MCP SERVER | NO CODE | CLOUD HOSTED

Stripe MCP

Manage every payment, customer, and invoice detail instantly.

Stripe lets your AI client take full control of your payments and billing infrastructure via conversation. Create payment intents, list customers, track invoices, and audit refunds—all from any MCP-compatible application. Check available balances, review product pricing, manage subscriptions, and audit every charge without ever leaving your IDE or chat window.

A+ Quality Score 100/100

payments | invoicing | customer-management | subscription-billing | financial-reporting | refund-management

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

Stripe MCP

13 tools available
Cloud-hosted on Vinkius

Dealing with payments usually means context switching. You open the Stripe dashboard to check a balance, then switch back to your code editor to write an API call. This MCP changes that. It connects your AI agent directly to your payment data, allowing you to manage your entire financial lifecycle using only natural language.

You can use it to build new customer profiles or simply verify if a client's account is currently past due on their invoice. Need to know the total remaining balance across multiple currencies? Ask. You get immediate access to all necessary tools. This makes the whole catalog of Vinkius available, so you don't have to write boilerplate code just to retrieve status data.

Whether you are debugging a checkout flow or running end-of-month reconciliation, your agent acts as a dedicated payments engineer, pulling everything from customer records and subscription statuses to historical charges. It’s pure data retrieval, delivered conversationally.

Core Capabilities

01 — Manage Customer Accounts

Create new user accounts or pull detailed profiles for existing customers using their ID or email.

02 — Process Payments and Intents

Generate payment tokens and track the status of payments in real time, from pending to succeeded.

03 — Review Billing History

Pull detailed lists of invoices, past charges, and refunds to reconcile financial records.

04 — Monitor Subscriptions

Check the status, billing cycle, and associated prices for all active user subscriptions.

05 — Audit Financial Status

Quickly check both available and pending balances across various currencies.

One Click on Vinkius — From Prompt to Execution

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

01 Subscribe to this MCP and input your Stripe Secret Key.

02 Your AI client uses the credentials to access all payment endpoints.

03 You prompt your agent with a request (e.g., 'List all open invoices for customer X').

The bottom line is, you get immediate, conversational access to Stripe's entire data set without writing boilerplate API calls.

Built For

This MCP serves the backend developer who needs to integrate payment logic quickly, the finance analyst running month-end reports on subscriptions and refunds, or the product manager needing real-time customer status checks.

Backend Developer

Writing checkout flows or user onboarding hooks. They use `create_payment_intent` to generate client secrets or `list_customers` to verify existing accounts before making a charge.

Finance Analyst

Reconciling month-end books. They run reports using `list_invoices`, checking `get_balance`, and reviewing historical charges via `list_charges`.

Product Manager

Analyzing subscription health or discount usage. They check the status of all user plans with `list_subscriptions` or audit promotional rates using `list_coupons`.

What Changes When You Connect

01 Audit refunds in seconds. Instead of hunting through the dashboard to find a specific transaction ID, you can run `list_refunds` directly via your agent, getting the amount, status, and date immediately.

02 Build checkout flows without leaving your chat window. Use `create_payment_intent` to generate client secrets for payment processing right where you're working, streamlining development cycles.

03 Track revenue health effortlessly. Check all active user plans using `list_subscriptions`, instantly knowing which customers are past due or still in their trial period.

04 Centralize customer data. Instead of multiple database lookups, use `get_customer` to pull a full profile—email, name, and metadata—for any client ID you provide.

05 Get instant financial oversight. Use `get_balance` to check total available and pending funds across all currencies with a single prompt.

Real-World Applications

Month-End Reconciliation

A finance analyst needs to prove that 47 subscriptions were active last month. The agent runs `list_subscriptions`, pulls the status and billing cycle for all users, and compiles a summary report detailing recurring revenue.

Debugging a Failed Payment

A developer sees a checkout fail. They ask the agent to review recent activity by calling `list_charges` using the customer ID, pinpointing the exact charge failure status and associated payment method details for debugging.

Handling Customer Support Disputes

A support team member needs to verify a refund request. They use the agent to call `list_refunds` and cross-reference that data with `get_customer` records to confirm identity and transaction history.

Product Pricing Review

A product manager wants to know which discount codes are most popular. They prompt the agent to call `list_coupons`, getting a full list of active coupons, their type (percent/amount), and remaining duration.

Patterns to Avoid

Trying to calculate total revenue piecemeal

✗ AVOID

Manually opening the 'Invoices' tab, then going to 'Subscriptions', and finally checking the 'Balances' dashboard to piece together a quarterly report.

✓ INSTEAD

Tell your agent, 'Give me a summary of all active subscriptions and any open invoices for Q2.' This single prompt will run `list_subscriptions` and `list_invoices`, providing an aggregated view.

Copying/Pasting IDs between tabs

✗ AVOID

Finding a customer's ID in the 'Customers' list, then copying it into the 'Invoices' search field, which is slow and error-prone.

✓ INSTEAD

Just tell the agent, 'Show me all invoices for customer cus_123.' The MCP handles the context switch internally, using `list_invoices` directly with the ID.

