Task 4

1-The difference between power query and SQL:

|  |  |  |
| --- | --- | --- |
|  | SQL | Power Query |
| **Learning Curve** | Medium. Requires understanding of tables, queries and data. | Requires understanding of Power BI and its outcomes. |
| **Dependency** | Requires database server and data-lake. | Requires external data sources. |
| **User Base** | Large developer community, database admin community. | Specific to the Power BI community. |
| **Programming Support** | SQL is the query language and also supported by other external languages. | It can be custom coded and extended with M language. |
| **Data Operations** | You can perform CREATE, READ, UPDATE and DELETE. | You can perform data transformations on already processed data. |
| **Custom Code** | You can custom code stored procedures and routines to automate. | You can write custom code using the power query to perform specific tasks. |
| **Data Source** | Database Server | Various data sources (files, services, database) etc. |
| **External Dependency** | SQL requires a lot of tools for you for reporting, visualization, query and storage. | Power query is specific to a tool so it is already dependent on single tool. |
| **Permissions** | Every step of the way you would require permission to access the data. | You take permission only when you are working on the data. |
| **Infrastructure** | Large and requires much understanding of data and database. | Limited to the tool. |
| **Refresh Times** | Initially slow, once cached lot easier. | Lot quicker as data is specifically queried by the tool. |
| **Shared Datasets** | You can share the data, reports and other queries using tools. | You share what you are using as a data source. You have limited access. |

Power Query advantages:

Power Query is a data preparation engine run by the Power BI tool. This engine allows the tool to transform the data and also connect the data with the various data sources out there. e.g. MySQL, CSV Files, Snowflake, etc.

Power Query allows you to connect with various sources that are out there—for example, files, databases, services, feeds, data lakes, etc. With it you can connect your desktop and a cloud service with all of those available data source

Power Query disadvantages:

* You can't use parameters passing while working with big data.
* Limited cells that can be previewed.
* Limited data size processed in buffer (tool limitation).
* Performance issues

SQL advantages:

SQL is a standardized programming language that makes it easier for you to design, process and manipulate data from the database server. You can literally do anything you want with the data and also get much quicker and faster performance from your queries.

It's lot faster in terms of performance than Power Query. It's transformations are lot quicker than the Power Query.

SQL disadvantages:

* Steep learning curve of SQL language.
* Simplicity of single tool like Power BI for use case.
* Not suitable for less technically inclined users.
* Requires external libraries and engines for predictive and prescriptive analysis.

**2-Auto-train a regression model:**

In this article, I assume you have already downloaded the data from [Azure Open Datasets](https://docs.microsoft.com/azure/open-datasets/overview-what-are-open-datasets?WT.mc_id=azuremedium-docs-lazzeri) and ran through the data preparation steps in [this tutorial](https://docs.microsoft.com/en-us/azure/machine-learning/service/tutorial-auto-train-models?WT.mc_id=azuremedium-blog-lazzeri) for the [NYC Taxi data](https://docs.microsoft.com/en-us/azure/machine-learning/service/tutorial-auto-train-models?WT.mc_id=azuremedium-blog-lazzeri) so it could be used to build our machine learning model.

**Configure workspace**

Let’s start by creating our workspace object from the existing workspace.

A [Workspace](https://docs.microsoft.com/python/api/azureml-core/azureml.core.workspace.workspace?view=azure-ml-py&WT.mc_id=azuremedium-docs-lazzeri) is a class that accepts your [Azure subscription](https://docs.microsoft.com/office365/enterprise/subscriptions-licenses-accounts-and-tenants-for-microsoft-cloud-offerings?WT.mc_id=azuremedium-blog-lazzeri) and resource information. It also creates a cloud resource to monitor and track your model runs. Specifically, the Workspace.from\_config() reads the file **config.json** and loads the authentication details into an object named ws.

## Split the data into train and test sets

You can start by splitting the data into training and test sets by using the train\_test\_split function in the sklearn library. This function segregates the data into the x, features, dataset for model training and the y, values to predict, dataset for testing.

The test\_size parameter determines the percentage of data to allocate to testing. The random\_state parameter sets a seed to the random generator

The purpose of this step is to have data points to test the finished model that haven’t been used to train the model, in order to measure true accuracy. In other words, a well-trained model should be able to accurately make predictions from data it hasn’t already seen.

You now have the necessary packages and data ready for autotraining your model.

3-AWS ml services:

If there is massive number of visitor like in black Friday or holiday sales that the service where unable to work if there is high or low number of people in website so we can use amazon web services

It was started in 2002 to improve wide services to developer and now aws provide more than 100 services for wide range of domain

The AWS cloud services platform is now used by morethan 45% of global market

Advanteges of AWS:

1-secuireso you have more privacy

2-experience:

You can benefit from from infra structure management practicing

3-flexable:

Allow the user to select OS,database,language

4-easy to use:

We can use it quickly and securely as we can

4-corn jop and how we use it:

**cron job** is a task automated using **cron**, a scheduler tool on a Unix system like Linux. Creating cron jobs helps improve web development and management efficiency, as you don’t need to execute the same tasks repeatedly.

For example, automating tasks like downloading files for backups or updating packages in a [**virtual private server is**](https://www.hostinger.com/tutorials/what-is-vps-hosting) a common cron job use case.

In this article, we will explain the basics of cron jobs, their types, syntax, special strings, and permissions. We will also share cron job best practices and provide command examples to help you understand how to use cron jobs.

Cron works similarly to background processes like **Services**.

A **cron file** is a text file that contains commands to run periodically at a specific time. The **cron table** or **crontab** configuration file is **/etc/crontab** by default.

Only system administrators can edit the system crontab file. However, since Unix-like operating systems support multiple admins, users can create their own files to schedule specific jobs.

With cron jobs, users can automate system maintenance, disk space monitoring, and backups at regular intervals. Due to their convenience, cron jobs are ideal for computers that work 24/7, such as a virtual private server.

While cron job scheduling is popular among system administrators, it is also useful for web developers.

For instance, they can set up three cron jobs to automatically back up a site every day at midnight, check for broken links every Monday at midnight, and clear its cache every Friday at noon.

While convenient, there are several limitations of cron jobs:

* **The shortest interval between jobs is 60 seconds**. Users can only set the cron job interval settings to one minute or more.
* **Missed jobs need a manual reset**. Admins can’t distribute cron jobs to multiple computers on a network. So, if the computer running cron crashes, the scheduled tasks won’t execute. You must restart the missed jobs manually.
* **No auto-retry mechanism**. Cron is designed to run at a given schedule. If a task fails, it won’t run until the next scheduled time. This makes cron unsuitable for incremental tasks.
* **No environment variables**. Crontab can’t read the [**environment variables**](https://www.geeksforgeeks.org/environment-variables-in-linux-unix/) from several files containing configuration data that is required to run some applications properly.

you must understand cron’s syntax and formatting to ensure the script runs properly. The [**crontab syntax**](https://www.hostinger.com/tutorials/crontab-syntax) consists of five fields with the following possible values:

* **Minute.** The minute of the hour the command will run, ranging from **0-59**.
* **Hour.** The hour the command will run, ranging from **0-23** in a 24-hour notation.
* **Day of the month**. The date of the month the user wants the command to run, ranging from **1-31**.
* **Month**. The month that the user wants the command to run. It ranges from **1-12**, representing January until December.
* **Day of the week.** The day of the week for a command to run, ranging from **0-6**. The value represents Sunday-Saturday. In some systems, the value 7 represents Sunday.

