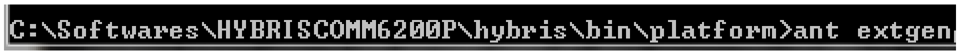
Q: How to generate your own backoffice-based extension to add new cockpit functionality?

Ans: - ybackoffice template is used to generate your own back office extension

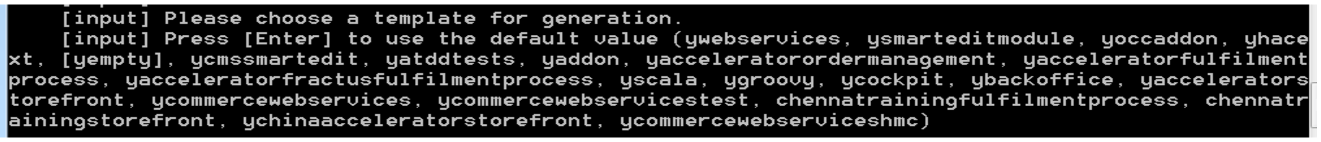
Explain "ybackoffice" using ant extgen?

- ✓ **Step 1 =** C:\Softwares\HYBRISCOMM6200P\hybris\bin\platform>**ant extgen** & Hit Enter



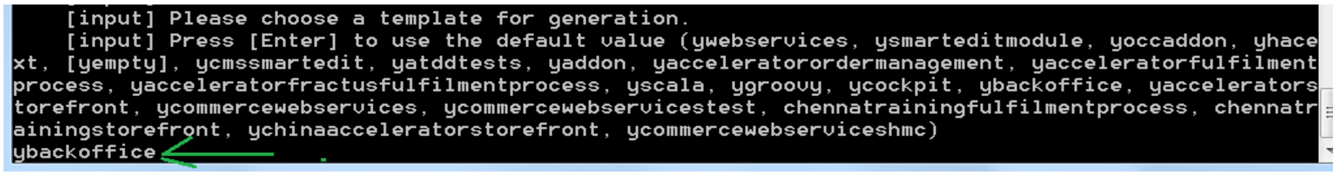
```
C:\Softwares\HYBRISCOMM6200P\hybris\bin\platform>ant extgen
```

NOTE: - It will ask you -- "please choose a template for generation".



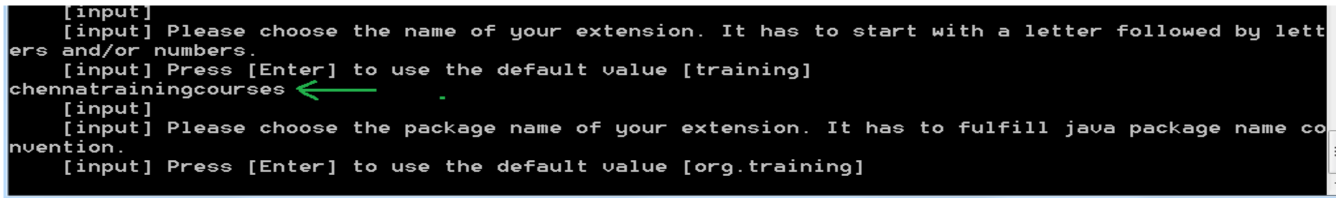
```
     [input] Please choose a template for generation.
     [input] Press [Enter] to use the default value (ywebservices, ysmarteditmodule, yoccaddon, yhace
xt, [yempty], ycmssmartedit, yatddtests, yaddon, yacceleratorordermanagement, yacceleratorfulfilment
process, yacceleratorfractusfulfilmentprocess, yscala, ygroovy, ycockpit, ybackoffice, yaccelerators
torefront, ycommercewebservices, ycommercewebservicestest, chennatrainingfulfilmentprocess, chennatr
ainingstorefront, ychinaacceleratorstorefront, ycommercewebserviceshmc)
```

- ✓ **Step 2 =** Choose **Template** for generation = **ybackoffice (**type ybackoffice) and hit enter.



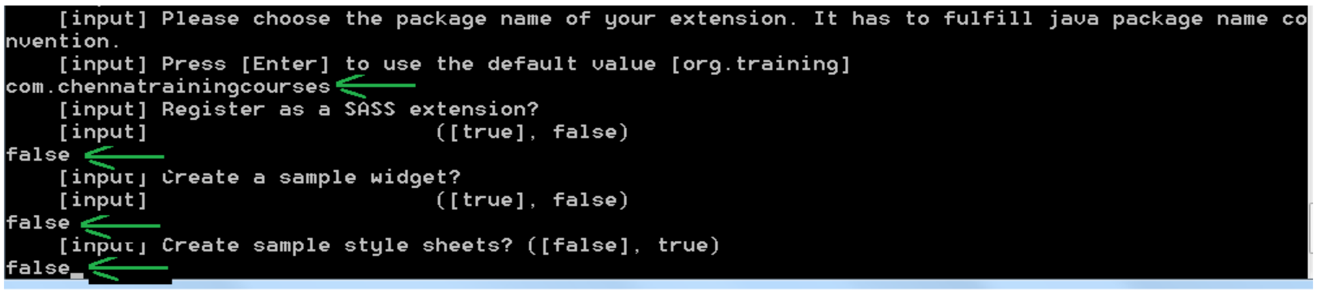
```
     [input] Please choose a template for generation.
     [input] Press [Enter] to use the default value (ywebservices, ysmarteditmodule, yoccaddon, yhace
xt, [yempty], ycmssmartedit, yatddtests, yaddon, yacceleratorordermanagement, yacceleratorfulfilment
process, yacceleratorfractusfulfilmentprocess, yscala, ygroovy, ycockpit, ybackoffice, yaccelerators
torefront, ycommercewebservices, ycommercewebservicestest, chennatrainingfulfilmentprocess, chennatr
ainingstorefront, ychinaacceleratorstorefront, ycommercewebserviceshmc)
ybackoffice
```

- ✓ **Step 3 =** Choose name of your extension = **chennatrainingcourses** & hit enter

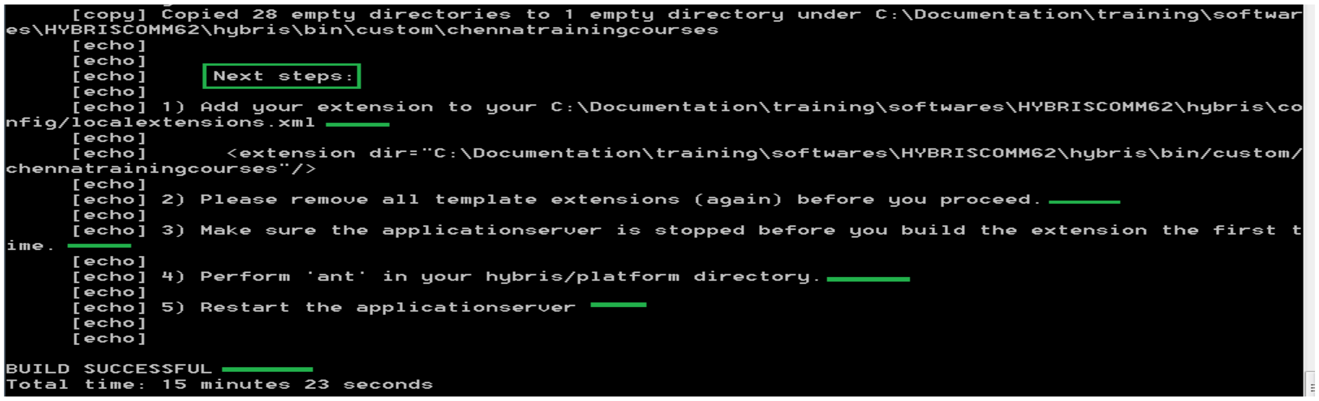


```
     [input]
     [input] Please choose the name of your extension. It has to start with a letter followed by lett
ers and/or numbers.
     [input] Press [Enter] to use the default value [training]
chennatrainingcourses
     [input]
     [input] Please choose the package name of your extension. It has to fulfill java package name co
nvention.
     [input] Press [Enter] to use the default value [org.training]
```

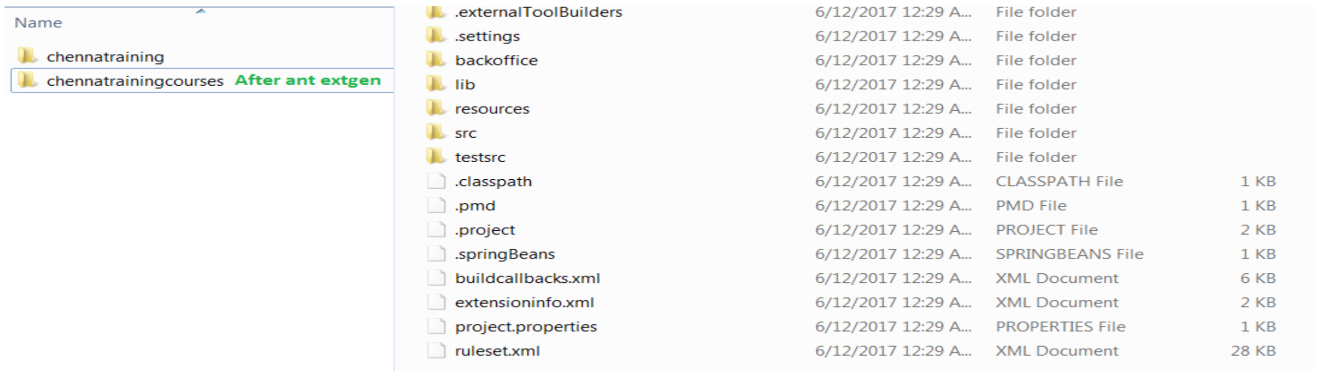
- ✓ **Step 4 =** Choose package name of your extension **= com.chennatrainingcourses** & other options as false



```
     [input] Please choose the package name of your extension. It has to fulfill java package name co
nvention.
     [input] Press [Enter] to use the default value [org.training]
com.chennatrainingcourses
     [input] Register as a SASS extension?
     [input]                         ([true], false)
false
     [input] Create a sample widget?
     [input]                         ([true], false)
false
     [input] Create sample style sheets? ([false], true)
false
```

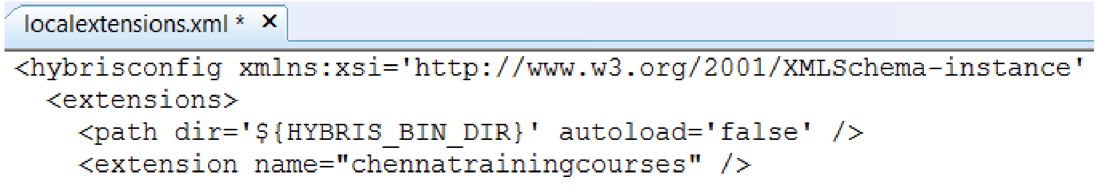


```
      [copy] Copied 28 empty directories to 1 empty directory under C:\Documentation\training\softwar
es\HYBRISCOMM62\hybris\bin\custom\chennatrainingcourses
      [echo]
      [echo]
      [echo]     Next steps:
      [echo]
      [echo] 1) Add your extension to your C:\Documentation\training\softwares\HYBRISCOMM62\hybris\co
nfig/localextensions.xml
      [echo]
      [echo]         <extension dir="C:\Documentation\training\softwares\HYBRISCOMM62\hybris\bin/custom/
chennatrainingcourses"/>
      [echo]
      [echo] 2) Please remove all template extensions (again) before you proceed.
      [echo]
      [echo] 3) Make sure the applicationserver is stopped before you build the extension the first t
ime.
      [echo]
      [echo] 4) Perform 'ant' in your hybris/platform directory.
      [echo]
      [echo] 5) Restart the applicationserver
      [echo]
      [echo]

BUILD SUCCESSFUL
Total time: 15 minutes 23 seconds
```



