MCP SERVER | NO CODE | CLOUD HOSTED

JSON5 Resilient Parser MCP

Stop failing on trailing commas and comments.

JSON5 Resilient Parser handles bad data structures, specifically messy JSON generated by language models. It takes strings with trailing commas, inline comments, single quotes, and unquoted keys—all things that break standard parsers—and outputs perfect, strict RFC 8259 JSON. If you're dealing with configuration files or API responses from an AI agent, this MCP guarantees clean, machine-readable data every time.

A+ Quality Score 100/100

json5 | data-parsing | resilient-parsing | config-files | data-conversion

MAINTAINED BY VINKIUS CLOUD FOR AI AGENTS · JUNE 2026
vinkius.com/mcp/json5-resilient-parser

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.

JSON5 Resilient Parser MCP

1 tools available
Cloud-hosted on Vinkius

When your agent spits out a config file or a data structure through a chat window, it rarely follows perfect JSON standards. You get trailing commas, comments embedded in the code, and single quotes instead of double quotes. Standard parsers fail immediately. This MCP fixes that failure point before you even write a line of cleanup code.

It acts as a shield for your data flow. It accepts everything from messy keys to hex values and normalizes them into perfect JSON. You connect this service via Vinkius, letting any compatible AI client handle the dirty work so you don't have to. The result is clean, strict RFC 8259 JSON that any downstream system can consume without a single modification.

Core Capabilities

01 — Clean messy data from LLMs

It processes malformed JSON strings—like those generated by generative AI—and corrects syntax errors automatically.

02 — Handle comments and trailing commas

The parser reads in code containing inline comments (// or /* */) and comma overkill, then strips them out while keeping the data intact.

03 — Ensure strict JSON output

It converts non-standard formats into perfectly valid RFC 8259 compliant JSON that every programming language can read reliably.

04 — Process unconventional data types

The MCP correctly handles unusual values like hexadecimal numbers, Infinity, and NaN within the data payload.

One Click on Vinkius — From Prompt to Execution

Available at **vinkius.com/mcp/json5-resilient-parser** — connect your AI agent in three steps.

01 You pass the messy JSON string—the output from your agent or a configuration file—to the `parse_json5` tool.

02 The MCP processes this input using its resilient engine, automatically correcting syntax issues like trailing commas and comments.

03 Your agent receives perfect RFC 8259 compliant JSON ready for immediate use in your application logic.

The bottom line is you get clean data structures without writing a single cleanup script.

Built For

This MCP is essential for anyone building complex agent workflows or integrating AI outputs into production code. If your process relies on reading structured data generated by an LLM, you need this. It's for the engineers who know that 'almost correct' JSON isn't good enough.

Prompt Engineer

Using this MCP ensures that complex prompts designed to generate structured data actually result in usable, parsable output every time.

Data Scientist

When feeding LLM-generated metadata or configuration snippets into analysis pipelines, the parser guarantees the input is clean and standardized for reliable metrics.

Backend Engineer

You rely on agents to pass back structured data; this MCP acts as a guaranteed parsing layer, preventing runtime serialization errors in your production service.

What Changes When You Connect

01 Prevents runtime crashes: Because the `parse_json5` tool handles common JSON syntax failures (like extra commas or unquoted keys), your agent workflow won't fail when receiving imperfect data from an LLM.

02 Guaranteed output format: It forces all input into strict RFC 8259 JSON. This means you don't need to write custom cleanup logic for every new type of messy syntax the AI might produce.

03 Handles complex edge cases: The parser correctly reads and converts unusual data types, including hexadecimal numbers and special floating-point values like Infinity/NaN that standard parsers choke on.

04 Saves development time: Instead of spending hours writing fragile regex or cleanup scripts to fix LLM output, you use this MCP as a single, reliable parsing step.

05 Supports multi-source data: You can confidently pipe structured data from various sources—whether it's code comments or chat responses—through one consistent cleaning layer.

Real-World Applications

Parsing a complex config file generated by an agent

A backend engineer asks their agent to generate a multi-part YAML configuration, but the output includes comments and trailing commas. Instead of throwing a JSON parsing error, running `parse_json5` cleans up the structure, allowing the service to load the settings correctly.

