Cloudinary SFRA Integration Guide

**Version 20.1.0 for year 2020**

Icon

Description automatically generated

****

**Table of Contents**

[1. Summary 3](#_Toc53745428)

[2. Component Overview 4](#_Toc53745429)

[2.1 Functional Overview 4](#_Toc53745430)

[2.2 Limitations, Constraints 4](#_Toc53745431)

[2.3 Compatibility 4](#_Toc53745432)

[2.4 Privacy, Payment 4](#_Toc53745433)

[3. Implementation Guide 5](#_Toc53745434)

[3.1 General Implementation 5](#_Toc53745435)

[3.1.1 Installing the Cartridges on the sandbox 5](#_Toc53745436)

[3.1.2 Setting Up Business Manager 5](#_Toc53745437)

[3.2 Configuring Cloudinary 9](#_Toc53745438)

[3.2.1 Adding Structured Metadata in Cloudinary 9](#_Toc53745439)

[3.2.2 Enabling Resource Listing 10](#_Toc53745440)

[3.3 Configuring the Cartridge Site Preferences: 10](#_Toc53745441)

[3.4 System Object Types: 19](#_Toc53745442)

[3.5 Jobs Details: 23](#_Toc53745443)

[3.6 Services Details: 27](#_Toc53745444)

[3.7 Cartridge Operation Modes: 28](#_Toc53745445)

[3.8 Code Changes 29](#_Toc53745446)

[3.9 Managing Videos 37](#_Toc53745447)

[4. Operations, Maintenance 38](#_Toc53745448)

[4.1 Data Storage 38](#_Toc53745449)

[4.2 Availability 38](#_Toc53745450)

[4.3 Support 39](#_Toc53745451)

[4.4 Cloudinary Workflow 39](#_Toc53745452)

[5. User Guide 40](#_Toc53745453)

[5.1 Roles, Responsibilities 40](#_Toc53745454)

[5.2 Business Manager 40](#_Toc53745455)

[5.3 Storefront Functionality 41](#_Toc53745456)

[6 Known Issues 42](#_Toc53745457)

[7 Release History 43](#_Toc53745458)

# Summary

This document provides technical instructions for using the Cloudinary SalesForce Commerce Cloud Cartridge to integrate Cloudinary images and videos into an SFCC storefront.

**Glossary**

|  |  |
| --- | --- |
| **Term** | **Description** |
| SFCC | SalesForce Commerce Cloud |
| Business Manager (BM) | The primary tool used to configure the SFCC platform and customer storefront |
| SFRA | The SFCC default ecommerce storefront |
| Cloudinary Transformation | A set of parameters specifying how to change an image or video in terms of size, visual style, format, quality and more. For details, see the transformation guides for images (<https://cloudinary.com/documentation/image_transformations>) and videos (<https://cloudinary.com/documentation/video_manipulation_and_delivery>). |

# Component Overview

## Functional Overview

The Cloudinary cartridge enables images and videos available in Cloudinary to be used on the merchant Storefront.

## Limitations, Constraints

Use of the Cloudinary SFCC Cartridge requires credentials, keys and URLs from Cloudinary. Please go to the Cloudinary website to sign up for the services (<https://cloudinary.com/signup>).

## Compatibility

The cartridge is designed for Salesforce Commerce Cloud (SFRA) version 5.0.1. Compatibility Mode: 19.10.

The Cloudinary cartridge works with all locales.

## Privacy, Payment

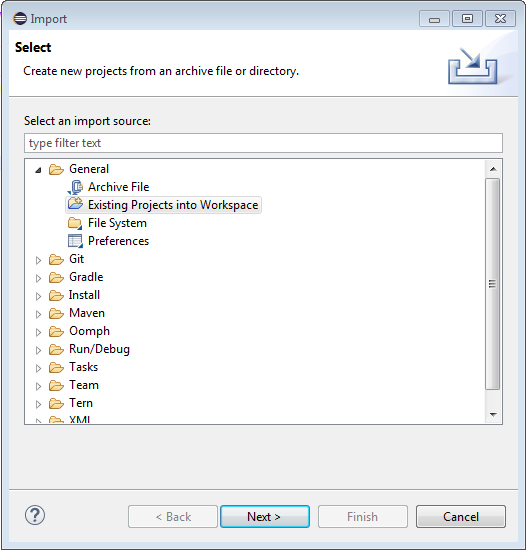
The Cloudinary cartridge does not collect and process any user profile information or billing information.

# Implementation Guide

## General Implementation

### 3.1.1 Installing the Cartridges on the sandbox

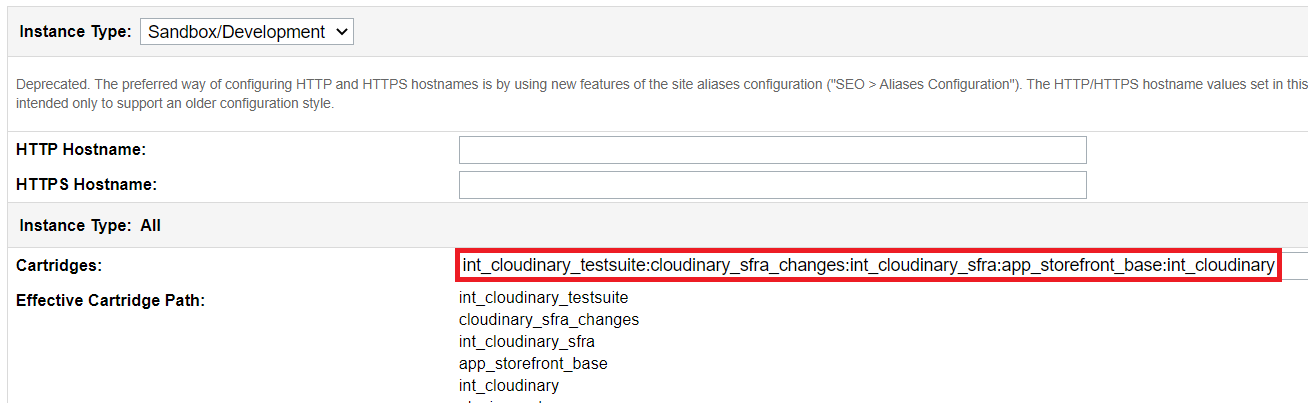
1. Download the int\_cloudinary\_sfra, int\_cloudinary, bm\_cloudinary and int\_cloudinary\_testsuite cartridges.
2. Go into the folder link\_cloudinary and run the following commands to build Cloudinary cartridges.
   1. npm run install
   2. npm run compile:js
3. Open Demandware Studio.
4. Import the downloaded cartridges from the Import dialog:



1. Link the cartridges to the sandbox. Select the sandbox connection and then select Properties. Select Project Reference and check in.

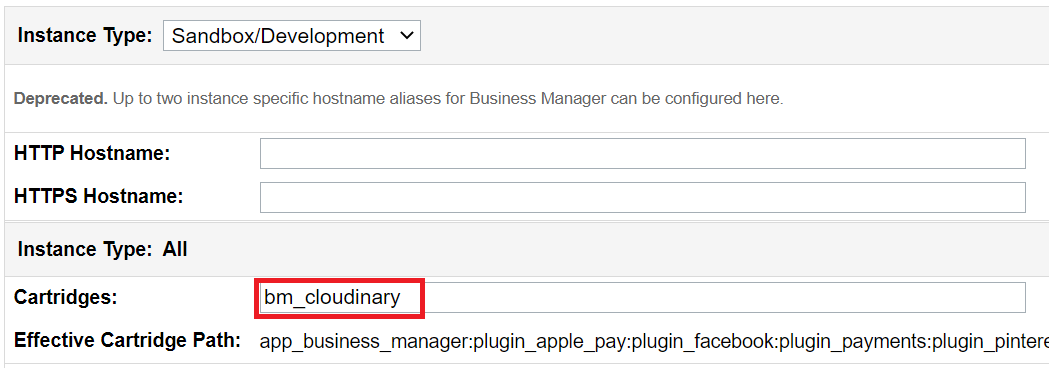
### 3.1.2 Setting Up Business Manager

1.  Go to Business Manager > Site > Manage Sites. Select the correct site, then select the  **Settings** tab. In the cartridge path add the int\_cloudinary, int\_cloudinary\_sfra and int\_cloudinary\_testsuite (optional) cartridges:

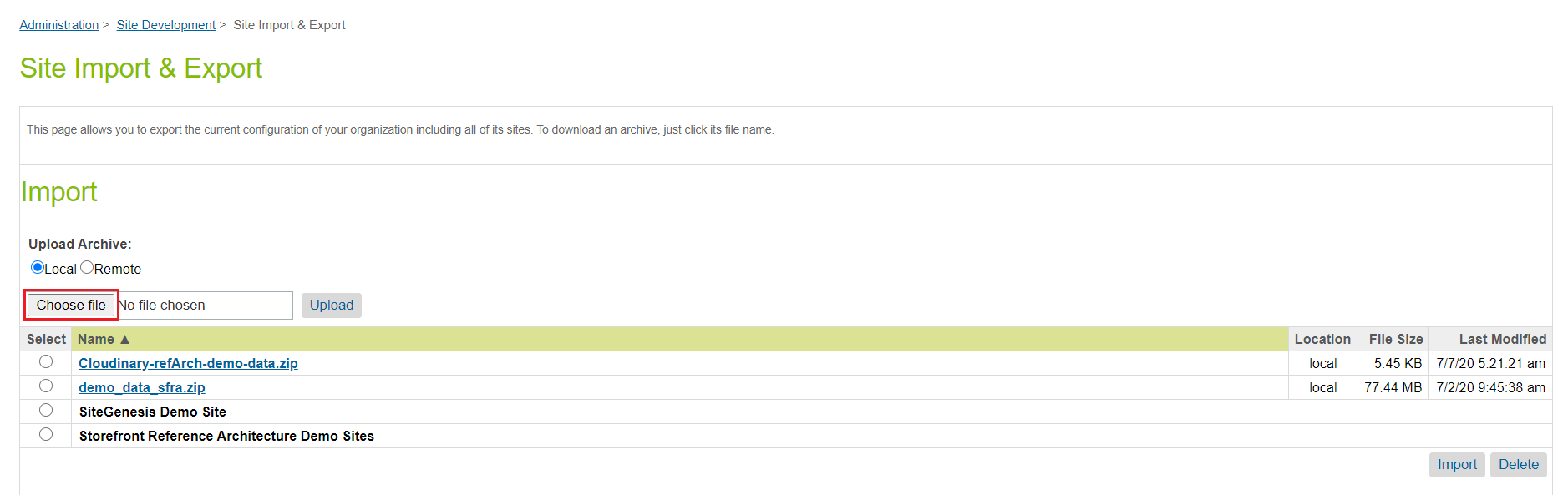


**Note:** Theint\_cloudinary\_testsuite cartridge is optional; it includes test harnesses for all script APIs. You may also include the cloudinary\_sfra\_changes cartridge if you want to see how everything is integrated on different pages, but it’s not recommended to use it on a production environment.

