MCP SERVER | NO CODE | CLOUD HOSTED

Convex MCP for AI Agents

Managing Real-Time Data Transactions and Queries

Convex MCP gives your AI agent direct access to execute queries, mutations, and actions on your real-time Convex database backend. Instead of building complex integration layers, you simply ask your agent for the data or changes you need, letting it handle all the read/write logic against your live application state.

F Quality Score 8.79/100

real-time-database | serverless-functions | typescript | backend-as-a-service | acid-transactions

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

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.

Convex MCP

4 tools available
Cloud-hosted on Vinkius

You can connect your AI client straight to your Convex backend using this MCP. It means your agent doesn't just talk about your app; it actually interacts with its database and serverless functions in real time. Think of it as giving your assistant the keys to your entire data layer.

Whether you need to pull a list of user records, update a project status, or trigger heavy background computation via an action, your agent handles it through natural conversation. You write 'get all active users who signed up last month,' and the MCP executes that query against Convex's tables instantly. If you need to modify data—like changing a subscription tier—it runs the mutation with full ACID guarantees right from the chat window.

It bypasses traditional API boilerplate entirely. Everything is managed through your AI client, which communicates with this MCP hosted on Vinkius. You get immediate visibility into your application's state and control over its logic without ever leaving your preferred interface.

Core Capabilities

01 — Execute read-only queries against Convex tables

Your agent runs the run_query tool to fetch current data and application state.

02 — Commit transactional updates using run_mutation

The agent uses run_mutation to modify multiple records, ensuring all changes are consistent across your database.

03 — Trigger external logic via run_action

You can make the agent execute background actions for API calls or heavy computation using run_action.

04 — Call any function by URL identifier with run_function

The agent runs the run_function tool to call a specific Convex function regardless of its internal naming structure.

One Click on Vinkius — From Prompt to Execution

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

01 First, you subscribe to this MCP and provide your Convex Deployment URL (and access key).

02 Next, when you talk to your AI client, simply ask it for a data action—like listing all records or updating a profile.

03 The agent translates that request into the correct function call (query, mutation, or action) and executes it directly against your live Convex environment.

The bottom line is you get to interact with your full-stack data layer using plain language prompts instead of writing boilerplate API calls.

Built For

This MCP is essential for developers and product managers who need immediate, secure access to the application's core data. It solves the problem of switching context between a chat interface and an IDE just to check or update records.

Full-stack Developer

Debugging state changes or running migrations directly from your chat window instead of setting up local testing scripts.

Product Manager

Querying live application metrics and specific user data to validate product assumptions using natural language, without needing a developer to write an ad-hoc report.

Support Team Lead

Inspecting, updating, or triggering administrative actions on user accounts through a secure, controlled AI interface for rapid issue resolution.

What Changes When You Connect

01 Instant data access: Use the run_query tool to pull live documents and state without writing a single line of query language.

02 Atomic updates: The run_mutation tool guarantees that when you change user records, all related fields update together or none do at all.

03 Logic execution: Run complex side effects via run_action. Your agent can trigger external API calls directly from the chat.

04 Flexible calling: With run_function, your agent can call any exposed backend function using its URL identifier for maximum compatibility across your app.

05 Context switching eliminated: You keep debugging data and running migrations right inside your IDE or chat window.

Real-World Applications

Validating a new feature's user impact

A Product Manager needs to know how many users in the last week viewed the checkout page but never completed a purchase. They ask their agent, which uses run_query, and instantly gets a count and list of IDs they can work with.

Handling an urgent account update

A support team member discovers a user's premium status is incorrect. Instead of opening multiple admin panels, they tell their agent to run_mutation the record, and it updates the subscription tier immediately.

Running post-deployment checks

A developer needs to confirm that a new background data sync worked correctly. They prompt their agent to check specific tables using run_query, validating state without needing a local environment setup.

Triggering external webhooks from chat

The system requires an external payment service to be notified when a user reaches a certain milestone. The developer instructs the agent to use run_action, which sends the necessary API call and completes the workflow.

Patterns to Avoid

Over-relying on manual UI navigation

✗ AVOID

The support team spends 15 minutes clicking through three different dashboards to manually find a user's full activity log and subscription status.

✓ INSTEAD

Ask your agent to use run_query. It pulls the entire complex view of data in one step, giving you all necessary information instantly.

Ignoring transaction guarantees

✗ AVOID

A developer writes code that updates a user's profile and then separately logs that update, risking inconsistency if one call fails.

✓ INSTEAD

Use run_mutation. It wraps both the profile change and the log entry into one atomic operation; it either all succeeds or none of it does.

The Right Fit

You should use this MCP if your core problem involves reading, writing, or triggering logic against a centralized, real-time database like Convex. It's perfect when you need to execute backend operations (like updating user roles or running complex queries) without leaving your AI client—think of it as an API layer that speaks natural language.

However, don't use this if your process is purely about data visualization; if you just need to view a dashboard chart, stick with standard BI tools. Also, if your logic requires reading from external systems that aren't connected through Convex actions, you might need an alternative integration pattern.

Convex MCP for AI Agents: Solving Real-Time Data Access Pain

Right now, managing your application data means context switching. You're in a chat with your agent

With this MCP, that friction disappears. You simply ask your agent about Sarah Jones's status. It uses

to brainstorm, but when you need to check if the user 'Sarah Jones' actually exists or what her current subscription level is, you have to jump into the database UI or run a separate CLI command. This back-and-forth slows down every decision.

read_query and returns the data instantly, keeping your focus on conversation while giving you immediate access to live application metrics.

Convex MCP for AI Agents: Streamlining Backend Logic with Mutations

The biggest headache is multi-step data changes. If a user upgrades their plan, you must update the `users` table AND increment the `subscription_count` in another service. Doing this manually or via multiple separate API calls introduces risk and complexity.

Now, when your agent runs a mutation, it handles both updates as a single, guaranteed transaction. You tell it 'upgrade Sarah's plan,' and all dependent data changes happen correctly—you get consistent state every time.

Convex: 4 Tools for Backend Data Operations

Use these tools to run read-only queries, modify records, or trigger complex background logic directly against your Convex backend.

#	TOOL	DESCRIPTION
01	`run_action`	This tool executes defined Convex action functions to run external logic and API calls.
02	`run_mutation`	The agent uses this to modify data records, ensuring all changes are handled atomically with ACID guarantees.
03	`run_query`	This tool retrieves documents and state by calling read-only Convex query functions.
04	`run_function`	The agent calls a specific Convex function using its full URL identifier for maximum compatibility.

See It in Action

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

U What are the top five users who haven't logged in this month?

Top Inactive Users

USER ID	NAME	LAST LOGIN
jd7a...	Marcus R.	2024-10-01
a8f3...	Chloe W.	2024-09-28
b1c2...	David P.	2024-09-15

These users are flagged for an outreach email campaign.

U I need to create a new project record and assign it to the team lead, Alex.

Done. The new project, **Aurora Initiative**, has been created successfully.

- **Project ID:** p-983c
- **Status:** Draft
- **Owner:** Alex Smith (ID: a1b2...)

Please let me know if you need to run any associated actions, like notifying the team.

U Check if there are any active tasks that require review by the billing department.

Searching Convex records for pending tasks...

Found 3 items requiring attention:

1. **Task ID: t-45k**: Billing Review (Client: BetaCorp)
2. **Task ID: t-67l**: License Renewal (Client: AlphaDev)
3. **Task ID: t-90m**: Usage Audit (Client: Gamma Inc.)

You can now ask me to update the status of any of these tasks.

Frequently Asked Questions

01 **How does the Convex MCP help manage my database without writing code?**

It allows your AI agent to talk directly to your live database using natural language. You simply tell it what data you need or what change needs making, and the MCP executes the necessary query or mutation behind the scenes.

02 **Can I use the Convex MCP to check if a user record exists?**

Yes. You can ask your agent to run a simple read-only query against any table. It confirms the existence of records and pulls out exactly what you need, saving you manual checks.

03 **Is updating data with Convex MCP safe from errors?**

Absolutely. When you use mutation functions, the MCP handles all the transactions to ensure atomicity—meaning every piece of related data updates correctly together or nothing changes at all.

04 **What if I need to run a complex calculation that involves external APIs?**

You can trigger these side effects using actions. The MCP lets your agent execute defined Convex actions, which handle the heavy lifting and connection to external services for you.

05 **How does connecting this MCP improve my development workflow?**

It eliminates context switching. You can debug data integrity, check state, or run migrations directly from your chat interface alongside writing code, making the entire process faster and more efficient.

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 `"convex": { "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

Convex 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 Convex. 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	Convex MCP
Server ID	`019e387d-3c6f-708c-aadd-f9e056b87fb1`
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/convex.