# Sitecore Connect for Salesforce CRM 7.0 Container Deployment Guide

A guide to deploying Sitecore Connect for Salesforce CRM to Docker and Azure Kubernetes Service

November 29, 2021

# Table of Contents

# 1. Introduction

Sitecore Connect for Salesforce CRM (SFCRM) enables you to synchronize data between Salesforce CRM and the Sitecore Experience Platform.

This guide shows you how to add the SFCRM connector to Sitecore container installations for Docker and Azure Kubernetes Service.

# 2. Prepare to deploy SFCRM to Sitecore containers

This section explains what you need to deploy the Sitecore Connect for Salesforce CRM (SFCRM) connector to Sitecore containers for Docker and Azure Kubernetes Service.

## 2.1. Requirements

Before you add the SFCRM module for Docker or AKS, you must have the following:

- Docker Desktop installed and running. For instructions on how to set up the Docker environment, see the Containers in Sitecore development documentation.
- if the installation is done on Docker, you must have the Sitecore Docker container files deployed on a local machine . For instructions on how to prepare the Sitecore containers, see the *Installation Guide for Developer Workstation with Containers* on the Sitecore download site.
- If the installation is done on Kubernetes, you must have the Sitecore AKS container files deployed on a local machine . For instructions on how to prepare a Sitecore environment with Kubernetes, see the *Installation Guide for Production Environment with Kubernetes* on the Sitecore download site.
- Access to a Salesforce CRM instance and a user account on Salesforce. This account must at the least have access to read data from Salesforce. If you are moving data to from Sitecore to Salesforce, the account must also have rights to write data to Salesforce.

To prepare for the installation, you must:

- prepare a Salesforce connection string to Sitecore.

## 2.2. Prepare a Salesforce connection string to Sitecore

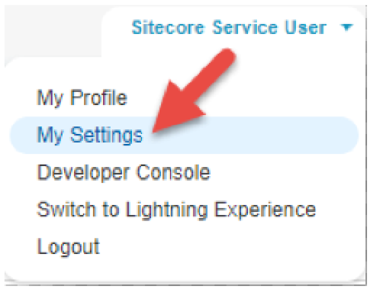
To construct a Salesforce connection string to your Sitecore installation you need to obtain some values from Salesforce. To obtain these values:

1. In Salesforce, make a note of the following values:
    - *User ID* - the ID that Sitecore uses to call the Salesforce API. This user ID does not need administrator rights, but it must have sufficient rights to perform the activities you want to perform from Sitecore. For example:
        - To have Salesforce contacts created in Sitecore, the user must have read-access on contacts and campaigns.
        - To push contact data from Sitecore into Salesforce, the user must have write-access on contacts.

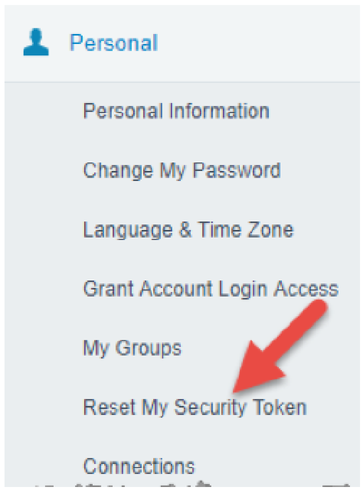
- *Password* - the password for the Salesforce user that Sitecore uses to call the Salesforce API.

