Implementation of Custom Portal

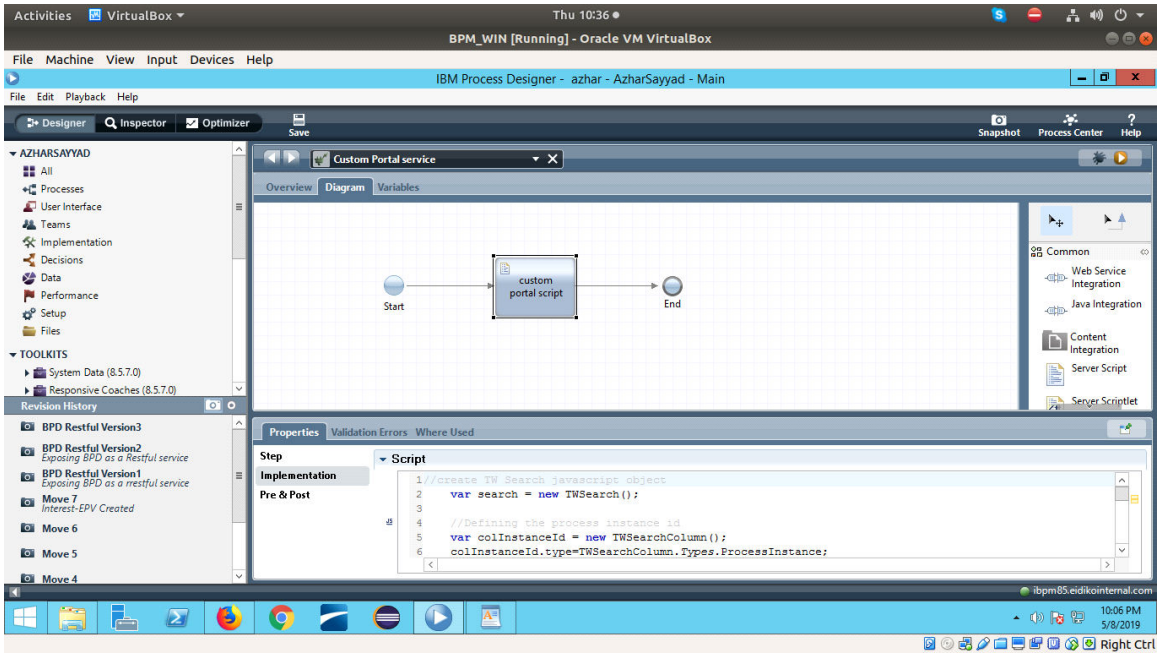
Description :

● To implement the custom portal we use TWSearch script .In order to get instance details,process details,task details ..etc.

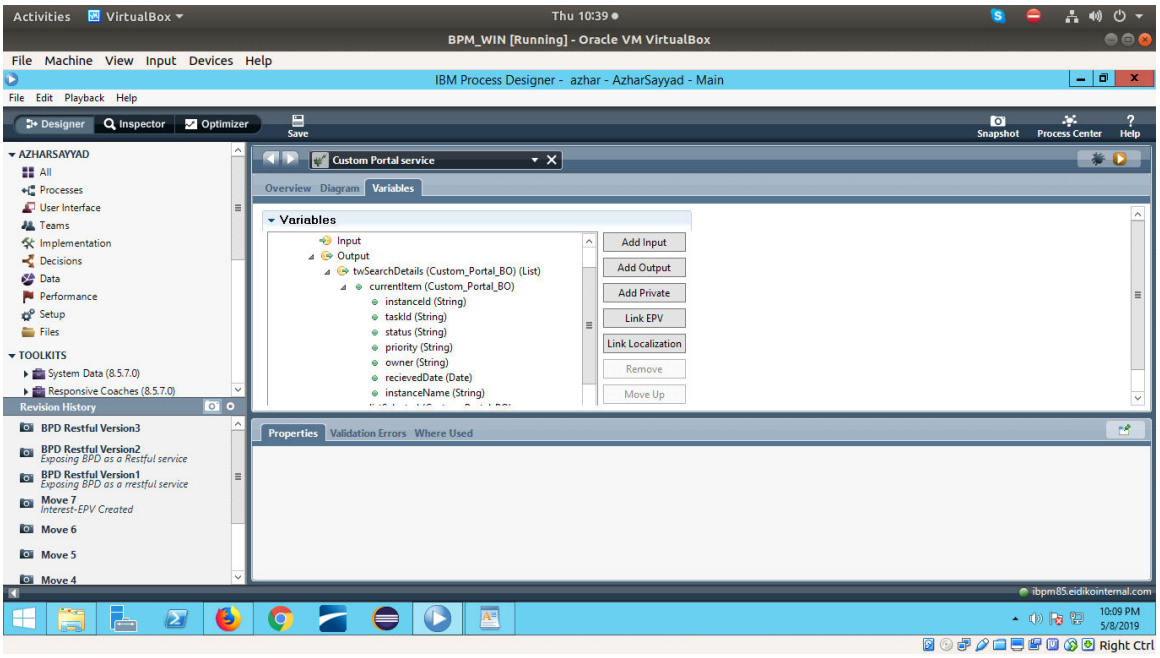
● Using TWSearch we can define which columns to retrieve, what filters to apply, how to sort and organize the results. You can also parse the results into a list of variables.

