MCP SERVER | NO CODE | CLOUD HOSTED

AfterShip MCP for AI Agents

Track global package shipments from any carrier's data

AfterShip connects your AI agent to global package tracking data. Monitor shipments from over 1,000 couriers worldwide using natural language commands. You can create new trackings, get the latest status updates, auto-detect which courier is handling a number, and even set up email or SMS alerts for delivery milestones.

A+ Quality Score 100/100

package-tracking | delivery-updates | courier-integration | shipment-monitoring | notifications

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

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.

AfterShip MCP

9 tools available
Cloud-hosted on Vinkius

This MCP lets your agent manage complex logistics tasks without you ever leaving your chat window. Instead of visiting dozen different carrier websites to check on a package, you just ask your AI client, and it does the work. You can instantly monitor shipments from major carriers like FedEx or DHL, even if you only have a tracking number. If the shipment status is unclear, the tool automatically detects which courier is responsible for that number so you don't have to guess.

It’s built to handle the full lifecycle of an order: starting by creating a new package record with all necessary details, and finishing by marking it as delivered when the carrier hasn't updated the status yet. If you're already using Vinkius for other integrations, adding AfterShip makes your agent capable of handling end-to-end supply chain visibility, simplifying operations immediately.

Core Capabilities

01 — Create a new package tracking entry

Initiate monitoring for a shipment by providing the core tracking number and optional details like order IDs or customer emails.

02 — Find out which carrier handles a number

Auto-detect the likely shipping company when you only know the tracking number, returning a ranked list of possibilities.

03 — Retrieve specific shipment details

Pull the current and full history for any active or past package tracking entry.

04 — List all monitored shipments

View a filtered list of every package you've tracked, narrowing down results by date range or delivery status.

05 — Mark a shipment as delivered

Update the record when physical evidence shows the package has arrived, even if the carrier hasn't posted the final status online.

06 — Restart monitoring for expired tracks

Refresh an old or stalled tracking entry to pull the most current checkpoints and updates from the carrier.

One Click on Vinkius — From Prompt to Execution

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

01 First, connect your preferred AI client to this MCP on Vinkius and enter your AfterShip API Key.

02 Next, prompt your agent with a specific request, like 'Track order 12345' or 'List all pending shipments.'

03 The agent uses the appropriate tool (like `get_tracking` or `list_trackings`) to gather real-time data and presents the status directly in plain language.

The bottom line is that your AI client acts as a single interface, consolidating information from thousands of carrier systems into one chat response.

Built For

Anyone managing e-commerce fulfillment or logistics data needs this. If you spend time manually cross-referencing tracking numbers across different courier websites (USPS, DHL, etc.), your agent needs AfterShip.

Operations Manager

They use it to list and monitor large batches of shipments, quickly identifying bottlenecks or packages that are stuck in transit.

Customer Service Agent

They rely on the MCP to provide an immediate, accurate status update for a customer by simply entering the tracking number into their chat window.

E-commerce Developer

They integrate it to build reliable workflows that automatically send out SMS notifications when delivery milestones are hit.

What Changes When You Connect

01 Instantly determine the courier: When you only have a tracking number, use `detect_courier` to find out which of the 1000+ carriers handles it. No guessing required.

02 Full visibility into history: Instead of seeing just 'In Transit,' the tool provides detailed checkpoints and timestamps, giving you complete delivery history with `get_tracking`.

03 Keep your records clean: Use `list_trackings` to pull all active shipments across multiple couriers, letting you manage status by date range or specific tag.

04 Automate finalization: When a package arrives but the carrier fails to update its system, use `mark_tracking_completed` to accurately close out your internal records.

05 Handle stalled deliveries: If a tracking number seems stuck in limbo, trigger `retrack_tracking` to force a fresh pull of status updates and checkpoint data.

Real-World Applications

A customer asks about an unknown package

The user pastes a 12-digit tracking number. The agent calls `detect_courier`, identifies it as USPS, and then uses `get_tracking` to provide the current status: 'Out for Delivery today.' The customer is immediately satisfied.

Batch updating old orders

The Ops Manager needs to update 50 packages that were physically delivered last week but haven't updated their internal tracking records. They use `list_trackings` to find the batch, and then call `mark_tracking_completed` for all of them in one go.

Auditing supply chain data

A developer needs a list of every shipment that is currently 'Pending' across multiple couriers. They use the filtering capabilities of `list_trackings` to pull a clean, actionable dataset for reporting.

Resuming an interrupted workflow

The agent was tasked with tracking a shipment but lost connection before completion. The user can now simply tell it to 'recheck that package status,' triggering `retrack_tracking` and getting the absolute latest checkpoints.

Patterns to Avoid

Listing tools manually

✗ AVOID

Trying to list all tracking functions by name (e.g., 'I need get_tracking, then also delete_tracking'). This is tedious and fails if a new tool is added.

✓ INSTEAD

Just tell your agent what you want done in plain English: 'Can you track the package for order 9876?' The MCP handles mapping that request to `get_tracking` or `create_tracking` automatically.

Assuming carrier knowledge

✗ AVOID

Getting frustrated because the agent asks, 'Which courier is this?' even though you know it's DHL. You waste time explaining basic logistics.

✓ INSTEAD

Don't worry about knowing; just give the number. The `detect_courier` tool handles that guesswork for you by returning a confident list of potential carriers.

