

Advanced Order/Cart Search Module

Technical Design Documentation

Document History

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Updated By | Date | Comments |
| 1.0 | Shashank Bezalwar | 10/02/2015 | Initial Draft |
| 2.0 | Shashank Bezalwar | 27/04/2015 | Updated document to add the additional modifications done during development. |
|  |  |  |  |

Document Sign-Off

|  |  |  |
| --- | --- | --- |
| Name | Role | Signature/Date |
| Pascal ESPINOUSE |  |  |
| Aliasgar Muchhala |  |  |
| Gyaneshwar Dubey |  |  |

Table of Contents

[1. Introduction 4](#_Toc411349056)

[1.1 Purpose 4](#_Toc411349057)

[1.2 Intended Audience 5](#_Toc411349058)

[1.3 Document Scope 5](#_Toc411349060)

[1.4 Functional Rules 5](#_Toc411349062)

[1.5 Assumptions/Restrictions 5](#_Toc411349071)

[1.6 Documents Structure 5](#_Toc411349077)

[1.7 Additional Resources 6](#_Toc411349078)

[2. High Level Design Diagram 7](#_Toc411349079)

[2.1 Overview 7](#_Toc411349080)

[2.2 Sequence Diagram 9](#_Toc411349081)

[2.3 Use Case Diagram 10](#_Toc411349082)

[3. Detailed Application Design 11](#_Toc411349083)

[3.1 Class Diagram 11](#_Toc411349084)

[3.2 Beans 12](#_Toc411349085)

[3.3 Facade 12](#_Toc411349086)

[3.4 Strategy 12](#_Toc411349087)

[3.5 Controller 12](#_Toc411349088)

[3.6 View/JSP 13](#_Toc411349089)

[4. Database architecture 14](#_Toc411349090)

[4.1 ER Diagram 14](#_Toc411349091)

[5. Exception Handling and Logging 15](#_Toc411349092)

[5.1 Exception Scenarios 15](#_Toc411349093)

[5.2 Exception Handling: 15](#_Toc411349096)

[5.3 Logging 15](#_Toc411349097)

[6. Installation and Configuration 17](#_Toc411349098)

[6.1 Installation steps for the addon 17](#_Toc411349099)

[6.2 Configuration steps for the addon 17](#_Toc411349100)

[7. Appendix A *– Glossary of terms* 19](#_Toc411349101)

1. Introduction

Purpose

This document is the Technical Design Document which provides information of advanced order/cart search addon and steps to configure the module. It covers:

1. High Level Design
2. Application Architecture
3. Database Architecture
4. Installation Steps

Intended Audience

This document is aimed at Hybris developers and Technical members who want to implement the advanced order/cart search addon in Hybris.

Document Scope

This document covers the details about the sprint 1 advanced order/cart search addon.

Functional Rules

* The order history page displays all by using different search attributes
* The Cart Search functionality is by default unavailable and can be enabled using property order.search.showcarts in local.properties file.
* The search results will show Cart(s)/Order(s) of the B2B Unit the user belongs to.
* The order history is accessible via two links
  + My Account: It allows to get into Order History page
  + Order History: This screen lists all carts/orders
* Below is the list of attributes used to search the order:
  + Cart Number (web order)
  + Cart Label
  + Product Number
  + Creation Date
  + User Name

Assumptions/Restrictions

This module has been created for version 5.4.0.0.

* Display: The order advanced search module is presented in a page with several search criteria. Results are displayed in the same page with a pagination system
* Import: No export is available
* Export: No export is available
* Search: Many search criteria fields are available

Documents Structure

|  |  |  |
| --- | --- | --- |
| Section | | Description |
| 1 | Introduction | Describes the purpose and intended audience of the document and explains its structure. |
| 2 | High Level Design | Overview of the module. |
| 3 | Detailed Application Design | Application architecture applicable to this module. |
| 4 | Detailed Database Design | Conceptual and logical architecture applicable to this module. |
| 5 | Exception Management & Logging | Information about the Exception handling and logging. |
| 6 | Installation & Configuration | Steps to install and configure the module. |
| 7 | Appendices | Supporting information. |