● In this concept we can retrieve data from process and task instances by using aJavaScript TWSearch object .

Create Integration or General System Service, which has server script.



In Variable section, Declare the customer variable type of Custom_Portal_BO. Here Business Object has seven parameters.



In Properties, Implementation write script using TWSearch.

Structure of a TWSearch query :

First we need to create TWSearch javascript object using below code.

var search = new TWSearch();

TWSearch query includes the following steps.

1.Define the columns of data that you want to retrieve.Each column that you want to return is defined as a TWSearchColumn object. The following code sample defines a column that returns the identifier of a process instance.

```
var colInstanceStatus=new TWSearchColumn();
colInstanceStatus.type=TWSearchColumn.Types.ProcessInstance;
colInstanceStatus.name=TWSearchColumn.ProcessInstanceColumns.Status;


 //Defining the task id
 var colTaskId=new TWSearchColumn();
 colTaskId.type=TWSearchColumn.Types.Task;
 colTaskId.name=TWSearchColumn.TaskColumns.ID;
```

```
//Defining Task Status
var colTaskStatus=new TWSearchColumn();
colTaskStatus.type=TWSearchColumn.Types.Task;
colTaskStatus.name=TWSearchColumn.TaskColumns.Status;

var colTaskPriority=new TWSearchColumn();
colTaskPriority.type=TWSearchColumn.Types.Task;
colTaskPriority.name=TWSearchColumn.TaskColumns.Priority;

var colTaskOwner=new TWSearchColumn();
colTaskOwner.type=TWSearchColumn.Types.Task;
colTaskOwner.name=TWSearchColumn.TaskColumns.AssignedToUser;

var colTaskRecievedDate=new TWSearchColumn();
colTaskRecievedDate.type=TWSearchColumn.Types.Task;
colTaskRecievedDate.name=TWSearchColumn.TaskColumns.ReceivedDate;

search.columns=newArray(colInstanceId,colInstanceName,colTaskId,colTaskStatus,co
lTaskPriority,colTaskOwner,colTaskRecievedDate);
```

2.Define any filters that you want to apply to a column. You can apply filters to anycolumn by defining conditions. You define each condition as a TWSearchConditionobject. For example, the following code sample defines a column that retrieves the name of a process instance and then narrows down the results to process instance names that start with Service Order Fulfillment.

```
//Defining Filter codition
 var condInstanceName=new TWSearchCondition;
 condInstanceName.column=colInstanceName;
 condInstanceName.operator=TWSearchCondition.Operations.StartsWith;
 condInstanceName.value="Testing";
```

```
 var condInstanceStatus=new TWSearchCondition;
 condInstanceStatus.column=colInstanceStatus;
 condInstanceStatus.operator=TWSearchCondition.Operations.Equals;
 condInstanceStatus.value="Active";

search.conditions=new Array(condInstanceName,condInstanceStatus);
```

