

Cloudera AI 1.5.4

## CDSW to Cloudera AI migration

Date published: 2022-12-14

Date modified: 2025-03-18

# CLOUDERA

<https://docs.cloudera.com/>

# Legal Notice

© Cloudera Inc. 2025. All rights reserved.

The documentation is and contains Cloudera proprietary information protected by copyright and other intellectual property rights. No license under copyright or any other intellectual property right is granted herein.

Unless otherwise noted, scripts and sample code are licensed under the Apache License, Version 2.0.

Copyright information for Cloudera software may be found within the documentation accompanying each component in a particular release.

Cloudera software includes software from various open source or other third party projects, and may be released under the Apache Software License 2.0 (“ASLv2”), the Affero General Public License version 3 (AGPLv3), or other license terms. Other software included may be released under the terms of alternative open source licenses. Please review the license and notice files accompanying the software for additional licensing information.

Please visit the Cloudera software product page for more information on Cloudera software. For more information on Cloudera support services, please visit either the Support or Sales page. Feel free to contact us directly to discuss your specific needs.

Cloudera reserves the right to change any products at any time, and without notice. Cloudera assumes no responsibility nor liability arising from the use of products, except as expressly agreed to in writing by Cloudera.

Cloudera, Cloudera Altus, HUE, Impala, Cloudera Impala, and other Cloudera marks are registered or unregistered trademarks in the United States and other countries. All other trademarks are the property of their respective owners.

Disclaimer: EXCEPT AS EXPRESSLY PROVIDED IN A WRITTEN AGREEMENT WITH CLOUDERA, CLOUDERA DOES NOT MAKE NOR GIVE ANY REPRESENTATION, WARRANTY, NOR COVENANT OF ANY KIND, WHETHER EXPRESS OR IMPLIED, IN CONNECTION WITH CLOUDERA TECHNOLOGY OR RELATED SUPPORT PROVIDED IN CONNECTION THEREWITH. CLOUDERA DOES NOT WARRANT THAT CLOUDERA PRODUCTS NOR SOFTWARE WILL OPERATE UNINTERRUPTED NOR THAT IT WILL BE FREE FROM DEFECTS NOR ERRORS, THAT IT WILL PROTECT YOUR DATA FROM LOSS, CORRUPTION NOR UNAVAILABILITY, NOR THAT IT WILL MEET ALL OF CUSTOMER’S BUSINESS REQUIREMENTS. WITHOUT LIMITING THE FOREGOING, AND TO THE MAXIMUM EXTENT PERMITTED BY APPLICABLE LAW, CLOUDERA EXPRESSLY DISCLAIMS ANY AND ALL IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO IMPLIED WARRANTIES OF MERCHANTABILITY, QUALITY, NON-INFRINGEMENT, TITLE, AND FITNESS FOR A PARTICULAR PURPOSE AND ANY REPRESENTATION, WARRANTY, OR COVENANT BASED ON COURSE OF DEALING OR USAGE IN TRADE.

# Contents

<b>Migrating Data Science Workbench to Cloudera AI.....</b>	<b>4</b>
Cloudera Data Science Workbench to Cloudera AI migration strategy.....	4
Comparison of migration tools.....	5
Impact of the migration on workload.....	6
Functional differences between CDSW and Cloudera AI.....	6
Inbuilt CDSW migration tool.....	7
Prerequisites for inbuilt CDSW migration tool.....	7
Repurposing CDSW nodes for Cloudera AI.....	9
Customizing CDSW for migrating host mounts.....	9
Incremental data migration.....	9
Validating migration readiness.....	10
Migrating with the inbuilt CDSW to Cloudera AI migration tool.....	12
Post-migration tasks with inbuilt CDSW migration tool - users, groups, and roles.....	21
Troubleshooting inbuilt CDSW migration tool.....	22
Known issues and limitations with the inbuilt CDSW migration tool.....	24
Project migration with the Cloudera AI command-line utility tool.....	25
Prerequisites for project migration with the command-line utility tool.....	25
Legacy engine migration to ML Runtimes.....	27
Migrating a project with the Cloudera AI command-line utility tool.....	28
Post migration tasks with Cloudera AI command-line utility tool migration.....	30
Troubleshooting Cloudera AI command-line utility tool.....	32
Setting up SSL certificates of source and target workbenches.....	34
Post migration tasks with CDSW migration tool and command-line utility tool.....	34

# Migrating Data Science Workbench to Cloudera AI

If you use Cloudera Data Science Workbench (CDSW) 1.10.0 and higher versions on premises, consider migrating to Cloudera AI as soon as possible. Cloudera recommends performing this migration because CDSW end-of-support is imminent.

Get familiar with the various aspects and considerations of the two migration tools before migrating CDSW to Cloudera AI.

## Cloudera Data Science Workbench to Cloudera AI migration strategy

Two approaches exist for migrating Cloudera Data Science Workbench (CDSW) to Cloudera AI on premises. The details on these approaches help you determine the more suitable migration tool for your needs.

The following approaches are available:

- CDSW migration tool
- Project migration command-line utility tool

### CDSW migration tool

The CDSW migration tool, built in the Cloudera AI on premises, completes a cluster-level replication of the entire CDSW cluster to a new Cloudera AI Workbench. When the migration is complete, users of the CDSW cluster have a cutover to the Cloudera AI Workbench at once.



**Note:** Only migration to Cloudera AI on Cloudera Embedded Container Service is supported. Migration to Cloudera AI on OpenShift is not supported.



**Note:** Using the CDSW migration tool requires downtime for the CDSW cluster when performing the final migration.

The migration process has the following main phases:

1. The user starts the migration in the Cloudera AI User interface.
2. When the migration is triggered for the first time, the replication copies the project files from the source CDSW to the target Cloudera AI Workbench in the background. You can use the source CDSW cluster during the replication.
3. Check Migration Readiness to verify that CDSW and Cloudera AI are prepared for the migration.
4. Once the initial migration is complete, the target Cloudera AI Workbench transitions to the Validation Started state, indicating that it is now available. At this stage, you can begin validating the projects within Cloudera AI.
5. The migration can be triggered multiple times to replicate the changes from the source CDSW to the target Cloudera AI Workbench. Note that any changes done in the target Cloudera AI Workbench will be overwritten.
6. Once you are ready for the cutover, a final migration can be triggered which stops the source CDSW cluster for the final replication. After this point, the source CDSW cluster will be inaccessible.

### Project migration command-line utility tool

The Cloudera AI command-line utility tool is a Cloudera open-source tool that handles migration at project level. Although this approach has lower risks than the CDSW migration tool, it requires more effort in setting up the target Cloudera AI Workbench and the migration process is likely to take longer.

The Cloudera AI command-line utility tool is built using Python and it requires Python 3.10. This tool must be installed on a staging host, referred to as bastion host. This host requires considerations on both the source CDSW and the target Cloudera AI cluster.

The concept of the Cloudera AI command-line utility tool is based on exporting projects from the source CDSW and importing them to the target Cloudera AI Workbench.

The tool is designed to be used by the Data Scientists who can migrate their own projects. However, Administrators can also use the tool to perform batch migration.

This tool is not limited to CDSW migration. It can also handle migration from Cloudera Machine Learning to Cloudera AI or from Cloudera AI on premises to Cloudera AI on cloud.

### Related Information

[Inbuilt CDSW migration tool](#)

[Project migration with the Cloudera AI command-line utility tool](#)

## Comparison of migration tools

Use the table and the hints to decide which migration tool is more suitable for you.

Consider the following main differences:

- The CDSW Migration tool approach requires a cutover of all CDSW users to Cloudera AI at once. In case of a large number of users and projects, this approach might pose a huge risk. In case of unexpected critical issues, rollback to CDSW is possible. However, the migration must be restarted from the beginning using a new workbench, since we cannot trigger a migration for the migrated workbench once it has completed the final migration.
- The Cloudera AI command line utility tool allows migration of users and their projects in batches. This allows you to perform the migration at your own pace. However, more effort might be required to coordinate with the users.

**Table 1: Comparison of migration tools**

	Cloudera AI command-line utility	CDSW Migration tool
Migration approach	Project level	Cluster level
Migration effort	Progressive High effort required Anything outside the projects need to be handled manually.	Quick, complete Low effort required Migration is fully automated.
Downtime	No	Yes During final migration
What is migrated		
Site-level settings (security, environment variables, quotas and so on)	✗	✓
Users	✗	✓ *
Runtime images	✗	✓
Projects (models, jobs, applications, environment variables)	✓	✓
* The CDSW migration tool replicates the CDSW PostgreSQL Database to the Cloudera AI Workbench database, which includes the user accounts. However, you will still need to create the users or groups on the Cloudera Management Console on premises (that is, User Management Service) to grant access to the migrated Cloudera AI Workbench.		

## Impact of the migration on workload

Learn about potential impacts to your existing workload running on CDSW when migrating to Cloudera AI.

The following aspects can have impact on your workload:

### Spark on Kubernetes and Spark Pushdown

By default, Cloudera AI runs Spark on Kubernetes which is very different from CDSW. Project owners or users can enable the Spark Pushdown feature on Cloudera AI project level which allows the Spark workload to run on the Cloudera Base on premises YARN. However, enabling this feature is optional and cannot be enforced.

