Sitecore XC Installation Guide for the Azure App Service

Sitecore Experience Commerce 10.0

August 13, 2020

Table of Contents

1. Getting started 3
1.1. Sitecore Experience Commerce solution 4
1.1.1. Using Azure Search or SolrCloud 4
1.2. Overview of the installation process 5
1.2.1. Installation workflow using Sitecore Experience Cloud from the Azure Portal 5
1.2.2. Installation workflow using the Sitecore Azure Toolkit 5
2. Preparing your environment 7
2.1. Set up Azure storage 8
2.2. Obtain a valid client certificate 9
2.2.1. Generate the certificate 9
2.2.2. Convert the PFX file to a Base64 string 10
2.3. Create a Braintree account 11
2.4. Download Web Deploy Packages 11
2.5. Sitecore Experience Commerce solution packages for Azure App Service 13
2.5.1. Sitecore Experience Commerce release packages for Azure App Service 13
2.5.2. Sitecore Experience Platform - release packages for XP Scaled 14
2.5.3. Sitecore Experience Accelerator release packages 15
2.5.4. Sitecore Identity service 15
2.5.5. Sitecore Azure Tookit Bootloader 15
3. Deploying from the Azure portal 16
4. Deploying using the Sitecore Azure Toolkit 18
4.1. Install the Sitecore Azure Toolkit 19
4.2. Upload the WDPs to Azure storage 20
4.3. Configure the ARM template 21
4.3.1. Obtain the ARM template and parameters file 21
4.3.2. Customize the Azure environment template 22
4.4. Deploy the solution to Azure 28
5. Post-installation steps 29
5.1. Bootstrap and initialize the Commerce Engine 30
5.1.1. Setup the environment in Postman 30
5.1.2. Bootstrap and initialize the Commerce Engine 30
5.2. Generate catalog templates 32
5.3. Configure user accounts 33
5.4. Create an SXA Storefront tenant and site 34
5.5. Republish the site and rebuild the search indexes 35
5.5.1. Restart the CD server role 35
5.6. Enable antiforgery 36
5.7. Deploy all marketing definitions 36

1. Getting started

This guide describes how to install Sitecore Experience Commerce™ (XC) 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.

> **IMPORTANT**
> Instructions in this guide assume that you have access to a Sitecore XC development (or DevOps) environment, with the Postman API samples deployed. The Postman samples are included as part of the Sitecore Commerce Engine SDK, available for download in the *Sitecore XC Packages for On Premise WDP* release package.

For details on installing Sitecore XC in a single-server on-premise configuration, see the *Sitecore Experience Commerce Installation Guide for On-Premise Solutions*.

1.1. Sitecore Experience Commerce solution

Sitecore Experience Commerce (Sitecore 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 for building a customized storefront.

1.1.1. Using Azure Search or SolrCloud

You can use Microsoft Azure Search or SolrCloud as the search provider when deploying Sitecore Experience Commerce™ (XC) in an Azure App Service configuration. Microsoft Azure Search is the default search engine. See the Sitecore Knowledge base Solr compatibility and Azure compatibility.

This documentation does not provide information on how to install SolrCloud itself. If you use SolrCloud as a search provider in Azure, follow the instructions on the Solr website to install Solr and create a scaled environment. For instructions on how to set up SolrCloud on Sitecore XC and create index collections, see this walkthrough.

NOTE

The default search engine for:

- Sitecore XC on-premise installation is Solr
- Sitecore XC Cloud deployment is Microsoft Azure Search
- Sitecore XC with Kubernetes is SolrCloud
- Sitecore XC on-premise with containers is Solr

1.2. Overview of the installation process

You can choose to install Sitecore Experience Commerce (XC) solution to Azure in the two following ways:

- Install Sitecore XC from the Microsoft Azure Portal
- Install Sitecore XC using the Sitecore Azure Toolkit

> **NOTE**
> The installation process for deploying your Sitecore XC solution to Azure using the Sitecore Azure Toolkit requires that you perform additional prerequisite steps (for setting up the Sitecore Azure Toolkit).

1.2.1. Installation workflow using Sitecore Experience Cloud from the Azure Portal

In this deployment scenario, you access the Sitecore Experience Cloud app from the Azure Marketplace, and use the Microsoft Azure Portal to deploy the Sitecore Experience Commerce solution.

Following is a list of high level tasks that you perform when deploying Sitecore Experience Commerce solution to the Azure App Service from the Azure Portal.

1. Prepare your environment by completing the following prerequisite tasks:
    - Obtain a valid client certificate.
    - Create a Braintree sandbox account (for web payment functionality).
2. Deploy the Sitecore Experience Commerce solution from the Azure Portal
3. Complete the post-installation setup steps to finish the installation.

1.2.2. Installation workflow using the Sitecore Azure Toolkit

Following is a list of the high level tasks that you perform when deploying Sitecore Experience Commerce to the Azure App Service using the Sitecore Azure Toolkit.

