Contents

[1) How to add the extend OOB? 2](#_Toc176733213)

[2) How to add the Search Fields Customization by Configuration (Json config)? 4](#_Toc176733214)

[3) How to add the custom columns and actions in the existing order-search-result page’s table 7](#_Toc176733215)

[Add Custom Columns – adding Extn\_brand 7](#_Toc176733216)

[4) How to add/remove the custom columns in the existing Order-search-result page’s table? 10](#_Toc176733217)

[Remove OOB Columns 10](#_Toc176733218)

[B. add Columns 11](#_Toc176733219)

[5) Adding summary fields to existing detail pages 13](#_Toc176733220)

[6) Adding custom actions to existing pages [Customer Details] 14](#_Toc176733221)

[7) Adding custom actions to existing pages [Order Details] 17](#_Toc176733222)

[8) How to add the custom popup? 19](#_Toc176733223)

[a) overflowMenuActions 19](#_Toc176733224)

[b) Taskbar button with action 25](#_Toc176733225)

[c) Header button with action 27](#_Toc176733226)

[9) How to customize Homepage? 35](#_Toc176733227)

[a) Search order by phone number 35](#_Toc176733228)

[b) Hide Label 35](#_Toc176733229)

[c) Add Label 35](#_Toc176733230)

[10) adjust pricing [overflow action] add UOM column at line level 36](#_Toc176733231)

[11) Custom dropdown using mashup call? 37](#_Toc176733232)

[a. Carrier Service Code 37](#_Toc176733233)

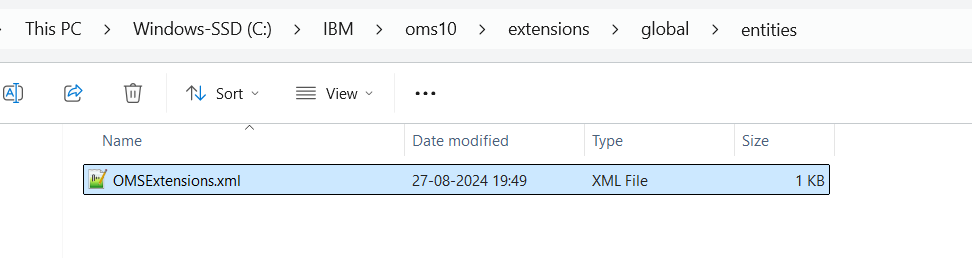
[b. Modification Reasons List 37](#_Toc176733234)

## How to add the extend OOB?

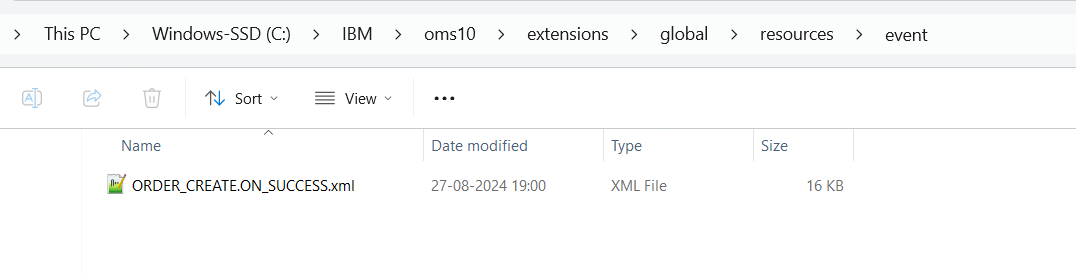
1. Create file : OMSExtensions.xml

Path: C:\IBM\oms10\extensions\global\entities

|  |
| --- |
| <DBSchema>  <Entities>  <Entity TableName="YFS\_ORDER\_HEADER">  <Attributes>  <Attribute ColumnName="EXTN\_IS\_MP" DataType="Flag" Description="Is Order belongs to MarketPlace." XMLGroup="Extn" Name="EXTN\_IS\_MP" Nullable="false" XMLName="IsMarketPlace" DefaultValue="N"/>  <Attribute ColumnName="EXTN\_BRAND" DataType="Text-100" Type="VARCHAR2" Description="Brand of the order" XMLGroup="Extn" Nullable="true" XMLName="Brand"/>  </Attributes>  </Entity>  </Entities>  </DBSchema> |



Path : C:\IBM\oms10\extensions\global\resources\event  
Refer OOB and extend EXTN tag at Order level



1. Build Resouces and entities jar

In cmd : cd C:\IBM\oms10\bin

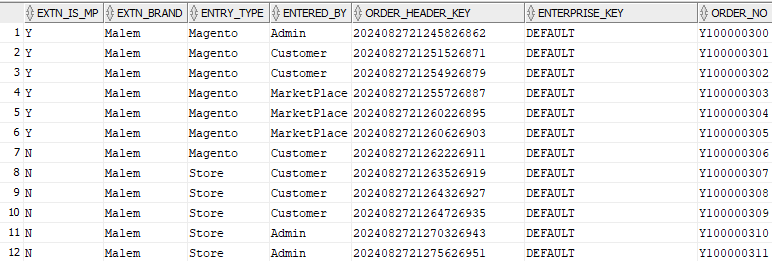
|  |
| --- |
| deployer.cmd -t resourcejar  deployer.cmd -t entitydeployer |

1. build & deploy smcfs

|  |
| --- |
| buildwar.cmd -Dappserver=websphere -Dwarfiles=smcfs -Ddevmode=true |

1. Testing by Create Order and test accordingly

|  |
| --- |
| <Order DocumentType="0001" EnterpriseCode="DEFAULT" OrderNo="" EntryType="Magento" EnteredBy="MarketPlace">  <Extn IsMarketPlace="Y" Brand="Malem" />  <PersonInfoShipTo Country="US" >  </PersonInfoShipTo>  <PersonInfoBillTo Country="US">  </PersonInfoBillTo>  <OrderLines>  <OrderLine DeliveryMethod="SHP" OrderedQty="1" PrimeLineNo="1" ShipNode="" SubLineNo="1">  <Item ItemID="ITEM1" UnitOfMeasure="EACH" ProductClass="NEW"/>  </OrderLine>  </OrderLines>  </Order> |



## How to add the Search Fields Customization by Configuration (Json config)?

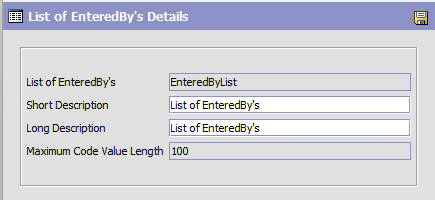
Adding enteredBy /entryType with dropdown.

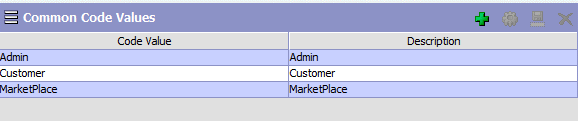
1. Path: C:\IBM\oms10\repository\callcenter\cc-customcode\call-center-order\packages\order-search\src-custom\assets\custom\search\_fields.json

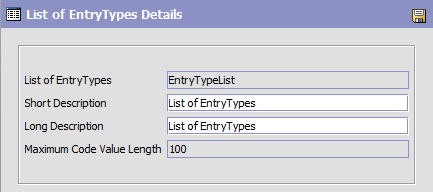
|  |
| --- |
| {      "orders": {          "fields": [              {              "id": "itemID",              "label": "ItemID",              "type": "dropdownQuery",              "request": "OrderLine.Item.ItemID",              "target": "orders",              "showWhen": "always",              "insertAfter": "",              "operator":"LIKE",              "fetch":{                  "api":"icc.order.order-search.getQueryTypeList",                  "type":"mashup",  "parameters":{},                       "response": {                           "listAttribute": "QueryTypeList.StringQueryTypes.QueryType",                           "map": {                               "id": "QueryType",                              "label": "QueryTypeDesc"                           }                        }                      }              },              {              "id": "orderLineKey",              "label": "Order Line Key",              "type": "text",              "request": "OrderLine.OrderLineKey",              "target": "orders",              "showWhen": "always",              "insertAfter": ""              },              {              "id": "reqShipDate",              "label": "ReqShipDate",              "type": "date",              "request": "ReqShipDate",              "target": "orders",              "showWhen": "always",              "insertAfter": ""              },              {              "id": "carrierServiceCode",              "label": "Carrier Service Code",              "type": "dropdown",              "request": "OrderLine.CarrierServiceCode",              "target": "orders",              "showWhen": "always",              "insertAfter": "",              "fetch": {                  "api": "icc.dataprovider.getCommonCodeDescForValue",                  "type": "mashup",                  "parameters": {                      "CommonCode.CallingOrganizationCode": "DEFAULT",                      "CommonCode.CodeType": "EXTN\_CARRIER\_SERVICE"                  },                  "response": {                      "listAttribute": "CommonCodeList.CommonCode",                          "map": {                              "id": "CodeValue",                              "label": "CodeShortDescription"                          }                      }                      }                  },                  {                      "id": "entryTypeCC",                      "label": "EntryType",                      "type": "dropdown",                      "request": "EntryType",                      "target": "orders",                      "showWhen": "always",                      "insertAfter": "",                      "fetch": {                          "api": "icc.dataprovider.getCommonCodeDescForValue",                          "type": "mashup",                          "parameters": {                              "CommonCode.CallingOrganizationCode": "DEFAULT",                              "CommonCode.CodeType": "EntryTypeList"                          },                          "response": {                              "listAttribute": "CommonCodeList.CommonCode",                              "map": {                                  "id": "CodeValue",                                  "label": "CodeShortDescription"                              }                          }                      }                  },                  {                      "id": "enteredBy",                      "label": "EnteredBy",                      "type": "dropdown",                      "request": "EnteredBy",                      "target": "orders",                      "showWhen": "always",                      "insertAfter": "",                      "fetch": {                          "api": "icc.dataprovider.getCommonCodeDescForValue",                          "type": "mashup",                          "parameters": {                              "CommonCode.CallingOrganizationCode": "DEFAULT",                              "CommonCode.CodeType": "EnteredByList"                          },                          "response": {                              "listAttribute": "CommonCodeList.CommonCode",                              "map": {                                  "id": "CodeValue",                                  "label": "CodeShortDescription"                              }                          }                      }                  },                  {                      "id": "extnIsMarketPlace",                      "label": "IsMarketPlace",                      "type": "text",                      "request": "Extn.IsMarketPlace",                      "target": "orders",                      "showWhen": "always",                      "insertAfter": ""                      }                ]          }      } |

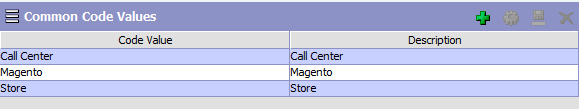
1. Create Custom CommonCodeType and add entries in commoncode according in application manager

Application Platform[DEFAULT]>>Presentation>>Custom Common Code Types









## How to add the custom columns and actions in the existing order-search-result page’s table

### Add Custom Columns – adding Extn\_brand

1. JSON object name: The main JSON body contains uniquely defined objects that map to different tables. To find the object names to use in the JSON file, you can check the log messages in the browser console while you navigate through the user interface. Search for the initializeTable message and you can see the object name. For example, the following snippet shows the object name order-table when you are on the outbound order search results page.

|  |
| --- |
| Time since init (ms): 1722956892187.  Log type: LOG.  Log generator: common-components.  Message: BaseTableComponent.initializeTable(): Initializing configuration for order-table |

2. Go to this path: \devtoolkit\_docker\callcenter-code\call-center-order  
Then, set the runAsCustomization as ‘true’ for order-search-result routes in overrides.json file.

3. Create a new buc-table-config.json file under this path “call-center-order\packages\order-search-result\src-custom\assets\custom\”

|  |
| --- |
| {      "order-table": {          "name": "order-table",          "headers": [              {                  "id": "orderHeaderKey",                  "name": "EXTN.ORDER\_SEARCH.OH\_KEY",                  "sortable": false,                  "dataBinding": "OrderHeaderKey",                  "sortKey": "OrderHeaderKey",                  "sequence": 1              } ,              {                  "name": "EXTN.ORDER\_SEARCH.LINE\_SUB\_TOTAL",                  "id": "lineTotal",                  "formatter": {                      "type": "currency",                      "value": "OverallTotals.LineSubTotal",                      "currencyCode": "PriceInfo.Currency"                  }              },              {                  "name": "EXTN.ORDER\_SEARCH.BRAND",                  "id": "extnBrand",                  "dataBinding": "Extn.Brand",                  "sequence": 2              },              {                  "name": "EXTN.ORDER\_SEARCH.ISMARKETPLACE",                  "id": "extnIsMarketPlace",                  "dataBinding": "Extn.IsMarketPlace"                }          ]      }  } |

Where,  
**headers** : It is the list of headers in the table.

**id** : The unique identifier of the column.

**name** : The text to display in the column header. If text is translated, you can add the translation files into the i18n folder and reference the label. For example,

|  |
| --- |
| “name” : “EXTN.ORDER\_TABLE.ORDER\_HEADER\_KEY” |

**sortable** : Boolean to indicate whether this column is sortable. The default value is true.

**sortKey** : The key to sort on when the column is sortable.

**databinding** : The attribute to retrieve from the API response and display in the column. The attribute must exist in the mashup API. Update the file if the attribute does not exist. For more information, see [Extending mashups in the web UI framework](#_2.2)_Incrementing_mashups).

