Clase 17 - Django Portfolio (parte 1)

Django

¿Qué es Django?

Ya sabemos programar en Python, es momento de empezar a aplicar un poco de todo esto para poder crear un proyecto web.

Para realizar un proyecto web necesitaremos usar un framework que nos facilite esta tarea, el más sencillo de todos los frameworks web de Python es Flask. Pero en este apartado usaremos Django que es mucho más completo y es de código abierto.

¿Qué es un framework?

Es un entorno de trabajo, un ecosistema que nos facilita crear algo, en este caso un sitio web, sin mayores esfuerzos.

Fundamentos de Django (MVC)

Modelo vista Controlador

Django se basa en el patrón de diseño web, denominado, modelo vista controlador.

Modelo: Maneja los datos.

Vista: Lo que el usuario ve.

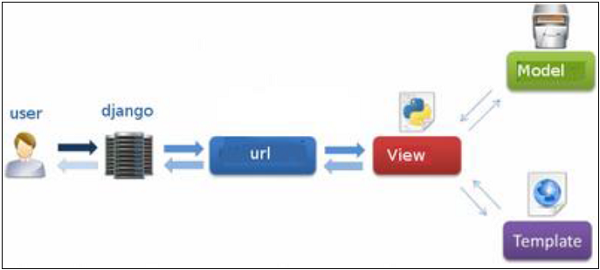
Controlador: Interacción entre los datos y lo que ve el usuario.



Esto nos permite que la aplicación sea más funcional, mantenible, escalable y colaborativa.

Este es el patrón de diseño clásico Web para cualquier proyecto web en cualquier lenguaje.

Django modifica este patrón por el modelo MTV, Model template, View.



Instalando Django

Antes que nada tenemos que tener Python instalado, nosotros ya lo tenemos.

🠮 Vamos a la página de Django: [www.djangoproject.com](http://www.djangoproject.com)

🠮 Descargamos la última o anteúltima versión.

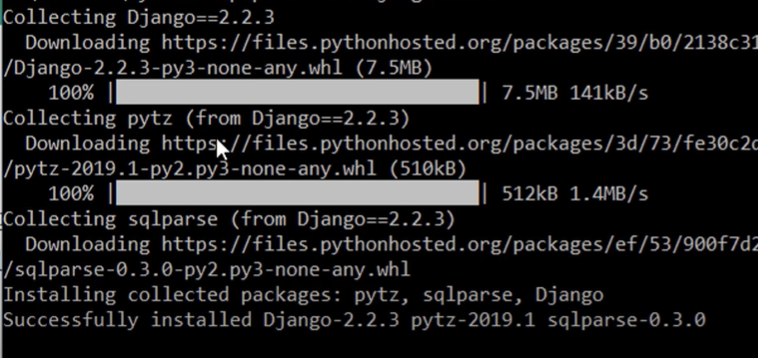
🠮 Con Windows, podemos instalarlo bajo el comando pip install Django

Instalando Django

1. Desde el cmd o desde la terminal de VCS.



1. El proceso suele tardar alrededor de 3 a 5 minutos.

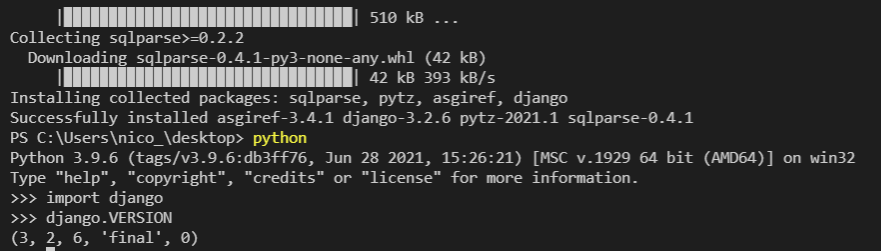


1. Comprobar que se instalo bien:

**python**

**import django**

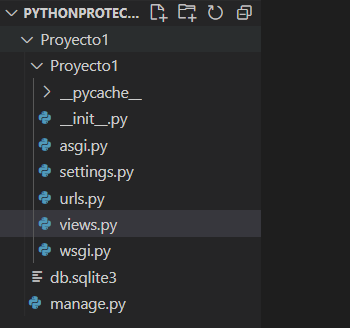
**django.VERSION**

****

Nuestro primer view

Lo primero que tendremos que hacer es crear un archivo para esta nueva vista o view.

