MCP SERVER | NO CODE | CLOUD HOSTED

# Infracost MCP

## Govern Cloud Spending and Tagging Policies Automatically

Infracost provides cloud cost estimates for Terraform workflows. It lets your AI agent manage cost guardrails, enforce tagging policies, and apply custom enterprise pricing directly from your CI/CD pipeline. Stop unexpected cloud bills before they hit production.

C Quality Score 78.04/100

terraform | cloud-cost | finops | cost-estimation | governance | pull-request

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

SCAN TO VIEW

# The infrastructure that powers AI agents in the real world.

Vinkius connects AI to the world's software through secure, enterprise-grade infrastructure — enabling real-world execution at scale, built on the Model Context Protocol (MCP).

# Your AI Connections Run Through Vinkius Cloud

The world's largest managed MCP catalog

Vinkius is the cloud infrastructure where AI agents connect 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.

# Infracost MCP

13 tools available
Cloud-hosted on Vinkius

Need to make sure your infrastructure changes don't bankrupt the company? This MCP connects financial governance right into your code reviews. When you run a Pull Request, it checks projected costs against predefined limits, blocking the merge if spending spikes too high. Beyond simple alerts, you can enforce organizational standards by updating tagging policies, guaranteeing every cloud resource is properly attributed to a department or project. If your company has negotiated discounts with AWS or Azure, this MCP lets you manage those custom price books so cost estimates reflect reality, not standard rates. You can even upload raw CSV data containing business properties from systems like ServiceNow, mapping vague cloud charges to your actual internal organizational hierarchy. All of this governance is accessible through Vinkius and available to any compatible AI client.

## Core Capabilities

**01 — Enforce spending limits on code changes**

You can list or create guardrails that automatically alert or block Pull Requests when the proposed infrastructure change exceeds a defined cost threshold.

**02 — Maintain resource attribution standards**

The agent updates tagging policies, ensuring every piece of cloud infrastructure is tagged correctly for accurate department or project billing.

**03 — Apply negotiated enterprise pricing**

You can create and manage custom price books using your company's specific discounted rates for major cloud providers like AWS and Azure.

**04 — Map costs to internal business units**

Upload raw data from external systems, allowing you to map general cloud expenses directly into your organization's cost center structure.

## One Click on Vinkius — From Prompt to Execution

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

01 Subscribe to this MCP and enter your Infracost API Token.

02 Connect your AI client (like Claude or Cursor) to the Vinkius catalog.

03 Tell your agent what you need—for instance, 'List all cost guardrails' or 'Update the tagging policy for Environment'—and it executes the governance action.

The bottom line is that you get real-time financial control over infrastructure changes without having to switch context from your coding environment.

## Built For

This MCP is for FinOps teams and DevOps engineers who hate getting caught off guard by massive cloud bills. It's for the engineer tired of running manual cost checks across dashboards before every single merge.

**DevOps Engineer**

They use this MCP to automate enforcement of cost policies and tagging standards directly within their CI/CD pipelines, ensuring no code deploys without budget clearance.

**FinOps Analyst**

They leverage the custom price book features to gain visibility into real-time infrastructure spend while accounting for negotiated enterprise discounts.

**Engineering Manager**

They use this MCP to set budget thresholds and receive automated summaries of cost impacts whenever a team proposes major changes to the cloud architecture.

## What Changes When You Connect

01 Prevents unexpected costs from hitting production. By using the `create_guardrail` tool, your agent blocks Pull Requests if cost increases exceed your defined thresholds.

02 Enforces compliance without manual checks. You use `update_tagging_policy` to ensure every resource is correctly tagged with department and project IDs.

03 Uses real-world pricing data. The MCP lets you manage custom enterprise discounts via `create_price_book`, so cost estimates reflect your actual negotiated rates, not public cloud prices.

04 Connects cloud spend to internal budgets. Upload raw data using `upload_custom_properties` to map technical costs directly into your company's financial hierarchy.

05 Saves time on audits. Instead of digging through multiple dashboards, you can programmatically query pricing data with `query_pricing` to compare resource types across regions.

## Real-World Applications

**Preventing a runaway microservice deployment**

A DevOps engineer submits a PR for a new service. The agent runs a check, sees the cost increase exceeds $200/month, and automatically blocks the merge, alerting the team that they need to optimize before deploying.

**Auditing resource ownership after a merger**

A FinOps analyst needs to assign costs from a newly acquired subsidiary. They use `upload_custom_properties` to ingest raw CSV data, successfully mapping the new cloud expenses into the existing corporate cost center structure.

**Ensuring compliance before launch**

An Engineering Manager updates the tagging policy using `update_tagging_policy`, making 'Environment' mandatory. The agent now rejects any resource creation that doesn't carry the correct operational tag, enforcing governance from day one.

**Verifying accurate cost projections**

A team is planning a multi-region rollout. They use `query_pricing` to programmatically compare costs between three different regions and two resource types, verifying the best financial approach before committing to architecture.

# Patterns to Avoid

### Manual cross-sheet cost tracking

✗ AVOID

Spending hours each month copying cloud billing data into a massive Excel spreadsheet just to calculate departmental burn rate and manually compare it against internal budget codes.

✓ INSTEAD

Use the MCP's ability to upload custom business properties via `upload_custom_properties`. This links raw cost records directly to your organization's hierarchy, automating the mapping process.

### Relying on default public pricing

✗ AVOID

Assuming that standard cloud provider pricing is always accurate when calculating costs for a new project, missing out on significant negotiated enterprise discounts.

✓ INSTEAD

Use `create_price_book` and then run the ingestion process. This ensures all cost estimates are calculated using your company's actual discounted rates.

