

## OpenText™ Documentum™ Content Management for Engineering

### **Cloud Deployment Guide**

Deploy and configure OpenText Documentum Content Management (CM) for Engineering on certified cloud platforms.

EEGAM250400-ICD-EN-01

---

## **OpenText™ Documentum™ Content Management for Engineering Cloud Deployment Guide**

EEGAM250400-ICD-EN-01

Rev.: 2025-Oct-31

**This documentation has been created for OpenText™ Documentum™ Content Management for Engineering CE 25.4.**  
It is also valid for subsequent software releases unless OpenText has made newer documentation available with the product, on an OpenText website, or by any other means.

### **Open Text Corporation**

275 Frank Tompa Drive, Waterloo, Ontario, Canada, N2L 0A1

Tel: +1-519-888-7111

Toll Free Canada/USA: 1-800-499-6544 International: +800-4996-5440

Fax: +1-519-888-0677

Support: <https://support.opentext.com>

For more information, visit <https://www.opentext.com>

### **© 2025 Open Text**

Patents may cover this product, see <https://www.opentext.com/patents>.

### **Disclaimer**

No Warranties and Limitation of Liability

Every effort has been made to ensure the accuracy of the features and techniques presented in this publication. However, Open Text Corporation and its affiliates accept no responsibility and offer no warranty whether expressed or implied, for the accuracy of this publication.

---

# Table of Contents

<b>1</b>	<b>Prerequisites for deploying and configuring OpenText Documentum CM for Engineering on cloud platforms .....</b>	<b>5</b>
1.1	Prerequisites .....	5
1.1.1	Database prerequisites .....	5
1.1.2	Intelligent Viewer prerequisites .....	9
1.1.3	Vault pre-requisites .....	9
1.1.4	OpenText Directory Services prerequisites .....	9
1.1.5	External Virtual Machine prerequisites .....	10
1.1.6	CloudOps Deployment .....	10
1.1.7	Object Storage .....	11
1.2	Downloading Helm and Helm charts, and container images, and configuring cloud platform-specific requirements .....	11
1.3	Configuring Google Cloud Platform .....	13
1.4	Configuring Amazon Web Services cloud platform .....	15
<b>2</b>	<b>Deploying OpenText Documentum CM for Engineering components .....</b>	<b>19</b>
2.1	Downloading OpenText Documentum CM Helm charts .....	19
2.2	Updating the Chart.yaml file .....	20
2.3	Updating documentum-components.yaml file .....	20
2.4	Updating dockerimages-values.yaml file .....	21
2.5	Updating values.yaml file .....	23
2.6	Updating configuration.yml file .....	34
2.7	Updating addons/ao/ao-config.yaml .....	46
2.8	Updating constant.yaml file .....	47
2.9	Updating passwords.yaml file .....	47
2.10	Updating passwords_vault.yaml file .....	49
2.11	Updating platforms/gcp.yaml .....	49
2.12	Updating platforms/aws.yaml .....	50
2.13	Integrating Brava! Enterprise .....	51
2.14	Integrating Intelligent Viewer .....	52
2.15	Integrating xECM .....	52
2.16	Updating xPlore .....	52
<b>3</b>	<b>Deploy Documentum Helm charts .....</b>	<b>55</b>
3.1	Checking deployment status .....	57
3.2	Accessing application .....	58
<b>4</b>	<b>Post deployment .....</b>	<b>59</b>
4.1	Setting up OpenText Directory Services .....	59
4.1.1	Creating an OAuth client .....	59

4.1.2	Configuring OTDS to sync users with Documentum repository .....	60
4.1.3	Licensing OpenText Documentum CM for Engineering .....	62
4.1.4	Updating OTDS configuration in Documentum Administrator .....	63
4.2	Configuring Reports .....	63
4.2.1	Enable License for Documentum Reports .....	63
4.2.2	Creating users in Documentum Administrator to access reports .....	64
4.2.3	Verifying the reports user in Documentum Reports .....	64
4.2.4	Configuring Reports in client configuration .....	64
4.2.5	Update Documentum Reports Templates folder .....	65
4.3	Configuring OTDS to enable Intelligent Viewer .....	65
4.4	Activating resource ID in Documentum Server .....	66
4.5	Configuring Transformation Services .....	67
4.5.1	Configuring the system for generating PDF rendition .....	67
4.6	Configuring Blazon with Brava Enterprise Viewer .....	69
4.6.1	Mounting Kubernetes volume on Windows virtual machine .....	69
4.7	Configuring Blazon details using Transmittal Comment Config .....	70
4.8	Configuring SMTP server details .....	71
4.9	Updating EmailID for Install owner user .....	71
4.10	Configuring Appworks Gateway .....	72
4.11	Configuring workflow designer .....	72
<b>5</b>	<b>Upgrading OpenText Documentum CM for Engineering .....</b>	<b>75</b>
5.1	Prerequisites .....	75
5.1.1	Database prerequisites .....	75
5.1.2	Helm version .....	75
5.1.3	Documentum Administrator and Workflow Designer .....	75
5.1.4	Backing up the database .....	76
5.1.5	Backing up the D2-Config .....	76
5.2	Upgrade steps .....	77
5.2.1	Download OpenText Documentum CM Helm chart .....	77
5.2.2	To upgrade from HashiCorp Vault-enabled 24.4 or HashiCorp Vault-enabled 25.2 to HashiCorp Vault-enabled 25.4 .....	78
5.3	Post upgrade steps .....	80
<b>6</b>	<b>Troubleshooting .....</b>	<b>81</b>
6.1	Cleaning the OpenText Documentum CM for Engineering deployment .....	81
6.2	Enable OpenText Documentum CM for Engineering and Connector debug logs .....	81
6.3	Resolve issues .....	83

## Chapter 1


# Prerequisites for deploying and configuring OpenText Documentum CM for Engineering on cloud platforms

This documentation provides prerequisites and information to set up and configure Google Cloud Platform and Amazon Web Services cloud platforms.

## 1.1 Prerequisites

### 1.1.1 Database prerequisites

Database requirement	Details
Name of the service using the database	ot-dctm-server
Database type	PostgreSQL on a virtual machine
Database version	PostgreSQL 17.x, AWS RDS PostgreSQL service 17.x, Google Cloud SQL PostgreSQL 17.x  Oracle RDS 21c
Database Admin user roles and Privileges	Create role and create database privileges

Database requirement	Details
Additional requirements like Tablespace creation, Database Tuning, specific parameters in postgresql.conf (for PostgreSQL database), enable dblink extension.	<p>When you use Google Cloud Postgres SQL instance for the DB creation, then follow the steps for allowing connections from Cluster to DB:</p> <ol style="list-style-type: none"> <li>1. In the Google Cloud console, go to the Cloud SQL Instances page.</li> <li>2. To open the Overview page of an instance, click the instance name.</li> <li>3. Select Connections from the SQL navigation menu.</li> <li>4. In the <b>Networking Tab &gt; Authorized networks</b> section, click <b>Add network</b> and type CIDR range of the GKE cluster.</li> <li>5. Click <b>Done</b>.</li> <li>6. Click <b>Save</b>.</li> </ol> <p>When you use AWS Postgres SQL instance for the DB creation, then follow the steps for allowing connections from Cluster to DB:</p> <ol style="list-style-type: none"> <li>1. In the AWS Cloud console, go to the RDS Instances page.</li> <li>2. Navigate to the available databases and copy the endpoint from the database.</li> <li>3. To enable SSL, navigate to <b>Database &gt; Configuration &gt; DB instance parameter group</b>.</li> <li>4. Select <b>SSL parameter group</b>.</li> <li>5. Click <b>Save</b>.</li> </ol> <p> <b>Note:</b> Use the following special characters for database password:</p> <ul style="list-style-type: none"> <li>• Oracle Database: ~ - _ = : , . ? /</li> <li>• PostgreSQL Database: ~ @ - _ = : , . ? /</li> </ul> <p>During a new deployment, you can select any one of the following options:</p> <ul style="list-style-type: none"> <li>• <b>Default Tablespace:</b> Database and repository owners will be created by the Documentum server automatically. There is no need of any database to be created manually by the database administrator and also there is no need to manually create any pre-defined table-spaces.</li> <li>• <b>Predefined Tablespace:</b> <ol style="list-style-type: none"> <li>1. Create a database user in PostgreSQL.</li> <li>2. Create a database inside the pre-defined tablespace with the created database user as the owner. The required format for the database name is: database name = dm_&lt;docbasename&gt;_docbase</li> </ol> </li> </ul>

Database requirement	Details
	<ol style="list-style-type: none"> <li>3. Grant all privileges on database to the new database user.</li> <li>4. Connect to database and grant all on schema public to repository owner. For example, \connect dm_&lt;docbasename&gt;_docbase &lt;docbaseOwner&gt;  Grant all on schema public to &lt;docbaseOwner&gt;; CREATE SCHEMA "&lt;docbaseOwner&gt;"; SET SEARCH_PATH to "&lt;docbaseOwner&gt;"; GRANT ALL ON schema "&lt;docbasename&gt;" TO "&lt;docbasename&gt;"; grant usage on schema "&lt;docbasename&gt;" to "&lt;docbasename&gt;"; grant create on schema "&lt;docbasename&gt;" to "&lt;docbasename&gt;";</li> <li>5. In the documentum/values.yaml file, the db_hostname, databaseusername, dbport, db_service with the newly created database username, database host, and port.  In documentum/config/passwords.yaml file, update the newly created database password in databaseAdminPassword field.</li> <li>6. For Oracle: <ol style="list-style-type: none"> <li>a. Non-default database user creation <ol style="list-style-type: none"> <li>i. Create a database user in Oracle database.</li> <li>ii. Grant the privileges to database user.  For example:  Login to oracle as system admin  SQL&gt; create user login_user identified by password;  User created. SQL&gt; GRANT ALL PRIVILEGES TO login_user;</li> </ol> </li> <li>b. Create a database inside the pre-defined tablespace by login with non-default DB Admin user.  database name = dm_&lt;docbasename&gt;_docbase</li> <li>c. For database creation in Oracle database as a service.  CREATE TABLESPACE DM_&lt;docbasename&gt;_docbase;  GRANT CREATE SESSION TO &lt;docbasename&gt; IDENTIFIED BY "Password_1234567890";  GRANT CREATE SYNONYM TO &lt;docbasename&gt;;  GRANT CREATE VIEW TO &lt;docbasename&gt;;</li> </ol> </li> </ol>

Database requirement	Details
	<pre> GRANT CREATE ANY TABLE, ALTER ANY TABLE, DROP ANY TABLE, LOCK ANY TABLE, SELECT ANY TABLE TO &lt;docbasename&gt;;  GRANT CREATE ANY TRIGGER TO &lt;docbasename&gt;;  GRANT CREATE ANY INDEX TO &lt;docbasename&gt;;  GRANT CREATE SEQUENCE TO &lt;docbasename&gt;;  GRANT CREATE PROCEDURE TO &lt;docbasename&gt;;  GRANT SELECT_CATALOG_ROLE TO &lt;docbasename&gt;;  GRANT EXECUTE_CATALOG_ROLE TO &lt;docbasename&gt;;  ALTER USER &lt;docbasename&gt;DEFAULT TABLESPACE DM_&lt;docbasename&gt;_docbase TEMPORARY TABLESPACE TEMP; CREATE TABLESPACE DM_&lt;docbasename&gt;_index;  ALTER USER &lt;docbasename&gt;quota unlimited on DM_&lt;docbasename&gt;_docbase;  ALTER USER &lt;docbasename&gt;quota unlimited on DM_&lt;docbasename&gt;_index; </pre> <p>7. In the <code>config/configuration.yaml</code> file, update the <code>userName</code> and <code>port</code>.</p> <pre> cs-secrets:  ### Database ### database:   userName: postgres content-server: ### Database ###   database:     port: 5432 </pre> <p>8. In the <code>config/passwords.yaml</code> file, update the following for non vault systems.</p> <pre> databaseAdminPassword: &amp;db_admin_password password </pre> <p>9. In the content server section of <code>documentum/config/configuration.yml</code>, enable the existing under repository (docbase) section and update the <code>id</code> and the <code>docbase</code> name in the <code>index</code> parameter respectively.</p> <pre> content-server:   ### Docbase ###   id: &lt;six digit number&gt; #Eg. 123456   existing: true   index: DM_&lt;docbasename&gt;_DOCBASE #Eg. DM_AODOCBASE_DOCBASE </pre>

## 1.1.2 Intelligent Viewer prerequisites

Intelligent viewer requirements	Details
Name of the service which is using the database	Intelligent Viewer
Database type	PostgreSQL Virtual machine
Database version	17.x
Database admin user roles and privileges	Create role and database privileges
Additional requirements such as Tablespace creation, Database tuning, specific parameters in Postgresql.conf (for PostgreSQL database), enable dblink extension.	Do not create a separate database and any pre-defined table spaces.

## 1.1.3 Vault pre-requisites



**Note:** This step is applicable only if Vault is enabled in your deployment.

To enable Vault, you must configure the Vault server and populate secrets/keys and its values.

For more information about Documentum Vault secrets and key names, see *OpenText Documentum Content Management - Cloud Deployment Guide (EDCSYCD-IGD)*.

## 1.1.4 OpenText Directory Services prerequisites

1. Create a NON-superuser OpenText Directory Services (OTDS) database user.

Connect to the PostgreSQL database as a PostgreSQL administrative user and create the OTDS database and user.

```
-- Setup the user for otds
CREATE USER otdsdb_user WITH LOGIN NOSUPERUSER NOCREATEDB INHERIT NOREPLICATION
CONNECTION LIMIT -1 PASSWORD ' <otdsdb_user_password>';
```

2. Create an OTDS database.

Connect to the PostgreSQL database as PostgreSQL administrative user and create the OTDS database and user.

```
-- Create the database and set permissions for OTDS database user
CREATE DATABASE "otdsdb" WITH OWNER = "otdsdb_user" ENCODING = 'UTF8' CONNECTION
LIMIT = -1;
```

3. Enable pg\_trgm extension on the OTDS database.

Connect to PostgreSQL database as PostgreSQL administrative user and connect to the new OTDS database. For example, otdsdb

```
-- Add the schema for otds to use (can skip this step as OTDS does it by itself.)
CREATE SCHEMA otds AUTHORIZATION otdsdb_user;
```

```
-- Enable the pg_trgm module (needed by otds, pg_trgm does not create tables so it
does not need a schema)
CREATE EXTENSION IF NOT EXISTS pg_trgm SCHEMA pg_catalog;
```

### 1.1.5 External Virtual Machine prerequisites

**Table 1-1: External virtual machine**

Version	Transformation Services	Brava Enterprise	Blazon Enterprise
OS version	Windows Server 2022 64-bit/Windows Server 2025 64-bit Enterprise Virtual Machines	Windows Server 2022 64-bit/Windows Server 2025 64-bit Enterprise Virtual Machines	Windows Server 2022 64-bit/Windows Server 2025 64-bit Enterprise Virtual Machines
Install user account permissions	Admin permissions	Admin permissions	Admin permissions
Additional software's required	Microsoft Office	Not applicable	Not applicable
Installation procedure	For more information about installing Transformation Services, see <i>OpenText Documentum Content Management - Transformation Services Installation Guide (EDCCT-IGD)</i> .	For more information about Brava Enterprise, see <i>Blazon Enterprise 16.6 Administration Guide</i> .	For more information about installing Blazon, see <i>OpenText Blazon Enterprise server Installation Guide</i>



**Note:** While installing Transformation Services (CTS), CTS documents must be installed and configured with OpenText Documentum Content Management Server 25.2.

### 1.1.6 CloudOps Deployment

JDK 21 and openssl for generating certificate.

### 1.1.7 Object Storage

1. Retrieve the following S3 object storage details:

- access\_key\_id
- secret\_access\_key
- endpoint url

2. Add the following GCPstore details:

- Endpoint URL
- JSON format store credentials



**Note:** OpenText recommends S3 as the default store.

## 1.2 Downloading Helm and Helm charts, and container images, and configuring cloud platform-specific requirements

Kubernetes is a portable, extensible open-source platform orchestration engine for automating deployment, scaling, and management of containerized applications. Kubernetes can be considered as a container platform, microservices platform, and portable cloud platform. It provides a container-centric management environment.

OpenShift is an enterprise-ready, subscription-based platform orchestration engine for automating deployment, scaling, and management of containerized applications. OpenShift is a container platform that provides capabilities to manage advanced clusters.

You can use the containerized deployment using the OpenText Documentum CM container images and OpenText Documentum CM Helm charts that are packaged with the release. The product Release Notes document contains detailed information about the list of applications and its supported versions.

To download the container images from OpenText Container Registry, do the following:

1. Log in to OpenText Container Registry using the following command format:

```
docker login registry.opentext.com
```

When prompted, provide your OpenText My Support login credentials.

