An Overview of Web Scraping: Technical Aspects and Exercises

Gustavo Pérez Molano
Computer Science Program
Jeffrey Duffany, Ph.D.
Electrical and Computer Engineering and Computer Science
Polytechnic University of Puerto Rico

Abstract *— Researchers and organizations conducting different types of research can benefit from studying and using Web Scraping in a correct manner to further their research goals. This study serves as a review on some of the web scraping techniques and the legal and ethical implications of web scraping. Technical, legal, and ethical aspects of web scraping are discussed to better understand benefits and risks of using the web scraping process. Three exercises involving Web Scraping techniques are presented. One is performed by using the BeautifulSoup library in Python. The second exercise is performed using the web scraping tool Octoparse. Lastly, web scraping is performed using ParseHub. The three experiences are discussed to provide insight on how the different techniques and programs compare.*

Key Terms *— BeautifulSoup, Octoparse, ParseHub, Web scraping.*

INTRODUCTION

As the amount of information and data that can be found on the internet keeps growing, more challenges arise when trying to gather, study, or use large quantities of that data. The desired data might be harder to extract from complex websites or when attempting to gather from multiple sites scattered across the internet [1]. The abundance of information on the internet can be seen by just using a search engine to investigate a topic of interest. People and organizations may want to sort through vast amounts of data sources on the internet for different reasons, for example, academic research and industry research [2]. Others might have personal reasons, such as searching for job opportunities or housing offers [3]. One must also take into consideration that the many websites with data available are constantly updating and changing with new information. People doing constant research would have to periodically visit these websites to make sure they are keeping the information they have up to date.

Web scraping can be a solution to efficiently acquire data as intended in the previously mentioned examples. Many techniques and processes can be employed to carry out the process of web scraping. The correct method should be selected depending on the specific needs of the project or research task and the technical knowledge of who is doing the action.

WEB SCRAPING OVERVIEW

We can define web scraping as a technique or process by which data is extracted from the internet and organized into a format that makes it easier to analyze and visualize [1]. Because there is an intent to utilize the data and analyze it after it has been extracted, it is important to prepare accordingly before carrying out the web scraping activity. Different techniques can be used to perform web scraping as we see explained in [1]. By this definition, a web scraping technique can include whenever a person accesses a website and manually copies and pastes the information elsewhere to further analyze and study it [1].

Krotov and Tennyson use a similar yet more detailed definition of the term by stating "Web scraping is defined here as programmatic extraction, preprocessing, and organization of data from the web for further quantitative or qualitative analysis of these data [2]". This definition is precise in establishing the importance of preparation and establishing that the process is done with the intent of studying the extracted data. As mentioned in [4], the process can be carried out by web crawlers or other programs that automate the process. With preparation and the proper input from users, web scraping can be used efficiently even though it is not always fully automated [2].

Technical Aspects of Web Scraping

Web scraping as a process is usually explained as having two or sometimes three phases. The first step would be studying and analyzing the targeted source of information and the second step would be later using the selected web scraping process to retrieve the information from the source [2], [4]. For the previous method, the website or websites that will be the targets of web scraping are studied to understand how they are structured. Without analyzing how the data is stored within the website, it is harder to extract the data that is needed. This might also result in scraping more information than was intended. In [5] it is argued that there are three phases instead and are named as “Website Analysis”, “Website Crawling”, and “Data Organization”. Each of the steps should be given importance to obtain the most accurate data sets. If one of the steps is not done properly, it could affect the outcome of the next one. For example, as described in [5], the first step is to study the source of data so the way in which it is stored within the website is understood. This later affects the second step, because if the storage of data within the website is studied, the program and tools used for extracting the data can be configured accurately. Proficiency and knowledge of programming languages such as Python and R can serve to better accomplish the second step [5]. After analyzing the target website and how the data is stored, different techniques can be used to perform the second step of web crawling and extraction of the data. Copying and pasting information manually can be considered a form of web scraping [1]. Some web scrapers can use HTTP programming or HTML parsing, for example, to obtain the HTML data from the website when the web crawling step is being performed [1]. Other web scrapers, such as the ones that will be examined in the upcoming sections, can simulate web browsing, and provide an interface to script the extraction of specific data. After the web crawling step, the data is organized in a way that is easy to read for further analysis. Sometimes data must be cleaned or pre-processed, as in the case of web scraping HTML content where code is added to the web scraper program to help remove HTML headers and other elements being extracted [3]. Other web scraping programs have scripts that organize the extracted data into databases, spreadsheets, or CSV format documents when the extraction is completed.

