Evaluación 2:

Desarrollo de aplicaciones Web y Mobile

**Profesor:**

* Claudio Fuenzalida Medina

**Alumnos:**

* Diego Jorquera Alarcón
* Conny Sanzana Ogalde

**Sección:**

* 002D

**Introducción**

A continuación, se podrá visualizar el desarrollo de la evaluación número dos del ramo Desarrollo de Aplicaciones Web y Mobile, el que consistirá en las capturas de pantalla de las fases contempladas en el desarrollo de la misma actividad, para la que utilizaremos el Framework Django en conjunto con Python además de GitHub (plataforma de desarrollo colaborativo de software que aloja proyectos públicos y privados).

Además de ello el proyecto contemplo el estudio y análisis del Framework Django y de los repositorios GIT, de los cuales utilizamos GitHub, la que si bien, tuvimos dudas respecto a su finalidad, pudimos utilizar como repositorio para almacenar nuestro informe.

Con todo esto damos paso al desarrollo de la evaluación, que se centrara principalmente en el Framework Django y en Python.

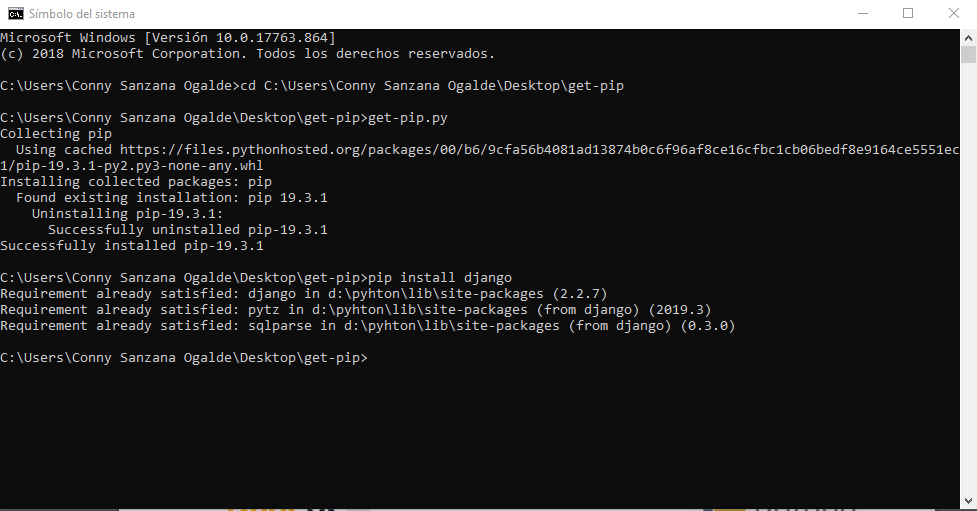
Acceso al GIT: <https://github.com/Connyni/proyecto2.0>

**Desarrollo**

A continuación, se visualizará el paso a paso de la guía de Django:

**Instalación:**

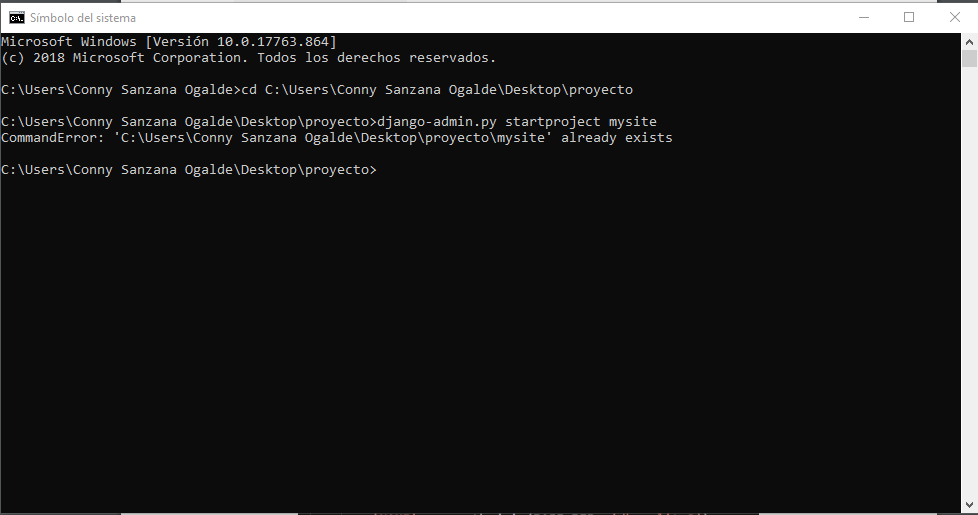
1. Paso 1 y 2; en esta etapa se instalo correctamente el paquete de Python, para ello se descargó y ejecuto desde consola el archivo get-pip.py. Para instalar Django se tuvo que digitar “pip install Django”. Ello dio como resultado la imagen que se mostrara a continuación;

