PIP Y ENTORNOS VIRTUALES

INDICE

Contenido

[PIP Y ENTORNOS VIRTUALES 1](#_Toc129802558)

[FLUJO DE TRABAJO PYTHON 2](#_Toc129802559)

[QUE ES PIP 3](#_Toc129802560)

[GRAFICAS EN PYTHON CON PIP 6](#_Toc129802561)

[USANDO ENTORNOS VIRTUALES EN PYTHON 6](#_Toc129802562)

[REQUIREMENTS 8](#_Toc129802563)

[SOLICITUDES HTTP CON REQUEST 9](#_Toc129802564)

[PANDAS 14](#_Toc129802565)

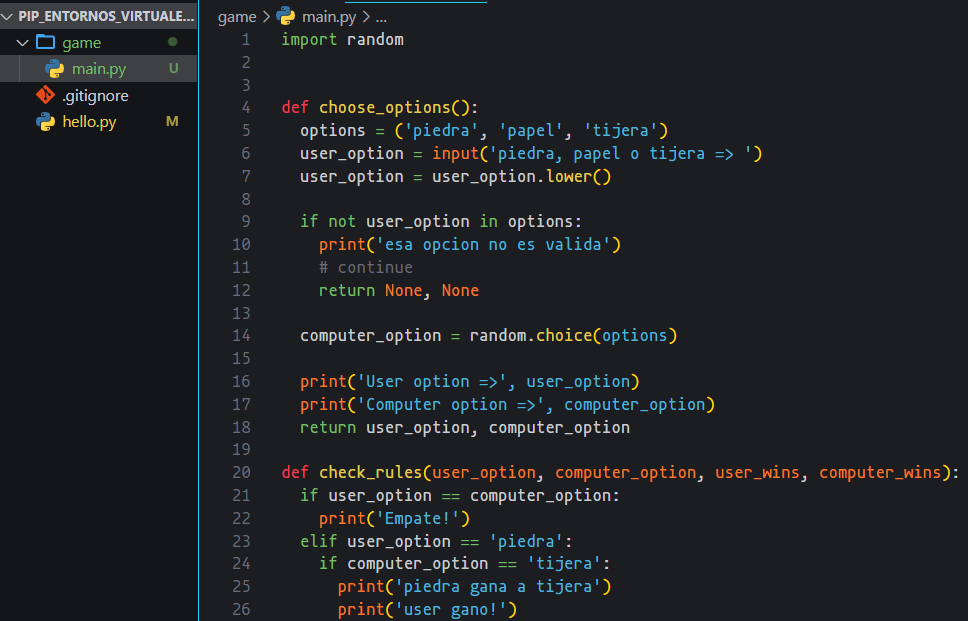
[PYTHON PARA BACKEND: WEBSERVER CON FAST API 17](#_Toc129802566)

INSTALACION EN WINDOWS WSL Y LINUX

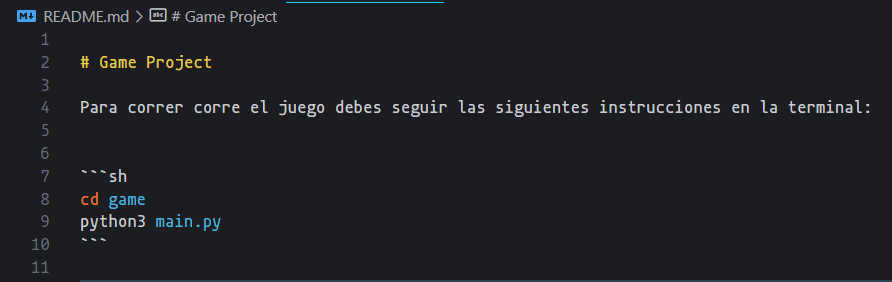
FLUJO DE TRABAJO PYTHON

Crear carpeta game y dentro el archivo main.py

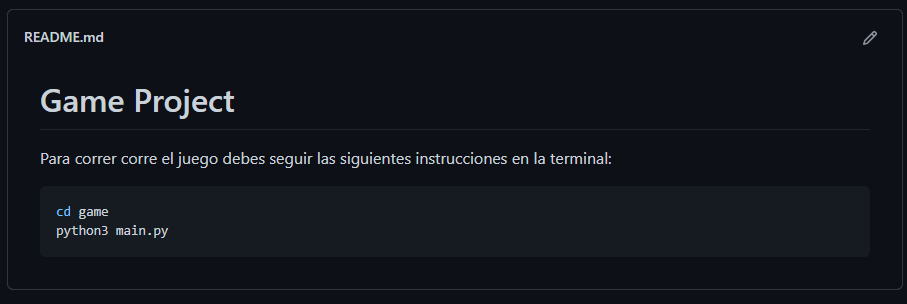
Este archivo contendrá el código de piedra papel tijera. Comprueba su funcionamiento.



