**Practical Exercise: CLI Tool and Micro Integrator Dashboard**

Training Objective

Once you have created your integration artifacts and deployed them in the Micro Integrator, you can use the command line tool and the monitoring dashboard to monitor the artifacts.

High Level Steps

* Create the integration artifacts (using WSO2 Integration Studio). Try the lab on

**Configuring a REST API**.

* Start the CLI Tool/Dashboard.
* View the required artifact details.

Detailed Instructions

* Using the CLI:

Managing Integrations with apictl

WSO2 API Controller, **apictl** allows you to monitor the Synapse artifacts (deployed in a specified Micro Integrator server) and perform various management and administration tasks from the command line.

**Info**

**Before you begin**

* Ensure that WSO2 Micro Integrator is started. See the instructions on [installing the Micro Integrator](https://apim.docs.wso2.com/en/4.1.0/install-and-setup/install/installing-the-product/installing-mi).
* Make sure the apictl is downloaded and initialized, if not follow the steps in [Download and Initialize the apictl](https://apim.docs.wso2.com/en/4.1.0/install-and-setup/setup/api-controller/getting-started-with-wso2-api-controller/#download-and-initialize-the-apictl).
* Ensure that the Micro Integrator management endpoint is added to the environment configurations of CTL, before you start working with the following CTL commands. For more information, see [Add an Environment](https://apim.docs.wso2.com/en/4.1.0/install-and-setup/setup/api-controller/getting-started-with-wso2-api-controller/#add-an-environment).

Login to a Micro Integrator

After adding an environment, you can login to the Micro Integrator instance of that environment using credentials.

1. Run any of the following CTL commands to login to a Micro Integrator.
   * **Command**

apictl mi login <environment-name> -k

apictl mi login <environment-name> -u <username> -k

apictl mi login <environment-name> -u <username> -p <password> -k

**Tip**

If you run apictl mi login <environment-name> you are prompted to provide both the username and the password. If you run apictl mi login <environment-name> --username <username>, you are prompted to provide the password.

**Info**

**Flags:**

* + - Optional :  
      --username or -u : Username for login  
      --password or -p : Password for login  
      --password-stdin : Get password from stdin

**Example**

apictl mi login dev -k

apictl mi login dev -u admin -p admin -k

apictl mi login dev --username admin --password admin -k

* + **Response**

Response Format

Logged into MI **in** '<environment-name>' environment

Example Response

**Warning**

Using --password in CTL is not secure. You can use --password-stdin instead. For example,

cat ~/.mypassword | ./apictl mi login dev --username admin --password-stdin -k

Logout from a Micro Integrator

1. Run the following command to logout from the current session of the Micro Integrator.
   * **Command**

apictl mi logout <environment-name>

**Example**

apictl mi logout dev

* + **Response**

Response Format

Logged out from MI **in** '<environment-name>' environment

Example Response

Manage Users

You can view details of users stored in the [external user store](https://apim.docs.wso2.com/en/4.1.0/install-and-setup/setup/mi-setup/user_stores/managing_users). If you are logged in to the apictl with administrator credentials, you can also add new users, and remove users from the user store.

Get information about users

1. List users of the Micro Integrator.
   * **Command**

apictl mi get users -e <environment>

**Info**

**Flags:**

* + - Required :  
      --environment or -e : Environment of the Micro Integrator to be searched
    - Optional :  
      --format : pretty-print using templates  
      --pattern or -p : Filter users by regex  
      --role or -r : Filter users by role

**Example**

apictl mi get users -e dev

apictl mi get users -r admin -e dev

apictl mi get users -p \*tester\* -e dev

* + **Response**

USER ID

admin

capp-tester

1. Get information on a specific user.
   * **Command**

apictl mi get users [user-name] -d [domain] -e <environment>

**Info**

**Flags:**

* + - Required :  
      --environment or -e : Environment of the Micro Integrator to be searched
    - Optional :  
      --domain or -d : Domain name of the secondary user store to be searched  
      --format : pretty-print using templates

**Example**

apictl mi get users capp-tester -d testing.com -e dev

* + **Response**

Name - TESTING.COM/capp-tester

Is Admin - false

Roles - TESTING.COM/tester, Application/applicationRole1

Add a new user

You can use the command below to add a new user to a Micro Integrator.

* **Command**

apictl mi add user [user-name] -e <environment>

**Info**

**Flags:**

* + Required :  
    --environment or -e : Environment of the Micro Integrator to be searched

**Example**

apictl mi add user capp-tester -e dev

* **Response**

Adding new user [ capp-tester ] status: Added

**Note**

To add a new user to a secondary user store, provide the corresponding user store domain when prompted.

Delete a user

You can use the command below to remove a user from the Micro Integrator.

* **Command**

apictl mi delete user [user-name] -e <environment>

**Info**

**Flags:**

* + Required :  
    --environment or -e : Environment of the Micro Integrator
  + Optional :  
    --domain or -d : The domain of the secondary user store from which the user should be deleted

**Example**

apictl mi delete user capp-tester -d testing.com -e dev

* **Response**

Deleting user [ capp-tester ] status: Deleted

Manage Roles

The Micro Integrator has limited role support without fine-grained permission tree support as in the Enterprise Integrator.

In Micro Integrator, we have one admin role and all the other roles from primary and secondary user stores are considered non-admin roles.

Get information about roles

1. List roles of the Micro Integrator.
   * **Command**

apictl mi get roles -e <environment>

**Info**

**Flags:**

* + - Required :  
      --environment or -e : Environment of the Micro Integrator to be searched

**Example**

apictl mi get roles -e dev

* + **Response**

ROLE

admin

primaryRole1

Application/applicationRole1

Internal/everyone

Internal/internalRole1

TEST.COM/testRole1

1. Get information on a specific role.
   * **Command**

apictl mi get roles [role-name] -e <environment>

**Info**

**Flags:**

* + - Required :  
      --environment or -e : Environment of the Micro Integrator to be searched
    - Optional :  
      --domain or -d : Domain of the secondary user store to be searched

**Example**

apictl mi get roles tester -d testing.com -e dev

* + **Response**

Role Name - TESTING.COM/tester

Users - TESTING.COM/capp-tester

**Note**

To get hybrid roles (application/internal) specify the role type as the domain.

apictl mi get roles tester -d application -e dev

Add a new role

* **Command**

apictl mi add role [role-name] -e <environment>

**Info**

**Flags:**

* + Required :  
    --environment or -e : Environment of the Micro Integrator

**Example**

apictl mi add role tester -e dev

* + **Response**

Adding new role [ tester ] status: Added

**Note**

To add a new role to a secondary user store, provide the corresponding user store domain when prompted.

