MCP SERVER | NO CODE | CLOUD HOSTED

Coder MCP for AI Agents

Manage Remote Development Environments and Build Statistics

The Coder MCP connects your remote development environment to any AI agent, letting you monitor builds, manage workspaces, and troubleshoot deployments using natural language. You get instant access to build statistics, detailed logs, and insights from active AI sessions without ever leaving your chat window.

A+ Quality Score 98.33/100

remote-development | workspace-management | coder | ai-bridge | infrastructure-as-code

MAINTAINED BY VINKIUS CLOUD FOR AI AGENTS · JUNE 2026
vinkius.com/mcp/coder-remote-dev

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.

Coder (Remote Dev) MCP

84 tools available
Cloud-hosted on Vinkius

Managing complex dev infrastructure usually means jumping between a terminal, a dashboard, and a log viewer. It's slow, frustrating, and you always lose context. This MCP changes that. It lets your agent connect directly to your Coder deployment, making remote development management conversational.

You can ask your AI client to check the latest build information or retrieve specific logs from an agent just by asking. Need to know how many users are active in a workspace? Just prompt it. You can audit AI-assisted workflows by fetching thread history from sessions, or even update core infrastructure settings like SSH configurations.

If you're building complex pipelines, this integration is huge. It surfaces deep platform insights—from deployment statistics to user activity duration—right into your chat stream. Since Vinkius hosts thousands of specialized MCPs, connecting Coder here means you have one central access point for all your remote development tooling.

Core Capabilities

01 — Monitor Deployment Health

Retrieve build details and deployment statistics to check the status of your infrastructure.

02 — Troubleshoot Agent Logs

Stream logs from specific workspaces or agents so you can instantly debug environment issues.

03 — Manage AI Sessions and Users

List active AI Bridge sessions, available models, and connected client details to audit usage.

04 — Configure Infrastructure Settings

Access and retrieve critical settings like SSH configurations or workspace ACLs.

05 — Handle User & Organization Management

List users, organizations, and manage associated roles, tokens, and secrets.

06 — Audit AI-Assisted Development Flows

Fetch the thread history from specific AI sessions to track how developers used AI assistance.

One Click on Vinkius — From Prompt to Execution

Available at **vinkius.com/mcp/coder-remote-dev** — connect your AI agent in three steps.

01 Subscribe to this MCP and enter your Coder URL along with a session token.

02 Your agent connects via Vinkius, authenticating access to your remote dev clusters.

03 You ask your AI client natural language questions—like 'What are the deployment stats?'—and receive structured data back instantly.

The bottom line is you stop needing dedicated CLI commands or dashboard navigation; you just talk to your infrastructure.

Built For

This MCP is for the DevOps Engineer who gets tired of context switching between terminal sessions and web dashboards. It's also built for Platform Teams that need centralized visibility into how AI agents are being used across multiple remote development workspaces.

DevOps Engineer

Checks deployment statistics or streams logs from a workspace agent to identify bottlenecks without leaving the chat interface.

Platform Architect

Uses this MCP to list connected AI clients and available models, ensuring all remote workspaces meet security standards.

Software Developer

Retrieves the SSH configuration or user profile details when setting up a new local development machine for a project.

What Changes When You Connect

01 Stop switching tabs. Instead of navigating to a dashboard, ask your agent to fetch deployment statistics using get_deployment_stats and get the data instantly.

02 Audit AI usage without manual effort. Use list_ai_sessions to see exactly which models are active in your workspaces right now.

03 Deep debugging capability: When an agent fails, use get_agent_logs to stream real-time logs directly into your chat conversation for immediate analysis.

04 Centralized access control: Managing users and groups is straightforward. You can list_users or check workspace ACLs without needing the dedicated admin portal.

05 Immediate configuration checks: Need to set up a new machine? get_ssh_config gives you the exact details required, saving manual lookups.

06 AI workflow review: Understand how AI was actually used by fetching threads from an AI session using get_ai_session_threads.

Real-World Applications

A build failed and I need to know why.

The agent immediately uses get_agent_logs. It pulls the last 10MiB of logs and summarizes the error, pointing directly to the missing dependency line in the code output.

We need to check if our team has permission to access a specific workspace.

The agent executes get_workspace_acl. It returns a clear list of roles and permissions assigned to the user, preventing unauthorized resource access before development starts.

I'm onboarding a new developer who needs SSH setup guides.

The agent runs get_ssh_config and prints the required hostname prefix and suffix. The developer copies this single block of text, eliminating the need to read complex documentation.

I need an overview of all active AI development sessions in the company.

The agent uses list_ai_sessions. It provides a table showing three different clients (e.g., Cursor, Claude), their current model usage, and how long each session has been running.