The risk of not enabling the Spark Pushdown feature is that users might unintentionally run their Spark on Kubernetes and use up most of the cluster resources if no quotas are set.



**Note:** Cloudera AI does not support Spark Pushdown when using legacy engines.

### End of Life (EOL) Python versions

Python 2.x, 3.6 and 3.7 have already reached EOL. The ML Runtimes for these Python versions are not supported anymore, but they still work and can be used in Cloudera AI. However, no security fixes for vulnerabilities are available in those Python versions.

### Validation and Testing

Cloudera AI projects must be validated and tested before formally migrating them to Cloudera AI. Pay special attention to the following issues:

- Projects that are switching to different images post migration, for example, from legacy to runtime
- Spark pushdown requires additional settings to be added to the spark-defaults.conf file

### Legacy engines are deprecated

While legacy engines can still be used, Cloudera recommends migrating to ML Runtimes as soon as possible.

## Functional differences between CDSW and Cloudera AI

Consider the following key differences between CDSW and Cloudera AI in some of the functions.

- CDSW has a Host Mount feature that allows mounting a local directory on CDSW host into the user sessions. A popular usage of the feature is to push files to users in sessions.

This feature is not available in Cloudera AI but consider using [Custom Runtime addons with Cloudera AI](#) as a replacement. When you create a new Runtime Addon, Cloudera AI creates a directory on the NFS share directory, for example, (nfs://<nfs-share>/addons/custom-addon-<myaddon>/...) which gets mounted in all the user sessions. You can then share the files by copying them to the custom addon directory on the NFS.



**Note:**

This share is mounted as read-only in the user sessions, that is, users can only read but cannot write to this directory. This is a key difference to host mounts, which are writable in CDSW.

- Only the Spark Pushdown feature works with external shuffle service, Spark on Kubernetes supports external shuffle service.
- In Cloudera AI the following Spark properties are hardcoded to true value:
  - spark.authenticate=true
  - spark.io.encryption.enabled=true
  - spark.network.crypto.enabled=true

Use the spark-defaults.conf file of the project, if you need to override any of the settings.

## Inbuilt CDSW migration tool

If you use Cloudera Data Science Workbench (CDSW) 1.10.0 and higher versions on premises, consider migrating to Cloudera AI as soon as possible. Cloudera recommends performing this migration because CDSW end-of-support is imminent.

Before you begin the migration tasks, you can optionally repurpose your CDSW hosts instead of adding hardware for installing Cloudera AI on premises. Then, you install Cloudera AI on the hosts. Finally, the UI-driven migration tool runs scripts in the background to migrate your workload automatically after installing Cloudera AI on the cluster with your deployed CDSW.

## Prerequisites for inbuilt CDSW migration tool

Before migrating from Cloudera Data Science Workbench (CDSW) to Cloudera AI in Cloudera on premises, you must meet a number of prerequisites to succeed. A prerequisite for migration is the installation of Cloudera AI on your Cloudera Data Science Workbench base cluster.

### About this task

The following table presents the supported migration version combinations for Cloudera Data Science Workbench and Cloudera AI:

**Table 2: Supported migration versions for Cloudera Data Science Workbench and Cloudera AI**

CDSW versions supported for migration	Target Cloudera AI on premises version
CDSW 1.10.0- 1.10.4	1.5.5 SP1 (recommended version)
Upgrade to CDSW 1.10.5 before migrating to Cloudera AI.	1.5.5 CHF1
	1.5.4 SP2 CHF1
CDSW 1.10.5	1.5.5 SP1 (recommended version)
	1.5.5 CHF1
	1.5.4 SP2 CHF1

Migration from CDSW, configured with LDAP, SAML, or LOCAL authentication, to Cloudera AI, is supported, the automatic migration is supported only if CDSW is running with LDAP or SAML. The migration process does not automatically migrate your authentication configurations. Therefore, setting up LDAP or SAML in CDSW before the migration is part of the migration procedure.

The migration does not migrate your CDSW endpoint connections. Therefore, post-migration instructions include setting up LDAP, SAML endpoint connections, and DNS on Cloudera AI, so you can upload them after migration to Cloudera AI.

### Procedure

1. Ensure you have a CDSW 1.10.0 or higher version cluster in Cloudera. Otherwise, choose one of the following options:
  - If you have a CDSW installation in either CDH or HDP, migrate to on premises 1.5.1 or higher versions, and then migrate CDSW to Cloudera AI.
  - If you have CDSW installation earlier than 1.10.0, upgrade to CDSW 1.10.0 or higher versions.
2. Ensure that LDAP or SAML is configured in your CDSW cluster on Cloudera. If LDAP or SAML is not yet configured, set up LDAP or SAML before the pre-migration tasks. For guidelines on setting up LDAP and SAML, see [Configuring External Authentication with LDAP and SAML](#).

The migration process cannot succeed without authentication.

3. Meet the Cloudera AI software requirements for on premises, including storage, for installing Cloudera AI on Cloudera on premises 1.5.1 or higher versions. For Cloudera AI on premises software requirements, see [Cloudera AI software requirements for Cloudera](#).
4. Backup CDSW data. For instructions on backing up CDSW data, see [Backup and Disaster Recovery for Cloudera Data Science Workbench](#).
5. In CDSW, export your Grafana dashboards. For instructions on exporting Grafana dashboards, see [Export and import | Grafana documentation](#).
6. Take notes of the connections of endpoints in your CDSW cluster and consider your custom settings. You must use this information after migration to set up endpoints in your on premises cluster.
7. Take notes of your custom settings, if you have customized your DNS configuration, to be able to customize your DNS configuration after migration.  
If you did not customize your DNS configuration, the migration tool configures DNS in your on premises cluster.
8. Gather information about your LDAP or SAML configurations on CDSW.  
After migration, you must set up LDAP or SAML again on the Cloudera AI cluster as the LDAP or SAML configuration is not migrated.
9. In CDSW, manually back up the custom DNS configuration for Kube-DNS, and then migrate your custom configuration to Cloudera AI.  
Cloudera AI uses the core-DNS, which is incompatible with the CDSW Kube-DNS.
10. In Cloudera Manager, select Install and Upgrade to Cloudera on premises 1.5.4 or higher versions using the Cloudera Embedded Container Service on your CDSW cluster.



**Note:** Migration of your CDSW workloads to Cloudera AI on OpenShift is not supported.

11. During the installation of Cloudera Data Services on premises using the Cloudera Embedded Container Service, if you select Airgap, set up a network connection between CDSW and the Cloudera on premises cluster.
12. Enable the Cloudera AI features during installation that you were using in CDSW.  
For example, enable model metrics and monitoring.

Production Cloudera AI

☐ Enable Governance ⓘ

☒ Enable Model Metrics ⓘ

Other Settings

☐ Enable TLS ⓘ

☒ Enable Monitoring ⓘ

If you do not enable the same, or similar, Cloudera AI features during installation that you were using in CDSW, you will not be able to use the Cloudera AI features.

### Related Information

[Configuring External Authentication with LDAP and SAML](#)

[Cloudera Upgrade and Migrations Paths](#)

[Cloudera AI on premises software requirements](#)

[Backup and Disaster Recovery for Cloudera Data Science Workbench](#)

[Export and import | Grafana documentation](#)

[Installation using the Embedded Container Service](#)

[Installing Cloudera Data Services on premises using Cloudera Embedded Container Service](#)

[Configuring your enterprise IdP to work with Cloudera as a service provider](#)



## Repurposing CDSW nodes for Cloudera AI

If you need to repurpose any CDSW nodes for Cloudera AI, several preparatory tasks must be completed before initiating the CDSW-to-Cloudera AI migration.

While repurposing CDSW nodes is possible, you must ensure that CDSW remains operational during the process. For details on repurposing CDSW nodes, see [Repurposing Cloudera Base on premises Nodes for Cloudera Data Services on premises on Cloudera Embedded Container Service](#).

## Customizing CDSW for migrating host mounts

For security reasons, Cloudera AI does not allow you to mount a directory directly from hosts. You must customize Cloudera Data Science Workbench (CDSW) runtime to make contents of a directory available to your Cloudera AI workloads.

### About this task

Before migration, you must perform pre-migration steps if you have mounted additional dependencies from the host in CDSW. For example, you might have mounted directories containing libraries needed for running your workloads. You must make these libraries available in Cloudera AI. In the pre-migration steps, you can set up CDSW for the migration to mount your libraries in Cloudera AI.

If you loaded libraries from a host mount in the past, Cloudera recommends creating a custom runtime in CDSW, and to change the project to use the new custom runtime, and then perform the migration. However, for anything other than the libraries, load the data to all the sessions in Cloudera AI using the custom Runtime addons procedure after migration to mount data in all the workloads in Cloudera AI. Custom runtime addons do not allow writing to the file system as the host mount in CDSW does.

### Procedure

1. Create a customized ML Runtime.
2. If libraries were loaded from the host mount, configure your CDSW project to use the custom runtime by adding the custom Runtime to CDSW before migration.



**Note:** After migration, use the custom Runtime Addons procedure to mount anything other than libraries.

Libraries you add to the custom Runtimes will be available to the Cloudera AI projects using that custom Runtime.

## Incremental data migration

From Cloudera on premises 1.5.1, data migration over multiple iterations is enabled instead of migrating in one single step. The migration tool does not stop the CDSW during the incremental migration activities.

After each incremental migration, the administrator can access the Cloudera AI Workbench and validate the migrated workloads or start new workloads in Cloudera AI as part of the validation.



**Note:** Any change in the Cloudera AI Workbench will be overwritten during the incremental CDSW to Cloudera AI migration. If the administrator finds any issues in the Cloudera AI Workbench, the issue has to be fixed in the CDSW, and the changes will be copied during the incremental migration.

Incremental migration can be started by selecting the Incremental CDSW migration option from the Actions menu of the workbench.



**Note:** During the incremental migration, a small window occurs when the new model build operation does not work. If you see any failures with the model build, restart the model builds after a short waiting time.

The incremental migration consists of the following phases:

1. Migration of CDSW Kubeconfig.

This is an optional parameter, therefore it is an optional step.

## 2. Final migration.

Select this option if the Cloudera AI validation is finished and all workloads work as expected. Once the final migration is completed, the CDSW will be in a stopped state.

- When performing the final migration, you can choose the following additional options:

- Stop Applications.

The Applications in the Cloudera AI are stopped after the migration. You have to start each application manually after the final migration.

- Stop Jobs.

The recurring jobs in the Cloudera AI are paused after the migration. You have to start each job manually after the final migration.

- Stop Models.

The models are stopped after the migration, and you have to start the models manually after the final migration.

Once all the validation is finished, the Administrator can perform the final migration. Following the final migration, the CDSW is in a stopped state, and you can start using the Cloudera AI.

## Validating migration readiness

From Cloudera on premises 1.5.1, you can validate CDSW to Cloudera AI migration readiness.

### Procedure

1. Select the **Validate Cloudera AI for CDSW migration readiness** button.

The validation page displays.

Provide the migration settings and the workbench details to complete a preflight check validation. This workflow creates a Cloudera AI on premises cluster and runs the preflight checks.

The migration readiness check summary can be viewed in the **Details** tab of the workbench page.

2. Complete at least one incremental migration in the workbench.

### Results

If the migration readiness check fails, the workbench will be in the Migration readiness check failed state. You can retry the migration readiness check again in this cluster though.

### Checks performed before migration

Consider the list of issues and workarounds prior to migration.

### Configuration readiness

In CDSW, Kube-DNS is used for handling the DNS, while in Cloudera AI, core DNS is used. If you add any custom kube-DNS configuration to the CDSW, those configuration details will not be copied to the Cloudera AI automatically. Checking this flag and verifies if any custom configuration is added to CDSW.

Troubleshooting failures

Check if any custom configurations are added to the kube-DNS and if the custom configurations are necessary.

### Registry readiness

Verify the registry readiness in Cloudera AI, by checking if the created Cloudera AI cluster has proper registry permissions.

Troubleshooting failures

Check if the `cdp-private-installer-embedded-registry-docker-pull-secret` secret, in the control plane namespace is present in the Cloudera AI.

If the above configuration is present, check if the docker configuration in this secret is correct.

### Host Mount readiness

CDSW supports the Host Mount feature, however, this is not supported in Cloudera AI. Migrate the workloads containing the host mounts is not possible.

Troubleshooting failures

The workloads containing the host mounts must be modified in CDSW before the migration. Convert the workloads using the engines to ML Runtimes.

### Engine type readiness

The legacy engine is deprecated in Cloudera AI. If the CDSW contains any workloads using the engine or custom engines, those workloads must be converted to ML Runtimes before the migration.

Troubleshooting failures

Convert the engine-based or custom engine-based workloads to runtime-based workloads.

### NFS filesystem readiness

The CDSW to Cloudera AI migration tool supports migrating project files from the CDSW internal NFS server to the Cloudera AI project storage. Checking readiness verifies the NFS filesystem size for the migration.

Troubleshooting failures

Check the storage configuration of the NFS storage in the Cloudera AI. Ensure enough storage is in the NFS storage to perform the migration.

### Runtime Addons' readiness

Cloudera AI supports numerous ML Runtime addons such as Spark and Hadoop CLI. This check verifies that all the ML Runtime addons are installed in the Cloudera AI properly.

Troubleshooting failures

In the Cloudera AI UI, go to `Site Administrator- Runtime` and check the status of the ML Runtime addons. If any of the ML Runtime addons is in the wrong state, select the `Reload` action from the Actions drop-down list.

### Service readiness

Check and verify that all the services active in CDSW are enabled and started in Cloudera AI.

Troubleshooting failures

Ensure that the necessary services are started in the Cloudera AI during the workbench provisioning.

### Versions readiness

CDSW to Cloudera AI migration is supported only from CDSW version 1.10.0. This check verifies the source CDSW version.

Troubleshooting failures

Update the CDSW to version 1.10.0.

## Migrating with the inbuilt CDSW to Cloudera AI migration tool

Learn about how to migrate Cloudera Data Science Workbench (CDSW) 1.10.0 and higher versions on premises to Cloudera AI. The migration process is streamlined using a UI-driven migration tool, which automatically transfers your workload from the deployed CDSW instance installed on the same cluster as Cloudera AI.

### About this task

You can automatically migrate CDSW 1.10.0 or a later cluster to Cloudera AI on premises 1.5.4 or higher versions. You can expect some downtime, which is proportional to the volume of the workloads you have to migrate.

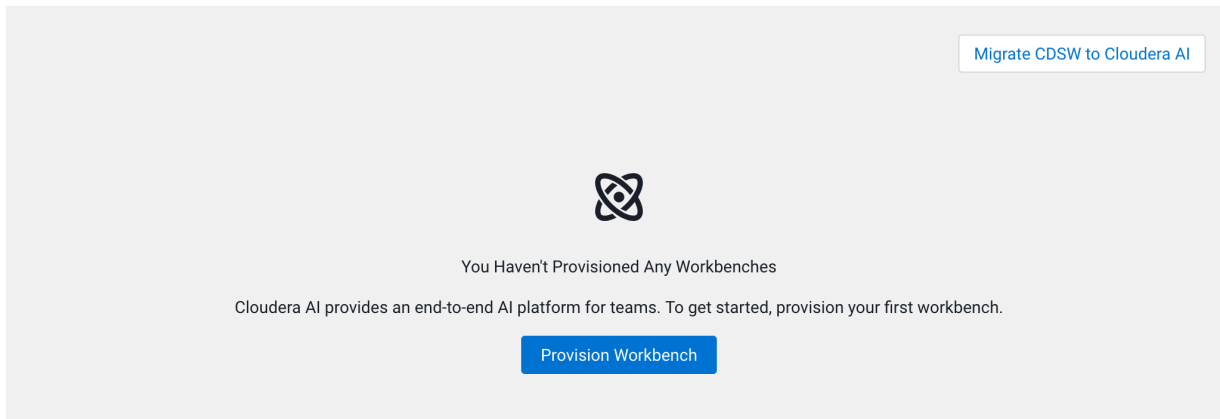
### Before you begin

Perform the migration readiness checks before starting migration.

### Procedure

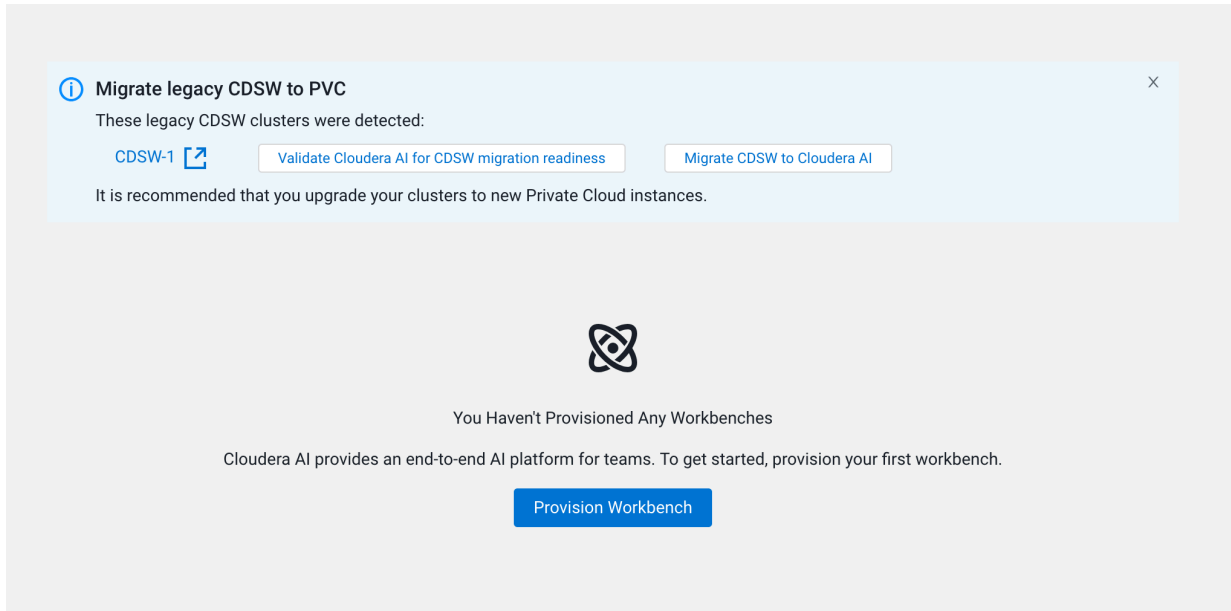
1. Log into Cloudera on premises, and navigate to Cloudera AI Workbenches .  
The system detects the presence of your legacy CDSW installation and provides the Migrate CDSW to Cloudera AI button.

Cloudera AI Workbenches



2. Click the Migrate CDSW to Cloudera AI button.

The Migration tool provides an option to Validate Cloudera AI for CDSW migration readiness or just continue with the migration.



3. Select an option to proceed.

- Validate Cloudera AI for CDSW migration readiness - The readiness or preflight validation creates a new workbench and runs readiness validation on the workbench before performing the migration.
- Migrate CDSW to Cloudera AI - The Migration tool provides a Cloudera AI Workbench provision window with additional options.

4. If you choose the Validate Cloudera AI for CDSW migration readiness option, the Migration tool displays the validation page.

Migrate CDSW to Cloudera AI

### Provision Cloudera AI Workbench

Provision an on-demand Cloudera AI Workbench.

#### Migration Settings

Source Name: CDSW-1

Source Cluster: Cluster 1

\* Kubeconfig

☒ File Upload ☐ Direct Input

[Choose File](#)

This file can be found at /etc/kubernetes/admin.conf on your CDSW cluster at host host-1.cdswe-sap.kc1oud.cloudera.com.

Migration timeout (in hours)

0 24(1 day) 336(14 days)

\* Workbench Name

\* Select Environment

Select Environment

Environment type: None selected

\* Namespace

NFS Server

☐ Internal ☒ External

This selection uses an external NFS export path (or a subdirectory within it).

\* Existing NFS

**Note:** An administrator must run `chown 8536:8536` on the NFS directory.

The directory must be empty and not used by another workbench.

NFS Protocol version

4.1

**Production Cloudera AI**

☐ Enable Governance

☒ Enable Model Metrics

**Other Settings**

☐ Enable TLS

☒ Enable Monitoring

☐ Skip Validation

Cloudera AI Static Subdomain

- a) To provide a Kubeconfig file for the migration check, click the File Upload button, then Choose File, and select the Kubeconfig file.

The Kubeconfig file can be found at the /etc/kubernetes/admin.conf file on the CDSW cluster.

If you cannot access the /etc/kubernetes/admin.conf file from the UI as instructed, download the file from your CDSW cluster to your local machine, and then try to select the Kubeconfig file from the UI again.

- b) In the Migration timeout section, accept the default 24 hours timeout, or if your CDSW workload is the size of hundreds of gigabytes, increase the migration time up to 336 hours, that is 14 days.

Increasing the migration timeout value does not cause a delay in the migration of a small workload.

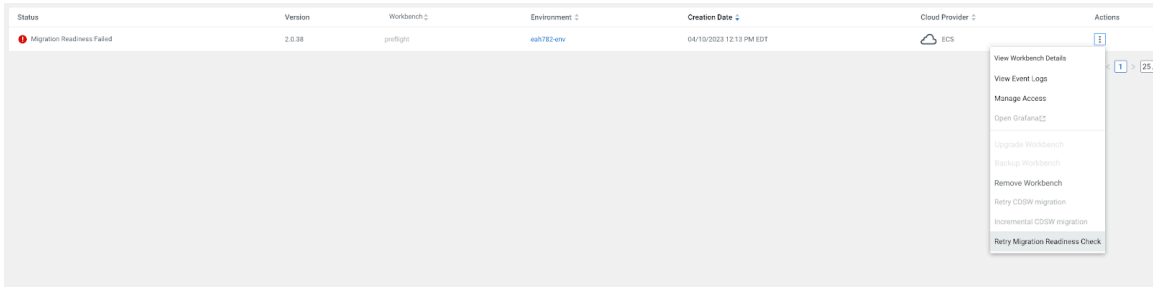
- c) In the Workbench Name field, type an arbitrary name.
- d) In the Select Environment field, select your Cloudera environment.

After the readiness validation has completed, the Migration tool displays a status. You can see the readiness validation summary in the Workbench Details page.

- If the readiness validation fails, you can obtain additional information about the failure on the Workbench Details page.

Status	Version	Workbench	Environment	Creation Date	Cloud Provider	Actions
Migration Readiness Failed	2.0.38	prof1gts	env782-env	04/10/2023 12:13 PM EDT	AWS	

- After you have addressed the issue resulting in the failed readiness validation, you can retry the readiness validation by choosing the Retry Migration Readiness Check option from the Actions drop-down list.



- e) After the readiness validation, incremental migration can be performed to continue the migration operation.

This performs workbench creation and the migration readiness check. If the migration readiness is successful, the migration process will continue with the incremental migration automatically.

## 5. Check the Migration Settings window.

When you proceed with the CDSW to Cloudera AI migration, the Migration tool displays the Migrations Settings window.

Migrate CDSW to Cloudera AI

### Provision Cloudera AI Workbench

Provision an on-demand Cloudera AI Workbench.

#### Migration Settings

Source Name: CDSW-1

Source Cluster: Cluster 1

\* Kubeconfig

☒ File Upload ☐ Direct Input

This file can be found at `/etc/kubernetes/admin.conf` on your CDSW cluster at host `host-1.cdsu-sep.kccloud.cloudera.com`.

Migration timeout (in hours)

0 24(1 day) 336(14 days)

\* Workbench Name

\* Select Environment

Select Environment

Environment type: **None selected**

\* Namespace

NFS Server

☐ Internal ☒ External

This selection uses an external NFS export path (or a subdirectory within it).

\* Existing NFS

**Note:** An administrator must run `chown 8536:8536` on the NFS's directory. The directory must be empty and not used by another workbench.

NFS Protocol version

**Production Cloudera AI**

☐ Enable Governance

☒ Enable Model Metrics

**Other Settings**

☐ Enable TLS

☒ Enable Monitoring

☐ Skip Validation

Cloudera AI Static Subdomain

- a) To provide a Kubeconfig file for the migration, click the File Upload button, then Choose File, and select the Kubeconfig file.

The Kubeconfig file can be found at the `/etc/kubernetes/admin.conf` file on the CDSW cluster.

If you cannot access the `/etc/kubernetes/admin.conf` file from the UI as instructed, download the file from your CDSW cluster to your local machine, and then try to select the Kubeconfig file from the UI again.

- b) In the Migration timeout section, accept the default 24 hours timeout, or if your CDSW workload is the size of hundreds of gigabytes, increase the migration time up to 336 hours, that is 14 days.

Increasing the migration timeout value does not cause a delay in the migration of a small workload.

- c) In the Workbench Name field, type an arbitrary name.
- d) In the Select Environment field, select your Cloudera environment.
- e) Accept default values for other options, and click the Provision Workbench button.

After the Cloudera AI installation, the migration readiness checks and the migration follow automatically.

Status indicators show the progress of the installation and migration. During the migration, you can access the CDSW cluster. The migration process does not stop CDSW pods. The Cloudera AI Workbench is stopped.





**Note:** Any changes to CDSW during the migration will not be copied to Cloudera AI. These changes will be copied in subsequent incremental migrations.

- f) Check the status of the migration.

When the initial migration is complete, the state changes to Validate Migration Started.

<div>  One or more workbenches are in 'Validate Migration Started' state. Any changes done to the workbench will be overwritten during the incremental migration. </div>						
<input type="text" value="Search Workbenches"/>		Environment: <span>All</span>	<div> <a href="#">Provision Workbench</a> </div>			
Status	Version	Workbench	Environment	Creation Date	Cloud Provider	Actions
Validate Migration Started	2.0.39	<a href="#">aqs8z4migration1</a>	<a href="#">aqs8z4-env</a>	04/26/2023 6:25 PM EDT	ECS	
<div>           Displaying 1 - 1 of 1           <span>&lt; 1 &gt;</span> <span>25 / page</span> </div>						

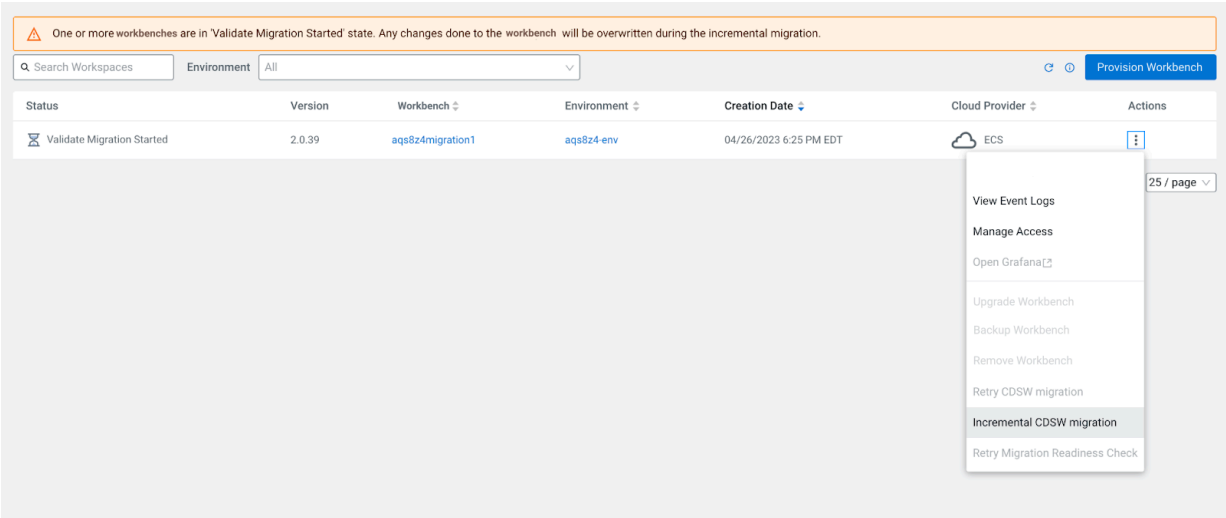
- g) Open the workbench by clicking the workbench name and validate the workloads.

Any changes made to the workbench while you are validating the workbench will be overwritten during the incremental migration and the final migration.

6. At this point, you can choose to perform multiple incremental migrations or a single, final migration.

- To perform incremental migrations, select the Incremental CDSW Migration option from the Actions menu.  
CDSW is not stopped during incremental migrations.
- To perform a single, final migration, select the Incremental CDSW Migration option from the Actions menu, select the Perform final migration checkbox, then click the OK button.

During the final migration, CDSW will be stopped and will not be restarted. After the final migration, only the Cloudera AI Workbench will be active.



The Migration tool displays the Incremental CDSW migration dialog box.

### Incremental CDSW migration ✕

Incremental CDSW migration to workbench : m4396nmigrate

Kubeconfig ⓘ

☒ File Upload ☐ Direct Input

Choose File

Migration timeout (in hours) ⓘ

02448

☐ Perform final migration

Cancel

OK

7. If you choose to perform incremental migrations, choose your parameters for the migrations.

**CDSW Kubeconfig**

This is an optional parameter. If no change to the CDSW Kubeconfig occurs, then you do not need to select this option. The system will use the Kubeconfig that was provided during the initial migration. This option can be helpful if CDSW is restarted during the migration and changes to the CDSW Kubeconfig occur.

**Migration timeout**

You can specify the amount of time allowed for the migration before it reaches a timeout. This timeout value is specified in hours and can range from 0 to 48 hours.

8. If you choose to perform incremental migrations, when the Cloudera AI validation is complete and all workloads work as expected. Select the Perform final migration checkbox and choose the appropriate option.

### Incremental CDSW migration ✕

Incremental CDSW migration to workbench : m4396nmigrate

Kubeconfig ⓘ

☒ File Upload ☐ Direct Input

Choose File

Migration timeout (in hours) ⓘ

0

24

48

☒ Perform final migration

☐ Stop applications

☐ Stop jobs

☐ Stop models

⚠

Do final migration of CDSW to this Cloudera AI workbench. In this case, source CDSW workbench will be tuned off after migration and workloads on migrated Cloudera AI workbench will start running.

Cancel

OK

After you perform the final migration, CDSW will be in a stopped state.

You can choose any of the following options for the Perform final migration checkbox:

**Stop applications**

If you select the Stop applications option, the applications in the Cloudera AI will be in the stopped state after the migration. You must start each application manually after the final migration.

**Stop jobs**

If you select the Stop jobs option, the recurring jobs in the Cloudera AI will be in the paused state after the migration. You must restart each job manually after the final migration.

### Stop models

If you select the Stop models option, the models in the Cloudera AI will be in the stopped state after the migration. You must start each model manually after the final migration.

9. To display the progress of the migration including events and logs while the workbench is in migration mode, navigate to the Workbench Details page and click the Migration Progress tab.

You can also view the details of the migration, events and logs by clicking the appropriate tabs.



## Results

Now that the migration is complete, you can use Cloudera AI

## Post-migration tasks with inbuilt CDSW migration tool - users, groups, and roles

After migrating Cloudera Data Science Workbench (CDSW) 1.10.0 or higher versions to Cloudera AI, you must perform several tasks before moving users to Cloudera AI.

### About this task

This task is about assigning the user, group and the roles in the on premises control plane.



#### Note:

The migration support from CDSW configured with SAML is available from Cloudera AI on premises 1.5.5 SP1 or higher releases.

The CDSW to Cloudera AI migration in Cloudera Manager updates the docker registry, engines, and model builds. You can assign roles to users and groups, import Grafana dashboards you previously exported, configure endpoints and DNS resolution, and configure LDAP or SAML.

### Procedure

1. In Cloudera Management Console, click **User Management**, and select **Users**.
2. Upload users by selecting the **Upload Users** action from the **Actions** drop-down list.
3. Select **Groups**, click the **Create Groups** button, and create user groups.
4. Start the Cloudera AI Workbench on the Cloudera AI cluster, and check workloads. Start new sessions, jobs, models, and applications. Try starting existing workloads that were migrated from CDSW.
5. Import the Grafana dashboards you exported earlier.
6. Configure cluster endpoint connectivity per the information about required connections you had in your CDSW cluster.

7. If you customized your DNS configuration on CDSW, manually configure your DNS in your on premises cluster.  
If you did not customize your DNS configuration, the migration tool sets up the default DNS configuration in your on premises cluster.
8. Configure LDAP or SAML on Cloudera AI, and grant user access on Cloudera AI .  
Cloudera AI on premises supports LDAP or SAML.
9. Decommission the CDSW cluster, and give users access to Cloudera AI.

### Related Information

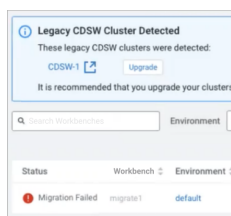
[Configuring your enterprise IdP to work with Cloudera as a service provider](#)

## Troubleshooting inbuilt CDSW migration tool

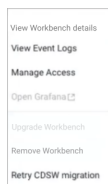
If your migration from Cloudera Data Science Workbench (CDSW) to Cloudera AI fails, you can restart migration. Also, learn how to find logs for debugging migration and other issues.

### Failed Migration

Problem: In the event your migration fails, the Migration failed indicator appears.



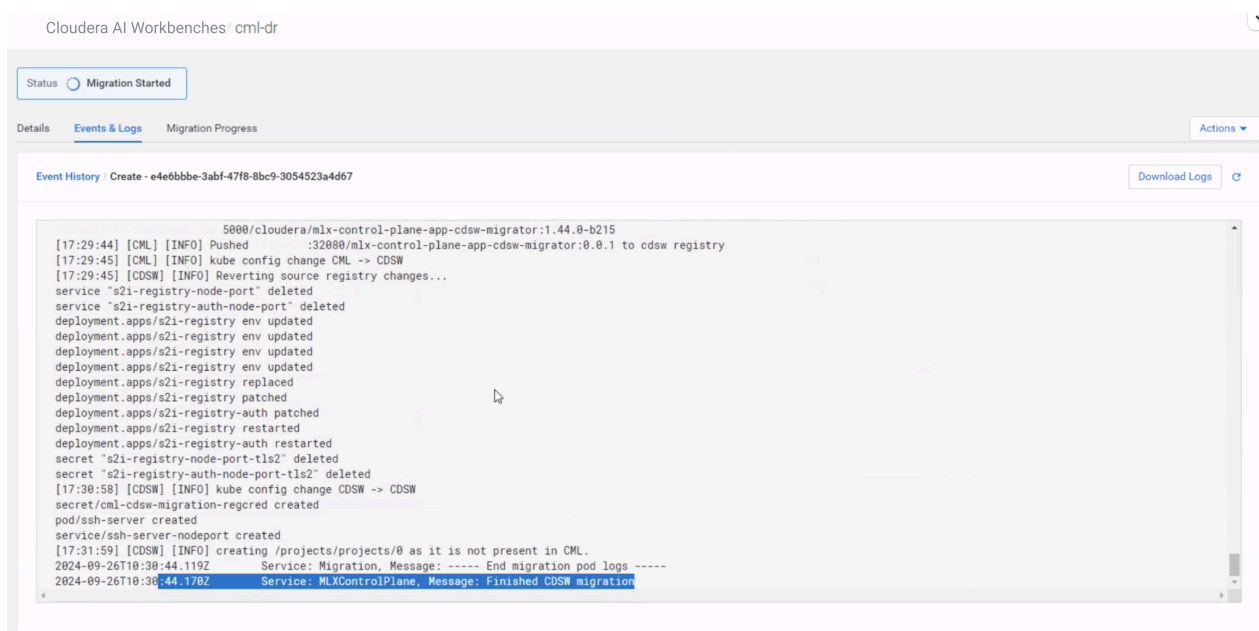
Solution: In workbenches, click the **Options** icon , and select the Retry CDSW Migration option:



### CDSW to Cloudera AI migration frozen with large Projects

Problem: If the Project being migrated from CDSW to Cloudera AI is large, it can get stuck, frozen, while waiting for the Project information to get migrated. After a while, the logs display that the migration was successful, however the UI is still unavailable.

**Figure 1: Successful migration with UI still unavailable**



Solution: Instruct Cloudera manually that the migration has finished:

1. Find the Customer Resource Number (CRN) of the new workbench from the Control Plane.
2. Open a shell into the cdp-embedded-db-0 pod, which is in the Cloudera Data Platform namespace. This can be done from the Kubernetes dashboard or using the kubectl command.
3. Add a new entry into the database that tells the system that the migration is completed.

Example:

```


psql
\c db-mlx
INSERT INTO event(instance_id, resource_type, status, operation, user_id)
SELECT id, 'mlx_instance', 'started', 'validateMigration', creatorcrn f
rom mlx_instance where mlx_instance.crn='##<crn>###';
  
```

## Result

The Cloudera UI is updated, and displays the information that the migration must be validated. You can continue the migration process.

## Viewing logs

Problem: If your migration fails, and the restarted migration also fails, you must get information about the failure.

Solution: In workbenches, click the Options icon , and select the View Event Logs option.

## Problem when migrating parcels to a new Cloudera AI Workbench

If parcels are located in a custom directory specified by parcel\_repo\_home in Cloudera Management Console, instead of the default (/opt/cloudera/parcels) location, then the CDSW to Cloudera AI migration script will not find the parcels.

Workaround:

1. Check that the Cloudera Embedded Container Service parcels are located at the /opt/cloudera/parcels directory and not at a custom location.

2. If not, then copy the existing parcels to the default location by using the following command:

```
cp -rp <custom-ecs-parcel-path>/ECS* /opt/cloudera/parcels/
```

## Known issues and limitations with the inbuilt CDSW migration tool

Review the known issues and limitations for the inbuilt CDSW migration tool.

### Unsupported CDSW features

- CDSW supports the Host Mount feature, however it is not supported in Cloudera AI.

Workaround:

Use custom Runtime addons as an alternative.

- Migration is supported only from a workbench configured with LDAP authentication.
- Custom configurations, such as Host, Kubernetes and DNS in the CDSW environment will not be copied by the migration tool. The configuration details must be identified in the migration planning, documented before the migration and must be copied manually after migration.
- Migrating the sessions created with a custom engine in CDSW will not work in the migrated Cloudera AI as the engine architecture differs for Cloudera AI and CDSW. You must move from the custom engine to a custom ML Runtime before the migration.
- If the **Allow containers to run as root** option is selected in CDSW Administration Security, and this workbench is migrated to Cloudera AI, the selected option will cause Spark job errors.

This setting has been removed from Cloudera AI, therefore if the value is set in the migrated CDSW it can lead to undefined behaviour.

Do not copy this setting when migrating to Cloudera AI.

Workaround

You can manually configure the Cloudera AI master node, and restart the relevant pods:

1. Access the db-0 pod and db container in the migrated workbench from the Kubernetes dashboard by using shell, in the Cloudera Management Console.
2. Run PostgreSQL.
3. Run the following command:

```
\c sense
```

4. Run the following command:

```
sense=# update site_config set pod_eval_allow_root=false;
UPDATE 1
```

5. Delete the two evaluator pods, so that the pod-evaluator component is restarted.
6. Restart web pods.

### Migration from CDSW to Cloudera AI using SAML

The migration from CDSW to Cloudera AI, using SAML is supported with the following limitations:

- The NameID at the IdP level must be transformed to ensure it does not include special characters.

Cloudera AI imposes a user identifier limitation that requires this specific format. This typically involves applying a transformation rule within your IdP to remove characters such as @, \_, or ., or to encode them in a format that Cloudera AI can process correctly. For instance, a user's email address, such as john.doe@example.com, could be transformed into johndoeexamplecom or a similar format.



- To configure group membership, you must map the attribute containing the user's groups to the specific object identifier (OID) `urn:oid:1.3.6.1.4.1.5923.1.5.1.1`, which is treated as the definitive attribute for group membership by both Cloudera AI and Cloudera.

If you previously used a different OID, such as `urn:oid:2.5.4.11` (intended for organizational unit name), to map groups in Cloudera Data Science Workbench (CDSW), you must update this mapping in your IdP's configuration to the new OID.

### Migration strategy

The migration tool expects a running CDSW instance and on premises cluster side-by-side.

### on premises platform

- CDSW to Cloudera AI migration is supported only on Cloudera Embedded Container Service clusters.
- The migration of CDSW to Cloudera AI is not supported for the Cloudera Embedded Container Service cluster installed with the internal registry alias option.
- Migration to Cloudera AI on premises in OpenShift Container Platform (OCP) environment is not supported.

### CDSW version

CDSW migration is supported only from CDSW version 1.10.0 and higher versions.

### Engine support

The usage of legacy engine is deprecated from Cloudera Machine Learning on premises 1.5.1. Transform all your workloads to use ML Runtimes before the migration.

### Custom configurations

Custom configurations, such as host or Kubernetes configurations, are not migrated. You must take notes of these configurations and configure your on premises cluster manually after migration.

### CDSW Projects

- CDSW Projects which access HBase might not work after migration.
- CDSW projects that use engines with Spark might not work as expected after migration.

### Folder path of migration

The CDSW to Cloudera AI migration tool host mounts the root folder from the Cloudera Embedded Container Service hosts and expects the docker binary to be present in the `/opt/cloudera/parcels/ECS/docker/docker` path. If any customization is made to this path in your environment, copy or softlink the correct docker to the correct path to unblock the migration.

## Project migration with the Cloudera AI command-line utility tool

The command-line utility tool enables seamless migration of projects across different environments, such as from CDSW to Cloudera AI. The migration includes project files stored on NFS, project settings, models, jobs, and applications.

### Prerequisites for project migration with the command-line utility tool

Before migrating from Cloudera Data Science Workbench (CDSW) to Cloudera AI on premises, you must meet a number of prerequisites to succeed.

The following source and target versions of Cloudera AI and CDSW are tested and certified versions:

**Table 3: Tested and certified target and source versions of Cloudera AI and CDSW**

Migration type	Source type	Source version	Target type	Target version
CDSW to Cloudera AI	CDSW	1.10.2 & higher versions	Cloudera AI on premises	1.5.5 SP1 1.5.5 CHF1 1.5.4 SP2 CHF1
Cloudera AI to Cloudera AI	Cloudera AI on premises	1.5.0 & higher versions	Cloudera AI on premises	1.5.5 SP1 1.5.5 CHF1 1.5.4 SP2 CHF1

**Migration requirements - additional machine required**

Project migration requires a third machine, such as user laptop or Bastion host, with connectivity to both CDSW and Cloudera AI.

Consider the followings:

- The project is first downloaded from CDSW to an intermediate machine using the `export` command. Then, it is uploaded to the Cloudera AI Workbench using the `import` command. The utility uses the `cdswctl` client for login and the creation of an SSH session and tunnel.
- The utility uses `rsync` command line utility tool for migration of project files through the SSH tunnel.
- Project artifacts, such as models, jobs, and applications, are migrated using APIs.
- Authentication is carried out using the API key provided during migration and only authorized users are allowed to migrate projects. The data in transit will remain encrypted as long as the workbench has https connections.

The additional machine must have the following configurations:

- Unix like system (MacOS or Linux)
- Connectivity to both the source and the target systems
- `Rsync` must be installed
- Python version must be 3.10 or higher
- Sufficient disk space is required to hold the project content. For stronger security, the disk, the file system, or both must be encrypted.
- Custom CA certificates if required

**Migration prerequisites with command-line utility tool**

Learn about the prerequisites for using the command-line utility tool before the migration.

The following prerequisites apply to the command-line utility tool before starting the migration:

- Cloudera AI Workbench must be created on the target with the configuration similar to the source workbench.
- All the users, roles or groups must be created on the Target workbench before starting the migration.
- All the teams must be created on the Target workbench before migration. The administrator must already have information on the teams from the Source workbench. Migration of projects created under team context is supported.
- All the custom Runtimes must be configured on the Target workbench before running project migration.
- You must have an Administrator, Owner or Collaborator role in the project to be able to migrate it.



**Note:** The user migrating the project will become the owner of the project in the target Cloudera AI Workbench.

- All the activities for jobs, models, or applications in the source workbench must be stopped or paused before migration to prevent data corruption. The jobs, models and applications are created in a stopped, paused state in the target workbench.
- The following workbench-level settings configured in the **Site Administration** tab will not be migrated with the utility tool.

- The following Target workbench settings must be manually configured with Administrator role:
  - User quota and custom quota settings
  - Runtime settings, such as Resource profile, and Environment variables
  - Security settings, such as Authentication settings - local, LDAP, SAML, and Kerberos configuration
  - General settings, such as SMTP settings



**Note:** Runtime addons are supported in Cloudera AI. For more information, see [Using ML Runtimes Addons](#).

### Additional steps to check both the source and the target workbenches

Learn about additional checks to take for the source and the target workbenches before the migration.

The following additional steps must be taken to check the source and the target workbenches before the migration:

- Make sure that you have the rsync-enabled Runtime image added to your Cloudera AI Workbench Runtime catalog. The image is hosted on [DockerHub](#). If the rsync image is not readily accessible, the image can be created from Dockerfile hosted [here](#) and hosted in any registry. If you do not succeed, ask your workbench Administrator to make these changes.
- Make sure that your default SSH public key is added under your user settings. If it is not, please add them in User Settings Remote Editing SSH public keys for session access . The default SSH public key must be available on your machine at the `~/.ssh/<something>.pub` location. The SSH public key must be uploaded to both the source and target workbenches.
- If no SSH key pair is defined in your system yet, create one. Cloudera recommends not setting a passphrase for the SSH key because SSH connections are established in multiple instances. If a passphrase is set, the automation using the utility might take too long.
- The Legacy API key must be available and must be noted down, as it will be required during migration. To generate a new key, go to User Settings API Keys Legacy API key .

## Legacy engine migration to ML Runtimes

Cloudera recommends using ML Runtimes for all new projects, and recommends the migration of existing engine-based projects to ML Runtimes. Learn about migrating the Cloudera AI workloads from legacy engines to ML Runtimes.

### Before you begin

Create the `<home-directory>/cmlutils/import-config.ini` file to populate the engine to ML Runtimes mapping. For instructions, see the [Import steps](#).

1. Run the cmlutil helpers `populate_engine_runtimes_mapping` command to populate the mapping.

The above command creates the `<home-directory>/cmlutils/legacy_engine_runtime_constants.json` file. Make sure that the tool has the necessary write permissions to create or update the file in the `<home-directory>/cmlutils/` folder.

2. Export the data.

The export command automatically picks up the available `<home-directory>/cmlutils/legacy_engine_runtime_constants.json` file and prepares the metadata accordingly.

3. Import the data.

Run the import command to make sure that the Cloudera AI Workbenches with legacy engines are automatically migrated to ML Runtimes.

The following example is a sample `<home-directory>/cmlutils/legacy_engine_runtime_constants.json` file:

```
{
  "python3": "docker.repository.cloudera.com/cloudera/cds/ml-runtime-workbench-python3.9-standard:2022.11.2-b2",
  "python2": "docker.repository.cloudera.com/cloudera/cds/ml-runtime-workbench-python3.9-standard:2022.11.2-b2",
}
```

```

    "r": "docker.repository.cloudera.com/cloudera/cdsw/ml-runtime-workbench-r4.1-standard:2022.11.2-b2",
    "scala": "docker.repository.cloudera.com/cloudera/cdsw/ml-runtime-workbench-scala2.11-standard:2022.11.2-b2",
    "default": "docker.repository.cloudera.com/cloudera/cdsw/ml-runtime-workbench-python3.9-standard:2023.05.2-b7"
  }

```

4. Modify the content of the mapping if required.

The mapping in the `<home-directory>/cmlutils/legacy_engine_runtime_constants.json` file can be edited to customise the ML Runtimes mapping.

## Migrating a project with the Cloudera AI command-line utility tool

The Cloudera AI command-line utility tool seamlessly migrates projects across different environments.

### Installing the command-line utility migration tool on a bastion-host

Install the utility on a third machine or a bastion host.

1. Clone the repository and run the `python3 -m pip install --editable .` command.
2. Check if the command `cmlutil` command is running.
3. Install the CLI in editable mode from the main branch or from a feature or release branch.

Any change to the source code is reflected in real time without the need for reinstalling the tool again.

- Install the tool from the main branch by using the following command:

```
python3 -m pip install git+https://github.com/cloudera/cmlutils@main
```

- Install the tool from a feature or release branch by using the following command:

```
python3 -m pip install git+https://github.com/cloudera/cmlutils@<branch-name>
```

### Exporting project data for migration

Learn about exporting data from the source target.

### Before you begin

Perform the following validation checks before the export activity:

- Check if the user exists and is authorised to migrate the project.
- Check if the rsync custom Runtime is available in the Source Runtime Catalog.
- Check if the machine to be used has sufficient disk space available to download the project.

### Procedure

1. Create the `export-config.ini` file inside the `<home-dir>/cmlutils` directory.
2. Create a section for each project in the `export-config.ini` file, where you can include project-specific configurations. Place the common configurations shared across projects in the `DEFAULT` section.

Example `export-config.ini` file:

```

[DEFAULT]
url=<Source-Workspace-url>
output_dir=~/.Documents/temp_dir
ca_path=~/.Documents/custom-ca-source.pem
username=user-default
apiv1_key=default-dummy-key
[Project-A]
username=user-1

```

```
apiv1_key=user1-api-key

[Project-B]
username=user-2
apiv1_key=user-2-api-key

[Project-C] # Uses [DEFAULT] configuration as it doesn't have specific c
onfiguration
```

The following configuration details are used in the example file:

- **username:** This is the username of the user who is migrating the project, which is a mandatory element.
  - **url:** This is the source workbench URL, which is a mandatory element.
  - **apiv1\_key:** This is the source API v1, Legacy API key, element, which is mandatory.
  - **output\_dir:** This is a temporary directory on the local machine where the project data or metadata is stored. This is a mandatory element.
  - **ca\_path:** This is the path to a Certifying Authority (CA) bundle to use, in case Python is not able to pick up the CA from the system and the SSL certificate verification fails. This issue might occur with MacOS. this is an optional element.
3. If you want to skip certain files or directories during the export, create the .exportignore file at the root of the source project, that is, /home/cdsw. The .exportignore file follows the same semantics as that of .gitignore file.
  4. Export the Project by running the following command:

```
cmlutil project export -p "Project-A"
or
cmlutil project export -p "Project-C"
```



**Note:** The Project name in this code must match one of the section names in the export-config.ini file.

## Results

- A folder with the Project name is created in the (~/Documents/temp\_dir) output directory. If the project folder already exists, the data will be overwritten.
- All the Project files, artifacts, and logs corresponding to the Project are downloaded to the Project folder.
- The export metrics JSON is created and contains information related to the exported Project.

## Importing project data for migration

Learn about importing Project data for migration.

## Before you begin

Consider the following validation activities before the export activity:

- Check if the user exists and is authorised to migrate the project.
- Check if the rsync custom Runtime is available in the Target Runtime Catalog.
- Check if the local output directory and the project metadata file exist on the third machine used.

## Procedure

1. Create the import-config.ini file inside the <home-dir>/cmlutils directory.
2. Create a section for each project in the import-config.ini file, where you can include project-specific configurations. Place the common configurations shared across projects in the DEFAULT section.

Example export-config.ini file:

```
[DEFAULT]
url=<Destination-Workspace-url>
```

```
output_dir=~/Documents/temp_dir
ca_path=~/Documents/custom-ca-target.pem
username=user-default
apiv1_key=user-default-dummy-key

[Project-A]
username=user-1
apiv1_key=user-1-api-key
[Project-B]
username=user-2
apiv1_key=user-2-api-key

[Project-C] # Uses [DEFAULT] configuration as it doesn't have specific co
nfiguration
```

The following configuration details have been used in the example file:

- **username:** This is the username of the user who is migrating the project, which is a mandatory element.
- **url:** This is the target workbench URL, which is a mandatory element.
- **apiv1\_key:** This is the source API v1, Legacy API key, element, which is mandatory.
- **output\_dir:** This is a temporary directory on the local machine where the project data or metadata is stored. This is a mandatory element.
- **ca\_path:** This is the path to a Certifying Authority (CA) bundle to use, in case Python is not able to pick up the CA from the system and the SSL certificate verification fails. This issue might occur with MacOS. this is an optional element.

### 3. Import the project by running the following command:

```
cmlutil project import -p "Project-A"
or
cmlutil project import -p "Project-B"
```



**Note:** The Project name in this code must match one of the section names in the import-config.ini file.

You can also import the project and initiate validation at the same time by using one of the following commands:

```
cmlutil project import -p "Project-A" -v
or
cmlutil project import -p "Project-B" --verify
```

The above cmlutil command initiates a session in the source and validates the following aspects:

- Consistency of project files between the source and local directories (directories created on the bastion host).
- Consistency of project files between the local directory and the target.
- Consistency in the count of Jobs, Models, and Applications between the source and target.
- Consistency in the metadata of Jobs, Models, and Applications between the source and target.

## Results

- The Project is created in the target workbench.
- All the Project files, artifacts and logs corresponding to the Project are downloaded to the Project folder.
- The import metrics JSON is created and it contains information related to the imported Project.

## Post migration tasks with Cloudera AI command-line utility tool migration

Learn about the post-migration tasks after using the cmlutil migration tool.

Consider the following aspects after using the cmlutil migration tool:

- The SSH keys and API keys will be different in the target Cloudera AI Workbench.

- The Access key and the API key of the models of the migrated projects will be different.

### Manual verification of the migration

Learn about the main tasks to perform manually to validate the migration.

Perform the following actions to validate the migration:

- Validate the number of project files, file sizes, and project settings in the target workbench.
- In the target workbench, the user's public SSH key, which can be found at [User Settings Outbound SSH](#), is different from the source SSH key. Update the SSH key in all external locations, such as the github repository.
- Following the migration, the Model API key and endpoint URL are different from the source. Update all applications that utilize these APIs.
- Jobs in the target workbench have the Spark addon enabled by default. Disable it or change the Spark version if needed.
- Administrators can change the ownership of the project after the migration, if necessary.
- To add collaborators to the project follow the guidelines in [Adding Project Collaborators](#).
- You can export the custom Grafana dashboards from CDSW and import them to Cloudera AI following the steps in [Import Dashboards - Grafana documentation](#).
- Validate the workloads running within the Project. All the Models, Jobs or applications are created in paused or stopped state in the destination workbench, so all the artifacts must be restarted after the migration. Before starting the Models, Jobs, or Applications in the target workbench, the corresponding workloads must be stopped in the source workbench to avoid any data corruption if both workbenches are accessing the same data simultaneously.

### Automated verification of the migration

After the migration, an automated verification starts. Learn about what is being validated, what to consider before running the validation, and what elements do not require validation.

### About this task

The automated verification ensures the validation of the following elements:

- The similarity of project files between the source and local directories.
- The similarity of project files between the local directory and the target.
- The consistency in the number of Jobs, Models, and Applications between the source and target.
- The similarity in the metadata of Jobs, Models, and Applications between the source and target.

No validation exists for the following elements:

- **Job dependencies**– The migration command does not validate job dependencies at the source and target workbenches. Ensure that all necessary job dependencies are manually verified to maintain workflow integrity.
- **Environment variables**– The verification process does not validate environment variables, as these variables might differ between the source and target projects. Manually verify any environment-specific configurations or variables to ensure proper functionality.
- **Hidden files or other files being continuously updated**– If your Project contains hidden files or files that are continuously updated, Cloudera recommends excluding such files from validation by adding them to the .exportignore file at the root of the source Project. This ensures a more accurate comparison and prevents unnecessary discrepancies during the verification process.
- **ML Runtimes or engine**– This tool does not undertake the validation of ML Runtimes or engine-related information between the source and target. Therefore, the user must ensure the accuracy and compatibility of such details.

### Before you begin

Before running the validation command ensure the following requirements:

- You have access to the Cloudera AI command line utility tool.
- Python and the necessary dependencies are installed.
- Project details are added in the export-config.ini and import-config.ini files.

- No changes are made to the source and target before starting validation.

## Procedure

Open a terminal or command prompt and run one of the following commands:



### Note:

If you want to skip certain files or directories during validation, add the details in the `.exportignore` file at the root of the source Project, that is, `/home/cds`. The `.exportignore` file follows the same semantics as that of `.gitignore` file.

```
cmlutil project validate-migration -p [***PROJECT NAME***]
```

or

```
cmlutil project validate-migration --project_name [***PROJECT NAME***]
```



### Note:

Replace the `[***PROJECT NAME***]` with the project that is already imported.

## Batch migration of projects

The Cloudera AI utility tool is primarily designed to facilitate the migration of individual projects. However, a wrapper script is available to enable batch migration of multiple projects.



### Note:

The batch migration scripts are provided for reference, and are not part of the utility. Batch migration was certified with 50 projects with a batch size of 10 due to resource constraint.

Two python scripts are available:

- [Cloudera AI utility tool batch export](#)
- [Cloudera AI utility tool batch import](#)

The batch migration script reads the list of project names from `export-config.ini` or the `import-config.ini` files. Each section defined here corresponds to a specific project, with the section name corresponding to the project name. You can include project-specific configurations within each respective section, while configurations shared across multiple projects can be placed inside the *default* section.

### BATCH\_SIZE

The `BATCH_SIZE` variable provided inside the script controls the number of projects that can be exported or imported simultaneously. To prevent system errors like running out of memory, you must select an appropriate batch size. Each export or import operation of a project generates a distinct session on the workspace, utilizing 1 CPU and 0.5 GB of memory. Therefore, the batch size must be determined considering the available resources on both the source and target workspaces.

Consider the following additional aspects before using the batch migration:

- Before initiating the batch migration, ensure that enough disk space is available on the host machine for downloading all or a batch of projects.
- In case of failure during batch migration, the script can be rerun. However the execution of the batch can be made quicker by deleting all the project names already exported or imported from the configuration file.
- Logs for each project are collected inside the individual project directory.

## Troubleshooting Cloudera AI command-line utility tool

Learn about the troubleshooting issues with the Cloudera AI command-line utility tool.



## Retrying migration

The command line utility tool has been designed to resume the export or import operation from the exact point where it was left off in the event of failures.

**Project files** - The command-line utility tool employs rsync to facilitate the migration of project files. When the export or import command is rerun, the tool synchronizes the project files from the source to the target.

**Project settings or artifacts, that is Models, Jobs or Applications** - During a rerun, the project settings or artifacts, that is Models, Jobs or Applications, that have already been migrated are not to be updated. Instead, only the missing artifacts in the target workbench are migrated. This behavior aligns with the resume operation support in case of failures. The command-line utility tool is not designed to support a synchronization operation between the source and target projects. If you want to update the project that has already been migrated, delete the project in the target workbench and then rerun the migration.

## Running the tool as a background process

There might be cases when the tool is required to run as a separate process detached from the terminal. In that case use the following command on your Linux or Mac machine:

```
nohup <complete CLI command> > <stdout-file> 2> <stderr-file> &
```

The command prints out the Process ID (PID) as well as appends the logs to the stdout and stderr file locations.

The process can be interrupted by sending a SIGINT to the PID returned by using the `kill -INT <PID>` command.

## SSH timeout issues

If an SSH timeout occurs during the project migration or the creation of Models, Sessions or Jobs, increase the `ServerAliveInterval` value in the `~/.ssh/config` command of the bastion host or third computer. The following example is a sample command to increase the SSH timeout in the `~/.ssh/config` command:

```
ServerAliveInterval 60
ServerAliveCountMax 30
```

## 'rsync connection closed' issues on the bastion host

The third computer's, or bastion host's rsync might stop with the connection broken error. In that case, make sure that the rsync version is upgraded to 3.X as rsync 3.X and higher versions have a retry functionality inherently implemented when stopped.

## Remote host identification has changed or Host key verification failed error messages

Remove the host key entry from the `.ssh/known_hosts` file by using the `ssh-keygen -R server-hostname-or-IP` command.

For example: `ssh-keygen -R '[localhost]:5104'`

## SSL certificate\_verify\_failed error message

Create a bundle of all root or intermediate Certification Authorities (CAs) in .pem format. Save this bundle file and provide its path (`ca_path`) inside the `export/import-config.ini` file. This ensures that Python can locate the trusted certificate and prevent the error from occurring.

## configparser.NoSectionError: No section: 'sample\_project' error message

The command-line utility tool cannot find a section for the `sample_project` line in the `export-config.ini` or the `import-config.ini` file. Add the section to the files, with the section name being the same as the Project name.

## Setting up SSL certificates of source and target workbenches

The migration script internally uses `cdswctl` which relies on systemwide configured certificates to establish connection with the source and target workbenches. SSL certificates belonging to both the source and target workbenches must be set up properly.

1. Copy your Certificate Authority (CA) to the dir `/usr/local/share/ca-certificates/` directory. Ensure that the CA file is in `.crt` format.
2. Run the `sudo cp foo.crt /usr/local/share/ca-certificates/foo.crt` command.
3. Update the CA store by using the `sudo update-ca-certificates` command.
1. Navigate to `Finder Applications Utilities Keychain Access`.
2. Select `System` in the left-hand column.
3. Import the certificate files into the `System` keychain by opening `File Import Items`.

Results:

The certificate is marked with a red X, meaning that it is entrusted.

4. Provide trust by double-clicking the certificate.

Under `Trust`, you can also change the setting to `Always Trust` (when using this certificate) value on the top.



### Note:

For the command-line utility tool to run with custom certificates on MacOS, you must add the certificate in the keychain as well as include the certificate, `ca_path`, inside the `export-config.ini` and `import-config.ini` files, because Python is not able to pick up system-wide certificates in MacOS.

## Post migration tasks with CDSW migration tool and command-line utility tool

The post-migration tasks apply to both migration tools and have to be considered after the migration.

Perform the following post-migration actions:

- Validate that your code can run on Cloudera AI, especially if the projects are switching from Legacy Engines to ML Runtime images. Differences exist between the built-in Python packages in the Legacy image compared to the Runtime image.
- All Models, Jobs, and Applications in the migrated projects will be in a suspended or stopped state in the target Cloudera AI Workbench, therefore all the artifacts must be restarted post migration. Before starting the Models, Jobs, or Applications in the destination workbench, the corresponding workloads must be stopped in the source workbench to avoid any data corruption if both workbenches are accessing the same data.
- Verify and update, if required, the ML Runtime image and the Spark Addon for each Model, Job, and Application in the migrated projects.
- Migrated projects do not have Spark Pushdown feature enabled. If required, the Spark Pushdown feature must be enabled at project level. Currently, this cannot be set globally for all projects but changing this for all projects in the Cloudera AI database is possible.
- Cloudera AI Applications' URL will be different. Users of the applications must be informed of the new URLs.