

AI Agent Observability

release date. 2026. 09. 11.

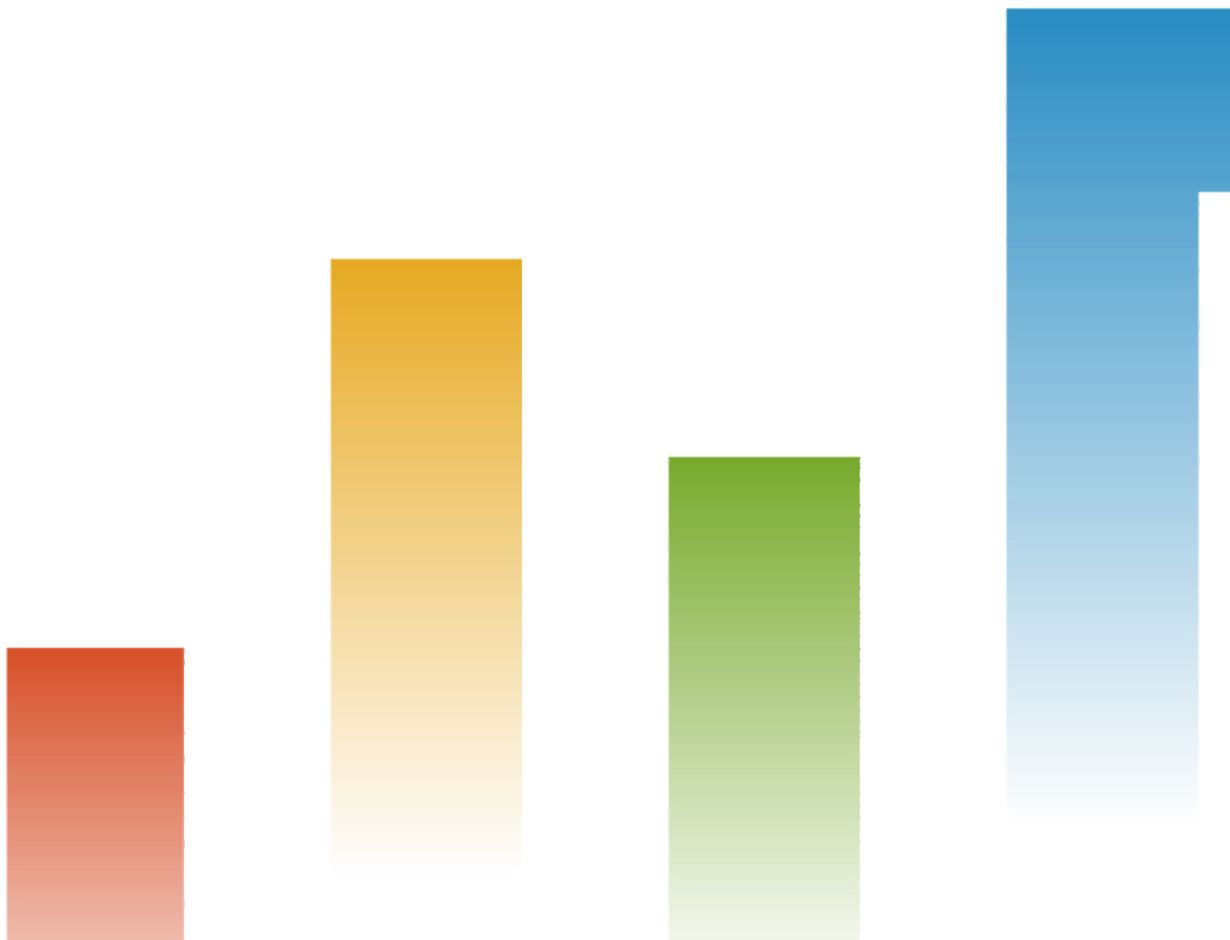


Table Of Contents

 This page is currently being translated. We will complete the translation as soon as possible.

• AI Agent Observability.....	4
◦ Getting Started.....	7
◦ Supported Environment.....	11
◦ Installation	
■ Python.....	18
◦ Configuration	
■ Python.....	26
■ LLM Transaction Configuration.....	28
■ Evaluation	32
■ Operation Type and Prompt Version	39
■ Configuring Session & User ID	41
◦ Dashboard	
■ LLM Dashboard	43
■ Common Features	47
■ Widgets	52
◦ Analysis	
■ Token Trend	76
■ Cost Analysis	86
■ LLM API Analysis.....	95
■ Prompt	
■ Prompt Log.....	103

■ Prompt analysis	110
■ Prompt comparison	117
■ Response quality analysis.....	126
◦ Metrics	
■ LLM Metrics.....	133
◦ OpenMetrics	
■ Template List.....	137
■ Explorer	140
◦ Log	
■ Live Tail	143
■ Log Explorer	156
■ Log Trend	163
■ Log Search.....	170
■ Log Setting	185
◦ Alert	
■ Event Config Overview	0
■ Event Notification.....	192
■ Event History.....	214

AI Agent Observability

AI Agent Observability is a platform for unified monitoring of performance, cost, and reliability in LLM (Large Language Model)-based applications. It collects request volume, response performance, token usage, cost, and error status of LLM APIs in real time, and provides detailed analysis by model, agent, and provider. Combined with WhaTap APM, server, and Kubernetes infrastructure, it enables end-to-end tracing of LLM calls from application transactions to GPU infrastructure.

Why LLM Monitoring?

Detecting LLM anomalies hidden behind HTTP 200 responses

LLM inference engines return HTTP 200 even when the model generates hallucinations or abnormal responses. Traditional server monitoring cannot detect this issue, causing delayed incident awareness and missed response windows. AI Agent Observability tracks anomalies in response time, token patterns, and error rates in real time, detecting model issues invisible through HTTP status codes alone.

Without cost visibility, you cannot control spending

LLM APIs are billed per token on every call. Per-call cost varies significantly depending on the model, prompt length, and response size, and unexpected costs can arise as traffic grows. Real-time visibility into which model, which request, and how much cost is incurred is essential for cost control. Since token costs are charged even for failed requests, error costs must be tracked separately to quantify wasted spend.

Users feel slow responses first

LLM responses can be seconds slower than traditional APIs. In streaming environments, if the first token arrives late or token generation speed drops, users perceive "the response has stopped." When most users are fine but some experience slowness, averages cannot detect it. Tracking which model, at what time, and in what pattern slows down -- with time series analysis and cross-model comparison -- is essential for meaningful improvement.

Preserving call context for prompt reproduction

LLM calls generate different responses even with the same prompt. Without preserving "which prompt, which model, which parameters were used," reproduction is impossible when issues occur. AI Agent Observability collects and preserves **system messages, input prompts, model responses, and tool calls** in their original form for every LLM call. When an issue occurs, the exact call context at that point can be restored, enabling immediate verification and reproduction.

Multi-model environments require comparative analysis

Using multiple LLM models and providers simultaneously in a single application is common. Data-driven evidence is needed for decisions such as selecting the best model for a workload by comparing performance, cost, and error rates, or replacing models with low cost-efficiency.

Scattered monitoring data prevents root cause analysis

Operating AI applications fragments logs across log platforms, metrics across infrastructure monitoring, costs across provider consoles, and traces across APMs. When issues occur, manually correlating data across multiple tools slows root cause identification. AI Agent Observability integrates performance, cost, errors, prompt logs, transaction traces, and GPU infrastructure in a **single platform**, enabling drill-down to root causes without context switching.

AI Agent Observability Key Features

Real-time LLM Dashboard

Monitor the real-time status of LLM APIs on a single screen. Provides widgets for key metrics including request volume, response performance, TTFT, TPOT, tokens, cost, and errors.

Token Trend Analysis and Model Comparison

Analyze token usage patterns and LLM response performance as time series. Compare performance distribution across models, agents, and providers to identify the most stable model.

Cost Analysis and Optimization

Provides token-level cost tracking, change rates compared to previous periods, and cross-model cost comparison. Supports cost optimization decisions with performance-vs-cost bubble charts.

Cache Efficiency Monitoring

Track prompt caching hit rates and cost savings by time period. Quantitatively verify whether caching strategies contribute to actual cost reduction.

Prompt Log Analysis

Collects and preserves system messages, input prompts, model responses, and tool calls for each individual LLM call. Drill down from dashboards to identify root causes after detecting anomalies.

LLM Transaction Tracing and GPU Correlation Analysis

Traces LLM calls as part of application transactions. In the transaction profile, view prompt, token, and cost details of LLM steps, along with GPU correlation analysis.

Next Steps

- [Getting Started](#) — Create a project and select an agent type
- [Supported Environment](#) — Check the Python supported environment
- [LLM Dashboard](#) — Start real-time monitoring

Getting Started With AI Agent Observability

To start WhaTap AI Agent Observability, create a project, select the agent type that matches your LLM application language, and follow the installation guide to apply the agent. Once the agent is applied, you can view the collected data on the LLM dashboard.

WhaTap AI Agent Observability monitoring starts with the following steps.

Step 1. Sign up

To use the WhaTap monitoring service, you must first sign up. For more information about signing up, see the [Account Management > Sign Up](#) documentation.

Step 2. Create a project

Create a project before installing the agent.

1. Log in to the [WhaTap Monitoring Service](#).
2. In the upper left corner of the screen, click  **All Projects** > **+ Create project**.
3.  On the **Select product** screen, select the **AI Observability** category.
4. Click the **Create** button for the **AI Agent Observability** product.
5.  On the **Create project** screen, enter the project information.

Item	Description
Project name	Enter a name that identifies the project.
Data server region	Select the server region where your data is stored. Your data is stored in the selected region.
Time zone	Set the reference time used for alerts and reports.

Item	Description
Notification language setting	Select the language for alert messages. Korean and English are supported.
Project groups (optional)	You can classify projects into groups.
Project description (optional)	You can enter an additional description for the project.

6. After entering all items, click the **Creating a project** button.

! When adding a project with an organization selected, you must set the **Groups of organization**. For more information about groups, see [the following documentation](#).

Step 3. Select an agent type

After creating a project, you are automatically redirected to the **Agent installation** page. Select the agent type that matches your LLM application language. After selecting an agent type, you are redirected to the installation guide page for that language.

1 Select product
2 Create project
3 **Agent installation guide**

Select the agent type(s) to install.


Python
It is scheduled to trace major LLM calls such as OpenAI and Anthropic ones, along with expanding the scope of supporting frameworks.
Agent setup >


Java
It is scheduled to trace LLM calls based on Spring AI, along with expanding the scope of supporting frameworks.
Agent setup >


Node.js
It is scheduled to support major libraries such as OpenAI SDKs, along with expanding the scope of supporting frameworks.
Agent setup >


Others
It is scheduled to sequentially support various language environments.
Agent setup >

Agent Type	Description	Status
Python	Agent for Python-based LLM applications	Supported
Java	Agent for Java-based LLM applications	Coming soon
Node.js	Agent for Node.js-based LLM applications	Coming soon
Others	LLM applications based on other languages	Coming soon

ⓘ If you are not redirected to the **Agent installation** page, select **Management > Agent installation** from the left menu.

Step 4. Check the access key

The **access key** is a unique ID for activating the WhaTap service. In the first step of the agent installation guide page, click the **Getting the access key** button to issue the access key.

You must enter the issued access key in the agent configuration file ([whatap.conf](#)).

Step 5. Install the agent

Install the agent according to the agent type you selected. For detailed installation instructions for each agent type, see the following documentation.

- [Python Agent Installation](#)

Step 6. Check the dashboard

After you install the agent and restart the application, data starts being sent to the WhaTap collection server within a few minutes.

1. In the left sidebar, click **Dashboard > LLM Dashboard**.
2. When LLM API calls occur, the request count, token usage, cost, and performance metrics appear in the widgets.

AI Agent Observability Supported Environment

This document describes the supported environments and version requirements for the AI Agent Observability agent by language. Check whether your application environment falls within the supported range before installing the agent.

Python agent-supported environment

Before installing the WhaTap Python monitoring agent, check the support environment.

! Last Updated: 2026-09-07

Supported Versions

Application Type	Python Version
WSGI Application	3.7 ~ 3.13
ASGI Application	3.7 ~ 3.13

Operating System

OS	Minimum Version	Architecture
Debian / Ubuntu	12.04	64bit
Red Hat / CentOS	6.x	64bit
macOS	10.15 Catalina	-
Windows	Windows Server 2008 R2, Windows 7	64bit

! AI Agent Observability does not currently support Windows.

Application Server

Framework	Supported Version
Bottle	-
Celery	-
CherryPy	-
Django	-
FastAPI	0.84.0 ~ 0.139.2
Flask	-
Frappe	-
Nameko	-
Odoo	v14 ~ v19
Tornado	-

Libraries

Category	Libraries
External Calls	HttpLib, HttpLib2, Requests, Urllib, HTTPX
Data Processing	SQLAlchemy, GraphQL

Category	Libraries
Logging	smtplib, logging, loguru
Messaging	Kombu, Pika
LLM SDK	OpenAI, Anthropic
Agent Orchestration	Hermes, LangGraph

Database

Database	Driver
MongoDB	pymongo
MySQL	MySQLdb, pymysql
Neo4j	neo4j
Oracle	cxOracle
PostgreSQL	psycopg2, psycopg, psycopg_pool
Redis	redis
SQLite	sqlite3

ⓘ The `whatap-python[llm]` package requires the external libraries `datasketches>=5.2.0` and `genai-prices`.

Planned Support

Agent Type	Status
Node.js	Coming soon
Others (Go, PHP, .NET, etc.)	Coming soon

Common supported environment

Service deployment models

You can use WhaTap Monitoring as a SaaS or on-premises (self-hosted) service. Whichever you choose, you use the same agents and the same monitoring features.

Deployment	Support	Description
SaaS		Connects to collection servers operated by WhaTap. See the firewall section for regional addresses
On-premises (self-hosted)		Collection servers you build and operate yourself (air-gapped environments supported). Contact your sales representative for the scope

ⓘ You can also deploy WhaTap on-premises in air-gapped environments without internet connectivity.

Feature differences by deployment model

Whichever you choose, the monitoring features are the same. The mobile app is the only exception and is available on the service.whatap.io SaaS service only.

! To use certain features on-premises, a separate agreement is required.

Supporting the browser

WhaTap Monitoring is available in web browsers and mobile Apps.

Browser	Recommended	Supported version
Google Chrome	O	111 or later
Mozilla FireFox	X	Latest version
Edge	X	Latest version
Safari	X	Latest version

! Support guidance

- For browser compatibility and performance reasons, the latest version of Chrome is recommended.
- The user interface (UI) is implemented using HTML5 standard technologies, so Internet Explorer is not supported.

! Limitations

WhaTap's web interface does not support mobile browsers. To access WhaTap on a mobile device, install the WhaTap app for Android or iOS. The WhaTap mobile app is optimized for mobile environments. For more information about the WhaTap mobile app, see [this document](#).

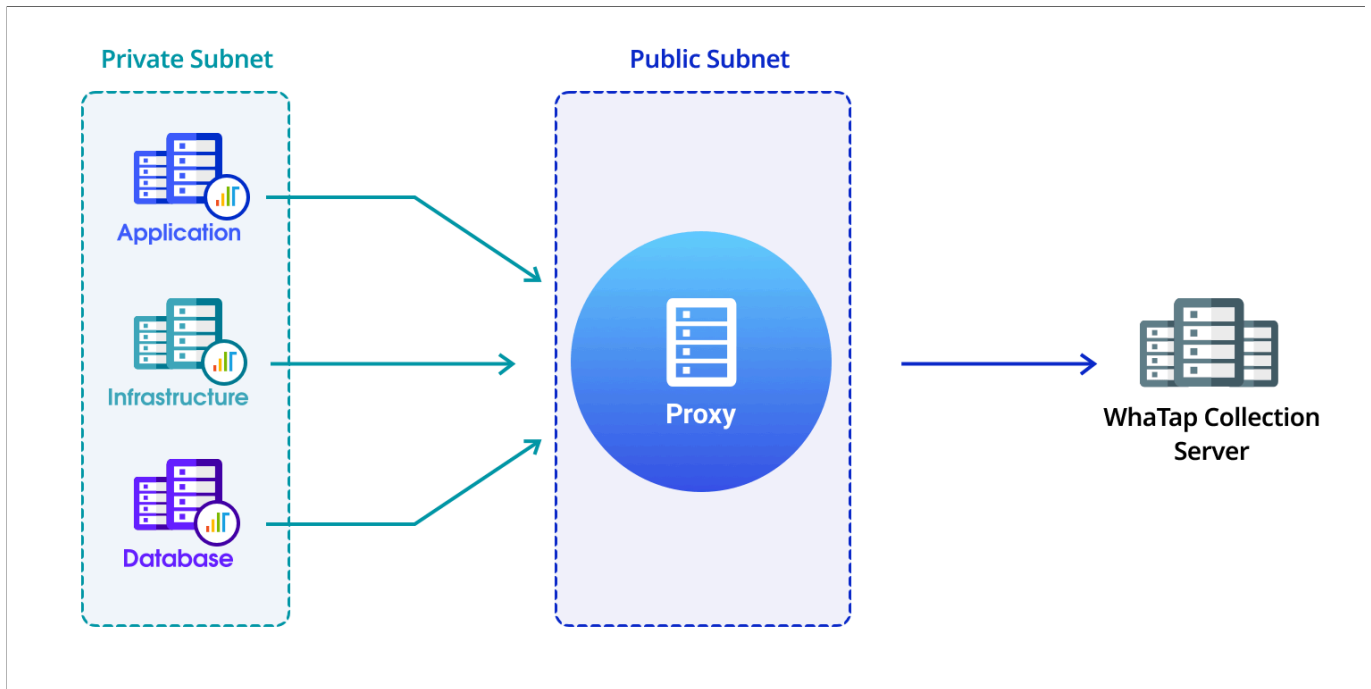
Firewall

The WhaTap agent must be able to access the collection server through **TCP port 6600**. Allow access to the collection server address that is closest to the monitored target.

Source: WhaTap agent

Cloud	Destination	Destination IP	Port
AWS	WhaTap collection server (Seoul)	13.124.11.223 / 13.209.172.35	TCP 6600
	WhaTap collection server (Tokyo)	52.68.36.166 / 52.193.60.176	TCP 6600
	WhaTap collection server (Singapore)	18.138.0.93 / 18.139.67.236	TCP 6600
	WhaTap collection server (Jakarta)	108.136.91.69 / 108.137.158.44	TCP 6600
	WhaTap collection server (Thailand)	43.209.109.219 / 43.209.48.86	TCP 6600
	WhaTap collection server (Mumbai)	13.127.125.69 / 13.235.15.118	TCP 6600
	WhaTap collection server (California)	52.8.223.130 / 52.8.239.99	TCP 6600
	WhaTap collection server (Virginia)	107.23.220.101 / 54.236.221.105	TCP 6600
	WhaTap collection server (Frankfurt)	3.125.142.162 / 3.127.76.140	TCP 6600
Azure	WhaTap collection server (Seoul)	52.231.66.38 / 20.194.5.115	TCP 6600
	WhaTap collection server (Tokyo)	52.246.169.54 / 20.210.27.232	TCP 6600
Kakao	WhaTap collection server (Seoul)	61.109.237.237 / 61.109.238.166	TCP 6600

If the agent cannot directly access the collection server, use the provided proxy module to access it.



LLM Python Agent Installation

The AI Agent Observability Python agent is installed via `pip install whatap-python[llm]` and applied by adding `whatap-start-agent` before your application's run command. This document guides you through the entire installation process, from activating a virtual environment to verifying the service.

Pre-installation checklist

Before installing the agent, verify the following:

- Confirm that you have completed [project creation](#) and [access key issuance](#) in [Getting Started](#).
- Check that your application environment falls within the supported range in [Supported Environment](#).

1 Activate virtual environment

If your application uses [virtualenv](#), run the [bin/activate](#) file to activate the virtual environment.

2 Download agent

After issuing an **access key**, go to the **Download agent** section. Run the following command to install the agent.

```
pip install whatap-python[llm]
```

- ✓ If you cannot install via the pip command, download the installation file from the [pypi WhaTap page](#). Extract the downloaded file and proceed with installation.

```
tar xzvf whatap_python-2.x.x.tar.gz
cd whatap_python-2.X.Y.Z
python setup.py install
```

❗ The `whatap-python[llm]` package requires the external libraries `datasketches>=5.2.0` and `genai-prices`. Missing dependencies may cause some data collection to be skipped. Include these dependencies even when installing manually.

Agent configuration files

The Python agent files consist of a tracer that extracts information required for application monitoring and sends it to the WhaTap collection server, along with supporting elements. See the following for the agent file structure.

📁 whatap

3

Configure agent

Set WHATAP_HOME path

Specify the `$WHATAP_HOME` path for log and configuration file paths. Creating a new whatap directory is recommended.

```
$ export WHATAP_HOME=[PATH]
```

Set access key and collection server IP

Run the `whatap-llm-setting-config` command to set the access key and collection server IP.

```
$ whatap-llm-setting-config \
--host [ COLLECTION_SERVER_IP ] \
--license [ ACCESS_KEY ] \
--app_name [ USER_DEFINED_AGENT_NAME ] \
--app_process_name [ APPLICATION_PROCESS_NAME(uwsgi, gunicorn etc..) ]
```

Verify configuration

The `whatap.conf` file is created and configured at the path specified in `$WHATAP_HOME`. Run the following command to verify the `whatap.conf` file was created.

```
$ cat $WHATAP_HOME/whatap.conf
```

whatap.conf

```
llm_license=[ACCESS_KEY]
llm.whatap.server.host=[COLLECTION_SERVER_IP]

llm_enabled=true

# application name
app_name=[ USER_DEFINED_AGENT_NAME ]

# middleware process name ex)uwsgi, gunicorn ..
app_process_name=[ APPLICATION_PROCESS_NAME(uwsgi, gunicorn etc..) ]
```

Install the LLM-only module

To use Python AI Agent Observability, you must additionally install the Go module that WhaTap developed for LLM.

```
cd $WHATAP_HOME

curl -fsSL -O https://repo.whatap.io/python/llm/whatap-python-llm.tar.gz

tar -xzf whatap-python-llm.tar.gz
```

The `whatap-python-llm` directory created by extracting the archive must be located under the `$WHATAP_HOME` path you set earlier.

⚠ If permission issues occur

- Read and write permissions for the `$WHATAP_HOME/whatap.conf` file for WhaTap configuration



- Read and write permissions for the `$WHATAP_HOME/logs` path and its sub-files for WhaTap logs
- If permission issues occur for the `$WHATAP_HOME` path, run the following command to grant permissions.

```
echo `sudo chmod -R 777 $WHATAP_HOME`
```

4 Apply per server environment

Select the method that matches your server environment.

- Command **uWSGI** **Gunicorn** **Supervisor** **WSGI** **Docker** **Code**

In a **Command** environment, add the `whatap-start-agent` command before the application start command as follows.

BASH

```
# $ whatap-start-agent [Application start command]
$ whatap-start-agent python manage.py runserver
```

In a **uWSGI** environment, add the `whatap-start-agent` command before the `uwsgi` command as follows.

BASH

```
$ whatap-start-agent uwsgi --ini myapp.ini
```

If running the application by registering uWSGI as a service, refer to the following example code.

/etc/init.d/uwsgi

```
description "uWSGI application server handling myapp"
start on runlevel [2345]
stop on runlevel [!2345]
exec whatap-start-agent [YOUR_APPLICATION_START_COMMAND]
# or if you are using a virtual environment
exec env WHATAP_HOME=[PATH] [ABSOLUTE_PATH]/whatap-start-agent [YOUR_APPLICATION_START_COMMAND]
```

/etc/init/uwsgi.conf

```
...
NAME="uwsgi"
PATH=/usr/local/bin:/sbin:/bin:/usr/sbin:/usr/bin
DAEMON=/usr/local/bin/uwsgi
PID=$RUN/$NAME.pid
INI_PATH=/etc/$NAME/$NAME.ini

##### WHATAP_AGENT_CONF #####
WHATAP_HOME=[PATH]
WHATAP_AGENT=[ABSOLUTE_PATH]/whatap-start-agent
...
do_start(){
  env WHATAP_HOME=$WHATAP_HOME $WHATAP_AGENT [YOUR_APPLICATION_START_COMMAND]
}
```

- ❗ • If changing the user, add the `WHATAP_HOME` environment variable.
- If using a virtual environment, add the agent start command as an absolute path.
- Modify the service execution file (`/etc/init/uwsgi.conf`) to start the application with the agent start command.
- The service execution file path may differ depending on your environment.

In a **Gunicorn** environment, add the `whatap-start-agent` command before the `Gunicorn` command as follows.

BASH

```
$ whatap-start-agent gunicorn myapp.wsgi
```

If running the application by registering Gunicorn as a service, refer to the following example code.

/etc/init.d/gunicorn

```
description "gunicorn application server handling myapp"
start on runlevel [2345]
stop on runlevel [!2345]
exec whatap-start-agent [YOUR_APPLICATION_START_COMMAND]
```

or if you are using a virtual environment

```
exec env WHATAP_HOME=[PATH] [ABSOLUTE_PATH]/whatap-start-agent [YOUR_APPLICATION_START_COMMAND]
```

/etc/init/gunicorn.conf

```
...
NAME="gunicorn"
PATH=/usr/local/bin:/sbin:/bin:/usr/sbin:/usr/bin
DAEMON=/usr/local/bin/gunicorn
PID=$RUN/$NAME.pid
INI_PATH=/etc/$NAME/$NAME.ini

##### WHATAP_AGENT_CONF #####
WHATAP_HOME=[PATH]
WHATAP_AGENT=[ABSOLUTE_PATH]/whatap-start-agent

...
do_start(){
    env WHATAP_HOME=$WHATAP_HOME $WHATAP_AGENT [YOUR_APPLICATION_START_COMMAND]
}
```

- ! • If changing the user, add the `WHATAP_HOME` environment variable.
- If using a virtual environment, add the agent start command as an absolute path.
- Modify the service execution file (`/etc/init/gunicorn.conf`) to start the application with the agent start command.
- The service execution file path may differ depending on your environment.

In a **Supervisor** environment, add the following settings to the `supervisor.conf` file.

supervisor.conf

```
[program:app-uwsgi]
environment = WHATAP_HOME=[PATH]
command = [ABSOLUTE_PATH]/whatap-start-agent /usr/local/bin/uwsgi --ini /home/blog/backend/config/uwsgi.ini

[program:nginx-app]
command = /usr/sbin/nginx
```

- ❗ • If changing the user, add the `WHATAP_HOME` environment variable.
- If using a virtual environment, add the agent start command as an absolute path.
- Modify the service execution file (`/etc/supervisor/conf.d/supervisor.conf`) to start the application with the agent start command.
- The service execution file path may differ depending on your environment.

The following guides how to directly implement a WSGI application.

PYTHON

```
import whatap

@whatap.register_app
def simple_app(envIRON, start_response):
    """Simplest possible application object"""
    status = '200 OK'
    response_headers = [('Content-type', 'text/plain')]
    start_response(status, response_headers)
    return ['Hello world!\n']
```

For applications served as Docker containers, add the following to the [Dockerfile](#).

1. Install the **Python agent**.

Dockerfile

```
ENV WHATAP_HOME /whatap
RUN mkdir -p /whatap
RUN pip install whatap-python[llm]
```

- ❗ If using `requirements.txt`, add `whatap-python[llm]` to the file.

2. Register the **access key** and **collection server IP** settings.

Dockerfile

```

RUN touch /whatap/whatap.conf
RUN echo "llm_license=[ ACCESS_KEY ]" > /whatap/whatap.conf
RUN echo "llm.whatap.server.host=[ COLLECTION_SERVER_IP ]" >> /whatap/whatap.conf
RUN echo "app_name=[ AGENT_NAME ]" >> /whatap/whatap.conf
RUN echo "app_process_name=[ APPLICATION_PROCESS_NAME(uwsgi, gunicorn etc..) ]" >> /whatap/whatap.conf

```

3. Add `whatap-start-agent` before the existing run command `python manage.py runserver` .

Dockerfile

```

CMD ["whatap-start-agent", "python", "manage.py", "runserver", "0.0.0.0:8000"]

```

You can apply the agent by adding API call code at the top of your code as follows.

When running the agent via code, add the following option to [whatap.conf](#).

whatap.conf

```

whatap_code_start=true

```

Add the following at the top of your code.

PYTHON

```

import whatap
whatap.agent()

```

Once you start the application server, the agent begins collecting monitoring data.

5 Verify service execution

Run the following command to verify the WhaTap Python service is running properly.

```

ps -ef | grep whatap_python

```

Python Agent Settings

Required options

```
llm_license= llm.whatap.server.host=
```

The Python agent can be used with an APM project and an LLMOBS project together. The values related to the LLM project must be tagged in these options.

```
llm_enabled=true
```

This option enables the LLM logsink pack and metric collection. When you use LLMOBS, this option must be set to `true`.

Recognizing LLM API hosts

```
llm_api_hosts=localhost:8000,api.openai.com:443
```

This option lets the agent treat calls to LLM API endpoints that automatic instrumentation does not know (vLLM, LiteLLM, a private gateway, and so on) as LLM calls. The httpc driver marks calls going to those hosts with the `LLM API` driver.

Entering model prices manually

```
llm_model_pricing=gpt-4o|2.50|10.00|1.25,gpt-4o-mini|0.15|0.60|0.075,my-custom-model|1.00|5.00
```

You can specify the price per token of a model manually. The configured value takes the highest precedence even when the actual cost is recognized.

Format: `model|input_per_1m|output_per_1m|cached_per_1m`

- 4 fields
- Models are separated by commas

- The fourth field (cached) is optional and can be left empty or omitted

LLM Transaction Configuration

This is a guide to using the public API for instrumenting workflow and agent patterns in AI Agent Observability.

Once you declare the scope of a workflow or AI agent, every LLM API call that occurs within that scope is collected as a **"step" under a single LLM transaction** instead of as a separate transaction. All you need to do is mark the scope.

Transaction types in AI Agent Observability

There are three types of LLM transactions.

- `Workflow` (default) — `[Workflow] <name>`
- `Agent` — `[Agent] <name>`
- `LLMAPI` — A standalone call that is not wrapped in a workflow is collected as `[LLMAPI] Pure LLM API`.

When another workflow runs inside a workflow, the parent and child transactions are linked as a Trajectory so you can trace the call relationship.

Configuring a workflow transaction

Both the decorator and the context manager forms are supported.

Decorator

Use this when the workflow name is fixed per function.

```
from whatap.llm import workflow

@workflow("support_chat")
async def support_chat(conversation_id, message, history):
    intent = await classify_intent(message) # Every LLM call in here
    docs = await retrieve(message) # becomes a "step" of this workflow
```

```
answer = await generate(message, docs)
return {"intent": intent, "answer": answer}
```

Context manager

Use this when you determine the workflow name at runtime or want to wrap only part of a function.

```
from whatap.llm import workflow

with workflow("ticket_triage"):
    classification = await _classify(ticket_text)
    entities = await _extract(ticket_text)
```

❗ Even in asynchronous code, use `with` as shown above, not `async with`. Because `workflow` and `agent` are synchronous context managers, using `await` inside the scope is fine, but opening them with `async with` raises an error.

Configuring an agent transaction

The usage is the same as for a workflow, and only the transaction name prefix changes to `[Agent]`. Both the decorator and the context manager forms are supported.

Decorator

Use this when the agent execution function is fixed.

```
from whatap.llm import agent

@agent("react_agent")
async def run_agent(task):
    ...
    return result
```

Context manager

Use this when you determine the agent name at runtime or want to wrap only part of the execution block.

```
from whatap.llm import agent

with agent("react_agent"):
    result = await react_agent.run(task)
```

Declaring a scope without modifying code

When you cannot modify the application code, you can specify workflow boundaries through configuration. List the target methods in [whatap.conf](#), and the execution of those methods becomes the workflow boundary. The method name is used as the workflow name.

whatap.conf

```
workflow_method_patterns=myapp:RagService.run, myapp.agent:run_agent
```

Use the `module:Class.method` or `module:function` form, separated by commas.

Frameworks with automatic instrumentation

For the following frameworks, LLM transactions are collected automatically without declaring a scope.

Framework	How it is collected
hermes-agent	Instruments the agent execution layer (<code>AIAgent.run_conversation</code>) and registers it as an <code>[Agent]</code> <code><name></code> transaction. The gateway inbound (<code>GatewayRunner.handle_message</code>) is registered as a <code>[Workflow]</code> <code>gateway/<platform></code> transaction, and agents executed within it are linked as child transactions.
LangGraph	Instruments the execution boundary of a compiled graph (<code>Pregel.invoke</code> / <code>ainvoke</code> / <code>stream</code> / <code>astream</code>) and registers one graph execution as one LLM transaction . A graph with a <code>tools</code> node

Framework	How it is collected
	(<code>create_react_agent</code>) is registered as <code>[Agent]</code> , and others as <code>[Workflow]</code> . Subgraphs are nested as child transactions.

Evaluation

WhaTap AI Agent Observability provides the **Evaluation** feature, which automatically measures the response quality and stability of the LLM application you operate.

Evaluation automatically applies evaluations such as hallucination, answer relevance, toxicity, prompt injection, factuality, PII leak, and URL inclusion to LLM responses. The evaluation results are produced as scores (0.0 to 1.0) and collected together with metrics and the logsink, so you can track response quality as a time series and configure threshold-based alerts.

ⓘ The Evaluation feature supports the Python agent only.

⚠ When you enable the evaluation pipeline, **additional LLM resources (tokens and cost) are consumed for the judge calls**, separately from the LLM calls that serve user responses.

Configuration options

Key	Default	Description
<code>llm_eval_enabled</code>	<code>false</code>	Master toggle of the evaluation pipeline. If <code>true</code> , the pipeline starts and the default evaluators are registered automatically. All other options are optional.
<code>llm_eval_evaluators</code>	<code>combined_judge,</code> <code>pii_leak, url_scan</code>	Selects the evaluation pipelines to run in the agent.

`combined_judge` is a combined evaluator that **calls the judge only once and infers 5 aspects** (`hallucination` , `answer_relevance` , `toxicity` , `prompt_injection` , `factuality`) at once.

