MCP SERVER | NO CODE | CLOUD HOSTED

RunPod MCP

Manage High-Powered Compute Through Conversation

RunPod MCP lets your AI agent act like a DevOp engineer right inside your chat window. You can provision GPU pods, check active instances, and manage serverless endpoints for compute-intensive tasks without touching a dashboard. It's instant infrastructure control.

A+ Quality Score 100/100

gpu-computing | serverless-deployment | cloud-instances | machine-learning-ops | container-orchestration | infrastructure-as-code

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

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.

RunPod MCP

7 tools available
Cloud-hosted on Vinkius

Need to run heavy machine learning models or complex computational workflows? This MCP connects your agent directly to RunPod, giving it full command over scalable GPU computing resources. Instead of logging into a cloud console and clicking through menus to get what you need, you just ask for it. You tell the system to spin up a specific type of hardware or check if any current pods are running idle—and it handles the rest. It's about treating infrastructure management like another natural language task. By connecting this RunPod MCP through Vinkius, your agent gains immediate access to professional-grade DevOp tools. You can audit all serverless endpoints and quickly provision new hardware nodes right from a simple conversation. It means you get the power of complex cloud orchestration without leaving your chat interface.

Core Capabilities

01 — Provisioning New Hardware

You instruct the system to build entirely new GPU pods using specified types and Docker images.

02 — Managing Running Instances

You check details for specific pods or halt running instances immediately to prevent unnecessary billing costs.

03 — Inventorying Resources

The agent lists every active pod, available GPU hardware type, and saved deployment template in your account.

04 — Auditing Deployments

You review all registered serverless endpoints that are routing containerized inference applications.

One Click on Vinkius — From Prompt to Execution

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

01 First, enable the RunPod orchestration integration inside your core interface.

02 Next, sign into your RunPod cloud console and generate a new API Key with Read/Write permissions; insert this key into the secure connection module below.

03 Finally, ask your agent to perform an action, like 'List all active GPU pods and point out any that are sitting idle without active usage.'

The bottom line is, you get chat-based control over mission-critical cloud resources.

Built For

This MCP is built for the engineer who spends too much time clicking through complex web dashboards just to manage compute. It's ideal for DevOps Engineers and ML Developers who need instant, reliable control over high-powered hardware resources without switching context or opening a new browser tab.

DevOps Engineer

Manages the full lifecycle of cloud infrastructure. They use this to provision and audit heavy workloads directly from chat, eliminating dashboard toggling.

ML Developer

Needs to quickly deploy and test high-power serverless LLM implementations. This allows them to manage complex compute resources using natural language requests.

What Changes When You Connect

01 Control costs instantly: You can use the `stop_pod` tool to halt running instances immediately, preventing unexpected hourly charges.

02 Provision hardware on demand: Instead of browsing menus, you simply ask your agent to create a new pod using `create_pod`, specifying exactly what GPU type and image you need.

03 Audit infrastructure easily: Use `list_endpoints` to review every single serverless endpoint connected to your application without opening any management consoles.

04 Understand options quickly: If you're unsure what hardware to use, the agent can run `list_gpu_types` to show all available GPU architectures for deployment.

05 Manage templates efficiently: Need a standard setup? Use `list_templates` to see your saved configurations and reference them when provisioning new resources.

Real-World Applications

Stopping an expensive test run

An ML developer finishes testing a model locally but forgot the remote pod is still running. They prompt their agent: 'Pause pod with ID X immediately to prevent recurring costs.' The agent uses `stop_pod` and confirms billing operations are halted.

Deploying a new service version

A DevOps engineer needs to test a containerized inference app. They ask the agent to list available templates, use `list_templates`, select the right one, and then run `create_pod` with minimal effort.

Checking hardware capacity

A team lead needs to know what kind of GPU power is even possible. They ask the agent which types are available, triggering a call to `list_gpu_types` so they can plan their next big deployment.

Reviewing current deployed services

A developer wants to see every live API connection point for their app. They instruct the agent to check all endpoints, using `list_endpoints`, ensuring nothing critical has been forgotten or misconfigured.

Patterns to Avoid

Over-relying on manual dashboards

✗ AVOID

A user spends 15 minutes navigating the RunPod web console, clicking through 'Pods,' then 'Templates,' and finally trying to find the correct API key section just to check a pod's status.

✓ INSTEAD

Just ask your agent: 'List all active GPU pods.' The tool handles checking `list_pods` and giving you an immediate answer without you ever leaving your chat window.

Manually copying API keys

✗ AVOID

The user gets frustrated, copies a key from their console, pastes it into the connection module, and then has to manually re-verify permissions every time they switch projects.

