iJRASET
International Journal For Research in Applied Science and Engineering Technology

# INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

**Volume:** 8 **Issue:** VI **Month of publication:** June 2020

**DOI:** http://doi.org/10.22214/ijraset.2020.6322

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
*ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.429*
*Volume 8 Issue VI June 2020- Available at www.ijraset.com*

# Web Information Retrieval using JShop and Python

Prashant Kumar[1], Pragati Sharma[2], Vartika Singh[3]
[1, 2, 3]*Department of Computer Science and Engineering, IMS Engineering College, Ghaziabad, Uttar Pradesh, India.*

***Abstract:*** ***We have generated a means for extracting web information present across millions of web-pages, in different formats using Jshop (A Library which can work on JSON, unlike Beautiful Soup) and Python script. Jshop is a tool based on Beautiful Soup. The proposed system extracts the unstructured data present in the scraped data and converts into a known/user's required format. The goal is to make the vast unstructured/structured data at any scale accessible efficiently as well as to streamline the process of alteration/management of the data.***
***Keywords: Data Retrieval, Web Scraping, Information Processing, Jshop, Web parsing.***

## I. INTRODUCTION

As per the international live stats reports, there are 1.75 billion websites(and counting, this number changes every second) on the world wide web. In fact, just the Google Search Index contains over 100,000,000 gigabytes of data[1]. According to a study conducted by Statista[2], There are 3.8 billion active social media users around the globe. Over 100 billion messages are sent and 1 billion posts. are shared through Facebook products every day. A vast amount of data is being created every second and on average, 11% of mobile website traffic comes from social media, which is the biggest source of unstructured data. The rapid growth of unstructured data is also increasing in an exponential pattern with time, as a result of 40% of the world population having internet connectivity today. This can be comprehended by the fact that approx. 90% of the data on the internet has been created only since 2016, according to a recent IBM marketing cloud study[3]. In 2014, there were 2.4 billion users of the internet and the number has now reached 4.4 billion as of the year 2019(Approximately 86% of which are also social media users). There has been an 83% increase in the number of people using the internet in just five years. As of 2025, the projected rate of data production per day is estimated to be 463 Billion GB per day. Most of this enormous data is unstructured and/or produced through various social media outlets, as 550 new social media users are being created each minute. Approximately 500+ million tweets are tweeted per day, over 38,000 Facebook updates per minute. Around 30 Billion messages are sent and received every day. All this 'Big Data' is in heterogeneous formats. And investigation and extraction of meaningful information and pattern are known as Big Data Analytics. Many a time, due to the unstructured format of data, although it can be accessed, the scope of work which can be done on the data is not much. There are many methods such as the various web scraping/ screen scraping(terminal emulation) tools but if the data is unstructured, or not present in the most known formats such as XML, CSV, JSON etc and is present only in human-readable formats, it is much less feasible to modify/ investigate/ parse the data or extract meaningful information and detect patterns in the data. If there can be an effective way of accessing and exploring the data and to convert it into a desired and practical format, its applications are immense in today's data-intensive world, in which the data is just expanding by each second. It can be used to parse all the websites which have no APIs for raw data access. The effective web scraping of unstructured data can be highly useful for various fields such as Social media sentiment analysis(When looked at collectively, social media activities clearly show valuable trends. While most social media websites have APIs that let 3rd parties access their data, this is not always sufficient), ecommerce pricing(To scrap and combine the prices from different sources efficiently in real-time), Investment opportunities/ Fintech(Renaissance Technologies LLC's prime hedge fund Medallion, which is based on pure technical analysis of trends in the data everywhere, is considered to be one of the most successful hedge funds ever[4]. It has averaged a 71.8% annual return, before fees, from 1994 through mid-2014 ). Therefore there is a need for effective web scrapers. It is not an independent process and involves various processes such as web crawling, navigating links etc.

### *A. What is web Scraping?, Why web Scraping?*

According to Wikipedia, Web scraping, web harvesting, or web data extraction is data scraping used for extracting data from websites[5]. It is a form of copying, in which specific data is gathered and copied from the web, usually into a central database or excel spreadsheet(CSV data etc), for later retrieval and analysis. Although, data featured on most sites can only be viewed using a web browser. [6]. They do not offer the functionality/APIs to save data copy for personal use. The only feasible(yet highly impractical) option then is to manually copy-paste the data - a very mundane job which can take many hours or sometimes days to complete.

