MCP SERVER | NO CODE | CLOUD HOSTED

Portable.io MCP

Manage every step of your data synchronization process.

Portable.io MCP manages your entire data pipeline workflow directly through your AI agent. It lets you check complex integration flows, view execution history for specific sync runs, and monitor destination details across platforms like Snowflake or BigQuery—all without leaving your chat window.

A+ Quality Score 100/100

etl | data-pipeline | saas-integration | data-sync | warehouse-management | data-engineering

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

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.

Portable.io MCP

6 tools available
Cloud-hosted on Vinkius

Need to know why the sales data didn't make it into BigQuery this morning? You don't have to open five different tabs to figure it out. This MCP connects your Portable.io account, letting you talk to your agent about complex ETL pipelines using natural language. Instead of digging through dashboards and API documentation, you ask questions like, 'What were the sync runs for HubSpot yesterday?' Your agent retrieves that history instantly, showing row counts and pinpointing failure logs so you know exactly what went wrong. You can also check account limits or list every data warehouse destination authorized to receive writes. It's a massive time saver. When working with thousands of available tools, Vinkius makes it simple to find the exact data pipeline control you need.

Core Capabilities

01 — Inventory all configured data flows

List every integration flow currently set up within your Portable workspace.

02 — Inspect specific flow configurations

Get the full setup and mapping details for one chosen data synchronization flow.

03 — Review historical run results

Track execution history, checking successful row counts or identifying failure logs for a given flow.

04 — List connected data targets

See all the data warehouses and SaaS extractors authorized to receive raw data from your flows.

05 — Check account usage limits

Instantly retrieve your workspace boundaries and billing execution limits for peace of mind.

One Click on Vinkius — From Prompt to Execution

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

01 Subscribe to this MCP and provide your Portable API key.

02 Your AI client authorizes access, connecting the agent to your data pipeline account.

03 You simply ask your agent a question—like 'Show me the run history for last week's Shopify sync'—and it delivers the answer.

The bottom line is that you get full control over complex data movement and syncing without ever leaving your chat interface.

Built For

This MCP is for the Data Engineer who's tired of switching between ETL dashboards, the Analytics Manager who needs quick answers about delayed loads, or the Ops Manager monitoring multiple SaaS environments.

Data Engineer

Troubleshoots pipeline runs and verifies data mappings without having to jump into a separate web application.

Analytics Manager

Traces delayed warehouse loads or checks connector configurations when preparing for quarterly reports.

DevOps/Ops Manager

Monitors synchronization health across several different SaaS environments to ensure continuous data flow.

What Changes When You Connect

01 Stop context switching. Instead of opening a dozen tabs to check sync status, ask for the `list_runs` history directly. You get immediate failure logs and row counts without leaving your chat.

02 Verify system boundaries instantly. Use `get_account` to check if you're hitting execution limits or need more workspace capacity before launching a major migration.

03 Quickly audit data sources. If you aren't sure what APIs feed your pipelines, use `list_connectors`. You can quickly see every pre-built source available for mapping.

04 Know where the data lands. Before running a new flow, call `list_destinations` to confirm which specific Snowflake or BigQuery schemas are authorized to accept writes.

05 Deep dive into setups. Need to check if a specific sync is configured correctly? Use `get_flow` for full configuration details instead of guessing through the UI.

Real-World Applications

A critical data load failed overnight.

An analytics manager notices missing sales records in Snowflake. They prompt their agent: 'Show me the recent runs for the Stripe sync flow and tell me if any failed.' The agent uses `list_runs`, immediately showing that the run from 3 AM failed due to an API rate limit, solving the mystery instantly.

Setting up a brand new data source.

A data engineer needs to integrate a niche SaaS tool. They use `list_connectors` to see if Portable has a pre-built connector for it. If not, they check the documentation and then confirm connectivity by running a small test flow.

Troubleshooting destination writes.

An ops manager is setting up a new replica database. They first use `list_destinations` to see which targets are already configured (like BigQuery and Snowflake). This confirms the correct write permissions before deploying any new data flows.

Checking resource capacity.

A team wants to run a massive, multi-day ETL job. Before committing resources, they use `get_account` to check their current billing status and ensure the workspace has enough allocated execution time for the entire project.

Patterns to Avoid

Manually checking sync failure logs

✗ AVOID

A user opens the Portable dashboard, clicks on 'Stripe Sync,' finds the run history tab, filters by date, and then manually reads every error message to find the root cause.

✓ INSTEAD

Instead, simply ask your agent: 'What were the last three sync runs for Stripe?' The agent uses `list_runs` to summarize the success/failure status and point out the specific failure log immediately.

