KNIME Business Hub Admin Guide

KNIME AG, Zurich, Switzerland

Version 1.15 (last updated on)

Table of Contents

Introduction

KNIME Business Hub is a customer-managed Hub instance. Once you have a license for it and proceed with installation you will have access to Hub resources and will be able to customize specific features, as well as give access to these resources to your employees, organize them into Teams and give them the ability to manage specific resources.

This guide provides information on how to administrate a KNIME Business Hub instance.

To install a KNIME Business Hub please refer to the KNIME Business Hub Installation Guide - Embedded Cluster or the KNIME Business Hub Installation Guide - Existing Cluster.

A user guide is also available here, which contains instructions on how to perform team administration tasks. Team admins are designated by the global Hub admin, and have control over their team's allocated resources, can add users to their team, create execution contexts and have an overview of the team's deployments. In this way the Hub administration tasks are distributed and reallocated to those users that have a better overview of their own team necessities.

KNIME Business Hub editions

KNIME Business Hub is available in three different editions:

- Basic
- Standard
- Enterprise

All the features described in this guide are available for all the editions, unless explicitly pointed out.

However, please consider that some of the features might be limited by the resources available for the different editions.

In particular the Basic edition:

- Does not allow consumers
- It only allows 1 execution context, which is created during the installation. It is therefore not possible to create a new one unless the other one is deleted.
- It only allows 1 team. Therefore the creation of a new team is not possible.

For an overview of the available features and resources for different KNIME Hub offerings please refer to the pricing page on the KNIME website.

Users and consumers

- **Consumers** on KNIME Business Hub have access to specific data apps and services available on KNIME Hub. In KNIME Business Hub Basic edition there are no consumers available. Only logged-in users that are members of the team have access to the workflows, spaces and deployments of the team. Standard and Enterprise edition instead allows consumers. They will have unlimited access to the data apps and services that are shared with them.
- **Users** on KNIME Business Hub are members of a team and have access to all the workflows, spaces and deployments of their team, and to the public spaces of the other teams. In KNIME Business Hub Basic and Standard edition licenses 5 users are included, while 20 are included for Enterprise edition licenses.
- Unlicensed users, instead, do not have read access to any of the resources of the KNIME Business Hub for Basic and Standard edition licenses, while they have read access in the Enterprise edition licenses.

Create and manage teams

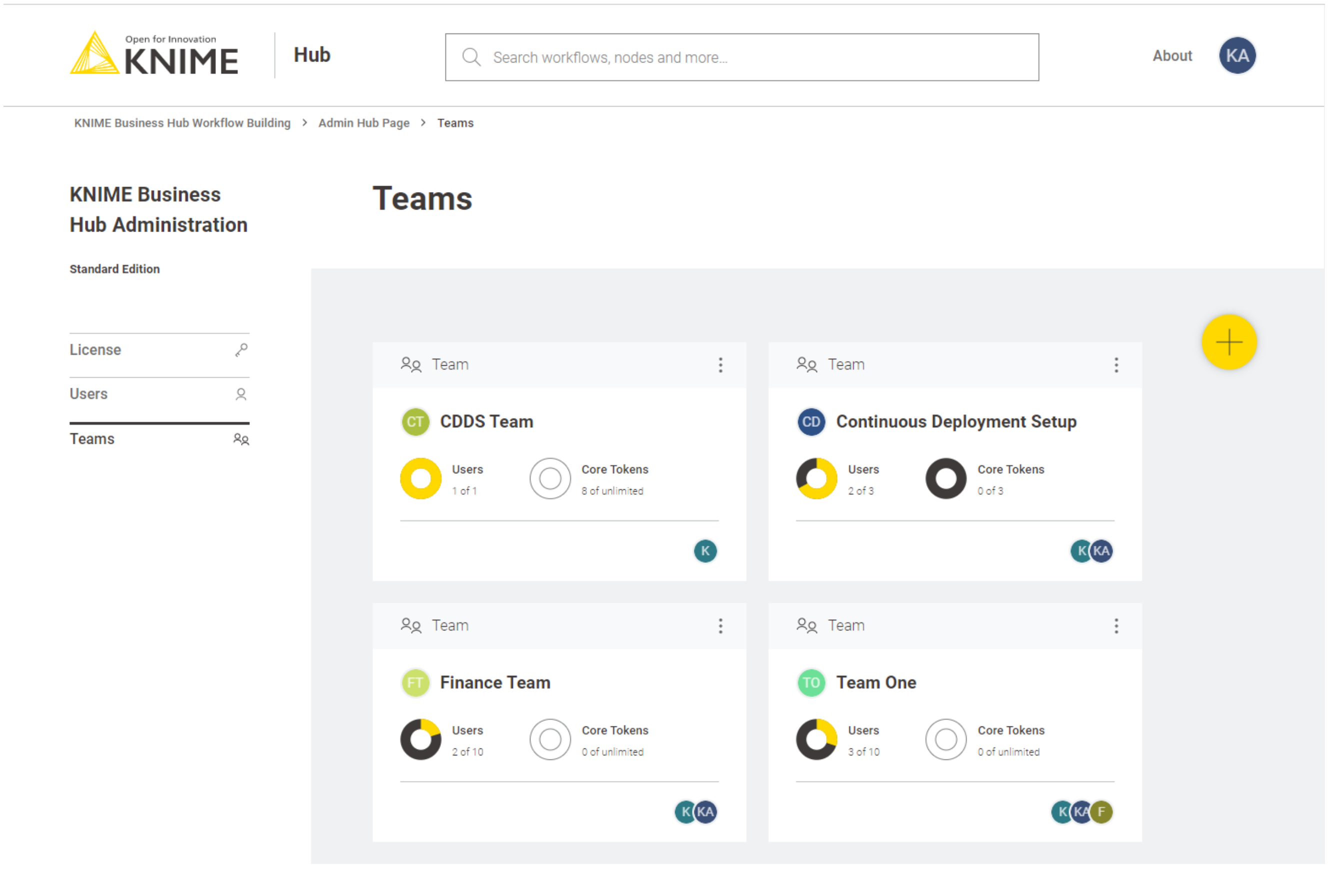
A team is a group of users on KNIME Hub that work together on shared projects. Specific Hub resources can be owned by a team (e.g. spaces and the contained workflows, files, or components) so that the team members will have access to these resources.

Sign in to the KNIME Business Hub instance with the admin user name by visiting the KNIME Business Hub URL.

Then click your profile picture on the right upper corner of the page and select *Administration* to go to the KNIME Business Hub Administration page. Click *Teams* in the menu on the left. Here you will be able to see an overview of the existing teams and you will be able to manage them.

Create a team

To create a new team click the yellow plus button on the right.



Open for Innovation KNIME
Hub
Search workflows, nodes and more...
About
KA
KNIME Business Hub Workflow Building > Admin Hub Page > Teams
KNIME Business Hub Administration
Standard Edition
License
Users
Teams
Teams
Team
CDDS Team
Users 1 of 1
Core Tokens 8 of unlimited
Team
Continuous Deployment Setup
Users 2 of 3
Core Tokens 0 of 3
Team
Finance Team
Users 2 of 10
Core Tokens 0 of unlimited
Team
Team One
Users 3 of 10
Core Tokens 0 of unlimited

Figure 1. Create a new team in the KNIME Business Hub Administration page

After you create a new team you will be redirected to the new team's page. Here you can change the name of the team. To do so click the name of the team under the team logo on the left side of the page. The name of the team can also be changed at any point in time by the team administrator.

From the team’s page you can:

- Add members to the team
- Change their role to, for example, promote a user to team admininistrator role

Here you might for example want to assign the team to a team administrator. To do so click *Manage team* and enter the user name of the user you want to assign as a team administrator for the current team. Then click on the role and select *Member* and *Admin*. At the same time you might want to delete the global admin user name from the team members list. To do so click the bin icon corresponding to that user. Click *Save changes* to finalize the setting.

Allocate resources to a team

To allocate resources to a team navigate to the KNIME Business Hub Administrator page and select *Teams* from the menu on the left.

Here you can see an overview of the teams available, their allocated resourced, and of their current usage. Click the three dots on the right upper corner of the card corresponding to the team you want to allocate resources to.

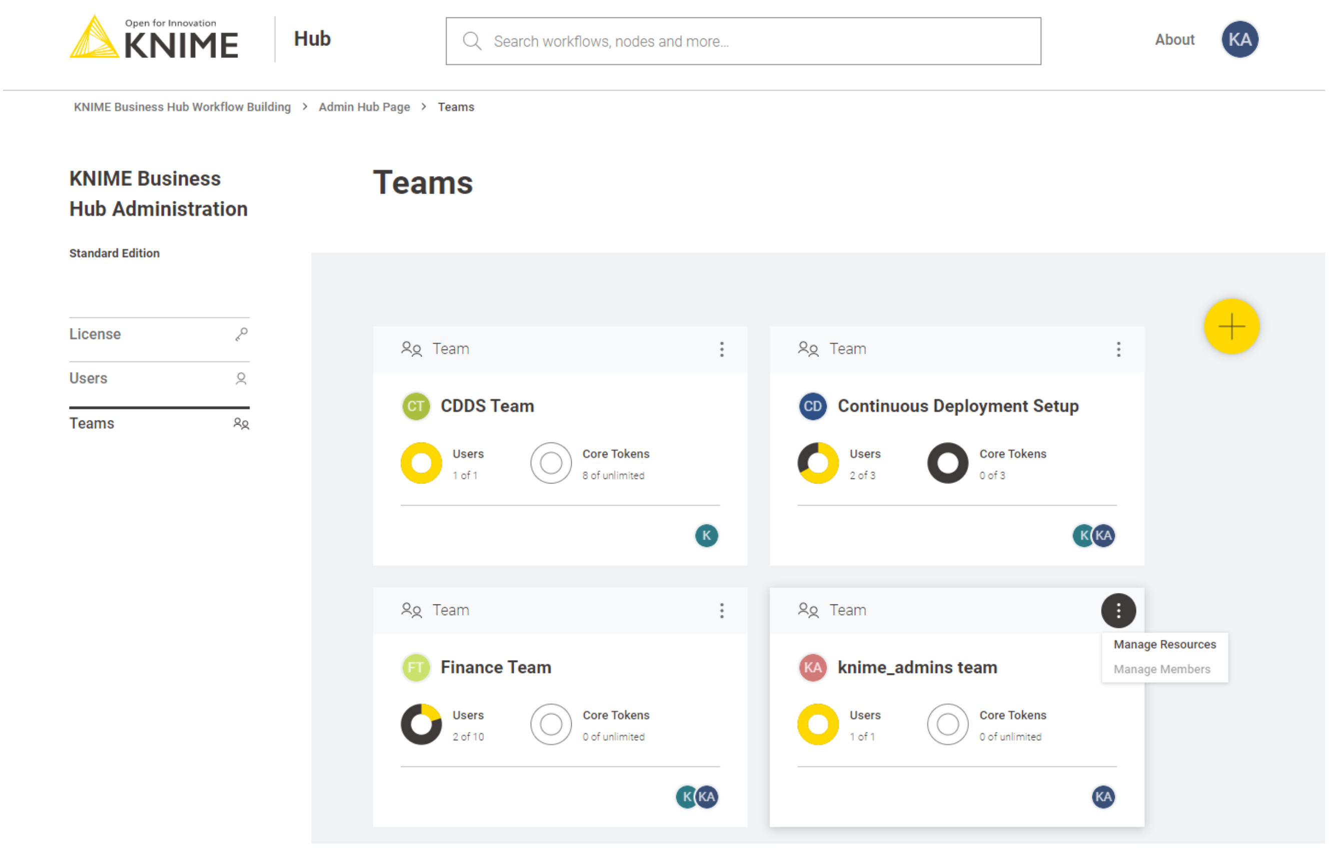
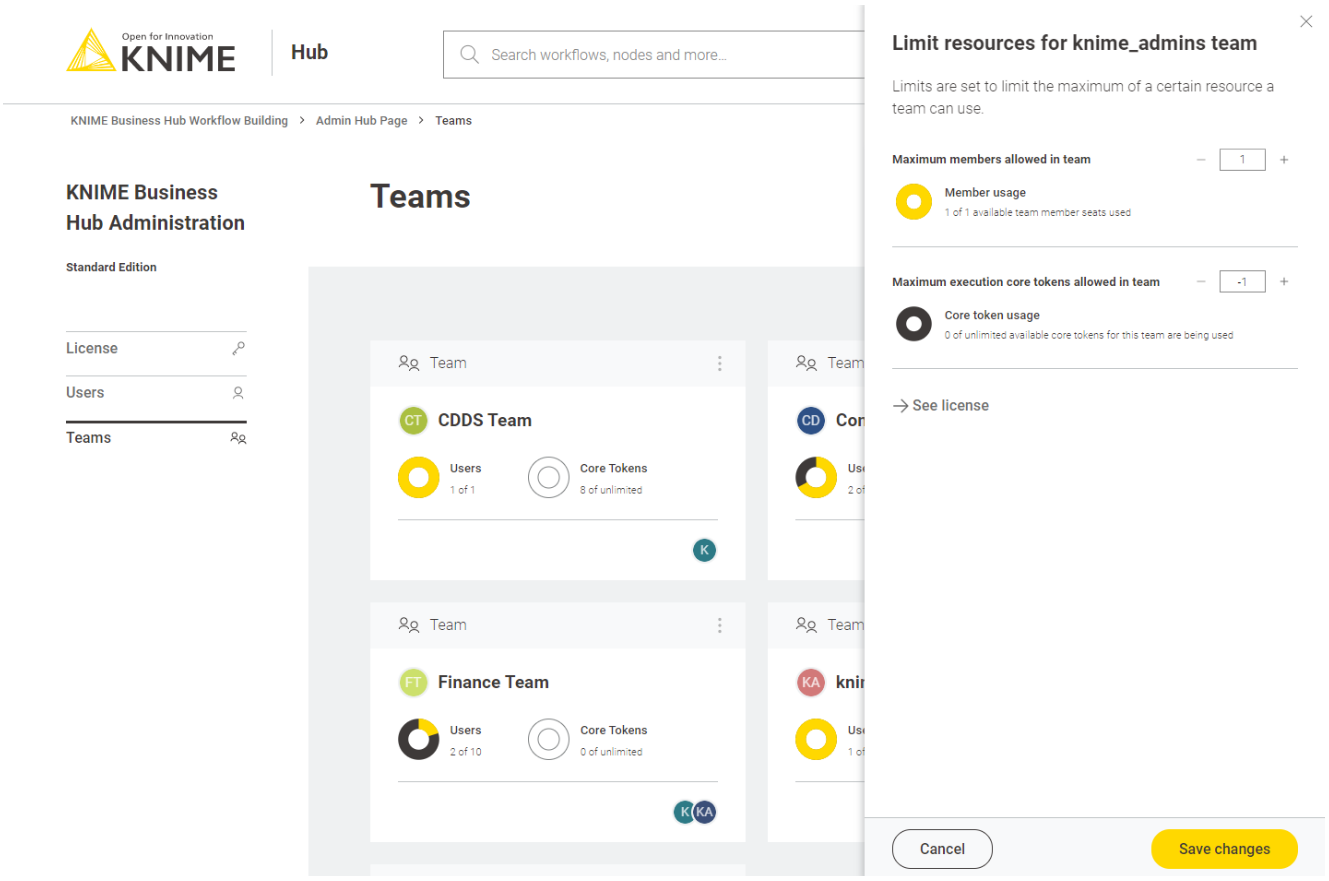

Open for Innovation
KNIME
Hub
Search workflows, nodes and more...
About
KA
KNIME Business Hub Workflow Building > Admin Hub Page > Teams
KNIME Business Hub Administration
Standard Edition
License
Users
Teams
Teams
Team
CDDS Team
Users 1 of 1
Core Tokens 8 of unlimited
Team
Continuous Deployment Setup
Users 2 of 3
Core Tokens 0 of 3
Team
Finance Team
Users 2 of 10
Core Tokens 0 of unlimited
Team
knime_admins team
Users 1 of 1
Core Tokens 0 of unlimited
Manage Resources
Manage Members

Figure 2. Manage resources of a team

Select *Manage resources* from the menu. A panel on the right will open where you can select the resources you want to allocate.



Limit resources for knime_admins team

Limits are set to limit the maximum of a certain resource a team can use.

Maximum members allowed in team: 1

Member usage
1 of 1 available team member seats used

Maximum execution core tokens allowed in team: -1

Core token usage
0 of unlimited available core tokens for this team are being used

→ See license

Figure 3. Allocate resources to a team

Here you can change:

- The maximum number of members allowed in that team
- The maximum number of execution vCore tokens allowed for that team

Click *Save changes* when you have set up the resources for the current team.

Manage team members

From the KNIME Business Hub Administration page you can also manage the team members.

Click the three dots on the right upper corner of the card corresponding to the team. From the menu that opens select *Manage members*. In the side panel that opens you can add members to a team, or change the team members role.

Delete a team

From the KNIME Business Hub Administration page you can also delete a team.

Click the three dots on the right upper corner of the card corresponding to the team. From the menu that opens select *Delete*. Be aware that this operation will delete also all the team resources, data and deployments.

Execution resources

In the following section you will find an explanation of some of the basic concepts around execution on KNIME Business Hub.

As mentioned in the previous section you as an Hub admin can assign execution resources to each team.

Team admins will then be able to build execution contexts according to the execution resources that you assigned to their team. These execution contexts will then be dedicated specifically to that team.

As an Hub admin you can also create a shared execution context. Once you create one you can share it with multiple teams.

For an overview of all the available execution contexts click your profile icon on the top right corner of the KNIME Hub and select *Administration* from the drop-down.

You will be then redirected to the KNIME Business Hub administration page.

Here, select *Execution resources* from the menu on the left.

In this page you can see an overview of *All* the execution contexts available on the Hub.

From the toggle at the top you can filter to see only a specific type of execution contexts available in the Hub instance.

Select:

- *Shared*: Shared execution contexts are created by the Hub admin. They can be made available to multiple teams.
- *Dedicated*: Dedicated execution contexts are created by the team admins for their team. Dedicated execution contexts are exclusively used by a single team.

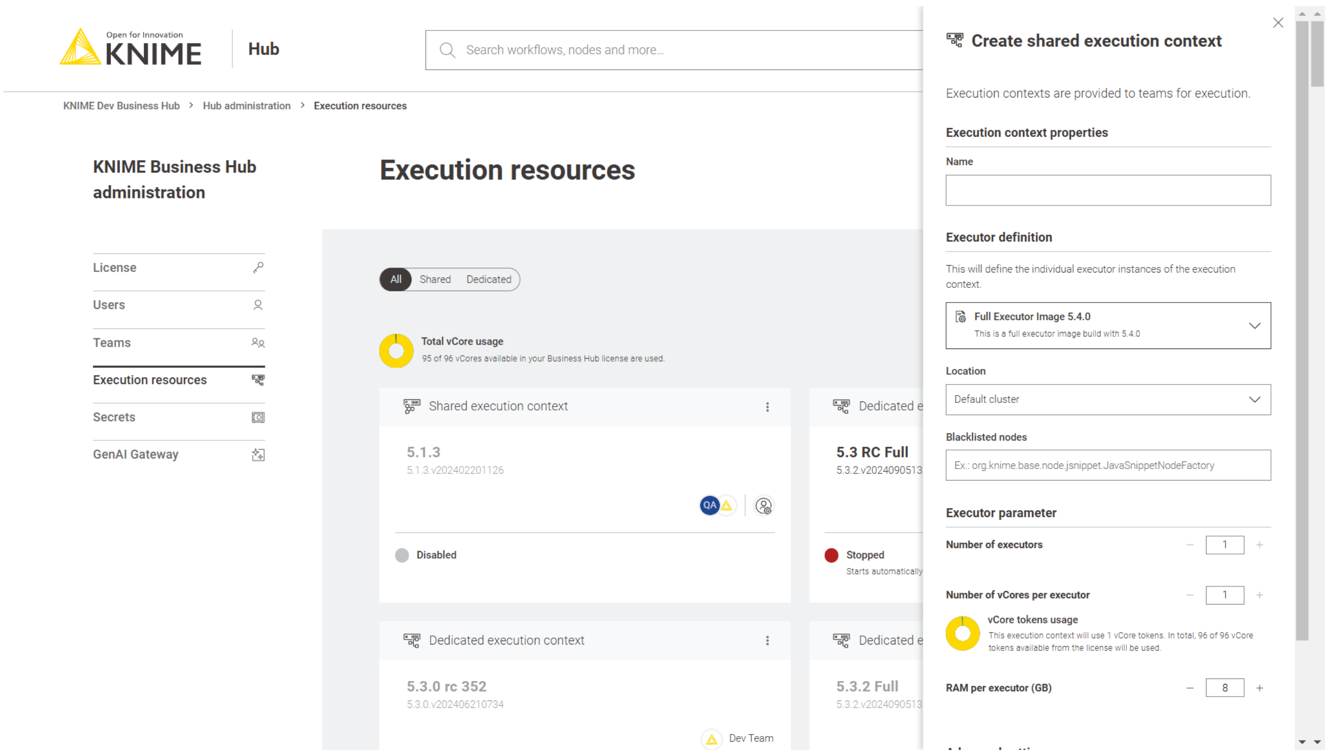
Each team can by default have a maximum of 10 execution contexts and 10000 jobs. As a KNIME Hub admin you can change these limits via a REST API call like the following:

```
PUT https://api.<base-url>/execution/limits/{scopeId}/{limitKey}
```

where `{scopeId}` is the team account ID and the `{limitKey}` is `account-execution-contexts` or `account-jobs` respectively.

Create a shared execution context

As an Hub admin you can create a shared execution context and make it available to multiple teams. To do so click the + button. A side panel opens where you can set up the new shared execution context.



Total vCore usage

95 of 96 vCores available in your Business Hub license are used.

Create shared execution context

Execution contexts are provided to teams for execution.

Execution context properties

Name

Executor definition

This will define the individual executor instances of the execution context.

Full Executor Image 5.4.0
This is a full executor image build with 5.4.0

Location: Default cluster

Blacklisted nodes: Ex.: org.knime.base.node.jsnippet.JavaSnippetNodeFactory

Executor parameter

Number of executors: 1

Number of vCores per executor: 1

vCore tokens usage

This execution context will use 1 vCore tokens. In total, 96 of 96 vCore tokens available from the license will be used.

RAM per executor (GB): 8

Figure 4. Create a shared execution context

Here you can give the execution context a name, set up which Docker image to use as the executor, give a list of blacklisted nodes, and assign the resources that the execution context will be able to use.

> Find more information about how to set up the execution context in the KNIME Business Hub User Guide.

In *Advanced settings*, you can configure more advanced settings for the execution context.

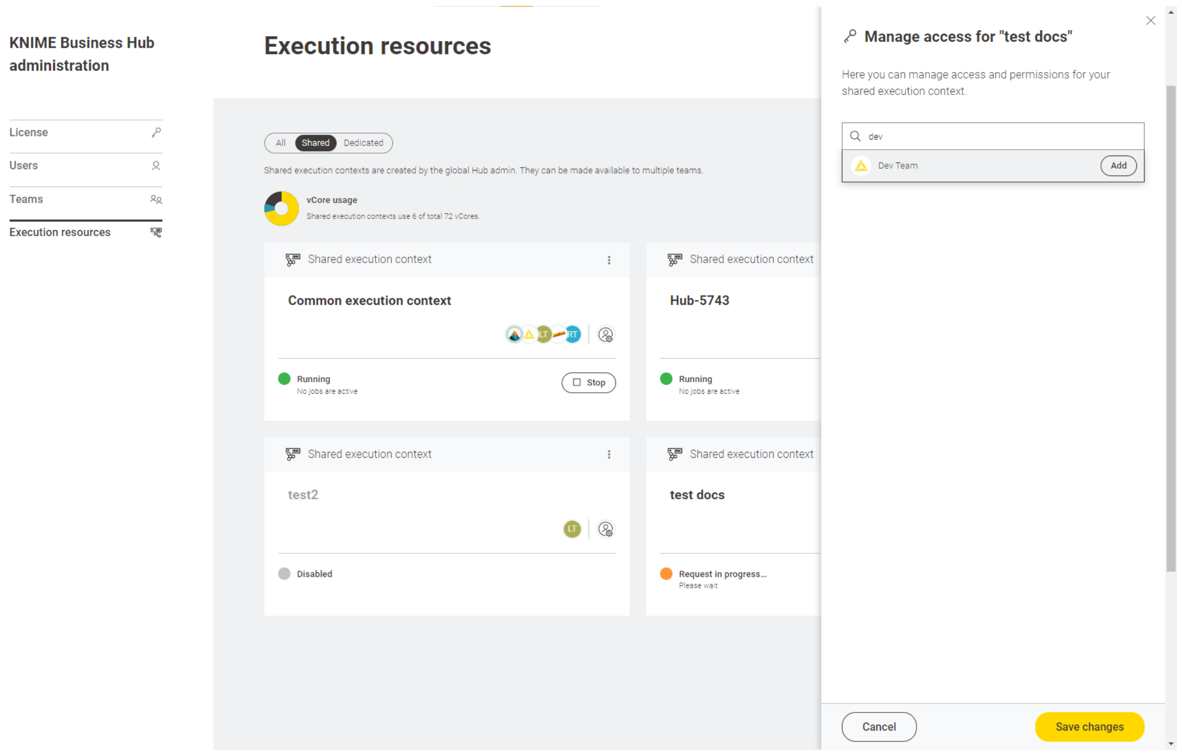
- *Configure start and stop behavior*: Configure if the execution context should automatically start and stop by selecting *On* (the setting is *On* by default) from the toggle on top. Then you can indicate the desired inactivity time (in minutes) for the execution context to stop. The execution context will start automatically when a queued workflow needs to run and stop automatically when there are no more active or queued workflows.
- *Job lifecycle*: Here you can decide when to discard a job, the maximum time a job will stay in memory, the job life time, or the options for timeout.

- *Additional settings*: Set up the report timeouts, CPU and RAM requirements, and
  - *Executor heap space limit (in %)*: Specifies the percentage of container memory available to the KNIME executor. If the container also runs other processes (e.g. Python, R, or Snowflake driver), reducing this percentage can help prevent memory issues.

> Settings set up on the deployment level will take precedence over the execution context settings.

Click *Submit* to create the execution context. A notification will appear where you can click *Manage access* to share the execution context with the teams.

At any time you can also manage the access to a shared execution context bx clicking the ⋮ icon in the corresponding tile and selecting *Manage access* from the menu.



Shared execution contexts are created by the global Hub admin. They can be made available to multiple teams.

Shared execution contexts use 6 of total 72 vCores.

Here you can manage access and permissions for your shared execution context.

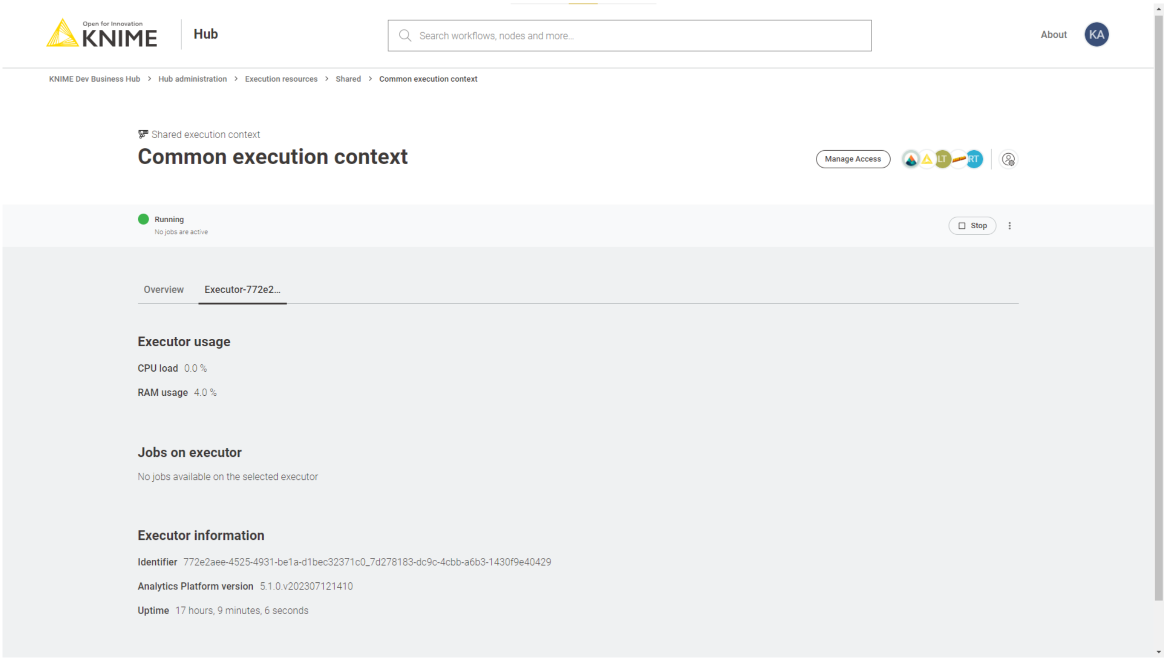
Figure 5. Manage access for a shared execution context

Manage shared execution contexts

Also from the *Execution resources* page you can have an overview about the current status of an execution context, which teams have access to it, how many jobs are running and also manage the execution context performing the following operations:

- *Start* and *Stop* an execution context by clicking the Start/Stop button in the tiles

- Click the ⋮ icon in the tile and from the menu that opens you can:
  - *Show details*: Selecting this option will open a new page with a list of all the jobs that are running on that execution context, the usage of the execution context (e.g. how many vCores are in use) and other information. You can also switch to the specific *Executor* tab to see more details about the executor.


Open for Innovation
KNIME
Hub
Search workflows, nodes and more...
About
KA
KNIME Dev Business Hub > Hub administration > Execution resources > Shared > Common execution context
Manage Access
Stop
Overview
Executor-772e2...

Shared execution context

Common execution context

Running

No jobs are active

Executor usage

CPU load 0.0 %

RAM usage 4.0 %

Jobs on executor

No jobs available on the selected executor

Executor information

Identifier 772e2aee-4525-4931-be1a-d1bec32371c0_7d278183-dc9c-4cbb-a6b3-1430f9e40429

Analytics Platform version 5.1.0.v202307121410

Uptime 17 hours, 9 minutes, 6 seconds

Figure 6. Additional executor information page

  - *Edit*: You can change the parameters and configurations in the right side panel that opens.
  - *Manage access*: Manage the access of teams to a shared execution context.
  - *Enable/Disable*: You will need first to delete the jobs associated to the execution context then proceed with disabling it.
  - *Delete*: As a Hub administrator you can delete a shared execution context. You will need to first, delete the jobs associated to the execution context then proceed with disabling it. Finally, you can delete the shared execution context.
  - *Download logs*: You can download the log files of an execution context - this feature allows for debugging in case the executors are not working as expected. You will download a `zip` file containing a log file for each executor of the execution context. Please note that to be able to download job logs you need an executor based on the following executor Docker images:
    - `knime/knime-lts-full:r-5.2.5-593` or higher bugfix releases of the 5.2.x release line
    - `knime/knime-lts-full:r-5.3.2-564` or higher bugfix releases of the 5.3.x release line

Advanced configuration of execution contexts

Execution contexts can be created and edited also via the Business Hub API.

Find more information on the available configurations in the Advanced configuration of execution contexts section in KNIME Business Hub User Guide.

Settings set up on the deployment level will take precedence over the execution context settings.

Environment variables, application arguments, and VM arguments

One aspect of the advanced configuration of execution contexts is the *operation information*, configurable via `operationInfo` in the execution context. Among others, the following configuration parameters can be set within the `operationInfo`.

- All environment variables set via `envVars` are propagated to the KNIME executor.
- Application arguments set via `applicationArguments` are the same arguments you could also set in the `knime.ini` file **before** the line containing `-vmargs`. In other words, they are different to VM arguments in that they are more general.
- VM arguments set via `vmArguments` are propagated to the JVM running on the executor. The prefix `-D` indicates a Java system property.

You can refer to the API documentation at the following URL for more details about the different calls you can make to the Execution Contexts endpoint.

```
https://api.<base-url>/api-doc/?service=execution-service#/Execution%20Contexts
```

KNIME Executor Process Watchdog Configuration

The KNIME Executor Process Watchdog helps ensure stable workflow execution by monitoring memory usage of the JVM and external processes (such as Python or Conda). When resource limits are reached, the watchdog takes corrective actions to prevent unexpected crashes. Adjusting these settings can help optimize performance and stability, especially for resource-intensive workflows.

> The features described below require specific KNIME Analytics Platform versions:
>
> - Stopping external processes: 5.2.6, 5.3.2, 5.4.0 or newer
> - Canceling node execution, native memory trimming, Grafana reporting: 5.5.0 or newer

Features

- **Stopping external processes**: When memory usage exceeds the configured limit, the executor stops the external process (Python, Conda, etc.) consuming the most memory. This may cause workflow failures but prevents crashes.
- **Canceling node execution**: If JVM memory usage exceeds 90% of the maximum, the executor cancels all running node executions.
- **Native memory trimming**: Periodically calls `malloc_trim()` to reduce memory fragmentation. Triggered when JVM memory usage exceeds 80%.
- **Reporting memory usage to Grafana**: Reports memory usage metrics to Grafana for debugging.

Configuration

Set the following properties in the `vmArguments` of the execution context:

- `-Dknime.processwatchdog.maxmemory=<value>`: Maximum memory limit for the executor and external processes. Specify in kilobytes (e.g., `2048000` for 2GB) or use `-1` to disable the watchdog.

  Default: container memory limit minus 128KB. Since: 5.4.1.

- `-Dknime.processwatchdog.pollinginterval=<value>` Interval (in milliseconds) for memory checks.

  Default: `250`. Since: 5.4.1.

- `-Dknime.processwatchdog.memorytrimintervalms=<value>`: Interval (in milliseconds) for calling `malloc_trim()`. Set to zero to disable periodic native memory trimming.

  Default: `60000` (1 min). Since: 5.5.0.

Setting `-Dknime.processwatchdog.maxmemory=-1` disables all features of the watchdog. This is not recommended, as it may lead to crashes of the executor and workflows.

Terminology

In this section you will find an explanation of some of the basic concepts around execution on KNIME Business Hub.

Executors and execution contexts

An *executor* is a headless instance of KNIME Analytics Platform that runs workflows behind the scenes. Executors typically run inside lightweight Kubernetes containers with limited memory and CPU. These containers are launched from an *executor image*, which includes a minimal operating system (Ubuntu), a KNIME Analytics Platform installation, and, if needed, tools like Python or R. KNIME provides base images, but you can customize and extend them to suit your own needs.

An *execution context* defines how one or more executors behave. It specifies which executor image to use, how much memory and CPU each executor gets, who is allowed to use it (for example, a specific team or all Hub users), and how many executors should be running. You can also configure whether executors start and stop automatically only when jobs are submitted or they need to be started and stopped manually. All executors within the same execution context are treated equally.

When someone runs a workflow, whether it is an ad-hoc execution or part of a deployment, they will be asked to choose an execution context. This defines the environment in which the job runs. One of the executors from the selected context will then pick up and execute the job.

Execution contexts can be in different states, depending on what they are doing:

Running

At least one executor is up and ready to handle jobs. If auto-stop is enabled, the execution context will shut down automatically after a configured period of inactivity.

Stopped

No executors are running, and no resources are in use. If auto-start is enabled, the context will start up automatically when someone submits a new job.

Starting up…

Executors are being launched. This involves setting up resources, downloading the container image (if needed), and initializing the executor. It usually takes a few seconds, but could take longer if the image needs to be pulled from scratch.

Shutting down…

Executors are being stopped. New jobs won’t be accepted, but current jobs will finish. If stopped forcefully, all jobs will be canceled immediately.

Restarting (updating)…

The execution context is being restarted to apply a configuration update. Executors will stop, then restart automatically using the new settings.

At full capacity

Executors are running, but all of them are busy. This might be due to a concurrency limit or resource usage (like CPU or memory). Jobs will resume once capacity frees up.

Request in progress…

A start or stop request is being handled. The state will change once the request is complete.

Disabled

The execution context has been disabled and can’t be used.

Unknown

The current status couldn’t be determined. If this doesn’t resolve, there may be an infrastructure issue.

Workflows and Jobs

Whenever a user runs a KNIME workflow, either through ad-hoc execution or a deployment, a *job* is created. This job consists of a copy of the workflow along with additional metadata such as the creator, permissions, and timestamps. Both the job (managed by KNIME Hub) and the corresponding workflow (executed by an executor) have a *state* that describes what stage they are in.

A *job* can be in one of the following states:

LOAD_REQUESTED

The job has been submitted to the queue and is waiting for an executor to pick it up.

LOADING

An executor is loading the workflow.

LOADED

The workflow has been fully loaded by an executor and is ready to run.

LOAD_ERROR

The workflow could not be loaded.

SWAP_REQUESTED

A request was sent to the executor to swap the job out of memory.

SWAPPING

The executor is in the process of swapping the job.

SWAPPED

The job was successfully swapped. Its workflow is no longer loaded in an executor but can be reloaded later from the swap area.

VANISHED

The job was loaded by an executor but can no longer be found. Usually it is because the executor crashed or was stopped.

DISCARD_REQUESTED

A request to discard the job was sent to the executor.

DISCARDED

The job was successfully discarded. Its workflow copy and all associated data have been deleted.

> When a job is **discarded** it is not removed from the history, but the job it is removed from the memory and it can't be loaded into an executor anymore to be inspected or open as data app.

The *workflow* being executed on the executor also has its own state. Once the workflow is no longer loaded, KNIME Hub shows its last known state.

CONFIGURED

The workflow is properly configured and ready to be executed.

IDLE

The workflow contains unexecuted nodes that can’t currently run. This often requires user interaction such as updating a node’s settings.

EXECUTING

One or more nodes are actively executing.

EXECUTION_FAILED

At least one node failed during execution.

EXECUTION_CANCELLED

Workflow execution was canceled by the user or system.

EXECUTION_FINISHED

All nodes have been executed.

INTERACTION_REQUIRED

The workflow is running as a data app and is waiting for user input.

EXECUTION_FAILED_WITH_CONTENT

A data app workflow failed, but some display content is still available to the user in the browser.

NOT_EXECUTABLE

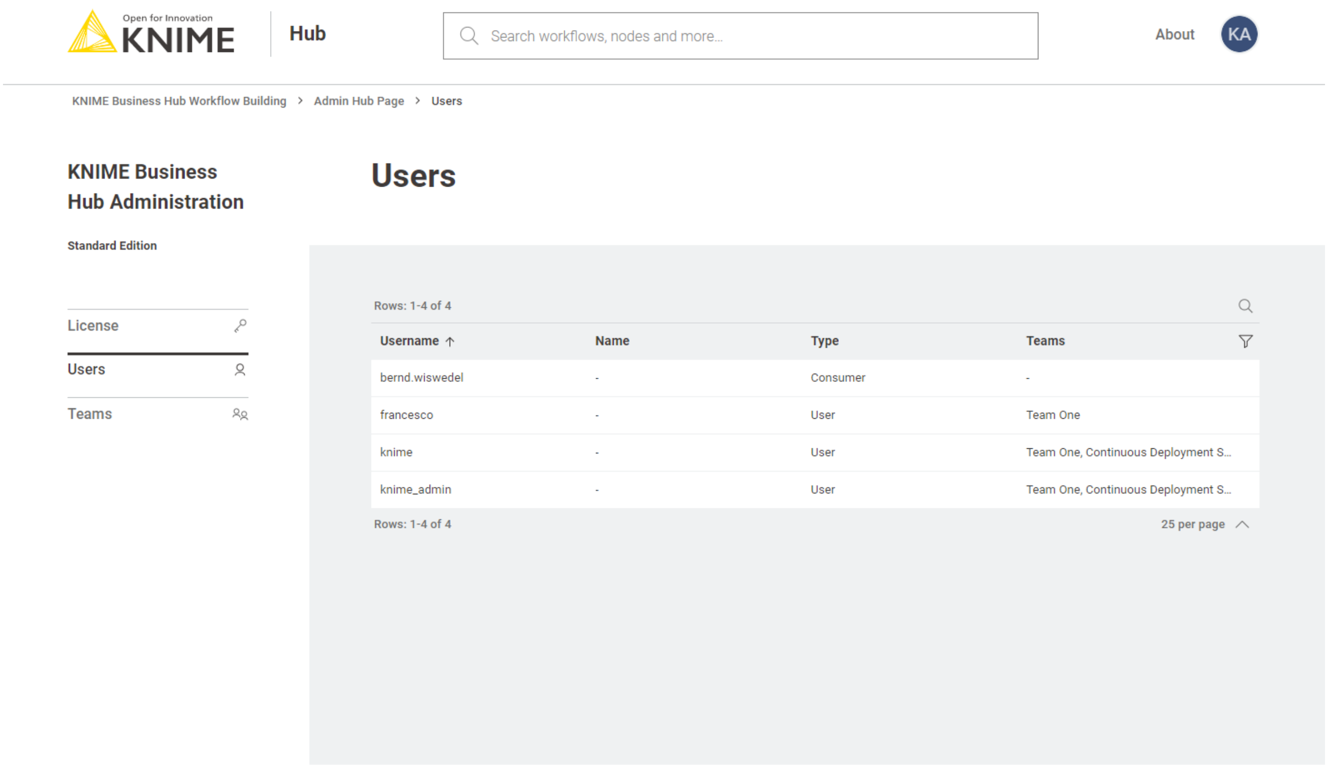
The workflow is not ready to run right after loading (similar to IDLE).

The job and workflow states are not directly shown in the KNIME Hub UI, but are accessible through the Hub API.

Users management

Keycloak, an open source identity and access management solution, is embedded in KNIME Business Hub and is where users are managed in the backend.

However, if you want to see the users that have access to your KNIME Business Hub instance you can go to the KNIME Business Hub Administration page and select *Users* from the menu on the left. The list shows all users that have already logged into KNIME Business Hub.


Open for Innovation KNIME
Hub
Search workflows, nodes and more...
About
KA
KNIME Business Hub Workflow Building > Admin Hub Page > Users
KNIME Business Hub Administration
Standard Edition
License
Users
Teams
Users
Rows: 1-4 of 4
25 per page

Username	Name	Type	Teams
bernd.wiswedel	-	Consumer	-
francesco	-	User	Team One
knime	-	User	Team One, Continuous Deployment S...
knime_admin	-	User	Team One, Continuous Deployment S...

Figure 7. See users on KNIME Business Hub Administration page

Here you can filter the users based on their team, the type of users and their username and name. To do so click the funnel icon in the users list. You can also search the users by using the magnifier icon and typing the keyword in the field that appears.

> Users that only exist in your identiy provider are not known to KNIME Business Hub. If you want to create users before they log in, e.g. in order to assign them to groups or share deployments with them, you can provision users with SCIM. See Provisioning users and groups with SCIM (Enterprise and Standard edition only) below for details.

Delete a user

To delete a user follow these steps in this order:

1. Delete the user from the KNIME Business Hub Administration page. Click the three dots and select *Delete*. You will need to confirm the user deletion in the window that opens by clicking *Delete user*. Be aware that this action will also delete all data from the deleted user and it will not be possible to restore the user.
2. Delete the user from Keycloak. Follow the steps in the next section to access Keycloak and manage users.

Access Keycloak for users management

1. First you will need to access the Keycloak admin console. To do so you will need the credentials that are stored in a kubernetes secret called `credential-knime-keycloak` in the `<business-hub-namespace>` namespace. To get the required credentials, you need to access the instance the Business Hub is running on and run the following command:

   ```
   kubectl -n knime get secret credential-knime-keycloak -o yaml
   ```

   This will return a file that contains the `ADMIN_PASSWORD` and the `ADMIN_USERNAME`. These files are both `base64` encrypted. In order to get the decrypted username and password, you can run the following commands:

   ```
   echo <ADMIN_PASSWORD> | base64 -d
   echo <ADMIN_USERNAME> | base64 -d
   ```

2. Then go to `http://auth.<base-url>/auth/` and log in.

Make a user Hub admin

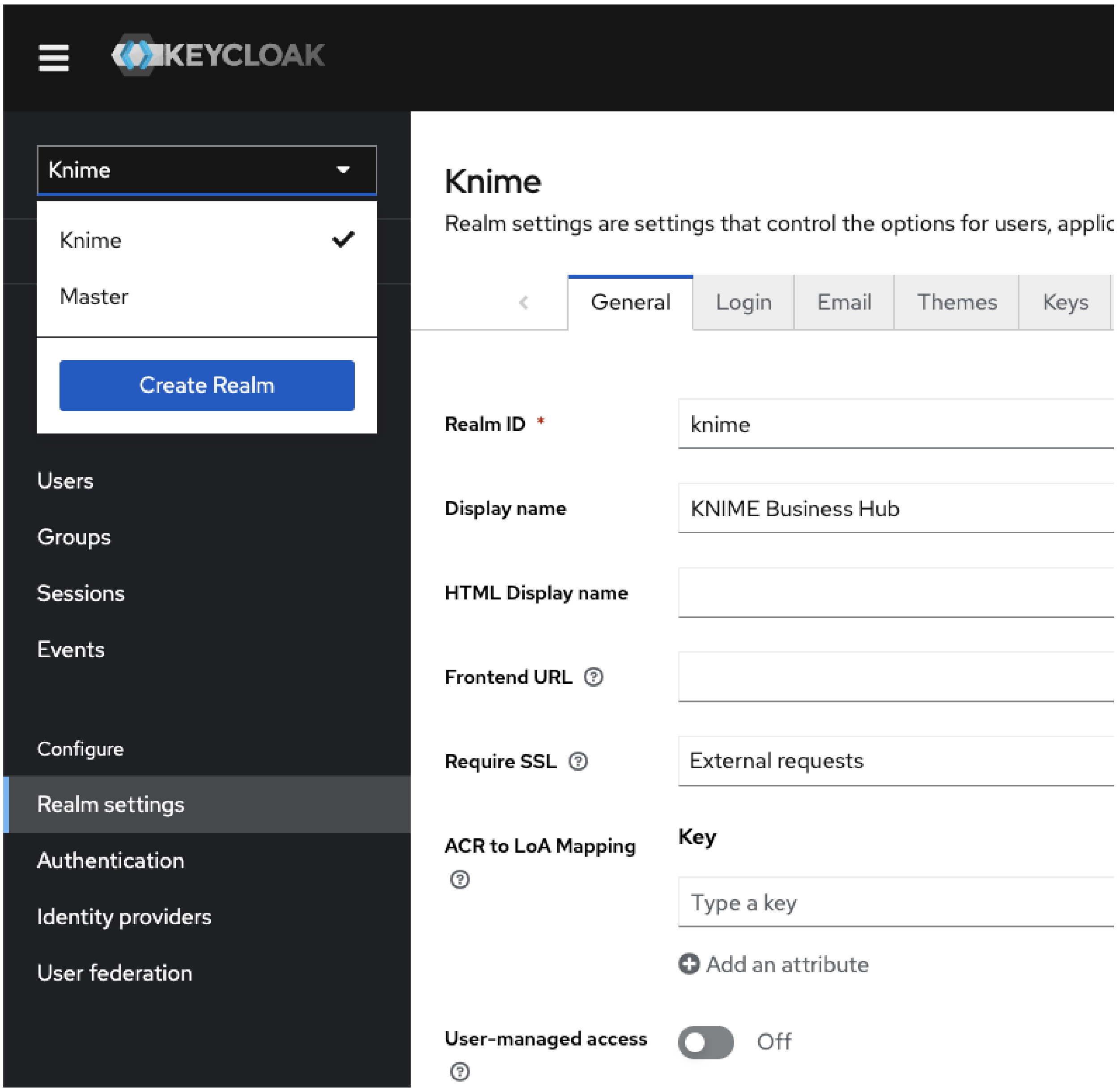
The operation of promoting a registered user to the role of Hub admin is done in Keycloak.

To promote a user to Hub admin role, follow these steps:

1. First, access Keycloak admin console by going to `http://auth.<base-url>/auth/` and logging in.

   > Follow the instructions in the section Access Keycloak for users management if you need to retrieve Keycloak credentials.

2. In the top left corner click the dropdown and select the "Knime" realm, if you are not there already.



Knime

Realm settings are settings that control the options for users, applic

Realm ID * knime

Display name KNIME Business Hub

HTML Display name

Frontend URL

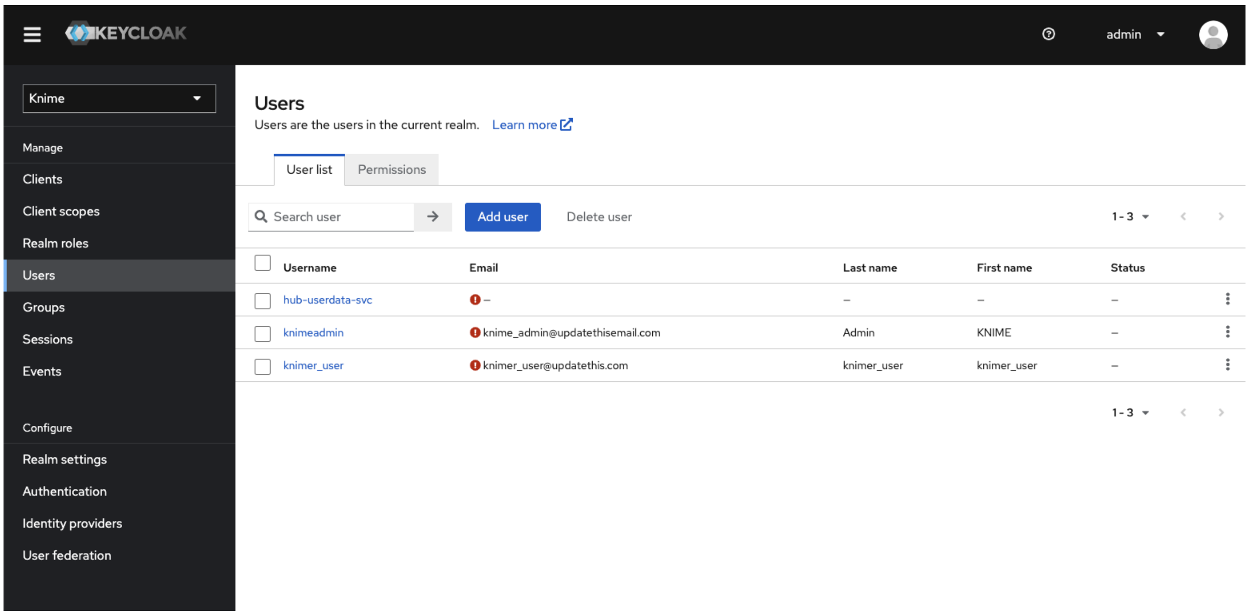
Require SSL External requests

ACR to LoA Mapping Key Type a key

User-managed access Off

Figure 8. Select the "Knime" realm

3. Navigate to the *Users* menu and search for the user by name or email:

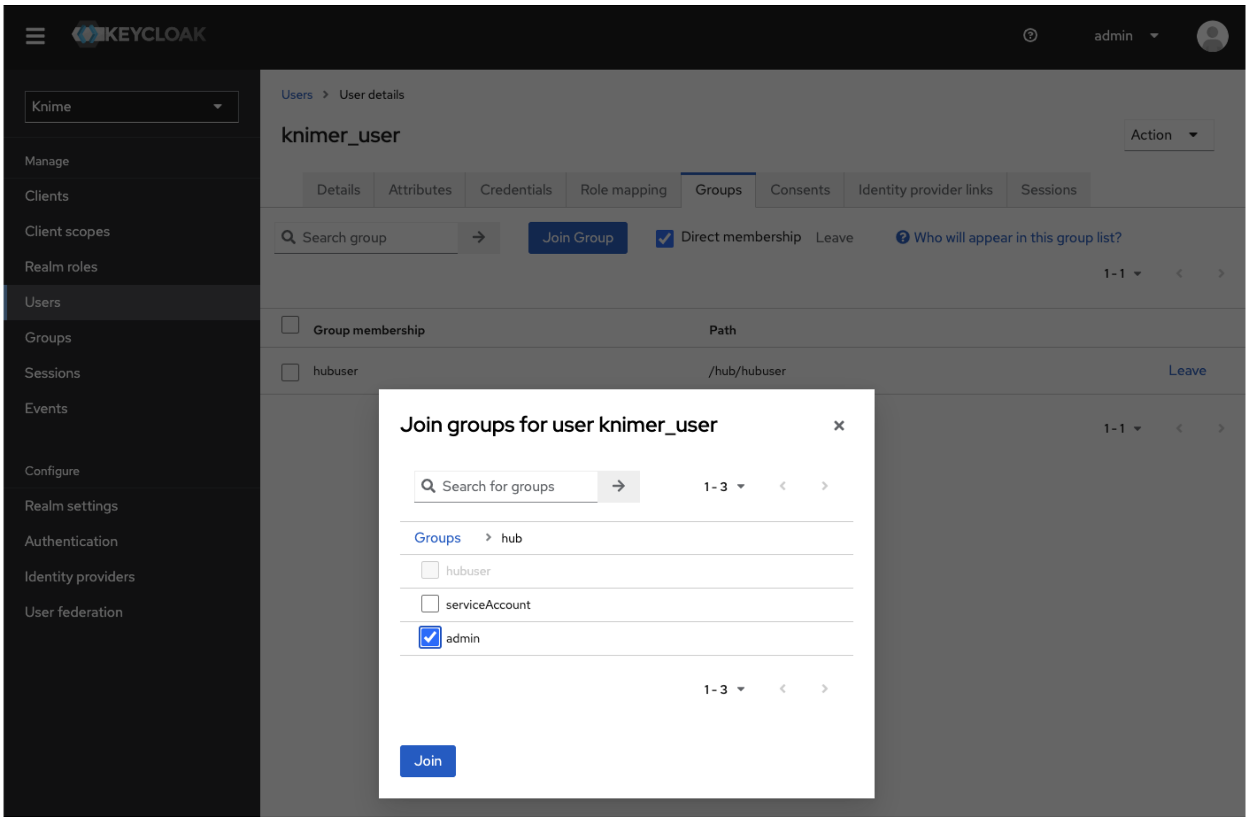


Username	Email	Last name	First name	Status
hub-userdata-svc	–	–	–	–
knimeadmin	knime_admin@updatethisemail.com	Admin	KNIME	–
knimer_user	knimer_user@updatethis.com	knimer_user	knimer_user	–

Figure 9. The Keycloak users menu

In order for a user to appear in this list, it is necessary that they have logged into your KNIME Business Hub installation at least once.

4. Click the user and go to the *Groups* tab. Click *Join Group* and either expand the *hub* group by clicking it, or search for "admin". Select the admin group and click *Join*:



Group membership	Path	
hubuser	/hub/hubuser	Leave

Join groups for user knimer_user

- hubuser
- serviceAccount
- [✓] admin

Figure 10. Making a user a Hub admin in Keycloak. If you are searching for the group then the group might show up under its full path "/hub/admin"

5. Done. The user now has the role of Hub admin, and can access the admin pages from within the Hub application to e.g., create teams or delete users.

Provisioning users and groups with SCIM (Enterprise and Standard edition only)

The System for Cross-domain Identity Management (SCIM) is an open industry standard for provisioning users and groups from an identity provider to an application. In contrast to authentication protocols such as OpenID/OIDC, a SCIM client in the identity provider actively pushes information about user and groups to the application.

This allows you to create users in KNIME Business Hub without them having to log in first. You can then assign them to teams or share deployments with them. In addition, groups can be provisioned with SCIM as well. They become *external* groups in KNIME Business Hub. In combination with the nested groups feature they allow you to automatically assign users to teams based on group memberships in the identity provider.

You can enable SCIM on a KNIME Business Hub installation that already contains users. Existing KNIME Business Hub user accounts will be matched with accounts provisioned by SCIM based on the username and e-mail address.

If your usernames in the identity provider are e-mail addresses you likely have created a mapper in Keycloak that removes the domain part because previous KNIME Business Hub releases did not allow usernames to contain the `@` character. This has changed for version 1.10 so that SCIM clients can provision users with e-mail addresses as usernames without issues. However, this also means you have to disable/remove the corresponding mapper in Keycloak, otherwise the username created by SCIM does not match the username in the authentication token provided by Keycloak. After you have removed the mapper, existing accounts in KNIME Business Hub will automatically be updated when the user logs in the next time.

We have tested and hence support SCIM with Okta and Microsoft Entra ID as SCIM clients. Other SCIM clients may work as well but as the SCIM protocol is pretty extensive therefore there may be edge-cases which we haven’t covered.

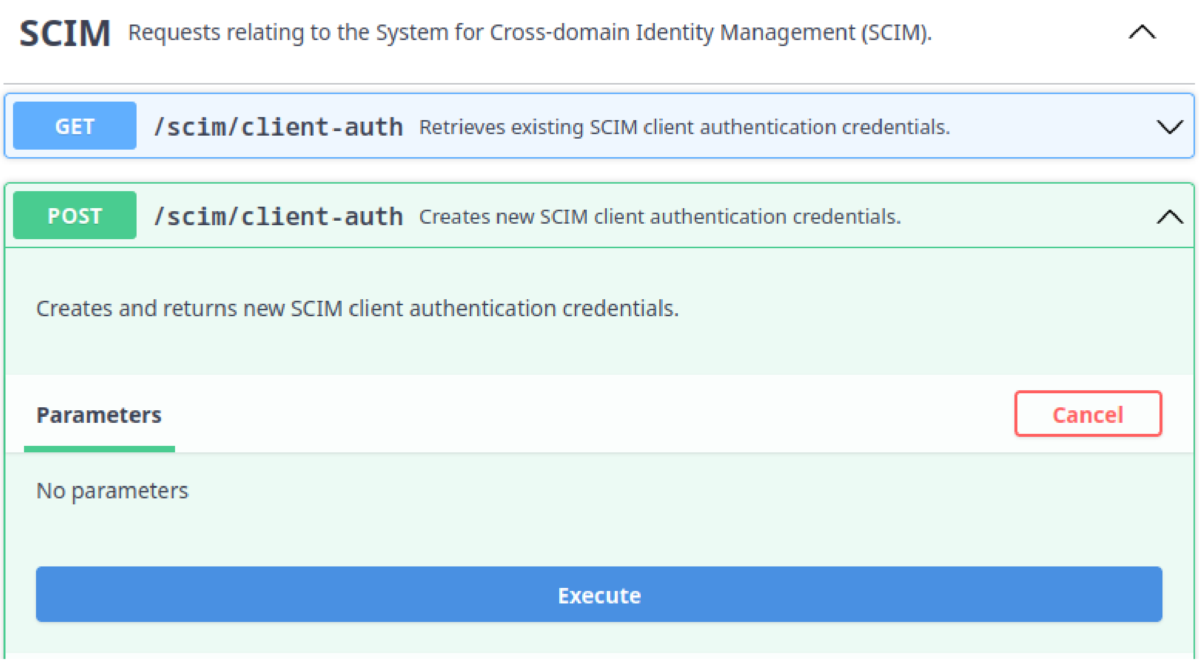
Create a SCIM application password

The first step for any kind of SCIM client is to create a dedicated application password with which the SCIM client authenticates against your KNIME Business Hub installation. This must be done by an administrator and can best be performed using the Swagger UI.

