Digital Experience for Communications

Implementing Buying Experience

April 2024

Digital Experience for Communications
Implementing Buying Experience

April 2024

F77904-12

Author: Tejaswi Tatavarthi

Contents

Get Help

There are a number of ways to learn more about your product and interact with Oracle and other users.

Get Help in the Applications

Use help icons to access help in the application. If you don't see any help icons on your page, click your user image or name in the global header and select Show Help Icons.

Get Support

You can get support at *My Oracle Support*. For accessible support, visit *Oracle Accessibility Learning and Support*.

Get Training

Increase your knowledge of Oracle Cloud by taking courses at *Oracle University*.

Join Our Community

Use *Cloud Customer Connect* to get information from industry experts at Oracle and in the partner community. You can join forums to connect with other customers, post questions, suggest *ideas* for product enhancements, and watch events.

Learn About Accessibility

For information about Oracle's commitment to accessibility, visit the *Oracle Accessibility Program*. Videos included in this guide are provided as a media alternative for text-based topics also available in this guide.

Share Your Feedback

We welcome your feedback about Oracle Applications user assistance. If you need clarification, find an error, or just want to tell us what you found helpful, we'd like to hear from you.

You can email your feedback to *oracle_fusion_applications_help_ww_grp@oracle.com*.

Thanks for helping us improve our user assistance!

1 About This Guide

Audience and Scope

This guide is for product administrators, tenant administrators, and system integrators, who intend to use the Buying Experience application to enable the omnichannel buying and ordering experience for customers.

The guide also provides information to back-office specialists to set up Buying Experience, and describes the various features in Buying Experience.

Related Guides

Refer to the guides in this table for more information about the tasks covered in this guide.

Guide	Description
Oracle Digital Experience for Communications REST API for Buying Experience	Describes how you can use TM Forum Open APIs to support your business scenarios.
Oracle CX for Industries Implementing CX Industries Framework	Describes how to use CX Industries Framework to set up the integrations and communicate with other applications.
Oracle CX for Industries REST API for CX Industries Framework	Describes how you can use TM Forum Open APIs to support your business and Oracle REST APIs to configure or integrate applications.
Oracle Digital Experience for Communications Implementing Launch Experience	Describes how to set up your instance of Launch Experience in conjunction with the activities that you perform to set up Oracle CX Sales and Oracle Fusion Service.
Oracle Digital Experience for Communications Implementing Care Experience	Describes how to set up your instance of Care Experience in conjunction with the activities that you perform to set up Oracle CX Sales and Oracle Fusion Service.
Oracle Applications Cloud Implementing Applications	Describes setup tasks that apply to multiple or all applications in Oracle Applications Cloud.
Oracle Communications Billing and Revenue Management REST Services Manager API	Describes how you can use the Oracle Communications Billing and Revenue Management Rest Services Manager APIs to support TM Forum Open APIs and retrieve information from Buying Experience.

2 Implementation Overview

Get Started

Buying Experience is automatically provisioned and deployed in your environment. Here are some tasks you must perform before implementing the application:

- Enable the integration with Launch Experience. For instructions, see the Integration with Launch Experience topic in this guide.
- If you're using Care Experience, associate the Care Experience users with the Oracle Fusion applications role named, Support Specialist. For instructions, see the Integration with Care Experience topic in this guide.
- If required, set up your application as the gatekeeper for publishing events by using the configuration APIs in CX Industries Framework. For instructions, see the Set Up Gatekeepers topic in the Implementing CX Industries Framework guide in *CX Industries Framework (2720527.1)* on My Oracle Support (https://support.oracle.com).

Related Topics

- Integrate with Launch Experience
- Integrate with Care Experience

Tasks Summary

Here are the tasks that you perform to set up the application. Use this table as a checklist to understand the sequence of the tasks and where you perform them. The tasks are categorized based on the tools that you use to do these tasks.

Setup Activity	Tasks	Work Area	Read More
Configure user experience	• Set up your own user experience component to create user journeys as needed.	REST APIs	Chapter: Configure user Experience
Create users and assign roles	• Create user accounts. • Review, edit, lock, or delete existing user accounts. • Assign roles to user accounts. • Reset users passwords.	Oracle Identity Cloud Service	Chapter: Create Users and Assign Roles
Integrate external tax application	• Integrate the external tax application if you're using anything other than Avalara AvaTax.	REST APIs in CX Industries Framework	Chapter: Integrate with Other Applications

Setup Activity	Tasks	Work Area	Read More
Set up listeners for events	• Register the external applications to listen to outbound events as needed.	REST APIs in CX Industries Framework	Chapter: Integrate with Other Applications
Configure abandoned cart settings	• Configure the retention period after which a shopping cart is abandoned or purged.	Buying REST APIs	Chapter: Configure Shopping Carts
Enable or disable outbound events	• Enable or disable an outbound event for publishing based on your business needs.	Buying REST APIs	Chapter: Manage Outbound Events
userKeytype	• You should set this to either partyNumber or accountNumber depending on the tenant's preference on whether the user account in the Identity Management should be associated with the subscribers' party or partyAccount entity. This is done in CX Industries Framework. This configuration is a one-time setup and can't be changed subsequently.	CX Industries Framework	Read CX Industries Framework documentation for more information.

3 Create Users and Assign Roles

Assign Job Roles

You use role-based access control to protect data and to limit access to application resources. For example, your role might enable you to view your shopping cart, but not other's shopping cart.

A job role defines a user's business function. As a tenant administrator, you must ensure that users are assigned appropriate job roles to access the application.

> **Note:** You can also create groups with user roles and map them to the user roles in the Buying Oracle Cloud Service in the Oracle Identity and Access Management (IAM).

Here are the predefined job roles for accessing Buying Experience. These job roles are mapped as groups in Oracle Identity Cloud Service.

User	Job Role	User Group	Description
Anonymous user	Anonymous	NA	Can view, create, or update anonymous shopping carts. This role isn't associated with any user. Anonymous users can use client credentials to access the application.
Digital Storefront user	Subscriber	Subscriber	Can view, create, or update accounts, shopping carts, orders, assets, and agreements specific to the user.
Care Agent	Support Specialist	Support Specialist	Can access all the users details.
Tenant Administrator	Back Office Specialist	Back Office Specialist	Can access Business Configuration REST APIs to set up the application.
Bulk Job Administrator	Bulk Job Administrator	Bulk Job Administrator	Grants user access for creating bulk jobs.
Contract Administrator	Contract Administrator	Contract Administrator	Grants user access for contract management-related setup.
Business Operations Specialist	Business Operations Specialist	Business Operations Specialist	Grants user access to do operations such as market release date updates for preorders.

User	Job Role	User Group	Description
Inventory Management Specialist	Inventory Management Specialist	Inventory Management Specialist	Grants user access to do operations such as create, update, or fetch for channel, store, and SKU inventory.
System Integrator	System Integrator	System Integrator	Grants user access to do operations on product offerings search API.
System Administrator	System Administrator	System Administrator	Grants user access to do operations on get product query by parameters on service address fields.

Note: In 22C and onward versions of the Buying Experience application that use the CX Industry Framework IDCS instead of the Fusion Applications IDCS, Buying job roles aren't mapped as IDCS groups. They're instead mapped as application roles in the Buying Experience application. Ensure that you recreate users and subscribers in the CX Industries Framework IDCS, and assign roles accordingly.

Create Users and Assign Groups

Buying Experience uses the Oracle Applications Cloud users created in Oracle Identity Cloud Service.

Note: You must be a tenant administrator with the User Administrator role to create and manage users in Oracle Identity Cloud Service.

You can create the users for the Subscriber role directly using the POST method of the Individual API endpoint. These users are automatically created in Oracle Identity Cloud Service and are assigned the Subscriber job role. For more information on the Individual REST API, refer to REST API for Buying Experience on Oracle Digital Experience for Communications Buying Experience (Doc ID 2730574.1) at My Oracle Support (https://support.oracle.com).

For other roles, such as Support Specialist and Back Office Specialist, you must create the users in Oracle Identity Cloud Service and assign the appropriate job roles. Note that the job roles are mapped as groups in Oracle Identity Cloud Service.

Here's how you can create a user and assign a group:

1. In Oracle Identity Cloud Service, navigate to **Users** and click **Add**.
2. On the Add User page, enter these details and then click **Next**:
    - First Name
    - Last Name
    - User Name or Email

    Note: The user name is case-sensitive.

3. On the Assign Group page, search for the group, for example, Support Specialist or Back Office Specialist.

   Note: A group in Oracle Identity Cloud Service is the equivalent of a job role in Oracle Applications Cloud.

4. In the search results, select the group and then click **Finish**.

An email is sent to the user to activate account and reset the password.

Remove Users

You use Oracle Identity Cloud Service to remove users from your application. Only the tenant administrator with the User Administrator role can perform this task.

You can remove users in either of these ways:

- Deactivate users: The user and all related information remain in the application. The user can be reactivated in the future.
- Delete users: The user and all related information are removed from the application. The user can't be reactivated.

Refer to the chapter Manage Oracle Identity Cloud Service Users in *Administering Oracle Identity Cloud Service* for more information.

4 Manage Data Privacy and Security

Manage Data Security Using Roles

You can use the security mechanisms such as role-based user access controls and anonymity to help protect data. The data can come from internal users and your customers, such as user credentials or account information.

You use predefined roles and REST APIs to access your data. Buying REST APIs are of these categories:

- **Generic entities**. These entities include ProductOfferings and ProductSpecifications. Any user can access these entities.
- **Anonymous entities**. These entities include ShoppingCart for anonymous users. Only anonymous users can access these entities. However, once this entity is associated with a subscriber account, only that subscriber can access this entity as an authenticated user.
- **User-specific entities**. These entities are subscriber-specific and include Party, Customer, PartyAccount, ShoppingCart, ProductOrder, and Assets. Only authenticated users with appropriate job roles can access these entities. For example, Subscriber 1 can't access the entities of Subscriber 2.

Here's the list of REST API entities with the corresponding data privileges for predefined roles.

REST Entity	Job Role	Privelege	Description
• ProductOffering	• Anonymous • Subscriber • Support Specialist	• GET	Can retrieve or view product offers.
• Anonymous ShoppingCart	• Anonymous	• GET • POST • PATCH	Can view, create, or update anonymous shopping carts.
• Individual • Customer • PartyAccount • Subscriber ShoppingCart • ProductOrder • Product(Asset) • Agreement • Quote • Promotion • Document	• Anonymous (applicable only for the individual entity) • Subscriber • Support Specialist	• GET • POST • PATCH	Can view, create, or update different type of accounts, subscriber's shopping carts, orders, assets, and agreements.
• Bulk Job	Bulk Job Administrator	• GET • POST	Can view, create, or update bulk jobs for performing bulk operations on product orders.

REST Entity	Job Role	Privelege	Description
		• PATCH	
• Configuration	Back Office Specialist	• GET • PATCH	Can view or update the configurations.
• TmfSpecs • ReferenceFields • Mappings • Template	Contract Administrator	• GET • POST • PATCH	Can view, create or update placeholder mappings and templates.
• Channel • Store • Inventory	Inventory Management Specialist	• GET • POST • PATCH • PUT	Can view, create, or update channels, stores, and inventory.

Here are a few things to know while working with REST entities:

- A subscriber can't view other subscribers' offers.
- A support specialist can view or manage all the subscribers REST entities.
- A subscriber can access the following REST entities that are created by a support specialist:
  - Party
  - Customer
  - Party Account
  - Shopping Cart
  - Product Order
- In an account hierarchy, a parent can place an order for a child. A parent can also view or manage child's REST entities. But the child can't view or manage parent's REST entities.
- A subscriber can't view or update another subscriber's REST entities.

Related Topics

- Integrate with Order Management

Manage Access Using OAuth

Buying Experience uses the OAuth 2.0 protocol to authenticate external applications and authorize users to access the REST APIs. The OAuth token-based authentication helps to prevent unauthorized access to application resources. The OAuth role-based authorization helps to protect the data viewed and managed by users.

The external applications use the confidential application created in Oracle Identity Cloud Service, referred to as trusted client, to access the Buying REST APIs. You can use the CX Industries Framework Gateway and the CX Industries Framework's trusted client to access these APIs. The request to access the REST APIs must include an OAuth access

token. You can generate these tokens by using the client ID, client secret, and scope of the CX Industries Framework's trusted client in Oracle Identity Cloud Service.

You can pass the access token in the header of the HTTP request. These tokens are converted into an authentication header and the sign-in URL is called to pass the authentication header to the authentication service. On authentication, you can access the REST APIs.

5 Integrate Buying

Preintegrated Applications

Buying Experience comes preintegrated with other cloud services. This enables the application to route API requests and receive or send necessary event data by using CX Industries Framework. The preintegrated applications include:

- Launch Experience: Acts as a product master and publishes design-time catalogs to this application.
- Care Experience: Provides Customer 360 and order care support for your business scenarios.

