MCP SERVER | NO CODE | CLOUD HOSTED

Markdown Frontmatter Harvester MCP

Query Metadata Across Your Entire Local Vault

Markdown Frontmatter Harvester indexes your local knowledge base by scanning Obsidian or Hugo vaults and extracting all YAML metadata into a single, queryable JSON file. It lets your AI agent instantly read tags, dates, statuses, and other notes' hidden data without needing to search thousands of scattered markdown files.

A+ Quality Score 100/100

yaml-parsing | metadata-extraction | markdown | obsidian | vault-management | structured-data

MAINTAINED BY VINKIUS CLOUD FOR AI AGENTS · JUNE 2026
vinkius.com/mcp/markdown-frontmatter-harvester

SCAN TO VIEW

The infrastructure that powers AI agents in the real world.

Vinkius connects AI to the world's software through secure, enterprise-grade infrastructure — enabling real-world execution at scale, built on the Model Context Protocol (MCP).

Your AI Connections Run Through Vinkius Cloud

The world's largest managed MCP catalog

Vinkius is the cloud infrastructure where AI agents connect 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.

Markdown Frontmatter Harvester MCP

1 tools available
Cloud-hosted on Vinkius

Writing with digital notes means using tools like Obsidian or Hugo, which rely on YAML 'frontmatter'—those little blocks at the top of a file that hold metadata like `status: draft` or `tags: [idea]`. When your AI client asks, 'Which posts are marked as drafts from 2024?', it usually fails because it can't quickly index every single local markdown file. This MCP fixes that. It acts like a hyper-fast librarian, recursively scanning your entire folder structure and stripping out only the YAML frontmatter from every document. The result is a clean JSON index of your whole vault. Your agent gets one structured data set it can filter, sort, and query instantly, giving you reliable answers about your scattered notes.

Core Capabilities

01 — Index entire vaults

The MCP scans massive local directories to build a unified index of metadata from all contained markdown files.

02 — Extract specific fields

It pulls out named data points like tags, dates, and status markers written in YAML frontmatter.

03 — Query structured data

Your agent queries the generated JSON index directly, allowing precise filtering across thousands of documents at once.

One Click on Vinkius — From Prompt to Execution

Available at **vinkius.com/mcp/markdown-frontmatter-harvester** — connect your AI agent in three steps.

01 You provide the MCP with the absolute path to your entire notes folder or vault.

02 The tool scans every markdown file in that directory and extracts all the YAML frontmatter data it finds, ignoring the body text.

03 It returns a single, unified JSON object containing metadata for every file found, which your AI client can then use for querying.

The bottom line is you get one clean, actionable index of your entire knowledge base instead of thousands of individual files.

Built For

This MCP is essential for researchers, technical writers, and content managers who rely on structured notes. If you're tired of asking your AI client questions that fail because it can only read the visible text in a single file, this tool gives you the metadata context you need.

Technical Writer

Uses it to quickly find all articles marked with a specific status or tag across multiple drafts before publishing.

Researcher / Academic

Leverages it to count how many notes touch upon a certain year or topic, allowing them to track research progress over time.

Content Strategist

Uses it to audit an entire blog repository and identify every piece of content that hasn't been updated since a specific date.

What Changes When You Connect

01 Instant Querying: Instead of manually searching file names or using complex local scripts, your agent queries a unified JSON index. You get immediate answers about metadata like tags and status.

02 Massive Scale: It scans 1,000+ files in milliseconds, making it practical for large Obsidian vaults without slowing down your AI client's response time.

03 Data Structure: The output is clean YAML frontmatter converted into structured JSON. This format is ideal for any agent to consume and reason over.

04 Air-Gapped Security: Your private journal entries and business notes never leave your machine; the processing happens locally, maintaining 100% privacy.

05 Zero Setup: You don't need complex coding or configuration files. Just point the MCP at your root folder, and it does the rest.

Real-World Applications

Finding all outdated drafts

A content manager needs to know which blog posts were created before 2023 but still have a 'status: draft' tag. They simply ask their agent, and the MCP uses `harvest_markdown_frontmatter` to generate an index, allowing the AI client to list every file that meets both criteria.