Guessing authorized data targets

✗ AVOID

A developer assumes that because they used BigQuery once, it's always a valid destination for new flows. They waste time configuring a flow only to find out later it failed due to permission errors.

✓ INSTEAD

Always call `list_destinations` first. This confirms exactly which data warehouses are currently authorized and ready to receive raw writes.

Bouncing between documentation and status

✗ AVOID

A user needs to know their account limits, so they go to the billing portal. Then they check the dashboard for flow details. They lose track of which number applies where.

✓ INSTEAD

Use `get_account` to get a single source of truth regarding resource usage and capacity right alongside your workflow questions.

The Right Fit

Use this MCP if your primary need is monitoring, auditing, or troubleshooting data movement between multiple SaaS platforms and data warehouses. You're asking: 'Did X move to Y successfully?' This tool manages the *movement* itself. Don't use it if you are trying to write complex SQL queries against the raw data; for that, you need a dedicated code execution MCP. Also, if your problem is optimizing the underlying transformation logic (the actual math or joins), this isn't enough. You need an MCP designed for database interaction. This tool focuses purely on orchestration and visibility into flow history and connection status.

The Context Switch Tax

Right now, checking data integrity means jumping between five different tools: the main ETL platform dashboard, the source SaaS provider's API page, the destination warehouse console (like Snowflake), and a dedicated documentation portal. You spend 80% of your time just clicking tabs and copy-pasting IDs to figure out if 'Stripe Orders' actually made it into BigQuery.

With this MCP, you keep everything in one chat window. You ask for the status of that sync run—the agent pulls the history, checks the row count, and tells you exactly where it stopped. It turns a 20-minute administrative chore into a single conversational exchange.

Portable.io: Centralized Data Visibility

You don't have to manually list flows, check connection status, or verify run outcomes in separate dashboards anymore. The agent pulls up the full configuration details using `get_flow` and gives you a single overview of what data is supposed to be moving.

It’s not just about viewing; it’s about instant diagnosis. When something breaks, your agent immediately shows you if the problem was the source connection (`list_connectors`) or the destination write permission (`list_destinations`).

Portable.io with 6 Tools

These tools give you full access to Portable's core functions: viewing configurations, checking run history, and monitoring connection status for all your data pipelines.

#	TOOL	DESCRIPTION
01	`list_flows`	Retrieves a list of all integration flows set up in Portable.
02	`get_flow`	Fetches the complete configuration details for one specific data flow.
03	`list_runs`	Shows historical execution records and results for a particular data flow.
04	`list_connectors`	Lists all available pre-built API connectors used as data sources.
05	`list_destinations`	Retrieves a list of authorized data warehouses and targets for receiving raw data writes.
06	`get_account`	Provides the exact billing details, workspace bounds, and account status limits.

See It in Action

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

U List all active ETL flows running in my Portable workspace.

I found 3 active flows. 'Stripe to Snowflake' runs every hour. 'Hubspot to BigQuery' runs daily. 'Shopify Orders' is paused. Would you like to view the recent execution runs for the Stripe sync?

U Show the recent runs for flow ID 4087 and tell me if any failed.

For flow 4087, the last 4 runs were successful, processing exactly 12,045 rows. However, the run on Tuesday at 4 AM failed with 'API Rate Limit Exceeded' from the source connector. Should I pull the destination configuration details?

U What destinations are currently configured to receive data?

You have 2 active destinations connected: 'Production Warehouse' (BigQuery) and 'Analytics Replica' (Snowflake). Connecting new flows will write directly into the authorized schemas on these platforms.

Frequently Asked Questions

01 **How does Portable.io MCP check data flow status?**

It checks by allowing your agent to use `list_runs` to retrieve historical execution records for a specific flow, showing if the last run succeeded and how many rows were processed.

02 **Can I see what destinations Portable.io MCP writes data to?**

Yes, you can call `list_destinations` to retrieve all configured data warehouses authorized to receive raw data from your active flows.

03 **What is the purpose of list_connectors in Portable.io?**

The `list_connectors` tool shows you every available, pre-built API source connector that can be used as a starting point for a data pipeline.

04 **Does Portable.io MCP help with billing limits?**

It does. You use the `get_account` tool to instantly retrieve your workspace bounds and current execution limits, ensuring you don't overspend or hit capacity ceilings.

05 **How do I check all my data pipelines with Portable.io MCP?**

You start by asking the agent to run `list_flows`, which retrieves a complete list of every integration flow configured in your account.

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 `"portableio": { "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

Portable.io 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 Portable.io. 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	Portable.io MCP
Server ID	`019d75f7-f670-7113-bf1f-7c89ad740c59`
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/portableio.