To control the aspects individually, remove `combined_judge` and list only the labels you need. The following values are available:

- LLM judge based: `hallucination` · `answer_relevance` · `toxicity` · `prompt_injection` · `factuality`

- Rule based (no LLM call): `pii_leak` · `url_scan`

whatap.conf

```
llm_eval_evaluators=hallucination,toxicity,pii_leak
```

Note that each individual label calls the judge separately, so specifying all 5 results in 5 judge calls per response. For cost reasons, we recommend using `combined_judge`.

How judge calls work

For judge LLM calls, the agent creates its own client using the **credential snapshot (endpoint and API key)** captured by the instrumentation from the user calls. Because it does not borrow the client instance of the user request as is, it also works safely in applications that create and close a client for every request.

The judge model is set to the same model the application used. This is because the snapshotted endpoint may be self-hosted or a proxy, so the cloud default model cannot be assumed. That is, if the application uses a high-quality or low-quality model, the judge calls run with the same model. To adjust the evaluation model, **explicitly specify** the model in `llm_eval_judge_model`.

Evaluation credentials

Specify Evaluation credentials directly in [whatap.conf](#) only in the following cases:

- The snapshot uses OAuth authentication, so there is no plaintext API key.
- The snapshotted endpoint does not serve the `chat.completions` path.
- You intentionally want to send evaluations to a different model or endpoint.

Use one of the following **two modes**. The moment you specify either mode, it **completely replaces** the credential snapshot path.

OpenAI-compatible custom endpoint

whatap.conf

```
llm_eval_judge_base_url=http://10.0.0.5:8000/v1
```

```
llm_eval_judge_api_key=EMPTY
llm_eval_judge_model=qwen2.5-7b-instruct
```

- You can omit `llm_eval_judge_provider` . If `base_url` is present, it is treated as OpenAI compatible.
- Calls go out as `POST <base_url>/chat/completions` .
- For a local engine that does not require a key, set `llm_eval_judge_api_key=EMPTY` .

Cloud provider API directly

Sends the judge to the default endpoint of OpenAI or Anthropic.

whatap.conf

```
llm_eval_judge_provider=openai
llm_eval_judge_api_key=OPENAI_API_KEY
llm_eval_judge_model=gpt-4o-mini
```

- Set `llm_eval_judge_provider` to `openai` or `anthropic` .
- Do not specify `llm_eval_judge_base_url` . The SDK default endpoint is used.

Cost and performance options

Key	Default	Description
<code>llm_eval_sample_rate</code>	<code>1.0</code>	Ratio of LLM judge evaluations. <code>0.1</code> evaluates only 10%, reducing the cost to 1/10. Rule-based evaluators are unaffected and always run at 100%.
<code>llm_eval_judge_timeout_sec</code>	<code>30</code>	Upper limit (seconds) for one judge call. <code>0</code> or a negative value means unlimited.
<code>llm_eval_workers</code>	<code>4</code>	Number of evaluation worker threads.
<code>llm_eval_buffer_limit</code>	<code>1000</code>	Maximum size of the evaluation queue. When exceeded, evaluations are dropped and the <code>LLM030</code> warning occurs. The application is not

Key	Default	Description
		blocked.
<code>llm_eval_track_judge_calls</code>	<code>false</code>	Whether to also collect the judge's own LLM calls in metrics and the logsink. If <code>true</code> , the cost and latency of the judge become visible, but the LLM call count increases (user calls + judge calls). In this case, the <code>operation_type</code> of judge calls is recorded as <code>whatap_evaluation</code> .

Applying to specific pipelines only

Use a decorator or a context manager to evaluate only the LLM calls of a specific function or block.

```
from whatap.llm.evaluators import evaluate_with, evaluation_scope
from whatap.llm.evaluators.builtins import CombinedJudgeEvaluator
```

```
@evaluate_with(CombinedJudgeEvaluator())
```

```
def chat(q):
```

```
    ...
```

```
def chat2(q):
```

```
    with evaluation_scope(CombinedJudgeEvaluator()):
```

```
        ...
```

The evaluator classes you can register in code are the same as the options you register through whatap.conf.

```
from whatap.llm.evaluators.builtins import (
```

```
    # LLM judge based — judge call cost is incurred
```

```
    CombinedJudgeEvaluator,
```

```
    HallucinationEvaluator,
```

```
    AnswerRelevanceEvaluator,
```

```
    ToxicityEvaluator,
```

```
    PromptInjectionEvaluator,
```

```
    FactualityEvaluator,
```

```
    # Rule based — no LLM call, zero cost
```

```
    PIILeakEvaluator,
```

```
URLScanEvaluator,
)
```

Evaluation metric descriptions

Evaluator	LABEL	Type	Score direction	Cost
CombinedJudgeEvaluator	combined_judge	LLM judge	Overall risk — lower is better	5 aspects with 1 judge call
HallucinationEvaluator	hallucination	LLM judge	Lower is better	1 judge call
AnswerRelevanceEvaluator	answer_relevance	LLM judge	Higher is better	1 judge call
ToxicityEvaluator	toxicity	LLM judge	Lower is better	1 judge call
PromptInjectionEvaluator	prompt_injection	LLM judge	Lower is better	1 judge call
FactualityEvaluator	factuality	LLM judge	Higher is better	1 judge call

Each Evaluation measures the following.

- **answer_relevance** : **Answer relevance (higher is better)**

Measures how faithfully the response answers the user question. It is scored **1.0** when the question is fully answered, **0.5** when it is answered only partially or tangentially, and **0.0** when it is unrelated to the question or evades or deviates from it.

- **hallucination** : **Hallucination (lower is better)**

Measures whether the response contains unsupported claims. When a context (ground truth) is given, it is evaluated based on faithfulness to that context; when there is no context, it is evaluated based on the self-

consistency of the response itself. `0.0` means completely faithful to the context, and `1.0` means entirely fabricated.

- **toxicity : Toxicity (lower is better)**

Determines whether the response contains harmful content across six categories — hate, harassment, violence, sexual, self_harm, and profanity — and also returns the list of detected categories. `0.0` means completely safe, and `1.0` means severely harmful.

- **prompt_injection : Prompt injection (lower is better)**

Determines whether an override attempt of the "ignore previous instructions" kind included in the user input succeeded, or whether the response leaked the system prompt, hidden instructions, or confidential information. `0.0` means the original task was performed as is and no protected information was exposed, and `1.0` means it was completely hijacked by the injection.

- **factuality : Factuality (higher is better)**

Measures the factual accuracy of the response. It is based on verifiable objective claims such as historical, scientific, geographical, and mathematical facts, as well as dates, names, and numbers, and excludes opinions and hedged expressions from evaluation. `1.0` means all factual claims are accurate, and `0.0` means many clearly false claims are included. Unlike `hallucination`, which looks at faithfulness to the context, factuality looks at the factual accuracy of the response itself regardless of the context.

Overall risk score (combined_judge)

`combined_judge` combines the 5 aspect scores into a single overall risk. At this point, the two groups with different score directions are unified on a risk basis.

- **Risk-direction aspects** (`hallucination` , `toxicity` , `prompt_injection`) — A higher score means higher risk, so the score is used as the risk as it is.
- **Quality-direction aspects** (`answer_relevance` , `factuality`) — A higher score is better, so `1 – score` is used as the risk.

Assuming each aspect is an independent risk, the overall risk is calculated as the complement of "the probability that all aspects are safe." This is the **Probabilistic OR** formula from statistics.

Probabilistic OR formula

Calculation example

For risks of $[0.7, 0.3, 0.1, 0.1, 0.2]$: $1 - (0.3 \times 0.7 \times 0.9 \times 0.9 \times 0.8) = 0.864$

Characteristics of risk combination (simulation)

Individual risks	Maximum (max)	Combined (compound)
$[0.5, 0, 0, 0, 0]$	0.50	0.50
$[0.5, 0.5, 0, 0, 0]$	0.50	0.75
$[0.3, 0.3, 0.3, 0.3, 0.3]$	0.30	0.83
$[1.0, 0, 0, 0, 0]$	1.00	1.00

The overall risk is always greater than or equal to the simple maximum. Even low risks raise the overall risk as they accumulate across several aspects, and if any single aspect is 1.0, the overall risk also becomes 1.0 regardless of the others. This effectively catches "responses that are slightly bad in several areas at once," which a single metric easily misses.

Operation Type And Prompt Version

Operation Type is a user-defined label for **grouping and filtering LLM calls by business context**. The label you specify becomes part of the metric tags, so you can analyze various metrics by your own unit.

Prompt Version is a user-defined label that indicates **which revision of the prompt text** was used within an `operation_type`. If `operation_type` answers "what does this call do" (business unit), `prompt_version` answers "which prompt did this call use" (revision unit).

⚠ Operation Type and Prompt Version labeling supports the Python agent only.

How to use

(a) Decorator (recommended)

```
from whatap.llm import prompt_meta

@prompt_meta(operation_type='checkout_chain', prompt_version='v3')
def checkout(question):
    return client.chat.completions.create(...)
```

(b) Context manager

```
from whatap.llm import prompt_meta_scope

def handler(req):
    with prompt_meta_scope(operation_type='greeting', prompt_version='v2'):
        return client.chat.completions.create(...)
```

Query the currently active meta (in custom logic)

```
from whatap.llm import get_prompt_meta

op_type, version = get_prompt_meta() # Returns ('default', 'v1') if there is no scope
```

When you do not specify values

When there is no label, `operation_type` is automatically replaced with the name of the enclosing workflow or agent. Whether the prompt text has been revised, however, is information the WhaTap agent cannot know, so `prompt_version` is recorded with the value of `llm_prompt_default_version`. If you do not set this option, it is recorded as `v1`.

Changing the prompt version in bulk

If you revised prompts in bulk but it is hard to touch every decorator, move the default version in whatap.conf. Calls that already carry an explicit label are not affected.

whatap.conf

```
llm_prompt_default_version=v2
```

The setting is applied at runtime, so the new default applies from the next call without a restart.

Configuring Session & User ID

Setting the Session ID and User ID lets you group and trace LLM calls by session and user. There are two configuration methods, and when more than one applies at the same time, only one takes effect according to precedence.

Code (API) method

When you call `set_session_id()` and `set_user_id()` in a request handler, the Session ID and User ID apply to every LLM call that occurs within that request context.

```
from whatap.llm import set_user_id, set_session_id

@app.post("/chat")
async def chat(req):
    set_user_id(req.user.id)          # who
    set_session_id(req.conversation_id) # which conversation
    ...
```

Both functions **record the values in the active transaction context**. You must therefore call them inside a request handler or inside an instrumented workflow; calling them outside a transaction (at module load time, before a worker starts, and so on) does nothing. The return value (`bool`) tells you whether they were applied.

Specifying a scope

To mark a block outside a request handler — in a background job or a batch, for example — use the context managers. On leaving the block, the previous values are restored.

```
from whatap.llm import session, user

with user(user_id):
    with session(conversation_id):
        client.chat.completions.create(...)
```

HTTP header method

When the Session ID and User ID are passed in HTTP request headers, you can apply them with [whatap.conf](#) settings alone, without modifying any code.

whatap.conf

```
llm_session_header=X-Conversation-Id  
llm_user_header=X-User-Id
```

In the example above, the `X-Conversation-Id` header value is used as the Session ID and the `X-User-Id` header value as the User ID.

Frameworks with automatic instrumentation

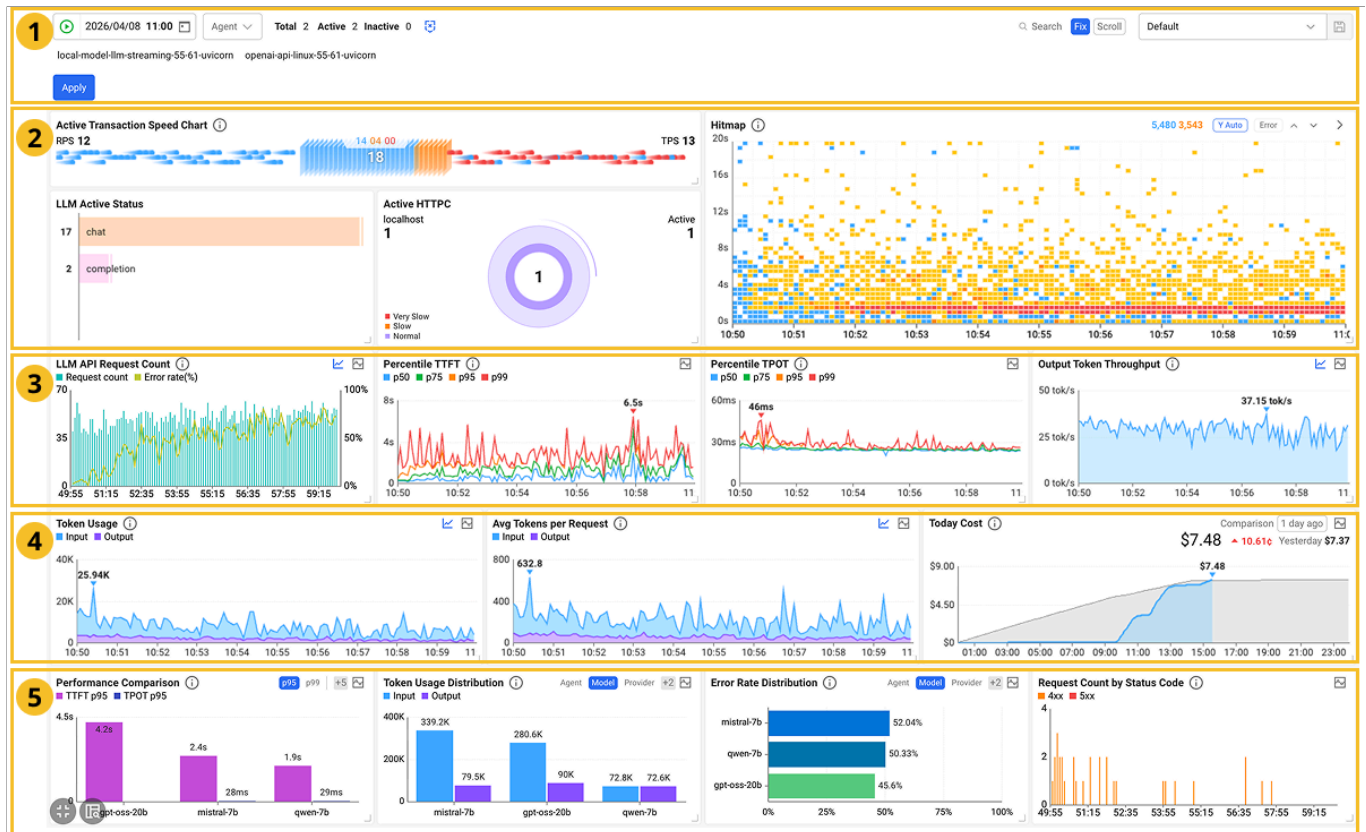
For the following frameworks, the Session ID and User ID are collected automatically without any configuration.

Framework	How it is collected
hermes-agent	Collected automatically from the event information at the gateway inbound (<code>GatewayRunner.handle_message</code>). The User ID is the user identifier of the messenger platform (the value filled in by the telegram, slack, or whatsapp adapter), and the Session ID uses the session key by which hermes groups conversation history.

LLM Dashboard

Home > Select Project > **Dashboard** > LLM Dashboard

The LLM Dashboard is a menu that provides real-time monitoring of LLM (Large Language Model) API request volume, response performance, token usage, cost, and error status in a single view. Widgets are arranged from top to bottom in the order **Real-time Status** → **Requests & Performance** → **Tokens & Cost** → **Comparison & Errors**, allowing you to naturally progress from status overview to root cause analysis. This layout is saved as the **default preset** and is ready to use immediately without additional configuration.



The LLM Dashboard consists of five areas based on the default preset.

Area	Name	Description
①	Option Bar	Set the lookup time, agent filter, screen mode, and preset.

Area	Name	Description
②	Real-time Status	View current transaction speed, LLM call types, active HTTPC, and hitmap in real time.
③	LLM Performance Metrics	Assess LLM response quality through API request count, TTFT/TPOT percentiles, and output token throughput.
④	Tokens & Cost	Track input/output token usage, average tokens per request, and today's cost.
⑤	Model Comparison & Analysis	Compare performance, tokens, and errors across models and check request counts by status code.

Option Bar

Set the lookup time, agent filter, screen mode (fixed/scroll), and preset in the option bar at the top of the dashboard. For detailed usage of each feature, see [Dashboard Common Features](#).

Real-time Status

Displays the real-time status of the application.

- **Active Transaction Speed** — Displays requests per second (RPS), responses per second (TPS), and ongoing transaction count in real time.
- **LLM Active Status** — Shows currently in-progress LLM calls categorized by type (`chat` , `completion` , etc.).
- **Active HTTPC** — Classifies the real-time status of outbound HTTP requests to external LLM APIs by speed range (Very Slow, Slow, Normal).
- **Hitmap** — Plots completed transactions as dots on a time axis (X) and elapsed time axis (Y). Dots higher on the chart indicate slower transactions.

LLM Performance Metrics

Provides key LLM API performance metrics at a glance.

- **LLM API Request Count** — Displays request count (bars) and error rate (line) by time period. If request spikes and error rate increases occur simultaneously, suspect Rate Limit exceeded or provider outage.
- **Percentile TTFT** — Shows Time To First Token in p50, p75, p95, and p99 percentiles. High TTFT means longer initial wait time perceived by users.
- **Percentile TPOT** — Shows Time Per Output Token in percentiles. High TPOT may cause streaming responses to stall or slow down.
- **Output Token Throughput** — Displays output tokens per second (tok/s). A sudden drop in throughput may indicate provider-side performance degradation.

Tokens & Cost

Tracks token usage and cost.

- **Token Usage** — Shows input and output token usage over time in a trend chart.
- **Average Tokens per Request** — Displays average input and output tokens per request. Useful for verifying prompt optimization effects.
- **Today's Cost** — Displays today's cumulative cost and change compared to the previous day. Enables quick detection of cost anomalies.

Model Comparison & Analysis

Compares performance, tokens, and errors across models.

- **Overall Performance Comparison** — Compares TTFT p95 and TPOT p95 per model in a single chart. Useful for evaluating model replacement or version upgrade impact.
- **Token Usage Distribution** — Compares input/output token usage per model by agent, provider, operation, and endpoint.

- **Error Rate Distribution** — Compares error rates per model by agent, provider, operation, and endpoint. Quickly identify whether errors are concentrated in a specific model.
- **Request Count by Status Code** — Displays request counts by HTTP status code (4xx, 5xx) over time.

ⓘ For dashboard common features such as widget editing, widget options, and presets, see [Dashboard Common Features](#).

For chart types, filters, and usage of each widget, see [LLM Dashboard Widgets](#).

LLM Dashboard Common Features

This guide covers features commonly available across dashboards. You can narrow data scope with lookup time and agent filters, manipulate charts with widget options, and save and manage widget configurations with presets.

Query Time Setting

The dashboard basically provides the real-time monitoring feature but the past data can be viewed using the time selector at the top of the screen. In the time selector, select **||** and then set the following time. It updates the data in widgets placed on the dashboard based on the set time.

Agent Check

Checking the agent connection status

On the upper left of the screen, the right area of the time selector provides information that allows you to check the status of agents connected to the project. This allows you to immediately check whether the target application server is running.

- **Total**: Number of all agents connected to the project
- **Active**: Number of active agents
- **Inactive**: Number of inactive agents
- : Can display or hide inactive agents.

Agent-based monitoring

- ✓ To select all agents again while one or more agents is selected, deselect them or click **Total**.

Searching the agent

If the agents connected to the project are numerous and difficult to find, use the agent search function. If you click  **Search** on the upper right of the screen, a box appears where you can enter a string.

The agents that match the entered string appear in the list. To filter the data in the dashboard based on the searched agents, click **Select** on the right of the text entry box.

To reset the selected agent entries, click **Close** on the right of the input field.

Widget Options

The icons displayed on the widget function as follows:

 The provided options may differ depending on the widget.

Widget Edit

The widgets on the dashboard can be adjusted to the desired size and placed in the desired positions. You can also delete unnecessary widgets or add them again.

Resizing a widget

Click and hold the mouse on the  element at the lower right of the widget, and then drag it to the desired size. Grids with a uniform horizontal-to-vertical ratio appear and the size of the widget can be adjusted in each grid.

Moving a widget

When you move the mouse cursor to the top of the widget, the cursor shape changes to . At this time, you can move the widget by dragging with your left mouse button and move to the desired position.

Deleting a widget

Right-click on a widget to delete. If you select **Delete**, the widget is deleted from the dashboard.

Adding a widget

Move the mouse cursor to an empty space on the dashboard and then right-click on it. Select a widget to add from the pop-up menu. Place the widget to the desired position and then resize it.

Preset Setting

Creating a new preset

1. Place the widgets in the desired format on the dashboard. You can also resize and place only the widgets to be frequently checked.
2. On the upper right of the screen, select .
3. Enter a new preset name.

To save the selected agent data, select **Include agent selection history**.

4. Select **Save**.

You can see the newly saved preset from the preset list.

- If any changes are made to the newly created preset, save the preset again. Select  and then save the preset with the same name. Any changes are overwritten in the existing preset.
- If you go to another menu without saving changes on the dashboard, the changes are not saved.
- Presets are saved on a project basis and can be shared with other users.

Deleting presets

If you have unused presets, you can delete them from the preset list. Select  on the right of the item to delete

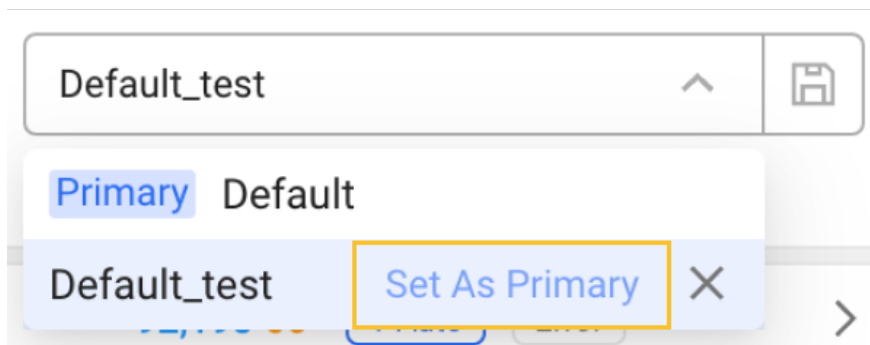
from the preset list.

! The **Default** preset cannot be deleted.

Changing the default preset

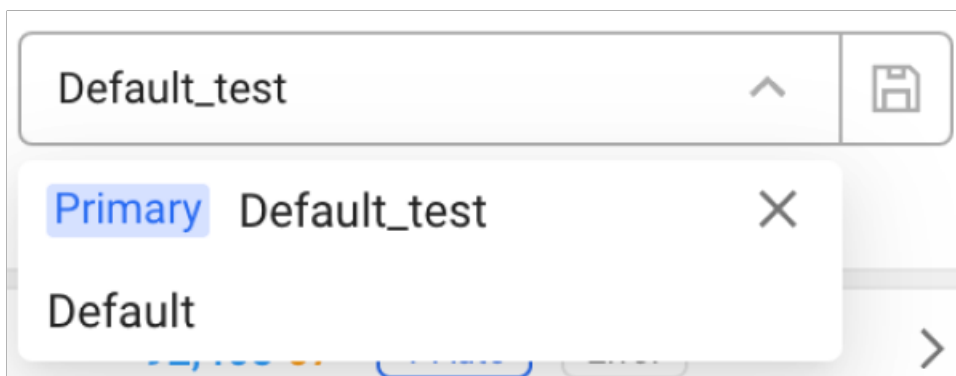
You can change the preset displayed when you first enter the **Application Dashboard** menu.

1. Check the preset selection box on the upper right of the dashboard.
2. Hover your mouse over a desired preset to set it as default.



3. If the **Set as primary** button appears, select it.

The selected preset is changed to the default preset, and the **Primary** tag appears in the preset list.



Real-time Alert

If you select  **Alert Status** on the upper right of the screen, the recent events appear. You can turn on or off the browser notifications by clicking the toggle menu.

ⓘ For more information about basic elements in the fixed menu area at the top of the screen, see [the following](#).

LLM Dashboard Widget Reference

This document is a reference that organizes the widgets provided by the LLM Dashboard by dashboard area. Check the chart type, filters, and usage for each widget. For the full screen layout and default preset, see [LLM Dashboard](#).

Available Widgets

LLM Performance Metrics

Widget	Description
LLM API Request Count	Request count + error rate by time period
Request Count by Status Code	HTTP status code (2xx/4xx/5xx) distribution
Latency Average	Average response time trend by time period
Latency Percentile	Latency p50/p75/p95/p99 trend
Latency Percentile Individual Widgets	Individual widget per percentile
Latency Distribution Comparison	Latency box plot by model
TTFT Average	TTFT average trend by time period
TTFT Percentile	TTFT p50/p75/p95/p99 trend
TTFT Percentile Individual Widgets	Individual widget per percentile
TTFT Distribution Comparison	TTFT box plot by model
TPOT Average	TPOT average trend by time period

Widget	Description
TPOT Percentile	TPOT p50/p75/p95/p99 trend
TPOT Percentile Individual Widgets	Individual widget per percentile
TPOT Distribution Comparison	TPOT box plot by model
Overall Performance Comparison	TTFT/TPOT/Output TPS comparison by model
Output Token Throughput	Output tokens per second (tok/s)

Tokens & Cost

Widget	Description
Token Usage	Input/output token trend by time period
Average Tokens per Request	Average input/output tokens per request
Token Usage Distribution	Input/output token total comparison by model
Today's Token Usage	Today's cumulative tokens + comparison with previous day
Cache Hit Rate	Cache token ratio (%) trend
Cache Hit Rate Distribution	Cache hit rate comparison by model
Cache Cost Savings	Cost saved through caching (\$) trend
Cache Cost Savings Distribution	Cache cost savings comparison by model
Cost Usage	Input/output cost (\$) trend by time period
Average Cost per Request	Average cost per request

Widget	Description
Cost Usage Distribution	Cost total comparison by model
Today's Cost	Today's cumulative cost + comparison with previous day

Model Comparison & Analysis

Widget	Description
Overall Performance Comparison	TTFT/TPOT/Output TPS comparison by model
Token Usage Distribution	Input/output token total comparison by model
Error Rate Distribution	Error rate (%) comparison by model
Error Count	API error and program error count by time period
Request Count by Status Code	HTTP status code (4xx/5xx) distribution

Live Observation

Widget	Description
Active LLM API	Running LLM API calls and the long-running list
Active Trajectory	Number of trajectories in progress and the distribution by level

Widget Details

LLM API Requests

A widget group for monitoring LLM API request volume and status codes.

LLM API Request Count

Property	Value
Chart Type	Bar chart (request count) + Line chart (error rate) combo
Filter	Tag filter
Action	View individually, View merged, View details

Displays request count and error rate (%) by time period for LLM API calls. The bar chart (left Y-axis) shows request volume, and the line (right Y-axis) shows error ratio against total requests.

- If request volume spikes with a simultaneous error rate increase, Rate Limit exceeded or provider outage is likely.
- If request volume is unchanged but error rate rises, an issue may be occurring in a specific model or agent. Use **View individually** to narrow down the cause.

Request Count by Status Code

Property	Value
Chart Type	Stacked bar chart
Filter	None
Action	View details

Displays HTTP status code distribution by time period, divided into 2xx (success), 4xx (client error), and 5xx (server error).

- **4xx increase** — Check for invalid request parameters, authentication failures, or token limit exceeded.
- **5xx increase** — Indicates LLM provider server failure or temporary overload.
- Cross-reference with the **LLM API Request Count** widget's error rate for quick error classification.

LLM API Response Performance

A widget group for analyzing LLM API overall response time (Latency) from various perspectives. Latency is the total elapsed time from when a request is sent to when the response is fully completed.

Latency Average

Property	Value
Chart Type	Line chart
Filter	Tag filter
Action	View individually, View merged, View details

Displays the Latency average trend over time.

- If Latency average rises overall, check for model response speed degradation or network delays.
- Use **View individually** to compare Latency by model and identify which model is causing delays.

Latency Percentile

Property	Value
Chart Type	Line chart (p50, p75, p95, p99 — 4 series)
Filter	None
Action	View details

Displays Latency percentile trends over time in four series: p50, p75, p95, and p99. For example, a p95 value of 3 seconds means 95% of all requests completed within 3 seconds.

- A large gap between p50 and p99 indicates most requests are fast but some experience severe delays.
- If p50 and p99 rise together, it indicates overall performance degradation.

Latency Percentile Individual Widgets

Property	Value
Chart Type	Line chart
Filter	Tag filter
Action	View individually, View merged, View details

Provides each percentile as an individual widget. The meaning of each percentile is as follows:

Percentile	Description
p50	Half of all requests completed within this time. Represents average response speed perceived by typical users.
p75	75% of all requests completed within this time. Gap from p50 reveals additional delay in top 25% of requests.
p95	95% of all requests completed within this time. Frequently used as SLA (Service Level Agreement) baseline metric.
p99	99% of all requests completed within this time. Represents the slowest 1% of requests for system stability assessment.

Latency Distribution Comparison

Property	Value
Chart Type	Box plot (x-axis: model)
Filter	Tag filter (always visible)
Action	View details

Compares Latency distribution by tag (model, etc.) using a box plot. The box area represents p25–p75 (middle 50% range), the center line is the median, and whiskers show min–max values.

- Models with wide boxes have high response time variance and may need stability improvements.
- Models with long whiskers experience occasional extreme delays.
- Compare distributions side by side to identify the most stable model.

TTFT (Time To First Token)

TTFT is the time from when a request is sent until the first response token arrives. As TTFT increases, users perceive that "the response hasn't started yet."

TTFT Average

Property	Value
Chart Type	Line chart
Filter	Tag filter
Action	View individually, View merged, View details

Displays the TTFT average trend over time.

- If TTFT spikes at a specific time, check for increased model queue wait or provider-side delay.

- Use **View individually** to compare which model has slower initial responses.

TTFT Percentile

Property	Value
Chart Type	Line chart (p50, p75, p95, p99 — 4 series)
Filter	None
Action	View details

Displays TTFT percentile trends over time in four series: p50, p75, p95, and p99. For example, a p95 value of 2 seconds means 95% of all requests received the first token within 2 seconds.

- If p50 is stable but p99 spikes, it signals intermittent queue wait occurrences.
- If all percentiles rise together, overall initial response delay is occurring.

TTFT Percentile Individual Widgets

Property	Value
Chart Type	Line chart
Filter	Tag filter
Action	View individually, View merged, View details

Provides each percentile as an individual widget.

Percentile	Description
p50	Initial response wait time perceived by most users.
p75	Gap from p50 reveals additional initial delay in top 25% of requests.

Percentile	Description
p95	SLA baseline metric: "This initial response wait is guaranteed for most users."
p99	Initial response time of the slowest 1%. Check for intermittent timeouts or delays under specific conditions.

TTFT Distribution Comparison

Property	Value
Chart Type	Box plot (x-axis: model)
Filter	Tag filter (always visible)
Action	View details

Compares TTFT distribution by tag (model, etc.) using a box plot. The box area represents p25–p75, the center line is the median, and whiskers show min–max values.

- Models with low median and narrow box provide the most stable and fast initial responses.
- Models with long whiskers experience occasional extreme initial delays.

TPOT (Time Per Output Token)

TPOT is the average time to generate one output token. Lower TPOT results in smoother streaming responses; higher TPOT may cause a "stuttering response" perception.

TPOT Average

Property	Value
Chart Type	Line chart
Filter	Tag filter

Property	Value
Action	View individually, View merged, View details

Displays the TPOT average trend over time.

- If TPOT spikes at a specific time, check for increased model load or Rate Limit impact.
- Use **View individually** to compare token generation speed by model.

TPOT Percentile

Property	Value
Chart Type	Line chart (p50, p75, p95, p99 — 4 series)
Filter	None
Action	View details

Displays TPOT percentile trends over time in four series: p50, p75, p95, and p99. For example, a p95 value of 40ms means 95% of all requests generated each token within 40ms.

- If p50 is stable but p99 rises, it signals intermittent token generation stalls.
- If all percentiles rise together, it indicates overall streaming speed degradation.

TPOT Percentile Individual Widgets

Property	Value
Chart Type	Line chart
Filter	Tag filter
Action	View individually, View merged, View details

Provides each percentile as an individual widget.

Percentile	Description
p50	Typical streaming speed. Identify models providing the smoothest streaming.
p75	Gap from p50 reveals streaming speed degradation in top 25% of requests.
p95	SLA baseline streaming speed metric.
p99	Token generation speed of the slowest 1%. Check for intermittent stall issues.

TPOT Distribution Comparison

Property	Value
Chart Type	Box plot (x-axis: model)
Filter	Tag filter (always visible)
Action	View details

Compares TPOT distribution by tag (model, etc.) using a box plot. The box area represents p25–p75, the center line is the median, and whiskers show min–max values.

- Models with low median and narrow box provide the most stable streaming performance.
- Models with long whiskers experience occasional significant token generation slowdowns.

Overall Performance Comparison

Property	Value
Chart Type	Horizontal bar chart (per model, 3 metrics)
Filter	Tag filter (always visible)
Action	View details, p95/p99 tab toggle

Compares three key performance metrics per model in a horizontal bar chart. Toggle between p95/p99 tabs to check typical performance and worst-case performance respectively.

Metric	Description	Direction
TTFT	Time users wait for the first response	Lower is better
TPOT	Time to generate one token (streaming speed)	Lower is better
Output TPS	Output tokens generated per second (overall throughput)	Higher is better

By combining all three metrics, you can identify at a glance models that "start quickly, stream smoothly, and deliver high throughput."

Tokens

A widget group for monitoring token usage patterns and cache efficiency.

Token Usage

Property	Value
Chart Type	Line chart (input, output — 2 series)
Filter	Tag filter
Action	View individually, View merged, View details

Displays input token count and output token count trends over time. Input tokens are tokens included in the prompt; output tokens are tokens generated by the model as responses.