Web Scraping is the technique of automating the process of navigating through links, and then navigating and collecting the relevant data from these relevant links. After this automation, instead of manually copying the data from websites, a well-defined Web Scraping software will replicate the same task within a fraction of the time which was being spent before. The various techniques used for web scraping are Text pattern matching, HTTP programming, Human copy-and-paste, DOM parsing, Vertical aggregation, Semantic annotation recognizing, Computer vision web-page analysis etc. Unlike the tedious and impractical process of manually extracting data, web scraping uses intelligent automation to retrieve hundreds, millions, or even billions of relevant/irrelevant data entities from the internet's seemingly endless frontier. Web scraping consists of two parts: a web crawler and a web scraper. The crawler leads the scraper, through the internet, where it extracts the data requested. We can access and collect data through browser page by page but this limits our scope greatly.

Web crawlers are excellent tools to navigate through links fastly and efficiently and then web scrappers gather and process the data which is found in the useful link.in this way, thousand or millions of pages can be visited and explored at once, and more importantly, crucial data and trends can be recognized in realtime. Even the parts of the web, which will be probably overlooked by search engines can be explored through web scrapers. If programmed accordingly, A web scrapper can also collect data by entering various search queries in the search bars of the concerned sites and not just what is present on their content page. A convenient stream of well-formatted data can also be provided by APIs from a server to another (for example, there are APIs for collecting data from wiki pages) however for the most part of the world wide web, such APIs do not exist. And even in the cases in which APIs exist, it is not warranted that those APIs would be sufficient for our purpose. This creates a need to create well-defined web scrapers, which can navigate through different types of sites and data formats and can gather data in the required format. With the help of a Python Script, we can work with all the data present over the world wide web.

### *B. Creating Crawlers And Scrapers*

The initial step involves inspecting the data source, deciphering the information in URLs, inspecting the site using developer tools, scraping HTML content from the pages etc. Python's request library can be used for this task. An HTML formatter can be used to 'clean' the extracted HTML code. Some pages contain information that's hidden behind a login, there are some advanced techniques that can be used with the requests to access the content behind logins. With a dynamic website, no HTML might be sent by the server at all[7].. Instead, a JavaScript code may be received as a response. The only way to go from the JavaScript code to the content we are interested in is to execute the code, just like the browser does. After successfully scraping the HTML content from the webpage, the HTML Code can be parsed with the BeautifulJShop library. Relevant information can be gathered through parsing.

## II. RELATED WORK AND COMPARISON

Beautiful Soup is a Python library/package for parsing HTML, XML docs(including pages having malformed markup, i.e. non-closed tags, it is named after tag soup). BeautifulSoup generates a parse tree for parsed HTML/XML pages which is useful for web scraping[8]. BeautifulSoup is accessible for Python version 2.7 and 3.

### *A. How Beautiful soup Works*

*1)* The library works mostly on the HTML and XML as a string.
*2)* String has a property of immutability.
*3)* You need to get to there on the basis of indexes.
*4)* When we do an edit on some index, a new string is created and HTML and XML contents, being longer and lengthy, make it difficult.
*5)* It works on regular expressions which is "finding by string combination", not by data and property.

### *B. Meanwhile, The Project i.e Jshop Is A High-Level Parser*

*1)* Jshop is mutable, i.e if the HTML content changes of a website, we are changing slight information into the parser.
*2)* Jshop allows you to do several operations.
*3)* It treats HTML and XML as mutable objects, with properties.
*4)* You can modify any element as per you are desiring.
*5)* The need is limitless, as you can edit an HTML content entirely, to produce something else from it.

*ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.429*
*Volume 8 Issue VI June 2020- Available at www.ijraset.com*

## III. PROPOSED METHODOLOGY AND IMPLEMENTATION

The proposed work is concerned with the parsing and analyzing of web pages (HTML code). We have developed a library JShop, which can be viewed as an enhanced/incremental version of BeautifulSoup library of Python. In this method, the web link transforms into visual blocks, which are actually segments of the webpage.
Progression of experimental work is shown below:-

*A.* Installation of JShop and Python
*B.* Python scripting
*C.* Execution of script(Python)
*D.* Construction of the content structure in desired format.

