

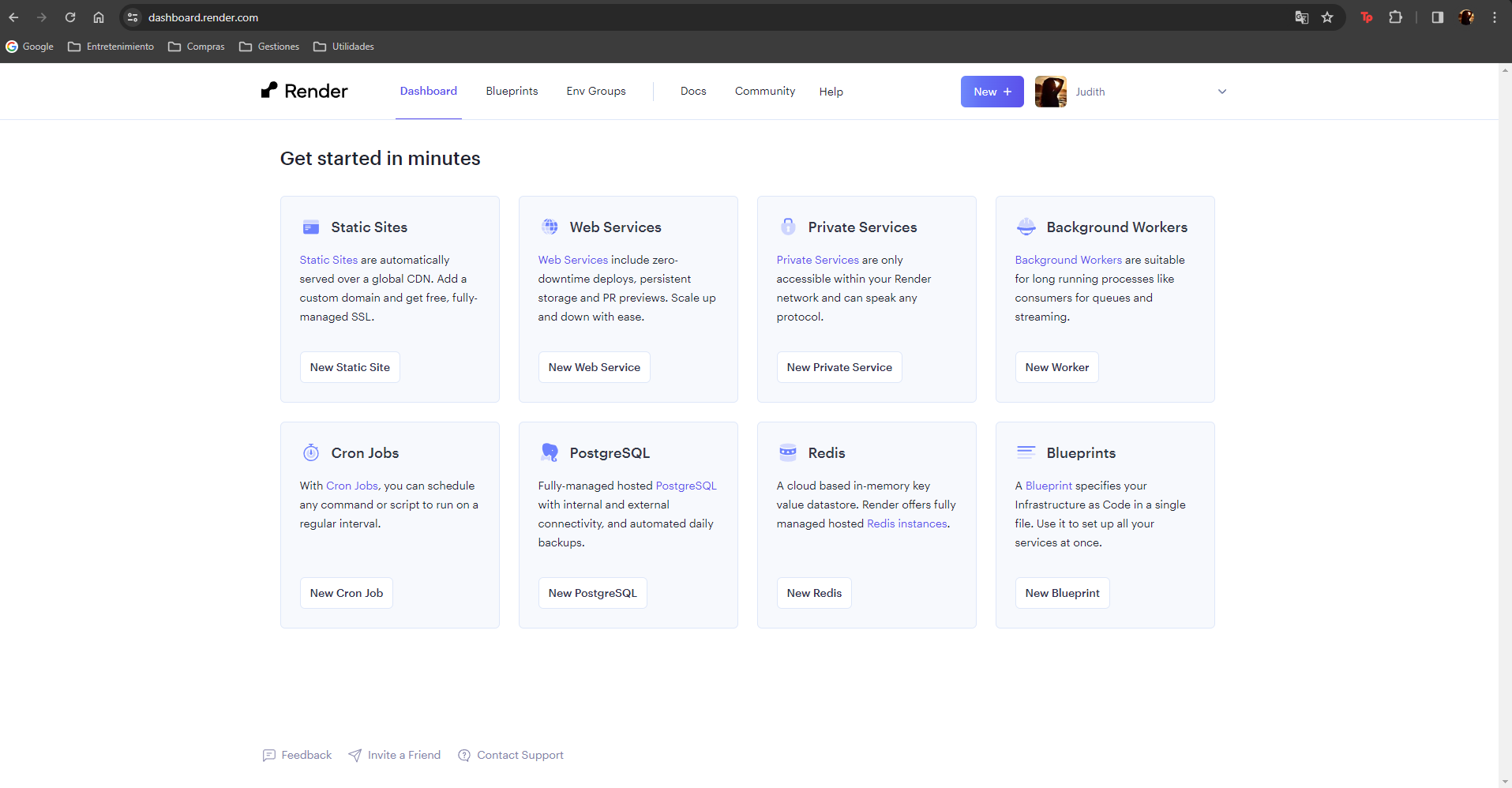
**—**

**PRÁCTICA 1**

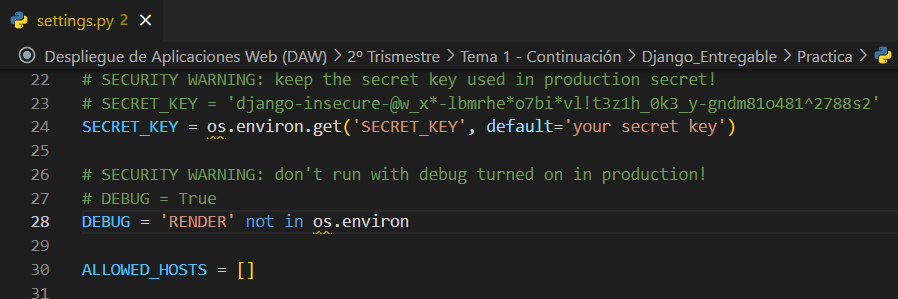
**DESPLIEGUE CON DJANGO Y RENDER**

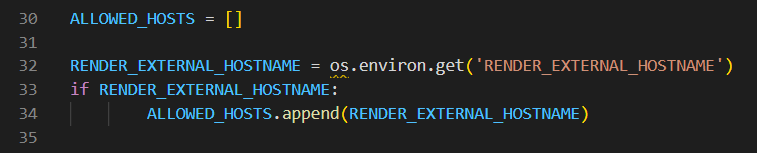
**Paso 1. Vamos a conocer Render.**

Render es una nube unificada para crear y ejecutar todas sus aplicaciones y sitios web con certificados TLS gratuitos, CDN global, redes privadas e implementaciones automáticas desde Git.

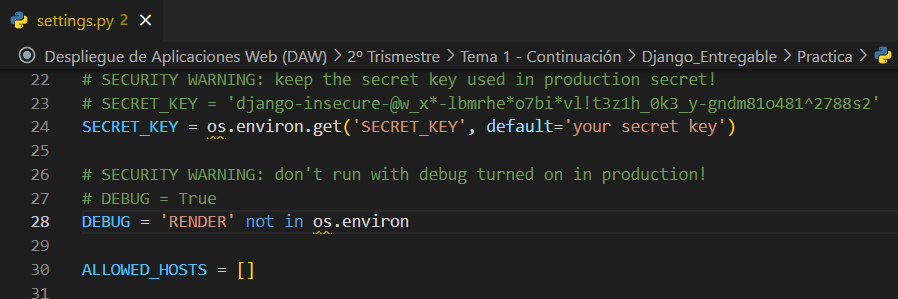
