/\*eslint-disable no-console, no-alert \*/

sap.ui.define([

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

"sap/m/MessageToast",

"sap/ui/core/Fragment",

"sap/ui/model/json/JSONModel"

], function (Controller, MessageToast, Fragment, JSONModel) {

"use strict";

//jQuery.sap.require("sap.suite.ui.commons.MicroProcessFlow");

//jQuery.sap.require("sap.suite.ui.commons.MicroProcessFlowItem");

//sap.ui.define("sap/suite/ui/commons/MicroProcessFlow");

//sap.ui.define("sap/suite/ui/commons/MicroProcessFlowItem");

var months = ["January", "February", "March", "April", "May", "June", "July", "August", "September", "October", "November",

"December"

];

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

onInit: function () {

// Save the reference to the current view in the app context to be used later within event listeners

sap.ui.getCore().appContext = new Object();

sap.ui.getCore().appContext.viewInvoiceList = this;

// this.setInitialSortOrder();

},

onInvoiceSelected: function (oEvent) {

var oSelectedInvoice = oEvent.getSource();

var oInvoiceDetails = oSelectedInvoice.getBindingContext().getObject();

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

sap.ui.getCore().purchaseOrder = oInvoiceDetails.purchaseOrder;

// Navigate to the detail page

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

oRouter.navTo("toInvoiceDetail", {

invoiceGUID: oInvoiceDetails.invoiceGUID

});

},

onUpdateFinished: function (oEvent) {

var oTable = oEvent.getSource();

var tableItems = oTable.getItems();

$.each(tableItems, function (index, tableItem) {

// Format the upload date

var oInvoiceUploadTS = tableItem.getCells()[3];

var uploadTS = new Date(oInvoiceUploadTS.getText());

// Set the formatted upload date

oInvoiceUploadTS.setText(months[uploadTS.getMonth()] + " " + uploadTS.getDate() + ", " + uploadTS.getFullYear() + " " +

uploadTS.toTimeString());

// Set the status indicator process flow accordingly

var invoiceStatusIndicator = tableItem.getCells()[2];

var invoiceStatusText = invoiceStatusIndicator.getItems()[0];

var invoiceStatusFlow = invoiceStatusIndicator.getItems()[1];

var invoiceStatus = invoiceStatusFlow.getAriaLabel();

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

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

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

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

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

switch (invoiceStatus) {

case "Scanned":

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

var statusText = "Scanned";

var statusIcon = "sap-icon://document-text";

var statusState = documentScanState;

break;

case "In\_Progress":

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

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

statusText = "In Progress";

statusIcon = "sap-icon://process";

statusState = inProcessState;

break;

case "In\_Exception":

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

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

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

statusText = "Exception";

statusIcon = "sap-icon://error";

statusState = exceptionState;

break;

case "Exception\_Complete":

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

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

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

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

statusText = "Exception Complete";

statusIcon = "sap-icon://accept";

statusState = exceptionCompletedState;

break;

case "Pending\_Approval":

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

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

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

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

statusText = "Pending Approval";

statusIcon = "sap-icon://accept";

statusState = exceptionCompletedState;

break;

case "Posted":

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

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

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

statusText = "Invoice Posted";

statusIcon = "sap-icon://accounting-document-verification";

statusState = invoicePostedStatus;

break;

}

// Set the status text, icon and state

invoiceStatusText.setText(statusText);

invoiceStatusText.setIcon(statusIcon);

invoiceStatusText.setState(statusState);

// Set the processflow with appropriate state assigned to the icons of the flow

var statusIndContent = invoiceStatusFlow.getContent();

statusIndContent[0].setState(documentScanState);

statusIndContent[1].setState(

inProcessState);

statusIndContent[2].setState(

exceptionState);

statusIndContent[3].setState(

exceptionCompletedState);

statusIndContent[4].setState(invoicePostedStatus);

});

// this.setInitialSortOrder();

},

//Function to Sort on columnkey

setInitialSortOrder: function () {

var oSmartTable = this.getView().byId("smtTblInvoicesList");

oSmartTable.applyVariant({

sort: {

sortItems: [{

columnKey: "invoiceUploadTS",// daysSinceUploaded

operation: "Descending"

}]

}

});

},

onSortPress: function () {

var smartTable = this.getView().byId("smtTblInvoicesList");

if (smartTable) {

smartTable.openPersonalisationDialog("Sort");

}

},

onFilterPress: function () {

var smartTable = this.getView().byId("smtTblInvoicesList");

if (smartTable) {

smartTable.openPersonalisationDialog("Filter");

}

},

onColumnSelectionPress: function () {

var smartTable = this.getView().byId("smtTblInvoicesList");

if (smartTable) {

smartTable.openPersonalisationDialog("Columns");

}

},

onAddInvoicesPress: function () {

this.\_getInvoiceUploadDialogFragment();

this.byId("uploadDialog").open();

},

\_getInvoiceUploadDialogFragment: function () {

// Instantiate the dialog fragment only once

if (typeof this.\_invoiceUploadDialogFragment === "undefined") {

var oCurrentView = this.getView();

this.\_invoiceUploadDialogFragment = sap.ui.xmlfragment(oCurrentView.getId(),

'com.deloitte.scp.app.dips.view.InvoiceUploadDialog', this);

oCurrentView.addDependent(this.\_invoiceUploadDialogFragment);

this.\_generateCSRFToken();

this.byId("ucInvoices").setModel(new JSONModel({

items: []

}));

}

var oDocModel = this.byId("ucInvoices").getModel();

oDocModel.getData().items = [];

oDocModel.refresh();

return this.\_invoiceUploadDialogFragment;

},

onDialogPress: function () {

//this.\_invoiceUploadDialogFragment.close();

this.byId("uploadDialog").close();

},

onUploadChange: function () {

var oUploadCollection = this.byId("ucInvoices");

oUploadCollection.addHeaderParameter(new sap.m.UploadCollectionParameter({

name: "X-CSRF-Token",

value: this.\_getCSRFToken()

}));

},

\_generateCSRFToken: function () {

if (typeof this.\_csrfToken === "undefined") {

var currentController = this;

$.get({

url: "/HDB\_SCP/coms/deloitte/apps/invoiceprocessor/services/invoiceUpload/invoiceUpload.xsjs",

headers: {

"X-CSRF-Token": "Fetch"

},

success: function (data, textStatus, xmlHttpRequest) {

currentController.\_csrfToken = xmlHttpRequest.getResponseHeader("X-CSRF-Token");

}

});

}

},

\_getCSRFToken: function () {

return this.\_csrfToken;

},

onBeforeUploadStarts: function () {

var oUploadCollection = this.byId("ucInvoices");

oUploadCollection.multiple = true;

},

onUploadComplete: function (event) {

var uploadedFiles = event.getParameters().files;

if (uploadedFiles && uploadedFiles.length) {

var filesData = uploadedFiles.map(function (file) {

var jResponse = JSON.parse(file.responseRaw);

return {

documentId: jResponse.id || file.fileName,

fileName: file.fileName || jResponse.name,

mimeType: jResponse.contentType,

url: jResponse.selfLink

};

});

var uploadItems = this.byId("ucInvoices").getModel().getData();

filesData.forEach(function (file) {

uploadItems.items.push(file);

});

this.byId("ucInvoices").getModel().refresh();

}

}

});

});