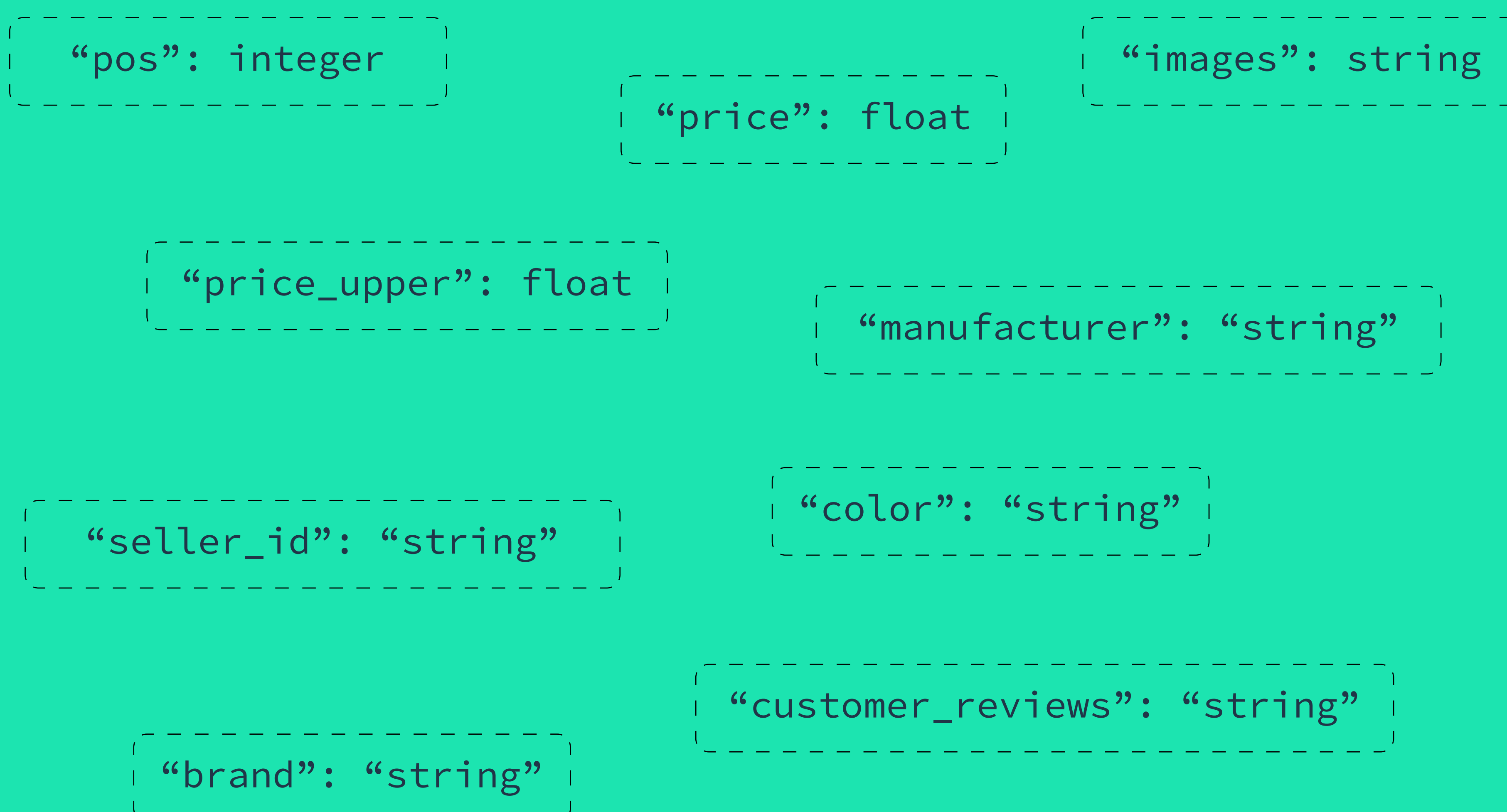


▶▶▶ smartproxy

Quick Start Guide

Scraping APIs



1. About Smartproxy	3
<hr/>	
2. Introduction to Scraping APIs	4
<hr/>	
3. How web scraping APIs work	6
<hr/>	
4. Scraping APIs products	
• Web Scraping API	8
• SERP Scraping API	14
• eCommerce Scraping API	19
• Social Media Scraping API	23
<hr/>	
5. Overview and integrations	
• Pricing	28
• Authentication methods	28
• API playground	29
• Requests usage	30
<hr/>	
6. Resources	
• GitHub	32
• Postman collections	32
<hr/>	
7. Conclusion	33
<hr/>	

1.

About Smartproxy

Smartproxy's data collection infrastructure helps you effortlessly extract web data from even the most challenging targets. Our products come with award-winning 24/7 support, intuitive self-service dashboard, and flexible pricing plans.

2.

Introduction to Scraping APIs

Our Scraping APIs are designed to simplify real-time data collection at scale. They lift the burden of managing proxies, running headless browsers, and overcoming bot detection systems. With a single API call, you can get structured data from the biggest search engines, social media platforms, and eCommerce stores, or raw HTML from any website anywhere in the world.

These APIs are highly scalable and charge only for successful requests, making your expenses predictable. If needed, you can even integrate them in place of a proxy server with very few adjustments.

Proxies vs Scraping APIs

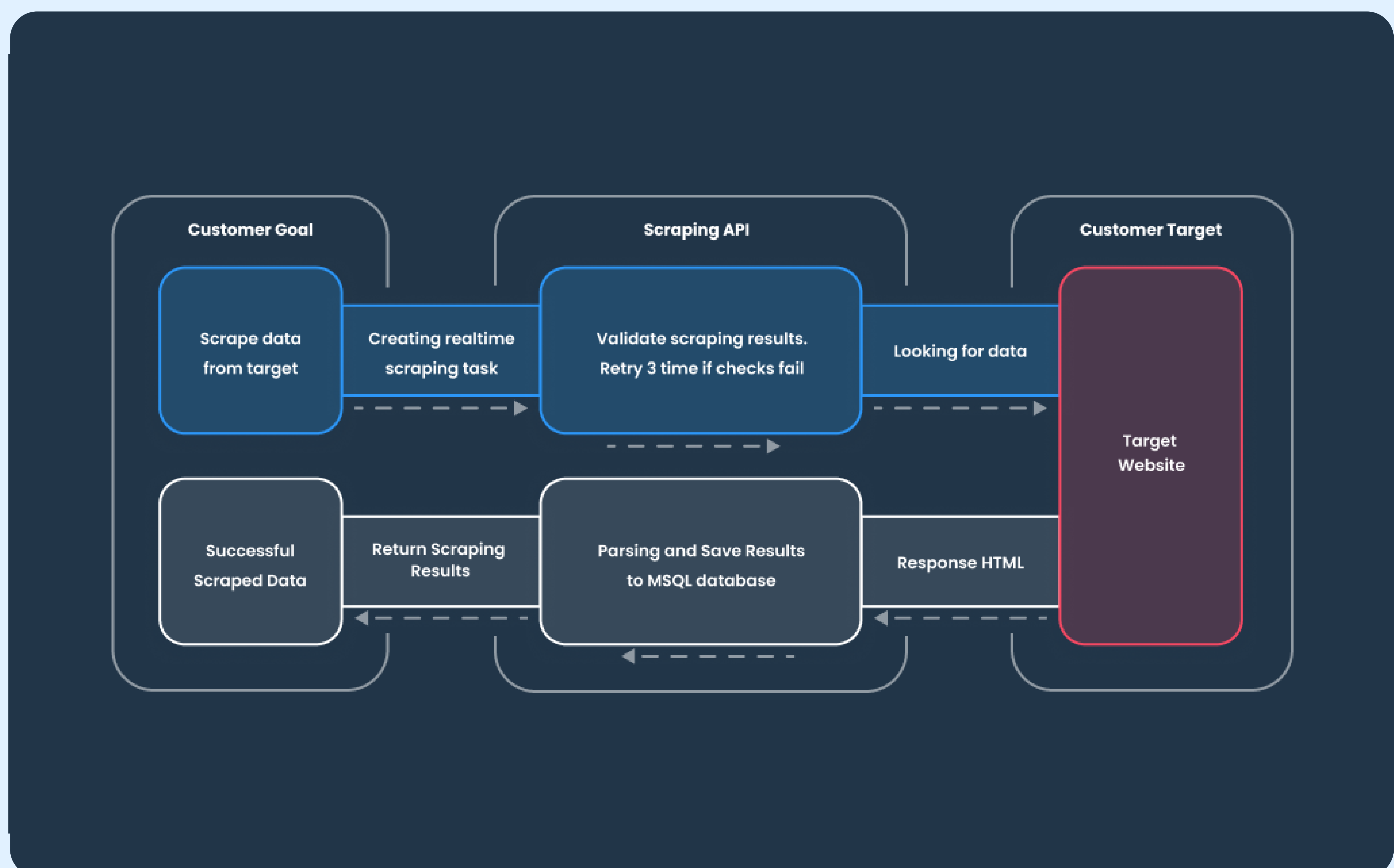
Smartproxy's product family consists of proxies and web scraping APIs. You may be wondering how the two compare. The table below explains their main features:

	Proxies	Scraping API
Integration methods	Proxy	API, proxy-like
Proxy servers	✓	✓
Locations	Worldwide	Worldwide
IP rotation	Automatic or manual	Automatic
Automated website unblocking	—	✓
CAPTCHA solving	—	✓
JavaScript rendering	—	✓
Data parsing	—	✓
Pricing model	Traffic and/or IPs	Successful requests

3.

How Scraping APIs work

Scraping APIs let you send API requests with your target and optional parameters like geolocation or JavaScript rendering. The API automatically applies appropriate proxies, headers, and retries the request if necessary until it scrapes the page. It then returns the result over an open HTTP connection.



4.

Scraping APIs products



Web



SERP



eCommerce



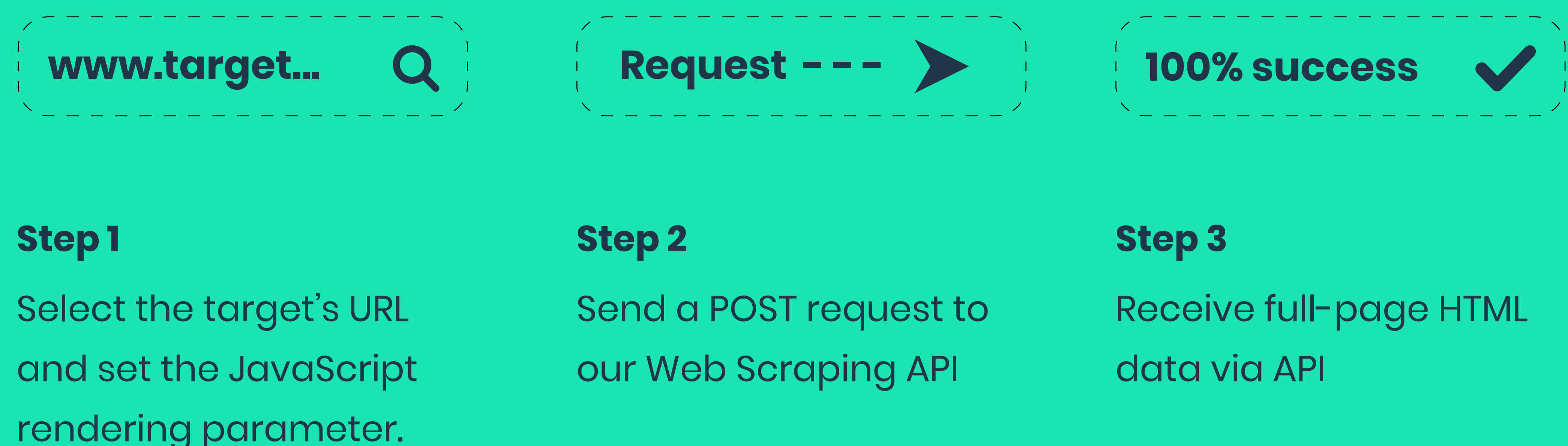
Social Media



Web Scraping API

Web Scraping API tries to scrape any URL you pass its way. It returns the page's HTML.

How does Web Scraping API work?



Web Scraping API can return the HTML of any webpage. It makes a perfect choice for use cases where you need a scalable web scraper that can render JavaScript and overcome website protection mechanisms.

Main features

- Ability to scrape any website you throw its way
- JavaScript rendering for pages that require headless browsers
- Rich targeting options with over 100 supported countries
- Ability to load pages using mobile & desktop device headers

Parameters

Web Scraping API accepts the following parameters. Most of them are optional. The only obligatory parameter is url.

Parameter	Type	Description
target	string	Should be always set to *universal* for Web Scraping API
url	string	Direct URL (link)
locale	string	This will change the web interface language. Example: – en-US – en-GB
geo	string	The geographical location that the result depends on. City location names, state names, country names, coordinates and radius, Google's Canonical
device_type	string	Device type and browser. Supported: desktop, desktop_chrome, desktop_firefox, mobile, mobile_android, mobile_ios.
headless	string	Enable JavaScript rendering. Supported: html, png

Output examples

```
{
  "results": [
    {
      "content": "<html> page content here</html>"
      "status_code": integer,
      "url": "string",
      "task_id": "string",
      "created_at": "string",
      "updated_at": "string"
    }
  ]
}
```

Authentication methods

The web scraping APIs use username and password authentication. We provide the username, and you can create a password in the dashboard. The credentials are encoded in Base64 and passed as an authorization header.

```
curl --request POST \
  -- url http://scrape.smartproxy.com/v1/tasks \
  -- header 'Content-Type: application/json' \
  -- header 'accept: application/json' \
  -- header 'authorization: Basic U1B1c2VybmFtZToTUHBhc3N3b3Jk' \
  {
    "target" "google search",
    "query" "world",
    "parse" "true",
    "locale" "en-GB",
    "geo" "London, England, United Kingdom",
  }
```

Integration methods

All our APIs support two integration methods: real-time and proxy-like. Both return data over an open connection, meaning that you send a request and wait for the response.