Patterns to Avoid

Checking logs manually in the UI

✗ AVOID

A dev gets an error, then clicks through five different tabs (logs, builds, agents) to find the timestamp and relevant message.

✓ INSTEAD

Instead of clicking around, prompt your agent using get_agent_logs. Give it the workspace identifier and ask for 'the logs from 10 minutes ago.' You get the data immediately in text format.

Assuming configuration is correct

✗ AVOID

A new engineer can't connect because they don't know if the SSH config needs a key or just a hostname.

✓ INSTEAD

Use get_ssh_config. This tool delivers the exact, current connectivity parameters, eliminating guesswork about networking requirements.

Asking for stats piecemeal

✗ AVOID

A manager asks 'How many workspaces?' then later asks 'What's the build count?', requiring multiple manual queries.

✓ INSTEAD

Ask one prompt: 'Show me the overall deployment statistics.' The agent uses get_deployment_stats and gives you a single, consolidated report.

The Right Fit

Use this MCP if your development workflow relies heavily on visibility into infrastructure health and remote environment status. Specifically, if you need to correlate build logs with AI session usage or manage complex user roles in one place, this is what you need. Don't use it if all you ever do is write code locally; then a basic text editor works fine. If your primary need is managing external billing systems or database records unrelated to dev environments, you should look at specialized CRMs or databases MCPs instead.

Coder MCP for AI Agents: Streamlining Deployment Monitoring and Build Stats

Right now, checking your deployment status is a chore. You have to log into the dashboard, check build info manually, then switch tabs to see if any updates are available, all while hoping you didn't miss an error message buried in the logs.

With this MCP, asking for 'the current Coder build information and deployment statistics' gives you a single, clean report. You get immediate answers on build version, active workspaces, and whether updates are pending, keeping your focus entirely on coding.

Coder MCP for AI Agents: Managing Remote Agent Logs and Workspaces

Troubleshooting remote environments used to mean SSHing into a jump box, running `tail -f`, and copy-pasting log segments across multiple chat windows until you found the root cause.

Now, simply ask your agent for 'the logs from this specific workspace'. It fetches and streams the necessary data directly to you. You get instant debug visibility without ever leaving your primary interaction point.

Coder (Remote Dev): 23 Tools for Build Statistics and Logs

Use these tools to monitor deployment health, list user accounts, stream agent logs, and check build configurations in your remote development clusters.

#	TOOL	DESCRIPTION
01	`get_agent_debug_magicsock`	Get Tailscale magicsock debug info (local agent API)
02	`get_agent_debug_manifest`	Get the startup manifest from the server (local agent API)
03	`get_agent_external_auth`	Get external authentication tokens for the agent
04	`get_agent_logs`	Streams detailed logs from a specific agent instance so you can debug runtime issues.
05	`get_ai_session_threads`	Get threads for an AI session
06	`get_api_root`	Get basic API information
07	`get_app_auth_redirect`	Redirect to a URI with an encrypted API key
08	`get_app_host`	Get the base host for applications
09	`get_appearance`	Get dashboard branding and banners
10	`get_audit_logs`	Get a paginated list of audit logs
11	`get_build_info`	Get Coder build info
12	`get_chat_messages`	Get messages for a chat session
13	`get_deployment_config`	Get deployment configuration
14	`get_deployment_stats`	Retrieves key deployment metrics like build counts and active workspaces.
15	`get_external_auth_device`	Initiate device-based OAuth
16	`get_insights_daus`	Get Daily Active User stats
17	`get_insights_templates`	Get usage data for templates
18	`get_insights_user_activity`	Get activity duration per user

#	TOOL	DESCRIPTION
19	`get_notifications_inbox`	List user notifications
20	`get_notifications_settings`	Get global notification settings
21	`get_notifications_templates`	List available notification templates
22	`get_prebuild_settings`	Get prebuild settings
23	`get_ssh_config`	Provides the necessary SSH configuration details to connect your local machine to the remote environment.
24	`get_update_check`	Check for Coder updates
25	`get_user_profile`	Get user profile
26	`get_workspace_acl`	Get workspace ACLs
27	`get_workspace_build_logs`	Get logs for a specific build
28	`get_workspace_build_params`	List parameters used for the build
29	`get_workspace_build`	Get details of a specific build
30	`get_workspace`	Get workspace metadata
31	`list_ai_clients`	List connected AI Bridge clients
32	`list_ai_interceptions`	List AI interceptions
33	`list_ai_models`	List available AI models
34	`list_ai_sessions`	Lists all active AI Bridge sessions, showing which models are in use and who is connected.
35	`list_chats`	List user chat sessions
36	`list_external_auth`	List linked external accounts (e.g., GitHub)
37	`list_groups`	List groups
38	`list_licenses`	List enterprise licenses
39	`list_org_members`	List members of an organization
40	`list_org_provisioner_daemons`	List active provisioner daemons
41	`list_org_provisioner_jobs`	List jobs for the organization provisioners