**Note**

To add hybrid roles (application/internal) specify the type in the role name.

apictl mi add role internal/InternalRole -e dev

Delete a role

* **Command**

apictl mi delete role [role-name] -e <environment>

**Info**

**Flags:**

* + Required :  
    --environment or -e : Environment of the Micro Integrator
  + Optional :  
    --domain or -d : The domain of the secondary user store from which the role should be deleted

**Example**

apictl mi delete role tester -d testing.com -e dev

* + **Response**

Deleting new role [ tester ] status: Deleted

**Note**

To delete hybrid roles (application/internal) specify the role type as domain.

apictl mi delete role InternalRole -d internal -e dev

Assign/revoke roles to/from users

* **Command**

apictl mi update user [user-name] -e <environment>

**Info**

**Flags:**

- Required :

`--environment` or `-e` : Environment **of** the Micro Integrator

**Example**

apictl mi update user capp-tester -e dev

* **Response**

Added/removed the roles

**Note**

Use a space-separated list of role names when entering the added/removed roles

Monitor Integration Artifacts

Follow the instructions below to display a list of artifacts or get information about a specific artifact in an environment using CTL:

Composite Applications (CApps)

1. List composite applications (CApps) in an environment.
   * **Command**

apictl mi get composite-apps -e <environment>

**Info**

**Flags:**

* + - Required :  
      --environment or -e : Environment of the Micro Integrator to be searched
    - Optional :  
      --format : pretty-print using templates

**Example**

apictl mi get composite-apps -e dev

* + **Response**

NAME VERSION

HealthCareCompositeExporter 1.0.0

FoodServiceCApp 2.0.0

1. Get information on a specific composite application in an environment.
   * **Command**

apictl mi get composite-apps [capp-name] -e <environment>

**Info**

**Flags:**

* + - Required :  
      --environment or -e : Environment of the Micro Integrator to be searched
    - Optional :  
      --format : pretty-print using templates

**Example**

apictl mi get composite-apps HealthCareCompositeExporter -e dev

* + **Response**

Name - HealthCareCompositeExporter

Version - 1.0.0

Artifacts :

NAME TYPE

sample-local-entry local-entry

email-connector lib

in-memory-message-store message-store

GrandOakEndpoint endpoint

sample\_seq\_template template

scheduled-msg-processor message-processors

sample\_template template

HealthcareAPI api

sample-sequence sequence

PineValleyEndpoint endpoint

StockQuoteProxy proxy-service

sample-cron-task task

httpInboundEP inbound-endpoint

Integration APIs

1. List integration APIs in an environment.
   * **Command**

apictl mi get apis -e <environment>

**Info**

**Flags:**

* + - Required :  
      --environment or -e : Environment of the Micro Integrator to be searched
    - Optional :  
      --format : pretty-print using templates

**Example**

apictl mi get apis -e dev

* + **Response**

NAME URL

HealthcareAPI http://localhost:8290/healthcare

FoodService http://localhost:8480/foodservice

1. Get information on a specific integration API in an environment.
   * **Command**

apictl mi get apis [api-name] -e <environment>

**Info**

**Flags:**

* + - Required :  
      --environment or -e : Environment of the Micro Integrator to be searched
    - Optional :  
      --format : pretty-print using templates

**Example**

apictl mi get apis HealthcareAPI -e dev

* + **Response**

Name - HealthcareAPI

Version - N/A

Url - http://localhost:8290/healthcare

Stats - disabled

Tracing - disabled

Resources :

URL METHOD

/doctor/{doctorType} [GET]

/report [GET]

Connectors

1. List connectors in an environment.
   * **Command**

apictl mi get connectors -e <environment>

**Info**

**Flags:**

* + - Required :  
      --environment or -e : Environment of the Micro Integrator to be searched
    - Optional :  
      --format : pretty-print using templates

**Example**

apictl mi get connectors -e dev

* + **Response**

NAME STATS PACKAGE DESCRIPTION

email enabled org.wso2.carbon.connector WSO2 email connector library

Data Services

1. List data services in an environment.
   * **Command**

apictl mi get data-services -e <environment>

**Info**

**Flags:**

* + - Required :  
      --environment or -e : Environment of the Micro Integrator to be searched
    - Optional :  
      --format : pretty-print using templates

**Example**

apictl mi get data-services -e dev

* + **Response**

NAME WSDL 1.1 WSDL 2.0

RESTDataService http://localhost:8290/services/RESTDataService?wsdl http://localhost:8290/services/RESTDataService?wsdl2

1. Get information on a specific data service in an environment.
   * **Command**

apictl mi get data-services [data-service-name] -e <environment>

**Info**

**Flags:**

* + - Required :  
      --environment or -e : Environment of the Micro Integrator to be searched
    - Optional :  
      --format : pretty-print using templates

**Example**

apictl mi get data-services RESTDataService -e dev

* + **Response**

Name - RESTDataService

Group Name - RESTDataService

Description - Exposing the data service as a REST service.

WSDL 1.1 - http://localhost:8290/services/RESTDataService?wsdl

WSDL 2.0 - http://localhost:8290/services/RESTDataService?wsdl2

Queries :

ID NAMESPACE

ReadStudents http://ws.wso2.org/dataservice/ReadStudents

DeleteStudent http://ws.wso2.org/dataservice

Endpoints

1. List endpoints in an environment.
   * **Command**

apictl mi get endpoints -e <environment>

**Info**

**Flags:**

* + - Required :  
      --environment or -e : Environment of the Micro Integrator to be searched
    - Optional :  
      --format : pretty-print using templates

**Example**

apictl mi get endpoints -e dev

* + **Response**

NAME TYPE ACTIVE

GrandOakEndpoint http true

PineValleyEndpoint http true

1. Get information on a specific endpoint in an environment.
   * **Command**

apictl mi get endpoints [endpoint-name] -e <environment>

**Info**

**Flags:**

* + - Required :  
      --environment or -e : Environment of the Micro Integrator to be searched
    - Optional :  
      --format : pretty-print using templates

**Example**

apictl mi get endpoints GrandOakEndpoint -e dev

* + **Response**

Name - GrandOakEndpoint

Type - HTTP Endpoint

Active - true

Method - GET

URI Template - http://localhost:9091/grand/doctors

Inbound Endpoints

1. List inbound endpoints in an environment.
   * **Command**

apictl mi get inbound-endpoints -e <environment>

**Info**

**Flags:**

* + - Required :  
      --environment or -e : Environment of the Micro Integrator to be searched
    - Optional :  
      --format : pretty-print using templates

**Example**

apictl mi get inbound-endpoints -e dev

* + **Response**

NAME TYPE

httpInboundEP http

1. Get information on a specific inbound endpoint in an environment.
   * **Command**

apictl mi get inbound-endpoints [inbound-name] -e <environment>

**Info**

**Flags:**

* + - Required :  
      --environment or -e : Environment of the Micro Integrator to be searched
    - Optional :  
      --format : pretty-print using templates

**Example**

apictl mi get inbound-endpoints httpInboundEP -e dev

* + **Response**

Name - httpInboundEP

Type - http

Stats - enabled

Tracing - enabled

Parameters :