1. Prepare your environment by completing the following prerequisite tasks:
    - Set up Azure storage.
    - Obtain a valid client certificate.
    - Create a Braintree sandbox account (for web payment functionality).
    - Download the Web Deployment Packages
2. Setup the Sitecore Azure Toolkit.
    - Install the Sitecore Azure Toolkit
    - Obtain the Web Deploy Packages
    - Configure the ARM template
3. Deploy the Sitecore Experience Commerce solution using the Sitecore Azure Toolkit.
4. Complete the post-installation setup steps to finish the installation.

2. Preparing your environment

This section outlines the tasks you must complete before installing your Sitecore XC solution on the Azure App Service:

- Set up Azure storage (only required if you use the Sitecore Azure Toolkit)
- Obtain a valid client certificate
- Create a Braintree account
- Download Web Deploy Packages (only required if you use the Sitecore Azure Toolkit)

2.1. Set up Azure storage

> **NOTE**
> This procedure is only required if you are using the Sitecore Azure Toolkit. You do not need to perform this procedure if you are deploying using the Sitecore Cloud App from the Azure Portal.

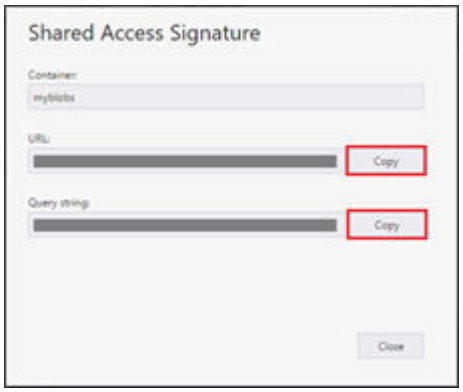
The WDPs containing the application code and resources must be accessible to the Azure Resource Manager over the Internet. You can use a Microsoft Azure® storage account to host the Sitecore WDPs and ARM templates for your deployment.

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.

Once you have created your blob containers, 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.

After you have created the SAS token, 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.



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

2.2. Obtain a valid client certificate

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 topic Client certificate for Sitecore deployments topic on the Sitecore Documentation site.

2.2.1. 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 Premise** link).
2. Extract the files.
3. Locate the `Sitecore.Commerce.Engine.SDK.x.x.xx` package and extract the contents.
4. Navigate to the `scripts` folder and open the `New-DevelopmentCertificate-Azure.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.
   Descriptions of all parameters in the certificate generation script are available here.

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.

