MCP SERVER | NO CODE | CLOUD HOSTED

Phone Validator Engine MCP

Stop passing bad phone numbers to your systems.

Phone Validator Engine validates and formats messy phone numbers into standardized E.164 format. This MCP stops your AI clients from hallucinating incorrect or unusable phone data, ensuring every number passed to downstream services is mathematically perfect and ready for telecom APIs.

A+ Quality Score 100/100

phone-validation | e164 | data-cleaning | formatting | telecom | input-sanitization

MAINTAINED BY VINKIUS CLOUD FOR AI AGENTS · JUNE 2026
vinkius.com/mcp/phone-validator-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.

Phone Validator Engine MCP

1 tools available
Cloud-hosted on Vinkius

LLMs are great at chat, but they hate structured data like phone numbers. When users type in messy inputs—like `(555) 123-4567` or `+1-555-123-4567`—your agent might pass the bad string to a system that needs pure E.164 format, and it will fail. This MCP brings Google's phone number validation logic directly into your workflow. It checks if an input is structurally valid and cleans up all the extra characters until you get one reliable standard: `+CCXXXXXXXXXX`. Connecting this validator through Vinkius means any compatible AI client can run a quick check before sending data, eliminating failure points related to formatting or country code ambiguity.

Core Capabilities

01 — Standardize phone number formats

It cleans up messy inputs—parentheses, dashes, spaces—and outputs only the required E.164 string.

02 — Validate structure and existence

The engine checks if the number sequence actually follows a plausible phone number pattern for its supposed country.

03 — Determine country code

It identifies the originating country of the number, even if the user didn't specify it initially.

04 — Prevent data hallucinations

Your agent can run a check on any raw text input to guarantee phone numbers are accurate before actioning anything.

One Click on Vinkius — From Prompt to Execution

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

01 You pass the raw, unvalidated phone number string (and optionally a country code) to the MCP.

02 The engine runs Google's library logic against that input, checking its structure and assigning it a definitive E.164 format.

03 You receive back a clear status: either a confirmed, standardized E.164 number or an explicit failure message.

The bottom line is you get guaranteed data quality for phone numbers before they leave your agent's hands.

Built For

Sales Operations teams, support engineers, and developers building customer-facing AI agents need this. If your workflow relies on accurately communicating with customers via phone number data, you need this validator.

Customer Support Engineer

They use it to clean up messy contact details scraped from user profiles or chat transcripts before creating a support ticket.

Sales Operations Analyst

They integrate it into lead generation pipelines, ensuring every phone number captured during outreach is correctly formatted for CRM entry.

Backend Developer

They use it to validate user inputs in code or agent actions, preventing API failures when sending data to communication services.

What Changes When You Connect

01 Avoid API failures. Instead of sending a messy string that breaks your telecom service, the `validate_phone` tool guarantees the output is in strict E.164 format.

02 Save manual cleanup time. You don't have to write custom regex for every country code; this MCP handles global phone number formatting automatically.

03 Improve data integrity instantly. Before a number hits your database, you run it through validation, confirming its structural validity and identifying the correct country source.

04 Reduce LLM risk. Your agents won't hallucinate bad contacts. They validate every input using `validate_phone` before suggesting an action or creating a record.

05 Support global operations. Whether the number is from Brazil, France, or the US, the engine processes it correctly and reliably.

Real-World Applications

Cleaning up scraped lead data

A Sales Ops analyst scrapes 100 leads from a website. Instead of copy-pasting raw numbers that use dashes and parentheses, they run the batch through `validate_phone`. The output is a clean list of E.164 numbers ready to import directly into the CRM.

Handling user chat inputs

A support agent uses an AI assistant to summarize a customer's issue, which includes phone numbers typed in various formats (e.g., `(212) 555-1234`). They pass the number through `validate_phone` before creating a ticket, guaranteeing the contact method is usable.

Validating agent conversation context

A developer builds an agent that needs to confirm a client's phone number. Rather than letting the raw chat text pass through, the agent first calls `validate_phone` on the user input, immediately flagging it if the structure is wrong.

Building reliable signup forms

When integrating this MCP into an automated sign-up flow, you use its validation to ensure that even if a user enters text instead of digits, the system catches it and forces correct formatting before accepting the data.

Patterns to Avoid

Passing raw input directly

✗ AVOID

An agent passes `555-1234` to a messaging API. The API rejects it because it expects an international code, causing the whole workflow to fail and requiring manual intervention.

