**PORTLET ASSIGNMENT**

**Question: Create and Deploy all Types of Portlet Script.**

**1. Simple From**

**2. Inline HTML**

**3. Links and Indents**

**4. Simple List**

**Answers:**

1. **Simple Form:** This type of portlet contains a form that allows users to input data, which can then be processed or submitted. It typically includes form elements like text fields, checkboxes, and buttons.

Here I have used serverWidget module as I have to Create a Custom Form with a Submit Button, Fields on the UI. As we all know there is only one entry point in Portlet. I.e. render.

I used title API to set the title of my custom portlet and addField API to add a custom field in it.

***Code:***

/\*\*

\* @NApiVersion 2.1

\* @NScriptType Portlet

\*/

define(['N/ui/serverWidget'],

/\*\*

\* @param{serverWidget} serverWidget

\*/

function (serverWidget) {

//Simple Form, Inline HTML, Links and Indents, Simple Listconst render = (params) => {

var myPortletObj = params.portlet

myPortletObj.title = 'My Portlet'

//Simple Form

myPortletObj.addField({

id: 'custpage\_name',

label: 'Enter Name',

type: serverWidget.FieldType.TEXT

})

myPortletObj.setSubmitButton({

url: '<https://hsg15.github.io/lucky/>',

label: 'Save'

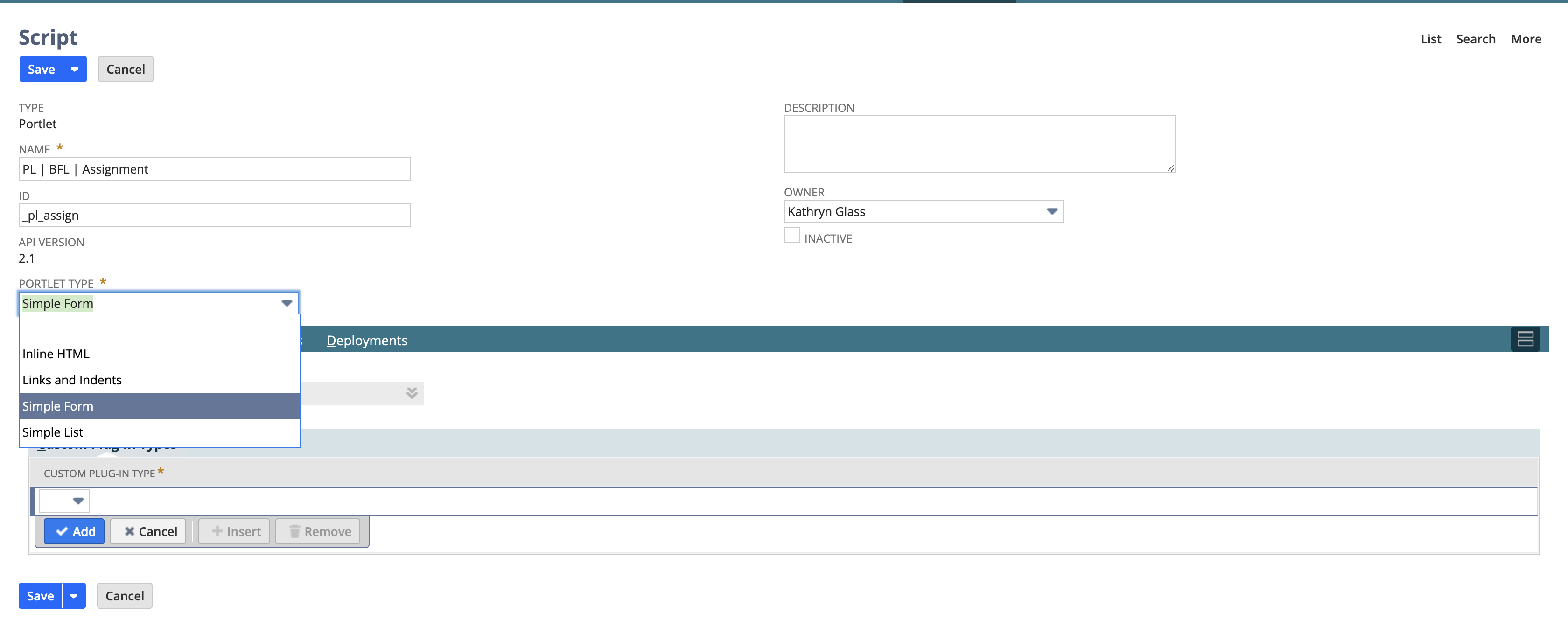
})