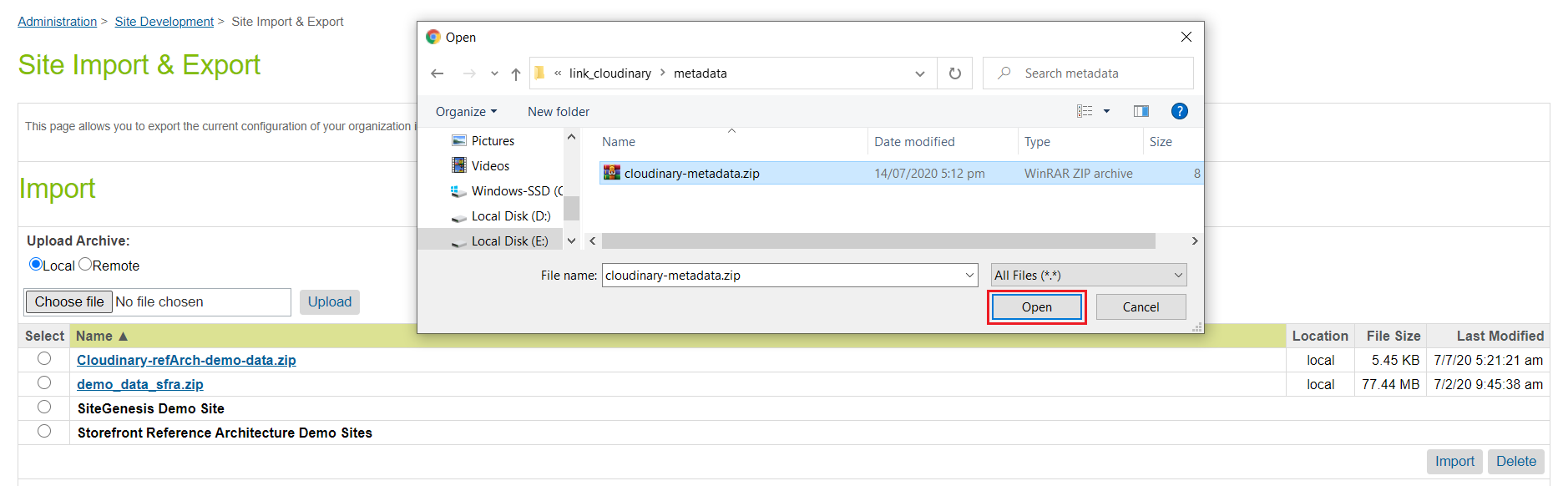
Add the bm\_Cloudinary cartridge to the cartridge path in Administration > Sites > Manage Sites > Business Manager - Settings:



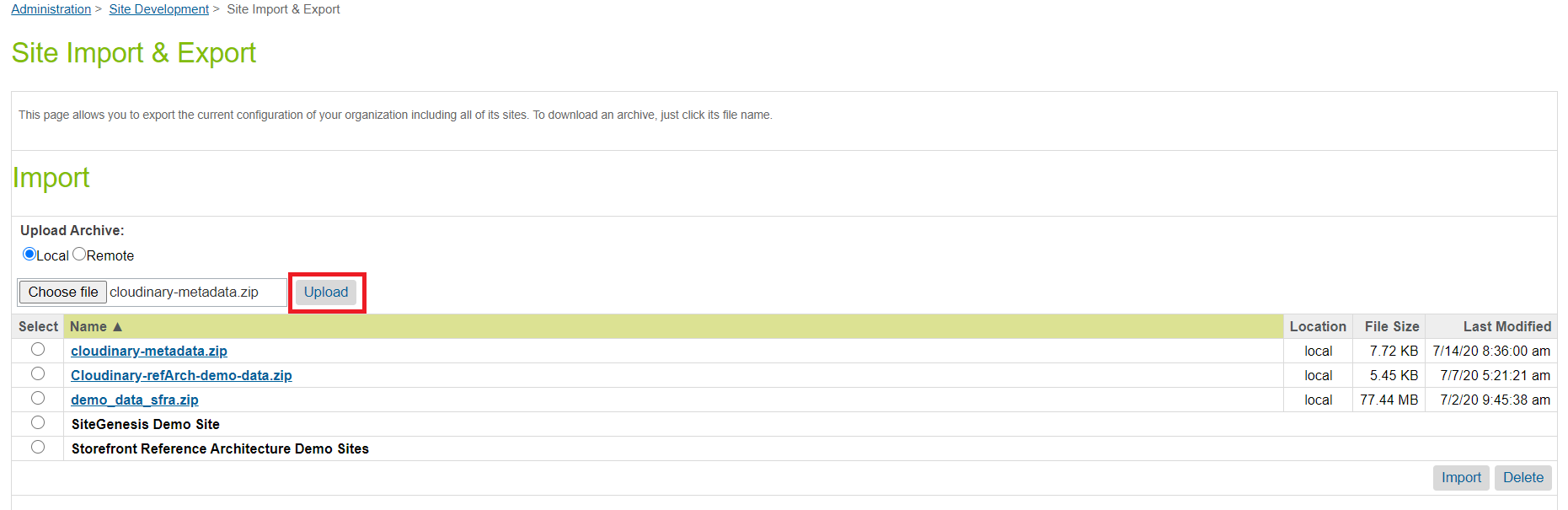
2.  Go to Administration > Site Development > Site Import & Export. In the Import section click on Choose File:



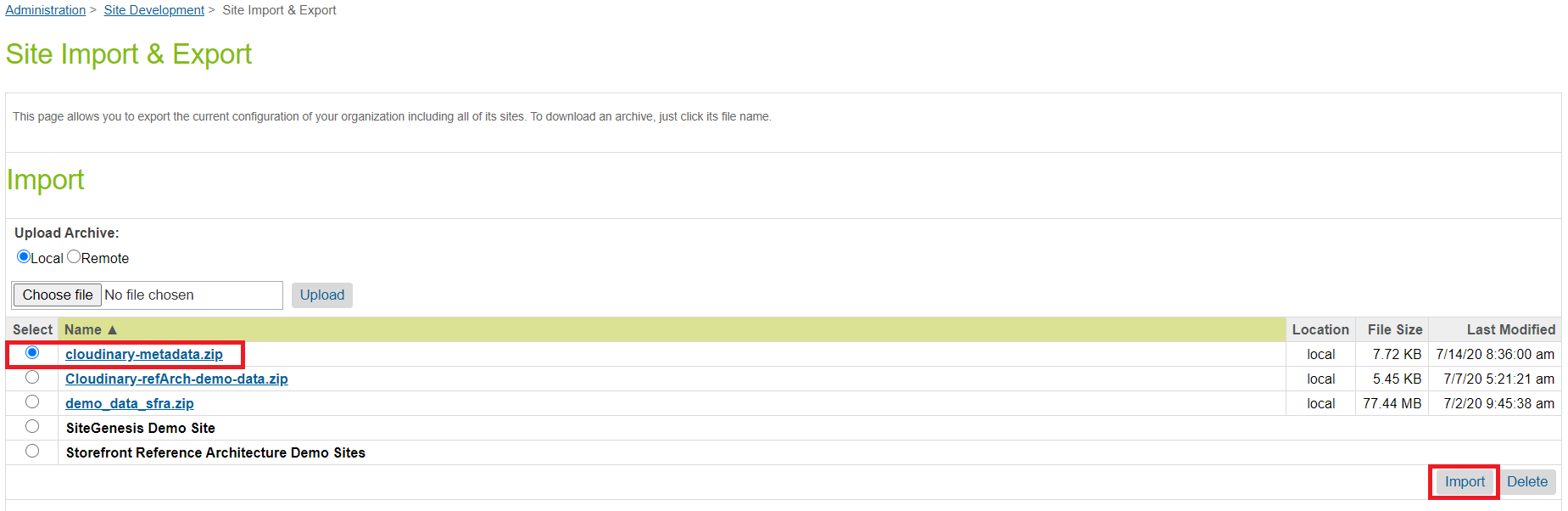
3.  Select cloudinary-metadata.zip in the metadata folder and click Open.



4. Click Upload.



5. Select cloudinary-metadata.zip in the Import section and click Import.



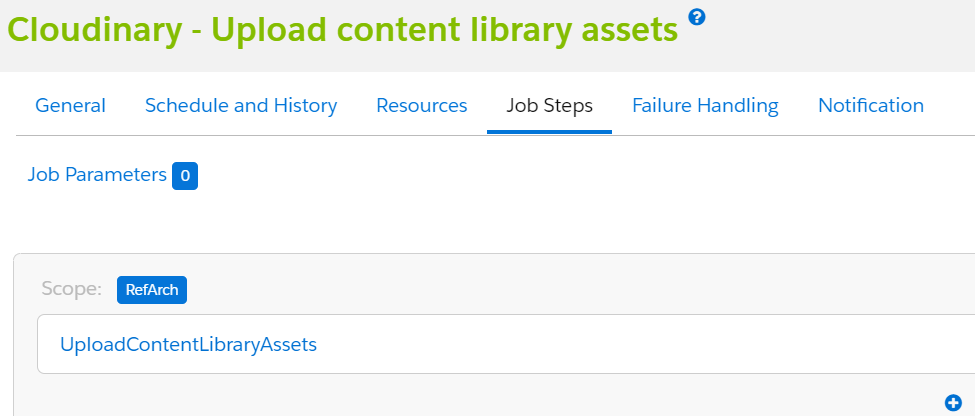
6. Click on OK.



**Note:** To avoid data errors while importing the cloudinary-metadata.zip file, Cloudinary related cartridges must be uploaded to the sandbox. If the code version is already active then reactivate the same code version when upload completes.

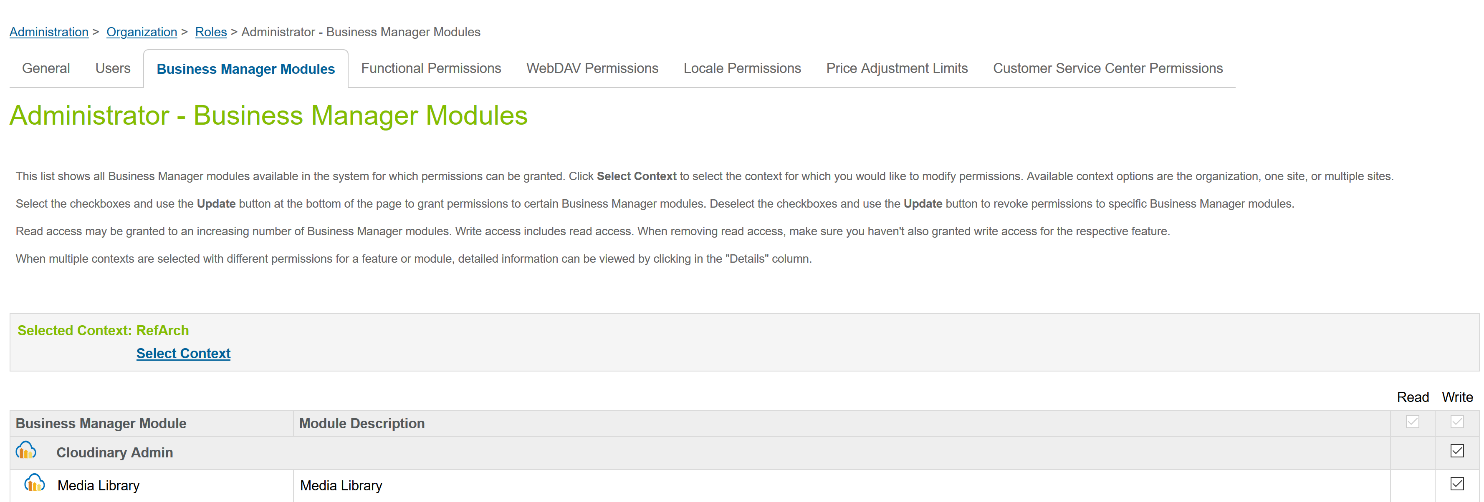
**Note**: There is a folder called “Sample content” inside “metadata” which contains sample data for specific sites. If you want to import sample data for SFRA then import the “RefArch.zip” file by following the above process.

7. In Administration > Operations > Job Schedules select any Cloudinary job e.g. “Cloudinary - Upload content library assets”. Go into the Job Steps tab and update the scope to the site name. Repeat this process for all other Cloudinary jobs listed in section [3.5](#_heading=h.4f1mdlm).



Note: For the “Cloudinary - Upload content library assets” job, We recommend assigning only one site if the library is shared across multiple sites; however, if it *is* shared across multiple sites the job will process that library content multiple times but not duplicate content in the Cloudinary DAM.