If you don’t have a specific value, avoid leaving these fields blank and enteran asteriskinstead. For example, if you want the cron daemon to run the **root/backup.sh**scriptevery Friday at 5:37 pm,

37 17 \* \* 5 root/backup.sh

5-sniffing and scraping in network:

A **network sniffer** “sniffs” or monitors network traffic for information (e.g., where it’s coming from, which device, the protocol used, etc.). Network administrators can use this information to help optimize their environment.

For example, a network sniffer can monitor network usage and track down someone using excessive bandwidth at a university or business organization. You can also use them to help find security holes in your environment. These are all legitimate uses for a network sniffer.

However, a common use for them today lies in black hat hacking. In the wrong hands, network sniffing tools can allow anyone with little to no hacking skills to monitor network traffic over unsecured WiFi networks in order to steal passwords and other private information. This can give network sniffing tools a bad reputation; however, there are still many legitimate uses for network sniffers.

Network packet sniffing can help enhance your security, performing network penetration testing by monitoring the data and ensuring it is encrypted. Other positive uses of network sniffers include:

* Tracking down network traffic bottlenecks
* Testing firewalls for network security efficacy
* Acquiring statistical data on network bandwidth, availability, etc

To best explain how network packet sniffers work, let’s take a step back to review what makes a network work. Networks function as a collection of “nodes,” such as your smartphone, laptop, server, etc., which transfer information over a networked connection. To speed these transfers along their route, networks use packets of data—chunks of data that are broken down and then reassembled after transmission is complete—to help avoid network congestion.

By using network sniffers to “sniff” the packets en route, a user can analyze the traffic via “passive sniffing” (i.e., snooping in on the inflight data) or “active sniffing” (i.e., directly interacting by sending packets and receiving responses from the target devices). The latter unfortunately also allows for cybercrime instances

Scraping and their tools:

Typically companies do not want their unique content to be downloaded and reused for unauthorized purposes. As a result, they don’t expose all data via a consumable API or other easily accessible resource. Scraper [bots](https://www.cloudflare.com/learning/bots/what-is-a-bot/), on the other hand, are interested in getting website data regardless of any attempt at limiting access. As a result, a cat-and-mouse game exists between web scraping bots and various content protection strategies, with each trying to outmaneuver the other.

The process of web scraping is fairly simple, though the implementation can be complex. Web scraping occurs in 3 steps:

1. First the piece of code used to pull the information, which we call a scraper bot, sends an [HTTP](https://www.cloudflare.com/learning/ddos/glossary/hypertext-transfer-protocol-http/) GET request to a specific website.
2. When the website responds, the scraper parses the HTML document for a specific pattern of data.
3. Once the data is extracted, it is converted into whatever specific format the scraper bot’s author designed.

Scraper bots can be designed for many purposes, such as:

1. [Content scraping](https://www.cloudflare.com/learning/bots/what-is-content-scraping/) - content can be pulled from the website in order to site in order to replicate the unique advantage of a particular product or service that relies on content. For example, a product like Yelp relies on reviews; a competitor could scrape all the review content from Yelp and reproduce the content on their own site, pretending the content is original.
2. **Price scraping** - by scraping pricing data, competitors are able to aggregate information about their competition. This can allow them to formulate a unique advantage.
3. **Contact scraping** - a lot of websites contain email addresses and phone numbers in plaintext. By scraping locations like an online employee directory, a scraper is able to aggregate contact details for bulk mailing lists, robo calls, or malicious social engineering attempts. This is one of the primary methods both spammers and scammers use to find new targets.

Typically, all content a website visitor is able to see must be transferred onto the visitor’s machine, and any information a visitor is able to access can be scraped by a bot.

Efforts can be made to limit the amount of web scraping that can occur. Here are 3 methods of limiting exposure to data scraping efforts:

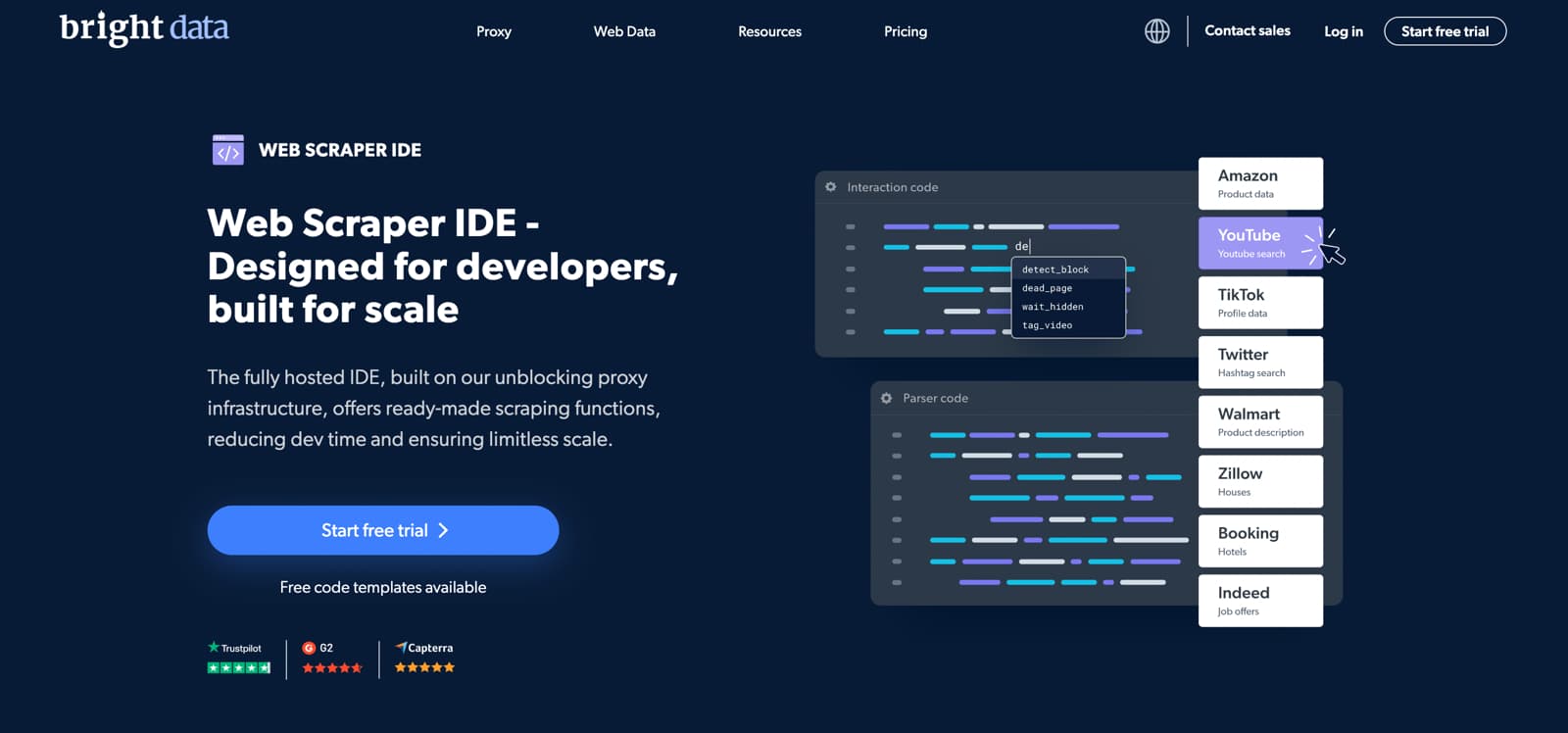
1. **Rate limit requests** - for a human visitor clicking through a series of webpages on a website, the speed of interaction with the website is fairly predictable; you’ll never have a human browsing 100 webpages a second, for example. Computers, on the other hand, can make requests orders of magnitude faster than a human, and novice data scrapers may use unthrottled scraping techniques to attempt to scrape an entire website very quickly. By rate limiting the maximum number of requests a particular [IP address](https://www.cloudflare.com/learning/ddos/glossary/tcp-ip/) is able to make over a given window of time, websites are able to protect themselves from exploitative requests and [limit the amount of data scraping that can occur within a certain window](https://www.cloudflare.com/rate-limiting/).
2. **Modify HTML markup at regular intervals** - data scraping bots rely on consistent formatting in order to effectively traverse website content and parse out and save useful data. One method of interrupting this workflow is to regularly change elements of the HTML markup so that consistent scraping becomes more complicated. By nesting HTML elements, or changing other aspects of the markup, simple data scraping efforts will be hindered or thwarted. For some websites, each time a webpage is rendered, some form of content protection modifications are randomized and implemented. Other websites will change up their markup code occasionally to prevent longer-term data scraping efforts.
3. **Use CAPTCHAs for high-volume requesters** - in addition to using a rate limiting solution, another useful step in slowing content scrapers is the requirement that a website visitor answers a challenge that’s difficult for a computer to surmount. While a human can reasonably answer the challenge, a headless browser\* engaging in data scraping most likely cannot, and certainly will not consistently across many instances of the challenge. However, constant [CAPTCHA](https://www.cloudflare.com/learning/bots/how-captchas-work/) challenges can negatively impact the user experience.

Another less common method of mitigation calls for embedding content inside media objects like images. Because the content does not exist in a string of characters, copying the content is far more complex, requiring optical character recognition (OCR) to pull the data from an image file. But this can also hinder web users who need to copy content such as an address or phone number off a website instead of memorizing or retyping it.

\*A headless browser is a type of web browser, much like Chrome or Firefox, but it doesn’t have a visual user interface by default, allowing it to move much faster than a typical web browser. By essentially running at the level of a command line, a headless browser is able to avoid rendering entire web applications. Data scrapers write bots that use headless browsers to request data more quickly, as there is no human viewing each page being scraped

7-Scrabing tools:

1. Bright Data

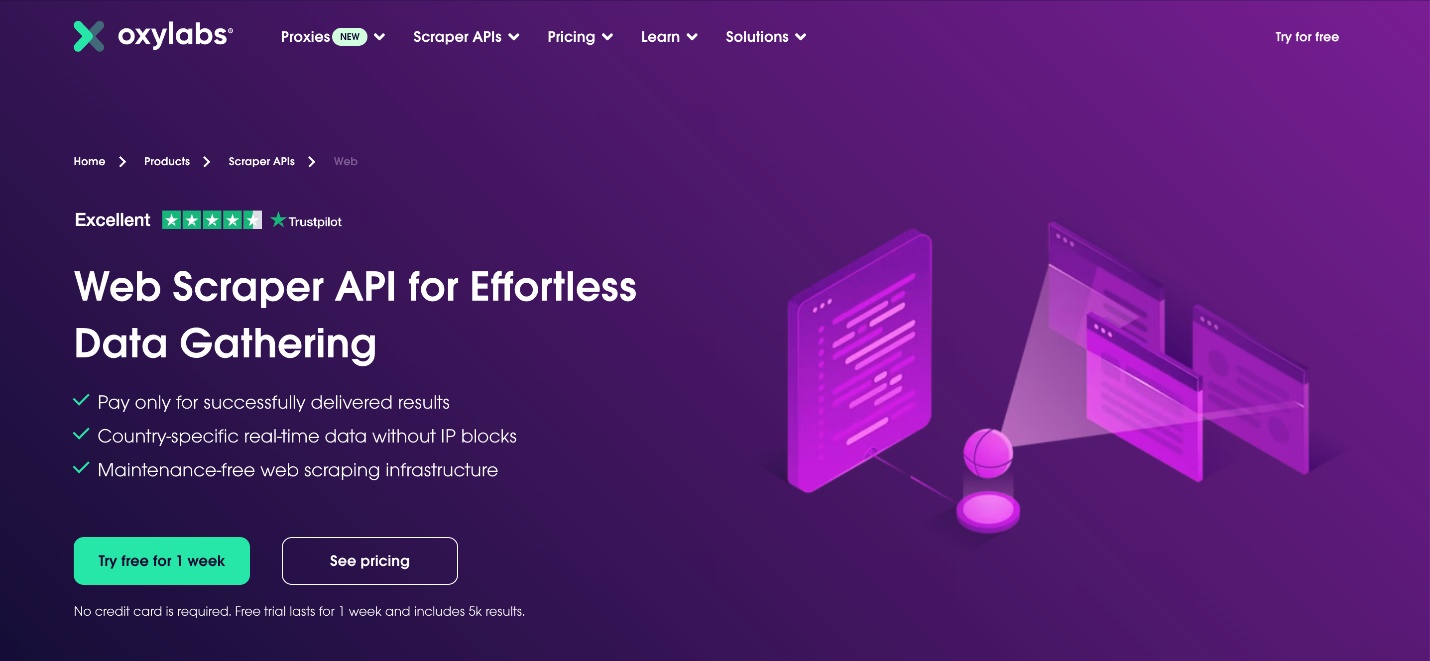
 [Bright Data Web Scraper IDE](https://get.brightdata.com/data-collection)- Designed for developers, built for scale. The fully hosted IDE, built on our unblocking proxy infrastructure, offers ready-made scraping functions, reducing dev time and ensuring limitless scale.