Creamos el archivo README.md y colocamos el siguiente contenido.



Una vez subido a github podemos ver lo siguiente:



QUE ES PIP

ES El gestor de paquetes de Python, En pypi.org podemos ver los paquetes de Python. Como ejemplo buscamos matplotlib.



Verifica que version de pip3 tienes



Creamos la carpeta charts y nos posesionamos dentro

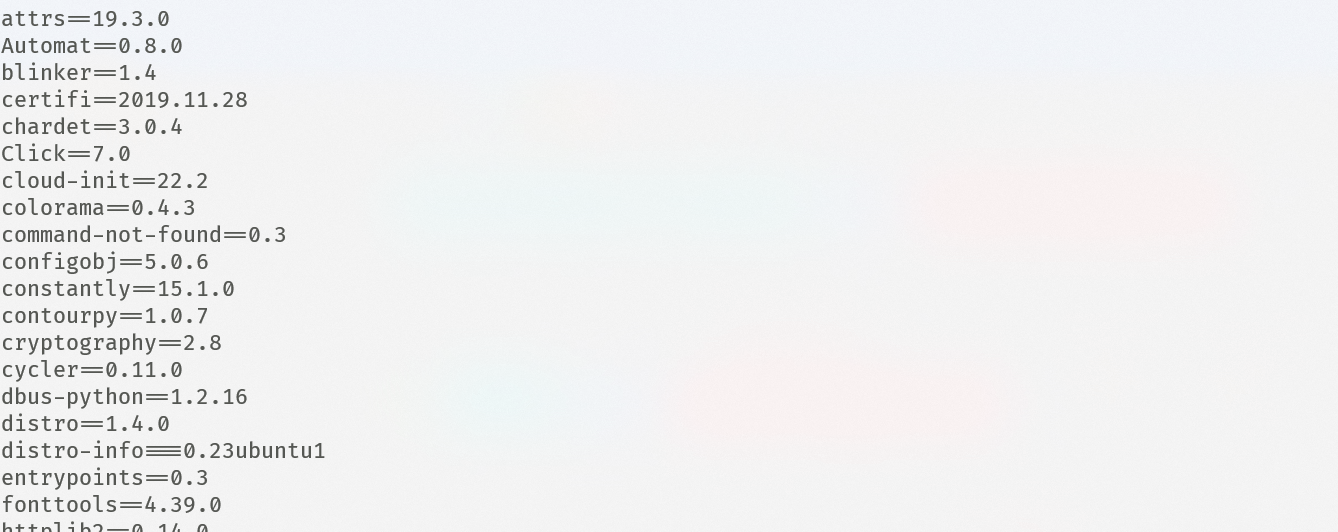


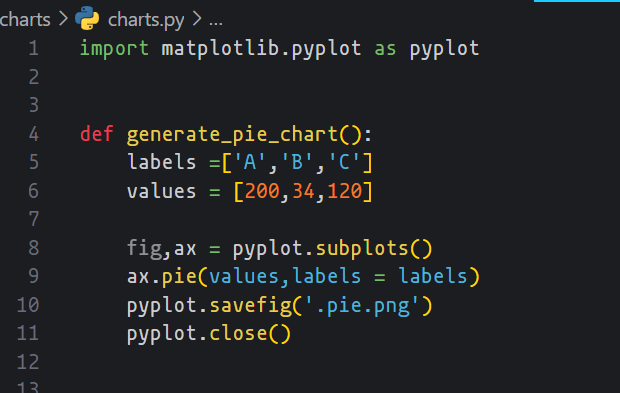
Dentro de charts instalamos matplotlib



Para ver el entorno de Python en general emplear freeze. Muestra la instalación global de Python.