NAME VALUE

inbound.http.port 8697

inbound.worker.pool.size.core 400

inbound.worker.pool.size.max 500

inbound.worker.thread.keep.alive.sec 60

inbound.worker.pool.queue.length -1

inbound.thread.id PassThroughInboundWorkerPool

Local Entries

1. List local entries in an environment.
   * **Command**

apictl mi get local-entries -e <environment>

**Info**

**Flags:**

* + - Required :  
      --environment or -e : Environment of the Micro Integrator to be searched
    - Optional :  
      --format : pretty-print using templates

**Example**

apictl mi get local-entries -e dev

* + **Response**

NAME TYPE

sample-local-entry Inline Text

1. Get information on a specific local entry in an environment.
   * **Command**

apictl mi get local-entries [local-entry-name] -e <environment>

**Info**

**Flags:**

* + - Required :  
      --environment or -e : Environment of the Micro Integrator to be searched
    - Optional :  
      --format : pretty-print using templates

**Example**

apictl mi get local-entries sample-local-entry -e dev

* + **Response**

Name - sample-local-entry

Type - Inline Text

Value - 0, 1

Message Processors

1. List message processors in an environment.
   * **Command**

apictl mi get message-processors -e <environment>

**Info**

**Flags:**

* + - Required :  
      --environment or -e : Environment of the Micro Integrator to be searched
    - Optional :  
      --format : pretty-print using templates

**Example**

apictl mi get message-processors -e dev

* + **Response**

NAME TYPE STATUS

scheduled-msg-processor Scheduled-message-forwarding-processor active

1. Get information on a specific message processor in an environment.
   * **Command**

apictl mi get message-processors [message-processor-name] -e <environment>

**Info**

**Flags:**

* + - Required :  
      --environment or -e : Environment of the Micro Integrator to be searched
    - Optional :  
      --format : pretty-print using templates

**Example**

apictl mi get message-processors scheduled-msg-processor -e dev

* + **Response**

Name - scheduled-msg-processor

Type - Scheduled-message-forwarding-processor

File Name - scheduled-msg-processor-1.0.0.xml

Message Store - in-memory-message-store

Artifact Container - [ Deployed From Artifact Container: HealthCareCompositeExporter ]

Status - active

Parameters :

client.retry.interval = 1000

interval = 1000

is.active = true

max.delivery.attempts = 4

max.delivery.drop = Disabled

max.store.connection.attempts = -1

member.count = 1

store.connection.retry.interval = 1000

target.endpoint = PineValleyEndpoint

throttle = false

Message Stores

1. List message stores in an environment.
   * **Command**

apictl mi get message-stores -e <environment>

**Info**

**Flags:**

* + - Required :  
      --environment or -e : Environment of the Micro Integrator to be searched
    - Optional :  
      --format : pretty-print using templates

**Example**

apictl mi get message-stores -e dev

* + **Response**

NAME TYPE SIZE

in-memory-message-store in-memory-message-store 0

1. Get information on a specific message store in an environment.
   * **Command**

apictl mi get message-stores [message-store-name] -e <environment>

**Info**

**Flags:**

* + - Required :  
      --environment or -e : Environment of the Micro Integrator to be searched
    - Optional :  
      --format : pretty-print using templates

**Example**

apictl mi get message-stores **in**-memory-message-store -e dev

* + **Response**

Name - in-memory-message-store

File Name - in-memory-message-store-1.0.0.xml

Container - [ Deployed From Artifact Container: HealthCareCompositeExporter ]

Producer - org.apache.synapse.message.store.impl.memory.InMemoryProducer@3d288f9e

Consumer - org.apache.synapse.message.store.impl.memory.InMemoryConsumer@5e6443d6

Size - 0

Properties :

No Properties found

Proxy Services

1. List proxy services in an environment.
   * **Command**

apictl mi get proxy-services -e <environment>

**Info**

**Flags:**

* + - Required :  
      --environment or -e : Environment of the Micro Integrator to be searched
    - Optional :  
      --format : pretty-print using templates

**Example**

apictl mi get proxy-services -e dev

* + **Response**

NAME WSDL 1.1 WSDL 2.0

StockQuoteProxy http://localhost:8290/services/StockQuoteProxy?wsdl http://localhost:8290/services/StockQuoteProxy?wsdl2

1. Get information on a specific proxy service in an environment.
   * **Command**

apictl mi get proxy-services [proxy-name] -e <environment>

**Info**

**Flags:**

* + - Required :  
      --environment or -e : Environment of the Micro Integrator to be searched
    - Optional :  
      --format : pretty-print using templates

**Example**

apictl mi get proxy-services StockQuoteProxy -e dev

* + **Response**

Name - StockQuoteProxy

WSDL 1.1 - http://localhost:8290/services/StockQuoteProxy?wsdl

WSDL 2.0 - http://localhost:8290/services/StockQuoteProxy?wsdl2

Stats - disabled

Tracing - disabled

Sequences

1. List sequences in an environment.
   * **Command**

apictl mi get sequences -e <environment>

**Info**

**Flags:**

* + - Required :  
      --environment or -e : Environment of the Micro Integrator to be searched
    - Optional :  
      --format : pretty-print using templates

**Example**

apictl mi get sequences -e dev

* + **Response**

NAME STATS TRACING

fault disabled disabled

main disabled disabled

sample-sequence disabled disabled

1. Get information on a specific sequence in an environment.
   * **Command**

apictl mi get sequences [sequence-name] -e <environment>

**Info**

**Flags:**

* + - Required :  
      --environment or -e : Environment of the Micro Integrator to be searched
    - Optional :  
      --format : pretty-print using templates

**Example**

apictl mi get sequences sample-sequence -e dev

* + **Response**

Name - sample-sequence

Container - [ Deployed From Artifact Container: HealthCareCompositeExporter ]

Stats - disabled

Tracing - disabled

Mediators - LogMediator, STRING

Scheduled Tasks

1. List scheduled tasks in an environment.
   * **Command**

apictl mi get tasks -e <environment>

**Info**

**Flags:**

* + - Required :  
      --environment or -e : Environment of the Micro Integrator to be searched
    - Optional :  
      --format : pretty-print using templates

**Example**

apictl mi get tasks -e dev

* + **Response**

NAME

sample-cron-task

CheckPriceTask

1. Get information on a specific scheduled task in an environment.
   * **Command**

apictl mi get tasks [task-name] -e <environment>

**Info**

**Flags:**

* + - Required :  
      --environment or -e : Environment of the Micro Integrator to be searched
    - Optional :  
      --format : pretty-print using templates

**Example**

apictl mi get tasks sample-cron-task -e dev

* + **Response**

Name - sample-cron-task

Trigger Type - cron

Cron Expression - 0 30 1 \* \* ?

