

MCP(Model Context Protocol)

release date. 2026. 09. 11.

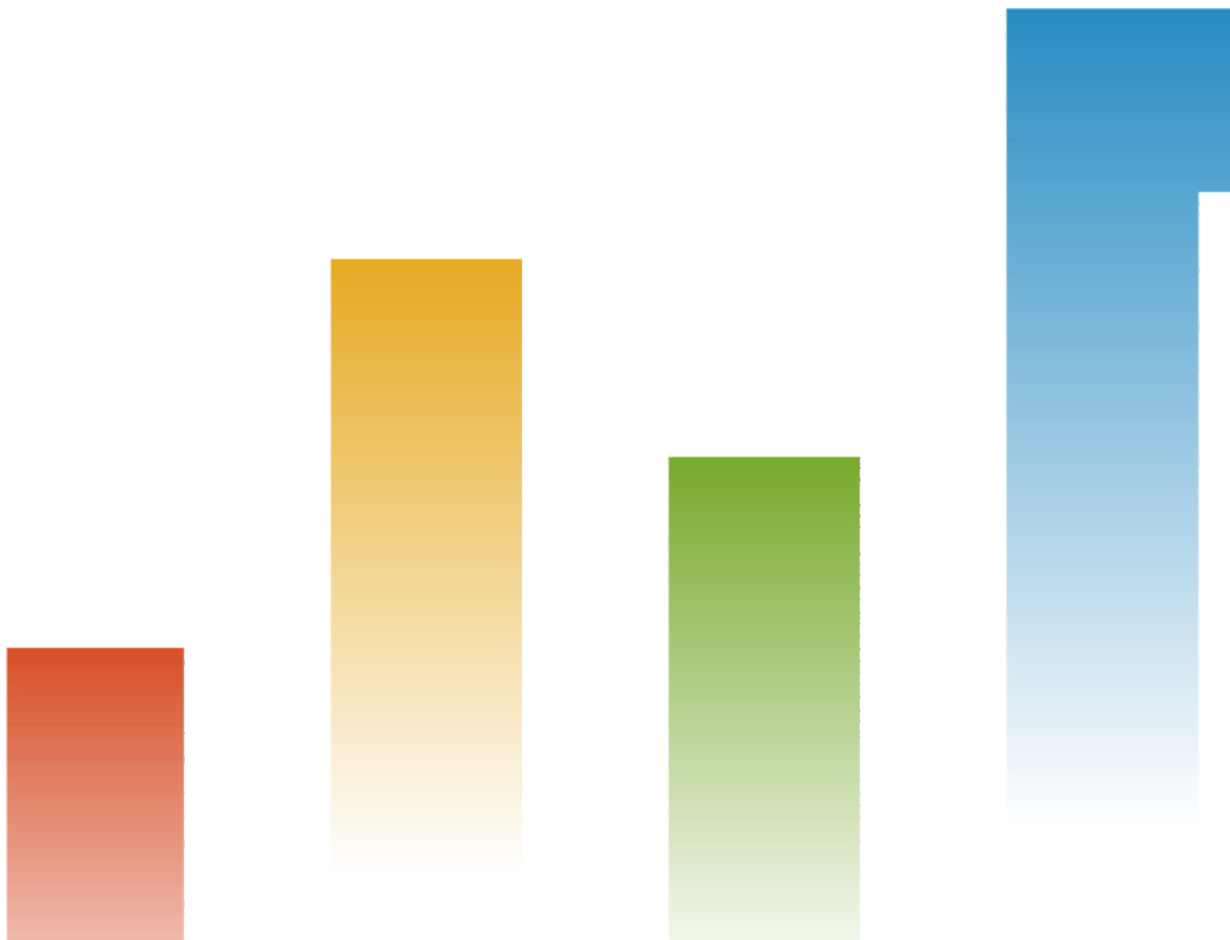


Table Of Contents

• MCP	3
◦ Getting Started.....	4
◦ Configuration	8
■ Environment Configuration	14
■ Uninstall	17
◦ Using WhaTap MCP	19
■ Examples	23
◦ Troubleshooting.....	28

MCP(Model Context Protocol)

WhaTap MCP is a server that integrates the WhaTap API via MCP (Model Context Protocol), the standard protocol for AI assistants to call external tools. Users can instantly query monitoring data such as server CPU, APM TPS, and K8s Pod status with a single line of natural language.

