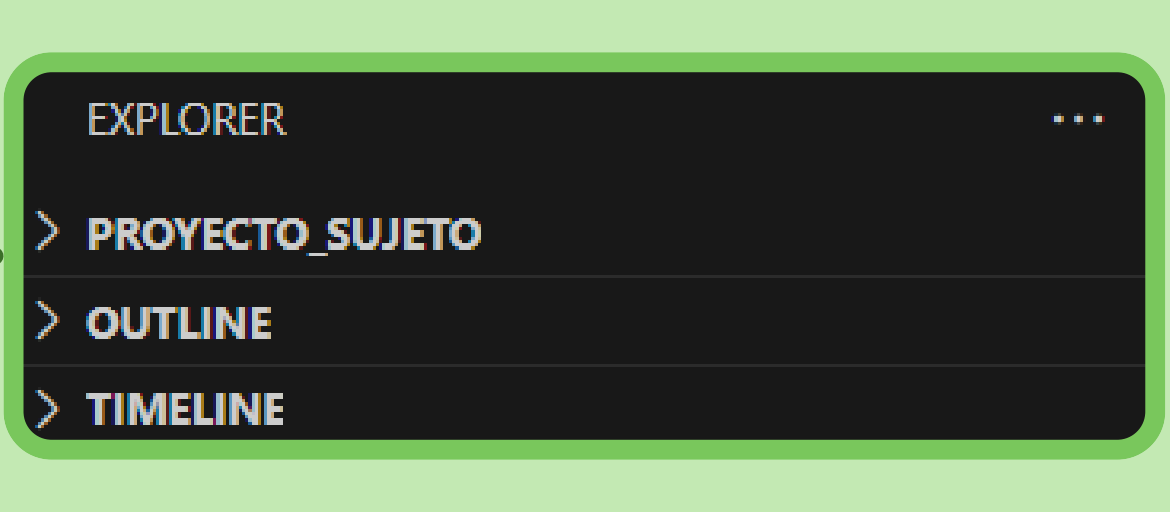
¿Cómo hacer una App en django?