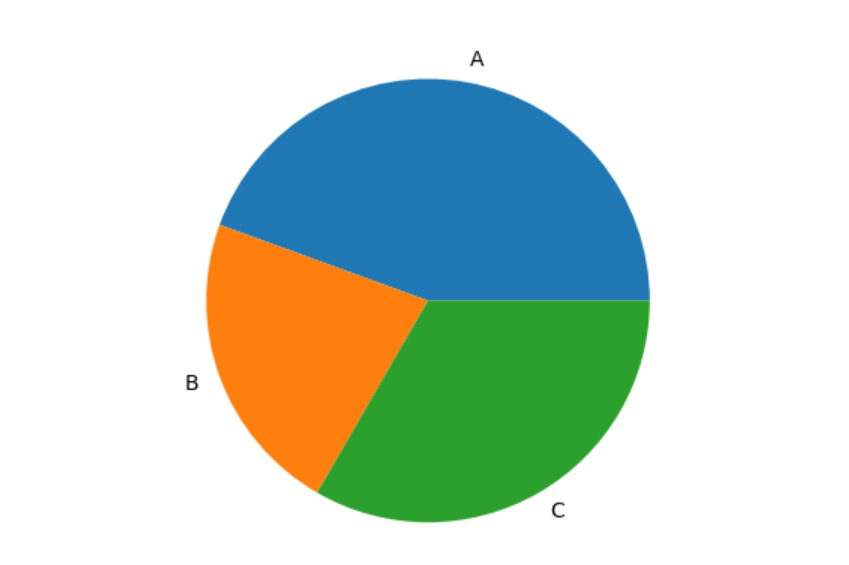




Ejecutamos main dentro de la carpeta charts



Al ejecutar generamos la imagen pie.png



GRAFICAS EN PYTHON CON PIP

USANDO ENTORNOS VIRTUALES EN PYTHON

PRIMER HAY QUE VER de donde se esta ejecutando Python, estas en un entorno global

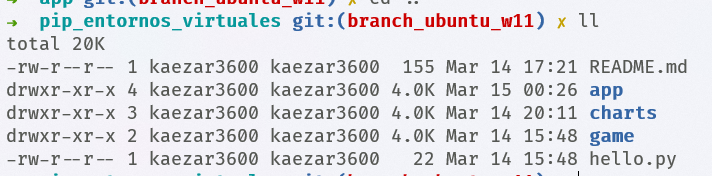




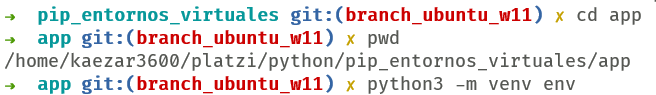
Se necesita instalar esto virtual enviroment si tienes Linux, no se necesita si tienes macOS



Como queremos que cada proyecto app, charts y game tenga su propio entorno debemos entrar carpeta por carpeta



Entramos a la carpeta, nos aseguramos que estamos allí con pwd y instalamos virtual enviroment



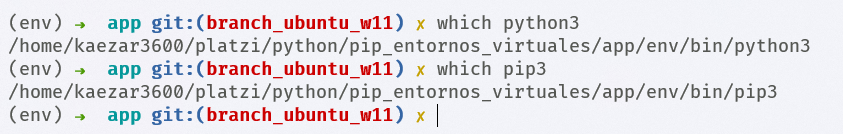
Para ingresar a ese ambiente virtual debes colocar source el nombre de tu ambiente (env) seguido de /bin/ y al final la palabra actívate.



Para desactivar



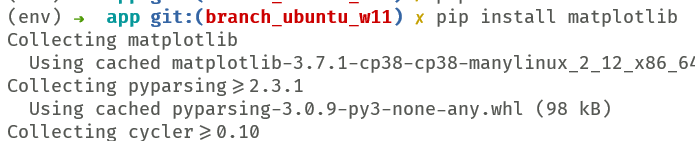
Con which sabemos en donde estamos.



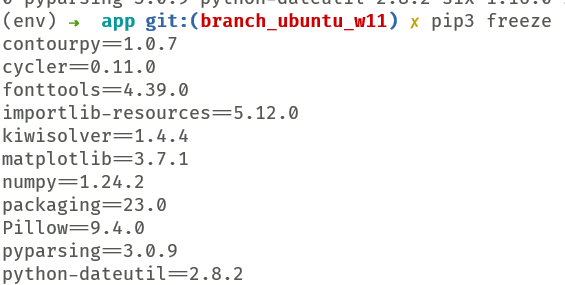
Si haces freeze no te devolverá nada por que en el ambiente virtual no se isntalo nada, esta vacio. Ahor toca instalar matplotlib



Se instala matplotlib.



Verificar con freeze.



