MCP SERVER | NO CODE | CLOUD HOSTED

# Lokalise MCP

## Manage keys and translations from your chat window.

Lokalise takes over your translation workflow. Connect this MCP to manage entire localization pipelines—from creating new keys to downloading finished files—all through natural conversation with your AI agent.

A+ Quality Score 100/100

localization | translation-management | i18n | workflow-automation | key-management | software-localization

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

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.

# Lokalise MCP

13 tools available
Cloud-hosted on Vinkius

Need to handle a massive, multi-language launch? This connector lets you take full control of your product's localization process without leaving your coding environment. You talk to your AI agent, and it handles the API calls to Lokalise. Want to see what projects exist or get details on a specific one? Just ask. Need to add fifty new strings across ten languages? It manages that complex data exchange for you. This means no more copying massive CSVs into separate web tabs. You can list all your translation keys and then tell the agent, 'Update this key with these new values.' The whole process is managed through natural conversation, making it feel like having a dedicated localization coordinator sitting right next to you. All of this powerful functionality is available through Vinkius, connecting Lokalise directly to your preferred AI client.

## Core Capabilities

**01 — Manage Project Structure**

List all existing translation projects and create new ones with a single command.

**02 — Control Translation Keys**

Create, update, or list specific text keys across your entire project setup.

**03 — Handle Translations Data**

Fetch translations for a key, add new language versions, and review existing content flags.

**04 — Process Files In and Out**

Upload localization files (like JSON or YAML) to the project or generate structured download bundles in any format.

**05 — Audit Team and Orders**

List team members' roles or check the status of professional translation orders.

## One Click on Vinkius — From Prompt to Execution

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

01 Subscribe to this MCP on Vinkius and provide your Lokalise API token.

02 Tell your AI agent what you need. For instance, 'List all active projects.'

03 Your agent executes the necessary calls and returns structured data—like a list of project names or available translation keys—directly into your conversation.

The bottom line is that you guide the whole localization process using natural language, letting the MCP handle all the complex backend API interactions.

## Built For

This tool is for product owners and developers who are tired of manually juggling spreadsheets, web consoles, and codebases just to get a simple string updated. It's built for people whose daily job involves coordinating content across multiple languages.

**Developer**

You use it to instantly create translation keys or push new localized files directly from your IDE without switching context.

**Product Manager**

You check translation progress across different languages and monitor key counts using the agent, getting an immediate status report in chat.

**Localization Specialist**

You automate bulk uploads of files or generate export bundles for QA teams by simply describing the required format and language set.

## What Changes When You Connect

01 Stop switching between tabs. You can list projects or get details using the `get_project` tool, all within the conversation flow.

02 Keep developers in their IDE. Use `create_key` to instantly add new strings that need translation without leaving your code editor.

03 Automate content gathering. Upload localization files with `upload_file`, allowing you to process large volumes of data programmatically.

04 Track team progress easily. You can use `list_team_members` or check professional orders via `list_orders` to see who's working on what.

05 Simplify bulk exports. Generating a download bundle with the `download_file` tool means you get clean, structured files ready for QA.

## Real-World Applications

**Need to update fifty keys across three languages immediately.**

The Product Manager asks the agent: 'Update the key 'login.button' with these new values.' The agent uses `add_translation` and `update_key`, ensuring all required language strings are changed correctly, which used to take hours of manual entry.

**Launching a new product line requires dozens of new phrases.**

The developer asks the agent to 'Create ten new keys for checkout flow.' The agent uses `create_key` repeatedly, populating all necessary strings in the project instantly so translators can start working.

**Need a full export of all content changes from last month.**

The localization team asks the agent to 'Download everything for Portuguese and German.' The agent uses `download_file`, giving them one clean, zip-ready bundle rather than multiple manual downloads.

**Checking if a new feature requires keys in every language.**

The PM asks the agent to 'List all translations for this key.' The agent uses `list_translations` and checks the status across all configured languages, giving an immediate compliance report.

# Patterns to Avoid

### Manually checking project scope.

✗ AVOID

Opening the Lokalise web console, navigating to the project dashboard, and clicking through multiple tabs just to count keys or check status reports.

✓ INSTEAD

Just ask your agent: 'List all projects' using `list_projects`, then follow up with 'Get details for [Project Name]' using `get_project`. It keeps you in one chat window.

### Updating keys via code snippets.

✗ AVOID

Hardcoding translation values directly into your application source files, which means every future change requires a full code push and manual review cycle.

✓ INSTEAD

Instead, use the agent to `create_key` or `update_key`. This keeps the source of truth centralized in Lokalise, making it easy for the team to manage.

### Mixing up file formats.

✗ AVOID

Forgetting whether you need JSON, YAML, or XLIFF when uploading content and having to restart the entire import process.

