PRACTICA 01

Sección:

ETL, APIs y WebScrapping con Python

Bloque:

2

Apellidos: Sánchez de la Fuente

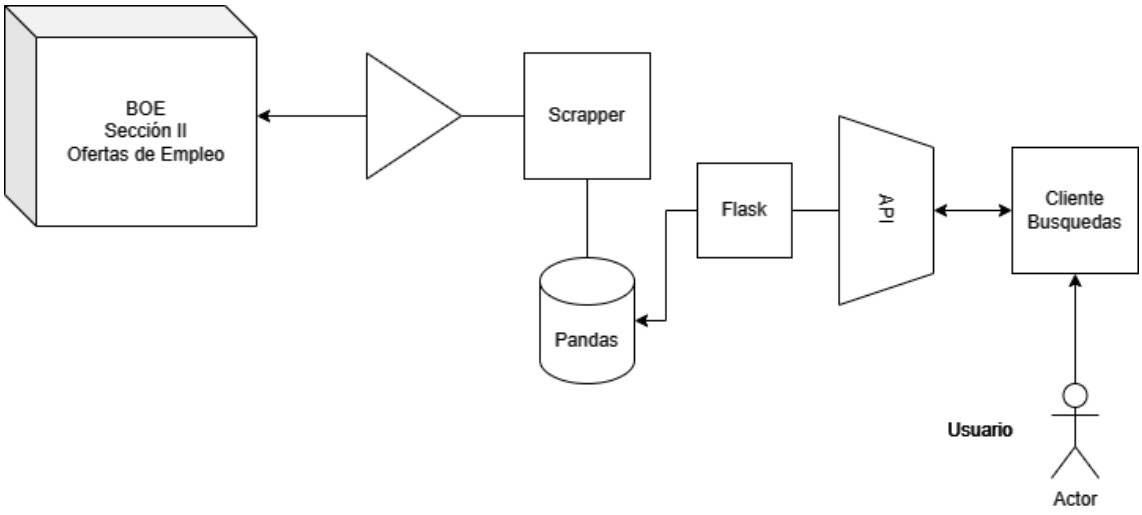
Nombre: Diego

1. Introducción:

En este documento se presenta el proyecto realizado por el alumno del Master de Big Data impartido con la colaboración de la Escuela de Organización Industrial, el proyecto realizado íntegramente en lenguaje de programación Python, busca implementar un sistema que realice un raspado del BOE, para buscar las ofertas de empleo, y ponga a disposición de un API la información obtenida, por otra parte nos conectaremos a dicho API para hacer búsquedas sobre las ofertas de empleo.

2. Arquitectura:

A continuación, se expone el diagrama de arquitectura lógica del proyecto.



Se detallan cada uno de los elementos que componen la arquitectura:

3. Módulo Scrapper:

Se integra en el notebook: Descarga BOE Ofertas Empleo y API Flask.ipynb, es la primera sección de este, se basa en una clase Python llamada: DescargaBOE, como dependencias esta clase necesita las siguientes librerías Python:

- requests
- csv
- pandas
- urllib
- datetime
- string
- urllib
- re, regex
- warnings

Esta clase contiene de manera genérica todos los pasos, para extraer información del BOE, en lo referente a la sección II Ofertas de empleo público.

Después de definir la clase, se instancia ésta, y se utiliza para descargar al menos 100 disposiciones de empleo público, empezando por el día más actual y en orden descendente, hasta completar el dataset.

```
# Rutina principal
N_REGISTROS_MINIMO = 100 # Limitamos a 100 las ofertas de empleo ya que si no se demora mucho
if __name__ == "__main__":
    BOE = DescargaBOE()
    i = 0
    while True:
        BOE.establecer_offset(i)
        if(BOE.generar_dataset() > N_REGISTROS_MINIMO):
            break
        i += 1
    BOE.obtener_dataset_final()

# Obtenemos el dataset
df = BOE.obtener_dataset_final()
```

4. Api Flask:

Se integra también en el notebook: Descarga BOE Ofertas Empleo y API Flask.ipynb, está a continuación de la parte donde se instancia la clase referente al Scrapper, esta parte se ejecuta en un bucle, por lo que hasta que no interrumpamos la ejecución (pinchando sobre la celda del Notebook donde se ejecuta, y pulsando dos veces la tecla “i”), no se parará.

