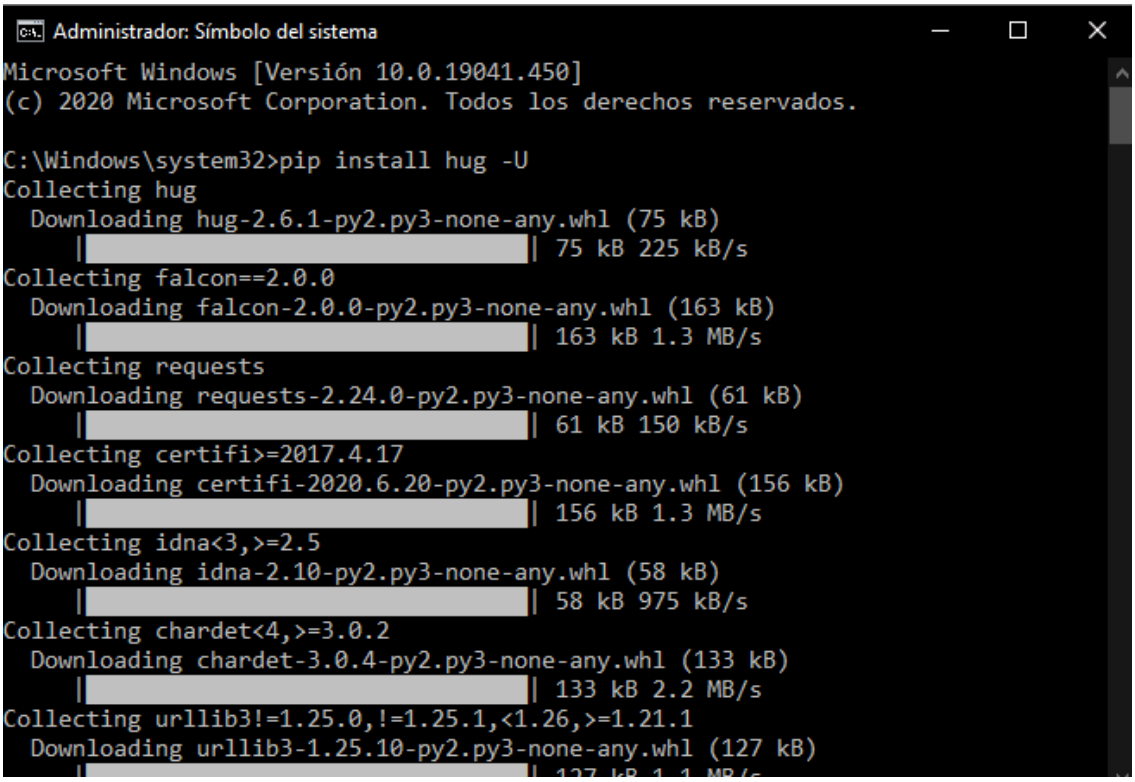
Microservicios y contenedores.

He optado por usar el framwork Hug de Python para la creación de la API Rest, primero instalé Hug



Después cree el método local.

```python
#Importamos lo modulos necesarios
import datetime
import hug

#Indica que será de solo acceso local
@hug.local()

#Definimos la función say_hello
def say_hello(name: hug.types.text, age: hug.types.number, hug_timer=3):#hug.typ
    year_of_birth = datetime.datetime.now().year - age
    return {#Decimos hola al usuario y calculamos su año de nacimiento
        'message': "Hola {0}, naciste el año {1}".format(name, year_of_birth),
        'took': float(hug_timer)
    }

if __name__ == '__main__':
    print(say_hello("panchito", 50))
```

