CRUD application

Before starting with application it is important to spend some
time in understanding the service we are going to use for the same.

Lets spend some time with our Odata Service

- https://services.odata.org/V2/Odata/Odata.svc/
- https://services.odata.org/V2/Odata/Odata.svc/$metadata
- https://services.odata.org/V2/OData/OData.svc/Suppliers
- https://services.odata.org/V2/OData/OData.svc/Suppliers?$format=json
- https://services.odata.org/V2/OData/OData.svc/Suppliers(1)?$format=json
- https://services.odata.org/V2/OData/OData.svc/Suppliers(1)?$format=json&$select=ID,Name
- https://services.odata.org/V2/OData/OData.svc/Suppliers(1)/Products
- https://services.odata.org/V2/(S(readwrite))/Odata/Odata.svc/

Now we understand Northwind lets start building our app

Configure Odata Destination

Go to SAP Cloud platform->Scroll down and you will see neo trial click on that->Choose Connectivity -> Destinations
Create a destination name MyDest

```
Name: MyDest
Type: HTTP
Description: My odata destination
url: https://services.odata.org
proxy type : Internet`
Authentication: NoAuthentication
WebIdeEnabled:true
WebIDESystem: MyDest
WebideUsage: odata_gen
```

Lets start with Application

- Create a new SAPUI5 application SecondApp with namespace Supplier and view as App.
- Once app is created check what all is created. Then add our odata service right click your project
  and choose New ->Odata service.
- Choose service url and destination MyDest. Enter relative url as below and choose next default model.

```
V2/Odata/Odata.svc/
```

- Lets check what it has done. In neo.app.json you will the destination reference. In local service metadata of the service.
  In manifest.json u will see reference to your service and data models.

Adding our views for models.

- Since Flexiblecolumn layout is part of sap.f lets add the reference in manifest.json

```
{
    "_version": "1.12.0",
    "sap.app": {
        "id": "Supplier.SecondApp",
        "type": "application",
        "i18n": "i18n/i18n.properties",
        "applicationVersion": {
            "version": "1.0.0"
        },
        "title": "{{appTitle}}",
        "description": "{{appDescription}}",
        "sourceTemplate": {
            "id": "servicecatalog.connectivityComponentForManifest",
            "version": "0.0.0"
        },
        "dataSources": {
            "Odata.svc": {
                "uri": "/MyDest/V2/Odata/Odata.svc/",
                "type": "OData",
                "settings": {
                    "localUri": "localService/metadata.xml"
                }
            }
        }
    },
    "sap.ui": {
        "technology": "UI5",
        "icons": {
            "icon": "",
            "favIcon": "",
            "phone": "",
            "phone@2": "",
            "tablet": "",
            "tablet@2": ""
        },
```

```
        "deviceTypes": {
            "desktop": true,
            "tablet": true,
            "phone": true
        }
    },
    "sap.ui5": {
        "flexEnabled": false,
        "rootView": {
            "viewName": "Supplier.SecondApp.view.App",
            "type": "XML",
            "async": true,
            "id": "App"
        },
        "dependencies": {
            "minUI5Version": "1.65.6",
            "libs": {
                "sap.ui.layout": {},
                "sap.ui.core": {},
                "sap.m": {},
                "sap.f":{}
            }
        },
        "contentDensities": {
            "compact": true,
            "cozy": true
        },
        "models": {
            "i18n": {
                "type": "sap.ui.model.resource.ResourceModel",
                "settings": {
                    "bundleName": "Supplier.SecondApp.i18n.i18n"
                }
            },
            "": {
                "type": "sap.ui.model.odata.v2.ODataModel",
                "settings": {
                    "defaultOperationMode": "Server",
                    "defaultBindingMode": "OneWay",
                    "defaultCountMode": "Request"
                },
                "dataSource": "Odata.svc",
                "preload": true
            }
        },
        "resources": {
            "css": [
                {
                    "uri": "css/style.css"
                }
            ]
        },
        "routing": {
            "config": {
                "routerClass": "sap.m.routing.Router",
                "viewType": "XML",
                "async": true,
                "viewPath": "Supplier.SecondApp.view",
                "controlAggregation": "pages",
                "controlId": "app",
                "clearControlAggregation": false
            },
            "routes": [
                {
                    "name": "RouteApp",
                    "pattern": "RouteApp",
                    "target": [
                        "TargetApp"
                    ]
                }
            ],
            "targets": {
                "TargetApp": {
                    "viewType": "XML",
                    "transition": "slide",
                    "clearControlAggregation": false,
                    "viewId": "App",
                    "viewName": "App"
                }
            }
        }
    }
}
```

