



Workshop NATO



Workshop for NATO ABAP/Fiori Elements

[Setup Workstation](#)

[Exercises](#)

[Blogs, Tips and Tricks](#)

[Additional Online Workshops](#)

Setup Workstation for Workshop.

Download latest version of Eclipse IDE for Enterprise Java and Web Developers ([2022-03](#))

Install [ADT](#) (ABAP Development Tools) for Eclipse.

Create SAP Trial Account and create a [SAP BTP ABAP Environment Trial User](#).

Exercises

1. [Create Package for Workshop](#)
2. [Create Dev Space in SAP Business Application Studio for Workshop](#)
3. [Create Simple CDS](#)
4. [Create Basic List Report](#)
5. [Build & Deploy our Application and Configure our Fiori Cloud Launchpad](#)
6. [Add Search and Filters to List Report](#)
7. [Create List Report Object Page](#)
8. [External Navigation and Visualiztion](#)
9. [Overview Page](#)
10. [Analytical List Page](#)
11. [Extend Fiori Elements List Report Object Page Application](#)
12. [Extend Fiori Elements Overview Page Application](#)
13. [RAP Basics](#)
14. [RAP Basics Delete, Update, Create](#)
15. [RAP with Draft version](#)

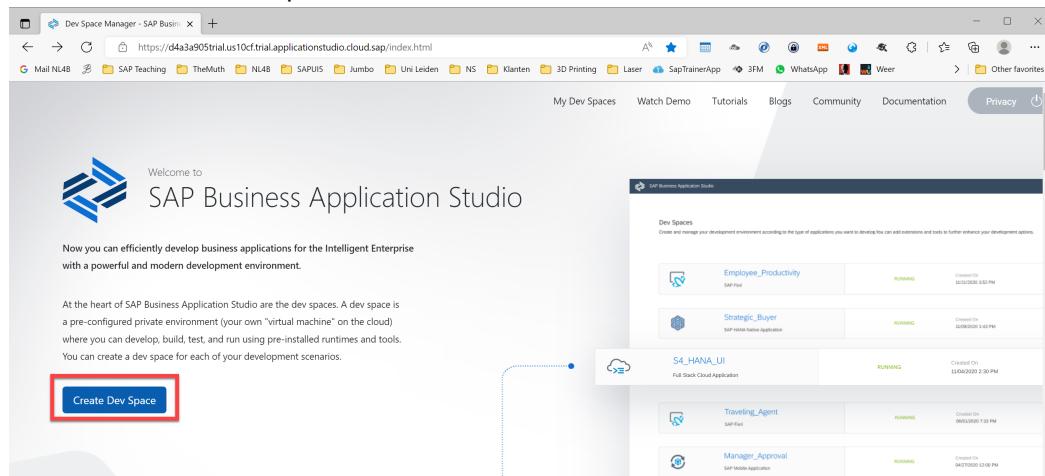
Create Package for Workshop

- Start Eclipse and open your ABAP cloud project.
- Right click and select **New ABAP Package**.
- Enter Name **Z_WKSP_##**, where ## is your assigned number.
- Enter Description **WKSP Exercises ##**
- Select **Next** and then **Finish**

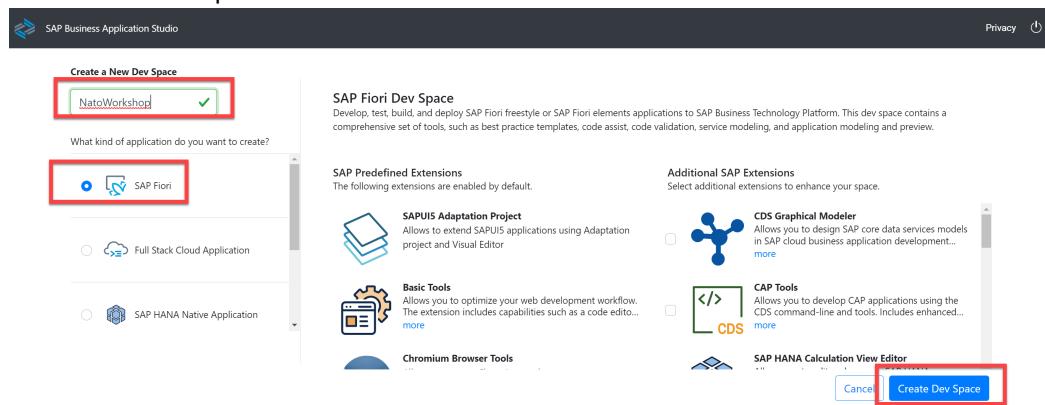
Create Dev Space in SAP Business Application Studio for Workshop

Start your SAP Business Application Studio.

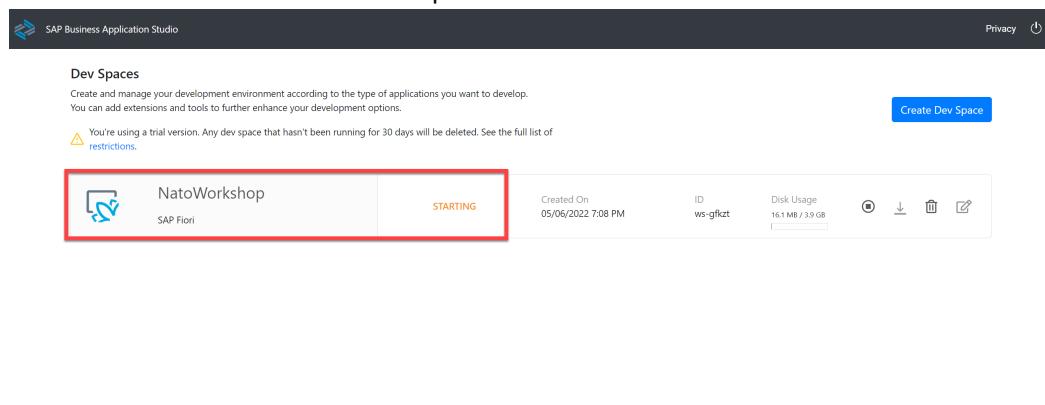
- Press the Create Dev Space button.



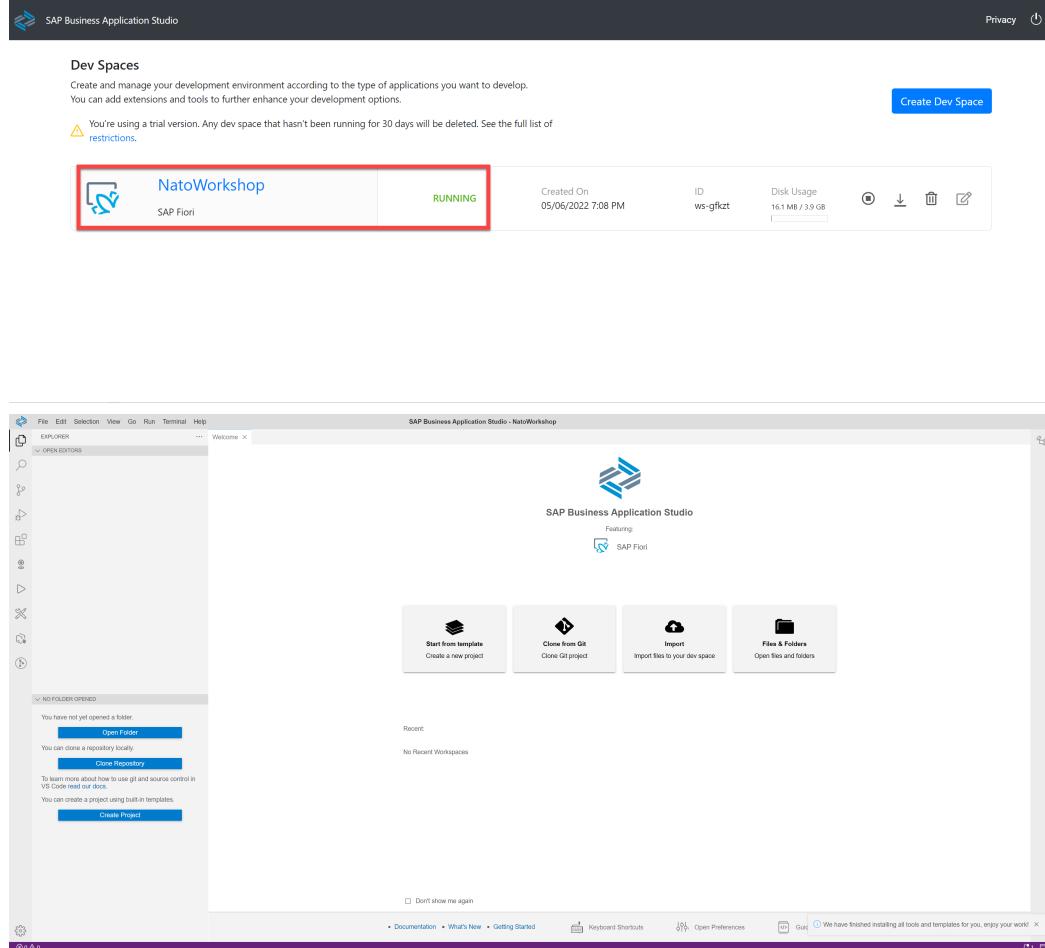
- Enter the name for your Dev Space, e.g.: **NatoWorkshop**, select **SAP Fiori** and press the Create Dev Space button.



- This will create and start the Dev Space.



- When the Dev Space is running you can click on the name and the Business Application Studio is opened.



Create a simple CDS

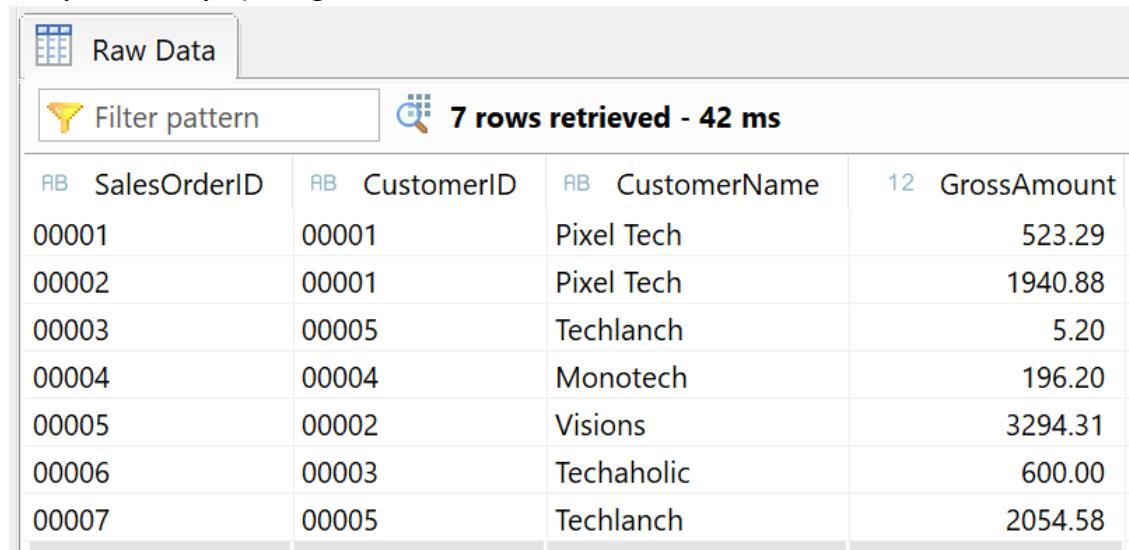
Create a simple CDS with name **ZWS##_CDS_SIMPLE** where ## is your number.
Add the fields: Id, CustomerId, _Customer.name and GrossAmount.

Template:

```
@AbapCatalog.sqlViewName: 'ZWK##CDSSIMPLE'
@AbapCatalog.compiler.compareFilter: true
@AbapCatalog.preserveKey: true
@AccessControl.authorizationCheck: #CHECK
@EndUserText.label: 'Simple CDS'
define view ZWS##_CDS_Simple as select from ztmcds9_i_so {
    key Id as SalesOrderID,
    CustomerId as CustomerID,
    _Customer.name as CustomerName,
    GrossAmount
}
```

Activate your CDS using CTRL-F3.

Test your CDS by opening it with **Data Preview**. This should result in a list of records.



SalesOrderID	CustomerID	CustomerName	GrossAmount
00001	00001	Pixel Tech	523.29
00002	00001	Pixel Tech	1940.88
00003	00005	Techlanch	5.20
00004	00004	Monotech	196.20
00005	00002	Visions	3294.31
00006	00003	Techaholic	600.00
00007	00005	Techlanch	2054.58

Create Service Definition

You are going to create a Service Definition in which you can list all the CDS views you want to expose in a OData service.

Right click on your Package and select **New Other ABAP Repository Object**, then type

Service and select **Service Definition**.

Enter Name **ZUI_WKSP_##** and enter a description.

Hint: Naming conventions are often used to specify if the service definition is used as a UI or API service.

Add your CDS as the entity name and give it a nice name like SimpleCDS.

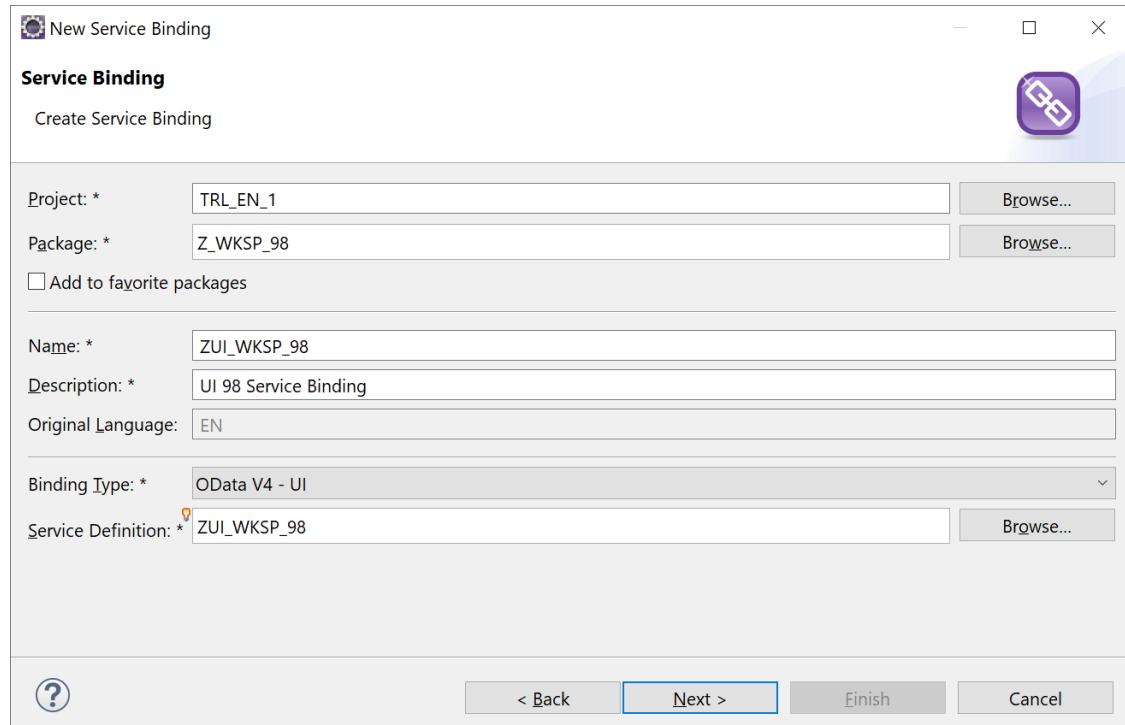
```
@EndUserText.label: 'UI ## Service Definition'
define service ZUI_WKSP_## {
    expose ZWS##_CDS_Simple as SimpleCDS;
}
```

Create Service Binding

Now we are going to expose the Service Definition by creating a Service Binding.

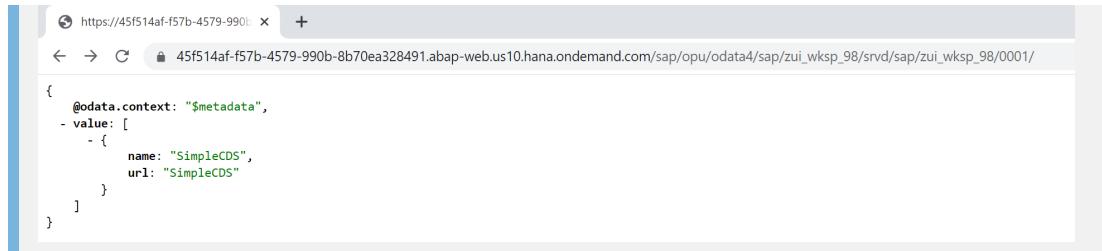
Right click on your Package and select **New Other ABAP Repository Object**, then type **Service** and select **Service Binding**.

Enter Name **ZUI_WKSP_##** and enter a description. Then select **Binding Type OData V4 - UI**, and then enter your Service Definition. You can start typing the first letters of your Service Definition and then use CTRL+SPACE for Auto Complete, or use the **Browse** button.



Activate your Service Binding and then push Publish (this may take a few seconds). Your Service Binding is now ready to be used.

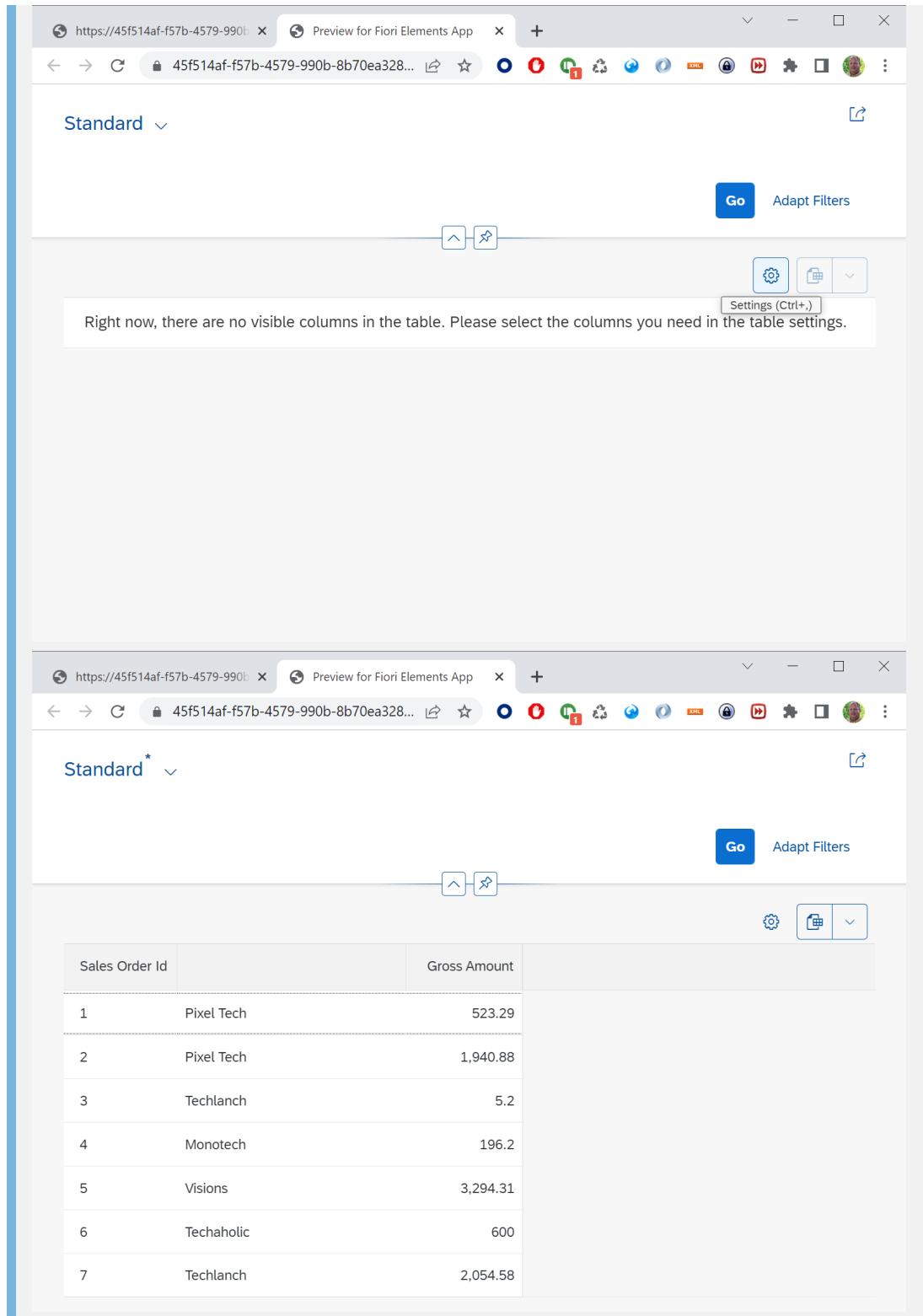
If you press the ServiceURL a browser will be started and it will open the ROOT URL for your service.



A screenshot of a web browser window displaying an OData JSON response. The URL in the address bar is `https://45f514af-f57b-4579-990b-8b70ea328491.abap-web.us10.hana.ondemand.com/sap/opu/odata4/sap/zui_wksp_98/srvd/sap/zui_wksp_98/0001/`. The page content shows a JSON object with the following structure:

```
{  
  "@odata.context": "$metadata",  
  "value": [  
    {  
      "name": "SimpleCDS",  
      "url": "SimpleCDS"  
    }  
  ]  
}
```

If you select an Entity Set and press the `Preview` button, it will open a browser with an example application. But you still need to goto the settings to select the columns you want to display.



Add Some UI Annotations

Now add some annotations to make the Preview application display lines directly without have to go to the settings.

Add Header to table and select by default the Sales Order Id, CustomerName and

GrossAmount.

Hint: @UI.headerInfo, @UI.lineItem

When you have added the annotations to your CDS, open the Preview again and see the result.

Option 1, Annotations in your CDS

```
@AbapCatalog.sqlViewName: 'ZWK##CDSSIMPLE'
@AbapCatalog.compiler.compareFilter: true
@AbapCatalog.preserveKey: true
@AccessControl.authorizationCheck: #CHECK
@EndUserText.label: 'Simple CDS'
@UI.headerInfo.typeName: 'Sales Order'
@UI.headerInfo.typeNamePlural: 'Sales Orders'
define view ZWS##_CDS_Simple as select from ztmcds9_i_so {
  @UI.lineItem: [{position: 10}]
    key Id as SalesOrderID,
    CustomerId as CustomerID,
  @UI.lineItem: [{position: 20},{ label: 'Customer Name' }]
    _Customer.name as CustomerName,
  @UI.lineItem: [{position: 30}]
    GrossAmount
}
```

Option 2, use a Metadata Extension

Create a Metadata Extension for your CDS. Add the annotation

@Metadata.allowExtensions: true to your Simple CDS above the *define* line.

```
@AbapCatalog.sqlViewName: 'ZWS##CDSBASIC'
@AbapCatalog.compiler.compareFilter: true
@AbapCatalog.preserveKey: true
@AccessControl.authorizationCheck: #CHECK
@EndUserText.label: 'Basic CDS'
@Metadata.allowExtensions: true
define view ZWS##_CDS_Basic as select from ztmcds9_i_so {
  key Id as SalesOrderID,
  CustomerId as CustomerID,
  _Customer.name as CustomerName,
  GrossAmount
}
```

Right click on your Package and select **New Other ABAP Repository Object**, then type **Core** and select **Metadata Extension**.

And add the annotations for the fields to this file.

```
@Metadata.layer: #CUSTOMER
@UI.headerInfo.typeName: 'Sales Order'
@UI.headerInfo.typeNamePlural: 'Sales Orders'
annotate view ZWS##_CDS_Basic with
{
    @UI.lineItem: [{position: 10 }]
    SalesOrderID;
    @UI.lineItem: [{position: 20 },{ label: 'Customer Name' }]
    CustomerName;
    @UI.lineItem: [{position: 30 }]
    GrossAmount;
}
```

Result:

The screenshot shows a Fiori Elements App interface titled "Preview for Fiori Elements App". The main content area displays a list of "Sales Orders (7)". The table has columns for "Sales Order Id", "Customer Name", and "Gross Amo...". Each row contains a number, a customer name, a gross amount, and a right-pointing arrow icon. The top right of the screen features a "Go" button, an "Enter" button, and a "Adapt Filters" link. There are also navigation icons for back, forward, search, and refresh.

Sales Order Id	Customer Name	Gross Amo...
1	Pixel Tech	523.29 >
2	Pixel Tech	1,940.88 >
3	Techlanch	5.2 >
4	Monotech	196.2 >
5	Visions	3,294.31 >
6	Techaholic	600 >
7	Techlanch	2,054.58 >

Create Basic List Report

Copy your `ZWS##_CDS_SIMPLE` to `ZWS##_CDS_LIST`.

Remember to change the `@AbapCatalog.sqlViewName` and `@EndUserText.label` values.

```

@AbapCatalog.sqlViewName: 'ZWK##CDSLST'
@AbapCatalog.compiler.compareFilter: true
@AbapCatalog.preserveKey: true
@AccessControl.authorizationCheck: #CHECK
@EndUserText.label: 'List Report'
@UI.headerInfo.typeName: 'Sales Order'
@UI.headerInfo.typeNamePlural: 'Sales Orders'
define view ZWS##_CDS_LIST as select from ztmcds9_i_so {
  @UI.lineItem: [{position: 10}]
    key Id as SalesOrderID,
    CustomerId as CustomerID,
  @UI.lineItem: [{position: 20},{ label: 'Customer Name' }]
    _Customer.name as CustomerName,
  @UI.lineItem: [{position: 30}]
    GrossAmount
}

```

Activate the CDS and add it to the Service Binding as **BasicList**

```

@EndUserText.label: 'UI 98 Service Definition'
define service ZUI_WKSP_98 {
  expose ZWS##_CDS_Simple as SimpleCDS;
  expose ZWS##_CDS_Basic as BasicCDS;
  expose ZWS##_CDS_LIST as BasicList;
}

```

Start your SAP Business Application Studio.

Open/Create your Dev Space.

There are two options to start the Fiori Template Wizard.

1. From the Welcome page you select *Start from template* and then select the **SAP Fiori application**
2. From the menu select *View* then *Find Command... (Ctrl+shift+P)* and then type **Fiori: Open Application Generator**

- Select the Application Type: **SAP Fiori elements**, and select the **List Report Object Page** and press **Next**.

In the Data source select *Connect to a System*, in the System select *your abap-cloud-default..(BTP)*. It will now look for the OData services on the Trial ABAP Cloud system, this may take a minute.

In the Service select your Service Binding ZUI_WKSP_##.

Data Source and Service Selection

Configure the data source and select a service.

Data source *

System ⓘ *

Service *

- Press **Next**
- Select your **BasicList** as the *Main entity*.

Entity Selection

Configure the selected service.

Main entity *

Automatically add table columns to the list page and a section to the object page if none already exists?

Yes No

- Press **Next**

Enter the Projects Attributes

Name	Value
Module name	basic-list
Application title	Basic List
Application namespace	nato.workshop
Description	Basic List
Project folder path	/home/user/projects
Minimum SAPUI5 version	leave as is
Add deployment configuration	Yes
Add FLP configuration	Yes
Configure advanced options	No

Project Attributes

Configure the main project attributes.

Module name  *

basic-list

Application title 

Basic List

Application namespace 

nato.workshop

Description 

Basic List

Project folder path *

/home/user/projects

Minimum SAPUI5 version  *

1.96.7 (Source system version)

Add deployment configuration

Yes No

Add FLP configuration

Yes No

Configure advanced options

Yes No

- Press Next

Deployment Configuration

Configure deployment settings

Please choose the target *

Cloud Foundry

▼

Destination name *

abap-cloud-default_abap-trial-d4a3a905trial-dev(SCP) - https://45f514af-f57b-4579-990b-8b70ea328491.abap

▼

Add application to managed application router?

Yes No

- Press Next

Enter Fiori Launchpad settings

Name	Value
Semantic Object	NATO
Action	BasicList
Title	Basic List
Subtitle (optional)	Workshop

Fiori Launchpad Configuration

Configure Fiori Launchpad settings

Semantic Object *

NATO

Action *

BasicList

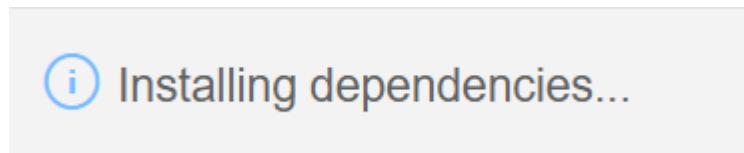
Title *

Basic List

Subtitle (optional)

Workshop

- Press Finish



- The installation of the dependencies may take a few moments.



- When is is done you will see a page with Application Information.

Application Information

Project Details

Module Name	nato.workshop.basiclist	Type	SAP Fiori elements
Application Title	Basic List	SAPUI5 Version	1.96.7
Namespace	nato.workshop	Project Type	EDMX Backend
Location	home/user/projects/basic-list/	Main Service	zui_wksp_98 (V4.0)
Files	package.json manifest.json		

Application Status

- Node modules are installed in directory 'node_modules'.
- Latest version of node module '@sap/ux-specification@1.96.11' installed.
- Latest version of node module '@sap/ux-ui5-tooling@1.5.5' installed.
- Latest version of node module '@sap/ux-ui5-fe-mockserver-middleware@1.5.5' installed.

What you can do

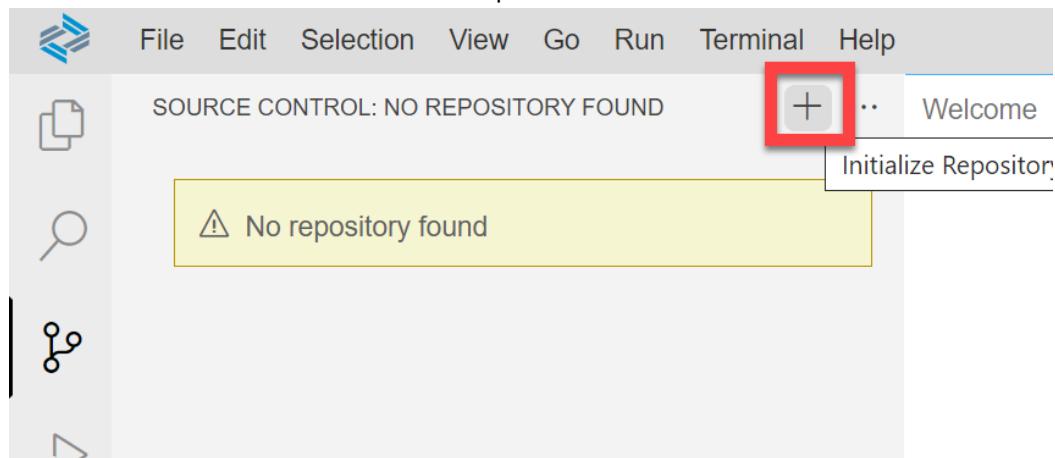
Preview Application Choose from start scripts to run the application preview.	Validate Application Validate the application.	Check Service Check service with annotation. You can copy annotations to local annotation file.	Manage Service Models You can sync metadata.xml with backend.
Local XML Annotations You can choose or create local XML annotation files.	Open Guided Development Guided Development helps solve common tasks.	Open Page Map Open the page map that shows application pages and navigation paths.	Add Deploy Config Add deploy configuration to the application.
Deploy Application Deploy the application according to the configuration by default stored in 'ui5-deploy.yaml'.	Undeploy Application Remove a deployed application from backend or cloud.	Check Node Modules Execute command that checks all node module dependencies for newer versions.	Add Variants Config Add Configuration for Variants Creation.
Maintain Mockdata Start editor for maintaining mock data.	Create Archive Zip the project excluding the node_modules for sharing (e.g. support cases)		

- Press *Preview Application* to start the application.