Para hacer uso de esta herramienta, es necesario que creemos una cuenta en [dashboard.render.com](https://dashboard.render.com). En nuestro caso la creamos haciendo uso de nuestra cuenta de GitHub.

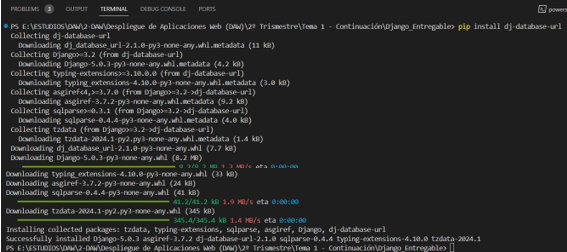
**Paso 2. Modificación de archivos: settings.py.**

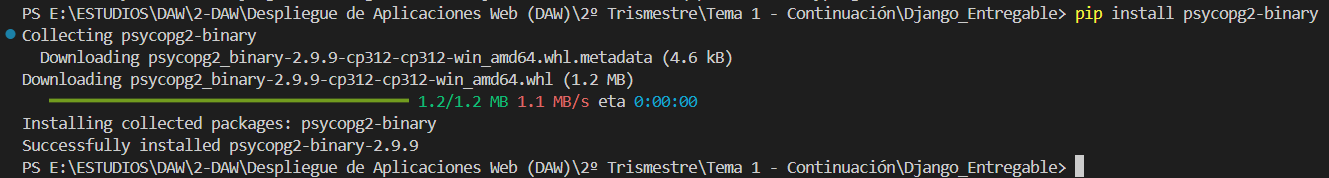
Modificaremos el archivo settings.py asegurando que exista una variable de entorno que mantenga la información protegida. Para ello, importaremos el módulo os. A su vez, modificamos el comando DEBUG para asegurar el correcto funcionamiento del Render.

