MCP SERVER | NO CODE | CLOUD HOSTED

Statsig MCP

Manage features and test product rollouts.

Statsig connects feature flagging and experimentation directly to your AI agent. Test product rollouts, manage dynamic configurations, and log custom events—all through natural conversation. Instead of writing code or jumping between dashboards, you ask your agent about a user's gate status or fetch an experiment's JSON settings immediately.

A+ Quality Score 100/100

feature-flags | ab-testing | product-experimentation | dynamic-config | statsig

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

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.

Statsig MCP

12 tools available
Cloud-hosted on Vinkius

Managing feature flags used to mean living in a separate UI for every single check. Now, you can talk to your AI client and have it handle the complexity of product experimentation. Ask your agent if 'dark_mode_v2' is enabled for user 123, or ask it to fetch the configuration values for an active experiment. You don't need to remember which API key goes where; you just describe what you want done.

This MCP lets developers and PMs verify evaluation logic right from their chat window. Need to create a new feature flag? Just tell your agent, and it handles the setup using its management tools. If you spot an interesting metric during testing, simply ask your agent to log a custom event for real-time analytics. Because Vinkius hosts this MCP, connecting it is easy—you link once from any compatible client, and suddenly, product experimentation is just another conversation.

Core Capabilities

01 — Validate Feature Status

Checks if one or more feature flags are enabled for a specific user.

02 — Manage Experiment Settings

Creates, updates, and deletes feature gates, allowing you to control product access without code changes.

03 — Retrieve Dynamic Configurations

Fetches specific JSON configurations or lists all available dynamic settings for an experiment.

04 — Inspect User Layers

Retrieves parameter values from a user's layer to understand how the feature evaluation works.

05 — Track Usage Metrics

Logs custom events directly for generating analytics and calculating metrics.

One Click on Vinkius — From Prompt to Execution

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

01 First, subscribe to this MCP in Vinkius and provide your Statsig Server Secret Key and Console API Key.

02 Next, tell your agent the exact task you need—for instance, 'List all feature gates' or 'Check gate X for user Y'.

03 Your agent executes the request through the tools, pulling back structured data (like True/False status or JSON configs) directly into the chat.

The bottom line is you talk to your AI client about product logic, and it handles all the API calls behind the scenes.

Built For

This MCP is for Product Managers who hate waiting on engineering tickets for a simple flag check. It's for Developers tired of switching between their IDE and a dozen different dashboard tabs just to test an endpoint. And it’s perfect for Data Scientists needing quick access to configuration layers.

Product Manager

Verifies if a new checkout flow is ready by checking gate statuses or creating temporary feature flags without bothering engineering.

Software Developer

Validates evaluation logic or logs test events directly from the IDE to confirm how code interacts with dynamic configs.

Data Scientist

Inspects experiment configurations and retrieves layer parameters to verify metric logging before running a full analysis pipeline.

What Changes When You Connect

01 You can check a user's access status immediately. Instead of navigating to the Statsig UI, asking your agent to run `check_gate` tells you instantly if 'new_checkout_flow' is active for that specific user.

02 Product changes are controlled directly via chat. Use `create_gate` or `update_gate` to manage feature flags and gates without needing a developer pull request just to flip a switch.

03 Configuration data comes straight to your agent. If you need the settings for an experiment, calling `get_config` pulls the JSON structure so your workflow can consume it instantly.

04 Monitoring happens in conversation. Use `log_event` when testing to send custom events that feed into your analytics dashboard, keeping track of every action without extra logging code.

05 You can build out a complete picture of user behavior. By using `get_layer`, you inspect the underlying parameters, which helps data scientists validate if their metric collection is working correctly.

Real-World Applications

Validating an A/B Test for Marketing

A Product Manager needs to know if a new landing page variant ('redesign') is live. They ask the agent, 'What's the status of the redesign gate for user X?' The agent runs `check_gate` and confirms its status (True/False), letting them greenlight the launch immediately.

Debugging a Broken Feature in Code

A Developer finds a bug related to settings. They ask the agent to fetch the configuration for 'payment_gateway' using `get_dynamic_config`. This returns the exact JSON they need, letting them fix the issue without searching through old documentation.

Tracking User Funnel Drop-off Points

A Data Scientist suspects a specific user action is failing. They ask the agent to `log_event` when that failure point is hit, tagging it with context. This creates a clean data stream for metric calculation.

Onboarding a New Team Member

A new PM needs to see all available features for the project. They ask the agent to `list_gates`, which instantly provides them with a comprehensive list of every flag and gate that exists, letting them start planning right away.

Patterns to Avoid

Trying to manage all data in one place

✗ AVOID

Copying configuration values from the Statsig web console and pasting them into a spreadsheet or another database tool.

✓ INSTEAD

Instead, use your agent. If you need the config, ask it to `get_dynamic_config` or `get_config`. It pulls structured data directly into your chat window for immediate consumption.