- **Tools:** 10 (project management 3, data exploration 3, composite analysis 2, installation & PromQL 2)
- **Supported domains:** Server, APM, Kubernetes, Database, Log
- **Current version:** v1.2.1
- **Open source:** github.com/whatap/whatap-open-mcp

 Run `npx -y github:whatap/whatap-open-mcp --version` to check the installed version.

Key Features

- Query monitoring data with natural language questions — no need to navigate complex dashboards.
- Access data across multiple WhaTap monitoring products, including server, APM, and Kubernetes.
- Seamlessly integrate monitoring context into your existing AI workflows.

Supported AI Clients

Client	Provider	Configuration
Claude Desktop	Anthropic	JSON config file
Claude Code	Anthropic	Single CLI command
Codex CLI	OpenAI	CLI command or TOML config
Gemini CLI	Google	CLI command or JSON config

Getting Started

This MCP server lets you query WhaTap monitoring data using natural language from AI assistants such as Claude, Codex, and Gemini. Get started with WhaTap MCP by following the steps below.

- Step 1. Prerequisites
- Step 2. Sign up
- Step 3. Issue API token
- Step 4. Configure AI client

Prerequisites

Prepare the following before using the WhaTap MCP server.

Item	Details
Node.js	18.0.0 or higher (includes <code>npm</code> and <code>npx</code>)
Git	Required on Windows only
WhaTap account	Free or paid account at whatap.io
AI client	One of: Claude Desktop, Claude Code, Codex CLI, Gemini CLI

Verify Node.js installation

Run the following commands in your terminal. If the version is too low or Node.js is not installed, refer to the installation guide below.

```
node --version # v18.x.x or higher
npx --version  # any version output is OK
```

Install Node.js

Install Node.js using the method that matches your operating system.

macOS

```
brew install node
```

ⓘ If Homebrew is not installed, install it from <https://brew.sh> first, or download Node.js directly from <https://nodejs.org>.

Windows

```
winget install OpenJS.NodeJS.LTS
```

ⓘ Alternatively, download the LTS installer from <https://nodejs.org>. After installation, **open a new terminal** for the changes to take effect.

Ubuntu / Debian

```
curl -fsSL https://deb.nodesource.com/setup_22.x | sudo -E bash -  
sudo apt install -y nodejs
```

ⓘ Running `apt install nodejs` installs an old version (v12–v16). Always use NodeSource.

Fedora / RHEL / Amazon Linux

```
sudo dnf install -y nodejs npm
```

Install via nvm (all OS)

```
curl -o- https://raw.githubusercontent.com/nvm-sh/nvm/v0.40.3/install.sh | bash
source ~/.bashrc
nvm install --lts
```

Install Git

The WhaTap MCP server references a GitHub repository directly, which requires Git. Git comes preinstalled on macOS and Linux, so install it on Windows only.

```
winget install --id Git.Git -e --source winget
```

❗ If you do not use Windows Package Manager (winget), download the installer from <https://git-scm.com>. After installation, **open a new terminal** for the changes to take effect.

After installation, run the following command in the terminal.

```
git --version # Check that a version is printed
```

Sign up

You need to [sign up](#) to use the WhaTap monitoring service. For more details, refer to the [Account Management > Sign up](#) document.

Issue API token

Issue an API token for the WhaTap MCP server to use when querying data.

1. Log in to [WhaTap Console](#).
2. Click the **profile icon** in the top right corner.

3. Select **Account Management**.
4. Copy the token from the **API Token** section.

ⓘ This is an **account-level token**. It can **read** data from all projects under this account. It cannot modify or delete data, so it is safe to use. However, keep the token secure and do not expose it externally.

