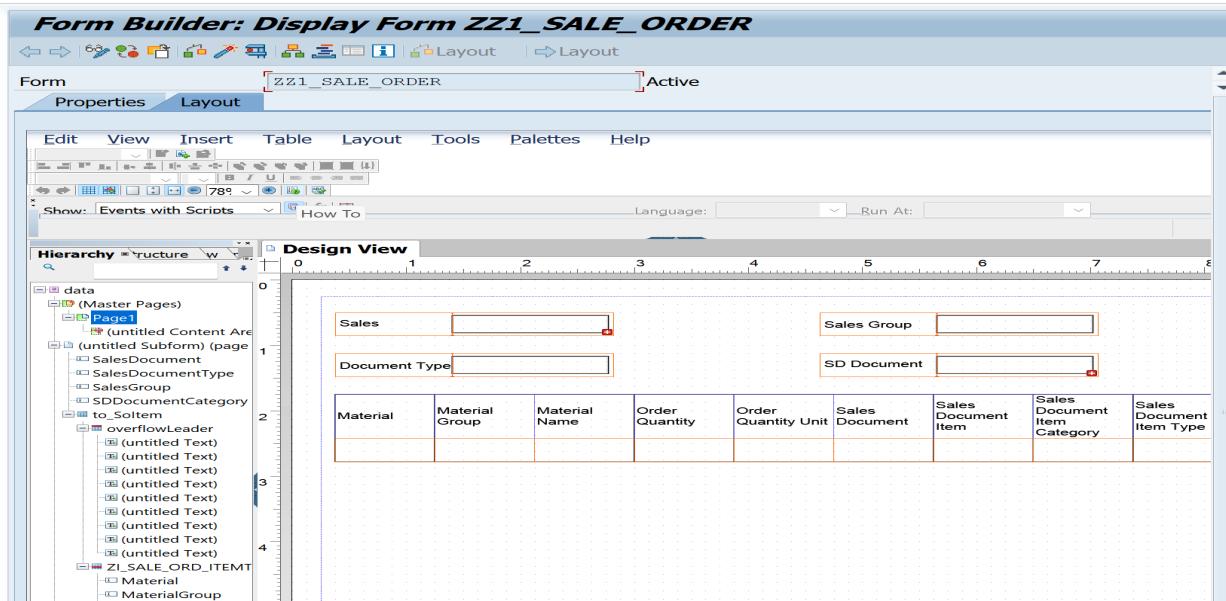


Steps to be followed.

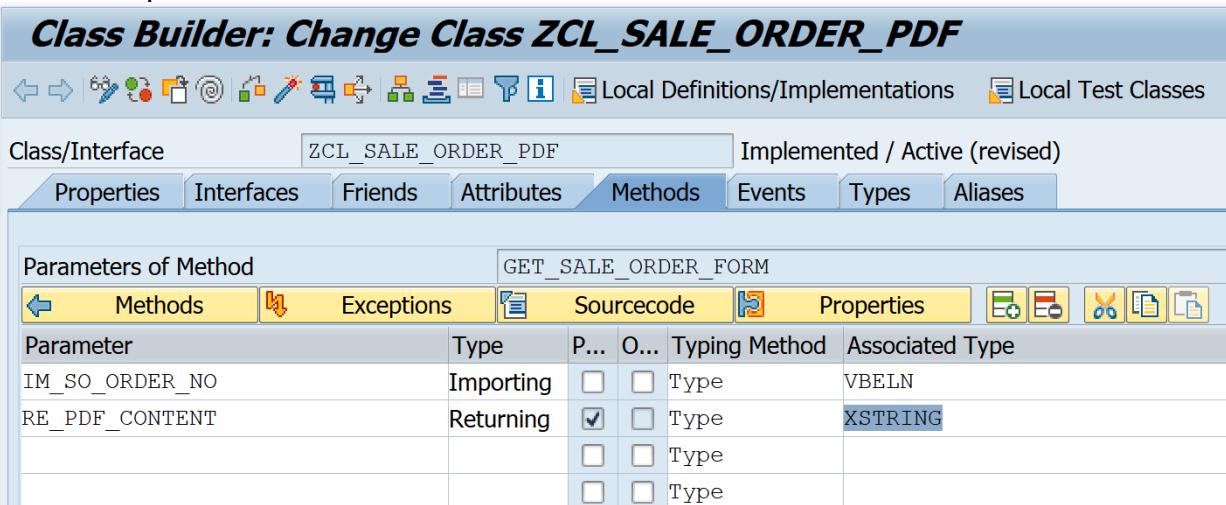
1. Develop an Adobe form and common method to give the form content output as Base64