7. Add the `-Provider` parameter to the new certificate code block (starting on line 27):

```
$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.

2.2.2. 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:

   ```
   [System.Convert]::ToBase64String([System.IO.File]::ReadAllBytes("path to pfx file"))
   ```

   The console displays the generated Base64-encoded string.

2. Copy the converted string from the console display, and save it as a text file. (In a subsequent step, you need to specify the generated 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.

2.3. 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.

2.4. Download Web Deploy Packages

> **NOTE**
> This procedure is only required if you are using the Sitecore Azure Toolkit. You do not need to perform this procedure if you are deploying using the Sitecore Cloud App from the Azure Portal.

Sitecore creates Web Deploy Packages (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 your Sitecore Experience Commerce solution, when deploying your Sitecore Experience Commerce solution using Sitecore Azure Toolkit.

The Web Deploy Packages (WDPs) for Sitecore Experience Commerce (XC) Solution for Azure App Service are available from the Sitecore Download site. The WDPs come in a single .zip file and are grouped on the Download Site, per version, and per topology.

> **NOTE**
> The Sitecore Experience Commerce (XC) for Azure packages contains Web Deploy Packages (WDPs) for Sitecore XC-specific components only. You must download WDPs for Sitecore XP, Sitecore SXA, Sitecore PowerShell Extension, and for the Sitecore Identity server separately.

To download WDPs required for a Sitecore Experience Commerce solution:

1. Open a browser, and download the following release packages (for example, to your *Download* folder):
   - Sitecore Experience Platform (XP) release packages for Azure AppService - Packages for XP Scaled (from the section Download options for Azure AppService)

     > **NOTE**
     > For Sitecore XC 10.0, only the Sitecore XP Scaled topology is supported

- Sitecore Experience Commerce (XC) release packages for Azure App Service
- Sitecore Experience Platform Accelerator (SXA) WDP for 10.0
- Sitecore Experience Platform Accelerator (SXA) CD WDP for 10.0
- Sitecore PowerShell Extension WDP for Sitecore
- Sitecore Identity service WDPs

2. Extract the contents of the zip file to obtain individual WDP files, (except the Sitecore Identity service release package), so that they are ready for subsequent deployment tasks. A complete list of required WDPs is available here.

> **NOTE**
> The Sitecore Identity service release package is a zipped WDP. You do not need to unzip it as it does not contain any zipped sub-folders.

2.5. Sitecore Experience Commerce solution packages for Azure App Service

This topic lists the Web Deploy Packages (WDPs) that are required for your Sitecore Experience Commerce Azure deployment.

- Sitecore Experience Commerce WDPs for Azure App Service
- Sitecore Experience Platform - release packages for XP Scaled
- Sitecore Experience Accelerator release packages
- Sitecore Identity service release package
- Sitecore Azure Tookit Bootloader

2.5.1. Sitecore Experience Commerce release packages for Azure App Service

The following tables list and describe the Web Deploy Packages (WDPs) included in the Sitecore Experience Commerce release package for Azure App Service. Each package has a unique version number and a `.scwdp.zip` extension.

> **NOTE**
> You also need the contents of the nested folder, which contains the .json and the infrastructure .json files.

Package	Description
Sitecore.Commerce.Engine.Azure	Contains the binary for the Commerce Engine, with support for Azure Search as the search provider.
Sitecore.Commerce.Engine.Solr	Contains the binary for the Commerce Engine, with support for Solr as the search provider.
SolrSchemas.Sitecore.Commerce	Contains Solr schema files for SolrCloud search configuration.
Sitecore.Identity.Config.Commerce	Contains configuration for interaction with the Sitecore Identity service.
Sitecore.BizFx	Contains an integrated suite of Sitecore XC business tools, built on the Angular application platform version 4.
Sitecore.Commerce.Habitat.Images	Contains images for the Habitat sample catalog.
Adventure Works Images	Contains images for the Adventure Works sample catalog.
Sitecore Commerce Connect Core	Contains a middleware integration layer between the Sitecore XC back-end and the front-end Storefront, and with the Sitecore XP.
Sitecore Commerce Connect Schema Definitions for IndexWorker	Contains schema definitions for the Commerce Connect IndexWorker.
Sitecore Commerce Connect Definitions for xConnect	Contains definitions for xConnect.
Sitecore Commerce Engine Connect CD Azure	Contains Commerce Engine Connect (a thin integration layer for integrating the Commerce Engine with Sitecore Commerce Connect Core) on the Content Delivery instance, configured for Azure Search.

Package	Description
Sitecore Commerce Engine Connect CD Solr	Contains Commerce Engine Connect (a thin integration layer for integrating the Commerce Engine with Sitecore Commerce Connect Core) on the Content Delivery instance, configured for Solr.
Sitecore Commerce Engine Connect CM Azure	Contains Commerce Engine Connect (a thin integration layer for integrating the Commerce Engine with Sitecore Commerce Connect Core) on the Content Management instance, configured for Azure Search.
Sitecore Commerce Engine Connect CM Solr	Contains Commerce Engine Connect (a thin integration layer for integrating the Commerce Engine with Sitecore Commerce Connect Core) on the Content Management instance, configured for Solr.
Sitecore Commerce Experience Accelerator	Contains Commerce-specific extensions to the Sitecore Experience Accelerator's (SXA) templated UX layouts (for example, UI renderings used to display a catalog in the Storefront).
Sitecore Commerce Experience Accelerator Storefront	Contains the sample and starter storefront as an integrated part of the Sitecore XC solution, and built using Sitecore's new SXA UX layouts.
Sitecore Commerce Experience Accelerator Habitat	Contains the Habitat sample catalog for the Storefront site.
Sitecore Commerce Experience Accelerator Storefront Themes	Contains the themes used for the SXA Storefront site.