- If token usage spikes at a specific time, check for traffic increase, prompt changes, or abnormal request influx.
- Use **View individually** to identify which model or agent is consuming the most tokens.

Average Tokens per Request

Property	Value
Chart Type	Line chart (input, output — 2 series)
Filter	Tag filter
Action	View individually, View merged, View details

Displays the trend of average input/output tokens used per request over time.

- If average input tokens per request spike, prompts may have gotten longer or long contexts may have been included.
- If average output tokens per request spike, response lengths have increased, directly impacting cost.

Output Token Throughput

Property	Value
Chart Type	Line chart
Filter	Tag filter
Action	View individually, View details

A throughput metric showing output tokens generated per second (Tokens Per Second).

- Periods with decreased throughput should be checked for model response bottlenecks or Rate Limit impact.
- Cross-reference with token usage to identify periods where "tokens are consumed heavily but processing speed is slow."

Token Usage Distribution

Property	Value
Chart Type	Histogram (grouped by tag)
Filter	Tag filter (always visible)
Action	View details

A distribution chart comparing input/output token totals by tag (model, etc.).

- Quickly identify which model consumes the most tokens.
- Models with high input ratio have room for cost reduction through prompt optimization.
- Models with high output ratio may benefit from adjusting response length limits (`max_tokens`).

Today's Token Usage

Property	Value
Chart Type	Daily comparison chart (today/previous day)
Filter	None
Action	Date comparison

Displays today's cumulative token usage (input + output) and change compared to the same time on the previous day.

- If token consumption rate is faster than the previous day, check for traffic increase or prompt size changes.
- Useful for proactively detecting Rate Limit thresholds or daily budget overruns.

Cache Hit Rate

Property	Value
Chart Type	Line chart (%)
Filter	Tag filter
Action	View individually, View details

Displays the ratio (%) of tokens retrieved from cache out of total input tokens over time.

- Higher hit rate means prompt caching is working effectively with significant cost savings.
- If hit rate drops suddenly, check for prompt content changes, cache TTL expiration, or new request type influx.
- Use **View individually** to compare cache efficiency by model.

! Cache Hit Rate = (sum of `cached_tokens` in period / sum of `input_tokens` in period) × 100

Cache Hit Rate Distribution

Property	Value
Chart Type	Histogram (by tag)
Filter	Tag filter (always visible)
Action	View details

A distribution chart comparing cache hit rates (%) by tag (model, etc.).

- Models with high hit rate have well-utilized caching; models with low hit rate need prompt structure review for caching targets.
- If hit rate gaps between models are large, differentiate caching strategies per model for additional cost

savings.

Cache Cost Savings

Property	Value
Chart Type	Line chart (\$)
Filter	Tag filter
Action	View individually, View details

Displays the actual cost savings (\$) from prompt caching over time. Cached tokens are charged at a lower rate than regular input tokens, and this difference is calculated as cost savings.

- Consistent cost savings indicate the caching strategy is working effectively.
- If cost savings decrease, cross-reference with cache hit rate to check for increased cache misses.
- Use **View individually** to identify which model yields the highest savings.

Cache Cost Savings Distribution

Property	Value
Chart Type	Histogram (by tag)
Filter	Tag filter (always visible)
Action	View details

A distribution chart comparing cache cost savings (\$) by tag (model, etc.).

- Models with high savings benefit from caching, so maintain or strengthen their caching configuration.
- Models with low savings should be reviewed alongside cache hit rate to improve prompt structure for caching or adjust cache TTL.

Cost

A widget group for monitoring LLM API usage cost.

Cost Usage

Property	Value
Chart Type	Line chart (input, output — 2 series)
Filter	Tag filter
Action	View individually, View merged, View details

Displays input/output cost (\$) trends over time. Input cost is prompt processing cost; output cost is response generation cost.

- If cost spikes at certain times, check request volume and tokens per request for that period.
- Compare the ratio of input cost to output cost to decide whether to focus on prompt optimization or response length limits.
- Use **View individually** to compare cost contribution by model.

Average Cost per Request

Property	Value
Chart Type	Line chart (input, output — 2 series)
Filter	Tag filter
Action	View individually, View merged, View details

Displays the trend of average input/output cost (\$) per request over time.

- If cost per request changes suddenly, check for model switching, provider pricing changes, or prompt size changes.

- If total cost increases but cost per request is unchanged, request volume has simply increased — focus on request volume management.

Cost Usage Distribution

Property	Value
Chart Type	Histogram (grouped by tag)
Filter	Tag filter (always visible)
Action	View details

A distribution chart comparing input/output cost totals by tag (model, etc.).

- Quickly identify which model has the highest cost.
- Check usage frequency and tokens per request for high-cost models to select cost optimization targets.

Today's Cost

Property	Value
Chart Type	Daily comparison chart (today/previous day)
Filter	None
Action	Date comparison

Displays today's cumulative cost (\$) and change compared to the same time on the previous day.

- If cost consumption rate is faster than the previous day, check for request volume increase, model switching, or prompt size changes.
- Useful for proactively detecting daily budget overruns.

Errors

A widget group for monitoring LLM API error status.

Error Rate Distribution

Property	Value
Chart Type	Horizontal bar chart
Filter	Tag filter
Action	View individually, View merged, View details

Compares error rates (%) by tag (model, etc.) in a horizontal bar chart. Error rate is the ratio of errors (`status_code` 400 or above) to total requests in the group.

- Quickly identify whether errors are concentrated in a specific model or agent.
- Models with high error rates should be checked for provider outage, invalid parameters, or Rate Limit exceeded.
- Use **View individually** to narrow down error causes by `Agent` , `Provider` , and other criteria.

Error Count

Property	Value
Chart Type	Stacked bar chart
Filter	None
Action	View details

Displays error counts by time period, divided into API errors and program errors.

- **API Error** — Errors returned by the LLM provider, including Rate Limit exceeded, server errors, and authentication failures.

- **Program Error** — Client-side errors, including response parsing failures, timeouts, and connection errors.
- Error type trends help quickly distinguish whether the issue is on the provider side or client side.

Live Observation

This widget group lets you check the LLM calls and trajectories that are running right now. A trajectory is the unit that groups the transactions participating in a single LLM workflow. The list screens of both widgets use running data queried directly from the agent, so there is no past range. If the dashboard is paused or you are viewing a past time, no path to the list is provided.

Active LLM API

Property	Value
Chart Type	Swarm chart (track per model)
Filter	Dashboard agent filter
Action	View details, Open in new window

Displays the LLM API calls that are currently running as dots on a track for each model. One dot is one call, and the horizontal position indicates the **Normal**, **Slow**, or **Very Slow** range. Data for the three ranges is aggregated according to the values set in the agent options.

- If dots pile up in the **Very Slow** range, check whether calls running longer than the criteria set in the agent are concentrated in a specific model or provider.
- Clicking the widget body opens the long-running list in a modal. Expanding the prompt lets you check the system prompt and input prompt that are running, and selecting a call in the list moves to the trajectory view that the call belongs to, where you can check the entire execution flow.
- Clicking the **View details** button in the widget header opens the same list on a wide screen. Clicking the **Open in new window** button in the modal header opens it in a separate window.

Active Trajectory

Property	Value
Chart Type	Orbit chart + count list by level
Filter	Dashboard agent filter
Action	View details, Open in new window

Displays the number of currently active trajectories and the number of steps of each trajectory, divided into the **Light**, **Middle**, and **Heavy** levels. A level indicates how many times and how repeatedly steps were called; it does not mean speed or execution time.

- If even one of the multi-transactions it belongs to is running, that trajectory is counted as live.
- The level boundaries follow the criteria set in the agent options.
- Clicking a level opens the trajectory list filtered by that level. Selecting a row in the list leads to the trajectory view.
- Clicking the orbit chart opens the entire list without the level filter.

Trajectory List

The list consists of a group header for each trajectory and the transaction rows below it. The group header shows **mtid**, the number of grouped transactions, the user, and the session. A standalone flow with no **mtid** is displayed as a standalone run. You can export the list as CSV.

Column	Description
Transaction	Transaction name
Status	Transaction progress status (Running, Completed, Failed)
mtid	Key that groups a trajectory
Steps	Number of steps the transaction performed

Column	Description
Agent	Agent that performed the transaction
LLM Calls	Number of LLM calls that occurred within the transaction
Elapsed	Time elapsed since the transaction started
User	<code>user_id</code> attached to the log
Session	<code>session_id</code> attached to the log

Reference

Collected Fields

Field	Description
<code>request_count</code>	LLM API call count
<code>status_code</code>	HTTP response status code (2xx, 4xx, 5xx)
<code>latency</code>	Request start to response completion total time (ms)
<code>ttft</code>	Time To First Token. Request start to first token received (ms)
<code>tpot</code>	Time Per Output Token. Average interval between output tokens (ms)
<code>input_tokens</code>	Total input token count (including cached)
<code>output_tokens</code>	Output token count
<code>cached_tokens</code>	Input tokens retrieved from cache (subset of <code>input_tokens</code>)
<code>input_tokens_cost</code>	Total input token cost (\$). Includes <code>cached_tokens_cost</code>

Field	Description
<code>output_tokens_cost</code>	Output token cost (\$)
<code>cached_tokens_cost</code>	Cached token cost (\$)
<code>error_type</code>	Error type (<code>api_error</code> , <code>program_error</code>)

Widget Group Summary

Group	Widgets	Description
LLM API Requests	2	Request volume and status code monitoring
LLM API Response Performance	8	Latency trend and distribution analysis
TTFT	7	Time to first token analysis
TPOT	7	Token generation speed analysis
Tokens	9	Token usage patterns and cache efficiency analysis
Cost	4	LLM API cost monitoring
Errors	2	Error status analysis by type
Live Observation	2	Status of running LLM calls and trajectories
Total	41	

Terminology

Term	Description
<code>Latency</code>	Total elapsed time from when an LLM API request is sent to when the response is fully completed (ms)

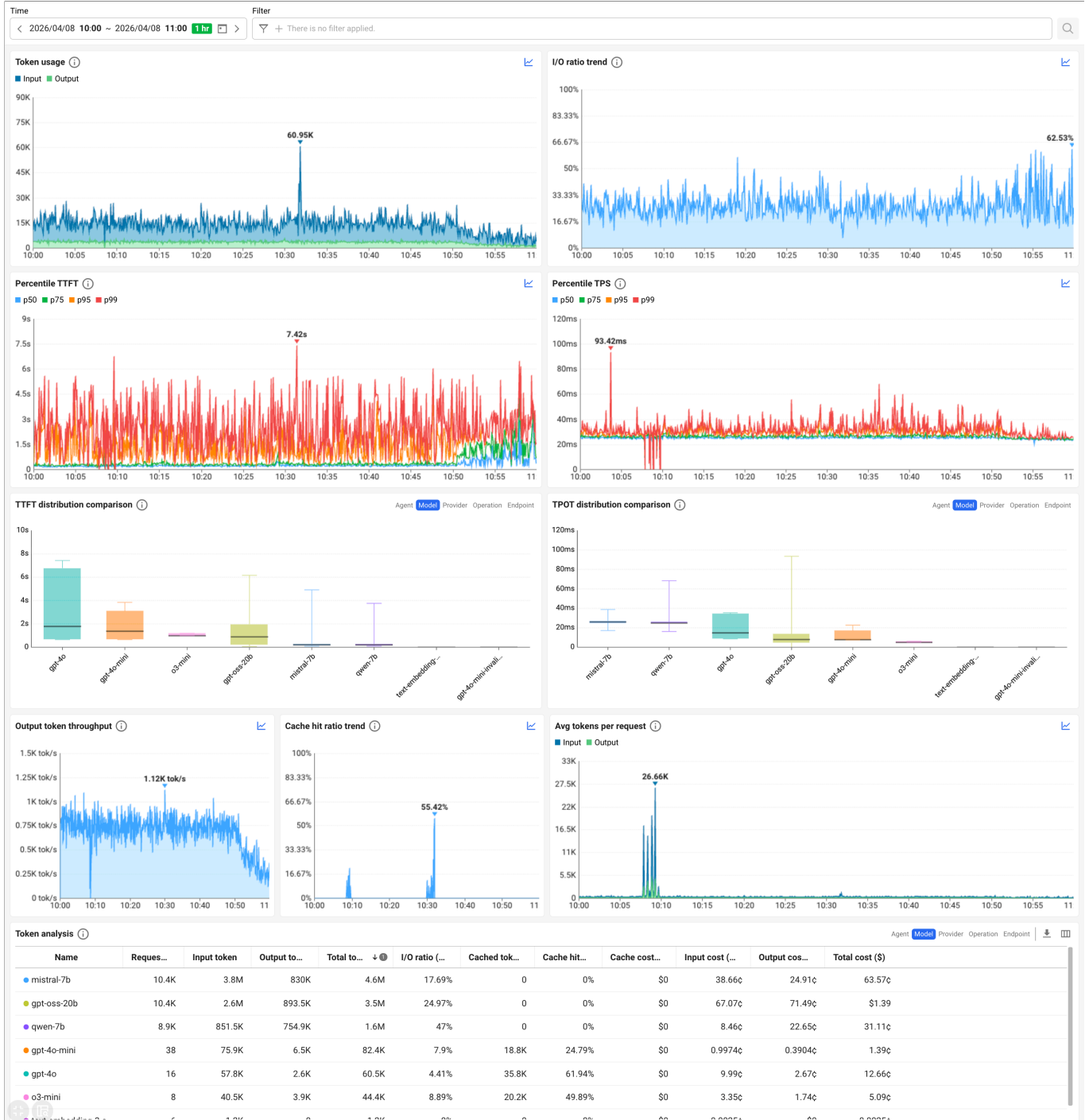
Term	Description
TTFT	Time To First Token. Time from when a request is sent until the first response token arrives (ms)
TPOT	Time Per Output Token. Average time to generate one output token (ms)
Output TPS	Tokens Per Second. Output tokens generated per second (tokens/s)
Percentile	The value below which a given percentage of data falls when sorted in order. Example: p95 = 95% of all data is at or below this value
Cache Hit Rate	Ratio of tokens retrieved from cache out of total input tokens (%). Higher values mean greater cost savings
Tag Filter	Feature to filter data by Agent, Model, Provider, Operation criteria
Preset	Saved widget configuration (type, position, size). Custom presets can be created
Trajectory	Unit that groups the transactions taking part in one LLM workflow by mtid

LLM Token Trend Analysis

Home > Select Project > **Analysis** > **Token Trend**

Token Trend analyzes LLM API token usage patterns and processing efficiency on a time-series basis. While the dashboard focuses on real-time status monitoring, this page is used for **trend comparison across time ranges** and **detailed analysis across models/agents**.

Set the time range, filters, and search conditions in the top option bar, then query widget data. You can narrow the range by dragging on charts, or navigate to the Prompt Log page by clicking on charts/boxplots.



Token Usage

Property	Value
Chart Type	Line chart (Input, Output -- 2 series)
Filter	Tag filter
Action	View individually, View merged, View details

Displays the trend of Input token count and Output token count by time. Input tokens are tokens included in the prompt; Output tokens are response tokens generated by the model.

- If token usage spikes at a specific time, check for traffic changes, prompt modifications, or abnormal request influx.
- Changes in the Input/Output ratio reveal shifts in workload characteristics (input-heavy vs. generation-heavy).
- Use **View individually** to identify which model or agent concentrates token consumption.

I/O Ratio Trend

Property	Value
Chart Type	Line chart (% , area fill)
Filter	Tag filter
Action	View individually, View merged, View details

I/O Ratio is the percentage (%) of Output tokens out of total tokens (Input + Output). A 50% baseline (dotted line) is displayed.

- **Above 50%:** Response-generation-heavy workload. Tasks with long outputs such as summarization or code generation are dominant.

- **Below 50%:** Input-heavy workload. Tasks with long prompts such as classification, embedding, or RAG retrieval are dominant.
- A sudden ratio change may indicate workload pattern or prompt structure modifications -- investigate accordingly.

! I/O Ratio = $\text{sum of output_tokens} / (\text{sum of input_tokens} + \text{sum of output_tokens}) \times 100$

TTFT Percentiles

Property	Value
Chart Type	Line chart (p50, p75, p95, p99 -- 4 series)
Filter	Tag filter
Action	View individually, View merged, View details, Select percentile

Displays the percentile trend of TTFT (Time To First Token). If TTFT rises alongside a spike in token usage, it indicates that increased token volume is affecting initial response latency.

- If p50 is stable but p99 spikes, it signals intermittent queue wait or provider delays.
- When an anomalous interval is found, use the TTFT distribution comparison to identify which model is the cause.

TPOT Percentiles

Property	Value
Chart Type	Line chart (p50, p75, p95, p99 -- 4 series)
Filter	Tag filter

Property	Value
Action	View individually, View merged, View details, Select percentile

Displays the percentile trend of TPOT (Time Per Output Token). Viewing alongside TTFT Percentiles helps distinguish between "slow first token and slow generation" intervals and "fast first token but slow generation" intervals.

- A p99 spike means intermittent requests where token generation stalls or significantly slows down.
- When an anomalous interval is found, use the TPOT distribution comparison to identify which model is the cause.

TTFT Distribution Comparison

Property	Value
Chart Type	Boxplot (x-axis: model)
Filter	Tag filter (always visible)
Action	View details

Compares the distribution of TTFT by tag (model, etc.) using boxplots. The box area represents p25-p75 (middle 50% range), the center line is the median, and whiskers indicate min-max values.

- Models with a low median and narrow box provide the most stable and fast initial response.
- Models with long whiskers experience intermittent extreme initial delays.
- Click a model to navigate to its prompt logs.

TPOT Distribution Comparison

Property	Value
Chart Type	Boxplot (x-axis: model)
Filter	Tag filter (always visible)
Action	View details

Compares the distribution of TPOT by tag (model, etc.) using boxplots.

- Models with a low median and narrow box provide the most stable streaming performance.
- Models with long whiskers experience intermittent significant slowdowns in token generation.

Output Token Throughput

Property	Value
Chart Type	Line chart
Filter	Tag filter
Action	View individually, View merged, View details

A throughput metric representing output tokens generated per second (Tokens Per Second).

- For intervals where throughput drops, check for model response bottlenecks or Rate Limit effects.
- Viewing alongside token usage helps identify intervals where "many tokens are used but processing speed is slow."
- Use **View individually** to compare throughput by model.

Cache Hit Rate Trend

Property	Value
Chart Type	Line chart (% , area fill)
Filter	Tag filter
Action	View individually, View merged, View details

Displays the percentage (%) of tokens retrieved from cache out of total input tokens, by time.

- Higher hit rates indicate effective prompt caching with greater cost savings.
- If the hit rate drops suddenly, check for prompt content changes, cache TTL expiration, or new types of request influx.
- Use **View individually** to compare cache efficiency by model and identify where caching is most effective.

! Cache Hit Rate = (sum of cached_tokens / sum of input_tokens) x 100

Average Tokens per Request

Property	Value
Chart Type	Line chart (Input, Output -- 2 series, area fill)
Filter	Tag filter
Action	View individually, View merged, View details

Displays the time-based trend of average Input/Output token count per request.

- A spike in Input tokens per request may indicate longer prompts or inclusion of longer context.

- A spike in Output tokens per request means longer responses, directly impacting cost increases.
- If this widget's values increase alongside token usage, it means request size has grown. If only token usage increases, it simply means request count has risen.

Token Analysis

Property	Value
Chart Type	Table
Filter	Agent / Model / Provider tab switch
Action	View details, CSV download, Column settings

Organizes key token usage, cache, and cost metrics by tag (Agent, Model, Provider) in a table.

Column	Description
Name	Tag value (agent name, model name, provider name)
Requests	Request count
Total Input	Total input tokens
Total Output	Total output tokens
Total Tokens	Total token count
I/O Ratio (%)	Output token ratio
Cached Tokens	Total cached tokens
Cache Hit (%)	Cache hit rate
Cached Cost (\$)	Cached token cost

Column	Description
Input Cost (\$)	Input cost
Output Cost (\$)	Output cost
Total Cost (\$)	Total cost

- Items with high **Total Tokens** account for the largest share of total cost and are priority optimization targets.
- Items with high **I/O Ratio** are output-heavy workloads, significantly affected by Output token pricing.
- Items with low **Cache Hit** should have their caching strategy reviewed for potential cost savings.
- Sort by **Total Cost** to quickly determine cost optimization priorities.
- Use the download button in the upper right to export current data as CSV.
- Use column settings to select which columns to display.

Reference

Difference from Dashboard

Item	LLM Dashboard	Token Trend
Time Mode	Real-time (Live) + past point	Past range (Range) only
Primary Purpose	Real-time status monitoring	Trend analysis across time ranges
Filter	Agent selection	Complex condition filter (Agent, Model, Provider, Operation, URL)
Prompt Log Integration	None	Navigate to Prompt Log on chart click

Item	LLM Dashboard	Token Trend
Data Export	None	CSV download supported

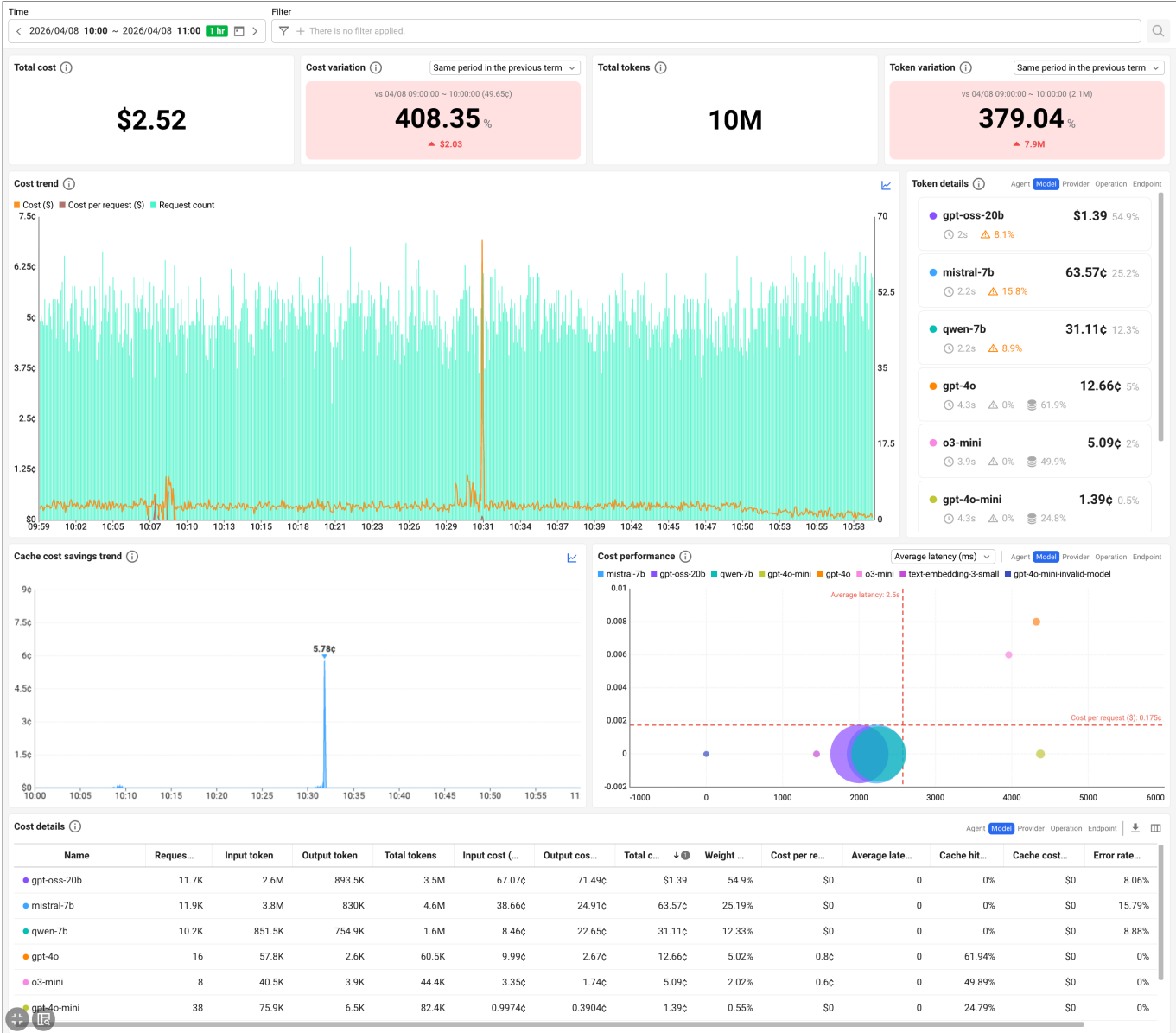
LLM Cost Analysis

Home > Select Project > **Analysis** > **Cost Analysis**

Cost Analysis analyzes LLM API usage costs from multiple angles. Use it to understand cost status, compare costs by model, analyze cost efficiency relative to performance, and identify optimization targets.

The screen is organized top to bottom in the order **Summary** > **Trend/Detail** > **Cache/Performance Efficiency** > **Table**, allowing you to naturally progress from understanding the current status to selecting optimization targets.

Set the time range, filters, and search conditions in the top option bar, then query widget data. In the summary cards, you can select a comparison period (**Previous Same Period** or **Same Period Last Week**), and drag on charts to narrow the time range.



Summary Cards

Cards that show key metrics for the selected time range at a glance.

Total Cost

Displays the total cost (\$) incurred within the selected time range. This is the sum of Input cost and Output cost.

- As the starting point of cost analysis, you can grasp the overall cost scale at a glance.
- Viewing alongside cost change shows how costs have shifted compared to the previous period.

Cost Change

Displays the cost change rate (%) and change amount (\$) compared to the previous period.

- **Positive (▲ red):** Costs increased. Check for increased request volume, higher usage of expensive models, or prompt size changes.
- **Negative (▼ blue):** Costs decreased. This may indicate optimization effects or traffic reduction.

Total Tokens

Displays the total number of tokens used (Input + Output) within the selected time range.

- Viewing alongside total cost helps determine whether cost increases are due to token usage or expensive model usage.
- If tokens increased but costs remain similar, it means the transition to cheaper models is working well.
- Click to navigate to the Token Trend page.

Token Change

Displays the token usage change rate (%) and change count compared to the previous period.

- Comparing with cost change helps identify the cause of cost fluctuations.
- **Token change rate > Cost change rate:** The proportion of cheaper models has increased.
- **Token change rate < Cost change rate:** The proportion of expensive models has increased.
- Click to navigate to the Token Trend page with the comparison period range.

Cost Trend

Property	Value
Chart Type	Bar chart (requests) + Line chart (cost, cost per request) combined
Filter	Tag filter
Action	View individually, View merged, View details

Displays the trend of requests, cost, and cost per request by time. Bar chart represents request count; lines represent cost (\$) and cost per request (\$).

- If requests and cost move together, it is a normal cost increase proportional to usage.
- If requests remain unchanged but cost rises, cost per request has increased -- check for higher proportion of expensive models or prompt size changes.
- A sudden spike in cost per request line may indicate model switching or provider pricing changes.

Token Detail

Property	Value
Chart Type	Card list (card per model)
Filter	Model / Provider / Operation / Agent tab switch
Action	View details

Organizes cost, share (%), average Latency, and error rate by tag (Model, Provider, Operation, Agent) in card format.

Each card displays the following information:

Property	Description
Model/Agent Name	Tag value
Cost	Total cost (\$) for the group
Share	Group cost / Total cost x 100 (%)
Avg Latency	Average Latency for the group
Error Rate	Group error count / Total request count x 100 (%)
Cache Hit Rate	Group cached_tokens / input_tokens x 100 (%)

- High-cost items are sorted at the top, allowing you to immediately identify models/agents with the highest cost contribution.
- Items with high error rates incur costs even on failed requests, so reducing errors has a direct cost-saving effect.

Cache Savings Trend

Property	Value
Chart Type	Line chart (\$)
Filter	Tag filter
Action	View individually, View merged, View details

Displays the time-based trend of savings (\$) achieved through prompt caching. Cached tokens are charged at a lower rate than regular input tokens, and the difference is calculated as the savings.

- Consistent savings indicate that the caching strategy is working effectively.
- If savings decrease, check whether cache hit rate has dropped.
- The ratio of savings to total cost shows the overall cost reduction contribution of caching.

Cost vs. Performance

Property	Value
Chart Type	Bubble chart (x-axis: performance metric, y-axis: cost per request, bubble size: request count)
Filter	Model / Provider / Operation / Agent tab switch
Action	View details, Select x-axis metric

Displays the relationship between performance and cost (cost per request) by model/agent as a bubble chart. Bubble size represents request count, and dotted lines indicate the overall average baseline.

Selectable x-axis metrics

Metric	Description
Avg Latency (ms)	Average response completion time
Avg TTFT (ms)	Average time to first token
Avg TPOT (ms)	Average time per output token
p95 Latency (ms)	95th percentile response time
p95 TTFT (ms)	95th percentile time to first token
p95 TPOT (ms)	95th percentile time per output token

Quadrant interpretation

Position	Meaning	Recommended Action
Bottom-left (fast and cheap)	Most cost-efficient area	Maintain or increase usage of this model

Position	Meaning	Recommended Action
Top-right (slow and expensive)	Area where cost optimization is most urgent	Consider alternative models or prompt optimization
Top-left (fast but expensive)	Good performance but high cost	Review whether the workload truly requires this performance level
Bottom-right (slow but cheap)	Low cost but insufficient performance	Suitable for workloads with low performance requirements

- Larger bubbles represent a greater share of total cost, so optimizing them yields the greatest impact.

Cost Details

Property	Value
Chart Type	Table
Filter	Model / Provider / Operation / Agent tab switch
Action	View details, CSV download, Column settings

Organizes key cost-related metrics by tag (Model, Provider, Operation, Agent) in a table.

Column	Description
Name	Tag value (model name, agent name, etc.)
Requests	Request count
Input Tokens	Input token count
Output Tokens	Output token count

Column	Description
Total Tokens	Total token count
Input Cost (\$)	Input cost
Output Cost (\$)	Output cost
Total Cost (\$)	Total cost
Share (%)	Cost share (group cost / total cost)
Cost/Req (\$)	Cost per request
Avg Latency	Average response time
Cache Hit (%)	Cache hit rate
Cached Cost (\$)	Cached token cost
Error Rate (%)	Error rate
Error Cost (\$)	Cost incurred from errors

- Focus on items with high **Total Cost (\$)** to identify cost optimization targets.
- Items with high **Cost/Req (\$)** indicate expensive models per request -- consider alternative models or prompt optimization.
- Items with high **Error Rate (%)** incur costs even on failed requests -- resolving errors has a direct cost-saving effect.
- **Error Cost (\$)** shows the exact amount wasted on errors.
- Use the download button in the upper right to export current data as CSV.
- Use column settings to select which columns to display.

Reference

Comparison Period Calculation

Comparison Mode	Calculation
Previous Same Period	Compares with data from a point equally far back as the current query period. Example: querying today 09:00-12:00 compares with 06:00-09:00
Same Period Last Week	Compares with data from the same time range exactly 7 days ago. Example: querying 4/6 09:00-12:00 compares with 3/30 09:00-12:00

Custom Token Pricing per Model

You can specify token rates per model directly. Values you specify take **precedence** over the agent's automatic calculation (the `genai-prices` package), so you can calculate costs for models that are not in the public rate table — such as self-hosted, fine-tuned, or proxied models — or correct the rate to the actual one when your contract price differs from the list price.

whatap.conf

```
llm_model_pricing=model|input_per_1m|output_per_1m|cached_per_1m
# llm_model_pricing=gpt-4o|2.50|10.00|1.25,gpt-4o-mini|0.15|0.60|0.075,my-custom-model|1.00|5.00
```

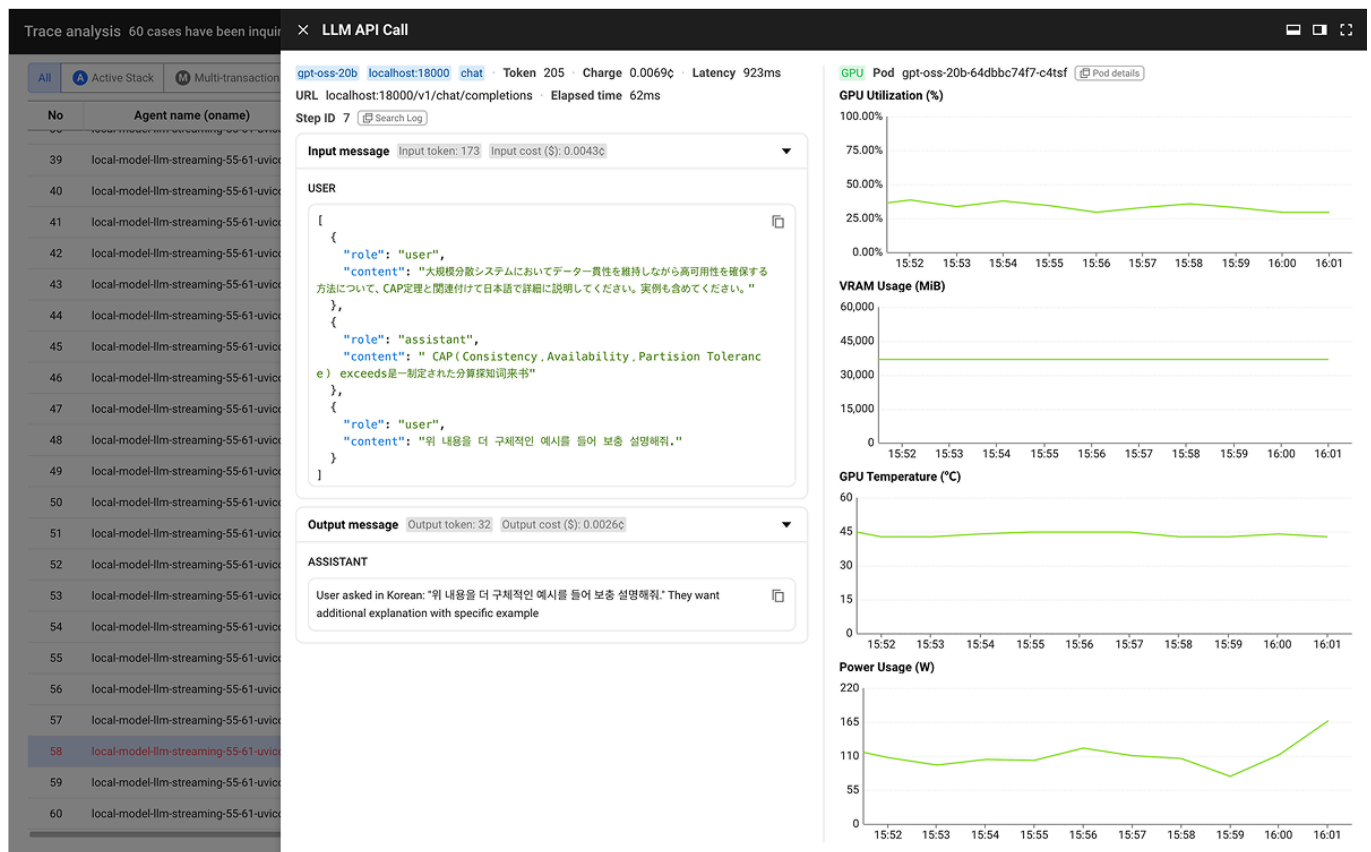
- Separate models with a comma (,) and fields with a pipe (|).
- The fourth field can be either omitted (model | 1.00 | 5.00) or left blank (model | 1.00 | 5.00 |).
- Configuration changes take effect immediately and do not require a restart.