9. Go to Administration > Organization > Roles & Permissions and select the specific user role (Administrator). Select the “Business Manager Modules” tab, then from “Select Context” choose your site and click Apply. In the list of “Business Manager Modules” find “Cloudinary Admin” and click to update the permissions for Media Library and click “Update”.



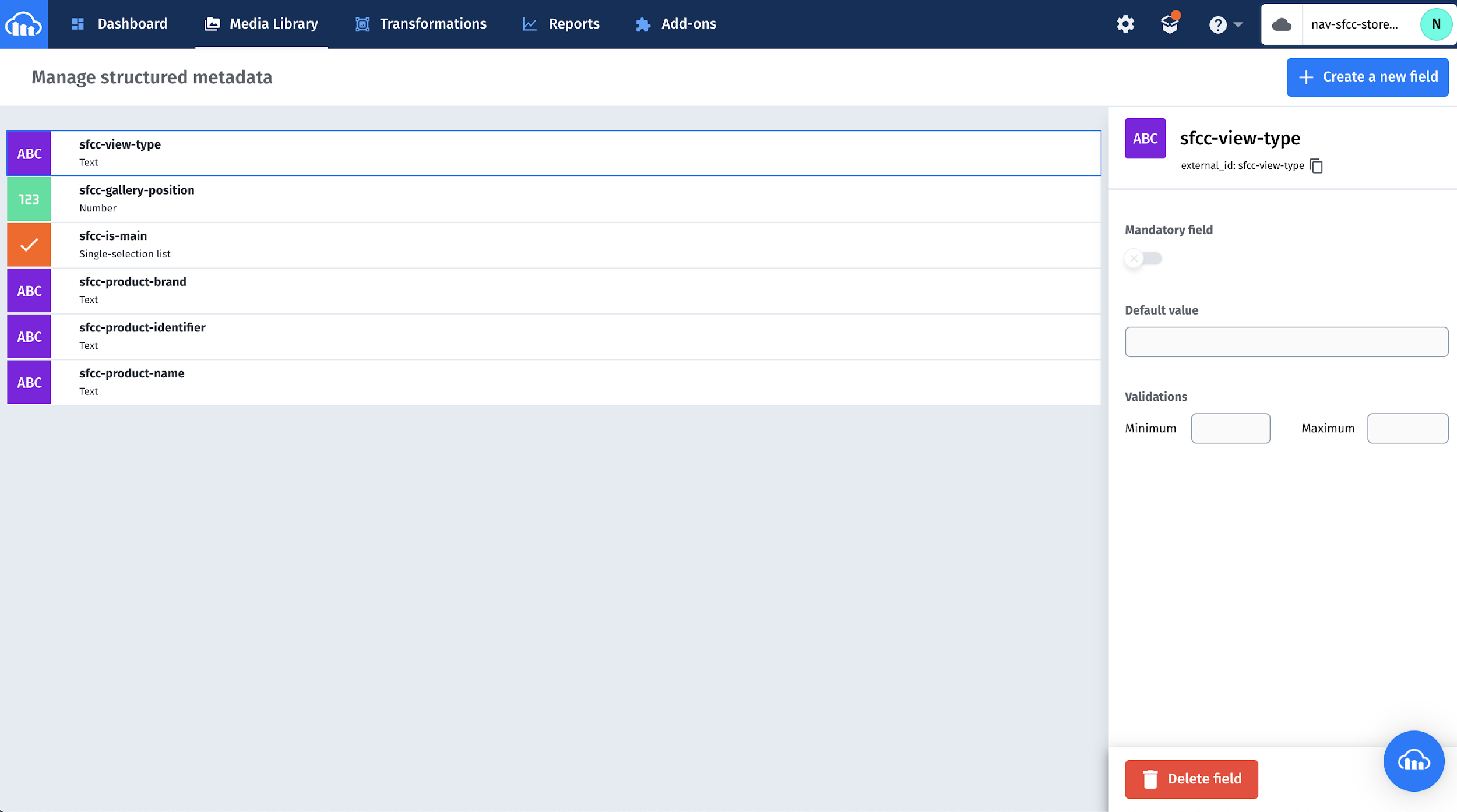
## 3.2 Configuring Cloudinary

### 3.2.1 Adding Structured Metadata in Cloudinary

Create the following structured metadata fields in your Cloudinary account. For instructions on creating structured metadata, see the Cloudinary documentation: <https://cloudinary.com/documentation/dam_media_library_administration#structured_metadata_management>.

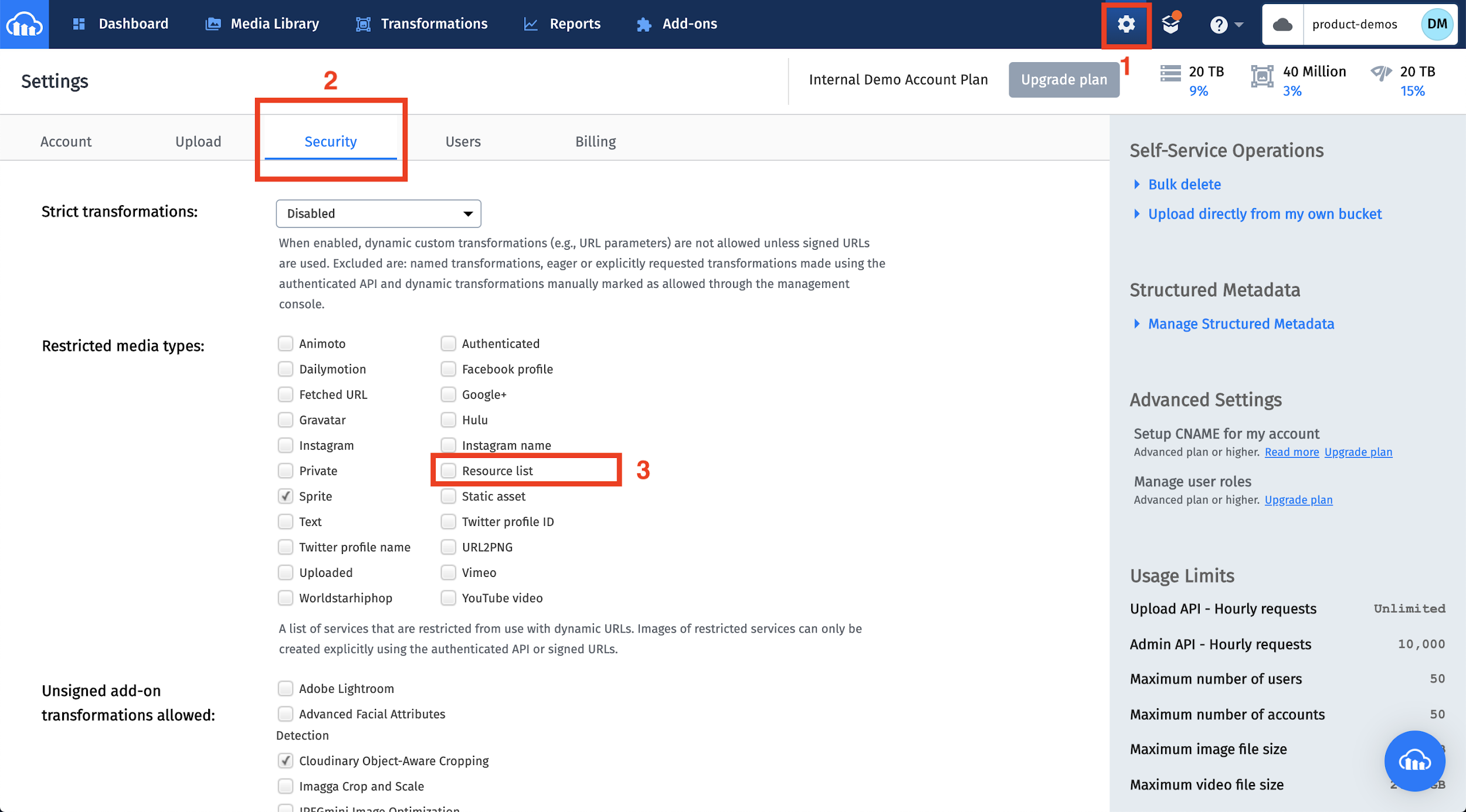
Note: Customize the external ID and the label of each field to match this table:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| External ID | Label | Type | Required? | Used for |
| sfcc-product-identifier | sfcc-product-identifier | Text | Yes | Stores the product ID. |
| sfcc-is-main | sfcc-is-main | Single-selection list  (Values: True, False) | Yes | Indicates which product image is used in the product listing. |
| sfcc-gallery-position | sfcc-gallery-position | Number | Yes | Indicates the position of the image in the product gallery. |
| sfcc-view-type | sfcc-view-type | Text | Yes | Stores the view type in SFCC. Required for swatches. |
| sfcc-product-name | sfcc-product-name | Text | Optional | Stores the product name, useful for searching in the DAM. |
| sfcc-product-brand | sfcc-product-brand | Text | Optional | Stores the product brand, useful for searching in the DAM. |



### 3.2.2 Enabling Resource Listing

In your account’s Security settings, enable public resource listing, by clearing the Resource list checkbox in Restricted media types.



## 3.3 Configuring the Cartridge Site Preferences:

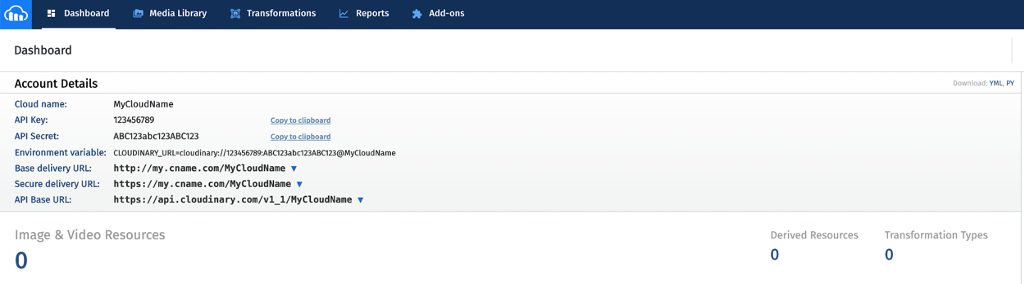
There are four Cloudinary custom preferences groups as follows:

**3.3.1 Cloudinary Core Configurations:**

Go to Business Manager: Sites > Your Site > Site Preferences > Custom Site Preferences > Cloudinary Core Configurations.