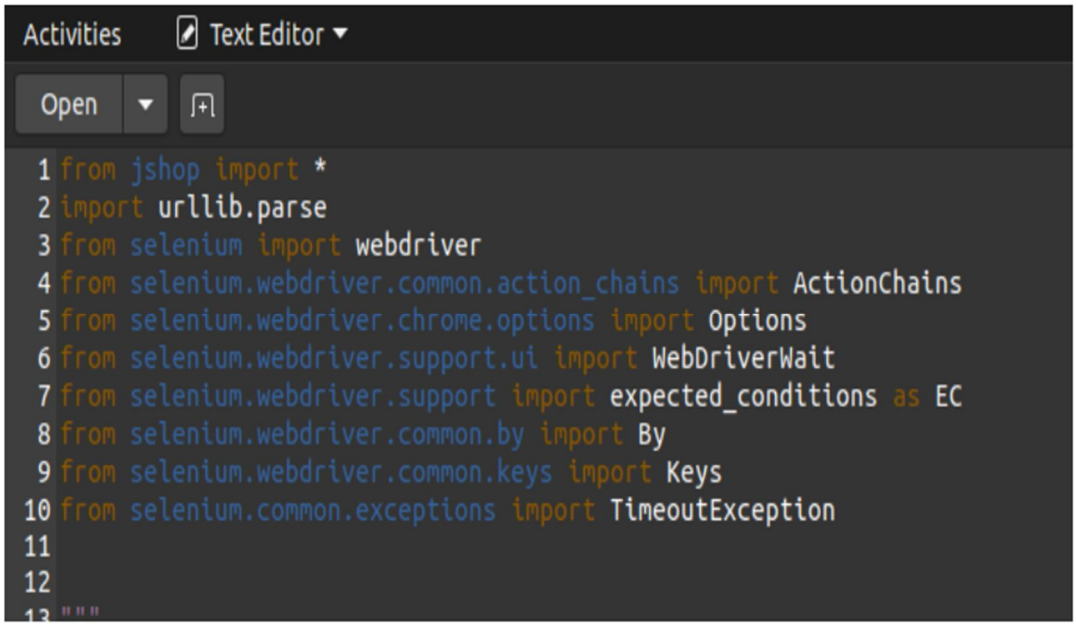



Figure 1: Initialization of Python

```
197
198
199 for i in range(1,11):
200     browser.get("https://www.lybrate.com/veterinarian?page="+str(i))
201     pages.append(browser.page_source)
202 for i in range(10):
203     fillfull(i)
204
205 pages=[]
206 import json,time
207
208 for i in range(len(full)):
209     browser.get(full[i]["link"])
210     time.sleep(2)
211     pages.append(browser.page_source)
212     attachmore(i)
213     print(json.dumps(full[i],indent=3))
```

Figure 2: URL Extraction Method

```
207
208 for i in range(len(full)):
209     browser.get(full[i]["link"])
210     time.sleep(2)
211     pages.append(browser.page_source)
212     attachmore(i)
213     print(json.dumps(full[i],indent=3))
214     x=[]
215     y=[]
216     for c in full[i].keys():
217         x.append("`"+c+"`")
218         y.append(str(full[i][c]).replace('\u20b9',"Rs. "))
219
220     x=" , ".join(x)
221     z=",".join(["%s"]*len(y))
222     sql = "INSERT INTO vet ("+x+") VALUES ("+z+")"
223     mycursor.execute(sql,y)
224     mydb.commit()
```

Figure 3: Execution of Retrieval Process

```
180 import mysql.connector
181 mydb = mysql.connector.connect(
182   host="localhost",
183   user="root",
184   database="vet"
185 )
186 mycursor = mydb.cursor()
187 try:
188     mycursor.execute("drop table vet")
189 except:
190     print("running the very first time")
191
192
193 p='rating TEXT, experience TEXT, fees TEXT, name TEXT, qualification TEXT, clinic_
    TEXT, link TEXT, additional_information TEXT, About_doctor TEXT, text_message_fee
194 mycursor.execute("""CREATE TABLE vet("""+p+""" , Speciality  TEXT, Education  TEXT
    `Professional Memberships`  TEXT,`Awards and Recognitions` TEXT)""")
195 mycursor.execute("ALTER TABLE vet ADD COLUMN id INT AUTO_INCREMENT PRIMARY KEY")
```

