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

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

<aura:attribute name="location\_MA" 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\_BA" type="List" default="[]"/>

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

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

<lightning:layout>

<lightning:layoutItem padding="around-small" size="1">

</lightning:layoutItem>

<lightning:layoutItem padding="around-small" size="10">

<div class="slds-box">

<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"

id="location\_BA"

value="{!v.location\_BA}"

onchange="{!c.getPredictions}"

placeholder="start typing..."/>

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

<ul class="slds-list--vertical">

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

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

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

</li>

</aura:iteration>

</ul>

</aura:if>

<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>

<br/>

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

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

</h3>

<lightning:input label=""

name="location"

id="location\_MA"

value="{!v.location\_MA}"

onchange="{!c.getPredictions}"

placeholder="start typing..."/>

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

<ul class="slds-list--vertical">

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

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

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

</li>

</aura:iteration>

</ul>

</aura:if>

<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>

<div class="slds-align\_absolute-center">

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

</div>

{!onSuccessMessage}

</lightning:recordEditForm>

</div>

</lightning:layoutItem>

<lightning:layoutItem padding="around-small" size="1">

</lightning:layoutItem>

</lightning:layout>

</aura:component>

({

getPredictions : function(component, event, helper){

debugger;

var inputEle = event.getSource().get("v.id");

var vinput = "";

if(inputEle == "location\_BA"){

vinput = component.get('v.location\_BA');

} else if(inputEle == "location\_MA"){

vinput = component.get('v.location\_MA');

}

var params = {

"input" : vinput,

}

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

var resp = JSON.parse(response);

console.log(resp.predictions);

if(inputEle == "location\_BA"){

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

} else if(inputEle == "location\_MA"){

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

}

},params);

},

getBASpecificDetails : function(component, event, helper){

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

var selectedItem = event.currentTarget;

var placeid = selectedItem.dataset.placeid;

var street = '';

var city = '';

var state = '';

var country = '';

var zip = '';

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

},params);

},

getMASpecificDetails : function(component, event, helper){

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

var selectedItem = event.currentTarget;

var placeid = selectedItem.dataset.placeid;

var street = '';

var city = '';

var state = '';

var country = '';

var zip = '';

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("ShippingStreet").set("v.value", street);

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

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

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

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

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

},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);

}

})

Apex Class:

public class AdressAutoComplete {

@AuraEnabled

public static string getSuggestions(String input) {

system.debug('input'+ input);

/\*string url ='http://cdldvabivap0001.es.ad.adp.com:8095/DataIntegrationService/WebService/SOAP\_Web\_Service\_Sugg';\*/

String url = 'https://maps.googleapis.com/maps/api/place/queryautocomplete/json?input='+ EncodingUtil.urlEncode(input, 'UTF-8')+'&key=AIzaSyC6jJbJREXUfG82xOUaLuB6cTqpYg9hsig' ;

/\*+ '&input=' + EncodingUtil.urlEncode(input, 'UTF-8');\*/

/\* String url = 'https://maps.googleapis.com/maps/api/place/autocomplete/json?'

+ 'input=' + EncodingUtil.urlEncode(input, 'UTF-8')

+ '&types=(regions)' + getKey();\*/

String response = getResponse(url);

system.debug('response'+response);

return response;

}

@AuraEnabled

public static string getPlaceDetails(String placeId) {

String url = 'https://maps.googleapis.com/maps/api/place/details/json?'

+ 'placeid=' + EncodingUtil.urlEncode(placeId, 'UTF-8')

+ getKey();

String response = getResponse(url);

return response;

}

public static string getResponse(string strURL){

Http h = new Http();

HttpRequest req = new HttpRequest();

HttpResponse res = new HttpResponse();

req.setMethod('GET');

req.setEndpoint(strURL);

req.setTimeout(120000);

res = h.send(req);

String responseBody = res.getBody();

return responseBody;

}

public static string getKey(){

string key = 'AIzaSyC6jJbJREXUfG82xOUaLuB6cTqpYg9hsig';

string output = '&key=' + key;

return output;

}

}

Visualforce page:

<apex:page standardController="Account" lightningStylesheets="true" >

<apex:includeScript value="/lightning/lightning.out.js"/>

<apex:includeLightning />

<apex:slds />

<div id="LcDisplayId"></div>

<script>

var accId = '{!$CurrentPage.parameters.Id}';

$Lightning.use("c:testApp", function() {

/\* 'LcForVf' is Lightning Component Name which we are Displaying In Vf Page

\* syntax for create lightning component dynamically :

\* $Lightning.createComponent(String type, Object attributes, String locator, function callback) \*/

$Lightning.createComponent("c:AddressAutoComplete",

{

recordId : accId

},

"LcDisplayId",

function(component) {

// create component Callback, Lightning Component has been Created,

// Now you can set more lightning Component attributes here,

// and do more cool stuff here

});

});

</script>

</apex:page>

Application

<aura:application access="GLOBAL" extends="ltng:outApp">

<!--<c:NewAddress/>-->

<c:AddressAutoComplete/>

<!--<aura:dependency resource="c:NewAddress1"/>-->

</aura:application>