Sitecore Commerce ExperienceAnalytics Core	Contains Commerce Experience Analytics installer and core files required to integration with Commerce.
Sitecore Commerce ExperienceProfile Core	Contains Commerce Experience Profile installer and core files required to integrate Experience Profile with Commerce.
Sitecore Commerce Marketing Automation Core	Contains Commerce Marketing Automation installation and core files.
Sitecore Commerce Marketing Automation for Azure	Contains Commerce Marketing automation files required to setup the Marketing Automation Engine service.

2.5.2. Sitecore Experience Platform - release packages for XP Scaled

Following is a list of the WDPs provided as part of the Sitecore Experience Platform Release package (XP Scaled).

> **NOTE**
> You also need the contents of the nested folder, which contains the .json and the infrastructure .json files.

Sitecore Experience Platform - Web Deploy Packages
Sitecore 10.0.0 rev.xxxxx (Cloud)_cd
Sitecore 10.0.0 rev.xxxxx (Cloud)_cm
Sitecore 10.0.0 rev.xxxxx (Cloud)_dds
Sitecore 10.0.0 rev.xxxxx (Cloud)_prc
Sitecore 10.0.0 rev.xxxxx (Cloud)_rep
Sitecore 10.0.0 rev.xxxxx (Cloud)_xp1collection
Sitecore 10.0.0 rev.xxxxx (Cloud)_xp1collectionsearch
NOTE If you are using SolrCloud as a search provider in your Azure deployment, make sure to replace this package with the Sitecore 10.0.0 rev.xxxxx (Cloud)_xp1collectionsearchSolr package.
Sitecore 10.0.0 rev.xxxxx (Cloud)_xp1collectionsearchSolr
NOTE This package is only used in Azure deployment that use SolrCloud as the search provider.
Sitecore 10.0.0 rev.xxxxx (Cloud)_xp1cortexprocessing
Sitecore 10.0.0 rev.xxxxx (Cloud)_xp1cortexreporting
Sitecore 10.0.0 rev.xxxxx (Cloud)_xp1marketingautomation
Sitecore 10.0.0 rev.xxxxx (Cloud)_xp1marketingautomationreporting
Sitecore 10.0.0 rev.xxxxx (Cloud)_xp1referencedata
Sitecore.Patch.EXM (Cloud)_CM.zip

2.5.3. Sitecore Experience Accelerator release packages

Following is a list WDPs provided as part of the Sitecore Experience Accelator release package.

- Sitecore Experience Accelerator WDP for 10.0
- Sitecore Experience Accelerator CD WDP for 10.0
- Sitecore PowerShell Extensions WDP 6.1.1 (or later) for Sitecore 10.0

2.5.4. Sitecore Identity service

The WDP for Sitecore Identity: Sitecore.IdentityServer.X.X.X.scwdp.zip.

> **NOTE**
> For information on the compatibility of Sitecore XC with browsers, operating systems, .NET frameworks, and database servers, see the Sitecore compatibility table.

2.5.5. Sitecore Azure Tookit Bootloader

The following WDP is packaged as part of the Sitecore Azure Toolkit zip file:

- Sitecore.Cloud.Integration.Bootload.wdp.zip

3. Deploying from the Azure portal

You can deploy your Sitecore Experience Commerce (XC) solution from the Azure Portal using the Sitecore Experience Cloud application, available from the Microsoft Azure Marketplace.

> **NOTE**
> This procedure is specifically for a deployment using the Azure Portal. For instructions on how to deploy using the Sitecore Azure Toolkit, see Deploying using the Sitecore Azure Toolkit.

When you deploy a Sitecore XC solution from the Azure Portal, you must obtain the Sitecore Experience Cloud and, during deployment, select the appropriate options and modules. The Sitecore Experience Cloud includes the Sitecore Experience Platform (XP) solution, Sitecore Experience Commerce and Sitecore Experience Accelerator (SXA) Storefront as modules of that deployment.

To deploy your Sitecore XC solution from the Azure Portal:

1. Login to your Microsoft Azure portal.
2. In the left page, click on **Create a resource**.
3. In the **Azure Marketplace** window, search for Sitecore Experience Cloud.
4. On the Sitecore Experience Cloud tab, click **Create**.
5. Select your Azure subscription.
6. Under Resource Group, select the **Create new** option and specify a name for the new resource group. Note that currently, all characters must be lower case.
7. Under Sitecore Settings, click on **Configure required settings** to open the settings panel.
8. On the **Environment** tab, select values for the following parameters, from each drop-down list:
   - **Version**: select the current version of Sitecore XC.
   - **Topology**: select **Sitecore Experience Cloud (XP)**.
   - **Configuration**: select **Scaled Development**.
   - **Size**: select **Small**, **Medium**, or **Large**, based on your deployment.
   - **Additional Modules**: select **Sitecore Experience Commerce 10.0 Initial Release**, and optionally, the corresponding **Sitecore Experience Accelerator Storefront**.

     > **NOTE**
     > When you select the SXA Storefront module, the Sitecore Experience Commerce 10.0 and Sitecore Experience Accelerator 10.0 modules are automatically selected as well.

9. Click **Next**.
10. On the **Credentials** tab, in the **Sitecore** dialog box, specify values for the Sitecore Experience Platform deployment:

- **Sitecore Administrator password**
- **Sitecore Administrator password confirmation**
- **SQL Server login**
- **SQL Server password**
- **SQL Server password confirmation**
- **License XML** file (upload to the portal)
- **Authentication certificate** (either auto-generate a self-signed certificate or use an existing certificate from a trusted authority, with name and password).
- **Search provider** (the Search Provider used with your Sitecore deployment)

11. On the **Credentials** tab, in the **Sitecore Commerce 10.0** dialog box, specify values for the following parameters for Sitecore Experience Commerce module:
    - **Braintree Merchant Id** (optional - for payment provider service)
    - **Braintree Public Key** (optional - for payment provider service)
    - **Braintree Private Key** (optional - for payment provider service)
    - **Environment Name** (default value is *HabitatShops*)
    - **Default Shop Name** (default value is *CommerceEngineDefaultStorefront*)
    - **Default Shop Currency** (default value is *USD*)

      NOTE
      If the value for the shop currency is not one of the default currencies supported by the Commerce Engine (USD, CAD, or EUR), you must add the currency to the **Default** currency set on the Commerce Control Panel after you deploy your solution to the Azure App service.

    - **Default Shop Language**
12. Click **Next**.