Name		Date modified	Type	Size
chennatraining	.externalToolBuilders	6/12/2017 12:29 A...	File folder	
chennatrainingcourses After ant extgen	.settings	6/12/2017 12:29 A...	File folder	
	backoffice	6/12/2017 12:29 A...	File folder	
	lib	6/12/2017 12:29 A...	File folder	
	resources	6/12/2017 12:29 A...	File folder	
	src	6/12/2017 12:29 A...	File folder	
	testsrc	6/12/2017 12:29 A...	File folder	
	.classpath	6/12/2017 12:29 A...	CLASSPATH File	1 KB
	.pmd	6/12/2017 12:29 A...	PMD File	1 KB
	.project	6/12/2017 12:29 A...	PROJECT File	2 KB
	.springBeans	6/12/2017 12:29 A...	SPRINGBEANS File	1 KB
	buildcallbacks.xml	6/12/2017 12:29 A...	XML Document	6 KB
	extensioninfo.xml	6/12/2017 12:29 A...	XML Document	2 KB
	project.properties	6/12/2017 12:29 A...	PROPERTIES File	1 KB
	ruleset.xml	6/12/2017 12:29 A...	XML Document	28 KB

✓ **Step 5 =** Add your extension "**chennatrainingcourses"** to your ... \config\localextensions.xml



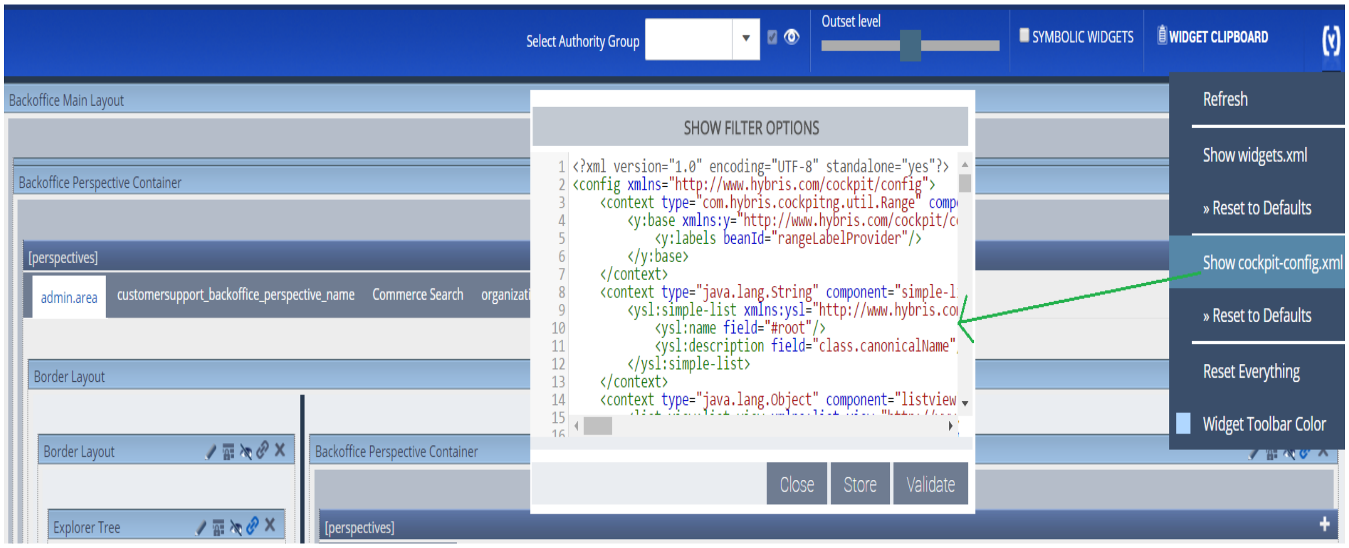
localextensions.xml * ✕

```
<hybrisconfig xmlns:xsi='http://www.w3.org/2001/XMLSchema-instance'
  <extensions>
    <path dir='${HYBRIS_BIN_DIR}' autoload='false' />
    <extension name="chennatrainingcourses" />
```

✓ **Step6=** Define the node structure & layout of screen in "**chennatrainingcourses-backoffice-config.xml**" file. Goto *\hybris\bin\hybris\bin\custom\chennatrainingcourses\resources\**chennatrainingcourses-backoffice-config.xml** file and add the xml configuration (define the node and columns)
Insert the following complete code block in place of the example node structure (remove all the existing code inside the <config> tags)

```
        <context component="explorer-tree" merge-by="module">
                <n:explorer-tree xmlns:n="http://www.hybris.com/cockpitng/config/explorertree">
                    <n:navigation-node id="ChennaTraining">
                        <n:type-node id="Courses" code="Courses" />
                    </n:navigation-node>
                 </n:explorer-tree>
              </context>
              <context type="Courses" component="listview">
                 <list:list-view xmlns:list="http://www.hybris.com/cockpitng/component/listView">
                        <list:column qualifier="code"/>
                        <list:column qualifier="name"/>
                        <list:column qualifier="duration"/>
                        <list:column qualifier="amount"/>
                 </list:list-view>
        </context>
```

Note = (1) You can also create the configuration directly using the Application Orchestrator without having to write it in the chennatrainingcourses-backoffice-config.xml file. To do this, open the editor using **Show cockpit-config.xml** and hit the **Store** button after changing the configuration. However, if you hit **Reset to Defaults** again, your dynamic configuration will be lost. To persist it permanently, it is recommended to create the configuration in your custom extension.



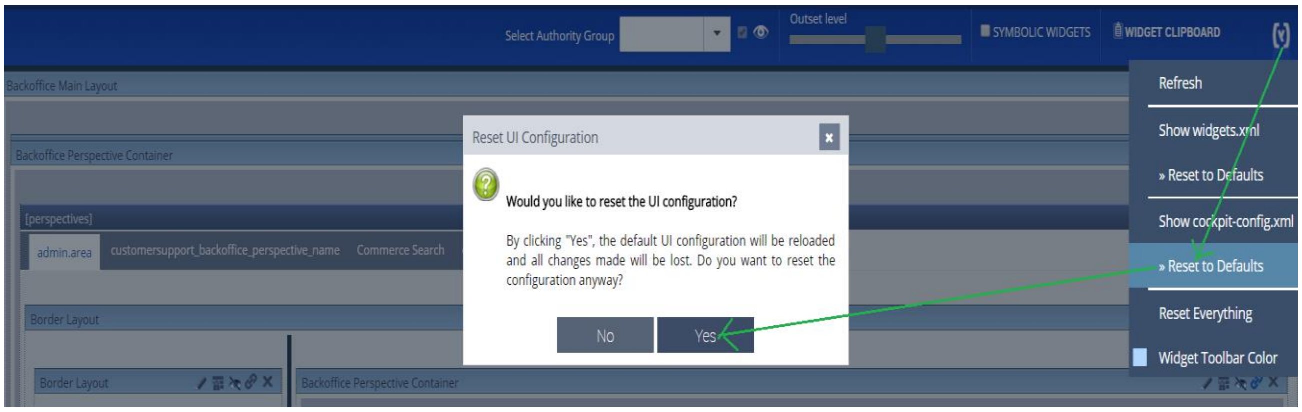
```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<config xmlns="http://www.hybris.com/cockpit/config">
    <context type="com.hybris.cockpitng.util.Range" comp
        <y:base xmlns:y="http://www.hybris.com/cockpit/c
            <y:labels beanId="rangeLabelProvider"/>
        </y:base>
    </context>
    <context type="java.lang.String" component="simple-l
        <ysl:simple-list xmlns:ysl="http://www.hybris.co
            <ysl:name field="#root"/>
            <ysl:description field="class.canonicalName"
        </ysl:simple-list>
    </context>
    <context type="java.lang.Object" component="listview
```

(2) To view these UI changes made in chennatrainingcourses-backoffice-config.xml without having to do a rebuild or restart follow the below steps.

a. Switch to the Application Orchestrator mode by pressing **F4** (you'll need to be logged in as an administrator).

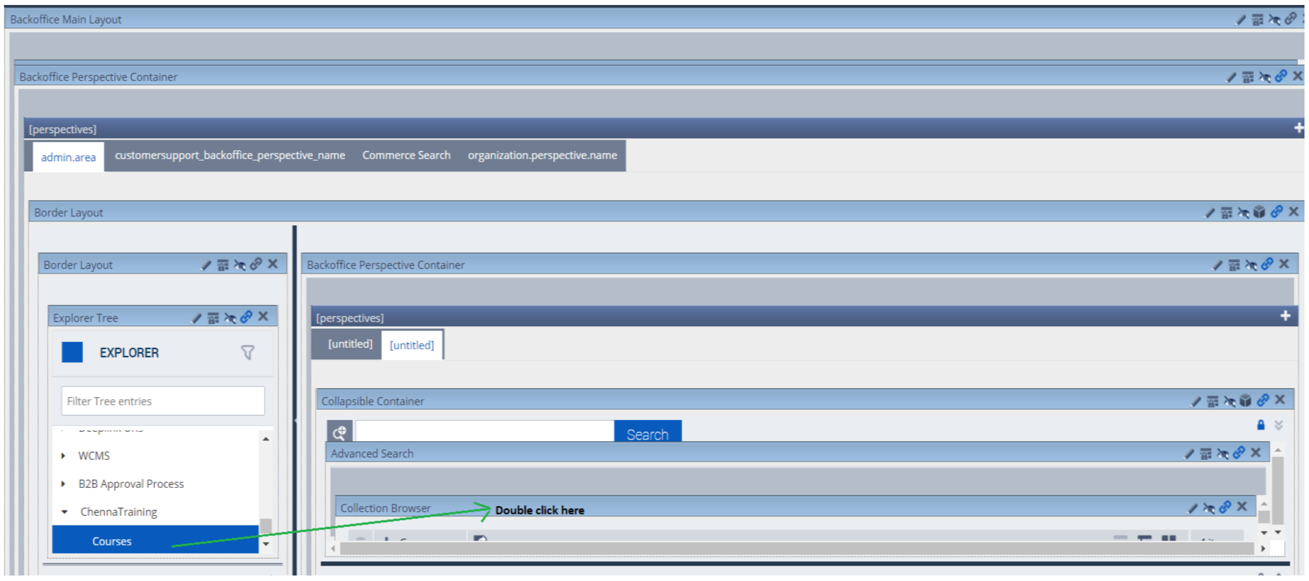
b. Click the Hybris logo in the upper right corner of the screen and select the **Reset to Defaults** option under **Show cockpit-config.xml**

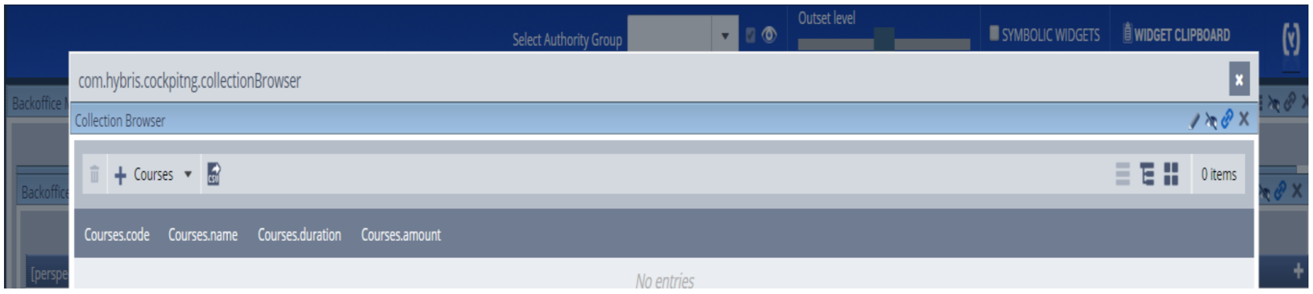
c. After the page reloads, press **F4** again to exit Application Orchestrator mode



Would you like to reset the UI configuration?

By clicking "Yes", the default UI configuration will be reloaded and all changes made will be lost. Do you want to reset the configuration anyway?