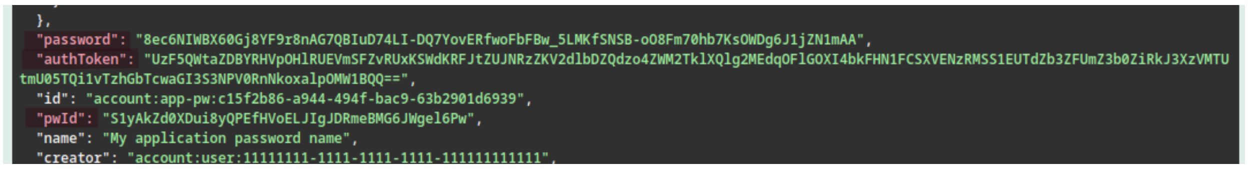
1. Log into your KNIME Business Hub installation as an administrator user.
2. Open the OpenAPI documentation of the account service functionality, which is

available at `https://api.<base-url>/api-doc/?service=accounts-service`

3. Go to the *SCIM* section, expand the `POST` request box, click *Try it out*, and then click *Execute*.


SCIM
Requests relating to the System for Cross-domain Identity Management (SCIM).
GET
/scim/client-auth
Retrieves existing SCIM client authentication credentials.
POST
/scim/client-auth
Creates new SCIM client authentication credentials.
Creates and returns new SCIM client authentication credentials.
Parameters
Cancel
No parameters
Execute

4. The result will be displayed below the button. It contains a `pwId`, a `password`, and an `authToken`. For Okta as a SCIM client you will need the `pwId` and the `password`. For Microsoft Entra ID you will need the `authToken`. Please note down these values in a safe place because you will not be able to retrieve them again without deleting and re-creating the SCIM application password.



```
  },
  "password": "8ec6NIWBX60Gj8YF9r8nAG7QBIuD74LI-DQ7YovERfwoFbFBw_5LMKfSNSB-oO8Fm70hb7KsOWDg6J1jZN1mAA",
  "authToken": "UzF5QWtaZDBYRHVpOHlRUEVmSFZvRUxKSWdKRFJtZUJNRzZKV2dlbDZQdzo4ZWM2TklXQlg2MEdqOFlGOXI4bkFHN1FCSXVENzRMSS1EUTdZb3ZFUmZ3b0ZiRkJ3XzVMTU
tmU05TQi1vTzhGbTcwaGI3S3NPV0RnNkoxalpOMW1BQQ==",
  "id": "account:app-pw:c15f2b86-a944-494f-bac9-63b2901d6939",
  "pwId": "S1yAkZd0XDui8yQPEfHVoELJIgJDRmeBMG6JWgel6Pw",
  "name": "My application password name",
  "creator": "account:user:11111111-1111-1111-1111-111111111111",
```

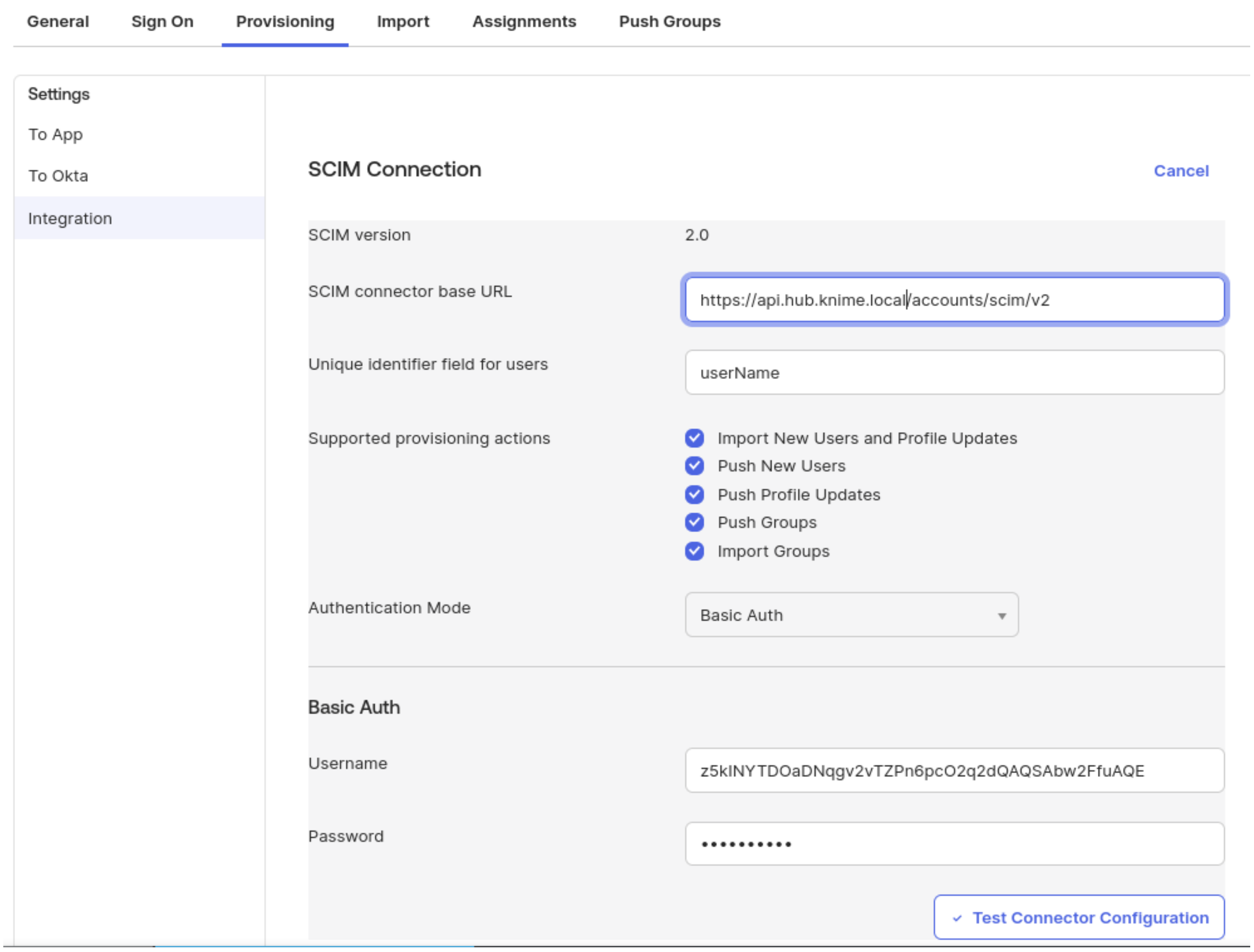
Configure Okta as a SCIM client

In order to set up Okta as a SCIM client for KNIME Business Hub, follow Okta's documentation about Adding SCIM provisioning to app integrations.

It is currently not possible to add SCIM to an OIDC app integration in Okta. But you can simply create a second integration for your KNIME Business Hub installation that is only responsible for provisioning via SCIM. For this create a *SWA app integration* as part of *Task 1* of the documentation.

When you configure the *SCIM Connection* of the app integration, provide the following values:

- Use `https://api.<your hub domain>/accounts/scim/v2` as the *SCIM connector base URL*.
- Use *userName* as *Unique identifier field for users*.
- Select all options for *Supported provisioning actions*.
- Select *Basic Auth* as *Authentication Mode*.
- From the application password created earlier use the `pwId` value as *Username* and the `password` value as *Password*.



SCIM Connection

SCIM version: 2.0

SCIM connector base URL: https://api.hub.knime.local/accounts/scim/v2

Unique identifier field for users: userName

Supported provisioning actions:
- ✓ Import New Users and Profile Updates
- ✓ Push New Users
- ✓ Push Profile Updates
- ✓ Push Groups
- ✓ Import Groups

Authentication Mode: Basic Auth

Basic Auth

Username: z5kINYTDOaDNqgv2vTZPn6pcO2q2dQAQSAbw2FfuAQE

Password: ••••••••••

After that you can assign users and/or groups to the application in the *Assignments* tab. Provisioning of the respective users will start automatically shortly afterwards. Likewise, group provisioning can be configured in the *Push Groups* tab.

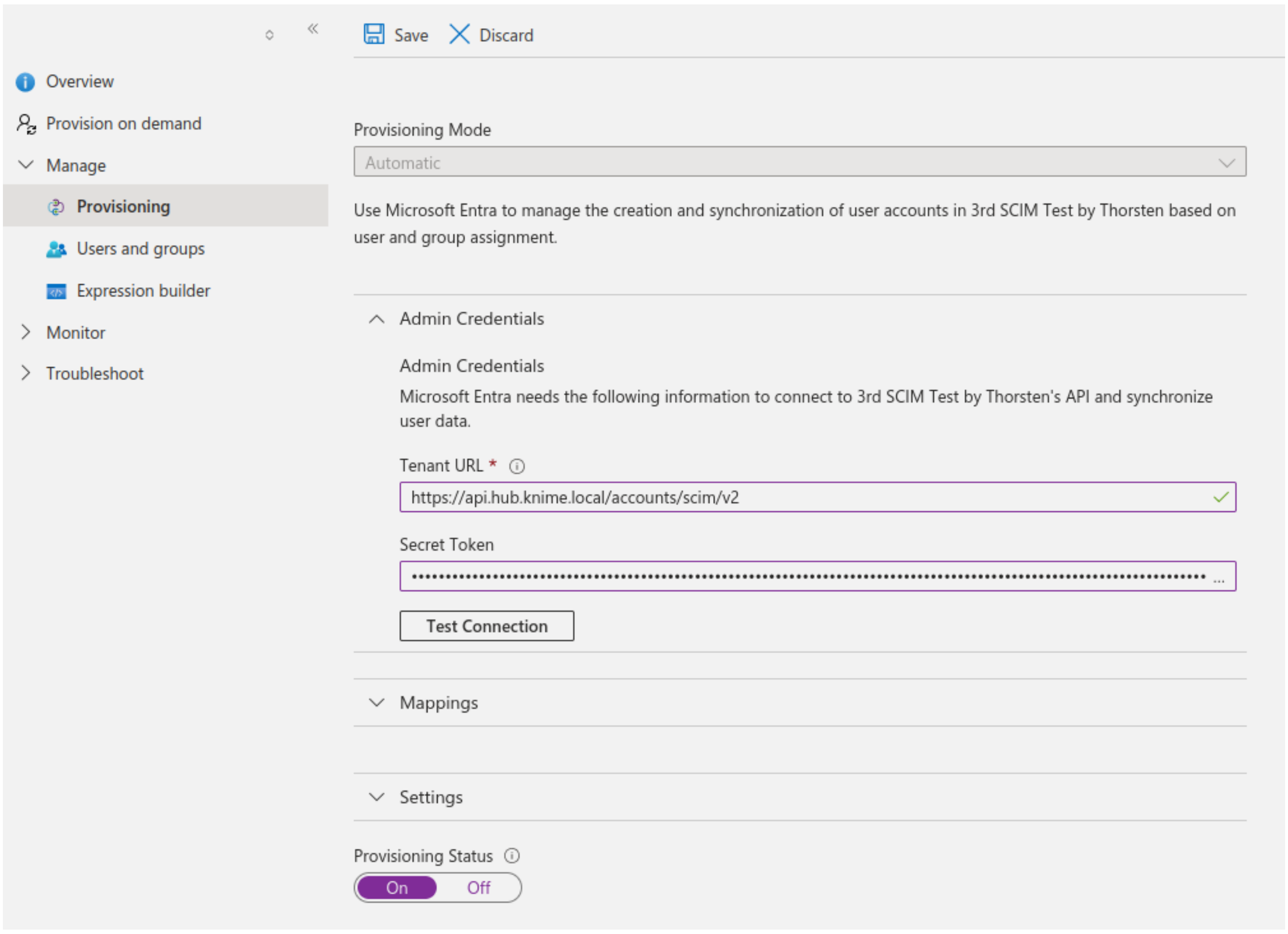
Configure Microsoft Entra ID as a SCIM client

In order to set up Microsoft Entra ID as a SCIM client for KNIME Business Hub, follow Microsoft's documentation about Integrating your SCIM endpoint with the Microsoft Entra provisioning service.

When you configure the *Provisioning* details of the Enterprise application, provide the following values:

- Use `https://api.<your hub domain>/accounts/scim/v2` as the *Tenant URL*
- Use the `authToken` value from the application password created earlier as the *Secret Token*

No changes are required in the *Mappings* section.



Provisioning Mode
Automatic

Use Microsoft Entra to manage the creation and synchronization of user accounts in 3rd SCIM Test by Thorsten based on user and group assignment.

Admin Credentials

Admin Credentials
Microsoft Entra needs the following information to connect to 3rd SCIM Test by Thorsten's API and synchronize user data.

Tenant URL *
https://api.hub.knime.local/accounts/scim/v2

Secret Token
••

Provisioning Status
[✓ On] Off

With Entra ID provisioning can take a long time, depending on the number of users and groups. For example, 250 users and 30 groups take about 40 minutes.

If your KNIME Business Hub installation is not reachable by Entra ID due to network restrictions, you can follow Microsoft’s instructions about Configuring the connection via the provisioning agent

Assign user to teams based on external groups

If you have provisioned groups from your identity provider to KNIME Business Hub you can use these external groups to manage team memberships. Once changes to group memberships in the identity provider have been provisioned to KNIME Business Hub they will immediately be reflected in team memberships. This also allows you to assign users to teams before they have logged in.

> Assigning a consumer account to a team requires a free user seat license. If there are no more unused user seats available in your KNIME Business Hub installation, the assignment will fail. If your license comes with an unlimited number of users this issue will not affect your installation. In case sufficient user seats are available, team sizes are adjusted automatically when using SCIM.

In order to assign team membership based on an external group, you have to add the external group as a *nested group* to the team's *accountMember* group.

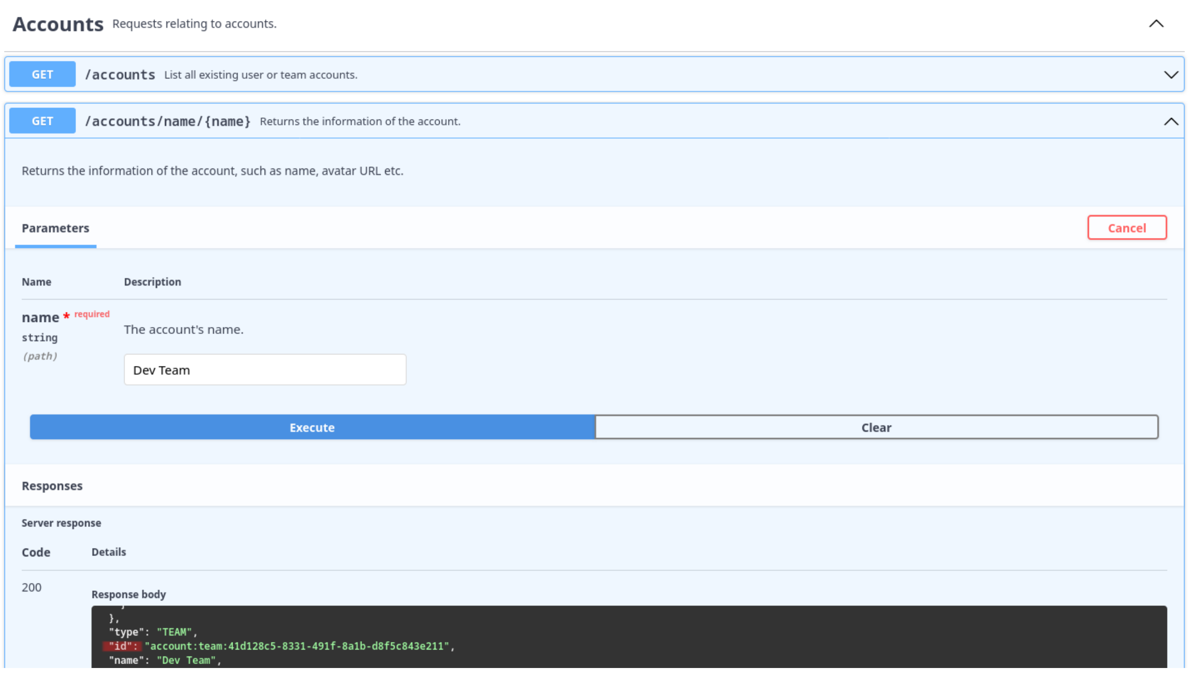
Currently, you can accomplish this by:

1. Using **API calls**, such as through the Swagger UI that you used to create the SCIM application password.
2. By using a dedicated **data application** deployed in your KNIME Business Hub instance.

Assign users based on external groups via API calls

Follow these steps in order to add an external group as a nested group to a team group:

1. Find the team's ID. You can search teams by their name and then get the ID from the `id` field in the response.


Accounts Requests relating to accounts.
GET
/accounts List all existing user or team accounts.
GET
/accounts/name/{name} Returns the information of the account.
Parameters
Cancel
Execute
Clear
Responses

Returns the information of the account, such as name, avatar URL etc.

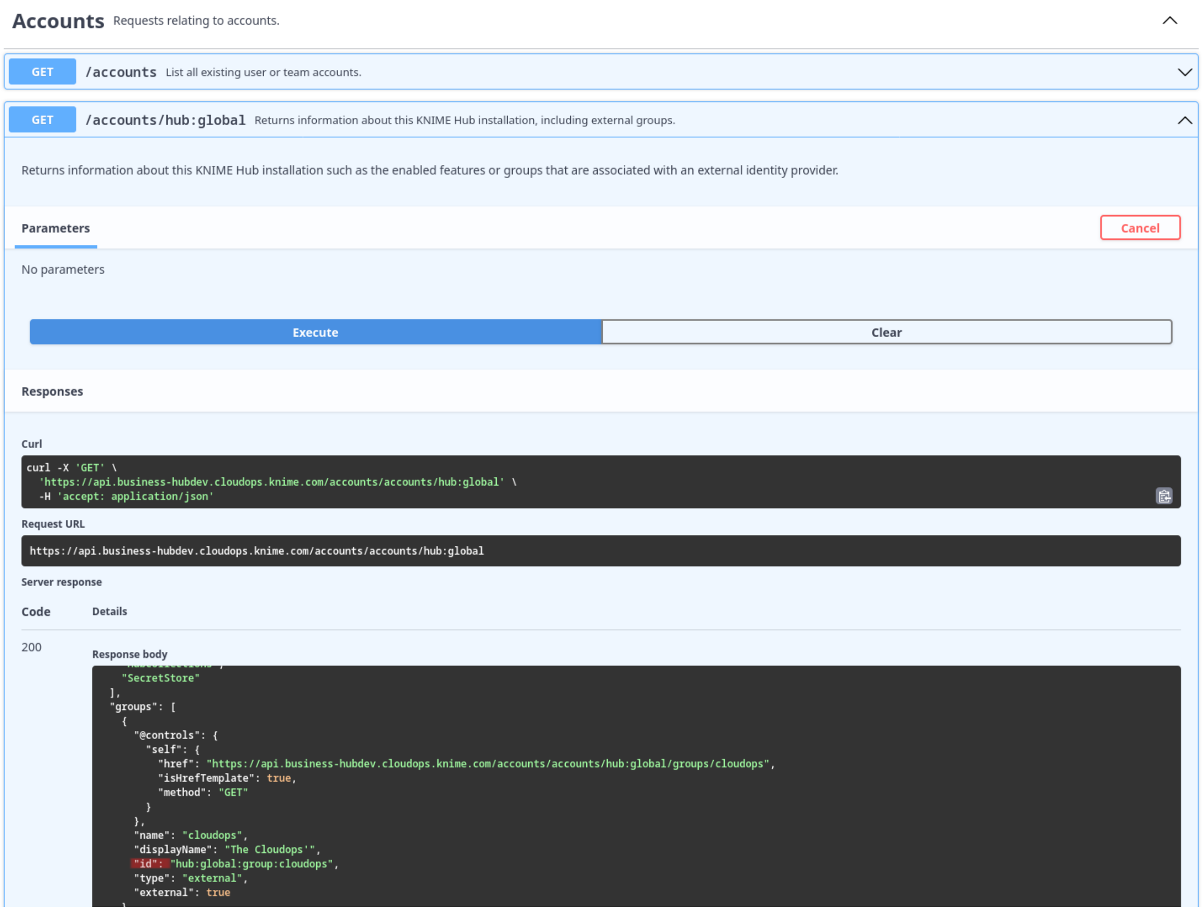
Name	Description
name * required string (path)	The account's name. Dev Team

Server response

Code	Details
200	Response body

```
},
"type": "TEAM",
"id": "account:team:41d128c5-8331-491f-8a1b-d8f5c843e211",
"name": "Dev Team",
```

2. Find the external group’s ID. Since the external group’s name in KNIME Business Hub may not be exactly the same as in the identity provider, the easiest way is to list all external groups, select the right one based on its `displayName`, and note down its `id`.



Accounts Requests relating to accounts.

GET /accounts List all existing user or team accounts.

GET /accounts/hub:global Returns information about this KNIME Hub installation, including external groups.

Returns information about this KNIME Hub installation such as the enabled features or groups that are associated with an external identity provider.

Parameters

No parameters

Responses

Curl

```
curl -X 'GET' \
  'https://api.business-hubdev.cloudops.knime.com/accounts/accounts/hub:global' \
  -H 'accept: application/json'
```

Request URL

```
https://api.business-hubdev.cloudops.knime.com/accounts/accounts/hub:global
```

Server response

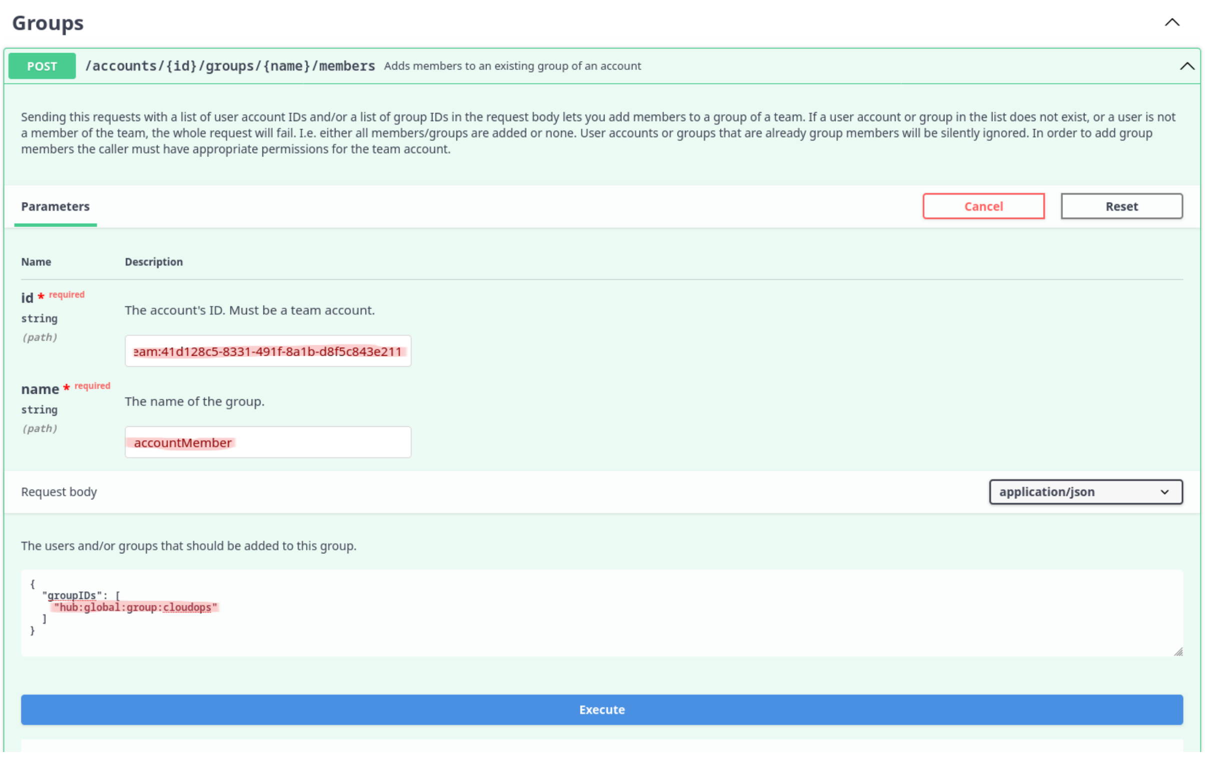
Code Details

200

Response body

```
    "SecretStore"
  ],
  "groups": [
    {
      "@controls": {
        "self": {
          "href": "https://api.business-hubdev.cloudops.knime.com/accounts/accounts/hub:global/groups/cloudops",
          "isHrefTemplate": true,
          "method": "GET"
        }
      },
      "name": "cloudops",
      "displayName": "The Cloudops'",
      "id": "hub:global:group:cloudops",
      "type": "external",
      "external": true
```

3. Add the external group as a member to the *accountMember* group of the desired team.



Groups

POST /accounts/{id}/groups/{name}/members Adds members to an existing group of an account

Sending this requests with a list of user account IDs and/or a list of group IDs in the request body lets you add members to a group of a team. If a user account or group in the list does not exist, or a user is not a member of the team, the whole request will fail. I.e. either all members/groups are added or none. User accounts or groups that are already group members will be silently ignored. In order to add group members the caller must have appropriate permissions for the team account.

Parameters

Name	Description
id * required string *(path)*	The account's ID. Must be a team account. eam:41d128c5-8331-491f-8a1b-d8f5c843e211
name * required string *(path)*	The name of the group. accountMember

Request body: application/json

The users and/or groups that should be added to this group.

```
{
  "groupIDs": [
    "hub:global:group:cloudops"
  ]
}
```

After that, the users in the external group will immediately show up as members of the respective team.

Please note that you must have enough users in your KNIME Business Hub license, otherwise this operation might fail due to unsufficient users. The team size limit will be updated automatically if the external group was provisioned by SCIM.

Assign users based on external groups via Data Application

Before running this application, ensure you have provisioned groups from your identity provider to KNIME Business Hub. Once you have done this, you can use these external groups to manage team memberships.

Data Application specifications

1. The aim of this application is to allow global admins to manage the assignment and deletion of users to teams based on external groups without having to do it manually.
2. Hence, only KNIME Business Hub global admins have permission to run this data application.
3. When using the data application in KNIME Business Hub, user authentication will align with the logged-in user's permissions, eliminating the need to provide an application password.

Data Application execution

1. Download the workflow from KNIME Community Hub.

   You can find the workflow in the KNIME Community Hub.

2. Upload the downloaded workflow to your KNIME Business Hub instance via the web UI.
3. Deploy the workflow as a data application. You can learn more about how to do it here.
4. Afterward, execute the newly created data application deployment.

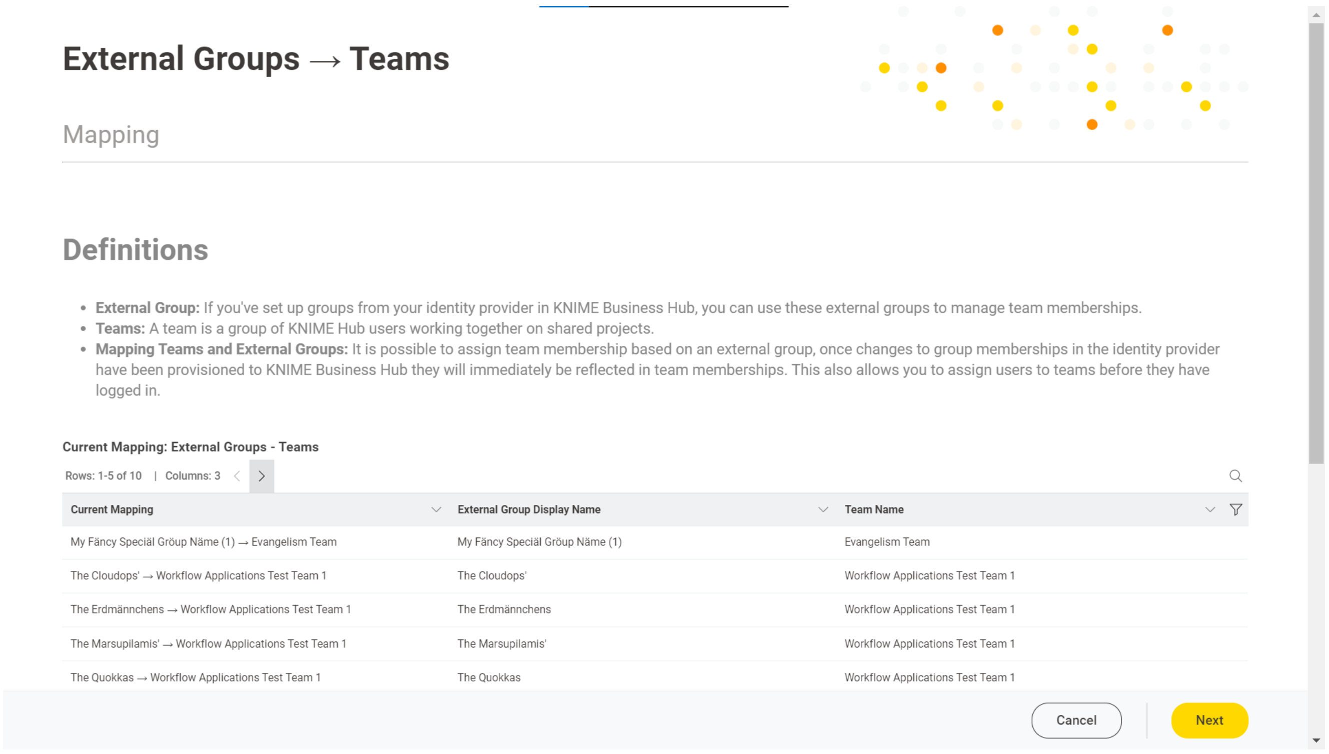
External groups - teams mapping

The one-page data application lets the global admin of KNIME Business Hub view the current mapping of the external groups and teams and make additions or deletions to modify it.

Current mapping

Below the main definitions is a table with three columns indicating the current mapping, the external groups, and the teams.

You can use the filters to narrow your research by team name or external group for convenience.



External Groups → Teams

Mapping

Definitions

- **External Group:** If you've set up groups from your identity provider in KNIME Business Hub, you can use these external groups to manage team memberships.
- **Teams:** A team is a group of KNIME Hub users working together on shared projects.
- **Mapping Teams and External Groups:** It is possible to assign team membership based on an external group, once changes to group memberships in the identity provider have been provisioned to KNIME Business Hub they will immediately be reflected in team memberships. This also allows you to assign users to teams before they have logged in.

Current Mapping: External Groups - Teams

Rows: 1-5 of 10 | Columns: 3

Current Mapping	External Group Display Name	Team Name
My Fäncy Speciäl Gröup Näme (1) → Evangelism Team	My Fäncy Speciäl Gröup Näme (1)	Evangelism Team
The Cloudops' → Workflow Applications Test Team 1	The Cloudops'	Workflow Applications Test Team 1
The Erdmännchens → Workflow Applications Test Team 1	The Erdmännchens	Workflow Applications Test Team 1
The Marsupilamis' → Workflow Applications Test Team 1	The Marsupilamis'	Workflow Applications Test Team 1
The Quokkas → Workflow Applications Test Team 1	The Quokkas	Workflow Applications Test Team 1

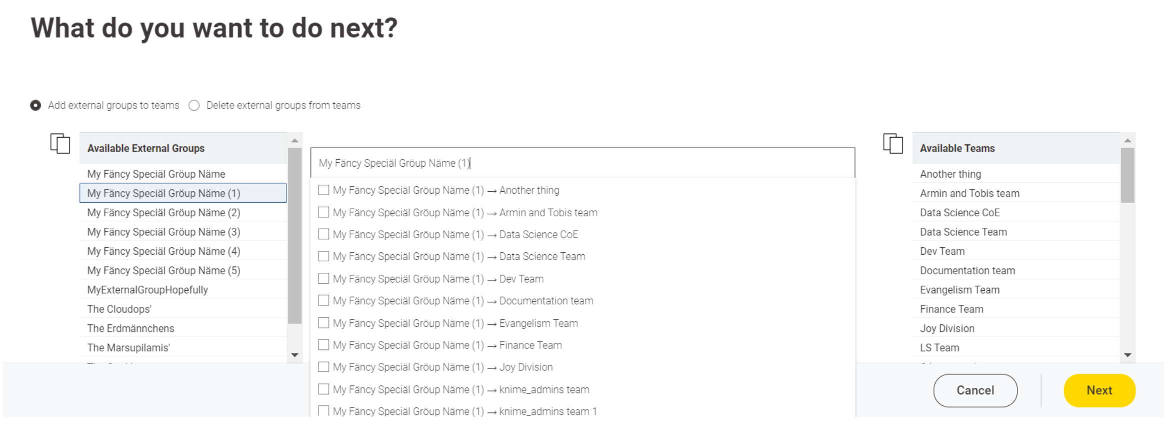
Figure 11. Table with the current external groups assigned to teams.

The table displays just five rows; use the right and left arrows to navigate.

Adding external groups to teams

Once you have a clear understanding of the current external groups assigned to teams, you can add new ones.

1. Choose the "Add external groups to teams" option to complete this task.
   a. The central search box allows you to add the "external groups - teams" pairs.
   b. To ease the research process, if too many options are available, you can **copy** either the *external groups* or the *team* names and paste them into the **search box**. This will display only items related to the pasted name.



What do you want to do next?

(● Add external groups to teams) ○ Delete external groups from teams

Available External Groups
My Fäncy Speciäl Gröup Näme
My Fäncy Speciäl Gröup Näme (1)
My Fäncy Speciäl Gröup Näme (2)
My Fäncy Speciäl Gröup Näme (3)
My Fäncy Speciäl Gröup Näme (4)
My Fäncy Speciäl Gröup Näme (5)
MyExternalGroupHopefully
The Cloudops'
The Erdmännchens
The Marsupilamis'

Search box: My Fäncy Speciäl Gröup Näme (1)

- ☐ My Fäncy Speciäl Gröup Näme (1) → Another thing
- ☐ My Fäncy Speciäl Gröup Näme (1) → Armin and Tobis team
- ☐ My Fäncy Speciäl Gröup Näme (1) → Data Science CoE
- ☐ My Fäncy Speciäl Gröup Näme (1) → Data Science Team
- ☐ My Fäncy Speciäl Gröup Näme (1) → Dev Team
- ☐ My Fäncy Speciäl Gröup Näme (1) → Documentation team
- ☐ My Fäncy Speciäl Gröup Näme (1) → Evangelism Team
- ☐ My Fäncy Speciäl Gröup Näme (1) → Finance Team
- ☐ My Fäncy Speciäl Gröup Näme (1) → Joy Division
- ☐ My Fäncy Speciäl Gröup Näme (1) → knime_admins team
- ☐ My Fäncy Speciäl Gröup Näme (1) → knime_admins team 1

Available Teams
Another thing
Armin and Tobis team
Data Science CoE
Data Science Team
Dev Team
Documentation team
Evangelism Team
Finance Team
Joy Division
LS Team

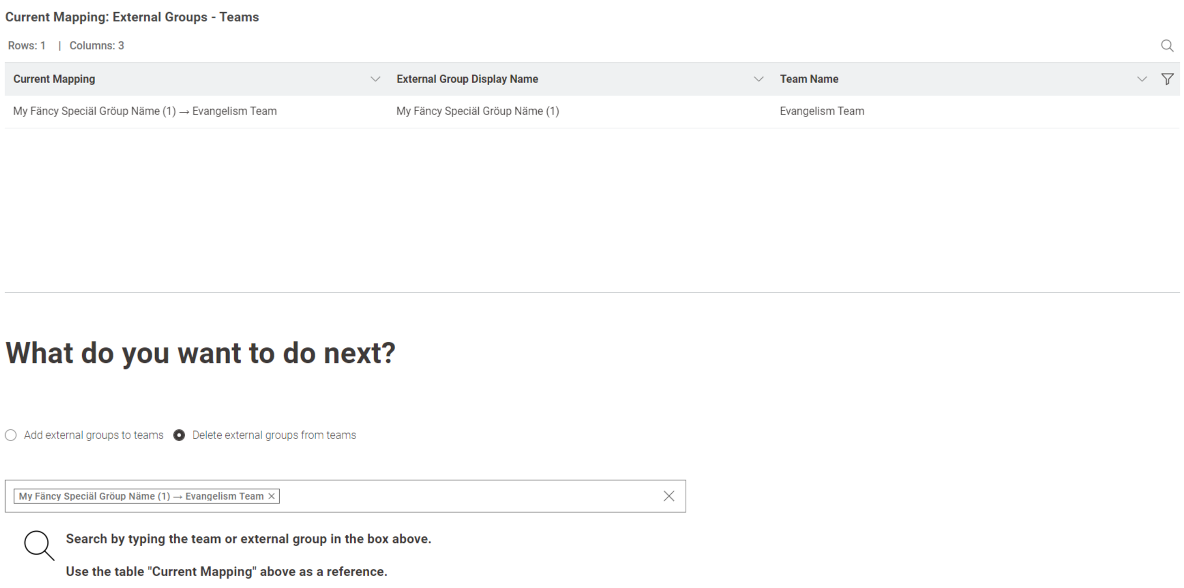
Figure 12. Copy the desired external group name and paste it into the search box.

2. After adding all the desired external groups to the target teams in the search box, click "Next." After a short wait, you will see the same page with the newly added external groups. You can use the table filters to find them.
3. Afterward, the users in the external group will **immediately** appear as members of the respective team.

Delete external groups from teams

To detach an external group from a specific team, you can follow these steps:

1. Select the "Delete external groups from teams" option.
2. Refer to the "Current Mapping: External Groups—Teams" table to determine which external groups are attached to which teams.
3. In the search box, enter the name of the external group or team that you want to delete.


Current Mapping: External Groups - Teams

Rows: 1 | Columns: 3

Current Mapping	External Group Display Name	Team Name
My Fäncy Speciäl Gröup Näme (1) → Evangelism Team	My Fäncy Speciäl Gröup Näme (1)	Evangelism Team

What do you want to do next?

Add external groups to teams [✓ Delete external groups from teams]

My Fäncy Speciäl Gröup Näme (1) → Evangelism Team

Search by typing the team or external group in the box above.

Use the table "Current Mapping" above as a reference.

Figure 13. The external group that will be deleted from the team is added to the search box.

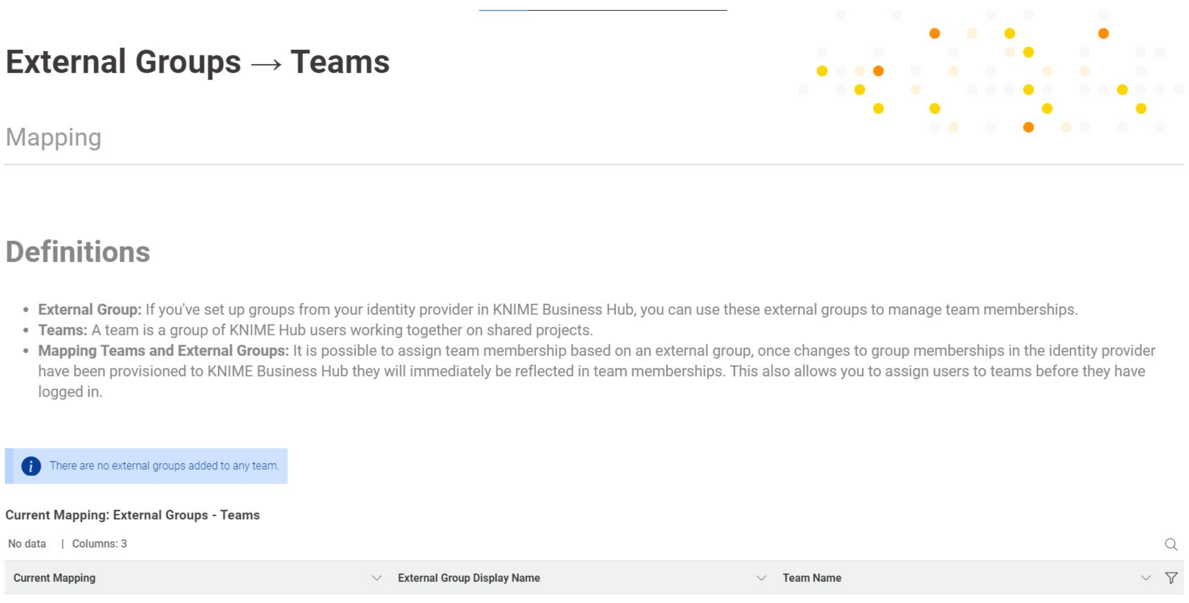
4. Once you have added all the external groups and their corresponding teams in the search box, click "Next."

After clicking "Next", all the users in the external group will **immediately** be removed as members of the respective team.

Warnings and errors

The data application may encounter various errors or issues during execution.

1. If the KNIME Business Hub API responds poorly, the data application has trouble communicating with it via REST requests. If this happens, a clear error message should be displayed to inform the user.
2. If external groups have yet to be provided for the current KNIME Business Hub, you must follow all the steps provided above before running the data application.
3. When running the data application for the first time, an information message indicates that no external groups are assigned to any team. To add them, use the search box and click "Next".



External Groups → Teams

Mapping

Definitions

- **External Group:** If you've set up groups from your identity provider in KNIME Business Hub, you can use these external groups to manage team memberships.
- **Teams:** A team is a group of KNIME Hub users working together on shared projects.
- **Mapping Teams and External Groups:** It is possible to assign team membership based on an external group, once changes to group memberships in the identity provider have been provisioned to KNIME Business Hub they will immediately be reflected in team memberships. This also allows you to assign users to teams before they have logged in.

There are no external groups added to any team.

Current Mapping: External Groups - Teams

No data | Columns: 3

Current Mapping	External Group Display Name	Team Name

Figure 14. There are no external groups associated with teams.

4. Suppose a KNIME Business Hub team admin or member tries to run the data application. In that case, an error message will appear, telling the user that only global admins can execute the application.
5. If you proceed without selecting any item in the search box and click "Next", a warning message will be displayed, prompting you to return to the external groups teams mapping stage by clicking "Next" again.

Expose external groups inside KNIME Business Hub (Standard and Enterprise editions only)

In case you cannot use SCIM, there are two other possibilities for bringing external groups into your KNIME Business Hub installation. As a Global KNIME Hub administrator you can configure groups that are provided via an external identity provider to be exposed inside the KNIME Business Hub instance.

Two possible sources for your external groups are:

1. Groups are provided within the access token of your OIDC provider
2. Groups are imported from LDAP by federating the login

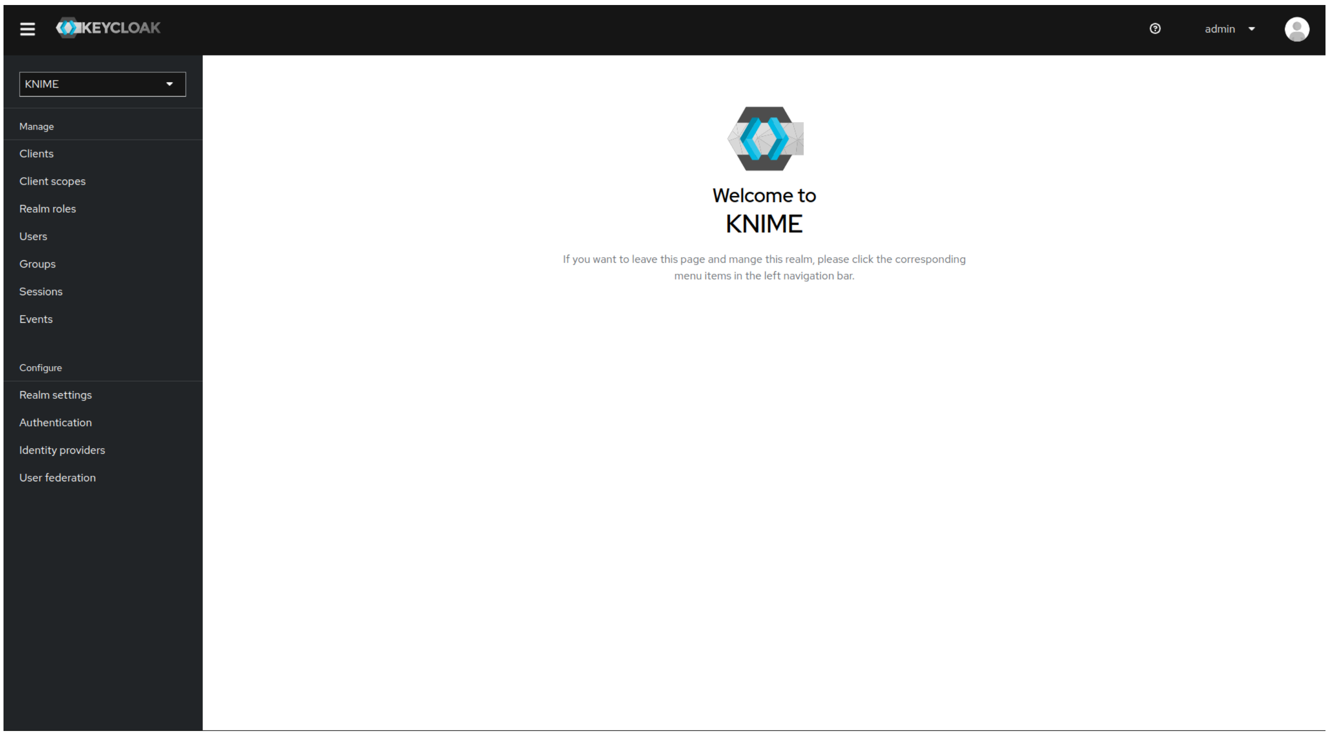
External OIDC provider

Assume you have an identity provider that provides groups through a **groups** claim in the access token.

```
{
    ...,
    "groups": [
        "finance",
        "marketing",
        "data"
    ]
}
```

First you need to configure Keycloak in such a way that it can map these groups to a **user attribute**. The second step is to add a mapper that maps these user attributes into the Keycloak’s tokens.

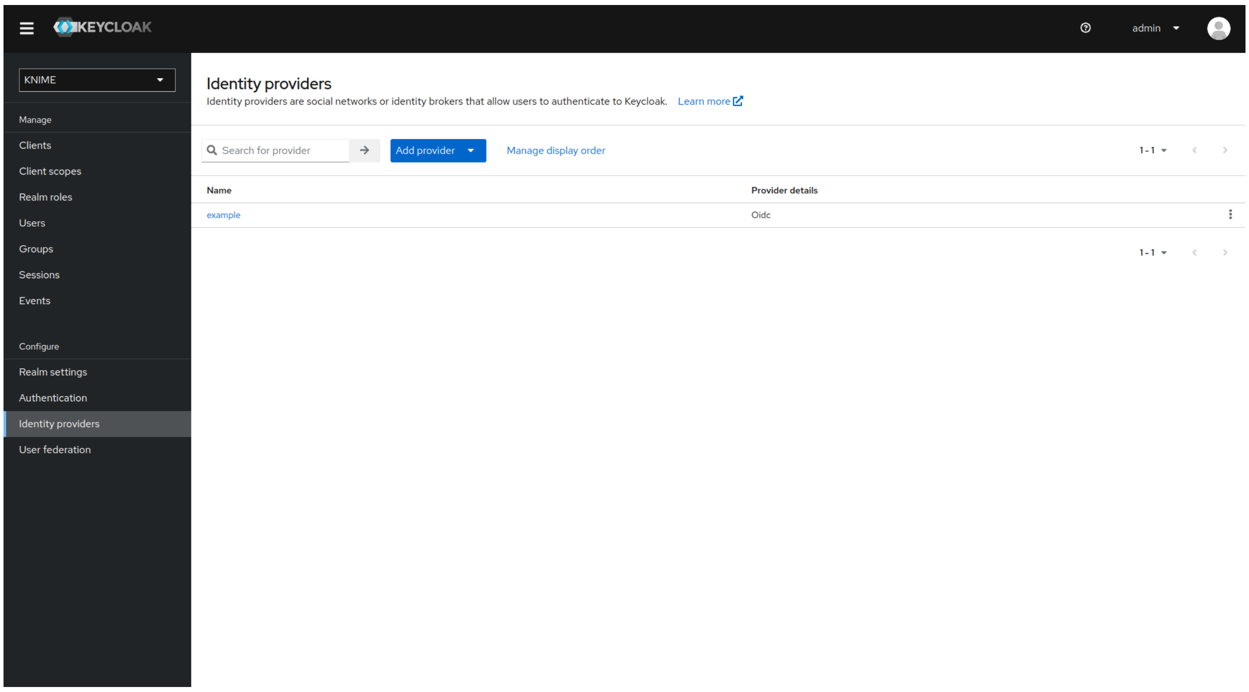
Your third-party identity provider should have been set up already. Keycloak has to be configured as follows:



If you want to leave this page and mange this realm, please click the corresponding menu items in the left navigation bar.

First step is to add an Attribute Importer Mapper.

1. In Keycloak select realm *Knime* in the top left dropdown menu
2. On the left tab select *Identity Providers*
3. Select your third-party provider
4. Switch to the tab *Mappers* and click on *Add mapper*
5. Provide a name for the mapper and set the *Sync mode override* to *Force* to ensure that the user’s group memberships are updated upon every login
6. Set *Mapper type* to *Attribute importer*
7. Enter the *Claim* that contains the external groups in the original token (in our example *groups*)
8. In the *User Attribute Name* field enter *external-groups*
9. Click on *Save*

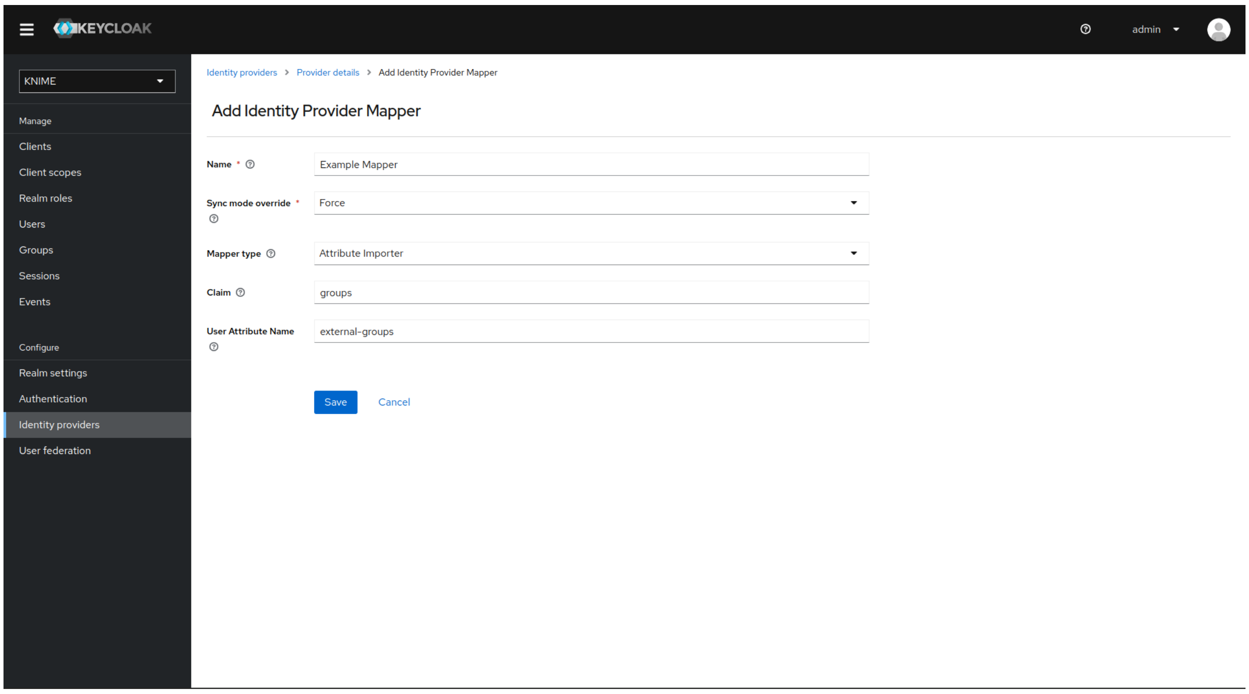


Identity providers

Identity providers are social networks or identity brokers that allow users to authenticate to Keycloak. Learn more

Search for provider | Add provider | Manage display order | 1 - 1

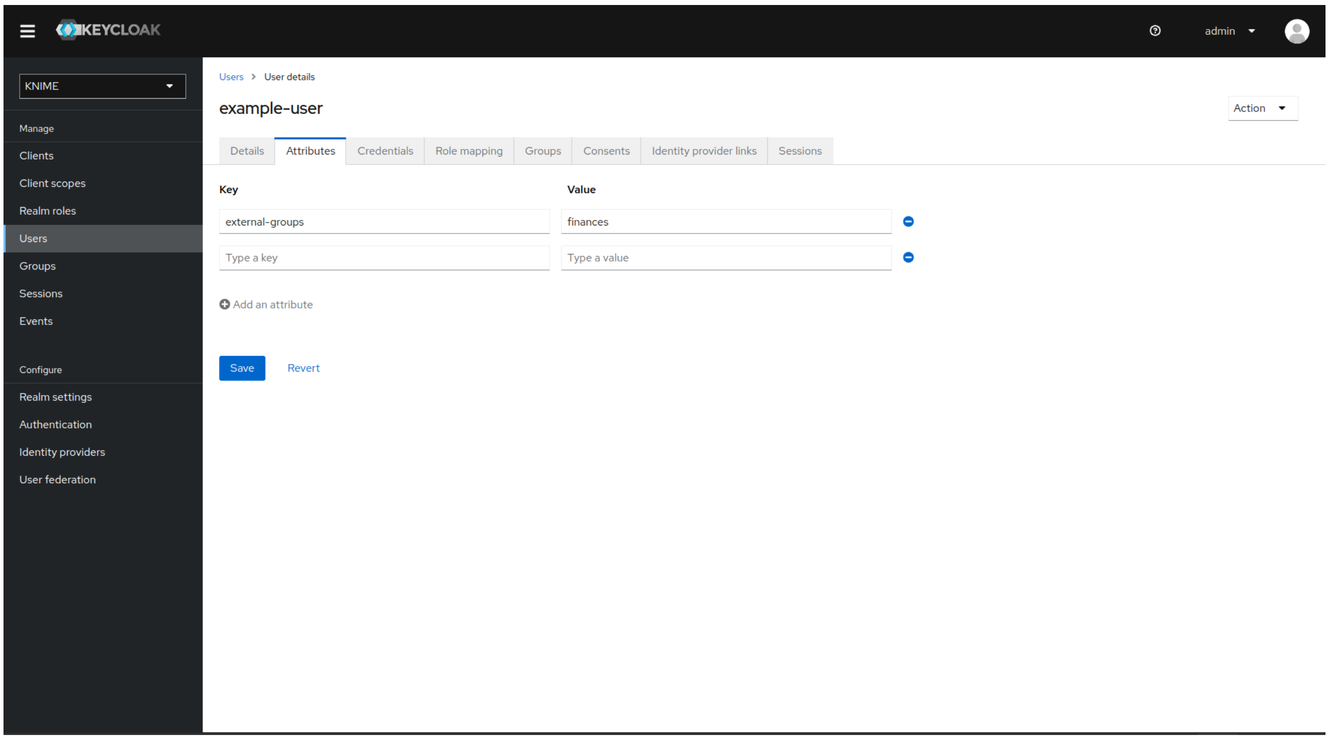
Name	Provider details
example	Oidc



Add Identity Provider Mapper

Name *	Example Mapper
Sync mode override *	Force
Mapper type	Attribute Importer
Claim	groups
User Attribute Name	external-groups

Now, every user in Keycloak who logged in after the mapper has been added will have an *external-groups* attribute associated like in the following picture:



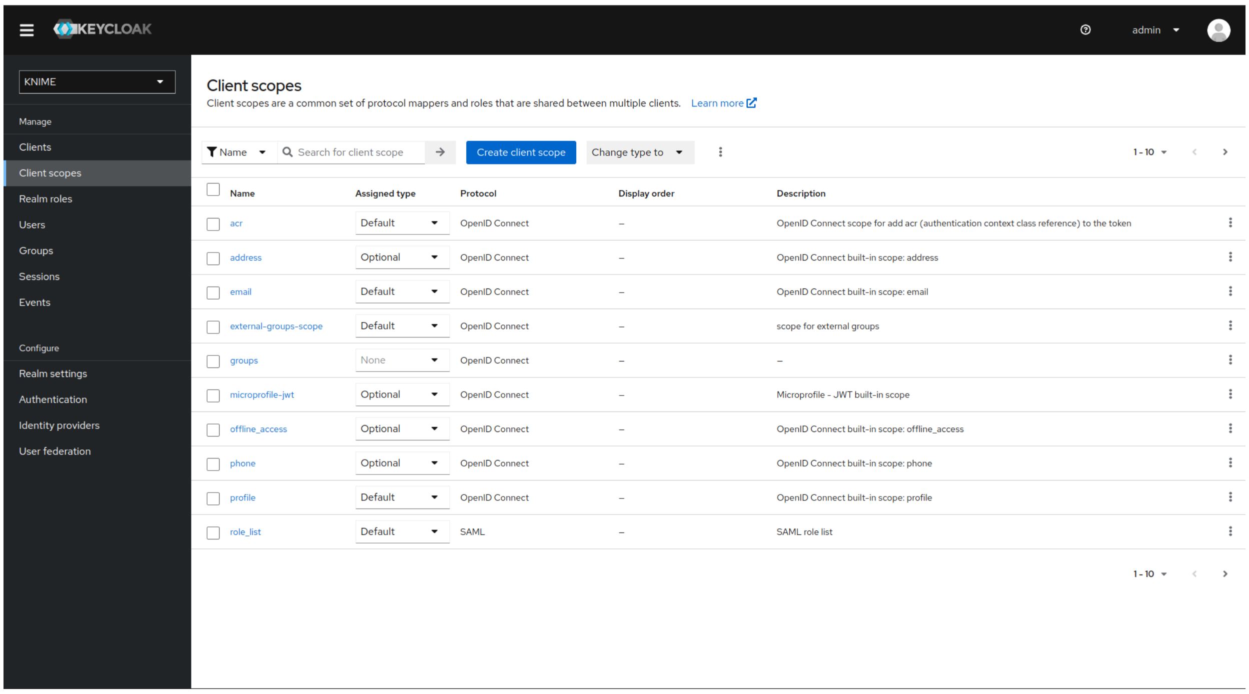
example-user

Key	Value
external-groups	finances
Type a key	Type a value

Now, the external groups are known to Keycloak. To expose them inside KNIME Business Hub they need to be mapped into the access tokens issued by Keycloak. For this a second mapper needs to be added, that maps the user attribute *external-groups* to a claim in the user's access token.

