MCP SERVER | NO CODE | CLOUD HOSTED

Swiftype MCP

Programmatic Search and Index Management

Swiftype connects your AI client directly to enterprise search engines, letting you manage complex indexing logic without touching a web dashboard. It lets your agent run precise queries against core data sources, list all available indexes, track top user clicks, and even programmatically create or delete entire sets of documents.

A+ Quality Score 100/100

enterprise-search | site-search | indexing | search-api | data-retrieval

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

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.

Swiftype MCP

10 tools available
Cloud-hosted on Vinkius

Swiftype gives your conversational AI the ability to act as an internal data architect for your organization's search infrastructure. Instead of opening a dozen different web portals just to check indexing status or run analytics, you let your agent interact directly with the core Swiftype endpoints. Your client can list every active index and scope, ensuring you know exactly what data exists. It handles everything from running raw queries against specific engines to tracking real-world user behavior by retrieving top searched terms and identifying high-conversion click paths. This capability moves search administration out of clunky web forms and straight into your prompt environment. By utilizing the Vinkius catalog within your agent, you gain secure, programmatic access to read, write, and audit structured data maps on demand.

Core Capabilities

01 — Inspect Search Infrastructure

List all available search engines and map schema blueprints to understand what data sources are connected.

02 — Execute Targeted Searches

Fire complex, raw queries into specific content engines and pull back structured JSON results for immediate analysis.

03 — Manage Content Records

Bulk create new documents or permanently delete stale records to keep your index clean and accurate.

04 — Analyze User Behavior

Retrieve concrete metrics on what users are searching for most often, and which specific pages they click through after a search.

05 — Validate Data Integrity

Extract stored metadata to audit document types and ensure your data map adheres to established rules.

One Click on Vinkius — From Prompt to Execution

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

01 You attach the Swiftype MCP to your AI client through Vinkius, specifying your API credentials for secure access.

02 Your agent then uses specific commands—like listing available engines or running a search query—to gather necessary data points.

03 The results return structured JSON data directly into your chat window, allowing your agent to summarize complex findings like top clicks and document counts.

The bottom line is that you get full, programmatic control over the entire lifecycle of your enterprise search index, without ever needing a web login.

Built For

This MCP is for technical roles and data owners who spend time validating complex information systems. If you're tired of clicking through multiple dashboards or writing boilerplate API calls just to check index status, this is for you.

Search Architect

Using the MCP, they can programmatically audit and validate structured data maps, ensuring new content adheres to strict indexing rules.

Content Strategist

They use it to pull real-world analytics, identifying exactly which search queries lead to high-conversion clicks, guiding content creation efforts.

Data Reliability Analyst

This role uses the MCP to programmatically test and audit isolated data maps, finding index bugs or schema mismatches before they break production.

What Changes When You Connect

01 You get immediate visibility into site performance. Instead of manually checking a web dashboard, your agent runs `st.analytics_top_searches` to tell you what users are searching for right now.

02 Manage data integrity on the fly. Need to update records? You can use `st.create_documents` or `st.delete_documents` directly through conversation, keeping your index clean.

03 Understand your search scope instantly. Use `st.list_engines` to see all available isolated indexes and confirm that every required data source is connected.

04 Deep dive into user conversion paths. Instead of guessing, the MCP executes `st.analytics_top_clicks` to pinpoint exactly which documents drive actual traffic.

05 Audit your content structure easily. Running `st.list_doc_types` provides a schema blueprint view, ensuring new content always fits your data map rules.

Real-World Applications

Debugging Missing Content

A Data Analyst notices search results are missing documentation for the 'billing' section. They ask their agent to first run `st.list_engines` to confirm all relevant indexes exist, then use `st.list_doc_types` to check if a billing schema was ever defined. This quickly diagnoses whether the problem is connectivity or definition.

Optimizing Marketing Copy

A Content Manager wants to know which product pages perform best. They ask their agent to run `st.analytics_top_clicks` and instantly see a list of the top-performing pages, allowing them to rewrite titles for better SEO.

Auditing Index Changes

A Search Architect needs to prove that old, obsolete policy documents are gone. They ask their agent to use `st.delete_documents` on a specific set of records and confirm the action using the API wrapper.

Testing New Keywords

A developer wants to test if a new product keyword works well without committing to a change. They ask their agent to run `st.post_suggest` on the internal wiki engine, getting instant feedback on relevant completions.

Patterns to Avoid

Over-reliance on Guesswork

✗ AVOID

Trying to search for data that might exist in an index but isn't correctly defined or connected. This results in vague, unactionable error messages.

✓ INSTEAD

Always start by checking the available scope. Use `st.list_engines` first to confirm the engine exists, then use `st.list_doc_types` to validate the required schema before running a search query with `st.post_search`.

Treating Data as Black Box

✗ AVOID

Manually exporting data and trying to figure out which records are outdated or need cleanup, wasting hours on Excel sheets.

✓ INSTEAD

Use the tools to make the system tell you what's wrong. Run `st.list_documents` to see all stored metadata, then use that list to programmatically identify keys for removal using `st.delete_documents`.

Writing monolithic queries

✗ AVOID

Attempting to extract every single piece of data in one giant query, which often fails due to scope limits or complexity.