- **Real-time**

This is the main integration method. It lets you send POST requests to the API endpoint with parameters in a JSON payload. This way, you can specify data sources (such as Google Search) instead of providing the full URL.

```
curl -u username:password 'https://scrape.smartproxy.com/v1/tasks' -H "Content-Type: application/json" -d '{"target": "google_search", "domain": "com", "query": "world"}'
```

- **Proxy-like**

This method lets you integrate the APIs as a proxy server. It's useful when your infrastructure is based on the proxy format, or you're transitioning from proxies. The method requires passing a full URL with parameters in the request headers.

```
curl -k -x scrape.smartproxy.com:60000 -U username:password -H "X-Smartproxy-Device-Type: desktop_firefox" -H "X-Smartproxy-Geo: California,United States" "https://www.google.com/search?q=world"
```


Response codes

The two tables below show the possible response codes you may encounter while using the APIs.

HTTP response codes:

Response	Description	Solution
200 - Success	Server has replied and given the requested response.	Celebrate!
204 - No content	Job not completed yet.	Wait a few seconds before trying again.
400 - Multiple error messages	Bad structure of the request.	Re-check your request to make sure it is in the correct format.
401 - Invalid / not provided authorization header	Incorrect login credentials or missing authorization.	Re-check your provided credentials for authorization.
403 - Forbidden	Your account does not have access to this resource.	Make sure your Google target is supported by us
404 - Not found	Your target was not found.	Re-check your targeted URL.
429 - Too many requests	Exceeded rate limit for your subscription.	Make sure you still have at least one request left. Wait a couple minutes and try again. If you are encountering the error often – chat with us to see if your rate limit can be increased.
500 - Internal error	Service unavailable, possibly due to some issues we are encountering.	Wait a couple minutes and send another request. Contact us for more information.
524 - Timeout	Service unavailable, possibly due to some issues we are encountering.	Wait a couple minutes and send another request. Contact us for more information.

Parsed result response codes:

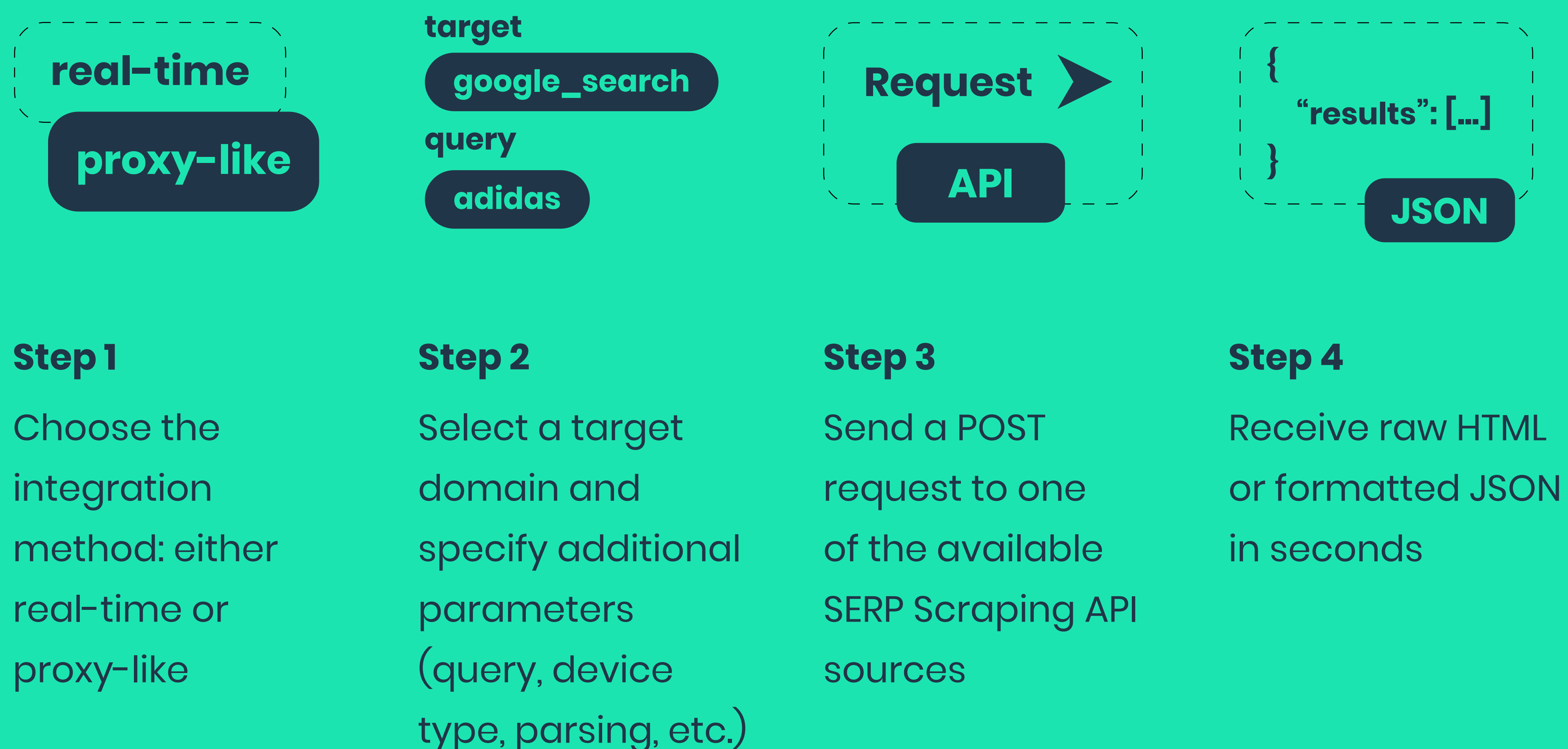
Response	Description
12000 - Success	Server has replied and given the requested response.
12002 - Error	Parsing has failed completely.
12003 - Not supported	Targeted website parsing is not supported.
12004 - Response not full	Some fields were not parsed and are missing.
12005 - Response not fully parsed	Some fields might not have been parsed and are returned unparsed.
12006 - Error	Unexpected error. Let us know the task ID and we will check what went wrong.
12007 - Unknown	We could not determine whether the data was parsed correctly.
12008 - Error	Failed to parse all the data.
12009 - Error	Target not found. Make sure the parameters you passed are correct and supported.



SERP Scraping API

SERP Scraping API lets you scrape Google, Baidu, Bing, and Yandex by entering a URL or sending the search query as a parameter. It returns data in HTML or, in the case of Google, parsed JSON.

How does SERP Scraping API work?



SERP Scraping API is able to extract structured real-time results from major search engines. It makes a perfect choice for search engine optimization, brand protection, and other use cases that involve search engine data.

Main features

- Ability to scrape Baidu, Bing, Google, and Yandex
- Localized results with country, state, city, and zip code targeting
- Option to enter a search query as a parameter for easier use
- Parsing capabilities for various Google data types like search results, ads, and Shopping

Main targets

- Google Search results with all page elements*
- Google Ads*
- Google Shopping (extract search, product, and pricing data) *
- Google Hotels
- Google Images
- Google Suggest
- Google Trends
- Baidu Search results
- Bing Search results
- Yandex Search results

* *parsable*

Parameters

SERP Scraping API accepts the following parameters. Most of them are optional. The only obligatory parameters are target and url if you're entering a link directly, or target and query.