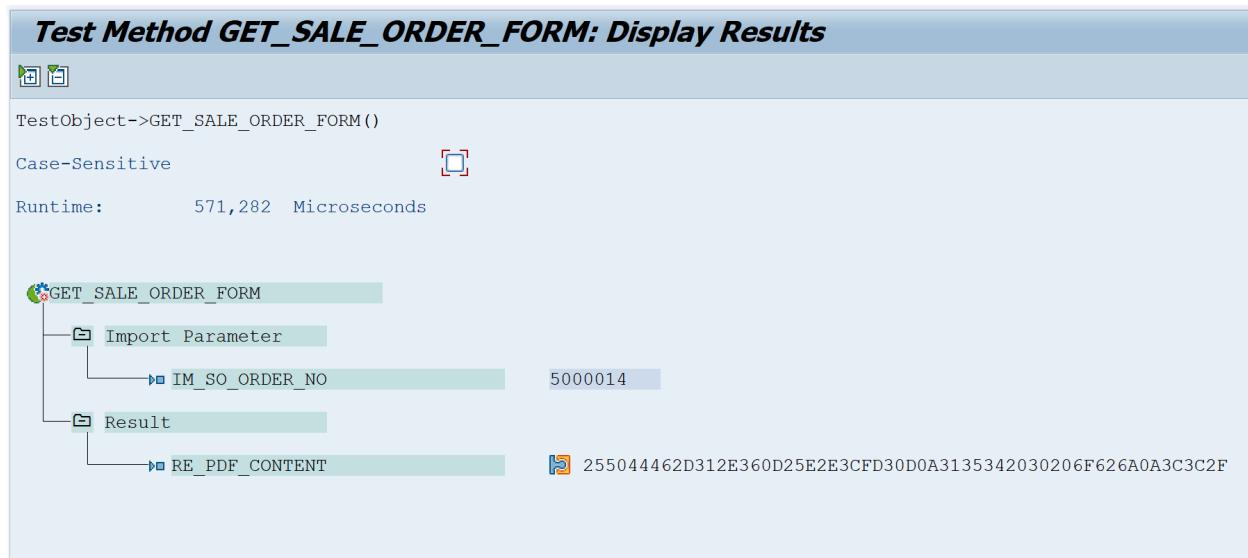
In this example I am using XML Based form interface but it is possible to use the ABAP Dictionary based interface as well.

Form Name: ZZ1_SALE_ORDER



Here I am creating a new class and build the driver Method to call the form using and return back the output form content





2. Now Let us build a RAP Application using Web UI -> Since we do not have Create, update and other functionalities, I am not creating behavior definitions.

Create a new base View entity: ZI_RAP_SALE_ORD

```

@AbapCatalog.viewEnhancementCategory: [#NONE]
@AccessControl.authorizationCheck: #NOT_REQUIRED
@EndUserText.label: 'Interface View for Sales Order'
@Metadata.ignorePropagatedAnnotations: true
@ObjectModel.usageType:{
  serviceQuality: #X,
  sizeCategory: #S,
  dataClass: #MIXED
}
define root view entity ZI_RAP_SALE_ORD
as select from I_SalesDocumentBasic
{
  key SalesDocument,
  SDDocumentCategory,
  SalesDocumentType,
  CreatedByUser,
  CreationDate
}
  
```

Create a Consumption View for the Above View Entity: ZC_RAP_SALE_ORD

```

@AccessControl.authorizationCheck: #NOT_REQUIRED
@EndUserText.label: 'Consumption View for Sales Order'
@Metadata.allowExtensions: true
define root view entity ZC_RAP_SALE_ORD
provider contract transactional_query
as projection on ZI_RAP_SALE_ORD
{
  key SalesDocument,
  SDDocumentCategory,
  SalesDocumentType,
  CreatedByUser,
  
```

```
    CreationDate  
}
```

Create a Metadata Extension file for the Consumption view: ZMDE_RAP_SALE_ORD
We can also directly give annotations in consumption view itself, but as best practice I am creating a new file to have these Annotation properties.

```
@Metadata.layer: #CORE  
@Search.searchable: true  
@UI.headerInfo:{  
    typeName: 'Sales Order',  
    typeNamePlural: 'Sale Orders',  
    title:{ type:#STANDARD,  
        label:'Document' ,  
        value:'SalesDocument' }  
  
}  
annotate view ZC_RAP_SALE_ORD with  
{  
    @UI:{ lineItem: [{ position : 10 }] }  
    @Search:{ defaultSearchElement: true,  
        fuzzinessThreshold: 0.7 }  
    SalesDocument;  
  
    @UI:{ lineItem: [{ position : 20}],  
        selectionField: [{ position : 20 }] }  
    @Search:{ defaultSearchElement: true,  
        fuzzinessThreshold: 0.7 }  
    SDDocumentCategory;  
  
    @UI:{ lineItem: [{ position : 30}] }  
    @Search:{ defaultSearchElement: true,  
        fuzzinessThreshold: 0.7 }  
    SalesDocumentType;  
  
    @UI:{ lineItem: [{ position : 40}],  
        selectionField: [{ position : 40 }] }  
    CreatedByUser;  
  
    @UI:{ lineItem: [{ position : 50}],  
        selectionField: [{ position : 50 }] }  
    CreationDate;  
}
```

Create a new service definition for the Consumption View: ZSDEF_ZC_RAP_SALE_ORD

```
@EndUserText.label: 'Service Definition for Sale Order'  
define service ZSDEF_ZC_RAP_SALE_ORD {  
    expose ZC_RAP_SALE_ORD;  
}
```

Create a new Service Bindings for the Service Definition: ZBIND_ZC_RAP_SALE_ORD of binding type ODATA V2 - UI and publish it. We can get the Metadata using Service URL

Create a new Fiori Application in Business Application Studio, Open BTP in Trail account and make sure your system is configured in Destinations

The screenshot shows the SAP BTP Cockpit interface. The left sidebar includes sections for Overview, Services, Service Marketplace, Instances and Subscriptions, Cloud Foundry (Spaces, Space Quotas, Org Members), HTML5 Applications, Connectivity (Destinations, Destination Certificates, Destination Trust, Cloud Connectors), Security, Entitlements, and Usage Analytics. The main area displays a subaccount titled "trial - Destinations" with two entries: "HTTP" and another "HTTP" entry. A modal dialog titled "Check Connection" is open over the second "HTTP" entry, showing a green checkmark and the message "Connection to [redacted] successful". Navigation links at the top include Trial Home, 6d1bbb6btrial, trial, and a search bar.