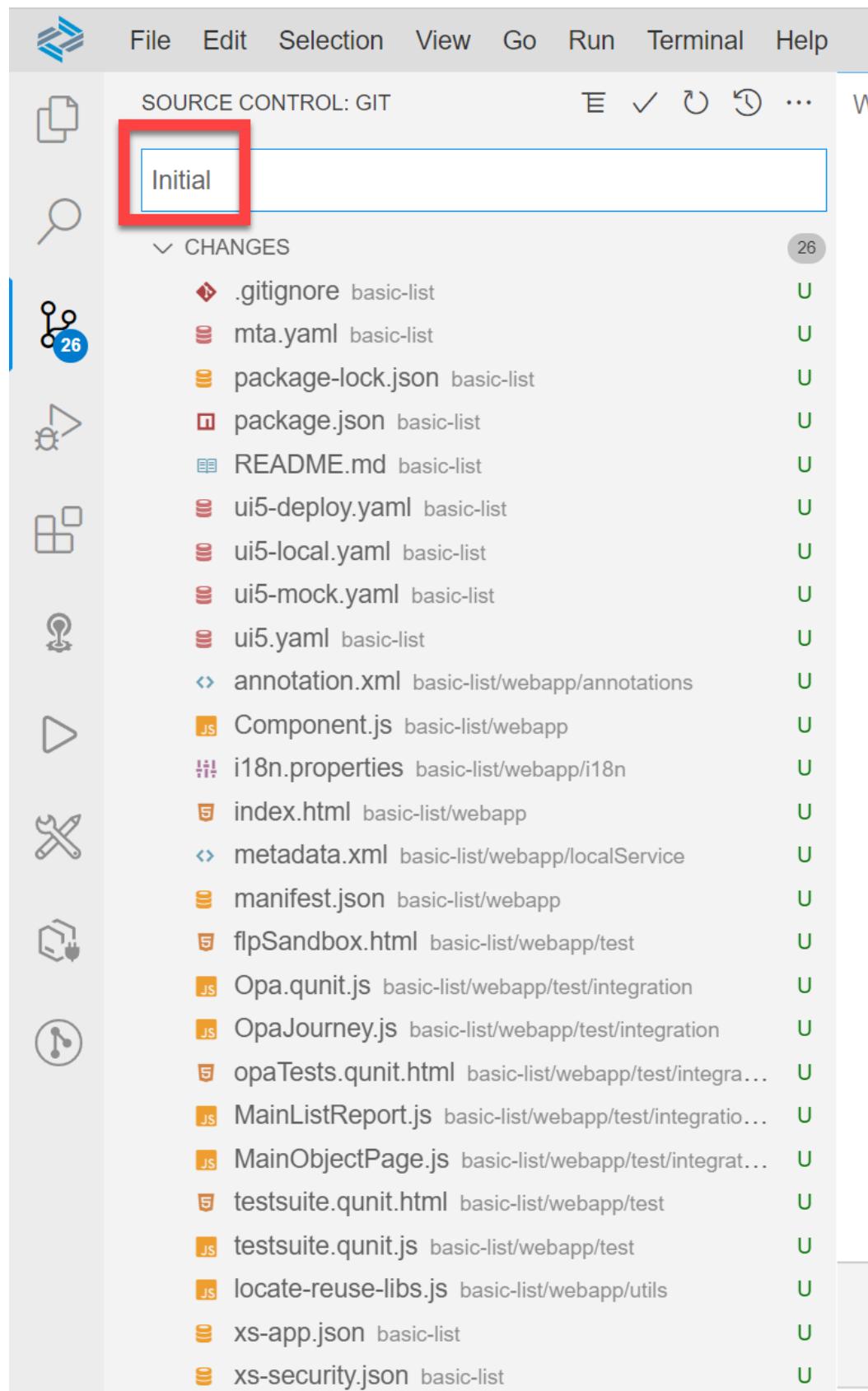
Open the project folder and activate GIT

- Open the projects folder.

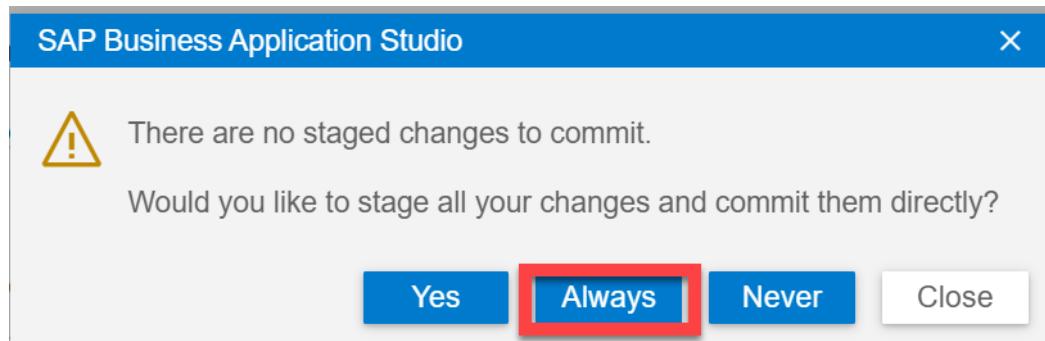
- Click on the *GIT* tab.  and press the + Plus button.



- This will instantiate the GIT source control. Enter a message, e.g.: **Initial** and press **CTRL+Enter**.



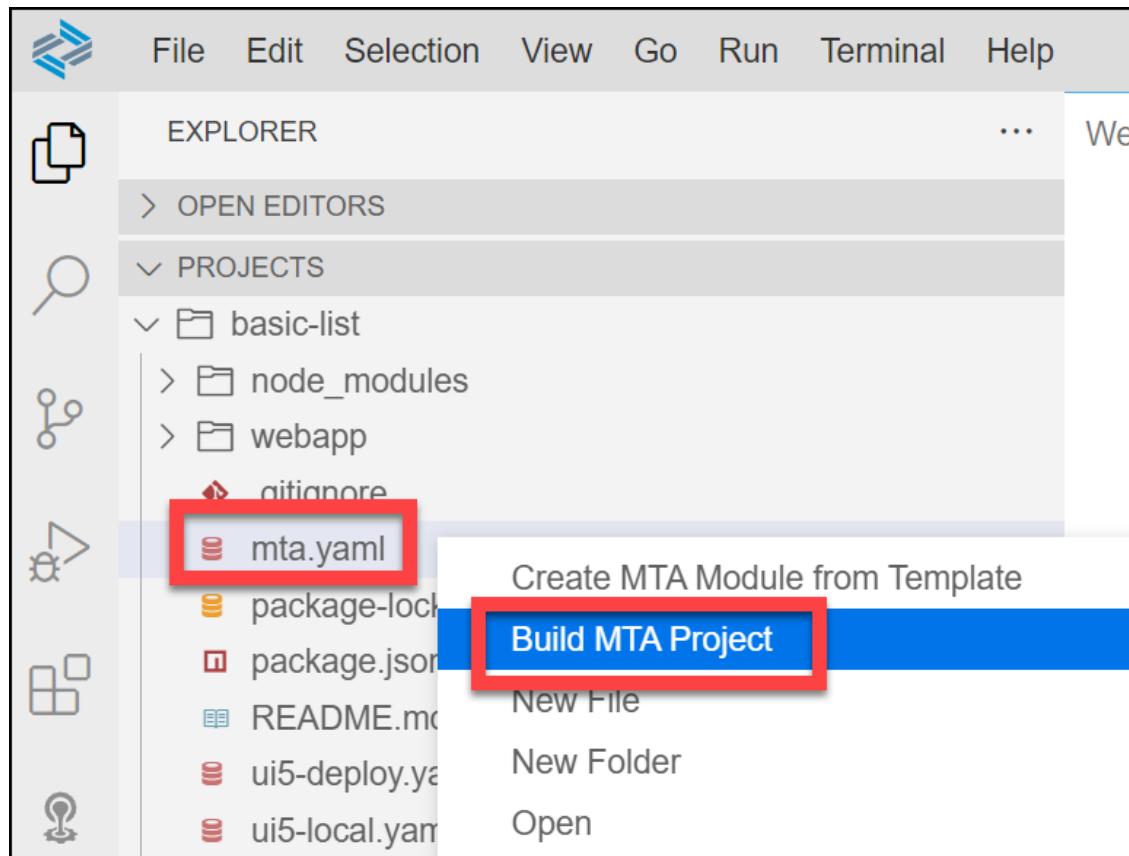
Select *Always* in the popup.



Build & Deploy Application and Configure Fiori Cloud Launchpad

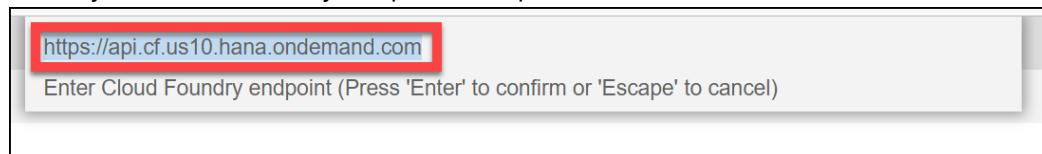
Build and Deploy

- Right click on the **mta.yaml** file and select *Build MTA Project*



This will start a terminal and it will create 3 new directories: *dist*, *mta_archives* and *resources*.

- Open the *mta_archives* folder and right click the **.mtar** file and select *Deploy MTA Archive*
- Check your Cloud Foundry Endpoint and press *Enter* to confirm.



- Enter your e-mail address and password of your trial account.
- Select your Organisation and Space

This will open another terminal that will deploy the **.mtar** to the cloud. This can take a moment.

- To see if all went well you can goto your SAP BTP Trial Cockpit and the option **HTML5 Applications** on the left side.

The screenshot shows the SAP BTP Cockpit interface. At the top, there is a navigation bar with a menu icon, the SAP logo, and the text "SAP BTP Cockpit". On the right side of the header, there are three buttons: a green button with a checkmark labeled "You", a blue button with a house icon labeled "Trial", and a grey button labeled "Sub...". Below the header, there is a sidebar with several items: "Overview", "Services" (with a dropdown arrow), "Service Marketplace", "Instances and Subscriptions" (which is highlighted with a light blue background), "Cloud Foundry" (with a dropdown arrow), "Spaces", "Quota Plans", "Org Members", "HTML5 Applications" (which is highlighted with a red border), "Connectivity" (with a dropdown arrow), "Destinations", and "Cloud Connectors".

- If you filter on *natoworkshop* you should see your application.

The screenshot shows the SAP BTP Cockpit interface. On the left, there is a sidebar with various navigation options: Overview, Services, Service Marketplace, Instances and Subscriptions, Cloud Foundry, Spaces, Quota Plans, Org Members, HTML5 Applications (which is selected), and Connectivity. The main area displays a message: "You are an active user, so your trial account was automatically extended for another 30-day interval." Below this, it says "Subaccount: trial - HTML5 Applications" and "Filtered: 1 of 22". A search bar contains the text "natoworkshop". The results table has columns for "Application Name" and "Active Version". One row is highlighted with a red box, showing "natoworkshopbasiclist" in the Application Name column and "0.0.1" in the Active Version column.

- Click on the application and your application will start.

The screenshot shows a web browser window titled "Basic List". The address bar shows the URL <https://d4a3a905trial.hana.ondemand.com/sap/bc/ui5/app/SalesOrders/SalesOrders.html>. The page title is "Sales Orders". At the top right, there are buttons for "Go" and "Adapt Filters". Below the title, there is a message: "To start, set the relevant filters." There is a table with columns: Sales Order Id, Customer Name, and Gross Amount.

Configure Fiori Launchpad Cloud

- Goto *Instances and Subscriptions* in your BTP Cockpit and goto the Launchpad Service

SAP BTP Cockpit

You are an active user, so your trial account was automatically extended for another 30-day interval.

Trial Home / d4a3a905trial / trial

Subaccount: trial - Instances and Subscriptions

All: 29

To manage the Cloud Foundry user-provided service instances, navigate to Cloud Foundry - Spaces, select the space, and then click the gear icon to open the service instance management interface.

Search All Services

Subscriptions (7) Instances (20) Environments (2)

Applications to which your subaccount is currently subscribed

Application	Plan
Cloud Transport Management	standard
Continuous Integration & Delivery	trial
Event Mesh	standard
Integration Suite	trial
SAP Business Application Studio	trial
Launchpad Service	standard
Workflow Management	saas-application

- In the Site Directory select the **provider manager**

Site Manager - Site Directory

Site Directory

This is a trial account.

+ Create Site

Import Site

NATO Workshop

Created May 7, 2022

- Refresh the HTML5 Apps

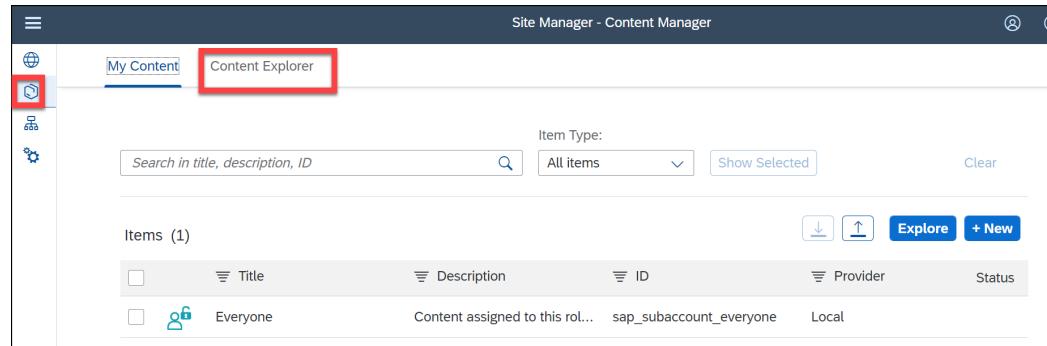
Site Manager - Provider Manager

Content Providers (1)

+ New

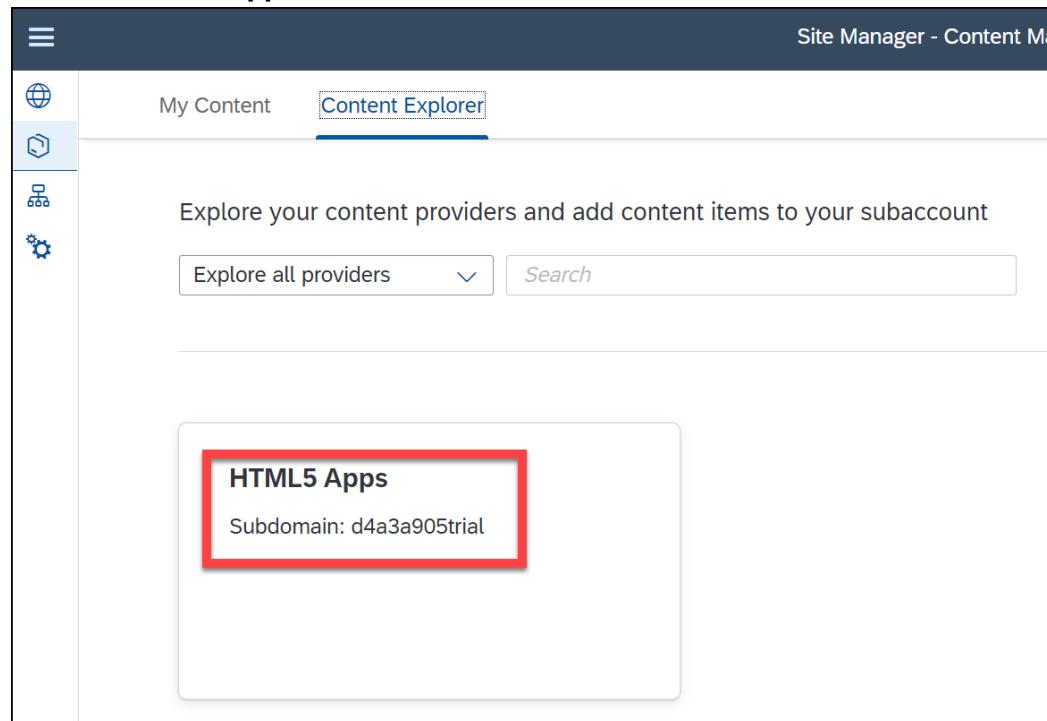
Title & Description	ID	Destinations / Sourc...	Last Modified...	Status	Actions
HTML5 Apps ⓘ Subdomain: d4a3a905trial Automatic Mode	saas_approuter	Source: HTML5 App Repository	Nov 12, 2021, 1:12:21 PM	Activated	

- Goto the **Content Manager** and click the tab **Content Explorer**



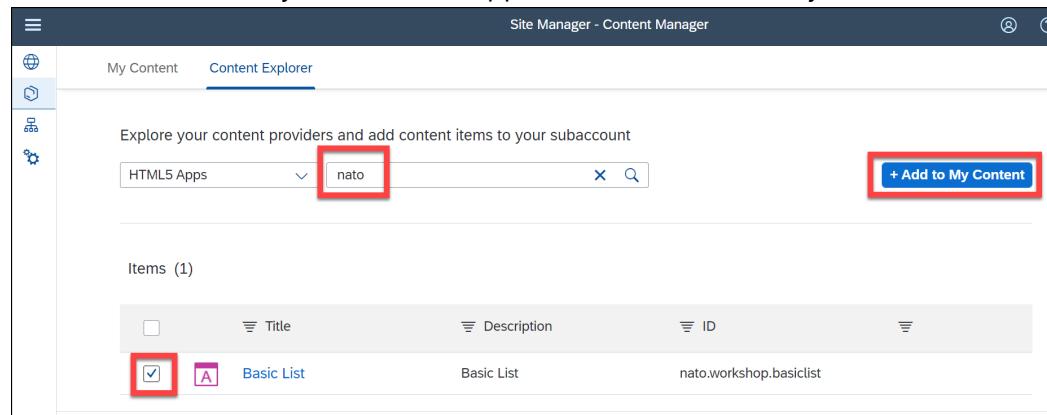
The screenshot shows the Site Manager - Content Manager interface. The top navigation bar has tabs for 'My Content' and 'Content Explorer'. The 'Content Explorer' tab is highlighted with a red box. Below the tabs is a search bar with placeholder 'Search in title, description, ID' and a dropdown for 'Item Type' set to 'All items'. There are also 'Show Selected' and 'Clear' buttons. A large section below is titled 'Items (1)'. It contains a single item listed in a table with columns: Title, Description, ID, Provider, and Status. The item is 'Everyone' with the description 'Content assigned to this rol...', ID 'sap_subaccount_everyone', Provider 'Local', and Status 'Local'. There are buttons for download, upload, explore, and new.

- Click the **HTML5 Apps**



The screenshot shows the Site Manager - Content Manager interface with the 'Content Explorer' tab selected. The main area displays a message: 'Explore your content providers and add content items to your subaccount'. Below this are buttons for 'Explore all providers' and 'Search'. A large callout box highlights the 'HTML5 Apps' section, which includes the text 'Subdomain: d4a3a905trial'.

- Use filter **nato** to list your **Basic List** application and add it to My Content



The screenshot shows the Site Manager - Content Manager interface with the 'Content Explorer' tab selected. The search bar at the top has 'HTML5 Apps' in the provider dropdown and 'nato' in the search input field, both highlighted with red boxes. To the right of the search bar is a blue button '+ Add to My Content' with a red box around it. Below the search bar is a table titled 'Items (1)'. It lists one item: 'Basic List' with ID 'nato.workshop.basiclist'. The 'Title' column shows a checked checkbox icon and a small icon for 'Basic List'.

- Goto the tab **My Content**, here you see your **Basic List** application.
Now lets create a Role.

Site Manager - Content Manager

My Content Content Explorer

Item Type: Search in title, description, ID All items Show Selected Clear

Items (2)

	Title	Description	ID	Provider
<input type="checkbox"/>	Basic List	Basic List	nato.workshop.basiclist	HTML5 Apps
<input type="checkbox"/>	Everyone	Content assigned to this role...	sap_subaccount_everyone	Local

Explore + New

App
 Catalog
 Group
 Role

- Enter the *Title*, *ID* and *Description*, then click on the search block under Assign Items; you now see a your application. Press the + button behind your application, then *Save* and then *Back*

Site Manager - Content Manager

ROLE Workshop

PROPERTIES TRANSLATION

General

Title: * Workshop

ID: NatoWorkshop

Description: Workshop for NATO

Assignments

Assign Items: + Search for items to assign

Results (1/1)

Basic List HTML5 Apps

Save Cancel

- We now have a new Role

Now lets create a Group.

Site Manager - Content Manager

My Content Content Explorer

Item Type: Search in title, description, ID All items Show Selected Clear

Items (3)

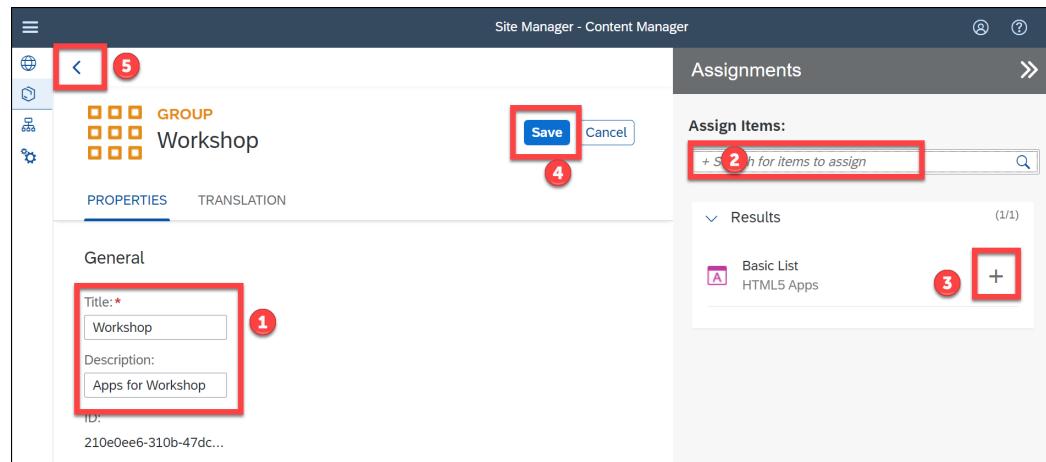
	Title	Description	ID	Provider
<input type="checkbox"/>	Basic List	Basic List	nato.workshop.basiclist	HTML5 Apps
<input type="checkbox"/>	Everyone	Content assigned to this role...	sap_subaccount_everyone	Local
<input type="checkbox"/>	Workshop	Workshop for NATO	NatoWorkshop	Local

Explore + New

App
 Catalog
 Group
 Role

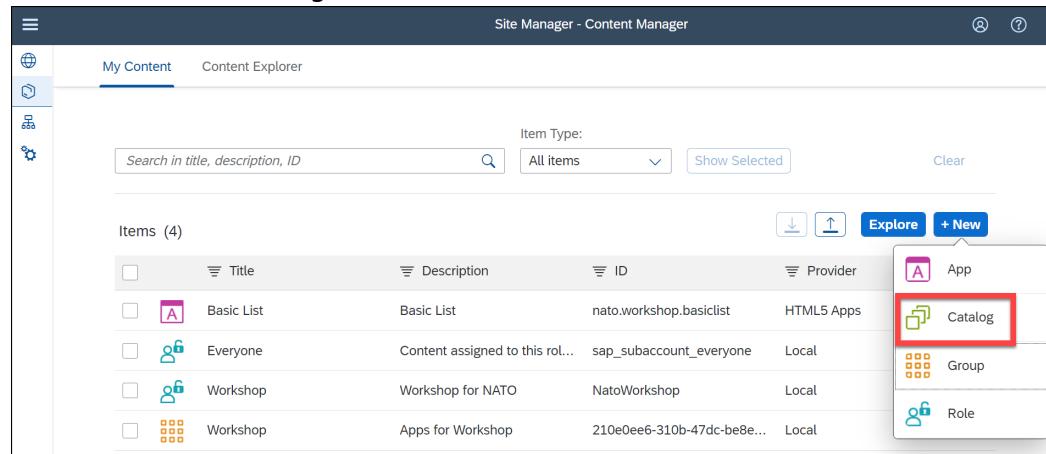
- Enter the *Title* and *Description*, then click on the search block under Assign Items; you now see a your application. Press the + button behind your application, then

Save and then Back

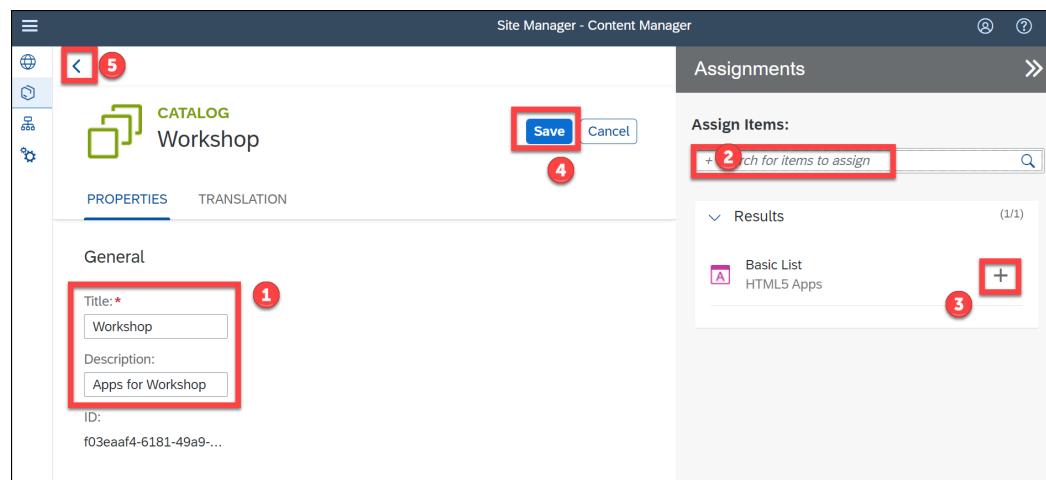


- We now have a new Group

Now lets create a Catalog



- Enter the *Title* and *Description*, then click on the search block under Assign Items; you now see a your application. Press the + button behind your application, then Save and then Back!



- We now have a new Catalog

The screenshot shows the Site Manager - Content Manager interface. The left sidebar has icons for Home, My Content (selected), Content Explorer, and Settings. The main area is titled "My Content" with "Content Explorer" as a tab. A search bar at the top says "Search in title, description, ID". Below it is a table titled "Items (5)" with columns: Title, Description, ID, Provider, and Status. The items listed are:

	Title	Description	ID	Provider	Status
<input type="checkbox"/>	Basic List	Basic List	nato.workshop.basiclist	HTML5 Apps	
<input type="checkbox"/>	Everyone	Content assigned to this role	sap_subaccount_everyone	Local	
<input type="checkbox"/>	Workshop	Workshop for NATO	NatoWorkshop	Local	
<input type="checkbox"/>	Workshop	Apps for Workshop	210e0ee6-310b-47dc-be8e...	Local	
<input type="checkbox"/>	Workshop	Apps for Workshop	f03eaaf4-6181-49a9-9adb...	Local	

Buttons at the bottom right include "Explore" and "+ New".

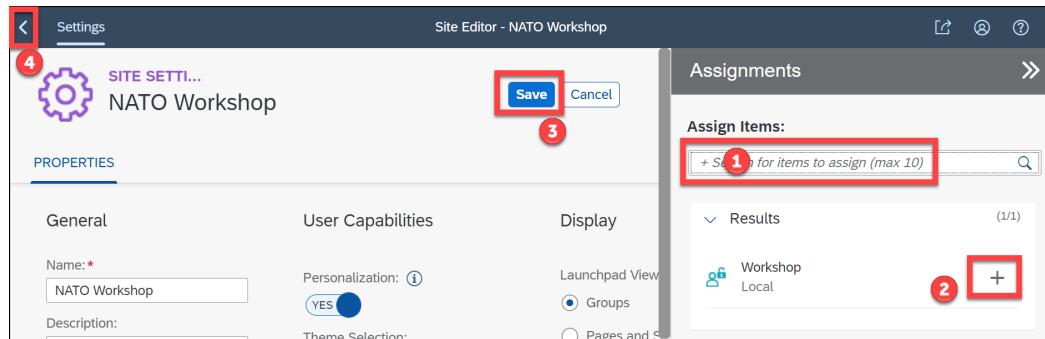
- Next we assign the created Role to our Site

The screenshot shows the Site Manager - Site Directory interface. The left sidebar has icons for Home (highlighted with a red circle containing a '1'), My Content, Content Explorer, and Settings. The main area is titled "Site Directory" with a message "This is a trial account.". It shows two cards: one for creating a site and one for the "NATO Workshop" site. The "NATO Workshop" card includes a "Site Settings" button with a red circle containing a '1'.

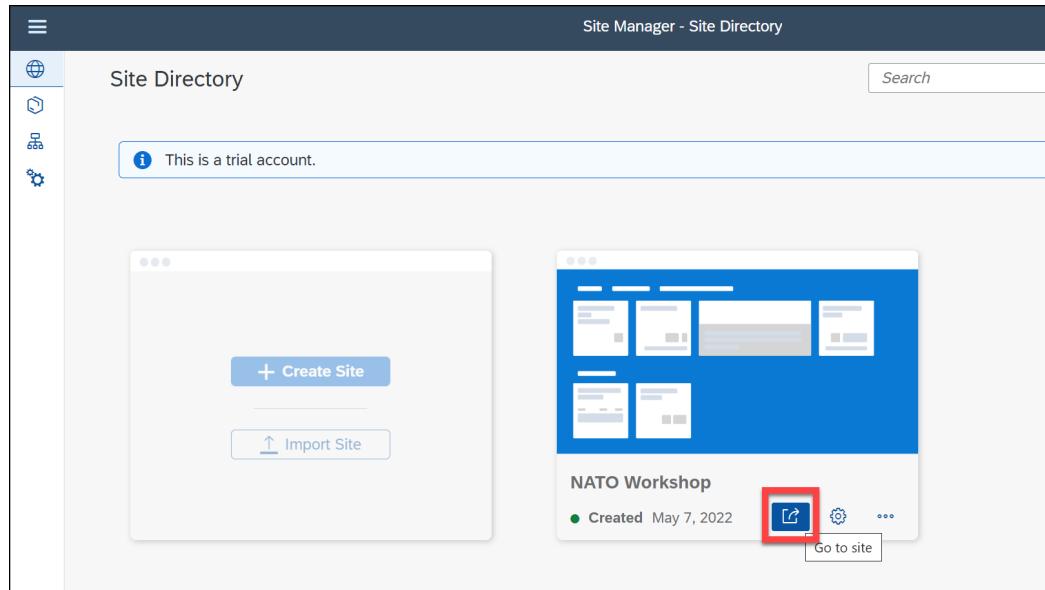
- We *Edit* the Site

The screenshot shows the Site Editor - NATO Workshop interface. The left sidebar has "Settings" selected. The main area shows the "NATO Workshop" site settings. On the right, there's a "Properties" section with tabs for General, User Capabilities, and Display. An "Edit" button is highlighted with a red box. To the right is an "Assignments" panel with a "Display Assigned Items:" search bar and a "Roles" section showing "(0/0)".