#	TOOL	DESCRIPTION
42	`list_org_roles`	List assignable roles in an organization
43	`list_organizations`	List organizations
44	`list_tasks`	Manage long-running AI tasks
45	`list_template_examples`	List starter template examples
46	`list_template_versions`	List versions for a template
47	`list_templates`	List all templates
48	`list_user_secrets`	List user secrets
49	`list_user_tokens`	List user tokens
50	`list_users`	Gets a list of all user accounts managed within the Coder deployment.
51	`list_workspace_port_shares`	List port shares for a workspace
52	`list_workspaces`	g., owner:me). List workspaces
53	`login_user`	Authenticate a user with email and password
54	`register_agent_log_source`	Register a new log source
55	`send_agent_logs`	Send logs to the server
56	`update_agent_app_status`	Update status of an application running on the agent
57	`update_appearance`	Update dashboard branding and banners
58	`update_organization`	Update an organization
59	`update_prebuild_settings`	Update prebuild settings
60	`update_user_profile`	Update user profile
61	`update_workspace_acl`	Update workspace ACLs
62	`update_workspace_autostart`	Update workspace autostart schedule
63	`update_workspace_autoupdates`	Update workspace autoupdates schedule
64	`upload_file`	Upload a file (tar or zip)
65	`add_license`	Add an enterprise license

#	TOOL	DESCRIPTION
66	auth_workspace_agent	Authenticate agent using cloud provider identity
67	cancel_workspace_build	Cancel a pending or running build
68	check_auth	Check if user has permission to perform actions
69	create_chat_message	Send a message to a chat session
70	create_chat	Start a new AI chat session
71	create_group	Create a group in an organization
72	create_organization	Create an organization
73	create_user_secret	Create a user secret
74	create_user_task	Create a long-running AI task for a user
75	create_user_token	Create a user token
76	create_user	Create a user
77	create_workspace_port_share	Create a port share for a workspace
78	create_workspace	Create a workspace
79	delete_license	Delete an enterprise license
80	delete_user	Delete a user
81	download_file	Download a file by ID
82	get_agent_connection	Get DERP and connection details for an agent
83	get_agent_containers	List running containers and devcontainers
84	get_agent_debug_logs	Get the last 10MiB of agent logs (local agent API)

See It in Action

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

U What are the deployment stats? I need to know about active workspaces and builds.

🚀 **Coder Deployment Health Report**

- **Active Workspaces:** 12
- **Total Builds This Cycle:** 8
- **Current Build Version:** v2.10.2
- **Update Check:** Yes, a minor update is available (v2.10.3). Would you like to initiate the check?

U Can I see the thread history for the AI session that ran last week? The one about microservices.

AI Session Audit: Microservices Flow

- **Session ID:** 9b27-73b0...
- **Date Range:** Oct 1 - Oct 5
- **Key Threads Intercepted:**

* [Thread 1] Defined service boundaries (Source: User)
* [Thread 2] Optimized API call structure (Source: AI Model)
* [Thread 3] Identified missing error handling block (Source: AI Model)

U Who are the users on this platform? I need a list of all accounts.

User Account Directory

USERNAME	EMAIL	STATUS
jdoe	john.d@corp.com	Active
psmith	patrick.s@corp.com	Suspended
rjones	r.jones@corp.com	Active

Note: The list can be filtered by group membership if you provide a specific group name.

Frequently Asked Questions

01 **How can I monitor the status of my remote development workspaces using Coder MCP for AI Agents?**

You simply ask your agent to provide deployment statistics. It gives you a real-time count of active workspaces and build cycles, keeping you updated on infrastructure health instantly.

02 **What if I need to check logs from an older failed build using Coder MCP for AI Agents?**

The agent lets you access specific build history. You can request the workspace build logs and review exactly what happened, which is much faster than checking manual log archives.

03 **Is this tool better than manually running commands in a terminal?**

Yes. Instead of opening a separate terminal for every query (like `ssh` or `docker logs`), you ask your agent, and it handles the command execution and result formatting inside the chat.

04 **Can Coder MCP for AI Agents help me manage user access rights?**

Yes. You can list users to see who has accounts, or check workspace ACLs to confirm which roles have permission to join a specific project environment.

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 `"coder-remote-dev": { "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

Coder (Remote Dev) 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 Coder (Remote Dev). 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	Coder (Remote Dev) MCP
Server ID	`019e387a-e4e5-72b8-b947-7356066dabc7`
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/coder-remote-dev.