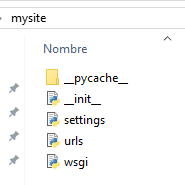


1. Dicha imagen muestra claramente que los packages fueron instalados y que la instalación fue un éxito.

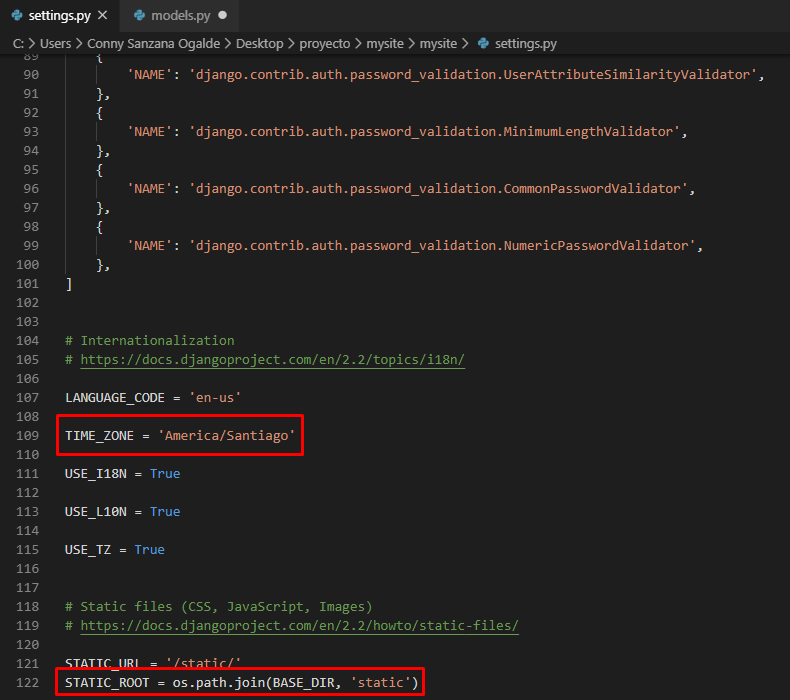
**Primer proyecto con Django**

1. En este paso se dará la instrucción que creara la estructura del proyecto, para ello creamos la carpeta “proyecto” y luego dentro de ella por comando se ejecuta django-admin.py. Luego podemos ver que en la carpeta “proyecto” se crearon satisfactoriamente los documentos solicitados (settings.py,irls.py,wsgi.py,\_\_init\_\_.py).



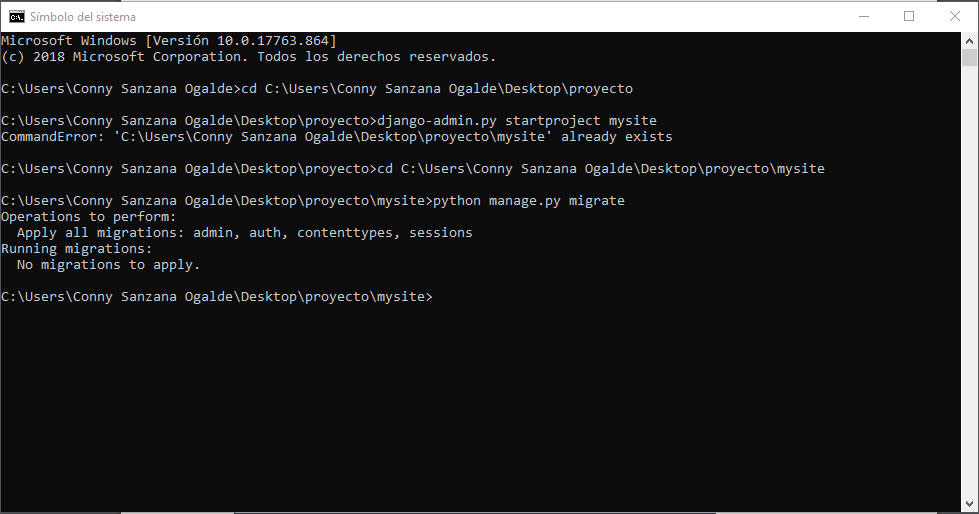


1. Luego de ello la guía pide el cambio del uso horario, para lo que debemos ingresar al archivo setting.py, para esto utilizamos Visual Code, modificando satisfactoriamente el horario, cambiando TIME\_ZONE =’America/Santiago’ y agregando en la última línea STATIC\_ROOT=os.path.join(BASE\_DIR,’static’)