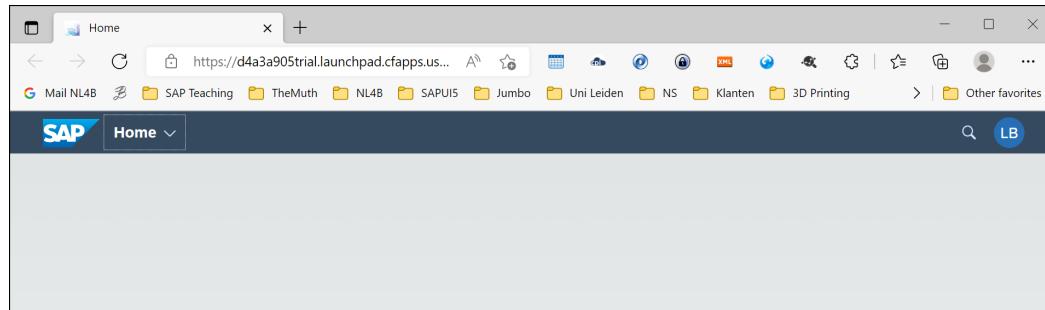
- Click on the search block under Assign Items; you now see a your Role. Press the + button behind your role, then Save and then Back



- Goto the Site/Launchpad



- Our Launchpad is still empty.



This is because we have not yet assigned the new Role to our user.

- Goto your BTP Cockpit and select **Role Collections**, Filter on **workshop** and click on your Role

SAP BTP Cockpit - Overview

You are an active user, so your trial account was automatically extended for another 30-day interval.

Subaccount: trial - Role Collections

Name	Description	Roles
NatoWorkshop	Workshop - Workshop for NATO	

Learn more about [building roles](#) and [maintain](#)

Role Collections

- Goto *Edit* mode and enter your e-mail address of your trial account in the fields **ID** and **E-Mail** and press *Save*

NatoWorkshop EDIT MODE

Description: Workshop - Workshop for NATO

Roles Users User Groups Attribute Mappings Description

Search

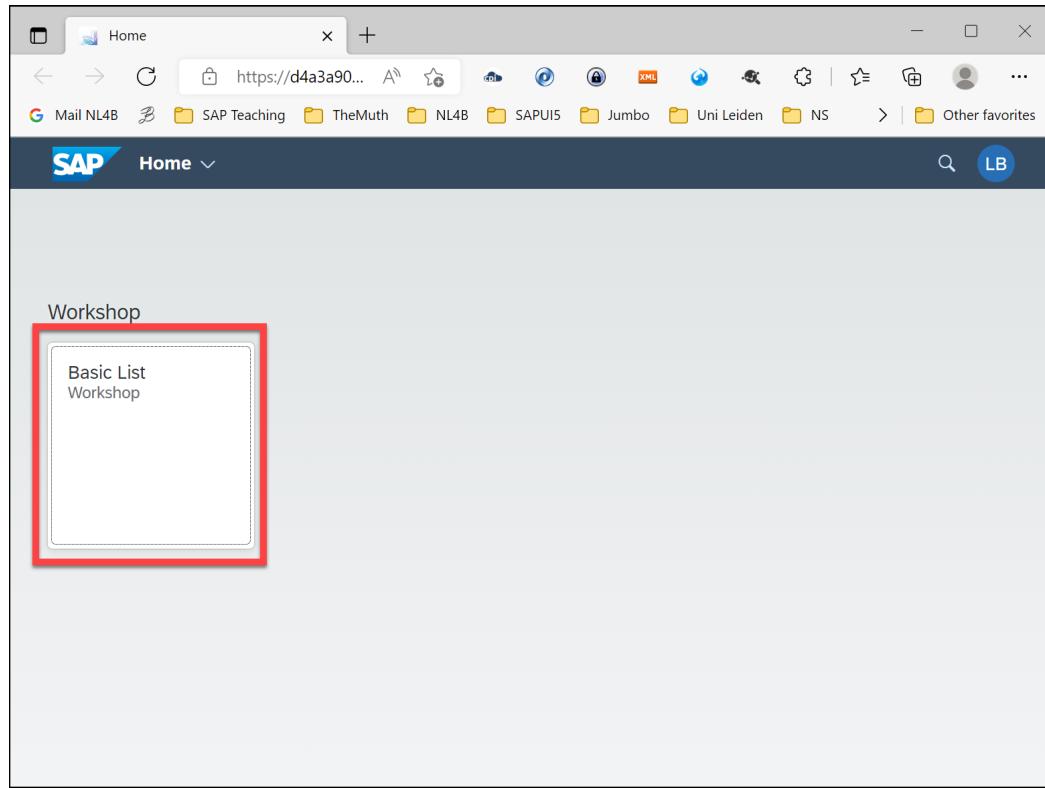
Role Name	Role Template	Application Identifier

Users

ID	Identity Provider	E-Mail	First Name	Last Name
leon.boeijen@nl4b.com	Default identity provider	leon.boeijen@nl4b.com		

User Groups

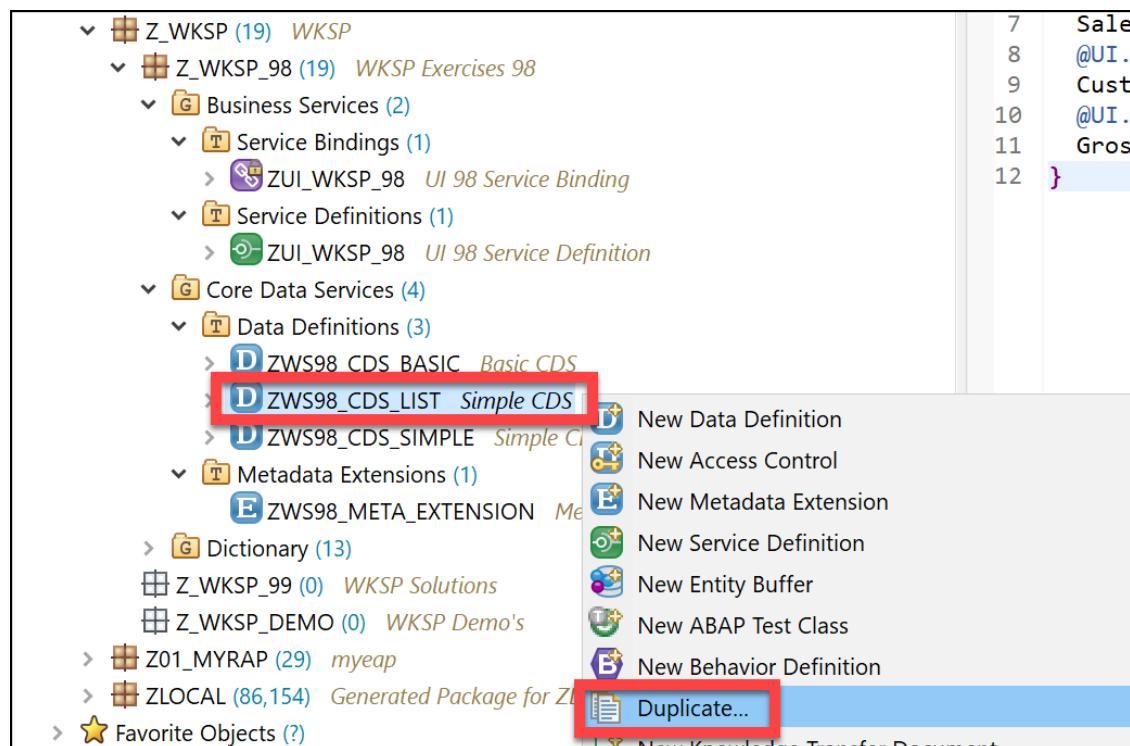
- Now lets start our Launchpad again.



Hint: it is sometimes needed to start a browser in *Private/Incognito mode* and lauch the Fiori Launchpad again, to make your changes available.

Add Search and Filters to List Report

Task 1: Duplicate CDS **ZWS##_CDS_LIST** to **ZWS##_CDS_LIST_SEARCH**



Make sure you change the `@AbapCatalog.sqlViewName`.

Add extra field for CreationDate and Customer Email.

```

@AbapCatalog.sqlViewName: 'ZWK##CDSLISTSRCH'
@AbapCatalog.compiler.compareFilter: true
@AbapCatalog.preserveKey: true
@AccessControl.authorizationCheck: #CHECK
@EndUserText.label: 'List Report'
@UI.headerInfo.typeName: 'Sales Order'
@UI.headerInfo.typeNamePlural: 'Sales Orders'
define view ZWS##_CDS_LIST_SEARCH
  as select from ztmcds9_i_so
{
  @UI.lineItem: [{position: 10}]
  key Id
  as SalesOrderID,
    CustomerId
  as CustomerID,
    @UI.lineItem: [{position: 20},{ label: 'Customer Name' }]
    _Customer.name
  
```

```

    as CustomerName,
    @UI.lineItem: [{position: 30 },{ label: 'E-mail' }]
    _Customer.email
  as CustomerEmail,
    @UI.lineItem: [{position: 40 }]
    GrossAmount,
    @UI.lineItem: [{position: 50},{ label: 'Creation Date' }]
    cast( left(cast(CreationDateTime as abap.char( 30)),8) as
abap.dats) as CreationDate
}

```

Task 2: Add the new CDS to your *Service Definition*

```

@EndUserText.label: 'UI ## Service Definition'
define service ZUI_WKSP_## {
  expose ZWS##_CDS_Simple as SimpleCDS;
  expose ZWS##_CDS_Basic as BasicCDS;
  expose ZWS##_CDS_LIST as BasicList;
  expose ZWS##_CDS_LIST_SEARCH as BasicSearch;
}

```

Task 3: Create a new Fiori elements application with the new CDS

Create a new Fiori Application using the Template Wizard for a *List Report*

Field	Value
Data source	Connect to a System
System	abap-cloud-default_xx-dev (BTP)
Service	ZUI_WKSP_##
Main entity	BasicSearch
Module name	basic-search
Application title	Basic List with Search
Application namespace	nato.workshop
Description	Basic List with Search
Project folder path	/home/user/projects
Add deployment configuration	Yes

Field	Value
Add FLP configuration	Yes
Deployment Target	Cloud Foundry
Destination name	abap-cloud-default_xx(SCP)
Add application to managed application router	Yes
Semantic Object	NATO
Action	BasicSearch
Title	Basic List with Search
Subtitle	Workshop

- Data Source and Service Selection

Data Source and Service Selection

Configure the data source and select a service.

Data source *

Connect to a System

System  *

abap-cloud-default_abap-trial-d4a3a905trial-dev (BTP)

Service *

ZUI_WKSP_98 > ZUI_WKSP_98 (0001) - OData V4

- Entity Selection

Entity Selection

Configure the selected service.

Main entity *

BasicSearch

Automatically add table columns to the list page and a section to the object page if none already exists?

Yes No

- Project Attributes

Project Attributes

Configure the main project attributes.

Module name ? *

basic-search

Application title ?

Basic List with Search

Application namespace ?

nato.workshop

Description ?

Basic List with Search

Project folder path *

/home/user/projects

Minimum SAPUI5 version ? *

1.96.7 (Source system version)

Add deployment configuration





yes



NO

Add FLP configuration



Yes



No

Configure advanced options



Yes



No

- Deployment Configuration

Deployment Configuration

Configure deployment settings

Please choose the target *

Cloud Foundry

Destination name *

abap-cloud-default_abap-trial-d4a3a905trial-dev(SCP) - https://

Add application to managed application router?



Yes



- Fiori Launchpad Configuration

Fiori Launchpad Configuration

Configure Fiori Launchpad settings

Semantic Object *

NATO

Action *

BasicSearch

Title *

Basic List with Search

Subtitle (optional)

Workshop

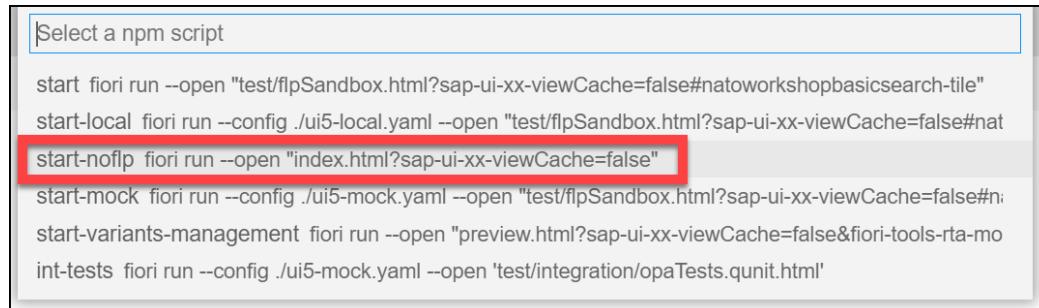
- Preview Application

</> **What you can do**

 **Preview Application**

Choose from start scripts to run the application preview.

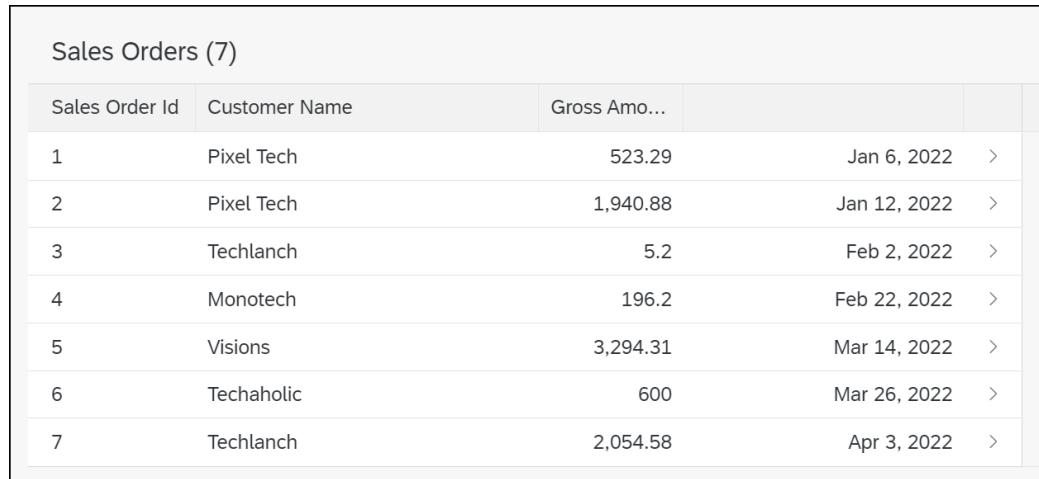
- Select start-noflp, this is the fastest test.



```
>Select a npm script

start fiori run --open "test/flpSandbox.html?sap-ui-xx-viewCache=false#natoworkshopbasicsearch-tile"
start-local fiori run --config ./ui5-local.yaml --open "test/flpSandbox.html?sap-ui-xx-viewCache=false#nat
start-noflp fiori run --open "index.html?sap-ui-xx-viewCache=false"
start-mock fiori run --config ./ui5-mock.yaml --open "test/flpSandbox.html?sap-ui-xx-viewCache=false#n:
start-variants-management fiori run --open "preview.html?sap-ui-xx-viewCache=false&fiori-tools-rt-a-mo
int-tests fiori run --config ./ui5-mock.yaml --open 'test/integration/opaTests.qunit.html'
```

- Result



Sales Order Id	Customer Name	Gross Amo...		
1	Pixel Tech	523.29	Jan 6, 2022	>
2	Pixel Tech	1,940.88	Jan 12, 2022	>
3	Techlanch	5.2	Feb 2, 2022	>
4	Monotech	196.2	Feb 22, 2022	>
5	Visions	3,294.31	Mar 14, 2022	>
6	Techaholic	600	Mar 26, 2022	>
7	Techlanch	2,054.58	Apr 3, 2022	>

Task 4: Add Search to CDS view

- Add a `@Search.searchable: true` annotation to your CDS view before the `define view....`
- Set fields `CustomerName` and `CustomerEmail` as default search elements.
- Set Fuzzy search level at 60%.
- Activate your CDS and Reload your fiori application to see the results.

```
@AbapCatalog.sqlViewName: 'ZWK##CDSLISTSRCH'
@AbapCatalog.compiler.compareFilter: true
@AbapCatalog.preserveKey: true
@AccessControl.authorizationCheck: #CHECK
@EndUserText.label: 'List Report'
@UI.headerInfo.typeName: 'Sales Order'
@UI.headerInfo.typeNamePlural: 'Sales Orders'
@Search.searchable: true
define view ZWS##_CDS_LIST_SEARCH
  as select from ztmcds9_i_so
{
  @UI.lineItem: [{position: 10 }]
  key Id
  as SalesOrderID,
```

```

CustomerId
as CustomerID,
    @UI.lineItem: [{position: 20 },{ label: 'Customer Name' }]
    @Search.defaultSearchElement: true
        _Customer.name
as CustomerName,
    @UI.lineItem: [{position: 30 },{ label: 'E-mail' }]
    @Search.defaultSearchElement: true
    @Search.fuzzinessThreshold : 0.6
        _Customer.email
as CustomerEmail,
    @UI.lineItem: [{position: 40 }]
GrossAmount,
    @UI.lineItem: [{position: 50},{ label: 'Creation Date' }]
    cast( left(cast(CreationDateTime as abap.char( 30)),8) as
abap.dats) as CreationDate
}

```

- Result

Sales Order Id	Customer Name	E-mail	Gross Amo...	Creation Date
1	Pixel Tech	info@Pixel.lu	523.29	Jan 6, 2022 >
2	Pixel Tech	info@Pixel.lu	1,940.88	Jan 12, 2022 >
5	Visions	mail@visions.lu	3,294.31	Mar 14, 2022 >

- Play a bit with the fuzzy search levels.

Task 4: Add Selection fields for Filtering

- Add filters for fields *CustomerId* and *CreationDate*.

```

@AbapCatalog.sqlViewName: 'ZWK##CDSLISTSRCH'
@AbapCatalog.compiler.compareFilter: true
@AbapCatalog.preserveKey: true
@AccessControl.authorizationCheck: #CHECK
@EndUserText.label: 'List Report'
@UI.headerInfo.typeName: 'Sales Order'
@UI.headerInfo.typeNamePlural: 'Sales Orders'
@Search.searchable: true
define view ZWS##_CDS_LIST_SEARCH
    as select from ztmccls9_i_so
{
    @UI.lineItem: [{position: 10 }]
    key Id

```

```

as SalesOrderID,
    @UI.selectionField: [{position: 10}]
    CustomerId
as CustomerID,
    @UI.lineItem: [{position: 20 },{ label: 'Customer Name' }]
    @Search.defaultSearchElement: true
    @Search.fuzzinessThreshold : 0.6
    _Customer.name
as CustomerName,
    @UI.lineItem: [{position: 30 },{ label: 'E-mail' }]
    @Search.defaultSearchElement: true
    _Customer.email
as CustomerEmail,
    @UI.lineItem: [{position: 40 }]
    GrossAmount,
    @UI.lineItem: [{position: 50},{ label: 'Creation Date' }]
    @UI.selectionField: [{position: 20}]
    cast( left(cast(CreationDateTime as abap.char( 30)),8) as
abap.dats) as CreationDate
}

```

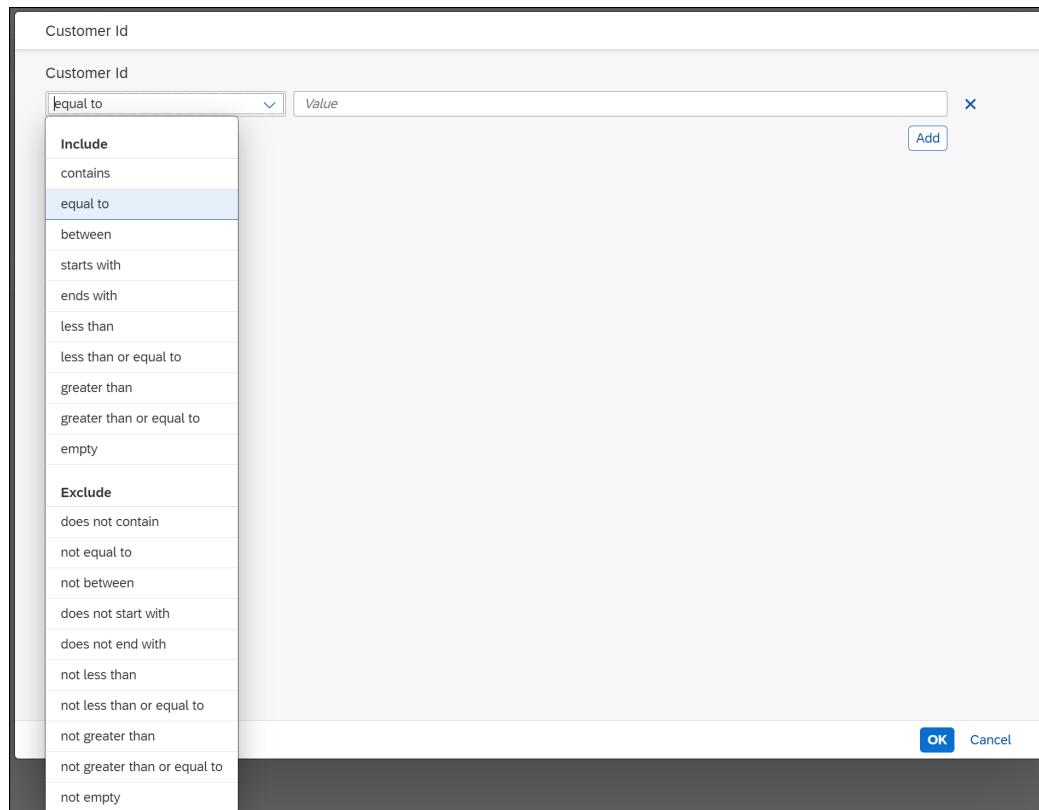
- Result

Sales Orders (7)				
Sales Order Id	Customer Name	E-mail	Gross Amo...	Creation Date
1	Pixel Tech	info@Pixel.lu	523.29	Jan 6, 2022 >
2	Pixel Tech	info@Pixel.lu	1,940.88	Jan 12, 2022 >
3	Techlanch	info@techlanch.be	5.2	Feb 2, 2022 >

- With active filter

Sales Orders (2)				
Sales Order Id	Customer Name	E-mail	Gross Amo...	Creation Date
1	Pixel Tech	info@Pixel.lu	523.29	Jan 6, 2022 >
2	Pixel Tech	info@Pixel.lu	1,940.88	Jan 12, 2022 >

- Basic Filter Screen



Task 5: Add Value Helps to Filters

- Have a look at CDS **ZTMCDS9_C_VH_CUST** in package **Z_TMC**.
- Use this CDS as Value Help for the *Customerid*.
- Add `@Consumption.filter.selectionType: #INTERVAL` to the *CreationDate* filter and see the result.

```

@AbapCatalog.sqlViewName: 'ZWK##CDSLISTSRCH'
@AbapCatalog.compiler.compareFilter: true
@AbapCatalog.preserveKey: true
@AccessControl.authorizationCheck: #CHECK
@EndUserText.label: 'List Report'
@UI.headerInfo.typeName: 'Sales Order'
@UI.headerInfo.typeNamePlural: 'Sales Orders'
@Search.searchable: true
define view ZWS##_CDS_LIST_SEARCH
  as select from ztmcds9_i_so
    association [1] to ztmcds9_c_vh_cust as _CustomerVH
      on $projection.CustomerID = _CustomerVH.SoldToParty
{
  @UI.lineItem: [{position: 10}]
  key Id
  as SalesOrderID,

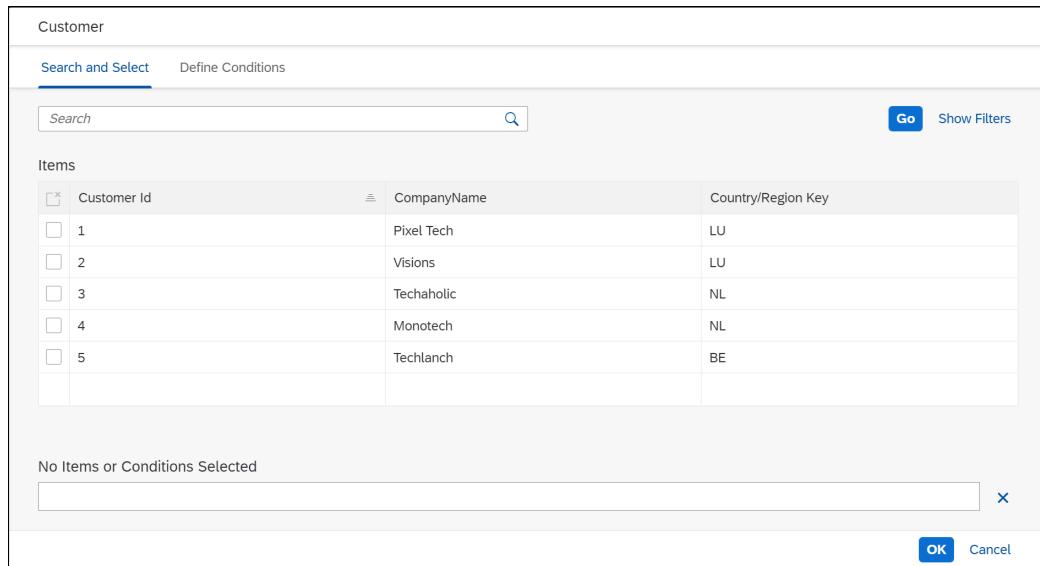
```

```

@UI.selectionField: [{position: 10}]
@Consumption.valueHelpDefinition: [{ entity:{ name:
'ZTMCDS9_C_VH_CUST', element: 'SoldToParty' }}]
    CustomerId
as CustomerID,
    @UI.lineItem: [{position: 20 },{ label: 'Customer Name' }]
    @Search.defaultSearchElement: true
    @Search.fuzzinessThreshold : 0.6
        _Customer.name
as CustomerName,
    @UI.lineItem: [{position: 30 },{ label: 'E-mail' }]
    @Search.defaultSearchElement: true
        _Customer.email
as CustomerEmail,
    @UI.lineItem: [{position: 40 }]
    GrossAmount,
    @UI.lineItem: [{position: 50},{ label: 'Creation Date' }]
    @UI.selectionField: [{position: 20}]
    @Consumption.filter.selectionType: #INTERVAL
        cast( left(cast(CreationDateTime as abap.char( 30)),8) as
abap.dats) as CreationDate,
        _CustomerVH
}

```

- Filter for *CustomerId*

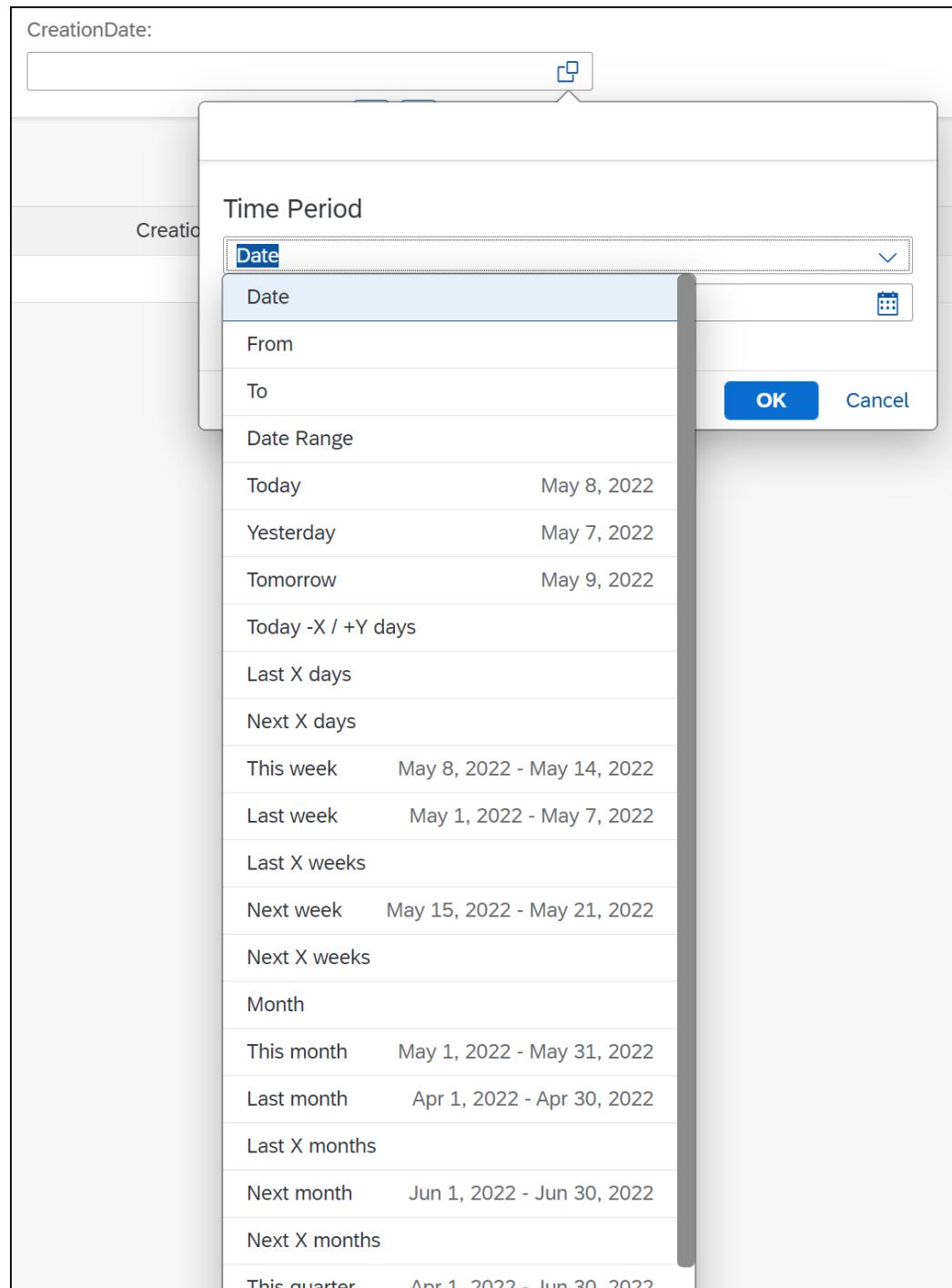


The screenshot shows a search interface for 'Customer'. At the top, there are tabs for 'Search and Select' (which is active) and 'Define Conditions'. Below the tabs is a search bar with a magnifying glass icon and a 'Go' button. To the right of the search bar is a 'Show Filters' link. The main area is titled 'Items' and contains a table with columns: 'Customer Id', 'CompanyName', and 'Country/Region Key'. The table has six rows, each with a checkbox next to the Customer ID. The data is as follows:

Customer Id	CompanyName	Country/Region Key
1	Pixel Tech	LU
2	Visions	LU
3	Techaholic	NL
4	Monotech	NL
5	Techlanch	BE

Below the table, a message says 'No Items or Conditions Selected'. At the bottom right are 'OK' and 'Cancel' buttons.

- Filter for *CreationDate*



Create List Report Object Page

Task 1: Duplicate CDS **ZWS##_CDS_LIST_SEARCH** to **ZWS##_CDS_LIST_OBJECT**

Task 2: Add the new CDS to your *Service Definition*

```
@EndUserText.label: 'UI ## Service Definition'
define service ZUI_WKSP_## {
    expose ZWS##_CDS_Simple as SimpleCDS;
    expose ZWS##_CDS_Basic as BasicCDS;
    expose ZWS##_CDS_LIST as BasicList;
    expose ZWS##_CDS_LIST_SEARCH as BasicSearch;
    expose ZWS##_CDS_LIST_OBJECT as ListObject;
}
```

Task 3: Create a new Fiori elements application with the new CDS

Create a new Fiori Application using the Template Wizard for a *List Report*

Field	Value
Data source	Connect to a System
System	abap-cloud-default_xx-dev (BTP)
Service	ZUI_WKSP_##
Main entity	ListObject
Module name	basic-object
Application title	Basic Object
Application namespace	nato.workshop
Description	Basic Object
Project folder path	/home/user/projects
Add deployment configuration	Yes
Add FLP configuration	Yes
Deployment Target	Cloud Foundry

Field	Value
Destination name	abap-cloud-default_xx(SCP)
Add application to managed application router	Yes
Semantic Object	NATO
Action	BasicObject
Title	Basic List Object
Subtitle	Workshop

- Data Source and Service Selection

Data Source and Service Selection

Configure the data source and select a service.

