sap.ui.define([

"sap/ui/core/mvc/Controller",

"sap/ui/model/json/JSONModel",

"sap/ui/model/odata/v2/ODataModel",

"sap/ui/model/Filter",

"sap/ui/model/FilterOperator",

"sap/m/MessageToast",

"sap/m/MessagePopover",

"sap/m/MessageItem",

"sap/m/MessageBox",

"sap/m/MessageView",

"sap/ui/core/BusyIndicator"

], function (Controller, JSONModel, ODataModel, Filter, FilterOperator, MessageToast, MessagePopover, MessageItem, MessageBox,

MessageView, BusyIndicator) {

"use strict";

// Constants pointing to the services used for saving and posting the invoice

var SAVE\_INVOICE\_SERVICE = "/HDB\_SCP/coms/deloitte/apps/invoiceprocessor/services/saveInvoice/saveInvoice.xsjs";

var POST\_INVOICE\_SERVICE = "/HDB\_SCP/coms/deloitte/apps/invoiceprocessor/services/postInvoice/postInvoice.xsjs";

var CAPTURE\_REJECTCOMMENT\_SERVICE = "/HDB\_SCP/coms/deloitte/apps/invoiceprocessor/services/workflowResponse/workflowResponse.xsjs";

var POST\_PENDING\_APPROVAL\_STATUS\_onSubmit = "/HDB\_SCP/coms/deloitte/apps/invoiceprocessor/services/workflowResponse/statusPendingApproval.xsjs";

var SEARCH\_HELP\_SERVICE = "/HDB\_SCP/coms/deloitte/apps/invoiceprocessor/services/searchHelpValues/searchHelpValues.xsjs";

var APPROVER\_ROLE = "approver";

// var sTaskId = "approver";

var oMessageTemplate = new MessageItem({

type: '{type}',

title: '{title}',

subtitle: '{description}',

description: '{description}'

});

var \_hasExceptions = false;

var \_hasNonDisplayExceptions=false;

var \_formData = null;

var oMessagePopover = new MessagePopover({

items: {

path: '/',

template: oMessageTemplate

}

});

return Controller.extend("com.deloitte.scp.app.dips.controller.InvoiceDetail", {

inputId: '',

\_valueHelpDialogs:{},

\_currentInvoiceItemSelected:null,

/\*\*

\* 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 com.deloitte.scp.app.InvoiceProcessor.view.InvoiceDetail

\*/

onInit: function () {

var oRouter = sap.ui.core.UIComponent.getRouterFor(this);

oRouter.getRoute("toInvoiceDetail").attachMatched(this.\_onHandleNavFromList, this);

var invoiceDetails = this.getView().getModel("invoiceDetails");

// this.byId("btnSubmit").setEnabled(false);

// var fileName = "{invoiceDetails>/invoiceFileName}";

// var fileExtension = fileName.replace(/^.\*\./, '');

//Hiding Submit,navigation button for exception roles,approvers

var btnSubmit=this.byId("btnSubmit");

var btnBack=this.byId("btnBack");

var btnBackTop=this.byId("btnBackTop");

if(this.getCurrentUserRole()!==''){

btnSubmit.setVisible(false);

btnBack.setVisible(false);

btnBackTop.setVisible(false);

}

// setTimeout(function(){

// btnSubmit.setEnabled(true);

// },10000);

},

invertBoolVal:function(value){

if(value){

if(value.toString()==="true"){

return false;

}

return true;

}

return true;

},

toTitleCalse:function(value){

if(value){

var str = value.toLowerCase().split('');

str[0] = str[0].toUpperCase();

return str.join('');

}

return 'Success';

},

/\*\*

\* Called while navigation from the list to the object page has been completed.

\* This handler makes retrieves details of the individual invoice and populates it in the respective fields

\*\*/

\_onHandleNavFromList: function (oEvent) {

this.byId("btnCancel").setVisible(false);

this.byId("btnSave").setVisible(true);

this.byId("btnApprove").setVisible(false);

this.byId("btnReject").setVisible(false);

// Show the busy indicator while retrieving and loading the invoice details

BusyIndicator.show(0);

var uriParameters = oEvent.getParameter("arguments");

// Get the PONumber and make poItemTab or nonpoItemtab visible depending on selection of the invoice from the list - Initial load of Detail view

// var poNumber = sap.ui.getCore().purchaseOrder;

this.\_setPoNonPoTabVisibility(false);

// }

// Get the invoiceGUID and invoiceID transferred from the list page

var invoiceGUID = uriParameters.invoiceGUID;

sap.ui.getCore().invoiceGUIDFromList = uriParameters.invoiceGUID;

var invoiceStatus = sap.ui.getCore().invoiceStatus;

if (invoiceStatus == "In\_Progress") {

this.byId("btnSave").setEnabled(false);

} else

this.byId("btnSave").setEnabled(true);

if (invoiceStatus == "Pending\_Approval") {

this.\_allowInvoiceModification(false);

}

// Get the model of the current view and generate an odata model using its service url

var oInvoiceDetailsModel = this.getOwnerComponent().getModel();

var oODataModel = new ODataModel(oInvoiceDetailsModel.sServiceUrl);//sServiceUrl: "/HDB\_SCP/com/deloitte/apps/invoiceprocessor/InvoiceServices.xsodata"

var currentViewController = this;

var invoiceDetailsJSON;

oODataModel.read("/Invoice(invoiceGUID='" + invoiceGUID + "')", {

// Get the response formatted as a JSON response and have the items association expanded

urlParameters: {

"$format": "json",

"$expand": "items,displayExceptions"

},

// make async false to wait for ajax response

async: false,

success: function (data, response)

{

// Parse the JSON response and generate a corresponding JSON model

invoiceDetailsJSON = JSON.parse(JSON.stringify(data));

sap.ui.getCore().invoiceStatus=invoiceDetailsJSON.invoiceStatus;

//payment terms f4 help data retrieval

var variablex =[];

var counter = 1;

invoiceDetailsJSON.items.results.forEach(function (invoiceDetailsnonPoItem) {

var invoiceItem = {};

invoiceItem.invoiceItemIndex = counter++;

invoiceItem.glAccount = invoiceDetailsnonPoItem.glAccount || "";

invoiceItem.glDescription = invoiceDetailsnonPoItem.glDescription || "";

invoiceItem.costCenter = invoiceDetailsnonPoItem.costCenter || "";

invoiceItem.wbsElement = invoiceDetailsnonPoItem.wbsElement || "";

invoiceItem.internalOrder = invoiceDetailsnonPoItem.internalOrder || "";

//added new after comparing with old version

invoiceItem.itemPrice = invoiceDetailsnonPoItem.itemPrice || "";

//added new after comparing with old version

invoiceItem.netAmount = invoiceDetailsnonPoItem.netAmount || "";

invoiceItem.totaltax = invoiceDetailsnonPoItem.totaltax || "";

variablex.push(invoiceItem);

});

invoiceDetailsJSON.nonPO = invoiceDetailsJSON.items.results;

//netamount field is created here in the header ,to push the value from S/4 for netamount.In S/4 total in each line item is added up and the final cumulative total which is net amount is stored in the final lineitem znetamount passed to ui,to capture that final znetamount and push it to the header ,we have this code.Here we are checking if

//invoiceDetailsJSON.items.results?(means if we ever have line items) then get the netamount value from the last line item or 0.

invoiceDetailsJSON.netAmount=invoiceDetailsJSON.items.results?invoiceDetailsJSON.items.results[invoiceDetailsJSON.items.results.length - 1].netAmount:'0';

invoiceDetailsJSON.totalTaxAmount=invoiceDetailsJSON.items.results?invoiceDetailsJSON.items.results[invoiceDetailsJSON.items.results.length - 1].totalTaxAmount:'0';

var autoCalcVal=invoiceDetailsJSON.autoCalculateTaxIndicator;

if(autoCalcVal && autoCalcVal.toString()==="true"){

autoCalcVal=true;

}else{

autoCalcVal=false;

}

invoiceDetailsJSON.autoCalculateTaxIndicator=autoCalcVal;

invoiceDetailsJSON.displayExceptions.results.forEach(function(ex){

ex.userbypassStatus=ex.userbypassStatus?(ex.userbypassStatus.toLowerCase()==="true"?true:false):false;

ex.bypassStatus=ex.bypassStatus?(ex.bypassStatus.toLowerCase()==="true"?true:false):false;

});

