**Paso 1: Crear el entorno virtual**

En la terminal o en el símbolo del sistema, navega hasta la carpeta de tu proyecto y usa el siguiente comando para crear un entorno virtual (normalmente llamado venv):

python -m venv venv

Aquí, venv es el nombre del entorno virtual. Puedes usar otro nombre si lo prefieres.

**Paso 2: Activar el entorno virtual**

Una vez creado, debes activar el entorno virtual. La forma de hacerlo depende de tu sistema operativo:

**En Windows**

venv\Scripts\activate

**En macOS y Linux**

source venv/bin/activate

**Paso 3: Verificar que el entorno virtual está activo**

Después de activarlo, verás que el nombre del entorno virtual aparece al principio de la línea de la terminal. Por ejemplo, puede verse así:

(venv) $

Esto indica que el entorno virtual está activo, y cualquier paquete que instales (por ejemplo, usando pip install) se instalará en este entorno, en lugar de globalmente.

**Paso 4: Desactivar el entorno virtual**

Cuando termines de trabajar, puedes desactivar el entorno virtual con el siguiente comando:

bash

Copiar código

deactivate

-- Django configuracion

pip install django

django-admin startproject mysite . (Poner un punto al final para que cree el proyecto en una sola carpeta)

python manage.py runserver

settings y url son los mas importante dentro de proyecto Raiz

path (admin/)

path (home/)

MIGRACIONES

Siempre ejecutar a inicio estos comandos para ejecutar las tablas de administración de django

python manage.py makemigrations

python manage.py migrate

**Crear las migraciones para los cambios en los modelos**

Cuando haces cambios en los modelos (como crear, modificar o eliminar tablas en la base de datos), debes crear migraciones para que Django registre estos cambios. Hazlo con:

bash

Copiar código

python manage.py makemigrations

**Nota**: Si quieres crear migraciones solo para una app específica (por ejemplo, una app llamada myapp), puedes especificarlo así:

bash

Copiar código

python manage.py makemigrations myapp

**2. Aplicar las migraciones a la base de datos**

Después de crear las migraciones, debes aplicarlas a la base de datos para que se reflejen en ella. Usa el siguiente comando:

bash

Copiar código

python manage.py migrate

**Nota**: También puedes especificar una app en particular si solo quieres aplicar las migraciones de esa app:

bash

Copiar código

python manage.py migrate myapp

**Resumen:**

1. **Cada vez que cambies los modelos**:
   * Ejecuta python manage.py makemigrations para crear las migraciones.
2. **Para aplicar los cambios en la base de datos**:
   * Ejecuta python manage.py migrate.

DJANGO SHELL

python manage.py shell

Investigar sobre los comando del Shell de django

-- from myapp.models import Project, Task

-- p = Project(name="aplicación movil")

-- p.save()

-- Project.objects.get(id=1)

Agregando tareas

from myapp.model import Project, Task

p = Project.objects.get(id=1)

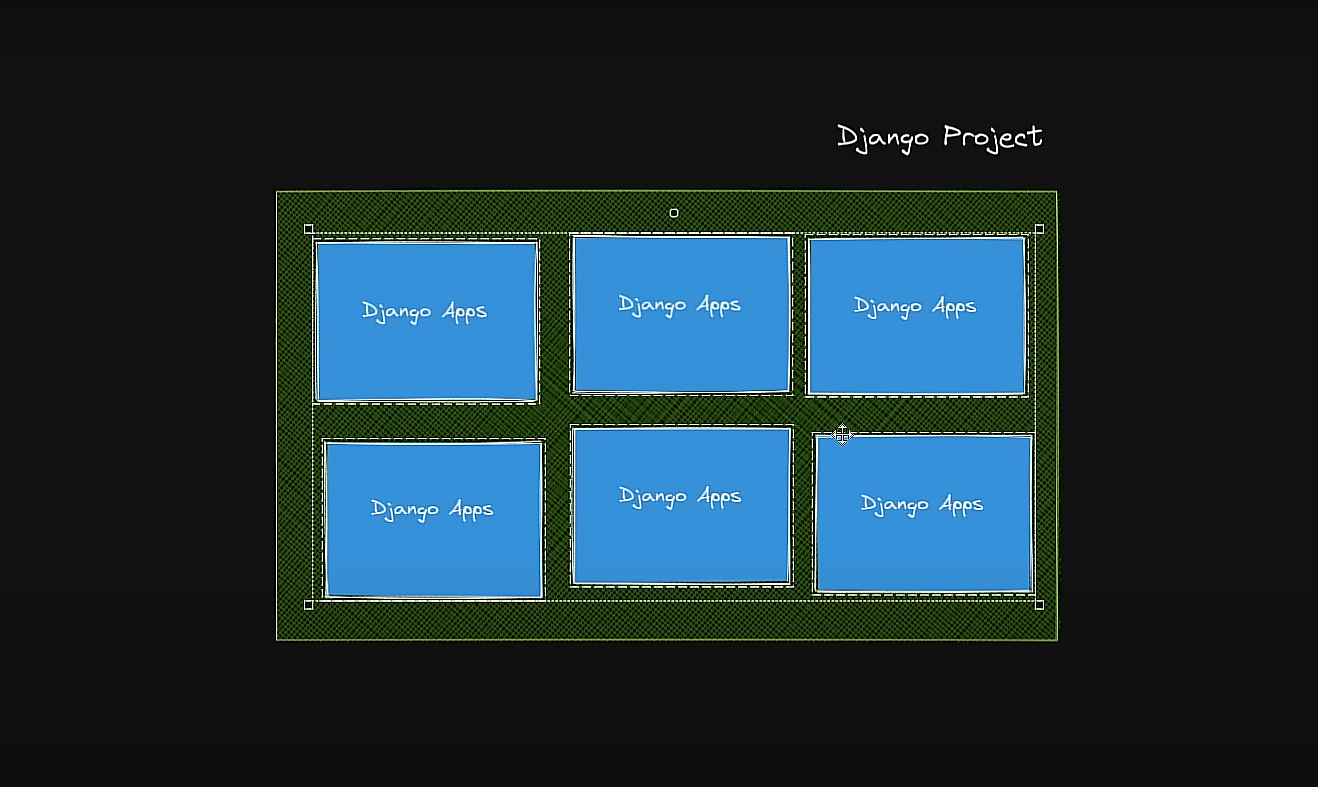
p.task\_set.all()

p.task\_set.create(title="descargar IDE")

p.task\_set.create(title="desarollar login")

PARAMSs

Estructura raíz Proyecto y Apps



APPS

Crear apps dentro el proyecto

python manage.py startapp blog

El archivo views.py es para ejecutar las vistas

INCLUDE

Agregar urls.py en la carpeta de raíz de la app