MCP SERVER | NO CODE | CLOUD HOSTED

CircleCI MCP for AI Agents

Manage CI/CD Builds and Deployments from Natural Language

CircleCI connects your automated build and deployment pipelines to any AI agent, letting you manage complex CI/CD workflows using plain conversation. You can list recent builds, check job statuses, trigger manual deployments, and view environment contexts without ever opening the CircleCI dashboard.

A+ Quality Score 100/100

continuous-integration | continuous-deployment | pipeline-automation | build-automation | workflow-orchestration | software-delivery

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

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.

CircleCI MCP

8 tools available
Cloud-hosted on Vinkius

Managing software releases used to mean clicking through endless dashboards—a tedious process of checking status codes, cross-referencing branch names, and manually hitting 'run'. This MCP changes that. It gives your AI client direct control over your entire CI/CD lifecycle. You can ask your agent to list all recent pipelines across multiple projects or check the deployment status of a specific job simply by talking to it.

With this integration, you're not just getting read-only access; you can initiate actions, like triggering an immediate run for a critical branch. If you already use other tools in your stack, connecting via Vinkius makes sure that all your operational intelligence is available from one place, letting you focus on writing code and shipping features, not managing dashboards.

Core Capabilities

01 — View Build History

List and retrieve full details for recent CI/CD pipelines across every configured organization.

02 — Start New Pipelines

Manually trigger a new pipeline run on specific projects or branches when needed.

03 — Check Workflow Statuses

Access comprehensive information about workflows and the individual jobs that make them up.

04 — Audit Job Failures

Get detailed metadata, including the exact execution status, for any specific job run.

05 — Manage Environments

List shared environment contexts used to secure sensitive project data across your organization's projects.

One Click on Vinkius — From Prompt to Execution

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

01 Subscribe to this MCP and obtain your CircleCI Personal API Token from your user settings.

02 Provide the token to your AI client through Vinkius. This authorizes your agent to interact with your pipelines.

03 Ask your agent a natural language question, like 'What's the status of the staging deployment?' The agent executes the necessary tools and reports the findings.

The bottom line is that you keep all your CI/CD control within your existing chat interface, eliminating context switching between dashboards.

Built For

This MCP is built for Ops Engineers and Software Developers who are tired of the 2 AM dashboard dive. If your job involves monitoring complex release cycles or debugging failed builds without opening a browser, this tool saves you hours.

DevOps Engineer

Monitoring pipeline health across multiple services and initiating manual builds when automated gates fail.

Software Developer

Debugging job failures or reviewing workflow progress instantly without navigating to the CircleCI UI.

Release Manager

Verifying the final status of a release pipeline and confirming environment variable requirements from a chat interface.

What Changes When You Connect

01 Immediate visibility into build failures. Use `get_job_details` to pull up specific job metadata, telling you exactly *why* a deployment failed without deep diving into logs.

02 Control the release cycle directly via chat. You can manually start deployments for critical branches using `trigger_cci_pipeline`, perfect when an automated gate needs human approval.

03 Understand your entire infrastructure scope. Running `list_cci_contexts` shows you every shared environment variable used by projects, helping manage sensitive data access.

04 Consolidated status checks. Instead of checking five different dashboards, running `list_cci_pipelines` gives you a single view of all recent activity across your organization.

05 Deep workflow intelligence. You can check the full scope of any process using `list_workflow_jobs` and understand which components feed into the final build.

Real-World Applications

The 'Staging Environment' Check

A release manager needs to confirm if the staging environment is ready for a new feature branch. They ask their agent, and it uses `list_cci_pipelines` to show the last three successful builds on that specific branch, confirming readiness.

Debugging an Intermittent Build Failure

A developer notices a flaky build in production. Instead of guessing, they ask their agent to use `get_job_details` for the failed job ID. The agent returns the specific error logs and execution status instantly.

Forcing an Urgent Hotfix Build

A critical bug is found in production. A team member asks their agent to trigger a new pipeline for the hotfix branch. The agent uses `trigger_cci_pipeline` and confirms the new run ID, starting the fix immediately.

Understanding Project Dependencies

An infrastructure engineer needs to know which shared variables are used across multiple microservices. They ask the agent to list all contexts using `list_cci_contexts`, giving a clear map of dependencies.

Patterns to Avoid

Manually checking every pipeline status

✗ AVOID

The developer opens the CircleCI dashboard, navigates to Project A's pipelines, then clicks into the job list. Repeats this for Project B and C.

✓ INSTEAD

Instead of clicking through dashboards, ask your agent to run `list_cci_pipelines`. It collects all recent build statuses from multiple projects instantly via natural language.

Forgetting required context details

✗ AVOID

The developer wants to debug a job but doesn't know the specific workflow ID or environment name, so they get vague error messages.

✓ INSTEAD

First, ask your agent to use `list_pipeline_workflows` on the failed build. This shows you all constituent workflows, giving you the necessary IDs for deeper investigation.