[Get Started For Free](https://brightdata.grsm.io/WS)

Features

* Leverage the Industry’s First Proxy Infrastructure
* Fully Hosted Cloud Environment
* Pre-made web scraper templates
* Browser scripting in JavaScript

2. Oxylabs Scraper API



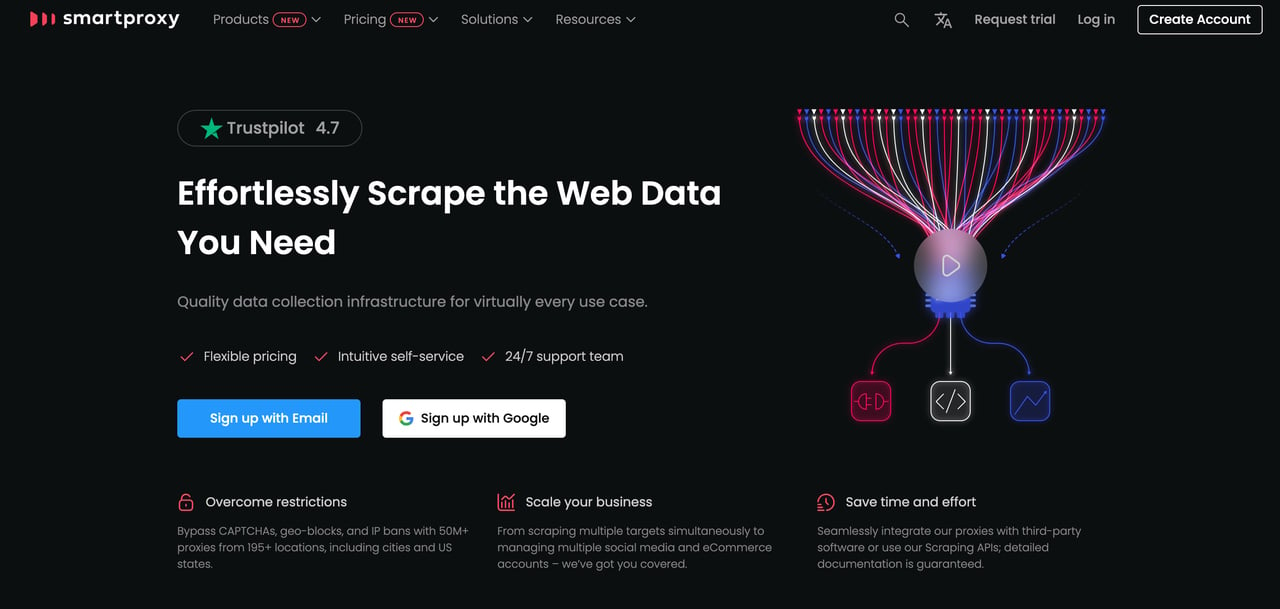
[Oxylabs' Web Scraper API](https://oxylabs.go2cloud.org/aff_c?offer_id=7&aff_id=848&url_id=85) is designed to collect real-time public web data from almost any page. It serves as a trustworthy solution for quick and reliable data extraction. As a result, Web Scraper API best fits, but isn't limited to, use cases such as fraud protection, market research, and travel fare monitoring. It gives free trial for one week.

[Use POPUP20 code to get 20% off for Residential Proxy & Web Scraping API](https://oxylabs.go2cloud.org/aff_c?offer_id=7&aff_id=848&url_id=85)

Features

* Patented Proxy Rotator for block management
* Auto-retry system for failed scraping attempts
* Country-specific geo-targeting
* JavaScript rendering
* Recurring jobs scheduling

3. Smartproxy



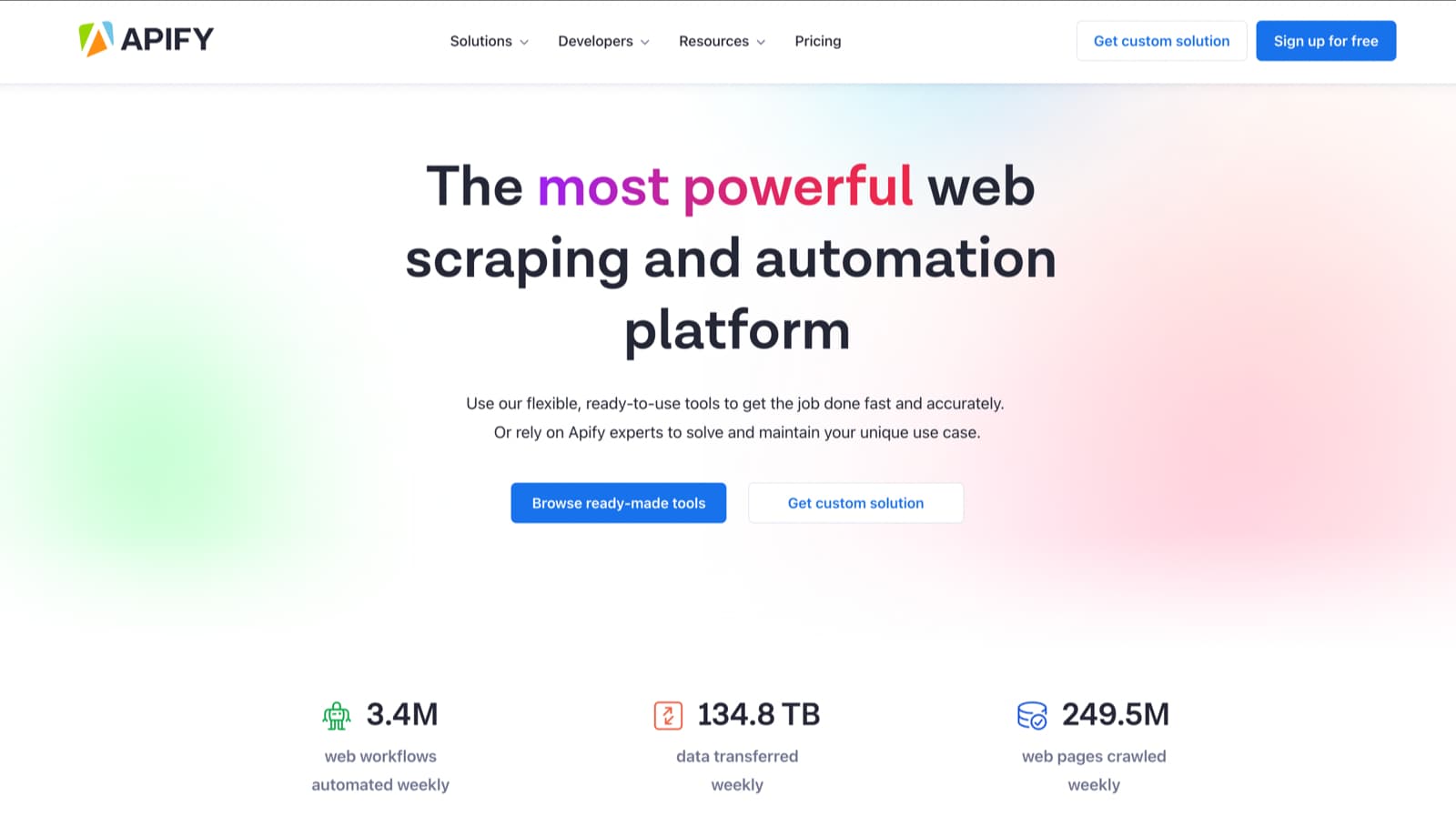
[Smartproxy](https://smartproxy.pxf.io/c/2950697/1480202/17480) offers a variety of Scraping APIs dedicated to different use cases: social media, SERP, eCommerce, and web scraping.