✓ **Step 7=** Define the table structure in “chennatrainingcourses-items.xml” file \hybris\bin\hybris\bin\custom\chennatrainingcourses\resources\chennatrainingcourses-items.xml file and define the table structure

```
<items  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamespaceSchemaLocation="items.xsd">
    <itemtypes>
        <itemtype code="Courses" generate="true" autocreate="true">
            <deployment table="TrainingCourses" typecode="10128" />
            <attributes>
            <attribute qualifier="code" type="java.lang.String" >
                <persistence type="property"/>
                <modifiers optional="false" unique="true"/>
            </attribute>
            <attribute qualifier="name" type="java.lang.String">
                <persistence type="property" />
                <modifiers optional="false" />
            </attribute>
            <attribute qualifier="duration" type="java.lang.String">
                <persistence type="property" />
                <modifiers optional="false" />
            </attribute>
            <attribute qualifier="amount" type="java.lang.Integer">
                <persistence type="property" />
                <modifiers optional="false" />
            </attribute>
            </attributes>
        </itemtype>
    </itemtypes>
</items>
```

✓ **Step 8=** Perform below steps: -

a) C:\Softwares\HYBRISCOMM6200P\hybris\bin\platform>**setantenv.bat**

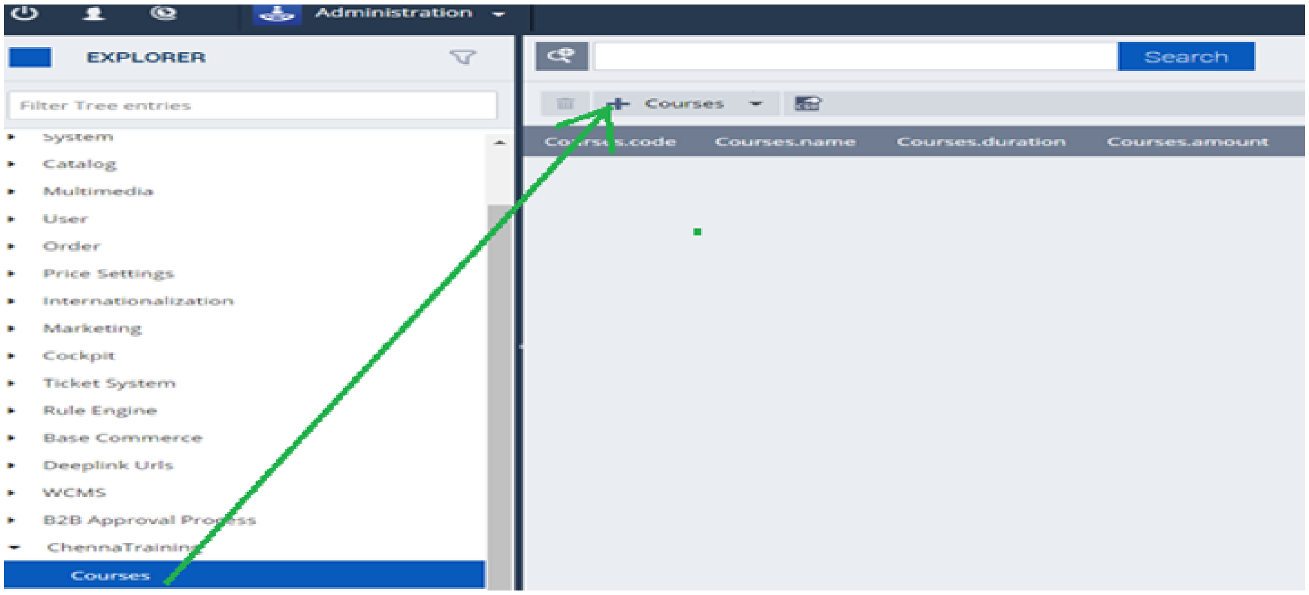
b) C:\Softwares\HYBRISCOMM6200P\hybris\bin\platform>**ant clean all**

c) C:\Softwares\HYBRISCOMM6200P\hybris\bin\platform>**hybrisserver.bat**

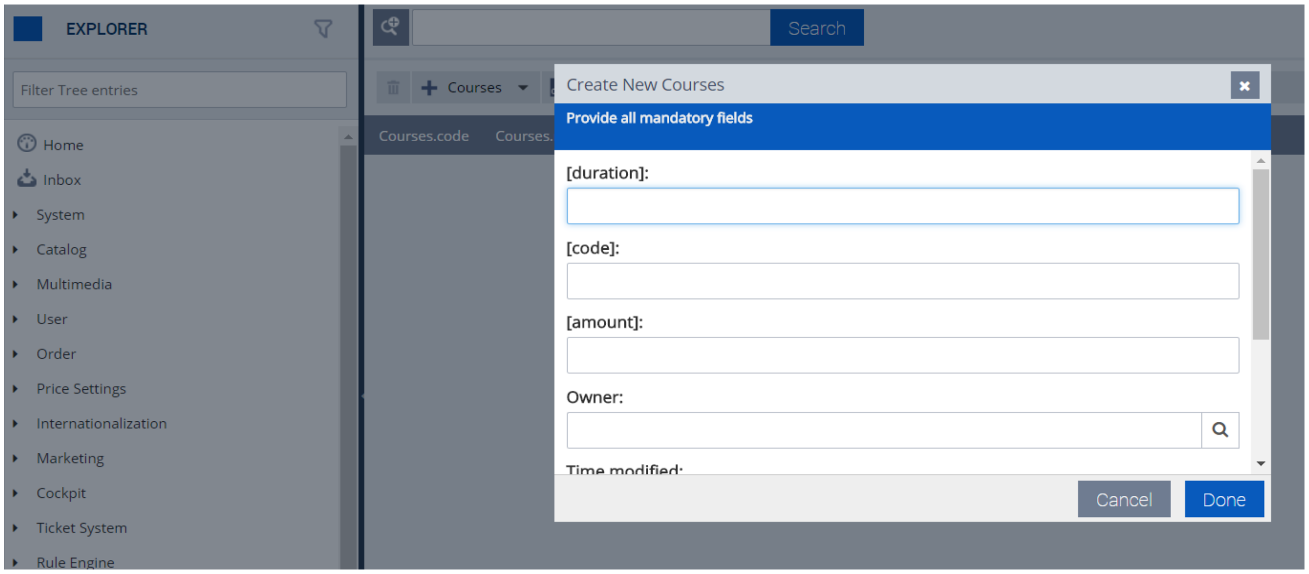
d) https://localhost:9002 → platform → update (Select only "Creating essential data") → update

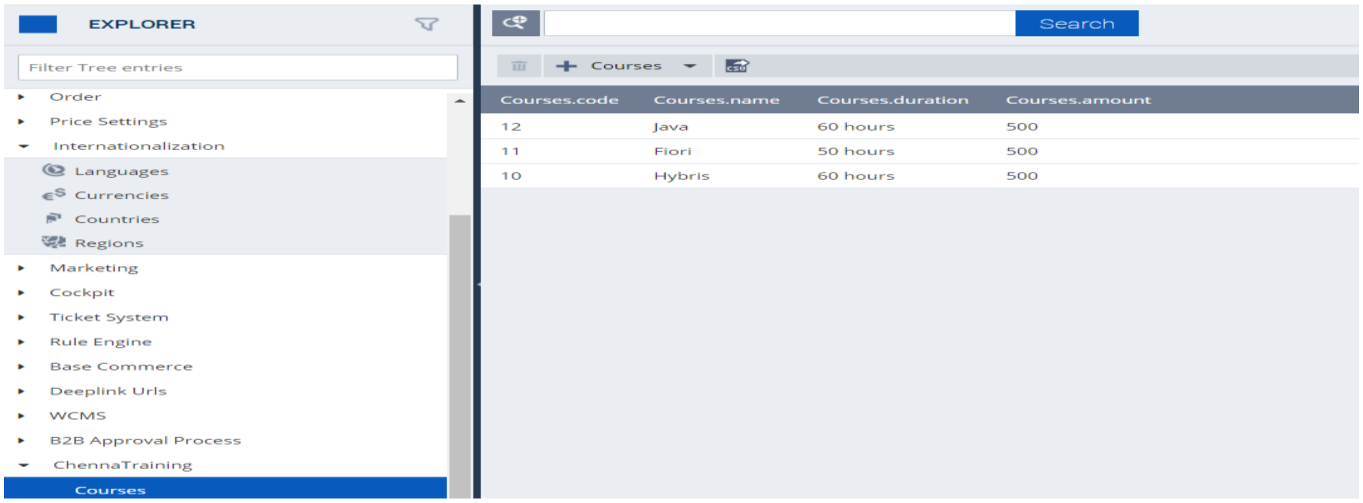
✓ **Step 9=** Goto backoffice and see the results of the new extension (There should "ChennaTraining" node with "Courses" as subnode / element.

https://localhost:9002/backoffice



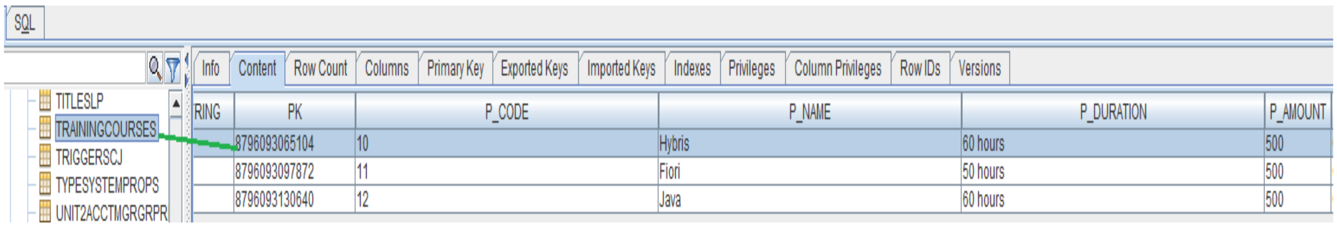
✓ **Step 10 =** Add the new course details by clicking + sign next to the Courses and Save





Courses.code	Courses.name	Courses.duration	Courses.amount
12	Java	60 hours	500
11	Fiori	50 hours	500
10	Hybris	60 hours	500

✓ **Step 11 =** Verify the course details exists in “**TrainingCourses”** table in database



RING	PK	P_CODE	P_NAME	P_DURATION	P_AMOUNT
	8796093065104	10	Hybris	60 hours	500
	8796093097872	11	Fiori	50 hours	500
	8796093130640	12	Java	60 hours	500

Q = How to change the label names in the back office?

Ans = By using **localization** you can change the label names

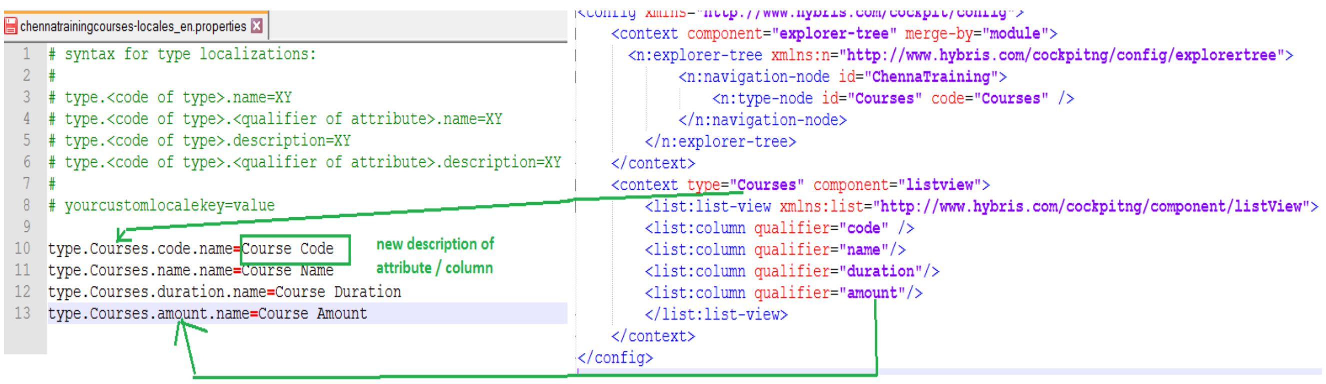
✓ **Step 12 =** Change the Course related label names using localization

(a) Goto

C:\Softwares\HYBRISCOMM6200P\hybris\ bin\custom\chennatrainingcourses\resources\localization

folder and add the localization in the proper file (let’s do the localization for English)