13. On the **Region** tab, select a region for the deployment location and the Application Insights region from the drop-down lists, and click **Next**.
14. On the **Summary** tab, review the information and click **Ok**.
15. Review the Legal terms and click **Create** to start the deployment.

You can monitor your deployment by going to your **Resource groups** page and checking the status of your new resource group.

Deployment takes approximately 2 hours.

When deployment is done, you must complete the tasks described in the Post-installation steps topic.

4. Deploying using the Sitecore Azure Toolkit

You can deploy your Sitecore XC solution using the Sitecore Azure Toolkit.

When you deploy your solution using the Sitecore Azure Toolkit, you upload the solution Web Deploy Packages (WDPs) into Azure storage, customize the Azure Resource Manager (ARM) template, and deploy the solution to Azure from a PowerShell session.

This chapter contains the following sections:

- Install the Sitecore Azure Toolkit
- Upload the WDPs to Azure storage
- Configure the ARM template
- Deploy the solution to Azure

4.1. 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, as described in the Getting started with the Sitecore Azure Toolkit topic.

4.2. Upload the WDPs to Azure storage

Your 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 all WDPs for your configuration (including WDPs for XP, XC, SXA, Sitecore Identity server, and the Sitecore Azure Toolkit Bootloader if you are deploying using the Sitecore Azure Toolkit) to the container that you created earlier.

   > **NOTE**
   > Make sure that you take note of the URL for each WDP that you upload, and append the SAS token that you generated earlier to it. You must specify this URL (with SAS token) in the ARM templates, in the next procedure.

3. Upload the `azuredeploy.json`

4.3. Configure 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.

4.3.1. Obtain the ARM template and parameters file

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

For this scenario, you use the Sitecore XP 10.0 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 10.0.0\XP* folder.
2. In the *Sitecore 10.0.0\XP* folder, click on the `azuredeploy.json` file to open it, and in the document header row, click on **Raw**.
3. Copy the URL directly from the address bar and make note of the information for later use in the Azure environment file (for example, paste it into Notepad so that you have it ready for later steps). The copied URL should be similar to the following: `https://raw.githubusercontent.com/Sitecore/Sitecore-Azure-Quickstart-Templates/master/Sitecore 10.0.0/XP/azuredeploy.json`
4. Return to the Sitecore GitHub and repeat the above steps to capture the URL of each of the following ARM templates (`azuredeploy.json` file):
   - The Sitecore XC 10.0.0 ARM template for Sitecore XP Scaled
   - The SXA 10.0 ARM template for Sitecore XP
   - The SXA Storefront 3.0 ARM template

   NOTE
   Make sure to copy the URLs in raw format and save them to a location where you can retrieve them later (to use as parameter values in the `azuredeploy.parameters.json`).

5. Return to the *Sitecore 10.0.0\XP* folder and download the `azuredeploy.parameters.json` file.

   NOTE
   You only need to download the Sitecore 10.0.0 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).

4.3.2. 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 parameters listed below. 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, contain no spaces, and not be longer than 64 characters.
`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.
`siMsDeployPackageUrl`	The URL to a Sitecore Identity WDP (for example, `Sitecore.IdentityServer *.zip`)
`cmMsDeployPackageUrl`	The URL to a Sitecore XM Content Management WDP (`Sitecore 10.0.X rev. * (Cloud)_cm.scwdp.zip`).
`cdMsDeployPackageUrl`	The URL for a Sitecore XM Content Delivery WDP (`Sitecore 10.0.X rev. *(Cloud)_cd.scwdp.zip`).
`prcMsDeployPackageUrl`	The URL for a Sitecore XP Processing WDP (`Sitecore 10.0.X rev. *(Cloud)_prc.scwdp.zip`).
`repMsDeployPackageUrl`	The URL for a Sitecore XP Reporting WDP (`Sitecore 10.0.X rev. *(Cloud)_rep.scwdp.zip`).
`xcRefDataMsDeployPackageUrl`	The URL for an xConnect Reference Data service WDP (`Sitecore 10.0.X rev. *(Cloud)_xp1referencedata.scwdp.zip`).
`xcCollectMsDeployPackageUrl`	The URL for an xConnect Collection service WDP (`Sitecore 10.0.2 rev. *(Cloud)_xp1collection.scwdp.zip`).
`xcSearchMsDeployPackageUrl`	The URL for an xConnect Collection Search service WDP (`Sitecore 10.*.* rev. *(Cloud)_xp1collectionsearch.scwdp.zip`) **NOTE** If you are using SolrCloud as the search provider, ensure to replace the `_xp1collectionsearch` package with the `_xp1collectionsearchsolr` package.

Parameter	Description
`cortexProcessingMsDeployPackageUrl`	The URL for a Cortex Processing service WDP (`Sitecore 10.0.X rev. *(Cloud)_xp1cortexprocessing.scwdp.zip`).
`cortexReportingMsDeployPackageUrl`	The URL for a Cortex Reporting service WDP (`Sitecore 10.0.0 rev. *(Cloud)_xp1cortexreporting.scwdp.zip`).
`maOpsMsDeployPackageUrl`	The URL for a Marketing Automation service WDP (`Sitecore 10.0.X rev. * (Cloud)_xp1marketingautomation.scwdp.zip`).
`maRepMsDeployPackageUrl`	The URL for a Marketing Automation Reporting service WDP (`Sitecore 10.0.0 rev. * (Cloud)_xp1marketingautomationreporting.scwdp.zip`).
`authCertificateBlob`	The Base64-encoded blob of the authentication certificate for the Sitecore XC solution.
`authCertificatePassword`	The password for the authentication certificate.
`templateLinkAccessToken`	The token to access XP `azuredeploy.json` template.
`exmDdsMsDeployPackageUrl`	The URL to a EXM deploy package (`Sitecore 10.0.X rev. *(Cloud)_dds.scwdp.zip`). (This parameter is not mandatory in a Commerce deployment. Refer to the Sitecore configurations and topology for Azure topic for more information about configurations.)
`exmCmMsDeployPackageUrl`	The URL to the EXM patch file (`Sitecore.Patch.EXM (Cloud)_CM.zip`). (This parameter is not mandatory for a Commerce deployment. Refer to the Sitecore configurations and topology for Azure topic for more information about configurations.)

2. If you are using SolrCloud as a search provider (instead of AzureSearch, for example), add the following parameter to specify the URL for the SolrCloud instance:

```
"solrConnectionString" : <URL for SolrCloud instance>
```

> **NOTE**
> Leave this parameter empty if you are using the default Azure search.
>
> If you are using SolrCloud as the search provider, make sure to replace the `Sitecore 10.*.* rev.xxxxx (Cloud)_xp1collectionsearch` package with the `Sitecore 10.*.* rev.xxxxx (Cloud)_xp1collectionsearchsolr` package.

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

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

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).

> **NOTE**
> The `templateLink` parameter value is the URL that you copied previously as part of this previous step.

> **NOTE**
> Regardless of the search provider you are using, you must provide a URL for all WDPs, including Azure and Solr packages.

```
"modules": {
  "value": {
    "items": [
      {
       "name": "Commerce",
       "templateLink": "<URL for the Sitecore XC ARM template>",
       "parameters": {
         "templateLinkAccessToken": "<SAS-token>",
         "commerceConnectMsDeployPackageUrl": "<URL for WDP>",
         "commerceEngineAzureMsDeployPackageUrl": "<URL for WDP>",
         "commerceEngineSolrCloudMsDeployPackageUrl": "<URL for WDP>",
         "commerceEngineConnectCdAzureMsDeployPackageUrl": "<URL for WDP>",
         "commerceEngineConnectCmAzureMsDeployPackageUrl": "<URL for WDP>",
         "commerceEngineConnectCdSolrMsDeployPackageUrl": "<URL for WDP>",
         "commerceEngineConnectCmSolrMsDeployPackageUrl": "<URL for WDP>",
         "identityConfigCommerceMsDeployPackageUrl": "<URL for WDP>",
         "bizfxCloudMsDeployPackageUrl": "<URL for WDP>",
         "adventureWorksImagesDeployPackageUrl": "<URL for WDP>",
         "habitatImagesDeployPackageUrl": "<URL for WDP>",
         "indexWorkerDefinitionsDeployPackageUrl": "<URL for WDP>",
         "xConnectDefinitionsDeployPackageUrl": "<URL for WDP>",
         "xAnalyticsCoreDeployPackageUrl": "<URL for WDP>",
         "xProfileCoreDeployPackageUrl": "<URL for WDP>",
         "maCoreDeployPackageUrl": "<URL for WDP>",
         "maAzureDeployPackageUrl": "<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": "<HabitatShops>",
         "defaultShopName": "<CommerceEngineDefaultStorefront>",
         "defaultShopCurrency": "<USD>",
         "defaultShopLanguage": "<en>",
         "commerceEngineConnectClientId": "CommerceEngineConnect"
       }
    },
```

The following table lists and describes the parameters.

Parameter	Description
`templateLinkAccessToken`	The token to access the XC `azuredeploy.json` template.
`commerceConnectMsDeployPackageUrl`	The URL for the WDP (`Sitecore Commerce Connect Core *.scwdp.zip`).

Parameter	Description
commerceEngineAzureMsDeployPackageUrl	The URL for the WDP (Sitecore.Commerce.Engine.Azure.*.scwdp). **NOTE** Required in both SolrCloud and Azure deployments.
commerceEngineSolrCloudMsDeployPackageUrl	The URL for the WDP (Sitecore.Commerce.Engine.Solr.*.scwdp.zip). **NOTE** Required in both SolrCloud and Azure deployments.
commerceEngineConnectCdAzureMsDeployPackageUrl	The URL for the WDP (Sitecore Commerce Engine Connect CD Azure *.scwdp.zip). **NOTE** Required in both SolrCloud and Azure deployments.
commerceEngineConnectCmAzureMsDeployPackageUrl	The URL for the WDP (Sitecore Commerce Engine Connect CM Azure *.scwdp.zip). **NOTE** Required in both SolrCloud and Azure deployments.
commerceEngineConnectCdSolrMsDeployPackageUrl	The URL for the WDP (Sitecore Commerce Engine Connect CD Solr *.scwdp.zip).0 **NOTE** Required in both SolrCloud and Azure deployments.
commerceEngineConnectCmSolrMsDeployPackageUrl	The URL for the WDP (Sitecore Commerce Engine Connect CM Solr *.scwdp.zip). **NOTE** Required in both SolrCloud and Azure deployments.
bizfxCloudMsDeployPackageUrl	The URL for the WDP (Sitecore.BizFx.*.scwdp.zip)
identityConfigCommerceMsDeployPackageUrl	The URL for the WDP (Sitecore.Identity.Config.Commerce.*.scwdp.zip).
adventureWorksImagesDeployPackageUrl	The URL for the WDP (Aventure Works Images.scwdp.zip).
habitatImagesDeployPackageUrl	The URL for the WDP(Sitecore.Commerce.Habitat.Images-*.scwdp.zip).
indexWorkerDefinitionsDeployPackageUrl	The URL for the WDP (Sitecore Commerce Connect Schema Definitions for IndexWorker *.scwdp.zip).

Parameter	Description
`xConnectDefinitionsDeployPackageUrl`	The URL for the WDP (`Sitecore Commerce Connect Schema Definitions for xConnect *.scwdp.zip`).
`xAnalyticsCoreDeployPackageUrl`	The URL for the WDP (`Sitecore Commerce ExperienceAnalytics Core *.scwdp.zip`).
`xProfileCoreDeployPackageUrl`	The URL for the WDP (`Sitecore Commerce ExperienceProfile Core *.scwdp.zip`).
`maCoreDeployPackageUrl`	The URL for the WDP (`Sitecore Commerce Marketing Automation Core *.scwdp.zip`).
`maAzureDeployPackageUrl`	The URL for the WDP (`Sitecore Commerce Marketing Automation for Azure *.scwdp.zip`).
`braintreeMerchantId`	You merchant ID for the Braintree payment provider.
`braintreePublicKey`	The public key associated to your Braintree account.
`braintreePrivateKey`	The private key associated to your Braintree account.
`braintreeEnvironment`	The Braintree environment.
`environmentName`	The environment name (the default value is *HabitatShops*).
`defaultShopName`	The default shop name, (the default value is *CommerceEngineDefaultStorefront*).
`defaultShopCurrency`	The currency to use by default by the shop (the default value *USD* (US dollar). **NOTE** If the value for `defaultShopCurrency` is not one of the default currencies supported by the Commerce Engine (USD, CAD, or EUR), you must add the currency to the **Default** currency set on the Commerce Control Panel after you deploy your solution to the Azure App service.
`commerceEngineConnectClientId`	The client ID assigned to Commerce Engine Connect for Sitecore Identity. This ID is used to identify the Commerce Engine Connect with Commerce Engine.

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:

> **NOTE**
> The `templateLink` parameter values in the following code samples refer to the URLs that you copied as part of this previous step.