**NOTE**
The password must not contain the semicolon (;) character.

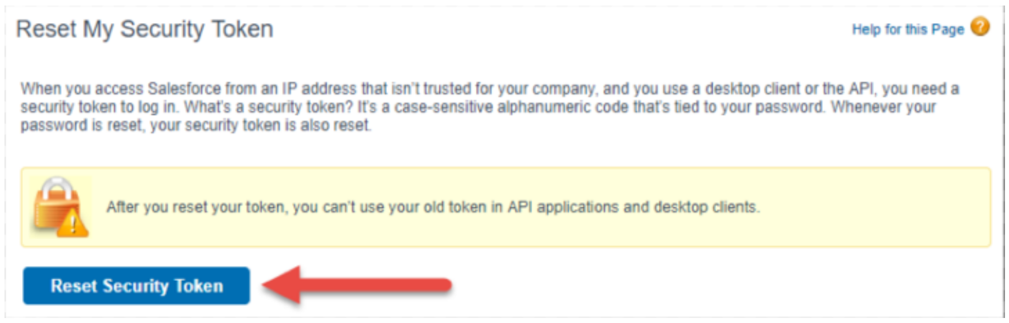
2. To obtain the *security token value*, log in to Salesforce with the user ID.
3. In the top menu, click the user name, then click **My Settings**.



4. In the left menu, click **Personal**, then click **Reset My Security Token**.



5. Read the warning about resetting security tokens. If you are ready to proceed, click **Reset Security Token**.



Reset My Security Token

Help for this Page

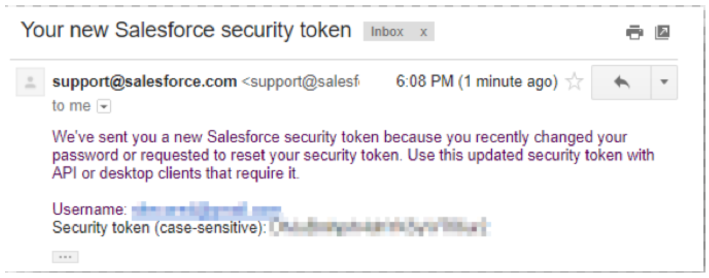
When you access Salesforce from an IP address that isn't trusted for your company, and you use a desktop client or the API, you need a security token to log in. What's a security token? It's a case-sensitive alphanumeric code that's tied to your password. Whenever your password is reset, your security token is also reset.

After you reset your token, you can't use your old token in API applications and desktop clients.

Reset Security Token

Salesforce informs you that the new security token will be emailed to you.

6. In your inbox, find the email message from Salesforce. Locate the *Security token (case-sensitive)* line, and make a note of the new security token value.



Your new Salesforce security token Inbox x

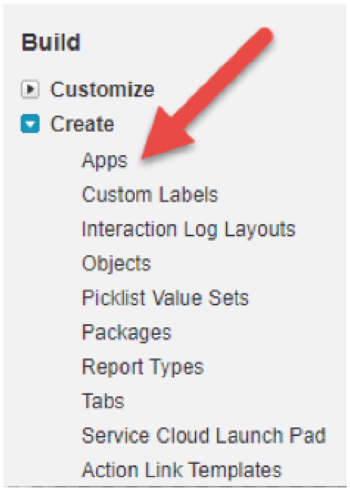
support@salesforce.com <support@salesf 6:08 PM (1 minute ago)

to me

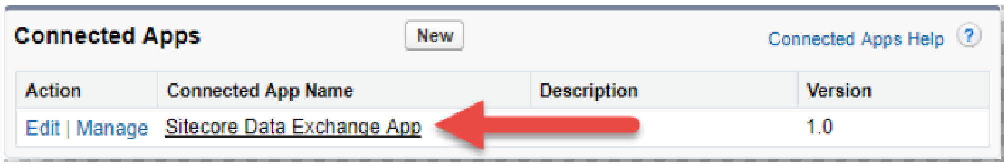
We've sent you a new Salesforce security token because you recently changed your password or requested to reset your security token. Use this updated security token with API or desktop clients that require it.

Username: [redacted]
Security token (case-sensitive): [redacted]

7. To obtain the *Consumer Key* and *Consumer Secret* values, in Salesforce, in the left menu, in the **Build** section, expand **Create** and then click **Apps**.



8. In the **Connected Apps** section, click the name of the connected app you created. For information on how to create a connected app, see the Sitecore Connect for Salesforce CRM 6.0 Installation Guide on the Sitecore download page.