Data source *

Connect to a System

System  *

abap-cloud-default_abap-trial-d4a3a905trial-dev (BTP)

Service *

ZUI_WKSP_98 > ZUI_WKSP_98 (0001) - OData V4

- Entity Selection

Entity Selection

Configure the selected service.

Main entity *

ListObject

Automatically add table columns to the list page and a section to the object page if none already exists?

Yes No

- Project Attributes

Project Attributes

Configure the main project attributes

Configure the main project attributes.

Module name  *

basic-object

Application title 

Basic Objbect

Application namespace 

nato.workshop

Description 

Basic Objbect

Project folder path *

/home/user/projects

Minimum SAPUI5 version  *

1.96.7 (Source system version)

Add deployment configuration

Yes No

Add FLP configuration

Yes No

Configure advanced options

Yes No

- Deployment Configuration

Deployment Configuration

Configure deployment settings

Please choose the target *

Cloud Foundry

Destination name *

abap-cloud-default_abap-trial-d4a3a905trial-dev(SCP) - https://

Add application to managed application router?

Yes No

- Fiori Launchpad Configuration

Fiori Launchpad Configuration

Configure Fiori Launchpad settings

Semantic Object *

NATO

Action *

BasicObject

Title *

Basic List Object

Subtitle (optional)

Workshop

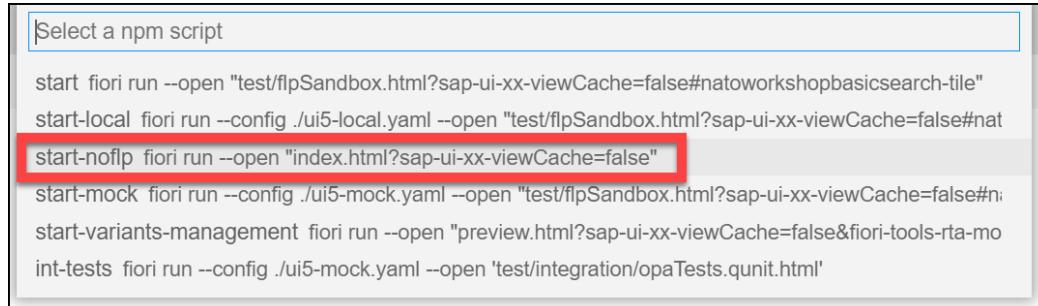
- Preview Application

</> **What you can do**

⌚ **Preview Application**

Choose from start scripts to run the application preview.

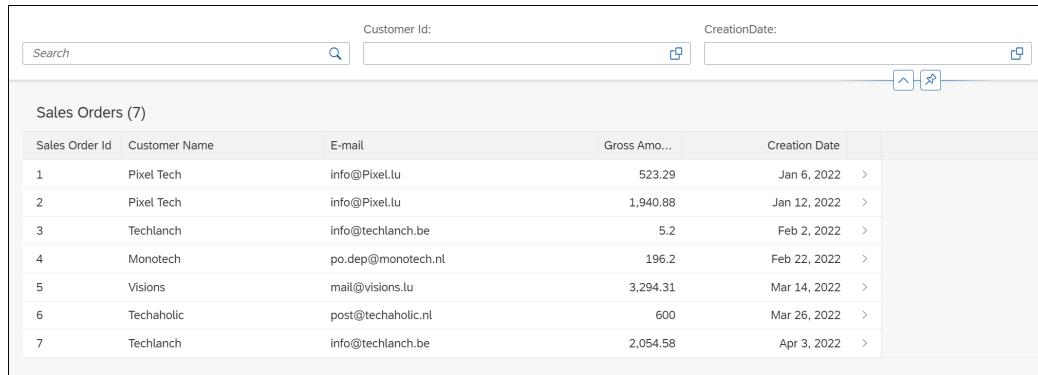
- Select start-noflp, this is the fastest test.



```
>Select a npm script

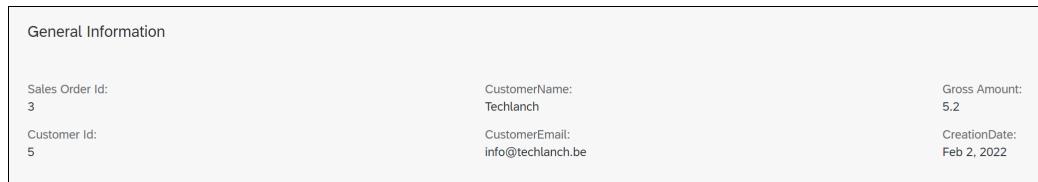
start fiori run --open "test/flpSandbox.html?sap-ui-xx-viewCache=false#natoworkshopbasicsearch-tile"
start-local fiori run --config ./ui5-local.yaml --open "test/flpSandbox.html?sap-ui-xx-viewCache=false#nat
start-noflp fiori run --open "index.html?sap-ui-xx-viewCache=false"
start-mock fiori run --config ./ui5-mock.yaml --open "test/flpSandbox.html?sap-ui-xx-viewCache=false#n:
start-variants-management fiori run --open "preview.html?sap-ui-xx-viewCache=false&fiori-tools-rta-mo
int-tests fiori run --config ./ui5-mock.yaml --open 'test/integration/opaTests.qunit.html'
```

- Result



Sales Order Id	Customer Name	E-mail	Gross Amo...	Creation Date	
1	Pixel Tech	info@Pixel.lu	523.29	Jan 6, 2022	>
2	Pixel Tech	info@Pixel.lu	1,940.88	Jan 12, 2022	>
3	Techlanch	info@techlanch.be	5.2	Feb 2, 2022	>
4	Monotech	po.dep@monotech.nl	196.2	Feb 22, 2022	>
5	Visions	mail@visions.lu	3,294.31	Mar 14, 2022	>
6	Techaholic	post@techaholic.nl	600	Mar 26, 2022	>
7	Techlanch	info@techlanch.be	2,054.58	Apr 3, 2022	>

- Select a line



General Information			
Sales Order Id:	3	CustomerName:	Techlanch
Customer Id:	5	CustomerEmail:	info@techlanch.be
		Gross Amount:	5.2
		CreationDate:	Feb 2, 2022

Task 4: Basic Object Page

- Set the title of every single sales order to the value of field **SalesOrderID**.
- Use *@UI.identification* and *@UI.facet* to add more fields to the object page

Position	Value
10	SalesOrderID
20	CustomerID
30	CustomerName
40	CustomerEmail

```
@AbapCatalog.sqlViewName: 'ZWK##CDSLISTOBJT'
@AbapCatalog.compiler.compareFilter: true
@AbapCatalog.preserveKey: true
@AccessControl.authorizationCheck: #CHECK
```

```

@EndUserText.label: 'Object Page'
@UI.headerInfo.typeName: 'Sales Order'
@UI.headerInfo.typeNamePlural: 'Sales Orders'
@UI.headerInfo.title.value: 'SalesOrderID'
@Search.searchable: true
define view ZWS##_CDS_LIST_OBJECT
    as select from ztmcds9_i_so
        association [1] to ztmcds9_c_vh_cust as _CustomerVH on
$projection.CustomerID = _CustomerVH.SoldToParty
{

    @UI.facet: [
        {
            id: 'COLLFAC1',
            label: 'General Information',
            type: #COLLECTION,
            position: 10
        },
        {
            label: 'General Information',
            type: #IDENTIFICATION_REFERENCE,
            parentId: 'COLLFAC1',
            position: 10
        }
    ]

    @UI.lineItem: [{position: 10 }]
    @UI.identification: [{position: 10}]
    key Id
    as SalesOrderID,
        @UI.selectionField: [{position: 10}]
        @UI.identification: [{position: 20}]
        @Consumption.valueHelpDefinition: [{ entity:{ name:
'ZTMCD9_C_VH_CUST', element: 'SoldToParty' }}]
            CustomerId
    as CustomerID,
        @UI.lineItem: [{position: 20 },{ label: 'Customer Name' }]
        @UI.identification: [{position: 30}]
        @Search.defaultSearchElement: true
        @Search.fuzzinessThreshold : 0.6
            _Customer.name
    as CustomerName,
        @UI.lineItem: [{position: 30 },{ label: 'E-mail' }]
        @UI.identification: [{position: 40}]
        @Search.defaultSearchElement: true
            _Customer.email
    as CustomerEmail,
        @UI.lineItem: [{position: 40 }]

```

```

        GrossAmount,
        NetAmount,
        TaxAmount,
        DeliveryStatus,
        BillingStatus,
        @UI.lineItem: [{position: 50},{ label: 'Creation Date' }]
        @Consumption.filter.selectionType: #INTERVAL
        cast( left(cast(CreationDateTime as abap.char( 30)),8) as
abap.dats) as CreationDate,
        CreationUser,
        @UI.hidden: true
        CurrencyCode,
        _CustomerVH
    }
}

```

- Result

5			
General Information			
General Information			
Sales Order Id: 5	Customer Id: 2	Visions	mail@visions.lu

Task 5: Add Header Facets

- Create two header facets: *AMOUNT* and *STATUS*

Group	Position	Field
AMOUNT	10	GrossAmount
AMOUNT	20	NetAmount
AMOUNT	30	TaxAmount
STATUS	10	DeliveryStatus
STATUS	20	BillingStatus

```

@AbapCatalog.sqlViewName: 'ZWK##CDSLISTOBJT'
@AbapCatalog.compiler.compareFilter: true
@AbapCatalog.preserveKey: true
@AccessControl.authorizationCheck: #CHECK
@EndUserText.label: 'Object Page'
@UI.headerInfo.typeName: 'Sales Order'
@UI.headerInfo.typeNamePlural: 'Sales Orders'
@UI.headerInfo.title.value: 'SalesOrderID'

```

```

@Search.searchable: true
define view ZWS##_CDS_LIST_OBJECT
  as select from ztmcds9_i_so
    association [1] to ztmcds9_c_vh_cust as _CustomerVH on
$projection.CustomerID = _CustomerVH.SoldToParty
{



  @UI.facet: [
  {
    id: 'COLLFAC1',
    label: 'General Information',
    type: #COLLECTION,
    position: 10
  },
  {
    label: 'General Information',
    type: #IDENTIFICATION_REFERENCE,
    parentId: 'COLLFAC1',
    position: 10
  },
  {
    label: 'Amount Information',
    purpose: #HEADER,
    type: #FIELDGROUP_REFERENCE,
    targetQualifier: 'AMOUNT',
    position: 10
  },
  {
    label: 'Status Information',
    purpose: #HEADER,
    type: #FIELDGROUP_REFERENCE,
    targetQualifier: 'STATUS',
    position: 10
  }
]

  @UI.lineItem: [{position: 10 }]
  @UI.identification: [{position: 10}]
  key Id
  as SalesOrderID,
    @UI.selectionField: [{position: 10}]
    @UI.identification: [{position: 20}]
    @Consumption.valueHelpDefinition: [{ entity:{ name:
'ZTMCD9_C_VH_CUST', element: 'SoldToParty' }}]
      CustomerId
  as CustomerID,
    @UI.lineItem: [{position: 20 },{ label: 'Customer Name' }]
    @UI.identification: [{position: 30}]

```

```

        @Search.defaultSearchElement: true
        @Search.fuzzinessThreshold : 0.6
        _Customer.name
    as CustomerName,
        @UI.lineItem: [{position: 30 },{label: 'E-mail' }]
        @UI.identification: [{position: 40}]
        @Search.defaultSearchElement: true
        _Customer.email
    as CustomerEmail,
        @UI.lineItem: [{position: 40 }]
        @UI.fieldGroup: [{position: 10, qualifier: 'AMOUNT'}]
        GrossAmount,
        @UI.fieldGroup: [{position: 20, qualifier: 'AMOUNT'}]
        NetAmount,
        @UI.fieldGroup: [{position: 30, qualifier: 'AMOUNT'}]
        TaxAmount,
        @UI.fieldGroup: [{position: 10, qualifier: 'STATUS'}]
        DeliveryStatus,
        @UI.fieldGroup: [{position: 20, qualifier: 'STATUS'}]
        BillingStatus,
        @UI.lineItem: [{position: 50},{ label: 'Creation Date' }]
        @Consumption.filter.selectionType: #INTERVAL
        cast( left(cast(CreationDateTime as abap.char( 30)),8) as
abap.dats) as CreationDate,
        CreationUser,
        @UI.hidden: true
        CurrencyCode,
        _CustomerVH
    }
}

```

- Result

Amount Information		Status Information	
Gross Amount: 3,294.31		Delivery Status: D	
Net Amount: 2,749.35		Billing Status: P	
Tax Amount: 544.96			

General Information

Sales Order Id:	Customer Id:	Visions	mail@visions.lu
5	2		

Task 6: Add Sections

- Create a collection facet for *General information*.
- Create a collection facet for *More information*.
- Assign qualifiers **AMOUNT** and **STATUS** to collection *More information*.

```
@AbapCatalog.sqlViewName: 'ZWK##CDSLISTOBJT'
@AbapCatalog.compiler.compareFilter: true
@AbapCatalog.preserveKey: true
@AccessControl.authorizationCheck: #CHECK
@EndUserText.label: 'Object Page'
@UI.headerInfo.typeName: 'Sales Order'
@UI.headerInfo.typeNamePlural: 'Sales Orders'
@UI.headerInfo.title.value: 'SalesOrderID'
@Search.searchable: true
define view ZWS##_CDS_LIST_OBJECT
    as select from ztmcds9_i_so
        association [1] to ztmcds9_c_vh_cust as _CustomerVH on
$projection.CustomerID = _CustomerVH.SoldToParty
{



@UI.facet: [
{
    id: 'COLLFACT1',
    label: 'General Information',
    type: #COLLECTION,
    position: 10
},
{
    id: 'COLLFACT2',
    label: 'More Information',
    type: #COLLECTION,
    position: 20
},
{
    label: 'General Information',
    type: #IDENTIFICATION_REFERENCE,
    parentId: 'COLLFACT1',
    position: 10
},
{
    label: 'Amount Information',
    parentId: 'COLLFACT2',
    type: #FIELDGROUP_REFERENCE,
    targetQualifier: 'AMOUNT',
    position: 10
},
{
    label: 'Status Information',
    parentId: 'COLLFACT2',
    type: #FIELDGROUP_REFERENCE,
    targetQualifier: 'STATUS',
    position: 20
}
```

```

    }
]

    @UI.lineItem: [{position: 10 }]
    @UI.identification: [{position: 10}]
key Id
as SalesOrderID,
    @UI.selectionField: [{position: 10}]
    @UI.identification: [{position: 20}]
    @Consumption.valueHelpDefinition: [{ entity:{ name:
'ZTMCDSD9_C_VH_CUST', element: 'SoldToParty' }}]
    CustomerId
as CustomerID,
    @UI.lineItem: [{position: 20 },{ label: 'Customer Name' }]
    @UI.identification: [{position: 30}]
    @Search.defaultSearchElement: true
    @Search.fuzzinessThreshold : 0.6
    _Customer.name
as CustomerName,
    @UI.lineItem: [{position: 30 },{ label: 'E-mail' }]
    @Search.defaultSearchElement: true
    _Customer.email
as CustomerEmail,
    @UI.lineItem: [{position: 40 }]
    @UI.identification: [{position: 40}]
    @UI.fieldGroup: [{position: 10, qualifier: 'AMOUNT'}]
    GrossAmount,
    @UI.identification: [{position: 50}]
    @UI.fieldGroup: [{position: 20, qualifier: 'AMOUNT'}]
    NetAmount,
    @UI.identification: [{position: 60}]
    @UI.fieldGroup: [{position: 30, qualifier: 'AMOUNT'}]
    TaxAmount,
    @UI.fieldGroup: [{position: 10, qualifier: 'STATUS'}]
    DeliveryStatus,
    @UI.fieldGroup: [{position: 20, qualifier: 'STATUS'}]
    BillingStatus,
    @UI.lineItem: [{position: 50},{ label: 'Creation Date' }]
    @UI.selectionField: [{position: 20}]
    @Consumption.filter.selectionType: #INTERVAL
    cast( left(cast(CreationDateTime as abap.char( 30)),8) as
abap.dats) as CreationDate,
    CreationUser,
    @UI.hidden: true
    CurrencyCode,
    _CustomerVH
}

```

- Result [Result](#)

Task 7: Move Annotations to Metadata Extension (Optional)

- Duplicate **ZWS##_CDS_LIST_OBJECT** to **ZWS##_CDS_C_LO**
- Add annotation `@Metadata.allowExtensions: true`
- Create a *Metadata Extension* **ZWS##_ME_CDS_C_LO**
- Move all UI annotations to the *Metadata Extension*
- Add `ZWS##_CDS_C_LO**` to your *Service Definition*
- Create a new Fiori Application using the Template Wizard for a *List Report*

```

@AbapCatalog.sqlViewName: 'ZWK98CDSCLISTOBJ'
@AbapCatalog.compiler.compareFilter: true
@AbapCatalog.preserveKey: true
@AccessControl.authorizationCheck: #CHECK
@EndUserText.label: 'Object Page'

@Metadata.allowExtensions: true
define view ZWS98_CDS_C_LO
  as select from ztmcds9_i_so
    association [1] to ztmcds9_c_vh_cust as _CustomerVH on
$projection.CustomerID = _CustomerVH.SoldToParty
{
  key Id
  as SalesOrderID,
      CustomerId
  as CustomerID,
      _Customer.name
  as CustomerName,
      _Customer.email
  as CustomerEmail,
      GrossAmount,
      NetAmount,
      TaxAmount,
      DeliveryStatus,
      BillingStatus,
      cast( left(cast(CreationDateTime as abap.char( 30)),8) as
abap.dats) as CreationDate,
      CreationUser,
      CurrencyCode,
      _CustomerVH
}

```

```
@Metadata.layer: #CUSTOMER
@UI.headerInfo.typeName: 'Sales Order'
@UI.headerInfo.typeNamePlural: 'Sales Orders'
@UI.headerInfo.title.value: 'SalesOrderID'
@Search.searchable: true
annotate view ZWS98_CDS_C_LO
    with
{
    @UI.facet: [
        {
            id: 'COLLFAC1',
            label: 'General Information',
            type: #COLLECTION,
            position: 10
        },
        {
            id: 'COLLFAC2',
            label: 'More Information',
            type: #COLLECTION,
            position: 20
        },
        {
            label: 'Basic Information',
            purpose: #HEADER,
            type: #FIELDGROUP_REFERENCE,
            targetQualifier: 'BASIC',
            position: 10
        },
        {
            label: 'General Information',
            type: #IDENTIFICATION_REFERENCE,
            parentId: 'COLLFAC1',
            position: 10
        },
        {
            label: 'Amount Information',
            parentId: 'COLLFAC2',
            type: #FIELDGROUP_REFERENCE,
            targetQualifier: 'AMOUNT',
            position: 10
        },
        {
            label: 'Status Information',
            parentId: 'COLLFAC2',
            type: #FIELDGROUP_REFERENCE,
            targetQualifier: 'STATUS',
            position: 20
        }
    ]
}
```

```
]  
  
    @UI.lineItem: [{position: 10 }]  
    @UI.identification: [{position: 10}]  
    SalesOrderID;  
    @UI.selectionField: [{position: 10}]  
    @UI.identification: [{position: 20}]  
    @Consumption.valueHelpDefinition: [{ entity:{ name:  
'ZTMCDS9_C_VH_CUST', element: 'SoldToParty' }}]  
    CustomerID;  
    @UI.lineItem: [{position: 20 },{ label: 'Customer Name' }]  
    @UI.identification: [{position: 30}]  
    @Search.defaultSearchElement: true  
    @Search.fuzzinessThreshold : 0.6  
    CustomerName;  
    @UI.lineItem: [{position: 30 },{ label: 'E-mail' }]  
    @Search.defaultSearchElement: true  
    CustomerEmail;  
    @UI.lineItem: [{position: 40 }]  
    @UI.identification: [{position: 40}]  
    @UI.fieldGroup: [{position: 10, qualifier: 'AMOUNT'}]  
    GrossAmount;  
    @UI.identification: [{position: 50}]  
    @UI.fieldGroup: [{position: 20, qualifier: 'AMOUNT'}]  
    NetAmount;  
    @UI.identification: [{position: 60}]  
    @UI.fieldGroup: [{position: 30, qualifier: 'AMOUNT'}]  
    TaxAmount;  
    @UI.fieldGroup: [{position: 10, qualifier: 'STATUS'}]  
    DeliveryStatus;  
    @UI.fieldGroup: [{position: 20, qualifier: 'STATUS'}]  
    BillingStatus;  
    @UI.lineItem: [{position: 50},{ label: 'Creation Date' }]  
    @UI.selectionField: [{position: 20}]  
    @Consumption.filter.selectionType: #INTERVAL  
    @UI.fieldGroup: [{position: 20, qualifier: 'BASIC', label:  
'Creation Date'}]  
    CreationDate;  
    @UI.fieldGroup: [{position: 10, qualifier: 'BASIC'}]  
    CreationUser;  
    @UI.hidden: true  
    CurrencyCode;  
  
}
```

```

@EndUserText.label: 'UI ## Service Definition'
define service ZUI_WKSP_## {
    expose ZWS##_CDS_Simple as SimpleCDS;
    expose ZWS##_CDS_Basic as BasicCDS;
    expose ZWS##_CDS_LIST as BasicList;
    expose ZWS##_CDS_LIST_SEARCH as BasicSearch;
    expose ZWS##_CDS_LIST_OBJECT as ListObject;
    expose ZWS98_CDS_C_LO as ListObjectME;
}

```

Field	Value
Data source	Connect to a System
System	abap-cloud-default_xx-dev (BTP)
Service	ZUI_WKSP_##
Main entity	ListObjectME
Module name	basic-object
Application title	Basic List with Object Page using Metadata Extension
Application namespace	nato.workshop
Description	Basic List with Object Page using Metadata Extension
Semantic Object	NATO
Action	BasicObjectME
Title	Basic List Object (ME)
Subtitle	Workshop

- Result

The screenshot displays a software interface for viewing a sales order. At the top left is a large number '5'. Below it is a section titled 'Basic Information' containing 'User Name: Robert' and 'Creation Date: Mar 14, 2022'. To the right of this section are two small icons: a blue square with an upward-pointing arrow and a blue square with a circular arrow. Below the basic information is a navigation bar with 'General Information' underlined in blue and 'More Information'.

General Information

Sales Order Id: 5	Visions Gross Amount: 3,294.31	Net Amount: 2,749.35
Customer Id: 2		Tax Amount: 544.96

More Information

Amount Information	Status Information
Gross Amount: 3,294.31	Tax Amount: 544.96
Net Amount: 2,749.35	Delivery Status: D
	Billing Status: P

External Navigation and Visualiztion

Task 1: New CDS **ZWS##_CDS_NAV_EXT**

- Create a new CDS **ZWS##_CDS_NAV_EXT** based on **ztmdt9_products** with an assosiation to **ztmdt9_suppliers**
- Add fields

Position	Field	Name
10	id	ProductId
20	name	
30	supplier_id	
40	_Supplier.name	SupplierName

- Make the field SupplierName an external link to <https://www.google.com/search?q=SupplierName>.
- Add the new CDS to your *Service Definition* and use the *Preview* option in your *Service Binding* to test your result.

```

@AbapCatalog.sqlViewName: 'ZWS##CDSNAVEXT'
@AbapCatalog.compiler.compareFilter: true
@AbapCatalog.preserveKey: true
@AccessControl.authorizationCheck: #CHECK
@EndUserText.label: 'Navigate to external link'
@UI.headerInfo.typeNamePlural: 'Products'
define view ZWS##_CDS_NAV_EXT
  as select from ztmdt9_products
    association [1] to ztmdt9_suppliers as _Supplier on _Supplier.id
    = ztmdt9_products.supplier_id
  {
    @UI.lineItem: [{ position: 10 }]
    key id
    as ProductId,
      @UI.lineItem: [{ position: 20 }]
      name
    as ProductDescription,
      @UI.lineItem: [{ position: 30 }]
      supplier_id
    as SupplierId,
      @UI.lineItem: [{ position: 40, url: 'SearchURL', type:
#WITH_URL }]
  }

```

```

    _Supplier.name
as SupplierName,
@UI.hidden: true
concat('https://www.google.com/search?q=', _Supplier.name)
as SearchURL
}

```

```

@EndUserText.label: 'UI ## Service Definition'
define service ZUI_WKSP_## {
    expose ZWS##_CDS_Simple as SimpleCDS;
    expose ZWS##_CDS_Basic as BasicCDS;
    expose ZWS##_CDS_LIST as BasicList;
    expose ZWS##_CDS_LIST_SEARCH as BasicSearch;
    expose ZWS##_CDS_LIST_OBJECT as ListObject;
    expose ZWS##_CDS_C_LO as ListObjectME;
    expose ZWS##_CDS_NAV_EXT as ExtNavi;
}

```

- Preview

Service Version Details

View information on selected service version

Service Information

Service URL: /sap/opu/odata4/sap/zui_wksp_98/srvd/sap/zui_wksp_98/0001/

type filter text

Entity Set and Association

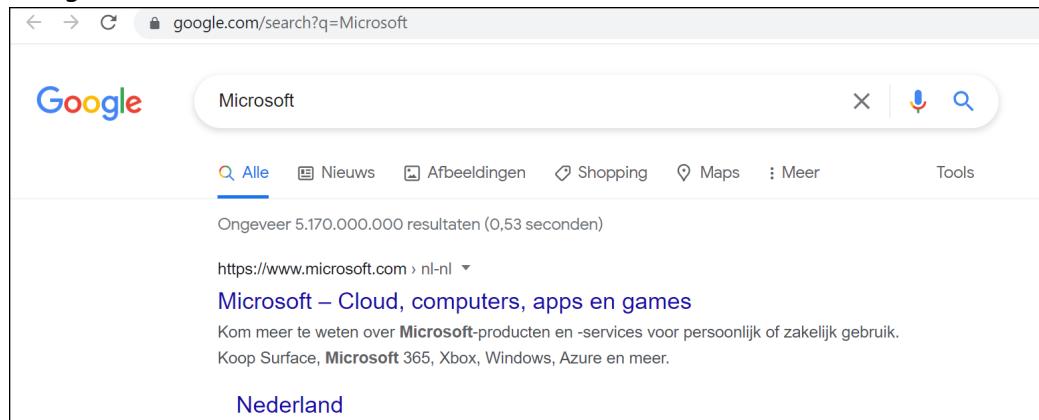
- BasicCDS
- ListObjectME
- BasicList
- ListObject
- BasicSearch
- ExtNavi**
- SimpleCDS

Preview...

- Result

Products (4)				
Product Id	Product Name	Supplier Id	Supplier Name	Action
90001	Galaxy S22	1	Samsung	>
90002	Gmail	3	Google	>
90003	Galaxy Tab 5e	1	Samsung	>
90004	Office 365	2	Microsoft	>

- Navigation



Task 2: Add visualization

- Add the fields *uom* and *stock* to the view
- When the stock is < 30 then color it Red
- When the stock is > 80 then color it Orange
- Else then color green

```

@AbapCatalog.sqlViewName: 'ZWS##CDSNAVEXT'
@AbapCatalog.compiler.compareFilter: true
@AbapCatalog.preserveKey: true
@AccessControl.authorizationCheck: #CHECK
@EndUserText.label: 'Navigate to external link'
@UI.headerInfo.typeNamePlural: 'Products'
define view ZWS##_CDS_NAV_EXT
    as select from ztmdt9_products
        association [1] to ztmdt9_suppliers as _Supplier on _Supplier.id
        = ztmdt9_products.supplier_id
    {
        @UI.lineItem: [{ position: 10 }]
        key id
        as ProductId,
            @UI.lineItem: [{ position: 20 }]
            name
        as ProductDescription,
            @UI.lineItem: [{ position: 30 }]
            supplier_id
        as SupplierId,
            @UI.lineItem: [{ position: 40, url: 'SearchURL', type:
#WITH_URL }]
            _Supplier.name
        as SupplierName,
            @UI.hidden: true
            concat('https://www.google.com/search?q=' , _Supplier.name)
    }
}

```

```
as SearchURL,
    @Semantics.unitOfMeasure: true
    uom,
    @UI.lineItem: [{ position: 50, criticality:
'StockLevelCritical' }]
    stock,
    @UI.hidden: true
    case when stock < 30 then 1
    when stock > 80 then 2
    else 3 end
as StockLevelCritical
}
```

- Result

Products (4)						
Product Id		Supplier Id		Quantity		
90001	Galaxy S22	1	Samsung	✖ 28.00	>	
90002	Gmail	3	Google	✓ 69.00	>	
90003	Galaxy Tab 5e	1	Samsung	✖ 3.00	>	
90004	Office 365	2	Microsoft	⚠ 95.00	>	

Overview Page

Task 1: Create new CDS **ZWS##_CDS_OVP**

- Create a new CDS based on cds *ZTMCD9_I_SO_ITEMS*
- Set annotation *@Metadata.allowExtensions: true*
- Add fields:

Field
Sold as SalesOrder
Id as SalesOrderItem
CustomerName
ProductId
ProductName
CurrencyCode
GrossAmount
NetAmount
'sap-icon://sales-order' as SAPIconUrl

```

@AbapCatalog.sqlViewName: 'ZWK##CDSOVP'
@AbapCatalog.compiler.compareFilter: true
@AbapCatalog.preserveKey: true
@AccessControl.authorizationCheck: #CHECK
@EndUserText.label: 'Overview Page'
@Metadata.allowExtensions: true
define view ZWS##_CDS_OVP
  as select from ZTMCD9_I_SO_ITEMS
{
  key SoId as SalesOrder,
  key Id as SalesOrderItem,
  CustomerName,
  ProductId,
  ProductName,
  CurrencyCode,
  GrossAmount,
  NetAmount,
  'sap-icon://sales-order' as SAPIconUrl
}

```

Task 2: Create Metadata Extension **ZWS##_ME_CDS_OVP**

- Create new Metadata Extension
- Set annotations

Field	Annotation
SalesOrder	lineItem position 10
SalesOrder	selectionField position 10
SalesOrderItem	lineItem position 20
GrossAmount	lineItem position 30

```
@Metadata.layer: #CUSTOMER
annotate view ZWS##_CDS_OVP with
{
    @UI.lineItem: [{ position: 10 }]
    @UI.selectionField: [{ position: 10 }]
    SalesOrder;
    @UI.lineItem: [{ position: 20 }]
    SalesOrderItem;
    @UI.lineItem: [{ position: 30 }]
    GrossAmount;
}
```

Task 3: Add **ZWS##_CDS_OVP** to Service Definition

- Add this new CDS to your *Service Definition* as OverviewPage

```
@EndUserText.label: 'UI ## Service Definition'
define service ZUI_WKSP_## {
    expose ZWS##_CDS_Simple as SimpleCDS;
    expose ZWS##_CDS_Basic as BasicCDS;
    expose ZWS##_CDS_LIST as BasicList;
    expose ZWS##_CDS_LIST_SEARCH as BasicSearch;
    expose ZWS##_CDS_LIST_OBJECT as ListObject;
    expose ZWS##_CDS_C_LO as ListObjectME;
    expose ZWS##_CDS_NAV_EXT as ExtNavi;
    expose ZWS##_CDS_OVP as OverviewPage;
}
```

Task 4: Create Service Binding OData V2 **ZUI_WKSP_##_V2**

- For Overview Pages we need to have a OData V2 service
- Create a new Service Binding with name **ZUI_WKSP_##_V2**
- Use *Binding Type* **OData V2 - UI**

New Service Binding

Service Binding

Create Service Binding

Project: * TRL_EN_1

Package: * Z_WKSP_98

Add to favorite packages

Name: * ZUI_WKSP_98_V2

Description: * UI 98 Service Binding V2

Original Language: EN

Binding Type: * **OData V2 - UI**

Service Definition: * ZUI_WKSP_98

- Publish the new Service Binding

Service Binding: ZUI_WKSP_98_V2

General Information
This section describes general information about this service binding
Binding Type: OData V2 - UI

Service Versions
Define service versions associated with the service binding
type filter text

Version	Service Definition
1.00	ZUI_WKSP_98

Add... **Remove**

Service Version Details
View information on selected service version
Default Authorization Values: 6388A4994D1D90CE93D10B65E9F9C1HT
Local Service Endpoint: Published
Unpublish

Service Information
Service URL: /sap/opu/odata/sap/ZUI_WKSP_98_V2

Preview...