Parameter	Type	Description
target	string	Data source. Available targets are listed here .
url	string	Direct URL (link)
domain	string	Top-level domain of your target.
query	string	...
page_from	integer	Starting page number.
num_pages	integer	Number of results to retrieve in each page.
locale	string	This will change the Google search page web interface language (not the results). Example: – en-US – en-GB
geo	string	The geographical location that the result depends on. City location names, state names, country names, coordinates and radius, Google's Canonical
device_type	string	Device type and browser. Supported: desktop, desktop_chrome, desktop_firefox, mobile, mobile_android, mobile_ios.

Parameter	Type	Description
parse	boolean	true' will return parsed output in JSON format. Leave blank for HTML – not all data sources can be parsed.
google_results_language	string	Shows results in a particular language. All of the supported languages are listed here .
google_tbm	string	This parameter lets you filter Google Search results for specific types of content (news, apps, videos...).
google_tbs	string	This parameter contains parameters, like limiting/ sorting results by date.
google_safe_search	string	Used to hide explicit content from the results.
stars	Integer array	2-5 stars, used with google_travel_hotels target
guests	Integer	Used with google_travel & google_travel_hotels targets
date_range	string	Y-m-d,Y-m-d used with google_hotels & google_travel_hotels targets
headless	string	Enable JavaScript rendering. Supported: html, png

Output example for

Google Shopping Pricing

```
{
  "results": [
    {
      "content": {
        "url": "string",
        "title": "string",
        "rating": float,
        "pricing": [
          {
            "price": float,
            "seller": "string",
            "details": "string",
            "currency": "string",
            "condition": "string",
            "price_tax": float,
            "price_total": float,
            "seller_link": "string",
            "price_shipping": float
          }
        ],
        "review_count": integer,
        "parse_status_code": 12000
      },
    ]
  }
}
```

Bing, Yandex, Baidu

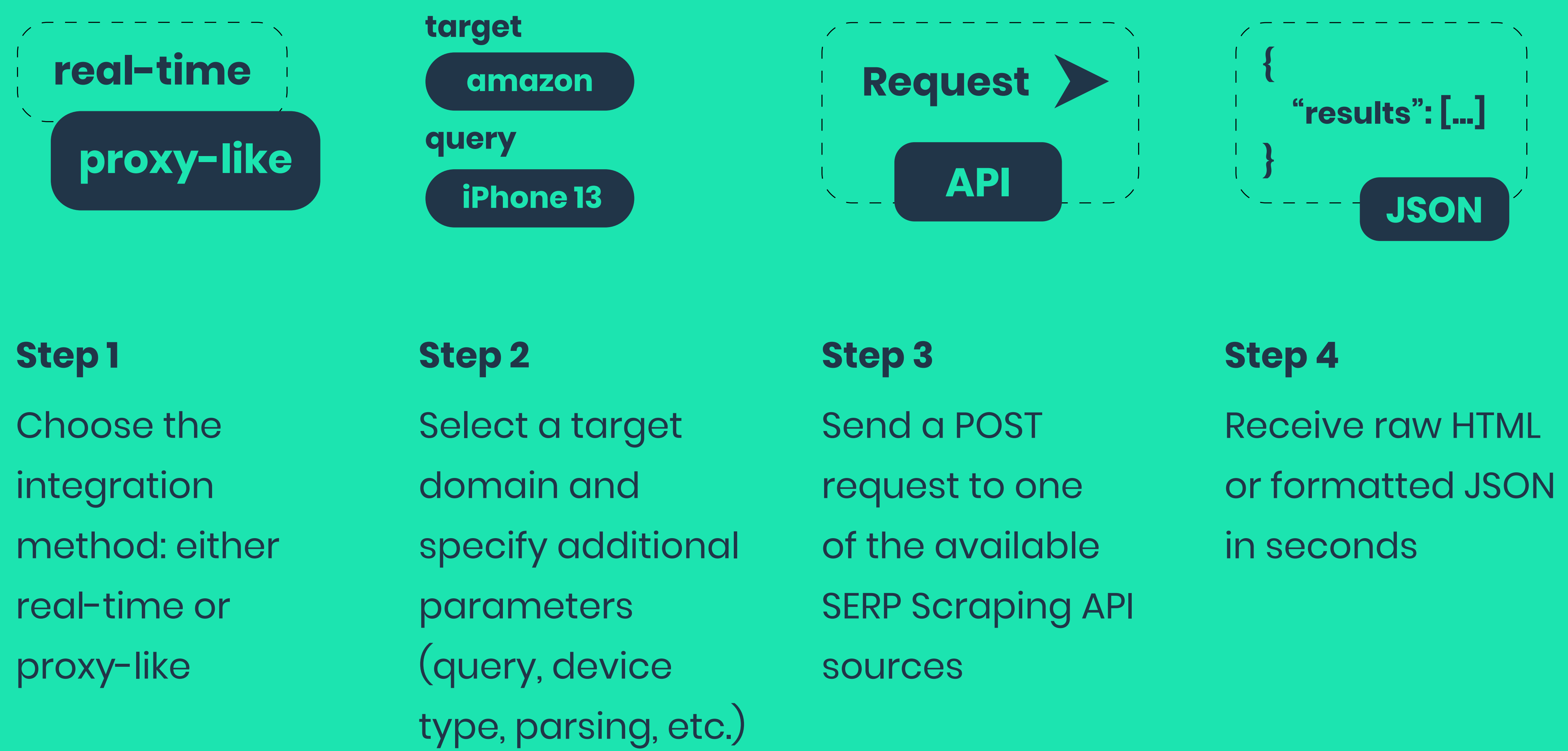
```
{
  "results": [
    {
      "content": "<html> page content here</html>"
      "status_code": 200,
      "url": "string",
      "task_id": "string",
      "created_at": "string",
      "updated_at": "string"
    }
  ]
}
```




eCommerce Scraping API

eCommerce Scraping API lets you scrape Amazon and Wayfair by entering a URL or sending the query as a parameter. It returns data in HTML or, in the case of Amazon, parsed JSON.

How does eCommerce Scraping API work?



eCommerce Scraping API is able to extract structured real-time results from major eCommerce websites. It makes a perfect choice for price monitoring, market research, and other use cases that involve eCommerce data.

Main features

- Ability to scrape Amazon and Wayfair
- JavaScript rendering for pages that require headless browsers
- Option to enter a search query or item code as a parameter for easier use
- Parsing capabilities for various Amazon data types like search results, product pages, and reviews

Main targets

- Amazon search pages*
- Amazon product pages*
- Amazon product pricing*
- Amazon product reviews*
- Amazon product questions*
- Amazon sellers*
- Wayfair product pages

* *parsable*

Parameters

eCommerce Scraping API accepts the following parameters. Most of them are optional. The only obligatory parameters are url if you're entering a link directly, or target and query.