Connected Apps New Connected Apps Help

| Action | Connected App Name | Description | Version |
|---|---|---|---|
| Edit \| Manage | Sitecore Data Exchange App | | 1.0 |

9. In the **API (Enable OAuth Settings)** section, copy the values for the *Consumer Key* and *Consumer Secret*.

10. Use the values to construct a connection string with this format:

```
MYSF_CONNECTIONSTRING=user id=<user id>;
                      password=<password>;
                      client id=<Consumer Key>;
                      secret key=<Consumer Secret>;
                      security token=<security token value>;
```

You will need this connection string when you add the SFCRM module to Docker or Kubernetes.

# 3. Add the SFCRM connector module to Sitecore in Docker

To add Sitecore Connect for Salesforce CRM (SFCRM) in Docker, you must do the following in this order:

- Prepare the installation files.
- Build the Docker images.
- Update the Solr indexes.

## 3.1. Prepare the installation files

To prepare the files you need for the installation:

1. Download the SFCRM container deployment package from the Sitecore Developer Portal. Extract it to your local workstation with the folder structure intact.
2. Go to the folder that you extracted the SFCRM container deployment package to. Go to the folder for the Windows version and topology you are using, for example, `compose\ltsc2019\xp1`.
3. Open the `.env-example` file in an editor. Copy all the variables to the clipboard.
4. Go to the Sitecore Experience Platform (SXP) container deployment folder on your local machine. Go to the folder for the Windows version and topology you are using, for example, `compose\ltsc2019\xp1`.
5. Open the `.env` file in an editor, and paste in the variables from the SFCRM `.env-example` file. Replace the default value for `MYSF_CONNECTIONSTRING` with the connection string you prepared in Prepare to deploy SFCRM to Sitecore containers.
6. From the SFCRM `compose\<version>\<topology>` folder, copy the `docker-compose.override.yml` file to the SXP container deployment `compose\<version>\<topology>` folder (where the `docker.compose.yml` file is).

## 3.2. Build the Docker images

When you have prepared the installation files, you must create Docker files for each role, and build the Docker images.

> **NOTE**
> For more information on image assets, see the documentation on how to Add Sitecore Modules.

To build the images:

1. Go to the Sitecore container deployment folder on your local machine. Go to the folder for the Windows version and topology you are using, for example, `compose/ltsc2019/xp1`. Create a folder and name it `module`.

2. In the `module` folder, create these subfolders:

   - `cm`
   - `xconnect`
   - `xdbsearchworker`

3. In each subfolder, create a new file and name it `Dockerfile`.

4. In the `cm` folder, in the `Dockerfile` file, enter the following instructions:

```
# escape=`
ARG BASE_IMAGE
ARG SFCRM_IMAGE
ARG DEF_IMAGE
ARG TOOLING_IMAGE

FROM ${DEF_IMAGE} as def
FROM ${SFCRM_IMAGE} as sfcrm
FROM ${TOOLING_IMAGE} as tooling
FROM ${BASE_IMAGE} as baseImage

SHELL ["powershell", "-Command", "$ErrorActionPreference = 'Stop'; $ProgressPreference =
'SilentlyContinue';"]

#Add tools from sitecore-docker-tools-assets
COPY --from=tooling \tools\ C:\tools\
FROM baseImage as defsfcrm

# Add DEF module
COPY --from=def \module\cm\content C:\inetpub\wwwroot

# Add SFCRM module
COPY --from=sfcrm \module\cm\content C:\inetpub\wwwroot

#Copy transformation files
COPY --from=sfcrm \module\xdttransform\cm\transforms\ C:\transforms\role

# Add SFCRM connection strings in Sitecore config file
RUN C:\tools\scripts\Invoke-XdtTransform.ps1 -Path C:\inetpub\wwwroot -XdtPath
C:\transforms\role

#Enable loadUserProfile to allow CMS to connect to salesforce
RUN C:\windows\System32\inetsrv\appcmd.exe set config -section:system.applicationHost/
applicationPools /applicationPoolDefaults.processModel.loadUserProfile:"true" /
commit:apphost;
```