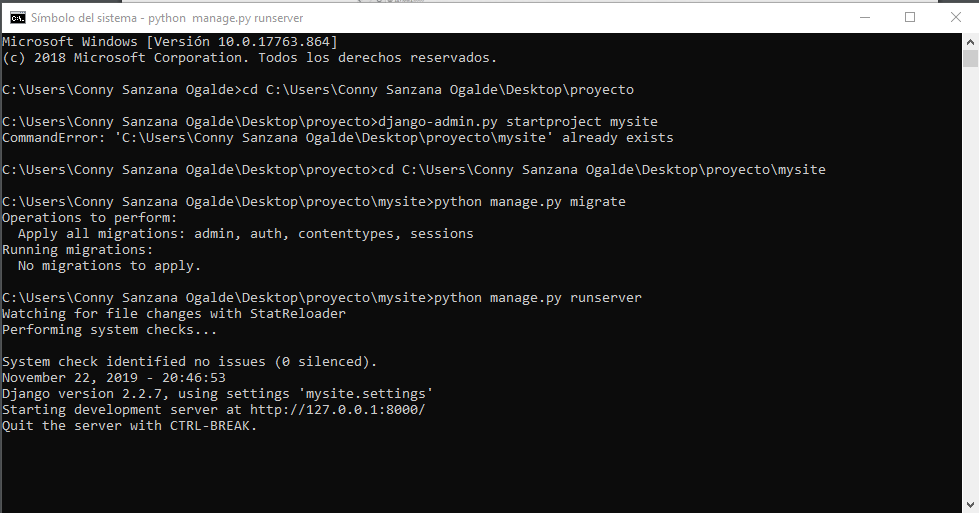


**Configurar una Base de Datos**

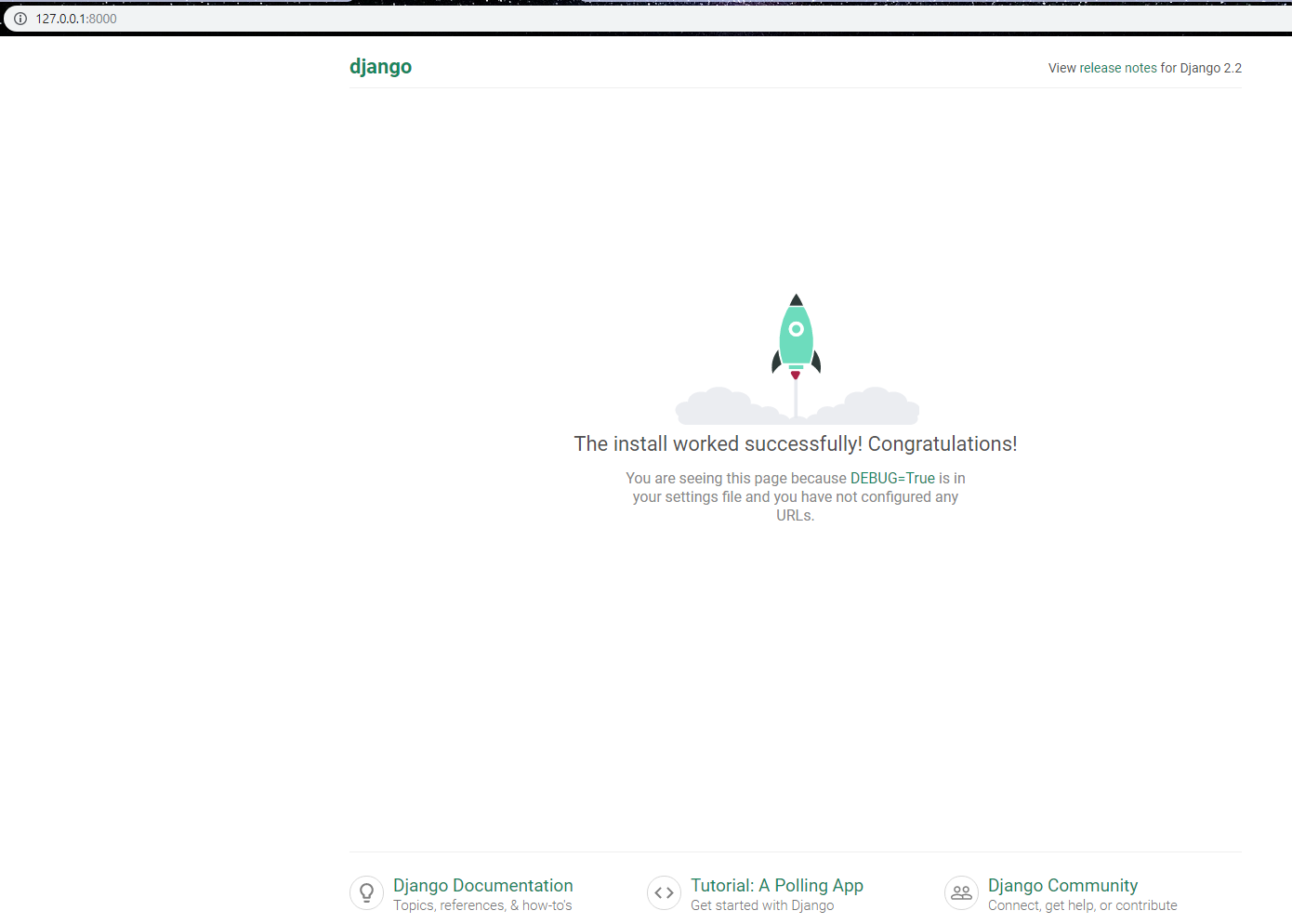
1. Podemos ver dónde está configurada por defecto la DATABASES, por defecto en sqlite3
2. A continuación se puede ver como se crea la base de datos, para ello ejecutamos Python manage.py migrate.