You can also integrate your own-built applications or any third-party applications to coexist with your application and listen to events generated by this application.

Integrate with Launch Experience

Launch Experience comes preintegrated with your application. You must perform some setup tasks to enable this integration.

Here's how you can enable it:

Note: You must be a tenant administrator to perform this task.

1. Go to **Tools** > **Security Console**.
2. Go to **Users** > **Add User Account**.
3. Check if user FABRIC_SYSTEM_USER is already there. See Create Oracle Fusion Users for Inbound Flows topic in the *Implementing Care Experince* guide.
4. Enter the following user information, if you are creating the user:
    - First Name
    - Last Name
    - Email: Provide any email address where you can receive the email notifications.
    - User Name: FABRIC_SYSTEM_USER.
    - Password: Any user-defined password
    - Confirm Password: Same as above password
5. Click **Add Role**, search for the ORA_ATC_COMMUNICATIONS_CATALOG_VIEWER_JOB role, click **Add Role Membership** and then click **Done**.
6. Click **Save and Close**.

Once enabled, you can publish the catalogs created in Launch Experience to your application by using initiatives. The lifecycle status management for all the catalog entities are done through these initiatives defined in Launch Experience.

Integrate with Care Experience

Care Experience comes preintegrated with this application. To access Buying Experience, you must associate the Care Experience users with the Support Specialist role in the Identity Cloud Service console. For instructions, see the Assign Groups to the User Account topic in the *Administering Oracle Identity Cloud Service* guide.

Care Experience users can then access the information about a customer's purchases, orders, and assets in the application.

Integrate with Oracle Content Management

Here's how you can enable this integration:

1. Add Buying Experience as a confidential application in the Oracle Identity Cloud Service instance used by Oracle Content Management to enable this integration. For example, you can create the confidential application as BuyingX-OCE-Client. For instructions, refer to the following topics in the *Integrating and Extending Oracle Content Management* guide:
    - Access using 2-Legged OAuth.
    - Create an OAuth Client and acquire a Client ID and Secret.
    - Grant the required Oracle Content Management Roles to the Client.
2. Create a service request on My Oracle Support at https://support.oracle.com to enable Buying Experience to use the Oracle Content Management (OCM) instance. Ensure that you provide the following details in the service request that you raise:
    - OCE URL: The URL of the OCE instance.
    - OCE IDCS URL: The URL of the Oracle Identity Cloud Service instance of OCE.
    - OCE IDCS Scope: The scope of the OCE instance as configured in the Oracle Identity Cloud Service instance of OCE.
    - OCE IDCS Client ID: The client ID generated by the confidential application.
    - OCE IDCS Client Secret: The client secret generated by the confidential application.

You can then create a folder in your OCE instance for storing files and job templates for bulk operations.

Integrate with Order Management

For Buying Experience to receive updates on orders submitted to the Order Management (OM) system for fulfillment, ensure that your order management system publishes the following events to the Buying listeners:

- ProductOrderAttributeValueChangeEvent: Use this event to send updates specific to order items, such as the order status change or milestone updates. Ensure that this event is published to the following listener: /

orchestrationProductOrderingManagement/listener/productOrderAttributeValueChangeEvent. The payload for this event is similar to the TMF 622 ProductOrderAttributeValueChangeEvent API.

- ProductOrderStateChangeEvent: Use this event to send state changes at the order-header level, such as order completion or order cancellation. Ensure that this event is published to the following listener: / orchestrationProductOrderingManagement/listener/productOrderStateChangeEvent. The payload for this event is similar to the TMF 622 ProductOrderStateChangeEvent API.
- CancelProductOrderStateChangeEvent: Use this event to send updates specific to Order Cancel, such as is order eligible for cancelling. Ensure that this event is published to the following listener: / orchestrationProductOrderingManagement/listener/cancelProductOrderStateChangeEvent.
    - When you raise a request to cancel order and it is in in-progress state and eligible for cancellation then Buying will generate "CancelProductOrderCreateEvent" which will be consumed by OM system and when OM generates CancelProductOrderStateChangeEvent with in-progress, then buying will move Order state to "Assessing Cancellation" by consuming that event. After receiving "CancelProductOrderStateChangeEvent" from Orchestration with "Done" state , Buying will change the Order state to "Pending Cancellation" and if the event state is "TerminatedWithError" then it moves Order state back to "in-progress".
- CancelProductOrderStateChangeEvent: Use this event to send updates specific to Order Cancel, such as is order eligible for canceling. Ensure that this event is published to the following listener: / orchestrationProductOrderingManagement/listener/cancelProductOrderStateChangeEvent.
    - When you raise a request to cancel order and it is in in-progress state and eligible for cancellation then Buying will generate "CancelProductOrderCreateEvent" which will be consumed by OM system and when OM generates CancelProductOrderStateChangeEvent with in-progress, then Buying will move order state to "Assessing Cancellation" by consuming that event. After receiving "CancelProductOrderStateChangeEvent" from orchestration with "Done" state, Buying will change the order state to "Pending Cancellation" and if the event state is "TerminatedWithError" then it moves order state back to "in-progress".

Related Topics

- Manage Access Using OAuth

Integrate with External Applications

External applications integrated with this application access Buying REST APIs by using the routing rules predefined in CX Industries Framework. The routing rule ensures that the API requests are routed to the target application through the appropriate API.

Integrate with Event Listeners

You can use the CX Industries Framework UI to set up external applications as listeners for the outbound events generated by Buying Experience.

For instructions, refer to the Integrate External Applications chapter in the CX Industries Framework Implementation Guide in *CX Industries Framework (2720527.1)* on My Oracle Support (https://support.oracle.com).

For information on outbound events, see the Manage Outbound Events chapter in this guide.

Integrate with Document Adapter

Here's how you can integrate Buying experience with the Framework document management adapter:

1. Configure the CX Industries Framework document management adapter to integrate with the content management system.
2. Create a service request on My Oracle Support to add the content management system as a spoke system in the CX Industries Framework, and configure Buying experience appropriately.
3. Ensure that you provide the following details in the service request that you raise:
    - Document Management System ID: Any string to indicate the instance ID such as OCM1.
    - In addition to these details, to generate contracts, you must provide:
        - CONTENTMANAGEMENT TEMPLATES FOLDERID: The folder in the content management system where uploaded contract templates will be stored.
        - CONTENTMANAGEMENT CONTRACTS FOLDERID: The folder in the content management system where generated contract documents will be stored.
    - In addition to these details, to store individual attachments, you must also provide:
        - CONTENT MANAGEMENT INDIVIDUAL DOCS FOLDERID: Folder in the content management system to store individual identification document.

Document management adapter is designed to integrate with any document management system. You must configure the preferred document management system as a spoke system in CX Industries Framework. See samples in the following sections.

Configure Oracle Content Management as a Spoke System in CX Industries Framework

The CX Industries Framework document management adapter is designed to work with Oracle Content Management (OCE) as the default document management system. Perform the following steps to configure a Oracle Content Management instance as a spoke in CX Industries Framework.

Prerequisites

The following information is required to configure:

- OCE URL: The URL of the OCE instance.
- OCE IDCS URL: The URL of the Oracle Identity Cloud Service instance of OCE.
- OCE IDCS Scope: The scope of the OCE instance as configured in the Oracle Identity Cloud Service instance of OCE.
- OCE IDCS Client ID: The client ID generated by the confidential application.
- OCE IDCS Client Secret: The client secret generated by the confidential application.
- Document Management System ID : Any arbitrary string to represent the OCM instance such as OCM1.
- > **Note:** You must use the value of the document management system ID configured here in the buying configuration also.

Configure Oracle Content Management as a Spoke System in Framework Cluster

Here are the steps that you must perform through the Framework APIs with the Framework OAuth credentials.

1. OCM API Definition in Framework.

   Verify API definition.

   DO GET is available at `https://<fabric api gw >/admin/apis`

   Verify that the API ocm-v1 is available as shown in the following API definition (GET) sample response:

```
{
 "api-name":"ocm-v1",
 "api-id":"ocm-100",
 "api-version":"v1",
 "api-resources":[
 {
 "resource-id":"file-id",
 "name":"file-id",
 "resource-path":"{document-management-system-id}/api/1.2/files/{id}",
 "path-parameters":[
 {
 "parameter-name":"id",
 "parameter-type":"string",
 "is-record-id":false
 },
 {
 "parameter-name":"document-management-system-id",
 "parameter-type":"string",
 "is-record-id":false
 }
 ],
 "cors-preflight-handling":"fabric",
 "routing-ambiguity-resolution-strategy":"HTTP400BadRequest"
 },
 {
 "resource-id":"file-id-data",
 "name":"file-id-data",
 "resource-path":"{document-management-system-id}/api/1.2/files/{id}/data",
 "path-parameters":[
 {
 "parameter-name":"id",
 "parameter-type":"string",
 "is-record-id":false
 },
 {
 "parameter-name":"document-management-system-id",
 "parameter-type":"string",
 "is-record-id":false
 }
 ],
 "cors-preflight-handling":"fabric",
 "routing-ambiguity-resolution-strategy":"HTTP400BadRequest"
 },
 {
 "cors-preflight-handling":"fabric",
 "name":"folder-id",
 "path-parameters":[
 {
 "is-record-id":false,
 "parameter-name":"id",
```

```
 "parameter-type":"string"
 },
 {
 "is-record-id":false,
 "parameter-name":"document-management-system-id",
 "parameter-type":"string"
 }
 ],
 "resource-id":"folder-id",
 "resource-path":"{document-management-system-id}/api/1.2/folders/{id}",
 "routing-ambiguity-resolution-strategy":"HTTP400BadRequest"
 }
 ],
 "alternative-root-path":"documents",
 "api-events":[

 ],
 "openapi-document-url":"https:/do-nothing",
 "id":"ocm-v1"
}
```

2. Create a new target type definition (TTD) or system descriptor.

   Go ahead and define the target type definition (TTD), also known as the system descriptor.

   REST API: POST

   ```
   https://<fabric api gw >/admin/systemDescriptors
   ```

   Note the id returned in the response.

   OCM sysDescriptors (post request payload)

   ```
   {
    "target-name":"ocm-v1",
    "external":{
    "apis":[
    {
    "api-id":"ocm-100",
    "api-version":"v1",
    "api-resources":[
    {
    "resources":[
    {
    "resource-id":"file-id"
    },
    {
    "resource-id":"file-id-data"
    },
    {
    "resource-id":"folder-id"
    }
    ]
    }
    ]
    }
    ]
    },
    "system":"ocm",
    "domain":"ocm",
    "type":"external"
   }
   ```

3. Create a new TIC or connection descriptor.

   REST API: POST

   ```
   https://<fabric api gw >/admin/connectionDescriptors
   ```

   The end point, client id, and secret of the target OCM instance are required.

   Note the id returned in the response.

   connectionDescriptor (Post request payload)

   ```
   {
    "system-descriptor": "ocm-api-ttd", #id obtained in the previous step
    "endpoint-name": "ocm-v1",
    "endpoint-url": "<OCE URL>", #actual OCM instance URL
    "fabric-facing-auth": {
    "oidc-client-credentials": {
    "client-id": "<OCE IDCS Client ID>", #actual client id
    "client-secret": "<OCE IDCS Client Secret>", # secret
    "identity-uri": "<OCE IDCS URL>/oauth2/v1/token", # IDCS used to authenticate OCM
    "scope": "<OCE IDCS Scope>" #actual scope
    }
    },
    "type": "external"
   }
   ```

4. Create routing criteria based on path param (documentManagementSystemId)

   POST

   ```
   https://<fabric api gw >/admin/routingCriteria
   ```

   As the user, you can give any arbitrary value. This should be used as the docAdapterDocumentManagementSystemId in the buying configuration or caller configuration.

   Routing Criteria (Post request payload)

   ```
   {
    "any-criteria-of": [
    {
    "path-param-criteria": {
    "path-parameters": [
    {
    "path-parameter-name": "document-management-system-id",
    "path-parameter-value-oneOf": [
    "<Document Management System ID>"
    ],
    "path-parameter-backend-handling": "routing-only"
    }
    ]
    }
    }
    ],
    "criteria-link-key": "OCM Criteria"
    }
   ```

   Note the id returned in the response. This id is required in the GKR claim process.

5. Claim GKR

You have defined the APIs , defined TTD for the system, and a TIC also. You must now claim the resources. After TTD and TIC are submitted, you will see an autogenerated GKR. Get the ID of the newly generated GKR and claim the resources along with any criteria, that you may want to define.

DO GET is available at `https://<fabric api gw >/admin/gatekeepingRules?endpoint-name=ocm-v1`

From the response get the id (gkrID) of the corresponding record and use the id to update gkr with the following steps (sample response):

```
[
 {
 "endpoint-name": "ocm-v1",
 "rule-name": "Generated gatekeeping rule for endpoint ocm-v1",
 "destination-selection": [
 {
 "api-id": "ocm-100",
 "api-version": "v1",
 "criteria": [
 {
 "rank": 10,
 "resource-ids": [
 "file-id",
 "file-id-data",
 "folder-id"
 ]
 }
 ]
 }
 ],
 "id": "gkr-ocm-v1nxccm"
 }
]
```

DO GET is available at `https://<fabric api gw >/admin/gatekeepingRules/{gkrID}`

Copy the response.