}

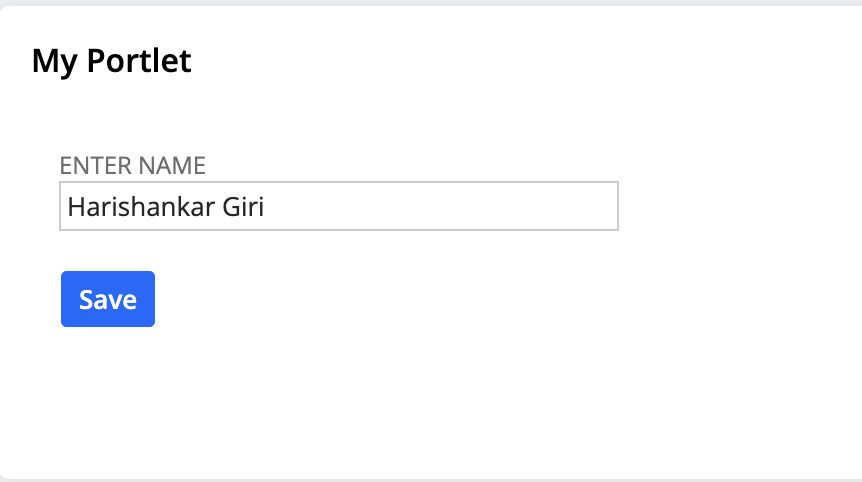
return {render}

});

While uploading the code by navigating to Customization -> Scripting -> Scripts -> New, make sure to select the PORTLET TYPE.



***Output:***



1. **Inline HTML**: An **Inline HTML** portlet displays custom HTML content directly within the portlet. It allows to modify the portlrt by adding formatted text, images, or other HTML components for customization.

***Code:***

const render = (params) => {

var myPortletObj = params.portlet

myPortletObj.title = 'My Portlet'

// Inline HTML

var inlineHTMLField = myPortletObj.addField({

id: 'divfield',

type: 'inlinehtml',

label: 'Test inline HTML'

});

inlineHTMLField.defaultValue =

'<div id="divfield\_elem" style="border: 1px dotted black; height:150px; width:150px;">' +

'<img src="https://encrypted-tbn0.gstatic.com/images?q=tbn:ANd9GcQ0qjaY1Si6xgGYzbsrgMCnGdumxOmYMCwhzw&s" ' +

'alt="Sample Image" style="height: 100%; width: 100%;" />' +

'</div>' +

'<p style="margin-top: 10px; text-align:center;">Harishankar Giri</p>';

