MCP SERVER | NO CODE | CLOUD HOSTED

AfterShip Returns MCP for AI Agents

Automating E-commerce Reverse Logistics and Return Tracking

AfterShip Returns automates your entire reverse logistics cycle. Manage return requests, process Return Merchandise Authorizations (RMAs), and generate shipping labels using natural conversation through AI agents.

F Quality Score 4.4/100

returns-management | rma-processing | shipping-labels | reverse-logistics | customer-experience | automated-returns

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

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 Returns MCP

4 tools available
Cloud-hosted on Vinkius

Managing returns shouldn't feel like a full-time job of checking spreadsheets and emailing people for tracking numbers. This MCP connects your existing AfterShip Returns account directly to your AI agent, giving you conversational control over complex reverse logistics. You can use the agent to audit pending requests, pulling up item details and return reasons for any specific RMA. Need a label? The agent handles that process instantly. Better yet, when items arrive at your warehouse, you just tell the AI client what condition they're in, and it logs everything so your team never misses an inventory grade or receipt date. It centralizes all that complicated status checking into one conversation flow. When you build this connection through Vinkius, your agent becomes the single source of truth for every piece of returned inventory.

Core Capabilities

01 — Audit Customer and Historical Return Requests

List all pending or past customer return requests, including their current processing status.

02 — Get Detailed RMA Status and Contents

Retrieve specific information about a given RMA, listing items and the original reason for the return.

03 — Approve Returns and Generate Shipping Labels

Authorize a pending request immediately, which triggers the system to create the necessary return shipping label.

04 — Log Item Arrival and Grading at Warehouse

Record when returned goods arrive and log their physical condition or grade in the inventory system.

One Click on Vinkius — From Prompt to Execution

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

01 First, subscribe to this MCP on Vinkius.

02 Next, enter your AfterShip API Key into the connection settings.

03 Finally, use your preferred AI client (Claude, Cursor, etc.) to chat with the agent and start managing returns.

The bottom line is that you talk to your agent like talking to a colleague; it handles the rest of the API calls and logistics updates for you.

Built For

This MCP is essential for e-commerce operations, warehouse managers, or customer support leads who spend too much time juggling multiple systems (spreadsheets, carrier websites, CRM) just to track a single return. If your team answers 'yes' to any of these pain points, you need this.

E-commerce Manager

Automating the approval flow for returns and monitoring overall customer satisfaction trends related to reverse logistics.

Warehouse Operations Lead

Logging item arrivals and grading condition in real time, ensuring inventory records match physical stock without manual data entry.

Customer Support Agent

Quickly pulling up detailed return statuses or generating a tracking label for a customer on demand during a chat interaction.

What Changes When You Connect

01 You instantly audit all return requests using `list_returns`, seeing which items are pending approval or stuck in a status.

02 Get deep context on any specific RMA with `get_return_details`, pulling up item names, original reasons for return, and current shipping location data.

03 Instantly approve returns via `approve_return`. This single action triggers the necessary label generation and notifies the customer—no manual clicks needed.

04 Streamline warehouse intake using `receive_items`. You log items upon arrival and record their physical condition/grade in one conversation, updating inventory records immediately.

05 Centralize process insights. Instead of checking five different dashboards, your agent aggregates common return reasons or identifies policy bottlenecks directly from the chat.

Real-World Applications

A customer calls asking where their refund is coming from.

The support agent asks the agent to look up the request. The agent uses `get_return_details` and reports that the return was received, graded 'Good', and the payment process has started, giving the customer a concrete answer immediately.

A batch of returns arrived at the warehouse unexpectedly.

The ops manager prompts the agent to log them. The agent uses `receive_items` for all 20 boxes, logging each item's condition (e.g., 'Minor scratch', 'Good') and updating the system inventory record.

A customer asks if they can send back an item that was marked as expired.

The manager uses `list_returns` to see all requests from the last week. They spot a pending request, use `get_return_details` to confirm the policy violation, and then manually approves it while documenting the exception.

An order needs to be returned quickly for a replacement shipment.

The agent uses `approve_return` on the request ID. The system confirms approval and generates the necessary return label right in the chat, allowing the customer to print it immediately.

Patterns to Avoid

Using Spreadsheets for Inventory Grading

✗ AVOID

A warehouse worker manually updates a giant Excel sheet every time an item arrives, risking data entry errors and losing track of which items need quality checks.

✓ INSTEAD

Use the agent to execute `receive_items`. You simply tell it the batch ID and the condition ('Good', 'Defective'), and it logs everything accurately in the source system.

Tracking Returns via Multiple Carrier Sites

✗ AVOID

The support team has to open separate carrier websites (UPS, FedEx) and manually check every tracking number just to tell a customer if their package is 'In Transit' or 'Delivered'.

✓ INSTEAD

Use the agent with `get_return_details` to pull all consolidated logistics status information for an RMA in one go. Your client gives you one answer.