These APIs are fuelled by more than 50M high-quality proxies worldwide, enabling users to target cities. The provider’s proxy network covers 195+ locations, allowing the users to expand their horizons even with the most localized projects.

Features

* Combine proxies, a web scraper, and, in some cases, a data parser
* Users pay for 100% successfully scraped results
* [No-Code Scraper](https://smartproxy.com/scraping/no-code) works based on the click-and-collect principle, allowing users to gather data without writing a single line of code
* 24/7 Smartproxy support can answer any product-related questions via LiveChat

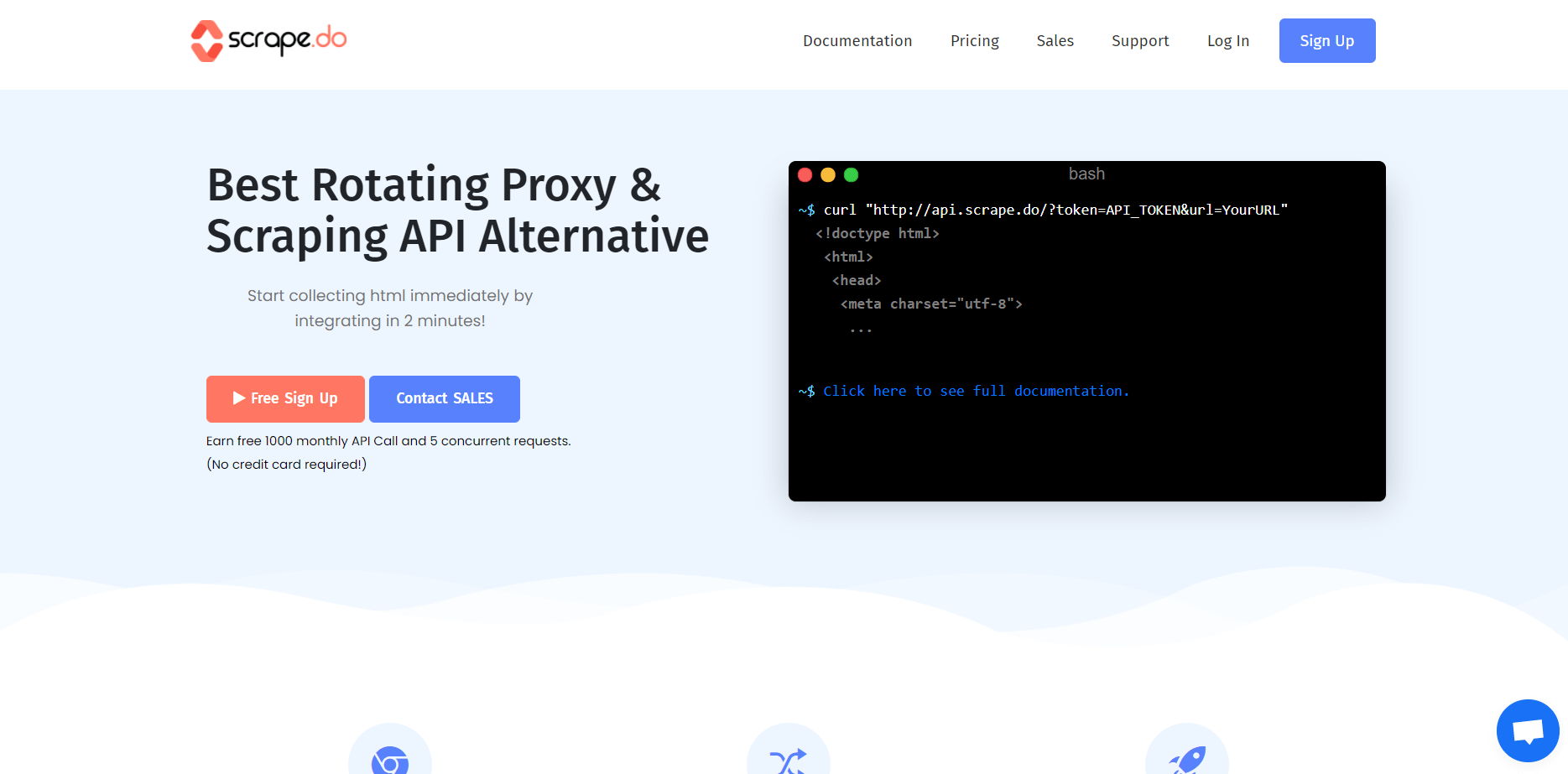
4. Apify

 [Apify](https://www.apify.com/?fpr=htihz" \t "_blank) is the no-code most powerful web scraping and automation platform.

Features

* Hundreds of ready-to-use tools
* No-code, open-source proxy management
* Search engine crawler
* Proxy API
* Browser extension

5. Scrape.do



[Scrape.do](https://scrape.do/?fpr=popupsmart)is an easy-to-use web scraper tool, providing a scalable, fast, proxy web scraper API in an endpoint. Based on cost-effectiveness and features, Scrape.do is on top of the list. As you will see in the continuation of this post, Scrape.do is one of the lowest cost web scraping tools out there.

-Unlike its competitors, Scrape.do does not charge extra for Google and other hard-to-scrape websites.

-It offers the best price/performance ratio in the market for Google scraping (SERP). (5,000,000 SERP for $249)

-Additionally, Scrape.do has 2-3 seconds average speed in collecting anonymous data from Instagram and a 99% success rate.

-Its gateway speed is also 4 times faster than its competitors.