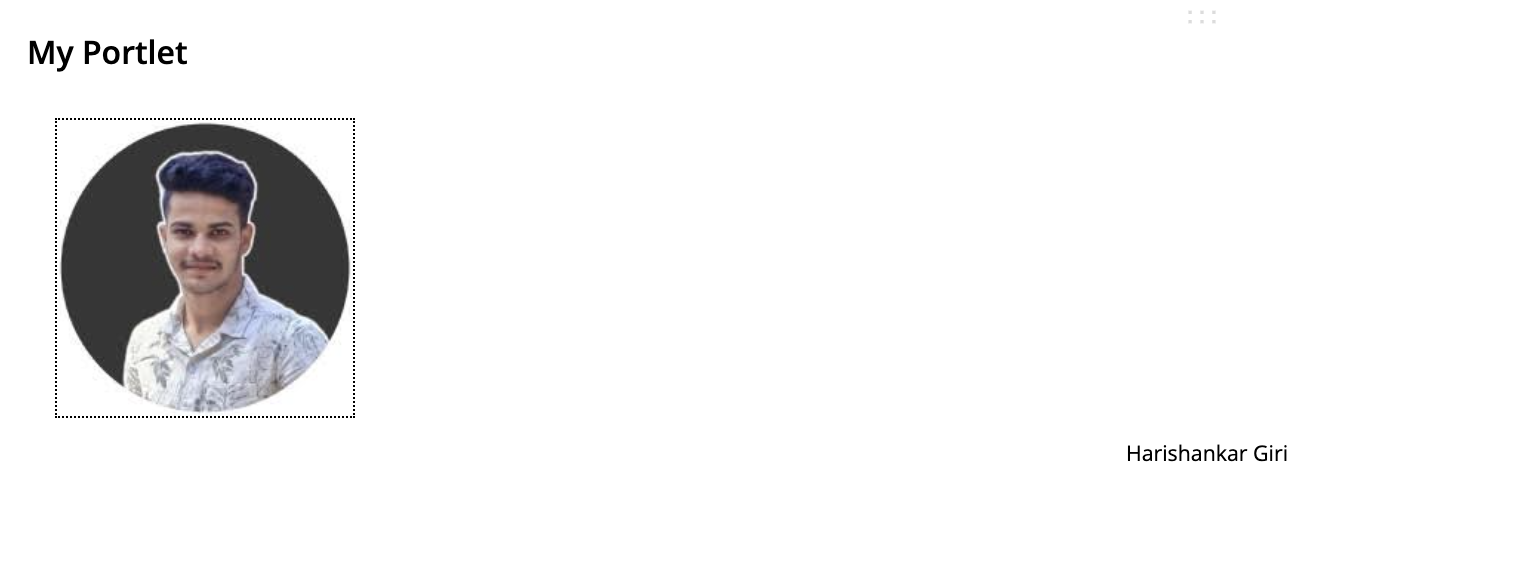
myPortletObj.addLine({

text: 'Harishankar Giri'

})

}

***Output:***



1. **Links and Indents**: This portlet presents a structured list of hyperlinks. So that if we click on the links we will simply redirect to the website that are given in the href section in html. Sometimes client need this feature to quickly access their daily needs. It’s useful for navigation menus or lists of resources.

***Code:***

const render = (params) => {

var myPortletObj = params.portlet

myPortletObj.title = 'Links & Indents'

// Links and Indents

var inlineHTMLField = myPortletObj.addField({

id: 'linksfield',

type: serverWidget.FieldType.INLINEHTML,

label: 'Links and Indents'

});

var linksHTML =

'<div>' +

'<a href="https://hsg15.github.io/lucky/" target="\_blank">My Portfolio</a><br><br>' +

'<a href="https://www.linkedin.com/in/hsg15/" target="\_blank">My LinkedIn</a><br><br>' +

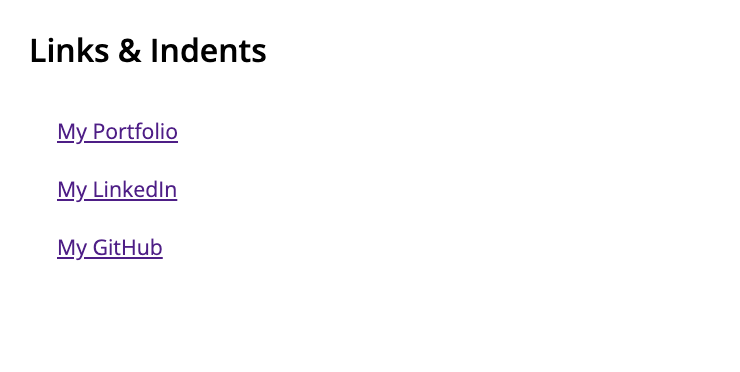
'<a href="https://github.com/HSG15" target="\_blank">My GitHub</a><br>' +

'</div>';

inlineHTMLField.defaultValue = linksHTML;

}

***Output:***



1. **Simple List**: This portlet shows a basic, unordered or ordered list of items that we oftenly used to create list in HTML, such as text entries or links. It is commonly used to display simple, structured data in a structured manner.

***Code:***

/\*\*

\* @NApiVersion 2.1

\* @NScriptType Portlet

\*/

define(['N/ui/serverWidget', 'N/search'],

/\*\*

\* @param{serverWidget} serverWidget

\*/

function (serverWidget, search) {

const render = (params) => {

var myPortletObj = params.portlet

myPortletObj.title = 'List of emails'

// Simple List

myPortletObj.addColumn({

id: 'internalid',

type: 'text',

label: 'ID',

align: 'LEFT'

});

myPortletObj.addColumn({

id: 'email',

type: 'text',

label: 'E-MAIL',

align: 'LEFT'

});

var filter = search.createFilter({

name: 'email',

operator: search.Operator.ISNOTEMPTY // Mail field should not be empty

});

var customerSearch = search.create({

type: 'customer',

filters: [filter],

columns: ['internalid','email']

});

customerSearch.run().each(function (result) {

myPortletObj.addRow({

internalid: result.getValue('internalid'),

email: result.getValue('email')

})

return true

})

}

return { render: render }

})

***Output:***