Modify App.view.xml to add our flexible column layout.

```
<mvc:View controllerName="Supplier.SecondApp.controller.App" xmlns:mvc="sap.ui.core.mvc" displayBlock="true" xmlns="sap.m"
xmlns:f="sap.f">
    <App id="app">
        <f:FlexibleColumnLayout id="flexibleColumnLayout" backgroundDesign="Translucent" layout="OneColumn"></f:FlexibleColumnLayout>
    </App>
</mvc:View>
```

Now lets create a new view for our Suppliers Supplier.view.xml by right clicking webapp

Lets add a list to Supplier.view.xml

```
<mvc:View controllerName="Supplier.SecondApp.controller.Supplier"   xmlns="sap.m"
    xmlns:semantic="sap.f.semantic"
    xmlns:mvc="sap.ui.core.mvc">

    <semantic:SemanticPage
        id="masterPage"
        preserveHeaderStateOnScroll="true"
        toggleHeaderOnTitleClick="false">
        <semantic:titleHeading>
            <Title
                id="masterPageTitle"
                text="{masterView>/title}"
                level="H2"/>
        </semantic:titleHeading>
        <semantic:content>
            <!-- For client side filtering add this to the items attribute: parameters: {operationMode: 'Client'}}" -->
            <List
                id="list"
                width="auto"
                class="sapFDynamicPageAlignContent"
                items="{
                    path: '/Suppliers',
                    sorter: {
                        path: 'Name',
                        descending: false
                    },
                    groupHeaderFactory: '.createGroupHeader'
                }"
                busyIndicatorDelay="{masterView>/delay}"
                noDataText="{masterView>/noDataText}"
                mode="{= ${device>/system/phone} ? 'None' : 'SingleSelectMaster'}"
                growing="true"
                growingScrollToLoad="true"
                updateFinished=".onUpdateFinished"
                selectionChange=".onSelectionChange">
                <infoToolbar>
                    <Toolbar
                        active="true"
                        id="filterBar"
                        visible="{masterView>/isFilterBarVisible}"
                        press=".onOpenViewSettings">
                        <Title
                            id="filterBarLabel"
                            text="{masterView>/filterBarLabel}"
                            level="H3"/>
                    </Toolbar>
                </infoToolbar>
                <headerToolbar>
                    <OverflowToolbar>
                        <SearchField
                            id="searchField"
                            showRefreshButton="true"
                            tooltip="{i18n>masterSearchTooltip}"
                            search=".onSearch"
                            width="auto">
                            <layoutData>
                                <OverflowToolbarLayoutData
                                    minWidth="150px"
                                    maxWidth="240px"
                                    shrinkable="true"
                                    priority="NeverOverflow"/>
                            </layoutData>
                        </SearchField>
                        <ToolbarSpacer/>
                        <Button
                            id="sortButton"
                            press=".onOpenViewSettings"
                            icon="sap-icon://sort"
                            type="Transparent"/>
                        <Button
                            id="filterButton"
                            press=".onOpenViewSettings"
                            icon="sap-icon://filter"
                            type="Transparent"/>
                        <Button
                            id="groupButton"
```

```
                        press=".onOpenViewSettings"
                        icon="sap-icon://group-2"
                        type="Transparent"/>
                    </OverflowToolbar>
                </headerToolbar>
                <items>
                    <ObjectListItem
                        type="Navigation"
                        press=".onSelectionChange"
                        title="{Name}"
                        number="{
                            path: 'ID',
                            formatter: '.formatter.currencyValue'
                        }"
                        numberUnit="{Address/City}">
                    </ObjectListItem>
                </items>
            </List>
        </semantic:content>
    </semantic:SemanticPage>
</mvc:View>
```