Entity Set and Association
SimpleCDS, BasicCDS, ListObjectME, BasicList, ListObject, BasicSearch, ExtNav, OverviewPage, SimpleCDS

Task 5: Create a new Fiori elements application with the new CDS/OData V2

Create a new Fiori Application using the Template Wizard for a *Overview Page*

Field	Value
Data source	Connect to a System
System	abap-cloud-default_xx-dev (BTP)
Service	ZUI_WKSP_##_V2
Main entity	OverviewPageType
Module name	overview-page
Application title	Overview Page
Application namespace	nato.workshop
Description	Overview Page
Project folder path	/home/user/projects
Add deployment configuration	Yes
Add FLP configuration	Yes
Deployment Target	Cloud Foundry
Destination name	abap-cloud-default_xx(SCP)
Add application to managed application router	Yes
Semantic Object	NATO
Action	OverviewPage
Title	Overview Page
Subtitle	Workshop

- Data Source and Service Selection

Data Source and Service Selection

Configure the data source and select a service.

Data source *

Connect to a System

System ⓘ *

abap-cloud-default_abap-trial-d4a3a905trial-dev (BTP)

Service *

ZUI_WKSP_98_V2 (1) - OData V2

- Entity Selection

Entity Selection

Configure the selected service.

Filter entity *

OverviewPageType

- Project Attributes

Project Attributes

Configure the main project attributes.

Module name ⓘ *

overview-page

Overview Page
Application title ?
Overview Page
Application namespace ?
nato.workshop
Description ?
Overview Page
Project folder path *
/home/user/projects
Minimum SAPUI5 version ? *
1.96.7 (Source system version)
Add deployment configuration
<input checked="" type="radio"/> Yes <input type="radio"/> No
Add FLP configuration
<input checked="" type="radio"/> Yes <input type="radio"/> No
Configure advanced options
<input type="radio"/> Yes <input checked="" type="radio"/> No

- Fiori Launchpad Configuration

Fiori Launchpad Configuration

Configure Fiori Launchpad settings

Semantic Object *

NATO

Action *

OverviewPage

Title *

Overview Page

Subtitle (optional)

Workshop

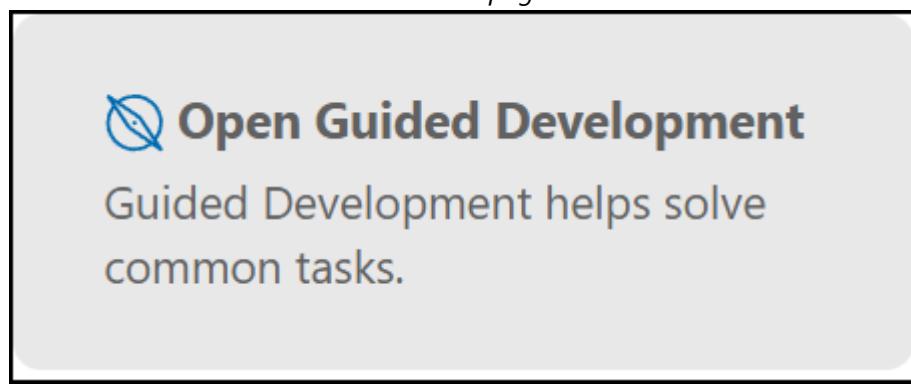
- Open the manifest.json file and add "**useBatch": false**, to the *settings* part of the *mainModel*

SAP Business Application Studio - NatoWorkshop

```
Welcome Application Info – overviewpage manifest.json × i18n.properties
overview-page > webapp > manifest.json > sap.ui5 > models > mainModel > settings > ...
103
104     },
105     "mainModel": {
106         "dataSource": "mainService",
107         "preload": true,
108         "settings": {
109             "defaultBindingMode": "TwoWay",
110             "defaultCountMode": "Inline",
111             "refreshAfterChange": false,
112             "useBatch": false, You, 11 hours ago • Start Overview Page
113             "metadataUriParams": {
114                 "sap-value-list": "none"
115             }
116         },
117     },
118     "@i18n": {
119         "type": "sap.ui.model.resource.ResourceModel",
120         "uri": "i18n/i18n.properties"
121     }
122 },
```

Task 6: Add Table Card

- From the *Application Info* page or via CTRL+SHIFT+P, *Open Guide Development* and select *Add a table card to an overview page



Open Guided Development

Guided Development helps solve common tasks.

- >guided

Fiori: Open Guided Development

Fiori: Open **Guided** Development to the Side
Guided Development

Guided Development - overview-page X

Project overview-p... ▾ | ⏪

Group by Page Type ▾

Overview Page

- Add a custom card to an overview page
- Add a custom filter to the filter bar
- Add a link list card to an overview page
- Add a list card to an overview page
- Add a stack card to an overview page
- Add a static link list card to an overview page
- Add a table card to an overview page ⓘ**
- Add an analytical card to an overview page
- Enable data label in analytical charts
- Enable semantic date range on smart filter bar
- Specify layout for the card container

- Start Guide

Add a table card to an overview page

[About](#) [Step 1](#) [Step 2](#) [Step 3](#) [Step 4](#) [Step 5](#) [Step 6](#)

Description

A table card displays a list of records in a 3-column table layout. A table card embeds the UI5 responsive control `sap.m.Table`. Note following:

- Always limit the width of the table to three columns.
- No row should display more than three lines of text.
- Each table row should be clickable and should navigate to a specific application screen.
- Single and multi-select actions are not supported on a card.

Annotation Term(s) used :

[UI.LineItem](#), [UI.DataPoint](#), [UI.SelectionVariant](#), [UI.Identification](#)

Sample Preview

- Select *Step 5* and set *Model*, *Entity Set* and *Card ID* under bullet **New Card Parameters**

Name	Value
Model	mainModel
Entity Set	OverviewPage
Card ID	Card00

- **New Card Parameters**
- Scroll down to bullet **Table Card Settings Parameters** and enter following settings:

Name	Value
Title	<code>{{card00_title}}</code>

Name	Value
Entity Type	OverviewPageType
Sort By	SalesOrder
Sort Order	Ascending
Tabs	No

- Select **Insert Snippet** and *Exit Guide*

Add a table card to an overview page

Step 5

• Table Card Settings Parameters

Enter a **Title** that will be the title of your newly created card.

Select the **Entity Type** that you will use for sort property.

Select a **SortBy Property**.

Select a **Sort Order**.

Select the **UI.LineItem Annotation Path** to represent data from multiple data instances in a table or a list in your table card.

Select the **UI.DataPoint Annotation Path** to visualize a single point of data in your table card.

Select the **UI.SelectionVariant Annotation Path** to define filters on the entity set of your table card.

Select the **UI.Identification Annotation Path** for your table card.

Title: {{card00_title}} **Entity Type ***: OverviewPageType

Sort By: SalesOrder **Sort Order**: Ascending

Tabs: No **LineItem Qualifier**: Select a lineitem qualifier

DataPoint Qualifier: Select a data point qualifier **Selection Qualifier**: Select a selection variant qualifier

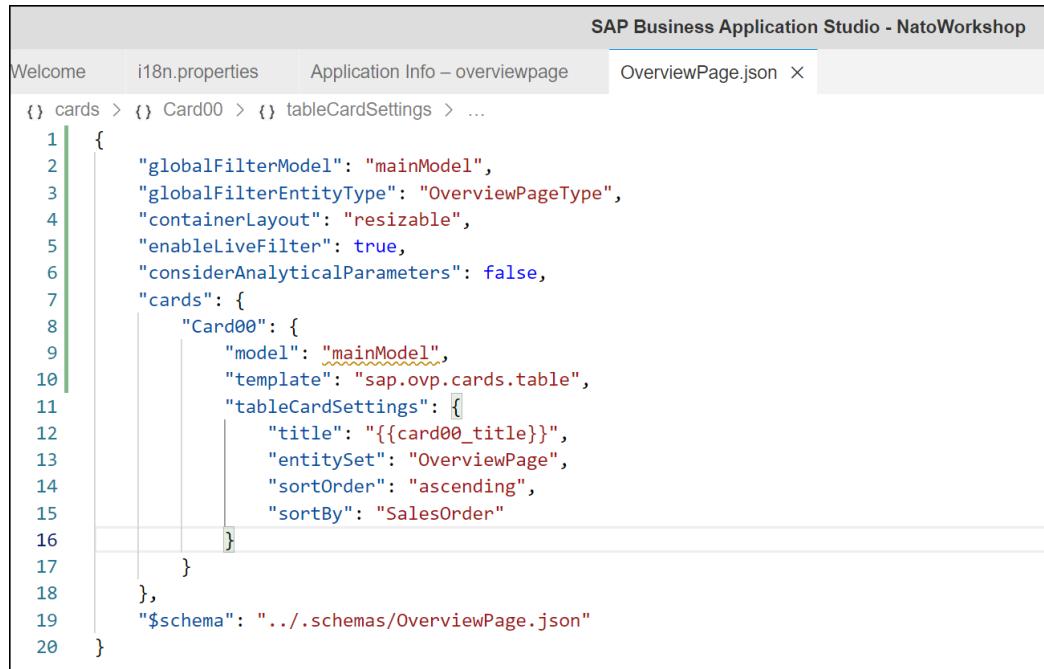
Identification Qualifier: Select an identification qualifier

Insert Snippet **Copy** **Reset**

```
1 "sap.ovp": {
```

Exit Guide

- OverviewPage.json



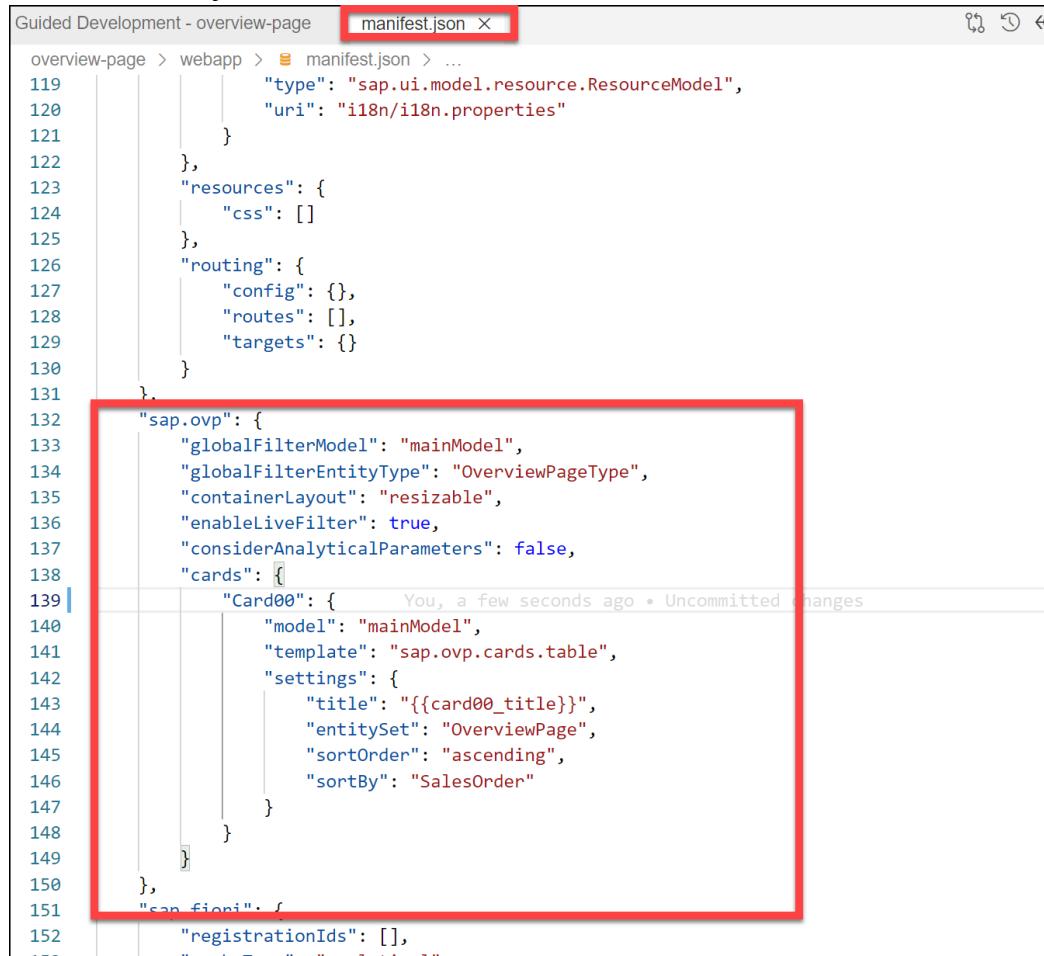
The screenshot shows the SAP Business Application Studio interface with the title bar "SAP Business Application Studio - NatoWorkshop". The tab bar includes "Welcome", "i18n.properties", "Application Info – overviewpage", "OverviewPage.json" (which is the active tab), and "X". Below the tabs, a breadcrumb navigation path shows the structure: "{} cards > {} Card00 > {} tableCardSettings > ...". The main content area displays the JSON code for OverviewPage.json:

```

1  {
2    "globalFilterModel": "mainModel",
3    "globalFilterEntityType": "OverviewPageType",
4    "containerLayout": "resizable",
5    "enableLiveFilter": true,
6    "considerAnalyticalParameters": false,
7    "cards": {
8      "Card00": {
9        "model": "mainModel",
10       "template": "sap.ovp.cards.table",
11       "tableCardSettings": {
12         "title": "{{card00_title}}",
13         "entitySet": "OverviewPage",
14         "sortOrder": "ascending",
15         "sortBy": "SalesOrder"
16       }
17     }
18   },
19   "$schema": "../../schemas/OverviewPage.json"
20 }

```

- Look at *manifest.json*



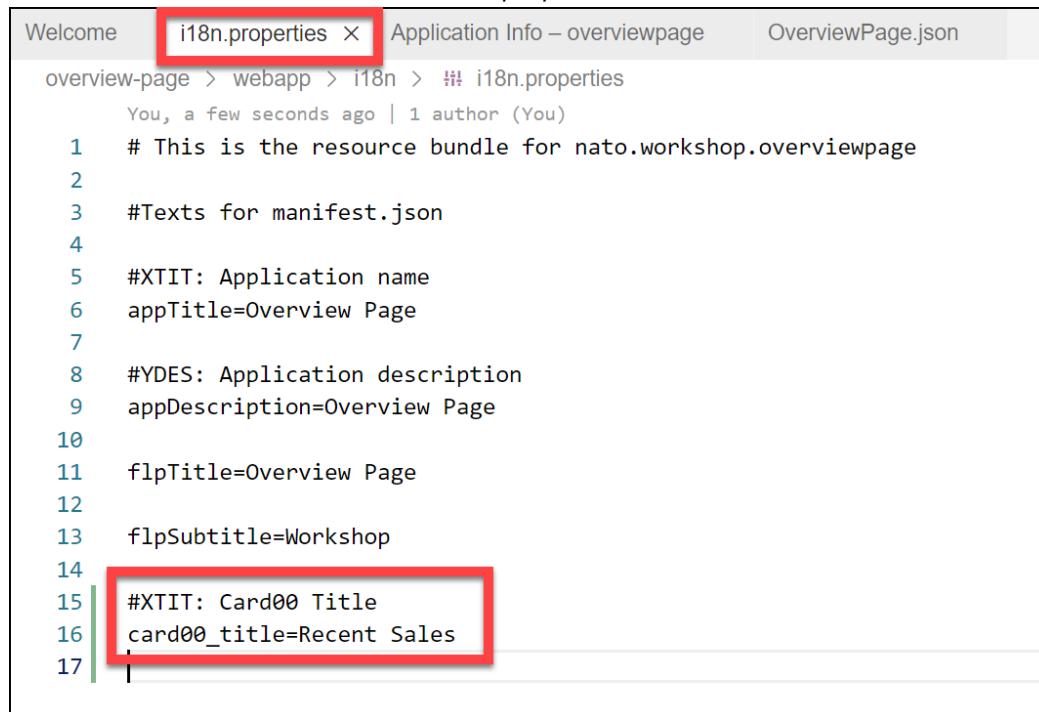
The screenshot shows the SAP Guided Development interface with the title bar "Guided Development - overview-page" and the tab "manifest.json" (highlighted with a red box). The breadcrumb navigation path shows the structure: "overview-page > webapp > manifest.json > ...". The main content area displays the JSON code for manifest.json, with a red box highlighting the "sap.ovp" section:

```

119  {
120   "sap.ui": {
121     "model": "sap.ui.model.resource.ResourceModel",
122     "uri": "i18n/i18n.properties"
123   },
124   "resources": {
125     "css": []
126   },
127   "routing": {
128     "config": {},
129     "routes": [],
130     "targets": {}
131   },
132   "sap.ovp": {
133     "globalFilterModel": "mainModel",
134     "globalFilterEntityType": "OverviewPageType",
135     "containerLayout": "resizable",
136     "enableLiveFilter": true,
137     "considerAnalyticalParameters": false,
138     "cards": {
139       "Card00": {
140         "model": "mainModel",
141         "template": "sap.ovp.cards.table",
142         "settings": {
143           "title": "{{card00_title}}",
144           "entitySet": "OverviewPage",
145           "sortOrder": "ascending",
146           "sortBy": "SalesOrder"
147         }
148       }
149     }
150   },
151   "sap Fiori": {
152     "registrationIds": []
153   }
154 }

```

- Add **card00_title=Recent Sales** to *i18n.properties* file



```
Welcome          i18n.properties X Application Info – overviewpage      OverviewPage.json
overview-page > webapp > i18n > i18n.properties
You, a few seconds ago | 1 author (You)
1 # This is the resource bundle for nato.workshop.overviewpage
2
3 #Texts for manifest.json
4
5 #XTIT: Application name
6 appTitle=Overview Page
7
8 #YDES: Application description
9 appDescription=Overview Page
10
11 flpTitle=Overview Page
12
13 flpSubtitle=Workshop
14
15 #XTIT: Card00 Title
16 card00_title=Recent Sales
17
```

- Now preview the application

The screenshot shows a user interface for a sales application. At the top, there is a dropdown menu labeled "Standard ▾". Below it is a search bar with the placeholder "Sales Order Id:" and a blue search icon. A horizontal line separates this from a table. The table has a header row with columns: "Sales Order Id", "SO Item Id", and "Gross Amount". The data rows are as follows:

Sales Order Id	SO Item Id	Gross Amount
1	10	523
2	10	1K
2	20	600
2	30	294
3	10	5

Task 7: Add Analytical Card

- Add annotations `@UI.chart` to the Metadata Extension

```
@Metadata.layer: #CUSTOMER

@UI.chart: [{
    title: 'Sales by Customer',
    chartType:#DONUT ,
    dimensions: [ 'CustomerName' ],
```

```
measures: [ 'GrossAmount' ],
dimensionAttributes: [
    dimension: 'CustomerName',
    role: #CATEGORY
],
measureAttributes: [
    measure: 'GrossAmount',
    role: #AXIS_1,
    asDataPoint: false
]
}

annotate view ZWS##_CDS_OVP with
{
    @UI.lineItem: [{ position: 10 }]
    @UI.selectionField: [{ position: 10 }]
    SalesOrder;
    @UI.lineItem: [{ position: 20 }]
    SalesOrderItem;
    @UI.lineItem: [{ position: 30 }]
    GrossAmount;
}
```

- Add **card01_title=Sales by Customer** to file *i18n.properties*

- Use *Guided Development* to **Add an analytical card to an overview page**

Guided Development - overview-page X

Project overview-p... | ⚡ | 🔍

Group by Page Type ▾

Overview Page

Add a custom card to an overview page

Add a custom filter to the filter bar

Add a link list card to an overview page

Add a list card to an overview page

Add a stack card to an overview page

Add a static link list card to an overview page

Add a table card to an overview page

Add an analytical card to an overview page (highlighted)

Enable data label in analytical charts

- Start the Guide
- Select *Step 5* and set *Model and Card ID* under bullet **New Card Parameters**

Name	Value
Model	mainModel
Card ID	Card01

Add an analytical card to an overview page

About Step 1 Step 2 Step 3 Step 4 **Step 5**

Enter the card settings in overview page config file

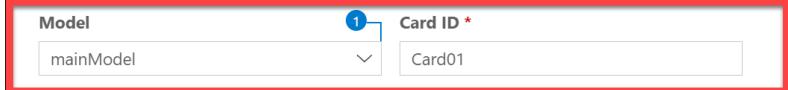
Specify the parameters below and click the Insert Snippet button to update the page config file. Alternatively, locate and expand the Application Modeler view from the Explorer side bar. In this view, search your project and open the page config file within the [Pages](#) folder. Update it to match the following code snippet.

- **New Card Parameters**

Select a [Model](#) name which you would like to use for the analytical card.

Enter a unique [Card ID](#) that can help you identify the newly created analytical card in the overview page.

Select a [Template Type](#) for your new card.



- Scroll down to bullet **Table Card Settings Parameters** and enter following settings:

Name	Value
Entity Set	OverviewPage
Title	{{card01_title}}
Chart Annotation Path	UI.Chart

- Select **Insert Snippet** and **Exit Guide**

• **New Analytical Card Parameters**

Enter a **Title** that will be the title of your newly created card.

Enter a **Subtitle** that will be the subtitle of your newly created card.

Select the **Entity Set** that you will use in your analytical card.

Select the **Presentation Annotation Path** that you want to use as the **UI.PresentationVariant** in your analytical card.

Select the **Chart Annotation Path** that specify the dimensions and measures that make up your analytical card.

Select the **DataPoint Annotation Path** to your **UI.DataPoint** annotation term that you want to use in your analytical card.

Enter the **Value Selection Info** for your analytical card.

Entity Set * **Title ***
OverviewPage {{card01_title}}

Subtitle **Presentation Annotation Path**
Enter a subtitle for the card Select a presentation annotation path

Chart Annotation Path **DataPoint Annotation Path**
UI.Chart Select a datapoint annotation path

Identification Qualifier **Value Selection Info**
Select an identification qualifier Enter a value selection info for the card

Insert Snippet Copy Reset

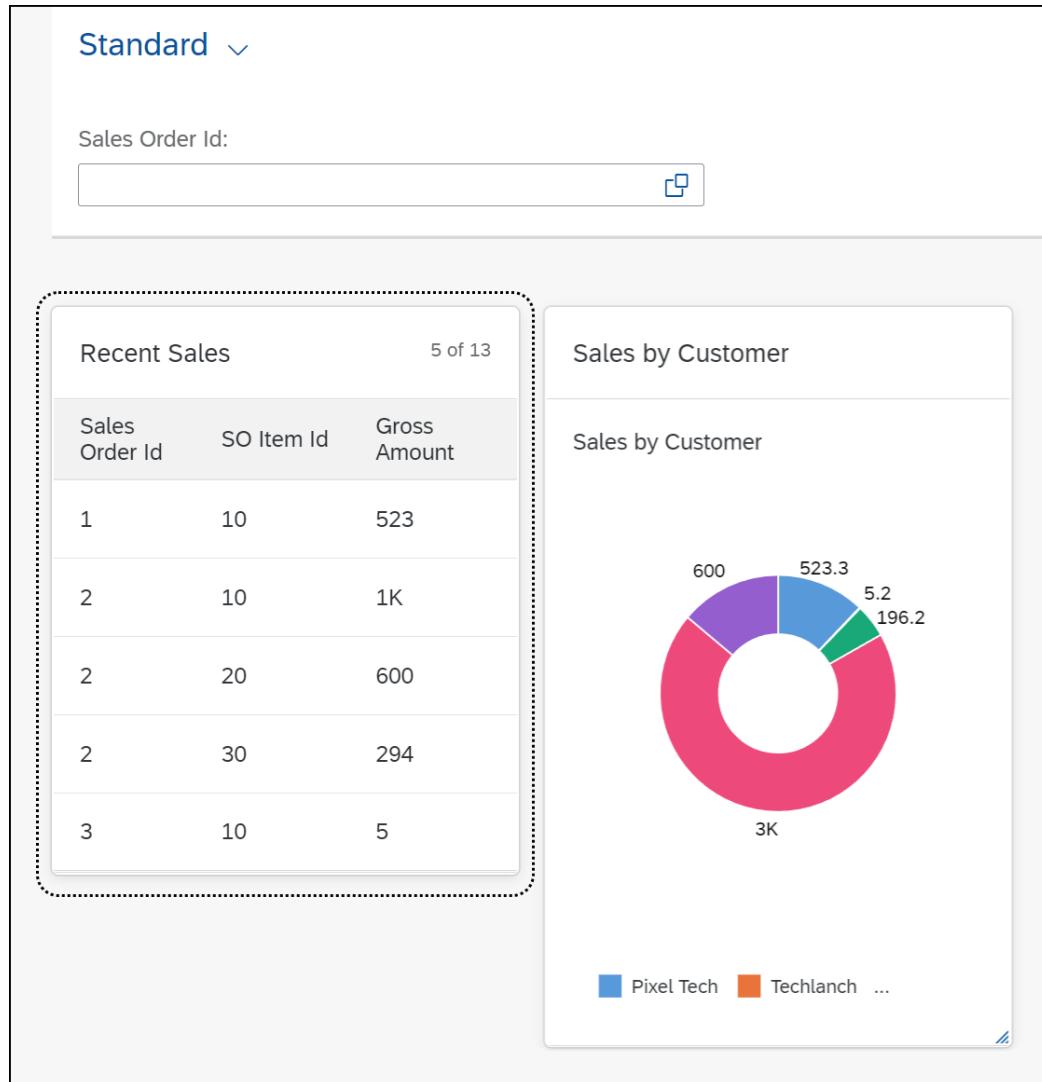
```
1 "sap.ovp": {  
2     "cards": {  
3         "Card01": {  
4             "model": "mainModel",  
5             "template": "sap.ovp.cards.charts.analytical",  
6             "analyticalCardSettings": {}  
7         }  
8     }  
9 }
```

< Back Next > **Exit Guide**

- OverviewPage.json

```
>Welcome i18n.properties OverviewPage.json X Application Info – overviewpage
{} cards > ...
1 {
2   "globalFilterModel": "mainModel",
3   "globalFilterEntityType": "OverviewPageType",
4   "containerLayout": "resizable",
5   "enableLiveFilter": true,
6   "considerAnalyticalParameters": false,
7   "cards": {
8     "Card00": {
9       "model": "mainModel",
10      "template": "sap.ovp.cards.table",
11      "tableCardSettings": {
12        "title": "{{card00_title}}",
13        "entitySet": "OverviewPage",
14        "sortOrder": "ascending",
15        "sortBy": "SalesOrder"
16      }
17    },
18    "Card01": {
19      "model": "mainModel",
20      "template": "sap.ovp.cards.charts.analytical",
21      "analyticalCardSettings": {
22        "title": "{{card01_title}}",
23        "entitySet": "OverviewPage",
24        "chartAnnotationPath": "com.sap.vocabularies.UI.v1.Chart"
25      }
26    }
27  },
28  "$schema": "../../schemas/OverviewPage.json"
29 }
```

- Now preview the application



Task 8: Add Stack Card

- Add annotation `@UI.headerInfo` to the Metadata Extension
- Add annotation `@UI.facet`
- Add annotations `@UI.fieldGroup` with qualifier **DETAILID** for fields **CustomerName** and **ProductName**

```
@Metadata.layer: #CUSTOMER

@UI.chart: [{
    title: 'Sales by Customer',
    chartType:#DONUT ,
    dimensions: [ 'CustomerName' ],
    measures: [ 'GrossAmount' ],
    dimensionAttributes: [{
```

```

        role: #CATEGORY
    }],
    measureAttributes: [
        measure: 'GrossAmount',
        role: #AXIS_1,
        asDataPoint: false
    ]
}

@UI.headerInfo: {
    typeNamePlural: 'Sales Orders',
    typeName: 'Sales Order',
    imageUrl: 'SAPIIconUrl',
    title: {
        label: 'Sales Order ID',
        value: 'SalesOrder'
    },
    description: {
        label: 'Sales Order Item',
        value: 'SalesOrderItem'
    }
}

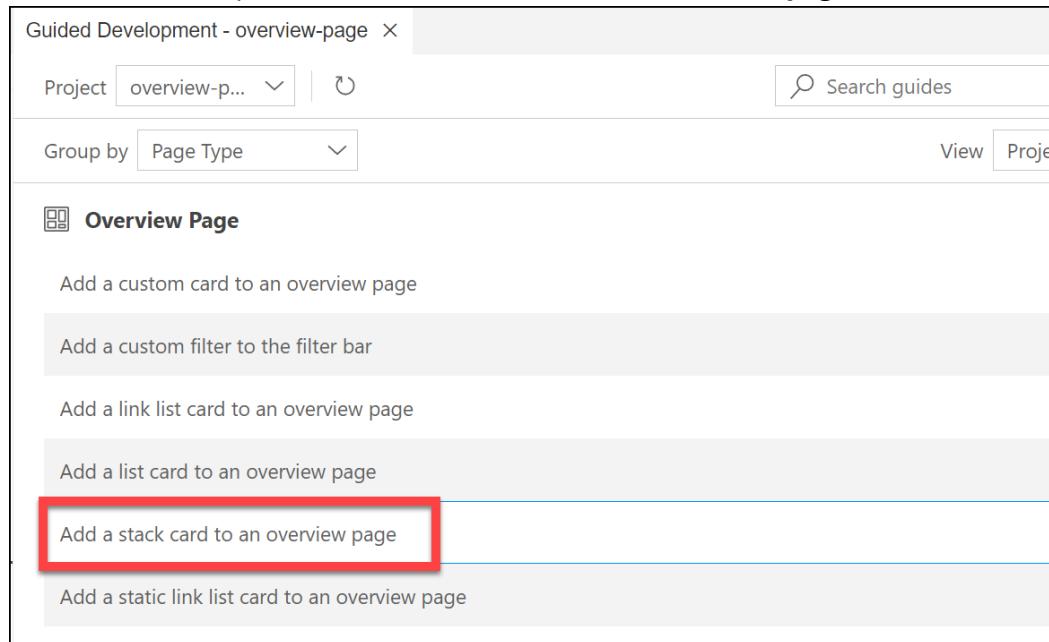
annotate view ZWS##_CDS_OVP with
{

    @UI.facet: [
        type: #FIELDGROUP_REFERENCE,
        targetQualifier: 'DETAILED',
        isSummary: true
    ]

    @UI.lineItem: [{ position: 10 }]
    @UI.selectionField: [{ position: 10 }]
    SalesOrder;
    @UI.lineItem: [{ position: 20 }]
    SalesOrderItem;
    @UI.lineItem: [{ position: 30 }]
    GrossAmount;
    @UI.fieldGroup:[{position: 10, qualifier: 'DETAILED', label:
    'Customer'}]
    CustomerName;
    @UI.fieldGroup:[{position: 20, qualifier: 'DETAILED', label:
    'Product'}]
    ProductName;
}

```

- Add **card02_title=Sales Orders** and **card02_subtitle=Item** to file *i18n.properties*
- Use *Guided Development* to **Add a stack card to an overview page**



- Start the Guide
- Select *Step 5* and set *Model, Entity Set and Card ID* under bullet **New Card Parameters**

Name	Value
Model	mainModel
Entity Set	OverviewPage
Card ID	Card02

Add a stack card to an overview page

Step 1 Step 2 Step 3 Step 4 **Step 5**

Enter card settings in the overview page config file

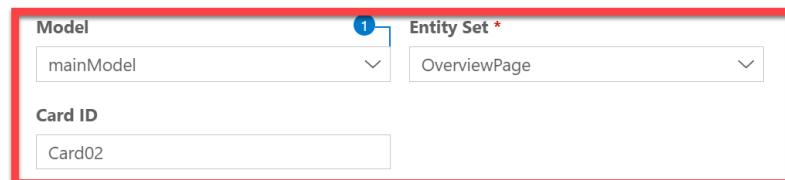
Specify the parameters below and click Insert Snippet to update the page config file. Alternatively, locate and expand the Application Modeler view from the Explorer side bar, search for your project, and open the page config file within the **Pages** folder to make the changes manually.

