MCP SERVER | NO CODE | CLOUD HOSTED

Bear MCP for AI Agents

Organize and retrieve markdown notes by tag or keyword

Bear MCP lets your AI agent manage your entire local markdown knowledge base. Instead of opening Bear and manually searching for notes or tags, you simply ask your AI client to find, edit, organize, or archive any piece of saved content—whether it's a code snippet from years ago or research notes.

F Quality Score 3.6/100

markdown | personal-knowledge-base | document-management | offline-storage | text-editing

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

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.

Bear MCP

10 tools available
Cloud-hosted on Vinkius

Your personal knowledge vault is in Bear App, but accessing it used to mean context switching and manual searching. This connector links your private local markdown data directly to any AI agent, letting you treat your entire archive like one big searchable document.

It handles everything from finding specific pieces of text across thousands of notes to restructuring your tags or creating entirely new drafts based on fragmented thoughts. If you're already using an advanced AI client and want it to actually *do* something with the massive amounts of writing, research, and code snippets you collect, this is it. Connecting through Vinkius means you get access to this Bear integration right alongside thousands of other tools your agent can use.

Think of it like having a hyper-efficient librarian who knows every corner of your digital filing cabinet and doesn't need you to tell them where to look.

Core Capabilities

01 — Search across all notes

Quickly finds specific information or topics mentioned in any Bear note.

02 — Read full markdown content

Pulls the complete, raw text and formatting of a selected Bear note for review.

03 — Create new knowledge items

Generates and saves brand-new notes directly into your local Bear App vault.

04 — Edit existing content

Adds or changes text blocks within a note without you having to copy and paste manually.

05 — Manage the archive and trash

Moves old notes out of circulation, either into the Archive or permanently deleting them.

06 — Structure and modify tags

Lists tag hierarchies, finds specific notes by tag, renames a tag across all relevant documents, or deletes an entire tag constraint.

One Click on Vinkius — From Prompt to Execution

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

01 Subscribe to this MCP and provide your Bear API Token. This lets the AI client talk directly to your local instance.

02 Use your preferred agent (like Claude or Cursor) to give a natural language command, such as 'Find all notes tagged #todo related to Q1'.

03 The agent executes the query against your vault and presents the results—whether that's opening a full note for review or confirming a tag was successfully renamed across thousands of items.

The bottom line is, you tell the AI what you need done with your knowledge base, and it handles the connection details to execute the action.

Built For

Anyone whose work depends on managing a large volume of personal documentation—especially writers, developers, or researchers. If you're tired of manually jumping between search fields and copy-pasting ideas across different projects, this MCP is for you.

Technical Writer

Needs to pull together disparate research notes, organize them by nested tags, and assemble a cohesive first draft without ever leaving their writing environment.

Software Developer

Relies on the MCP to search old configuration blocks or technical specifications saved in markdown notes, injecting raw code snippets directly into their active coding session.

Academic Researcher

Manages hundreds of source documents and research findings. They use this tool to aggregate all material related to a specific theory or author using tag searches.

What Changes When You Connect

01 You don't lose focus when your agent handles the research. Instead of manually opening Bear to find old snippets, you just ask it to pull them up for context.

02 Managing tags becomes instant. If you need to rename a project tag across hundreds of notes, the `rename_tag` tool updates everything globally in one step.

03 Writing drafts is faster because the agent can inject raw saved content directly into your document using `add_text`, eliminating copy/pasting friction.

04 Your archive stays clean. You use the MCP to automatically move outdated research into the Archive or Trash, keeping only actionable items visible.

05 The system gives you a full picture of your knowledge structure by listing all tags and their relationships via `list_tags` before you even start writing.

Real-World Applications

Finding an old code snippet for a client meeting

A developer needs to reference a specific API implementation from last year. They ask the agent to search their notes for 'API endpoint /user'. The tool uses `search_notes` and returns the relevant document, which they then open using `open_note` to pull the exact code block.

Cleaning up a massive research folder

A researcher finishes a topic. They ask the agent to identify all notes related to 'Pre-Columbian history' and archive them, while also confirming that any abandoned draft notes are moved to trash using `trash_note`.

Standardizing project names across documentation

A team lead notices the tag '#project/v1' is used inconsistently. They instruct the agent to globally rename it to '#project/legacy', ensuring all notes are updated via `rename_tag`.

Drafting a meeting summary from scattered ideas

A writer has several fragmented thoughts saved in different places. They ask the agent to collect all notes tagged 'work/meetings' and then use `add_text` to compile them into one cohesive draft.

Patterns to Avoid

Searching by memory only

✗ AVOID

The user tries to search for a note about the 'Q3 budget' but forgets if they tagged it #finance or used the date tag @today. They end up running multiple, separate searches.

✓ INSTEAD

First, ask the agent to list all notes matching both the keyword and any relevant tags using `search_notes` (e.g., search for 'Q3 budget' AND tag '#finance'). This combines your criteria into one clean query.

Manually updating old project names

✗ AVOID

A team member manually goes through dozens of notes to change the tag from '#project/v1' to '#project/legacy'. They risk missing a few instances, leaving obsolete tags floating around.

