



# Extend Integration for Salesforce Commerce Cloud

Version 20.1.0



## Table of Contents

<b>1. Summary .....</b>	<b>3</b>
<b>2. Component Overview .....</b>	<b>3</b>
<i>Functional Overview .....</i>	<i>3</i>
<i>Use Cases .....</i>	<i>3</i>
<i>Compatibility.....</i>	<i>3</i>
<b>3. Implementation Guide.....</b>	<b>3</b>
<i>Setup of Business Manager.....</i>	<i>4</i>
<i>Configuration .....</i>	<i>5</i>
<i>Custom Code .....</i>	<i>5</i>
<i>External Interfaces .....</i>	<i>19</i>
<i>Firewall Requirements .....</i>	<i>20</i>
<b>4. Testing.....</b>	<b>21</b>
<b>5. Operations, Maintenance .....</b>	<b>26</b>
<i>Support.....</i>	<i>26</i>

## 1. Summary

Extend provides an easy way for any merchant to offer extended warranties - generating new revenue, increasing purchase conversion, and dramatically improving the customer experience.

This implementation guide describes how to integrate Extend Warranty into SFCC SiteGenesis v20.1.0.

## 2. Component Overview

### Functional Overview

This integration cartridge makes it easy to enable the following features:

- Daily export catalog products to Extend
- Purchase extended warranties for eligible products
- Present offers and an up-sell modal in Product Display Page (PDP) and Cart pages for eligible products
- Generate contracts for newly purchased Extend warranties

### Use Cases

For a merchant, the main use cases are:

- Possibility to enable/disable the main functionalities for Extend warranties in Business Manager
- Display warranty products in PDP & Cart
- Display an up-sell modal in PDP & Cart
- Generate warranty contracts upon order placement

For a customer, the main use cases are:

- Attach an extended warranty to a catalog product, in PDP, Modal & Cart and subsequently checkout products
- View details about the warranty products
- Remove warranties from the Cart (either by removing the main product or just the warranty product)

### Compatibility

This cartridge has been developed and tested against Commerce Cloud Digital 20.1.0 and integrated on top of SiteGenesis 104.1.5.

### Privacy Policies on Payments

This cartridge implements a custom object for temporary storing order data needed for Extend warranty contract generation. This custom object (*ExtendContractsQueue*) was implemented based on a queue pattern that gets consumed by a job configured to execute every 5 minutes. This data is not exposed publicly and used only in a server-to-server communication setup.

Extend does not handle payments from the end-user. The merchant is responsible for charging the end-user for the warranty plans purchased, using the existing tenders that are currently supported on the specific environment. The merchant is responsible for generate Extend contracts only after charging the end-user for the warranty plans purchased. Please visit to <https://extend.com/privacy> for more information.

### 3. Implementation Guide

The SiteGenesis version of this integration follows the best practices for this storefront version. In order to achieve the features described in this document, you will need to manually insert the code indicated below.

### 3.1 Setup of Business Manager

### 3.1.1 Upload cartridge & include in the cartridge path

1. Upload the **int\_extend** cartridge to the sandbox.
2. Go to Business Manager → Administration → Site → Manage Sites. Select desired site and go to Settings. At the beginning of the Cartridge Path add “int\_extend:” in order to include the cartridge in the current site’s configuration.

### 3.1.2 Import metadata

1. Go to Business Manager → Administration → Site Development → Import & Export
2. Upload the following files from `./metadata/meta/` folder
  - a. `system-objecttype-extensions.xml`
  - b. `custom-objecttype-definitions.xml`
3. Go to Business Manager → Administration → Site Development → Import & Export → Meta Data → Import. You should see both previous files as available for import. Proceed to importing both.

### 3.1.3 Import custom jobs

1. Go to Business Manager → Administration → Operations → Import & Export
2. Upload the following file: `./metadata/jobs.xml`
3. Go to Business Manager → Administration → Operations → Import & Export → Jobs → Import. You should see the previous file as available for import. Proceed to importing it.

### 3.1.4 Import service

1. Go to Business Manager → Administration → Operations → Import & Export
2. Upload the following file: `./metadata/services.xml`
3. Go to Business Manager → Administration → Operations → Import & Export → Services → Import. You should see the previous file as available for import. Proceed to importing it.
4. Note: by default, the service URL is configured to point to the demo instance of the Extend API. Prior to going live with the integration, this will need to be changed and tested against the live production API.

### 3.1.5 Import Extend catalog

1. Select the site you're integrating Extend with
2. Go to Business Manager → Merchant Tools → Products and Catalogs → Import & Export.
3. Upload the following file: `./metadata/catalog.xml`. Before uploading replace the catalog-id 'apparel-m-catalog' with master catalog id of your site.

```
<?xml version="1.0" encoding="UTF-8"?>  
<catalog xmlns="http://www.demandware.com/xml/impex/catalog/2006-10-31" catalog-id="apparel-m-catalog">  
  <header>  
    <image-settings>  
      <internal-location base-path="/images"/>  
      <view-types>  
        <view-type>large</view-type>  
        <view-type>medium</view-type>  
        <view-type>small</view-type>  
        <view-type>swatch</view-type>  
        <view-type>hi-res</view-type>
```

4. Then find and replace the catalog-id in classification-category element in catalog.xml. Do a bulk find and replace of 'storefront-catalog-m-en' with the storefront catalog id of the site on which Extend will be integrated.

```

</options>
<classification-category catalog-id="storefront-catalog-m-en">root</classification-category>
<pinterest-enabled-flag>false</pinterest-enabled-flag>
<facebook-enabled-flag>false</facebook-enabled-flag>
<store-attributes>
  <force-price-flag>false</force-price-flag>
  <non-inventory-flag>false</non-inventory-flag>
  <non-revenue-flag>false</non-revenue-flag>
  <non-discountable-flag>false</non-discountable-flag>
</store-attributes>
</product>

```

This will import 3 dummy option products, which are used in order to create dynamic product line items in cart. We recommend keeping this as a separate master catalog, for visibility purposes.