To find your Cloudinary account specific information:

1. Log in to the Cloudinary console at <https://cloudinary.com/console>. Use your Cloudinary credentials to log in.
2. Navigate to the Dashboard to see your Cloudinary account details.



|  |  |
| --- | --- |
| **Preference Name** | **Description** |
| Cloudinary Enabled | Enables or disables the Cloudinary cartridge for the site. |
| Cloudinary Base Delivery Path | The base delivery path configured for your Cloudinary account. |
| Cloudinary Cloud Name | The cloud name configured for your Cloudinary account. |
| Cloudinary API Key | The API key configured for your Cloudinary account. |
| Cloudinary API Secret | The API secret configured for your Cloudinary account. |
| Cloudinary Image Page Type Settings | The page types for Cloudinary responsive images on different pages. You can configure any page types of your choice but the ones shown below are mandatory for SFRA pages and these are hardcoded in the cartridge’s constants file.  {  "pdp": {},  "plp": {},  "searchSuggestions": {},  "cldPlpSwatch": {},  "cldPdpSwatch": {},  "cldProductNav": {},  "cart": {},  "quickview": {},  "miniCart": {},  "checkout": {},  "categoryBanner": {},  "recommendationTile": {},  "cldhomePageProductTile": {},  "cldCartProductTile": {}  }  Note: Although these page type keys are mandatory, you may keep them empty if you don’t want responsive images on specific pages. You may also define your own page types for your custom pages.  If you want to make images responsive on a page, for example, “plp”, then its json must have the following structure:  "plp": {  "automateResponsivenessWithJS": true,  "autoResponsiveDimensions": "w\_auto,c\_scale",  "dimensions": "w\_250",  "breakpoints": {  "size200w": {  "style": "(max-width: 480px) 200px",  "width": "200",  "height": "300"  },  "size113w": {  "style": "(max-width: 768px) 113px",  "width": "200",  "height": "150"  }  }  }  By using page types you can achieve responsiveness in two ways,  **(1)** **Generate html srcset and sizes attributes for <img> tag**  To enable this feature, set “automateResponsivenessWithJS” to false. It will then read configurations for all defined breakpoints and generate the required HTML attributes for the <img> tag. For this “dimensions” and ”breakpoints” are mandatory and you may exclude “autoResponsiveDimensions” from the settings.  **(2) Use Cloudinary responsiveness**  If you want to handle image responsiveness using Cloudinary client side libraries then set “automateResponsivenessWithJS” to true. For this “autoResponsiveDimensions” is mandatory and you may exclude “dimensions” and “breakpoints” from the settings.  Use these page types to make images responsive on different pages:   * **pdp**: product details page * **plp**: product tiles on the product listing page * **searchSuggestions**: suggested products under the search bar * **cldPlpSwatch**: swatch images on the product listing page * **cart**: product cards on the cart page * **quickview**: quick view modal * **miniCart**: mini cart drop down * **checkout**: all parts of the checkout page including shipping, billing and placing an order * **orderConfirmation**: order confirmation page |
| Cloudinary Core Shrinkwrap Static JS URL | The URL for the Cloudinary Shrinkwrap client side JS library. |
| Cloudinary Gallery Static JS URL | The URL for the Cloudinary Product Gallery Widget client side JS library. |
| Cloudinary Video Player Static JS URL | The URL for the Cloudinary Video Player client side JS library. |
| Cloudinary Video Player Static CSS URL | The URL for the Cloudinary Video Player client side CSS library. |
| Base folder for Catalog Images in SFCC | This is the base path for images in the SFCC WebDav for the Product catalog. Pick it from Catalog Settings > Image Settings > Base Path. |
| View type for high res images in the catalog | The view type used for the highest resolution of images in the catalog, e.g.: “large”. |
| View type for swatch images in the catalog | The view type used for swatches in the catalog, e.g.: “swatch”. |
| Folder for catalog images in Cloudinary | The folder in the Cloudinary DAM, where catalog images will be stored. |
| Folder For Content Images In Cloudinary | The folder in the Cloudinary DAM, where catalog images will be stored. |
| Folder For Product Images In Cloudinary | The folder in the Cloudinary DAM, where product images will be stored. |
| Folder For Product Videos In Cloudinary | The folder in the Cloudinary DAM, where product videos will be stored. |
| Cloudinary Rest Service Credentials ID | This is the credentials id used by Cloudinary rest service. See the [section 3.6](#_heading=h.2u6wntf) for more details. |

**3.3.2 Cloudinary Asset Management Configurations:**

Go to Business Manager: Sites > Your Site > Site Preferences > Custom Site Preferences > Cloudinary Asset Management Configurations.

|  |  |
| --- | --- |
| **Preference Name** | **Description** |
| Cloudinary Cartridge Operation Mode | Controls how assets are mapped to products. |
| Cloudinary Auto Upload Mapping | The folder configured in the Cloudinary DAM for auto upload mapping. For information on auto upload, see <https://cloudinary.com/documentation/fetch_remote_images#auto_upload_remote_resources>. |
| Cloudinary Cartridge Content Operation Mode | Controls how content assets are mapped to Cloudinary images and videos. |
| Use Cloudinary Gallery On PDPs | Enables or disables the Cloudinary Gallery Widget on PDPs. |
| Cloudinary Gallery Look And Feel | Controls the global look and feel for the Cloudinary gallery on PDPs.  Using the gallery studio generates config which looks like below <https://demo.cloudinary.com/product-gallery/?editor>  {  "container": "#cld-gallery",  "cloudName": "demo",  "mediaAssets": [{  "tag": "shoes\_product\_gallery\_demo",  "mediaType": "image"  }, {  "tag": "shoes\_product\_gallery\_demo",  "mediaType": "video"  }, {  "tag": "shoes\_product\_gallery\_demo\_spinset",  "mediaType": "spin"  }],  "displayProps": {  "mode": "expanded",  "columns": 2,  "spacing": 15  },  "aspectRatio": "3:4",  "transformation": {  "crop": "fill"  },  "bgColor": "transparent",  "carouselLocation": "left",  "carouselOffset": 10,  "navigation": "always",  "thumbnailProps": {  "mediaSymbolSize": 42,  "spacing": 20,  "width": 90,  "height": 90,  "navigationFloat": true,  "navigationShape": "square",  "navigationSize": 40,  "navigationColor": "#ffffff",  "selectedStyle": "border",  "selectedBorderPosition": "bottom",  "selectedBorderWidth": 4  },  "navigationButtonProps": {  "shape": "rectangle",  "iconColor": "#ffffff",  "color": "#000",  "size": 52,  "navigationPosition": "offset",  "navigationOffset": 12  },  "themeProps": {  "primary": "#000000",  "active": "#777777"  }  }  This custom preference expects this configuration without the cloudName and mediaAssets properties as shown below.  {  "container": "#cld-gallery",  "displayProps": {  "mode": "expanded",  "columns": 2,  "spacing": 15  },  "aspectRatio": "3:4",  "transformation": {  "crop": "fill"  },  "bgColor": "transparent",  "carouselLocation": "left",  "carouselOffset": 10,  "navigation": "always",  "thumbnailProps": {  "mediaSymbolSize": 42,  "spacing": 20,  "width": 90,  "height": 90,  "navigationFloat": true,  "navigationShape": "square",  "navigationSize": 40,  "navigationColor": "#ffffff",  "selectedStyle": "border",  "selectedBorderPosition": "bottom",  "selectedBorderWidth": 4  },  "navigationButtonProps": {  "shape": "rectangle",  "iconColor": "#ffffff",  "color": "#000",  "size": 52,  "navigationPosition": "offset",  "navigationOffset": 12  },  "themeProps": {  "primary": "#000000",  "active": "#777777"  }  } |
| Cloudinary Videos Enabled | Enables videos from Cloudinary on the site, globally. |
| Use Cloudinary Video Player | Enables the Cloudinary video player on the site. Cloudinary Videos Enabled should be true for this to work. |
| Cloudinary Include Videos By View Type In PGW | Includes videos fetched by view type in the Product Gallery, when Get Assets By View Type is chosen for Cloudinary Cartridge Operation Mode. |
| Cloudinary Video Player Style | Controls the global look and feel for the Cloudinary video player.  You can use the Cloudinary video studio (<https://studio.cloudinary.com/>) to generate the configuration, which includes the player style (highlighted here):  const player = cld.videoPlayer('player', {  "fluid": true,  "controls": true,  "fontFace": "Lobster",  "colors": {  "base": "#de1e1e",  "accent": "#180cff",  "text": "#222121"  }  });  player.source("elephants", {});  This custom preference expects the configuration for the player’s style only. Therefore, in this case, enter:    {  "fluid": true,  "controls": true,  "fontFace": "Lobster",  "colors": {  "base": "#de1e1e",  "accent": "#180cff",  "text": "#222121"  }  } |
| Cloudinary Use Video Custom Mapping | Enables or disables custom mapping for product videos in Cloudinary. If enabled, this overrides cartridge operation mode settings for videos. |
| Cloudinary Video Custom Map Path | The custom mapping path scheme for Cloudinary videos only, based on video naming conventions in Cloudinary. It is required if Cloudinary Use Video Custom Mapping is enabled. |
| Cloudinary Custom Mapping Video Format | Appends video format while building URLs by using custom mapping. |
| Product identifier used for tags in Cloudinary | The product attribute identifier which should be used to create tags in Cloudinary, e.g. standard attribute like ID or a custom attribute in the catalog. |
| Enable Cloudinary Swatches On PLP | Shows or hides swatch images from Cloudinary on PLP. |
| Cloudinary Metadata Filtering Endpoint | The Cloudinary dynamic endpoint to fetch resources based on metadata and tag name. |
| Cloudinary Structured Metadata Key | The Cloudinary structured metadata key representing a primary image. |
| Cloudinary Structured Metadata Value | The Cloudinary structured metadata value representing a primary image. |

**3.3.3 Cloudinary Transformation Configurations:**

Go to Business Manager: Sites > Your Site > Site Preferences > Custom Site Preferences > Cloudinary Transformation Configurations

|  |  |
| --- | --- |
| **Preference Name** | **Description** |
| Cloudinary Global Image Transformation | Transformations that are applied to all images delivered by Cloudinary. |
| Global Image Format | The default format for all images delivered by Cloudinary. |
| Global Image DPR | The default Device Pixel Ratio (DPR) setting for all images delivered by Cloudinary. |
| Global Image Quality | The default quality setting for all images delivered by Cloudinary. |
| Cloudinary Video Transformation | Transformations that are applied to all videos delivered by Cloudinary. |
| Global Video Format | The default format for all videos delivered by Cloudinary.  **Note:** if using cartridge operation mode 1 ([Map Assets using tags and metadata in Cloudinary](#_heading=h.19c6y18)), auto format (f\_auto) is not supported for videos. |
| Global Video Quality | The default quality setting for all videos delivered by Cloudinary. |

**3.3.4 Cloudinary Jobs Configurations:**

Go to Business Manager: Sites > Your Site > Site Preferences > Custom Site Preferences > Cloudinary Jobs Configurations

|  |  |
| --- | --- |
| **Preference Name** | **Description** |
| Content Library Job Last Execution Date | Stores the last date the ‘Cloudinary - Upload content library assets’ job ran. |
| Catalog Content Job Last Execution Date | Stores the last date the ‘Cloudinary - Upload catalog assets’ job ran. |
| Maximum upload limit for assets in MBs | Limits the size of assets uploaded to Cloudinary. Any assets larger than this limit are skipped. |
| Upload Preset | Sets the upload preset while uploading assets on Cloudinary DAM. This preset should exist in the Cloudinary DAM. Leave this empty if an upload preset is not required. |

## 3.4 System Object Types:

Cloudinary metadata contains custom attributes for System Object Types; these are added after successful import of metadata.