Additional Resources

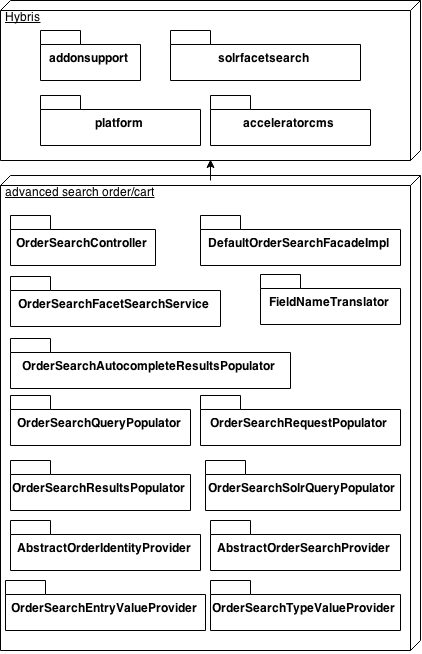
|  |  |  |  |
| --- | --- | --- | --- |
| # | Description | URL | Purpose |
| 1 | Wiki Portal | <https://wiki.hybris.com> | To gather more information |
| 2 | Coconet | <https://coconet2.capgemini.com/sf/go/doc3313610?nav=1> | To refer functional specifications |

:

1. High Level Design Diagram

Overview

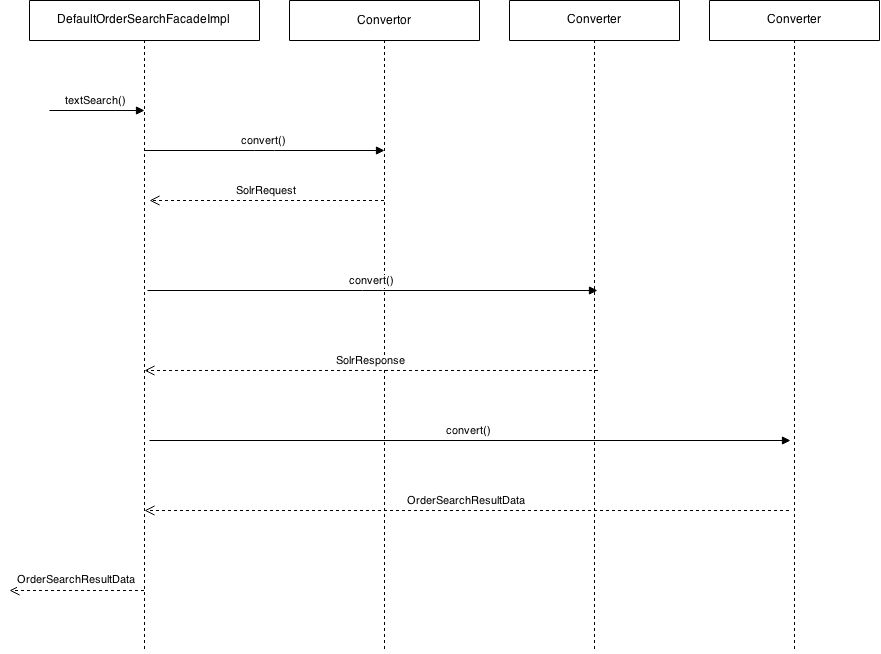
Below is the High level overview of the advanced order/cart search addon.



