MCP SERVER | NO CODE | CLOUD HOSTED

ConfigCat MCP for AI Agents

Manage Feature Flags and Remote Configuration Values Across Environments

ConfigCat manages feature flags and remote configurations directly through your AI client. This MCP lets you list environments, create settings, toggle features, and update configuration values without touching code. It provides granular control over application logic in development, staging, and production.

A+ Quality Score 98.33/100

feature-flags | remote-config | release-management | toggles | configuration-management

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

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.

ConfigCat MCP

18 tools available
Cloud-hosted on Vinkius

You shouldn't need to redeploy code every time a feature flag needs tweaking or an environment variable changes. ConfigCat gives developers and product teams the ability to manage all that complexity from chat alone. Using your AI client, you can list environments (like Test, Staging, Production) and instantly check specific setting values for any given configuration. Need to turn on a beta feature? Your agent handles it: you call the necessary tools to create or update a setting value immediately. You can also define user segments, which is critical for running targeted A/B tests or canary releases. This capability means your release workflow moves at the speed of conversation, not the speed of CI/CD pipelines. When you connect ConfigCat via Vinkius, your AI client gets access to this entire catalog of feature toggles and environment controls.

Core Capabilities

01 — View and organize system configurations

List all available product configurations, environments, and user segments to understand the current setup.

02 — Manage core settings and flags

Create new feature flags or settings—whether they're boolean switches, strings, or numbers—and delete unused ones.

03 — Dynamically adjust values in real-time

Retrieve the current value of any setting for a specific environment and update that value instantly to trigger immediate application changes.

04 — Build user targeting groups

Define new user segments or view existing ones, allowing you to target feature rollouts to specific user groups.

One Click on Vinkius — From Prompt to Execution

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

01 First, subscribe to the MCP and provide your ConfigCat API Key ID and Secret.

02 Second, reference the desired action with your agent—for example, 'What's the current value of the Beta Feature flag in Staging?'

03 Third, your AI client calls the appropriate tools. It returns a clear confirmation or the requested configuration data to you.

The bottom line is: you talk to your agent, and it executes complex configuration changes across development environments for you.

Built For

This MCP is essential for developers who hate the friction of manual config management and DevOps engineers whose job relies on rapid, safe environment synchronization. If your team struggles with coordinating feature rollouts across multiple product versions or environments, this tool saves you hours.

DevOps Engineer

Managing environments using tools like `list_environments` and ensuring that configurations are synced correctly before a deployment.

Software Developer

Quickly toggling features or checking flag statuses directly from their IDE to test logic without needing a full build/deploy cycle.

Product Manager

Overseeing feature rollouts and user segments using simple chat commands, confirming that the right users see the new functionality at the right time.

What Changes When You Connect

01 Control feature rollouts without code changes. Use `update_setting_value` to flip a switch, activating or deactivating features in production instantly.

02 Maintain environment parity across your product line. List all configurations using `list_configs` and verify that Staging matches Production before launch.

03 Target specific users for testing. Create new segments via `create_segment` so you can run A/B tests only on a small, controlled group of users.

04 Speed up debugging by verifying settings. You can check the exact state of any flag in any environment using `get_setting_value`, eliminating guesswork.

05 Streamline your release process. By managing environments and configurations with this MCP, you reduce reliance on manual deployment steps.

Real-World Applications

Testing a new payment flow for premium users

A product manager needs to test a paid feature only on paying customers. They use the agent to `create_segment` specifically for 'Premium Users' and then verify that the associated setting value is active for that group using `get_setting_value`.

Rolling back a broken checkout button

A developer realizes a new feature flag broke the site. Instead of deploying an emergency hotfix, they ask their agent to use `update_setting_value` to immediately set the problematic flag's value back to 'false'.

Setting up isolated staging testing

A DevOps engineer needs a clean test slate. They instruct their agent to `create_environment` for 'Pre-Prod QA', ensuring it has all necessary settings before the next major build.

Patterns to Avoid

Treating config management as code

✗ AVOID

When a flag needs to be changed, developers often treat it like a PR, requiring merging and deployment cycles. This is slow and introduces unnecessary risk.

✓ INSTEAD

Use the ConfigCat MCP's `update_setting_value` tool directly via your agent. This bypasses code deployment entirely, changing the feature state in real-time.

Confusing environments with configurations

✗ AVOID

A user might try to list all settings across their entire product without knowing which container they are checking, leading to ambiguous results.

✓ INSTEAD

Always start by listing the available configuration containers using `list_configs` first. Then, specify the exact environment (like Staging) before asking your agent to check setting values.

The Right Fit

Use this MCP if you need runtime control over application features and environmental settings. If your workflow requires toggling flags, updating variables, or running A/B tests without recompiling or redeploying code, this is for you. Don't use it if your primary goal is data storage or complex user authentication; for those, a dedicated database connector works better. Also, if you only need to read static documentation about settings, the `get_setting` tool will suffice, but you won't be able to modify anything.

