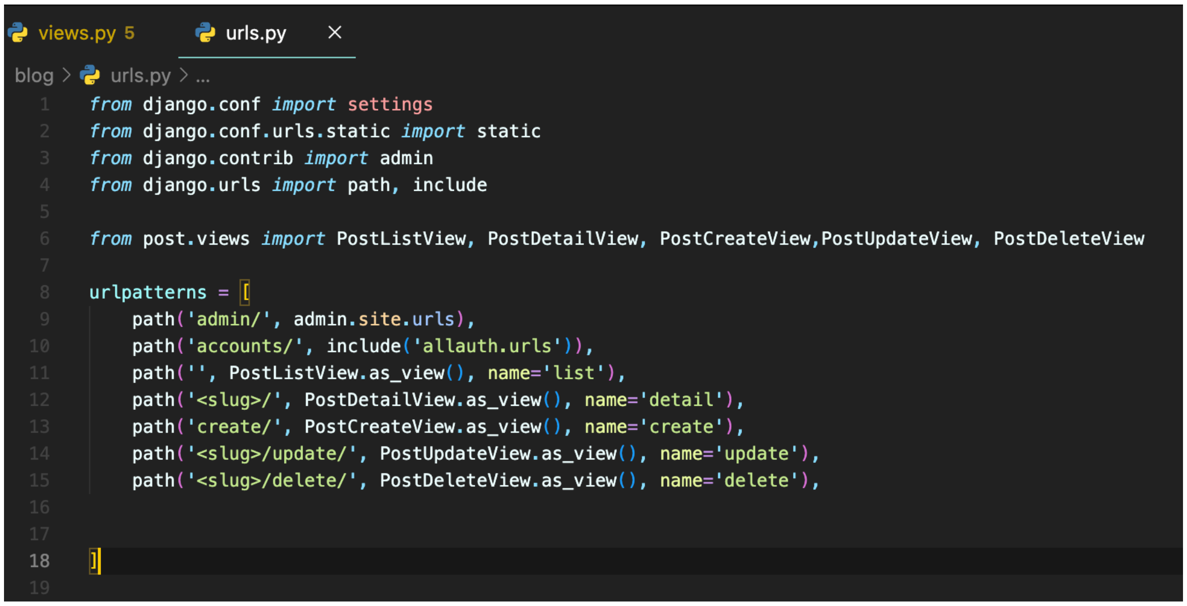


views.py × models.py urls.py

post > views.py > PostCreateView

```python
from django.shortcuts import render
from django.views.generic import ListView, DetailView, CreateView, UpdateView, DeleteView
from .models import Post, PostView, Like, Comment

# Create your views here.
class PostListView(ListView):
    model = Post

class PostDetailView(DetailView):
    model = Post

class PostCreateView(CreateView):
    model = Post

class PostUpdateView(UpdateView):
    model = Post

class PostDeleteView(DeleteView):
    model = Post
```

2. **Vamos a las urls. Py del proyecto.. en blog importamos las vistas, y creamos cada unas de los path para poder llamarlas.**

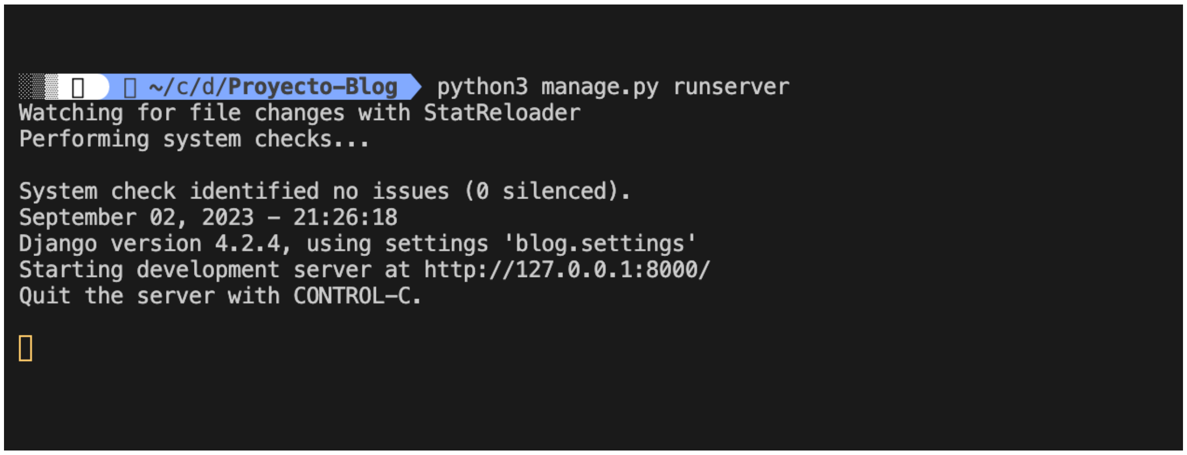


views.py 5 urls.py ×

blog > urls.py > ...