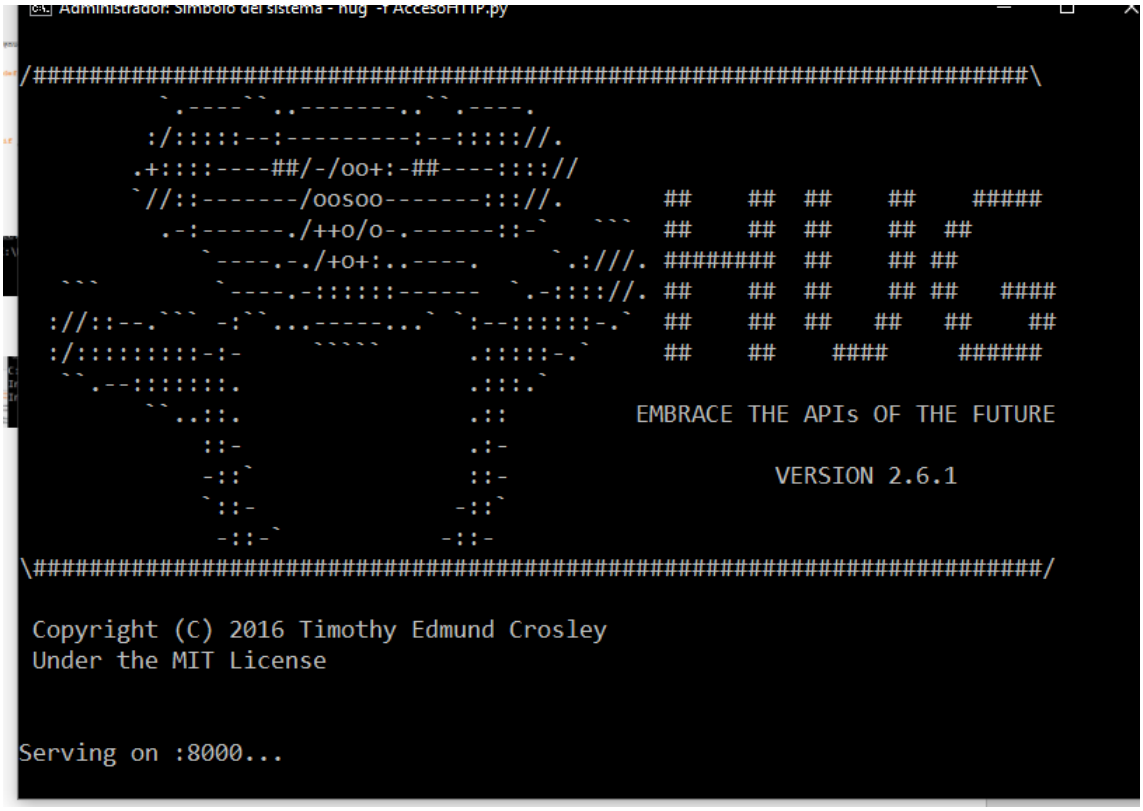
Después cree el método HTTP

```python
import datetime
import hug


@hug.get(examples="name=Jhon Doe&amp;amp;age=30")
@hug.local()
def say_hello(name: hug.types.text, age: hug.types.number, hug_timer=3):
    """Decimos hola al usuario y calculamos su año de nacimiento"""
    year_of_birth = datetime.datetime.now().year - age
    return {
        'message': "Hola {0}, naciste el año {1}".format(name, year_of_birth),
        'took': float(hug_timer)
    }


if __name__ == '__main__':
    print(say_hello("Juanito", 23))
```

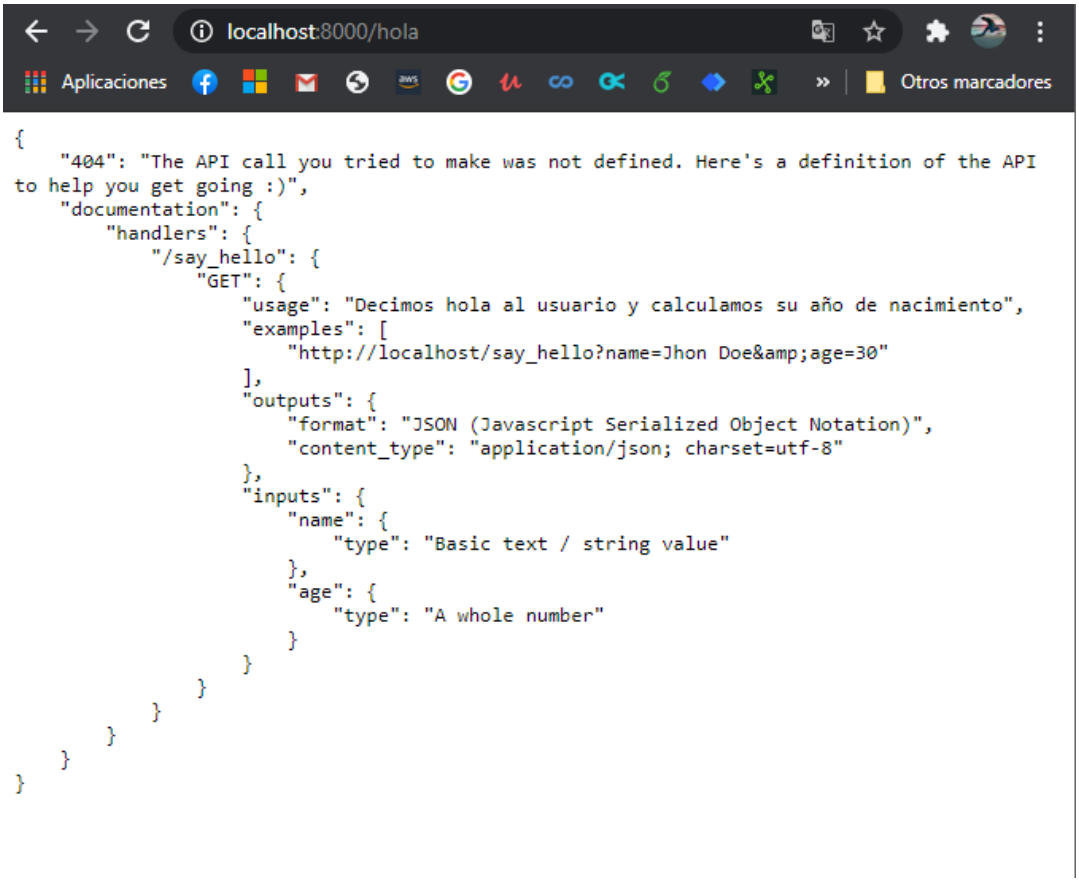
El método HTTP, lo guarde en la dirección api\sps\helloworld\v1 y ejecute el método.

```
C:\Users\KzyCk\Documents\Python\api\sps\helloworld\v1>hug -f AccesoHTTP.py
```