## 3.2 Configuration

### 3.2.1 Catalog setup

- The 3 Extend products that need configuration are:
  - EXTEND-12
  - EXTEND-24
  - EXTEND-36
- In order to make the Extend products merchandisable, you need to:
  - Assign them to your storefront catalog
  - Allocate an inventory record in the inventory file assigned to the current site
    - Configure this entry as perpetual, since these are intangible goods and need to always be in-stock
- These don't need any description, pricing or any other regular product attributes to be configured

### 3.2.2 Custom preferences

1. Go to Business Manager → Merchant Tools → Site Preferences → Custom Preferences → Extend
2. Configure the provided Extend credentials in the following fields
  - a. Access Token: this is used for authenticating web service calls to Extend's API
  - b. Store ID: this is used for the client-side Javascript SDK to identify the current store
3. Configure the image breakpoint for using when exporting the product feed
  - a. The Image View Type is by default configured to *large*
  - b. Configure this to specify the breakpoint for the largest image size you have available across your products
4. Extend Javascript SDK URL  
By default, this points to the demo JS SDK. Make sure that by the end of the integration process this is switched to the live production version.
5. Use the provided feature switches in order to enable / disable Extend features
  - a. Enable Extend  
Enables / disables Extend globally
  - b. Enable Up-sell in Cart  
Enables / disables the up-sell button and modal in Cart
  - c. Enable in PDP  
Enables / disables Extend in PDP
  - d. Enable Up-sell in PDP  
Enables / disables the up-sell modal in PDP

## 3.3 Jobs

### 3.3.1 Extend Products Export

This job uses Extend's API to traverse the entire storefront catalog and export each Simple and Variant Product that is, at the time of the job run, merchandisable (online and in stock).

1. Go to Business Manager → Operations → Jobs. Select the Extend Products Export job.
2. Go to Job Steps and select Scope. Configure this to run in the scope of the site you're implementing Extend for.
3. Schedule this to run once a day, at a time that's most suitable for your recurring job setup, ideally after the catalog is being updated.
4. Make sure you're running this once and notify Extend after the successful run. Warranty products or plans are dynamically generated based on this stream of products.

### 3.3.2 Extend Contracts Creation

This job uses Extend's API to consume contracts queue, stored in the ExtendContractsQueue custom object. As Extend plans are being ordered, an instance of this object is being created. Thereafter, this job will send a contract generation request for each queue element and then it will remove it from the queue.

## 3.3 Custom Code

### 3.3.1 Include extend\_footer.isml

- In: \*/app\_storefront\_core/cartridge/templates/default/components/footer/footer\_UI.isml
- Where: at the end of the file

**Code:**

```
<iscomment>Extend Integration</iscomment>  
<isinclude template="components/footer/extend_footer"/>
```

```

1  <iscontent type="text/html" charset="UTF-8" compact="true"/>
2
3  <!--[if gte IE 9 | !IE]><!-->
4  | <script src="${URLUtils.staticURL('/lib/jquery/jquery-2.1.1.min.js')}" type="text/javascript"></script>
5  <!--<![endif]-->
6
7  <!--[if lte IE 8]>
8  | <script src="${URLUtils.staticURL('/lib/jquery/jquery-1.11.1.min.js')}" type="text/javascript"></script>
9  | <script src="//cdn.rawgit.com/weblinc/media-match/master/media.match.min.js" type="text/javascript"></script>
10 | <script src="//cdnjs.cloudflare.com/ajax/libs/es5-shim/3.4.0/es5-shim.min.js"></script>
11 <![endif]-->
12
13 <!--[if IE 9]>
14 | <script src="//cdn.rawgit.com/paulirish/matchMedia.js/master/matchMedia.js" type="text/javascript"></script>
15 | <script src="//cdn.rawgit.com/paulirish/matchMedia.js/master/matchMedia.addListener.js" type="text/javascript"></script>
16 <![endif]-->
17
18 <script src="${URLUtils.staticURL('/lib/jquery/ui/jquery-ui.min.js')}" type="text/javascript"></script>
19
20 <iscomment>third-party add-ons</iscomment>
21 <script src="${URLUtils.staticURL('/lib/jquery/jquery.jcarousel.min.js')}" type="text/javascript"></script>
22 <script src="${URLUtils.staticURL('/lib/jquery/jquery.validate.min.js')}" type="text/javascript"></script>
23 <script src="${URLUtils.staticURL('/lib/jquery/jquery.zoom.min.js')}" ></script>
24 <script type="text/javascript"><isinclude template="resources/appresources"/></script>
25 <script type="text/javascript"><isinclude url="${URLUtils.url('Resources-LoadTransient')}"></script>
26 <script>var consent = ${session.custom.consentTracking};</script>
27 <script src="${URLUtils.staticURL('/js/app.js')}"></script>
28 <isif condition="${(!('pageContext' in this) || empty(pageContext))}">
29 | <iscript>pageContext = new Object();</iscript>
30 </isif>
31 <script>pageContext = <isprint value="${JSON.stringify(pageContext)}" encoding="off"/></script>
32 <script>
33 var meta = "${pdict.CurrentPageMetaData.description}";
34 var keywords = "${pdict.CurrentPageMetaData.keywords}";
35 </script>
36
37 <iscomment>Extend Integration</iscomment>
38 <isinclude template="components/footer/extend_footer"/>

