MCP SERVER | NO CODE | CLOUD HOSTED

OpenAI MCP

Manage your entire AI resource lifecycle conversationally.

OpenAI MCP manages your entire AI resource stack conversationally. List and track all models, monitor fine-tuning jobs, manage Assistants, and run cost-effective batch processing—all without leaving your agent's chat window.

A+ Quality Score 100/100

llm-management | fine-tuning | model-discovery | ai-assistants | file-management | api-orchestration

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

OpenAI MCP

13 tools available
Cloud-hosted on Vinkius

Connecting your OpenAI account to your agent means you get full oversight of your model infrastructure through natural conversation. Instead of jumping between dashboards just to check job statuses or audit resources, you talk to your AI client. You can discover every available model, from GPT-4o to DALL-E 3, and pull up its ownership details. Need to stop a bad training run? The MCP lets you monitor fine-tuning jobs and cancel them instantly. It also handles file management for all your uploaded data, making it simple to delete old or unused assets. For bulk work, you can create batch processing jobs to handle hundreds of API calls cost-effectively. You'll find managing model lifecycles much easier when connected through Vinkius, letting your agent act as a dedicated ML ops assistant.

Core Capabilities

01 — Audit Model Inventory

Discover all models available to your account and check their metadata, like ownership or creation date.

02 — Manage Fine-Tuning Pipelines

Monitor the status of training jobs, track progress, and cancel long-running fine-tuning processes.

03 — Control Assistant Configurations

List and inspect all configured Assistants, checking their instructions, models, and tools before deployment.

04 — Process Bulk API Requests

Set up and track batch processing jobs to run large volumes of requests cost-effectively.

05 — Handle File Assets

List, manage, and delete uploaded files used for fine-tuning or Assistants.

One Click on Vinkius — From Prompt to Execution

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

01 Subscribe to this MCP on Vinkius and provide your OpenAI API Key.

02 Your AI client connects the key and pulls a real-time view of all your existing resources (models, files, etc.).

03 You ask your agent to perform an action—like listing assistants or canceling a batch job—and it executes the command directly.

The bottom line is you manage complex AI operations using simple conversational prompts instead of navigating multiple web dashboards.

Built For

ML Engineers, DevOps teams, and Product Managers who are tired of jumping between the OpenAI dashboard, CLI, and IDE to handle model lifecycles. You need one central place to audit resources.

Machine Learning Engineer

They monitor fine-tuning jobs using list_fine_tunes and track batch processing with list_batches, all without leaving their IDE.

DevOps Specialist

They audit uploaded files via list_files to clean up unused data or review the status of batch processing jobs using get_batch.

Product Manager

They inspect configured Assistants by calling list_assistants to ensure the right models and tools are attached before a product launch.

What Changes When You Connect

01 Stop jumping between dashboards. You manage model versions, fine-tuning jobs, and Assistant configurations entirely through chat with your agent.

02 Save time on bulk processing. Instead of manually submitting requests, you use create_batch to run large API workloads cost-effectively.

03 Maintain a clean workspace. Use list_files and delete_file to audit and remove old or unnecessary training data and assets.

04 Quickly diagnose problems. If an Assistant isn't working right, you can get_assistant to inspect its model, tools, and instructions immediately.

05 Track every job status in one place. From monitoring a running fine-tuning process with get_fine_tune to checking list_models for availability, it’s all conversational.

Real-World Applications

The Model Discovery Check

A product manager needs to know if the newest embedding model is available. Instead of reading through documentation or hitting an API endpoint manually, they ask their agent: 'What models do I have?' The agent runs list_models and reports back a clean list with capabilities.

Auditing Assistant Dependencies

A developer needs to verify which models are attached to a key customer-facing bot. They ask their agent to list_assistants, getting an instant report detailing every linked model and tool for audit purposes.

Stopping Wasteful Training

An ML engineer realizes they uploaded the wrong training data. Rather than waiting hours for a job to fail, they ask their agent to check the jobs using list_fine_tunes and then immediately execute cancel_fine_tune on the incorrect run ID.

Patterns to Avoid

Guessing file IDs

✗ AVOID

A user tries to cancel a running job but uses a random or outdated file ID, resulting in an API error and wasted time.

✓ INSTEAD

First, always run list_files to get the correct file ID, then use cancel_fine_tune with the specific fine-tune job ID. This confirms your asset is active before you try to modify it.

Running a massive batch manually

✗ AVOID

A team needs 50,000 completions and decides to write a complex script that runs locally, risking rate limits or incomplete data.

✓ INSTEAD

Use create_batch. This MCP handles the heavy lifting of bulk processing securely, letting you track progress using list_batches and get_batch.

Ignoring resource usage

✗ AVOID

A team repeatedly creates new Assistants without documenting which model or files they use, leading to unexpected costs.

✓ INSTEAD

Use list_assistants first. This provides an immediate audit of all active configurations so you know exactly what resources are consuming capacity.

The Right Fit

You should use this MCP if your workflow requires continuous management of multiple, interconnected OpenAI assets: fine-tuning pipelines, Assistants, file storage, and bulk API calls. If you need to audit *how* these things connect or monitor their status over time, this is essential. Don't use it if you just need to run a single, isolated model call—your agent can handle that without the full MCP. Also, don’t use it if your primary goal is simple data storage; for file management alone, list_files works, but for the complete lifecycle control, this MCP is required.

The pain of checking model statuses across different dashboards

Today, managing your AI resources means hopping between three places: the OpenAI dashboard to check fine-tune status; a separate console for batch jobs; and then maybe an IDE just to list available models. You're constantly copying IDs, refreshing tabs, and cross-referencing statuses—it’s slow, error-prone detective work.

With this MCP, your agent handles it all conversationally. You ask about the status of a job or model, and you get a clean, consolidated answer instantly. It cuts out the dashboard hopping entirely.

OpenAI MCP: Full Model Lifecycle Control

You eliminate the need for custom scripts just to pull metadata or cancel jobs. You can list all available models using list_models, confirm an Assistant's setup with get_assistant, and then track that Assistant’s dependency files via list_files.

The difference is control. Instead of managing resources reactively by clicking through web forms, you manage the whole lifecycle proactively with a few simple commands.

OpenAI MCP with 13 Tools

These tools let you perform every required administrative task for your OpenAI account, from discovering model types to canceling complex training jobs.

#	TOOL	DESCRIPTION
01	`cancel_batch`	Stops an active batch processing job using its unique ID.
02	`cancel_fine_tune`	Halts a running fine-tuning job when you need to restart or change the source material.
03	`create_batch`	Initiates a new, large-scale batch processing job using specific input files and endpoints.
04	`delete_file`	Permanently removes an uploaded file asset from your account. Be careful with this one.
05	`get_assistant`	Retrieves detailed information about a specific Assistant by its ID.
06	`get_batch`	Pulls the current status and details for a specified batch job.
07	`get_fine_tune`	Checks the detailed status, base model, and progress of a fine-tuning job ID.
08	`get_model`	Verifies if a specific OpenAI model exists and retrieves its metadata (e.g., owner or creation date).
09	`list_assistants`	Generates an audit list of all configured Assistants, detailing their models and tools.
10	`list_batches`	Provides a status overview for every batch processing job in your account.
11	`list_files`	Shows all uploaded files, noting if they are intended for fine-tuning, Assistants, or batches.
12	`list_fine_tunes`	Lists the status of all your model training jobs to monitor the overall pipeline health.
13	`list_models`	Displays every available OpenAI model ID and its capability flags for discovery.

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 available GPT models.

I found 12 GPT models in your account: gpt-4o (owned by openai, created 2024-05), gpt-4o-mini (lightweight, cost-effective), gpt-4-turbo, gpt-4, gpt-3.5-turbo, plus several fine-tuned variants. Would you like details on any specific model?

U Check the status of my latest fine-tuning job.

Your latest fine-tune job ftjob-abc123 is currently 'running' at 67% completion. It's training on 1,240 examples using gpt-3.5-turbo as the base model. Estimated finish is in approximately 2 hours. The training file is file-xyz789.

U List all my uploaded files and their purposes.

You have 8 files uploaded: 3 for fine-tuning (training JSONL files), 2 for Assistants (knowledge base PDFs), 2 for batch processing (request JSONL files) and 1 vector store file. Total storage used: 45MB.

Frequently Asked Questions

01 **How do I check if a specific model is available using OpenAI MCP?**

You use get_model to verify existence and pull metadata. This tool confirms if the model ID you need—like 'gpt-4o'—is active in your account before you write any code that depends on it.

02 **What is the difference between list_files and list_models?**

list_files tracks data assets (PDFs, JSONL files) used for training or Assistants. list_models tracks the actual AI model types themselves (e.g., Whisper-1). You need both to run a full pipeline.

03 **Can I use OpenAI MCP to stop an expensive fine-tune job?**

Yes, you can. First, check status with list_fine_tunes, then execute cancel_fine_tune using the specific job ID. This stops unnecessary spending immediately.

04 **How do I manage my Assistants through the OpenAI MCP?**

You start by listing all assistants with list_assistants to get an overview. Then, you use get_assistant if you need deep details on one specific bot's configuration.

05 **Is batch processing done via OpenAI MCP safe for large volumes?**

Yes, create_batch is designed for this. It handles the workload of thousands of requests in a cost-effective way, and you track its status using list_batches.

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 `"openai-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

OpenAI 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 OpenAI. 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	OpenAI MCP
Server ID	`019d8464-d3db-71f1-9691-fe5ece927cfb`
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/openai-alternative.