En este momento la API ya esta expuesta, por medio del puerto 8000.

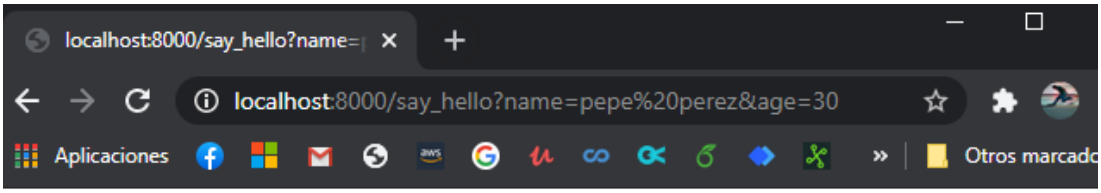


Visualizando desde el navegador se ve lo siguiente.




```
{
    "404": "The API call you tried to make was not defined. Here's a definition of the API
to help you get going :)",
    "documentation": {
        "handlers": {
            "/say_hello": {
                "GET": {
                    "usage": "Decimos hola al usuario y calculamos su año de nacimiento",
                    "examples": [
                        "http://localhost/say_hello?name=Jhon Doe&amp;age=30"
                    ],
                    "outputs": {
                        "format": "JSON (Javascript Serialized Object Notation)",
                        "content_type": "application/json; charset=utf-8"
                    },
                    "inputs": {
                        "name": {
                            "type": "Basic text / string value"
                        },
                        "age": {
                            "type": "A whole number"
                        }
                    }
                }
            }
        }
    }
}
```


Y marcando los valores de nombre y edad, observamos que se despliega en la ventana.




```
{"message": "Hola pepe perez, naciste el año 1990", "took": 0.0}
```

Después cree el método Cli.

```python
import datetime
import hug


@hug.cli()#Se usa como decorador
@hug.get(examples="name=Jhon Doe&amp;amp;age=30")
@hug.local()

#Definimos la función say_hello
def say_hello(name: hug.types.text, age: hug.types.number, hug_timer=3):#hug.typ
    year_of_birth = datetime.datetime.now().year - age
    return {#Decimos hola al usuario y calculamos su año de nacimiento
        'message': "Hola {0}, naciste el año {1}".format(name, year_of_birth),
        'took': float(hug_timer)
    }


if __name__ == '__main__':
    print(say_hello("panchito", 50))
```

Y ejecute desde la línea de comandos dicho método, se muestra la existencia del endpoint say_hello.

```
C:\Users\KzyCk\Documents\Python\api\sps\helloworld\v1>hug -f AccesoCli.py -c say_Hello "Panchito" 50
AccesoCli

Available Commands:

 - say_hello
```

Haciendo pruebas en el cliente Rest de Postman, y haciendo una solicitud tipo GET, obtenemos, lo siguiente.