Using Oracle Commerce Agent Console

F56126-01
July 2022

Using Oracle Commerce Agent Console,

F56126-01

Contents

4 View Orders

5 Understand Accounts

6 Manage Orders

7 Customize the Oracle Commerce Agent Console

8 Work with Navigation Widgets

9 Configure Search Widgets

10 Create and Edit Orders Widgets

1

Understand the Agent Console

Customer representatives use the Agent Console to access all the shopper, order, and other information necessary for performing support tasks related to your online store.

The customizable, widget-based Agent Console provides options to modify the application for your specific business requirements, giving you control over what your agents see when performing their tasks. Commerce provides a fully-functioning default version of the Agent Console that you can use if you do not require any customization or if you are interested in seeing how the Agent Console works.

This guide describes the default version of the Agent Console, as well as information on customizing the Agent Console. Note that the Agent application can only be used in Storefront Classic. Hence, this guide is only applicable if you are running Storefront Classic.

Before accessing the Agent Console, contact your system administrator to set up the proper permissions to use the service. Ensure you have a current user ID and password for a profile with the privilege CS Agent or CS Agent Supervisor.

The Commerce Administrator privilege does not grant access to the Agent Console.

You should also be provided with the browser address for logging into the Agent Console. For a list of supported browsers, see Supported Browsers in Commerce.

Navigate the Agent Console

In your browser, enter the page location provided to you by your agent supervisor or systems administrator.

Log into the Agent Console using your credentials (user ID and password).

If you require a password reset, click the **Can’t Sign In?** link. Clicking the link displays the **Reset Password** page. Enter your email address, and click Send Request. An email is sent to the address with a link to a page where you can reset your password. The **Update Password** page requires your user name, the new password, and password confirmation.

The dashboard you see after you log in provides graphs and lists related to agent-specific orders, including details pertaining to return requests and pending actions. The price group menu provides a graph showing the number of orders and order value in each selected price group. By default, your storefront has a minimum of one price group. Lists are also provided to show announcements and quick links to assist with agent productivity.

Use the icons on the top right side of the page to open an order or customer search, access return processing, access help, or log out of the Agent Console.

Accessibility of the Agent Console

Oracle software and documentation includes features that make information available to users of assistive technology. For more information, visit the Oracle Accessibility Program Web site at http://www.oracle.com/us/corporate/accessibility/index.html.

About keyboard shortcuts

Commerce supports standard keyboard shortcuts that are used in many software applications. The following table describes general keyboard shortcuts for use with the Agent Console.

Keyboard Shortcut	Effect
Tab	Navigates to the next focusable element. Use the Tab key to change the focus from the Dashboard icon to the User icon.
Shift+Tab	Navigates to the previous focusable element.
Up and Down Arrows	With a menu field in focus, use the up and down arrows to change the items within the menu.
Right and Left Arrows	Use right and left arrows to navigate in the graphs. Also, use the right and left arrow keys to navigate among the Dashboard, Orders, Customers, and Returns Processing icons that appear at the top of every page in the Agent Console.
Space	Opens a menu field. Use the up and down arrows to navigate within the menu. Click the Enter key to select an item from the menu.

Note: For keyboard users navigating the Agent Console in Firefox on a Mac, tabbing to the Can't Sign In? link on the login screen does not work as expected. To fix this issue, follow these steps:

1. Select **System Preferences** on the Mac.
2. Select **Keyboard** and display the **Shortcuts** tab.
3. Select the **All Controls** option.

2

Configure Agent Console Settings

Configuration options in the Agent Console allow you to modify product functionality, such as the length of the customer remorse period, as well as user interface display and navigation elements, such as the search feature.

To make changes to the Agent Console settings, you must have administrator permissions.

Assign roles

Each user accessing the Oracle Commerce Agent Console must have a valid account with a role containing a privilege that grants access to the Agent Console.

A default Administrator account is included with your Oracle Commerce instance. The administrator creates additional user accounts as required and assigns them the appropriate Agent Console role. Oracle Commerce provides two predefined roles that you can use: the CS Agent role, containing the Agent privilege, and the CS Agent Supervisor role, containing the CS Agent Supervisor privilege. The administrator can also define custom roles that contain these privileges.

The CS Agent Supervisor can process manual refund adjustments, carry out price overrides on cart items, and initiate refunds. For more information, see Process returns.

To create an Agent Console user account:

1. Log into the Oracle Commerce administration interface using your administrator user ID and password.
2. Click the **User Management** icon.
3. Click **New User**.
4. Enter the information that identifies the new user, and select a role containing the CS Agent or CS Agent Supervisor role.
5. Click **Create**.

If a user who does not have the CS Agent or CS Agent Supervisor privilege tries to log into the Agent Console, he or she receives a dialog stating the user has entered an incorrect user name or password.

Understand role-based access control

Roles and access rights can affect how the agent works with the Agent Console.

The agent can process orders in two ways:

- Exclusively in the Agent Console; or
- On behalf of the shopper in the storefront.

If your administrator has placed access restrictions on properties of shoppers or orders, such as properties that hold shoppers' personal data, then an agent's roles and access rights may

determine his or her access to those properties. This access can affect whether the agent can process an order for a shopper or whether the shopper is required to complete the order.

Use roles or access rights or both to provide the agent with access to properties required for placing an order or processing returns. For more information on creating agent roles and access rights, see Implement role-based access control for internal users in *Extending Oracle Commerce*.

Understand the effects of roles and access rights

When an agent uses the Agent Console to create or edit an order, or to process a return for a shopper, the agent's roles and access rights may determine which properties are displayed and which can be updated.

To create an order, an agent must have access to the following:

- Shopper profile information (such as first and last name and email).
- Shipping and billing addresses.
- Shipping methods.

If roles or access rights restrict the agent from viewing any of this information, the agent cannot process the order. A shopper may also restrict an agent's ability to view personal data. In either case, it would be the responsibility of the shopper to complete the order.

Understand roles when shopping as a shopper

Roles and access rights determine the properties the agent can see when working in the Agent Console. However, when the agent shops on behalf of a shopper in the storefront, only agent roles—not access rights—are used. The roles control the agent's access to properties such as a shopper's personal data. If you restrict access to a property based on access rights, the property is not displayed to the agent on the storefront.

If the shopper does not grant permission for any of this personal data to be viewed, or if the agent role limits the agent from viewing any of this information, the agent is not able to create an order. The shopper would have to complete the order.

Create announcements and quick links

Your business can create announcements and quick links for your agents to see on the dashboard.

To create announcements and quick links:

1. Log into the Oracle Commerce dashboard using your admin user ID and password.
2. Click the **Settings** icon.
3. Click **Agent Console Settings**, and then the **Announcements** or **Quick Links** tab.
4. For announcements, do the following:
    a. Click **New Announcement**. The **Announcements** dialog appears.

b. Use the text field to create the announcement. Text tools are available to customize the announcement.
c. Click **Enabled** to publish the announcement to the agent dashboard.
d. Click **Create**.

5. For quick links, do the following:
   a. Click **New Link**. The **New Link** dialog appears.
   b. Enter the link title text and the URL for the link in the fields.
   c. Click **Enable** to publish the quick link to the agent dashboard.
   d. Click **Create**.

Set price overrides

Your business may permit agent supervisors to override order prices in two situations: during order creation and during order amendment within the remorse period.

To set the price override permissions:

1. Log into the Oracle Commerce dashboard using your admin user ID and password.
2. Click the **Settings** icon.
3. Click **Agent Console Settings** and display the **Price Override** tab.
4. Select or clear the appropriate checkboxes: **Order creation** and **Order amendment**.
5. Click **Save**.

Set the customer remorse period

A remorse period is a prescribed amount of time, for example an hour or twenty-four hours, during which a shopper can cancel or amend an order.

The order is not fulfilled until the remorse period is over. Note that this concept is not the same as the length of time your business may allow for returns.

Important: As a best practice, keep the remorse period as short as possible to avoid delays in fulfilling orders. Order submission to the fulfillment system – including webhook activity – is suspended for all shoppers until the remorse period is concluded.

To configure the remorse period, complete these steps:

1. Log into the Oracle Commerce dashboard using your administration interface user ID and password.
2. Click the **Settings** icon.
3. Click **Agent Console Settings** and display the **Remorse Period** tab.
4. Select the **Enable Remorse Period** checkbox.
5. Enter the hours and minutes that correspond to the amount of time your business designates as the remorse period.
6. Click **Save**.
7. Click on the **Extended Remorse Period** tab to enable and determine the number of days after the date of submission that an order can be canceled.

8. Click **Save**.

Manage account-based commerce accounts and roles

Shoppers who place orders using an account are assigned roles that allow them to perform different types of activities in your store.

Account-based shoppers can have the following roles:

- Administrator
- Buyer
- Approver
- Account Address Manager
- Profile Address Manager

Administrators can make purchases and are also authorized in the Agent Console to add new contacts to the list of account buyers, administrators, approvers, and address managers. All edit and update actions can only be done by a delegated administrator of the account.

As a buyer, a shopper is authorized to make purchases on the account. By default, every contact assigned to an account receives the buyer role.

An account approver is able to approve purchases that exceed the account spending limit. Approvers may also be able to approve an account's order if the account's contract expires.

To set account-based commerce content and roles:

1. Log into the administration interface.
2. Click the **Accounts** icon, and click the appropriate account link.
3. Click **Contacts**.
4. To edit the contact's permissions, click the **contact** link.
5. On the **General** page, select the checkbox to make the contact active or clear it to make the contact inactive.
6. On the **Account Memberships** page, make changes to the storefront roles and add accounts to the contact.

Conduct account administration

In the Oracle Commerce administration interface, click Accounts to do the following:

- Add and view details for current accounts for your site.
- Add and edit contacts to an account.
- Add or edit account addresses, including setting up default addresses.
- Set up approvals, including purchase limits.
- Make the account active or inactive.
- Identify buyers who are connected to other accounts.
- View and edit account contracts.

- View account contact lists, including editing status and storefront roles.
- Set email preferences per site or turn off email updates to all sites.

For more information, see Configure Business Accounts in *Using Oracle Commerce*.

Set the price hold period

For account orders with pending payments, your business can set a price hold period during which the price of the order will not change. After the period passes, the order is cancelled.

To set the price hold period:

1. Open the administration interface.
2. Click the **Settings** icon.
3. Click **Payment Processing**.
4. Click **Setup**.
5. Enter the amount of time in days and hours to allow the shopper to provide missing payment information before an order is cancelled. The order's prices will not change during this time.
   If you leave both Days and Hours blank, there is no time limit.

For more information, see Set a price hold period in *Using Oracle Commerce*.

Customize displayed text

Oracle Commerce uses text snippets to customize the text for state values displayed on your store and in the Agent Console.

These state values include order states, shipping item relationship status, and payment group status. The state values appear in the search menus for customer and order searches.

Text snippets can be edited in the administration interface, under Design, Code tab, Advanced. For more information on using text snippets, see Modify Your Storefront Using Code Editing Tools in *Using Oracle Commerce*.

Configure customer and order search

By default, the Agent Console uses Repository Based Search to search for orders and customers. It can be configured to use `textSearch` instead through the `textSearch` Admin endpoint.

When `TextSearch` is enabled, the Agent user interface uses `TextSearch` internally. When `TextSearch` is disabled, the Agent user interface uses Repository Based Search.

To enable `TextSearch`, issue a `PUT` request to the `textSearch` endpoint as follows:

```
PUT /ccadmin/v1/merchant/textSearch
 {
  "enable": true,
  "scheduledJobs": [
    {
      "scheduledJobName": "orderIncrementalIndexingSchedule",
      "scheduleType": "periodic",
      "schedule": {
```

```
        "period": 20000
      }
    },
    {
      "scheduledJobName": "profileIncrementalIndexingSchedule",
      "scheduleType": "periodic",
      "schedule": {
        "period": 20000
      }
    },
    {
      "scheduledJobName": "orderOptimizationSchedule",
      "scheduleType": "calendar",
      "schedule": {
        "occurrenceInDay": 2
      }
    },
    {
      "scheduledJobName": "profileOptimizationSchedule",
      "scheduleType": "calendar",
      "schedule": {
        "occurrenceInDay": 1
      }
    }
  ]
}
```

The `scheduledJobs orderIncrementalIndexingSchedule` and `profileIncrementalIndexingSchedule` are responsible for ensuring that new customers and orders and the latest modifications made to these entities are searchable.

Determine an appropriate schedule for these jobs based on an acceptable interval to search for a customer or order after it is created or modified. Since `TextSearch` is enabled as default, these jobs are configured to run every 20 seconds.

The `scheduledJobs orderOptimizationSchedule` and `profileOptimizationSchedule` are responsible for improving search performance by running an optimization service.

Determine an appropriate schedule for these jobs based on how search performance degrades with the increase in the rate of updates and creations of orders and customers. It is recommended that a schedule frequency of not more than once per day be established.

After issuing a request to enable `TextSearch`, an offline setup process is initiated which may take a few minutes or more. The setup status can be queried by issuing a `GET` request to the same endpoint.

If `TextSearch` was already enabled, issuing another request to enable `TextSearch` updates `scheduledJobs` with the new schedule settings and does not initiate any offline setup process.

To disable `TextSearch`, issue a `PUT` request to the `textSearch` endpoint as follows:

```
PUT /ccadmin/v1/merchant/textSearch { "enable": false }
```

For more details on this endpoint, refer to the Agent REST API. For more information on how to provide schedule settings, see Configure the scheduled order service in *Extending Oracle Commerce*.

Create account custom properties

Using the API, administrators can create contact, address, and account level custom property fields that are displayed on the Customer Profile page.

These properties can be viewed or edited by the agent acting as a delegated account administrator or the actual delegated account administrators.

For more information, see Add custom properties to a shopper type in *Extending Oracle Commerce*.

Understand dynamic commerce item properties

Your business may establish custom properties for commerce line items in your catalog (for example, engraving for a watch or piece of jewelry, or personalizing for a mug).

However, because of the nature of these dynamic commerce item properties, all available properties may be displayed even if the properties do not apply to the product (for example, engraving for a camera). Best practices include creating titles for the custom properties to distinguish one from the other (for example, Watch – Engraving or Mug – Personalize).

When these custom properties are available, a Customize link is displayed under the product. Clicking the link displays all the properties. You can then assign values to these properties.

In addition, a Use these properties for all the items option is provided. This option is displayed only if there is more than one quantity for a product. If this option is cleared, the item can be split into multiple items of one quantity each. All of the new items can have different values for the customization properties.

For example, a shopper is buying three mugs, and a custom property called Cup Caption has been created. If the Use these properties for all the items option has been cleared, the customization properties appear three times, once for each mug. You can provide different captions for each of the shopper's mugs. The result is three different items in the cart with one quantity each, instead of the initial one item with three quantities.

Collect and delete shopper information

The Agent Console uses custom properties to allow agents to manage shopper consent for the collection of personal data.

The shopper information deletion/retention feature promotes compliance with the EU General Data Protection Regulation (GDPR), a set of legal requirements designed to control the collection and storage of personal data. For more information, see Manage the Use of Personal Data.

In the Agent Console, to allow agents to collect consent for retention of shopper personal data, implement custom properties and add them as needed to appropriate pages, for example the pages agents use to create and edit orders. For more information, see Create custom properties for orders.

Oracle Commerce provides endpoints in the Agent API that enable you to support GDPR right to erasure. Use these endpoints to remove personal data by redacting orders and

deleting shopper profiles. For more information, see Delete shopper information in *Extending Oracle Commerce*.

Set agent framework version

You can change the version of the agent libraries used in your framework.

When the agent library version is updated, you have two different versions available to you. You can use the *current* version, which is the updated and newest version, or you can use the *latest* version, which is the version you had previous to the update.

To display any changes to the libraries in the agent's administration interface, set the `OptimizationOptions_production.loadModuleInParellel` component in the `index.js` file to `true`. Note that the agent console does not support loading libraries form an external system.

Identify the agent libraries version

To see which version of the agent libraries is current, issue a `GET` command to the `agentVersion` endpoint. For example:

```
GET  /ccadmin/v1/jsframeworkadmin/agentVersion
```

This will display a response similar to the following:

```
{
  "success": true,
  "previousAgentVersion": "",
  "agentVersion": "19.5.6-SNAPSHOT",
  "links": [
    {
      "rel": "self",
      "href": "http://localhost:9080/ccadmin/v1/jsframeworkadmin/
agentVersion"
    }
  ],
  "currentAgentVersion": "19.5.4-SNAPSHOT"
}
```

Upgrade the agent libraries version

If you want to upgrade the libraries that your framework is using, issue a `POST` command using the `upgradeAgentVersion` endpoint. For example:

```
POST  /ccadmin/v1/jsframeworkadmin/upgradeAgentVersion
```

This returns a response similar to this, showing the previous version and the new current version:

```
{
  "success": true,
  "previousAgentVersion": "19.5.4-SNAPSHOT",
  "agentVersion": "19.5.6-SNAPSHOT",
```

```
  "links": [
    {
      "rel": "self",
      "href": "http://localhost:9080/ccadmin/v1/jsframeworkadmin/
upgradeAgentVersion"
    }
  ],
  "currentAgentVersion": "19.5.6-SNAPSHOT"
}
```

Rollback the agent libraries version

If you want to roll the library versions back to the previous version, you must issue a `POST` command to the `rollbackAgentVersion` endpoint. For example:

```
POST  /ccadmin/v1/jsframeworkadmin/rollbackAgentVersion
```

This returns a response similar to the following:

```
{
  "success": true,
  "previousAgentVersion": "",
  "agentVersion": "19.5.6-SNAPSHOT",
  "links": [
    {
      "rel": "self",
      "href": "http://localhost:9080/ccadmin/v1/jsframeworkadmin/
rollbackAgentVersion"
    }
  ],
  "currentAgentVersion": "19.5.4-SNAPSHOT"
}
```

Configure returns and exchanges tax calculation dates

You can configure the date used for tax calculations of returns and exchanges.

You can use the `useOrderSubmittedDateForTax` flag in the merchant settings Admin API, to use the order's submitted date for tax calculations. This allows you to calculate tax refunds for returns using the original order date, rather than the current date. This sends the date of the original order to external tax processors, such as Vertex and Avalara.

To modify this setting, issue a `PUT` command to the `/ccadmin/v1/merchant/useOrderSubmittedDateForTax` flag to `true`. For example:

```
{"userOrderSubmittedDateForTax":true}
```

To continue using the current date for tax calculations, set the `useOrderSubmittedDateForTax` flag to `false`.

If you are using Oracle Commerce 20.1 or earlier, the default for this flag is set to `false`. Later versions of Oracle Commerce have this flag default to `true`.

3

Search for Customers and Orders

The search functionality in Agent Console allows your agents to locate customers and orders in the system.

Search for customers

In the Agent Console, many of the tasks an agent performs for customers are initiated by searching for a customer profile. To start the search, click the Customers icon in the Agent Console dashboard and use the Customer Search page to locate the profile you want to work with.

The Customer Search page includes following tabs:

- Customer Search
- Registration Requests: Use to view shopper requests for inclusion as buyers on an account. See Search for and view account and contact registration requests.

Use the following fields to search for customers in your system:

- First or last name
- Email
- Phone number
- Zip code
- Account (for account-based commerce sites)

For all fields, the search feature uses a Starts With pattern. For example, typing K in the First Name search field could produce results of Kara, Kim, and Kyle. Typing Ki would limit the search to Kim. You can use either letters or numbers.

Create a new customer

If you need to enter a new shopper into the system, click **Create New Customer**. Select the checkbox to provide site updates to the shopper via email. You can also see any custom properties that have been created for shopper profiles.

When creating a new customer profile in a multiple site environment, do the following:

1. In the **Select Site** menu, choose the site the shopper wants to be associated with initially.
2. Select the **email updates** checkbox to enable email notifications for the selected site only.
3. To manage email notifications for multiple sites (opt-in or opt-out of notifications for the sites associated with your business), go to the shopper's profile. The right menu displays the current site properties for the shopper.
4. Under Email Preferences, select or clear the checkbox to enable or disable email notifications for the site.
5. In the Site field, select a different site to tailor shopper notifications for each site.

Use the Profile Details tab

Agents can use the Profile Details tab to view a shopper's basic information and assist the shopper with profile-related tasks. The tab also provides access to order-related activities, for example placing or completing an order for this shopper.

- View and edit the shopper's first or last name. The email address cannot be edited by an agent.
- Reset the shopper's password.
  Clicking Reset Password immediately sends a password reset link to the customer at the email address shown.
- Edit any available custom properties, both internal and customer-facing.
  Customer profile attributes can be edited at any time on the Customer Profile page. If a field is designated as required, a default value is supplied.
- Create a new order or complete an existing incomplete order.
  If you have a single-site environment, the Complete Order link displays the number of items currently in the shopper’s cart as well as the order amount; for example, Complete Order - 2 items - $79.98.

  If you have a multiple site environment, use the menu to select a site to perform actions specific to that site. You can start a new order for the site or select the link to show the details of the incomplete order.
- As an agent, launch the storefront and shop on behalf of the shopper.
  Click the Launch Store as Customer button to open the storefront application as this shopper. For more information, see Understand shopping on behalf of the shopper.
- View current loyalty point details.
  If your business uses loyalty points for your shoppers, the details for the loyalty program are listed, including the program ID/name, shopper membership number, category, and total loyalty points accrued.

  For more information, see Work with Loyalty Programs.
- View and edit contact level custom properties.
  Using the API, administrators can create contact level custom property fields that are displayed on the Customer Profile page. These properties can be viewed or edited by the agent or the delegated account administrators.

Use the Address Book tab

Agents can use the Address Book tab to view and manage addresses associated with the shopper.

You can add or edit custom properties associated with an address. Custom properties for addresses are created by an administrator through the API. For more information, see Add custom properties to a shopper type.

For those account-based commerce users who are strictly buyers, you cannot add or edit the shipping or billing address information as it is set for the account associated with the buyer. When you are creating the new address on behalf of the customer, the default shipping country that is populated is specific to the site selected on the Customer Profile page.

Buyers with the role of Account Address Manager, Profile Address Manager, or Administrator can add, edit, or delete addresses for specific areas of the account. Account address managers cannot set default addresses.

Sub-accounts inherit the addresses of the main account and are listed in the Inherited Addresses section.

Addresses can contain first and last names other than the shopper of record.

Use the Order History tab

Agents use the Order History tab to view and manage the shopper's past orders.

Click the Details button to view information about each order, including payment status, and access tasks such as emailing order details to the customer or copying the order.

For more information, see Manage Orders.

Use the Scheduled Orders tab

Agents can use the Scheduled Orders tab to view and manage any scheduled orders the shopper has placed.

Click the Name link for complete details about the order. You can make changes to the scheduling details but not to the items included in the order. For more information, see Schedule an order.

Use the Purchase Lists tab

You use the Purchase Lists tab on the Customer Profile page to view or create lists of items the shopper may be interested in purchasing at a later time.

The shopper adds items to a purchase list by clicking the Add to Purchase List button on a product details page. A down arrow button next to the button allows the shopper to add the item to one of their purchase lists or create a new purchase list.

As a customer service representative, you can open and add items to one of the shopper's current purchase lists. You can create a new purchase list by clicking Create a Purchase List button and entering a name for the new list, as well as adding items from the Order Details pages. Items are added to new or existing lists by SKU or product name.

To add items from a purchase list to a new or existing order:

1. On the shopper's profile page, click the **Purchase Lists** tab.
2. Click the link for the purchase list where the item is listed.
3. Select the checkboxes for the item or items you want to include in the order, and click **Add to Order**. The **Create Order** page is displayed with the items you have selected. If this is a previously created order that requires completion, the items previously added are also included in this order.
4. If the shopper wants to include other items from other purchase lists, on the **Create Order** page you can add the items in the cart to any of the available purchase lists. The available lists can be viewed using the Add to Purchase List menu. If a custom-defined purchase list is not available, you can select **Create New Purchase List** and define your own list. Sharing of a purchase list is available using the shopping on behalf of shopper feature. To do this:

- On the shopper's **Profile Details** tab, click **Launch Store as Customer** to act on behalf of the shopper. You can then share purchase lists as directed.

For more information, see Understand shopping on behalf of the shopper.

Use account-related profile tabs

For account-based commerce buyers with a delegated admin role, additional tabs are displayed in the Customer Profile page.

- Account Contacts: includes the standard shopper information with the addition of buyer purchase limit, storefront role, whether this buyer is connected to other accounts, and status (active or inactive) of the account. The properties displayed are specific to the account, so all contacts in this account honor these settings.
- Account Details: includes account status, description, classification, type, and DUNS, VAT, and tax reference numbers.
- For account-based commerce buyers with the Approver role, an Order Approvals tab is displayed. This tab shows orders made by buyers who have exceeded their account level purchase limit and require approval prior to order placement. The tab includes order ID, order date, submitted by (buyer), approval reason, and order total columns. A search field is provided to filter the results.

All tabs may include custom properties set up by the account administrator.

Understand account-based commerce customers

If account-based commerce functionality has been enabled for your environment, your agents should understand the following:

- Account-based commerce buyers purchase for an account and use contract pricing when ordering. For example, item prices and order discounts may be different for these shoppers. In addition, pricing may also be set up to reflect volume-based pricing, which can include bulk and tiered purchases. Calculations and catalog choices are made automatically by Oracle Commerce.
- Approval settings, for example, require approval, and purchase limits are site- and account-specific. Settings change automatically when a site/account combination is selected.
- If one is enabled, you can include a purchase approval management system to incorporate the organization's approval rules.
  For more information, see Enable Order Approvals.
- On account-based commerce buyer profile pages, information is provided to show the buyer's account level purchase limit, storefront role (Admin, Approver, Buyer, Account Address Manager, and Profile Address Manager), and whether or not the buyer's status is active.
- An account-based commerce buyer may be associated with one or more than one account. Use the Account Name menu to select each account you want to view associated with the buyer.
- Account-based commerce associates a price group with a contract and this price group setting cannot be changed.
- Account address managers and administrators are the only roles that can edit, delete, or set to default shipping and billing addresses for the account.

- Profile address managers can edit or delete an address but cannot set as default any billing or shipping addresses.
- Depending on your business model, the account-based commerce buyer may be able to use invoices (purchase orders) as well as credit cards. Invoice and credit card payment options for order amendment are the same as for other shoppers: once a credit card is used, the invoice option is no longer available.

Important: Store credit is not available as a payment option for account-based commerce accounts.

If the order amount changes during order amendment, and the shopper wants to use a credit card instead of the original invoice, there are no restrictions to the combinations she or he can use: combine invoice with any payment method, such as credit card, gift card, or cash, depending on your organization's setup.

For more information, see Manage pages for account-based shoppers in *Using Oracle Commerce*.

Conduct shopper profile activities for an account

Accounts are created in the Oracle Commerce administration interface. With delegated administrator permission, agents can do the following:

- Create a new contact under the same account as the delegated administrator.
- Assign, change, or remove roles for a contact.
- Update contact details.
- Search for and view all contacts with roles for the account based on first name, last name, email, or role.
- On the Profile page, view and manage account contact information.
- View, add, edit, or delete addresses for an account.
- Mark and edit addresses as a shipping or billing address.

Note: This capability can be made unavailable to the agent.

An agent acting as a delegated administrator is distinct from an agent shopping on behalf of a shopper. For more information, see Understand shopping on behalf of the shopper.

To manage account-based contacts and addresses:

1. Establish the identity of the shopper as an administrator for the account. Contact your manager for the method by which you are to identify administrators.
2. Search for the contact and open the customer profile. For more information, see Search for customers.
3. On the **Customer Profile** page, **Profile Details** tab, under **Storefront Role**, if the person you are dealing with is an administrator, the **Administrator** checkbox will be selected. At a minimum, the **Buyer** role is selected. Additional roles include **Approver, Account Address Manager**, and **Profile Address Manager**. Also ensure the **Active** checkbox is selected.
4. For all account delegated administrators, an additional tab called **Account Contacts** is available. Click the **Account Contacts** tab to view all contacts registered to the account. Information on this tab includes:
    - First and last name.

- Active column: indicates whether the contact is currently active.
- Storefront role: buyer, approver, profile address manager, account address manager, and/or administrator.
- Email address.

5. Use the filter field to narrow the search for specific contacts.
6. Click the **New Contact** button to register a new contact to the account. The **Add New Contact** dialog appears.
7. Enter the first and last name, email address, the storefront role, and whether the contact is active. By default, the new contact has Buyer selected. Click **Save**. The new contact now appears in the **Account Contacts** list.
8. From the **Account Contacts** list, click a current contact first or last name link to edit the information. Make any changes to the storefront role and active status here. For these currently-registered contacts, the email address cannot be edited. For more information, see View customer results.
   To add a new address or edit current addresses, click the **Account Address Book** tab. For more information, see Use the address book.

Search for and view account and contact registration requests

Only account administrators have permissions to approve shopper requests for inclusion as buyers on an account. Once the functionality is configured, a shopper can fill out a form on the storefront and submit a request. While agents cannot approve requests, they can search for and view the requests to provide status information to a shopper.

Once the administrator approves a request, the request is cleared from the search and a new buyer is added to the associated account. The new buyer now has privileges based on the access given to him or her, which can include contractual pricing and special catalog selections.

Agents can view the contact registration request form by searching for contact first name, last name, email provided on the registration form and request ID and other criteria. A delegated administrator can act upon a Contact registration request form using on-behalf-of functionality.

To search for and view a registration request:

1. Click the **Customers** icon at the top of the page, and click the Registration Requests tab.
2. In the fields, enter the search criteria provided by the shopper, which may include first name, last name, email, company name, request ID, or other criteria.
3. Click **Search** to display a table showing all the requests that meet your criteria. The Status column provides the information on the current status of the request (for example, Review, More Info Needed, or Rejected) short of approval. Once the request is approved, you must search for a buyer using the **Customer Search** tab.

To view the details of a particular request, click the request ID link. A new page is displayed with the following tabs:

- Request: includes status, ID, request date, origination site, and notes added by the requester.
- General: includes account-specific information, such as company name, DUNS, VAT, and tax reference numbers, and account type.

- Contact: includes requester information, such as name and email.
- Address: includes the requester's address.

The information on these tabs is read-only. There may also be custom properties you want to be collected for each request. The custom properties appear at the bottom of each tab and are tab-specific. Custom properties may be either read-only or editable based on the configuration.

Administrators can also view account information by searching for a buyer and clicking the **Account Details** tab.

For more information, see Manage pages for account-based shoppers in *Using Oracle Commerce*.

Search for orders

On the top right side of the page, click the **order search** icon.

On the **Order Search** page, you can use the following criteria to search for all types of orders in your system:

- Customer first and last name, email address, and phone number. Note that anonymous orders do not contain profile information. Refer below to searching for anonymous orders using profile criteria.
- Order number.
- SKU, product name, order states, order status, and orders submitted previously in days.
- For those companies that have implemented a multiple site environment, the agent can also search for orders by site.
- Account (available if account-based commerce functionality has been enabled for your environment; this column is populated for shoppers with accounts only).
- Approver (available if account-based commerce functionality has been enabled for your environment; this column is populated for shoppers with accounts only).
  Approvers have the ability to approve or reject orders. If the order is approved, the order is placed. If the order is rejected, the order is not placed and its status is changed to Rejected.

Click the **Show Advanced Search** link to show expanded criteria fields, including specific start and end dates, and customer order-level shipping or billing addresses. Click the **Hide Advanced Search** link to display only the standard search criteria fields.

The search criterion is based on Starting With, either letters or numbers. For example, typing a K in the First Name search field could produce results of Kara, Kim, and Kyle. Typing Ki would limit the search to Kim as a related result.

Search for anonymous orders using profile criteria

You can use a standard order search for anonymous orders, which includes searching for an order ID, order state or status, and the criteria listed earlier. However, you can search for anonymous orders using profile criteria by searching for the first and/or last name that is used on a shipping and billing address. Use the **Advanced Search** page to search for either the **First Name** or **Last Name** of the **Shipping/Billing Details** section.

You can also search for anonymous orders by using a shopper's email ID. However, this requires using the Agent API to retrieve the orders. For example, you could issue a REST call similar to the following:

```
/ccagent/v1/orders?q=shippingGroups.email sw ""&&queryFormat=SCIM
```

For additional information, refer to the Agent API documentation.

View order results

The results of your order search appear in the Order Results table.

Orders can have the following types of status:

Status	Description
Being Processed	The order is being prepared for shipment to the customer.
Fulfilled	The order is complete, shipped, and delivered to the customer.
Removed	The order has been canceled and not shipped to the customer.
Submitted to Fulfillment	The order has been submitted to the warehouse staff to begin fulfillment.
Pending Approval	For account-based commerce orders, this order requires approval prior to order placement.
Pending Payment	For account-based orders, the order is approved and is awaiting payments on that particular order.

Orders and return requests belonging to deleted and inactive sites are displayed on the following pages:

- Order Search Results
- Return Search Results

Orders and return requests belonging to the deleted sites cannot be viewed. Those belonging to inactive sites can be viewed, but no operations can be performed on them.

In the table, you can view individual orders by clicking the Order Number link. For more information, see View order details.

Search for return requests

To search for return requests associated with completed orders, click the **Returns Processing** icon.

Completing a search provides you with a list of requests.

To initiate refunds or perform price adjustments to refunded amounts, you must have a role containing the Agent Supervisor privilege. For more information, see Assign roles and Process returns.

