

## Instalación y Configuración Básica de Django

### 1) ¿Qué es pip y para qué sirve?

pip es el gestor de paquetes de Python. Permite instalar, actualizar y desinstalar paquetes desde el repositorio PyPI. Ejemplos: `pip install django`, `pip install requests`.

### 2) Crear entorno virtual y preparar el sistema

Usa siempre un entorno virtual para evitar conflictos entre proyectos.

Linux / macOS

```
python3 -m venv venv
source venv/bin/actívale
```

Windows (PowerShell)

```
python -m venv venv
.\venv\Scripts\Activate.ps1
```

Windows (CMD)

```
python -m venv venv
venv\Scripts\activate.bat
```

Actualizar pip (opcional pero recomendable):

```
python -m pip install --upgrade pip
```

Si pip no se encuentra, usa `python -m pip ....`

---

### 3) Instalar Django

Con el entorno activado:

```
pip install django
```

Comprobar instalación / versión:

```
django-admin --version
```

# o

```
python -m django --version
```

Salida esperada: número de versión (por ejemplo 4.2.3 o similar).

---

### 4) Crear un proyecto Django

Crear proyecto (ejemplo nombre `mi_proyecto`):

```
django-admin startproject mi_proyecto
```

```
cd mi_proyecto
```

Crear una app ejemplo (opcional, pero útil para explorar):

```
python manage.py startapp inventario
```

---

## 5) Estructura del proyecto (árbol típico)

```
mi_proyecto/  
├─ manage.py  
├─ mi_proyecto/  
│   ├── __init__.py  
│   ├── settings.py  
│   ├── urls.py  
│   └─ asgi.py  
│   └─ wsgi.py  
├─ inventario/  
│   ├── migrations/  
│   ├── __init__.py  
│   ├── models.py  
│   ├── views.py  
│   ├── admin.py  
│   └─ apps.py  
└─ venv/ # tu entorno virtual (fuera del repo idealmente)
```

---

## 6) Archivos clave y sus funciones

- **manage.py**  
Script para administrar el proyecto: migraciones, servidor, crear superusuario, etc. Ej.:  
python manage.py runserver, python manage.py migrate.
  - **settings.py**  
Configuración del proyecto: INSTALLED\_APPS, DATABASES, TEMPLATES, ALLOWED\_HOSTS, STATICFILES, etc.
  - **urls.py** (en la carpeta del proyecto)  
Define rutas de URL globales y enlaces a include() de apps.
  - **init.py**  
Marca el directorio como paquete Python.
- 

## 7) Migraciones iniciales y comprobar DB (SQLite por defecto)

Aplica migraciones iniciales:

```
python manage.py migrate
```

Crea un superusuario:

```
python manage.py createsuperuser
```

# te pedirá username, email y contraseña

---

## 8) Levantar servidor de desarrollo

```
python manage.py runserver
```

Abre en el navegador: <http://127.0.0.1:8000/> → deberías ver la página Django welcome.

Panel admin: <http://127.0.0.1:8000/admin/> → inicia sesión con el superusuario creado.

Para detener el servidor: Ctrl+C.

---

## 9) Configuración mínima de TEMPLATES y urls.py

### a) Ajustar settings.py para templates

En settings.py modifica/asegura la sección TEMPLATES para incluir una carpeta templates en la raíz del proyecto:

```
import os
```

```
BASE_DIR = os.path.dirname(os.path.dirname(os.path.abspath(__file__)))
```

```
TEMPLATES = [  
    {  
        "BACKEND": "django.template.backends.django.DjangoTemplates",  
        "DIRS": [ os.path.join(BASE_DIR, "templates") ], # <-- carpeta global de 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",  
            ],  
        },  
    ],  
]
```

Comprueba también:

```
INSTALLED_APPS = [  
    # apps de Django...  
    'django.contrib.admin',  
    'django.contrib.auth',  
    # ...  
    # tu app  
    'inventario',  
]
```