5. In the `xconnect` folder, in the `Dockerfile` file, enter the following instructions:

```
# escape=`
ARG BASE_IMAGE
ARG SFCRM_IMAGE
ARG TOOLING_IMAGE

FROM ${SFCRM_IMAGE} as sfcrm
FROM ${TOOLING_IMAGE} as tooling
FROM ${BASE_IMAGE} as baseImage

SHELL ["powershell", "-Command", "$ErrorActionPreference = 'Stop'; $ProgressPreference =
'SilentlyContinue';"]

#Add tools from sitecore-docker-tools-assets
COPY --from=tooling \tools\ C:\tools\

# Copy models file into index worker
COPY --from=sfcrm \module\xconnect\content C:\inetpub\wwwroot

#Copy transformation files
COPY --from=sfcrm \module\xdttransform\xconnect\transforms\ C:\transforms\role

# Update value for IndexPIISensitive and IndexAnonymousContactData in IndexerSetting
RUN C:\tools\scripts\Invoke-XdtTransform.ps1 -Path C:\inetpub\wwwroot -XdtPath
C:\transforms\role
```

6. In the `xdbsearchworker` folder, in the `Dockerfile` file, enter the following instructions:

```
# escape=`
ARG BASE_IMAGE
ARG SFCRM_IMAGE
ARG TOOLING_IMAGE

FROM ${SFCRM_IMAGE} as sfcrm
FROM ${TOOLING_IMAGE} as tooling
FROM ${BASE_IMAGE} as baseImage

SHELL ["powershell", "-Command", "$ErrorActionPreference = 'Stop'; $ProgressPreference =
'SilentlyContinue';"]

#Add tools from sitecore-docker-tools-assets
COPY --from=tooling \tools\ C:\tools\

# Copy models file into index worker
COPY --from=sfcrm \module\xdbsearchworker\content C:\service\

#Copy transformation files
COPY --from=sfcrm \module\xdttransform\xdbsearchworker\transforms\ C:\transforms\role

# Update value for IndexPIISensitive and IndexAnonymousContactData in IndexerSetting
RUN C:\tools\scripts\Invoke-XdtTransform.ps1 -Path C:\service -XdtPath C:\transforms\role
```

7. In the `compose\<version>\<topology>\docker-compose.override.yml` file, add build instructions for each role. If you are using, for example, the XP0 topology, the file will look like this:

```
services:
  cm:
    image: sitecore-sfcrm-xp0-cm:${SITECORE_VERSION}
    build:
      context: ./module
      dockerfile: ./cm/Dockerfile
      args:
        BASE_IMAGE: ${SITECORE_DOCKER_REGISTRY}sitecore-xp0-cm:${SITECORE_VERSION}
        DEF_IMAGE: ${SITECORE_DOCKER_REGISTRY}modules/sitecore-def-xp0-assets:$
{DEF_VERSION}
        SFCRM_IMAGE: ${SITECORE_DOCKER_REGISTRY}modules/sitecore-sfcrm-xp0-assets:$
{SFCRM_VERSION}
        TOOLING_IMAGE: ${SITECORE_TOOLS_REGISTRY}sitecore-docker-tools-assets:$
{TOOLS_VERSION}
        environment:
          Sitecore_ConnectionStrings_mysf: ${MYSF_CONNECTIONSTRING}
xdbsearchworker:
  image: sitecore-sfcrm-xp0-xdbsearchworker:${SITECORE_VERSION}
  build:
    context: ./module
    dockerfile: ./xdbsearchworker/Dockerfile
    args:
      BASE_IMAGE: ${SITECORE_DOCKER_REGISTRY}sitecore-xp0-xdbsearchworker:$
{SITECORE_VERSION}
      SFCRM_IMAGE: ${SITECORE_DOCKER_REGISTRY}modules/sitecore-sfcrm-xp0-assets:$
{SFCRM_VERSION}
      TOOLING_IMAGE: ${SITECORE_TOOLS_REGISTRY}sitecore-docker-tools-assets:$
{TOOLS_VERSION}
xconnect:
  image: sitecore-sfcrm-xp0-xconnect:${SITECORE_VERSION}
  build:
    context: ./module
    dockerfile: ./xconnect/Dockerfile
    args:
      BASE_IMAGE: ${SITECORE_DOCKER_REGISTRY}sitecore-xp0-xconnect:${SITECORE_VERSION}
      SFCRM_IMAGE: ${SITECORE_DOCKER_REGISTRY}modules/sitecore-sfcrm-xp0-assets:$
{SFCRM_VERSION}
      TOOLING_IMAGE: ${SITECORE_TOOLS_REGISTRY}sitecore-docker-tools-assets:$
{TOOLS_VERSION}
```