3.Create a new Fiori Application using the template and use our above ODATA V2-UI service in BAS

Project From Template 

SAP Fiori generator 

Select Template and Target Location
SAP Fiori generator

Template Selection
Template: List Report Page

Data Source and Service Selection
Data Source: Connect to a System
System: 
Service: ZBIND_ZC_RAP_SALE_ORD (1) - OData V2

Entity Selection
Main Entity: ZC_RAP_SALE_ORD
Table Type: Responsive

Project Attributes
Module Name: project1
Application Title: App Title
Application Namespace:
Description: An SAP Fiori application.
[More...](#)

Project Attributes
Configure the main project attributes.

Module name 

Application title 

Application namespace 

Description 

Project folder path  

Minimum SAPUI5 version  

Enable TypeScript
 Yes No

Add deployment configuration
 Yes No

Add FLP configuration

  The generated project will not open in a stand-alone folder.

Once the Project is created and all the dependencies are installed. You can preview the application by right click project1 and choose preview application -> start Fiori-run

The screenshot shows a SAP Fiori application interface. At the top, there is a header bar with a back arrow, forward arrow, refresh icon, and a URL bar containing "port8080-workspaces-ws-4pdzq.us10.trial.applicationstudio.cloud.sap/test/fipSandbox.html?sap...". Below the header are navigation links for ALM, Solman, SIDF, and SAMI LAND, along with a "All Bookmarks" link. The main title is "App Title" with a "DU" button. A "Standard" dropdown menu is open. On the left, there is a SAP logo. The main content area is titled "Sale Orders (53)" and displays a table with 14 rows of sales document data. The columns are: Sales Document, SD Documen..., Sales Docum..., Created By, and Created On. The data shows various entries with dates ranging from Aug 26, 2024, to Dec 5, 2024.

Sales Document	SD Documen...	Sales Docum...	Created By	Created On
1	C	OR	EWMUSER	Aug 26, 2024 >
2	C	OR	ABAPER1	Sep 2, 2024 >
3	C	OR	ABAPER1	Sep 4, 2024 >
4	C	OR	ABAPER1	Sep 17, 2024 >
5	C	OR	ABAPER1	Sep 18, 2024 >
6	C	OR	ABAPER1	Oct 1, 2024 >
7	C	OR	ABAPER1	Oct 1, 2024 >
8	C	OR	ABAPER1	Nov 13, 2024 >
9	C	OR	ABAPER1	Nov 13, 2024 >
10	C	OR	ABAPER1	Nov 13, 2024 >
11	C	OR	ABAPER1	Nov 18, 2024 >
12	C	OR	ABAPER1	Nov 18, 2024 >
13	C	OR	ABAPER1	Nov 19, 2024 >
14	C	OR	ABAPER1	Dec 5, 2024 >

Now our RAP Application is ready and separate form driver program is also ready. Let us see on how to integrate it. This Integration can be Done by the following two ways

- ➔ Using SEGW ODATA -> GET_STREAM
- ➔ Using Custom View Entity that is implemented by ABAP class using Object model query

4.Create a New SEGW ODATA Project and Implement GET Stream method That consumes the common driver program and send back the content

Entity properties Properties

	Name	Is Key	Edm T...	Prec.	Scale	Max L...	Unit Pr...	Creat...	Updat...
	docId	<input checked="" type="checkbox"/>	Edm.St...	0	0	0		<input type="checkbox"/>	<input type="checkbox"/>
	value	<input type="checkbox"/>	Edm.St...	0	0	0		<input type="checkbox"/>	<input type="checkbox"/>
	mimeType	<input type="checkbox"/>	Edm.St...	0	0	0		<input type="checkbox"/>	<input type="checkbox"/>
	fileName	<input type="checkbox"/>	Edm.St...	0	0	0		<input type="checkbox"/>	<input type="checkbox"/>

GET Stream Code:

""->Read Document ID from input

DATA(lv_doc) = VALUE numc10(IT_KEY_TAB[name = 'docId']-value OPTIONAL).

""->Call Form Driver Program to get the Form Content

DATA(lv_form_content) = new ZCL_SALE_ORDER_PDF()->GET_SALE_ORDER_FORM(im_so_order_no = conv #(lv_doc)).

""->Build Document Name and Set it in Header

set_header(is_header = VALUE #(name = 'Content-Disposition'
value = |inline; filename={ lv_doc } ;|)).

copy_data_to_ref(exporting
is_data = VALUE /IWFDN/IF_MGW_CORE_RUNTIME=>TY_S_MEDIA_RESOURCE(
mime_type = 'application/pdf'
value = lv_form_content)
changing
cr_data = er_stream).

Testing the SEGW ODATA Service

The screenshot shows the SAP Gateway Client interface. The top navigation bar includes 'SAP Gateway Client', 'Execute', 'Select', 'Service Administration', 'Service Implementation', 'Switch User', 'Entity Set', and 'Add URI Option'. The main area has tabs for 'HTTP Request' and 'HTTP Response - Processing Time = 513 ms'. The 'HTTP Request' tab shows a table with 'Header Name' and 'Value' columns. The 'HTTP Response' tab shows a table with 'Header Name' and 'Value' columns, including entries like ~status_code=200, ~status_reason=OK, and cache-control=no-store, no-cache. Below these tabs is a large table with various fields and data rows, such as Sales Document, Sales Group, Document Type, SD Document Category, and a detailed table for Material, Material Group, Material Name, Order Quantity, Order Quantity Unit, Sales Document, Sales Document Item, Sales Document Item Category, and Sales Document Item Type.