```

### 3.3.2 Include extend\_productcontent.isml

- In: \*/app\_storefront\_core/cartridge/templates/default/product/productcontent.isml
- Where: after the availability section, before the quantity section

#### Code:

```
<iscomment>
Extend Integration
=====
</iscomment>
<include template="product/extend_productcontent" />
```

```
137      <div class="availability-instore">
138          <i class="fa fa-briefcase fa-lg pull-left"></i>
139          <label for="Stock">${Resource.msg('product.instorestock','product',null)}</label>
140          <isif condition="${empty(pdct.CurrentHttpParameterMap.uuid.value)}">
141              <div id="${pdct.Product.ID}" class="availability-results availability-msg store-stock">
142                  <span class="label set-preferred-store"><a href="${URLUtils.url('StoreInventory-SetZipCodeCore'
143                  </div>
144              <iselse/>
145              <div id="${pdct.CurrentHttpParameterMap.uuid.value}" class="availability-results store-stock"></div>
146          </isif>
147      </div>
148  </div>
149  <iselse/>
150      <div class="availability-web">
151          <label for="Stock">${Resource.msg('global.availability','locale',null)}</label>
152          <isif condition="${!pdct.Product.master && !pdct.Product.variationGroup}">
153              <span class="value"><include template="product/components/availability"/></span>
154          <iselse/>
155              <div class="availability-novariation">${Resource.msg('product.selectforstock','product',null)}</div>
156          </isif>
157      </div>
158  </isif>
159
160  <iscomment>
161      Extend Integration
162      =====
163  </iscomment>
164  <include template="product/extend_productcontent" />
165
166  <iscomment>
167      product quantity
168      =====
169  </iscomment>
170
```



### 3.3.3 Include PDP functionality

- In: \*/app\_storefront\_core/cartridge/js/pages/product/variant.js
- Where: function updateContent(), at the end of the \$.ajax() callback function

#### Code:

```
if (window.EXT_GLOBAL_SWITCH) {  
    window.extendPDP();  
}
```

```
1  'use strict';  
2  
3  var ajax = require('../ajax'),  
4      image = require('./image'),  
5      progress = require('../progress'),  
6      productStoreInventory = require('../storeinventory/product'),  
7      tooltip = require('../tooltip'),  
8      util = require('../util');  
9  
10  
11  /**  
12   * @description update product content with new variant from href, load new content to #product-content panel  
13   * @param {String} href - url of the new product variant  
14   */  
15  var updateContent = function (href) {  
16      var $pdpForm = $('#pdpForm');  
17      var qty = $pdpForm.find('input[name="Quantity"]').first().val();  
18      var params = {  
19          Quantity: isNaN(qty) ? '1' : qty,  
20          format: 'ajax',  
21          productListid: $pdpForm.find('input[name="productlistid"]').first().val()  
22      };  
23  
24      progress.show($('#pdpMain'));  
25  
26      ajax.load({  
27          url: util.appendParamsToUrl(href, params),  
28          target: $('#product-content'),  
29          callback: function () {  
30              if (SitePreferences.STORE_PICKUP) {  
31                  productStoreInventory.init();  
32              }  
33              image.replaceImages();  
34              tooltip.init();  
35              if (window.EXT_GLOBAL_SWITCH) {  
36                  window.extendPDP();  
37              }  
38          }  
39      });  
40  };  
41
```

### 3.3.4 Include Upsell / cross-sell modal functionality

- In: \*/app\_storefront\_core/cartridge/js/pages/product/addToCart.js
- Where: function addToCart(), after the \$form variable declaration and before addItemToCart()

#### Code:

// Extend Integration

```
if ($('#extend-offer').length) {  
    extendAddToCart($form, page, minicart, dialog, addItemToCart);  
    return;  
}
```

```
35  /**  
36  * @description Handler to handle the add to cart event  
37  */  
38  var addToCart = function (e) {  
39      e.preventDefault();  
40      var $form = $(this).closest('form');  
41  
42      // Extend Integration  
43      if ($('#extend-offer').length) {  
44          extendAddToCart($form, page, minicart, dialog, addItemToCart);  
45          return;  
46      }  
47  
48      addItemToCart($form).then(function (response) {  
49          var $uuid = $form.find('input[name="uuid"]');  
50          if ($uuid.length > 0 && $uuid.val().length > 0) {  
51              page.refresh();  
52          } else {  
53              // do not close quickview if adding individual item that is part of product set  
54              // @TODO should notify the user some other way that the add action has completed successfully  
55              if (!$this.hasClass('sub-product-item')) {  
56                  dialog.close();  
57              }  
58              minicart.show(response);  
59          }  
60      }).bind(this));  
61  };
```

### 3.3.5 Import extendHelpers.js in CartModel.js

- In: \*/app\_storefront\_controllers/cartridge/scripts/models/CartModel.js
- Where: include this script at the API Includes section

#### Code:

// Import Extend Script Helpers

var extendHelpers = require('\*/cartridge/scripts/extendHelpers');

```
1  'use strict';
2  /**
3   * Model for cart functionality. Creates a CartModel class with payment, shipping, and product
4   * helper methods.
5   * @module models/CartModel
6   */
7  var Transaction = require('dw/system/Transaction');
8
9  /* API Includes */
10 var AbstractModel = require('./AbstractModel');
11 var ArrayList = require('dw/util/ArrayList');
12 var BasketMgr = require('dw/order/BasketMgr');
13 var Money = require('dw/value/Money');
14 var MultiShippingLogger = dw.system.Logger.getLogger('multishipping');
15 var OrderMgr = require('dw/order/OrderMgr');
16 var PaymentInstrument = require('dw/order/PaymentInstrument');
17 var Product = require('~/cartridge/scripts/models/ProductModel');
18 var ProductInventoryMgr = require('dw/catalog/ProductInventoryMgr');
19 var ProductListMgr = require('dw/customer/ProductListMgr');
20 var QuantityLineItem = require('~/cartridge/scripts/models/QuantityLineItemModel');
21 var Resource = require('dw/web/Resource');
22 var ShippingMgr = require('dw/order/ShippingMgr');
23 var StoreMgr = require('dw/catalog/StoreMgr');
24 var TransientAddress = require('~/cartridge/scripts/models/TransientAddressModel');
25 var UUIDUtils = require('dw/util/UUIDUtils');
26
27 var lineItem;
28 var app = require('~/cartridge/scripts/app');
29 var ProductList = app.getModel('ProductList');
30
31 // Import Extend Helpers
32 var extendHelpers = require('*/cartridge/scripts/extendHelpers');
```