Usualmente se suele llamar view.py, debemos crearlo en la carpeta del proyecto, es decir en la misma ruta que tenemos urls, \_\_init\_\_, wsgi, etc. Debería quedar así



Vamos a nuestro archivo views.py, e importamos los elementos de un Response de la siguiente manera:

**from django.http import HttpResponse**



Luego iniciamos creando nuestra primer vista, por medio

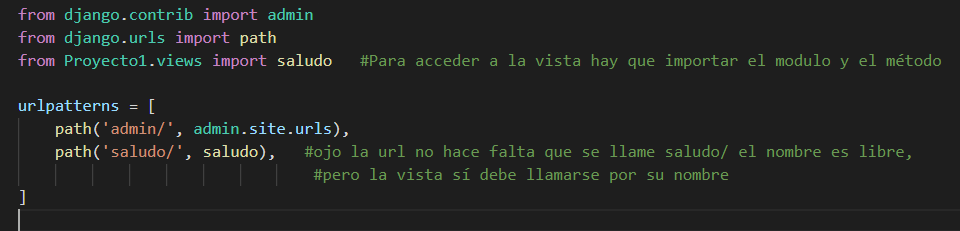
de un método que recibe como parámetro la request y nos da por resultado un response:

**def saludo(request):**

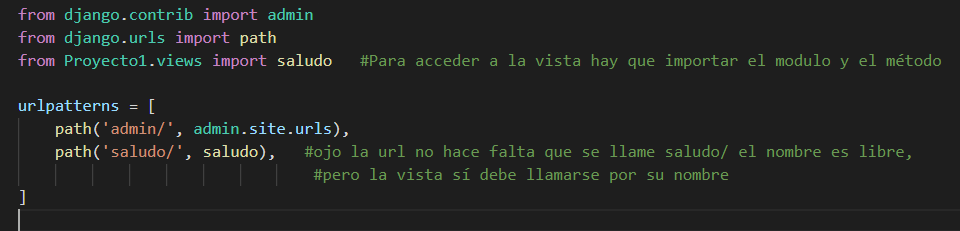
**return HttpResponse(“Hola Django - Coder”)**

****

Listo, ahora solo necesitamos avisarle a Python y Django cual será la URL que nos llevará a la vista que creamos. Eso lo hacemos en el archivo **urls.py.**



Para esto debemos primero importar Proyecto1.views y generar el vínculo entre una url y la vista, nos debería quedar así:



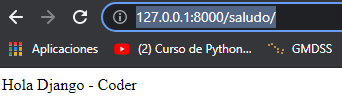
Solo resta probar que funcione lo que hicimos.

Para esto nos apoyamos en lo visto anteriormente y arrancamos el servidor de la siguiente manera:

**python manage.py runserver**

Nuestro primer view

Luego entramos a: [**http://127.0.0.1:8000/saludo/**](http://127.0.0.1:8000/saludo/)

****

Ya podemos ver nuestra primer página, o mejor dicho nuestra vista

Agregar otra view

Hagamos lo mismo para ver lo sencillo que ha sido:



Pasaje de parametros

¿Qué es?

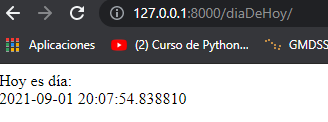
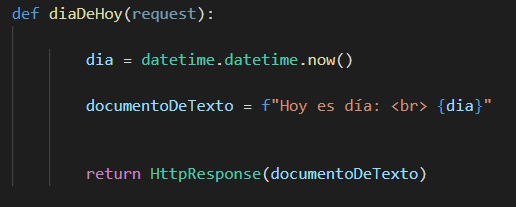
Muchas veces queremos mostrar en una vista el resultado de algún proceso interno realizado en Python, esto es básicamente enviar parámetros por medio de la vista.

Veamos cómo se hace, es muy simple y parecido a la anterior.

Usar para slides de texto e imagen. Si no alcanza, no sobrecargar, usar otra con el mismo título para indicar que continúa el mismo módulo.

Ejemplo

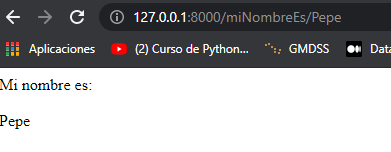
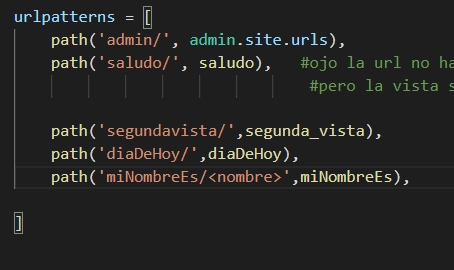
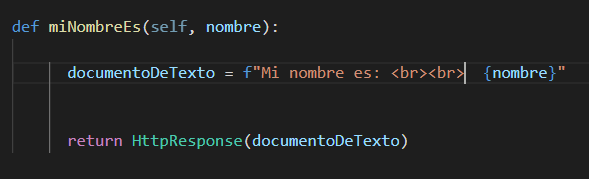
Así es el proceso de creación de una nueva vista que muestra DíaDeHoy.



