

MCP SERVER

NO CODE

CLOUD HOSTED

Coolify MCP for AI Agents

Manage self-hosted cloud infrastructure and container deployments

Coolify lets you manage your self-hosted private cloud and container deployments using natural conversation. From checking server health to triggering complex build pipelines for PostgreSQL, MySQL, or Redis, you control every aspect of your infrastructure directly from any AI agent.

A+ Quality Score 98.33/100

self-hosting

docker

server-monitoring

deployment-automation

infrastructure-as-code

container-management



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.

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.

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.

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

Encryption at rest

Ed25519

PKI vault signatures

24h TTL

Ephemeral session keys

V8 Isolate

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.

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.

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.

06 — Honeypot Trap System

Phantom credentials are injected into isolated environments. If a honeypot 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.

FinOps

Cost tracking per tool, payload compression savings, budget optimization signals, and consumption trends.

Firewall & DLP

PII redaction activity, sensitive data protection counters, and security event timeline.

Agent Activity

Which AI clients are connecting, how often, and what they're doing — real-time session tracking.

Tool Health

Slowest and most error-prone tools, with actionable root-cause insights and performance baselines.

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.

Coolify MCP

10 tools available

Cloud-hosted on Vinkius

Running a private cloud means juggling multiple dashboards: one for networking status, another for application metrics, and yet a third just for database credentials. This MCP lets you bypass the clicking entirely. You connect your Coolify instance to Vinkius and give your AI client full read/write access to your self-hosted environment.

Instead of manually SSHing into nodes or navigating through deployment menus, you talk to your agent. Need to know if a server node has active Docker Swarm status? Just ask. Want to spin up a container that's currently suspended? Tell it to do so. You can manage the entire application lifecycle—starting, stopping, and restarting services—and even initiate full build pipelines for Nixpacks images or pull the latest Git commits. For databases like PostgreSQL and Redis, you instantly get the connection strings needed to link your backend apps. It's total infrastructure control, conversational style.

Core Capabilities

01 — Inventory Server Nodes

Lists all self-hosted nodes and retrieves detailed networking parameters, including IP properties and Docker Swarm status.

02 — Monitor Application Topology

Retrieves internal metrics for any application, mapping GitHub branches, Traefik proxy paths, and rollout toggles.

03 — Control Application Lifecycle

Allows you to start, stop, or restart applications by recycling container states instantly.

04 — Automate Deployments

Triggers raw build pipelines that fetch the latest commits, rebuild Nixpacks images, and roll out updated Docker versions.

05 — Manage Database Credentials

Lists managed PostgreSQL, MySQL, and Redis configurations and extracts internal connection strings for linking applications to data sources.

One Click on Vinkius — From Prompt to Execution

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

- 01 Subscribe to this MCP on Vinkius.
- 02 Enter your Coolify Instance URL and API Token (you find this in the Keys & Tokens section of your Coolify dashboard).
- 03 Use your preferred AI client—whether it's Claude, Cursor, or something else compatible—to issue commands like 'list all servers' or 'restart application X'.

The bottom line is you get to use natural language conversation to perform complex infrastructure tasks that normally require multiple logins and manual console work.

Built For

This MCP is built for the people who live in the command line or on a dashboard all day. If your job involves monitoring server node health, managing container states across environments, or deploying updates without touching the GUI, you need this.

DevOps Engineer

Monitoring server node health and application lifecycle states across multiple private cloud environments.

Full-stack Developer

Triggering builds, restarting containers, and retrieving database connection strings directly from a chat window so they can pass credentials to their local scripts.

System Administrator (SysAdmin)

Auditing Docker Swarm configurations and verifying server connectivity across the whole cluster using plain language questions.

What Changes When You Connect

-
- 01 Get instant status checks on server health. Instead of logging into the dashboard, you ask your agent to run `get_server` and get confirmation that all Docker Swarm services are running fine.

