



Quickstart

Introduction

Yellowcake is an end-to-end data retrieval tool - think an evolved webscraper - that lets you retrieve information from any website with a URL and a prompt (which is a description of the content you want extracted). For more advanced functionality, like bypassing auth-walled websites, avoiding rate-limiting, stealth mode, etc - there are additional optional arguments you can provide. **The content in this page is all you'll need to use Yellowcake.**

When should I use Yellowcake?

Yellowcake is best used for when you have to *dynamically* fetch deeply-nestled and targeted content from specific websites, that may otherwise be hard to access or build custom code for - and where potentially longer latency (>5 minutes depending on request) is acceptable. Use cases like giving your agents a webfetch that doesn't suck, enriching already existing but potentially incomplete data, etc - are considered use cases that Yellowcake is great for.

What you probably shouldn't use Yellowcake for - is very large bulk data extraction within the same request, as not only will latency severely suffer, but cost will as well. (e,g - go to x.com and fetch EVERY single post - this is considered unsupported)

Prerequisites

- A Yellowcake API key (get one by registering at yellowcake.dev and generating an API key via the dashboard)
- Given Yellowcake is E2E, no installation is needed. The only requirement is a working API key
- A paid / upgraded plan IF you're using the `authorizedURLs` / `loginURL` optional parameters

Make your first request

The primary method to interact with the API is via POST, which both return SSE streams. You can do this two ways:

cURL (Direct POST Query)

```
curl --no-buffer -X POST https://api.yellowcake.dev/v1/extract-stream \
-H "Content-Type: application/json" \
-H "X-API-Key: YOUR-YELLOWCAKE-API-KEY" \
-d '{
  "url": "https://example.com",
  "prompt": "Extract all product names and prices"
}'
```

Javascript

```
async function extractStream(url, prompt) {
  const res = await fetch("https://api.yellowcake.dev/v1/extract-
stream", {
    method: "POST",
    headers: {
      "Content-Type": "application/json",
      "X-API-Key": "YOUR-YELLOWCAKE-API-KEY",
    },
    body: JSON.stringify({ url, prompt }),
  });

  return res.body; // ReadableStream (SSE)
}
```

Return Structuring

Progress is streamed via SSE, however complete and final payload is not streamed until end of the request. What this means is that while results might start streaming relatively quickly, they may be incomplete as the engine searches for the necessary data to complete them. Results are returned in JSON format with columns determined in a manner that matches your request. Here is an example payload, with request:

request

```
curl --no-buffer -X POST 'https://api.yellowcake.dev/extract-stream' \  
  -H 'Content-Type: application/json' \  
  -H 'Authorization: Bearer YELLOWCAKE-API-KEY' \  
  -d '{"url": "https://ycombinator.com", "prompt": "get 25 winter 2026 companies including company name and description"}'
```

