MCP SERVER | NO CODE | CLOUD HOSTED

ScraperAPI MCP

Extract Structured Data from Any Website.

ScraperAPI equips your AI agent with professional web scraping capabilities, letting it bypass IP bans and CAPTCHAs to extract data at scale. Use proxy rotation and headless browsers to reliably pull structured HTML from difficult sites like Amazon or Google SERPs, even when they use JavaScript rendering or aggressive anti-bot systems.

A+ Quality Score 100/100

proxy-rotation | headless-browser | captcha-solving | data-extraction | html-parsing | web-automation

MAINTAINED BY VINKIUS CLOUD FOR AI AGENTS · JUNE 2026
vinkius.com/mcp/scraperapi

SCAN TO VIEW

The connectivity layer between AI and the world's software.

Vinkius sits between AI and every application. All communication passes through Vinkius Cloud via the Model Context Protocol (MCP) — with governance, observability, and security at every layer.

Your AI Connections Run Through Vinkius Cloud

The world's largest managed MCP catalog

Vinkius is the connectivity layer where AI connects to the software your business already runs. We handle the hosting, the security, the credentials, the uptime — you get agents that actually do things.

We operate the world's largest managed MCP catalog. Major SaaS platforms, CRMs, databases, and cloud providers — running, monitored, production-ready. This MCP server is hosted and maintained by the Vinkius Cloud for AI Agents.

> *The agent doesn't manage credentials, doesn't manage uptime, doesn't manage security. Vinkius does.*
>
> — Architecture principle

Four Pillars of the Vinkius Runtime

01 — Security by design

Credentials stay encrypted at rest via AES-256. The AI agent never touches raw keys — they're injected into a sandboxed V8 isolate at runtime. Actions are logged, and connections have an emergency kill switch.

02 — Built on MCP Fusion

This MCP server was built with **MCP Fusion**, the open-source framework (Apache 2.0) that powers the entire Vinkius catalog. Schema-as-firewall strips undeclared fields, compiled PII redaction runs at zero overhead, and cryptographic lockfiles produce git-diffable audit trails.

03 — Deterministic observability

Eight immutable metrics per endpoint: request volume, p95 latency, error rate, active connections, cost attribution. A live payload feed logs every tool call with mutation detection.

04 — Autonomous operations

Servers are deployed, monitored, and patched autonomously. New capabilities and security patches ship weekly. Zero-downtime deployments ensure continuous availability across all managed MCP servers.

AES-256	Ed25519	24h TTL	V8 Isolate
Encryption at rest	PKI vault signatures	Ephemeral session keys	Sandboxed execution

One Token. Instant Access.

Every MCP server on Vinkius is accessed through a **Connection Token**. Tokens are generated in the cloud dashboard and produce a unique MCP endpoint URL. Paste this URL into any MCP-compatible client — no SDK required.

A single token can serve **multiple AI clients simultaneously**, or you can issue separate tokens per client for granular access control. Each token tracks its own request count, last activity timestamp, and can be individually enabled or revoked.

MCP ENDPOINT

```
https://edge.vinkius.com/{token}/mcp
```

Claude · Cursor · VS Code · Windsurf · Grok · Gemini

Security Is the Architecture

Security in Vinkius is not a feature — it's the foundation of the runtime. The gateway enforces multiple independent protection layers between AI agents and third-party APIs.

01 — Ed25519 PKI Vault

Every workspace has an Ed25519 Master Key. Session keys are generated ephemerally (24h TTL) and signed by the Master Key. Credentials never leave the vault boundary.

02 — V8 Isolate Sandboxing

Tool code runs inside isolated-vm V8 isolates with 64 MB memory caps and per-request timeouts. No filesystem access, no network access except through the SSRF-guarded fetch bridge.

03 — SSRF Guard

All outbound HTTP requests are DNS-resolved and validated before execution. Private IP ranges (10.x, 172.16-31.x, 192.168.x, AWS metadata 169.254.x) are blocked at the network layer.

04 — DLP & PII Redaction

A ResponseGuard pipeline intercepts every tool response. Configurable redaction patterns strip sensitive fields (emails, SSNs, card numbers) before data reaches the AI agent.

05 — Cryptographic Audit Trail

Every request is signed into a SHA-256 hash chain with Ed25519 signatures. Events form a tamper-proof, SIEM-exportable forensic record.

06 — Honeytoken Trap System

Phantom credentials are injected into isolated environments. If a honeytoken is used outside Vinkius infrastructure, the server is quarantined instantly.