**3.4.1 Catalog Custom Attributes Details:**

Navigate to Administration > System Object Types > Catalog. After successful metadata import these attributes are added under *Cloudinary Site Cartridge*.

|  |  |
| --- | --- |
| **Attribute Name** | **Description** |
| Cloudinary Image Transformation | Transformations used to transform Cloudinary images. This attribute is used for two types of image assets.   1. **Catalog images:** When the cartridge tries to fetch the catalog images for category banners then this attribute is used to add image transformations specified on the catalog level in the URL. The transformations format for image transformations is **global\_image\_tranformations/catalog\_image\_tranformations/category\_image\_tranfromations**. If this attribute is empty then it skips adding catalog level image transformations in the URL.   **Note:** It doesn’t override global image transformations specified at the site level.   1. **Product Images:** While fetching product images used on different pages on the storefront, this attribute is used to add catalog level image transformations in the URL. This attribute for transformations will be overridden by transformations specified at the product level. |
| Cloudinary Video Transformation | Transformations used to transform Cloudinary videos. This attribute is used for two types of video assets.   1. **Catalog videos:** While fetching catalog videos, these transformations are used to add in the URL. The format for transformation is **global\_video\_transformations/catalog\_video\_transformations/category\_video\_transformation.** If this attribute is empty then it skips adding catalog level video transformations in the URL.   **Note:** It doesn’t override global video transformations specified at the site level.   1. **Product videos:** While fetching product videos used to display on PDPs, these transformations are used to add catalog level video transformations in the URL.This attribute for transformations will be overridden by transformations specified at the product level. |

**3.4.2 Category Custom Attributes Details:**

Navigate to Administration > System Object Types > Category. After successful metadata import, these attributes are added under *Cloudinary*.

|  |  |
| --- | --- |
| **Attribute Name** | **Description** |
| Cloudinary Image Transformation | Transformations used to transform Cloudinary images. When the cartridge tries to fetch the catalog images for category banners then this attribute is used to add image transformations specified on the category level in the URL. The format for image transformations is **global\_image\_tranformations/catalog\_image\_tranformations/category\_image\_tranfromations**.  If this attribute is empty then it skips adding category level image transformations in the URL.  **Note:** It doesn’t override global image transformations specified at the site level. |
| Cloudinary Video Transformation | Transformations used to transform Cloudinary videos. While fetching catalog videos, these transformations are used to add in the URL. The format for transformations is **global\_video\_transformations/cataog\_video\_transformations/category\_video\_transformation.**  If this attribute is empty then it skips adding category level video transformations in the URL.  **Note:** It doesn’t override global video transformations specified at the site level. |

**3.4.3 Product Custom Attributes Details:**

Navigate to Administration > System Object Types > Product. After successful metadata import these attributes are added under *Cloudinary Site Cartridge*.

|  |  |
| --- | --- |
| **Attribute Name** | **Description** |
| Cloudinary Tag Name | A tag configured in the Cloudinary DAM. It overrides the global setting for tag name specified on the site level. |
| Cloudinary Use Video | Enables or disables use of Cloudinary video. If we want to turn off the Cloudinary video for specific products then we can set it to No and the video will be disabled. It overrides the global setting specified on the site level. |
| Cloudinary Use Video Player | Enables or disables use of the Cloudinary video player. If we want to turn off the Cloudinary video player for specific products then we can set it to No and the video player will be disabled. It overrides the global setting specified on the site level. |
| Cloudinary video player options | Enables or disables Cloudinary video player options. It overrides the global setting specified on the site level. |
| Cloudinary Gallery Styles | Applies or removes Cloudinary gallery styles. It overrides the global setting specified on the site level. |
| Cloudinary Image Transformation | Transformations used to transform Cloudinary images. It overrides the image transformations specified on catalog level. |
| Cloudinary Video Transformation | Transformations used to transform Cloudinary videos. It overrides the video transformations specified on catalog level. |

**3.4.4 Library Custom Attributes Details:**

Navigate to Administration > System Object Types > Library. After successful metadata import these attributes are added under *Cloudinary Site Cartridge*.

|  |  |
| --- | --- |
| **Attribute Name** | **Description** |
| Cloudinary Image Transformation | Transformations used to transform Cloudinary images.While fetching library images, these transformations are used to add in the URL. The format for transformations is **global\_image\_transformations/library\_image\_transformations.**  **Note:** It doesn’t override global transformations specified on the site level. |
| Cloudinary Video Transformation | Transformations used to transform Cloudinary videos. While fetching library videos, these transformations are used to add in the URL. The format for transformations is **global\_video\_transformations/library\_video\_transformations.**  **Note:** It doesn’t override global transformations specified on the site level. |
| Cloudinary Use Video | Enables or disables Cloudinary video. If we want to turn off the Cloudinary video for a specific library then we can set it to No and the video will be disabled. It overrides the global setting specified on the site level. |
| Use the Cloudinary video player | Enables or disables use of the Cloudinary video player. If we want to turn off the Cloudinary video player for a specific library then we can set it to No and the video player will be disabled. It overrides the global setting specified on the site level. |
| Cloudinary default video player options | Options used by the Cloudinary video player. It overrides the global setting specified on the site level. |

## 3.5 Jobs Details:

Navigate to Administration > Jobs. After successful import of metadata, these jobs are added.

**Job:**

|  |  |
| --- | --- |
| **Name** | **Description** |
| Cloudinary - Delta Upload catalog assets | Uploads recently changed catalog assets from SFCC to Cloudinary. |

**Steps:**

|  |  |  |
| --- | --- | --- |
| **Job Step Name** | **Parameter Name** | **Description** |
| UploadCatalogAssets | CLDAssetRenameReportEmail | The email address used to receive the assets rename notification. |
|  | CLDNumberOfAssets | Specifies how many assets are uploaded to Cloudinary in Debug mode. |
|  | CLDJobExecutionMode | Configures the job execution mode. In Prod mode, all assets are uploaded. In Debug mode, the number of assets configured in CLDNumberOfAssets are uploaded. |
|  | CLDSyncMode | Specifies whether to run the job in Full or Delta mode. |

**Job:**

|  |  |
| --- | --- |
| **Name** | **Description** |
| Cloudinary - Delta upload library assets | Uploads recently changed content library asset files from SFCC to Cloudinary. |

**Steps:**

|  |  |  |
| --- | --- | --- |
| **Job Step Name** | **Parameter Name** | **Description** |
| UploadContentLibraryAssets | CLDAssetRenameReportEmail | The email address used to receive the assets rename notification. |
|  | CLDNumberOfAssets | Specifies how many assets are uploaded to Cloudinary in Debug mode. |
|  | CLDJobExecutionMode | Configures the job execution mode. In Prod mode, all assets are uploaded. In Debug mode, the number of assets configured in CLDNumberOfAssets are uploaded. |
|  | CLDSyncMode | Specifies whether to run the job in Full or Delta mode. |

**Job:**

|  |  |
| --- | --- |
| **Name** | **Description** |
| Cloudinary - Upload catalog assets | Uploads catalog assets from SFCC to Cloudinary. |

**Steps:**

|  |  |  |
| --- | --- | --- |
| **Job Step Name** | **Parameter Name** | **Description** |
| UploadCatalogAssets | CLDAssetRenameReportEmail | The email address used to receive the assets rename notification. |
|  | CLDNumberOfAssets | Specifies how many assets are uploaded to Cloudinary in Debug mode. |
|  | CLDJobExecutionMode | Configures the job execution mode. In Prod mode, all assets are uploaded. In Debug mode, the number of assets configured in CLDNumberOfAssets are uploaded. |
|  | CLDSyncMode | Specifies whether to run the job in Full or Delta mode. |

**Job:**

|  |  |
| --- | --- |
| **Name** | **Description** |
| Cloudinary - Upload content library assets | Uploads content library asset files from SFCC to Cloudinary. |

**Steps:**

|  |  |  |
| --- | --- | --- |
| **Job Step Name** | **Parameter Name** | **Description** |
| UploadContentLibraryAssets | CLDAssetRenameReportEmail | The email address used to receive the assets rename notification. |
|  | CLDNumberOfAssets | Specifies how many assets are uploaded to Cloudinary in Debug mode. |
|  | CLDJobExecutionMode | Configures the job execution mode. In Prod mode, all assets are uploaded. In Debug mode, the number of assets configured in CLDNumberOfAssets are uploaded. |
|  | CLDSyncMode | Specifies whether to run the job in Full or Delta mode. |

**Job:**

|  |  |
| --- | --- |
| **Name** | **Description** |
| Cloudinary - Upload product assets | Uploads product asset files from SFCC to Cloudinary. |

**Steps:**

|  |  |  |
| --- | --- | --- |
| **Job Step Name** | **Parameter Name** | **Description** |
| UplaodProductLargeImages | CLDAssetRenameReportEmail | The email address used to receive the assets rename notification. |
|  | CLDNumberOfAssets | Specifies how many assets are uploaded to Cloudinary in Debug mode. |
|  | CLDJobExecutionMode | Configures the job execution mode. In Prod mode, all assets are uploaded. In Debug mode, the number of assets configured in CLDNumberOfAssets are uploaded. |
|  | CLDSyncMode | Specifies whether to run the job in Full or Delta mode. |
|  | CLDCatalogIds | Specifies from which catalogs the assets are uploaded. If empty, then the catalog assigned to the site is used for uploading. |
|  | CLDViewType | Specifies which view type is uploaded to Cloudinary. |
| UploadProductSwatchImages | CLDAssetRenameReportEmail | The email address used to receive the assets rename notification. |
|  | CLDNumberOfAssets | Specifies how many assets are uploaded to Cloudinary in Debug mode. |
|  | CLDJobExecutionMode | Configures the job execution mode. In Prod mode, all assets are uploaded. In Debug mode, the number of assets configured in CLDNumberOfAssets are uploaded. |
|  | CLDSyncMode | Specifies whether to run the job in Full or Delta mode. |
|  | CLDCatalogIds | Specifies from which catalogs the assets are uploaded. If empty, then the catalog assigned to the site is used for uploading. |
|  | CLDViewType | Specifies which view type is uploaded to Cloudinary. |

Note: You can configure multiple job steps to upload assets from different view types; we have already configured two steps used to upload “large” and “swatch” view type images. For example, follow these steps to add a new job step to upload videos from the view type for videos, if it exists in the catalog.

1- Go to Administration -> Operations -> Jobs.

2- Select the job you want to change.

3- Select the Job Steps tab.

4- Click the + button to add a step.

5- Select ‘Custom.Cloudinary.UploadProductAssets’ and configure it.

6- Enter an ID for this step e.g ‘UploadProductVideos’.

7- Enter your email into the ‘CLDAssetRenameReportEmail’ field.

8- Select ‘CLDJobExecutionMode’ as required.

9- Select ‘CLDSyncMode’ as required.

10- Enter ‘video’ in ‘CLDViewType’ to associate this step with the video view type.

## 3.6 Services Details:

Navigate to Administration > Services. After successful import of metadata the following is added.

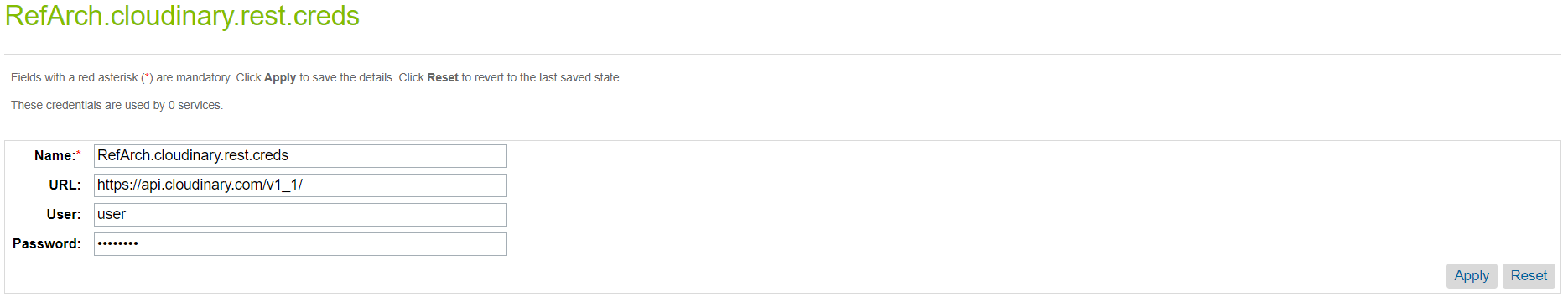
|  |  |  |
| --- | --- | --- |
| **Name** | **Profile** | **Credentials** |
| cloudinary.rest.upload | cloudinary.rest.upload.profile | Need to create site specific credentials. e.g. RefArch.cloudinary.rest.creds |
| cloudinary.rest.list | cloudinary.rest.list.profile | cloudinary.rest.list.creds |

Configure the URL, User and Password in RefArch.cloudinary.rest.creds and cloudinary.rest.list.creds. You can find all your account details in your Cloudinary console Dashboard, at <https://cloudinary.com/console>.

**Note:** The credentials for service ‘cloudinary.rest.upload’ are site specific. Make sure these credentials exist for the site. Then the credentials id should be placed in custom preferences (Cloudinary Rest Service Credentials ID).

For RefArch.cloudinary.rest.creds:

1. Enter your base upload API URL in the URL field.
2. Enter your API Key in the User field.
3. Enter your API Secret in the Password field.