✓ INSTEAD

Follow the setup guide: generate the new Read/Write API Key once in the RunPod console and insert it securely. The agent uses this credential automatically for all subsequent calls.

Trying to manage billing without scope

✗ AVOID

The user sees a list of pods but doesn't know which ones are actually incurring costs or if they are idle, leading to unexpected bill spikes.

✓ INSTEAD

Use the combined power of `list_pods` and asking the agent to identify any pod that is running without active usage. This immediately flags potential cost sinks.

The Right Fit

You should use this MCP if your primary pain point is managing complex, stateful infrastructure (like GPUs or serverless functions) via a chat interface. Specifically, you need to provision resources (`create_pod`), audit existing connections (`list_endpoints`), or manage costs by shutting down instances (`stop_pod`). Don't use it if all you need is simple data retrieval from a single source; for that, a basic read-only tool might suffice. This MCP is overkill if your goal is just to write code—it's about managing the *platform* where the code runs. If your core workflow involves iterating between cloud consoles and your AI agent, this is exactly what you need.

Managing Cloud Infrastructure Shouldn't Feel Like a Web Debugging Session

Today, checking on your computational resources means jumping through hoops. You open the cloud console, find the 'Instances' tab, click into the pod ID to check its status, then maybe you have to go back to a separate 'Templates' section just to see what hardware types are even available. It's clicking, copy-pasting IDs, and context switching until your fingers hurt.

With this MCP, that whole manual process collapses. You tell the agent exactly what you need—say, 'Show me all running GPU pods.' And in a single breath, it gathers all the necessary data, presenting you with a clean report without you ever having to click more than once.

RunPod MCP: Instant Pod Management

You no longer need to manually list pods and then use another dashboard to find their status. Instead, you simply ask the agent to 'List all GPU pods in the account,' and it runs the `list_pods` tool instantly.

It's a fundamental shift: your AI client treats infrastructure management as conversational data requests, making high-power computing accessible without needing a DevOps PhD just to check status.

RunPod MCP: 7 Tools for Cloud Compute

Use these tools to orchestrate everything from listing available GPU types to provisioning and stopping live computing pods.

#	TOOL	DESCRIPTION
01	`create_pod`	This tool builds a new GPU pod by specifying the desired name, type of GPU hardware, and Docker image.
02	`get_pod`	It pulls up specific details for one particular GPU pod you want to check on.
03	`list_endpoints`	The agent compiles a list of every registered serverless endpoint in the account.
04	`list_gpu_types`	It shows you all the different GPU hardware types that are currently available for deployment.
05	`list_pods`	This tool generates a comprehensive list of every GPU pod in your account.
06	`list_templates`	It retrieves all the saved pod templates you've configured previously.
07	`stop_pod`	You use this to halt a running GPU pod instance, which cuts off billing for that specific compute target.

See It in Action

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

U Show me our stopped GPU pods.

I successfully verified the RunPod platform logs. You have 2 pods currently in a paused stopped status in your configured account.

U Check what GPU templates are available to deploy a new Llama-3 inference instance.

I have loaded the RunPod catalog template arrays. There are several pre-built images with focused PyTorch and vLLM installations tuned perfectly for Llama-3 text deployments. Would you like me to provision one specific GPU?

U Pause pod with ID 'pod_xyz_980' immediately to prevent recurring costs throughout the evening.

Pod 'pod_xyz_980' has been carefully stopped securely. Active hourly billing operations to compute cycles for this specific cloud target are halted.

Frequently Asked Questions

01 **How do I use the RunPod MCP to provision hardware?**

You instruct your agent using `create_pod`. You'll need to specify the name you want for the pod, the GPU type, and the Docker image. The agent handles building the instance for you.

02 **Can I use RunPod MCP to stop a running pod?**

Yes. If you need to halt an expensive running pod immediately, just ask your agent to run `stop_pod` with the specific ID. This is key for controlling costs.

03 **Does RunPod MCP list all my templates?**

Absolutely. Use the `list_templates` tool name to see every saved pod template you have configured, helping you reuse successful setups quickly.

04 **What if I need a different type of GPU? How do I find it using RunPod MCP?**

You can check all available options by running `list_gpu_types`. This gives you the definitive list of hardware types that your agent can use for provisioning.

05 **Is RunPod MCP only good for LLMs?**

No. While great for LLMs, it handles all computational workloads—from general ML training to running any containerized inference application via `list_endpoints`.

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 `"runpod": { "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

RunPod 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 RunPod. 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	RunPod MCP
Server ID	`019d7601-08c5-7349-b02e-b54e23527f25`
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/runpod.