Forgetting to Approve Returns Promptly

✗ AVOID

A customer's return request sits in the queue because nobody had time to click the 'Approve' button, delaying the label and stalling the refund process.

✓ INSTEAD

Use `list_returns` to identify all stalled requests. Then use `approve_return` on the pending IDs to clear the backlog and instantly generate necessary labels.

The Right Fit

You need this MCP if your return process involves multiple handoffs, conditional approvals, or physical logging steps that currently require manual data transfer between systems. Use it when you have a queue of requests (use `list_returns`) and the next step depends on specific item details (`get_return_details`). Don't use it if you only need to count how many items are in stock; for basic inventory counts, a dedicated inventory management tool is simpler. Only use this MCP when your process involves the full lifecycle: request status -> detailed review -> physical receipt logging.

AfterShip Returns MCP for AI Agents: Automating E-commerce Return Request Audits

Right now, managing returns means jumping between your e-commerce backend and carrier portals. You have to manually pull lists of pending requests, check if they've been approved, cross-reference the items against original order data, and then figure out who needs to take action.

With this MCP, you prompt your agent with a question like 'Show me all returns for Q3 that are stuck.' The agent uses `list_returns` and `get_return_details` behind the scenes. You get a consolidated list of issues, pinpointing exactly why an RMA is stalled without opening a single tab.

AfterShip Returns MCP for AI Agents: Streamlining Warehouse Logistics Grading

Manual warehouse processing involves receiving boxes, writing down the item name and condition on paper, and then having someone manually enter that data into an inventory sheet later. This is slow and error-prone.

Now, when items arrive, you tell your agent they're here. The agent uses `receive_items` to log the arrival and grade—saying 'Good working order' or 'Minor scratch'—and updates the system record instantly. The whole process happens in chat.

AfterShip Returns: 4 Tools for Return Management & Grading

Use these tools to manage the entire returns lifecycle, from listing pending requests to logging physical item grades at your warehouse.

#	TOOL	DESCRIPTION
01	`list_returns`	Gets a list of pending and historical customer returns, including their processing status.
02	`get_return_details`	Retrieves specific item details, reasons for return, and the current logistics status for any RMA number.
03	`approve_return`	Authorizes a pending return request immediately, which triggers the generation of the customer's shipping label.
04	`receive_items`	Records the arrival of returned goods at the warehouse and logs their physical condition or grade.

See It in Action

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

U Hey, list all return requests that are pending approval from last week.

Pending Returns Report (Last 7 Days)

- **Request ID:** ret-456xyz
* Customer: John Doe
* Items: Blue Shirt (Size M)
* Reason: Wrong size.
* Status: Awaiting Manager Approval.
- **Request ID:** ret-789abc
* Customer: Jane Smith
* Items: Coffee Maker
* Reason: Defective part.
* Status: Needs Manual Review.

You have 2 items that need action.

U Show me the details for RMA number 'RMA-98765'.

Details for RMA-98765

- **Status:** In Transit (via UPS)
- **Items Included:**
1. 'Blue Cotton Shirt' (Reason: Wrong size)
2. 'Canvas Tote Bag' (Condition: Defective)
- **Notes:** Requires specialized grading due to defect type.

The shipment left the origin on 6/10 and is expected back by 6/15.

U We just received a batch of returns. Log them now.

Item Receipt Confirmation

Successfully logged 4 items:
- Item SKU: TOTE-001 | Condition: Good | Grade: A
- Item SKU: SHRT-BLU | Condition: Damaged | Grade: C (Requires repair)
- Item SKU: MUG-RED | Condition: Good | Grade: B
- Item SKU: BOOK-XYZ | Condition: Good | Grade: A

All items are now marked 'Received' in the inventory system.

Frequently Asked Questions

01 **How does AfterShip Returns MCP handle my return label process?**

It automates the whole thing. You simply approve a pending request through your agent, and it automatically triggers the generation of the correct shipping label for the customer—you don't have to leave the chat.

02 **What if I need to check item details on a specific RMA?**

The MCP lets you pull granular data on any return by specifying the RMA number. You immediately see exactly what items are included and why they were returned, saving deep dives into multiple systems.

03 **Can I use AfterShip Returns MCP to log physical inventory condition?**

Yes, you can. When returns arrive at your warehouse, the agent lets you record the item's arrival date and its grading (e.g., 'Good', 'Minor Scratch') right in the chat interface.

04 **Is AfterShip Returns MCP better than just using my e-commerce backend?**

It's more powerful because it gives you conversational control over everything. Instead of clicking through multiple tabs to check status, approvals, and item grades, you ask the agent, and it does all that work for you.

05 **Does AfterShip Returns MCP help with tracking stalled returns?**

Absolutely. By listing historical requests, your agent shows you which return statuses are stuck or pending action, helping your team clear backlogs faster than manually checking the queue.

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-returns": { "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 Returns 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 AfterShip Returns. 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 Returns MCP
Server ID	`019d7549-4468-7391-8002-2a1a091515e0`
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-returns.