\_formData = JSON.stringify(invoiceDetailsJSON);

var loadExceptionStatus = null;

invoiceDetailsJSON.userRole=currentViewController.getCurrentUserRole();

currentViewController.getSearchHelpValues().done(function(searchHelpValues){

invoiceDetailsJSON.allsearchHelpValues=searchHelpValues;

// invoiceDetailsJSON.allPaymentTerms=paymentTerms;

var invoiceDetails = new JSONModel(invoiceDetailsJSON);

invoiceDetails.setSizeLimit(10000);

//Setting PO number to check for PO and NonPO for tabs

sap.ui.getCore().poNumDetailSceenFromWF = invoiceDetailsJSON.purchaseOrder;

// Set the JSON model as the data model for the current view

currentViewController.getView().setModel(invoiceDetails, "invoiceDetails");

// Set the status indicator accordingly

currentViewController.\_setStatusIndicator(sap.ui.getCore().invoiceStatus);

//Send data to WF Obj OR Check the status for selected invoice

var contextData = {};

contextData = invoiceDetailsJSON;

sap.ui.getCore().invoiceDetailsWF = contextData; //invoiceDetailsWF - To be used in WF call

// hide the busy indicator, once the view has been updated completely

BusyIndicator.hide();

});

// If the invoice is already posted, disable making any changes to it.

// If not posted, allow changes and fetch the exceptions, if any, corresponding to the invoice

if (invoiceDetailsJSON.postedInvoice || invoiceDetailsJSON.invoiceStatus === "Pending\_Approval") {

currentViewController.\_allowInvoiceModification(false);

if (currentViewController.getCurrentUserRole() === APPROVER\_ROLE) {

currentViewController.onPageLoadForWorkflow();

}

//if the invoice is not even saved once, disable submit button

if (invoiceDetailsJSON.invoiceStatus === "Scanned"

|| invoiceDetailsJSON.invoiceStatus === "Posted"

|| invoiceDetailsJSON.invoiceStatus === "Pending\_Approval") {

currentViewController.byId("btnSubmit").setEnabled(false);

}

} else {

currentViewController.\_allowInvoiceModification(true);

loadExceptionStatus = currentViewController.\_loadExceptions(oInvoiceDetailsModel, invoiceGUID);

loadExceptionStatus.done(function () {

if (currentViewController.getCurrentUserRole() === APPROVER\_ROLE) {

currentViewController.onPageLoadForWorkflow();

}

});

}

currentViewController.\_setPoNonPoTabVisibility(false);

},

error:function(error){

if(error.statusCode==="404"){

MessageBox.error("Invoice does not exist.Invalid Invoice Id");

}else{

MessageBox.error("Cannot get invoice details");

}

}

});

},

getFileDescription: function () {

var image = document.getElementById("image11");

if ('mimeType' in image) {

MessageBox.error(image.mimeType);

} else {

MessageBox.error("Your browser doesn't support the mimeType property.");

}

},

\_setStatusIndicator: function (invoiceStatus) {

var statusIndicator = this.getView().byId("invoiceStatusIndicatorBox");

var statusIndicatorText = statusIndicator.getItems()[0];

var statusIndicatorPF = statusIndicator.getItems()[1];

var documentScanState = sap.ui.core.ValueState.None;

var inProcessState = sap.ui.core.ValueState.None;

var exceptionState = sap.ui.core.ValueState.None;

var exceptionCompletedState = sap.ui.core.ValueState.None;

var invoicePostedStatus = sap.ui.core.ValueState.None;

switch (invoiceStatus) {

case "Scanned":

statusIndicatorText.setText("Scanned");

statusIndicatorText.setIcon("sap-icon://document-text");

statusIndicatorText.setState(sap.ui.core.ValueState.Success);

documentScanState = sap.ui.core.ValueState.Success;

this.byId("btnSubmit").setEnabled(false);

break;

case "In\_Progress":

statusIndicatorText.setText("In Progress");

statusIndicatorText.setIcon("sap-icon://process");

statusIndicatorText.setState(sap.ui.core.ValueState.Warning);

documentScanState = sap.ui.core.ValueState.Success;

inProcessState = sap.ui.core.ValueState.Warning;

this.byId("btnSave").setEnabled(true);

this.byId("btnSubmit").setEnabled(false);

break;

case "In\_Exception":

statusIndicatorText.setText("Exception");

statusIndicatorText.setIcon("sap-icon://error");

statusIndicatorText.setState(sap.ui.core.ValueState.Warning);

documentScanState = sap.ui.core.ValueState.Success;

inProcessState = sap.ui.core.ValueState.Success;

exceptionState = sap.ui.core.ValueState.Warning;

if(this.getCurrentUserRole()!==''){

this.byId("btnSubmit").setVisible(false);

}else{

this.byId("btnSubmit").setEnabled(false);

}

this.byId("btnSave").setEnabled(true);

break;

case "Exception\_Complete":

statusIndicatorText.setText("Exception Complete");

statusIndicatorText.setIcon("sap-icon://accept");

statusIndicatorText.setState(sap.ui.core.ValueState.Warning);

documentScanState = sap.ui.core.ValueState.Success;

inProcessState = sap.ui.core.ValueState.Success;

exceptionState = sap.ui.core.ValueState.Success;

exceptionCompletedState = sap.ui.core.ValueState.Warning;

this.byId("btnSave").setEnabled(true);

if(this.getCurrentUserRole()===''){

this.byId("btnSubmit").setEnabled(true);

}

break;

case "Pending\_Approval":

statusIndicatorText.setText("Pending Approval");

statusIndicatorText.setIcon("sap-icon://accept");

statusIndicatorText.setState(sap.ui.core.ValueState.Warning);

documentScanState = sap.ui.core.ValueState.Success;

inProcessState = sap.ui.core.ValueState.Success;

exceptionState = sap.ui.core.ValueState.Success;

exceptionCompletedState = sap.ui.core.ValueState.Success;

this.byId("btnSave").setEnabled(false);

this.byId("btnEdit").setEnabled(false);

this.byId("btnSubmit").setEnabled(false);

break;

case "Posted":

statusIndicatorText.setText("Posted");

statusIndicatorText.setIcon("sap-icon://accounting-document-verification");

statusIndicatorText.setState(sap.ui.core.ValueState.Warning);

documentScanState = sap.ui.core.ValueState.Success;

inProcessState = sap.ui.core.ValueState.Success;

invoicePostedStatus = sap.ui.core.ValueState.Success;

exceptionState=sap.ui.core.ValueState.Success;

exceptionCompletedState = sap.ui.core.ValueState.Success;

this.byId("btnSubmit").setEnabled(false);

break;

default:

statusIndicatorText.setText("");

statusIndicatorText.setIcon("sap-icon://document-text");

statusIndicatorText.setState(sap.ui.core.ValueState.None);

break;

}

var statusIndContent = statusIndicatorPF.getContent();

statusIndContent[0].setState(documentScanState);

statusIndContent[1].setState(inProcessState);

statusIndContent[2].setState(exceptionState);

statusIndContent[3].setState(exceptionCompletedState);

statusIndContent[4].setState(invoicePostedStatus);

},

\_allowInvoiceModification: function (isModificationAllowed) {

if (isModificationAllowed) {

// Enable the buttons to save, edit, submit the invoice

this.byId("btnSave").setEnabled(true);

this.byId("btnEdit").setEnabled(true);

// Also, enable the button, displaying the exceptions message pop over

this.byId("btnExceptions").setEnabled(true);

} else {

// Disable the buttons to save, edit, submit the invoice

this.byId("btnSave").setEnabled(false);

this.byId("btnEdit").setEnabled(false);

// Also, disable the button, displaying the exceptions message pop over

this.byId("btnExceptions").setEnabled(false);

}

},

// onBackButtonPress: function () {

// this.onHandleCancelPress();

// var oRouter = sap.ui.core.UIComponent.getRouterFor(this);

// oRouter.navTo("toInvoiceList");

// },

// fnImageFmtr: function (value) {

// var imgSrc = value;

// return imgSrc;

// },

// onExit: function () {

// this.onHandleCancelPress();

// for (var sPropertyName in this.\_formFragments) {

// if (!this.\_formFragments.hasOwnProperty(sPropertyName) || this.\_formFragments[sPropertyName] == null) {

// return;

// //if the invoice is not even saved once, disable submit button

// if (invoiceDetailsJSON.invoiceStatus === "Scanned") {

// currentViewController.byId("btnSubmit").setEnabled(false);