8. In the `compose\<version>\<topology>\.env` file, add the asset image version. For example:

```
DEF_VERSION=<image version for your topology>
SFCRM_VERSION=<image version for your topology>
SITECORE_TOOLS_REGISTRY=scr.sitecore.com/tools/
TOOLS_VERSION=<image version for your topology>
```

> **NOTE**
> You can find the image version in the Sitecore Docker Images repository.

9. In the Windows console, go to the folder containing the `docker-compose.override.yml` file. Run the command `docker-compose build`.
10. After the build has completed, run the command `docker-compose up -d`.
11. When the compose command has completed, update your search indexes.

# 4. Add the SFCRM connector module to Sitecore in Azure Kubernetes Service

To add the Sitecore Connect for Salesforce CRM (SFCRM) connector in Azure Kubernetes Service (AKS) you must do the following in this order:

- Build the SFCRM images and push them to Azure.
- Prepare files and folders for deployment.
- Deploy the containers using *kubectl* commands.
- Update your Solr indexes.

## 4.1. Build images and push them to Azure

To build the images for SFCRM and push them to Azure:

1. Build the images for SFCRM as explained in Add the SFCRM connector module to Sitecore in Docker.

   **NOTE**
   The Kubernetes deployment requires an `mssql-init` image. You must ensure that you include `mssql-init` in your `docker-compose-override.yml` file.

2. Open the Windows console, and use the `docker tag` command to tag the images. For example:

```
docker tag sitecore-sfcrm-xp1-cm:10.1.0.005207.643-10.0.17763.1757-ltsc2019 $registry/
sitecore-sfcrm-xp1-cm:<tag version>
```

3. In the console, use the `docker push` command to push the images to your Azure registry. For example:

```
docker push $registry/sitecore-sfcrm-xp1-cm:<tag version>
```

## 4.2. Prepare files and folders for deployment

To prepare files and folders in your installation for deployment:

1. Download the Sitecore SFCRM container deployment package from the Sitecore download page and extract it to a folder on your local workstation.

2. Open the folder that you extracted the Sitecore SFCRM container deployment package to.
3. Navigate to the `k8s\<version>\<topology>` folder, for example, `k8s\ltsc2019\xp1`. Copy the `overrides` subfolder to the Sitecore Experience Platform (SXP) container deployment package folder `k8s\<version>` (on the same level as the *xp1* folder).
4. In the SXP container deployment package, in each of the `overrides`, `overrides\xp1\init`, and `overrides\xp1\secrets` folders, locate the `kustomization.yaml` file. In each file, update the `bases` parameter with the appropriate folder names for your installation, for example, `../../xp1`.

   **NOTE**
   The `bases` parameter contains the placement of the original Sitecore container deployment files that the `kustomization.yaml` files override.