result

```
event: complete  
data: {"success":true,"sessionId":"YOUR SESSION ID WILL BE HERE","data":  
  [{"company_name":"Chasi","description":"AI Concierge for Equipment sales,  
service & rentals"}, {"company_name":"o11","description":"The AI Platform  
for Capital Market Firms"}, {"company_name":"Bubble  
Lab","description":"Open-source, Typescript-native agentic workflow  
builder"}, {"company_name":"Pocket","description":"The world's best AI  
note-taking device and app"},  
{"company_name":"Constellation","description":"AI for satellite mission  
assurance."}, {"company_name":"AxionOrbital  
Space","description":"Foundation models for 24/7 Earth Observation"},  
{"company_name":"Caretta","description":"24/7 Sales Intelligence"},  
{"company_name":"Ressl AI","description":"AI Agents for ERP / CRM  
Configuration, starting with Salesforce"},  
{"company_name":"Patientdesk.ai","description":"AI voice agent for patient  
calls & admin workflows"}, {"company_name":"Prana","description":"An AI  
primary care doctor in your pocket"},  
{"company_name":"Luel","description":"Turning everyday words and actions  
into usable training data."},  
{"company_name":"Skillsync","description":"Find anyone in open source"},  
{"company_name":"MirageDoodle, Inc.","description":"AI-native workspace  
for municipal development review"},  
{"company_name":"21st","description":"Product design and prototyping tool  
for AI native teams"}, {"company_name":"Avoice","description":"The AI  
Workspace For Architects"}, {"company_name":"LegalOS","description":"The  
AI-powered immigration law firm for complex work visas"},  
{"company_name":"Sonarly","description":"Production-bug context for AI  
agents"}, {"company_name":"Didit","description":"AI-native, developer-first  
identity verification platform"},  
{"company_name":"Hlabs","description":"Developer friendly robotics"},  
{"company_name":"Valgo","description":"Algorithmic safety validation tools  
for autonomy."}, {"company_name":"Sarah AI","description":"AI Agents for
```

```
CPG ops"}, {"company_name": "Mantis Biotechnology", "description": "The  
Infrastructure Powering Human-In-Computer Models"},  
{"company_name": "Voltair", "description": "Self Charging Drones"},  
{"company_name": "Oxus", "description": "AI-powered automation for internal  
audit workflows"}, {"company_name": "Travo", "description": "We use AI to  
collect and analyze real estate data"}], "metadata":  
{"duration": 79323, "url": "https://ycombinator.com", "prompt": "get 25 winter  
2026 companies including company name and  
description", "itemCount": 25}, "timestamp": 1766344782885}
```

Parameters

url

This one is self explanatory - the target URL that you're seeking content from - the haystack.

prompt

The prompt is the heart of the request - the needle to locate. This determines not only the content extracted from the page, but the return format as well. While results are always in JSON, column/key names are inferred from the prompt. Moreover, the prompt also decides how deep the yellowcake algorithm goes in its search - i.e how many pages it navigates, when it decides it's gone far enough, etc. Hence, your prompt should be as detailed as reasonably possible, as poorly written prompts might yield poor results and waste your compute time.

throttle - optional

This is an *optional argument* that puts the algorithm into throttle (slow) mode. This is best for non-public auth websites, or websites prone to rate limiting. Speed is slightly impacted, but not extremely so. **NOTE: By default this is set to true when you provide a loginURL parameter. You CANNOT unthrottle an auth-walled request.**

loginURL - optional

This is an *optional argument* containing a URL that must be signed into, that **lets you mark a request as authwalled**. If this argument is not provided, the URL will be assumed to be public,

and not require an authentication. Keep an eye on SSE events as you will be requested to authenticate manually. We do NOT store any authentication credentials or data AT ALL, which means in the odd-case an auth expiry occurs, you might be required to sign in again. **NOTE: requests with a loginURL by default flag throttleMode as true and hence take longer.**

authorizedURLs - optional

This is an *optional argument* containing a list of URLs that are allowed to be navigated to. **This is how you'd introduce compliance and avoid websites you wouldn't want to navigate to.** When providing a url, the *base url* is considered fair game. So, for example docs.apple.com -> docs.apple.com/help/ is fair game, but NOT apple.com - we highly suggest that for auth-walled requests (i.e requests with a loginURL provided) that you provide a list of authorizedURLs because this can help prevent things like phishing, unintended data retrieval, etc.

MFA

Mutli-factor authentication can pose a challenge to most E2E solutions. There are a couple of ways you can get MFA to not be a burden with Yellowcake. The first, and simplest - disable MFA. Another option is manually entering MFA via the control link provided during initial authentication. For those more security conscious but wanting of an E2E solution, Yellowcake is also capable of monitoring email and SMS, but this is a custom feature you'll need to [contact](#) us for.

Disclaimer

ALWAYS check the Terms-Of-Service of the source you're trying to fetch data from before you make a request. Certain platforms and websites prohibit the use of automations which means that you may, if you for example make an auth request, find yourself in hot water. We hold no nor assume any responsibility for the misuse of this tool - it is YOUR duty to verify you're allowed to do what you wish to.

Next steps

- Make a request and have some fun! Yellowcake was designed to be easy to use, hence our docs (for now) are just this one pager.

- If you still find yourself confused, shoot us an email at hello@yellowcake.dev or join the [discord](#)