API PUT is available at `https://<fabric api gw >/admin/gatekeepingRules/{gkrID}`

GKR Claim (Put request payload): Update the GKR with the criterion-link field having the ID of the criterion obtained from the previous step.

```
{
 "endpoint-name":"ocm-v1",
 "rule-name":"Generated gatekeeping rule for endpoint ocm-v1",
 "destination-selection":[
 {
 "api-id":"ocm-100",
 "api-version":"v1",
 "criteria":[
 {
 "rank":10,
 "resource-ids":[
 "file-id",
 "file-id-data",
 "folder-id"
 ],
 "criterion-link":"UNI1702899361" //ID From previous step
 }
 ]
```

```
  }
 ]
}
```

Note: You can download the latest version of the Framework Implementation guide from the My Oracle Support (Article ID 2720527.1) and see the Set Up Custom APIs chapter.

6 Manage Entities

Use REST APIs to Manage Your Entities

Before you start using the REST APIs, ensure that you have the required roles and privileges.

For more information, see the Create Users and Assign Roles chapter in this guide. Let's see how you can use the REST APIs to manage your buying and ordering entities.

You can extend various entities in line with your business requirements apart from the corresponding TMF specifications and Oracle extensions:

- Product Offering (Product Offering Information)
- Party (Individual)
- Customer
- Party Account
- Shopping Cart
- Product Order

For more information, see REST API for Buying Experience in *Oracle Digital Experience for Communications Buying Experience (2730574.1)* on My Oracle Support at https://support.oracle.com.

Manage Sales Catalog

You create and use the sales catalog in the runtime. It contains a collection of the following:

- Product offers
- Catalog categories
- Product lines
- Product specifications
- Price lists
- Price or Alterations
- Recommendation rules
- Eligibility rules or Compatibility rules
- Migration rules

Each product offer in a catalog describes the relationships between products, the services used to realize the products, and the resources they require. You use the category to organize the product offers in the sales catalog. You can use the product catalog management REST APIs to get and manage various catalog entities, such as:

- Product offering: You can get all the product offers, bundles, and packages from the design-time applications. You can now add multiple commercial bundles to a package. The API response includes banners, marketing features, and product offer recommendations. You can separate catalogs by the catalog type using the product catalog management API. For example, product, service and resource. You can also define a rank for each

of the attachments for product offers in the design time. Use the ranks to display the default image for the product offering in the customer shopping journey.

In addition, you can use the queryProductRecommendation REST API to query product offer recommendations based on the product offering ID. These recommendations are very essential for e-commerce businesses in upsell or cross-sell scenarios.

- Product offering price: You can get different product pricing schemes to associate with all types of product offers such as package, bundles, and simple offers. In addition to component based pricing model, you can also associate prices to packages and Bundles (commercial and service bundle). The price associated at the package or bundle level will be inclusive of the prices of the required components of the package or bundle. Optional components of the bundle or package have a separate price. The various supported pricing schemes are listed here:
    - One-time, recurring, usage, or penalty fees
    - Attribute-based pricing, where prices are based on combination of characteristics values derived from product specifications
    - Product-based alterations and component alterations (in packages)
    - Volume-based discounts for non-asset product offerings
    - Filter product offering price at aggregate and compatible-offering levels based on the price list.
    - Define tax inclusive price by providing the tax percentage with the tax code associated to the product offering price.
- Price lists: You can get the prices for product offerings according to the price lists configured in design time. Price lists can be region-specific (for example, different currencies). To get prices for a product offering that belongs to a specific price list, provide the price list ID or name as the query parameter in the GET productOffering endpoint. Similarly, to get the correct price for a line item, specify the price list information in the payload you create for the shopping cart and product order requests. Otherwise, the application creates the items with only 0 (zero) price. You can perform price adjustments at a component level, such as a bundle.

  Product lines: You can get a product line to group-related product offers together.

  Product specifications: You can get the product attributes associated with a product offer to define the technical aspects of that offer.

  Inventory availability: You can pass the SKU numbers with comma separated values along with locations to check the inventory availabilities. The API response has SKU number, product offer Id, channel types along with regular and preorder stock available as true or false. True means available.

Search in the Sales Catalog

Prospects and subscribers can find plans, devices, and accessories using the Oracle Buying Search Service. Search service options include:

- Catalog browsing:

  The customers can browse and filter different product offers based on their requirement using the GET productOffering by list APIs. The customer can apply multiple filter criteria to minimise the search result for

a better browsing experience. You can also get the details of a specific product offer using the GET by ID API. Here are the lists of filters and other functionalities that are supported by the productOffering APIs:

a. Filters product offers based on the type (packages, services, devices, accessories, service bundles, and commercial bundles).
b. Filters product offers based on provided categories.
c. Filters product offers based on multiple eligibility criteria.
d. Filters products offers that are on a specific price range.
e. Filters product offers based on price list name.
f. Filters products that are shippable, sellable, and for which installation is required.
g. Filters product offers based on user defined characteristics that are enabled in the design time.
h. Filters product offers that have banners or promotions associated with.
i. Filters product offers based on certain search tags and banner names.
j. Sorting of product offers based on price and name in ascending or descending order.
k. In case of packages and bundles, the customer can see the detailed price break-up for the each bundled products, including list price, adjusted price, tax, and any other alterations.
l. Supports filters on product offers on after marketing extension fields.
m. Supports browsing and also shows price break up of compatible items (device, service, and accessories) in the context of the parent (package and bundles).
n. Supports query on related offers: Related Offers refers to offers that are linked or associated in some way. If a customer is viewing a specific offer, related offer could be the additional services that the provider thinks the customer might be interested in.

- Auto suggest:

  Search provides (Type-ahead) API, that can help the customers to provide out-of-the-box support of providing suggestions of product offers as they start typing any keyword in the search bar.

- Keyword search:

  The search also supports search APIs for keyword search, sorting, and browsing product offers using POST APIs:

  - Keyword (free-text) search:

    For the search bar use-case where if a customer tries to find the product offers with a specific keyword, and if the keyword is present in any searchable fields (name, description, category name, banner name, searchTag, banner description, marketingFeature name, marketingFeature description, marketingFeature shortDescription, and commercial name), then those products are returned in the API response.
  - Sorting:

    Search API supports sorting by name#and sorting by prices of product offers. Sorting order can be ascending or descending based on input parameters. If no sorting criteria is passed then the product offers are sorted in the default order that is, Package > Device > Accessory > Service > Commercial Bundle > Service Bundle.

- Browsing and filtering of product offers POST:

  The Search API also supports the feature of browsing and filtering product offers using the POST API. You can modify the request payload using the filter criteria to filter product offers based on type, categories, eligibility criteria, pricelist name, price range, characteristics, and extension fields. You can also filter product offers that have banners or promotions, or that can be shipped, sold, or require installation, using this API.

Manage Quotes

You can create a quote for package purchases using the Quoting API. You can create quotes using multiple price lists that are defined in different currencies. You can update quotes, such as changing the account details, packages, optional items, devices and services, header level notes and email addresses. You can save your quote using the Enterprise buyer email address and retrieve it when needed.

Manage Shopping Cart

Your customers use shopping carts to temporarily save the selected products and retrieve them later for a purchase. The customer can be a subscriber or an anonymous user. You can use the shopping carts REST API to create, update, retrieve, and check out shopping carts. You can also do the following using this API:

- Group cart items to represent items that belong to a specific person, such as a member in a family plan, and assign it a nickname.
- Let your customers purchase devices that aren't part of any package and associate the device with the package that they purchase. They can also associate any existing device with their new package. When they terminate the package, all the associated devices will be terminated.
- Let your customers procure devices on a lease basis for first-time purchases. The device that you procure can be part of a package or an individual product, but you must associate it with the package by defining a relationship.
- Get detailed item price break-up for your products, including alterations:
    - List Amount: Starting price of the offer.
    - Alteration Amount: The discount amount that's applied on the offer. Currency value in case of discount is %.
    - Total Alteration Amount: The total discount amount on the offer.
- Support advanced discounting features such as:
    - Time Based Discounts: Discounts can be provided, that are valid only for a certain period after the purchase or service activation.
    - Limited Time Discounts: Discounts can be provided, that are short lived and are valid during a specific occasion.
    - Criteria Based Discounts: Discounts can be provided based on user's (Subscriber and Anonymous user or guest) location parameters. Also, this could be extended in field to support discounts based on extension parameters.
- Send shipping method, tracking ID, and URL in the API response to help in order fulfillment.
- Enable or disable eligibility and compatibility-related validations.
- Send the channel-specific URL from external clients in the shopping carts.
- Support manual price override for simple offers, packages, and bundles by a support specialist role during cart creation or update. The manual override allowed can't be above the floor limit.
- Support ability to either apply manual price overrides on top of the system discounts (discount amount or discount percentage on the item price) or completely override the system discount with the manual price overrides.
- Support out of the box extensions in the shopping cart.

- In addition to the existing eligibility criteria, Customer Type is a newly supported criterion in the shopping cart. Subscribers with at least one active product are treated as existing subscribers where as others are treated as new subscribers.
- Support family plan and account hierarchy in the shopping cart.
    - The shopping cart API with accountHierarchyChange flag will create an order when the account has atleast one active asset present under it and internally the account hierarchy will be changed. The new target parent will be able to query the asset after the hierarchy change and the old parent (source parent account) won't be able to query the asset.
- Support change of owner account, billing account, or service account in the case of family plan within the account hierarchy.
- Support package termination when aggregate discounts were applied in the shopping cart.
- Support for location based discounts in the shopping cart for new purchase, update, and termination.

  > **Note:** To apply the location based discount, you must pass location parameters under eligibility profile in the request payload.

- Create preorder for future dated product offerings of type, device or accessory with subtype as preorder.
- Support ability to enable a tenant or a system integrator to centrally define the business logic using Business Logic Extensions (BLE). The business logic can then be triggered from the shopping cart. The business logic can vary from simple to complex validations, algorithm, or calculations.
- Support computation of tax of a cart item internally. If the tax percentage is associated to the price of a product offer then the tax percentage will be used to compute the tax internally.

  > **Note:** An example use case for this could be integrating with an external pricing system to manually override the price of a cart item.

- You can now abandon and purge shopping carts that can be triggered using specific user-roles, using a new Bulk job API that can be triggered as an on-demand job (this functionality is specifically available for Bulk job administrator and Business Operations Specialist roles). This complements the current functionality that abandons and purges the shopping carts as a scheduled background job run at mid-night every day.

For stock increment or decrement, in the API request, you have to pass SKU number, channel code, quantity, action (INCREMENT/DECREMENT) and stock type (REGULAR/PREORDER). In API response, you will get id, href, SKU number, channel code, regular, or preorder stock final value after the action taken, stock availability date, and market release date. In addition, you can use the configuration REST API to configure the abandoned shopping cart settings. For more information, see Configure Shopping Carts chapter in this guide.

Manage Product Configurator

You may sometimes configure products in the runtime. For example, add simple products to one or more complex bundles or update subscriber's existing assets at the runtime. You typically do this to add a promotion bundle or a standalone configurable product to a quote or order. You can use the product order management REST APIs to do the following:

- Get all the existing product offers and bundles.
- Configure promotion bundles.
- Add optional child products.
- Update configurable attributes for bundles and child products, for example, update the speed for high-speed internet service.

Manage Product Orders

You can use the product order management REST APIs for product orders.

The product ordering API resource lets you create, view, get, update, and cancel product orders and also get an order cancellation request. You can also do the following using these APIs:

- Let your customers purchase devices that aren't part of any package and associate the device with the package that they purchase. They can also associate any existing device with their new package. When they terminate the package, all the associated devices will be terminated.
- Let your customers procure devices on a lease basis for first-time purchases. The device that you procure can be part of a package or an individual product, but it has to be associated with the package by defining a relationship.
- Search for specific product orders.
- Enable or disable eligibility and compatibility-related validations.
- Get detailed item price break-up for your products, including alterations:
  - List Amount: Starting price of the offer.
  - Alteration Amount: The discount amount that's applied on the offer. Currency value in case of discount is %.
  - Total Alteration Amount: The total discount amount on the offer.
- Support advanced discounting features, such as aggregate discounts, for first-time purchases.
- Group order items to represent items that belong to a specific person, such as a member in a family plan, and assign it a nickname.
- Send shipping method, tracking ID, and URL in the API response to help in order fulfillment.
- Capture the estimated fulfillment start date received from the fulfillment application in your product order.
- Defer submission of orders to the fulfillment application when created using the POST method of the product ordering API. These orders are created in the pending state. Your subscribers or support specialists can later update these orders created based on their requirement and then submit for fulfillment using the PATCH method.
- Track the base product order life cycle state in accordance with the Cancel product order life cycle states as the cancellation request is being processed by the downstream fulfillment system.
- Support manual price override for simple offers, packages, and bundles by a support specialist role during order creation or update. The manual override allowed can't be above the floor limit.
- Support ability to either apply manual price overrides on top of the system discounts (discount amount or discount percentage on the item price) or completely override the system discount with the manual price overrides.
- Support out of the box extensions during order creation or update.
- Query product order using anonymous role.
- In addition to the existing eligibility criteria, Customer Type is a newly supported criterion in the product order. Subscribers with at least one active product are treated as existing subscribers where as others are treated as new subscribers.
- Support family plan and account hierarchy in product order.
- Support change of owner account, billing account, or service account in the case of family plan within the account hierarchy.
- Support package termination when aggregate discounts were applied in product order.

