Django 5 + venv

Antonio Espin Herranz

venv

- Es conveniente trabajar con un entorno de ejecución distinto para cada proyecto de django.
- Para crear un entorno virtual podemos utilizar la librería venv.
- **python –m venv nombre_entorno**
- Por ejemplo: python –m venv **venv**

Pasos

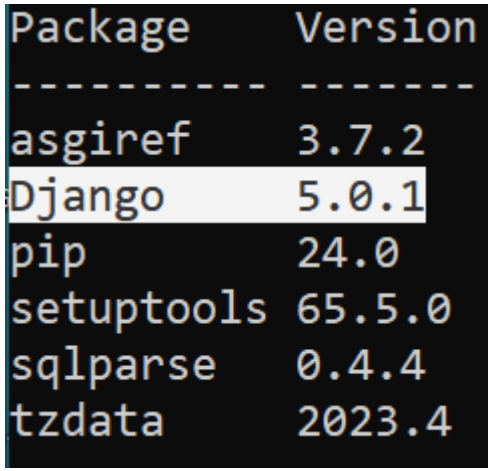
- Crear una carpeta para contener los proyectos
  - md sites
  - cd sites
- Crear una carpeta para el proyecto:
  - md entorno_proyecto_1
  - cd entorno_proyecto_1
- Crear el entorno
  - python –m venv venv
- Activar el entorno:
  - Proyecto>venv\scripts**activate**
- Para desactivar el entorno: deactivate

Se puede crear un entorno
De ejecución por cada proyecto
O uno para todos los proyectos
(Van a tener las mismas librerías)

Para instalar la versión 5 de Django necesitamos Python >= 3.10.
Si tenemos varios Python instalados podemos Lanzar: **\python311\python.exe -m venv venv**

Instalar librerías, crear y arrancar el proyecto

- Con el entorno ya activado:
- **(venv)** … entorno_proyecto_1> pip install django
- Después comprobar: **pip list**



```
Package    Version
---------- -------
asgiref    3.7.2
Django     5.0.1
pip        24.0
setuptools 65.5.0
sqlparse   0.4.4
tzdata     2023.4
```

- Crear el proyecto:
  - **django-admin startproject nombre_proyecto**

- Testear el proyecto:
  - **cd nombre_proyecto**
  - **python manage.py runserver**
  - **http://localhost:8000**

Página de inicio de un proyecto de Django

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.

El servidor se corta
En la consola con
Control+C

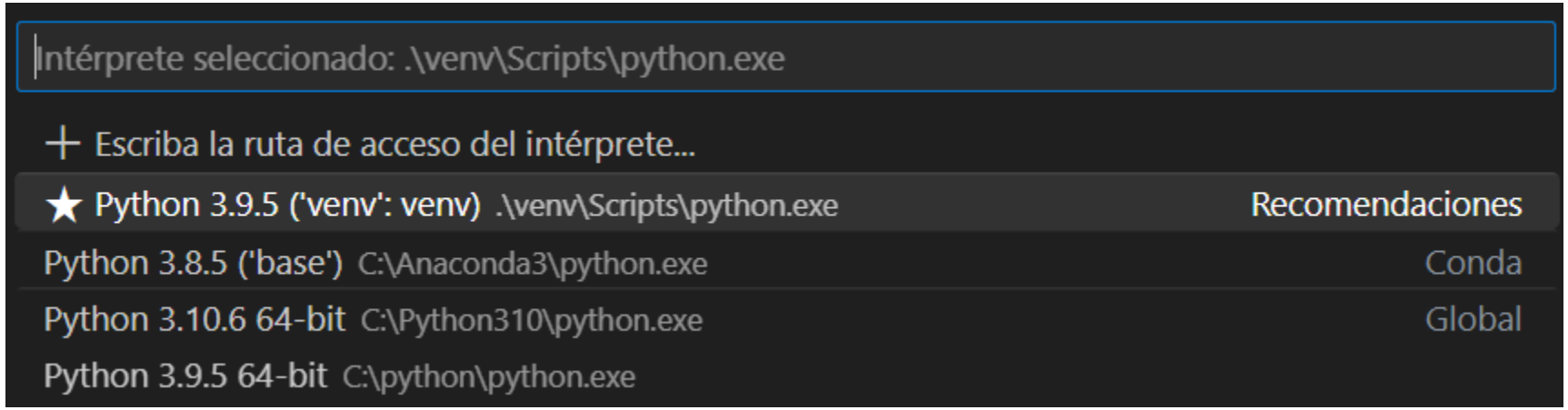
Estructura de los proyectos

- sites
  - Entorno_proyecto_1
    - proyecto1
    - Venv
  - Entorno_proyecto_2
    - proyecto2
    - Venv
  - …
  - Entorno_proyecto_N
    - proyectoN
    - Venv

