MCP SERVER | NO CODE | CLOUD HOSTED

Regex Extractor Engine MCP

Guaranteed Extraction. Zero Hallucination.

Regex Extractor Engine runs pure, deterministic JavaScript Regular Expressions on large text blocks. Stop relying on an AI agent to guess data; this MCP guarantees 100% accurate array extraction for emails, UUIDs, IPs, and custom tokens without hallucinating a single match.

A+ Quality Score 100/100

regex | pattern-matching | data-extraction | deterministic-parsing | text-processing | validation

MAINTAINED BY VINKIUS CLOUD FOR AI AGENTS · JUNE 2026

vinkius.com/mcp/regex-extractor-engine

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.

Regex Extractor Engine MCP

1 tools available
Cloud-hosted on Vinkius

When you're dealing with logs, scraped websites, or massive dumps of raw text, you need precision. Generic language models are great at summarizing content, but they struggle when the task is strict pattern matching. They might miss complex nested structures or, worse, invent data that looks plausible but isn't real. This MCP solves that problem by bringing pure JavaScript RegExp evaluation directly to your agent.

It forces mathematical accuracy onto data extraction. You define the exact pattern you need—say, a specific UUID format—and this tool only pulls matches that fit that mold. If the pattern doesn't exist in the text, it returns nothing. It never guesses an email address or invents a fake phone number. This level of deterministic control is critical for reliable data pipelines. You connect it through Vinkius and suddenly your agent can perform surgical extractions on complex documents, giving you clean, predictable arrays every single time.

Core Capabilities

01 — Extract specific formats from text

The tool takes a large body of text and a defined regular expression pattern, returning only the exact array of matches found.

02 — Validate data structure integrity

You can use the engine to check if strings—like IP addresses or UUIDs—adhere perfectly to established format rules.

One Click on Vinkius — From Prompt to Execution

Available at **vinkius.com/mcp/regex-extractor-engine** — connect your AI agent in three steps.

01 Provide your agent with two inputs: the massive block of text you want to analyze, and the specific regular expression pattern defining what you are looking for.

02 The MCP runs this definition using pure JavaScript logic against the provided text, checking every character against the rules you set.

03 You get back a clean, precise array containing only the strings that perfectly match your required format.

The bottom line is: it gives you mathematical certainty when extracting structured data from unstructured noise.

Built For

Data analysts and QA engineers who spend their days sifting through log files, JSON dumps, or scraped content need this. If your workflow relies on accurate pattern matching—like finding every valid transaction ID in a multi-page report—you're hitting the wall with standard LLM outputs.

Data Analyst

They use it to process large datasets of unstructured text, reliably pulling out key metrics like tracking IDs and dates without manual cleanup.

QA Engineer

They run it against sample system logs or API outputs to validate that every expected error code, UUID, or session token is present in the correct format.

DevOps Engineer

They use it when parsing complex server logs to extract specific operational data points, like IP addresses and timestamps, for incident reporting.

What Changes When You Connect

01 Absolute Precision: Instead of relying on an agent's best guess, you define the rules and get mathematically perfect extractions for emails, phone numbers, or UUIDs using regex_extractor_extract.

02 Eliminate Hallucinations: This MCP never makes up data. If your pattern isn't in the text, it returns nothing. You stop wasting time correcting plausible-sounding but fake matches.

03 Handles Complexity: It processes complex nested patterns that standard LLM context windows often fail to parse correctly on the first try.

04 Native Speed: Running the regex engine natively means you get lightning-fast processing for massive text blocks, a speed advantage over general-purpose AI parsing.

05 Universal Patterns: You don't have to change tools when your data changes. Whether you need to validate IPv4 addresses or custom tokens, the underlying logic remains deterministic.

Real-World Applications

Parsing multi-line server logs

A DevOps engineer needs to find every unique UUID mentioned across a 50MB log file. Instead of asking their agent to 'extract the IDs,' they use regex_extractor_extract with a UUID pattern, guaranteeing zero missed records and no fake IDs.

Scraping contact information from websites

A data analyst pulls text dumps from several competitor sites. To reliably gather all valid email addresses, they use the engine to run against a comprehensive regex pattern for emails, getting a clean list without needing manual filtering.

Validating batch transaction records

A QA engineer has received a large file containing simulated financial transactions. They use the MCP to validate that every single record's associated account number matches the specific format, failing fast if any data is malformed.

Extracting structured metadata from documents

A technical writer has a document containing mixed text and embedded codes. To pull out all internal reference numbers (e.g., 'REF-XXXX-YYYY'), they use regex_extractor_extract to isolate the exact pattern across the entire file.

