**Documentation for Doctor Profile Scraper using Streamlit and BeautifulSoup**

**Overview**

The app.py script is a Streamlit-based web application that scrapes the number of doctor profiles from the Practo website based on a user's chosen location and medical specialization. The application uses the requests library to make HTTP requests and BeautifulSoup from the bs4 library to parse HTML content and extract the desired data.

**Requirements**

To run this script, you will need to have the following Python libraries installed:

* streamlit: Used for creating the web-based UI.
* requests: Used for sending HTTP requests.
* beautifulsoup4: Used for parsing and scraping HTML content.

You can install these libraries using the following commands:

bash

Copy code

pip install streamlit

pip install requests

pip install beautifulsoup4

**Code Explanation**

**1. Importing Libraries**

python

Copy code

import streamlit as st

import requests

from bs4 import BeautifulSoup

* streamlit is used to build the web application interface.
* requests is used to send HTTP requests to the Practo website.
* BeautifulSoup is used to parse the HTML response and extract the required information.

**2. Function: scrape\_doctors\_count**

This function scrapes the number of doctor profiles available for a given location and specialization from the Practo website.

python

Copy code

def scrape\_doctors\_count(location, specialization):

* **Parameters**:
  + location: The city or area for which to search for doctors.
  + specialization: The medical specialization (e.g., Cardiologist, Dermatologist).

**a. Constructing the URL**

python

Copy code

search\_url = (f'https://www.practo.com/search/doctors?results\_type=doctor'

f'&q=%5B%7B%22word%22%3A%22{specialization}%22%2C%22autocompleted%22%3Atrue%2C%22category%22%3A%22subspeciality%22%7D%5D'

f'&city={location}')

* A dynamic URL is constructed using the base URL of Practo's search page and incorporating the user-provided location and specialization.
* The q parameter in the URL is encoded with the search term for specialization.

**b. Setting the Headers**

python

Copy code

headers = {

'User-Agent': 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/58.0.3029.110 Safari/537.3'}

* Sets HTTP headers to mimic a browser request, which helps in avoiding request blocks by the website.

**c. Sending the HTTP GET Request**

python

Copy code

response = requests.get(search\_url, headers=headers)

* Sends an HTTP GET request to the Practo website with the constructed URL and headers.

**d. Parsing the HTML Response**

python

Copy code

if response.status\_code == 200:

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

* Checks if the request was successful (status\_code 200).
* Parses the HTML response using BeautifulSoup if the request is successful.

**e. Extracting the Doctor Count**

python

Copy code

doctors\_count = soup.find('h1', class\_='u-xx-large-font u-bold')

* Attempts to find an HTML element (h1 tag) with a specific class name that contains the count of doctor profiles.
* The doctors\_count is expected to contain the total number of doctor profiles for the selected location and specialization.

**f. Returning the Scrape Result**

python

Copy code

if doctors\_count:

return doctors\_count.text.strip()

else:

return "No doctors information found."

* Returns the extracted doctor count after stripping any whitespace, or returns a message indicating no information was found.

**3. Streamlit UI Components**

This section defines the Streamlit interface components and interactions.

**a. Application Title**

python

Copy code

st.title("Doctor Profile Scraper")

* Sets the title of the web application to "Doctor Profile Scraper".

**b. User Input for Location**

python

Copy code

location = st.text\_input("Enter Location:")

* Creates an input field where users can enter a location (city or area).

**c. Dropdown Menu for Specialization**

python

Copy code

specializations = [

'General Physician', 'Cardiologist', 'Dermatologist', 'Neurologist',

'Orthopedic', 'Pediatrician', 'Psychiatrist', 'Radiologist'

]

specialization = st.selectbox("Select Medical Specialization:", specializations)

* Provides a dropdown menu for users to select a medical specialization from a predefined list.

**d. Scrape Button and Action**

python

Copy code

if st.button("Scrape"):

result = scrape\_doctors\_count(location, specialization)

st.write(f"Result: {result}")

* Adds a "Scrape" button to the UI. When clicked, it triggers the scrape\_doctors\_count function using the provided location and specialization.
* Displays the scraping result (doctor count or an error message) on the web interface.

**How to Run the Application**

1. Save the code in a file named app.py.
2. Run the Streamlit app from the command line:

bash

Copy code

streamlit run app.py

1. Open the provided local URL in a web browser to interact with the application.

**Conclusion**

This script provides a simple and interactive way to scrape doctor profile counts from Practo using location and specialization inputs. It demonstrates the use of web scraping techniques, HTTP requests, and a user-friendly interface with Streamlit.

**Note**: Web scraping is subject to the website's robots.txt policy and terms of service. Ensure that your usage complies with Practo's policies.