MCP SERVER | NO CODE | CLOUD HOSTED

New Relic AI (LLM Observability) MCP

Get total cost and performance metrics via conversation.

New Relic AI (LLM Observability) lets you pull performance data, token costs, and user feedback directly from your LLMs using natural conversation. Instead of logging into dashboards to check p95 latency or calculating total USD spend, you ask your agent for the metrics immediately. Track every chat completion, audit model behavior, and verify infrastructure health—all in one place.

A+ Quality Score 100/100

llm-monitoring | token-cost-tracking | performance-analytics | ai-observability | latency-tracking

MAINTAINED BY VINKIUS CLOUD FOR AI AGENTS · JUNE 2026
vinkius.com/mcp/new-relic-ai-llm-observability

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.

New Relic AI (LLM Observability) MCP

10 tools available
Cloud-hosted on Vinkius

You run complex AI agents that use Large Language Models (LLMs). Things break, costs spike unexpectedly, or performance dips when nobody is looking. This MCP connects New Relic AI to your existing agent workflow, giving you full visibility into everything happening under the hood. You can ask for total token usage across all models in dollars and cents. Need to know why responses slow down? Check the p95 latency metrics instantly. Want to audit model behavior? Review raw chat completion messages to understand exactly what the LLM saw or generated. This access means you don't have to jump between cost dashboards, performance monitoring tools, and logs just to get a complete picture. By connecting this MCP via Vinkius, your agent becomes an operational detective for your AI stack.

Core Capabilities

01 — Audit LLM Performance Metrics

Get average response times and the 95th percentile latency data to ensure your models remain fast.

02 — Track Token Expenditure

Calculate precise USD costs for all token usage across your entire AI infrastructure.

03 — Review Model Interactions

Retrieve detailed chat completion messages and original prompts to audit model behavior in real-time.

04 — Measure User Satisfaction

Fetch chronological user feedback and 1-5 rating scores provided by human supervisors.

05 — Execute Custom Queries

Run advanced, read-only queries using the New Relic Query Language (NRQL) against your AI datasets.

06 — Monitor Infrastructure Health

Examine active APM apps, dashboards, and alert policies to check overall system integrity.

One Click on Vinkius — From Prompt to Execution

Available at **vinkius.com/mcp/new-relic-ai-llm-observability** — connect your AI agent in three steps.

01 Subscribe to this MCP and enter your New Relic API Key and Account ID.

02 Connect your preferred AI client—Claude, Cursor, or any compatible agent—to Vinkius.

03 Ask a natural language question about your LLM activity. Your agent executes the necessary queries and reports back with performance metrics or cost breakdowns.

The bottom line is you talk to your agent like talking to a teammate; it handles the complex monitoring data retrieval for you.

Built For

This MCP is built for anyone who owns AI infrastructure but hates manual dashboard navigation. It's for the Observability Lead who needs global token cost visibility, or the AI Engineer who gets frustrated having to check ten different logs just to debug a slow prompt.

AI Engineer

You use this MCP to verify model accuracy and check prompt performance by pulling raw chat completion messages directly into your conversation flow.

Observability Lead

You track global token costs and latency benchmarks in real-time, allowing you to optimize infrastructure spending without leaving your primary workspace.

DevOps Team Member

You audit the structural health of your AI environment by listing active APM apps, dashboards, and checking alert policy configurations across multiple services.

What Changes When You Connect

01 Stop guessing about spending. Use `query_llm_costs` to get the exact dollar amount of your token usage, giving you tight control over infrastructure spend.

02 Debug slowness fast. Running `query_llm_latency` provides p95 latency matrices and average response times so you know exactly when your LLM generation is dipping below acceptable speed.

03 Audit model behavior instantly. Instead of digging through raw logs, use the agent to retrieve detailed chat completion messages, allowing you to verify what the LLM saw or generated.

04 Measure quality with real data. `query_llm_feedback` pulls in human supervisor ratings and feedback messages, letting you spot quality regressions immediately after deployment.

05 Stay ahead of system decay. Running `list_apm_apps` and `list_dashboards` lets DevOps check the structural health of your entire environment without leaving the chat window.

Real-World Applications

Debugging an unexpected cost spike

An AI Engineer notices their LLM costs are higher than normal. They ask the agent, 'What was my total token spend last week?' The agent executes `query_llm_costs` and reports that a specific integration caused a massive spike in usage, allowing the engineer to immediately pinpoint the source.

Checking user acceptance of new prompts

An Observability Lead wants to know if recent prompt changes affected quality. They ask the agent for `query_llm_feedback`. The agent pulls up a list of ratings, showing that user satisfaction dropped sharply after the change was deployed.

Validating system readiness before launch

A DevOps team member needs to ensure all monitoring is active. They instruct the agent to run `list_apm_apps` and check `list_alert_policies`. The agent confirms that all necessary applications are running and alert triggers are correctly configured.

Analyzing slow agent responses

An AI Engineer reports that sometimes the chat feels sluggish. They ask the agent to run `query_llm_latency`, which returns a matrix showing that the average response time exceeds 2 seconds during peak usage hours.

Patterns to Avoid

Over-relying on raw logs

✗ AVOID

A developer manually filters through thousands of log entries trying to find a specific token cost or latency metric for one single transaction. This takes hours and is prone to human error.

✓ INSTEAD

Instead, ask the agent to run `query_llm_costs` or `query_llm_latency`. The tool aggregates this data automatically and presents the precise metrics in plain language.

Assuming system health

✗ AVOID

The team assumes everything is fine because no alerts have triggered, without checking for underlying architectural decay. This leads to unexpected outages.