Patterns to Avoid

Asking an AI agent for extraction

✗ AVOID

Prompting your agent: 'Find all UUIDs in this log.' The agent might miss a tricky nested ID or accidentally invent one that looks correct.

✓ INSTEAD

Use the regex_extractor_extract tool. You provide both the text and the precise pattern, forcing the engine to return only mathematically guaranteed matches.

Using simple string search

✗ AVOID

Manually searching for 'email' and then copy-pasting results into a spreadsheet, risking missing emails with slight variations.

✓ INSTEAD

Use regex_extractor_extract. Define an email pattern (e.g., `[a-zA-Z0-9._%+-]+@[a-zA-Z0-9.-]+.[a-zA-Z]{2}`) and run it across the full text block to capture everything.

Relying on context inference

✗ AVOID

Asking your agent, 'What are the important numbers here?' The response is vague because the AI doesn't know if you mean IPs, dates, or serial codes.

✓ INSTEAD

Use regex_extractor_extract and define the exact pattern for what you need. For instance, run a specific IP address regex to get only valid IPv4 formats.

The Right Fit

Use this MCP if your primary goal is deterministic data extraction—meaning the output *must* be 100% accurate based on defined rules. If you are pulling structured data (like UUIDs, phone numbers, or specific codes) from unstructured text, this tool is non-negotiable.

Don't use it if your task requires contextual understanding, summarization, or creative writing. For example, if you need the agent to summarize a chapter of text or explain *why* an error occurred, using regex_extractor_extract will fail because it only handles patterns, not meaning. If all you have is general messy text and no pattern idea, try a standard LLM; but if you know the format (e.g., 'every line starts with DATE:'), this MCP gives you the reliable structure you need.

The Pain of Copy-Pasting Data from Logs

When a system fails, engineers are dumped massive log files. Today's process involves reading through thousands of lines, manually scanning for error codes or transaction IDs, and then copy-pasting those snippets into a spreadsheet or ticketing system. It’s tedious work, prone to human oversight, and you almost always miss something.

With this MCP, the same task changes completely. You point your agent at the log file and define what an 'error code' looks like using its specific pattern. The engine processes the entire text block in seconds, returning a clean array of only the matching codes. What you get is reliable data, not just guesswork.

---

regex_extractor_extract: Deterministic Extraction

You eliminate the need to read through error messages line by line and manually isolate IDs or timestamps. You don't have to worry about inconsistent formatting—if it doesn't match your regex, it simply won't be extracted.

This MCP gives you control back. It ensures that every piece of data pulled into your workflow is verifiable against the rules you set, making your entire process dependable.

Regex Extractor Engine: 1 Tool

Use this single tool to process raw text blocks by applying precise regular expressions to extract structured data arrays.

#	TOOL	DESCRIPTION
01	`regex_extractor_extract`	Passes a text block and a pattern to retrieve an array of all exact string matches using regular expressions.

See It in Action

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

U Extract all UUIDs from this server log file.

✅ **Extracted (2):**
- `123e4567-e89b-12d3-a456-426614174000`
- `987f6543-a21c-34f5-b678-998877665544`

U Find every email ending in @vinkius.com in this text block.

✅ **Matches Found:**
1. `admin@vinkius.com`
2. `sales-team@vinkius.com`

U Validate if '192.168.1.255' matches a standard IPv4 address format.

✅ **Validation Passed:** The string perfectly matches the IPv4 regular expression.

Frequently Asked Questions

01 **Does regex_extractor_extract work on very large text files?**

Yes, it's designed for massive blocks of text. Because it uses native JavaScript evaluation, performance is fast and scalable even with huge log dumps.

02 **Can I use regex_extractor_extract to find phone numbers in different countries?**

Absolutely. You just need to modify the pattern you provide. The engine handles the complexity; you just define the required format.

03 **Is this better than asking an AI agent for UUIDs using regex_extractor_extract?**

Yes, because it's deterministic. An AI agent might hallucinate or miss matches; this MCP only extracts what mathematically fits the pattern you define.

04 **What kind of data patterns can I use with regex_extractor_extract?**

You can write any standard JavaScript RegExp pattern, covering emails, IPs, custom tokens, date formats, and anything else that follows a defined 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 `"regex-extractor-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

Regex Extractor 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 Regex Extractor 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	Regex Extractor Engine MCP
Server ID	`019eb8f6-9acb-7050-a685-306668450237`
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/regex-extractor-engine.