5. In each of the `kustomization.yaml` files, in the `images:` section, update the `newName` and `newTag` parameters with the values for the `mssql-init`, `cm`, `xdbcollection`, `xdbsearch` and `xdbsearchworker` images you pushed to the Azure Registry.
6. In the `overrides\xp1\secrets` folder, in the `sitecore-salesforce-crm-connection-string.txt` file , replace the content with the connection string you prepared in Prepare to deploy SFCRM to Sitecore containers.

## 4.3. Deploy the containers

To deploy the containers and the necessary Kubernetes components:

1. Prepare the AKS cluster configuration and deploy the ingress controller. For information on how to do this, see the *Installation Guide for Production Environment for Kubernetes*, which is available on the Sitecore download page.
2. Open the Windows console, and navigate to the folder containing the `xp1` and `overrides` folders.
3. Deploy the secrets. Use this command:

```
kubectl apply -k ./overrides/xp1/secrets/
```

4. Run the `external` folder. Use this command:

```
kubectl apply -k ./xp1/external/
```

5. Wait for all containers to have the status *Ok/Running*. You can check the status with this command:

```
kubectl get pods -o wide
```

6. Run the `init` folder. Use this command:

```
kubectl apply -k ./overrides/xp1/init/
```

7. Wait for all containers to have the status *Completed*. You can check the status with this command:

   ```
   kubectl get pods
   ```

8. To create persistent volumes, run this command:

   ```
   kubectl apply -f ./xp1/volumes/azurefile
   ```

9. Run the Sitecore containers with the SFCRM changes. Use this command:

   ```
   kubectl apply -k ./overrides/xp1/
   ```

10. Wait for all containers to have the status *Ok/Running*. You can check the status with the `kubectl get pods` command.

11. Update the local host file. For information on how to do this, see the *Installation Guide for Production Environment for Kubernetes*, which is available on the Sitecore download page.

12. Update your search indexes.

# 5. Rebuild the search indexes

When you have deployed the containers, you must rebuild your search indexes.

To rebuild the indexes:

1. Browse to your Sitecore URL, for example, `https://xp1cm.localhost/`. Open the control panel.
2. In the **Indexing** section, click **Populate Solr Managed Schema**.
3. In the **Schema Populate** dialog box, click **Select All**, then click **Populate**. Wait for the process to finish.
4. On the Control Panel, in the **Indexing** section, click **Indexing Manager**. In the **Indexing Manager** dialog, click **Select All**.
5. Click **Rebuild**. When the indexes have been rebuilt, click **Close**.
6. Open the Content Editor with Master as the content database.
7. In the content tree, navigate to */sitecore/system/DataExchange*. On the **Folder** tab, verify that the **Empty Data Exchange Tenant** and **Connect for Salesforce Tenant** buttons are available.

# 6. Upgrade from SFCRM 6.0 to SFCRM 7.0 on Docker

This section explains how to upgrade from Sitecore Connect for Salesforce CRM (SFCRM) 6.0 to SFCRM 7.0 on a container platform.

## 6.1. Requirements

Before you perform the upgrade for SFCRM container 6.0 to SFCRM container 7.0, you must have the following:

- Sitecore Experience Platform (SXP) 10.1 deployed on Docker
- SFCRM 6.0 deployed on Docker
- An up to date back up of the current mssql database

## 6.2. Upgrade process

You must upgrade your SXP installation to 10.2 and your SFCRM installation to 7.0 at the same time.