LLM API Analysis (Transaction Linkage)

Home > Select Project > Analysis > LLM API Analysis

AI Agent Observability integrates with WhatTap APM to trace LLM API calls as part of application transactions. Clicking an LLM HTTPC step in a transaction profile opens the **LLM API Detail Drawer**, where you can view prompt input/output, token usage, cost, error information, and GPU infrastructure correlation all in one screen.

! Viewing prompt input/output requires log read permission.



How to Access

1. Select a transaction from the **hitmap** or **Transaction Search** on the LLM Dashboard.
2. In the transaction profile, click the LLM HTTPC step (HTTP external call made to an LLM provider URL).
3. The **LLM API Detail Drawer** opens on the right.

Summary and Prompt

This area shows key information about the LLM call at a glance and allows you to view the original prompt input/output.

Identity Tags

Displays the target model and provider information as tags.

Tag	Description	Example
Model	LLM model used	gpt-4o , claude-sonnet-4-20250514
Provider	LLM API provider	api.openai.com , api.anthropic.com
Operation	Operation type of the call	chat , completion

If the call failed (success=false), a red **Error** tag is additionally displayed.

Key Metrics

Displayed inline next to the Identity tags.

Metric	Description
Token	Total token count (Input + Output)

Metric	Description
Cost	Cost of the call (\$)
Latency	Time from request start to response completion (ms)

HTTP Information

Property	Description
URL	Call endpoint (host:port + path)
Elapsed	HTTP call elapsed time (ms)
Step ID	Unique identifier for this LLM call. Used as the key for navigating to log search.

Click the **Log Search** button next to **Step ID** to open the log explorer filtered by that Step ID in a new window.

Error Information

Displayed only when an error occurred in the call.

Property	Description
Error Class	Error class name (e.g., <code>RateLimitError</code> , <code>TimeoutError</code>)
Error Message	Detailed error message

Prompt Input (Input)

A collapsible section that displays the input messages sent to the LLM.

Header Badges

Badge	Description
Input Tokens	Input token count
Cached Tokens	Token count retrieved from cache (displayed only when applicable)
Input Cost (\$)	Input cost

Message Types

Label	Description
SYSTEM	System message. Instructions defining the model's role and behavior.
USER	User input message. The actual prompt content.

When messages are collected in chunks, they are automatically assembled in the correct order to display the complete text.

Model Response (Output)

A collapsible section that displays the response generated by the model.

Header Badges

Badge	Description
Output Tokens	Output token count
Reasoning Tokens	Reasoning token count (displayed only when applicable)
Output Cost (\$)	Output cost

Message Types

Label	Description
ASSISTANT	Text response from the model.
TOOL CALL	Tool call (Function Calling) requested by the model.
TOOL RESULT	Execution result of the tool call.

GPU Correlation

Displayed on the right side of the Drawer when a multi-transaction (mtid exists) is present. Shows real-time charts of GPU infrastructure status for the Pod that processed the LLM call, helping determine whether model response delays are caused by GPU resource shortages.

GPU Information Header

Property	Description
Pod	Kubernetes Pod name that performed the LLM inference
Pod Detail button	Click to open the Pod detail page in Kubernetes monitoring in a new window.

GPU Charts

Displays GPU metrics as line charts for a 5-minute window around the LLM call time.

Chart	Metric	Unit	Description
GPU Utilization	DCGM_FI_DEV_WEIGHTED_GPU_UTIL	%	GPU compute resource utilization. Near 100% indicates GPU saturation, which may cause LLM response delays.
VRAM Usage	DCGM_FI_DEV_FB_USED	MiB	GPU memory (Video RAM) usage. Used for model

Chart	Metric	Unit	Description
			loading and inference; insufficient memory causes OOM or swapping.
GPU Temperature	DCGM_FI_DEV_GPU_TEMP	°C	GPU temperature. Overheating triggers automatic throttling, degrading performance.
Power Usage	DCGM_FI_DEV_POWER_USAGE	W	GPU power consumption. Viewed alongside Utilization to gauge actual compute load.

! GPU correlation is displayed when all of the following conditions are met:

- The transaction is a multi-transaction (mtid exists)
- The Pod name that performed LLM inference is identified
- DCGM GPU metrics are being collected in the linked Kubernetes project

Analysis Scenarios

Identifying LLM Response Delay Causes

1. Click a transaction with long response time from the hitmap on the LLM Dashboard.
2. Check the **Elapsed** time of the LLM HTTPC step in the transaction profile.
3. Open the LLM API Detail Drawer and compare the **Latency** value with the **GPU Utilization** chart.
4. If GPU Utilization is high when Latency spikes, GPU resource shortage is the cause.
5. If GPU Utilization is low but Latency is high, suspect provider-side delays (queue wait, Rate Limit, etc.).

Reproducing Failed LLM Calls

1. Click the LLM HTTPC step with an error in the transaction profile.
2. Check the error type from **Error Class** and **Error Message** in the Drawer.

3. Review the SYSTEM and USER message originals in the **Prompt Input (Input)** section.
4. Use the collected prompt originals and model/parameter information to reproduce the same call and analyze the issue.

Analyzing High-Cost LLM Calls

1. Identify high-cost time periods on the Cost Analysis page.
2. Search transactions for that time period, open the profile, and click the LLM HTTPC step.
3. Check the **Token** count and **Cost** in the Drawer.
4. Check the Input Tokens badge in the **Prompt Input** section and review whether prompts are unnecessarily long.
5. If the **Cached Tokens** badge shows 0, caching was not applied to this call -- review whether the prompt is a caching candidate.

Reference

Data Collection Architecture

The LLM API Detail Drawer combines two data sources.

Data Source	Description	Usage
APM Trace	HTTP call information (URL, host, port, elapsed, error class/message), Step ID	Call identification in transaction profile and HTTP-level information display
#LlmCallLog Log	Model, provider, tokens, cost, success status, message content	Prompt original restoration and LLM-level metadata display

The two data sources are linked by **Step ID**.

Message Chunk Assembly

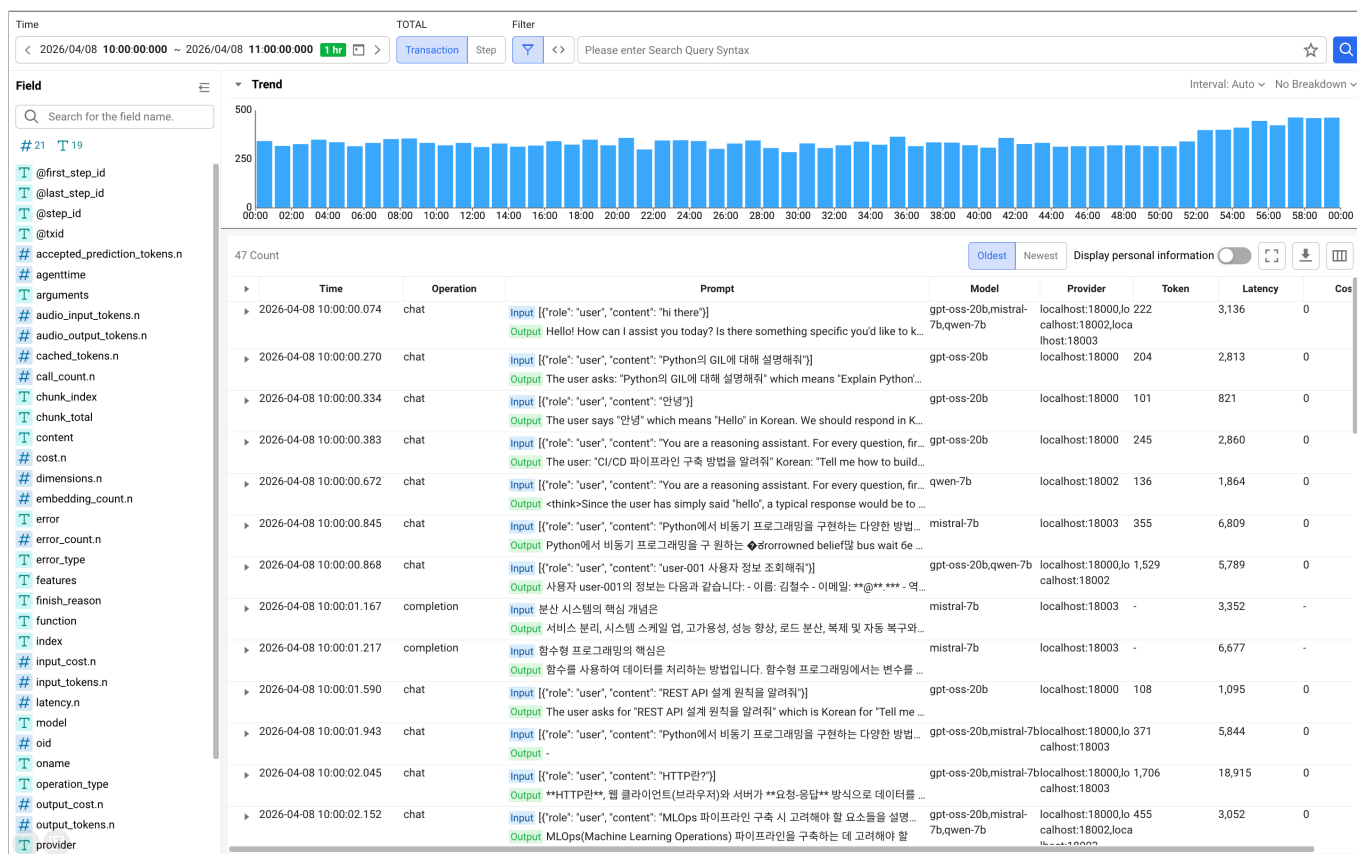
LLM prompts and responses can be long, so the server collects them split into multiple chunks. The Drawer assembles chunks in the correct order based on the `chunk_index` field to restore the complete message. Up to 100 chunks per type are supported.

LLM Prompt Log

Home > Select Project > Log > Prompt Log

Prompt Log is a menu for viewing and analyzing detailed logs of LLM API calls. You can check inputs (prompts), outputs (responses), token usage, costs, and performance metrics for each individual call. Use this menu to drill down and identify root causes after spotting anomalies in token trends or cost analysis.

ⓘ This menu requires **Log Read** permission. Without the permission, the menu is not accessible.



Screen Layout

Prompt Log consists of a three-panel layout: **top option bar**, **left filter panel**, and **right main area** (trend chart + log list).

Top Option Bar

- **Time range selector:** Specify start and end times. Quick select options are available (last 5 min, 10 min, 30 min, 1 hour, 3 hours, 6 hours, 12 hours). You can specify down to seconds and milliseconds.
- **Merge criteria:** Select the criteria for merging logs.
 - **Transaction:** Merges all LLM calls within a single transaction into one row. Call Count and Error Count are displayed together.
 - **Step:** Displays each individual LLM call (Step) as a separate row. You can check details such as model, temperature, and TTFT for each call.
- **Filter:** Enter search queries to view only logs matching specific conditions.

Left Filter Panel

Provides tag-based filters for log fields. Value lists for each field are displayed and can be clicked to apply filters.

Trend Chart

Displays the log count for the selected time range as a bar chart by time period. You can collapse or expand the chart to adjust space, and drag the divider between the chart and log list to adjust proportions.

Log List

Displays LLM call logs in a table format. The columns displayed vary depending on the merge criteria (Transaction/Step).

Default Columns

Column	Description
Status	Indicates error status by color. Rows with errors are displayed in red.
Time	Log timestamp
Operation	Operation Type of the LLM call
Prompt	Summary of prompt input and response. Click to open the transaction profile pop-out.
Model	LLM model name used
Provider	LLM provider (e.g., api.openai.com)
Tokens	Total token count (Input + Output)
Latency	Time from request start to response completion (ms)
Cost	Cost of the call (\$)

Transaction Mode Columns

Column	Description
Call Count	Number of LLM calls within the transaction
Error Count	Number of calls with errors within the transaction

Step Mode Columns

Column	Description
Step	LLM call sequence index within the transaction
Step ID	Unique identifier of the individual LLM call
temperature	Temperature parameter value set for the LLM call
TTFT	Time To First Token (ms). Displayed only in streaming mode.

Additional Columns

Additional columns can be displayed through column settings.

Column	Description
Input Tokens	Number of input tokens
Output Tokens	Number of output tokens
Cached Tokens	Number of tokens retrieved from cache
Reasoning Tokens	Number of reasoning tokens
Input Cost	Input cost (\$)
Output Cost	Output cost (\$)
Stream	Streaming mode status (True/False)
Features	Enabled feature flags
Agent	Agent name

Column	Description
Error	Error message (displayed only when an error occurs)
Error Type	Error type (<code>api_error</code> , <code>program_error</code> , etc.)
TXID	Transaction ID

Log Detail View

Click the **Prompt** column in the log list to open a transaction profile pop-out. You can view the full transaction details containing the LLM call.

- **Transaction mode:** Navigates to the full transaction view.
- **Step mode:** Opens the transaction view with the corresponding Step highlighted.

In the transaction profile, click an LLM HTTPC step to view the original prompt input/output, token/cost details, and GPU correlation analysis. For more details, see the [LLM API Analysis](#) document.

Merge Criteria Details

Transaction Merge

Merges all LLM calls within a single transaction (@txid) into one row.

- **Prompt display:** The input of the first Step and the output of the last Step are displayed.
- **Tokens/Cost:** The sum of all calls within the transaction is displayed.
- **Model:** The representative model of the transaction is displayed.
- **Call Count / Error Count:** The total call count and error call count are displayed.

Step Merge

Displays each individual LLM call (@step_id) as a separate row.

- **Prompt display:** The input (Input Message) and output (Output Message) of the corresponding Step are displayed.
- **Tokens/Cost:** Individual values for the call are displayed.
- **Additional info:** Details such as temperature, TTFT, and Stream status can be checked at the call level.

Entry from Other Menu

Prompt Log is frequently used when drilling down from other analysis menus.

Entry Path	Description
Token Trend > Click time series chart	The time range for that point is automatically set and Prompt Log opens.
Token Trend > Click box plot model	The filter for that model is automatically applied and Prompt Log opens.

Conversely, clicking the **Prompt** column in Prompt Log navigates to the transaction profile, where you can view all steps (DB, HTTP, LLM, etc.) within the transaction in chronological order.

Reference

Log Type

Prompt Log internally queries logs in the `#LlmCallLog` category.

Log Type	Description
<code>llm_step_status</code>	Final result of an individual LLM call (model, tokens, cost, performance, etc.)
<code>llm_tx_status</code>	Transaction-level aggregated result (call count, error count, summed tokens/cost)
<code>system_message</code>	System message (system role part of the prompt)
<code>input_message</code>	Input message (user prompt)

Log Type	Description
output_message	Output message (model response)
tool	Tool call (Function calling)
tool_result	Tool call result

Prompt Analysis

This menu identifies the prompts in use in your application and analyzes call volume, tokens, latency (TTFT), cost, and errors per prompt. Use it to find out which prompts are used the most and which are slow, expensive, or unstable, so you can select optimization targets.

Calls are identified by two units: **operation** and **prompt version**. An operation is a group of calls at the feature level, and a prompt version is the version of the system prompt used in that operation. One operation can have several prompt versions, and most charts are aggregated by prompt version.

From top to bottom, the screen is organized as **Summary cards** → **Calls and trends** → **Distribution and errors** → **Feature usage** → **Table**, so you can naturally follow it from grasping the status to checking individual operations. Set the time range and filters in the top option bar, and then query the widget data.

Prompt analysis

Summary cards

Cards that show the key metrics of the selected time range at a glance.

Unique prompts

Shows the number of distinct prompts called at least once within the selected time range. The bottom of the card also shows the **total number of calls** for those prompts.

- This is the starting point for grasping the scale of your prompt operations at a glance.
- If the number of unique prompts spikes above normal, check whether unintended prompt variants have been deployed.

Total prompt calls

Shows the total number of prompt calls that occurred within the selected time range. The bottom of the card also shows the **average per prompt**. Viewing it together with the average per prompt lets you gauge whether calls are concentrated on a few prompts or spread evenly.

- $\text{Average per prompt} = \text{Total prompt calls} / \text{Unique prompts}$

Total prompt-related cost

Shows the total cost (\$) incurred by LLM API calls in the selected time range. The bottom of the card also shows the version name and amount of the **most expensive operation**, along with its share (%) of the total.

- A single prompt with a high share is a priority target for cost optimization.
- If a specific prompt accounts for a large portion of the total cost, review the token length or model selection of that prompt first.

TOP 10 prompts by call volume

Property	Value
Chart Type	Vertical bar chart
Aggregation Unit	Prompt version
Action	Select the displayed metric (dropdown at the top right)

Shows the top 10 prompt versions by call volume, sorted by bar length. You can switch the displayed metric (call count, total tokens, total cost) using the dropdown at the top right.

- If calls are concentrated on a specific version, that version has a large impact on overall performance and cost, so manage it first.
- If a newly deployed version quickly enters the top ranks, it means the traffic switch (rollout) is proceeding normally.

Prompt usage trend

Property	Value
Chart Type	Stacked area chart
Aggregation Unit	Prompt version
Action	Toggle legend items

Shows the call volume trend of each prompt version over time as stacked areas. The legend colors correspond to each prompt version, and the thickness of an area indicates the call volume of that version.

- You can grasp the flow of the total call volume by time of day from the change in the thickness of the whole area.
- If the share of a specific version changes suddenly in a specific time slot, check the impact of a change in traffic pattern or a new version deployment.
- If the area thickens quickly right after a new version is deployed, it means the traffic switch is in progress.

TTFT distribution by prompt

Property	Value
Chart Type	Box plot (x-axis: prompt version)
Aggregation Unit	Prompt version
Action	View details

Compares the distribution of **TTFT (Time To First Token)** by prompt version as box plots. The box area represents p25 to p75 (the middle 50% range), the center line is the median, and the whiskers represent the minimum to the maximum.

- The version with a low median and a narrow box provides the most stably fast initial response.
- A version with long whiskers experiences extreme initial latency intermittently, so check the input length or the model load.

Finish reason distribution

Property	Value
Chart Type	Donut chart
Aggregation Unit	Finish reason (length / stop / none)
Action	Toggle legend items

Aggregates the finish reason values that the LLM API returns in its responses as they are, and shows them as a distribution. The values and names shown **may differ by provider and model**. WhaTap does not define them separately and uses the values returned by the API as they are.

- If the value that indicates normal termination (for example, `stop`) accounts for most of the distribution, it means responses are finishing as intended.
- If the value that indicates truncation due to a length limit (for example, `length`) has a high share, check whether the maximum token (`max_tokens`) setting is too small and responses are being cut off in the middle.
- Calls whose finish reason is empty or not recorded may be grouped and shown under a separate value (for example, `none`).

ⓘ The finish reason values shown follow the response specification of the LLM provider and model you use. For the exact meaning of each value, see the API documentation of that provider.

Error rate by prompt

Property	Value
Chart Type	Vertical bar chart (x-axis: prompt version)
Aggregation Unit	Prompt version
Action	View details

Shows the ratio (%) of errors to calls by prompt version.

- A version with a high error rate may be caused by an incorrect tool definition, a schema mismatch, or a token overflow, so check the metrics of that version together in the operation list.
- Cost is incurred even for failed requests, so improving a version with a high error rate also has a direct effect on cost savings.

Error rate = Error calls / Total calls x 100

Feature usage ratio by operation

Property	Value
Chart Type	Card grid (one card per operation)
Aggregation Unit	Operation
Action	-

One card is created per operation, and the operation name and total call count are shown at the top of the card. Inside the card, the ratio at which each feature (for example, `reasoning`) was used in the calls of that operation is shown as a bar and a percentage.

- A single call can use several features at the same time, so the ratios may not add up to 100%, and features

that were not used are not shown.

- Feature usage patterns let you quickly understand how an operation behaves and discover feature usage that differs from your intent.

Operation list

Property	Value
Chart Type	Table
Aggregation Unit	Operation + prompt version
Action	Sort

Lists the operations identified within the period by prompt version and organizes the key metrics in a table. Even for the same operation, a different prompt version is shown as a separate row.

Column	Description
Operation	Name of the group of calls at the feature level
Prompt version	Version of the system prompt used in that operation
Prompt preview	The beginning of the original system prompt (— if none)
Calls	Total number of calls within the period
Avg input	Average number of input tokens per call
Avg output	Average number of output tokens per call
Avg TTFT	Average time to the first token (ms)
Avg cost	Average cost per request (¢)

Column	Description
Error rate (%)	Ratio of errors to calls

- The default sort is descending by call count, and you can change the sort criteria by clicking a column header.
- A row with a high Avg cost is a prompt with an expensive unit price per request, so consider reducing the prompt length or reviewing the model.
- Prompt preview lets you quickly compare the prompt differences between versions of the same operation.

Reference

How prompts are identified

Unit	Description
Operation	A group of calls at the feature level (for example, cs_chat_support)
Prompt version	The version of the system prompt used in that operation (for example, v2.1-ko)

- One operation can have several prompt versions, and the TOP 10, usage trend, TTFT distribution, and error rate widgets are aggregated by prompt version.
- To quantitatively compare the performance differences between prompt versions, use the **Prompt comparison** menu. If you need response quality and safety evaluation, use the **Response quality analysis** menu together.

Prompt Comparison

This menu places two prompts side by side and compares their call volume, cost, performance, and response quality. Use it to quantitatively verify whether a prompt change actually improved performance and cost, or to analyze the differences in characteristics between different operations and versions.

You specify each comparison target as a combination of **operation + prompt version + date**, placing the baseline on **A** and the comparison target on **B**. Normally you put the existing version on A and the changed version on B to check the difference.

From top to bottom, the screen is organized as **Summary (calls and cost) → Prompt diff → Metric comparison → Trend comparison → Distribution comparison → Prompt logs**, so you can follow it from "what changed" (diff) to "what result that change produced" (metrics, trends, and logs).

Prompt comparison

Setting the comparison conditions

Specify the two prompts to compare in the top option bar.

- **Base date / Operation A / Prompt version A:** Select the baseline prompt (A) and the query date.
- **Comparison date / Operation B / Prompt version B:** Select the comparison target prompt (B) and the query date.

✓ You can specify different dates for A and B, so you can also compare the same prompt before and after a deployment across different periods. In every widget, A is shown consistently in blue and B in purple.

Total prompt calls

Property	Value
Chart Type	Bar chart (A vs B)
Comparison	A (baseline) · B (comparison)
Action	Show values on mouseover

Compares the total number of calls of the two selected prompts as bars side by side. Each bar is labeled with `date · prompt version`.

- If the difference in call count is large, also consider the difference in sample size when interpreting average metrics or trends.
- If the call count of the new version (B) is growing quickly, it means a traffic switch (rollout) is in progress.

Total prompt-related cost

Property	Value
Chart Type	Bar chart (A vs B)
Comparison	A (baseline) · B (comparison)
Action	Show values on mouseover

Compares the total cost of the two selected prompts as bars side by side.

- If the call counts are similar but the total costs differ greatly, the cause may be a difference in prompt length or model unit price.
- You can compare the cost per call independently of sample size on the `Cost per call` card in **Metric comparison** below.

System prompt diff

Property	Value
Chart Type	Text diff (left: A / right: B)
Action	Switch between Side-by-side and Inline views

Compares the original system prompts of A and B line by line. Added and deleted lines are distinguished by color, and you can switch between the **Side-by-side** (split) and **Inline** (single flow) views using the icon at the top right.

- Interpret this widget together with the metric and trend widgets below: this one shows "what changed," and those show "what result that change produced."
- If the prompt of one version is empty, it means that version was newly introduced or completely replaced.

Metric comparison cards

Property	Value
Chart Type	Metric cards (A · B values shown in parallel)
Comparison	A (baseline) · B (comparison)
Action	-

Shows the key performance, cost, and quality metrics with the A and B values side by side on each card.

Metric	Description
Average latency	Average time from request start to response completion
Average TTFT	Average time to the first token (perceived responsiveness)

Metric	Description
Average TPOT	Average generation interval per output token (streaming speed)
Cost per call (\$)	Average cost per request
Error rate (%)	Ratio of errors to calls
Cache hit rate	Ratio of input tokens served from the cache

- By directly contrasting the A and B values, you can see at a glance whether the change improved or degraded each metric.
- If the cost per call went down while the cache hit rate went up, an increase in the common prompt prefix may have improved caching efficiency.

TTFT trend (A vs B)

Property	Value
Chart Type	Line chart (2 series: A, B)
Comparison	A (baseline) · B (comparison)
Action	Show values on mouseover

Overlays the average TTFT by time of day for A and B.

- Check whether the gap widens only in specific time slots (for example, traffic peaks) or is spread evenly across the whole range.
- If intermittent spikes appear on one side, you can suspect queue waiting or provider latency.

Cost per call trend (A vs B)

Property	Value
Chart Type	Line chart (2 series: A, B)
Comparison	A (baseline) · B (comparison)
Action	Show values on mouseover

Compares the trend of cost per call by time of day for A and B.

- You can see over time how cache hit patterns or changes in output length affected the cost per call.
- You can discover patterns where the cost spikes only in specific ranges even though the averages are the same.

Call volume trend (A vs B)

Property	Value
Chart Type	Line chart (2 series: A, B)
Comparison	A (baseline) · B (comparison)
Action	Show values on mouseover

Compares the call volume trend of the two prompts by time of day.

- You can see the progress of the traffic switch to the new version (B).
- Ranges with a large volume difference provide the basis for the difference in sample size when interpreting other metrics.

Error trend (A vs B)

Property	Value
Chart Type	Line chart (2 series: A, B)
Comparison	A (baseline) · B (comparison)
Action	Show values on mouseover

Compares the error rate trend by time of day for A and B.

- Check whether errors clustered at a specific point right after the change, or stayed stably low.
- If there are ranges where the error rate of B is higher than A, trace back the change that caused it in the system prompt diff.

Finish reason distribution

Property	Value
Chart Type	Donut chart (A · B each)
Aggregation Unit	Finish reason value
Action	Toggle legend items

Shows the distribution of the response finish reason for A and B side by side as donuts. The values and names shown may differ by provider and model. WhaTap does not define them separately and uses the values returned by the API as they are.

- If the proportion of the value that indicates truncation due to a length limit (for example, `length`) decreased in B, it means output control improved and fewer responses were cut off.
- You can gauge the difference in response completeness from the change in the proportion of the value that

indicates normal termination (for example, `stop`).

ⓘ The finish reason values shown follow the response specification of the LLM provider and model you use. For the exact meaning of each value, see the API documentation of that provider.

Feature usage comparison

Property	Value
Chart Type	Bar chart (A vs B)
Comparison	A (baseline) · B (comparison)
Action	-

Compares the usage ratio of each feature for A and B.

- You can verify whether the prompt change (for example, adding examples or strengthening instructions) drove the intended change in feature usage patterns.
- A single call can use several features at the same time, so the ratios may not add up to 100%.

Model distribution

Property	Value
Chart Type	100% stacked horizontal bar (one row each for A · B)
Comparison	A (baseline) · B (comparison)
Action	Toggle legend items

Compares the composition ratio of the models called by each prompt as A and B rows. The legend colors

correspond to each model.

- If the model composition of A and B differs, you must distinguish whether the latency and cost differences come from the prompt change or from the model difference.
- If the proportion of a specific model changed significantly, check whether the routing rules or model designation were changed.

Prompt logs (A · B)

Property	Value
Chart Type	Table (A · B each)
Action	Expand rows, go to prompt logs

Shows the recent call logs of A and B in tables so you can check, case by case, how the metric differences appear in the actual responses.

Column	Description
Time	Time the call occurred
Prompt	Summary of the input and the output
Model	Name of the LLM model used
Token	Total number of tokens
Latency	Time from request start to response completion
Cost	Cost of the call

- For calls to which response quality evaluation was applied, the score of each evaluation item (for example, Hallucination, Factuality, Relevance, Toxicity, Prompt Injection, PII Leak, URL Scan, Judge) is also shown in the **Eval** area. For details about the evaluation items, see the **Response quality analysis** document.

- Use the **Go to prompt logs** button at the top right of each table to move to the original logs of that prompt.

Reference

How to specify comparison targets

Term	Description
Operation	A group of calls at the feature level (for example, <code>cs_chat_support</code>)
Prompt version	The version of the system prompt used in that operation (for example, <code>v3.0-structured</code>)
Date	The base date for querying each target (A and B can be specified differently)

- To first find comparison target candidates, use **Prompt analysis**. If you need evaluation from the perspective of response quality and safety, use the **Response quality analysis** menu.
- In every widget, A is shown in blue and B in purple.

Response Quality Analysis

This menu automatically evaluates the quality and safety of the responses generated by an LLM, and monitors risks such as hallucination, toxicity, PII leak, prompt injection, factuality, and answer relevance by category. Use it to diagnose whether "the response content itself is trustworthy and safe" — something that operational metrics such as call volume, cost, and latency alone do not reveal.

Evaluation is performed in a **hybrid manner** that combines several evaluators such as regular expressions, NLI, and LLM Judge. To control cost, only a **sample** of all calls is evaluated. Each evaluation item produces a score, and violations are aggregated based on the **violation criteria (threshold)** of each widget.

From top to bottom, the screen is organized as **Summary** → **Violation trend and Hitmap** → **Score distribution and model comparison** → **Time-of-day Hitmap** → **Violation cases**, so you can follow it from grasping the overall status to checking individual violation cases. Set the time range and filters in the top option bar, and then query the widget data.

Prompt comparison

Summary cards

Cards that show the key quality metrics of the selected time range at a glance.

Evaluation rate

Shows the ratio of calls for which quality evaluation was performed among all LLM calls. The bottom of the card also shows the number of evaluation LLM calls and the total number of calls. (For example,

Evaluation LLM calls 170 / 134.8K)

- Evaluation incurs cost, so it runs within the sampling ratio. If the evaluation rate is low, the sample is small, so interpret the ratios in the other widgets with care.

Response quality

Shows the overall quality score of the evaluated responses.

- This is a representative metric that aggregates several evaluation items. Read a downward trend as a signal of quality degradation.

Suspected hallucination rate

Shows the ratio of calls whose hallucination score is at or above the criteria. The bottom of the card also shows the decision criteria and the corresponding number of calls. (For example, `score ≥ 0.5 11.4K`)

- This indicates how often content that differs from fact is generated in RAG and search-based responses. If the ratio rises, check the context composition or the model.

Factuality

Shows the average level of how well the responses conform to fact (factuality).

- The higher the value, the more faithful the responses are to fact. Viewing it together with the suspected hallucination rate lets you grasp the overall trend of factual distortion.

Toxicity

Shows the ratio of responses evaluated as toxic.

- This monitors how often offensive or harmful expressions that must not be exposed to users occur. If the value spikes, check the actual responses in Violation cases.

Violation trend by category

Property	Value
Chart Type	Stacked area chart
Aggregation Unit	Evaluation category
Action	Select violation criteria (threshold), toggle legend items

Shows the trend of violations for each risk category over time as stacked areas. The legend colors correspond to each category (hallucination, toxicity, PII leak, prompt injection, factuality, answer relevance).

- Use the **Violation criteria** dropdown at the top right to adjust the score threshold that counts as a violation.
- If a specific category spikes at a specific point in time, trace the cause by correlating it with deployments or traffic changes.

Operation × category violation rate Hitmap

Property	Value
Chart Type	Hitmap table (rows: operation, columns: category)
Aggregation Unit	Operation × category
Action	Select violation criteria (threshold)

Rows are operations and columns are risk categories, and each cell shows the violation rate (%) of that combination. The higher the violation rate, the redder the cell; the lower it is, the greener it is.

- This widget lets you pinpoint at a glance "which operation is vulnerable to which risk."
- For example, if the answer relevance column of a specific operation appears red, it means that prompt frequently generates responses that miss the intent of the question.
- When you adjust the threshold with the **Violation criteria** dropdown at the top right, the cell colors and values are updated together.

Hallucination score distribution

Property	Value
Chart Type	Histogram (x-axis: score range, y-axis: evaluated calls)
Aggregation Unit	Score range (0–10% ... 100%)

Property	Value
Action	Show values on mouseover

Shows the distribution of hallucination evaluation scores as the number of calls per score range.

- If the distribution is concentrated in the low score range (0–10%), it means hallucination is not suspected in most responses.
- If a thick tail appears in the high score range (especially 100%), a certain volume of responses is suspected of factual distortion, so check the actual cases in Violation cases.

Quality comparison by model

Property	Value
Chart Type	Bar chart (x-axis: model)
Aggregation Unit	Model
Action	Show values on mouseover

Compares the response quality score of each model as bars.

- A model with a low quality score may cause more violations under the same workload.
- Review it together with the cost and latency metrics to decide whether to switch to a model with better efficiency relative to quality.

