**Documentation for Multiple Domains Webscraper (phone numbers, facebook links, addresses)**

**Instructions:**

You can run the script in Visual Studio Code by typing ***python app.py*** in the terminal once you opened the project folder. Please ensure you have API credentials for Algolia if you wish to use the indexing feature. The program has a **public API key** in the **app.py** code that should be replaced with your **private key** from your own Algoria project along with your **application ID** and **Index name**. Please note that you will have to replace the **public API key** as well inside the **Search.jsx** . **Or you can leave the program as it is and only modify the APY key from app.py and change it into my private key (delivered in email).** Once the python script is complete you can type **cd my-app** in the terminal and then type **npm start**, this will start the local host for the search API, you can now access the search engine by typing the addresses from the Local or On Your Network into your browser. You are now free to search any informations for the company you wish to access.  
Inside the main folder there is a scrapeTimeTrials.txt file that has all my trials for the script, how much time it took to scrap each website while changing the amout of max\_workers and the timeout value.

**Bonus question:**  
Think of a way of measuring the accuracy of your matches. The match accuracy refers to how well the provided input matches the returned entry.  
  
**Answer:**

There are multiple ways of measuring the accuracy of my matches, one of them would be the error rate, by looking at how many mistakes does the script return once I search for a company name that is similar to other companies, phone numbers, or addresses, it is a straight forward way to see if those mistakes are due to missing something important (a recall issue) or including something irrelevant (a precision issue). Another option would be MAP (Mean Average Precision), MAP would measure how accurate the guesses were and how well the information is ranked according to preferences. The last one would be precision and recall, when extracting phone numbers, Facebook links, and addresses from websites, think of Precision as ensuring your program picks up only the correct information (e.g., every extracted phone number is actually a phone number). High precision means no mix-ups. Recall is about capturing all the information out there; it ensures your program misses nothing. High recall means every relevant phone number, Facebook link, and address on those websites is found and extracted.

**Overview**

This script is designed to scrape company-related information from specified websites and their subpages. It searches for contact details, such as phone numbers and Facebook links, as well as the company's commercial, legal, and all available names. The script utilizes concurrency for efficiency and saves the aggregated data to a CSV file. Additionally, it uploads the records to an Algolia search index.

**Requirement**s

* Python 3.x
* requests: For making HTTP requests.
* BeautifulSoup4: For parsing HTML content.
* re: For regex operations.
* concurrent.futures: For concurrent execution.
* csv: For CSV file operations.
* algoliasearch: For integrating with Algolia search services.

**Installation**

To install the required packages, run the following command:

**pip install requests beautifulsoup4 algoliasearch**

**Features**

1. Fetches and processes information from multiple websites concurrently.
2. Extracts phone numbers, Facebook profile links, and addresses from web pages.
3. Removes duplicate words from addresses for cleaner data.
4. Reads company names and domains from a CSV file.
5. Writes extracted information to a CSV file.
6. Indexes records in an Algolia search index for quick searching capabilities.

**Script Functions**

* **read\_company\_info(file\_path)**

Reads additional company information from a CSV file and returns a dictionary with the domain as the key and a dictionary of company names as the value.

* deduplicate\_address(address)

Takes a string address as input and returns a string with duplicate words removed.

* **make\_request\_and\_search(domain)**

Accepts a domain and makes HTTP requests to it and its subpages. It scrapes for phone numbers, Facebook links, and addresses. Returns a dictionary with the results.

* **read\_domains\_from\_csv(file\_path)**

Reads the list of domains from a CSV file and returns them as a list.

* **main()**

Coordinates the entire scraping process, including reading input files, processing each domain, writing the results to a CSV file, and indexing records in Algolia.

**Output**

The script produces the following outputs:

A CSV file (PhoneNumber\_FacebookLink\_Addresses\_CompanyNames.csv) with the scraped information with an index in Algolia with the scraped data, if the API credentials are correctly set.

**Error Handling**

The script includes basic error handling for HTTP request exceptions. It prints out an error message and continues with the next domain if a request fails.

**Scalability**

The script uses a thread pool executor to manage concurrent tasks, which improves efficiency. Adjust the max\_workers parameter according to your system's capabilities for optimal performance.

**Security Notes**

The script sends a User-Agent string in headers to mimic a web browser, which is a common practice for web scraping.