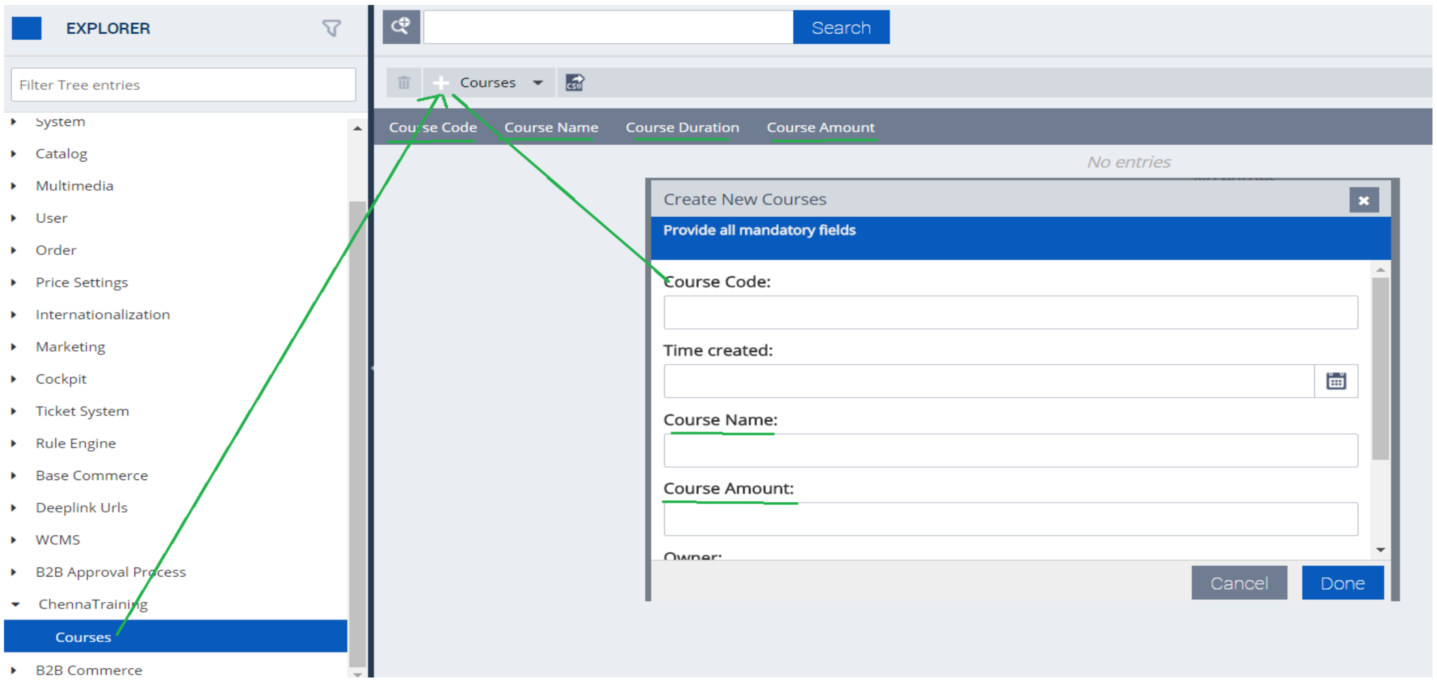
Filename = chennatrainingcourses-locales_en.properties

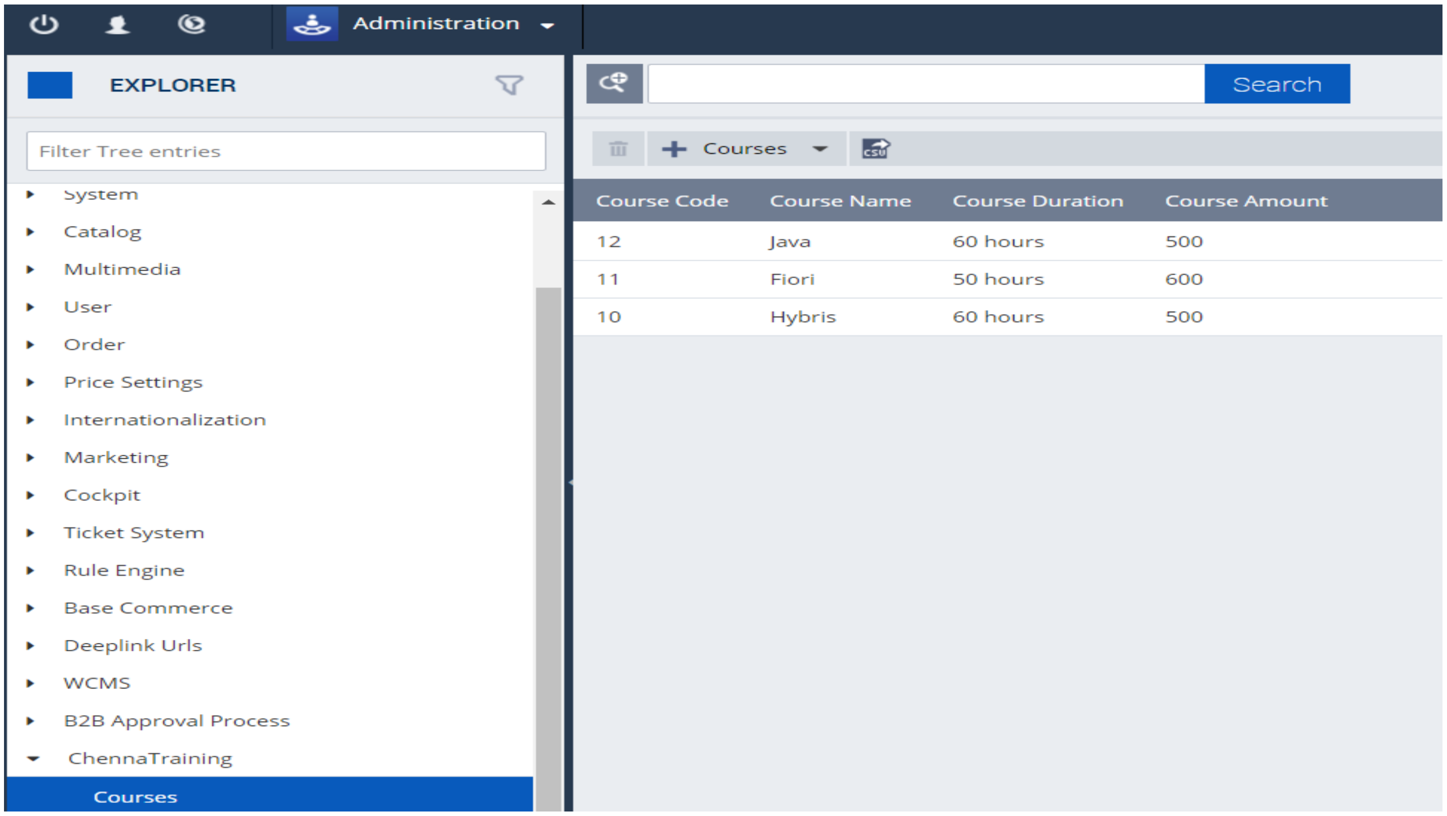


```
# syntax for type localizations:
#
# type.<code of type>.name=XY
# type.<code of type>.<qualifier of attribute>.name=XY
# type.<code of type>.description=XY
# type.<code of type>.<qualifier of attribute>.description=XY
#
# yourcustomlocalekey=value

type.Courses.code.name=Course Code
type.Courses.name.name=Course Name
type.Courses.duration.name=Course Duration
type.Courses.amount.name=Course Amount
```

```
<config xmlns="http://www.hybris.com/cockpit/config">
    <context component="explorer-tree" merge-by="module">
      <n:explorer-tree xmlns:n="http://www.hybris.com/cockpitng/config/explorertree">
            <n:navigation-node id="ChennaTraining">
                <n:type-node id="Courses" code="Courses" />
            </n:navigation-node>
        </n:explorer-tree>
    </context>
    <context type="Courses" component="listview">
        <list:list-view xmlns:list="http://www.hybris.com/cockpitng/component/listView">
        <list:column qualifier="code" />
        <list:column qualifier="name"/>
        <list:column qualifier="duration"/>
        <list:column qualifier="amount"/>
        </list:list-view>
    </context>
</config>
```

(b) https://localhost:9002 → platform → update (Select only "Creating essential data") → update



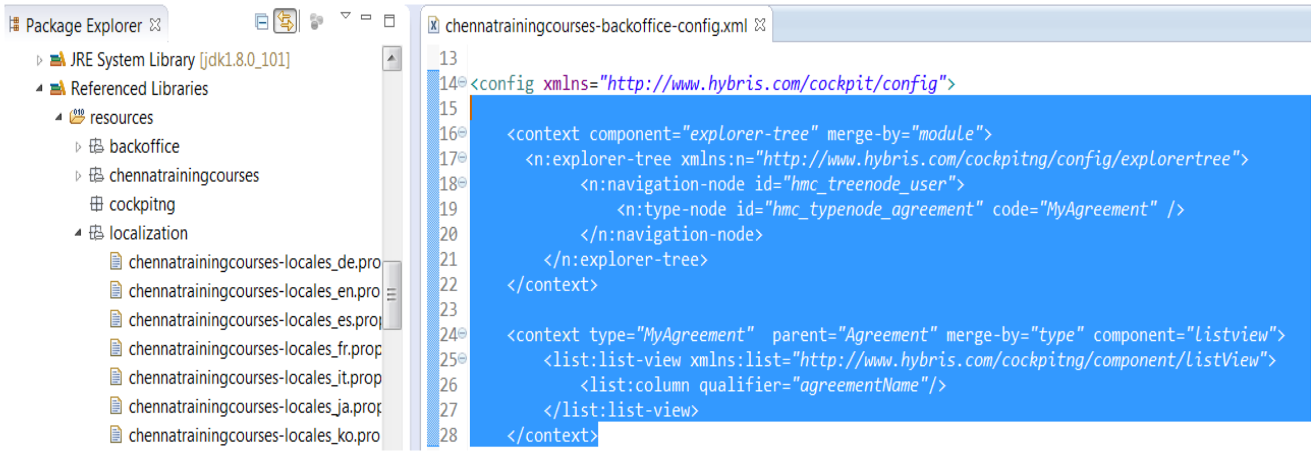


Course Code	Course Name	Course Duration	Course Amount
12	Java	60 hours	500
11	Fiori	50 hours	600
10	Hybris	60 hours	500

Q = How to add a new field / new attribute in the back office to the existing node? / How to customize the back office with new attribute in the existing node?

Scenario = Let's add "Agreement Name" in User → Agreements

✓ **Step 1** = Goto /chennatrainingcourses/resources/chennatrainingcourses-backoffice-config.xml file and add the new attribute in the xml config file and save the changes

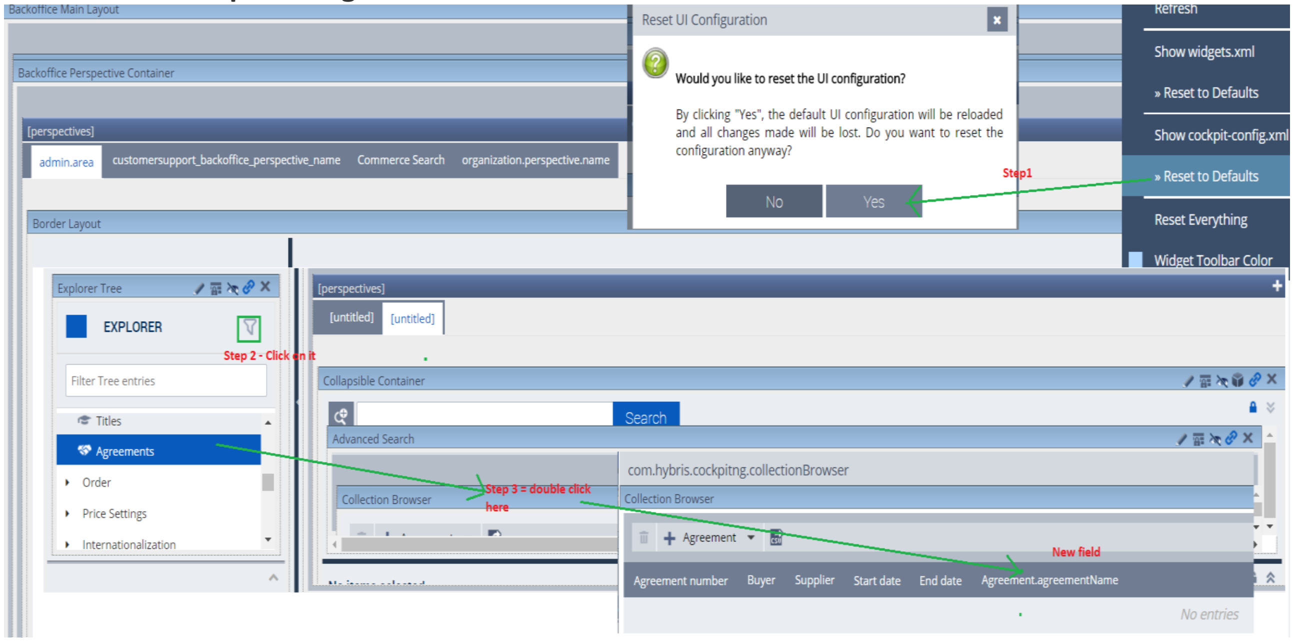

Package Explorer
JRE System Library [jdk1.8.0_101]
Referenced Libraries
resources
backoffice
chennatrainingcourses
cockpitng
localization
chennatrainingcourses-locales_de.pro
chennatrainingcourses-locales_en.pro
chennatrainingcourses-locales_es.prop
chennatrainingcourses-locales_fr.prop
chennatrainingcourses-locales_it.prop
chennatrainingcourses-locales_ja.prop
chennatrainingcourses-locales_ko.pro
chennatrainingcourses-backoffice-config.xml

```
<config xmlns="http://www.hybris.com/cockpit/config">

    <context component="explorer-tree" merge-by="module">
      <n:explorer-tree xmlns:n="http://www.hybris.com/cockpitng/config/explorertree">
            <n:navigation-node id="hmc_treenode_user">
                <n:type-node id="hmc_typenode_agreement" code="MyAgreement" />
            </n:navigation-node>
        </n:explorer-tree>
    </context>

    <context type="MyAgreement"  parent="Agreement" merge-by="type" component="listview">
        <list:list-view xmlns:list="http://www.hybris.com/cockpitng/component/listView">
            <list:column qualifier="agreementName"/>
        </list:list-view>
    </context>
```