Para instalar otro ambiente en otra carpeta debes desactivar el ambiente actual

REQUIREMENTS

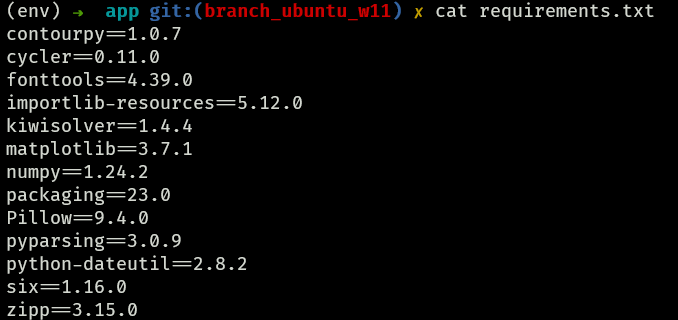
Se quiere Instalar otra librería, la solución es automatizar las instalaciones

Para esto crearemos un archivo llamado requirementx.txt que contendrá la lista de las librerías requeridas para que funcione la aplicación.

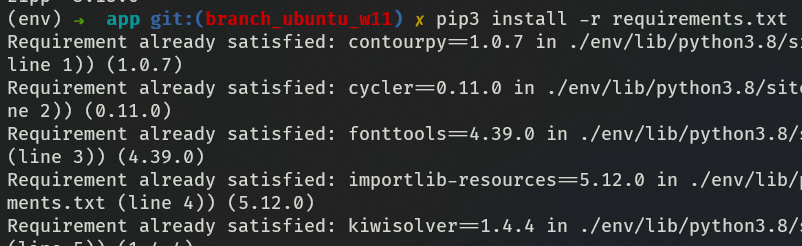
El símbolo > significa que lo que salga de freeze(output) a un archivo de texto requirement.txt



Sirve para instalar las dependencias de un proyecto en una sola línea de comando. Para no instalar dependencia por dependencia.



De esta forma se instala el ambiente par que trabaje el proyecto



Luego sigue la forma de trabajar en grupo haciendo fork

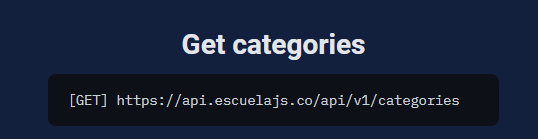
PRACTICA

SOLICITUDES HTTP CON REQUEST

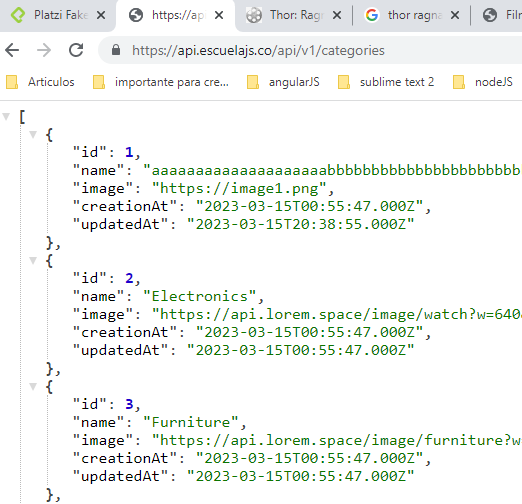
Hace peticiones a otro tipo de servidores web desde python



Vamos a utilizar get para categories



Puedes buscar los datos utilizando la url en el navegador.



