MCP SERVER | NO CODE | CLOUD HOSTED

Drone CI MCP for AI Agents

Monitor build status and manage deployment pipelines

Drone CI lets your AI client manage entire DevOps pipelines right from chat. Connect it to monitor builds, handle repository settings, and manage critical environment secrets without logging into a dashboard. It brings continuous integration and deployment control directly into your natural conversation.

A+ Quality Score 98.33/100

ci-cd | build-automation | repository-management | pipeline-monitoring | secrets-management | deployment

MAINTAINED BY VINKIUS CLOUD FOR AI AGENTS · JUNE 2026
vinkius.com/mcp/drone-ci

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.

Drone CI MCP

39 tools available
Cloud-hosted on Vinkius

Need to keep track of build failures or manually trigger a test run? This MCP connects your AI agent to the Drone CI system, giving you full command over your development lifecycle via plain language commands. Instead of jumping between dashboards, you talk to your agent and it performs actions like listing recent builds, getting detailed log reports, or even restarting failed deployments.

It handles more than just code checking; you can manage infrastructure pieces too. You'll use it to create necessary environment secrets, set up scheduled tasks using cron jobs, or control repository access. If your current setup feels scattered across multiple tools, connecting this via Vinkius lets all those functions live under one roof. It gives developers and SREs the ability to monitor pipeline health and manage resources directly from their chat interface.

Core Capabilities

01 — View and Control Build Status

Get a list of recent builds, check detailed build information, or restart specific failed pipelines.

02 — Manage Repository Settings

Enable, update, or synchronize repository settings with your source control provider.

03 — Handle Secrets and Credentials

Create, read, update, or delete sensitive environment secrets and credentials needed for deployment.

04 — Schedule Automated Tasks (Cron)

Set up new scheduled cron jobs or manually trigger existing background tasks.

05 — Administer Users and Access

Create, delete, or update user accounts and manage overall organizational access permissions (requires admin rights).

One Click on Vinkius — From Prompt to Execution

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

01 Subscribe to the Drone CI MCP in Vinkius and provide your specific Drone Server URL and Personal Access Token.

02 Select this MCP within any compatible client, like Cursor or Claude. Your agent now has access to all build, repo, and secret tools.

03 Tell your AI client what you need done—for example, 'List the last five builds for my web app'—and it executes the command.

The bottom line is that once connected, your agent translates natural language into specific DevOps API calls, giving you immediate control over complex CI/CD systems.

Built For

This MCP solves the problem of context switching for technical teams. It's built for the SRE who gets tired of hopping between a dashboard and Slack to manage secrets, or the developer who needs build logs immediately in their IDE. If your job involves checking deployment status or managing infrastructure credentials, this is for you.

Site Reliability Engineer (SRE)

Managing core platform secrets and setting up scheduled maintenance tasks using tools like `create_secret` or `create_cron_job`.

DevOps Engineer

Monitoring pipeline health, checking build logs with `get_build_logs`, and manually restarting deployments (`restart_build`) when needed.

Software Developer

Quickly checking the status of a feature branch or viewing user accounts using `list_users` without leaving their primary coding environment.

What Changes When You Connect

01 Instead of manually checking a web dashboard, you can ask your agent to list recent builds or get details on the latest run using `list_builds` or `get_build`.

02 You eliminate context switching when managing credentials. Use the MCP to create and retrieve secrets via `create_secret` or `get_secret`, all within your chat window.

03 The agent handles complex admin tasks, like setting up recurring maintenance jobs using `create_cron_job` and keeping track of them with `list_cron_jobs`.

04 When a deployment fails, you don't hunt through logs. You simply ask to get build logs using `get_build_logs` and the failure point appears instantly.

05 Admin tasks become conversational. Need to onboard someone? Use your agent to create new users or sync repositories with `create_user` or `sync_user_repos`.

Real-World Applications

Investigating a Failed Deployment

A developer notices the staging environment build failed. They ask their agent to 'Show me the logs for the latest API build.' The agent uses `get_build_logs` and immediately identifies that the error is a connection timeout, allowing them to fix it instantly.

