Hi Friends,

For the reason of integrating two orgs which in our case between two separate Salesforce orgs, we have to assume one org as Client or Source and another as Server or Destination, therefore data from Account object at Source org will be pushed in Account object Data at Destination Org.

Prerequisites:

**Source org endpoint:** [**https://akashmishra-dev-ed.my.salesforce.com/**](https://akashmishra-dev-ed.my.salesforce.com/) **(Don’t get confused as this is the custom url I used, you can use the Instance url for your org like:** [**https://ap5.salesforce.com**](https://ap5.salesforce.com)**).**

**Destination Org URL:** [**https://ap4.salesforce.com**](https://ap4.salesforce.com)

First we talk what we have to do in the in the Destination org:

Step 1. Create a Connected app in Destination org giving the callback url with below details as mentioned in the image



Now don’t forget to copy the Consumer Key (Client id) and Consumer Secret (Client Secret) we’ll need these in Source org Rest Callout Class.

Step 2. Create a Rest Webservice with @httpPost verb, Though it’s not required in this case. Find the class below.

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

 \* Description - Apex REST service with GET and POST methods

 \* Author - AP

{

"name" : "Akash",

"phone" : "8826031286",

"website" : "www.akashmishra.co.in"

}

 \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

@RestResource(urlMapping='/v3/accounts/\*')

global with sharing class REST\_AccountService\_V4 {

 @HttpPost

 global static AccountWrapper doPost(String name, String phone, String website) {

 RestRequest req = RestContext.request;

 RestResponse res = RestContext.response;

 AccountWrapper response = new AccountWrapper();

 Account acct = new Account();

 acct.Name = name;

 acct.Phone = phone;

 acct.Website = website;

 insert acct;

 response.acctList.add(acct);

 response.status = 'Success';

 response.message = 'Your Account was created successfully.';

 return response;

 }

 @HttpGet

 global static AccountWrapper doGet() {

 RestRequest req = RestContext.request;

 RestResponse res = RestContext.response;

 AccountWrapper response = new AccountWrapper();

 String accountId = req.requestURI.substring(req.requestURI.lastIndexOf('/')+1);

 if(doSearch(accountId)) {

 searchAccounts(req, res, response);

 }

 else {

 findAccount(res, response, accountId);

 }

 return response;

 }

 // If the item to the right of the last forward slash is "accounts", the request went to v3/accounts?Name=United

 // Else the request went to v3/accounts/<something>, which is not a search, but a specific entity

 private static boolean doSearch(String accountId) {

 if(accountId == 'accounts') {

 return true;

 }

 return false;

 }

 //If the request came to /v3/accounts, then we want to execute a search

 private static void searchAccounts(RestRequest req, RestResponse res, AccountWrapper response) {

 //Use the RestRequest's params to fetch the Name parameter

 String searchTerm = req.params.get('Name');

 if(searchTerm == null || searchTerm == '') {

 response.status = 'Error';

 response.message = 'You must provide a Name for your search term.';

 res.StatusCode = 400;

 }

 else {

 String searchText = '%'+searchTerm+'%';

 List<Account> searchResults = [SELECT Id, Name, Phone, Website FROM Account WHERE Name LIKE : searchText];

 if(searchResults != null && searchResults.size() > 0) {

 response.acctList = searchResults;

 response.status = 'Success';

 response.message = searchResults.size() + ' Accounts were found that matched your search term.';

 }

 else {

 response.status = 'Error';

 response.message = 'No Accounts where found based on that Name, please search again.';

 }

 }

 }

 //If the request came to v3/accounts/<external\_Id>, then we want to find a specific account

 private static void findAccount(RestResponse res, AccountWrapper response, String accountId) {

 // Provided we recevied an External Id, perform the search and return the results

 if(accountId != null && accountId != '') {

 List<Account> result = [SELECT Id, Name, Phone, Website FROM Account WHERE External\_Id\_\_c =: accountId];

 if(result != null && result.size() > 0) {

 response.acctList.add(result[0]);

 response.status = 'Success';

 }

 else {

 response.status = 'Error';

 response.message = 'This account could not be found, please try again.';

 res.StatusCode = 404;

 }

 }

 // If the request came to /v3/accounts/ (without an Account Id), return an error

 else {

 response.status = 'Error';

 response.message = 'You must specify an External Id.';

 res.StatusCode = 400;

 }

 }

 global class AccountWrapper {

 public List<Account> acctList;

 public String status;

 public String message;

 public AccountWrapper(){

 acctList = new List<Account>();

 }

 }

}

Note: Please ignore the Naming conventions as they may look funny.

