MCP SERVER | NO CODE | CLOUD HOSTED

# imgix (Real-time Image Processing) MCP

## Control the entire lifecycle of your media assets.

imgix MCP gives you full control over your image CDN infrastructure through natural conversation. You manage sources, purge assets, and monitor connections to ensure every picture loads fast and correctly across all devices.

A+ Quality Score 100/100

cdn | image-processing | asset-optimization | real-time-transformation | media-delivery | cache-purging

MAINTAINED BY VINKIUS CLOUD FOR AI AGENTS · JUNE 2026
vinkius.com/mcp/imgix-real-time-image-processing

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.

# imgix (Real-time Image Processing) MCP

10 tools available
Cloud-hosted on Vinkius

Managing media delivery used to mean jumping between dashboards: checking if the origin was connected, manually purging specific asset URLs from cache, or running reports on source health. Now, you connect your imgix account to any agent via Vinkius and manage your entire image pipeline using natural language commands. You can tell your AI client to list all your current sources, verify which ones are live, and even delete old ones with a single prompt. Need to fix an asset? Simply ask the agent to purge a specific file path across the global network. This capability lets you treat complex infrastructure management like talking to a teammate: direct, immediate, and actionable. You don't need to learn Imgix API calls; your agent handles the complexity.

## Core Capabilities

**01 — Manage Image Sources**

You can list, create, update, or delete connections between your original storage (like S3) and the imgix CDN.

**02 — Invalidate Cache Assets**

Send a command to purge specific files across the global network, forcing all viewers to download the newest version from your source.

**03 — Audit Asset Libraries**

Retrieve comprehensive lists of assets within any connected source, including file paths and sizes.

**04 — Monitor Source Health**

Get detailed status checks on your sources, confirming if they are active or need maintenance.

**05 — Control Deployment State**

Enable or disable entire image sources instantly to manage traffic flow during updates or downtime.

## One Click on Vinkius — From Prompt to Execution

Available at **vinkius.com/mcp/imgix-real-time-image-processing** — connect your AI agent in three steps.

01 Subscribe to this MCP and provide your imgix API Key.

02 Connect the credential set to your preferred AI client (Claude, Cursor, etc.).

03 Tell your agent exactly what you need—for example, 'Purge all assets related to our holiday campaign'—and execute commands through conversation.

The bottom line is that you talk to the MCP like talking to a DevOps teammate; it handles the complex API calls for image delivery management.

## Built For

Frontend developers who dread manual cache clearing. SREs and DevOps engineers tired of clicking through multiple dashboards just to verify source connections or deploy updates. Digital Asset Managers needing a single pane of glass view for their entire image library.

**Frontend Developer**

You use this MCP to verify origin connections and trigger cache purges directly from your IDE after updating any media asset.

**SRE / DevOps Engineer**

You monitor source availability, manage CDN deployment states, and run audits on all configured sources through natural conversation.

**Digital Asset Manager**

You audit entire asset libraries and organize origin storage connections across multiple projects without leaving your workflow.

## What Changes When You Connect

01 Fixing stale images is simple. Instead of guessing which cache layer needs clearing, just ask the agent to run `purge` on a specific file path and invalidate it globally across the imgix Edge network.

02 You can build complex pipelines without leaving your IDE. Use this MCP within Cursor or VS Code to verify origin connections or manage CDN deployment states with simple conversation prompts.

03 Never lose track of what assets belong where. Running `list_assets` gives you an instant inventory check, listing paths and sizes for any source so you know exactly what's available.

04 When a new product line launches, simply tell the agent to use `create_source`. It handles connecting your raw storage (S3/GCS) directly into the CDN pipeline.

05 Manage traffic flow with confidence. You can instantly disable an entire source using `disable_source` if you need maintenance time, or enable it again when ready.

## Real-World Applications

**A hero image is showing a cached version after I updated the original file.**

The developer asks their agent: 'Purge this specific banner image from the CDN.' The agent executes the `purge` tool, invalidating the asset globally and ensuring the viewer gets the latest version immediately.

**We need to verify if our new user upload folder is correctly connected to imgix.**

The SRE asks their agent: 'List my sources and get details for the user uploads connection.' The agent runs `list_sources` followed by `get_source`, confirming deployment status before traffic hits it.

**Before deleting a source, I need to know what files are stored there.**

The DAM asks their agent: 'List all assets in the marketing-assets source.' The agent runs `list_assets` and provides an immediate inventory count and sample file paths for audit purposes.

**We need to temporarily take a product line off sale without deleting its entire connection.**

The developer asks their agent: 'Disable the e-commerce sources.' The agent calls `disable_source`, instantly stopping traffic flow and preventing accidental image serving until it's re-enabled.

# Patterns to Avoid

### Managing media assets outside of imgix

✗ AVOID

Manually checking if a file was uploaded to S3, then logging into the CDN dashboard separately to see if it processed correctly.

✓ INSTEAD

Use this MCP. Ask your agent to `list_assets` on the source that connects to your storage. This verifies both the existence and the readiness of the asset in one conversation.