3.Define the sort order for the returned records.You can define ordering criteria for any column. Each ordering is defined as a TWSearchOrdering object. For example, the following code sample lists the retrieved process instance identifiers in ascending order.

```
//Defining the sorting order
 var orderInstanceId=new TWSearchOrdering();
 orderInstanceId.column=colInstanceId;
 orderInstanceId.Order=TWSearchOrdering.Orders.Ascending;
 search.orderBy=new Array(orderInstanceId);
```

4.Define how results are organized. You must also specify whether the results should be organized by process instance or by task. To do so, you use the TWSearch.organizedBy object. The behavior is the same as that of saved searches in Process Portal. If you choose to organize by process instance, you get only one result per process instance. The following code sample organizes search results by task.

search.organizedBy = TWSearch.OrganizeByTypes.Task;

5.Execute the search and parse the results into a list of complex variables.After you have specified the columns, conditions, ordering, and organization for the search, you can execute it to retrieve an array of JavaScript rows.

var results = search.execute();

The execute method returns TWSearchResults objects.

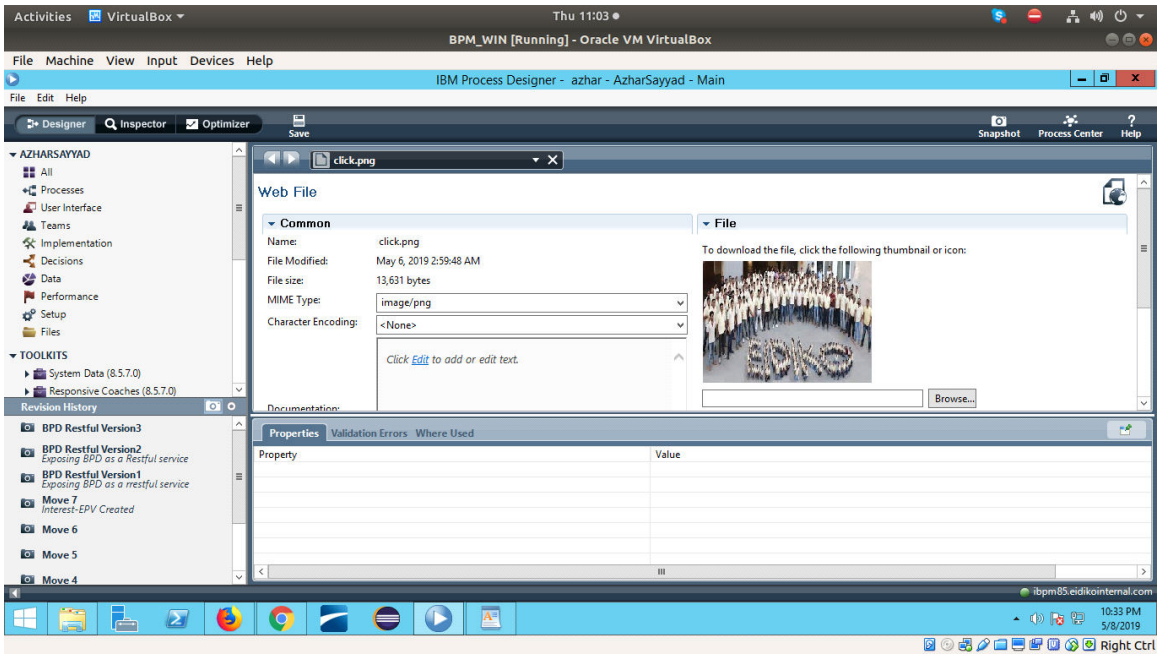
If you want to use these results outside of your script block, you must parse them and initialize equivalent variables.

The following code parses the native JavaScript array to create an array of custom variables.