Templates

1. List all templates in an environment.
   * **Command**

apictl mi get templates -e <environment>

**Info**

**Flags:**

* + - Required :  
      --environment or -e : Environment of the Micro Integrator to be searched
    - Optional :  
      --format : pretty-print using templates

**Example**

apictl mi get templates -e dev

* + **Response**

NAME TYPE

sample\_seq\_template Sequence

sample\_template Endpoint

1. List a specific type of template in an environment.
   * **Command**

apictl mi get templates [template-type] -e <environment>

**Info**

**Flags:**

* + - Required :  
      --environment or -e : Environment of the Micro Integrator to be searched
    - Optional :  
      --format : pretty-print using templates

**Example**

apictl mi get templates endpoint -e dev

apictl mi get templates sequence -e dev

* + **Response**

NAME

sample\_seq\_template

1. Get information on a specific template in an environment.
   * **Command**

apictl mi get templates [template-type] [template-name] -e <environment>

**Info**

**Flags:**

* + - Required :  
      --environment or -e : Environment of the Micro Integrator to be searched
    - Optional :  
      --format : pretty-print using templates

**Example**

apictl mi get templates endpoint sample\_template -e dev

* + **Response**

Name - sample\_template

Parameters : name, uri

Change status of an Artifact

You can use the commands below to activate or deactivate endpoints, message processors or proxy services deployed in a Micro Integrator.

Endpoint

1. Activate an endpoint deployed in a Micro Integrator.
   * **Command**

apictl mi activate endpoint [endpoint-name] -e <environment>

**Info**

**Flags:**

* + - Required :  
      --environment or -e : Environment of the Micro Integrator to be searched

**Example**

apictl mi activate endpoint GrandOakEndpoint -e dev

* + **Response**

GrandOakEndpoint is switched On

1. Deactivate an endpoint deployed in a Micro Integrator.
   * **Command**

apictl mi deactivate endpoint [endpoint-name] -e <environment>

**Info**

**Flags:**

* + - Required :  
      --environment or -e : Environment of the Micro Integrator to be searched

**Example**

apictl mi deactivate endpoint GrandOakEndpoint -e dev

* + **Response**

GrandOakEndpoint is switched Off

Message Processor

1. Activate a message processor deployed in a Micro Integrator.
   * **Command**

apictl mi activate message-processor [message-processor-name] -e <environment>

**Info**

**Flags:**

* + - Required :  
      --environment or -e : Environment of the Micro Integrator to be searched

**Example**

apictl mi activate message-processor scheduled-msg-processor -e dev

* + **Response**

scheduled-msg-processor : is activated

1. Deactivate a message processor deployed in a Micro Integrator.
   * **Command**

apictl mi deactivate message-processor [message-processor-name] -e <environment>

**Info**

**Flags:**

* + - Required :  
      --environment or -e : Environment of the Micro Integrator to be searched

**Example**

apictl mi deactivate message-processor scheduled-msg-processor -e dev

* + **Response**

scheduled-msg-processor : is deactivated

Proxy Service

1. Activate a proxy service deployed in a Micro Integrator.
   * **Command**

apictl mi activate proxy-service [proxy-name] -e <environment>

**Info**

**Flags:**

* + - Required :  
      --environment or -e : Environment of the Micro Integrator to be searched

**Example**

apictl mi activate proxy-service StockQuoteProxy -e dev

* + **Response**

Proxy service StockQuoteProxy started successfully

1. Deactivate a proxy service deployed in a Micro Integrator.
   * **Command**

apictl mi deactivate proxy-service [proxy-name] -e <environment>

**Info**

**Flags:**

* + - Required :  
      --environment or -e : Environment of the Micro Integrator to be searched

**Example**

apictl mi deactivate proxy-service StockQuoteProxy -e dev

* + **Response**

Proxy service StockQuoteProxy stopped successfully

Manage Loggers used in Micro Integrator

Get information on a specific logger

* **Command**

apictl mi get log-levels [logger-name] -e <environment>

**Info**

**Flags:**

* + Required :  
    --environment or -e : Environment of the Micro Integrator to be searched
  + Optional :  
    --format : pretty-print using templates

**Example**

apictl mi get log-levels org-apache-coyote -e dev

* **Response**

NAME LOG LEVEL COMPONENT

org-apache-coyote WARN org.apache.coyote

Add a new logger

You can use the command below to add a new logger to a Micro Integrator.

* **Command**

apictl mi add log-level [logger-name] [class-name] [log-level] -e <environment>

**Info**

**Flags:**

* + Required :  
    --environment or -e : Environment of the Micro Integrator to be searched

**Example**

apictl mi add log-level synapse-api org.apache.synapse.rest.API DEBUG -e dev

* **Response**

Successfully added logger for ('synapse-api') with level DEBUG for class org.apache.synapse.rest.API

Update a logger

You can use the command below to update the log level of an existing logger.

* **Command**

apictl mi update log-level [logger-name] [log-level] -e <environment>

**Info**

**Flags:**

* + Required :  
    --environment or -e : Environment of the Micro Integrator to be searched

**Example**

apictl mi update log-level org-apache-coyote DEBUG -e dev

* **Response**

Successfully added logger for ('org-apache-coyote') with level DEBUG

Download log files

List available log files

* **Command**

apictl mi get logs -e <environment>

**Info**

**Flags:**

* + Required :  
    --environment or -e : Environment of the Micro Integrator to be searched
  + Optional :  
    --format : pretty-print using templates

**Example**

apictl mi get logs -e dev

* **Response**

NAME SIZE

wso2carbon.log 429.5 KB

correlation.log 0 B

wso2carbon-trace-messages.log 0 B

wso2-mi-api.log 11.9 KB

patches.log 15.7 KB

audit.log 0 B

wso2-mi-service.log 10.3 KB

http\_access\_.log 35.8 KB

wso2error.log 156.2 KB

Download a specific log file

* **Command**

apictl mi get logs [file-name] -p [download-location] -e <environment>

**Info**

**Flags:**

* + Required :  
    --environment or -e : Environment of the Micro Integrator to be searched
  + Optional :  
    --path or -p : Path the file should be downloaded (default is current executable directory)

**Example**

apictl mi get logs wso2carbon.log -p log-files -e dev

* **Response**

Log file downloaded to log-files/wso2carbon.log

Monitor transactions

Transaction Counts

You can use the command below to get information about the inbound transactions received by the Micro Integrator.

* **Command**

apictl mi get transaction-counts -e <environment>

apictl mi get transaction-counts [year] [month] -e <environment>

**Info**

**Flags:**

* + Required :  
    --environment or -e : Environment of the Micro Integrator to be searched
  + Optional :  
    --format : pretty-print using templates

**Example**

apictl mi get transaction-counts -e dev

apictl mi get transaction-counts 2021 01 -e dev

* **Response**
* YEAR MONTH TRANSACTION COUNT

2021 1 126

Transaction Reports

You can use the command below to generate the transaction count summary report based on the inbound transactions received by the Micro Integrator.