**sequence** : (Optional) The absolute position of the custom column in the table; starting at 1. If the specified position contains a fixed column, then this column is moved to the next nonfixed position in the table. If not defined, the custom column is added to the end of the table.

**formatter** : Format the data in the column. Each formatter is an object with different properties. The type property identifies the type of formatter. Supported type values are **currency**, **dateTime**, and **quantity**. To use the formatter attribute, you must omit the **dataBinding** attribute and instead, specify the binding in the formatter.value attribute.

|  |
| --- |
| **type**: **currency**   1. **value**: The attribute to retrieve from the API response and display in the column. 2. **currencyCode**: (Optional) The currency code to use when formatting the value. The currency code is also fetched from a property in the API response. |
| **type : dateTime**   * **value:** The attribute to retrieve from the API response and display in the column. * **dateFormat:** (Optional) The data format to use. Refer to the Moment JS library for formatting options. Default is L LT. For more information on date format, go to [Moment.js](https://momentjs.com/) |

4. Change the assets path in angular.json file (path: call-center-order\angular.json

|  |
| --- |
| "assets": [                      {                          "glob": "\*.json",                            "input": "packages/order-search-result/src-merged/assets/order-search-result",                            "output": "assets/order-search-result"                        },                        {                            "glob": "\*.json",                            "input": "packages/order-search-result/src-merged/assets/call-center-order",                            "output": "assets/call-center-order"                            },                            {                            "glob": "\*\*",                            "input": "packages/order-search-result/src-merged/assets",                              "output": "assets"                            },                           {                             "glob": "\*\*",                               "input": "node\_modules/@buc/svc-angular/assets",                               "output": "assets"                           },                           {                             "glob": "\*\*",                             "input": "node\_modules/@buc/common-components/assets",                             "output": "assets"                           }                   ] |

Also replace the contents in the projects > [route-name] > architect > build > configurations > merged-prod > assets array.  
**Note :** The yellow highlighted is the route name and the green highlighted is the module name.

5. Copy the env folder from this path “call-center-order\packages\order-search-result\src\environments”  
to your src-custom folder path call-center-order\packages\order-search-result\src-custom\environments.  
Then, add the below syntax at the end of the file.

|  |
| --- |
| environment.customization=true; |

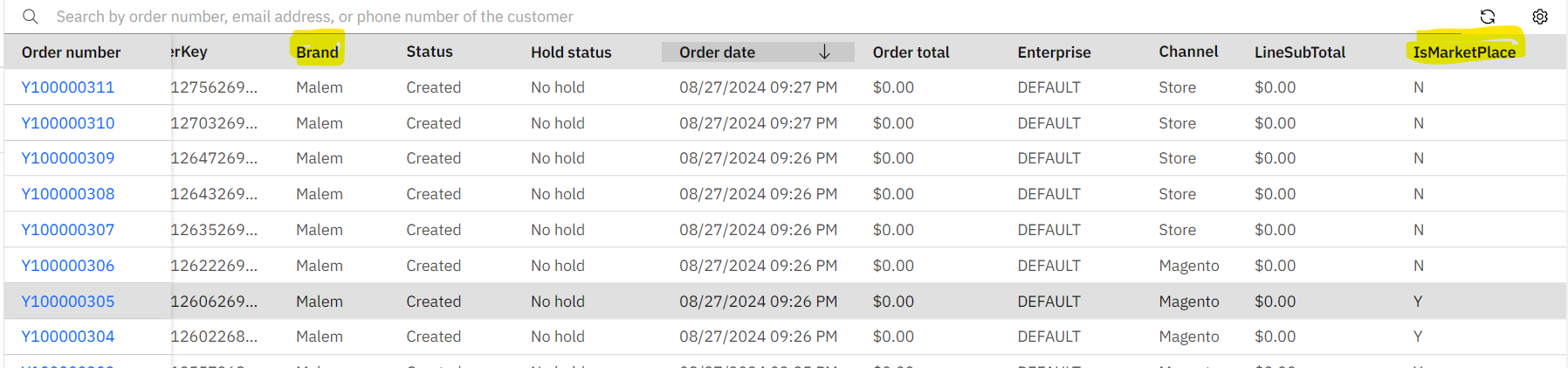
6. Create a new en.json file under custom folder :

Path: call-center-order\packages\order-search-result\src-custom\assets\custom\i18n\en.json

|  |
| --- |
| {      "EXTN": {          "ORDER\_SEARCH": {              "REQ\_SHIP\_DATE": "Request Ship Date",              "REQ\_DELIVERY\_DATE": "Request Delivery Date",              "OH\_KEY": "OrderHeaderKey",              "LINE\_SUB\_TOTAL": "LineSubTotal",              "BRAND": "Brand",              "ISMARKETPLACE": "IsMarketPlace"          }      }  } |

7. Restart Yarn.

|  |
| --- |
| #to stop yarn yarn stop-app #to start yarn yarn start-app |

8. Results  


## How to add/remove the custom columns in the existing Order-search-result page’s table?

### Remove OOB Columns

1. JSON object name: The main JSON body contains uniquely defined objects that map to different tables. To find the object names to use in the JSON file, you can check the log messages in the browser console while you navigate through the user interface. Search for the initializeTable message and you can see the object name. For example, the following snippet shows the object name order-table when you are on the outbound order search results page.

|  |
| --- |
| Time since init (ms): 9334.  Log type: LOG.  Log generator: common-components.  Message: BaseTableComponent.initializeTable(): Initializing configuration for order-table |

2. Go to this path: C:\IBM\oms10\repository\callcenter\cc-customcode\call-center-order\packages\order-search-result\src\assets\order-search-result\buc-table-config.json

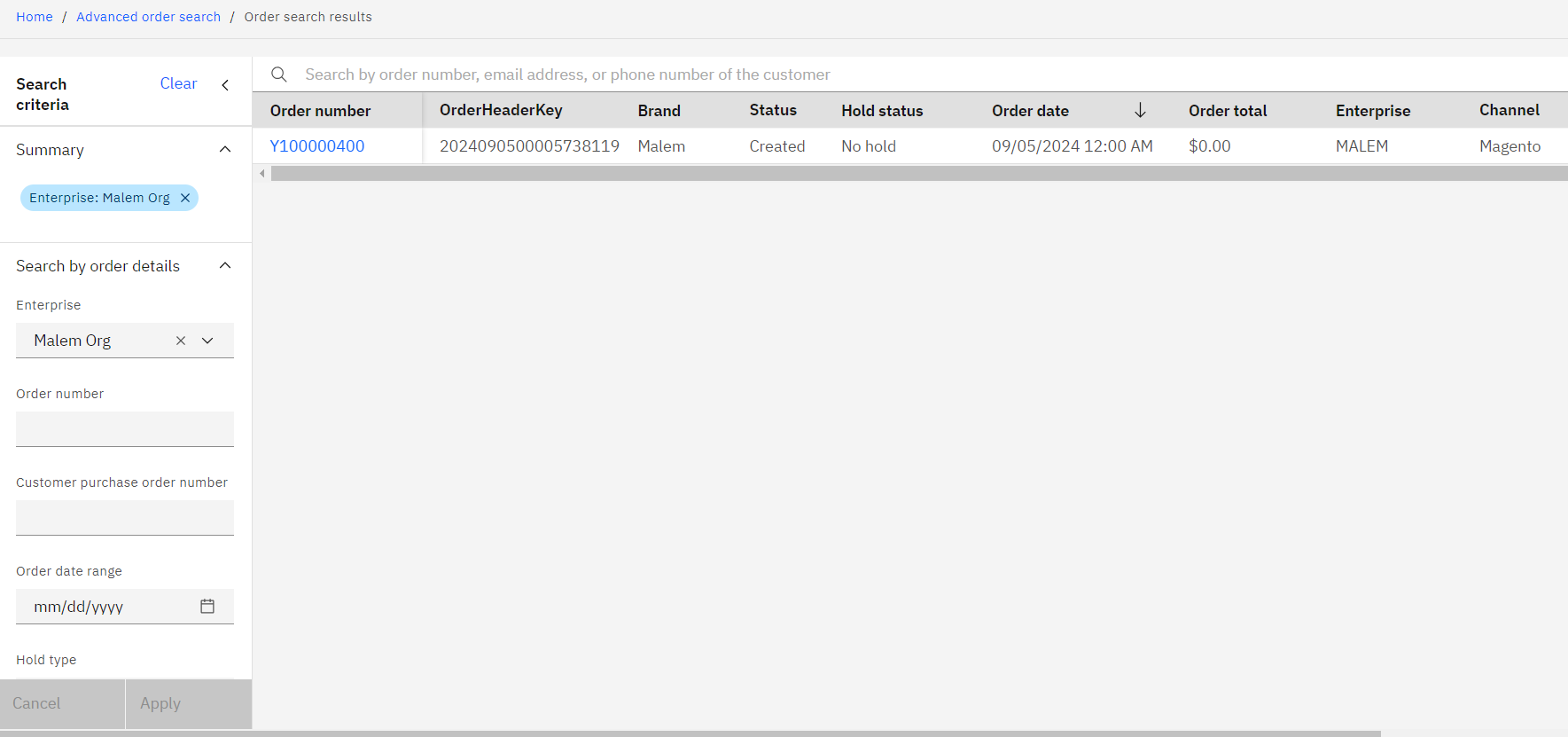
[Note: Path will be found with the help of step 1]

3. Remove the fields at your desire

|  |
| --- |
| {  "name": "ORDER\_SEARCH.GENERAL.LABEL\_CUSTOMER\_NAME",  "id": "customerName",  "sortKey": "CustomerFirstName",  "sortable": true  },  {  "name": "ORDER\_SEARCH.GENERAL.LABEL\_EMAIL\_ADDRESS",  "id": "customerEMailID",  "sortable": true,  "sortKey": "CustomerEMailID",  "dataBinding": "CustomerEMailID"  },  {  "name": "ORDER\_SEARCH.GENERAL.LABEL\_CUSTOMER\_TELEPHONE",  "id": "CustomerPhoneNo",  "sortable": true,  "sortKey": "CustomerPhoneNo",  "dataBinding": "CustomerPhoneNo"  },  {  "name": "ORDER\_SEARCH.GENERAL.LABEL\_POSTAL\_CODE",  "id": "customerZipCode",  "sortable": true,  "sortKey": "CustomerZipCode",  "dataBinding": "CustomerZipCode"  },  {  "name": "ORDER\_SEARCH.GENERAL.LABEL\_ADDRESS",  "id": "address",  "sortable": false  }, |

4. Restart Yarn.

|  |
| --- |
| #to stop yarn yarn stop-app #to start yarn yarn start-app |

5. Results  
  
No customer related info

### B. add Columns

1. Path : C:\IBM\oms10\repository\callcenter\cc-customcode\call-center-order\packages\order-search-result\src-custom\assets\custom\buc-table-config.json

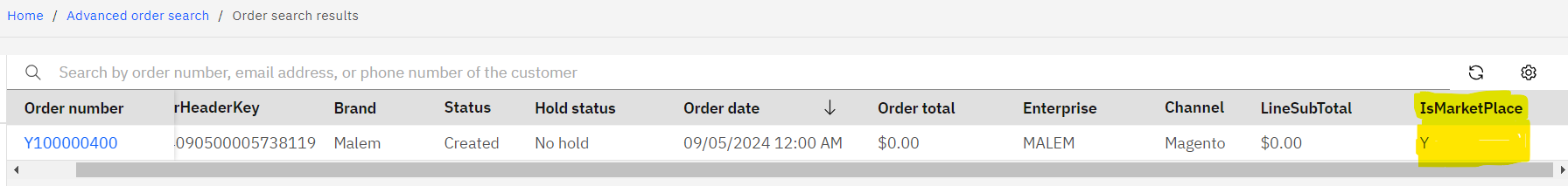
2. Add the fields to the table determined.

|  |
| --- |
| {      "order-table": {          "name": "order-table",          "headers": [              {                  "id": "orderHeaderKey",                  "name": "EXTN.ORDER\_SEARCH.OH\_KEY",                  "sortable": false,                  "dataBinding": "OrderHeaderKey",                  "sortKey": "OrderHeaderKey",                  "sequence": 1              } ,              {                  "name": "EXTN.ORDER\_SEARCH.LINE\_SUB\_TOTAL",                  "id": "lineTotal",                  "formatter": {                      "type": "currency",                      "value": "OverallTotals.LineSubTotal",                      "currencyCode": "PriceInfo.Currency"                  }              },              {                  "name": "EXTN.ORDER\_SEARCH.BRAND",                  "id": "extnBrand",                  "dataBinding": "Extn.Brand",                  "sequence": 2              },              {                  "name": "EXTN.ORDER\_SEARCH.ISMARKETPLACE",                  "id": "extnIsMarketPlace",                  "dataBinding": "Extn.IsMarketPlace"              }          ]      }  } |

3. Restart Yarn.

|  |
| --- |
| #to stop yarn yarn stop-app #to start yarn yarn start-app |

4. Results:



Note: make sure necessary perquisite steps are done in order to see output

## Adding summary fields to existing detail pages

1. Understand the buc-field-details.json syntax that defines the fields that are displayed on the Summary page of the details pages.