Hardcoding feature logic in client code

✗ AVOID

Writing an 'if (user.isPremium)' check every time a developer builds a new screen, which requires full redeployment.

✓ INSTEAD

Use the MCP to manage this instead. Just ask your agent to `create_gate` and control access via feature flags, letting you roll out changes without touching the code base.

Forgetting to track key user actions

✗ AVOID

Not logging when a critical button is clicked in testing because it's not part of the core flow.

✓ INSTEAD

Use `log_event`. You can easily instruct your agent to log custom events anytime, ensuring every metric you care about gets tracked for analytics.

The Right Fit

Use this MCP if your team needs to test product hypotheses and manage features dynamically. This is the tool for controlling access, running A/B tests, and verifying configurations without code changes. You absolutely need it if Product Managers or Developers need immediate answers on gate status (using `check_gate`) or need to log custom events (`log_event`) during development.

Don't use this MCP if you simply need general API access—like reading a public list of users, or managing unrelated infrastructure resources. For pure data storage and retrieval that has nothing to do with product feature toggles, look for dedicated database connectors instead. This is highly specialized for the product experimentation lifecycle.

The pain of manually checking feature flags across multiple dashboards

Right now, if you want to know if a new payment method is enabled for your test user, you have to open the Statsig UI. Then, you might need to jump to a different dashboard to view the related dynamic config, and then check another system just to confirm which layer parameters are active. It's a painful sequence of tabs and copy-pasting.

With this MCP, you simply ask your agent about it. You don't click anything. You tell it what user and what feature flag you're checking, and the agent returns the status—true or false—in one simple reply.

Statsig MCP: Instant Feature Flag Status Checks

Manual processes involve writing a small script just to run `check_gate` for every test user. Then, you have to manually read the results and cross-reference them with other internal systems.

Now, your agent handles that entire process. It performs the evaluation logic instantly through natural language conversation. You get immediate answers on feature availability without writing a single line of boilerplate code.

Statsig: 12 Tools for Product Experimentation

These tools allow you to manage every part of the product experimentation lifecycle—from creating a flag to logging the final event.

#	TOOL	DESCRIPTION
01	`check_gate`	Evaluates if one or more feature gates are enabled for a specific user.
02	`create_dynamic_config`	Creates an entirely new dynamic configuration within the Statsig project.
03	`create_gate`	Establishes a brand-new feature gate for use in product rollouts.
04	`delete_gate`	Removes an existing, unused feature gate from the project entirely.
05	`get_config`	Fetches configuration values for a specific dynamic config or experiment type.
06	`get_dynamic_config`	Reads and returns the details of an existing dynamic configuration.
07	`get_gate`	Retrieves all information about a specific, named feature gate.
08	`get_layer`	Fetches parameter values from a user's layer to show how the evaluation works for that user.
09	`list_dynamic_configs`	Provides an index of every dynamic configuration currently set up in your project.
10	`list_gates`	Lists all feature gates that exist within the current Statsig project.
11	`log_event`	Logs custom events, allowing you to track specific user actions for analytics purposes.
12	`update_gate`	Modifies the settings or description of an existing feature gate.

See It in Action

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

U Check if the gate 'new_checkout_flow' is enabled for user 'user_99'.

I've checked the gate 'new_checkout_flow' for user 'user_99'. The gate is currently **Open** (True).

U List all feature gates in my Statsig project.

I've retrieved the list of gates. You have 3 active gates: 'ai_beta_access', 'dark_mode_v2', and 'new_checkout_flow'.

U Create a new feature gate named 'mobile_redesign' with description 'Testing new UI'.

The feature gate 'mobile_redesign' has been successfully created in your project.

Frequently Asked Questions

01 **How do I use Statsig MCP to check if a gate is active?**

You ask your agent to run the `check_gate` tool and provide the specific feature name and user ID. The agent evaluates the flag using the secret key and tells you the status (True/False).

02 **Can I use Statsig MCP to create new features?**

Yes, you can manage your flags directly. Use `create_gate` or `update_gate` when you need to establish a feature flag or modify an existing one's settings.

03 **What is the difference between dynamic configs and gates in Statsig MCP?**

Gates control whether a feature can be used at all. Dynamic configs store flexible data, like API keys or UI colors, that the feature uses once it's active.

04 **If I need to track user actions, which tool do I use in Statsig MCP?**

You use `log_event`. This tool lets you send custom events for analytics. You tell the agent what action happened and when it occurred.

05 **Does Statsig MCP require specific keys to run?**

Yes, the setup requires your Statsig Server Secret Key for evaluation tools (like `check_gate`) and your Console API Key for management tasks (like `create_gate`).

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 `"statsig": { "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

Statsig 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

DOCUMENT INFORMATION

Generated	June 2026
MCP Server	Statsig MCP
Server ID	`019e38f3-2cec-7239-aaa7-da3df1e2b898`
Platform	Vinkius Cloud for AI Agents
Endpoint	`https://edge.vinkius.com/{token}/mcp`

LICENSE & USAGE