Factuality score distribution

Property	Value
Chart Type	Histogram (x-axis: score range, y-axis: evaluated calls)

Property	Value
Aggregation Unit	Score range (0–10% ... 100%)
Action	Show values on mouseover

Shows the distribution of factuality evaluation scores as the number of calls per score range.

- If the distribution is concentrated in the high score range (especially 100%), it means most responses are faithful to fact.
- If the proportion of the low score range (0–10%) is thick, a certain volume of responses differs from fact, so interpret it together with the hallucination score distribution.

Violation Hitmap by time of day

Property	Value
Chart Type	Hitmap (day of week × time of day)
Aggregation Unit	Day of week (Sun–Sat) × hour (0–23)
Action	Select violation criteria (threshold)

Shows the density of violations on a grid of days of the week (vertical) and hours (horizontal). The darker the cell, the more violations are concentrated in that time slot.

- If violations cluster in specific days or hours (for example, weekday afternoons), check the correlation with traffic load or specific user patterns.
- Use the **Violation criteria** dropdown at the top right to adjust the threshold.

Violation cases

Property	Value
Chart Type	Table
Action	Select violation criteria (threshold), expand rows

Lists the individual calls determined to be violations by the evaluation, one row per call. This widget is for checking actual cases directly rather than statistics.

Column	Description
Time	Time the call occurred
Operation	Operation in which the violation occurred
Prompt	Summary of the input and the output
Model	Name of the LLM model used
Provider	LLM provider
Token	Total number of tokens
Latency	Time from request start to response completion
Cost	Cost of the call

- Expanding a row shows the original input and response along with the score of each evaluation item in the **Eval** area. (Hallucination, Factuality, Relevance, Toxicity, Prompt Injection, PII Leak, URL Scan, Judge)
- Use the **Violation criteria** dropdown at the top right to adjust the threshold for marking violations.

Reference

Evaluation categories

The category names on the screen correspond to the evaluation items in the **Eval** area as follows.

Category	Eval Item	Description
Hallucination	Hallucination	Degree to which content not in the context was generated
Factuality	Factuality	Degree to which the response conforms to fact
Answer relevance	Relevance	Degree to which the response matches the intent of the question
Toxicity	Toxicity	Degree to which offensive or harmful expressions are included
Prompt injection	Prompt Injection	Attempts to hijack or bypass the system prompt
PII leak	PII Leak	Degree of personally identifiable information exposure
-	URL Scan	Whether URLs included in the response are risky
-	Judge	Overall decision score of the LLM Judge

Violation criteria (threshold)

- For the violation trend by category, violation rate Hitmap, violation Hitmap by time of day, and violation cases widgets, you can select the **Violation criteria** (score threshold) at the top right.
- When you adjust the threshold, the criteria by which that widget counts violations changes, and the data is updated together.
- Evaluation is performed on a sample of all calls, so the violation rate is calculated based on the evaluated calls.

LLM Metrics Glossary

This document is a glossary of the key metrics and concepts used in AI Agent Observability. It contains the definitions and units of indicators commonly used across multiple menus such as the dashboard, token trend, and cost analysis.

LLM Performance Metrics

Metric	Description
Latency	Total elapsed time from LLM API request to full response completion (ms). Overall user-perceived wait time
TTFT	Time To First Token. Time until the first response token arrives after the request (ms). Perceived response start point in streaming environments
TPOT	Time Per Output Token. Average time to generate a single output token (ms). Higher values create a choppy response feeling
Output TPS	Tokens Per Second. Output tokens generated per second. Overall model throughput indicator

Tokens and Cost

Metric	Description
Input Tokens	Total input token count in the prompt. Includes Cached Tokens
Output Tokens	Response token count generated by the model. Directly affects cost

Metric	Description
Cached Tokens	Input token count retrieved from cache. Subset of Input Tokens. Lower rate than regular input
Cache Hit Rate	Ratio of cached tokens to total input tokens (%). Higher values mean greater cost savings
I/O Ratio	Ratio of output tokens to total tokens (%). Used to identify workload characteristics (input-heavy vs. generation-heavy)

Percentiles

Percentile	Description
p50	50% of all data at or below this value. Typical user-perceived performance
p75	75% of all data at or below this value. Gap from p50 shows additional latency for upper-tier requests
p95	95% of all data at or below this value. Commonly used as SLA threshold
p99	99% of all data at or below this value. Slowest 1% performance, used to assess system stability

Quality Evaluation Metrics

Metric	Description
Hallucination	Hallucination score. Degree of false information or contradiction with context. Lower is better (0.0 ~ 1.0)
Answer Relevance	Answer relevance score. Whether the question was answered appropriately. Higher is better (0.0 ~ 1.0)

Metric	Description
Toxicity	Toxicity score. Level of hate speech, profanity, violence, or sexual content. Lower is better (0.0 ~ 1.0)
Combined Judge	Overall risk score. $\max(\text{hallucination}, \text{toxicity}, 1 - \text{answer_relevance})$. Maximum risk value among the three metrics. Lower is better (0.0 ~ 1.0)
Prompt Injection	Degree of detected prompt injection attempts. Lower is better (0.0 ~ 1.0)
Factuality	Factual accuracy score of the response. Higher is better (0.0 ~ 1.0)
PII Leak	Degree of personally identifiable information (PII) exposure
URL Scan	Risk level of URLs included in the response
Eval Success	Success of the evaluation itself (1 = normal, 0 = judge call/parsing failure)

Tag-Based Filters

All data in AI Agent Observability can be filtered or grouped by the following tags.

Tag	Description	Example
Agent	LLM agent (application instance)	llm-app-01, llm-app-02
Model	LLM model used	gpt-4o, claude-sonnet-4-20250514
Provider	LLM API provider	api.openai.com, api.anthropic.com
Operation	User-defined call unit label (chain/prompt/agent name, etc.). Defaults to default if not set. Configured via prompt_meta	checkout_chain, rag_query, default

Tag	Description	Example
<code>Prompt</code> <code>Version</code>	Version of the prompt/chain	<code>v1</code> , <code>v2</code> , <code>v3</code>

OpenMetrics Template List

OpenMetrics provides default templates and predefined templates so that first-time users can quickly collect and visualize metrics without complex setup. Frequently used open-source metrics can be easily applied, lowering the initial adoption barrier and helping operators build a scalable monitoring environment.

Template List

Template List provides default open-source template cards. Each card includes a brief description, template navigation, and collection guide.

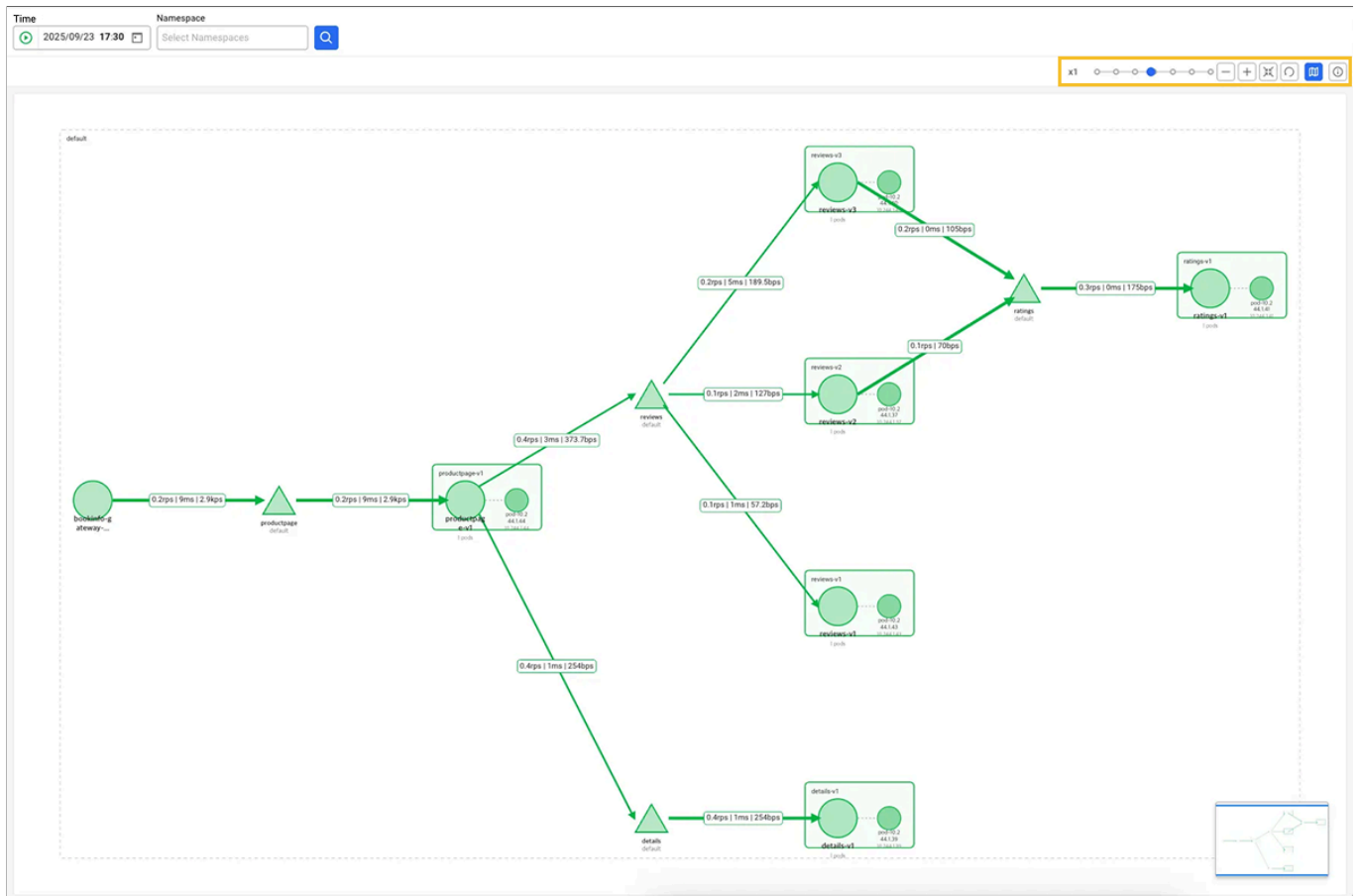
- → **Template:** Click to navigate directly to the template screen.
- ⓘ **Collection Guide:** Click to view the data collection configuration guide.

Nginx Template

Monitor Apache Nginx to track and analyze web server performance in real time.

Istio Template

Visually represents the service relationships and traffic flows in an Istio service mesh environment. Provides an integrated monitoring template by linking with WhaTap APM data. Click the button at the top right of the screen to zoom in or out on the template.



Legend

Click the ⓘ button at the top right of the screen to view the legend.

Node Shapes

- Triangle: Kubernetes Service
- Circle: Workload
- Rectangle: Application
- Small circle: Pod

Node Status

Ratio of errors (4xx, 5xx) in total HTTP requests/responses

- **Green:** Error rate below 5%
- **Orange:** Error rate 5% or higher
- **Red:** Error rate 10% or higher
- **Gray:** No requests

Pod Node Status

Kubernetes Pod Phase

- **Green:** Pod in Running state
- **Orange:** Pod in Pending state
- **Red:** Pod in Failed state

Edge Colors

Ratio of errors (4xx, 5xx) in HTTP requests between two nodes

- **Green:** Error rate below 10%
- **Orange:** Error rate 10% or higher
- **Red:** Error rate 20% or higher
- **Blue:** TCP communication
- **Gray:** Status unknown

OpenMetrics Explorer

Home > Select Project > **Analysis** > **OpenMetrics Explorer** [Beta](#)

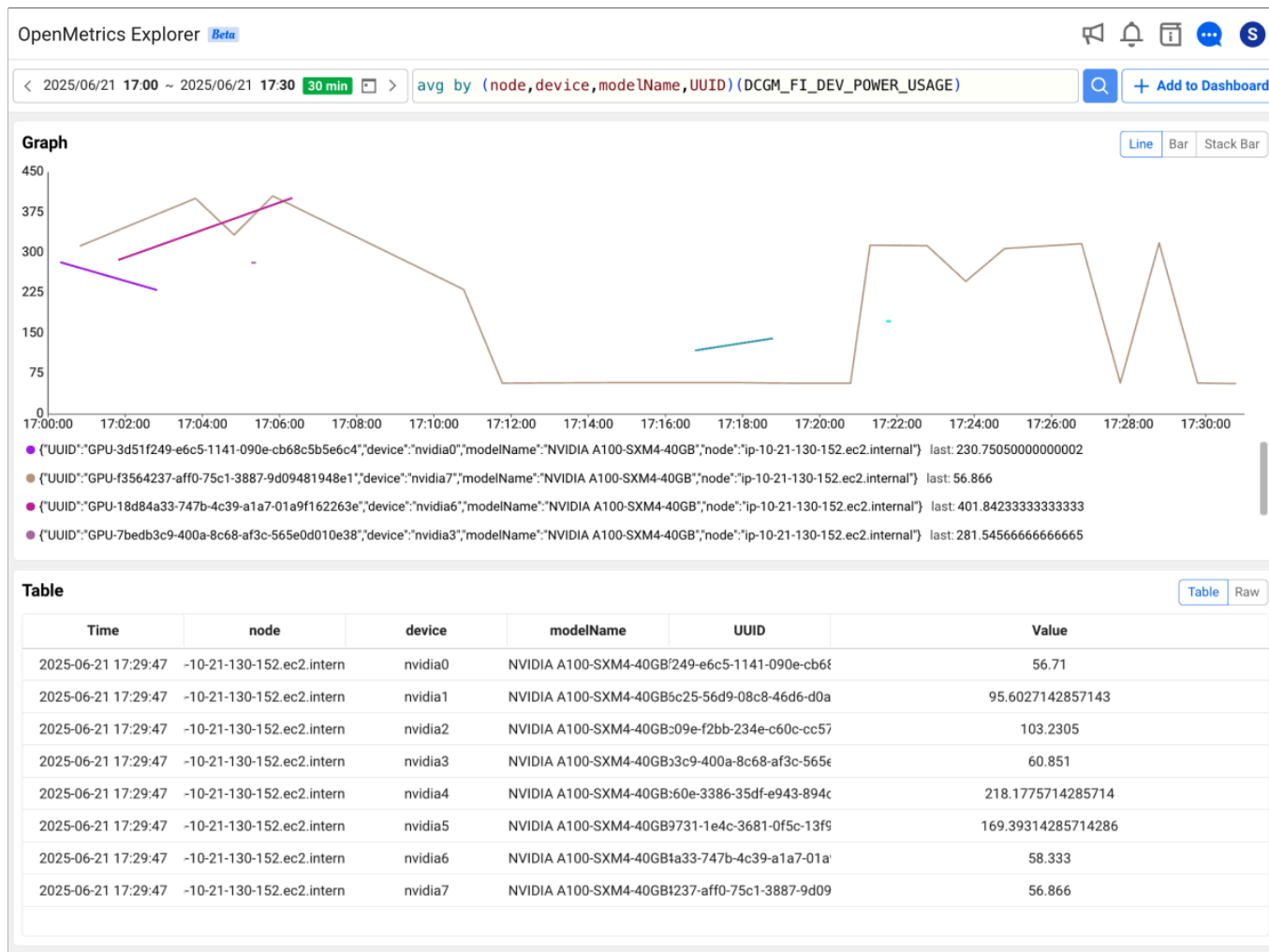
OpenMetrics Explorer lets you explore Prometheus-based custom metrics through PromQL, visualize them in various formats, and check metric status or gain insights.

Use this feature to freely query desired metrics, extract meaningful data, or link to dashboards for continuous monitoring.

ⓘ To use **OpenMetrics Explorer**, the open agent must be installed.

Exploring PromQL

In **OpenMetrics Explorer**, you can execute PromQL and check results with various visualization options.



Graph

- **Line:** Default time series line graph
- **Bar:** Visualization in bar chart format
- **StackBar:** Stacked bar chart showing cumulative values

Table / Raw

- **Table:** Displays values of each time series sorted chronologically in a table
- **Raw:** View the Prometheus API response (JSON format) as-is

PromQL Syntax Support and Autocomplete

When entering metrics such as `node_cpu_seconds_total` or `container_memory_usage_bytes`, the autocomplete feature suggests metric and operator lists.

Label filtering in the `{pod="example"}` format is also supported.

Adding to Dashboard

Save queried PromQL queries as dashboard widgets to continuously monitor desired metrics.

How to Add

1. Click the **+ Add to Dashboard** button at the top right of the **OpenMetrics Explorer** screen to open the **Add to Dashboard** modal.
2. In the **Add to Dashboard** modal, select an existing dashboard or create a new one.
3. Click the **Apply** button to add the widget to the specified dashboard.
 - You can enter a widget name and description in the added dashboard.
 - The added widget can be freely repositioned in the Flex Board Grid layout, and the PromQL can be modified.

Live Tail

! Without the **Log Read** role, you cannot enter the menu.

Home > Select Project > **Log** > **Live Tail**

In the **Live Tail** menu, you can directly check the log data stream on the monitoring screen without connecting to the server console. Even with large volumes of complex logs, you can filter and quickly identify them using the **Filter** and **Highlight** features. The log data retrieval cycle is 2 seconds.

The major terms are as follows:

- **Category:** Unit for log collection and inquiry
- **Content:** Log messages
- **Search Key:** It is generated through the log parser settings.
- **Tag:** Key to search the collected logs

! You can change the column width by dragging an edge of the log table column.

Agent option

If the agent option is set, the log levels are collected and the colors for each log level appear as follows:

2023-12-18	14:31:02.563	[level]	INFO	[pcode]	2277	[agenttime]	1702877462042	[oid]	778873916	[category]	AppLog	[loggerName]	io.home.test.logback02starter.base.web.LogbackControllerGreeting
													INFO log in our greeting method.
2023-12-18	14:31:02.563	[level]	WARN	[pcode]	2277	[agenttime]	1702877462043	[oid]	778873916	[category]	AppLog	[loggerName]	io.home.test.logback02starter.base.web.LogbackControllerError
												[threadName]	http-nio-19090-exec-2
													io.home.test.logback02starter.base.errors.exception.ApiException: [2204] Process failure. Please try again.
2023-12-18	14:31:02.586	[level]	error	[pcode]	2277	[agenttime]	1702877462043	[oid]	778873916	[category]	AppServer	[_event_status_]	error
												[_event_status_literal_name_]	level
													Servlet.service() for servlet [dispatcherServlet] in context with path [] threw exception [Request processing failed: io.home.test.logback02starter.base.error]
2023-12-18	14:31:02.586	[level]	error	[pcode]	2277	[agenttime]	1702877462057	[oid]	778873916	[category]	AppServer	[_event_status_]	error
												[_event_status_literal_name_]	level
													Servlet.service() for servlet [dispatcherServlet] in context with path [] threw exception [Request processing failed: io.home.test.logback02starter.base.error]
2023-12-18	14:31:02.586	[level]	error	[pcode]	2277	[agenttime]	1702877462081	[oid]	778873916	[category]	AppServer	[_event_status_]	error
												[_event_status_literal_name_]	level
													Servlet.service() for servlet [dispatcherServlet] in context with path [] threw exception [Request processing failed: io.home.test.logback02starter.base.error]
2023-12-18	14:31:02.586	[level]	error	[pcode]	2277	[agenttime]	1702877462087	[oid]	778873916	[category]	AppServer	[_event_status_]	error
												[_event_status_literal_name_]	level
													Servlet.service() for servlet [dispatcherServlet] in context with path [] threw exception [Request processing failed: io.home.test.logback02starter.base.error]

! Setting the agent options



- The agent options are as follows:

```
# whatap.conf
weaving=log4j-2.17
weaving=logback-1.2.8
```

- In Java agent 2.2.22 or later, it can be used when setting log4j-2.17 or logback-1.2.8 in the weaving setting. It is required to restart the agent.
- The log level is determined based on the `level` and `type` among the parsed keywords. When there are keys parsed as `level` and `type`, and the parsed values have **FATAL**, **CRITICAL**, **ERROR**, **WARN**, **WARNING**, and **INFO**, the log level color appears.

1 Filter area

Apply Filter

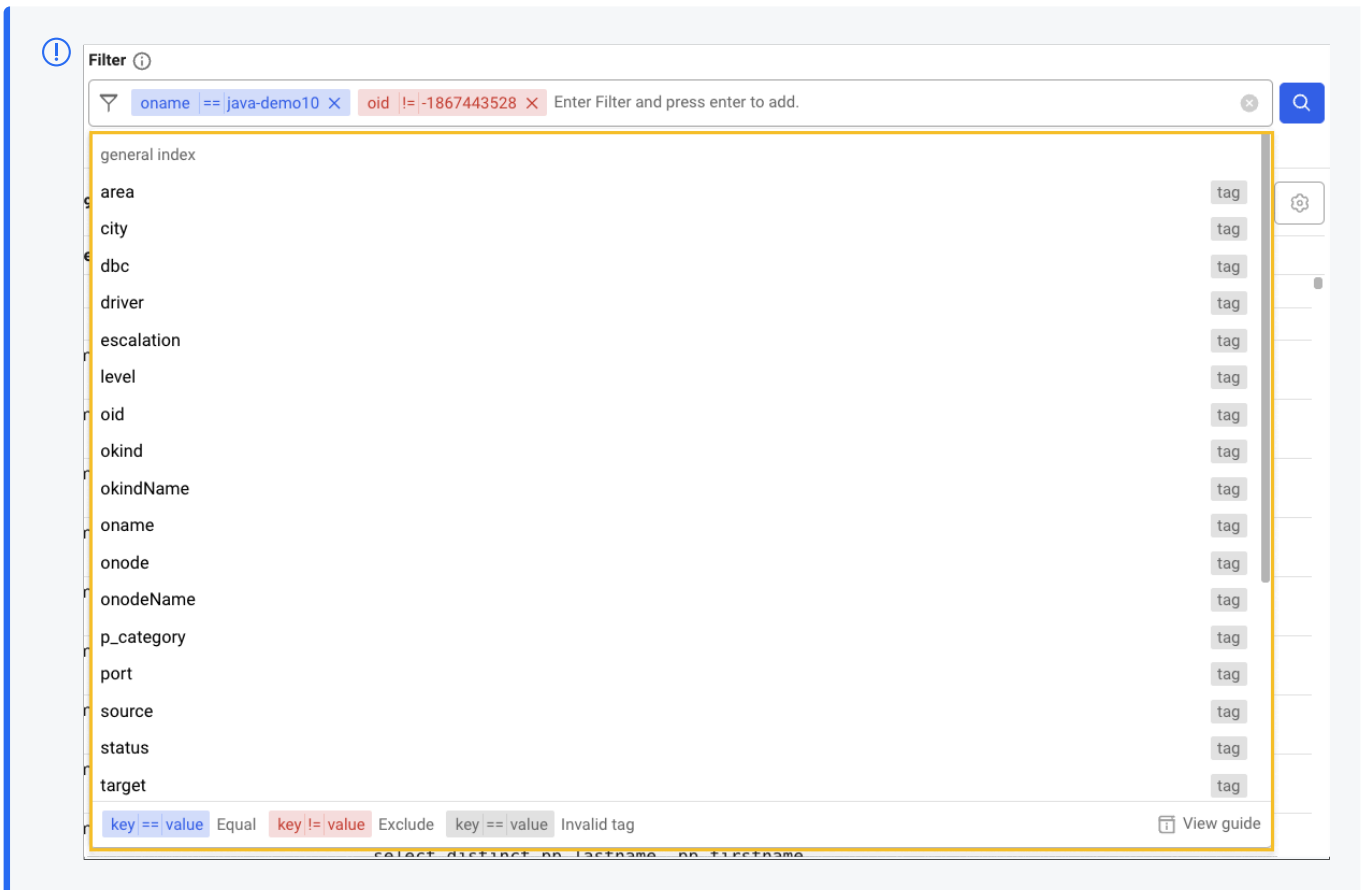
When a filter is applied, the logs that match the specified conditions are filtered. You can enter multiple filters. OR (`||`) if the filters have the same tags, otherwise AND (`&&`).

You can specify a filter by directly entering a value in the input field or clicking the **Filter** field. Filter tags are entered in the following order: `Search key`, `Operator`, and `Search value`. If you select **Search**, the data with the filter applied can be viewed log list area.



Guide UI

The **Guide UI** is provided under the input field as follows: You can close the **Guide UI** via the ESC key when your mouse cursor is over the input field or filter tag.



Entry of the search key, operator, and search value

- Upon entry of the **search key**, recommended values are provided by distinguishing for general index, reserved word index, and numeric index.
- When entering an **operator**, for a general index search key, **==** and **!=** options are provided at the bottom. In case of the numeric index, the **>**, **<**, **<=**, **>=**, **==**, and **!=** options are provided.
- Upon entry of the **search value** for matched search (**>**, **<**, **<=**, **>=**, **==**), it is highlighted in **blue**. In case of excluded search (**!=**), it is highlighted in **red**. Upon entry of the **search value**, you can search using the case-sensitive option.

- If the filter tags exceed 2 lines, you can fold it by selecting the **^ Fold** icon.
- After entering a filter tag, you can use the Shift and Tab keys simultaneously in the input field to go to the previous filter tag.



- After entering a filter tag, you can use the Tab key in the input field to go to the next filter tag.

Adding filter tags

To add a filter tag, type the desired text in the input field and press **Enter** or **Tab**.

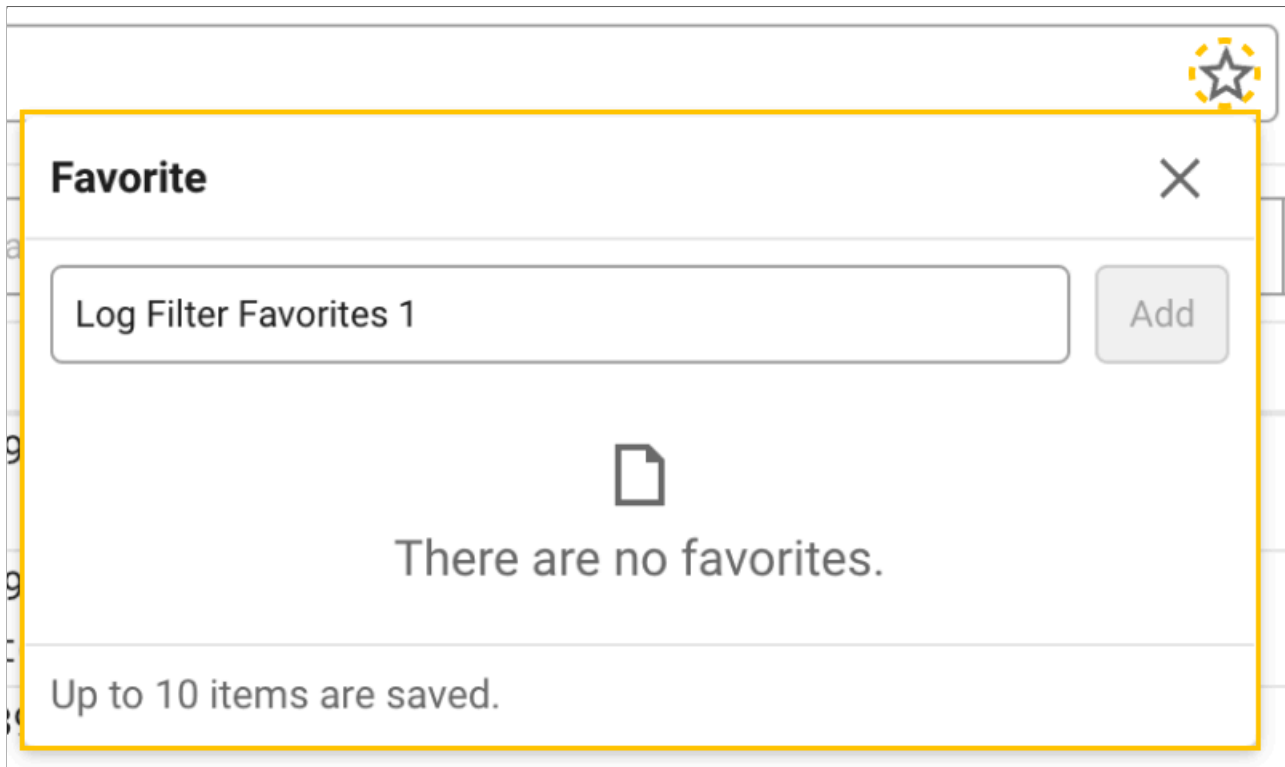
You can also select a recommended value from the **Guide UI** below the input field by clicking it or navigating with the arrow keys.

Deleting filter tags

To remove a single tag, press **Backspace** or click the **X** icon on the tag. To clear all tags at once, click the **X** icon located on the right side of the input field.

Filter favorites

The filter **favorites** feature is provided. You can select the ☆ **Favorites** icon on the right of the input window to **add**, **delete**, and select desired filter search conditions in the favorites. **Favorites** can store up to 50 cases.



Exceptional cases for application of filters

- In the tag with a numeric index (`search key` that ends with `.n`), only numbers can be input for the `search value` .
- Duplicate 'search key' and 'search value' cannot be entered.
- Searching is not possible if there is a tag whose `search key` or `search value` is missing. Invalid tags appear in gray.

ⓘ You cannot enter a `category` with the **Live Tail** `search key` .
The expression under the entered filter value is a preview of the filter expression for querying the log data.

Applying non-parsing keyword filters

You can search the logs containing keywords that have not been parsed, or for which no index has been created.

In this case, all logs in the specified range are fully scanned. Therefore, the search speed may be slower compared to the indexed key. For standardized log data, it is recommended to search using the index key through **Log parser setting**.

Filter ⓘ

🔍
oid != -1009464250 ✕
okind == -398596773 ✕
content == *select* ✕
Enter Filter and press enter to add.
✕

oid != -1009464250 && okind == -398596773 && content == *select*

1. Select **Category**.
2. In the **Filter** field, enter a desired keyword for a search after leaving spaces on the **content** basis.
e.g. `content *select*`
3. Click 🔍 **Search** to view the logs.

- ⓘ • In case of **Live Tail**, all logs can be viewed. Thus, the category is not required to be specified.
- For more information about the parser configuration, see [the following](#).

Modifying the filter

Enter a value in the filter and then click the specified value to modify it.



- You can modify the filter by entering text again in the input field.
- You can modify the recommended values by selecting them in the **Guide UI** below the input field.

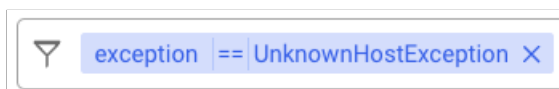
Search Key

In the following image, the blue boxed part is the search key for parsing. In **Log Configuration**, you can register and set the parsing logic in **Log parser setting** tab with the search key.

Search key

Filter input syntax

A tag consists of a search key and a search value. In the following example, the search key is `exception` and the search value is `UnknownHostException`. The example searches for the log data that includes an exception (`UnknownHostException`) that is generated when the server cannot connect the host because the IP address and domain address do not match among the collected log data.



Search key types

Search key types	Search key format	Meaning	Example for Search key and search value	Search example
String keyword	keyword	File Name	- Key: fileName - Value: /data/whatap/logs/yard.log	fileName:/data/whatap/logs/yard.log
Numeric keyword	keyword.n	Response Time	- Key: response_time.n - Value: 2945	response_time.n>=2945
Reserved keyword (predefined keyword)	@keyword	Transaction ID	- Key: @txid - Value: 85459614215434144	-
Log body's keyword	content	Log body	- Key: content - Value: user input	content: *ERROR*

ⓘ Content search key

- The content search key searches the bodies of unindexed logs. For example, if you enter `content *ERROR*`, the logs containing `ERROR` are searched among the log bodies.
- If you do not know which keywords need to be used for indexing, use the content search keys to identify the logs containing problematic keywords. Afterwards, you can improve the search performance by setting a parser with the keyword through the log parser settings in **Log Configuration** and creating indexes.

Common syntaxes

Syntax type	Description	Example
<code>==searchValue</code>	Searches for the logs that match the search value.	<code>exception==RuntimeExceptionexception</code>

Syntax type	Description	Example
<code>!=searchValue</code>	Searches for the logs that exclude the search value.	<code>exception!=RuntimeException</code>
<code>*searchValue</code>	Searches for the logs ending with the search value.	<code>word==*hello</code>
<code>searchValue*</code>	Searches for the logs starting with the search value.	<code>word==hello*</code>
<code>*searchValue*</code>	Searches for the logs that contain the search value in the middle.	<code>word==*hello*</code>
<code>*search*Value*</code>	Searches for the logs that contain the search value.	<code>word==*he*llo*</code>
<code>re:{regex}</code>	Searches for the logs that match a regular expression.	<code>caller==re:^(i\.w\.a\.w\.s\.v\.r\.</code>
<code>**</code>	Searches all logs that match the search key.	

Syntax in case the search key has a numeric value (keyword.n)

The following syntax is supported only when the search key is in `keyword.n` format.

- Search values must only be numbers.
- No prefix is attached to the value of the `.n` keyword. All keywords other than `.n` must be attached with a prefix.

For example, `+>searchValue` is invalid.

Syntax type	Description	Example
<code>>searchValue</code>	Searches for the logs containing values greater than the search value.	<code>response_time.n>3000</code>

Syntax type	Description	Example
<code>>=searchValue</code>	Searches for the logs containing values greater than or equal to the search value.	<code>response_time.n>=3000</code>
<code>==searchValue</code>	Searches for the logs containing the same search value.	<code>response_time.n==3000</code>
<code>!=searchValue</code>	Searches for the logs that contain values different from the search value.	<code>response_time.n!=3000</code>
<code><searchValue</code>	Searches for the logs containing values less than the search value.	<code>response_time.n<3000</code>
<code><=searchValue</code>	Searches for the logs containing values less than or equal to the search value.	<code>response_time.n<=3000</code>

Viewing personal information

If you have the **log personal information query** role, you can display de-identified personal information by enabling the **Display personal information**  toggle button.

- For more information about the **log personal information query** role, see [the following](#).
- For more information about **De-identification of personal information**, see [the following](#).

2 Content highlight area

The highlight function allows you to easily identify desired keywords in log content. You can filter with single or multiple keywords.