To do this you need to add a client scope, which includes a mapper for the user attribute.

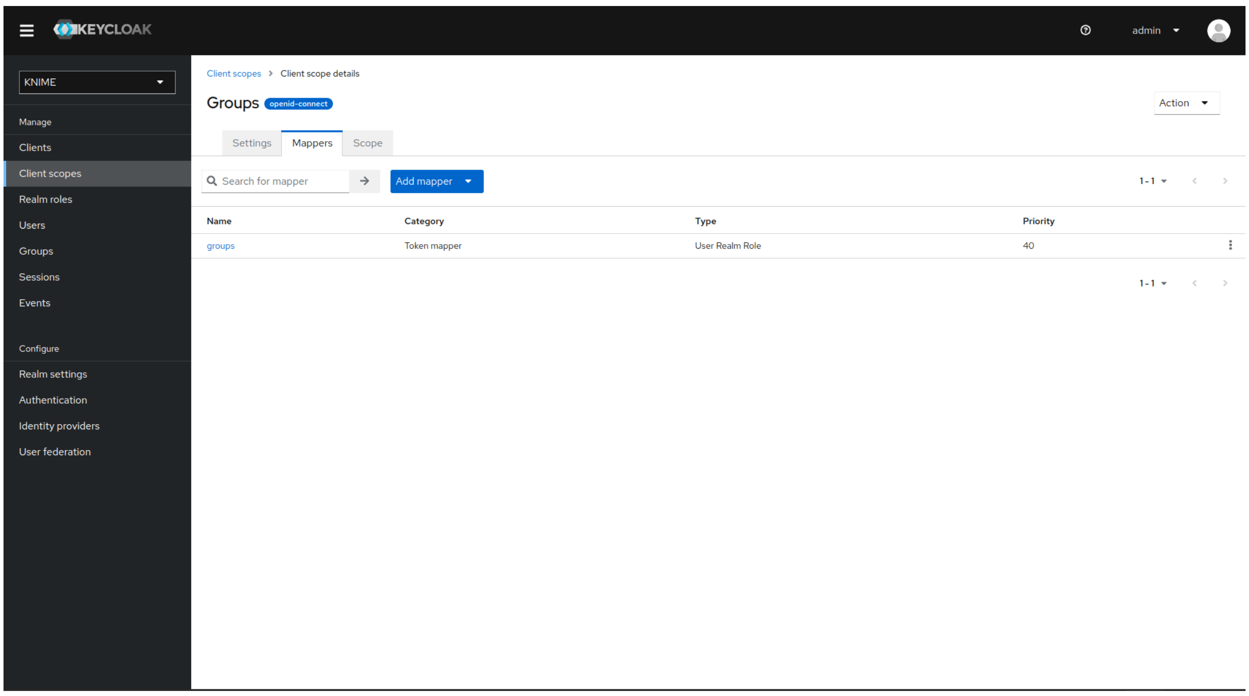
1. On the left tab select *Client scopes*
2. Select *groups*
3. Switch to the tab *Mappers*
4. Click on *Add mapper > By configuration* and select *User Attribute* from the list
5. Provide a name, e.g. *external-groups-attribute-mapper*
6. Set both fields *User Attribute* and *Token Claim Name* to *external-groups*
7. Ensure that *Add to ID token*, *Add to access token* and *Aggregate attribute values* are turned off
8. Ensure that *Add to userinfo* and *Multivalued* are turned on
9. Click on *Save*



Client scopes

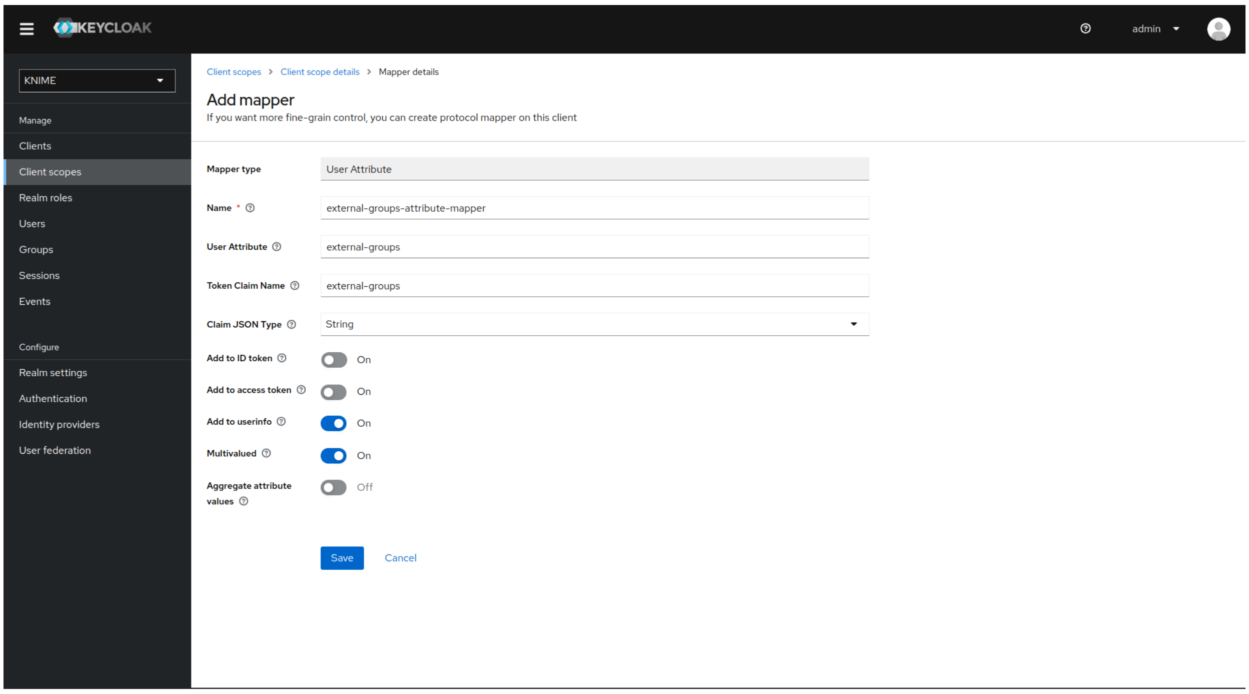
Client scopes are a common set of protocol mappers and roles that are shared between multiple clients. Learn more

Name	Assigned type	Protocol	Display order	Description
acr	Default	OpenID Connect	–	OpenID Connect scope for add acr (authentication context class reference) to the token
address	Optional	OpenID Connect	–	OpenID Connect built-in scope: address
email	Default	OpenID Connect	–	OpenID Connect built-in scope: email
external-groups-scope	Default	OpenID Connect	–	scope for external groups
groups	None	OpenID Connect	–	–
microprofile-jwt	Optional	OpenID Connect	–	Microprofile - JWT built-in scope
offline_access	Optional	OpenID Connect	–	OpenID Connect built-in scope: offline_access
phone	Optional	OpenID Connect	–	OpenID Connect built-in scope: phone
profile	Default	OpenID Connect	–	OpenID Connect built-in scope: profile
role_list	Default	SAML	–	SAML role list



Groups openid-connect

Name	Category	Type	Priority
groups	Token mapper	User Realm Role	40



Add mapper

If you want more fine-grain control, you can create protocol mapper on this client

Mapper type	User Attribute
Name *	external-groups-attribute-mapper
User Attribute	external-groups
Token Claim Name	external-groups
Claim JSON Type	String
Add to ID token	On
Add to access token	On
Add to userinfo	On
Multivalued	On
Aggregate attribute values	Off

With both mappers in place, the external groups are part of userinfo response returned by Keycloak. By this, the external groups are exposed inside KNIME Business Hub. In order to enable external groups to be used for permissions and access management they need to be configured separately through the admin REST API as described in Enable external groups.

LDAP federation

Before proceeding with the following steps you need to have a user federation configured for an LDAP instance.

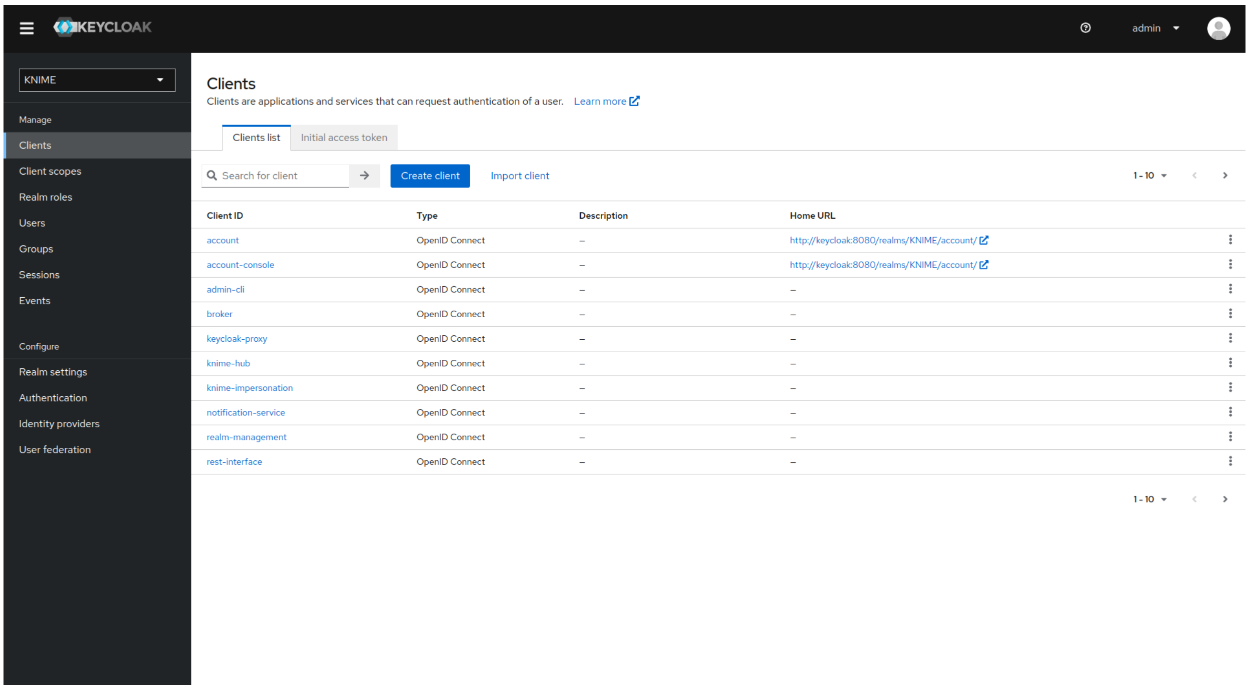
Once you have configured user federation for an LDAP instance that also supplies external group names, you need to configure mappers that map these groups into the access tokens used inside the Hub instance.

To ensure that groups from Keycloak groups and groups from LDAP are not mixed we recommend to treat external groups as realm roles.

In order to do this we recommend to first create a dummy client for which roles can be created based on the LDAP groups. This will guarantee that any changes will be compatible with future changes to the KNIME Hub client in Keycloak.

To create a new client follow these steps:

1. In Keycloak select realm *Knime* in the top left dropdown menu
2. On the left tab select *Clients* and click *Create client*



KNIME

Manage
Clients
Client scopes
Realm roles
Users
Groups
Sessions
Events

Configure
Realm settings
Authentication
Identity providers
User federation

Clients

Clients are applications and services that can request authentication of a user. Learn more

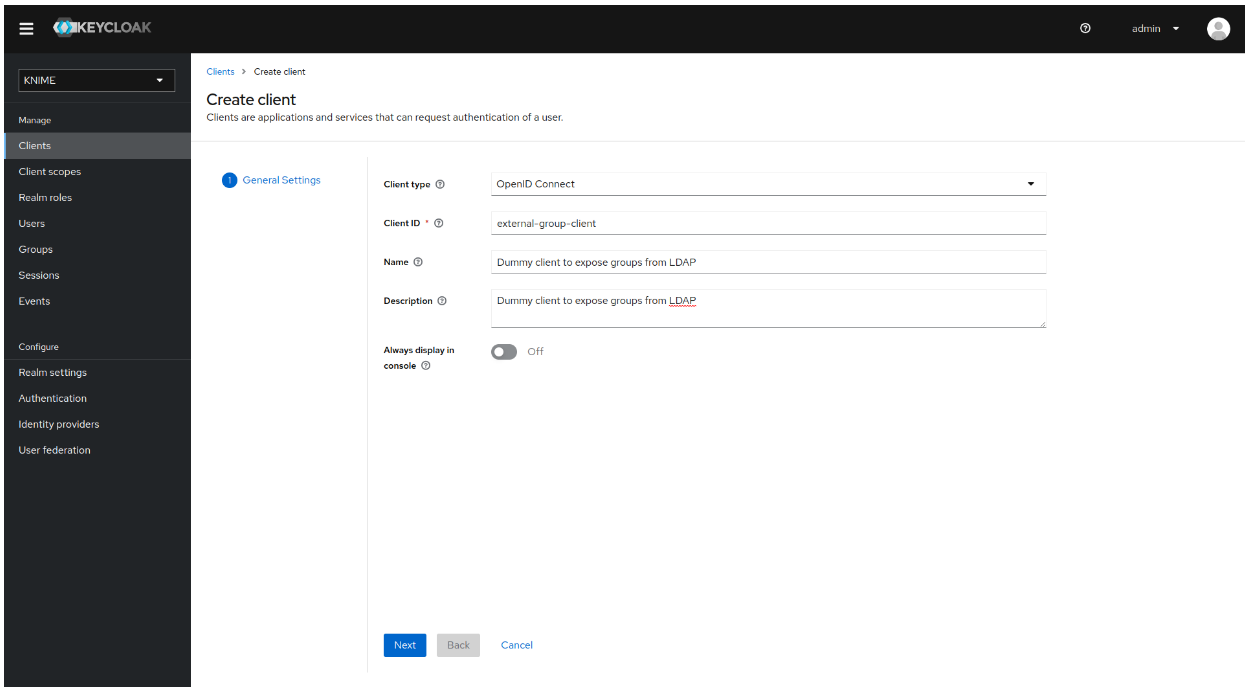
Clients list | Initial access token

Search for client | Create client | Import client | 1 - 10

Client ID	Type	Description	Home URL
account	OpenID Connect	–	http://keycloak:8080/realms/KNIME/account/
account-console	OpenID Connect	–	http://keycloak:8080/realms/KNIME/account/
admin-cli	OpenID Connect	–	–
broker	OpenID Connect	–	–
keycloak-proxy	OpenID Connect	–	–
knime-hub	OpenID Connect	–	–
knime-impersonation	OpenID Connect	–	–
notification-service	OpenID Connect	–	–
realm-management	OpenID Connect	–	–
rest-interface	OpenID Connect	–	–

1 - 10

3. Set *Client type* to *OpenID Connect*
4. Enter a *Client ID* (in our example *external-group-client*), and a useful *Name* and *Description*
5. Click on *Next*



KNIME

Manage
Clients
Client scopes
Realm roles
Users
Groups
Sessions
Events

Configure
Realm settings
Authentication
Identity providers
User federation

Clients > Create client

Create client

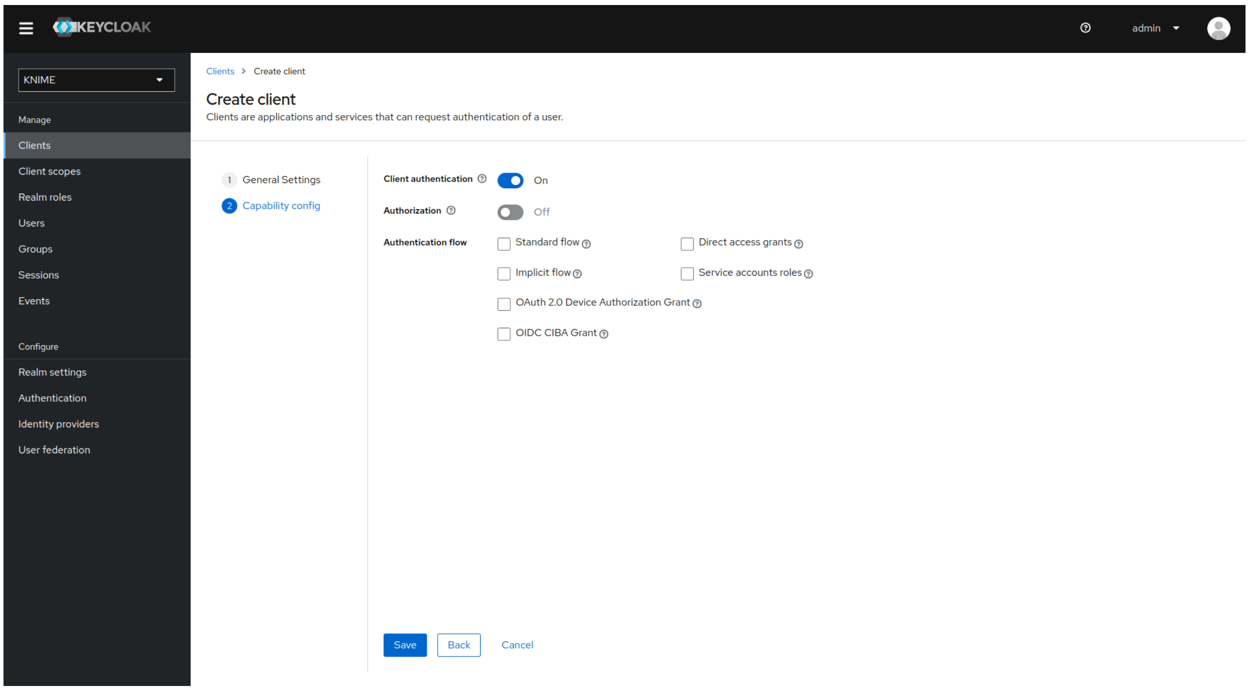
Clients are applications and services that can request authentication of a user.

1 General Settings

Client type	OpenID Connect
Client ID *	external-group-client
Name	Dummy client to expose groups from LDAP
Description	Dummy client to expose groups from LDAP
Always display in console	Off

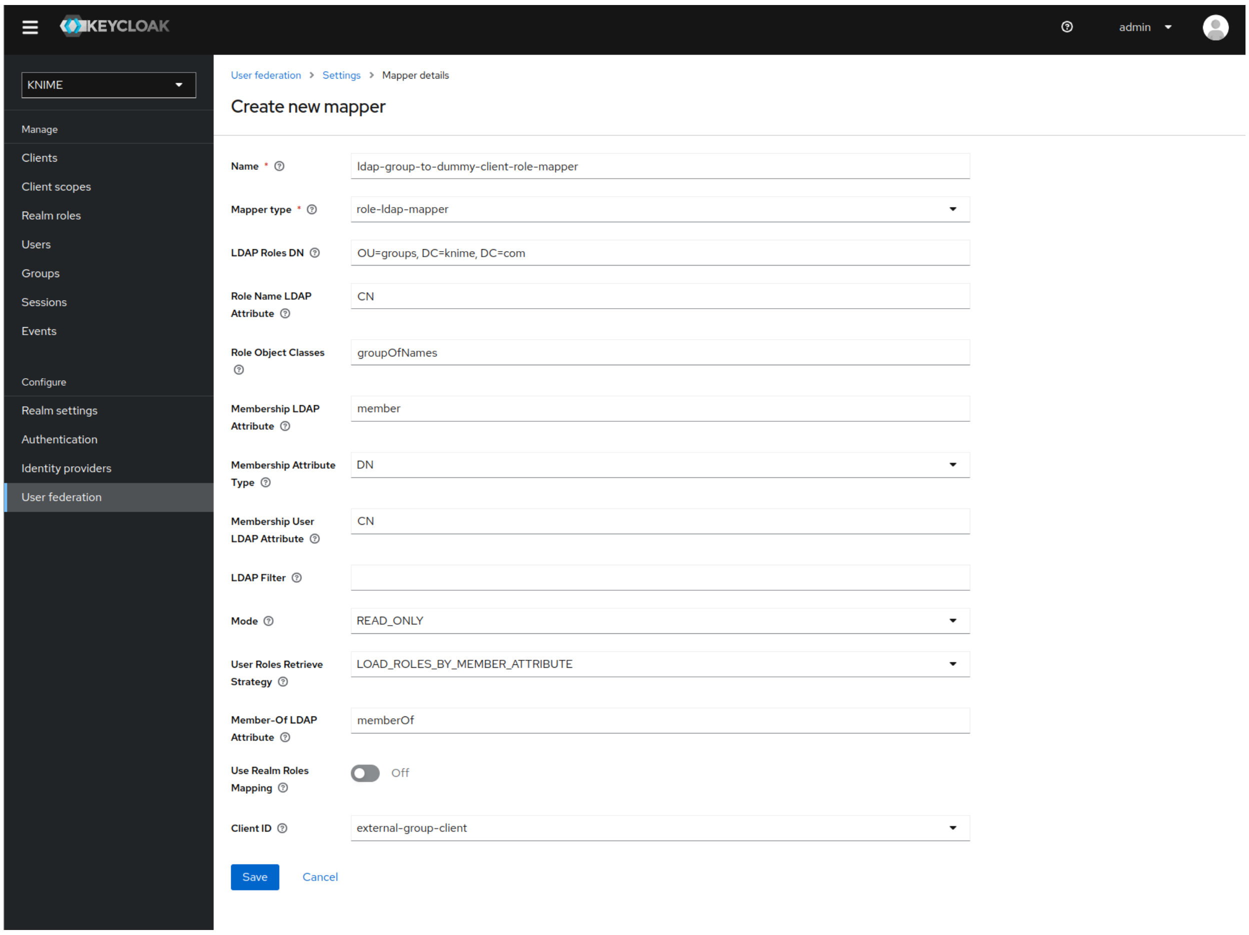
Next | Back | Cancel

6. De-select all checkboxes of *Authentication flow* in the *Capability config* section, since this client will not require any capabilities
7. Enable *Client authentication*
8. Click on *Save*



Now that the dummy client is set up, you can proceed to create a mapper that maps the user groups from LDAP to roles inside the dummy client:

1. On the left tab select *User federation* and click on your LDAP configuration
2. Switch to the tab *Mappers*
3. Click on *Add mapper*
4. Provide a name, e.g. *ldap-group-to-dummy-client-role-mapper*
5. Set *Mapper type* to role-ldap-mapper
6. Setup the mapper according to your LDAP
7. Disable *User Realm Roles Mapping*
8. Set *Client ID* to the previously created dummy client (in our example *external-group-client*)
9. Click on *Save*



Create new mapper

Name *: ldap-group-to-dummy-client-role-mapper

Mapper type *: role-ldap-mapper

LDAP Roles DN: OU=groups, DC=knime, DC=com

Role Name LDAP Attribute: CN

Role Object Classes: groupOfNames

Membership LDAP Attribute: member

Membership Attribute Type: DN

Membership User LDAP Attribute: CN

LDAP Filter:

Mode: READ_ONLY

User Roles Retrieve Strategy: LOAD_ROLES_BY_MEMBER_ATTRIBUTE

Member-Of LDAP Attribute: memberOf

Use Realm Roles Mapping: Off

Client ID: external-group-client

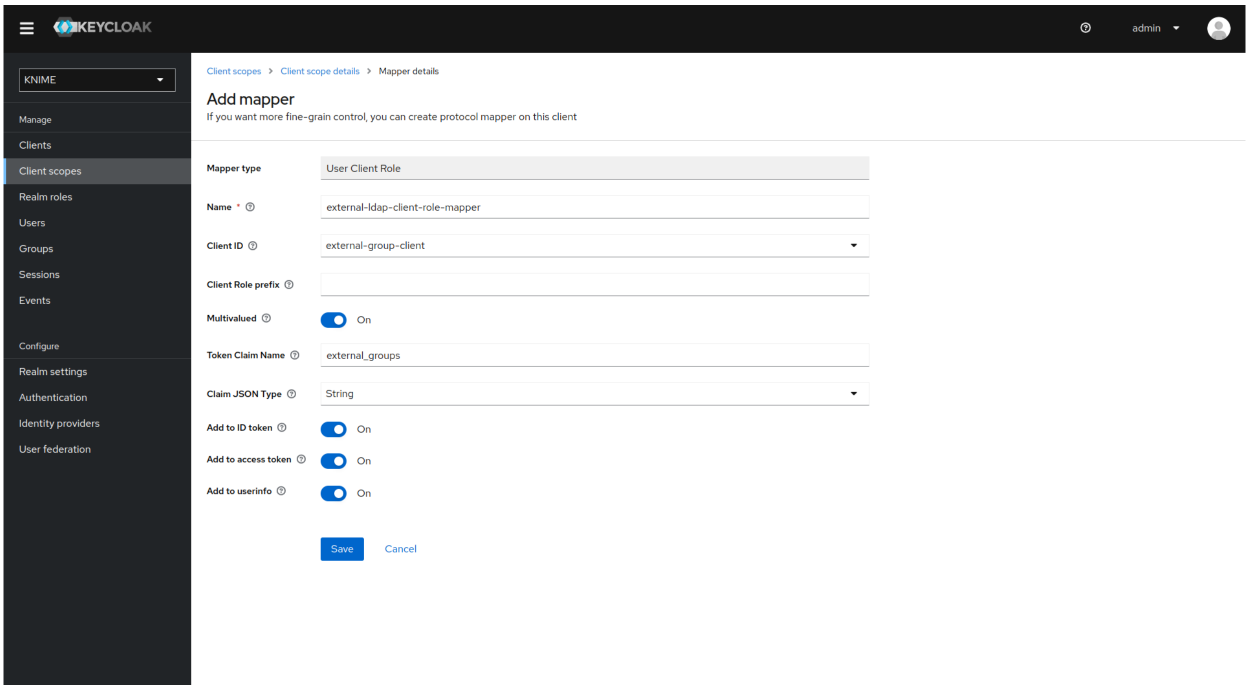
Save Cancel

Now if a user logs in with the LDAP credentials the user's groups will be mapped to ad-hoc created client roles inside the *'external-group-client'*.

Next, you need to create a mapper that maps a user's realm roles from the dummy realm to the access tokens:

1. On the left tab select *Client scopes*
2. Select *groups*
3. Switch to the tab *Mappers*
4. Click on *Add mapper > By configuration* and select *User Client Role* from the list
5. Provide a name, e.g. *external-ldap-client-role-mapper*
6. Set *Client ID* to the previously created dummy client (in our example *external-group-client*)
7. Set *Token Claim Name* to *external-groups*
8. Set *Claim JSON Type* to *String*
9. Ensure that *Add to ID token*, *Add to access token*, *Add to userinfo*, and *Multivalued* are turned on

10. Click on *Save*



Add mapper

If you want more fine-grain control, you can create protocol mapper on this client

Mapper type	User Client Role
Name *	external-ldap-client-role-mapper
Client ID	external-group-client
Client Role prefix	
Multivalued	On
Token Claim Name	external_groups
Claim JSON Type	String
Add to ID token	On
Add to access token	On
Add to userinfo	On

Enable external groups

Once you have configured the external groups in Keycloak you need to create the groups that you want to be available inside KNIME Business Hub.

To do so you have to make a `PUT` request to the corresponding endpoint:

```
PUT https://api.<base-url>/accounts/hub:global/groups/<external-group-name>
```

where <external-group-name> is the name of the group and it must match the group name in the external identity provider.

You can use the following JSON type body in order to set a display name for the group to be shown on KNIME Business Hub.

```
{
  "displayName": "My Group Display Name",
  "external": true
}
```

Docker executor images

KNIME release types: LTS vs standard releases

KNIME provides two types of releases with different update schedules and support lifecycles:

LTS (Long-Term Support) releases:

- Released every 6 months
- Supported for 2 years with bug fixes and security updates
- Do not receive new features during the support period
- Recommended for production environments that should not be updated too frequently

Standard releases:

- Released every 6 weeks as a rolling release
- Include the latest features, improvements, bug fixes, and security updates
- Supported only until the next release (approximately 6 weeks)
- Each release is production-quality and fully tested
- Recommended for users who want access to the latest features and can update regularly

For KNIME Business Hub executor images, **most customers probably want to use LTS releases** to reduce maintenance overhead. Thus, this guide primarily focuses on LTS images.

For more information about KNIME's release strategy, see the KNIME Analytics Platform Release FAQ.

Available Docker executor images

In order to create execution contexts for their teams, team admins will need to select the KNIME Executor that they want to use. They can:

- Select it from a drop-down in the *Create execution context* panel, if images are registered as explained below.
- Add the Docker image name in the corresponding field in the *Create execution context* panel, if no image has been registered.

In the first case, further actions need to be taken to populate this panel.

You can add:

- One of the Docker executor images made available by KNIME.
  - You can do this using the "Manage List of Executor Images" data application. Follow the instructions in the next section to know how to use the data application.
- A customized Docker executor image, e.g. by adding specific extensions or default conda environments.
  - You can do this using a data application or manually. Follow the instructions in the Add extensions to an existing Docker image section to do so.

The following Docker executor images are made available by KNIME.

You can have access to:

- Base images of KNIME Executor versions 4.7.4 and higher (see here)
- Full images of KNIME Executor versions 4.7.4 and higher (see here)

LTS (Long-Term Support) images

Here is a list of the currently available **LTS full** images (recommended for most customers):

- `knime/knime-lts-full:r-4.7.4-179`
- `knime/knime-lts-full:r-4.7.5-199`
- `knime/knime-lts-full:r-4.7.6-209`
- `knime/knime-lts-full:r-4.7.7-221`
- `knime/knime-lts-full:r-4.7.8-231`
- `knime/knime-lts-full:r-5.1.0-251`
- `knime/knime-lts-full:r-5.1.1-379`
- `knime/knime-lts-full:r-5.1.2-433`
- `knime/knime-lts-full:r-5.1.3-594`
- `knime/knime-lts-full:r-5.2.0-271`
- `knime/knime-lts-full:r-5.2.1-369`
- `knime/knime-lts-full:r-5.2.2-445`
- `knime/knime-lts-full:r-5.2.3-477`

- `knime/knime-lts-full:r-5.2.4-564`
- `knime/knime-lts-full:r-5.2.5-593`
- `knime/knime-lts-full:r-5.2.6-758`
- `knime/knime-lts-full:r-5.3.1-498`
- `knime/knime-lts-full:r-5.3.2-564`
- `knime/knime-lts-full:r-5.3.3-666`
- `knime/knime-lts-full:r-5.4.0-62`
- `knime/knime-lts-full:r-5.4.1-169`
- `knime/knime-lts-full:r-5.4.2-185`
- `knime/knime-lts-full:r-5.4.3-218`
- `knime/knime-lts-full:r-5.4.4-316`
- `knime/knime-lts-full:r-5.4.5-416`
- `knime/knime-lts-full:r-5.5.0-84`
- `knime/knime-lts-full:r-5.5.1-154`

Standard release images

Standard releases are updated every 6 weeks and contain the latest features. They are supported only until the next release. Consider using these only if you need the latest features and can update frequently.

Here is a list of currently available **standard release full** images:

- `knime/knime-full:r-5.7.0-345`

In order to have access to the execution context logs for debugging purposes the execution context needs to be based on the following executor Docker images:

- `knime/knime-lts-full:r-5.2.6-758` or higher bugfix releases of the 5.2.x release line
- `knime/knime-lts-full:r-5.3.2-564` or higher bugfix releases of the 5.3.x release line
- Any new major version released (e.g. 5.4) of the executor will also support this feature

Please note that with the release of 5.2.2 KNIME executors will have HTML sanitization of old JavaScript View nodes and Widget nodes turned on by default. This ensures that no malicious HTML can be output. For more information check the KNIME Analytics Platform 5.2.2 changelog.

Manage KNIME Executor images available on the registry

We provide the "Manage List of Executor Images" data application that will help you managing KNIME Executor images as they appear in the list when configuring an execution context.

You can choose one of these three actions:

- Make new images available in the entries list.
- Modify existing images already present in the entries list.
- Delete images from the available entries.

Removing images from the list does not free up disk space, since this list is independent of the registry where images are stored.

Each entry consists of a name, a description, the URL to the image and the scope, that determines to which team this entry is available.

The image URL can be set to point to an image that is hosted by KNIME. You can select the executor version, and whether it should be a minimal image with only the base extensions installed, or a full image with all trusted extensions installed.

Alternatively, you can provide your own images, e.g. built with the Executor Image Builder data app, and set the URL to the location where these images are hosted.

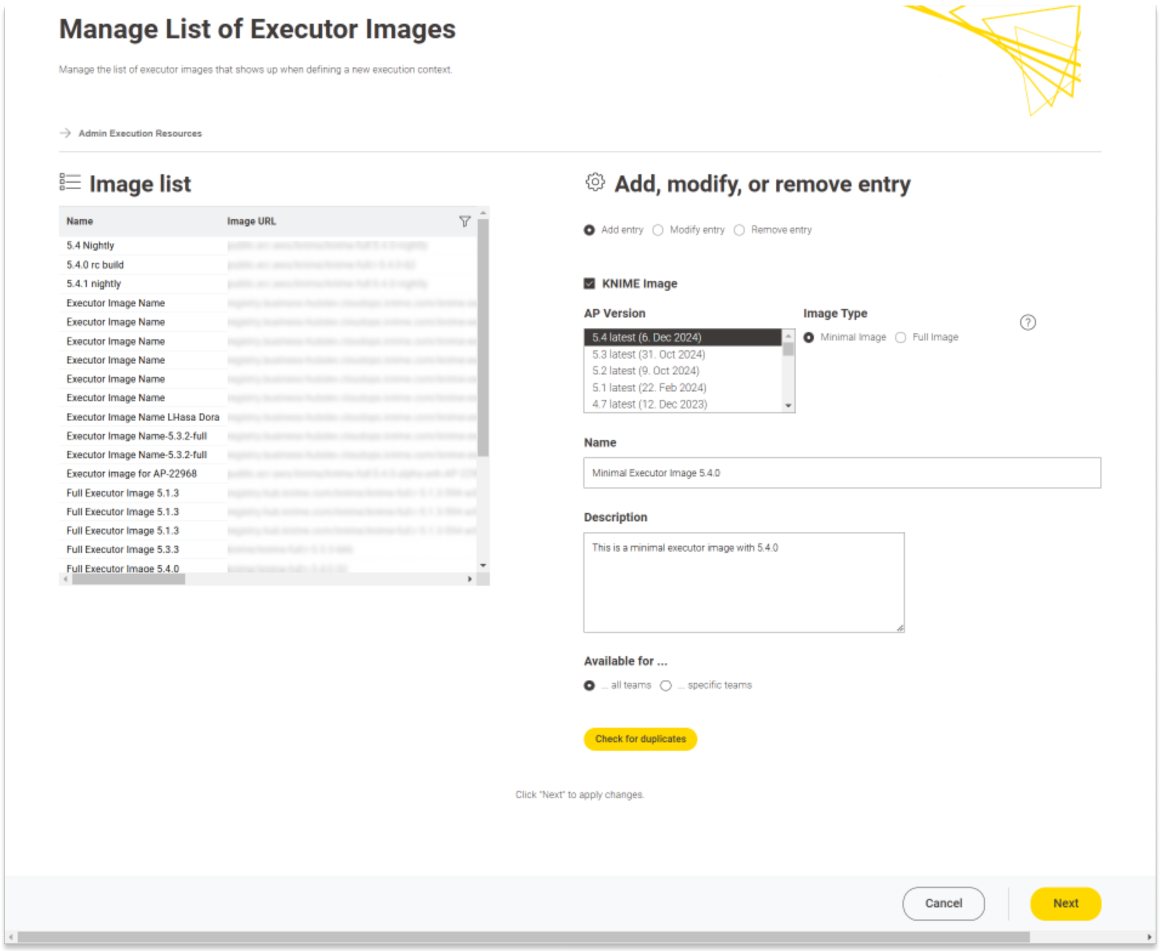
Download the workflow and upload it to your KNIME Business Hub, then *Run* it and follow the instructions on the screen.

On the left side of the data application you will see a list of the existing images.

> The images highlighted in red are the duplicates, meaning there are two or more images with the same name or based on the same Docker image.

Add entry

- Select *Add entry* to add a new executor image to the list of available executors. You can select:
  - KNIME Image to select an image from the Public Registry.
    - You can then choose if you want to add a *Full Image* or a *Minimal image*, and then select the executor version accordingly from the list.
    - You can give the entry a name, a description, and select whether you want to make it available for all teams or for specific teams only.



Manage List of Executor Images

Manage the list of executor images that shows up when defining a new execution context.

→ Admin Execution Resources

Image list

Name	Image URL
5.4 Nightly	[illegible]
5.4.0 rc build	[illegible]
5.4.1 nightly	[illegible]
Executor Image Name	[illegible]
Executor Image Name	[illegible]
Executor Image Name	[illegible]
Executor Image Name	[illegible]
Executor Image Name	[illegible]
Executor Image Name	[illegible]
Executor Image Name LHasa Dora	[illegible]
Executor Image Name-5.3.2-full	[illegible]
Executor Image Name-5.3.2-full	[illegible]
Executor image for AP-22968	[illegible]
Full Executor Image 5.1.3	[illegible]
Full Executor Image 5.1.3	[illegible]
Full Executor Image 5.1.3	[illegible]
Full Executor Image 5.3.3	[illegible]
Full Executor Image 5.4.0	[illegible]

Add, modify, or remove entry

[✓ Add entry] Modify entry Remove entry

[✓] KNIME Image

AP Version: 5.4 latest (6. Dec 2024) (selected); 5.3 latest (31. Oct 2024); 5.2 latest (9. Oct 2024); 5.1 latest (22. Feb 2024); 4.7 latest (12. Dec 2023)

Image Type: [✓ Minimal Image] Full Image

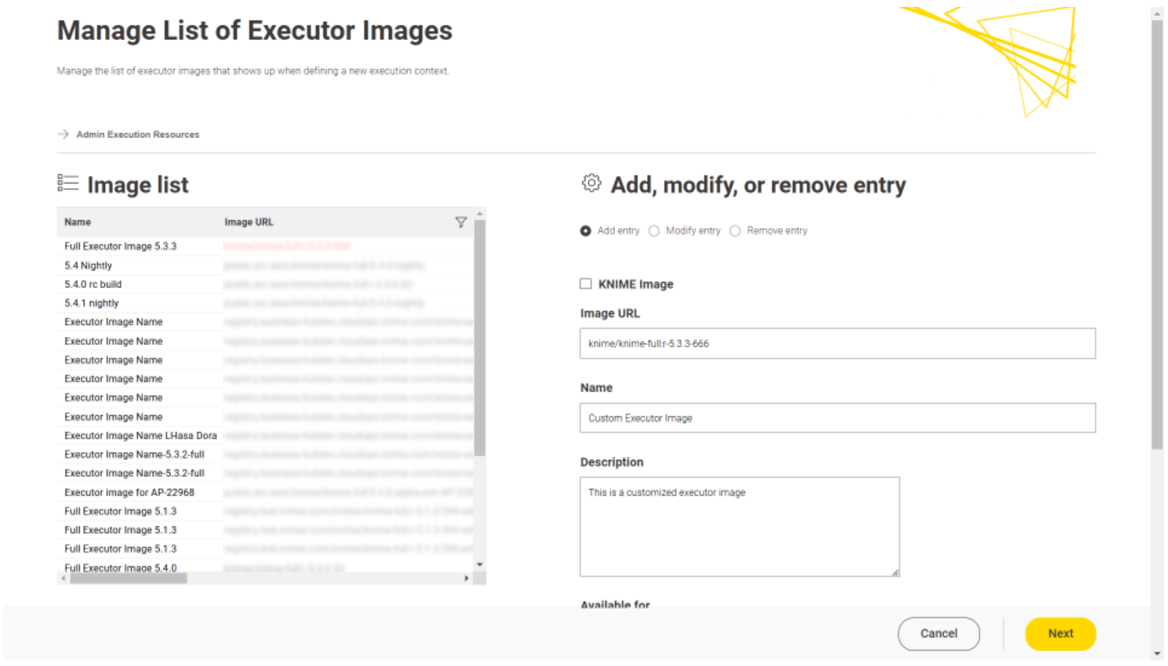
Name: Minimal Executor Image 5.4.0

Description: This is a minimal executor image with 5.4.0

Available for ...: [✓ ... all teams] ... specific teams

Click "Next" to apply changes.

  - Unselect *KNIME Image* to be able to add the URL of an image on your own registry.



Manage List of Executor Images

Manage the list of executor images that shows up when defining a new execution context.

Admin Execution Resources

Image list

Name	Image URL
Full Executor Image 5.3.3	[redacted]
5.4 Nightly	[redacted]
5.4.0 rc build	[redacted]
5.4.1 nightly	[redacted]
Executor Image Name	[redacted]
Executor Image Name	[redacted]
Executor Image Name	[redacted]
Executor Image Name	[redacted]
Executor Image Name	[redacted]
Executor Image Name	[redacted]
Executor Image Name LHasa Dora	[redacted]
Executor Image Name-5.3.2-full	[redacted]
Executor Image Name-5.3.2-full	[redacted]
Executor image for AP-22968	[redacted]
Full Executor Image 5.1.3	[redacted]
Full Executor Image 5.1.3	[redacted]
Full Executor Image 5.1.3	[redacted]
Full Executor Image 5.4.0	[redacted]

Add, modify, or remove entry

[✓ Add entry] Modify entry Remove entry

☐ KNIME Image

Image URL: knime/knime-full:r-5.3.3-666

Name: Custom Executor Image

Description: This is a customized executor image

Available for

- Click *Check for duplicates* to check if you gave a unique name to the entry or the entry is based on the same image as an existing entry.
- Click *Next* to apply the changes.

Now, the executor will be available to the team admins to create execution contexts for their team.

Modify entry

- Select *Modify entry*
- Select the image you want to modify from the table on the left.
- Change the name and or the description of the image you selected.
- Click *Check for duplicates* to check if you gave a unique name to the entry.
- Click *Next* to apply the changes. You should now see the updated entry in the list.

Image URL and team availability can not be modified since execution contexts that use the image would not be automatically updated, leading to inconsistet states. If you want to update URL or team availability, delete the respective entry first and add a new one.

Remove entry

- Select *Remove entry*
- Select the image you want to delete from the table on the left.
- Click *Next* to apply the changes. You should not see the entry in the list anymore.

Removing images from the list will not change execution contexts that are already running.

Removing images from the list does not free up disk space, since this list is independent of the registry where images are stored.

Add extensions to an existing Docker image

In order to install additional extensions and features to the KNIME Executor image, you will need to first create a `Dockerfile`, build a Docker image from it, and finally make it available by pushing it to a registry.

You can do this in two ways:

1. Via a data application, provided by us and described in the **Using the Executor Image Builder data application** section.
   Prerequisites:
   - The Execution Image Builder needs to be enabled: *KOTS Admin Console > Config > Execution Image Builder >* check *Enable Execution Image Builder*.
   - You need global admin privileges in order to build and push the image. Deploy the data app while configuring it with an application password of the global admin to make it available for others, e.g. team admins (the application password is not visible to others that way). If you use the data app not as a global admin, you can still create the Dockerfile, but you won’t be able to build and push it to the registry.
2. Via a manual approach, as described in the **Adding extensions manually** section, if the Execution Image Builder service is not available. This might be the case for airgapped installations, since the Execution Image Builder service references update sites, base images, and other external resources.
   Prerequisites:
   - `docker` should be installed.

   If you need to install `docker` please make sure not to install it on the same virtual machine (VM) where the KNIME Business Hub instance is installed, as it might interfere with `containerd`, which is the container runtime used by Kubernetes.

Registry prerequisites

Enabling the Image Builder allows to build an execution image in the cluster and to push it to a specified registry.

There are three different possibilities to specify a registry to which the Image Builder can have access:

1. If you are in an embedded cluster environment and want to use the default **Embedded Docker Registry**: activate the Embedded Docker Registry by going to *KOTS Admin Console > Config > Embedded Registry* and check the option *Enable Embedded Registry*. If you follow the manual approach, have the username and password ready that is defined in the KOTS Admin Console. See also KNIME Business Hub Installation Guide for more information.
2. If you are in an environment where the embedded registry is not available you will need to create a secret for the registry you want to use and point the Hub to the secret in the *Custom Execution Image Pull Secret* section of the KOTS Admin Console.
   a. First create a secret and add it to the cluster in the `knime` namespace via the command:

      ```
      kubectl -n knime create secret docker-registry <secret-name> --docker
      -server=<registry-url> --docker-username=<username> --docker
      -password=<secret>
      ```

      where `secret-name` is the name of the secret you want to create and `<registry-url>` is the URL of the registry you want to use.

   b. Point the Hub to the secret you added to the cluster. To do so go to the KOTS Admin Console, navigate to the *Execution Contexts* section and check the option *Enable Custom Execution Image Pull Secret*. Here under *Execution Image Pull Secret* add the `<secret-name>` from the command above. You will the be able to access the defined registry from the data app. If you are following the manual approach instead, you don't need to create the secret, but you will only need the URL, username, and password of the registry.
3. If you are in an airgapped environment instead you will need to specify a registry in the *Registry settings* tab of the KOTS Admin Console. If you follow the manual approach, have the username and password for this registry ready.

Building a Docker image requires enhanced container privileges. To be precise, the container needs to run as root with following capabilites:

- `CHOWN`
- `SETUID`
- `SETGID`
- `FOWNER`
- `DAC_OVERRIDE`

Depending on the size of the image, the build requires a lot of resources and can potentially slow down the KNIME Business Hub during the build time.

Using the Executor Image Builder data application

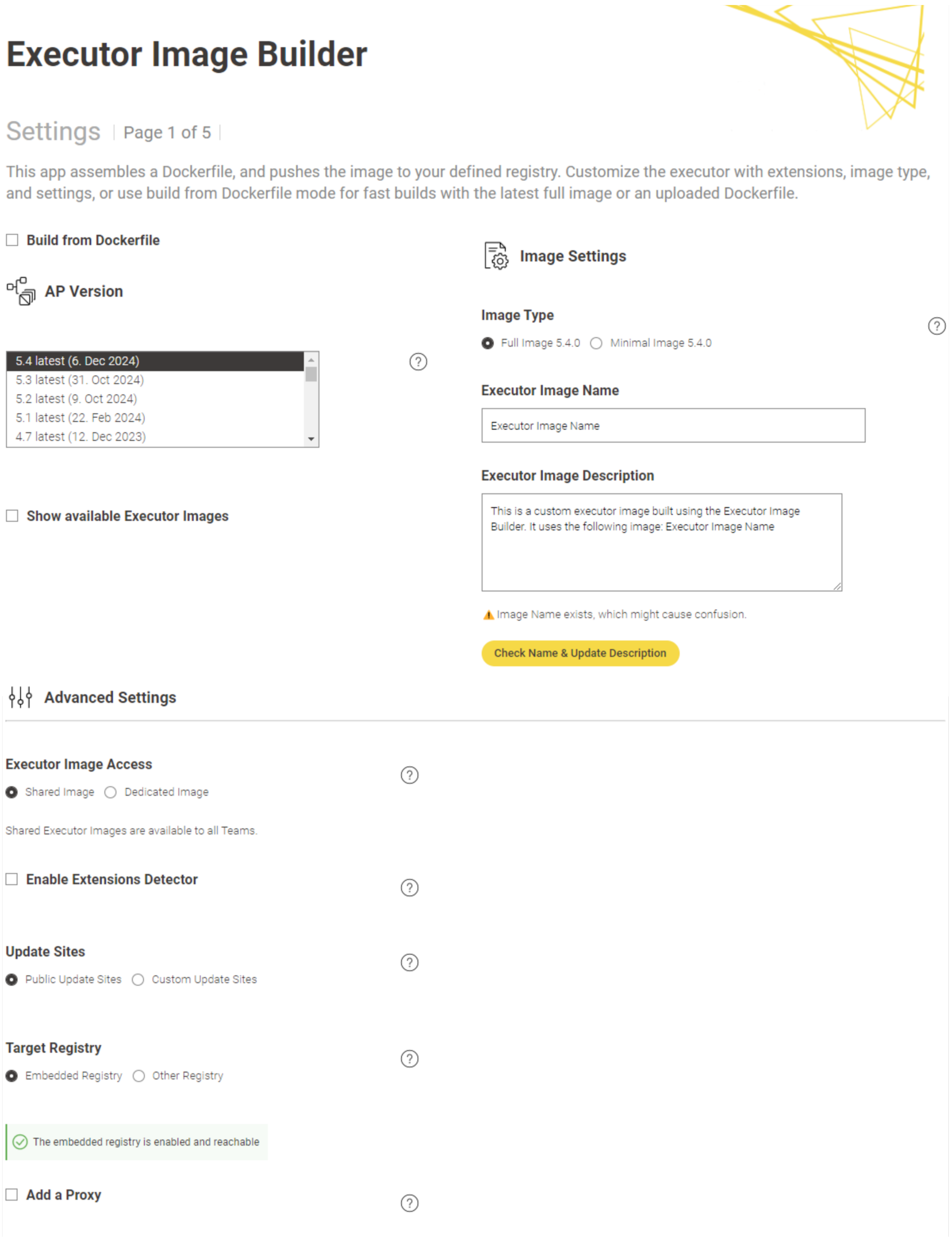
We provide the "Executor Image Builder" data application that will help you create the file, build an executor image, and push it to a registry to make it available when creating or editing an execution context. Download the data application, upload it to your KNIME Business Hub instance, and *Run* it.

Follow the steps 1 through 3 of the Data apps on KNIME Hub for detailed instructions.

The data app consists of five steps, displayed on five pages.

First page: Settings

First, you will find the basic settings.



Executor Image Builder

Settings | Page 1 of 5 |

This app assembles a Dockerfile, and pushes the image to your defined registry. Customize the executor with extensions, image type, and settings, or use build from Dockerfile mode for fast builds with the latest full image or an uploaded Dockerfile.

☐ Build from Dockerfile

AP Version

- 5.4 latest (6. Dec 2024) [✓]
- 5.3 latest (31. Oct 2024)
- 5.2 latest (9. Oct 2024)
- 5.1 latest (22. Feb 2024)
- 4.7 latest (12. Dec 2023)

☐ Show available Executor Images

Image Settings

Image Type

[✓ Full Image 5.4.0] Minimal Image 5.4.0

Executor Image Name

Executor Image Name

Executor Image Description

This is a custom executor image built using the Executor Image Builder. It uses the following image: Executor Image Name

⚠ Image Name exists, which might cause confusion.

Check Name & Update Description

Advanced Settings

Executor Image Access

[✓ Shared Image] Dedicated Image

Shared Executor Images are available to all Teams.

☐ Enable Extensions Detector

Update Sites

[✓ Public Update Sites] Custom Update Sites

Target Registry

[✓ Embedded Registry] Other Registry

The embedded registry is enabled and reachable

☐ Add a Proxy

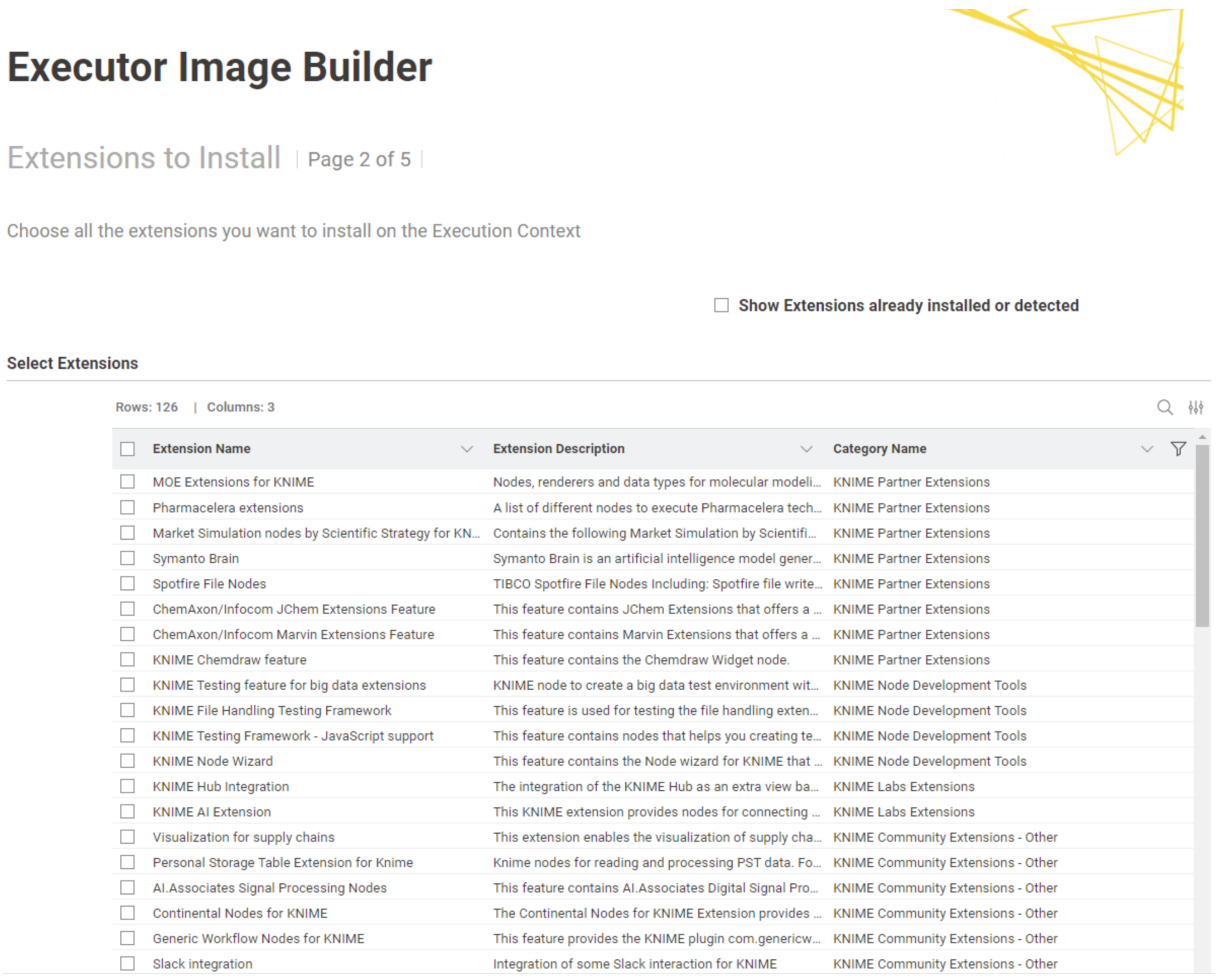
Figure 15. Executor Image Builder Settings on the first page as seen by a global admin.

If you hover over the setting or question mark in the data app, you will find a detailed description of each setting.

The settings related to image building and pushing are only available to global admins.

You proceed by clicking the *Next* Button.