1. En este paso se inicia el servidor web, ejecutando Python manage.py runserver

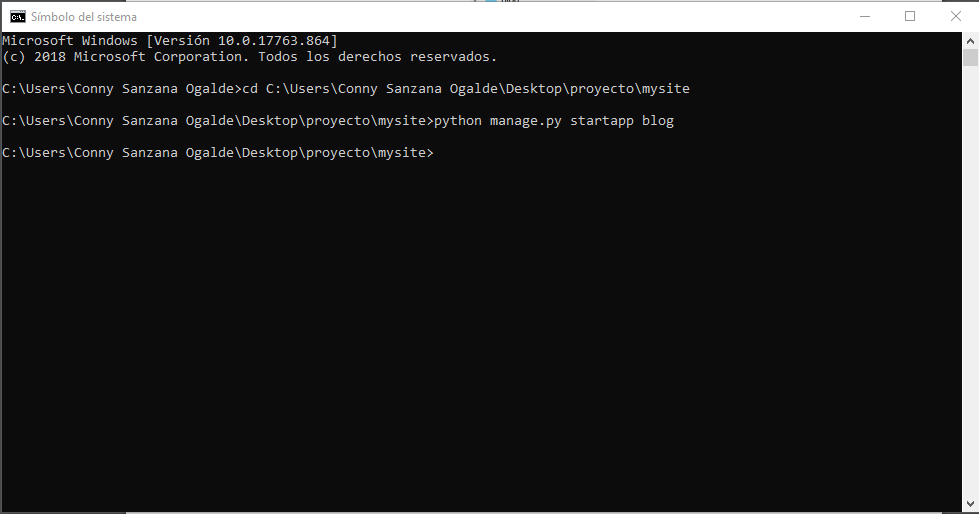


1. Verificación de si el sitio esta corriendo, ingresando a <http://127.0.0.1:8000/>

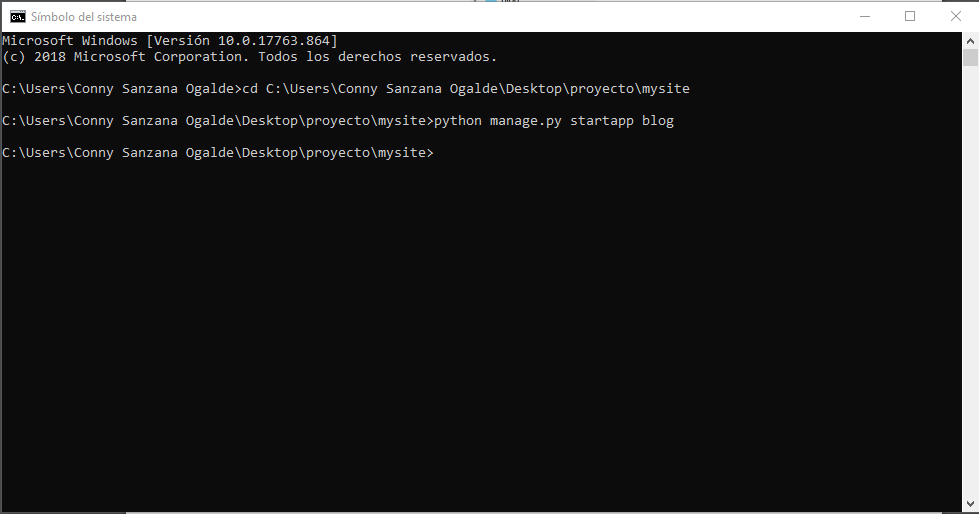


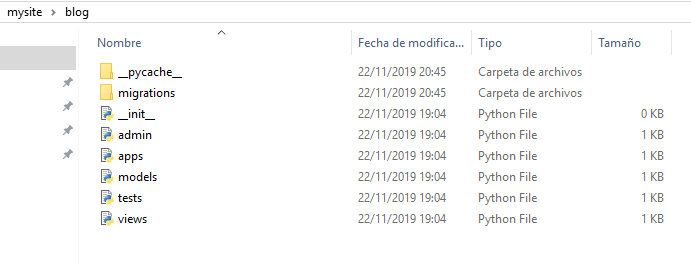
**Model en Django**

1. Abrir consola

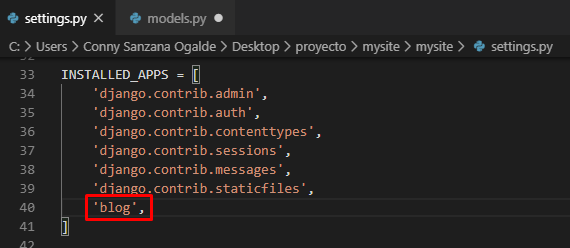


1. Debemos ejecutar en cmd el comando Python manage.py satartapp blog, el que creara la carpeta blog



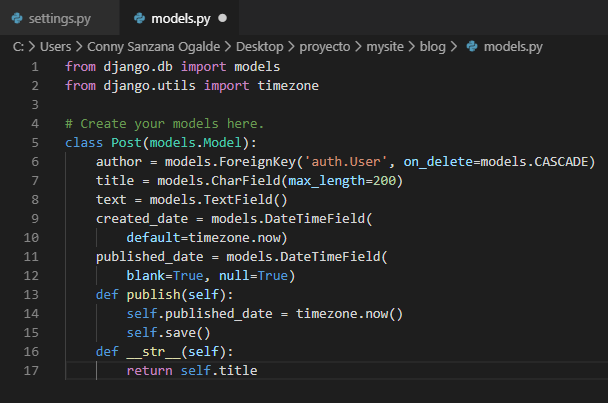


1. A continuación, le decimos a Django como utilizar una aplicación. Ello lo hicimos en el archivo mysite/settings.py, en INSTALLED\_APPS

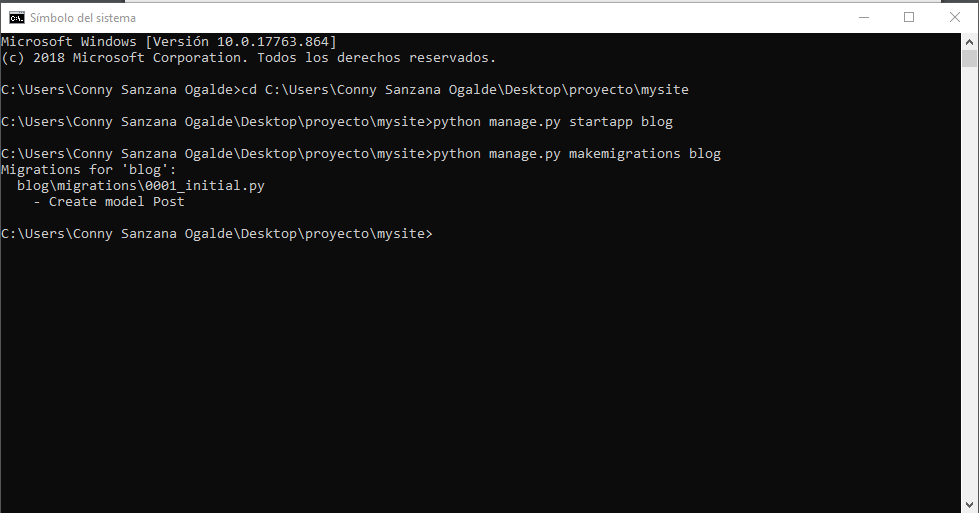


**Creando Modelo Post**

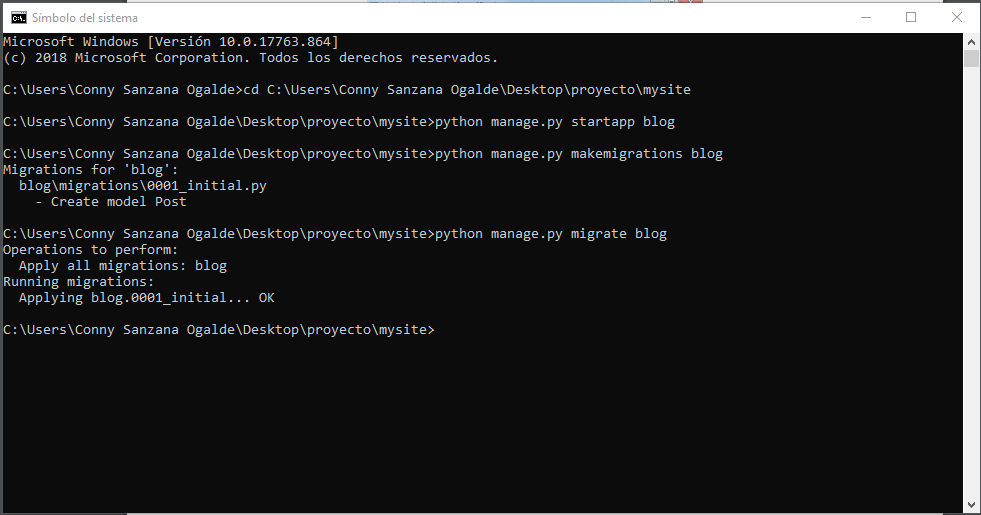
1. Creamos nuestro modelo en el archivo blog/models.py, que nos da como resultado:



1. En esta etapa añadimos nuestro nuevo modelo a nuestra base de datos. Para ello primero ejecutamos Python manage.py makemigrations blog. Así django sabrá que tenemos algunos cambios en nuestro modelo.