```
    {
     "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>"
       "templateLinkAccessToken": "<Token>"
      }
  },
```

```
    {
     "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>",
  },
```

Following is a table that describes the parameters used in the above code sample.

Parameter	Description
`cdSxaMsDeployPackageUrl`	The URL for the WDP (`Sitecore Experience Accelerator * CD.scwdp.zip`)
`cmSxaMsDeployPackageUrl`	The URL for the WDP (`Sitecore Experience Accelerator 10.0 rev. * for 10.0 .scwdp.zip`)
`speMsDeployPackageUrl`	The URL for the WDP (`Sitecore PowerShell Extensions *.scwdp.zip`)
`templateLinkAccessToken`	The token to access the SXA ARM template.
`sxaMsDeployPackageUrl`	The URL for the WDP (`Sitecore Commerce Experience Accelerator *.scwdp.zip`)
`sxaStorefrontMsDeployPackageUrl`	The URL for the WDP (`Sitecore Commerce Experience Accelerator Storefront *.scwdp.zip`).
`sxaHabitatCatalogMsDeployPackageUrl`	The URL for the WDP (`Sitecore Commerce Experience Accelerator Habitat Catalog *.scwdp.zip`).
`sxaStorefrontThemesMsDeployPackageUrl`	The URL for the WDP (`Sitecore Commerce Experience Accelerator Storefront *.scwdp.zip`).

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:

> **NOTE**
> The Bootloader ARM template is packaged with the Sitecore Azure Toolkit, and is available on Sitecore GitHub (in the *Sitecore * /XP/addons/ folder)*.