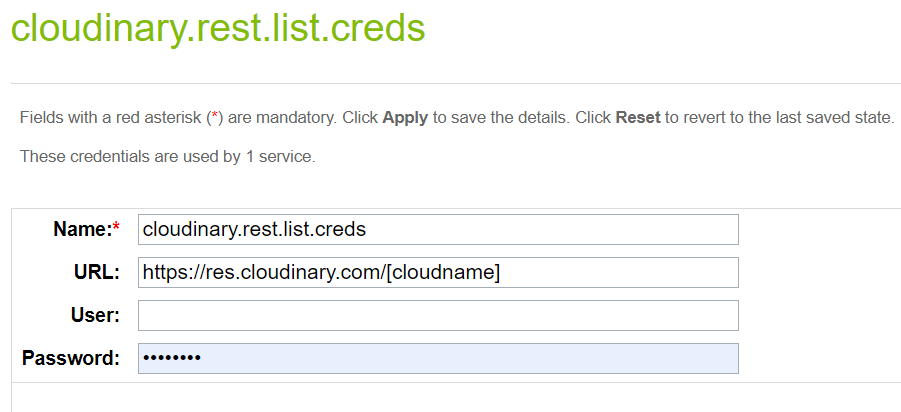
For cloudinary.rest.list.creds:

1. Enter your base delivery URL in the URL field.
   * If you have a private CDN, this will be of the form: https://<cloud\_name>-res.cloudinary.com
   * If you have a Custom Domain Name (CNAME), this will be of the form:

https://<custom domain name>

* + If it’s a simple Cloudinary list API URL then you also need to add a placeholder for cloudname i.e. [cloudname]: https://res.cloudinary.com/[cloudname]

1. The User and Password fields do not need to be set.



## 3.7 Cartridge Operation Modes:

The cartridge can operate in three different modes:

* + 1. **Map Assets using tags and metadata in Cloudinary:**

Note: if you intend to use this option, contact Cloudinary support (<https://support.cloudinary.com/hc/en-us>) to set up a special endpoint that will enable media to be delivered based on metadata and tags.

* Cloudinary is the single source of all media for all catalogs.
* For the product catalog, assets in Cloudinary are mapped to products using tags and structured metadata.
* For the content and site catalogs, specify URLs to Cloudinary assets in content slots.
* If migrating from an existing site, you can run jobs to bulk upload your media to Cloudinary. Tags and metadata are automatically added to the assets in Cloudinary for the product catalog.

**Note:** For product variations, only images are supported, not videos. Both are supported for the master product.

* + 1. **Map assets using view types in the catalog:**
* SFCC is the source of mapping for all media for all catalogs.
* For the product catalog, assets are expected to reside in Cloudinary using the same paths as in SFCC, so that assets are ultimately delivered from Cloudinary. You can run jobs to upload and synchronize your media residing in SFCC to Cloudinary.
* For the content and site catalogs, specify URLs to Cloudinary assets in content slots.
  + 1. **Use existing website URLs to Auto upload into Cloudinary and deliver:**
* Media is automatically uploaded from its existing location to Cloudinary, when requested by the end user, and subsequently delivered from Cloudinary. The auto-upload functionality is described here: <https://cloudinary.com/documentation/fetch_remote_images#auto_upload_remote_resources>
* This option is not recommended, but exists to provide a quick and simple way to deliver your media from Cloudinary.

## 3.8 Code Changes

In order to display Cloudinary widgets on different pages, you need to include Cloudinary custom templates and custom code in your templates. For reference you may use the cloudinary\_sfra\_changes cartridge.

**3.8.1 Include Cloudinary Static Libraries**

You need to include Cloudinary static client side JS/CSS libraries in your header to display widgets on different pages. Open the file named “htmlHead.isml” and place the following code at the bottom of this file.

1. **<iscomment>** Custom Start: include cloudinary static libs. **</iscomment>**
2. **<isinclude** template="include/cloudinaryStaticLibs"**/>**
3. **<iscomment>** Custom End: include cloudinary static libs. **</iscomment>**

**3.8.2 Include Cloudinary Gallery on PDP**

Open the “productDetails.isml” template and place the following code snippet where you want to display the Cloudinary Gallery. If Cloudinary is disabled, the images carousel defaults to SFCC.

1. **<iscomment>**Custom start: add cloudinary PGW**</iscomment>**
2. **<isif** condition="${pdict.cloudinary && pdict.cloudinary.isEnabled}"**>**
3. **<isinclude** template="product/components/cloudinaryGallery" **/>**
4. **<isset** name="isBundleOrSet" value="${false}" scope="page" **/>**
5. **<iselse>**
6. <!-- Product Images Carousel -->
7. **<isinclude** template="product/components/imageCarousel" **/>**
8. **</isif>**
9. **<iscomment>**Custom end: add cloudinary PGW**</iscomment>**

Note: You need to send all the required pdict variables used in the included templates from the controller to work the Product Gallery widget properly. For reference, take a look at the “Product-Show” controller being extended in the “int\_cloudinary\_sfra” cartridge.

**3.8.3 Include Cloudinary Video Player on PDP**

Open the “productDetails.isml” template and place the following code snippet where you want to display the Cloudinary Video Player.

1. **<iscomment>**Custom start: Add cloudinary video player**</iscomment>**
2. **<isif** condition="${pdict.cloudinary && pdict.cloudinary.isEnabled && pdict.cloudinary.videoEnabled}"**>**
3. **<isset** name="cldProductID" value="${product.id}" scope="page" **/>**
4. **<isinclude** template="experience/pages/cloudinaryVideo" **/>**
5. **</isif>**
6. **<iscomment>**Custom end: Add cloudinary video player**</iscomment>**

Note: you need to send all the required pdict variables used in the included templates from the controller to work the Cloudinary Video Player properly. For reference, take a look at the “Product-Show” controller being extended in the “int\_cloudinary\_sfra” cartridge.

**3.8.4 Load Product Swatch Images from Cloudinary on PDP**

Open the “variationAttribute.isml” template and place the following code to load product swatch images from Cloudinary. If Cloudinary is disabled, the swatch images default to SFCC.

1. **<iscomment>** Custom Start: Use cloudinary swatch images if it's enabled**</iscomment>**
2. **<isset** name="cldPageTypeQueryParam" value="" scope="page" **/>**
3. **<isif** condition="${cloudinary.isEnabled}"**>**
4. **<isset** name="swatchImageURL" value="${attrValue.images.cloudinaryProductSwatchImg ? attrValue.images.cloudinaryProductSwatchImg.url : ''}" scope="page" **/>**
5. **<isset** name="cldPageTypeQueryParam" value="${'&isBundleOrSet=' + isBundleOrSet}" scope="page" **/>**
6. **<iselse/>**
7. **<isset** name="swatchImageURL" value="${attrValue.images['swatch'].length > 0 ? attrValue.images['swatch'][0].url : ''}" scope="page" **/>**
8. **</isif>**
9. **<iscomment>** Custom Start: Use cloudinary swatch images if it's enabled**</iscomment>**

**3.8.5 Load Product Tile Image from Cloudinary on PLP**

Open the “productTileImage.isml” template and place the following code to load product tile images from Cloudinary. If Cloudinary is disabled, the product tile image defaults to SFCC.

1. **<iscomment>**Custom start: add cloudinary product image**</iscomment>**
2. **<isif** condition="${product.CLDEnabled && product.images && product.images.cloudinaryProductImage}"**>**
3. **<isset** name="isResponsive" value="${product.images.cloudinaryProductImage.isResponsive}" scope="page" **/>**
4. **<img** class="tile-image cld-responsive"
5. src="${product.images.cloudinaryProductImage.url}"
6. alt="${product.productName}"
7. title="${product.productName}"
8. data-src="${product.images.cloudinaryProductImage.url}"
9. **<isif** condition="${!isResponsive}"**>**
10. **<isif** condition="${!empty(product.images.cloudinaryProductImage.srcset)}"**>**
11. srcset="${product.images.cloudinaryProductImage.srcset}"
12. **</isif>**
13. **<isif** condition="${!empty(product.images.cloudinaryProductImage.sizes)}"**>**
14. sizes="${product.images.cloudinaryProductImage.sizes}"
15. **</isif>**
16. **</isif>**
17. **/>**
18. **<iselse/>**
19. **<img** class="tile-image"
20. src="${product.images.medium[0].url}"
21. alt="${product.productName}"
22. title="${product.images.medium[0].title}"
23. **/>**
24. **</isif>**
25. **<iscomment>**Custom end: add cloudinary product image**</iscomment>**

Note: This template can also be used in other places but make sure you send all required pdict variables used in this template from the respective controllers.

**3.8.6 Load Product Swatch Images from Cloudinary on PLP**

Open the “productTileSwatch.isml” template and place the following code to load product swatch images from Cloudinary. If Cloudinary is disabled, the product swatch images default to SFCC.

1. **<iscomment>** Custom Start: Use cloudinary swatch images if it's enabled**</iscomment>**
2. **<isif** condition="${product.CLDEnabled && swatch.images && swatch.images.cloudinaryProductSwatchImg}"**>**
3. **<isset** name="isResponsive" value="${swatch.images.cloudinaryProductSwatchImg.isResponsive}" scope="page" **/>**
4. **<img** class="swatch swatch-circle cld-responsive"
5. data-index="${loopState.count-1}"
6. style="background-image: url(${swatch.images.cloudinaryProductSwatchImg.url})"
7. src="${swatch.images.cloudinaryProductSwatchImg.url}"
8. alt="${swatch.images.swatch && !empty(swatch.images.swatch[0].alt) ? swatch.images.swatch[0].alt : product.productName}"
9. data-src="${swatch.images.cloudinaryProductSwatchImg.url}"
10. **<isif** condition="${!isResponsive}"**>**
11. **<isif** condition="${swatch.images.cloudinaryProductSwatchImg.srcset}"**>**
12. srcset="${swatch.images.cloudinaryProductSwatchImg.srcset}"
13. **</isif>**
14. **<isif** condition="${swatch.images.cloudinaryProductSwatchImg.sizes}"**>**
15. sizes="${swatch.images.cloudinaryProductSwatchImg.sizes}"
16. **</isif>**
17. **</isif>**
18. **<iselse/>**
19. **<img** class="swatch swatch-circle"
20. data-index="${loopState.count-1}"
21. style="background-image: url(${swatch.images.swatch[0].url})"
22. src="${swatch.images.swatch[0].url}"
23. alt="${swatch.images.swatch[0].alt}"**/>**
24. **</isif>**
25. **<iscomment>** Custom End: Use cloudinary swatch images if it's enabled**</iscomment>**

Note: This template can be used on other pages but make sure you send all required pdict variables from the respective controllers.

**3.8.7 Include Cloudinary Gallery on Quick View**

Open the “quickView.isml” template and place the following code to load the Cloudinary Gallery in the quick view modal. If Cloudinary is disabled, the product image carousel defaults to SFCC.

1. **<iscomment>**Custom start: add cloudinary PGW**</iscomment>**
2. **<isif** condition="${pdict.cloudinary.isEnabled}"**>**
3. **<div** class="cloudinary-data-container d-none" data-cloudinary="<isprint value="${JSON.stringify(pdict.cloudinary)}" encoding="htmlunquote" **/>**"**></div>**
4. **<isset** name="isBundleOrSet" value="${false}" scope="page" **/>**
5. **<div** class="primary-images col-12 col-sm-6"**>**
6. **<div** id="cld-gallery"**></div>**
7. **</div>**
8. **<iselse>**
9. <!-- Product Images Carousel -->
10. **<isinclude** template="product/components/imageCarousel" **/>**
11. **</isif>**
12. **<iscomment>**Custom end: add cloudinary PGW**</iscomment>**

**3.8.8 Include Cloudinary Gallery/Video player on Product Set Page**

Open the “setDetails.isml” template and place the following code to load the Cloudinary Gallery on the product set details page. If Cloudinary is disabled, the product image carousel defaults to SFCC.