{

"order-summary": {

"name": "order-summary",

"attributes": [

{

"id": "seller",

"name": "Seller",

"value": "",

"attrTID": "OrderStatusComponent-seller",

"type": "label",

"dataBinding": "SellerOrganizationName"

},

{

"id": "currency",

"name": "Currency",

"value": "",

"attrTID": "orderDetails - Currency",

"type": "dropdown",

"dataBinding": "PriceInfo.Currency",

"options": {

"editPermission": "",

"editAttr": "PriceInfo.Currency", //If user changes the value in the dropdown, the PriceInfo.Currency attribute is updated

"fetch": { //Identify that the dropdown options need to be fetched

"api": "icc.common-components.getCurrencyList", //Fetch dropdown options from getCurrencyList

"type": "mashup", //Must be "mashup" to fetch from API

"parameters": {}, //Parameters to pass to getPaymentTypeList call. In this case, no parameters needed

"response": {

"listAttribute": "CurrencyList.Currency", //Retrieve the 'ListCurrency.Currency' from the response

"map": {

"id": "Currency", //Sets the id to the values of the Currency response

"label": "CurrencyDescription" //Sets the dropdown options to the values of the CurrencyDescription response

}

}

}

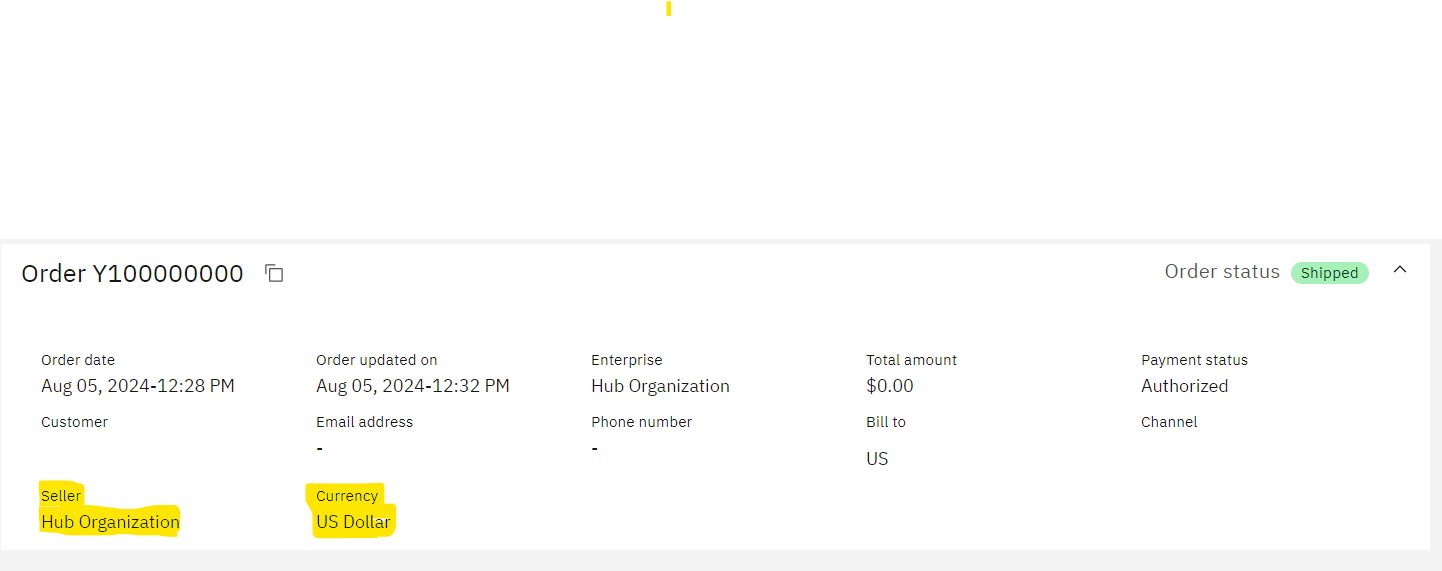
}

}

]

}

}

1. Results in :   
   Note : for more details refer [link](https://www.ibm.com/docs/en/call-center/10.0?topic=configuration-adding-summary-fields-existing-detail-pages).

## Adding custom actions to existing pages [Customer Details]

**1. Update the src-custom/environments folder and the angular.json file**

Angular.js

"assets": [

{

"glob": "\*\*",

"input": "packages/customer-search-result/src-merged/assets",

"output": "assets"

},

{

"glob": "\*.json",

"input": "packages/customer-search-result/src-merged/assets/call-center-customer",

"output": "assets/customer-search-result"

},

{

"glob": "\*\*",

"input": "node\_modules/@buc/svc-angular/assets",

"output": "assets"

},

{

"glob": "\*\*",

"input": "node\_modules/@buc/common-components/assets",

"output": "assets"

}

],

Override.json

"customer-details": {

      "runAsCustomization": true

    }

2. to update the customer details page, update the packages/customer-details/src-custom/assets/custom/buc-page-definitions.json file with the following details:

displays an alert window:

{

"customer-details": {

"name": "customer-details",

"actions": [

{

"id": "custom-action",

"label": "custom.LABEL\_CUSTOM\_ACTION",

"resourceId": "BUCORD0014AT0001"

}

]

}

}

3. Add translation strings for the custom action.

1. Create or edit the en.json file.

For example, for the customer details page, packages/customer-details/src-custom/assets/custom/i18n/en.json.

1. Write the following JSON content into the en.json file. This label is used for the new custom action that you add to the page

{

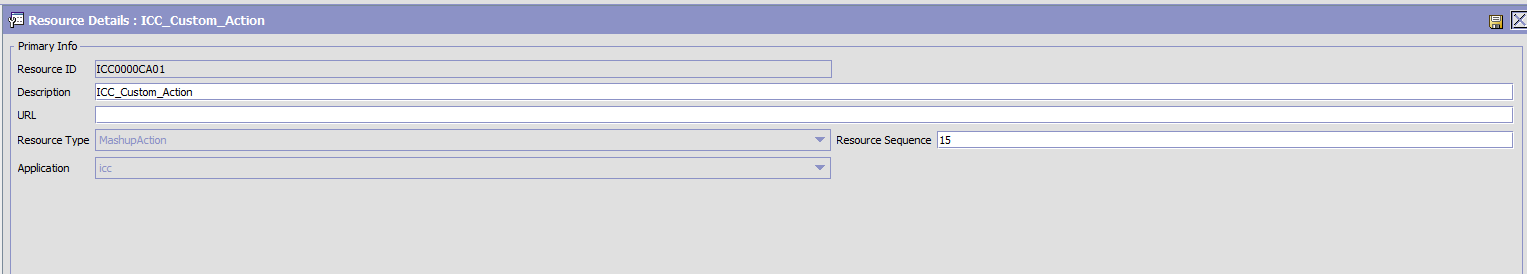
"custom": {

"LABEL\_CUSTOM\_ACTION": "Custom action"

}

}

Application Platform[Default]>>Presentation>>ResourceHierachy >>ICC\_Product\_Name



4. Create an action service file custom-action.service.ts and pass an action ID (which was specified as **id** for custom action in the buc-page-definitions.json file) to the select method of **ActionProcessorService**.

For example, for the customer details page, packages/customer-details/src-custom/app/service/custom-action.service.ts is updated in the following code, and **Custom action** displays an alert window:

import { CustomActionService } from "./service/custom-action.service";

import { ActionProcessorService } from '@buc/common-components';

import { InjectionToken } from "@angular/core";

import { BucActionsModule, ExtensionModule,

    BucCommonComponentsModule, BucIconsModule, BucFeatureComponentsModule } from "@buc/common-components";

export class AppCustomizationImpl {

    static readonly components = [];

    static readonly providers = [

        CustomActionService

    ];

    static readonly imports = [

        BucActionsModule.forRoot(

            [

                {

                    name: 'custom-action',

                    action: CustomActionService

                }

            ]

        ),

        BucCommonComponentsModule,

        BucFeatureComponentsModule,

        BucIconsModule

    ];

}

5. Register a custom action service in the app-customization.impl.ts file.

For example, for the customer details page, packages/customer-details/src-custom/app/app-customization.impl.ts.

1. Import a custom action service, which was created in the previous step.
2. Import BucActionsModule from "@buc/common-components".
3. Add custom action service to imports as shown in the following example:

import { CustomActionService } from "./service/custom-action.service";

import { ActionProcessorService } from '@buc/common-components';

import { InjectionToken } from "@angular/core";

import { BucActionsModule, ExtensionModule,

    BucCommonComponentsModule, BucIconsModule, BucFeatureComponentsModule } from "@buc/common-components";

export class AppCustomizationImpl {

    static readonly components = [];

    static readonly providers = [

        CustomActionService

    ];

    static readonly imports = [

        BucActionsModule.forRoot(

            [

                {

                    name: 'custom-action',

                    action: CustomActionService

                }

            ]

        ),

        BucCommonComponentsModule,

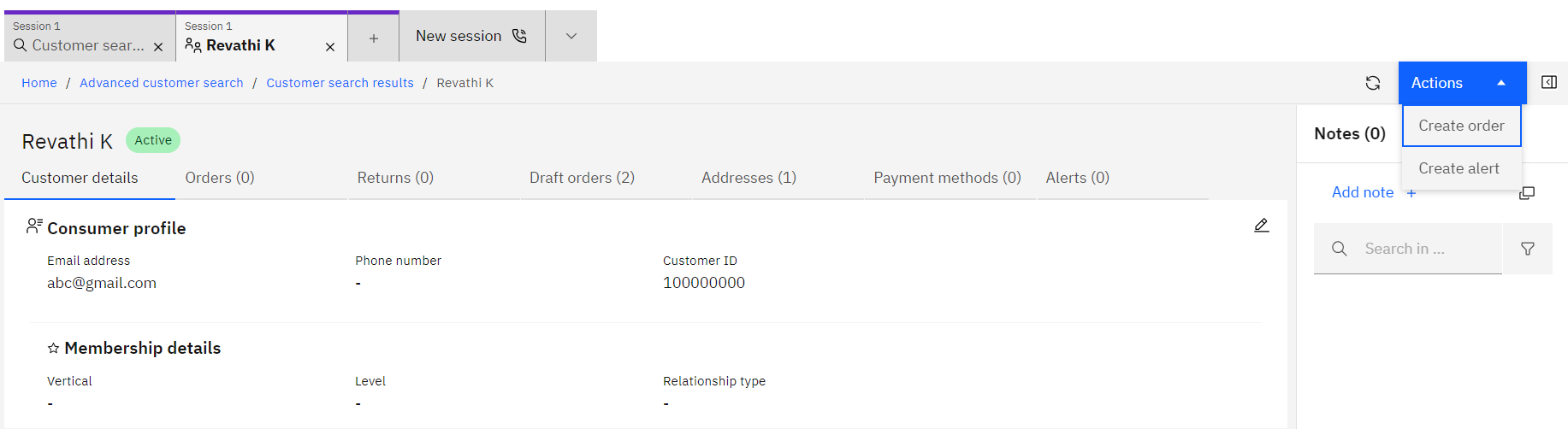
        BucFeatureComponentsModule,

        BucIconsModule

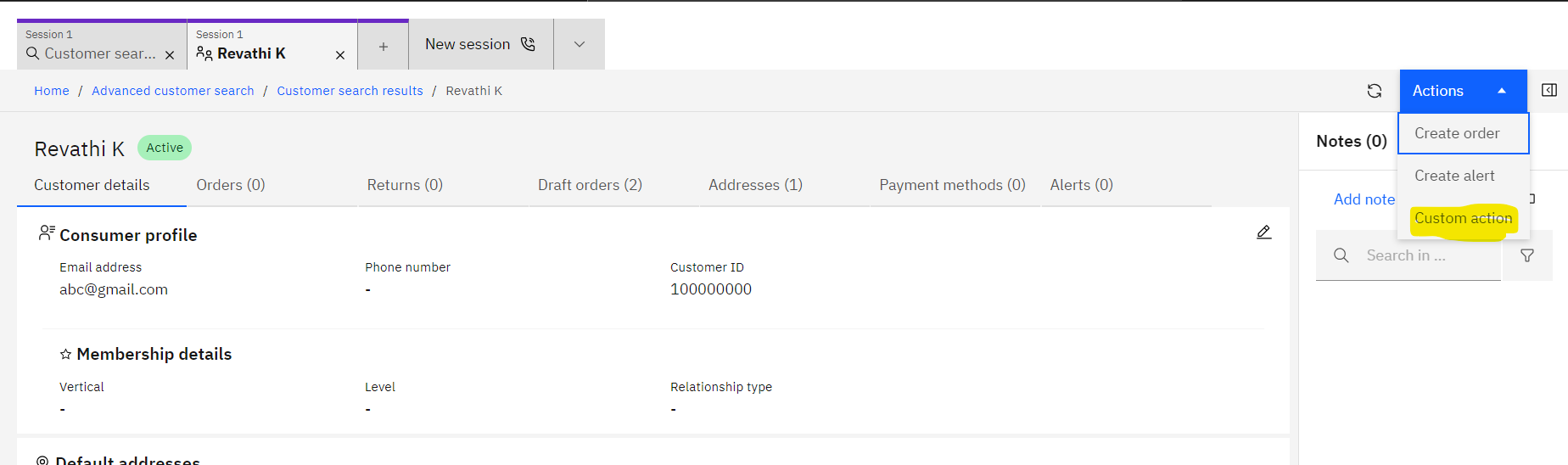
    ];