Creamos un a nueva carpeta llamada web-server



Ingresamos a la carpeta



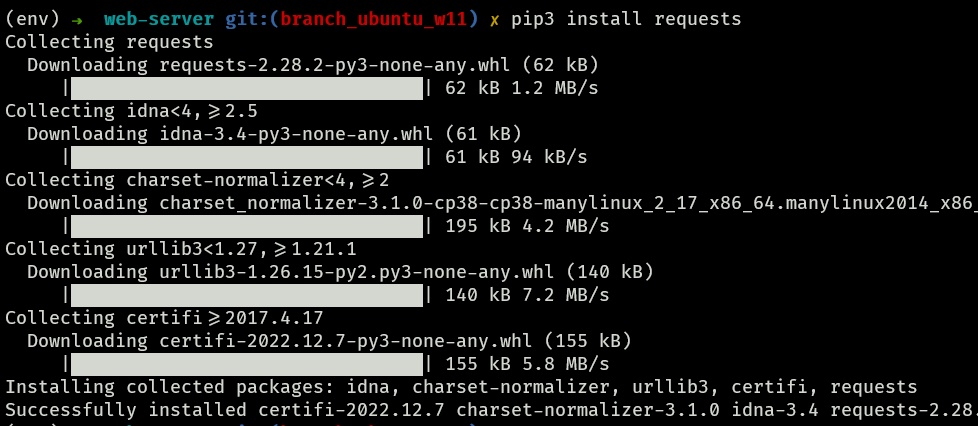
Creamos un ambiente virtual



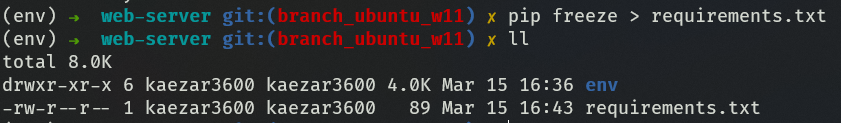
Activamos el ambiente



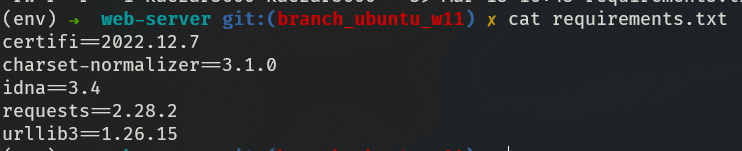
Instalamos requests



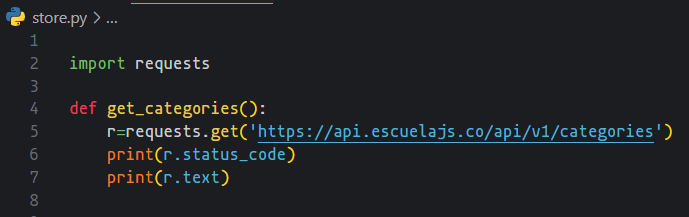
Creamos un requirements.txt y verificamos su existencia con ll



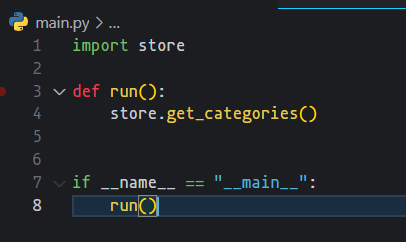
Verificamos el contenido de requirements.txt



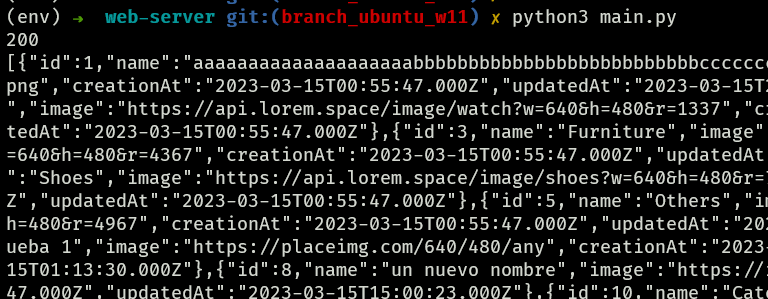
Se crea el archivo store.py



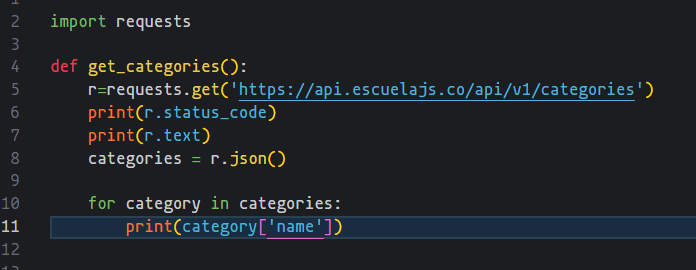
Y el archivo main.py



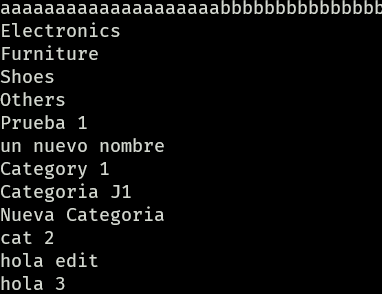
Si ejecutas el código, veras el código 200 que significa que esta bien y el string que tiene forma de lista.



El método json() transforma el request que es un string a una lista



Al ejecutar main.py vemos que se itero el resultado de request

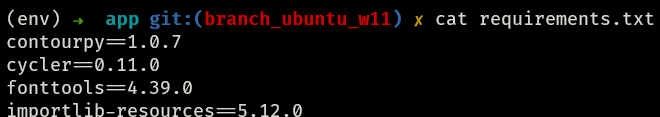


PANDAS

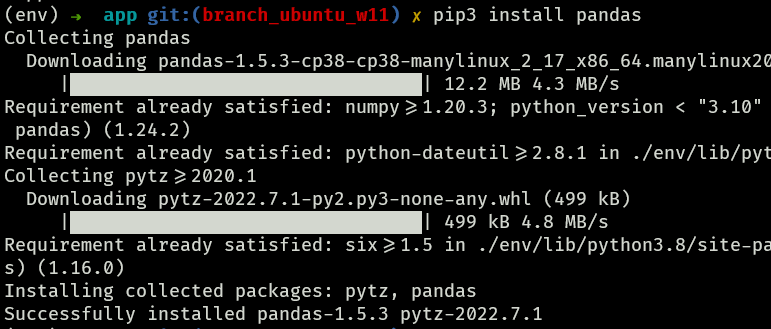
Librería que sirve para analizar y manipular datos de archivos duros