Auditing research topics

A researcher wants to count how many notes they wrote in 2024 about 'quantum computing' based on the date and tag fields. The agent runs `harvest_markdown_frontmatter` against their vault path, generating a clean dataset that lets the AI client perform an accurate count.

Listing urgent items

A student asks to see every note marked with the 'urgent' tag across three different subfolders. The agent runs `harvest_markdown_frontmatter` on the parent directory, providing a single index that lets it pull all relevant file names instantly.

Patterns to Avoid

Passing raw folders to AI

✗ AVOID

Asking your agent to 'read my notes folder' and expecting it to magically know the metadata structure, or relying on a simple search function that only reads visible text.

✓ INSTEAD

You must use the `harvest_markdown_frontmatter` tool. By providing the absolute directory path, you force the MCP to index the YAML data first, giving your agent structured access to tags and dates.

The Right Fit

Use this MCP if your problem is about *metadata*—if you need to filter based on things like `status: draft`, or count notes by a specific `date` field. This isn't for simple text searching; it's for structured querying of hidden data.

Don't use this if all you want is to search the actual words inside your documents, like finding every instance of 'project failure'. For that, you just need standard file reading capabilities. You only need this MCP when you have a knowledge base full of files and you need to query their *properties*, not their content.

The Metadata Black Box Problem

Today, if your notes live in Obsidian or Hugo, the important details—like whether a note is marked 'draft' or belongs to the 'research' tag—are tucked away in YAML frontmatter blocks at the top of files. When you try to ask an AI client about these details, it often fails because it can't read thousands of local files quickly enough to build a complete picture.

With this MCP, your agent sees the full story. It scans the whole folder and rips out only that hidden metadata, packaging it into one clean JSON index. Your AI gets a unified database view, letting you ask sophisticated questions about your entire archive without any guesswork.

Markdown Frontmatter Harvester gives you structured data access

Manually auditing notes means opening folders, searching file names, and copy-pasting tags or dates into a spreadsheet just to count items. It's slow, prone to error, and terrible for large vaults.

Now you give the MCP the path, and it handles all the indexing work instantly. You get back a single JSON index that makes querying your entire knowledge base fast, reliable, and simple.

Markdown Frontmatter Harvester: 1 Tool

Use the available tool to scan your entire notes directory and create an instant, queryable index of all metadata found in your markdown files.

#	TOOL	DESCRIPTION
01	`harvest_markdown_frontmatter`	Provide the absolute directory path to scan local Markdown files and extract all YAML tags, dates, and metadata into an index.

See It in Action

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

U Scan my Obsidian vault at C:/Notes and list all files that have the tag 'urgent'.

I found 5 notes with the tag 'urgent'. Here are their filenames: ProjectX.md, MeetingNotes.md...

U Harvest the frontmatter from my blog repo and tell me which posts are still marked as 'status: draft'.

Based on the frontmatter, you have 12 posts still marked as 'draft'. Would you like the list?

U Count how many notes I created in the year 2023 based on the YAML 'date' field.

According to the metadata, you created exactly 142 notes in 2023.

Frequently Asked Questions

01 **How does Markdown Frontmatter Harvester read my local Obsidian vault?**

The MCP scans the absolute directory path you provide. It specifically targets YAML frontmatter blocks within markdown files to extract tags, dates, and status markers.

02 **Is this tool private or does it upload my notes?**

No, it's entirely air-gapped. Your journal entries and business notes never leave your machine; the processing happens locally on your system for maximum privacy.

03 **What file types can harvest_markdown_frontmatter handle?**

It is designed to scan Markdown files (like those used in Obsidian or Hugo) and extract the YAML frontmatter contained within them.

04 **Does this MCP read the body text of my notes?**

No, it only reads the metadata. It extracts the structured YAML data at the top of the file; the actual content of your note is ignored during indexing.

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 `"markdown-frontmatter-harvester": { "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

Markdown Frontmatter Harvester 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 Markdown Frontmatter Harvester. 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	Markdown Frontmatter Harvester MCP
Server ID	`019e38bc-5d28-737c-95e1-1dd334257389`
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/markdown-frontmatter-harvester.