### 3.3.6 Include Extend line item creation in CartModel.js

- In: \*/app\_storefront\_controllers/cartridge/scripts/models/CartModel.js
- Where: function addProductToCart(), before the return statement

#### Code:

// Extend Create Warranty LineItem

extendHelpers.createOrUpdateExtendLineItem(cart, params, Product);

```
106     // Adds a product.
107   } else {
108     var previousBonusDiscountLineItems = cart.getBonusDiscountLineItems();
109     productToAdd = Product.get(params.pid.stringValue);
110
111     if (productToAdd.object.isProductSet()) {
112       var childPids = params.childPids.stringValue.split(',');
113       var childQtys = params.childQtys.stringValue.split(',');
114       var counter = 0;
115
116       for (var i = 0; i < childPids.length; i++) {
117         var childProduct = Product.get(childPids[i]);
118
119         if (childProduct.object && !childProduct.isProductSet()) {
120           var childProductOptionModel = childProduct.updateOptionSelection(params);
121           cart.addProductItem(childProduct.object, parseInt(childQtys[counter]), childProductOptionModel);
122         }
123         counter++;
124       }
125     } else {
126       productOptionModel = productToAdd.updateOptionSelection(params);
127       cart.addProductItem(productToAdd.object, params.Quantity.doubleValue, productOptionModel);
128     }
129
130     // When adding a new product to the cart, check to see if it has triggered a new bonus discount line item.
131     newBonusDiscountLineItem = cart.getNewBonusDiscountLineItem(previousBonusDiscountLineItems);
132   }
133
134   // Extend Create Warranty LineItem
135   extendHelpers.createOrUpdateExtendLineItem(cart, params, Product);
136
137   return {
138     format: format,
139     template: template,
140     BonusDiscountLineItem: newBonusDiscountLineItem
141   };
142 },
143
```

### 3.3.7 Import extendHelpers.js in Cart

- In: \*/app\_storefront\_controllers/cartridge/controllers/Cart.js
- Where: include this script at the API Includes section

#### Code:

var extendHelpers = require('\*/cartridge/scripts/extendHelpers');

```
1  'use strict';
2
3  /**
4   * Controller that adds and removes products and coupons in the cart.
5   * Also provides functions for the continue shopping button and minicart.
6   *
7   * @module controllers/Cart
8   */
9
10 /* API Includes */
11 var ArrayList = require('dw/util/ArrayList');
12 var ISML = require('dw/template/ISML');
13 var Resource = require('dw/web/Resource');
14 var Transaction = require('dw/system/Transaction');
15 var URLUtils = require('dw/web/URLUtils');
16
17 /* Script Modules */
18 var app = require('~/cartridge/scripts/app');
19 var guard = require('~/cartridge/scripts/guard');
20
21 /* EXTEND */
22 var extendHelpers = require('*/cartridge/scripts/extendHelpers');
```

### 3.3.8 Add functionality to remove Extend warranty line items in Cart.js

- In: \*/app\_storefront\_controllers/cartridge/controllers/Cart.js
- Where: 'deleteProduct' action handler; replace this entire function with the following code

#### Code:

```
'deleteProduct': function (formgroup) {

    // EXTEND - Check if there is a warranty associated with the product that is being deleted
    var warrantyLineItem = extendHelpers.checkForWarrantyLI(cart, formgroup.getTriggeredAction().object);

    Transaction.wrap(function () {
        cart.removeProductLineItem(formgroup.getTriggeredAction().object);

        // EXTEND - if there is a warranty associated with the product remove the corresponding warranty
        if (warrantyLineItem) {
            cart.removeProductLineItem(warrantyLineItem);
        }
    });

    return {
        cart: cart
    };
}
```

```
168     'deleteGiftCertificate': function (formgroup) {
169         Transaction.wrap(function () {
170             cart.removeGiftCertificateLineItem(formgroup.getTriggeredAction().object);
171         });
172
173         return {
174             cart: cart
175         };
176     },
177     'deleteProduct': function (formgroup) {
178
179         // EXTEND - Check if there is a warranty associated with the product that is being deleted
180         var warrantyLineItem = extendHelpers.checkForWarrantyLI(cart, formgroup.getTriggeredAction().object);
181
182         Transaction.wrap(function () {
183             cart.removeProductLineItem(formgroup.getTriggeredAction().object);
184
185             // EXTEND - if there is a warranty associated with the product remove the corresponding warranty
186             if (warrantyLineItem) {
187                 cart.removeProductLineItem(warrantyLineItem);
188             }
189         });
190
191         return {
192             cart: cart
193         };
194     },
195     'editLineItem': function (formgroup) {
```

### 3.3.9 Disable edit in cart for Extend line items

- In: \*/app\_storefront\_core/cartridge/templates/default/checkout/cart/cart.isml
- Where: line 176: <iscomment>Call module to render product</iscomment>

#### Replace:

```
<isplayliproduct p_productli="${lineItem}" p_formli="${FormLi}" p_editable="${true}" p_hideprice="${true}" p_hidepromo="${false}" />
```