// }

// //Send data to WF Obj

// var contextData = {};

// contextData = invoiceDetailsJSON;

// sap.ui.getCore().invoiceDetailsWF = contextData; /\*invoiceDetailsWF - To be used in WF call\*/

// // hide the busy indicator, once the view has been updated completely

// BusyIndicator.hide();

// }

// }

// },

error: function (error) {

// Hide the busy indicator and display the error returned from the service

BusyIndicator.hide();

MessageBox.error("Error: " + error.toString());

},

// });

// },

// Get Current user Role

getCurrentUserRole: function () {

var roleParameterValueTemp = location.href.split("&role=");

var sTaskTemp = roleParameterValueTemp[0].split("?TaskID=");

var iGUIDTemp = sTaskTemp[0].split("Invoice/");

// sap.ui.getCore.sTask = sTaskTemp[1];

if (roleParameterValueTemp[1] && roleParameterValueTemp[1].toLowerCase() === APPROVER\_ROLE) {

// this.getCurrentWFTask();

sap.ui.getCore.sTask = sTaskTemp[1];// Capture Task ID from WF Invoice Link

sap.ui.getCore.iGUIDFromWFTask = iGUIDTemp[1];// Capture invoiceGUID from WF Invoice Link

return APPROVER\_ROLE;

}

if(roleParameterValueTemp[1] && roleParameterValueTemp[1].toLowerCase() !== APPROVER\_ROLE){

sap.ui.getCore.sTask = sTaskTemp[1];// Capture Task ID from WF Invoice Link

sap.ui.getCore.iGUIDFromWFTask = iGUIDTemp[1];// Capture invoiceGUID from WF Invoice Link

return roleParameterValueTemp[1];

}

return "";

},

// Get Current WF Task ID

getCurrentWFTask: function () {

// var roleParameterValue = location.href.split("?role=");

var sTask = location.href.split("&TaskID=");

if (sTask[1]) {

return sTask[1];

}

return "";

},

onBackButtonPress: function () {

this.onHandleCancelPress();

var oRouter = sap.ui.core.UIComponent.getRouterFor(this);

oRouter.navTo("toInvoiceList");

},

fnImageFmtr: function (value) {

var imgSrc = value;

return imgSrc;

},

onExit: function () {

this.onHandleCancelPress();

for (var sPropertyName in this.\_formFragments) {

if (!this.\_formFragments.hasOwnProperty(sPropertyName) || this.\_formFragments[sPropertyName] == null) {

return;

}

this.\_formFragments[sPropertyName].destroy();

this.\_formFragments[sPropertyName] = null;

}

},

onHandleEditPress: function () {

this.\_toggleButtonsAndView(true);

//window.\_control=this.byId("productInput");

this.byId("btnSave").setVisible(true);

this.byId("btnSave").setEnabled(true);

this.byId("btnCancel").setVisible(true);

this.byId("btnDelete").setEnabled(false);

this.byId("btnCopy").setEnabled(false);

this.byId("btnEdit").setEnabled(false);

this.byId("btnSubmit").setEnabled(false);

this.byId("btnApprove").setEnabled(false);

this.byId("btnReject").setEnabled(false);

// Edit Fragment - Get the PONumber and make poItemTab or nonpoItemtab visible

// var poNumber = sap.ui.getCore().purchaseOrder;

this.\_setPoNonPoTabVisibility(true);

},

onHandleCancelPress: function () {

this.\_toggleButtonsAndView(false);

// this.byId("btnSave").setVisible(false);

// this.byId("btnSave").setEnabled(true);

this.byId("btnCancel").setVisible(false);

this.byId("btnDelete").setVisible(true);

this.byId("btnCopy").setVisible(true);

this.byId("btnEdit").setEnabled(true);

this.\_setPoNonPoTabVisibility(false);

},

onHandleSavePress: function () {

this.\_toggleButtonsAndView(false);

this.\_saveInvoice();

// window.location.reload();

var btnException = this.byId("btnExceptions");

if (btnException.getText() === "0") {

this.byId("btnApprove").setEnabled(true);

this.byId("btnReject").setEnabled(true);

}

var statusIndicator = this.getView().byId("invoiceStatusIndicatorBox");

var statusIndicatorText = statusIndicator.getItems()[0];

var statusIndicatorPF = statusIndicator.getItems()[1];

if(sap.ui.getCore().invoiceStatus==="Pending\_Approval" || this.getCurrentUserRole() === APPROVER\_ROLE)

{

this.byId("btnSave").setEnabled(false);

this.byId("btnEdit").setEnabled(false);

}

else{

this.\_setStatusIndicator(sap.ui.getCore().invoiceStatus);

}

this.\_setPoNonPoTabVisibility(false);

},

//function to remove null values

filterNull: function (obj) {

for (var key in obj) {

if (obj[key] === null) {

obj[key] = "";

}

// else if (Object.prototype.toString.call(obj[key]) === '[object Object]') {

// filter(obj[key]);

// }

}

},

//save

\_saveInvoice: function () {

if(sap.ui.getCore().invoiceStatus==="Scanned"){

sap.ui.getCore().invoiceStatus="In\_Progress";

}

var currentViewController = this;

var invoiceDetails = this.getView().getModel("invoiceDetails");

var invoiceDetailsJSON = invoiceDetails.getData();

//If the data has not changed after loading, do not call the service

if (this.getCurrentUserRole() === APPROVER\_ROLE && JSON.stringify(invoiceDetailsJSON) === \_formData) {

return false;

}

var invoiceItems = [];

var counter = 1;

invoiceDetails.getProperty("/items/results").forEach(function (invoiceDetailsItem) {

var invoiceItem = {};

invoiceItem.invoiceID = invoiceDetails.getProperty("/invoiceID");

invoiceItem.invoiceItemIndex = counter++;

invoiceItem.invoiceStatus = sap.ui.getCore().invoiceStatus;

invoiceItem.documentType = invoiceDetails.getProperty("/documentType") || "";

invoiceItem.documentDate = invoiceDetails.getProperty("/documentDate") || "";

invoiceItem.invoiceDate = invoiceDetails.getProperty("/invoiceDate") || "";

invoiceItem.postingDate = invoiceDetails.getProperty("/postingDate") || "";

invoiceItem.invoicingParty = invoiceDetails.getProperty("/invoicingParty") || "";

invoiceItem.companyCode = invoiceDetails.getProperty("/companyCode") || "";

invoiceItem.paymentCurrency = invoiceDetails.getProperty("/paymentCurrency") || "";

invoiceItem.invoiceType = invoiceDetails.getProperty("/invoiceType") || "";

invoiceItem.headerText = invoiceDetails.getProperty("/headerText") || "";

invoiceItem.supplierNumber = invoiceDetails.getProperty("/supplierNumber") || "";

invoiceItem.paymentMethod = invoiceDetails.getProperty("/paymentMethod") || "";

invoiceItem.paymentBlock = invoiceDetails.getProperty("/paymentBlock") || "";

invoiceItem.partnerBank = invoiceDetails.getProperty("/partnerBank") || "";

invoiceItem.houseBank = invoiceDetails.getProperty("/houseBank") || "";

invoiceItem.houseBankAccountID = invoiceDetails.getProperty("/houseBankAccountID") || "";

invoiceItem.paymentMethodSupp = invoiceDetails.getProperty("/paymentMethodSupp") || "";

invoiceItem.autoCalculateTaxIndicator = invoiceDetails.getProperty("/autoCalculateTaxIndicator").toString() || "";

invoiceItem.baselineDate = invoiceDetails.getProperty("/baselineDate") || "";

invoiceItem.dueDate = invoiceDetails.getProperty("/dueDate") || "";

invoiceItem.vatAmount = invoiceDetails.getProperty("/vatAmount") || "";

invoiceItem.vatAmount1 = invoiceDetails.getProperty("/vatAmount1") || "";

invoiceItem.vatAmount2 = invoiceDetails.getProperty("/vatAmount2") || "";

invoiceItem.vatAmount3 = invoiceDetails.getProperty("/vatAmount3") || "";

invoiceItem.vatAmount4 = invoiceDetails.getProperty("/vatAmount4") || "";

invoiceItem.vatRate = invoiceDetails.getProperty("/vatRate") || "";

invoiceItem.vatRate1 = invoiceDetails.getProperty("/vatRate1") || "";

invoiceItem.vatRate2 = invoiceDetails.getProperty("/vatRate2") || "";