|  |  |  |
| --- | --- | --- |
| **#** | **Component/Extension/Module** | **Description** |
| 1 | **Addonsupport** | The addonsupport extension is an optional toolkit extension for developers of Accelerator AddOns. |
| 2 | **Platform** | The hybris Platform consists of a standard set of extensions providing the main functionality of a hybris installation. |
| 3 | **Solrfacetsearch** | The solrfacetsearch enables to do a faceted (also called dimensional) search over order/cart. |
| 4 | **Acceleratorcms** | The acceleratorcms extension provides a number of WCMS Components designed to be used in generated storefront. |
| 5 | **OrderSearchController** | Handles the advanced order search request. |
| 6 | **OrderSearchFacetSearchService** | OrderSearchFacetSearchService extends DefaultFacetSearchService and overrides checkCatalogVersions() method. |
| 7 | **DefaultOrderSearchFacadeImpl** | DefaultOrderSearchFacadeImpl does a text search on the order fields. |
| 8 | **FieldNameTranslator** | FieldNameTranslator provides the localized field name. |
| 9 | **OrderSearchAutocompleteResultsPopulator** | OrderSearchAutocompleteResultsPopulator populates translated field names of SolrSearchResponse |
| 10 | **OrderSearchQueryPopulator** | OrderSearchQueryPopulator populates SolrSearchRequest based on SearchQueryPageableData |
| 11 | **OrderSearchRequestPopulator** | OrderSearchResultsPopulator populates SolrSearchResponse based on SolrSearchRequest |
| 12 | **OrderSearchResultsPopulator** | OrderSearchResultsPopulator populates FacetSearchPageData based on SolrSearchResponse |
| 13 | **AbstractOrderIdentityProvider** | Provides order identifier |
| 14 | **AbstractOrderSearchProvider** | Provides order field values |
| 15 | **OrderSearchEntryValueProvider** | Provides order entries |
| 16 | **OrderSearchTypeValueProvider** | Provides if the field is of type cart or order |
| 17 | **OrderSearchB2BUnitValueProvider** | Provides the B2B Unit name |
| 18 | **OrderSearchUserValueProvider** | Provides User id |
| 19 | **OrderSearchNameValueProvider** | Provides name of Cart/Order |

Sequence Diagram

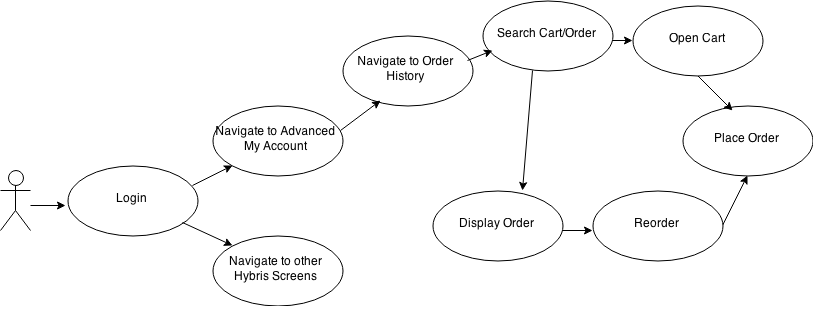
Sequence of events during the advanced search



|  |  |  |
| --- | --- | --- |
| **#** | **Component** | **Description** |
| 1 | **DefaultOrderSearchFacadeImpl** | The textSearch method gets fields and field values. The hash map contains all possible values for each order field. |
| 2 | **Convertor** | The convertor converts the hash map of the order fields to the SolrSearchRequest |
| 3 | **Convertor** | The convertor converts the SolrSearchRequest to SolrSearchResponse |
| 4 | **Convertor** | The convertor converts the hash map of the order SolrSearchResponse to the OrderSearchResultData. |

