MCP SERVER | NO CODE | CLOUD HOSTED

Lago MCP

Automate Complex Usage Billing Logic

Lago connects your AI agent to robust metering and usage-based billing. Handle complex pricing models—including custom plans, prepaid credits, and real-time consumption tracking—without leaving your chat interface.

A+ Quality Score 100/100

usage-based-billing | metering | saas-metrics | subscription-management | revenue-operations | api-integration

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

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.

Lago MCP

12 tools available
Cloud-hosted on Vinkius

This MCP lets you automate the entire cycle of usage-based revenue operations. Forget juggling multiple dashboards or writing custom scripts every time a customer consumes resources; you manage it all through natural language. You can create new pricing plans, assign them to customers, and even set up prepaid credit wallets for immediate use. When activity happens, you send simple usage events that trigger accurate billing calculations in real-time. Need to correct something? Your agent handles applying coupons or generating detailed invoices instantly. Connecting Lago via the Vinkius catalog makes your entire revenue stack available from one place, letting you manage everything—from customer setup to final invoice generation—directly through your preferred AI client.

Core Capabilities

01 — Manage Customer Profiles

Create or update full customer records and retrieve existing details using a unique external ID.

02 — Control Subscriptions and Plans

Set up new billing plans, assign those plans to customers, or pull up the status of any active subscription.

03 — Track Usage Data for Billing

Send single usage events or large batches of data points that automatically trigger consumption and billing updates.

04 — Handle Financial Operations

Generate invoices, create coupon codes, or top up customer prepaid credit wallets.

05 — View Billing Metrics

Pull specific data points about what can be billed, ensuring your usage tracking is accurate before billing cycles end.

One Click on Vinkius — From Prompt to Execution

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

01 First, subscribe to this MCP and provide your API key within your AI client.

02 Next, tell your agent exactly what you need—like 'Create a new plan for 100 units at $0.05' or 'Send usage data for the last hour'.

03 Finally, the MCP executes the billing action in Lago and provides confirmation of the change.

The bottom line is that you talk to your agent like talking to a finance analyst, and it handles all the complex API calls needed to manage billing logic.

Built For

This connector is built for Revenue Operations Managers, SaaS Founders, and Finance Analysts. If your business relies on usage-based billing (where costs change based on consumption), you'll need this. It solves the pain of manually reconciling complex usage logs across different systems.

SaaS Founder

Needs to quickly check a customer's subscription status or adjust their plan via natural conversation without logging into a separate billing dashboard.

Revenue Operations Manager

Automates the creation of promotional coupons, managing credit wallets, and ensuring usage events are logged correctly for immediate billing purposes.

Finance Analyst

Retrieves historical invoices or verifies billable metrics for quarterly reporting by simply asking the AI agent.

What Changes When You Connect

01 Stop manual billing calculations. You can send usage data via the `send_event` or `batch_events` tools, making sure your customers are billed instantly and accurately based on real-time consumption.

02 Manage customer finances without leaving chat. Use `create_wallet` to issue prepaid credits and `apply_coupon` to run promotions instantly, all through natural language queries.

03 Build complex pricing models easily. The `create_plan` tool lets you define new tiers (e.g., 'Tier 2: $0.10/unit') that your customers can subscribe to using `create_subscription`.

04 Maintain clean data integrity. Use `upsert_customer` and `get_customer` together to ensure every customer profile is accurate before running any billing operations.

05 Simplify reporting. Your agent handles listing all invoices via `list_invoices`, giving you immediate visibility into outstanding revenue without needing a dashboard view.

Real-World Applications

A new feature was released and needs immediate metering.

The growth engineer knows the service is going live. They ask their agent to 'Create a billable metric called API_CALLS' using `create_billable_metric`. This ensures that when users start making calls, those specific actions are immediately accounted for in billing.

A large corporate client needs an immediate discount.

The sales rep asks the agent to 'Apply a 25% coupon to Acme Corp'. The agent uses `apply_coupon` and confirms the discount, ensuring the revenue team can process the change instantly.

End-of-quarter billing reconciliation is due.

The finance analyst simply asks the agent to 'List all invoices for Q2'. The agent runs `list_invoices` and provides a summarized view, eliminating hours of spreadsheet work.

A beta user needs immediate access but has no paid plan.

The product manager asks the agent to 'Create a $50 credit wallet for Beta User 789'. The agent uses `create_wallet`, giving the user instant, pre-paid access while the billing team finalizes their subscription.

Patterns to Avoid

Manual Usage Logging

✗ AVOID

Copying usage counts from a separate metrics dashboard into a spreadsheet and manually calculating charges. This is slow, prone to errors, and misses real-time events.

✓ INSTEAD

Instead of logging data manually, use the `batch_events` tool. You send all historical or accumulated usage data in one go, letting Lago calculate and record it automatically.