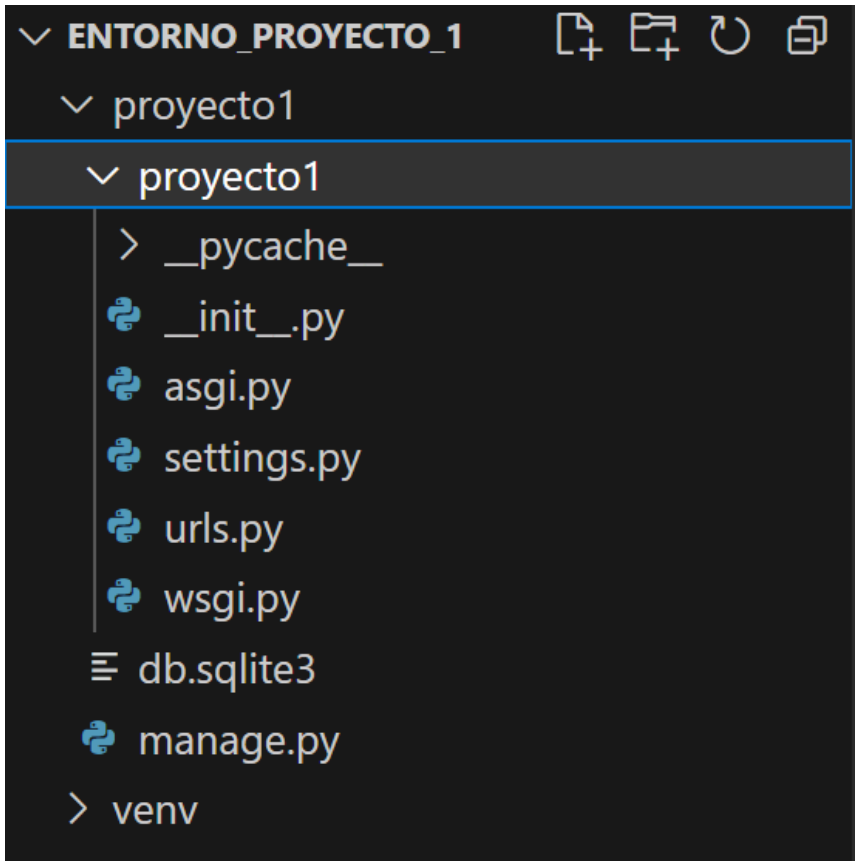
De esta forma podemos tener múltiples Configuraciones de proyectos con distintas versiones De django o de otras que nos hagan falta.

En code

- Una vez descargado e instalado Visual Studio Code.
  - Instalamos la extensión **pylance** para Python
  - Después (con entorno virtual creado y activado) tenemos que seleccionar el intérprete con el que vamos a trabajar.
  - Pulsar las teclas **control+shift+p** y teclear: **select interpreter** (seleccionar nuestro entorno virtual).



En code



- En la carpeta de code vemos el proyecto y venv.
- Al pulsar las teclas: **control+shift+p**
- Nos dará como sugerencia el intérprete de Python del entorno virtual.

- Después chequear el proyecto:
  - Desde la consola ejecutar:
    - **python manage.py runserver**
    - http://localhost:8000