* **Command**

apictl mi get transaction-reports [start] [end] -e <environment>

**Info**

**Flags:**

* + Required :  
    --environment or -e : Environment of the Micro Integrator to be searched
  + Optional :  
    --path or -p : Path the file should be downloaded (default is current executable directory)

**Example**

apictl mi get transaction-reports 2020-05 2020-06 -e dev

apictl mi get transaction-reports 2020-05 -e dev -p reports/mi

* **Response**

Transaction Count Report created in reports/mi/transaction-count-summary-1610597725520763836.csv

Update HashiCorp AppRole Pull secret ID

You can use the command below to update the HashiCorp AppRole Pull secret ID that is used by the Micro Integrator to connect with HashiCorp.

**Note**

* The HashiCorp secret ID is only applicable when **AppRole Pull** authentication is used between the Micro Integrator and HashiCorp.
* This command only updates the SecretId for the current session of the Micro Integrator. To persist the Secret Id, you need to update the deployment.toml file and restart the Micro Integrator.

See [Using HashiCorp Secrets](https://apim.docs.wso2.com/en/4.1.0/install-and-setup/setup/mi-setup/security/using-hashicorp-secrets) for details.

* **Command**

apictl mi update hashicorp-secret [secret\_id] -e <environment>

**Info**

**Flags:**

* + Required :  
    --environment or -e : Environment of the Micro Integrator for which the HashiCorp secret ID should be updated.

**Example**

apictl mi update hashicorp-secret 47c39b09-c0a9-6ebf-196e-038eb7aad336 -e dev

* **Response**

SecretId value is updated in HashiCorp vault runtime configurations. To persist the new SecretId in the next ser

Using the dashboard:

Monitoring MI Artifacts and Logs

The Micro Integrator (MI) dashboard monitors the MI instances in a deployment. This can be a single MI instance or multiple MI instances in a group (cluster). It provides a graphical view of the integration artifacts that are deployed in the MI instances. You can also perform various management and administration tasks using the dashboard.

The dashboard communicates with the management APIs of each Micro Integrator instance in the group (cluster) to get and manipulate data.

Capabilities of the MI dashboard

You can use the dashboard to perform the following administration tasks related to your Micro Integrator deployment:

* **View the MI servers in the deployment**

View basic information of each server node.

* **View integration artifacts deployed in a group**

View details of the artifacts deployed in a cluster or group of Micro Integrator instances.

* **Identify the MI servers where a specified artifact is deployed**

View the MI server instances where each artifact is deployed.

* **Update deployed artifacts**

**Note**

When you update an artifact, only the specified MI instance will be updated. Cluster-wide updates are not available with the dashboard.

You can activate/deactivate the following artifacts from the dashboard: *Proxy Services*, *Endpoints*, and *Message Processors*.

You can enable/disable tracing for the following artifacts: *Proxy Services*, *Endpoints*, *APIs* *Sequences* and *Inbound Endpoints*.

* **View logs**

You can view the log files generated for each Micro Integrator instance of the cluster/group.

* **View, update, and add loggers**

This page can be accessed by users with admin rights only. You can view log configurations of each instance and update the log level. You can update the log levels on a single node or apply the change to the entire cluster/group as well. Furthermore, you can add new loggers, which will be applied to the entire cluster/group.

* **Manage users**

This page can be accessed by users with admin rights only. You can view details of users stored in the [external user store](https://apim.docs.wso2.com/en/4.1.0/install-and-setup/setup/mi-setup/user_stores/managing_users). You can also add new users to the specified cluster/group.

Using the MI Dashboard

Follow the steps given below to get started with the Micro Integrator Dashboard.

Step 1 - Download the MI Dashboard

Download the binary distribution of the product, and then follow the instructions to start the Micro Integrator and the dashboard.

* [Install the Micro Integrator](https://apim.docs.wso2.com/en/4.1.0/install-and-setup/install/installing-the-product/installing-mi).
* [Install the Micro Integrator Dashboard](https://apim.docs.wso2.com/en/4.1.0/install-and-setup/install/installing-the-product/installing-mi-dashboard).

Step 2 - Configure the MI servers

Follow the steps given below to configure the MI servers to publish data to the dashboard.

1. To connect the MI servers with the dashboard, add the following configuration to the deployment.toml file (stored in the <MI\_HOME>/conf/ folder) of each server instance.

[dashboard\_config]

dashboard\_url = "https://{hostname/ip}:{port}/dashboard/api/"

heartbeat\_interval = 5

group\_id = "mi\_dev"

node\_id = "dev\_node\_2"

If the Micro Integrator server is deployed in a Kubernetes environment, add the following configuration to the deployment.toml file.

Limitation: When there are replicas in the deployment, the write operations will not work properly.

If communicating via Ingress, use the following configuration:

dashboard\_url = "https://{hostname/ip}:{port}/dashboard/api/"

management\_hostname = "<INGRESS\_HOSTNAME>"

If communicating via Service instead, use the following configuration:

dashboard\_url = "https://{hostname/ip}:{port}/dashboard/api/"

management\_hostname = "<SERVICE\_NAME>"

management\_port = <SERVICE\_PORT>

|  |  |
| --- | --- |
| dashboard\_url | **Required**. This is the URL to access dashboard server. Replace the hostname/IP and port (default - 9743) with relevant values from your environment. |
| heartbeat\_interval | **Optional**. The time interval (in seconds) between two heartbeats sent from the Micro Integrator to the dashboard server. By default, the heartbeat\_interval is set to 5. |
| group\_id | **Optional**. In a clustered deployment, the group ID should be the same in all Micro Integrator Instances. The dashboard displays information from one group at a time. By default, the group\_id is set to default. |
| node\_id | **Optional**. By default, in a clustered deployment, the relevant node\_id is used as this configuration. For more information about the cluster node ID, see the instructions on [configuring an MI cluster](https://apim.docs.wso2.com/en/4.1.0/install-and-setup/setup/mi-setup/deployment/deploying_wso2_ei/#node-id). In a non-clustered deployment, a random UUID is used if the node\_id is not set for this configuration. |
| management\_hostname | **Required if MI server is deployed in a Kubernetes environment**. Hostname for the Micro Integrator management endpoint. |
| management\_port | **Optional**. Port of the Micro Integrator management endpoint. |

1. **Optionally**, configure the [Micro Integrator user store](https://apim.docs.wso2.com/en/4.1.0/install-and-setup/setup/mi-setup/user_stores/setting_up_a_userstore).

**Tip**

Note the following about your user store configurations.

* + The user credentials for signing in to the dashboard should be stored in your user store. This can be the default **file-based user store** or an **external LDAP/RDBMS** user store.
  + [User management](https://apim.docs.wso2.com/en/4.1.0/install-and-setup/setup/mi-setup/user_stores/managing_users) is possible only if you have an RDBMS or LDAP user store for your Micro Integrator.
  + If you have an [external RDBMS user store](https://apim.docs.wso2.com/en/4.1.0/install-and-setup/setup/mi-setup/user_stores/setting_up_a_userstore/#configuring-an-rdbms-user-store), be sure that the RDBMS driver is correctly added to the <MI\_HOME>/lib folder. You will not be able to sign in without the driver.