Add the Following line in the Interface View: ZI_RAP_SALE_ORD

We are adding two fields

ShowPDF -> Has the text Preview(SEGW) in the table

LinkToPDF -> Get Stream Link to be used to navigate as link to open PDF File

Lines:

```
'Preview(SEGW)' as ShowPDF,  
concat('/sap/opu/odata/sap/ZODATA_TRAIN_SRV/SaleOrderDocumentSet("",  
    concat(SalesDocument, "")/$value')) as LinkToPDF  
  
0 define root view entity ZI_RAP_SALE_ORD  
1   as select from I_SalesDocumentBasic  
2 {  
3   key SalesDocument,  
4     SDDocumentCategory,  
5     SalesDocumentType,  
6     CreatedByUser,  
7     CreationDate,  
8     'Preview(SEGW)' as ShowPDF,  
9     concat('/sap/opu/odata/sap/ZODATA_TRAIN_SRV/SaleOrderDocumentSet('',  
0     concat(SalesDocument, '')/$value')) as LinkToPDF  
1 }  
2
```

Add the Following highlighted Line in the Consumption view ZC_RAP_SALE_ORD to expose the fields

```
@AccessControl.authorizationCheck: #NOT_REQUIRED  
@EndUserText.label: 'Consumption View for Sales Order'  
@Metadata.allowExtensions: true  
define root view entity ZC_RAP_SALE_ORD  
provider contract transactional_query  
as projection on ZI_RAP_SALE_ORD  
{  
    key SalesDocument,  
    SDDocumentCategory,  
    SalesDocumentType,  
    CreatedByUser,  
    CreationDate,  
    | ShowPDF,  
    | LinkToPDF  
}
```

Add the Following Lines in the Metadata extension file to establish the link to:

```
@UI:{  
lineItem: [{ position : 60, label : 'PDF'}, { type : #WITH_URL, url:'LinkToPDF' }]  
}  
ShowPDF;
```

```

@Metadata.layer: #CORE
@Search.searchable: true
@UI.headerInfo:
  typeName: 'Sales Order',
  typeNamePlural: 'Sale Orders',
  title:{ type:#STANDARD,
    label:'Document' ,
    value:'SalesDocument'}
}

annotate view ZC_RAP_SALE_ORD with
{
  @UI:{ lineItem: [{ position : 10}] }
  @Search:{ defaultSearchElement: true,
    fuzzinessThreshold: 0.7 }
  SalesDocument;

  @UI:{ lineItem: [{ position : 20}],
    selectionField: [{ position : 20 }] }
  @Search:{ defaultSearchElement: true,
    fuzzinessThreshold: 0.7 }
  SDDocumentCategory;

  @UI:{ lineItem: [{ position : 30}] }
  @Search:{ defaultSearchElement: true,
    fuzzinessThreshold: 0.7 }
  SalesDocumentType;

  @UI:{ lineItem: [{ position : 40}],
    selectionField: [{ position : 40 }] }
  CreatedByUser;

  @UI:{ lineItem: [{ position : 50}],
    selectionField: [{ position : 50 }] }
  CreationDate;
  @UI:{ lineItem: [{ position : 60, label : 'PDF'}, { type : #WITH_URL, url:'LinkToPDF' }]}
  ShowPDF;
}

```

Testing the RAP Application- Now we can see the New Field as clickable link
standard ▾



The screenshot shows a SAP RAP application interface. At the top, there are search fields for 'SD Document Category' and 'Created By', and buttons for 'Search' and 'Reset'. Below the header is a toolbar with icons for search, refresh, and other functions. The main area displays a table titled 'Sale Orders (53)' with 16 rows of data. The columns are: Sales Document, SD Documen..., Sales Docum..., Created By, Created On, and PDF. The 'PDF' column contains links labeled 'Preview(SEGW)' followed by a right-pointing arrow. The entire 'ShowPDF' column is highlighted with a blue box.

Sales Document	SD Documen...	Sales Docum...	Created By	Created On	PDF
1	C	OR	EWMUSER	Aug 26, 2024	Preview(SEGW) >
2	C	OR	ABAPER1	Sep 2, 2024	Preview(SEGW) >
3	C	OR	ABAPER1	Sep 4, 2024	Preview(SEGW) >
4	C	OR	ABAPER1	Sep 17, 2024	Preview(SEGW) >
5	C	OR	ABAPER1	Sep 18, 2024	Preview(SEGW) >
6	C	OR	ABAPER1	Oct 1, 2024	Preview(SEGW) >
7	C	OR	ABAPER1	Oct 1, 2024	Preview(SEGW) >
8	C	OR	ABAPER1	Nov 13, 2024	Preview(SEGW) >
9	C	OR	ABAPER1	Nov 13, 2024	Preview(SEGW) >
10	C	OR	ABAPER1	Nov 13, 2024	Preview(SEGW) >
11	C	OR	ABAPER1	Nov 18, 2024	Preview(SEGW) >
12	C	OR	ABAPER1	Nov 18, 2024	Preview(SEGW) >
13	C	OR	ABAPER1	Nov 19, 2024	Preview(SEGW) >
14	C	OR	ABAPER1	Dec 5, 2024	Preview(SEGW) >
15	C	OR	ABAPER1	Dec 11, 2024	Preview(SEGW) >
16	C	OR	ABAPER1	Dec 11, 2024	Preview(SEGW) >

On clicking the Link, the SEGW Service is called and PDF is shown

Material	Material Group	Material Name	Order Quantity	Order Quantity Unit	Sales Document	Sales Document Item	Sales Document Item Category	Sales Document Item Type
52		In operation (5	H	5000014	10	J3GP	
52		In operation (1	H	5000014	20	J3GP	
52		In operation (1	H	5000014	30	J3GP	

5. Using Custom View Entity that is implemented by ABAP class using Object model query

Create a new Custom View Entity: ZI_SALE_ORDER_PDF

```
@EndUserText.label: 'Sale Order PDF'  
@ObjectModel.query.implementedBy: 'ABAP:ZCL_SALE_ORDER_PDF'  
define custom entity ZI_SALE_ORDER_PDF  
with parameters  
p_so_document_id : vbeln  
{  
    key so_doc_pdf : abap.string(0);  
}
```

