Scrapping Data From Real World

G.J. Rahul
27/12/2024

This project scrapes real-world data from websites using Selenium, handles captchas, and presents the data in a user-friendly format via Streamlit. Integration with BrightData ensures efficient and scalable scraping. Streamlit secrets are used to securely manage sensitive information for seamless data extraction and presentation.

1.0 Problem Statement

The goal is to scrape data from real-world websites, which includes bypassing captcha to gather valuable information.

2.0 Requirements

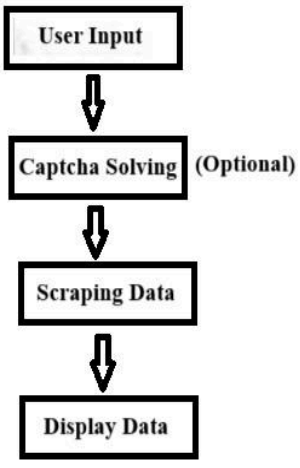
2.1 Technical Requirements

- **Python:** Programming language
- **Streamlit:** Hosting
- **Selenium:** Scrape data from website

2.2 Service Requirements

BrightData: Service that provides selenium (deployed) which could be used to scrape data from the website.

3.0 Flow Diagram



- **User Input:** The user inputs the website's address.

- **Captcha Solving:** BigData service checks if any captcha needs to be solved.
- **Scraping Data:** Selenium scraps the data from the website.
- **Display Data:** Streamlit displays the user data in an expander.

4.0 Implementation Details

4.1 Setting Up the Environment

1. Installing Streamlit

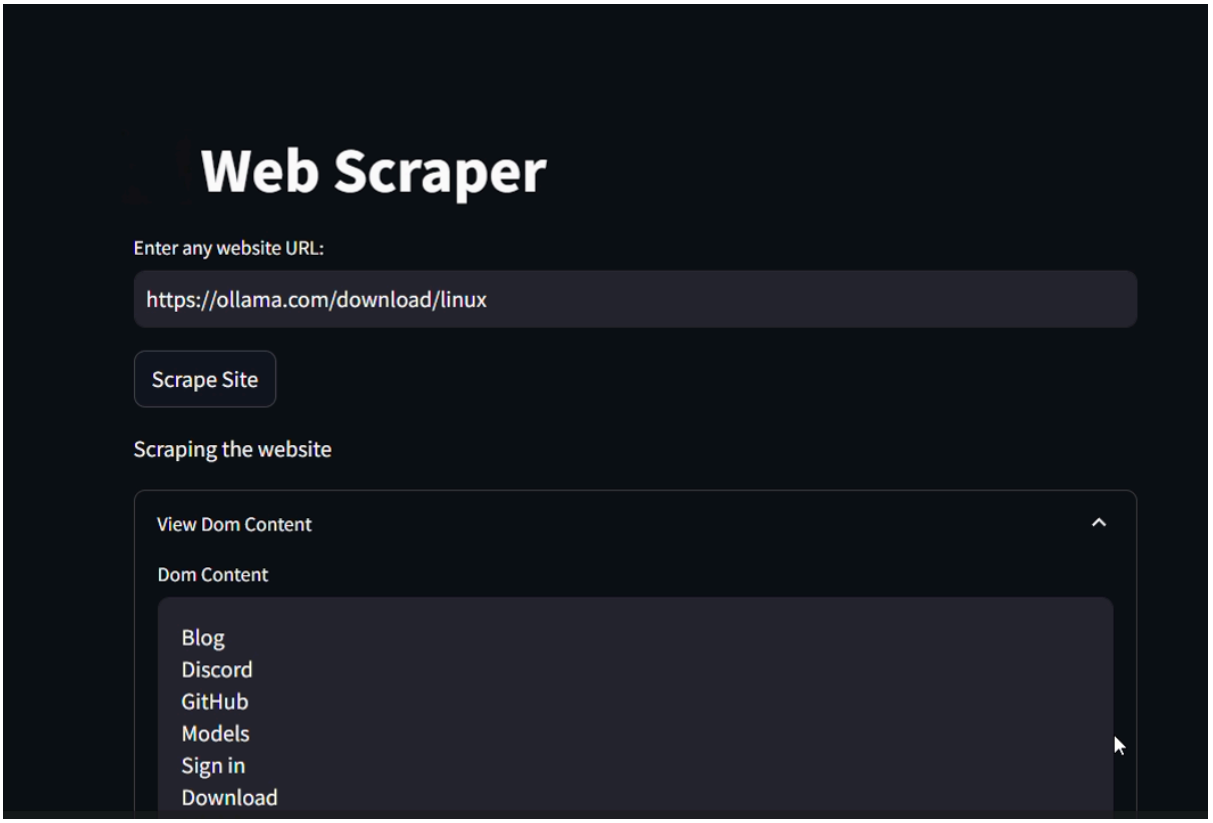
```
pip install streamlit
```

2. Create a folder named '.streamlit'
3. Create a file named secrets inside the .streamlit secrets.toml
4. The code inside secrets.toml should be as follows:

```
[proxy]
url = "bright-data-url"
```

Replace big-data-url with the url generated using big-data.

5.0 Screenshots



5.1 Presentation of Data

The image shows shows a Streamlit interface where users can enter a URL to scrape data from. After clicking the "Scrape Site" button, the website's content is displayed in a user-friendly format. For instance, the interface successfully displays the DOM content of "https://ollama.com/download/linux," showing various links and sections of the page.

6.0 Challenges Faced

Integrating BigData with Streamlit using streamlit secrets.

7.0 Project Summary

- The project uses **Streamlit** for creating a user-friendly interface and **Selenium** for scraping data from websites.
- **BigData** service is integrated to deploy Selenium instances for efficient data scraping.
- **Streamlit Secrets**: A secure way to manage and access sensitive information like API keys, ensuring that the application can safely interact with external services.

Final Notes: The project successfully addresses the challenge of scraping real-world data and handling captchas, presenting the information in an accessible format. By leveraging Streamlit and Selenium, and securely managing credentials with Streamlit secrets, the project provides a robust solution for data scraping needs.