Implementing Scheduled Maintenance

The SRE needs to run a cleanup job every Monday at 3 AM. Instead of logging into the scheduler UI, they ask their agent to 'Set up a cron job for user data cleanup.' The MCP uses `create_cron_job` and handles the scheduling.

Adding New Security Credentials

A new microservice requires database access. The developer asks the AI client to 'Create a secret key for the production DB credentials.' The agent runs `create_secret`, making sure the credential is stored securely and ready for use.

Auditing User Access

The Ops Manager needs to know who has access. They ask their agent to 'List all registered users' which uses `list_users`. The manager gets a full list of accounts, speeding up compliance checks.

Patterns to Avoid

Manual Dashboard Navigation

✗ AVOID

A developer has to open the CI dashboard, click into the repo, find the build number, then navigate to logs, and finally copy the error message. This takes five minutes.

✓ INSTEAD

Just ask your agent: 'What was wrong with the last web app build?' The agent uses `get_build` and `get_build_logs` and presents the summarized failure reason instantly.

Using General API Tools

✗ AVOID

Trying to manage secrets or builds using a generic tool that doesn't understand DevOps context, forcing manual parameter entry for every action.

✓ INSTEAD

Use this MCP. It understands the specific domain, so you just say: 'Update the database password secret.' The agent handles calling `update_secret` with the correct parameters.

Skipping Ownership Changes

✗ AVOID

A team member leaves, and their repositories are left under their account ownership. Trying to manually transfer access rights across multiple systems.

✓ INSTEAD

Run `chown_repo`. This tool correctly changes the repository's ownership to your current user account, fixing permissions in one step.

The Right Fit

You need this MCP if managing build status, secrets, and scheduled tasks is a frequent part of your day-to-day job. Use it when you want to avoid jumping between the CI dashboard, a secret manager UI, and Slack just to finish one task. Don't use it if all you need is simple code review—you'll be fine with any general AI client. But if your workflow involves triggering builds (`create_build`), updating credentials (`update_secret`), or viewing build logs (`get_build_logs`), this specialized tool gives you the specific, granular control needed for professional DevOps work.

Drone CI MCP: Automating Build and Deployment Pipelines

Right now, managing a full software deployment cycle involves constant context switching. You check the build status in one tab, retrieve secrets from another dashboard, manually trigger a test run, and then copy-paste error logs into Slack for your teammate to look at. It’s click heavy, tedious, and prone to human error.

With this MCP, you simply tell your agent what needs doing—'Restart the web app build.' The system handles checking the status, restarting the process, and confirming completion all in one conversation thread. You get immediate, conversational control over every stage of deployment.

Drone CI MCP: Managing Repository Secrets and Credentials

Manually handling secrets is a major pain point. Today, you're forced to log into the secret manager, find the correct key (like the database password), update it with the new value, and then make sure every application consuming that key knows about the change.

This MCP makes credential management conversational. You can ask your agent to 'Update the production API key.' It handles verifying the key's existence, safely updating its value, and logging the action—all without you ever leaving your chat interface.

39 Drone CI Tools: Build Monitoring & Repository Management

Use these tools to manage everything related to your DevOps pipeline, including user accounts, secrets, builds, and scheduled jobs.

#	TOOL	DESCRIPTION
01	approve_build	Approves a build that is currently blocked and awaiting manual sign-off.
02	chown_repo	Changes the ownership of a repository to be assigned to your current user account.
03	create_build	Starts a custom build process for a specific branch in a repository.
04	create_cron_job	Sets up and registers a new automated, scheduled task (a cron job).
05	create_secret	Generates and stores a new secret credential for use within the repository.
06	create_template	Creates a reusable template that can define build or deployment steps.
07	create_user	Adds a new user account to the organization (Admin privilege required).
08	decline_build	Declines a build that was blocked and awaiting manual sign-off.
09	delete_cron_job	Removes an existing automated cron job from the system.
10	delete_secret	Permanently deletes a repository secret credential.
11	delete_template	Deletes a reusable template that was previously created.
12	delete_user	Removes an existing user account from the organization (Admin privilege required).
13	disable_repo	Disables or completely removes a repository from service.
14	enable_repo	Activates and registers a repository within the Drone CI system.
15	get_build_logs	Retrieves detailed logs for a specific step during a build stage, helping you find errors.
16	get_build	Pulls all the details about a specific build, including its stages and steps.
17	get_cron_job	Retrieves the current configuration and status of an existing cron job.