Over-complicating status updates

✗ AVOID

Manually writing complex instructions like 'If it says delivered, use mark_tracking_completed.' This requires rigid scripting.

✓ INSTEAD

Just prompt: 'The package arrived this morning.' The agent interprets that action and correctly calls `mark_tracking_completed`, simplifying the whole process.

The Right Fit

Use AfterShip if your core problem is visibility. If you frequently deal with packages coming from multiple, disparate carriers (UPS, USPS, DHL) and need one single source of truth for status updates or history, this MCP is essential. Don't use it if your only goal is to manage a simple database of numbers; then, a basic data storage tool will suffice. You also don't need this if you are building a system that handles payment processing or inventory counts—for those, look at finance or warehouse management tools instead.

AfterShip MCP for AI Agents: Solving Multi-Carrier Tracking Pain Points

Right now, tracking an e-commerce shipment is a nightmare of tabs. You get the number, and then you have to copy it into FedEx's site, switch over to USPS's page, and check DHL's portal. If anything looks weird or if the carrier changes mid-route, you waste minutes just cross-referencing status updates.

With this MCP, your agent handles that entire mess for you. You give it a number, and whether it's UPS, USPS, or one of the 1000+ others, the data appears consolidated in your chat window. It's instant, comprehensive tracking history without opening a single browser tab.

AfterShip MCP for AI Agents: Managing Delivery Status Updates and History

The most tedious part of logistics is managing the final status. You know the package arrived, but the carrier's API hasn't updated its website. You have to manually log into a portal and change the record from 'In Transit' to 'Delivered.'

Now, your agent handles that manual step. If you confirm delivery, you simply tell it, and the MCP executes `mark_tracking_completed`. It keeps your records clean and accurate, giving you reliable data without any extra clicks.

AfterShip: 9 Tools for Shipment Tracking & Logistics Management

Use these tools to create new tracking records, check statuses from thousands of couriers, list all shipments, and manage delivery milestones automatically.

#	TOOL	DESCRIPTION
01	create_tracking	Creates a new package tracking record, accepting the number and optional details like customer emails or order IDs.
02	delete_tracking	Removes a specific tracking entry from your monitored list; this action cannot be undone.
03	detect_courier	Uses a provided tracking number to return a ranked list of the most likely shipping companies responsible for it.
04	get_tracking	Retrieves all details and checkpoints for a single, known package tracking ID.
05	list_couriers	Returns a comprehensive list of all supported courier companies that can be used for monitoring packages.
06	list_trackings	Retrieves a filtered list of all monitored package trackings, allowing searches by date range or status.
07	mark_tracking_completed	Updates the status of an existing tracking record to 'delivered' when manual confirmation is available.
08	retrack_tracking	Restarts the monitoring process for a stalled tracking entry, fetching fresh checkpoint updates from the carrier.
09	update_tracking	Makes minor updates to an existing tracking record without changing the core number or courier information.

See It in Action

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

U I need to know what happened with 9400111899223344556677?

📦 **Tracking Summary for USPS**

- **Status:** Delivered
- **Estimated Delivery:** Jan 18, 2024
- **Latest Checkpoint:** Signed for at recipient's home on Jan 18.

History Log:
- Arrived at local facility: Jan 17 (3 PM)
- Out for delivery: Jan 18 (6 AM)

U What couriers handle this number?

🔍 **Courier Detection Results**

Based on the format, we're highly confident it's a USPS package.

- **Primary Match:** USPS (98% confidence)
- **Secondary Options:** DHL eCommerce (1.5%), UPS Ground (0.5%)

U Show me all shipments pending delivery this week.

📋 **Active Shipment List**

You have 3 packages flagged for action:

- **Order #A12:** FedEx | Status: Out for Delivery. Expected today.
- **Order #B45:** DHL | Status: Pending Pickup. Needs attention.
- **Order #C90:** UPS | Status: In Transit (Chicago Facility). ETA tomorrow.

Frequently Asked Questions

01 **How does the AfterShip MCP help me track packages from multiple carriers?**

The AfterShip MCP consolidates tracking data from over 1,000 couriers into a single chat interface. You don't need to visit separate carrier websites; your agent pulls all status updates for you.

02 **Can I use the AfterShip MCP if I only have a partial or unknown tracking number?**

Yes. If you just provide the number, the tool can automatically detect which courier handles it by analyzing the number's format. It returns a list of likely carriers and their confidence scores.

03 **Is there an easy way to update my records when packages are delivered?**

The AfterShip MCP lets you mark shipments as delivered using `mark_tracking_completed`. This keeps your internal inventory records accurate, even if the carrier hasn't updated its site yet.

04 **Does AfterShip help me see my shipment history?**

Yes. You can use the MCP to retrieve complete delivery histories, including specific checkpoint timestamps and locations throughout the entire journey of the package.

05 **What if a tracking status seems stuck or expired?**

You don't have to wait for it to update. The AfterShip MCP has functions that restart monitoring on stalled entries, forcing a fresh pull of checkpoint data from the courier.

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 `"aftership": { "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

AfterShip 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 →

INDEPENDENT PLATFORM DISCLAIMER

Vinkius is an independent platform and is not affiliated with, endorsed by, sponsored by, verified by, or otherwise authorized by AfterShip. 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	AfterShip MCP
Server ID	`019d8413-86ca-72f7-bad8-4fddabe6533c`
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/aftership.