4

View Orders

You can view current orders by clicking the Order # link on the Order Results page for all orders in the system or on the Order History page for a specific customer.

The Order page shows the following information:

- Order details. The Order Details page is displayed when the order is in Pending Payment status.
- Order items. Click the image to view the product details page for the item.
- Shipping details
- Payment details
- Order notes
- Any custom order properties, internal or customer-facing. These properties are read-only.
- For account-based commerce orders, a Pending Approval page is available for those orders that have exceeded a shopper's account level purchase limit. A link to return to the account Order Approvals page is provided at the top of the page.

To view an order using an order search:

1. At the top of the page, click the **Orders** icon, and complete an order search by entering information in one or more of the fields. Click the **Show Advanced Search** link if necessary. The Order Results table appears at the bottom of the page.
2. Click the appropriate link from the Order Number column.

To view an order using a customer search:

1. At the top of the page, click the **Customers** icon, and complete a customer search by entering information in one or more of the fields. The Customer Results table appears at the bottom of the page.
2. Access the order details using either the Orders column link or the **Customer Profile** page:
    a. In the Orders column, click the number link. The **Order History** tab appears, displaying the shopper orders.
    b. Click the **Details** button for the appropriate order.
    Or
    a. Click the icon in the Profile column.
    b. On the **Customer Profile** page, click the **Order History** tab.
    c. Click the **Details** button for the appropriate order.

View order details

In the Order Details section, you can access comprehensive information about an order and the customer profile associated with it.

In the Order Details section, click the customer name to display the associated Customer Profile page. You can edit the customer name, reset the customer password, and edit and add addresses to the shopper's address book that can be used for shipping information.

- Customer email.
- Date and time of this order.
- Merchant site name (used when the business has implemented a multiple site environment).
- Organization name, if there is one for this shopper.
- Price group used for this order.
- Email Order Details button.
  When an order is placed, the shopper automatically receives an email containing the order, shipping, and general payment details. If the email needs to be re-sent, click the Email Order Details button.
- Copy Order button.
  Use an already existing order and make a copy of it in cases where, for example, a shopper repeats the content of or has a few changes to a previous order.
- An order summary, including a sub-total and order total, any discounts, shipping charges, and tax.
  In the Order Summary section, presentation of tax information in the Agent Console varies based on the Pricing Includes Tax flag in the Oracle Commerce administration interface. When tax is configured as included (the checkbox is selected), there is no line item indicating tax. A note indicating the tax total is provided in the Order Summary and Order Items sections. Otherwise, when tax is not configured as included, a separate line for tax is present.

View order items

In the Order Items section, you can view information related to the specific items in an order, for example the SKUs and any promotions applied.

The Order Items section displays the following:

- The **Exchange Items** and **Return Items** buttons for orders with a status of Fulfilled.
- The **Add to Purchase List** button, available for all order states except incomplete and orders being amended.
- A list of the items associated with this order, including product image and number, SKU, item status, any price adjustments, quantity purchased and returned, and the subtotal. Click the product image to view the product details page for the item.
  Note: The status for a fulfilled order is Added to Order.
- A promotional summary listing the specific discounts applied to this order.
- A list of the order subtotal, discount total, shipping costs, tax, and order total.

In the **Order Summary** section, presentation of tax information in the Agent Console varies based on the Pricing Includes Tax flag in the Oracle Commerce administration interface. When tax is configured as included (the checkbox is selected), there is no line item indicating tax. A note indicating the tax total is provided in the Order

Summary and Order Items sections. Otherwise, when tax is not configured as included, a separate line for tax is present.

View shipping details

In the Shipping Details section, you can view information about the address the order was shipped to and the shipping method.

The section displays the following:

- The address or addresses to which the order was shipped.
- The shipping method.
- A Track Shipments button.

View tracking references

The items in an order may be shipped in one or more packages. You may need to track these shipments to determine their status. The Agent Console provides information on each of the packages associated with an order.

To track the packages:

1. Search for the order.
2. Under **Shipping Details**, click **Track Shipments**. The **Tracking Shipments** dialog appears, showing the name of the carrier or carriers and a tracking reference link for each package.
3. Click the link for the package you are investigating.

The link directs you to the carrier site where the information on the chosen package is displayed. The number of lines in the Tracking Shipment dialog corresponds to the number of packages associated with the order.

View payment details

In the Payment Details section, you can view the following:

- Customer billing address.
- Credit card details.
- Gift card details.
- Invoice details.
- Loyalty payment details.

This feature may not be enabled in your environment.

View notes

This section provides a table containing any notes associated with this order, including the date and time the note was created and the contents of the note.

Notes are for internal use only.

View custom order properties

This section provides additional internal and/or customer-facing properties that have been set up by your business to collect information on the order.

Once the order is placed, these properties are read-only. This information is collected during order placing and amendment.

5

Understand Accounts

Your store may have been configured to support account-based commerce transactions. A company that does business with you has one or more accounts containing one or more buyers. Each account is associated with a contracted list of catalog items, a price group, and a price list. Accounts may also include sub-accounts.

When you change from one account to another, the system automatically loads these details, so you may notice differences between accounts.

Only a single incomplete cart can be associated with an account. You can select the incomplete cart and edit it before processing instead of creating a new order.

A customer profile can be associated with one or more accounts depending on the business practices of that company. For example, June, who works for the ABC Company, may be a buyer with privileges to purchase products for the US and European branches. June's buyer profile would list both the US and European accounts and show that she is an active buyer on both.

To set up buyers on an account, you must be an administrator for that account. If configured for your site, requests can be made by the shopper to be added to an account. While approval for such requests is not available to agents, you can view the requests that have been made by using the search functionality on the Registration Request tab in the Customer Search section. For more information, see Search for Customers.

Understand account buyers

You can search for a buyer as you would any other shopper contact, or you can search for the specific account itself.

If an account buyer you select is associated with multiple accounts, you can see the associated accounts in the **Customer Results** table by hovering over the Account field for the buyer. A list of accounts appears.

On the **Customer Search** page, use the **Account** field to search for the account name. The **Customer Results** table displays all buyers associated with the account. You can then select the buyer by clicking the link in the Profile column.

Note: When using the typeahead feature to search for account names, only 10 accounts are displayed in the drop-down. To display more than 10 accounts, provide more detailed criteria for the search.

- In the **Cart** column, the **Select Cart** link is displayed if there is more than one active account or active site for this buyer. Clicking **Select Cart** displays a dialog from which you can select the account and site for which the order is to be placed. Clicking the **New** link opens a fresh order.
- If you want to change the account from the **Create Order** page, at the top of the page use the Site and Account menus. All sites and accounts associated with this buyer are provided in the menu lists.

- On the **Order History** page, all past orders are displayed. This list can be filtered based on account, site, order, and/or order state. Note that the **Order History** page does not display orders for inactive contacts.
- On a dedicated account admin profile page, on the **Account Contacts** tab, the manage contacts table shows all buyers associated with an account. Change the account name from the menu to view buyers for those accounts.

To display this tab, the buyer must be delegated as an account admin.

- A dedicated account admin can remove a buyer from an account or make him or her inactive. A buyer who is inactive on one account may still be active on other associated business account. Inclusion on other accounts remains intact even when the buyer is removed from one account.
- The **Require Approval** checkbox and the **Set Purchase Limit** field values are applicable for the account selected on the **Customer Profile** details page. Each account can have its own values for these settings.

Note: If you have a large list of organizations and your environment is configured use typeahead searches, your search may return a limited number of results. Use more letters within your search to fine tune your search results. For example, searching for "org" may return a limited amount of results. Searching for "org - ExampleCompanyLtd" would return more specific results.

Understand account addresses

Delegated administrators, account address managers, and profile address managers can edit, add, and delete addresses for an account.

Only administrators and account address managers can change an address to the default billing or shipping address.

The address book contains the following expandable sections:

- Default shipping and billing addresses.
- Other addresses associated with the account.
- Profile addresses.
- Addresses inherited from the parent account.

Note: You must enter the account billing and shipping addresses separately when creating or editing an order.

Understand external catalog and price group assignment

If your business provides external catalogs to your account buyers, additional information is needed to select the correct catalog and price list group from this external system.

To use the external catalog and price group assignment:

1. Complete a customer search and either start a new order (+New) or select an order already in progress.
   In a multiple site environment, the properties can be updated in the **Select Cart** dialog.

2. At the top of the **Create Order** page are site picker menus with options for choosing the account and site for which the buyer has access.
3. If the external catalog and price group system is available, an **Additional Custom Factors** link is displayed. If the buyer requires access to the external system, click the link to display the dialog.
4. In the **Additional Custom Factors** dialog, enter the appropriate information for the following properties:
    - `additionalPlg` indicates additional price group IDs; this field is optional. These are dynamic properties with `externalShopperContext`.
    - `plg` is the primary, default price group ID; required.
    - `catalog` is the valid catalog ID; required.
    - `Msg` is an informational message associated with access to the catalog and price group; required.
    - `responseCode` that determines the webhook to generate the response; required.

If the `responseCode` is set to 1, then the details (`additionalPlg`, `plg`, and `catalog`) are used and the external system is set. For `responseCode!=1`, the external system is not changed and continues with the default/ previously set catalog. Once the external system is set, the buyer can access the new catalog and price groups.

Orders placed with external system are editable and exchangeable.

Since the account-based commerce system does not support additional price groups, orders cannot be placed with additional price groups for an account contact.

For more information, see Assign Catalogs and Price Groups to Shoppers in *Extending Oracle Commerce*.

Understand registered shopper conversion

Oracle Commerce provides the functionality to convert a registered shopper to an account-based shopper.

Once converted, the new account-based shopper profile behaves as any other account-based shopper would for the agent, with the following exceptions:

- The shopper, once converted, is unable to view any orders, including open orders. You as the agent are also unable to view them.
- Fulfilled orders remain attached to profile. Agents can see these orders, cancel them, and process returns, but not exchanges.

For more information, see Convert registered shoppers to account-based shoppers in *Extending Oracle Commerce*.

6
Manage Orders

Order management features in the Agent Console allow your agents to place orders, amend and cancel orders, view order history, process returns, and initiate exchanges.

Note: The Price column may contain rounded values. However, during all pricing calculations, the actual price value is used.

Place orders

As an agent, you can perform a range of tasks to assist customer in placing orders.

Tasks you can perform are as follows:

- Create a new order.
- Shop on behalf of the shopper; this feature involves actually launching the storefront application from within the Agent Console for the selected shopper, including all storefront customization associated with the individual shopper.
- Complete an order already in progress.
- Amend an already-placed order.
- Cancel a placed order if it is within the remorse period.

Note: The use of bookmarks is not available to access sections on the Create Order page.

If configured by your administrator, custom properties may be available for the payment group when placing or editing an order. The custom property field or fields appear on the Create Order page in the Payment Details section.

Custom payment properties are configured using the External Payment Property Metadata Retrieval webhook. For more information, see Understand webhooks and Send custom properties to a payment gateway in *Extending Oracle Commerce*.

Create new orders

In cases where the customer does not have an order in progress, you can create a new order.

Be aware that the workflow for creating a new order is influenced by your environment: single-site or multiple sites.

Complete these steps to create a new order:

1. Complete a customer search. If the customer is new, click the **Create New Customer** link to create a new customer record.
2. Under **Customer Results**, in the Cart column, click **New**. The **Order Details** page appears.
   In a multiple site environment, the **Select Cart** link is displayed. Click the link to display a menu from which you can choose to create a new order or select an incomplete order on a particular site.

If you have a single-site environment, the **New** or **Complete** link is displayed. If the customer has an order in progress, the **Cart Action** is displayed as Complete.

3. In a multiple site environment, when you are creating a new order or updating an incomplete order, a site picker is displayed on the Create Order page. From the menu, you can choose the site to which the order applies.
The site picker is not displayed for a single-site environment. To use split shipping, select the site in the single shipping mode, and then switch to the split shipping mode when finished.

4. Add items to the cart using SKU, Product Name, and Catalog searches. Once you add an item to the order, in the **Order Items** section, under the order total, the **Schedule Order** checkbox appears. For more information, see Schedule an order.
If you are working in an environment that uses split shipping, perform the actions needed to add items to the cart in single shipping mode, and then switch to split shipping mode.

   Note: A TypeAhead feature may be enabled for your site. TypeAhead is a separate, auto-suggest searchable field ranking list that can return results in a type-ahead format when you or a shopper type in a menu field. For more information, see Manage Search Settings in *Using Oracle Commerce*.

5. Once the item is added to the cart, to make any changes to the item variant or modify the add-on products, click the product image or name to display the product details page.
The system supports backorder and preorder items. Add these items as you would any other items. A notice is displayed on the Create Order page to show the backorder and preorder status along with the date by which these items are available. Once the order is placed, the notice changes to show the item has been backordered or preordered. For more information, see Manage inventory in *Using Oracle Commerce*.

   Some base items may also include add-on items (additional items purchased to enhance or modify the base item for sale, for example, warranties and monogramming). While they do possess their own SKU, description, and attributes, these add-on items cannot be purchased separately and are flagged as not for individual sale in the administration interface. Add-on items appear in the same product detail line as the base item. You can view details and edit add-ons before placing the order. Once the order has been placed, you can review the order details, included together in the product line. You cannot return an add-on item separate from the main product.

   Any regular item can also be an add-on product and can be purchased separately as well as an add-on item to any product as long as it is associated to the main product.

   When using the catalog to add items, click **Complete Order** next to the **Cart** icon to return to the **Order Details** page.

   You cannot search for items with deactivated SKUs.

   You may see items in the catalog that are listed as having no price associated with them (displayed as Price Unavailable). You cannot add such an item to a shopper's cart. Contact your supervisor for details.

   Note: Products cannot be added to cart from the **Checkout** page. Only shipping can be selected on this page. Return to the cart to add or remove items from the order.

6. To add multiple quantities of an item, you can do the following:

- Click **Update Quantity** to directly add a quantity in the shopping cart.
- Click **Add from Catalog**. From the catalog list, click the link to open the product details page. In the quantity field, enter the quantity number for that item, and click **Add to Order**.

7. For items with variants (for example, size and color), to see the in-stock status of different item variant combinations, click the item link to open the product details page. On the page, a table is displayed showing each SKU variant combination along with how many of each product is in stock.
   For example, for a camera whose variants are color (black, blue, gold, and brown) and resolution (14.1, 22.3, and 30.8 megapixels), the list would show how many black, 14.1 megapixel cameras are currently in stock, as well as all the other combinations offered.
8. If your business uses the Oracle CPQ Configurator integration, and the item you are adding has been identified as a configurable item, you can click the **Configure** button to open a frame that allows you to choose between the configurable options associated with the product. Once you have selected the configurable options, click the **Add to Cart** button to add the configured item to the shopper's order.
   There is a validation check before the order is processed to ensure that the configuration options selected are valid. If they are valid, the order process completes and the order is placed. If they are not valid, an error message is displayed telling you that the configuration is invalid and that the order cannot be placed.
9. If your business uses a system that externally prices items (for example, with shoppers bidding on items), items may already be included in the Order Items section. Notes are provided explaining the quantities covered by the external price. Anything beyond that quantity is priced at list price. For example, up to 5 items may be listed at the external price of $90.00, with each additional item at the list price of $100.00.
   You cannot add an item and give it an external price. However, if the item appears in the shopper's order with an external price, you can change the quantity to reflect the shopper's request.

   Important: When dealing with externally-priced items included in the Order Items section, use extreme caution with the Delete Product button located to the right of the quantity field. If you delete a product from the order using external pricing, there is no way to recover the product.
10. For BOGO (Buy One, Get One) promotions—which may include other similar purchasing combinations, for example, buy two, get one free—the free items are not added to the order automatically, but must be added manually by either the shopper or the agent. Refer to your business promotion information to determine which promotions are currently valid for BOGO.
11. If your storefront uses multiple price groups, select the appropriate price group from the menu. Multiple price groups can have same currency.
    Price groups can only be selected during order placing. If your environment is set up for account-based commerce functionality, customers logged in see only the prices of the price list group associated with their contract. The menu option for selecting other price list groups is not displayed.

    When setting up a shipping method, you can specify an associated price group or region. Depending on the configuration, the shipping method may change based on the shipping region and price group of the order. For example, selecting a US shipping address and price group might provide a free shipping method while other addresses and price groups may not. Select the correct address and price group combination to ensure the order totals are correct for the shopper.

    To set up a split shipping method, configure the order with single shipping and then switch to split shipping when finished.

For account-based commerce buyers, the price group is associated with the contract and addresses are associated with the account. Therefore, you cannot edit either of these. This feature may not be enabled in your environment.

12. A line item called **Shipping Surcharge** may appear under **Order Items** in the **Order Total** section. This extra charge is automatically added to the shipping of some items depending on criteria set by your business, for example, for an oversized item such as a kayak. Shipping surcharges are set up in the Oracle Commerce administration interface.
If an item with a shipping surcharge is returned, the corresponding shipping surcharge is deducted from the Shipping value under Refund.

    There are discounts that may appear under Order Items. These discounts are set up in the Oracle Commerce administration interface and appear as a separate line item. If there are any discounts set up, and the order data satisfies the criteria, the discounts are automatically applied.

13. Change the quantity of the items to be purchased by highlighting the number in the Quantity field and edit the number. Once you have edited the quantity, click **Update**.

14. Delete items from the cart by clicking the Delete icon to the right of the Quantity field.
Clicking the Delete icon immediately removes the item from the cart.

15. If the system has been set up for agent supervisors to complete item price overrides during order creation, a pencil icon appears to the right of the item.

    - Click the pencil icon to display the **Price Override** dialog.
    - Enter the corrected price in the Subtotal field.
    - Select a reason from the menu.
    - Click **Save**. The new price appears below the original price.

    Once the order is placed, an information icon on the Order Details page provides details about the person who made the change as well as the reason.

    Note that you can not perform a price override in a split shipping environment. Perform the override in the standard shipping mode and then switch to the split shipping mode when finished.

16. Enter promotion codes, shipping details, payment details, and any notes pertaining to this order.
Coupons can be set up for use on an unlimited, limited, or one-time basis, and a single coupon code can apply multiple promotions to the order. Coupon codes are entered under the Promotion Summary section when creating an order. If a one-time or limited-use coupon is used beyond its limits, a notice appears explaining the code has already been used. For more information, see Understand coupons and gifts.

    If your storefront uses multiple currencies, promotions can be associated with a price group as well. The details are shown in the Promotion Summary section, except when the applied coupon is not configured for the selected price group. Messages are shown only when the applied coupon is valid for the order.

    If the order results in a zero order total, see Understand zero payment orders.

17. For information on shipping items, see Understand shipping.
For account-based commerce buyers, you cannot edit the shipping or billing address information as it is set for the account associated with the shopper. If you click the Edit button and modify the displayed address, the changes are saved as

a new address. Note that the new address will not be displayed on the customer's profile page until the new order has been placed.

If no shipping method has been specified or if the shipping address is incomplete, an icon appears next to the order values to inform you the values are inaccurate. Once the correct shipping details are added, the order values are re-calculated and the icon is removed.

Note: The shipping address cannot be edited on the **Create Order** page, Address Book. To update an address, go to the Customer Profile page, Address Book tab.

18. For orders sent to a single address, Under **Payment Details**, if the billing address is the same as the shipping address, select the checkbox. No other details are necessary. If the billing address is different, clear the checkbox, and manually enter the shopper's name, country, address, and contact phone number.
Note: The shipping address cannot be edited on the **Create Order page > Address Book**. To update an address, go to the **Customer Profile** page > **Address Book** tab.

    For orders sent to multiple addresses, use the **Billing Address** menu to select the appropriate address.

    For more information, see Understand shipping.

    For account-based commerce buyers, you can only choose from the shipping and billing addresses listed as they are set for the account associated with the shopper.

19. For order payment, under **Payment Details**, switch to either single or multiple payments depending on the shopper's preference. For more information, see Understand single and multiple payments.

20. Click **Credit/Debit Card** option, or, if enabled, the Gift Card, Cash (also known as Deferred Payment) or store credit option. Your business may also allow the use of invoices (purchase orders) as a payment option. If the shopper has selected multiple payments, add additional credit/debit cards.
Note: Gift cards only appear as an option if they are enabled as a payment option in Oracle Commerce administration interface.

    - For the credit/debit card option, on the right side of the **Payment Details** section, enter the card information in the appropriate fields.

    Only one credit/debit card can be used per order unless the shopper has chosen to use the multiple payments option. For more information, see Understand single and multiple payments.

    - For the gift card option, on the left side of the **Payment Details** section, enter the gift card number and pin in the **Gift Card Payment** fields.

    You can use multiple gift cards per order even in the single payment mode.

    - For the invoice option, enter the invoice/purchase order number in the field. This feature may not be enabled in your environment.
    - For the store credit option, the shopper's store credit balance and store credit number field are displayed. This option is not available for account-based commerce accounts.

    While the system does allow for PayPal to be used as a payment option, because PayPal requires a user ID and password, agents are not able to use PayPal as an option for order payment when helping a shopper. For more information, see Configure the PayPal integration in *Using Oracle Commerce*.

Credit card and gift card information is not stored in the system. Only tokenized transaction references given by the payment gateways are stored. Card information cannot be edited once the order has been placed.

Only one credit card can be used for payment during payment processing. Because of this, the order may eventually end up associated with multiple credit cards if there are multiple amendments to the order.

The Cash (Deferred Payment) option only appears if your system is set up to accept cash payments. The Cash option can be configured to appear for specific countries. For valid shipping countries, once the order is successfully placed, it is sent to the fulfillment system after any applicable remorse period.

Cash orders can only be amended and any increase in the total cost of the order must be paid for using cash.

21. If necessary, click **New Note** to add a customer note. This may include, for example, special delivery instructions.
22. Once all item and customer information has been added to the order, click **Place Order**.
    At this point, the order can either immediately be submitted for fulfillment or, if your business uses a remorse / order hold period, placed on hold in cases where the customer wants to make changes to the order before it is sent on to be fulfilled.
23. For orders placed by account-based commerce buyers, if the order total exceeds the buyer's account level purchase limit, an **Approval Required** dialog appears. The buyer should be informed of the need for approval.
    Agents cannot approve orders that require approval. This action must be done by approvers with an active status on the account. For more information, see Understand Accounts.

Once you click **Place Order**, the order data is immediately saved or persists in the system to avoid situations where order data is lost when a third-party application (for example, a credit card system) is for some reason unavailable and the order fails. This functionality eliminates the need to re-do a failed order.

Copy an order

Use an existing order and make a copy of it in cases where, for example, a shopper repeats a previous order or has only a few changes to it. Any order, except incomplete orders, can be copied no matter the status. However, orders placed by anonymous users cannot be copied.

To copy an order:

1. Complete a customer or order search to find the appropriate order to be copied.

   The Copy Order button is displayed for every order in the Order details section.
2. Click the button.

   A new incomplete Create Order page is created that includes the items and quantities from the previous order except for the payment details information, which is not saved in the system. If an incomplete order for the shopper is present, a dialog is displayed advising the shopper that the products in the current cart will be merged with the incomplete cart if he/she proceeds.
3. Make any necessary changes to the order, and click **Place Order**.

Understand single and multiple payments

Shoppers can make a single payment with one credit/debit card or multiple payments with one or more payment options to satisfy the balance for an order. Account-based commerce buyers can select the payment methods for a specific account and site.

In single payment mode, the shopper can use a single credit or debit card and multiple gift cards and has a single invoice. When single (full) payment is required, and any part of the payment is rejected, the entire order is rejected and must be completed by the shopper with a new payment method.

In multiple payment mode, the shopper can use a combination of credit and debit cards, gift cards, cash, and invoices. If any part of the payment is rejected, the order status is changed to pending payment, or it remains in an incomplete state depending on the payment options set in the administration interface.

For more information, see Understand split payments in *Extending Oracle Commerce*.

To use payment modes:

1. Create an order on the customer's behalf, add items to the cart, and select the shipping method.
2. Under **Payment Details**, click the **Multiple Payment Options** link or **Single Payment Option**, depending on which mode the order is currently in. By default, single payment mode is displayed and multiple payments mode is enabled.
   During order creation, you can switch from one payment mode to another; however, payment information is cleared as the default version of the Agent Console does not retain payment information.
3. For single payment, enter information for a single credit/debit card, one or more gift cards, or cash. If enabled, you can include a single invoice. Complete any additional activities for the order, and click **Place Order**.
4. For multiple payments, do the following:
   - Click the **Multiple Payment Options** link. The **Payment Amount** field is displayed.
   - Click the **Payment Option**: credit/debit card, gift card, cash, or invoice and store credit, if configured for your site.
   - In the **Payment Amount** field, enter the amount you want to apply using the payment option chosen.
   - Once the information is entered, click **Add Payment**. A line item is added above **Payment Options** to identify each payment made (for example, Payment 1 and Payment 2).
   - Billing addresses can be edited for each payment to allow for a different billing address for each credit card or invoice. Click **Update Payment** to attach a different billing address to the payment.
   - When using a gift card, click the **Pay With Gift Card** checkbox. Enter the gift card number in the field, and click **Add Payment**. The line item appears along with the list of other payments made to cover this order.
   - Make changes to the remaining payment amount and use payment options until the balance is zero.
     The system does not allow for overpayment. In this case, an error message is displayed next to the Payment Amount field.

- Edit the amounts and payment types by using the Edit and Remove links located next to each payment. Click Edit to make amount changes or Remove to delete the payment. The Payment Amount field is updated to reflect the change or removal of the payment.
- Make any final changes to the order items: adding, changing amounts, and removing.
- Click **Place Order**. Notification is displayed to show the order has been placed successfully. On the completed order page, under **Payment Details**, each payment is listed along with associated billing address and any specifics about the credit or debit cards, gift cards, invoices, or cash used along with the amount for each payment. The status of the payment (for example, The Authorization is Successful) is also displayed.

Understand invalid payments

You may encounter invalid payments (for example, when the order is placed, if the administrator has set any limitations, like a cash limit of $99 or a rejected credit card) in either single or multiple payment mode.

Note: The pending payment state is only available when partial payments are enabled in the administration interface and a payment failure occurs. In the case of account-based commerce orders, an order can move to the pending payment state during order approval flows despite this setting.

For single payment mode, the entire order is rejected, and a valid payment must be made to move the order to fulfillment.

In this case, the system displays a payment error notification, and the order is moved to the Pending Payment state. The state is displayed at the top of the order page next to the order number (for example, Order Number 12345 - Pending payment). At this point, the order has not been authorized.

In pending payment state, you can remove any of the payments by clicking the Void Payment link next to the payment. The Payment Amount field reflects any changes. Additional cards can be added by entering the information and clicking Update Payment.

For any orders moved to the pending payment state, an email is sent to the shopper informing him of the need to edit the order so it can be successfully fulfilled. The shopper can go back to his order or have the agent help him complete the payment.

Once all the changes have been made, click Place Order. If successful, the order is moved to the fulfillment state.

Understand zero-cost orders

There may be cases where a shopper places an order that requires no payment. This situation could occur because of, for example, an introductory promotion. The system is able to process these orders. In addition, your business may choose to collect information on these orders.

For information on setting up your business to accept zero-cost orders, see Support zero-cost orders in *Extending Oracle Commerce*.

If the order results in a zero order total, the **Payment Details** section does not display the typical fields required when collecting payment option information. Instead, the Payment Due line shows the zero balance.

Click the **Enable Payment Details** link to display the fields for the information your business wants to collect. The type of information varies and can be set up in the administration interface.

Click the **Disable Payment Details** link if no additional information collection is required.

Understand coupons and gifts

Your store may be configured to use a range of default and custom promotions, including discount coupons and gifts with purchase, which affect the totals for orders being placed.

- Single use and limited use coupon codes: your business may provide shoppers with coupon codes that can be used once or can be used multiple times, either during a specified period of time or for a limited number of purchases.
- Multiple promotion coupon codes: a single code can be tied to more than one discount. For example, when a shopper spends a certain amount of money, he or she receives a percentage off an item or order and receives free shipping.
- Coupon stacking rules: each stacking rule includes one or more promotions that are treated as a group. Rules are intended to prevent simple combination strategies, for example to keep customers from combining buy-one-get-one (BOGO) offers with gift with purchase (GWP) offers.
  Stacking rules are created in the administration interface. For more information, see Manage promotions with stacking rules.
- Gift with purchase: automatic promotions can be tied to individual products or order amounts. When a specific product is added to the order or an order amount is reached, the shopper automatically receives a gift or a choice of gifts.
  For example, when a shopper adds a gaming console, a free game controller could appear in the cart.

  A gift may also allow a shopper to select from a menu of choices, for example, choosing one of several gift cards for different venues. If this is the case, a **Select the Gift** button appears in the item list. You can select the gift from the menu for the shopper and click **Add to Order**.

  If the main product is removed from the order, the gift product is removed or re-priced automatically based on the promotion settings.

The Promotion Summary lists the details of the discounts and gifts applied to the order. The summary is updated as additional coupons, discounts, and gifts are added to the cart.

Agents should consult with their internal business resources to determine the current coupon codes, promotions, and valid code rules that may affect order totals.

Understand loyalty programs

Your business may use a loyalty program where shoppers can earn points and then redeem them by choosing items from either a separate loyalty program catalog or an integrated catalog that includes both standard and loyalty items.

For sites that use loyalty programs, the cart must either include only items that are paid for in loyalty points or are paid for using standard currency methods (such as credit/debit cards).

There cannot be a mixture of the two types in a single cart. You may also choose to have shoppers pay for items in loyalty points while having tax and shipping charges completed using standard monetary currency methods (such as credit/debit cards or gift cards). These choices are made using the Commerce Admin API. Check with your internal support resources to discover which methods for payment are being used in regards to your loyalty program.

The system maintains information on loyalty point totals for each shopper. The details on a specific shopper's points can be found on the shopper's customer profile page in the Agent Console.

A shopper creates an order by choosing items from the loyalty catalog just as she would from the regular site. The shopper's total for the loyalty order is deducted from their loyalty point total.

With returns, both the loyalty points and any taxes and shipping charges are refunded to the shopper, depending on how the transaction was completed.

For more details, see Work with Loyalty Programs in *Extending Oracle Commerce*. For information on setting up store credit, see Integrate with a Store Credit Payment Gateway in *Extending Oracle Commerce*.

Understand shipping

With Oracle Commerce, you can ship items in a single order to either a single shipping address, using a single shipping method, or to multiple shipping addresses using multiple shipping methods. If a shopper purchases more than one item, they can have the items shipped to separate addresses.

In a multiple site environment, the shipping methods are site-specific, that is, each site can have its own shipping methods. The shipping methods are loaded based on the site selected. The default shipping country also is loaded based on the site selected during order creation.

Keep in mind the following concepts when working with multiple shipping in the Agent Console:

- When you create a multiple shipping address order where the number of shipping groups is more than one, the cash payment method is disabled.
- You cannot change the price list group or site on the multiple shipping page. You must make necessary changes to the cart before navigating to multiple shipping cart. You also cannot add products to the cart by SKU, product name, or catalog. If you want to modify the cart, site, or account, click Ship to Single Address and modify the cart products, for example by adding a new item to the cart, or by changing the site or account.
- Shipping surcharges and discounted prices are not displayed on multiple shipping pages after you apply a promotion to the product.
- Once you open the multiple shipping page and add the address and shipping methods, if you return to the single shipping address page, payment details are not displayed. You must return to the multiple shipping page to complete the payment.
- Once you have opened the multiple shipping cart page and selected all the address and shipping methods, the item is repriced. At this point, if you click Ship to Single Address, the payment options are not available. You can only complete payment in the multiple shipping cart.

- When you copy an order originally placed using the multiple shipping cart method, the new copied order is displayed as a single shipping page only.
- For account-based commerce, a buyer can only view the shipping methods specific to the account and site.

To use order shipping for single or multiple shipping addresses:

1. On the **Create Order** page, in the **Order Items** section, you can choose to ship to a single address or multiple shipping addresses.
   Clicking the **Ship to Multiple Addresses** link allows the shopper to ship items available in the order to multiple addresses using multiple shipping methods. Clicking the **Ship To Single Address** link returns the order the single shipping address page but retains the multiple shipping addresses if any have been selected.

   The columns displayed on this page for orders shipped to a single address are Product, Price, Quantity, and Subtotal. The columns displayed for orders shipped to multiple addresses are Product, Price, Quantity, Ship To, Shipping Method, and Subtotal.
2. Add items to the order as necessary.
3. For orders shipped to a single address, in the **Shipping Details** section, select the address from the address book and the shipping method from the menu.
   The chosen address determines the shipping method available in the menu.
4. For orders shipped to multiple addresses, do the following:
   - Use the **Ship To** menu to select the address for the item or items. The menu lists addresses and address nicknames (to abbreviate the address information) currently available in the address book for this shopper. Addresses are listed with first and last name, street/post office box, and city/state. If the address is not listed, click the Add a New Address choice from the menu.
     Note: Adding the address using the **Ship To** menu adds the address within the order but does not add it to the address book. Click the **Address Book** button located in the **Shipping Details** section to add the new address to the address book, or add it on the **Address Book** tab of the **Customer Profile** page.
   - Click the **Change Address** link and select any address provided in the **Address Book** dialog. You can view complete address details in the dialog.
   - If there are no addresses available, add a new address and select **Use This As My Billing Address** if necessary as provided in **Address Book** dialog.
   - Depending upon the shipping address chosen from the menu, the **Shipping Method** menu lists the available methods.
   - With multiple quantities of a single item (for example, 3 @ $24.99), shoppers can also send items to multiple addresses.
     After selecting the address, the shipping method menu is enabled, and you can select any shipping method available; if a site is selected, the site-related methods are loaded. When the item quantity added to cart is greater than 1, a Dispatch to Additional Addresses link is displayed. When the link is clicked, the specified item quantity is decreased by one (1). You can then ship each item to separate addresses with separate shipping methods. If you delete a split item or change the quantity to zero and update it, the remaining item quantity does not increase. You must change the quantity manually.