Live Tail

Category: ApplLog Filter: content == *select* Enter Filter and press enter to add.

select

	oname	Timestamp	Log
b	demo-8105	2023-11-24 14:08:35.620	@txid 5935183317842444630 pcode 5490 oname demo-8105 onodeName node-1 oid -857948929 okindName demo-okind-1 okind -628198688 select
	demo-8105	2023-11-24 14:08:35.624	@txid 4922255873619405363 pcode 5490 oname demo-8105 onodeName node-1 oid -857948929 okindName demo-okind-1 okind -628198688 select productmodelid, avg(listprice) as [average list price]
	demo-8105	2023-11-24 14:08:35.630	@txid 4922255873619405363 pcode 5490 oname demo-8105 onodeName node-1 oid -857948929 okindName demo-okind-1 okind -628198688 select productid, avg(unitprice) as [average price]
	demo-8105	2023-11-24 14:08:35.635	@txid 4922255873619405363 pcode 5490 oname demo-8105 onodeName node-1 oid -857948929 okindName demo-okind-1 okind -628198688 select

- Enter the keyword you want to highlight in the keyword input field.
 - Example: `select`
- The searched keyword will be highlighted in the Content of the log list.
 - If you click the  **Log full screen** icon, you can view **Log** and **Timestamp** in fullscreen.
 - You can navigate between highlighted keywords using shortcuts.
 - `Enter` : Move to the next row
 - `Shift+Enter` : Move to the previous row
 - `^` / `v` icons: Move between rows
 - `Cmd + F` (Mac) / `Ctrl + F` (Windows): Search keywords

Multi-keyword condition

When highlighting with multiple keywords, write as follows:

String to input	Description	Result
a b c	Separate each keyword with a space.	a, b, c
"Whatap is good."	To include spaces in keywords, wrap them in <code>'</code> or <code>"</code> .	Whatap is good.
"Whatap\\ is good."	If <code>\</code> is included in the keyword surrounded by <code>"</code> , enter it as <code>\\</code> .	Whatap\ is good.

Setting the highlight color

By selecting  **Highlight**, you can set the keyword and its color for highlighting.

- In addition, enter the keyword to set the color for.
- If you click **Color** on the left of the input field, available colors appear.
- Basically, different highlighting by the level (**WARN**, **ERROR**, **FATAL**) is applied.
- The settings are saved in **Project Rules**.

Table settings

Column settings

Right of the  area, select  **Column settings**. You can add columns or set order.

- **Adding columns**

You can add columns to the table by selecting tags. If you unselect the **log** column, you cannot see **Log view detail setting**. Be sure to select at least one column.

- **Setting the column order**

When you add a column, the column is added to **Display columns**. Drag a desired column to change the order of the columns.

Log view detail setting

On the right of , select  **Log view detail setting**. **content** and **Tag** are all checked and two items are also displayed. Be sure to select any of **content** and **Tag**.

Unchecked items are not displayed in the table. If you deselect the **Tag** as follows, the log's **Tag** does not appear in the table.

Log table tag invisible

When you add tags in the **Tag management** list, the tags in the log are listed in the order you added. You can change the order of the tags by dragging them. If you disable an added tag, the disabled tag does not appear.

- **Column settings** and **Log view detail setting** can be used in **Live Tail**, **Log Search**, and **Log Trend**.
- In the same project, the **Live Tail**, **Log Search**, and **Log Trend** menus share **Column settings** and **Log view detail setting**.

Log Explorer

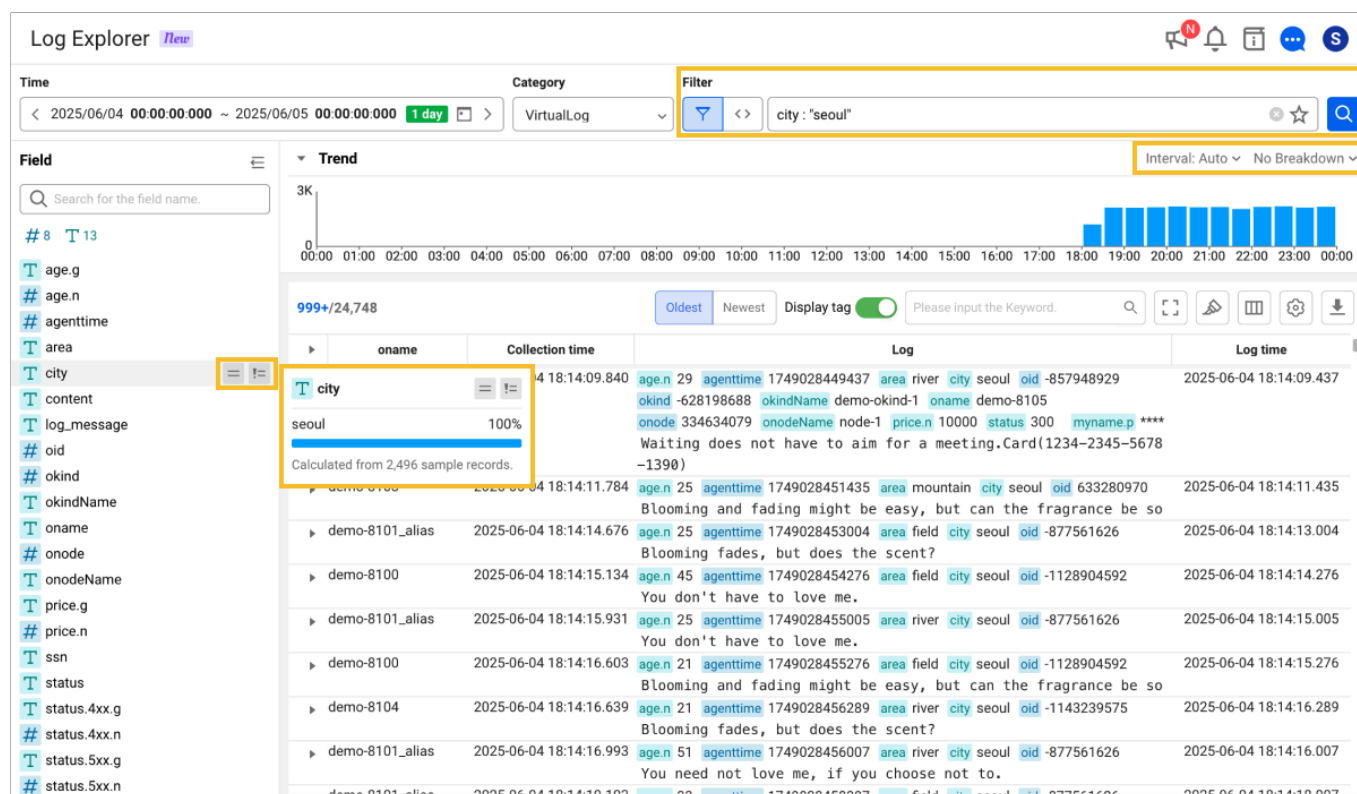
ⓘ Without the **Log Read** role, you cannot enter the menu.

Home screen > Select Project > **Log** > **Log Explorer** *New*

The **Log Explorer** feature allows you to quickly search, filter, and analyze log data. You can understand the field structure, perform query-based filtering to efficiently search for logs, and visualize the results. This feature is based on the Lucene index and offers the following key functions:

1. **Log trend analysis and search:** Enables both log trend visualization and search, helping you understand log patterns over time.
2. **Fast search and filtering:** Search for problematic logs quickly using simple keywords. Supports both WhaTap log search query and Lucene query syntax for defining various search conditions.
3. **Autocomplete functionality:** Helps users write and learn queries easily through auto-suggestions.
4. **Data visualization:** Uses stacked graphs to visualize the volume of logs over time. Drill-down capability allows deep analysis of specific logs.
5. **Favorites and sharing:** Save frequently used filters as favorites and share them easily via URL.

Basic Screen Overview



Time Range Selection

Set the time range for the log search using the **Time Selector** in the top option bar.

Category

Select the type of log you want to explore in **Select Category** on the top option bar (e.g., AppLog).

Filter

In the **Filter** on the top option bar, select the query language and enter the search conditions to apply the filter. The search query syntax supports both WhaTap Log Search Query and Lucene Query.

-  : WhaTap Log Search Query
-  : Lucene Query

Search Query Syntax

WhaTap log search query

WhaTap log search queries support autocomplete and allow quick log searches with simple keyword input. You can define various search conditions using operators like `and` , `or` , etc.

- Example: Entering `error` in the search filter will search within the log content.
 1. In the top option bar, click the filter icon and select a field from the query search box (fields are shown based on selected time and category).
 2. Choose an operator:
 - a. Default: `:` , `.*` , `and` , `or`
 - b. If the field is of type `Number` : Default + `<=` , `>=` , `<` , `>`
 - c. If the field is of type `String` : Default only
 3. Select a value (values shown based on selected time, category, and field).

Search Index Field Types

Only three field types are supported: `Number` , `String` , and `Boolean` .

- `Number` : Numeric values like long, double, float
- `String` : Full text such as email body
- `Boolean` : A value of either True or False

 For more details about WhaTap log search query syntax, refer to the [documentation](#).

Lucene

Use the standard syntax of Apache Lucene version 8.11.1. Autocomplete is not supported.

 For more information on the Lucene QueryParser Classic package, see the [official documentation](#).

Field List

Can be used for drill-down analysis.

Apply Field Info to Filter

Check field names in the **Field List** and click the `=` or `!=` buttons to apply them as filter conditions.

- Hovering over a field shows the `=` and `!=` buttons, which can be clicked to apply filters.
- Use the field search function to quickly find the desired field.

Trend Chart

Use **Trend Chart** to view logs in graph form with **interval** and **breakdown** applied.

- **Graph Interaction:** Zoom in using mouse drag or the  button, and click bars to view logs in that area
- **Interval:** Choose from Auto, 1s, 5s, 30s, 1m, 5m, 10m, 30m, 1h
- **Breakdown:** Visualize distribution of Top 3 values + Others for a specific field, and apply drill-down filters

Breakdown

Breakdown is a visualization feature that analyzes log count trends by a specific field.

- Displays a stacked graph separating the Top 3 values and Others for a given field
- Useful for drill-down analysis. You can click parts of the legend or stacked graph to highlight specific values or add filter conditions.

View Logs

You can view logs that match your search conditions. In the **Field List**, field data and log tags are displayed in matching highlight colors.

Content Highlight

The highlight function allows you to easily identify desired keywords in log content. You can filter with single or multiple keywords.

1. Enter the keyword you want to highlight in the keyword input field.
 - Example: `select`
2. The searched keyword will be highlighted in the Content of the log list.
 - If you click the  **Log full screen** icon, you can view **Log** and **Timestamp** in fullscreen.
 - You can navigate between highlighted keywords using shortcuts.
 - `Enter` : Move to the next row
 - `Shift+Enter` : Move to the previous row
 - `^` / `↓` icons: Move between rows
 - `Cmd + F` (Mac) / `Ctrl + F` (Windows): Search keywords

Multi-keyword condition

When highlighting with multiple keywords, write as follows:

String to input	Description	Result
<code>a b c</code>	Separate each keyword with a space.	a, b, c
<code>"Whatap is good."</code>	To include spaces in keywords, wrap them in <code>'</code> or <code>""</code> .	Whatap is good.
<code>"Whatap\\ is good."</code>	If <code>\</code> is included in the keyword surrounded by <code>""</code> , enter it as <code>\\</code> .	Whatap\ is good.

Setting the highlight color

By selecting  **Highlight**, you can set the keyword and its color for highlighting.

- In addition, enter the keyword to set the color for.
- If you click **Color** on the left of the input field, available colors appear.
- Basically, different highlighting by the level (**WARN**, **ERROR**, **FATAL**) is applied.
- The settings are saved in **Project Rules**.

Column settings

Click the  button to add columns or change their order.

- **Add Columns**

In **Column Settings**, select tags to add them as columns in the table.

 If no log columns are selected, **Log view detail setting** will not be visible. At least one column must be selected.

- **Reorder Columns**

Added columns will appear in **Display columns**. Drag them to change the display order.

Log view detail setting

Click the  button to configure log display options. By default, both **content** and **Tag** are selected and shown.

- Either **content** or **Tag** must be selected.
- Tags added in **Tag management** appear in order in the log display. You can reorder tags by dragging. Disabled tags are not shown.
- If an item is unchecked, it is not shown in the table. For example, if **Tag** is unchecked, log tags are not displayed.

log table contents remove tag sc

! Scope of Column and Log Display Settings

- **Column settings** and **Log view detail setting** can be used in **Live Tail**, **Log Search**, and **Log Trend**.
- In the same project, **Live Tail**, **Log Search**, and **Log Trend** share **Column settings** and **Log view detail setting**.

Download Log File

You can download the collected logs that match your search results in [CSV](#) or [TXT](#) file formats.

1. Click the  **Download** button at the right center of the screen.
2. Click **Download CSV** or **Download Log** to download the file in your preferred format.

Log Trend

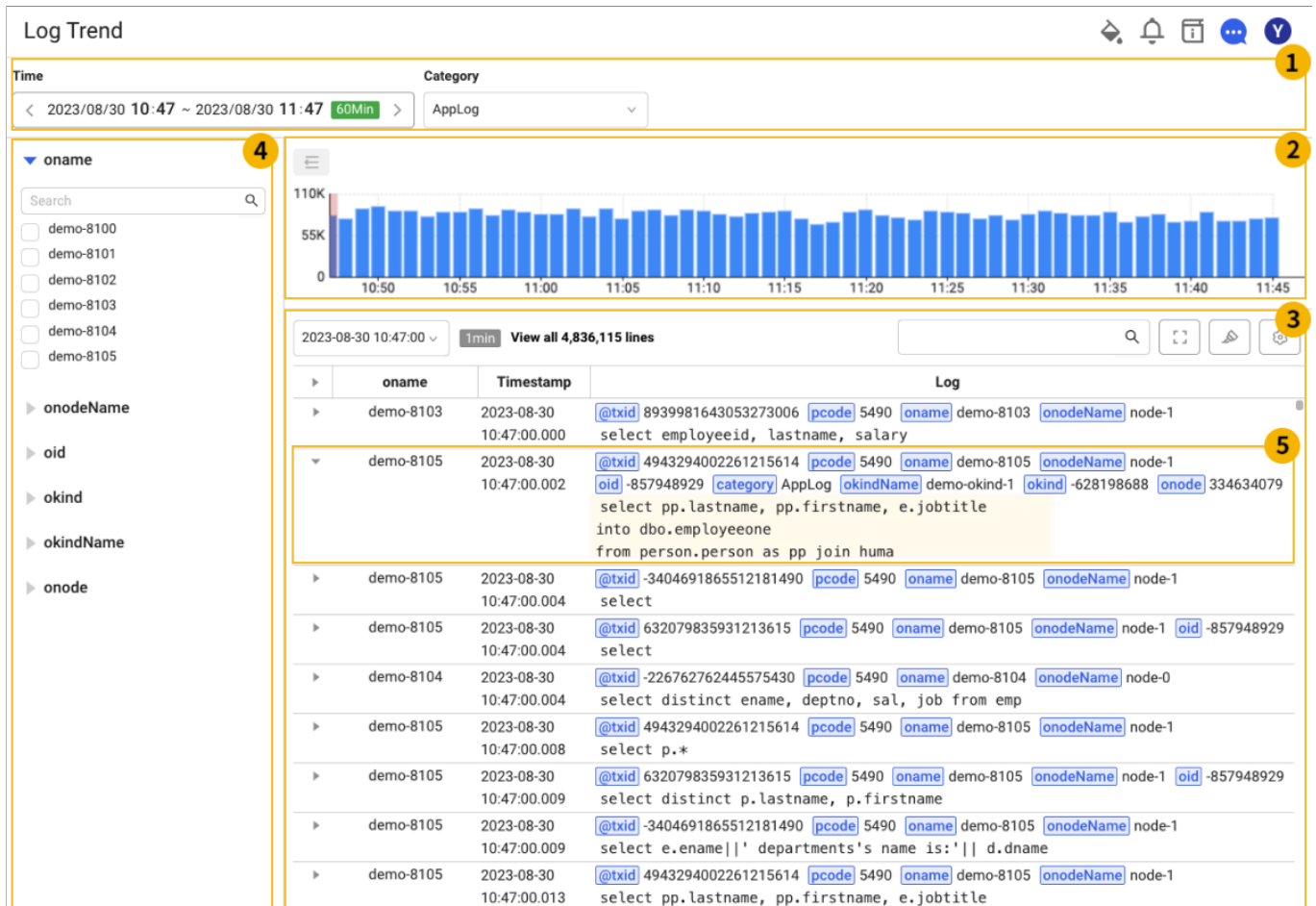
ⓘ Without the **Log Read** role, you cannot enter the menu.

Home > Select Project > **Log** > **Log Trend**

In **Log Trend**, you can check the occurrence pattern of a specific error type through the occurrence count trend of the logs classified by type and see the log data details by time. You can quickly see the desired log through the highlighting feature. You can also check the trends of the collected logs by category.

The major terms are as follows:

- **Category:** Unit for log collection and inquiry
- **Content:** Log messages
- **Search Key:** It is generated through the log parser settings.
- **Tag:** Key to search the collected logs



Request Data

- If the scroll reaches the bottom, the next data is retrieved.
- In 1, specify the **time** and collection unit, **category**, of the log data to be searched.
- If the **category** is changed, the logs that correspond to the selected category are searched. You can see the logs in 2 bar chart and 3 log table.
- If you click a bar in 2 bar chart, the logs are viewed that correspond to the time range of the bar.
- 3 You can see the total number of logs viewed from the upper left of the log table.
- On the upper right of the 3 log table, select the  **full screen** icon, you can see the **log** and **collection time** in the full screen.

- **4** In Side menu, you can see the logs filtered by tags. Up to two search keys can be selected, and multiple search values can be selected.
- If the agent option is set, the log levels are collected and the colors for each log level appear as follows:

2023-12-18 14:31:02.563	level	INFO	pcode	2277	agenttime	1702877462042	oid	778873916	category	AppLog	loggerName	io.home.test.logback02starter.base.web.LogbackControllerGreeting
INFO log in our greeting method.												
2023-12-18 14:31:02.563	level	WARN	pcode	2277	agenttime	1702877462043	oid	778873916	category	AppLog	loggerName	io.home.test.logback02starter.base.web.LogbackControllerError
io.home.test.logback02starter.base.errors.exception.ApiException: [2204] Process failure. Please try again.												
2023-12-18 14:31:02.586	level	error	pcode	2277	agenttime	1702877462043	oid	778873916	category	AppServer	event_status	error
Servlet.service() for servlet [DispatcherServlet] in context with path [/] threw exception [Request processing failed: io.home.test.logback02starter.base.error]												
2023-12-18 14:31:02.586	level	error	pcode	2277	agenttime	1702877462057	oid	778873916	category	AppServer	event_status	error
Servlet.service() for servlet [DispatcherServlet] in context with path [/] threw exception [Request processing failed: io.home.test.logback02starter.base.error]												
2023-12-18 14:31:02.586	level	error	pcode	2277	agenttime	1702877462081	oid	778873916	category	AppServer	event_status	error
Servlet.service() for servlet [DispatcherServlet] in context with path [/] threw exception [Request processing failed: io.home.test.logback02starter.base.error]												
2023-12-18 14:31:02.586	level	error	pcode	2277	agenttime	1702877462087	oid	778873916	category	AppServer	event_status	error
Servlet.service() for servlet [DispatcherServlet] in context with path [/] threw exception [Request processing failed: io.home.test.logback02starter.base.error]												

① Setting the agent options

- The agent options are as follows:

```
# whatap.conf
weaving=log4j-2.17
weaving=logback-1.2.8
```

- In Java agent 2.2.22 or later, it can be used when setting log4j-2.17 or logback-1.2.8 in the weaving setting. It is required to restart the agent.
- The log level is determined based on the `level` and `type` among the parsed keywords. When there are keys parsed as `level` and `type`, and the parsed values have **FATAL**, **CRITICAL**, **ERROR**, **WARN**, **WARNING**, and **INFO**, the log level color appears.

- ① You can change the column width by dragging an edge of the log table column.

Viewing personal information

If you have the **log personal information query** role, you can display de-identified personal information by enabling the **Display personal information**  toggle button.

- For more information about the **log personal information query** role, see [the following](#).
- For more information about **De-identification of personal information**, see [the following](#).

Checking the log content

! What is content?

Content means the log messages.

- The first line of the log column is the log's parsed key and value, and the second line is the log's content.
- **3** For each row (log) of the log table, the **More** button exists. If you select the **More** button, you can check the entire content of the log like **5**.

Configure Highlight

- The highlight function is provided to easily identify the desired keywords.
- **3** In the **input field** on the right, enter the keyword to highlight. You can check the highlighted keywords like this.

Content Highlight

The highlight function allows you to easily identify desired keywords in log content. You can filter with single or multiple keywords.

1. Enter the keyword you want to highlight in the keyword input field.
 - Example: `select`
2. The searched keyword will be highlighted in the Content of the log list.
 - If you click the **Log full screen** icon, you can view **Log** and **Timestamp** in fullscreen.
 - You can navigate between highlighted keywords using shortcuts.
 - `Enter` : Move to the next row
 - `Shift+Enter` : Move to the previous row
 - `^ / v` icons: Move between rows
 - `Cmd + F` (Mac) / `Ctrl + F` (Windows): Search keywords

Multi-keyword condition

When highlighting with multiple keywords, write as follows:

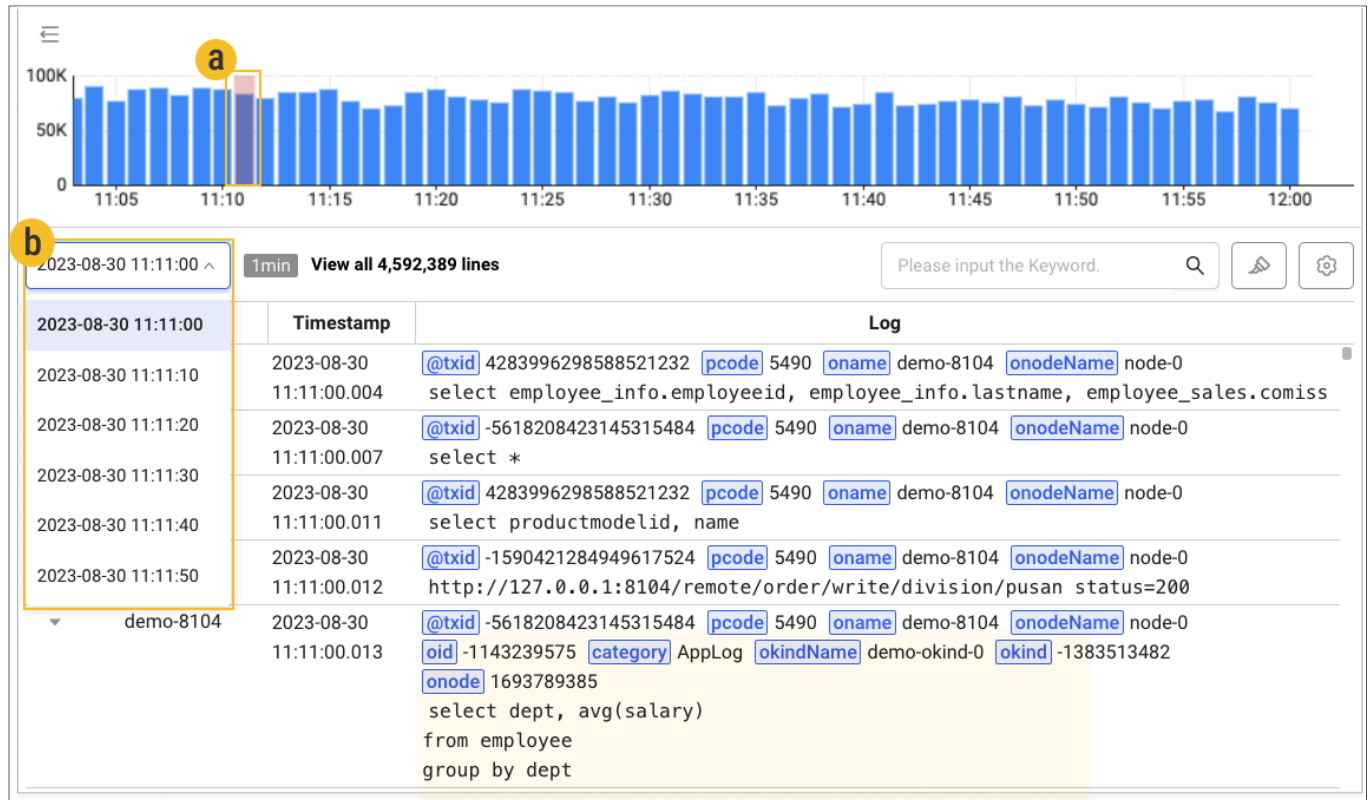
String to input	Description	Result
a b c	Separate each keyword with a space.	a, b, c
"Whatap is good."	To include spaces in keywords, wrap them in "'" or "\"".	Whatap is good.
"Whatap\\ is good."	If \ is included in the keyword surrounded by "", enter it as \\.	Whatap\ is good.

Setting the highlight color

By selecting  **Highlight**, you can set the keyword and its color for highlighting.

- In addition, enter the keyword to set the color for.
- If you click **Color** on the left of the input field, available colors appear.
- Basically, different highlighting by the level (**WARN**, **ERROR**, **FATAL**) is applied.
- The settings are saved in **Project Rules**.

Viewing the logs with a chart



- You can click or drag the desired time range bar on the bar chart (a) to view the logs for the time range.
- You can use the (b) time selection option on the upper left of the log table under the bar chart to further refine your log search in the selected time zone:
 - 1min: interval (spacing between bars on the chart)
 - Select a time option: Time zone whose time range is divided into 6 segments.

Setting the table

Column settings

Right of the (3) area, select **Column settings**. You can add columns or set order.

- **Adding columns**

You can add columns to the table by selecting tags. If you unselect the **log** column, you cannot see **Log view detail setting**. Be sure to select at least one column.

- **Setting the column order**

When you add a column, the column is added to **Display columns**. Drag a desired column to change the order of the columns.

Log view detail setting

On the right of 3, select  **Log view detail setting**. **content** and **Tag** are all checked and two items are also displayed. Be sure to select any of **content** and **Tag**.

Unchecked items are not displayed in the table. If you deselect the **Tag** as follows, the log's **Tag** does not appear in the table.

Log table tag invisible

When you add tags in the **Tag management** list, the tags in the log are listed in the order you added. You can change the order of the tags by dragging them. If you disable an added tag, the disabled tag does not appear.

- **Column settings** and **Log view detail setting** can be used in **Live Tail**, **Log Search**, and **Log Trend**.
- In the same project, the **Live Tail**, **Log Search**, and **Log Trend** menus share **Column settings** and **Log view detail setting**.

Log Search

ⓘ Without the **Log Read** role, you cannot enter the menu.

Home > Select Project > **Log** > **Log Search**

In **Log Search**, you can search a large number of integrated logs under various conditions and specify the desired logs. Multiple search conditions can be specified with parsed keys and values, so only the log data that matches the desired conditions are extracted.

The searched log data is imported to the dynamic page line by line. If the scroll reaches the bottom, the next data is automatically retrieved and displayed.

The major terms are as follows:

- **Category:** Unit for log collection and inquiry
- **Content:** Log messages
- **Search Key:** It is generated through the log parser settings.
- **Tag:** Key to search the collected logs

1 Log Search

Time: < 2023/08/30 11:22 ~ 2023/08/30 12:22 60Min > Filter: category AppLog Enter Filter and press enter to add. category == AppLog

2 AppLog (4,517,902)

AppLog @txid oid okind okindName oname onode onodeName

AppStdErr (461)

AppStdErr @txid oid okind okindName oname onode onodeName

AppStdOut (19,839)

AppStdOut oid okind okindName oname onode onodeName

3 Of the total 4,538,718 results, 9999+ match your search. Oldest Newest Display tag Please input the Keyword.

▶	oname	Timestamp	Log
▶	demo-8101	2023-08-30 11:22:00.019	@txid -2940368816900096605 pcode 5490 oname demo-8101 onodeName node-1 http://127.0.0.1:8101/remote/account/pickup/dept/daegu status=200
▶	demo-8101	2023-08-30 11:22:00.025	@txid -6539276394531757588 pcode 5490 oname demo-8101 onodeName node-1 select ename, job, sal + 100 from emp
▼	demo-8101	2023-08-30 11:22:00.020	@txid -877561626 pcode 5490 oname demo-8101 onodeName node-1 oid -877561626 category AppLog okindName demo-okind-1 okind -628198688 onode 336434079 select distinct pp.lastname, pp.firstname from person.person pp join humanresources.employee e on e
▶	demo-8101	2023-08-30 11:22:00.032	@txid -6539276394531757588 pcode 5490 oname demo-8101 onodeName node-1 select productid, avg(orderqty) as averagequantity, sum(linetotal) as total
▶	demo-8101	2023-08-30 11:22:00.038	@txid -3664464458700254001 pcode 5490 oname demo-8101 onodeName node-1 select distinct ename, deptno, sal, job from emp
▶	demo-8101	2023-08-30 11:22:00.032	@txid -3664464458700254001 pcode 5490 oname demo-8101 onodeName node-1 select e.ename, d.dname
▶	demo-8101	2023-08-30 11:22:00.043	@txid -3664464458700254001 pcode 5490 oname demo-8101 onodeName node-1 select productmodelid, name
▶	demo-8101	2023-08-30 11:22:00.049	@txid -3664464458700254001 pcode 5490 oname demo-8101 onodeName node-1 select ename, sal from emp
▶	demo-8101	2023-08-30 11:22:00.054	@txid -3664464458700254001 pcode 5490 oname demo-8101 onodeName node-1 select ename, sal+1000 from emp
▶	demo-8101	2023-08-30 11:22:00.062	@txid -129020984714615108 pcode 5490 oname demo-8101 onodeName node-1 select employee_id, dept, name, age, salary
▶	demo-8101	2023-08-30 11:22:00.060	@txid -3664464458700254001 pcode 5490 oname demo-8101 onodeName node-1 select ename, job, sal + 100 from emp
▶	demo-8101	2023-08-30 11:22:00.067	@txid -3664464458700254001 pcode 5490 oname demo-8101 onodeName node-1 select p.name as productname,
▶	demo-8101	2023-08-30 11:22:00.067	@txid -129020984714615108 pcode 5490 oname demo-8101 onodeName node-1 select e.ename ' departments's name is:' d.dname
▶	demo-8101	2023-08-30 11:22:00.072	@txid -3664464458700254001 pcode 5490 oname demo-8101 onodeName node-1 select salesorderid, sum(linetotal) as subtotal

4

Request Data

- If the scroll reaches the bottom, the next data is retrieved. You can view 10,000 logs at a time.
- **3** You can see the total number of logs viewed from the upper left of the log table.
- Log data can be queried in chronological order. On the upper right of the **3** log table, select a desired search method among **Oldest** and **Newest**.
- After specifying the time range, select **Apply** to finish the search time setting, and then select **Search** to search for data.
- On the upper right of the **3** log table, select the **Log full screen** icon, you can see the **Log** and

Timestamp in the full screen.

- If the agent option is set, the log levels are collected and the colors for each log level appear as follows:

2023-12-18 14:31:02.563	[level]	INFO	[pcode]	2277	[agenttime]	1702877462042	[oid]	778873916	[category]	AppLog	[loggerName]	io.home.test.logback02starter.base.web.LogbackControllerGreeting
2023-12-18 14:31:02.563	[level]	WARN	[pcode]	2277	[agenttime]	1702877462043	[oid]	778873916	[category]	AppLog	[loggerName]	io.home.test.logback02starter.base.web.LogbackControllerError
2023-12-18 14:31:02.586	[level]	error	[pcode]	2277	[agenttime]	1702877462043	[oid]	778873916	[category]	AppServer	[event_status]	error
2023-12-18 14:31:02.586	[level]	error	[pcode]	2277	[agenttime]	1702877462057	[oid]	778873916	[category]	AppServer	[event_status]	error
2023-12-18 14:31:02.586	[level]	error	[pcode]	2277	[agenttime]	1702877462087	[oid]	778873916	[category]	AppServer	[event_status]	error

! Setting the agent options

- The agent options are as follows:

```
# whatap.conf
weaving=log4j-2.17
weaving=logback-1.2.8
```

- In Java agent 2.2.22 or later, it can be used when setting log4j-2.17 or logback-1.2.8 in the weaving setting. It is required to restart the agent.
- The log level is determined based on the `level` and `type` among the parsed keywords. When there are keys parsed as `level` and `type`, and the parsed values have **FATAL**, **CRITICAL**, **ERROR**, **WARN**, **WARNING**, and **INFO**, the log level color appears.

- ! You can change the column width by dragging an edge of the log table column.

Checking the log content

! What is content?

Content means the log messages.

- The first line of the log column is the log's parsed key and value, and the second line is the log's content.

- **3** For each row (log) of the log table, the **More** button exists. If you select the **More** button, you can check the entire content of the log like **4**.
- If you select a tag in the log, a drop-down menu appears that allows you to perform **Copy**, **Search**, **Exclusion search**, and **Adjacent Log**.