1. Regardless of the user who logs in, the dashboard uses the user configured in its deployment.toml to fetch the data to the dashboard server. Then the dashboard renders these data in the UI according to logged-in user. Hence, configure the super admin user credentials in the user store as mentioned below in the deployment.toml file (stored in the <MI-DASHBOARD\_HOME>/conf/ folder).

[mi\_user\_store]

username = "admin"

password = "admin"

Step 3 - Start the MI Dashboard

Follow the steps given below.

1. Open a terminal and navigate to the <MI-DASHBOARD\_HOME>/bin folder.
2. Execute one of the commands given below.

On MacOS/Linux

./dashboard.sh

On Windows

Step 4 - Start the MI servers

Follow the steps given below.

1. Open a terminal and navigate to the <MI\_HOME>/bin folder.
2. Execute one of the commands given below.

On MacOS/Linux

./micro-integrator.sh

On Windows

Step 5 - Sign in to the Dashboard

Once you have [set up and started the dashboard](https://apim.docs.wso2.com/en/latest/observe/mi-observe/working-with-monitoring-dashboard/#setting-up-the-dashboard), you can access the dashboard URL.

**Before you begin**

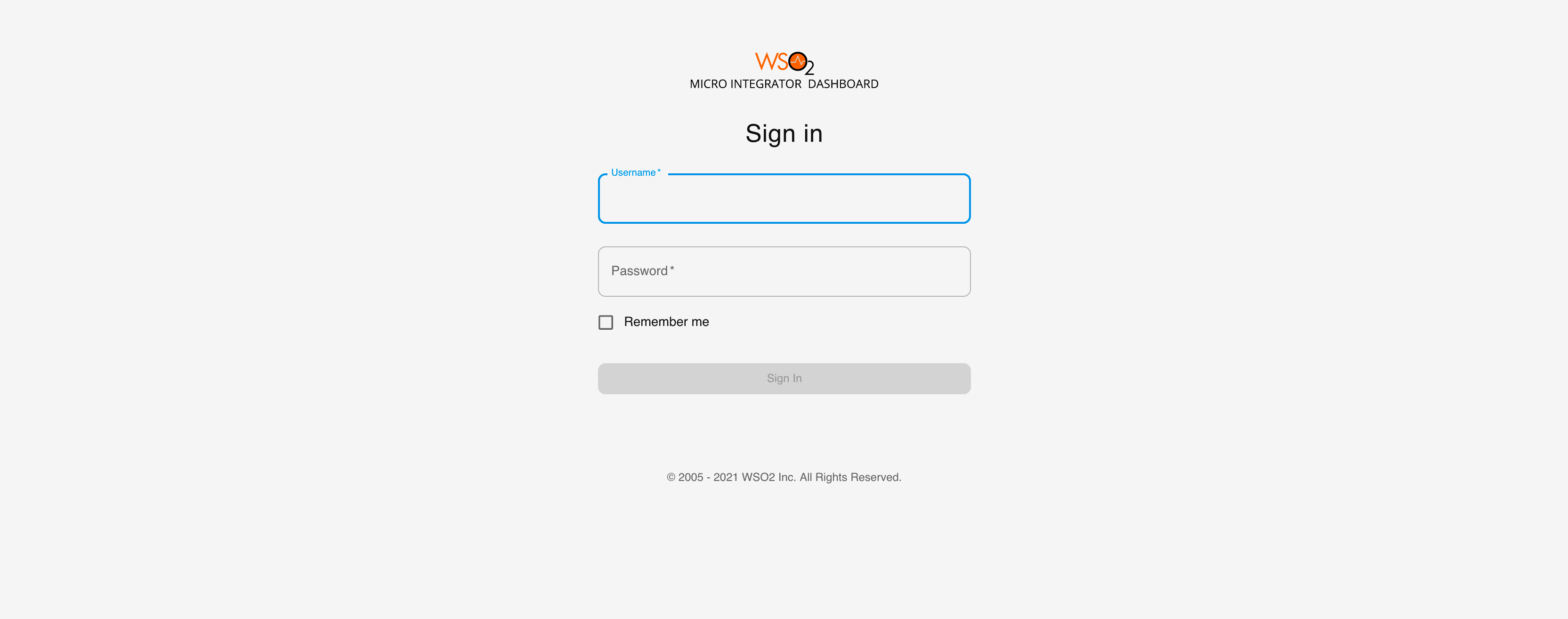
Be sure to have at least one Micro Integrator server connected to the dashboard before attempting to sign in to it. This can be verified by checking the presence of the following log.

New node <node\_id> **in** **group** : <group\_id> **is** registered. Inserting heartbeat information

1. Copy the following dashboard URL to your browser:

https://localhost:9743/login

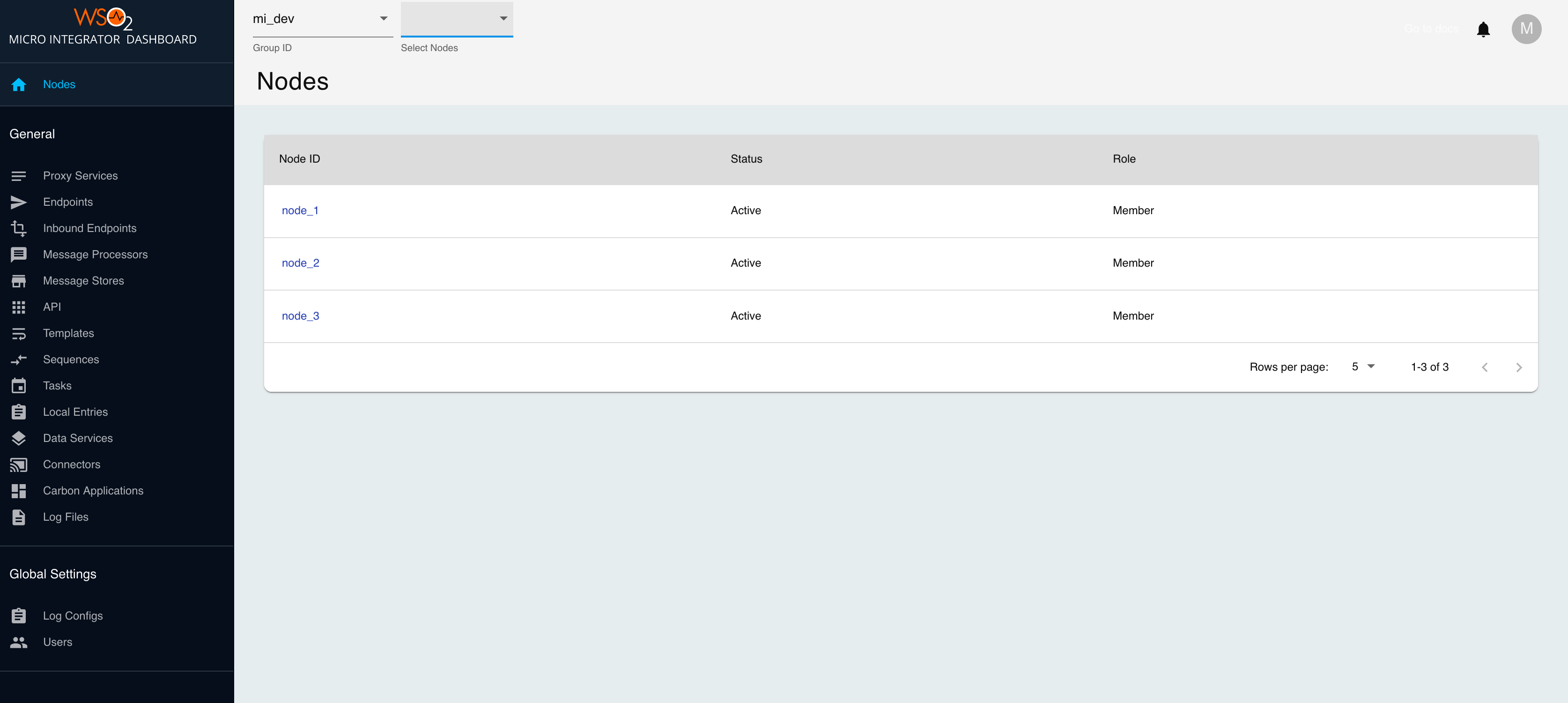
1. Enter the following details to sign in.