Lets add this view in App.view.xml Flexible column layout begin pages aggregation

```
<mvc:View controllerName="Supplier.SecondApp.controller.App" xmlns:mvc="sap.ui.core.mvc" displayBlock="true" xmlns="sap.m"
xmlns:f="sap.f">
    <App id="app">
        <f:FlexibleColumnLayout id="flexibleColumnLayout" backgroundDesign="Translucent" layout="OneColumn">
            <f:beginColumnPages>
                <mvc:XMLView id="beginView" viewName="Supplier.SecondApp.view.Supplier"/>
            </f:beginColumnPages>
        </f:FlexibleColumnLayout>
    </App>
</mvc:View>
```

Now we have the supplier view we will now move towards the next view which is Supplier detail and products
We create another view SupplierDetail.

```
<mvc:View xmlns:core="sap.ui.core" xmlns:mvc="sap.ui.core.mvc" xmlns="sap.m"
controllerName="Supplier.SecondApp.controller.SupplierDetail"
    xmlns:html="http://www.w3.org/1999/xhtml">

</mvc:View>
```

Change App.view.xml to include this view in middle.

```
<mvc:View controllerName="Supplier.SecondApp.controller.App" xmlns:mvc="sap.ui.core.mvc" displayBlock="true" xmlns="sap.m"
xmlns:f="sap.f">
    <App id="app">
        <f:FlexibleColumnLayout id="flexibleColumnLayout" backgroundDesign="Translucent" layout="OneColumn">
            <f:beginColumnPages>
                <mvc:XMLView id="beginView" viewName="Supplier.SecondApp.view.Supplier"/>
            </f:beginColumnPages>
            <f:midColumnPages>
            <mvc:XMLView id="detailView" viewName="Supplier.SecondApp.view.SupplierDetail"/>
        </f:midColumnPages>
        </f:FlexibleColumnLayout>
    </App>
</mvc:View>
```

So now our initial view is one column on click of button we will change it to two columns.
This is done sap/f/library control.

```
sap.ui.define([
    "sap/ui/core/mvc/Controller",
    'sap/f/library'
], function (Controller, fioriLibrary) {
    "use strict";

    return Controller.extend("Supplier.SecondApp.controller.Supplier", {

        /**
         * Called when a controller is instantiated and its View controls (if available) are already created.
         * Can be used to modify the View before it is displayed, to bind event handlers and do other one-time initialization.
         * @memberOf Supplier.SecondApp.view.Supplier
         */
        onInit: function () {
            this.oView = this.getView();
        },
        onSelectionChange: function () {
                var oFCL = this.oView.getParent().getParent();

                oFCL.setLayout(fioriLibrary.LayoutType.TwoColumnsMidExpanded);
            }
            /**
             * Similar to onAfterRendering, but this hook is invoked before the controller's View is re-rendered
             * (NOT before the first rendering! onInit() is used for that one!).
             * @memberOf Supplier.SecondApp.view.Supplier
             */
            //  onBeforeRendering: function() {
            //
            //  },

        /**
         * Called when the View has been rendered (so its HTML is part of the document). Post-rendering manipulations of the HTML
could be done here.
         * This hook is the same one that SAPUI5 controls get after being rendered.
         * @memberOf Supplier.SecondApp.view.Supplier
         */
        //  onAfterRendering: function() {
        //
        //  },

        /**
         * Called when the Controller is destroyed. Use this one to free resources and finalize activities.
         * @memberOf Supplier.SecondApp.view.Supplier
         */
        //  onExit: function() {
        //
        //  }

    });

});
```