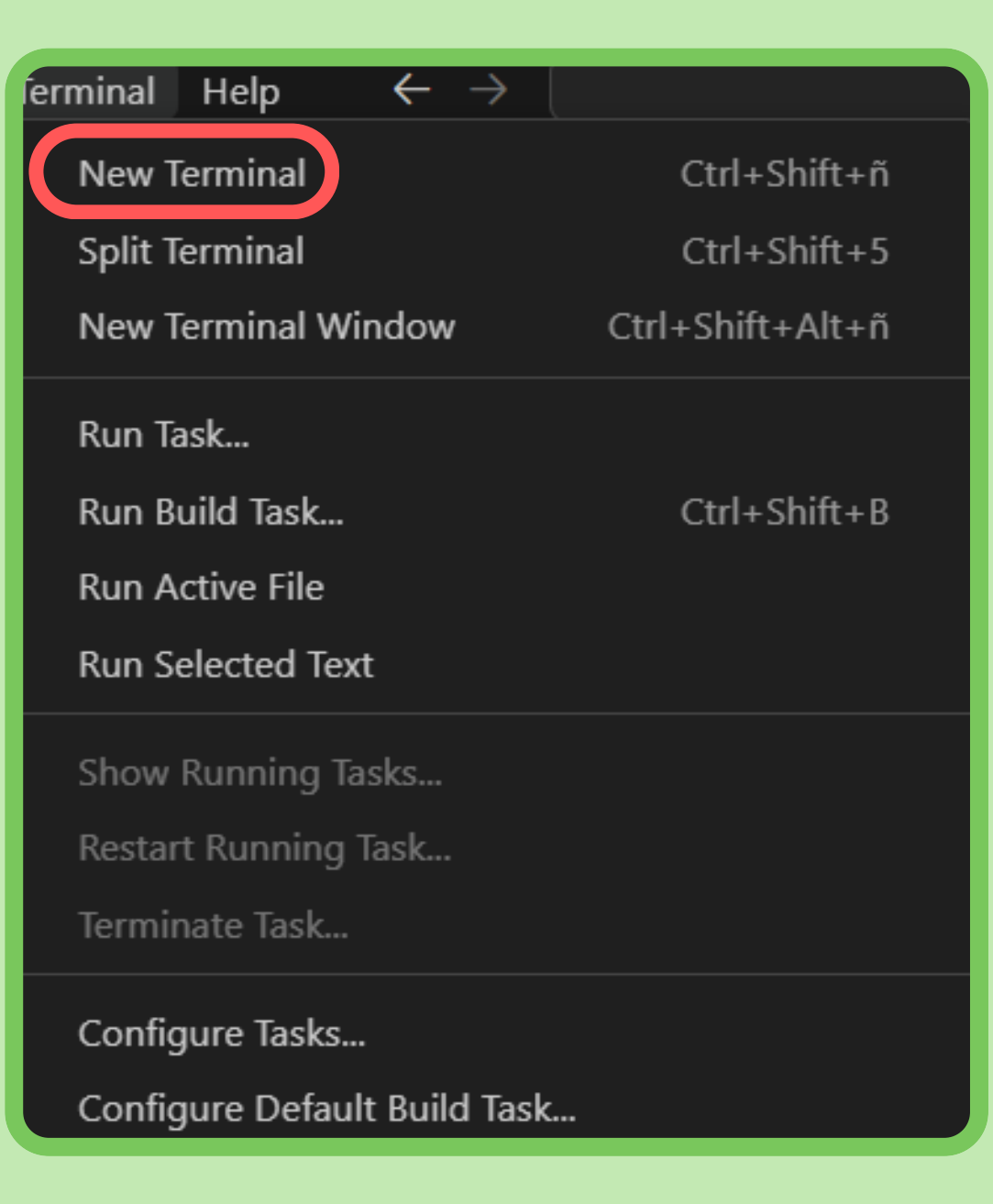
1 Crear una nueva carpeta

En esta nueva carpeta es en donde vamos a crear nuestra app, en nuestro caso la nombramos 'proyecto_sujeto' y abrimos esta nueva carpeta en **VScode**



2 Abrir una nueva Terminal en VScode

Al abrir Vscode vamos a la sección "**Terminal**" en la barra de herramientas y seleccionamos "**New Terminal**"



Con la terminal abierta vamos a escribir el siguiente comando para crear el ambiente consola:

python -m venv venv

Luego ponemos el siguiente comando:

.venv\Scripts\activate

Después de poner los comandos, la terminal debería verse así:

```
(venv) C:\Users\User\Desktop\proyecto_sujeto>
```

3 Descargar Django

Para descargar Django desde nuestra consola, vamos a poner el siguiente comando:

pip install django

4 Crear Proyecto

Con Django ya descargado, para crear un nuevo proyecto vamos a utilizar el siguiente comando:

django-admin startproject mysite (nombre_de_su_proyecto)

Este comando genera algunas carpetas y archivos tipo .py necesarias para la creación de la página. Nuestra carpeta proyecto_sujeto deberia tener la misma estructura que el siguiente ejemplo:

```
djangotutorial/
    manage.py
    mysite/
        __init__.py
        settings.py
        urls.py
        asgi.py
        wsgi.py
```

Para finalizar la preparación del entorno de django escribimos en la terminal el siguiente comando:

python manage.py startapp polls

Este comando crea un nuevo directorio llamado "polls" que tendrá algunos archivos similares a cuando iniciamos el proyecto, entre ellos uno que usaremos más adelante será models.py

```
polls/
    __init__.py
    admin.py
    apps.py
    migrations/
        __init__.py
    models.py
    tests.py
    views.py
```

5 Agregar Base de Datos

Django tiene la particularidad de que a la hora de agregar bases de datos hay que crearlas directamente en **models.py** para luego migrarlas en un proceso dentro del propio **framework**.