### b) urls.py del proyecto (mi\_proyecto/urls.py)

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

```
urlpatterns = [  
    path('admin/', admin.site.urls),  
    path("", include('inventario.urls')), # delega rutas a la app inventario  
]
```

### c) inventario/urls.py (crear este archivo)

```
from django.urls import path  
from . import views
```

```
urlpatterns = [  
    path("", views.index, name='index'),  
]
```

#### **d) inventario/views.py ejemplo simple**

```
from django.shortcuts import render
```

```
def index(request):
```

```
    return render(request, 'index.html', {'mensaje': 'Hola desde Django!'})
```

#### **e) Crear template templates/index.html**

Crea carpeta templates/ en la raíz del proyecto y archivo index.html:

```
<!-- templates/index.html -->  
<!doctype html>  
<html>  
  <head><meta charset="utf-8"><title>Inicio</title></head>  
  <body>  
    <h1>{{ mensaje }}</h1>  
  </body>  
</html>
```

Ahora al ir a <http://127.0.0.1:8000/> verás tu página.

---

### **10) Manejo de archivos estáticos (opcional básico)**

En settings.py:

```
STATIC_URL = '/static/'
```

Para desarrollo, django.contrib.staticfiles sirve estáticos automáticamente si está en INSTALLED\_APPS.

---

### **11) Comandos útiles resumen**

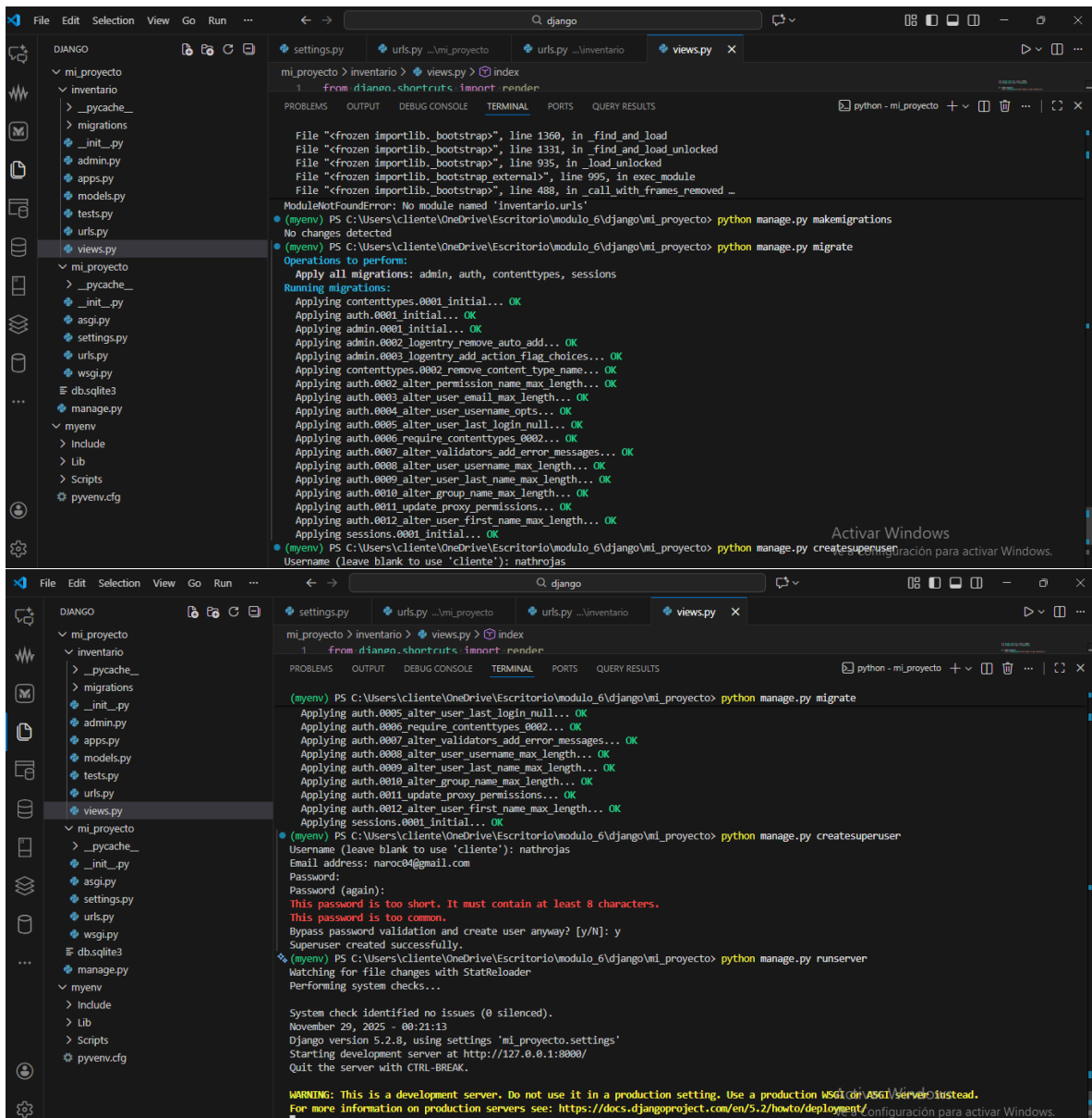
- Crear venv:
  - `python -m venv venv`
  - `source venv/bin/activate`
  - Instalar Django:
  - `pip install django`
  - Crear proyecto:
  - `django-admin startproject mi_proyecto`
  - Crear app:
  - `python manage.py startapp inventario`
  - Migrar:
  - `python manage.py migrate`
  - Crear superusuario:
  - `python manage.py createsuperuser`
  - Ejecutar servidor:
  - `python manage.py runserver`
-