Esta parte levanta un API en la siguiente URL:

"http://127.0.0.1:5000"

Como se puede mostrar en la salida del notebook:

```
 * Serving Flask app '__main__'
 * Debug mode: off
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
 * Running on http://127.0.0.1:5000
Press CTRL+C to quit
```

Se ha dispuesto de la siguiente línea de código, en la parte inferior del notebook, por si no se dispone de conexión a internet o la parte del Scrapping se demora:

```
# Ejecutar antes del API Flask si se desea omitir la ejecución del modulo Scrapper
df = pd.read_csv('datos_offline/datos_obtenidos.boe.csv', sep='|')
```

Se puede ejecutar para omitir la ejecución de la parte de descarga vía Scrapper que tarda un tiempo, en función de la velocidad de conexión a internet de la que se disponga.

5. Cliente del API:

Se desarrolla en el siguiente cuaderno: "Cliente_API_BOE_Ofertas_Empleo.ipynb" un cliente, que lanza un prompt al usuario, preguntando el termino de búsqueda, después se comunica con el API den notebook anterior de la siguiente forma:

"http://127.0.0.1:5000/buscar/busqueda=<TerminoDeBusqueda>"

Con la respuesta de este API, el programa elabora un dataset en pandas que es consultado en la siguiente línea de código.

Si se decide separar ambos módulos en la siguiente línea:

```
[2]: # Definicion Endpoint del API
     DOMINIO_ENDPOINT='localhost'
     PUERTO_ENDPOINT=5000
```

Podemos seleccionar el dominio donde se ejecuta el módulo del API: Descarga BOE Ofertas Empleo y API Flask.ipynb.

6. Ejecución:

Inicialmente ejecuta el notebook:

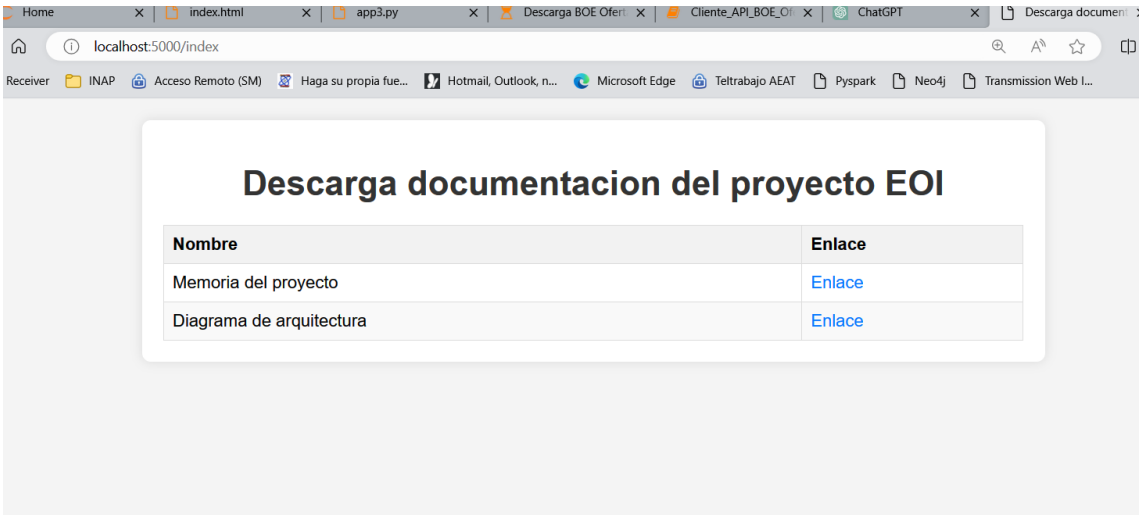
- ***Descarga BOE Ofertas Empleo y API Flask.ipynb***

Para probar que el API está sirviendo correctamente, poner la siguiente url en un navegador:

http://<url_depliegue_api>:<puerto_despliegue>/index

Por defecto:
url: localhost
puerto_despliegue: 5000

Aparecerá la siguiente página:



Una vez comprobada esta url, podemos ejecutar el siguiente notebook: "Cliente_API_BOE_Ofertas_Empleo.ipynb", en la línea del prompt introducir el termino por el que se quiere filtrar la información, como, por ejemplo:
"Alicante", "A1", "A Coruña", "Informática", etc.

7. A tener en cuenta:

El buscador es muy básico, es decir que es case sensitive, y no busca frases completas, tenedlo en cuenta para las pruebas.

8. Conclusiones:

Con los conceptos asumidos en la sección 2 del master y con un poco más de mi parte (conocimientos de Flask) se ha desarrollado un scrapper que extrae las ofertas de empleo del BOE y las pone a disposición de un API REST con la librería Flask de Python, después se ha desarrollado una parte cliente que se conecta al API, y obtiene información del mismo.