• **New Card Parameters**

Select the **Model** name that you would like to use for the stack card.

Enter a unique **Card ID** that will help you identify the newly created stack in the overview page.

Select the **Entity Set** that you will use in your stack card.



- Scroll down to bullet **Stack Card Settings Parameters** and enter following settings:

Name	Value
Title	{{card02_title}}
Subtitle	{{card02_subtitle}}

- Select **Insert Snippet** and **Exit Guide**

• **Stack Card Settings Parameters**

Enter the **Title** that will be the title of your newly created card.

Enter a **Subtitle** that will be the sub heading of your newly created card.

Enter an **App Authorization** which is used to set authorization check at card level

Enter an **Item Text** which represents the user defined string in placeholder card.

Set **showFirstActionInFooter** boolean property which represents the flag to show first action in footer of the Quickview cards.

Title
{{card02_title}}

Subtitle
{{card02_subtitle}}

App Authorization
Select an app authorization

Item Text
Enter an item text

Show Action in Footer
true

• **Stack Card Annotation Settings Parameters**

Select the **UI.Facets** Annotation Path to visualize the detailed information about the record.

Select the **UI.Identification** Annotation Path for your stack card.

Facet Qualifier
Select a facet qualifier

Identification Qualifier
Select an identification qualifier

Insert Snippet **Copy** **Reset**

```

1 "sap.ovp": {
2   "cards": {
}

```

< Back Next > **Exit Guide**

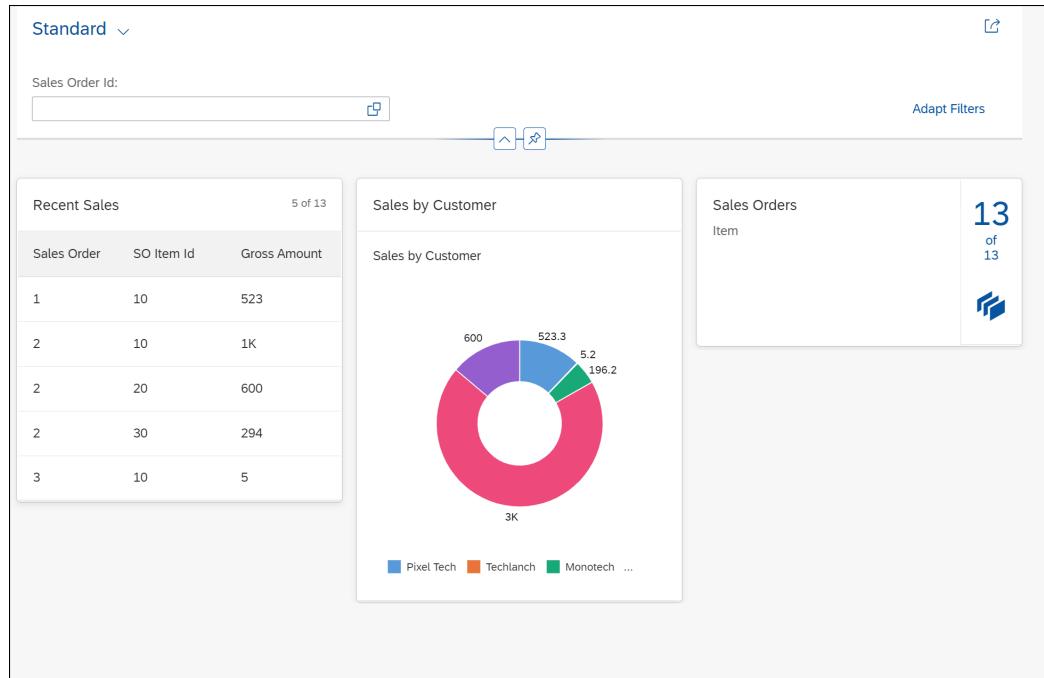
- OverviewPage.json

```

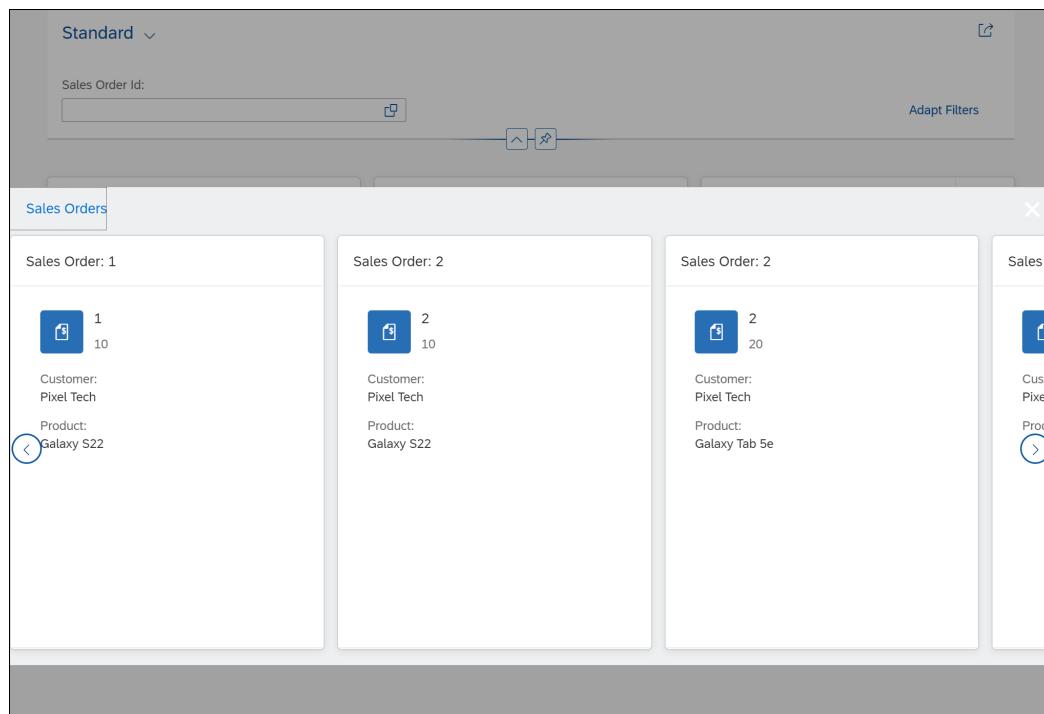
  },
  "Card02": {
    "model": "mainModel",
    "template": "sap.ovp.cards.stack",
    "stackCardSettings": {
      "title": "{{card02_title}}",
      "entitySet": "OverviewPage",
      "subTitle": "{{card02_subtitle}}",
      "objectStreamCardsSettings": {
        "showFirstActionInFooter": "false"
      }
    }
  },
  "icon": "/schemas/OverviewPage.icon"

```

- Now preview the application



- Click on icon of **Stack Card** to show all **Sales Orders**



Analytical List Page

Task 1: Create a new CDS view **ZWS##_CDS_ALP**

- Create a new CDS base on CDS **ztmcds9_c_items_cube**
- Set the annotation `@Metadata.allowExtensions: true`
- Add the following fields

Field	Annotation
Sold	
Id	
CustomerName	<code>@AnalyticsDetails.query.display: #TEXT_KEY</code>
ProductId	
ProductName	
CurrencyCode	
GrossAmount	<code>@Aggregation.default: #SUM</code>
NetAmount	<code>@Aggregation.default: #SUM</code>
Quantity	<code>@Aggregation.default: #SUM</code>

```

@AbapCatalog.sqlViewName: 'ZWS##CDSALP'
@AbapCatalog.compiler.compareFilter: true
@AbapCatalog.preserveKey: true
@AccessControl.authorizationCheck: #CHECK
@EndUserText.label: 'Analytical List Page'
@Metadata.allowExtensions: true
define view ZWS##_CDS_ALP as select from ztmcds9_c_so_items_cube {
    key SoId,
    key Id,
    @AnalyticsDetails.query.display: #TEXT_KEY
    CustomerName,
    ProductId,
    ProductName,
    CurrencyCode,
    @Aggregation.default: #SUM
    GrossAmount,
    @Aggregation.default: #SUM
    NetAmount,
    @Aggregation.default: #SUM

```

```
    Quantity
}
```

Task 2: Create new Metadata Extension **ZWS##_ME_CDS_ALP**

- Add annotation `@UI.lineitem: [{position}]` to all fields
- Make field `Sold` a filter selection field
- Add `@UI.headerInfo` to your CDS
- Add `@UI.selectionPresentationVariant`, `@UI.presentationVariant`, `@UI.chart` and `@UI.selectionVariant` to your CDS

```
@Metadata.layer: #CUSTOMER

@UI.headerInfo: { typeName: 'Sales Order Item',
                  typeNamePlural: 'Sales Order Items',
                  title.value: 'CustomerName',
                  description.value: 'ProductName'
                }

@UI.selectionPresentationVariant: [
  {
    qualifier: 'Default',
    presentationVariantQualifier: 'Default',
    selectionVariantQualifier: 'Default'
  }
]
@UI.presentationVariant: [
  {
    qualifier: 'Default',
    visualizations: [
      {
        type: #AS_CHART,
        qualifier: 'ChartDefault'
      }
    ]
  }
]
@UI.chart: [
  {
    qualifier: 'ChartDefault',
    title: 'Sales by Customer',
    chartType: #COLUMN,
    dimensions: [ 'CustomerName', 'ProductName' ],
    measures: [ 'GrossAmount' ],
    dimensionAttributes: [
      {
        name: 'CustomerName',
        type: #CDS
      }
    ]
  }
]
```

```

        dimension: 'CustomerName',
        role: #CATEGORY
    },
    {
        dimension: 'ProductName',
        role: #CATEGORY
    }
],
measureAttributes: [
    {
        measure: 'GrossAmount',
        role: #AXIS_1,
        asDataPoint: false
    }
]
}
]
@UI.selectionVariant: [
{
    qualifier: 'Default',
    text: 'Default'
}
]

annotate view ZWS##_CDS_ALP
with
{
    @UI.lineItem : [{ position: 10}]
    @UI.selectionField: [{ position: 10 }]
    SoId;
    @UI.lineItem : [{ position: 20}]
    Id;
    @UI.lineItem : [{ position: 30}]
    CustomerName;
    @UI.lineItem : [{ position: 40}]
    ProductId;
    @UI.lineItem : [{ position: 50}]
    ProductName;
    @UI.lineItem : [{ position: 60}]
    CurrencyCode;
    @UI.lineItem : [{ position: 70}]
    GrossAmount;
    @UI.lineItem : [{ position: 80}]
    NetAmount;
    @UI.lineItem : [{ position: 90}]
    Quantity;
}

```

Task 3: Add the new CDS to your *Service Definition*

- Add your CDS as **AnalyticalListPage** to your *Service Definition*

```
@EndUserText.label: 'UI ## Service Definition'
define service ZUI_WKSP_## {
    expose ZWS##_CDS_Simple as SimpleCDS;
    expose ZWS##_CDS_Basic as BasicCDS;
    expose ZWS##_CDS_LIST as BasicList;
    expose ZWS##_CDS_LIST_SEARCH as BasicSearch;
    expose ZWS##_CDS_LIST_OBJECT as ListObject;
    expose ZWS##_CDS_C_LO as ListObjectME;
    expose ZWS##_CDS_NAV_EXT as ExtNavi;
    expose ZWS##_CDS_OVP as OverviewPage;
    expose ZWS##_CDS_ALP as AnalyticalListPage;
}
```

Task 4: Create a new Fiori elements application with the CDS/OData V2

Create a new Fiori Application using the Template Wizard for a *Analytical List Page*

Field	Value
Data source	Connect to a System
System	abap-cloud-default_xx-dev (BTP)
Service	ZUI_WKSP_##_V2
Main entity	AnalyticalListPage
Qualifier	Default
Table type	Analytical
Allow multi select	No
Auto hide	Yes
enable smart variant management	No
Module name	analytic-page
Application title	Analytical List Page
Application namespace	nato.workshop
Description	Analytical List Page
Project folder path	/home/user/projects

Field	Value
Add deployment configuration	Yes
Add FLP configuration	Yes
Deployment Target	Cloud Foundry
Destination name	abap-cloud-default_xx(SCP)
Add application to managed application router	Yes
Semantic Object	NATO
Action	AnalyticPage
Title	Analytical List Page
Subtitle	Workshop

- Data Source and Service Selection

Data Source and Service Selection

Configure the data source and select a service.

Data source *

Connect to a System

System ⓘ *

abap-cloud-default_abap-trial-d4a3a905trial-dev (BTP)

Service *

ZUI_WKSP_98_V2 (1) - OData V2

- Entity Selection

Entity Selection

Configure the selected service.

Main entity *

AnalyticalListPage

Qualifier ⓘ *

Default

Table type ⓘ *

Analytical

Allow multi select ⓘ

Yes No

Auto hide ⓘ

Yes No

Enable smart variant management ⓘ

Yes No

- Project Attributes

Project Attributes

Configure the main project attributes.

Module name ⓘ *

analytic-page

Application title ⓘ

Analytical List Page

Application namespace ⓘ

nato.workshop

Description 

Analytical List Page

Project folder path *

/home/user/projects

Minimum SAPUI5 version  *

1.96.7 (Source system version)

Add deployment configuration

Yes No

Add FLP configuration

Yes No

Configure advanced options

Yes No

- Fiori Launchpad Configuration

Fiori Launchpad Configuration

Configure Fiori Launchpad settings

Semantic Object *

NATO

Action *

AnalyticPage

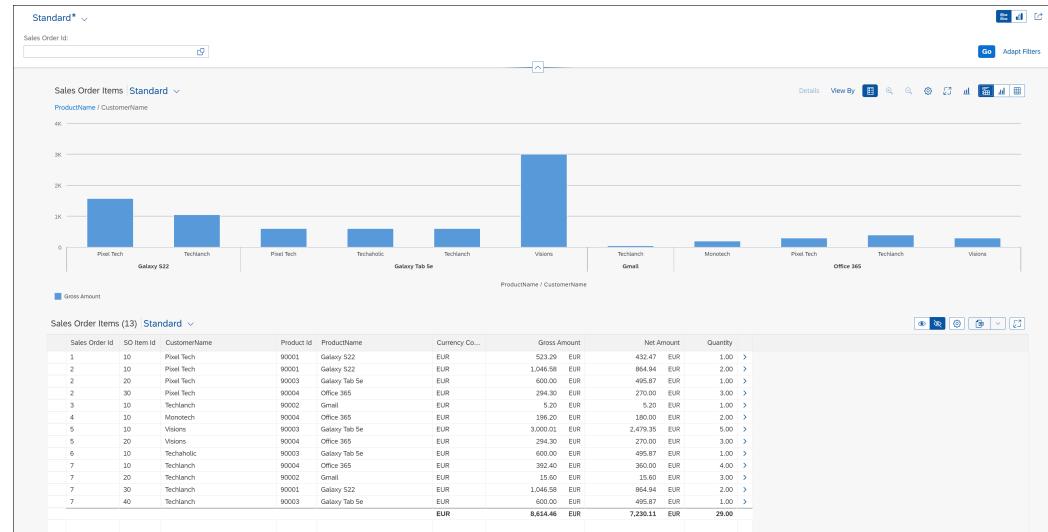
Title *

Analytical List Page

Subtitle (optional)

Workshop

- Preview the Application



Extend Fiori Elements List Report Object Page Application

Task 1: Create a Fiori elements Extension application **

Create a new Fiori Application using the Template Wizard for a *List Report*

Field	Value
Data source	Connect to a System
System	abap-cloud-default_xx-dev (BTP)
Service	ZUI_WKSP_##
Main entity	ListObject
Module name	extend-object
Application title	Extend List Object
Application namespace	nato.workshop
Description	Extend List Object
Project folder path	/home/user/projects
Add deployment configuration	Yes
Add FLP configuration	Yes
Deployment Target	Cloud Foundry
Destination name	abap-cloud-default_xx(SCP)
Add application to managed application router	Yes
Semantic Object	NATO
Action	ExtendList
Title	Extend List Object
Subtitle	Workshop

- Entity Selection

Entity Selection

Configure the selected service.

Main entity *

ListObject

Automatically add table columns to the list page and a section to the object page if none already exists?

Yes No

- Project Attributes

Project Attributes

Configure the main project attributes.

Module name [?](#) *

extend-object

Application title [?](#)

Extend List Object

Application namespace [?](#)

nato.workshop

Description [?](#)

Extend List Object

Project folder path *

/home/user/projects

Minimum SAPUI5 version [?](#) *

1.96.7 (Source system version)

Add deployment configuration

Yes No

Add FLP configuration

Yes No

Configure advanced options

Yes No

- Fiori Launchpad Configuration

Fiori Launchpad Configuration

Configure Fiori Launchpad settings

Semantic Object *

Action *

Title *

Subtitle (optional)

- Preview Application

Standard ▾

Customer Id:					
Search				Go Adapt Filters Enter	
Sales Orders (7)					
Sales Order Id	Customer Name	E-mail	Gross Amo...	Creation Date	
1	Pixel Tech	info@Pixel.lu	523.29	Jan 6, 2022	>
2	Pixel Tech	info@Pixel.lu	1,940.88	Jan 12, 2022	>
3	Techlanch	info@techlanch.be	5.2	Feb 2, 2022	>
4	Monotech	po.dep@monotech.nl	196.2	Feb 22, 2022	>
5	Visions	mail@visions.lu	3,294.31	Mar 14, 2022	>
6	Techaholic	post@techaholic.nl	600	Mar 26, 2022	>
7	Techlanch	info@techlanch.be	2,054.58	Apr 3, 2022	>

Task 2: Add extra column to Table

- Start the *Guided Development* and select **Add and edit table columns**

Guided Development - extend-object x

Project extend-obj... Group by Page Type View Project Guides

Search guides

List Report Page

Add a custom action to a page using extensions

Add a progress indicator column to a table

Add a rating indicator column to a table

Add a smart micro chart to a table

Add and edit filter fields

Add and edit table columns

Add semantic highlights to line items in tables based on their criticality

- Select **ListObjectType** as the *Entity Type*

Add and edit table columns

About
Guide information

Step 1
Add a new record to the
UI.LineItem annotation term

Add a new record to the UI.LineItem annotation term

Complete the form below to update the code snippet with your new or updated table columns.

Once you have finished configured the desired table columns, click the **Insert Snippet** button to add or update the corresponding local annotation file.

Alternatively, copy the code snippet and update the entity's **UI.LineItem** annotation term in the [webapp/annotations folder](#).

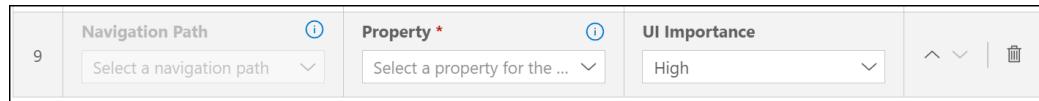
- Entity Type Parameters**

Select the **Entity Type** for which you would like to configure **UI.LineItem** to add or update the table columns.

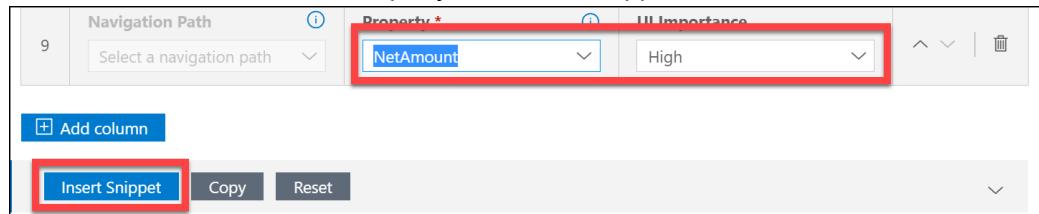
On selection, the code is updated with the **UI.LineItem** annotation if it is defined in the selected entity type.

Entity Type *

- Press button *Add Column*



- Enter **NetAmmount** as the *Property* and *Insert Snippet*



- Preview Application

Sales Order Id	Customer Name	E-mail	Gross Amo...	Creation Date	Net Amount	
1	Pixel Tech	info@Pixel.lu	523.29	Jan 6, 2022	432.47	>
2	Pixel Tech	info@Pixel.lu	1,940.88	Jan 12, 2022	1,630.81	>
3	Techlanch	info@techlanch.be	5.2	Feb 2, 2022	5.2	>
4	Monotech	po.dep@monotech.nl	196.2	Feb 22, 2022	180	>
5	Visions	mail@visions.lu	3,294.31	Mar 14, 2022	2,749.35	>
6	Techaholic	post@techaholic.nl	600	Mar 26, 2022	495.87	>
7	Techlanch	info@techlanch.be	2,054.58	Apr 3, 2022	1,736.41	>

Task 3: Add a custom Action

- Start the *Guided Development* and select **Add a custom action to a page using extensions**

- *Start Guide*
- Select **List Report Page** in *Page*, enter **onCalculate** as *Function Name* and Press *Insert Snippet* followed by *Next*

Add a custom action to a page using extensions

About
Guide information

Step 1
Add a function in controller for the action

Step 2
Update the manifest

Add a function in controller for the action

In this step, you will create a controller extension to define a handler function for the custom action button. If a controller extension is already available for a page, you can copy the code and use it.

- Page Parameters**

Select the **Page** to which you want to attach new custom action.

Page *

- New Function Parameters**

Enter a **Function Name** that will be the name of your new custom function.

Function Name *

- **Insert Snippet** Copy Reset

```

1 sap.ui.define([], 
2   function (){ 
3     "use strict"; 
4     return { 
5       onCalculate: function(oEvent) { 
6         alert('onCalculate'); 
7       } 
8     }; 
9   }; 
10

```

< Back
Next >
Exit Guide

- This will add a new **folder** **custom** and new **file** **ListReportExtController.js** with a template snippet for the javascript code

```
1 | sap.ui.define([],  
2 |   function (){  
3 |     "use strict";  
4 |     return {  
5 |       onCalculate: function(oEvent) {  
6 |         alert('onCalculate');  
7 |       }  
8 |     });  
9 | );  
10|
```

- At the next step set the following parameters, *Insert Snippet* and *Exit Guide*

Name	Value
Entity Set	ListObject
Action Position	Table Toolbar
Action ID	Calculate
Button Text	Calculate Difference
Row Selection	yes

Add a custom action to a page using extensions

About
Guide information

Step 1
Add a function in controller for the action

Step 2
Update the manifest

Update the manifest

In this step, you will add the extension settings of the custom action to the manifest.

• **Entity Set Parameters**

Select the **Entity Set** in which you would like to add your new custom action.

Entity Set *
ListObject

• **New Actions Parameters**

New Actions Parameters

Action Position * Action ID *
Table Toolbar Calculate

Button Text * Row Selection
Calculate Difference yes

Insert Snippet Copy Reset

```
1 {
2   "sap.ui5": {
3     "routing": {
4       "targets": {
5         "ListObjectList": {
6           "options": {
7             "settings": {
8               "controlConfiguration": {
9                 "@com.sap.vocabularies.UI.v1.LineItem": {
10                   "actions": {
11                     "Calculate": {
12                       "id" : "CalculateButton",
13                       "text" : "Calculate Difference",
14                       "press" : "nato.workshop.extendobject.custom.L
15                     }
16                   }
17                 }
18               }
19             }
20           }
21         }
22       }
23     }
24   }
25 }
```

< Back Next > **Exit Guide**

- Manifest.json file is updated

Welcome Application Info – extendobject ListReportExtController.js manifest.json X

tend-object > webapp > manifest.json > sap.ui5 > routing > targets > ListObjectList > options > settings >

```
128     "name": "sap.fe.templates.ListReport",
129     "options": {
130         "settings": {
131             "entitySet": "ListObject",
132             "variantManagement": "Page",
133             "navigation": {
134                 "ListObject": {
135                     "detail": {
136                         "route": "ListObjectObjectPage"
137                     }
138                 }
139             },
140             "controlConfiguration": {
141                 "@com.sap.vocabularies.UI.v1.LineItem": {
142                     "actions": {
143                         "Calculate": {
144                             "id": "CalculateButton",
145                             "text": "Calculate Difference",
146                             "press": "nato.workshop.extendobject.custom.ListRe
147                             "requiresSelection": true
148                         }
149                     }
150                 }
151             }
152         }
153     }
```

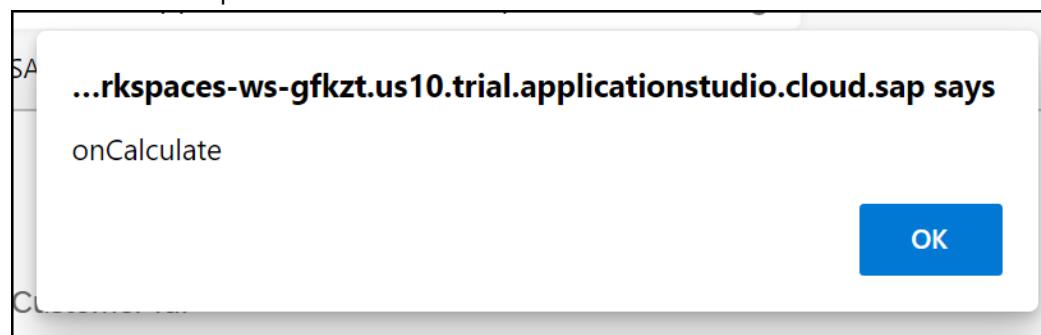
You, 6 minutes ago • Extend List Report

- Preview Application

Standard ▾

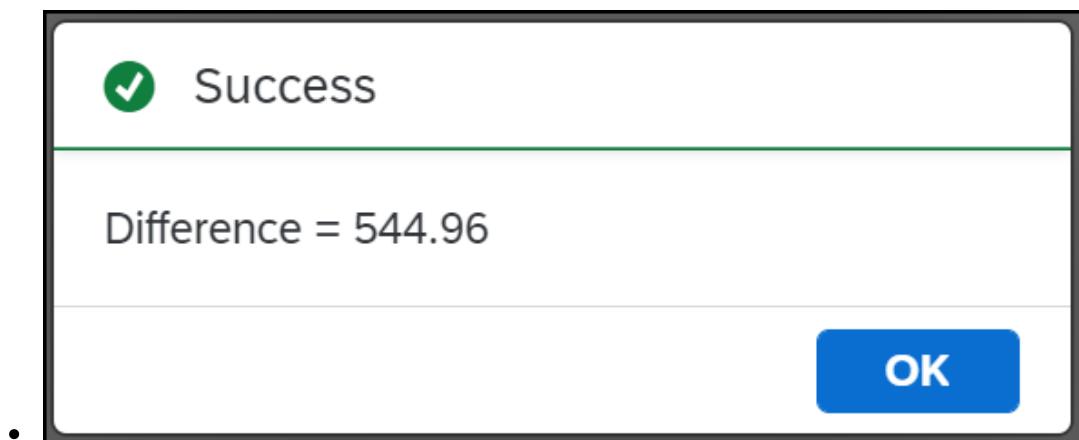
Sales Orders (7)							Calculate Difference
	Sales Order Id	Customer Name	E-mail	Gross Amo...	Creation Date	Net Amount	
<input type="checkbox"/>	1	Pixel Tech	info@Pixel.lu	523.29	Jan 6, 2022	432.47	>
<input type="checkbox"/>	2	Pixel Tech	info@Pixel.lu	1,940.88	Jan 12, 2022	1,630.81	>
<input type="checkbox"/>	3	Techlanch	info@techlanch.be	5.2	Feb 2, 2022	5.2	>
<input type="checkbox"/>	4	Monotech	po.dep@monotech.nl	196.2	Feb 22, 2022	180	>
<input checked="" type="checkbox"/>	5	Visions	mail@visions.lu	3,294.31	Mar 14, 2022	2,749.35	>
<input type="checkbox"/>	6	Techaholic	post@techaholic.nl	600	Mar 26, 2022	495.87	>
<input type="checkbox"/>	7	Techlanch	info@techlanch.be	2,054.58	Apr 3, 2022	1,736.41	>

- Select a line and press the **Calculate Difference** button



- Now implement the javascript template in the new **ListReportExtController.js**, Calculate the difference between de *GrossAmount* and *NetAmount* and show it using a MessageBox

```
sap.ui.define([
    "sap/m/MessageBox"
],
function (MessageBox){
    "use strict";
    return {
        onCalculate: function(oEvent) {
            let nGrossAmount =
parseFloat(this.getSelectedContexts()[0].getObject().GrossAmount);
            let nNetAmount = parseFloat(this.getSelectedContexts()
[0].getObject().NetAmount);
            let nDiff = nGrossAmount - nNetAmount;
            MessageBox.success("Difference = " + nDiff);
        }
    };
});
```



Extend Fiori Elements Overview Page Application

Task 1: Create a new Extension application CDS/OData V2

- Create a new Fiori Application using the Template Wizard for a *Overview Page*

Field	Value
Data source	Connect to a System
System	abap-cloud-default_xx-dev (BTP)
Service	ZUI_WKSP_##_V2
Main entity	OverviewPageType
Module name	extend-ovp
Application title	Extend Overview Page
Application namespace	nato.workshop
Description	Extend Overview Page
Project folder path	/home/user/projects
Add deployment configuration	Yes
Add FLP configuration	Yes
Deployment Target	Cloud Foundry
Destination name	abap-cloud-default_xx(SCP)
Add application to managed application router	Yes
Semantic Object	NATO
Action	ExtendOVP
Title	ExtendOVP
Subtitle	Workshop

- Open the manifest.json file and add "**useBatch": false**, to the *settings* part of the *mainModel*

```
90 },
91     "" : {
92         "dataSource": "mainService",
93         "preload": true,
94         "settings": {
95             "useBatch": false,
96             "synchronizationMode": "None",
97             "operationMode": "Server",
98             "autoExpandSelect": true,
99             "earlyRequests": true,
100            "groupId": "$direct"
101        }
102    },
103    "@i18n": {
104        "en": {
105            "label": "Main Model"
106        }
107    }
108 }
```

Task 2: Add a table card

- Start a *Guided Development* for **Add a table card to an overview page**
- In Step 5 use following **New Card Parameters**

Name	Value
Model	mainModel
Entity Set	OverviewPage
Card ID	Card00

Add a table card to an overview page

Step 1 Step 2 Step 3 Step 4 Step 5 Step 6

Enter card settings in the overview page config file

Specify the parameters below and click Insert Snippet to update the page config file. Alternatively, locate and expand the Application Modeler view from the Explorer side bar. In this view, search your project and open the page config file within the **Pages** folder. Update it to match the following code snippet.

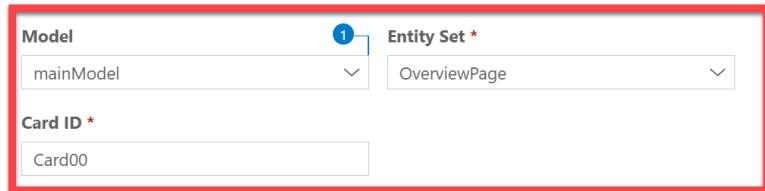
- **New Card Parameters**

Select a **Model** name that you would like to use for the table card.