- Support location based discounts in product order for new purchase, update, and termination.

  > **Note:** To apply the location based discount, you must pass location parameters under eligibility profile in the request payload.

- Create preorder for future dated product offerings of type, device or accessory with subtype as preorder.
- Bulk submission of preorders to fulfillment (by bulk job administrator).
- Bulk update to market release date for preorders (by business operations specialist).

You can perform bulk operations on product orders by creating and running jobs using the bulkJob API. You can use this REST API to do the following:

- You can create the product orders in bulk. Also, submit the previously created product orders to fulfillment in bulk.
- Get bulk job details by ID
- Get all the bulk jobs matching specific criteria, for example, status, completion date, or both.

A new background job added runs every day in the 24 hours interval. This job queries all the preorders that are in pending status and for which the market release date (MRD) is today (current date). This job submits these orders to the fulfilment system for further processing. For more information, see the Manage Bulk Operations chapter in this guide.

Manage Products (Assets)

You can use the product management REST APIs to get the products or assets.

A product consists of one or more services or resources. The Products API resource lets you view and change group name and alias for products. You can calculate disconnect penalty for a product and get package change options using this resource. You can also do the following:

- Search for specific products.
- Get products that are purchased on lease basis.
- Get products that are purchased outside of a package but are associated with the package. Terminating the package terminates all the devices associated with that package.
- Get detailed item price break-up for your products, including alterations:
  - List Amount: Starting price of the offer.
  - Alteration Amount: The discount amount that's applied on the offer. Currency value in case of discount is %.
  - Total Alteration Amount: The total discount amount on the offer.
- Modify assets, such as adding a new optional product or changing the asset characteristics.
- Get the list of assets upto root parent (package) level, installed in a given service address or service ID (This functionality is specifically available for System Integrator and Support Specialist roles.)

Manage Agreements

You can use the agreements REST API to get the terms and conditions committed to enter into a contract.

Manage Accounts

You can use the account management, customer management, and party REST APIs to get all the accounts to whom the service is delivered or billed. For example, party accounts, customers, and individuals. These resources help you to create, delete, get, and update party accounts, individual accounts, and customers.

You can get payment methods associated with an account. Also, search for customers and party accounts using the engaged party ID or related party ID respectively. You can create account hierarchy between accounts. You can update account hierarchy of accounts using Account Management API when there are no open carts, orders, or assets. Single level account hierarchy is supported.

You can create accounts with valid account types that are defined in business configuration.

You can attach one or more documents of the following types: pdf, jpg, jpeg, png, docx, doc, gif, and bmp for party entity (Individual) to support use cases such as identification documents.

> **Note:** You can store the documents in any document management system through the document management adapter. You must configure the CXIF document management adapter to work with the preferred document management system.

Manage Promotions

You can use the promotion REST API to get all the available promotions and apply them to the customers who meet the specified criteria. You can get all the promotion details or a specific promotion by using the GET by ID method. Here are the filter criteria you can use to apply promotions:

- Product Offering
- Product Line
- Product Specification
- Location-Specific attributes
- Start date and end date based criteria
- Partial or exact promotion name
- Price list name or price list ID
- Entity ID or entity name
- State, City, Postal Code, and Country

Manage Contracts

Generate Contract on Order Completion

The ProductOrder based contracts will get generated only when the Business Process Config 'generateContractOnOrderCompletion' is set to true.

After a Product order is created (via Shoppingcart checkout or ProductOrder API), you can use the generateContract REST APIs on these entities to create a Contract creation job. The system will generate a contract document as a PDF when the product order is fulfilled by the order management system. You can view the generated contract document under the related documents of agreement. The contract document will have a unique contract number added.

Manage Generic Contracts

Here's how you can use the Generic Contracts API when the Business Process Config 'generateContractOnOrderCompletion' is set to false.

1. You can use the Contracts REST APIs for managing the contracts.
2. The Contract Management API resource lets you create the generic contract document, download the contract document, get, and update the contract record information.
    a. Generate Contract Document: By using POST API, you can generate the generic contract document for any external entity such as 'ShoppingCart' or 'ProductOrder' by passing the externalEntityId, entityType, and templateName in the request payload. The allowed roles are Anonymous, Subscriber, and Support Specialist.
    b. Update Contract Life Cycle Status: By using PATCH API, the contract administrator can update contract document life cycle status with 'accepted', 'active', 'inactive', and 'obsolete' statues based on the business needs.
    c. Download Contract Document: You can download the contract document. The allowed roles are Subscriber, Support Specialist, and Contract Administrator. The subscribers are allowed to download their contract documents only. Support specialists and contract administrators are allowed to download any contract documents.
3. The Generic Contract API expects input in the form of JSON data. The JSON data is divided into two sections: a fixed flat attributes section and a dynamic section that varies based on the template selected for generating the contract.
    - The fixed flat attribute section includes fields such as templateName, externalEntityId, entityType, externalEntityHref, accountId, accountNumber, externalContractNumber, validFrom, and validUpto.
    - The dynamic section starts with the field input and is further divided into header, body, and footer. The fields under the header, body, and footer vary based on the selected template to generate the contract.
    - Values for placeholders existing in the header section of the template should be provided under the "header" key in the JSON payload. Similarly, for placeholders existing in the body section of the template, values should be placed under the "body" key, and for placeholders in the footer section, under the "footer" key.
    - For table entries, values should be organized in an array because a table can have more than one row or column.

Manage Subscriber User Creation

As part of the subscriber purchase, you can create user accounts in the identity management system using the following two APIs:

1. https://<server>/api/buyingUserManagement/v1/onboardUser
2. https://<server>/api/buyingUserManagement/v1/user

> **Note:** In the case of family plan, you can also create users for the child accounts in the hierarchy using this API: https://<server>/api/buyingUserManagement/v1/user

For more information, see REST API for Buying Experience in *Oracle Digital Experience for Communications Buying Experience (2730574.1)* on My Oracle Support at https://support.oracle.com.

Manage SKU Inventory

You can create and use the SKU inventory in the runtime. It contains a collection of the following:

- SKUs with Product Offers
- Channels
- Stores
- Inventory for SKU and Channel combination
- Global Threshold Configuration
- Channel and SKU Threshold Configuration

SKUs with Product Offer: SKUs with product offer will be published from Launch. You can get all the product offers along with their SKUs, only SKUs with their attributes or you can fetch only specific SKU or product offer by Id or SKU number or with Siebel Id. You can use PATCH and PUT APIs to modify existing SKU or add new SKU to existing product offer.

Channels: Channels will be created by Inventory Management Specialist. You can get all the channels or fetch only specific channel by Id. You can use PUT and PATCH API to update the channel information. You can't modify the channel code.

Stores: Store will be created by Inventory Management Specialist associating to one channel code. You can't assign same channel code to multiple stores. You can get all the stores or fetch a store by Id.

Inventory: Inventory will be created by Inventory Management Specialist. You can get all the inventories or specific inventory by passing SKU number and channel code. The API response will have SKU number, channel code, regular stock or preorder stock or both, stock availability date, and market release date.

Global Threshold Configuration: Global threshold configuration will be set by Back Office Specialist.

Channel and SKU Threshold Configuration: The threshold configuration will be set to specific channel and SKU combination. This configuration will be set by Back Office Specialist.

Events are generated:

- If the stock quantity is less than or equal to threshold value, an event will be generated.
- If the stock quantity is zero, an event will be generated.
- If the stock quantity is less than or equal to safe limit value, an event will be generated.

Onboarding New User in the Shopping Cart Checkout Process

Onboarding User

- When you’re checking out, you’ll capture details of the customer information.

  These include customer details such as last name and first name. These also include customer type such as retailer, end customer, customer from an organization, or partner. The payment method is also captured. Entities such as individual, customer, and partyAccount are created.
- When you checkout, along with checkout, order processing also occurs in the shopping cart API. An order is created for the particular checkout.

  The payment details are processed and then the application realizes that the user is anonymous and not yet registered. Buying Experience application will send an email notification to you based on the email ID that

you shared as a part of checkout process. In case of family plan, Buying Experience application will send email notification to child accounts as well based on the email address.

- You'll receive the email.

  In the email, there's a URL.
- Onboarding supports hierarchy of accounts also. You must provide entities such as individual, customer, and partyAccount in ArrayList format.

Registering User

- Click the URL to register yourself as the user.

  After clicking the URL, you'll reach the user registration page.
- Provide personal details such as personal information and credit card information.
- Click **Register**, the onboard user API is run.
- Provide all the required sign in credentials and these will be stored in the Identity Cloud Service.

Note: If you click the registration link beyond the expiration time then you'll receive another email notification with the latest link to register.

- You're created as the user in the Identity Cloud Service.

  The Identity Cloud Service will send an email to you to reset the password.
- When you click the reset password link in the email, you'll reach the Identity Cloud Service page and you can reset the password.
- After you reset the password, you can use your user ID and password to sign into the Buying Experience application.
- **Note:** You can complete the registration process using APIs for standard attributes such as code or user name or account number or order number, or email address attribute.

Custom Claim

Custom claim is any attribute other than the standard attributes of user ID or email address. Custom claim is an attribute such as party number for example, using which the Buying Experience application can identify and register the user.

7 Manage Events

Manage Outbound Events

Buying Experience generates outbound REST events or notifications for each action against Buying resources.

You can use these events to notify the listeners about the events occurred in Buying Experience. You can enable or disable an event and manage all the outbound events by using the configuration REST API.

Here's a list of outbound events generated by Buying Experience.

Outbound Events	Description
IndividualCreateEvent	Generated when an Individual account is created.
IndividualAttributeValueChangeEvent	Generated when individual account attributes are updated.
CustomerCreateEvent	Generated when a customer account is created.
CustomerAttributeValueChangeEvent	Generated when customer account attributes are updated.
PartyAccountCreateEvent	Generated when a party account is created.
PartyAccountAttributeValueChangeEvent	Generated when party account attributes are updated.
PartyAccountStateChangeEvent	Generated when the status of a party account is changed.
ShoppingCartCreateEvent	Generated when a shopping cart is created.
ShoppingCartAttributeValueChangeEvent	Generated when shopping cart attributes are updated.
ShoppingCartAbandonmentEvent	Generated when no actions are taken on shopping carts for specific period. This event is used to send abandoned cart notifications.
ShoppingCartDeleteEvent	Generated when a shopping cart isn't updated for a specific duration after abandoned cart notification. This event is used to delete shopping carts.
ProductOrderCreateEvent	Generated when a product order is created.
CancelProductOrderCreateEvent	Generated when a product order is canceled.
ProductOrderStateChangeEvent	Generated when the status of a product order is changed.

Outbound Events	Description
CancelProductOrderStateChangeEvent	Generated when the order fulfillment application cancels the base product order and sends the ProductOrderStateChangeEvent for the base product order.
IndividualStateChangeEvent	Generated when the status of the Individual (Party) entity is changed.
CustomerStateChangeEvent	Generated when the status of the Customer entity is changed.
ProductOrderSubmitEvent	Generated when the product order is submitted for fulfillment.
ProductCreateEvent	Generated when an asset is created after the fulfillment of an order.
ProductAttributeValueChangeEvent	Generated when one or more attributes of a product (asset) are updated.
ProductStateChangeEvent	Generated when the state of the asset is updated.
QuoteCreateEvent	Generated when a quote is created.
QuoteAttributeValueChangeEvent	Generated when one or more attributes of a quote are updated.
UserRegistrationEmailEvent	Generated when the shopping cart is checked out by an anonymous user.
UserRegistrationCreateEvent	Generated when a self-service user is created when the anonymous user completes the user registration process.
MigrationJobCreateEvent	Generated when a new migration job is created in the system.
MigrationJobStateChangeEvent	Generated when there is any change in the status of the migration job.
ThresholdBreachRegularEvent	Generated during regular stock increment or decrement if stock quantity is less than or equal to threshold configuration.
ThresholdBreachPreorderEvent	Generated during preorder stock increment or decrement if stock quantity is less than or equal to threshold configuration.
SafeLimitBreachRegularEvent	Generated during regular stock increment or decrement if stock quantity is less than or equal to safeLimit configuration.
SafeLimitBreachPreorderEvent	Generated during preorder stock increment or decrement if stock quantity is less than or equal to safeLimit configuration.
NoStockRegularEvent	Generated during regular stock increment or decrement if both stock quantity and safeLimit are zero.
NoStockPreorderEvent	Generated during preorder stock increment or decrement if both stock quantity and safeLimit are zero.

