<aura:component controller="AdressAutoComplete" implements="flexipage:availableForAllPageTypes,force:lightningQuickAction,force:hasRecordId,force:hasSObjectName,force:appHostable,lightning:actionOverride" access="global">

<aura:attribute name="location" type="string" default=""/>

<aura:attribute name="ld" type="Account" default="{ 'sobjectType': 'Account' }"/>

<aura:attribute name="item" type="List" default="[]" />

<aura:attribute name="recordId" type="String"/>

<aura:attribute name="predictions" type="List" default="[]"/>

<aura:attribute name="onSuccessMessage" type="string" default=""/>

<aura:if isTrue="{!v.predictions.length > 0}">

<ul class="city\_predictions">

<aura:iteration items="{!v.predictions}" var="prediction" >

<li class="slds-listbox\_\_item">

<a onclick="{!c.getCityDetails}" data-placeid="{!prediction.place\_id}">{!prediction.description}</a>

</li>

</aura:iteration>

</ul>

</aura:if>

<lightning:recordEditForm aura:id="recordViewForm" onsuccess="{!c.submitRecord}" recordId="{!v.recordId}" objectApiName="Account">

<lightning:messages />

<br/>

<h3 class="slds-section\_\_title slds-theme\_shade">

<span style="font-weight: bold;">&nbsp; Billing Address</span>

</h3>

<lightning:input label=""

name="Location"

aura:id="location"

value="{!v.location}"

onchange="{!c.getCities}"

placeholder="Search Accounts..."/>

<lightning:layout multipleRows="true">

<lightning:layoutItem padding="around-small" size="4">

<lightning:inputField fieldName="BillingStreet" aura:Id="BillingStreet"/>

</lightning:layoutItem>

<lightning:layoutItem padding="around-small" size="4">

<lightning:inputField fieldName="BillingState" aura:Id="BillingState"/>

</lightning:layoutItem>

<lightning:layoutItem padding="around-small" size="4">

<lightning:inputField fieldName="BillingCity" aura:Id= "BillingCity"/>

</lightning:layoutItem>

<lightning:layoutItem padding="around-small" size="4">

<lightning:inputField fieldName="BillingPostalCode" aura:Id="BillingCountry"/>

</lightning:layoutItem>

<lightning:layoutItem padding="around-small" size="4">

<lightning:inputField fieldName="BillingCountry" aura:Id="BillingPostalCode"/>

</lightning:layoutItem>

</lightning:layout>

<lightning:recordEditForm aura:id="recordViewForm" onsuccess="{!c.submitRecord}" recordId="{!v.recordId}" objectApiName="Account">

<lightning:messages />

<br/>

<h3 class="slds-section\_\_title slds-theme\_shade">

<span style="font-weight: bold;">&nbsp; Shipping Address</span>

</h3>

<lightning:input label=""

name="location"

aura:id="location2"

value="{!v.location}"

onchange="{!c.getCities}" />

<lightning:layout multipleRows="true">

<lightning:layoutItem padding="around-small" size="4">

<lightning:inputField fieldName="ShippingStreet" aura:Id="ShippingStreet"/>

</lightning:layoutItem>

<lightning:layoutItem padding="around-small" size="4">

<lightning:inputField fieldName="ShippingState" aura:Id="ShippingState"/>

</lightning:layoutItem>

<lightning:layoutItem padding="around-small" size="4">

<lightning:inputField fieldName="ShippingCity" aura:Id= "ShippingCity"/>

</lightning:layoutItem>

<lightning:layoutItem padding="around-small" size="4">

<lightning:inputField fieldName="ShippingPostalCode" aura:Id="ShippingCountry"/>

</lightning:layoutItem>

<lightning:layoutItem padding="around-small" size="4">

<lightning:inputField fieldName="ShippingCountry" aura:Id="ShippingPostalCode"/>

</lightning:layoutItem>

</lightning:layout>

</lightning:recordEditForm>

<lightning:button aura:id="submit" type="submit" label="Submit" class="slds-m-top\_medium" />

{!onSuccessMessage}

</lightning:recordEditForm>

</aura:component>

({

getCities : function(component, event, helper){

debugger;

var params = {

"input" : component.get('v.location'),

}

helper.callServer(component,"c.getSuggestions",function(response){

var resp = JSON.parse(response);

console.log(resp.predictions);

component.set('v.predictions',resp.predictions);

var arr = resp.predictions;

for(var obj in arr){

console.log('Inside loop');

console.log(arr[obj]);

}

},params);

},

getCityDetails : function(component, event, helper){

component.set('v.predictions',[]);

var selectedItem = event.currentTarget;

var placeid = selectedItem.dataset.placeid;

var street = '';

var city = '';

var state = '';

var country = '';

var zip = '';

var shippingstreet = '';

var shippingcity = '';

var shippingstate = '';

var shippingcountry = '';

var shippingzip = '';

var params = {

"placeId" : placeid

}

helper.callServer(component,"c.getPlaceDetails",function(response){

debugger;

var placeDetails = JSON.parse(response);

var details = placeDetails.result.address\_components;

for(var obj1 in details){

console.log('\*\*\*\*\*\*Selected Address\*\*\*\*');

console.log(details[obj1]);

if(details[obj1].types.includes("street\_number") || details[obj1].types.includes("route")){

street = street + ' '+ details[obj1].short\_name;

}

if(details[obj1].types.includes("locality")){

city = details[obj1].long\_name;

}

if(details[obj1].types.includes("administrative\_area\_level\_1")){

state = details[obj1].long\_name;

}

if(details[obj1].types.includes("country")){

country = details[obj1].short\_name;

}

if(details[obj1].types.includes("postal\_code")){

zip = details[obj1].long\_name;

}

}

component.find("BillingStreet").set("v.value", street);

component.find("BillingCity").set("v.value", city);

component.find("BillingState").set("v.value", state);

component.find("BillingPostalCode").set("v.value", country);

component.find("BillingCountry").set("v.value", zip);

component.find("ShippingStreet").set("v.value", shippingstreet);

component.find("ShippingCity").set("v.value", shippingcity);

component.find("ShippingState").set("v.value", shippingstate);

component.find("ShippingPostalCode").set("v.value", shippingcountry);

component.find("ShippingCountry").set("v.value", shippingzip);

component.set('v.item',details);

//component.set('v.predictions',[]);

},params);

},

submitRecord: function(cmp, event, helper) {

debugger;

cmp.set("v.onSuccessMessage", "Address updated successfully");

var accId = cmp.get("v.recordId");

window.open("/"+accId, "\_self");

},

})

({

callServer : function(component,method,callback,params) {

var action = component.get(method);

if (params) {

action.setParams(params);

}

action.setCallback(this,function(response) {

var state = response.getState();

if (state === "SUCCESS") {

// pass returned value to callback function

callback.call(this,response.getReturnValue());

var result= response.getReturnValue();

var resp = JSON.parse(result);

console.log(resp);

//alert(resp);

} else if (state === "ERROR") {

// generic error handler

var errors = response.getError();

if (errors) {

console.log("Errors", errors);

if (errors[0] && errors[0].message) {

throw new Error("Error" + errors[0].message);

}

} else {

throw new Error("Unknown Error");

}

}

});

$A.enqueueAction(action);

}

})