*Rev: September 10, 2018*

# Sitecore Experience Commerce Installation Guide for the Azure App Service

## Sitecore Experience Commerce 9.0

*Installation guide for administrators and developers*

sitecore®
Own the experience™

## Table of Contents

# Chapter 1

## Getting started

This guide describes how to install Sitecore Experience Commerce™ (XC) 9.0 in an Azure App Service configuration.

The Microsoft Azure App Service® is a cloud-computing platform that provides a rich variety of services to help you create and run scalable applications without high upfront infrastructure investments.

For details on installing Sitecore XC in a single-server on-premises configuration, see the *Sitecore Experience Commerce Installation Guide*.

The document contains the following chapters:

- **Chapter 1 – Getting started**
  An introduction to the Sitecore Experience Commerce (XC) 9.0 solution and an overview of the installation process.
- **Chapter 2 – Preparing your environment**
  Describes tasks you must complete before deploying Sitecore XC to the Azure App Service.
- **Chapter 3 – Downloading the Web Deploy Packages**
  Describes the Sitecore XC deployment script and how to customize your configuration for Sitecore XC installation in your environment.
- **Chapter 4 – Configuring the Azure Resource Manager (ARM) templates**
  Describes the Sitecore XC deployment script and how to customize your configuration for Sitecore XC installation in your environment.
- **Chapter 5 – Deploying the solution to Azure**
  Describes the Sitecore XC deployment script and how to customize your configuration for Sitecore XC installation in your environment.
- **Chapter 6 – Post-installation steps**
  Describes the necessary actions after you deploy the Sitecore XC solution to Azure.

## 1.1 Sitecore Experience Commerce solution

Sitecore Experience Commerce (XC) is an e-commerce solution, built on the Sitecore Experience Platform (Sitecore XP).

The Sitecore XC solution provides a core framework for rapidly delivering commerce functionality through the following components:

- **Commerce Engine**
  An extensible commerce core framework, hosting commerce services such as Cart, Order, Pricing, Promotions, Catalogs, and Inventory. The Commerce Engine includes a pluggable framework for extending the engine to modify or add to existing functionality.

- **Commerce Business Tools**
  A set of rich business tools for merchandisers and customer service representatives. The business tools are built on the Angular framework, and can also be extended using the same pluggable framework.

- **Sitecore Experience Accelerator (SXA) Storefront**
  A sample storefront website that is integrated with the Commerce Engine. You can use the SXA Storefront as a starting point to building a customized storefront.

For more information about Sitecore XP, refer to the Sitecore Documentation site – http://doc.sitecore.net.

For assistance, or to report any discrepancies between this document and the product, please contact http://support.sitecore.net/helpdesk/.

## 1.2 Overview of the installation process

The installation process for deploying your Sitecore XC solution to Azure has several tasks.

To install an instance of Sitecore Experience Commerce to the Azure App Service:

1. Prepare your environment by completing the following prerequisite tasks:
    - Install the Sitecore Azure Toolkit.
    - Obtain a valid client certificate.
    - Set up Azure storage.
    - Create a Braintree sandbox account (for payment functionality).
2. Obtain the Web Deploy Packages (WDPs) for Sitecore XP, Sitecore XC, and Sitecore Experience Accelerator (SXA), then upload the WDPs to Azure storage.
3. Configure the Azure Resource Manager (ARM) templates and specify the values for your environment in the ARM parameters file.
4. Deploy the Sitecore XC solution to Azure.
5. Complete the post-installation set-up steps to finish the installation.

# Chapter 2

## Preparing your environment

This chapter outlines the tasks you must complete before installing Sitecore XC 9.0 on the Azure App Service.

This chapter includes the following sections:

- Install the Sitecore Azure Toolkit
- Set up Azure storage
- Obtain a valid client certificate
- Create a Braintree account

## 2.1 Install the Sitecore Azure Toolkit

The Sitecore Azure Toolkit contains the tools and resources necessary to prepare and deploy Sitecore solutions to the Microsoft Azure App Service®.

Download and install the Sitecore Azure Toolkit according to the instructions provided here.

## 2.2 Set up Azure storage

The WDPs containing the application code and resources must be accessible to the Azure Resource Manager over the Internet. You can use a Microsoft Azure® storage account to host the Sitecore WDPs and ARM templates for your deployment.

