MCP SERVER | NO CODE | CLOUD HOSTED

Railway MCP

Manage Cloud Ops via Conversation

The Railway MCP lets your AI agent manage cloud deployments and infrastructure settings through natural conversation. List projects, check service statuses across environments like staging or production, track deployment histories, audit persistent storage volumes, and adjust environment variables—all without opening a browser dashboard.

A+ Quality Score 100/100

paas | deployment | infrastructure-as-code | microservices | environment-management | cloud-hosting

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

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.

Railway MCP

11 tools available
Cloud-hosted on Vinkius

Managing multi-service applications often means clicking through half a dozen tabs in the web console just to find one status update. This MCP changes that. It connects your AI agent directly to your Railway account, letting you handle complex cloud operations using plain chat. Need to know if production variables are set correctly? Ask. Want to see why the last deployment failed? Just ask for the history. Your agent acts like a dedicated ops engineer, pulling data on everything from project details and service configurations to persistent storage volumes. Because this MCP is part of Vinkius's massive catalog, you connect once to your preferred AI client (like Cursor or Claude) and gain access to all your infrastructure tools in one place. It gives you full control over your deployments right inside your existing workflow.

Core Capabilities

01 — Audit Project Structure

Retrieve a complete list of your cloud projects, including names and basic details.

02 — Inspect Services and Environments

See all deployed services (web apps, databases) within a project and filter them by specific environments like development or staging.

03 — Track Deployment Status

Get the deployment history for any service to check success/failure statuses and timestamps.

04 — Manage Config Variables

Check what environment variables exist or set new values for a specific service in a given environment.

05 — Review Custom Domains

List custom domains assigned to services and check their SSL certificate status to ensure public access is working.

One Click on Vinkius — From Prompt to Execution

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

01 Subscribe to this MCP and provide your Railway Personal Access Token.

02 Connect the MCP to your preferred AI client (e.g., Cursor or Claude).

03 Give your agent a natural language command, like 'List all services in production for my main API project,' and get the data back.

The bottom line is you manage complex cloud infrastructure by talking to your AI agent instead of clicking through dashboards.

Built For

This MCP solves the problem of context switching. It's built for engineers who spend too much time toggling between their IDE, the terminal, and a complex cloud dashboard just to check status or adjust a variable.

DevOps Engineer

Needs to quickly audit project configurations, review deployment statuses across multiple environments (dev, staging, prod), and manage persistent volumes.

Backend Developer

Uses it to inspect specific services or check if the correct environment variables are set before running local tests that need cloud context.

Team Lead / Architect

Monitors project health, ensuring proper separation and variable security between different environments and teams.

What Changes When You Connect

01 Check deployment history without leaving your IDE. Your agent runs `list_deployments` and tells you if the latest build succeeded or failed, saving clicks.

02 Audit project scope instantly. Use `list_projects` to get an overview of every service group you manage across multiple applications.

03 Control environment variables directly. You can use `set_variable` when your team needs a database connection string updated for staging without logging into the web UI.

04 Monitor system health with domain checks. Run `list_domains` to quickly confirm if custom URLs are correctly pointing to services and have valid SSL certificates.

05 Understand data persistence. Use `list_volumes` to see exactly which services rely on persistent storage and how large those volumes are.

Real-World Applications

Diagnosing a Failed Deployment

A developer notices the web app is down. Instead of checking the dashboard, they ask their agent to run `list_deployments`. The agent reports that the most recent deployment failed because the required environment variable was missing for that service.

Onboarding a New Service

A team lead needs to check if the new worker API is properly configured. They ask their agent to run `list_services` filtered by the 'staging' environment, immediately seeing all related containers and databases.

Changing Production Configs

The database password changes. The DevOps engineer asks the agent to use `set_variable` for the main API service in production, getting instant confirmation that the variable was updated successfully.

Reviewing Infrastructure Scope

A team needs to know which services are using custom URLs. They ask their agent to run `list_domains`, which quickly provides a comprehensive list of all registered domains and verifies if the SSL certificates are ready.

Patterns to Avoid

Checking status manually

✗ AVOID

Opening the Railway dashboard, navigating to 'Services,' clicking on the specific service, then scrolling down through deployment logs until finding the desired information.

✓ INSTEAD

Ask your agent directly. You can use `list_deployments` or check domain statuses with `list_domains`. This keeps you in your IDE and skips all the clicks.

Guessing variable names

✗ AVOID

Trying to remember if a variable is scoped at the project, service, or environment level, leading to failed configuration attempts.

✓ INSTEAD

Before setting anything, run `list_variables`. This command shows you all existing variables and clearly defines their scope, preventing accidental misconfiguration.