 - 02 Automate complex deployment cycles like a human doing it with SSH keys. Triggering raw builds via `trigger_deployment` pulls the latest Git commit and rolls out updated containers without manual steps.

 - 03 Never lose track of credentials again. Using `list_databases` instantly provides connection strings for PostgreSQL, MySQL, and Redis so your developer doesn't have to copy-paste from a settings page.

 - 04 Control the application state on demand. Need to pause billing or test something? Use `stop_application` to suspend services, then use `start_application` when you're ready to go.

 - 05 Simplify development auditing. You can retrieve connection strings using `get_database` and track which repositories are associated with specific applications via `list_applications`.
-

Real-World Applications

A new feature requires a backend database link.

The dev asks their agent: 'What's the connection string for our user DB?' The agent runs `list_databases` and returns the internal URI, allowing the developer to immediately update the environment variables without accessing any admin panel.

A critical server node appears offline.

The ops engineer asks: 'Check the status of the main node.' The agent runs `get_server` and returns IP properties and SSH validation statuses, immediately telling the engineer if the issue is network-related or container-related.

A code push needs to go live, but we need a full rebuild.

The founder tells their agent: 'Deploy the latest commit for the API.' The agent executes `trigger_deployment`, pulling the newest Git commits and rolling out the updated Docker image instantly.

We need to test an application's functionality without running it 24/7.

The developer asks: 'Can you temporarily disable the frontend?' The agent runs `stop_application`, suspending the service and preventing web perimeter ingress until testing is complete.

Patterns to Avoid

Checking dependencies manually

X AVOID

Copying connection strings from a dashboard or having to log into multiple services just to verify if the correct UUID links an app and its database.

✓ INSTEAD

Use `list_databases` to get all connection strings, and use `get_application` combined with `list_applications` to audit which repositories are linked to which operational UUIDs.

Assuming the latest code is live

X AVOID

Seeing a 'Deployment successful' message but realizing the container hasn't actually updated, or that old env vars are still active.

✓ INSTEAD

Always use `trigger_deployment` to guarantee a full build pipeline runs. If you suspect environment variables aren't taking effect, run `restart_application` immediately after deployment.

Confusing node status with application state

X AVOID

Only checking that the main server is 'online,' but not realizing a specific container (like Redis) has been suspended or hasn't updated its configuration.

✓ INSTEAD

First, check the overall cluster health using `list_servers` and `get_server`. Then, use `list_applications` to confirm all services are listed before checking application status with `stop_application` or `start_application`.

The Right Fit

You should use this MCP if your daily job involves managing self-hosted infrastructure where the command line and API calls are faster than clicking through dashboards. Specifically, you need to audit networking parameters (`get_server`), manage multiple container states (start/stop/restart), or automate complex CI/CD

steps across diverse technologies like PostgreSQL and Nixpacks builds. Don't use it if your only requirement is checking basic connectivity—a simple ping check will suffice. Also, don't rely on this for advanced infrastructure-as-code planning; while you can read the state using `list_applications`, setting up complex multi-stage deployments requires a dedicated orchestration platform.

Managing Self-Hosted Docker Swarm Status with Coolify

When running applications on self-hosted infrastructure, the most tedious part is manually verifying that every node in your container swarm is healthy and correctly configured. You spend time SSHing into nodes just to check IP properties or confirm if a service port has resolved across the cluster.

With this MCP, you eliminate that manual overhead. Running `get_server` gives you an instant status report on every physical endpoint, confirming both network connectivity and Docker Swarm operational status in one go.

Automating Application Deployments with Coolify

Previously, rolling out a new version meant triggering the build pipeline, waiting for it to finish, and hoping that all the necessary environment variables (like those for MySQL) were correctly injected into the running container's memory.

Now, you simply tell your agent to deploy. The MCP handles the whole sequence: fetching the latest Git commit, rebuilding the Nixpacks image, and ensuring the new Docker version rolls out cleanly, leaving you with a fully updated service.