Assuming a charge is paid

✗ AVOID

Looking at an invoice and assuming because it's listed doesn't mean funds are available or that the payment succeeded.

✓ INSTEAD

Always verify the transaction status. Use `list_charges` to confirm the actual success state, rather than relying only on the invoice status.

The Right Fit

Use this MCP if your workflow requires querying specific financial data points or executing core billing actions (like creating a payment intent or listing invoices). It's built for retrieval and process control. Don't use it, however, if you need to perform UI interactions that rely on visual confirmation, such as manually approving an invoice in a complex web portal, or designing custom dashboards with advanced filtering not exposed via API endpoints. If your goal is simply data reporting—like generating simple CSV exports of customer lists—a dedicated dashboard tool might be better. But for operational actions like 'What was the status of this payment?' or 'List all active subscriptions,' this MCP is essential.

The Pain of Switching Context to Check Payments

Today, checking a client's account history means navigating away from your main work. You open the payment dashboard, find the customer ID, switch tabs to view their invoices, and then copy data into an email or report. It’s a series of clicks and context switches that slows down every conversation.

With this MCP, you talk to your agent like normal. Instead of clicking through multiple dashboards, you simply ask: 'What are the open invoices for customer X?' And boom—you get the full data set back immediately. Your AI acts as a dedicated payments expert sitting right next to you.

Stripe MCP: Instant Visibility Into Billing Cycles

The manual steps that disappear include cross-referencing billing cycles, checking if the account is past due, and manually listing product prices to confirm a quote. You stop having to copy IDs from one tab into another just to check status.

Now you get real-time visibility into every payment state, whether it's tracking an `list_payment_intents` failure or verifying that the next payout date is correct. It’s about keeping your workflow in one place.

Stripe MCP: 13 Payment & Billing Tools

These tools let your agent perform specific financial operations, allowing you to create customers, check balances, list invoices, or audit charges with precise control.

#	TOOL	DESCRIPTION
01	`create_customer`	Creates a new customer record, optionally adding name and email details for future billing.
02	`create_payment_intent`	Generates a payment intent object that requires an amount and currency to process a charge later.
03	`get_balance`	Returns the available and pending total funds, broken down by type (e.g., card, bank transfer) and currency.
04	`get_customer`	Retrieves all details for a specific customer ID, including contact info and metadata.
05	`list_charges`	Lists multiple charges, providing the amount, status (succeeded/failed), customer ID, and date of each transaction.
06	`list_coupons`	Lists all available discount coupons, detailing their type (percent or amount) and when they expire.
07	`list_customers`	Retrieves a list of customers by ID, email, or name to find records in your system.
08	`list_invoices`	Lists all invoices for a customer, showing amounts due, status (paid/open), and creation dates.
09	`list_payment_intents`	Retrieves the current state of payment intents, including their amount, currency, and required action status.
10	`list_prices`	Lists available pricing tiers for products, specifying if they are one-time or recurring subscriptions.
11	`list_products`	Retrieves the catalog of all defined products by name and description, which are used to set prices.
12	`list_refunds`	Lists historical refunds, detailing the amount, status, charge ID, and date they were issued.
13	`list_subscriptions`	Retrieves details on all active subscriptions for a customer, including their current billing cycle and status.

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 active subscriptions and their monthly revenue.

I found 47 active subscriptions. Total monthly recurring revenue is $12,450. The largest plan is the Enterprise tier at $499/mo (3 customers), followed by Pro at $99/mo (18 customers) and Starter at $29/mo (26 customers).

U Show me all refunds issued in the last 30 days.

I found 8 refunds in the last 30 days totaling $347.50. The largest was $149.00 for charge ch_3Oa... (Annual Plan cancellation). All others were under $50 each. Would you like the details for any specific refund?

U What's my current Stripe balance?

Your available balance is $8,234.50 USD and €1,120.00 EUR. Pending balance: $2,450.00 USD (3 payment intents still processing). Next payout is scheduled for tomorrow.

Frequently Asked Questions

01 **How do I find a customer ID using the Stripe MCP?**

You use `list_customers` to search for customers by name or email address. This tool returns a list of IDs, allowing you to then target that specific user with other commands.

02 **Can I see my current payout status using the Stripe MCP?**

Yes, use `get_balance`. This function provides both your available funds and any pending balances, broken down by currency and type, so you know exactly when the money hits.

03 **What if I need to list all subscriptions for a single customer?**

Use `list_subscriptions`. You provide the customer ID, and it returns every active plan, along with its status (like 'active' or 'past_due') and billing cycle details.

04 **How does Stripe MCP handle refunds?**

You use `list_refunds` to get a comprehensive list of all issued refunds. This includes the refund amount, the status (success/failed), and which original charge it was tied to.

05 **Can I check product prices before creating an invoice?**

Absolutely. Use `list_prices` to browse current pricing tiers for products, checking if they are one-time fees or recurring subscription amounts.

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 `"stripe-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

Stripe 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 Stripe. 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	Stripe MCP
Server ID	`019d8486-276e-72d3-9bed-9493ba1924e8`
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/stripe-alternative.