Follow the instructions on the Microsoft Azure Storage site to create a storage account (with type *Blob storage*) and use the Azure Storage Explorer to create blob containers to store your Sitecore WDPs and ARM templates.

## 2.3 Obtain a valid client certificate

The Sitecore XP is designed to be secure by default. This means that the Sitecore roles use the Secure Sockets Layer (SSL) with client certificate validation to communicate with each other.

In addition, the Sitecore Identity Server certificate must contain information to enable a key signing service for its token creation and validation services.

The client certificate validation requires you to set up a client certificate as part of the Sitecore installation process on Azure. You can choose to generate a self-signed certificate to meet this installation requirement or obtain one from a certificate authority.

Before you start your Sitecore XC deployment, you must obtain or generate an authentication certificate and store it in PKCS #12 format (.pfx). For developer environments, you can generate a self-signed certificate using PowerShell. For productions environments, you must obtain a certificate from a certified authority because of potential security concerns.

For more information on client certificate for Sitecore XP deployments, see the information here.

### 2.3.1 Generate the certificate

The Sitecore XC Commerce Engine SDK contains a script for generating a self-signed certificate.

Sitecore recommends that you modify the existing certificate generation script and save it with a new name (to preserve a record of the factory default script). After you specify values and paths to reflect your own environment you can run the script to generate the certificate.

To generate the certificate:

1. Go to the Sitecore Experience Commerce Download page and download the Sitecore XC solution package for on-premises (using the **Packages for On Premises** link).
2. Extract the files.
3. Locate the `Sitecore.Commerce.Engine.SDK` package and extract the contents.
4. Navigate to the *scripts* folder and open the `New-DevelopmentCertificate.ps1` file in a text editor.
5. Save a copy of the file with a new name (for example, `MyNew-Certificate.ps1`).
6. Specify values for the following parameters, according to your own environment. All other values in the script should remain the same as the specified defaults. .

| Parameter | Description |
|---|---|
| `certificateDnsName` | The host name associated with the certificate. |
| `certificatePassword` | The password for the certificate private key. |
| `certificateStore` | The store in which to store the new certificate. |
| `certificateFriendlyName` | A friendly name for the certificate. |
| `certificateOutputDirectory` | Output directory for the certificate. |

Descriptions of all parameters in the certificate generation script are available here.

7. Add the `-Provider` parameter to the new certificate code block (starting on line 14):

```
$certificate = New-SelfSignedCertificate `
     -Subject $certificateDnsName `
     -DnsName $certificateDnsName `
     -KeyAlgorithm RSA `
     -KeyLength 2048 `
     -NotBefore (Get-Date) `
     -NotAfter (Get-Date).AddYears(1) `
     -CertStoreLocation $certificateStore `
     -FriendlyName $certificateFriendlyName `
     -HashAlgorithm SHA256 `
```

```
            -KeyUsage DigitalSignature, KeyEncipherment, DataEncipherment `
            -Provider "Microsoft Enhanced RSA and AES Cryptographic Provider" `
            -TextExtension @("2.5.29.37={text}1.3.6.1.5.5.7.3.1")
```

8. Save the file and run the script in an elevated PowerShell console.

### 2.3.2 Convert the PFX file to a Base64 string

You must convert the PFX certificate you generated to a Base64-encoded string so that you can specify the value in the ARM template when you deploy your solution to Azure.

To convert the PFX certificate:

1. In a PowerShell session, run the following command:

```
$fileContentBytes = get-content '<path-to-certificate.pfx>' -Encoding Byte
[System.Convert]::ToBase64String([System.IO.File]::ReadAllBytes(“path to pfx file”))
```

2. Note the location of the output file. You need to specify the Base64 string in the `azuredeploy.parameters.json` file.

   **Note**
   The `authCertificateBlob` parameter in the `azuredeploy.parameters.json` file requires the full Base64 string from step 1, and not the path to the certificate.

## 2.4 Create a Braintree account

You need a Braintree sandbox account to enable web payment functionality through the Commerce Engine.

To create a Braintree sandbox account:

1. Open a browser and go to https://www.braintreepayments.com/sandbox.
2. Enter the required information in the **Sign up for the sandbox** pane, and click **Try the sandbox**.
3. Follow the instructions to activate and log in to your Braintree sandbox account.