✓ INSTEAD

Run `list_apm_apps` and check `list_alert_policies`. This validates the operational status of every core component in your AI environment.

Ignoring user sentiment

✗ AVOID

The team focuses only on technical performance (latency) but misses that users are finding the output inaccurate or confusing, leading to poor adoption.

✓ INSTEAD

Use `query_llm_feedback`. This retrieves direct human ratings and comments, providing a critical layer of quality monitoring beyond just technical metrics.

The Right Fit

Use this MCP if your primary pain point is understanding the cost, performance, or user reception of your LLM agents without navigating multiple dashboards. It's essential for observability leads who need global visibility into token consumption and latency benchmarks. You must use it when you need to answer questions like 'How much did that run cost?' or 'Why was this response slow?'

Don't use this if you just need simple, single-point data retrieval (like checking a status code). For those limited checks, an API integration might suffice. However, because of its ability to consolidate metrics —from `query_llm_costs` to `list_apm_apps` —it's the superior choice for comprehensive auditing.

The Visibility Gap: Where AI Costs and Performance Go Missing

Right now, understanding your LLM stack is a nightmare. To figure out why costs spiked or why responses slowed down, you have to jump between New Relic's billing dashboards, the APM console, and raw chat logs. You spend time clicking through tabs, copying metrics, and trying to stitch together one single story: 'It cost X dollars because it was slow.'

With this MCP, that manual process vanishes. You simply ask your agent a question like, 'What's the token usage trend over the last week?' The tool runs `query_llm_costs` and provides the answer immediately in conversation, connecting performance metrics to actual dollars spent.

Get LLM Observability with New Relic AI (LLM Observability)

Manual monitoring requires checking multiple endpoints: logging into the query interface for `query_llm_latency`, going to a separate dashboard tool to check `list_dashboards`, and then manually calculating costs via an external spreadsheet. It's slow, and it's incomplete.

Now, your agent handles all of that complexity. You get instant access to performance data, error logs (`query_llm_errors`), and resource usage checks—all through a single chat interface. Your focus shifts from dashboard maintenance to making the AI better.

New Relic AI (LLM Observability): 10 Tools

Use these tools to manage everything from calculating precise LLM token costs and checking system latency to auditing user feedback ratings.

#	TOOL	DESCRIPTION
01	`list_alert_policies`	Checks all existing automated alerts configured for the system's plan math.
02	`list_apm_apps`	Retrieves a list of currently running APM applications to validate service status.
03	`custom_nrql`	Runs sophisticated, read-only queries using the New Relic Query Language (NRQL) for deep data insights.
04	`list_dashboards`	Finds all active operational dashboards tied to native Gateway authentication.
05	`query_llm_errors`	Identifies and lists specific error logs related to LLM processing.
06	`query_llm_costs`	Calculates the precise monetary cost of tokens used by your agents over a specified period.
07	`query_llm_events`	Retrieves bounded records tracking general activity within the New Relic platform.
08	`query_llm_feedback`	Gathers human-submitted feedback and rating scores associated with LLM outputs.
09	`query_llm_latency`	Measures the speed of your LLMs by retrieving p95 latency matrices and average response times.
10	`post_custom_event`	Sends custom telemetry rows to track unique internal states or behaviors within your agent workflow.

See It in Action

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

U Show me the last 5 LLM events for the 'OpenAI' vendor

Retrieving LLM events... I've identified 5 recent messages through the OpenAI module. Highlights: 1) Chat completion 'task-123' (Tokens: 1,500, Cost: $0.03), 2) Prompt 'User Query Alpha' (Status: Success). Would you like to see the literal prompt text for any of these?

U What is my total LLM token cost for the last 24 hours?

Retrieving cost metrics... Your total LLM token spend for the last 24 hours is $12.45. This is distributed across 'gpt-4o' ($8.50), 'gpt-3.5-turbo' ($2.45), and 'claude-3-sonnet' ($1.50). Spend is trending 5% lower than yesterday.

U Run NRQL: SELECT count(*) FROM LlmEvent WHERE duration > 2 SINCE 1 hour ago

Executing custom NRQL query... I've identified 12 LLM events in the last hour that exceeded 2 seconds in duration. This represents approximately 3% of your total traffic. Would you like me to facet these slow events by model or region?

Frequently Asked Questions

01 **How does New Relic AI (LLM Observability) track token costs?**

This MCP uses `query_llm_costs` to calculate your total LLM token spend. It gives you the exact USD consumption across different models and services, so you never lose money tracking usage.

02 **Can I check my LLM performance latency with this MCP?**

Yes, use `query_llm_latency`. It pulls p95 latency matrices and average response times, helping you pinpoint exactly when your agent's responses slow down.

03 **What kind of data can I audit with New Relic AI (LLM Observability)?**

You can audit everything: chat completion messages for model behavior, human supervisor feedback using `query_llm_feedback`, and raw internal agent states via `post_custom_event`.

04 **Is New Relic AI (LLM Observability) read-only?**

Yes. The tool uses mechanisms like `custom_nrql` which are strictly read-only queries, meaning you can pull insights without risking any changes to your live infrastructure.

05 **Does this MCP help with general system health checks?**

It does. You can use tools like `list_apm_apps` and `list_alert_policies` to check the operational status of your entire environment, not just the LLM component.

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 `"new-relic-ai-llm-observability": { "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

New Relic AI (LLM Observability) 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 New Relic AI (LLM Observability). 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	New Relic AI (LLM Observability) MCP
Server ID	`019d75dc-e7ba-70bb-8f02-309d5f2787c7`
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/new-relic-ai-llm-observability.