Create a new Class : ZCL_SALE_ORDER_PDF and use interface IF_RAP_QUERY_PROVIDER and redefine the Select method

```
class ZCL_SALE_ORDER_PDF definition  
public  
final  
create public .
```

public section.

```
interfaces IF_RAP_QUERY_PROVIDER .  
PROTECTED SECTION.  
PRIVATE SECTION.  
ENDCLASS;
```

```
CLASS ZCL_SALE_ORDER_PDF IMPLEMENTATION.
```

```
METHOD if_rap_query_provider~select.
```

""->Internal Table Declaration

```

DATA : lt_pdf TYPE TABLE OF zi_sale_order_pdf.
""->Object Declaration
DATA : lo_somu_form_services TYPE REF TO cl_somu_form_services.

DATA : lv_form_out TYPE string.

TRY.
  IF io_request->is_data_requested( ).
    DATA(lv_offset) = io_request->get_paging()->get_offset().
    DATA(lv_page_size) = io_request->get_paging()->get_page_size().
    DATA(lv_max_rows) = COND #( WHEN lv_page_size = if_rap_query_paging=>page_size_unlimited
      THEN 0 ELSE lv_page_size ) .

    DATA(lt_params) = io_request->get_parameters().
    ""->Take the Input Parameter Name from Custom Entity which is using this ABAP Class
    DATA(lv_so_doc_id) = VALUE numc10(
      lt_params[ parameter_name = 'P_SO_DOCUMENT_ID' ]-value OPTIONAL).
    ""->Call the PDF Content Common Driver Method
    data(lv_so_form) = get_sale_order_form( im_so_order_no = conv #( lv_so_doc_id ) ).
    IF lv_so_form IS NOT INITIAL.
      ""->Convert Xstring to Base64
      CALL FUNCTION 'SCMS_BASE64_ENCODE_STR'
        EXPORTING
          input = lv_so_form
        IMPORTING
          output = lv_form_out.
    ENDIF.
    lt_pdf = VALUE #( ( so_doc_pdf = lv_form_out ) ).
    io_response->set_total_number_of_records( 1 ).
    io_response->set_data( lt_pdf ).
  ENDIF.
  CATCH cx_rap_query_provider.
ENDTRY.
ENDMETHOD.

```

ENDCLASS;

Create a new Service Definition for the Custom Entity: ZSDEF_SALE_ORD_PDF

@EndUserText.label: 'Service Definition for Sale Order PDF'

```

define service ZSDEF_SALE_ORD_PDF {
  expose ZI_SALE_ORDER_PDF;
}

```

Create a new Service binding for the Custom Entity of type ODATA V2 Web API and publish it

The screenshot shows the SAP Fiori Launchpad interface. A modal dialog is open for creating a new service binding. The title bar says "Service Binding: ZBIND_SALE_ORD_PDF". The dialog is divided into several sections:

- General Information:** Describes the general information about the service binding.
- Binding Type:** OData V2 - Web API.
- Service Versions:** A table showing one version entry: Version 1.0.0, API State Not Released, Service Definition ZSDEF_SALE_ORD_PDF. Buttons for "Add..." and "Remove" are available.
- Service Version Details:** Shows the local service endpoint as Published and a button for Unpublish.
- Service Information:** Displays the Service URL as /sap/opu/odata/sap/ZBIND_SALE_ORD_PDF.
- Entity Set and Association:** Shows the entity set ZI_SALE_ORDER_PDFSet.

Test the Service Definition using Service URL:

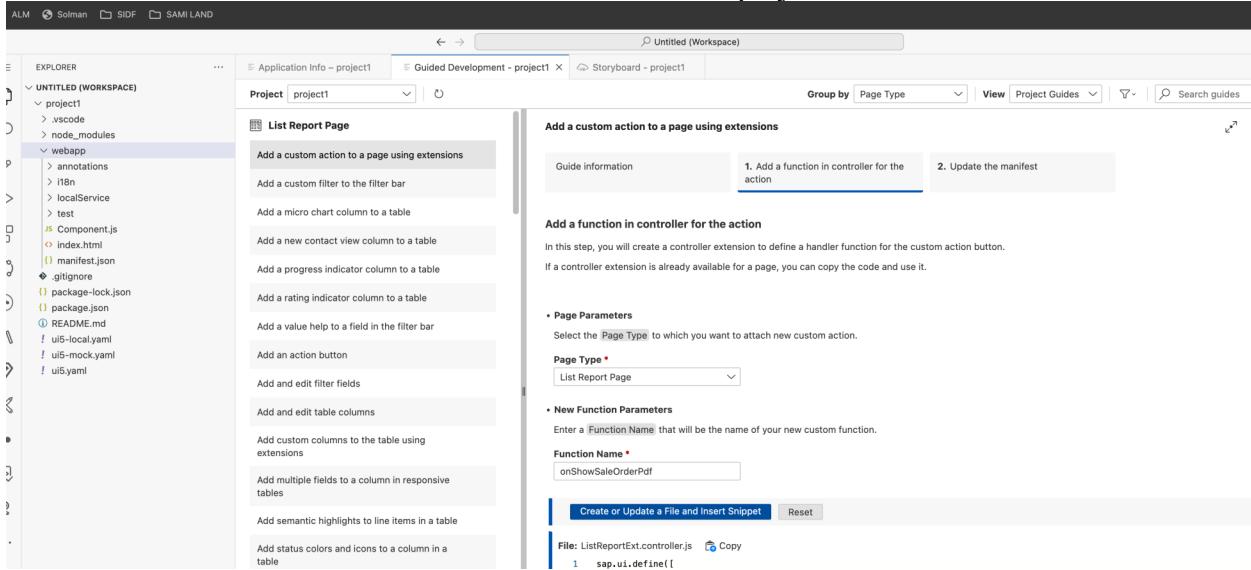
The screenshot shows the SAP Application Studio interface. At the top, there are buttons for GET, POST, PUT, PATCH, MERGE, DELETE, and HEAD methods, with GET selected. Below that is a dropdown for protocol (HTTP or HTTPS) set to HTTP. The URL entered is /sap/opu/odata/sap/ZBIND_SALE_ORD_PDF/ZL_SALE_ORDER_PDF(p_so_document_id='5000014')/Set?\$format=json. The request section shows a single row with 'Value' under 'Value'. The response section shows an 'HTTP Response - Processing Time = 286 ms' with headers and a JSON payload. The JSON payload is as follows:

```
1  {
2    "d" : {
3      "results" : [
4        {
5          "_metadata" : {
6            ...
7          },
8          "p_so_document_id" : "5000014",
9          "so_doc_pdf" : "JVBERi0xLjYNJeLjz9MNCjE1NCAwIG9iago8PC9GaWx0ZXIvRmxhdGVEZWNVZGUvRmlyc3QgODk3"
11     }
12   }
13 }
```