A variety of wide-use tools exist such as:

- Octoparse
- ScrapingBot
- ParseHub
- Import.io
- Web Content Extractor
- Mozenda Web Scraper
- UiPath
- Out Wit Hub
- Screen Scraper
- WebHarvy
- Easy Web Extract

In addition to the ones mentioned in the previous list, there are also other Web Scraping Plugins, Extensions, and Web-based applications [1], [6]. However, the use of custom tools and programs made by the user could be considered a more efficient and economical solution than the more generic wide-use tools which sometimes require payment to use or customize [5].

Because there are so many programs and techniques for web scraping, completely preventing web scraping seems like a difficult task. However, there are measures and steps that can be taken to mitigate the effects of web scraping or stop some types of programs and bots that are trying to web scrape data. Most of these methods do not usually prevent a user from manually extracting the data but can block access to bots and web scraping programs. The use of CAPTCHA tests and requiring login credentials to access the website can be used to slow down web crawlers or other bots being used for web scraping, although solutions to get around these security measures can be found

[7]. Another way website owners can prevent web scraping is by using IP tracking to block users or bots from accessing the website [7]. This can be done when multiple requests to access the data are detected.

Legality and Ethics of Web Scraping

Web scraping has been a topic of discussion in legal forums in recent years, as there have been various cases where its legality is put into question [5]. People and organizations across different industries can use Web Scraping to gather information on potential clients or on competition, such as when information is gathered on the price listings [8]. The websites being accessed to gather information are usually accessible to the public. However, the activity of web scraping can be considered illegal when the data being extracted is used for illegal activity, when the data contains copyrighted or sensitive, protected information, or when it violates the terms of use for the website [5]. The person using web scraping could unwillingly incur illegal activity if they use it without making sure they can access and retrieve the data legally.

Of the laws that could apply to determining the legality of web scraping in the U.S., the Computer Fraud and Abuse Act or CFAA is mentioned substantially throughout the literature. This law is discussed and has been used to try and determine the legality of web scraping, depending on the case [5]. The CFAA, without explicitly pertaining to web scraping or providing details and elaboration on what is meant, "prohibits intentionally accessing a computer without authorization or in excess of authorization" [9]. As seen in recent rulings for cases such as Van Buren v. United States and hiQ v. LinkedIn, people are not violating CFAA when retrieving information that they are authorized to access on computers [10]. However, websites that want to discourage or prohibit web scraping might need to find other protections such as implementing authentication in order to access the information [11]. This type of measure should be considered when it will not deter regular users from accessing the website for the use intended by the website owners.

Web scraping might also present other legal issues when it is used in copyright infringement, when there is breach of contract, when it causes damage to the website, and when it is used to access confidential or protected information resulting in substantial damage [12]. Care must be taken into making sure that the information being scraped is public and not protected by copyright, otherwise, the use of the data obtained by web scraping could result in copyright infringement [5]. User generated data that is found on the website would not necessarily fall under copyright but could be protected by other privacy laws [12]. An example of this case could be user data found in social media or data generated by users writing or providing feedback on the website [5], [13]. Cases where web scraping might cause a breach of contract happen when a website has prohibited web scraping in its Terms of Use and the user has actively agreed to these terms [5]. Lastly, the website being scraped could be overloaded if the web scraping activity is being done too frequently or in moments where the website has high traffic, resulting in possible damage to the servers [14].

While the legal issues that might arise from web scraping could seem complicated to solve, the ethical issues that might come from web scraping seem to be even more debatable. This could be because ethics are an even more subjective topic than the laws that determine the legality of web scraping. Both the ethical and legal ways to conduct web scraping should be in accordance with the different cultures of people performing it and the owners of websites. A scenario where web scraping is being done so frequently that it might cause harm to the web servers was mentioned previously. This would represent an ethical issue because the user performing the web scraping should benefit from extracting the data without jeopardizing the performance of the website so that other users can benefit from it and so the owners do not have to suffer damages because of it [4].

