

Advanced Product Search Module

Technical Design Documentation

Document History

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Updated By | Date | Comments |
| 1.0 | Shashank Bezalwar | 09/02/2015 | Initial Draft |
| 1.1 | Jayachandra gudiwada | 05/04/2015 | Initial Draft |
|  |  |  |  |

Document Sign-Off

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

Table of Contents

[1. Introduction 3](#_Toc418526311)

[1.1 Purpose 3](#_Toc418526312)

[1.2 Intended Audience 3](#_Toc418526313)

[1.3 Document Scope 3](#_Toc418526314)

[1.4 Functional Rules 3](#_Toc418526315)

[1.5 Assumptions/Restrictions 3](#_Toc418526316)

[1.6 Documents Structure 4](#_Toc418526317)

[1.7 Additional Resources 4](#_Toc418526318)

[2. High Level Design Diagram 5](#_Toc418526319)

[2.1 Overview 5](#_Toc418526320)

[2.2 Sequence Diagram 7](#_Toc418526321)

[2.3 Use Case Diagram 8](#_Toc418526322)

[3. Detailed Application Design 9](#_Toc418526323)

[3.1 Class Diagram 9](#_Toc418526324)

[3.2 Beans 10](#_Toc418526325)

[3.3 Facade 10](#_Toc418526326)

[3.4 Strategy 10](#_Toc418526327)

[3.5 Controller 10](#_Toc418526328)

[3.6 View/JSP 11](#_Toc418526329)

[4. Database architecture 13](#_Toc418526330)

[4.1 ER Diagram 13](#_Toc418526331)

[5. Exception Handling and Logging 14](#_Toc418526332)

[5.1 Exception Scenarios 14](#_Toc418526333)

[5.2 Exception Handling: 14](#_Toc418526334)

[5.3 Logging 15](#_Toc418526335)

[6. Installation and Configuration 16](#_Toc418526336)

[6.1 Installation steps for the addon 16](#_Toc418526337)

[6.2 Configuration steps for the addon 16](#_Toc418526338)

[7. Appendix A *– Glossary of terms* 20](#_Toc418526339)

1. Introduction

Purpose

This document is the Technical Design Document which provides information of advanced product 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 product search addon in Hybris.

Document Scope

This document covers the details about the sprint 1 advanced product search addon.

Functional Rules

* The product search page displays all products (viewable products for the B2B unit) by using different search attributes.
* The product search page is accessible via advanced search.
* Below is the list of attributes used to search the product:
* SKU (Product Code)
* Product Name
* Description
* Summary

Assumptions/Restrictions

This module has been created for version 5.4.0.0.

**Display:** The product 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 import 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. |

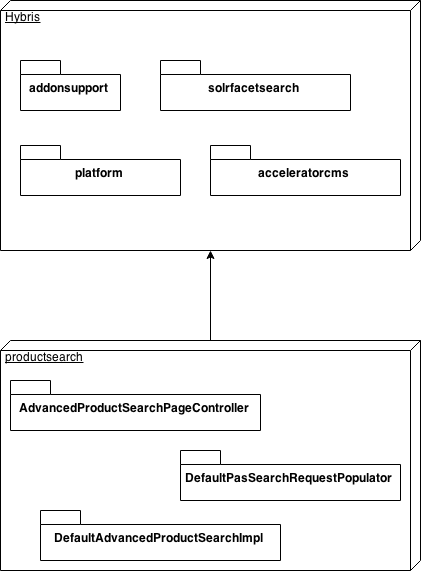
Additional Resources

|  |  |  |  |
| --- | --- | --- | --- |
| # | 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>  <https://coconet2.capgemini.com/sf/go/doc3383939?nav=1> | To refer functional specifications |