Entrar a proyecto app y activar el entorno

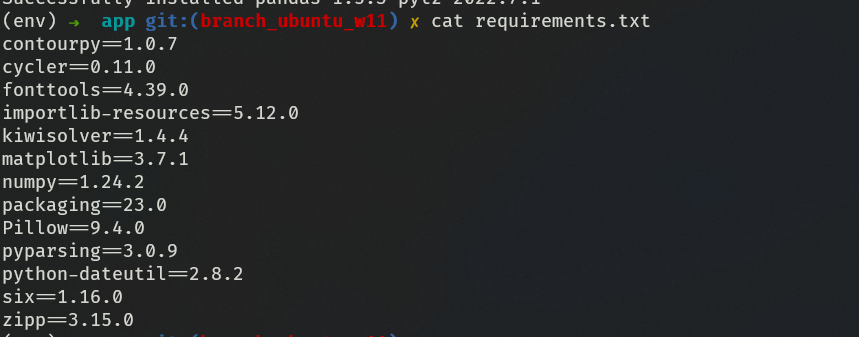
Verificar el contenido de requirement



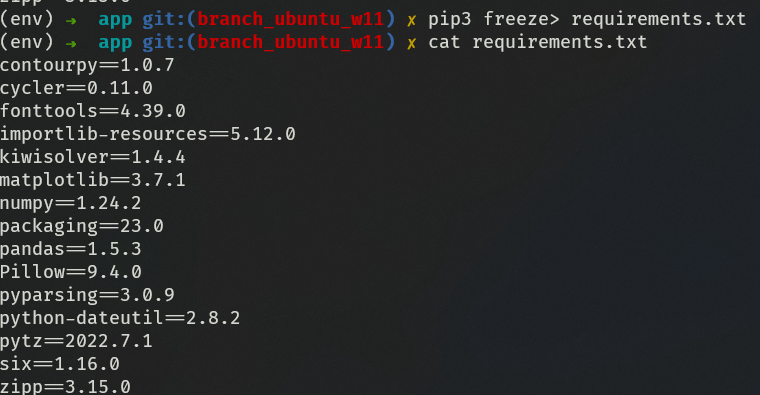
No existe pandas, instalar pandas



Si volvemos a visualizar el contenido de requirements.txt veremos que no estas pandas



PARA ESTO se debe actualizar el archivo volviendo a ejecutar freeze > requirements.txt. Ahora veras la librería pandas en la lista



Modificaremos el main.py

Utilizaremos panda como pd



PYTHON PARA BACKEND: WEBSERVER CON FAST API

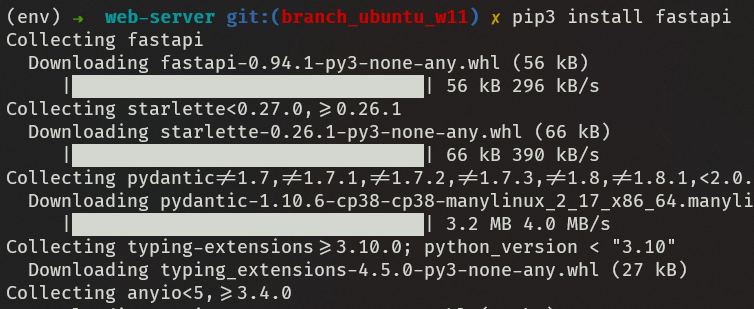
USAREMOS FAS T API

Uvicorn es un servidor web

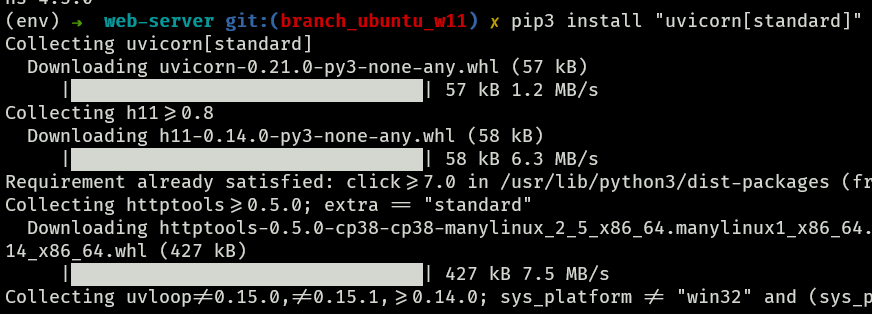
Trabajaremos sobre el proyecto web-server