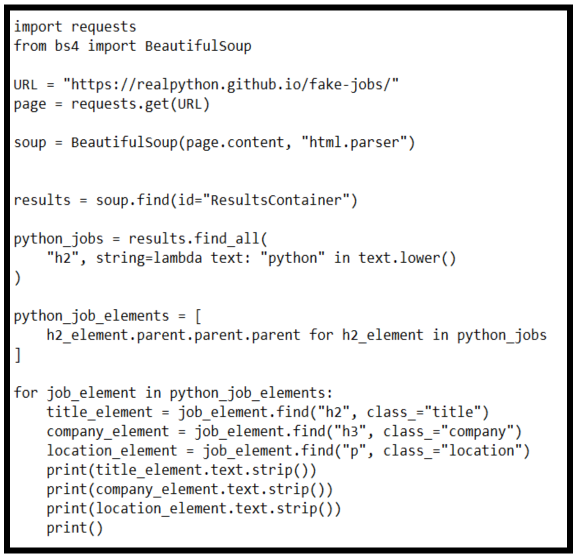
WEB SCRAPING EXERCISES

Three examples of web scraping using different techniques will be presented along with a qualitative comparison between them. The first example will involve the use of the programming language Python, specifically using the requests library and the BeautifulSoup library to scrape the desired data as presented in [3]. For the second example, a basic overview of web scraping using the program Octoparse shall be discussed. The third exercise presented would be web scraping using the program ParseHub. All three web scraping methods will be tested on the same two websites to verify and compare results between the three programs:

- https://realpython.github.io/fake-jobs/
- https://www.indeed.com/jobs?q=programmer&l=Puerto+Rico&from=searchOnHP&vjk=0df75d5af0d33ab7

Web Scraping Using Python

The Python library BeautifulSoup is commonly used for web scraping [15]. In this section, the tutorial presented in [3] was performed. The article begins by explaining the preparation steps before extracting data. It first focuses on the importance of understanding how the information is structured and stored within the website that is being targeted for scraping. The data structure is seen by using the developer tools in the browser to inspect the web page. To easily visualize how the web scraping is performed, the target website used in the tutorial is a static website with job listings that was made to be used in the example [3]. The rest of the example consists of scraping the contents of the selected website to find the job listings related to Python, which can be accessed by using the following address: https://realpython.github.io/fake-jobs/. The different steps in the tutorial show how to gradually refine the scraping technique until the output consists of the targeted data showing the jobs on the website that are related to Python. The code is used first to perform a HTTP GET request and store all the HTML data from the website into the "page" object, as can be seen in Figure 1. [3]. This is the extraction process. The data can then be cleaned and organized to present a specific portion or a targeted set of the data. The rest of the code used to clean and print the desired data can be viewed in Figure 1 [3]. A for loop is used along with the find() function to specify the targeted elements in the data that need to be extracted and printed [3].



```
import requests
from bs4 import BeautifulSoup

URL = "https://realpython.github.io/fake-jobs/"
page = requests.get(URL)

soup = BeautifulSoup(page.content, "html.parser")


results = soup.find(id="ResultsContainer")

python_jobs = results.find_all(
    "h2", string=lambda text: "python" in text.lower()
)

python_job_elements = [
    h2_element.parent.parent.parent for h2_element in python_jobs
]

for job_element in python_job_elements:
    title_element = job_element.find("h2", class_="title")
    company_element = job_element.find("h3", class_="company")
    location_element = job_element.find("p", class_="location")
    print(title_element.text.strip())
    print(company_element.text.strip())
    print(location_element.text.strip())
    print()
```

Figure 1
Python Code for Web Scraping Tutorial

The second website to be web scraped with the different web scraping methods is the Indeed website. Specifically, the webpage being scraped is that of the search results within Indeed after searching for "Programmer" jobs within the location of Puerto Rico. This would be attempted by changing the URL in Figure 1 to the one from the Indeed website.

Web Scraping using Octoparse

A second web scraping exercise was conducted using the web scraping tool Octoparse 8. While the Octoparse tool has many different features, only the features without cost were used within this exercise. Some advanced features include pre-made templates that are configured for other projects. These templates include scraping information from user comments on websites such as Youtube and user reviews within the Google Play application store. After installing and configuring Octoparse 8 for Windows, the first step consists of creating a new task within the Octoparse user interface. A separate task should be created and processed for each of the two websites being web scraped. Afterwards, the next screen asks to insert the website address or addresses that will be scraped in the exercise. After choosing the website addresses, the Octoparse tool shows the website as if being seen through a regular web browser. A script is then made by navigating through the browser and selecting what will be searched, which pages need to be viewed, which objects on the page must be selected and clicked, and what data being viewed on the page is to be extracted. These instructions are stored by the tool into a script that can be saved as a task and programmed to be performed periodically. The task can then be performed on a cloud server, or it can be done by the local machine. After extracting the data, Octoparse then provides the option of exporting the data into a document of either Excel, CSV, HTML, JSON, or XmL format. The tool also allows the data to be exported to a Google Sheets database, SqlServer database, or a MySql database.