\*\*\*

Now let’s configure the Source org:

Step 1. Create Remote site in Source like in my case I created the below one.



Step 2. Create a Rest Callout in the Source org.

Goto: Create🡺Apex Class🡺Click New

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*Description: Rest Callout

\*Owner: AP

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

public class AgainSendAccountUsingRestApi {

String clientId

=3MVG9YDQS5WtC11o4mAP\_I\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*OoLrKY3FUrb6221Ms9hjkqhZo\_oTNbjKu2LJBHN\_tFmZ1.'; //Use your Client Id

 String clientsecret='314426\*\*\*\*\*\*97831'; //Use your Client Secret

 string username='akash4trailhead@gmail.com'; //Your Destination org username

 String password='Qwerty@10064OMwL3ZNSh6WP5lVqRxapaHUK'; //Your Destination org Password

 String accesstoken\_url='https://login.salesforce.com/services/oauth2/token';

 String authurl='https://login.salesforce.com/services/oauth2/authorize';

 public class deserializeResponse{

 public String id;

 public String access\_token;

 }

 public String ReturnAccessToken(AgainSendAccountUsingRestApi Acc){

 String reqbody = 'grant\_type=password&client\_id='+clientId+'&client\_secret='+clientSecret+'&username='+username+'&password='+password;

 // String reqbody='{"grant\_type":"password","client\_id":clientId,"client\_secret":clientSecret,"username":username,"password":password}';

 Http h= new Http();

 HttpRequest req= new HttpRequest();

 req.setBody(reqbody);

 req.setMethod('POST');

 req.setEndpoint('https://ap4.salesforce.com/services/oauth2/token'); //Change "ap4" in url to your Target Org Instance

 HttpResponse res=h.send(req);

 System.debug(res.getBody()+'By-AP-1986-Response');

 deserializeResponse resp1=(deserializeResponse)JSON.deserialize(res.getBody(),deserializeResponse.class);

 System.debug(resp1+'By-AP-deserializeresponse');

 return resp1.access\_token;

 }

 @future(callout=true)

 public static void createAccount(String Accname, String Phone, String Website){

 AgainSendAccountUsingRestApi acc1= new AgainSendAccountUsingRestApi();

 String accessToken=acc1.ReturnAccessToken(acc1);

 System.debug(accessToken+'###AP');

 if(accessToken!=null){

 String endPoint='https://ap4.salesforce.com/services/data/v32.0/sobjects/Account'; //Change "ap4" in url to your Destination Org

 String jsonstr='{"Name":"'+ Accname +'","Phone":"'+ Phone +'","Website":"'+ Website +'"}';

 Http h2= new Http();

 HttpRequest req2= new HttpRequest();

 req2.setHeader('Authorization','Bearer ' + accessToken);

 req2.setHeader('Content-Type','application/json');

 req2.setHeader('accept','application/json');

 req2.setBody(jsonstr);

 req2.setMethod('POST');

 req2.setEndpoint(endPoint);

 HttpResponse res2=h2.send(req2);

 System.debug(res2+'###1203createresp');

 deserializeResponse deresp2=(deserializeResponse)System.JSON.deserialize(res2.getBody(),deserializeResponse.class);

 System.debug('###1203createdeser'+deresp2);

 }

 }

 }

Now Create a Trigger to make callout when some data is inserted in the Account Object.

Try This:

trigger SendAccount on Account (after insert)
{
    for(Account a:Trigger.new)
    {
        AgainSendAccountUsingRestApi.createAccount(a.name, a.Phone, a.Website);
    }
}

Now save and start testing by creating a record on Account Object in Source org.

Hope you got it right.

Thanks everyone,

Akash Mishra (AP)

**NOTE: Please don’t use this information for blog as I already have planned for the same.**