Emergency Kill Switch

EU AI Act Art. 14(1) Compliant

The kill switch is an **emergency halt** mechanism — not a simple toggle. When triggered, it executes three actions atomically:

01 — Server deactivated

The MCP server is immediately taken offline across the entire cluster.

02 — All tokens revoked

Every connection token is invalidated. Total lockout — reconnection blocked until new tokens are issued.

03 — WebSocket connections killed

Active connections terminated via Redis pubsub broadcast. Propagates to every runtime node in the cluster.

Full Visibility. Zero Guesswork.

The Vinkius cloud dashboard includes a full MCP Governance suite — real-time analytics and security controls for production AI operations.

Control Plane

KPI dashboard with request volume, latency, success rate, token consumption, and AI-generated operational briefings.

Agent Activity

Which AI clients are connecting, how often, and what they're doing — real-time session tracking.

FinOps

Cost tracking per tool, payload compression savings, budget optimization signals, and consumption trends.

Tool Health

Slowest and most error-prone tools, with actionable root-cause insights and performance baselines.

Firewall & DLP

PII redaction activity, sensitive data protection counters, and security event timeline.

Incident Log

Error trends, failure rates, status-code breakdowns, and forensic audit trail access.

> Get started at **cloud.vinkius.com** — connect your AI agent in under 60 seconds.

ScraperAPI MCP

10 tools available
Cloud-hosted on Vinkius

You need your AI agent to do more than just read text; you need it to see the web as a data source. This MCP connects your agent to an industrial-grade scraping layer that handles all the messy infrastructure stuff—proxies, CAPTCHAs, and anti-bot systems. Instead of getting blocked when hitting major sites, your agent can pull structured product details from Amazon or get full Google search result layouts in clean JSON format. It's built for scale. Whether you're an analyst pulling competitor pricing across dozens of ASINs or a developer needing to render data from a Single Page Application (SPA), this lets your agent do the heavy lifting without you writing complex networking code. When you connect it via Vinkius, you just keep giving clear commands through your preferred AI client and let it manage the scraping process entirely.

Core Capabilities

01 — Extract structured e-commerce data

The agent retrieves formatted product details, pricing, and reviews directly from Amazon listings.

02 — Capture Google search results (SERPs)

You get the full structure of a Google search page, including featured snippets and organic rankings, in JSON format.

03 — Scrape dynamic web pages

The agent fetches data from modern sites built with JavaScript frameworks like React or Vue.

04 — Bypass anti-bot systems

The connection automatically rotates proxies and uses residential IP pools to avoid getting blocked by major websites.

05 — Run background scraping jobs

You can initiate massive data pulls that run in the background, keeping your conversation thread clean while waiting for results.

One Click on Vinkius — From Prompt to Execution

Available at **vinkius.com/mcp/scraperapi** — connect your AI agent in three steps.

01 Subscribe to this MCP and input your unique ScraperAPI key.

02 Your AI agent sends a command telling it which website or data point to scrape (e.g., 'Get the price for ASIN X').

03 The MCP executes the request, handles all proxy rotation and rendering, and delivers the clean, structured data back to your chat.

The bottom line is that you give a simple prompt, and the system gives you reliable, clean data from complicated websites.

Built For

Data Engineers who need massive archives of web content scraped; SEO Specialists tracking global keyword rankings; or Growth Analysts needing real-time competitor pricing.

SEO Specialist

Runs automated checks on Google SERPs across different regions to monitor how many rich snippets appear for a target keyword.

Growth Analyst

Pulls pricing and product metadata from competitor sites like Amazon, running these queries multiple times a day without manual intervention.

Data Engineer

Dispatches parallel background scraping jobs to pull large archives of public web data without slowing down the core workflow.

What Changes When You Connect

01 You get reliable data extraction even when facing high security. Use the `scrape_premium` tool to pull content using residential proxies, bypassing aggressive Cloudflare protection.

02 Stop struggling with dynamic sites. If a site uses JavaScript (like React), use `scrape_js_rendered` to force the browser to load all assets before pulling data, ensuring you get the full picture.

03 Handle massive data sets without clogging your conversation history. Use `create_async_job` to kick off long-running scrapes in the background and check status later with `get_async_job`.

04 Get specialized results for key platforms. Running `scrape_amazon` or `scrape_google_serp` gives you structured JSON output tailored specifically for e-commerce or SEO analysis, respectively.

05 Improve reliability across all tasks by using the basic `scrape_html` tool, which automatically manages proxy rotation to keep your IP address clean and active.