Second page: Extension selection



Executor Image Builder

Extensions to Install | Page 2 of 5 |

Choose all the extensions you want to install on the Execution Context

☐ Show Extensions already installed or detected

Select Extensions

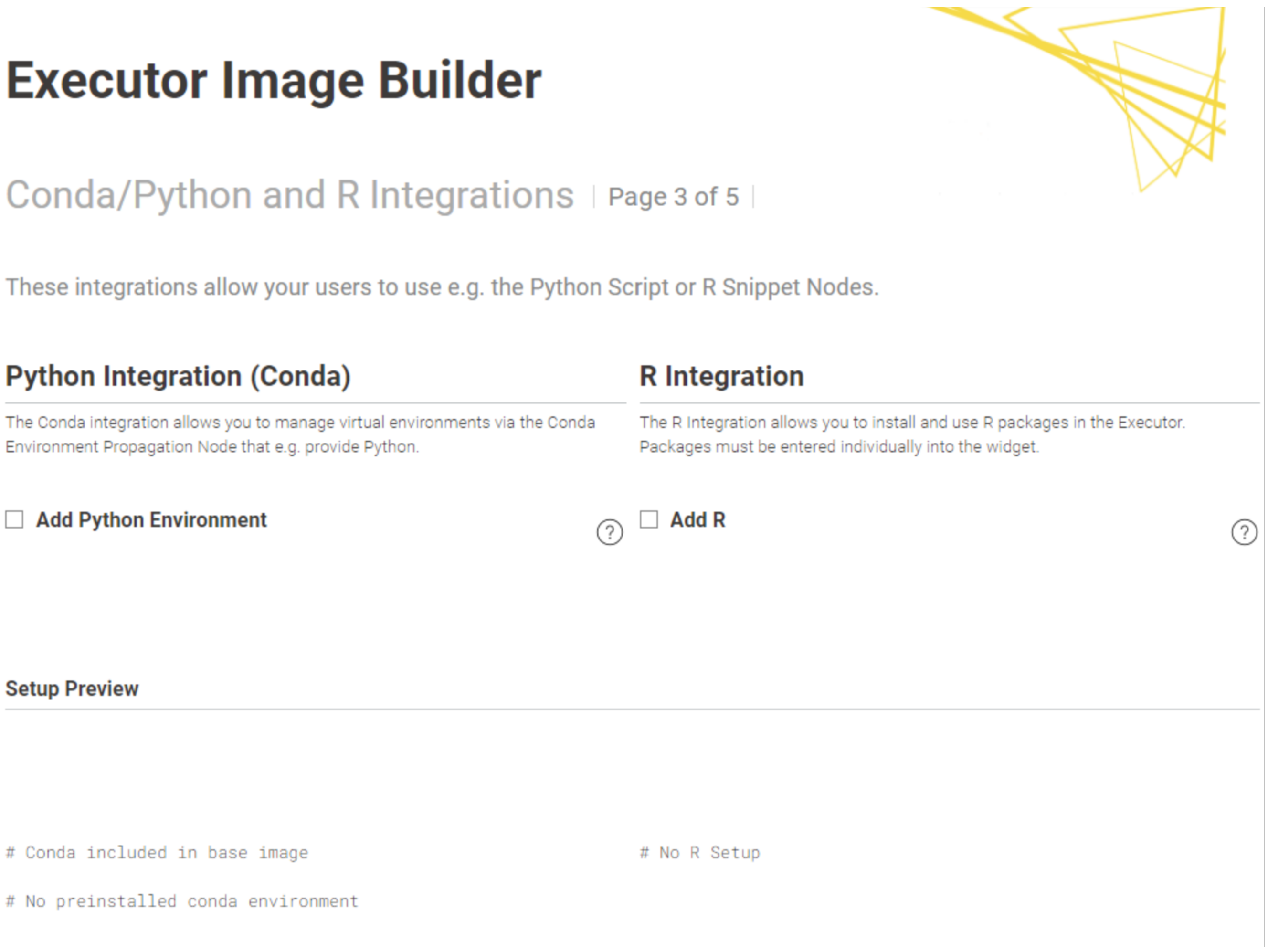
Rows: 126 | Columns: 3

☐	Extension Name	Extension Description	Category Name
☐	MOE Extensions for KNIME	Nodes, renderers and data types for molecular modeli...	KNIME Partner Extensions
☐	Pharmacelera extensions	A list of different nodes to execute Pharmacelera tech...	KNIME Partner Extensions
☐	Market Simulation nodes by Scientific Strategy for KN...	Contains the following Market Simulation by Scientifi...	KNIME Partner Extensions
☐	Symanto Brain	Symanto Brain is an artificial intelligence model gener...	KNIME Partner Extensions
☐	Spotfire File Nodes	TIBCO Spotfire File Nodes Including: Spotfire file write...	KNIME Partner Extensions
☐	ChemAxon/Infocom JChem Extensions Feature	This feature contains JChem Extensions that offers a ...	KNIME Partner Extensions
☐	ChemAxon/Infocom Marvin Extensions Feature	This feature contains Marvin Extensions that offers a ...	KNIME Partner Extensions
☐	KNIME Chemdraw feature	This feature contains the Chemdraw Widget node.	KNIME Partner Extensions
☐	KNIME Testing feature for big data extensions	KNIME node to create a big data test environment wit...	KNIME Node Development Tools
☐	KNIME File Handling Testing Framework	This feature is used for testing the file handling exten...	KNIME Node Development Tools
☐	KNIME Testing Framework - JavaScript support	This feature contains nodes that helps you creating te...	KNIME Node Development Tools
☐	KNIME Node Wizard	This feature contains the Node wizard for KNIME that ...	KNIME Node Development Tools
☐	KNIME Hub Integration	The integration of the KNIME Hub as an extra view ba...	KNIME Labs Extensions
☐	KNIME AI Extension	This KNIME extension provides nodes for connecting ...	KNIME Labs Extensions
☐	Visualization for supply chains	This extension enables the visualization of supply cha...	KNIME Community Extensions - Other
☐	Personal Storage Table Extension for Knime	Knime nodes for reading and processing PST data. Fo...	KNIME Community Extensions - Other
☐	AI.Associates Signal Processing Nodes	This feature contains AI.Associates Digital Signal Pro...	KNIME Community Extensions - Other
☐	Continental Nodes for KNIME	The Continental Nodes for KNIME Extension provides ...	KNIME Community Extensions - Other
☐	Generic Workflow Nodes for KNIME	This feature provides the KNIME plugin com.genericw...	KNIME Community Extensions - Other
☐	Slack integration	Integration of some Slack interaction for KNIME	KNIME Community Extensions - Other

Figure 16. Executor Image Builder Extension selection on the second page.

This is the page where you select the extensions you want to install additionally. If you select the checkbox in the top right corner, you will see both the extensions detected automatically (if enabled) and the extensions already installed in the base or full image.

Third page: Integration selection (R, Python)


Executor Image Builder

Conda/Python and R Integrations | Page 3 of 5 |

These integrations allow your users to use e.g. the Python Script or R Snippet Nodes.

Python Integration (Conda)

The Conda integration allows you to manage virtual environments via the Conda Environment Propagation Node that e.g. provide Python.

☐ Add Python Environment

R Integration

The R Integration allows you to install and use R packages in the Executor. Packages must be entered individually into the widget.

☐ Add R

Setup Preview

```
# Conda included in base image

# No preinstalled conda environment
```

```
# No R Setup
```

Figure 17. Executor Image Builder Python and R Integration configuration on the third page.

On the third page, you decide on whether to install conda/python, respectively additional default environments to the image, or the R integration.

Python

Learn more about how the KNIME Python Integration works with the KNIME Python Integration Guide

If your choice of image type (base or full) does not already include conda (specifically: micromamba as the management software of the virtual environments), you can choose to add it here. You need this, if the workflows you run on this executor contain e.g. a Conda Environment Propagation node.

If conda/micromamba is installed, you can further choose to provide a virtual environment that is already installed on the executor. This has the advantage that after a restart of the executor, the Conda Environment Propagation node executes faster, as the environment does not need to be recreated first. You add an environment by uploading a `.yml` file, which describes the environment. You can use the following example `.yml` file and modify it to your needs, e.g. by adding packages (and their versions) that you require:

```
name: py311_knime                # Name of the created environment
channels:                        # Repositories to search for packages
- knime
- conda-forge
dependencies:                    # List of packages that should be installed
- knime-python-base=5.3          # Metapackage for KNIME-connectivity
- python=3.11                    # Specify your desired Python version
#- your-module (required)=module-version (optional)
```

To learn more about how to create conda environment files, visit the conda docs. If you want to pre-install multiple conda environments, please edit the Dockerfile manually on the next page of the data app.

For KNIME executions prior to version 5.2, the executor needs to know where the conda installation is located so that the execution context can properly access conda. This is done as a final step and is described below.

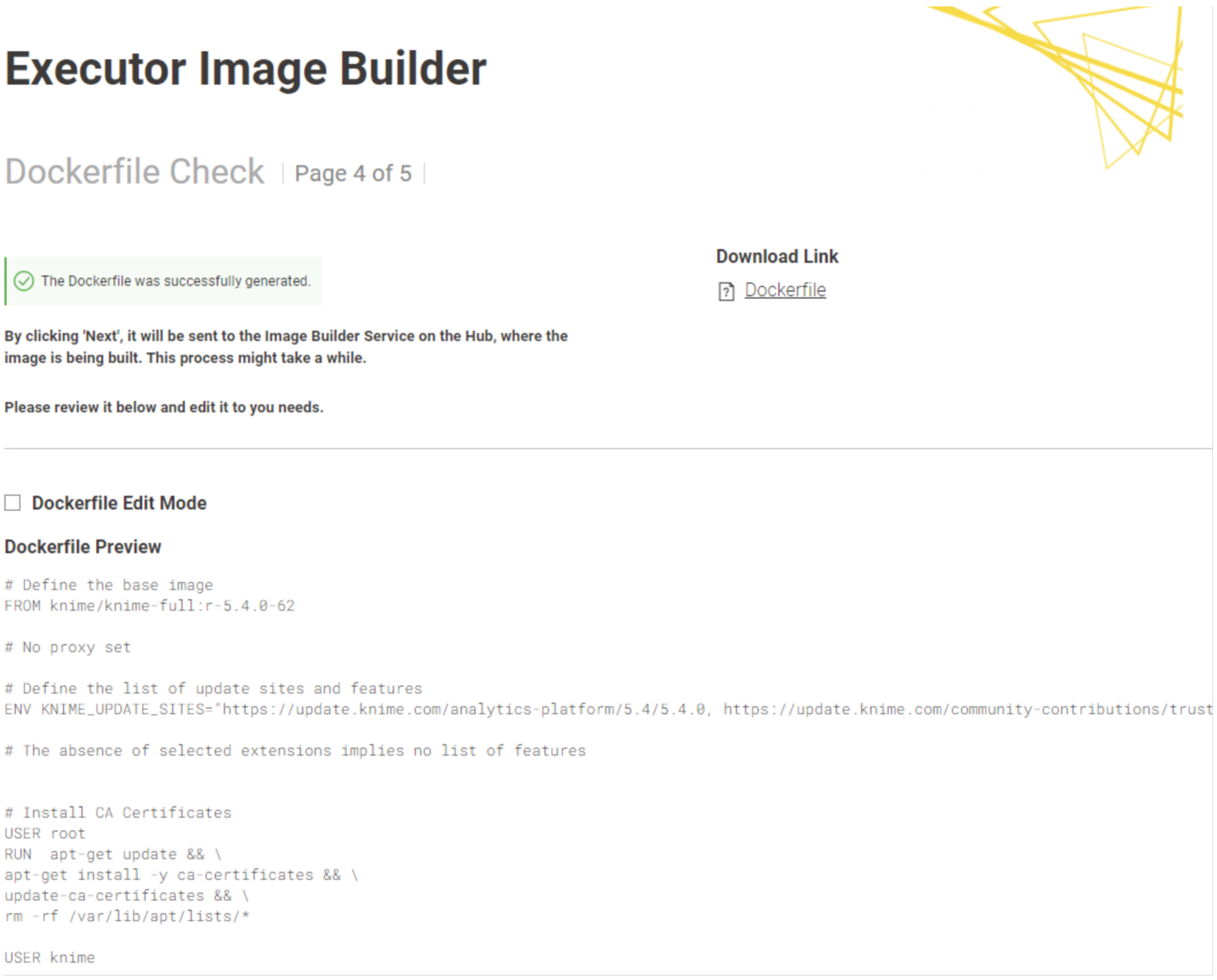
Unless modified, the path to the conda installation is `/home/knime/miniconda3/`, and the path to the environment (see below) is `<path to default conda environment dir>=<path to conda installation dir>/envs/<name of the env>`.

In the full images of KNIME Executor version 5.2 and above, where conda and Python are preinstalled, the path to conda is the same, i.e. `/home/knime/miniconda3/` and the paths to the default environments are `/home/knime/miniconda3/py2_knime` and `/home/knime/miniconda3/py3_knime`.

R

You can make R available as an integration if the workflows are expected to contain nodes from the KNIME R Scripting extension. Provide a comma-separated list of additional packages in the provided input field.

Fourth page: Dockerfile review



Executor Image Builder

Dockerfile Check | Page 4 of 5 |

The Dockerfile was successfully generated.

By clicking 'Next', it will be sent to the Image Builder Service on the Hub, where the image is being built. This process might take a while.

Please review it below and edit it to you needs.

Download Link

Dockerfile

☐ Dockerfile Edit Mode

Dockerfile Preview

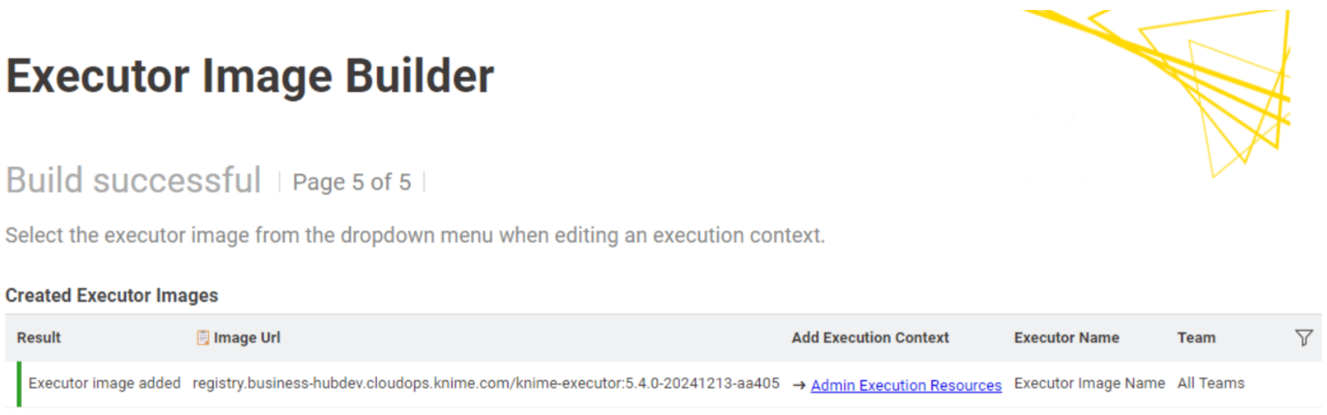
```
# Define the base image
FROM knime/knime-full:r-5.4.0-62

# No proxy set

# Define the list of update sites and features
ENV KNIME_UPDATE_SITES="https://update.knime.com/analytics-platform/5.4/5.4.0, https://update.knime.com/community-contributions/trust

# The absence of selected extensions implies no list of features


# Install CA Certificates
USER root
RUN  apt-get update && \
apt-get install -y ca-certificates && \
update-ca-certificates && \
rm -rf /var/lib/apt/lists/*

USER knime
```

Figure 18. Executor Image Builder Dockerfile review on the fourth page. This is the final page for users that are not global admins.

Here you are able to review the Dockerfile and do manual edits, if needed. To do this, enable the Dockerfile Edit Mode, and change the Dockerfile based on your needs.

- If you are **not** connected as a **global admin**, you can proceed to build and push the Dockerfile manually.
- If you are the **global admin** and proceed to the next page, the Dockerfile is sent to the image builder endpoint, where the image is actually built, and then pushed to the executor image list.

Fifth page: Dockerfile building and pushing



Executor Image Builder

Build successful | Page 5 of 5

Select the executor image from the dropdown menu when editing an execution context.

Created Executor Images

Result	Image Url	Add Execution Context	Executor Name	Team
Executor image added	registry.business-hubdev.cloudops.knime.com/knime-executor:5.4.0-20241213-aa405	→ Admin Execution Resources	Executor Image Name	All Teams

Figure 19. Executor Image Builder final page with the results.

On the final page of the Executor Image Builder, you will see the result of the building process. If no errors occurred during the build, the final image URL will be displayed. For KNIME Business Hub 1.13, you can simply select the execution context from the dropdown menu in the Execution Resources panel. For earlier versions, copy the image URL and then proceed to the Execution Resources page to create a new execution context or edit an existing one, pasting the URL where requested.

Finally, you can test the execution of a workflow containing a node from the freshly installed extensions or one that uses python.

Adding extensions manually

Instead of using the data app, you can also go through the steps manually. This might be necessary for airgapped installations, as the service that builds the docker image would require access to e.g. update sites and other resources that are referenced in your docker file.

Create the Dockerfile

You can use the example below which demonstrates how to extend an existing executor image with a custom set of update sites and features.

```
# Define the base image
FROM knime/knime-lts-full:r-5.3.3-666

# Change to root user to be able to install system packages
USER root

# Update/upgrade package manager and install ca-certificates to enable ca certificates
that micromamba (for python) is asking for
RUN apt-get update && \
    apt-get upgrade -yq && \
    apt-get install -yq \
        ca-certificates && \
    # cleanup
    rm -rf /var/lib/apt/lists/*

# Change to knime user to handle extensions
USER knime

# Define the list of update sites and features
# Optional, the default is the KNIME Analytics Platform update site (first entry in the
list below)
ENV KNIME_UPDATE_SITES=https://update.knime.com/analytics-
platform/5.2,https://update.knime.com/community-contributions/trusted/5.2
# Install a feature from the Community Trusted update site
ENV KNIME_FEATURES="org.knime.features.geospatial.feature.group"

# Execute extension installation script
RUN ./install-extensions.sh
```

The `KNIME_UPDATE_SITES` environment variable determines the update sites that will be used for installing KNIME Features. It accepts a comma-delimited list of URLs. The `KNIME_FEATURES` environment variable determines the extensions which will be installed in the KNIME Executor. It accepts a comma-delimited list of feature group identifiers. A corresponding update site must be defined in the `KNIME_UPDATE_SITES` list for feature groups to be successfully installed. You can get the necessary identifiers by looking at *Help → About KNIME → Installation Details → Installed Software* in a KNIME instance that has the desired features installed. Take the identifiers from the "Id" column and make sure you do not omit the `.feature.group` at the end (see also screenshot on the next page). The base image contains a shell script `install-extensions.sh` which lets you easily install additional extensions in another `Dockerfile`.

The Executor Image Builder Data App allows you to create a Dockerfile, which you can use as a starting point.

Python

If you want the executor to run workflows with the Conda Environment Propagation node, you need to have conda installed (respectively micromamba or any other conda environment manager).

- KNIME version < 5.2: If you are attempting to build an executor using a KNIME version prior to 5.2, you must explicitly install conda. There are several ways to set up and use Python on KNIME Business Hub Execution Contexts.
- KNIME version >5.2: For versions later than 5.2, the full image includes a bundled Python environment, while the minimal image does not have Python installed.
- If you want to start from the minimal `knime` image and want to install conda as well as a persisted conda environment, you can adapt your dockerfile based on the following:

```
# Define the knime minimal base image
# Select other KNIME versions from https://hub.docker.com/r/knime/knime-lts/tags
FROM knime/knime-lts:r-5.3.2-575
# Install the conda Extension to make the conda environment propagation node
available,
# as well as the Python Integration for the Python Script and Python View nodes.
# Include further update sites and extension IDs when desired.
ENV KNIME_UPDATE_SITES="https://update.knime.com/analytics-platform/5.3"
ENV
KNIME_FEATURES="org.knime.features.conda.feature.group,org.knime.features.python3.
scripting.feature.group"
RUN ./install-extensions.sh
# Download and install Miniforge, a miniconda installer configured to use conda-
forge only
# Note: '/home/knime/miniconda3' is the default path where KNIME will look for
conda.
# If conda is found at a different location, you need to configure KNIME to look
for
# conda at that location via a customization location profile. See
# https://docs.knime.com/latest/python_installation_guide/index.html#executor
# for detailed documentation and
# https://hub.knime.com/-/spaces/-/~CbbkOSFJ-SXFfvxn/most-recent/
# for a data app to assist you with customization profiles.
USER root
ENV conda_DIR=/home/knime/miniconda3
RUN apt update && \
    apt install curl bzip2 ca-certificates --yes && \
    apt clean && \
    curl -L -O "https://github.com/conda-
forge/miniforge/releases/latest/download/Miniforge3-$(uname)-$(uname -m).sh" && \
    bash Miniforge3-$(uname)-$(uname -m).sh -b -p $conda_DIR && \
    rm Miniforge3-$(uname)-$(uname -m).sh && \
    chown -R knime:knime $conda_DIR
USER knime
# Add conda to the PATH
ENV PATH=$conda_DIR/bin:${PATH}
```

```
# Create a new conda environment "my_env" with the knime-python-base package.
# Using EOF to introduce a heredoc. Adopt the lines 36 to 44.
RUN cat <<EOF > /tmp/my_env.yml
name: my_env                # Name of the created environment
channels:                   # Repositories to search for packages
- conda-forge
- knime
dependencies:               # List of packages that should be installed
- knime-python-base=5.3     # Metapackage for KNIME-connectivity
- python=3.11               # Specify your desired Python version
- <your package>=version
EOF
RUN conda env create -f /tmp/my_env.yml && \
    conda clean --all --yes && \
    # (Optional) Initialize conda in this shell, so that "conda activate" works
when running the image in a shell
    conda init bash
```

If you do this and you change the location of the conda installation to anything else than `/home/knime/miniconda3`, remember to configure the location of conda to use with KNIME in your customization profile as described in the section below.

- In case you want to start from a `knime-full` image that already has conda and the KNIME Python Integration pre-installed, please adopt your Dockerfile like this:

```
# Define the knime-full base image
# Select other KNIME versions from https://hub.docker.com/r/knime/knime-lts-
full/tags
FROM knime/knime-lts-full:r-5.3.2-575
# Create a new conda environment "my_env" with the knime-python-base package.
# Using EOF to introduce a heredoc. Adopt the lines 7 to 15.
RUN cat <<EOF > /tmp/my_env.yml
name: my_env                # Name of the created environment
channels:                   # Repositories to search for packages
- conda-forge
- knime
dependencies:               # List of packages that should be installed
- knime-python-base=5.3     # Metapackage for KNIME-connectivity
- python=3.11               # Specify your desired Python version
- <your package>=version
EOF
RUN conda env create -f /tmp/my_env.yml && \
    conda clean --all --yes && \
    # (Optional) Initialize conda in this shell, so that "conda activate" works
when running the image in a shell
    conda init bash
```

Learn more about the KNIME Python Integration with the KNIME Python Integration Guide.

> In order for the executor to find the conda installation, the execution context needs to know where to find conda. This is done as a final step and is described below.
>
> Unless modified, the path to the conda installation is `/home/knime/miniconda3/`, and the path to the environment (see below) is `<path to default conda environment dir>=<path to conda installation dir>/envs/<name of the env>`.
>
> In the full images of KNIME Executor version 5.2 and above, where conda and Python are preinstalled, the path to conda is the same, i.e. `/home/knime/miniconda3/` and the paths to the default environments are `/home/knime/miniconda3/py2_knime` and `/home/knime/miniconda3/py3_knime`.

Build a Docker image from the Dockerfile

Once the `Dockerfile` has been customized appropriately, you can build a Docker image from it by using the following command after replacing `<image_name>` and `<tag_name>` with actual values:

```
docker build -t <image_name>:<tag_name> .
```

This process can take a few minutes to be completed. In order to check if the new image has been built you can use the command `docker images`.

Push the Docker image to the Docker Embedded Registry

Finally you can push the image to the Docker Embedded Registry.

1. Authenticate against the registry with the credentials obtained from the *KOTS Admin Console > Config > Embedded Registry* via

   ```
   docker login --username <username> registry.<base-url>
   ```

   > If TLS is not configured, the registry URL must be added as an insecure registry.

2. Tag the previously created image with the format of the Embedded Docker Registry

   ```
   docker tag <old-name> registry.<base-url>/<new_name>
   ```

3. Push the image to the Embedded Docker Registry

   ```
   docker push registry.<base-url>/<image_name>:<tag_name>
   ```

4. Now the Docker image (e.g. `registry.hub.example.com/knime-full:5.2-with-additional-extension`) is available to create an execution context from the Hub UI.

Push the Docker image to the Container Embedded Registry for airgapped installations

For airgap installations you will need to create a custom Docker image and push it to the Container Embedded Registry using `containerd` command line (`ctr`).

As Containerd is installed as container runtime in the cluster, you can make use of the `ctr` commands to pull and push the images into the embedded registry.

1. Build the image on a machine with access to the internet and installed `docker`:

   ```
   docker build . -t registry.<base-url>/<image_name>:<tag_name>
   ```

2. Save the Docker image as a `tar` file on the machine where Docker is installed:

   ```
   docker save -o docker.tar <tag_name>
   ```

3. Copy the image to the machine where the Hub instance is running
4. On the machine where the Hub instance is running follow these steps. You can use the `ctr` or `docker` commands. However, be aware that using `docker` commands requires Docker to be installed on the machine.
   a. If you use `ctr`:
      i. First import the image into `containerd`:

      ```
      ctr image import docker.tar
      ```

      ii. Tag the image:

      ```
      ctr images tag <old-image_name>:<old-tag_name> registry.<base-
      url>/<image_name>:<tag_name>
      ```

      iii. Push the image to the Container Registry:

```
ctr images push --user <username> -k registry.<base-
url>/<image_name>:<tag_name>
```

where `-k` parameter is to skip the TLS check, and `--user` parameter is to provide the username for the registry.

b. If you use `docker`:

   i. First load the image into Docker:

   ```
   docker load -i docker.tar
   ```

   ii. Then login to the registry:

   ```
   docker login --username <username> registry.<base-url>
   ```

   iii. Tag the image:

   ```
   docker image tag <old-image_name>:<old-tag_name> registry.<base-
   url>/<image_name>:<tag_name>
   ```

   iv. Push the image to the Container Registry:

   ```
   docker push registry.<base-url>/<image_name>:<tag_name>
   ```

5. Now you can verify that the images are available on the Container Registry using the below endpoints:

```
http://registry.<base-url>/v2/_catalog
http://registry.<base-url>/v2/<repo>/tags/list
```

Push the Docker image to a non-embedded Registry

For non-embedded registries you will push the image from your local Docker instance to the remote registry, using the authentication required for that particular registry. Then you can use the image name to create the executor images, given that you have added the correct pull secrets to the KNIME Business Hub configuration.

Python and Conda in Docker images

When you create an Execution Context on KNIME Business Hub based on a full build you will have the Python environment bundled with the KNIME Python Integration available. If you need additional libraries or are using the Python 2 (legacy) extension, you need to create a custom Python environment to make them available on the Hub instance.

You can do this in several ways:

1. You can use the Conda Environment Propagation node in all your workflows using Python. To get started with the Conda Environment Propagation node, check out the KNIME Python Integration Guide. This has the advantage that no further setup is needed, and you are done with this guide. Any libraries installed using the Conda Environment Propagation node will be removed, however, when the executor restarts and are installed again next time the node executes, so libraries that are used often should be installed as part of the executor Docker image to save time. This is described in the following.
2. You can customize the executor image. To do so, you need to create a Docker image with Python, either via the Executor Image Builder data application described above, or by creating (Dockerfile examples), building, and pushing the Dockerfile manually. Be sure to note down the paths where conda was installed, as you will need add them in the `.epf` file of the customization profile during the set up of the execution context. The default installation paths are:

   ```
   <path to conda installation dir>        = /home/knime/miniconda3/
   <path to default conda environment dir> = <path to conda installation
   dir>/envs/<name of the env>
   ```

3. Use a full image of version 5.2 and above, since starting with KNIME Executor version 5.2 the KNIME Python extension is already installed in the Docker images. There are two default environments installed, py2_knime and py3_knime:
   - `py2_knime`:

```
name: py2_knime              # Name of the created environment
channels:                    # Repositories to search for packages
  - defaults
  - conda-forge
dependencies:                # List of packages that should be installed
  - python=2.7               # Python
  - pandas=0.23              # Table data structures
  - jedi=0.13                # Python script autocompletion
  - parso=0.7.1              # Jedi dependency this is the last version
compatible with 2.7
  - python-dateutil=2.7      # Date and Time utilities
  - numpy=1.15               # N-dimensional arrays
  - cairo=1.14               # SVG support
  - pillow=5.3               # Image inputs/outputs
  - matplotlib=2.2           # Plotting
  - pyarrow=0.11             # Arrow serialization
  - IPython=5.8              # Notebook support
  - nbformat=4.4             # Notebook support
  - scipy=1.1                # Notebook support
  - jpype1=0.6.3             # Databases
  - protobuf=3.5             # Serialization for deprecated Python nodes
```

◦ py3_knime:

```
name: py3_knime              # Name of the created environment
channels:                    # Repositories to search for packages
  - defaults
  - conda-forge
dependencies:                # List of packages that should be installed
  - nbformat=4.4             # Notebook support
  - scipy=1.1                # Notebook support
  - pillow=5.3               # Image inputs/outputs
  - cairo=1.14               # SVG support
  - ipython=7.1              # Notebook support
  - numpy=1.16.1             # N-dimensional arrays
  - python=3.6               # Python
  - matplotlib=3.0           # Plotting
  - jpype1=0.6.3             # Databases
  - pyarrow=0.11             # Arrow serialization
  - jedi=0.13                # Python script autocompletion
  - python-dateutil=2.7      # Date and Time utilities
  - pandas=0.23              # Table data structures
  - libiconv=1.15            # MDF Reader node
  - asammdf=5.19.14          # MDF Reader node
```

If you choose to modify the executor image or use the full build of version 5.2 and above, you further need to set up the execution context for it to know where to find the conda/python installations. This is described in the section below.

Set up the execution context

Once you have created the Docker image with Conda/Python and the desired environments, create an execution context that uses the newly created Docker image.

Now you need to set up and customize the execution context. This process is described in the KNIME Python Integration Guide in detail, and the relevant parts are repeated here.

You specify the paths where the execution context will find the conda installation and environments in a customization profile applied it to the execution context.

1. Build the `.epf` file by following the steps in KNIME Python Integration Guide and exporting the `.epf` file.
2. To export the `.epf` file from KNIME Analytics Platform, first switch to classic user interface (you can find the option in the Modern UI under *Menu*), then go to *File > Export Preferences…*
3. Open the file and use only the parts related to Python/conda.

The `.epf` file could look like the following:

```
/instance/org.knime.conda/condaDirectoryPath=<path to conda installation dir>
/instance/org.knime.python3.scripting.nodes/pythonEnvironmentType=conda
/instance/org.knime.python3.scripting.nodes/python2CondaEnvironmentDirectoryPath=<path
to default conda environment dir>
/instance/org.knime.python3.scripting.nodes/python3CondaEnvironmentDirectoryPath=<path
to default conda environment dir>
```

Find more details on how to setup the `.epf` file in the Executor configuration section of the KNIME Python Integration Guide.

Now follow these steps to customize the execution context:

1. Build the `.zip` file containing the customization profile using the `.epf` file you just created.
2. Upload the customization profile `.zip` file to KNIME Business Hub.
3. Apply the customization profile to the execution context.

You are done, and can test the setup by running a workflow that contains a Conda Environment Propagation node.

Delete a custom Docker image

In this section you can find instructions on how to delete the Docker images that you pushed to the Embedded Docker Registry. This is especially important since the MinIO storage allocated for the registry is, by default, limited to ~30 GB.

To check how much disk space is occupied you can run the command `kubectl exec -it -n minio minio-<id> -- /bin/sh -c "df -h"` with the correct `minio` pod id. The `/data` directory contains the space occupied by the registry.

The first step to delete the custom Docker images is to use the following script:

delete_registry_image.sh

```
#!/bin/bash

# exit when any command fails
set -e

registry='registry.<base-url>'

# concatenates all images listed in json file into single line string seperated with
blank
echo "Image Name:"
read images
echo "Image Tag (Space seperated for multiple tags or leave empty if all should be
deleted):"
read tags
echo "Registry User:"
read user
echo "Registry Password:"
read -s password

for image in $images; do

    if [[ -z $tags ]]
    then
        # get tag list of image, with fallback to empty array when value is null
        tags=$(curl --user $user:$password "https://${registry}/v2/${image}/tags/list" |
jq -r '.tags // [] | .[]' | tr '\n' ' ')
    fi

    echo "DELETING image: " $image
    echo "DELETING tags: " $tags

    # check for empty tag list, e.g. when already cleaned up
    if [[ -n $tags ]]
    then
        for tag in $tags; do
```

```
            curl --user $user:$password -X DELETE
"https://${registry}/v2/${image}/manifests/$(
                curl --user $user:$password -I \
                    -H "Accept: application/vnd.docker.distribution.manifest.v2+json" \
                    "https://${registry}/v2/${image}/manifests/${tag}" \
                | awk '$1 == "Docker-Content-Digest:" { print $2 }' \
                | tr -d $'\r' \
            )"
            echo "DELETED:" $image "with tag" $tag
        done
        echo "Deleted images. To free disk space run the garbage collection command
inside the registry pod: 'bin/registry garbage-collect --delete-untagged /
etc/docker/registry/config.yml'. See documentation at: https://docs.knime.com/2024-
06/business_hub_admin_guide/index.html#garbage-collection"
    else
        echo "SKIP:" $image
    fi
done
```

To run the script you will need:

1. `jq`: `jq` is a lightweight and flexible command line JSON processor that is used to format the JSON output of `curl` calls. To install it on the machine where you want to run the shell script you can use the following command:

   ```
   sudo apt-get update
   sudo apt-get -y install jq
   ```

2. You will need to adapt the value `registry.<base-url>` at the line 6 of the script with the `<base-url>` of your KNIME Business Hub, e.g. for `hub.example.com` will be `registry.hub.example.com`.
3. You will need to know the Embedded Docker Registry username and password.

> If your KNIME Business Hub instance does not have TLS enabled, the script will cause SSL certificate issues.
>
> To solve this you can:
>
> 1. Change the script where a `curl` command is calling an `https://` endpoint to call an `http://` instead, or
> 2. Add `--insecure` to each line of the script with a `curl` command.

Identify Docker image names and tags

To be able to run the script you will need to know the image names and tags present on the embedded registry that you want to delete. You can for example run the following `GET` requests against the Docker Registry API. Again you will need to first adapt the `<base-url>` entries to your specific Hub URL, and use your `<username>` and `<password>` for the Embedded Docker Registry.

```
# listing images on the remote registry
$ curl -u <username>:<password> -X GET registry.<base-url>/v2/_catalog | jq
```

This should output a list of the Docker Images available:

```
{
    "repositories": ["executor-image-name1",
                     "executor-image-name2",
                     "executor-image-name3"]
}
```

Then you can run the following command for each Docker Image you are interested in, e.g. `executor-image-name1`, to retrieve the Image tag:

```
# listing tags on the remote registry
$ curl -u <username>:<password> -X GET registry.<base-url>/v2/<executor-image-
name>/tags/list | jq
```

This should output a list of the Docker Image’s tags:

```
{
    "name": "executor-image-name1",
    "tags": ["tag1"]
}
```

Run the script

Now you can run the script to delete one or multiple image tags. The script will ask you to provide an Image name, one or more Image tag and the Embedded Docker Registry username and password.

```
$ ./delete-registry-image.sh

Image Name:
<image-name>
Image Tag (Space seperated for multiple tags or leave empty if all should be deleted):
<image-tag>
Registry User:
<usr>
Registry Password:
<pwd>
```

Run garbage collection on the `registry` pod

Since the above script only removes the image tags and manifests, leaving the actual image layers in storage, garbage collection is required to scan the registry storage and remove unreferenced or orphaned layers, reclaiming disk space and fully cleaning up after the deletion.

The garbage collection needs to be performed on one of the `registry` pods inside the `kurl` namespace.

You can either use the following commands via `kubectl` or use your preferred tool to manage the cluster, e.g. OpenLens.

1. Connect to the cluster where the Business Hub instance is installed
2. List the pods in the `kurl` namespace to find the `registry` pod on which you will run the garbage collection

   ```
   kubectl get pods -n kurl
   ```

   a. Identify the `registry` pod and proceed with the next steps

   b. You **only** need to perform garbage collection on one registry pod in the `kurl` namespace. There is no need to do it for the other registry pods. This is because all the registry pods in the Kubernetes cluster share the same underlying storage.

3. Next, open a shell into the selected registry pod, `<registry-name>`, and ensure you select the correct container

   a. Do **not** use the `registry-backup` container, to avoid risk of data loss

   b. Do not upload an image during garbage collection. If you were to upload an image while garbage collection is running, there is the risk that the image’s layers are mistakenly deleted leading to a corrupted image.

4. Run the garbage collection command inside the shell:

```
kubectl exec -it -n kurl <registry-name> -c registry -- /bin/sh -c "/bin/registry
garbage-collect --delete-untagged /etc/docker/registry/config.yml"
```

After the garbage collection has run through, it takes a while to fully free the disk space. You can speed up the process by restarting the `registry` and `minio` pods. To check how much disk space is occupied now run the command `kubectl exec -it -n minio minio-<id> -- /bin/sh -c "df -h"` with the correct `minio` pod id again. The `/data` directory contains the space occupied by the registry.

> Even though all tags of an image are deleted, the image is still listed in the MinIO pod under the `minio` namespace when running a `GET` request to the `registry.<base-url>/v2/_catalog` endpoint. However, these images should not have any tags and therefore they do not occupy disk space anymore.

Customization profiles

Customization profiles are used to deliver KNIME Analytics Platform configurations from KNIME Hub to KNIME Analytics Platform clients and KNIME Hub executors.

This allows defining centrally managed:

- Update sites
- Preference profiles (such as Database drivers, Python/R settings)

A profile consists of a set of files that can:

- Be applied to the client during startup once the KNIME Analytics Platform client is configured. The files are copied into the user's workspace.
- Be applied to the KNIME Hub executors of the execution contexts.

Customization profiles can be:

- Global customization profiles that need to be uploaded and managed by the global admin. These can be applied across teams via shared execution context or to specific teams.
- Team's customization profiles, which are scoped to the team, can be uploaded either by a global admin or a team admin.

Once uploaded, the customization profile can then be downloaded or used in KNIME Analytics Platform clients and executors.

ℹ The access to the customization profile is not restricted meaning that anyone with the link can download it and use it.

Currently, customization profiles can be managed on KNIME Hub via REST or using the dedicated Customization Profile data application available here.

Structure of a customization profile

A customization profile minimally consists of a folder, named according to the *profile name*, containing at least one *preference file*. A preference file is a simple text file with the extension `.epf`.

Each line in a preference (`.epf`) file specifies key and value of a setting, separated by `=`.

```
<key1>=<value1>
<key2>=<value2>
# ...
```

If two lines specify identical keys, the value later in the file overrides a value specified earlier.

If the profile folder contains more than one `.epf` files, the files are read in lexicographic order.

A customization profile may contain additional arbitrary files. These are distributed as part of the profile and can be referenced in `.epf` files.

Variable replacement

It is possible to use variables inside the preference files (only those files ending in `.epf`) which are replaced on the client right before they are applied. This makes the Hub-managed customizations even more powerful. These variables have the following format: `${prefix:variable-name}`. The following prefixes are available:

- **`env`**: the variable is replaced with the value of an environment value. For example, `${env:TEMP}` will be replaced with `/tmp` under most Linux systems.
- **`sysprop`**: the variable is replaced with a Java system property. For example, `${sysprop:user.name}` will be replaced with the current user's name. For a list of standard Java system properties see the Java documentation. Additional system properties can be defined via execution contexts.
- **`profile`**: the variable will be replaced with a property of the profile in which the current preference file is contained in. Currently `location` and `name` are supported as variable names. For example, `${profile:location}` will be replaced by the file system location of the profile on the client. This can be used to reference other files that are part of the profile, such as database drivers:
  `org.knime.workbench.core/database_drivers=${profile:location}/db-driver.jar`

In case you want to have a literal in a preference value that looks like a variable, you have to use two dollar signs to prevent replacement. For example `$${env:HOME}` will be replaced with the plain text `${env:HOME}`. If you want to have two dollars in plain text, you have to write three dollars (`$$${env:HOME}`) in the preference file.

Once you use variables in your preference files, they are not standard Eclipse preference files anymore and cannot be imported as they are.

Create a customization profile

Follow the steps below to create a customization profile. You can export the preference file from a KNIME Analytics Platform installation, with the needed configuration.

Then create a folder with the preference file and any additional file that you might need to distribute with the customization profile. Finally, you compress the folder to a `.zip` format and

1. Set up the needed configuration in a local KNIME Analytics Platform installation.
2. Export the `.epf` file from KNIME Analytics Platform. To do so, first switch to classic user interface (you can find the option in the Modern UI under *Menu*), then go to *File > Export Preferences…*.
3. Open the created `.epf` file, and look for the lines related to your needed settings. Remove all other settings (as some contain e.g. local paths on your machine, which will inevitably cause issues when applying to another installation). You can also further modify the file with the customization options below.
4. Place the `.epf` file in a folder, together with any additional files that need to be distributed along the profile (e.g. database drivers).
5. Create a `.zip` from that folder.

Finally you can proceed with the next step, and upload the file to the KNIME Hub instance.

When creating a zip file on macOS using the built-in functionality, two files are automatically added that cause the next steps (i.e. applying the profile in KNIME Analytics Platform) to fail. There is a way to prevent creation of these files if creating the `.zip` via command line, see here. If in doubt, use a Windows or Linux machine to create the `.zip` file.

The customization profiles on the KNIME Hub instance are going to be accessible without user authentication. Therefore, they shouldn't contain any confidential data such as passwords.

For further details and an example on how to distribute JDBC driver files, go to the Hub-managed customization profiles section of the KNIME Database Extension Guide.

Customization options

Besides the preferences that are exportable by KNIME Analytics Platform, there are additional settings that can be added to the preference files to customize clients.

Since KNIME Analytics Platform 5.3, some complex settings can be specified in YAML format.

1. Create a file `<filename>.yml` in the profile directory.
2. Reference the file in a preference (`.epf`) file by setting the following:

   ```
   /instance/org.knime.core/knime.core.ap-customization-
   configuration=${profile:location}/<filename>.yml
   ```

   where `<filename>.yml` is the name of the YAML file you created.

> The YAML file should declare the document version at the beginning of the file, i.e. `version: 'customization-v1.0'`. This is to ensure that the document can still be read in the future even if the format changes.

Logging

In KNIME Executors, there are two main logging targets available, namely `stdout` and `stderr` (in contrast to the KNIME Analytics Platform). Logs can be downloaded from execution contexts, including the job logs. The log level of job logs is set to `WARN`.

Using Eclipse preferences (`.epf`) files bundled in customization profiles, the following preference can be used to set the `<log-level>` to one of the supported minimum log levels: `DEBUG`, `INFO`, `WARN`, `ERROR`, and `OFF`.

This preference configures the log level of the *standard output* of a KNIME Executor (default is `INFO`).

```
/instance/org.knime.workbench.core/logging.loglevel.stdout=<log-level>
```

The *standard error* of a KNIME Executor is set to the log level `ERROR`, and is always non-overlapping with the standard output.

Restrict access to nodes

The property path `nodes.filter` allows to configure a sequence of *filters* which are evaluated on startup of the KNIME Analytics Platform.

The `scope` of a filter expresses to what extent access is restricted by this filter.

1. `view`: If a node does not match all filters with `view` scope, it can be loaded, configured and executed as part of a workflow, but it will not be presented as a choice in the node repository or similar places.
2. `use`: If a node does not match all filters with `use` scope, it will not be loaded and will not appear as a choice.

> `use` takes precedence over `view`. Meaning that a node that cannot be used can also never be viewed, regardless of whether it matches filters with the `view` scope.

Whether a node matches a filter is defined by its `predicate`, consisting of one or several regular expressions. The regular expression patterns are evaluated against the node factory class name. The value of `rule` (`allow` or `deny`) specifies whether the predicate is applied as-is or inverted.

Examples

- Completely ignore any Java Snippet or Row Filter nodes. Any other nodes are not affected.

```
version: 'customization-v1.0'
nodes:
  filter:
    - scope: use
      rule: deny
      predicate:
        type: pattern
        patterns:
          - .+JavaSnippet.+
          - .+RowFilter.+
        isRegex: true
```

- Completely ignore any Java Snippet node. Restrict node repository and related user interface elements to nodes from given modules.

```
version: 'customization-v1.0'
nodes:
  filter:
    - scope: use
      rule: deny
      predicate:
        type: pattern
        patterns:
          - .+JavaSnippet.+
        isRegex: true
    - scope: view
      rule: allow
      predicate:
        type: pattern
        patterns:
          - org\.<vendor>\.<package>\..+
          - org\.<vendor>\.<package>\..+
        isRegex: true
```

Custom UI menu entries

In the YAML customization file, use the property `ui.menuEntries` as illustrated by the following example.

```
version: 'customization-v1.0'
ui:
  menuEntries:
    - name: "My custom entry"
      link: "https://help.company.com/knime"
    - name: "Another entry"
      link: "https://support.company.com/contact"
```

These entries will appear in the *Help* menu revealed by the button at the top-right of the KNIME Analytics Platform user interface.

To configure custom menu entries in the **classic user interface**, use the following settings in a preference (`.epf`) file.

`/instance/com.knime.customizations/helpContact.buttonText=<label>`

If set together with `/instance/com.knime.customizations/helpContact.address` a button with the provided label will occur under `Help` in KNIME Analytics Platform. Clicking on the button will, depending on the `helpContact.address`, either open the default mail client or the default browser with the provided address.

`/instance/com.knime.customizations/helpContact.address=<uri>`

Sets the address of the support contact, e.g. `mailto:support@company` or `https://company/support`.
This option only takes effect in combination with `/instance/com.knime.customizations/helpContact.buttonText`.

`/instance/com.knime.customizations/documentation.buttonText=<label>`

Sets the label of the documentation button that can be found under `Help` in KNIME Analytics Platform. Clicking on the button will open the default browser and navigate to the documentation. If set to `-` the button will be hidden.

`/instance/com.knime.customizations/documentation.address=<uri>`

Sets the address of the documentation, e.g. `https://company/documentation` or `file:///sharedSpace/documentation`.
By default, the documentation address points to the KNIME Analytics Platform documentation.

Disallow storage of weakly encrypted passwords

You can use the `workflow.disablePasswordSaving` property to configure whether or not it should be possible to save weakly encrpypted passwords in workflows.

Example

- Prevent the user to save passwords in workflows.

```
version: 'customization-v1.0'
workflow:
  disablePasswordSaving: true
```

Disable KNIME AI Assistant on KNIME Analytics Platform client

KNIME Analytics Platform version 5.4.1 and later

If you want to deactivate the AI assistant and hide it from the side panel, you can do so by:

- Setting `kai.disable` property via customization profile.
- Setting the Java system property `-Dorg.knime.ui.feature.ai_assistant=false` (see here).

The Java system property will always take precedence over the customization profile.

KNIME Analytics Platform version 5.4.0
If you are using KNIME Analytics Platform of version 5.4.0 the Java system property is not available, therefore you need to set up the corresponding parameter via the customization profile by using the `kai.disable` property.

KNIME Analytics Platform version 5.3.x or before
If you are using KNIME Analytics Platform of version 5.3.x or before, the setting in the customization profile is not available, therefore you need to set the Java system property in the `knime.ini` file or via execution contexts.

Example

- Disable K-AI on KNIME Analytics Platform via customization profile.

```
kai:
    disable: true
```

Restrict which KNIME Hubs the KNIME AI Assistant is allowed to connect to

The `kai.hub.filter` property allows you to configure a sequence of filters that are evaluated on startup. These filters determine which KNIME Hubs can be selected as the backend for K-AI on the KNIME AI Assistant preferences page. This is similar to the `nodes.filter` property but with two key differences:

- **Filter target**: The filters match the URL host of a Hub (e.g., `api.hub.knime.com` for the KNIME Community Hub).
- **Scope**: There is no scope property as the filters only control which hubs K-AI is allowed

to connect to.

Examples

- Prevent K-AI from connecting to KNIME Community Hub.

```
version: 'customization-v1.0'
kai:
  hub:
    filter:
      - rule: deny
        predicate:
          type: pattern
          patterns:
            - hub.knime.com
          isRegex: false
```

- Allow only hubs of a certain domain.

```
version: 'customization-v1.0'
kai:
  hub:
    filter:
      - rule: allow
        predicate:
          type: pattern
          patterns:
            - .+\.<custom domain>\.com
          isRegex: true
```

Mount points

See here for more information on mounting, connecting to, and using mount points in KNIME Analytics Platform. Below, you will find configuration options for mount points, using customization profiles.

Disallow use of mount points by host name

These configuration options allow for completely restricting the use of certain mount points. Specifically, disallowed mount points will be unlisted from the KNIME space explorer and cannot be re-added, e.g., via *Preferences > KNIME > KNIME Explorer*.

- **Filter target**: The filters match the URL host of a Server or Hub (e.g., `api.hub.knime.com` for the KNIME Community Hub).
- **Scope**: There is no scope property as the filters only control which URL hosts are allowed for mount points in general.

Examples

- Prevent the user from using KNIME Community Hub.

```
version: 'customization-v1.0'
mountpoint:
  filter:
    - rule: deny
      predicate:
        type: pattern
        patterns:
          - api.hub.knime.com
        isRegex: false
```

- Only allow a KNIME Hub instance at a custom domain.

```
version: 'customization-v1.0'
mountpoint:
  filter:
    - rule: allow
      predicate:
        type: pattern
        patterns:
          - .+\.<custom domain>\.com
        isRegex: true
```

- Disallow all remote mount points, leaving only `LOCAL` available.

```
version: 'customization-v1.0'
mountpoint:
  filter:
    - rule: deny
      predicate:
        type: pattern
        patterns:
          - .*
        isRegex: true
```

Define and exclude default mount points by ID

These configuration options allow defining and excluding *default* mount points. Note that default mount points are the set of initially-configured mount points when starting KNIME Analytics Platform. For example, setting the `defaultMountpoint/enforceExclusion` option will not prevent the user from interacting with "excluded" mount points, as these can be re-added afterwards.

`/instance/org.knime.workbench.explorer.view/defaultMountpoint/defaultMountpoints=<mount id1>,<mount id2>,…`

A comma separated list of default mount points that should be loaded,e.g. `LOCAL,EXAMPLES,My-KNIME-Hub`. Changes to this list only affects new workspaces, i.e. workspaces which already contain default mount points will still contain them even though they haven't been defined here. If this option is absent and `defaultMountpoint/enforceExclusion` isn't set to true then all default mount points will be added. The current default mount points are `LOCAL`, `EXAMPLES`, and `My-KNIME-Hub`.

`/instance/org.knime.workbench.explorer.view/defaultMountpoint/enforceExclusion=<true|false>`

If set to true then all default mount point not defined by `/instance/org.knime.workbench.explorer.view/defaultMountpoint/defaultMountpoints` will be removed on start up. Please note that if you want to use this option, the default mount points you want to include should **only** be listed in `/instance/org.knime.workbench.explorer.view/defaultMountpoint/defaultMountpoints`, and **not** in their full definition like when exporting the preferences.epf file from a KNIME Analytics Platform.

Update Sites

`/instance/com.knime.customizations/updateSite.uris=<uri>,<uri>,…`

Adds the provided addresses to the update sites.

`/instance/com.knime.customizations/updateSite.names=<name>,<name>,…`

The names that are shown under `Available Software Sites` for the provided update sites. Note that the number of names must match the number of provided URIs.

`/instance/com.knime.customizations/updateSite.default.disable=<true|false>`

> Disables the default update sites added by KNIME after a fresh installation or update. If a user enables these update sites again they will remain enabled.

`/instance/com.knime.customizations/updateSite.default.forceDisable=<true|false>`

> Disables the default update sites added by KNIME after a fresh installation or update. If a user enables these update sites again they will be disabled with the restart of their client.

Columnar Backend Memory configuration

The columnar backend makes use of off-heap memory to store data tables. The memory is managed by the columnar backend and is not part of the Java heap (which is configured via the Xmx setting). Therefore, the memory configured for the columnar backend must be taken into account to avoid out-of-memory errors.

Depending on the executor version, different preferences can be set.

KNIME Analytics Platform version 4.7.x

`/instance/org.knime.core.data.columnar/knime.core.data.columnar.use-defaults=<true|false>`

> If true, the other preferences in this section are ignored.

`/instance/org.knime.core.data.columnar/knime.core.data.columnar.small-cache-size=<size>`

> Size of cache for tables smaller than 1MB in MB. Example value: `32`.

`/instance/org.knime.core.data.columnar/knime.core.data.columnar.data-cache-size=2048`

> Size of cache for other tables in MB. Example value: `2048`.

KNIME Analytics Platform version 5.x

`/instance/org.knime.core.data.columnar/knime.core.data.columnar.off-heap-limit=<size>`

Total off-heap memory limit in MB. Example value: `2048`.

Make sure to configure the memory settings according to the available resources on the executor. Make sure to leave enough memory for the Java heap and other processes running on the executor.

> The default configuration for 5.1.0 to 5.1.2 is to use all memory that is not reserved for the Java heap. This configuration is likely to cause the executor to crash with an Out Of Memory Exception. For the default heap size of 60% of the available memory, we recommend setting the off-heap limit to 20% of the available memory.

> On executor images before 5.1.2 and 4.7.7, an unfavorable interaction between the JVM and the glibc native memory allocator can cause higher than expected memory usage when using the Columnar Table Backend. Set the environment variable `MALLOC_ARENA_MAX=1` to prevent this issue. Also see, Columnar Backend High Memory Usage on Linux.

Proxy configuration

KNIME Analytics Platform has preferences to configure proxies platform-wide. More details on those, including the below-mentioned proxy providers, can be found here.

To distribute proxy configurations to **executors**, use the following settings in a preference (`.epf`) file.

`/instance/org.eclipse.core.net/proxiesEnabled=<true|false>`

If true, proxies are enabled. Setting this to false corresponds to using the `Direct` proxy provider.

`/instance/org.eclipse.core.net/systemProxiesEnabled=<true|false>`

If true, the `Native` proxy provider is active, otherwise the `Manual` proxy provider is active. All following preferences are only relevant in the context of manually configured proxies.

`/instance/org.eclipse.core.net/nonProxiedHosts=<pipe-separated list of excluded hosts>`
: A pipe-separated list of host names that should permanently be excluded from the manually configured proxies. It usually makes sense to set `localhost|127.0.0.1` as default here.

`/instance/org.eclipse.core.net/proxyData/<HTTP|HTTPS|SOCKS>/host=<host>`
: The host name of the proxy server.

`/instance/org.eclipse.core.net/proxyData/<HTTP|HTTPS|SOCKS>/port=<port>`
: The port of the proxy server.

`/instance/org.eclipse.core.net/proxyData/<HTTP|HTTPS|SOCKS>/hasAuth=<true|false>`
: Whether the proxy requires authentication. Proxy credentials cannot be configured in preference (`.epf`) files but have to be supplied separately, as described below.

Proxy authentication

Configured proxy credentials are persisted in the Eclipse secure storage, which by default is located within the user’s home directory. It is encrypted using a master password.

If a proxy requires authentication (i.e. setting `hasAuth=true`), the secure storage is queried for that proxy host, in conjunction with the master password. This should work out-of-the-box on Windows, macOS, and Linux. However, KNIME Hub executors need additional configuration.

1. The user’s home directory will not be resolved correctly when a KNIME Hub executor runs as system service, since it is not associated to a system user. Hence, the storage location must be configured explicitly as an application argument.
2. Usually, either a UI prompt for the master password appears on KNIME startup, or the master password is sourced relative to the user’s home directory. Both are not feasible using a headless executor running as system service. Hence, the master password must also be configured as an application argument.
3. Do not forget to restart the executor service after applying the following configuration changes.

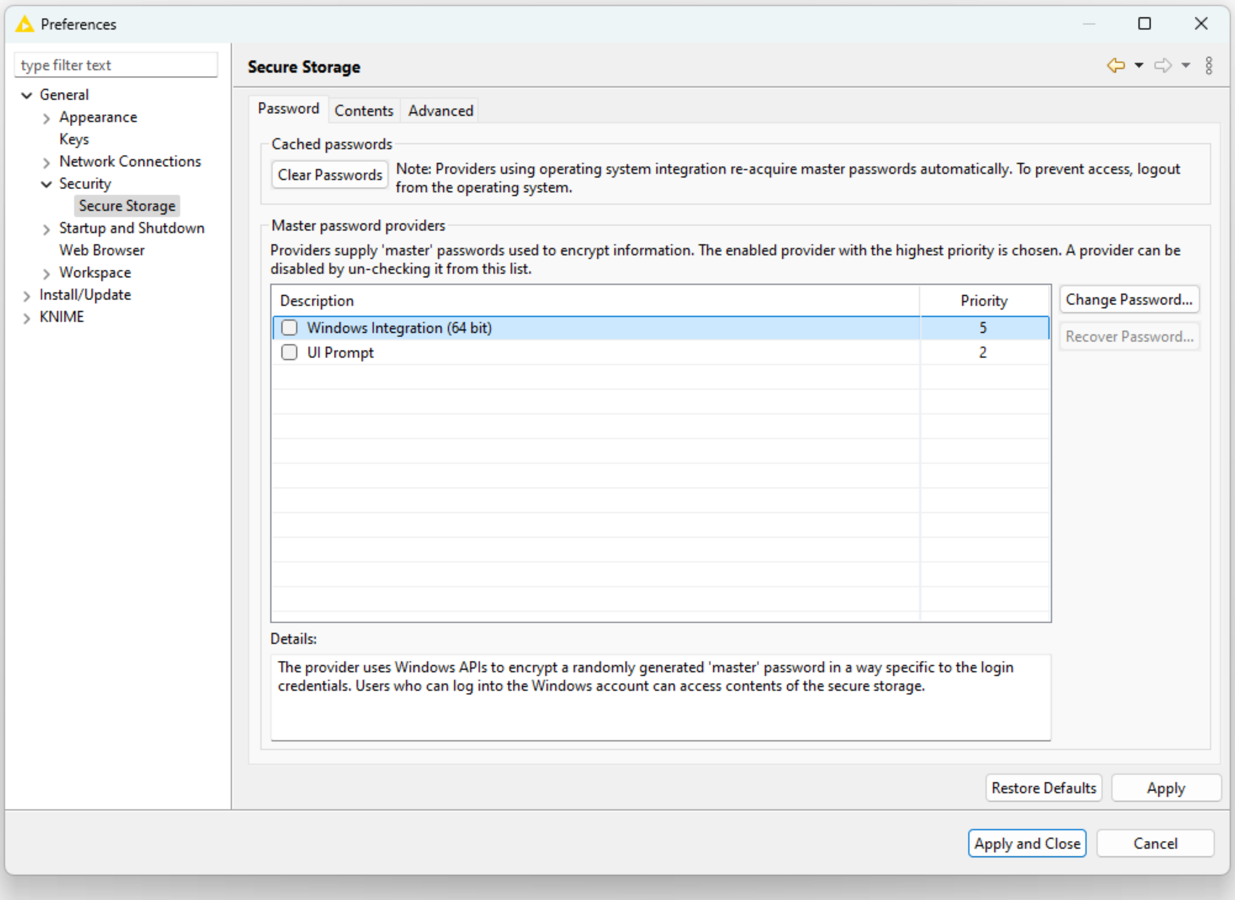
Custom secure storage location
By default, the secure storage is stored in

`~/.eclipse/org.eclipse.equinox.security/secure_storage` where `~` denotes the user's home directory. In order to configure a custom `<storage location>`, you have to set the following application argument in the execution context of the KNIME executor.

```
-eclipse.keyring
<storage location>
```

Custom secure storage password

To define a custom master password, you first need to disable active master password providers by deselecting all entries in the preferences at *General > Security > Secure Storage*. If this preferences page is not available for you in KNIME Modern UI, try switching to the preferences in the KNIME Classic UI.



Note: Providers using operating system integration re-acquire master passwords automatically. To prevent access, logout from the operating system.

Providers supply 'master' passwords used to encrypt information. The enabled provider with the highest priority is chosen. A provider can be disabled by un-checking it from this list.

Description	Priority
☐ Windows Integration (64 bit)	5
☐ UI Prompt	2

The provider uses Windows APIs to encrypt a randomly generated 'master' password in a way specific to the login credentials. Users who can log into the Windows account can access contents of the secure storage.

Figure 20. The Secure Storage preferences page.

Alternatively — without using the UI — create the file `<install-dir>/configuration/.settings/org.eclipse.equinox.security.prefs` with the following content. Here, `<install-dir>` denotes the installation directory of the KNIME Hub executor. This file is generated for you if you disable the master password providers on the preferences page.

```
eclipse.preferences.version=1
org.eclipse.equinox.security.preferences.disabledProviders=org.eclipse.equinox.security.
windowspasswordprovider64bit,org.eclipse.equinox.security.osxkeystoreintegration,org.ecl
ipse.equinox.security.linuxkeystoreintegrationjna,org.eclipse.equinox.security.ui.defaul
tpasswordprovider
```

In order to configure a custom `<password location>`, you again have to set an application argument in the execution context. At this custom location, create a text file containing nothing but your new master password.

```
-eclipse.password
<password location>
```

Further details on proxies

More useful proxy configuration options are listed here.