Filter

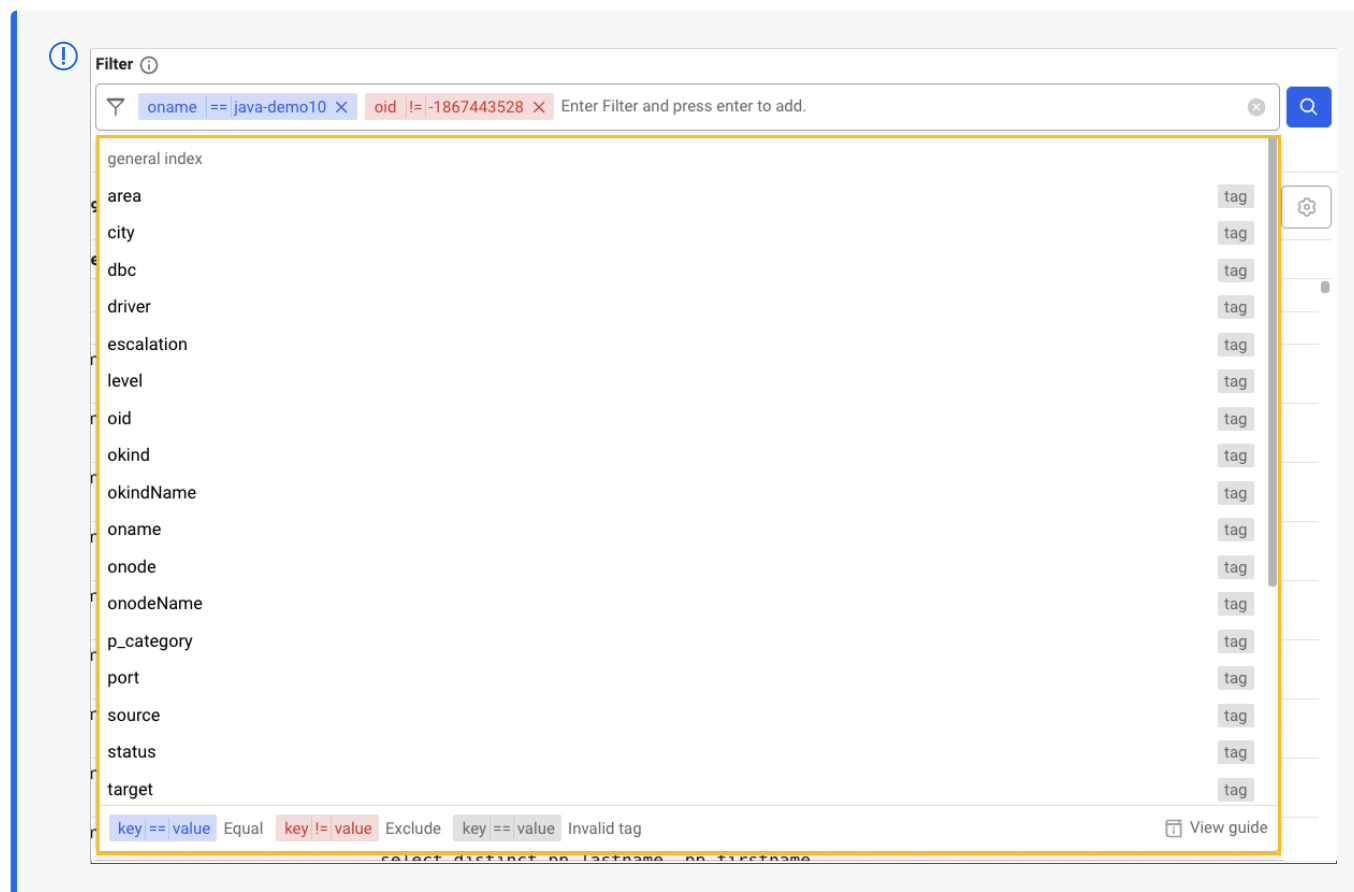
Apply Filter

In **Time selector** on the left of **1**, you can specify the time range. On the right, When a filter is applied, the logs that match the specified conditions are filtered. You can enter multiple filters. OR (||) if the filters have the same tags, otherwise AND (&&).

You can specify a filter by directly entering a value in the input field or clicking the **Filter** field. Filter tags are entered in the following order: **Search key** , **Operator** , and **Search value** . If you select **Search**, the data with the filter applied can be viewed log list area.

Guide UI

The **Guide UI** is provided under the input field as follows: You can close the **Guide UI** via the ESC key when your mouse cursor is over the input field or filter tag.



Entry of the search key, operator, and search value

- Upon entry of the **search key**, recommended values are provided by distinguishing for general index, reserved word index, and numeric index.
- When entering an **operator**, for a general index search key, **==** and **!=** options are provided at the bottom. In case of the numeric index, the **>**, **<**, **<=**, **>=**, **==**, and **!=** options are provided.
- Upon entry of the **search value** for matched search (**>**, **<**, **<=**, **>=**, **==**), it is highlighted in **blue**. In case of excluded search (**!=**), it is highlighted in **red**. Upon entry of the **search value**, you can search using the case-sensitive option.

- If the filter tags exceed 2 lines, you can fold it by selecting the **^ Fold** icon.
- After entering a filter tag, you can use the Shift and Tab keys simultaneously in the input field to go to the previous filter tag.



- After entering a filter tag, you can use the Tab key in the input field to go to the next filter tag.

Adding filter tags

To add a filter tag, type the desired text in the input field and press **Enter** or **Tab**.

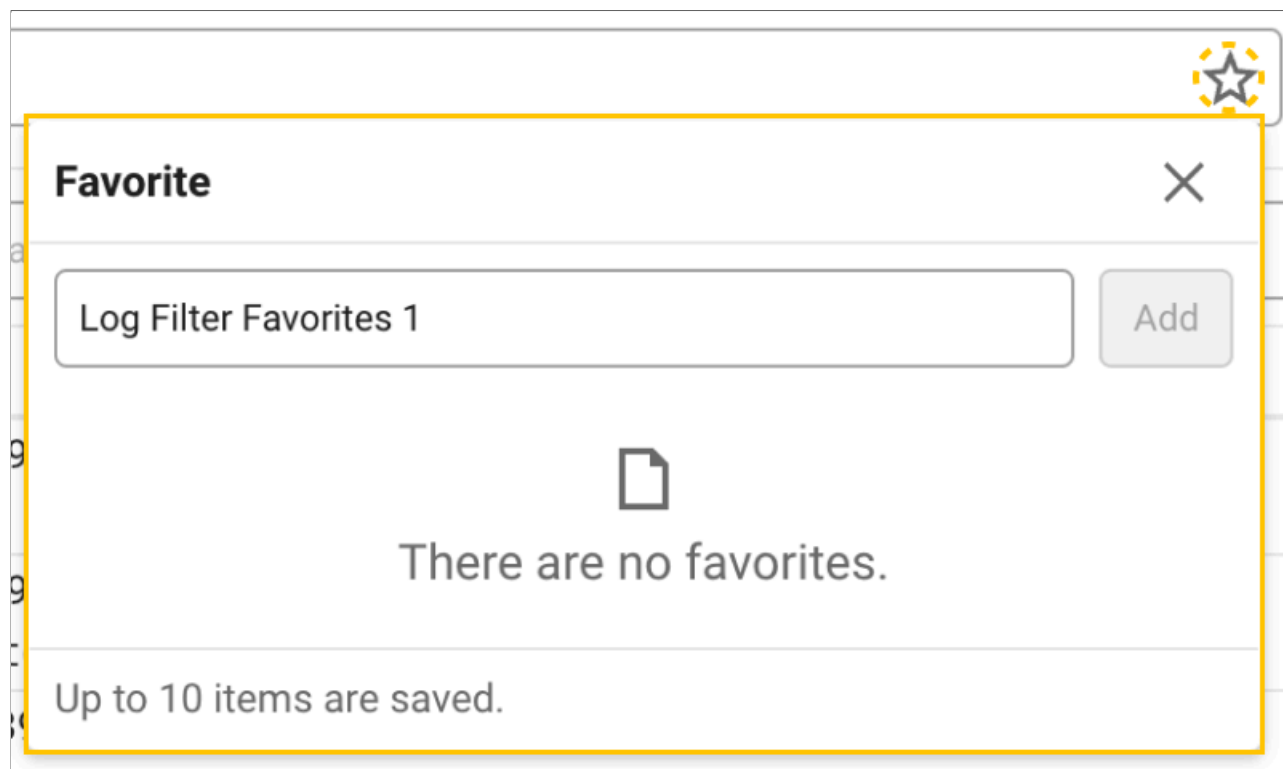
You can also select a recommended value from the **Guide UI** below the input field by clicking it or navigating with the arrow keys.

Deleting filter tags

To remove a single tag, press **Backspace** or click the **X** icon on the tag. To clear all tags at once, click the **X** icon located on the right side of the input field.

Filter favorites

The filter **favorites** feature is provided. You can select the ☆ **Favorites** icon on the right of the input window to **add**, **delete**, and select desired filter search conditions in the favorites. **Favorites** can store up to 50 cases.



Exceptional cases for application of filters

- In the tag with a numeric index (`search key` that ends with `.n`), only numbers can be input for the `search value` .
- Duplicate 'search key' and 'search value' cannot be entered.
- Searching is not possible if there is a tag whose `search key` or `search value` is missing. Invalid tags appear in gray.

Applying non-parsing keyword filters

You can search the logs containing keywords that have not been parsed, or for which no index has been created. In this case, all logs in the specified range are fully scanned. Therefore, the search speed may be slower compared to the indexed key. For standardized log data, it is recommended to search using the index key through **Log parser setting**.

Filter ⓘ

🔍 oid != -1009464250 okind == -398596773 content == *select* Enter Filter and press enter to add.

oid != -1009464250 && okind == -398596773 && content == *select*

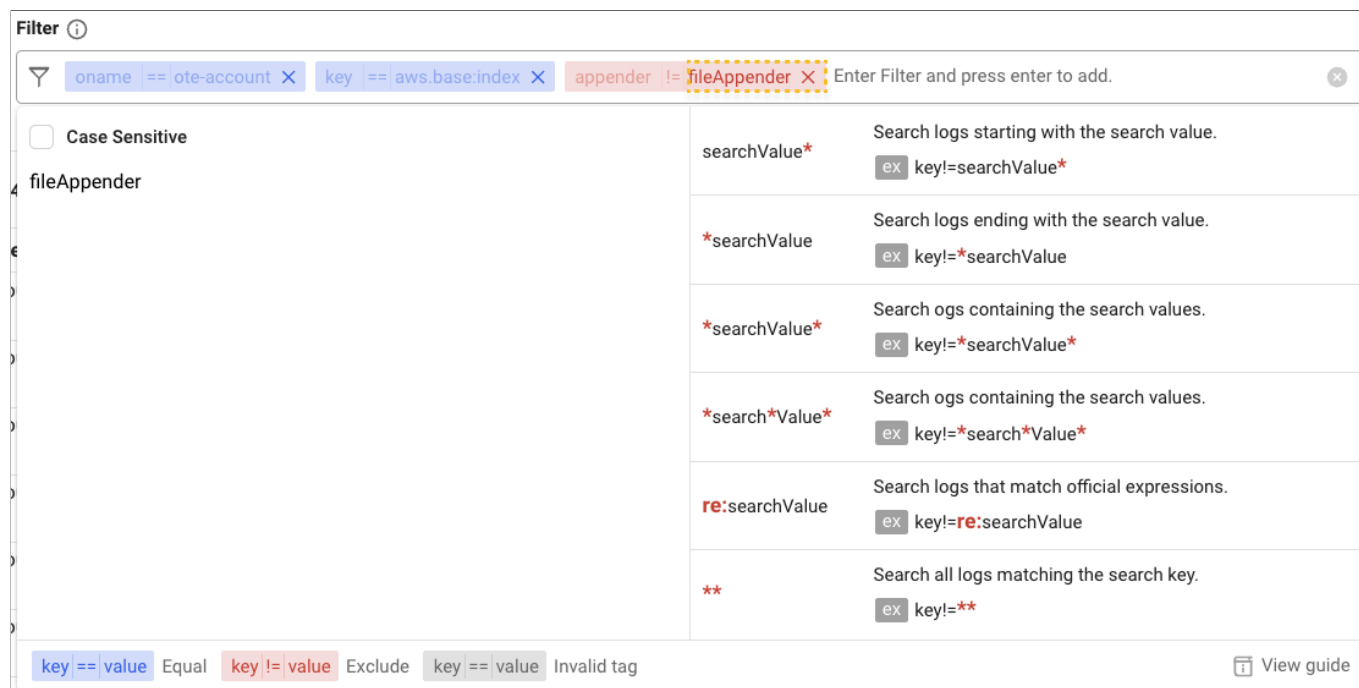
1. Select **Category**. It is required to set the category.
2. In the **Filter** field, enter a desired keyword for a search after leaving spaces on the **content** basis.
e.g. `content *select*`
3. Click **Search** to view the logs. Several logs are viewed first. You can view a maximum of 10,000 logs per each search.
4. You can make additional inquiries by scrolling down and selecting **View more** at the bottom.

The screenshot shows the Log Search interface. At the top, there's a filter bar with the same filters as the previous image. Below it, a message states: "You have viewed 10,500 of the total 36,841 results, and the results retrieved are 9999+." The main area displays a list of logs with columns for time, category, and content. A modal dialog is open, asking "Want to view more logs?" and providing instructions: "Adding a log parser enables search without additional lookups." It includes buttons for "Setting up a log parser" and "View more".

- Among all logs, 10,000 logs are viewed for a server search range. The default range for a server search is 200,000, but the ratio may differ depending on the total log volume.
- For more information about the parser configuration, see [the following](#).

Modifying the filter

Enter a value in the filter and then click the specified value to modify it.



- You can modify the filter by entering text again in the input field.
- You can modify the recommended values by selecting them in the **Guide UI** below the input field.

Search Key

The search key Indicates an identifier for accessing a desired value within the log data. The actual data for the search key is the search value. The tags on the left ② area are the search keys parsed for each category. You can enter filters by selecting tags. The orange tag is the category, and the blue tag is the search key.

For example, **AppLog** and **AppStdOut** in ② are categories. The tags below such as **oid** are search keys parsed. In **Log > Log Configuration**, you can register and set the parsing logic in **Log parser setting** tab with the search key.

Filter input syntax

A tag consists of a search key and a search value. In the following example, the search key is **exception** and the search value is **UnknownHostException**. The example searches for the log data that includes an exception (**UnknownHostException**) that is generated when the server cannot connect the host because the IP address and domain address do not match among the collected log data.



exception

==

UnknownHostException

×

Search key types

Search key types	Search key format	Meaning	Example for Search key and search value	Search example
String keyword	keyword	File Name	- Key: fileName - Value: /data/whatap/logs/yard.log	fileName:/data/whatap/logs/yard.log
Numeric keyword	keyword.n	Response Time	- Key: response_time.n - Value: 2945	response_time.n>=2945
Reserved keyword (predefined keyword)	@keyword	Transaction ID	- Key: @txid - Value: 85459614215434144	-

Common syntaxes

Syntax type	Description	Example
<code>==searchValue</code>	Searches for the logs that match the search value.	<code>exception==RuntimeExceptionexception</code>
<code>!=searchValue</code>	Searches for the logs that exclude the search value.	<code>exception!=RuntimeException</code>
<code>*searchValue</code>	Searches for the logs ending with the search value.	<code>word==*hello</code>
<code>searchValue*</code>	Searches for the logs starting with the search value.	<code>word==hello*</code>
<code>*searchValue*</code>	Searches for the logs that contain the search value	<code>word==*hello*</code>

Syntax type	Description	Example
	in the middle.	
*search*Value*	Searches for the logs that contain the search value.	word==*he*llo*
re:{regex}	Searches for the logs that match a regular expression.	caller==re:^(i\.w\.a\.w\.s\.v\.r\.
**	Searches all logs that match the search key.	

Syntax in case the search key has a numeric value (keyword.n)

The following syntax is supported only when the search key is in `keyword.n` format.

- Search values must only be numbers.
- No prefix is attached to the value of the `.n` keyword. All keywords other than `.n` must be attached with a prefix.

For example, `+>searchValue` is invalid.

Syntax type	Description	Example
>searchValue	Searches for the logs containing values greater than the search value.	response_time.n>3000
>=searchValue	Searches for the logs containing values greater than or equal to the search value.	response_time.n>=3000
==searchValue	Searches for the logs containing the same search value.	response_time.n==3000
!=searchValue	Searches for the logs that contain values different from the search value.	response_time.n!=3000
<searchValue	Searches for the logs containing values less than the search value.	response_time.n<3000

Syntax type	Description	Example
<code><=searchValue</code>	Searches for the logs containing values less than or equal to the search value.	<code>response_time.n<=3000</code>

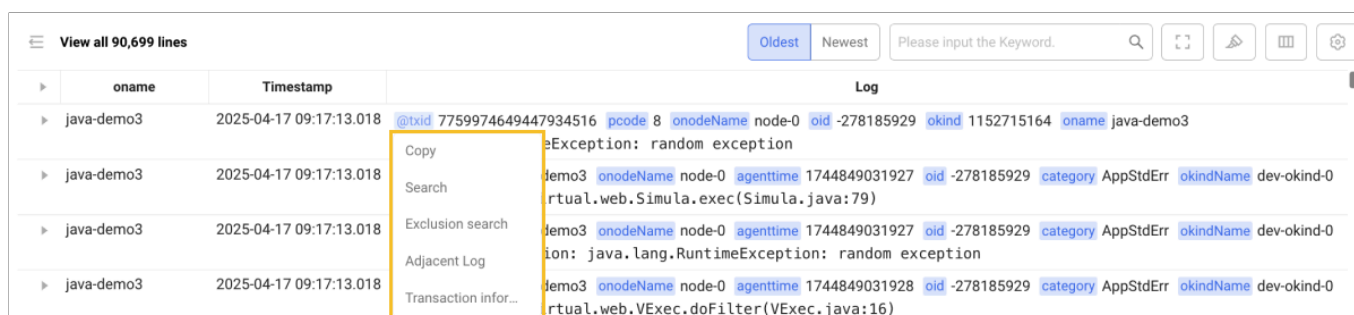
Viewing personal information

If you have the **log personal information query** role, you can display de-identified personal information by enabling the **Display personal information** ☒ toggle button.

- For more information about the **log personal information query** role, see [the following](#).
- For more information about **De-identification of personal information**, see [the following](#).

Log tagging options

When you select a log tag, the following drop-down menu appears. You can find the **Search**, **Exclusion search**, **Adjacent Log** and **Transaction information** options.



- Copy
- Search

If you select the **Search** option, the tags that correspond to **Filter** are entered with the inclusion (`==`) condition.

- Exclusion search

If you select the **Exclusion search** option, the tags that correspond to **Filter** are entered with the exclusion (`!=`) condition.

- **Adjacent Log**

If you select the **Adjacent Log** option, the **Adjacent Log** detail window appears. Search for the logs of the target server and the logs in time zones adjacent to the selected log. You can click **Time selector** to see the logs in adjacent time zones. Reference logs appear with a blue background.

- **Transaction**

If the log contains the `@txid` and `@mtid` tags, you can see the information in the transaction Information details window by clicking the transaction information option.

Content highlighting

The highlight function allows you to easily identify desired keywords in log content. You can filter with single or multiple keywords.

1. Enter the keyword you want to highlight in the keyword input field.
 - Example: `select`
2. The searched keyword will be highlighted in the Content of the log list.
 - If you click the  **Log full screen** icon, you can view **Log** and **Timestamp** in fullscreen.
 - You can navigate between highlighted keywords using shortcuts.
 - `Enter` : Move to the next row
 - `Shift+Enter` : Move to the previous row
 - `^ / v` icons: Move between rows
 - `Cmd + F` (Mac) / `Ctrl + F` (Windows): Search keywords

Multi-keyword condition

When highlighting with multiple keywords, write as follows:

String to input	Description	Result
<code>a b c</code>	Separate each keyword with a space.	a, b, c

String to input	Description	Result
"Whatap is good."	To include spaces in keywords, wrap them in ' ' or " ".	Whatap is good.
"Whatap\\ is good."	If \ is included in the keyword surrounded by "", enter it as \\.	Whatap\ is good.

Setting the highlight color

By selecting  **Highlight**, you can set the keyword and its color for highlighting.

- In addition, enter the keyword to set the color for.
- If you click **Color** on the left of the input field, available colors appear.
- Basically, different highlighting by the level (**WARN**, **ERROR**, **FATAL**) is applied.
- The settings are saved in **Project Rules**.

Setting the table

Column settings

Right of the  area, select  **Column settings**. You can add columns or set order.

- **Adding columns**

You can add columns to the table by selecting tags. If you unselect the **log** column, you cannot see **Log view detail setting**. Be sure to select at least one column.

- **Setting the column order**

When you add a column, the column is added to **Display columns**. Drag a desired column to change the order of the columns.

Log view detail setting

On the right of , select  **Log view detail setting**. **content** and **Tag** are all checked and two items are also displayed. Be sure to select any of **content** and **Tag**.

Unchecked items are not displayed in the table. If you deselect the **Tag** as follows, the log's **Tag** does not appear in the table.

Log table tag invisible

When you add tags in the **Tag management** list, the tags in the log are listed in the order you added. You can change the order of the tags by dragging them. If you disable an added tag, the disabled tag does not appear.

- **Column settings** and **Log view detail setting** can be used in **Live Tail**, **Log Search**, and **Log Trend**.
- In the same project, the **Live Tail**, **Log Search**, and **Log Trend** menus share **Column settings** and **Log view detail setting**.

Log Setting

You can configure the log monitoring related settings in **Log Configuration**. Using the tab at the top, you can use menus for checking the agent settings, determining whether to enable log monitoring, setting the log data retention period and lookup password, registering the log parser, and setting the quick indexes.

- To use the **Activate log monitoring** function, the **Edit project** role is required.
- The **Log edit** role allows you to modify the **Log Configuration** menu other than **Activate log monitoring**.

Starting the log monitoring

At the top, select the **Getting started with log monitoring** tab. If you select the  **View guide** icon and the **View plans** button, the corresponding guide screen appears.

Set up the agent and enable log monitoring

Log monitoring data setting

In the  area, you can see the **Log usage**. You can also change the **Log retention period** and **Log lookup password** settings.

Log retention period

This is the default data retention period to be applied commonly. If not specified, the default value is 1 day. In addition to the option to select a log retention period, you can enter a desired period of time. If not set separately in the **Log usage** list, this log data retention period is applied by default. If you set the data retention periods for each category in the **Log usage** list and select **Initialized**, the default data retention period is reset.

Log usage amount

You can specify the log data retention periods for each category in the **Log usage** list. **Log Count** means the number of logs stacked for the period. For example, **Daily log count** is the number of logs accumulated during the day, and **Expected log count** is the number of logs of the today's log count multiplied by the days of data retention.

You can specify the log data retention period as follows. You can free the space by deleting old data according to the specified period.

- **Trial Project**

You can select 1, 2, or 3 days for the data retention period.

- **Paid Project**

For the data retention period, you can select one of 1 day, 2 days, 3 days, 4 days, 5 days, 6 days, 7 days, 10 days, 30 days, and 40 days.

- **Data size-based billing**

The price differs depending on the data retention period.

For example, if an average of 2 million logs per day accumulates and the data retention period is set to 3 days, an average of 6 million logs is kept on the collection server and subject to billing.

Log lookup password

To enhance security, set the **Log lookup password**. The log lookup password is optional. If you are using a log lookup password, enter the password to go to the log screen.

In case you forgot the password

If you have the **Edit Log** role, you can modify it with a new password in **Log Configuration**.

Event Configuration

Home screen > Select project  **Alert** > **Event Configuration**

In **Event setting**, you can configure and manage events based on anomaly conditions that may occur in monitored systems or applications. When the configured conditions are met, notifications can be delivered through your preferred channels such as email, SMS, messengers, or app push notifications. You can tailor event configurations according to your monitoring objectives.

Event Config Overview

Event configuration menu for Java alerts and how to use it.

Event Notification

Configure channels and recipients for Java alert events.

Event History

View Java event history and monitor trends.

Home Screen > Select Project >  **Alert** > **Event Configuration**

Event setting provides features to configure and manage events based on anomaly signals that may occur in monitored systems or applications. When configured conditions are met, you can receive notifications through your preferred channels such as email, SMS, messengers, or app push notifications. You can design event configurations according to your monitoring objectives.

 Event Config Overview

Event configuration menu for Java alerts and how to use it.

 Event Notification

Configure channels and recipients for Java alert events.

 Event History

View Java event history and monitor trends.

 Reference

- When configuring events, use appropriate threshold values to prevent unnecessary event generation.
- When setting up notifications, consider enabling recovery notifications (alerts sent when issues are resolved) to support effective operational response.

Common Features

All event configurations can be downloaded in **JSON or Excel format**. By sharing the same type of product configuration, you can reduce repetitive event setup work.

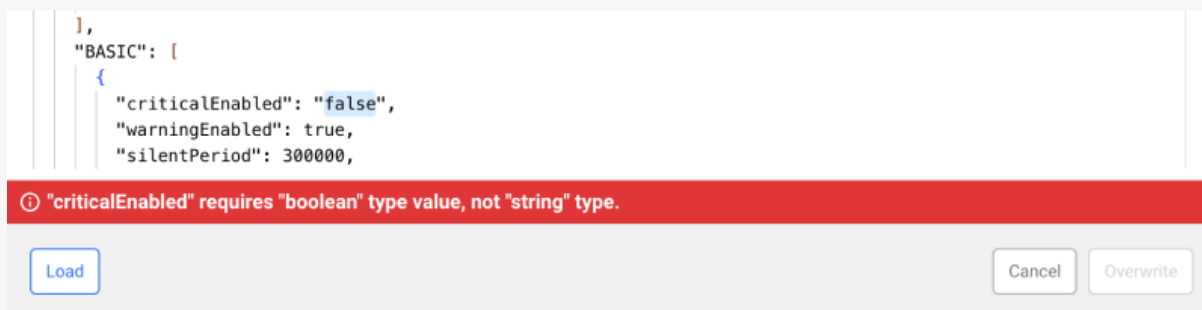
- **Batch JSON Edit:** Edit all event conditions in bulk using JSON format.
- **Batch JSON Download:** Save all currently registered event conditions as a JSON file.
- **Batch Excel Download:** Download the event list as an Excel file for external reporting or backup purposes.

Batch JSON Edit

1. Click the [Batch JSON Edit] button at the top of the screen.
2. Import a JSON file downloaded via the **Batch JSON Download** feature, or edit the content directly.
3. Click the [Overwrite] button to save.
4. When a confirmation message appears, click the [Overwrite] button.

⚠ After overwriting, you cannot restore the previous event configuration.

- Arbitrarily modifying the values of the `productType` or `platform` properties may cause events to malfunction in other projects.
- The JSON editor includes built-in validation. If the format or values are invalid, an error message appears below the editor and the [Overwrite] button is disabled.



JSON Data Structure



```
{
  "productType": "{productType}",
  "platform": "{platform}",
  "events": {
    "HITMAP": {
      ...
    },
    "LOG_REALTIME": [
      {...}
    ],
  },
}
```



```
"METRICS": [
  {...}
],
"COMPOSITE_LOG": [
  {...}
],
"BASIC": [
  {...}
],
"COMPOSITE_METRICS": [
  {...}
],
"ANOMALY": [
  {...}
]
}
}
```

JSON Field	Description
productType	Product type (e.g. APM, Infra)
platform	Platform information (e.g. java, node)
events	Event data
⬠ HITMAP	Hitmap pattern
⬠ LOG_REALTIME	Real-time log events
⬠ METRICS	Metrics
⬠ COMPOSITE_LOG	Composite log events
⬠ BASIC	Basic events
⬠ COMPOSITE_METRICS	Composite metrics

!	JSON Field	Description
	◇ ANOMALY	Anomaly detection

Batch JSON Download

Click the [Batch JSON Download] button at the top of the screen.

- The JSON file name follows the format `integrated-event-YYYYMMDD.json`.
- The structure of the JSON data and the sub-properties under the `events` attribute may vary depending on the product (`platform`) or product type (`productType`).

! If you arbitrarily modify the values of the `productType` or `platform` properties in the JSON data, events may not function correctly in other projects.

Batch Excel Download

Click the [Batch Excel Download] button at the top of the screen.

Event Notification Settings

Home > Select Project >  **Alert** > **Notifications**

Event Notification Settings per User

 Information about other members in **Event notification settings per-user** can only be viewed by members with the **Member Management** permission. For all other members, required information is masked.

- Mobile device management for bulk event notification receipt settings and access control can be managed in **Account Information**.
- For detailed information about member permissions, refer to the [following document](#).

Notification Methods

Alerts are available via [Email](#), [SMS](#), [WhatsApp](#), and [Mobile](#).

In the **Event notification settings per-user** section, select the checkbox for the desired notification method to receive alerts. If you clear the checkbox for a notification method, alerts will not be sent through that channel.

 All users except the project super administrator can change only their own notification settings.

Email Alerts

Email alerts are sent to the email address provided during sign-up.

SMS Alerts

You can receive alerts via SMS.

ⓘ The SMS alert feature is available only for paid projects.

1. Click your **profile icon** at the top right of the screen, then go to **Manage Accounts**.
2. In **Account Information**, click the [General Mobile Phone] button in the **User Phone Number** section.
3. Enter the **phone number** that will receive the verification code.
 - Only **Korean mobile phone numbers** can be registered to receive SMS alerts.
4. Click the [Save] button.

✔ Change or delete the registered phone number

- To change the phone number, click the [Change] button. SMS verification is required.
- To delete the phone number, click the [Delete] button. After deletion, you will no longer receive alerts via SMS.

WhatsApp Alerts

You can receive alerts via WhatsApp.

ⓘ The WhatsApp alert feature is available only for paid projects.

1. Click your **profile icon** at the top right of the screen and go to **Manage Accounts**.
2. In **Account Information**, click the [WhatsApp] button in the **User Phone Number** section.
3. Enter the **phone number** that will receive the verification code.
4. Click the [Send Verification Code] button.
5. Enter the 6-digit verification code sent to your WhatsApp application.
 - If you do not receive the verification code within 10 minutes, click the [Resend Verification Code] button.
6. Click the [Verify] button.

Mobile Alerts

Users who need to receive mobile alerts must install the WhaTap application on their mobile device and log in.

- Download the WhaTap application – [Android](#)
- Download the WhaTap application – [iOS](#)

Repeated Alerts (Escalation) Settings

If the alert condition is not resolved after an alert is triggered, you can configure repeated alert intervals based on the time of the initial alert. This setting applies only to **Critical** level alerts.

For example, to repeat alerts immediately (0 minutes), 1 hour later, and 1 day later after the initial alert, enter `0,1H,1D` in the **Repeated Alerts (Escalation)** column.

1. Go to **Alert > Notifications** and navigate to the **Event notification settings per-user** section.
2. Configure the settings in the **Repeated Alerts (Escalation)** field.
 - **M**: minutes, **H**: hours, **D**: days. If the unit is omitted, the value is interpreted as minutes.
 - Enter a number or a number with a unit (**M**, **H**, **D**). If the input format is invalid, an error message is displayed.
3. Click the [Save] button. The settings will not be applied unless they are saved.

Event Reception Tags

When configuring events, you can select **event reception tags** to send alerts to project members and 3rd-party plugins associated with those tags. If no event reception tag is set, alert notifications are sent to **all project members**.

Add / Edit / Delete Event Reception Tags

 Event reception tags that are **currently applied to an event** cannot be deleted.

1. Go to **Alert > Notifications**.

2. In the user list under the **Event notification settings per-user** section, click + **Add Tag** or +.
3. In the **Event reception tag** window, click + **Create New Tag**.
4. In the **Create Tag** dialog, enter a tag name, select a color, and click the [Create Tag] button.
5. To edit or delete a created tag, click the  icon in the tag list.
6. Select the desired tag from the **Tag List** in the **Event reception tag** window to apply it.

Remove Event Reception Tags

1. Go to **Alert > Notifications** and open the **Event notification settings per-user** section.
2. Click + in the **Event reception tag** field for the user whose event reception tag you want to remove.
3. In the **Event reception tag** window, click the **X** on the tag you want to remove. The tag will be removed.

Receive Alerts Without Reception Tags

When configuring events, you can select event reception tags to send alerts to project members and 3rd-party plugins associated with those tags.

- To receive alerts **without event reception tags**, enable the **Receive alerts without reception tags** option.
- If you want to receive **only alerts that have event reception tags**, disable this option.

 To stop receiving all alerts, disable this option and make sure that no **Event reception tag** is selected.

Selecting the event reception mode

In the **Event notification settings per-user** section, you can select the event alert reception scope per member or third-party channel from the following three modes. The reception mode is selected in the **Event reception tags** column of each member or channel row.

Mode	Description
Receive all alerts	Receive all alerts regardless of the event reception tags
Receive alerts for selected tags	Receive only alerts that have the selected event reception tags. Select at least

Mode	Description
only	one tag
Receive no alerts	Receive no event alerts. The selected tags are preserved without being deleted

- In the **Receive alerts for selected tags only** mode, select at least one tag to receive alerts in the tag selection area.
- To also receive alerts that have no event reception tag specified, select the **Unclassified** virtual tag in the tag selection area. The **Unclassified** tag can be selected or deselected like a normal tag, but the tag itself cannot be edited or deleted.

ⓘ In the **Receive no alerts** mode, the corresponding member or channel receives no event alerts at all. The selected tags are preserved without being deleted, so if you switch back to the **Receive alerts for selected tags only** mode, you can use the previously selected tag settings as is.

Select Notification Language

You can choose the language in which you receive alert notifications.

1. Go to **Alert > Notifications**.
2. In the user list under the **Event notification settings per-user** section, click the ► button at the far left of the user whose language you want to configure.
3. Select the **Notification Language**.

Day and Time-based Notification Settings

You can configure whether a user receives notifications by day of the week and time of day.

1. Go to **Alert > Notifications**.
2. In the user list under the **Event notification settings per-user** section, click the ► button at the far left of the user whose day and time settings you want to configure.

3. Select the days and/or enter the time ranges during which you want to receive alerts.

- Settings can be configured separately for each notification method.

3rd-party Plugins

You can receive alert notifications through external applications.

- Supported applications:** Slack, Telegram, Teams, Teams Workflows, Jandi, AlertNow, Webhook, Webhook JSON, Opsgenie, PagerDuty, LINE, ilert

ⓘ Internal messengers that are not included in WhaTap Labs' supported scope can be integrated via standard **Webhook** or **Webhook JSON**.