Real-World Applications

Tracking competitor price changes daily

A growth analyst needs to know if a rival changed their Amazon Buy Box pricing. They tell their agent: 'Run `scrape_amazon` for ASIN X.' The system returns structured data, letting the analyst immediately compare it to yesterday's record.

Analyzing keyword trends across regions

An SEO specialist wants to see how Google ranks a term in Japan vs. Germany. They prompt their agent to use `scrape_google_serp` for both locations, getting two separate JSON outputs to compare organic positioning.

Capturing complex website layouts

A developer needs a visual reference of a site's current design before building a scraper. They use `get_screenshot_link` to instantly pull a high-res PNG capture, bypassing the need for manual browser checks.

Pulling data from slow or restricted sites

An engineer is trying to scrape an internal dashboard that only loads content after complex scripts run. They use `scrape_js_rendered` via the agent, ensuring the AI gets the fully loaded and correct view.

Patterns to Avoid

Treating scraping like a simple read operation

✗ AVOID

Asking your agent to scrape a URL with a standard prompt. The system might fail or only get basic HTML because the site loads content via JavaScript after initial page load.

✓ INSTEAD

If the website is modern, you must use `scrape_js_rendered` to force the AI client to execute all necessary scripts before data extraction. For maximum stability, try `scrape_premium`.

Running multiple complex scrapes at once

✗ AVOID

Trying to scrape ten different pages in one massive prompt block. This risks overwhelming your rate limits or making the conversation thread slow and unpredictable.

✓ INSTEAD

Use `create_async_job` first. Kick off all 10 scraping tasks as background jobs, then use `get_async_job` later when you're ready to check results. It keeps the flow clean.

Needing specialized data formats

✗ AVOID

Asking for general HTML scrape (`scrape_html`) when you actually need Amazon product details or Google SERP structure.

✓ INSTEAD

Use the dedicated tools. Run `scrape_amazon` if it's a marketplace item, or run `scrape_google_serp` if you're tracking search results. The output will be structured and ready to use.

The Right Fit

Use this MCP if your task involves extracting data from the live internet—meaning you are dealing with websites that block basic bots or require JavaScript execution. Don't use it if you only need to read a simple, static text file or access a clean API endpoint (in which case, an API connector is better). If you need structured e-commerce data, stick to `scrape_amazon`. If you are scraping the results of search engines, use `scrape_google_serp`. If the target site is difficult and prone to blocking, always prioritize using `scrape_premium` or initiating a background job with `create_async_job` so your agent doesn't hang up while waiting for proxies.

Web data extraction used to be slow, fragile, and expensive.

Today, getting accurate web data means copy-pasting results from a handful of tabs. You open Amazon, scrape the price; then you switch to Google and run a separate search tool; after that, you have to check if the site is running JavaScript, which often breaks your initial scrape. It's manual, it takes hours, and it costs time.

With this MCP, your agent handles all of that complexity in one go. You just tell it what data you need—say, product metadata for a list of ASINs. The system manages the proxies, renders the JavaScript, and returns clean JSON arrays, allowing your workflow to continue instantly.

ScraperAPI gives you structured Amazon data.

Before this MCP, getting product details meant a laborious process of visiting individual URLs and manually compiling the price, rating, and stock status. If one site changed its layout slightly, your entire spreadsheet broke.

Now, running `scrape_amazon` feeds structured data directly into your agent's context window. The output isn't just text; it's organized fields that you can immediately process or load into a database.

ScraperAPI: 10 Tools for Web Scraping

These tools allow your agent to perform every kind of structured data extraction, from Amazon product pages to complex JavaScript-rendered sites.

#	TOOL	DESCRIPTION
01	`create_async_job`	Starts a scrape job that runs in the background and returns a tracking ID.
02	`custom_scrape`	Allows you to perform a general web scrape using specific, advanced parameters.
03	`get_account_stats`	Retrieves current usage statistics for your scraping account.
04	`get_async_job`	Checks the status and retrieves the final result of a previously started background job.
05	`get_screenshot_link`	Generates a temporary URL that captures a full-page screenshot image.
06	`scrape_amazon`	Retrieves structured product details, pricing, and metadata specifically from Amazon listings.
07	`scrape_google_serp`	Fetches the full structure of a Google search results page for analysis.
08	`scrape_html`	Scrapes standard static HTML content from any given URL while automatically rotating IPs.
09	`scrape_js_rendered`	Handles complex URLs by running JavaScript code to make sure all data loads before scraping.
10	`scrape_premium`	Scrapes a URL using high-quality residential proxies for maximum bypass capability.