Environment variables as fallback
Starting from KNIME version 5.3, environment variables are supported as fallback when choosing the “Manual” or “Native” proxy provider. This is the format required for the proxy environment variables.

```
<http|https|socks|all>_proxy=<user>:<password>@<host>:<port>
```

In addition, the `no_proxy` variable defines a comma-separated list of hosts that are excluded from proxies defined in environment variables. Proxy authentication is directly specified via the user information prefix `<user>:<password>@` in the variable value.

Prior to KNIME version 5.3, environment variables were supported as part of the “Native” proxy provider on Linux systems, with the only difference being that `all_proxy` was not supported there.

Specifically on Linux, we recommend using environment variables in conjunction with the “Native” proxy provider, given that you accept a less secure storage of proxy credentials. This method is simple and independent of users and configuration directories.

Bootstrapping customization profiles behind a proxy
If the profile location of the preferences themselves are hidden behind a network proxy, preferences obviously cannot be used to define the proxy configuration. For this case, the proxy has to be configured as a VM argument in the execution context using Java networking

properties. This proxy is independent of proxy configurations mentioned above, and will not be used by the KNIME Hub executor for anything except fetching preference profiles.

- To use the equivalent of the “Manual” proxy provider, add the `-Djava.net.useSystemProxies=false` property and the required properties for your proxy configuration (see Java’s documentation).
- To use the equivalent of the “Native” proxy provider, add the `-Djava.net.useSystemProxies=true` property. Unfortunately, it is a bit unclear what Java defines as system properties. We could not verify that environment variables work on Linux with this method. The Java documentation only provides this information about system proxies.

As stated in the Java SE 17 & JDK 17 docs, on Windows systems, macOS systems, and GNOME systems it is possible to tell the `java.net` stack, setting this property to true, to use the system proxy settings (all these systems let you set proxies globally through their user interface).

Upload a customization profile

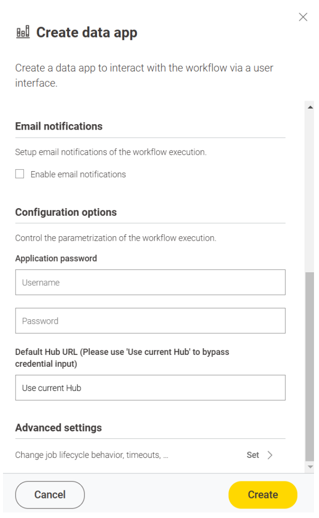
After creating a customization profile, you need to upload it to KNIME Business Hub. You can do it via REST or via data application.

Upload via Data Application

1. If you have not done so already, on KNIME Business Hub, create an application password for the user uploading the profile.
   a. For global customization profiles, use a global admin user.
   b. For team-scoped customization profiles, the user can be either a global admin or a team admin user.
2. Download the workflow:

   Click here and download the workflow.

3. Upload the downloaded workflow to your KNIME Business Hub instance.
4. Deploy the workflow as a data application. You can learn more about how to do it here. In the configuration dialog, you can set the application password and the Hub URL for the user uploading the customization profile.


Create data app

Create a data app to interact with the workflow via a user interface.

Email notifications

Setup email notifications of the workflow execution.

☐ Enable email notifications

Configuration options

Control the parametrization of the workflow execution.

Application password

Username

Password

Default Hub URL (Please use 'Use current Hub' to bypass credential input)

Use current Hub

Advanced settings

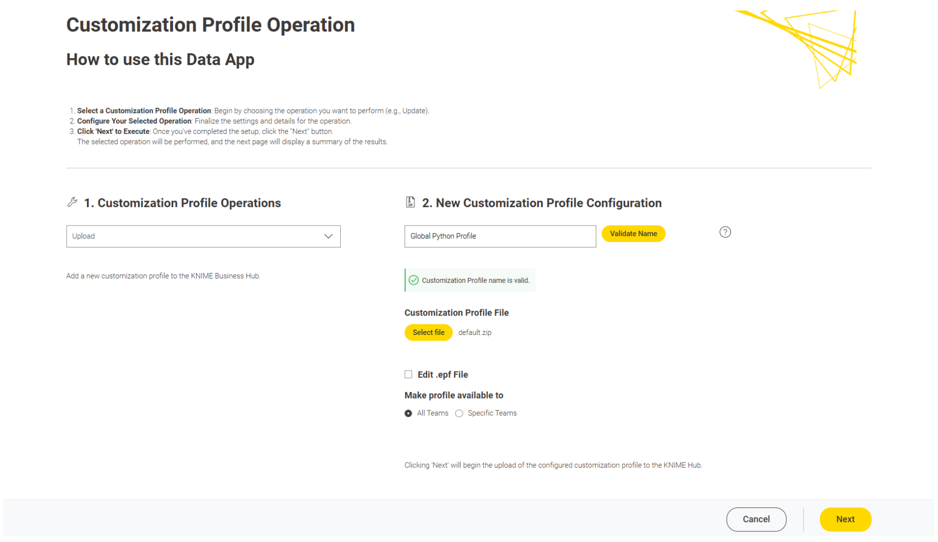
Change job lifecycle behavior, timeouts, ... Set

Cancel Create

Figure 21. Deployment configuration of the data app.

If you set the "Default Hub URL" configuration default value to "Use current Hub," user authentication on KNIME Business Hub will reflect the logged-in user's permissions.

5. Since we configured the connection directly in the deployment configuration, the data application will, by default, open on the *Customization Profile Operations* page with the *Upload* option selected.


Customization Profile Operation

How to use this Data App

1. **Select a Customization Profile Operation**: Begin by choosing the operation you want to perform (e.g., Update).
2. **Configure Your Selected Operation**: Finalize the settings and details for the operation.
3. **Click 'Next' to Execute**: Once you've completed the setup, click the "Next" button.
   The selected operation will be performed, and the next page will display a summary of the results.

1. Customization Profile Operations

Upload

Add a new customization profile to the KNIME Business Hub.

2. New Customization Profile Configuration

Global Python Profile — Validate Name

Customization Profile name is valid.

Customization Profile File

Select file default.zip

☐ Edit .epf File

Make profile available to

[✓ All Teams] Specific Teams

Clicking 'Next' will begin the upload of the configured customization profile to the KNIME Hub.

Cancel Next

Figure 22. The upload option is selected in the "Customization Profile Operations" drop-down menu.

6. In the section *New Customization Profile Configuration*, you can configure your customization profile using the following steps:
   a. **Name Your Customization Profile:** Provide a unique name for the customization profile.
      i. Use the *Validate Name* button to ensure no duplicate customization profile names exist, as uploading two with the same name is not allowed.

b. **Upload Your File:** Upload your customization profile file. Refer to the documentation for guidance on crafting a valid file.

c. **Edit .epf File (Optional):**

   i. Select the one you wish to edit if the uploaded ZIP file contains multiple `.epf` files.

   ii. You can edit content only for files with `.epf` and `.yml` extensions, as other file types are not editable within this data app.

d. **Set Customization Type:** Specify the type of customization profile:

   i. *Global Admin:* You can choose the customization profile scope if logged in to the Data App as a global admin.

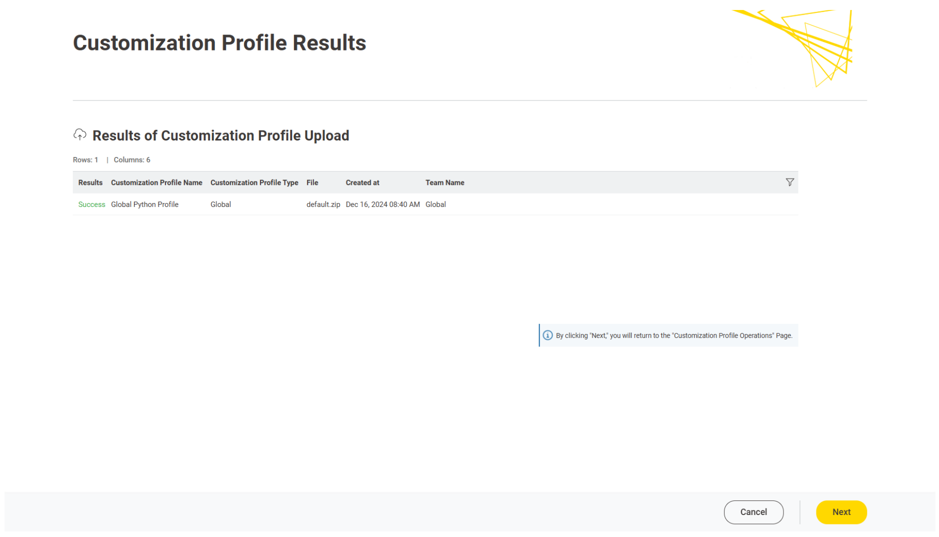
      A. Global scope: After being uploaded to KNIME Business Hub, the global customization profile can be applied to a shared or a team execution context.

      B. Team scope: When a customization profile with team scope is uploaded, it can be applied to the team's executors. It is possible to upload the same customization profile to multiple teams. The global admin can upload the customization profile to any team.

   ii. *Team Admin:* If logged in as a team admin, you can only upload the customization profile as a team-scoped type. You can upload the same customization profile for multiple teams. The availability of teams is determined by the permissions of the user who is logged in.

7. On the **Results** page, a table displays the results of the customization profile upload operation.

   If everything goes well, the column *Results* will show the value *Success* in green, indicating the operation was successful. You can now apply it to a KNIME Analytics Platform installation or to a KNIME Business Hub executor.



Customization Profile Results

Results of Customization Profile Upload

Rows: 1 | Columns: 6

Results	Customization Profile Name	Customization Profile Type	File	Created at	Team Name
Success	Global Python Profile	Global	default.zip	Dec 16, 2024 08:40 AM	Global

By clicking "Next," you will return to the "Customization Profile Operations" Page.

Figure 23. The customization profile has been uploaded successfully to KNIME Hub.

8. **Error Handling**: The data application is equipped to identify and display errors encountered during operations, whether they occur while retrieving data from the KNIME Business Hub API or executing actions via the API. Errors are captured both server-side and application-side and are displayed in the **"Error Overview"** table. This table provides a detailed summary, including:
   a. The API call status code.
   b. The error message.
   c. The specific part of the data application where the error occurred.

      Users can consult this table to diagnose issues and take appropriate corrective actions before retrying the operation.
9. Click *Next* to finish the operation.You can return to the *Customization Profile Operations* page to perform additional actions or close the data application directly.

Upload via REST request

1. If you have not done so already, on KNIME Business Hub, create an application password for the user uploading the profile.
   a. For global customization profiles, use a global admin user.
   b. For team's customization profiles, the user can be either a global admin user or a team admin user.
2. Send a `POST` request to `https://api.<base-url>/execution/customization-profiles` with the following set up:
   a. Authorization: select Basic Auth, using username and password from the created application password

b. Body: select form-data as request body type and add the following entries:

   i. Add a new key, set content to “File”. Name of the key needs to be “content”. The value of the key is the `.zip` file containing the profile, and the content type is `application/zip`.

   ii. Add a new key, set content to “Text”. Name of the key needs to be “metadata”.

      A. The value of the key is:

         For global customization profiles enter:

```
{
  "name": "<profile_name>",
  "scope": "hub:global"
}
```

         For team’s customization profiles you first need to obtain the `<team_ID>` of the team you want to upload the customization profile to. To do so you can use the following `GET` request:

```
GET api.<base-url>/accounts/identity
```

         You will find the `<team_ID>` in the body response under `teams`:

```
...
    "teams": [
        {
            "id": "account:team:<team_ID>",
            "name": "<team_name>",
            ...
        }
    ]
...
```

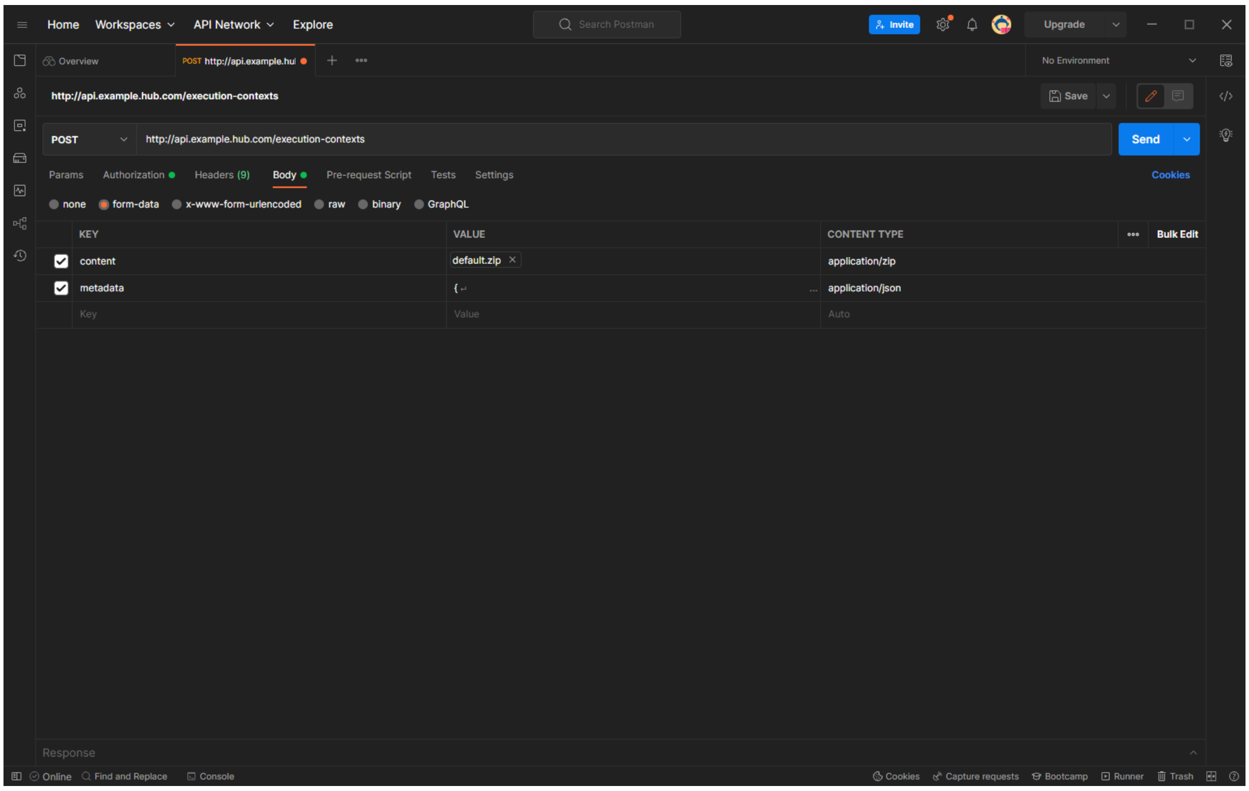
         Then the value of the key is:

```
{
  "name": "<profile_name>",
  "scope": "account:team:<team_ID>"
}
```

      B. Set the content type for the second key to `application/json`

   c. When using Postman there is no need to manually adjust the headers

3. Send the request


POST http://api.example.hub.com/execution-contexts

Body: form-data

KEY	VALUE	CONTENT TYPE
content	default.zip	application/zip
metadata	{	application/json

Figure 24. Set up Postman for a REST call to upload a customization profile to a KNIME Hub instance

4. If successful, you will receive a 201 status response. Make note of the created `<profile_ID>`, as this will be used to refer to the profile when requesting it.

> You can refer to the API documentation at the following URL for more details about the different calls you can make to the Customization Profiles endpoint.
>
> ```
> http://api.<base-url>/api-doc/execution/#/Customization%20Profiles
> ```

Apply a customization profile

Apply a customization profile to KNIME Hub executor

Apply via Data App

To apply a customization profile to all executors running in a KNIME Business Hub execution context, we can use the data application. However, the type of user logged in affects the operations that can be performed. Refer to the table below for a comprehensive overview.

Table 1. Apply customization profile: user rights

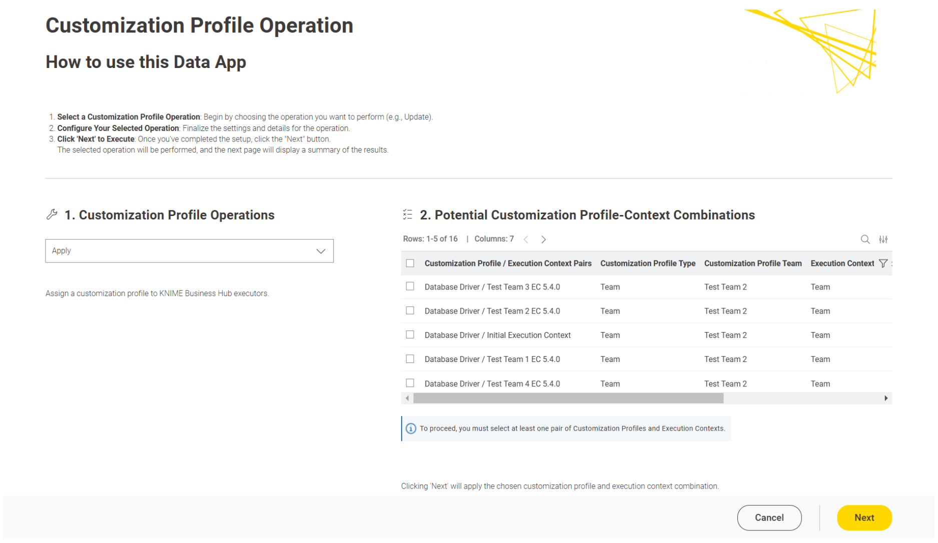
User type	Eligible customization profile	Customization profile type	Eligible Execution Context
Global admin	All uploaded within the KNIME Business Hub instance	Global	Shared and any team specific execution context
		Team-scoped	Any team execution contexts
Team admin	Only those uploaded in teams in which the user has team admin rights	Team-scoped only	Only the team execution contexts where the customization profile was uploaded. Shared execution contexts are not eligible

To provide a better understanding of the table, here are some examples to demonstrate its functionality:

- A global admin can choose one team-scoped customization profile and apply it to any team-scoped execution context within the KNIME Business Hub instance. For instance, the global admin can apply *Team A's customization profile* to the team-scoped execution context of *Team B*.
- A global admin can also select a **Global** customization profile and apply it to any shared and team-scoped execution context within the KNIME Business Hub instance.
- A team admin can **only** choose team-scoped customization profiles uploaded within the teams with admin rights. For example, they can only apply a customization profile uploaded in *Team A* within the Team A execution contexts.

1. Learn how to download the data app from Community Hub, upload and deploy it in KNIME Business Hub, and authenticate with your application password in the Upload a customization profile section.

2. Select *Apply* in the *Customization Profile Operation* menu to apply a Customization Profile.
3. The table *Potential Customization Profile-Context Combinations* displays the **possible combinations** between the customization profiles accessible to the logged-in user and the available execution contexts.#



Customization Profile Operation

How to use this Data App

1. **Select a Customization Profile Operation**: Begin by choosing the operation you want to perform (e.g., Update).
2. **Configure Your Selected Operation**: Finalize the settings and details for the operation.
3. **Click 'Next' to Execute**: Once you've completed the setup, click the "Next" button.
   The selected operation will be performed, and the next page will display a summary of the results.

1. Customization Profile Operations

Apply

Assign a customization profile to KNIME Business Hub executors.

2. Potential Customization Profile-Context Combinations

Rows: 1-5 of 16 | Columns: 7

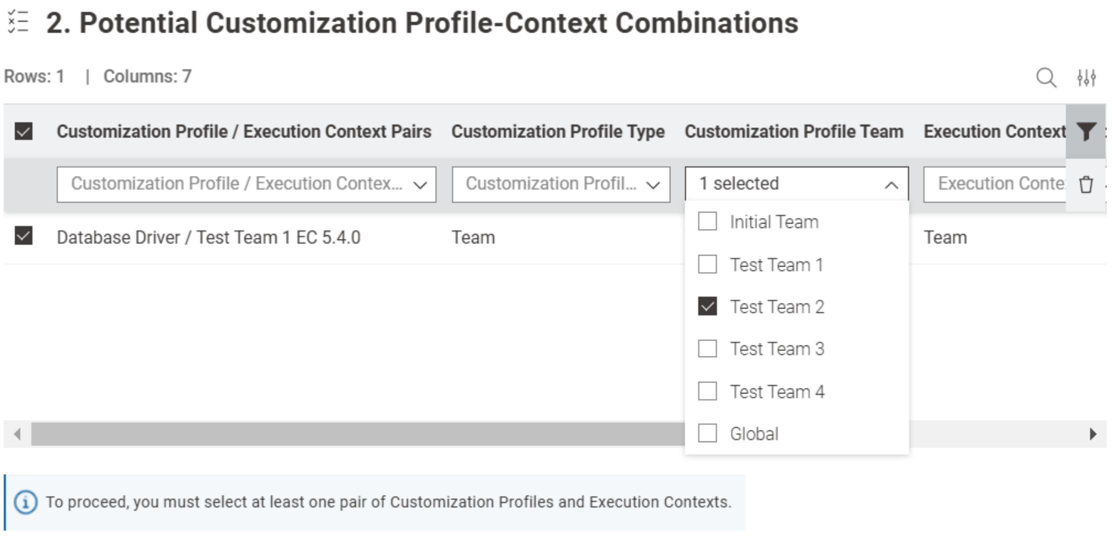
Customization Profile / Execution Context Pairs	Customization Profile Type	Customization Profile Team	Execution Context
Database Driver / Test Team 3 EC 5.4.0	Team	Test Team 2	Team
Database Driver / Test Team 2 EC 5.4.0	Team	Test Team 2	Team
Database Driver / Initial Execution Context	Team	Test Team 2	Team
Database Driver / Test Team 1 EC 5.4.0	Team	Test Team 2	Team
Database Driver / Test Team 4 EC 5.4.0	Team	Test Team 2	Team

To proceed, you must select at least one pair of Customization Profiles and Execution Contexts.

Clicking 'Next' will apply the chosen customization profile and execution context combination.

Figure 25. Multiple combinations of potential customization profile execution contexts are available for a global admin.

4. As mentioned above, a **Global admin** can apply a *team-scoped* customization profile to multiple team-scoped execution contexts or a *global customization profile* to any execution context in KNIME Hub. This may lead to **many options**, so it's advisable to use the table filters to narrow down the choices.
5. Select the desired **pair** of customization profile and execution context from the table, then click *Next*.



2. Potential Customization Profile-Context Combinations

Rows: 1 | Columns: 7

☑	Customization Profile / Execution Context Pairs	Customization Profile Type	Customization Profile Team	Execution Context
	Customization Profile / Execution Contex...	Customization Profil...	1 selected	Execution Conte
☑	Database Driver / Test Team 1 EC 5.4.0	Team		Team

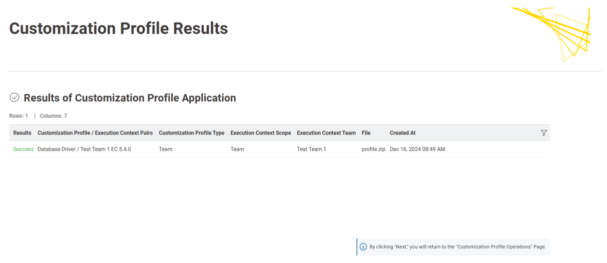
- ☐ Initial Team
- ☐ Test Team 1
- ☑ Test Team 2
- ☐ Test Team 3
- ☐ Test Team 4
- ☐ Global

To proceed, you must select at least one pair of Customization Profiles and Execution Contexts.

Figure 26. We want to apply the team-scoped customization profile from Test Team 2 to Test Team 1’s team-scoped execution context.

On the **Results** page, a table displays the results of the customization profile application operation.

If everything goes well, the column *Results* will show the value *Success* in green, indicating that your customization profile has been successfully applied to KNIME Business Hub executors for the selected execution contexts.



Customization Profile Results

Results of Customization Profile Application

Rows: 1 | Columns: 7

Results	Customization Profile / Execution Context Pairs	Customization Profile Type	Execution Context Scope	Execution Context Team	File	Created At
Success	Database Driver / Test Team 1 EC 5.4.0	Team	Team	Test Team 1	profile.zip	Dec 16, 2024 08:49 AM

By clicking "Next," you will return to the "Customization Profile Operations" Page.

Figure 27. The user applies a JDBC database driver within a team-scoped execution context.

6. **Error Handling**: The data application is equipped to identify and display errors encountered during operations, whether they occur while retrieving data from the KNIME Business Hub API or executing actions via the API. Errors are captured both server-side and application-side and are displayed in the "**Error Overview**" table. This table provides a detailed summary, including:
    a. The API call status code.
    b. The error message.
    c. The specific part of the data application where the error occurred.

    Users can consult this table to diagnose issues and take appropriate corrective actions before retrying the operation.

7. Click *Next* to finish the operation.You can return to the *Customization Profile Operations* page to perform additional actions or close the data application directly.

Apply via REST request

In order to apply a customization profile to all executors running in a KNIME Business Hub execution context, you will need to send a request to the execution context endpoint.

First you need to get the execution context ID. To do so you can use the following `GET` request to get a list of all the execution contexts that are owned by a specific team:

```
GET api.<base-url>/execution-contexts/account:team:<team_ID>
```

If you are a global admin you can also `GET` a list of all the execution contexts available on the Hub instance with the call `GET api.<base-url>/execution-contexts/`.

Now you can apply the new customization profile to the selected execution context.

You will need to obtain the `<profile_ID>` using the following `GET` request:

1. For global customization profiles:

   ```
   GET api.<base-url>/execution/customization-profiles/hub:global
   ```

2. For team's customization profiles:

   ```
   GET api.<base-url>/execution/customization-profiles/account:team:<team_ID>
   ```

   Refer to the above section to find out how to get the `<team_ID>`.

Then you need to update the execution context by using the following `PUT` request:

```
PUT api.<base-url>/execution-contexts/<execution-context_ID>
```

You need to select `Body > raw > JSON` and add the following to the request body:

```
{
"customizationProfiles": [
    "<profile_ID>"
  ]
}
```

This will cause a restart of all executor pods of this execution context, after which the profile will be applied.

At present, this will also terminate all jobs currently running on the executor.

Update a customization profile

It is possible to update customization profiles uploaded to KNIME Business Hub. This can be done even if the customization profile *has already been applied* to the KNIME Business executor via execution context. You can update two features:

1. The customization profile **name**.
2. The customization profile zip file can be overwritten. Refer to the create a customization profile section to learn how to make one.

Update via Data Application

Users can use the customization profile data app to **update** customization profiles previously uploaded to the KNIME Business Hub.

Depending on the user’s role as a global admin or a Team Admin, they can access and update specific customization profiles:

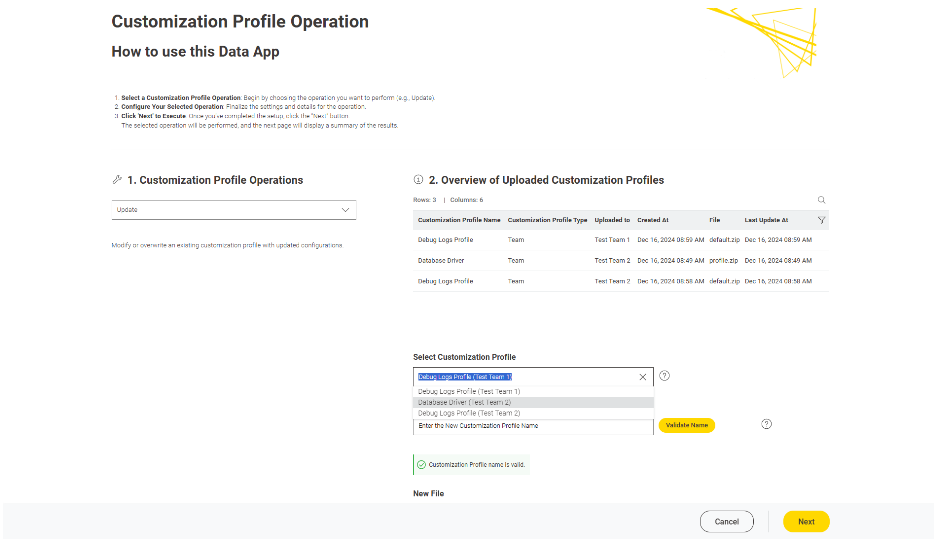
- **Global Admin:** can **update** all uploaded customization profiles within the current KNIME Business Hub instance, either team or global-scoped customization profiles.
- **Team Admin:** can only update team-scoped customization profiles from their own teams.

1. Learn how to download the data app from Community Hub, upload and deploy it in KNIME Business Hub, and authenticate with your application password in the Upload a customization profile section.
2. Select *Update* in the *Customization Profile Operations* drop-down menu to update a customization profile.

3. A table titled *Overview of Uploaded Customization Profiles* will display a list of customization profiles available to the logged-in user. The table includes details such as:
   a. The type of profile (global or team-scoped).
   b. The upload location.
   c. The creation date.
   d. The file used for the upload.

   Only one customization profile can be updated at a time.

4. After reviewing the table, select the profile you want to update from the "**Select Customization Profile**" drop-down menu. The format of the selection will appear as a `customization profile name (upload location)`.



Customization Profile Operation

How to use this Data App

1. **Select a Customization Profile Operation**: Begin by choosing the operation you want to perform (e.g., Update).
2. **Configure Your Selected Operation**: Finalize the settings and details for the operation.
3. **Click 'Next' to Execute**: Once you've completed the setup, click the "Next" button.
   The selected operation will be performed, and the next page will display a summary of the results.

1. Customization Profile Operations

Update

Modify or overwrite an existing customization profile with updated configurations.

2. Overview of Uploaded Customization Profiles

Rows: 3 | Columns: 6

Customization Profile Name	Customization Profile Type	Uploaded to	Created At	File	Last Update At
Debug Logs Profile	Team	Test Team 1	Dec 16, 2024 08:59 AM	default.zip	Dec 16, 2024 08:59 AM
Database Driver	Team	Test Team 2	Dec 16, 2024 08:49 AM	profile.zip	Dec 16, 2024 08:49 AM
Debug Logs Profile	Team	Test Team 2	Dec 16, 2024 08:58 AM	default.zip	Dec 16, 2024 08:58 AM

Select Customization Profile

Debug Logs Profile (Test Team 1)

Debug Logs Profile (Test Team 1)
Database Driver (Test Team 2)
Debug Logs Profile (Test Team 2)

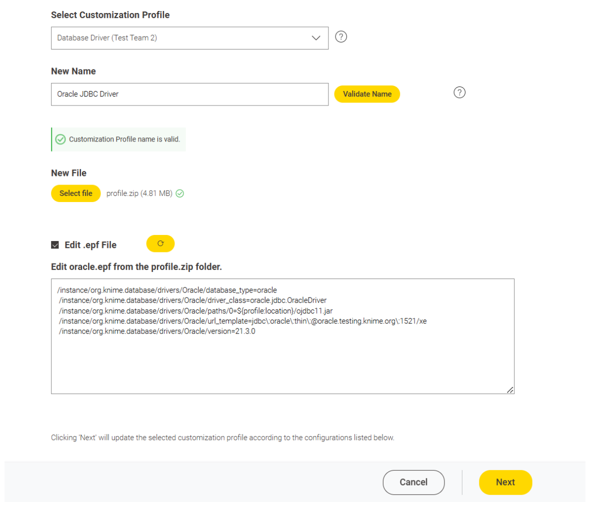
Enter the New Customization Profile Name

Customization Profile name is valid.

New File

Figure 28. A team admin could select only team-scoped customization profiles.

5. Once a profile is selected, you can:
   a. Assign a **new name** to the customization profile.
   b. Upload a **new ZIP file** for the customization profile.
   c. If you choose "**Edit .epf file**", you can directly edit the content of files with `.epf` or `.yml` extensions within the data app. Other file types cannot be edited.



Select Customization Profile

Database Driver (Test Team 2)

New Name

Oracle JDBC Driver

Customization Profile name is valid.

New File

profile.zip (4.81 MB)

Edit .epf File [✓]

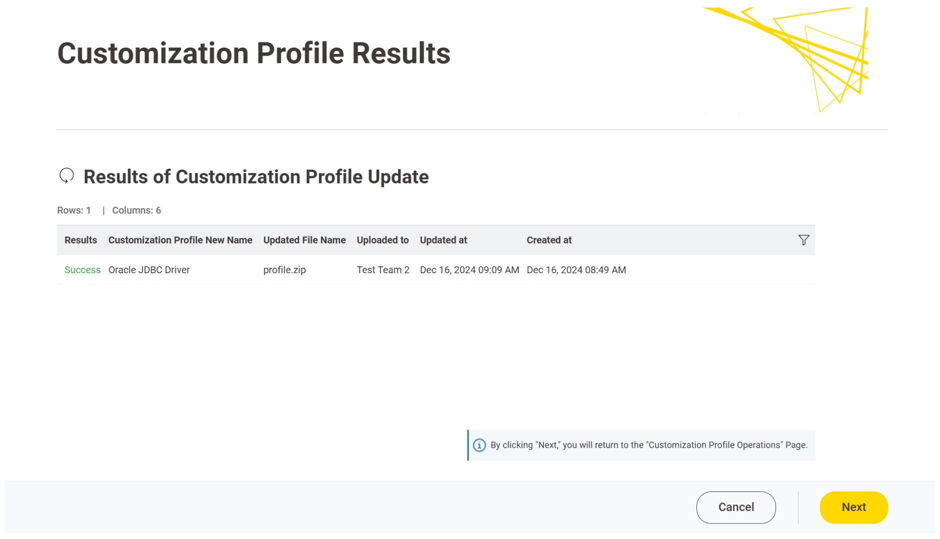
Edit oracle.epf from the profile.zip folder.

```
/instance/org.knime.database/drivers/Oracle/database_type=oracle
 /instance/org.knime.database/drivers/Oracle/driver_class=oracle.jdbc.OracleDriver
 /instance/org.knime.database/drivers/Oracle/paths/0=${profile:location}/ojdbc11.jar
 /instance/org.knime.database/drivers/Oracle/url_template=jdbc\:oracle\:thin\:@oracle.testing.knime.org\:1521/xe
 /instance/org.knime.database/drivers/Oracle/version=21.3.0
```

Clicking 'Next' will update the selected customization profile according to the configurations listed below.

Figure 29. The .epf from the zip folder file can be edited within the data app.

6. On the *Results* page, a table will display the outcome of the update operation: If successful, the "**Results**" column will show Success in green, indicating the update was applied.



Customization Profile Results

Results of Customization Profile Update

Rows: 1 | Columns: 6

Results	Customization Profile New Name	Updated File Name	Uploaded to	Updated at	Created at
Success	Oracle JDBC Driver	profile.zip	Test Team 2	Dec 16, 2024 09:09 AM	Dec 16, 2024 08:49 AM

By clicking "Next," you will return to the "Customization Profile Operations" Page.

Figure 30. The outcome of updating one customization profile.

The updated customization profiles will be **automatically applied to execution contexts or KNIME Analytics Platform clients using them.**

7. **Error Handling**: The data application is equipped to identify and display errors encountered during operations, whether they occur while retrieving data from the KNIME Business Hub API or executing actions via the API. Errors are captured both server-side and application-side and are displayed in the "**Error Overview**" table. This table provides a detailed summary, including:
   a. The API call status code.
   b. The error message.

   c. The specific part of the data application where the error occurred.

      Users can consult this table to diagnose issues and take appropriate corrective actions before retrying the operation.

8. Click *Next* to finish the operation.You can return to the *Customization Profile Operations* page to perform additional actions or close the data application directly.

Update via REST request

Updating a customization profile is like replacing an existing profile with a new file and name.

If you want to update a customization profile via REST request, the process is similar to the uploading process. The only difference is that instead of a POST Request, you need to perform a **PUT Request** and specify the ID of the customization profile.

1. If you have not done so already, on KNIME Business Hub, create an application password for the user uploading the profile.
   a. For global customization profiles, use a global admin user.
   b. For team's customization profiles, the user can be either a global admin user or a team admin user.
2. To begin, you will need to get the list of all uploaded customization profiles on KNIME Business Hub:

   You can obtain the `<profile_ID>` using the following `GET` request:

   For global customization profiles:

   ```
   GET api.<base-url>/execution/customization-profiles/hub:global
   ```

   For team's customization profiles:

   ```
   GET api.<base-url>/execution/customization-profiles/account:team:<team_ID>
   ```

   Refer to the above section to learn how to get the `<team_ID>`.

3. **Updating** a customization profile is similar to uploading it via a REST Request. However, unlike uploading, we only need to provide the name and file of the customization profile for updating. So, we don't need to provide the scope as it remains the same.
   a. Send a `PUT` request to `https://api.<base-url>/execution/customization-`

`profiles/<customization-profile-ID>` with the following set up:

b. Authorization: select Basic Auth, using username and password from the created application password

c. Body: select form-data as request body type and add the following entries:

    i. Add a new key, set content to “File”. Name of the key needs to be “content”. The value of the key is the `.zip` file containing the profile, and the content type is `application/zip`.

    ii. Add a new key, set content to “Text”. Name of the key needs to be “metadata”.

        A. The value of the key is the same for global and team-scoped customization profiles:

        ```
        {
          "name": "<profile_name>"
        }
        ```

        B. Set the content type for the second key to `application/json`

d. When using Postman there is no need to manually adjust the headers

4. Send the request

Detach a customization profile

It’s important to know that detaching a customization profile is not the same as deleting it. When you detach a customization profile, it isn’t removed from KNIME Business Hub. Instead, it just means that the customization profile won’t be applied to KNIME Analytics Platform or the KNIME Business executors.

However, the customization profile is still available in KNIME Business Hub, so it can be reused again whenever needed.

Detach a customization profile from a KNIME Business Hub executor

Detaching a customization profile applies **only** to those customization profiles applied to a KNIME Business Hub executor via execution context. Separating a customization profile from its execution context is the prerequisite step to deleting a customization profile from a KNIME Business Hub instance.

Detaching a customization profile from an execution context can also be done if the user no longer wants to apply the customization to the executors.

Detach via Data Application

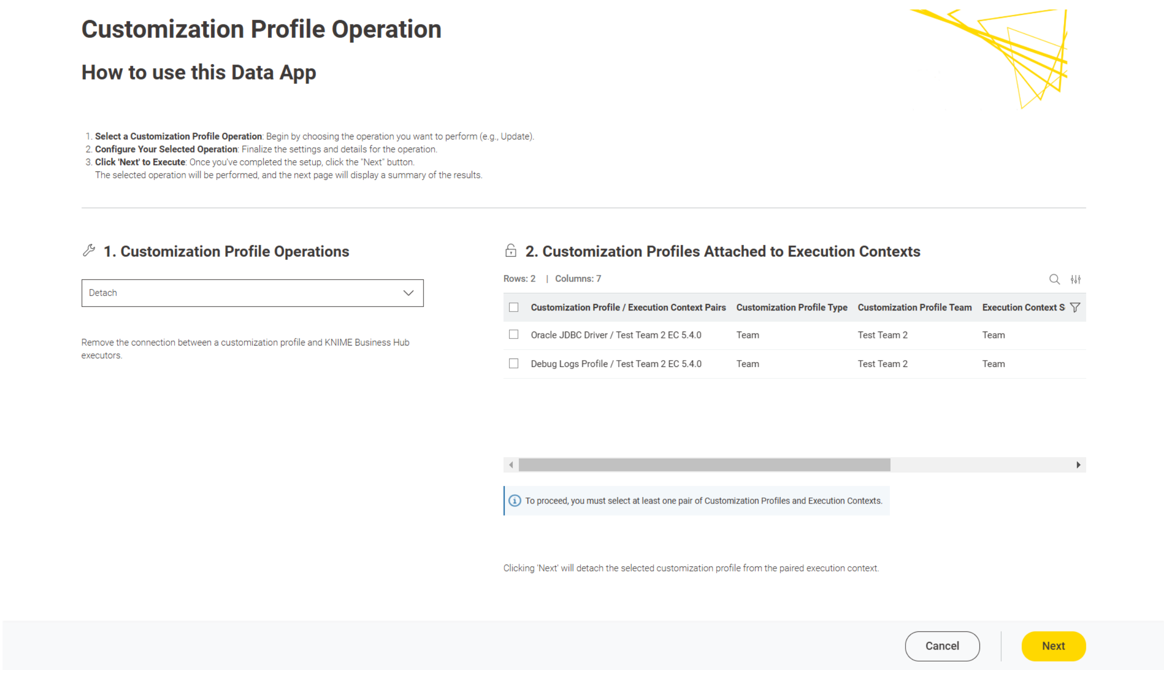
The data application allows users to **detach** customization profiles applied to execution contexts in KNIME Business Hub.

For instructions on how to apply a customization profile to an execution context, refer to this section.

The customization profiles available for detachment depend on the user type:

- **Global Admin:** can **detach** all applied customization profiles within the current KNIME Business Hub instance, either team or global-scoped customization profiles.
- **Team Admin:** can only detach team-scoped customization profiles from their own teams.

    1. Learn how to download the data app from Community Hub, upload and deploy it in KNIME Business Hub, and authenticate with your application password in the Upload a customization profile section.
    2. Select *Detach* from the *Customization Profile Operations* drop-down menu to begin the detachment process.



Customization Profile Operation

How to use this Data App

1. **Select a Customization Profile Operation**: Begin by choosing the operation you want to perform (e.g., Update).
2. **Configure Your Selected Operation**: Finalize the settings and details for the operation.
3. **Click 'Next' to Execute**: Once you've completed the setup, click the "Next" button.
   The selected operation will be performed, and the next page will display a summary of the results.

1. Customization Profile Operations

Detach

Remove the connection between a customization profile and KNIME Business Hub executors.

2. Customization Profiles Attached to Execution Contexts

Rows: 2 | Columns: 7

	Customization Profile / Execution Context Pairs	Customization Profile Type	Customization Profile Team	Execution Context S
☐	Oracle JDBC Driver / Test Team 2 EC 5.4.0	Team	Test Team 2	Team
☐	Debug Logs Profile / Test Team 2 EC 5.4.0	Team	Test Team 2	Team

To proceed, you must select at least one pair of Customization Profiles and Execution Contexts.

Clicking 'Next' will detach the selected customization profile from the paired execution context.

Figure 31. Team admin can detach customization from its teams.

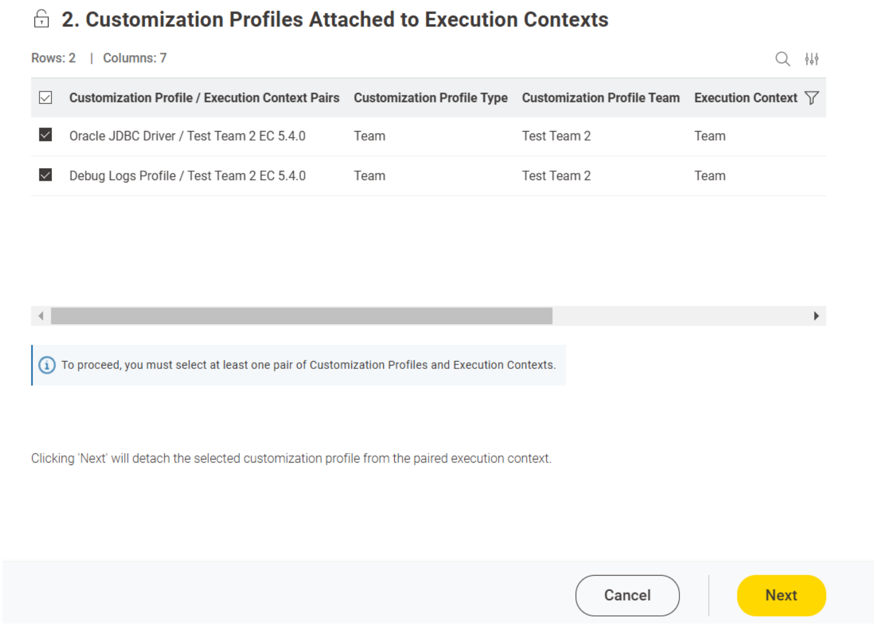
    3. Use the table titled *Customization Profiles Attached to Execution Contexts* to select the desired customization profile and execution context **pairs** you wish to detach.

a. *Global admins* may encounter a large table with all possible combinations available for detachment.

b. Use the provided **filters** to narrow the options, such as by filtering by customization profile upload location or execution context type.

> You can select multiple customization profiles for detachment.

4. Ensure you select at least one customization profile and execution context pair from the table. Once you have made a selection, click "**Next**" to proceed.



2. Customization Profiles Attached to Execution Contexts

Rows: 2 | Columns: 7

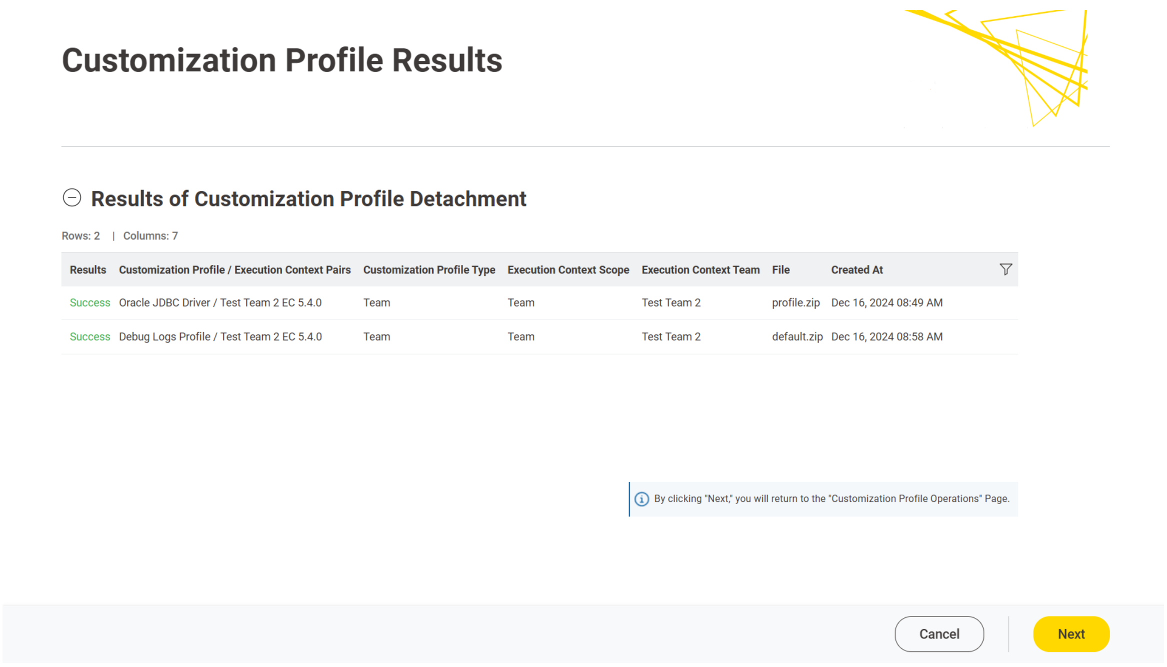
☐	Customization Profile / Execution Context Pairs	Customization Profile Type	Customization Profile Team	Execution Context
☑	Oracle JDBC Driver / Test Team 2 EC 5.4.0	Team	Test Team 2	Team
☑	Debug Logs Profile / Test Team 2 EC 5.4.0	Team	Test Team 2	Team

To proceed, you must select at least one pair of Customization Profiles and Execution Contexts.

Clicking 'Next' will detach the selected customization profile from the paired execution context.

Figure 32. Two team-scoped customization profiles will be detached.

5. On the *Results* page, a table displays the outcome of the detach operation: If successful, the "**Results**" column will show Success in green, indicating the detachment was applied.



Customization Profile Results

Results of Customization Profile Detachment

Rows: 2 | Columns: 7

Results	Customization Profile / Execution Context Pairs	Customization Profile Type	Execution Context Scope	Execution Context Team	File	Created At
Success	Oracle JDBC Driver / Test Team 2 EC 5.4.0	Team	Team	Test Team 2	profile.zip	Dec 16, 2024 08:49 AM
Success	Debug Logs Profile / Test Team 2 EC 5.4.0	Team	Team	Test Team 2	default.zip	Dec 16, 2024 08:58 AM

By clicking "Next," you will return to the "Customization Profile Operations" Page.

Figure 33. The outcome of detaching two customization profiles.

Detaching a customization profile from an execution context doesn't remove it from the Business Hub or other execution contexts using the same profile.

6. **Error Handling**: The data application is equipped to identify and display errors encountered during operations, whether they occur while retrieving data from the KNIME Business Hub API or executing actions via the API. Errors are captured both server-side and application-side and are displayed in the **"Error Overview"** table. This table provides a detailed summary, including:
    a. The API call status code.
    b. The error message.
    c. The specific part of the data application where the error occurred.

       Users can consult this table to diagnose issues and take appropriate corrective actions before retrying the operation.
7. Click *Next* to finish the operation.You can return to the *Customization Profile Operations* page to perform additional actions or close the data application directly.

Detach via REST request

You can detach a customization profile from a KNIME Business Hub execution context, which is the inverse of applying it. The steps for detaching a customization profile are similar to applying one.

To detach a customization profile from all executors running in a KNIME Business Hub execution context, you must send a request to the execution context endpoint, not including the customization profile ID that you want to detach.

1. If you have not done so already, on KNIME Business Hub, create an application password for the user uploading the profile.
   a. For global customization profiles, use a global admin user.
   b. For team's customization profiles, the user can be either a global admin user or a team admin user.
2. First, you need to get the execution context ID. To do so, you can use the following `GET` request to get a list of all the execution contexts that are owned by a specific team:

   ```
   GET api.<base-url>/execution-contexts/account:team:<team_ID>
   ```

   If you are a global admin you can also `GET` a list of all the execution contexts available on the Hub instance with the call `GET api.<base-url>/execution-contexts/`.
3. Retrieve the existing customization profiles in the execution context from the above Get Request response. Look for a key in the JSON body similar to:

   ```
   "customizationProfiles" : [ "<customization-profile-ID_1>",<customization-profile-ID_2>" ]
   ```
4. Now, you can detach the target customization profile from the selected execution context. To do so, you need to update the execution context by using the following `PUT` request:

   ```
   PUT api.<base-url>/execution-contexts/<execution-context_ID>
   ```
5. To detach a customization profile, e.g. `<customization-profile-ID_1>`, from the target execution context, follow these steps. Select `Body > raw > JSON` and ensure you do not include the customization profile you wish to remove. Use the syntax of the request body shown below:

```
{
"customizationProfiles": [
    "<customization-profile-ID_2>"
  ]
}
```

6. If the target execution context has only **one** customization profile attached, you can detach it by doing an empty request.

```
{
"customizationProfiles": []
}
```

Delete a customization profile

Deleting a customization profile from KNIME Business is a straightforward operation, and you can do it either through a REST request or the data application. Please follow the step-by-step guide below to understand how it works.

Please note that the type of user determines which customization profiles they can delete:

- **Global Admin:** can **delete** all customization profiles within the current KNIME Business Hub instance, either team or global-scoped customization profiles.
- **Team Admin:** can only **delete** team-scoped customization profiles from their own teams.

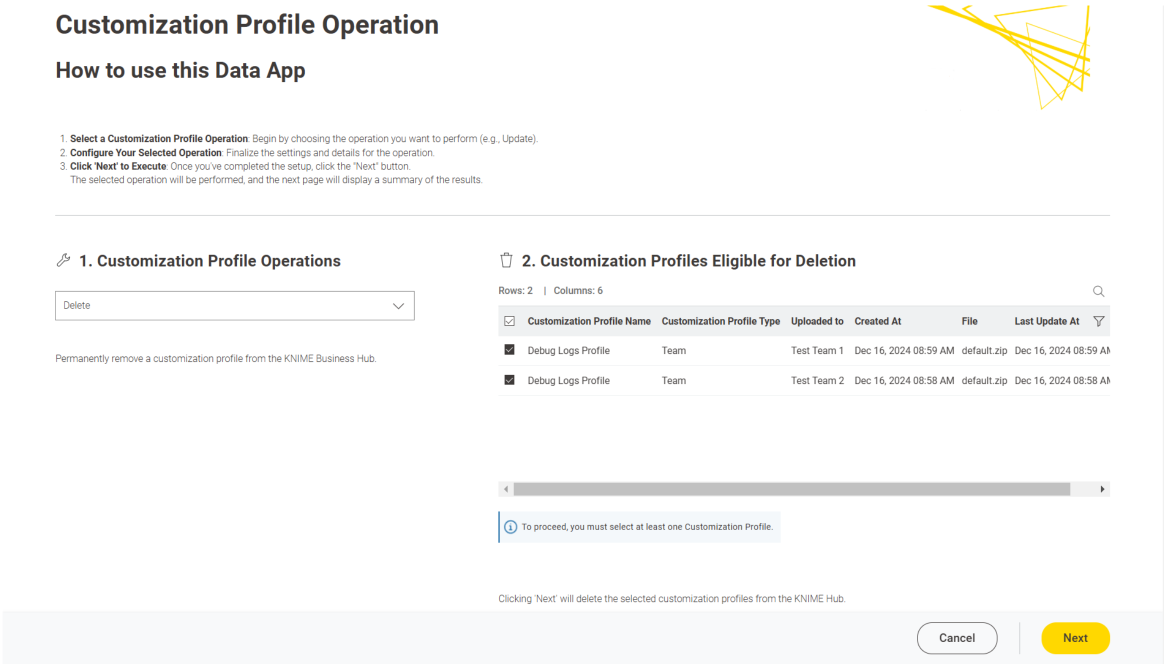
Deleting a customization profile from KNIME Business Hub requires first **detaching** it from any execution context where it was applied.

Please refer to the Detach a customization profile from a KNIME Business Hub executor section to understand how to detach a customization profile.

Delete via Data Application

1. Learn how to download the data app from Community Hub, upload and deploy it in KNIME Business Hub, and authenticate with your application password in the Upload a customization profile section.
2. Select *Delete* from the *Customization Profile Operations* drop-down menu to delete a customization profile from your KNIME Business Hub instance.

3. Review the table titled *Customization Profiles Eligible for Deletion*, which lists all customization profiles that can be deleted.
    a. Only customization profiles that are **fully detached from all execution contexts** are displayed in this table. Customization profiles still attached to execution contexts are excluded to prevent errors during the deletion process.



Customization Profile Operation

How to use this Data App

1. **Select a Customization Profile Operation**: Begin by choosing the operation you want to perform (e.g., Update).
2. **Configure Your Selected Operation**: Finalize the settings and details for the operation.
3. **Click 'Next' to Execute**: Once you've completed the setup, click the "Next" button.
   The selected operation will be performed, and the next page will display a summary of the results.

1. Customization Profile Operations

Delete

Permanently remove a customization profile from the KNIME Business Hub.

2. Customization Profiles Eligible for Deletion

Rows: 2 | Columns: 6

☑	Customization Profile Name	Customization Profile Type	Uploaded to	Created At	File	Last Update At
☑	Debug Logs Profile	Team	Test Team 1	Dec 16, 2024 08:59 AM	default.zip	Dec 16, 2024 08:59 AM
☑	Debug Logs Profile	Team	Test Team 2	Dec 16, 2024 08:58 AM	default.zip	Dec 16, 2024 08:58 AM

To proceed, you must select at least one Customization Profile.

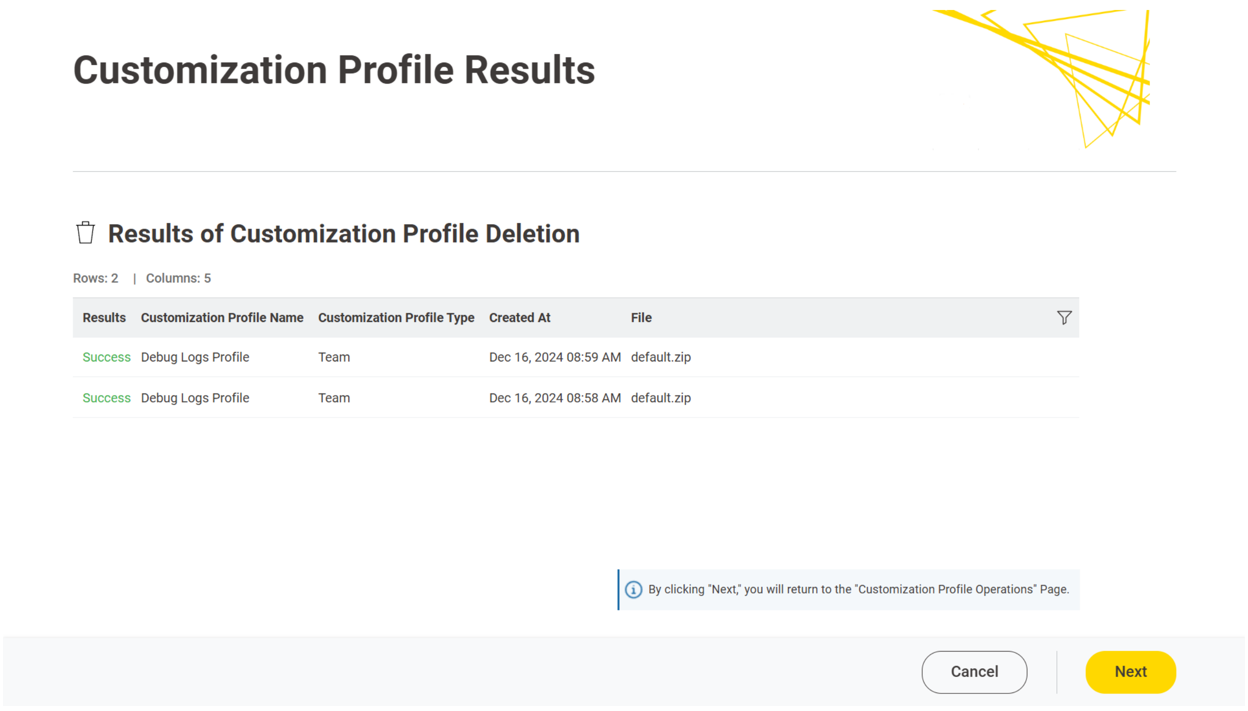
Clicking 'Next' will delete the selected customization profiles from the KNIME Hub.

Figure 34. Team admins can only delete customization profiles from teams they are admin of.

