MCP SERVER | NO CODE | CLOUD HOSTED

UptimeRobot MCP

Manage every site health check from your chat.

UptimeRobot lets your AI agent actively manage your entire website infrastructure. You can list all running services, create new health checks for any endpoint, set up alert contacts like Slack or Email, and pull historical metrics—all without opening a browser.

A+ Quality Score 100/100

uptime-monitoring | http-checks | alerting | server-health | incident-response | api-monitoring

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

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.

UptimeRobot MCP

10 tools available
Cloud-hosted on Vinkius

Managing uptime usually means jumping between dashboards, checking status pages, and manually updating who needs to be notified when things break. This MCP changes that. You connect it once through the Vinkius catalog, giving your AI client direct control over your monitoring system. Your agent handles everything from listing all configured services to creating new endpoints, whether they use HTTP or Ping checks. Need to know if an old contact still works? Just ask. Want to clear out stale data? It can reset historical stats for specific monitors. This lets you get a real-time view of your entire infrastructure's health right inside your chat window, making incident response instant and conversational.

Core Capabilities

01 — View and Manage Monitors

List all configured services to check their current status, retrieve detailed historical logs, or create brand new endpoints to monitor.

02 — Configure Alerts

Manage who gets notified when an outage happens by listing existing contacts or creating new recipients for Email and Slack alerts.

03 — Handle Account Limits

Check your UptimeRobot account usage to see how many monitor slots you have left under your current plan.

04 — Adjust Monitoring Settings

Update existing monitor configurations or permanently delete services that are no longer needed.

05 — Analyze Historical Data

Reset monitoring logs for a specific service or pull raw data arrays of up ratios and response times for graphing.

One Click on Vinkius — From Prompt to Execution

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

01 Subscribe to this MCP in Vinkius and enter your main UptimeRobot API Key.

02 Your AI agent connects the key, gaining real-time access to all monitoring data and account controls.

03 You simply ask for status reports or command an action (like creating a new monitor) through natural conversation.

The bottom line is you control your entire uptime strategy from one place: your AI client's chat window.

Built For

This connector is built for engineers and operations staff who spend too much time clicking through dashboards to check service health. If you're tired of manually managing alerts or pulling data into spreadsheets, this gives your AI agent the hands-on control it needs.

DevOps Engineer

You use it to instantly set up standard monitoring checks for newly deployed microservices straight from your terminal.

SRE Team Lead

You pull Service Level Agreement (SLA) metrics and incident reaction logs directly into your incident management context window for review.

System Administrator

You quickly add new on-call employees to notification targets, ensuring nobody is left out of critical alerts.

What Changes When You Connect

01 You can instantly see the status of all sites. Instead of navigating a dashboard, ask your agent to list all monitors and get an immediate pass/fail report.

02 Incident response is faster. You don't need to manually update contact lists; just tell your agent to create an alert contact for a new on-call employee or delete old ones.

03 Deep data analysis is simple. Need metrics? Your agent can pull raw uptime ratio and response time arrays, letting you graph the data without exporting CSVs.

04 Setup happens instantly. If Marketing launches a new site, your agent can create an HTTP monitor in seconds; no manual form filling required.

05 Resource management is clear. Before starting, check your UptimeRobot account usage to know exactly how many monitoring slots you have left.

Real-World Applications

Responding to a Critical Outage

A core team member notices Site B is down. Instead of opening the dashboard, they tell their agent: 'List all monitors and highlight anything failing.' The agent instantly reports the failure and can then fetch the complete log history for that specific failed process.

Onboarding a New Team Member

An administrator needs to add three new on-call staff members. They use their agent to list all contacts, then execute 'create alert contact' three times, adding them by email and Slack without leaving the chat interface.

Auditing Old Services

A developer knows a microservice was decommissioned last month. They tell their agent to get monitor details for that service ID. The agent retrieves all configuration info so they can confirm if it needs manual deletion using 'delete uptime monitor'.

Preparing for Audit Reports

The SRE lead needs quarterly data. They ask the agent to pull raw historical logs and up ratios from several monitors, providing a clean, structured array that's ready for immediate reporting.

Patterns to Avoid

Over-relying on Dashboards

✗ AVOID

Having to open the UptimeRobot website, click through multiple tabs (Monitors > Contacts > History), and copy/paste data into a separate document.

✓ INSTEAD

Use your agent to perform the action directly. For example, instead of checking history manually, use 'list monitors' for a summary, then follow up with specific commands like 'get monitor details' or 'reset monitor logs'.

Manual Contact Management

✗ AVOID

The team changes on-call personnel. An admin has to log in and manually edit dozens of email addresses across various alert settings.

✓ INSTEAD

Use the agent to list all contacts, then use 'delete alert contact' for old users, followed by 'create alert contact' for new ones. It handles the whole flow conversationally.

Confusing Status with Data

✗ AVOID

Thinking that seeing a monitor name means you have access to all its underlying data points.