2. Download the container image(s) using the following command format:

```
docker pull registry.opentext.com/<image_name>:<image_tag>
```

The following container images are available for download:

Product or component name	Image name	Image tag
OpenText Documentum CM Server	ot-dctm-server	25.4.0 or 25.4
OpenText Documentum CM Server	ot-dctm-fluentd	25.4.0 or 25.4
OpenText Documentum CM Server	ot-dctm-dsis	25.4.0 or 25.4
OpenText Documentum CM client	ot-dctm-client-smartview	25.4.0 or 25.4
OpenText Documentum CM client	ot-dctm-client-mobile	25.4.0 or 25.4
OpenText Documentum CM Administrator	ot-dctm-admin	25.4.0 or 25.4
OpenText Documentum CM Administrator	ot-dctm-admin-tomcat	25.4.0 or 25.4
OpenText Documentum CM Workflow Designer	ot-dctm-workflow-designer	25.4.0 or 25.4
xPlore	dctm-xplore-indexserver	22.1.14
xPlore	dctm-xplore-indexagent	22.1.14
xPlore	dctm-xplore-cps	22.1.14
OpenText Documentum CM Reports	ot-dctm-reports-client	25.4.0 or 25.4
OpenText Documentum CM Reports	ot-dctm-reports-base	25.4.0 or 25.4
OpenText Documentum CM Reports	ot-dctm-reports-installer	25.4.0 or 25.4
OpenText Directory Services	otds-server	25.4.0 or 25.4
OpenText AppWorks Gateway	otawg	25.4.0 or 25.4
OpenText AppWorks Gateway	otawg-init	25.4.0 or 25.4
OpenText Intelligent Viewing	otiv-amqp	25.4.0 or 25.4
OpenText Intelligent Viewing	otiv-asset	25.4.0 or 25.4
OpenText Intelligent Viewing	otiv-config	25.4.0 or 25.4
OpenText Intelligent Viewing	otiv-highlight	25.4.0 or 25.4
OpenText Intelligent Viewing	otiv-markup	25.4.0 or 25.4
OpenText Intelligent Viewing	otiv-publication	25.4.0 or 25.4
OpenText Intelligent Viewing	otiv-init-otds	25.4.0 or 25.4
OpenText Intelligent Viewing	otiv-publisher	25.4.0 or 25.4
OpenText Intelligent Viewing	otiv-viewer	25.4.0 or 25.4
Base Tomcat	ot-dctm-tomcat	25.4.0 or 25.4
Documentum Content Management for Engineering	ot-dctm-dcme-installer	25.4.0 or 25.4

3. Upload the container image(s) to your container registry using the following command format:

```
docker push <image>
```

4. Download the Helm charts from OpenText Container Registry and extract them into a temporary location.

- a. Add the OpenText Helm repository to the Helm's list of repositories using the following command format:

```
helm repo add opentext https://registry.opentext.com/helm --username  
<user@example.com> --password <password>
```

where <user@example.com> is your My Support logon ID, and <password> is the password of your My Support logon ID.

- b. Refresh your Helm repository information using the following command:

```
helm repo update
```

- c. Obtain the version of the required Documentum Helm chart using the following command format:

```
helm pull opentext/documentum --version=<##.#.#>
```

where <##.#.#> is the version of the Helm chart.

For example:

```
helm pull opentext/documentum --version=25.4.0
```

- d. After you obtain the Helm chart ZIP file, upload it to the computer that you will use to run the Helm commands.
- e. Extract the ZIP file using gunzip or a similar utility and then extract its contents using the following command format:

```
tar xvf documentum-<##>.<#>.<#>.tgz
```

## 1.3 Configuring Google Cloud Platform

This section describes the basic requirements for a Kubernetes environment, and detailed information on how to deploy OpenText Documentum CM for Engineering after the Kubernetes environment is set up. For information about setting up prerequisites in a Kubernetes environment, see *Kubernetes Documentation*.

### To set up your Kubernetes environment:

1. Make sure that you complete all the relevant tasks described in [“Downloading Helm and Helm charts, and container images, and configuring cloud platform-specific requirements”](#) on page 11.
2. Set up a Google Cloud Platform (GCP) Kubernetes cluster.
3. If deployment is Vault enabled, refer to [“Vault pre-requisites”](#) on page 9.
4. Install kubectl cli on your machine and ensure you can use the kubectl command to access the Kubernetes environment. For more information about kubectl on Windows, see *Install kubectl on Windows* on the Kubernetes website.

5. Use the client Helm version that exists between 3.17.x.
6. Ensure Trident storage is available for Google Cloud Platform. For Amazon Web Services you must ensure that Elastic File System (EFS) storage is available.
7. Use the Win2022 64-bit/Win2022 64-bit Enterprise Virtual Machines with Microsoft Office (2016/2019) for Brava Enterprise and OpenText Documentum CM Transformation Services.
8. For Brava Enterprise Viewer and Blazon Enterprise Web Package, use Win2022 64 bit/Win2022 64-bit Enterprise Virtual Machines.
9. Set up the Brava server with SSL enabled and make sure that the URL and user credentials are available.
10. Make sure that the SMTP server is running and is available. This SMTP server must be accessible from the cluster on which the OpenText Documentum CM for Engineering deployment is planned.
11. OpenText Documentum CM docker images should be accessible from the cluster.
12. Provide the domain name of OpenText Documentum CM for Engineering or the URL details.
13. Ensure that you have the following licenses:
  - OpenText Documentum CM license
  - OpenText Documentum CM for Engineering license
  - Intelligent Viewer license - Applicable only if Intelligent Viewer is enabled in your deployment.
14. Ensure that TLS 1.0, 1.1, and weaker ciphers in TLS 1.2 are disabled at infrastructure level.

In TLS 1.2, enable only the following ciphers:

  - TLS\_ECDHE\_RSA\_WITH\_AES\_128\_GCM\_SHA256
  - TLS\_ECDHE\_RSA\_WITH\_AES\_256\_GCM\_SHA384

## 1.4 Configuring Amazon Web Services cloud platform

### Prerequisites

1. Request for Amazon Web Services access with the OpenText cloud services team.
2. Sign in to `https://<Amazon Web Services hosted path>` with the AWS ID and Secret key ID details.
3. On the **Sign In** page, select **Amazon Web Services**.
4. Click **Sign in**.
1. Make sure that you complete all the relevant tasks as described in *"Downloading Helm and Helm charts, and container images, and configuring cloud platform-specific requirements"* on page 11.

2. Download and run the AWS CLI MSI installer for Windows (64-bit): (Version2) file.

*Amazon Web Services* documentation contains detailed information.

3. Install and configure Amazon Web Services CLI.

For example,

```
c:\<dcme deployment path>\aws>msiexec.exe /
ihttps://<Amazon Web Services installer path>/AWSCLIv2.msi
```

4. Verify the installation using the following command format:

```
C:\<dcme deployment path>\aws> aws --version
```

5. Configure the Amazon Web Services using the following command format:

```
C:\<dcme deployment path>\aws> aws configure
AWS Access Key ID[None]: <Provide Access Key ID>
AWS Secret Access Key [None]: <Provide Secret Access Key>
Default region name[None]: us-east-2
Default output format[None]:
```

6. Install Chocolate using the following command format:

```
@"%SystemRoot%\System32\WindowsPowerShell\v1.0\powershell.exe" -NoProfile -
InputFormat None -ExecutionPolicy Bypass
-Command "[System.Net.ServicePointManager]::SecurityProtocol = 3072; iex ((New-
Object System.Net.WebClient)
.DownloadString('https://community.chocolatey.org/install.ps1'))"
&& SET "PATH=%PATH%;%ALLUSERSPROFILE%\chocolatey\bin"
```

For example:

```
C:\dcme\25.4\deployment\aws>@"%SystemRoot%\System32\WindowsPowerShell
\v1.0\powershell.exe" -NoProfile -InputFormat None -ExecutionPolicy Bypass -
Command "[System.Net.ServicePointManager]::SecurityProtocol = 3072; iex ((New-
Object System.Net.WebClient).DownloadString('https://community.chocolatey.org/
install.ps1'))" && SET "PATH=%PATH%;%ALLUSERSPROFILE%\chocolatey\bin"
```

7. Install `eksctl` and `aws-iam-authenticator` using the following command:

```
C:\choco install -y eksctl aws-iam-authenticator
```

If already installed, you can use the upgrade command.

For example,

```
C:|<dcme deployment path>|aws>choco install -y eksctl aws-iam-authenticator
```

8. Verify the `eksctl` installation using the following command format:

```
eksctl version
```

9. Create an Amazon Elastic Kubernetes Service (EKS) cluster and a namespace within the cluster.

- a. Install the `kubectl` using the following command format:

```
curl.exe -O https://<region-amazonaws>/amazon-eks/1.32.0/2024-12-20/bin/windows/amd64/kubectl.exe
```

Add the `kubectl` binary to the path variable.

- b. Verify the `kubectl` installation using the following command format:

```
kubectl version --client
```



**Note:** You must use a `kubectl` version that is within one minor version difference of your Amazon EKS cluster control plane. For example, a 1.20 `kubectl` client must work with Kubernetes 1.19, 1.20, and 1.21 clusters.

The *OpenText Documentum CM Release Notes* contains the supported versions of the Kubernetes cluster.

- c. Create an EKS Cluster.

For example,

```
eksctl create cluster --name dcme --nodes 4 --region us-east-2
```

- d. Create a Kubernetes cluster namespace on the Cloud shell using the following command format:

```
kubectl create namespace <name of namespace>
```

- e. Verify if the namespace is created using the following command format:

```
kubectl get namespace
```

10. Configure the `kubeconfig` file to access a defined EKS cluster using the following command format:

```
aws eks update-kubeconfig --region <region> --name dcme
```

11. Update the `aws-auth ConfigMap` file in the `kube-system` namespace using the following command format:

```
kubectl edit cm aws-auth -n kube-system
```

12. Verify if the `aws-auth ConfigMap` exists in the `kube-system` namespace using the following command format:

```
kubectl get cm aws-auth -n kube-system -o yaml
```

13. Authenticate Amazon Elastic Container Registry (ECR) from CLI using the following command format:

```
aws ecr get-login-password --region <region> | docker login --username AWS --password-stdin <AWS account ID>.dkr.ecr.<region>.amazonaws.com
```

For example:

```
aws ecr get-login-password --region us-east-2 | docker login --username AWS --password-stdin 961386287445.dkr.ecr.us-east-2.amazonaws.com
```

14. Create a docker registry using the following command format:

```
aws ecr create-repository --repository-name awsdockerhub --region <region> --image-scanning-configuration scanOnPush=true
```

15. Retrieve the details of the docker repository using the following command format:

```
aws ecr describe-repositories --repository-name awsdockerhub --region <region>
```



## Chapter 2

# Deploying OpenText Documentum CM for Engineering components

## 2.1 Downloading OpenText Documentum CM Helm charts

- Download the Helm charts from OpenText Container Registry and extract them into a temporary location.

- a. Add the OpenText Helm repository to the Helm's list of repositories using the following command format:

```
helm repo add opentext https://registry.opentext.com/helm --username  
<user@example.com> --password <password>
```

where <user@example.com> is your My Support logon ID, and <password> is the password of your My Support logon ID.

- b. Refresh your Helm repository information using the following command:

```
helm repo update
```

- c. Obtain the version of the required documentum Helm chart using the following command format:

```
helm pull opentext/documentum --version=<##.##.>
```

where <##.##.> is the version of the Helm chart.

For example:

```
helm pull opentext/documentum --version=25.4.0
```

- d. After you obtain the Helm chart ZIP file, upload it to the computer that you will use to run the Helm commands.
- e. Extract the ZIP file using gunzip or a similar utility and then extract its contents using the following command format:

```
tar xvf documentum-<##>.<#>.<#>.tgz
```

## 2.2 Updating the Chart.yaml file

1. Verify if there are required versions of sub charts in the documentum/charts folder.
2. Delete the extracted sub charts from the documentum/charts directory.
3. Update the Chart.yaml file with the required sub charts version.
4. Run the following command to download all the required sub charts:

```
helm repo add <repo name> <repository url>

# For example: helm repo add otrepo https://registry.opentext.com/helm
cd <path to Documentum helm chart>

# Use the following command to download all the required sub-charts
helm dependency update .
```



**Note:** Delete the Chart.lock file and add your Helm repository in the repositories.yaml file.

## 2.3 Updating documentum-components.yaml file

**To enable and disable products or components, update mandatory variables, and deploy pods:**

1. Make sure that you have completed all the relevant tasks as described in *“Configuring Google Cloud Platform” on page 13*.
2. To enable or disable the products and components in the documentum/documentum-components.yaml file, do the following:
  - a. Make sure that the value of enabled is set to true of those products and components you want to deploy.
  - b. Set the value of enabled to false for all those products and components that you do not want to deploy.
3. Provide the appropriate values for the following mandatory variables to pass them to your templates for the enabled products and components:
  - otds
  - docbroker
  - content-server
  - cs-secrets
  - dctm-workflow-designer
  - dctmclientinstaller
  - peinstaller
  - da

- d2classic
- d2config
- d2smartview
- otiv
- dtrbase
- ao-installer

Enable or disable other components as per the requirement.

4. To enable OpenText Documentum CM Mobile support, update the following value to true:

```
appworks-gateway:
  enabled: true
```

5. To enable xPlore components, update the following value to true:

```
#Documentum xPlore

xPlore:
  enabled: true
```

6. To enable the cloud db, update the following values to true.

```
#db
db:
  #Description: Indicates if the database pod deployment is enabled.
  enabled: true
```

## 2.4 Updating dockerimages-values.yaml file

1. Update the repository, fluentdImage details according to your environment:

```
repository: &docker_repo <docker repository path> # Eg. lit-vaartpxym-
p001.gxsonline.net/ot2-paas
fluentdImage: &fluentd_image <fluentd_image_full_path_with_tag>

# Eg. lit-vaartpxym-p001.gxsonline.net/ot2-paas/ot-dctm-fluentd:25.4
```

2. In content-server section, ensure that the image repo and the build number is as per the updated version.

```
content-server:
  extraInitContainers

- name: dctmclientinstallerinit
  image: artifactory.otxlab.net/bpdockerhub/ot-dctm-client-installer:25.4
  imagePullPolicy: *pull_policy_type
  command: ['/bin/sh', '-c', './customscripts/d2-startup.sh']
  volumeMounts:
  - name: dcs-data-pvc
    mountPath: /opt/dctm_docker/customscriptpvc
    subPath: initcontainercustomscripts/dcs-pg
- name: "peinstaller-init"
  image: "artifactory.otxlab.net/bpdockerhub/ot-dctm-bpm-installer:25.4"
  imagePullPolicy: *pull_policy_type
  command: ['/bin/sh', '-c', 'yes |sudo cp -pRf /pescripts/* /opt/dctm_docker/
customscriptpvc/
```

```

        && sudo rm -f /opt/dctm_docker/customscriptpvc/10peijsdeploy.sh']
volumeMounts:
  - name: dcs-data-pvc
    mountPath: /opt/dctm_docker/customscriptpvc
    subPath: initcontainercustomscripts/dcs-pg
  ..
# --- Comment out or remove init-container entry for disabled components. Refer
below example ---
#- name: cc-dar-installer
# image: artifactory.otxlab.net/bpdockerhub/ot-dctm-content-connect:25.4
# imagePullPolicy: *pull_policy_type
# volumeMounts:
#   - name: dcs-data-pvc
#     mountPath: /opt/dctm_docker/customscriptpvc
#     subPath: initcontainercustomscripts/dcs-pg
# ..
# Uncomment below dcme-installer-ini section. Also make sure indentation is
correct.
- name: dcme-installer-ini
  image: artifactory.otxlab.net/bpdockerhub/ot-dctm-dcme-installer:25.4
  imagePullPolicy: *pull_policy_type
  command: ['/bin/sh', '-c', '/aoscripts/startup.sh CS']
  volumeMounts:
    - name: dcs-data-pvc
      mountPath: /opt/dctm_docker/customscriptpvc
      subPath: initcontainercustomscripts/dcs-pg
  envFrom:
    - configMapRef:
        name: ao-configmap
    - secretRef:
        name: ao-supplierexchange

```

3. Uncomment the dsisinit section if the Vault is enabled in both connection broker and content server section.

```

docbroker:
  ...
  extraInitContainers:
    - name: dsisinit
      image: artifactory.otxlab.net/bpdockerhub/ot-dctm-dsis:25.4
      imagePullPolicy: *pull_policy_type
      command: ['/bin/sh', '-c', 'sudo cp -pf /opt/dsis_binary/dsis.zip /opt/
dctm_docker/dsis_binary_volume']
      volumeMounts:
        - name: dsis-binary-volume
          mountPath: /opt/dctm_docker/dsis_binary_volume
          subPath: dsis-binary

content-server:
  ...
  extraInitContainers:
    #uncomment the lines between "#start - dsisinit" and "#end - dsisinit"section if
    vault is enabled.

    Also make sure indentation is correct.
    #start - dsisinit
    - name: dsisinit
      image: artifactory.otxlab.net/bpdockerhub/ot-dctm-dsis:25.4
      imagePullPolicy: *pull_policy_type
      command: ['/bin/sh', '-c', 'sudo cp -pf /opt/dsis_binary/dsis.zip /opt/
dctm_docker/dsis_binary_volume']
      volumeMounts:
        - name: dsis-binary-volume
          mountPath: /opt/dctm_docker/dsis_binary_volume
          subPath: dsis-binarycontent-server:

```

4. To update the `ot-dctm-dcme-installer` to the release version in `content-server`, `dctm-workflow-designer`, `d2config`, `d2classic`, and `d2smartview`, update the following details:

```
content-server:
  .....
  - name: dcme-installer-init
    component: ao-installer
    image: artifactory.otxlab.net/bpdockerhub/ot-dctm-dcme-installer:25.4

dctm-workflow-designer:
  ....
  - name: dcme-installer-init
    component: ao-installer
    image: artifactory.otxlab.net/bpdockerhub/ot-dctm-dcme-installer:25.4

d2config:
  .....
  - name: dcme-installer-init
    component: ao-installer
    image: artifactory.otxlab.net/bpdockerhub/ot-dctm-dcme-installer:25.4

d2classic:
  .....
  - name: dcme-installer-init
    component: ao-installer
    image: artifactory.otxlab.net/bpdockerhub/ot-dctm-dcme-installer:25.4

d2smartview:
  - name: dcme-installer-init
    component: ao-installer
    image: artifactory.otxlab.net/bpdockerhub/ot-dctm-dcme-installer:25.4
```

## 2.5 Updating values.yaml file

1. Global variables in the `documentum/values.yaml` file.

- `rwoStorage`
- `rwmStorage`
- `docbase`
- `csSecrets`
- `isVaultEnabled`
- `installOwnerUsername`
- `globalRegistryUsername`
- `otdsEnabled`
- `otdsUserName`
- `d2classicWebappName`
- `d2configWebappName`
- `d2restWebappName`
- `d2smartviewWebappName`
- `documentumserviceaccount`

- tomcatbase\_usecommonpvc
- tomcatbase\_commonpvcname
- ingressUrl
- aekLocation
- kmsUrl
- connectMode
- oauthClient
- otdsAuthSvcProtocol
- otdsAuthSvc
- ingressProtocol
- hostShortName
- ingressDomain
- multiIngress
- publicHostShortName
- publicIngressDomain
- vpnHostShortName
- vpnIngressDomain
- webappServiceType
- businessUnit
- datacenter
- environment
- persistLogs
- newrelic
- newrelicProxyHost
- newrelicProxyPort
- newrelicProxyProtocol
- sessionAllowTrustedLogin
- useCertificate
- dbrServiceName
- dbr\_port
- dbrConfigmapName
- dbrdataPVCName
- openshiftEnable

- openshiftTls
- externalAccessEnabled
- isCSORCLdb
- kafka\_admin\_user\_name
- kafka\_topic\_name
- kafkaBrokerList
- fluentd
- fluentdTcpPort
- fluentdRestPort
- fluentdUdpPort
- eventLogLevel
- dfcRPCTracing
- markupPort
- publicationPort
- msgHost
- msgUser
- ccExtension
- dccPrefix
- cslogrotate
- jmsServiceName
- jmsPort
- secrets\_Change
- removeDocumentation
- mtenantid
- msClientId
- db\_hostname
- databaseusername
- dbport
- dbtype
- db\_ssl
- db\_ssl\_mode
- dbSchema\_Name
- db\_service

- `driverClass_Name`
  - `amqConfigmap`
  - `amqSecret`
  - `fsGroup`
  - `licenseKeyName`
  - `logrotate_enable`
  - `logrotate_interval`
  - `dmsServiceName`
  - `dmsHttpPort`
  - `locale`
2. Update the `rwo_storage_class` and `rwm_storage_class` to fit your environment. You can replace any instance of the `*rwo_storage_class` and `*rwm_storage_class` anchor references with a different storage class name to fit your environment.



