Portronics Webside WebScrapping

A Project Submitted to the

IT VEDANT Institute, Thane.



Data Science & Data Analytics With AI

Python-Web-Scrapping Project

 \mathbf{BY}

Prerana Vijay Rokade
Under the Guidance of
Mr. Sameer Warsolkar

Description

This guide outlines the steps required to scrape product information from the Portronics website using Python. Web scraping involves extracting data from websites and saving it in a structured format, such as a CSV file, for further analysis. This guide will walk you through the process of inspecting the website, defining the URL, creating a function to scrape the webpage, and storing the data in a tabular format. Additionally, it highlights the benefits and risks associated with web scraping.

Outline:

- 1. Choose the Website and Webpage URL
- 2. Inspect the Website
- 3. Install the Important Libraries
- 4. Write the Python Source Code
- 5. Export the Extracted Data

Steps:

Outline: Steps to Scrape www.portronics.com

1. Choose the Website and Webpage URL

Website: www.portronics.com

2. <u>Inspect the Website</u>

Analyze the HTML structure of the webpage to identify the tags and classes used for the elements you want to scrape, such as product names, prices, and descriptions.

Benefits and Risks:

Benefits:

Automation: Enables efficient and large-scale data collection.

Speed: Allows quick retrieval of updated information.

Consistency: Ensures uniform data collection processes.

Risks:

Legal Issues: Ensure compliance with the website's terms of service to avoid legal repercussions.

IP Blocking: Excessive scraping can lead to your IP address being banned.

Data Quality: Dynamic content or JavaScript-loaded data might present challenges.

Step-by-Step Guide:

1. Import Required Libraries

First, import the necessary Python libraries for web scraping and data handling:

1. Import Required Libraries: First, you need to import all the necessary libraries for the task. These include requests for making HTTP requests, BeautifulSoup for parsing HTMLcontent, csv for writing data to a CSV file, re for regular expressions, and PrettyTable for displaying data in a table format.

```
In [1]: import requests from bs4 import BeautifulSoup from tabulate import tabulate
```

2. <u>Define the URL of the Website to Scrape</u>

Specify the URL of the website you want to scrape. In this case, it's the "www.portronics.com" website.

```
In [3]: # Fetch the page content
   page = requests.get("https://www.portronics.com")
   soup = BeautifulSoup(page.content, 'html.parser')
   soun
```

3. Define the Function to Scrape the Webpage

Create a function called protronics that takes a URL as an argument, sends an HTTP GET request to that URL, and parses the HTML content to extract book information.

```
In [8]: import requests from bs4 import BeautifulSoup
               from tabulate import tabulate
               # Fetch the page content
page = requests.get("https://www.portronics.com")
soup = BeautifulSoup(page.content, 'html.parser')
               # Initialize Lists to store data
              # Initialize lists to
product_names = []
product_details = []
prices = []
regular_prices = []
quick_adds = []
               # Extract product names
products = soup.find_all(class_="card_heading h5")[3:13]
for product in products:
    product_names.append("13-in-1 Design " + product.text.strip())
               # Extract product details
details = soup.find_all(class_="card-detail")[3:13]
               for detail in details:
                     product_details.append(detail.text.strip().replace("|", ","))
               prices_data = soup.find_all(class_="money")[3:13]
for price in prices_data:
    prices.append(price.text.strip().replace("₹", ""))
               regular_prices_data = soup.find_all(class_="price-item price-item--regular")[3:13] for regular_price in regular_prices_data:
                    regular_prices.append(regular_price.text.strip().replace("₹", ""))
               quick_add data = soup.find_all(class_="quick-add mt-3 no-js-hidden")[3:13]
for quick_add in quick_add_data:
    quick_adds.append(quick_add.text.strip())
               max_length = max(len(product_names), len(product_details), len(prices), len(regular_prices), len(quick_adds))
              " Pod tiss to the same tength = len(product_names))
product_names += [''] * (max_length - len(product_details))
prices += [''] * (max_length - len(prices))
regular_prices += [''] * (max_length - len(regular_prices))
quick_adds += [''] * (max_length - len(quick_adds))
               # Arrange data in a table
table_data = []
for idx, (name, detail, price, regular_price, quick_add) in enumerate(zip(product_names, product_details, prices, regular_prices
table_data.append([idx, name, detail, price, regular_price, quick_add])
               print(tabulate(table_data[:10], headers=["Sr. No.", "Product Name", "Details", "Price", "Regular Price", "Quick Add"], tablefmt=
```

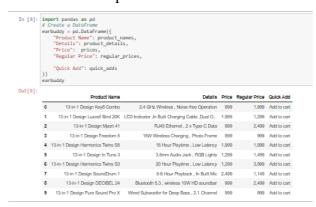
4. Scrape the Products from the First Page

Use the function to scrape product data:

```
In [4]: products = scrape_portronics(url)
```

5. Display the Data in a Tabular Format Using PrettyTable

Present the scraped data in a readable table format:



6. Save the Data to a CSV File

Store the scraped data in a CSV file for further analysis:

```
In [5]: import csv
print(f"Data has been saved to {csv_filename}")
Data has been saved to products.csv
```

Conclusion

By following this guide, you can effectively scrape product information from the Portronics website and save it in a CSV file for further use. Remember to address the potential risks associated with web scraping and ensure you comply with legal and ethical guidelines.