invoiceItem.vatRate3 = invoiceDetails.getProperty("/vatRate3") || "";

invoiceItem.vatRate4 = invoiceDetails.getProperty("/vatRate4") || "";

invoiceItem.paymentTerms = invoiceDetails.getProperty("/paymentTerms") || "";

invoiceItem.day1 = invoiceDetails.getProperty("/day1") || "";

invoiceItem.day2 = invoiceDetails.getProperty("/day2") || "";

invoiceItem.day3 = invoiceDetails.getProperty("/day3") || "";

invoiceItem.discount1 = invoiceDetails.getProperty("/discount1") || "";

invoiceItem.discount2 = invoiceDetails.getProperty("/discount2") || "";

invoiceItem.exchangeRate = invoiceDetails.getProperty("/exchangeRate") || "";

invoiceItem.withholdingTax = invoiceDetails.getProperty("/withholdingTax") || "";

invoiceItem.unplannedDeliveryCost = invoiceDetails.getProperty("/unplannedDeliveryCost") || "";

invoiceItem.instanceKey = invoiceDetails.getProperty("/instanceKey") || "";

invoiceItem.bankAccount = invoiceDetails.getProperty("/bankAccount") || "";

invoiceItem.bankName = invoiceDetails.getProperty("/bankName") || "";

invoiceItem.bankNumber = invoiceDetails.getProperty("/bankNumber") || "";

invoiceItem.bankPartnerType = invoiceDetails.getProperty("/bankPartnerType") || "";

invoiceItem.vatID = invoiceDetails.getProperty("/vatID") || "";

invoiceItem.vatID1 = invoiceDetails.getProperty("/vatID1") || "";

invoiceItem.vatID2 = invoiceDetails.getProperty("/vatID2") || "";

invoiceItem.vatID3 = invoiceDetails.getProperty("/vatID3") || "";

invoiceItem.vatID4 = invoiceDetails.getProperty("/vatID4") || "";

invoiceItem.requestorEmail = invoiceDetails.getProperty("/requestorEmail") || "";

invoiceItem.invoiceFreightAmount = invoiceDetails.getProperty("/invoiceFreightAmount") || "";

invoiceItem.invoiceHandlingCharges = invoiceDetails.getProperty("/invoiceHandlingCharges") || "";

invoiceItem.alternatePayee = invoiceDetails.getProperty("/alternatePayee") || "";

invoiceItem.documentItemInInvoice = invoiceDetails.getProperty("/documentItemInInvoice") || "";

invoiceItem.purchaseOrder = invoiceDetails.getProperty("/purchaseOrder") || "";

invoiceItem.itemPurchaseOrder = invoiceDetails.getProperty("/itemPurchaseOrder") || "";

invoiceItem.uomPurchaseOrder = invoiceDetails.getProperty("/uomPurchaseOrder") || "";

invoiceItem.amountDocumentCurrency = invoiceDetails.getProperty("/amountDocumentCurrency") || "";

invoiceItem.deliveryNoteNumber = invoiceDetails.getProperty("/deliveryNoteNumber") || "";

invoiceItem.materialNumber = invoiceDetails.getProperty("/materialNumber") || "";

invoiceItem.taxCode = invoiceDetails.getProperty("/taxCode") || "";

invoiceItem.accountAssigned = invoiceDetails.getProperty("/accountAssigned") || "";

invoiceItem.directPostingGL = invoiceDetails.getProperty("/directPostingGL") || "";

invoiceItem.directPostingMaterial = invoiceDetails.getProperty("/directPostingMaterial") || "";

invoiceItem.lastUser = "";

invoiceItem.currentUser = "";

invoiceItem.lastUpdatedTS = invoiceDetails.getProperty("/lastUpdatedTS") || "";

invoiceItem.lastRole = invoiceDetails.getProperty("/lastRole") || "";

invoiceItem.invoiceCategory = invoiceDetails.getProperty("/invoiceCategory") || "";

invoiceItem.shipToAddress1 = invoiceDetails.getProperty("/shipToAddress1") || "";

invoiceItem.shipToAddress2 = invoiceDetails.getProperty("/shipToAddress2") || "";

invoiceItem.billToCompany = invoiceDetails.getProperty("/billToCompany") || "";

invoiceItem.billToName = invoiceDetails.getProperty("/billToName") || "";

invoiceItem.billToPhone = invoiceDetails.getProperty("/billToPhone") || "";

invoiceItem.billToAddress1 = invoiceDetails.getProperty("/billToAddress1") || "";

invoiceItem.billToAddress2 = invoiceDetails.getProperty("/billToAddress2") || "";

invoiceItem.billToCountry = invoiceDetails.getProperty("/billToCountry") || "";

invoiceItem.billToVAT = invoiceDetails.getProperty("/billToVAT") || "";

invoiceItem.billToEmail = invoiceDetails.getProperty("/billToEmail") || "";

invoiceItem.remitName = invoiceDetails.getProperty("/remitName") || "";

invoiceItem.remitToAddress1 = invoiceDetails.getProperty("/remitToAddress1") || "";

invoiceItem.remitToAddress2 = invoiceDetails.getProperty("/remitToAddress2") || "";

invoiceItem.remitToPin = invoiceDetails.getProperty("/remitToPin") || "";

invoiceItem.remitToStreet = invoiceDetails.getProperty("/remitToStreet") || "";

invoiceItem.remitToVAT = invoiceDetails.getProperty("/remitToVAT") || "";

invoiceItem.remitToEmail = invoiceDetails.getProperty("/remitToEmail") || "";

invoiceItem.remitToPhone = invoiceDetails.getProperty("/remitToPhone") || "";

invoiceItem.supplierName = invoiceDetails.getProperty("/supplierName") || "";

invoiceItem.supplierName2 = invoiceDetails.getProperty("/supplierName2") || "";

invoiceItem.supplierPOBox = invoiceDetails.getProperty("/supplierPOBox") || "";

invoiceItem.supplierPOBoxZip = invoiceDetails.getProperty("/supplierPOBoxZip") || "";

invoiceItem.supplierStreet = invoiceDetails.getProperty("/supplierStreet") || "";

invoiceItem.supplierState = invoiceDetails.getProperty("/supplierState") || "";

invoiceItem.supplierZip = invoiceDetails.getProperty("/supplierZip") || "";

invoiceItem.netAmount = invoiceDetails.getProperty("/netAmount") || "";

invoiceItem.ibanNumber = invoiceDetails.getProperty("/ibanNumber") || "";

invoiceItem.swiftCode = invoiceDetails.getProperty("/swiftCode") || "";

invoiceItem.grossAmount = invoiceDetails.getProperty("/grossAmount") || "";

invoiceItem.glAccount = invoiceDetails.getProperty("/glAccount") || "";

invoiceItem.glDescription = invoiceDetails.getProperty("/glDescription") || "";

invoiceItem.costCenter = invoiceDetails.getProperty("/costCenter") || "";

invoiceItem.costCenterDescription = invoiceDetails.getProperty("/costCenterDescription") || "";

invoiceItem.wbsElement = "";

invoiceItem.wbsElementDescription = "";

invoiceItem.internalOrder = invoiceDetails.getProperty("/internalOrder") || "";

invoiceItem.internalOrderDescription = "";

invoiceItem.assetID = invoiceDetails.getProperty("/assetID") || "";

invoiceItem.assetDescription = invoiceDetails.getProperty("/assetDescription") || "";

invoiceItem.invoiceFileName = invoiceDetails.getProperty("/invoiceFileName") || "";

invoiceItem.postedInvoice = "";

invoiceItem.invoiceGUID = invoiceDetails.getProperty("/invoiceGUID");

if (invoiceItem.autoCalculateTaxIndicator.toString() === "true"){

invoiceItem.autoCalculateTaxIndicator = "true";

invoiceItem.totalTaxAmount = "";

}

else{

invoiceItem.autoCalculateTaxIndicator = "false";

invoiceItem.totalTaxAmount = invoiceDetails.getProperty("/totalTaxAmount") || "";

}

if (!(invoiceItem.purchaseOrder === "")) {

// if (!(invoiceItem.invoiceID === "")) {

// Items

invoiceItem.itemNumber = invoiceDetailsItem.itemNumber || "";

invoiceItem.itemCode = invoiceDetailsItem.itemCode || "";

// invoiceItem.itemCompanyCode = invoiceDetailsItem.itemCompanyCode;

invoiceItem.itemDescription = invoiceDetailsItem.itemDescription || "";

invoiceItem.itemQuantity = invoiceDetailsItem.itemQuantity || "";

invoiceItem.unitPrice = invoiceDetailsItem.unitPrice || "";

invoiceItem.itemUnit = invoiceDetailsItem.itemUnit || "";

invoiceItem.itemQuantityShipped = invoiceDetailsItem.itemQuantityShipped || "";

invoiceItem.itemTax = invoiceDetailsItem.itemTax || "";

invoiceItem.taxJurisdiction = invoiceDetailsItem.taxJurisdiction || "";

invoiceItem.itemTaxRate = invoiceDetailsItem.itemTaxRate || "";

invoiceItem.totalAmount = invoiceDetailsItem.totalAmount || "";

invoiceItem.pgiDate = invoiceDetailsItem.pgiDate || "";

}

else {

invoiceDetails.getProperty("/nonPO").forEach(function (invoiceDetailsnonPOItem) {

invoiceItem.invoiceItemIndex = invoiceDetailsnonPOItem.invoiceItemIndex || "";

invoiceItem.glAccount = invoiceDetailsnonPOItem.glAccount || "";

invoiceItem.glDescription = invoiceDetailsnonPOItem.glDescription || "";

invoiceItem.internalOrder = invoiceDetailsnonPOItem.internalOrder || "";

invoiceItem.costCenter = invoiceDetailsnonPOItem.costCenter || "";

invoiceItem.itemPrice = invoiceDetailsnonPOItem.itemPrice || "";

invoiceItem.wbsElement = invoiceDetailsnonPOItem.wbsElement || "";

invoiceItem.itemTax = invoiceDetailsnonPOItem.itemTax || "";

invoiceItem.taxJurisdiction = invoiceDetailsnonPOItem.taxJurisdiction || "";

});

}

invoiceItems.push(invoiceItem);

});

//function to remove null values

this.filterNull(invoiceItems);

var displayExceptionItems=[];

var modelDisplayExceptions=invoiceDetails.getProperty("/displayExceptions/results");

if(modelDisplayExceptions && modelDisplayExceptions.length){

modelDisplayExceptions.forEach(function (exceptionItem){

var displayExceptionItem={

invoiceGUID:exceptionItem.invoiceGUID,

invoiceWorkflowExceptionId:exceptionItem.invoiceWorkflowExceptionId,

messageText:exceptionItem.messageText,

userbypassStatus:exceptionItem.userbypassStatus.toString()

};

displayExceptionItems.push(displayExceptionItem);

});

}

var servicePayloadJSON = JSON.stringify({

invoiceItems: invoiceItems,

displayExceptions:displayExceptionItems

});

$.post({

url: SAVE\_INVOICE\_SERVICE,

contentType: 'application/json',

async: false,

data: servicePayloadJSON

}).done(function (responseData, textStatus, xhr) {

// Update the successfully derived fields in the UI and make them available on the details page

\_formData = JSON.stringify(invoiceDetailsJSON);

currentViewController.\_updateModelAfterInvoiceSave(responseData.updatedInvoiceItems, invoiceDetails);

currentViewController.\_updateDisplayExceptionModelAfterInvoiceSave(responseData.displayExceptions,invoiceDetails);

// Load the exception messages in the error message pop over

if (responseData.exceptions) {

currentViewController.\_displayExceptions(responseData.exceptions,

(responseData.exceptions && (responseData.exceptions.length > 0)));

currentViewController.byId("btnSubmit").setEnabled(false);

// sap.ui.getCore().lengthExcep = responseData.exceptions.length;

} else {

oMessagePopover.setModel(new JSONModel(""));

var btnException = currentViewController.byId("btnExceptions");

btnException.setText(0);

// currentViewController.\_displayExceptions(((!responseData.exceptions) ),false);

currentViewController.byId("btnSubmit").setEnabled(true);

}

//check if exceptions are cleared and complete the workflow

var role=currentViewController.getCurrentUserRole();

sap.ui.getCore().invoiceStatus=responseData.updatedInvoiceItems[0].invoiceStatus;

if(role && sap.ui.getCore.sTask){

var exceptionsCleared=true;

invoiceDetails.getData().displayExceptions.results.forEach(function(ex){

if(ex.assignTo===role && ex.exceptionStatus!=="SUCCESS"){

exceptionsCleared=false;

return false;

}

return true;

});

if(exceptionsCleared){

currentViewController.\_completeUserTask(sap.ui.getCore.sTask,{status:"COMPLETED",context:{exceptionCleared:true}});

}

}

//Send data to WF Obj

var contextData = {};

contextData = invoiceItems[0];

sap.ui.getCore().invoiceDetailsWF = contextData; /\*invoiceDetailsWF - To be used in WF call\*/

sap.ui.getCore().invoiceDetailsWF.displayExceptions={results:responseData.displayExceptions};

\_hasNonDisplayExceptions=false;

responseData.exceptions.forEach(function(ex){

if(ex.exceptionMessage.indexOf("E")!==0){

\_hasNonDisplayExceptions=true;

return false;

}else{

return true;

}

});

MessageToast.show("Invoice saved");

// currentViewController.byId("btnSubmit").setEnabled(true);

}).fail(function (xhr, textStatus, error) {

// Display an error message box with the error message

MessageBox.error("Invoice Save Failed: " + xhr.responseText);

});

},

\_formFragments: {},

handleValueHelp : function (oEvent) {

var paths=[{fragment:"paymenttermDialog",fields:["PmntTerm","Text"],inputid:"\_\_xmlview1--payment"},{fragment:"companycodeDialog",fields:["CompCode","Text"],inputid:"\_\_xmlview1--company"},{fragment:"remitsupplierDialog",fields:["RemitToSupID","Name","CompCode"],inputid:"\_\_xmlview1--remit"},{fragment:"itemcodeDialog",fields:["matnr","maktx","bukrs"],inputid:"inpitemCode"},{fragment:"unitofmeasureDialog",fields:["MSEH3","MSEHL","ISOCODE"],inputid:"unitofmeasure"}];

var sInputValue = oEvent.getSource().getValue();

this.inputId = oEvent.getSource().getId();

var y = this.inputId ;

var obj = paths.find(function(x) {

if(y.indexOf(x.inputid)>=0) {

return x;

}});

if(obj.inputid=== "inpitemCode"){

this.\_currentInvoiceItemSelected=oEvent.getSource().getBindingContext("invoiceDetails").getPath();

}

if(obj.inputid=== "unitofmeasure" ){

this.\_currentInvoiceItemSelected=oEvent.getSource().getBindingContext("invoiceDetails").getPath();

}

var fragmentId = "com.deloitte.scp.app.dips.view." + obj.fragment;

if (!this.\_valueHelpDialogs[fragmentId]) { //creating an attribute called fragmentId in the empty object \_valueHelpDialogs ,which is globally declared above

// this.\_valueHelpDialog = sap.ui.xmlfragment(this.getView().getId(), "com.deloitte.scp.app.dips.view.paymenttermDialog",this);

this.\_valueHelpDialogs[fragmentId] = sap.ui.xmlfragment(this.getView().getId(),fragmentId,this);

this.getView().addDependent(this.\_valueHelpDialogs[fragmentId]);

}

this.\_valueHelpDialogs[fragmentId].open(sInputValue);

var filter=new Filter({

filters:

obj.fields.map(function(field){

return new Filter({

path:field,

operator: sap.ui.model.FilterOperator.Contains,

value1: sInputValue

});

})

});

this.\_valueHelpDialogs[fragmentId].getBinding("items").filter([filter]);

},

\_handleValueHelpSearch : function (evt) {

var paths=[{fragmentid:"\_\_xmlview1--fragment\_payment",fields:["PmntTerm","Text"]},{fragmentid:"\_\_xmlview1--fragment\_company",fields:["CompCode","Text"]},{fragmentid:"\_\_xmlview1--fragment\_remit",fields:["RemitToSupID","Name","CompCode"]},{fragmentid:"\_\_xmlview1--fragment\_item",fields:["matnr","maktx","bukrs"]},{fragmentid:"\_\_xmlview1--fragment\_uom",fields:["MSEH3","MSEHL","ISOCODE"]}];

var sValue = evt.getParameter("value");

this.inputId = evt.getSource().getId();

var y = this.inputId ;

var obj = paths.find(function(x) {

if(x.fragmentid === y) {

return x;