When you receive your Braintree Sandbox account information, note the **Merchant ID**, **Public Key**, and **Private Key** values. You need to specify this information in the ARM template parameters file for your deployment.

Chapter 3

# Obtaining the Web Deploy Packages

Sitecore creates WDPs from standard Sitecore deployment packages during the packaging process that have been modified to run on Azure during the packaging process. Use these WDPs for a standard original deployment of Sitecore.

You can download pre-built WDPs from the Sitecore Download page. The WDP(s) come in a single .zip file and are grouped per version and per topology. After you download the WDPs, you must extract them from the .zip file before the Sitecore Azure Toolkit can use them.

This chapter contains the following sections:

- List of required WDPs
- Download the WDPs
- Upload the WDPs to Azure storage

## 3.1 List of required WDPs

The following table lists the required WDPs for your Sitecore XC 9.0 solution to Azure. All WDP files have an extension of `.scwdp.zip`.

**Note**
For the Sitecore XC 9.0. Initial Release and 9.0.1, only the Sitecore XP Scaled topology is supported.

| Product | Web Deploy Packages |
|---|---|
| Sitecore XP 9.0.1<br>Available from the Sitecore Experience Platform Download page. | Sitecore 9.0.1 rev.xxxxx (Cloud)_cd<br>Sitecore 9.0.1 rev.xxxxx (Cloud)_cm<br>Sitecore 9.0.1 rev.xxxxx (Cloud)_dds<br>Sitecore 9.0.1 rev.xxxxx (Cloud)_prc<br>Sitecore 9.0.1 rev.xxxxx (Cloud)_rep<br>Sitecore 9.0.1 rev.xxxxx (Cloud)_xp1collection<br>Sitecore 9.0.1 rev.xxxxx (Cloud)_xp1collectionsearch<br>Sitecore 9.0.1 rev.xxxxx (Cloud)_xp1marketingautomation<br>Sitecore 9.0.1 rev.xxxxx (Cloud)_xp1marketingautomationreporting<br>Sitecore 9.0.1 rev.xxxxx (Cloud)_xp1referencedata |
| Sitecore XC 9.0 Update-1<br>Available from the Sitecore Experience Commerce Download page. | Sitecore Commerce Connect Core.xx.x.xxx<br>Sitecore Commerce Connect Schema Definitions for IndexWorker. .xx.x.xxx<br>Sitecore Commerce Connect Definitions for xConnect.xx.x.xxx<br>Sitecore.BizFx.x.x.xxx<br>Sitecore.IdentityServer.x.x.x<br>Sitecore.Commerce.Engine.x.x.xxxx<br>Adventure Works Images<br>Sitecore.Commerce.Engine.Connect.CD.x.x.xxx<br>Sitecore.Commerce.Engine.Connect.CM.x.x.xxx<br>Sitecore Commerce Experience Accelerator.x.x.xxxx<br>Sitecore Commerce Experience Accelerator Habitat Catalog.x.x.xxxx<br>Sitecore Commerce Experience Accelerator Storefront.x.x.xxxx<br>Sitecore Commerce Experience Accelerator Storefront Themes.x.x.xxxxs |
| SXA 1.6<br>Available from the Sitecore Experience Accelerator Download page. | Sitecore Experience Accelerator 1.6 rev.xxxxx for 9.0<br>Sitecore Experience Accelerator 1.6 rev. xxxxx for 9.0 CD<br>Sitecore PowerShell Extensions-4.7.2 for Sitecore 9 |
| Sitecore Azure Toolkit Bootloader<br>Available from the *Addons* folder in the Sitecore Azure Toolkit zip file. | Sitecore.Cloud.Integration.Bootload.wdp.zip |

## 3.2 Download the WDPs

For each set of WDPs you need, perform the following steps:

1. Go to the appropriate Sitecore Download page (as listed in the table above).
2. To download the zipped file of WDPs, scroll to the *Download options for Azure AppService* section and click on the appropriate link for your Sitecore product.
3. When you have downloaded the zip file, unzip the file to unpack the individual WDP files.