See It in Action

Real prompts you can use once this MCP is connected to your AI agent through Vinkius Cloud.

U Scrape an Amazon product page for this ASIN: B08J5F3G18 and list its price.

I've fetched the Amazon structured data. The ASIN B08J5F3G18 is the 'Apple iMac Desktop'. The current Buy Box price is $1,299.00 and it holds a rating of 4.7 with over 4,000 reviews. It's 'In Stock'.

U Run a Google SERP check for the keyword 'best LLM orchestration frameworks'.

I've scraped the Google search results for you. LangChain occupies the first organic spot. LlamaIndex holds the featured snippet, and emerging frameworks like Vurb and Semantic Kernel are dominating the 'People Also Ask' sections. Would you like me to dump the full organic ranking array?

U Take a screenshot of https://netflix.com homepage so I can check its layout.

I've invoked the screenshot endpoint. You can view the full rendered layout capture generated by ScraperAPI here: `https://api.scraperapi.com/v1/screenshots?...`. It successfully bypassed the initial bot-checks saving a high-resolution PNG.

Frequently Asked Questions

01 **How do I use ScraperAPI MCP to scrape JavaScript sites?**

You must use the `scrape_js_rendered` tool. This tells your agent that the site is dynamic and needs a full browser render before scraping can begin, which captures all loaded content.

02 **Is ScraperAPI MCP better than basic web scraping tools?**

Yes. Basic tools often fail when websites detect bots or use modern JavaScript frameworks. This MCP uses proxy rotation and headless browsers to ensure successful data extraction at scale.

03 **How do I scrape multiple pages without interrupting my chat flow using ScraperAPI MCP?**

Start by calling `create_async_job`. This spins up the scraping task in the background. You then use `get_async_job` later to retrieve results without waiting live.

04 **Can I get structured data from Google Search using ScraperAPI MCP?**

Absolutely. Use the dedicated `scrape_google_serp` tool. It pulls search results and structures them into JSON, giving you much more than just a raw list of links.

05 **What is the difference between scrape_html and scrape_premium?**

`scrape_html` handles standard scraping with basic proxy rotation. `scrape_premium`, however, uses high-quality residential proxies, offering a much higher chance of success when hitting heavily protected targets.

Go Live in 60 Seconds

Get your connection token from **cloud.vinkius.com**, then paste the endpoint URL into any MCP-compatible client.

YOUR MCP ENDPOINT

```
https://edge.vinkius.com/[TOKEN]/mcp
```

CLIENT	WHERE TO CONFIGURE
Claude AI	Profile → Customize → Connectors → "+" → Add custom connector → Paste endpoint
Cursor	Settings → Features → MCP Servers → "+ Add New MCP Server" → Type: SSE → Paste endpoint
VS Code	Ctrl/Cmd+Shift+P → "MCP: Add Server" → add `"scraperapi": { "url": "…" }`
Windsurf	MCP Settings → `mcp_settings.json` → Add endpoint URL
ChatGPT	Settings → Tools & plugins → Add MCP server → Paste endpoint
Gemini	Extensions → Add MCP Server → Paste endpoint URL

ASK AN AI ABOUT THIS

Let your preferred AI explain this MCP server

- Ask ChatGPT
- Ask Claude
- Ask Perplexity
- Ask Gemini
- Ask Grok

READY TO CONNECT

ScraperAPI is live on Vinkius Cloud.

Get your connection token, paste it into your AI agent, and start building. No SDK. No deployment. Just results.

Start at cloud.vinkius.com →

vinkius.com · support@vinkius.com

INDEPENDENT PLATFORM DISCLAIMER

Vinkius is an independent platform and is not affiliated with, endorsed by, sponsored by, verified by, or otherwise authorized by ScraperAPI. All third-party trademarks, logos, and brand names are the property of their respective owners. Their use in this document is strictly for informational purposes to identify service compatibility and interoperability.

DOCUMENT INFORMATION

Generated	June 2026
MCP Server	ScraperAPI MCP
Server ID	`019d7604-7810-7019-9fed-3bbd6bca7844`
Platform	Vinkius Cloud for AI Agents
Endpoint	`https://edge.vinkius.com/{token}/mcp`

LICENSE & USAGE

This document is generated automatically by the Vinkius PDF Engine. Content reflects the MCP server configuration at the time of generation and may change as updates are deployed. For the most current information, visit vinkius.com/mcp/scraperapi.