## Anexos

The first screenshot shows the initial setup of a Django project. The file explorer on the left displays the project structure, including files like `__init__.py`, `asgi.py`, `settings.py`, `urls.py`, `wsgi.py`, `manage.py`, and a `myenv` directory. The terminal window shows the execution of `python -m venv myenv`, `myenv\Scripts\activate`, and `pip install Django`. The output indicates that Django 5.2.8 and its dependencies (asgiref 3.8.1, sqlparse 0.5.4, tzdata 2025.2) are being collected and installed.

The second screenshot shows the next steps in the setup. The file explorer now includes `migrations`, `admin.py`, `apps.py`, `models.py`, `tests.py`, and `views.py`. The terminal window shows the execution of `django-admin startproject mi_proyecto` and `cd mi_proyecto`. It also shows the installation of Django 5.2.8 and its dependencies.

The third screenshot shows the final steps in the setup. The file explorer now includes `db.sqlite3` and `manage.py`. The terminal window shows the execution of `python manage.py startapp inventario`. The output indicates that the app is being created.





The install worked successfully! Congratulations!

View [release notes](#) for Django 5.2

You are seeing this page because `DEBUG=True` is in your settings file and you have not configured any URLs.

**django**

[Django Documentation](#)  
Topics, references, & how-tos

[Tutorial: A Polling App](#)  
Get started with Django

[Django Community](#)  
Connect, get help, or contribute

Activar Windows  
Ve a Configuración para activar Windows.

## Django administration

WELCOME, NATHROJAS. [VIEW SITE](#) / [CHANGE PASSWORD](#) / [LOG OUT](#)

### Site administration

#### AUTHENTICATION AND AUTHORIZATION

Groups	<a href="#">+ Add</a>	<a href="#">Change</a>
Users	<a href="#">+ Add</a>	<a href="#">Change</a>

#### Recent actions

##### My actions

None available

Activar Windows  
Ve a Configuración para activar Windows.

