MCP SERVER | NO CODE | CLOUD HOSTED

JSON Path Query Engine MCP

Pull Specific Data From Massive Payloads

JSON Path Query Engine is an MCP that lets you surgically extract specific data points from massive, complex JSON payloads. Instead of sending a huge API response to your agent and risking context window overload, this tool uses precise JSONPath expressions to pull out only the fields you need—like all email addresses or every order total—saving tokens and keeping your conversation focused.

A+ Quality Score 100/100

json-query | data-extraction | api-parsing | token-optimization | data-filtering

MAINTAINED BY VINKIUS CLOUD FOR AI AGENTS · JUNE 2026
vinkius.com/mcp/json-path-query-engine

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.

JSON Path Query Engine MCP

1 tools available
Cloud-hosted on Vinkius

Working with APIs often means receiving enormous data dumps. You might get a 5,000-token payload containing dozens of records when really, you just needed three specific values: the user ID, the item name, and the final price. Sending that whole blob to your AI client is wasteful and noisy. This MCP lets you bypass that problem entirely. It runs specialized pathing logic to look through deeply nested data structures using standard JSONPath syntax. You give it the raw JSON and a precise expression (like `$.orders[0].total_price`), and it spits out only the matching values. This capability means your agent doesn't have to read gigabytes of unnecessary text; it gets clean, filtered results immediately. It's essential for keeping your prompts efficient, whether you connect Vinkius through Claude or Cursor.

Core Capabilities

01 — Extract specific fields from JSON

You pass the MCP a raw JSON string and a path expression to pull out only matching data points.

02 — Query nested arrays

The tool can search through complex, deeply nested structures within the payload using advanced paths like `$.users[*].email`.

03 — Reduce context window load

You eliminate sending large amounts of unnecessary data to your agent, saving tokens and improving response reliability.

One Click on Vinkius — From Prompt to Execution

Available at **vinkius.com/mcp/json-path-query-engine** — connect your AI agent in three steps.

01 First, you provide the MCP with two things: the full raw JSON string that came from an API, and a specific JSONPath expression defining the data you want.

02 The engine processes this input, running the pathing logic against the massive payload to locate every piece of data that matches your defined criteria.

03 You get back only a clean list of values. The raw JSON remains hidden; your agent only sees the filtered results.

The bottom line is you stop sending entire API responses and start sending targeted, pre-filtered datasets.

Built For

This MCP is for data engineers, backend developers, and advanced analysts who deal with complex APIs daily. If you're tired of your AI client getting confused or losing context because the input payload was too large to process cleanly, this tool saves hours of debugging.

Data Engineer

You use it when integrating multiple API sources, needing to extract only specific metrics (e.g., every user ID and associated timestamp) without having your agent process the whole transaction log.

Backend Developer

You leverage it during prototyping to test how structured data is extracted from varied payloads, ensuring your application logic only receives clean, necessary values for follow-up actions.

API Analyst

You use it to validate API documentation or troubleshoot unexpected responses, pulling out specific elements like all error codes or all associated IDs in one go.

What Changes When You Connect

01 You save tokens by not dumping entire API responses. Instead of sending a 5,000-token payload to your agent, you use the query_json tool to pull just the three fields you need, keeping costs down and context clear.

02 The ability to search deeply nested data is huge. You can target specific records, like finding all email addresses (`$.users[*].email`) across a whole user list without manually looping through the JSON structure.

03 Your agent processes clean results, not noise. By filtering the payload first, your AI client doesn't waste time reading irrelevant metadata or boilerplate text; it acts on facts.

04 It vastly improves reliability when dealing with complex data sets. Instead of relying on the LLM to 'figure out' where a value is buried, you tell it exactly where using JSONPath syntax.

05 You get structured output for scripting. This MCP ensures that whether you are building an agent pipeline or running a local test, the extracted values are consistent and ready for immediate use.

Real-World Applications

Needing all author emails from a bookstore API payload

A developer receives a massive JSON file containing details for hundreds of books. Instead of asking the agent to 'find all emails,' they use query_json with `$.bookstore[*].authors[*].email` to get one clean list of every single email address, ignoring book titles and prices entirely.

Extracting multiple order totals for a billing process

An analyst needs to calculate the total value from an array of line items. They use query_json on the raw payload with `$.orders[*].total_price` to pull every single price into a list, making it easy to sum up and verify against a database record.

Finding all user IDs associated with failed transactions

The operations team gets an error log containing thousands of records. They use the MCP to specifically query for `$.transactions[?(@.status=='FAILED')].user_id`, instantly generating a precise, actionable list of only the affected users.

Parsing complex nested user profiles

A system needs to pull the secondary phone number and primary billing address from a deeply structured JSON profile. They use query_json with multiple specific paths (e.g., `$.contact.secondary_phone` and `$.billing.address`) to guarantee they get exactly what they need.

Patterns to Avoid

Treating the LLM like a JSON parser

✗ AVOID

Asking your agent, 'Can you find all user emails from this huge block of API data?' The agent reads everything, gets distracted by irrelevant fields, and sometimes hallucinates or misses records.

✓ INSTEAD

Use the query_json tool. Feed it the raw JSON and a precise path like `$.users[*].email`. This forces extraction directly, bypassing the LLM's interpretation layer entirely.

Passing overly large payloads

✗ AVOID

Sending an entire 8,000-token transaction log just to check one field. Not only is it slow, but you hit token limits and lose money.

✓ INSTEAD

Use the MCP first. Run query_json with a targeted path like `$.transactions[*].user_id`. You reduce the payload size before your agent even sees it.

Relying on simple text searches

✗ AVOID

Trying to manually search or prompt for 'all numbers that look like IDs.' This is vague, inefficient, and prone to picking up non-ID data.

✓ INSTEAD

Define the structure using JSONPath. Use a path query to hit the exact field you want, such as `$.user_records[*].id`, ensuring accurate extraction every time.

The Right Fit

Use this MCP when your goal is data retrieval: When the information you need is buried inside a large JSON object, and you must extract it programmatically. This tool excels at surgical precision—it answers the question, 'What value exists at this exact path?' Don't use it if you are asking for interpretation (e.g., 'Summarize these API

results') or comparison (e.g., 'Tell me which user is best suited'). For summarization or analysis, send the filtered data *after* using query_json. If your task involves simple filtering based on a single key-value pair across many records, this MCP handles it perfectly with pathing expressions.

The Problem: Context Bloat and Data Noise

Every time you build an agent or script that talks to an API, you get a huge JSON response. This data dump often contains thousands of tokens—user comments, timestamps, unrelated metadata, and dozens of array records. Today, the typical manual process involves copy-pasting this entire raw block into your chat interface or code window and then asking your agent, 'Find me all the author names.' You're forcing an AI model to spend massive compute power reading everything you don't care about.

With this MCP, that whole problem vanishes. Instead of dumping gigabytes of noise, you define exactly what you need using a JSONPath expression and run the query_json tool first. The agent doesn't see the raw data; it only receives a perfectly curated list of results—a clean set of values ready for the next step in your workflow.

JSON Path Query Engine: Pinpoint Values with Precision

The most time-consuming part of API integration is manual data cleanup. You're constantly copying blocks of text, filtering out the metadata fields, and then pasting the few crucial values into a spreadsheet or another system. This process requires multiple clicks across tabs and risks human error every single time.

Now, you define the path once using query_json. The MCP handles the parsing and extraction instantly. You don't copy anything; you just run the tool, and the clean data appears in your agent's context window. It’s immediate, reliable, and precise.

JSON Path Query Engine: 1 Tool

Use this single tool to query raw JSON strings and pull out precise values using specific JSONPath expressions.

#	TOOL	DESCRIPTION
01	`query_json`	Pass a raw JSON string and a path expression to extract every value that matches the defined path.

See It in Action

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

U Extract all author names from this bookstore JSON.

JSONPath Result: 4 matches found.

Frequently Asked Questions

01 **How do I use JSON Path Query Engine with multiple arrays?**

You reference the array structure directly within the path expression using wildcards like `[*]` or specific indices. For example, to get emails from all users, you'd use `$.users[*].email`.

02 **Is JSON Path Query Engine better than just asking my AI client to extract data?**

Yes, because it enforces structure. Your agent relies on its interpretation of the text; this MCP uses established, mathematical pathing logic, guaranteeing that if a value exists at that path, you'll retrieve it.

03 **What is the best way to use query_json?**

Pass the raw JSON string into the tool first. Then, construct your specific path expression based on where the data lives (e.g., `$.results[0].value`). Always start with the most precise path possible.

04 **Does JSON Path Query Engine handle different types of payloads?**

It handles any raw JSON payload, provided that structure is consistent and follows standard JSON syntax. It doesn't care if the data is finance, health records, or gaming scores.

05 **Can I use this MCP to search across multiple nested levels?**

Absolutely. You can use advanced pathing (like `$..field`) to recursively search for a field regardless of how many layers deep it is within the JSON structure.

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 `"json-path-query-engine": { "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

JSON Path Query Engine 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 JSON Path Query Engine. 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	JSON Path Query Engine MCP
Server ID	`019e38b1-ce77-739a-9dec-6ccd355ceb0f`
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/json-path-query-engine.

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