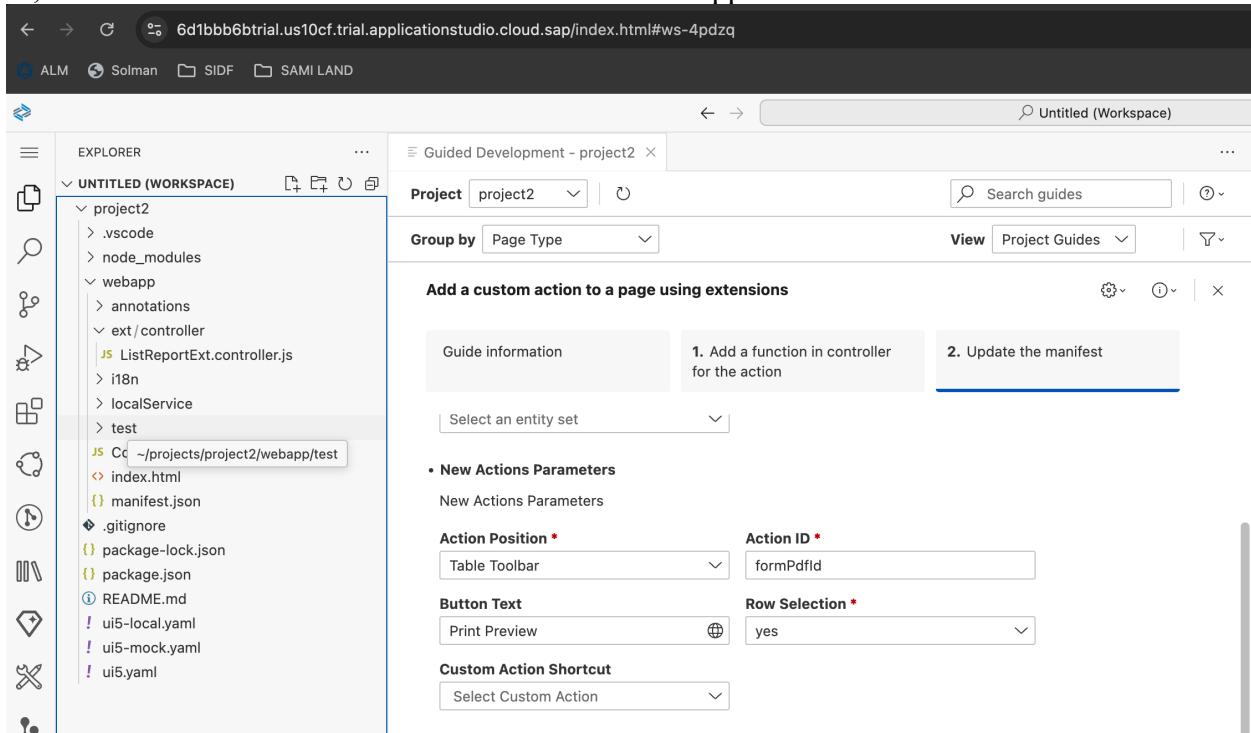
Since we get the Data in Base64 format we need to create a extension in the UI application.
Right Click the project and open guided Development

The screenshot shows the SAP Application Studio Explorer view. The left sidebar has icons for ALM, Solman, SIDF, and SAMI LAND. The main area shows a workspace named 'UNTITLED (WORKSPACE)' containing a folder 'project1'. A context menu is open over 'project1', with the 'Open Guided Development' option highlighted in blue. Other options in the menu include New File..., New Folder..., Open in Integrated Terminal, Preview Application, Open Application Info, Show Page Map, Override Annotations, Open Data Editor, Archive Project, Add Folder to Workspace..., Open Folder Settings, Remove Folder from Workspace, Find in Folder..., and Paste.

Choose the “1. Add Custom Action to page Using Extensions”, then give the Function Name and click on create. This will add a new Controller file in the project



Now Choose the “2. Update the manifest”, then give the Entity Set as the root entity set, Action Id, Row selection and Button text. Then click insert snippet



Extensions will be added by guided development in manifest file

```
,  
    "extends": {  
        "extensions": {  
            "sap.ui.controllerExtensions": {  
                "sap.suite.ui.generic.template.ListReport.view.ListReport": {  
                    "controllerName": "project3.ext.controller.ListReportExt",  
                    "sap.ui.generic.app": {  
                        "ZC_RAP_SALE_ORD": {  
                            "EntitySet": "ZC_RAP_SALE_ORD",  
                            "Actions": {  
                                "formPdfId": {  
                                    "id": "formPdfIdButton",  
                                    "text": "Print Preview",  
                                    "press": "onShowSaleOrderPdf",  
                                    "requiresSelection": true  
                                }  
                            }  
                        }  
                    }  
                }  
            }  
        }  
    }  
}
```