}});

var filters=new Filter({

filters: obj.fields.map(function(field){

return new Filter({

path:field,

operator: sap.ui.model.FilterOperator.Contains,

value1: sValue

});

})

});

//var oFilter = new Filter("PmntTerm",sap.ui.model.FilterOperator.Contains, sValue);

evt.getSource().getBinding("items").filter([filters]);

},

\_handleValueHelpClose : function (evt) {

var paths=[{fragmentid:"\_\_xmlview1--fragment\_payment",fields:["PmntTerm","Text"],assignto:"/billToEmail"},{fragmentid:"\_\_xmlview1--fragment\_company",fields:["CompCode","Text"],assignto:"/companyCode"},{fragmentid:"\_\_xmlview1--fragment\_remit",fields:["RemitToSupID","Name","CompCode"],assignto:"/supplierName2"},{fragmentid:"\_\_xmlview1--fragment\_item",fields:["matnr","maktx","bukrs"],assignto:"/itemCode"},{fragmentid:"\_\_xmlview1--fragment\_uom",fields:["MSEH3","MSEHL","ISOCODE"],assignto:"/itemUnit"}];

var oSelectedItem = evt.getParameter("selectedItem");

this.inputId = evt.getSource().getId();

var y = this.inputId ;

var obj = paths.find(function(x) {

if(x.fragmentid === y) {

return x;

}});

if (oSelectedItem) {

var oModel=this.getView().getModel("invoiceDetails");

// oModel.setProperty("/paymentTerms",oSelectedItem.getDescription());

if(obj.fragmentid === "\_\_xmlview1--fragment\_payment"){

oModel.setProperty(obj.assignto,oSelectedItem.getTitle());

oModel.setProperty("/paymentTerms",oSelectedItem.getDescription());

}

else if(obj.fragmentid==="\_\_xmlview1--fragment\_item"){

oModel.setProperty(this.\_currentInvoiceItemSelected +"/itemCode",oSelectedItem.getTitle());

oModel.setProperty(this.\_currentInvoiceItemSelected +"/itemDescription",oSelectedItem.getDescription());

}

else if(obj.fragmentid==="\_\_xmlview1--fragment\_uom"){

oModel.setProperty(this.\_currentInvoiceItemSelected +"/itemUnit",oSelectedItem.getTitle());

}

else{

oModel.setProperty(obj.assignto,oSelectedItem.getTitle());

}

}

// evt.getSource().getBinding("items").filter([]);

},

suggestionItemSelected: function (evt) {

var oItem = evt.getParameter("selectedItem");

var oModel=this.getView().getModel("invoiceDetails");

if(oItem){

oModel.setProperty("/paymentTerms",oItem.getAdditionalText());

}

},

suggestionItemSelected1: function (evt) {

var oItem = evt.getParameter("selectedItem");

var oModel=this.getView().getModel("invoiceDetails");

if(oItem){

oModel.setProperty(evt.getSource().getBindingContext("invoiceDetails").getPath()+"/itemDescription",oItem.getAdditionalText());

}

},

// for company code

handleValueHelp1 : function (oEvent) {

//var paths=[{fragment:"paymenttermDialog",fields:["PmntTerm","Text"],inputid:"\_\_input34"},{fragment:"companycodeDialog",fields:["CompCode","Text"],inputid:"\_\_input10"}];

var sInputValue = oEvent.getSource().getValue();

this.inputId = oEvent.getSource().getId();

//var obj = paths.find(o => o.inputid = this.inputId);

// var fragmentid = "com.deloitte.scp.app.dips.view." + obj.

//if(this.inputId = paths.inputid){

if (!this.\_valueHelpDialog1) {

this.\_valueHelpDialog1 = sap.ui.xmlfragment(this.getView().getId(), "com.deloitte.scp.app.dips.view.itemcodeDialog",this);

this.getView().addDependent(this.\_valueHelpDialog1);

}

this.\_valueHelpDialog1.open(sInputValue);

var filters=new Filter({

filters: [

new Filter({

path: 'RemitToSupID',

operator: sap.ui.model.FilterOperator.Contains,

value1: sInputValue

}),

new Filter({

path: 'Name',

operator: sap.ui.model.FilterOperator.Contains,

value1: sInputValue

})

]

});

this.\_valueHelpDialog1.getBinding("items").filter([filters]);

},

// \_handleValueHelpSearch1 : function (evt) {

// var sValue = evt.getParameter("value");

// var filters=new Filter({

// filters: [

// new Filter({

// path: 'CompCode',

// operator: sap.ui.model.FilterOperator.Contains,

// value1: sValue

// }),

// new Filter({

// path: 'Text',

// operator: sap.ui.model.FilterOperator.Contains,

// value1: sValue

// })

// ]

// });

// //var oFilter = new Filter("PmntTerm",sap.ui.model.FilterOperator.Contains, sValue);

// evt.getSource().getBinding("items").filter([filters]);

// },

// \_handleValueHelpClose1 : function (evt) {

// var oSelectedItem = evt.getParameter("selectedItem");

// if (oSelectedItem) {

// var oModel=this.getView().getModel("invoiceDetails");

// // oModel.setProperty("/paymentTerms",oSelectedItem.getDescription());

// oModel.setProperty("/companyCode",oSelectedItem.getTitle());

// }

// evt.getSource().getBinding("items").filter([]);

// },

// suggestionItemSelected1 : function (evt) {

// var oItem = evt.getParameter("selectedItem");

// var oModel=this.getView().getModel("invoiceDetails");

// if(oItem){

// oModel.setProperty("/companyCode",oItem.getAdditionalText());

// //oModel.setProperty("/companyCode",oItem.getTitle());

// }

// },

getSearchHelpValues : function(){

var searchHelpValuesQuery = new $.Deferred();

$.get({

url: SEARCH\_HELP\_SERVICE,

contentType: 'application/json',

async: false

}).done(function (responseData, textStatus, xhr){

var searchHelpValuesJSON = JSON.parse(responseData);

searchHelpValuesQuery.resolve(searchHelpValuesJSON);

});

return searchHelpValuesQuery.promise();

},

\_toggleButtonsAndView: function (bEdit) {

// Set the right form type

this.\_showFormFragment(bEdit ? "InvoiceDetailChange" : "InvoiceDetailDisplay");

},

\_getFormFragment: function (sFragmentName) {

// sap.ui.getCore().sFragmentName = sFragmentName;

var oFormFragment = this.\_formFragments[sFragmentName];

if (oFormFragment) {

return oFormFragment;

}

oFormFragment = sap.ui.xmlfragment(this.getView().getId(), "com.deloitte.scp.app.dips.view." + sFragmentName,this);

this.\_formFragments[sFragmentName] = oFormFragment;

return this.\_formFragments[sFragmentName];

},

\_showFormFragment: function (sFragmentName) {

//setting the id of the Dynamic Page created in the view//

var oPage = this.byId("dpInvoicedetailId");

oPage.setContent(this.\_getFormFragment(sFragmentName));

},

\_loadExceptions: function (oInvoiceModel, invoiceGUID) {

var currentViewController = this;

var invoiceServiceURL = oInvoiceModel.sServiceUrl;

var queryStatus = new $.Deferred();

$.get(invoiceServiceURL + "/InvoiceExceptions?$filter=invoiceGUID eq '" + invoiceGUID + "'&$format=json")

.done(function (

responseData) {

var exceptionsJSON = JSON.parse(JSON.stringify(responseData));

if (exceptionsJSON.d.results && exceptionsJSON.d.results.length) {

\_hasExceptions = true;

} else {

\_hasExceptions = false;

}

if(\_hasExceptions){

\_hasNonDisplayExceptions = false;

exceptionsJSON.d.results.forEach(function(ex){

if(ex.exceptionMessage.indexOf("E") !== 0){

\_hasNonDisplayExceptions = true;

return false;

}

return true;

});

}

currentViewController.\_displayExceptions(exceptionsJSON.d.results);

queryStatus.resolve();

})

.fail(function (jqXHR, textStatus, error) {

MessageBox.error("Error" + error.toString());

queryStatus.reject();

});

return queryStatus.promise();

},

\_displayExceptions: function (exceptionsJSON, showExceptions) {

var exceptionData = exceptionsJSON.map(function (e) {

return {

type: "Error",

title: "Exception",

description: e.exceptionMessage

};

});

oMessagePopover.setModel(new JSONModel(exceptionData));