ConfigCat MCP for AI Agents: Managing Feature Flags in Software Development

Today, changing a simple feature toggle often requires a full deployment cycle. You write the change, commit it, open a PR, wait for review, and then deploy across multiple environments—all just

With this MCP, you simply ask your agent to check or update flags. For instance, telling your agent to `get_setting_value` for a specific feature in the Test environment gives you instant confirmation.

to see if one flag works correctly on Staging. This friction slows down every release.

You gain control over your software's behavior without touching the build pipeline.

ConfigCat MCP for AI Agents: Controlling Environment Parity with Remote Configurations

Manually verifying that Production, Staging, and Test environments all have identical configurations is a tedious, error-prone chore. Engineers spend hours cross-referencing dashboards just to confirm one variable didn't get missed.

The ConfigCat MCP lets you automate this check. You can use `list_environments` and then instruct your agent to compare settings across multiple locations using `get_setting_value`. Now, configuration drift is a chat command away from detection.

ConfigCat: 18 Tools for Configuration Management

These tools allow your agent to list, create, delete, retrieve, and update every element within your configuration system, from entire environments down to single feature flags.

#	TOOL	DESCRIPTION
01	`create_config`	Creates a brand new configuration container for your product.
02	`create_environment`	Builds a dedicated environment (like QA or Production) to isolate testing.
03	`create_segment`	Defines a new group of users for targeted testing or release management.
04	`create_setting`	Adds a specific feature flag or setting (e.g., 'beta_checkout') to an existing configuration.
05	`delete_config`	Removes an entire product configuration container.
06	`delete_environment`	Deletes a specific testing or live environment.
07	`delete_segment`	Removes a predefined user segment from the system.
08	`delete_setting`	Deletes an existing feature flag or setting key.
09	`get_config`	Retrieves all details about a specific configuration container.
10	`get_environment`	Gets detailed information on a specified environment instance.
11	`get_segment`	Fetches the criteria and details of a specific user segment.
12	`get_setting`	Retrieves metadata about a feature flag or setting key.
13	`get_setting_value`	Checks the current value of a setting within a specific environment.
14	`list_configs`	Lists all configuration containers available for your product ID.
15	`list_environments`	Shows all existing environments (e.g., Test, Staging) tied to a product.
16	`list_segments`	Lists all defined user segments for the entire product.
17	`list_settings`	Shows every feature flag or setting key within a given configuration.
18	`update_setting_value`	Changes the value of an existing setting, which triggers real-time behavior updates in your application.

See It in Action

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

U What are the different environments I have for my main application?

Product ID: prod_123

We currently maintain 4 active environments:

- **Production**: Live customer traffic. (ID: env_prod)
- **Staging**: Final QA testing before release. (ID: env_staging)
- **Beta**: Limited internal testing group. (ID: env_beta)
- **Development**: Local developer sandboxes. (ID: env_dev)

U Can you check the value of 'new_checkout_ui' in Staging?

Found it. The feature flag `new_checkout_ui` is active for the **Staging** environment (`env_staging`).

- **Key:** new_checkout_ui
- **Type:** Boolean
- **Current Value:** True
- **Last Updated:** 2026-06-12 (by Product Manager)

This means the staging environment is currently running the new UI.

U I need to turn off the beta feature flag for everyone.

Got it. I am updating `beta_feature` in **Production** now, setting its value to `False`.

✅ **Update Successful:** Feature Flag 'beta_feature' set to False.

- **Product ID:** prod_123
- **Environment:** Production
- **Change Effective:** Immediately

This change will be reflected in the application within minutes.

Frequently Asked Questions

01 **How does the ConfigCat MCP help manage feature flags across different environments?**

The ConfigCat MCP lets you view and manipulate settings for multiple environments (Test, Staging, Production) from a single chat interface. You can check if the flag status is consistent everywhere, saving hours of manual cross-checking.

02 **Can I use this with my AI client to run A/B tests?**

Yes. By managing user segments and dynamically updating setting values, you can easily target a small group (a segment) for an A/B test without affecting the general user base.

03 **What if I need to delete a whole set of flags or settings?**

You have tools available to remove configuration containers, environments, and individual settings. This helps keep your product's configuration clean and prevents clutter from old features.

04 **Does the ConfigCat MCP require me to write code for simple flag changes?**

No. The entire point of this connector is to manage these controls through natural conversation with your AI client, eliminating the need to commit code just to flip a switch.

05 **Is ConfigCat MCP safe to use in production? How do I know it's secure?**

The tool allows you to get and update values for Production environments. Because all actions are logged and initiated through your AI client, you retain an audit trail of every change.

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 `"configcat": { "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

ConfigCat 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 ConfigCat. 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	ConfigCat MCP
Server ID	`019e387c-cd2f-73db-a339-12e91fd1fc2a`
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/configcat.