✓ INSTEAD

Use `upload_file` with clear instructions about the format. If you're done, use `download_file`—the agent handles the required output structure.

## The Right Fit

You should use this MCP if your primary bottleneck is context switching. Specifically, if you find yourself moving between a code editor, a spreadsheet, and a dedicated web console just to manage text strings or localization assets. Use it to list projects, get key details, or update translations programmatically.

You don't need this if you are only performing ad-hoc tasks like checking one single string manually. For those simple checks, the native Lokalise UI is fine. But if your workflow involves bulk actions —like running `upload_file`, generating comprehensive bundles with `download_file`, or managing dozens of keys using `create_key` and `update_key` —this MCP saves you time and sanity.

## Localization workflows are a mess of tabs and copy-paste.

Today, updating translations for a global product means jumping between tools: the code editor to grab a key; a spreadsheet to manage language variants; and finally, the web console to execute the changes. This process is slow, error-prone, and requires constant context switching that grinds development momentum to a halt.

With this MCP, you talk directly to your AI agent. You ask it to handle the update. It executes the necessary calls—using tools like `update_key` or `add_translation` —and returns confirmation. The whole process stays in one conversation thread. You just get results.

## Lokalise gives you control over your entire content pipeline.

Manual steps that vanish include navigating to the project list, fetching metadata using `get_project`, and manually checking team member roles via `list_team_members`. These administrative checks used to be a chore of clicks and dashboard reviews.

Now, you just ask. The agent retrieves the data for you. You aren't limited by the UI structure; you're limited only by how well you can articulate your needs.

# Lokalise: 13 Localization Tools

These tools let you control every aspect of translation management—from listing basic projects to generating final download bundles.

| # | TOOL | DESCRIPTION |
|---|---|---|
| 01 | `add_translation` | Adds specified translations to one or more existing keys within a project. |
| 02 | `create_key` | Creates entirely new translation keys inside your Lokalise project setup. |
| 03 | `create_project` | Establishes a brand-new, dedicated localization project in the system. |
| 04 | `download_file` | Generates and prepares a download bundle of translations for you to retrieve. |
| 05 | `get_project` | Retrieves detailed metadata about a specific, existing Lokalise project. |
| 06 | `list_keys` | Fetches a list of all translation keys available within a given project scope. |
| 07 | `list_languages` | Shows which languages are currently configured for your Lokalise project. |
| 08 | `list_orders` | Retrieves a list of professional translation orders associated with your account. |
| 09 | `list_projects` | Lists all the localization projects that currently exist under your Lokalise account. |
| 10 | `list_team_members` | Retrieves a roster of team members and their roles for administrative review. |
| 11 | `list_translations` | Shows all recorded translations for a single key within the project. |
| 12 | `update_key` | Modifies existing translation keys to change their content or metadata. |
| 13 | `upload_file` | Uploads external localization files, such as JSON or XLIFF, into the project for processing. |

# 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 Lokalise translation projects and show their current status.

I've retrieved your Lokalise projects. You have 3 active projects: 'Web App' (98% localized, 5 languages), 'Mobile App' (84% localized, 4 languages), and 'Marketing Site' (100% localized, 8 languages).

**U** Create a new translation key 'checkout.success.message' in my Web App project for the web platform.

Successfully created the key `checkout.success.message` in your Web App project for the web platform. It's now ready for translators to fill in across all configured languages.

**U** Download all Portuguese (pt-BR) translations from my Mobile App project in JSON format.

I've generated the download bundle for Portuguese (pt-BR) translations in JSON format. The file contains 342 translated strings with 98% completion. You can download it from the provided link.

# Frequently Asked Questions

01 **How do I list all my projects using Lokalise MCP?**

You simply ask the agent to 'List all projects.' The agent uses `list_projects` and returns a comprehensive list of every project under your account, letting you see what's ready for work.

02 **Can I bulk upload files with Lokalise MCP?**

Yes. You use the `upload_file` tool to send localization files like JSON or XLIFF directly into your project queue, handling large data sets in one go.

03 **What if I need to change a key's value?**

You tell the agent exactly which key and what the new text should be. It then uses the `update_key` tool to modify that string across all relevant languages.

04 **How do I get translated content for a specific feature?**

Use the agent to list translations via `list_translations`. This shows you every language and its current status for a single, identified key.

05 **Is Lokalise MCP useful for developers?**

Absolutely. Developers love it because they can use the agent to run commands like `create_key` directly from their IDE chat window instead of needing to open a web browser tab.

# 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 `"lokalise": { "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

# Lokalise 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 Lokalise. 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 | Lokalise MCP |
| Server ID | `019d75c9-ed31-739e-ae5d-834da7dad862` |
| 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/lokalise.