1. **<iscomment>**Custom start: add variable for cloudinary PGW**</iscomment>**
2. **<isif** condition="${pdict.cloudinary && pdict.cloudinary.isEnabled}"**>**
3. **<isset** name="isBundleOrSet" value="${true}" scope="page" **/>**
4. **</isif>**
5. **<iscomment>**Custom end: add variable for cloudinary PGW**</iscomment>**

Add video player

1. **<iscomment>**Custom start: Add cloudinary video player**</iscomment>**
2. **<isif** condition="${pdict.cloudinary && pdict.cloudinary.isEnabled && pdict.cloudinary.videoEnabled}"**>**
3. **<isset** name="cldProductID" value="${product.id}" scope="page" **/>**
4. **<isinclude** template="experience/pages/cloudinaryVideo" **/>**
5. **</isif>**
6. **<iscomment>**Custom end: Add cloudinary video player**</iscomment>**

Then open the “setItems.isml” template and place following code snippet:

1. **<iscomment>**Custom start: add cloudinary PGW**</iscomment>**
2. **<isif** condition="${pdict.cloudinary && pdict.cloudinary.isEnabled}"**>**
3. **<isinclude** template="product/components/cloudinaryProductSetGallery"**/>**
4. **<iselse/>**
5. <!-- Product Images Carousel -->
6. **<isinclude** template="product/components/imageCarousel" **/>**
7. **</isif>**
8. **<iscomment>**Custom end: add cloudinary PGW**</iscomment>**

**3.8.9 Include Cloudinary Gallery/Video player on Product Bundle Page**

Open the “bundleDetails.isml” template and place the following code to load the Cloudinary Gallery on the product bundle details page. If Cloudinary is disabled, the product image carousel defaults to SFCC.

1. **<iscomment>**Custom start: add variable for cloudinary PGW**</iscomment>**
2. **<isif** condition="${pdict.cloudinary && pdict.cloudinary.isEnabled}"**>**
3. **<isset** name="isBundleOrSet" value="${true}" scope="page" **/>**
4. **</isif>**
5. **<iscomment>**Custom end: add variable for cloudinary PGW**</iscomment>**
6. **<iscomment>**Custom start: add cloudinary PGW**</iscomment>**
7. **<isif** condition="${pdict.cloudinary && pdict.cloudinary.isEnabled}"**>**
8. **<isinclude** template="product/components/cloudinaryGallery" **/>**
9. **<iselse>**
10. <!-- Product Images Carousel -->
11. **<isinclude** template="product/components/imageCarousel" **/>**
12. **</isif>**
13. **<iscomment>**Custom end: add cloudinary PGW**</iscomment>**
14. **<iscomment>**Custom start: add cloudinary video player**</iscomment>**
15. **<isif** condition="${pdict.cloudinary && pdict.cloudinary.isEnabled && pdict.cloudinary.videoEnabled}"**>**
16. **<isset** name="cldProductID" value="${product.id}" scope="page" **/>**
17. **<isinclude** template="experience/pages/cloudinaryVideo" **/>**
18. **</isif>**
19. **<iscomment>**Custom end: add cloudinary video player**</iscomment>**

Then open the “bundleItem.isml” template to add the following code snippet.

1. **<iscomment>**Custom start: add cloudinary PGW**</iscomment>**
2. **<isif** condition="${pdict.cloudinary && pdict.cloudinary.isEnabled}"**>**
3. **<isinclude** template="product/components/cloudinaryProductSetGallery"**/>**
4. **<iselse/>**
5. <!-- Product Images Carousel -->
6. **<isinclude** template="product/components/imageCarousel" **/>**
7. **</isif>**
8. **<iscomment>**Custom end: add cloudinary PGW**</iscomment>**

**3.8.10 Load Category Banner Image from Cloudinary on PLP**

Open the “searchResultsNoDecorator.isml” template and place the following code to load a category banner image from Cloudinary. If Cloudinary is disabled, the banner image defaults to SFCC.

1. **<iscomment>**Custom start: add cloudinary category banner image**</iscomment>**
2. **<isif** condition="${pdict.productSearch.cldBannerImage && pdict.productSearch.cldBannerImage.url}"**>**
3. style="background-image: url(${pdict.productSearch.cldBannerImage.url})"
4. **<iselseif** condition="${pdict.productSearch.bannerImageUrl}" **/>**
5. style="background-image: url(${pdict.productSearch.bannerImageUrl})"
6. **</isif>**
7. **<iscomment>**Custom end: add cloudinary category banner image**</iscomment>**

**3.8.11 Load Suggested Product Image from Cloudinary on Search**

Open the “suggestions.isml” template and place the following code to load the suggested product image from Cloudinary. If Cloudinary is disabled, the product image defaults to SFCC.

1. **<iscomment>**Custom strat: add clouidnary img**</iscomment>**
2. **<isif** condition="${!empty(product.cloudinaryProductImage)}"**>**
3. **<isset** name="isResponsive" value="${product.cloudinaryProductImage.isResponsive}" scope="page" **/>**
4. **<img** class="swatch-circle hidden-xs-down cld-responsive"
5. alt="${product.name}"
6. src="${product.cloudinaryProductImage.url}"
7. data-src="${product.cloudinaryProductImage.url}"
8. **<isif** condition="${!isResponsive}"**>**
9. **<isif** condition="${product.cloudinaryProductImage.srcset}"**>**
10. srcset="${product.cloudinaryProductImage.srcset}"
11. **</isif>**
12. **<isif** condition="${product.cloudinaryProductImage.sizes}"**>**
13. sizes="${product.cloudinaryProductImage.sizes}"
14. **</isif>**
15. **</isif>**
16. **/>**
17. **<iselse/>**
18. **<img** class="swatch-circle hidden-xs-down" alt="${product.name}" src="${product.imageUrl}" **/>**
19. **</isif>**
20. **<iscomment>**Custom end: add clouidnary img**</iscomment>**

**3.8.12 Load Product Image from Cloudinary on Checkout & Order Confirmation pages**

Open the “productCard.isml” template and place the following code to load the product image from Cloudinary on checkout and order confirmation pages. If Cloudinary is disabled, then the product image defaults to SFCC.

1. **<iscomment>**Custom start: add cloudinary product image**</iscomment>**
2. **<isif** condition="${pdict.cloudinary && pdict.cloudinary.isEnabled && lineItem.images.cloudinaryProductImage}"**>**
3. **<isset** name="isResponsive" value="${lineItem.images.cloudinaryProductImage.isResponsive}" scope="page" **/>**
4. **<img** class="product-image cld-responsive"
5. src="${lineItem.images.cloudinaryProductImage.url}"
6. alt="${lineItem.images.small[0].alt}"
7. title="${lineItem.images.small[0].title}"
8. data-src="${lineItem.images.cloudinaryProductImage.url}"
9. **<isif** condition="${!isResponsive}"**>**
10. **<isif** condition="${!empty(lineItem.images.cloudinaryProductImage.srcset)}"**>**
11. srcset="${lineItem.images.cloudinaryProductImage.srcset}"
12. **</isif>**
13. **<isif** condition="${lineItem.images.cloudinaryProductImage.sizes}"**>**
14. sizes="${lineItem.images.cloudinaryProductImage.sizes}"
15. **</isif>**
16. **</isif>**
17. **>**
18. **<iselse/>**
19. **<img** class="product-image"
20. src="${lineItem.images.small[0].url}"
21. alt="${lineItem.images.small[0].alt}"
22. title="${lineItem.images.small[0].title}"**>**
23. **</isif>**
24. **<iscomment>**Custom end: add cloudinary product image**</iscomment>**

**3.8.13 Load Bundle Product Image from Cloudinary on Cart Page**

Open the “cartProductCardBundledItems.isml” template and place the following code to load the bundle product image from Cloudinary on the cart page. If Cloudinary is disabled, the bundle product image defaults to SFCC.

1. **<iscomment>**Custom start: add cloudinary product image**</iscomment>**
2. **<isif** condition="${pdict.cloudinary && pdict.cloudinary.isEnabled && bundledLineItem.images.cloudinaryProductImage}"**>**
3. **<isset** name="isResponsive" value="${bundledLineItem.images.cloudinaryProductImage.isResponsive}" scope="page" **/>**
4. **<img** class="product-image cld-responsive"
5. src="${bundledLineItem.images.cloudinaryProductImage.url}"
6. alt="${bundledLineItem.images.small[0].alt}"
7. title="${bundledLineItem.images.small[0].title}"
8. data-src="${bundledLineItem.images.cloudinaryProductImage.url}"
9. **<isif** condition="${!isResponsive}"**>**
10. **<isif** condition="${bundledLineItem.images.cloudinaryProductImage.srcset}"**>**
11. srcset="${bundledLineItem.images.cloudinaryProductImage.srcset}"
12. **</isif>**
13. **<isif** condition="${bundledLineItem.images.cloudinaryProductImage.sizes}"**>**
14. sizes="${bundledLineItem.images.cloudinaryProductImage.sizes}"
15. **</isif>**
16. **</isif>**
17. **>**
18. **<iselse/>**
19. **<img** class="product-image"
20. src="${bundledLineItem.images.small[0].url}"
21. alt="${bundledLineItem.images.small[0].alt}"
22. title="${bundledLineItem.images.small[0].title}"
23. **>**
24. **</isif>**
25. **<iscomment>**Custom end: add cloudinary product image**</iscomment>**

**3.8.14 Load Product Image from Cloudinary on Cart Page**

Open the “cartProductCard.isml” template and place the following code to load the product image from Cloudinary on the cart page. If Cloudinary is disabled, the product image defaults to SFCC.

1. **<iscomment>**Custom start: add cloudinary product image**</iscomment>**
2. **<isif** condition="${pdict.cloudinary && pdict.cloudinary.isEnabled && lineItem.images.cloudinaryProductImage}"**>**
3. **<isset** name="isResponsive" value="${lineItem.images.cloudinaryProductImage.isResponsive}" scope="page" **/>**
4. **<img** class="product-image cld-responsive"
5. src="${lineItem.images.cloudinaryProductImage.url}"
6. alt="${lineItem.images.small[0].alt}"
7. title="${lineItem.images.small[0].title}"
8. data-src="${lineItem.images.cloudinaryProductImage.url}"
9. **<isif** condition="${!isResponsive}"**>**
10. **<isif** condition="${lineItem.images.cloudinaryProductImage.srcset}"**>**
11. srcset="${lineItem.images.cloudinaryProductImage.srcset}"
12. **</isif>**
13. **<isif** condition="${lineItem.images.cloudinaryProductImage.sizes}"**>**
14. sizes="${lineItem.images.cloudinaryProductImage.sizes}"
15. **</isif>**
16. **</isif>**
17. **>**
18. **<iselse/>**
19. **<img** class="product-image" src="${lineItem.images.small[0].url}" alt="${lineItem.images.small[0].alt}" title="${lineItem.images.small[0].title}"**>**
20. **</isif>**
21. **<iscomment>**Custom end: add cloudinary product image**</iscomment>**

**3.8.15 Load Library Content Image from Cloudinary on Content Pages**

Open any content asset where you want to load images from Cloudinary, for example open the “about-us” content asset and place the following code to load a top hero banner background image from Cloudinary.

$Include('Cloudinary-GetContentImage', 'url', 'images/AboutUsBanner/aboutus.jpg', 'pageType', 'aboutUs', 'isUrl', true)$

The query params are as follows:

**url**: The content image relative URL

**pageType**: Used to load responsive images include srcset and sizes attributes

**isURL**: Indicates either load only transformed URL or image tag having all srcset and sizes attributes.

**3.8.16 Include Cloudinary Video Player on Content Pages**

Open any content asset where you want to load images from Cloudinary, for example open the “about-us” content asset and place the following code to load the Cloudinary Video Player.

$Include('Cloudinary-GetContentVideo', 'url', 'videos/cat.mp4')$

Here, url is the video relative URL present in your static content library.