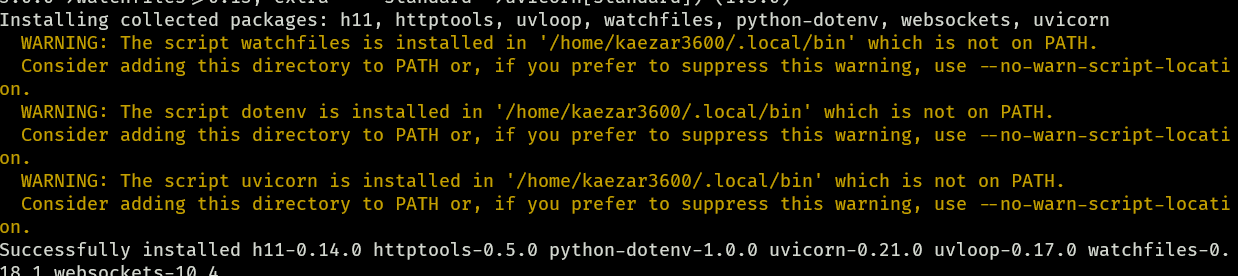
Iremos a la carpeta web-server y activaremos el entorno virtual

Se instalará fast api

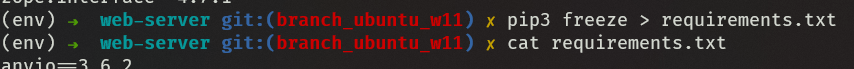


Ahora instalaremos uvicorn





Actualizamos y verificamos requirements

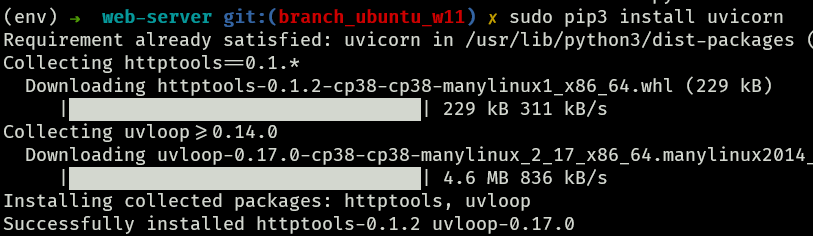


PROBLEMAS CON LA INSTALACION DE UVICORN

ME HA FUNCIONADO

Pip3 list| grep uvicorn





Creamos el archivo store.py

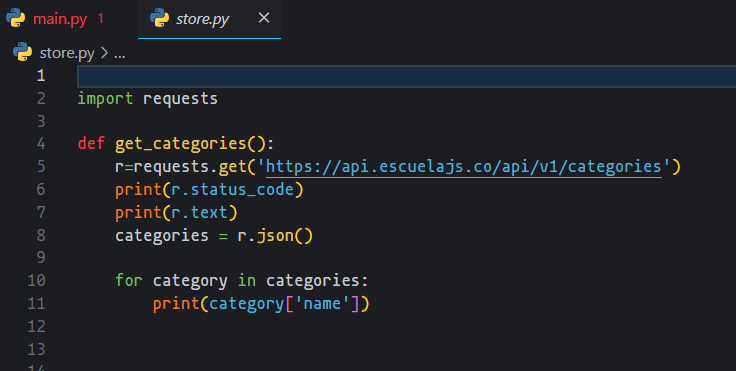
Dentro importamos requests

Creamos la función get\_categories

Mediante el método get de request recibe la respuesta del servicio y se almacena en la variable r

R debe ser transformada en json sino se mantendría como string

El contenido será de tipo list se hará loops y se accedera al nombre



En main.py

Se importa store y fastapi.

Se inicializa y se almacena en la variable app.

Se define la función get\_list que retorna un array.

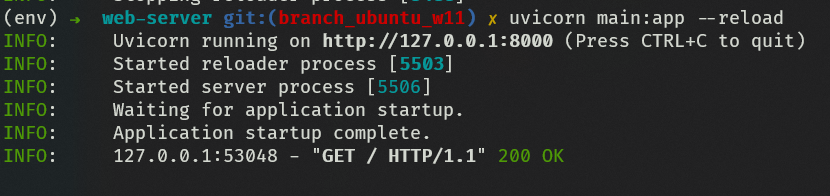
Luego se define un decorator : @app.get(‘/’) que idica que si en el navegador te encuentras en la raíz entonces ejecutes get\_list.