Parameter	Type	Description
target	string	Data source. Available targets are listed here .
url	string	Direct URL (link)
parse	boolean	True' will return parsed output in JSON format. Leave blank for HTML – not all data sources can be parsed.
domain	string	Top-level domain of your target.
query	string	...
page_from	integer	Starting page number.
num_pages	integer	Number of results to retrieve in each page.
locale	string	This will change the web interface language. Example: – en-US – en-GB
geo	string	The geographical location that the result depends on. City location names, state names, country names, coordinates and radius, Google's Canonical

Parameter	Type	Description
device_type	string	Device type and browser. Supported: desktop, desktop_chrome, desktop_firefox, mobile, mobile_android, mobile_ios.
headless	string	Enable JavaScript rendering. Supported: html, png

Output example for Amazon Pricing

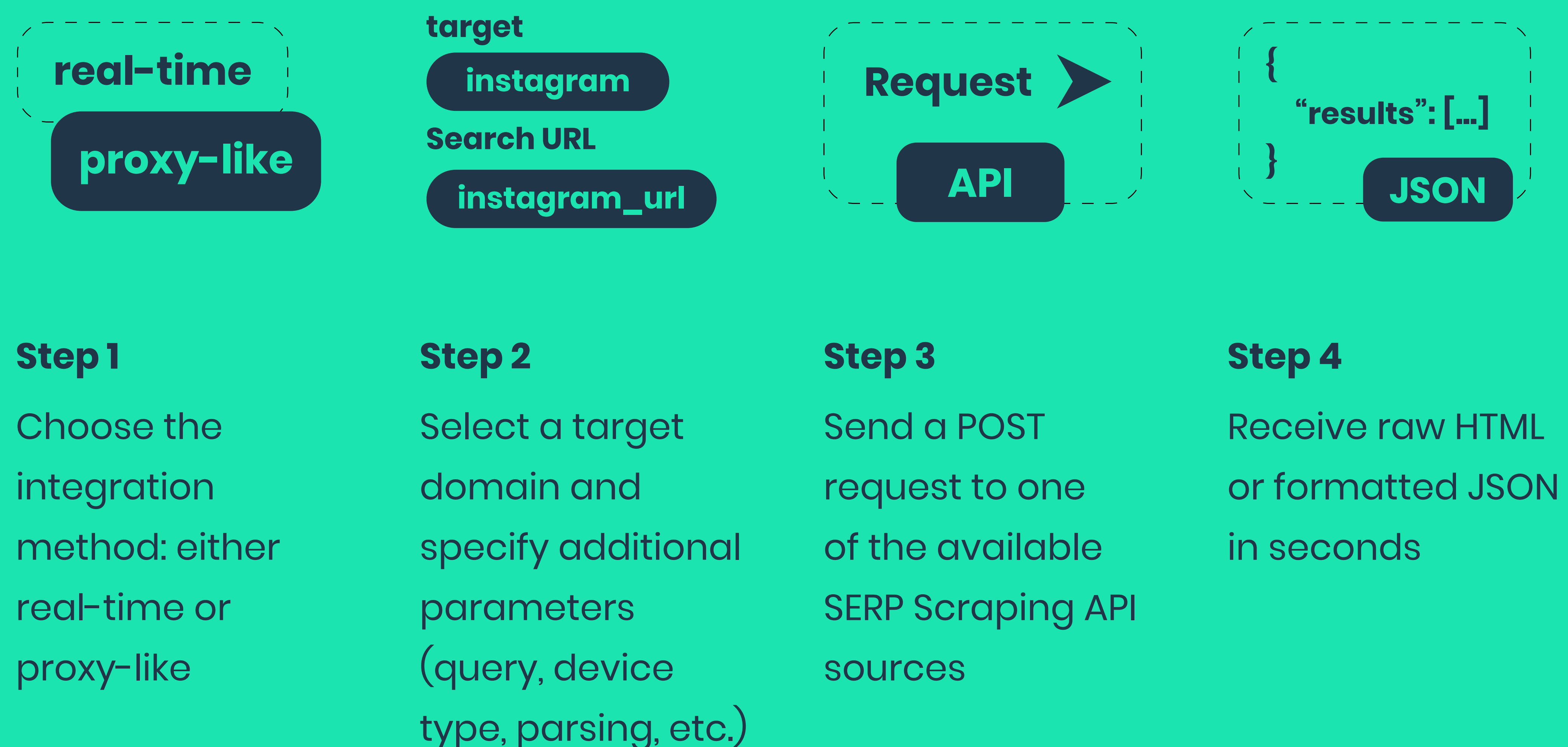
```
{
  "results": [
    {
      "content": {
        "url": "string",
        "asin": "string",
        "page": integer,
        "title": "string",
        "pricing": [
          {
            "price": float,
            "seller": "string",
            "currency": "string",
            "delivery": "string",
            "condition": "string",
            "seller_id": "string",
            "seller_link": "string",
            "rating_count": integer,
            "price_shipping": float,
            "delivery_options": []
          }
        ],
        "asin_in_url": "string",
        "review_count": integer,
        "parse_status_code": 12000
      }
    }
  ]
}
```



Social Media Scraping API

Social Media Scraping API lets you scrape Instagram and TikTok by entering a URL or sending the query as a parameter. It returns data in HTML or parsed JSON.

How does Social Media Scraping API work?



Social Media Scraping API is able to extract structured results from major social media platforms in real time or on demand. It makes a perfect choice for sentiment analysis, influencer marketing, and other use cases that involve social media data.

Main features

- Ability to scrape Instagram and TikTok
- Option to target a GraphQL endpoint or fully render the page
- Real-time or on-demand data delivery
- Parsing capabilities for extracting structured results

Parameter	Type	Description
url	url	Social Media URL
target	string	Desired target
locale	string	Language Locale
geo	string	Geolocation

Output example for TikTok

```
{
  "data": {
    "content": {
      "nickname": "string",
      "verified": boolean,
      "avatarThumb": "string",
      "openFavorite": boolean,
      "ttSeller": boolean,
      "postInfo": {
        "id": "string",
        "description": "string",
        "postedAtTimestamp": integer,
        "postedAt": "string",
        "author": "string",
        "music": {
          "id": "string",
          "title": "string",
          "playUrl": "string",
          "coverLarge": "string",
          "coverMedium": "string",
          "coverThumb": "string",
          "authorName": "string",
          "original": boolean,
          "duration": integer,
          "scheduleSearchTime": integer
        },
        "shareCount": integer,
        "commentCount": integer,
        "playCount": integer,
        "accountLikes": integer
      }
    },
    "errors": [],
    "status_code": integer
  },
  "task_id": "string",
  "url": "string"
}
```

Output example for Instagram

```
{
  "data": {
    "content": {
      "user": {
        "biography": "string",
        "bio_links": [
          {
            "title": "string",
            "lynx_url": "string",
            "url": "string",
            "link_type": "string"
          }
        ],
        "biography_with_entities": {
          "raw_text": "string",
          "entities": []
        },
        "blocked_by_viewer": boolean,
        "restricted_by_viewer": boolean,
        "country_block": boolean,
        "external_url": "string",
        "external_url_linkshimmed": "string",
        "edge_followed_by": {
          "count": integer
        },
        "fbid": "string",
        "followed_by_viewer": boolean,
        "edge_follow": {
          "count": 1111
        },
        "follows_viewer": boolean,
        "full_name": "string",
        "group_metadata": "string",
        "has_ar_effects": boolean,
        "has_clips": boolean,
        "has_guides": boolean,
        "has_channel": boolean,
        "has_blocked_viewer": boolean,
        "highlight_reel_count": integer,
        "has_requested_viewer": boolean,
        "hide_like_and_view_counts": boolean,
        "id": "string",
        "is_business_account": boolean,
        "is_professional_account": boolean,
```