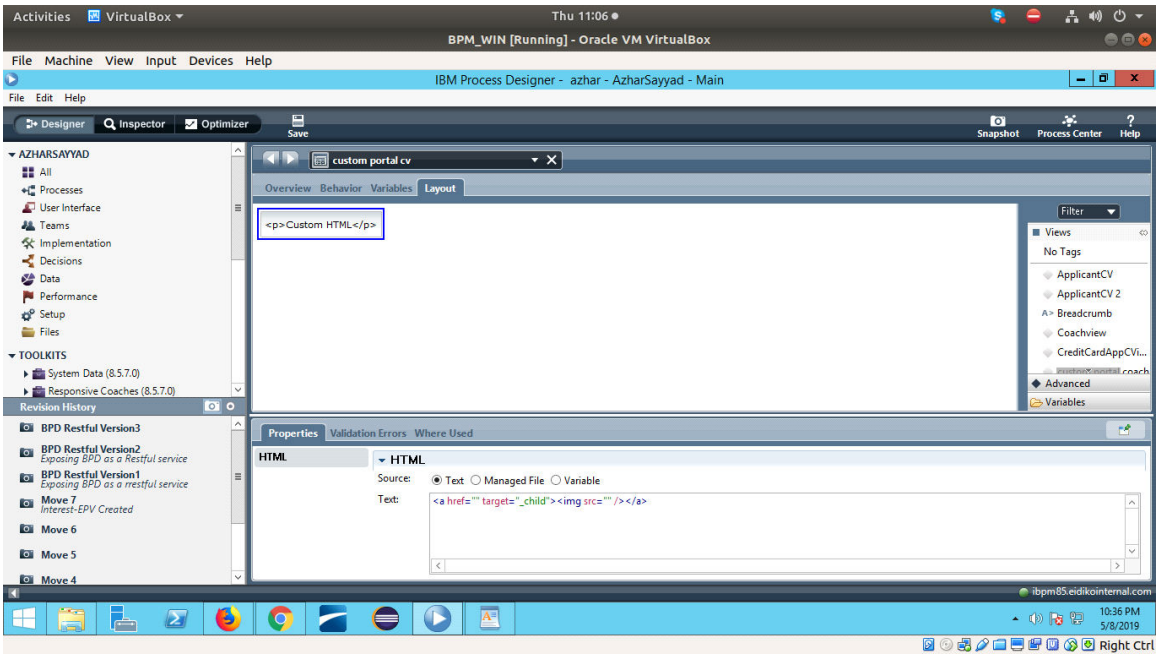
```
tw.local.twSearchDetails=new tw.object.listOf.Custom_Portal_BO();
 for(var i=0;i<results.rows.length;i++){
```

```
        var row=results.rows[i];
        tw.local.twSearchDetails[i] = new tw.object.Custom_Portal_BO();
        tw.local.twSearchDetails[i].instanceId=row.values[0].toString();
        tw.local.twSearchDetails[i].instanceName=row.values[1];
        tw.local.twSearchDetails[i].taskId=row.values[2].toString();
        tw.local.twSearchDetails[i].status=row.values[3];
        tw.local.twSearchDetails[i].priority=row.values[4];
        tw.local.twSearchDetails[i].owner=row.values[5];
        if(row.values[6]!=null){
        tw.local.twSearchDetails[i].recievedDate=row.values[6];
       }
    }
```

Create Web File, browse image from local machine.



Create Coach View drag Custom HTML from pallet add text in properties of HTML section.



In Coach View behaviour section -> Event handler -> Load.

add below code.

```
var role = this.context.options.role ? this.context.options.role.get("value") : "";

var step = this.context.options.step ? this.context.options.step.get("value") : "";

var taskurl="teamworks/process.lsw?zWorkflowState=1&zResumable=false&zForceNew=true&zTaskId=";

var link=dojo.query("a",this.context.element)[0];

var Img=dojo.query("img",this.context.element)[0];

var id=this.context.binding.get("value");

var url=taskurl+id.toString();

var imgUrl="";

//console.log("role"+role);

if(role == "SSO" || role == "RM")

{

imgUrl=com_ibm_bpm_coach.getManagedAssetUrl("click.png",com_ibm_bpm_coach.assetType_WEB);

}

else
```