}

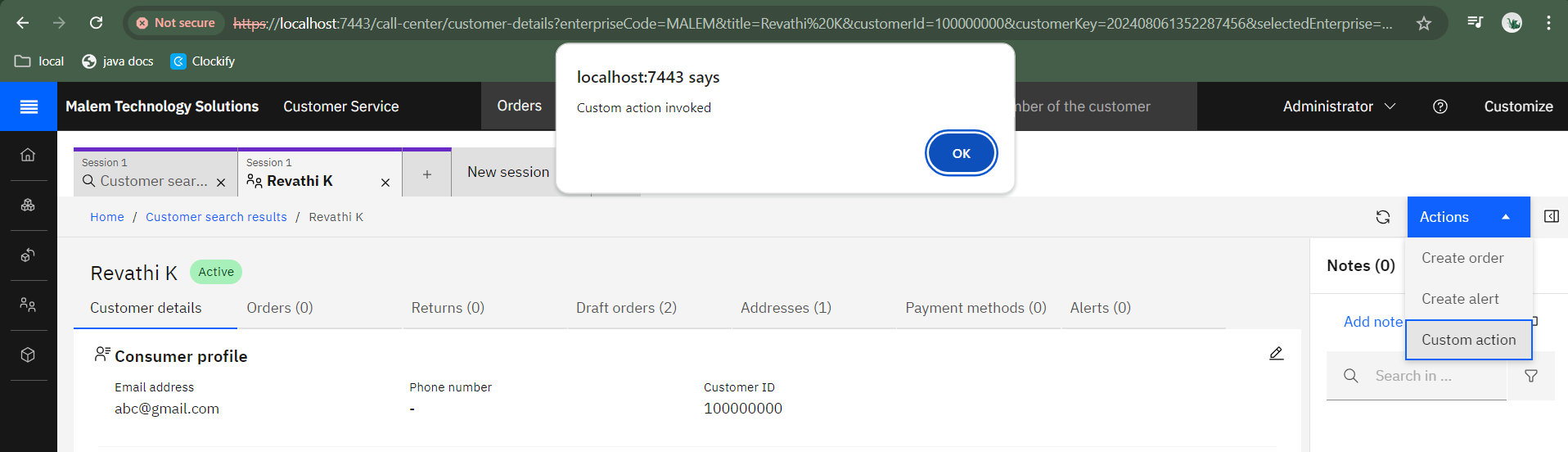
Before:

****

After:

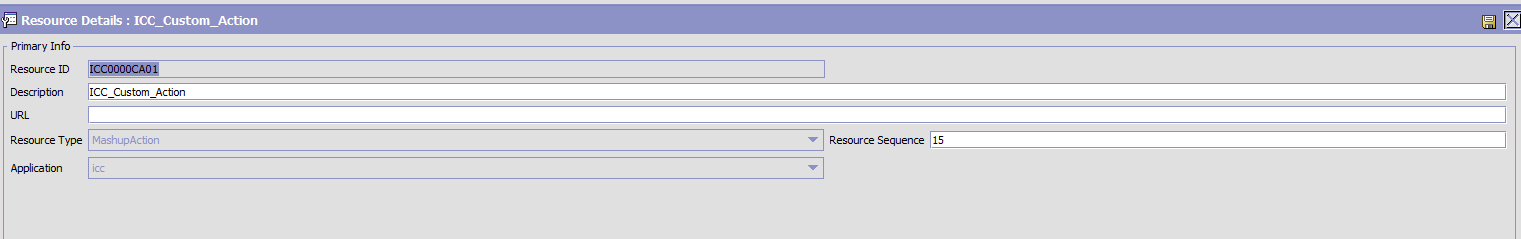


On click:



## Adding custom actions to existing pages [Order Details]

Application Platform[Default]>>Presentation>>ResourceHierachy >>ICC\_Product\_Name



C:\IBM\oms10\repository\callcenter\cc-customcode\call-center-order\packages\order-details\src-custom\assets\custom\buc-page-definitions.json

"order-details": {

        "name": "order-details",

        "actions": [{

            "id": "custom-action",

            "label": "custom.LABEL\_CUSTOM\_ACTION",

            "resourceId": "ICC0000CA01"

        }]

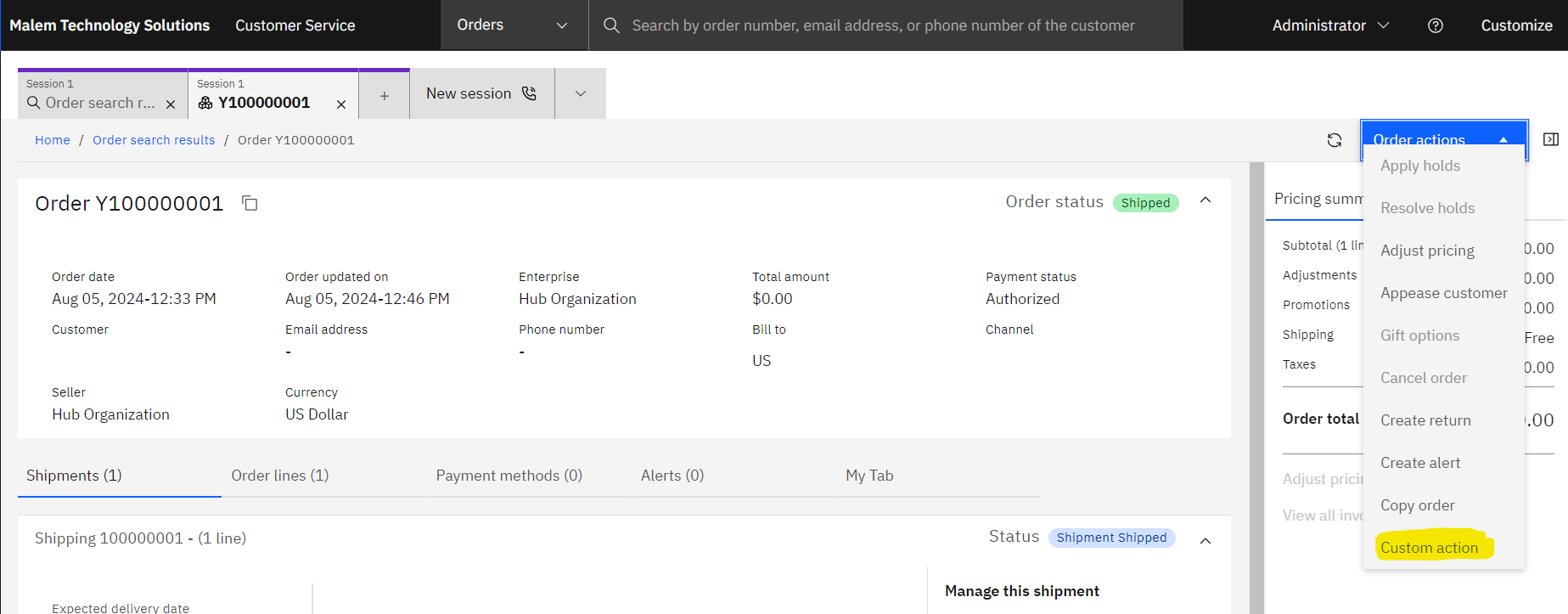
C:\IBM\oms10\repository\callcenter\cc-customcode\call-center-order\packages\order-details\src-custom\assets\custom\i18n\en.json

 "custom": {

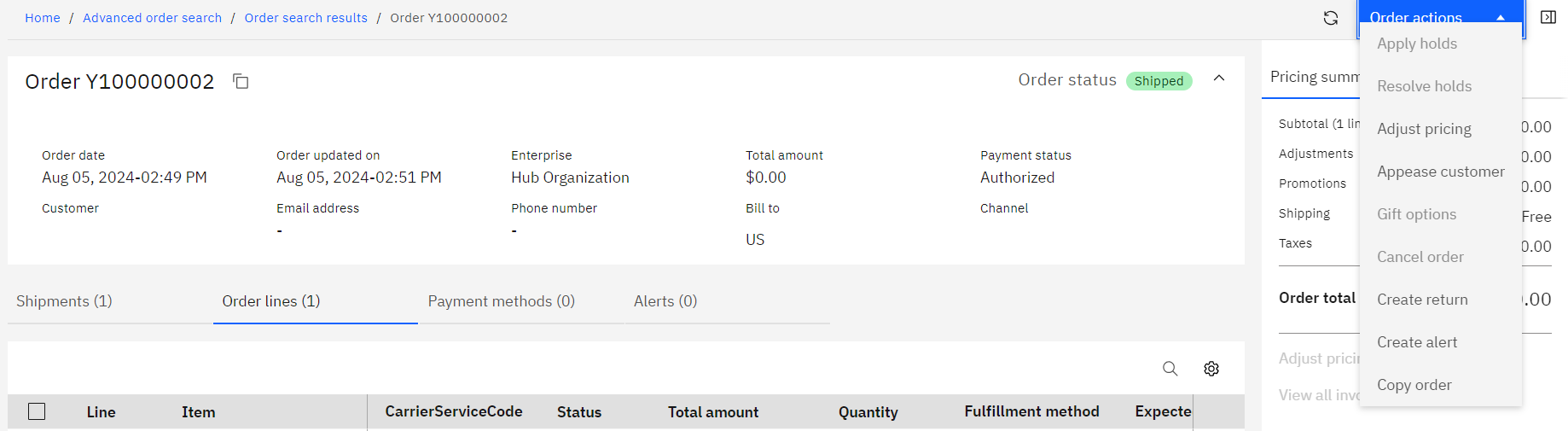
        "LABEL\_CUSTOM\_ACTION": "Custom action"

    }

**Results :**

****

b. To Restrict Order Action with “Custom Action”.



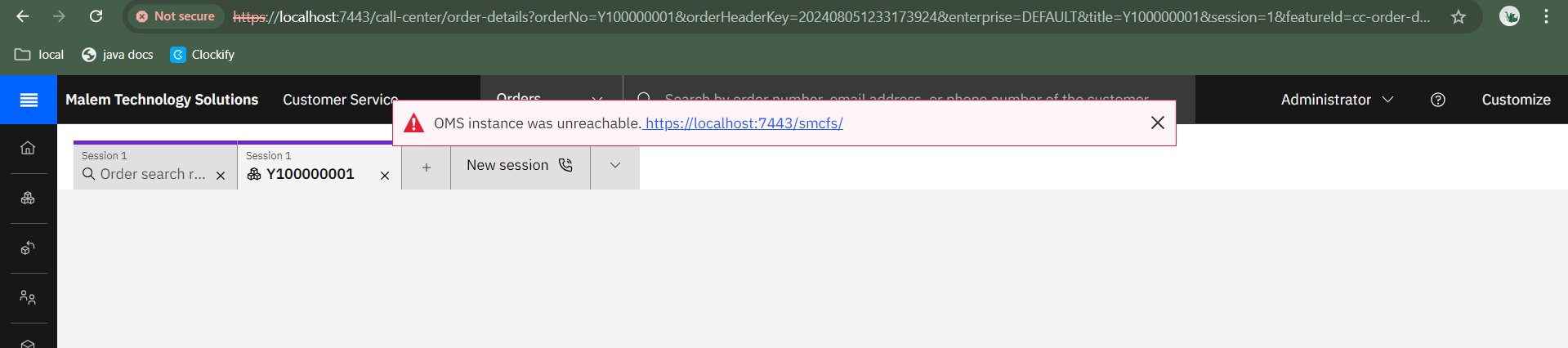
Path: C:\IBM\oms10\repository\callcenter\cc-customcode\call-center-order\packages\order-details\src-custom\app\features\order\order-routing.module.ts

***Before : [working]***

|  |
| --- |
| path: 'home',  component: OrderDetailsComponent,      data: {        id: 'ICC0000CA01'      } |

***after :***

|  |
| --- |
| {      canActivate: [BucCommonClassesConnectorResourcePermissionsGuard],      path: 'home',      component: OrderDetailsComponent,      data: {        id: 'ICC0000CA01'      }    }, |



## How to add the custom popup?

## overflowMenuActions

1. Add OverFlow Menu Header

Path: C:\IBM\oms10\repository\callcenter\cc-customcode\call-center-order\packages\order-details\src-custom\assets\custom\buc-table-config.json

|  |
| --- |
| "overflowMenuActions": [  {  "id": "AddNoteActionService",  "label": "EXTN.ORDER\_SUMMARY\_LINES.LABEL\_OVERFLOW\_ADD\_NOTE\_ACTION”  }  ] |