✓ INSTEAD

Use the `rename_tag` tool. Tell the agent to rename the old tag globally. It handles every single instance automatically across your entire knowledge base.

Overwriting important context

✗ AVOID

The user uses a generic text injection method that overwrites the first paragraph of a note, losing critical introductory details.

✓ INSTEAD

Always use `add_text` and specify if you want to prepend or append content. This ensures your new information is added adjacent to the existing markdown without accidentally deleting anything.

The Right Fit

Use this MCP when your primary workflow involves retrieving, organizing, or modifying structured knowledge stored in Bear App notes. You need an agent that can read complex markdown and understand tag hierarchies; if you only ever write simple bullet points, the tool might be overkill.

Don't use it if your goal is simply to *write* a note from scratch without referencing old material. If you just want a blank canvas, a basic text editor will do fine. Also, don't expect it to manage files outside of Bear App; its scope is strictly the contents and structure within Bear.

If you need to build an entire complex system that interacts with external databases (like Salesforce or Jira), this MCP isn't built for that—you'd need a specialized API connector. But if your pain point is purely 'How do I get my AI agent to understand the connections between all these markdown documents I wrote over five years?', then Bear MCP is exactly what you need.

Bear MCP: Solving Fragmented Markdown Note Retrieval

Today, managing your notes means opening Bear and mentally navigating a huge file structure. You're constantly running searches, clicking through tag lists, and manually copying the relevant text snippet into your current draft. It's slow, tedious, and you almost always miss something.

With this MCP, you talk to your agent instead of the app. Simply ask for all notes related to 'Q1 Strategy' or search by a specific date range. The agent pulls the full markdown content—the raw truth—and presents it instantly, letting you build context without ever clicking away from your main task.

Bear MCP: Organizing Knowledge Tags and Hierarchies

Manually keeping track of tags is a nightmare. You might have multiple similar tags (e.g., #meeting, #meetings, #meet). When a project changes direction, you waste hours renaming tags across potentially thousands of notes.

This MCP solves that structural problem. The agent can list your entire tag taxonomy and then perform a global rename on the fly. It doesn't just change the name; it updates every single note linked to that tag, keeping your knowledge graph clean.

Bear: 10 Tools for Markdown Note Management

These tools allow your agent to search, create, edit, archive, and restructure all content within your Bear App notes.

#	TOOL	DESCRIPTION
01	`search_notes`	Searches across all your Bear app notes for specified keywords, tags, or dates.
02	`open_note`	Retrieves and displays the full Markdown content of a specific note by its ID.
03	`create_note`	Creates a completely new, blank note within your Bear App vault.
04	`add_text`	Appends or prepends raw markdown text chunks to an existing note's content.
05	`trash_note`	Moves a selected Bear Note directly into the Trash bin.
06	`archive_note`	Removes an explicit Bear Note from active view and places it in the Archive section.
07	`list_tags`	Retrieves a complete map of your tags, showing their full nested parent/child taxonomy structure.
08	`open_tag`	Lists all Bear notes that specifically match one or more provided tags.
09	`rename_tag`	Globally changes the name of a tag, updating every note using it automatically.
10	`delete_tag`	Completely removes a specific tag constraint from your entire knowledge base.

See It in Action

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

U Find all notes tagged #todo from last month mentioning 'database'.

✅ **Found 3 relevant action items:**

- **[Due: May 20th]** Re-evaluate user onboarding flow. (Source: UUID: ABC1...)
- **[Urgent]** Draft initial API specs for migration. (Source: UUID: XYZ4...)
- **[Review]** Check legacy data access points. (Source: UUID: PQR9...)

Would you like me to open the full markdown content of the 'API specs' note so you can copy the details?

U What tags do I have for product launch planning?

🗂️ **Bear Tag Taxonomy Map:**

The current structure includes:

- `#project/launch` (Parent)
  * `#project/launch/marketing` (Child)
  * `#project/launch/devops` (Child)
- `#work/meetings` (Parent)
  * `#work/meetings/v2` (Child)

Let me know if you want to see all notes under a specific branch.

U Create a new draft note for the 'Phoenix' project and tag it #project/phoenix

✅ Success. The note titled 'Project Phoenix Draft - 2026-07-15' has been created in your vault.

It is already tagged with `#project/phoenix`.
You can now start adding text or details to this new location.

Frequently Asked Questions

01 **Can the AI precisely update a note without overwriting its entire content?**

Yes. It uses the `add_text` mutation tool, seamlessly attaching blocks of text to either the absolute bottom (append) or the explicit top (prepend) of the given UUID note, leaving the core intact.

02 **Does it understand nested tags (like #work/design/logo)?**

Bear relies heavily on tagging workflows. The agent natively queries and navigates explicit sub-tag pathways exactly like the application UI, mapping out your distinct taxonomy rules efficiently.

03 **Can it search for uncompleted action items across many notes?**

Simply ask the agent to search for the specialized string '@todo'. Bear exposes these native markers directly via the API, returning every unique UUID containing a matching string checklist efficiently.

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 `"bear": { "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

Bear 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 Bear. 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	Bear MCP
Server ID	`019d7559-f6f8-7121-a954-b37d390cbd04`
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/bear.