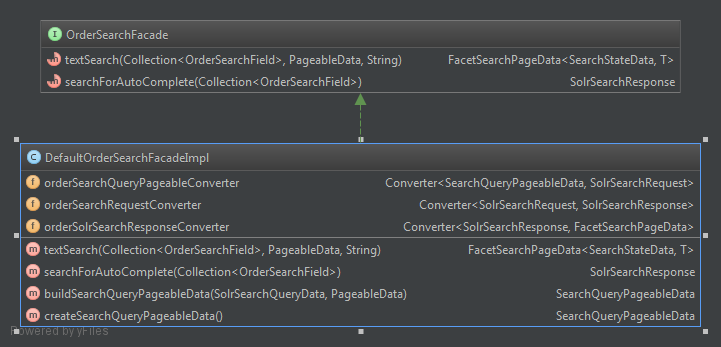
Use Case Diagram

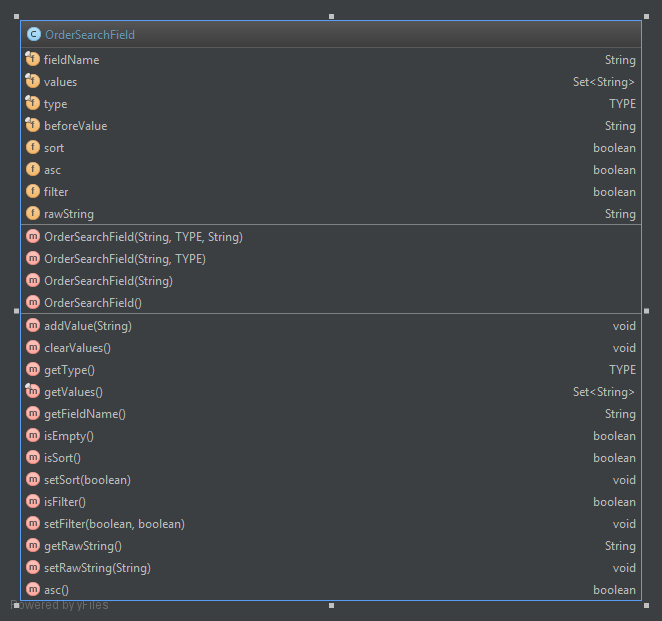
Use case diagram for the advanced order/cart search addon



1. Detailed Application Design

Class Diagram





The order search module is a package with new search features. The Hybris Solr layer is used and extended.

The following features are integrated:

* Order and Cart search
* Pagination
* Sort

Hybris does not support advanced search. The default search engine used all order/cart fields with the same value. This new module allows the user to search for each order/cart field.

Beans

**OrderSearchResultData:** The bean OrderSearchResultData is used for storing order search result.

Facade

**DefaultOrderSearchFacadeImpl:** DefaultOrderSearchFacadeImpl facade provides a simple interface to search order/cart.

Strategy

N.A

Controller

**OrderSearchController:** OrderSearchController extends AbstractAddOnPageController. This controller handles all the advanced cart/order search requests.

View/JSP

**orderSearchLayoutPage.jsp:** orderSearchLayoutPage.jsp is the custom jsp page that allows the user to search the order/cart using the order/cart fields.

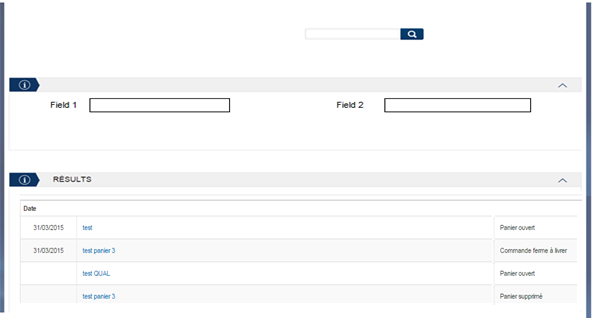
The JSP page consists of two sections.

1. Search Section
2. Results Section

The search section consists of the fields that can be used to search the cart/order. A single field or a combination of the fields can be used to search the cart/order.

The results section displays the orders/carts that qualify the selection criteria. The results are shown in the same page with pagination.

Below is the mock up of the JSP page. The page will be enhanced in future releases.



1. Database architecture

ER Diagram

N.A.

1. Exception Handling and Logging

Exception Scenarios

The method populate in Class OrderSearchRequestPopulatorcan throw FacetSearchException. This exception is thrown if the search query is not configured correctly. It is logged using Log.error() method in the log file.

The value provider classes can throw FieldValueProviderException. This exception is thrown whenever the specialized field value provider cannot obtain the value.

The populator classes can throw ConversionException. This exception is thrown if an error occurs during conversion.

Below is the list of exceptions thrown and their exception messages:

|  |  |
| --- | --- |
| **Exception** | **Message** |
| FacetSearchException | Exception while executing SOLR search |
| FieldValueProviderException | N.A. |
| ConversionException | N.A. |

Currently there is no error codes used in the module. The future releases might have error code associated with it.