#### With:

```
<isif condition="${lineItem.custom.parentLineItemUUID}">
  <isplayliproduct p_productli="${lineItem}" p_formli="${FormLi}" p_editable="${false}" p_hideprice="${true}" p_hidepromo="${false}" />
<iselse>
  <isplayliproduct p_productli="${lineItem}" p_formli="${FormLi}" p_editable="${true}" p_hideprice="${true}" p_hidepromo="${false}" />
</isif>
```

```
163      <iscomment> Skip bonus-choice products in first pass. </iscomment>
164      <isif condition="${lineItem.bonusDiscountLineItem == null}">
165        <tr class="cart-row" data-uuid="${lineItem.getUUID()}">
166          <td class="item-image">
167            <isif condition="${lineItem.product != null && lineItem.product.getImage('small',0) != null}">
168              
170              
171            </isif>
172          </td>
173          <td class="item-details">
174
175            <iscomment>Call module to render product</iscomment>
176            <isif condition="${lineItem.custom.parentLineItemUUID}">
177              <isplayliproduct p_productli="${lineItem}" p_formli="${FormLi}" p_editable="${false}" p_hideprice="${true}" p_hidepromo="${false}" />
178            <iselse>
179              <isplayliproduct p_productli="${lineItem}" p_formli="${FormLi}" p_editable="${true}" p_hideprice="${true}" p_hidepromo="${false}" />
180            </isif>
181          </td>
182          <isif condition="${dw.system.Site.getCurrent().getCustomPreferenceValue('enableStorePickUp')}">
183            <td class="item-delivery-options">
184              <isinclude template="checkout/cart/storepickup/deliveryoptions" />
185            </td>
186          </isif>
187          <td class="item-quantity">
188            <isif condition="${lineItem.bonusProductLineItem}">
189              <isprint value="${lineItem.quantity}" />
190            <iselse/>
191              <input type="number" min="0" name="${FormLi.quantity.htmlName}" maxlength="6" value="${lineItem.quantity.value.toFixed()}" class="input-t
192            </isif>
193          </td>
```

### 3.3.10 Remove Add to Wishlist from cart for Extend products

- In: \*/app\_storefront\_core/cartridge/templates/default/checkout/cart/cart.isml
- Where: line 212

#### Replace:

```
<isif condition="${empty(pdct.ProductAddedToWishlist) || pdct.ProductAddedToWishlist != lineItem.productId}">
  <isif condition="${!isInWishList && !empty(lineItem.product)}">
    <a class="add-to-wishlist" href="${URLUtils.https('Cart-AddToWishlist', 'pid', lineItem.productId)}"
name="${FormLi.addToWishList.htmlName}" title="${Resource.msg('global.addthisitemtowishlist.title','locale',null)}">
      ${Resource.msg('global.addtowishlist','locale',null)}
    </a>
  </isif>
</iselse/>
<div class="in-wishlist">
  ${Resource.msg('global.iteminwishlist','locale',null)}
</div>
</isif>
```

#### With:

```
<isif condition="${!lineItem.custom.parentLineItemUUID}">
  <isif condition="${empty(pdct.ProductAddedToWishlist) || pdct.ProductAddedToWishlist != lineItem.productId}">
    <isif condition="${!isInWishList && !empty(lineItem.product)}">
      <a class="add-to-wishlist" href="${URLUtils.https('Cart-AddToWishlist', 'pid', lineItem.productId)}"
name="${FormLi.addToWishList.htmlName}" title="${Resource.msg('global.addthisitemtowishlist.title','locale',null)}">
        ${Resource.msg('global.addtowishlist','locale',null)}
      </a>
    </isif>
  </isif>
<div class="in-wishlist">
  ${Resource.msg('global.iteminwishlist','locale',null)}
</div>
</isif>
</isif>
```

```
208 <div class="item-user-actions">
209   <isif condition="${!lineItem.bonusProductLineItem || lineItem.getBonusDiscountLineItem() != null}">
210     <button class="button-text" type="submit" value="${Resource.msg('global.remove','locale',null)}" name="${FormLi.deleteProduct.htmlName}"><span>${Resource.msg('global.remove','locale',null)}</span></button>
211   </isif>
212   <isif condition="${!lineItem.custom.parentLineItemUUID}">
213     <isif condition="${empty(pdct.ProductAddedToWishlist) || pdct.ProductAddedToWishlist != lineItem.productId}">
214       <isif condition="${!isInWishList && !empty(lineItem.product)}">
215         <a class="add-to-wishlist" href="${URLUtils.https('Cart-AddToWishlist', 'pid', lineItem.productId)}" name="${FormLi.addToWishList.htmlName}"
216           ${Resource.msg('global.addtowishlist','locale',null)}
217         </a>
218       </isif>
219     </isif>
220     <div class="in-wishlist">
221       ${Resource.msg('global.iteminwishlist','locale',null)}
222     </div>
223   </isif>
224 </isif>
225 </div>
```



### 3.3.11 Add contract generation snippet to COPlaceOrder.js

- In: \*/app\_storefront\_controllers/cartridge/controllers/COPlaceOrder.js
- Where: function start(), in the if (!orderPlacementStatus.error) statement

Code:

```
// Add Extend products to Contracts queue
var extendHelpers = require('*/cartridge/scripts/extendHelpers');
extendHelpers.addContractToQueue(order);
164 } else if (handlePaymentsResult.missingPaymentInfo) {
165     return Transaction.wrap(function () {
166         OrderMgr.failOrder(order);
167         return {
168             error: true,
169             PlaceOrderError: new Status(Status.ERROR, 'confirm.error.technical')
170         };
171     });
172 }
173
174 var orderPlacementStatus = Order.submit(order);
175 if (!orderPlacementStatus.error) {
176     clearForms();
177
178     // Add Extend products to Contracts queue
179     var extendHelpers = require('*/cartridge/scripts/extendHelpers');
180     extendHelpers.addContractToQueue(order);
181 }
182 return orderPlacementStatus;
183 }
```