See How to Manage Buying Configurations topic in this guide for more information on managing, enabling, and disabling outbound events.

8 Configure Buying

Manage Buying Configurations

Buying Experience provides a standardized mechanism for all the runtime configurations required by Buying services.

Overview

Buying configuration APIs help you to enable or disable outbound events, update abandoned cart settings, configure template placeholder mappings, and the template document metadata.

Manage Configurations

You can manage the buying runtime configurations by retrieving details using the configuration API. You can get all the available configuration details using this API.

Here's how you can get the configuration details:

- To get the configuration details, run the following cURL command or use a REST API client:

```
curl -H Authorization: Bearer <accessToken> https://<hostName>/cx/industry/buying/configManagement/v1/configurations -X GET
```

- To get the configuration details for a specific configuration, run the following cURL command or use a REST API client:

```
curl -H Authorization: Bearer <accessToken> https://<hostName>/cx/industry/buying/configManagement/v1/configurations/{configurationId} -X GET
```

Insert Inventory Configuration

Here's how you insert or update the global threshold configuration for inventory:

1. Create a payload for specifying the global threshold for all the SKU and channel combinations with details described in this table:

   Table

Field	Description
id	Configuration Id
name	The valid value is "Inventory Configuration".
configuration	List of configurations.

Field	Description
type	The valid values are: o Regular o Preorder
threshold	Global threshold value
safeLimit	Global safe limit value
individualThreshold	List of individual thresholds.
skuNumber	SKU number
channelCode	Channel Code
threshold	threshold value
safeLimit	safe limit value

2. Call the configuration API by running this cURL command or by using a REST API client:

```
curl -H Authorization: Bearer <accessToken> https://<hostName>/cx/industry/buying/configManagement/v1/
configuration/inventoryConfig -X POST -H "Content-Type: application/json" -d @sc-data.json
```

where

- <accessToken> is the OAuth access token for Back Office Specialist role.
- <hostName> is the URL for the CX Industries Framework API Gateway.

Response Body:

```
{
 "id": "6",
 "name": "Inventory Operations",
 "configuration": [
 {
 "type": "regular",
 "threshold": 8,
 "safeLimit": 7,
 "individualThreshold": [
 {
 "skuNumber": "HH01-R6260-71-A",
 "channelCode": "10000",
 "threshold": 10,
 "safeLimit": 10
 }
 ]
 },
 {
 "type": "preorder",
 "threshold": 8,
 "safeLimit": 7,
 "individualThreshold": [
 {
 "skuNumber": "HH01-R6260-71-A",
 "channelCode": "10000",
 "threshold": 10,
```

```
 "safeLimit": 10
 }
 ]
 }
 ]
}
```

How to Enable or Disable Outbound Events

You can enable or disable publishing for a specific outbound event by using the configuration API. With this option, you can publish only the outbound events that are needed for your business scenarios.

Here's how you enable or disable publishing for an outbound event:

1. Create a payload for specifying whether an event is publishable with the details described in this table.

Field	Description
id	Unique id
name	"Events"
configuration	List of configuration objects.
name	The name of the event to be enabled or disabled.
enabled	Specify whether this event is publishable. The valid values are: - FALSE: The specified event isn't published to the listeners. - TRUE: The specified event is published to the listeners.
groupName	Group to which this event belongs to.

2. Call the configuration API by running this cURL command or by using a REST API client:

```
curl -H Authorization: Bearer <accessToken> https://<hostName>/cx/industry/buying/configManagement/v1/
outboundConfig-X PUT -H "Content-Type: application/json" -d @sc-data.json
```

where:

- <accessToken> is the OAuth access token for **Back Office Specalist** role.
- <hostName> is the URL for the CX Industries Framework API Gateway.

The API returns a response if the duration is updated.

Sample request payload:

```
{
 "id": "1",
 "href": "https://cptadceqy-buy-cluster4.dxbuying.ocs.oraclecloud.com/orch9f35cbf4/cx/industry/buying/
configManagement/v1/internal/configuration/outboundConfig/1",
 "name": "Events",
 "@type": "OracleOutboundConfiguration",
 "configuration": [
 {
```

```
 "name": "ThresholdBreachRegularEvent",
 "enabled": true,
 "groupName": "individual",
 "@type": "OracleOutboundConfiguration"
 }
 ]
}
```

How to Update Abandoned Cart, Shopping Cart and User Expiry Duration

Here's how you update the abandoned cart or purge cart or user expiry duration:

1. Create a payload for specifying the duration with the details described in this table.

Field	Description
id	Unique id
name	"Buying Configurations"
configuration	List of configuration objects.
name	The name of the event to be enabled or disabled.
duration	duration
uom	Unit of measure

2. Call the configuration API by running this cURL command or by using a REST API client:

```
curl -H Authorization: Bearer <accessToken> https://<hostName>/cx/industry/buying/configManagement/v1/
buyingConfig-X PUT -H "Content-Type: application/json" -d @sc-data.json
```

   Where:

   - <accessToken> is the OAuth access token for **Back Office Specalist** role.
   - <hostName> is the URL for the CX Industries Framework API Gateway.

The API returns a response if the duration is updated.

Here's a sample request payload:

```
{
 "name": "Buying Operations",
 "configuration": [
 {
 "name": "shoppingCartAbondenment",
 "duration": 12,
 "uom": "month"
 },
 {
 "name": "shoppingCartPurge",
 "duration": 20,
 "uom": "days"
 },
 {
 "name": "shoppingCartExpiry",
 "duration": 48,
```

```
"uom": "weeks"
},
{
"name": "userRegistrationExpiry",
"duration": 48,
"uom": "hours"
}
]
}
```

Work with Generic Configuration

Tenants can define configurations that they want to store in a central place and can be used by different channels or roles. These configurations can be as simple as that of a key-value pair to a complex structure. The omni channel ordering engine allows tenants to define their custom configurations that are flexible. The configurations must be provided in json format.

The definition of the generic business configuration is done by back office specialists but it must be readable by any other role. To maintain security, we must also enable the back office specialist to define the roles for a particular configuration, that any user with the role can read. To get the configuration details, run the following cURL command or use a REST API client:

```
curl -H Authorization: Bearer <accessToken> https://<hostName>/cx/industry/buying/configManagement/v1/
genericConfig?name=<Config name> -X GET
```

Insert Generic Configuration

Here's how you insert or update the generic configuration:

1. Create a payload for specifying the generic configuration with these details:

   Generic configuration table

Field	Description
id	Id. It can be either user passed unique id or auto generated by Config MS service.
name	Unique generic configuration name
description	It is optional field. Description of configuration.
permittedRoles	Roles permitted to view these configurations. Multiple roles must be sent as comma separated.
configuration	Configuration in json format

2. Call the configuration API by running this cURL command or by using a REST API client:

   ```
   curl -H Authorization: Bearer <accessToken> https://<hostName>/cx/industry/buying/configManagement/v1/
   configuration/genericConfig -X POST -H "Content-Type: application/json" -d @sc-data.json
   ```

   where
   - <accessToken> is the OAuth access token for Back Office Specialist role.
   - <hostName> is the URL for the CX Industries Framework API Gateway.

For update call, use PATCH HTTP method.

Response Body:


```
[
 {
 "id": "Id_1705516512",
 "href": "https://cptadceqy-buy-cluster4.dxbuying.ocs.oraclecloud.com/orch9f35cbf4/cx/industry/buying/
configManagement/v1/configuration/genericConfig/Id_1705516512",
 "name": "Config_1705516512",
 "description": "Store front Configuration",
 "permittedRoles": "Back Office Specialist",
 "startDate": "2024-01-18T18:35:10.387Z",
 "endDate": "2024-02-16T18:35:10.387Z",
 "configuration": {
 "isBundle": false,
 "syncTime": "2023-08-02T08:41:44.076088Z",
 "popRelationship": [
 {
 "name": "",
 "id": "0CX-1Z42O5",
 "href": "https://api-dev-search.cxcomms.dxbuying.ocs.oraclecloud.com/orch168a6d8b/cx/industry/search/
tmf-api/productCatalogManagement/v4/productOfferingPrice/0CX-1Z42O5"
 },
 {
 "name": "",
 "id": "0CX-1Z42O5",
 "href": "https://api-dev-search.cxcomms.dxbuying.ocs.oraclecloud.com/orch168a6d8b/cx/industry/search/
tmf-api/productCatalogManagement/v4/productOfferingPrice/0CX-1Z42O5"
 }
 ],
 "lifecycleStatus": "active",
 "validFor": {
 "startDateTime": "2021-01-01T00:00:00.000Z",
 "endDateTime": "2999-12-31T00:00:00.000Z"
 },
 "@type": "ProductOfferingPriceOracle",
 "priceType": "alteration",
 "description": "",
 "priceList": [
 {
 "name": "DX4C NA Pricelist",
 "id": "0CX-1YU1OD",
 "href": "https://api-dev-search.cxcomms.dxbuying.ocs.oraclecloud.com/orch168a6d8b/cx/industry/search/
tmf-api/productCatalogManagement/v4/priceList/0CX-1YU1OD"
 }
 ],
 "@baseType": "ProductOfferingPrice",
 "percentage": 100.0,
 "name": " Fire TV Discount",
 "isTaxInclusive": false,
 "href": "https://api-dev-search.cxcomms.dxbuying.ocs.oraclecloud.com/orch168a6d8b/cx/industry/search/
tmf-api/productCatalogManagement/v4/productOfferingPrice/0CX-1YUGFP",
 "id": "0CX-1YUGFP",
 "relativeValidFor": {
 "endRelativeDateTime": {
 "relativeEvent": "PURCHASE",
 "offSetUnit": "MONTHS",
 "offSetValue": 5
 },
 "startRelativeDateTime": {
 "relativeEvent": "PURCHASE",
 "offSetUnit": "MONTHS",
 "offSetValue": 1
```

```
 }
 }
 },
 "createdOn": "2024-01-17T18:35:12.450Z",
 "updatedOn": "2024-01-17T18:35:12.450Z"
 }
]
```

Delete Generic Configuration

Here's how you delete the generic configuration:

1. Call the configuration API by running this cURL command or by using a REST API client:

   ```
   curl -H Authorization: Bearer <accessToken> https://<hostName>/cx/industry/buying/configManagement/v1/configuration/genericConfig/{id} -X DELETE -H "Content-Type: application/json"
   ```

   where

   - <accessToken> is the OAuth access token for Back Office Specialist role.
   - <hostName> is the URL for the CX Industries Framework API Gateway.

   Response Body:

```
[
 {
 "id": "Id_1705516512",
 "href": "https://cptadceqy-buy-cluster4.dxbuying.ocs.oraclecloud.com/orch9f35cbf4/cx/industry/buying/configManagement/v1/configuration/genericConfig/Id_1705516512",
 "name": "Config_1705516512",
 "description": "Store front Configuration",
 "permittedRoles": "Back Office Specialist",
 "startDate": "2024-01-18T18:35:10.387Z",
 "endDate": "2024-02-16T18:35:10.387Z",
 "configuration": {
 "isBundle": false,
 "syncTime": "2023-08-02T08:41:44.076088Z",
 "popRelationship": [
 {
 "name": "",
 "id": "0CX-1Z42O5",
 "href": "https://api-dev-search.cxcomms.dxbuying.ocs.oraclecloud.com/orch168a6d8b/cx/industry/search/tmf-api/productCatalogManagement/v4/productOfferingPrice/0CX-1Z42O5"
 },
 {
 "name": "",
 "id": "0CX-1Z42O5",
 "href": "https://api-dev-search.cxcomms.dxbuying.ocs.oraclecloud.com/orch168a6d8b/cx/industry/search/tmf-api/productCatalogManagement/v4/productOfferingPrice/0CX-1Z42O5"
 }
 ],
 "lifecycleStatus": "active",
 "validFor": {
 "startDateTime": "2021-01-01T00:00:00.000Z",
 "endDateTime": "2999-12-31T00:00:00.000Z"
 },
 "@type": "ProductOfferingPriceOracle",
 "priceType": "alteration",
 "description": "",
 "priceList": [
 {
 "name": "DX4C NA Pricelist",
 "id": "0CX-1YU1OD",
```

```
 "href": "https://api-dev-search.cxcomms.dxbuying.ocs.oraclecloud.com/orch168a6d8b/cx/industry/search/
tmf-api/productCatalogManagement/v4/priceList/0CX-1YU1OD"
 }
 ],
 "@baseType": "ProductOfferingPrice",
 "percentage": 100.0,
 "name": " Fire TV Discount",
 "isTaxInclusive": false,
 "href": "https://api-dev-search.cxcomms.dxbuying.ocs.oraclecloud.com/orch168a6d8b/cx/industry/search/
tmf-api/productCatalogManagement/v4/productOfferingPrice/0CX-1YUGFP",
 "id": "0CX-1YUGFP",
 "relativeValidFor": {
 "endRelativeDateTime": {
 "relativeEvent": "PURCHASE",
 "offSetUnit": "MONTHS",
 "offSetValue": 5
 },
 "startRelativeDateTime": {
 "relativeEvent": "PURCHASE",
 "offSetUnit": "MONTHS",
 "offSetValue": 1
 }
 },
 "createdOn": "2024-01-17T18:35:12.450Z",
 "updatedOn": "2024-01-17T18:35:12.450Z"
 }
]
```


You can also pass configuration name as query parameter to delete the configuration.

```
curl -H Authorization: Bearer <accessToken> https://<hostName>/cx/industry/buying/configManagement/v1/
configuration/genericConfig/{id} -X DELETE -H "Content-Type: application/json"
```

where

- <accessToken> is the OAuth access token for Back Office Specialist role.
- <hostName> is the URL for the CX Industries Framework API Gateway.

