MCP SERVER | NO CODE | CLOUD HOSTED

Jellyfin MCP

Manage Your Entire Media Library with Conversational AI

Use Jellyfin to manage your entire media library directly through conversation. This MCP lets you browse movies, TV shows, and music collections, check server health, and track playback progress without logging into an interface. You can get detailed information on any item or user account instantly from your AI agent.

F Quality Score 3.6/100

media-server | streaming | home-theater | self-hosted | video-streaming

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

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.

Jellyfin (Plex Alternative) MCP

12 tools available
Cloud-hosted on Vinkius

Connecting your self-hosted Jellyfin instance to your favorite AI client gives you complete command over your media vault using natural conversation. Instead of navigating complex web dashboards, you talk to the system and it performs actions. You can ask what's available in your root libraries or request a list of specific movies and shows currently stored on the server. Furthermore, if you need to monitor how the system is running—checking things like the version number or OS details—you just ask. This capability lets developers test API interactions easily, while media enthusiasts get total control over their collection from right in your chat window. When you connect this MCP through Vinkius, you gain a single access point to manage everything, making sure your media management stays inside your workflow.

Core Capabilities

01 — Browse and list content

See the root libraries (like Movies or Music) and query all items within them.

02 — Check specific item details

Retrieve deep metadata, including IDs and descriptions, for a single movie or show episode.

03 — Manage playback status

Tell the server when you start watching something, update your current position, or notify it that playback has ended.

04 — Monitor system health

Get core operational details about the Jellyfin setup, including version and OS information.

05 — Manage user accounts

List all active users or verify a user's credentials by name and password.

One Click on Vinkius — From Prompt to Execution

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

01 First, subscribe to this MCP within Vinkius and provide your specific Jellyfin Server URL and API Key.

02 Next, tell your AI agent exactly what you need—for example, 'List all TV shows in the library' or 'What is the server version?'

03 Your agent sends a request through the connection; the MCP retrieves the data from Jellyfin and passes it back to you as readable text.

The bottom line is that this connector turns your media server into an actionable resource for any compatible AI client.

Built For

Anyone managing large, self-hosted digital collections needs this. It's critical for home lab users who need to monitor system integrity without a dashboard, and developers building applications that rely on media metadata.

Media Enthusiast

Uses the MCP to check their entire video library or track playback progress from an AI chat instead of opening Plex/Jellyfin.

DevOps Engineer

Uses it to run status checks, like getting basic server information (`get_system_info`), ensuring the media stack is healthy and operational.

Software Developer

Tests API interactions or needs specific metadata details for a project, using tools like `list_items` to query content in natural language.

What Changes When You Connect

01 Don't manually click through folders. You can check your entire catalog instantly by using `list_items` or asking to browse root libraries, making content discovery effortless.

02 Keep track of viewing progress without hassle. Use functions like `session_playing_progress` and `session_playing` so the server knows exactly where you left off when you step away.

03 System monitoring becomes part of your chat routine. Getting system facts via `get_system_info` lets you check the OS or version number instantly, which is huge for home lab users.

04 Handling user accounts is simple. You can list all connected users with `list_users`, or validate credentials using `authenticate_by_name` without switching apps.

05 Getting deep content context is fast. If a friend asks about a title, you use `get_item` to retrieve detailed metadata and IDs instantly for them.

Real-World Applications

Diagnosing playback issues

A user notices their media player isn't tracking progress. They ask the agent, 'What is the current session status?' The agent responds by using `get_playback_info`, allowing them to diagnose if the server needs a manual update via `session_playing_progress`.

Building an inventory script

A developer needs all available titles for a new app feature. They ask the agent to 'List every show and movie in the library.' The agent uses `list_items`, providing a comprehensive list of content that can then be processed by code.

Checking system readiness

A user just installed Jellyfin on a new box. They ask, 'Tell me about the server.' The agent uses `get_system_info` to report the current version and operating system details in plain text.

Onboarding new users

An admin needs to verify a contractor's access. They ask, 'Can I check if John Doe has an account?' The agent runs `authenticate_by_name`, confirming the user's status and providing their details.

Patterns to Avoid

Trying to get data without context

✗ AVOID

Asking, 'Show me the episode details.' The agent can't answer because it doesn't know *which* item you mean. It needs more information than just a vague request.

✓ INSTEAD

Always narrow your query first. Use `list_virtual_folders` to identify the library (e.g., TV Shows), then use `list_items` within that folder, and finally ask for details using `get_item` with the specific ID.

Ignoring system status checks

✗ AVOID

Assuming everything works because the media loads fine on your screen. The underlying API might be stale or running an old version.

✓ INSTEAD

Run a health check first. Use `get_system_info` to verify the server's current version and OS details before attempting any major operations.

Attempting playback without setup

✗ AVOID

Just asking, 'Play this movie.' The system fails because it needs explicit permission and data on what to play.

✓ INSTEAD

Before playing, you must use `get_playback_info` to get the required start data. Then, tell the agent to initiate playback using `session_playing`.

The Right Fit

Use this MCP if your workflow requires continuous interaction with a self-hosted media server's internal state—like monitoring playback or verifying user access. It is perfect for developers building tools that need read/write access to content metadata and system health metrics, or for advanced users who want full control without GUI clicks. Don't use this if you only need basic file storage viewing; those are better handled by simple cloud file connectors. If your goal is purely managing user credentials across dozens of different services, look at a dedicated Identity Provider connector instead.

Managing media libraries usually means switching between too many apps.

Today, if you want to know what's in your library or update the status on a show you just watched, you have to open the Jellyfin web interface. You click into the 'TV Shows' tab, scroll down, find the title, and manually hit the progress button—all while switching contexts and dealing with browser lag.

With this MCP, that entire process disappears. You simply tell your AI agent what you want to do, whether it's querying all available shows or updating `session_playing_progress`. The data flows directly from the server into your chat window. It’s instant control.

Get full system visibility with get_system_info

Before, checking if the server was running the latest OS or version required SSHing into the machine and running manual commands. You’d have to interpret logs and check multiple dashboard widgets just to feel safe.

Now, you ask your agent, 'What is the system status?' It uses `get_system_info` and gives you a clean, direct answer about the version and OS right where you are working. That's immediate peace of mind.

Jellyfin (Plex Alternative) MCP with 11 Tools

These tools let you perform specific actions on your Jellyfin server—from checking system versions to managing playback status—all through natural language commands.

#	TOOL	DESCRIPTION
01	`authenticate_by_name`	Verifies a user's identity by submitting their name and password.
02	`get_current_user`	Fetches information about the account that is currently logged into the server.
03	`get_item`	Retrieves full details, including descriptions and IDs, for a specific media item.
04	`get_playback_info`	Gathers the necessary information to start playing any piece of content on the server.
05	`list_items`	Searches and reports all available media items within the entire library.
06	`list_users`	List all users (Admin only)
07	`list_virtual_folders`	Shows a list of all top-level libraries configured on the server (e.g., Movies, Music).
08	`session_playing_progress`	Updates the system with how far along you are in watching a piece of media.
09	`session_playing_stopped`	Sends a signal to the server that your current playback session has ended.
10	`session_playing`	Notifies the server when you begin playing media content.
11	`get_system_configuration`	Retrieves the global settings and configuration details for the entire Jellyfin instance.
12	`get_system_info`	Provides essential basic server facts, such as its version number and operating system.

See It in Action

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

U List all the root libraries on my Jellyfin server.

I've retrieved your root libraries. You have 'Movies' (ID: 1a2b), 'TV Shows' (ID: 3c4d), and 'Music' (ID: 5e6f) configured.

U Show me the movies available in my library.

Fetching movies... I found titles including 'Big Buck Bunny' and 'Elephants Dream'. Would you like details for any of these?

U What is the current status and version of my Jellyfin server?

Your Jellyfin server is running version 10.9.11 on Linux. The system is healthy and all services are operational.

Frequently Asked Questions

01 **How do I list all my root libraries using Jellyfin MCP?**

You simply ask your agent to list the virtual folders. The tool `list_virtual_folders` gives you a clean breakdown of every top-level library—like 'Movies' or 'Music'—configured on your server.

02 **Can I check if a user exists in Jellyfin using the MCP?**

Yes, you can. Use `list_users` to get a full roster of all accounts. If you need to verify credentials for a specific person, use `authenticate_by_name`.

03 **What is the difference between list_items and get_item in Jellyfin MCP?**

`list_items` gives you a broad query of all content available within a library. `get_item`, however, requires knowing the specific item ID to retrieve deep, detailed metadata about just that single piece of media.

04 **How do I tell Jellyfin that I started watching something?**

You initiate playback using the `session_playing` function. This notifies the server immediately and ensures your activity is logged correctly for tracking purposes.

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 `"jellyfin-plex-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

Jellyfin (Plex Alternative) 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 Jellyfin (Plex Alternative). 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	Jellyfin (Plex Alternative) MCP
Server ID	`019e38b0-1621-72fc-aa03-f67dddefdb15`
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/jellyfin-plex-alternative.