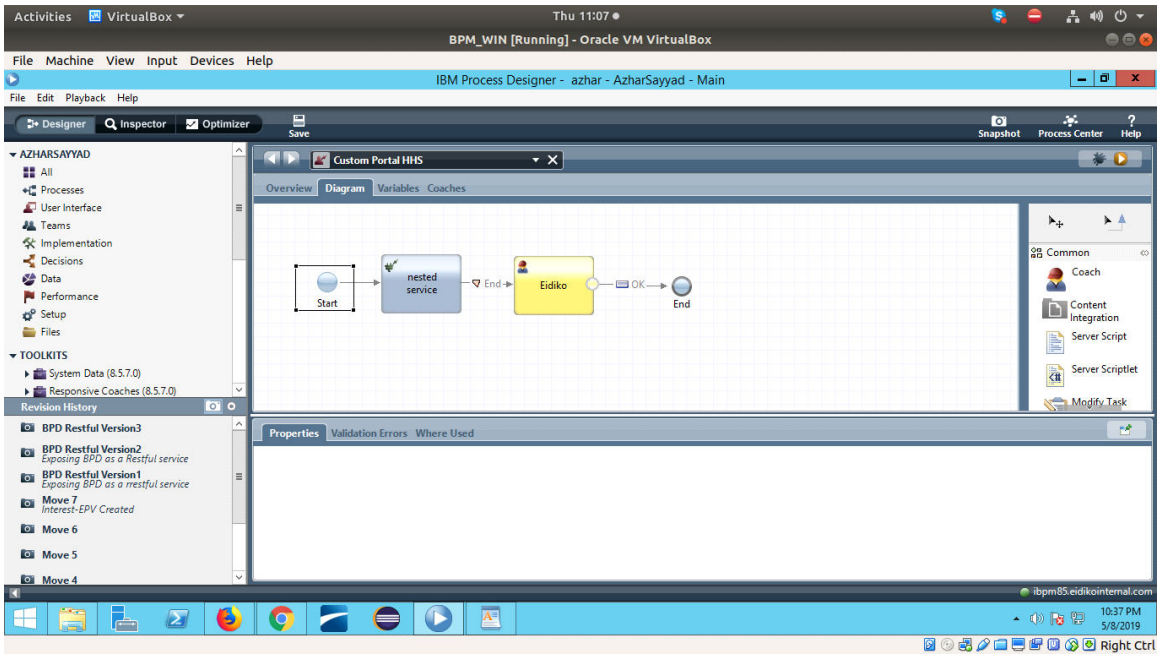
```
{

imgUrl=com_ibm_bpm_coach.getManagedAssetUrl("click.png",com_ibm_bpm_coach.assetT
ype_WEB);

}

//FLEX_FAILURE CONDITION

if(step == "FLEX_FAILURE"){

imgUrl=com_ibm_bpm_coach.getManagedAssetUrl("play.jpg",com_ibm_bpm_coach.assetTy
pe_WEB);

}

dojo.setAttr(link, "href", url);

dojo.setAttr(Img, "src", imgUrl);
```

In Coach View behaviour section -> Event handler -> unload.

add below code.