Web Scraping using Parsehub

The web scraper ParseHub can also be useful for scraping data from various websites. The ParseHub program, like Octoparse, does not require advanced knowledge in programming to use. It is also free to download and use with limited features. Using the more advanced features requires a paid subscription to one of the various monthly plans offered on the Parsehub website. A set of tutorials on how to use the program are also included and can be accessed at any time while the scraping project is being done.

Both websites in the previous exercises were scraped again using ParseHub. The program is similar to what was described in the Octoparse method because ParseHub also has a built-in web browser where the website can be previewed. A script is also generated and is executed each time the web scraping is to be done. One difference between both programs is that ParseHub only runs the web scraping project on their cloud servers and does not offer the option to locally run the web scraping like Octoparse does.

Results and Discussion

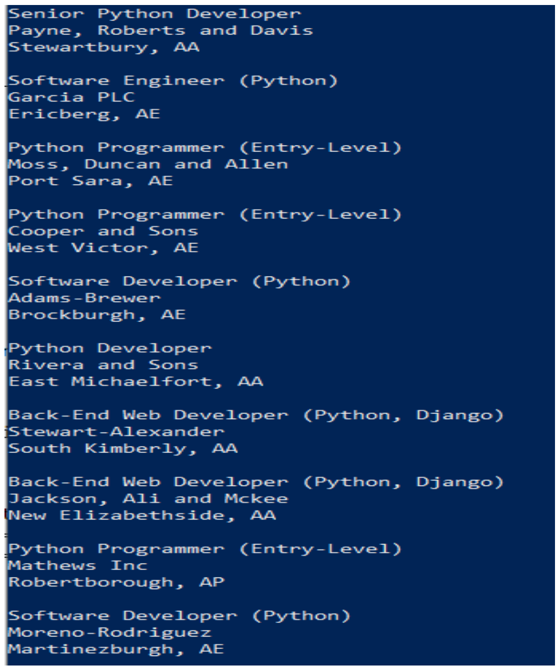
Exercises have been performed by web scraping using ParseHub, Octoparse, and by using the Python library BeautifulSoup in separate examples, first with the Python Jobs website and second with the webpage for Indeed Programmer job results in Puerto Rico. When comparing the web scraping techniques that have been presented, it should be noted that each can be used for different scenarios. A summary of the results can be viewed in Table 1.

Table 1
Job Data Web Scraped

	Python Jobs Website (Total number of job entries scraped)	**Indeed Website (Total number of job entries scraped)**
ParseHub	100	87*
Python (BeautifulSoup)	100	0
Octoparse	100	87

*Job entries scraped in Test Runs

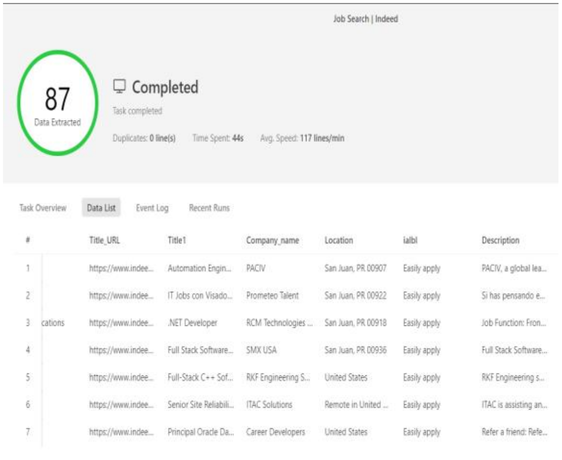
The Python web scraper found in [3] was successful in extracting all the data in the first website as presented in the tutorial. The total one hundred jobs presented on the website can be found in the HTML data extracted. In Figure 2 [3] the data obtained from web scraping is presented after selecting to only filter and present the jobs found on the website that are related to Python. The output consists of the job title, company name, and location for each of the ten jobs fitting the description and they are cleaned from HTML elements found on the original data extracted. For the first website, results from the web scraping exercise with ParseHub and Octoparse are similar with each other. All the data within the website was extracted by both programs and they are easily organized and exported to a spreadsheet document by the programs. This includes the one hundred job listings found on the site and the data related to each job. Organizing and presenting the extracted data in a simpler format is more complicated when using the Python web scraper, than with Octoparse or ParseHub as it requires more steps. The example using Python and BeautifulSoup, would be best suited for scraping simple and static HTML websites. This first method requires at least basic knowledge in programming to be completed. As the structure of the website and security measures within the website become more complicated, the level of programming knowledge should be higher to achieve web scraping using BeautifulSoup. This contrasts with using Octoparse, as there are pre-configured tasks and features that can be paid for to compensate for lack of advanced programming knowledge and skill.



```
Senior Python Developer
Payne, Roberts and Davis
Stewartbury, AA

Software Engineer (Python)
Garcia PLC
Ericberg, AE

Python Programmer (Entry-Level)
Moss, Duncan and Allen
Port Sara, AE

Python Programmer (Entry-Level)
Cooper and Sons
West Victor, AE

Software Developer (Python)
Adams-Brewer
Brockburgh, AE

Python Developer
Rivera and Sons
East Michaelfort, AA

Back-End Web Developer (Python, Django)
Stewart-Alexander
South Kimberly, AA

Back-End Web Developer (Python, Django)
Jackson, Ali and Mckee
New Elizabethside, AA

Python Programmer (Entry-Level)
Mathews Inc
Robertborough, AP

Software Developer (Python)
Moreno-Rodriguez
Martinezburgh, AE
```