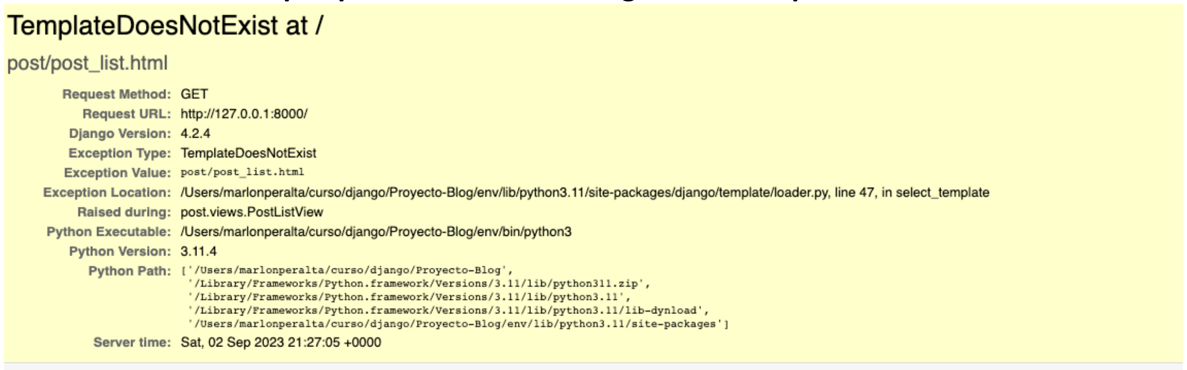
```python
from django.conf import settings
from django.conf.urls.static import static
from django.contrib import admin
from django.urls import path, include

from post.views import PostListView, PostDetailView, PostCreateView,PostUpdateView, PostDeleteView

urlpatterns = [
    path('admin/', admin.site.urls),
    path('accounts/', include('allauth.urls')),
    path('', PostListView.as_view(), name='list'),
    path('<slug>/', PostDetailView.as_view(), name='detail'),
    path('create/', PostCreateView.as_view(), name='create'),
    path('<slug>/update/', PostUpdateView.as_view(), name='update'),
    path('<slug>/delete/', PostDeleteView.as_view(), name='delete'),


]
```

Corramos nuestro servidor



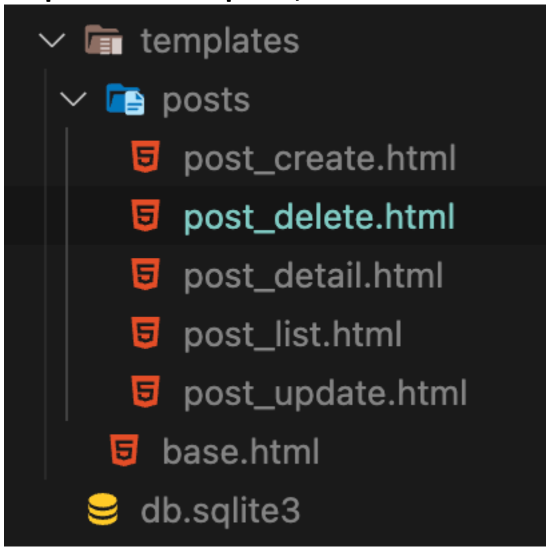
```
~/c/d/Proyecto-Blog  python3 manage.py runserver
Watching for file changes with StatReloader
Performing system checks...

System check identified no issues (0 silenced).
September 02, 2023 - 21:26:18
Django version 4.2.4, using settings 'blog.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CONTROL-C.
```

Carguemos en http://127.0.0.1:8000/
Observemos el error porque aun no hemos cargado los templates.



```
TemplateDoesNotExist at /

post/post_list.html

     Request Method:  GET
        Request URL:  http://127.0.0.1:8000/
     Django Version:  4.2.4
     Exception Type:  TemplateDoesNotExist
    Exception Value:  post/post_list.html
 Exception Location:  /Users/marlonperalta/curso/django/Proyecto-Blog/env/lib/python3.11/site-packages/django/template/loader.py, line 47, in select_template
      Raised during:  post.views.PostListView
  Python Executable:  /Users/marlonperalta/curso/django/Proyecto-Blog/env/bin/python3
     Python Version:  3.11.4
        Python Path:  ['/Users/marlonperalta/curso/django/Proyecto-Blog',
                       '/Library/Frameworks/Python.framework/Versions/3.11/lib/python311.zip',
                       '/Library/Frameworks/Python.framework/Versions/3.11/lib/python3.11',
                       '/Library/Frameworks/Python.framework/Versions/3.11/lib/python3.11/lib-dynload',
                       '/Users/marlonperalta/curso/django/Proyecto-Blog/env/lib/python3.11/site-packages']
        Server time:  Sat, 02 Sep 2023 21:27:05 +0000
```

Por lo tanto debemos de crear nuestros templates, en la carpeta template creemos una carpeta llamada posts, dentro de ella iran los archivos html de cada template.



```
templates
  posts
    post_create.html
    post_delete.html
    post_detail.html
    post_list.html
    post_update.html
  base.html
db.sqlite3
```