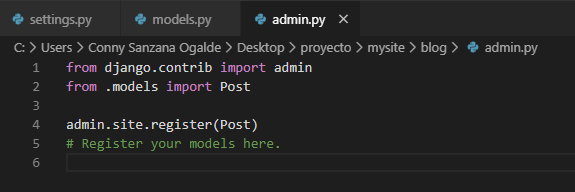


1. Luego ejecutamos Python manage.py migrate blog, con lo que Django preparara un archivo de migración que aplicara a nuestra nueva base de datos.

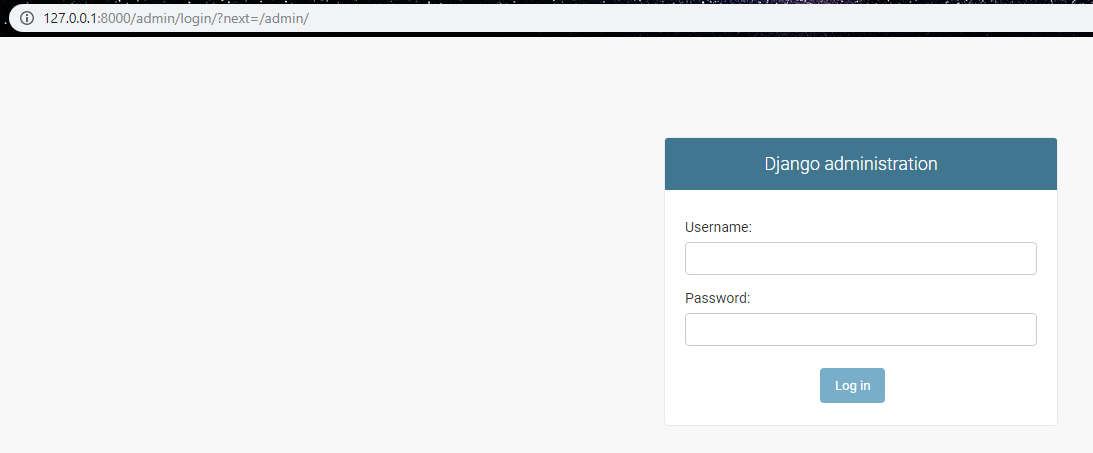


**Administrador de Django**

1. En este paso utilizamos el administrador de Django con el que se puede editar o borrar post modelados, para ello abrimos blog/admin y ejecutamos los codigos que se pueden ver en la imagen.

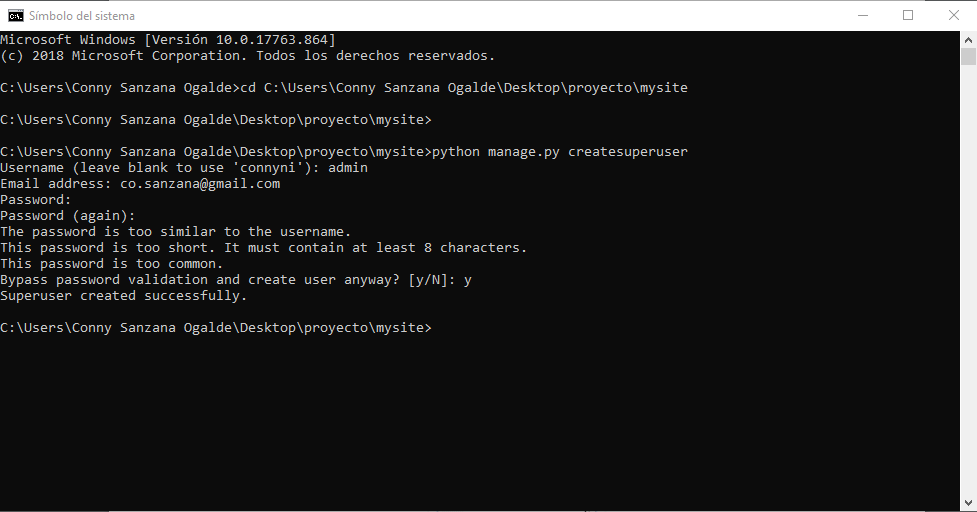


1. En la etapa actual veremos el modelo, para esto se hace correr el servidor con el comando Python manage.py runserver, e ir: http://127.0.0.1:8000/admin/

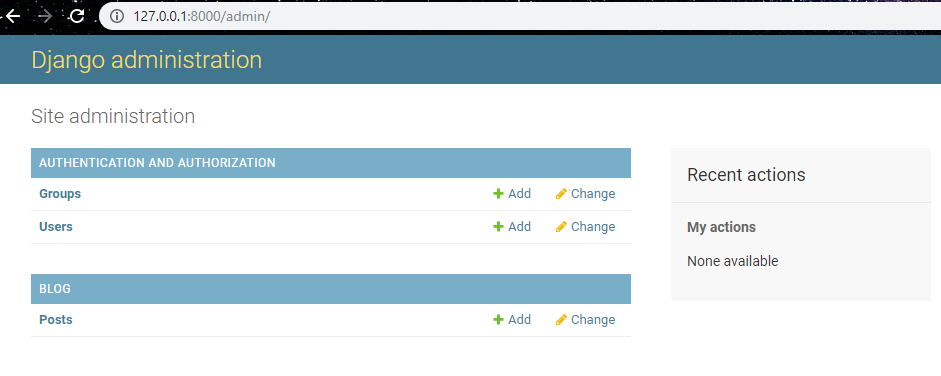


1. A continuación ingresaremos con el superusuario que tiene control en todo el sitio , para ello ejecutamos Python manage.py createsuperuser.