Exception Handling:

Below is the Exception handling best practice used in the module

* Never swallow the exception in catch block
* Declare the specific checked exceptions that your method can throw
* Do not catch the Exception class rather catch specific sub classes
* Never catch Throwable class
* Always correctly wrap the exceptions in custom exceptions so that stack trace is not lost
* Either log the exception or throw it but never do the both
* Never throw any exception from finally block
* Always catch only those exceptions that you can actually handle
* Don't use printStackTrace() statement or similar methods
* Use finally blocks instead of catch blocks if you are not going to handle exception
* Remember "Throw early catch late" principle
* Always clean up after handling the exception
* Throw only relevant exception from a method
* Never use exceptions for flow control in your program
* Validate user input to catch adverse conditions very early in request processing
* Always include all information about an exception in single log message
* Pass all relevant information to exceptions to make them informative as much as possible
* Always terminate the thread which it is interrupted
* Use template methods for repeated try-catch
* Document all exceptions in your application in javadoc

Logging

* For logging in the advanced order/cart search module Log4j will be used.
* The logging level will be set at INFO level.
* The exceptions are logged using Log.error() method.

1. Installation and Configuration

Installation steps for the addon

Add the ordersearch module dependency in hybris/Assets/hybris/config/localextension.xml file as follows.

<extension name=" ordersearch "/>

Next, execute the following commands marked in green using command prompt.

Cd hybris.directory/bin/platform

# **Note:** hybris.directory is the installation location of the hybris

setantenv.bat

ant addoninstall -Daddonnames=" ordersearch" -DaddonStorefront.yacceleratorstorefront="yb2bacceleratorstorefront"

WARNING: yb2bacceleratorstorefront must be replaced by your custom strorefront

Configuration steps for the addon

The hybris WCMS Cockpit provides a central facility for managing content. CMS pages are laid out according to their Page Template.

A Page Template in the hybris Multichannel Accelerator project has:

* A list of Content Slots that are predefined for the template, for the hybris Multichannel Accelerator this includes the mini cart, homepage image, and navigation bars.
* A FrontendTemplateName that corresponds to the name of a JSP that drives the layout of the page.
* A list of PageTypes to which the template is restricted to.
* A VelocityTemplate used to drive the layout of the WCMS Cockpit Page View.

**Below are the steps to add a content page in WCMS using impex.**

**Note: The content page can be added using hmc also, This document concentrates on addin it via impex.**

1. **The Page Template is configured using the following ImpEx script:**

INSERT\_UPDATE PageTemplate;$contentCV[unique=true];uid[unique=true];name;frontendTemplateName;restrictedPageTypes(code);active[default=true]

;;OrderSearchTemplate;Order search template;addon:/ordersearch/pages/orderSearchLayoutPage;ContentPage

1. **After the page template is defined the ContenSlotNames for the template are set up:**

INSERT\_UPDATE ContentSlotName;name[unique=true];template(uid,$contentCV)[unique=true][default='OrderSearchTemplate'];validComponentTypes(code)

;SiteLogo;;CMSImageComponent,BannerComponent

;HeaderLinks;;CMSLinkComponent,CMSParagraphComponent

;MiniCart;;MiniCartComponent

;NavigationBar;;NavigationBarComponent

;Footer;;CMSLinkComponent,CMSParagraphComponent,FooterComponent

1. **Then the list of ContentSlotForTemplate bindings for the PageTemplate is added:**

INSERT\_UPDATE ContentSlotForTemplate;$contentCV;uid[unique=true];position[unique=true];pageTemplate(uid,$contentCV)[unique=true][default='OrderSearchTemplate'];contentSlot(uid,$contentCV)[unique=true];allowOverwrite

;;SiteLogo-ContentPage2;SiteLogo;;SiteLogoSlot;true

;;HomepageLink-ContentPage2;HomepageNavLink;;HomepageNavLinkSlot;true

;;NavigationBar-ContentPage2;NavigationBar;;NavigationBarSlot;true

;;MiniCart-ContentPage2;MiniCart;;MiniCartSlot;true

;;Footer-ContentPage2;Footer;;FooterSlot;true

;;HeaderLinks-ContentPage2;HeaderLinks;;HeaderLinksSlot;true

1. **The last step is to add the new content page.**

INSERT\_UPDATE ContentPage;$contentCV[unique=true];uid[unique=true];name;masterTemplate(uid,$contentCV);label;defaultPage[default='true'];approvalStatus(code)[default='approved'];homepage[default='false']

;;orderSearch;Order search Page;OrderSearchTemplate;/orderSearch

**How to add a new field to search**

It is possible to add new field to be able to search order/cart. This can be a new field or existing order/cart attribute. These steps below explain how to configure the existing field.