✓ INSTEAD

Break down the task using specific tools. First, use `st.list_domains` to verify crawler limits, then run smaller, targeted queries with `st.post_search` for distinct pieces of information.

The Right Fit

Use this MCP if your core problem is programmatic control over a complex, multi-faceted indexing pipeline. You need to audit schemas (`st.list_doc_types`), manage content lifecycle (using `st.create_documents` and `st.delete_documents`), or pull deep analytics beyond basic search results. Don't use this if you just need simple text retrieval; a standard document storage connector will

suffice. Also, don't use it if your only goal is to read one specific field from one single record; those tools are overkill. This MCP is for the system administrators and power users who manage how data *gets* into the search engine and what metrics that data generates.

Tired of Web Dashboards?

Today, checking your site's performance means navigating to the analytics portal. You click through tabs labeled 'Searches,' then another tab for 'Clicks.' If you want to know which search term led to a high-value page conversion, you have to pull data from three separate reports and manually cross-reference timestamps—it's tedious, slow, and prone to human error.

With this MCP, your agent handles that entire process in a single conversation. You simply ask it to correlate user intent with performance metrics. It runs the necessary sequence of commands, like `st.analytics_top_searches` followed by `st.analytics_top_clicks`, and gives you one clean answer.

Swiftype: Direct Access to Your Search Data

The biggest pain point today is that data lives behind complex APIs and proprietary web interfaces. To audit the structure, you might have to guess which endpoints are correct or spend hours calling internal teams for schema documentation.

Now, your agent directly interrogates the core system using tools like `st.list_doc_types` and `st.list_engines`. You don't ask a person; you talk to the database structure itself. It's immediate, verifiable, and accurate.

Swiftype: 10 Tools for Search Management

These tools give your agent the power to query, write, read, and audit every aspect of your enterprise search index.

#	TOOL	DESCRIPTION
01	st.create_documents	Adds new structured records, such as billing information, to your active data map.
02	st.delete_documents	Permanently removes cached pages and associated metrics from the search index.
03	st.list_documents	Retrieves a list of all stored metadata, tracking IDs for every document type in your system.
04	st.list_domains	Verifies the limits set by the automated crawler across specific index scopes.
05	st.list_engines	Identifies all active, isolated search engines bound to your platform tenant.
06	st.post_search	Runs a raw query against a specific engine and returns detailed JSON results including property extraction.
07	st.post_suggest	Provides predictive, spelling-tolerant suggestions for users without querying the main index.
08	st.analytics_top_clicks	Finds and reports on the precise pages that receive the most clicks after a search.
09	st.analytics_top_searches	Inspects internal data to discover which keywords are being searched for across your entire site.
10	st.list_doc_types	Extracts the schema blueprints, showing exactly what types of data maps are supported by the system.

See It in Action

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

U List all my available Swiftype search engines, then run a search for 'documentation' on the most relevant one and show me the top 3 analytics clicks it generated last week.

Connecting natively to Swiftype... I found 2 engines: 'prod-store-engine' and 'internal-docs'.
Running `st.post_search` on 'internal-docs' explicitly for 'documentation'.
Found 15 documents. Now fetching CTR metrics (`st.analytics_top_clicks`)...

Top Converging Documentation Documents:
1. '/docs/api/endpoints' - 145 Clicks
2. '/help/getting-started' - 93 Clicks
3. '/support/auth-faqs' - 42 Clicks

U List all active engines in our Swiftype account.

Scanning Elastic instances ('list_engines')...
Found 3 active engines:
1. 'Global-Help-Center' (Language: EN)
2. 'Ecommerce-Catalog-US' (Language: EN)
3. 'Internal-Company-Wiki' (Language: EN)

Let me know if you want me to index new documents into any of these.

U Run a test suggestion for 'passw' in the internal wiki engine.

Executing `post_suggest` constraint on 'Internal-Company-Wiki'...
The engine suggests the following completions based on hits:
1. 'password reset policy'
2. 'password manager installation'
3. 'passport visa procedures'

Frequently Asked Questions

01 **How do I check what search engines Swiftype can access?**

You run `st.list_engines`. This tool extracts a list of every active index bound to your account, giving you an immediate overview of all available data sources.

02 **Can I use st.post_search for simple keyword searches?**

Yes, `st.post_search` fires raw queries into a specific engine and returns detailed JSON results. This means you can ask complex questions that go beyond just matching keywords.

03 **What is the difference between st.analytics_top_searches and st.analytics_top_clicks?**

`st.analytics_top_searches` tells you what people are looking for, while `st.analytics_top_clicks` tells you which documents they actually clicked on after searching.

04 **How do I add new data to my index using Swiftype?**

You use the `st.create_documents` tool. This allows your agent to inject new, structured payload data into the system in bulk without you needing to upload a file.

05 **Does Swiftype MCP help me audit old records?**

Yes. You can use `st.list_documents` to dump all stored metadata and track IDs, helping you identify which documents are stale and need cleaning up or deletion using `st.delete_documents`.

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 `"swiftype": { "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

Swiftype 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 Swiftype. 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	Swiftype MCP
Server ID	`019d760f-aa0f-71da-9603-6e7eee2418d3`
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/swiftype.