## 3.3 Upload the WDPs to Azure storage

Your Sitecore WDPs must be accessible over the Internet to the Azure Resource Manager. These instructions assume that you are using Azure Storage to host the WDPs.

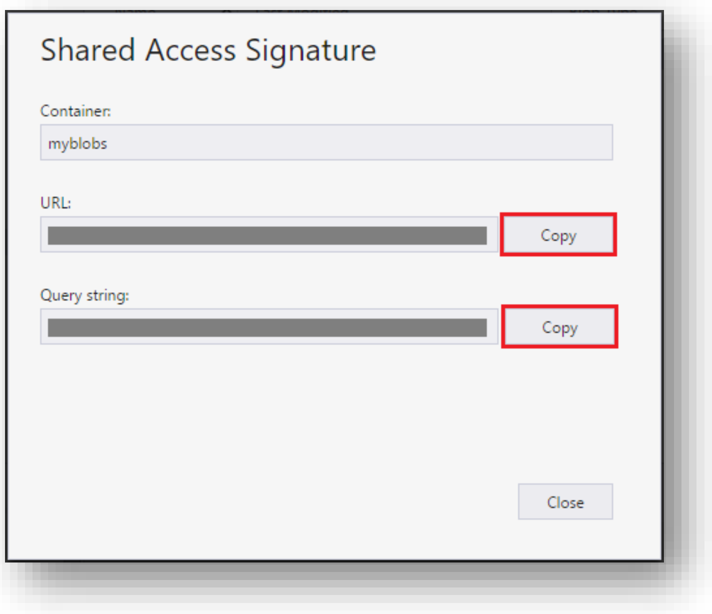
When you upload your WDPs to an Azure storage container, you create a Shared Access Signature (SAS) token for the storage container. The SAS allows temporary access to the WDPs during the deployment process.

For each set of downloaded WDPs, perform the following steps:

1. Open **Azure Storage Explorer** and connect to a Microsoft Azure storage account.
2. Upload the WDPs for your configuration to the container that you created earlier.
3. Create a Shared Access Signature (SAS) token for the storage container. When setting the *Start time* and *Expiry time* for the SAS token, make sure that you allow enough time to guarantee access during the scheduled deployment.
4. After you have created the SAS, copy and make a note of the following values in the **Shared Access Signature** dialog:

   **URL**: location to access the WDPs in Azure storage

   **Query string**: contains the SAS token.



Shared Access Signature

Container: myblobs

URL: [redacted]

Query string: [redacted]

**Note**
The URL and SAS token are required for later use in the ARM templates, so for each WDP that you upload, take a note of its URL and append the SAS token to it.

# Chapter 4

## Configuring the ARM template

ARM templates are distributed on Github and provide a description of Sitecore environments hosted on Microsoft Azure App Service. The templates are compatible with the WDPs available on dev.sitecore.net or those produced by the Sitecore Azure Toolkit.

The structure and organization of the templates are nested. The main template (`azuredeploy.json`) references a parameters file (`azuredeploy.parameters.json`), an infrastructure template and an application template. The deployment responsibilities are then split respectively between templates to setup resources and templates to install the Sitecore application.

This chapter contains the following sections:

- Obtain the ARM template and parameters file
- Customize the Azure environment

## 4.1 Obtain the ARM template and parameters file

When you deploy to Azure, you must specify an ARM template (`azuredeploy.json`), which contains the instructions for the Azure deployment, and include environment-specific parameter values in an Azure environment file (`azuredeploy.parameters.json`).

You can specify the ARM template for your Azure deployment by specifying its location (in this case, the ARM templates are located on Github).

For this scenario, you use the Sitecore XP 9.0.1 ARM template to deploy all required software and modules to the Azure App Service.

To obtain the ARM template and parameters file:

1. Go to the Sitecore GitHub repository and navigate to the *Sitecore 9.0.1\XP* folder.
2. In the *Sitecore 9.0.1\XP* folder, click on the `azuredeploy.json` file to open it.

   The file opens in the browser window.
3. In the `azuredeploy.json` file, click on **Raw** in the document header row.

   The file opens in a new browser window.
4. Copy the URL directly from the address bar and make a note of the information for use in the Azure environment file.
5. Return to the *Sitecore 9.0.1\XP folder* and download the `azuredeploy.parameters.json` file.

**Note**
You only need to download the Sitecore 9.0.1 XP version of the `azuredeploy.parameters.json` file. This environment configuration file specifies parameters for all of the components of your Azure deployment (including Sitecore XC and SXA Storefront).

## 4.2 Customize the Azure environment template

You specify values for your deployment environment in the Sitecore XP `azuredeploy.parameters.json` file. The ARM template uses the parameters file to obtain environment-specific values during deployment.

By adding modules to the `azuredeploy.parameters.json` file, you can instruct the ARM template to deploy Sitecore XC and the SXA Storefront as part of the Sitecore XP deployment to Azure.

To customize the Azure environment template:

1. Open the `azuredeploy.parameters.json` file and specify values for the following parameters.

   These parameters are not Sitecore XC-specific, but are required for the Sitecore XP deployment to Azure:

| Parameter | Description |
|---|---|
| `deploymentID` | The unique identifier for the deployment. This string must be all lower-case characters and contain no spaces. |
| `location` | The geographical region of the current deployment. |
| `sitecoreAdminPassword` | Password for the *sitecore\admin* user when Sitecore is deployed. |
| `licenseXml` | Path to the Sitecore license.xml file. |
| `repAuthenticationApiKey` | A unique value, for example, a GUID, which is used to authenticate when communicating from Content Management role to the Reporting Web App. The minimum length required is 32 characters. |
| `sqlServerLogin` | Name of the administrator account for the Azure SQL Server (for core, master, web, and reporting SQL Azure databases). |
| `sqlServerPassword` | Password for the administrator account for the Azure SQL server. |
| `cmMsDeployPackageUrl` | The URL to a Sitecore XM Content Management deployment package (WDP). |
| `cdMsDeployPackageUrl` | The URL for a Sitecore XM Content Delivery WDP. |
| `prcMsDeployPackageUrl` | The URL for a Sitecore XP Processing WDP. |
| `repMsDeployPackageUrl` | The URL for a Sitecore XP Reporting WDP. |
| `xcRefDataMsDeployPackageUrl` | The URL for an xConnect Reference Data service WDP. |
| `xcCollectMsDeployPackageUrl` | The HTTP(s) URL for an xConnect Collection service WDP. |
| `xcSearchMsDeployPackageUrl` | The HTTP(s) URL for an xConnect Collection Search service WDP. |
| `maOpsMsDeployPackageUrl` | The HTTP(s) URL for a Marketing Automation service WDP. |
| `maRepMsDeployPackageUrl` | The HTTP(s) URL for a Marketing Automation Reporting service WDP. |
| `authCertificateBlob` | The Base64-encoded blob of the authentication certificate for the Sitecore XC solution. |
| `authCertificatePassword` | The password for the authentication certificate. |

2. If you are using self-signed certificates, you must add the following parameter to force xConnect to accept self-signed certificates:

```
"allowInvalidClientCertificates": {
    "value": true
```

3. Add the following parameters for the Sitecore XP EXM packages:

```
"exmDdsMsDeployPackageUrl": {
    "value": "<Url to Sitecore EXM DDS package (Ends with _dds.scwdp.zip)>"
 },
 "exmCmMsDeployPackageUrl": {
    "value": "<Url to Sitecore EXM Patch package (Starts with Sitecore.Patch.EXM…)>"
 },
```

4. Add a `modules` block after the last parameter in the file using the following code snippet):

```
"modules": {
  "value": {
    "items": [
       /* add module entries here */
    ]
  }
}
```

5. Add the Commerce module to the modules block to specify the location of the Sitecore XC ARM template, the location of the Sitecore XC WDPs, and Braintree account information (optional).

```
"modules": {
  "value": {
    "items": [
      {
        "name": "Commerce",
        "templateLink": "<URL for the Sitecore XC ARM template>",
        "parameters": {
          "templateLinkAccessToken": "<SAS-token>",
          "commerceConnectMsDeployPackageUrl": "<URL for WDP>",
          "commerceEngineCloudMsDeployPackageUrl": "<URL for WDP>",
          "commerceEngineConnectCdMsDeployPackageUrl": "<URL for WDP>",
          "commerceEngineConnectCmMsDeployPackageUrl": "<URL for WDP>",
          "idServerCloudMsDeployPackageUrl": "<URL for WDP>",
          "bizfxCloudMsDeployPackageUrl": "<URL for WDP>",
          "adventureWorksImagesDeployPackageUrl": "<URL for WDP>",
          "habitatImagesDeployPackageUrl": "<URL for WDP>",
          "indexWorkerDefinitionsDeployPackageUrl": "<URL for WDP>",
          "xConnectDefinitionsDeployPackageUrl": "<URL for WDP>",
          "braintreeMerchantId": "<merchant ID for Braintree account>",
          "braintreePublicKey": "<public key for Braintree account>",
          "braintreePrivateKey": "<private key for Braintree account>",
          "braintreeEnvironment": "<Braintree environment>",
          "environmentName=": "<environment name>"
          "defaultShopName": "<default shop name>",
          "defaultShopCurrency": "<default shop currency>",
          "defaultShopLanguage": "<default shop language>",
        }
      },
```

6. Add the SXA and SXA Storefront modules to the modules block, to specify the location of the SXA ARM template, and the location of the SXA and Storefront WDPs:

```
      {
        "name": "sxa",
        "templateLink": "<URL for SXA ARM template>",
        "parameters": {
          "cdSxaMsDeployPackageUrl": "<URL for SXA CD WDP>",
          "cmSxaMsDeployPackageUrl": "<URL for SXA CM WDP>",
          "speMsDeployPackageUrl": "<URL for Powershell Extensions WDP>"
        }
      },
      {
        "name": "sxa-sf",
        "templateLink": "<URL for SXA Storefront ARM template>",
        "parameters": {
          "sxaMsDeployPackageUrl": "<URL for WDP>",
          "sxaStorefrontMsDeployPackageUrl": "<URL for WDP>",
          "sxaHabitatCatalogMsDeployPackageUrl": "<URL for WDP>",
          "sxaStorefrontThemesMsDeployPackageUrl": "<URL for WDP>",
          "environmentName"": "<environment name>",
          "siteName": "<site name>"
        }
      },
```

7. Add the Bootloader module to the modules block, to specify the location of the Bootloader ARM template and the location of the Bootloader WDP:

```
              {
                "name": "bootloader",
                "templateLink": "<URL for Bootloader ARM template>",
                "parameters": {
                  "msDeployPackageUrl": "<URL for Bootloader WDP>"
                }
              },
```

8. Save your changes to the file.

Chapter 5

# Deploying the solution to Azure

After you have successfully uploaded the Sitecore WDPs and configured the ARM template with the values necessary for your environment, you can deploy your solution to Azure.

## 5.1 Deploying to Azure

When you are ready to deploy your Sitecore solution to Azure, do the following:

1. Open an elevated PowerShell window and navigate to the *Sitecore Azure Toolkit* folder.
2. Add the following Azure account to your PowerShell session:

```
Add-AzureRMAccount
```

   The system prompts you to login to your Azure Subscription account.

3. Load the Sitecore Azure Toolkit module:

```
Import-Module .\tools\Sitecore.Cloud.Cmdlets.psm1
```

4. Select the subscription that you want to deploy to:

```
Set-AzureRMContext -SubscriptionName "<name of the subscription>"
```

5. Enter the following command to initialize deployment:

```
Start-SitecoreAzureDeployment -location <String> -Name <String>
-ArmTemplateUrl <String> -ArmParametersPath <String> -LicenseXmlPath <String>
```

   The following table describes the parameters accepted by the `Start-SitecoreAzureDeployment` cmdlet:

| Parameter | Description |
|---|---|
| `Location` | The geographical region of the current deployment. |
| `Name` | The name of the resource group for the new environment. It can refer to a new or an existing resource group, and is usually the same as the deployment ID. |
| `ArmTemplateURL` | The URL of the Sitecore XP 9.0.0 ARM template file. It points to the HTTP(S) location that is hosting the templates. |
| `ArmParametersPath` | The path to the customized `azuredeploy.parameters.json` file. |
| `LicenseXmlPath` | The path to the Sitecore license file that you want to deploy to the environment. |

# Chapter 6

## Post-installation steps

Once you have successfully installed the Sitecore XC software, you must complete the following tasks to complete your deployment.