1. High Level Design Diagram

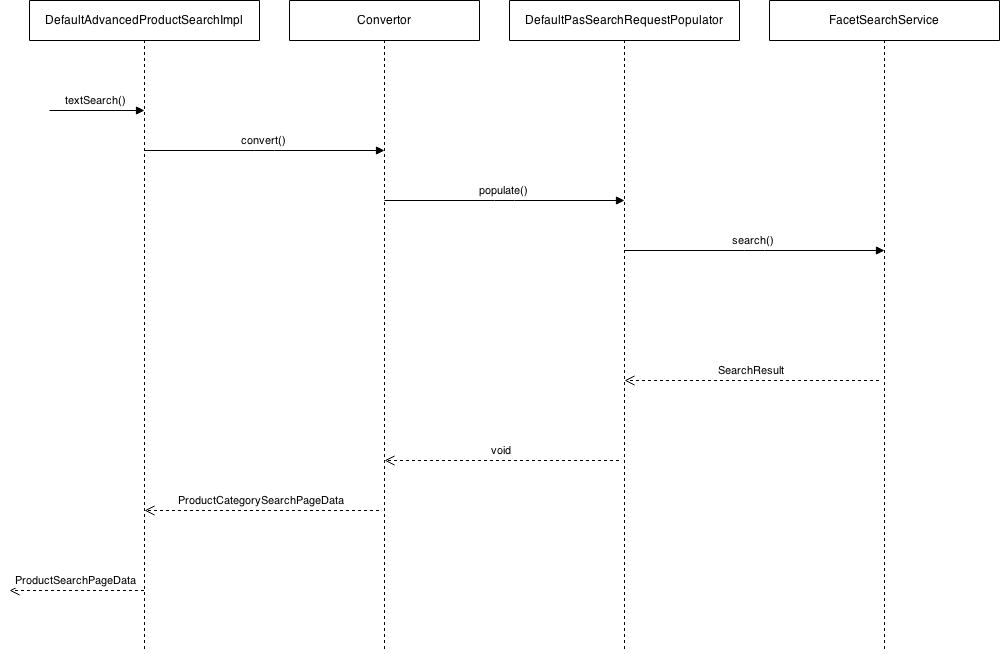
Overview

Below is the High level overview of the advanced product 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 products. |
| 4 | **Acceleratorcms** | The acceleratorcms extension provides a number of WCMS Components designed to be used in generated storefront. |
| 5 | **AdvancedProductSearchPageController** | Handles the advanced search request. |
| 6 | **DefaultSearchRequestPopulator** | DefaultSearchRequestPopulator populates SolrSearchResponse values based on SolrSearchRequest. |
| 7 | **DefaultAdvancedProductSearchImpl** | DefaultAdvancedProductSearchImpl does a text search on the product fields. |

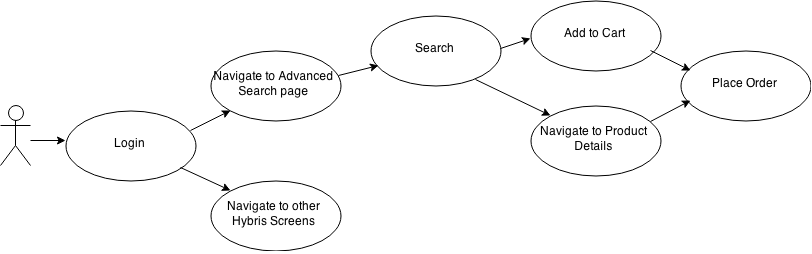
Sequence Diagram

Sequence of events during the advanced search 

|  |  |  |
| --- | --- | --- |
| **#** | **Component** | **Description** |
| 1 | **DefaultAdvancedProductSearchImpl** | The textSearch method gets fields and field values. The hash map contains all possible values for each product field. |
| 2 | **Convertor** | The convertor converts the hash map of the product fields to the SolrSearchRequest |
| 3 | **DefaultPasSearchRequestPopulator** | The DefaultPasSearchRequestPopulator populates SolrSearchResponse based on SolrSearchRequest. |
| 4 | **FacetSearchService** | The FacetSearchService does the search with the specific search query. |

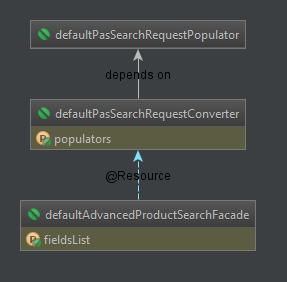
Use Case Diagram

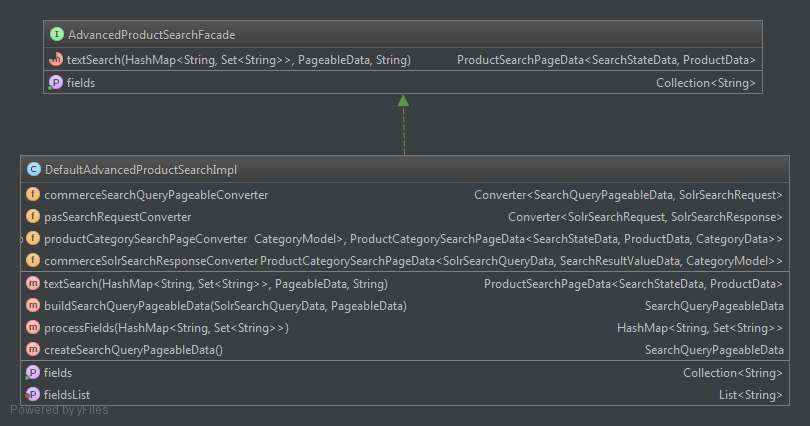
Use case diagram for the advanced product search addon



1. Detailed Application Design

Class Diagram





The advanced search module is a package with new search features. The Hybris Solr layer is used and extended. The following features are integrated:

* Search in one field
* Pagination
* Sort

Hybris does not support advanced search. The default search engine used all product fields with the same value. This new module allows the user to search for each product field. The next section explains in detail the components of the module.

Beans

**PASSolrSearchQueryData:** The bean PASSolrSearchQueryData contains hash map of fields used for advanced product searching.

Facade

**AdvancedProductSearchFacade:** AdvancedProductSearchFacade façade provides a simple interface to search products.

Strategy

N.A

Controller

**AdvancedProductSearchPageController:** AdvancedProductSearchPageController extends AbstractSearchPageController. This controller handles all the advanced product search requests.

View/JSP

**searchPageLayoutPage.jsp:** searchPageLayoutPage.jsp is the custom jsp page that allows the user to search the product using the product fields.

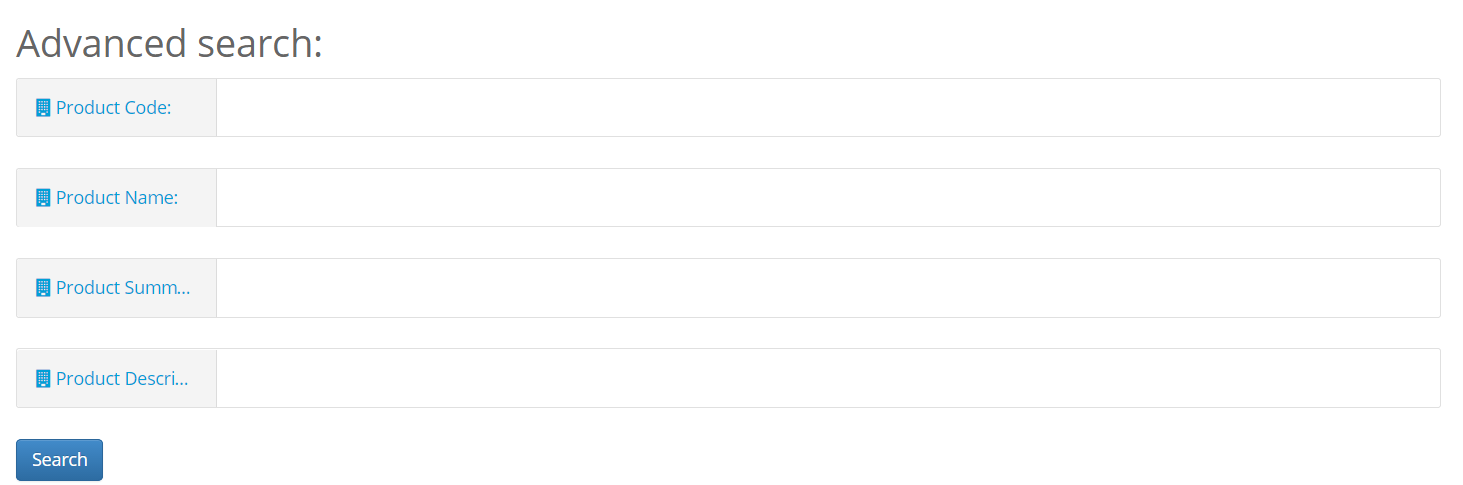
The JSP page consists of two sections.

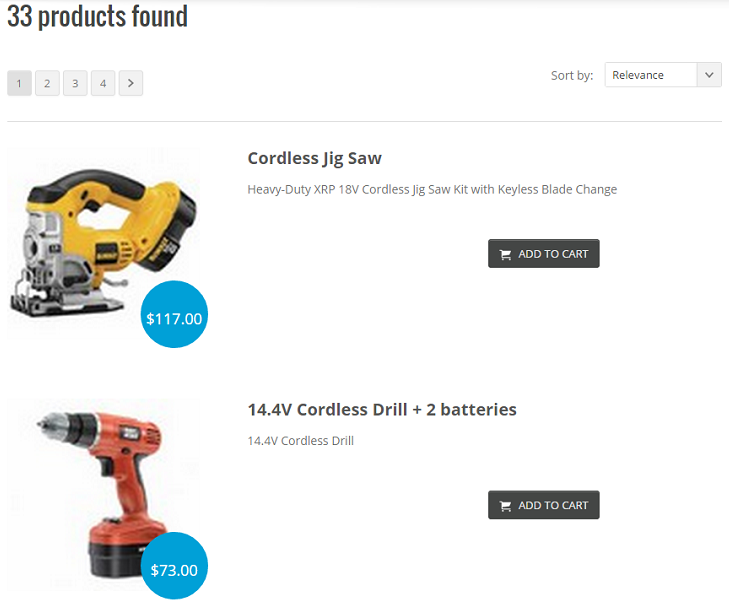
1. Search Section
2. Results Section

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

The results section displays the products 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 DefaultPasSearchRequestPopulator can 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. Below is the list of exceptions thrown and their exception messages:

|  |  |
| --- | --- |
| **Exception** | **Message** |
| FacetSearchException | Exception while executing SOLR search |

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 product 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 productsearch module dependency in hybris/Assets/hybris/config/localextension.xml file as follows.

<extension name="productsearch"/>

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="productsearch" -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]

;;ProductAdvancedSearchTemplate;Product advanced search template;addon:/productsearch/pages/searchPageLayoutPage;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='ProductAdvancedSearchTemplate'];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='ProductAdvancedSearchTemplate'];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.**

# Simple Content Pages

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

;;productAdvancedSearch;Product advanced search Page;ProductAdvancedSearchTemplate;/advancedSearch

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

It is possible to add new field to be able to search product. This can be a new field or existing product attribute. The steps below explains how to configure the existing field.

Below are the steps to add the new field to search the product.

1. Edit file /productsearch/resources/productsearch-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="defaultAdvancedProductSearchFacade"**

**class="com.generic.productsearch.service.impl.DefaultAdvancedProductSearchImpl">**

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

**<list>**

**<value>code</value>**

**<value>classificationNumber</value>**

**<value>summary</value>**

**<value>description</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 AdvancedProductSearchPageController to get the new field and pass the field to AdvancedProductSearchFacade, so that the product 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; new\_field\_name ;text ; ; ;true; ; ; ;;**

After line

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

**How to remove a field from product search**

Below are the steps to remove a field from the product search.

1. Edit file /productsearch/resources/productsearch-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="defaultAdvancedProductSearchFacade"**

**class="com.generic.productsearch.service.impl.DefaultAdvancedProductSearchImpl">**

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

**<list>**

**<value>code</value>**

**<value>classificationNumber</value>**

**<value>summary</value>**

**<value>description</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 AdvancedProductSearchPageController 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 |
| SKU | Stock Keeping Unit |