|  |  |
| --- | --- |
| Username | The user name to sign in.  **Note**: This should be a valid username that is saved in the Micro Integrator server's user store. By default, the 'admin' user name is configured in the default user store.  See [configuring user stores](https://apim.docs.wso2.com/en/4.1.0/install-and-setup/setup/mi-setup/user_stores/setting_up_a_userstore) for information. |
| Password | The password of the user name. By default, 'admin' is the user name and password. |

1. Click **Sign In**.

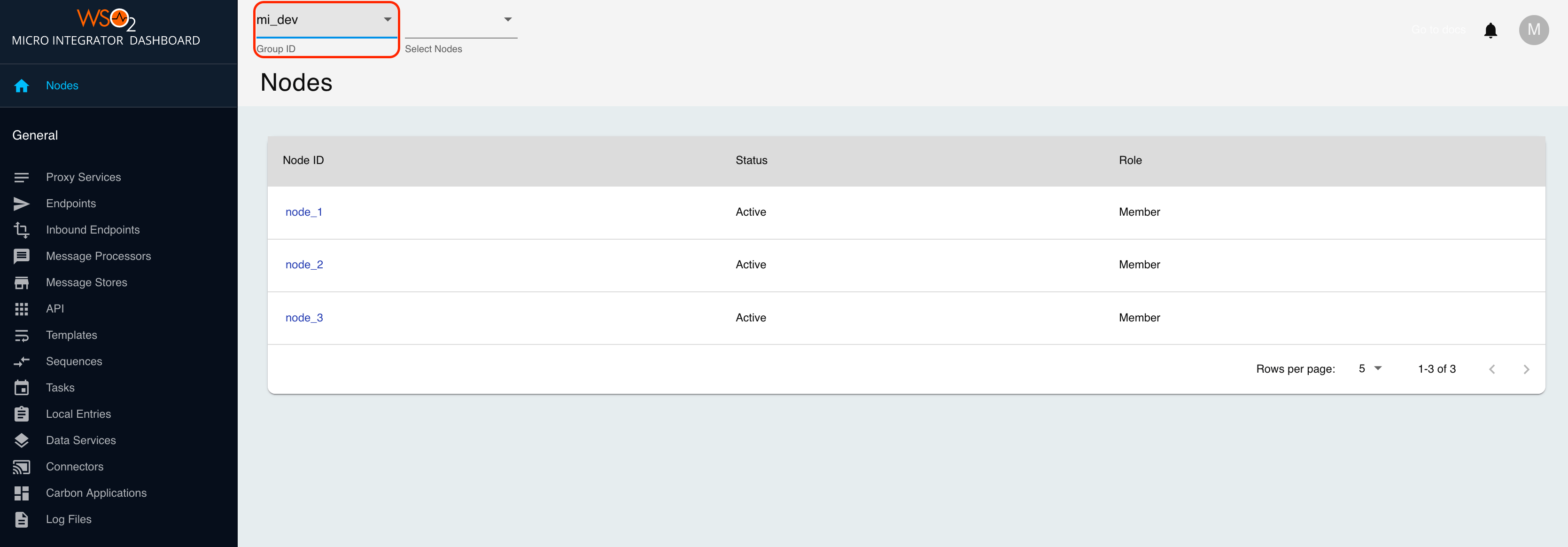
You are redirected to the home page of the Micro Integrator dashboard.

[](https://apim.docs.wso2.com/en/4.1.0/assets/img/integrate/monitoring-dashboard/dashboard-artifact-home.png)