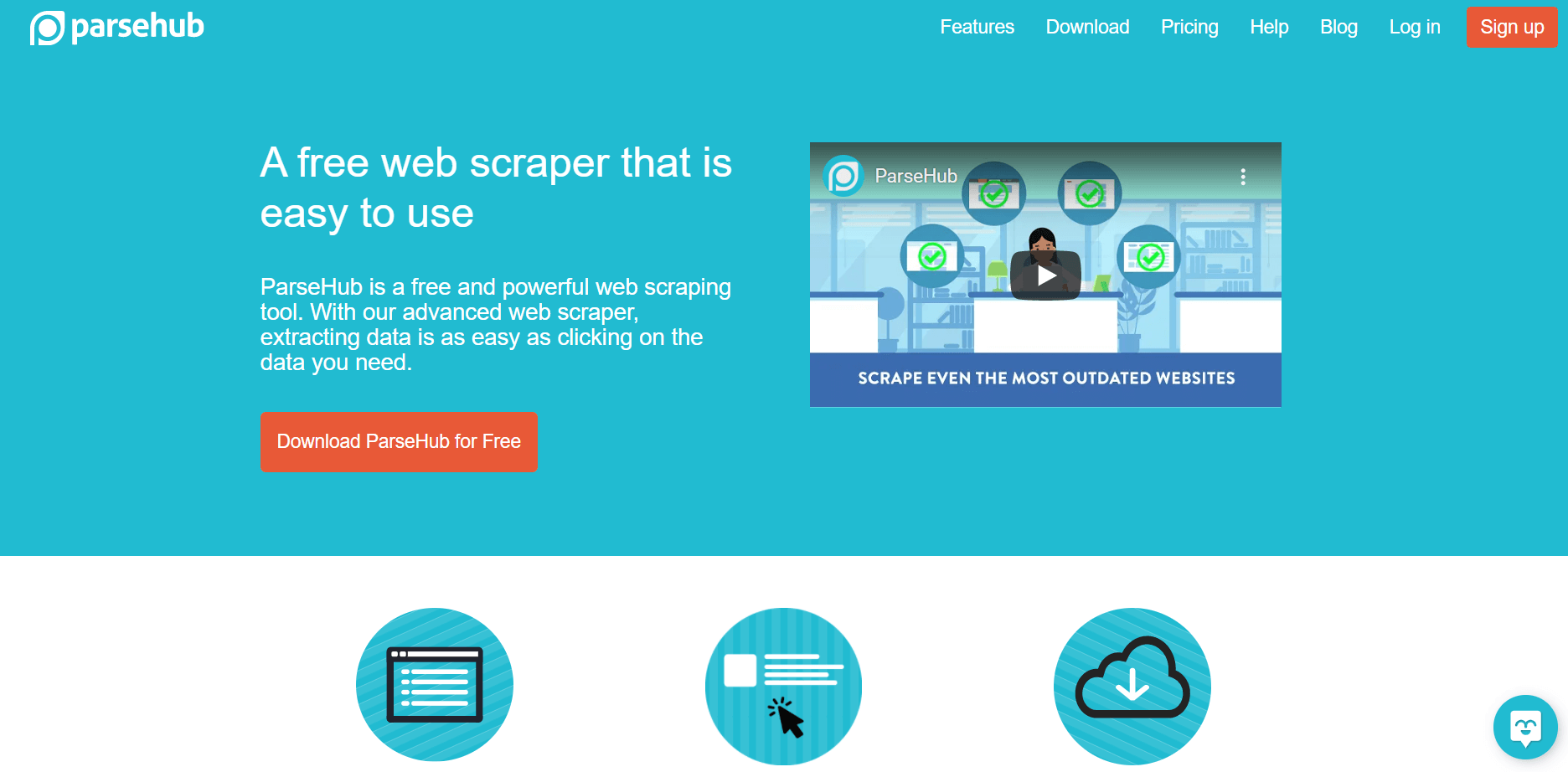
-Moreover, this tool is providing residential and mobile proxy access twice as cheaper.

Here are some of its other features.

Features

* Rotating proxies; allow you to scrape any website. Scrape.do rotates every request made to the API using its proxy pool.
* Unlimited bandwidth in all plans
* Fully customizable
* Only charges for successful requests
* Geotargeting option for over 10 countries
* JavaScript render which allows scraping web pages that require to render JavaScript
* Super proxy parameter: allows you to scrape data from websites with protections against data center IPs.

6. ParseHub

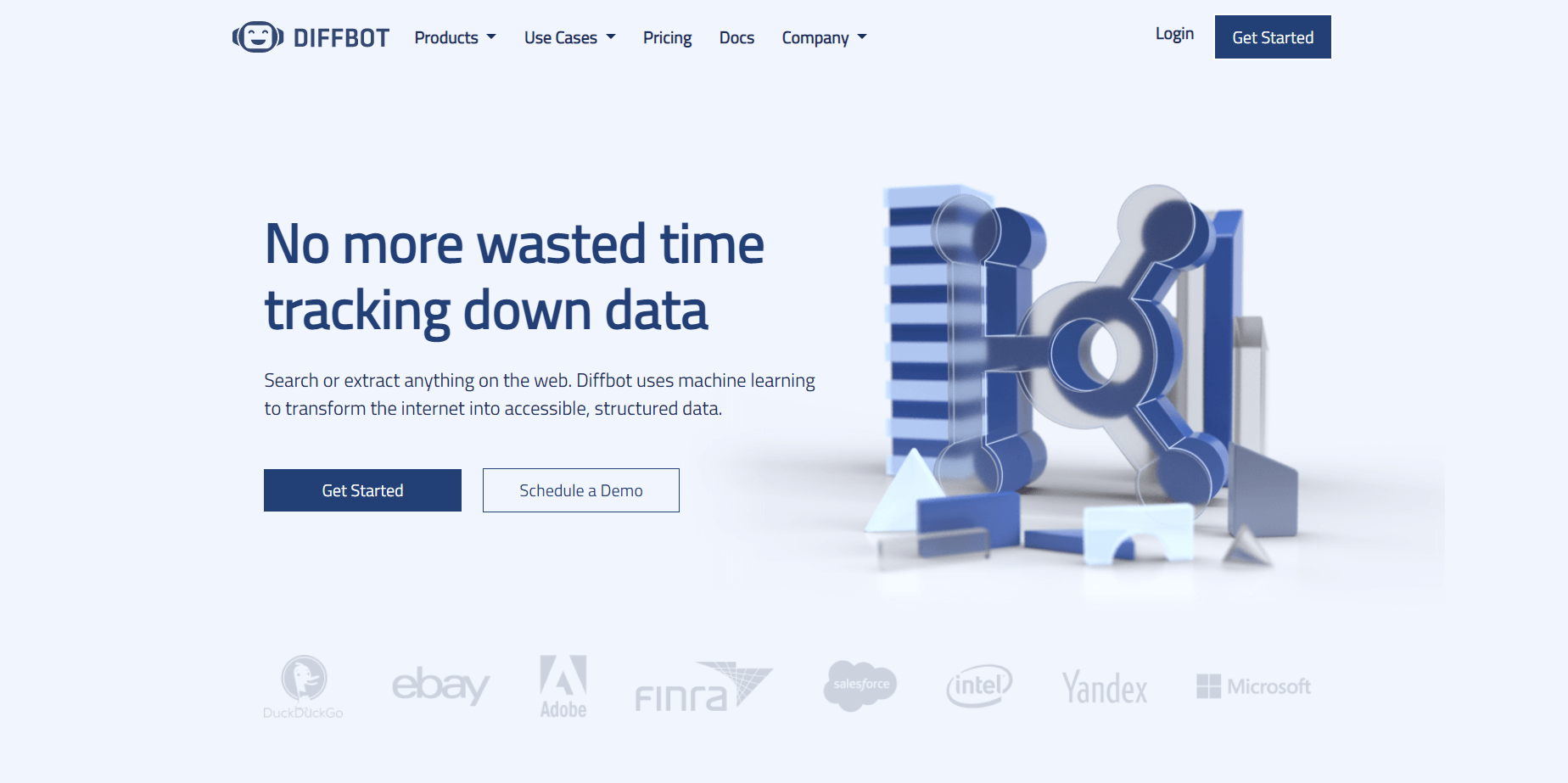


[ParseHub](https://www.parsehub.com/) is a free web scraper tool developed for extracting online data. This tool comes as a downloadable desktop app. It provides more features than most of the other scrapers, for example, you can scrape and download images/files, download CSV and JSON files. Here’s a list of more of its features.