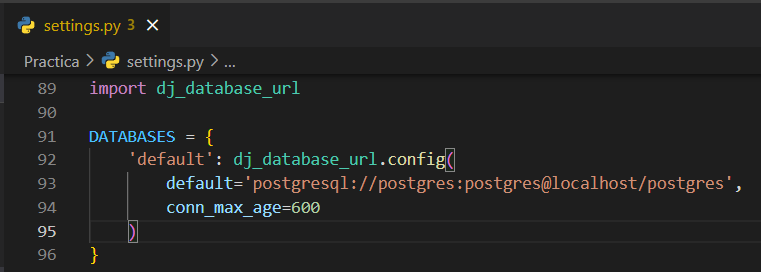
Adicionalmente, añadiremos nuevas instrucciones que garanticen el correcto funcionamiento del sistema.

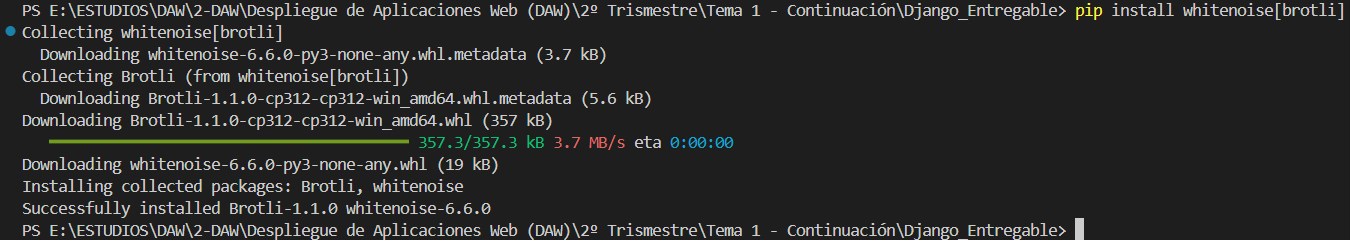
**Paso 2. Configuración de Django**

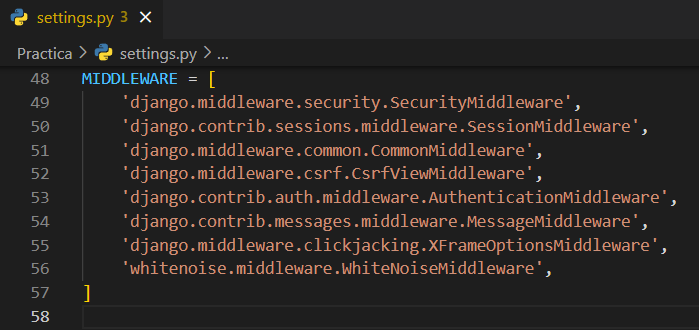
Siempre se debe iniciar activando el entorno mediante el comando .\venv\Scripts\activate. Una vez activado, realizamos la instalación de los paquetes que nos permitirán configurar la BBDD mediante URL de conexión en Django.

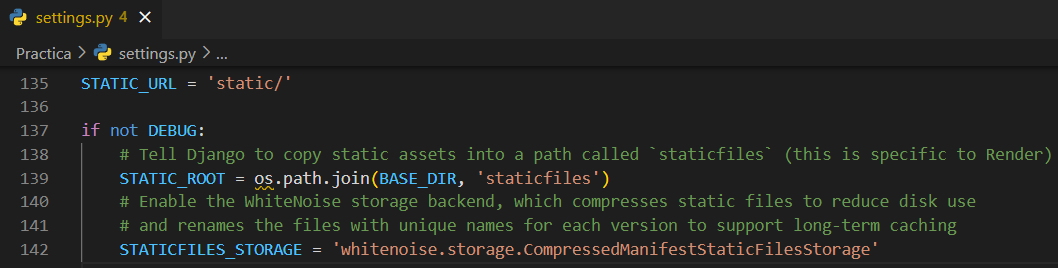


Continuamos instalando una herramienta que permitirá interactuar PostgreSQL en Python.

