RESTAURANT SCRAPER

PROJECT

1. Problem Definition

The project aims to scrape information about specific restaurants listed on Zomato. This includes:

Extracting the name, address/phone number, and rating of the restaurants.

- Aggregating the data for educational purposes and demonstrating web scraping techniques.

Key Objectives:

- Automate the process of data collection from Zomato restaurant pages.
- Learn about Python-based web scraping methods using tools like BeautifulSoup and requests.
- Handle challenges such as dynamic websites, HTTP status errors, and robust parsing.

2. Tools and Technologies Used

The following tools and libraries are used in the project:

- Python: The programming language for developing the scraper.
- Requests Library: To make HTTP GET requests to fetch HTML content of web pages.
- BeautifulSoup: For parsing HTML content and extracting data.
- Zomato: Source platform for scraping restaurant details.

3. Step-by-Step Implementation Process

Step 1: Import Libraries

python
Copy code
import requests
from bs4 import BeautifulSoup

These libraries form the backbone of the scraper:

- requests to retrieve web page content.
- BeautifulSoup to parse and traverse the HTML structure.

Step 2: Define the Scraper Function

The function scrape_zomato_restaurant takes a Zomato restaurant URL as an argument. It:

1. Sends a GET request to fetch the HTML content of the page.
2. Extracts specific details (name, phone number, and rating) using CSS classes and tag structures.

python
Copy code

```
def scrape_zomato_restaurant(restaurant_url):
  headers = {
    "User-Agent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64)"
  }
  response = requests.get(restaurant_url, headers=headers)
  soup = BeautifulSoup(response.text, "html.parser")

  # Extract restaurant data
  name_tag = soup.find("h1", class_="sc-7kepeu-0 sc-iSDuPN fwzNdh")
  name = name_tag.text.strip() if name_tag else "No Name"

  number_tag = soup.find("div", class_="sc-bFADNz euONpu")
  phone = number_tag.text.strip() if number_tag else "No Phone Number"

  rating_tag = soup.find("div", class_="sc-1q7bklc-1 cILgox")
  rating = rating_tag.text.strip() if rating_tag else "No Rating"

  return {"Name": name, "Number": phone, "Rating": rating}
```

Step 3: Define Restaurant URLs

Multiple restaurant pages are looped over for scraping.

python
Copy code
```
restaurant_urls = [
   "https://www.zomato.com/kozhikode/hotel-anhar-palayam/order",
   "https://www.zomato.com/kozhikode/topform-restaurant-kuttichira/order",
   ...
]
```

Step 4: Loop Through URLs

- For each URL, call scrape_zomato_restaurant.
- Display the returned data (or an appropriate error message).

python
Copy code
```
for url in restaurant_urls:
   data = scrape_zomato_restaurant(url)
   if data:
      print(f"Restaurant: {data['Name']}")
      print(f"Number: {data['Number']}")
      print(f"Rating: {data['Rating']}\n")
   else:
```

```
print(f"Failed to fetch data for {url}.")
```

4. Challenges Faced and Solutions

1. Challenge:

HTTP status codes like 403 Forbidden due to website restrictions.

    a. Solution: Use custom headers with a User-Agent string to mimic browser requests.
2. Challenge:

Changes in Zomato's dynamic HTML structure.

    a. Solution: Inspect the webpage for relevant CSS classes and update the parser accordingly.
3. Challenge:

Some pages might lack data for specific attributes like ratings or phone numbers.

    a. Solution: Add conditional checks to avoid errors and provide fallback values (e.g., "No Rating").
4. Challenge:

Website restrictions or CAPTCHA.

    a. Solution: Consider integrating selenium for browser simulation in advanced projects.

5. Improvements and Recommendations

- Dynamic Page Scraping: Use Selenium to handle JavaScript-based content.
- Data Normalization: Store the scraped data in a structured format (e.g., CSV or database).
- Rate-Limiting: Introduce delays between requests to prevent being flagged by the server.
- Exception Handling: Add comprehensive error-handling mechanisms for unexpected HTML structures.

6. Example Output

yaml
Copy code
Scraped Data:

_____
Restaurant: Hotel Anhar
Number: +91 9876543210
Rating: 4.3

Restaurant: Topform Restaurant
Number: +91 1234567890
Rating: 4.5

...

7. Conclusion

The project successfully demonstrates how to extract structured data (name, phone number, and ratings) from Zomato for educational purposes. By addressing challenges like dynamic page content and error handling, the scraper showcases the robust use of web scraping techniques in Python.

8.Code

https://colab.research.google.com/drive/17wEXyP9di7hcR7gYrGratiarkJfEru5t#scrollTo=wC93wWSLTju8

code and output