Features

* IP rotation
* Cloud-based for automatically storing data
* Scheduled collection (to collect data monthly, weekly, etc.)
* Regular expressions to clean text and HTML before downloading data
* API & webhooks for integrations
* REST API
* JSON and Excel format for downloads
* Get data from tables and maps
* Infinitely scrolling pages
* Get data behind a log-in

7. Diffbot

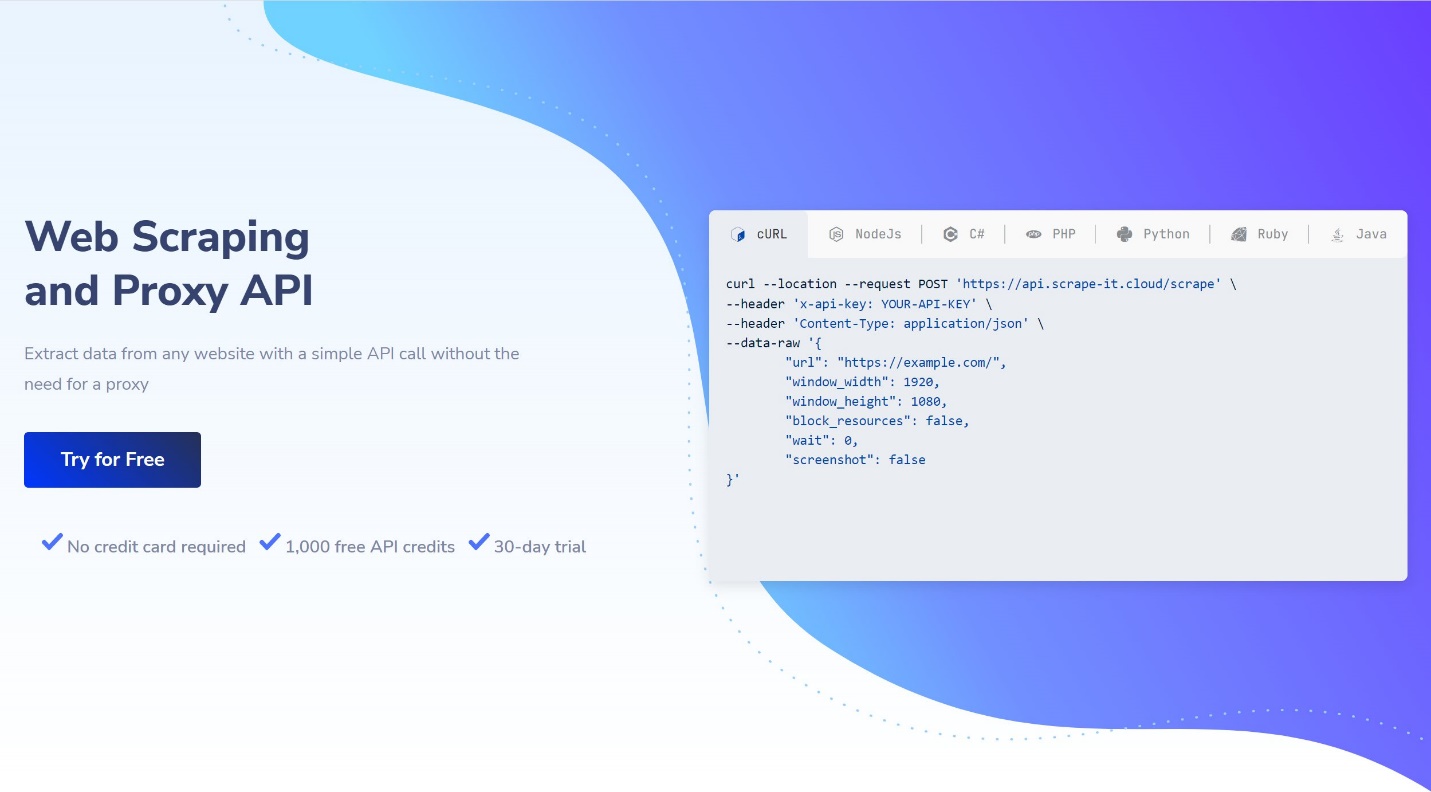


[Diffbot](https://www.diffbot.com/) is another web scraping tool that provides extracted data from web pages. This data scraper is one of the top content extractors out there. It allows you to identify pages automatically with the Analyze API feature and extract products, articles, discussions, videos, or images.

Features

* Product API
* Clean text and HTML
* Structured search to see only the matching results
* Visual processing that enables scraping most non-English web pages
* JSON or CSV format
* The article, product, discussion, video, image extraction APIs
* Custom crawling controls
* Fully-hosted SaaS

8. Scrape-It.Cloud



[Scrape-It.Cloud](https://scrape-it.cloud/) is a web scraping API that helps users extract data from any website with a simple API call, no proxy required.

With features like automatic proxy rotation, automatic scaling, and Javascript rendering, it simplifies and speeds up web scraping.