4. Select the customization profiles you want to delete from the table and click **"Next."**
    a. You can choose to delete multiple customization profiles at the same time.

Clicking **"Next"** will immediately delete the selected customization profiles from KNIME Hub. Ensure your selection is accurate before proceeding, as this action cannot be undone.

5. On the *Results* page, a table displays the outcome of the delete operation: If successful, the **"Results"** column will show Success in green, indicating the deletion was applied.


Customization Profile Results

Results of Customization Profile Deletion

Rows: 2 | Columns: 5

Results	Customization Profile Name	Customization Profile Type	Created At	File
Success	Debug Logs Profile	Team	Dec 16, 2024 08:59 AM	default.zip
Success	Debug Logs Profile	Team	Dec 16, 2024 08:58 AM	default.zip

By clicking "Next," you will return to the "Customization Profile Operations" Page.

Figure 35. The customization profile deletion from KNIME Business Hub was successful.

6. **Error Handling**: The data application is equipped to identify and display errors encountered during operations, whether they occur while retrieving data from the KNIME Business Hub API or executing actions via the API. Errors are captured both server-side and application-side and are displayed in the **"Error Overview"** table. This table provides a detailed summary, including:
   a. The API call status code.
   b. The error message.
   c. The specific part of the data application where the error occurred.

      Users can consult this table to diagnose issues and take appropriate corrective actions before retrying the operation.
7. Click *Next* to finish the operation.You can return to the *Customization Profile Operations* page to perform additional actions or close the data application directly.

Delete via REST request

Deleting a customization profile from KNIME Business Hub is possible via a REST request.

Below are the steps to accomplish this:

1. If you have not done so already, on KNIME Business Hub, create an application password for the user uploading the profile.

   a. For global customization profiles, use a global admin user.
   b. For team’s customization profiles, the user can be either a global admin user or a team admin user.

2. To start, you need to retrieve the list of the uploaded customization profiles to KNIME Business Hub:

   You can obtain the `<customization_profile_ID>` using the following `GET` request:

   For global customization profiles:

   ```
   GET api.<base-url>/execution/customization-profiles/hub:global
   ```

   For team’s customization profiles:

   ```
   GET api.<base-url>/execution/customization-profiles/account:team:<team_ID>
   ```

   Refer to the above section to find out how to get the `<team_ID>`.

3. Once you have the `<customization-profile-ID>` that you want to delete, perform a DELETE Request.

   a. Send a `DELETE` request to `https://api.<base-url>/execution/customization-profiles/<customization-profile-ID>` with the following set up:

      i. Authorization: select Basic Auth, using username and password from the created application password

         After a successful deletion, you will receive a 20* status code.

Advanced configuration

This section covers some of the configuration settings that are available for your KNIME Business Hub instance.

The following configurations are available in the KOTS Admin Console and can be changed after the installation and first minimal configuration steps are concluded successfully.

You can access the KOTS Admin Console via the URL and password you are provided in the output upon installation.

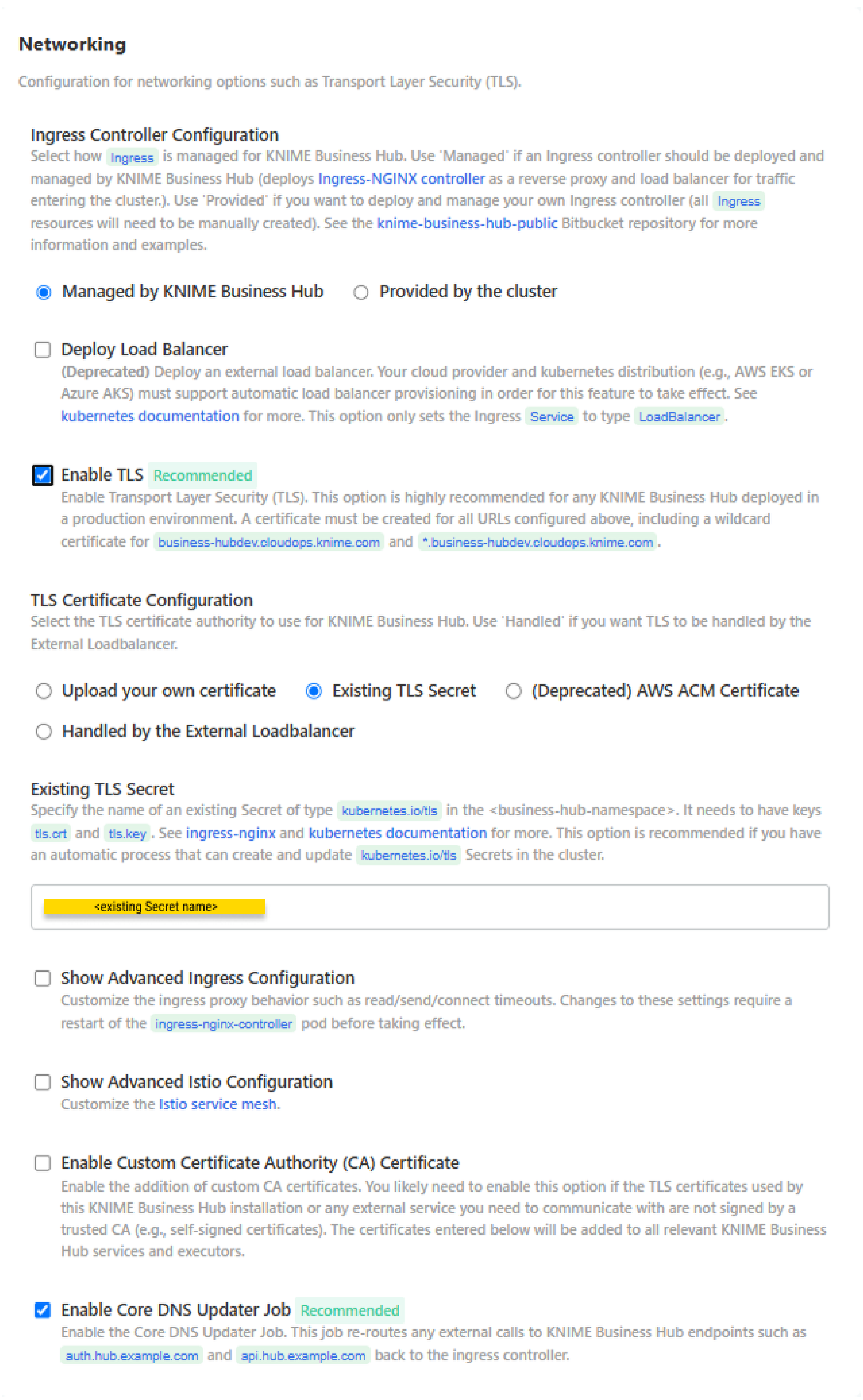
Configure networking

In the "Networking" section of the KOTS Admin Console you can:

- Deploy an external load balancer for traffic ingress: this feature takes effect only if your cloud provider and kubernetes distribution support automatic load balancer provisioning.
- Enable Transport Layer Security (TLS): the encryption protocol that provides communications security is highly recommended especially for KNIME Business Hub instances deployed in a production environment.

  If TLS is not enabled some HTTPS-only browser’s features will not be available. For example, it will not be possible for a user to copy generated application passwords.

- Enable advanced ingress configuration: you can customize the ingress proxy behavior, for example configuring the read/send/connect timeouts.
- Enable Custom Certificate Authority (CA) Certificate: Add custom CA certificates. You likely need to enable this option if the TLS certificates used by this KNIME Business Hub installation or any external service you need to communicate with are not signed by a trusted CA, for example self-signed certificates. The certificates entered will be added to all relevant KNIME Business Hub services and executors.



Networking

Configuration for networking options such as Transport Layer Security (TLS).

Ingress Controller Configuration

Select how `Ingress` is managed for KNIME Business Hub. Use 'Managed' if an Ingress controller should be deployed and managed by KNIME Business Hub (deploys Ingress-NGINX controller as a reverse proxy and load balancer for traffic entering the cluster.). Use 'Provided' if you want to deploy and manage your own Ingress controller (all `Ingress` resources will need to be manually created). See the knime-business-hub-public Bitbucket repository for more information and examples.

[✓ Managed by KNIME Business Hub] Provided by the cluster

☐ Deploy Load Balancer
(Deprecated) Deploy an external load balancer. Your cloud provider and kubernetes distribution (e.g., AWS EKS or Azure AKS) must support automatic load balancer provisioning in order for this feature to take effect. See kubernetes documentation for more. This option only sets the Ingress `Service` to type `LoadBalancer`.

☑ Enable TLS Recommended
Enable Transport Layer Security (TLS). This option is highly recommended for any KNIME Business Hub deployed in a production environment. A certificate must be created for all URLs configured above, including a wildcard certificate for `business-hubdev.cloudops.knime.com` and `*.business-hubdev.cloudops.knime.com`.

TLS Certificate Configuration

Select the TLS certificate authority to use for KNIME Business Hub. Use 'Handled' if you want TLS to be handled by the External Loadbalancer.

Upload your own certificate [✓ Existing TLS Secret] (Deprecated) AWS ACM Certificate Handled by the External Loadbalancer

Existing TLS Secret

Specify the name of an existing Secret of type `kubernetes.io/tls` in the <business-hub-namespace>. It needs to have keys `tls.crt` and `tls.key`. See ingress-nginx and kubernetes documentation for more. This option is recommended if you have an automatic process that can create and update `kubernetes.io/tls` Secrets in the cluster.

<existing Secret name>

☐ Show Advanced Ingress Configuration
Customize the ingress proxy behavior such as read/send/connect timeouts. Changes to these settings require a restart of the `ingress-nginx-controller` pod before taking effect.

☐ Show Advanced Istio Configuration
Customize the Istio service mesh.

☐ Enable Custom Certificate Authority (CA) Certificate
Enable the addition of custom CA certificates. You likely need to enable this option if the TLS certificates used by this KNIME Business Hub installation or any external service you need to communicate with are not signed by a trusted CA (e.g., self-signed certificates). The certificates entered below will be added to all relevant KNIME Business Hub services and executors.

☑ Enable Core DNS Updater Job Recommended
Enable the Core DNS Updater Job. This job re-routes any external calls to KNIME Business Hub endpoints such as `auth.hub.example.com` and `api.hub.example.com` back to the ingress controller.

Configure TLS

If you enable the Transport Layer Security (TLS) you need to have a certificate that is valid for all the URLs defined during the installation. We recommend to create a wildcard certificate for `<base-url>` and `*.<base-url>`, e.g. `hub.example.com` and `*.hub.example.com`.

Check *Enable TLS* in the "Networking" section of the KOTS Admin Console.

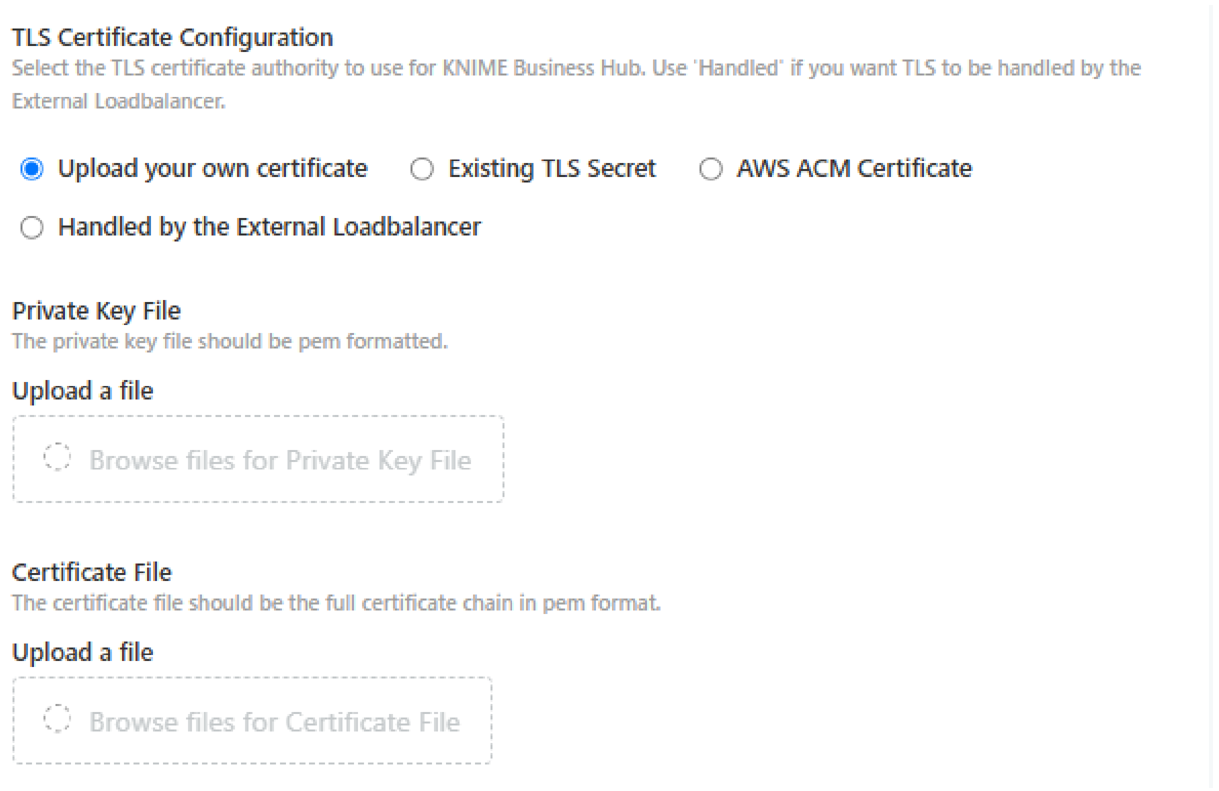
You can choose between four options to configure the TLS certificate:

- **Upload your own certificate**: Select *Upload your own certificate* to upload the certificate files. You can usually get a certificate from your company's IT department or Certificate Authority (CA). Another possibility, if you have a public domain name, is to use letsencrypt to obtain a certificate.
  Notice that if the certificates added here are signed by private or unknown CA then you need to also enable the Enable Custom Certificate Authority (CA) Certificate option and add the certificates of the CA in the *Custom Certificate Authority (CA) Certificate* field.

  > The certificates need to be `PEM` formatted as requested by the `ingress-nginx-controller` (see the relevant documentation here).

  You will need:

  - An **unencrypted** private key file
  - A certificate file in `PEM` format that contains the **full** certificate chain.
    In the certificate chain, it is important that the certificates are ordered top to bottom as follows:
    1. Server Certificate
    2. Intermediate Certificates
    3. Root Certificate


TLS Certificate Configuration
Select the TLS certificate authority to use for KNIME Business Hub. Use 'Handled' if you want TLS to be handled by the External Loadbalancer.

[✓ Upload your own certificate] Existing TLS Secret AWS ACM Certificate
Handled by the External Loadbalancer

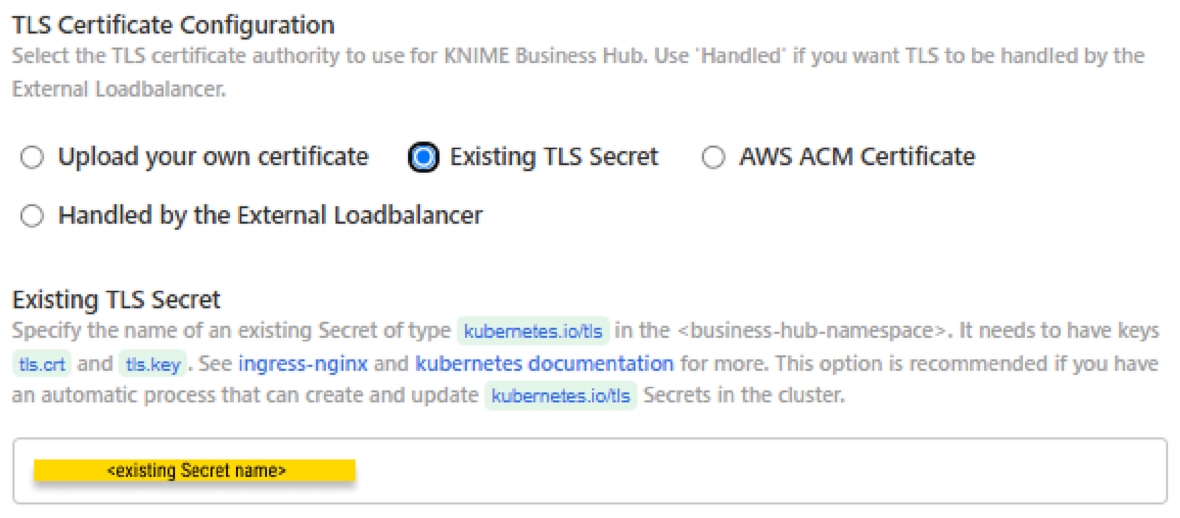
Private Key File
The private key file should be pem formatted.
Upload a file
Browse files for Private Key File

Certificate File
The certificate file should be the full certificate chain in pem format.
Upload a file
Browse files for Certificate File

- **Existing TLS Secret**: Select *Existing TLS Secret* to specify the name of an existing Secret of type `kubernetes.io/tls` in the `<business-hub-namespace>` namespace. It needs to have keys `tls.crt` and `tls.key`, which contain the PEM formatted private key and full chain certificate.

  This option is recommended if you have an automatic process that can create and renew `kubernetes.io/tls` Secrets in the cluster, like the cert-manager project.

  See ingress-nginx and kubernetes documentation on TLS secrets for more details.

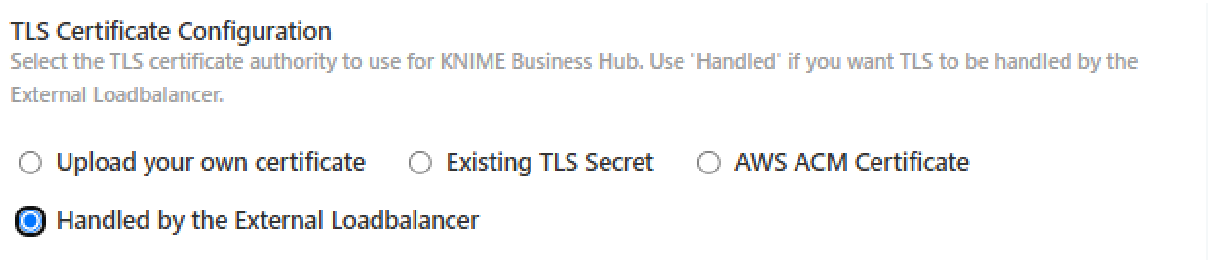

TLS Certificate Configuration
Select the TLS certificate authority to use for KNIME Business Hub. Use 'Handled' if you want TLS to be handled by the External Loadbalancer.

Upload your own certificate [✓ Existing TLS Secret] AWS ACM Certificate
Handled by the External Loadbalancer

Existing TLS Secret
Specify the name of an existing Secret of type kubernetes.io/tls in the <business-hub-namespace>. It needs to have keys tls.crt and tls.key. See ingress-nginx and kubernetes documentation for more. This option is recommended if you have an automatic process that can create and update kubernetes.io/tls Secrets in the cluster.

<existing Secret name>

- **Handled by the External Loadbalancer**: Select *Handled by the External Loadbalancer* if you have an external load balancer that handles the TLS termination. In this case, the load balancer will be responsible for the TLS configuration.



TLS Certificate Configuration

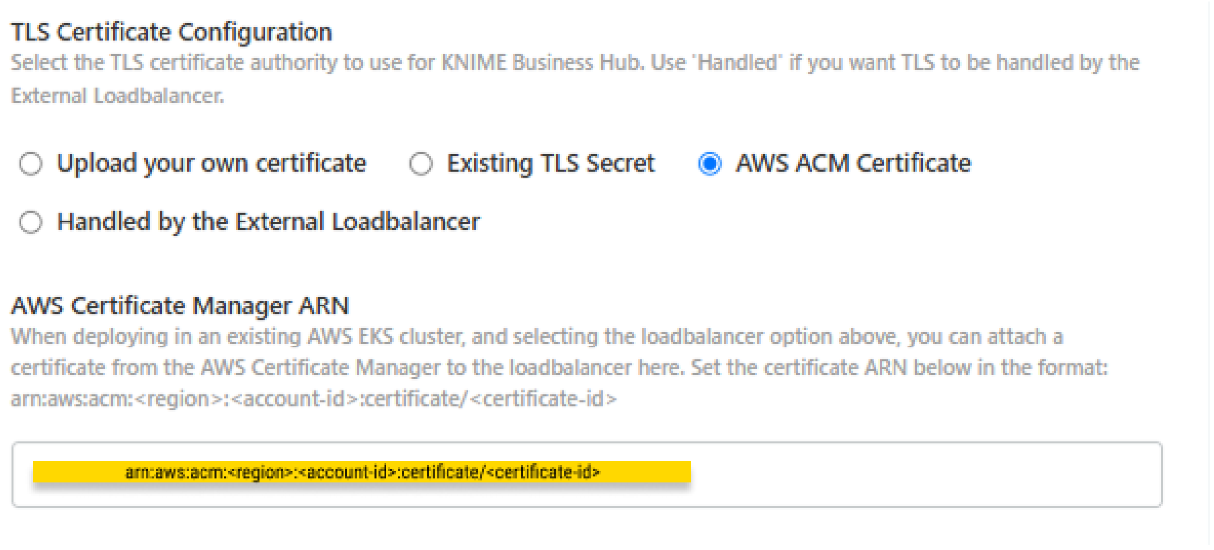
Select the TLS certificate authority to use for KNIME Business Hub. Use 'Handled' if you want TLS to be handled by the External Loadbalancer.

○ Upload your own certificate ○ Existing TLS Secret ○ AWS ACM Certificate [✓ Handled by the External Loadbalancer]

- **AWS ACM Certificate (Deprecated)**: Select *AWS ACM Certificate* if, instead, you have deployed an AWS Elastic Load Balancer (ELB).
  Notice that deploying an external load balancer through Business Hub is not recommended. Instead, you should deploy `ingress-nginx` independently. As a result, the *AWS ACM Certificate* field should no longer be used. If an external load balancer or proxy is handling TLS termination, use the Handled by the External Load Balancer option instead.

  In case you still need to use this option you can use AWS Certificate Manager (ACM) and set the certificate as an annotation directly on the load balancer. You can find more information in AWS documentation for ACM here.

  Once you obtained the certificate Amazon Resource Name (ARN) in the form `arn:aws:acm:<region>:<account-id>:certificate/<certificate-id>`, insert the ARN in the corresponding field as shown in the image below.



TLS Certificate Configuration

Select the TLS certificate authority to use for KNIME Business Hub. Use 'Handled' if you want TLS to be handled by the External Loadbalancer.

○ Upload your own certificate ○ Existing TLS Secret [✓ AWS ACM Certificate] ○ Handled by the External Loadbalancer

AWS Certificate Manager ARN

When deploying in an existing AWS EKS cluster, and selecting the loadbalancer option above, you can attach a certificate from the AWS Certificate Manager to the loadbalancer here. Set the certificate ARN below in the format: arn:aws:acm:<region>:<account-id>:certificate/<certificate-id>

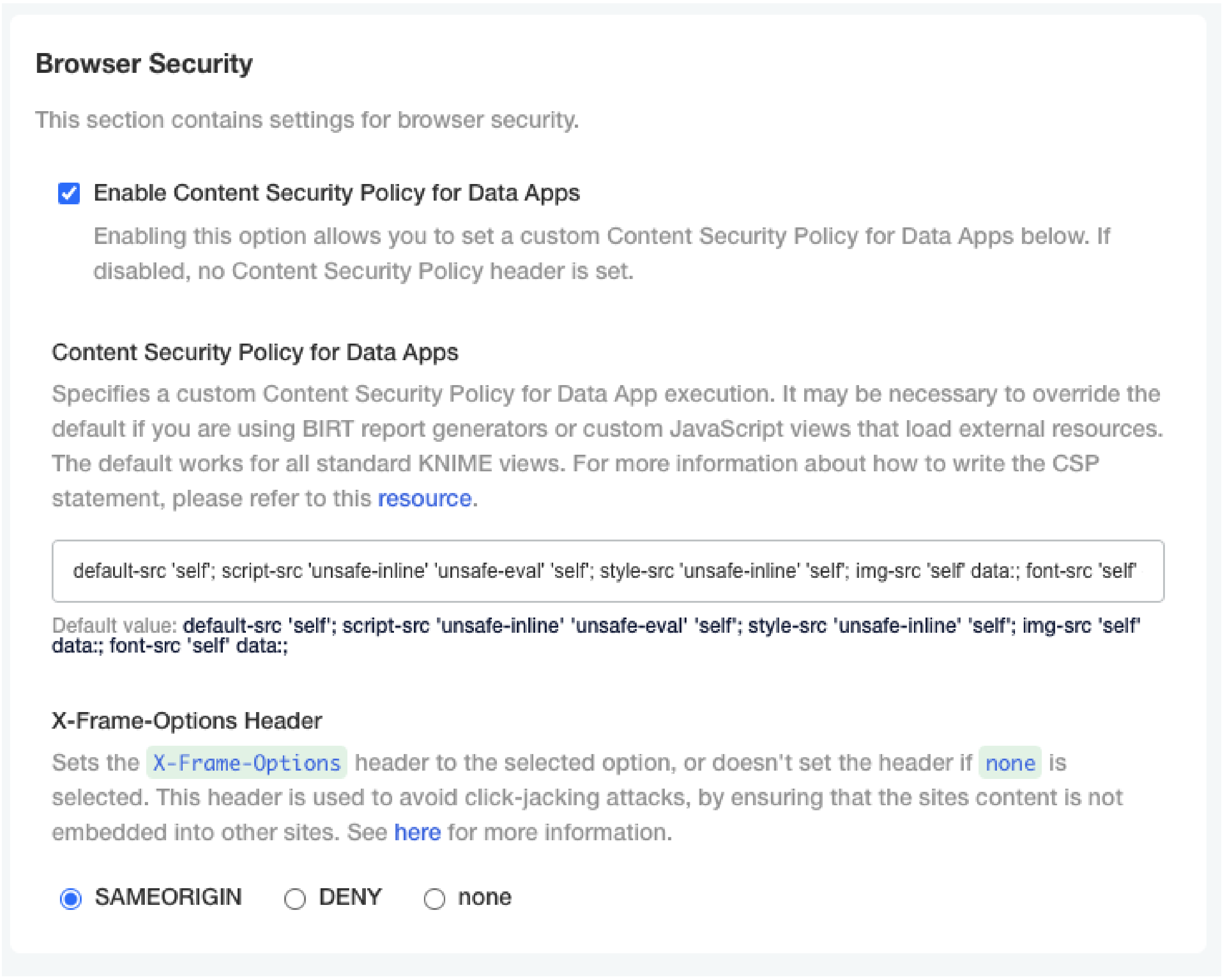
arn:aws:acm:<region>:<account-id>:certificate/<certificate-id>

Configure Browser Security

In the "Browser Security" section of the KOTS Admin Console you can:

- Specify a custom Content Security Policy for Data App execution. It may be necessary to override the default if you are using custom JavaScript views that load external resources. The default works for all standard KNIME views. For more information about how to write the CSP statement, please refer to this resource.
- Configure the X-Frame-Options header being set by webapps. This header is used to avoid click-jacking attacks, by ensuring that the sites content is not embedded into other sites. See here for more information.

> To be able to embedd data apps in an iframe, the *X-Frame-Options* must be set to `none`. Further, make sure that no other part in the infrastructure sets a CSP frame-ancestors rule or make sure it includes the embedding website. Setting frame-ancestors basically overwrites `X-Frame-Options`.


Browser Security

This section contains settings for browser security.

[✓] Enable Content Security Policy for Data Apps

Enabling this option allows you to set a custom Content Security Policy for Data Apps below. If disabled, no Content Security Policy header is set.

Content Security Policy for Data Apps

Specifies a custom Content Security Policy for Data App execution. It may be necessary to override the default if you are using BIRT report generators or custom JavaScript views that load external resources. The default works for all standard KNIME views. For more information about how to write the CSP statement, please refer to this resource.

```
default-src 'self'; script-src 'unsafe-inline' 'unsafe-eval' 'self'; style-src 'unsafe-inline' 'self'; img-src 'self' data:; font-src 'self'
```

Default value: default-src 'self'; script-src 'unsafe-inline' 'unsafe-eval' 'self'; style-src 'unsafe-inline' 'self'; img-src 'self' data:; font-src 'self' data:;

X-Frame-Options Header

Sets the `X-Frame-Options` header to the selected option, or doesn't set the header if `none` is selected. This header is used to avoid click-jacking attacks, by ensuring that the sites content is not embedded into other sites. See here for more information.

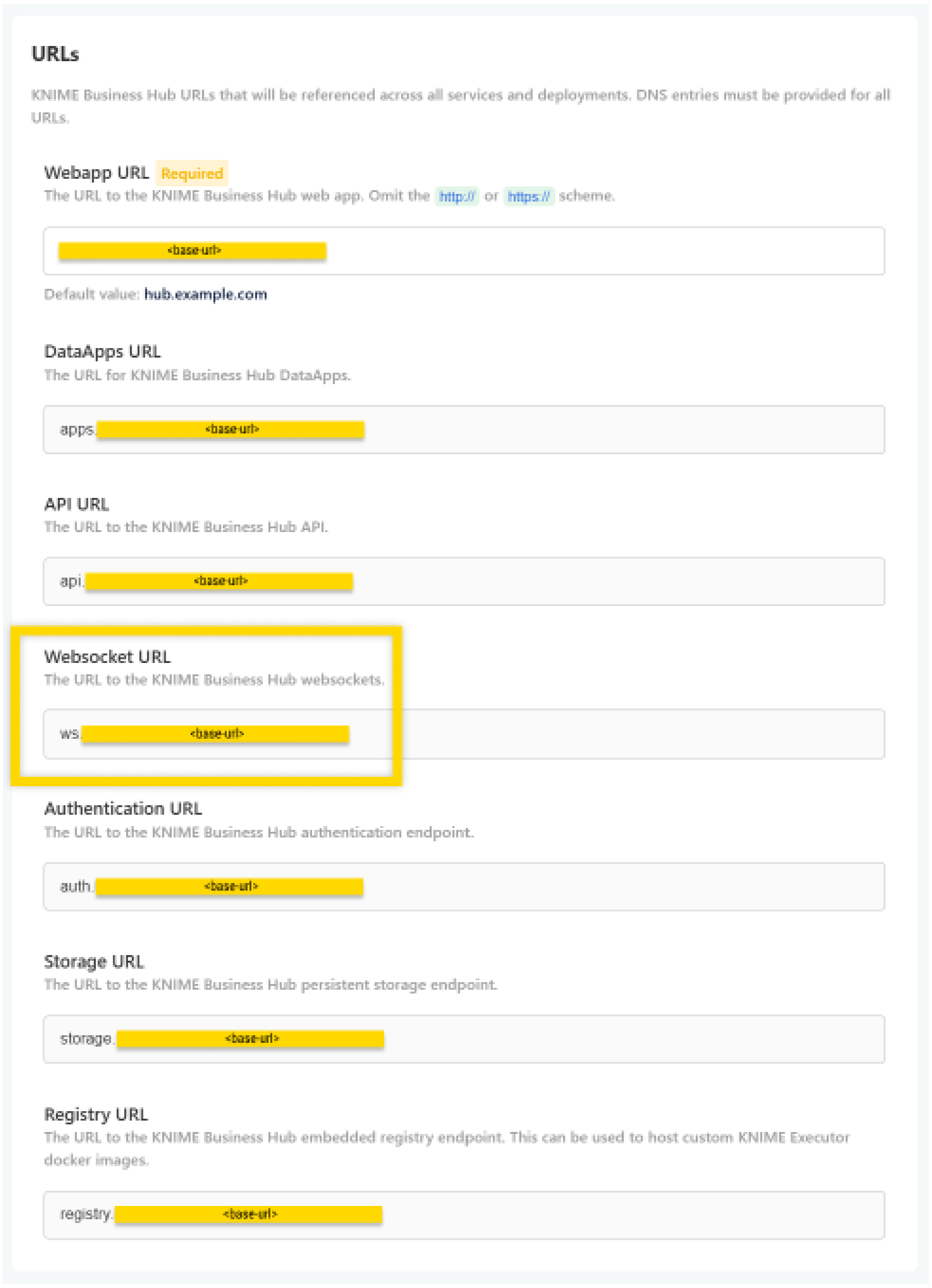
[✓ SAMEORIGIN] DENY none

Job viewer feature configuration

With the job viewer users are able to inspect a job at its current state in a browser based read only mode, without the need to open their local KNIME Analytics Platform.

In order for this feature to be available you will need:

- **KNIME Analytics Platform 5.2.3** or newer to be used as executor
- Load balancers / proxies in front of Business Hub need to be **websocket compatible**.
- Additionally, make sure that the DNS entry of the Websocket URL (`ws.<base-url>`) is set up correctly. You can find the Websocket URL in the KOTS Admin Console under *URLs > Websocket URL*.



URLs

KNIME Business Hub URLs that will be referenced across all services and deployments. DNS entries must be provided for all URLs.

Webapp URL Required
The URL to the KNIME Business Hub web app. Omit the http:// or https:// scheme.

<base-url>

Default value: hub.example.com

DataApps URL
The URL for KNIME Business Hub DataApps.

apps.<base-url>

API URL
The URL to the KNIME Business Hub API.

api.<base-url>

Websocket URL
The URL to the KNIME Business Hub websockets.

ws.<base-url>

Authentication URL
The URL to the KNIME Business Hub authentication endpoint.

auth.<base-url>

Storage URL
The URL to the KNIME Business Hub persistent storage endpoint.

storage.<base-url>

Registry URL
The URL to the KNIME Business Hub embedded registry endpoint. This can be used to host custom KNIME Executor docker images.

registry.<base-url>

Configure workflow upload behavior

In the *KOTS Admin Console*, under the *Config* tab in the *Item Upload* section, you can configure how workflows are uploaded from your local KNIME Analytics Platform to a KNIME Business Hub instance.

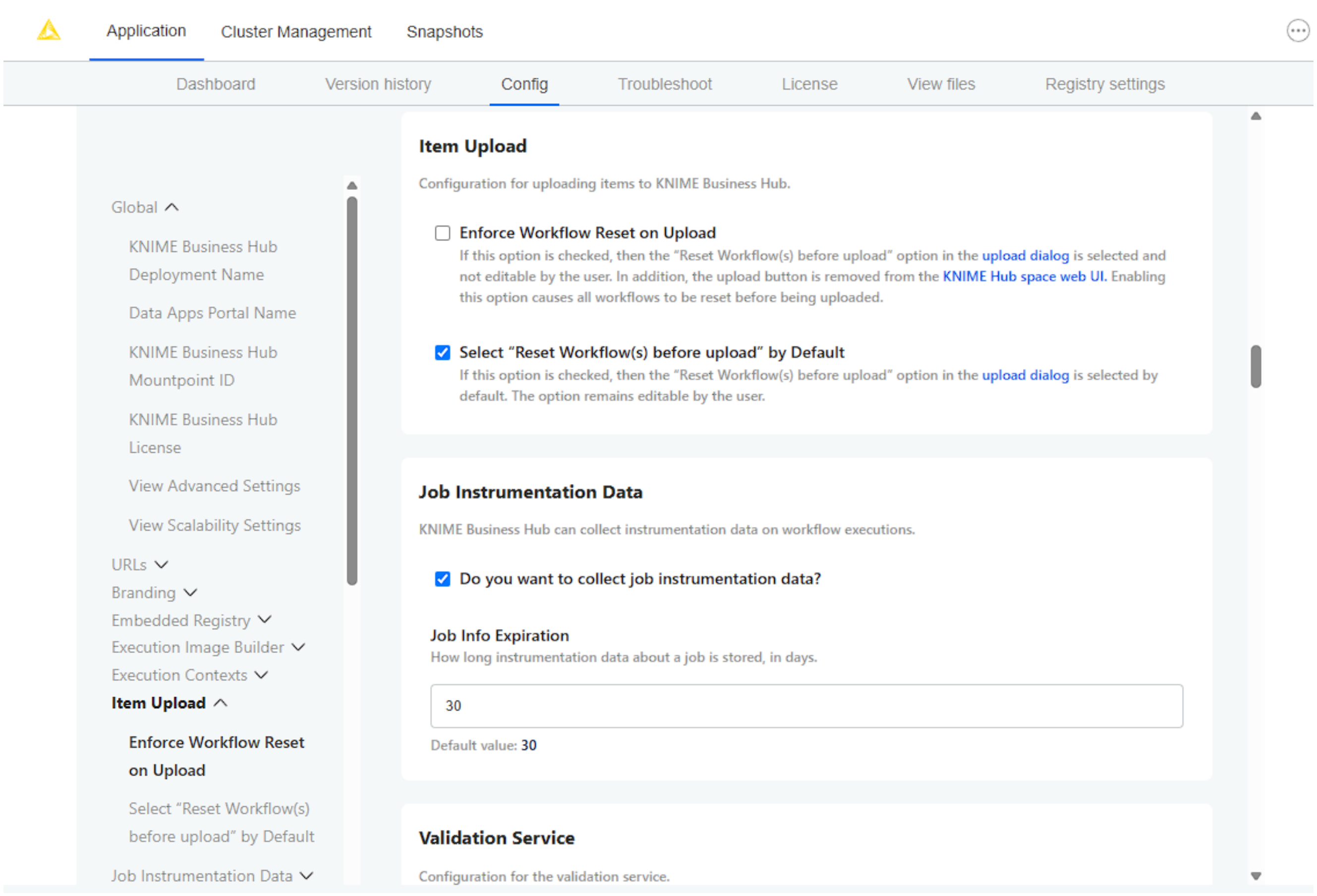
Two settings are available:

- *Enforce Workflow Reset on Upload*: Forces all workflows to be reset before upload. This cannot be changed by users in the upload dialog.
- *Select “Reset Workflow(s) before upload” by Default*: Preselects the reset option in the upload dialog, but still allows users to change it.

Depending on how you combine these settings, you will create one of the following experiences for users:

Enforce Workflow Reset on Upload	*Select “Reset Workflow(s) before upload” by Default*	**User Experience**	**Upload Button on Hub**
Yes	No	"Reset Workflow(s) before upload" is checked and *cannot* be changed.	No (hidden)
No	No	"Reset Workflow(s) before upload" is *not* checked and can be changed.	Yes (visible)
No	Yes *Default*	"Reset Workflow(s) before upload" is *checked by default*, but can be changed.	Yes (visible)

The third scenario is the *default configuration*: workflows are reset by default before upload, but users can choose to disable this behavior.



Item Upload

Configuration for uploading items to KNIME Business Hub.

☐ Enforce Workflow Reset on Upload

If this option is checked, then the "Reset Workflow(s) before upload" option in the upload dialog is selected and not editable by the user. In addition, the upload button is removed from the KNIME Hub space web UI. Enabling this option causes all workflows to be reset before being uploaded.

☑ Select "Reset Workflow(s) before upload" by Default

If this option is checked, then the "Reset Workflow(s) before upload" option in the upload dialog is selected by default. The option remains editable by the user.

Job Instrumentation Data

KNIME Business Hub can collect instrumentation data on workflow executions.

☑ Do you want to collect job instrumentation data?

Job Info Expiration

How long instrumentation data about a job is stored, in days.

30

Default value: 30

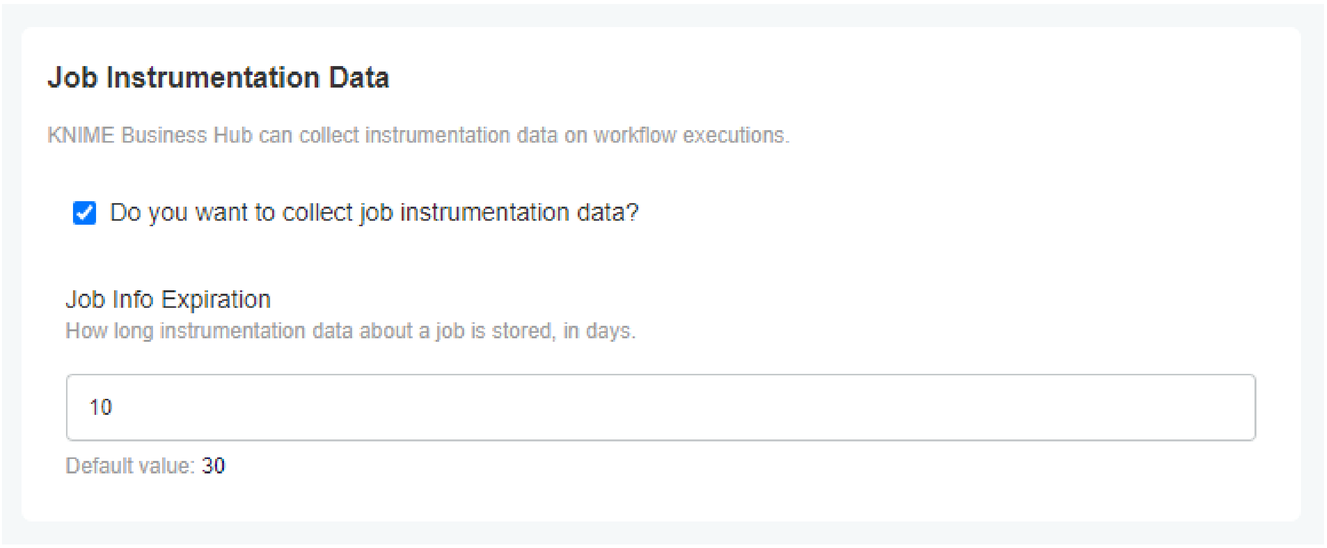
Validation Service

Configuration for the validation service.

Activate and configure Job Instrumentation Data

To make use of the service that stores job’s data you need to check the relative option in the "Job Instrumentation Data" section of the KOTS Admin Console. By default the collected data are deleted after 30 days. Here, you can also change this value to the amount of desired days.

You can use the Workflows Jobs Monitoring data application for obtaining insights about the status of workflow executions, usage of execution resources across different teams, or identify faulty jobs.


Job Instrumentation Data

KNIME Business Hub can collect instrumentation data on workflow executions.

☑ Do you want to collect job instrumentation data?

Job Info Expiration

How long instrumentation data about a job is stored, in days.

10

Default value: 30

Installation of AI services (Enterprise edition only)

The AI service is a Business Hub Enterprise feature that enables end-users to connect the AI features of their Analytics Platform (such as the KNIME AI Assistant and Code Generation) to KNIME Business Hub.

The AI service is configured via the KOTS Admin Console.

The configuration consists of the following parts:

- LLM Backend
- Disclaimer & Welcome Messages
- AI Service Q&A Content
- AI History Service

LLM Backend

Currently, it is possible to configure the AI service to use either OpenAI or Azure OpenAI as backend.

To configure OpenAI as LLM Backend do the following steps:

1. Create an OpenAI account
2. Generate an API key on the API keys page. It is recommended to create a fresh API key that is used exclusively by the AI service.

To configure Azure OpenAI as LLM Backend do the following steps:

1. Create an Azure account and get access to Azure OpenAI
2. Create an Azure OpenAI resource to be used by the service and enter it in the KOTS Admin Console. It is recommended to create a separate resource that is used exclusively by the AI service to avoid any interference with other applications. See the Azure OpenAI documentation to learn how to create a resource.
3. Deploy the following models with API version 2023-07-01-preview or newer:
    - Lightweight model: *gpt-4o-mini*
    - High intelligence model: *gpt-4o*
    - Embeddings model: *text-embedding-3-small*

   See the Azure OpenAI documentation for more details.

Models usage

Refer to this section for the latest details.

The gpt-4o-mini (Lightweight model) and the embeddings model (text-embedding-3-small) are used for the Q&A mode of the KNIME AI Assistant.

The GPT 4 model is used for code generation and the build mode of the KNIME AI Assistant. The gpt-4o (High intelligence model) is used for code generation and the build mode of the KNIME AI Assistant.

The reason for this distinction is that the gpt-4o model is more capable than the gpt-4o-mini model but also much slower to respond and more expensive. If response time and cost are no concerns, gpt-4o can also be used for the Q&A mode.

Disclaimer & welcome messages

The KOTS Admin Console also allows you to customize the initial messages displayed by the KNIME AI Assistant. The disclaimer is not shown by the KNIME AI Assistant if it is left empty.

AI Service Q&A Content

Enable these options to include workflows and/or components in the Q&A mode of the AI Assistant. The AI Assistant will then be able to answer questions about the public workflows and components available in the Business Hub.

Only public workflows and components are searched, and their descriptions are shared with the selected GenAI provider.

AI History Service

The AI service exposes an endpoint that allows you to retrieve the requests that were made to the AI service via the KNIME AI Assistant. The KOTS Admin Console allows you to configure which Keycloak groups can access the endpoint.

It is recommended to only give access to admins.

The address depends on the hub but it is typically located at `https://api.<base-url>/ai-history/kai`.

GET requests to this endpoint need to have the authorization header set with a valid bearer token from the Hub.

You have the possibility to filter data by date directly within the endpoint, for instance:

```
https://api.<base-url>/ai-history/code/python?start_time=2023-12-
31T00:00:00&end_time=2024-04-10T00:00:00
```

The format of the parameters `start_time` and `end_time` needs to be `yyyy-mm-ddThh:mm:ss` as in the example above.

The Gen AI Monitoring data app helps you monitor and govern K-AI usage in your KNIME Business Hub instance. Find more information and a step-by-step guide here.

Node affinity

Node affinity makes it possible to ensure that cluster resources intended for a specific task, e.g. execution resources, run on a specific set of nodes. There are two roles that each pod is grouped into: `core` and `execution`. Pods in the `core` group consist of KNIME Business Hub control plane resources, and pods in the `execution` group relate to execution contexts.

In order to use the node affinity feature in your KNIME Hub cluster, you can apply one or both of the following labels to nodes within your cluster:

- `hub.knime.com/role=core`
- `hub.knime.com/role=execution`

To label a node, you can execute the following command (where `<node-name>` is the name of the node you want to label):

```
kubectl label node <node-name> hub.knime.com/role=core
```

For more information about labeling nodes, see the Kubernetes documentation.

Pods will have to be restarted in order to be rescheduled onto labeled nodes. You can use the following example commands to restart the pods in a live cluster:

- `kubectl rollout restart deployment -n istio-system`
- `kubectl rollout restart deployment -n <business-hub-namespace>`
- `kubectl delete pods -n <business-hub-namespace> --selector app.kubernetes.io/name=knime-hub-execution`

  This command will restart all execution context pods.

There are a few things to note about the behavior of this feature:

- Node affinity uses a "best effort" approach to pod scheduling.
  - If one or both of the `hub.knime.com/role` labels are applied, cluster resources will attempt to be scheduled onto the nodes based on their role.
  - If no nodes have a `hub.knime.com/role` label, pods will be scheduled onto any available node.
  - If labeled nodes reach capacity, pods will be scheduled onto any available node.
  - If a labeled node is shut down, pods will be rescheduled onto other nodes in the

cluster with a preference towards using nodes that have a matching label.

  - Node affinity for KNIME Business Hub uses the `preferredDuringSchedulingIgnoredDuringExecution` approach (see the Kubernetes documentation for more details).

- It is possible to use only one of the labels above, e.g. labeling nodes for the `execution` role but not specifying any node labels for the `core` role.

Usage of taints and tolerations

Taints and tolerations are used to repel or attract pods to or from certain nodes within a Kubernetes cluster.

- A **taint** is added to one or more node(s) in a cluster. Any pods which do not tolerate the taint will not be scheduled on the node(s) with the taint.
- **Tolerations** are added to pods, and those tolerations allow the pods to be scheduled on nodes with matching taints.

The Kubernetes scheduler will attempt to match pods to nodes based on taints and tolerations. If no suitable matches are found, the pod will generally not be scheduled. This is commonly used to separate workloads in a Kubernetes cluster.

The official Kubernetes documentation on taints and tolerations can be found here.

How to use taints and tolerations in Business Hub

You can use taints and tolerations in existing cluster deployments. These existing clusters are usually running more than one application. To provide separation of resources between applications, you can taint nodes for specific applications and Hub is required to add tolerations allowing it to be deployed on the customer specified nodes.

Here is an example of how to add a taint to a given node:

```
## Add a taint to a node.
kubectl taint nodes <node-id> key=value:effect

## Example (core node):
kubectl taint nodes node1 hub.knime.com/role=core:NoSchedule
```

where:

- `node1`: is the **name** of the node you want to taint.
- `hub.knime.com/role=core`: is the **key-value pair** defining the taint.
- `NoSchedule`: is the **effect**, preventing pods without matching tolerations from being scheduled on the node.
  Refer to the Kubernetes documentation for details on the **effects** that are supported.

Deploying KNIME Business Hub with tolerations

Since KOTS is a framework integrated into all KNIME Business Hub deployments, all pods started by KOTS also have to have tolerations added. Otherwise, the KOTS pods will be missing any tolerations and may not be scheduled successfully if all nodes have one or more taint(s).

This is an example of how to use the `kots install` command to start a Business Hub deployment with the default tolerations applied to KOTS pods (requires the `kubectl kots` plugin `v1.123.0` or higher):

```
kubectl kots install knime-hub --tolerations hub.knime.com/role:Equal:Core:NoSchedule
```

This example will cause the toleration specified to be set on the pods started by KOTS, namely `kotsadm`, `minio`, and `rqlite`.
Multiple tolerations can be specified on the command line by repeating the `--tolerations` flag, for example:

```
kubectl kots install knime-hub \
--tolerations hub.knime.com/role:Equal:Core:NoSchedule \
--tolerations hub.knime.com/Project1234:Equal:Value1234:NoSchedule
```

where:

- `hub.knime.com/role:Equal:Core:NoSchedule` specifies the toleration for nodes tainted with the key.value pair `hub.knime.com/role=Core` and the `NoSchedule` effect.
- `hub.knime.com/Project1234:Equal:Value1234:NoSchedule` specifies the toleration for nodes tainted with the key-value pair `hub.knime.com/Project1234=Value1234` and the `NoSchedule` effect.

The tolerations can be customized as needed; see `kubectl kots install --help` for more information.

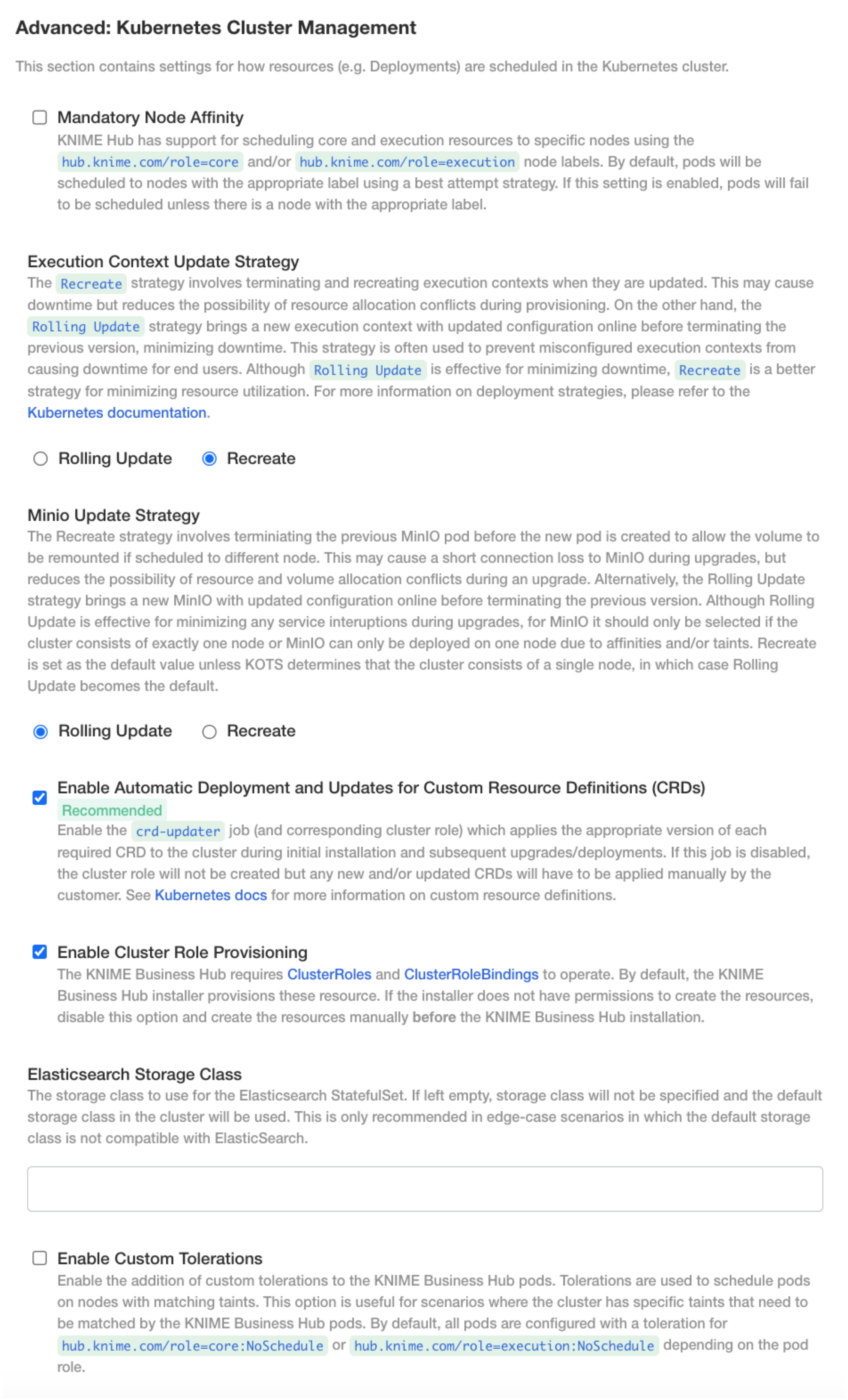
Configuring custom tolerations in KOTS Admin Console

The instructions so far have shown how to set tolerations for KOTS pods.

By default, the pods that comprise the KNIME Business Hub application (as of version 1.13.1+) support `hub.knime.com/role=core:NoSchedule` and `hub.knime.com/role=execution:NoSchedule` taints for core and execution nodes, respectively. Execution Context pods will have a `hub.knime.com/role=execution:NoSchedule` toleration whereas all other pods will have a `hub.knime.com/role=core:NoSchedule` toleration. If customization is needed, the default tolerations configuration can be customized via the KOTS Admin Console.

In order to add custom tolerations to KNIME Business Hub components, enable the *View Advanced Settings* option in KOTS Admin Console and navigate to the *Advanced: Kubernetes Cluster Management* section. Then, enable the *Enable Custom Tolerations* setting.

If any node(s) in the cluster have taint(s) applied to them, it is recommended to review and configure tolerations prior to initial installation in order to ensure that persistent volumes are provisioned on the correct nodes.



Advanced: Kubernetes Cluster Management

This section contains settings for how resources (e.g. Deployments) are scheduled in the Kubernetes cluster.

☐ **Mandatory Node Affinity**
KNIME Hub has support for scheduling core and execution resources to specific nodes using the `hub.knime.com/role=core` and/or `hub.knime.com/role=execution` node labels. By default, pods will be scheduled to nodes with the appropriate label using a best attempt strategy. If this setting is enabled, pods will fail to be scheduled unless there is a node with the appropriate label.

Execution Context Update Strategy
The `Recreate` strategy involves terminating and recreating execution contexts when they are updated. This may cause downtime but reduces the possibility of resource allocation conflicts during provisioning. On the other hand, the `Rolling Update` strategy brings a new execution context with updated configuration online before terminating the previous version, minimizing downtime. This strategy is often used to prevent misconfigured execution contexts from causing downtime for end users. Although `Rolling Update` is effective for minimizing downtime, `Recreate` is a better strategy for minimizing resource utilization. For more information on deployment strategies, please refer to the Kubernetes documentation.

Rolling Update [✓ Recreate]

Minio Update Strategy
The Recreate strategy involves terminiating the previous MinIO pod before the new pod is created to allow the volume to be remounted if scheduled to different node. This may cause a short connection loss to MinIO during upgrades, but reduces the possibility of resource and volume allocation conflicts during an upgrade. Alternatively, the Rolling Update strategy brings a new MinIO with updated configuration online before terminating the previous version. Although Rolling Update is effective for minimizing any service interuptions during upgrades, for MinIO it should only be selected if the cluster consists of exactly one node or MinIO can only be deployed on one node due to affinities and/or taints. Recreate is set as the default value unless KOTS determines that the cluster consists of a single node, in which case Rolling Update becomes the default.

[✓ Rolling Update] Recreate

☑ **Enable Automatic Deployment and Updates for Custom Resource Definitions (CRDs)**
Recommended
Enable the `crd-updater` job (and corresponding cluster role) which applies the appropriate version of each required CRD to the cluster during initial installation and subsequent upgrades/deployments. If this job is disabled, the cluster role will not be created but any new and/or updated CRDs will have to be applied manually by the customer. See Kubernetes docs for more information on custom resource definitions.

☑ **Enable Cluster Role Provisioning**
The KNIME Business Hub requires ClusterRoles and ClusterRoleBindings to operate. By default, the KNIME Business Hub installer provisions these resource. If the installer does not have permissions to create the resources, disable this option and create the resources manually **before** the KNIME Business Hub installation.

Elasticsearch Storage Class
The storage class to use for the Elasticsearch StatefulSet. If left empty, storage class will not be specified and the default storage class in the cluster will be used. This is only recommended in edge-case scenarios in which the default storage class is not compatible with ElasticSearch.

☐ **Enable Custom Tolerations**
Enable the addition of custom tolerations to the KNIME Business Hub pods. Tolerations are used to schedule pods on nodes with matching taints. This option is useful for scenarios where the cluster has specific taints that need to be matched by the KNIME Business Hub pods. By default, all pods are configured with a toleration for `hub.knime.com/role=core:NoSchedule` or `hub.knime.com/role=execution:NoSchedule` depending on the pod role.

Figure 36. Enable custom tolerations setting in KOTS Admin Console

Once you have enabled the *Enable Custom Tolerations* setting, two new options should be displayed:

1. *Custom Tolerations for Core Pods*
2. *Custom Tolerations for Execution Pods*

This configuration is sensitive to error, so a valid toleration specification must be supplied in order for the deployment to complete successfully. You can customize the toleration that displays by default or add additional toleration(s).