Parámetros desde la URL

¿Qué es?

Mi vista puede trabajar sobre algún dato que nos enviar por url, veamos cómo funciona así la respuesta aparece sola:



Plantillas Django

Plantillas

¿Qué son?

Son archivos que nos permiten separar la vista de la estética, es decir guardar en un archivo separado de todo lo que guardabamos en “documento”, para así enviar por la HttpsResponse.

Entonces, ¿podríamos decir que así se crea un template?

¡Exacto!

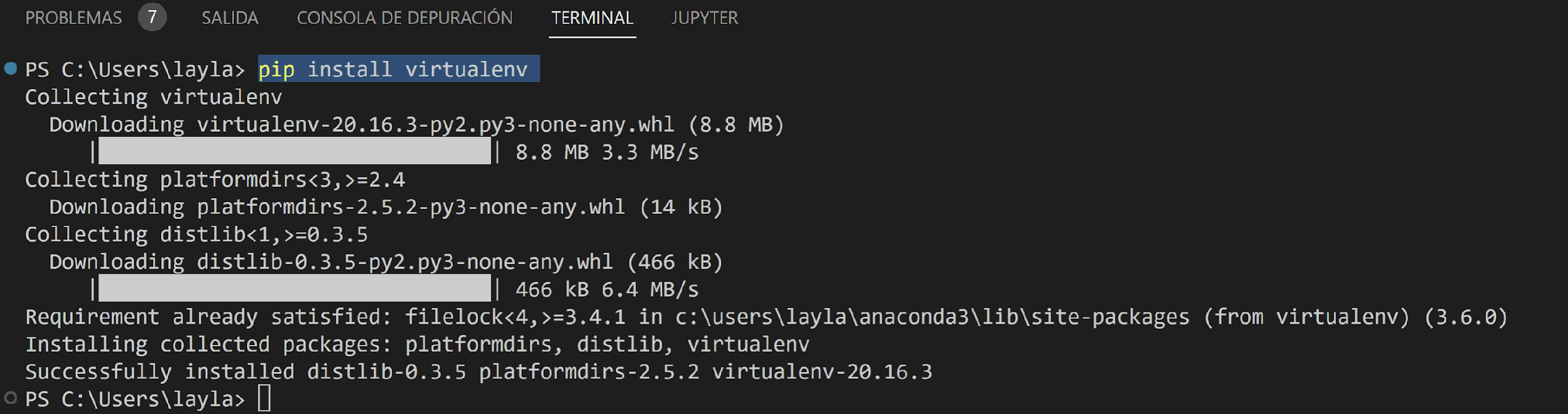
Recordemos que Django se basaba en Modelo,Vista, Template.

Por último, hablemos de entornos Virtuales y Paquetes en Python…

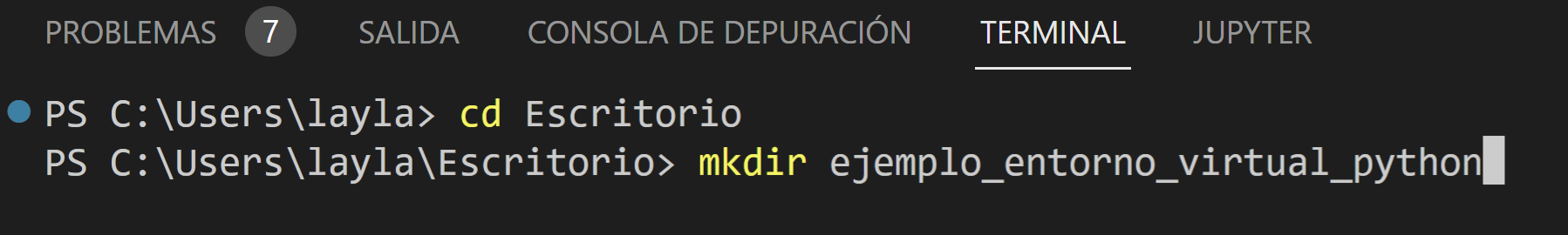
Los entornos virtuales se pueden describir como directorios de instalación aislados. Este aislamiento te permite localizar la instalación de las dependencias de tu proyecto, sin obligarte a instalarlas en todo el sistema.