Table | 3rd-party Plugin List

Item	Description
Plugin Name	Name of the plugin
Auth Key	Authentication key
Auth Value	Authentication value
Receive Level	Notification level: Warning, All, or Critical
Repeated Alerts (Escalation)	Re-notification interval from the initial alert time if the issue is not resolved - Applies only to Critical alerts
Smart Message (AI)	AI-powered notifications available when adding the Telegram plugin - Supported languages: Korean, English, Japanese
Event Reception Tag	Whether to receive alerts for specific members or teams in the project
Delete	Delete the 3rd-party plugin

Add a 3rd-party Plugin

1. Go to **Alert > Notifications**.
2. In the **3rd-party Plugins** section, click **Add**.
3. In the **Add 3rd-party Plugin** window, select the desired service tab.
4. Follow the instructions for the selected service (see [How to add 3rd-party plugin notifications](#)).
5. Click **Test Receive** to verify the configuration.
6. When finished, click **Add** or **Register**.

How to Add 3rd-party Plugin Notifications

Slack

- ⓘ In the **Add 3rd-party Plugin** window, click the [**Add To Slack**] button on the **Slack** tab to register **WhaTapNotiBot** to your Slack team and channel.

Telegram

- ⓘ
1. In **Telegram**, search for **@WhaTapApmNotiBot**.
 2. Select the bot and click the [**Start**] button.
 3. In the chat window, enter the `/license` command followed by the access key of the project for which you want to receive alerts.

Telegram

ⓘ Available commands

- `/help` : View the list of commands
- `/license` : Enter the access key of the project to receive alerts

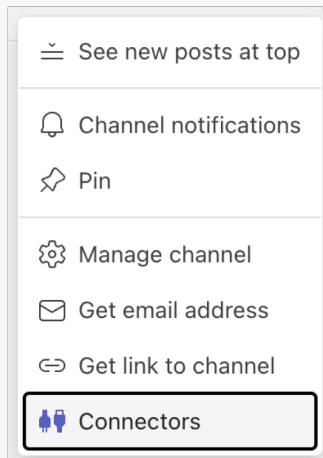


- `/remove` : Enter the project code for which you want to stop receiving alerts
- `/list` : View the list of project codes currently receiving alerts

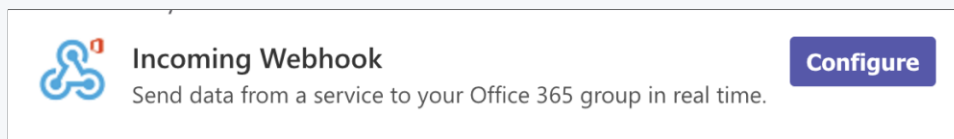
Teams



1. Go to the **connector** of the channel where you want to receive alerts.



2. Add a **Webhook** and click the [Configure] button.



3. Enter a webhook name and click the [Create] button.



Incoming Webhook

[Send feedback](#)

The Incoming Webhook connector enables external services to notify you about activities that you want to track. To use this connector, you'll need to create certain settings on the other service, which needs to support a webhook that's compatible with the [Office 365 connector format](#).

Fields marked with * are mandatory

To set up an Incoming Webhook, provide a name and select Create. *

Customize the image to associate with the data from this Incoming Webhook.

Upload Image



Default Image

Create

Cancel

4. Copy the generated **Webhook URL**.

Copy the URL below to save it to the clipboard, then select Save. You'll need this URL when you go to the service that you want to send data to your group.

<https://whatap.webhook.office.com/webhook>



Url is up-to-date.

Done

Remove

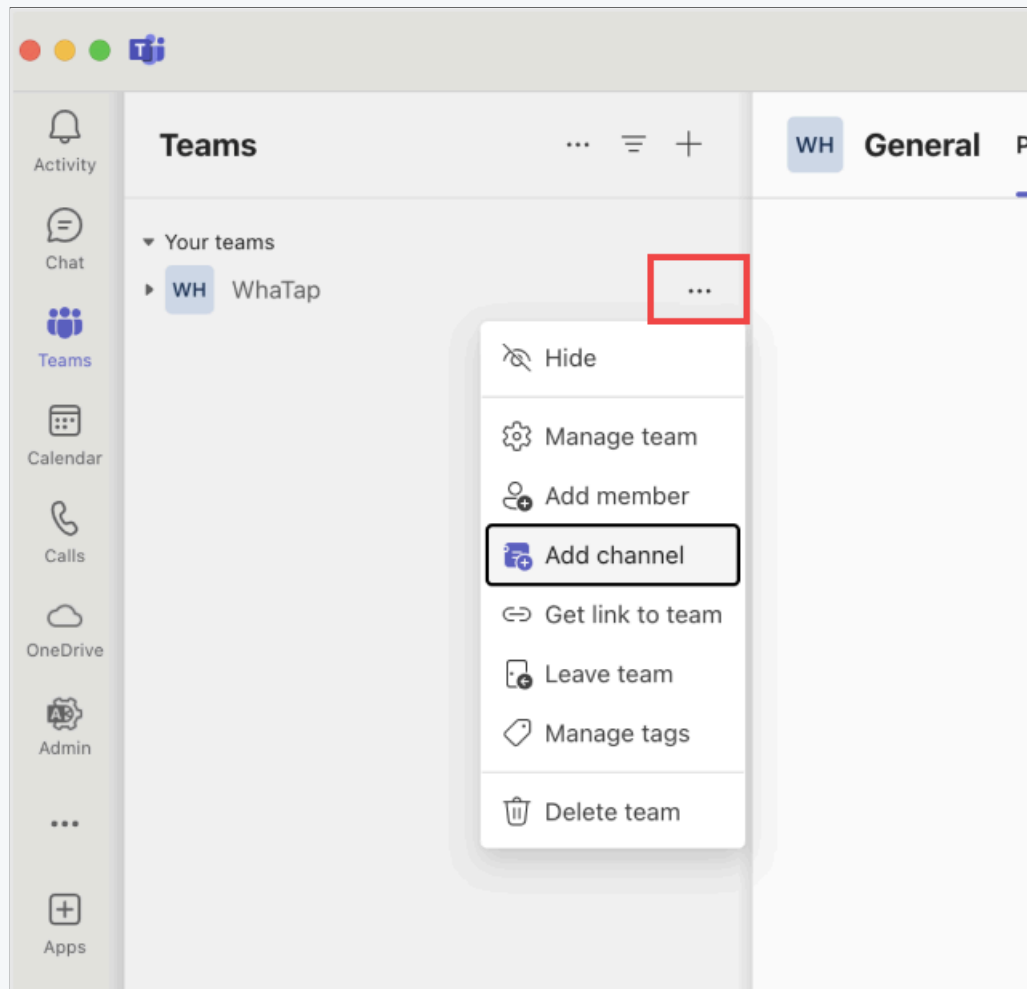
5. Return to the **Add a third-party plug-in** window in the WhaTap Monitoring service, and enter values for **Channel Name** and **Webhook URL** at the bottom of the **Teams** tab.
6. Click the [**Registration**] button.

Teams Workflows



- ⓘ To use workflows, you need an account that can access **Microsoft Teams** and a **Business plan** subscription.

1. In the **Teams** menu, click **More actions (...)** > **Add channel**.



2. In the **Create a channel** window, enter the **Channel name** and **Description**. For **Channel type**, select **Standard**, then click the [Create] button.



Create a channel

Channel name *

WhaTap Event Channel

Description

Help others find the right channel by providing a description

Choose a channel type * ⓘ

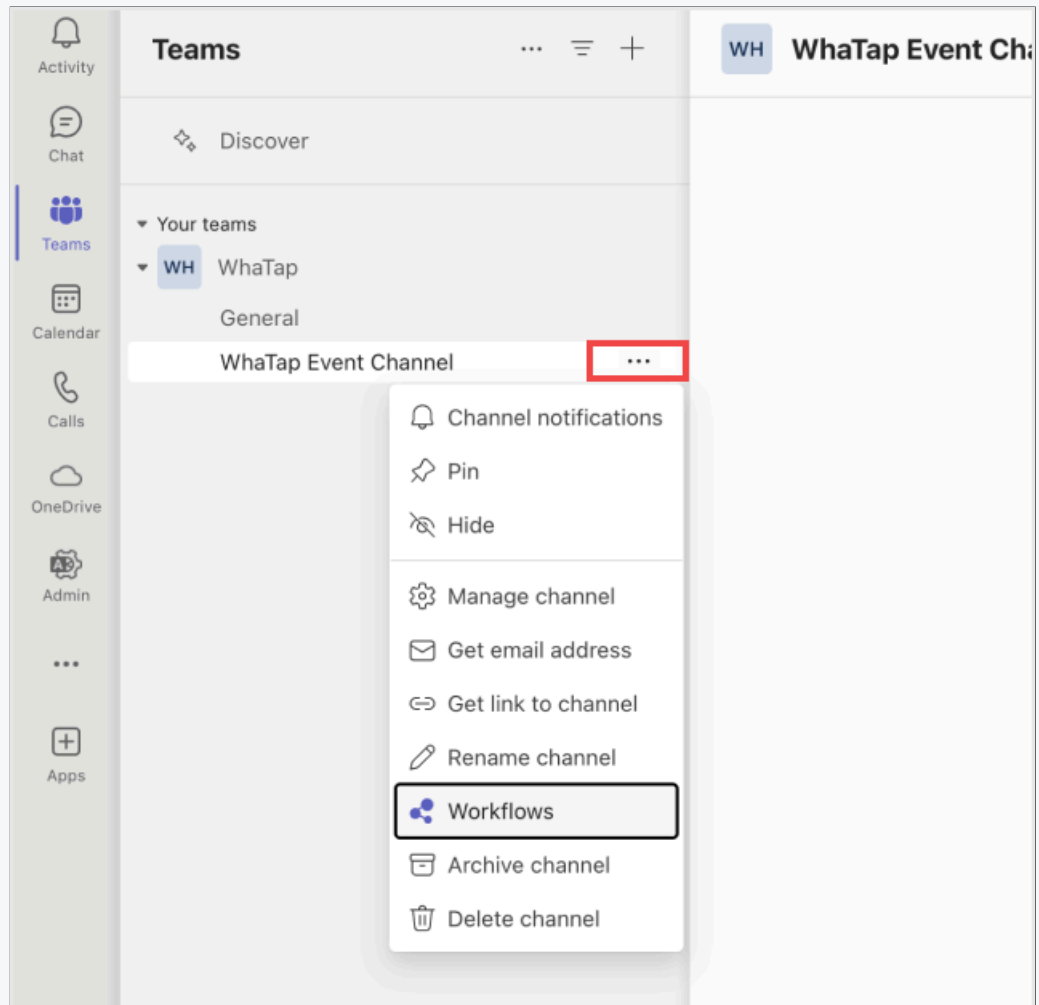
Select

- Standard**
Everyone on the team has access.
- Shared**
People or teams in your org or outside your org can have access.
- Private**
Specific people on the team have access.

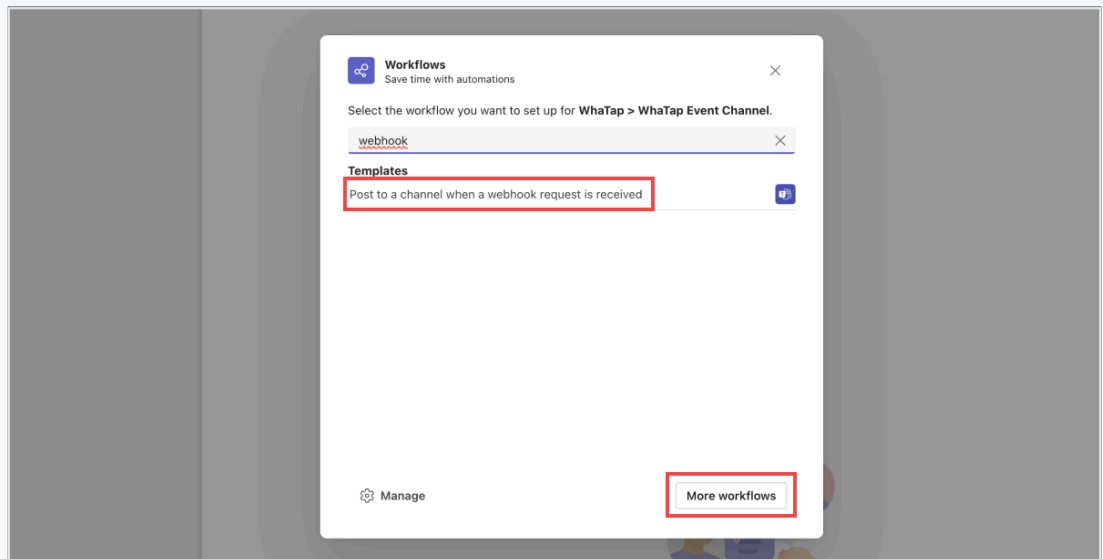
Cancel Create

ⓘ If you select **Shared** as the channel type, workflows cannot be used. If you select **Private**, workflows can be added but notifications cannot be received.

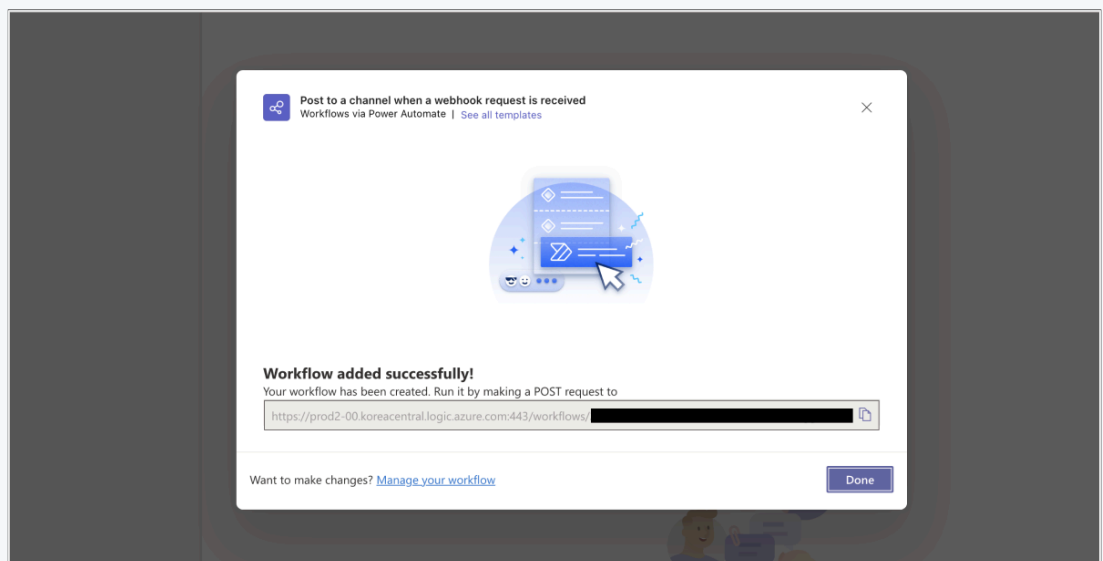
3. Click **More actions (...)** of the created channel, then select **Workflows**.



4. In the **Workflows** dialog, select **Post to a channel when a webhook request is received** from the workflow list, then click the [Add workflow] button.



5. Check the **Name** and **Connection** status, then click the [Next] button.
6. Review the **Details**, then click the [Add workflow] button.
7. The workflow has been added. Copy the generated URL.



8. Return to the **Add a third-party plug-in** window in the WhaTap Monitoring service, and enter values for **Channel Name** and **Webhook URL** at the bottom of the **Teams Workflows** tab.



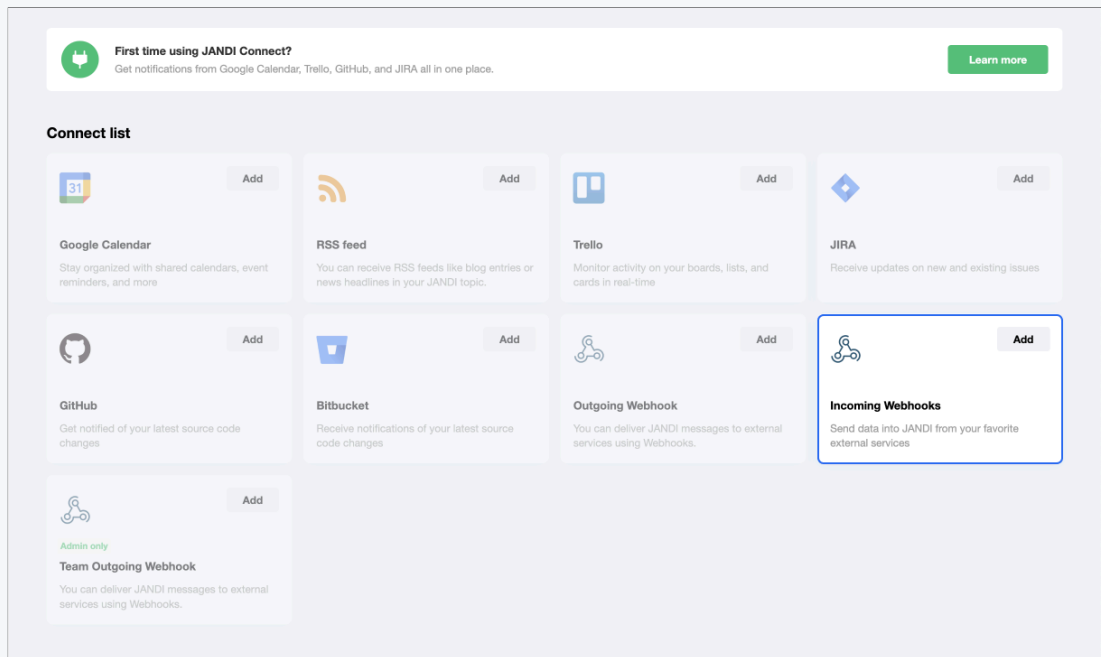
9. Click the [**Registration**] button.



Jandi



1. Go to **Jandi Connect settings** in the topic or chat where you want to receive notifications.
2. In the **Connect list**, find **Webhook Receiver (Incoming Webhook)** and click the [**Connect**] button.



3. Click the [**Add Webhook Receiver**] button.
4. Copy the Webhook URL from the **Register Webhook URL** section.



Connect settings

Profile settings

Change the name or profile image that appears for messages from this integration

 Incoming Webhooks

Topic / JANDI selection

Please select a topic or DM to receive integration messages. You change it after saving it.

JANDI

Webhook URL Registration

Webhook URL

Copy the generated Webhook URL and register with the service.

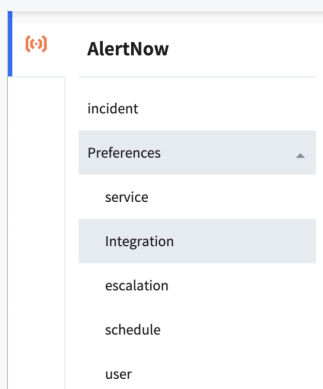
<https://wh.jandi.com/connect-api/webhook/31257583/d3...>
Copy

- Return to the **Add a third-party plug-in** window in the WhaTap Monitoring service, and enter values for **Channel Name** and **Webhook URL** at the bottom of the **Jandi** tab.
- Click the [**Registration**] button.

AlertNow



- Access the **AlertNow** service, then go to **Settings > Integration**.



- On the **Integration** screen, click the [Create Integration] button.
- From the list of integration types, select **WhaTap**.
- On the **Create Integration** screen, configure the **Name** and **Naming Rules**.

Integration

Entire

Integration name *

Enter integration name

Integration type *

WhaTap

Set naming convention *

Define and manage incident naming rules.

naming convention

[[level]][title] {projectName} {oname} {oid}

Preview rule settings

Incident names are limited to 3500 bytes in length.

[Warning]CPU Used > 10 % OTE-SERVER ote-front-330494549

sample data

```
{
  "metricName": "cpu",
  "pcode": 188,
  "level": "Warning",
  "metricValue": "26.888296",
  "oid": "-330494549",
  "title": "CPU Used > 10 %",
  "message": "MEMORY is too high 98%",
  "uid": "4dfe0b7686-c4335f0ef4c1cfa45754",
  "metricThreshold": "18.0",
  "oname": "ote-front",
  "time": 1592887274971,
  "projectName": "OTE-SERVER",
  "status": "on"
}
```

56/1600 byte

5. Click the [Confirm] button to create the integration.
6. Check the list of created integrations and select the newly created integration.

Integration

Create integration

Enter search term

Last update: 2024-02-21 11:31:17

Integration name	Integration type	Integration key	service name
whatap-test	WhaTap		whatap-test

7. On the details screen, click the copy icon on the right side of the URL to copy it.

Integration

whatap-test

Basic Information

Integration type

WhaTap

Integration key

fda527a5140

URL

https://alertnowitgr.opsnow.com/integration/whatap/v1/

constructor

Lisa

Copy to clipboard

8. Return to the **Add a third-party plug-in** window in the WhaTap Monitoring service, and enter values for **Channel Name** and **Webhook URL** at the bottom of the **AlertNow** tab.
9. Click the [**Registration**] button.

Webhook



1. Register the Webhook that will receive notifications. The alert `title` and `text` are sent to the registered URL via the **POST** method.

```
{
  "title": "Alert Title",
  "text": "Sample warning message"
}
```

2. Enter values for **Webhook Name** and **Webhook URL**.
3. Click the [**Registration**] button.

Webhook JSON



1. Register the Webhook that will receive notifications. Alert data is sent to the registered URL in **JSON format** via the **POST** method. For substitution variables that can be included in the JSON payload, see **Available Variables**.
 - `${something}` represents a substitution value provided by WhaTap. Refer to **Available Variables** for the full list and descriptions.
 - The event status (`status`) is categorized as **event triggered** (`on`) or **event recovered** (`off`).
 - The `level` is categorized as `None` , `Info` , `Warning` , or `Critical` .

```
{
  "pcode": "${pcode}",
  "projectName": "${projectName}",
  "time": "${time}",
  "oid": "${oid}",
  "oname": "${oname}",
  "title": "${title}",
  "message": "${message}",
  "level": "${level}",
  "status": "${status}",
}
```



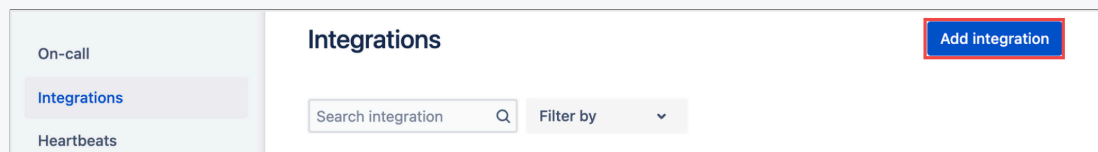
```
"metricName": "${metricName}",
"metricValue": "${metricValue}",
"metricThreshold": "${metricThreshold}",
"uuid": "${uuid}",
"okind": "${okind}",
"onode": "${onode}"
}
```

2. Enter values for the **Webhook Name** and **Webhook URL** fields.
3. To add a header value, click + **Add** in **Add user header value**, then enter the key and value.
4. Click the [**Registration**] button.

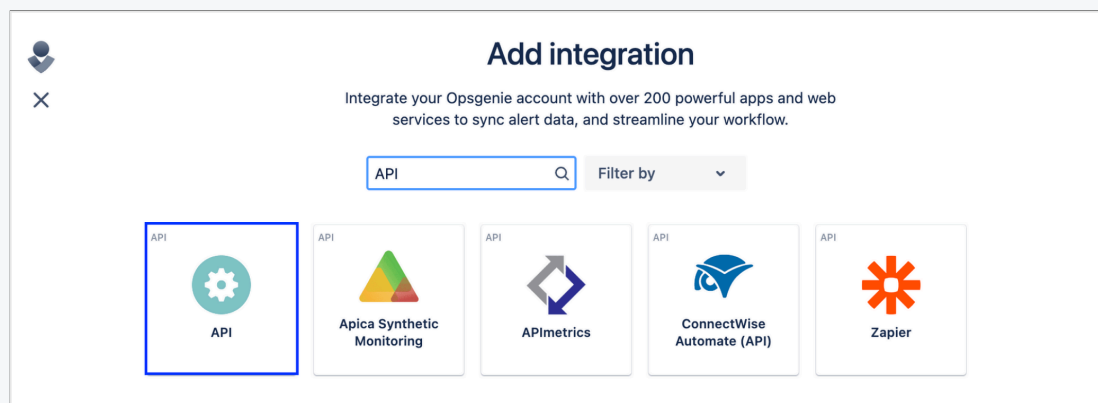
Opsgenie



1. In the **Teams** menu, select the team that will receive notifications.
2. Go to the **Integration** menu and click the [Add Integration] button.



3. In the integration list, search for and select **API**.





4. Set the API name, assign a team, configure the required permissions, and then click the [Save Integration] button.

5. Return to the **Add a third-party plug-in** window in the WhaTap monitoring service, and enter values for **Name** and **API Key** at the bottom of the **Opsgenie** tab.
6. Click the [**Registration**] button.

PagerDuty



1. Sign up for a [PagerDuty](#) account and log in.
2. In PagerDuty, go to **Services > Service Directory**, then click the [+ New Service] button.

PagerDuty

3. Enter the service information (**Name**) and click the [Next] button.

PagerDuty

4. Configure the **Escalation Policy**, then click the [Next] button.



PagerDuty

5. Configure **Alert Grouping** and **Transient Alerts**, then click the [Next] button.

PagerDuty

6. In **Integrations**, select **Event API V2**, then click the [Create Service] button.

PagerDuty

7. On the created service page, copy the **Integration Name** and **Integration Key** values.

PagerDuty

8. Return to the **Add a third-party plug-in** screen in the WhaTap monitoring service and enter the **Integration Name** and **Integration Key** values.
9. Click the [Register] button.

ⓘ If you do not receive messages or the plugin is not registered, check whether the **Integration Name** or **Integration Key** values are correct.

LINE



1. Install the LINE app.
2. Scan the QR code below to add the official **WhaTapLabs** channel as a friend.

Add LINE 3rd-party plugin

3. Use the following commands to link your project with LINE and configure notifications.

Commands

```
/help # Show available commands
/list # View the list of linked projects
/license [ACCESS_KEY] # Add a new project linkage
/remove [PROJECT_CODE] # Remove a linked project
/language [ko|ja|en] # Set notification language (Korean/Japanese/English)
```



- You can find the project access key under **Management > Project Management**.
- Using the `/language` command in LINE applies **only to the current LINE chat room**. It does not change the notification language for other 3rd-party plugins or projects.

Examples

```
/license XXXXXXXXXX # Link a project
/language ko # Set language to Korean
/list # Check linked projects
```



- **Project access key:** Available under **Management > Project Management** in the WhaTap project.
- **Language settings:** The `/language` command applies only within the LINE chat room and does not affect other notification channels.

ilert



1. In the ilert console, click [+ Create alert source] at the top right of the **Alert sources** page.
2. In the **Integration type** step, enter `WhaTap` in the **Filter integrations** search field at the bottom to search.
3. From the search results, select the **WhaTap** tile and click the [Next] button.
4. Enter the alert source name and assign a team, then click the [Next] button.
 - You can create a new escalation policy or select an existing one.
5. Select the alert grouping method and click the [Continue setup] button.
 - The initially selected **Do not group alerts** option can be changed later.
6. On the next screen, you can configure additional settings such as **alert templates and priorities**. For now, click the [Finish setup] button to complete the setup.
7. When the setup is complete, an **API key** or **Webhook URL** will be generated on the completion screen.



8. Copy the generated **Webhook URL** from the **Webhook JSON** tab and paste it into the **Webhook URL** input field below.
9. Click the [Register] button.

Preventing Alert Storms

If a large number of alerts occur within a short period, alert notifications are temporarily paused for a specified duration. For example, if 20 events occur within 5 minutes, alert notifications will be suspended for the next 5 minutes. Once the configured **Pause Duration** has elapsed, the alert-storm prevention feature is automatically released.

1. Go to **Alert > Notifications > the Avoids a large number of alerts** section.
2. Click the toggle button to **Enable** or **Disable** the feature.
 -  **Enabled:** Temporarily pause alerts when an alert storm is detected
 -  **Disabled:** Resume alert notifications
3. Configure the **Detection Time**, **Detection Count**, and **Pause Duration**.
 - If the number of events exceeds the **Detection Count** within the **Detection Time**, alert notifications are paused for the configured **Pause Duration**.



When alerts are blocked due to an alert storm, a notification banner is displayed. To immediately release the alert-storm block, click the **Stop Alert Storm Prevention** button in the banner.

Event History And Monitoring

Home > Select Project >  **Alert** > **Event History**

Event History allows you to view events that have occurred in the project and analyze their detailed information. It supports Elasticsearch DSL queries, enabling you to find events that match specific conditions quickly and accurately.

Viewing Event History

Click the **event title** in the search results to move to a screen where you can analyze detailed data for the time period when the event occurred. This allows you to comprehensively understand the situation before and after the event.

1. Go to **Alert > Event History**.
2. In the **Event History** screen, use **Time Selection** to set the period for viewing event records.
3. Enter a query in the filter input field and click the [Magnifying Glass] button.
 - If no query is entered, all events are displayed.

Column Settings

You can choose which columns to display or hide.

1. In the **Event History** screen, click the [ Column Settings] button at the top left.
2. In the **Column Settings** window, select the columns you want to show or hide.
 - You can change the column order in **Displayed Items**.
 - You can search for specific columns using the **Column Name Search** field.

Column Item	Description
Event Name	The title of the event

Column Item	Description
Message	The event message or snapshot information
Event Status	The current status of the event
Agent Name	The name (Oname) of the agent where the event occurred - Displayed as blank if the event is not associated with a specific agent
Agent Group	The agent type (OkindName) where the event occurred - Displayed as blank if the event is not associated with a specific agent
Agent Node	The server (OnodeName) of the agent where the event occurred - Displayed as blank if the event is not associated with a specific agent
Event Occurred Time	The time when the event occurred
Event Resolved Time	The time when the event was resolved - Displayed as - for events without a resolved state

- After completing the settings, click the [Confirm] button to apply the column configuration.

↓ CSV Download

You can download the query results as a CSV file.

- Go to **Alert > Event History** and click the [CSV] button.
- The list of events that match the current filter conditions is saved as a CSV file.

View Only Events in Progress

You can view only events that are currently in progress.

- Go to **Alert > Event History** and click the [View Only Events in Progress] button.
- The following condition is automatically added to the filter query and applied to the search.

Stateful: **true** and Status : **"ON"**

Event Search

- **Precise Search:** Specify exact conditions to retrieve only the events you need
- **Composite Conditions:** Search across multiple fields at once to handle complex conditions in a single query
- **Flexible Patterns:** Supports various search methods such as wildcards and partial matches
- **High Performance:** Fast searches powered by indexed data

Searchable Fields

Field names and values are not case sensitive.

- **Field Name Tag:** Event types that support the field
- **Field Type:** The data type of the field
 - #: Number
 - T: String
 - : Boolean
 - : Date

Event Information Fields

Field Name	Type	Description	Example Value
Title	String	Event title	"Database Connection Error"
OffTitle	String	Event recovery title	"RECOVERED: Database Connection Error"
Message	String	Event message	"Connection timeout occurred"
OffMessage	String	Event recovery message	"RECOVERED: Connection timeout occurred"

Field Name	Type	Description	Example Value
Level	String	Current event level	Critical, Warning, Info
OriginLevel	String	Original event level	Critical, Warning, Info
Status	String	Event status	ON, OFF, CANCEL, ACKNOWLEDGE, MAINTENANCE, DISABLED
ActCount	Number	Number of occurrences	2
MetricName	String	Metric name	"memory"
MetricValue	String	Metric value	"85.5"
OffValue	String	Recovery value	"72.8"
MetricThreshold	String	Threshold value	"80"
alertType	String	Event type	"BASIC", "METRICS", "TRANSACTION", etc.
alertId	String	Unique identifier of the event rule	"zf3uojer0fv4v7"

Type-specific Supported Fields

ⓘ These fields are supported only for specific event types. If the event type does not match, the value may be empty.

Field Name	Type	Event Type	Description	Example Value
eventRule	String	Basic, Metrics	Event trigger rule	"memory ≥ 80"
field	String	Real-	Log search	"content"

Field Name	Type	Event Type	Description	Example Value
		time Log	key	
keyword	String	Real-time Log	Log search value	"Error"
logCategory	String	Real-time Log	Log category	"AppLog"
logContent	String	Real-time Log	Log content	"00:00000:00009:2025/04/06 23:03:52.55 server Error: 1601, Severity: 17, State: 3\n00:00000:00009:2025/04/06 23:03:52.55 server There are not enough 'user connections' available to start a new process."

Target Fields

Field Name	Type	Description	Example Value
Oid	Number	Agent unique identifier	-98765432
Oname	String	Agent name	"web-server-01"
Okind	Number	Agent type unique identifier	867318026
OkindName	String	Agent type name	"web-server"
Onode	Number	Agent node unique identifier	334634079
OnodeName	String	Agent node name	"production-node-1"

Unique Event Fields

Field Name	Type	Description	Example Value
Id	Number	Unique identifier of the event	5768121
UUID	String	Unique event identifier	"550e8400-e29b-41d4-a716"

State / Flag Fields

Field Name	Type	Description	Example Value
Stateful	Boolean	Whether the event is state based	true, false
Disabled	Boolean	Whether the event is disabled	true, false
Escalation	Boolean	Whether escalation is applied	true, false
SystemEvent	Boolean	Whether it is a system event	true, false

Query Syntax by Search Type

! For the basic search syntax, refer to the [WhaTap log search query syntax](#) documentation.

1. Keyword Search

Events that contain the string "Database Connection"

```
title: "Database Connection"
```

2. Multiple Value Search

Retrieves events that include any one of the specified values.

```
title: Database Connection
```

Events whose current event level is Info or Warning

```
level: info warning
```

3. Pattern Search

You can search using patterns with wildcards.

Find events that occurred on specific agents using a name pattern

```
oname: web-*-prod
```

タイトルが Connection で終わるイベントを検索

```
title: *Connection
```

4. Complex Condition Search

You can combine multiple conditions to perform precise searches.

Events with a Critical level and a title containing Database

```
level: critical and title: Database
```

Warning or Critical events from a specific agent

```
oname: web-server-01 and level: Warning Critical
```

5. OR Condition Search

Retrieves events that satisfy **at least one** of the specified conditions.

Events where the title or message contains a specific keyword

```
title: Connection or message: Connection
```

6. Exclusion Conditions

You can exclude specific conditions from the search.

All events except those with an Info level

```
not level: info
```