✓ **Step 2 =** Goto https://localhost:9002/backoffice and follow these steps to see the new field

(a) Switch to the Application Orchestrator mode by pressing **F4** (you'll need to be logged in as an administrator).

(b) Click the Hybris logo in the upper right corner of the screen and select the **Reset to Defaults** option under **Show cockpit-config.xml**

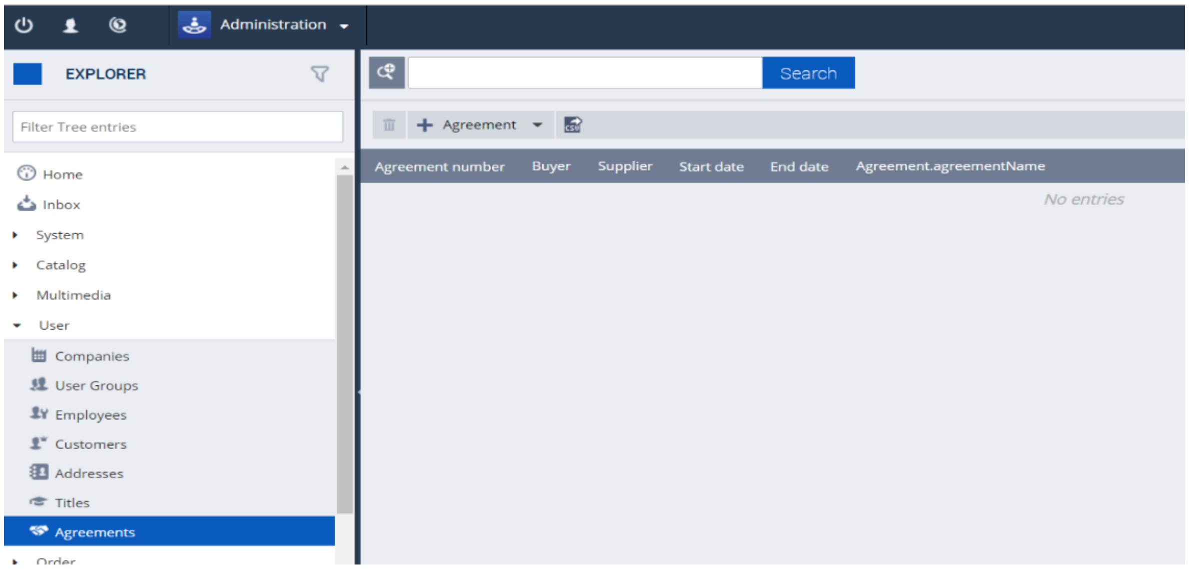


Would you like to reset the UI configuration?

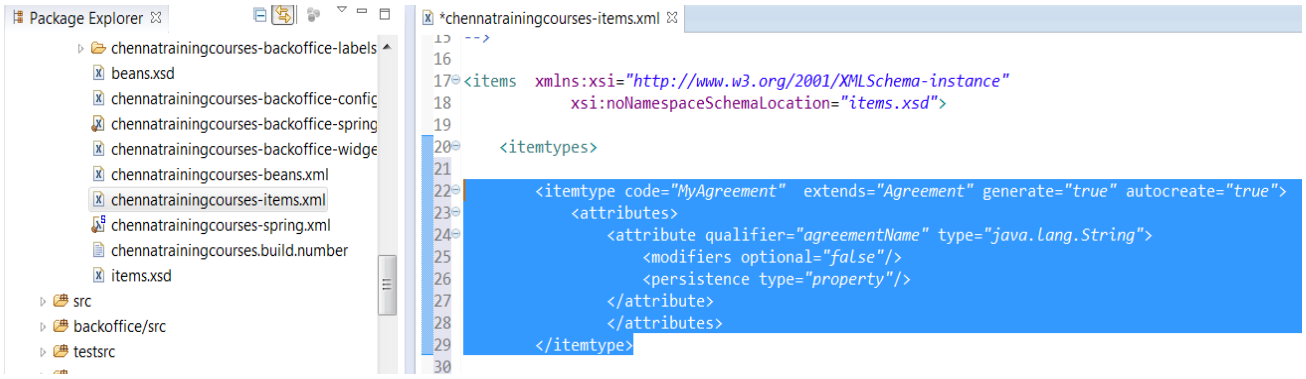
By clicking "Yes", the default UI configuration will be reloaded and all changes made will be lost. Do you want to reset the configuration anyway?

(c) After the page reloads, press **F4** again to exit Application Orchestrator mode

Contact us for more information → chennareddytraining@gmail.com (Java–Salesforce–SAP Portal–UI5/Fiori–Hybris)



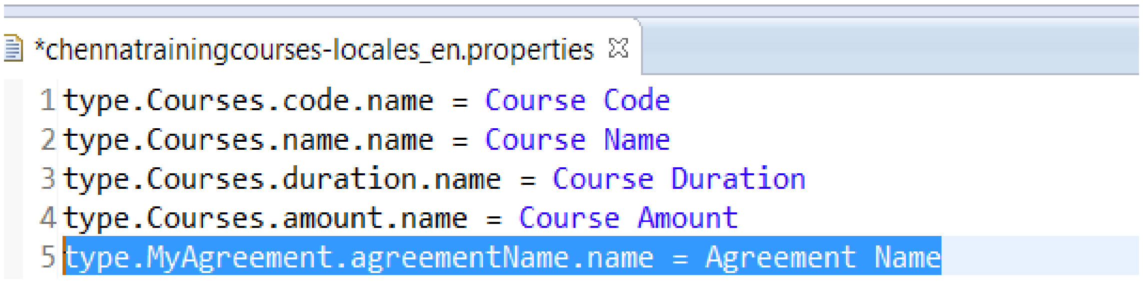
✓ **Step 3 =** Define the attribute in /chennatrainingcourses/resources/chennatrainingcourses-items.xml file & save the file


Package Explorer
chennatrainingcourses-backoffice-labels
beans.xsd
chennatrainingcourses-backoffice-config
chennatrainingcourses-backoffice-spring
chennatrainingcourses-backoffice-widge
chennatrainingcourses-beans.xml
chennatrainingcourses-items.xml
chennatrainingcourses-spring.xml
chennatrainingcourses.build.number
items.xsd
src
backoffice/src
testsrc
*chennatrainingcourses-items.xml

```
-->

<items  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:noNamespaceSchemaLocation="items.xsd">

    <itemtypes>

        <itemtype code="MyAgreement"  extends="Agreement" generate="true" autocreate="true">
            <attributes>
                <attribute qualifier="agreementName" type="java.lang.String">
                    <modifiers optional="false"/>
                    <persistence type="property"/>
                </attribute>
                </attributes>
        </itemtype>
```

✓ **Step 4 =** Change the label name using localization

Goto /chennatrainingcourses/resources/localization/chennatrainingcourses-locales_en.properties" file and add this line & Save the file



```
type.Courses.code.name = Course Code
type.Courses.name.name = Course Name
type.Courses.duration.name = Course Duration
type.Courses.amount.name = Course Amount
type.MyAgreement.agreementName.name = Agreement Name
```

✓ **Step 5 = =** Perform below steps: -

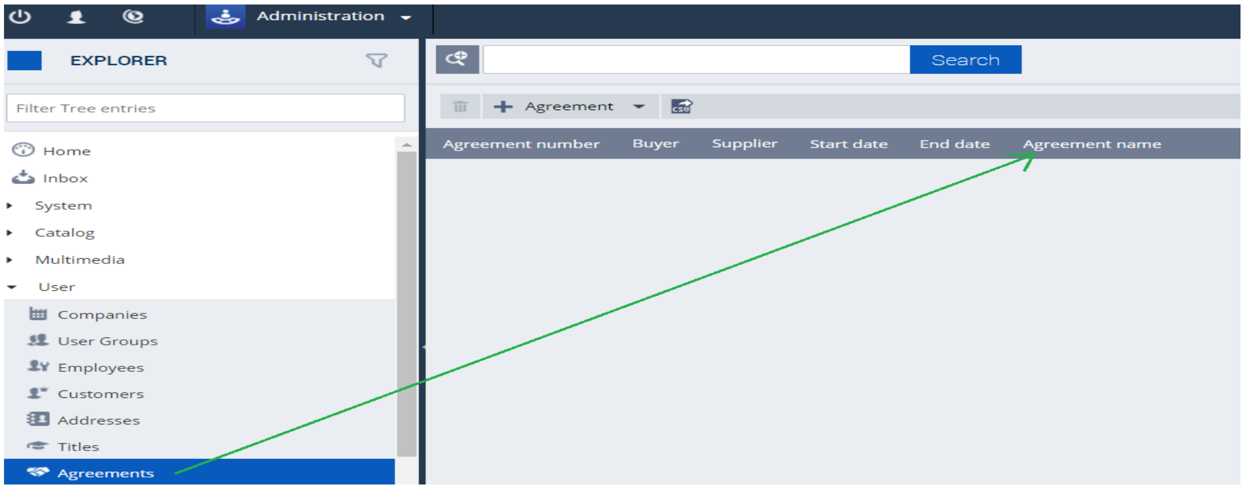
a) C:\Softwares\HYBRISCOMM6200P\hybris\bin\platform>**setantenv.bat**