2. Path : C:\IBM\oms10\repository\callcenter\cc-customcode\call-center-order\packages\order-details\src-custom\app\features\order\order-lines\order-lines.component.ts

|  |
| --- |
| modificationReasonsList = [{ content: 'Gift Warp the Order' },{ content: 'Delivery on Weekend' }, { content: 'Delivery at Doorstep' }];  reasonComments='';  invQty: number;  chargeAmount:number;    public readonly ACTION\_MODIFY\_ADD\_NOTE\_SERVICE = 'AddNoteActionService';  @ViewChild('modifyAddNoteModal', { static: true }) modifyAddNoteModal: TemplateRef<any>;  protected onOverflowMenuActionSelected(id: string): any {  switch (id) {  case this.ACTION\_ADD\_OR\_VIEW\_NOTES:  this.openViewAllNotesModal(this.ofmData.line);  break;  /\*we send the Boolean value is true to indicate that, it is for overflow actions. For toolbar, we don’t pass the Boolean value i.e only only one parameters for data\*/  case this.ACTION\_MODIFY\_ADD\_NOTE\_SERVICE:  this.openAddNoteCustomPopup([this.ofmData.line], true);  break;  }  }    /\* this function will called from both toolbar and overflow actions \*/  openAddNoteCustomPopup(selectedOlk?, overflow?: boolean) {  console.log("openCustomPopup :: start", selectedOlk);  var selectedLines: any;  var olks: any;  var ohks: any;  /\*here we are checking that the data is pass from toolbar or overflow\*/  if (overflow) {  //data is passing from overflow  selectedLines = selectedOlk;  olks = selectedLines[0].OrderLineKey;  ohks = selectedLines[0].OrderHeaderKey;  } else {  selectedLines = this.loadedOrderLines.filter((item) => selectedOlk.includes(item.OrderLineKey));  console.log("selectedLines", selectedLines);  ohks = selectedLines.map(x => x.OrderHeaderKey);  olks = selectedOlk;  }  //forming input for Popup by using its OOB component’s template  let inputs = {  modalText: {  header: 'Add Note Custom Change',  //this is where we insert the template in the OOB PopUp modal, to customize the screen view.  template: this.templates.modifyAddNoteModal,  templateData: selectedLines  },  optionOne: {  text: 'Cancel',  },  optionTwo: {  text: 'Save',  callback: this.popupModifyCallBackFn.bind(this, selectedLines),  class: {  primary: true  },  }  };  //this CommonBinaryOptionModalComponent is the OOB poup template which we can customize.  this.modalService.create({  component: CommonBinaryOptionModalComponent,  inputs  })  }  //this method will trigger when the user selects the ModficationReason  selectModficationReason($event) {  if ($event.item.selected) {  this.modificationReasonCode = $event.item.content;  console.log('the modificationReasonCode is ', this.modificationReasonCode);    if(this.modificationReasonCode==='Gift Warp the Order'){  this.chargeAmount=12;  }else if(this.modificationReasonCode==='Delivery on Weekend'){  this.chargeAmount=1;  }else if(this.modificationReasonCode==='Delivery at Doorstep'){  this.chargeAmount=5;  }  }  }    /\*this method will trigger when the Save button is click, it will call the manageCommonCode api to insert the orderheaderkey and orderlinekey, so simply we are calling some service on click of custom actions in the orderline table\*/  async popupModifyCallBackFn(selectedLines) {  console.log("the data are selectedLines: ", selectedLines);  //the selected lines can have multiple rows of data, so we are using loop to call the api.  for (let i = 0; i < selectedLines.length; i++) {  //forming api input  const mashupInput = {  Order: {  Action: "Modify",  OrderHeaderKey: selectedLines[0].OrderHeaderKey,  ModificationReasonCode:this.modificationReasonCode,  ModificationReasonText:this.reasonComments,  HeaderCharges: {  HeaderCharge:  [  {  ChargeAmount: this.chargeAmount,  ChargeCategory:"Handling"  }  ]    }    }  };  try {  //we used this BucCommOmsMashupService to call the api in callcenter  this.bucCommOmsMashupService.callMashup('extn\_icc.order.order-lines.ExtnModifyOrder', mashupInput, {}).toPromise()  .then(mashupOutput => {  this.bucCommOmsMashupService.getMashupOutput(mashupOutput, 'extn\_icc.order.order-lines.ExtnModifyOrder');  console.log("mashupOutput:::", mashupOutput);  //this is for showing the notification banner when the api calls successfully  this.ccNotificationService.notify({  type: "success",  title: 'changeOrder api called successfully.'  });  });  } catch (error) {  }  }  } |

3. Path: C:\IBM\oms10\repository\callcenter\cc-customcode\call-center-order\packages\order-details\src-custom\app\features\order\order-lines\order-lines.component.html

|  |
| --- |
| <ng-template bucId="modifyAddNoteModal" let-data="data">  <div style="background-color: rgb(255, 255, 255);">  <div style="display: flex; flex-direction: row;">  <div>  <h3 style="margin-bottom: 20px;">Custom Pop Up</h3>  </div>  </div>  <div>  <h5>Add Note</h5>  <p>Additional Request, Handling Chargers are included : </p>  <div class="bx--row margin-top--2\_5rem">  <div class="bx--col-lg-8 margin-bottom--0\_5rem">  <buc-dropdown  label="Select Modification Reason" [items]="modificationReasonsList"  (selected)="selectModficationReason($event)">  </buc-dropdown>  </div>  <div class="bx--col-lg-8 margin-bottom--0\_5rem custom-textarea">  <buc-textarea class="width--100p" [(inputValue)]="reasonComments"  rows="4" maxlength="250" label="Comments (optional)" placeholder="">  </buc-textarea>  </div>  </div>  </div>  </div>  </ng-template> |

4. Create a new Mashup xml as shown:

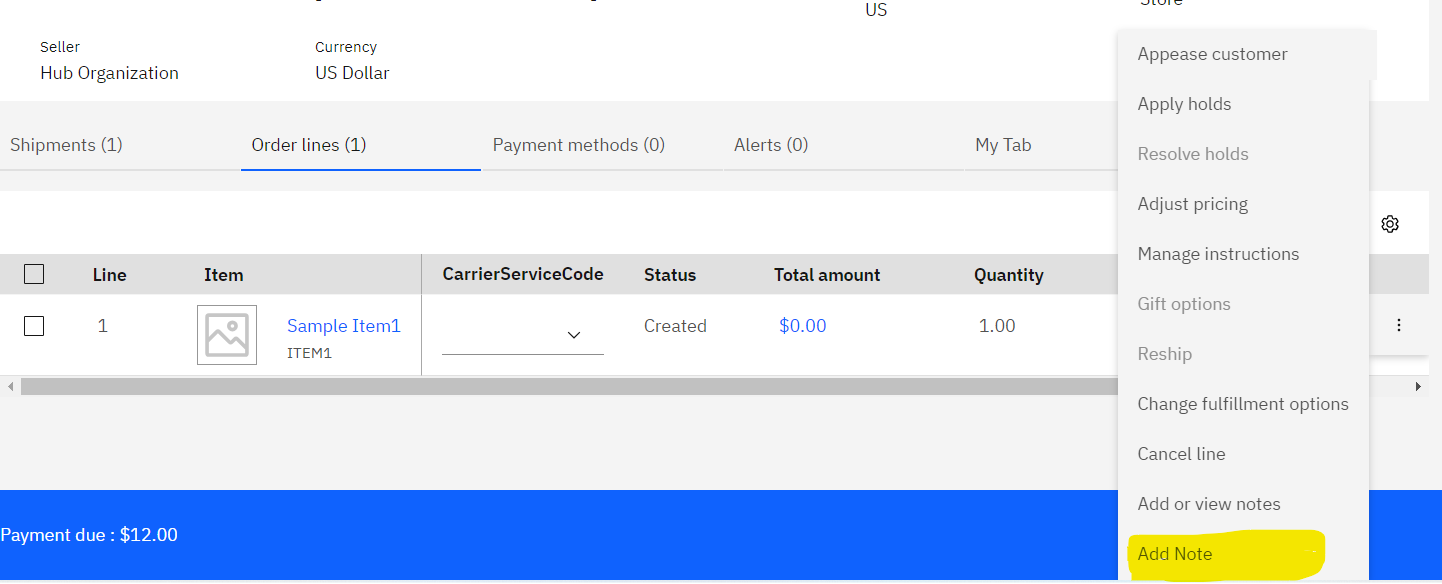
Path: C:\IBM\oms10\extensions\icc\webpages\mashupxmls\additional\_mashups.xml

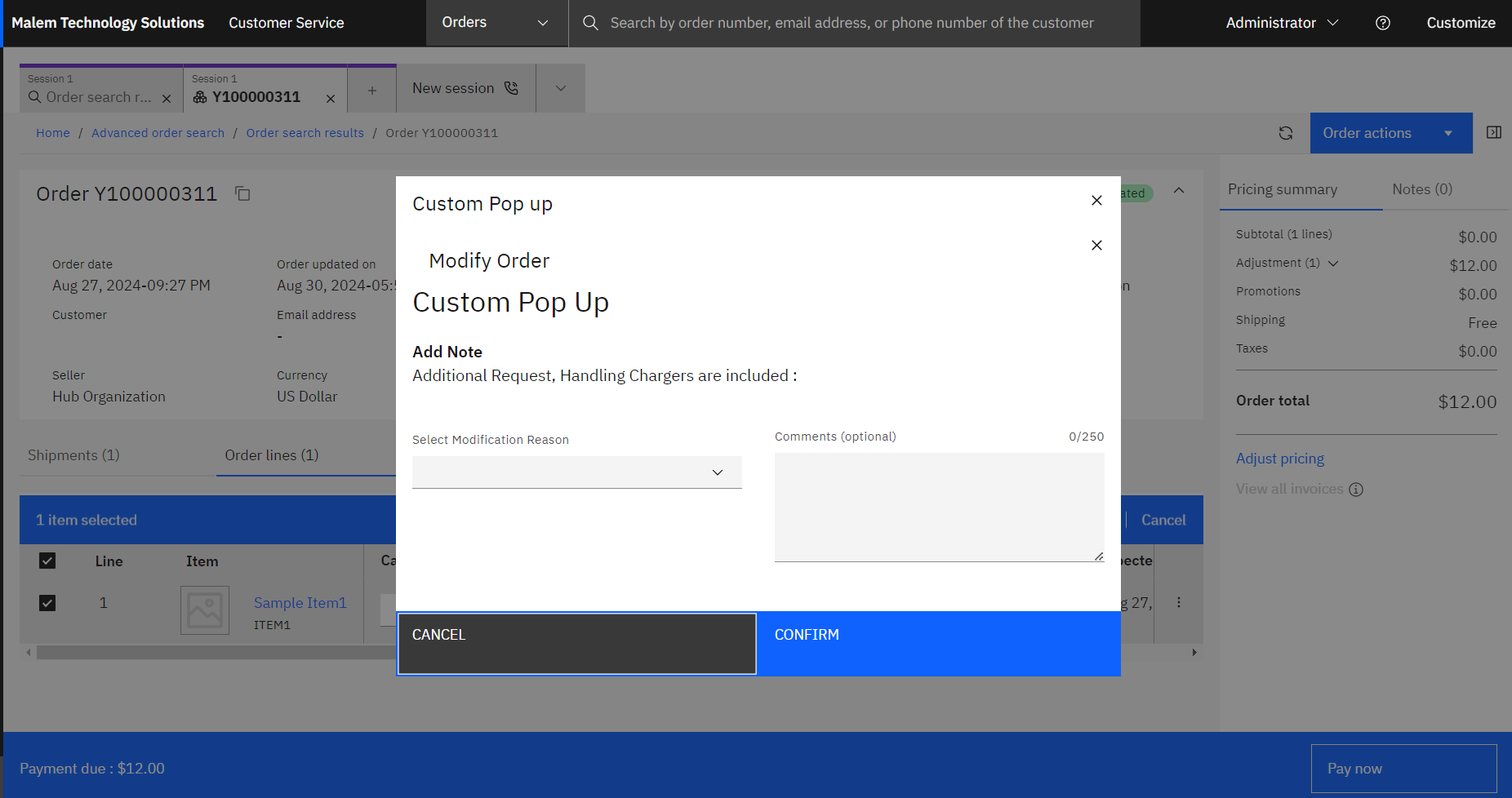
|  |
| --- |
| <mashup description="changeOrder to set the handling charge with reason" id="extn\_icc.order.order-lines.ExtnModifyOrder" mashuptype="XAPI" transactional="true" skipDataProvider="true">  <classInformation name="com.ibm.icc.common.mashups.ICCBaseMashup"/>  <API Name="changeOrder">  <Input>  <Order Action='Modify' OrderHeaderKey='' ModificationReasonCode='' ModificationReasonText=''>  <HeaderCharges>  <HeaderCharge ChargeAmount='' ChargeCategory='Handling' ChargeName='' Reference=''/>  </HeaderCharges>  </Order>  </Input>  </API>  </mashup> |

5. Restart Yarn.

|  |
| --- |
| #to stop yarn yarn stop-app #to start yarn yarn start-app |

6. Results:





## Taskbar button with action

1. Add TaskBar Menu Header

Path: C:\IBM\oms10\repository\callcenter\cc-customcode\call-center-order\packages\order-details\src-custom\assets\custom\buc-table-config.json

|  |
| --- |
| "orderline-table": {  "name": "orderline-table",  "headers": [{  "id": "carrierServiceCode",  "name": "CarrierServiceCode",  "sortable": false,  "isEditable": true,  "sequence": 1,  "fieldConfig": {  "type": "dropdown",  "invalidText": "invalidText",  "helperText": "helperText"  }  } ],  "toolbarActions": [  {  "id": "addNoteActionService",  "elem": "button",  "type": "primary",  "value": "EXTN.ORDER\_SUMMARY\_LINES.ADD\_NOTE\_ACTION",  "sequence": 1  }  ],  "overflowMenuActions": [  {  "id": "addNoteActionService",  "label": "EXTN.ORDER\_SUMMARY\_LINES.ADD\_NOTE\_ACTION"  }  ]  }, |