Step 6 - Monitor MI artifacts and logs

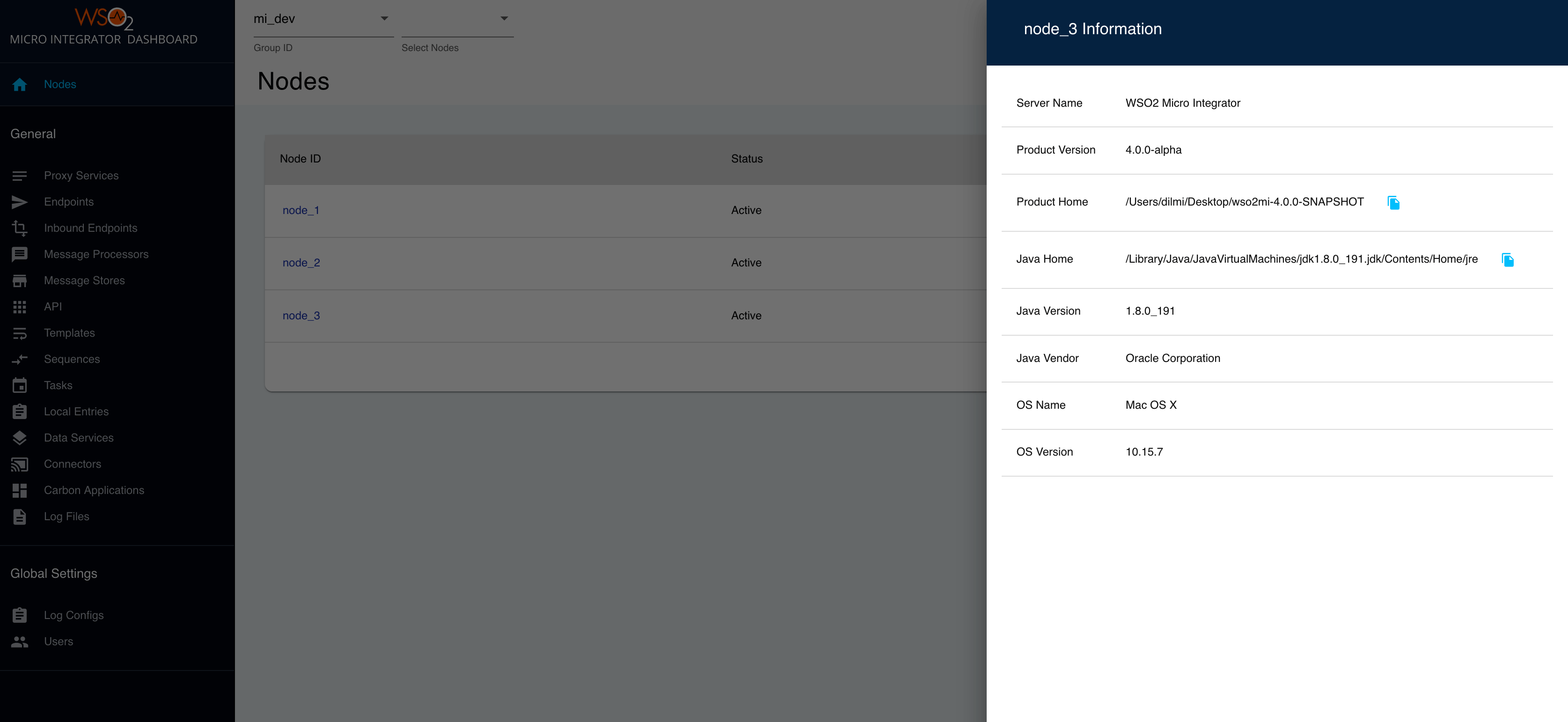
Follow the steps given below.

1. Select the group ID that you want to view from the upper left menu.

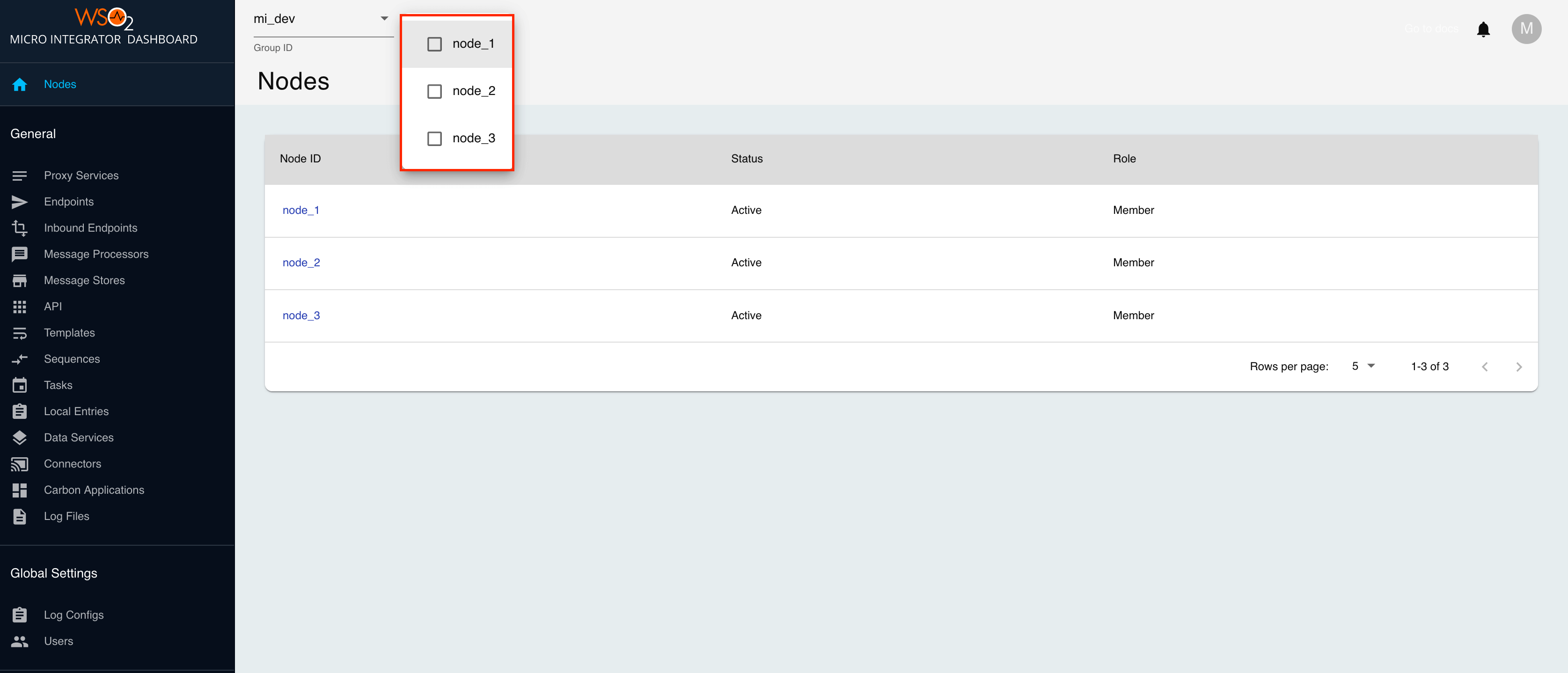
[](https://apim.docs.wso2.com/en/4.1.0/assets/img/integrate/monitoring-dashboard/dashboard-select-group.png)

You can see the list of server nodes in each group, as shown in the above diagram.

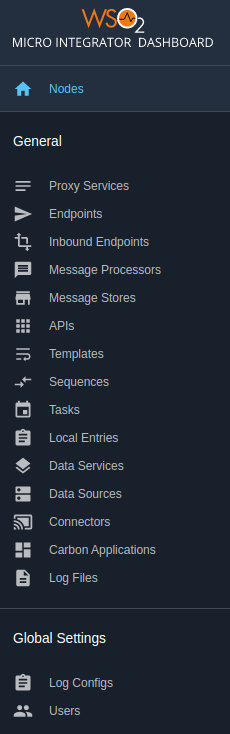
1. Click a node ID, and a side navigational panel opens to display the server information.

[](https://apim.docs.wso2.com/en/4.1.0/assets/img/integrate/monitoring-dashboard/dashboard-server-sidepanal.png)

1. Select the set of nodes you want to monitor, as shown in the below figure.

[](https://apim.docs.wso2.com/en/4.1.0/assets/img/integrate/monitoring-dashboard/dashboard-select-nodes.png)

Now you can view details of artifacts, update artifacts, and perform various other administration tasks. Select the required option from the left-hand navigator.

[](https://apim.docs.wso2.com/en/4.1.0/assets/img/integrate/monitoring-dashboard/dashboard-artifact-list.png)