Enter a unique **Card ID** that will help you identify the newly created table card in the overview page.

Select a **Template Type** for your new card.

Select the **Entity Set** that you will use in your table card.


- Add following **Table Card Settings Paramters**

Name	Value
Title	{{card00_title}}
Entity Type	OverviewPageType
Tabs	No

- *Insert Snippet*

• **Table Card Settings Parameters**

Enter a **Title** that will be the title of your newly created card.

Select the **Entity Type** that you will use for sort property.

Select a **SortBy Property**.

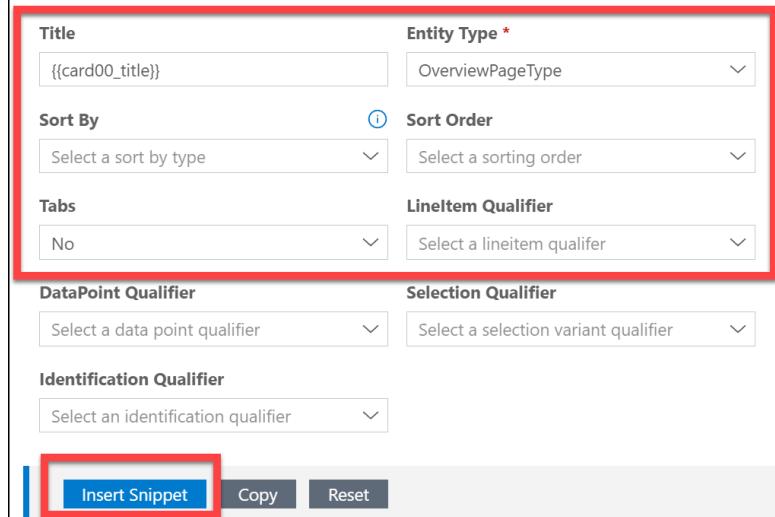
Select a **Sort Order**.

Select the **UI.LineItem Annotation Path** to represent data from multiple data instances in a table or a list in your table card.

Select the **UI.DataPoint Annotation Path** to visualize a single point of data in your table card.

Select the **UI.SelectionVariant Annotation Path** to define filters on the entity set of your table card.

Select the **UI.Identification Annotation Path** for your table card.



Title		Entity Type *	
{{card00_title}}		OverviewPageType	
Sort By		Sort Order	
Select a sort by type		Select a sorting order	
Tabs		LineItem Qualifier	
No		Select a lineitem qualifer	
DataPoint Qualifier		Selection Qualifier	
Select a data point qualifer		Select a selection variant qualifer	
Identification Qualifier			
Select an identification qualifer			
<input type="button" value="Insert Snippet"/> <input type="button" value="Copy"/> <input type="button" value="Reset"/>			

- *OverviewPage.json*



```

1  {
2      "globalFilterModel": "mainModel",
3      "globalFilterEntityType": "OverviewPageType",
4      "containerLayout": "resizable",
5      "enableLiveFilter": true,
6      "considerAnalyticalParameters": false,
7      "cards": {
8          "Card00": {
9              "model": "mainModel",
10             "template": "sap.ovp.cards.table",
11             "tableCardSettings": {
12                 "title": "{{card00_title}}",
13                 "entitySet": "OverviewPage"
14             }
15         },
16         "$schema": "../schemas/OverviewPage.json"
17     }
18 }

```

- Preview Application

The screenshot shows a SAP Fiori application interface. At the top, there is a blue header bar with the text "Standard" and a dropdown arrow. Below the header, a search bar is labeled "Sales Order Id:" with a placeholder "Search" and a magnifying glass icon. A modal dialog box is displayed in the center. The dialog has a title bar with "card00_title" on the left and "5 of 13" on the right. The main content area of the dialog is a table with five rows of data:

Sales Order Id	SO Item Id	Gross Amount
1	10	523
2	10	1K
2	20	600
2	30	294
3	10	5

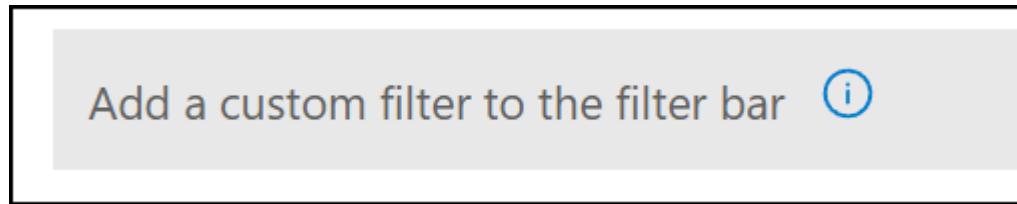
Task 3: Add a custom filter to the filter bar

- Add **card00_title=Recent Sales** **card01_title=Recent Products sold** to *i18n.properties*

```
#XTIT: Card00_title
card00_title=Recent Sales

#XTIT: Card01_title
card01_title=Recent Products sold
```

- Start a *Guided Development* for **Add a custom filter to the filter bar**



- In *Step 1* use following parameters

Name	Value
Fragment File Name	CustomFilter
Custom Filter Key	CustomProductName
Custom Filter Name	Custom Product Name
Control ID	CustomProductName

- Insert Snippet and Next*

Add a custom filter to the filter bar

Step 1
Create a custom filter field

Step 2
Create a controller extension

Step 3
Add extensions to the manifest

Create a custom filter field

In this step, you will create a new fragment as a view extension for a filter bar and add the filter field within the fragment.

• **File Name Parameters**
Input the `filename` that will be used as prefix in the fragment filename.

Fragment File Name *
CustomFilter

• **Custom Filter Parameters**
Input an `unique key` for the custom filter.
Input a `name` for your new custom filter.
Input an `ID` for the control.

Custom Filter key *
CustomProductName

Custom Filter name *
Custom Product Name

Control ID *
CustomProductName

Insert Snippet **Copy** **Reset**

```

1 <core:FragmentDefinition xmlns="sap.m" xmlns:smartfilterbar="sap.ui.comp.smartfilterbar" xmlns:control="sap.ui.core"
2   <smartfilterbar:ControlConfiguration groupId="_BASIC"
3     key="CustomProductName"
4     label="Custom Product Name"
5     visibleInAdvancedArea="true">

```

< Back **Next >** Exit Guide

- This creates a new folder **ext/fragments** with file **CustomFilter.fragment.xml**



The screenshot shows the SAP Studio interface. The Explorer view on the left has 'extend-ovp' selected. The code editor on the right displays the XML content of 'CustomFilter.fragment.xml'. The XML defines a fragment with a smart filter bar configuration.

```
<core:FragmentDefinition xmlns="sap.m" xmlns:smartfilterbar="sap.ui.comp.smartfilterbar" xmlns:ctrl="sap.ui.core.Control">
  <smartfilterbar:ControlConfiguration groupId="_BASIC">
    <key>"CustomProductName"</key>
    <label>"Custom Product Name"</label>
    <visibleInAdvancedArea>"true"</visibleInAdvancedArea>
    <smartfilterbar:customControl>
      <Input id="CustomProductName" type="Text"/>
    </smartfilterbar:customControl>
  </smartfilterbar:ControlConfiguration>
</core:FragmentDefinition>
```

- In *Step 2* use following parameters

Name	Value
Model	mainModel
Entity Type	OverviewPageType
Entity Type Property	ProductName
Custom Filter Property	ProductName

- Insert Snippet and Next

Add a custom filter to the filter bar

About Step 1 Step 2 Step 3

Guide information Create a custom filter field Create a controller extension Add extensions to the manifest

Create a controller extension

In this step, you will create a controller extension to define the methods `getCustomAppStateDataExtension` and `restoreCustomAppStateDataExtension`. These methods handle the storage and retrieval of custom filter input values in the app state.

If a controller extension is already available for a page, you can copy the code and use it.

Entity Type Parameters

Select a **Model** name for the filter.

Select the **Entity Type** you would like to use. You will add data from the selected entity type for your new filter.

Select the **Entity Type Property** you would like to use.

Controller Extension Parameters

Input a property name for the custom filter

Custom Filter Property *

ProductName

Insert Snippet Copy Reset

1 sap.ui.define([

< Back Next > Exit Guide

- This creates a new folder **ext/controller** with file **OverViewPageExt.controller.js**

EXPLORER

... Welcome Application Info – extendop CustomFilter.fragment.xml OverViewPageExt.controller.js ...

OPEN EDITORS

PROJECTS

.vscode analytic-page basic-list basic-object basic-object-me basic-search extend-object

extend-opv node_modules annotations ext controller OverViewPageExt.controller.js

webapp fragments CustomFilter.fragment.xml i18n localService

```
1 sap.ui.define([
2   "sap/ui/model/Filter"
3 ], function (Filter) {
4   "use strict";
5   // controller for custom filter, navigation param, action(quick view and global filter)
6   // controller class name can be like app.opv.ext.CustomFilter where app.opv can be removed
7   return {
8     getCustomFilters: function () {
9       /* This method returns a filter object to the OVP library. If there are multiple filters, they must be clubbed into single Filter object. */
10      var oValue1 = this.oView.byId("CustomProductName").getValue();
11      var aFilters = [], oFilter1;
12      if (oValue1) {
13        oFilter1 = new Filter({
14          path: "ProductName",
15          operator: "EQ",
16          value1: oValue1
17        });
18        aFilters.push(oFilter1);
19      }
20      if (aFilters && aFilters.length > 0) {
21        return (new Filter(aFilters, true));
22      }
23    }
24  }
25});
```

- In Step 3 Insert Snippet and Exit Guide

Add a custom filter to the filter bar

About
Guide information

Step 1
Create a custom filter field

Step 2
Create a controller extension

Step 3
Add extensions to the manifest

Add extensions to the manifest

In this step, you will add the newly created filter fragment and controller extension to the manifest.

Insert Snippet **Copy** **Reset**

```

1  "sap.ui5": {
2      "dependencies": {
3          },
4          "models": {
5          },
6          "extends": {
7              "extensions": {
8                  "sap.ui.controllerExtensions": {
9                      "sap.ovp.app.Main": {
10                         "controllerName": "nato.workshop.extendovp.ext.controller.OverviewPageExt",
11                         }
12                     },
13                     "sap.ui.viewExtensions": {
14                         "sap.ovp.app.Main": {
15                             "SmartFilterBarControlConfigurationExtension|OverviewPageType": {
16                                 "className": "sap.ui.core.Fragment",
17                                 "fragmentName": "nato.workshop.extendovp.ext.fragments.CustomFilter",
18                                 "type": "XML"
19                             }
20                         }
21                     }
22                 }
23             }
24         }
25     }

```

< Back **Next >** **Exit Guide**

- Implement **CustomFilter.fragment.xml** by adding a Select

```

<core:FragmentDefinition xmlns="sap.m"
xmlns:smartfilterbar="sap.ui.comp.smartfilterbar"
xmlns:core="sap.ui.core">
    <smartfilterbar:ControlConfiguration id="CustomFilter"
groupId="_BASIC"
        key="CustomProductName"
        label="Custom Product Name"
        visibleInAdvancedArea="true">
        <smartfilterbar:customControl>
            <Select id="CustomProductName">
                <core:Item id="all" text="All" key="All"/>
                <core:Item id="Gal" text="Galaxy" key="Gal"/>
                <core:Item id="Sub" text="Subscription"
key="Sub"/>
            </Select>
        </smartfilterbar:customControl>

```

```
</smartfilterbar:ControlConfiguration>
</core:FragmentDefinition>
```

- Implement **OverViewPageExt.controller.js**

```
sap.ui.define([
    "sap/ui/model/Filter",
    "sap/ui/model/FilterOperator"
], function (Filter, FilterOperator) {
    "use strict";
    // controller for custom filter, navigation param,
    action(quick view and global filter), navigation target
    // controller class name can be like app.ovp.ext.CustomFilter
    where app.ovp can be replaced with your application namespace
    return {
        getCustomFilters: function () {
            /* This method returns a filter object to the OVP
            library. If there are multiple filters, they should
            be clubbed into single Filter object. */
            var sSelectedKey =
                this.oView.byId("CustomProductName").getSelectedKey();
            var aFilters = [], oFilter1;

            switch (sSelectedKey){
                case "All":
                    break;
                case "Gal":
                    oFilter1 = new Filter({path:"ProductName",
operator: FilterOperator.StartsWith, value1: "Gal"})
                    aFilters.push(oFilter1);
                    break;
                case "Sub":
                    oFilter1 = new Filter({path:"ProductName",
operator: FilterOperator.NotStartsWith, value1: "Gal"})
                    aFilters.push(oFilter1);
                    break;
            }

            if (aFilters && aFilters.length > 0) {
                return (new Filter(aFilters, true));
            }
        },
        getCustomAppStateDataExtension: function (oCustomData) {
            //the content of the custom field will be stored in
            the app state, so that it can be restored later, for example after
            a back navigation.
        }
    }
});
```

```
//The developer has to ensure that the content of the
field is stored in the object that is returned by this method.
    if (oCustomData) {
        var oCustomField1 =
this.oView.byId("CustomProductName");
        if (oCustomField1) {
            oCustomData.ProductName =
oCustomField1.getSelectedKey();
        }
    }
},
restoreCustomAppStateDataExtension: function (oCustomData)
{
    //in order to restore the content of the custom field
    //in the filter bar, for example after a back navigation,
    //an object with the content is handed over to this
    //method. Now the developer has to ensure that the content of the
    //custom filter is set to the control
    if (oCustomData) {
        if (oCustomData.ProductName) {
            var oCustomField1 =
this.oView.byId("CustomProductName");

            oCustomField1.setValue(oCustomData.ProductName);
        }
    }
}
});
```

- Preview Application

Standard* ▾

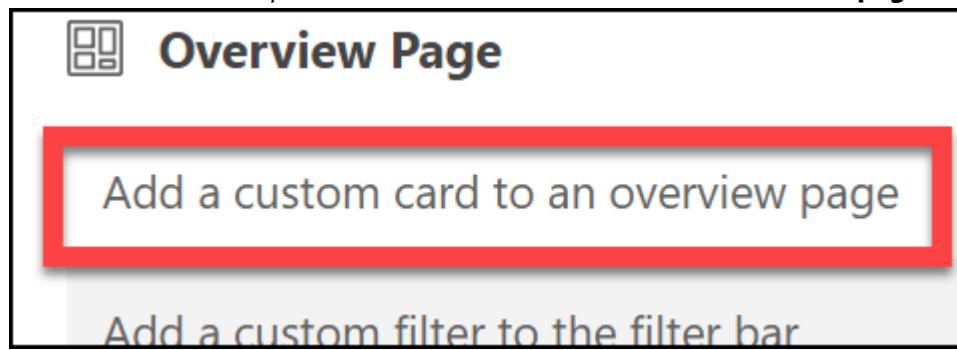
Sales Order Id:

Custom Product Name:

Recent Sales		
Sales Order Id	SO Item Id	Gross Amount
2	30	294
3	10	5
4	10	196
5	20	294
7	10	392
7	20	16

Task 4: Add a custom card

- Start a *Guided Development* for **Add a custom card to an overview page**



- In Step 1 use following parameters

Name	Value
Folder name	customCard
Card ID	card01
Fragment file name prefix	CustomCardContent
Controller file name prefix	CustomCard

*Insert Snippet and Next***Add a custom card to an overview page**

About Step 1 Step 2 Step 3 Step 4

Create a Component.js file

Specify the parameters below and click Insert Snippet to create the required **Component.js** file.

- Folder name Parameters**

Input a **folder name** of custom component.

Input a unique ID for the custom card.

Folder name *

customCard

Card ID *

card01

- File name Parameters**

Input the **fragment file name prefix**.

Input the **controller file name prefix**.

Fragment file name prefix *

CustomCardContent

Controller file name prefix *

CustomCard

Insert Snippet

Copy

Reset

```

1 sap.ui.define(["sap/ovp/cards/custom/Component", "jquery.sap.global"], {
2   function (CardComponent, jQuery) {
3     "use strict";
4
5     return CardComponent.extend("nato.workshop.extendovp.ext.customCard.Component", {
6       // use inline declaration instead of component.json to save 1 round trip
7       metadata: {
8         properties: {
9           contentFragment: {
10             type: "string"
11           }
12         }
13       }
14     });
15   }
16 }
17 
```

< Back

Next >

Exit Guide

- This creates a new folder **customCard** with file **Component.js**



- In Step 2 Insert Snippet and Next

Add a custom card to an overview page

About Step 1 **Step 2** Step 3 Step 4

Create a CustomList.controller.js file

Click Insert Snippet to create the controller file `CustomList.controller.js` in the folder `customList`.

Insert Snippet Copy Reset

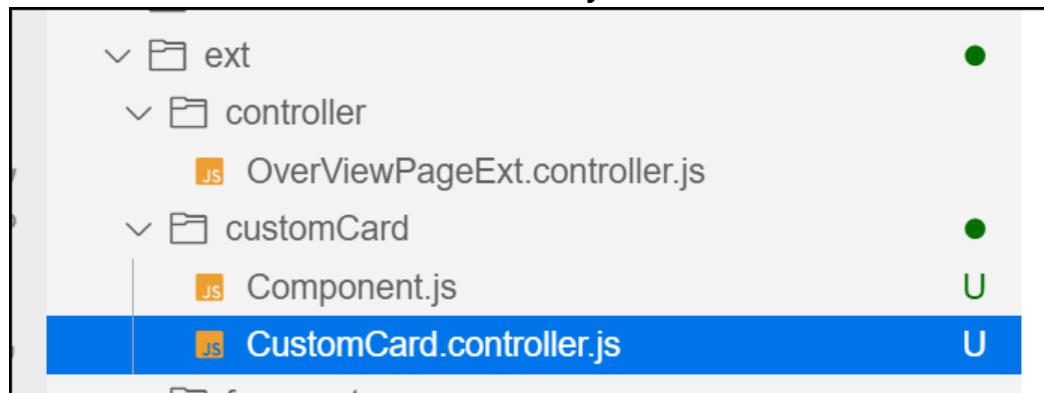
```

1  (function () {
2      "use strict";
3
4      /* controller for custom card */
5      // Controller : https://ui5.sap.com/#/topic/121b8e6337d147af9819129e428f1f75
6      // controller class name can be like app.ovp.ext.customList.CustomList where app.ovp can be removed
7      sap.ui.define([], function() {
8          return {
9              OnInit: function () {},
10
11             onAfterRendering: function () {},
12
13             onExit: function () {}
14         };
15     });
16 })(());

```

< Back **Next >** Exit Guide

- This creates a new file **CustomCard.controller.js**



- In Step 3 Insert Snippet and Next

Add a custom card to an overview page

About Step 1 Step 2 **Step 3** Step 4

Create a `CustomList.fragment.xml` file

Click Insert Snippet to create the fragment for the card content in file `CustomList.fragment.xml`.

Insert Snippet Copy Reset

```

1 <core:FragmentDefinition xmlns="sap.m" xmlns:core="sap.ui.core" xmlns:ovp="sap.ovp.ui"
2 xmlns:template="http://schemas.sap.com/sapui5/extension/sap.ui.core.template/1">
3 |   <!-- Card's Content area code goes here --&gt;
4 &lt;/core:FragmentDefinition&gt;</pre>


< Back Next > Exit Guide


```

- This creates a new file `CustomCardContent.fragment.xml`

The screenshot shows a file tree structure. At the top level, there is a folder named "extend-ovp". Inside it, there are several sub-folders: "node_modules", "webapp", "ext", and "fragments". The "ext" folder contains a "controller" sub-folder with a file named "OverViewPageExt.controller.js". Inside the "webapp" folder, there is a "customCard" folder containing two files: "Component.js" and "CustomCard.controller.js". A new file, "CustomCardContent.fragment.xml", has been created and is highlighted with a red box. The "fragments" folder also contains a file named "CustomFilter.fragment.xml". To the right of the file names, there are status indicators: green dots for most files, and a blue "U" for "Component.js", "CustomCard.controller.js", and "CustomCardContent.fragment.xml", indicating they are under version control.

- In Step 4 Insert Snippet and Exit Guide

Add a custom card to an overview page

About Step 1 Step 2 Step 3 Step 4

Update the manifest file

Click Insert Snippet to update the `manifest.json` file.

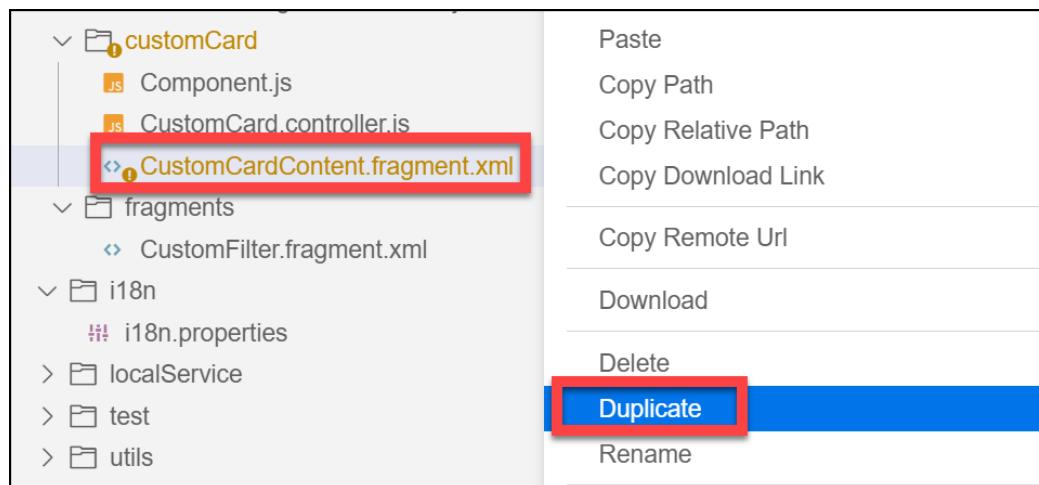
Insert Snippet Copy Reset

```

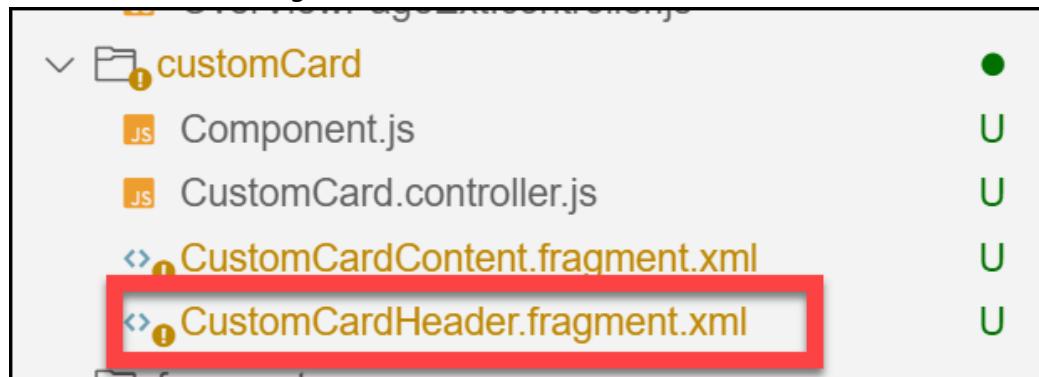
1  "sap.ovp": {
2      "cards": {
3          "card01": {
4              "template": "nato.workshop.extendovp.ext.customCard",
5              "settings": {
6                  "title": "CustomCard"
7              }
8          }
9      }
10 }
```

< Back Next > | Exit Guide

- Duplicate the **CustomCardContent.fragment.xml** file to **CustomCardHeader.fragment.xml**, because we also want a header in our custom card



- CustomCardHeader.fragment.xml



- in file **ext/customCard/Component.js** copy the *defaultValue* from **ContentFragment** to **headerFragment** and change it to **CustomCardHeader**

```

Welcome Application Info – extendovp Component.js X manifest.json CustomCard.controller.js CustomCardContent.fragment.xml
extend-ovp > webapp > ext > customCard > Component.js > sap.ui.define() callback > ...
1 sap.ui.define(["sap/ovp/cards/custom/Component", "jquery.sap.global"], ...
2 function (CardComponent, jQuery) {
3   "use strict";
4
5   return CardComponent.extend("nato.workshop.extendovp.ext.customCard.Component", {
6     // use inline declaration instead of component.json to save 1 round trip
7     metadata: {
8       properties: {
9         contentFragment: {
10           type: "string",
11           defaultValue: "nato.workshop.extendovp.ext.customCard.CustomCardContent",
12         },
13         headerFragment: {
14           type: "string"
15           defaultValue: "nato.workshop.extendovp.ext.customCard.CustomCardHeader",
16         },
17         footerFragment: {
18           type: "string",
19           defaultValue: "",
20         },
21       },
22     }

```

- Implement **CustomCardHeader.fragment.xml**

```

<core:FragmentDefinition xmlns="sap.m" xmlns:core="sap.ui.core"
  xmlns:ovp="sap.ovp.ui"
  xmlns:template="http://schemas.sap.com/sapui5/extension/sap.ui.core.template/1">
  <VBox id="idVbox">
    <Title id="idTitle" text="{i18n>card01_title}"
      class="sapUiSmallMargin"/>
  </VBox>
</core:FragmentDefinition>

```

*Implement **CustomCardContent.fragment.xml**

```

<core:FragmentDefinition xmlns="sap.m" xmlns:core="sap.ui.core"
  xmlns:ovp="sap.ovp.ui"

```

```
xmlns:template="http://schemas.sap.com/sapui5/extension/sap.ui.core.template/1">
  <List id="idList" items="{ path: '/OverviewPage',
  templateShareable: false}"
    growing="true"
    growingThreshold="3">
    <items>
      <StandardListItem id="idSLI" title="{ProductId}"
description="{ProductName}" />
    </items>
  </List>
</core:FragmentDefinition>
```

- Preview Application

The screenshot displays a SAP Fiori application interface. At the top, there is a header bar with a dropdown menu labeled "Standard". Below the header, there are two input fields: "Sales Order Id:" and "Custom Product Name:", both with dropdown menus. A horizontal line with arrows connects these two fields. To the right of the "Custom Product Name:" field is a "More" button and a status indicator "[3 / 13]".

The main content area is divided into two sections:

- Recent Sales:** A table showing sales data. The columns are "Sales Order Id", "SO Item Id", and "Gross Amount". The data is as follows:

Sales Order Id	SO Item Id	Gross Amount
1	10	523
2	10	1K
2	20	600
2	30	294
3	10	5
4	10	196
5	10	3K
5	20	294

Recent Products sold: A list of products sold. The data is as follows:

- 90001 Galaxy S22
- 90001 Galaxy S22
- 90003 Galaxy Tab 5e

At the bottom right of this section is a "More" button and a status indicator "[3 / 13]".

RAP Basics

In this exercise we are going to create a RAP based Application.

- We will create a table for Customers, build a Class to fill the table with some sample data, then create a Root View CDS for that table.
- Next we will create a Service Definition for that CDS, and create 2 Service Bindings. A V2 UI and a V4 UI Service Binding. This to show that there are sometimes a few differences between V2 and V4.
- Next we will create a two UI5 Fiori elements applications, one for each version V2 and V4.

Next we are going to add **delete**, **update** and **create** functionality to the application.

- We are going to add a **managed Behavior Definition** for the CDS.
- First the Delete
- Then the Edit
- Last the Create

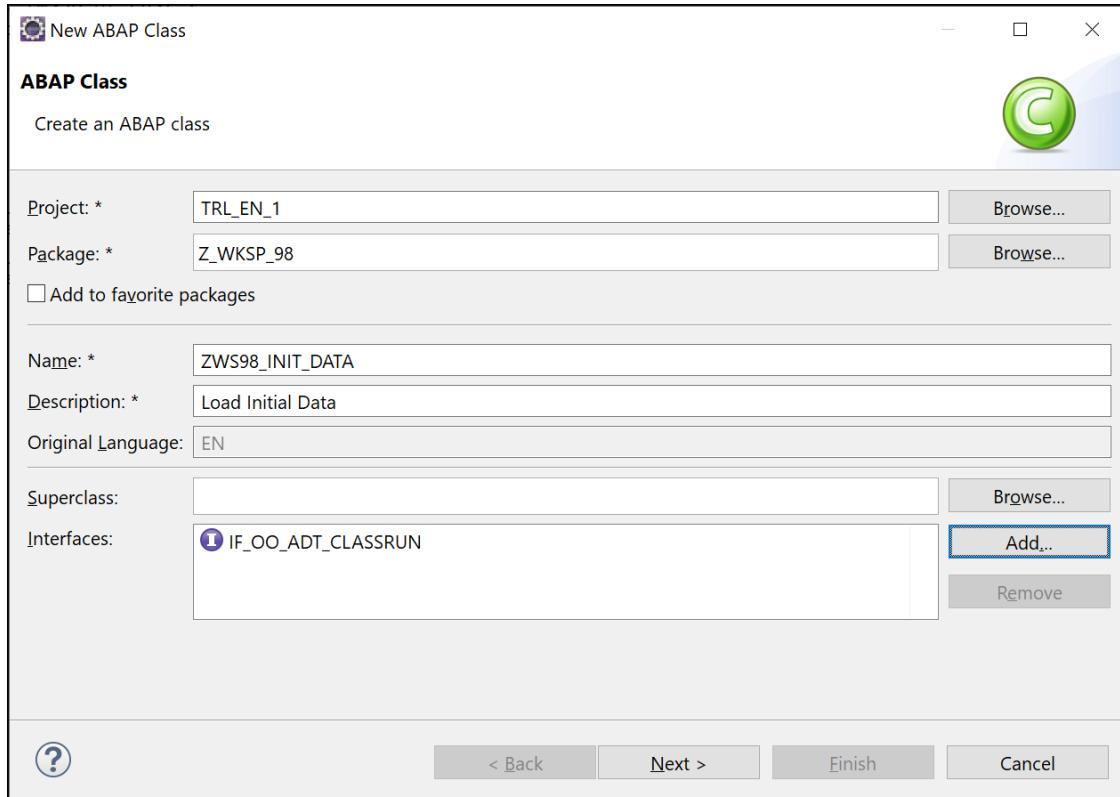
Create a table for Customers **ZWS##_DT_CUST**

```
@EndUserText.label : 'Customers'
@AbapCatalog.enhancement.category : #NOT_EXTENSIBLE
@AbapCatalog.tableCategory : #TRANSPARENT
@AbapCatalog.deliveryClass : #A
@AbapCatalog.dataMaintenance : #RESTRICTED
define table zws##_DT_CUST {
    key client      : abap.clnt not null;
    key id          : ztmde9_customer_id not null;
    name           : abap.char(30);
    street         : abap.char(50);
    city           : abap.char(50);
    country        : land1;
    email          : abap.char(249);
    crea_date_time : timestamppl;
    crea_uname     : syuname;
    lchg_date_time : timestamppl;
    lchg_uname     : syuname;

}
```

Create a ABAP Class **ZWS##_INIT_DATA**

Create the ABAP Class and add the interface **IF_OO_ADT_CLASSRUN**, this will allow you to run the class from eclipse using the F9 function key.



```

CLASS zws##_init_data DEFINITION
  PUBLIC
  FINAL
  CREATE PUBLIC .

  PUBLIC SECTION.

    INTERFACES if_oo_adt_classrun .
    DATA: t_customers      TYPE STANDARD TABLE OF zws##_dt_cust.
    METHODS:
      delete_data,
      load_data.
  PROTECTED SECTION.
  PRIVATE SECTION.
ENDCLASS.

CLASS zws##_init_data IMPLEMENTATION.

  METHOD if_oo_adt_classrun~main.
    DATA: lv_reset  TYPE abap_boolean VALUE 'X',
          lv_delete TYPE abap_boolean VALUE ''.
    CASE abap_true.

