MCP SERVER | NO CODE | CLOUD HOSTED

Quip MCP

Search, read, and update documents without leaving your IDE.

Quip MCP connects Quip collaboration data—documents, threads, and spreadsheets—directly to any AI client. Use it to search across all your accessible files instantly, pull historical messages attached to documents, retrieve folder structures, or programmatically update content right within the thread.

A+ Quality Score 98.33/100

document-collaboration | spreadsheets | team-messaging | content-management | knowledge-base

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

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.

Quip MCP

12 tools available
Cloud-hosted on Vinkius

This connector brings your real-time work from Quip into your natural workflow. Instead of switching between your IDE and Quip's browser, your AI agent handles the heavy lifting. You can perform full-text searches across every document in your accessible space, pulling out exactly what you need, whether it's a specific technical specification or an old product requirement. Need context on a document? Your agent can check the attached message threads for feedback without opening the file. The system also lets you pull down the folder hierarchy and even make programmatic changes; you can update content by passing HTML payloads back into specific Quip threads. Connecting through Vinkius allows your AI client to access this entire library of functionality, making it a single source of truth right where you're working.

Core Capabilities

01 — Search all documents

Perform full-text searches across every document in your Quip workspace.

02 — Retrieve thread details

Fetch the complete content of a single Quip document or thread using its unique ID.

03 — List recent work

See a list of documents you or your team have recently viewed or edited.

04 — View conversation history

Check all messages and comments attached to any given document thread.

05 — Edit content

Programmatically update existing Quip documents by sending new HTML-formatted text payloads.

One Click on Vinkius — From Prompt to Execution

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

01 Subscribe to the MCP and authorize it using your Quip Personal Access Token.

02 Instruct your AI agent to perform an action, like searching for a document or summarizing comments attached to a thread.

03 Your AI client executes the necessary commands and returns the requested data—whether it's raw text, file lists, or structured metadata.

The bottom line is that your AI agent treats Quip like another API endpoint, allowing you to interact with its content using natural conversation rather than manual clicks.

Built For

Product Managers who need immediate context on documentation changes, Developers stuck jumping between ticket trackers and specs, or Account Executives drafting client reports based on shared team files.

Product Manager

Summarizing recent feedback attached to product strategy documents or fetching technical specifications from Quip links before writing PRDs.

Software Developer

Pulling current requirements or architecture specs directly from a shared document in Quip, then using that context to write code or update tickets.

Account Executive

Drafting client updates and pushing the finalized text into a shared Quip document for immediate team review.

What Changes When You Connect

01 Context is always available. If you need to know what was discussed about the 'Q3 Roadmap,' simply ask your agent to search using `search_threads`, pulling in relevant discussions instantly.

02 Stop wasting time on outdated info. By running `get_recent_threads`, your agent immediately gives you a list of documents you or your team worked on, so you never lose context.

03 You can update content directly from the chat. Using `edit_document` means you don't have to manually open Quip and copy/paste; you just send the final HTML payload.

04 Get granular details about a document. Need everything in one place? Use `get_thread` or `list_blobs` to pull out all attached messages, files, and core text together.

05 Manage your workspace structure. Instead of navigating manually, use `get_folders` to batch-fetch multiple folder IDs, giving you a map of the entire Quip library.

Real-World Applications

Finding buried requirements

A dev needs to know if a feature was discussed last month. They ask their agent: 'Search Quip for documents containing 'API rate limits'.' The agent runs `search_threads`, finds the document, and pulls the relevant text right into your chat.

Client update drafting

An AE has finalized talking points. They use their agent to run `edit_document` on a shared Quip file, pushing the polished HTML version directly into the document for immediate client viewing.

Catching up on project status

A PM returns from vacation and asks: 'What were the last 10 things I looked at in Quip?' The agent uses `get_recent_threads` to give an instant list of modified documents, skipping the manual folder browse.

Deep dive on a topic

A team member needs all comments around a key decision. They tell their agent to check the thread and run `get_messages` for that ID, gathering every piece of feedback in one place.

Patterns to Avoid

Trying to find content by folder name

✗ AVOID

Asking your AI agent generally, 'Find the marketing documents.' This fails because it needs specific IDs or keywords.

✓ INSTEAD

First, tell your agent to run `get_folders` if you have a list of container IDs. Better yet, use `search_threads` with precise terms like 'Q3 Marketing Strategy' for guaranteed results.

Overloading the chat with raw data

✗ AVOID

Just asking the agent to 'give me all Quip stuff.' This returns thousands of IDs and unreadable metadata.

✓ INSTEAD

Be specific. If you need content, ask it to `get_thread` for a single ID. If you need context on multiple items, provide a comma-separated list to `get_threads`.

Assuming the document is always open

✗ AVOID

Thinking you can edit content without knowing where it lives. You might try to paste HTML into an empty chat window.

✓ INSTEAD

You must first use `search_threads` or `get_recent_threads` to locate a specific document ID, and then pass that ID to the `edit_document` tool.

The Right Fit

Use this MCP if your job requires moving information *between* Quip and another application—like pulling specs from a shared doc into a Jira ticket or updating content in a document directly after generating it elsewhere. You need the full, bidirectional context of the collaboration platform.

Don't use this if you only need to read general public data or search across multiple independent platforms (e.g., Quip and Google Docs). For that, you'd need an integration that supports both services simultaneously. If all your content lives in one place, but you just want basic retrieval without writing back, a simple document reading tool might suffice. But if the workflow involves generating drafts *and* placing them into the original source, this Quip MCP is what you need.

The headache of context switching

Today, getting a complete picture requires jumping through hoops. You check your IDE for code snippets, open Quip to review the product spec document, then switch back out to Slack just to see the last comment on that doc. It's copy-paste hell. You spend more time managing tabs than doing actual work.

With this MCP, you keep everything in one place. Your agent handles the switching. You tell it what you need—a spec summary or a list of related files—and the data streams directly into your chat interface. It's seamless.

Quip elevates document collaboration

You no longer have to manually navigate folder by folder, or open a thread just to check if someone left feedback in the comments. Your agent handles reading those attached messages and fetching related files automatically.

The result is immediate action. Instead of documenting findings somewhere else and then remembering to go back into Quip, you generate the content with your agent and push it straight into the correct document using `edit_document`. It's done.

Quip MCP: 12 Tools for Document Ops

Use these tools to perform specific actions across your entire Quip workspace, from searching content to managing user profiles.

#	TOOL	DESCRIPTION
01	`get_current_user`	Retrieves basic profile information for the user who is currently logged in.
02	`get_thread`	Fetches all content and metadata for a specific Quip document or thread ID.
03	`get_threads`	Retrieves the full content of multiple Quip threads when you provide a list of IDs.
04	`get_recent_threads`	Lists documents that have been recently viewed or modified by any user.
05	`search_threads`	Searches across all accessible Quip documents to find content matching your query.
06	`get_folder`	Retrieves the details and metadata for a single Quip folder by its ID.
07	`get_folders`	Fetches information about multiple Quip folders when you provide a list of IDs.
08	`get_messages`	Lists all chat messages or comments that have been attached to a specific thread.
09	`get_user`	Retrieves detailed profile information for any specified Quip user.
10	`get_contacts`	Lists all collaborators and users associated with the currently logged-in account.
11	`list_blobs`	Retrieves a list of embedded files, images, or media blocks inside a document thread.
12	`edit_document`	Updates the content within an existing Quip document using HTML format.

See It in Action

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

U Search Quip for documents containing 'Q3 Roadmap'.

I ran `search_threads` for 'Q3 Roadmap' and found 2 relevant documents:
1. 'Engineering Q3 Roadmap' (ID: ABC123DEF)
2. 'Design Team Syncs - Q3 Roadmap' (ID: XYZ987JKL)
Would you like me to retrieve the contents of the Engineering thread?

U What documents did I work on recently in Quip?

Using `get_recent_threads`, here are your latest documents:
- 'Sprint 44 Retrospective' - Modified today.
- 'Marketing Copy Ideas' - Modified yesterday.
- 'Q3 Budget Tables' - Modified on Tuesday.

U Add a new heading called 'Conclusion' and a paragraph 'All tests passed' to document ABC123DEF.

I successfully edited the thread 'ABC123DEF' using the HTML payload: `<h2>Conclusion</h2><p>All tests passed</p>`. The content has been added to the document.

Frequently Asked Questions

01 **How do I securely obtain my Quip Access Token?**

Sign in to your Quip account. Navigate to your API token generator page (usually at `platform.quip.com` or via Quip's Developer Settings). Generate a **Personal Access Token**, copy the string, and securely save it in the configuration fields below.

02 **Can I provide a Quip document URL to my AI, or just the ID?**

You can provide the standard URL. The agent is capable of recognizing the URL structure, extracting the alphanumeric `thread_id`, and performing the operation accurately.

03 **Can it read embedded images or files inside the document?**

The agent detects their existence by reading the raw structure and uses `list_blobs` to catalog attachments. However, downloading the actual files depends on your client's supported capabilities.

04 **Is the edit operation destructive?**

The `edit_document` function uses Quip's native append logic to add new HTML blocks at the end of the document, mitigating complete overwrites. Additionally, Quip maintains a full version history allowing rollbacks if necessary.

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 `"quip": { "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

Quip 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 Quip. 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	Quip MCP
Server ID	`019d75fb-c7b0-7098-991a-9d67e31aeed8`
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/quip.

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