```
    {
     "name": "bootloader",
     "templateLink": "",
     "parameters": {
      "msDeployPackageUrl": ""
    }
  },
```

The value for the `"msDeployPackageUrl"` parameter is the URL for the WDP (`Sitecore.Cloud.Integration.Bootload.wdp.zip`).

8. Save your changes to the file.

4.4. Deploy 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.

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 .\Cmdlets\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 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.

5. Post-installation steps

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

- Bootstrap and initialize the Commerce Engine
- Generate catalog templates
- Configure user accounts
- Create an SXA Storefront tenant and site
- Republish the site and rebuild the search indexes
- Disable Application Request Routing Cookie
- Enable antiforgery

5.1. 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 run the Sitecore XC bootstrap operation to ensure that all settings are written to the global database), and then initialize the Sitecore services.

The Sitecore Commerce Engine SDK includes samples of API calls 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**
> The following instructions assume that you have access to a Sitecore XC development (or DevOps) environment, with the Postman API samples deployed. The Postman samples are included as part of the Sitecore Commerce Engine SDK, available for download in *Sitecore XC Packages for On Premise WDP*.

5.1.1. Setup the environment in Postman

You must setup the environment in Postman to point to your Azure deployment before you can exercise the API samples.

To setup the environment in Postman, for example, the *Habitat Environment*:

1. In the top right corner of Postman, click **Settings**.
2. In the **Manage Environments** dialog, click the *Habitat Environment* and edit the environment current values as follows:

Variable	Current value
Environment	HabitatAuthoring
ShopperId	ShopperId
Language	en-US
Currency	USD
ServiceHost	https://<deployment name>-authoring.azurewebsites.net
OpsApiHost	https://<deployment name>-ops.azurewebsites.net
AuthoringHost	https://<deployment name>-authoring.azurewebsites.net
MinionsHost	https://<deployment name>-minions.azurewebsites.net
ShopsHost	https://<deployment name>-shops.azurewebsites.net
SitecoreIdServerHost	https:<deployment name>-si.azurewebsites.net
HostName	<deployment-name>

5.1.2. Bootstrap and initialize the Commerce Engine

To run the bootstrap and initialize operations:

1. In the top right corner of Postman, click the environment selector and select the environment, for example the *Habitat Environment*.

2. In the Postman **Collections** pane, navigate to the *SitecoreCommerce_DevOps* folder.
3. Open the *1 Environment Bootstrap* folder, and execute the **Bootstrap Sitecore Commerce** call.
4. Open the *3 Environment Initialize* folder, and execute the **Ensure\Sync default content paths** call.

> **NOTE**
> If the `status` of a request is `WaitingForActivation`, you can execute the Check Long Running Command Status request. When you execute the `CheckCommandStatus` request, you must ensure you are calling the same service that the previous command was executed in.

5. In the *3 Environment Initialize* folder, execute the **Initialize Environment** call.

> **NOTE**
> If the `status` of a request is `WaitingForActivation`, you can execute the Check Long Running Command Status request. When you execute the `CheckCommandStatus` request, you must ensure you are calling the same service that the previous command was executed in.

6. Repeat step 4 and 5 above for other environments is applicable (for example, for the AdventureWorks environment).
7. After you have run the bootstrap and initialize operations, go to the Azure Portal and restart the Minions service, Authoring service, CM service and Shop service.

5.2. 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** (for example, https://<deployment name>-cm.azurewebsites.net/sitecore).
2. Click on **Content Editor**.
3. In the Content Editor, click on the **Commerce** tab.
4. Click on **Refresh Commerce Cache** (in the **Caches** tile).
5. Click on **Update Data Templates** (in the **Catalog** tile)**.**

5.3. Configure user accounts

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

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

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

Refer to the User roles and permissions topic for information on the pre-defined roles and associated permissions for the Sitecore XC Business Tools.

5.4. Create an SXA Storefront tenant and site

If you want to use the SXA Storefront site as a starting point to create your own e-commerce site, you must create a new tenant and storefront site using this procedure.

5.5. Republish the site and rebuild 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**.

> **NOTE**
> It is best practice to rebuild each index separately. It does not matter which one you choose to build first.

To re-publish the site:

1. Log in to Sitecore Launchpad and click on **Content Editor**.
2. Click on the **Publish** tab.
3. Click **Publish** and select **Publish site** from the drop-down list.
4. In the **Publish Site** window, select **Republish – publish everything** and click **Publish**.

To re-index the Search indexes:

1. In the Control Panel, click on **Indexing Manager**.
2. In the **Indexing Manager** dialog box, select **sitecore_master_index** and click **Rebuild**.
3. When the Sitecore master index is rebuilt, repeat the steps above to rebuild the **sitecore_web_index**.

5.5.1. Restart the CD server role

In the Azure Portal, restart the CD server role.

5.6. Enable antiforgery

Enable antiforgery to create antiforgery tokens that protect Sitecore Experience Commerce (XC) applications against cross-site request forgery (CSRF).

> **NOTE**
> This procedure assumes you have already setup your own custom domain for Azure App Service. Due to browser restrictions, you cannot use the *azurewebsites.net* domain to enable antiforgery protection.

When you enable antiforgery for a specific domain, Sitecore XC applies the protection to its sub-domains.

To enable antiforgery on a domain and its sub-domains:

1. In the Azure Portal, select the appropriate Sitecore XC app service, for example, the instance of the Commerce Engine hosting the authoring service (*<xc92xxxxxx>-authoring)*.
2. Open the Azure App Editor, and open the `\wwwroot\wwwroot\config.json` file.
3. Set `"AntiForgeryEnabled"` to `true`.
4. Set `"CommerceServicesHostPostfix"` to specify `<your domain>`.

The following example shows a configuration that creates antiforgery tokens for the domain **.yoursite.com* (and its sub-domains). This approach allows each website, for example, each Commerce Engine BizFx website, to have its own distinct sub-domain with antiforgery protection enabled.

```
"AntiForgeryEnabled": true
CommerceServicesPostfix": "yoursite.com"
```

5.7. Deploy all marketing definitions

To make sure that the most up-to-date order information is visible in Sitecore Experience Analytics reports and in Sitecore Experience Platform, you must deploy all marketing definitions and taxonomies.