- *Custom Tolerations for Core Pods*:

```
- key: hub.knime.com/role
  value: core
  operator: Equal
  effect: NoSchedule
- key: hub.knime.com/custom
  value: corecustom
  operator: Equal
  effect: NoSchedule
```

- *Custom Toleration for Execution Pods*:

```
- key: hub.knime.com/role
  value: execution
  operator: Equal
  effect: NoSchedule
- key: hub.knime.com/custom
  value: executioncustom
  operator: Equal
  effect: NoSchedule
```

Scalability options for selected Hub services

KNIME Business Hub comes with preconfigured resources for the various services. These resource allocations will be fine for average sized deployments. Depending on factors like number of users, execution load, and type of usage, it will be necessary to give more resources to certain services.

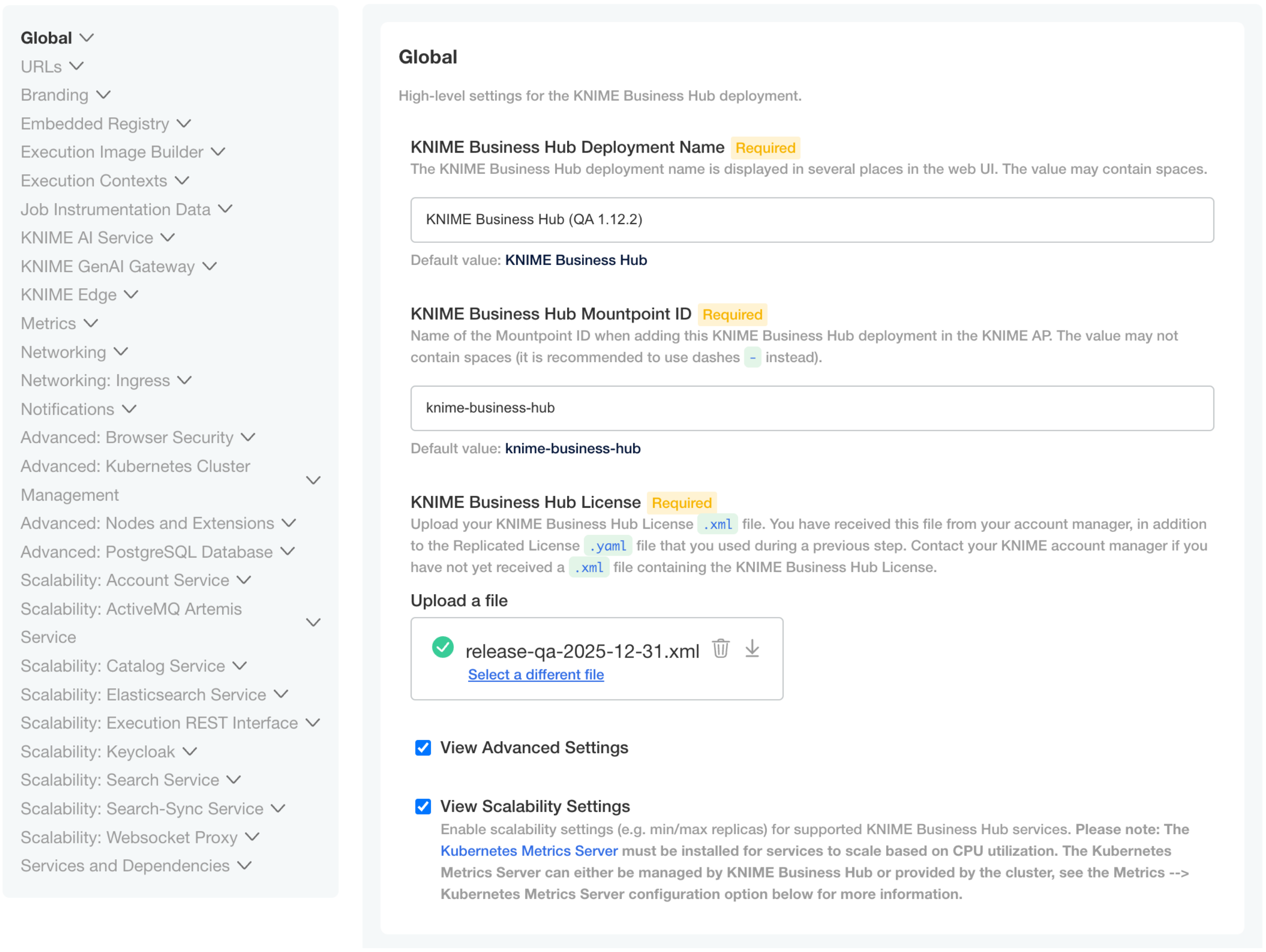
Enabling scalability options for selected Hub services adds more flexibility in setting scalability and resource usage options for your Hub instance services. Your Business Hub pods will be able to scale according to load in an automatic way.

You can specify resource usage for the following Hub services:

- Accounts
- Catalog
- Execution Rest Interface
- Search
- Websocket Proxy
- Search-sync-service
- Artemis
- Elasticsearch

Additionally, you can enable Postgres read replicas. This option takes load off of the main Postgres instance. The services that are configurted to use read replicas will send all read-only SQL requests to one of the read replicas, allowing Postgres to take on more load.

In order to enable the scalability settings go to the KOTS Admin Console and under the *Config* tab, go to the *Global* section and check the option *View Scalability Settings*.


Global
URLs
Branding
Embedded Registry
Execution Image Builder
Execution Contexts
Job Instrumentation Data
KNIME AI Service
KNIME GenAI Gateway
KNIME Edge
Metrics
Networking
Networking: Ingress
Notifications
Advanced: Browser Security
Advanced: Kubernetes Cluster Management
Advanced: Nodes and Extensions
Advanced: PostgreSQL Database
Scalability: Account Service
Scalability: ActiveMQ Artemis Service
Scalability: Catalog Service
Scalability: Elasticsearch Service
Scalability: Execution REST Interface
Scalability: Keycloak
Scalability: Search Service
Scalability: Search-Sync Service
Scalability: Websocket Proxy
Services and Dependencies

Global

High-level settings for the KNIME Business Hub deployment.

KNIME Business Hub Deployment Name Required
The KNIME Business Hub deployment name is displayed in several places in the web UI. The value may contain spaces.

KNIME Business Hub (QA 1.12.2)

Default value: KNIME Business Hub

KNIME Business Hub Mountpoint ID Required
Name of the Mountpoint ID when adding this KNIME Business Hub deployment in the KNIME AP. The value may not contain spaces (it is recommended to use dashes `-` instead).

knime-business-hub

Default value: knime-business-hub

KNIME Business Hub License Required
Upload your KNIME Business Hub License `.xml` file. You have received this file from your account manager, in addition to the Replicated License `.yaml` file that you used during a previous step. Contact your KNIME account manager if you have not yet received a `.xml` file containing the KNIME Business Hub License.

Upload a file

release-qa-2025-12-31.xml
Select a different file

☑ View Advanced Settings

☑ View Scalability Settings
Enable scalability settings (e.g. min/max replicas) for supported KNIME Business Hub services. **Please note: The Kubernetes Metrics Server must be installed for services to scale based on CPU utilization. The Kubernetes Metrics Server can either be managed by KNIME Business Hub or provided by the cluster, see the Metrics --> Kubernetes Metrics Server configuration option below for more information.**

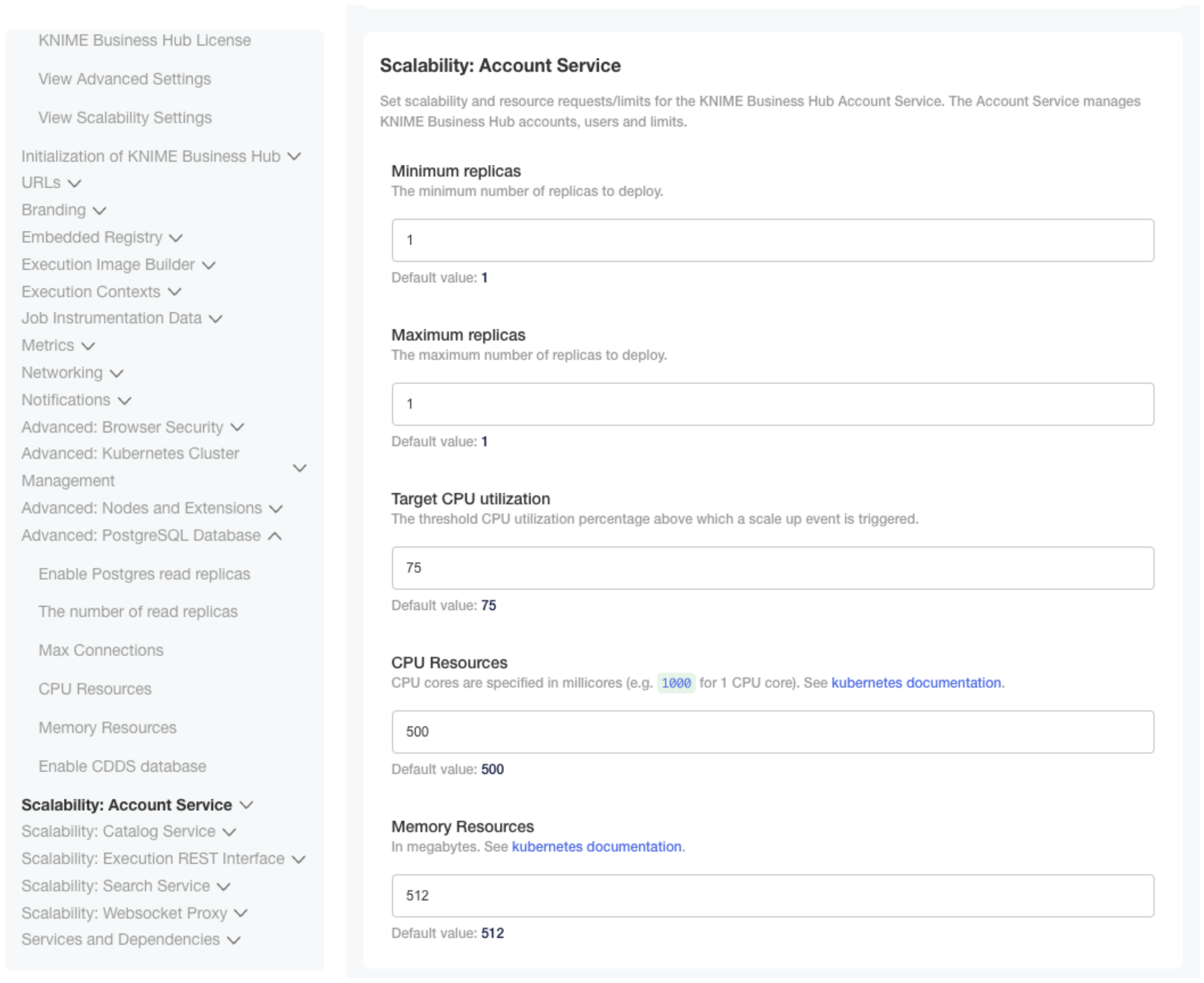
Figure 37. Enable scalability settings for you Hub instance from the KOTS Admin Console

Specify resource usage for Hub services

Once you have enabled the feature by checking the *View Scalability Settings* option, you can configure the following settings for the desired service:

- Minimum replicas: The lowest number of replicas to deploy.
- Maximum replicas: The highest number of replicas to deploy.
- Target CPU Utilization: The CPU utilization percentage threshold. When CPU usage exceeds this value the system will be triggered to scale up.
- CPU Resources: The CPU resources assigned to a container. This sets the CPU `requests`, meaning the guaranteed amount of CPU allocated to the service instance. CPU `limits`, meaning the maximum CPU usage allowed, are set equal to `resources`.
- Memory Resources: The memory resources assigned to each instance. This sets the memory `requests`, meaning the guaranteed memory allocation. Memory `limits`, meaning the maximum allowed memory, are set equal to `resources * 1.5`.

You can do so from the KOTS Admin Console by going to the *Scalability* section of each service, e.g. *Scalability: Accounts Service* for the accounts service and so on. In Figure 38 you can see an example of how to set up the scalability for the accounts service.


KNIME Business Hub License
View Advanced Settings
View Scalability Settings
Initialization of KNIME Business Hub
URLs
Branding
Embedded Registry
Execution Image Builder
Execution Contexts
Job Instrumentation Data
Metrics
Networking
Notifications
Advanced: Browser Security
Advanced: Kubernetes Cluster Management
Advanced: Nodes and Extensions
Advanced: PostgreSQL Database
Enable Postgres read replicas
The number of read replicas
Max Connections
CPU Resources
Memory Resources
Enable CDDS database
Scalability: Account Service
Scalability: Catalog Service
Scalability: Execution REST Interface
Scalability: Search Service
Scalability: Websocket Proxy
Services and Dependencies

Scalability: Account Service

Set scalability and resource requests/limits for the KNIME Business Hub Account Service. The Account Service manages KNIME Business Hub accounts, users and limits.

Minimum replicas
The minimum number of replicas to deploy.
1
Default value: 1

Maximum replicas
The maximum number of replicas to deploy.
1
Default value: 1

Target CPU utilization
The threshold CPU utilization percentage above which a scale up event is triggered.
75
Default value: 75

CPU Resources
CPU cores are specified in millicores (e.g. `1000` for 1 CPU core). See kubernetes documentation.
500
Default value: 500

Memory Resources
In megabytes. See kubernetes documentation.
512
Default value: 512

Figure 38. Configure scalability settings for Hub services - Accounts service example

Configure Postgres read replicas

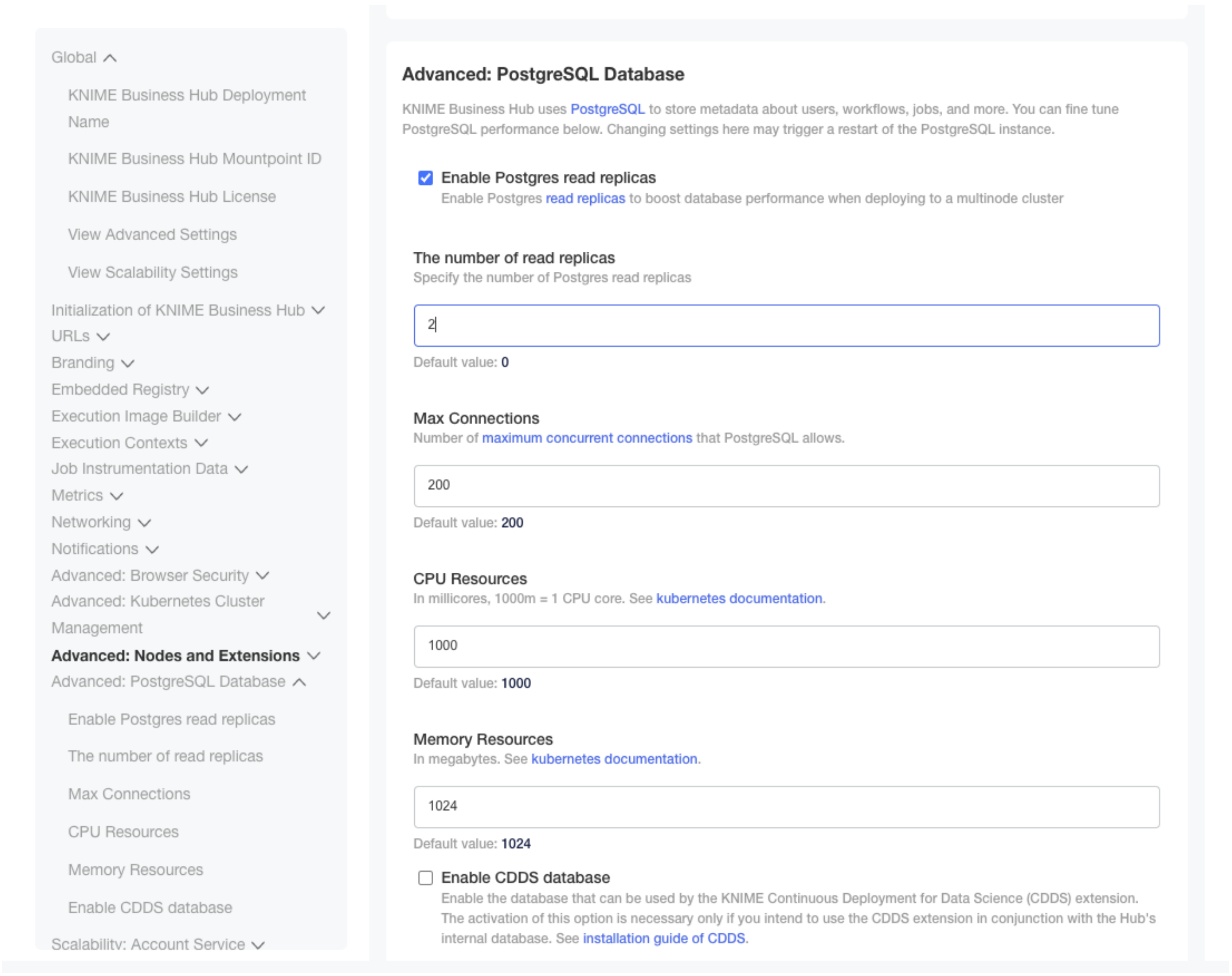
Enabling the Postgres read replicas settings requires one additional step, since this section is hidden //by default. In the KOTS Admin Console *Config* tab, in the *Global* section, enable:

- *View Advanced Settings*, and
- *View Scalability Settings*.

Then navigate to the section *Advanced: PostgreSQL Database*. Here check the option *Enable Postgres read replicas*. This will allow you to configure the following settings:

- Number of read replicas: The number of PostgreSQL read replicas.

- Max Connections: The maximum number of concurrent connections that can be established with PostgreSQL.
- CPU Resources: The CPU resources assigned to a container. This sets the CPU `requests`, meaning the guaranteed amount of CPU allocated to the service instance. CPU `limits`, meaning the maximum CPU usage allowed, are set equal to `resources`.
- Memory Resources: The memory resources assigned to each instance. This sets the memory `requests`, meaning the guaranteed memory allocation. Memory `limits`, meaning the maximum allowed memory, are set equal to `resources * 1.5`.


Global
KNIME Business Hub Deployment Name
KNIME Business Hub Mountpoint ID
KNIME Business Hub License
View Advanced Settings
View Scalability Settings
Initialization of KNIME Business Hub
URLs
Branding
Embedded Registry
Execution Image Builder
Execution Contexts
Job Instrumentation Data
Metrics
Networking
Notifications
Advanced: Browser Security
Advanced: Kubernetes Cluster Management
Advanced: Nodes and Extensions
Advanced: PostgreSQL Database
Enable Postgres read replicas
The number of read replicas
Max Connections
CPU Resources
Memory Resources
Enable CDDS database
Scalability: Account Service

Advanced: PostgreSQL Database

KNIME Business Hub uses PostgreSQL to store metadata about users, workflows, jobs, and more. You can fine tune PostgreSQL performance below. Changing settings here may trigger a restart of the PostgreSQL instance.

[✓] Enable Postgres read replicas
Enable Postgres read replicas to boost database performance when deploying to a multinode cluster

The number of read replicas
Specify the number of Postgres read replicas
2
Default value: 0

Max Connections
Number of maximum concurrent connections that PostgreSQL allows.
200
Default value: 200

CPU Resources
In millicores, 1000m = 1 CPU core. See kubernetes documentation.
1000
Default value: 1000

Memory Resources
In megabytes. See kubernetes documentation.
1024
Default value: 1024

[ ] Enable CDDS database
Enable the database that can be used by the KNIME Continuous Deployment for Data Science (CDDS) extension. The activation of this option is necessary only if you intend to use the CDDS extension in conjunction with the Hub's internal database. See installation guide of CDDS.

Figure 39. Configure Postgres read replicas settings

KNIME GenAI Gateway (Enterprise edition only)

The GenAI Gateway allows Hub admins to centrally configure GenAI models inside of the Business Hub which can then be accessed in KNIME workflows with dedicated nodes. The GenAI Gateway supports chat and embeddings models which can be queried using the KNIME Hub LLM Selector and KNIME Hub Embedding Model Selector, respectively. Both nodes use the KNIME Hub Authenticator to connect to the Hub that is running the GenAI Gateway.

This is a feature of KNIME Business Hub - Enterprise edition.

Installation

The GenAI Gateway is part of the KNIME Business Hub Enterprise offering. It can be enabled via the KOTS admin page by checking the *Enable GenAI Gateway* checkbox in the *KNIME GenAI Gateway* section.

If you have a KNIME Business Hub Enterprise license and the **section is not shown**, notify your KNIME contact person to update your license.

Model management

The Business Hub admin can manage the available models on the Hub administration page by selecting the GenAI Gateway menu entry. The page lists the already registered models and allows to add new or remove existing models.

Models are added via the + button which will open a sidepanel where the model configuration can be added.
The following configurations have to be provided for all model vendors:

- **Name**: The name under which the model will be available in the respective connector node. This name is also displayed on GenAI Gateway admin page.
- **Description**: An optional description of the model that is displayed on the GenAI Gateway admin page.
- **Type**: The type of the model, i.e. either *Chat* or *Embedding*. This information is used to filter the model list for the respective connector nodes and is also displayed on the GenAI Gateway admin page.

See below for more details and examples for the individual vendors.

OpenAI

OpenAI is an AI research and deployment company most widely known for their ChatGPT chat bot. They also offer an API that provides access to their LLM and embeddings models.

It requires the following parameter to be configured:

- **Model**: The name of the OpenAI model.
- **API Key**: A valid OpenAI API key.

Here is an example for configuring the GPT-3.5 Turbo model:

```
Model: gpt-3.5-turbo
API Key: <your OpenAI API key>
```

Azure OpenAI

Azure OpenAI provides access to OpenAI's GenAI models with the security and enterprise promise of Azure. It allows to manage details of deployed Azure OpenAI Resources and endpoints. This includes which region of the world the workloads are computed in and which models are deployed. For more details on the different configuration options please refer to the official documentation.

It requires the following parameters to be configured:

- **Model**: The name of the deployed model prefixed with `azure/`
- **API Key**: One of the API keys of the deployment.
- **API Base**: The endpoint URL of the deployment.
- **API Version**: The API version of the deployment.

Here is an example configuration for a deployment with an endpoint at `https://example.openai.azure.com` that deploys the model `my-gpt4` with API version `2024-05-01-preview`.

```
Type: Chat
Model: azure/my-gpt4
API Key: <One of the deployment API keys>
API Base: https://example.openai.azure.com
API Version: 2024-05-01-preview
```

Amazon Bedrock

Amazon Bedrock is a service managed by AWS that provides access to a variety of GenAI models from different AI companies.

It requires the following parameters to be configured:

- **Model**: The name of the model prefixed with `bedrock/`
- **Additional Model Params**
  - **aws_access_key_id**: The ID of the AWS access key to use for authentication.
  - **aws_secret_access_key**: The AWS secret key to use.
  - **aws_region_name**: The region in which to operate the resources.

Here is an example for configuring Amazon's Titan Text Express model:

```
Model: bedrock/amazon.titan-text-express-v1
Additional Model Params:
{
    "aws_access_key_id": "...",
    "aws_secret_access_key": "...",
    "aws_region_name": "eu-central-1"
}
```

Google AI Studio

Google AI Studio allows to access Google's Gemini LLMs.

It requires the following parameters to be configured:

- **Model**: The name of the model to configure prefixed with `gemini/`.
- **API Key**: A valid Goolge AI Studio API key.

Here is an example for configuring Google Gemini 1.5 Pro:

```
Model: gemini/gemini-1.5-pro-latest
API Key: <Your Goggle AI Studio API key>
```

Anthropic

Anthropic is an AI company most well-known for their Claude models.

It requires the following parameters to be configured:

- **Model**: The name of the model without any prefix.
- **API Key**: A valid Antropic API key.

Here is an example of configuring their latest model Claude 3.5 Sonnet:

```
Model: claude-3-5-sonnet-20240620
API Key: <Your Anthropic API key>
```

OpenAI API Compatible Providers

OpenAI's API is also used by a variety of other providers for example … It is also used by a number of inference solution that allow hosting your own model such as …

In order to configure such a model, the following parameters are required:

- **Model**: The name of the model prefixed with ´openai/`. This distinction from models hosted by OpenAI is crucial.
- **API Key**: A valid API key for the provider if it requires one.
- **API Base**: The base URL of the OpenAI-compatible API.

Hugging Face Hub Dedicated Inference Endpoints

Hugging Face Hub is the platform for accessing open source GenAI models. Via its Dedicated Inference Endpoints it also provides an easy way to quickly deploy a variety of LLMs and embeddings models to the cloud of your choice.

It requires the following parameters to be configured:

- **Model**: The repository of the model prefixed with `huggingface`.
- **API Key**: A valid Hugging Face API key.

- **API Base**: The URL of the deployed endpoint.

Here is an example of configuring a dedicated endpoint that deploys Mistral /B Intruct 0.3:

```
Model: huggingface/mistralai/Mistral-7B-Instruct-v0.3
API Key: <A valid Hugging Face API key>
API Base: https://example.eu-west-1.aws.endpoints.huggingface.cloud
```

Create a collection (Enterprise edition only)

It is possible to create collections on your KNIME Business Hub instance.

KNIME Collections on KNIME Hub allow upskilling users by providing selected workflows, nodes, and links about a specific, common topic.

One example of a collection can be found on KNIME Community Hub here.

This is a feature of KNIME Business Hub - Enterprise edition.

In order to create a new collection page you need to be a global admin of your KNIME Business Hub instance.

The creation of a collection is possible via REST API, and a description of the different configurations can be found in your KNIME Business Hub API doc at the following URL:

```
https://api.<base-url>/api-doc/?service=catalog-service#/Collections
```

e.g. `https://api.hub.example.com/api-doc/?service=catalog-service#/Collections`.

In order to create a collection the items (i.e. workflows and nodes) that are collected need to be stored and accessible on the same KNIME Business Hub instance where collection is created.

To create the collection you will need then to build a json file with the schema that is available in the API doc in the Collections section, under the `POST /collections` request description.

The following is an example that would allow you to build a collection, similar to the one available on KNIME Community Hub here.

In the first section you can for example set up a title, a description, a so-called hero, which is the banner image at the top right of the example collection page, and tags:

```
{
  "title": "Spreadsheet Automation",
  "description": "On this page you will find everything to get started with spreadsheet
automation in KNIME",
  "ownerAccountId": "account:user:<global-admin-user-id>",
  "hero": {
    "title": "New to KNIME?",
    "description": "Get started with <strong>KNIME Analytics Platform</strong> to import
all the examples and nodes you need for spreadsheet automation right now!",
    "actionTitle": "Download",
    "actionLink": "https://www.knime.com/downloads"
  },
  "tags": [
    "Excel",
    "XLS"
  ],
```

Next you can add different sections and subsections, each with a title and a description, choose a layout, and select the `itemType` such as *Space*, *Component*, *Workflow*, *Node*, *Extension*, or *Collection*. For each of these items you will need to provide the `id` under which they are registered in your Business Hub installation.

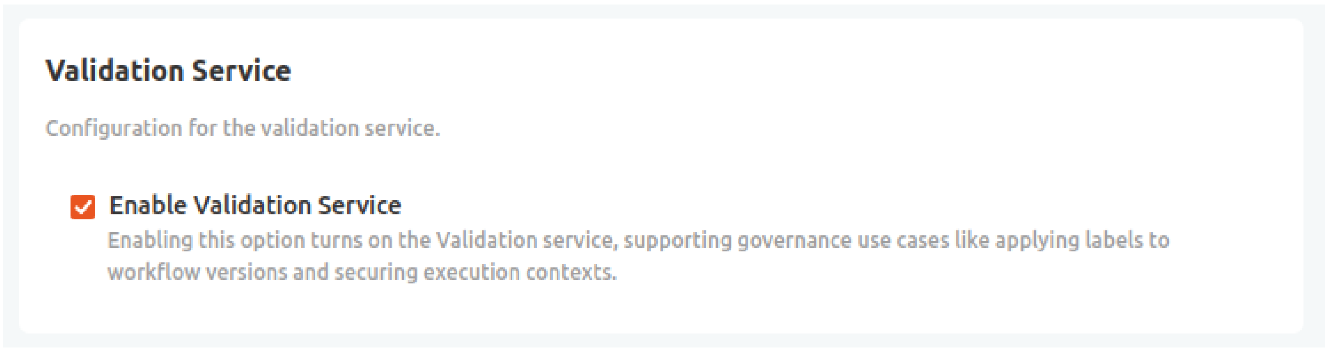
The `id` for workflows, spaces, components, and collections can be build by taking the last part of their URL, after the `~`, and adding a `*` at the beginning. For example, the following workflow on the KNIME Community Hub has URL https://hub.knime.com/-/spaces/-/latest/~1DCip3Jbxp7BWz0f/ so its `id` would be `*1DCip3Jbxp7BWz0f`. The `id` for node and extensions instead needs to be retrieved with a REST call, for example to the `search` endpoint of your KNIME Business Hub instance.

```
  "sections": [
    {
      "title": "Workflow examples",
      "description": "Some subtitle text here. Can have <strong>bold format</strong>",
      "iconType": "Workflow",
      "subsections": [
        {
          "title": "How to do basic spreadsheet tasks in KNIME",
          "description": "Some examples on how to do common things",
          "layout": "SingleColumn",
          "numberOfTeaseredItems": 2,
          "items": [
            {
              "title": "Click Here!",
              "itemType": "Link",
              "absoluteUrl": "https://knime.com"
            },
            {
              "id": "*SJW5zSkh1R3T-DB5",
              "itemType": "Space"
            },
            {
              "id": "*vpE_LTbAOn96ZOg9",
              "itemType": "Component"
            },
            {
              "id": "*MvnABULBO35AQcAR",
              "itemType": "Workflow"
            },
            {
              "showDnD": true,
              "id": "*yiAvNQVn0sVwCwYo",
              "itemType": "Node"
            },
            {
              "id": "*bjR3r1yWOznPIEXS",
              "itemType": "Extension"
            },
            {
              "id": "*QY7INTkMW6iDj7uC",
              "itemType": "Collection"
            }
          ]
        }
      ]
    }
  ]
}
```

Version labels

Version labels link information to specific versions of components or workflows. For example, they can indicate that a particular workflow has passed quality control. They can also enforce workflow execution restrictions, ensuring that a production execution context only runs workflows that have successfully passed quality control.

Prerequisite: The *Validation Service* needs to be enabled in the *KOTS Admin Console*, under > Config > Validation Service_ check *Validation Service* (see Figure 40).



Validation Service

Configuration for the validation service.

✓ **Enable Validation Service**
Enabling this option turns on the Validation service, supporting governance use cases like applying labels to workflow versions and securing execution contexts.

Figure 40. Enable the validation service in the KOTS Admin console to make use of version labels. This service is responsible for version labels, label assignment, requirements, and checking labels against requirements.

You need to first define the labels to be able to assign them to versions and use them to restrict execution contexts. The following three data apps allow you to manage version labels.

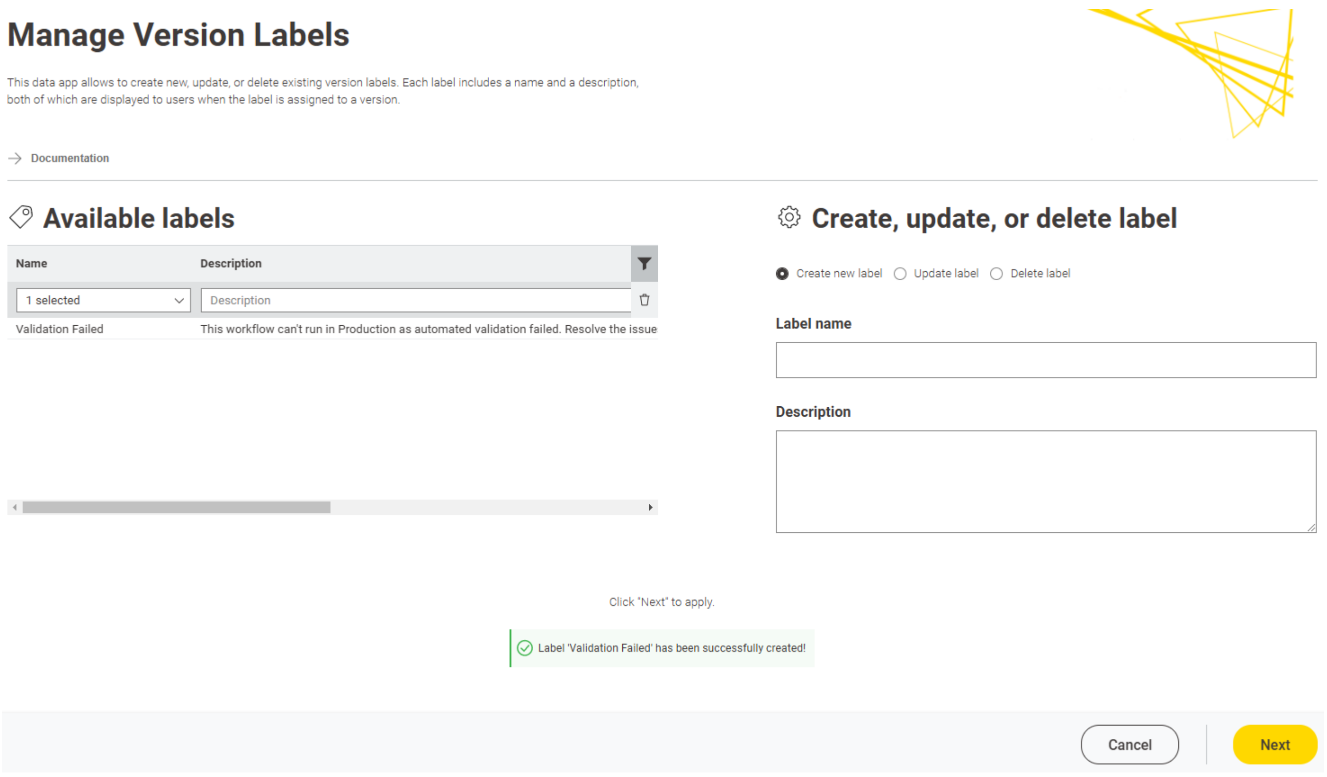
- Manage Version Labels: define, update, and delete labels.
- Assign Version Label: assign labels to component or workflow versions.
- Restrict Execution Contexts: restrict execution contexts to only allow execution and deployment of workflow versions that have a specific label assigned.

The data apps can be found in the Version Labels folder. Download the workflows, and upload them to your KNIME Business Hub to run them. They are described in more detail below.

Manage version labels

The Manage Version Labels data app allows to create, update, or delete existing version labels. Each label includes a name and a description, both of which are displayed to users when the label is assigned to a version.

The data app can be executed ad-hoc or deployed as a data app. No additional configuration is required, and only global admin users are authorized to execute it.



Manage Version Labels

This data app allows to create new, update, or delete existing version labels. Each label includes a name and a description, both of which are displayed to users when the label is assigned to a version.

→ Documentation

Available labels

Name	Description
1 selected	Description
Validation Failed	This workflow can't run in Production as automated validation failed. Resolve the issue

Create, update, or delete label

[✓ Create new label] Update label Delete label

Label name

Description

Click "Next" to apply.

Label 'Validation Failed' has been successfully created!

Figure 41. "Manage Version Labels" data app to create, update or delete labels. The screenshot shows the newly created label "Validation Failed", and the table is set to show only this label.

As shown in Figure 41, the left side shows the already created and currently available labels. On the right side, you can choose to create a new label, or update or delete an existing label. A label requires a name and a description, both of which will be shown in the version panel of a component or workflow if that label is assigned to a version.

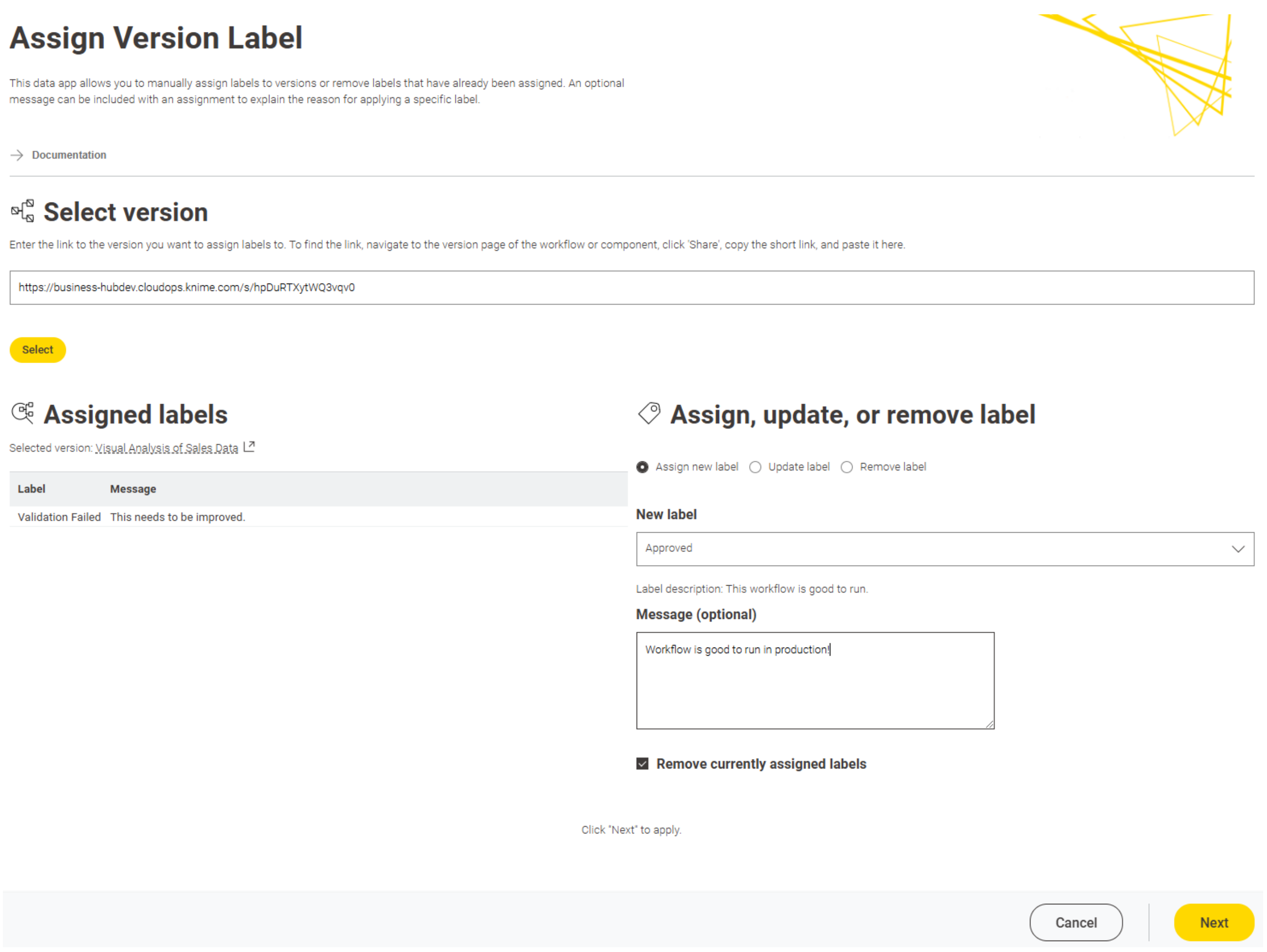
To apply any changes, click the *Next* button on the bottom of the page.

Deleting a label will remove it from all versions it was assigned to.

Assign Version Label

The Assign Version Label data app allows you to manually assign labels to versions or remove labels that have already been assigned. An optional message can be included with an assignment to explain the reason for applying a specific label.

Only global admin users have permissions to assign labels to versions. To allow non-global admin users to assign labels, the data app can be configured with a global admin user’s application password and shared with those users. The users will only be able to assign labels to versions visible to them.


Assign Version Label

This data app allows you to manually assign labels to versions or remove labels that have already been assigned. An optional message can be included with an assignment to explain the reason for applying a specific label.

→ Documentation

Select version

Enter the link to the version you want to assign labels to. To find the link, navigate to the version page of the workflow or component, click 'Share', copy the short link, and paste it here.

https://business-hubdev.cloudops.knime.com/s/hpDuRTXytWQ3vqv0

Select

Assigned labels

Selected version: Visual Analysis of Sales Data

Label	Message
Validation Failed	This needs to be improved.

Assign, update, or remove label

● Assign new label ○ Update label ○ Remove label

New label: Approved

Label description: This workflow is good to run.

Message (optional): Workflow is good to run in production!

☑ Remove currently assigned labels

Click "Next" to apply.

Figure 42. "Assign Version Label" data app to assign labels to component or workflow versions. The example shows that the label "Validation Failed" is assigned to the latest version of workflow "Visual Analysis of Sales Data" as can be seen on the left hand side. Clicking Next would change this to the label "Approved", as the "Remove currently assigned labels" setting is selected.

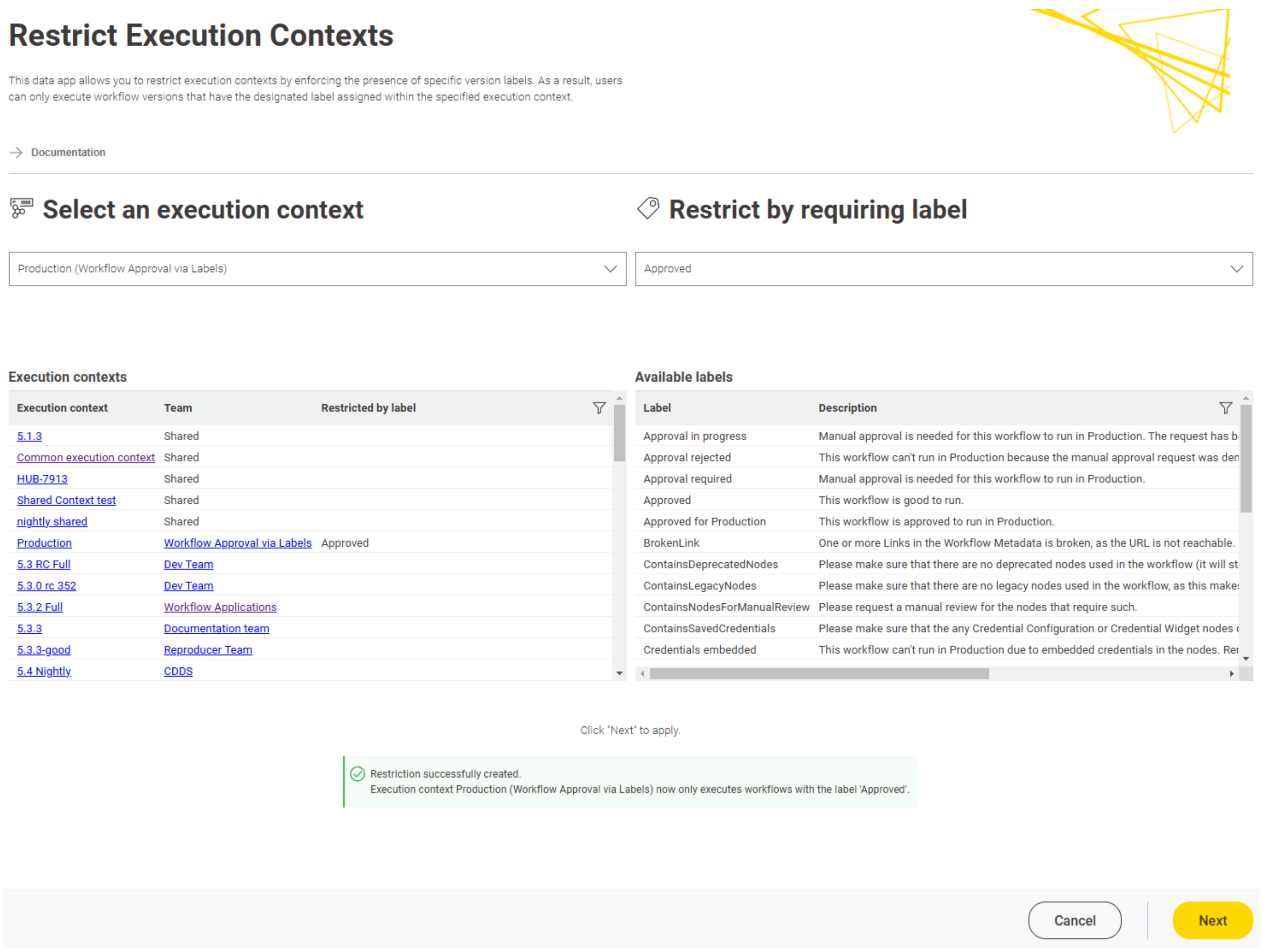
To assign a label to a version, enter the URL of a workflow or component version on the Hub in the top most field as shown in <a anchor="img-assign-version-labels">Figure 42</a>. You can find this URL on the page of the component or workflow, as shown <a href="https://docs.knime.com/hub-1.15/business_hub_user_guide/index.pdf#<em>workflows">here</a>. After clicking _Select</em>, you can see the labels currently assigned to this version on the left side. On the right side, select the label you want to assign and optionally enter a message to be displayed.

To apply any changes, click the *Next* button on the bottom of the page.

Restrict Execution Contexts

The Restrict Execution Contexts data app allows you to restrict execution contexts by enforcing the presence of specific version labels. As a result, users can only execute workflow versions that have the designated label assigned within the specified execution context.

The data app can be executed ad-hoc or deployed as a data app. No additional configuration is required, and only global admin users are authorized to execute it.



Restrict Execution Contexts

This data app allows you to restrict execution contexts by enforcing the presence of specific version labels. As a result, users can only execute workflow versions that have the designated label assigned within the specified execution context.

→ Documentation

Select an execution context

Production (Workflow Approval via Labels)

Restrict by requiring label

Approved

Execution contexts

Execution context	Team	Restricted by label
5.1.3	Shared	
Common execution context	Shared	
HUB-7913	Shared	
Shared Context test	Shared	
nightly shared	Shared	
Production	Workflow Approval via Labels	Approved
5.3 RC Full	Dev Team	
5.3.0 rc 352	Dev Team	
5.3.2 Full	Workflow Applications	
5.3.3	Documentation team	
5.3.3-good	Reproducer Team	
5.4 Nightly	CDDS	

Available labels

Label	Description
Approval in progress	Manual approval is needed for this workflow to run in Production. The request has b
Approval rejected	This workflow can't run in Production because the manual approval request was den
Approval required	Manual approval is needed for this workflow to run in Production.
Approved	This workflow is good to run.
Approved for Production	This workflow is approved to run in Production.
BrokenLink	One or more Links in the Workflow Metadata is broken, as the URL is not reachable.
ContainsDeprecatedNodes	Please make sure that there are no deprecated nodes used in the workflow (it will st
ContainsLegacyNodes	Please make sure that there are no legacy nodes used in the workflow, as this make:
ContainsNodesForManualReview	Please request a manual review for the nodes that require such.
ContainsSavedCredentials	Please make sure that the any Credential Configuration or Credential Widget nodes
Credentials embedded	This workflow can't run in Production due to embedded credentials in the nodes. Rer

Click "Next" to apply.

Restriction successfully created.
Execution context Production (Workflow Approval via Labels) now only executes workflows with the label 'Approved'.

Figure 43. "Restrict Execution Contexts" data app that allows to restrict execution to only workflow versions with a specific label assigned. The example shows that the execution context "Production" of Team "Workflow Approval via Labels" was set to only allow the execution of workflow versions with the label "Approved".

The left side shows available executions contexts and currently configured restrictions. On the right side, you can see available labels. Select the execution context and the restriction you want to configure. You can remove a restriction by selecting `<none>`.

To apply any changes, click the *Next* button at the bottom of the page.

When a restriction is added to an execution context, existing deployments will continue running, even if the workflow version does not have the required label assigned. This ensures uninterrupted operation for existing deployments.

Business Hub metrics via Grafana Dashboards

As an administrator you have access to metrics from Kubernetes, KNIME Business Hub services, and tools like Keycloak and MinIO to understand KNIME Business Hub operations better and troubleshoot performance issues more effectively, via Grafana Dashboards.

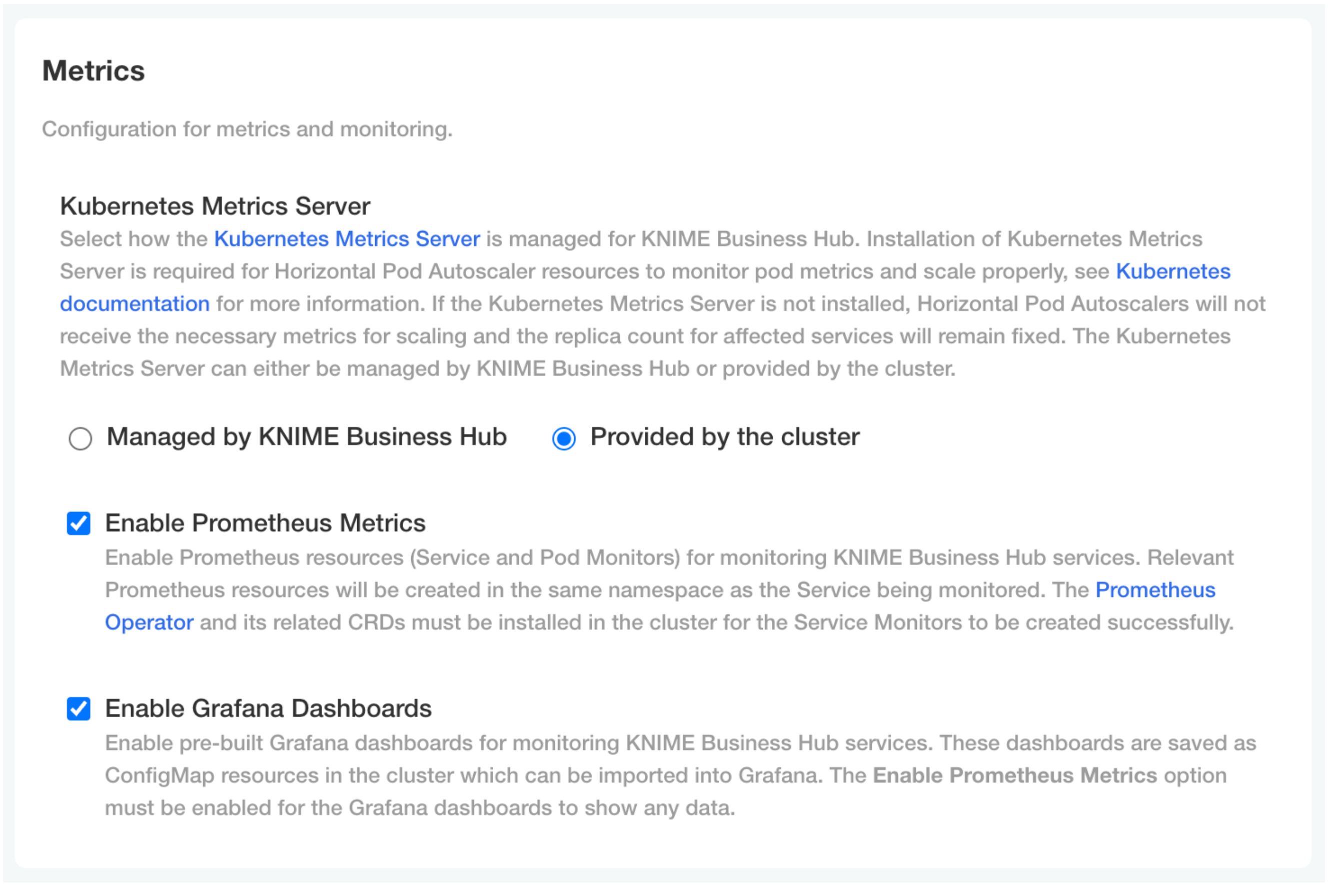
In this section you will learn how to enable Prometheus to collect and store metrics and use Grafana Dashboards to consume the collected metrics.

- **Grafana** is used to visualize metrics collected from your Kubernetes cluster.
  For **embedded cluster installations**, Grafana comes preconfigured with dashboards tailored to Kubernetes components like nodes, pods, and services. These dashboards provide out-of-the-box visibility into cluster health, performance, and resource usage, simplifying the monitoring and debugging of KNIME Business Hub.
  For KNIME Business Hub, Grafana depends on Prometheus and the Prometheus Operator to be available in the cluster.
- **Prometheus** is a monitoring system that scrapes metrics from Kubernetes components and services. It uses Kubernetes service discovery to automatically find and monitor targets within the cluster, providing a robust and dynamic way to track performance metrics.
  The Prometheus Operator simplifies and automates the deployment, configuration, and management of Prometheus instances in a Kubernetes cluster. It provides Kubernetes-native custom resources such as Prometheus, ServiceMonitor, and PodMonitor for configuring Prometheus instances.
  For KNIME Business Hub, the Prometheus Operator operator manages one or more Prometheus instances which scrape and store metrics from various services in the cluster.

For **existing cluster installations** additional steps are necessary to use monitoring via Prometheus Metrics and Grafana Dashboards, such as installing Prometheus and Grafana and deploying Prometheus Operator. Contact your designated (technical) account manager to know more about monitoring on an existing cluster.

Enabling Prometheus Metrics and Grafana Dashboards

In KOTS Admin Console, under the *Metrics* section, ensure that the *Enable Prometheus Metrics* and *Enable Grafana Dashboards* options are selected. Deploy the configuration changes if needed.


Metrics

Configuration for metrics and monitoring.

Kubernetes Metrics Server

Select how the Kubernetes Metrics Server is managed for KNIME Business Hub. Installation of Kubernetes Metrics Server is required for Horizontal Pod Autoscaler resources to monitor pod metrics and scale properly, see Kubernetes documentation for more information. If the Kubernetes Metrics Server is not installed, Horizontal Pod Autoscalers will not receive the necessary metrics for scaling and the replica count for affected services will remain fixed. The Kubernetes Metrics Server can either be managed by KNIME Business Hub or provided by the cluster.

Managed by KNIME Business Hub [✓ Provided by the cluster]

✓ Enable Prometheus Metrics

Enable Prometheus resources (Service and Pod Monitors) for monitoring KNIME Business Hub services. Relevant Prometheus resources will be created in the same namespace as the Service being monitored. The Prometheus Operator and its related CRDs must be installed in the cluster for the Service Monitors to be created successfully.

✓ Enable Grafana Dashboards

Enable pre-built Grafana dashboards for monitoring KNIME Business Hub services. These dashboards are saved as ConfigMap resources in the cluster which can be imported into Grafana. The **Enable Prometheus Metrics** option must be enabled for the Grafana dashboards to show any data.

Figure 44. Enable metrics and dashboards in KOTS Admin Console.

Grafana

Accessing Grafana

To access Grafana follow these steps:

1. **Retrieve Grafana login credentials**: The Grafana login credentials are saved in a Kubernetes Secret (called `grafana-admin` for embedded cluster installations) within the `monitoring` namespace. The commands below can be used to retrieve and decrypt the admin username and password.

```
## Find the name of the Grafana admin credentials secret.
kubectl get secrets -A | grep grafana-admin

## Retrieve Grafana admin username.
## The secret name may be different depending on cluster type and install method.
kubectl get secret -n monitoring grafana-admin -o jsonpath="{.data.admin-user}" |
base64 -d

## Retrieve Grafana admin password.
## The secret name may be different depending on cluster type and install method.
## The output will likely show a % character appended to the end, which should be
omitted.
kubectl get secret -n monitoring grafana-admin -o jsonpath="{.data.admin-
password}" | base64 -d
```

2. **Access the Grafana Console**: In order to access Grafana, the `grafana` service needs to be port-forwarded to `localhost`. Below is an example command which port-forwards the `grafana` service to `localhost:9000`.

```
## Find the name of the Grafana service.
kubectl get services -A | grep grafana

## Port-forward the Grafana service to localhost:9000.
## The service name may be different depending on cluster type and install method.
kubectl -n monitoring port-forward service/grafana 9000:80
```

After port-forwarding the `grafana` service, the web console should be accessible locally via `http://localhost:9000`. Enter the credentials that were retrieved from the `grafana-admin` secret and login.