A continuación, presentamos la base que hemos definido para este ejemplo:

```
from django.db import models

class Frase(models.Model):
    texto = models.CharField(max_length=200)

    def __str__(self):
        return self.texto
```

*El nombre de la clase es el nombre de la columna y las variables dentro de la clase representan las columnas y su tipo

6 Migrar Models

Las migraciones en Django son el sistema para aplicar y revertir cambios en la estructura de la base de datos de forma reproducible y consistente, para aplicar la migración en consola debemos usar los siguientes comandos:

python manage.py makemigrations

python manage.py migrate

Después de realizar la migración aparecerá un nuevo archivo en la carpeta **migrations**, el nombre de este archivo va a ser **0001_initial.py** (dependiendo del número de la migración)

```
migrations
  > __pycache__
    __init__.py
    0001_initial.py
```

7 Agregar a la Base de Datos

Para agregarle un valor a nuestra tabla **Frase**, vamos a necesitar los siguientes comandos:

python manage.py Shell

Se nos abrirá el **Shell** de Python, en donde escribiremos los siguientes comandos para añadir la frase a nuestra base de datos:

from app_sujeto.models import Frase

Frase.objects.create(texto="Hola mundo!")

exit()

8 Crear Sujeto

Ahora necesitamos al **sujeto** que va a decir la frase, este lo crearemos en un archivo aparte dentro de nuestro directorio junto a dónde tengamos **models.py**, para nuestro ejemplo será un archivo llamado **sujeto.py** en el que crearemos la clase sujeto con el siguiente fragmento de código:

```
from app_sujeto.models import Frase

class Sujeto:
    def __init__(self):
        self.frase = ""

    def modificar_frase(self, nueva_frase):
        self.frase = nueva_frase

    def hablar(self):
        return self.frase
```

9 Crear las Vistas

Las vistas son funciones o clases que reciben una solicitud web, procesan la lógica (como interactuar con la base de datos o con otros servicios) y devuelven una respuesta, generalmente en formato HTML.

```
from django.shortcuts import render
from app_sujeto.models import Frase
from app_sujeto.sujeto import Sujeto

def mostrar_sujeto(request):
    frase = None
    if request.method == "POST":
        sujeto = Sujeto()
        primera_frase = Frase.objects.first()
        if primera_frase:
            sujeto.modificar_frase(primera_frase.texto)
            frase = sujeto.hablar()
    return render(request, "app_sujeto/index.html", {"frase": frase})
```

En este ejemplo, la vista usa **render** para mostrar una plantilla **HTML**. Su objetivo es hacer que el Sujeto muestre una frase. Se inicializa la frase y, si la solicitud es **POST**, se crea un objeto Sujeto con esa frase (usando solo la primera frase disponible con **.first()**)

10 Agregar la vista a urls.py

El urls.py del proyecto es el que se crea automáticamente con **django-admin startproject**

Mientras que el de cada app se debe crear manualmente cuando se necesite organizar las urls por aplicación.

11 Crear Plantilla HTML

Para la plantilla HTML, vamos a crear una carpeta llamada "**templates"** dentro del directorio de la aplicación.

Dentro de esa carpeta, añadimos otra subcarpeta con el nombre de la app (en este caso, "**app_sujeto**"), y dentro de ella creamos el archivo **index.html.**

```
<!DOCTYPE html>
<html>
<head>
    <title>Página del Sujeto</title>
</head>
<body>
    <h1>Página del Sujeto</h1>
    <form method="POST">
        {% csrf_token %}
        <button type="submit">Crear Sujeto</button>
    </form>

    {% if frase %}
        <h2>El sujeto dice: "{{ frase }}"</h2>
    {% endif %}
</body>
</html>
```

12 Correr la Página

Por último, solo necesitamos poner a correr la página, **django** solo nos permitirá correrlo de manera local, brindándonos en la consola el link al cual acceder para abrir la página con el sigueinte comando:

python manage.py runserver

```
atching for file changes with StatReloader
erforming system checks...

ystem check identified no issues (0 silenced).
ctober 23, 2025 - 22:34:14
jango version 5.2.7, using settings 'mi_proyecto.settings'
tarting development server at http://127.0.0.1:8000/
uit the server with CTRL-BREAK.

ARNING: This is a development server. Do not use it in a production setting. Use a production WSGI or ASGI server
stead.
or more information on production servers see: https://docs.djangoproject.com/en/5.2/howto/deployment/
```

*El link que nos proporcione la consola, en este caso (http://127.0.0.1:8000/), será el que deberemos ingresar en cualquier navegador con la terminal en visual abierta para ver finalmente la página que hayamos creado.