MCP SERVER | NO CODE | CLOUD HOSTED

Flagsmith MCP

Control feature releases and remote configs via chat.

Flagsmith helps you manage feature flags and remote configurations directly through natural conversation. Instead of clicking through multiple dashboards to check or change a setting, your AI agent connects to Flagsmith to handle complex release logic. You can instantly retrieve default flag states for any environment, set user traits, or update entire batches of features without leaving your chat window.

A+ Quality Score 100/100

feature-flags | remote-config | feature-toggles | ab-testing | release-management

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

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.

Flagsmith MCP

8 tools available
Cloud-hosted on Vinkius

Managing feature toggles and remote settings used to mean jumping between the code editor, the staging dashboard, and the documentation—a productivity killer. This MCP lets you skip all that friction. You can talk to your agent and have it query flagsmith's data source for whatever configuration detail you need. For example, you can ask what a new user sees on day one or check if a specific feature is enabled in the staging environment. If you're building complex product flows, this MCP gives you programmatic control over those releases. You connect to Vinkius and get instant access to flagsmith's entire suite of tools, allowing your agent to handle everything from creating new testing environments to updating user attributes based on their current subscription level. It turns a multi-step UI process into a simple conversation.

Core Capabilities

01 — Check the state of all features

Retrieve default flag statuses and full configuration details for any specified environment.

02 — Determine user feature access

Fetch a specific list of flags and traits assigned to an individual user identity or allow you to set those attributes for testing new profiles.

03 — Update multiple features at once

Modify the values of several flags simultaneously, or update single flag states quickly using batch commands.

04 — Manage project scope and settings

Create new isolation environments for testing, fetch full configuration documents for deep local evaluation, or remove specific feature overrides.

One Click on Vinkius — From Prompt to Execution

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

01 Subscribe to this MCP in Vinkius and input your Flagsmith Environment Key or Admin API Key.

02 Tell your agent what change you need; for example, 'Check if maintenance mode is on' or 'Enable beta search flags.'

03 Your agent executes the necessary calls against your account and reports back the current state, or confirms the successful update.

The bottom line is that this MCP takes complex configuration management out of dashboard clicks and puts it into natural language conversation with your AI client.

Built For

This connector is essential for any role deeply involved in the product release cycle. It's for the developer who doesn't want to switch context to check a feature flag, or the product manager tired of manually coordinating rollouts across staging environments.

Software Developer

Debugging new features by asking your agent to check user traits for specific identities or retrieving the full environment configuration document directly from the chat window.

Product Manager

Running A/B tests and coordinating feature rollouts by toggling flags, managing segment overrides, or updating remote configs without logging into the main dashboard.

DevOps / SRE Engineer

Automating environment setup for testing new versions of code by creating new environments or fetching default flag states across multiple project scopes.

What Changes When You Connect

01 Debug feature logic instantly. Instead of switching to the code editor, ask your agent to run 'get_identity_flags' for a specific user ID to see exactly what features they should access.

02 Coordinate rollouts without touching dashboards. Use 'update_flag_v2' or 'update_flag_v1' to change feature states in bulk or individually using natural language commands.

03 Setup isolated testing environments fast. Need to test a new flow? Run 'create_environment' and get a clean scope for your developers, all from the chat interface.

04 Know exactly what every user sees. Use 'get_environment_document' to pull the entire configuration document into your session for deep evaluation or auditing purposes.

05 Manage complex user segmentation. You can use 'identify_user' to simulate different customer types (like 'premium' vs. 'free') and then check their access using 'get_identity_flags'.

06 Clean up old test settings easily. If a feature flag override is no longer needed, simply run 'delete_segment_override' through your agent.

Real-World Applications

A Product Manager needs to check beta access.

The PM wants to ensure only paying customers see the new dashboard widget. Instead of logging into the staging dashboard, they ask their agent to 'get_identity_flags' for a test user and confirm if the required 'beta_access' flag is set correctly.

A Developer needs to debug code locally.

The developer runs into an unexpected bug. They ask their agent to retrieve the full configuration document using 'get_environment_document' so they can verify if the local build environment settings match the expected remote values.

A DevOps Engineer needs a new sandbox.

The team is preparing for a major release and requires an isolated testing area. The engineer simply asks their agent to 'create_environment' in the project, which sets up a clean scope without manual clicks.

A Marketing Team wants to test different rollouts.

The team needs to quickly switch between two versions of a landing page for testing. They use 'update_flag_v2' to toggle both the old and new flag values in seconds, allowing immediate comparison.

Patterns to Avoid

Treating flags as code constants

✗ AVOID

A developer hardcodes a feature boolean value (e.g., `if (featureEnabled == true)`), making the product impossible to turn off without a deploy.

✓ INSTEAD

Always rely on your agent and 'get_environment_flags' to check the current flag state at runtime, ensuring that all features are managed remotely via flagsmith.