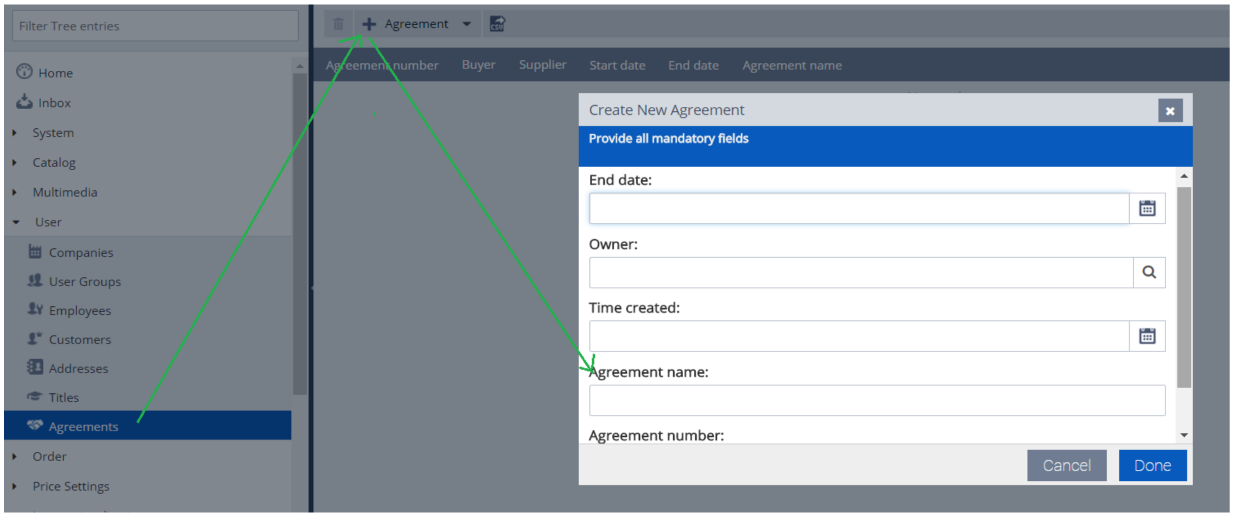
b) C:\Softwares\HYBRISCOMM6200P\hybris\bin\platform>**ant clean all**

c) C:\Softwares\HYBRISCOMM6200P\hybris\bin\platform>**hybrisserver.bat**

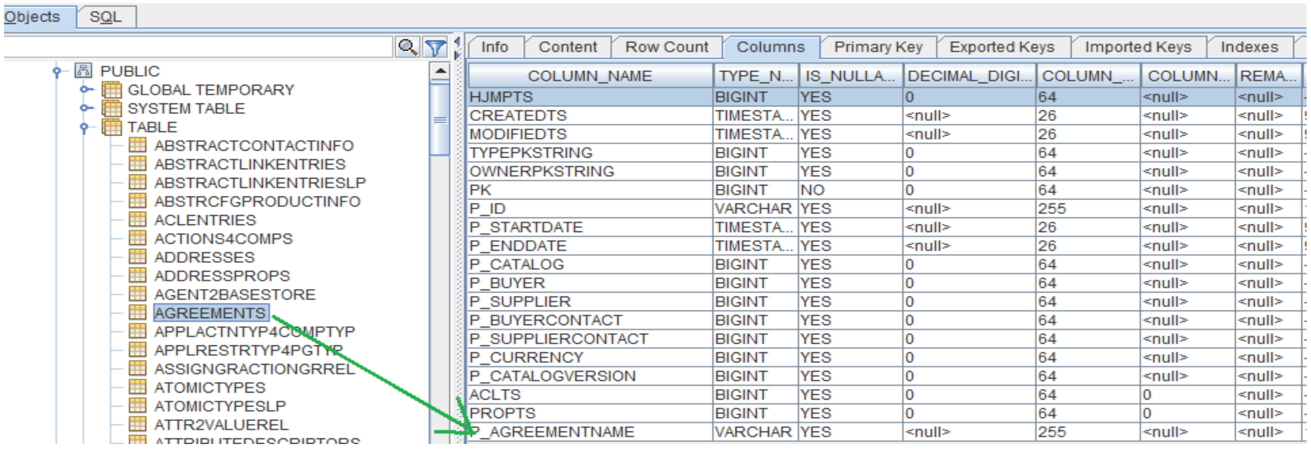
d) Perform update - https://localhost:9002 → platform → update

e) Goto https://localhost:9002/backoffice and see the results



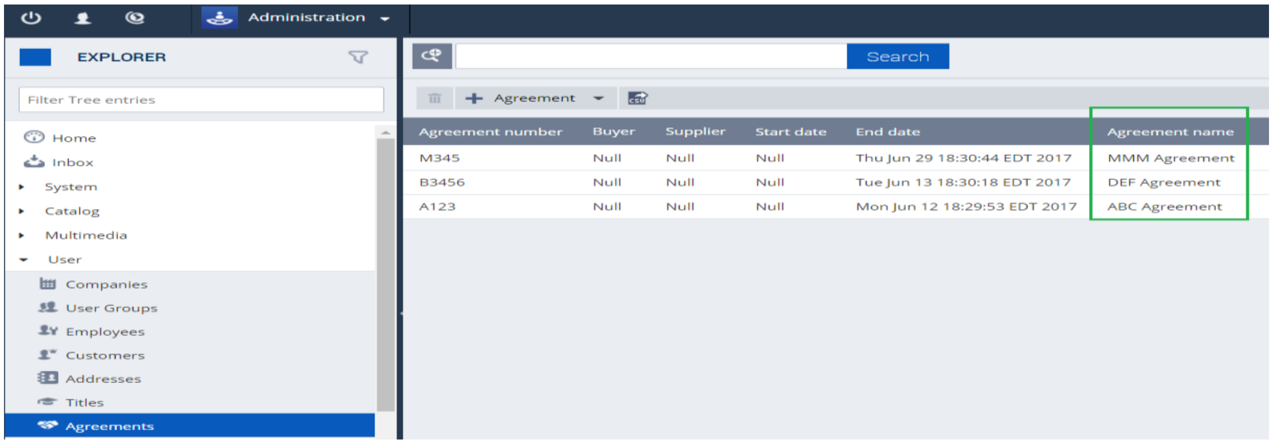


✓ **Step 6 =** Verify the new attribute in Agreements database table



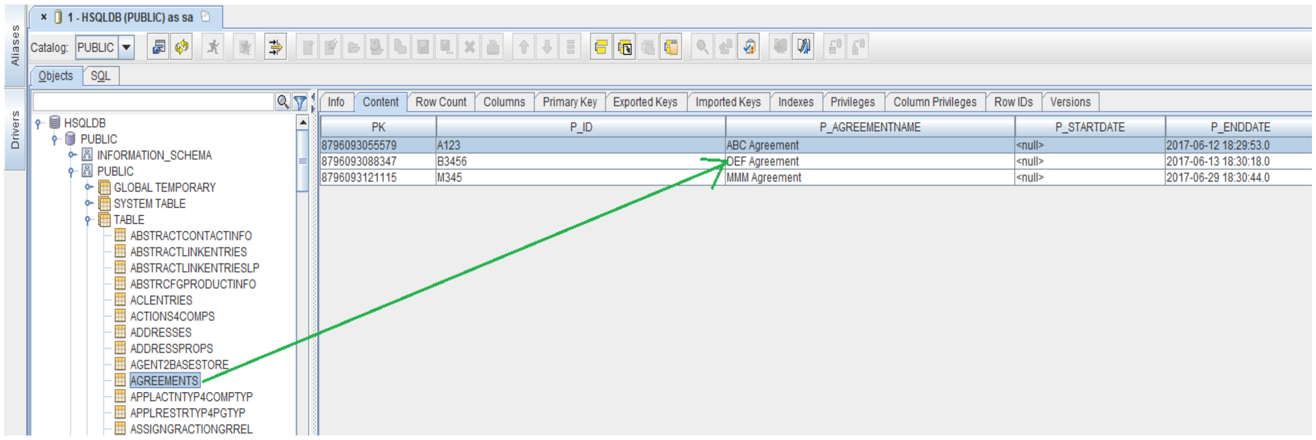
COLUMN_NAME	TYPE_N...	IS_NULLA...	DECIMAL_DIGI...	COLUMN_...	COLUMN...	REMA...
HJMPTS	BIGINT	YES	0	64	<null>	<null>
CREATEDTS	TIMESTA...	YES	<null>	26	<null>	<null>
MODIFIEDTS	TIMESTA...	YES	<null>	26	<null>	<null>
TYPEPKSTRING	BIGINT	YES	0	64	<null>	<null>
OWNERPKSTRING	BIGINT	YES	0	64	<null>	<null>
PK	BIGINT	NO	0	64	<null>	<null>
P_ID	VARCHAR	YES	<null>	255	<null>	<null>
P_STARTDATE	TIMESTA...	YES	<null>	26	<null>	<null>
P_ENDDATE	TIMESTA...	YES	<null>	26	<null>	<null>
P_CATALOG	BIGINT	YES	0	64	<null>	<null>
P_BUYER	BIGINT	YES	0	64	<null>	<null>
P_SUPPLIER	BIGINT	YES	0	64	<null>	<null>
P_BUYERCONTACT	BIGINT	YES	0	64	<null>	<null>
P_SUPPLIERCONTACT	BIGINT	YES	0	64	<null>	<null>
P_CURRENCY	BIGINT	YES	0	64	<null>	<null>
P_CATALOGVERSION	BIGINT	YES	0	64	<null>	<null>
ACLTS	BIGINT	YES	0	64	0	<null>
PROPTS	BIGINT	YES	0	64	0	<null>
P_AGREEMENTNAME	VARCHAR	YES	<null>	255	<null>	<null>

✓ **Step 7 =** Enter the data in the UI & save



Agreement number	Buyer	Supplier	Start date	End date	Agreement name
M345	Null	Null	Null	Thu Jun 29 18:30:44 EDT 2017	MMM Agreement
B3456	Null	Null	Null	Tue Jun 13 18:30:18 EDT 2017	DEF Agreement
A123	Null	Null	Null	Mon Jun 12 18:29:53 EDT 2017	ABC Agreement

✓ **Step 8** = Validate the agreement name stored in the "Agreements" table



PK	P_ID	P_AGREEMENTNAME	P_STARTDATE	P_ENDDATE
8796093055579	A123	ABC Agreement	<null>	2017-06-12 18:29:53.0
8796093088347	B3456	DEF Agreement	<null>	2017-06-13 18:30:18.0
8796093121115	M345	MMM Agreement	<null>	2017-06-29 18:30:44.0

Q: How to add a new attribute to edit view area in Product Cockpit?

Step 1 = Create a new Model (**MyProduct**) that extends **ProductModel** in **Items.xml**, add attribute in this model:-

```
"<itemtype code="MyProduct" extends="Product">
<description>my product that contains additional attributes.</description>
<attributes>
<attribute qualifier="attribute1" type="localized:java.lang.String">
<description>example for product cockpit</description>
</attribute>
</attributes>
</itemtype>"
```

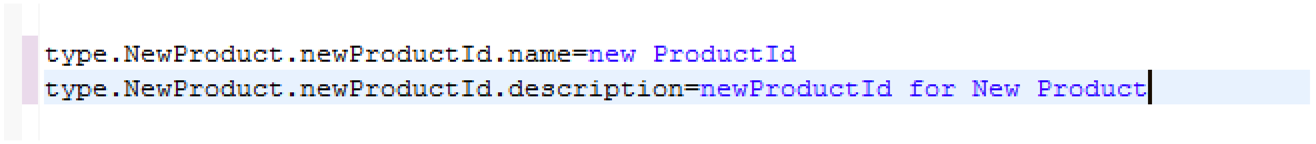
Note: - Now Build the project, Start Server & Perform Update Operation.

Step 2 = Create **editorArea_MyProduct.xml** file in your Cockpit extension & add key for new attribute in xml file

```
<group qualifier="General" visible="true" initially-opened="true">
    <label key="cockpit.config.label.General" />
    <property qualifier="product.code" />
    <property qualifier="product.name" />
    <property qualifier="product.catalogversion"  editor="shortListEditor"/
    <property qualifier="product.approvalStatus" />
    <property qualifier="product.onlineDate" />
    <property qualifier="product.offlineDate" />
    <property qualifier="product.unit" />
    <property qualifier="product.ean" />
    <property qualifier="product.description" />
    <property qualifier="product.summary" />
    <property qualifier="myProduct.attribute1" />
</group>
```

Contact us for more information → chennareddytraining@gmail.com (Java–Salesforce–SAP Portal–UI5/Fiori–Hybris)

Step 3 = Localized the key name in **ProjetcNamecockpits-locales_en.properties** and i3-label.properties file.

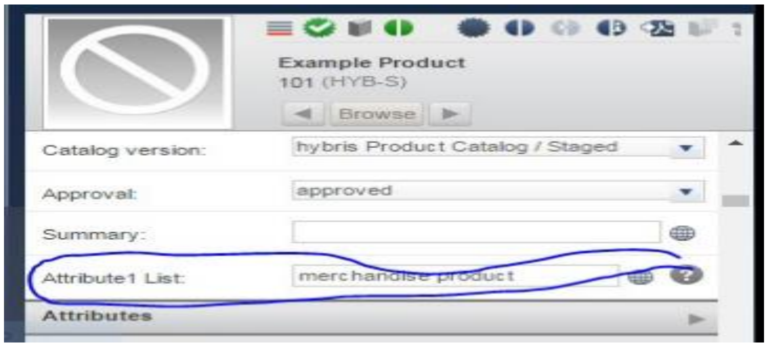


```
type.NewProduct.newProductId.name=new ProductId
type.NewProduct.newProductId.description=newProductId for New Product
```

Step 4 = Restart server & do update from HAC.

