MCP SERVER | NO CODE | CLOUD HOSTED

Discourse MCP for AI Agents

Manage community forum structure & member activity

Discourse MCP equips your AI agent with complete control over community forums. It lets you manage topics, research user profiles, track group memberships, and monitor entire category structures through natural conversation. Stop clicking through dashboards; start asking questions about your community's health.

A+ Quality Score 100/100

community-forum | discussion-boards | user-engagement | content-moderation | open-source | api-integration

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

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.

Discourse MCP

10 tools available
Cloud-hosted on Vinkius

This MCP connects your AI client directly to the Discourse platform, giving your agent deep visibility into your online community. You can manage everything from tracking new discussion threads to researching specific user history and group alignment—all without leaving your chat window.

Instead of logging in and clicking through multiple sections just to get a snapshot, you simply ask your agent what's happening. Need to know who belongs to the 'Premium Users' group? Or maybe you want to find out which topics are generating buzz right now? Your agent pulls that data instantly. When you connect it via Vinkius, your AI client can orchestrate complex moderation tasks and content strategy analysis through simple conversation. It turns a massive forum into an actionable data source.

Core Capabilities

01 — Retrieve general platform settings

Fetches metadata and configuration data about the entire Discourse site instance.

02 — Get full topic content details

Retrieves every post and the overall body content for a specific discussion thread.

03 — Search all community content by keyword

Searches across users, posts, and topics for any content matching your provided phrase or term.

One Click on Vinkius — From Prompt to Execution

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

01 First, connect your AI client to this MCP on Vinkius and authorize access using your Discourse API Key, Username, and instance URL.

02 Next, you prompt your agent with a request, like 'List all users in the Moderators group,' or 'Show me the details for topic X.'

03 Your agent executes the necessary tool calls to pull the required data (user profiles, topic content, or group lists) and presents it back to you in plain text.

The bottom line is, your AI client handles all the API interaction; you just talk to it like normal.

Built For

This MCP helps community managers and moderators who are tired of switching between their internal dashboard, spreadsheet software, and communication tools. If you spend time manually checking user history or tracking content trends, this is for you.

Community Manager

Uses the MCP to monitor trending discussions and list active members across different groups to plan community events.

Moderator

Checks detailed user profiles and trust levels during moderation, researching a member's history before taking action.

Content Strategist

Analyzes the full category tree structure and lists the latest topics to determine where the community needs more content focus.

What Changes When You Connect

01 Quickly assess the health of your discussions. Use `list_trending_discussions` to pinpoint what topics are gaining momentum, allowing you to focus moderation efforts where they're needed most.

02 Never forget who has access to what. With `list_community_groups`, you can map out all user roles and then use `list_group_members` to see exactly who belongs in each one.

03 Deep dive into user behavior. You can check a single person's history by calling `get_user_profile`, giving moderators the context they need before escalating an issue.

04 Keep track of content flow. If you use `list_latest_topics` and then follow up with `get_topic_details`, your agent gives you both the summary list and the full post count immediately.

05 Understand platform limitations. Use `get_site_configuration` to know if there are any API constraints or metadata issues that might affect future automation efforts.

Real-World Applications

Investigating a new user's behavior

A moderator suspects 'john_doe' is posting off-topic. They ask their agent to run `get_user_profile` and also check if the user belongs to sensitive groups using `list_group_members`. The agent returns their full history, trust level, and group affiliations in one response.

Identifying evergreen content needs

A Content Strategist wants to know what discussions are worth turning into blog posts. They ask the agent to run `list_trending_discussions` which highlights high-engagement threads, providing immediate material ideas.

Mapping out user permissions

The Admin needs a full picture of who can do what. The agent runs `list_community_groups` to get all groups, and then iterates through them using `list_group_members` to create a master roster.

Responding to a sudden content spike

The team notices a flurry of posts on 'Q2 Feedback.' They ask the agent to run `get_topic_details` for that specific topic, instantly providing all 50 new comments and allowing them to respond immediately.

Patterns to Avoid

Treating user activity as a single list

✗ AVOID

Asking the agent simply to 'show me active users' without specifying context. This gives an overwhelming, unorganized list that doesn't help with moderation priorities.

✓ INSTEAD

Instead of that, ask the agent to combine `list_active_members` and then filter those results by running a targeted search using `search_community_content` for keywords like 'billing' or 'support'.

Assuming all topics are equally important

✗ AVOID

Running general searches when the goal is to find high-value, current discussions. This wastes time sifting through old threads that aren't relevant right now.

✓ INSTEAD

Always start by calling `list_trending_discussions` or `list_latest_topics`. That focuses your attention immediately on what people are talking about *today*.

Ignoring group hierarchy

✗ AVOID

Trying to manage user roles only through individual profiles, which fails when users belong to multiple overlapping groups.

✓ INSTEAD

First, use `list_community_groups` to see the structure. Then, for a specific role, run `list_group_members` so you get an accurate roster of all members associated with that group.

The Right Fit

Use this MCP if your primary job involves monitoring, moderating, or analyzing user-generated content across complex community forums. Specifically, if you need to cross-reference a user's profile details (`get_user_profile`) against their membership in multiple groups (`list_group_members`), this is the right tool. It handles deep

relational data queries that simple messaging tools can't touch.

Don't use it if your only goal is simple notification—like getting an email when a topic hits 10 replies. For simple alerts, you probably just need a dedicated webhook or subscription service. If you just want to search content without needing the full context of who posted it, basic keyword search tools might suffice; but for deep moderation and structural analysis, this MCP is necessary.

Discourse MCP: Managing Community Forum Structure and Topics

Today, managing a large community forum means jumping between pages. You check the 'Latest' tab for new posts, then click into a thread to see who replied, then maybe you have to open another panel just to figure out what groups that user belongs to. It’s constant clicking and copy-pasting of usernames.

With this MCP, your agent handles all those manual steps. You ask it to find the top 10 most active discussions and simultaneously list which community groups are involved in those threads. What you get back is a single, actionable summary that saves hours of navigation.

Discourse MCP: Tracking User Profiles and Group Dynamics

Tracking user credibility is painful because membership status often lives in one place, while activity logs are stored elsewhere. You have to manually check the profile page for trust levels and then cross-reference that with the group roster to see if they're supposed to be there.

Now, your agent pulls all that data together. Tell it which user you want to investigate, and it returns their complete profile details alongside every single group they belong to—all in one conversation turn.

Discourse: 10 Tools for Forum Structure & Member Data

Use these tools to retrieve site settings, list user profiles, map out forum categories, or find content matching any keyword across the platform.

#	TOOL	DESCRIPTION
01	`get_site_configuration`	Retrieves general site settings and metadata for your Discourse instance, giving you an overview of platform limits and status.
02	`get_topic_details`	Pulls all posts and the complete content history for a single discussion topic.
03	`get_user_profile`	Fetches detailed profile data for one specific user based on their username.
04	`list_active_members`	Provides a list of all users currently active in the community (requires admin access).
05	`list_forum_categories`	Lists every public category configured on the platform, including structural relationships.
06	`list_group_members`	Generates a roster of all users belonging to a specified community group.
07	`list_community_groups`	Lists every user group configured within the entire community structure.
08	`list_latest_topics`	Returns a list of all discussion topics that were created most recently across the platform.
09	`list_trending_discussions`	Identifies and lists topics that have seen the highest amount of engagement over a recent period.
10	`search_community_content`	Searches across all topics, posts, and user names for content matching a specific keyword or phrase.

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 most talked-about topics this week?

Top 3 Trending Discussions

1. **Major System Update:** (25 new posts today) - Focus is on API changes.
2. **Community Guidelines RFC:** (12 new posts) - High disagreement, requires moderator review.
3. **Q2 Feedback Thread:** (7 new posts) - Positive sentiment, needs action items.

I can pull the full post list for any of these if you want to see who started it.

U Who are the premium members and what groups are they in?

Premium Group Members Roster

- john_doe (Trust Level 3 - Regular): Member of 'Moderators' and 'Beta Testers'.
- jane_smith (Trust Level 4 - Power User): Member of 'Beta Testers' only.
- admin_bot (Trust Level 5 - Admin): Member of all groups listed.

U Find me any posts about payment issues from new users.

I found three relevant discussions:

- **Topic: Billing Question:** Contains 4 posts. User 'newbie_user' mentioned a payment failure.
- **Topic: Subscription Help:** Contains 12 posts. Multiple users discussed payments, but no specific failures were noted in the summary.

The user profile for 'billing_help' shows they joined yesterday and are currently listed as an active member.

Frequently Asked Questions

01 **How does the Discourse MCP help me manage community discussions?**

The MCP lets your AI agent track, list, and retrieve details for any discussion topic or user post. Instead of checking the forum manually, you ask your agent to summarize what's trending or pull the full history on a specific thread.

02 **Can I use the Discourse MCP to check who is in which group?**

Yes. Your AI client can list all community groups and then, for any given group name, it generates a complete roster of every user belonging to that specific membership.

03 **What kind of information about users can the Discourse MCP provide?**

It provides detailed profile data, including the user's trust level, join date, and how many topics or posts they have created. This is vital context for moderation decisions.

04 **Is this helpful for content strategy planning in Discourse MCP?**

Absolutely. You can ask your agent to list all public categories to map out the forum structure, and then run a search across all content using keywords like 'roadmap' or 'feedback' to see what needs attention.

05 **Does the Discourse MCP let me know which users are currently active?**

Yes. You can ask your agent to pull a list of currently active community members, giving you an immediate overview of who is online or highly engaged right now.

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 `"discourse": { "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

Discourse 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 Discourse. 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	Discourse MCP
Server ID	`019d7586-c02b-7202-8b12-19476f3dd6ed`
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/discourse.