This chapter contains the following section:

- Bootstrap and initialize the Commerce Engine
- Generate catalog templates
- Configure user accounts
- Create an SXA Storefront tenant and site
- Re-index the Search indexes

## 6.1 Bootstrap and initialize the Commerce Engine

After you have deployed the Sitecore XC solution to Azure, you must log into the Sitecore Content Management (CM) instance to initialize Sitecore services, then run the Sitecore XC bootstrap operation to ensure that all settings are written to the global database.

The Sitecore Commerce Engine SDK includes code samples for DevOps operations, so that you can access the Sitecore XC API directly. The following instructions assume that you are using Postman to exercise the Sitecore XC API.

**Note**
When you place a call to the Commerce Engine API from outside the Commerce Business Tools (for example, using Postman), you must disable the antiforgery protection setting in the `wwwroot\config.json` file.

To run the bootstrap operation:

1. Install Postman and launch the application.
2. Navigate to the *Sitecore Commerce Engine SDK* folder and open the *Postman* folder.
3. Import the Adventure Works and Habitat environments into Postman.

   **Note**
   Make any necessary changes to the global variables in the environment you are using (such as ServiceHost) under the **Settings**, **Manage Environments** menu in Postman.

4. In the **Collections** pane in Postman, navigate to the *Authentication* folder.
5. Open the *Sitecore* folder and execute the *GetToken* call.

   When Postman displays an access token in the **Body** pane, authentication is successful.
6. In the **Collections** pane, navigate to the *SitecoreCommerce_DevOps* folder.
7. Open the *1 Environment Bootstrap* folder, and execute the *Bootstrap Sitecore Commerce* call.
8. Open the *3 Environment Initialize* folder, and execute the *Initialize Environment* call.

## 6.2 Generate catalog templates

After you have deployed your Sitecore XC solution, you must refresh the cache and generate catalog templates, then republish the site.

You can perform both of these operations from the **Content Editor** on the **Sitecore Launchpad**.

To generate catalog templates:

1. Open a browser, and open the **Sitecore Launchpad** (e.g., https://<server>/sitecore).
1. Click on **Content Editor**.
2. In the Content Editor, click on the **Commerce** tab.
3. Click on **Refresh Commerce Cache** (in the **Caches** tile).
4. Click on **Update Data Templates** (in the **Catalog** tile)**.**

## 6.3 Configure user accounts

After you have deployed your Sitecore XC solution, you must create user accounts and assign the appropriate roles.

Sitecore XC provides the following pre-defined roles:

- Commerce Administrator
- Commerce Business User
- Customer Service Representative
- Customer Service Representative Administrator
- Pricer
- Pricer Manager
- Promotioner
- Promotioner Manager

You configure users and assign roles using the **User Manager** tool on the **Sitecore Launchpad**.

To create users and assign roles:

1. Open a browser, and open the Sitecore Launchpad ( https://<server>/sitecore).
2. Click **User Manager** and create the user accounts with appropriate roles that you need for your deployment, as described on the Documentation Portal.

> **Note**
> Every Sitecore XC user who requires access to the Business Tools must have the *Commerce Business User* role assigned, at a minimum.

## 6.4 Create an SXA Storefront tenant and site

If you want to use the SXA Storefront sample site for testing purposes, or as a starting point to create your own site, you must create a new tenant and site using the Commerce Storefront Template.

1. Open a browser, and open the **Sitecore Launchpad** (e.g., https://<server>/sitecore).
2. Navigate to the **Content Editor** and create a new tenant and site with commerce features as described here.
3. Publish the site as described here.

**Note**
To install the sample site, you must select the Commerce Storefront Template feature when you run the **Create a new Experience Accelerator site** wizard.

## 6.5 Re-index the Search indexes

You must re-index the master and web indexes for the Sitecore XC site. You can perform both of these operations from the **Control panel** on the **Sitecore Launchpad**.

To re-index the Search indexes:

1. On the **Sitecore Launchpad** click on **Control Panel**.
2. In the Control Panel, click on **Indexing Manager**.
3. In the Indexing Manager window, select the following indexes:
   - sitecore_master_index
   - sitecore_web_index
4. Click on **Rebuild** at the bottom of the window to initiate the re-indexing process.