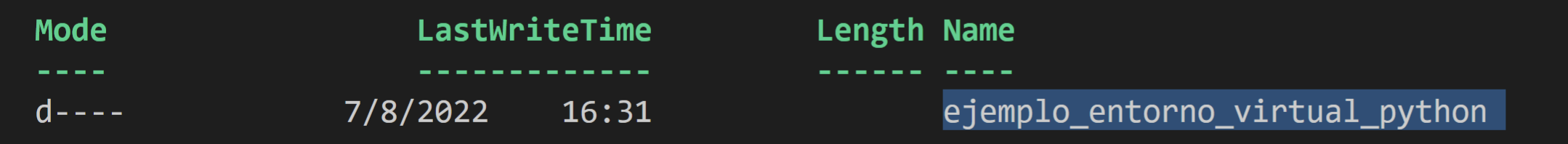
Muchas veces se recomienda a la hora de programar, usar este tipo de herramientas. Es importante mencionar que los entornos virtuales, pueden crearse en diferentes lenguajes de programación, en este caso en particular, veremos como crearlos en Python.

Primero de todo, instalamos la librería: pip install virtualenv

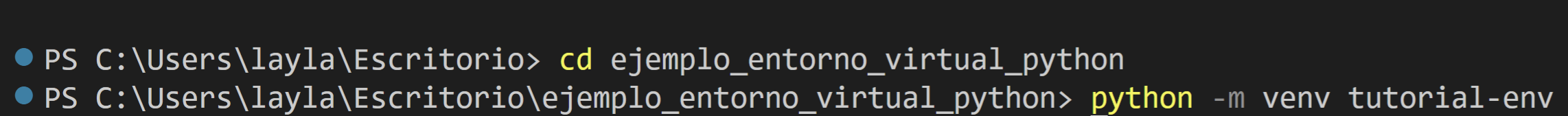


Para crear un entorno virtual, debemos decidir en que carpeta lo queremos crear y ejecutar el módulo venv como script con la ruta a la carpeta. Por ejemplo, creamos una carpeta en el Escritorio y allí lo ejecutamos.

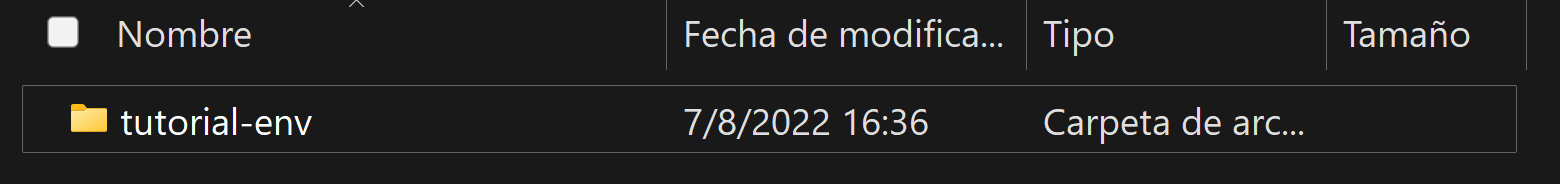
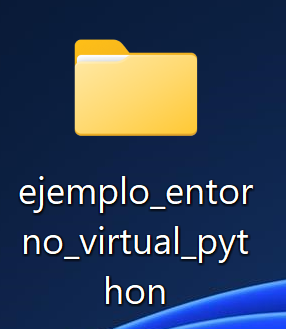




Ingresamos a la carpeta y ejecutamos el siguiente comando: python -m venv tutorial-env



Chequeamos el Escritorio y la creación de las carpetas:



El comando anterior, creó el directorio tutorial-env si no existe, y también directorios dentro de él que contienen una copia del intérprete de Python y varios archivos de soporte.

¿Quieres saber más?

Te dejamos material ampliado de la clase

Django

* [Tutorial Django básico (Python)](https://developer.mozilla.org/es/docs/Learn/Server-side/Django) | Developer

Entornos virtuales

* [Creación de entornos virtuales](https://docs.python.org/es/3.8/library/venv.html#:~:text=Un%20entorno%20virtual%20es%20un,parte%20de%20tu%20sistema%20operativo).
* [Entornos virtuales y paquetes](https://docs.python.org/es/3/tutorial/venv.html)