     Add the quantity to ship, the shipping address, and shipping method to this new row. Click Update to make changes to the row.

     Note: The original quantity does not automatically change to reflect the quantity sent to the additional shipping address. For example, if the shopper placed a total quantity

of 3 glasses in her cart, and wanted 1 glass shipped to another address, entering 1 in the new Quantity row does not decrease the original amount to 2. Manually change the original quantity to 2 to avoid charging the shopper for 4 glasses.

- In the **Shipping Details** section, the addresses are displayed associated with the order, along with the shipping choices and the prices associated with each shipment: method, subtotal, shipping costs, tax, and order total for each shipment.
- In the **Payment Details** section, the **Billing Address** menu shows a list of addresses the shopper can choose from. The fields under the menu display the associated information. You can update these fields if necessary.
  Note: The updated billing information is not saved when changes are made to the **Billing Address** fields. If the shopper wants the address updated or if the shopper wants to use a new billing address for future orders, you must make the changes in the address book.

When editing an order using a multiple shipping cart:

1. Click **Edit Order** at the top of the **Order** page. (Note that this option may not be enabled, for example if the remorse period is over.) The multiple shipping cart page is displayed. If you want to modify the cart, site, or account, you must click **Ship To Single Address** and then complete the changes.
2. Once you complete the changes, if any pending amount due is displayed, provide payment details on the multiple shipping cart page and complete the order.

Understand shopping on behalf of the shopper

From a selected shopper's profile page, clicking the Launch Store as Customer button allows you to open the storefront application on behalf of the shopper. This action provides the same experience as if the shopper had logged into the storefront.

Note: This action is different from creating a new order or completing an order from the Agent Console where the catalog information is provided entirely from the Agent Console application and not directly from the storefront.

Once you launch the storefront, you can participate in all store-specific processes, such as browsing the catalog and creating an order as the shopper. The view is shopper-specific and customized; the pages change according to the selected shopper profile and access (for example, the number of categories the shopper sees as well as the associated pricing and items available).

At the top of the page, an icon appears showing that you are shopping for a specific shopper. Click the icon to log out of the storefront application and return to the Agent Console.

The agent's ID is associated with any order completed in this way. Once the order is placed and the order is accessed from the Order History page, a line item in order details lists the agent responsible for placing the order in the name of the shopper.

For information on viewing storefront content, see Customize your store layouts.

Shop on behalf of the shopper when carts are shared across multiple sites

If your environment has been configured with multiple sites, the environment can also be configured to allow shoppers to add items to their cart from multiple sites. This

functionality is enabled by the implementation of cart sharing groups. For information on this, refer to Create Carts Shared Between Multiple Sites.

Create multiple site orders with shared carts

For an Agent to assist a shopper who is creating an order from multiple sites, you must shop on behalf of the shopper. However, note the following limitations:

- Any items you add from different sites that belong to the same site group, but are not part of the current site's catalog, will be removed from the cart.
- All of the items in the cart will be associated with the current site.
- If the same SKU is added from different sites associated with the same site group, all of the items will be merged into one item, and the item will be associated with the current site. For example, if you add a t-shirt from SiteB and a t-shirt from SiteA, instead of seeing two separate line items, you will see a single line item that has a quantity of 2.
- Inventory validation and stock reservation happens based upon the location of the site associated with the order.

Edit multiple site order with shared carts

You can not edit a cart order unless all of the sites in the site group are associated with the same catalog and inventory. Therefore, it is not recommended that you edit orders that were created with carts shared across multiple sites unless you shop on behalf of the shopper. The following limitations apply:

- Items that do not belong to the current site's catalog will be removed from the cart when you edit the order.
- If your environment is using Open Storefront (OSF), editing an order shared across multiple sites can result in incorrect inventory deductions and is therefore not recommended.

Work with exchange or return orders that share carts across multiple sites

An Agent can return any order that was created from multiple sites. When you work with an order that contains an exchange, it is recommend that you create a return request and then create a new order. If the items' sites use the same catalog, the inventory will be added back correctly, however, new order inventory validations and stock reservations are based upon the location of the site associated with the order.

Understand the Oracle CPQ integration

Oracle Configure, Price and Quote (CPQ) is designed to provide functionality that improves response time to customer inquiries, improves customer interaction with your business, and allows greater customer flexibility in selecting complex products.

There are two integrations between Oracle CPQ and Oracle Commerce, both of which can be accessed through the Agent Console if they are set up for your environment:

- **CPQ Configurator**: Allows an agent to configure complex products for purchase in Commerce. If a product has been identified as a configurable item, there is a Configure button on the Product Detail page for the item. If you click the button, you can select between the customizable options that are associated with the product in question.
- **CPQ Quoting**: Allows you to request a quote on an order, thereby initiating an Oracle CPQ transaction that a sales specialist can modify, reconfigure, or discount. Once

finalized in Oracle CPQ, the quote is returned to Commerce where it can be accepted or rejected, or you can request a requote.

Understand Oracle CPQ product orders

In the Oracle CPQ environment, products consist of a number of sub-items a shopper can select, each with its own properties. For example, a laptop could have the sub-items processing speed, hard drive, and RAM capacity.

To add an Oracle CPQ product to an order:

1. Add the item to the order. A dialog appears with the name of item, price, quantity, along with other product information. Because there are sub-items associated with this product (for example, a laptop), these sub-items must be chosen separately. To do this, click the **Configure** button in the dialog.
2. The next dialog displays a list of the sub-items with default values selected, along with pricing that includes the currently selected sub-items. Scroll through the list and select the appropriate options. Click **Update** to show how the changes affect the price of the item.
3. Once the options have all been selected, click **Add to Cart**. The product appears in the **Order Items** section, product details column, of the **Create Order** page. On the **Product Details** page, a non-returnable flag is displayed for items that cannot be returned. Click the product image to view a product details page for the item.
4. Click the **+ Details** link to display the sub-items, showing the price and quantity added. Click the sub-item links to display a dialog with information and pricing.
5. Click the **+** to expand the tree menu and reveal additional information on sub-items included with the product.
6. Click **Re-configure** to return to the sub-item dialog list. When re-configuring the product, click **Update** whenever changes have been made, and click **Add to cart** to return to the **Create Order** page with the changes to the product.
7. When the order is ready, click **Place Order**.
   When the order is placed, the system completes a validation check. This check ensures that if there are any configured items in the cart, the configuration and price are both still valid.

The details associated with the product and sub-items are available for orders in other statuses, such as edit order, return orders, and exchange orders.

Request a quote for an order

If your business has the Oracle CPQ Quoting integration enabled, you can request a quote for an order from within the Agent Console.

To request a quote for an order:

1. Conduct a customer search.
2. Click **New** to create a new order or select an existing unfulfilled order for the customer.
3. Once you have an order with items in the cart, click the **Request Quote** link in the order edit page. You can switch between the **Request Quote** page and the **Create Order** page by clicking on the appropriate link.
4. Add text to the **Quote Notes** text field as desired and click the **Request Quote** button.

Once you have submitted the quote request, the Request Quote webhook is triggered and all relevant information is passed to Oracle CPQ for a decision on the quote. The order status is changed to Pending. When an order is in Pending status, you cannot perform any operations on the order.

A confirmation email is sent to the shopper informing them of the status of their order.

5. Once a response is received, the order status changes to **This Order is a Quote**. You then have a number of options on how to proceed:

   - **Accept the quote:** If the customer is satisfied with the quoted price returned from Oracle CPQ, you can accept the quote on their behalf by clicking the Accept button and proceeding with the order as normal.
     Once payment information has been entered and the order is placed, the order status changes to Submitted for Fulfillment. At this point, the Update Quote webhook is triggered and Oracle CPQ is informed that the quote has been accepted.

     At this stage, you can click the Edit Order button, but the only edits allowed to the quote are changes to the shipping group or the application of shipping discounts or promotions. You may not add or remove items from the cart or change the quantities of items included in the order. The order status changes to Order Being Amended until you click the Complete Order button.
   - **Request a requote:** If desired, you can enter more details in the **Request Quote** text box, and click the **Request Requote** button to request an updated quote. When you request a requote, the order status changes to **Pending Quote**. When an order is in **Pending** status, you cannot perform any operations on the order.
   - **Reject the quote:** You can click the **Reject Quote** button to reject the quote. This cancels the shopper's order and the order status changes to **This Quote has been Rejected**.

   The response to a quote request includes provision for an expiry date for the quote. If the quote has expired, the **Accept Quote** and **Reject Quote** buttons are disabled, but you can request a requote for the order.

   Once you respond to the quote, a confirmation email is sent to the shopper informing them of the status of their order.

Understand punchout orders

Punchout ordering is an account-based commerce flow in which buyers can purchase products directly from merchants; the buyer's procurement system is integrated with the merchant's commerce system.

While buyers browse in their own procurement system, they can "punch out" to the merchant's site where they can receive details about products and account-specific pricing. Buyers add products to their carts from these product catalogs. When a buyer is ready to check out, instead of going through the buyer's commerce account checkout process, the system sends the product information back to the buyer's procurement system so the buyer can either continue shopping at other merchants or submit the order through the procurement system for approval.

Once the order has been verified and approved, a purchase order is sent to the merchant. A single purchase order can constitute a collection of multiple buyers going as one or an individual. Depending on the system capabilities, the purchase order can go directly to a merchant's fulfillment system without the need for the Oracle Commerce system. In other cases (for example, the merchant's fulfillment system is not set up to receive purchase

orders), this may not occur. The Commerce system can then receive the purchase orders, validate them, and send them through the Commerce fulfillment process.

All the main flows (for example, editing or canceling an order, or returning or exchanging products) have been built into the Commerce system for orders.

Currently, only direct punchout orders are supported, which involve direct buying from one buyer's organization to a single merchant. Third-party networks, also known as marketplaces, are not supported.

Understand punchout orders in the Agent Console

An agent cannot place a punchout order for a buyer because of the way the orders are placed. However, for those punchout orders that enter the Commerce system, you can search for the order as you would any other order:

- When a buyer launches the Storefront application and adds items to the cart, the punchout order is created and associated with a pseudo-user. The order is incomplete prior to submission. On the **Order Details** page, the **Origin of Order** field is set to **Punchout**. This is the state you see if you search for and open an incomplete cart of the pseudo-user.
- With submission of the punchout order, the **Origin of Order** field is updated to `PurchaseOrder` and remains in that state unless you create an exchange order.
- If your business has set up a remorse period, you can edit punchout orders that fit the criteria.
- Once the punchout order is fulfilled, you can search for the order and complete returns and exchanges as you would with any other order.
- With exchanges, because a new order is created from the original order, the new order's **Origin of Order** state is updated to `contactCenter`. The original order's **Origin of Order** remains as `PurchaseOrder`.

Understand account-based commerce approvals

During order creation, the **Check for Approval** button is displayed so the agent is aware that the order may be sent for approval based on the account level purchase limit. If the order requires approval, the agent may not need to request payment details; only deferred payment is accepted, for example, an invoice.

Order approval and rejection are completed by an account approver, after order approval has been enabled in the administration interface, and orders have been submitted that require approval.

When creating an order for a buyer, once the items have been added to the cart, in the **Payment Details** section, on the right side of the page, click **Check for Approval** to see if the order requires an approval. A message is displayed to inform the buyer if approval is required. If the order requires approval, in the **Payment Details** section, credit/debit card information does not need to be added as the default version of the Agent Console does not retain credit/debit card information during the approval process. Only deferred payments (for example, cash and invoice) are accepted.

To approve or reject orders, the approver can log into his or her account and view a list of orders on the **Orders Pending Approval** tab. The approver can then view the details of an order and either click Reject or Approve. An email notification is sent to the shopper after an order has been approved or rejected.

Agents can assist approvers and account buyers by:

- Approving pending orders on behalf of an approver.
- Amending approved orders during the remorse period.

It is solely the approver's responsibility to approve or reject orders. However, an account buyer can contact an agent to amend an order.

To approve an order on behalf of an approver:

An example might involve an approver who is unable to access his or her Order Approvals tab because of lack of access to the internet.

1. Complete a customer search to confirm the shopper has the role of approver. On the Customer Profile page, refer to the Storefront Role section to identify if the shopper is a delegated approver.
2. On the approver's profile, under Account Name, select the account for the orders for approval.
3. Click the **Order Approvals** tab to view pending approvals.
   The **Order Approvals** tab is only displayed if the shopper has the approver role.
4. From the list of pending orders, click the order link for the appropriate order. The **Order** page is displayed with the status of Pending Approval.
5. Optionally, add information in the **Approval Comments** field.
6. Click **Reject** or **Approve**. On the confirmation dialog, click **Approve** or **Cancel**.
   Once approved, the status of the order changes to **Pending Payment** if a credit or debit card was used, or Submitted to Fulfillment if another method, like invoice, was used. The buyer receives an email informing him or her that the order has been approved.

   The order can also be set to Pending Payment in cases where no payment has been provided for an order sent for approval.

   The buyer can now access the order and add payment information if necessary. If a credit or debit card was used for the original order, in the default version of the Agent Console, the buyer needs to re-enter the credit card information. Once approved, orders created using the invoice payment method are submitted directly to fulfillment and do not require additional buyer intervention. In cases where credit or debit card are used, card data is not saved; the buyer must provide credit or debit card information.

   For orders with pending payments, your business can set a price hold period during which the price of the order will not change. After the period passes, the order is cancelled. For more information, see Set the price hold period.

To amend a buyer's order:

1. Conduct an order search and open the order to confirm the order is within the remorse period.
   Amendments to a buyer's order can only be made if the order is still within the remorse period.
2. On the **Order** page, click **Edit Order**. A dialog is displayed to inform the buyer that the order may require an approval.
3. Complete the order amendment process.
   The buyer can add an invoice number at this point instead of a credit or debit card. With an invoice, once the order is approved, the order goes immediately to Submitted to Fulfillment. Credit and debit card information is not retained by the system.
4. Ensure that the credit or debit card information is complete.
   If approval is not required, once the order is placed, the order is submitted for fulfillment.

5. Once the order is completed, click **Place Order**. If the order requires approval, a dialog is displayed to inform you of that fact. Click **OK** to proceed with the approval process.

Because a new order requires approval, the new order is generated and the previously placed order is cancelled. If a credit or debit card was used for the order, once the order is approved, the buyer must open the order and re-enter the credit card information as the default version of the Agent Console does not retain this information.

Schedule an order

Schedule automatically recurring orders for items the shopper needs to buy periodically. When the scheduled time is reached, the system automatically creates an order for the product. You can edit the schedule and the quantity later from Scheduled Orders page in the shopper account.

To schedule a recurring order:

1. Create a new order or open an order already in progress (complete the order) for the shopper.
2. Add an item or items to the order. In the Order Items section, under the order total, the **Schedule Order** checkbox appears.
3. Select the checkbox. The Schedule Instruction section appears.
4. In the Schedule Instruction section, enter the following information in the fields:
   - In the **Order Name** field, enter a significant name to help identify this particular scheduled order in the future.
   - In the **Start** field, enter the date on which this scheduled order begins. You can also use the **date picker** icon to select the date.

   This date must be later than the current date.
   - (Optional) In the **End** field, enter a date on which this scheduled order ends. You can also use the **date picker** icon to select the date.
   - From the **Frequency** menu, select how often this scheduled order occurs. Choices span from daily to quarterly.

   If the shopper decides to order weekly, an additional set of options appears. Choose the day or days of the week. You can also choose which week of the month the order runs.
   - Select the **Suspend** checkbox if the shopper wants the scheduled order, once scheduled, to be placed on hold.

   The total price of the scheduled order may change based on the prices of items in the order at the time it is submitted.
5. Complete the order per instructions in the Create New Orders section.
   Note: There is one exception with scheduled orders concerning payment: Scheduled orders must be paid for using an invoice or purchase order number as the default version of the Agent Console does not retain shopper credit card information.
6. For scheduled orders, the **Place Order** button is replaced with the **Schedule Order** button. Click the button to schedule the order.

7. Once the order is scheduled, the **Customer Profile** page for the shopper appears with the **Scheduled Orders** tab displayed. On this tab, you can review all of the scheduled orders created for this shopper.
8. On the tab, the table provides information about each order: name, order ID, created date, frequency, next ship date, status, and last order date. Click **Delete** to permanently stop the scheduled order from recurring.
   If in the future you want to find the scheduled orders for a shopper, open the shopper Customer Profile page and click the **Scheduled Orders** tab.
9. To view a scheduled order in more details, click the name link. The **Scheduled Order** page for that order appears.
10. On this page, you can make changes to the schedule instructions, changing name, dates and frequency, and suspend the order.
    Note: You cannot add or remove items from the order. One solution may be to delete the scheduled order and create a new one.
11. In the **Order History** section, view the orders that have been fulfilled previously, along with order outcomes and any failure reasons if they occur.

Understand the effect of order approvals on scheduled orders

For account-based scheduled order, the order requires approval if one instance of the scheduled order exceeds the purchase limit. When a scheduled order is approved, the approval applies to every instance of the order created based on the schedule. The approval persists even if the prices of the schedule order change. Conversely, if Oracle Commerce determines that a new scheduled order does not require approval, that determination persists even if the prices change in the schedule order and cause its total value to exceed the purchase limit at some point in the future.

The approval status of a scheduled order affects what the shopper can do with the order. When a scheduled order has been approved, the order is locked down and the only elements that can be edited are the schedule, the active or inactive setting, and the payment method. Note that a scheduled order does not go back for re-approval if the schedule is edited.

When a scheduled order has been rejected, none of the order's instances are allowed to proceed and the order cannot be edited. When a scheduled order is pending approval, its contents are repriced when the order's details are viewed. The shopper can modify the schedule and the active/inactive setting of a scheduled order that is pending approval.

Scheduled orders that already exist when the order approval feature is enabled are allowed to proceed without approval.

Complete in-process orders

There may be situations where the customer is having issues with placing an order, so the order is in progress but incomplete.

To assist with an in-process or incomplete customer order:

1. Conduct a customer search.
2. Under **Customer Results**, in the **Cart** column, click the **Complete** link. The **Order Details** page appears.
   If the customer does not have an order in progress, the **Cart Action** is displayed as **New**.
3. Add or delete items, add promotion codes, shipping and payment details, and add notes as necessary. At this point, the shopper can use either single or multiple payments options. For more information, see Understand single and multiple payments.

For more information on adding and editing order details, see Create new orders. For information on scheduling an order, see Schedule an order.

4. Once all item and customer information has been added to the order, click **Place Order**.

Once you click **Place Order**, the order data (except for payment information) is immediately saved or persists in the system to avoid situations where order data is lost when a third-party application (for example, a credit card system) is for some reason unsuccessful and the order fails. This functionality eliminates the need to rebuild the entire order from nothing. Because the order data persists, you can recover the order if necessary.

Use the address book

The address book is available to you when creating an order. On the **Order** page, Shipping Details, under Shipping Address, you can view the current default address. If you want to select a different shipping address for the order, he or she can click the Address Book button. The Address Book dialog is displayed showing the current list of available shipping addresses.

Note: You can only add dynamic properties when creating a new address and not from the address book. You cannot update existing addresses on **Checkout** page for registered shoppers.

On the **Address Book** page, you can:

- Click **Ship to this Address** to select an address for the order.
- Click **New Shipping Address** to create an additional address for this account. When creating a new address, the address can be set as the billing address. Go to the **Customer Profile** page to set an address as the default.

The address book is available to you when amending orders.

For account-based commerce buyers, you cannot edit the shipping or billing address information as it is set for the account associated with the shopper. If you click the Edit button and modify the displayed address, the changes are saved as a new address.

Account-based commerce buyers with the Account Address Manager, Profile Address Manager, or Administrator role can add new, edit, or delete addresses. In addition, the account address manager and administrator can set default addresses for the account. These addresses are then available for account buyers to use.

Understand the remorse period

A customer remorse period, for example an hour or twenty-four hours, can be set by your system administrator. It represents the time between when the order is placed and when the order is in Submitted to Fulfillment status, thus holding the order before it is packaged, shipped, and delivered.

Note: As a best practice, it is recommended that you keep the remorse period as short as possible to avoid delays in fulfilling orders. Order submission to the fulfillment system – webhook activity included – is suspended until the remorse period is concluded.

Before the order is in Fulfilled status, you can search for the order and cancel or amend the order. See Cancel an order or Amend a placed order. Once the period has passed, this option is no longer available.

See also Set the customer remorse period.

Amend a placed order

If an order is still within the remorse period configured for your business, the order details can be amended.

Note: Be aware of cases in which the item was initially available at the time of placing an order, but later, before order amendment, the item became unavailable (for example, because the SKU was deactivated). In these cases, the item is removed from the shopper's cart.

For more information, see Create and work with SKUs in *Using Oracle Commerce*.

Complete these steps to amend an order:

1. Complete an order search and select the appropriate order. The **Order** status is **Submitted to Fulfillment**. The **Order Details** page appears.
   If the **Cancel Order** and **Edit Order** buttons appear at the top of the page, the order is still within the remorse period. If the order is outside the remorse period, the customer may want to start a return or exchange.
2. Click **Edit Order**. The order, shipping, and payment details are now enabled.
   - If the order includes add-on items (additional items purchased to enhance or modify a base item for sale, for example, warranties and monogramming), deleting the base item also deletes the associated add-on.
   - When paying for the added charges of the amended order, you can accept a gift card, the same credit/debit card from the original order, or a different credit/debit card, or an invoice (purchase order) if the original order was paid for using an invoice. The invoice payment option feature may not be enabled in your environment. Credit/debit card details must be collected from the shopper and entered manually even if the same credit/debit card is used for payment.

     In the single payment mode, only a single credit/debit card can be used for payment processing, while multiple gift cards can be used. The shopper can choose to switch to multiple payments mode if desired. For more information, see Understand single and multiple payments.

     Once a credit card is used, the invoice payment option is no longer available. If during order amendment, the order amount changes and the shopper wants to use a credit card instead of the original invoice, the credit card must be used for the full amount in single payment mode. If the shopper wants to combine credit cards with invoices, switch to multiple payments mode. The invoice payment option feature may not be enabled in your environment.
   - An agent cannot use PayPal as a payment option for added charges, because PayPal requires a shopper's user ID and password.
3. If your business uses a system that externally prices items (for example, with shoppers bidding on items), make changes to the quantity as you would with any other included item. However, shoppers may also include items beyond the number of externally-priced items. Notes are provided explaining the quantities covered by the external price. Anything beyond that quantity is priced at list price. For example, up to five items may be listed at the external price of $90.00, with each additional item listed at the list price of $100.00.

You cannot add an item and give it an external price. However, if the item appears in the shopper's order with an external price, you can change the quantity to reflect the shopper's request.

Important: When dealing with externally-priced items included in the Order Items section, use extreme caution with the Delete Product button located to the right of the quantity field. If you delete a product from the order using external pricing, there is no way to recover the product.

4. If the environment has been set up for agent supervisors to complete item price overrides during order amending, a pencil icon appears to the right of the item.
   - Click the **pencil** icon to display the **Price Override** dialog.
   - Enter the corrected price in the Subtotal field.
   - Select a reason from the menu.
   - Click **Save**. The original price is lined out and the new price appears below the original price.

   Once the order is placed, an information icon appears on the order details page and provides details about the person who made the change as well as the reason.
5. Once you complete your amendments, click Place Order. The amended order takes the place of the previous order. The Order Number remains the same.

Use price override

The agent supervisor may encounter situations where, for example, the price of an item changes before the order is shipped, or you want to provide assistance to a shopper due to issues with a previous order.

The price override feature allows an agent supervisor during the remorse period to impose a more favorable price to order items. This feature can be activated or deactivated in the administration interface under Agent Settings.

To use price override:

1. Open the order.
2. In the Order Items section, click the **pencil** (Override Price) icon next to the product subtotal amount. The **Override Price** dialog appears.
3. Make any changes to the Subtotal and Override Reasons required fields, and click **Save**.

When the order is placed, and the order is opened for review, under Order Items, next to the subtotal of an item where a price override occurred, an icon is displayed showing the Price Override Reason and the agent who completed the override.

Amend PayPal orders

Orders cannot be placed with PayPal through the Agent Console. Other operations, such as cancel, return, and exchange, can be performed.

Note: If the order is paid for using the PayPal Authorize option in the storefront, such orders are available for operations such as cancel, edit, return, and exchange. If the PayPal Settle option is selected while placing the order, the edit order operation is not available.

Depending on the PayPal Payment Capture Action option set in the Oracle Commerce administration interface, PayPal payments involve one of the following:

- Immediate authorization and settlement/capture of funds from the shopper's PayPal account. In the Payment Capture Action options, this is called Order Placed. This may be used, for example, in instances where you are selling digital downloads.
  If this option is used, an order cannot be amended.
- Authorization only, awaiting shipment of the product. In the Payment Capture Action options, this is called Order Shipped. In this case, the funds are not captured from the shopper's PayPal account until the order is fulfilled. The system does not immediately transfer funds to your account but places a hold on the funds in the shopper's PayPal account. This capture is completed outside of the Oracle Commerce environment. The shopper is able to amend his or her order if this option is selected and you allow order amendment.

When working with orders involving the Order Shipped PayPal payment option, consider the following:

- If there is an amendment to the order that results in no change to the order amount (for example, a straight substitution of one item for another), no action is necessary as Oracle Commerce uses the previous authorization received from PayPal.
- If there is an amendment that results in a decrease in the order amount, no action is taken on the previous authorization. The extra amount in the authorization lapses, and the reduced amount becomes the amount PayPal captures.
- If there is an amendment that results in an increase in the order amount, the shopper is required to use a credit/debit card, gift card, or combination to resolve the difference. PayPal is also not involved as they have already authorized the original amount. The shopper is not expected to share his or her PayPal credentials with agent.
- If there is a cancellation of the entire order, the PayPal authorization is voided and not allowed to lapse. Because no funds have been captured, there may be no fees assessed by PayPal.

In the case of the Order Placed option, if a refund is required, the Oracle Commerce system initiates a refund call to PayPal, because the payment has already been captured. PayPal completes the refund to the shopper's account.

Note: In the case of refunds, depending upon your arrangement with PayPal, you may be charged a fee.

For more information, see Configure the PayPal integration in *Using Oracle Commerce*.

Cancel an order

You can cancel an order through the Agent Console if specific criteria are met.

You can cancel an order in the following circumstances:

- If your business has implemented a remorse (order hold) period and the order falls within the designated time period. The order status is set to Submitted for Fulfillment, but the order can still be amended.
- You are working with an order that has not been placed. In this case, delete all the items in the customer shopping cart. The status of the customer cart is returned to New from Complete.

If the order falls within the remorse period, the order status is still designated as Submitted for Fulfillment.

To cancel an order within the remorse period, complete these steps:

1. Locate the order using the Order search feature and click the order link. The **Order Details** page appears.
2. Click **Cancel Order**.
3. (Optional) Enter a reason for cancellation.
4. Click **OK**.

If the order is outside the remorse period, the **Cancel Order** button is not displayed. The shopper may want to start a return or exchange.

An order using PayPal as a payment option can also be cancelled. The cancel order request is made through PayPal.

View order history

You can review all orders placed by a shopper through the Order History display.

Complete the following steps:

1. Click the **Customers** button at the top of the page and complete a customer search.
2. From the Customer Results table, click the number in the Orders column that corresponds to the customer you are interested in. The **Order History** table is displayed.
   The number in the Order column indicates how many orders the customer has previously placed.
3. To view a specific order, click the associated link in the Order # column. Clicking the link displays the **Order Details** page.

Process returns

Either agents or shoppers can begin the returns process. Only someone with the role containing the Agent Supervisor privilege has permissions to complete refund price adjustments and initiate refunds.

The Oracle Commerce system provides automatic refunds (handled directly for credit card refunds, for example) and manual refunds (handled by the merchant). Manual refunds are handled in cases where the order was placed using cash or a third-party gift card, for example.

Note: If the partial returns feature is enabled for your site and the amount to be credited is more than the amount to be debited, the manual refund method is required by the merchant as the system cannot handle this type of return automatically.

Shopper-initiated returns are available in the storefront by default. If the item is designated as returnable, the shopper accesses his/her order history and selects the item to be returned. The shopper enters the quantity and return reason, and submits the request. A confirmation is sent to the shopper displaying a unique return ID as well as a return email with instructions on how to send the package back. By following the instructions and returning the item to your store, a refund is issued.

The return request returns the values for the refund. The refund amount is displayed in the refund screen, and, if you have the correct privilege, you can adjust the amount. The `refundBalance` endpoint is triggered whenever the amount in the refund methods is adjusted manually, and returns suggestions for the refund amount. Note that it is recommended that you re-verify the suggested amount due to return exchange behaviors. In some cases, you may be unable to make changes to the suggested refund values. This may occur if you create multiple return requests for an order. If this is the case, you can add and make modifications using another refund type and complete the transaction.

Some base items may also include add-on items (additional items purchased to enhance or modify the base item for sale, for example, warranties and monogramming). Any regular item can also be an add-on product and can be purchased separately as well as an add-on item to any product as long as it is associated to the main product. While they do possess their own SKU, SKU description, and attributes, these add-on items cannot be purchased separately and are flagged as not for individual sale in the administration interface. Add-on items appear in the same product detail line as the base item. You can view details and edit add-ons dialogues prior to placing the order. Once the order has been placed, you can review the order details, included together in the product line. You cannot return an add-on item separate from the main product.

Once the item or items are received in the warehouse and the returns are acknowledged by Oracle Commerce, the stock update for returned items is dependent on the disposition reason. On the Return Request – Acknowledge Receipts page, you can add the disposition reason for return items. This action can update the receipt of items in the warehouse.

Based on the condition of the items received, the receipt is marked against each return item and the inventory stock is updated. If the condition of the item received is good, the return reason can be marked as Accepted, meaning items received are accepted by the merchant, are in good condition, and can be added back into the inventory stock.

If the return reason is marked as Rejected for a particular item, the item is not added back into stock. Disposition reasons such as Rejected With Refund and Rejected Without Refund do not result in an inventory update or a return of the item received back into the warehouse inventory.

For returns that involve a mix of currency types, `allowMixOfCurrencies` must be set to true at order level while creating an order.

To add disposition reasons, use the `createReason` endpoint in the Admin API. For more information, see the REST API Admin documentation.

Processing returns as an agent involves the following phases:

- Obtaining a role with the Agent Supervisor privilege to perform any refund price adjustments or to initiate a return.
- Creating the return request for the item or items to be returned.
- Accepting the items received from the customer and approving the Return Request.
- Processing the refund to the customer from either single or multiple payments mode. For more information, see Understand single and multiple payments.

A note about non-returnable items

If an item has been designated as non-returnable (for example, when there are no shippable goods or a service), neither the shopper nor the agent can create a return request for that item. Any other items contained in the same order that are not marked as non-returnable are eligible for a return request. For example, if an order contains only items that are non-shippable (where the `Shippable` flag has been set to `false`) or assetable (where the

`Assetable` flag has been set to `true` to indicate that the product should be tracked as an asset) none of the item in the order can be returned. If the order contains both non-shippable and shippable items, the shippable items can be returned.

Note: If an item, which is configurable and returnable, has a child item that has a non-returnable service (meaning that it is non-shippable or assetable), neither the parent or the child can be returned.

To process and submit the initial return request:

1. Complete an order search, and click the appropriate order number link to open the Order Details page.
   To continue with a return, the order must be in Fulfilled status. The customer may have already received the order, but this is not a requirement to process and submit the initial return request.

   If you search by status, the following table shows the choices:

Status	Description
Being Processed	An intermediate system status used prior to moving to another status.
Fulfilled	The order is complete, shipped, and delivered to the customer.
Pending Removal	An intermediate system status used prior to moving to another status.
Removed	The order has been canceled and not shipped to the customer.
Submitted to Fulfillment	The order has been submitted to the warehouse staff to begin fulfillment. If the order is within the remorse period, the order can be amended.

2. On the Order Details page, you can click the **Return History** link to review any past returns associated with this order. The Return History shows the following:
   - Any previous Return Request numbers. If your sites use an external returns validation/tracking system, a return merchandise authorization (RMA) number and possible return tracking information may also be displayed.
   - When each request was made.
   - The number of items involved.
   - The refund amount, status, and actions associated with the return.
3. In the Order Items section, click **Return Items** to create a return request for the order. The Return Items button is displayed even if there are no items available to be returned. In this case, when you click the button, a message is displayed informing you that there are no items available to be returned.
   When a shopper is returning a product whose quantity is made up of externally-priced (for example, a shopper bidding on an item) and additional standard list priced items, the standard list priced items are returned first. If additional items are selected for return beyond the number of standard list priced items, the externally-priced items would be returned.