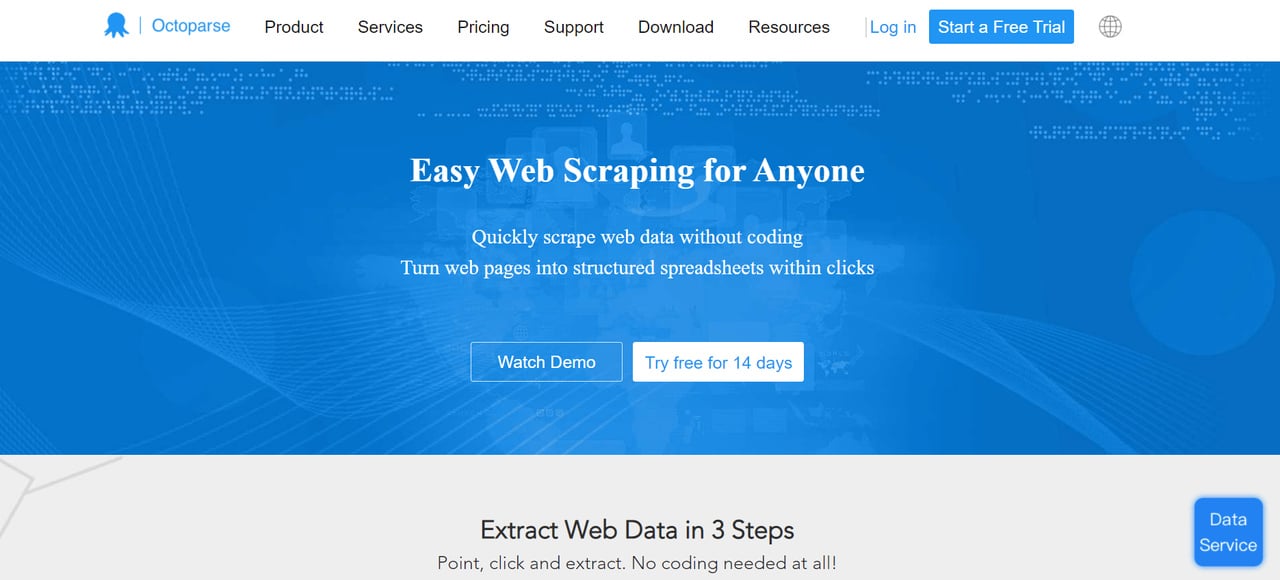
In addition to its web scraping API, the service also offers ready-made no-code scrapers that allow users to easily retrieve the data they need in popular formats such as CSV, Excel, and JSON.

These scrapers can be useful for those who do not have programming skills or do not want to write their own scripts from scratch.

Features

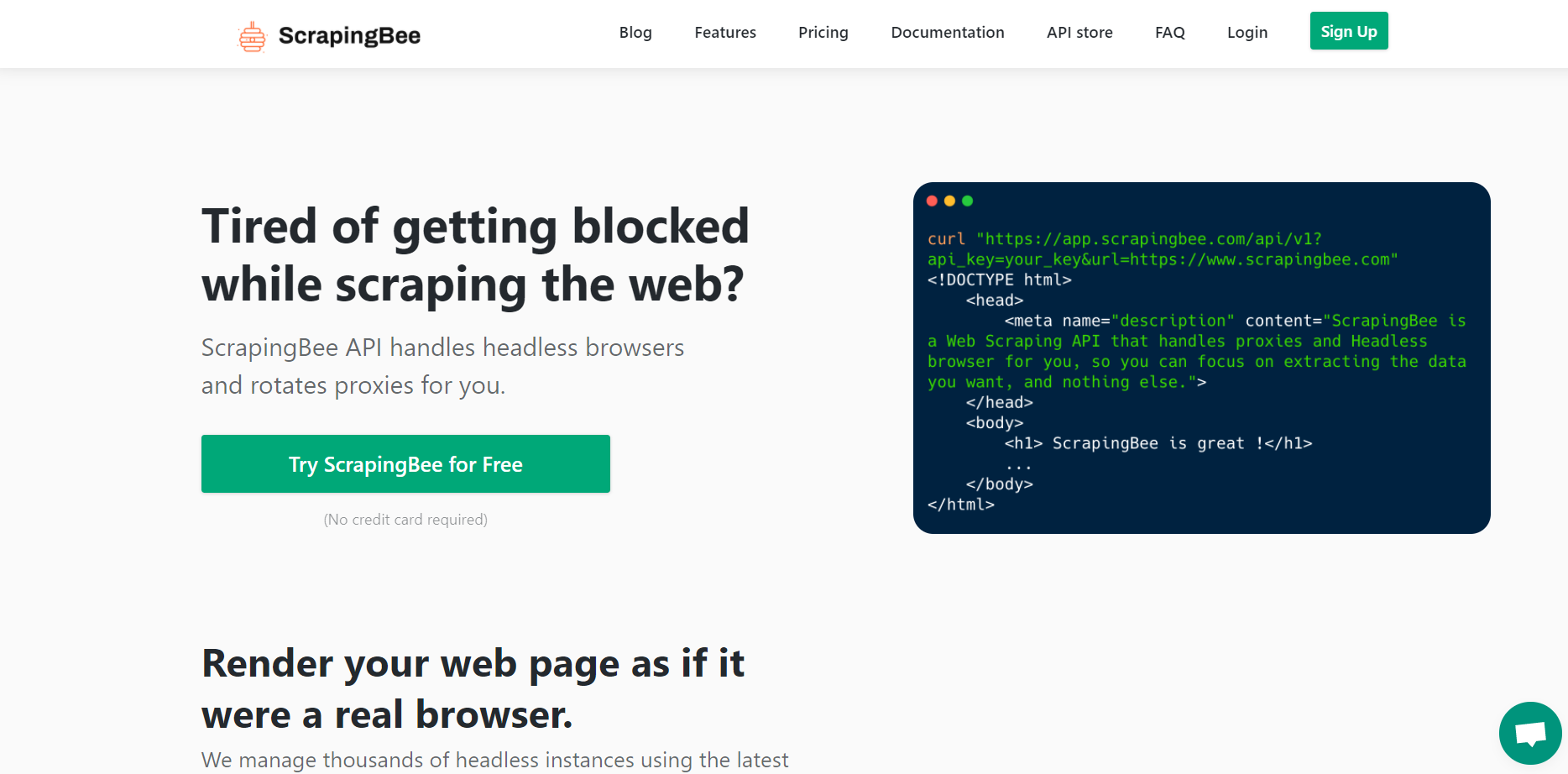
* JavaScript rendering
* Automatic proxy rotation
* Worldwide geotargeting
* Unlimited bandwidth
* Google SERP API

## 9. Octoparse



[Octoparse](https://www.octoparse.com/) stands out as an easy-to-use, **no-code web scraping tool**. It provides cloud services to store extracted data and IP rotation to prevent IPs from getting blocked. You can schedule scraping at any specific time. Besides, it offers an infinite scrolling feature. Download results can be in CSV, Excel, or API formats

10. ScrapingBee



[ScrapingBee](https://www.scrapingbee.com/) is another popular data extraction tool. It renders your web page as if it was a real browser, enabling the management of thousands of headless instances using the latest Chrome version.

So, they claim dealing with headless browsers as other web scrapers do is time-wasting and eating up your RAM & CPU. What else does ScrapingBee offer?

Features

* JavaScript rendering
* Rotating proxies
* General web scraping tasks like real estate scraping, price-monitoring, extracting reviews without getting blocked.
* Scraping search engine results pages
* Growth hacking (lead generation, extracting contact information, or social media.)

8-EERD for banking system:

Bank

\*code\*address\*manages()

ATM withdraw

\*amount\*account no

ATM transfer

\*amount\*account no

Debit card

\*card no\*owner\*access()

Customers

\*name\*address\*owns()

ATM

\*location\*managedby\*maintain()

\*transaction()

ATM transaction

\*data\*type\*update()

Account

\*type \*owner