Configuration

Follow the setup instructions for your AI client. All clients use the same MCP server — only the configuration format differs.

!

All clients reference the GitHub repository directly: `npx -y github:whatap/whatap-open-mcp`

Referencing the repository requires Git, so install it first on Windows. For more details, refer to the [Getting Started > Install Git](#) document.

AI Client	Format	Config file	Command
Claude Desktop	JSON	<code>claude_desktop_config.json</code>	Edit file directly
Claude Code	-	-	<code>claude mcp add whatap -e WHATAP_API_TOKEN=... -- npx -y github:whatap/whatap-open-mcp</code>
Codex CLI	TOML	<code>~/.codex/config.toml</code>	<code>codex mcp add whatap --env ... -- npx -y github:whatap/whatap-open-mcp</code>
Gemini CLI	JSON	<code>~/.gemini/settings.json</code>	<code>gemini mcp -s user add whatap -e ... -- npx -y github:whatap/whatap-open-mcp</code>

Claude Desktop

Add the WhaTap MCP server to your configuration file.

OS	Path
macOS	<code>~/Library/Application Support/Claude/claude_desktop_config.json</code>
Linux	<code>~/.config/Claude/claude_desktop_config.json</code>

OS	Path
Windows	%APPDATA%\Claude\claude_desktop_config.json

Add configuration

Open the config file and add the following content.

Claude Desktop configuration

```
{
  "mcpServers": {
    "whatap": {
      "command": "npx",
      "args": ["-y", "github:whatap/whatap-open-mcp"],
      "env": {
        "WHATAP_API_TOKEN": "your_token_here"
      }
    }
  }
}
```

⚠ If other MCP servers are already configured, add only the `"whatap": { ... }` block inside the existing `"mcpServers"` object.

Verify registration

Save the config file and restart Claude Desktop to connect the Whatap MCP server.

⚠ Claude Desktop must be **fully quit** (not just the window closed) and restarted for the configuration to take effect.

Claude Code (CLI)

Register the WhaTap MCP server using a terminal command.

Choose scope

Decide the scope before registering.

Scope	Option	Description
All projects	<code>-s user</code>	Available across all projects in your account
Current project	<code>-s project</code>	Available only in the current project (saved to <code>.mcp.json</code>)

Register via command

Claude Code (CLI) registration

```
# All projects (user scope)
claude mcp add whatap -s user \
-e WHATAP_API_TOKEN=your_token_here \
-- npx -y github:whatap/whatap-open-mcp

# Current project (project scope)
claude mcp add whatap -s project \
-e WHATAP_API_TOKEN=your_token_here \
-- npx -y github:whatap/whatap-open-mcp
```

Verify registration

Claude Code (CLI) verification

```
claude mcp list
```

Codex CLI (OpenAI)

Register the WhaTap MCP server via command or config file.

Register

Choose one of the two methods below.

- [Register via command](#) **Register by editing config file**

Codex CLI (OpenAI) command registration

```
codex mcp add whatap \  
--env WHATAP_API_TOKEN=your_token_here \  
-- npx -y github:whatap/whatap-open-mcp
```

~/codex/config.toml

```
[mcp_servers.whatap]  
command = "npx"  
args = ["-y", "github:whatap/whatap-open-mcp"]  
  
[mcp_servers.whatap.env]  
WHATAP_API_TOKEN = "your_token_here"
```

ⓘ Codex uses TOML format, not JSON.

Verify registration

Codex CLI (OpenAI) verification

```
codex mcp list
```

Gemini CLI (Google)

Register the WhaTap MCP server via command or config file.

Choose scope

Decide the scope before registering.

Scope	File location	Purpose
User (global)	<code>~/.gemini/settings.json</code>	Personal settings, all projects
Project	<code>.gemini/settings.json</code> (project root)	Shared with team

Register

Choose one of the two methods below.