Lets now add things to Supplier details. We will be using object detail class lets add that dependency to manifest.json

```
{
    "_version": "1.12.0",
    "sap.app": {
        "id": "Supplier.SecondApp",
        "type": "application",
        "i18n": "i18n/i18n.properties",
        "applicationVersion": {
            "version": "1.0.0"
        },
        "title": "{{appTitle}}",
        "description": "{{appDescription}}",
        "sourceTemplate": {
            "id": "servicecatalog.connectivityComponentForManifest",
            "version": "0.0.0"
        },
        "dataSources": {
            "Odata.svc": {
                "uri": "/MyDest/V2/Odata/Odata.svc/",
                "type": "OData",
                "settings": {
                    "localUri": "localService/metadata.xml"
                }
            }
        }
    },
    "sap.ui": {
        "technology": "UI5",
        "icons": {
            "icon": "",
            "favIcon": "",
            "phone": "",
            "phone@2": "",
            "tablet": "",
```

```
            "tablet@2": ""
        },
        "deviceTypes": {
            "desktop": true,
            "tablet": true,
            "phone": true
        }
    },
    "sap.ui5": {
        "flexEnabled": false,
        "rootView": {
            "viewName": "Supplier.SecondApp.view.App",
            "type": "XML",
            "async": true,
            "id": "App"
        },
        "dependencies": {
            "minUI5Version": "1.65.6",
            "libs": {
                "sap.ui.layout": {},
                "sap.ui.core": {},
                "sap.m": {},
                "sap.f": {},
                "sap.uxap": {}
            }
        },
        "contentDensities": {
            "compact": true,
            "cozy": true
        },
        "models": {
            "i18n": {
                "type": "sap.ui.model.resource.ResourceModel",
                "settings": {
                    "bundleName": "Supplier.SecondApp.i18n.i18n"
                }
            },
            "": {
                "type": "sap.ui.model.odata.v2.ODataModel",
                "settings": {
                    "defaultOperationMode": "Server",
                    "defaultBindingMode": "OneWay",
                    "defaultCountMode": "Request"
                },
                "dataSource": "Odata.svc",
                "preload": true
            }
        },
        "resources": {
            "css": [
                {
                    "uri": "css/style.css"
                }
            ]
        },
        "routing": {
            "config": {
                "routerClass": "sap.m.routing.Router",
                "viewType": "XML",
                "async": true,
                "viewPath": "Supplier.SecondApp.view",
                "controlAggregation": "pages",
                "controlId": "app",
                "clearControlAggregation": false
            },
            "routes": [
                {
                    "name": "RouteApp",
                    "pattern": "RouteApp",
                    "target": [
                        "TargetApp"
                    ]
                }
            ],
            "targets": {
                "TargetApp": {
                    "viewType": "XML",
                    "transition": "slide",
                    "clearControlAggregation": false,
                    "viewId": "App",
                    "viewName": "App"
                },
                "Supplier": {
                    "viewType": "XML",
                    "viewName": "Supplier"
                },
```

```
                "SupplierDetail": {
                    "viewType": "XML",
                    "viewName": "SupplierDetail"
                }
            }
        }
    }
}
```