Coolify: 10 Tools for Container Management & Deployments

Use these tools to query server health, manage application state, retrieve database credentials, and trigger full build pipelines across your private cloud.

#	TOOL	DESCRIPTION
01	<code>get_application</code>	Fetches internal topology metrics for a specific application, showing its mapped GitHub branches and proxy paths.
02	<code>get_database</code>	Retrieves the necessary configuration arrays to link newly provisioned web apps to their backend databases.
03	<code>get_server</code>	Verifies a specific server node's IP properties, SSH connection status, and active Docker ports across the cluster.
04	<code>list_applications</code>	Generates a map of all managed applications by listing their project repositories and UUIDs needed for mutations.
05	<code>list_databases</code>	Lists all configured database types, including PostgreSQL, MySQL, and Redis, along with their connection details.
06	<code>list_servers</code>	Identifies the raw physical endpoints running Docker swarms that host subsequent applications.
07	<code>restart_application</code>	Bounces an application container, ensuring updated environment variables take effect immediately in runtime memory.
08	<code>start_application</code>	Restores service by spinning up containers mapped to a previously suspended UUID.
09	<code>stop_application</code>	Halts an application's execution, useful for pausing services or restricting web access during maintenance.
10	<code>trigger_deployment</code>	Performs a full build pipeline, fetching the latest Git commit and rolling out the updated Docker image over the active version.

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 active servers and tell me which ones are running Swarm mode.



Server Cluster Status

We have 3 nodes reporting:

- `main-prod-node` : Active (Swarm Ready) | IP: 192.168.1.5
- `staging-backup-node` : Idle (Not Swarm Ready) | IP: 10.0.0.7
- `dev-sandbox-node` : Active (Swarm Ready) | IP: 192.168.1.12

The `main-prod-node` and `dev-sandbox-node` are ready for container orchestration.

U I need the connection string for our main user database.



Database Connection Details

Found 3 managed data sources:

1. **PostgreSQL (User DB):** `postgres://user:*@coolify-db-abc:5432/user_db`
2. **MySQL (Analytics):** `mysql://admin:*@coolify-db-xyz:3306/analytics`
3. **Redis Cache:** `redis://localhost:6379`

Please remember these URIs are only valid within your internal network.

U Can you check the status of the 'auth' app and restart it if needed?



Application Status Check

- **App Name:** Auth Service
- **Status:** Currently Suspended (Stopped)
- **Last Update:** 2026-06-10

I've restarted the container for 'Auth Service.' The new state is **Running**. I recommend running a quick health check now to confirm everything settled properly.

Frequently Asked Questions

01 How does Coolify MCP help me manage my self-hosted containers?

This MCP lets you control your entire private cloud through conversation. You can start, stop, or restart applications and monitor the health of every server node without ever logging into a dashboard.

02 Can I get database credentials using Coolify MCP for AI Agents?

Yes, you can list all your managed databases (PostgreSQL, MySQL, etc.) and retrieve their internal connection strings instantly. This is vital for linking new backend services.

03 What if I need to deploy a massive code update?

You don't have to do it manually. The MCP triggers the full deployment pipeline, which fetches the latest Git commits and rolls out updated Docker images across your cluster automatically.

04 Does Coolify MCP help with server monitoring?

Absolutely. You can list all attached servers to verify their IP properties and check the live status of the underlying Docker Swarm configuration, ensuring maximum uptime.

05 Is this better than using a dashboard for daily tasks?







It is much faster. Instead of navigating multiple tabs and clicking through menus, you ask your agent to perform complex multi-step actions—like checking server health *and* restarting an app—all in one chat session.

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 <code>"coolify": { "url": "..." }</code>
 Windsurf	MCP Settings → <code>mcp_settings.json</code> → 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

Coolify 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](https://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 Coolify. 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	Coolify MCP
Server ID	019d757b-e463-7017-a891-3ae482987d4f
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/coolify.