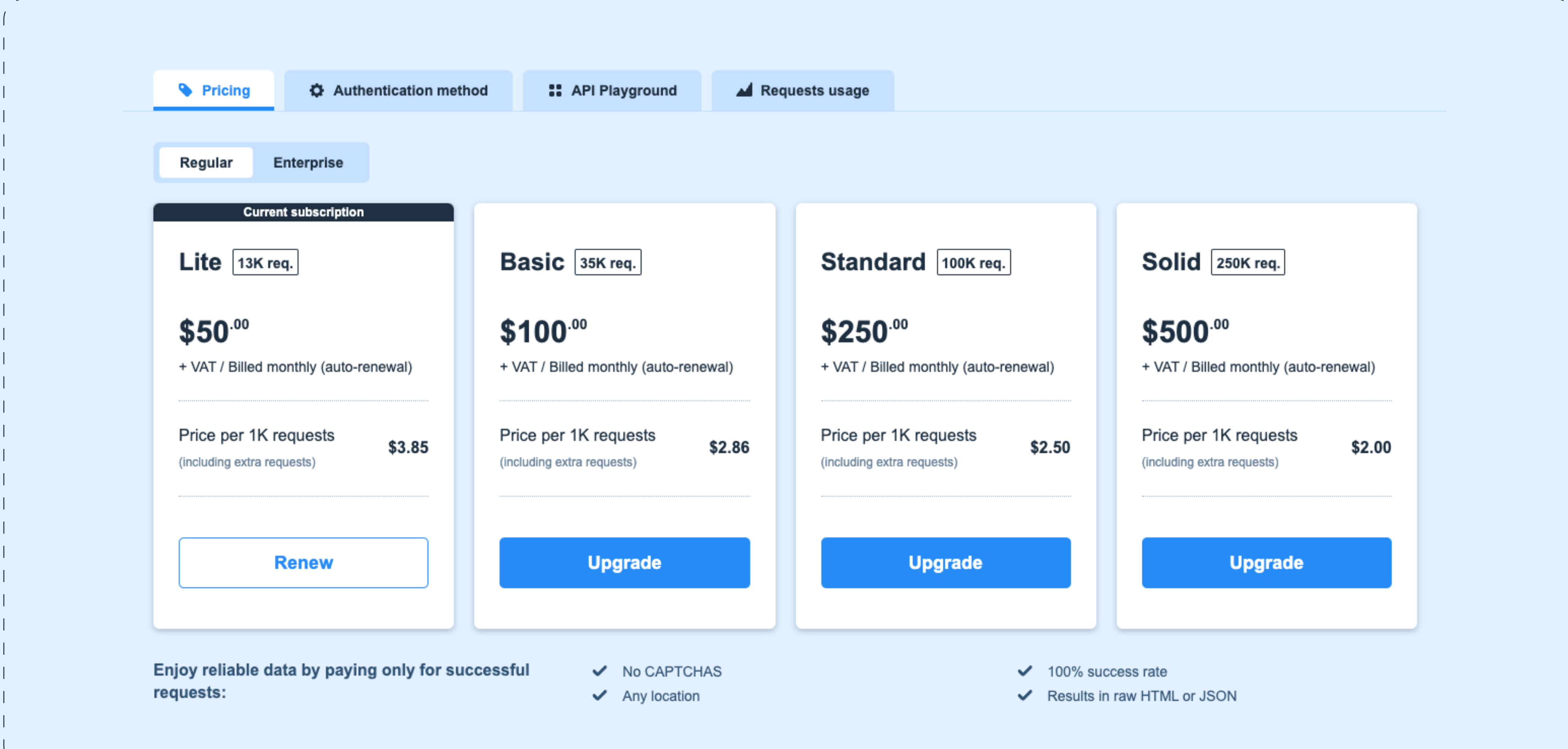

5.

Overview and integrations

Each scraping API has its own section in the dashboard. There, you can manage your subscription, set up the API, and track usage statistics.

Pricing

The Pricing tab lets you buy, upgrade, or renew a subscription.



The screenshot shows the Pricing tab interface with four subscription options: Lite (13K req.), Basic (35K req.), Standard (100K req.), and Solid (250K req.). Each option displays its price, VAT, and price per 1K requests. The Lite plan is the current subscription and has a 'Renew' button, while the others have 'Upgrade' buttons. Below the plans, there are three checkmarks indicating features: 'No CAPTCHAS', 'Any location', '100% success rate', and 'Results in raw HTML or JSON'.

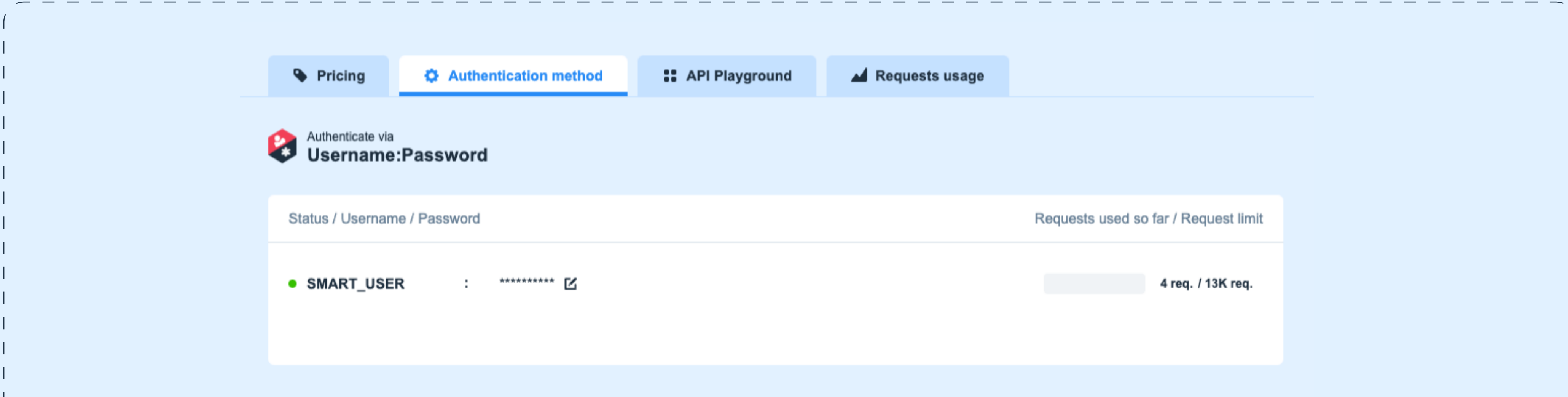
Plan	Request Limit	Price	Price per 1K requests	Action
Lite	13K req.	\$50.00	\$3.85	Renew
Basic	35K req.	\$100.00	\$2.86	Upgrade
Standard	100K req.	\$250.00	\$2.50	Upgrade
Solid	250K req.	\$500.00	\$2.00	Upgrade

Enjoy reliable data by paying only for successful requests:

- ✓ No CAPTCHAS
- ✓ Any location
- ✓ 100% success rate
- ✓ Results in raw HTML or JSON

Authentication method

The Authentication method tab lets you create a password for accessing the APIs.



The screenshot shows the Authentication method tab interface. It displays the authentication method as 'Username:Password'. Below this, there is a table showing the status of the user 'SMART_USER' with 4 requests used out of a 13K limit.

Status / Username / Password	Requests used so far / Request limit
● SMART_USER : *****	4 req. / 13K req.

Try Scraping API playground

Continue to dashboard

API playground

The API playground tab includes an interactive widget for configuring the API.

You can use it to test requests even without a subscription, and it generates dynamic code samples for easier integration.

The screenshot displays the 'SERP Scraping API playground' interface. At the top, there are navigation tabs for 'Pricing', 'Authentication method', 'API Playground' (which is active), and 'Requests usage'. Below the tabs, the title 'SERP Scraping API playground' is followed by a brief description: 'This API search is best for testing target's request. Our API provides plenty of setting parameters and filters that allow you to refine your query.'

The main configuration area is divided into several sections:

- Target:** A dropdown menu set to 'Google search' and a blue 'Send request' button.
- Parameters:**
 - Search query:** A text input field containing 'poker'.
 - Website domain:** A dropdown menu set to '.com'.
 - Language:** A dropdown menu set to 'English'.
 - Location:** A dropdown menu set to 'United States'.
 - Device type and browser:** A dropdown menu set to 'Desktop'.
 - Starting page number:** A numeric input field set to '1'.
 - Page number:** A numeric input field set to '10'.
- Advanced parameters:**
 - Parse:** A toggle switch that is turned on.
 - Autocorrection:** A toggle switch that is turned on.
 - Safe search:** A toggle switch that is turned on.
 - Tbm:** A dropdown menu set to 'No selection'.
 - Tbs:** A text input field containing 'Enter Google TBS'.
 - Results language:** A dropdown menu set to 'English'.

On the right side, there are two panels:

- Request:** A dark blue panel with a 'Copy' button. It displays a cURL command and a corresponding JSON payload. The JSON payload includes fields for 'target', 'query', 'parse', 'domain', 'locale', 'google_results_language', 'geo', 'device_type', 'page_from', 'num_pages', and 'google_nfr'.
- Response:** A light blue panel with a 'Copy' button and a 'JSON' icon. It displays a JSON response structure with 'results' and 'content' fields, including details like 'url', 'page', 'results', 'paid', 'pos', 'desc', 'title', 'data_rw', 'url_shown', and 'pos_overall'.

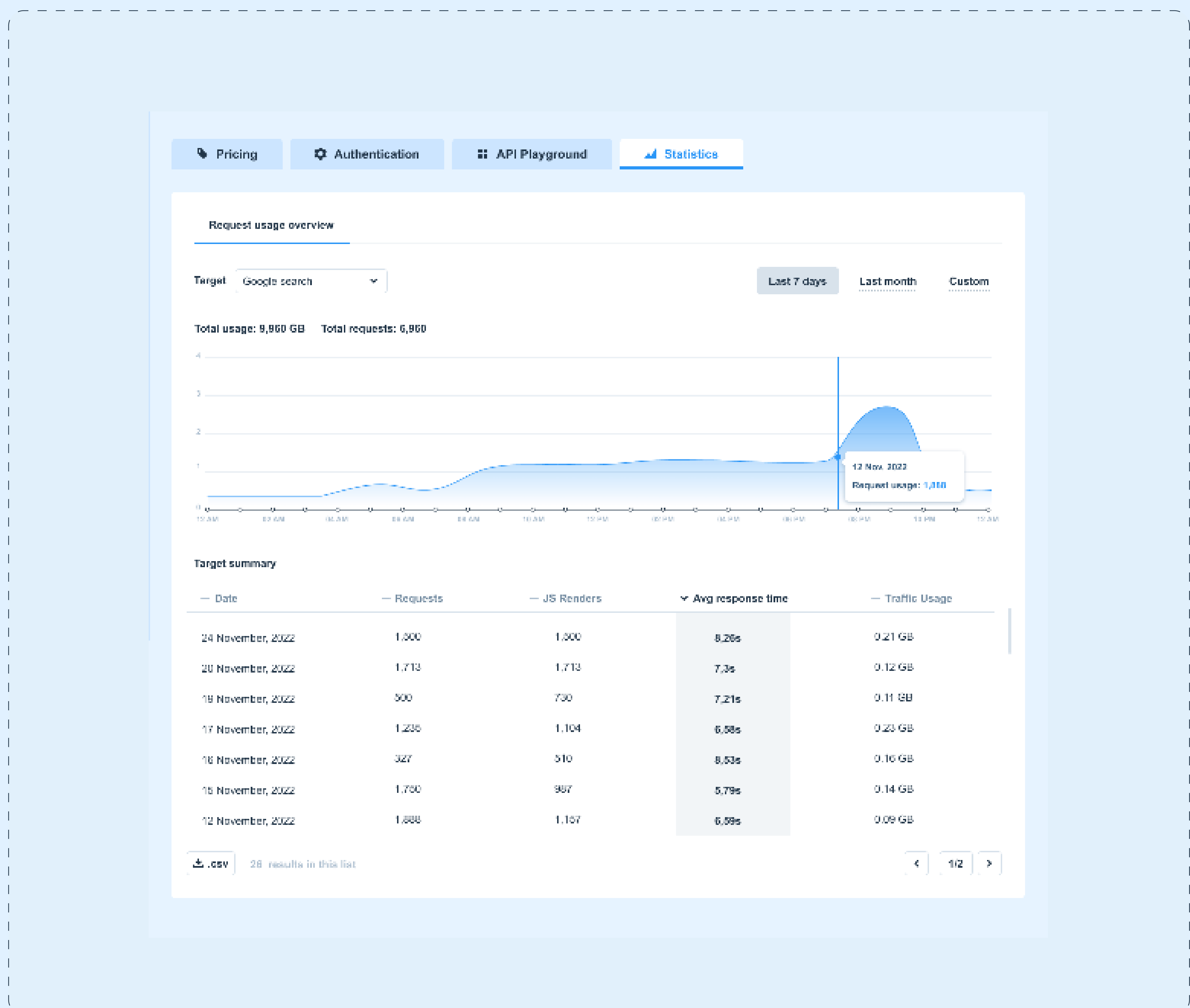
Try Scraping API playground

Continue to dashboard

Statistics

Statistics tab shows your request expenditure over time.

You can select a preset time period (such as week or month) or enter custom dates.



[Try Scraping API playground](#)

[Continue to dashboard](#)

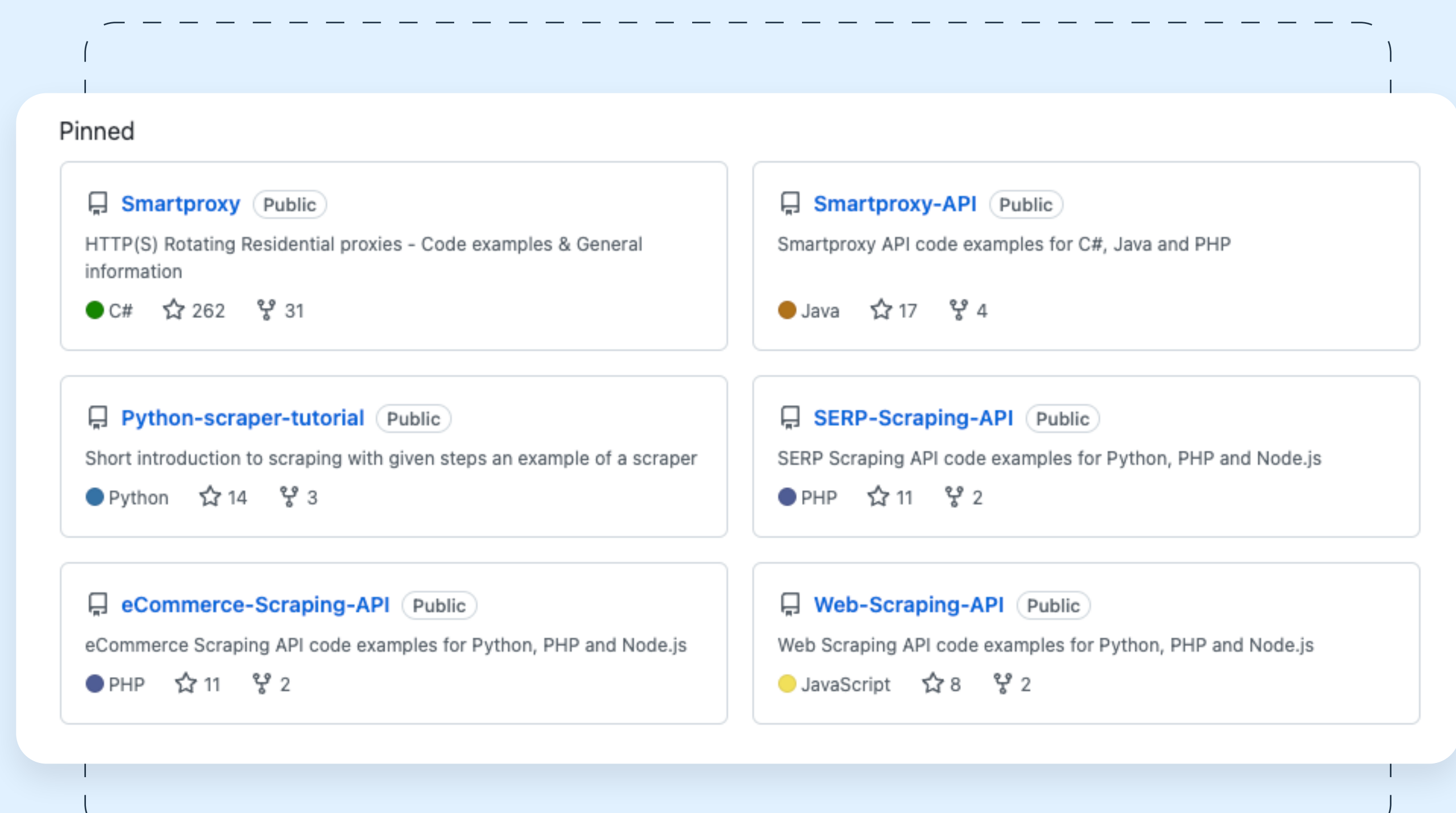
6.

Resources

The following resources can help you learn more about the implementation and functionality of the scraping APIs.

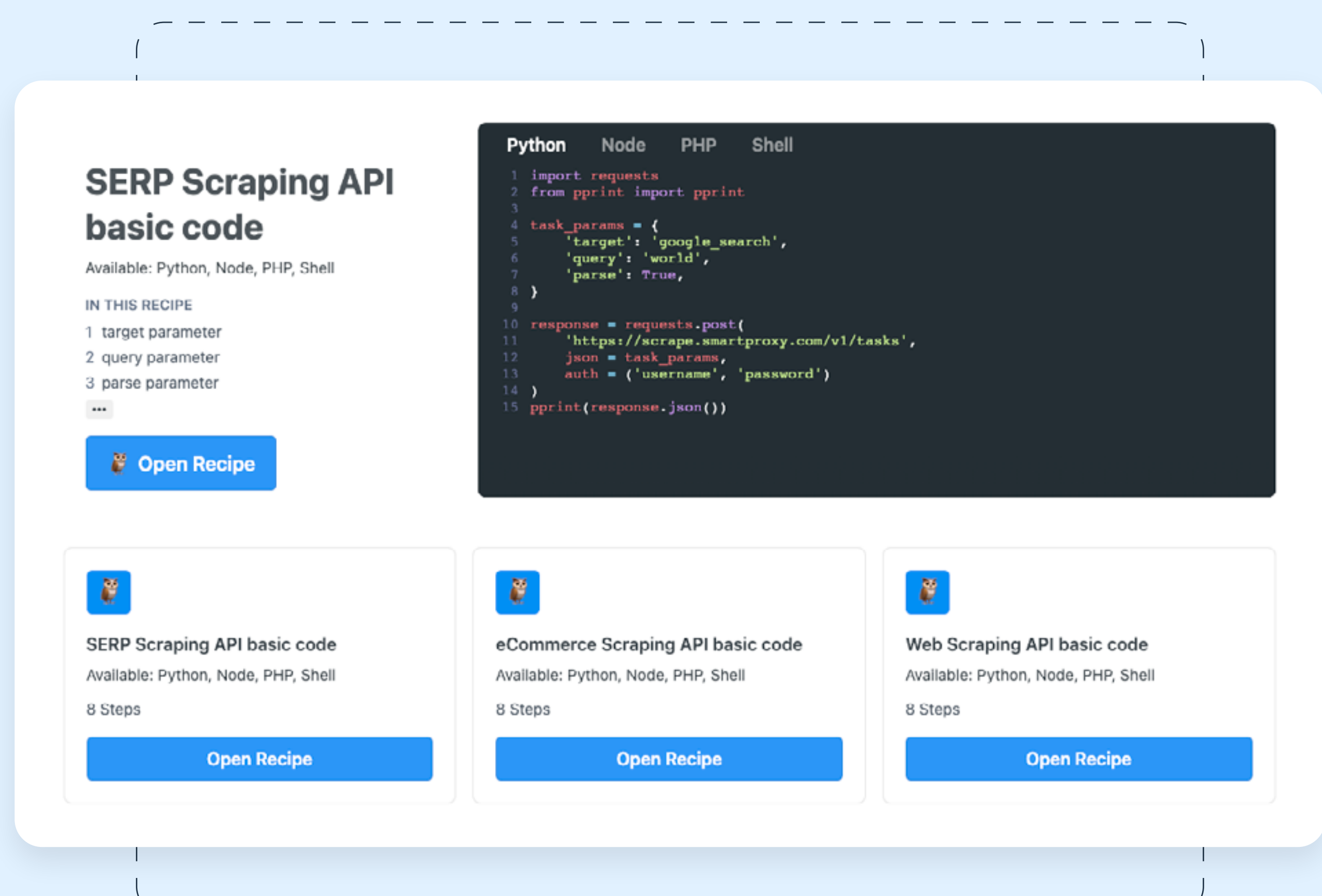
GitHub

Our [GitHub](#) includes detailed code samples for the most popular programming languages like Python, PHP, and Node.js.



Postman collections

You can also take a look at our [Postman recipes](#) which explain each API line by line.



7.

Conclusion

Smartproxy's web scraping APIs were designed to help you effortlessly gather web data at any scale. With the ability to target a wide range of locations, render JavaScript, and parse major search, social media, and e-commerce websites, they can make your data collection operations more efficient and predictable.

We hope that you've found this guide helpful. We'd love to talk to you about how Smartproxy's web scraping APIs can support your organization. You can book a call with us.

[Talk to our expert](#)

[Contact our support](#)