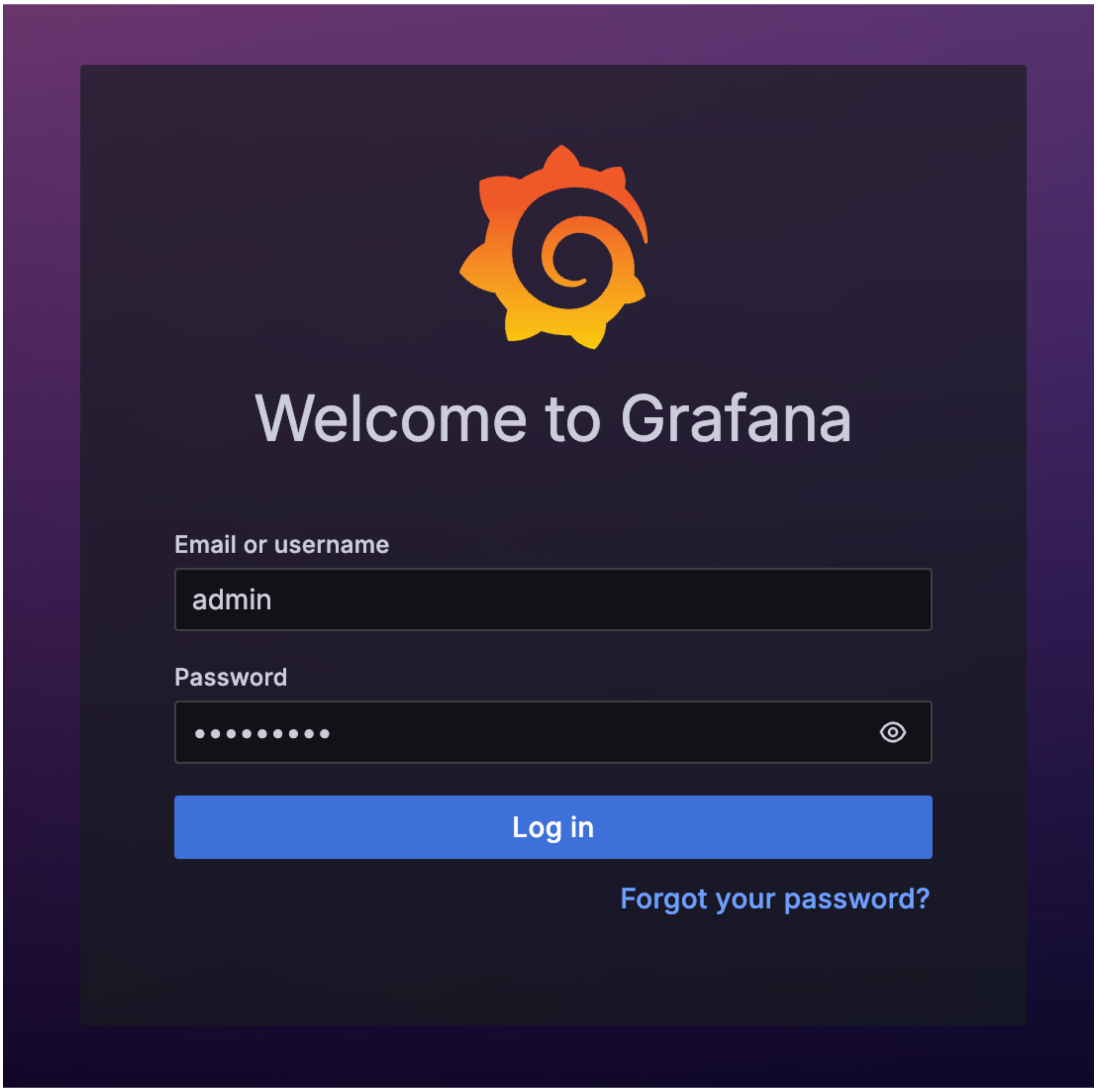
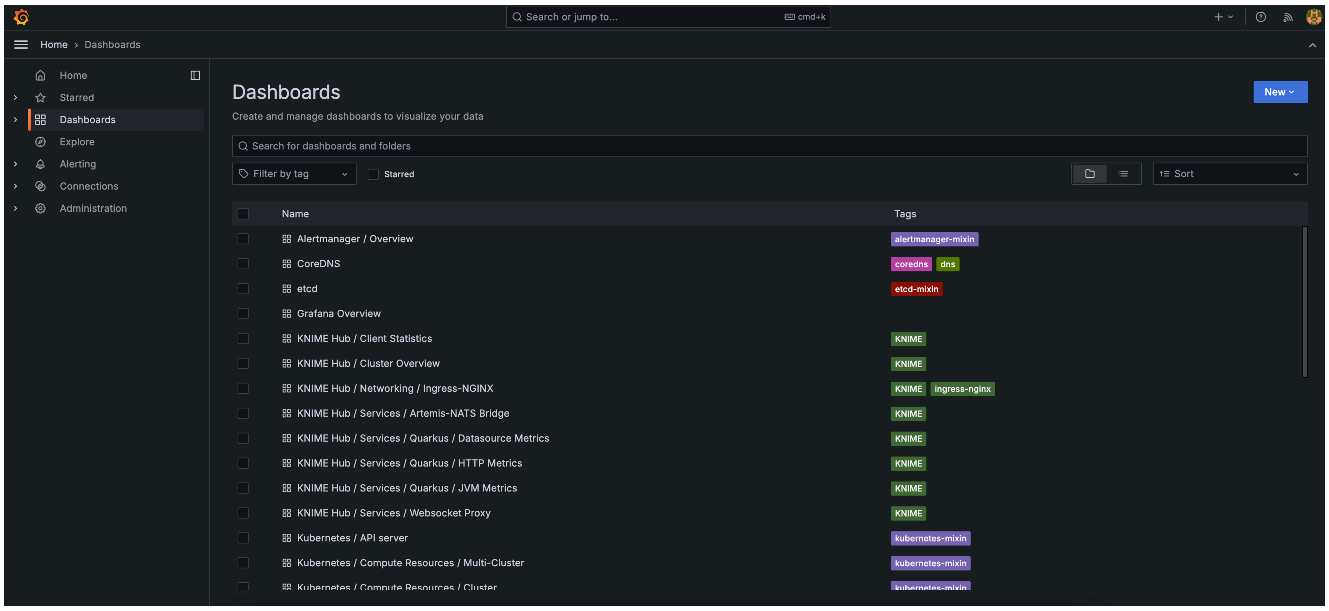


Figure 45. Log in to Grafana

Grafana web console usage

After following the above instructions, you should now have access to the Grafana web console. Navigate to *Dashboards* on the left-hand sidebar, and multiple dashboards should display. This collection includes a number of dashboards from KNIME Business Hub as well as some generic Kubernetes dashboards sourced from the `kube-prometheus-stack` Helm chart, which is installed by default as a plugin for KURL clusters.



Home › Dashboards

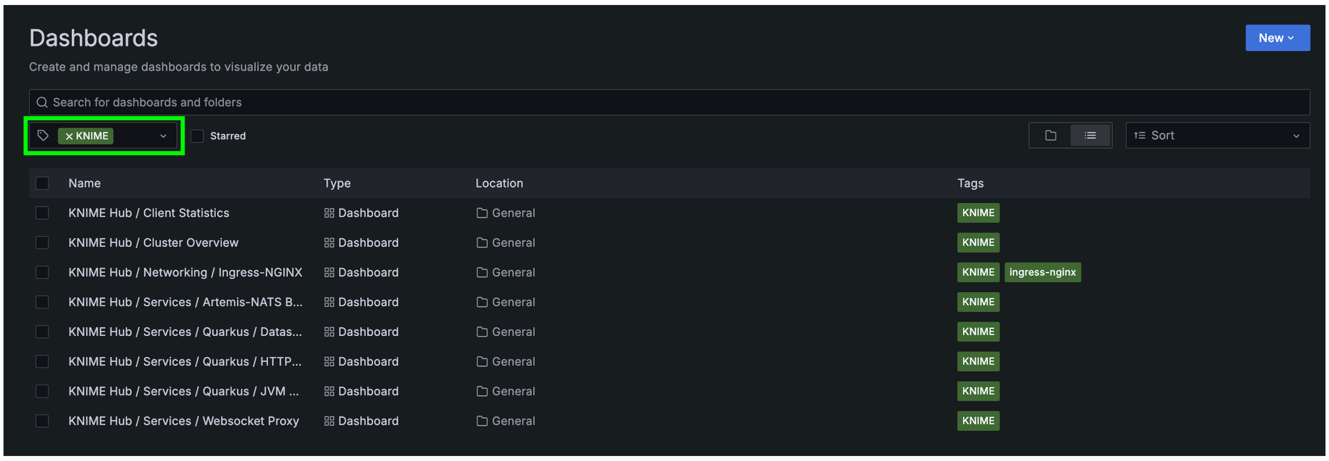
Dashboards

Create and manage dashboards to visualize your data

Name	Tags
Alertmanager / Overview	alertmanager-mixin
CoreDNS	coredns dns
etcd	etcd-mixin
Grafana Overview	
KNIME Hub / Client Statistics	KNIME
KNIME Hub / Cluster Overview	KNIME
KNIME Hub / Networking / Ingress-NGINX	KNIME ingress-nginx
KNIME Hub / Services / Artemis-NATS Bridge	KNIME
KNIME Hub / Services / Quarkus / Datasource Metrics	KNIME
KNIME Hub / Services / Quarkus / HTTP Metrics	KNIME
KNIME Hub / Services / Quarkus / JVM Metrics	KNIME
KNIME Hub / Services / Websocket Proxy	KNIME
Kubernetes / API server	kubernetes-mixin
Kubernetes / Compute Resources / Multi-Cluster	kubernetes-mixin
Kubernetes / Compute Resources / Cluster	kubernetes-mixin

Figure 46. Grafana dashboards available in KNIME Business Hub

You can optionally use the tag filter control to filter the list of dashboards down to KNIME dashboards only.



Dashboards

Create and manage dashboards to visualize your data

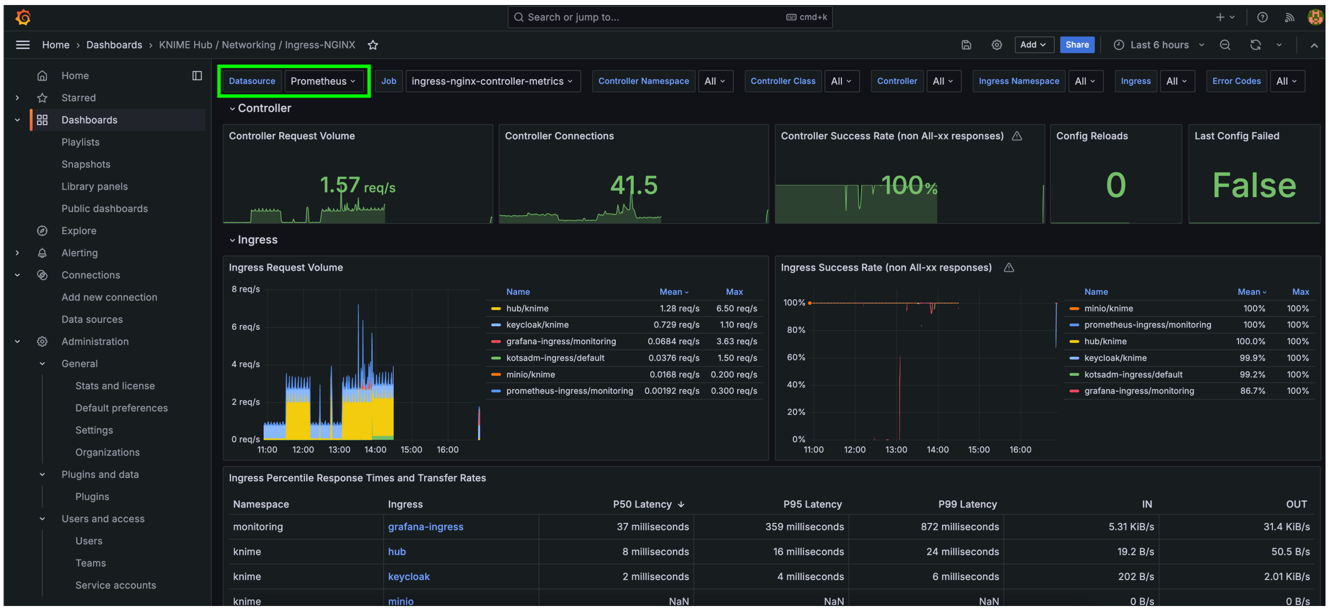
Name	Type	Location	Tags
KNIME Hub / Client Statistics	Dashboard	General	KNIME
KNIME Hub / Cluster Overview	Dashboard	General	KNIME
KNIME Hub / Networking / Ingress-NGINX	Dashboard	General	KNIME ingress-nginx
KNIME Hub / Services / Artemis-NATS B...	Dashboard	General	KNIME
KNIME Hub / Services / Quarkus / Datas...	Dashboard	General	KNIME
KNIME Hub / Services / Quarkus / HTTP...	Dashboard	General	KNIME
KNIME Hub / Services / Quarkus / JVM ...	Dashboard	General	KNIME
KNIME Hub / Services / Websocket Proxy	Dashboard	General	KNIME

Figure 47. Grafana dashboards filtered by `KNIME` tag

You can then click on any given dashboard to view and interact with it.

> See Grafana Docs: Dashboards for more information on using Grafana dashboards.

Each dashboard with the `KNIME` tag has a *Datasource* parameter which defaults to “Prometheus”. The *Datasource* parameter dynamically populates with any compatible Grafana Datasources (must be of type Prometheus). This allows pointing dashboards to a custom datasource with a different name than the default. For any custom datasources, they must be properly configured in order for data to display in the dashboards.


Datasource: Prometheus | Job: ingress-nginx-controller-metrics | Controller Namespace: All | Controller Class: All | Controller: All | Ingress Namespace: All | Ingress: All | Error Codes: All

Controller

Controller Request Volume: 1.57 req/s | Controller Connections: 41.5 | Controller Success Rate (non All-xx responses): 100% | Config Reloads: 0 | Last Config Failed: False

Ingress

Ingress Request Volume

Name	Mean	Max
hub/knime	1.28 req/s	6.50 req/s
keycloak/knime	0.729 req/s	1.10 req/s
grafana-ingress/monitoring	0.0684 req/s	3.63 req/s
kotsadm-ingress/default	0.0376 req/s	1.50 req/s
minio/knime	0.0168 req/s	0.200 req/s
prometheus-ingress/monitoring	0.00192 req/s	0.300 req/s

Ingress Success Rate (non All-xx responses)

Name	Mean	Max
minio/knime	100%	100%
prometheus-ingress/monitoring	100%	100%
hub/knime	100.0%	100%
keycloak/knime	99.9%	100%
kotsadm-ingress/default	99.2%	100%
grafana-ingress/monitoring	86.7%	100%

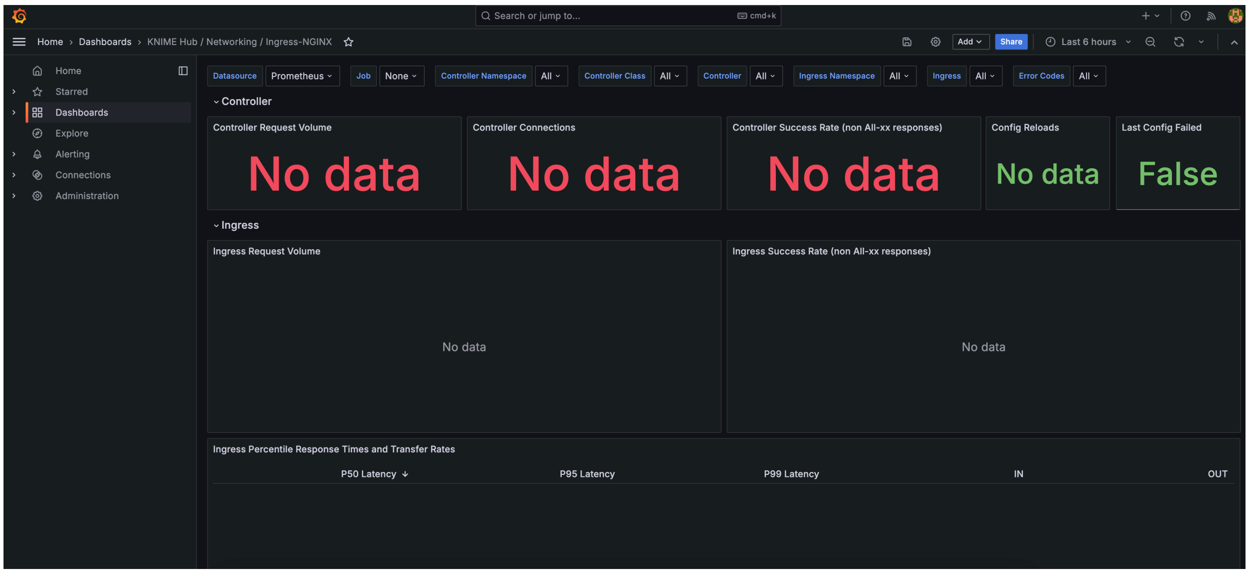
Ingress Percentile Response Times and Transfer Rates

Namespace	Ingress	P50 Latency	P95 Latency	P99 Latency	IN	OUT
monitoring	grafana-ingress	37 milliseconds	359 milliseconds	872 milliseconds	5.31 KiB/s	31.4 KiB/s
knime	hub	8 milliseconds	16 milliseconds	24 milliseconds	19.2 B/s	50.5 B/s
knime	keycloak	2 milliseconds	4 milliseconds	6 milliseconds	202 B/s	2.01 KiB/s
knime	minio	NaN	NaN	NaN	0 B/s	0 B/s

Figure 48. Example of a Dashboard and Datasource parameter

Troubleshooting

- *Dashboard shows no data*: If your dashboard shows no data, it is likely because the Enable Prometheus Metrics option has not been enabled (and deployed) via the KOTS Admin Console.


Datasource: Prometheus | Job: None | Controller Namespace: All | Controller Class: All | Controller: All | Ingress Namespace: All | Ingress: All | Error Codes: All

Controller

Controller Request Volume: No data | Controller Connections: No data | Controller Success Rate (non All-xx responses): No data | Config Reloads: No data | Last Config Failed: False

Ingress

Ingress Request Volume: No data | Ingress Success Rate (non All-xx responses): No data

Ingress Percentile Response Times and Transfer Rates

P50 Latency	P95 Latency	P99 Latency	IN	OUT

Figure 49. Dashboard does not show any data

- *Deployment errors are shown when Prometheus Metrics are enabled*: If you attempt to enable the Enable Prometheus Metrics and see deployment errors like the ones below, that indicates that the Custom Resource Definitions (CRDs) for the Prometheus Operator are not installed. For example, you might have an Existing Cluster installation of Business Hub.

```
------- ingress.nginx -------
Error: UPGRADE FAILED: resource mapping not found for name: "ingress-nginx-
controller"
namespace: "knime" from "": no matches for kind "ServiceMonitor" in version
"monitoring.coreos.com/v1"
ensure CRDs are installed first
```

How Grafana Dashboards are saved in KNIME Business Hub

Grafana Dashboards for KNIME Business Hub are saved as Kubernetes ConfigMap resources within the cluster (which contain the JSON configuration of the dashboard). This ensures that accidentally deleting a dashboard from the Grafana web console will not result in a permanent loss of the dashboard. It also ensures that changes cannot be saved directly to the dashboard - however, it is possible to create a copy of the dashboard and edit it freely.

```
## Search for all Grafana dashboard ConfigMap resources in the cluster.
## Most dashboards are prefixed with `grafana-dashboard`.
kubectl get configmaps -A | grep grafana
```

If you would like to create a custom dashboard for KNIME Business Hub, it is recommended to save it as a ConfigMap resource in the cluster to ensure it is persisted if the Grafana pod is restarted. See Grafana Docs: JSON model for more information on how to retrieve the JSON data for a custom dashboard created in the Grafana web console. Also refer to existing Grafana Dashboard ConfigMap resources as a reference, and note that the `grafana_dashboard="1"` label is required for Grafana to recognize your custom ConfigMap.

Auditing

As KNIME Business Hub administrator you can access secret store audit logs to track and analyze user activity. This is essential for identifying potential security issues and ensuring compliance with organizational policies.

For details about the structure and contents of audit log entries, see the Auditing section of the KNIME Secrets User Guide.

Viewing audit logs

Audit log data is stored in a **NATS stream** called `audit_secret-store`. You can use the `nats` command-line tool to view and filter these logs.

Prerequisites

Before you begin, ensure that you have:

- Installed the `kubectl` command-line tool.
- Access to the cluster through `kubectl`.

Steps

The following example shows how to extract all audit messages from the last two hours:

```
# Export the Kubernetes secret for NATS
secret=$(kubectl get secret -n knime nats-audit-credentials -o json)

# Extract the user name from the secret
user=$(echo "$secret" | jq -r .data.USER_ID_NATS_AUDIT | base64 -d)

# Extract the password from the secret
pass=$(echo "$secret" | jq -r .data.PASSWORD_NATS_AUDIT | base64 -d)

# Forward the NATS port (default 4222) to localhost:4222
kubectl port-forward -n knime services/nats 4222:4222 &

# Display messages from the last 2 hours
nats stream view --since=2h --server "nats://localhost:4222" \
  --user $user --password $pass audit_secret-store
```

Adjust the `--since` parameter to view audit logs for a different timeframe.

KNIME Business Hub API documentation

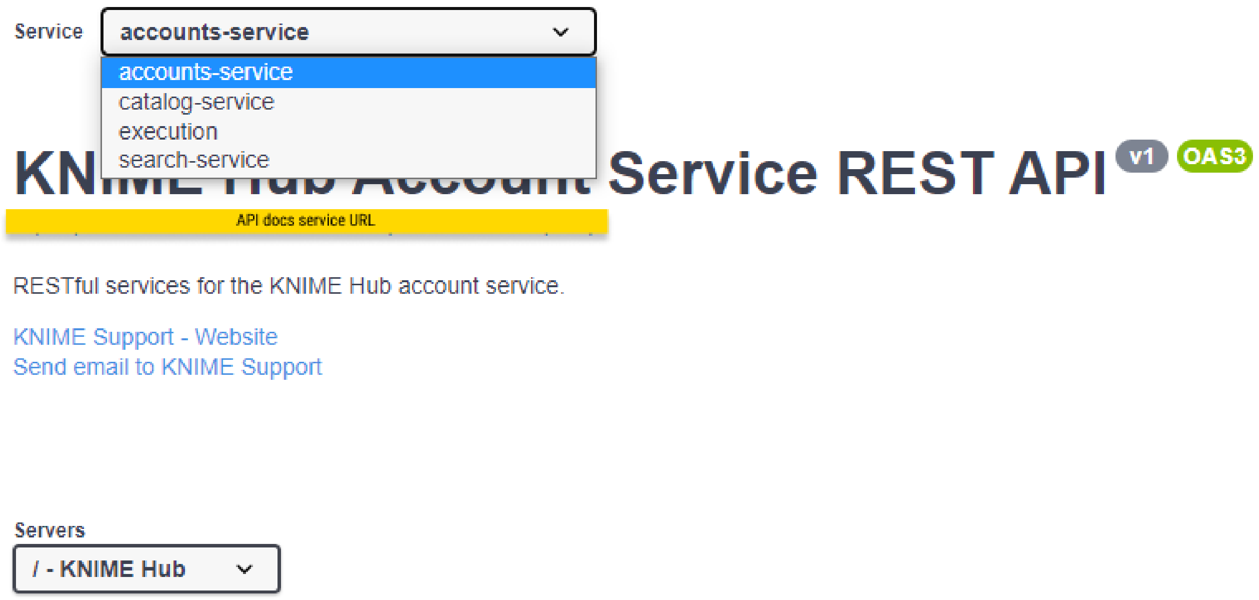
Most KNIME Business Hub functionalities are also available via REST API allowing you to perform several actions.

You can access the API documentation by navigating to the following URL:

```
api.<base-url>/api-doc
```

where `<base-url>` is your Business Hub instance URL, e.g. `hub.example.com`.

Here you can select from the drop-down menu the service you want to use.



RESTful services for the KNIME Hub account service.

KNIME Support - Website
Send email to KNIME Support

Support Bundles and Troubleshooting

When generating a support bundle, ***no data leaves the cluster***.

If necessary, you can download the support bundle and send it to KNIME for the purpose of troubleshooting. Contact us by sending an email to support@knime.com. Under extreme circumstances, the KNIME team may forward the support bundle to the Replicated support team for additional help.

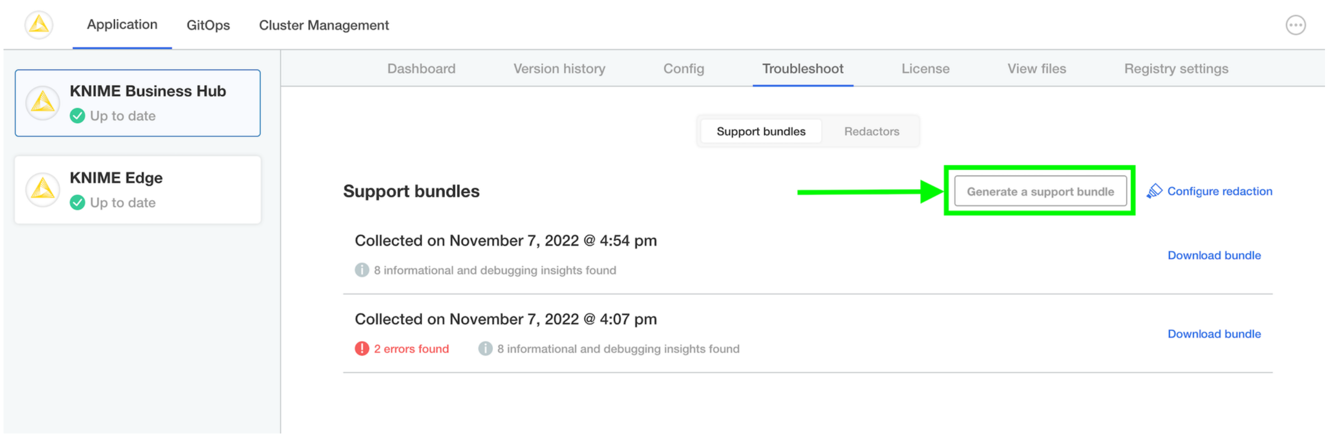
When generating a support bundle, a limited amount of information will be automatically redacted (IPv4 addresses, connection strings, etc.). You can configure additional redactions and/or manually redact information prior to sending the bundle. See the **Configuring redaction in support bundles** section for more details.

KNIME Business Hub is capable of generating support bundles in a standard format, even when the admin console isn't working. This ensures that users are able to provide all of the necessary information for KNIME to be able to identify the problem and prescribe a solution.

Generating a support bundle (GUI)

In order to help troubleshoot an installation, or to simply inspect the logs of the cluster in a user-friendly format, you will need to generate a support bundle.

Simply open the KOTS Admin Console, navigate to the **Troubleshoot** pane, and click the **Generate a support bundle** button to generate a support bundle.



Collected on November 7, 2022 @ 4:54 pm
8 informational and debugging insights found

Collected on November 7, 2022 @ 4:07 pm
2 errors found
8 informational and debugging insights found

Figure 50. Generate a support bundle

All generated support bundles will display in the list above. Click the **Download bundle** button to download the bundle(s) you want to share with KNIME, and please see the **Configuring redaction in support bundles** section for information on how to redact confidential/personal information before sending.

Generating a support bundle (CLI)

See https://docs.replicated.com/vendor/support-bundle-generating [Replicated documentation] for instructions on how to generate a support bundle via the Replicated CLI.

Generating host bundles for kURL

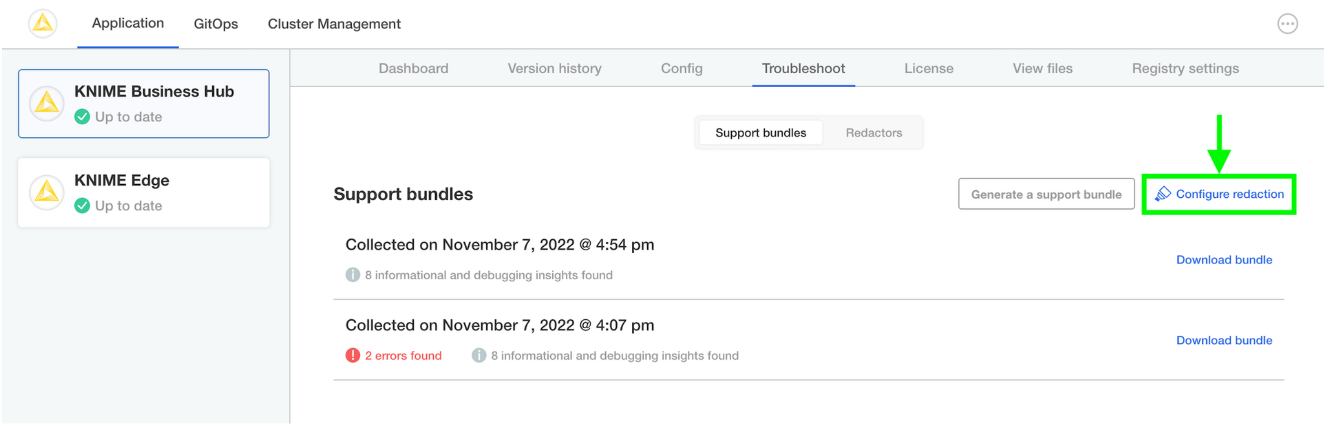
See the Replicated documentation for instructions on how to generate host bundles for kURL.

Configuring redaction in support bundles

When generating a support bundle, a limited amount of information will be automatically redacted (IPv4 addresses, connection strings, etc.) but it is not guaranteed to be a comprehensive set of redactions. You may have additional information in your logs or configuration that you do not wish to share with the KNIME engineering team.

One option is to unzip the generated .zip support bundle and manually review/redact information prior to sending the bundle to KNIME. However, there is a lot of information to review and the redaction of certain information can be automated fairly easily. The ideal option is to configure automated redactions via Redactor resources, which will automatically redact information for all future support bundles.

In order to configure automated redactors, first open the KOTS Admin Console. Navigate to the **Troubleshoot** pane and click **Configure Redaction**.



Collected on November 7, 2022 @ 4:54 pm
8 informational and debugging insights found

Collected on November 7, 2022 @ 4:07 pm
2 errors found
8 informational and debugging insights found

Figure 51. Configure Redaction

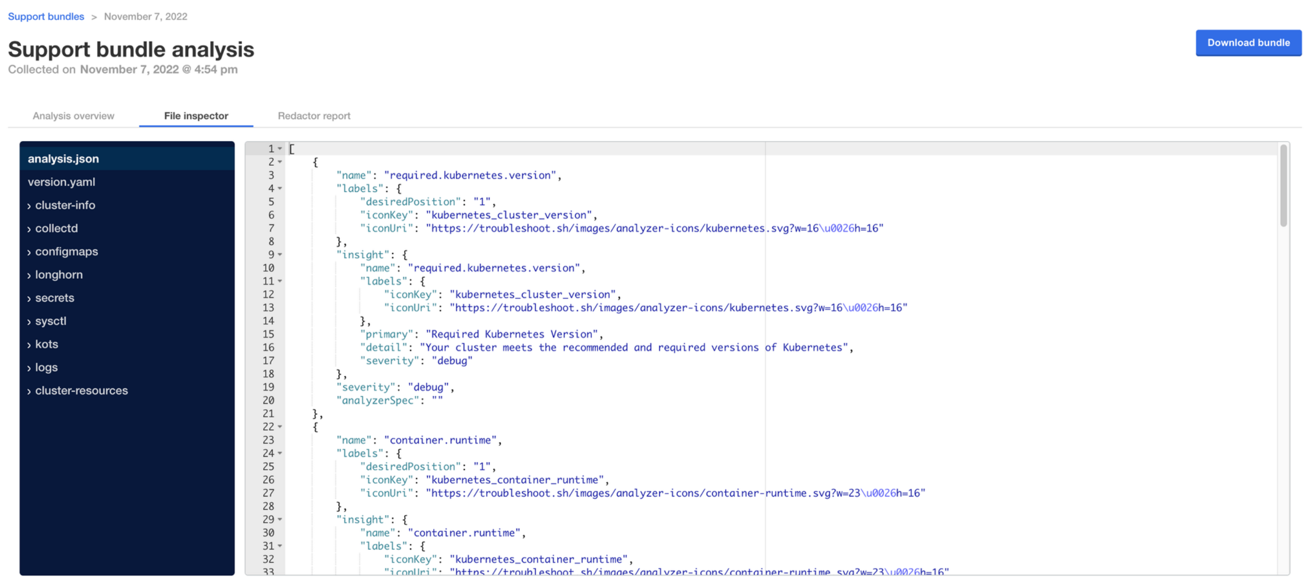
If you have configured your own custom redactions that you feel would be valuable to other users of KNIME Business Hub, please feel encouraged to share the configuration with KNIME so that it can be considered & potentially added to future releases.

See this link and this link for more information.

Inspecting support bundles

There are quite a number of of files generated in a support bundle. Not necessarily every file is useful for every problem. However, by collecting the same information in the same way each time, KNIME can ensure the best quality support possible for customers.

It is possible to inspect a support bundle entirely in the admin console. See below for an example screenshot.



```
[
    {
        "name": "required.kubernetes.version",
        "labels": {
            "desiredPosition": "1",
            "iconKey": "kubernetes_cluster_version",
            "iconUri": "https://troubleshoot.sh/images/analyzer-icons/kubernetes.svg?w=16\u0026h=16"
        },
        "insight": {
            "name": "required.kubernetes.version",
            "labels": {
                "iconKey": "kubernetes_cluster_version",
                "iconUri": "https://troubleshoot.sh/images/analyzer-icons/kubernetes.svg?w=16\u0026h=16"
            },
            "primary": "Required Kubernetes Version",
            "detail": "Your cluster meets the recommended and required versions of Kubernetes",
            "severity": "debug"
        },
        "severity": "debug",
        "analyzerSpec": ""
    },
    {
        "name": "container.runtime",
        "labels": {
            "desiredPosition": "1",
            "iconKey": "kubernetes_container_runtime",
            "iconUri": "https://troubleshoot.sh/images/analyzer-icons/container-runtime.svg?w=23\u0026h=16"
        },
        "insight": {
            "name": "container.runtime",
            "labels": {
                "iconKey": "kubernetes_container_runtime",
                "iconUri": "https://troubleshoot.sh/images/analyzer-icons/container-runtime.svg?w=23\u0026h=16"
```

Figure 52. Inspect a support bundle in the admin console

Here are the most important folders/files and their purposes:

Path	Purpose	Example (may have properties omitted)
`./analysis.json`	• Collects the highest-level insights possible for the installation. • Often times, the issue and/or resolution may be identified in this file by inspecting the `[].name.insight.detail` property.	`[ { "name": "kotsadm.status", "insight": { "name": "kotsadm.status", "primary": "kotsadm Status", "detail": "At least 1 replica of the Admin Console API is running and ready", "severity": "debug" }, "severity": "debug", "analyzerSpec": "" } ]`
`./logs`	• Contains logs of individual pods. • Execution Context logs are stored in `./logs/execution-contexts`.	(typical application logs)

Path	Purpose	Example (may have properties omitted)

`./cluster-resources`	• Contains the configuration of each visible resource in the cluster.
	• For example, to see all pods in the cluster, navigate to the `./cluster-resources/pods` directory which contains one file per namespace in the cluster.
		`{`
		`  "kind": "PodList",`
		`  "apiVersion": "v1",`
		`  "metadata": {`
		`    "resourceVersion":`
		`"1686941"`
		`  },`
		`  "items": [ ... ]`
		`}`

Maintenance operations

Shutting down and restarting a node

You might need to reboot a node if you are performing maintenance on the operating system level of the node, e.g. after a kernel update, rebooting the node will apply the changes.

Before rebooting a node on a cluster managed by kURL, make sure that snapshots completed without errors. To shut down a cluster node, call the shutdown script on the node as elevated user:

```
/opt/ekco/shutdown.sh
```

See more documentation on rebooting nodes here.

Otherwise, after a VM restart old pods might be in `Failed` or `Shutdown` state.

In case that happens, delete failed pods after the restart with the following command:

```
kubectl delete pod --field-selector=status.phase==Failed --all-namespaces
```

Backup and restore with Velero Snapshots and Kotsadm

Snapshot backups and restore features are available into Replicated deployments via Velero, a tool for backing up Kubernetes cluster resources and persistent volumes.

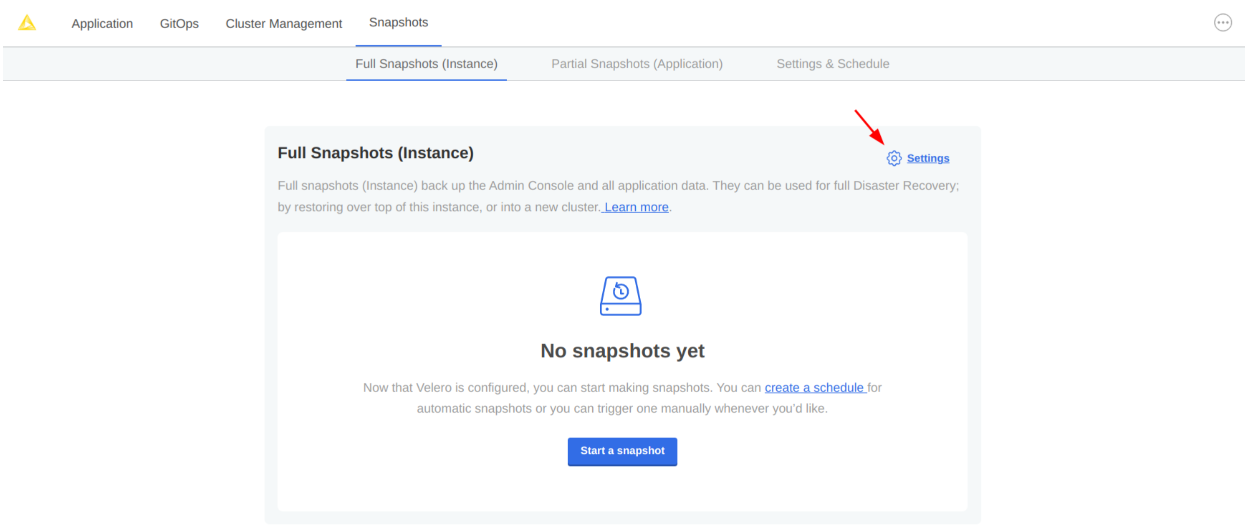
If you use HostPath for Velero backups, make sure you follow the requirements described in the Replicated documentation. This may include setting specific permissions or ownership for your backup directory. For details and the latest instructions, see the Replicated HostPath backup documentation.

One-time snapshots as well as automated scheduled snapshots can be managed from the *Snapshots* panel within your Kotsadm dashboard at `https://<base-url>:8800/app/knime-hub`.

Snapshot creation and restoration are disruptive processes. KNIME applications, and Replicated admin access may be unavailable during an active backup or restore operation.

Creating snapshot backups

1. First, configure storage for your backups. Navigate to the *Snapshots* tab of your Kotsadm dashboard. Click the 'Settings' button to edit backup settings where you'll be able to add a storage target.



Full Snapshots (Instance)

Full snapshots (Instance) back up the Admin Console and all application data. They can be used for full Disaster Recovery; by restoring over top of this instance, or into a new cluster. Learn more.

No snapshots yet

Now that Velero is configured, you can start making snapshots. You can create a schedule for automatic snapshots or you can trigger one manually whenever you'd like.

Figure 53. Snapshots tab with settings link

2. Velero supports local storage (not recommended), Amazon S3, Azure Blob Store, Google Cloud Storage, and S3 API compatible storage endpoints such as MinIO. Select your preferred snapshot storage type from the 'Destination' drop-down menu, and fill in the required fields with parameters specific to your storage endpoint. Click the 'Update storage settings' button and wait for Velero to verify backup storage access.

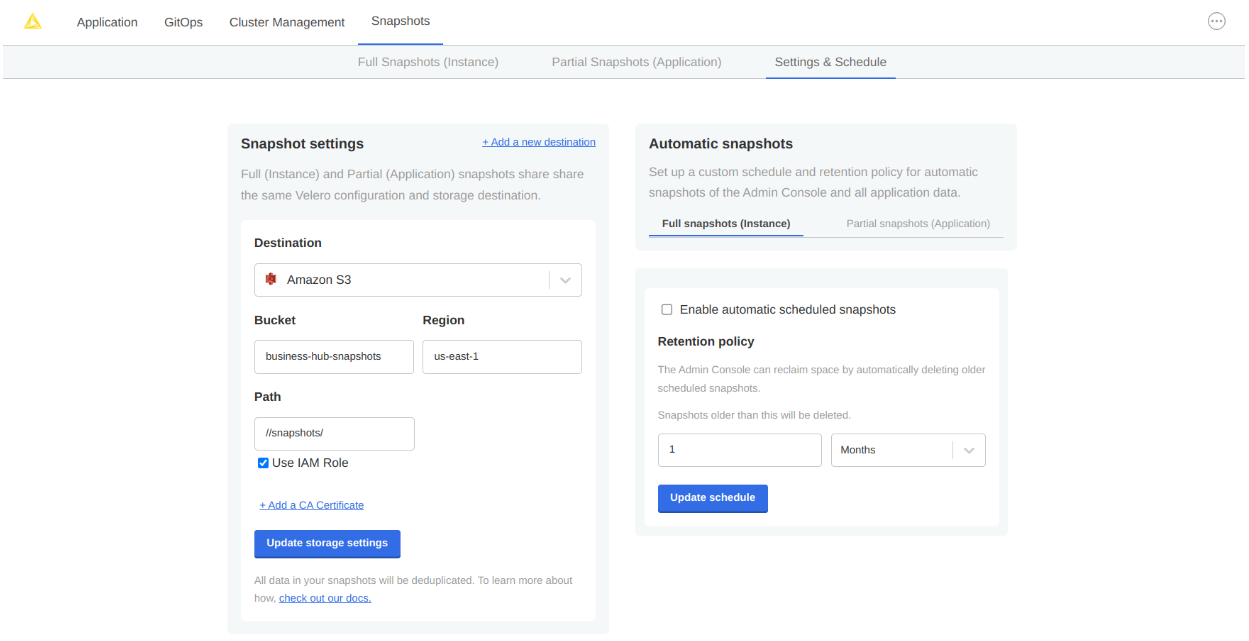
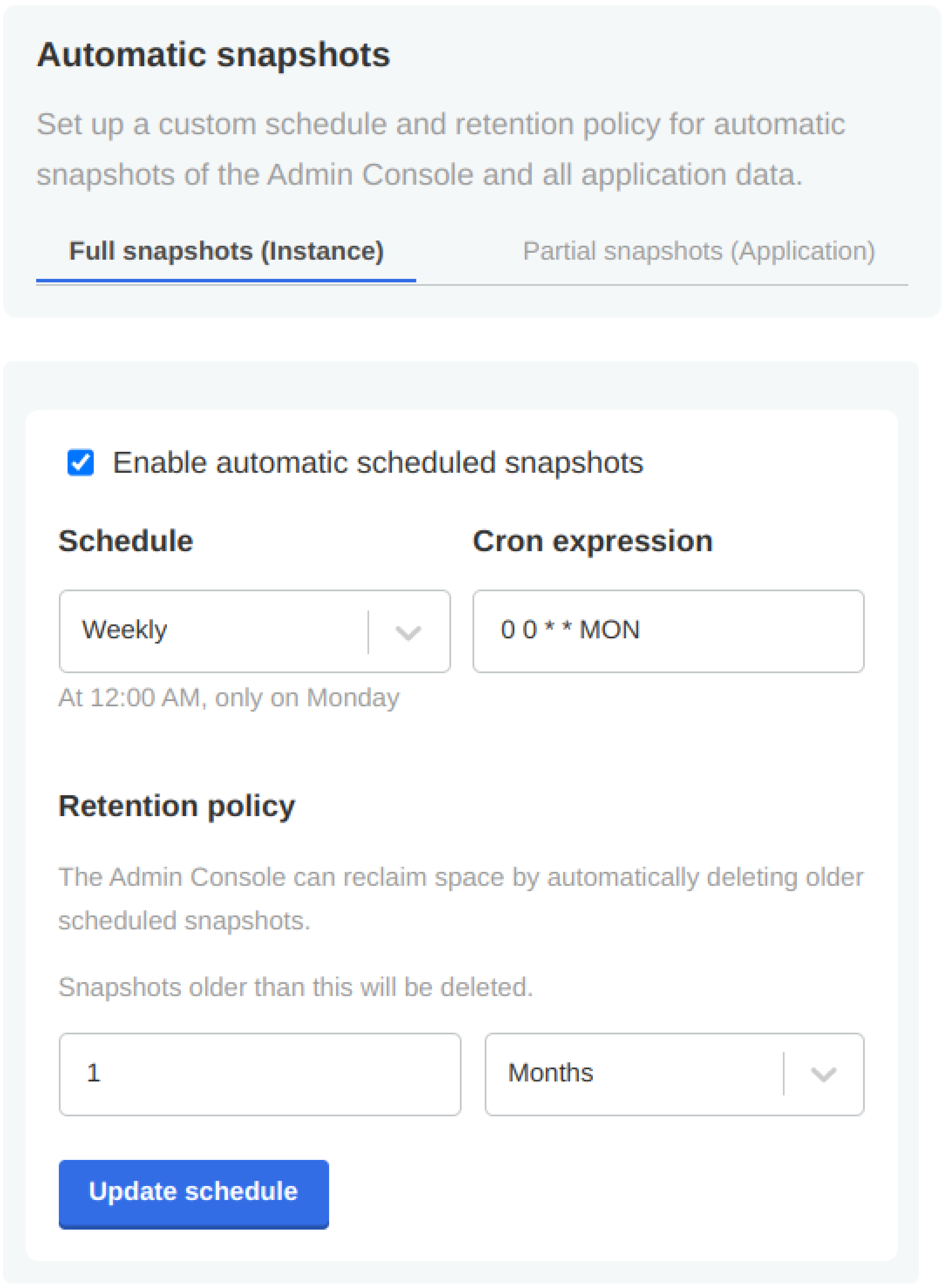

Application
GitOps
Cluster Management
Snapshots
Full Snapshots (Instance)
Partial Snapshots (Application)
Settings & Schedule
Snapshot settings
+ Add a new destination
Full (Instance) and Partial (Application) snapshots share share the same Velero configuration and storage destination.
Destination
Amazon S3
Bucket
business-hub-snapshots
Region
us-east-1
Path
//snapshots/
Use IAM Role
+ Add a CA Certificate
Update storage settings
All data in your snapshots will be deduplicated. To learn more about how, check out our docs.
Automatic snapshots
Set up a custom schedule and retention policy for automatic snapshots of the Admin Console and all application data.
Full snapshots (Instance)
Partial snapshots (Application)
Enable automatic scheduled snapshots
Retention policy
The Admin Console can reclaim space by automatically deleting older scheduled snapshots.
Snapshots older than this will be deleted.
1
Months
Update schedule

Figure 54. Snapshots destination settings for AWS S3 storage

3. With a valid backup storage configured, you can create a Snapshot of your KNIME deployment by clicking the *Full Snapshots* tab, and then the *Start a snapshot* button. This may take a few minutes to complete.

4. Once your snapshot is complete, from the same *Full Snapshots* screen, you can click the 'Settings' button to manage snapshot retention, or configure automatic snapshots by checking the *Enable automatic scheduled snapshots* box and setting a schedule using a CRON expression.


Automatic snapshots

Set up a custom schedule and retention policy for automatic snapshots of the Admin Console and all application data.

☑ Enable automatic scheduled snapshots

Schedule: Weekly

Cron expression: 0 0 * * MON

At 12:00 AM, only on Monday

Retention policy

The Admin Console can reclaim space by automatically deleting older scheduled snapshots.

Snapshots older than this will be deleted.

1 Months

Figure 55. Example automatic snapshot scheduled to run at 12:00am weekly with a 1 month retention policy.

Backup troubleshooting

Velero is installed into the embedded `Kurl` Kubernetes cluster with default settings and resource allocations.

As the number of objects or overall size of data to be backed up increases, it may eventually occur that the CPU and memory resources allocated for Velero processes are no longer sufficient to successfully complete the backup.

In the event that backup failures are encountered, it is recommended to increase the CPU and memory allocation **directly** to the Velero's node agent process via `kubectl`.

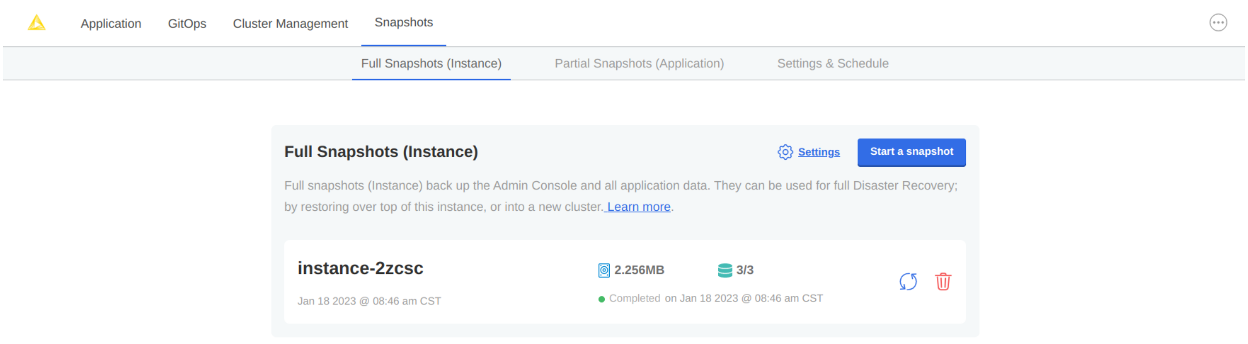
```
$ kubectl patch daemonset node-agent -n velero --patch \
'{"spec":{"template":{"spec":{"containers":[{"name": "node-agent", "resources":
{"limits":{"cpu": "2", "memory": "2048Mi"}, "requests": {"cpu": "1", "memory":
"512Mi"}}}]}}}}'
```

The CPU and memory resources and limit values can be adjusted as needed to find sufficient values for backup process. Typically, only the **limit** values will need to be increased.

At this time, the resource allocation override to Velero will **revert** after a `Kurl` upgrade has been performed. Please ensure any changes to the Velero node agent are reapplied after any `Kurl` cluster-level upgrades.

Restoring a snapshot in the **same cluster**

1. Prerequisites:
    a. Kubectl is installed and configured to access the cluster.
    b. Helm is installed and configured to access the cluster. Find more information in the Helm install guide.
    c. `KUBECONFIG` environment variable is set to the correct cluster kubeconfig file.
    d. The cluster is running the same version of KNIME Business Hub as the cluster from which the snapshot was taken.
2. Navigate to the list of available snapshot restore points from your Kotsadm dashboard by browsing to *Snapshots*→ *Full Snapshots*. From this screen, identify the snapshot instance you would like to use, and take note of the instance ID.



Full snapshots (Instance) back up the Admin Console and all application data. They can be used for full Disaster Recovery; by restoring over top of this instance, or into a new cluster. Learn more.

instance-2zcsc
Jan 18 2023 @ 08:46 am CST
2.256MB
3/3
Completed on Jan 18 2023 @ 08:46 am CST

Figure 56. In this example, there is only one snapshot available and its ID is `instance-2xcsc`

A list of snapshots can also be retrieved by command line:

```
$ kubectl kots get backups
NAME              STATUS       ERRORS    WARNINGS    STARTED
COMPLETED                        EXPIRES
instance-2zcsc    Completed    0         0           2023-01-18 14:46:26 +0000 UTC
2023-01-18 14:46:53 +0000 UTC    29d
```

3. **Caution: highly destructive process**

   This next step is **highly destructive** and **irreversible**. It should only be executed if there are backups available in ***Complete*** state, to ensure that the cluster can be fully restored.

   A restore script is available to automate the restore process. The script will prepare the environment to restore the snapshot to the cluster.

   The script can be downloaded from the following link:
   Download the restore script

   To make the script executable, run the following command: `chmod +x knime-business-hub-restore-1_2.sh`

   Before proceeding with executing the script, ensure that the correct cluster is targeted and that the snapshot storage is accessible from the new cluster. This can be confirmed by running the following command: `kubectl config current-context` or checking / setting the `KUBECONFIG` environment variable.

   The script can be executed by running the following command: `bash <path to script>/knime-business-hub-restore-1_2.sh`.

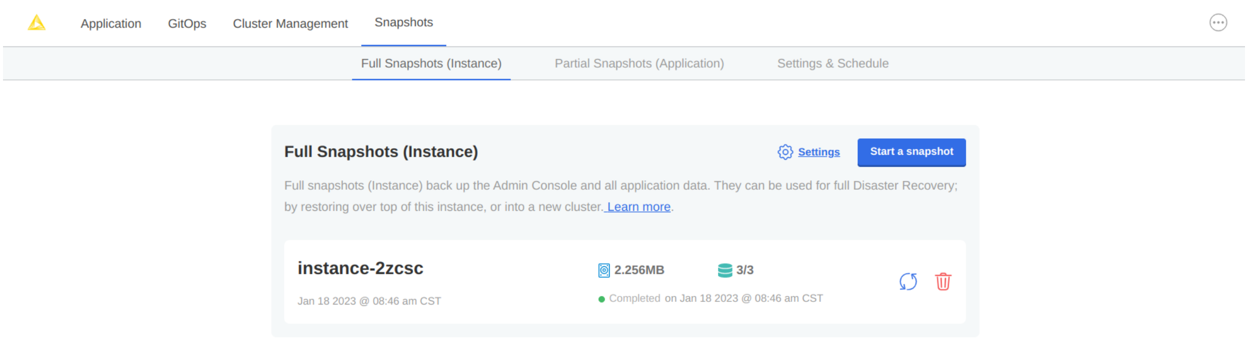
4. Next, **redeploy the Hub instance from the KOTS Admin Console** and trigger the restart of all executors by performing any change to each execution context in the Hub UI, e.g. decreasing/increasing the memory.

> It is recommended to restart all pods after the restore process and the redeployment is completed. This can be done by running the below command: `kubectl delete pods --all -n < knime hub namespace >`

1. Finally, assuming the restore completed without errors, you can verify your Hub installation is functioning as expected.

Restoring a snapshot to a **different cluster**

1. Prerequisites:
   a. Kubectl is installed and configured to access the target cluster.
   b. Helm is installed and configured to access the target cluster. Helm install guide
   c. KUBECONFIG environment variable is set to the target cluster kubeconfig file.
   d. The target cluster is running the same version of KNIME Business Hub as the cluster from which the snapshot was taken.
   e. The snapshot storage is accessible from the target cluster.
2. Configure the new cluster with the same snapshot storage settings as the original cluster, and ensure that the storage target is accessible from the new cluster.
   a. If NFS is used as the snapshot storage, ensure that the new cluster has access to the same NFS server and path as the original cluster and that the old cluster is not configured to access it anymore.
   b. If S3 is used as the snapshot storage, ensure that the new cluster has access to the same S3 bucket as the original cluster.
   c. If HostPath is used as the snapshot storage, ensure that the new cluster has access to the same HostPath as the original cluster. This can be done by either mounting the same volume or copying the snapshot data to the new cluster and configuring the HostPath volume as the snapshot storage.
3. Navigate to the list of available snapshot restore points from your Kotsadm dashboard by browsing to *Snapshots* → *Full Snapshots*. From this screen, identify the snapshot instance you would like to use, and take note of the snaphot ID (e.g. `instance-2xcsc`).



Full Snapshots (Instance)

Full snapshots (Instance) back up the Admin Console and all application data. They can be used for full Disaster Recovery; by restoring over top of this instance, or into a new cluster. Learn more.

instance-2zcsc
Jan 18 2023 @ 08:46 am CST
2.256MB
3/3
Completed on Jan 18 2023 @ 08:46 am CST

Figure 57. In this example, there is only one snapshot available and its ID is `instance-2xcsc`

A list of snapshots can also be retrieved by command line:

```
$ kubectl kots get backups
NAME              STATUS       ERRORS    WARNINGS    STARTED
COMPLETED                        EXPIRES
instance-2zcsc    Completed    0         0           2023-01-18 14:46:26 +0000 UTC
2023-01-18 14:46:53 +0000 UTC    29d
```

4. **Caution: highly destructive process**

   This next step is **highly destructive** and **irreversible**. It should only be executed if there are backups available in ***Complete*** state, to ensure that the cluster can be fully restored.

   A restore script is available to automate the restore process. The script will prepare the environment to restore the snapshot to the cluster. The script can be downloaded from the following link: Download the restore script

   To make the script executable, run the following command: `chmod +x knime-business-hub-restore-1_2.sh`

   Before proceeding with executing the script, ensure that the correct cluster is targeted and that the snapshot storage is accessible from the new cluster. This can be confirmed by running the following command: `kubectl config current-context` or checking / setting the `KUBECONFIG` environment variable.

   The script can be executed by running the following command: `bash <path to script>/knime-business-hub-restore-1_2.sh`.

5. Next, **redeploy the Hub instance from the KOTS Admin Console** and trigger the restart of all executors by performing any change to each execution context in the Hub UI, e.g.

decreasing/increasing the memory.

> It is recommended to restart all pods after the restore process and the redeployment is completed. This can be done by running the below command: `kubectl delete pods --all -n < knime hub namespace >`

6. Finally, assuming the restore completed without errors, you can verify your Hub installation is functioning as expected.

About the restore script

The restore script automates the process of restoring a snapshot to a cluster. The script performs the following steps:

1. Uninstalls the knime-hub-persitence helm chart by running `helm uninstall knime-hub-persistence -n <knime hub namespace>`.
2. Deletes all PVCs that were created by the knime-hub-persistence helm chart by running `kubectl delete pvc -n <knime hub namespace> <pvc name>`.
3. Patches all keycloakClient custom resources to remove the finalizer by running `kubectl patch keycloakClient <keycloak client name> -n <knime hub namespace> --type merge --patch '{"metadata":{"finalizers":null}}'`.
4. Deletes Istio Helm secrets with `kubectl delete secret` if Istio was deployed as part of knime-hub. This step is necessary in order to keep istiod running during the restore process to prevent resource conflicts.
5. Restores the snapshot to the cluster by running `kubectl kots restore --from-backup <snapshot ID>`.

Changelog (KNIME Business Hub 1.15)

> You can find the changelog and release notes for older version of KNIME Business Hub releases in the KNIME Business Hub Release Notes (All Versions) document.

KNIME Business Hub 1.15.1

(released July 24, 2025)

Before upgrading read the **technical changes and upgrade notes here.**

Infrastructure changes

- **restricted-v2:** Added support for restricted-v2 Security Context Constraint profile on OpenShift for new installs.

Improvements

New features

- Recycle bin
  - It is now possible to retrieve items such as workflows from a recycle bin after they have been deleted. By default, items remain in the recycle bin for 30 days.
- Enforce reset on upload
  - Allow Hub administrators to prevent uploading of sensitive data within executed workflows. This feature introduces two new KOTS admin settings. The first setting allows the administrator to define the default value of the "Reset workflow(s) before upload" option in KNIME Analytics Platform. The second option allows them to enforce workflow reset which will disable the previously mentioned checkbox and removes the Upload button from the Hub UI.
- Auditing for secret store
  - With this release, secret store will provide audit information of secret events e.g. secret creation, editing, deletion and consumption.

Additional changes

- New "At full capacity" status for execution contexts

  - When every executor in an execution context is busy and can’t accept new jobs—because existing tasks are using up nearly all CPU or memory—the context will now display "At full capacity."
- For newly created deployments, the default access role for team members is *Manage*, so that all team members can edit, share, and delete a deployment. For deployments where this is not desired, the role for team members can be switched to *Run & Inspect*.
- For newly created deployments, the default notification setting is now to send an email to the deployment creator in case the deployment fails during execution.
- Added improved navigation for data apps.

Notable bug fixes

- Fixed a bug where logging out from Hub did not work correctly.
- Fixed a bug where reloading a swapped job changed its swapped at timestamp.
- Job initiator is now recorded correctly by instrumentation service for data apps running in team scope mode.
- Fixed an issue where MinIO volumes may not be backed up correctly by Velero in version 1.15.0
- Corrected Istio image sources to avoid pulling from `gcr.io`
- Fixed misconfigured registry URL in the CoreDNS update job which impacted in-cluster DNS resolution for customers using the internal registry
- A new version of the application restore script is now available and must be used when restoring from backups on or after version 1.15.1. Find documentation about the restore script here.

New/improved Data Apps

- Improvement of the Restrict Execution Context Data App: now allows to set labels that are **not** allowed for execution on a workflow version. This allows workflow validation for shared execution contexts within a single team on the Hub without affecting workflow versions used by other teams.

Open for Innovation
KNIME

KNIME AG
Talacker 50
8001 Zurich, Switzerland
www.knime.com
info@knime.com