Ignoring Customer Updates

✗ AVOID

When a customer changes emails or names, you have to find their profile ID first, then manually update the details in a separate system. This often leads to orphaned records.

✓ INSTEAD

Use `upsert_customer`. You give your agent the external ID and the new data; the tool either updates the existing record or creates it if it's brand new.

Hardcoding Pricing Changes

✗ AVOID

If you need to change a pricing tier from $0.10/unit to $0.09/unit, you have to contact an engineer and deploy code changes just for that one rule.

✓ INSTEAD

Use the `create_plan` tool through your agent. You define the new billing parameters naturally in conversation, and the system updates the pricing logic immediately.

The Right Fit

You need this MCP if your business charges customers based on variable usage (like API calls, stored data volume, or transactions) and requires complex financial rules (discounts, credits). You must use it because standard payment gateways only handle simple transaction processing, not the underlying metering logic. Don't use

Lago if all you do is charge a flat $9.99 per month; for that, a simpler subscription service might suffice. However, if your pricing structure includes coupons (`apply_coupon`), credit wallets (`create_wallet`), or usage-based tiers, this MCP handles the entire financial lifecycle from one place.

The Struggle of Billing Reconciliation

Today, tracking revenue is a painful process. You jump between your usage dashboard, your customer database, and your billing system. If you need to check what a specific client used last month, you spend time finding the right ID, cross-referencing the usage logs, then manually calculating if they hit a discount threshold. It's clicking through tabs until your eyes bleed.

With this MCP, your agent handles that whole mess in seconds. You simply ask it to 'Show me the billing details for Client X.' The process—retrieving the customer record, pulling the usage data, and calculating the final charge—happens automatically when you use `get_customer` or `list_invoices` . It’s instant context.

Automating Billing Logic with Lago

The manual steps that disappear are the data synchronization and the calculation itself. You no longer have to manually send usage counts or update customer plans across multiple interfaces; you just tell your agent what needs to happen.

What's different now is control. You maintain complete oversight of every billing action—from setting up a new `create_plan` to applying a coupon with `apply_coupon` —all through simple conversation. The system handles the complexity, and you get clear results.

Lago with 12 Tools

These tools give your agent direct access to all core functions of a modern metering and billing platform.

#	TOOL	DESCRIPTION
01	apply_coupon	Applies a specific coupon code to an existing customer's account.
02	batch_events	Sends multiple usage events at once, useful for syncing historical data or large batches of consumption records.
03	create_billable_metric	Defines a new type of resource that can be measured and charged against customer usage.
04	create_coupon	Generates a new, usable coupon code for your customers to redeem.
05	create_plan	Builds and saves an entirely new billing plan structure within the system.
06	create_subscription	Assigns a specific, pre-existing pricing plan to a customer account, activating their subscription.
07	create_wallet	Creates a dedicated wallet balance for prepaid credits or stored value for any user.
08	get_customer	Retrieves all current profile details for a customer using their unique external ID.
09	get_subscription	Fetches the full status and details of any active subscription account.
10	list_invoices	Pulls a list of all past and pending invoices associated with your accounts.
11	send_event	Sends a single, real-time usage event immediately to trigger billing updates.
12	upsert_customer	Creates a new customer record or safely updates an existing one using its unique external ID.

See It in Action

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

U Create a new customer in Lago with external ID 'user_123' and email 'dev@example.com'.

I've successfully created the customer record for 'user_123'. Their profile is now active in Lago and ready for subscriptions.

U Show me the subscription details for ID 'sub_98765'.

Fetching subscription 'sub_98765'... It is currently active on the 'Premium' plan, started on 2023-10-01, and is set for monthly billing.

U List all invoices for my Lago account.

I've retrieved the latest invoices. You have 3 pending invoices for this month and 12 paid invoices from the previous quarter. Would you like to see the details for a specific one?

Frequently Asked Questions

01 **How do I handle usage-based billing with Lago?**

You use the `send_event` or `batch_events` tools to push real-time consumption data. This triggers the metering system, which accurately tracks and bills against defined metrics.

02 **Can I change a customer's plan using Lago?**

Yes, you can assign new billing plans by running `create_subscription`. You just need to know the target plan ID beforehand.

03 **What is the difference between `upsert_customer` and creating a customer?**

`Upsert_customer` is for updating records. It's safer because it tries to update an existing client first; if that fails, it creates them instead.

04 **Does Lago handle prepaid credits?**

Yes. You can set up and manage prepaid funds by using the `create_wallet` tool to issue stored value or credit to a customer account.

05 **How do I verify if I've created a metric correctly with Lago?**

You should use the `get_customer` tool first, then follow up by calling `create_billable_metric`. This lets you confirm the resource is ready to track usage.

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 `"lago": { "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

Lago 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 Lago. 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	Lago MCP
Server ID	`019e38b5-9e2d-7046-8fd1-f950abf0b129`
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/lago.

© 2026 Vinkius · vinkius.com · support@vinkius.com