   As an example, a shopper has purchased a product. The first two of the quantity have been externally priced at $50.00, and the shopper has purchased an additional item at the list price of $75.00. The shopper is now returning two and keeping one. The first item would be returned at $75.00 and the second item

would be returned at $50.00. The shopper would still have one item priced at $50.00.

For returns that result in orders with a negative refund, see Create return requests for negative refunds.

4. Click the icon to review the quantity shipped, the quantity previously returned, and the quantity of items available for return.
5. In the item Quantity field, type the quantity of items the customer wants to return. The quantity must be equal to or less than the number of items available for return.
   Note: If the order was sent to multiple addresses, select the item and address associated with the item to be returned. For more information, see Understand shipping.
6. In the Return Reason column, select a reason from the menu.
7. Enter the quantity and reason for return for any other items in this order that are available for return.
8. Click **Calculate** to show amount of estimated refund for the return request.
   Tax is included in the item price, so a refund of the item price automatically refunds the corresponding tax amount. You can modify the tax amount on the Refund page by clicking the Edit icon next to the tax amount. Modification of the tax amount does not impact the refund amount.
9. Once you complete the processing of the return items, click **Submit**. The return request has been created.
   The customer should be advised as to how to package and ship the items to the appropriate location. When the items are returned to your ship-to location, the refund can be processed.

Once the customer has returned the item or items to the appropriate ship-to location, you can process the return to reimburse the customer.

To accept the items and process the refund:

1. Click **Returns Processing** at the top of the page and complete a return request search. Return request search criteria you can use include the following:
   - Return request number.
   - Order number.
   - Site.
   - Created in last days/weeks/months.
   - RMA number; a return merchandise authorization number available if your sites use an external returns validation/tracking system.
   - Email.
   - Status.
   - Account; available if your sites have implemented account-based commerce.
2. Once you locate the Return Request, you can either review the entire order and its history by clicking the Order number link, or click Receive in the Actions column.
3. If you reviewed the order, click Receive for the appropriate item in the Return History section.
4. Once you click **Receive**, the Acknowledge Receipts page appears.
5. From the menus, select the quantities to receive and the disposition of the return request. The disposition can be one of the following:

- Accepted with refund.
- Rejected with refund.
- Rejected without refund.

6. Add any comments to the field.
7. Click **Save** to save the quantity and other information.
8. Click **Refund**. The Return Request – Refund page appears.
9. On the Return Request – Refund page, do the following:
   - When completing a return, you can refund the amount due to any of the credit/debit cards or gift cards used for the order. The order amount can be distributed in any way you find necessary. For example, if a shopper paid for the order using two gift cards and a credit card, and the refund was $30, you could put the full amount on a single card or put $10 on each of the three cards.
     For more information on refunding from the multiple payments mode, see Understand single and multiple payments.
   - If no changes to the system-generated refund amounts are made, click **Initiate Refund**.
     If the Initiate Refund button is replaced by the Mark Manual Refund button, see Mark return requests for manual refunds.
   - If you make changes to any of the amounts, click **Apply** to save the changes, and then click **Initiate Refund**.

   Any agent can edit and save the refund values. If the values are edited, only an Agent Supervisor can process the refund.

   In an order using the PayPal option, the Refund Mode section on the Return Request page shows the refund is made through PayPal along with the amount to be refunded.

The status for this order under Return History changes to Complete.

Mark return requests for manual refunds

You may encounter situations where the system does not have the information required to initiate a refund through the application. In this case, on the Return Request -- Refund page, the Initiate Refund button is replaced by the Mark Manual Refund button.

When you click this button, the Return Request is moved to the Manual Refund status for further actions outside of the Agent Console. Other means may then be required to process the refund, for example, processing the refund through the payment gateway console or having a check written for the amount.

Create return requests for negative refunds

You may encounter cases involving order level discount adjustments, buy-one-get-one (BOGO) adjustments, or gift with purchase (GWP) adjustments where the refund calculation is shown as a negative refund value. Oracle Commerce allows for the creation and processing of a return request. The amount of the refund is shown as $0 with the negative amount associated with the order in the event of any future returns.

For example, a shopper purchases several items totaling $104.00 and uses the promotion code SAVE20 to save 20% on an order of $100 or more. After the order has

been fulfilled, if the shopper decides to return one of the items totaling $5, the order total would fall below the $100 required to use the promotion code. The 20% discount is no longer valid, and the result is a negative value for the refund.

To process a negative refund:

1. Create the return request as you normally would. The Promotion Summary shows that the promotion code has been removed.
2. Once the item or items have been received, acknowledge the receipt. With the request successfully received, a note is displayed informing you that the Other Adjustment has been updated from $0 to the negative amount to allow for processing of the refund.
3. Apply the refund and mark it for manual refund.
4. Mark the refund complete, and include any comments associated with the refund. The refund in this case is $0.00.

Complete returns in multiple payments mode

Once an order has been fulfilled, if it was completed in multiple payments mode, you can adjust the amounts originally assigned to each payment type.

To complete a return in multiple payments mode:

1. Search for and display the order.
2. In the Order Items section, click **Return Items**.
3. Identify those items and/or quantities to be returned.
4. Click **Refund**. A return request is generated.
   On the Return Request Refund page, under the Return Items section, the list of payments used for this order is displayed along with, by default, the amounts assigned to each payment group and the total amount to be refunded.
5. You can edit these refund amounts as long as the total refund amount does not exceed the amount authorized.
6. Click **Initiate Refund**. The refund is made to the cards or invoices used that equal the refund amount.

Returned fulfill orders

You can ensure that orders are updated correctly using the `updateOrder` endpoint so that returns are initiated properly.

To ensure that fulfilled orders can be returned properly:

1. Obtain the order details using the `getOrder` Admin endpoint. For example:

   ```
   GET  /ccadmin/v1/orders/{id}
   ```

2. Change the order's state to `NO_PENDING_ACTION`.
3. For each object in the shipping groups, change the state to `NO_PENDING_ACTION` and change the `ShippingGroupCommerceItemRelationship` state to `DELIVERED`.
4. For each object in the payment groups, perform the following. Note that you do not need to perform these steps if the payment group is already in the `SETTLED` state:
   a. Add a `debitStatus` element with the required details.

b. Add the `amountDebited` element and make the value equal to the amount settled.

c. Change the state of each object to `SETTLED`.

5. Update the order using the `updateOrder` Admin endpoint. For example:

```
PUT  /ccadmin/v1/orders/{id}
```

To return partially fulfilled orders:

1. Obtain the order details using the `getOrder` Admin endpoint. For example:

```
GET  /ccadmin/v1/orders/{id}
```

2. Change the `ShippingGroupCommerceItemRelationship` state to `DELIVERED`.
3. If the payment is settled for a payment group, the payment should contain the `debitStatus`, `amountDebited` and its status should be `SETTLED` to handle refunds.

The refund amount is refunded to the payment group only if the sum of the settled amount matches the refund amount. If the sum does not match the refund amount, the return request will be marked for a manual refund.

Manually refund orders:

Note that you will have to manually modify the order to create refunds in the following cases, as payment groups are part of the order and the amount settled is not included when calculating the overall settled amount:

- If the amount debited at the payment group level is zero.
- If one of the following payment methods have been used when creating an order:
  - PayULatm
  - Cash
  - Invoice
  - Chase physical gift card
- If the orders state is in NO_PENDING_ACTION by the payment group is still in the authorized state.

Exchange products

Agents can exchange products in the catalog without requiring agent supervisor approval or the creation of another transaction. Only the same item can be exchanged. Items may be exchanged for a variant of the same item (for example, size or color differences). No change in value is allowed.

If a product has no variants (for example, a different size or color), the customer receives a replacement for the product he or she requests.

Completing an exchange involves the following procedures:

- Creating an exchange request for the associated order.
- Accepting the returned item.
- Processing the exchange.

Be aware of cases in which the item was initially available at the time of placing an order, but later the item became unavailable (for example, because the SKU was deactivated). In these cases, an exchange is not possible, and you should proceed instead with a refund for the item.

For more information, see Create and work with SKUs in *Using Oracle Commerce*.

To create an exchange request:

1. Complete an order search, and click the appropriate order number link to open the **Order Details** page.
   To continue with an exchange, the order must be in Fulfilled status. The customer may have already received the order, but this is not a requirement to process and submit the initial exchange request.

   Note: You can change the values for the custom commerce item properties when receiving the exchange. However, you can only change values for the customization properties selected during order creation. For more information, see Understand dynamic commerce item properties.
2. Click **Exchange Items**. An Exchange Request page associated with the order appears. The **Exchange Items** button is displayed even if there are no items available to be exchanged. In this case, when you click the button, a message is displayed informing you that there are no items available to be exchanged.
3. Click the **information** icon to review the quantity of items that can be exchanged and available for return.
4. In the Quantity field, type the quantity to be exchanged. This number cannot be higher than the quantity available for return.
   If the order was sent to multiple addresses, select the item and address associated with the item to be exchanged. For more information, see Understand shipping.
5. Select the reason for the exchange from the menu. The Exchange Amount displays the amount associated with this exchange.
6. Click **Exchange**. The Exchange History for this order now shows the new Exchange Request.
7. At this time, an exchange order is created for the same value of the item with zero shipping and tax. This order is in a hold state, pending receipt of items.
   If there are no variants available for a product (for example, a different size or color), then the customer receives an exact replacement for the item. This would be in cases, for example, where a product was received damaged.

   If a customer is interested in receiving a similar product (for example, an entirely different DVD for the one he or she purchased), then process the order as a return for refund and have the customer order the new product.

To accept a returned item for exchange:

1. With the exchange item returned, search for the Exchange Request associated with the order by doing the following:
2. Click **Return Processing** at the top of the page and search for the exchange request. In the Return Request Results table, find the associated Return number, and in the Actions column, click **Receive**.
3. Do one of the following:
   - Complete an order search. In **Order Details**, under **Exchange History**, in the **Action** column, click **Receive**. The Acknowledge Receipts page appears.

- Enter a number in the **Quantity to Receive** field and select a disposition from the menu. Add any comments in the field as needed.

4. Do one of the following:
   - Click **Save** to place the request in the queue for later attention. The request appears in the Exchange History and in Returns Processing. The new order is an Exchange Order type. You can use the hyperlink of the exchange order in the Exchange History region to view it.
   - Click **Process Exchange Order** to complete the exchange. This action releases the hold on the exchange order and it is ready to be picked up for fulfillment.
   - Click **Refund** to complete a refund rather than an exchange. For more information, see Process returns.

Generate Appeasements

Appeasements are a way that you can alleviate customer dissatisfaction.

Should one of your shoppers have a bad experience with their order, issuing an appeasement is a way to prevent them from canceling their orders. Appeasements allow an Agent to provide some form of monetary, or non-monetary, compensation.

For example, if a shopper places an order and is unsatisfied with the products, an Agent could issue an appeasement for a single item in the order, or for the total order and shipping charges.

Authorized Agents have the capability to provide appeasements to your shoppers by offering a refund, store credit, credit memo or other method. Once the appeasement has been made, an email is sent to the shopper with the appeasement information.

Understand appeasements

You can use the Admin and the Agent API to develop an appeasement workflow. The Agent API contains endpoints that allow an Agent to create and review appeasements, as well as to submit appeasements. Agents who have the role of Supervisor can issue appeasements for a submitted or fulfilled order. Appeasements can be issued using one or more appeasement refunds. When an Agent issues an appeasement, they also provide notes and reasons for the appeasement. Agents are able to see a history of any previous appeasements for a customer or an order.

Appeasements are typically associated with an order, and the default appeasement type is `order`. You can provide custom properties for the order at both the appeasement level and the refund level. Appeasements totals are limited by the order total, meaning that the total number of appeasements and returns cannot exceed the total of the order. However, you can add limits and provide additional validation as needed by using a validation webhook.

You can issue appeasements in any currency, but when performing order-based appeasements, the currency must match that used in the order. If an order contains multiple currencies, such as in both dollars and loyalty points, appeasements for that order can be issued in both dollars and loyalty points.

Once generated, the system associates the appeasement with the Agent who performed the appeasement. All submitted appeasements are validated by default. For example, the system validates that an appeasement does not exceed the total value of the order.

Use appeasements

There are several scenarios where you might want to use appeasements. For example, if a shopper places an order with priority shipping, and the order arrives late, the Agent could create an appeasement for the order.

Should a shopper not receive their order on time, and then discover that one of the items within the order is damaged, an Agent can create multiple appeasements associated with the order. For example, the Agent could provide an appeasement for the shipping total, and then another appeasement for store credit for the damaged item.

Agents can refund the appeasement value using the payment instrument used to create the order. The refund structure and refund type used for different payment types is described later in this section. Although the amount credited and other details of the original payment method used in the order are updated automatically, an Agent can issue appeasement refunds to another payment method, such as issue store credit or loyalty points.

When issuing an appeasement, the Agent must set an appeasement value that does not exceed the order total. Earlier appeasements and refund values are also included in the equation when creating a new appeasement.

You can configure your appeasement process to include work with an external system, which receives and returns appeasement information. An Agent can send appeasement details to use, as well as get, validations from an external system.

You cannot create appeasements for invalid or deleted sites, but you can create an appeasement for an inactive organization. This applies only to Agent-based appeasements, and does not apply to Admin-based appeasements.

Use the Appeasement API

Use the Appeasement API to generate appeasements, which includes an Agent API and an Admin API. The Agent API allows an Agent to create and submit an appeasement. The Admin API allows you to create and submit appeasements, as well as update the status of an appeasement. In general, the Agent API has more restrictions, such as it allows only the creation and review of appeasements. However, the appeasements are made in the `INCOMPLETE` state for the Agent API. The Admin API allows actions in appeasements that are in any state.

The Admin APIs allow you to configure the type of appeasement that can be provided by the Agent Supervisor, as well as the reason for appeasements

When an appeasement is made using the Agent API, the default `OriginOfAppeasement` is set to `contactCenter`. An Agent can create an appeasement for a submitted or fulfilled order or any custom appeasement types that have been created in the API.

You can use the Admin API to import appeasements form an external appeasement system. The Admin API can also provide a way to configure appeasement types and appeasement reasons. Whenever the `originOfAppeasement` does not contain a value, the default is `external`. You can also create custom properties for appeasement and refunds.

The following properties are used by the endpoints in the Appeasement API:

Property	Description
`agentId`	The ID of the Agent who issued the appeasement.

Property	Description
amount	Indicates the amount to be appeased with a specific payment instrument. This optional field is used for each refund in the appeasementRefunds array.
appeasementRefund	This array includes the following properties: amount paymentGroupId currencyCode refundType
comments	Comments, stored in a JSON array format, added by the Agent.
creationDate	The date and time that the appeasement was initiated. Different than submittedDate, which identifies when an appeasement has been submitted.
currencyCode	This indicates the currency in which a monetary appeasement is to be paid. The currency code is added to the appeasement refund level to support orders that use multiple currencies.
id	The ID of the appeasement.
notes	Notes that provide additional information regarding the appeasement.
orderId	The order ID is mandatory for all appeasement types. The order must be a fulfilled order or a submitted order that is past its remorse period.
originOfAppeasement	If you are using the Agent API, this defaults to contactCenter. If you are using the Admin API, this defaults to external.
paymentGroupId	The payment group ID, which is part of the appeasementRefund property, is required to identify to which card the refund should be applied.
profileId	The profile ID be should be that of a registered customer. Customers who are pending self-registration are not valid.
reason	The reason for the appeasement. Reasons are created in by using the createReason endpoint in the Admin API. Default reasons include: orderArrivedLate orderArrivedDamaged itemArrivedLate itemArrivedDamaged didNotLikeItem goodwillGesture productComplaint

Property	Description
refundType	This property, which is in appeasementRefund, identifies the type of refund to issue. The supported refund types are: creditCard – This refund type is used for Chase Credit Card. customCurrencyGroup – This refund type is used for loyalty. externalRefund – This refund type is used for invoice, cash, payUlantam, coupons and instorePaymentMethod. onlinePaymentGroup – This refund type includes PayPal. physicalGiftCard – This refund type is used for gift cards. storeCredit – This issues a store credit. tokenizedCreditCard – This refund type is used for CyberSource payments.
state	The state of the appeasement, which is available at both the appeasement and appeasementRefund level. Note that the state can be updated if you use the Admin API, however, when using the Agent API, you can only view the property.
submittedDate	Identifies when an appeasement has been submitted. Different than creationDate, which identifies when an appeasement has been initiated.
type	Provides the type of the appeasement. The default type is order.

Initiate an appeasement

The initiateAppeasement endpoint in the Agent API validates the order and returns the appeasement refund types with the maximum amount that can be issued. To initiate an appeasement, issue a POST command. Use this endpoint to see the possible refund types, refund structure and the maximum refund amount allowed for each refund type.

For example, you could create an appeasement by issuing the following command:

```
POST /ccagent/v1/appeasements/initiate

{
 "orderId":"o30425"
}
```

If you do not specify an order ID, the system returns information for an appeasementRefund of an externalRefundType. You can use the response to create a new appeasement or update

and submit an existing incomplete appeasement. The response payload might be similar to the following:

```
{
 "orderId": "o30425",
 "appeasementRefunds": [
   {
     "paymentGroupId": "pg30442",
     "refundType": "tokenizedCreditCard",
     "amount": 0,
     "maximumRefundAmount": 97.18,
     "id": "100020",
     "state": "INCOMPLETE",
     "currencyCode": "USD"
   }
 ],
 "links": [
   {
     "rel": "self",
     "href": "http://localhost:9080/ccagent/v1/appeasements/initiate"
   }
 ]
}
```

Create an appeasement

When you create an appeasement, it must have at least one associated `appeasementRefund` that contains the mandatory `refundType` property. With the exception of refunds that use the `externalRefund` refund type, all refunds require the `paymentgGroupId` and the amount properties.

Create an appeasement with the Agent API

You can use the Agent API `createAppeasement` endpoint to create an appeasement. Use the endpoint when working with a valid site or organization. The endpoint performs a number of validations. A POST request might be similar to the following:

```
POST /ccagent/v1/appeasements

{
  "notes": "The customer complained that the order arrived late,
      and asked for a refund of the shipping amount. Providing 15
USD.",
  "orderId": "o30444",
  "appeasementRefunds": [
    {
      "currencyCode": "USD",
      "amount": 5,
      "refundType": "externalRefund",
    },
    {
      "amount": 10,
      "description": "Store Credit",
      "refundType": "storeCredit",
      "paymentGroupId": 1223232
```

```
    }
  ],
  "profileId": "120222",
  "type": "order",
  "reason": "orderArrivedLate",
  "comments": [
    {
      "comment": "Appeasement to be settled as a priority. Valued customer."
    }
  ]
}
```

The response payload to the `POST` request may be similar to the following:

```
{
  "agentId": "BobAFrette",
  "notes": "The customer complained that the order arrived late, and asked
for a
     refund of shipping amount. Providing 15 USD.",
  "orderId": "o30444",
  "damagedLineItemId": "ci56873",
  "lastModifiedDate": "2020-03-31T10:35:14.016Z",
  "appeasementRefunds": [
    {
      "amount": 5,
      "customerPreferredGiftCardId": "FC13213",
      "id": "1100005",
      "state": "INCOMPLETE",
      "refundType": "externalRefund",
      "currencyCode": "USD",
      "paymentGroupId": null
    },
    {
      "amount": 10,
      "id": "1100006",
      "state": "INCOMPLETE",
      "refundType": "storeCredit",
      "currencyCode": "USD",
      "paymentGroupId": 1324255
    }
  ],
  "creationDate": "2020-03-31T10:35:14.016Z",
  "profileId": "120222",
  "state": "INCOMPLETE",
  "id": "app160003",
  "originOfAppeasement": "agent",
  "type": "order",
  "reason": "orderArrivedLate",
  "links": [
    {
      "rel": "self",
      "href": "http://localhost:9080/ccagent/v1/appeasements"
    }
  ],
```

```
  "comments": [
    {
      "agentId": " BobAFrette ",
      "comment": "Appeasement to be settled on priority. Valued
customer.",
      "id": 100001,
      "creationDate": "2020-03-20T04:56:13.935Z"
    }
  ]
}
```

Once the appeasement has been created, it remains in the `INCOMPLETE` state. The `SUBMITTED` state is available only in the Admin API.

Create an appeasement with the Admin API

The Admin API `createAppeasement` endpoint creates an appeasement, and is also used to import appeasements from an external system. Whenever the `originOfAppeasement` does not contain a value, the default is `external`.

The following example `POST` request issues an appeasement because an item arrived late:

```
POST /ccadmin/v1/appeasements
  {
    "notes": "The customer complained that the order arrived late,
    and asked for a refund of the shipping amount. Providing 15 USD.",
    "orderId": "o30444",
    "appeasementRefunds": [
      {
        "currencyCode": "USD",
        "amount": 5,
        "refundType": "externalRefund",
        "state": "COMPLETE"
      },
      {
        "amount": 10,
        "description": "Store Credit",
        "refundType": "storeCredit",
        "paymentGroupId": 1223232,
        "state": "COMPLETE"
      }
    ],
    "profileId": "120222",
    "type": "order",
    "reason": "orderArrivedLate",
    "comments": [
      {
        "comment": "Appeasement to be settled as a priority. Valued
customer."
      }
    ],
    "state": "COMPLETE"
  }
```

The response may be something similar to the following:


```
{
  "reason": {
    "id": "itemArrivedLate",
    "readableDescription": "Item Arrived Late"
  },
  "type": {
    "id": "order",
    "displayName": "Order",
    "description": "Appeasement to be applied on a given order."
  },
  "agentId": " BobAFrette",
  "notes": "The customer complained that the order arrived late, and asked
for
      a refund of shipping amount. Providing 15 USD.",
  "appeasementRefunds": [
    {
      "amount": 10,
      "currencyCode": "USD",

      "id": "1100006",
      "state": "INCOMPLETE",
      "refundType": "storeCredit",
      "paymentGroupId": "pg100122"
    },
    {
      "mode": "giftCard",
      "amount": 5,
      "currencyCode": "USD",
      "state": "INCOMPLETE",
      "id": "1100005",
      "refundType": "externalRefund",
      "paymentGroupId": "pg100111"
    }
  ],
  "orderId": "o10052",
  "lastModifiedDate": "2020-03-06T14:49:48.546Z",
  "creationDate": "2020-03-06T14:49:48.546Z",
  "profileId": null,
  "state": "INCOMPLETE",
  "originOfAppeasement": "external",
  "id": "app30024",
  "comments": [
    {
      "agentId": " BobAFrette",
      "comment": "Appeasement to be settled as a priority. Valued customer.",
      "id": 100001,
      "creationDate": "2020-03-20T04:56:13.935Z"
    }
  ]
}
```


Update an appeasement

The `updateAppeasement` endpoint allows you to update an existing appeasement when you issue a `PUT` command and specify the appeasement ID.

Update an appeasement with the Agent API

Note that only incomplete appeasements can be updated in the Agent API. For example:

```
PUT /ccagent/v1/appeasements/{id}
```

The following is an example of a request payload:

```
{
  "notes": "The customer complained that the order arrived late, and
asked for a
    refund of shipping amount. Providing 15 USD.",
  "appeasementRefunds": [
    {
      "amount": 5,
      "refundType": "externalRefund",
      "id": "1100005",
      "customerPreferredGiftCardId": "FC13213",
      "paymentGroupId": "122323",
      "currencyCode": "USD"
    },
    {
      "amount": 10,
      "id": "1100006",
      "refundType": "storeCredit",
      "paymentGroupId": 1223234,
      "currencyCode": "USD"
    }
  ],
  "comments": [
    {
      "comment": "Appeasement to be settled as a priority. Valued
customer."
    }
  ]
}
```

Note that the `damagedLineItemId` and the `customerPreferredGiftCardId` are custom properties that have been created at the appeasement level and the appeasement refund level.

The response payload to the above request might be similar to the following:

```
{
  "agentId": " BobAFrette",
  "notes": "The customer complained that the order arrived late, and
asked for a
    refund of shipping amount. Providing 15 USD.",
  "orderId": "o30444",
  "damagedLineItemId": "ci56873",
```

```
  "lastModifiedDate": "2020-03-31T10:35:14.016Z",
  "appeasementRefunds": [
    {
      "amount": 5,
      "customerPreferredGiftCardId": "FC13213",
      "id": "1100005",
      "state": "INCOMPLETE",
      "refundType": "externalRefund",
      "currencyCode": "USD",
      "paymentGroupId": null
    },
    {
      "amount": 10,
      "id": "1100006",
      "state": "INCOMPLETE",
      "refundType": "storeCredit",
      "currencyCode": "USD",
      "paymentGroupId": 1324255
    }
  ],
  "creationDate": "2020-03-31T10:35:14.016Z",
  "profileId": "120222",
  "state": "INCOMPLETE",
  "id": "app160003",
  "originOfAppeasement": "agent",
  "type": "order",
  "reason": "orderArrivedLate",
  "links": [
    {
      "rel": "self",
      "href": "http://localhost:9080/ccagent/v1/appeasements/app10001"
    }
  ],
  "comments": [
    {
      "agentId": " BobAFrette",
      "comment": "Appeasement to be settled as a priority. Valued customer.",
      "id": 100001,
      "creationDate": "2020-03-20T04:56:13.935Z"
    }
  ]
}
```


Update an appeasement using the Admin API

The Admin API `updateAppeasement` endpoint updates an appeasement with the information provided. This endpoint also updates appeasements refund information. To use this endpoint, issue a `PUT` command:

```
PUT /ccadmin/v1/appeasements/{id}
```

A request may be similar to the following:

```
{
  "appeasementRefunds": [
    {
      "state": "COMPLETE",
      "id": "1100005",
      "refundType": "externalRefund"
    },
    {
      "state": "COMPLETE",
      "id": "1100006",
      "refundType": "storeCredit"
    }
  ],
  "state": "COMPLETE",
  "comments": [
    {
      "agentId": " BobAFrette",
      "comment": "Appeasement to be settled as a priority. Valued
customer.",
      "id": 100001,
      "creationDate": "2020-03-20T04:56:13.935Z"
    },
    {
      "agentId": " BobAFrette",
      "comment": "Appeasement settled",
      "creationDate": "2020-04-20T04:56:13.935Z"
    }
  ]
}
```

The response might be something similar to the following:

```
{
  "reason": {
    "id": "itemArrivedLate",
    "readableDescription": "Item Arrived Late"
  },
  "type": {
    "id": "order",
    "displayName": "Order",
    "description": "Appeasement to be applied on a given order"
  },
  "agentId": " BobAFrette",
  "notes": "The customer complained that the order arrived late, and
asked for
      a refund of shipping amount. Providing 15 USD.",
  "appeasementRefunds": [
    {
      "mode": null,
      "amount": 20,
      "currencyCode": "USD",
      "transactionNumber": "TRX1006",
```

```
      "description": "Store Credit",
      "id": "1100006",
      "state": "COMPLETE",
      "refundType": "storeCredit",
      "paymentGroupId": "pg100122"
    },
    {
      "mode": "giftCard",
      "amount": 10,
      "currencyCode": "USD",
      "transactionNumber": "TRX1089",
      "description": "Amazon Gift Card",
      "state": "COMPLETE",
      "id": "1100005",
      "refundType": "externalRefund",
      "paymentGroupId": "pg100111"
    }
  ],
  "orderId": "o10052",
  "lastModifiedDate": "2020-03-06T14:49:48.546Z",
  "creationDate": "2020-03-06T14:49:48.546Z",
  "profileId": null,
  "state": "COMPLETE",
  "originOfAppeasement": "external",
  "id": "app30024",
  "comments": [
    {
      "agentId": " BobAFrette",
      "comment": "Appeasement to be settled as a priority. Valued customer.",
      "id": 100001,
      "creationDate": "2020-03-20T04:56:13.935Z"
    },
    {
      "agentId": " BobAFrette",
      "comment": "Appeasement settled",
      "id": 100010,
      "creationDate": "2020-03-20T04:56:13.935Z"
    }
  ]
}
```


Submit an appeasement

The `submitAppeasement` endpoint in the Agent API allows you to submit an incomplete appeasement to the system when you issue a `POST` command. You can also use it to create and immediately submit a new appeasement or to submit an existing incomplete appeasement by providing the existing appeasement ID in the payload.

Note that only `INCOMPLETE` appeasements may be submitted using the Agent API.

The following example displays the format you should use to submit a request:

```
POST /ccagent/v1/appeasement/submit
```

A request payload might look like the following:

```
{
  "notes": "The customer complained that the order arrived late, and
asked for a
    refund of shipping amount. Providing 15 USD.",
  "appeasementRefunds": [
    {
      "amount": 5,
      "refundType": "externalRefund",
      "id": "1100005",
      "customerPreferredGiftCardId": "FC13213",
      "paymentGroupId": "122323",
      "currencyCode": "USD"
    },
    {
      "amount": 10,
      "id": "1100006",
      "refundType": "storeCredit",
      "paymentGroupId": 1223234,
      "currencyCode": "USD"
    }
  ],
  "comments": [
    {
      "comment": "Appeasement to be settled as a priority. Valued
customer."
    }
  ]
}
```

Delete an appeasement

The `deleteAppeasement` endpoints in both the Agent and the Admin API allow you to delete an incomplete appeasement from the system. Note that this endpoint in the Agent API is restricted to incomplete appeasements. The following is the format you should use to create a `DELETE` command:

```
DELETE /ccagent/v1/appeasements/{id}
```

Note: When you delete an order-based appeasement that is in the `COMPLETE` state, the appeasement history for that order will not contain that appeasement, allowing you to inadvertently provide appeasements that total more than the amount of the order.

List appeasements

The `listAppeasements` endpoints in both the Agent and Admin API list the appeasements that match a specified search criteria provided in SCIM query format. The following example looks for a specific profile that starts with the first name “Kim”:

```
GET /ccagent/v1/appeasements?q=profile.fistName co “kim”
```

This command might return a response such as this:


```
{
  "total": 2,
  "totalResults": 2,
  "offset": 0,
  "limit": 8,
  "items": [
    {
      "reason": {
        "id": "itemArrivedLate",
        "readableDescription": "Item Arrived Late"
      },
      "type": {
        "id": "order",
        "displayName": "Order",
        "description": "Appeasement to be applied on a given order"
      },
      "agentId": " BobAFrette",
      "notes": "Appeasement issued on customer's complaint about item
delivery.",
      "appeasementRefunds": [
        {
          "amount": 100,
          "currencyCode": "USD",
          "description": "Appeasement refund on item purchase",
          "state": "INCOMPLETE",
          "id": "AppRef0211",
          "refundType": "externalRefund",
          "paymentGroupId": "pg10331"
        }
      ],
      "orderId": "o10050",
      "lastModifiedDate": "2020-03-20T05:15:05.645Z",
      "creationDate": "2020-03-20T04:56:13.935Z",
      "profileId": "se-570031",
      "state": "PENDING_REFUND",
      "originOfAppeasement": "external",
      "id": "App00090",
      "comments": [
        {
          "agentId": "service",
          "comment": "Appeasement settlement has been expedite.",
          "id": 100011,
          "creationDate": "2020-03-20T04:26:13.935Z"
        }
      ]
    },
    {
      "reason": {
        "id": "itemArrivedLate",
        "readableDescription": "Item Arrived Late"
      },
      "type": {
        "id": "order",
```

```
        "displayName": "Order",
        "description": "Appeasement to be applied on a given order."
      },
      "agentId": " BobAFrette",
      "notes": "Appeasement issued because of a bad shipping
experience.",
      "appeasementRefunds": [
        {
          "amount": 20,
          "currencyCode": "USD",
          "state": "COMPLETE",
          "id": "101001",
          "refundType": "storeCredit",
          "paymentGroupId": "pg100992"
        }
      ],
      "orderId": "o10052",
      "lastModifiedDate": "2020-03-18T19:44:31.156Z",
      "creationDate": "2020-03-18T19:44:31.156Z",
      "profileId": "se-570031",
      "state": "COMPLETE",
      "originOfAppeasement": "external",
      "id": "App10001",
      "comments": [
        {
          "agentId": " BobAFrette",
          "comment": "Customer asked to settle appeasement amount to
the same
              credit card used for order payment.",
          "id": 100011,
          "creationDate": "2020-03-20T04:26:13.935Z"
        }
      ]
    }
  ]
}
```

Get an appeasement type

The `getType` endpoints in both the Agent and Admin API allow you to get details of a specific appeasement type by ID. The types that follow certain criteria can be obtained using the SCIM query format on the list appeasement types endpoint. Issue a `GET` command to list appeasement types:

```
GET /ccagent/v1/appeasementTypes/{id}
```

Provide a valid appeasement type, such as shipping, and the response may look similar to the following:

```
{
  "displayName": "Shipping Level",
  "description": "This option can be chosen to provide appeasement for
     the shipping amount in an order.",
  "active": true,
```

```
  "id": "shipping",
  "isOrderRequired" : true
}
```

List all appeasements types

The `listTypes` endpoints in both the Agent and Admin API provide a list of appeasement types when you issue a `GET` command. For example the following would return all active appeasement types:

```
GET /ccagent/v1/appeasementsTypes?q=active eq true
```

The default appeasement type is `order`. However, you can create custom appeasement types such as profile, shipping, item level, shipping group level, etc.