To do so, you must:

1. Build new docker images for SXP 10.2 and SFCRM 7.0.
2. Build an mssql-upgrade image
3. Perform the upgrade

### 6.2.1. Build new Docker images

To build the SXP 10.2 and SFCRM 7.0 Docker images:

1. Download the Sitecore Experience Platform container deployment 10.2 package from the Sitecore download page. Extract it to a new folder on your local workstation with the folder structure intact. Name the new folder, for example, `SFCRM7`.
2. Download the SFCRM container deployment 7.0 package from the Sitecore download page. Extract it to your local workstation with the folder structure intact. Copy the files in the `sfcrm\compose\<windows version>\<topology>` folder and paste them into the `\compose\<windows version>\<topology>` folder in the SXP 10.2 deployment structure.
3. Navigate to your Sitecore 10.1 container deployment folder. Copy the databases from the `mssql-data` folder and paste them in the Sitecore 10.2 container deployment `mssql-data` folder.
4. To deploy Sitecore 10.2, open a PowerShell window with administrator rights, navigate to the SC 10.2 Container deployment folder, and run the following commands:

```
docker-compose build
docker-compose up
```

5. Verify that Sitecore Container 10.2 is up and running successfully.

### 6.2.2. Build the mssql-upgrade image

You use a custom mssql-upgrade image to upgrade a Sitecore solution that has SFCRM installed. To build a custom mssql-upgrade image, you must download the latest Sitecore XP mssql-upgrade image from the container registry and create a custom Dockerfile on top of it.

1. From the **Resource files for Modules 1.0.0** section on the Sitecore download page, download the SFCRM Connect Upgrade resources 1.0.0 package. Extract it to a folder on your local machine.
2. From the upgrade resources, copy the `Salesforce CRM Connect Upgrade resources 1.0.0\6.0.0\Data` folder and paste it into the upgrade folder for the Windows version and topology you are using, for example, `ltsc2019\upgrade\xp1`.
3. In the `Salesforce CRM Upgrade resources 1.0.0\6.0.0` folder, create a docker file and name it `Dockerfile`. In the file, add instructions to point its base image to the 10.2 mssql-upgrade image. The file will look, for example, like this:

```
ARG BASE_IMAGE= ${SITECORE_DOCKER_REGISTRY} /sxp/sitecore-xp1-mssql-upgrade : tag
FROM ${BASE_IMAGE}
SHELL ["powershell", "-Command", "$ErrorActionPreference = 'Stop'; $ProgressPreference =
'SilentlyContinue';"]
# Add SFCRM module
COPY <Data folder local path> "C:\data\ResourceItems\10.1\modules"
```

**NOTE**
In the Dockerfile, ensure that ARG BASE_IMAGE parameter points to the mssql-upgrade image, and that the Data folder local path is set to the 6.0.0\Data folder in the folder you created in Step 1.

4. To build the mssql-upgrade image, open a PowerShell window, navigate to the folder where you placed the Dockerfile, and run the following command:

```
docker build . -t "<imageName>:<available port number>”
```

5. Verify that Docker has created an image with the name you specified.

### 6.2.3. Perform upgrade process

To perform the upgrade:

1. On your local machine, in a PowerShell window, navigate to the Sitecore Container Deployment 10.2 folder. Navigate to the upgrade folder for the Windows version and topology you are using, for example, `ltsc2019\upgrade\xp1`.
2. In the topology folder, run the `compose-init.ps1 script`. This script updates the environment configuration file with the appropriate values for all the environment variables including the SQL username, SQL password, SQL Server address, and the Sitecore license file.

**NOTE**
For more information about running the script to prepare for the deployment, see the Installation Guide for Developer Workstation with Containers on the Sitecore download page.

3. In the `docker-compose.upgrade.yml` file, update the `image` setting with the mssql-upgrade image you created previously.
4. Open a new PowerShell window with administrator rights. Navigate to the upgrade directory.
5. To perform the upgrade, run this command:

```
docker-compose.exe -f .\docker-compose.upgrade.yml --env-file .\upgrade.env up
```

6. To check the status of the upgrade, run this command:

```
docker-compose.exe -f .\docker-compose.upgrade.yml --env-file .\upgrade.env ps
```

7. When the upgrade process is completed, you can clean up your environment.
   If you ran the upgrade container in Docker Compose, from the Docker Compose folder for the topology that you upgraded, for example, `sitecore-xp1`, run the following PowerShell cmdlet:

```
docker-compose.exe -f .\docker-compose.upgrade.yml --env-file .\upgrade.env down
```

# 7. Upgrade from SFCRM 6.0 to SFCRM 7.0 on Kubernetes

This section explains how you upgrade Sitecore Connect for Salesforce CRM (SFCRM) container in AKS from version 6.0 to 7.0.