//Disable submit button if there are exceptions

if (exceptionData.length !== 0) {

this.byId("btnSubmit").setEnabled(!exceptionData.length);

}

var btnException = this.byId("btnExceptions");

btnException.setText(exceptionData.length || 0);

if (showExceptions) {

btnException.firePress();

}

},

\_updateModelAfterInvoiceSave: function (updatedItems, oInvoiceModel) {

if (updatedItems && updatedItems.length) {

updatedItems.forEach(function (item,ind) {

var index = item.invoiceItemIndex;

if (!index) {

return;

}

var invoiceItem = oInvoiceModel.getProperty("/items/results")[index - 1];

Object.keys(item).forEach(function (key) {

if(key==="autoCalculateTaxIndicator"){

var autoCalcVal=oInvoiceModel.getProperty("/autoCalculateTaxIndicator");

if(autoCalcVal && autoCalcVal.toString()==="true"){

autoCalcVal=true;

}else{

autoCalcVal=false;

}

oInvoiceModel.setProperty("/autoCalculateTaxIndicator",autoCalcVal);

}else{

if (typeof oInvoiceModel.getProperty("/" + key) !== "undefined" && oInvoiceModel.getProperty("/" + key) !== item[key]) {

oInvoiceModel.setProperty("/" + key, item[key]);

}

if (typeof invoiceItem[key] !== "undefined" && invoiceItem[key] !== item[key]) {

oInvoiceModel.setProperty("/items/results/" + (index - 1) + "/" + key, item[key]);

}

}

});

});

}

},

\_updateDisplayExceptionModelAfterInvoiceSave: function (displayExceptions, oInvoiceModel) {

if (displayExceptions && displayExceptions.length) {

if (!oInvoiceModel.getProperty("/displayExceptions/results").length) {

var lstExceptions = [];

displayExceptions.forEach(function (item) {

item.userbypassStatus=item.userbypassStatus==="true" ? true:false;

lstExceptions.push(item);

});

oInvoiceModel.setProperty("/displayExceptions/results", lstExceptions);

}else{

displayExceptions.forEach(function (item,ind) {

var exceptionItem = oInvoiceModel.getProperty("/displayExceptions/results")[ind];

Object.keys(item).forEach(function (key) {

if (typeof exceptionItem[key] !== "undefined" && exceptionItem[key] !== item[key]) {

if(key!=="userbypassStatus"){

oInvoiceModel.setProperty("/displayExceptions/results/" + ind + "/" + key, item[key]);

}

}

});

});

}

}

},

onMessagePopoverPress: function (oEvent) {

oMessagePopover.toggle(oEvent.getSource());

},

// WorkFlow Ftech Token and call instance methods

onSubmitPress: function () {

var token = this.\_fetchToken(); //Fetch Token for WF from bpmruntime

if(token){

this.\_startInstance(token); //Start WF Instance - My Inbox

this.\_setPendingApprvalStatusOnSubmit(); //Post the invoice status as "Pending\_Approval"

var msg = "Invoice is submitted for the approval.";

MessageToast.show(msg);

this.byId("btnSubmit").setEnabled(false);

// statusIndContent[2].setState(invoicePostedStatus);

}

},

onAssignExcetionsPress: function () {

var token = this.\_fetchToken(); //Fetch Token for WF from bpmruntime

var context=sap.ui.getCore().invoiceDetailsWF;

if(\_hasNonDisplayExceptions){

MessageToast.show("Please clear all validation errors before assigning");

return;

}

if(context.displayExceptions && context.displayExceptions.results){

context.displayExceptions.results.forEach(function(ex){

});

}

if(context.displayExceptions && context.displayExceptions.results){

var assignToList=[];

context.displayExceptions.results.forEach(function(ex){

if(ex.exceptionStatus!=="SUCCESS" && !assignToList.indexOf(ex.assignTo)>=0){

assignToList.push(ex.assignTo);

}

});

if(token){

if(assignToList.length){

for(var i=0;i<assignToList.length;i++){

context.assignTo=assignToList[i];

this.\_startInstance(token,'dipsexception\_workflow',context); //Start WF Instance - My Inbox

}

sap.ui.getCore().invoiceStatus="In\_Exception";

this.\_saveInvoice();

this.\_setStatusIndicator(sap.ui.getCore().invoiceStatus);

MessageToast.show("Exceptions Submitted.");

}else{

MessageToast.show("No Exceptions to submit");

}

}

}

},

\_displayPostingErrors: function (postingErrorsJSON) {

sap.m.MessageBox.error("Error: " + postingErrorsJSON.postingErrors[0].errorMessage);

/\* var oPostingErrorTemplate = new sap.m.MessageItem({

type: '{type}',

title: '{title}',

description: '{description}',

subtitle: '{subtitle}',

counter: '{counter}',

markupDescription: '{markupDescription}'

});

var postingErrors = postingErrorsJSON.map(function (postingError) {

return {

Error: postingError.errorMessage

};

});

// sap.m.MessageBox.error("Error: "+ postingErrorsJSON.postingErrors[0].errorMessage);

// sap.m.MessageBox.error(JSON.stringify(postingErrorsJSON[0].errorMessage));

var oPostingErrorsMessageView = new MessageView({

showDetailsPageHeader: false,

itemSelect: function () {

oBackButton.setVisible(true);

},

items: {

path: "/",

template: oMessageTemplate

}

});

var oBackButton = new sap.m.Button({

icon: sap.ui.core.IconPool.getIconURI("nav-back"),

visible: false,

press: function () {

oPostingErrorsMessageView.navigateBack();

this.setVisible(false);

}

});

oPostingErrorsMessageView.setModel(new JSONModel(postingErrors));

var oPostingErrorsDialog = new sap.m.Dialog({

resizable: true,

content: oPostingErrorsMessageView,

state: 'Error',

beginButton: new sap.m.Button({

press: function () {

this.getParent().close();

},

text: "Close"

}),

customHeader: new sap.m.Bar({

contentMiddle: [

new sap.m.Text({

text: "Posting Errors"

})

],

contentLeft: [oBackButton]

}),

contentHeight: "300px",

contentWidth: "500px",

verticalScrolling: false

});

oPostingErrorsDialog.open();\*/

},

onPageLoadForWorkflow: function () {

this.byId("btnApprove").setVisible(true);

this.byId("btnReject").setVisible(true);

this.byId("btnApprove").setEnabled(true);

this.byId("btnReject").setEnabled(true);

this.byId("btnEdit").setEnabled(true);

this.byId("btnSave").setVisible(true);

this.byId("btnSubmit").setVisible(false);

this.byId("btnDelete").setVisible(false);

this.byId("btnCopy").setVisible(false);

this.byId("btnBackTop").setEnabled(false);

this.byId("btnBack").setEnabled(false);

if (\_hasExceptions) {

this.byId("btnApprove").setEnabled(false);

}

// Get the PONumber and make poItemTab or nonpoItemtab visible depending on selection of the invoice from the list - Initial load of Detail view

// var poNumber = sap.ui.getCore().purchaseOrder;

this.\_setPoNonPoTabVisibility(false);

},

// Post the Invoice to SAP System

onApprovePress: function (oEvent) {

// var token = this.\_fetchToken();

// this.\_startInstance(token);

// var msg = "Invoice is submitted for the approval.";

// MessageToast.show(msg);

// var odata\_service\_url = "/HDB\_SCP/com/deloitte/apps/invoiceprocessor/services/postInvoice/postInvoice.xsjs";

var invoiceDetails = new JSONModel();

invoiceDetails = this.getView().getModel("invoiceDetails");

// var context = JSON.parse(invoiceDetails);

var invoiceItems = {};

// var invoiceItemsNew = {};

// var Items = [];

var JSONString;

// invoiceItems.invoiceID = invoiceDetails.getProperty("/invoiceID");

invoiceItems.invoiceGUID = invoiceDetails.getProperty("/invoiceGUID");

JSONString = JSON.stringify(invoiceItems);

var currentViewController = this;

// var servicePayloadJSON = JSON.stringify({

// invoiceGUID: sap.ui.getCore().invoiceGUID

// });

$.ajax({

url: POST\_INVOICE\_SERVICE,

method: "POST",

contentType: 'application/json',

dataType: 'json',

// async : false,

// data: servicePayloadJSON,

data: JSONString,

headers: {

"Accept": "application/json",

"Access-Control-Allow-Origin": "\*"

},

success: function (responseData, textStatus, jqXHR) {

// sap.m.MessageToast.show('Data Created Successfully');