Need to add the following lines in the Manifest.json file

->In the dataSources property:

```
,  
    "dataSources": {  
        "mainService": {  
            "uri": "/sap/opu/odata/sap/ZBIND_ZC_RAP_SALE_ORD/",  
            "type": "OData",  
            "settings": {  
                "annotations": [  
                    "ZBIND_ZC_RAP_SALE_ORD_VAN",  
                    "annotation"  
                ],  
                "localUri": "localService/metadata.xml",  
                "odataVersion": "2.0"  
            }  
        },  
        "ZBIND_ZC_RAP_SALE_ORD_VAN": {  
            "uri": "/sap/opu/odata/IWFND/CATALOGSERVICE;v=2/Annotations(TechnicalName='ZBIND_ZC_RAP_SALE_ORD_VAN',Version='0001')/$value/",  
            "type": "ODataAnnotation",  
            "settings": {  
                "localUri": "localService/ZBIND_ZC_RAP_SALE_ORD_VAN.xml"  
            }  
        },  
        "ZBIND_SALE_ORD_PDF": {  
            "uri": "/sap/opu/odata/sap/ZBIND_SALE_ORD_PDF",  
            "type": "OData",  
            "settings": {  
                "localUri": "localService/ZBIND_SALE_ORD_PDF/metadata.xml"  
            }  
        },  
        "annotation": {  
            "type": "ODataAnnotation",  
            "uri": "annotations/annotation.xml",  
            "settings": {  
                "localUri": "annotations/annotation.xml"  
            }  
        }  
    }  
}
```

->In the models property:

```
manifest.json X  
bapp > manifest.json > sap.ui5 > models  
'sap.ui5': {  
    ...  
    "models": [  
        "i18n": {  
            "type": "sap.ui.model.resource.ResourceModel",  
            "settings": {  
                "bundleName": "project4.i18n.i18n"  
            }  
        },  
        "": {  
            "dataSource": "mainService",  
            "preload": true,  
            "settings": {  
                "defaultBindingMode": "TwoWay",  
                "defaultCountMode": "Inline",  
                "refreshAfterChange": false,  
                "metadataUrlParams": {  
                    "sap-value-list": "none"  
                }  
            }  
        },  
        "oPdfModel": {  
            "dataSource": "ZBIND_SALE_ORD_PDF",  
            "preload": true,  
            "type": "sap.ui.model.odata.v2.ODataModel",  
            "settings": {  
                "defaultBindingMode": "TwoWay",  
                "defaultOperationModel": "Server",  
                "defaultCountMode": "Request",  
                "useBatch": "false"  
            }  
        },  
        "@i18n": {  
            "type": "sap.ui.model.resource.ResourceModel",  
            "uri": "i18n/i18n.properties"  
        }  
    ]  
},
```

We can write out custom code in the added Controller file to open PDF Preview
 To get the PDF data from RAP ODATA Web API



```

> .vscode
> node_modules
< webapp
  > annotations
  < ext/controller
    JS ListReportExt.controller.js
      > i18n
      > localService
      > test
      JS Component.js
      <> index.html
      {} manifest.json
    & .gitignore
    {} package-lock.json
    {} package.json
    README.md
    ! ui5-local.yaml
    ! ui5-mock.yaml

```

```

4  ],
5   function (MessageToast,PDFViewer) {
6     'use strict';
7     return [
8       getPdffromOdata: function () {
9         var oModel = this.getView().getModel("oPdfModel");
10        var oGrid = this.getView().byId("project4::sap.suite.ui.generic.template.ListRepo
11        // get Sales Order from selected index
12        var saleDocument = oGrid.getSelectedItem().getBindingContext().getObject().SalesD
13        return new Promise((resolve, reject) => {
14          // Perform Read operation and pass billingdoc as parameter to URL
15          oModel.read("/ZI_SALE_ORDER_PDF(p_so_document_id='"+ saleDocument + "')/Set",
16          {
17            success: function (oData, Response) {
18              resolve(oData);
19            },
20            error: function (oError) {
21              reject(oError);
22            }
23          });
24        },
25        onShowSaleOrderPdf: async function (oEvent) {
26