Figure 2
Web Scraping Tutorial Output

The results when attempting web scraping on the Indeed website are more varied between the three methods used than those for the first website. When using Octoparse, the web scraping process was successful because the data was extracted as intended. For this example, the task was only performed once on the local computer and a total of eighty-seven job entries were extracted, as seen on the preview screen in Figure 3. The data selected to be extracted included the URL for the job entry, the job title, company name, exact location as presented in the listing, the short description provided by the employer, and the time that has passed in days since the job was posted to the Indeed website. The data on the jobs listed was later exported by Octoparse and organized into a spreadsheet without any issues.



Job Search | Indeed

87 Data Extracted

Completed
Task completed
Duplicates: 0 line(s) Time Spent: 44s Avg. Speed: 117 lines/min

#		Title_URL	Title1	Company_name	Location	ialbl	Description
1		https://www.indee...	Automation Engin...	PACIV	San Juan, PR 00907	Easily apply	PACIV, a global lea...
2		https://www.indee...	IT Jobs con Visado...	Prometeo Talent	San Juan, PR 00922	Easily apply	Si has pensando e...
3	cations	https://www.indee...	.NET Developer	RCM Technologies ...	San Juan, PR 00918	Easily apply	Job Function: Fron...
4		https://www.indee...	Full Stack Software...	SMX USA	San Juan, PR 00936	Easily apply	Full Stack Software...
5		https://www.indee...	Full-Stack C++ Sof...	RKF Engineering S...	United States	Easily apply	RKF Engineering s...
6		https://www.indee...	Senior Site Reliabili...	ITAC Solutions	Remote in United ...	Easily apply	ITAC is assisting an...
7		https://www.indee...	Principal Oracle Da...	Career Developers	United States	Easily apply	Refer a friend: Refe...

Figure 3
Octoparse 8 Data Extraction Preview

For ParseHub, the data was previewed correctly and scraped during the Test Run, like with Octoparse. However, the data was not extracted and exported correctly when performing the regular run of the program. This could be due to IP blocking and the need for IP rotation when the regular run is being performed from the ParseHub servers [7], [16]. When attempting to scrape the Indeed website using the Python program created from the tutorial in [3], the retrieved code shows the web scraping process to be unsuccessful. Among the lines of HTML code extracted from the HTTP GET request, a message can be found stating a security issue is triggered when this action is performed, and that web scraper is blocked from accessing the website and extracting the data. This result is to be expected considering that the Python web scraping tool being used is simpler in comparison to the other two programs used. The Python web scraping program used in [3] is not equipped to perform web scraping in websites with added security measures such as login, CAPTCHA, and IP blocking [7].

By using a premade tool such as Octoparse or ParseHub, users also have access to other features that might help perform web scraping more efficiently. For example, Octoparse has an Auto-detection feature and cloud services features [6]. The auto-detection feature in Octoparse can help automatically select data that will be extracted. This can be helpful on websites with a large quantity of entries of simple data points that will be retrieved. The cloud services feature can help with processing tasks that are too demanding for a personal computer because the processing is done by servers provided by Octoparse. This feature also provides the user with storage and access within remote servers to the data that is being extracted. In the case of ParseHub, the options to use IP rotation and custom proxy addresses would be beneficial features acquired from the premium plans [16].