Open **ProductCockpit** in url "**http://localhost:9001/productcockpit&#8221**; and search your product which you added in "**MyProduct**" model .

Open that product and in left side you will find your new attribute in **Editor_area**.



Example Product
101 (HYB-S)

Catalog version:	hybris Product Catalog / Staged
Approval:	approved
Summary:	
Attribute1 List:	merchandise product

Note: - Hybris Cockpit framework is highly customizable. Depending on amount need, we can distinguish different levels of customization, easy, medium, and expert.

- ✓ **Easy Customization =** Can be done by configuration within existing cockpit & needs no own implementation. The hybris Cockpit framework enables to configure user interface (UI) components through XML. Each configuration is defined by a code, an object template, UI role, and a XML configuration file. We can change the configuration defining how an item type is represented in lists, editors, advanced search, or reference selectors.
- ✓ **Medium Customization =** Needs only little implementation, because it can be done by using an existing cockpit as a template to be modified. Need valid **ZK Studio** Enterprise Edition license for medium customization. We can:
  - Add own editors for all view types to the system and make them available for configuration.
  - Add custom section to editor area.
  - Add custom column to list view.
  - Add custom sections to navigation or editor area.
- ✓ **Expert Customization =** Needs experienced implementation, because it uses the hybris Cockpit framework independently from the existing cockpits. We need a valid ZK Studio Enterprise Edition license for expert customization. We can:
  - Customize type service and change default types, created by default on base of items and add the custom extended types to the cockpit type system.
  - Implement own perspective by customizing all areas, these are Navigation Area, Browser Area, and Editor Area.
  - Create your own wizard for creating items, pages and page flows.

Q: What are the cockpit related xml files: We have to give the file name as same below. in place of * - we have to give the name of component name.

contentEditor_*.xml	editorArea_*.xml	wizardConfig_*.xml
advancedSearch_*.xml	base_*.xml	

Note: - If U understand how to do customization inside hMC. Then Customization of other Cockpits is similar.

-- **Product Cockpit =** You can see Navigation Section, Browser Section, Edit Area, ...

Q: How to add something in "Right Side Editor Area" options (Adding another tab -- Prices/Variants /...)

Ans: - Open "**editorArea_Product.xml** (--chennatrainingcockpits/resources/chennatrainingcockpits-config/cockpitgroup/editorArea_Product.xml

You can see "General, Category, Price /..." tabs.
To create new tab, you need to create new group.

Hence, configure new xml, create new group & add properties inside that group.

In General, Cockpit customization means, adding tabs & properties. (They don't worry about colors and all).

Q: Another Requirement. Inside Browser Area (Go to List View Instead Grid View). You will see columns like "Thumbnail, Status, Article Number...". How to add one more columns here?

-- Eclipse -- listViewContentBrowser_Product.xml (--chennatrainingcockpits/resources/chennatrainingcockpits-config/cockpitgroup/listViewContentBrowser_Product.xml). Add the properties here.

Whatever you want to see the properties, make them as "visible=true". Else visible = false.

Note: - Similarly, We have many XML files & each is having its own purpose for controlling the Cockpits.

Q: If you want to change the Grid View properties? = **gridView_Product.xml**

Q: Whenever you trying to do the search (Advanced Search), you will get the options. Let's say you want to add more options / Sort Options / ...?= **base_Product.xml**

Here you can see the Search Properties, Sort properties.

Note: - For localization of Cockpits, we have separate localization properties files.

-- chennatrainingcockpits/resources/localization/chennatrainingcockpits-locales_en.properties

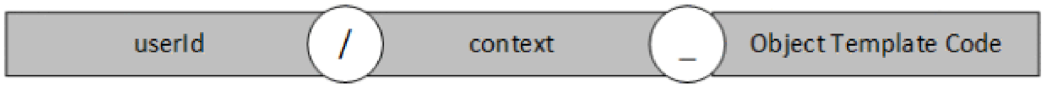
Q: Explain Relative URLs in Hybris Cockpits?

Generally, in several businesses, it is required to share direct URLs of order, cart or product with internal teams. **Example: -** After order is placed → (1) Send confirmation email to customer. (2) Also send email to a team of internal staff with direct URL of that order (So that they can see directly without going Cockpits).

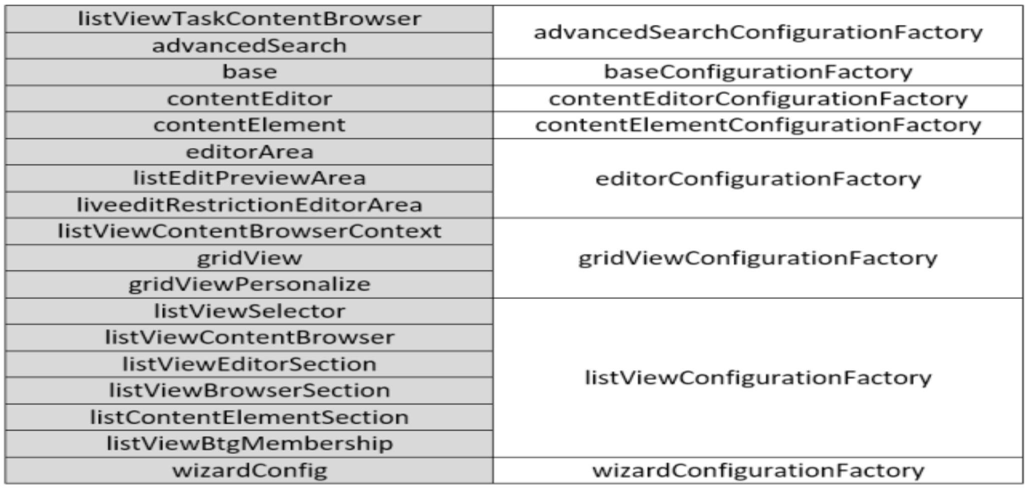
Solution = https://<xxx>:<port>/cscockpit/index.zul?persp=cscockpitPerspective&events=activation&act-item=8796093055021

A request event handler parses request parameters and dispatches a Cockpit business event.
According to above URL, Activation request events is triggered for order item with PK – 8796093055021

- **Explain Hybris Cockpit Configuration?**
  Cockpit FWK uses config files for customizing UI components. There are 3 levels (**Easy, Medium & Expert**) customizations. Easy customization = XML files.
  Cockpit configuration XML will have naming conventions = Example: - "editorArea_Product.xml"



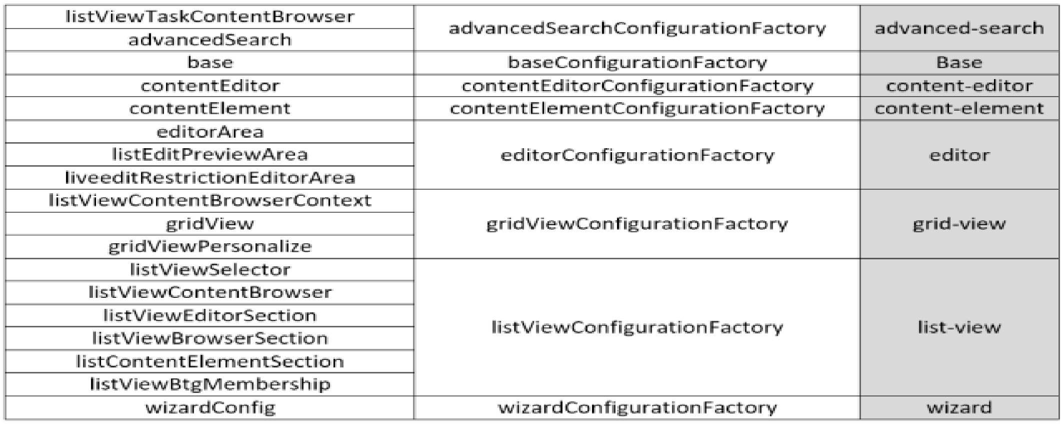
  User ID = Directory and **_.xml** is template used for naming.
  Contexts =



listViewTaskContentBrowser	advancedSearchConfigurationFactory
advancedSearch	
base	baseConfigurationFactory
contentEditor	contentEditorConfigurationFactory
contentElement	contentElementConfigurationFactory
editorArea	editorConfigurationFactory
listEditPreviewArea	
liveeditRestrictionEditorArea	
listViewContentBrowserContext	gridViewConfigurationFactory
gridView	
gridViewPersonalize	
listViewSelector	listViewConfigurationFactory
listViewContentBrowser	
listViewEditorSection	
listViewBrowserSection	
listContentElementSection	
listViewBtgMembership	
wizardConfig	wizardConfigurationFactory

Q: How to fetch the configuration for the Component?
  UIComponentConfiguration uiConfigService.getComponentConfiguration(objectTemplate, , <*Configuration.class>)

Object Template Code = Mostly Name of the hybris item type (Product / CMSSite.AcceleratorWebsite /…)
Configuration file syntax = All Configs are XMLs & have XSD. All XMLs have 1 root element (Right Col):



listViewTaskContentBrowser	advancedSearchConfigurationFactory	advanced-search
advancedSearch		
base	baseConfigurationFactory	Base
contentEditor	contentEditorConfigurationFactory	content-editor
contentElement	contentElementConfigurationFactory	content-element
editorArea	editorConfigurationFactory	editor
listEditPreviewArea		
liveeditRestrictionEditorArea		
listViewContentBrowserContext	gridViewConfigurationFactory	grid-view
gridView		
gridViewPersonalize		
listViewSelector	listViewConfigurationFactory	list-view
listViewContentBrowser		
listViewEditorSection		
listViewBrowserSection		
listContentElementSection		
listViewBtgMembership		
wizardConfig	wizardConfigurationFactory	wizard

- **Explain Cockpit Framework Customization?**
  Cockpit FWK are there from many years. But now it's decommissioned. But due to many reasons, in old / new projects actively using this cockpit FWKs.
  Cockpit core architecture = Cockpit FWK is based on ZK FWK 3.6 (Rich internet app fwk).
  Starting page for all cockpits = **index.zul**

<window xmlns="http://www.zkoss.org/2005/zul" id="mainWin" mode="embedded" height="100%" sclass="hywin" use="de.hybris.platform.cockpit.components.impl.MainWindow" shadow="false" === >
</window>

Parameter "use" of tag "window" defines an implementation of the component.

Main window implementation class is de.hybris.platform.cockpit.components.impl.MainWindow.

MainWindow & Perspectives =

Perspectives = Similar to "tabs" in UI, but with different look.

Each perspective has same page structure, if you use built-in BasePerspectiveComponent.

MainWindow creates a UICockpitSession. It manages available cockpit perspectives.

Q: How to define your own Perspective? = By Creating bean as below: -

<bean id="MyCockpitSession" class="de.hybris.platform.cockpit.session.impl.UISessionImpl" scope="session" init-method="**registerAdditionalPerspectives**"> ====
<property name="availablePerspectives">
<list> <ref bean="MyPerspective1"/> <ref bean="MyPerspective2"/> </list>
</property> === </bean>

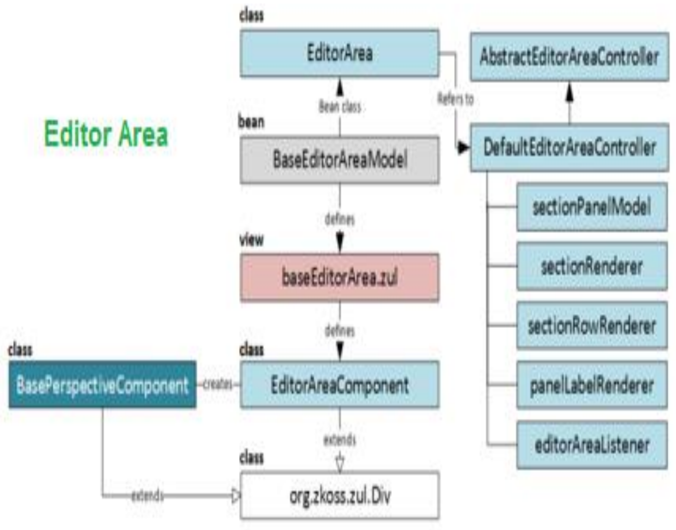
registerAdditionalPerspectives adds all perspectives defined in beans of **PerspectivePluginList.class** into a collection named avPerspUnrestricted (attribute of UICockpitSession).