- [Register via command](#) Register by editing config file

Gemini CLI (Google) registration

```
gemini mcp -s user add whatap \
-e WHATAP_API_TOKEN=your_token_here \
npx -- -y github:whatap/whatap-open-mcp
```

`~/.gemini/settings.json`

```
{
  "mcpServers": {
    "whatap": {
      "command": "npx",
      "args": ["-y", "github:whatap/whatap-open-mcp"],
      "env": {
        "WHATAP_API_TOKEN": "your_token_here"
      }
    }
  }
}
```

```
}  
}  
}
```

Verify registration

Gemini CLI (Google) verification

```
gemini mcp list
```

Verify connection

1. After completing the setup, open your AI client.
2. Enter the following:

```
Show me my WhaTap project list
```

3. If a list with project codes (pcode), names, platforms, and product types appears, the connection is successful. If nothing appears, refer to the troubleshooting guide.

Environment Configuration

The WhaTap MCP server works out of the box in SaaS environments with just `WHATAP_API_TOKEN` . If you are using an on-premises environment, you must also set `WHATAP_API_URL` .

Environment variables

Variable	Required	Default	Description
<code>WHATAP_API_TOKEN</code>	O	-	WhaTap account API token
<code>WHATAP_API_URL</code>	X	<code>https://api.whatap.io</code>	API endpoint (no change needed for SaaS)

API endpoint

In the WhaTap SaaS environment, all regions (Seoul, Tokyo, Singapore, Frankfurt, etc.) use the same API URL (`https://api.whatap.io`). Region-based routing is handled automatically on the server side — **no additional configuration is required**.

Refer to the table below and update `WHATAP_API_URL` accordingly:

Environment	URL	Notes
SaaS (default)	<code>https://api.whatap.io</code>	No change needed
On-premises	Customer internal URL	Set <code>WHATAP_API_URL</code>

On-premises configuration example (MCP server IP: 192.168.1.100)

Claude Desktop

Claude Desktop

```
{
  "mcpServers": {
    "whatap": {
      "command": "npx",
      "args": ["-y", "github:whatap/whatap-open-mcp"],
      "env": {
        "WHATAP_API_TOKEN": "your_token_here",
        "WHATAP_API_URL": "http://192.168.1.100"
      }
    }
  }
}
```

Claude Code

Claude Code

```
claude mcp add whatap \
-e WHATAP_API_TOKEN=your_token_here \
-e WHATAP_API_URL=http://192.168.1.100 \
-- npx -y github:whatap/whatap-open-mcp
```

Codex CLI

Codex CLI

```
[mcp_servers.whatap]
command = "npx"
args = ["-y", "github:whatap/whatap-open-mcp"]

[mcp_servers.whatap.env]
WHATAP_API_TOKEN = "your_token_here"
```

```
WHATAP_API_URL = "http://192.168.1.100"
```

Gemini CLI

Gemini CLI

```
gemini mcp -s user add whatap \
-e WHATAP_API_TOKEN=your_token_here \
-e WHATAP_API_URL=http://192.168.1.100 \
npx -- -y github:whatap/whatap-open-mcp
```

WhaTap service regions

The region is selected when creating a project in WhaTap Console. The API URL is the same regardless of region.

Region	Location	Cloud
Seoul	South Korea	AWS, Azure, Kakao Cloud
Tokyo	Japan	AWS, Azure
Singapore	Singapore	AWS
Jakarta	Indonesia	AWS
Mumbai	India	AWS
California	US West	AWS
Virginia	US East	AWS
Frankfurt	Germany	AWS

Uninstall

If you no longer need the WhaTap MCP server or want to reset the configuration, remove the MCP server from your AI client.

How to remove

Instructions for removing the WhaTap MCP server from major AI clients including Claude, Codex, and Gemini.

Claude Desktop

1. Delete the "whatap" block from `claude_desktop_config.json` .
2. Restart Claude Desktop.

Claude Code

```
claude mcp remove whatap
```

Codex CLI

Delete the `[mcp_servers.whatap]` section from `~/.codex/config.toml` .