Lets add fields to our SupplierDetail.view.xml.

```
<mvc:View   xmlns="sap.uxap"
    xmlns:m="sap.m"
    xmlns:f="sap.f"
    xmlns:form="sap.ui.layout.form"
    xmlns:mvc="sap.ui.core.mvc">
    <ObjectPageLayout
        id="ObjectPageLayout"
        showTitleInHeaderContent="true"
        alwaysShowContentHeader="false"
        preserveHeaderStateOnScroll="false"
        headerContentPinnable="true"
        isChildPage="true"
        upperCaseAnchorBar="false">
        <headerTitle>
            <ObjectPageDynamicHeaderTitle>
                <actions>
                    <m:ToggleButton
                        text="Edit"
                        type="Emphasized"/>
                    <m:Button
                        text="Delete"
                        type="Transparent"/>
                    <m:Button
                        text="Copy"
                        type="Transparent"/>
                    <m:Button
                        icon="sap-icon://action"
                        type="Transparent"/>
                </actions>
            </ObjectPageDynamicHeaderTitle>
        </headerTitle>

        <headerContent>
            <m:FlexBox wrap="Wrap" fitContainer="true" alignItems="Stretch">

                <m:VBox justifyContent="Center" class="sapUiSmallMarginEnd">
                    <m:Label text="--SupplierName---"/>
                </m:VBox>

            </m:FlexBox>
        </headerContent>
        <sections>
            <ObjectPageSection title="General Information">
                <subSections>
                    <ObjectPageSubSection>
                        <blocks>
                            <form:SimpleForm
                                maxContainerCols="2"
                                editable="false"
                                layout="ResponsiveGridLayout"
                                labelSpanL="12"
                                labelSpanM="12"
                                emptySpanL="0"
                                emptySpanM="0"
                                columnsL="1"
                                columnsM="1">
                                <form:content>
                                <m:Label text="Street"/>
                                    <m:Text text="{Street}"/>
                                    <m:Label text="City"/>
                                    <m:Text text="{City}"/>
                                    <m:Label text="State"/>
                                    <m:Text text="{State}"/>
                                </form:content>
                            </form:SimpleForm>
                        </blocks>
                    </ObjectPageSubSection>
                </subSections>
            </ObjectPageSection>

            <ObjectPageSection title="Products">
                <subSections>
                    <ObjectPageSubSection>
                        <blocks>
                            <m:Table
                                id="productsTable"
```

```
                                items="{path : '/Suppliers(1)/Products'}">
                                <m:columns>
                                    <m:Column/>
                                </m:columns>
                                <m:items>
                                    <m:ColumnListItem type="Navigation">
                                        <m:cells>
                                            <m:ObjectIdentifier text="{Name}"/>
                                        </m:cells>
                                    </m:ColumnListItem>
                                </m:items>
                            </m:Table>
                        </blocks>
                    </ObjectPageSubSection>
                </subSections>
            </ObjectPageSection>
        </sections>
    </ObjectPageLayout>

</mvc:View>
```

Lets add a save button if we edit etc. we will use statechange event to commuicate

```
<mvc:View controllerName="Supplier.SecondApp.controller.App" xmlns:mvc="sap.ui.core.mvc" displayBlock="true" xmlns="sap.m"
xmlns:f="sap.f">
    <App id="app">
        <f:FlexibleColumnLayout id="flexibleColumnLayout" backgroundDesign="Translucent" layout="OneColumn"
stateChange="onStateChanged" >
            <f:beginColumnPages>
                <mvc:XMLView id="beginView" viewName="Supplier.SecondApp.view.Supplier"/>
            </f:beginColumnPages>
            <f:midColumnPages>
            <mvc:XMLView id="detailView" viewName="Supplier.SecondApp.view.SupplierDetail"/>
        </f:midColumnPages>
        </f:FlexibleColumnLayout>
    </App>
</mvc:View>
```