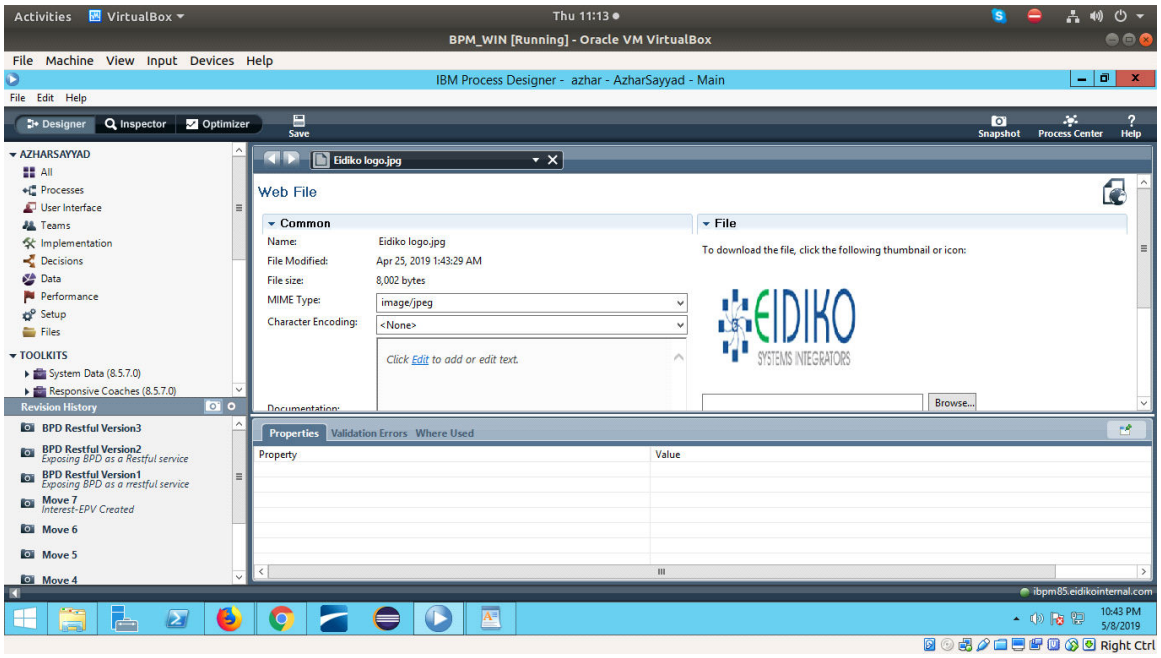
```
this.context.trigger();
```

Create Human Service, which has Nested Service and Coach.