### Catching budget overruns too late

✗ AVOID

Waiting until a billing alert comes in weeks later, only to find out that a single PR deployment caused an unexpected $10,000 spike.

✓ INSTEAD

Implement `create_guardrail`. This forces the cost check into your CI/CD pipeline, stopping financial overruns at the moment of code commit.

## The Right Fit

Use this MCP if your primary pain point is linking infrastructure code changes to verifiable spending limits and internal accounting structures. If you need to stop costs from spiraling out of control during development or deployment, this is essential. However, don't use it if your goal is general data transformation or purely abstract business logic. For instance, if you just want to read a financial report written in PDF format and summarize the text, you need a document parsing tool, not one that manages Terraform guardrails. If your problem is merely querying historical billing records without enforcing any current rules, basic reporting tools might suffice; but if governance (guardrails or tagging) is required, this MCP handles it.

## The headache of cloud spending visibility

Today, managing infrastructure costs feels like playing whack-a-mole. You wait for the monthly bill to drop, then spend days manually cross-referencing service IDs against internal project codes in a massive spreadsheet. If you change one microservice or deploy one feature flag, nobody knows if that small commit will trigger an unexpected 30% cost spike because tagging was forgotten.

With this MCP, the process changes entirely. When code hits the review queue, the agent instantly checks the proposed deployment against your established limits and custom pricing rules. You get a pass or fail on budget status before anyone even merges the pull request. It’s governance built right into your workflow.

## Infracost: Governance Built into Your Workflow

The manual steps that disappear are endless dashboard refreshes, copy-pasting resource tags between different tickets, and the painful cycle of running ad-hoc cost reports just to answer 'who owns this budget?'

Now, you tell your agent what needs to happen—like updating tagging policies or creating a new guardrail. The MCP executes complex governance actions instantly, giving you certainty that every resource deployed meets both technical requirements and financial policy.

# Infracost: 13 Tools for Cloud Governance

These tools let you programmatically create, modify, and query every aspect of your cloud cost governance model, from setting budget limits to updating tagging policies.

| # | TOOL | DESCRIPTION |
|---|---|---|
| 01 | `list_guardrails` | Retrieves a complete list of all active spending guardrails for the entire organization. |
| 02 | `query_pricing` | Runs complex GraphQL queries against the Cloud Pricing API to compare costs across different resource types and regions. |
| 03 | `create_guardrail` | Sets up a new cost guardrail rule that will monitor spending changes in your environment. |
| 04 | `create_price_book` | Initiates the process to create and manage a custom, enterprise-specific price book for cloud billing. |
| 05 | `delete_guardrail` | Permanently removes an existing cost guardrail rule from your organization's policy set. |
| 06 | `get_guardrail` | Retrieves the specific details of a single, existing guardrail by its unique ID. |
| 07 | `get_price_book_files_url` | Generates a secure URL that you must use to upload your custom price book CSV file. |
| 08 | `get_price_book` | Checks the current status and details of an uploaded price book, waiting for successful completion. |
| 09 | `ingest_price_book` | Triggers the actual processing and ingestion of a newly uploaded custom price book CSV file. |
| 10 | `update_guardrail` | Modifies parameters, thresholds, or conditions on an already active cost guardrail rule. |
| 11 | `update_organization` | Updates core organizational settings, such as activating a previously created custom price book ID. |
| 12 | `update_tagging_policy` | Changes the rules and allowed values within an existing tagging policy for cloud resources. |
| 13 | `upload_custom_properties` | Accepts raw CSV data to upload custom business properties, linking vague costs to internal departments or projects. |

# See It in Action

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

**U** List all cost guardrails for the organization 'acme-corp'.

I've retrieved the guardrails for 'acme-corp'. You have two active rules: 1. A $500 increase threshold that blocks PRs, and 2. A $5,000 total monthly budget alert.

**U** Create a new guardrail for 'acme-corp' that triggers when cost increase exceeds $200 and blocks the PR.

Successfully created the guardrail. Any Pull Request in 'acme-corp' that increases monthly costs by more than $200 will now be automatically blocked until approved.

**U** Update the tagging policy for 'acme-corp' to make the 'Environment' tag mandatory.

I've updated the tagging policy for 'acme-corp'. The 'Environment' tag is now marked as mandatory for all resources managed under this policy.

# Frequently Asked Questions

01 **How does Infracost use the `create_guardrail` tool?**

The `create_guardrail` tool lets you set specific cost thresholds. You define rules that automatically block a Pull Request or send an alert if infrastructure changes exceed your defined dollar amount.

02 **What is the difference between `update_tagging_policy` and `upload_custom_properties`?**

Updating a tagging policy sets the rules for *what* tags must exist (like 'Environment' or 'Owner'). Uploading custom properties uses raw CSV data to map existing, complex cloud costs to your specific internal cost centers.

03 **Can Infracost help me with my enterprise discounts?**

Yes. You use the `create_price_book` tool to manage a custom price book. This ensures that all cost estimates reflect your company's negotiated, discounted rates for AWS and Azure.

04 **Which tool should I use if I need to check existing rules?**

Start with `list_guardrails` to see everything active. If you only want details on one specific rule, you can then use `get_guardrail`.

05 **Is Infracost good for complex cost comparisons?**

The `query_pricing` tool handles this by allowing you to run detailed GraphQL queries. This lets you programmatically compare costs across different regions and resource types accurately.

# 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 `"infracost": { "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

# Infracost 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 Infracost. 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 | Infracost MCP |
| Server ID | `019e38ad-7412-7334-88d7-ecb77298445d` |
| 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/infracost.