MCP SERVER | NO CODE | CLOUD HOSTED

Payload CMS MCP

Control your content and database logic via AI.

Payload CMS MCP lets your AI client interact with complex headless content management systems directly. It provides structured tools for reading, writing, and updating data across collections, singletons, and custom schemas without needing hand-written API endpoints.

A+ Quality Score 100/100

schema-validation | json-api | database-management | content-orchestration | node-framework | rest-api

MAINTAINED BY VINKIUS CLOUD FOR AI AGENTS · JUNE 2026
vinkius.com/mcp/payload-cms

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.

Payload CMS MCP

10 tools available
Cloud-hosted on Vinkius

You can tell your AI agent to manage your website's backend data as if it were a native interface. Instead of building complex code just to fetch or write content, you simply ask for the action—like listing all posts in a collection or updating a specific JSON field on a profile document. Your agent handles the Payload CMS logic automatically, letting you focus on generating the final output, not managing the database plumbing. For example, if you need to update a global site setting, your AI client can execute that change with minimal prompts. Connecting this MCP via Vinkius gives your agent access to over 4,000 other tools, meaning it's one place for all your data needs. You write the prompt; your AI handles the database interaction.

Core Capabilities

01 — Create new content documents

The MCP writes brand new JSON documents into specified Payload collections.

02 — Update existing data fields

It patches specific fields within a document or collection using its unique ID.

03 — Find single site settings

You can extract configuration details from global, singular items that control the entire website.

04 — List all content collections

The MCP scans and retrieves the list of structured data types available in your CMS.

05 — Identify specific users or roles

It queries existing user records to check permissions or find active admin accounts.

One Click on Vinkius — From Prompt to Execution

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

01 First, you subscribe and provide your specific Payload Base URL and private API token parameters.

02 Next, you tell your AI client exactly what data you need—for example, 'list all documents in the posts collection' or 'update the site title to X'.

03 The MCP runs the command against your CMS, returning structured JSON data that your agent can immediately use for the next step.

The bottom line is you get a natural language way to control and manipulate complex database structures.

Built For

This MCP is crucial for backend developers, content engineers, and frontend teams who need to test or modify data structures without constantly dropping into the CMS admin interface. If your job involves reading, updating, or validating site content via API calls, this is what you need.

Backend Developer

You use it to manipulate complex database items and run deep schema revisions when testing new features.

Content Engineer

You map dynamic content updates directly, applying JSON payload changes across multiple collections for site staging or pre-launch checks.

Frontend Team Member

You isolate singleton configuration outputs to validate the global state of your API during response testing.

What Changes When You Connect

01 Instead of writing boilerplate code to read data, you can simply ask the agent to list all documents in a collection using `list_collection_documents`. This saves hours of repetitive API setup.

02 Need to make a change? Use `patch_cms_document` to modify fields on an existing document without having to write full update scripts. It targets specific blocks by ID, making the action precise.

03 The MCP handles complex validation for you. If your site relies on global settings, use `get_singleton_global` to pull that configuration state and test your logic against it immediately.

04 Content lifecycle management gets easier with tools like `create_cms_document`. You can instruct the agent to write a new post or category record simply by providing the JSON data payload.

05 Don't waste time figuring out who has access. Use `list_payload_users` and `verify_token_identity` to audit your system's user roles before deploying any major changes.

Real-World Applications

Staging a new feature page

The content engineer needs to test how the site handles 20 different product descriptions. Instead of manually running 20 API calls, they ask their agent to run `list_collection_documents` for 'products' and then loop through the results, using `get_single_document` on each one to validate its structure.

Fixing a broken site setting

The frontend team notices the primary navigation header is wrong. They ask their agent to use `get_singleton_global` to retrieve the current global settings, spot the error, and then use `patch_cms_document` to correct just the title field.

Deprecating old content

An admin needs to remove an entire outdated media gallery. Instead of running a complex deletion script, they tell their agent to target and wipe the specific record using `wipe_cms_document` by its ID.

Auditing user permissions

The security team wants to know if only admins can create new content. They prompt their agent to run `list_payload_users`, then use the results to check which roles have permission to perform a write operation.

Patterns to Avoid

Treating the CMS like simple key/value store

✗ AVOID

Attempting to update global site settings by just providing two fields, assuming the API will handle everything. This often fails because singletons have complex relational structures.

✓ INSTEAD

Always use `get_singleton_global` first to understand the full schema before making changes. Then, use `patch_cms_document` with all required IDs and structured data to ensure a complete update.

Manually writing every CRUD endpoint

✗ AVOID

Writing separate Python functions for listing, creating, and patching documents just because the CMS is complex.