Confusing workflow definition with run status

✗ AVOID

The developer thinks that just because a job ran successfully doesn't mean the *entire* deployment was approved.

✓ INSTEAD

To verify all steps, ask your agent to use `list_workflow_jobs` on the specific workflow. This confirms not only if jobs ran but what their final state (Success, On Hold) is.

The Right Fit

Use this MCP when you need to manage or audit CI/CD processes without leaving your chat window. It's perfect for quick status checks (`list_cci_pipelines`) or initiating manual actions (`trigger_cci_pipeline`). Don't use it if you are writing the actual pipeline configuration YAML; that requires direct access and editing within CircleCI. If your goal is simply to read documentation, you don't need this MCP. You only need this when natural language control over the build lifecycle is critical.

CircleCI MCP: Streamlining Automated Build Monitoring

Today, monitoring a multi-stage software release requires context switching. You jump from the deployment dashboard to check job statuses, then open another tab to list recent pipelines just to confirm if the build actually started. It's a painful cycle of clicking through menus and tabs just to get an answer.

With this MCP, your agent handles that complexity. You simply ask for a status update, and it uses tools like `list_cci_pipelines` and `get_job_details` to aggregate all the necessary information—the success/fail state, the job name, and the project ID—and delivers a clean summary back to you.

CircleCI MCP: Managing Cross-Project Dependencies

When working on microservices architecture, knowing which environment variables are shared across five different teams is a nightmare. You have to contact three people and cross-reference documentation just to verify if the staging context variable names match.

This MCP eliminates that guesswork. By asking the agent to list all shared contexts using `list_cci_contexts`, you get an immediate, authoritative inventory of every environment variable used across your entire organization's projects.

8 Tools in the CircleCI MCP for Pipeline Job Management

Use these tools to check build history, get workflow details, list environments, or manually start new pipelines across your organization.

#	TOOL	DESCRIPTION
01	`get_my_cci_profile`	Retrieves basic information about the authenticated user within your CircleCI account.
02	`get_workflow_details`	Fetches detailed information for a specific, named workflow template.
03	`list_cci_contexts`	Lists all shared environment contexts available across your entire organization.
04	`list_workflow_jobs`	Retrieves a list of every individual job that belongs to a specific workflow template.
05	`list_cci_pipelines`	Lists the status and details for your most recent CI/CD pipelines across all projects.
06	`trigger_cci_pipeline`	Manually starts a new pipeline run for a designated project or repository branch.
07	`get_job_details`	Retrieves detailed execution metadata and status for a specific, running job instance.
08	`list_pipeline_workflows`	Shows all available workflows associated with a single, completed pipeline run.

See It in Action

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

U What were the results for my web app deployment yesterday?

Build Status Summary

- Project: `web-app` (Repo ID 456)
- Last Run: Yesterday, 3:15 PM UTC
- Status: **FAILED**

Details: The workflow ran successfully up to the testing stage. However, the final deployment job failed with error code 500. You need to check `web-app`'s logs for more info.

U Can you start a new build right now for the main branch?

🚀 **Pipeline Initiated!**

I've triggered a fresh pipeline run for `main` on project `api-gateway`. The process is starting immediately. You can track its progress using this ID: `550e8400-...`.

Please monitor the job list below for status updates.

U Show me all available workflows for the API service:

API Service Workflows Found:

- `build-core`: Builds and packages core libraries.
- `test-integration`: Runs end-to-end integration tests.
- `deploy-staging`: Manually deploys to the staging environment (Requires approval).

Which one do you want details on?

Frequently Asked Questions

01 **How can I use the CircleCI MCP for AI Agents to check build status?**

You simply ask your agent for a summary of recent builds or a specific job's status. The agent uses the necessary tools to pull up the details, giving you an instant report without needing to open the dashboard.

02 **Does CircleCI MCP allow me to trigger manual deployments?**

Yes. You can tell your agent to start a new pipeline for any project or branch. This is useful for hotfixes when you need an immediate, controlled deployment run.

03 **Can I find out what environment variables are used across my projects?**

Absolutely. By asking the MCP to list shared contexts, your agent retrieves a complete inventory of all defined environment variables for your entire organization, which is critical for security and auditing.

04 **What if I need details on a specific job that failed?**

You can tell the agent about the failing job ID or workflow. It will use the available tools to pull detailed execution metadata, allowing you to pinpoint exactly where and why the code broke.

05 **Is CircleCI MCP useful for new developers joining the team?**

Yes. New hires can ask the agent to list all current workflows or retrieve their user profile information, giving them immediate context on how the company's pipelines are structured and managed.

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 `"circleci": { "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

CircleCI 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 CircleCI. 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	CircleCI MCP
Server ID	`019d7570-3b24-7176-9b8c-c0a9a4e6d27c`
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/circleci.