Create a type of appeasement

The `createType` endpoint in the Admin API allows you to create an appeasement type, as well as any associated translations. You can create an appeasement type by issuing a `POST` command. For example:

```
POST /ccadmin/v1/appeasementTypes
```

The following example shows how to create a new Shipping appeasement type:

```
{
  "id": "shipping",
  "displayName": "Shipping Level",
  "description": "This appeasement type can be chosen to provide appeasement
for
    the shipping amount in an order.",
  "active": true,
 "isOrderRequired" : true
}
```

You can extend the default appeasement types, which contain the following properties:

- `description`
- `active`
- `isOrderRequired`
- `type`
- `id`

Update appeasement types

The `updateType` endpoint in the Admin API allows you to update an appeasement type by issuing a `PUT` command using the following format:

```
PUT /ccadmin/v1/appeasementTypes/{id}
```

The following is an example of updating the a newly created Shipping appeasement type:

```
{
  "description": "This option can be chosen to provide appeasement for
the shipping amount in an order.",
  "isOrderRequired" : true
}
```

`Get an appeasement reason`

Appeasements contain appeasement reasons that identify why the appeasement was made. Default appeasement reasons include the following:

- `orderArrivedLate`
- `orderArrivedDamaged`
- `itemArrivedLate`
- `itemArrivedDamaged`
- `didNotLikeItem`
- `goodwillGesture`
- `productComplaint`

The `getReason` endpoint in the Agent API obtains the appeasement reason corresponding to the reason ID you provide in the path parameter when you issue a `GET` command. Use the following format when using this endpoint:

```
GET /ccagent/v1/reasons/{id}?type=appeasementReason
```

The `getReasons` endpoint in the Admin API gets all of the appeasement reasons available when you issue a `GET` command with the type query parameter. Use the following format:

```
GET /ccadmin/v1/reasons?type=appeasementReason
```

Use the following format to get a specific reason by its ID:

```
GET /ccadmin/v1/reasons?type=appeasementReason&id={id}
```

List all appeasement reasons for specific criteria

The `listReasons` endpoint displays all of the appeasement reasons that match the search criteria for reasons that you specify using the SCIM query format. The following example searches for all appeasement reasons that are active:

```
GET /ccagent/v1/reasons?q=active eq true&type=appeasementReason
```

The response to the example request might be similar to the following:

```
{
  "total": 2,
  "totalResults": 2,
  "offset": 0,
  "limit": 8,
  "items": [
    {
      "id": "goodwillGesture",
      "active": false,
      "description": "goodwillGesture",
      "readableDescription": "Goodwill Gesture"
    },
    {
      "id": "itemArrivedLate",
      "active": true,
      "description": "itemArrivedLate",
      "readableDescription": "Item Arrived Late"
    }
  ]
}
```

Create an appeasement reason

The `createReason` endpoint in the Admin API allows you to create an appeasement reason, as well as any translations needed when you issue a `POST` command. Use the following format:

```
POST /ccadmin/v1/reasons?type=appeasementReason
```

The following is an example of creating a Goodwill Gesture appeasement reason:

```
{
  "id": "goodwillGesture",
  "active": true,
  "description": "goodwillGesture",
  "readableDescription": "Goodwill Gesture"
}
```

Note that an external ID can be used if it is passed in with the request payload. The system generated ID is used by default.

The `active`, `description` and `readableDescription` properties are used to support additional reason types used in addition to appeasements and should be provided when you create an appeasement reason.

Update an appeasement reason

The `updateReason` endpoint in the Admin API updates an appeasement reason when you issue a `PUT` command. Use the following format:

```
PUT /ccadmin/v1/reasons?type=appeasementReason&id={id}
```

The following is an example of updating the Goodwill Gesture appeasement reason so that it is no longer active:

```
{
  "active": false,
  "description": "goodwillGesture",
  "readableDescription": "Goodwill Gesture"
 }
```

Delete an appeasement reason

The `deleteReason` endpoint in the Admin API allows you to delete an appeasement reason when you issue a `DELETE` command using the following format:

```
DELETE /ccadmin/v1/reasons?type=appeasementReason&id={id}
```

Get appeasement information

You can use the `getItemType` endpoint in the Admin API to get information about an appeasement, an appeasement refund or an appeasement comment item descriptor by issuing a `GET` command in the following format:

```
GET /ccadmin/v1/itemTypes/{id}
```

Note that the `id` can be `appeasement`, `appeasementRefund` or `appeasementComment`.

Update an existing appeasement

You can use the `updateItemType` endpoint in the Admin API to update an existing appeasement or an appeasement refund item type. You can also create custom properties at the appeasement or appeasement refund level with this endpoint. Use the following format:

```
PUT /ccadmin/v1/itemTypes/{itemTypeIdentifier}
```

Note that the `itemTypeIdentifier` can be `appeasement`, `appeasementRefund` or `appeasementComment`.

Create Comments

You can create a comment on an appeasement. The `comments` property at the appeasement level is a list of comments provided by the Agent with the Agent API, or an external system using the Admin API. Comments is a separate entity referenced by the appeasement. The appeasement level comments are a list of the references to different associated comments. Comments are created/updated with the appeasement payload using appeasement endpoints. Note that comments can never be deleted. There is no separate endpoint for comments.

`Comments` have the following properties for the API:

- `id` – The ID of the comment, which can identify a previous comments that should be updated, and should also be associated to the appeasement that is being updated.
- `comment` – The actual text of the comment, which is required to create comments.

- `agentId` – The ID of the Agent who saved the comment. This can also be the ID of an Agent that is provided by an external system. If this field is not provided in the payload, it defaults to `external`. If the comment is added using the Agent API, the `agentId` will automatically be the ID of the logged in Agent.
- `createDate` – The date that the comment was created.

The following is an example of a comment that was added to a request:

```
"comments": [
  {
    "agentId": " BobAFrette",
    "comment": "Customer asked to settle the appeasement amount to the same
        credit card he used to pay for the order.",
    "id": 100011,
    "creationDate": "2020-03-20T04:26:13.935Z"
  }
]
```

Use the Appeasement Webhook API

When you want to work with an external system to issue refund types, there are two webhooks that can help you configure your environment. The first webhook provides validation and the second webhook submits the appeasement.

To enable an Agent to select the payment tender type for the amount credited/refunded to the shopper, you must use an Server Side Extension (SSE) to build logic of what instruments are available.

Use the Appeasement Validation Webhook

The system performs validation, such as the appeasement amount cannot exceed the order total, but also allows you to create custom validations using a webhook. The validation webhook validates the appeasement request with a JSON response that contains a success and a failure `responseCode`. If an external system requires additional data, it must use the API to obtain the necessary data. Note that because validation is synchronous, it could possibly create a performance lag.

Once the appeasement has been validated, it is sent to the external system through the `AppeasementSubmitWebhook` event webhook. Any external system needs to settle the appeasement, and then update the appeasement using the update Admin API.

The `AppeasementValidationWebhook` validates the appeasement with an external system. The webhook payload contains the following:

- Current appeasement information
- Current site, profile and organization context
- Agent information, including name, role and email
- Complete order information if the appeasement is associated with an order
- Complete profile information if the appeasement is associated with a non-anonymous order or a profile
- All return request details of the current order
- The latest 50 appeasements associated with the shopper

The entirety of the order and the profile data is included, which is the same as the submit order and update profile webhooks. This webhook contains order data that is similar to the submit order webhook, with the return data corresponding to the return request webhook.

The following is an example of an `AppeasementValidationWebhook` request:


```
{
"appeasement": {
"reason": {
"readableDescription": "Order Arrived Late",
"id": "orderArrivedLate"
},
"agentId": "BobAFrette",
"notes": "The customer complained that the order arrived late, and
asked for a
 refund of shipping amount. Providing 15 USD.",
"comments": [],
"orderId": "o10411",
"lastModifiedDate": null,
"submittedTime": null,
"appeasementRefunds": [
{
"paymentGroupId": null,
"refundType": "externalRefund",
"amount": 0.01,
"id": "100004",
"state": "INCOMPLETE",
"currencyCode": "USD"
}
],
"submittedDate": "2020-06-23T17:02:55.502Z",
"type": {
"displayName": "Order",
"description": "Appeasement to be applied on a given order",
"id": "order"
},
"creationDate": "2020-06-23T17:02:55.519Z",
"profileId": "se-570031",
"appeasementAmount": 0,
"state": "INCOMPLETE",
"id": "app10004",
"originOfAppeasement": "agent"
},
"site": {
"name": "Commerce Cloud Site",
"id": "siteUS",
"url": "http://kkm00aqi.in.example.com:8080"
},
"agentInfo": {
"lastName": "Weber",
"firstName": "Damon",
"roles": [
{
"name": "CS Agent Supervisor",
```

```
"description": "CS Agent Supervisor Role",
"accessRights": [],
"id": "csAgentSupervisorRole",
"category": "Agent App"
}
],
"id": "service",
"email": null
},
"previousAppeasements": [
{
"reason": {
"readableDescription": "Order Arrived Late",
"id": "orderArrivedLate"
},
"agentId": "service",
"notes": "The customer complained that the order arrived late, and asked for
a
  refund of shipping amount. Providing 15 USD.",
"comments": [],
"orderId": "o10411",
"lastModifiedDate": "2020-06-23T13:52:40.748Z",
"submittedTime": null,
"appeasementRefunds": [
{
"paymentGroupId": null,
"refundType": "externalRefund",
"amount": 0.01,
"id": "100001",
"state": "INCOMPLETE",
"currencyCode": "USD"
}
],
"submittedDate": "2020-06-23T13:52:40.720Z",
"type": {
"displayName": "Order",
"description": "Appeasement to be applied on a given order.",
"id": "order"
},
"creationDate": "2020-06-23T13:52:40.748Z",
"profileId": "se-570031",
"appeasementAmount": 0,
"state": "SUBMITTED",
"id": "app10001",
"originOfAppeasement": "agent"
}
],
"profile": {
"lastPurchaseDate": "2020-06-23T13:50:49.686Z",
"dynamicPropertyMapLong": {},
"GDPRProfileP13nConsentDate": null,
"GDPRProfileP13n ConsentGranted": false,
"secondaryAddresses": {
"Work": {
"country": "US",
```

```
"lastName": "Anderson",
"types": [],
"address3": null,
"city": "Buffalo",
"address2": "",
"prefix": null,
"address1": "451 Brooks Ave",
"postalCode": "14201",
"companyName": null,
"county": null,
"suffix": null,
"firstName": "Kim",
"phoneNumber": "212-555-2150",
"item-id": null,
"repositoryId": "se-970031",
"faxNumber": null,
"middleName": null,
"state": "NY"
},
"Mom' s house": {
"country": "US",
"lastName": "Anderson",
"types": [],
"address3": null,
"city": "Dewitt",
"address2": null,
"prefix": null,
"address1": "41 Wexford Rd ",
"postalCode": "13214",
"companyName": null,
"county": null,
"suffix": null,
"firstName": "Dolores",
"phoneNumber": "212-555-4321",
"item-id": null,
"repositoryId": "se-140010",
"faxNumber": null,
"middleName": null,
"state": "NY"
},
"Home": {
"country": "US",
"lastName": "Anderson",
"types": [],
"address3": null,
"city": "Syracuse",
"address2": "",
"prefix": null,
"address1": "21 Cedar Ave",
"postalCode": "13202",
"companyName": null,
"county": null,
"suffix": null,
"firstName": "Kim",
"phoneNumber": "212-555-1977",
```

```
"item-id": null,
"repositoryId": "se-980031",
"faxNumber": null,
"middleName": null,
"state": "NY"
}
},
"shippingSurchargePriceList": {},
"firstPurchaseDate": "2020-06-23T13:50:49.686Z",
"profileType": null,
"loyaltyPrograms": [],
"lastPurchaseAmount": 80.99,
"registrationDate": "2020-06-23T12:36:03.000Z",
"sessionOrganization": null,
"lifetimeAOV": 80.99,
"id": "se-570031",
"derivedSalePriceList": null,
"homeAddress": {
"country": "US",
"lastName": null,
"types": [],
"address3": null,
"city": "Not available",
"address2": null,
"prefix": null,
"address1": "Not available",
"postalCode": "14201",
"companyName": null,
"county": null,
"suffix": null,
"firstName": null,
"phoneNumber": null,
"item-id": null,
"repositoryId": "se-960031",
"faxNumber": null,
"middleName": null,
"state": "Not available"
},
"daytimeTelephoneNumber": null,
"customerContactId": null,
"taxExempt": false,
"dynamicPropertyMapBigString": {},
"active": true,
"lastVisitDate": null,
"taxExemptionCode": null,
"previousVisitDate": null,
"version": 8,
"abandonedOrderCount": 0,
"firstName": "Kim",
"defaultCreditCard": {
"expirationYear": "2017",
"tokenExpiryDate": null,
"gatewayConfigId": null,
"expirationMonth": "1",
"creditCardType": "Visa",
```

```
"source": null,
"iin": null,
"token": null,
"cardProps": {},
"nameOnCard": null,
"creditCardNumber": "4539082039396288",
"tokenCreatedDate": "2020-06-23T12:36:03.388Z",
"cardSavedDate": null,
"billingAddress": {
"country": "US",
"lastName": "Anderson",
"types": [],
"address3": null,
"city": "Buffalo",
"address2": "",
"prefix": null,
"address1": "451 Brooks Ave",
"postalCode": "14201",
"companyName": null,
"county": null,
"suffix": null,
"firstName": "Kim",
"phoneNumber": "212-555-2150",
"item-id": null,
"repositoryId": "se-990031",
"faxNumber": null,
"middleName": null,
"state": "NY"
},
"id": "se-usercc110031",
"expirationDayOfMonth": null
},
"lifetimeCurrencyCode": "USD",
"derivedTaxExemptionCode": null,
"currentOrganization": null,
"secondaryOrganizations": [],
"shippingAddresses": [
{
"country": "US",
"lastName": "Anderson",
"types": [],
"address3": null,
"city": "Syracuse",
"address2": "",
"prefix": null,
"address1": "21 Cedar Ave",
"postalCode": "13202",
"companyName": null,
"county": null,
"suffix": null,
"firstName": "Kim",
"phoneNumber": "212-555-1977",
"item-id": null,
"repositoryId": "se-980031",
"faxNumber": null,
```

```
"middleName": null,
"state": "NY"
}
],
"derivedPriceListGroup": null,
"lastName": "Anderson",
"roles": [],
"numberOfOrders": 1,
"locale": "en",
"login": "kim@example.com",
"receiveEmailDate": null,
"sitePropertiesList": [
{
"site": {
"id": "siteUS"
},
"properties": {
"numberOfVisits": 0,
"GDPRProfileP13nConsentDate": null,
"GDPRProfileP13nConsentGranted": false,
"receiveEmail": "no",
"receiveEmailDate": null
}
}
],
"lifetimeSpend": 80.99,
"dynamicPropertyMapString": {},
"email": "kim@example.com",
"numberOfVisits": 0,
"siteProperties": {
"siteUS": {
"numberOfVisits": 0,
"GDPRProfileP13nConsentDate": null,
"GDPRProfileP13nConsentGranted": false,
"receiveEmail": "no",
"receiveEmailDate": null
}
},
"comments": [],
"receiveEmail": "no",
"priceListGroup": {
"isTaxIncluded": false,
"endDate": null,
"displayName": "Default Price Group",
"listPriceList": {},
"active": true,
"isPointsBased": false,
"locale": "en_US",
"basePriceListGroup": null,
"shippingSurchargePriceList": {},
"deleted": false,
"taxCalculationType": null,
"ancestorPriceListGroups": [],
"salePriceList": {},
"currency": {
```

```
"currencyType": null,
"symbol": "$",
"deleted": false,
"displayName": "US Dollar",
"fractionalDigits": 2,
"currencyCode": "USD",
"numericCode": "840"
},
"id": "defaultPriceGroup",
"includeAllProducts": true,
"startDate": null
},
"dateOfBirth": "1979-02-03T00:00:00.000Z",
"shippingAddress": {
"country": "US",
"lastName": "Anderson",
"types": [],
"address3": null,
"city": "Syracuse",
"address2": "",
"prefix": null,
"address1": "21 Cedar Ave",
"postalCode": "13202",
"companyName": null,
"county": null,
"suffix": null,
"firstName": "Kim",
"phoneNumber": "212-555-1977",
"item-id": null,
"repositoryId": "se-980031",
"faxNumber": null,
"middleName": null,
"state": "NY"
},
"firstVisitDate": null,
"middleName": null,
"lastActivity": null,
"billingAddress": {
"country": "US",
"lastName": "Anderson",
"types": [],
"address3": null,
"city": "Buffalo",
"address2": "",
"prefix": null,
"address1": "451 Brooks Ave",
"postalCode": "14201",
"companyName": null,
"county": null,
"suffix": null,
"firstName": "Kim",
"phoneNumber": "212-555-2150",
"item-id": null,
"repositoryId": "se-990031",
"faxNumber": null,
```

```
"middleName": null,
"state": "NY"
},
"dynamicPropertyMapDouble": {},
"derivedShippingSurchargePriceList": null,
"abandonedOrders": []
},
"order": {
"gwp": false,
"secondaryCurrencyCode": null,
"submittedDate": "2020-06-23T13:51:47.000Z",
"salesChannel": "default",
"configuratorId": null,
"uuid": "c0431f69-bb09-4d36-8d86-c75d041c6f49",
"organizationId": null,
"relationships": [
{
"paymentGroupId": "pg10413",
"amount": 80.99,
"relationshipType": "ORDERAMOUNTREMAINING",
"id": "r30390"
}
],
"exchangeRate": null,
"id": "o10411",
"state": "SUBMITTED",
"taxCalculated": true,
"combinedPriceInfos": null,
"commerceItems": [
{
"gwp": false,
"deactivationDate": null,
"returnedQuantity": 0,
"availabilityDate": null,
"billingProfileId": null,
"externalData": [],
"billingAccountId": null,
"preOrderQuantity": 0,
"assetKey": null,
"commerceItemId": "ci1000411",
"priceInfo": {
"discounted": false,
"amount": 49.99,
"secondaryCurrencyShippingSurcharge": 0,
"rawTotalPrice": 49.99,
"salePrice": 0,
"orderDiscountInfos": [],
"priceListId": "listPrices",
"itemDiscountInfos": [],
"quantityDiscounted": 0,
"amountIsFinal": false,
"onSale": false,
"shippingSurcharge": 0,
"discountable": true,
"currentPriceDetailsSorted": [
```

```
{
"secondaryCurrencyTaxAmount": 0,
"discounted": false,
"amount": 49.99,
"quantity": 1,
"configurationDiscountShare": 0,
"amountIsFinal": false,
"range": {
"lowBound": 0,
"highBound": 0,
"size": 1
},
"tax": 4,
"orderDiscountShare": 0,
"detailedUnitPrice": 49.99,
"currencyCode": "USD"
}
],
"currencyCode": "USD",
"listPrice": 49.99
},
"catalogId": null,
"externalRecurringChargeDetails": null,
"externalPriceDetails": null,
"actionCode": null,
"id": "ci1000411",
"state": "INITIAL",
"serviceId": null,
"locationInventoryInfoMap": {},
"serviceAccountId": null,
"quantity": 1,
"pointOfNoRevision": false,
"productId": "Product_21Cxi",
"parentAssetKey": null,
"externalId": null,
"originalCommerceItemId": null,
"rootAssetKey": null,
"transactionDate": null,
"catalogRefId": "Sku_21Dxy",
"customerAccountId": null,
"recurringChargePriceInfo": null,
"lineAttributes": [],
"catalogKey": null,
"productDisplayName": "Dora the Explorer - Season 1",
"shopperInput": {},
"activationDate": null,
"asset": false,
"backOrderQuantity": 0
}
],
"shippingGroups": [
{
"shippingMethod": "standardShippingMethod",
"description": "sg30411",
"submittedDate": null,
```

```
"priceInfo": {
"secondaryCurrencyTaxAmount": 0,
"discounted": false,
"shippingTax": 2,
"secondaryCurrencyShippingAmount": 0,
"amount": 25,
"rawShipping": 25,
"amountIsFinal": false,
"currencyCode": "USD"
},
"shipOnDate": null,
"actualShipDate": null,
"specialInstructions": {},
"shippingAddress": {
"country": "US",
"lastName": "Anderson",
"address3": null,
"city": "Syracuse",
"address2": null,
"prefix": null,
"address1": "21 Cedar Ave",
"companyName": null,
"jobTitle": null,
"postalCode": "13202",
"county": null,
"ownerId": null,
"suffix": null,
"firstName": "Kim",
"phoneNumber": "212-555-1977",
"faxNumber": null,
"middleName": null,
"state": "NY",
"email": "kim@example.com"
},
"commerceItemRelationships": [
{
"availablePickupDate": null,
"inventoryLocationId": null,
"amount": 0,
"quantity": 1,
"pointOfNoRevision": false,
"relationshipType": "SHIPPINGQUANTITY",
"returnedQuantity": 0,
"preferredPickupDate": null,
"range": {
"lowBound": 0,
"highBound": 0,
"size": 1
},
"commerceItemExternalId": null,
"commerceItemId": "ci1000411",
"state": "INITIAL",
"id": "r30388"
}
],
```

```
"state": "INITIAL",
"id": "sg30411",
"stateDetail": null,
"trackingNumber": null,
"handlingInstructions": [],
"shippingGroupClassType": "hardgoodShippingGroup"
}
],
"freezeDate": null,
"taxExempt": false,
"profile": {
"lastName": "Anderson",
"firstName": "Kim",
"loyaltyPrograms": [],
"shippingAddress": {
"country": "US",
"lastName": "Anderson",
"types": [],
"address3": null,
"city": "Syracuse",
"address2": "",
"prefix": null,
"address1": "21 Cedar Ave",
"postalCode": "13202",
"jobTitle": null,
"companyName": null,
"county": null,
"ownerId": null,
"suffix": null,
"version": 2,
"firstName": "Kim",
"phoneNumber": "212-555-1977",
"repositoryId": "se-980031",
"faxNumber": null,
"middleName": null,
"state": "NY",
"id": "se-980031"
},
"middleName": null,
"login": "kim@example.com",
"parentOrganization": null,
"email": "kim@example.com"
},
"queuedOrderSubmitData": null,
"cartName": "o10411",
"paymentInitiatedEmailSent": false,
"payShippingInSecondaryCurrency": false,
"shippingGroupCount": 1,
"taxExemptionCode": null,
"createdByOrderId": null,
"orderAction": "order",
"submissionErrorMessages": [],
"profileId": "se-570031",
"activeQuoteOrderId": null,
"approverIds": [],
```

```
"agentId": "service",
"lastModifiedTime": 1592920307148,
"priceGroupId": "defaultPriceGroup",
"creationTime": 1592920249000,
"sourceSystem": "Cloud Commerce",
"gwpMarkers": [],
"locale": "en",
"paymentGroups": [
{
"expirationYear": "2035",
"amountAuthorized": 80.99,
"amount": 80.99,
"gatewayName": "CS-A",
"paymentProps": {
"uiIntervention": "sop"
},
"expirationMonth": "12",
"submittedDate": "2020-06-23T13:51:46.000Z",
"authorizationStatus": [
{
"authorizationDecision": "ACCEPT",
"transactionUuid": "ebb2112ff7da441d97ac67b4f9c35d90",
"amount": 80.99,
"statusProps": {
"req_card_type": "001",
"auth_cv_result": "M",
"auth_response": "100",
"req_transaction_type": "authorization,create_payment_token",
"req_locale": "en",
"req_payment_method": "card",
"decision_rmsg": "Test Review",
"auth_trans_ref_no": "77997054RWTFHZC4",
"auth_time": "2020-06-23T13:51:46.732Z",
"auth_code": "888888"
},
"transactionSuccess": true,
"errorMessage": "Request was processed successfully.",
"currency": "USD",
"reasonCode": "100",
"transactionId": "fj9o00reciqc7afedjlrohess2"
}
],
"IIN": null,
"token": "9997000108950573",
"paymentGroupClassType": "tokenizedCreditCard",
"creditCardNumber": "1111",
"paymentMethod": "tokenizedCreditCard",
"state": "AUTHORIZED",
"id": "pg10413",
"billingAddress": {
"country": "US",
"lastName": "Anderson",
"address3": null,
"city": "Syracuse",
"address2": null,
```

```
"prefix": null,
"address1": "21 Cedar Ave",
"companyName": null,
"jobTitle": null,
"postalCode": "13202",
"county": null,
"ownerId": null,
"suffix": null,
"firstName": "Kim",
"phoneNumber": "212-555-1977",
"faxNumber": null,
"middleName": null,
"state": "NY",
"email": "kim@example.com"
},
"debitStatus": [],
"currencyCode": "USD"
}
],
"payTaxInSecondaryCurrency": false,
"priceInfo": {
"secondaryCurrencyTaxAmount": 0,
"discounted": false,
"secondaryCurrencyShippingAmount": 0,
"amount": 49.99,
"secondaryCurrencyTotal": 0,
"manualAdjustmentTotal": 0,
"discountAmount": 0,
"tax": 6,
"rawSubtotal": 49.99,
"total": 80.99,
"shipping": 25,
"primaryCurrencyTotal": 49.99,
"amountIsFinal": false,
"currencyCode": "USD"
},
"submissionProgress": [],
"catalogId": null,
"totalCommerceItemCount": 1,
"externalContext": false,
"cancelReason": null,
"quoteInfo": null,
"taxPriceInfo": {
"secondaryCurrencyTaxAmount": 0,
"discounted": false,
"valueAddedTax": 0,
"amount": 6,
"countyTax": 3,
"isTaxIncluded": false,
"miscTax": 0,
"districtTax": 0,
"stateTax": 3,
"miscTaxInfo": {},
"countryTax": 0,
"cityTax": 0,
```

```
"amountIsFinal": false,
"currencyCode": "USD"
},
"lastModifiedDate": "2020-06-23T13:51:47.148Z",
"allowAlternateCurrency": false,
"approvalSystemMessages": [],
"approverMessages": [],
"paymentGroupCount": 1,
"submissionErrorCodes": [],
"recurringChargePriceInfo": null,
"organization": null,
"siteId": "siteUS"
}
}
```

A typical response from this webhook would be:

```
{
  "responseCode": "8001"
}
```

External systems should respond with the response code 8001 when the validation is successful and an appeasement can be provided. Should the validation fail, the response code returns 8002, with a reason recorded in the `reasonForValidationFailure` field.

Use the Appeasement Submit Webhook

The `AppeasementSubmitWebhook` event webhook carries the required appeasement payload to external systems once the appeasement has been validated and saved. Its payload contains only the current appeasement information.

A typical request may be similar to the following:

```
{
"appeasement": {
"agentId": " BobAFrette",
"notes": "Customer complained that the order has arrived damaged, and asked
for a
   refund. Providing 15 USD.",
"orderId": "o11038",
"damagedLineItemId": "ci1235471",
"lastModifiedDate": "2020-07-08T10:34:09.698Z",
"appeasementRefunds": [
{
"refundType": "externalRefund",
"amount": 9,
"transactionNumber": "Refund Transaction Number",
"id": "100269",
"state": "INCOMPLETE",
"paymentGroupId": "pg11017",
"customerPreferredCardNumber": 0,
"refundDetails": "Refund Details",
"refundNotes": "Refund Notes",
"currencyCode": "USD",
"dateOfRefundProcess": "2017-02-08T18:30:00.000Z",
```

```
"refundSupportedTypes": true
}
],
"creationDate": "2020-07-08T10:34:09.698Z",
"profileId": "120000",
"state": "SUBMITTED",
"id": "app10438",
"originOfAppeasement": "agent",
"type": {
"displayName": "Order",
"description": "Appeasement to be applied on the order.",
"id": "order"
},
"reason": {
"readableDescription": "Item Arrived Damaged",
"id": "itemArrivedDamaged"
},
"comments": [
{
"agentId": " BobAFrette",
"comment": "Appeasement to be settled as a priority. Valued customer.",
"id": "100008",
"creationDate": "2020-07-08T10:34:09.687Z"
}
],
"submittedDate": "2020-07-08T10:34:09.688Z",
"damageDetails": "Item is Partially Damaged",
"customerType": true,
"appeasementAmount": 0,
"dateOfRequest": "2017-02-08T18:30:00.000Z"
}
}
```


Understand appeasement validation

By default the following validations are made when creating an order-based appeasement:

- The total appeasement amount, which includes the current appeasement refunds and the already completed appeasement refunds on the order, and the total refund amount from the already completed return request does not exceed the order total
- The refund amount for each refund type does not exceed the amount of the associated payment group
- The order on which the appeasement has been created should be in the FULFILLED state or the SUBMITTED state. The remorse period for the order should be over.

Create custom validations for appeasements

You can create custom validations for order level appeasements by setting the `isOrderRequired` flag to `true`. For example, you might want to create custom validations such as an Agent appeasement limit where the Agent cannot give more than a specified amount regardless of the order or profile. Or, you might want to create an order limit validation, where the Agent might not be able to give more than 15% of the order total. If you were to issue a shipping limit or a shipping total limit, the Agent

could not issue anything beyond the shipping total, or a percentage of the shipping total.

Creating an appeasement total limit, which takes all the given appeasement totals from the appeasement history and sums them up, prohibits a particular profile from ever receiving an appeasement beyond this limit.

For example, if you have set the `isOrderRequired` flag to `false`, your system will not allow order-based appeasements. However, if you want to create a single order-based appeasement, bypassing your current setting, you would create a custom property to store the order ID. This would prevent the default `orderId` from triggering the `isOrderRequired` flag and allow you to create an order-based appeasement.

You could configure your system to skip order-based validations by performing the following steps:

- Create an `appeasementType` where the `isOrderRequiredFlag` is set to `false`.
- Create a custom property at the appeasement level that holds the `orderId`, such as `orderId2`. Then, when you create the appeasement, do not use the default `orderId` property but the new custom `orderId2` field instead.
- Ensure that the appeasement refund type is set to `external`, and that all of the payments are handled externally.

When you create custom properties, set the properties for which validation occurs by default to a system-allowed value:

- `orderId` - An order that must exist in the system.
- `state` – The state for the given appeasement should be `FULFILLED`.
- `currencyCode` – This should match the currency of the order.
- `paymentGroup` – When creating external refunds, it is best to use the webhook to validate the refund. When you use the webhook, you do not need to provide a `paymentGroupId` value in the `appeasementRefund` input.
- `amount` – The appeasement cannot exceed the total of the order that corresponds to the `orderId`.
- `profile` – You can choose not to pass a `profileId` and use a different custom property instead that skips profile ID validation for order-based appeasements.

Validate APIs

By default, validations are performed on these fields when you create, update and submit appeasements.

Property	Description
reason	The reason property indicates the reason for the appeasement. Possible reasons are: orderArrivedLate orderArriveDamaged itemArrivedLate itemArrivedDamaged didNotLikeItem incorrectItem goodwillGesture productComplaint
state	Appeasements can have the following states: INCOMPLETE – An appeasement remains in this state until submitted. SUBMITTED – An appeasement reaches this state once its validation is successful. COMPLETE – An appeasement is marked COMPLETE by an external system once all of the refunds are settled. Note that the state property can only be sent with the Admin API.
orderId	If the appeasement type has a set isOrderRequired flag, the appeasement must have an order ID of an order in the FULLFILLED state, or a submitted order post its remorse period.
profileId	The profile ID should be a registered customer. If the appeasement type does not have an isOrderRequired flag set, the profileId is mandatory in the Agent API.
pymentGroupId	This ID must be provided to identify the applicable refund for each card.
currencyCode	This property allows you to work with multi-currency orders.
type	The type property contains an order. If you have created a custom appeasement type with the Admin API, you can also provide its ID.
refundType	The externalRefund is used for externally settled refunds. When using the Agent API, use the externalRefund for the refund type when invoking the validating webhook. By default, the system updates the following refund instruments: creditCard storeCredit tokenizedCreditCard onlinePaymentGroup physicalGiftCard customCurrencyPaymentGroup

Property	Description
`amount`	All default appeasement refund objects should have non-zero amount values.

To create additional validation, define rules/validation externally and have the appeasement request validated by a validation webhook.

Configure email

Each appeasement transaction is given a unique appeasement and authorization ID. The system also ensures that the credit is passed to downstream payment systems. Once the appeasement has been submitted and confirmed, the system triggers an email. Note that email notification occurs only when using the Agent API, and no email is triggered when using the Admin API. Emails will be issued when either the order ID, the profile ID or both are available.

If an order ID is provided and, if the order has been placed by a registered user, the email ID is picked from the profile. However, if the order has been a placed by an anonymous user, the system uses the email ID obtained from the shipping address.

Email templates contain the appeasement ID and level, the amount of the appeasement, payment mode and the payment mode identification number, any notes added, the submitted date and who the appeasement was submitted by, as well as any custom properties, order properties, profile properties and site properties that have been configured.

Perform returns with appeasements

Appeasements and returns have an impact on one another. Refund amounts are considered when generating an appeasement, and similarly, when an item is returned, the calculations for the refund amount considers all appeasement amounts that are specific to that order. For example, if there is an order with $10 and an appeasement is given for $2, a maximum of $8 can be refunded through return requests.

When the $2 has been removed from the order, the customer will not know what happened. You must add the removal to the widget to allow your customers to see the adjustments that have occurred due to the appeasements.

Displaying Appeasement adjustments

You can use the following properties to customize your template so that various details are property displayed.

Appeasement adjustment shares are captured in the `appeasementRefundAdjustment` and `secondaryCurrencyAppeasementRefundAdjustment` properties of the `refundInfo` object. These properties are available by default in the Storefront and Agent APIs. You can configure a widget to display these properties by using the appropriate endpoints. By default in the Storefont and Agent interfaces, the refund details of a return request are also captured in the `refundInfo` object in the returns view model, the `return.js` file.

If you are using the default Agent return initiate refund page, the administration interface identifies any changes you have made in the appeasement adjustments. Based on these adjustments, the **initiate refund** button is disabled and the **apply** button is enabled so that the latest appeasement adjustment share details are reflected in the administration interface.

Use Storefront endpoints

When working with returns that contain appeasements, you can use the following Storefront endpoints:

Endpoint	Description
`listReturnRequests`	When you issue a `GET` command to `/orders/{id}/returnRequests`, you get a list of return requests.
`createReturnRequest`	When you issue a `POST` request, the values are populated based on the current appeasement requests and return request of the order.
`getReturnRequest`	When you issue a `GET` request to `/returnRequests/{id}`, the values are recalculated based on the appeasements operations, such as new appeasements that are completed on the same order or the removal of an existing completed appeasement except for the completed state return requests.
`calculateRefund`	When you issue a `POST` request to `/returnRequests/calculateRefund`, the values are populated based on the current appeasement requests and return requests of the order.
`initiateReturn` and `validateReturns`	A `POST` request ensures that the `appeasementRefundAdjustment` and `seoncdaryAppeasementRefundAdjustment` properties are populated with values other than the default 0 value.

Use Admin endpoints

The `appeasementRefundAdjustment` and the `secondaryAppeasementRefundAdjustment` properties are added to the returns payload to capture the share of the appeasement requests. You can use the following Admin endpoints to show the property details:

Endpoint	Description
`getReturnRequest/returnRequests{id}`	This endpoint returns the available data. The values for the `appeasementRefundAdjustment` and `secondaryAppeasementRefundAdjustment` properties are not recalculated.
`updateReturnRequest/returnRequests/{id}`	This endpoint will show the available data, but does not update the `appeasementRefundAdjustment` and `secondaryRefundAdjustment` properties as these properties are populated by the system.

Use Agent endpoints

The Agent API uses the following endpoints to work with returns:

Endpoint	Description
`initiateReturn/returnRequests`	When you issue a `POST` request, this endpoint populates the `appeasementRefundAdjustment` and `secondaryAppeasementRefundAdjustment` properties to values based on the return amount. If you use the `createReturnRequest` and `calculateRefundAmounts` endpoints, the values of the `appeasementRefundAdjustment` and `secondaryAppeasementRefundAdjustment` properties are populated based on the current appeasement requests and return requests of the order
`searchReturns/returnRequests`	When you these issue a `GET` request, this endpoint returns the data available; however, there is no recalculation of the `appeasementRefundAdjustment` and `secondaryAppeasementRefundAdjustment` properties.
`getReturnRequest/returnRequests/{id}`	When you issue a `GET` command, the values of the `appeasementRefundAdjustment` and `secondaryAppeasementRefundAdjustment` will be recalculated based on the appeasements operations.
`updateReturnRequest/returnRequests/{id}`	When you issue a `PUT` command, the `appeasementRefundAdjustment` and `secondaryAppeasementRefundAdjustment` property values will be recalculated based on the appeasements operations. This could be operations like creating new appeasements that are completed on the same order or removing an existing completed appeasement yet maintain the completed state return requests for the operation's `adjustRefundAmounts` and `initiateRefund` totals.
`calculateRemainingRefund / returnRequests/{id}/ calculateRemainingRefund`	When you issue a `POST` request, the values of the `appeasementRefundAdjustment` and `secondaryAppeasementRefundAdjustment` properties is recalculated based on the appeasements operations such as new appeasements that are completed on the same order.
`receiveReturnRequest /returnRequests/ {id}/receive`	When you issue a `PUT` request, this endpoint returns the available data . There is no recalculation of the `appeasementRefundAdjustment` and `secondaryAppeasementRefundAdjustment` properties.

Note that by default, Commerce does not display the `appeasementRefundAdjustment` and `secondaryAppeasementRefundAdjustment` in the administration interface.

For an order appeasement type, use a refund mode that corresponds to a payment method that was used in the order. However, payments that are not settled will not be available for the internal refund process, so for these payment groups, the system assumes an `externalRefund` option. Validations for other refund modes generated from an external refund are done using external refund validation webhooks.

Note that orders that have multiple appeasements can use multiple currencies. Additionally, you can perform refunds with multiple currencies within a single appeasements.

Understand refund calculations with appeasements

By default, an appeasement's refund type corresponds to the default payment types that are available for the appeased order. Default internal refund types include:

- `creditCard` – This refund type includes CyberSource, Chase Credit Card and `genericCreditCard` and is the accesses using the default refund API
- `physicalGiftCard` – which includes `genericGiftCard`
- `customCurrencyPaymentGroup` – This includes `loyaltyPoints`
- `onlinePaymentGroup` – which includes PayPal
- `storeCredit` – which uses the Generic payment webhook for issuing the refund
- `tokenizedCreditCard` – This includes the CyberSource payment gateway

Default refund types for external systems include:

- `payU Latem`
- `instore`
- `cash`
- `invoice`
- `coupon` – no refund amount is processed when using this refund type

By default, refund calculations are made on the maximum refund amount (the allowed limit based on existing refunds on the payment type), the order total and previous appeasements made on the order. When you create an external appeasement refund, the appeasement is calculated separately.

Work with External Systems

Appeasements refunds can be customized to use an external appeasement refund mode, which is then handled by an external system and then returned to the Commerce system. External appeasements can be issued using credit cards, gift cards or other methods. You can also issue non-monetary appeasements, such as coupons and vouchers. Note that multiple external appeasement modes are available, however you cannot combine monetary and non-monetary appeasements.

For orders that are placed with cash, invoices, and gift cards, appeasements must be performed using an external payment because Commerce cannot settle them internally. You can also configure external payment to allow gift vouchers or promotion codes that are not part of your order payment system. When you create an external appeasements, you need to configure the Submit Appeasement Event Webhook in the Admin API so that you will receive notification when an appeasement is successfully created.

Once appeasements are validated, they are sent to the external system through the Appeasement Submit Event webhook. You can import appeasements from an external appeasement system using the Admin API.

Update external appeasements

You must update the Commerce system whenever you generate external appeasements. Once the appeasement has been settled in the external system, the state of the appeasement and appeasement refund should be marked in the system as `COMPLETED`. This identifies the appeasement amount in the total amount validation. If the external appeasement is not marked `COMPLETED`, it will not be considered for subsequent returns and refunds. You can mark the appeasement with the update appeasement endpoint by issuing a PUT command in the following format:

```
/ccadmin/v1/appeasements/{appeasementId}
```

7

Customize the Oracle Commerce Agent Console

You can perform a number of customizations and actions by extending the Oracle Commerce agent widgets.

Assumptions

Before you create widgets for your agent environment, you should be familiar with specific technology.

You should be comfortable with site administration, have experience with the following technologies and have reviewed the necessary documentation:

- JavaScript
- Data binding using Knockout
- Bootstrap
- Standard CSS and CSS Less
- MVVM Architecture
- Familiarity with Developing Storefront Classic Widgets for Oracle Commerce, which provides a detailed overview of how widgets work in Oracle Commerce
- Familiarity with the Oracle Commerce agent console

About extensions

Oracle Commerce agent console provides tools that you can use to extend the capabilities of your system.

The tools consist of the following:

- Widgets that allow you to extend the functionality of your agent's environment by accessing features that are not exposed by default. This guide focuses on describing agent console-specific widgets. For general information on other widgets, see Developing Storefront Classic Widgets for Oracle Commerce.
- An extensive set of REST APIs allows external applications to make calls into the Oracle Commerce server. This guide focuses on agent-specific APIs.

Work with widgets

Widgets allow you to display content to your agents or to execute specific functions. Widgets are comprised of templates, JavaScript, CSS, locale resources and images. For detailed information on widgets and extensions, refer to the Developing Storefront Classic Widgets for Oracle Commerce guide.

When you customize widgets, you create functionality that appears on your website pages. These widgets allow you to add pre-made and pre-configured snippets of text or information to your web pages without having to recreate them for each page. Widgets also allow you to

extend the functionality of your storefront by communicating with the Oracle Commerce Agent server to access a variety of features.

You can use agent-specific widgets to create a customized agent environment. Widgets are comprised of a set of resource and source files. Using these files, as well as auxiliary files that contain information such as meta-data, you can customize an agent's environment.

The agent environment uses agent-specific widgets, as well as Oracle Cloud Commerce storefront widgets. There are separate instances for agent and storefront widgets. Note that these instances differ in the template, and sometimes the configuration properties, they use. If you delete an agent widget instance, which uses a different template than the Display template, you will be unable to recreate the widget.

Important: Please note that Oracle Commerce Agent Console, unlike Oracle Commerce Storefront, does not support application JS, global widgets, or global elements. Any element used in default widgets should be downloaded and customized.

Tasks done by widgets

The following are some general tasks that can be performed by widgets:

- Agent Navigation - Enables an agent to navigate throughout the console to review a variety of data. Additionally, you can rename fields and buttons, and hide or display certain fields or navigational items.
- Search - Allows an agent to search for customers, orders or returns.
- Create Orders - You can display navigation, customer cart dialog, shipping details, address books, promotions, loyalty details, scheduled orders and product details, in addition to other displays.
- Order Details - You can create widgets to display order details for account orders, return and exchange orders, quotes and order approvals. You can also customize carts to show subtotals and logic for payment and pricing.
- Customer Details - You can create customized widgets to display account addresses and details, customer profile information, order histories, purchase lists, account contacts and other profile information.
- Returns and Exchanges - You can create widgets that display or create return requests and refunds, as well as process returns. You can also create widgets that create exchanges.

This documentation provides the following information for each widget:

- Description- A description of the widget.
- Widget Name - The code name of the widget. The name of the widget is identified in the `widget.json` file. For example:

```
"javascript": "agent-order-search",
"widgetFamily": "agentOrderSearch",
"widgetType": "agentOrderSearch"
```

- Display Name - The widget's name as displayed in the Design page. The display name can also be found in the `widget.json` file:

  ```
  "name": "Order Search - Agent"
  ```

- Supported Page Type - The types of pages that the widget can be applied to. This is also identified in the `widget.json` file:

  ```
  "pageTypes": ["agentOrderSearch"],
  ```

  For information on page types, refer to Define stack meta-data in stack.json.

- Layouts - The page layouts that this widget is associated with. A widget's associated layouts are identified on the Layout tab of the Design page. For detailed information on layouts, see Create Page Layouts that Support Different Types of Shoppers.
- Elements - The elements used by this widget. Each element represents a single piece of the structure of the widget. Elements can be configured as drag-and-drop sub-components, which allow you to control their location on a page's layout. Elements are added to the widget's `display.template` file and can be reviewed using the Design tab code view.
  For detailed information on elements, see Fragment a Widget into Elements.

For information on Storefront widgets refer to Appendix: Layout Widgets and Elements.

Access widgets

You can review Oracle Commerce default widgets by opening the Design page on your administration server. This page displays both the layouts that are used in the agent console, but also the components, such as widgets, that are available. Click the Component tab to see the list of widgets.

Each widget displays the associated page layouts. By selecting the page layouts, you can customize the pages by adding or removing widgets. For information on working with widgets and layouts, see Appendix: Layout Widgets and Elements.

Note that widgets that are specific to the agent console contain the word "Agent". The agent console also uses a number of widgets that are also used by the Storefront.

Extend functionality with widgets

You can customize widgets by downloading the widget source code and making modifications to the JavaScript and HTML files. For example, you could add custom JavaScript to an existing widget. Once you have made all of your changes, you then upload the widget by adding it to an extension as described in Understand Extension Features.

Important: When creating or renaming a widget, ensure that the name is less than 50 characters. Errors will occur if the widget name is greater than 50 characters.

You can also extend a widget's JavaScript by using the JavaScript Code Layering feature that allows you to layer custom JavaScript on top of the widget. For information on the JavaScript Code Layering feature, see Use the JavaScript Code Layering User Interface feature.

Note: For a widget's JavaScript code to be editable, ensure that the `jsEditable` flag in the `widget.json` file is set to true. By default, the flag is set to `false`, indicating that the JavaScript associated with the widget cannot be edited.

When you upgrade or modify a widget, you must remove any existing instances of it from the page layouts and replace it with the new widget.

Upgrade widgets

Widgets that have been deployed are not automatically updated when a newer version of the widget is released. This allows you to customize widgets without the fear of them being overwritten. To update a widget, you must remove the old widget from all of your page layouts and replace them with the new widget. Additionally, you must recreate templates or style sheets that you have customized for the new widget.

For detailed information on how to upgrade widgets, see Customize your store layouts.

Migrate widgets with multiple templates

Some widgets use more than one template. If you migrate your widgets and load the latest version of the widget, the agent-specific template is not available for newly created widget instances. To update the widget with the new version, you must ensure that the widget's new template file is recognized by the layout.

To do this, perform the following steps:

1. Open the widget in the administrative interface using the **Settings** icon.
2. Open the About tab. Click the **Go to widget code** button.
3. Click the **Download Source** button.
4. Unzip the file that you downloaded.
5. Open the `/widget/widgetName/layouts/layoutThatContainsTheLayout` file and copy the contents of the newer version of the `widget-layoutAndVersion.template` file.
   For example, you might copy the `widget-orderDetailsDefaultLayout_v3.template` file.
6. Return to the administrative interface from where you downloaded the source. In the **Template** tab, replace the existing code with the code you have copied from the new version of the `widget.template` file.
7. Click **Save**.
8. Ensure that you update the associated layouts.

Migrate widget with multiple LESS files

If you have recently migrated, your widgets will be listed in the page layout components. However, if you use a widget in a page layout region, and then create a new widget instance, the widget instance will be created with the default template, which won't be applicable to the page layout. To update these widgets, perform the following steps:

1. Open the widget in the administrative interface using the **Settings** icon.
2. Open the About tab. Click the **Go to widget code** button.
3. Click the **Download Source** button.
4. Unzip the file that you downloaded.
5. Open the `/widget/widgetName/less/` file and copy the contents of the newer version of the `widget.less` file.

For example, you might copy the `widget-agentAccountOrderDetailsInst_v4.less` file.

6. Return to the administrative interface from where you downloaded the source for the widget. In the **Style Sheet** tab, replace the code with the code you have copied from the new version of the LESS file.
7. Click **Save**.
8. Ensure that you update the associated layouts.

Recreate shared widgets after deletion

Some widgets also have corresponding storefront widgets. These widgets, although designed to perform similar tasks, use different templates and, sometimes, configuration properties. If an agent-specific widget is deleted, and the agent widget uses a different template than the storefront widget, you will not be able to recreate the widget from the administrative interface.

To rectify this, download the agent-specific widget and recreate the template for the instance.

Use the Agent REST API

Oracle Commerce uses REST APIs that consist of several sets of endpoints, which allow you to perform storefront, administrative and agent tasks.

These include Store, Admin, Social Wish List and Agent APIs.

The Agent API endpoints provide access to agent-specific functionality on the administration server. These endpoints can be used in conjunction with agent-specific widgets, for example to pass a response filter key in a REST call that was made from a widget.

Note that there are many similar endpoints that exist in each API. For example, each set of APIs may have endpoints for working with customers, although each endpoint differs in the functionality that they perform.

You can access a `ccdebug` REST client on your administration server in your test environment. This client is available at the following URL:

```
http://<admin-server-hostname>/ccebug
```

Each API is available only in certain environments. The Agent API is available only on administration servers. You can find information on endpoints in the REST API documentation, which is available through the Oracle Help Center at the following URL:

```
http://docs.oracle.com/cloud/latest/commercecs_gs/CX0CC/
```

Note that the documentation on the Oracle Help Center reflects the most recent version of Oracle Commerce. If you are using an earlier version of Commerce, the API documentation on the Oracle Help Center may include endpoints that are not available on your version.

For additional information on working with REST APIs, see Use the REST APIs.

Find additional resources

You can find additional information on working with widgets in the following documents:

- Developing Storefront Classic Widgets for Oracle Commerce - Describes how to work with and implement widgets in your environment. This document contains many examples of how to implement and customize widgets.
- Extending Oracle Commerce - Describes various task-specific widgets, as well as how to use REST APIs for customization.
- Layout Widgets and Elements - Provides information on Storefront-specific widgets.

8

Work with Navigation Widgets

Oracle Commerce provides default widgets that provide navigation.

This allows you to customize the way that agents navigate around the application.

Note: While navigation can be customized for the agent, the Log In page cannot be customized.

Customize agent navigation

This widget allows you to customize the navigation bar at the top of the page.

This widget is specifically for navigation between agent pages.

Widget Name: `agentNavigation`

Display Name: `Navigation - Agent`

Supported Page Types:

- `agentAccountContacts`
- `agentAccountDetails`
- `agentAddressBook`
- `agentCheckout`
- `agentCreateExchange`
- `agentCreateReturn`
- `agentCustomerDetails`
- `agentCustomerSearch`
- `agentHome`
- `agentMultiShipCheckout`
- `agentOrderDetails`
- `agentOrderHistory`
- `agentOrderSearch`
- `agentOrdersPendingApproval`
- `agentProcessReturns`
- `agentPurchaseList`
- `agentRefunds`
- `agentRegtistrationRequestDetail`
- `agentReturnSearch`

- agentScheduledOrder
- agentSelfRegistrationPage
- agentViewReturnRequest

Layouts:All

Elements: None

Note: The Customer Profile page, and the Registration Request page, do not contain a back navigation button. To provide a button, you must create a custom property on the navigation widget.

Customize guided navigation

This widget presents an overlayed guided navigation to the agent.

Widget Name: overlayedGuidedNavigation

Display Name: Overlayed Guided Navigation

Supported Page Types: category, searchResults

Layouts:

- Collection Layout – Agent
- No Search Results Layout – Agent
- Search Results Layout - Agent

Elements: None

Customize the agent dashboard

This widget displays the agent dashboard.

This widget allows you to update the KPT/Graphics, announcements, quick links, recent orders and return requests, as well as pending actions. This widget displays updated information, information on recent customers and a quick search for customers and orders.

Widget Name: agentDashboard

Display Name: Dashboard - Agent

Supported Page Types: agentHomePageType

Layouts: Home Layout - Agent

Elements: To understand how these elements are used, refer to the widget's code view:

- agent-pending-actions
- announcements
- ordersChart
- price-group-drop-down

- `quick-links`
- `recent-orders`
- `returnRequestsChart`

Present errors

This widget presents errors if it encounters an invalid `siteId`, `organizationId` or `profileId`.

Migrate widgets

If you have recently migrated and this widget may be visible in your widget list. However, the widget is using the default template file, which may not be available to the page layout.

Widget Name: `agentError`

Display Name: `Page Not Found Widget - Agent`

Supported Page Types: All

Layouts:

- Error Layout - Agent

Elements: None

9

Configure Search Widgets

The widgets described in this section allow you to customize agent searches for customers, returns and orders. These widgets also provide you with the following functionality:

- The ability to determine which search attributes are displayed.
- The ability to rename the labels of search fields and results tables.
- You can rearrange sections or fields to customize the search page.
- You can choose to display only relevant columns for search results.
- You may add other widgets or elements within the search page.

Search for a customer

The Customer Search - Agent widget allows you to customize how an agent searches for a customer.

It holds criteria needed to search through customer, account and custom profile level attributes. This widget also displays the search button that the agent uses to initiate the search.

Widget Name: `agentCustomerSearch`

Display Name: Customer Search - Agent

Supported Page Types: `agentCustomerSearchPageType`

Layouts: Customer Search Layout - Agent

Elements: This widget uses the `account-search` element. To understand how this element is used, refer to the widget's code view.

Search for returns

The Return Search - Agent widget allows you to configure what an agent sees when searching for returns.

It holds all of the applicable criteria needed to search through return request, account and custom profile level attributes. This widget also provides a search button that allows the agent to initiate the search, while displaying both navigation and cancel elements.

Widget Name: `agentReturnSearch`

Display Name: Return Search - Agent

Supported Page Types: `agentReturnSearch` page

Layouts: Returns Search Layout - Agent

Elements: This widget uses the `account-search` element. To understand how this element is used, refer to the widget's code view.

Search for orders

The Order Search - Agent widget allows you to customize order searches.

It holds all of the applicable criteria needed to search through return request, account and custom profile level attributes, as well as advanced search attributes, such as shipping or billing addresses. This widget also provides a search button that allows the agent to initiate the search, while displaying both navigation and cancel elements.

Widget Name: `agentOrderSearch`

Display Name: Order Search - Agent

Supported Page Types: `agentOrderSearch` page

Layouts: Order Search Layout - Agent

Elements: This widget uses the following elements. To understand how these elements are used, refer to the widget's code view:

- `account-search`
- `product-search`

Find search results

The No Search Results - Agent widget is displayed to the agent when no search results are found.

Widget Name: `noSearchResults`

Display Name: No Search Results- Agent

Supported Page Types: `noSearchResults`

Layouts: No Search Results Layout - Agent

Elements: None

Customize search pages

Agents spend a great deal of time searching for customer information. By customizing your search environment, you can make these searches more efficient.

One of the most common ways that an agent obtains customer information is through searches. Therefore, you may want to customize the search pages so that you can determine what an agent needs when obtaining pertinent information.

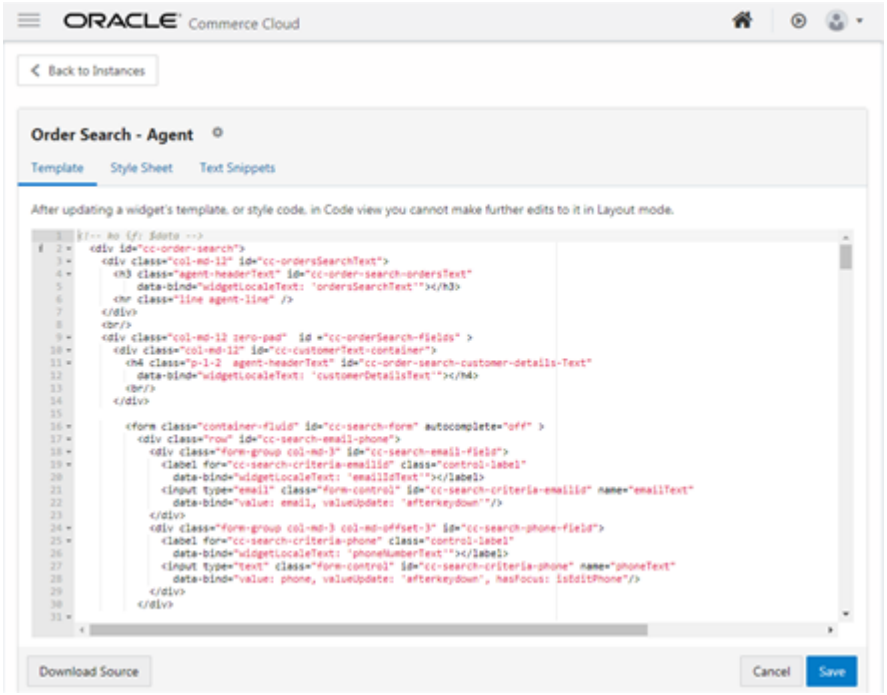
Before making any customizations, you should be familiar with how widgets are downloaded, created and added to the server.

Note that the example code in this section is for illustrative purposes only; it is not intended to be production-ready, and may not adequately handle all possible use cases or implement the exact behavior you want.

Remove fields from a search page

The agent console can be modified so that it displays only certain search fields. The following information explains how to remove fields from a search page.

Remove a field using the code view screen on the **Templates** tab.



You can any add any number of fields using the code view, but only the fields that are supported by the Search API will display results. For example, to prevent the order ID field from displaying on the screen, you could remove the `<div>` with the ID "`cc-search-criteria-id-field`", then save the code and publish it.

Add fields to a search page

Use the Oracle Commerce REST web services APIs to add customer properties to your search pages. See Extending Oracle Commerce Cloud for information you need to know before using the services.

To add a property, for example, `myNewProperty`, to the screen you must make changes to the widget's JavaScript as well as the HTML template. For example, you can use the `searchOrders` endpoint to view a new property, `paymentType`, which you may want to add to the page to allow an agent to search for an order based on the type of payment used. The endpoints that you use for this example are:

```
/ccagent/v1/orders/serachOrders
/ccagent/v1/profiles/searchProfiles
/ccagent/v1/returnRequests/searchReturns
```

To make modifications to a widget, you must first download the widget. Then you make changes to the JavaScript and HTML template file. Once you have completed your changes, upload the widget. This upload process is described in detail in the Developing Storefront Classic Widgets for Oracle Commerce guide.

To modify the widget's JavaScript:

1. Identify the fields in the search API for the new field. For example, `paymentType`.
2. In the widget's JavaScript add a new observable in the `initOrderSearchCriteriaViewModel` method.

```
initOrderSearchCriteriaViewModel: function () {
   var self = this;
   self.accountNameSelected = ko.observableArray([]);
   self. paymentType = ko.observable('');
   self.orderId = ko.observable('');
   self.email = ko.observable('');
   self.firstName = ko.observable('');
   self.lastName = ko.observable('');
   self.account = ko.observable('');
   self.approver = ko.observable('');
   self.selectedSite = ko.observable('');
   self.selectedOrderState = ko.observable('');
   self.skuId = ko.observable('');
   self.productId = ko.observable('');
   self.timeValueForLastOrders = ko.observable('');
   self.timeUnitForLastOrders = ko.observable('');
   self.phone = ko.observable('');
   self.isEditPhone = ko.observable(false);
   self.isAdvancedSearch = ko.observable(false);
   self.timeUnits = [];
```

3. Clear the observable that was just added in the `resetBasicSearch` function.

```
/** * Resets the basic search fields.
*/
resetBasicSearch: function() {
this.orderId('');
this.email('');
this. paymentType ('');
this.firstName('');
this.lastName('');
this.accountNameSelected([]);
this.approver('');
this.selectedSite('');
this.selectedOrderState('');
this.timeValueForLastOrders('');
this.timeUnitForLastOrders(this.resources().days);
this.skuId('');
this.productId('');
this.phone('');
```

4. Modify the `getTextSearchCriteria` function to include the newly added property for text search queries.

```
/**
* Gets the Text search criteria.
*/
getTextSearchCriteria: function() {
```

```
  var self = this;
  var searchCriteria = self.getBasicSearchCriteria();
  searchCriteria[CCConstants.LIMIT] = this.orderSearchViewModel.
              itemsPerPage;
  searchCriteria[CCConstants.REQUIRE_COUNT] = false;
  return searchCriteria;
},
```

5. Modify the `getSCIMSearchCriteria` function to include the newly added property for SCIM search.

```
getSCIMSearchCriteria: function() {
  var self = this;
  var data = {};
  ...
  data.fields = "id,priceGroupId,siteId, paymentType,
    submittedDate,state,profile, priceInfo,
    payShippingInSecondaryCurrency,payTaxInSecondaryCurrency
    secondaryCurrencyCode,organization";
  return data;
},
```

6. Modify the widget's HTML template. To do this, add the required `<div>` in the template, this can be done in any form whose ID is `cc-search-form`.

Note that the search API also supports search by dynamic properties. To add a dynamic property to the search screen, such as `isVipMember`, ensure that you follow the naming convention for the search criteria given in the order search API documentation.

10

Create and Edit Orders Widgets

The widgets described in this section allow you to customize how an agent creates or edits an order. You may customize the following:

- Customer Details section
- Cart operations such as Add by SKU, Add by Product Name and Add from Catalog)
- Address book for both shipping and billing
- Shipping Method
- Payment Details. Note that if you have already customized payment properties, you should use the storefront payment widget to work with your customized properties. Refer to Appendix: Layout Widgets and Elements for information on storefront widgets.
- Notes
- Display and arrange custom order properties both internal and external
- Ability to make agent-specific operations such as price override.

Help a customer with their shopping cart

This widget displays the customer's shopping cart in the agent's context, allowing an agent to log on and assist a shopper.

You can use this widget to display and manage agent-specific functions, such as override prices, change quantities, customize items or edit and delete add-on products.

Widget Name: `agentShoppingCart`

Display Name: Shopping Cart - Agent

Supported Page Types: `agentCheckoutPageType`

Layouts:

- B2B Edit Layout - Agent
- B2B Checkout Layout - Agent
- Checkout Edit Layout - Agent
- Checkout Layout - Agent

Elements: None

Search for and add items to a shopper's cart

This widget presents an interface that allows an agent to search for a product or SKU and add it to the shopper's cart.

The agent can perform the search by the product name or ID, or the SKU ID.

Widget Name: `searchAndAddItemsToCart`

Display Name: Search And Add Items To Cart

Supported Page Types: `agentCheckoutPageType`

Layouts:

- B2B Checkout Layout - Agent
- B2B Edit Layout - Agent
- Checkout Edit Layout - Agent
- Checkout Layout - Agent

Elements: This widget contains the `sku-search` and `product-search` elements. It also contains the `add-from-catalog` widget-specific element. To understand how these elements are used, refer to the widget's API documentation.

Note: Ensure that you have the latest version of these widgets and their associated layouts.

Place an order for a shopper during checkout

This widget displays the place order or scheduled order button to the agent, during the checkout process, allowing the agent to place an order on behalf of the shopper.

Widget Name: `agentCheckoutPlaceOrderSummary`

Display Name: Place Order - Agent Checkout

Supported Page Types:

- `agentCheckoutPageType`
- `agentMultiShipCheckoutPageType`

Layouts:

- B2B Checkout Layout - Agent
- B2B Edit Layout - Agent
- Checkout Edit Layout - Agent
- Checkout Edit Layout For Multi Ship
- Checkout Layout - Agent
- Checkout Layout For Multi Ship
- Checkout Layout For Pending Payment - Agent

Elements: This widget uses the `dynamic-property` element. To understand how this element is used, refer to the widget's code view.

Work with an organization address book

This widget, which is used only in account-based environments, displays an interface that allows an agent to manage an account's address book.

Note that a version of this widget exists within the Storefront framework, as well. See Appendix: Layout Widgets and Elements for information on Storefront widgets.

Migrate widgets

If you have recently migrated and this widget may be visible in your widget list. However, the widget is using the default template and .less file, which may not be available to the page layout. Additionally, this widget's style format must be updated so that it appears in the layout after a migration.

To migrate, you must modify the following `widget.template` and `.less` files:

- `/layouts/organizationAddressSelectorAgentLayout/widget.template`
- `/less/widget-organizationAddressSelectorAgentInst_latest_version_number.less`

Widget Name: `organizationAddressSelector`

Display Name: Address book for B2B - Agent

Supported Page Types:

- `agentCheckoutPageType`
- `checkout`

Layouts:

- B2B Checkout Layout - Agent
- B2B Edit Layout - Agent

Elements: This widget uses the `dynamic-property` element. To understand how this element is used, refer to the widget's code view.

View product details

This widget displays an interface that allows an agent to view the Product Detail page when an access point to the product detail page has been selected.

The agent-specific widget contains stock table and add-ons elements.

Note that a version of this widget exists within the Storefront framework, as well. See Appendix: Layout Widgets and Elements for information on Storefront widgets.

Widget Name: `agentProductDetails`

Migrate widgets

If you have recently migrated and this widget may be visible in your widget list. However, the widget is using the default template file, which may not be available to the page layout. The `widget.template` instances that must be migrated are the following:

- `/layouts/addToPurchaseListLayout/widget.template`
- `/layouts/agentProductDetailsDefaultLayout/widget.template`
- `/layouts/productDetailsCPQChildeItemsLayout/widget.template`

Display Name: Product Detail - Agent

Supported Page Types:

- `agentCheckoutPageType`
- `agentCreateExchangePageType`
- `agentOrderDetailsPageType`
- `agentPurchaseLsitPageType`
- `categoryPageType`
- `productPageType`
- `searchResultsPageType`

Layouts:

- Agent Collection Layout
- B2B Checkout Layout - Agent
- B2B Edit Layout - Agent
- Checkout Edit Layout - Agent
- Checkout Layout - Agent
- Create Exchange Request Layout - Agent
- Create Return Layout - Agent
- Process Returns Layout - Agent
- Product Layout - Agent
- Refunds Layout - Agent
- Search Results Layout - Agent
- View Return Request Layout - Agent

Elements: To understand how these elements are used, refer to the widget's code view:

- `add-to-purchase-list`
- `agent-product-details-add-item-to-purchase-list`
- `agent-product-details-product-addons`
- `agent-product-details-product-add-to-cart`
- `agent-product-details-product-backorder-message`
- `agent-product-details-product-configure`
- `agent-product-details-product-description`
- `agent-product-details-product-external-price`
- `agent-product-details-product-image`
- `agent-product-details-product-in-stock-message`
- `agent-product-details-product-list-price`
- `agent-product-details-product-long-description`
- `agent-product-details-product-out-of-stock-message`
- `agent-product-details-product-preorder-message`

- `agent-product-details-product-price-range`
- `agent-product-details-product-quantity`
- `agent-product-details-product-sale-price`
- `agent-product-details-product-shipping-surcharge`
- `agent-product-details-product-title`
- `agent-product-details-product-variants`
- `dynamic-property`

Review an order summary during checkout

This widget displays a summary of order details, shipping method and also allows an agent to place orders.

Note that a version of this widget exists within the Storefront framework, as well. See Appendix: Layout Widgets and Elements for information on Storefront widgets.

Migrate widgets

If you have recently migrated and this widget may be visible in your widget list. However, the widget is using the default template and .less file, which may not be available to the page layout. Additionally, this widget's style format must be updated so that it appears in the layout after a migration.

To perform the migration, you must update the following `widget.template` and `.less` files:

- `/layouts/checkoutOrderSummaryAgentLayout/widget.template`
- `/less/widget-agentCheckoutOrderSummaryInst_`*`latest_version_number`*`.less`

Widget Name: `checkoutOrderSummary`

Display Name: Checkout Order Summary - Agent

Supported Page Types: `agentCheckoutPageType`

This widget is also used in the Storefront user interface, and can be applied to storefront page types.

Layouts:

- B2B Checkout Layout - Agent
- B2B Edit Layout - Agent
- Checkout Edit Layout - Agent
- Checkout Edit Layout For Multi Ship - Agent
- Checkout Layout - Agent
- Checkout Layout For Multi Ship - Agent

Elements: None.

Review order details during checkout

This widget displays specific information about the order, such as name, email, account, to the agent.

Note that a version of this widget exists within the Storefront framework, as well. See Appendix: Layout Widgets and Elements for information on Storefront widgets.

Migrate widgets

If you have recently migrated and this widget may be visible in your widget list. However, the widget is using the default template and .less file, which may not be available to the page layout. Additionally, this widget's style format must be updated so that it appears in the layout after a migration.

To migrate, you must update the following `widget.template` and `.less` files:

- `/layouts/orderDetailsDefaultLayout/widget.template`
- `/less/widget-agentAccountOrderDetailsInst_latest_version_number.less`

Widget Name: `checkoutOrderDetails`

Display Name: Checkout Order Details

Supported Page Types: `agentCheckoutPageType`

Layouts:

- B2B Checkout Layout - Agent
- B2B Edit Layout - Agent
- Checkout Edit Layout - Agent
- Checkout Edit Layout For Multi Ship - Agent
- Checkout Layout - Agent
- Checkout Layout For Multi Ship - Agent

Elements: None

Note: Quick links are not available when creating, editing or viewing order details. To enable quick links, you must create a custom property on this widget.

Review order details with pending payments

This widget displays specific information about an order that is pending payment, such as name, email, account, etc.

Note that a version of this widget exists within the Storefront framework, as well. See Appendix: Layout Widgets and Elements for information on Storefront widgets.

Migrate widgets

If you have recently migrated and this widget may be visible in your widget list. However, the widget is using the default template and .less file, which may not be available to the page layout. Additionally, this widget's style format must be updated so that it appears in the layout after a migration.

To migrate, you must update the following `widget.template` and `widget.less` files:

- `/layouts/pendingPaymentOrderDetailsLayout/widget.template`
- `/less/widget.less`

Widget Name: `agentAccountOrderDetails`

Display Name: Order Details for Pending Payment - Agent

Supported Page Types: `agentCheckoutPageType`

Layouts:

- B2B Checkout Layout - Agent
- B2B Edit Layout - Agent
- Checkout Edit Layout - Agent
- Checkout Edit Layout For Multi Ship - Agent
- Checkout Layout - Agent
- Checkout Layout For Multi Ship - Agent

Elements: To understand how these elements are used, refer to the widget's code view:

- `agent-email-order-details`
- `agent-order-payment-details`
- `agent-order-price-details`
- `agent-order-refresh`
- `agent-order-summary`
- `agent-promotion-summary`
- `scheduled – order-actions`
- `scheduled-order-executionList`
- `scheduled-order-instruction`
- `shopping-cart-details`

Provide notes

This widget allows access to the agent-notes global element that enables an agent to view and add notes to the order or profile.

Widget Name: `agentNotes`

Display Name: Notes Widget - Agent

Supported Page Types:

- `agentCheckoutPageType`
- `agentMultiShipCheckoutPageType`
- `agentOrderDetailsPageType`

Layouts:

- Agent Checkout Layout - Pending Payment
- B2B Checkout Layout - Agent
- B2B Edit Layout - Agent
- Checkout Edit Layout - Agent
- Checkout Edit Layout For Multi Ship - Agent
- Checkout Layout - Agent
- Checkout Layout For Multi Ship - Agent
- Order Details Layout - Agent

Elements: This widget uses the `agent-notes` element. To understand how this element is used, refer to the widget's code view.

Apply split payments

This widget provides a way for an agent to apply multiple payment types within an order.

It reflects applied payments, the amount due and any pending payments. The widget supports both single and split payment mode.

Note: Shoppers can pay for an order using either loyalty points or in a monetary currency or a mix of currencies. Loyalty points can be used if `allowAlternateCurrency` is enabled.

Widget Name: `agentSplitPayments`

Display Name: Split Payments - Agent

Supported Page Types:

- `agentCheckoutPageType`
- `agentMultiShipCheckoutPageType`

Layouts:

- Agent Checkout Layout - Pending Payment
- B2B Checkout Layout - Agent
- B2B Edit Layout - Agent
- Checkout Edit Layout - Agent
- Checkout Edit Layout For Multi Ship - Agent
- Checkout Layout - Agent
- Checkout Layout For Multi Ship - Agent

Elements: To understand how these elements are used, refer to the widget's code view:

- `agentsplitpayment-addpayment`
- `agentsplitpayment-cash`
- `agentsplitpayment-creditcard`

- `agentsplitpayment-giftcard`
- `agentsplitpayment-header`
- `agentsplitpayment-invoice`
- `agentsplitpayment-placeorder`
- `agentsplitpayment-storecredit`

Display shipping options

This widget displays various shipping options to the agent.

An agent can use the functionality of this widget to assist a customer in picking up an item in a store. The widget also displays to the agent the number of stores available. For working with Buy Online Pick Up In Store, see Configure Buy Online Pick Up In Store.

Migrate widgets

If you have recently migrated and this widget may be visible in your widget list. However, the widget is using the default template file, which may not be available to the page layout.

To update, you must update the following `widget.template` file:

- `/layouts/multishipAddressBookAgentLayout/widget.template`

Widget Name: `agentShippingOptions`

Display Name: Shipping Options - Agent

Supported Page Types: `agentAdvancedCheckoutPageType`

Layouts: Checkout Layout For Multi Ship - Agent

Elements: None

Work with promotions

This widget allows an agent to see which promotions have been applied to the order, as well as to apply a specific promotion.

Note that a version of this widget exists within the Storefront framework, as well.

Migrate widgets

If you have recently migrated and this widget may be visible in your widget list. However, the widget is using the default template file, which may not be available to the page layout. Additionally, this widget's style format must be updated so that it appears in the layout after a migration.

To do this, you must migrate the following `widget.template` and `.less` files:

- `/layouts/promotionAgentLayout/widget.template`
- `/less/widget-promotionAgentInst_`*`latest_version_number`*`.less`

Widget Name: `promotions`

Display Name: Promotion Widget - Agent

Supported Page Types:

- `agentCheckoutPageType`
- `agentMultiShipCheckoutPageType`

Layouts:

- B2B Checkout Layout - Agent
- B2B Edit Layout - Agent
- Checkout Edit Layout - Agent
- Checkout Edit Layout For Multi Ship - Agent
- Checkout Layout - Agent
- Checkout Layout For Multi Ship - Agent

Elements: This widget uses the `promotions-summary` global element. To understand how this element is used, refer to the widget's code view.

Review cart shipping details

This widget allows an agent to view shipping details from the cart.

Note that a version of this widget exists within the Storefront framework, as well. See Appendix: Layout Widgets and Elements for information on Storefront widgets.

Migrate widgets

If you have recently migrated and this widget may be visible in your widget list. However, the widget is using the default template and .less file, which may not be available to the page layout. Additionally, this widget's style format must be updated so that it appears in the layout after a migration.

To migrate, you must update the following `widget.template` and `.less` files:

- `/layouts/agentShippingMethodsDropdownLayout/widget.template`
- `/less/widget-cartShippingDetailsAgentInst_`*`latest_version_number`*`.less`

Widget Name: `cartShippingDetails`

Display Name: Cart Shipping - Agent

Supported Page Types:

- `agentCheckoutPageType`
- `cart`

Layouts:

- B2B Checkout Layout - Agent
- B2B Edit Layout - Agent
- Checkout Edit Layout - Agent
- Checkout Layout - Agent

Elements: The Storefront version uses `cart-shipping-title`, `cart-shipping-address`, and `cart-shipping-options`. The Agent version uses only the `cart-`

`shipping-options` element. To understand how these elements are used, refer to the widget's code view.

Manage a checkout address book

These widget display a page that enables the agent to add a new address or to select an address that has been stored on a registered shopper's profile during the order process.

Note that a version of the Checkout Address Book widget exists within the Storefront framework, as well. See Appendix: Layout Widgets and Elements for information on Storefront widgets.

Migrate widgets

If you have recently migrated, these widgets may be visible in your widget list. However, these widgets use the default template and .less file, which may not be available to the page layout. Additionally, these widgets' style format must be updated so that they appear in the layout after a migration.

To perform the migration, you must edit the following `template.widget` and `.less` files:

- `/layouts/checkoutAddressBookAgentLayout/widget.template`
- `/less/widget-agentCheckoutOrderSummaryInst_latest_version_number.less`

Widget: `checkoutAddressBook`

Display Name: Checkout Address Book

Supported Page Types:

- `checkout`

Layouts:

- Checkout Edit Layout - Agent
- Checkout Layout - Agent

Elements: None

The following is an agent-specific checkout address book widget:

Widget: `agentCheckoutAddressBook`

Display Name: Agent Checkout Address Book

Supported Page Types:

- `agentCheckoutPageType`

Layouts:

- Checkout Edit Layout - Agent
- Checkout Layout - Agent

Elements: None

Review product purchase list information

This widget displays purchase list information for the product.

For information on working with purchase lists, refer to Enable Purchase Lists.

Widget: `agentProductDetails`

Display Name: Purchase List Product Details - Agent

Supported Page Types:

- `product`
- `category`
- `searchResults`
- `agentCheckout`
- `agentPurchaseList`
- `agentOrderDetails`
- `agentCreateExchange`

Layouts: Purchase List Layout - Agent

Elements: To understand how these elements are used, refer to the widget's code view:

- `agent-product-details-add-item-to-purchase-list`
- `agent-product-details-product-addons`
- `agent-product-details-product-add-to-cart`
- `agent-product-details-product-add-to-space`
- `agent-product=details-product-back-link`
- `agent-product-details-product-backorder-message`
- `agent-product-details-product-configure`
- `agent-product-details-product-description`
- `agent-product-details-product-external-price`
- `agent-product-details-product-image`
- `agent-product-details-product-image-caroursel`
- `agent-product-details-product-in-stock-message`
- `agent-product-details-product-list-price`
- `agent-product-details-product-long-description`
- `agent-product-details-product-out-of-stock-message`
- `agent-product-details-product-preorder-message`
- `agent-product-details-product-price-range`
- `agent-product-details-product-quantity`

- agent-product-details-product-sale-price
- agent-product-details-product-shipping-surcharge
- agent-product-details-product-stock-status-table
- agent-product-details-product-title
- agent-product-details-product-variants
- agent-product-details-text

Review shopping cart product information

This widget displays information to the agent on the current shopping cart.

Note that a version of this widget exists within the Storefront framework, as well. See Appendix: Layout Widgets and Elements for information on Storefront widgets.

Widget: agentProductDetails

Display Name: Purchase List Product Details - Agent

Supported Page Types:

- product
- category
- searchResults
- agentCheckout
- agentPurchaseList
- agentOrderDetails
- agentCreateExchange

Layouts:

- B2B Edit Layout – Agent
- B2B Layout – Agent
- Checkout Edit Layout - Agent
- Checkout Layout - Agent

Elements: To understand how these elements are used, refer to the widget's code view:

- agent-product-details-add-item-to-purchase-list
- agent-product-details-line-break
- agent-product-details-product-addons
- agent-product-details-product-add-to-cart
- agent-product-details-product-add-to-space
- agent-product=details-product-back-link
- agent-product-details-product-backorder-message
- agent-product-details-product-configure

- agent-product-details-product-description
- agent-product-details-product-external-price
- agent-product-details-product-image
- agent-product-details-product-image-caroursel
- agent-product-details-product-in-stock-message
- agent-product-details-product-list-price
- agent-product-details-product-long-description
- agent-product-details-product-out-of-stock-message
- agent-product-details-product-preorder-message
- agent-product-details-product-price-range
- agent-product-details-product-quantity
- agent-product-details-product-sale-price
- agent-product-details-product-shipping-surcharge
- agent-product-details-product-stock-status-table
- agent-product-details-product-title
- agent-product-details-product-variants
- agent-product-details-text

Define a scheduled order during checkout

This widget displays an interface that allows an agent to define a scheduled order based on the current order.

The agent can select a start or end date, the frequency of the order, and the ability to suspend the scheduled order. Note that this widget exists within the Storefront framework, as well. See Create Scheduled Orders for more information. See Appendix: Layout Widgets and Elements for information on Storefront widgets.

Widget Name: `checkoutScheduledOrder`

Display Name: Scheduled Order - Checkout

Supported Page Types: `agentCheckoutPageType`

Layouts:

- B2B Checkout Layout - Agent
- Checkout Layout - Agent
- Checkout Layout For Multi Ship - Agent

Elements: None

Display order information

This widget displays site, account and pricelist group selections to an agent.

This widget also provides backwards navigation. It contains the title, account and site dropdowns, as well as the links to navigate to all sections, etc.

If you have an account-based environment, available selectors include `site` and `account` selection. For standard storefront environments, selectors include `site` and `pricelist` group.

Note that a version of this widget exists within the Storefront framework, as well. See Appendix: Layout Widgets and Elements for information on Storefront widgets.

Migrate widgets

If you have recently migrated and this widget may be visible in your widget list. However, the widget is using the default template file, which may not be available to the page layout.

To migrate, you must update the following `widget.template` instances:

- `/layouts/createOrderDefaultLayout/widget.template`
- `/layouts/editOrderLayout/widget.template`

Widget Name: `createOrderHeader`

Display Name: Header - Agent Create Order

Supported Page Types:

- `agentCheckoutPageType`
- `agentMultiShipCheckoutPageType`

Layouts:

- B2B Checkout Layout - Agent
- B2B Edit Layout - Agent
- Checkout Edit Layout – Agent
- Checkout Edit Layout For Multi Ship - Agent
- Checkout Layout - Agent
- Checkout Layout For Multi Ship - Agent
- Checkout Layout For Pending Payment - Agent

Elements: The `header-text` element and `additional-custom-factors` are widget-specific elements. The `additional-custom-factors` elements allow you to create and display customizations. To understand how these elements are used, refer to the widget's code view:

- `account-selector`
- `additiona-custom-factors`
- `back-button`
- `header-text`
- `plg-selector`
- `site-selector`

Note: To provide header-level quick links, you must create a custom property on this widget.

Display an account's order information

This widget displays information on an account's order.

In an account-based environment, available selectors include `site` and `account` selection.

Note that a version of this widget exists within the Storefront framework, as well. See Appendix: Layout Widgets and Elements for information on Storefront widgets.

Widget Name: `createOrderHeaderB2BLayout`

Display Name: Create Order Header B2B Layout

Supported Page Types:

- `agentCheckoutPageType`
- `agentMultiShipCheckoutPageType`

Layouts:

- B2B Edit Layout - Agent
- Checkout Edit Layout – Agent
- Checkout Edit Layout For Multi Ship - Agent
- Checkout Layout For Multi Ship - Agent
- Checkout Layout For Pending Payment - Agent

Elements: The `header-text` element and `back-button` are widget-specific elements. To understand how these elements are used, refer to the widget's code view.

Work with the order states and numbers

This widget displays an order's state and number.

It also supports **Cancel**, **Edit** and **Make Payment** buttons.

Migrate widgets

If you have recently migrated and this widget may be visible in your widget list. However, the widget is using the default template, which may not be available to the page layout. In this case, you will need to manually add it by copying and pasting the new widget's template code into the existing widget's template code. Additionally, this widget may not be available in the layout. You must update the widget's LESS file.

Widget Name: `agentOrderDetailsHeader`

Display Name: Header - Agent Order Details

Supported Page Types: `agentOrderDetails` page

Layouts: Order Details Layout - Agent

Elements: None

View the catalog page

This widget displays the header for the catalog page.

When an agent clicks on the Complete Order button, they are taken to the agent checkout page, which allows them to complete an order on behalf of a shopper.

Note that a version of this widget exists within the Storefront framework, as well. See Appendix: Layout Widgets and Elements for information on Storefront widgets.

Widget Name: `agentCatalogHeader`

Display Name: Header - Agent Catalog

Supported Page Types:

- `category`
- `nosearchresults`
- `product`
- `searchresults`

Layouts:

- Collection Layout - Agent
- No Search Results Layout - Agent
- Product Layout - Agent
- Search Results Layout - Agent

Elements: To understand how these elements are used, refer to the widget's code view:

- `dropdown-minicart`

View collections

This widget displays navigation that allows an agent to view collections.

For information on collections, see Organize products in collections.

Widget Name: `megMenu`

Display Name: Collection Navigation Widget

Supported Page Types: All

Layouts:

- Product Layout – Agent
- No Search Results Layout – Agent
- Collection Layout – Agent
- Search Results Layout - Agent

Elements: None

Review loyalty payments

This widget displays information to an agent regarding the shopper's loyalty payments.

For detailed information on working with loyalty programs, see Work with Loyalty Programs.

Note that a version of this widget exists within the Storefront framework, as well. See Appendix: Layout Widgets and Elements for information on Storefront widgets.

Widget Name: `loyaltyPayment`

Display Name: Loyalty Payment

Supported Page Types:

- `agentCheckoutPageType`
- `agentMultiShipCheckoutPageType`
- `checkout`

Layouts:

- Agent Checkout Layout - Pending Payment
- B2B Edit Layout - Agent
- Checkout Edit Layout - Agent
- Checkout Edit Layout For Multi Ship - Agent
- Checkout Layout - Agent
- Checkout Layout For Multi Ship - Agent

Elements: This widget uses the `select-redeem-points` element. To understand how this element is used, refer to the widget's code view.

Obtain loyalty details

This widget allows an agent to see a shopper's loyalty information, such as information on points accumulated, if the points are available for spending or have already been used.

For detailed information on working with loyalty programs, see Work with Loyalty Programs.

Note that a version of this widget exists within the Storefront framework, as well. See Appendix: Layout Widgets and Elements for information on Storefront widgets.

Widget Name: `loyaltyDetails`

Display Name: Loyalty Details

Supported Page Types: All

Layouts:

- Agent Checkout Layout - Pending Payment
- B2B Edit Layout - Agent

- Checkout Edit Layout - Agent
- Checkout Edit Layout For Multi Ship - Agent
- Checkout Layout - Agent
- Checkout Layout For Multi Ship - Agent

Elements: This widget uses the `loyalty-details-basic` element. To understand how this element is used, refer to the widget's code view.

Configure CyberSource payment authorization

This widget provides functionality that allows an agent to enter data that is used during payment authentication.

For information on working with CyberSource authorization, see Configure Payment Processing.

Note that a version of this widget exists within the Storefront framework, as well. See Appendix: Layout Widgets and Elements for information on Storefront widgets.

Widget Name: `cyberSourcePaymentAuthorization`

Display Name: CyberSource Payment Authorization

Supported Page Types: `paymentPageType`

Layouts: Payer Authentication Layout

Elements: None

Customize a create order page

The following widgets, which are associated with the Checkout Layout - Agent, provide an interface that allows an agent to create an order.

- Navigation – Agent
- Notification Widget – Agent
- Header – Agent Create Order
- Additional Shopper Context
- Checkout Order Details
- Checkout Order Summary – Agent
- Search And Add Items To Cart
- Product Details – Agent
- Shopping Cart – Agent
- Shopping Cart Product Details – Agent
- Promotion Widget – Agent
- Scheduled Order – Agent Checkout
- Address book for B2C Customer – Agent
- Cart Shipping – Agent

- Loyalty Payment
- Loyalty Details
- Split Payments – Agent
- Notes Widget – Agent
- Request Quote Widget – Agent
- Place Order – Agent Checkout

You can customize this layout by adding or removing widgets. An example of how you could modify the default checkout layout is described in Configure layouts and widgets for in-store pick up.

Note: The list price is not displayed in the shopping cart price column. To display the list price, you must update the Shopping Cart - Agent widget.

Note that the following layouts also exist for an agent. These layouts contain a subset of the widgets listed above, as well as widgets specific to the layout's purpose:

- Checkout Edit Layout – Agent
- Checkout Layout For Pending Payment – Agent
- Checkout Layout For Multi Ship – Agent
- Checkout Edit Layout For Multi Ship - Agent

Note: To add quick links when creating, editing or viewing order details, you must create custom properties on the Header - Agent Create Order widget.

11

Work with Customer Information Widgets

This section describes the widgets that allow an agent to work with customer information.

There are a number of ways to work with customer information, you can access customer information from a customer profile, orders or purchase lists.

Work with customer profile details

This widget allows an agent to view in-depth details of the shopper.

The agent can use the interface provided by this widget to view the customer information, loyalty details, storefront roles, store credit, and email references. The agent can also use this interface to resend the password, create an order or view the cart link, as well as launch the store on behalf of the shopper.

You could customize this widget to display addresses in a different format, display the customer's order history, display orders pending approval or display scheduled orders.

Widget Name: `customerProfileDetails`

Display Name: Customer Profile Details - Agent

Supported Page Types: `agentProfilePageType`

Layouts: Customer Details Layout - Agent

Elements: To understand how these elements are used, refer to the widget's code view:

- `agent-notes`
- `customer-details`
- `customer-loyalty-programs`
- `customer-pending-orders`
- `customer-profile-reset-password`
- `customer-profile-save-cancel`
- `customer-profile-status`
- `customer-store-credit-balance`
- `customer-store-roles`
- `dynamic-property`
- `launch-store-as-customer`

Note: The Customer Profile page does not have a back navigation button by default. To create one, you must update the widget with a custom property.

Display saved credit cards

You can modify the Customer Profile Details widget to display saved credit cards on the profile details page. This allows a customer to save their credit card information and reuse it whenever they want. The following is an overview of the steps you would take to display stored credit cards.

For detailed information on working with and creating saved credit cards widgets, see Support stored credit cards.

1. Use the REST API to initiate the `listCreditCards` endpoint for a shopper profile.
2. This returns the customer's credit card information. Each item in the results displays card information. Save the results.
3. Create a new instance of the `CreditCard` view model.
4. Use the `populateData` method to add the information returned in the response.
5. 5. The shopper's card information is stored in the `allCreditCards` observable array, in the Customer Profile Details widget.
6. Modify the Customer Profile Details widget to display the card details in the UI.

View a customer summary

This widget is used in the profile page to show a summary of customers' basic information along with site selector and account selector options.

Note that a version of this widget exists within the Storefront framework, as well. See Appendix: Layout Widgets and Elements for information on Storefront widgets.

Widget Name: `customerSummary`

Display Name: Customer Summary Widget

Supported Page Types: `agentProfilePageType`

Layouts:

- Account Contacts Layout - Agent
- Account Details Layout - Agent
- Address Book Layout - Agent
- Customer Details Layout - Agent
- Order History Layout - Agent
- Orders Pending Approval Layout - Agent
- Purchase List Layout - Agent
- Scheduled Order Layout - Agent

Elements: To understand how these elements are used, refer to the widget's code view:

- `account-selector`
- `customer-basic-information`
- `site-selector`

Note: The Customer Profile page does not have a back navigation button by default. To create one, you must update the widget with a custom property.

Review order history

This widget can display a shopper's order history.

Note that a version of this widget exists within the Storefront framework, as well. See Appendix: Layout Widgets and Elements for information on Storefront widgets.

Migrate widgets

If you have recently migrated and this widget may be visible in your widget list. However, the widget is using the default template and .less file, which may not be available to the page layout. Additionally, this widget's style format must be updated so that it appears in the layout after a migration.

To migrate, you must update the following `widget.template` and `.less` files:

- `/layouts/agentOrderHistoryLayout/widget.template`
- `/less/widget-orderHistoryWidgetInst_agent_`*`latest_version_number`*`.less`

Widget Name: `orderHistoryWidget`

Display Name: Order History Widget - Agent

Supported Page Types:

- `agentOrderHistoryPageType`
- `agentProfilePageType`
- `orderHistoryPageType`
- `profilePageType`

Layouts: Order History Layout - Agent

Elements: None

Note: To filter order histories by Order ID, you must create a custom property for this widget.

View purchase lists

This widget displays a list of purchase lists.

Note that a version of this widget exists within the Storefront framework, as well. See Appendix: Layout Widgets and Elements for information on Storefront widgets.

Migrate widgets

If you have recently migrated and this widget may be visible in your widget list. However, the widget is using the default template and .less file, which may not be available to the page layout. Additionally, this widget's style format must be updated so that it appears in the layout after a migration.

To migrate, you must update the following `widget.template` and `.less` files:

- `/layouts/agentPurchaseListsLayout/widget.template`

- `/less/widget-purchaseListsDetailsWidgetAgent_latest_version_number.less`

Note: An agent can no longer indicate that the purchase list is applicable for all sites when working in the administrative interface. To allow a purchase list to be applicable for all sites, you must create a customized widget, similar to the widget used in the storefront application.

Widget Name: `purchaseLists`

Display Name: Purchase List

Supported Page Types: All

Layouts: Purchase List Layout - Agent

Elements: None

Provide additional shopper context

This widget displays shopper context information to an agent.

Note that a version of this widget exists within the Storefront framework, as well. See Appendix: Layout Widgets and Elements for information on Storefront widgets.

Widget Name: `additionalShopperContext`

Display Name: Additional Shopper Context

Supported Page Types: `agentCheckoutPageType`

Layouts:

- B2B Checkout Layout - Agent
- Checkout Layout - Agent

Elements: None

Display address for account-based contacts

This widget displays customer address information.

It also allows an agent to select a site or account if there are multiple options.

Widget Name: `checkoutAddressBook`

Display Name: Address Book for B2C Customer - Agent

Supported Page Types:

- `Checkout`
- `agentCheckout`

Layouts:

- Checkout Edit Layout – Agent
- Checkout Layout - Agent

Elements: This widget uses the `dynamic-property` element. To understand how this element is used, refer to the widget's code view.

Work with tab navigation

This widget provides tab navigation on the profile pages.

Note that a version of this widget exists within the Storefront framework, as well. See Appendix: Layout Widgets and Elements for information on Storefront widgets.

Widget Name: `secondaryNavigation`

Display Name: Secondary Navigation

Supported Page Types: To understand how these elements are used, refer to the widget's code view:

- `agentAccountContactsPageType`
- `agentAccountDetailsPageType`
- `agentAddressBookPageType`
- `agentCustomerDetailsPageType`
- `agentCustomerSearchPageType`
- `agentOrderHistoryPageType`
- `agentOrdersPendingApprovalPageType`
- `agentPurchaseListPageType`
- `agentScheduledOrderPageType`
- `agentSelfRegistrationPagePageType`

Layouts:

- Account Contacts Layout - Agent
- Account Details Layout - Agent
- Address Book Layout - Agent
- Customer Details Layout - Agent
- Order History Layout - Agent
- Orders Pending Approval Layout - Agent
- Purchase List Layout - Agent
- Scheduled Order Layout - Agent

Elements: None

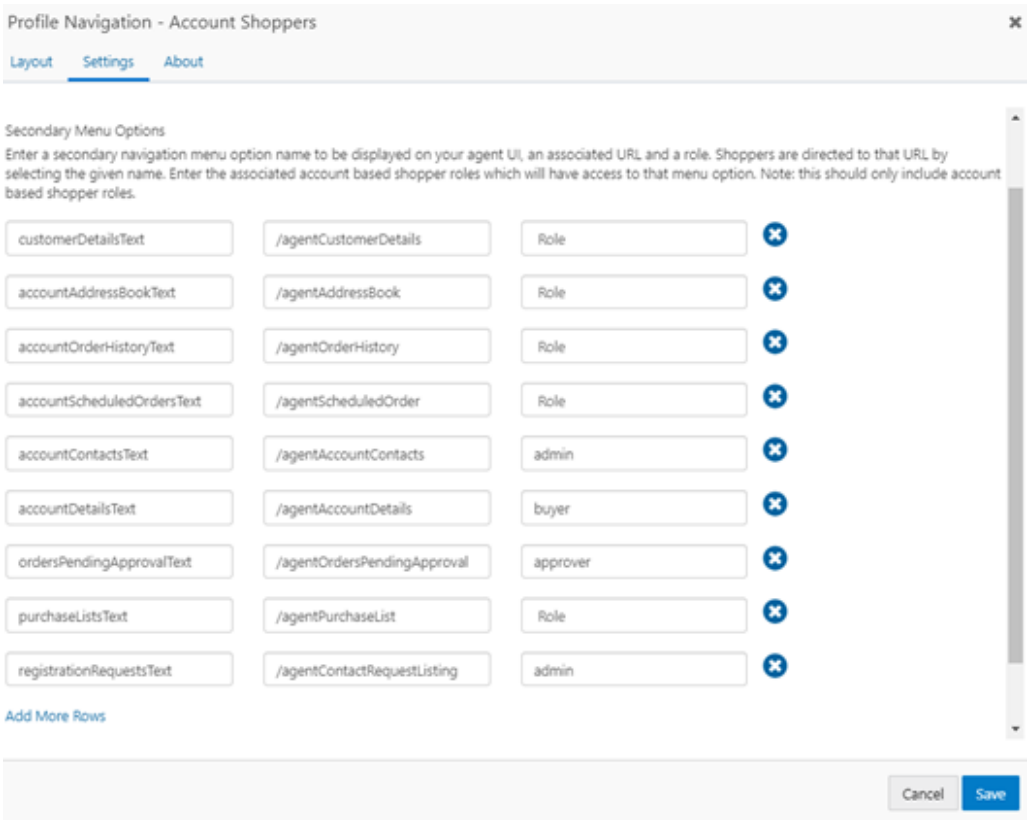
Migrating widgets

If you have recently migrated and this widget may not be visible in your widget list. In this case, you will need to manually add it by copying and pasting the new widget's template code into the existing widget's template code.

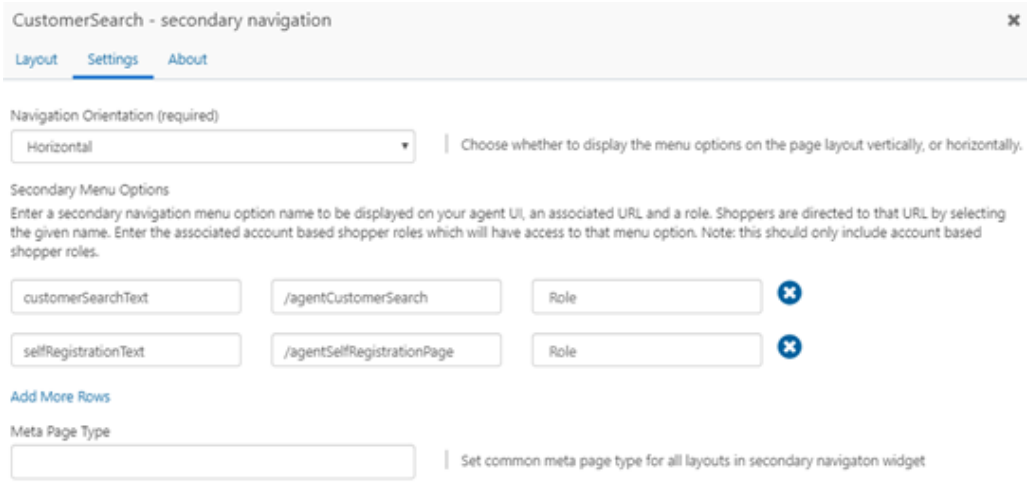
The `widget.template` must be migrated for each instance of the widget:

- `/layouts/agentSecondaryNavigationLayout/widget.template`

When migrating the template code for the widget, ensure that the following settings are enabled in the Profile Navigation-Account Shoppers instance:



When migrating the template code for the Customer Search - Secondary Navigation instance, ensure that the following settles are enabled:



12

Customize Self-Registration Widgets

This section describes widgets that can be customized for self-registration. These include widgets that allow you to customize registration details and configure notification or navigation.

These include widgets that allow you to customize registration details and configure notification or navigation.

Search registration requests

This widget presents a page that allows an agent to search for account and contact registration requests.

Widget Name: `registrationRequestSearch`

Display Name: Registration Request Search

Supported Page Types: `agentSelfRegistrationPagePageType`

Layouts: Self Registration Layout - Agent

Elements: None

View self registration details

This widget enables the agent to view read-only data for accounts or contacts self registration requests.

Note that a version of this widget exists within the Storefront framework, as well. See Appendix: Layout Widgets and Elements for information on Storefront widgets.

Widget Name: `selfRegistrationDetail`

Display Name: Self Registration Detail

Supported Page Types: `agentRegistrationRequestDetailPageType`

Layouts: Registration Request Detail Layout - Agent

Elements: To understand how these elements are used, refer to the widget's code view:

- `dynamic-property`
- `back-button`

Note: Backward navigation is not available on the view self registration details page. To create a back button, create a custom property on this widget.

Configure notifications

This widget provides an interface that allows an agent to view notifications when an error or a success message occurs.

Widget Name: `notifications`

Display Name: Notification Widget - Agent

Page Types: All

Layouts:

- Account Contacts Layout - Agent
- Account Details Layout - Agent
- Address Book Layout - Agent
- B2B Checkout Layout - Agent
- B2B Edit Layout - Agent
- Checkout Edit Layout - Agent
- Checkout Edit Layout For Multi Ship - Agent
- Checkout Layout - Agent
- Checkout Layout For Multi Ship - Agent
- Collection Layout - Agent
- Create Exchange Request Layout - Agent
- Create Return Layout - Agent
- Customer Details Layout - Agent
- Customer Search Layout - Agent
- No Search Results Layout - Agent
- Order Details Layout - Agent
- Order History Layout - Agent
- Orders Pending Approval Layout - Agent
- Process Returns Layout - Agent
- Product Layout - Agent
- Purchase List Layout - Agent
- Refund Layout - Agent
- Registration Request Detail Layout - Agent
- Scheduled Order Layout -Agent
- Search Results Layout - Agent
- View Return Request Layout - Agent

Elements: This widget uses the `notification-message-box` element. To understand how this element is used, refer to the widget's code view.

Review and create customer registration

This widget provides an interface that allows an agent to create and register a new contact or account.

Widget Name: `agentCustomerRegistration`

Display Name: Customer Registration - Agent

Page Types: `agentCustomerSearchPageType`

Layouts: Customer Search Layout - Agent

Elements: To understand how these elements are used, refer to the widget's code view:

- `dynamic-property`
- `site-selector`

View contact registration

This widget displays a list of pending contact registrations requests.

For information on account registration, see Understand Accounts.

Note that a version of this widget exists within the Storefront framework, as well. See Appendix: Layout Widgets and Elements for information on Storefront widgets.

Widget Name: `agentCustomerRegistration`

Display Name: Customer Registration - Agent

Supported Page Types:

- `agentContactRequestsPageType`
- `agentCustomerSearchPageType`

Layouts: Contact Registration Listing

Elements: None

Customize navigation for customer search and self registration

This widget displays the tab navigation in the Customer Search page.

Note that a version of this widget exists within the Storefront framework, as well. See Appendix: Layout Widgets and Elements for information on Storefront widgets.

Widget Name: `secondaryNavigation`

Display Name: Secondary Navigation

Supported Page Types:

- `agentCustomerSearchPageType`
- `agentSelfRegistrationPagePageType`

Layouts:

- Customer Search Layout - Agent
- Self Registration Layout - Agent

Elements: None

Note: There is no paging available for addresses displayed in the Address tab. To enable paging, you must create a custom property.

13

Configure Returns and Exchanges Widgets

The following widgets allow you to customize how an agent works with return requests and exchanges.

These widgets allow you to configure how returns and exchanges are processed or created. These widgets also allow you to view history information and customize refunds.

Work with returns

The following widgets can be used by an agent to view, create and process returns.

Create return requests

This widget displays an interface used by an agent to initiate a return. The agent can make or edit comments for each return item.

Widget Name: `agentCreateReturn`

Display Name: Create Return - Agent

Supported Page Types: `agentCreateReturnPageType`

Layouts: Create Return Layout - Agent

Elements: This widget uses the `back-button` and `custom-properties` global elements. To understand how these elements are used, refer to the widget's code view.

Process returns

This widget displays an interface used by an agent to acknowledge the receipt of items that have been shipped back to the warehouse. This interface also allows the agent to update the system with the quantity received against a return request or a reason for the return. The agent can make or edit comments for each return item.

Widget Name: `agentProcessReturns`

Display Name: Process Returns - Agent

Supported Page Types: `agentProcessReturnsPageType`

Layouts: Process Returns Layout - Agent

Elements: This widget uses the `back-button` and custom-properties global elements. To understand how these elements are used, refer to the widget's code view.

View return history details

This widget displays return history for the order. It also displays return-specific details, as well as the ability for the agent to take actions such as receive the return or refund payment.

Widget Name: `agentReturnHistoryDetails`

Display Name: Return History - Agent

Supported Page Types: `agentOrderDetails` page

Layouts: Order Details Layout - Agent

Elements: None

View a return request

This widget displays a return request to an agent once the return has been processed or submitted, and the items are marked as complete.

Widget Name: `agentViewReturnRequest`

Display Name: View Return Request - Agent

Supported Page Types: `agentViewReturnRequestPageType`

Layouts: View Return Request Layout - Agent

Elements: This widget uses the `back-button` and `custom-properties` global elements. To understand how these elements are used, refer to the widget's code view.

Work with exchanges

The following widgets allow an agent to create and review exchanges.

Create an exchange

This widget provides an interface that can be used by an agent to initiate or create an exchange request. This interface also allows agents to select the items and quantities to be exchanged. Additionally, the agent may add a reason for the exchange. Once the exchange has been submit, a new exchange order is created.

Widget Name: `agentCreateExchange`

Display Name: Create Exchange - Agent

Supported Page Types: `agentCreateExchangePageType`

Layouts: Create Exchange Layout - Agent

Elements: This widget uses the `back-button` and `custom-properties` global elements. To understand how these elements are used, refer to the widget's code view.

View exchange history details

This widget displays exchange history for the order. It also displays exchange-specific details, as well as a way for the agent to take actions, such as receive or process the exchange.

Widget Name: `agentExchangeHistoryDetails`

Display Name: Exchange History - Agent

Supported Page Types: `agentOrderDetails` page

Layouts: Order Details Layout - Agent

Elements: None

Adjust refunds

This widget displays an interface that allows an agent to initiate the refund process. The agent may adjust the system-generated refund before processing the refund.

Widget Name: `agentRefunds`

Display Name: Refunds - Agent

Supported Page Types: `agentRefundsPageType`

Layouts: Refunds Layout - Agent

Elements: This widget uses the `back-button` and `custom-properties` global elements. To understand how these elements are used, refer to the widget's code view.

14

Use Account-Based Widgets

You can create widgets for account-based environments, that work specifically for business-to-business customers.

The following widgets are used with account-based contacts and organizations.

View account customer carts

This widget displays a page that allows an agent to view customer carts.

This widget also displays the search results in a table, and allows the agent to access different pages, such as clicking on an order ID that directs the agent to the order details page.

Widget Name: `agentCustomerCartsDialog`

Display Name: Customer Carts Dialog - Agent

Supported Page Types:

- `agentCheckoutPageType`
- `agentCustomerSearchPageType`

Layouts: Customer Search Layout-Agent

Elements: To understand how these elements are used, refer to the widget's code view:

- `account-selector`
- `additional-shopper-context`
- `site-selector`

View account order details

This widget displays the contents of the order, including order items, shipping details payment details and other items.

You could customize this widget by adding widgets for loyalty information, price overrides and split shipping.

Note that this widget can be used in both account-based and storefront environments.

Widget Name: `agentAcountOrderDetails`

Display Name: Order Details - Agent

Supported Page Types: `agentOrderDetails` page

Layouts: Order Details Layout - Agent

Elements: To understand how these elements are used, refer to the widget's code view:

- `add-to-purchase-list`
- `agent-email-order-details`
- `agent-order-payment-details`
- `agent-order-price-details`
- `agent-order-refresh`
- `agent-order-summary`
- `agent-promotion-summary`
- `copy-order`
- `dynamic-property`
- `return-order`
- `scheduled-order-actions`
- `scheduled-order-executionList`
- `scheduled-order-instruction`
- `shopping-cart-details`

Add an image to an account detail page example

The following is an example of a customization that you could make to the account detail page. This example shows you how to add an image of a company logo to the account's page.

Note that the example code in this section is for illustrative purposes only; it is not intended to be production-ready, and may not adequately handle all possible use cases or implement the exact behavior you want.

1. Download the `agentAccountOrderDetails` element.
2. Copy the element to change. For example, the `agent-order-summary` element.
3. Edit the template code. Access the `$parent.user().organizations` array can access the `organizationLogo`.
   Note: When editing this code, ensure that you have a check for an array index. The `organizationLogo` is returned in response to the `getOrganization` endpoint. Organization details are stored in the user view model by default, which can be accessed from the widget using `widget.user()`.
4. Upload the element as an extension.
5. Drag and drop the new element and remove the existing `agent-order-summary` element from the **Design** page.
6. Publish the changes.

Display shipping tracking information

The following is an example of a customization that you could make to the account detail page. This example shows you how to display shipping tracking information for each line item of the order, or for each shipping group.

1. Download the `agentAccountOrderDetails` element.
2. Copy the `shopping-carts-details` element.

3. Edit the template code where `trackingInfo` is read from `$data.trackingInfo`.
4. Upload the edited element as an extension.
5. Drag and drop the new element and remove the existing `shopping-carts-details` element from the design studio.
6. Publish the changes.

View account details

This widget allows the agent to view account details of an account-based contact.

This is an account-based-specific widget that allows an agent to see static and dynamic properties of an account.

Widget Name: `agentAccountDetails`

Display Name: Account Details - Agent

Supported Page Types: `agentProfilePageType`

Layouts: Account Details Layout - Agent

Elements: To understand how these elements are used, refer to the widget's code view:

- `account-approval-setting`
- `account-general-info`
- `dynamic-property`
- `registration-request-details`

Manage account contacts

This widget allows the agent to view a list of account contacts.

It allows an agent to view, add, remove or modify account contacts. This is an account-based-specific widget.

Note that a version of this widget exists within the Storefront framework, as well. See Appendix: Layout Widgets and Elements for information on Storefront widgets.

Migrate widgets

If you have recently migrated and this widget may be visible in your widget list. However, the widget is using the default template and .less file, which may not be available to the page layout. Additionally, this widget's style format must be updated so that it appears in the layout after a migration.

To migrate, you must update the following `widget.template` and `.less` files:

- `/layouts/delegatedAdminContactsAgentLayout/widget.template`
- `/less/widget-agentAccountContactsInst_`*`latest_version_number`*`.less`

Widget Name: `delegated-admin-contacts`

Display Name: Account Contacts Widget - Agent

Supported Page Types:

- `agentAccountContactsPageType`
- `agentProfilePageType`
- `profilePageType`

Layouts: Account Contacts Layout - Agent

Elements: None.

Display contact information

This widget displays an interface that allows an agent to navigate through a contact's information.

Widget Name: `secondaryNavigation`

Display Name: Profile Navigation – Account Shoppers

Supported Page Types: All

Layouts:

- Account Contacts Layout – Agent
- Account Details Layout – Agent
- Address Book Layout – Agent
- Customer Details Layout – Agent
- Order History Layout – Agent
- Orders Pending Approval Layout – Agent
- Purchase List Layout – Agent
- Scheduled Order Layout - Agent

Elements: None

Manage account addresses

This widget displays an interface where an agent can manage an account contact's address book.

This interface allows the agent to add, view, delete or update addresses. It can also be used to view, update, delete or manage the shopper's addresses.

Note that a version of this widget exists within the Storefront framework, as well.

Migrate widgets

If you have recently migrated and this widget may be visible in your widget list. However, the widget is using the default template and .less file, which may not be available to the page layout. Additionally, this widget's style format must be updated so that it appears in the layout after a migration..

To migrate, update the following `widget.template` and `.less` files:

- `/layouts/agentAccountAddressesLayout/widget.template`

- `/less/widget-accountAddressesDetailsWidgetAgent_latest_version_number.less`

Widget Name: `agentAccountAddress`

Display Name: Account Address Details - Agent

Supported Page Types:

- `agentProfilePageType`
- `profilePageType`

Layouts: Address Book Layout - Agent

Assign a Delegated Administrator

This widget provides a display that allows the agent to perform administration on an account similar to that of a Delegated Administrator.

Note that a version of this widget exists within the Storefront framework, as well. See Appendix: Layout Widgets and Elements for information on Storefront widgets.

Widget Name: `delegatedAdminContacts`

Display Name: Account Contacts

Supported Page Types:

- `agentProfilePageType`
- `profilePageType`

Layouts: Account Contacts Layout - Agent

Elements: None.

View pending contact registration requests

This widget allows an agent to see a list of pending contact registrations requests.

Note that a version of this widget exists within the Storefront framework, as well. See Appendix: Layout Widgets and Elements for information on Storefront widgets.

Widget Name: `agentContactRegistrationRequests`

Display Name: Contact Registration Requests

Supported Page Types: `agentContactRequestsPageType`

Layouts: Registration Request Listing

Elements: None

Work with scheduled orders

The following widgets provide an interface that allows an agent to view and create scheduled orders for an account.

View a scheduled order list

Note that a version of this widget exists within the Storefront framework, as well. See Appendix: Layout Widgets and Elements for information on Storefront widgets.

Migrate widgets

If you have recently migrated and this widget may be visible in your widget list. However, the widget is using the default template and .less file, which may not be available to the page layout. Additionally, this widget's style format must be updated so that it appears in the layout after a migration.

To migrate, you must update the following `widget.template` and `.less` files:

- `/layouts/agentScheduledOrderListLayout/widget.template`
- `/less/widget-scheduledOrderListInst_agent_`*`latest_version_number`*`.less`

Widget Name: `scheduledOrderList`

Display Name: Scheduled Order Listing Widget - Agent

Supported Page Types:

- `agentScheduledOrderPageType`
- `profilePageType`

Layouts:

- Scheduled Order Layout – Agent

Elements: None

Create a scheduled order

This widget displays an interface that allows an agent to create scheduled orders

Widget Name: `checkoutScheduledOrder`

Display Name: Scheduled Order – Agent Checkout

Supported Page Types:

- `agentCheckout`
- `agentMultiShipCheckout`
- `checkout`

Layouts:

- B2B Checkout Layout – Agent
- Checkout Layout – Agent
- Checkout Layout For Multi Ship - Agent

- Scheduled Orders

Elements: None

Work with quotes

Agents can request quotes for account-based customers. Customizing quote widgets allows you to configure this process.

Create a request for a quote

This widget creates an interface that allows the agent to create either an order or a request for a quote. This interface also enables an agent to disable payment, and hides the place order button and displays the “request for quote” text.

Migrate widgets

If you have recently migrated and this widget may be visible in your widget list. However, the widget is using the default template and .less file, which may not be available to the page layout. Additionally, this widget's style format must be updated so that it appears in the layout after a migration.

You must migrate the following `widget.template` file and `.less` files:

- `/layouts/requestQuoteWidgetAgentLayout/widget.template`
- `/less/widget-requestQuoteInst_latest_version_number.less`

Note that a version of this widget exists within the Storefront framework, as well. See Appendix: Layout Widgets and Elements for information on Storefront widgets.

Widget Name: `requestQuote`

Display Name: Request Quote - Agent

Supported Page Types: All

Layouts:

- B2B Checkout Layout - Agent
- B2B Edit Layout - Agent
- Checkout Edit Layout For Multi Ship - Agent
- Checkout Layout - Agent
- Checkout Layout For Multi Ship - Agent

Elements: To understand how these elements are used, refer to the widget’s code view:

- `quote-notes-history`
- `requester-notes-text-area`
- `request-quote-button`

View quote details

This widget provides an interface that allows the agent to review a quoted order, accept or reject quotes and to re-request quotes.

Note that a version of this widget exists within the Storefront framework, as well. See Appendix: Layout Widgets and Elements for information on Storefront widgets.

Widget Name: `agentQuoteOrderDetails`

Display Name: Quote Order Details - Agent

Supported Page Types: `agentOrderDetails` page

Layouts: Order Details Layout - Agent

Elements: To understand how these elements are used, refer to the widget's code view:

- `agent-accept-quote-button`
- `agent-add-note-button`
- `agent-notes-history-text-area`
- `agent-quote-note-text-area`
- `agent-reject-quote-button`
- `agent-request-requote-button`

Work with approvals

The following widgets can be used by an agent with the correct role to access layouts that allow them to approve or review pending approvals.

Approve an order

This widget provides an interface that can be used by an agent with the correct role to approve an account-based order in the Pending Approval state. This interface also shows, and allows the agent to add to, the approval comments associated with an order.

Note that a version of this widget exists within the Storefront framework, as well. See Appendix: Layout Widgets and Elements for information on Storefront widgets.

Widget Name: `agentOrderApproval`

Display Name: Order Approval - Agent

Supported Page Types: `agentOrderDetails`

Layouts: Order Details Layout - Agent

Elements: None

View orders pending approval

Agents can view orders that are pending approval if they are members of the Approval role. This widget lists the pending orders and approval reasons, and allows the agent to approve or reject the request.

Note that a version of this widget exists within the Storefront framework, as well. See Appendix: Layout Widgets and Elements for information on Storefront widgets.

If you have recently migrated and this widget may be visible in your widget list. However, the widget is using the default template and .less file, which may not be

available to the page layout. Additionally, this widget's style format must be updated so that it appears in the layout after a migration.

To migrate, you must update the following `widget.template` and `.less` files:

- `/layouts/agentOrderPendingApprovalsLayout/widget.template`
- `/less/widget-orderPendingApprovalDetailsWidgetAgent_`*`latest_version_number`*`.less`

Note: The agent can no longer select the Check for approval button from the administrative interface when creating or editing an order. To implement this functionality, use the REST API.

Widget Name: `ordersPendingApproval`

Display Name: Orders Pending Approval

Supported Page Types: `agentOrdersPendingApproval`

Layouts: Orders Pending Approval Layout - Agent

Elements: None

15

Understand Agent-Based Layouts

This section describes agent-based layouts. Layouts are what comprise the pages that are displayed to the agent.

Each individual layout represents a various page, and act as a template for the agent's console. Additionally, layouts contain the widgets that allow you to define the page structure. Layouts can contain more than one widget, with each widget containing specific functionality. These widgets combined to for the page layout.

Note: Always ensure that you are using the latest version of the widgets and the layouts.

Using the Design page, you can view and customize layouts to suit your environment. When you click the design page, the default Layout page is displayed, which contains information on each layout used by the agent console. You can drag components from the component pane onto layouts, customizing each page.

Important: While you can work with and create customized layouts, the Preview function does not work with agent-based layouts.

Use the layout's configuration to customize the agent console, and extend the functionality of agent-specific widgets.

For detailed information on working with layouts, see Design Your Store Layout.

Agent-specific page layouts

The following layouts have been created for agent console widgets:

Layout	Associated Widgets
Account Contacts Layout - Agent	Navigation – Agent, Notifications Widget – Agent, Customer Summary Widget, Profile Navigation – Account Shoppers, Account Contacts Widget – Agent
Account Details Layout - Agent	Navigation – Agent, Notifications Widget – Agent, Customer Summary Widget, Profile Navigation – Account Shoppers, Account Details – Agent
Address Book Layout -Agent	Navigation – Agent, Notifications Widget – Agent, Customer Summary Widget, Profile Navigation – Account Shoppers, Account Address Details – Agent
Checkout Edit Layout - Agent	Navigation – Agent, Notifications Widget – Agent, Create Order Header B2B Layout, Checkout Order Details, Checkout Order Summary – Agent, Search And Add Items To Cart, Shopping Cart – Agent, Promotions Widget – Agent, Address Book for B2C Customer – Agent, Cart Shipping – Agent, Loyalty Payment, Loyalty Details, Split Payments – Agent, Notes Widget – Agent, Place Order – Agent Checkout

Layout	Associated Widgets
Checkout Edit Layout For Multi Ship - Agent	Navigation – Agent, Notifications Widget – Agent, Create Order Header B2B Layout, Checkout Order Details, Checkout Order Summary – Agent, Shipping Options - Agent, Promotion Widget – Agent, Loyalty Payment Loyalty Details, Split Payments – Agent, Notes Widget – Agent, Request Quote Widget – Agent, Place Order – Agent Checkout
Checkout Layout - Agent	Navigation – Agent, Notifications Widget – Agent, Header – Agent Create Order, Checkout Order Details, Checkout Order Summary – Agent, Search And Add Items To Cart, Shopping Cart – Agent, Promotion Widget – Agent, Scheduled Order – Agent Checkout, Address Book for B2C Customer – Agent, Cart Shipping – Agent, Loyalty Payment, Loyalty Details, Split Payments – Agent, Notes Widget – Agent, Request Quote Widget – Agent, Place Order – Agent Checkout
Checkout Layout For Multi Ship - Agent	Navigation – Agent, Notifications Widget – Agent, Create Order Header B2B Layout, Checkout Order Details, Checkout Order Summary – Agent, Shipping Options – Agent, Promotion Widget – Agent, Scheduled Order – Agent Checkout, Loyalty Payment, Loyalty Details, Split Payments – Agent, Notes Widget – Agent, Request Quote Widget – Agent, Place Order – Agent Checkout
Checkout Layout for Pending Payment - Agent	Navigation – Agent, Create Order Header B2B Layout, Order Details for Pending Payment – Agent, Loyalty Payment, Loyalty Details, Split Payments – Agent, Notes Widget – Agent, Place Order – Agent Checkout
Collection Layout - Agent	Navigation – Agent, Notifications Widget – Agent, Overlayed Guided Navigation, Header – Agent Catalog, Collection Navigation Widget, Product Listing Widget - Agent
Create Exchange Request Layout - Agent	Navigation – Agent, Notifications Widget – Agent, Create Exchange – Agent
Create Return Layout - Agent	Navigation – Agent, Notifications Widget – Agent, Create Return – Agent
Customer Details Layout - Agent	Navigation – Agent, Notifications Widget – Agent, Customer Summary Widget, Profile Navigation – Account Shoppers, Customer Profile Details – Agent
Customer Search Layout - Agent	Navigation – Agent, Notifications Widget – Agent, Customer Search and Self Registration – secondary Navigation, Customer Registration – Agent, Customer Search - Agent
Home Layout - Agent	Navigation – Agent, Dashboard - Agent
No Search Results Layout - Agent	Navigation – Agent, Notifications Widget – Agent, Overlayed Guided Navigation, Header – Agent Catalog, Collection Navigation Widget, No Search Results - Agent

Layout	Associated Widgets
Order Details Layout - Agent	Navigation – Agent, Notifications Widget – Agent, Header – Agent Order Details, Order Details – Agent, Return History – Agent, Exchange History – Agent, Order Approval – Agent, Quote Order Details – Agent, Notes Widget - Agent
Order History Layout - Agent	Navigation – Agent, Notifications Widget – Agent, Customer Summary Widget, Profile Navigation – Account Shoppers, Order History Widget - Agent
Order Search Layout - Agent	Navigation – Agent, Order Search - Agent
Orders Pending Approval Layout - Agent	Navigation – Agent, Notifications Widget – Agent, Customer Summary Widget, Profile Navigation – Account Shoppers, Orders Pending Approval Details – Agent
Payer Authentication Layout	CyberSource Payment Authorization
Process Returns Layout - Agent	Navigation – Agent, Notifications Widget – Agent, Process Returns - Agent
Product Layout - Agent	Navigation – Agent, Notifications Widget – Agent, Header – Agent Catalog, Collection Navigation Widget, Product Details - Agent
Purchase List Layout - Agent	Navigation – Agent, Notifications Widget – Agent, Customer Summary Widget, Profile Navigation – Account Shoppers, Purchase List
Refunds Layout - Agent	Navigation – Agent, Notifications Widget – Agent, Refunds - Agent
Registration Request Detail Layout - Agent	Navigation – Agent, Notifications Widget – Agent, Self Registration Detail
Return Search Layout - Agent	Navigation – Agent, Return Search - Agent
Scheduled Order Layout - Agent	Navigation – Agent, Notifications Widget – Agent, Customer Summary Widget, Profile Navigation – Account Shoppers, Scheduled Orders Listing Widget –Agent
Search Results Layout - Agent	Navigation – Agent, Notifications Widget – Agent, Overlayed Guided Navigation, Header – Agent Catalog, Collection Navigation Widget, Search Results Widget - Agent
Self Registration Layout - Agent	Navigation – Agent, Notifications Widget – Agent, Customer Search and Self Registration – secondary navigation, Registration Request Search
View Return Request Layout - Agent	Navigation – Agent, Notifications Widget – Agent, View Return Request – Agent

Create a new layout

You can create a new layout by cloning an existing layout and them making modifications to the copy. This allows you to configure things like site settings, notes, viewports and other information.

1. After accessing the **Design** page, click the **Layouts** tab.
2. Highlight the layout that you want to clone and click the **Clone Layout** option from the toolbar.
3. Enter the new layout's name and other information.

4. Click **Save** to confirm all of the new settings.

Once you have created a new layout, you can add widgets or other components to it as necessary. For detailed information on creating and customizing new layouts, see Design Your Store Layout.

Note: When creating a new layout, make sure that the roles are set correctly. To begin using the new copy, you must delete the roles from the old copy of the layout and add them to the new layout.

16

Use Agent Themes

You can use an agent-specific theme when creating your application.

A theme is made up of a number of LESS files that are then compiled into a set of CSS files. These themes contain pre-configured settings that assist you in creating the look and feel of your site. You can have a number of different themes for your sites. For each active theme, there is a set of CSS files.

Note: If the storefront CSS files have been updated or added to a theme, ensure that you have also update the agent files have also been updated.

Your agent application uses its own specific agent theme. Because there can only be one agent theme, a configuration property lets you specify the theme ID to use for the agent application.

The agent theme is listed on the Theme tab of the Theme Manager page. However, note the following differences from the storefront themes:

- There can be only one agent theme.
- Although they are displayed, the agent theme cannot be edited using the Theme Manager page in the administration interface.
- Agent themes cannot use the Go to Theme Code button in the administration interface.
- Agent themes are not listed in the dropdown for themes in Site Settings in the administration interface.

The agent theme is stored in the `AdminPageRepoistory` and uses the endpoints listed below. Set the `AgentTheme` flag in the repository to `true` to use an agent theme.

For information on working with storefront themes, see Customize Your Store's Design Theme.

Work with themes

The Agent REST API contains theme-based endpoints that allow you to perform various functions.

For detailed information on these endpoints, refer to the Agent REST API documentation.

Get all themes

Stores can use many themes; however, the agent console can only use a single theme. You can review all of the themes available and make modifications as necessary. To see all themes, use the Themes API:

Issue a `GET` command to get all themes, including the agent theme:

```
/ccadmin/v1/themes?includeagenttheme=true
```

Note: If the `includeagenttheme` query parameter is set to `false` or omitted, the endpoint will not return the agent theme.

Get the active agent theme

An active theme is a theme that is currently in use. You can issue a `GET` command to get the active agent theme:

```
/ccadmin/v1/themes/agentThemeDetails
```

The response may be something similar to this:


```
{
  "isAgentTheme": true,
  "thumbnail": "",
  "theme_additional_fonts": {},
  "notes": "This is the Agent Theme.",
  "is_active": false,
  "theme_styles_color": {
    "@ButtonPrimarySize": "medium",
    "@ButtonSecondaryTextDecoration": "none",
    "@ButtonPrimaryTextColor": "#ffffff",
    "@SubNavigationTextColor": "#6F7178",
    "@ButtonPrimaryUseGradient": "false",
    "@ButtonPrimaryFontStyle": "normal",
    "@ButtonSecondaryFontFamily": "@sansFontFamily",
    "@ButtonPrimaryFontWeight": "normal",
    "@NavbarLinkColor": "#3D3D3D",
    "@NavbarLinkHoverColor": "#195D8D",
    "@NavbarTextColor": "#333333",
    "@mobileNavBarButtonColor": "#333333",
    "@SubNavigationLinkHoverColor": "#195D8D",
    "@NavbarBackgroundHoverColor": "#FFFFFF",
    "@SitePageBorderColor": "#CCD7DF",
    "@ButtonSecondaryUseGradient": "false",
    "@LinkVisitedColor": "#195d8d",
    "@LinkColor": "#195d8d",
    "@ButtonPrimaryBackgroundColor": "#0572ce",
    "@NavbarBackgroundColor": "#FFFFFF",
    "@SubNavigationLinkColor": "#3D3D3D",
    "@ButtonPrimaryFontFamily": "@sansFontFamily",
    "@ButtonPrimaryTextDecoration": "none",
    "@ButtonSecondaryBorderRadius": "4px",
    "@ButtonSecondaryFontStyle": "normal",
    "@ButtonSecondaryTextColor": "#3d3d3d",
    "@SubNavigationBackgroundColor": "#FFFFFF",
    "@ButtonSecondaryFontWeight": "normal",
    "@ButtonSecondaryBackgroundColor": "#ffffff",
    "@ButtonPrimaryBorderRadius": "4px",
    "@LinkHoverColor": "#114062",
    "@ButtonSecondarySize": "medium",
    "@SubNavigationBackgroundHoverColor": "#FFFFFF",
    "@TextColor": "#000000"
  },
```

```
"usingCodeView": false,
"is_default": false,
"associatedSites": [],
"theme_styles_typography": {
  "@H6LineHeight": "150%",
  "@H5TextDecoration": "inherit",
  "@H3FontFamily": "inherit",
  "@H5FontStyle": "inherit",
  "@H2FontStyle": "inherit",
  "@H2TextColor": "inherit",
  "@SiteLineHeight": "150%",
  "@SiteFontFamily": "@sansFontFamily",
  "@H1TextAlign": "inherit",
  "@ParagraphFontWeight": "normal",
  "@H5FontSize": "1.00rem",
  "@H4TextAlign": "inherit",
  "@H1TextDecoration": "inherit",
  "@H3LineHeight": "150%",
  "@H2FontWeight": "bold",
  "@H3FontSize": "1.75rem",
  "@ParagraphLineHeight": "150%",
  "@H3TextColor": "inherit",
  "@H3TextAlign": "inherit",
  "@H4FontFamily": "inherit",
  "@SiteFontSize": "14px",
  "@SiteTextAlign": "left",
  "@H5FontWeight": "normal",
  "@H6FontFamily": "inherit",
  "@H1FontSize": "2.75rem",
  "@H6FontStyle": "inherit",
  "@H6TextDecoration": "inherit",
  "@H2TextDecoration": "inherit",
  "@ParagraphTextColor": "inherit",
  "@H1FontStyle": "inherit",
  "@ParagraphTextAlign": "inherit",
  "@H4LineHeight": "150%",
  "@H1FontWeight": "bold",
  "@SiteTextColor": "#000000",
  "@ParagraphFontSize": "1.00rem",
  "@H5FontFamily": "inherit",
  "@H4FontWeight": "bold",
  "@ParagraphFontFamily": "inherit",
  "@H1FontFamily": "inherit",
  "@H2TextAlign": "inherit",
  "@H1LineHeight": "150%",
  "@H6FontSize": "0.85rem",
  "@H3TextDecoration": "inherit",
  "@H4TextColor": "inherit",
  "@ParagraphFontStyle": "inherit",
  "@ParagraphTextDecoration": "inherit",
  "@H1TextColor": "inherit",
  "@SiteTextDecoration": "none",
  "@H4FontSize": "1.25rem",
  "@SiteFontStyle": "normal",
  "@H3FontWeight": "bold",
```

```
    "@H6TextAlign": "inherit",
    "@H4TextDecoration": "inherit",
    "@H2FontFamily": "inherit",
    "@H3FontStyle": "inherit",
    "@SiteFontWeight": "normal",
    "@H5LineHeight": "150%",
    "@H2FontSize": "2.25rem",
    "@H6TextColor": "inherit",
    "@H4FontStyle": "inherit",
    "@H2LineHeight": "150%",
    "@H5TextColor": "inherit",
    "@H5TextAlign": "inherit",
    "@H6FontWeight": "normal"
}
```

Clone a theme

To create a new theme, clone an existing theme.

1. Issue a `POST` command to:

   ```
   /ccadmin/v1/themes/<name of theme to clone>/clone
   {name: <name of new theme>}
   ```

   For example:

   ```
   POST /ccadmin/v1/themes/PrimarySiteTheme/clone
   {name:"SecondarySiteTheme"}
   ```

2. Make the modifications to the theme as necessary.
3. To set the theme as the active agent theme, issue a `POST` command to:

   ```
   /ccadmin/v1/themes/{id}/setAsAgentTheme
   ```

   For example:

   ```
   POST /ccadmin/v1/themes/SecondarySiteTheme/setAsAgentTheme
   ```

Important: Once you have cloned a theme, you must use the `setAsAgentTheme` endpoint to indicate which theme is the assigned theme. This ensures that you do not have multiple agent themes.

Compile a theme

Once you have created a new agent theme and identified it as such, you compile it.

Issue a `POST` command to:

```
/ccadmin/v1/themes/compileAgentTheme
```

For example:

```
POST /ccadmin/v1/themes/compileAgentTheme
```

Update theme details

To update the agent theme's detail, such as the theme name, notes or details about the theme, do the following:

1. Issue a `GET` command to:

   ```
   /ccadmin/v1/themes/agentThemeDetails
   ```

2. This returns the theme's details. Update the theme elements as necessary.
3. Once you have finished making edits, issue a `PUT` command, with the updated JSON values, to:

   ```
   /ccadmin/v1/themes/agentThemeDetails
   ```

   For example:

   ```
   POST /ccadmin/v1/themes/agentThemeDetails
   {"assetType", "pageLayout", "assetId", "404PageLayout"}
   ```

Update the theme source

1. Issue `GET` command to:

   ```
   /ccadmin/v1/themes/agentThemeSource
   ```

2. Make the modifications as necessary.
3. To reload the theme, issue a `PUT` command with the updated JSON values to:

   ```
   /ccadmin/v1/themes/agentThemeSource
   ```