Work with Smart Menu Configuration

For UX to be accessible, menu items require in-built help text.

These menu items lead to a subset of product offerings that are filtered by the product offering filter criteria. Only the tenant back office specialist can create or update the menu configurations for the given channel.

- To get the menu configuration details for a channel, run the following cURL command or use a REST API client:

  ```
  curl -H Authorization: Bearer <accessToken> https://<hostName>/cx/industry/buying/configManagement/v1/
  menuConfig/{channel_id} -X GET
  ```

- To get specific menu configuration for given channel, run the following cURL command or use a REST API client:

  ```
  curl -H Authorization: Bearer <accessToken> https://<hostName>/cx/industry/buying/configManagement/v1/
  menuConfig/{channel_id}/{menu_id}? -X GET
  ```

Insert Menu Configuration

Here's how you can insert or update the menu configuration:

1. Create a payload for specifying the menu configuration with details described in the table:

Menu Configuration Details

Field	Description
channelId	Channel id
name	Channel name
description	Description
menu	Array of menu objects.
id	menu id
name	menu name
description	Menu description
order	order sequence number
startDate	Start date
endDate	End date
displayInfo	Object has attributes with display text.
name	name
description	description
locale	List of objects.
key	Locale key
value	object
name	Name
description	Description
facet	Facet object
id	facet id
queryFilter	Query filter object
key	filter key
value	value

2. Call the configuration API by running this cURL command or by using a REST API client:
3. `curl -H Authorization: Bearer <accessToken> https://<hostName>/cx/industry/buying/configManagement/v1/menuConfig -X POST -H "Content-Type: application/json" -d @sc-data.json`

   where
   - <accessToken> is the OAuth access token for Back Office Specialist role.
   - <hostName> is the URL for the CX Industries Framework API Gateway.

Manage Buy Business Process Configuration

Buy Business Process Configuration is used for enabling or disabling the generate contract document based on order completion.

Manage Configurations

You can manage the buying runtime configurations by retrieving details using the configuration API. You can get all the available configuration details using this API.

Here's how you can get the configuration details:

- To get the configuration details, run the following cURL command or use a REST API client.

  ```
  curl -H Authorization: Bearer <accessToken> https://<hostName>/cx/industry/buying/configManagement/v1/configuration/businessProcessConfig -X GET
  ```

- To get the configuration details for a specific configuration, run the following cURL command or use a REST API client.

  ```
  curl -H Authorization: Bearer <accessToken> https://<hostName>/cx/industry/buying/configManagement/v1/configuration/businessProcessConfig?name=generateContractOnOrderCompletion -X GET
  ```

How to Enable or Disable Generate Contract on Order Completion

By default, the generateContractOnOrderCompletion flag is set to false. When this flag is set to false, you can use Generic Contract APIs to generate contracts document and update the life cycle status of the contract. When this flag is set to true, then the contract document is generated based on order completion event.

You can enable or disable generating contracts on order completion using Business Process Config 'generateContractOnOrderCompletion' by following these steps:

1. Create the JSON request payload with the details described. For example, create 'update_config.json' file with the following JSON and use it in cURL command.

   > **Note:** You can use "enabled": false in the following request payload to disable generateContractOnOrderCompletion.

   Fields and descriptions table

Field	Description
name	The valid value is "Buy Business Process".
configuration	List of configurations.
name	configuration name. The valid value is "generateContractOnOrderCompletion".
enabled	Enable configuration name by passing either true or false.

```
{
 "name": "Buy Business Process",
 "configuration": [
 {
 "name": "generateContractOnOrderCompletion",
 "enabled": true
 }
 ]
}
```

2. You can use the created JSON (update_config.json) to update the generateContractOnOrderCompletion to enabled or disabled using the following API endpoint.
3. Call the configuration API by running this cURL command or by using any REST API client:

```
curl -X PATCH 'https://<hostName>/api/configManagement/v1/configuration/businessProcessConfig' -H
 'Content-Type: application/json' -H 'Authorization: Bearer <accessToken>' -d @update_config.json
```

   where
   - <accessToken> is the OAuth access token for **Back Office Specialist** role.
   - <hostName> is the URL for the CX Industries Framework API Gateway.

4. The API returns the following response:

```
{
 "id": "3f22f9ae-6dba-4975-b769-ef5ba6fcec92-1698212341",
 "name": "Buy Business Process",
 "href": "https://<hostName>/api/configManagement/v1/configuration/buyProcessConfig/3f22f9ae-6dba-4975-
b769-ef5ba6fcec92-1698212341",
 "configuration": [
 {
 "name": "generateContractOnOrderCompletion",
 "enabled": true
 }
 ]
}
```

Manage Template Placeholder Mappings

You can manage the mappings by retrieving the details using the Mappings API. You can get all the available mapping details using this API.

Here's how you can get the mapping details:

- To get the mapping details, run the following cURL command or use a REST API client:

```
curl -H Authorization: Bearer <accessToken> https://<hostName>/cx/industry/buying/configManagement/v1/
mappings -X GET
```

How to Update Template Placeholder Mappings

You can configure placeholders for the template file using the available TmfSpecs and ReferenceFields list.

Here's how you can add new placeholder mappings:

1. Create payload to create new mapping with the details described in the table

Field	Description
name	The unique placeholder name
action	The action to be performed on the placeholder. Accepted values are: o add o modify o delete
referenceField o id	The id of the reference field to be used for placeholder creation. The ID should be taken from the available list of reference fields.

2. Call the Mapping API by running this cURL command or by using a REST API client:

```
curl -H Authorization: Bearer <accessToken> https://<hostName>/cx/industry/buying/configManagement/v1/
mappings -X PATCH -H "Content-Type: application/json" -d @sc-data.json
```

To get the supported TmfSpec list, call the TmfSpec API by running this cURL command or by using a REST API client:

```
curl -H Authorization: Bearer <accessToken>
https://<hostName>/cx/industry/buying/configManagement/v1/tmfSpecs -X GET
```

To get the supported ReferenceField list, Call the TmfSpec API by running this cURL command or by using a REST API client:

```
curl -H Authorization: Bearer <accessToken>
https://<hostName>/cx/industry/buying/configManagement/v1/referenceFields -X GET
```

Where:

- <accessToken> is the OAuth access token for **Contract Administrator** Role.
- <hostName> is the URL for the CX Industries Framework API Gateway.

Manage Contract Templates

You can manage the Templates file by retrieving the details using the Template API. You can get all the available Template file details using this API.

> **Note:** Contract templates can be stored in any document management system through the document management adapter. The CXIF document management adapter must be configured to work with the preferred document management system.

Here's how you can get the template details:

- To get the template details , run the following cURL command or use a REST API client:
  ```
  curl -H Authorization: Bearer <accessToken> https://<hostName>/cx/industry/buying/configManagement/v1/template -X GET
  ```
- To download one specific template file, run the following cURL command or use a REST API client:
  ```
  curl -H Authorization: Bearer <accessToken> https://<hostName>/cx/industry/buying/configManagement/v1/template/{id} -X GET
  ```

How to Upload Contract Template

A contract template is used to generate the contract for the user. You need to upload the template using the Template API. While uploading the template file, you should add metadata along with the file.

Here's how you can upload the contract template file:

1. Create payload to upload the template with the details described. The payload is multipart:
   - First body part (template file in binary format)
   - - Param Name: **file**
     - Content-Type: application/vnd.openxmlformats-officedocument.wordprocessingml.document
   - Second body part (JSON Payload with MetaData)
   - - Param Name: **templateParameters**
     - Content-Type: application/json
   - As part of templateParameters, you must add
     - **name**: Indicates name of the template. It is a required string attribute.
     - **specificationBased**: Indicates whether the template will be specification based or not. It is a required boolean attribute. Templates that support specification based placeholder mappings will have the value for this field as true. Where as, for static templates and for the generic contract templates, the specificationBased property will have the value as false.
     - **contentType**: Whether the content of file is **Static** -Doesn't contain any placeholder and should be in html format. It will be used for before checkout scenarios or **Dynamic** - May contain Placeholder and would be used for the actual contract generation and should be in docx format.

- **domain**: Currently it can have only "Consumer", for the B2C Scenario.
- **Tags**: Tags for the uploaded template file for which this template file should be used for Contract Generation. Supported tags are accountType, country, stateOrProvince.

2. Call the Template API by running this cURL command or by using a REST API client:

```
curl --location --request POST 'https://<hostname>/cx/industry/buying/configManagement/v1/template'
--header 'Authorization: Bearer <Access-Token>' --form 'file=@"contractTemplate.docx"' --form
'templateParameters=@"template-request.json";type=application/json'
```

Where:

- <accessToken> is the OAuth access token for **Contract Administrator** Role.
- <hostName> is the URL for the CX Industries Framework API Gateway.

Update Contract Template

You can update the already uploaded template file and the associated metadata with the file.

To update the contract, call the Template API by running this cURL command or by using a REST API client:

```
curl --location --request PATCH 'https://<hostname>/cx/industry/buying/
configManagement/v1/template' --header 'Authorization: Bearer <Access-Token>'
 --form 'file=@"contractTemplate.docx"' --form 'templateParameters=@"template-
request.json";type=application/json'
```

Where:

- <accessToken> is the OAuth access token for **Contract Administrator** Role.
- <hostName> is the URL for the CX Industries Framework API Gateway.

Key Points to Consider for a Contract Template

1. Templates of type DOCX are only accepted.
2. Templates should have a placeholder defined as **${placeholder_name}** for replacement with actual data. For example **${Account_Name}**
3. The table that mentions the terms and conditions of the different product offers that are part of the contract is called the Service Agreement Details table, which should have a header row and a data row with the required placeholder for each column's data that are dynamically expanded. The Templates can only have placeholder names from the below list, for the **Service Agreement Details** table that can also be fetched using the following URL:

   ```
   curl -H Authorization: Bearer <accessToken>
    https://<hostName>/cx/industry/buying/configManagement/v1/mappings -X GET
   ```

   and thereafter looking at "tmfSpec.id" field for value " **TMF651**".

Placeholder Name	TMF Spec ID	reference field path in Payload
ProductOffering_Name	651	agreementItem.productOffering.name
Agreement_StartDate	651	agreementPeriod.startDateTime
Agreement_EndDate	651	agreementPeriod.endDateTime
OwnerAccount_Name	651	ownerAccount.name

Placeholder Name	TMF Spec ID	reference field path in Payload
Term_Start_Date_Time	651	agreementItem.termOrCondition.validFor.startDateTime
Term_End_Date_Time	651	agreementItem.termOrCondition.validFor.endDateTime
Agreement_Name	651	name

4. You should specify the table caption **"SERVICE AGREEMENT DETAILS"** as table alternative text using MS Word's table properties.
5. The attributes mentioned in the Service agreement details table are mandatory for any contract template and are not configurable.
6. If any part of the template has content in a table format without a border, when this template is used to generate the contract in PDF, the system applies an outline border in the final PDF document.
7. We are not supporting textbox for a DOCX template.
8. DOCX template with the image as bullet point doesn't keep the image. Icons in bullet points are converted to simple bullet points.

For more information on using the contract templates, refer to the sample contract template provided in the support article for *Buying Experience (2730574.1)* on My Oracle Support at https://support.oracle.com.

Manage Generic Contract Templates

Generic contract templates are independent of placeholder mappings, allowing the use of any placeholder in the template. While using these templates to generate a contract, you must provide values for the defined placeholders within the template.

You can include a placeholder of the name 'ContractNumber' in the template header. The system will automatically generate the contract number and replace this placeholder, eliminating the need to provide this value in the JSON payload request. Also, if you prefer, you can place this placeholder in the footer or body of the template. 'ContractNumber' is a reserved placeholder, indicating that even if the caller provides a value for this placeholder, it can always be overridden with the generated value.

9 Configure Shopping Carts

Manage Shopping Carts

Buying Experience provides a standardized mechanism for shopping cart management.

It generates events when shopping carts are created or updated and sends the event data to other applications registered as listeners. If a customer leaves a cart unattended for certain number of hours, days, or weeks, an event is generated and published to the listener. The listener can use this event to send notifications to the customer.

Buying Experience uses the default abandon and purge duration to identify abandoned shopping carts and delete them after the specified duration. You can update this duration as required.

How to Update Abandoned Cart Settings

You specify the number of hours, days, or weeks after which a shopping cart is considered abandoned and an abandoned cart event is generated. By default, the duration for abandoned cart is set to two weeks and the duration to delete abandoned cart is set to four weeks. See How to Manage Buying Configurations topic in this guide for more information on updating abandoned cart settings.

Verify Status of Purge Jobs

The purge job automatically deletes all the abandoned shopping carts that aren't updated for the specified duration after the abandonment notification.

To check the status of last five purge jobs, call the purgeJob API by running this cURL command or by using a REST API client:

```
curl -H Authorization: Bearer <accessToken> https://<hostName>/bulk-api/v1/shoppingCart/purgeJob/ -X GET
```

where:

- <accessToken> is the OAuth access token for your account.
- <hostName> is the URL for the CX Industries Framework API Gateway.

The API returns a response with the details of the last five purge jobs.

Here's a sample response payload:

```
 [
{
"jobId": "8eeb2971-1b03-11eb-941f-657f728923d4",
"jobName": "Purge",
"createdOn": "2020-10-31T04:29:26.156Z",
"startedOn": "2020-10-31T04:29:26.273Z",
"completedOn": "2020-10-31T04:29:26.434Z",
"status": "CompletedError",
"request": "[Updated]='10/02/2020'",
"successfulRecords": {
```

```
"count": 3,
"records": [
"35-TLXSUT",
"38-GYU6QU",
"71-OWYGGC"
]
},
"failedRecords": {
"count": 2,
"records": [
{
"shoppingCartId": "96-XALB0G",
"error": "Database Error"
},
{
"shoppingCartId": "29-3AAJ0Y",
"error": "Database Error"
}
]
}
}
]
```

10 Manage Bulk Operations

Create and Run Jobs

You do bulk operation tasks, such as submitting product orders, by creating and running jobs using the bulkJob REST API.

With this API, you can create and submit product orders in bulk across party accounts. To provide the input for creating and running the jobs, you can use the job templates.

Note: You can do bulk operations only on product orders.

Here are some points to consider before you get started with bulk operation tasks:

- You must be a back-office specialist to perform this task. See the Create Users and Assign Roles chapter in this guide for more information.
- You must integrate Buying Experience with Oracle Content Management (OCM). For instructions, see the Integrate with Oracle Content Management topic in this guide.
- Create a folder in Oracle Content Management (OCM) for storing the job templates. You can then add the confidential application that you created in the Oracle Identity Cloud Service as a contributor or owner for the download access.
- Download the template named Product Order (Excel file) from My Oracle Support. This template is available in *Oracle Digital Experience for Communications Buying Experience (2730574.1)* on My Oracle Support at https://support.oracle.com.

Here's how you create and run the job:

1. Create a copy of the job template and enter the information in the Accounts and Product Order worksheets. You need all the fields in both the sheets to run the bulk operation tasks on product orders.

   Let's see what you can enter in the Accounts sheet:

   - Owner Account Number: Specify the owner account number.
   - Service Account Number: Specify the service account number.
   - Billing Account Number: Specify the billing account number.

   Let's see what you can enter in the Product Order sheet:

   - Sequence: Enter a unique integer value to identify each row in the Product Order Sheet (Subsequent rows have to be populated with integers starting from 1).
   - Action: Enter the type of operation you want to perform on the product offering. Here's the valid value: Add.
   - Product Offering Name: Enter the name of the product offering that you're planning to purchase.
   - Product Offering Type: Enter the type of the product offering for each row entry. Here are the valid Values:
     - Accessory

        - Charge
        - Commercial Bundle
        - Device
        - Package
        - Service Bundle
        - Service
    - Is Add On: Specify whether the product offering is an add-on or not. Here are the valid values: Y and N.

> **Note:** All these values are case-sensitive.

2. Upload the file to the folder you created in your OCE cloud instance. Ensure that you make note of the unique document ID of the uploaded file. You need it in the later steps.
3. Call the bulkJob API using the input payload provided in the *Buying Experience REST API* guide. You can run this cURL command or use a REST API client to call the bulkJob API:

```
curl -H 'Authorization: Bearer <accessToken>' https://<hostName>/cx/industry/bulk-api/v1/bulkJob/ -X
 POST -H "Content-Type: application/json" -d @sc-data.json
```

   Your payload must include the entity name and document ID of the file you uploaded in step 2.

   Here's the sample payload for bulk operations:

```
{
 "entityName": "ProductOrder",
 "document": {
 "documentId": "DB14B64C188D90924E926451295F5EDCCD607B5115C7",
 "version": 1
 }
}
```

> **Note:** The version here is optional. If not provided, the application downloads the latest version. You need to provide this value only when you want to download a previous version.

Here's the sample Product Order sheet for the two scenarios supported for bulk operations on product orders:

- Purchase of a single package: In the Account sheet, you include all the default components for a given package in the product orders for each account entry.

Sequence	Action	Product Offering Name	Product Offering Type	Is Add On
1	Add	Package ABC	Package	N

- Purchase of a single package with optional root-level product offerings as add-ons (with default quantity as 0): In the Account sheet, you include all the default components of this package, along with optional product offerings as add-ons, in the product orders for each account entry.

Sequence	Action	Product Offering Name	Product Offering Type	Is Add On
1	Add	Package ABC	Package	N

Sequence	Action	Product Offering Name	Product Offering Type	Is Add On
2	Add	Streaming Service 1	Service	Y
3	Add	Voice Roaming	Service	Y

Job Process Flow

The application uses the unique document ID to download the file. It then validates the file type, structure, and file content, and creates the job for running bulk operations job. Once you create the job, the application processes the job asynchronously and sends a response with the basic details. It includes information such as job ID, entity name, created on, status, and file name.

You can also get the status of the jobs you created by calling the get by ID method of the bulkJob API. Once the application processes the job and creates all the orders for the provided accounts, it updates the job with product order IDs. You can then call the bulkJob API using the GET by ID method to view the following job details along with the product order information:

- `jobId`
- `entityName`
- `createdOn`
- `startedOn`
- `completedOn`
- `status`
- `filename`
- `result`

You can find these details for each account number that you provided in your input. The result is always a link. You can click this link or call the API to get the actual results.

Related Topics

- Integrate with Oracle Content Management
- Assign Job Roles

Rerun Failed Jobs

The job process stops if any individual job fails. When a bulk operation job fails partially, the application returns the job status as CompletedError. In case if the job fails completely, the application returns the job status as Error. At this point, you can fix the payload data and submit the request for reprocessing the incomplete job.

You can update the bulk operation job by calling the PATCH method of the bulkJob API with the new document ID. You can run the following cURL command or use a REST API client:

```
curl -H 'Authorization: Bearer <accessToken>' https://<hostName>/cx/industry/bulk-api/v1/bulkjob/{jobID} -X
 PATCH -H "Content-Type: application/json" -d @sc-data.json
```

Here's the sample payload with the revised document ID:

```
{
 "document": {
 "documentId": "DB14B64C188D90924E926451295F5EDCCD607B5115C7",
 "version": 1
 }
}
```

The application updates the failed bulk operation job with the new data and processes all the jobs. Though the old data is overwritten by the new data, the old data remains in the OCE folder.

Submit Product Orders for Fulfillment

Use the PATCH method of the bukJob API to submit the created product orders for fulfillment.

You can do this only if the requested job ID is of type bulk job and it supports the Product Order entity. The job status must be Completed or CompletedError for you to submit the orders. The Completed status indicates that all the jobs have been completed and the CompletedError indicates that jobs have failed partially. In the case of "CompletedError", only the product orders that were successfully created are submitted for fulfillment.

To submit product orders for fulfillment, run the following cURL command or use a REST API client:

```
curl -H 'Authorization: Bearer <accessToken>' https://<hostname>/cx/industry/bulk-api/v1/bulkjob/{jobID} -X
 PATCH -H "Content-Type: application/json" -d @sc-data.json
```

Here's the sample payload for submitting product orders:

```
{
 "operation": "Submit"
}
```

Search for Jobs

You can search for jobs using the Get By ID or Get method of the bulkJob API on the bulk job resource. Use the Get by ID method to search for a job by ID and the Get method to search for jobs using a specific criteria.

You can search for jobs based on the following fields: status, completedOn, or by both. Here's a sample search criteria that you can enter in the `searchspec` parameter for retrieving the matching jobs:

```
REST URL?
searchspec=<search_criterion_or_search_criteria>&startnum=<numeric_value>&pagesize=<a_numeric_value_between_1_and_10
search_criterion = <any_allowed_field_name>[<search_operator>]<value>
search_criteria =
 <any_allowed_field_name>[<search_operator>]<value>&<any_allowed_field_name>[<search_operator>]<value>
```

Here are the supported field names and search operators:

Field Name	Valid Search Operator
status	=
completedOn	<, >, <=, >=

By default, pagination is enabled for the bulkJob API. When the pagination parameters aren't provided as query parameters, the API returns only 5 bulk jobs at a time starting from the first matching row.

Note: Date value in the specification should always be in the yyyy-MM-dd format. Otherwise, an exception occurs. Even if the date is in the correct format, an exception is returned if the date is invalid.

11 Use Payment Tokens

Use Tokenized Payments

Payments include collection of one-time payments for a purchase and recurring payments based on a specific schedule. You might perform refunds on one-time or recurring payments to resolve any disputes or when a service is canceled. Buying Experience facilitates the payment process by capturing and passing the token payment method.

With tokenized payments, the payment data is replaced by a token and passed during the payment transactions. This helps to secure the payment data. These tokens are issued in real time and they expire at first use or after a specified period.

Here's the list of scenarios where the token payment method is used in the buying and ordering journey:

- In unassisted purchase, shoppers enter the token payment method at the time of purchase.
- In unassisted care, subscribers update the token payment method in payment care.
- In assisted purchase, care agents enter the token payment method at the time of purchase.
- In assisted care, on request from a customer, care agents update the token payment method in payment care.

Use Payment Process

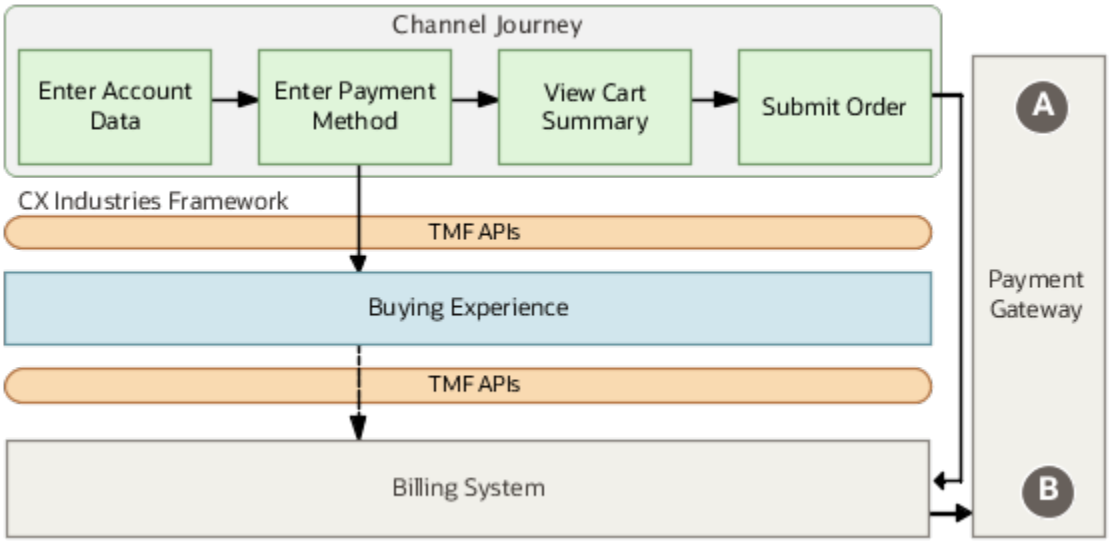
Payment data flows from engagement channels into Buying Experience and then from this application into your billing system through CX Industries Framework. The billing system can be Oracle Communications Billing and Revenue Management or any third-party billing system.

Ensure that your billing system is registered for listening to buying outbound events. Use the same payment gateway instance for all the engagement channels and the billing system so that the same instance can be used to collect all the payments.