### Add contract generation snippet to COPlaceOrder.js in submit function

- In: \*/app\_storefront\_controllers/cartridge/controllers/COPlaceOrder.js
- Where: function submit(), in the if (!orderPlacementStatus.error) statement

```
// Add Extend products to Contracts queue
var extendHelpers = require('*/cartridge/scripts/extendHelpers');
extendHelpers.addContractToQueue(order);
/* Identifies if an order exists, submits the order, and shows a confirmation message.
*/
function submit() {
    var order = Order.get(request.httpParameterMap.order_id.stringValue);
    var orderPlacementStatus;
    if (order.object && request.httpParameterMap.order_token.stringValue === order.getOrderToken()) {
        orderPlacementStatus = Order.submit(order.object);
        if (!orderPlacementStatus.error) {
            // Add Extend products to Contracts queue
            var extendHelpers = require('*/cartridge/scripts/extendHelpers');
            extendHelpers.addContractToQueue(order);

            clearForms();
            return app.getController('COSummary').ShowConfirmation(order.object);
        }
    }
    app.getController('COSummary').Start();
}
```

### 3.3.12 calculate.js hook

- Open your calculate.js hook script (\*/cartridge/scripts/cart/calculate.js)
- Add the following script at the beginning of the script file

```
11 var HashMap = require('dw/util/HashMap');
12 var PromotionMgr = require('dw/campaign/PromotionMgr');
13 var ShippingMgr = require('dw/order/ShippingMgr');
14 var ShippingLocation = require('dw/order/ShippingLocation');
15 var TaxMgr = require('dw/order/TaxMgr');
16 var Logger = require('dw/system/Logger');
17 var Status = require('dw/system/Status');
18 var HookMgr = require('dw/system/HookMgr');
19 var collections = require('*/cartridge/scripts/util/collections');
20
21 // Extend normalization cart hook
22 var normalizeCartQuantities = require('*/cartridge/scripts/normalizationCartHook');
23
24 /**
25  * @function calculate
26  *
27  * calculate is the arching logic for computing the value of a basket. It makes
28  * calls into cart/calculate.js and enables both SG and OCAPI applications to share
29  * the same cart calculation logic.
```

- Before the tax calculation hook is called, add the following line

```
81
82 // =====
83 // ===== CALCULATE TAX =====
84 // =====
85
86 HookMgr.callHook('dw.order.calculateTax', 'calculateTax', basket);
87
88 // =====
89 // ===== EXTEND NORMALIZE CART HOOK =====
90 // =====
91
92 normalizeCartQuantities(basket);
93
94 // =====
95 // ===== CALCULATE BASKET TOTALS =====
96 // =====
97
98 basket.updateTotals();
99
100 // =====
101 // ===== DONE =====
102 // =====
103
104 return new Status(Status.OK);
105 };
106
```

- Find the function calculateProductPrices(basket) {...}
- Change the “handle non-catalog products” if statement to be:  
if (!productLineItem.catalogProduct || productLineItem.custom.parentLineItemUUID)

```

96  /**
97   * @function calculateProductPrices
98   *
99   * Calculates product prices based on line item quantities. Set calculates prices
100  * on the product line items. This updates the basket and returns nothing
101  *
102  * @param {object} basket The basket containing the elements to be computed
103  */
104  function calculateProductPrices(basket) {
105      // get total quantities for all products contained in the basket
106      var productQuantities = basket.getProductQuantities();
107      var productQuantitiesIt = productQuantities.keySet().iterator();
108
109      // get product prices for the accumulated product quantities
110      var productPrices = new HashMap();
111
112      while (productQuantitiesIt.hasNext()) {
113          var prod = productQuantitiesIt.next();
114          var quantity = productQuantities.get(prod);
115          productPrices.put(prod, prod.priceModel.getPrice(quantity));
116      }
117
118      // iterate all product line items of the basket and set prices
119      var productLineItems = basket.getAllProductLineItems().iterator();
120      while (productLineItems.hasNext()) {
121          var productLineItem = productLineItems.next();
122
123          // handle non-catalog products
124          if (!productLineItem.catalogProduct || productLineItem.custom.parentLineItemUUID) {
125              productLineItem.setPriceValue(productLineItem.basePrice.valueOrNull);
126              continue;
127          }
128
129          var product = productLineItem.product;
130
131          // handle option line items
132          if (productLineItem.optionProductLineItem) {
133              // for bonus option line items, we do not update the price
134              // the price is set to 0.0 by the promotion engine
135              if (!productLineItem.bonusProductLineItem) {

```

### 3.3.13 Include extend\_header.isml

- In: \*/app\_storefront\_core/cartridge/templates/default/components/header/htmlhead\_Ul.isml
- Where: at the end of the file