Below are the steps to add the new field to search the order/cart.

1. Edit file /ordersearch/resources/ordersearch-spring.xml. Add the field name to the configuration.

Below is the snippet from the existing configuration. Please note this is a sample configuration:

**<bean id="** **defaultOrderSearchFacade"**

**class="** **com.generic.ordersearch.facade.impl.DefaultOrderSearchFacadeImpl ">**

**<property name="fieldsList">**

**<list>**

**<value>Cart Number (web order)</value>**

**<value>Cart Label</value>**

**<value>Product Number</value>**

**<value>Creation Date</value>**

**<value>User Name</value>**

**</list>**

**</property>**

**</bean>**

1. Add the new field to the list mentioned above. The new field would now be available in JSP page.
2. Update the OrderSearchController to get the new field and pass the field to DefaultOrderSearchFacadeImpl, so that the order/cart can be searched on the new field.
3. Edit the hybris/Assets/hybris/bin/custom/b2bassets/b2bassetscore/resources/b2bassetscore/import/stores/powertools/solr.impex file and add line to add new field or existing field to solr

**;$solrIndexedType ; code ;string ; ; ; ; ; ; ; ;**

**After line**

**;$solrIndexedType; firstCategoryNameList ;string ; ; ; ; ; ; ;firstVariantCategoryNameListValueProvider;**

**How to remove a field from cart/order search**

Below are the steps to remove a field from the order/cart search.

1. Edit file /ordersearch/resources/ordersearch-spring.xml. Remove the field name from the configuration.

Below is the snippet from the existing configuration. Please note this is a sample configuration:

**<bean id="** **defaultOrderSearchFacade"**

**class="** **com.generic.ordersearch.facade.impl.DefaultOrderSearchFacadeImpl ">**

**<property name="fieldsList">**

**<list>**

**<value>Cart Number (web order)</value>**

**<value>Cart Label</value>**

**<value>Product Number</value>**

**<value>Creation Date</value>**

**<value>User Name</value>**

**</list>**

**</property>**

**</bean>**

1. Remove the field from the list mentioned above. The field would now not be visible in JSP page.
2. Update the OrderSearchController to remove the field.
3. Edit the hybris/Assets/hybris/bin/custom/b2bassets/b2bassetscore/resources/b2bassetscore/import/stores/powertools/solr.impex file and remove the field from solr impex.

**How to enable sorting on a field**

The steps below enable the user to sort results on a given field.

1. Edit the hybris/Assets/hybris/bin/custom/b2bassets/b2bassetscore/resources/b2bassetscore/import/stores/powertools/solr.impex file and add the field name that you wish to sort on. Example below:

**Add line**

**;sortRef5;$solrIndexedType;price-asc;false**

**After line**

**INSERT\_UPDATE SolrSort;&sortRefID;indexedType(identifier)[unique=true];code[unique=true];useBoost**

The line above adds a new sort option.

1. Then add the field name that should be used for sorting. Example below:

**Add line**

**;$solrIndexedType:price-asc;priceValue;true**

**After line**

**INSERT\_UPDATE SolrSortField;sort(indexedType(identifier),code)[unique=true];fieldName[unique=true];ascending[unique=true]**

The above line will add a sort option based on price in ascending order. To add a descending sort option

**Add line**

**;$solrIndexedType:price-asc;priceValue;false**

1. Appendix A *– Glossary of terms*

|  |  |
| --- | --- |
| **Term/Abbreviation/Acronym** | **Meaning** |
| Solr | Solr is an open source enterprise search platform used for text based search in Hybris |
| B2B | Business to Business |