Ignoring persistent storage

✗ AVOID

Assuming that when a service restarts or deploys, its data is still there without checking the volume status.

✓ INSTEAD

Use `list_volumes` first. This tells you exactly which services rely on persistent storage and how much space they are using before any deployment starts.

The Right Fit

Use this MCP if your workflow involves frequent, repetitive operations across different environments (dev, staging, prod) that require checking status or modifying configuration details. If you constantly find yourself switching between the Railway web UI and your terminal, this is for you. Don't use it just because you need to see a list of projects; use specialized tools if you only need basic directory listing functionality.

However, don't use it if your primary need is code generation or complex data transformation outside of infrastructure context. For pure code suggestions, stick with standard AI coding assistants. This MCP is purely for operational control and auditing. It doesn't write code; it executes commands against your existing cloud setup.

The Pain of Dashboard Overload

Today, updating a simple variable or checking if a domain is live means logging into the Railway dashboard. You click on the project, then select the environment, navigate to 'Variables' just to check a value, and finally open another tab to view deployment logs. It’s a frustrating cycle of clicking through multiple menus.

With this MCP, you don't touch the browser. You simply ask your agent: 'What is the SSL status for my main API domain?' or 'Show me the latest build failure details.' The answer comes back in plain text, right where you are working.

Getting full operational visibility with Railway MCP

Before this, auditing a project's state meant running multiple commands—one for services, one for volumes, and another to list all environments. It was slow, manual, and error-prone.

Now, you ask the agent to 'Audit the whole API stack.' The system combines `list_services`, checks associated persistent storage via `list_volumes`, and validates deployment statuses in a single conversational response. You get full visibility instantly.

Railway Alternative: 10 Tools for Ops Management

These tools let your agent perform every essential operation on your cloud infrastructure, from listing projects to managing persistent storage.

#	TOOL	DESCRIPTION
01	`delete_variable`	Permanently removes an environment variable from a service, which stops it from being used in future deployments.
02	`list_deployments`	Retrieves the full deployment history for a service, showing status, timestamps, and IDs.
03	`list_domains`	Checks which custom domains are configured for a service and verifies their SSL certificate status.
04	`list_environments`	Lists the operational environments (like production or staging) configured within a given project.
05	`get_project`	Retrieves detailed information about a single specific Railway project using its ID.
06	`list_projects`	Lists every cloud project associated with your account, giving you the starting point for all operations.
07	`list_services`	Shows all deployable units, such as web apps and databases, for a specified environment in a project.
08	`set_variable`	Sets an environment variable value that will be available to all deployments of a specified service.
09	`list_variables`	Lists environment variables for a service, indicating their scope (service, environment, or project).
10	`get_viewer`	Use this to verify which account the API token belongs to. Get current authenticated Railway user details
11	`list_volumes`	Lists persistent storage volumes, showing their size and the services they are attached to.

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 Railway projects and their services.

I found 3 projects: 'my-api' (2 services: api-web, postgres), 'frontend-app' (1 service: nextjs), and 'worker-service' (1 service: redis). Would you like the deployment history for any specific service?

U Set the DATABASE_URL variable for my api-web service in production.

Done! I've set the `DATABASE_URL` environment variable for the api-web service in the production environment. The new value will be available on the next deployment.

U What's the deployment status of my api-web service?

The api-web service has 4 deployments. The most recent one deployed successfully 2 hours ago with image tag v2.3.1. The previous deployment (v2.3.0) failed due to a missing environment variable. Would you like more details?

Frequently Asked Questions

01 **How do I use the list_projects tool with Railway MCP?**

You simply ask your agent to 'List all my Railway projects.' The system uses `list_projects` and returns a clean list of every project name, allowing you to select the right one for subsequent actions.

02 **Does Railway MCP support environment variable deletion?**

Yes. If you need to remove an old or deprecated key, your agent can run `delete_variable` after confirming the service ID, environment ID, and variable name with you.

03 **Can I check deployment status using list_deployments?**

Absolutely. By running `list_deployments`, your agent gives you a clear history of every attempt, showing success or failure statuses and when they occurred for the specific service.

04 **What is the difference between listing services and list_environments?**

Use `list_projects` first to see all your projects. Then you use `list_environments` on a project to narrow down to 'staging' or 'production,' before finally running `list_services` for that specific environment.

05 **Does Railway MCP show me which services are using volumes?**

Yes. Running the `list_volumes` tool shows you every persistent volume, and crucially, it links each volume ID to the specific service that is relying on that data.

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 `"railway-alternative": { "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

Railway 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 Railway. 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	Railway MCP
Server ID	`019d8474-6282-712f-bff9-e3bd0a47f2d9`
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/railway-alternative.