Add footer to SupplyDetail.view.xml, edit button press and controller.

```
<mvc:View   controllerName="Supplier.SecondApp.controller.SupplierDetail" xmlns="sap.uxap"
    xmlns:m="sap.m"
    xmlns:f="sap.f"
    xmlns:form="sap.ui.layout.form"
    xmlns:mvc="sap.ui.core.mvc">
    <ObjectPageLayout
        id="ObjectPageLayout"
        showTitleInHeaderContent="true"
        alwaysShowContentHeader="false"
        preserveHeaderStateOnScroll="false"
        headerContentPinnable="true"
        isChildPage="true"
        upperCaseAnchorBar="false">
        <headerTitle>
            <ObjectPageDynamicHeaderTitle>
                <actions>
                    <m:ToggleButton
                        text="Edit"
                        type="Emphasized"
                        press=".onEditToggleButtonPress"/>
                    <m:Button
                        text="Delete"
                        type="Transparent"/>
                    <m:Button
                        text="Copy"
                        type="Transparent"/>
                    <m:Button
                        icon="sap-icon://action"
                        type="Transparent"/>
                </actions>
            </ObjectPageDynamicHeaderTitle>
        </headerTitle>

        <headerContent>
            <m:FlexBox wrap="Wrap" fitContainer="true" alignItems="Stretch">

                <m:VBox justifyContent="Center" class="sapUiSmallMarginEnd">
                    <m:Label text="--SupplierName---"/>
                </m:VBox>

            </m:FlexBox>
        </headerContent>
        <sections>
            <ObjectPageSection title="General Information">
                <subSections>
                    <ObjectPageSubSection>
                        <blocks>
```

```
                        <form:SimpleForm
                            maxContainerCols="2"
                            editable="false"
                            layout="ResponsiveGridLayout"
                            labelSpanL="12"
                            labelSpanM="12"
                            emptySpanL="0"
                            emptySpanM="0"
                            columnsL="1"
                            columnsM="1">
                            <form:content>
                            <m:Label text="Street"/>
                                <m:Text text="{Street}"/>
                                <m:Label text="City"/>
                                <m:Text text="{City}"/>
                                <m:Label text="State"/>
                                <m:Text text="{State}"/>
                            </form:content>
                        </form:SimpleForm>
                    </blocks>
                </ObjectPageSubSection>
            </subSections>
        </ObjectPageSection>

        <ObjectPageSection title="Products">
            <subSections>
                <ObjectPageSubSection>
                    <blocks>
                        <m:Table
                            id="productsTable"
                            items="{path : '/Suppliers(1)/Products'}">
                            <m:columns>
                                <m:Column/>
                            </m:columns>
                            <m:items>
                                <m:ColumnListItem type="Navigation">
                                    <m:cells>
                                        <m:ObjectIdentifier text="{Name}"/>
                                    </m:cells>
                                </m:ColumnListItem>
                            </m:items>
                        </m:Table>
                    </blocks>
                </ObjectPageSubSection>
            </subSections>
        </ObjectPageSection>
    </sections>
    <footer>
        <m:OverflowToolbar>
            <m:ToolbarSpacer/>
            <m:Button type="Accept" text="Save"/>
            <m:Button type="Reject" text="Cancel"/>
        </m:OverflowToolbar>
    </footer>
</ObjectPageLayout>

</mvc:View>
```

In SupplierDetail controller if we click edit we read the state passed earlier and react.

```
sap.ui.define([
    "sap/ui/core/mvc/Controller"
], function (Controller) {
    "use strict";

    return Controller.extend("Supplier.SecondApp.controller.SupplierDetail", {

        /**
         * Called when a controller is instantiated and its View controls (if available) are already created.
         * Can be used to modify the View before it is displayed, to bind event handlers and do other one-time initialization.
         * @memberOf Supplier.SecondApp.view.SupplierDetail
         */
        onInit: function () {

        },
            onEditToggleButtonPress: function() {
            var oObjectPage = this.getView().byId("ObjectPageLayout"),
                bCurrentShowFooterState = oObjectPage.getShowFooter();

            oObjectPage.setShowFooter(!bCurrentShowFooterState);
        }
        /**
         * Similar to onAfterRendering, but this hook is invoked before the controller's View is re-rendered
         * (NOT before the first rendering! onInit() is used for that one!).
         * @memberOf Supplier.SecondApp.view.SupplierDetail
         */
        //  onBeforeRendering: function() {
        //
        //  },

        /**
         * Called when the View has been rendered (so its HTML is part of the document). Post-rendering manipulations of the HTML
could be done here.
         * This hook is the same one that SAPUI5 controls get after being rendered.
         * @memberOf Supplier.SecondApp.view.SupplierDetail
         */
        //  onAfterRendering: function() {
        //
        //  },

        /**
         * Called when the Controller is destroyed. Use this one to free resources and finalize activities.
         * @memberOf Supplier.SecondApp.view.SupplierDetail
         */
        //  onExit: function() {
        //
        //  }

    });

});
```