#	TOOL	DESCRIPTION
18	get_current_user_repos	Lists all repositories that belong to your currently authenticated user account.
19	get_current_user	Provides information about the user who is currently logged into the system.
20	get_repo	Retrieves detailed information for a specific repository by name or ID.
21	get_secret	Shows the current details and metadata of a stored repository secret.
22	get_template	Retrieves detailed information about a specific reusable template.
23	get_user	Looks up details for another user account using their unique login name (Admin privilege required).
24	list_builds	Retrieves a list of the most recent build attempts for your repository.
25	list_cron_jobs	Shows all scheduled cron jobs configured for a specific repository.
26	list_secrets	Lists all the secret credentials currently stored for your repository.
27	list_templates	Displays a list of reusable templates available across the entire organization.
28	list_users	Lists every user registered in the system (Admin privilege required).
29	promote_build	Moves a successful build from one environment to a more advanced target environment.
30	repair_repo	Fixes any issues with the repository's webhooks, ensuring proper communication.
31	restart_build	Forces a specific build run to start again from scratch.
32	stop_build	Stops a running build process immediately (Admin privilege required).
33	sync_user_repos	Updates and synchronizes the list of repositories associated with your user account.
34	trigger_cron_job	Manually executes an existing scheduled cron task right now.
35	update_cron_job	Modifies the schedule or parameters of a running cron job.
36	update_repo	Changes general configuration settings for an existing repository.
37	update_secret	Modifies the value or metadata of a stored repository secret.
38	update_template	Makes changes to the structure or content of an existing reusable template.

#	TOOL	DESCRIPTION
39	update_user	Modifies details for another user account in the organization (Admin privilege required).

See It in Action

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

U Show me all my repositories and what their latest build status is.

Current User Repositories

REPO NAME	LATEST BUILD STATUS	LAST UPDATED
acme/web-app	✅ Success (Build #45)	2026-06-10
acme/api	🟡 Failed (Build #43)	2026-06-11
docs/site	✅ Success (Build #12)	2026-06-05

Looks like the API build failed. Would you like me to fetch the logs for that run?

U I need a new secret key for the staging environment database.

Secret Creation Initiated

- **Repository:** acme/web-app
- **New Secret Name:** `STAGING_DB_PASSWORD`
- **Status:** Pending Value Input

Please provide the secure value for this secret. Once you send it, I'll store it and update the configuration automatically.

U What cron jobs are running for the API repo?

Scheduled Tasks for acme/api

- **Job Name:** Daily Health Check

Schedule: *0 3* (3 AM daily)

- **Status:** Active
- **Last Run:** 2026-06-12 at 3:00 AM

I can also trigger this job right now if you need to test it.

Frequently Asked Questions

01 **How does the Drone CI MCP help me monitor build status?**

You can ask your agent for a list of builds or check specific details about a recent run. It pulls this data directly, showing you if deployments succeeded or failed without needing to visit the dashboard.

02 **Can I use Drone CI MCP to manage my sensitive credentials?**

Yes, it lets you create, read, update, and delete secrets right from the chat. This keeps your process secure and fast; you never have to copy/paste passwords into a UI.

03 **Does Drone CI MCP handle scheduling tasks?**

Absolutely. You can use it to set up new automated cron jobs or manually trigger existing ones, managing all recurring work from one place.

04 **What if a build fails? Can the Drone CI MCP help me find the error?**

When you ask about a failure, your agent can retrieve detailed logs for that specific step. It gives you the exact technical message and line number needed to fix the problem immediately.

05 **Is Drone CI MCP only for admins?**

No, while some tools require admin rights (like deleting users), most common tasks—checking status or viewing logs—are available to any user who needs visibility into the pipeline health.

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 `"drone-ci": { "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

Drone CI 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 Drone CI. 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	Drone CI MCP
Server ID	`019e388e-a3c8-73ab-8785-d5901571820c`
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/drone-ci.