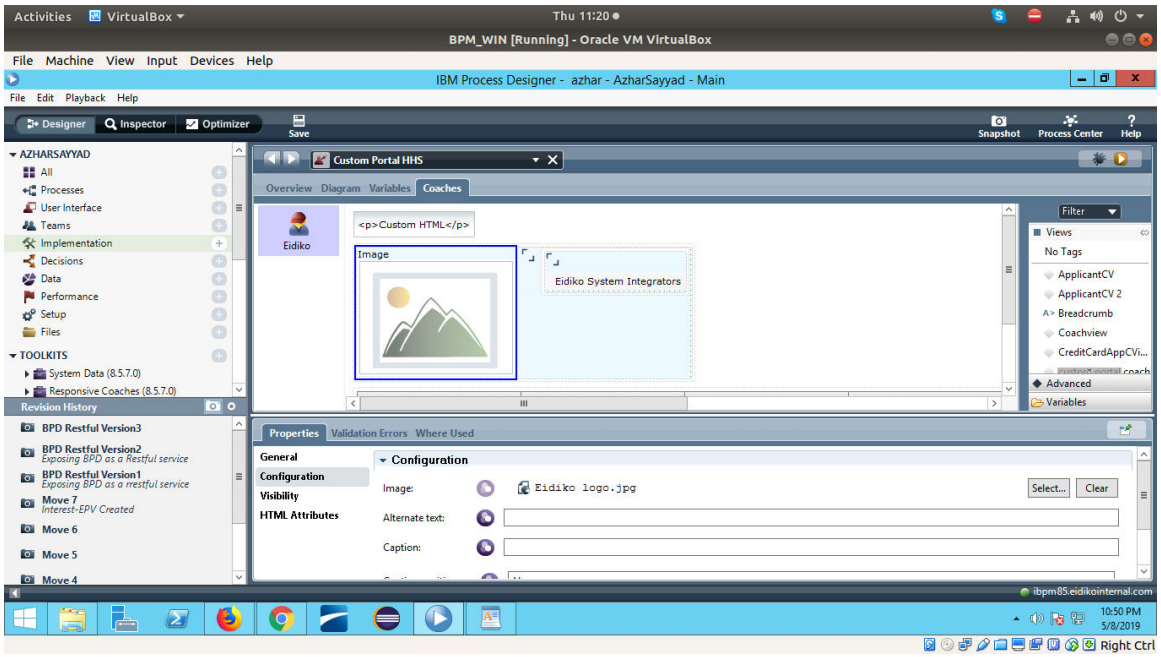


add company logo and company name

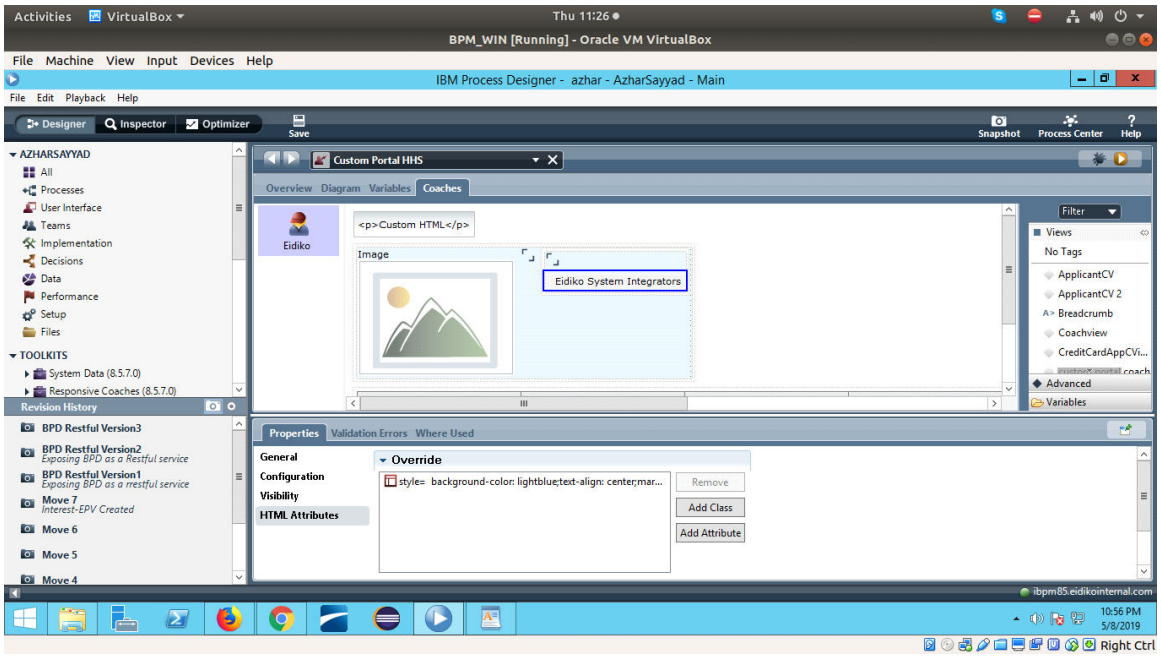
a) create web file , browse image from local machine



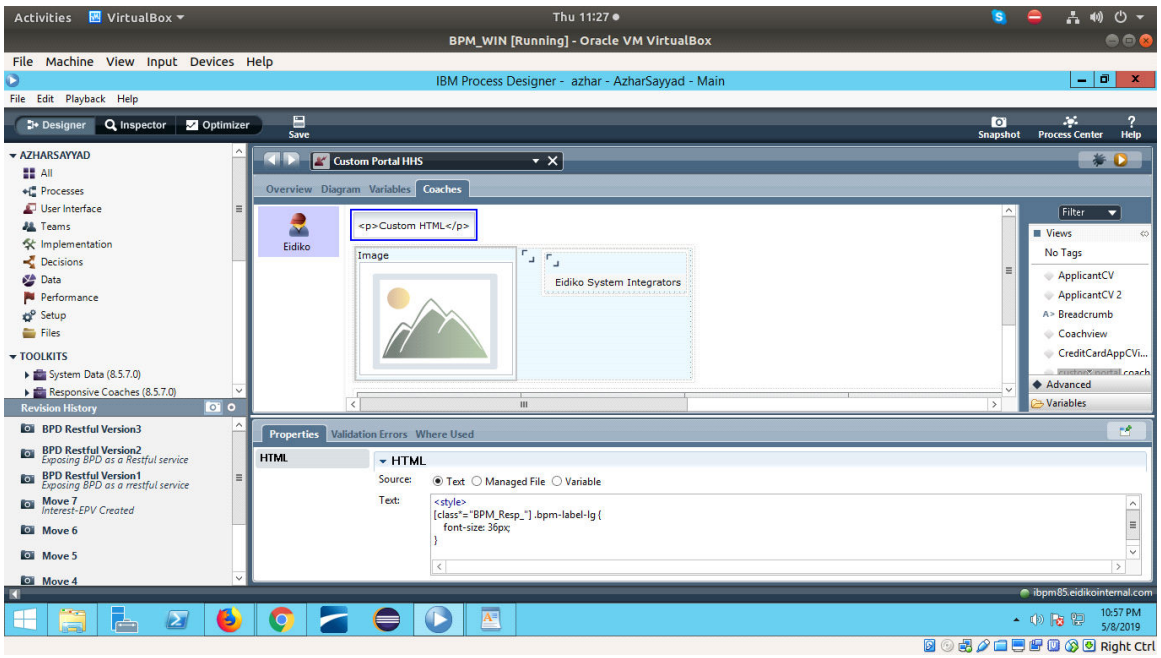
b) drag and drop horizontal section.inside horizontal section drag and drop image controller and in configuration option select image and attached eidiko logo image.