✓ INSTEAD

Don't just list monitors. Use 'get monitor details' or 'list_monitors' for the current status, and use 'reset monitor logs' when you need to start fresh metrics.

The Right Fit

Use this MCP if your core problem is operationalizing monitoring checks: getting real-time health data, managing alert recipients, or updating service configurations using only natural language. You should connect here if you want your agent to actively *manage* the system (e.g., 'Can we create a new Ping check?'). Don't use this MCP if all you need is simple archival storage for logs; that requires a specialized data warehouse tool. Also, don't use it if you only need to write a basic shell script to ping a single URL once—an external scripting language will be faster for that. However, if the task involves checking multiple services or updating who gets notified when things break, this is your definitive choice.

The Dashboard Trap

Right now, monitoring means navigating a mess of tabs and pages. You have to open the UptimeRobot site, check the 'Monitors' tab for status updates, then click over to 'Alert Contacts' just to see who needs notification changes. If you need history, it’s another deep dive into a separate log view.

With this MCP, you talk directly to your agent. You simply ask about service health, and the response is immediate, conversational data. Your agent handles the entire cross-tab process for you.

UptimeRobotMCP: Direct Control Over Service Health

Manual steps that vanish include checking status via 'list monitors', updating contact lists using 'create alert contact' or 'delete alert contact', and setting up checks with 'create uptime monitor'. All these actions are done in a single chat thread.

What's different now is control. You don't just view the data; you actively change it, managing your entire infrastructure coverage through simple conversation.

UptimeRobot: 10 Monitoring Tools

Use these tools to manage every aspect of your service health checks, from creating new endpoints to handling alert contacts.

#	TOOL	DESCRIPTION
01	`delete_alert_contact`	Permanently removes an email or Slack address from the list of people who receive system alert notifications.
02	`delete_uptime_monitor`	Irrevocably deletes a specific website monitor that was set up for health checking.
03	`update_uptime_monitor`	Changes the settings, like the URL or check type, for an already existing uptime monitor.
04	`get_account_info`	Retrieves usage data and limits for your UptimeRobot account to show remaining monitoring slots.
05	`get_monitor_details`	Pulls the complete, specific details for one particular uptime monitor you want to investigate.
06	`list_alert_contacts`	Shows all notification targets configured in your account (Email, Slack, Webhook, etc.).
07	`list_monitors`	Returns a summary of all configured uptime monitors, including their pass/fail status and response times.
08	`create_alert_contact`	Sets up a new recipient for alerts, specifying if it's an Email address or a Slack channel.
09	`create_uptime_monitor`	Creates a brand-new health check monitor by providing a friendly name, URL, and type (HTTP or Ping).
10	`reset_monitor_logs`	Quickly resets the historical statistics and logging data for one specific uptime monitor.

See It in Action

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

U List all my monitors and highlight any that are currently down.

You currently have 15 active Uptime monitors. 14 are responding properly (Status: UP). 1 monitor (`API Authentication Service` , ID 9923812) is currently DOWN. It has been failing ping requests for the last 14 minutes. Do you want me to fetch the complete log history for that failed process?

U Create an HTTP monitor for our newly deployed marketing site (`https://marketing.vinkius.com/`).

I have successfully created an HTTP monitor named 'marketing.vinkius.com'. Its unique ID is `1884931` . Your account indicates you have 34 monitor slots still freely accessible. Please keep in mind it might take approximately 2–5 minutes before it collects its first meaningful round of SLA availability stats.

U Delete alert contact 'old-devops@acme.com'. Our team rotated users.

I've permanently removed notification target ID `78233` (old-devops@acme.com) from UptimeRobot global alert contacts list safely. Future downtimes will no longer dispatch there. Would you like me to create an external endpoint webhook alert point to replace it immediately in the pipeline?

Frequently Asked Questions

01 **How do I check if an endpoint is up using UptimeRobotMCP?**

You can list all configured services with 'list monitors' to see the pass/fail status, or use 'get monitor details' for a deep dive into any specific service.

02 **Can I add a Slack channel as an alert contact using UptimeRobotMCP?**

Yes. First, you list all contacts with 'list_alert_contacts', and then use 'create_alert_contact' specifying the correct ID for Slack.

03 **What if I need to delete a service monitor? Do I have to do it manually?**

No. You can permanently remove the monitor using the 'delete uptime monitor' tool, provided you understand this action is irreversible.

04 **Does UptimeRobotMCP help me find out if my account has enough slots?**

Yes. Use the 'get_account_info' tool; it retrieves your current usage and tells you exactly how many monitor slots are left on your subscription.

05 **How do I start monitoring a brand new URL with UptimeRobotMCP?**

You call the 'create uptime monitor' tool. You just need to provide a friendly name, the target URL, and whether it's an HTTP or Ping check.

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 `"uptimerobot": { "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

UptimeRobot 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 UptimeRobot. 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	UptimeRobot MCP
Server ID	`019d761a-0cf4-70ae-9f06-06a1684ac974`
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/uptimerobot.