2. Path : C:\IBM\oms10\repository\callcenter\cc-customcode\call-center-order\packages\order-details\src-custom\app\features\order\order-lines\order-lines.component.ts

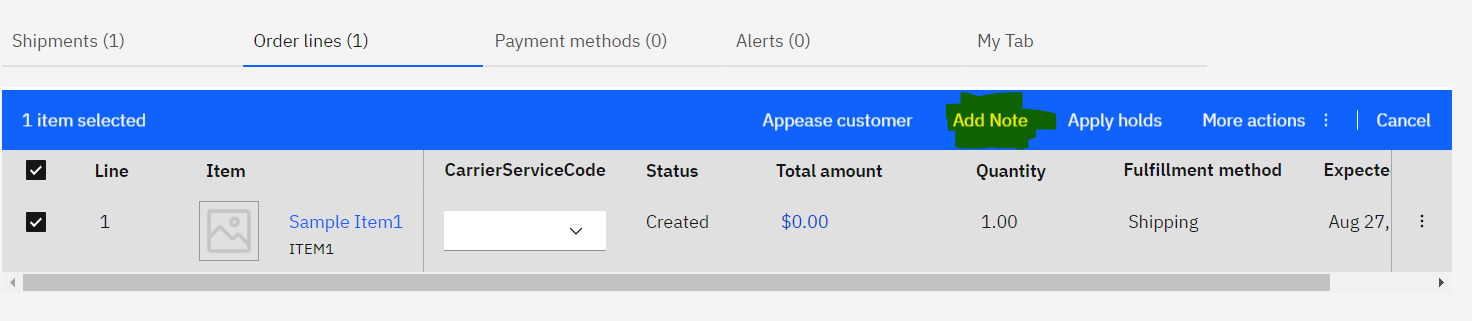
|  |
| --- |
| protected onToolbarActionClicked(id: string, event?: any): any {  case this.ACTION\_MODIFY\_ADD\_NOTE\_SERVICE:          this.openAddNoteCustomPopup(this.selected);          break;  } |

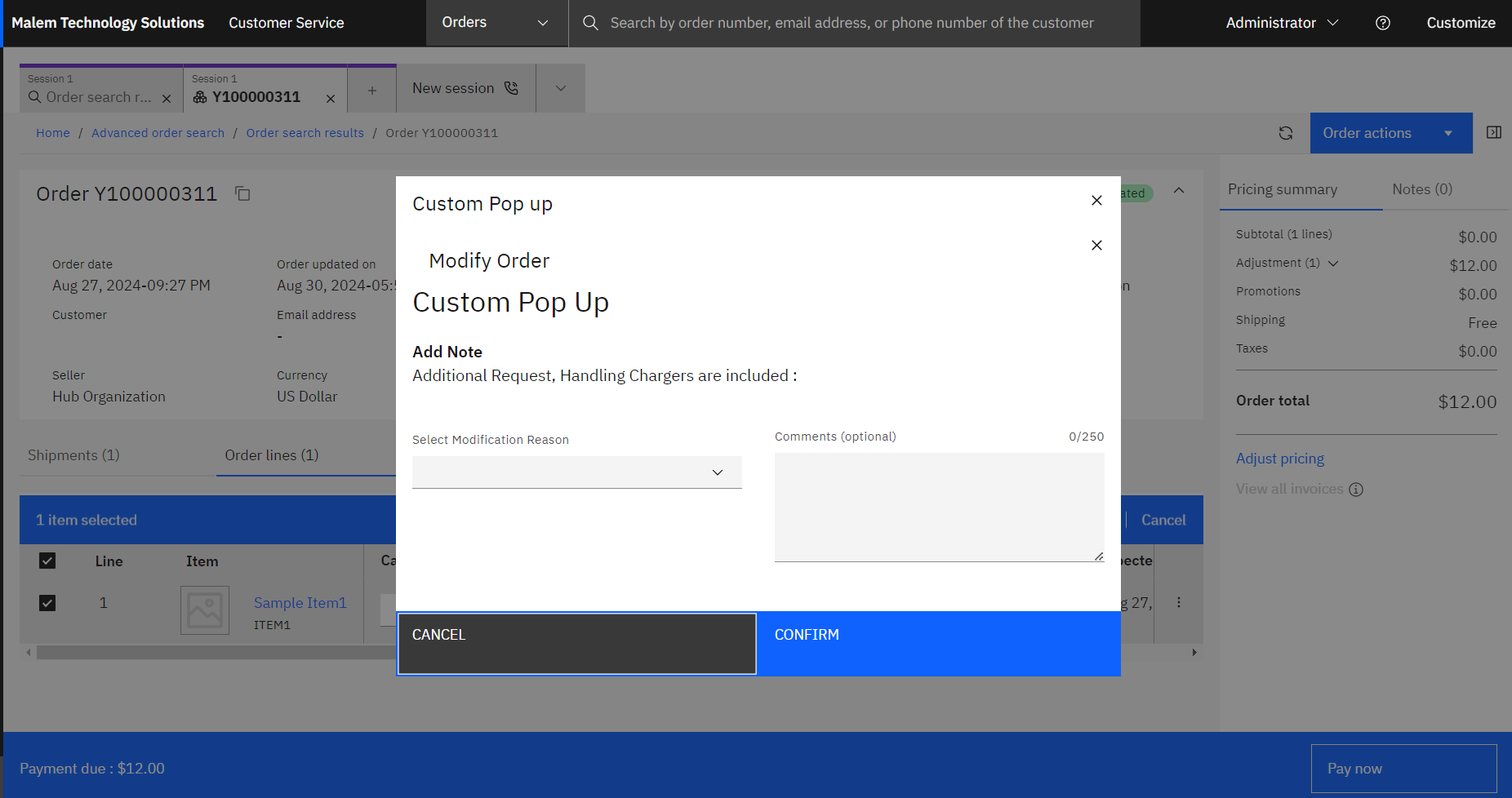
Note : including the necessary function from Topic 8.a

3. Restart Yarn.

|  |
| --- |
| #to stop yarn yarn stop-app #to start yarn yarn start-app |

4. Results:

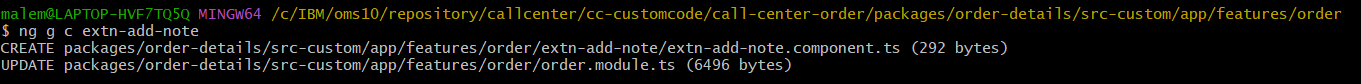


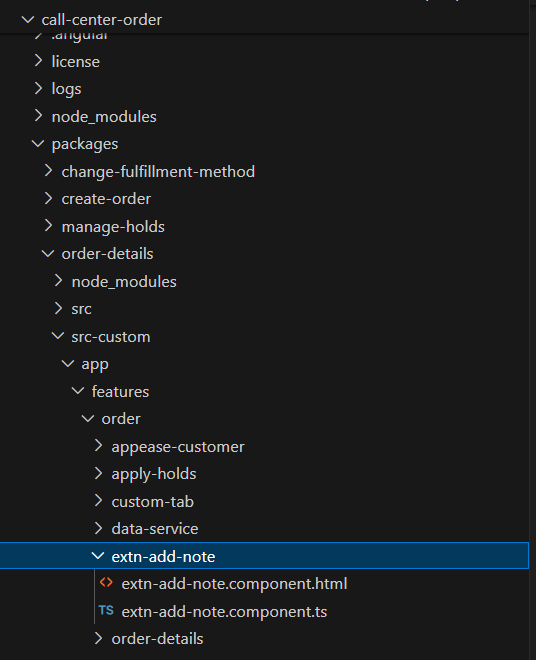


## Header button with action

1. In gitBash -> cd "C:\IBM\oms10\repository\callcenter\cc-customcode\call-center-order\packages\order-details\src-custom\app\features\order"

|  |
| --- |
| ng g c extn-add-note |





2. C:\IBM\oms10\repository\callcenter\cc-customcode\call-center-order\packages\order-details\src-custom\app\features\order\extn-add-note\extn-add-note.component.html

|  |
| --- |
| <ng-template #changeOrderPopUpTemplate1>    <p1>You click the popup button</p1>  </ng-template>  <ng-template #changeOrderPopUpTemplate>    <ibm-modal-header (closeSelect)="closeModal()"> Modify Order    </ibm-modal-header>    <div style="background-color: rgb(255, 255, 255);">      <div style="display: flex; flex-direction: row;">        <div>          <h3 style="margin-bottom: 20px;">Custom Pop Up</h3>        </div>      </div>      <div>        <h5>Add Note</h5>        <p>Additional Request, Handling Chargers are included : </p>        <div class="bx--row margin-top--2\_5rem">          <div class="bx--col-lg-8 margin-bottom--0\_5rem">            <buc-dropdown              label="Select Modification Reason" [items]="modificationReasonsList"              (selected)="selectModficationReason($event)">            </buc-dropdown>          </div>          <div class="bx--col-lg-8 margin-bottom--0\_5rem custom-textarea">            <buc-textarea class="width--100p" [(inputValue)]="reasonComments"              rows="4" maxlength="250" label="Comments (optional)" placeholder="">            </buc-textarea>          </div>        </div>      </div>    </div>  </ng-template> |

3. C:\IBM\oms10\repository\callcenter\cc-customcode\call-center-order\packages\order-details\src-custom\app\features\order\extn-add-note\extn-add-note.component.ts

|  |
| --- |
| import { Component, OnInit,Inject,Output,EventEmitter,TemplateRef,QueryList,ViewChild,ViewChildren } from '@angular/core';  import { BucCommOmsMashupService, CallCenterNavigationService } from '@buc/svc-angular';  import { ModalService } from 'carbon-components-angular';  import{CCNotificationService,BucNotificationService,CommonBinaryOptionModalComponent,    BucNotificationModel,TemplateIdDirective,BucTemplateDirective  } from '@buc/common-components';  import { TranslateService } from '@ngx-translate/core';  import { BaseModal } from 'carbon-components-angular';  import { ExtensionConstants } from '../../extension.constants';  @Component({    selector: 'call-center-extn-add-note',    templateUrl: './extn-add-note.component.html'  })  export class ExtnAddNoteComponent extends BaseModal implements OnInit {  private templates: { [id: string]: TemplateRef<any> } = {};    @ViewChildren(TemplateIdDirective) set \_templates(a: QueryList<TemplateIdDirective>) {      if (a) {        a.forEach(({ id, template }) => this.templates[id] = template);      }      }    @ViewChildren(BucTemplateDirective) templateRefs: QueryList<BucTemplateDirective>;    @ViewChild('changeOrderPopUpTemplate1', { static: true }) changeOrderPopUpTemplate1: TemplateRef<any>;    @ViewChild('changeOrderPopUpTemplate', { static: true }) changeOrderPopUpTemplate: TemplateRef<any>;    // componentId = 'extn-add-note';    componentName = 'ExtnAddNoteComponent';    selectedOrders;     public setModalClose:boolean;    modificationReasonsList = [{ content: 'Gift Warp the Order' },      { content: 'Delivery on Weekend' }, { content: 'Delivery at Doorstep' }];      modificationReasonCode:string;      chargeAmount:number;      reasonComments='';    constructor(      private bucCommOmsMashupService: BucCommOmsMashupService,      private ccNotificationService: CCNotificationService,      public translate: TranslateService,      private notificationService: CCNotificationService,      public bucNotificationService: BucNotificationService,public modalService: ModalService,      @Inject('modalData') public modalData,    ) {      super();      this.bucNotificationService = new BucNotificationService();     }    //custom start    @Output() closeCustomModal = new EventEmitter();    ngOnInit(init?, childPageAction?, childTab?): void {      console.log("the modal data input is", this.modalData);      this.selectedOrders = this.modalData.orderDetails;      this.setModalClose=false;      this.openPopUp(init, childPageAction, childTab);    }    openPopUp(init?, childPageAction?, childTab?) {      //creates button to provide option function can also be binded with function call      const optionOne = {        primary: '',        text: 'CANCEL',        callOnClose: true,        ckCb: true      };      const optionTwo = {        class: {          primary: true        },        text: 'CONFIRM',        callback: this.testMethod.bind(this,init, childPageAction, childTab),        ckCb: false      };      //ckcb should be false to enable the closeModal after button click for those option which has callBack property.      console.log("popup")      //below is the way to call our .html template      const modal = this.modalService.create({        component: CommonBinaryOptionModalComponent,        inputs:        {          modalText: {            header: 'Custom Pop up',            textData: 'You click the Modify Action button',            template: this.changeOrderPopUpTemplate          },          optionOne, optionTwo        }      });      console.log("popup modal create")    }    async testMethod(init?, childPageAction?, childTab?) {      const good: any = [];      //framing input to call mashup      const mashupInput = {        Order: {          Action: "Modify",          OrderHeaderKey: this.selectedOrders,          ModificationReasonCode:this.modificationReasonCode,          ModificationReasonText:this.reasonComments,          HeaderCharges: {            HeaderCharge:              [                {                  ChargeAmount: this.chargeAmount,                  ChargeCategory:"Handling"                }              ]          }        }      };      try {        //we used this BucCommOmsMashupService to call the api in callcenter        this.bucCommOmsMashupService.callMashup('extn\_icc.order.order-lines.ExtnModifyOrder', mashupInput, {}).toPromise()          .then(mashupOutput => {            this.bucCommOmsMashupService.getMashupOutput(mashupOutput, 'extn\_icc.order.order-lines.ExtnModifyOrder');            console.log("mashupOutput:::", mashupOutput);            //this is for showing the notification banner when the api calls successfully            this.ccNotificationService.notify({              type: "success",              title: 'changeOrder api called successfully.'            });            this.modalData.successCallback();            this.closeModal();            let msg="success msg";            this.notificationService.notify({              type: 'success',              title: msg            });          });          this.closeModal();        this.closeCustomModal.emit;        this.setModalClose=true;      } catch (err) {        console.log("error", err);      }  }    selectModficationReason($event) {      if ($event.item.selected) {        this.modificationReasonCode = $event.item.content;        console.log('the modificationReasonCode is ', this.modificationReasonCode);          if(this.modificationReasonCode==='Gift Warp the Order'){          this.chargeAmount=12;        }else if(this.modificationReasonCode==='Delivery on Weekend'){          this.chargeAmount=1;        }else if(this.modificationReasonCode==='Delivery at Doorstep'){          this.chargeAmount=5;        }      }    }    async popupModifyCallBackFn(selectedLines) {      console.log("the data are OHK: ", selectedLines);      //forming api input        const mashupInput = {          Order: {            Action: "Modify",            OrderHeaderKey: selectedLines,            ModificationReasonCode:this.modificationReasonCode,            ModificationReasonText:this.reasonComments,            HeaderCharges: {              HeaderCharge:                [                  {                    ChargeAmount: this.chargeAmount,                    ChargeCategory:"Handling"                  }                ]              }            }        };        try {  //we used this BucCommOmsMashupService to call the api in callcenter          this.bucCommOmsMashupService.callMashup('extn\_icc.order.order-lines.ExtnModifyOrder', mashupInput, {}).toPromise()            .then(mashupOutput => {              this.bucCommOmsMashupService.getMashupOutput(mashupOutput, 'extn\_icc.order.order-lines.ExtnModifyOrder');              console.log("mashupOutput:::", mashupOutput);  //this is for showing the notification banner when the api calls successfully              this.ccNotificationService.notify({                type: "success",                title: 'changeOrder api called successfully.'              });              });        } catch (error) {          }    }  } |