Conclusion

Programs used for web scraping can be important tools when conducting research. This can be seen by their efficiency in retrieving and extracting data. The availability of various web scraping solutions available also demonstrates the need for understanding and investigating how they work. Users in need of collecting large quantities of data from the internet may benefit from knowing which of these programs are best suited for them or for their specific project. Because websites may have security measures put in place that could prevent some web scraping programs from working as intended [7], it is necessary to study the target website and the proper web scraping tool that will be used. From the programs investigated, it can be seen that the commercial solutions provide better results in the short term. However, customizable programs are better suited for users with more programming knowledge or for more specific projects. Future research could be done to compare other web scrapers not used in this project and see how they work. Also, further research could focus on testing different programs to see how they measure against a specific security measure that is used to mitigate or prevent web scraping.

References

[1] S.C.M.S. Sirisuriya, "A Comparative Study on Web Scraping", *Proceedings of 8th International Research Conference,* pp. 135-140, Nov. 2015.

[2] V. Krotov, M. Tennyson, "Research Note: Scraping Financial Data from the Web Using the R Language", *Journal of Emerging Technologies in Accounting*, Vol. 15, No. 1, pp.169-181, 2018, DOI: 10.2308/jeta-52063.

[3] M. Breuss, "Beautiful Soup: Build a Web Scraper With Python." Real Python. [Online] https://realpython.com/beautiful-soup-web-scraper-python/ (Accessed September 2, 2023).

[4] B. Zhao, "Web Scraping", *Encyclopedia of Big Data*, 2017, DOI: 10.1007/978-3-319-32001-4_483-1.

[5] V. Krotov, L.R. Johnson, L. Silva, "Tutorial: Legality and Ethics of Web Scraping", *Communications of the Association for Information Systems*, Vol. 47, pp.555-581, August 2020.

[6] A. Barrett, "10 Free Web Scrapers That You Cannot Miss in 2023" Octoparse. [Online] https://www.octoparse.com/blog/9-free-web-scrapers-that-you-cannot-miss (Accessed September 2, 2023).

[7] A. Barrett. "5 Anti-Scraping Techniques You May Encounter." Octoparse. [Online] https://www.octoparse.com/blog/5-anti-scraping-techniques-you-may-encounter. (Accessed October 8, 2023)

[8] J.S. Barufka, S.E. Iverson. "Web Scraping Watch: Cases Set to Clarify Application of the Computer Fraud and Abuse Act." Internet + Social Media Blog. [Onlimne] https://www.internetandtechnologylaw.com/web-scraping-cfaa/ (Accessed September 9, 2023).

[9] "Computer Fraud and Abuse Act (CFAA)." National Association of Criminal Defense Lawyers. [Online]. https://www.nacdl.org/Landing/ComputerFraudandAbuseAct (Accessed September 9, 2023).

[10] A. Crocker. "Scraping Public Websites (Still) Isn't a Crime, Cout of Appeals Declares." Electronic Frontier Foundation. https://www.eff.org/deeplinks/2022/04/scraping-public-websites-still-isnt-crime-court-appeals-declares (Accessed September 9, 2023).

[11] E. Hanson, H.M. Kim. "hiQ's Preliminary Injunction Affirmed. A Green Light for Data Scraping or Not?" White & Case LLP International Law Firm, Global Law Practice.

[Online]. https://www.whitecase.com/insight-our-thinking/hiqs-preliminary-injunction-affirmed-green-light-data-scraping-or-not (Accessed September 9, 2023).

[12] V. Krotov, L. Silva, “Legality and Ethics of Web Scraping”, *Twenty-fourth Americas Conference on Information Systems*, 2018.

[13] M. Mancosu, F. Vegetti, “What You Can Scrape and What Is Right to Scrape: A Proposal for a Tool to Collect Public Facebook Data.” *Social Media + Society*, pp. 1-11, 2020, DOI: 10.1177/2056305120940703.

[14] C. Dilmegani. “Is Web Scraping Legal? Ethical Web Scraping Guide in 2023.” AIMultiple: AI Usecases & Tools To Grow Your Business. [Online] https://research.aimultiple.com/web-scraping-ethics/ (Accessed September 27, 2023).

[15] M. Kasthuri, A. Nayyar, “Data Analysis: Web Scripting with Python”, *Open Source For You*, September 2023.

[16] S.O.Garcia. “My Test Run Works but the Actual Run Has Issues.” ParseHub. [Online] https://help.parsehub.com/hc/en-us/articles/360000113414-My-test-run-works-but-the-actual-run-has-issues (Accessed October 8, 2023)