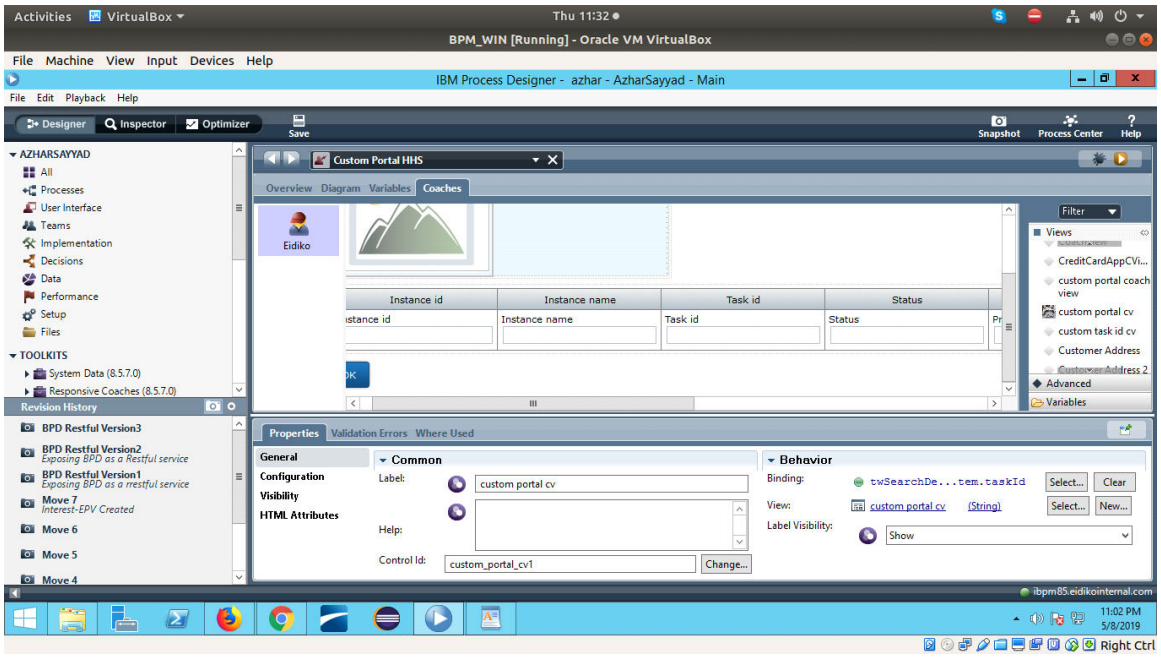
c) darg and drop output text controller inside horizontal section and give the name as EIDIKO SYSTEM INTEGRATORS.and do some css for styles



d) To increse the size of company name(EIDIKO SYSTEM INTEGRATORS) add custom html in coach



add service to the nested service and drag and drop table, variables from pallet in coach and drag coach view in coach.



while running Human Service it will get the following screen as shown below.