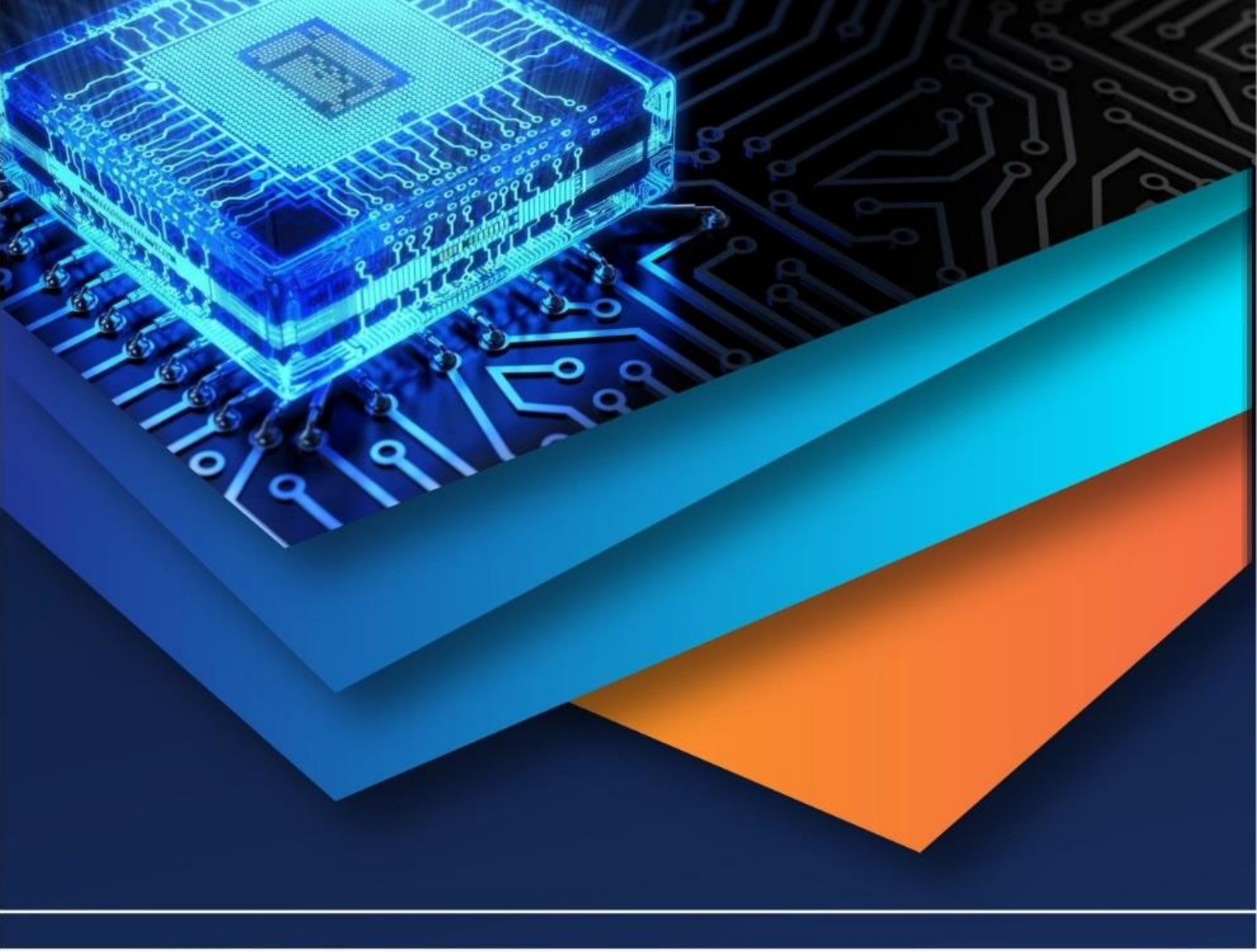
Figure 4: Data collected from the URL

## IV. CONCLUSION

Due to frequently changing and accruing nature of web information, Web information retrieval is one of the most challenging and significant areas of research. There is a need for efficient tools to retrieve and analyze the Big Data generated through the internet, social media etc and parse web-pages. This vast amount of Web info is mostly in unstructured/ill-structured format. Till now the state of the art technique of parsing the HTML content is treating it as a string, then applying regular expressions over that. In the proposed design, this HTML data is synthesized as a parsable object and creates a direct link to every node, making parsing, manipulation etc extremely fast and efficient.

## REFERENCES

[1] https://www.internetlivestats.com/
[2] https://hostingfacts.com/internet-facts-stats/
[3] https://blog.microfocus.com/how-much-data-is-created-on-the-internet-each-day/
[4] https://en.wikipedia.org/wiki/Renaissance_Technologies#:~:text=The%20Medallion%20fund%20is%20considered,past%20employees%20and%20their%20families.
[5] https://en.wikipedia.org/wiki/Web_scraping
[6] https://scrapinghub.com/what-is-web-scraping
[7] https://realpython.com/beautiful-soup-web-scraper-python/
[8] https://en.wikipedia.org/wiki/Beautiful_Soup_(HTML_parser)



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

# INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY