# Web scraping for Competitive Pricing Intelligence

Hardik Prabhu, James Joseph, K K Koushik, Karthikeyan J, Mrs. Deeksha M

*Department of Computer Science and Engineering, Alva's Institute of Engineering and Technology, Moodbidri*

E-mail: hardikprabhu123@gmail.com, jamesjoseph20107@gmail.com, kkoushik163@gmail.com, jskarthik5@gmail.com, deeksha_m@aiet.org.in

## Abstract

There is a wealth of human-established facts and data sources available on the internet. But because it will consist of a huge variety of disorganized and divergent data, it will be challenging to collect physically and to use in mechanical activities. Recently, a range of devices and procedures have been developed to help B2C and B2B systems gather data and organize it into information. Several web scraping issues will be covered in this essay, beginning with an introduction and brief summary of the numerous tools and resources that are available. Before concluding with an overview of the web scraping process, we also talked about the many types of web scraping tactics and their advantages and disadvantages.

## Keywords

Web Scrapping, Internet, Big Data, Business Intelligence.

## 1. Introduction

Web scraping is the process of automatically extracting web data rather than manually duplicating it; it is also referred to as screen scraping, web data extraction, web harvesting, etc. This method involves taking useful information out of a website's HTML and putting it into a central spreadsheet or local database. For this, it makes use of the website's URL. can be simply arranged so that it can be used on any website. A Web scraper's primary objective is to convert unstructured material while preserving it in databases that are ordered. HTML parsers, DOM parsing, and HTTP programming are a few techniques used in Web scraping. Afterwards, the created data is used for analysis or retrieval. It has many benefits, like giving us error-free data, saving time for really fast outcomes, and centrally storing all of the data. Additionally, we get to pick the format in which it is made available to us. This facilitates access and eases the process of data analysis. Website price comparison, weather data monitoring, website change detection, web mashup, web research, and web data integration are just a few of the areas where web scraping is being used. It should be mentioned that online scraping may occur concurrently with the periods of inactivity of certain websites.

## 2. Literature Survey

### 2.1 Web Scraping for Data Analytics: A BeautifulSoup Implementation

This paper explores the design and implementation of a web scraper that extracts data from the Amazon website using Python's BeautifulSoup package. The main objective was gathering essential product information for analysis, including the name, price, ratings, reviews, and links.

The implementation showed that five pages could be scraped effectively in only 10 seconds, highlighting the scraper's speedy data extraction and visualization capabilities. But during the execution stage, some shortcomings became apparent. Because the scraper's effectiveness depended on particular product names, it was not as flexible for general searches. Additionally, the scraper compromised on the range of information acquired from each product by prioritizing efficiency over complete data extraction due to speed considerations. Notwithstanding these limitations, the study creates opportunities for further investigation by extending an invitation to researchers and developers to adapt this implementation to their particular data extraction demands, analytical requirements, and various web scraping projects. This implementation is an invaluable resource, especially for researchers pursuing small-scale data analytics projects and novices in the web scraping space. It is useful not only for demonstrating a productive procedure for scraping and visualizing data, but also for providing a basic framework for customization and additional research in the broad field of online scraping and data analysis.

## 2.2 Web Scraping: State-of-the-Art and Areas of Application

Web scraping is a critical method for gathering important information from many websites and condensing it into formats that are simple to access, such as databases, spreadsheets, or CSV files. But carrying it out by hand is frequently difficult, time-consuming, and resource-intensive. The goal of this article is to review and investigate the field of web scraping techniques, covering a range of automated solutions created in earlier research. It explores the many methods and instruments used in web scraping while illuminating the wide range of industries in which it finds use. Web scraping is a sophisticated process that involves more than just extracting data from websites; it also involves navigating anti-scraping techniques, dealing with different site structures, and processing and storing the gathered data in an effective manner. This essay aims to provide insights into solving these problems with automated solutions by reexamining current methods and classifications. It also explores the various fields in which web scraping is useful, including e-commerce, market research, sentiment analysis, and more, demonstrating its applicability and importance in the data-driven world of today.

## 2.3 Design and Implementation of Data Acquisition System Based on Scrapy Technology

The development of a reliable data collection system built around the Scrapy crawler framework is described in this paper. The design of the system prioritizes the capacity to collect data according to user-specified needs and provides streamlined job management for data collection. Utilizing the Model-Template-View (MTV) architecture of Django, the development process incorporates Scrapy as the primary asynchronous Python crawler application framework. Regular expressions along with XPath are used to extract data from web pages and perform efficient analysis. Apart from its fundamental features, the system incorporates supplementary tools to improve job delegation. The addition of the Tree jQuery tree plug-in improves the system's organizing capabilities by enabling

user-friendly tree-based job management. Additionally, using Bootstrap makes the UI more user-friendly by enabling effective task name and keyword combination queries that improve the user experience when navigating pages and retrieving data. Utilizing these technologies together highlights the system's dedication to provide a feature-rich and user-centric environment for data administration and collecting.

## 3. System analysis and design

The web scraping project's System Analysis and Design employs a thorough methodology to guarantee the smooth and productive operation of every part. The "Product Retriever" is in charge of getting product details out of the database and creating a fundamental connection to the data that already exists. After that, the "Product Fetcher" takes over, searching websites and extracting URLs for the things it wants to buy.

The "Scheduler," a vital part that coordinates the entire operation, receives these URLs in a methodical manner. The "Scheduler" coordinates with the "Price Extraction" component and schedules the extraction jobs while controlling the information flow. The latter methodically retrieves product prices from the captured URLs by using a "Extraction Template". The scheduler makes sure that jobs are completed on time, allowing the entire system to run in a continuous cycle. In order to accomplish the project's goals, the System Analysis and Design process include defining each component's capabilities, creating data flow and dependencies, and making sure the components integrate seamlessly. For a reliable and flexible online scraping system, consideration should also be paid to error management, scalability, and adaptability to various websites and databases.

### 3a. Flow Diagram

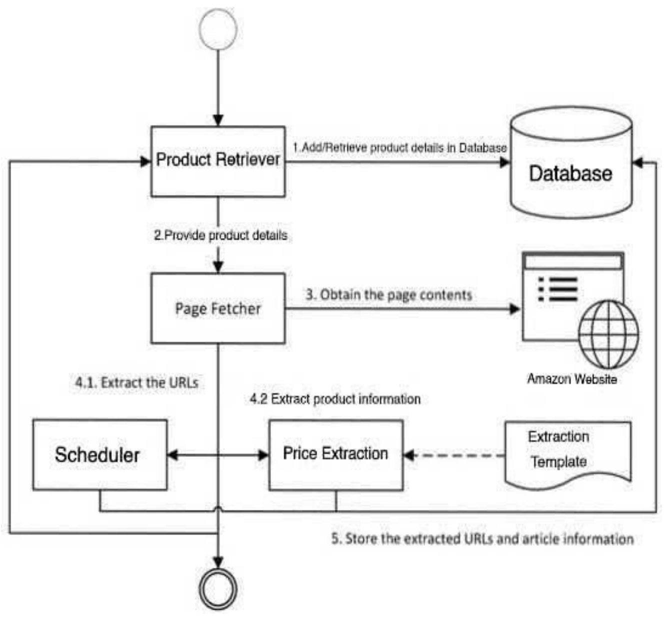
Product Retriever
1.Add/Retrieve product details in Database
Database
2.Provide product details
Page Fetcher
3. Obtain the page contents
Amazon Website
4.1. Extract the URLs
4.2 Extract product information
Scheduler
Price Extraction
Extraction Template
5. Store the extracted URLs and article information

**Figure 1. Flow Diagram.**

Here's a breakdown of the steps, incorporating insights from the diagram:

Product Retriever:

The Product Retriever is the main component that facilitates access to and retrieval of product-related information kept in the project's database. This essential part is essential to bridging the information gap between the stored data and the running operations of the web scraping framework. The Product Retriever, a crucial link in the data retrieval chain, guarantees that later components have access to the most recent and pertinent product details by effectively managing data extraction from the database.

Page Fetcher:

The important job of getting content straight from the intended web pages is handled by the Page Fetcher component. It retrieves HTML text for additional processing and serves as a gateway to external sources. This part, which focuses on efficient content retrieval, makes sure that the information used in the latter phases of the web scraping process is correct and current. Through managing the complexities of obtaining content from several sources, the Page Fetcher enhances the project's capacity to adjust and retrieve important information from a range of websites.

Price Extraction:

With a special emphasis on pricing information, the Price Extraction component is made to extract precise details from the collected page content. This module finds and isolates pertinent data points by navigating through the HTML structure using extraction templates or specified criteria. The Price Extraction component contributes to the project's capacity to deliver accurate and significant information by following established extraction criteria, which guarantees the correct retrieval of product pricing and related details.

Scheduler:

The Scheduler component runs in a loop or at predetermined intervals to orchestrate the web scraping activity. Its main duty is to oversee the scheduling and performance of different system tasks. This component maximizes the overall effectiveness of data extraction, retrieval, and storage by automating the scheduling process.

Database:

This part acts as the main archive for all the information that is collected while scraping websites. This contains details on the products, their costs, and any other pertinent information. Effective data management is made possible by the Database component's organized data store and retrieval mechanisms. It is essential to making sure that the data gathered is arranged, easily accessed, and prepared for analysis, which enhances the web scraping project's overall efficacy.

**3b. Component Diagram**

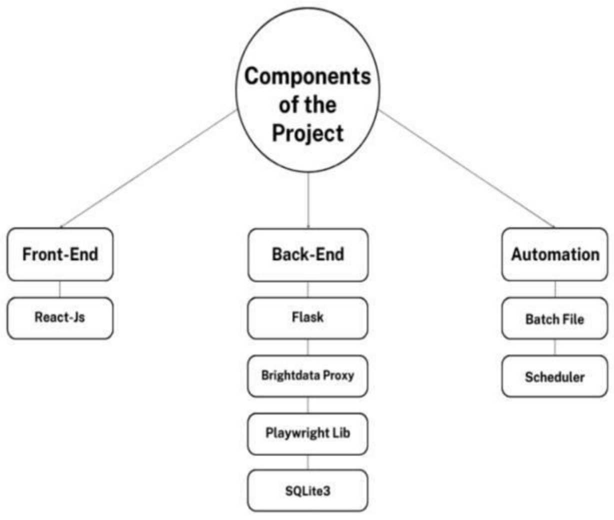

**Figure 2. Data Flow Diagram**

Here's a breakdown of the steps, incorporating insights from the diagram:

1. Front-End with React-Js:

The building of an incredibly responsive and user-friendly interface is made possible using React-Js. Because of its component-based architecture, components can be added and removed, improving consistency and maintenance ease. Interaction: Using React-Js guarantees smooth interactions and gives users an easy-to-use interface to interact with the features of the application. Its virtual DOM approach enhances rendering and makes for a more comfortable user interface.

2. Flask Back-End:

Lightweight Communication: Flask provides a strong foundation for backend programming and is renowned for its minimalist style. Its usefulness is unaffected by its simplicity, allowing for effective front-end and database connectivity. Data Management: The backend process is streamlined by Flask's adaptability in managing data operations and database connections. By supporting several databases, it enhances data retrieval and management.

3. Python Web Scraper with Playwright:

Extensive Data Extraction: Playwright and Python work well together to provide a powerful scraping combination. Playwright's scripting power is enhanced by its ability to handle intricate interactions and anti-scraping methods, allowing for comprehensive and automated data extraction. Efficiency and Customization: Customized scraping scripts that meet specific data requirements may be created thanks to the Python-Playwright connection, which maximizes speed. Its asynchronous design makes it more effective at managing several scraping jobs at once.

4. Bright Data's Integration:

Resilience Against Anti-Scraping Measures: The system's resistance to anti-scraping techniques used by websites is strengthened by Bright Data's integration of scraping browsers. Its strong infrastructure makes it possible to get around problems and guarantee continuous data collection without sacrificing accuracy or legality.

5. SQLite:

Effective Database Operations: The lightweight database engine SQLite provides effective data saving and retrieval. Its self-contained design and ease of use make it perfect for smaller-scale applications, offering dependable database operations.

6. Automation and Scheduling Regular Updates:

Automating periodic scraping with no human involvement is achieved by using a Windows Batch File. By enabling timely changes, this scheduling function preserves the correctness and relevance of the extracted data.

7. Mechanisms for Handling Errors:

Improved Dependability Ensuring system reliability requires robust error handling techniques. They enable the detection, recording, and correction of faults, guaranteeing that the system keeps running smoothly even in the face of sporadic glitches or interruptions.

**3C. Use Case Diagram**

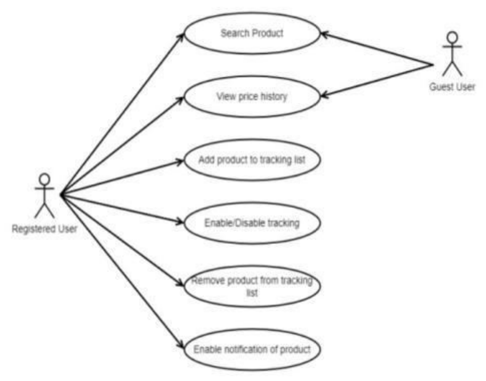
Search Product
View price history
Guest User
Add product to tracking list
Registered User
Enable/Disable tracking
Remove product from tracking list
Enable notification of product

**Figure 3. Class Diagram**

Here's a breakdown of the steps involved:

Absolutely, here's an elaboration on each component in the context of the use case diagram

1. Search Products:

This feature helps customers find and search for particular products inside the system.
• Interaction: To start searches, users enter product names, categories, or pertinent keywords.
- System functionality: Based on user queries, matching products are retrieved using search algorithms.
• Presents search results along with pertinent product details.

2. View Price History:

This feature's goal is to give consumers access to and a chance to go over past pricing information for the products they're tracking.
• Interaction: To display a product's pricing history,

users choose it from their tracked list. The system's functionality includes retrieving and displaying a detailed record or visual representation of price variations over time. gives users access to historical pricing trends so they may make well-informed decisions.

3. Include Item in Tracking List:

• Goal: Enables consumers to add particular products to their tracking list for continuous observation.
• Interaction: After searching or looking at pricing history, users can choose to add products to their tracking list.
• System functionality: • Adds the chosen products for ongoing monitoring to the user's tracking list.
• Starts the tracking procedure for the recently added goods.

4. Turn on or off tracking:

• Goal: Gives users the ability to manage each product's tracking status.
• User interaction: Users can change the tracking status of products in their list by toggling it.
• System functionality: Adapts the monitoring parameters of the system to the preferences of the user.
• Cease or restarts the ongoing monitoring of particular products.

5. Take Items Off of the Tracking List:

• Goal: Gives users the option to take particular products off of their tracking list.

• Interaction: Using the system's interface, users can start the process of removing products from their tracking list.
• System functionality: Updates the user's tracking list by eliminating the products they have chosen, so ending the tracking.

6. Permit Product Notifications:

• Goal: Provides users with the choice to turn on alerts for modifications to tracked product information.
• Interaction: Users choose to receive alerts for particular products that they find interesting.
• System functionality: Sets up the system to produce and deliver alerts or notifications in response to modifications in the data or status of the tracked product.
By offering features that let consumers manage their tracked products, examine pertinent information, and manage tracking, each use case enhances the user experience.

## 4. Methodology

Our web scraping project's suggested system methodology takes a thorough and all-encompassing approach to addressing the complexities and difficulties involved in extracting data from internet sources. This methodology's fundamental component is the combination of state-of-the-art technologies and strategic considerations meant to maximize data extraction while navigating a number of obstacles that are frequently encountered in web scraping endeavors. This methodology relies heavily on Bright Data's integration, which provides a powerful scraping browser that can bypass IP rotation, CAPTCHAs, and access limitations, hence reducing the impact of anti-scraping efforts. The foundation for a more seamless and continuous data extraction procedure is this integration. A key topic this methodology addresses is handling dynamic content, which is made possible by Playwright's integration. The system acquires a strong tool to efficiently extract data from webpages that largely rely on JavaScript by utilizing Playwright's capabilities. This capacity outperforms traditional methods such as Beautiful Soup, guaranteeing thorough data extraction from dynamic online content and so expanding the range of information retrieved. Moreover, the approach places a strong emphasis on scalability and robust automation by strategically combining React, Flask, and Playwright. Scalability for large-scale and real-time data extraction is ensured by the combination of Playwright's sophisticated scraping functionalities with React's frontend capabilities and Flask's lightweight yet effective backend architecture. This strong technology foundation ensures continued effectiveness and performance by enabling the system to easily manage growing loads and adjust to changing web architectures. This methodology promotes effective error handling methods during the scraping process in addition to technological prowess. The project prioritizes error management in order to strengthen the system's dependability. By taking great care, this methodical approach reduces interruptions and irregularities, guaranteeing a reliable and consistent data collection procedure. Apart from the technological benefits, the suggested approach suggests possible directions for corporate growth. When properly utilized, the retrieved data has the potential to offer enterprises strategic value. It provides useful information that may influence market strategies and decision-making procedures, enhancing the project's total effect and usefulness beyond its technical limits. This all-encompassing system technique essentially combines cutting-edge technological elements with strategic considerations, laying the groundwork for a reliable, effective, and scalable web scraping strategy. In addition to guaranteeing thorough data extraction, it puts the project in a position to possibly impact and improve strategic decision-making processes across a range of corporate disciplines.

## 5. Application

The suggested web scraping project's applications cut across multiple businesses and disciplines, taking use of its sophisticated approach to extract rich data and deliver useful insights. A wide range of applications with significant effects are made possible by the strong technological foundation and tactical approach.

1. E-Commerce and Retail:

The project's capacity to pull historical patterns and real-time pricing information from online marketplaces such as Amazon, eBay, and others is extremely useful in the e-commerce space. Businesses can use this information to make well-informed decisions in a dynamic market by using it to inform pricing strategies, competitor analysis, and inventory management.

2. Market Research and Analysis:

Market research benefits greatly from the extensive data extraction capabilities, particularly when it comes to dynamically loaded web sites. A deeper understanding of customer preferences is made possible by compiling market trends, product reviews, and consumer sentiment from a variety of sources. This allows for well-informed product creation and market strategy.

3. Finance and investing:

Observing market movements can be greatly aided by real-time data extraction and price tracking from financial websites, such as stock exchanges and investing platforms. Financial analysts, traders, and investors can use this information to make data-driven investment decisions.

4. Medicines and Health Care:

Tracking pharmaceutical product availability, costs, and regulatory updates from different healthcare websites can be made easier with the help of web scraping. Keeping abreast of the most recent advancements in medicine, inventory management, and pricing analysis can all benefit from this data.

5. Property & Real Estate:

Real estate professionals can gain information for pricing strategies, market analysis, and spotting new trends in particular areas by extracting property listings, price trends, and market conditions from real estate websites.

6. Competitive Analysis and Brand Observation:

Keeping an eye on rivals' pricing, new product releases, and marketing tactics on various web channels gives companies a competitive advantage. This information helps in understanding market positioning, fine-tuning pricing methods, and improving marketing techniques.

7. Strategic Planning and Business Expansion:

When the retrieved data is properly analyzed, it can provide insightful information on new market identification, market entry tactics, and business expansion. For strategic planning and well-informed decision-making processes, these insights are essential.

## 6. Results

The The expected results of applying the suggested system methodology to our web scraping project include a range of noteworthy breakthroughs and successes in the areas of technology, operations, and strategy.

1. Increased Data Extraction Efficiency:

Playwright's dynamic content management in conjunction with Bright Data's scraping browser should greatly increase data extraction efficiency. By combining many websites, this combination guarantees a more efficient and thorough retrieval procedure while getting beyond obstacles like dynamic content, anti-scraping policies, and JavaScript dependencies.

2. Enhanced Error Handling and Reliability:

It is expected that the methodology's focus on effective error handling techniques will improve the system's reliability. The project seeks to provide a more dependable and consistent data collecting procedure by reducing interruptions and irregularities during the scraping process.

3. Scalability for Real-time Data Retrieval:

It is projected that combining React, Flask, and Playwright will give the system the capacity to scale for large-scale, real-time data extraction. This technology stack enables the system to function reliably even under increased traffic, effectively manage growing data loads, and adjust to changing web topologies.

4. Streamlined Automation and Task Scheduling:

It is anticipated that the process will be more efficient with the inclusion of automation for scheduling scraping tasks through Windows Batch File. The workflow is optimized by this automated feature, which guarantees periodic and timely data extraction without the need for human participation.

5. Strategic Business Value:

The project's ability to yield strategic insights from the gathered data is extremely valuable, even if its technical capabilities are limited. When used wisely, these insights may facilitate market research, strategic planning, and well-informed decision-making, hence creating opportunities for company expansion and competitiveness.

6. Flexibility and Prospects for the Future:

The approach of the project places a strong emphasis on flexibility to various websites and data sources. Its ability to adapt and alter with web structures makes it a flexible tool for changing data extraction requirements in a range of sectors and domains. To summarize, the project's goals are to improve data extraction effectiveness, consistency, expandability, and strategic significance. These anticipated advancements not only elevate the project's technical capabilities but also position it as a valuable asset for businesses seeking data-driven insights and informed decision-making strategies.

## 7. Conclusion

Online scraping is a well-known term that has gained more attention due to the need for "free" data that is gathered from online pages or PDF documents. The data is needed by many professionals and researchers in order to process, analyze, and extract important implications. On the other hand, individuals working on business-to-business use cases need access to data from multiple sources in order to incorporate it into creative applications that provide unique features and additional benefits. We have examined the many facets of Web Scrapper in this paper. After examining the web scraping tools and software, we looked at the system's workings, advantages, and disadvantages before seeing its applications.

## References

[1]. Osmar Castrillo-Fernández, “Web Scraping: Applications and Tools”, European Public Sector Information Platform Topic Report No. 2015 / 10, December 2015.

[2]. Kanehisa M, Goto S, Sato Y, et al. KEGG for integration and interpretation of large-scale molecular data sets. Nucleic Acids Res 2012;40:D109–14.

[3]. http://adamsoft.sourceforge.net/.

[4]. https://nutch.apache.org/.

[5]. https://lucene.apache.org/solr/.

[6] Zhang Boheng, Liu Jian, Zhu Yuxiang.(2017). Microblog User Behavior Analysis System Based on Big Data and Machine Learning. Computer Knowledge and Technology, 2017, 13 (6):212-213.

[7]. William Marble, “Web Scraping With R”, stanford.edu, August 11, 2016.