✓ INSTEAD

Always use the `validate_phone` tool first. If you pass the number with context (e.g., country code 1), the tool ensures the output is standardized as `+15551234` before it leaves your agent.

Relying on LLM formatting

✗ AVOID

You ask your AI client to 'clean up this number' and trust its text output. The AI might correctly format the dashes but miss a necessary country code or misinterpret location.

✓ INSTEAD

Don't rely on natural language processing for structured data. Use `validate_phone` because it leverages industry-standard phone number libraries, not just pattern matching.

Skipping validation steps

✗ AVOID

A multi-step agent flow processes contact details but skips the validation step to save latency time. Later in the process, a downstream system fails because of an improperly formatted phone number.

✓ INSTEAD

Make `validate_phone` a required checkpoint early in your workflow. It's a microsecond cost that prevents hours of data cleanup and failed API calls.

The Right Fit

Use this MCP if the primary goal is ensuring the structural integrity and standardized format of phone numbers. If you are dealing with any user-provided or scraped contact number, validation should be step one. Don't use it if your task is merely to translate text or summarize a conversation—you need a general NLP tool for that. However, if you only need to remove dashes from a simple list and don't care about whether the resulting string represents a real phone number (e.g., just cleaning up titles), then this MCP is overkill. The key difference is structure: `validate_phone` doesn't just clean text; it validates against global telecommunication standards, telling you if the number *could* exist.

Contact data cleanup is a constant headache.

Today, every time you get contact info from a website form or scrape a list, you face messy inputs. You've got parentheses, dashes, spaces, and sometimes the country code is missing entirely. Your team spends hours building complicated scripts just to strip out those characters and guess which format the recipient system needs.

With this MCP, that manual cleanup vanishes. You feed in the messiest string—the one with every punctuation mark imaginable—and get back a single, clean E.164 number. It's reliable data that hits your downstream services immediately.

Using the validate_phone tool ensures perfect compliance.

You don't have to worry about writing specific logic for every region, like how US numbers differ from UK numbers or Brazilian numbers. You just send it to `validate_phone`, and the engine handles all those international rules for you.

The difference is simple: your agent doesn't guess at formats anymore; it uses industry-grade libraries to guarantee accuracy. The data flow is clean, compliant, and ready to use.

Phone Validator Engine: 1 Tool

Use this single tool to take messy phone numbers from any source and guarantee they are clean, structured, and ready for telecom systems.

#	TOOL	DESCRIPTION
01	validate_phone	Pass a raw phone number string and an optional country code; the tool returns a validated, formatted E.164 standard number or flags it as invalid.

See It in Action

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

U Clean and format this messy user input into an E.164 phone number: '+44 (0) 20 7946 0958'

✅ **Result:** Perfectly mapped to E.164 format: `+442079460958`

U Check if the input `11 98888-0000` is structurally valid, assuming the default country is Brazil (`BR`).

✅ **Validation Status:** Valid.
- **Country:** `BR`
- **E.164:** `+5511988880000`

U Extract the Alpha-2 Country Code from this international number: `+1-415-555-2671`.

✅ **Country Identified:** The number strictly belongs to `US` (United States).

Frequently Asked Questions

01 **What kind of inputs does the Phone Validator Engine handle?**

It handles virtually any raw phone string input—including those with spaces, dashes, parentheses, or international prefixes. The tool strips all non-numeric characters to standardize the data.

02 **Does validate_phone detect fake numbers?**

Yes, it validates structure against known global patterns. If a number sequence is structurally impossible for its purported country code, the tool flags it as invalid.

03 **Is E.164 format mandatory when using validate_phone?**

Yes, the primary function of this MCP is to convert all inputs into the universally accepted E.164 standard. This ensures compatibility with global communication APIs.

04 **Can I tell validate_phone what country code to assume?**

You can pass an optional country code alongside the raw number string. This helps the tool accurately determine the intended format if the number itself is ambiguous.

05 **What happens if validate_phone fails?**

The tool returns a clear failure message, allowing your agent to catch the error immediately and prompt the user for corrected input instead of proceeding with bad data.

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 `"phone-validator-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

Phone Validator 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 Phone Validator 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	Phone Validator Engine MCP
Server ID	`019e38d6-a71c-7013-9431-5b88cffc19b7`
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/phone-validator-engine.