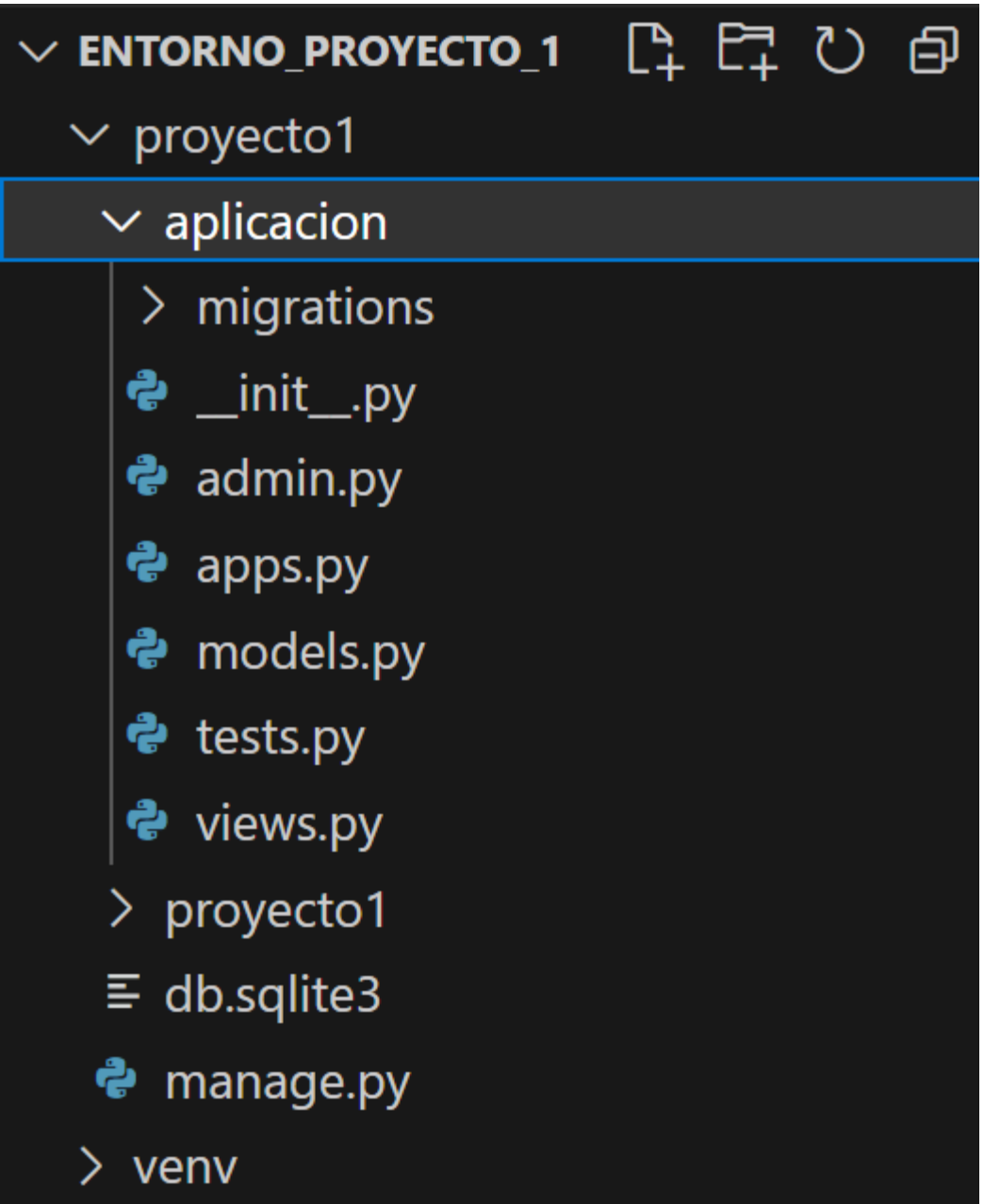
Crear una aplicación

- Después de crear y chequear el proyecto:
- Siempre a la altura de donde se encuentra el fichero: **manage.py**

- **python manage.py startapp nombre_aplicación**

- **La aplicación hay que añadirla al fichero settings.py en la lista de:**
  - `INSTALLED_APPS`

En code



- Cuando creamos la aplicación dentro del proyecto tendremos la siguiente vista del proyecto.

- Se habrá creado una carpeta con el nombre de la aplicación dentro del proyecto.

Estructura de los proyectos (más detallado)

- sites
  - Entorno_proyecto_1
    - Proyecto1
      - Proyecto1 (Ficheros de configuración)
      - Aplicación (Ficheros de la aplicación)
      - Db.sqlite3 (Base de datos)
      - Manage.py (Fichero de comandos de django)
    - Venv (Entorno de ejecución)
  - Entorno_proyecto_2
    - Proyecto2
      - Proyecto2
      - Aplicación
      - Db.sqlite3
      - Manage.py
    - Venv
  - …

A parte podremos tener otras Carpetas donde se colocarán:
- Plantillas
- Contenido estático del sitio
  - Css
  - Imágenes
  - Etc.

Como continuar

- Podemos definir nuestro modelo en objetos y generamos el modelo relacional → próximos temas

- O al revés partimos de una base de datos (modelo relacional) ya creada y queremos generar el modelo de objetos de forma automática.