Gemini CLI

```
gemini mcp remove whatap
```

Or delete the "whatap" block from `~/.gemini/settings.json` .

Clear cache (optional)

Clearing the npx cache ensures the latest version is downloaded on next run.

```
npx clear-npx-cache
```

How To Use

This guide explains how to query monitoring data using an AI assistant after connecting to the WhaTap MCP server.

How it works

When you ask the AI assistant a question in natural language, it automatically handles the following steps.

- 1. **Retrieve project list** — Identify the project code (pcode).
- 2. **Explore data catalog** — Find what types of data are available.
- 3. **Execute query** — Retrieve the requested data.

 Start with "Show me my project list" to confirm your project code. Using the code in follow-up questions will produce more accurate results.

Available tools

The MCP server provides 10 tools to the AI assistant. You do not need to call tools directly — just ask in natural language and the AI will automatically select the appropriate tool.

Project management tools

Tool	Function
<code>whatap_list_projects</code>	List all monitoring projects (pcode, name, platform)
<code>whatap_project_info</code>	Retrieve detailed information for a specific project
<code>whatap_list_agents</code>	List agents (servers/instances) within a project

Data exploration tools

Tool	Function
<code>whatap_data_availability</code>	Browse 900+ MXQL queries across 20+ categories, detect live data, list OpenMetrics metrics and saved PromQL queries
<code>whatap_describe_query</code>	Retrieve details for an MXQL path or OpenMetrics metric (parameters, output fields, metric type, recommended PromQL)
<code>whatap_query_data</code>	Execute MXQL, PromQL, or saved queries (results include unit conversion, thresholds, and analysis guidance)

Composite analysis tools

Tool	Function
<code>whatap_apm_anomaly</code>	APM anomaly detection — simultaneously runs 4 queries (TPS, response time, errors, Active TX) for statistical analysis
<code>whatap_service_topology</code>	Detect service connections and network bottlenecks (requires NPM agent)

Installation and PromQL tools

Tool	Function
<code>whatap_install_agent</code>	Generate agent installation commands for 29 platforms (auto-includes connection info, auto-detects platform)
<code>whatap_create_promql</code>	Create, validate, and save PromQL queries (reusable)

Supported platforms for whatap_install_agent

Category	Platforms
Infrastructure OS	Debian/Ubuntu, Amazon Linux, RHEL/CentOS, SUSE, FreeBSD, Windows, and 3 more (9 total)
APM	Java, Node.js, Python, PHP, .NET, Go
Database	PostgreSQL, Oracle, MySQL, Redis, MongoDB, DB2, and 6 more (12 total)
Container	Kubernetes (Helm)
Server apps	Kafka, NGINX, Apache, and 4 more (7 total)

Tool usage flow

Data query workflow (MXQL)

```

whatap_list_projects → Confirm project code
↓
whatap_data_availability(projectCode) → Check available MXQL paths
↓
whatap_describe_query(path) → Review parameters, fields, MXQL content
↓
whatap_query_data(projectCode, path) → Retrieve data

```

PromQL workflow (Kubernetes/OpenMetrics)

```

whatap_data_availability(projectCode) → Check OpenMetrics metric list
↓
whatap_create_promql(projectCode, name, query) → Validate and save PromQL
↓
whatap_query_data(projectCode, savedQuery="name") → Reuse saved query

```

Agent installation workflow

```

whatap_list_projects → Confirm projectCode + platform

```

↓

whatap_install_agent(projectCode) → Output connection info + installation commands

↓

whatap_list_agents(projectCode) → Verify agent registration

✓ This flow is handled automatically by the AI. Just ask your question in natural language.

MXQL path reference

Commonly used paths you can query directly without calling `whatap_data_availability` first.

Domain	Path	Description
Server	<code>v2/sys/server_base</code>	CPU, memory, basic metrics
Server	<code>v2/sys/server_disk</code>	Disk usage
Server	<code>v2/sys/server_network</code>	Network I/O
APM	<code>v2/app/tps_pcode</code>	TPS (project-wide)
APM	<code>v2/app/resp_time_pcode</code>	Response time (project-wide)
APM	<code>v2/app/tx_error_pcode</code>	Error count (project-wide)
K8s	<code>v2/container/kube_pod</code>	Pod status
K8s	<code>v2/container/kube_node</code>	Node status
K8s	<code>v2/container/kube_event</code>	Cluster events
DB	<code>v2/db/instance_active_session</code>	Active sessions

Examples

Explore how to use the WhaTap MCP server through natural language query examples and real conversation flows.

Frequently used queries

Situation	Example query
Getting started	"List all my WhaTap projects"
Check server status	"Show CPU usage for project 12345 over the last 5 minutes"
Check APM anomalies	"Detect APM anomalies with high sensitivity"
Incident response	"Summarize error status for the last 5 minutes"
Data exploration	"What data can I query in this project?"
K8s check	"Show pod status for project 33194"
Service relationships	"Show service topology"
OpenMetrics exploration	"Show available OpenMetrics metrics for project 33194"
PromQL creation	"Create a PromQL query for CPU usage by Pod"
Run saved query	"Run the saved 'CPU by Pod' query"
Agent installation	"Install the server monitoring agent for project 12345"

Natural language query examples

Sample queries for common monitoring tasks. Choose or adapt the queries that match your needs.

Project exploration

List all my WhaTap projects
Show me the details of project 12345
What agents are in project 12345?

Server monitoring

Show me server CPU usage for the last hour
Are there any servers with memory usage over 80%?
Check disk usage
Show me the top 5 servers by CPU usage

APM (Application Performance)

Show me TPS trends for the last 5 minutes
What is the average response time?
Have any transaction errors occurred?
Check the status of currently active transactions

Kubernetes

Check Pod status
Show CPU and memory usage by node
Are there any Warning events in recent cluster events?

Database

Show me the DB instance list
Check current active sessions

Anomaly detection

Find agents with anomalies in the last 5 minutes
Run anomaly detection with high sensitivity for the past hour

Service topology

Show service connections
Are there any network bottlenecks?

OpenMetrics / PromQL

Show available OpenMetrics metrics for project 33194
Create a PromQL query for CPU usage by Pod
Run the saved 'CPU by Pod' query

Agent installation

How do I install a monitoring agent for project 12345?
Install the Java APM agent
Show me how to set up the Kubernetes agent
Install PostgreSQL DB monitoring
Install the infrastructure agent on a Debian server

Composite queries

Do a morning health check.

Check server status across all projects,
and analyze whether there are any anomalies in APM projects.

Conversation examples

The conversations below are examples. Actual responses will vary depending on your project's monitoring data.

Example 1. Morning server check

User: Do a morning check. Summarize server CPU and memory status.

AI: I checked 3 servers.

- web-01: CPU 45%, Memory 62% (Normal)
- web-02: CPU 87%, Memory 71% (CPU warning — exceeded recommended 70%)
- db-01: CPU 23%, Memory 55% (Normal)

web-02's CPU is high at 87%. Would you like to check its processes?

Example 2. APM anomaly detection

User: Check if there are any anomalies in the Java APM project.

AI: I analyzed 5 agents.

Anomaly detected (1 instance):

- api-server-03: Response time spike (avg 120ms → max 2.3s, $+4.2\sigma$)

The remaining 4 agents are within normal range.

Would you like to check the transaction details for api-server-03?

Example 3. Incident response

User: I just got an error alert. Summarize the error situation for the last 5 minutes.

AI: Here is the transaction error summary for the last 5 minutes.

Agent	TPS	Errors	Error rate	
---	---	---	---	
api-01	45.2	23	10.2%	
api-02	42.8	2	0.9%	
api-03	44.1	0	0%	

api-01 has an error rate of 10.2%, which is abnormal.

Would you like to check the response time trend and Active TX as well?

Example 4. Executive report data

User: Summarize this week's response time and error rate trends across all services.

AI: Here is the service status for the last 7 days.

- Average response time: 142ms (-8% vs. previous week)
- Average error rate: 0.3% (no change vs. previous week)
- Average TPS: 1,240/s (+12% vs. previous week)

Overall the service is in good shape. Despite the TPS increase, response time has actually decreased, indicating improved performance.

Troubleshooting

If you encounter a problem while connecting to or using the MCP server, check the items below.

"WHATAP_API_TOKEN environment variable is required"

Q The API token is not configured or not being passed to the MCP server.

A Check the following items in each client's configuration file.

- Claude Desktop: Verify the `"env"` block contains the `"WHATAP_API_TOKEN"` key (not `"WHATAP_TOKEN"`)
- Codex CLI: Verify the `[mcp_servers.whatap.env]` section exists in the TOML file
- Gemini CLI: Verify the `"env"` block exists in [settings.json](#)
- Claude Code: Verify the `-env` flag is included in the `claude mcp add` command

"No API token found for project XXXXX"

Q The account token is valid, but the project code does not exist or you do not have access.

A Enter "Show me my project list" in the AI assistant to confirm the correct project code.

"npx: command not found"

Q Node.js is not installed or not registered in PATH.

A Install Node.js 18 or higher.

```
node --version # Check for v18 or higher  
npx --version # Check that a version is printed
```

"spawn git ENOENT" or "An unknown git error occurred"

Q Git is not installed or its executable path is not registered in PATH.

A The WhaTap MCP server references a GitHub repository directly, so it requires Git. This error occurs mainly on Windows.

```
winget install --id Git.Git -e --source winget
```

After installation, open a new terminal and run the following command.

```
git --version # Check that a version is printed
```

For more details, refer to the [Getting Started > Install Git](#) document.

MCP server fails to start or times out

Q The MCP server does not start or a timeout occurs.

A Run the server directly to check for error messages.

```
WHATAP_API_TOKEN=your_token_here npx -y github:whatap/whatap-open-mcp
```

If the server starts normally, it will wait for input. Press **Ctrl+C** to stop. If error messages appear, check the token value or Node.js version.

Tools not appearing in Claude Desktop

Q MCP tools are not displayed in Claude Desktop.

A Check the following items in order.

1. Verify the JSON file is valid (missing commas, mismatched brackets, etc.).
2. **Fully quit and restart** Claude Desktop (quit the app, do not just close the window).
3. Check the logs: `~/Library/Logs/Claude/mcp*.log` (macOS)

Tools not appearing in Codex CLI

Q MCP tools are not displayed in Codex CLI.

A Check the following items in order.

1. Verify registration with `codex mcp list` .
2. Verify `~/.codex/config.toml` is in valid TOML format (JSON syntax is not supported).
3. Remove and re-register: `codex mcp add whatap ...`

Tools not appearing in Gemini CLI

Q MCP tools are not displayed in Gemini CLI.

A Check the following items in order.

1. Verify registration with `gemini mcp list` .
2. Verify `~/.gemini/settings.json` is valid JSON.
3. Check the scope: `-scope user` (global) or `-scope project` (project-level)

"WhaTap API error (401)" or "(403)"

Q The API token is invalid or has expired.

A Go to **Account Management > API Token** in WhaTap Console to issue a new token.

Slow responses

Q Responses are slow.

A Reduce the query time range. `"5m"` (5 minutes) takes a few seconds, while `"1d"` (1 day) may take tens of seconds. Consecutive requests may hit WhaTap API rate limits. Wait a moment and try again.

No data returned

Q No data is returned.

A Refer to the causes and remedies below.

Cause	How to check
Project type mismatch (e.g. APM query on a Server project)	Run <code>whatap_data_availability(projectCode)</code> to check active categories → "Show me what data I can query in this project"
Agent not installed or inactive	Run <code>whatap_list_agents(projectCode)</code> to check agent status → "Show me the agent list"
Time range too narrow	Expand the range (e.g. <code>timeRange="1h"</code>) and retry