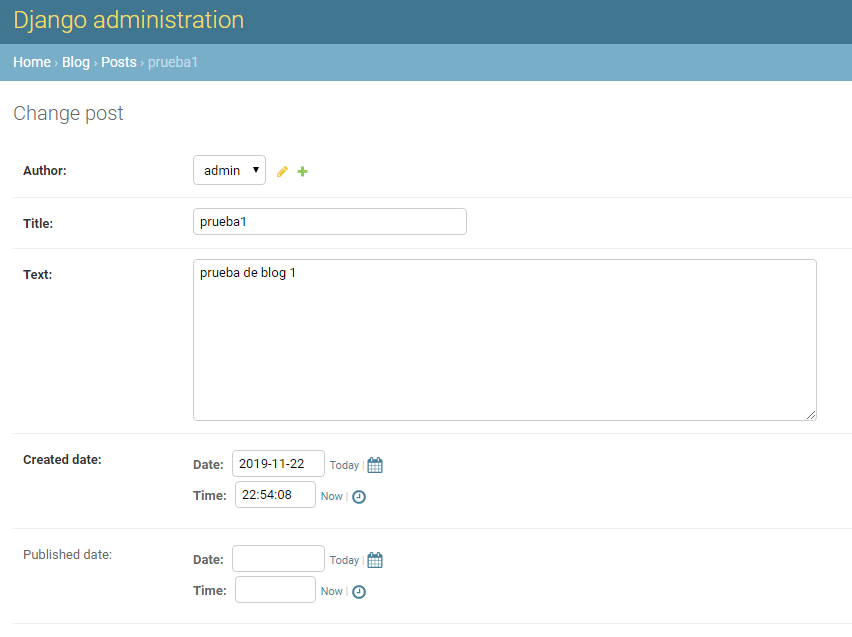
Ingresamos el correspondiente nombre de user(admin), correo(co.sanzana@gmail.com) y pass(admin).

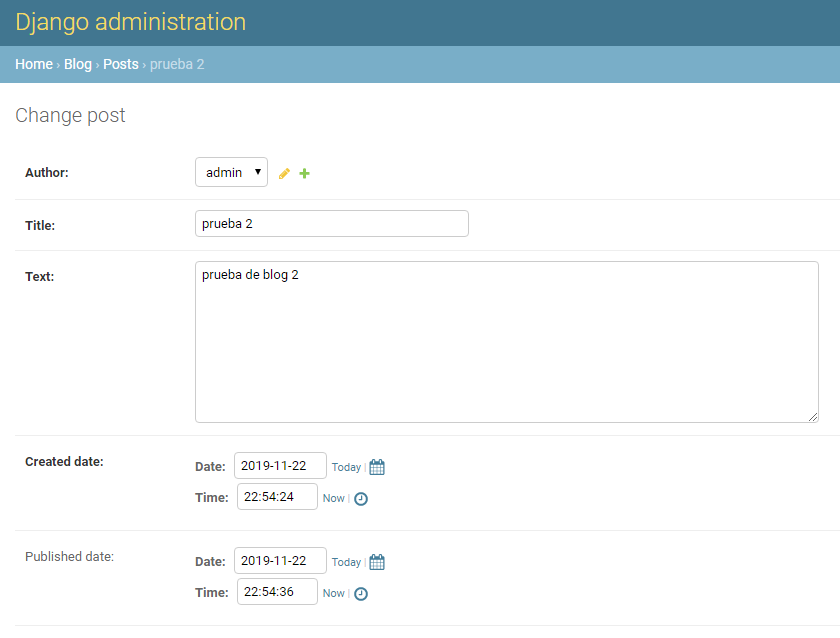


Para volver al navegador ingresamos las credenciales de superusuario elegidas.

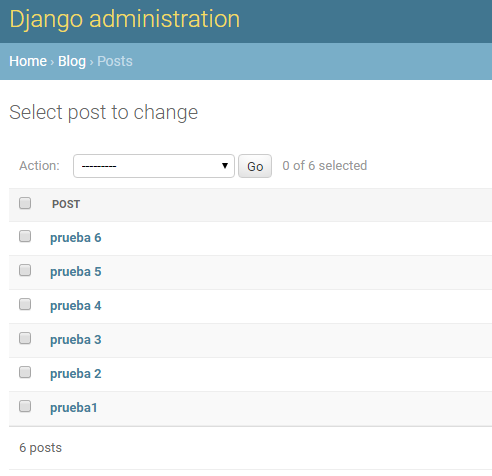


1. Experimentación de Post, agregando cinco o seis post al blog.



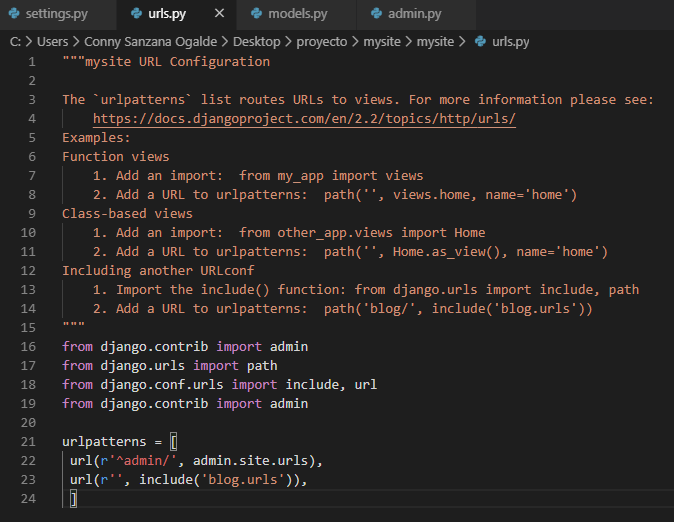


Podemos ver que tienen su respectiva fecha de publicación, por ende se listan.



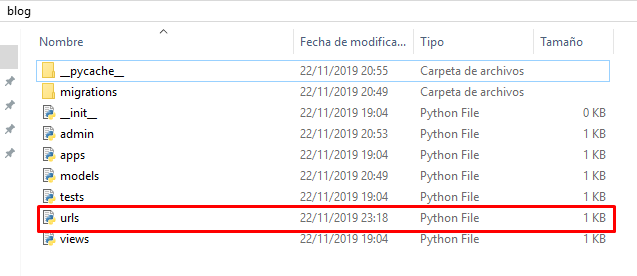
**Django urls**

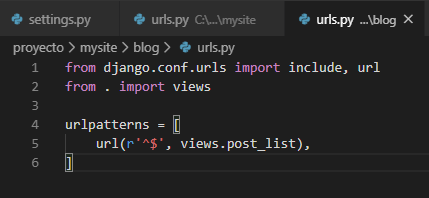
1. En primer lugar, se nos pide abrir el archivo urls.py en mysite, para ello copiamos el código señalado en la guía, lo que nos da como resultado:



En este paso importamos URLs de nuestro blog al archivo misite/urls.py principal. Con ello Django redijera todo lo que va hacia la página, es decir al blog.urls y buscara las instrucciones allí.

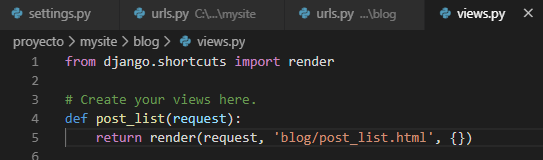
1. Luego debemos crear el archivo urls.py en blog, agregamos el código correspondiente y señalado en la guía. Con ello Django sabrá que views.post\_list es el lugar correcto al que ir si se ingresa al sitio web con la dirección: 'http://127.0.0.1:8000/'.





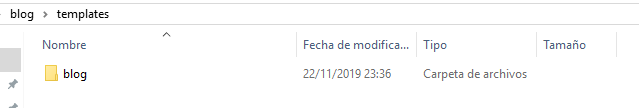
**Vistas de Django**

1. En este paso se muestra la View de la aplicación, solicitándonos información del model creado anteriormente. Colocamos las Vistas en el archivo views.py y agregamos la views al archivo blog/views.py.

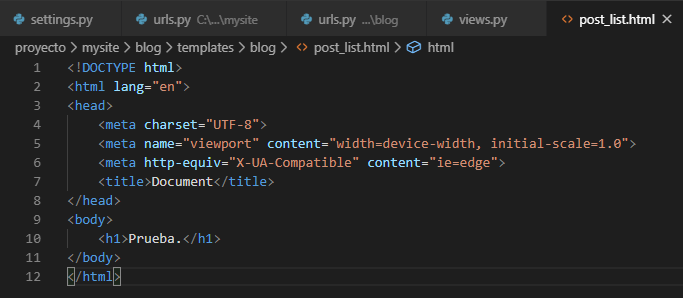


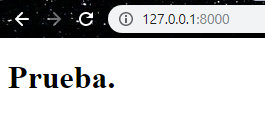
**Crear Plantillas**

1. A continuación, podemos ver que se creó el directorio blog/templates/blog donde se guardan las plantillas. Luego creamos un archivo post\_list.html, en blanco, dentro de templates.

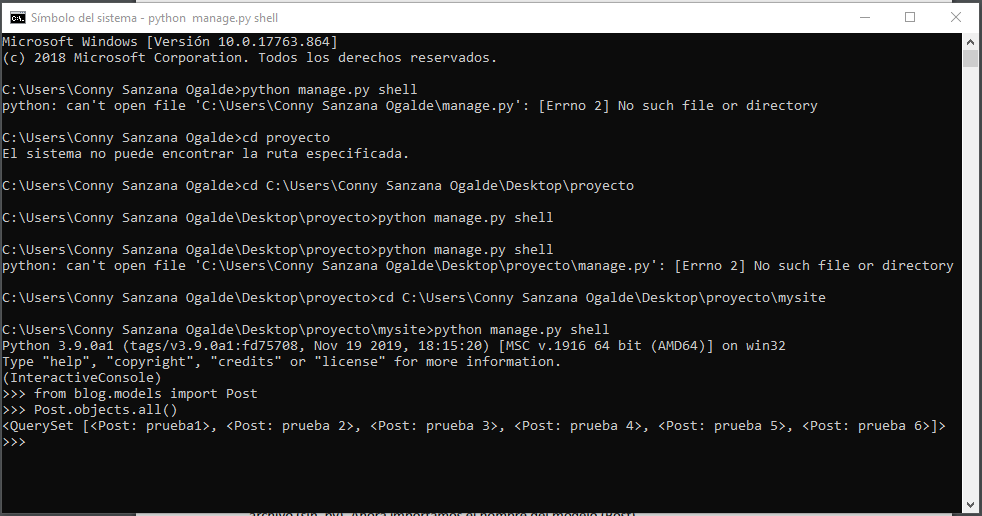


1. En este paso nos piden agregar el código html y ejecutar .



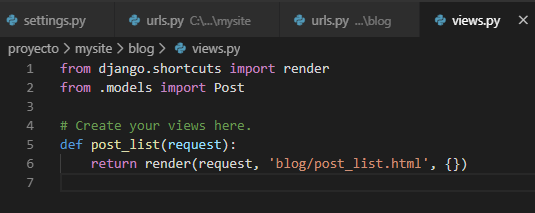


1. En el paso 6, nos muestran los ORM de Django o Querysets.



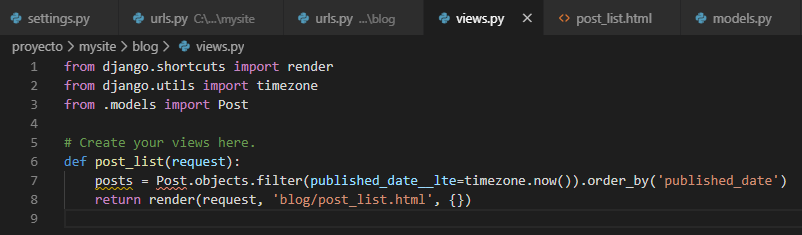
**Datos dinámicos en las plantillas**

1. En esta etapa haremos que nuestros post aparezcan en la plantilla HTML, para ello abrimos nuestro archivo blog/views.py, como se puede visualizar a continuación.

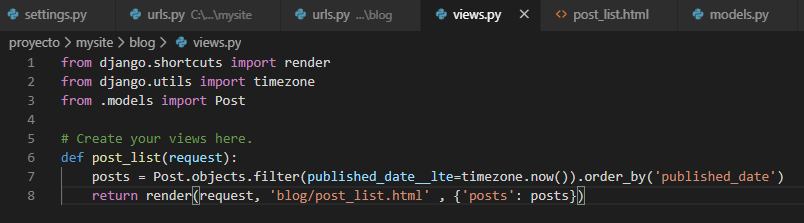


**QuerySet**

1. A continuación, mostraremos el resultado al ordenar por fecha o published\_date.

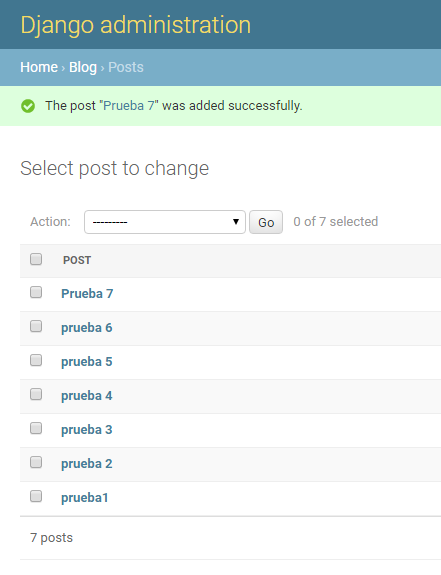


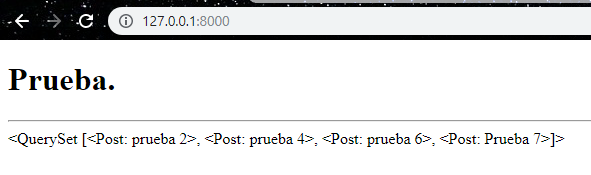
1. Agregamos lo señalado para que se pueda utilizar la plantilla.

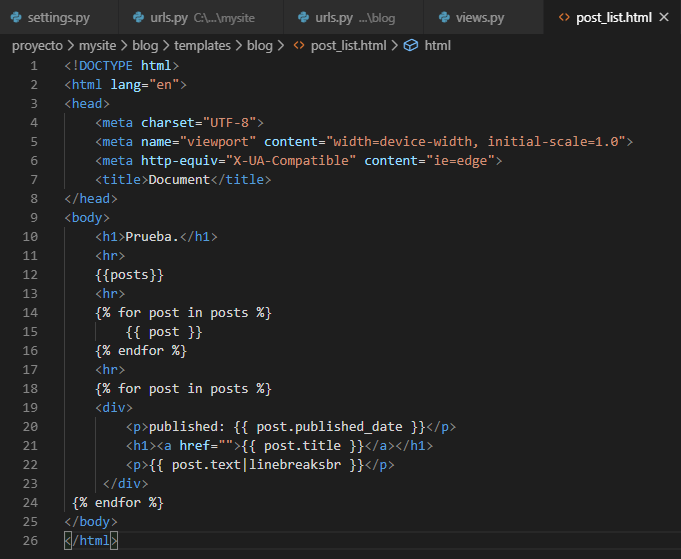


**Plantillas de Django**

1. En este paso imprimiremos una variable en plantilla de Django



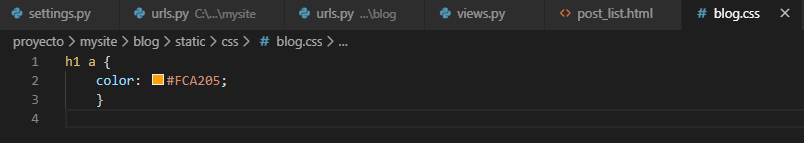




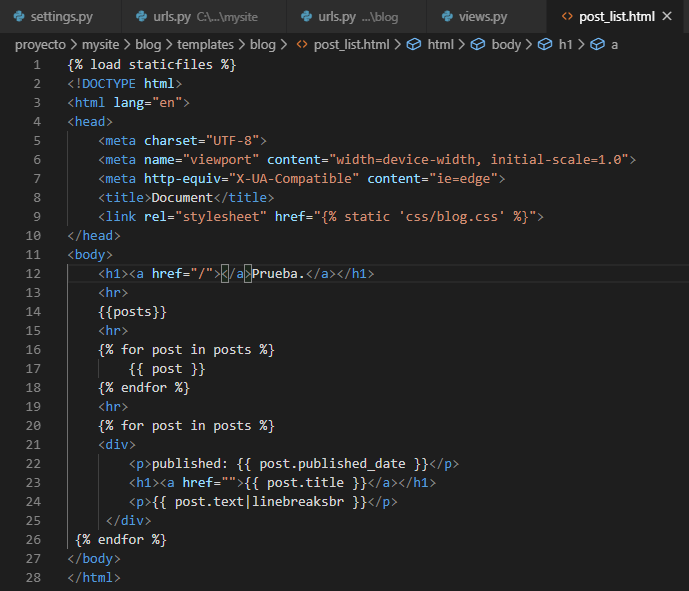


**Crear archivos Estáticos**

1. Creamos archivos estáticos, primero creando la carpeta llamada static dentro de blog. Luego agregamos el archivo css.



Después le decimos a la plantilla HTML que se añade el CSS, para ello se abre el fichero blog/templates/blog/post\_list.html, y se añade al principio; {% load staticfiles %} y también <link rel="stylesheet" href="{% static 'css/blog.css' %}">



**Conclusión**

Al término del proyecto aprendimos a utilizar aspectos básicos del repositorio GitHub, el que permite a los informáticos, trabajar en proyectos variados de forma expedita y limpia. Además de ello al desarrollar el proyecto adquirimos conocimientos sobre Django y su interacción con Python. Lo aprendido varia desde la simple instalación de Django, cambio de horario y configuración de sus bases de datos, ejecución y creación de servidores web y modelos de post, administración, hasta elementos de mayor complejidad como la creación de vistas o views, plantillas, datos dinámicos, QuerySet, plantillas hasta archivos estáticos.

Con todo ello esperamos interiorizar dichos aprendizajes y poder utilizarlos de forma óptima cuando sea requerido, tanto en tareas, como en proyectos a futuro.