Taller Web Scraping con Python

Objetivos:

1. Introducir a los estudiantes en la técnica de Web scraping.
2. Enseñar a utilizar herramientas de Python para obtener y procesar datos.
3. Aprender y aplicar técnicas de Web Scraping para extraer datos de páginas web utilizando **BeautifulSoup**.

Requisitos previos:

- Conocimientos básicos de Python.
- Tener instalado `requests`, `BeautifulSoup4`.
- Tener un navegador **Google Chrome** y el **ChromeDriver** correspondiente.

ACTIVIDADES POR DESARROLLAR:

Teniendo en cuenta lo visto en Web Scraping realiza lo siguiente:

Parte 1: Introducción a Web Scraping

📌 Conceptos clave

1. ¿Qué es Web Scraping?

💡 Ejercicio 1: Explorar el archivo robots.txt

- Busca el archivo `robots.txt` de una página web e identifica qué partes están permitidas para el scraping.
    - Ejemplo: https://www.wikipedia.org/robots.txt
- ¿Qué indican las reglas `User-agent`, `Disallow` y `Allow`?

✍️ Pregunta reflexiva:

¿Por qué algunos sitios web bloquean el Web Scraping?
¿Cuándo es preferible usar una API en lugar de Web Scraping?
Herramientas populares para Web Scraping en Python.

Parte 2: Scraping con BeautifulSoup

📌 Conceptos clave

- ¿Qué es **HTML** y cómo se estructura una página web?
- ¿Qué es el **DOM (Document Object Model)**?
- Introducción a `requests` y `BeautifulSoup`.

💡 Ejercicio 2: Extraer títulos de noticias

1. Instala las librerías necesarias:

   pip install requests beautifulsoup4

2. Escribe un script en Python para obtener los títulos de noticias de la página principal de un sitio de noticias.

✍️ Preguntas:

1. ¿Qué significa `soup.find_all("h2")`?
2. ¿Cómo podríamos modificar el código para extraer solo los títulos que contengan una palabra clave específica?

Parte 3: Scraping de Datos en Tablas

📌 Conceptos clave

- Uso de `find_all` para extraer datos tabulares.

💡 Ejercicio 3: Extraer datos de una tabla de Wikipedia

1. Busca una página de Wikipedia con una tabla de datos.
2. Modifica el siguiente código para extraer los datos de una tabla específica.

```
url = "https://es.wikipedia.org/wiki/Lista_de_pa%C3%ADses_por_PIB_(nominal)"
response = requests.get(url)
soup = BeautifulSoup(response.text, "html.parser")
tabla = soup.find("table", {"class": "wikitable"})
filas = tabla.find_all("tr")
for fila in filas:
 columnas = fila.find_all("td")
 datos = [col.text.strip() for col in columnas]
```

```
print(datos)
```

✍️ Preguntas:

1. ¿Cómo podemos almacenar los datos en un archivo CSV?
2. ¿Cómo modificarías el código para obtener solo los 10 países con mayor PIB?

Parte 4: Buenas Prácticas y Ética en Web Scraping

📌 Conceptos clave

- Respetar `robots.txt`.
- No sobrecargar servidores con muchas solicitudes en poco tiempo.
- Usar cabeceras HTTP adecuadas (`User-Agent`).
- Preferir APIs oficiales cuando estén disponibles.

💡 Ejercicio 4: Rotación de User-Agent

Modifica tu código de **BeautifulSoup** para incluir un User-Agent diferente en cada solicitud.

✍️ Preguntas:

1. ¿Por qué es importante rotar User-Agents?
2. ¿Qué pasa si realizamos muchas solicitudes a un sitio sin control?

Parte 5: Uso de IA en Web Scraping

Conceptos clave:

- ¿Cómo puede ayudar la IA en el análisis de datos obtenidos mediante Web Scraping?
- Introducción a herramientas de IA como **GPT, Hugging Face, y Google Vision API**.

Ejercicio 5: Resumen Automático de Contenido Web

1. Extraer el texto de un artículo de noticias usando Web Scraping.
2. Enviar el texto a la API de OpenAI (GPT-4) o Google Gemini para obtener un resumen.
3. Comparar el resumen generado por IA con el contenido original.

Código ejemplo

```
import requests
from bs4 import BeautifulSoup
import openai

# Scraping de una noticia
url = "https://ejemplo.com/noticia"
response = requests.get(url)
soup = BeautifulSoup(response.text, "html.parser")
parrafos = soup.find_all("p")
texto = " ".join([p.text for p in parrafos])

# Uso de la API de OpenAI para resumir
openai.api_key = "TU_CLAVE_API"
resumen = openai.ChatCompletion.create(
  model="gpt-4",
  messages=[{"role": "user", "content": f"Resume este texto en 3 frases: {texto}"}]
)

print(resumen["choices"][0]["message"]["content"])
```

✍️ **Preguntas:**

1. ¿Qué ventajas tiene usar IA para procesar los datos extraídos?
2. ¿Cómo podríamos adaptar este ejercicio para clasificar noticias por temática?

El taller se debe entregar en formato pdf. incluya los scripts completos como anexo.

Responda las preguntas de cada sección.

Recuerde que todo debe ser cargado a través del aula virtual.

Ing. Narly Beatriz Sánchez Caviedes