MCP SERVER | NO CODE | CLOUD HOSTED

LeanCloud MCP

Manage all your data, users, and notifications via conversation.

LeanCloud MCP connects your AI agent directly to a scalable backend service, letting you manage application data and user accounts through natural conversation. Your agent can list records, create new content, update profiles, audit files, or trigger push notifications—all without needing to log into the console.

A+ Quality Score 100/100

backend-as-a-service | data-modeling | user-management | push-notifications | api-development | cloud-storage

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

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.

LeanCloud MCP

10 tools available
Cloud-hosted on Vinkius

You shouldn't have to jump between an IDE, a dashboard, and a database terminal just to run a simple data check. This MCP lets your agent treat LeanCloud like another API endpoint you already use every day. Instead of building complex scripts to handle user data or content changes, you just ask it. Your agent can list specific object types within any data class, pull detailed metadata on individual records, and even manage the full lifecycle of an application's membership base. Need to update a profile? Just tell your agent. Want to send out a mass alert? It handles that too. Everything—from listing user accounts to initiating notifications—is accessible through your AI client via Vinkius.

Core Capabilities

01 — Manage data records

Create, read, update, and delete content objects within specific application classes.

02 — Handle user accounts

Browse or retrieve detailed profiles for every user registered in the system.

03 — Audit files and statistics

List all cloud-stored application files, or check real-time metrics on API usage and storage capacity.

04 — Send alerts

Trigger immediate push notifications to specific users across the platform.

One Click on Vinkius — From Prompt to Execution

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

01 Subscribe to this MCP and supply your unique LeanCloud App ID and App Key.

02 Connect the credentials to any compatible AI client, like Cursor or VS Code.

03 Tell your agent what needs doing; for example, 'List all users who haven't logged in this month.' The agent executes the command using the proper tools.

The bottom line is you talk to your backend infrastructure naturally, and it responds with structured data.

Built For

This MCP is for developers or product managers who spend too much time manually querying APIs or clicking through multiple administrative dashboards. It's built for anyone whose job requires monitoring application health and manipulating live data records without writing a single line of boilerplate code.

Backend Developer

Automates tedious data maintenance tasks, such as creating new objects or checking if existing records need updates before deployment.

Product Operations Manager

Handles user lifecycle events, like auditing membership bases using list_users and coordinating targeted push notifications for feature announcements.

Data Analyst

Gathers system statistics or retrieves historical data by running commands to get application stats or listing objects in a class.

What Changes When You Connect

01 Instead of running separate queries to see what objects exist, simply ask the agent to list_objects. It handles the class structure so you don't have to worry about naming conventions.

02 You can audit your entire user base by calling list_users and then quickly get detailed info on specific people using get_user, all in one conversational flow.

03 Never lose track of what data is available; use list_files to instantly inventory every single asset stored in your cloud storage area.

04 Update records without the headache. Use update_object to modify fields or correct errors across multiple pieces of data points with a single command.

05 Keep users engaged by sending out push_notification alerts right from your agent workspace, triggering immediate action.

Real-World Applications

The weekly user audit

A Product Ops Manager needs to check who signed up last week and if any of their records are missing key data points. They ask the agent to list_users, then run get_user on a few profiles they suspect are incomplete. The agent reports back which profiles need manual cleanup.

Responding to system alerts

The developer notices an old feature needs deactivation and wants to clean up the associated data. They use list_objects to confirm the object type, then call delete_object on specific records, preventing accidental data loss.

Deployment status check

A developer is preparing a new feature release and needs confirmation that all necessary files are uploaded. They ask the agent to list_files, verifying every required document is present before code deployment can proceed.

Coordinating marketing campaigns

The marketing team launches a campaign and needs to notify 50 specific users immediately. They use push_notification, passing the recipient list and message content directly through their agent without touching the notification service console.

Patterns to Avoid

Assuming all data is in one place

✗ AVOID

Trying to find a user's score by only querying the 'Users' table, but realizing the actual scores are stored in a separate 'GameScore' object class.

✓ INSTEAD

First use list_objects to check what classes exist. Then, you can specifically ask the agent to retrieve records from the correct class using get_object or list_objects.

Writing boilerplate API calls

✗ AVOID

Having to write multi-step Python code involving headers, authentication tokens, and multiple `requests.post()` calls just to change a user's email address.

✓ INSTEAD

Tell your agent directly: 'Update the object for User 123 with the new email.' The agent handles the secure API formatting using update_object.

Overlooking application metrics

✗ AVOID

Launching a feature and assuming it will scale because it worked on staging, without checking if the backend can handle the sudden load.

✓ INSTEAD

Before launch, ask the agent to get_app_stats. This gives you immediate data on your current request volume and storage capacity.

The Right Fit

Use this MCP when your problem is about managing, reading, or modifying structured data within a known backend system. If you need to list users (list_users) or modify object details (update_object), this is perfect. Don't use it if your core problem involves connecting two completely separate services—for example, sending an email through Mailchimp and updating a spreadsheet in Google Sheets. For that, you would need a general integration tool like Zapier or similar workflow automation connectors. If you only have unstructured text data (like raw chat logs) that needs analysis without CRUD operations, this MCP is too focused on structured records.

The endless loop of dashboard clicking

Today, managing your backend means hopping between consoles. You check the user list in one tab, then switch to a separate 'Analytics' section to see if those users are active, and finally open the data class editor just to create a new record for them. It's slow, it requires constant context switching, and you always worry about remembering which filter or dropdown menu you left selected.

With this MCP, that process collapses. You talk to your agent like talking to an employee who already knows where everything is. You ask it to pull the user list AND check their last login date, getting a single, immediate answer without touching a dashboard.

LeanCloud MCP: Data Control via Conversation

The painful manual steps disappear. You no longer need to manually execute `list_objects` just to see what data types are available, or run a separate query using get_app_stats just for usage numbers.

Now, your agent acts as the single control plane. It gives you instant access to everything—user profiles via get_user, file audits via list_files, and content updates using create_object—all in one conversation.

LeanCloud: 10 Tools for Backend Management

These ten tools give your AI client full control over every aspect of your LeanCloud application, from listing user profiles to sending push notifications.

#	TOOL	DESCRIPTION
01	`create_object`	Creates and saves a brand-new object record into your defined data classes.
02	`delete_object`	Permanently removes an existing, specified object from the database.
03	`get_app_stats`	Retrieves key performance metrics for your application, like total API calls or storage usage.
04	`get_object`	Pulls the full set of details for a single, specified data object by its ID.
05	`get_user`	Fetches all profile information associated with one specific user account.
06	`list_files`	Lists and reports on every file currently uploaded to your application's cloud storage.
07	`list_objects`	Generates a list of all available object types within a specific data class structure.
08	`list_users`	Retrieves a comprehensive roster of every single user currently registered in the system.
09	`push_notification`	Sends an immediate, targeted notification message to one or more users.
10	`update_object`	Modifies existing content records with new data without deleting and recreating them.

See It in Action

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

U List the last 5 objects in the 'GameScore' class.

I've retrieved the latest records from the GameScore class. You have 5 recent entries, including scores from 'User8821' and 'User9920'. Would you like to view the full metadata for the highest score?

U Show me all users in the application.

I've listed the users for your LeanCloud application. You have 150 total registered members. Should I retrieve the detailed profile for the most recently active user?

U Check the application statistics for today.

I've retrieved the app statistics. Today, your backend has handled 5,000 API requests with a 99.9% success rate. Storage usage is currently at 1.2GB. Would you like a breakdown of the most called endpoints?

Frequently Asked Questions

01 **Can I use LeanCloud MCP to retrieve old user data?**

Yes. You can use list_users to see a roster of all accounts, and then get_user to pull detailed profile information for any specific person in the system.

02 **How do I update content using LeanCloud MCP?**

You use the update_object tool. You simply tell your agent which object needs changing and what the new value should be, preventing data corruption from manual API calls.

03 **What is the difference between list_objects and list_users in LeanCloud MCP?**

list_objects lists all available *data types* (or classes) you have defined. list_users specifically provides a roster of actual user accounts that exist within your application.

04 **Does LeanCloud MCP handle notifications?**

Yes, it does. You can trigger immediate alerts to users by running the push_notification tool directly through your agent interface.

05 **Can I check system performance with LeanCloud MCP?**

Absolutely. Use get_app_stats to retrieve critical metrics like API request counts and overall storage usage, helping you monitor application health at a glance.

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 `"leancloud": { "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

LeanCloud 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 LeanCloud. 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	LeanCloud MCP
Server ID	`019d8451-f6d8-704c-88f9-eadc98aa5fd5`
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/leancloud.

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