```

ListReportExt.controller.js

```

sap.ui.define([
  "sap/m/MessageToast",
  "sap/m/PDFViewer"
], function (MessageToast,PDFViewer) {
  'use strict';
  return {
    getPdffromOdata: function () {
      var oModel = this.getView().getModel("oPdfModel");
      var oGrid =
this.byId("project4::sap.suite.ui.generic.template.ListReport.view.ListReport::ZC_RAP_SALE_ORD--responsiveTable");
      // get Sales Order from selected index
      var saleDocument = oGrid.getSelectedItem().getBindingContext().getObject().SalesDocument;
      return new Promise((resolve, reject) => {
        // Perform Read operation and document number as parameter to URL
        oModel.read("/ZI_SALE_ORDER_PDF(p_so_document_id='"+ saleDocument + "')/Set",
        {
          success: function (oData, Response) {
            resolve(oData);
          },
          error: function (oError) {
            reject(oError);
          }
        });
    },
    onShowSaleOrderPdf: async function (oEvent) {
      var opdfViewer = new PDFViewer();

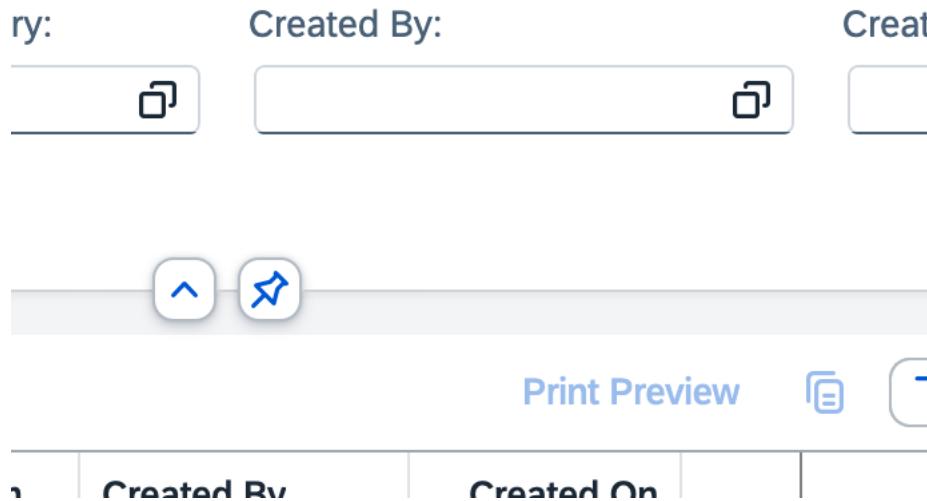
```

```

this.getView().addDependent(opdfViewer);
var oBusyDialog = new sap.m.BusyDialog({
    title: 'Generating Form...'
});
oBusyDialog.open();
// Get the PDF data
var vPDF = await this.getPdffromOdata();
let base64EncodedPDF = vPDF.results[0].so_doc_pdf;
let decodedPdfContent = atob(base64EncodedPDF);
let byteArray = new Uint8Array(decodedPdfContent.length);
for (var i = 0; i < decodedPdfContent.length; i++) {
    byteArray[i] = decodedPdfContent.charCodeAt(i);
}
var blob = new Blob([byteArray.buffer], {
    type: 'application/pdf'
});
var pdfurl = URL.createObjectURL(blob);
jQuery.sap.addUrlWhitelist("blob"); // register blob url as whitelist
opdfViewer.setSource(pdfurl);
opdfViewer.setVisible(true);
opdfViewer.setTitle("Billing Document ");
opdfViewer.open();
oBusyDialog.close();
}
};

});
```

Now test the Application: Print Preview Button is added



You can select any Sale order to enable the Print Preview button and click it to open the PDF file

← → ⌂ port8080-workspaces-ws-4pdzq.us10.trial.applicationstudio.cloud.sap/test/flpSandbox.html?sap-... ☆

ALM Solman SIDF SAMI LAND

SAP App Title ▾

Standard ▾

No filters active

Sale Orders (53)

Print Preview

Print Preview

Sales Document SD Documen... Sales Docum... Created By Created On

35	C	CMR	ABAPER1	Jan 19, 2025 >
36	C	CMR	ABAPER1	Jan 19, 2025 >
37	C	CMR	ABAPER1	Jan 19, 2025 >
38	C	CMR	ABAPER1	Jan 19, 2025 >
39	C	OR	EWMUSER	Jan 27, 2025 >
5000001	C	J3G1	ABAPER1	Sep 13, 2024 >
5000002	C	J3G1	ABAPER1	Sep 13, 2024 >
5000003	C	J3G2	ABAPER1	Sep 13, 2024 >
5000004	C	J3G1	ABAPER1	Sep 13, 2024 >
5000005	C	J3G2	ABAPER1	Sep 13, 2024 >
5000006	C	J3G1	ABAPER1	Sep 13, 2024 >
5000007	C	J3G2	ABAPER1	Sep 13, 2024 >
5000008	C	J3G1	ABAPER1	Sep 13, 2024 >
5000009	C	J3G2	ABAPER1	Sep 13, 2024 >
5000010	C	J3G1	ABAPER1	Sep 13, 2024 >
5000011	C	J3G2	ABAPER1	Sep 13, 2024 >
5000012	C	J3G1	SEDDE	Sep 26, 2024 >
5000013	C	J3G1	SEDDE	Sep 30, 2024 >
5000014	C	J3G2	SEDDE	Sep 30, 2024 >



App Title ▾

Standard ▾

Filters active



Sale Orders (53)

Print Preview



Sales Document	SD Documen...	Sales Docum...	Created By	Created On	
35	C	CMR	ABAPER1	Jan 19, 2025	>
36	C	CMR	ABAPER1	Jan 19, 2025	>
37	C	CMR	ABAPER1	Jan 19, 2025	>
38	C	CMR	ABAPER1	Jan 19, 2025	>
39	C	OR	EWMUSER	Jan 27, 2025	>
5000001	C	J3G1	Generating Form...		>
5000002	C	J3G2			>
5000003	C	J3G1			>
5000004	C	J3G1	ABAPER1	Sep 13, 2024	>
5000005	C	J3G2	ABAPER1	Sep 13, 2024	>
5000006	C	J3G1	ABAPER1	Sep 13, 2024	>
5000007	C	J3G2	ABAPER1	Sep 13, 2024	>
5000008	C	J3G1	ABAPER1	Sep 13, 2024	>
5000009	C	J3G2	ABAPER1	Sep 13, 2024	>

Generating Form...



blob:https://port8080-workspaces-ws-4pdzq.us10.trial.applicationstudio.cloud.sap/dc8aea7a...

ALM Solman SIDF SAMI LAND All Books

dc8aea7a-4573-486f-938d-ecbf87... 1 / 1 77% + 🔍 ⌂

Sales Document 5000014 Sales Group []

Document Type J3G2 SD Document Category C

Material	Material Group	Material Name	Order Quantity	Order Quantity Unit	Sales Document	Sales Document Item	Sales Document Item Category	Sales Document Item Type
52		In operation (5	H		5000014	10	J3GP	
52		In operation (1	H		5000014	20	J3GP	
52		In operation (1	H		5000014	30	J3GP	