SFCRM 7.0 is compatible with Sitecore Experience Platform (SXP) 10.2.

## 7.1. Requirements

Before you perform the upgrade for SFCRM 6.0 to SFCRM 7.0, you must:

- Have a Sitecore AKS 10.1 deployment with SFCRM 6.0
- Back up the 10.1 database.

## 7.2. Build and push the mssql-upgrade image

To upgrade SFCRM you must build an mssql-upgrade image. To do so:

1. Build the images for SFCRM as explained in the *Build the mssql-upgrade image* section in Upgrade from SFCRM 6.0 to SFCRM 7.0 on Docker.
2. In a PowerShell window, navigate to the folder containing the Dockerfile and build and tag the image. For example:

```
docker build . -t sitecore-sfcrm-xp1-cm:<imageVersionTag> $registry/sitecore-sfcrm-xp1-
cm:<newTag>
```

3. Check that Docker has created the image.
4. Push the created image to your Azure registry. For example:

```
docker push $registry/sitecore-dcrm-xp1-cm:<newTag>
```

## 7.3. Perform upgrade process

To upgrade SXP to version 10.2 and SFCRM to version 7.0:

1. Download and extract Sitecore 10.2 Sitecore Container Deployment Package from the Sitecore download page. Extract it to your local workstation with the folder structure intact.
2. Navigate to the upgrade folder for the Windows version and topology you are using, for example, `k8s\ltsc2019\upgrade\xp1`. In the `kustomization.yaml` file, update the images section with `newName` and `newTag` attributes of the custom mssql-upgrade image you created and pushed previously.
3. In the `configuration` folder, update the secrets files. For more information on the secrets files, please refer to the *Installation Guide for Production Environment with Kubernetes* guide available on the Sitecore download page.
4. Download the SFCRM container deployment 7.0 package from the Sitecore download page. Extract it to your local workstation with the folder structure intact. Copy the `\k8s\<windows version>\overrides` folder and paste it into the `\k8s\<windows version>\` folder in the SXP 10.2 deployment structure.
5. In the `secrets` folder in the SXP 10.2 structure, for example, `k8s\ltsc2019\overrides\xp1\secrets`, update the `sitecore-salesforce-crm-connection-string.txt` secrets file.
6. Log in to the Azure CLI and set a subscription.

   ```
   az login
   az account set --subscription "Your Subscription"
   ```

7. Get the credentials for the Kubernetes cluster that was created with the AKS cluster.

   ```
   az aks get-credentials --resource-group <10.2 resource group>--name <10.2 cluster>
   ```

8. To deploy the Sitecore upgrade job, go to the folder where the updated files are, for example `k8s\ltsc2019\upgrade\xp1`, and run this command:

   ```
   kubectl apply -k .\
   ```

9. To check if the job has completed, run this command:

   ```
   kubectl get pod
   ```

10. When the upgrade process is completed, you can delete the Kubernetes upgrade job and upgrade secrets. In the console, go to the folder you used in Step 2. and run these commands:

    ```
    kubectl delete -f .\
    kubectl delete -k .\
    ```

**NOTE**
Upgrading with a custom mssql-image includes cleaning up SFCRM items from the database since SFCRM 7.0 has moved to using resources files. This requires rebuilding and upgrading the Sitecore role instances cm, xconnect, and xdbsearchworker.

For detailed instructions on how to build and deploy the Sitecore role instances please refer to the Sitecore *Upgrade Container Deployment Guide* on the Sitecore download page.