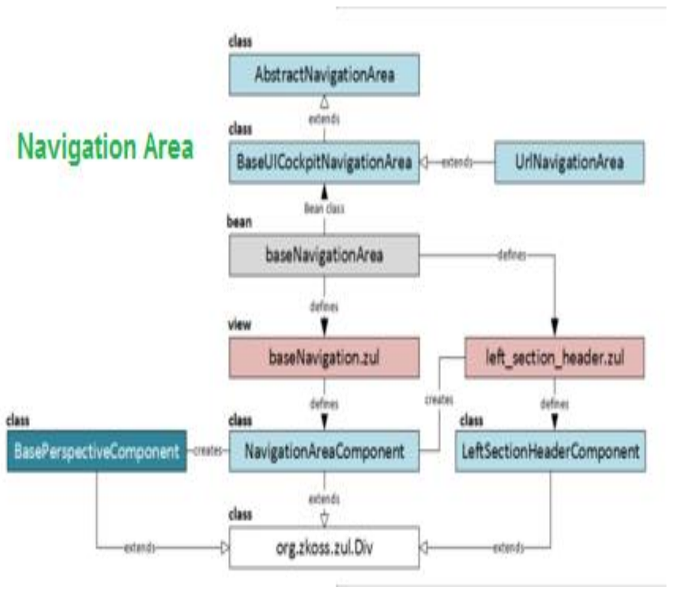
Cockpit Framework applies restrictions to this list, the results are saved in availablePerspectives.

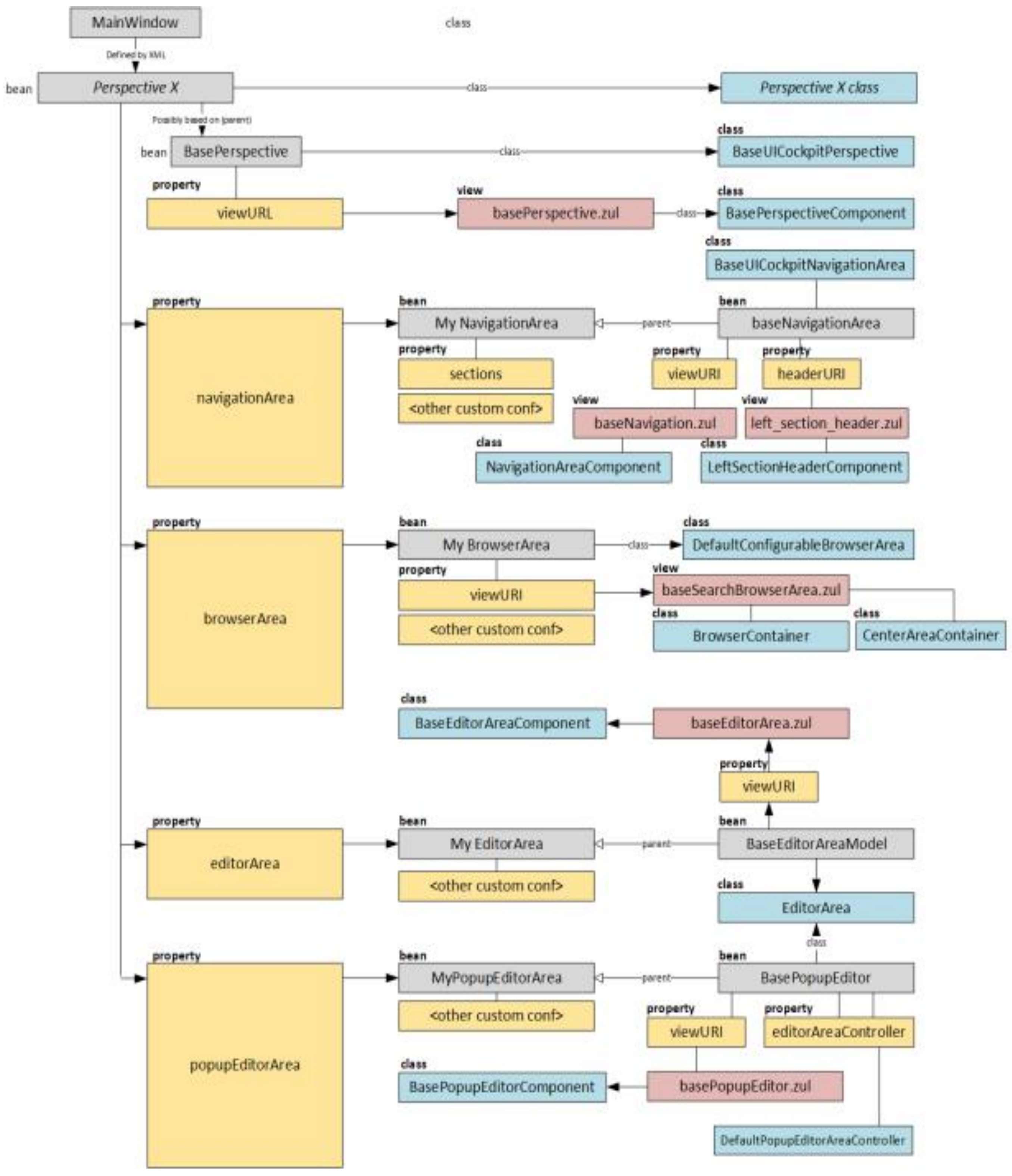
Q: What subcomponents are initialized by BasePerspectiveComponent?

- baseNavigationComponent in the **navigationArea**
- BasePopupEditorComponent in **popupEditorArea**
- BrowserContainer in **browserArea**
- BaseEditorAreaComponent in **editorArea**
- CustomCsaIncludeHelper in **customCsaURI** (*CSA.zul).

Editor & Navigation Areas: -





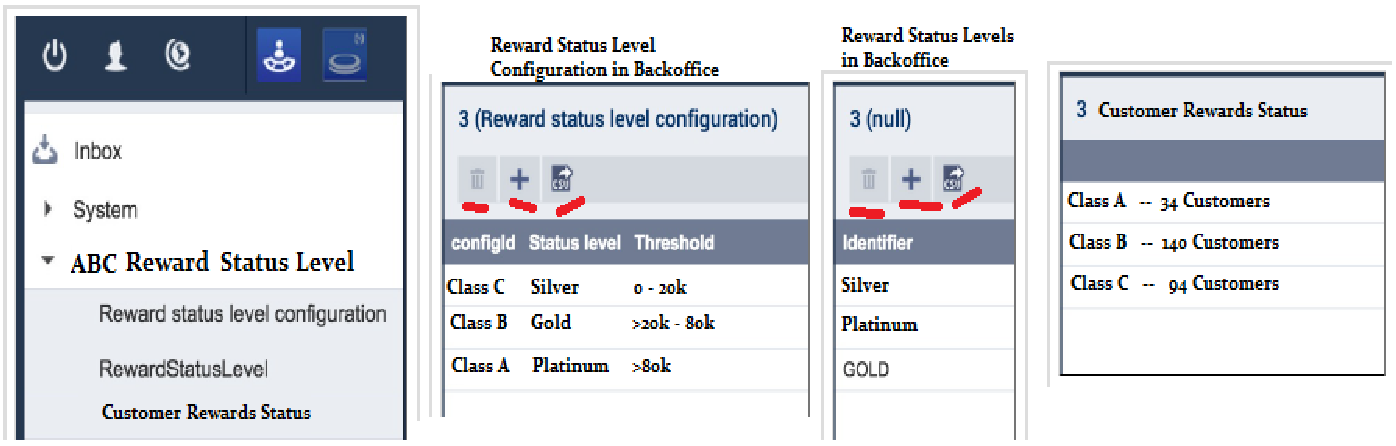


Contact us for more information → chennareddytraining@gmail.com (Java–Salesforce–SAP Portal–UI5/Fiori–Hybris)

Explain "Custom Backoffice Extension" (or) Cockpit NG Customization (or) Modify Backoffice?

Business Scenario: - ABC Classification is a frequently used analytical method to classify objects (Customers, Products or Employees) based on a particular measure (Revenue or Profit).
Your customers are classifying into three classes A, B and C according to the sales revenue they generate.

Threshold values used for individual ABC classes. Example: - All customers generating profit of **0 – 20K** belong to class C, **>20K – 80K** belong to class B, and **>80K** belong to **class A**.



Reward Status Level Configuration in Backoffice

3 (Reward status level configuration)

configId	Status level	Threshold
Class C	Silver	0 - 20k
Class B	Gold	>20k - 80k
Class A	Platinum	>80k

Reward Status Levels in Backoffice

3 (null)

Identifier
Silver
Platinum
GOLD

3 Customer Rewards Status

Class A -- 34 Customers
Class B -- 140 Customers
Class C -- 94 Customers

Note: - This solution is similar to above "ChennaTraining - Courses" example.

Solution =
Step 1 = Creating **ABCLevel Backoffice** Extension: -
Backoffice extension can be created by using **ant extgen** with "**ybackoffice**" template.
Name your extension = abclevelbackoffice
Package Name = com.sat.abclevelbackoffice
Note: - Do not create a default widget / css file (Answer = false to both)
Finally add it in "**localextensions.xml**" file.
Now run **ant all**

Now you have new extension & can import into your IDE.

Note: - If you see any error in IDE because of jar files, then add **5 jar** files (cockpitcore-2.0.5.2.jar, cockpitframework-2.0.5.2.jar, zcommon-6.5.1.1.jar, zk-6.5.1.1.jar & zul-6.5.1.1.jar) in build path.

Step 2 = Modify abclevelbackoffice-backoffice-config.xml to display: - **(1)** Rewards Status Level Configuration, **(2)** Reward Status Levels & **(3)** Customer Rewards Status as shown in above PIC.
${Path}/hybris/bin/custom/abclevelbackoffice/resources/abclevelbackoffice-backoffice-config.xml.

Q: Explain Backoffice? = Its next generation cockpit & replaces the cockpit FWK. In future all versions of existing cockpits (hmc, wcms, pcm…) will be replaced by new one using backoffice cockpit.

- ✓ **Q:** What are the key features of backoffice FWK?
  - Concept of Widget = It's an independent component that can be reused & connected to other.
  - Application orchestrator = Allows you to build a new backoffice from the backoffice itself.
- ✓ **Q:** How to create new Backoffice application? = ant extgen **ybackoffice**
- ✓ **Q:** What are important things inside **backoffice folder**?

Classes	src	testscr
Resources	.css	.zul

  cockpit-conig.xml = Default configuration for your widget.
  deinition.xml = Your widget definition
  labels = Folder for widget's localization
  editors / actions = folder for new editors and actions
  widgets = Widget components configuration
  extensioninfo.xml =
- ✓ **Q:** What is the shared configuration file in backoffice? = XXX- -backoffice-config.xml (where XXX = custom extension built from ybackoffice template).
- ✓ **Q:** How to enable backoffice application without restarting server during the backoffice development? = backoice.cockpitng.hotDeployment.enabled=true
- ✓ **Q:** What do we need to create new backoffice perspective?
  - Create a new **BackofficeRole** group
  - Create a new **UserGroup** member of the new **BackofficeRole** group and **employeegroup** group
  - Create **users** member of the new group
  - Update your backoffice configuration, add a new perspective by configuring principal attribute of contexts to your new backoffice role.
- ✓ **Legacy Cockpit Modifications =** Legacy cockpit FWK is removed in 2018. So just learn high level.
  - **Q:** How to import cockpit configuration?
    - Automatically (If you follow OOB naming convention)
    - Manually (Not recommended)
  - Q: To automatically import cockpit configuration what are the convention requires?
    - A folder **resources/-config**
    - Main folder cannot be empty and must contain at least:
      - XML configuration that affects all users and user groups
      - Folder with XML configuration inside. The folder name needs to be **user id** or a **group id** (validated during the import).
        It contain only XML files & no subfolders, inside this folder you have config files: -
      - They must match this pattern **contextID _ ObjectTemplateCode**.xml
    - Available context IDs are defined within the bean **ctxID2FactoryMappings**
    - •ObjectTemplateCode are item types and are validated by the type service