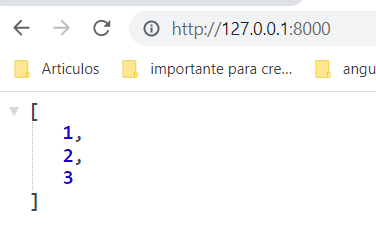
Lo mimo ocurre debajo donde si ingresas a /contact devuelve un diccionario.

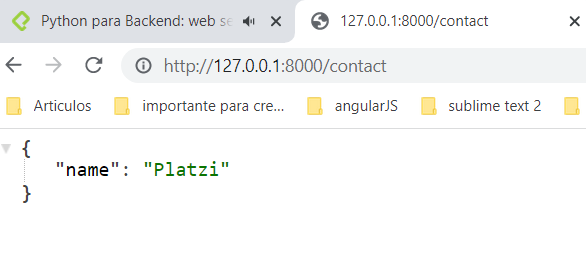
Debajo la función run es ejecutada por defecto y esta ejecuta la función get\_categories de store.



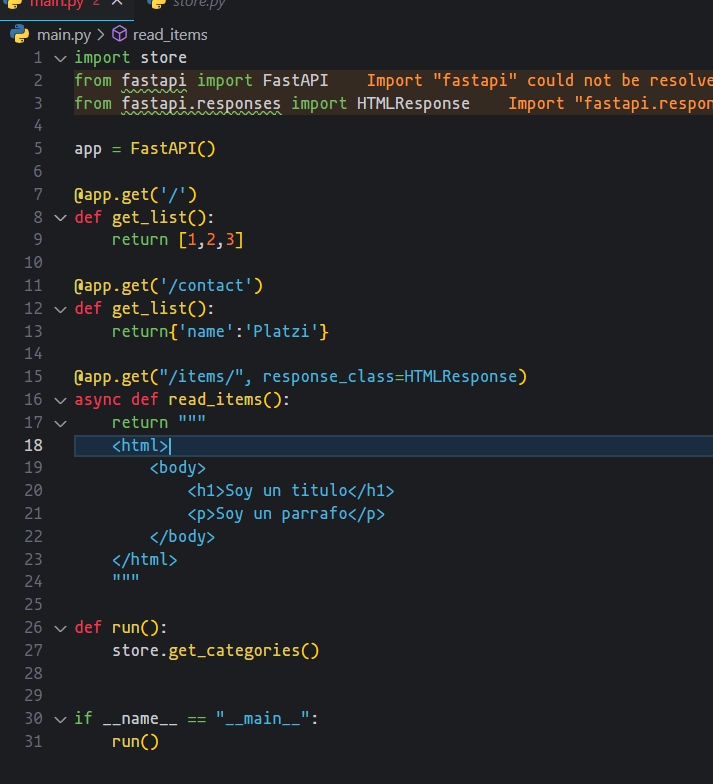
Ejecutar uvicorn, reload sirve para actualizar

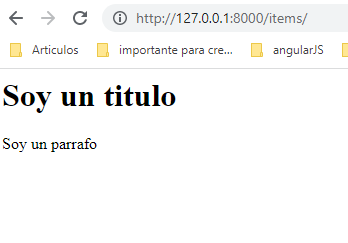




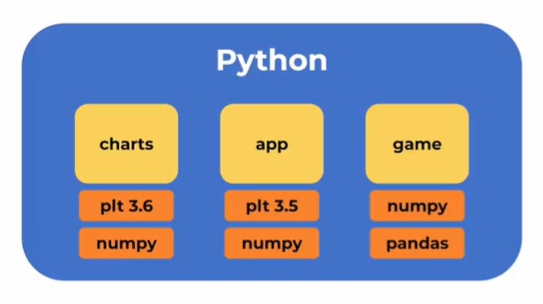


Fast api también puede devolver paginas dinámicas

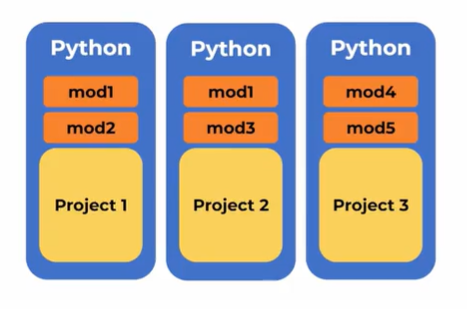




Que es Docker?



Docker aisla entornos de ejeucion



DOCKERIZACION SCRIPT DE PYTHON