**3.8.17 Cloudinary Script APIs**

The cartridge includes low level script APIs that are used by the cartridge itself to fetch assets from Cloudinary and apply transformations on the URLs. Integrators may use them to customize the cartridge according to their needs, or make use of them in their own custom cartridges. Each of the methods has documentation in the code that explains the method’s usage.

Here are the details about the script APIs:

1. **cloudinaryApi:** Used to communicate directly with Cloudinary APIs to fetch different types of assets and apply transformations on the URLs.
2. **cloudinaryTranformationApi:** Used to provide transformations specified on several levels for different types of assets.

**3.8.18 Site version for Cloudinary Test Suite**

There is a cartridge int\_cloudinary\_testsuite which is used to test script APIs that directly communicate with Cloudinary web services.

To make it work with any Commerce Cloud version either it’s SFRA or SiteGenesis, you need to update a constant present inside cloudinaryConstants.js file accordingly. The constant name and possible values are as follows.

|  |  |  |
| --- | --- | --- |
| **Commerce Cloud Site Version** | **Constant Name** | **Value** |
| SFRA | SITE\_VERSION\_FOR\_TEST\_SUITE | sfra |
| SiteGenesis | SITE\_VERSION\_FOR\_TEST\_SUITE | sitegenesis |

**Note:** By default, it’s set to “sfra”.

## 3.9 Managing Videos

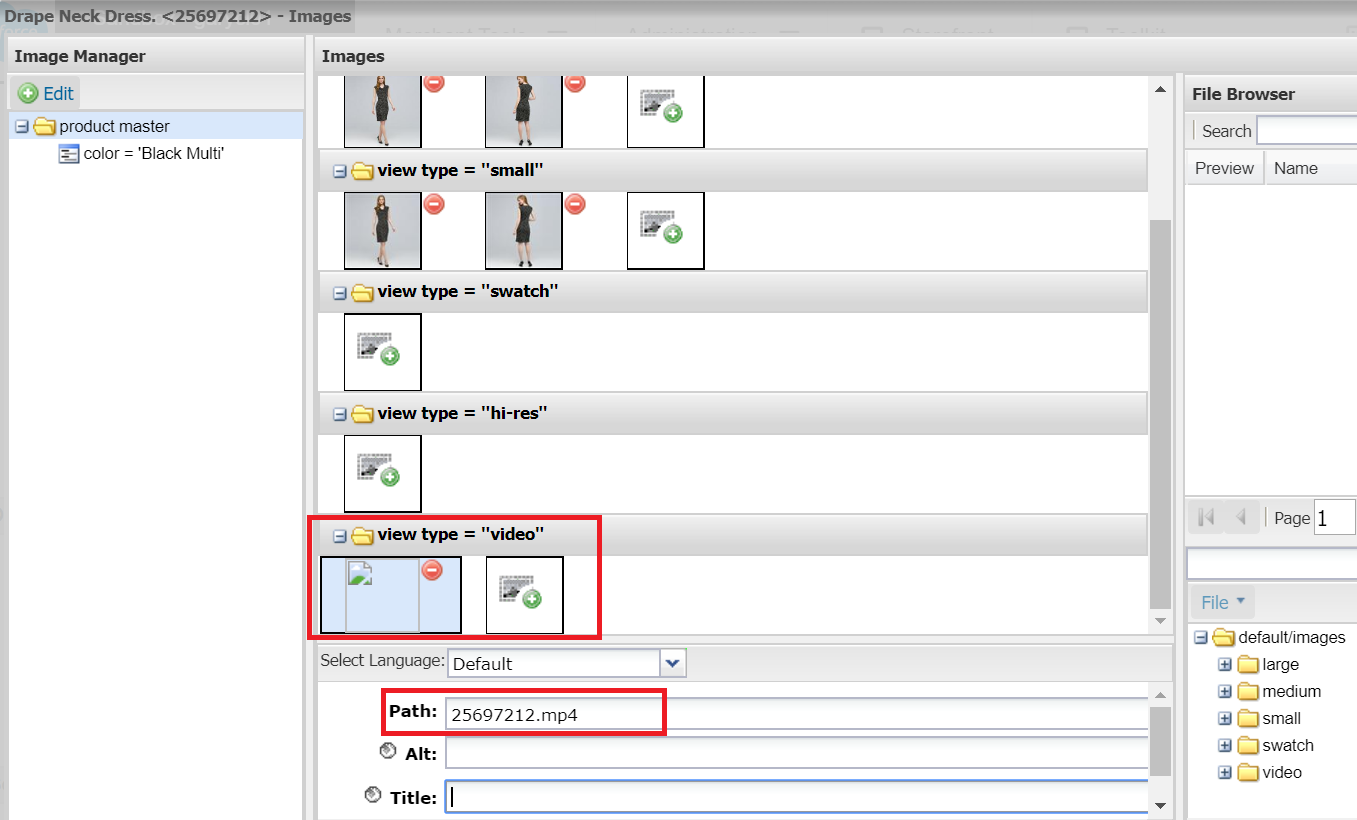
There are two ways to make videos available for products,

**3.9.1 By Using SFCC view type:**

First, create a view type at the catalog level by navigating through the following path **Merchant Tools > Products and Catalogs > Catalogs > (Your Catalog) > Image Settings.** Against **View Types** you can see there are already standard view types provided by SFCC. Click on the **Add View Type** link to add a new view type with name, “video”.



After adding the view type, go to any product and click on the edit button to edit product images. Notice the new view type in the pop-up window with the name, “video”. Click on the icon to assign video to this view type for this product and in the field “path”, add the public ID of a video that is already present in your Cloudinary DAM.



**Note**: At present, only one video can be assigned to the view type, so if you upload multiple videos then it would pick the first one.

* + 1. **By using custom mapping**

Custom mapping assumes the video is already present in your Cloudinary DAM. Define the custom mapping scheme using the site level preference, “Cloudinary Video Custom Map Path”.

Note: Make sure you have enabled custom mapping for videos by using site preference, “Cloudinary Use Video Custom Mapping”.

# 4. Operations, Maintenance

## 4.1 Data Storage

*Intentionally left blank.*

## 4.2 Availability

Cloudinary services are based on a highly available infrastructure designed for 100% uptime. The status of Cloudinary’s services can be found on our service status page -<https://status.cloudinary.com/>.  From here, you can subscribe to service status updates.  In the unlikely event that Cloudinary services go down, storefronts will continue to display any assets previously requested from Cloudinary since they are cached on the edge using CDNs.  However, assets that are being requested by end users for the very first time will not be displayed by storefronts.  Please contact the Cloudinary support team to address such incidents.

## 4.3 Support

Contact our support team if you have any issues with the cartridge.

<https://support.cloudinary.com/hc/en-us>

## 4.4 Cloudinary Workflow

Workflow Steps:

1. Set up images on your staging environment through the Sync Jobs. Make sure job executions are complete before doing any production replication.
2. Test the image functionality on the staging storefront. There may be a time gap where missing images appear on your staging environment while the job is executing.
3. If images appear correctly on staging, replicate to production.

# 5. User Guide

## 5.1 Roles, Responsibilities

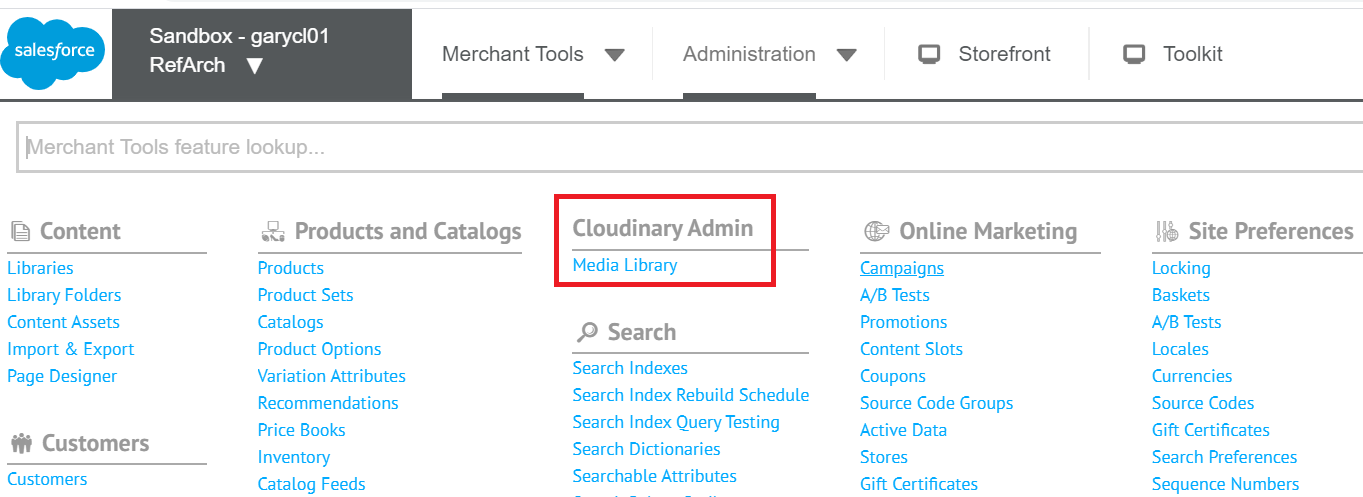
*Intentionally left blank.*

## 5.2 Business Manager

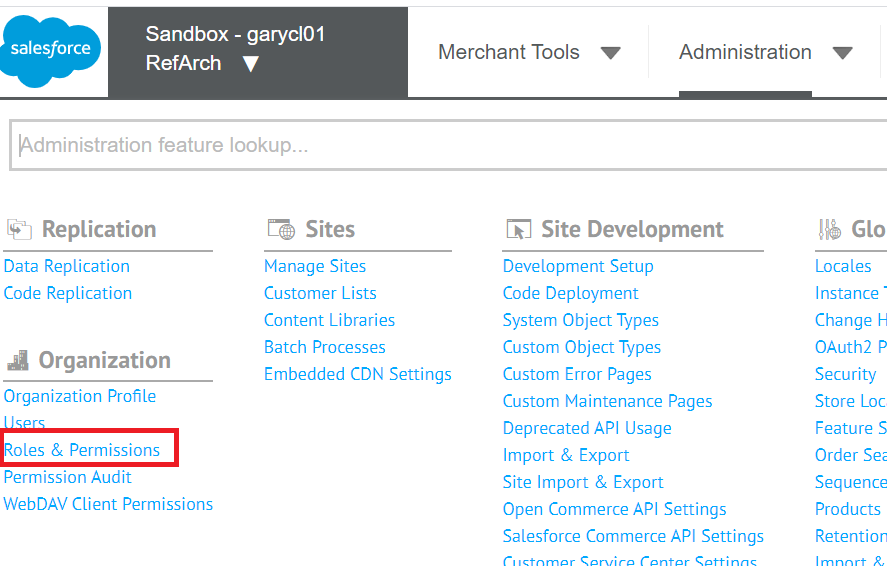
**5.2.1 Access Cloudinary Media Library through Business Manager**

To open the Cloudinary Media Library within your Business Manager:

* Open Business Manager and select your site.
* Click Merchant Tools -> Cloudinary Admin -> Media Library. The Media Library opens.



* If you don’t find this module then try granting BM module permission to the logged in user.
* To assign BM module permissions click Administration > Organization > Roles & Permissions.



* Click “Administration” and go into the “Business Manager Modules” tab.
* Select your context and select both checkboxes for the “Cloudinary Admin” and “Media Library” modules, then save.



* If still you aren't able to see the Media Library module then try logging in again.

Note: For the Media Library, the “bm\_cloudinary” cartridge must be in your Business Manager site effective path.

## 5.3 Storefront Functionality

*Intentionally left blank.*

# 6 Known Issues

*Intentionally left blank.*

# 7 Release History

|  |  |  |
| --- | --- | --- |
| **Version** | **Date** | **Changes** |
| 1.0.0 | 03/31/2020 | Initial release |
|  |  |  |