```

```

WHEN lv_reset.
    delete_data( ).
    load_data( ).
    out->write( 'Data is reset!' ).

WHEN lv_delete.
    delete_data( ).
    out->write( 'Data is deleted!' ).

ENDCASE.

ENDMETHOD.

METHOD delete_data.
    DELETE FROM zws##_dt_cust WHERE id IS NOT NULL.
    COMMIT WORK AND WAIT.
ENDMETHOD.

METHOD load_data.
    DATA: lv_timestamppl TYPE timestamppl.
    GET TIME STAMP FIELD lv_timestamppl.

    t_customers = VALUE #(
        ( id = '00001' name = 'Pixel Tech' street = 'Rue de la
Meuse 12' city = 'Arlon' country = 'LU' email = 'info@Pixel.lu'
            crea_date_time = lv_timestamppl crea_uname = sy-uname
lchg_date_time = lv_timestamppl lchg_uname = sy-uname )
        ( id = '00002' name = 'Visions' street = 'Rue Thomas
Edison 56' city = 'Luxembourg' country = 'LU' email =
'mail@visions.lu'
            crea_date_time = lv_timestamppl crea_uname = sy-uname
lchg_date_time = lv_timestamppl lchg_uname = sy-uname )
        ( id = '00003' name = 'Techaholic' street = 'De dam 15'
city = 'Amsterdam' country = 'NL' email = 'post@techaholic.nl'
            crea_date_time = lv_timestamppl crea_uname = sy-uname
lchg_date_time = lv_timestamppl lchg_uname = sy-uname )
        ( id = '00004' name = 'Monotech' street = 'Schoolstraat
1' city = 'Alphen' country = 'NL' email = 'po.dep@monotech.nl'
            crea_date_time = lv_timestamppl crea_uname = sy-uname
lchg_date_time = lv_timestamppl lchg_uname = sy-uname )
        ( id = '00005' name = 'Techlanch' street = 'Bronstraat
6' city = 'Brussels' country = 'BE' email = 'info@techlanch.be'
            crea_date_time = lv_timestamppl crea_uname = sy-uname
lchg_date_time = lv_timestamppl lchg_uname = sy-uname )
    ).

    MODIFY zws##_dt_cust FROM TABLE @t_customers.
    COMMIT WORK AND WAIT.
ENDMETHOD.

ENDCLASS.

```

- Run the Class using F9, you will see the message *Data is reset!* in the Console of Eclipse.
- Check the content of the table by opening the table and then pressing F8 or right click on the table and select *Open With -> Data Preview*.

Create a CDS **ZWS\$\$_CDS_RAP_BASIC**

- Create a CDS with Template **Define View**
- Add the word **root** between *define* and *view*

```
define root view ...
```

- and add all the elements, but keep the names of the last 4 elements with the _.
- and set the @AccessControl.authorizationCheck to **#NOT_REQUIRED**.

```
@AbapCatalog.sqlViewName: 'ZWS##CDSRAPB1'
@AbapCatalog.compiler.compareFilter: true
@AbapCatalog.preserveKey: true
@AccessControl.authorizationCheck: #NOT_REQUIRED
@EndUserText.label: 'Basic RAP CDS'
define root view ZWS##_CDS_RAP_BASIC
  as select from zws##_dt_cust
{
  key id          as Id,
  name           as Name,
  street         as Street,
  city           as City,
  country        as Country,
  email          as Email,
  crea_date_time as Crea_Date_Time,
  crea_uname     as Crea_Uname,
  lchg_date_time as Lchg_Date_Time,
  lchg_uname    as Lchg_Uname
}
```

Add Basic Annotations for *lineitem* and *facet*

- Set *lineitem* & *identification* annotations for the fields Id, Name, Street, City, Country and Email.
- Add *@UI.facet*

```

@AbapCatalog.sqlViewName: 'ZWS##CDSRAPB1'
@AbapCatalog.compiler.compareFilter: true
@AbapCatalog.preserveKey: true
@AccessControl.authorizationCheck: #NOT_REQUIRED
@EndUserText.label: 'Basic RAP CDS'
@UI.headerInfo.typeName: 'Customer'
@UI.headerInfo.typeNamePlural: 'Customers'
define root view ZWS##_CDS_RAP_BASIC
  as select from zws##_dt_cust
{
  @UI.facet: [{id: 'Customer', purpose: #STANDARD, type: #IDENTIFICATION_REFERENCE, label: 'Customer', position: 10 }]

    @UI: { lineItem: [{position: 10, importance: #HIGH, label: 'ID' }],
           identification: [{position: 10, label: 'ID' }] }
    key id          as Id,
    @UI: { lineItem: [{position: 20, importance: #HIGH, label: 'Name' }],
           identification: [{position: 20, label: 'Name' }] }
    name          as Name,
    @UI: { lineItem: [{position: 30, importance: #HIGH, label: 'Street' }],
           identification: [{position: 30, label: 'Street' }] }
    street        as Street,
    @UI: { lineItem: [{position: 40, importance: #HIGH, label: 'City' }],
           identification: [{position: 40, label: 'City' }] }
    city          as City,
    @UI: { lineItem: [{position: 50, importance: #HIGH, label: 'Country' }],
           identification: [{position: 50, label: 'Country' }] }
    country       as Country,
    @UI: { lineItem: [{position: 60, importance: #HIGH, label: 'Email' }],
           identification: [{position: 60, label: 'Email' }] }
    email         as Email,
    crea_date_time as Crea_Date_Time,
    crea_uname    as Crea_Uname,
    lchg_date_time as Lchg_Date_Time,
    lchg_uname    as Lchg_Uname
}

```

Create Service Definition **ZUI_WKSP_##_RAP**

- Create a Service Definition and expose your CDS as *Customers*

```
@EndUserText.label: 'Service definition Customers'
define service ZUI_WKSP_##_RAP {
    expose zws##_cds_rap_basic as Customers;
}
```

Create 2 Service Bindings **ZUI_WKSP_##_RAP_V2** and **ZUI_WKSP_##_RAP_V4**

- Create a Service Binding with *Binding Type OData V2 - UI* for *Service Definition ZUI_WKSP_##_RAP*

The screenshot shows the SAP Fiori Launchpad interface. A modal dialog is open for "Create Service Binding". The "Service Binding" tab is selected. The "Name" field contains "ZUI_WKSP_98_RAP_V2". The "Description" field contains "WKSP98 RAP V2". The "Original Language" field contains "EN". The "Binding Type" dropdown is set to "OData V2 - UI". The "Service Definition" dropdown is set to "ZUI_WKSP_98_RAP". Other fields like "Project" and "Package" are also filled in.

- Activate and Publish the Service Binding.
- Test/Preview the Entity Set **Customers**

The screenshot shows the SAP Fiori Launchpad interface. A modal dialog is open for the "Customers" entity set. The table displays five rows of customer data:

ID	Name	Street	City	Country	Email
1	Pixel Tech	Rue de la Meuse 12	Arlon	LU	info@Pixel.lu
2	Visions	Rue Thomas Edison 56	Luxembourg	LU	mail@visions.lu
3	Techaholic	De dam 15	Amsterdam	NL	post@techaholic.nl
4	Monotech	Schoolstraat 1	Alphen	NL	po.dep@monotech.nl
5	Techlanch	Bronstraat 6	Brussels	BE	info@techlanch.be

Customer

Customer	Street: Rue Thomas Edison 56	Country: LU
ID: 2	City: Luxembourg	Email: mail@visions.lu
Name: Visions		

- Create a Service Binding with *Binding Type OData V4 - UI* for *Service Definition*

ZUI_WKSP_##_RAP

Service Binding

Create Service Binding

Project: *	TRL_EN_1	Browse...
Package: *	Z_WKSP_98	Browse...
<input type="checkbox"/> Add to favorite packages		
Name: *	ZUI_WKSP_98_RAP_V4	
Description: *	WKSP98 RAP V4	
Original Language:	EN	
Binding Type: *	OData V4 - UI	
Service Definition: *	ZUI_WKSP_98_RAP	Browse...

- Activate and Publish the Service Binding.

- Test/Preview the Entity Set **Customers**

Standard

Customers (5)

ID	Name	Street	City	Country	Email
1	Pixel Tech	Rue de la Meuse 12	Arlon	LU	info@Pixel.lu
2	Visions	Rue Thomas Edison 56	Luxembourg	LU	mail@visions.lu
3	Techaholic	De dam 15	Amsterdam	NL	post@techaholic.nl
4	Monotech	Schoolstraat 1	Alphen	NL	po.dep@monotech.nl
5	Techlanch	Bronstraat 6	Brussels	BE	info@techlanch.be

Customer

ID: 2	Street: Rue Thomas Edison 56	Country: LU
Name: Visions	City: Luxembourg	Email: mail@visions.lu

Create 2 Fiori Elements application with floorplan **List Report Object Page**, for the OData V2 and the OData V4

V2

- Logon to your SAP Business Application Studio and create a *New Project From Template*
- Create a Fiori Application with *SAP Fiori Elements* and floorplan **List Report Object Page**
- Select *Connect to a System* and select your **abap-cloud-default_abap-trial-xxx (BTP)**
- Select your **V2** Service Binding
- Select *Customers* as your main entity
- Set Project Attributes:

Name	Value
Module name	rap-basic-v2
Application title	RAP Basic V2
Application namespace	nato.workshop
Description	RAP Basic V2
Project folder path	/home/user/projects
Minimum SAPUI5 version	leave as is
Add deployment configuration	Yes
Add FLP configuration	Yes
Configure advanced options	No

- Set Deployment Configuration

Deployment Configuration

Configure deployment settings

Please choose the target *

Cloud Foundry

Destination name *

abap-cloud-default_abap-trial-d4a3a905trial-dev(SCP) - https://45f514af-f57b-4579-990b-8b70ea328491.abap

Add application to managed application router?

Yes No

- Enter Fiori Launchpad Configuration

Name	Value
Semantic Object	NATO
Action	RapBasicV2
Title	RAP Basic V2
Subtitle (optional)	Workshop

- Run the Preview of the application

V4

- Logon to your SAP Business Application Studio and create a *New Project From Template*
- Create a Fiori Application with *SAP Fiori Elements* and floorplan **List Report Object Page**
- Select *Connect to a System* and select your **abap-cloud-default_abap-trial-xxx (BTP)**
- Select your **V4** Service Binding
- Select *Customers* as your main entity
- Set Project Attributes:

Name	Value
Module name	rap-basic-v4
Application title	RAP Basic V4
Application namespace	nato.workshop
Description	RAP Basic V4
Project folder path	/home/user/projects
Minimum SAPUI5 version	leave as is
Add deployment configuration	Yes
Add FLP configuration	Yes
Configure advanced options	No

- Set Deployment Configuration

Deployment Configuration
Configure deployment settings

Please choose the target *

Cloud Foundry

Destination name *

abap-cloud-default_abap-trial-d4a3a905trial-dev(SCP) - https://45f514af-f57b-4579-990b-8b70ea328491.abap

Add application to managed application router?

Yes No

- Enter Fiori Launchpad Configuration

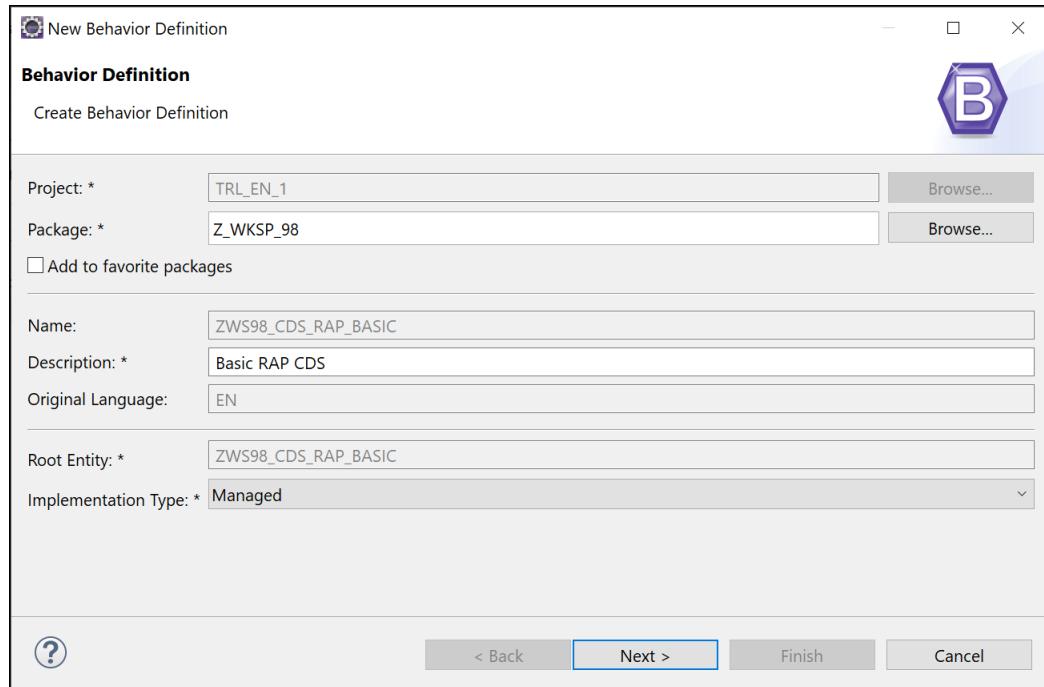
Name	Value
Semantic Object	NATO
Action	RapBasicV4
Title	RAP Basic V4
Subtitle (optional)	Workshop

- Run the Preview of the application

RAP Basics add Delete, Update, Create

Create a *Managed Behavior* for your CDS only for Delete

- Right click on your CDS and select **New Behavior Definition**



- Set the *alias* to *Customer
- Remove the **create** and **update** options:

```
managed implementation in class zbp_ws##_cds_rap_basic unique;

define behavior for ZWS##_CDS_RAP_BASIC alias Customer
persistent table ZWS##_DT_CUST
lock master
authorization master ( instance )
//etag master <field_name>
{
  delete;
}
```

- Activate the Behavior
- Use **CTRL + SHIFT + 1** to open the *Quick Assist* tab.
- Next click on the Class name in the Behavior and see the proposal in the *Quick Assist* tab.
- Double click the Proposal to generate the class

```
CLASS zbp_ws##_cds_rap_basic DEFINITION PUBLIC ABSTRACT FINAL FOR
BEHAVIOR OF zws##_cds_rap_basic.
ENDCLASS.
```

```
CLASS zbp_ws##_cds_rap_basic IMPLEMENTATION.
ENDCLASS.
```

```
CLASS lhc_Customer DEFINITION INHERITING FROM
cl_abap_behavior_handler.
PRIVATE SECTION.
```

```
METHODS get_instance_authorizations FOR INSTANCE AUTHORIZATION
IMPORTING keys REQUEST requested_authorizations FOR Customer
RESULT result.
```

```
ENDCLASS.
```

```
CLASS lhc_Customer IMPLEMENTATION.
```

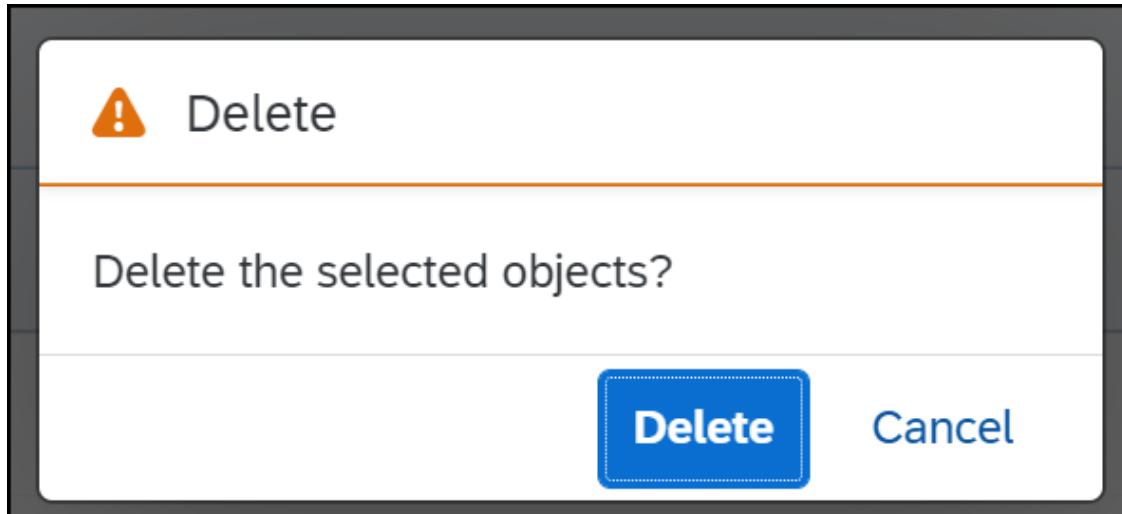
```
METHOD get_instance_authorizations.
ENDMETHOD.
```

```
ENDCLASS.
```

- Test the delete function from either the preview from Eclipse or the BAS application.
- There is now a **Delete** button, when you select one or more lines you can delete them. Check out the difference between the V2 and V4 applications.
- Or click on a line and use the **Delete** button in the Object Page.

If you have deleted all your records, just run the Class **ZWS##_INIT_DATA** again with F9.

ID	Name	Street	City	Country	Email	
1	Pixel Tech	Rue de la Meuse 12	Arlon	LU	info@Pixel.lu	>
2	Visions	Rue Thomas Edison 56	Luxembourg	LU	mail@visions.lu	>
3	Techaholic	De dam 15	Amsterdam	NL	post@techaholic.nl	>
4	Monotech	Schoolstraat 1	Alphen	NL	po.dep@monotech.nl	>
5	Techlanch	Bronstraat 6	Brussels	BE	info@techlanch.be	>



Customer			
ID: 4	Street: Schoolstraat 1	Country: NL	
Name: Monotech	City: Alphen	Email: po.dep@monotech.nl	

Add *Edit* Option to the Behavior Definition

- Add the **update** option to the Behavior Definition.

```
managed implementation in class zbp_ws##_cds_rap_basic unique;

define behavior for ZWS##_CDS_RAP_BASIC alias Customer
persistent table ZWS##_DT_CUST
lock master
authorization master ( instance )
//etag master <field_name>
{
    update;
    delete;
}
```

- Activate the Behavior Definition.
- Test the application again. See the difference between the V2 and V4 versions.

The V4 is not displaying the **Edit** Button whereas the V2 is. This is because for the V4 version we need to add the **Draft** option. We will do this later. For now just test with the V2 application.

- When you press the edit button you are able to change the *key* field **ID**. That is not what we want.
- To prevent this we add **field (readonly) ID;** to the Behavior Definition. This will make the ID field read only.

- Activate and the V2 application again.

Add *Create* Option to the Behavior Definition

- Add the **create** option to the Behavior Definition.

```
managed implementation in class zbp_ws##_cds_rap_basic unique;

define behavior for ZWS##_CDS_RAP_BASIC alias Customer
persistent table ZWS##_DT_CUST
lock master
authorization master ( instance )
etag master Lchg_Date_Time
{
  create;
  update;
  delete;

  field (readonly) ID;
}
```

- Activate the Behavior Definition.
- Test the application again.
- You now see that you can only *once* add a new record, because the second new record that you want to make gives you an error:

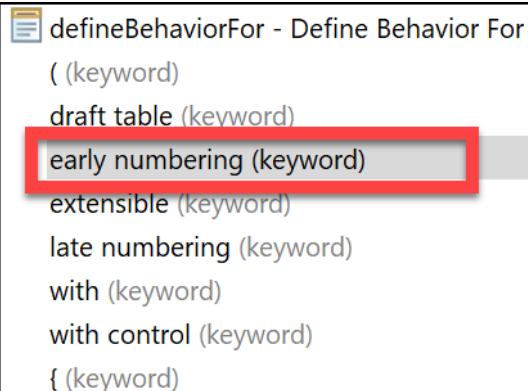
Other Messages

 The key value is already in use. Please enter a different one.

- This can be solved by using **early numbering**

Add **early numbering** to the Behavior Definition

- Add the **early numbering** to the Behavior Definition
- Use the Code Completion option **CTRL+SPACE** to display the options



defineBehaviorFor - Define Behavior For
 (keyword)
 draft table (keyword)
early numbering (keyword)
 extensible (keyword)
 late numbering (keyword)
 with (keyword)
 with control (keyword)
 { (keyword)}

- Double click the word **create** to use the *Quick Assist* to generate a new method
- Implement the new method by checking the MAX number in the table and adding 1 to that max number.

```
METHOD earlynumbering_create.

  LOOP AT entities INTO DATA(entity) WHERE id IS NOT INITIAL.
    APPEND CORRESPONDING #( entity ) TO mapped-customer.
  ENDLOOP.

  DATA(entities_without_id) = entities.
  DELETE entities_without_id WHERE id IS NOT INITIAL.

  "Get max travel ID from standard table
  SELECT SINGLE FROM zws##_dt_cust FIELDS MAX( id ) INTO
  @DATA(max_cust_id).

  "Set Customer Id
  LOOP AT entities_without_id INTO entity.
    max_cust_id += 1.
    entity-Id = max_cust_id.

    APPEND VALUE #( %cid = entity-%cid
                  %key = entity-%key
```

```
) TO mapped-customer.  
ENDLOOP.  
  
ENDMETHOD.
```

- Activate the Behavior Definition.
- Test the application again.
- You now see that when you press save the record gets the next number in the table.

RAP with Draft Version

Now we are going to add *Draft version* to the process. To keep the basic version separated from the new Draft version we need to copy/create a new CDS.

- Copy CDS **ZWS##_CDS_RAP_BASIC** to **ZWS##_CDS_RAP_BASIC_DRAFT**
- Change the *sqlViewName*
- Change the words *Customer* to *Client*, that way we can see which CDS we are using when we test the applications.

```

@AbapCatalog.sqlViewName: 'ZWS##CDSRAPB2'
@AbapCatalog.compiler.compareFilter: true
@AbapCatalog.preserveKey: true
@AccessControl.authorizationCheck: #NOT_REQUIRED
@EndUserText.label: 'Basic RAP CDS'
@UI.headerInfo.typeName: 'Client'
@UI.headerInfo.typeNamePlural: 'Clients'
define root view ZWS##_CDS_RAP_BASIC_DRAFT
    as select from zws##_dt_cust
{
    @UI.facet: [{id: 'Client', purpose: #STANDARD, type: #IDENTIFICATION_REFERENCE, label: 'Client', position: 10 }]

        @UI: { lineItem: [{position: 10, importance: #HIGH, label: 'ID' }],
            identification: [{position: 10, label: 'ID' }] }
        key id           as Id,
        @UI: { lineItem: [{position: 20, importance: #HIGH, label: 'Name' }],
            identification: [{position: 20, label: 'Name' }] }
        name          as Name,
        @UI: { lineItem: [{position: 30, importance: #HIGH, label: 'Street' }],
            identification: [{position: 30, label: 'Street' }] }
        street         as Street,
        @UI: { lineItem: [{position: 40, importance: #HIGH, label: 'City' }],
            identification: [{position: 40, label: 'City' }] }
        city          as City,
        @UI: { lineItem: [{position: 50, importance: #HIGH, label: 'Country' }],
            identification: [{position: 50, label: 'Country' }] }
        country        as Country,
        @UI: { lineItem: [{position: 60, importance: #HIGH, label: 'Country' }] }

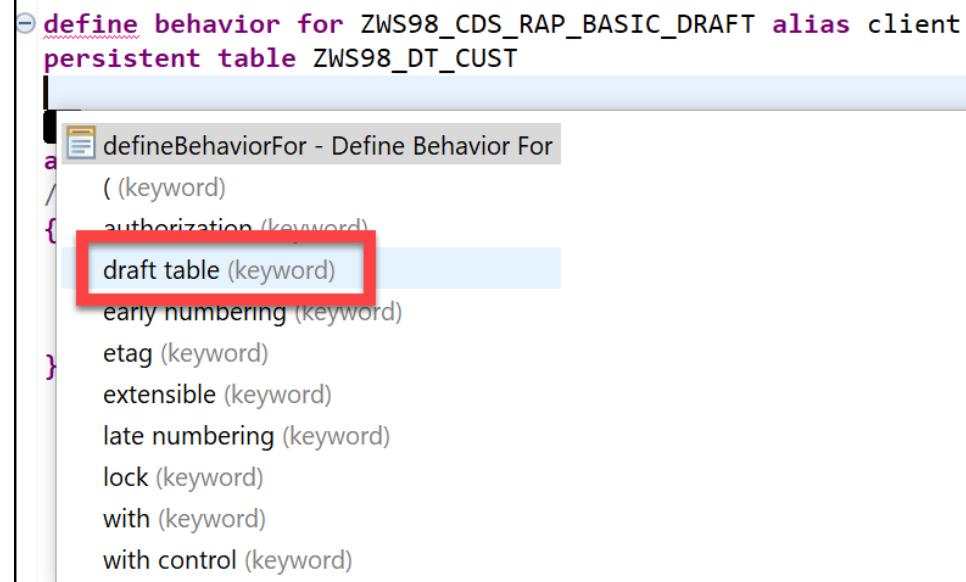
```

```
'Email' }],
    identification: [{position: 60, label: 'Email'}] }
email      as Email,
crea_date_time as Crea_Date_Time,
crea_uname   as Crea_Uname,
lchg_date_time as Lchg_Date_Time,
lchg_uname   as Lchg_Uname
}
```

- Add the new CDS to the Service Definition as *Clients*

```
@EndUserText.label: 'Service definition Customers'
define service ZUI_WKSP_##_RAP {
    expose ZWS##_CDS_RAP_BASIC as Customers;
    expose ZWS##_CDS_RAP_BASIC_DRAFT as Clients;
}
```

- Add a new *managed Behavior Definition* to the new CDS
- Add the option **with draft**;
- Set the Alias to **client**
- Under *persistent table* add **draft table ZWS##_DT_CUST_DR**, use the Code Completion



- Use the **Quick Assist** to generate the **Draft table** for you, See the newly added field **"%admin"**

```
@EndUserText.label : 'Draft table for entity
ZWS##_CDS_RAP_BASIC_DRAFT'
```

```

@AbapCatalog.enhancement.category : #EXTENSIBLE_ANY
@AbapCatalog.tableCategory : #TRANSPARENT
@AbapCatalog.deliveryClass : #A
@AbapCatalog.dataMaintenance : #RESTRICTED
define table zws##_dt_cust_dr {
    key mandt      : mandt not null;
    key id        : ztmde9_customer_id not null;
    name          : abap.char(30);
    street        : abap.char(50);
    city          : abap.char(50);
    country       : land1;
    email         : abap.char(249);
    crea_date_time : timestamppl;
    crea_uname    : syuname;
    lchg_date_time : timestamppl;
    lchg_uname    : syuname;
    "%admin"      : include sych_bdl_draft_admin_inc;

}

```

- Add the options: **lock master total etag xx** and **etag master xx**
- Set the *field (readonly) id*;
- Result of the Behavior Definition should look like this.

```

managed implementation in class zbp_ws##_cds_rap_basic_draft
unique;
with draft;

define behavior for ZWS##_CDS_RAP_BASIC_DRAFT alias client
persistent table ZWS##_DT_CUST
draft table ZWS##_DT_CUST_DR
lock master total etag Lchg_Date_Time
authorization master ( instance )
etag master Lchg_Date_Time
{
    create;
    update;
    delete;

    field ( readonly ) id;
}

```

- Put your cursor on the class name in the first line of the Behavior Definition and use the *Quick Assist* to Create the implementation class.

- Test your application using the preview of **Clients** in the *Service Binding*
- Delete works as expected
- Edit, you now have the option to see the **Display Saved Version** or the **New Draft Version**
- If you make a change you shortly see **Saving Draft** at the bottom and then **Draft saved**. This tells you if there is a draft version available.
- With the *Cancel* button you get the option to *Discard* all changes.
- Create works again only 1 time, after that you get an error that the Key Value is already in use.

Fix the Create

- To fix the option to create multiple draft versions you need to add the option **late numbering** to the Behavior Definition.

```
managed implementation in class zbp_ws##_cds_rap_basic_draft
unique;
with draft;
//strict; //Comment this line in to enable strict mode. The strict
mode is prerequisite to be future proof regarding syntax and to be
able to release your BO.

define behavior for ZWS##_CDS_RAP_BASIC_DRAFT alias client
late numbering
persistent table ZWS##_DT_CUST
draft table ZWS##_DT_CUST_DR
lock master total etag Lchg_Date_Time
authorization master ( instance )
etag master Lchg_Date_Time
{
  create;
  update;
  delete;

  field ( readonly ) id;
}
```

- When you activate the new Behavior Definition, you need to fix 2 things.
- First: use the *Quick Assist* to recreate the draft table, this because we need an extra key field.

```

@EndUserText.label : 'Draft table for entity
ZWS98_CDS_RAP_BASIC_DRAFT'
@AbapCatalog.enhancement.category : #EXTENSIBLE_ANY
@AbapCatalog.tableCategory : #TRANSPARENT
@AbapCatalog.deliveryClass : #A
@AbapCatalog.dataMaintenance : #RESTRICTED
define table zws98_dt_cust_dr {
    key mandt      : mandt not null;
    key id        : ztmde9_customer_id not null;
    key draftuuid : sdraft_uuid;
    name          : abap.char(30);
    street        : abap.char(50);
    city          : abap.char(50);
    country       : land1;
    email         : abap.char(249);
    crea_date_time : timestamppl;
    crea_uname   : syuname;
    lchg_date_time : timestamppl;
    lchg_uname   : syuname;
    "%admin"     : include sych_bdl_draft_admin_inc;
}

}

```

- To be able to generate the new definition of the Draft Table you first may have to delete the Draft Table or make sure there are no records in the Draft Table.
- Second: use the *Quick Assist* to add a new method for *late numbering*
- Implement the new method **adjust_numbers**, to get the latest number from the table

```

METHOD adjust_numbers.
  IF mapped-client IS NOT INITIAL.
    " Get max travel ID from standard table
    SELECT SINGLE FROM zws98_dt_cust FIELDS MAX( id ) INTO
@DATA(max_cust_id).
    max_cust_id += 1.
    mapped-client[ 1 ]-%key-Id = max_cust_id.
  ENDIF.
ENDMETHOD.

```

Blogs and Tips and Tricks

[Connecting from SAP Business Application Studio to SAP ABAP Environment in BTP](#)

[Set Up Local Development Using VS Code](#)

If cf --version does not work after install of Cloud Foundry, then you need to make a symbolick link manually:

Open a CMD in administrator mode and execute following command:

```
mklink "c:\Program Files\Cloud Foundry\cf.exe" "c:\Program  
Files\Cloud Foundry\cf8.exe"
```

Login to cf

Start a terminal and execute the following:

```
cf login -a <API-URL>  
and enter your email, password and select your organisation and space.
```

```
cf login -a https://api.cf.us10.hana.ondemand.com
```

If Fiori Template wizard niet wil starten:

Uninstall the existing generator with the command: `npm uninstall -g @sap/generator-fiori`.

Then try and install the generator again by executing: `npm i -g @sap/generator-fiori`.

[SAP Help, Develop Applications With RAP](#)

[Space in your SAP Business Application Studio is low.](#)

- You can delete the **node_modules** folders out of you projects.
- Then clean the *Trash* folder, the *yarn cache* and *npm cache* folder, using the next commands in a terminal

```
rm -rf /home/user/.local/share/Trashfiles  
rm -rf /home/user/.cache  
rm -rf /home/user/.npm
```

Additional Online Workshops

Workshops about the ABAP RESTful Application Programming Model (RAP)

CodeJam - SAP Business Technology Platform, ABAP Environment & ABAP RESTful Application Programming Model