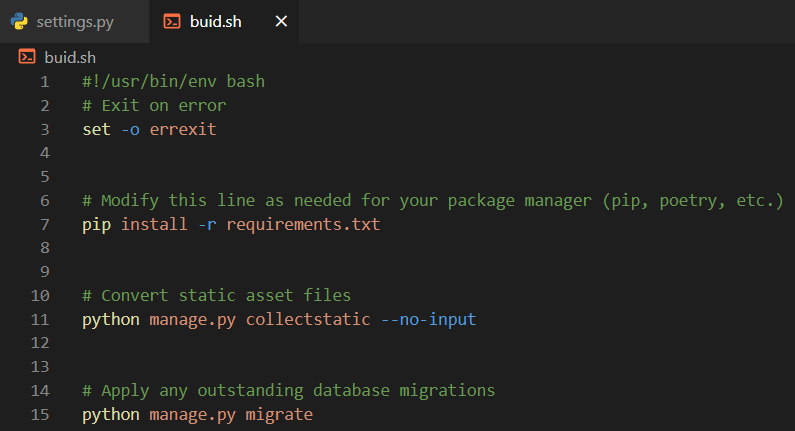
Implementamos en settings.py la herramienta recién instalada.

A continuación, instalamos WhiteNoise, aplicación destinada a dar soporte y mejorar la eficiencia en la gestión de archivos.

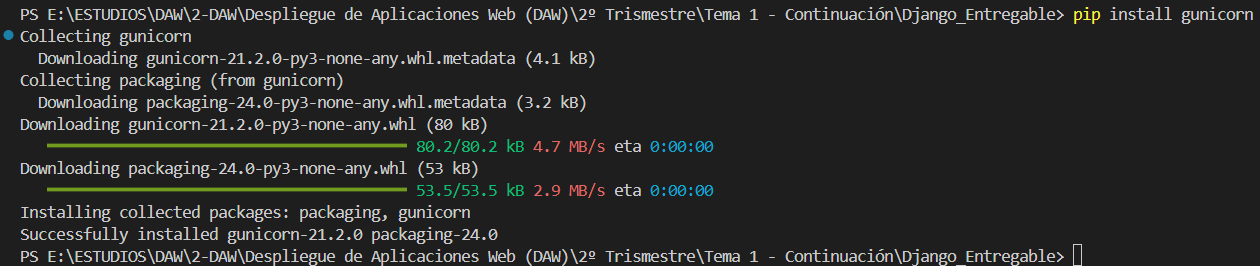
Implementamos la nueva aplicación en el apartado Middleware del fichero settings.py.

Implementamos una condición para asegurar la correcta gestión de los archivos estáticos.

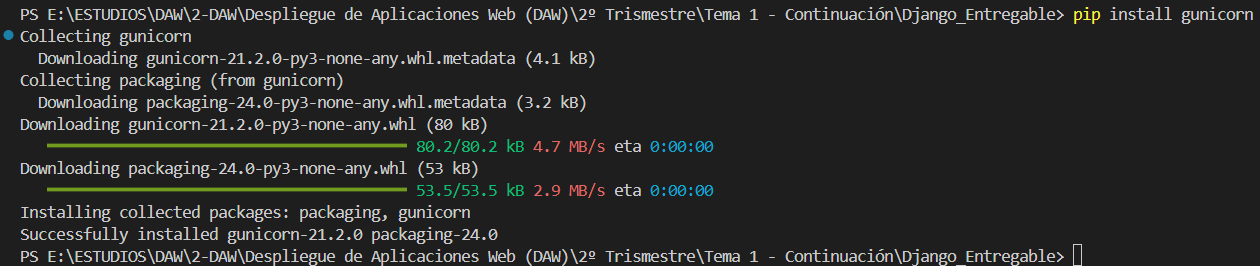
**Paso 3. Configuración de archivo build.sh.**

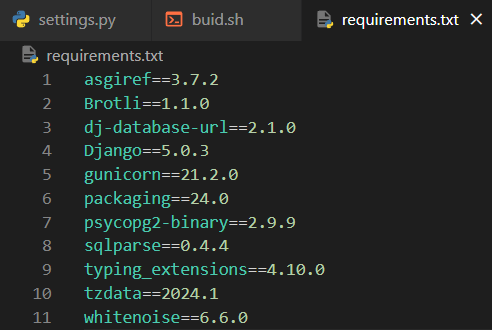
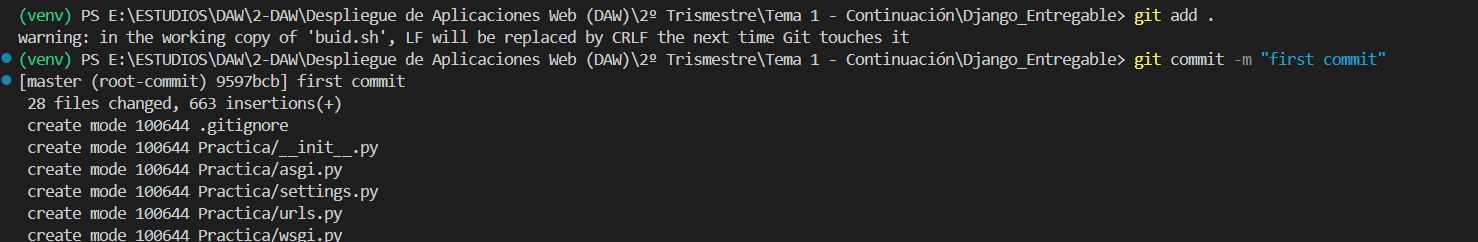
Creamos un archivo build.sh que realizará alguna tarea común asociada al desarrollo de aplicaciones web Django.

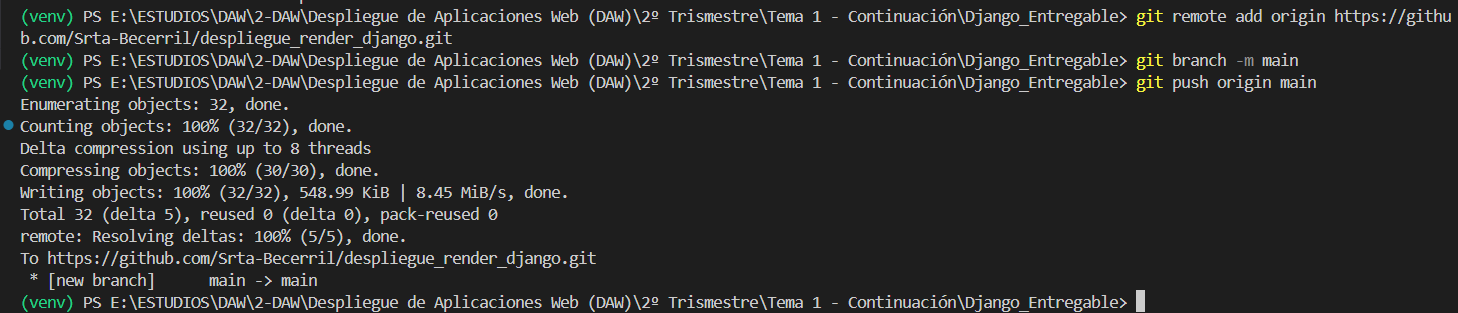
**Paso 4. Instalación de otras herramientas.**

Instalamos Gunicorn. Actuará como servidor HTTP para aplicaciones web Python.

**Paso 5. Creación, configuración y uso de Git.**

Creamos un fichero requierements.txt.

Ahora procederemos a usar Git para subir los archivos a la plataforma.

Creamos un repositorio en GitHub con el que trabajaremos. Subimos todos los archivos.

**Paso 6. Creación, configuración y uso de Render.**

Creamos