Report

**Project Title: Web Scraping for Stock Prices in Python**

**Objective**: The main goal of this project is to scrape stock prices from a website and analyze the data to gain insights into stock market trends.

**Data Source**: The project uses web scraping to extract stock prices from a chosen website (for example, Yahoo Finance, GROW).

**Tools and Libraries**: The project uses

* Python libraries like requests for making HTTP requests
* BeautifulSoup for parsing HTML
* for data manipulation and analysis
* pandas
* matplot
* seaborn.

**Methodology**:

1. **Web Scraping**: Use the requests library to fetch the webpage content.
2. **HTML Parsing**: Use BeautifulSoup to parse the HTML and extract the required data (stock prices).
3. **Data Storage**: Store the extracted data in a structured format using pandas.
4. **Data Analysis**: Analyze the data using various statistical methods and visualize the trends.

**Steps**:

1. **Import Libraries**:

python

Copy

import requests

from bs4 import BeautifulSoup

import pandas as pd

**Required Modules**

In this article, we’ll look at how to work with the **Requests**, **Beautiful Soup,** and **Pandas** Python packages to consume data from websites.

* The [**Requests**](https://www.geeksforgeeks.org/python-requests-tutorial/)module allows you to integrate your Python programs with web services.
* The[**Beautiful Soup**](https://www.geeksforgeeks.org/implementing-web-scraping-python-beautiful-soup/) module is designed to make screen scraping a snap. Using Python’s interactive console and these two libraries, we’ll walk through how to assemble a web page and work with the textual information available on it.
* The [**Pandas**](https://www.geeksforgeeks.org/pandas-tutorial/)module is designed to provide high-performance data manipulation in Python. It is used for data analysis that requires lots of processing, such as **restructuring**, **cleaning** or **merging**, etc.

**Approach**

* Initially, we are going to import our required libraries.
* Then we take the URL stored in our list.
* We will feed the URL to our soup object which will then extract relevant information from the given URL based on the class id we provide it.
* Store all the data in the [Pandas Dataframe](https://www.geeksforgeeks.org/python-pandas-dataframe/) and save it to a CSV file.

1. **Fetch Webpage Content**:

python

Copy

url = 'YOUR\_TARGET\_WEBSITE'

response = requests.get(url)

soup = BeautifulSoup(response.text, 'html.parser')

1. **Extract Data**:

python

Copy

stock\_data = []

for item in soup.find\_all('YOUR\_TARGET\_HTML\_TAG'):

stock\_data.append(item.text)

df = pd.DataFrame(stock\_data, columns=['Stock Price'])

1. **Data Analysis**:

python

Copy

# Perform your analysis here

1. **Visualization**:

python

Copy

import matplotlib.pyplot as plt

df.plot()

plt.show()

**Results**: The project successfully scrapes stock prices and visualizes the data to identify trends. For example, you might find that certain stocks perform better on specific days of the week.

**Challenges**: Potential challenges include handling website structure changes, dealing with large amounts of data, and ensuring ethical web scraping practices.

**Future Work**: Future improvements could include automating the scraping process, expanding the project to scrape data from multiple sources, and applying machine learning algorithms for predictive analysis.