#### Code:

```

<iscomment>Extend Integration</iscomment>
<include template="components/header/extend_header"/>

```

```

1 <iscontent type="text/html" charset="UTF-8" compact="true"/>
2 <iscomment>THIS FILE ONLY CONTAINS DIFFERENCES BETWEEN SIMPLE AND RICH UI LIB INCLUDES</iscomment>
3 <iscomment>
4 The <!--BEGIN/END--> comments are control statements for the build cartridge which can be found in xChange https://xchange.demandware.com/doc
5 If you are not using the build cartridge the comments can be safely removed.
6 </iscomment>
7
8 <iscomment>THIRD PARTY STYLESHEETS STYLING</iscomment>
9 <link href="{URLUtils.staticURL('/lib/jquery/ui/jquery-ui.min.css')}" type="text/css" rel="stylesheet" />
10 <link href="//maxcdn.bootstrapcdn.com/font-awesome/4.3.0/css/font-awesome.min.css" rel="stylesheet">
11
12 <iscomment>Extend Integration</iscomment>
13 <isinclude template="components/header/extend_header"/>
14
15 <!--[if lt IE 9]>
16 <script src="{URLUtils.staticURL('/js/lib/html5.js')}"></script>
17 <![endif]-->
18
19 <iscomment>Salesforce Commerce Cloud active data scripts</iscomment>
20 <isactivedatahead/>
21

```

### 3.3.13 Add data attribute to link in minilineitems.isml

- In: \*/app\_storefront\_core/cartridge/templates/default/checkout/components/minilineitems.isml
- Where: line 89

#### Code:

```

<a href="{itemUrl}" data-is-extend-product="{productLineItem.custom.parentLineItemUUID ? true : false}"
title="{Resource.msgf('product.label','product',null, productLineItem.productName)}">
<isprint value="{productLineItem.productName}"/>
</a>

```

## External Interfaces

All requests are done through the Extend API. The full reference guide, along with the resource structure for requests, can be found on the developer site - <https://developers.extend.com/latest>

## Firewall Requirements

None

## 4. Testing

### 4.1 Enable Extend

- Make sure the following switches are set to Yes
  - Enable Extend
  - Enable in PDP
  - Enable Up-sell in PDP
  - Enable Up-sell in Cart
- Make sure the product catalog has been exported to Extend and your Extend representative enabled warranty plans in the system

### 4.2 PDP

#### 4.2.1 Extend Plan

Go to any simple or variant product that has an Extend warranty enabled. Select a variation and you will be presented with the Extend offer and widget to select a warranty plan.

FREE 2-Day SHIPPING FOR ORDERS OVER \$300

salesforce commerce cloud

Enter Keyword or Item No.

6

NEW ARRIVALS


WOMENS



MENS

ELECTRONICS

TOP SELLERS

Must Have Washable No-Iron Georgette Blouse







Must Have Washable No-Iron Georgette Blouse


Item No. 008885538465M

\$69.00

SELECT COLOR







IVORY

SELECT SIZE

6

8

10

12

8

AVAILABILITY

In Stock

Add product protection from **Extend** [Learn more](#)