✓ INSTEAD

Use this MCP. The agent handles the complexity of the Payload REST API calls when you simply ask to `list_collection_documents` or `create_cms_document`.

Ignoring content schemas

✗ AVOID

Trying to write a document using data that doesn't match the required fields for the collection, causing the whole operation to fail.

✓ INSTEAD

Use `list_collection_documents` first. This confirms what structure you're dealing with and helps your agent generate correctly formatted JSON payloads.

The Right Fit

You should use this MCP if your workflow requires interacting with the complex, structured data of a headless CMS—specifically needing to read global settings, create new documents, or patch existing records based on their ID. It's perfect for testing and migration logic.

Don't use it if you just need simple data fetching (e.g., 'get all user names'). If your needs are purely additive or require complex external business logic that doesn't touch the CMS, a general API connector might be better. But when the core problem is 'How do I tell my AI agent to write content *to* this specific Payload structure?' then you need this MCP.

Managing Content in Headless CMSs Is Messy Work

Right now, if your team needs to audit the site's current configuration, you have to jump into three different areas: checking global singletons for primary settings, listing collections to see what content types exist, and then running separate queries just to pull a few specific documents. It’s tedious clicking through dashboards and copy-pasting IDs.

With this MCP, your agent handles the complexity of those database calls automatically. You simply ask it to check the site's global settings or list all available document types. The result is clean data that you can use immediately in your next step.

Payload CMS: Control Content with Write Operations

The biggest time sink used to be the manual process of updating a document. You'd find an ID, copy it, open the correct collection view, patch the JSON payload, and hope you didn't break something else in the process.

Now, your agent manages that whole cycle. You tell it to update the content using `patch_cms_document`, provide the target ID, and define the new data. The job is done; no manual clicking required.

Payload CMS MCP: 10 Tools Available

These tools let your agent perform all standard database operations—from creating new records to checking global configurations—directly within the Payload CMS structure.

#	TOOL	DESCRIPTION
01	`create_cms_document`	Write new, fully structured JSON documents into designated Payload collections.
02	`wipe_cms_document`	Permanently remove specific document rows from the active CMS collection.
03	`get_single_document`	Inspect deep internal arrays to retrieve data for a single, specified record.
04	`get_singleton_global`	Extract properties that control global settings or site-wide configurations.
05	`list_collection_documents`	Find and list all the available structured data types within your CMS collections.
06	`list_payload_users`	List active user accounts, checking which admin identities are currently set up.
07	`verify_token_identity`	Check the current user's identity and permissions associated with the API token.
08	`search_collection_where`	Perform advanced searches across a collection based on specific criteria or filters.
09	`patch_cms_document`	Modify and update existing documents by substituting specific database blocks using their ID.
10	`update_singleton_global`	Dispatch an automated check to ensure global settings are correctly updated across 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 standard explicit documents isolated under the 'posts' collection.

Called target schema mapping collection 'posts'. Pulled 3 documents natively limit parameters. Docs map details: 1 (ID: a3...), Status: Published. 2 (ID: a4...). Shall I explore JSON content inside ID a3?

U Create natively new doc under 'categories', set JSON data `{ "name": "Tech" }`.

Payload confirmed endpoint bounding execution. Created efficiently! New ID assigned logically is '8z9xy2'. Target fields updated explicitly mapping to `{ "name": "Tech" }`. Waiting for next workflow boundaries.

U Wipe document logically bounding the ID 'abc12' from the 'media' collection.

Initiated DELETE binding rules. Headless API verified action gracefully. Document ID explicitly 'abc12' has been purged completely from the active 'media' target collection.

Frequently Asked Questions

01 **How do I find out what collections exist in Payload CMS using the Payload CMS MCP?**

You use `list_collection_documents`. This tool scans your entire setup and returns a list of all structured data types, letting you know exactly where to look for content.

02 **Can I update global site settings with the Payload CMS MCP?**

Yes. You use `patch_cms_document` or `update_singleton_global`. These tools are designed specifically to manage those crucial, site-wide configuration items.

03 **Is the payload cms mcp safe for deleting content?**

It is safe when used correctly. The MCP provides `wipe_cms_document` to irreversibly delete specific records after your agent has verified which documents need removal.

04 **Does Payload CMS MCP work with my private API key?**

Yes, you provide the required Private API Token when setting up the connection. This ensures that only authorized calls can manipulate your backend data.

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 `"payload-cms": { "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

Payload CMS 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 Payload CMS. 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	Payload CMS MCP
Server ID	`019d75f0-1783-72bc-bcd9-507139d9a2aa`
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/payload-cms.