// var jsonUpdatedData = data;

if (responseData.postingErrors && responseData.postingErrors.length) {

// currentViewController.\_displayPostingErrors(responseData.postingErrors[0]);

sap.m.MessageBox.error("Error: " + responseData.postingErrors[0].errorMessage);

}

if (responseData.postedInvoice) {

sap.m.MessageBox.information(

"Invoice: " + responseData.postedInvoice + " successfully posted"

);

// Refresh the list of tasks on WF - Remove the WF task from list

var sTaskId = sap.ui.getCore.sTask;

currentViewController.\_completeUserTask(sTaskId, {status:"COMPLETED",context:{approved:true}});

// Make all the buttons disabled after you approve/post to SAP.

currentViewController.byId("btnApprove").setEnabled(false);

currentViewController.byId("btnReject").setEnabled(false);

currentViewController.byId("btnEdit").setEnabled(false);

currentViewController.byId("btnSave").setEnabled(false);

currentViewController.byId("btnCancel").setEnabled(false);

// currentViewController.byId("btnBackTop").setEnabled(false);

// currentViewController.byId("btnBack").setEnabled(false);

}

// Make back buttons disabled on "role"= approver screen while posting to SAP.

currentViewController.byId("btnBackTop").setEnabled(false);

currentViewController.byId("btnBack").setEnabled(false);

// Refresh the list of tasks on WF - Remove the WF task from list

// var sTaskId = currentViewController.getCurrentWFTask();

// var sTaskId = sap.ui.getCore.sTask;

// currentViewController.\_completeApproval(sTaskId, true);

},

error: function (jqXHR, textStatus, errorThrown) {

sap.m.MessageToast.show("Invoice Posting Failed: " + errorThrown.toString());

}

});

},

// Loads from WF Inbox Task - Reject with comments

onRejPress: function (oEvent) {

var currentViewController = this;

var sText;

var dialog = new sap.m.Dialog({

title: 'Reject',

type: 'Message',

content: [

new sap.m.Label({ text: 'Are you sure you want to reject the invoice?', labelFor: 'rejectDialogTextarea'}),

new sap.m.TextArea('rejectDialogTextarea', {

width: '100%',

placeholder: 'Add note (optional)'

})

],

beginButton: new sap.m.Button({

text: 'Reject',

press: function () {

sText = sap.ui.getCore().byId('rejectDialogTextarea').getValue();

// sap.m.MessageToast.show('Note is: ' + sText);

currentViewController.\_postComments(sText);

dialog.close();

}

}),

endButton: new sap.m.Button({

text: 'Cancel',

press: function () {

dialog.close();

}

}),

afterClose: function() {

dialog.destroy();

}

});

dialog.open();

},

// POST call to save reject comments

\_postComments: function (sText) {

var currentViewController = this;

var serviceRejPayloadJSON = JSON.stringify({

invoiceGUID: sap.ui.getCore.iGUIDFromWFTask,

responseType: "Rejected",

comments:sText

});

$.ajax({

type: 'POST', //GET or POST or PUT or DELETE verb

async : false,

data: serviceRejPayloadJSON, //Data sent to server

url: CAPTURE\_REJECTCOMMENT\_SERVICE, // Location of the service

contentType: 'application/json',

dataType: 'json', //Expected data format from server

success: function(responseData, textStatus, xhr){ //On Successfull service call

if (textStatus === "success") {

sap.m.MessageBox.information("The invoice got rejected");

}

//Complete the task with status as Reject / Remove from My Inbox with a refresh

var sTaskId = sap.ui.getCore.sTask;

currentViewController.\_completeUserTask(sTaskId, {status:"COMPLETED",context:{approved:true}});

// Make all the buttons disabled after someone Reject the invoice.

currentViewController.byId("btnApprove").setEnabled(false);

currentViewController.byId("btnReject").setEnabled(false);

currentViewController.byId("btnEdit").setEnabled(false);

currentViewController.byId("btnSave").setEnabled(false);

currentViewController.byId("btnCancel").setEnabled(false);

currentViewController.byId("btnBackTop").setEnabled(false);

currentViewController.byId("btnBack").setEnabled(false);

},

error: function (xhr) {

// console.log(xhr.responseText);

MessageBox.error(xhr.responseText);

} // When Service call fails

});

},

\_fetchToken: function () {

var token;

$.ajax({

url: "/bpmworkflowruntime/rest/v1/xsrf-token",

method: "GET",

async: false,

headers: {

"X-CSRF-Token": "Fetch"

},

success: function (result, xhr, data) {

token = data.getResponseHeader("X-CSRF-Token");

}

});

return token;

},

\_startInstance: function (token,workflowName,workflowContext) {

var model = this.getView().getModel();

var workflow=workflowName||"dipsapprovers\_workflow";

var obj = {};

obj = workflowContext||sap.ui.getCore().invoiceDetailsWF;

$.ajax({

url: "/bpmworkflowruntime/rest/v1/workflow-instances",

method: "POST",

async: false,

contentType: "application/json",

headers: {

"X-CSRF-Token": token

},

data: JSON.stringify({

definitionId: workflow, //workflow id

context: obj

}),

success: function (result, xhr, data) {

model.setProperty("/result", JSON.stringify(result, null, 4));

}

});

},

\_completeUserTask: function (sTaskId, data) {

var sWorkflowToken = this.\_fetchToken();

var fnRemoveFromTaskList = function () {

// remove task from the approver 1 list

return new Promise(function (resolve, reject) {

$.ajax({

url: "/bpmworkflowruntime/rest/v1/task-instances/" + sTaskId,

method: "PATCH",

contentType: "application/json",

async: false,

data: JSON.stringify(data),

headers: {

"X-CSRF-Token": sWorkflowToken

},

success: function () {

resolve();

// currentViewController.getComponentData().startupParameters.inboxAPI.updateTask("NA", sTaskId);

}

});

});

};

// My Inbox - Tasklist Auto Refresh

var fnRefreshTask = function () {

// this.getComponentData().startupParameters.inboxAPI.updateTask("NA", sTaskId);

sap.m.MessageToast.show("WF Task Completed");

}.bind(this);

fnRemoveFromTaskList().then(fnRefreshTask());

},

// POST method for Invoice Status- "Pending Approval" after Submission of Invoice for Approval

\_setPendingApprvalStatusOnSubmit: function () {

var currentViewController = this;

var servicePayloadJSON\_PendingApproval = JSON.stringify({

invoiceGUID: sap.ui.getCore().invoiceGUIDFromList

});

$.ajax({

type: 'POST', //GET or POST or PUT or DELETE verb

async : false,

data: servicePayloadJSON\_PendingApproval, //Data sent to server

url: POST\_PENDING\_APPROVAL\_STATUS\_onSubmit, // Location of the service

contentType: 'application/json',

dataType: 'json', //Expected data format from server

success: function(responseData, textStatus, xhr){ //On Successfull service call

if (textStatus === "success") {

// Make all the buttons disabled after submit the invoice for Approval.

currentViewController.byId("btnEdit").setEnabled(false);

currentViewController.byId("btnSave").setEnabled(false);

currentViewController.byId("btnCancel").setEnabled(false);

// return "success";

}

},

error: function (xhr) {

// console.log(xhr.responseText);

MessageBox.error(xhr.responseText);

} // When Service call fails

});

},

\_setPoNonPoTabVisibility:function(isEdit){

//window.\_myctrl=this;

var ctrlList=[this.byId("poItemTab"),this.byId("poItemTabDisp"),this.byId("nonpoItemTabDisp"),this.byId("poItemTabEdit"),this.byId("nonpoItemTab"),this.byId("nonpoItemTabEdit")];

ctrlList.forEach(function(ctrl){

if(ctrl){

ctrl.setVisible(false);

}

});

if (sap.ui.getCore().purchaseOrder || sap.ui.getCore().poNumDetailSceenFromWF) {

if(isEdit){

this.byId("poItemTabEdit").setVisible(true);

}else{

this.byId("poItemTab").setVisible(true);

if(this.byId("poItemTabDisp")){

this.byId("poItemTabDisp").setVisible(true);

}

}

} else {

if(isEdit){

this.byId("nonpoItemTabEdit").setVisible(true);

}else{

this.byId("nonpoItemTab").setVisible(true);

if(this.byId("nonpoItemTabDisp")){

this.byId("nonpoItemTabDisp").setVisible(true);

}

}

}

}

});

});