1 Year - \$6.99

2 Years - \$12.99

3 Years - \$17.99

QTY 1

ADD TO CART



Add to Wishlist

Add to gift registry

Extend Integration for Salesforce Commerce Cloud

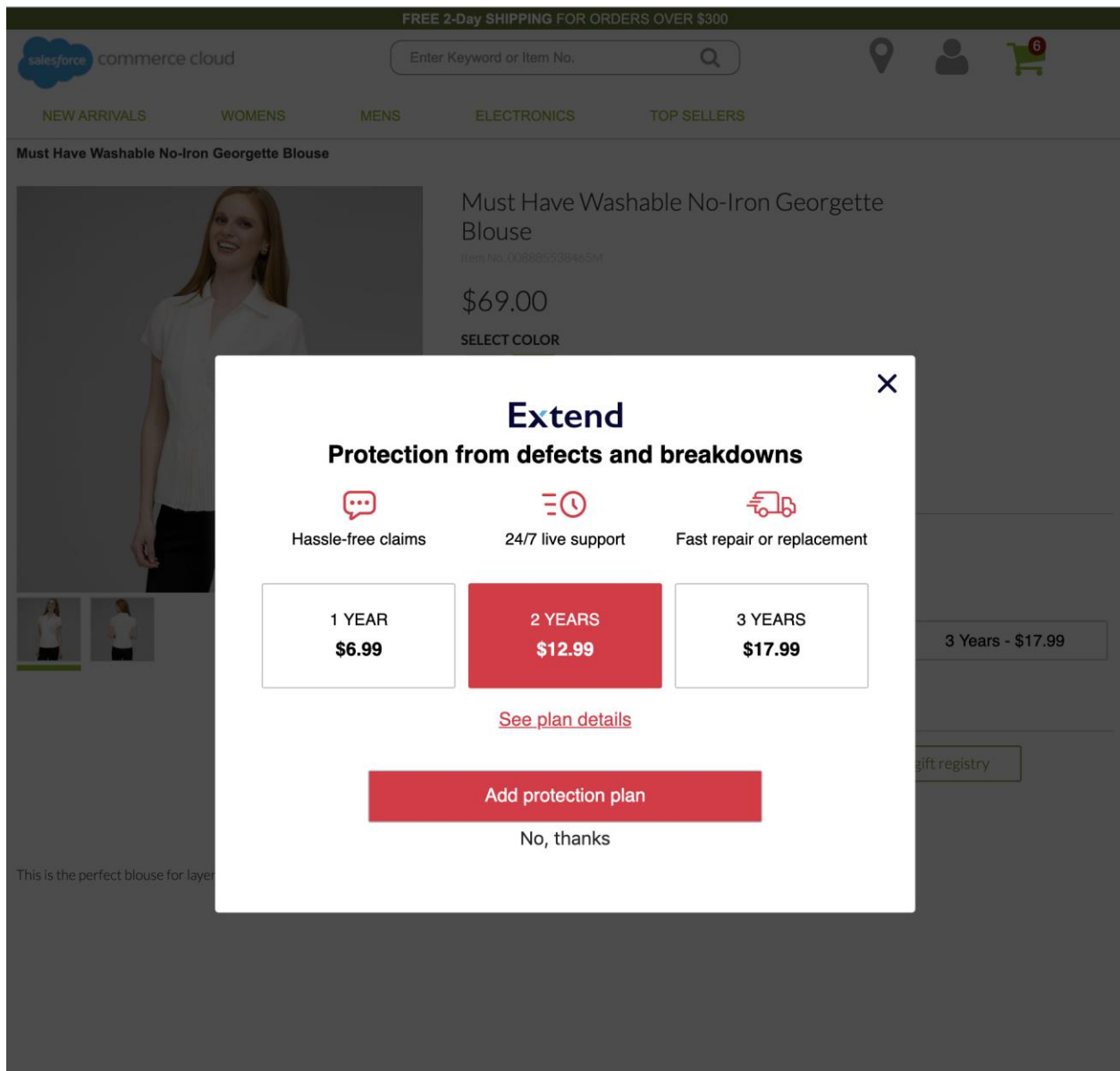
21

#### 4.2.2 Add to Cart

Select an Extend warranty plan and click Add to Cart. You will notice 2 products were added to the cart – the SFCC catalog product along with the Extend warranty plan.

#### 4.2.3 Up-sell Modal

Click Add to Cart on the PDP without selecting any Extend warranty plan. You will be presented with a modal recommending the associated Extend plans for this current catalog product. Selecting “Add protection plan” will proceed to add to cart the Extend plan, alongside the catalog product. Selecting “No, thanks” will only add to cart the catalog product.





## 4.3 Cart




### 4.3.1 Extend Plans in Cart

Confirm an Extend plan has been added to the cart.

FREE 2-Day SHIPPING FOR ORDERS OVER \$300









[NEW ARRIVALS](#) [WOMENS](#) [MENS](#) [ELECTRONICS](#) [TOP SELLERS](#)

[« Continue Shopping](#) [CHECKOUT](#)

PRODUCT	DELIVERY OPTIONS	QTY	PRICE	TOTAL
 <div><b>Must Have Washable No-Iron Georgette Blouse</b> Item No.: 008885538465M Color Ivory Size 8 <a href="#">Edit Details</a></div>	Home Delivery	<input type="text" value="1"/>	<b>In Stock</b> <a href="#">Remove</a> <a href="#">Add to Wishlist</a>	\$69.00 <b>\$69.00</b>
 <div><b>Extend Product Protection: 2 years for Must Have Washable No-Iron Georgette Blouse</b> Item No.: EXTEND-24 Extend Term: 2Y</div>	Home Delivery	<input type="text" value="1"/>	<b>In Stock</b> <a href="#">Remove</a>	\$12.99 <b>\$12.99</b>

[Apply](#)[Update Cart](#)

**Subtotal**  
Shipping  
Sales Tax

**\$81.99**  
-  
-

**Estimated Total**  
**\$81.99**


[« Continue Shopping](#) [CHECKOUT](#)

### 4.3.2 Up-sell button for eligible products in cart

Add to cart an eligible for warranty product, without selecting any Extend plan for it. Go to the cart page. You will notice a button that recommends adding a protection plan.

[« Continue Shopping](#)

[CHECKOUT](#)

PRODUCT	DELIVERY OPTIONS	QTY	PRICE	TOTAL
 <div><i>Platinum Blue Stripes Easy Care Fitted Shirt</i> Item No.: 008884304016 Color White Size 16 <a href="#">Edit Details</a> <a href="#">Add product protection for \$7.99</a></div>	Home Delivery	<input type="text" value="1"/>	<div>In Stock</div> <div><a href="#">Remove</a></div> <div><a href="#">Add to Wishlist</a></div> <div>\$75.00</div>	\$75.00
<input type="text" value="Enter coupon code"/>				
<div><a href="#">Apply</a></div> <div><a href="#">Update Cart</a></div>				
Subtotal				\$75.00
Shipping				-
Sales Tax				-
<b>Estimated Total</b>				<b>\$75.00</b>

#### 4.3.3 Up-sell modal in cart

Click on the up-sell button. You will be presented with a modal recommending the associated Extend plans for this current catalog product. Selecting “Add protection plan” will proceed to add to cart the Extend plan. Selecting “No, thanks” will close the modal.

### 4.4 Business Manager

#### 4.4.1 ExtendContractsQueue custom object



After placing an order containing Extend warranty plans, a new instance of this custom object is created. This holds all the data needed for generating an Extend contract.



General

## Manage '8f6c7fc79e60c41637321dab6a' (ExtendContractsQueue)

Fields with a red asterisk (\*) are mandatory. You can view and edit the name and description in other languages, if required. Click **Apply** to save the details.

contractData	
Site Currency:	USD 
Extend Warranty Plan:	{"purchasePrice":1364,"planId":"10001-misc-elec-base
SFCC Product:	{"referenceId":"008885538465M","purchasePrice":724
Last Modified:*	<div>12/18/2019 : 9:50 pm</div> <div>MM/dd/yyyy h:mm a</div>
Creation Date:*	<div>12/18/2019 : 9:50 pm</div> <div>MM/dd/yyyy h:mm a</div>
Order Customer:	{"phone":"4157477799","email":"kris@extend.com","nan
Log:	
 Line Item UUID:*	8f6c7fc79e60c41637321dab6a
Order Total:	9,238.00 (Number)
Order Number:	00000907

### 4.4.2 Extend Contracts Creation Job

This job will process the ExtendContractsQueue object instances and will generate a contract for each instance. After successfully executing the request, the objects will be deleted.

### 4.4.3 Custom Preference configuration for Extend

Update the Extend custom preference configuration for the website. Add access token and store ID provided by Extend.

To Enable Extend in production mode, select **Use Extend In Production Mode** as true.

## 5. Operations, Maintenance

### Support

If an issue appears, contact [integrations@extend.com](mailto:integrations@extend.com) for help.