4. C:\IBM\oms10\repository\callcenter\cc-customcode\call-center-order\packages\order-details\src-custom\app\service\custom-action.service.ts

|  |
| --- |
| import { Injectable,OnDestroy,TemplateRef,ViewChild,ViewChildren,QueryList} from '@angular/core';  import {ActionProcessorService,CommonBinaryOptionModalComponent,TemplateIdDirective,BucTemplateDirective } from '@buc/common-components';  import { Constants } from '@call-center/order-shared';  import { ModalService } from 'carbon-components-angular';  import { Subscription } from 'rxjs';  import { ModalActionParams } from '../features/order/shared/order-details-paramtypes';  import { ExtnAddNoteComponent } from "../features/order/extn-add-note/extn-add-note.component";  import { Router } from '@angular/router';  @Injectable()  export class customActionService implements OnDestroy {      subscriptions: Subscription[] = [];      constructor(          private actionProcessorService: ActionProcessorService,          private modalService: ModalService,          protected router: Router ) {          const sub = this.actionProcessorService.select<ModalActionParams>('custom-action').subscribe(({ params }) => {            this.openCustomModificationActionModal(params);          });          this.subscriptions.push(sub);        }        private openCustomModificationActionModal(params): any {          console.log("start action :: ", params.data)          console.log("selectedLines :: ", params.data.pageObject.OrderHeaderKey)    //calls the component which we created earlier          this.modalService.destroy();          this.modalService.create({            component: ExtnAddNoteComponent,            inputs: {              modalData: {                orderDetails: params.data.pageObject.OrderHeaderKey,                successCallback: () => {                  this.successCallBack(params);                },                size: 'lg'              }            }          });        }        successCallBack(params: any): any {          this.actionProcessorService.dispatchUpdate(params.component, {            action: 'custom-action',            refresh: true,            success: true          });  //To refresh the parent screen          window.location.reload();        }        ngOnDestroy(): any {          this.subscriptions.forEach(s => s.unsubscribe());        }  } |

Note: Can create a new custom Action, here We are using existing one which we created in Topic 8

5. Create a new Mashup xml as shown:

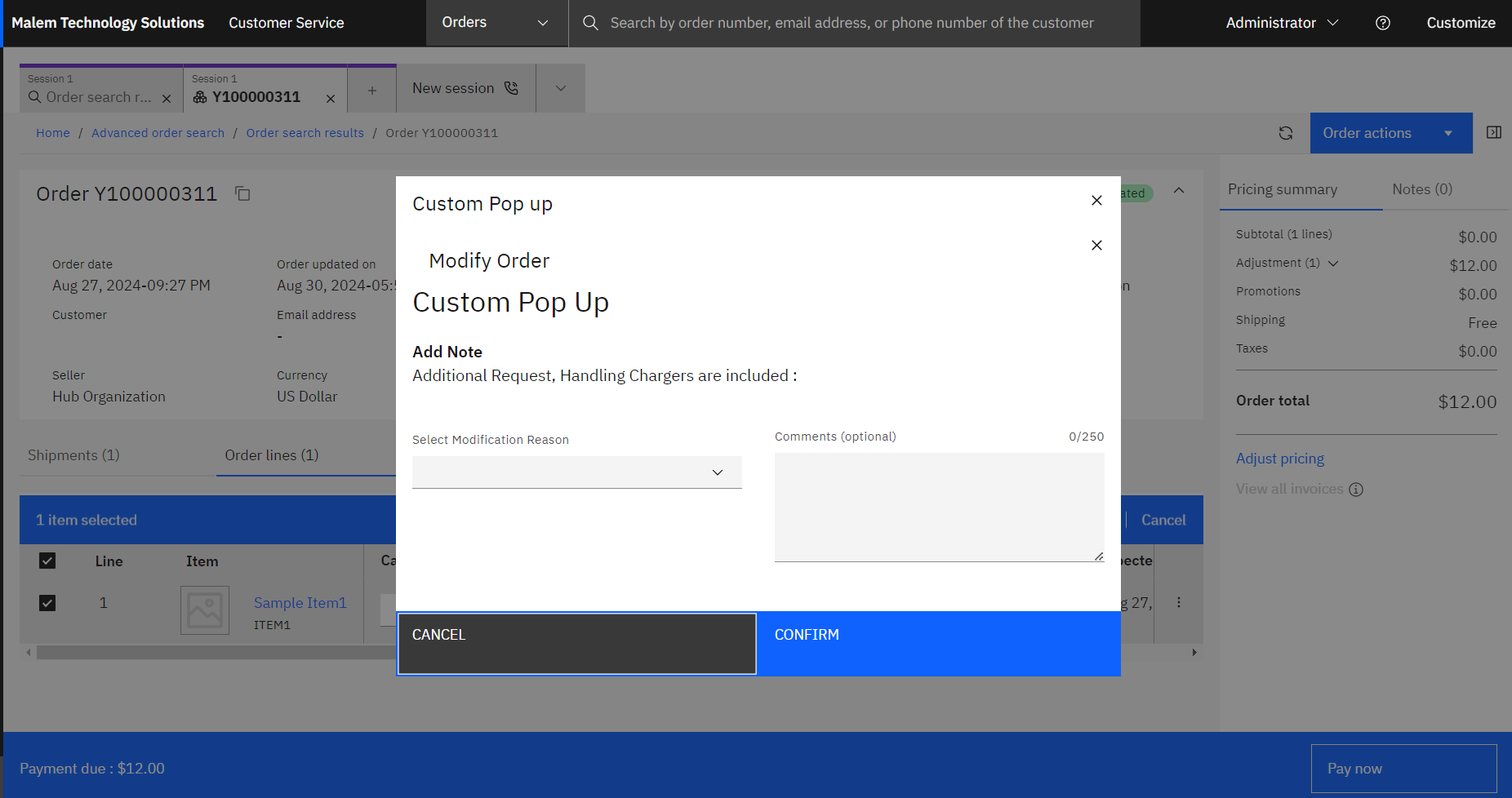
Path: C:\IBM\oms10\extensions\icc\webpages\mashupxmls\additional\_mashups.xml

|  |
| --- |
| <mashup description="changeOrder to set the handling charge with reason" id="extn\_icc.order.order-lines.ExtnModifyOrder" mashuptype="XAPI" transactional="true" skipDataProvider="true">  <classInformation name="com.ibm.icc.common.mashups.ICCBaseMashup"/>  <API Name="changeOrder">  <Input>  <Order Action='Modify' OrderHeaderKey='' ModificationReasonCode='' ModificationReasonText=''>  <HeaderCharges>  <HeaderCharge ChargeAmount='' ChargeCategory='Handling' ChargeName='' Reference=''/>  </HeaderCharges>  </Order>  </Input>  </API>  </mashup> |

6. Restart Yarn.

|  |
| --- |
| #to stop yarn yarn stop-app #to start yarn yarn start-app |

7. Results:

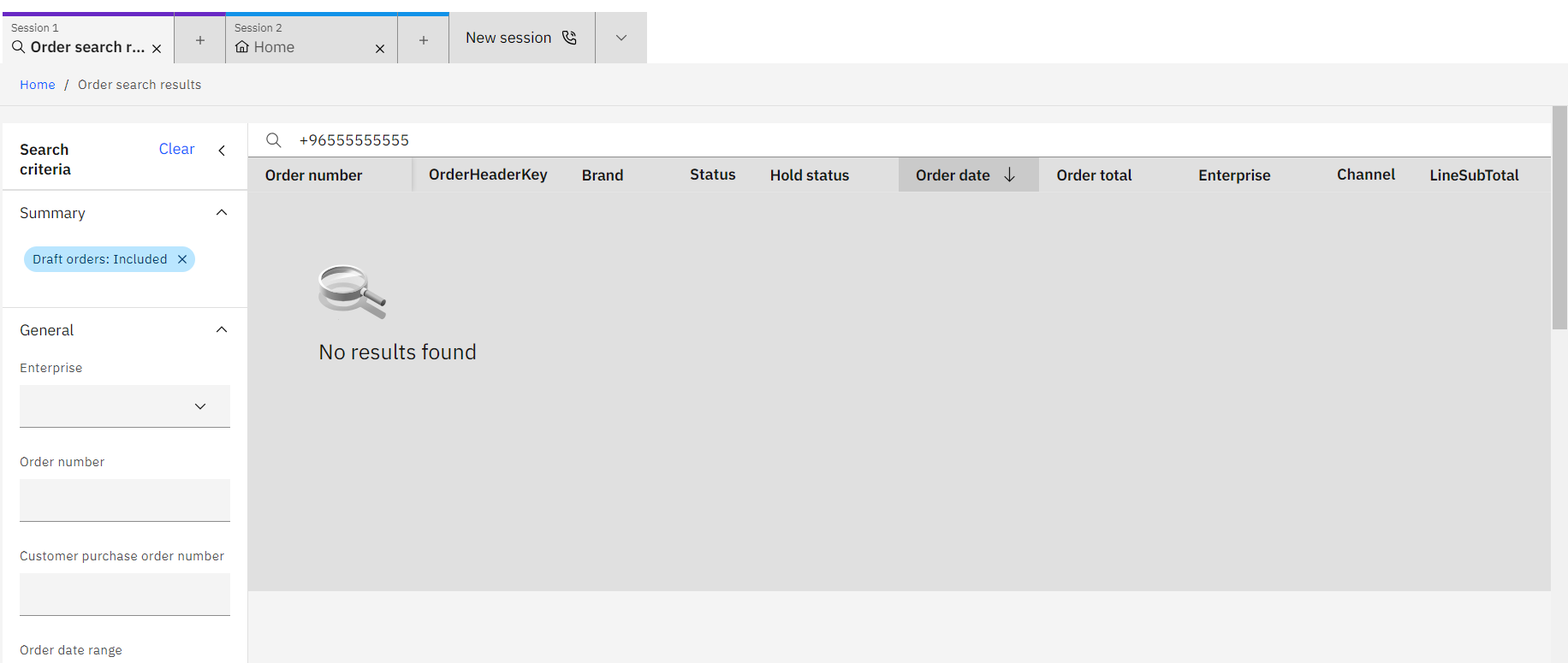


Output:   
Modification Reasons will have options to select, based on options Header Charger would be added accordingly.