**Note:** The following step is applicable when you deploy in a GCP:  
The `rwo_storage_class` and `rwm_storage_class` values are applicable only for trident storage.

For example, `rwoStorage:&rwo_storage_class trident-cvs-standard`  
`rwoStorage:&rwm_storage_class trident-cvs-standard`

3. Update the namespace anchor tag:
- ```
namespace: <namespace> #Eg. d2
```
4. Update the `isVaultEnabled` to true if the deployment is using Vault.
- ```
isVaultEnabled: &isVaultEnabled true
```
5. Update the `ingressUrl` anchor tag with the complete ingress host name used for the OpenText Documentum CM client webapps. For example:

```
ingressUrl: &ingressUrl dcm-ingress.d2.cfcr-lab.bp-paas.otxlab.net/
```



#### Notes

- Use the following `ingressUrl` pattern for Google Cloud Platform:  
`ingressUrl: &ingressUrl <cluster_name>.<project_name>.otxlab.net/`  
For example, `ingressUrl: &ingressUrl gke-cluster01.otl-documentum-ccp.otxlab.net/`
- Use the following `ingressUrl` pattern for Amazon Web Services:  
`ingressUrl: &ingressUrl <cluster_name>.<project_name>.otxlab.net/`

For example, for ALB classingressUrl: &ingressUrl dctm-ingress.<clustername>.<region>.elb.amazonaws.com

For NGINX, ingressUrl: &ingressUrl dctm-ingress.<clustername>.<region>.otxlab.net

6. Update the <namespace> placeholder with namespace in env property.

```
env: <namespace>.svc.cluster.local # Eg. d2.svc.cluster.local
```

7. Search for <unified/public ingress full hostname> and replace it with public ingress full host name such as, dctm-ingress.d2.cfcrlab.bp-paas.otxlab.net.



**Note:** Search for <unified/public ingress full hostname> and replace it with the public Google Cloud Platform ingress full host name such as gke-cluster01.otl-documentum-ccp.otxlab.net.

8. Search for all instances of <docbase\_name> and replace it with the required repository name.
9. By default, dmadmin is the user name for the install owner. You can update the dmadmin by using the following parameters:

```
installOwnerUsername: &installOwner_username Administrator
```



**Note:** OpenText Documentum CM for Engineering Cloud Deployment Guide uses dmadmin as the install owner. If you want to use any other install owner name, you must update the required commands.

10. For a new deployment, update the documentumserviceaccount and otivServiceAccountName anchor tags to the common service account if you want to enable the service account for the deployment.

If you are upgrading from a previous release version and if there is no service account created, then you must not update the anchor value tag. Retain the values as follows:

```
documentumserviceaccount: &documentum_service_account default
<custom_service_account>

otivServiceAccountName: &ot_service_account <custom_service_account>
```



**Note:** By default, createserviceaccount is set to False. If you want to enable a service account for all components with one service account name, then set True only for one of the components like connection broker and False for other components since the service account must be created only once and used for all other components.

OTDS and OTIV pods use their own service accounts as they have different permissions configured. If required, you can update the service account name in the respective OTDS and OTIV components.

Other components do not have the option to configure service account and therefore uses the default service account.

If your deployment is Vault enabled, use the same service account that is configured with the Vault server.

11. To customize the OpenText Documentum CM client webapp names, then update the `d2classicWebappName`, `d2configWebappName`, `d2restWebappName`, and `d2smartviewWebappName` anchor tags with the custom webapp name that you want to use for the different webapps.

Use unique custom webapp names. For example:

```
d2classicWebappName: &d2classic_webapp_name OTD2
d2configWebappName: &d2config_webapp_name OTD2Config
d2restWebappName: &d2rest_webapp_name OTD2rest
d2smartviewWebappName: &d2smartview_webapp_name OTD2Smartview
```

12. Update the following values for OTDS properties – related update:

```
otdsUserName: &otds_db_username <DB username>

#Database user name, used to connect to DB. Eg. postgres
oauthClient: d2_oauth_client

# Use the same name to create oauthclient in OTDS. Refer to Setting up OTDS section
for more detailed steps.

otdsAuthSvcProtocol: https

otdsAuthSvc: <INGRESS_URL>/otdsws

# For example, dctm-ingress.ao-qa.cfcr-lab.bp-paas.otxlab.net/otdsws
```

13. Update the `ingressProtocol`, `hostShortName`, and `ingressDomain` when Single Ingress or Internal Ingress is enabled.

```
ingressProtocol: &ingress_protocol https
hostShortName: dctm-ingress
ingressDomain: &ingress_domain d2.cfcr-lab.bp-paas.otxlab.net
```

14. If `multiIngress` is enabled, set `multiIngress` to true and update `publicHostShortName`, `publicIngressDomain`, `vpnHostShortName`, and `vpnIngressDomain`.

```
multiIngress: &multipleIng false

# Short name of the ingress for Public Ingress
publicHostShortName: public-ingress

# Domain of Public Ingress Controller in the cluster
publicIngressDomain: d2.cfcr-lab.bp-paas.otxlab.net
# Short name of the ingress for VPN Ingress
vpnHostShortName: vpn-ingress
# Domain of VPN Ingress Controller in the cluster
vpnIngressDomain: d2.cfcr-lab.bp-paas.otxlab.net
```

15. Specify the service type for the web application. By default the value is ClusterIP.

```
Description: Service type of the web application.

webappServiceType: &webapp_service_type ClusterIP
```

16. If SSL is enabled, set `useCertificate` to `true`.
17. Specify the service name for the connection broker, requiring a fully qualified domain name (FQDN) with a maximum length of 59 characters. Default value is `dbrServiceName: dbr`.
18. Specify the reserved port for the connection broker. Default value is `dbr_port: 1489`.
19. Specify the ConfigMap name for the connection broker. Default value is `dbrConfigmapName:&dbr_configmap_name dbr.configmap`.
20. Specify the PVC name for the connection broker. Default value is `dbrdataPVCName: certdbr-data-pvc`

```
dbrServiceName: dbr
dbr_port: 1489
dbrConfigmapName: dbr.configmap
dbrdataPVCName: certdbr-data-pvc
```

21. If `fluentd` is enabled, set `fluentd` to `true` and update `fluentdTcpPort`, `fluentdRestPort`, and `fluentdUdpPort`.

```
fluentd: false
fluentdTcpPort: 24224
fluentdRestPort: 8888
fluentdUdpPort: 20001
```

22. To enable the `logrotate`, set `cslogrotate` to `true`.
23. Update the following values that are applicable for OpenText Documentum CM for Engineering installation:

```
global:
  # Description: Name of the storage class with Persistent Volume Claim (PVC)
  access mode as ReadWriteOnce.
  rwoStorage: &rwo_storage_class trident-nfs

  # Description: Name of the storage class with PVC access mode as ReadWriteMany.
  rwmStorage: &rwm_storage_class trident-nfs

  # Description: Name of the repository.
  docbase: &docbase_name <docbase_name>

  # Description: Name of the secret configuration file.
  csSecrets: &cs_secret_name cs-secret-config

  # Description: Indicates to use Vault secrets for the deployment instead of plain
  passwords.
  isVaultEnabled: &isVaultEnabled false

  # Description: User name of the installation owner.
  installOwnerUsername: &installOwner_username dmadmin

  # Description: User name of the global registry user.
  globalRegistryUsername: dm_bof_registry

  # Description: Indicates whether OTDS is enabled.
  otdsEnabled: &otds_enabled true

  # Description: Database user name to connect to OTDS.
  otdsUserName: &otds_db_username postgres

  # Description: Default name for Documentum Client web application.
  d2classicWebappName: &d2classic_webapp_name D2
```

```

# Description: (New variable)
d2configWebappName: &d2config_webapp_name D2-Config

# Description: Default name for D2 REST.
d2restWebappName: &d2rest_webapp_name d2-rest

# Description: Default name for the Documentum Smart View web application.
d2smartviewWebappName: &d2smartview_webapp_name D2-Smartview

# Description: Service account information.
documentumserviceaccount: &documentum_service_account default

# Description: Indicates to use the common PVC if the PVC already exists.
tomcatbase_usecommonpvc: true

# Description: Name of the common PVC.
tomcatbase_commonpvcname: d2-extension-pvc
# Description: Provide the Ingress url used for D2, D2-Smartview and D2-rest.
ingressUrl: &ingressUrl <unified/public ingress full hostname>/

# Description: Path where Application Encryption Key (AEK) is available.
# Note: If Vault is enabled, the value must be Remote_Vault. If Vault is not
enabled, use the default value, Local.
aekLocation: Local

# Unsupported feature
kmsUrl: https://kmsurl:port

# Description: Type of the connection mode.
connectMode: dual
# Description: OAuth client information.
oauthClient: d2_oauth_client

# Description: Type of the communication mode.
ingressProtocol: &ingress_protocol https

# Short name of the ingress for Single / Internal Ingress (in case Multi-ingress
is enabled)
hostShortName: dctm-ingress
# Domain of Single / Internal Ingress Controller in the cluster
ingressDomain: &ingress_domain d2.cfcf-lab.bp-paas.otxlab.net

# Description: Indicates if multiples ingresses are enabled. By default, the
value is false. Set the value to true to enable multiple ingress.
# Note: When the value is set to true, it enables d2publicingress, vpningress,
and dctmingress.
multiIngress: &multipleIng false

# Short name of the ingress for Public Ingress
publicHostShortName: public-ingress
# Domain of Public Ingress Controller in the cluster
publicIngressDomain: d2.cfcf-lab.bp-paas.otxlab.net
# Short name of the ingress for VPN Ingress
vpnHostShortName: vpn-ingress
# Domain of VPN Ingress Controller in the cluster
vpnIngressDomain: d2.cfcf-lab.bp-paas.otxlab.net

# Description: Service type of the web application.
webappServiceType: &webapp_service_type ClusterIP

# Description: Indicates if Graylog is enabled for use.
grayLogEnable: &graylog_enable true

# Description: Name of the Graylog or Logstash server.
graylogServer: &graylog_server graylog-proxy-bp.otxlab.net

# Description: Port that listens for the incoming messages from fluentd.
graylogPort: &graylog_port 9000

# Description: Indicates the port on which graylog is listening for TCP

```

```

connections.
  # It is assigned to Documentum Server and other products or components.
  graylogTCPPort: &graylogTCPPort 7075

  # Description: Mandatory static field as per Graylog standards Business Unit.
  businessUnit: &businessUnit ems_dev

  # Description: Data center information such as lit, wok, all, and so on.
  datacenter: &datacenter bp

  # Description: Environment information such as Dev, QA, production, and so on.
  environment: &environment dev

  # Description: Indicates if Graylog is disabled for xPlore.
  persistLogs: &persist_logs false

  # Description: Indicates if New Relic is enabled.
  # Note: If you set the value of this variable to true, the New Relic feature is
  enabled for the connection broker and Documentum Server pods.
  newrelic: &newrelic_enabled false

# Description: IP address of the proxy server.
newrelicProxyHost: &newrelic_proxy_host bp2-prox01-1001.otxlab.net

# Description: Port reserved for the New Relic server.
newrelicProxyPort: &newrelic_proxy_port 3128

# Description: Protocol to connect to the pod.
newrelicProxyProtocol: &newrelic_proxy_protocol http

# Description: Indicates if trusted login is enabled for the session.
# Note: As a security best practice, DFC trusted login for D2 applications in a
production environment should be disabled.
sessionAllowTrustedLogin: false

# Description: Indicates if certificate-based communication is enabled.
useCertificate: false

# Description: Service name of connection broker.
# You must provide the fully qualified domain name (FQDN) and it must not be
greater than 59 characters.
dbrServiceName: dbr

# Description: Port reserved for the connection broker.
dbr_port: 1489

# Description: ConfigMap name of the connection broker.
dbrConfigmapName: &dbr_configmap_name dbr.configmap

# Description: PVC name of the connection broker.
dbrdataPVCName: certdbr-data-pvc

# Description: Indicates if deployment in Red Hat OpenShift is enabled.
openshiftEnable: false

# Description: Indicates if deployment in Red Hat OpenShift is enabled with HTTPS
configuration.
openshiftTls: false

# Description: Indicates if the external access of the connection broker is
enabled.
externalAccessEnabled: true

# Description: Indicates to use the Oracle database.
isCSORCLdb: false

# Description: Administrator user name used to communicate among the replicas in
the Apache Kafka cluster.
kafka_admin_user_name: kafka-user

# Description: Topic name provided while deploying the Apache Kafka cluster for

```

```

storing the events.
kafka_topic_name: Cs-Audit-Topic

# Description: Broker list information of Apache Kafka.
kafkaBrokerList: &brokerList localhost:9092

# Description: Indicates if Fluentd is enabled for use.
fluentd: &fluentd_enabled false

# Description: Port on which Fluentd is listening for TCP connection.
# Note: This port is used for communication between Documentum Foundation Classes
based applications and Fluentd.
fluentdTcpPort: &fluentd_tcp_port 24224

# Description: Port on which Fluentd is listening for REST connection.
fluentdRestPort: &fluentd_rest_port 8888

# Description: Port on which Fluentd is listening for UDP connection.
# Note: Documentum Server uses the UDP connection.
fluentdUdpPort: &fluentd_udp_port 20001

# Description: Log level for the Documentum Foundation Classes events.
# Set any value from 0 to 5 where 0 is for NO LOG, 1 is for ERROR, 2 is for WARN,
3 is for INFO, 4 is for DEBUG, and 5 is for TRACE.
eventLogLevel: 0

# Description: Indicates if Documentum Foundation Classes RPC tracing log is
enabled.
dfcRPCTracing: false

# Description: Indicates the markup service port number.
markupPort: &markup_port 80

# Description: Indicates the publication service port number.
publicationPort: &publication_port 80

## messaging host and user by the publication and publisher services ##

# Description: Indicates the messaging host used by the publication and publisher
services.
msgHost: &msg_host otiv-amqp

# Description: Indicates the messaging user used by the publication and publisher
services.
msgUser: &msg_user user

# Description: Extension name for Content Connect to be used in accessing Admin
console.
ccExtension: &ccExtension cc

# Description: Indicates the prefix for components to be deployed.
dccPrefix: &dccPrefix dcc

# Description: Indicates if the usage of the logrotate tool is enabled.
cslogrotate: false

# Description: JMS service name created by Documentum Server.
jmsServiceName: dcs-pg-jms-service

# Description: (New variable)
jmsPort: 9080

# Description: Indicates if any changes to variables are made in cs-secrets.
# To reflect the changes, you must change the value to a different numeric value
than the one provided for the previous deployment.
# When you change the value and then run the helm upgrade command, the Documentum
Server and connection broker pods are recreated automatically.
secrets_Change: 1

# Description: Indicates whether the Swagger documentation is available for D2
REST or Documentum Smart View.

```

```

# Set this variable to true to disable Swagger documentation for D2 REST or
Documentum Smart View.
removeDocumentation: true

# Description: Microsoft Azure app registration tenant ID for Documentum Content
Management for Microsoft 365.
mtenantid: ""

# Description: Microsoft Azure app registration client ID for Documentum Content
Management for Microsoft 365.
msClientId: ""

# Description: Database IP to be connected.
db_hostname: &db_hostname db-pg

# Description: User name of the database.
databaseusername: postgres

# Description: Database port to be connected.
dbport: 5432

# Description: Database type to be connected. Valid values are:
# * postgres for PostgreSQL
# * oracle for Oracle
dbtype: postgres

# Description: Set the value to true to enable the SSL database for PostgreSQL or
Oracle.
db_ssl: false

# Description: SSL connection mode that is used to deploy the SSL-enabled
PostgreSQL database.
# The valid values are prefer, require, verify-ca, and verify-full.
db_ssl_mode: <ssl_mode>

# Description: Schema name of the JDBC database. For example, public.
# Note: For ORACLE DB value, set the variable value to oracle.
dbSchema_Name: postgres

# Description: Database service name.
# Note: For ORACLE DB value, set the variable value to oracle.
db_service: MyPostgres

# Description: The className of the JDBC driver.
# · For PostgreSQL: org.postgresql.Driver
# · For Oracle: oracle.jdbc.driver.OracleDriver
# Note: For ORACLE DB value, set the variable value to
oracle.jdbc.driver.OracleDriver.
driverClass_Name: org.postgresql.Driver

# Description: ConfigMap name of ActiveMQ.
amqConfigmap: amq-configmap

# Description: Name of the ActiveMQ secret configuration file.
amqSecret: amq-secret

# Description: (New variable)
fsGroup: 1000

# Description: (New variable)
licenseKeyName: newrelicLicensekey

# Description: (New variable)
logrotate_enable: true

# Description: (New variable)
logrotate_interval: 1d

# Description: (New variable)
LSRestNewrelicName: LSS-LSREST-<hyperscalar>-<namespace>

```

```
# Description: (New variable)
LSPrintNewrelicName: LSS-LSPRINT-<hyperscalar>-<namespace>

# Description: (New variable)
LSSFTPNewrelicName: LSS-LSSFTP-<hyperscalar>-<namespace>

# Description: Name of the Documentum Messaging Service pod.
dmsServiceName: &dmsServiceName dctm-dms

# Description: HTTP port of the Documentum Messaging Service application.
dmsHttpPort: 8489

# Description: Documentum's locale code
locale: en
```

24. Update the following values to update external PostgreSQL database configuration database host:

```
db_hostname: &db_hostname <Postgres DB IP> # Eg. 10.194.41.53

dbport: <Postgres DB Port> # Eg. 5432
```

25. Run the following command to enable the external connection broker and Documentum CM Server access:

```
externalAccessEnabled: true
```

## 2.6 Updating configuration.yml file

1. Search for all instances of <unified/public ingress full hostname> and replace with public ingress full hostname: `dctm-ingress.d2.cfcrlab.bp-paas.otxlab.net`.



### Notes

- Search for <unified/public ingress full hostname> and replace it with the public GCP ingress full host name such as `gke-cluster01.otl-documentum-ccp.otxlab.net`.
- Search for <unified/public ingress full hostname> and replace it with the public AWS ingress full host name.

2. Configuring OpenText Intelligent Viewing configuration for Google Cloud Platform:

The defined format for OTIV ingress URLs in a Google Cloud Platform or Anthos or AWS is `http://<service-name>-<namespace>.<cluster-name>.<project-name>.<domain-name>`.

For example, `https://otiv-viewer-ao-dev-gcp.gke-cluster01.otl-documentum-ccp.otxlab.net`

This format is applicable for the following ingress URLs:

- OTIV\_HIGHLIGHT\_SERVICE\_URL
- OTIV\_MARKUP\_SERVICE\_URL

- OTIV\_PUBLICATION\_SERVICE\_URL
- OTIV\_VIEWER\_SERVICE\_URL

### 3. Update the following if Vault is enabled:

- OTDS

```
....
....
....
## OTDS will read secrets from Vault rather than using k8s secrets
vault:
  ## the URL to the vault server. Use http://localhost:8200 if using the
  Vault Agent Injector.
  url: <https://vaulturl> Ex: https://vault.otxlab.net
  ## if using Vault namespaces, provide the namespace
  namespace: <name-space> Ex: private-cloud
  ## auth path for k8s auth
  authpath: <auth-path>

  ## audience for the projected service account token (optional)
  tokenAudience:
  ## if the connection to Vault needs to go through a proxy, specify it
  here (e.g. http://proxyhost:port)
  ## (only applicable if using the agentInjector)
  proxyAddress:
  ## the name of the Vault role for OTDS to use
  role: <vault-role> Ex: otl-dcme
  ## the path in Vault to the following secrets:
  ## - CryptKey (with key 'CryptKey'). See description for 'cryptKey'.
  ## - AdminPassword (with key 'AdminPassword'). See description for
  'adminPassword'.
  ## - JDBCCreds (with keys 'username' and 'password'). See description
  for 'otdsdb.username' and 'otdsdb.password'.
  # - OpenDJPassword (with key 'OpenDJPassword') **ONLY REQUIRED IF
  MIGRATING FROM OPENDJ. See description for 'migration'. **
  ## The path must *NOT* include the /v1 prefix, namespace, or the /data/
  qualifier. These are automatically added by OTDS.
  secretsPath: <key-path> Ex: kv/lab/dcme

  ## Vault Agent Injector - set to true to use the Vault Agent Sidecar
  Injector rather than having OTDS connect directly to Vault
  agentInjector: false
```

- cs-logging-configMap

```
cs-logging-configMap:
...
...
vault:
  configmap: |-
    spring.cloud.vault.scheme=https
    spring.cloud.vault.host=vault.otxlab.net
    spring.cloud.vault.port=443
    spring.cloud.vault.namespace=<namespace> Ex: private-cloud

    spring.cloud.vault.authentication=KUBERNETES
    spring.cloud.vault.kubernetes.role=<role> Ex: otl-dcme

    spring.cloud.vault.kubernetes.kubernetes-path=<auth-path> Ex: jwt-dcme

    spring.cloud.vault.kubernetes.service-account-token-file=<secrets-
    token-path>

    Ex: /var/run/secrets/kubernetes.io/serviceaccount/token

    #Must end with forward slash
```

```

    dsis.dctm.kvpath= <key-path> /kv/data/lab/dcme/
    #host must be either localhost or 127.0.0.1 always
    dsis.dctm.host=localhost
    #post can be changed based on the availability
    dsis.dctm.port=8200
    dsis.dctm.executorThreadCount=10
    dsis.dctm.token=
    dsis.dctm.tokenNeeded=false
    dsis.dctm.retryFailure=5
    dsis.dctm.retrySleepInterval=5
    dsis.dctm.enforceListDuringInit=true

```

- appworks-gateway

```

appworks-gateway:
...
...
  vault:
    # As of now direct integration with Opentext Vault is supported
    type: Opentext
    # Provide the vault host

    host: <host-name> Ex: vault.otxlab.net

    Ex: vault.otxlab.net

  # Provide the vault port

  port: <port> Ex:443

  # if using Vault namespaces, provide the namespace

  namespace: <name-space> Ex: private-cloud

  #secret configured in vault

  secrets: <key-path> Ex: kv/lab/dcme/

  # vault authentication config path

  authConfigPath: <auth-config-path> Ex: /var/run/secrets/kubernetes.io/
  serviceaccount/token

  # vault authentication path

  authPath: <auth-path> Ex: jwt-dcme

  # vault role name

  role: <role> Ex: otl-dcme
  # auth type

  authType: jwt

```

- otiv

```

otiv:
...
...
  secretlink:
    loglevel: INFO
  vault:
    ## provide vault server address e.g https://vaultserver:port/
    address: <vault-host> Ex: https://vault.otxlab.net/

    ## provide vault secret mount path e.g kv

    mountpoint: kv

    ## vault secret path e.g dev/foo

```

```

path: <secret-key-path> Ex: lab/dcme/

## vault namespace e.g default

namespace: <name-space> Ex: private-cloud

## vault authentication path to verify authentication. e.g auth/k8s

authpath: <auth-path> Ex: auth/jwt-dcme

## vault role name

role: <role-name> Ex: otl-dcme

```

4. Search all instances of <namespace> and update with the required namespace.
5. Search all instances of <docbase\_name> and update as required.
6. Perform the following for OTDS properties update.

- a. Specify the OTDS database details:

```

otds:
  otdsws:
    otdsdb:
      url: <Database URL> # For example, jdbc:postgresql://10.9.77.234:5432/
otdsdb

```



**Note:** You must ensure that the database name configured is available in the PostgreSQL database.

- b. Update the *public hostname* property.

```

otds:
  otdsws:

    publicHostname: <INGRESS_URL>

    # For example d2-ingress.ao-qa.cfcr-lab.bp-paas.otxlab.net

```

- c. **Optional** Automatic OTDS configuration:

Follow these steps only if you want to configure the OTDS automatically:



**Note:** Update the config.yml file as per the required values.

1. Navigate to the ao/values.yml file and set *enableBootstrapConfig* to true in the **otdsws** section.

For example:

```

otds:
  otdsws:
    enableBootstrapConfig: true

```

2. Navigate to the ao/charts/otds/charts/otdsws/config.yml file, and update the config.yml with the required values.

- a. Add the system attribute details.

```

systemAttributes:
- name: otds.as.SameSiteCookieVal
  value: None

```

- b. You can update the partition name and description to create non-sync partition in OTDS.

For example:

```
partitions:
- name: <partition_name> # For example, ao_partition
  description: <partition_description> # For example, A0 Partition
```

- c. Update users and respective partition IDs. For example:

```
users:
- name: <user_name> #Eg. author
  userPartitionID: <partition_name> #Eg. ao_partition
  values:
  - name: userPassword
    values:
    - <user_password> #Eg. Password@123
  - name: givenName
    values:
    - <user_given_name> # Eg. Author
  - name: mail
    values:
    - <user_mail> # Eg. author@demo.com
```

- d. Update the resources section with the required details.

```
resources:
- resourceName: <resource_name> # Eg. A0Resource
  connectorName: REST (Generic)
  connectorid: rest
  userSynchronizationState: true
  pcCreatePermissionAllowed: true
  pcModifyPermissionAllowed: true
  userAttributeMapping:
  - sourceAttr:
    - cn
    destAttr: default_folder
    mappingFormat: "/Users/A0Users"
  - sourceAttr:
    -
    destAttr: client_capability
    mappingFormat: "2"
  - sourceAttr:
    - oTEternalID3
    destAttr: user_login_name
    mappingFormat: "%s"
  - sourceAttr:
    -
    destAttr: user_type
    mappingFormat: "dm_user"
  - sourceAttr:
    - cn
    destAttr: user_name
    mappingFormat: "%s"
  - sourceAttr:
    - mail
    destAttr: user_address
    mappingFormat: "%s"
  - sourceAttr:
    -
    destAttr: create_default_cabinet
    mappingFormat: "F"
  - sourceAttr:
    -
    destAttr: user_rename_enabled
    mappingFormat: "F"
  - sourceAttr:
    - cn
```

```

      destAttr: __NAME__
      mappingFormat: "%s"
    - sourceAttr:
      -
      destAttr: user_privileges
      mappingFormat: "O"
    - sourceAttr:
      -
      destAttr: user_rename_unlock_locked_obj
      mappingFormat: "T"
    - sourceAttr:
      -
      destAttr: user_xprivileges
      mappingFormat: "32"
    - sourceAttr:
      - oTObjectGUID
      destAttr: user_global_unique_id
      mappingFormat: "%s"

groupAttributeMapping:
  - sourceAttr:
    - mail
    destAttr: group_address
    mappingFormat: "%s"
  - sourceAttr:
    - oTObjectGUID
    destAttr: group_global_unique_id
    mappingFormat: "%s"
  - sourceAttr:
    -
    destAttr: group_rename_enabled
    mappingFormat: "F"
  - sourceAttr:
    - cn
    destAttr: group_name
    mappingFormat: "%s"
  - sourceAttr:
    - cn
    destAttr: __NAME__
    mappingFormat: "%s"

connectionParamInfo:
  - name: fBaseURL
    value: http://<serviceName in ao/values.yml under content
server>

    -jms-service:<ports.jmsport in content-server values.yml>/
dmtodsrest
  # For example, http://dcs-pg-jms-service:9080/dmtodsrest
  - name: fUsername
    value: <docbase_name>\<install_owner_name> # For example,
docbase1\dmdadmin
  - name: fPassword
    value: <install_owner_password> # For example, password

```

- e. Update access roles section with partition and resource details. For example:

```

accessRoles:
  - name: <accessrole_name> # For example, Access to AOResource
    description: <accessrole_description> # For example, Access to
AOResource
    accessRoleMembers:
      userPartitions:
        - userPartition: <partition_name> # For example, ao_partition
resources:
  - resourceName: <resource_name> # For example,AOResource
attributeList:
  - name: pushAllGroups

```

```
values:
  - "True"
```

- f. Update OAuth clients section with the redirect URLs. For example:

```
oauthClients:
  - name: <oauth_client_id> # For example, d2_oauth_client
    description: <oauth_client_description> # For example, A0
  Auth Client
  redirectURLs:
    - "<INGRESS_URL>/D2/d2_otds.html" # For example, "https://
dctm-ingress.ao-qa.cfcr-lab.bp-paas.otxlab.net/D2/d2_otds.html"
    - "<INGRESS_URL>/D2/OTDSLogoutResponse.html"
      # For example, "https://dctm-ingress.ao-qa.cfcr-lab.bp-
paas.otxlab.net/D2/OTDSLogoutResponse.html"
    - "<INGRESS_URL>/D2-Smartview/ui" # For example, "https://
dctm-ingress.ao-qa.cfcr-lab.bp-paas.otxlab.net/D2-Smartview/ui"
```

- g. Ensure the http.negotiate auth handler is disabled. For example:

```
authHandlers:
  - _name: http.negotiate
    _id: http.negotiate
    _description: Handles "Negotiate" authentication with the browser.
    _class:
com.opentext.otds.as.drivers.http.NegotiateHttpAuthenticationHandler
    _enabled: false
    _priority: 20
    _scope:
    _properties:
      - _key: com.opentext.otds.as.drivers.http.negotiate.enablemobile
        _value: 'false'
    _authPrincipalAttrNames:
      - oTExternalID4
```

- h. Add trusted sites as required.

```
trustedSites:
  - https://dctm-ingress.ao-qa.cfcr-lab.bp-paas.otxlab.net
  - https://*.sharepoint.com
  - https://*.officeapps.live.com/
  - https://outlook.office.com
  - https://outlook.office365.com
  - https://inc-word-edit.officeapps
```

7. Update the following OTDS details for **Workflow Designer** configuration properties:

```
dctm-workflow-designer:
  otds:
    url: <OTDS URL> # For example: https://dctm-ingress.ao-qa.cfcr-lab.bp-
paas.otxlab.net
    reverseproxy_url: <Ingress Url> # For example: https://dctm-ingress.ao-
qa.cfcr-lab.bp-paas.otxlab.net
```

8. Update the following OTDS details for **Documentum Administrator** configuration properties:

```
da:
  otds:
    url:<OTDS url>/otdsws
    For example: https://dctm-ingress.ao-qa.cfcr-lab.bp-paas.otxlab.net/otdsws
```

9. Perform the following for **External Postgres DB** configuration:

```
dctm-server:
  cs-secrets:
    database:
```

```

      userName: <DB username> # For example:postgres

    content-server:
      database:
        port: 5432

```

10. Update the <docabse\_name> for the DM\_<docbase\_name\_DOCBASE> value in content-server section.

```

dctm-server:
  content-server:
    docbase:
      index: DM_<Docbase Name>_DOCBASE # For example, DM_DOCBASE1_DOCBASE

```

11. Ensure that the **extraEnv** section under content server has the following values set to 7. In addition, update the <docbase\_name> placeholder with the repository name under the **extraEnv** section.

```

dctm-server:
  content-server:
    extraEnv:
      - name: LSS_CC_ENABLED
        value: "T"
      - name: DA_PRIVILEGE_ENABLED
        value: "T"
      - name: <docbase_name>_resource_id # For example, docbase1_resource_id
        value: ""
      - name: <docbase_name>_secretKey   # For example, docbase1_secretKey
        value: ""
      - name: <docbase_name>_MIGRATE_LDAP_CONFIGS # For example,
        DOCBASE1_MIGRATE_LDAP_CONFIGS
        value: ""
      - name: MIGRATE_LDAP_DOCBASES
        value: "<docbase_name>" # For example, docbase1

```

12. In OTDS section, update the following OTDS configuration values:

```

otds:
  # Description: Indicates if you want the OTDS URL in the HTTPS mode. For OTDS
  # URL to work in the HTTPS mode, set the value to true.
  # To verify if the URL works, validate otds_rest_credential_url
  # in otdsauth.properties located at $DM_JMS_HOME/ webapps/ OTDSAuthentication/
  # WEBINF/ classes.
  ssl: false

  # Description: Expertise level of the user.
  clientCapability: 0

  # Description: Privileges assigned to the user.
  userPrivileges: 0

  # Description: Extended privileges assigned to the user.
  userXPrivileges: 0

  # Description: User name of dm_user is updated during OTDS synchronization.
  userRenameEnabled: F

  # Description: Indicates to unlock all the objects checked out by the user to
  # be renamed.
  # This is applicable only if userRenameEnabled is set to T.
  userRenameUnlockLockedObject: T

  # Description: Indicates if the group name of dm_group can be updated during
  # OTDS synchronization.
  groupRenameEnabled: F

  # Description: Indicates to update the OTDS certificate details on a restart.
  updateOTDScertonrestart: true

```

```

# Description: Indicates to use oAuth token for password authentication.
passauth_use_oauth2_token: false

# Description: Client ID information.
client_id:

# Description: Client secret information.
client_secret:

# Description: Synchronized user login name.
synced_user_login_name:

# Description: Indicates to automatically refresh certificate information.
auto_cert_refresh: false

# Description: JSON Web Key Sets (JWKS) URL information for certificates.
cert_jwks_url: http://otdsws:80/otdsws/oauth2/jwks

# Description: User partition in user name. The default value is false which
means user name has partition.
# Notes:
# 1. If you set the value to true, you must set the value of synced_user_
login_name to sAMAccountName.
# 2. If you want to update the value of synced_user_login_name
# (sAMAccountName) in otdsauth.properties, you must set the value of
passauth_use_oauth2_token to true.

is_hybrid: false

```

13. Update the following `ingressPrefix` (prefix for the ingress name) and also enable or disable components under `dctm-ingress` as per the customer requirement:

```

dctm-ingress:
  #prefix for the ingress name
  ingressPrefix: dctm
  ingress:
    # Description: To accomodate cluster 1.22. Retain the default value.
    class: nginx
    # Description: Define the service annotations if you want to deploy on Google
    Cloud Platform. You can retain the default values or modify as per your requirement.
    annotations: {}
    # Description: Indicates if the ingress host is configured. It is not required
    to configure host and clusetrDomainName if the configureHost variable is set as
    false.
    configureHost: true

```

14. Enable the external connection broker and Documentum CM Server by performing the following steps:

- a. **Externalizing connection broker**

To access Documentum CM Server outside the Kubernetes cluster (for example: This step is required to configure Transformation Services), enable the **ExtDocbroker** in the `docbroker` section.

- i. Run the following command to get the list of already used ports.

You must select two consecutive ports (ranging from 30000-32767) that are not listed with the following command.

```
kubectl get services --all-namespaces -o json | grep nodePort | sort
```

- ii. Verify the availability of free ports by running the following command:

```
# Command will return empty if the port is available
kubectl get services --all-namespaces -o json | grep nodePort | sort |
grep <port>
```

iii. Configure the consecutive available ports.

For example, *nativeExtPort=30133* (lower value) and *sslExtPort=30134* (higher value).

```
docbroker:
  ExtDocbroker:
    extNativeNodePort: <available port from above command> # Eg. 30133
    extSSLNodePort: <available port from above command> # Eg. 30134
```

b. **Externalizing Documentum CM Server**

For accessing Documentum CM Server outside of the Kubernetes cluster (for example, this step is required to configure Transformation Services), enable ExtCS in the Documentum CM Server section.

tcp\_route under ExtCS: Must be assigned with one of the node's external IP obtained using the following command:

```
kubectl get node -o wide
```

Based on the output, you can select two consecutive unused port numbers.

Run the following command to get the list of already used port numbers:

```
kubectl get services --all-namespaces -o json | grep nodePort | sort
```

Verify the availability of free ports by running the following command:

```
# Command will return empty if the port is available
kubectl get services --all-namespaces -o json | grep nodePort | sort | grep
<port>
```

Use the following command to configure the consecutive available ports.

For example, *nativeExtPort=30135* (lower value) and *sslExtPort=30136* (higher value)

```
content-server:
  ExtCS:
    tcp_route: <node's external IP> For example, 10.9.203.241
    nativeExtPort: <available port from above command> # For example, 30135
    sslExtPort: <available port from above command> # For example, 30136
    extDbrPort: 1491
```

- For GCP deployment, in the ExtCS section, set nativeExtPort as 80 and sslExtPort as 81. Keep the other default values as is.

```
ExtCS:
  tcp_route: 10.0.0.0
  nativeExtPort: 80
  sslExtPort: 81
  extDbrPort: 1491
```

If the Google Cloud Platform cluster is internal or if you want to use only the internal Loadbalancer for ExtCS/ExtDocbroker, the Loadbalancer should be created internally. Set the following values in both the connection broker and content-server sections to create the internal Load Balancer service:

```

useLBAnnotations: true
LBAnnotations:
networking.gke.io/load-balancer-type: "Internal"

```

- c. After the Documentum CM Server pod is deployed, you can access the Documentum CM Server externally using the following command:

```

# Node external IP which was given in tcp_route Eg: 10.9.203.241
dfc.docbroker.host[0]=10.9.203.241
# Docbroker external native port Eg:30133
dfc.docbroker.port[0]=30133

```

15. **Appworks-Gateway/d2mobile:** Update the following values to enable d2mobile:

- a. Enable appworks-gateway, and update PostgreSQL server, port, and PostgreSQL admin user. Use the following details to connect with PostgreSQL database.

```

appworks-gateway:
  database:
    vendor: PostgreSQL
    server:
      port: <Postgres server port> # For example: 5432
    admin:
      user: <Postgres user> # For example: postgres

```

- b. Update appworksdb details. If the database does not exist, pg init container uses the following command to create a database.

```

appworks-gateway:
  database:
    appworksdb:
      user: <appworksdb user> # For example: gateway201
      database: <appworksdb database> # For example: postgresdatabase

```

- c. Update OTDS admin user details and OTDS partition and resource. These details are used to connect to OTDS and create partition and resource for appworks-gateway.

```

appworks-gateway:
  otds:
    admin:
      user: <OTDS admin user> # For example: admin
    partition:
      new: <appworks gateway OTDS partition> # For example: otag17
    resource:
      new: <appworks gateway OTDS resource> # For example: OTAG17

```

- d. Update appworks gateway admin user details. This admin user is created during deployment.

```

appworks-gateway:
  awg:
    admin:
      newadminuser: <appworks gateway admin user> # For example: otag17

```

- e. Update the following appworks gateway external URL and ingress host details:

```

appworks-gateway:
  awg:
    externalurl: "<appworks gateway external url>" # For example "https://
appworks-gateway.ao-dev.cfcrlab.bp-paas.otxlab.net"

  ingress:

```

```
hosts:

  - host: <ingress host>

    # For example, appworks-gateway.ao-qa.cfcf-lab.bp-paas.otxlab.net
```

16. Remove the following DCTM-Reports-Application-<x.x>-Export-Config.zip file.

```
d2config:
  customConfigurations:
    Remove the DCTM-Reports-Application-25.4.0-Export-Config.zip.
    filename:
      -<docbase_name>
```

17. Update Proxy settings and JAVA\_OPTS in content-server extraEnv section.

```
- name: http_proxy
  value: ""
- name: https_proxy
  value: ""
- name: HTTP_PROXY
  value: ""
- name: HTTPS_PROXY
  value: ""
- name: NO_PROXY
  value: "ds-search-agent"
- name: no_proxy
  value: "ds-search-agent"
- name: JAVA_OPTS
  value: " "
```

18. Update CATALINA\_OPTS under extraEnv section for d2classic.

```
- name: JAVA_OPTS
  value: ""

.....
- name: CATALINA_OPTS
  value: "<D2_CATALINA_OPTS>"
```

For example: - .preferIPv4Stack=true -Dhttp.proxyHost=<hostname/IP address> -Dhttp.proxyPort=3128 -Dhttp.nonProxyHosts='localhost|127.0.0.1|10.94.6.\*|100.64.\*.\*|ccls.cust.cloud.opentext.com|\*.svc.cluster.local|\*.default.svc|ds-search-agent' -Dhttps.proxyHost=<hostname/IP address> -Dhttps.proxyPort=3128 -Dhttps.nonProxyHosts='localhost|127.0.0.1| 10.94.6.\*|100.64.\*.\*|ccls.cust.cloud.opentext.com|\*.svc.cluster.local|\*.default.svc|ds-search-agent'

## 2.7 Updating addons/ao/ao-config.yaml

1. Comment the following if you are not using Brava or Intelligent Viewer with *extraVolumeMounts* in the content-server section.

Mount is used as a temporary shared directory for generating the *PDF+comment sheet* for the Share Feedback feature with Brava Enterprise Viewer.

```
content-server:
  extraVolumeMounts:
    - mountPath: /opt/dctm/aoblazon
      name: dcs-vct
      subPath: aoblazon
```

2. Update the Brava! Server URL details.

```
ao-installer:
  configuration:
    bravaServerURL:<Brava! Enterprise Server URL within double quotes> For
example, https://brava.otxlab.net:8443
```

3. Update the following connection details to enable OpenText Core Collaboration for Engineering Connector.

```
ao-installer:
  aoc:
    enabled: false
    xchangeVersion: <Core Collaboration for Engineering (Supplier Exchange)
version, v2 or v3 within double quotes.> # For example, "v3"
    xchangeUrl: <Core Collaboration for Engineering URL within double quotes> #
For example, "https://xchange.net"

    xchangeKeyId: <Core Collaboration for Engineering Client-ID within double
quotes> # For example, "0f410055-5fa9-4d13-aa72-943951338095"

    xchangePrivateKey: <Core Collaboration for Engineering Client-Secret> # For
example, 2aae4b250ae64cce8f190f7517165877

    xchangeSubscriptionName: <Core Collaboration for Engineering subscription
name within double quotes> # For example, "xchange-sub-name"

    xchangeProxyHost: <Proxy Server host within double quotes> # For example,
"proxy.otxlab.net"

    xchangeProxyPort:<Proxy Server port within double quotes> # For example,
4545
```

4. Update the following value to true to support xECM:

```
ao-installer:
  xecm:
    enabled: false
```

5. Update the following value to true to support Brava:

```
ao-installer:
  brava:
    enabled: false
```

6. Update the repository name in d2config section.

```
d2config:
  customConfigurations:
    filename:
```

```
- <docbase_name>:DCME-25.2_Export_Config.zip # For example,
docbase1:DCME-25.2_Export_Config.zip
```

### 7. Update the application name.

```
d2classic:
  shiro:
    OTDS:
      appName: Opentext Documentum Content Management for Engineering 25.2
d2smartview:
  restApiRuntime:
    OTDS:
      appName: Opentext Documentum Content Management for Engineering 25.2
```

## 2.8 Updating constant.yaml file

Update the dctm-ingress to point to correct port for all SSL enabled applications.

```
# Description: For the enabled components that are deployed in the secured mode, change
the value to secured ports.
# For example: If Foundation REST API is enabled and deployed in the secured mode,
change the value of
# dctm-ingress.restService.servicePort to 8443.
```

```
dctm-ingress:
  daService:
    servicePort: 443
  dtrbaseService:
    servicePortCore: 5002
    servicePortServlet: 8443
  otdstenant:
    servicePort: 443
```

## 2.9 Updating passwords.yaml file

### 1. Update the installOwnerPassword, otdsAdminPassword, databaseAdminPassword, globalRegistryPassword, and trustStorePassword anchor tags as required.

```
installOwnerPassword: &installOwner_password Password@123
otdsAdminPassword: &otds_admin_password Opentext1!
databaseAdminPassword: &db_admin_password password
globalRegistryPassword: &global_registry_password Password@1234567890
trustStorePassword: &trust_store_password password
```

### 2. Update the values for OTDS.

- a. CryptKey is the encryption key for encrypting secrets and passwords to other systems in the OTDS DB. The value is a 16-character ASCII string that is base64 encoded.

For example, set the value to the base64 encoded value of the key. Here, "MTIzNDU2Nzg5YWwiZGVmZW==" is the base64 encoded value of "123456789acdefg".

Use the following to convert any text to base64 encoded format:

```
> echo -n "123456789acdefg" | base64 -w 0 && echo
MTIzNDU2Nzg5YWwiZGVmZW==
```

- b. Update the admin and database password values.

```
otdsAdminPassword: &otds_admin_password <Admin user password> # Admin user
password used to login into OTDS. Eg. Opentext1!
otds:
  otds_ws:
    cryptKey: Z2hkN2hyNDBkbWNGcVQ0TA==
    otdsdb:
      password: <password>
```

3. External PostgreSQL database configuration:

```
cs-secrets:
  database:
    password:*db_admin_password
```

4. Update the Appworks-Gateway/d2mobile values.

a. Update the database admin and appworksdb user password details:



**Note:** If database does not exist, the pg init container creates the database.

```
appworks-gateway:
  database:
    admin:

    password: <database admin password> # For example, password

  appworksdb:
    password: <appworksdb user password> # For example, password
```

b. Update appworks gateway admin user password.

```
appworks-gateway:
  awg:
    admin:
      newadminpassword:<appworks gateway admin user password> # For example,
password
```

5. **cs-secrets:** Secrets file contains sensitive information such as passwords and certificates.

1. Update the Documentum Reports with the required password.

```
#Client app secrets are going to add here
clients:
  drServiceAccountUser: dctmreports
  drServiceAccountPassword: <Password for Reports service account
Ex:Password@1234567890>
```

## 2.10 Updating passwords\_vault.yaml file

The `./documentum/config/passwords_vault.yaml` file contains the passwords updated as per the required format.

For more information about Documentum Vault secrets and key names, see *OpenText Documentum Content Management - Cloud Deployment Guide (EDCSYCD-IGD)*.

The following is a sample template for `password_vault` file:

```
installOwnerPassword: &installOwner_password INSTALL_OWNER_PASSWORD/dcs-pg-0.dcs-
pg.<namespace>.svc.cluster.local
databaseAdminPassword: &db_admin_password DATABASE_PASSWORD/<DBservicename>_<username> #
eg: DATABASE_PASSWORD/MyPostgres_postgres

globalRegistryPassword: &global_registry_password GLOBAL_REGISTRY_PASSWORD/<docbasename>

---
cs-secrets:
  docbase:
    password: DOCBASE_PASSWORD/<docbasename>

---
aek:
  oldPassphrase:
  passphrase: AEK_PASSWORD/<aek_name>

---
install:
  appserver:
    admin:
      password: DCTM_SERVER_JMX_PASSWORD/dcs-pg-0.dcs-pg.<namespace>.svc.cluster.local

      root:
        password: USER_PASSWORD/cs_root_user

---
```

## 2.11 Updating platforms/gcp.yaml

In the `documentum/platforms/gcp.yaml` file, update the `ingress.class` for `dctm` internal ingress, `vpn` ingress, `D2` public ingress, `appworks` gateway ingress, and `otiv` ingress ingress objects.

```
#These are specific configuration values passed for GCP. Other values for dctm-ingress
still need to be configured in main values.yaml only.

dctm-ingress:
  ingress:
    class: nginx

vpningress:
  ingress:
    class: nginx

d2publicingress:
  ingress:
    class: nginx

appworks-gateway:
  ingress:
    annotations:
      kubernetes.io/ingress.class: nginx
```

```
otiv:
  global:
    ingressClass: nginx
```

## 2.12 Updating platforms/aws.yaml

In the `documentum/platforms/aws.yaml` file, update the `ingress.class` for `dctm` internal ingress, `vpn` ingress, `D2` public ingress, `appworks` gateway ingress, and `otiv` ingress ingress objects.

```
#These are specific configuration values passed for AWS. Other values for dctm-ingress
still need to be configured in main values.yaml only
dctm-ingress:
  ingress:
    class: alb

  annotations:
    alb.ingress.kubernetes.io/scheme: internet-facing
    alb.ingress.kubernetes.io/subnets: <Available public subnet for the instance
separated by comma>

vpningress:
  ingress:
    class: alb

  annotations:
    alb.ingress.kubernetes.io/scheme: internet-facing
    alb.ingress.kubernetes.io/subnets: <Available public subnet for the instance
separated by comma>

d2publicingress:
  ingress:
    class: alb

  annotations:
    alb.ingress.kubernetes.io/scheme: internet-facing
    alb.ingress.kubernetes.io/subnets: <Available public subnet for the instance
separated by comma>

appworks-gateway:
  ingress:
    class: alb

  annotations:
    alb.ingress.kubernetes.io/scheme: internet-facing
    alb.ingress.kubernetes.io/subnets: <Available public subnet for the instance
separated by comma>

  annotations:
    alb.ingress.kubernetes.io/scheme: internet-facing
    alb.ingress.kubernetes.io/subnets: <Available public subnet for the instance
separated by comma>

otiv:
  global:
    ingressClass: alb

ingressannotations:
  alb.ingress.kubernetes.io/scheme: internet-facing
  alb.ingress.kubernetes.io/subnets: <Available public subnet for the instance
separated by comma>
```

## 2.13 Integrating Brava! Enterprise



**Note:** The followings steps are applicable when Brava! Viewer is enabled.

1. Enable component in the `documentum/documentum-components.yaml` file.  
For additional information about enabling Brava! Viewer component, see [“Updating documentum-components.yaml file” on page 20](#).

2. Update the following in the `documentum/charts/d2classic/templates/d2classic-bravaparameters-properties.yaml` file:

```
data:
  brava_parameters.properties: |-
    # Brava! For D2 Configuration Parameters

    EnableMarkupConsolidate=TRUE
    EnableConsolidateIfMarkupOpenedForEdit=TRUE
    html.markupsFeature.EnableCommonOwnership=TRUE
    html.enable.Markup.changeOwner=TRUE
    html.markup.enable.consolidate.changeownership.with.permission.higher.than=4
```

3. Update the following in the `documentum/charts/d2smartview/templates/d2smartview-bravaparameters-properties.yaml` file.

```
data:
  brava_parameters.properties: |-
    # Brava! For D2 SmartView Configuration Parameters

    EnableMarkupConsolidate=TRUE
    html.markupsFeature.EnableCommonOwnership=TRUE
    html.enable.Markup.changeOwner=TRUE
    html.markup.enable.consolidate.changeownership.with.permission.higher.than=4
```

4. Copy the certificate that is used during the Brava! Server SSL setup to the following path:

```
ao/charts/d2classic/certificates/<certificate_name>.crt
ao/charts/d2smartview/certificates/<certificate_name>.crt
```

5. Enable `bravatls` and certificate name in the `documentum/values.yaml`.

```
d2classic:
  bravatls:
    enable: true
    certName: <certificate_name> For example: brava.crt

d2smartview:
  bravatls:
    enable: true
    certName: <certificate_name> For example: brava.crt
```

## 2.14 Integrating Intelligent Viewer

1. Enable OTIV component in the `documentum/documentum-components.yaml` file.  
For additional information about enabling OTIV component, see [“Updating documentum-components.yaml file” on page 20](#).
2. Update the `trustedSourceOrigins` in the `documentum/config/configuration.yaml` file with the ingress domain URL through which the Smart View service can access.

```
otiv:
  global:
    trustedSourceOrigins: <ingress domain> For example, https://dctm-
    ingress.d2.cfcf-lab.bp-paas.otlab.net
```

## 2.15 Integrating xECM

Perform these steps only if *Extended ECM for Documentum SAP Solution* integration is required.

1. Update `authMode`, enable `CookieConfiguration`, and `ContentConnect` and update the following values in the `documentum/config/configuration.yaml` file.

```
d2smartview:
  restApiRuntime:
    OTDS:
      authMode: ct-otds_ticket-otds_token
      CookieConfiguration:
        enable: true
      ContentConnect:
        enable: true
        restAllowedOrigins: frame-ancestors $PROTOCOL$://$OTCS_PUBLIC_URL$ $PROTOCOL
        $://$SAP_HTMLVIEWER_URL$

# Eg. https://otcs.ao-dev.cfcf-lab.bp-paas.otxlab.net https://
saphtmlphtmlviewer.sap.com
```

2. Update the `disableUserAgents` with the following values:

```
d2smartview:
  restApiRuntime:
    disableUserAgents: Poco
```

## 2.16 Updating xPlore



**Note:** For more information about xPlore, see *xPlore Patch 14 and xPlore documentation*.

1. Enable xPlore by updating flag in the `documentum/documentum-components.yaml` file.

```
#Documentum xPlore
xplore:
  enabled: true
```

For additional information about enabling xPlore component, see [“Updating documentum-components.yaml file” on page 20](#).

2. Update ingress hostName and domain in documentum/config/configuration.yml file.

```
xplore:
  ingress:
    hostName: <ingress prefix> # For example, dctm-ingress
    domain: <ingress domain> # For example, ao-qa.cfcr-lab.bp-paas.otxlab.net
```



## Chapter 3

# Deploy Documentum Helm charts

Documentum chart has resources YAML files. The following table displays resources YAML files of different sizes.

Deployment size	Resource values-file name
Small	documentum-resources-values-small-enhanced.yaml
Small-Medium	documentum-resources-values-small-medium-enhanced.yaml
Medium	documentum-resources-values-medium-enhanced.yaml
Medium-Large	documentum-resources-values-medium-large-enhanced.yaml
Large	documentum-resources-values-large-enhanced.yaml
Extra-Large	documentum-resources-values-extra-large-enhanced.yaml
DevOPS Test	documentum-resources-values-test-small.yaml

**To deploy the Helm charts, perform the following steps:**

1. Run the following Helm command to validate the YAML files:

```
cd <Documentum chart directory>

For Vault enabled deployment

helm install <deployment_name> --values config/configuration.yaml --values config/
constants.yaml --values .\config\passwords.yaml (or) passwords_vault.yaml

platforms/<platform>.yaml --values dockerimages-values.yaml --values documentum-
resources-values-<config>.yaml --values documentum-components.yaml

--values addons/ao/ao-config.yaml -n <namespace> --debug --dry-run

#For example: helm install aodeployment . --values config/configuration.yaml --
values config/constants.yaml --values config/passwords_vault.yaml

--values platforms/gcp.yaml --values dockerimages-values.yaml --values documentum-
resources-values-test-small.yaml

--values documentum-components.yaml --values addons/ao/ao-config.yaml -n ao --debug
--dry-run

For non Vault enabled deployment

helm install <deployment_name> --values config/configuration.yaml --values config/
constants.yaml --values config/passwords.yaml --values
```

```
platforms/<platform>.yaml --values dockerimages-values.yaml --values documentum-
resources-values-<config>.yaml --values documentum-components.yaml

--values addons/ao/ao-config.yaml -n <namespace> --debug --dry-run

#For example: helm install aodeployment . --values config/configuration.yaml --
values config/constants.yaml --values config/passwords.yaml

--values platforms/gcp.yaml --values dockerimages-values.yaml --values documentum-
resources-values-test-small.yaml

--values documentum-components.yaml --values addons/ao/ao-config.yaml -n ao --debug
--dry-run

#For Vault enabled deployment with vaultType as HarshiCorp

helm install <documentumDeployment_name> . --values .\config\configuration.yaml --
values .\config\constants.yaml --values .\config\passwords_vault.yaml --
values .\platforms\<platform_yaml> --values .\dockerimages-values.yaml --values
<resources_yaml_file>.yaml --values .\documentum-components.yaml -n <namespace>
```



**Note:** The dry run install Helm command must not return any error message.

2. Run the following Helm install command to deploy the Helm charts:



**Note:** Based on the required deployment size (small, medium, large, and extra), select the resource values file from the documentum directory and use it in the deployment command.

```
cd <Docuemtnum chart directory>

helm install <deployment_name>. --values config/configuration.yaml --values config/
constants.yaml --values config/passwords.yaml --values platforms/<platform>.yaml --
values dockerimages-values.yaml --values documentum-resources-values-<config>.yaml
--values documentum-components.yaml --values addons/ao/ao-config.yaml -n <namespace>

For example, helm install aodeployment . --values config/configuration.yaml --values
config/constants.yaml --values config/passwords.yaml --values platforms/gcp.yaml --
values dockerimages-values.yaml --values documentum-resources-values-test-
small.yaml --values documentum-components.yaml --values addons/ao/ao-config.yaml -n
ao
```

3. Verify the status of the deployment of Helm using the following command format:
 

```
helm status <release name>
```
4. For more information about installing Transformation Services, see *OpenText Documentum CM Transformation Services Installation Guide*.



**Note:** Only Transformation Services documents has to be installed and configured with Documentum Server.

5. Install Blazon. For more information about installing Blazon, see *OpenText Blazon Enterprise server Installation Guide*.

## 3.1 Checking deployment status

1. After you deploy the Helm charts, run the following command to verify that all the pods are up and running:



**Note:** Some pods may take 70–90 minutes for the instances to start.

```
kubectl get pods -n <namespace>
```

2. a. If all the pods are up and running and if there are two instances of dcs-pg, perform the following steps to decrease (descale) or increase (scale) the number of replicas:

- Perform the following steps to decrease the replicas:

- A. Retrieve the statefulset of the namespace using the following command:

```
kubectl get sts -n <name-space>
```

- B. Edit the dcs-pg statefull set:

```
kubectl edit sts dcs-pg -n <name-space>
```

- C. Update the replicas set to 1 and wait for the dcs-pg-1 pod to go down:

```
spec:
  ...
  ...
  replicas: 1
```

- b. Perform the following steps to increase the replicas:

- Repeat the steps to get the statefulset and update the replicas set to 2.

3. You can use `kubectl logs <pod name>` to check all the pods.

```
kubectl logs -f <pod name> -c <container_name> -n <namespace>
```

4. If you do not know the container name, you can run the following command to retrieve the container names that exist in the pod.

```
kubectl describe pod <podname> -n <namespace>
```

## 3.2 Accessing application

You can access the Documentum Administrator and OTDS Admin site through ingress with the following URLs:

You can access other clients after you complete the post deployment steps.

Component	URL format	Example
Documentum Administrator	<ingress-protocol>://<dctm-ingress-host>/da?skipssso=true	https://dctm-ingress.otxlab.net/da?skipssso=true
OTDS Admin site	<ingress-protocol>://<dctm-ingress-host>/otds-admin	https://dctm-ingress.otxlab.net/otds-admin
Appworks Gateway	<ingress-protocol>://<appworks-ingress-host>	https://appworks-gateway.otxlab.net

## Chapter 4

# Post deployment



**Note:** For xECM integration, deploy xECM Helm chart and then continue with OTDS configuration. For more information about xECM deployment, see *OpenText Documentum Content Management for Engineering - Integration Guide* (EEGAM250400-ING).

## 4.1 Setting up OpenText Directory Services

### 4.1.1 Creating an OAuth client

For more information about creating a OAuth client, see *OpenText Directory Services - Installation and Administration Guide* (OTDS250300-IWC).

**To create an OAuth client:**

1. You can use the following customized values for OpenText Documentum CM for Engineering.

Step Heading	Description
ClientID	Provide the same oauthClient name as provided in the values .yaml file. For example: <code>oauthClient: &amp;otds_oauth_client d2_oauth_client</code>
Partitions	Select <b>Global</b> option.

Step Heading	Description
Redirect	<p>Specify the following URLs:</p> <ul style="list-style-type: none"> <li>• &lt;Ingress URL&gt;/D2/d2_otds.html For example: https://dctm-ingress.dcme.otxlab.net/D2/d2_otds.html</li> <li>• &lt;Ingress URL&gt;/D2/OTDSLogoutResponse.html For example: https://dctm-ingress.dcme.otxlab.net/D2/OTDSLogoutResponse.html</li> <li>• &lt;Ingress URL&gt;/D2-Smartview/ui For example: https://dctm-ingress.dcme.otxlab.net/D2-Smartview/ui</li> <li>• &lt;Ingress URL&gt;/D2-Config/d2config_otds.html For example: https://dctm-ingress.dcme.otxlab.net/D2-Config/OTDSLogoutResponse.html</li> <li>• &lt;Ingress URL&gt;/D2-Config/OTDSLogoutResponse.html For example: https://dctm-ingress.dcme.otxlab.net/D2-Config/d2config_otds.html</li> <li>• &lt;Ingress URL&gt;/da For example: https://dctm-ingress.ao-qa.cfcrlab.bp-paas.otxlab.net/da</li> </ul>

2. In the **Auth Handlers** page, select **http.negotiate**, click **Actions** and select **Disable**.

### 4.1.2 Configuring OTDS to sync users with Documentum repository

1. Access the OTDS admin website <dctm-ingress-url>/otds-admin with the admin as the user name and the value of the password specified in Helm chart's values.yaml file.
2. Add a partition. Select New Non-synchronized User Partition. For information about creating a Partition, see *OpenText Directory Services - Installation and Administration Guide (OTDS250300-IWC)*.
3. Create roles that define user privileges and client capability.



**Note:** OpenText recommends to use OTDS to manage the privileges and client capability instead of updating Documentum repository.

4. Create the required roles. For information about creating Roles, see *OpenText Directory Services - Installation and Administration Guide (OTDS250300-IWC)*.

You can assign roles as per the required user privilege and client capability. The following are sample user privileges and client capabilities.

Role Name	Description
<b>User Privileges</b>	
privilege_none	Specifies that there is no privilege.
privilege_createtype	Specifies the create privilege.
privilege_createcabinet	Specifies create cabinet privileges.
privilege_createtypeandcabinet	Specifies the create type and cabinet privilege.
privilege_creategroup	Specifies the create group privilege.
privilege_creategroupandtype	Specifies the create group and type privilege.
privilege_creategroupandcabinet	Specifies the create group and cabinet privilege.
privilege_creategroupcabinetandtype	Specifies the create group, cabinet and type privilege.
privilege_sysadmin	Specifies the sysadmin privilege.
privilege_superuser	Specifies the superuser privilege.
<b>Client capability</b>	
Client_Consumer	Specifies the consumer capability.
Client_Contributor	Specifies the contributor capability.
Client_Coordinator	Specifies the coordinator capability.
Client_System_Administrator	Specifies the system Administrator capability.

5. Add the required resources. For more information about resources, see *OpenText Directory Services - Installation and Administration Guide (OTDS250300-IWC)*.

The following values can be customized for resources.

### Synchronization

Select **User and group synchronization**, **Create users and groups**, and **Modify users and groups**.

For Synchronization connector, you can select REST (Generic).

### Connection information

Base URL: You can use

`http://<jms-service-name>:<jms port>/dmotdsrest` For example,  
`http://dcs-pg-jms-service:9080/dmotdsrest`. These values are specified in `values.yaml` helm chart.

```
jmsServiceName: &jmsServiceName dcs-pg-jms-service
jmsPort: &jms_port 9080
```

Username: `<docbase name>/<install owner name>`. For example, `docbase1\dmadmin`

Password: <install owner password>

---

### User Attribute Mappings

Add the following resource attributes.

Resource Attribute	OTDS Attribute(s)	Format
default_folder	cn	/Users/AOUsers
user_name	cn	%s
user_address	mail	%s
client_capability	null	2
create_default_cabinet	null	F
user_privileges	null	0
user_rename_enabled	null	F
user_rename_unlock_locked_obj	null	T
user_type	null	dm_user
user_xprivileges	null	32
user_login_name	oTExternalID3	%s
user_global_unique_id	oTObjectGUID	%s

---

Use the existing values for other resource attributes and click **Save**.

6. Create the required access roles. For information about creating Access Roles, see *OpenText Directory Services - Installation and Administration Guide (OTDS250300-IWC)*.

If *otadmin@otds.admin* user or *otdsadmins@otds.admin* group exists, remove the same and save the access role configurations.

7. In the **Resources**, select the required resource and click **Actions > Consolidate** to sync members to Documentum repository.

## 4.1.3 Licensing OpenText Documentum CM for Engineering

OpenText Documentum CM uses OTDS to apply the licenses for all the OpenText Documentum CM components.

For more information about procuring the license file and configuring OTDS and license, see *OpenText Documentum Content Management - Cloud Deployment Guide (EDCSYCD-IGD)*.

### 4.1.4 Updating OTDS configuration in Documentum Administrator

For more information, see *OpenText Documentum Content Management - Administrator User Guide (EDCAC250400-UGD)*.

You can access other clients using the following links:

Component	URL format	Example
Documentum Administrator	<ingress-protocol>://<dctm-ingress-host>/da?skipssso=true	https://dctm-ingress.otxlab.net/da?skipssso=true
D2-Config	<ingress-protocol>://<dctm-ingress-host>/D2-Config	https://dctm-ingress.otxlab.net/D2-Config
D2 Client	<ingress-protocol>://<dctm-ingress-host>/D2	https://<dctm-ingress-host>/D2
D2 Smart View	<ingress-protocol>://<dctm-ingress-host>/D2-SmartView	https://<dctm-ingress-host>/D2-SmartView

## 4.2 Configuring Reports

To enable and configure Documentum Content Management Reports, administrators must ensure secure access and integrate with the reporting framework. This includes enabling the license, creating and verifying report users in Documentum Administrator and the reporting application, and configuring report options in the Client Configuration tool. Finally, apply permissions for the Reports Templates folder in Documentum Administrator to allow controlled access and report execution in SmartView.

### 4.2.1 Enable License for Documentum Reports

Create a Documentum Reports user and a Reports Admin user in OTDS, and enable the required license to access the reporting feature.

#### To create documentum reports user:

1. For OpenText Documentum CM for Engineering application, in OTDS, create an OTDS user with the user name as **dctmreports** and keep the password as blank.
2. In OTDS, go to **Partition > Actions > Allocate to License**.
3. Select the **SYSTEM ACCOUNT** license.

You must not consolidate this user.

**To create reports admin user:**

1. For OpenText Documentum CM for Engineering application, in OTDS, create an OTDS user with the user name as **Reports Admin**.
2. In OTDS, go to **Partition > Actions > Allocate to License**.
3. Select the required **X** plan and **Documentum.Eng** license.  
You must consolidate this user.

## **4.2.2 Creating users in Documentum Administrator to access reports**

Configure the Reports users in Documentum Administrator to access the reports.

**To assign group and role:**

1. Go to **Administration > User Management > Users**.
2. Select the **Reports Admin** user, right-click Reports Admin user, and select **Assign Group Membership**.
3. Search for **dctm-reports-user** group and add it to the selected list using the arrow keys.
4. Search for **dctm-reports-designer** role, add it to the selected list using the arrow keys, and click **OK**.

## **4.2.3 Verifying the reports user in Documentum Reports**

Login to OpenText Documentum CM for Engineering application with the newly created user credentials. For example: `<Ingress URL>/D2-Smartview/`

You can view the Reports workspace landing page.

## **4.2.4 Configuring Reports in client configuration**

1. Sign in to OpenText Documentum CM for Engineering client configuration application.
2. Select **Tools > Options** from the menu bar.
3. Select the **DCTM Reports Options** tab.
4. In the **DCTM Reports Configurations** pane, provide the following values:
  - a. Core Application URL: Type a core application path. For example: `http://<fully qualified ingress host>dtr`
  - b. Date Pattern: Type a date pattern. For example: `M/dd/yyyy hh:mm:ss a`
  - c. Framework: Type a framework value. For example: `HTML5`

- d. **Preview Line:** Select this option to display reports execution results within the SmartView context.

### 4.2.5 Update Documentum Reports Templates folder

1. Sign in to Documentum Administrator with Administrator credentials.
2. Select **Cabinets > Templates**.
3. Right-click **DCTM Reports Templates** and select **Properties**.
4. In the **Properties** dialog box, select the **Permissions** tab.
5. Click **Select** permission set name.
6. In the **Choose a permission set** dialog box, search for **DCTM-Reports-Private-ACL**.
7. Select the **DCTM-Reports-Private-ACL** option.
8. Click **OK**.

## 4.3 Configuring OTDS to enable Intelligent Viewer

Follow the steps documented in the respective sections to configure OTDS:

1. **Synchronizing Intelligent Viewing users to repository:** For information about Synchronizing Intelligent Viewing users to repository, see *OpenText Documentum Content Management - Cloud Deployment Guide (EDCSYCD-IGD)*.
2. **Allocating the license to users:** For information about Allocating the license to users, see *OpenText Documentum Content Management - Cloud Deployment Guide (EDCSYCD-IGD)*.
3. **Adding or updating OAuth clients and system attributes:** For information about Adding or Updating OAuth clients and system attributes, see *OpenText Documentum Content Management - Cloud Deployment Guide (EDCSYCD-IGD)*.

After installing Intelligent Viewing and applying OTDS licenses, you must configure Intelligent Viewing using OpenText Documentum CM for Engineering client configuration. For details on configuring Intelligent Viewing, see *OpenText Documentum Content Management - Client Configuration Guide (EDCCL-AGD)*.

## 4.4 Activating resource ID in Documentum Server



**Note:** This is applicable only for xECM integration.

You can activate the resource ID by using the OTDS Swagger user interface.

### To activate the resource ID:

1. Copy the resource ID from OTDS.
  - a. Go to OTDS admin `<dctm-ingress-url>/otds-admin` website with admin as the username and the value of the password as specified in the Helm chart's `values.yaml` file.
  - b. Click **Resources**.
  - c. Copy the Documentum CS resource ID.
2. Perform the following to activate the resource ID:
  - a. Open the following Swagger URL: `https://<otds url>/otdsws/api/index.html?rest#/resources/activateResource_1`.  
For example, `https://dctm-ingress.ao.cfcrlab.bp-paas.otxlab.net/otdsws/api/index.html?rest#/resources/activateResource_1`
  - b. Go to **resources**.
  - c. In **resources**, select **Activate a resource**.
  - d. Click **Try it out**.
  - e. Type the Documentum Server resource ID. You can locate the resource ID from the **Resources** in OTDS.
  - f. Click **Execute**.
  - g. Copy the `secretKey` from the response.
3. Update the `<docbase>_resource_id` and `<docbase>_secretKey` environment variables with resource ID and `secretKey` in `documentum/config/configuration.yml` file.

```
content-server:
  extraEnv:
    - name: <docbase_name>_resource_id # Eg. docbase1_resource_id
      value: "<Documentum CS resource id from OTDS resources tab>" # Eg.
"413f48bb-dea5-49dd-999e-7ed6ad565730"
    - name: <docbase_name>_secretKey # Eg. docbase1_secretKey
      value: "<Secret Key obtained by activating resource>" # Eg.
"5vk0bYZSt8Pe04gSby3sVA=="
```

4. Run the following command to perform Helm upgrade:

```
cd <Asset Operations master chart directory>

helm upgrade <deployment_name>. --values config/configuration.yml --values config/
constants.yaml
```

```
--values config/passwords.yaml --values platforms/<platform>.yaml

--values dockerimages-values.yaml --values documentum-resources-values-<config>.yaml

--values documentum-components.yaml -n <namespace>

# For example:helm upgrade aodeployment . --values config/configuration.yaml --
values config/constants.yaml

--values config/passwords.yaml

--values platforms/cfcr.yaml --values dockerimages-values.yaml --values documentum-
resources-values-test-saml1.yaml

--values documentum-components.yaml -n d2
```

## 4.5 Configuring Transformation Services

For more information about installing Transformation Services, see *OpenText Documentum Content Management - Transformation Services Installation Guide (EDCCT-IGD)*.

### 4.5.1 Configuring the system for generating PDF rendition

1. Sign in to Documentum Administrator with administrator credentials.
2. Navigate to **Cabinets > System > Media Server > System Profiles**.
3. In **Filter** by list, select **Show All Objects and Versions**.
4. Edit the CURRENT `register.xml` and `register_legacy.xml` files.



**Note:** Do not modify the headers.

Add the following content in the `register_legacy.xml` file.

```
<InnerProfile path="/System/Media Server/System Profiles/document_to_pdf"
waitOnCompletion="true" useTargetFormat="true">

<InnerTokenMapping LocalProfileToken="pdf"
InnerProfileToken="doc_token_targetFormat" Literal="true"/>

</InnerProfile>
```

Add the following content in the `register.xml` file.

Formats section:

```
<Format source="pdf" target="pdf"/>
<Format source="excel8book" target="excel8book"/>
<Format source="excel8template" target="excel8template"/>
<Format source="excel12book" target="excel12book"/>
<Format source="excel12template" target="excel12template"/>
<Format source="excel14book" target="excel14book"/>
<Format source="excel14template" target="excel14template"/>
<Format source="msw8" target="msw8"/>
<Format source="msw8template" target="msw8template"/>
<Format source="msw12" target="msw12"/>
<Format source="msw12template" target="msw12template"/>
<Format source="msw14" target="msw14"/>
<Format source="msw14template" target="msw14template"/>
<Format source="msg" target="msg"/>
```

```
<Format source="ppt8" target="ppt8"/>
<Format source="ppt8_slide" target="ppt8_slide"/>
<Format source="ppt8_template" target="ppt8_template"/>
<Format source="ppt8slideshow" target="ppt8slideshow"/>
<Format source="ppt12" target="ppt12"/>
<Format source="ppt12_slide" target="ppt12_slide"/>
<Format source="ppt12template" target="ppt12template"/>
<Format source="ppt12slideshow" target="ppt12slideshow"/>
<Format source="ppt14" target="ppt14"/>
<Format source="ppt14_slide" target="ppt14_slide"/>
<Format source="ppt14template" target="ppt14template"/>
<Format source="ppt14slideshow" target="ppt14slideshow"/>
<Format source="pub_html" target="pub_html"/>
<Format source="tiff" target="tiff"/>
<Format source="ppt15" target="ppt15"/>
<Format source="ppt15_slide" target="ppt15_slide"/>
<Format source="ppt15template" target="ppt15template"/>
<Format source="ppt15slideshow" target="ppt15slideshow"/>
<Format source="msw15" target="msw15"/>
<Format source="msw15template" target="msw15template"/>
<Format source="excel15book" target="excel15book"/>
<Format source="excel15template" target="excel15template"/>
```

Filters section:

```
<Filter name="CTSPProduct" value="ADTS"/>
```

ProfileSequence section:

```
<InnerProfile path="/System/Media Server/System Profiles/document_to_pdf"
waitOnCompletion="true" useTargetFormat="true">
  <InnerTokenMapping LocalProfileToken="pdf"
  InnerProfileToken="doc_token_targetFormat" Literal="true"/>
  <InnerTokenMapping LocalProfileToken="PDFVersion14"
  InnerProfileToken="doc_token_pdfVersion" Literal="true"/>
  <InnerTokenMapping LocalProfileToken="Automatic"
  InnerProfileToken="doc_token_usePrinterMetrics" Literal="true"/>
  <InnerTokenMapping LocalProfileToken="600" InnerProfileToken="doc_token_resolution"
  Literal="true"/>
  <InnerTokenMapping LocalProfileToken="Yes" InnerProfileToken="doc_token_optimize"
  Literal="true"/>
  <InnerTokenMapping LocalProfileToken="Yes"
  InnerProfileToken="doc_token_enableBookMarks" Literal="true"/>
  <InnerTokenMapping LocalProfileToken="DocumentContent"
  InnerProfileToken="doc_token_printType" Literal="true"/>
  <InnerTokenMapping LocalProfileToken="true" InnerProfileToken="overwrite_rendition"
  Literal="true"/>
  <InnerTokenMapping LocalProfileToken="legacy"
  InnerProfileToken="transformation_type" Literal="true"/>
  <InnerTokenMapping LocalProfileToken="No"
  InnerProfileToken="doc_token_enableSecurity" Literal="true"/>

  <InnerTokenMapping LocalProfileToken="40bit"
  InnerProfileToken="doc_token_encryptionMode" Literal="true"/>
  <InnerTokenMapping LocalProfileToken="Disabled"
  InnerProfileToken="doc_token_changesAllowed" Literal="true"/>
  <InnerTokenMapping LocalProfileToken="Disabled"
  InnerProfileToken="doc_token_enableAccess" Literal="true"/>
  <InnerTokenMapping LocalProfileToken="Disabled"
  InnerProfileToken="doc_token_docAssembly" Literal="true"/>
  <InnerTokenMapping LocalProfileToken="Disabled"
  InnerProfileToken="doc_token_formFieldFilling" Literal="true"/>
  <InnerTokenMapping LocalProfileToken="Disabled"
  InnerProfileToken="doc_token_printing" Literal="true"/>
  <InnerTokenMapping LocalProfileToken="Disabled"
  InnerProfileToken="doc_token_allowCopy" Literal="true"/>
  <InnerTokenMapping LocalProfileToken=" " InnerProfileToken="doc_token_sec0pass"
  Literal="true"/>
  <InnerTokenMapping LocalProfileToken=" " InnerProfileToken="doc_token_secCpass"
```

```
Literal="true"/>
</InnerProfile>
```

5. Execute the following query using the DQL editor:

```
update dm_format objects set richmedia_enabled=1 where name like 'msw%'
```

6. Restart all CTS services.

## 4.6 Configuring Blazon with Brava Enterprise Viewer

Use the following content to configure comment consolidation with Brava Enterprise viewer.

For more information about installing Blazon, see *OpenText Blazon Enterprise server Installation Guide*.

### 4.6.1 Mounting Kubernetes volume on Windows virtual machine

`/opt/dctm/aoblazon` is the volume claim template referred in Helm charts for Blazon folder mapping.

**To enable the NFS service on the Blazon virtual machine:**

1. Sign in to the Windows Blazon virtual machine and enable the NFS service.

- a. Run the following command:

```
powershell -Command "Start-Process PowerShell -Verb RunAs"
```

- b. Use the following command to enable the NFS service:

```
Import-Module ServerManager
Install-WindowsFeature -Name FS-NFS-Service
Install-WindowsFeature NFS-Client
```

- c. Stop **nfsadmin** client using the following command:

```
nfsadmin client stop
```

- d. Add registry keys in Windows virtual machine using the following command:

```
New-ItemProperty HKLM:\SOFTWARE\Microsoft\ClientForNFS\CurrentVersion\Default -
Name AnonymousUID -Value 1000

-PropertyType "DWord"

New-ItemProperty HKLM:\SOFTWARE\Microsoft\ClientForNFS\CurrentVersion\Default -
Name AnonymousGID -Value 1000

-PropertyType "DWord"
```

- e. Start **nfsadmin** client using the following command:

```
nfsadmin client start
```

- f. In command prompt, execute the following command:

```
nfsadmin client localhost config fileaccess=755 SecFlavors=sys -krb5 -krb5i
```

2. Run the following command to retrieve the NFS path:

```
kubect1 exec -it dcs-pg-0 -c dcs-pg -it -n <namespace> -- df /opt/dctm/aoblazon
```

3. Sign in to the Windows Blazon virtual machine.
4. Map the NFS path to the Windows Blazon virtual machine.
  - a. In the virtual machine, open **File Explorer**, right-click **This PC**, and select **Map network drive**.
  - b. In the **Map network Drive** dialog box:
    - i. Select the required drive.
    - ii. For Folder, type the NFS path.



**Note:** Update the path with a backward slash and remove the colon.

For example: \\<host\_name>\trident\_qtree\_pool\_ao\_qa\_PNUUDIJZHI\ao\_qa\_pvc\_b9fac521\_bedb\_4146\_bed1\_9125484178b8\aoablazon

- iii. Click **Finish**.
5. Restart the Blazon virtual machine.

## 4.7 Configuring Blazon details using Transmittal Comment Config

You can configure burn-in comments on PDF using Blazon:

**To configure Blazon in client configuration:**

1. Sign in to client configuration with administrator credentials.
2. Navigate to **D2 > Dictionary > Transmittal Comment Config**.
3. On the **Alias** tab, update **URL**, **Time Out**, **Source Location**, and **Publish Location**.

For example, URL = `http://<Blazon Server Hostname>:8090/QueueServer/push.aspx`  
Time Out = 180000  
Source Location = `/opt/dctm/aoblazon`  
Publish Location = (Provide Trident NFS path)
4. Click **Save**.

## 4.8 Configuring SMTP server details

1. Sign in to Documentum Administrator with administrator credentials. For example, `<dctm-ingress-url>/da?skipssso=true` with install owner credentials.
2. Navigate to **Cabinets > System > EPFM > AO**.
3. Edit the current **System Transmittal Connection Config** file with SMTP details. For example:

host\_name = smtp\_hostname  
Port Number = smtp Port

```
<param name="Encrypted Password" encryption="dfc">password</param>
<param name="key">Email</param>
<param name="Host Name">host_name</param>
<param name="Port Number">25</param>
<param name="Is SSL"></param>
<param name="Is Start TLS"></param>
<param name="SSL Protocol"></param>

<param name="SSL Socket Factory Class"></param>
<param name="Is Socket Factory Fallback"></param>
<param name="Username">abc@domain.com</param>

<param name="Requires Authentication">false</param>

<param name="Enable Send As">false</param>
<param name="Proxy Host">IP Address</param>

<param name="Proxy Port">3128</param>

<param name="Proxy Username"></param>
<param name="Proxy Pwd"></param>
```

If you are using Microsoft Office 365 for sending emails, you must set **Enable Send As** to false.



**Note:** Certain email servers do not allow the user to send emails on behalf of other users. In such cases, you must set **Enable Send As** to false.

For other Mail servers, you can set **Enable Send As** to true.

4. Save the content and check in the file.

## 4.9 Updating EmailID for Install owner user



**Note:** In Documentum Administrator, the email ID of the dmadmin user must be in the `dmadmin@<domain>.com` format. Unless the dmdmin email ID is in this format, you cannot share the documents with Core Collaboration for Engineering .

1. Sign in to Documentum Administrator with administrator credentials. For example, `<dctm-ingress-url>/da?skipssso=true` with install owner credentials.
2. Navigate to **Repository > Administration > User Management > Users**.

3. Search for dmadmin or install owner user.
4. Click dmadmin or install owner user and select **Properties**.
5. Update the email ID of the dmadmin or install owner user. For example:  
dmadmin@opentext.com

## 4.10 Configuring Appworks Gateway

1. Sign in to the Appworks Gateway website with the awg administrator user credentials. For example, <https://appworks-gateway.ao.cfcrlab.bp-paas.otxlab.net>.
2. Click **Installed Applications**.
3. Click **Setting** in the OpenText Documentum CM Mobile tile.
4. Navigate to the **Settings** tab.
5. Update the Smart View URL. For example, <https://dctm-ingress.ao.cfcrlab.bp-paas.otxlab.net/D2-Smartview>.
6. Click **Update Configuration**.

## 4.11 Configuring workflow designer



**Note:** **step 1** and **step 2** are applicable only if OTDS is enabled for OpenText Documentum Content Management (CM) Workflow Designer.

**To configure workflow designer:**

1. Update the redirect URL in OTDS.
  - a. Go to the OTDS admin website, `<dctm-ingress-url>/otds-admin`.
  - b. Click **OAuth Clients**.
  - c. Select `d2_oauth_client`.
  - d. Click **Actions** and select **Properties**.
  - e. Click **Redirect URLs**.
  - f. Click **Add**.
  - g. For Resource URL, add workflow designer URL `<dctm-ingress-url>/DocumentumWorkflowDesigner`. For example, <https://dctm-ingress.ao.cfcrlab.bp-paas.otxlab.net/DocumentumWorkflowDesigner>.
  - h. Click **Save**.
2. Update OTDS client\_secret in Workflow Designer.
  - a. Update the `documentum/config/passwords.yaml` file with client\_secret values.

```
dctm-workflow-designer:
  otds:
```

```
client_secret: <CLIENT_SECRET> # For example,
721V2FwJzht9sY0uk0w2uoE77t1U9JTI
```

To retrieve the client secret key:

- i. Go to OTDS admin website <dctm-ingress-url>/otds-admin.
  - ii. Click **OAuth Clients**.
  - iii. Select **d2\_oauth\_client**.
  - iv. Click **Actions** and select **Properties**.
  - v. In the **Properties** dialog box, click **General**.
  - vi. Select **Confidential**.
  - vii. Copy the generated secret\_key.
- b. Run the following command to upgrade the Helm chart:

```
cd <Asset Operations master chart directory>

helm upgrade <deployment_name> . --values config/configuration.yaml --values
config/constants.yaml

--values config/passwords.yaml

--values platforms/<platform>.yaml --values dockerimages-values.yaml --values
documentum-resources-values-<config>.yaml

--values documentum-components.yaml -n <namespace>

# For example: helm upgrade aodeployment . --values config/configuration.yaml --
values config/constants.yaml

--values config/passwords.yaml

--values platforms/cfcr.yaml --values dockerimages-values.yaml --values
documentum-resources-values-test-saml1.yaml

--values documentum-components.yaml -n d2
```

If the Workflow Designer pod is not created after Helm upgrade, run the following command to recreate the pod:

```
# Scale down dctm-workflow-designer deployment
kubectl scale deployment dctm-workflow-designer --replicas=0 -n <namespace>

# Scale up dctm-workflow-designer deployment
kubectl scale deployment dctm-workflow-designer --replicas=1 -n <namespace>
```

3. To add the user to the *documentum\_workflow\_designer* role:



**Note:** Use the following steps to add OTDS users to OpenText Documentum Content Management (CM) Workflow Designer. Added users will be able to create and update the workflows in Workflow Designer.

- a. Sign in to Documentum Administrator with administrator privileges.
- b. Navigate to **Repository > Administration > User Management > Roles > Search**.
- c. Double-click **documentum\_workflow\_designer**.
- d. Click **File > Add Member(s)** and add the user.



## Chapter 5

# Upgrading OpenText Documentum CM for Engineering

### 5.1 Prerequisites

This section provides the information about the prerequisites tasks.

#### 5.1.1 Database prerequisites

Database requirement	Details
Name of the service using the database	ot-dctm-server
Database type	PostgreSQL on a virtual machine
Database version	PostgreSQL 17.3
Database Admin user roles and Privileges	Create role and create database privileges

#### 5.1.2 Helm version

You must use Helm client V3.15.2.

#### 5.1.3 Documentum Administrator and Workflow Designer



**Note:** Do a fresh Documentum Administrator deployment. Perform the following to downgrade or roll back the image details:

1. In the `dockerimages-values.yaml`, update the image tag and recreate the pod.
2. In the `documentum-components.yaml`, set the Documentum Administrator component to false and then run the helm upgrade.
3. Update the image tag with the required version.
4. After you upgrade the HELM, enable the Documentum Administrator component and re-run the HELM upgrade with the Documentum Administrator image details.

## 5.1.4 Backing up the database

Ensure that you back up the database before proceeding to the upgrade. Before the database backup, set the repository to the Dormant state.



**Note:** Contact OpenText Technical Services team to perform backup and restoration of your database.

### To convert the repository to dormant state:

1. Sign in to Documentum Administrator using `dmadmin` user credentials.
2. Create a new user with superuser privileges. For example: `backupadmin`
3. Add this `backupadmin` user to `dm_datacenter_manager` group. In addition, add `dmc_wdk_presets_owner`, `dmc_wdk_preferences_owner`, `dm_bof_registry` users to `dm_datacenter_manager` group.
4. Sign in to Documentum Administrator using `backupadmin` user credentials.
5. Navigate to **Administration > Basic Configuration > Repository**.
6. Right-click the repository name and click **Make Dormant**.  
Repository status changes to a dormant state and will not allow any new connections from clients.

## 5.1.5 Backing up the D2-Config

Backup the D2-config configuration before you start the upgrade tasks.

### To backup the D2-Config:

1. In OpenText Documentum CM client administration, click **File > Export Configuration**.
2. In the list, select **All Elements**.
3. Select **Full Config Export**.
4. Click **OK**.

Use the following code to clear the data dictionary and persistent cache:

```
First connect to primary cs pod using:
kubectl exec -ti pod/<CS_podname>-c <Cs_container_name>-n <namespace> bash
Start IAPI using iapi <docbase_name>

flush the cache by running the below commands through IAPI on Content Server:

flush,c,ddcache,dm_type
flush,c,ddcache,dmi_type_info
flush,c,ddcache,dm_aggr_domain
flush,c,ddcache,dm_domain
flush,c,ddcache,dm_dd_info
flush,c,ddcache,dm_nls_dd_info
flush,c,ddcache,dm_foreign_key
flush,c,persistentcache
```

```
then you can publish data dictionary:
```

```
publish_dd,c
reinit,c
```

Use 25.2 release deployed chart as a reference to enable the required pods in 25.4 release charts. Use `documentum-component.yaml` file to disable the remaining pods.

After you complete the backup, you must convert the repository dormant state to active state.

#### To convert the repository to active state:

1. Sign in to Documentum Administrator using `backupadmin` user credentials.
2. Navigate to **Administration > Basic Configuration > Repository**.
3. Right-click the repository name and click **Make Active**.

## 5.2 Upgrade steps

### 5.2.1 Download OpenText Documentum CM Helm chart

1. Download the OpenText Documentum CM from the OpenText registry server.

```
# Add Opentext registry
helm repo add opentext https://registry.opentext.com/helm --username <user email address> --password <user password>

# download Documentum helm chart
helm pull opentext/documentum --version 25.4
```

Use the following steps to download from LTI artifactory:

*Helm Chart Path:* `https://<LTI server path>/artifactory/BPhelm`

*Chart Name:* `documentum-25.4.0tgz`

2. Unzip helm chart using tar using the following command.

```
# Command to unzip helm chart
tar zxvf documentum-25.4.tgz
```

3. Use 25.2 release deployed chart as a reference to enable the required pods in 25.4 release charts. For example: namespace, host, database name, and docbase.
4. Use `documentum-component.yaml` file to disable the remaining pods that are not required for upgrade.

For example: The following example shows how to disable adminconsole by enabling the flag to false.

```
adminconsole:
  # Description: Indicates if the adminconsole pod deployment is enabled.
  enabled: false
```

5. Verify the latest OpenText Documentum CM for Engineering image tags in the `dockerimages-values.yaml` file.

6. Use the following steps to verify if the database version is same in existing and new deployment:

```
For example, if your 25.2 deployment is running with PostgreSQL 14.4
database container, then in your 25.4 upgrade, ensure that you change the
database version to 14.4 as follows:
· Value of version in the documentum/charts/db/Chart.yaml file.
· Value of db.images.db.tag in the documentum/dockerimages-values.
yaml file
```

7. Copy all the required configuration values from the content-server section of the previous deployment to the content-server section in the documentum/config/passwords.yaml, documentum/config/configuration.yaml, and documentum/values.yaml files of 25.4.  
If required, add appropriate values for the new variables.
8. For more details, see “Deploying OpenText Documentum CM for Engineering components” on page 19 to update the Helm chart.

### 5.2.2 To upgrade from HashiCorp Vault-enabled 24.4 or HashiCorp Vault-enabled 25.2 to HashiCorp Vault-enabled 25.4

1. Update the new image location details of db, docbroker, and content-server in the documentum/config/passwords\_vault.yaml file of 25.4.
2. Update the value of global.isVaultEnabled to true and the value of global.aekLocation to Remote\_Vault in the documentum/values.yaml file of 25.4. Then, update the value of global.vaultType to HashiCorp.
3. Set the value of global.secretConfigName to the name of the Kubernetes secret created using the documentum/config/vault\_secret.yaml file in the documentum/values.yaml file of Kubernetes native secrets-enabled 25.4 environment.
4. Update the value of global.isVaultEnabled to true and the value of global.aekLocation to Remote\_Vault in the documentum/values.yaml file of 25.4. Then, update the value of global.vaultType to HashiCorp.

Ensure that you copy all the password information from the previous deployment and store them in the HashiCorp Vault server in the <secret\_name> / <key\_name> format as mentioned in the documentum/config/passwords\_vault.yaml file.

5. To upgrade the AEK key, do the following:
  - a. Ensure that you update the new AEK key name (CSaek) in contentserver.contentserver.aek.name in the documentum/config/configuration.yaml file.
  - b. Upload the AEK passphrase from the previous deployment and the new AEK passphrase to the HashiCorp Vault server.
  - c. In the documentum/config/passwords\_vault.yaml file, do the following:

- i. Provide the secret name value for `cs-secrets.contentserver.aek.oldPassphrase`.
- ii. Set the value of `cs-secrets.contentserver.aek.passphrase` with the new key name.

For example:

```

aek:
oldPassphrase: AEK_PASSWORD/aek_name
passphrase: AEK_PASSWORD/CSaek

```

6. Upgrade the 24.4 or 25.2 Documentum CM Server pod using the following command formats:

```
helm list
```

7. Run the following command to validate the YAML files:

```

cd <path_to_ao_helm_chart_directory>

helm upgrade <deployment_name> . --values config/configuration.yml --values config/
constants.yml --values config/passwords.yml --values platforms/<platform>.yaml --
values dockerimages-values.yml --values addons/ao/ao-config.yml --values
documentum-resources-values-<config>.yaml --values documentum-components.yaml -n ao
--debug --dry-run

```

8. Use the following command if Vault is not enabled:

```

cd <path_to_ao_helm_chart_directory>

helm upgrade <deployment_name> . --values <location where Helm charts are
extracted>/config/configuration.yml --values <location where Helm charts are
extracted>/config/constants.yml --values <location where Helm charts are
extracted>/config/passwords.yml --values <location where Helm charts are
extracted>/platforms/<cloud platform>.yaml --values <location where Helm charts are
extracted>/dockerimages-values.yml --values <location where Helm charts are
extracted>/addons/ao/ao-config.yml --values <location where Helm charts are
extracted>/documentum-resources-values-<config>.yaml --values <location where Helm
charts are extracted>/documentum-components.yaml -n ao

For example: helm upgrade dcmedeploymentdev . --values config/configuration.yml --
values config/constants.yml --values config/passwords.yml --values platforms/
anthos.yaml --values dockerimages-values.yml --values addons/ao/ao-config.yml --
values documentum-resources-values-test-small.yaml --values documentum-
components.yaml -n ao

```

9. Use the following command if Vault is enabled:

```

cd <path_to_ao_helm_chart_directory>

helm upgrade <deployment_name> . --values <location where Helm charts are
extracted>/config/configuration.yml --values <location where Helm charts are
extracted>/config/constants.yml --values <location where Helm charts are
extracted>/config/passwords_vault.yaml --values <location where Helm charts
are extracted>/platforms/<cloud platform>.yaml --values <location where Helm charts
are extracted>/dockerimages-values.yml --values <location where Helm charts are
extracted>/addons/ao/ao-config.yml --values <location where Helm charts are
extracted>/documentum-resources-values-<config>.yaml --values <location where Helm
charts are extracted>/documentum-components.yaml -n ao

For example: helm upgrade dcmedeploymentdev . --values config/configuration.yml --
values config/constants.yml --values config/passwords_vault.yaml --values
platforms/anthos.yaml --values dockerimages-values.yml --values addons/ao/ao-
config.yml --values documentum-resources-values-test-small.yaml --values
documentum-components.yaml -n ao

```

## 5.3 Post upgrade steps

1. OpenText Documentum CM uses OTDS to apply the licenses for all the OpenText Documentum CM components. For more information about procuring the license file and configuring OTDS and license, see *OpenText Documentum Content Management - Cloud Upgrade and Migration Guide (EDCSYCD-AUM)*.
2. Add the OpenText Documentum CM for Engineering OTDS partition back to the **Access to OTAG17** access role post upgrade to enable access for the AO partition users to OpenText Documentum CM mobile.
3. If you notice the primary CS pod status is not changed to the ready state with the following info in `tmp/readiness.log` file.

```
Data-Dictionary installation is still going on
```

Use the following command in the primary CS pod to manually create the file:

```
kubectl exec -it dcs-pg-0 -c dcs-pg -n <namespace> --bash  
[dmadmin@dcs-pg-0 /]$ touch -f opt/dctm/kube/available_locales
```
4. If there are any additional customer-specific client configurations and creation profiles that were created dynamically during Project creation in Active Projects, then you must import:

- Customer specific configuration
- Creation profile of existing Active Projects



**Note:** You must compare the backed up version of client configuration and the latest version of client configuration for differences before you perform the following steps:

- a. In client configuration, select **File > Import Configurations**.
  - b. Select the following options:
    - **Overwrite existing elements**
    - **Do not overwrite auto-naming values in the new configuration**
    - **Do not overwrite cache URLs**
    - **Do not overwrite the mail servers configuration**
  - c. In the Dictionary section, select **Asset PO Number**.
  - d. In the Creation profile section, select the required profiles. For example, `ProjectName-ProjectTitle`
5. If you have added client URLs in client configuration, you must update these URLs with their respective ingresses. Use the **Tools** menu to **Reload** and **Refresh Cache**.

## Chapter 6

# Troubleshooting

### 6.1 Cleaning the OpenText Documentum CM for Engineering deployment



**Note:** OpenText does not recommended to use this command as it deletes all the container data.

To clean up the D2 deployment, run the following script:

```
# Command to delete deployment
helm uninstall <d2deployment> --namespace <your namespace>

# Command to delete pvc
kubectl delete pvc --all --namespace <your namespace>
```

### 6.2 Enable OpenText Documentum CM for Engineering and Connector debug logs

1. Run the following command to edit the serverapps-logging-configmap ConfigMap:

```
kubectl edit configmap serverapps-logging-configmap -n <namespace>
```

2. Add the following properties to ConfigMap:

```
appender.EPFMA.type = RollingFile
appender.EPFMA.name = EPFMA
appender.EPFMA.filePattern = ${logfolderpath}/EPFMA.log.%i.%d{yyyy-MM-dd}
appender.EPFMA.layout.type = PatternLayout
appender.EPFMA.fileName = ${logfolderpath}/EPFMA.log
appender.EPFMA.layout.pattern = %d{ABSOLUTE} %5p [%t] %c - %m%n
appender.EPFMA.policies.type = Policies
appender.EPFMA.policies.time.type = TimeBasedTriggeringPolicy
appender.EPFMA.policies.time.interval = 1
appender.EPFMA.policies.time.modulate = true
appender.EPFMA.policies.size.type = SizeBasedTriggeringPolicy
appender.EPFMA.policies.size.size = 10MB
appender.EPFMA.strategy.type=DefaultRolloverStrategy
appender.EPFMA.strategy.max = 5
logger.EPFMA.name = com.documentum.epfma
logger.EPFMA.level = DEBUG
logger.EPFMA.additivity = false
logger.EPFMA.appenderRef.rolling.ref = EPFMA

logger.D2.name = com.documentum.d2
logger.D2.level = DEBUG
logger.D2.additivity = false
logger.D2.appenderRef.rolling.ref = EPFMA

logger.CDF.name = com.documentum.cdf
logger.CDF.level = DEBUG
logger.CDF.additivity = false
logger.CDF.appenderRef.rolling.ref = EPFMA
```

```

logger.UTILS.name = com.documentum.utils
logger.UTILS.level = DEBUG
logger.UTILS.additivity = false
logger.UTILS.appenderRef.rolling.ref = EPFMA

logger.OT.name = com.opentext
logger.OT.level = DEBUG
logger.OT.additivity = false
logger.OT.appenderRef.rolling.ref = EPFMA

#----- ao connector-----

appender.XCHANGE.type=RollingFile
appender.XCHANGE.name= XCHANGE
appender.XCHANGE.filePattern=${logfolderpath}/aoconnector.log.%i.%d{yyyy-MM-dd}
appender.XCHANGE.layout.type=PatternLayout
appender.XCHANGE.fileName=${logfolderpath}/aoconnector.log
appender.XCHANGE.layout.pattern=%d{ABSOLUTE} %5p [%t] %c - %m%n
appender.XCHANGE.policies.type=Policies
appender.XCHANGE.policies.time.type=TimeBasedTriggeringPolicy
appender.XCHANGE.policies.time.interval=1
appender.XCHANGE.policies.time.modulate=true
appender.XCHANGE.policies.size.type=SizeBasedTriggeringPolicy
appender.XCHANGE.policies.size.size=10MB
appender.XCHANGE.strategy.type=DefaultRolloverStrategy
appender.XCHANGE.strategy.max=5

logger.XCHANGE.name = com.emc.eix.ao
logger.XCHANGE.level = DEBUG
logger.XCHANGE.additivity = false
logger.XCHANGE.appenderRef.rolling.ref = XCHANGE

logger.EIX.name = com.emc.eix
logger.EIX.level = DEBUG
logger.EIX.additivity = false
logger.EIX.appenderRef.rolling.ref = XCHANGE

logger.XCPCLIENT.name = com.emc.xcpclient
logger.XCPCLIENT.level = DEBUG
logger.XCPCLIENT.additivity = false
logger.XCPCLIENT.appenderRef.rolling.ref = XCHANGE

logger.ECS.name = com.ecs
logger.ECS.level = DEBUG
logger.ECS.additivity = false
logger.ECS.appenderRef.rolling.ref = XCHANGE

logger.EIF.name = com.eif
logger.EIF.level = DEBUG
logger.EIF.additivity = false
logger.EIF.appenderRef.rolling.ref = XCHANGE

logger.HTTP.name = com.emc.documentum.http
logger.HTTP.level = DEBUG
logger.HTTP.additivity = false
logger.HTTP.appenderRef.rolling.ref = XCHANGE

```

3. Run the following command to restart the Documentum CM Server pod:

```

# Scale down dcs-pg statefulset
kubectl scale statefulsets dcs-pg --replicas=0 -n <namespace>

# Scale up dcs-pg statefulset
kubectl scale statefulsets dcs-pg --replicas=1 -n <namespace>

```

## 6.3 Resolve issues

1. If the application logs on and off immediately after you log in to Classic View using OTDS login credentials, follow these steps:

- a. Sign in to d2config pod:

```
kubectl exec d2config-0 -it -n <namespace> -- bash
```

- b. Execute the following command to regenerate DFC keystore:

```
cd $CATALINA_HOME/webapps/D2-Config/Utils/d2keystore/
cp /opt/tomcat/CustomConf/dfc.properties .
./D2KeyStoreUtil.sh -u dmadm -p password
./D2KeyStoreUtil.sh -u dmadm -p password -w
```

2. Email notification fails for supported notification features. The following is the error message log:

```
java.lang.Exception: jakarta.mail.Provider: com.sun.mail.imap.IMAPProvider not a
subtype
```

Perform these steps whenever the DCS pod restarts or is recreated:

- a. Sign in to dcs pod.

- b. Navigate to the DmMethods lib directory:

```
# Login into dcs pod
kubectl exec -it dcs-pg-0 -c dcs-pg -it -n <namespace>

# Navigate to DmMethods lib directory
cd /opt/dctm/tomcat10.0.13/webapps/DmMethods/WEB-INF/lib/
```

- c. Rename duplicate javax.mail.jar.

```
mv javax.mail.jar javax.mail.jar_bk
```

- d. Restart the Documentum CM Server Tomcat server.

```
# CS tomcat directory
cd /opt/dctm/tomcat10.0.13/bin

# Stop server
./stopMethodServer.sh

# Start Server
./startMethodServer.sh
```