Here's the payment process flow:

1. Each engagement channel captures the token from the payment gateway for first-time purchase by using Hosted Payment page and sends it to Buying Experience. Channels have to embed their own Hosted Payment page. Channels can collect one-time payments for purchase.
2. The application accepts the token and creates the token payment method. It also stores the payment method if the account is a Customer or Party account.
3. The application generates TMF events to notify the billing system about the payment method creation.
4. The application passes the token and the external payment method ID as correlation ID to the billing system.
5. The billing system creates tokenized payment method using the external payment method ID and token received from the application. The billing system stores the external payment method ID as external payment method number.
6. The billing system sends the external payment method number and the payment amount to the payment gateway for collecting payments.
7. The billing system sends the tokenized payment method for billing accounts to collect one-time payment, recurring payments, and also trigger refunds.

Here's an image that illustrates the payment process flow for a purchase.



12 Integrate with Tax Applications

Use Taxation Process

Buying Experience doesn't calculate or manage taxes. It comes with a prebuilt integration with Avalara AvaTax for Communications to calculate sales and use taxes for products and services purchased by your customers.

You can use the Avalara tax integration with TM Forum Open APIs to retrieve the tax amount based on tax codes and customer service address. The application connects to Avalara by default. If you use a different tax application or a custom tax engine, you can integrate that with the application by using CX Industries Framework.

Buying Experience uses your tax application to calculate taxes for every purchase. The offers purchased can include products with different tax implications; for example, a bundle with a set-top box and a sign-up fee. Additionally, they can be provisioned across multiple tax territories. When your customer initiates the checkout process and submits an order, Buying Experience passes the information required to calculate taxes to the tax application. This includes the purchase cost, customer service address, advice of charge, tax code, and tax exemption code (if applicable). It passes the information to the tax application by using the taxation adapter in CX Industries Framework. The tax application first validates the address and if it's valid, it calculates the total tax amount and sends it back to Buying Experience. The response can include individual tax components, such as the sales tax calculated for both the state and county.

Here are the scenarios where this application calls the tax application:

- To show tax inclusive or exclusive price of all offers in the quote to the buyers.
- To show total tax on purchases to customers before they submit the purchase request.

For bill payments, the billing system calculates the tax amount for each bill based on the taxation criteria used in Digital Experience for Communications. Use the same tax application and a single taxation gateway instance in both Buying Experience and the billing system. This ensures that all the taxes are calculated using a single instance.

Integrate with Avalara AvaTax

You don't have to configure anything in Buying Experience to enable the Avalara tax integration. If a tax application isn't configured, the application directly connects to the Avalara tax integration by using the taxation adapter in CX Industries Framework.

You only need an active Avalara AvaTax account to use this integration. For information on how to sign up for an account and set up the account, refer to the documentation in the Avalara Help Center.

If you don't have an active Avalara AvaTax account at the time of provisioning, this integration is disabled. The tax amount appears as 0 (zero) in the buying and ordering journey. If you create an Avalara account later, you must raise a service request on My Oracle Support at https://support.oracle.com to enable this integration.

Integrate with External Tax Applications

If you're using any third-party tax application other than Avalara, you must build the integration to use the external tax application for calculating taxes. To build the integration, create an adapter in the external tax application and register that adapter in CX Industries framework to connect to Buying Experience.

This adapter is responsible for transforming the data shared between the applications. For more information on integrating the external tax application, see the Integrate External Applications chapter in the Implementing CX Industries Framework guide in CX Industries Framework (2720527.1) on My Oracle Support.

Related Topics

13 Migrate Bulk Data

Migrate Entity Data

You can migrate a large set of records of various entities from one system to the Buying system using the bulk migration APIs. Currently, the following TMF entities are supported:

- Party
- Customer
- PartyAccount
- Product (Asset)

Here are some points to consider before you get started with the bulk migration task:

1. You must be a **Bulk Job Administrator** to perform this task. You must integrate Buying Experience with Oracle Content Management (OCM). For details, see Integration with Other Applications chapter in this guide.
2. Create a folder in Oracle Content Management (which will act as the parent migration folder) for storing the migration data files. You can then add the confidential application that you created in the Oracle Identity Cloud Service as a Downloader or Owner for the download access.

Bulk migration of existing subscriber data is implemented using the following approach involving two steps:

- The first step is called createJob where the JSON data is read from Oracle Content Management (OCM) and inserted into a staging table.
- The second step is called processJob where it reads the data from staging table and migrates to Buying system.

As a prerequisite, party account migration is required to be completed in order to migrate the asset data corresponding to the given account. The JSON files for migration of entity data can be placed in any directory.

You can use the createJob POST API to submit the job to perform the first step. It will return the response with jobId. You can invoke the GET API by using the jobId to track whether the job is in progress, is successful, or is in error. By using the GET API, you can monitor the job status. After the job is successful, you can run the processJob POST API to migrate the data. Here again, you can use the GET API to get the status. If any of the jobs fail, you can use the error GET API to see the error message.

1. Call the createJob API using the input payload provided in the Buying Experience REST API guide. You can run this cURL command or use a REST API client to call the createJob API:

   ```
   curl -H 'Authorization: Bearer <accessToken>' https://<hostName>/cx/industry/buying/bulk-api/v1/
   migration/createJob -X POST -H "Content-Type: application/json"
   ```

   Your payload must contain the directory ID. Here's the sample payload for the create job API:

   ```
   {
    "id": "FF6C2D52A0BFE87B52829F13EB2", # OCE folder Id
    "entityName": "Product" # name of the entity
   }
   ```

2. Call the processJob API using the jobId that you have from the Step 1. You can run this cURL command or use a REST API client to call the processJob API:

```
curl -H 'Authorization: Bearer <accessToken>' https://<hostName>/cx/industry/buying/bulk-api/v1/
migration/<jobId>/processJob -X POST -H "Content-Type: application/json"
```

To view the status of the job:

Call the GET API using the jobId that you have from the Step 1 or Step 2. You can run this cURL command or use a REST API client to call the GET API:

```
curl -H 'Authorization: Bearer <accessToken>' https://<hostName>/cx/industry/buying/bulk-api/v1/
migration/<jobId> -X GET
```

To view the error details in case if the job fails:

Call the GET API using the jobId that you have from the Step 1 or Step 2. You can run this cURL command or use a REST API client to call the error API:

```
curl -H 'Authorization: Bearer <accessToken>' https://<hostName>/cx/industry/buying/bulk-api/v1/
migration/{jobId}/error -X GET
```

After these entity data are migrated into Buying, it's possible to transact on these data for various operations from Buying.

Migrate Contract Document References

Migration of contract reference from an external system to buying is supported.

As part of the support, the reference of the contract documentation is migrated. Support for query of the contract details is enabled. Also, support for download of the contract document using API is provided.

Work with Migration Job Events

There are two migration job events published that denote the different life cycle stages of the migration job.

1. MigrationJobCreateEvent: The event gets published when a new migration job is created.
2. MigrationJobStateChangeEvent: The event gets published when there is any change in the status of the migration job. This is helpful to know the job status in special scenarios (such as when the job goes to error state) without using the GET API repeatedly to monitor the job status.

14 Sync Transaction Data

Sync Account, Product Order, and Asset Data

The process involves syncing of entity data from Buying to an external system and syncing the data from an external system to Buying through various Buying listeners.

This includes entity data such as party, customer, party account, and product order, with after-market extensions. As part of this process, key information such as the external system code and the entity ID in the external system are captured.

Support is available to update product (asset) attributes such as group name and alias from Buying to an external system and vice-versa. You can modify assets through actions such as package change, suspend, resume, and terminate from Buying or from an external system through a new order or shopping cart.

In addition to the existing listeners, there are two more listeners that are now available:

- **productOrderSyncEvent Listener**: Creates and updates product orders in Buying when these orders are coming in from an external system.
- **productAttributeValueChangeEvent Listener**: Supports product attribute updates for group name and alias when these are updated from the external system.

Party, Customer, and Account

Here's how sync occurs for the party, customer, and account:

Syncing External System Account related Data to Buying

1. Buying has listeners for Party (Individual), Customer, and Party Account when these entities are created, updated or have a state change initiated from an external system.
2. When the external system generates events related to these entities, an adaptor should convert this information received from the external system into a TMF format that Buying understands and can pass on to the CX Industries Framework.
3. CX Industries Framework calls the appropriate Buying listeners to create or update the corresponding entity data in Buying.
4. When creating a party account in Buying, if Party and Customer entities aren't available in the external system, these entities are created in Buying.
5. The entitlements are created for these entities appropriately.

Syncing Account related Data to External System

1. Buying generates events for create or update, and state change for the entities such as Party, Customer, or Party Account.
2. External systems can listen to these events and take appropriate actions.

Product Order

Here's how sync occurs for the product order:

Syncing External System Product Order Data to Buying

1. Buying has a listener for Product order create and update happening from the external system (ProductOrderSyncEvent listener).

   When the external system creates or updates the product order, an adaptor should convert this information received from the external system into a TMF format that Buying understands and can pass on to the CX Industries Framework.
2. CX Industries Framework calls the Buying listener and orders are created or updated in Buying.
3. When the external order is fulfilled, the Buying listener for productOrderStateChange event is initiated and the order is completed in Buying.
4. At this time, the external Asset ID and the contract document references that are created in the external system too are captured.
5. Buying then initiates the auto-asset process against this external asset reference.

Syncing Buying Product Order Data to External System

1. Buying generates events for create or update, and state change of product orders.
2. External systems can listen to these events and take appropriate actions.

Note: Orders created in external system can't be updated in Buying. Similarly, for orders created in Buying, these shouldn't be updated from the external system.

Product (Asset)

As mentioned in the section on Product order sync, Product (Asset) is not synced from the external system but the external Asset references and the related contract document references are captured when the order state is completed (productOrderStateChangeEvent listener). Buying will do the Auto-asset based on the order status received as Completed.

However, certain asset attributes such as Group Name and Alias can be updated in external system and synced to Buying and vice-versa. The productAttributeValueChange event and the corresponding listener are used for the same.

Exception Handling During Transactional Data Sync

You can identify and handle exceptions raised when data from external system is synced into Buying service through event listeners.

When data is coming in from an external system, the listener can receive the entity data but can't process the data due to functional errors. Examples of such functional errors include missing or incorrect values for required attributes and issues with reference entity. For example, party account is the reference entity for product order.

You can retrieve the error information with details on the entity, the source system, the reason, related entity errors, along with status. Based on it, you can correct the incorrect data and send again for sync into Buying. Ater this is completed, the error record status can be set to resolved. Status of error records can be marked for verification too as required and after verification, they can be updated as resolved.

15 Manage Lifecycle of Data

Manage Lifecycle of Data across Instances

Here's the process for migrating functional data across instances such as test to production.

You can move the data belonging to these categories from test instance to production:

- CX Industries Framework technical configurations such as routing rules and gatekeeping rules.
- Initiatives defined in Launch that include product offers, pricing, adjustments and Digital Experience for Communications common business configuration (such as geo location attributes, account type, and market segment code) and also offer info extensions.
- Buying-specific business configurations such as subscriber user key configuration, shopping cart abandonment or purge, and outbound events.
- Buying entity extensions.

In addition to these, there are other data such as product images and collaterals (Launch), contract templates, bulk product order templates, and data migration files (Buying) that will be stored in Oracle Content Management (OCM) cloud - Test instance. These must be moved to the OCM production instance.

Process Flow

1. After the initiatives are tested in the Launch Experience test instance, these are exported from the test instance and then imported into Launch Experience production instance, through the export-import mechanism.
2. CX Industries Framework configurations are exported from CX Industries Framework test instance to CX Industries Framework production instance as appropriate, through the export-import mechanism.
3. CX Industries Framework and Buying entity extensions must be created in the production instance either manually or automated by executing version of the Postman that contains entity extension scripts.
4. The functional data now available in the Launch production instance is synced to the Buying production instance. This includes product offers and Digital Experience for Communications common business configuration data.
5. Buying-specific business configuration data that must be available in the Buying production instance can be handled in any one of these ways:
    - Manually added to the Buying production instance.
    - Through a script that can query configurations defined in test instance using GET APIs and then import them into the production instance using POST APIs.

16 Manage Order Operations and Fallout

Work with Order Operations and Fallout

Business operations specialist works with order operations and fallout.

Order Operations

It provides a consolidated view of order statistics. Using order operations dashboard, you can monitor order throughput including those in fallout.

You can:

- View order throughput, orders created, orders in fallout, orders by status (in progress or pending), and fallout orders by date range.
- Select predefined date range, or provide custom date range (minimum range is 1 day and maximum is 30 days) by specifying start-by and end-by dates.
- View changed data based on the selected date-range.

Fallout Management

This serves as an interface for you to see the list of orders that are in fallout. Using fallout management, you can retry the orders gone into fallout due to the destination system being unreachable.

You can:

- View orders in fallout based on the date range.
- Select orders and retry or resubmit them in batch.
- View status of retry or keep retrying.

Use the refresh option to view the current data.

> **Note:** At a time, the user can retry only 25 orders.After the orders are submitted, they move out of the fallout orders list.