Cleaning data from an LLM chat window

A prompt engineer gets a large list of user objects from their AI client. The output is formatted like JSON but has single quotes and unquoted keys. Running `parse_json5` instantly converts it into perfect, usable structured data.

Standardizing legacy configuration inputs

A data scientist needs to process a set of old config files that use non-standard JSON syntax (like hex numbers). The MCP handles these unconventional values and outputs them in standard format for the analysis pipeline.

Patterns to Avoid

Assuming clean LLM output

✗ AVOID

Writing code that assumes any data structure received from a generative model will pass through `JSON.parse()`, leading to immediate runtime failures whenever the AI adds an extra comma or comment.

✓ INSTEAD

Always run agent-generated configuration snippets through the `parse_json5` tool first. This guarantees conversion to strict JSON before your application logic attempts to read it.

The Right Fit

Use this MCP if, and only if, the source of your structured data is generated by a large language model or an external system that cannot guarantee perfect RFC 8259 compliance. If you are dealing with configuration files or data pulled from chat interfaces, running it through `parse_json5` is non-negotiable.

Don't use this if your input JSON is already guaranteed to be perfectly formed and clean (e.g., reading directly from a database column that enforces strict typing). If the source is reliable, the overhead of passing data through an extra parsing layer isn't worth it. When in doubt—and with LLMs, you should always be in doubt—run it through `parse_json5`.

The pain point: Copying structured data from chat windows

Right now, if your agent returns a list of records or a complex configuration object, you copy the text into your local file or paste it directly into an API call. You quickly realize that while the structure looks right to the naked eye, every fifth line has a trailing comma, and the keys aren't properly quoted. Copying this data means manually fixing syntax error after syntax error.

With this MCP, you just feed the raw output text to the parser. It automatically detects all those minor structural violations—the missing quotes, the misplaced commas, the embedded comments—and hands you a perfect JSON object every single time. You get clean, production-ready data instantly.

JSON5 Resilient Parser: Getting guaranteed structured output

The manual steps that vanish are the tedious checks for quotes, the hunt for trailing commas after arrays, and the frustration of debugging a `SyntaxError` just because the AI got chatty. You stop relying on guesswork and start trusting the parser.

This MCP eliminates the weakest link in your agent chain: human error or LLM imperfection. The output is always clean.

JSON5 Resilient Parser: 1 Tool Available

Use the available tool to process messy data structures generated by AI, ensuring your workflows always receive perfectly formatted JSON.

#	TOOL	DESCRIPTION
01	parse_json5	This tool accepts any JSON5-compliant string and reliably outputs perfect RFC 8259 JSON, making it essential for cleaning data from generative models.

See It in Action

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

U Parse this JSON with trailing commas and comments: {name: 'Alice', age: 30, // years old}

Strict JSON Output: {"name": "Alice", "age": 30}

U Clean up this LLM-generated config that has single quotes and trailing commas.

Strict JSON Output: Clean RFC 8259 JSON generated.

U Convert this JSON5 with hex values and Infinity to strict JSON.

Strict JSON Output: All values normalized to standard JSON types.

Frequently Asked Questions

01 **Does JSON5 Resilient Parser handle standard JSON correctly?**

Yes, it does. Since it's designed to output strict RFC 8259 JSON, any perfectly formed JSON will be processed and returned in the same correct format. It doesn't break on clean data.

02 **Can I use parse_json5 for configuration files?**

Absolutely. This is one of its primary uses. If your config file was generated by an agent and contains comments or extra commas, `parse_json5` cleans it up so your application can read the settings.

03 **What kind of data does JSON5 Resilient Parser handle?**

It handles any string that resembles JSON but might be malformed. This includes unquoted keys, trailing commas, and comments (// or /* */).

04 **Is this better than just using standard JSON libraries?**

Yes, because standard libraries fail on messy input. This MCP is designed specifically to catch the common syntax errors that LLMs inevitably introduce, providing a robust layer of resilience.

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 `"json5-resilient-parser": { "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

JSON5 Resilient Parser 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 JSON5 Resilient Parser. 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	JSON5 Resilient Parser MCP
Server ID	`019e38b2-286a-73a2-b4a0-7cf4f2436c1d`
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/json5-resilient-parser.