### Trying to purge everything manually

✗ AVOID

Copying dozens of image URLs into a cache invalidation tool, hoping you didn't miss any paths.

✓ INSTEAD

Instead, ask your agent to `list_assets` first. Then, send the specific file path back and run the `purge` command for comprehensive removal.

### Assuming a source is active

✗ AVOID

Deploying code that relies on an image connection without verifying if the source was enabled.

✓ INSTEAD

Always use `get_source` to check the deployment status before relying on any source. This confirms it's live and ready for traffic.

## The Right Fit

Use this MCP if your core problem involves controlling, monitoring, or invalidating images that are delivered via a Content Delivery Network (CDN). Specifically, you need to manage the connection between an origin storage (like S3) and the public-facing image pipeline. You'll use it when you must purge cached files, check source health, or create new connections.

Don't use this if your only goal is basic file storage management—for example, simply moving a raw JPG from one bucket to another. For those tasks, simple cloud storage tools are enough. This MCP deals with the *delivery and transformation* of files; it assumes the file already exists in an origin you manage.

## Dealing with stale images is a nightmare of clicks.

You update the original hero image on your server. You hit save, but when QA pulls up the staging site, it shows the old picture. You spend fifteen minutes logging into multiple dashboards, guessing which cache layer needs clearing, and copy-pasting URLs to purge assets that might not even exist.

With this MCP, you simply tell your agent: 'Purge the main hero banner image.' The agent executes the command behind the scenes. Done. The stale content is instantly invalidated across the entire network, letting you focus on code instead of cache headers.

## Get total visibility with the list_assets tool.

Before deleting an old source connection or updating its settings, you usually have to manually run a series of reports and check file lists. This is slow and prone to missing important metadata about which files are actually dependent on that source.

Now, running `list_assets` gives you an instant inventory report directly in the chat window. You see file paths, sizes, and content types all at once. It’s immediate, definitive knowledge of your media assets.

# Imgix (Real-time Image Processing) MCP: 10 Tools

Use these tools to manage every aspect of your image CDN setup, from connecting new storage origins to purging stale assets.

| # | TOOL | DESCRIPTION |
|---|---|---|
| 01 | list_sources | Lists all connected imgix sources, providing a quick overview of your current CDN setup. |
| 02 | get_source | Retrieves specific details about one imgix source, including its domain and deployment type. |
| 03 | create_source | Connects a new origin storage location (like S3 or GCS) to the imgix CDN for processing. |
| 04 | update_source | Modifies settings on an existing source, such as its caching rules or deployment type. |
| 05 | enable_source | Activates a configured imgix source, making it live and ready to serve images across the network. |
| 06 | disable_source | Takes an imgix source offline immediately, stopping all image serving from that connection. |
| 07 | purge | Removes a specific file and all its processed versions from the global imgix cache, forcing immediate re-fetch. |
| 08 | list_assets | Returns a list of files within an imgix source, showing their paths, sizes, and content types. |
| 09 | get_asset | Retrieves the full metadata for one specific asset path in an imgix source. |
| 10 | delete_source | Permanently removes a configured imgix source from your account. |

# See It in Action

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

**U** List all my current imgix sources

I've found 3 imgix sources: 'marketing-assets' (ID: src_123), 'e-commerce-products' (ID: src_456), and 'user-uploads' (ID: src_789). Which one would you like to inspect?

**U** Purge this image from CDN: https://mycompany.imgix.net/logos/v2.png

Purge request sent. I've invalidated 'logos/v2.png' across the global imgix Edge network. All processed versions of this file will now be re-fetched and transformed from your origin.

**U** List the assets in the 'marketing-assets' source

I've retrieved the assets for source 'src_123'. Recent files include '/banners/hero-winter.jpg', '/icons/social-bundle.svg', and '/product-shots/model-01.webp'. Total assets in this source: 1,245.

# Frequently Asked Questions

01 **How do I check if a specific image source is active using imgix MCP?**

Use the `get_source` tool to retrieve details about that connection. This returns its name, domain, and current deployment status, confirming if it's live or disabled.

02 **What should I use if I need to remove a file from the cache?**

You must run the `purge` tool, providing the exact asset path. This action invalidates all processed derivatives of that file across the global network immediately.

03 **Can imgix MCP help me audit my current image sources?**

Yes, start by using `list_sources`. This gives you a comprehensive list of every connection currently configured on your account in one go.

04 **Do I need to use the create_source tool for every new image upload?**

No. You only use `create_source` when you are connecting an entirely new storage location (like a second GCS bucket) to the CDN pipeline.

05 **What's the difference between list_assets and get_asset in imgix MCP?**

`list_assets` shows you everything available in a source. `get_asset` lets you drill down and retrieve detailed metadata for one specific file path.

# 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 `"imgix-real-time-image-processing": { "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

# imgix (Real-time Image Processing) 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 imgix (Real-time Image Processing). 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 | imgix (Real-time Image Processing) MCP |
| Server ID | `019d75b8-6c9e-711a-956a-4daa8d847e68` |
| 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/imgix-real-time-image-processing.