Over-relying on UI dashboards

✗ AVOID

Having to log into a separate dashboard just to see if a user has 'premium' traits or what the current environment configuration is.

✓ INSTEAD

Use your agent. Ask it to 'get_identity_flags' directly, pulling that data instantly without leaving your chat window.

Manual flag updates during peak hours

✗ AVOID

During a critical hotfix deployment, having multiple engineers fighting over who gets to change the feature state in the UI.

✓ INSTEAD

Use 'update_flag_v2' via your agent. This centralizes control and executes changes based on structured conversation commands.

The Right Fit

Use this MCP if your workflow involves managing feature toggles, controlling releases, or setting user-specific configurations across multiple environments. You need the ability to change what features are available (the state) without changing the code itself. Don't use it if you simply need to read raw data—for instance, if you just want a list of usernames that exist but don't care about their feature access. In those cases, a general database connector might be better. However, if your goal is *governance* and *control over release*

state, this MCP provides the necessary tools like 'get_environment_document' and 'identify_user'.

The Configuration Dashboard Nightmare

Right now, controlling your product releases means clicking through a dozen different dashboards. You check environment A for the default flags; then you jump to dashboard B to set up segment overrides for beta users. If you need to test how the feature looks on a specific user type, you have to manually simulate that identity and copy/paste values between tabs.

With this MCP, all that manual jumping stops. You tell your agent exactly what needs changing or checking—'Show me the default flags in staging for premium users.' The agent executes the complex logic using 'get_identity_flags', giving you a single, definitive answer instantly.

Flagsmith MCP: Control feature toggles via conversation

You no longer need to manually manage environment creation or remember which key is for which project. Asking your agent to 'create_environment' handles the setup process, and running a command like 'get_environment_document' gives you the full config data without any copy-pasting.

The difference now is speed and centralization. You treat feature configuration as data that can be queried or updated via text commands, eliminating entire categories of context switching.

Flagsmith with 8 Tools

These tools allow your agent to perform every necessary action: from creating new testing environments to updating specific feature flags based on your instructions.

#	TOOL	DESCRIPTION
01	`create_environment`	Creates a brand-new, isolated development environment within your projects.
02	`delete_segment_override`	Removes a specific feature flag rule that was previously set for a segment of users.
03	`get_environment_document`	Retrieves the complete, detailed configuration document for an entire environment.
04	`get_environment_flags`	Fetches all default status values (on or off) for every feature flag in your current environment.
05	`get_identity_flags`	Gets the specific flags and traits that are assigned to a particular user identity.
06	`identify_user`	Designates a user by setting or updating their key attributes, such as subscription level, for targeted testing.
07	`update_flag_v1`	Changes the status of one single feature flag with an API key command.
08	`update_flag_v2`	Updates multiple flags at once, allowing you to apply batch changes across several features.

See It in Action

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

U Get all default flag states for my current environment.

I've retrieved the flags for your environment. There are 12 active flags, including 'new_onboarding_flow' (Enabled) and 'beta_search_algorithm' (Disabled). Would you like the details of any specific flag?

U Identify user 'user_99' and set their 'subscription' trait to 'premium'.

User 'user_99' has been identified and their traits updated. Based on the 'premium' subscription trait, they now have access to the 'advanced_analytics' and 'priority_support' flags.

U Update the flag value for environment 'ser_key_123' to enable the 'maintenance_mode' feature.

I have successfully updated the flag. The 'maintenance_mode' feature is now enabled for environment 'ser_key_123'.

Frequently Asked Questions

01 **How do I check the default flag states using Flagsmith MCP?**

You run 'get_environment_flags'. This tool fetches all default statuses (on/off) for every feature in your current environment, letting you see the baseline configuration instantly.

02 **Can I update multiple flags at once with Flagsmith MCP?**

Yes. Use 'update_flag_v2' to apply batch changes across several features simultaneously using a single command instead of updating them one by one.

03 **What is the difference between identifying and getting identity flags in Flagsmith MCP?**

You use 'identify_user' when you want to *set* or update user traits (e.g., set subscription='premium'). You then use 'get_identity_flags' to retrieve those current settings for verification.

04 **Do I need special permissions to create environments using Flagsmith MCP?**

The agent handles the necessary API calls after you provide your credentials. If you have proper write access, 'create_environment' will build a new isolated space for testing.

05 **How do I get the full configuration document with Flagsmith MCP?**

Run 'get_environment_document'. This retrieves every piece of configuration data in that environment, making it ideal for detailed local evaluation or auditing purposes.

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 `"flagsmith": { "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

Flagsmith 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 Flagsmith. 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	Flagsmith MCP
Server ID	`019e3898-85ce-73c7-8eaf-b8210379c8fe`
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/flagsmith.