## How to customize Homepage?

## Search order by phone number

Before :



## Hide Label

## Add Label

## adjust pricing [overflow action] add UOM column at line level

1. JSON object name: The main JSON body contains uniquely defined objects that map to different tables. To find the object names to use in the JSON file, you can check the log messages in the browser console while you navigate through the user interface. Search for the initializeTable message and you can see the object name. For example, the following snippet shows the object name order-table when you are on the outbound order search results page.

|  |
| --- |
| Time since init (ms): 1725031064540.  Log type: LOG.Log generator: common-components.  Message: BaseTableComponent.initializeTable():  Initializing configuration for adjust-pricing-orderline-table |

Note : facing issue -> If Override.json : For Order-Details is turned on then Adjust Pricing button doesn’t work

C:\IBM\oms10\repository\callcenter\cc-customcode\call-center-order\packages\order-details\src-custom\app\app-customization.impl.ts

|  |
| --- |
| import { customActionService } from "./service/custom-action.service";  import { BucActionsModule,      BucCommonComponentsModule, BucIconsModule, BucFeatureComponentsModule } from "@buc/common-components";  import { Constants as OrderConstants} from "@call-center/order-shared";  import { AdjustPricingActionService} from "@call-center/order-shared";  export class AppCustomizationImpl {      static readonly components = [];      static readonly providers = [          customActionService      ];      static readonly imports = [  BucActionsModule.forRoot(              [                  {                      name: 'custom-action',                      action: customActionService                  },                  {                      name: OrderConstants.ADJUST\_PRICING,                      action: AdjustPricingActionService                  }              ]          ),          BucCommonComponentsModule,          BucFeatureComponentsModule,          BucIconsModule      ];  } |

2. Add New column

Path: C:\IBM\oms10\repository\callcenter\cc-customcode\call-center-order\packages\order-details\src-custom\assets\custom\buc-table-config.json

|  |
| --- |
| "adjust-pricing-orderline-table": {      "name": "adjust-pricing-orderline-table",      "headers": [        {          "id": "uom",          "name": "EXTN.ORDER\_SUMMARY\_LINES.LABEL\_UOM",          "sortable": false,          "sortKey": "ItemShortDesc",          "style": {            "min-width": "6rem"          }        }      ],      "maxToolbarDisplayActions": 3,      "toolbarActions": [],      "toolbarContent": [],      "overflowMenuActions": []    } |

## Custom dropdown using mashup call?

## a. Carrier Service Code

1. Path: C:\IBM\oms10\repository\callcenter\cc-customcode\call-center-order\packages\order-details\src-custom\app\features\order\order-lines\order-lines.component.ts

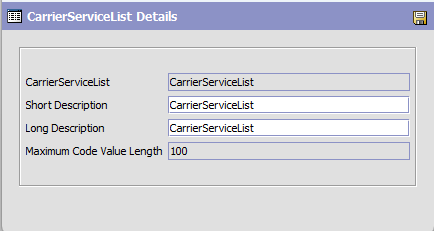
|  |
| --- |
| //Comment out the existing variable  //public carrierSvcCodeList = [{ content: 'UBER' }, { content: 'FedEx' }, { content: 'UPS' }];    //including empty list  public carrierSvcCodeList = [];  //call required function in ngOnInit()  ngOnInit(): void {      this.initialize();      if (this.actionSub) {        this.actionSub.unsubscribe();      }      this.actionSub = this.actionProcessorService.selectUpdate<any>(this.componentId).subscribe(res => {        if (res.params.refresh) {          this.parentPage.initialize(null,false);          this.initialize();        }      });      this.getcarrierSvcCodeList();    }  //function description passing data for carrier service code  async getcarrierSvcCodeList(){      var data: any;      data = "CarrierServiceList";      console.log("getcarrierSvcCodeList data: ",data);      await this.callgetCommonCodeList(data);      console.log("the getCommonCodeListOutput api output is ", this.getCommonCodeListOutput);      this.carrierSvcCodeList = this.getCommonCodeListOutput.CommonCodeList.CommonCode.map((item)=>        ({content:item.CodeShortDescription,}));    }  //function description for calling commoncode api with mashup integration  async callgetCommonCodeList(data: any) {      const getCommonCodeList = {        CommonCode: {          CodeType: data,          OrganizationCode: "DEFAULT"        }      };      console.log("input mashup :: ", getCommonCodeList)      try {        await this.bucCommOmsMashupService.callMashup('extn\_icc.order.order-lines.ExtngetCommonCodeList', getCommonCodeList, {}).toPromise()          .then(mashupOutput => {            this.bucCommOmsMashupService.getMashupOutput(mashupOutput, 'extn\_icc.order.order-lines.ExtngetCommonCodeList');            console.log("mashupOutput:::", mashupOutput);            this.getCommonCodeListOutput = mashupOutput.mashupResponse.controllerData.MashupRefs.MashupRef[0].Output;          });      } catch (error) {      }    } |

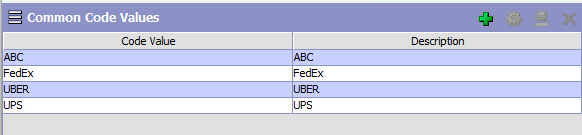
2. Create New Mashup xml as shown below :

Path : C:\IBM\oms10\extensions\icc\webpages\mashupxmls\additional\_mashups.xml

|  |
| --- |
| <mashup description="Get Common Code API" id="extn\_icc.order.order-lines.ExtngetCommonCodeList" mashuptype="XAPI" transactional="true" skipDataProvider="true">  <classInformation name="com.ibm.icc.common.mashups.ICCBaseMashup"/>  <API Name="getCommonCodeList">  <Input>  <CommonCode OrganizationCode="" CodeType="">  </CommonCode>  </Input>  <Template>  <CommonCodeList>  <CommonCode CodeShortDescription=""/>  </CommonCodeList>  </Template>  </API>  <AlternateResourceIds>  <AlternateResourceId altResourceId="ICCSYS00001"/>  </AlternateResourceIds>  </mashup> |

3. Create CommonCode List and CommonCode accordingly as shown below:

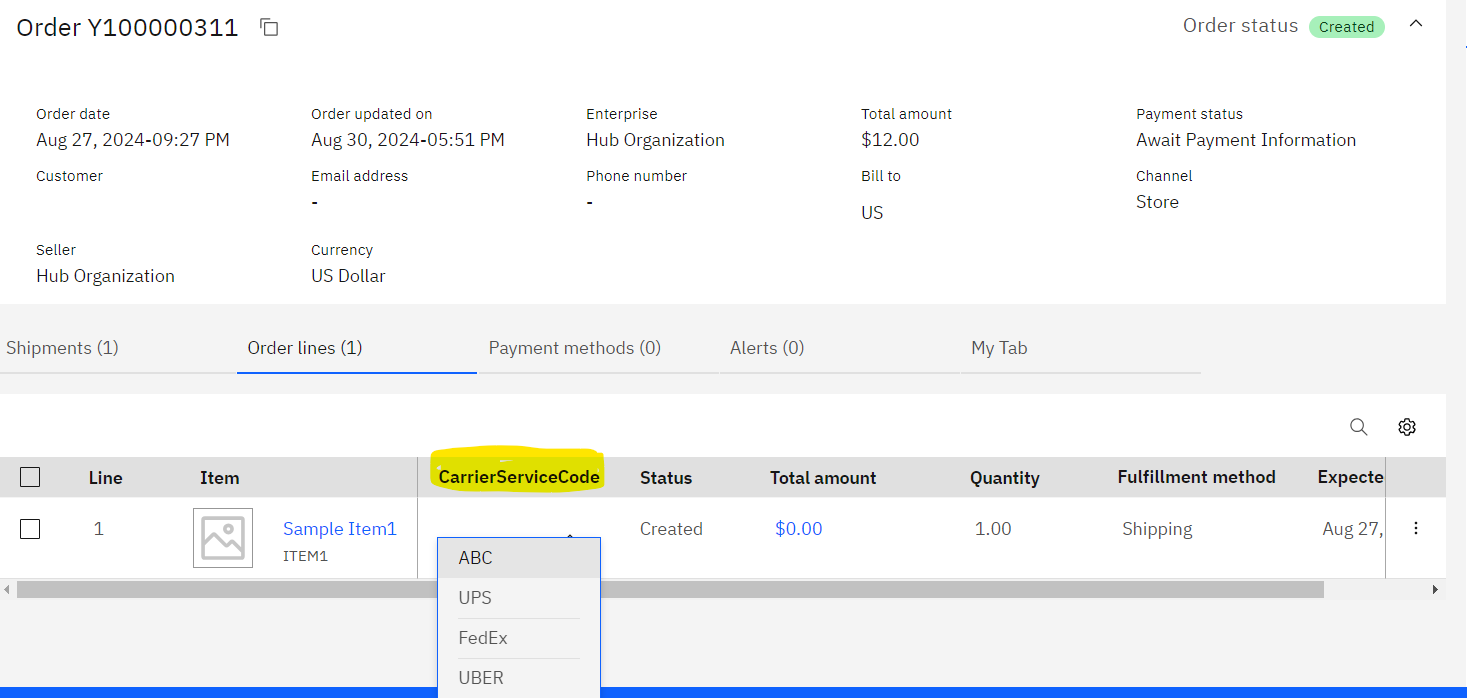




4. Restart Yarn.

|  |
| --- |
| #to stop yarn yarn stop-app #to start yarn yarn start-app |

5. Results:



## b. Modification Reasons List

1. Path: C:\IBM\oms10\repository\callcenter\cc-customcode\call-center-order\packages\order-details\src-custom\app\features\order\order-lines\order-lines.component.ts

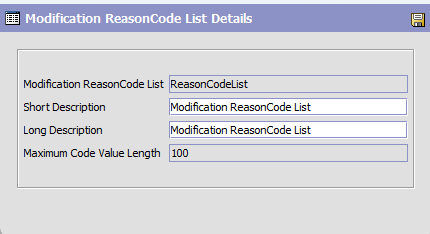
|  |
| --- |
| //Comment out the existing variable  //modificationReasonsList = [{ content: 'Gift Warp the Order' },{ content: 'Delivery on Weekend' }, { content: 'Delivery at Doorstep' }];  //Define empty List  public modificationReasonsList = [];  //In an existing function include the highlighted code  async openAddNoteCustomPopup(selectedOlk?, overflow?: boolean) {      console.log("openCustomPopup :: start", selectedOlk);      var selectedLines: any;      var olks: any;      var ohks: any;      var data: any;      data = "ReasonCodeList";  /\*here we are checking that the data is pass from toolbar or overflow\*/      if (overflow) {  //data is passing from overflow        selectedLines = selectedOlk;        olks = selectedLines[0].OrderLineKey;        ohks = selectedLines[0].OrderHeaderKey;      } else {        selectedLines = this.loadedOrderLines.filter((item) => selectedOlk.includes(item.OrderLineKey));        console.log("selectedLines", selectedLines);        ohks = selectedLines.map(x => x.OrderHeaderKey);        olks = selectedOlk;      }      //calling getcommoncode for dropdown      console.log("openAddNoteCustomPopup data: ", data);      await this.callgetCommonCodeList(data);      console.log("the getCommonCodeListOutput api output is ", this.getCommonCodeListOutput);      this.modificationReasonsList = this.getCommonCodeListOutput.CommonCodeList.CommonCode.map((item)=>        ({content:item.CodeShortDescription,}));  async callgetCommonCodeList(data: any) {      const getCommonCodeList = {        CommonCode: {          CodeType: data,          OrganizationCode: "DEFAULT"        }      };      console.log("input mashup :: ", getCommonCodeList)      try {        await this.bucCommOmsMashupService.callMashup('extn\_icc.order.order-lines.ExtngetCommonCodeList', getCommonCodeList, {}).toPromise()          .then(mashupOutput => {            this.bucCommOmsMashupService.getMashupOutput(mashupOutput, 'extn\_icc.order.order-lines.ExtngetCommonCodeList');            console.log("mashupOutput:::", mashupOutput);            this.getCommonCodeListOutput = mashupOutput.mashupResponse.controllerData.MashupRefs.MashupRef[0].Output;          });      } catch (error) {      }    } |

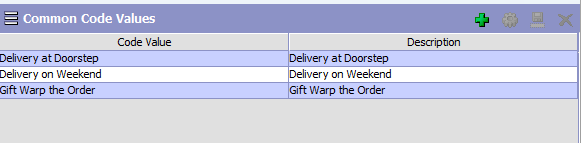
2. Create New Mashup xml as shown below :

Path : C:\IBM\oms10\extensions\icc\webpages\mashupxmls\additional\_mashups.xml

|  |
| --- |
| <mashup description="Get Common Code API" id="extn\_icc.order.order-lines.ExtngetCommonCodeList" mashuptype="XAPI" transactional="true" skipDataProvider="true">  <classInformation name="com.ibm.icc.common.mashups.ICCBaseMashup"/>  <API Name="getCommonCodeList">  <Input>  <CommonCode OrganizationCode="" CodeType="">  </CommonCode>  </Input>  <Template>  <CommonCodeList>  <CommonCode CodeShortDescription=""/>  </CommonCodeList>  </Template>  </API>  <AlternateResourceIds>  <AlternateResourceId altResourceId="ICCSYS00001"/>  </AlternateResourceIds>  </mashup> |

3. Create CommonCode List and CommonCode accordingly as shown below:





4. Restart Yarn.

|  |
| --- |
| #to stop yarn yarn stop-app #to start yarn yarn start-app |

5. Results:

