

API LEARNING FULL PICTURE FROM SCRATCH

Simple Web callout:-

Child Class: (Processing) ::

URL defined, Method defined

```
@RestResource(urlMapping='/ContactFetcher/*')
```

```
global class childClass {
```

```
//information processing
```

```
    @HttpGet
```

```
    global static List<Contact> childClassMethod(){
```

```
        List<Contact> li = [SELECT Id, Name, Phone, Account.Name FROM Contact];
```

```
        System.debug('List of Contact==>' + li);
```

```
        return li;
```

```
    }
```

```
//Data Creating
```

```
    /* @HttpPost
```

```
    global static List<Contact> childClassMethod2(){
```

```
        List<Contact> li = [SELECT Id, Name, Phone, Account.Name FROM Contact];
```

```
        System.debug('List of Contact==>' + li);
```

```
        return li;
```

```
    }*/
```

```
}
```

Target Org Configuration: Child:-

Connected app

- 1) Client_id
- 2) Consumer_key
- 3) OAUTH Policy_ relaxation of ip network

How to prevent connected Auth usage?

Setup > Connected App -> Connected app Oauth usage -> Manage and block

Parent Class: (schedule setup, schedule, batch, state) ::

In batch: Request and Response

For Request: (header and data combined in a wrapper)

For Request:-

- 1) Endpoint url
- 2) Method
- 3) Authorized access token

Note:- Authorized access token, only needed :- If data is not publicly available.

If data is publicly available then directly use **fetch** that's it.

If data is not publicly available and to get Authorized access token, do the authentication:-

For Normal Authentication, Need :-

If data/ API that is publicly available but need client key in parameter

For OAUTH Authentication, Need:-

User name, password, security token, client_id, client_key

In Salesforce for OAUTH Authentication:automatic, manual processes are there.

For Response:

```
System.debug('>>res '+res.getBody());  
List<Object> resultMap= (List<Object>)JSON.deserializeUntyped(res.getBody());  
System.debug('resultMap==>' + resultMap);
```

Source Org Configuration: Parent:-

Remote site setting

<https://login.salesforce.com>

<https://brave-bear-1zteug-dev-ed.my.salesforce.com>

For Automatic Auth process : Auth Provider and Named credential setting

/* ----- OAUTH AUTHENTICATION IN SALESFORCE ----- */

// for automatic setup use this mechanism

// Connected App in target org, callback of source org,

// Auth provider is source org, named Cred in source org

// for manual:-

//session id and access url

//set session id and access url on header

//call out

For manual process:

Do Authentication: Need userName, password, Security Token extra.

```
public AuthRes authenticationMethod(){
    Http h = new Http();
    HttpRequest hrequest = new HttpRequest();
    HttpResponse hresponse = new HttpResponse();

    String userName = 'dhanuka.shubham@brave-bear-1zteug.com';
    String password = 'Dhanuka12@hlfGkKHB0jlwjpzaaEF4KbV';
    String client_id = '3MVG9fe4g9fhX0E5aqHBK6CdKceKovLT_TWq1.1jhZ5AbGYcqNvz_x0JjrU3qnn5VAUudB3xjEkB8oElh4p8o';
    String client_key = 'FB3201A1368CAFE51BDBF93A8897AA6A655F317CCDBBFE3368763F5A29161D8F';

    String auth = 'grant_type=password&client_id='+client_id+'&client_secret='+client_key+'&username='+userName+'&password='+password;
    String endpoint = 'https://login.salesforce.com/services/oauth2/token';

    hrequest.setEndpoint(endpoint);
    hrequest.setMethod('POST');
    hrequest.setBody(auth);

    hresponse = h.send(hrequest);

    if (hresponse.getStatusCode() == 200) {
        System.debug('hresponse ==>' + hresponse);
        System.debug('response body==>' + hresponse.getBody());
    }
    ParentClass1.AuthRes authRes = new ParentClass1.AuthRes();
    authRes = (AuthRes)JSON.deserialize(hresponse.getBody(),AuthRes.class);
    return authRes;
}

public class AuthRes{
    public string access_token;
    public string instance_url;
    public string id;
    public string token_type;
    public string issued_at;
    public string signature;
}
```

Then in request, set header

req.setHeader('Authorization','Bearer '+authRes.access_token);

For automatic setup use this mechanism:

- 1) Define auth provider and Named credentials.
- 2) In end point, Define like this

```
request.setEndpoint('callout:Apex_Rest_Services_Test/services/apexrest/retrieveDeliveries');
```

Then u just need to define set method and send the request

No need to define access token, Named credential and oauth provider will do the job

Example:-

```
Http http = new Http();
HttpRequest request = new HttpRequest();
//Set timeout to 1 minute to avoid read timed out error (only if it appears)
request.setTimeout(60000);
request.setEndpoint('callout:Apex_Rest_Services_Test/services/apexrest/retrieveDeliveries');
request.setMethod('GET');

HttpResponse response = http.send(request);
while (response.getStatusCode() == 302) {
    request.setEndpoint(response.getHeader('Location'));
    response = new Http().send(request);
}

// If the request is successful, parse the JSON response.
System.debug(response.getBody());
```

Example of Automatic OAUTH in Salesforce:-

Target Org:-

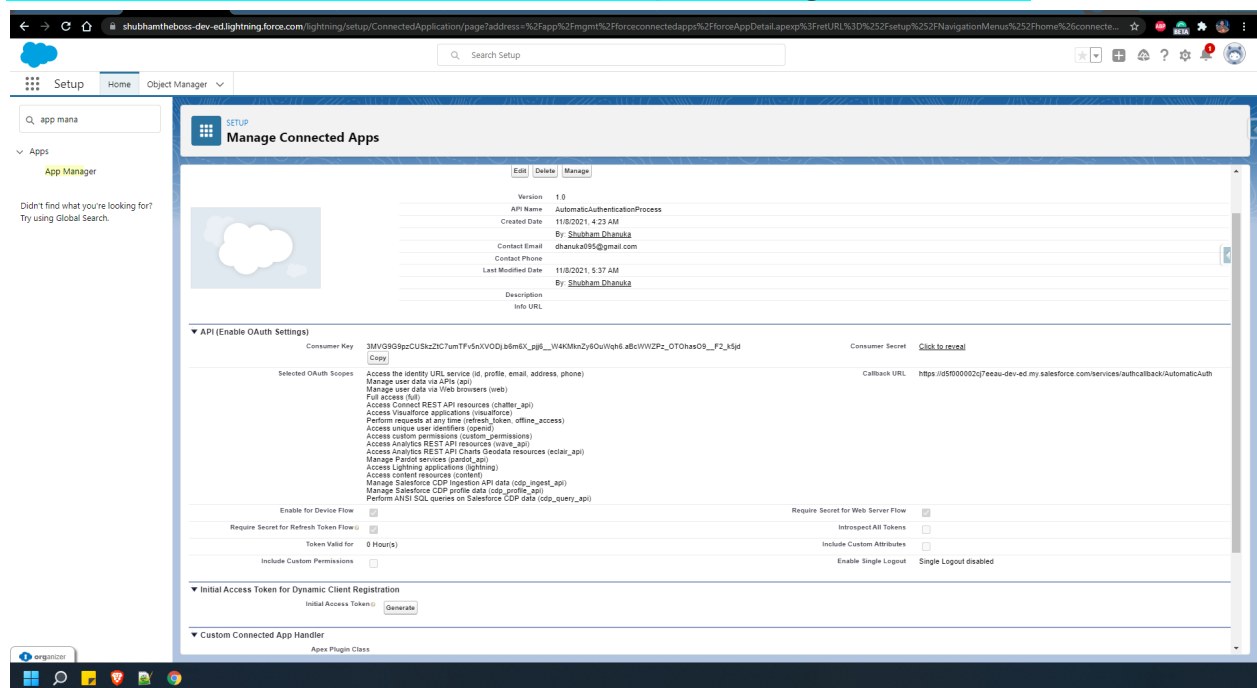
Code:-

@RestResource(urlMapping='/ContactFetcher/*')

```
global class childClass {  
    //information processing  
    @HttpGet  
    global static List<Contact> childClassMethod(){  
        List<Contact> li = [SELECT Id, Name, Phone, Account.Name FROM Contact];  
        System.debug('List of Contact==>' + li);  
        return li;  
    }  
}
```

Configuration:- Connected app

Note: callback url must be source auth provider generated url.



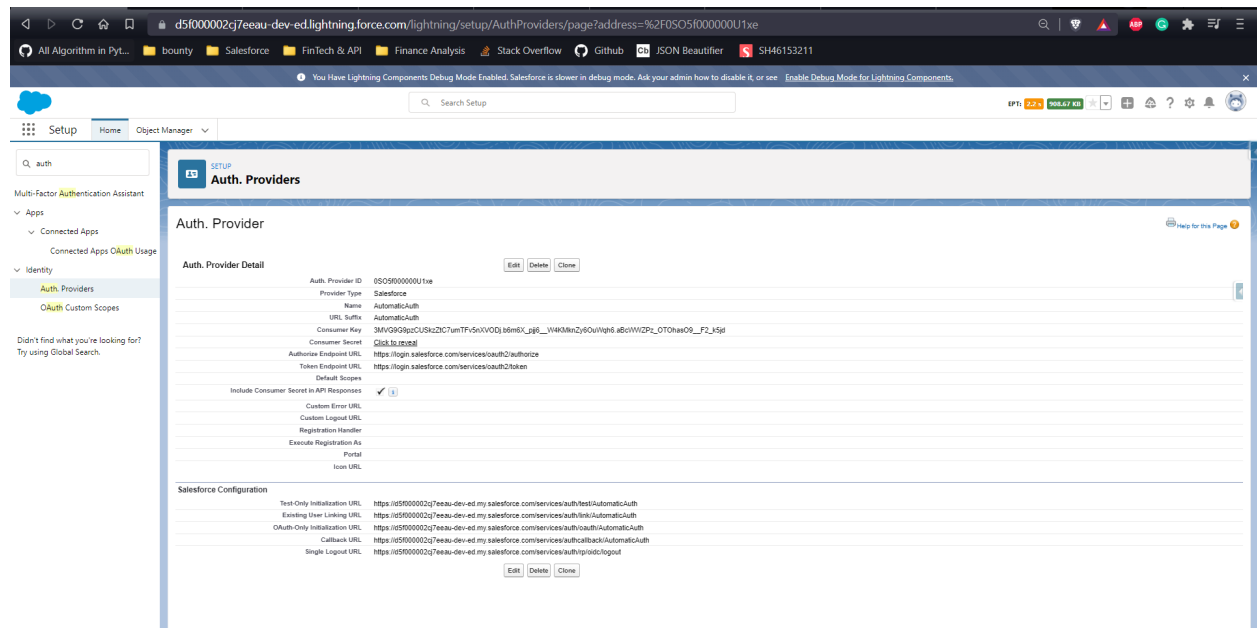
Source Org:-

Code:-

```
global class Webcallout {  
    global static void WebcalloutMethod(){  
        Http http = new Http();  
        HttpRequest request = new HttpRequest();  
        request.setEndpoint('callout:AccessToken/services/apexrest/ContactFetcher/');  
        request.setMethod('GET');  
  
        HttpResponse response = http.send(request);  
        while (response.getStatusCode() == 302) {  
            request.setEndpoint(response.getHeader('Location'));  
            response = new Http().send(request);  
        }  
        System.debug('response' + response);  
        System.debug(response.getBody());  
    }  
}
```

Configuration:-

Auth Provider:-



The screenshot shows the Salesforce Setup page for Auth Providers. The left sidebar contains navigation links for Setup, Home, and Object Manager. The main content area is titled 'Auth. Providers' and displays a table of Auth. Provider details. The table has columns for Auth. Provider ID, Provider Type, Name, URL, Suffix, Consumer Key, Consumer Secret, Authorize Endpoint URL, Token Endpoint URL, Default Scopes, Include Consumer Secret in API Responses, Custom Error URL, Custom Logout URL, Registration Handler, External Registration As, Portal, and Icon URL. The table shows one provider with ID 050900000U1xe, Provider Type Salesforce, Name AutomaticAuth, and URL https://login.salesforce.com/services/oauth2/authorize. The table also includes a 'Salesforce Configuration' section with links for Test Only Initialization URL, Existing User Linking URL, OAuth-Only Initialization URL, Callback URL, and Single Logout URL.

Auth. Provider ID	Provider Type	Name	URL	Suffix	Consumer Key	Consumer Secret	Authorize Endpoint URL	Token Endpoint URL	Default Scopes	Include Consumer Secret in API Responses	Custom Error URL	Custom Logout URL	Registration Handler	External Registration As	Portal	Icon URL
050900000U1xe	Salesforce	AutomaticAuth	https://login.salesforce.com/services/oauth2/authorize		3MVG9G9pccUSkz2C7mTFv5xVODj8d6X_aj8_YH4KMinZu6OuVh6a86WwZPL0TOHaw0S_F2_k5d	Click to reveal	https://login.salesforce.com/services/oauth2/authorize	https://login.salesforce.com/services/oauth2/token		<input checked="" type="checkbox"/>						

Salesforce Configuration

Configuration	URL
Test Only Initialization URL	https://d5f000002q7eeau-dev-ed.my.salesforce.com/services/oauth/test/AutomaticAuth
Existing User Linking URL	https://d5f000002q7eeau-dev-ed.my.salesforce.com/services/oauth/link/AutomaticAuth
OAuth-Only Initialization URL	https://d5f000002q7eeau-dev-ed.my.salesforce.com/services/oauth/auth/AutomaticAuth
Callback URL	https://d5f000002q7eeau-dev-ed.my.salesforce.com/services/oauth/callback/AutomaticAuth
Single Logout URL	https://d5f000002q7eeau-dev-ed.my.salesforce.com/services/oauth/tp/logout

Named Credential:-

Note: <https://domain.my.salesforce.com> is important

<https://github.com/douglascayers-org/sfdx-mass-action-scheduler/wiki/Specify-the-Running-User-via-Named-Credentials>

The screenshot shows the Salesforce Setup interface for Named Credentials. The page title is "Named Credentials". Below the title, it says "Named Credential: AccessToken". A note states: "Specify the callout endpoint's URL and the authentication settings that are required for Salesforce to make callouts to the remote system." There is a link "< Back to Named Credentials".

At the top right of the configuration area are "Edit" and "Delete" buttons.

The configuration details are as follows:

Label	AccessToken
Name	AccessToken
URL	https://shubhamtheboss-dev-ed.my.salesforce.com

Below this is the "Authentication" section, which is expanded. It shows:

Certificate	
Identity Type	Named Principal
Authentication Protocol	OAuth 2.0
Authentication Provider	Automatic Auth
Scope	refresh_token full
Authentication Status	Authenticated as shubhamdhanuka11@gmail.com

Below the authentication section is the "Callout Options" section, which is also expanded. It shows:

Generate Authorization Header	<input checked="" type="checkbox"/>
Allow Merge Fields in HTTP Header	<input checked="" type="checkbox"/>
Allow Merge Fields in HTTP Body	<input checked="" type="checkbox"/>
Outbound Network Connection	<input type="checkbox"/>

Remote site setting:-

<https://login.salesforce.com>

<https://brave-bear-1zteug-dev-ed.my.salesforce.com>

Note:

Remote site setting: site authentic

Auth, Named : Person authentic

Data : data authentic

>>> BASIC OF WEB CALL OUT COMPLETED <<<

Some Useful Terminology:-

- 1) Remote Site setting :
- 2) CSP Site :
- 3) Auth Provider :
- 4) Named Credential :
- 5) Custom Setting :
- 6) Custom Metadata :
- 7) Custom Label :
- 8) Custom URL :
- 9) REST : (XML + JSON Supported) (WebCallout / HTTP Class)
- 10) SOAP : (XML Supported) (Web Service) (Partner WSDL)
- 11) Trusted URL for redirect
- 12) Outbound connection setting

Serialization and Deserialization process:-

Object to string --> JSON, Serialize

String

Blob --> Binary => (0,1) =>(by using ascii)(bit, byte measurement unit)

Encoding

Decoding

Blob

String

String to object--> JSON Deserialize

```
String str1 = 'b';
Blob b = Blob.valueOf(str1);
System.debug('blob is ' + b);

String paramvalue = EncodingUtil.base64Encode(b);
System.debug('Encoded ' + paramvalue);

Blob paramvalue2= EncodingUtil.base64Decode(paramvalue);
System.debug('Decoded blob ' + paramvalue2);

String str2 = paramvalue2.toString();
System.debug('Decoded string ' + str2);
```


Response parsing on parent side:-

To parse the response of json use wrapper class that is the best solution.

JSON TO APEX CONVERSION:-

Class:-

```
public class JSON2Apex {
    public Attributes attributes;
    public Account Account;
    public String Id;
    public String Name;
    public String Phone;
    public String AccountId;

    \
    public class Attributes {
        public String type;
        public String url;
    }
    public class Account {
        public Attributes attributes;
        public String Id;
        public String Name;
    }

    public static JSON2Apex parse(String json) {
        return (JSON2Apex) System.JSON.deserialize(json, JSON2Apex.class);
    }
}
```

Test Class:-

// Generated by JSON2Apex <http://json2apex.herokuapp.com/>

@IsTest

```
public class JSON2Apex_Test {

    static testMethod void testParse() {
        String json = '{'+
            '\nattributes\": {'+
            '\n\"type\": \"Contact\",'+
            '\n\"url\": \"/services/data/v51.0/subjects/Contact/0035g000005QA6RAAW\"'+
            '},'+
            '\n\"Id\": \"0035g000005QA6RAAW\",'+
            '\n\"Name\": \"Rose Gonzalez\",'+
            '\n\"Phone\": \"(512) 757-6000\",'+
            '\n\"AccountId\": \"0015g000000Cvck7AAB\",'+
            '\n\"Account\": {'+
            '\n\"attributes\": {'+
            '\n\"type\": \"Account\",'+
            '\n\"url\": \"/services/data/v51.0/subjects/Account/0015g000000Cvck7AAB\"'+
            '},'+
            '\n\"Id\": \"0015g000000Cvck7AAB\",'+
            '\n\"Name\": \"Edge Communications\"'+
            '}'+'+
        }';
        JSON2Apex obj = JSON2Apex.parse(json);
        System.assert(obj != null);
    }
}
```

Uniqueidentifier: server (path)

Classname (defined in resources url) = programme/ folder_name / **class**

Method: **invoking Action method (get, put, post annotation)**

?Q = query **parameter passing**

DATA sending. Passing:-

<https://www.salesforcecodecrack.com/2018/12/how-to-read-rest-api-get-parameters.html>

Three Ways to pass and send data:-

URL parameter:

URL ID:

Request body:

1) URL parameter:

Query passing in end point ?q=

Parameter passing in end point ?sfgroupId = "

Query:-

```
request.setEndpoint('callout:AccessToken/services/data/v53.0/query/?q=SELECT+Name+FROM+Account');
```

Parameter Passing:-

Target:-

```
request.setEndpoint('callout:AccessToken/services/apexrest/ContactFetcher?Name=Forbes');
request.setMethod('POST');
request.setHeader('Content-type','application/json');
request.setHeader('Content-Length', '0');
```

Source:-

```
@HttpPost
global static List<Contact> childClassMethod2(){
    RestRequest restReq = RestContext.request;
    RestResponse restRes = RestContext.response;
    //reading entire request body
    //String jsonString = RestContext.request.requestBody.toString();

    // Reading parametrs from URL
    String Name= restReq.params.get('Name');

    List<Contact> li = [select Id, Account.Name, LastName, Phone from Contact where LastName =:Name];
    System.debug('List of Contact==>' + li);
    return li;
}
```

2) URL ID:

Parameter Passing:-

Target:-

```
request.setEndpoint('callout:AccessToken/services/apexrest/ContactFetcher/0032v000037Ps3mAAC');
request.setMethod('POST');
request.setHeader('Content-type', 'application/json');
request.setHeader('Content-Length', '0');
```

Source:-

```
@HttpPost
global static List<Contact> childClassMethod2(){
    RestRequest restReq = RestContext.request;
    RestResponse restRes = RestContext.response;
    //reading entire request body
    //String jsonString = RestContext.request.requestBody.toString();

    String contactId = restReq.requestURI.substring(restReq.requestURI.lastIndexOf('/') + 1);
    List<Contact> li = [select Id, Account.Name, LastName, Phone from Contact where Id =:contactId];
    System.debug('List of Contact==>' + li);
    return li;
}
```

3) Request body:

Parameter Passing:- (XML Format)

Target:-

```
request.setEndpoint('callout:AccessToken/services/apexrest/ContactFetcher/');
request.setMethod('POST');
request.setHeader('Content-Type', 'application/xml; charset=utf-8');
request.setHeader('Content-Length', '0');
request.setBody('<?xml version="1.0" encoding="UTF-8" ?><request><Name>'+accName+'</Name></request>');
```

Source:-

```
@HttpPost
global static List<Contact> childClassMethod2(String IName){
    RestRequest restReq = RestContext.request;
    RestResponse restRes = RestContext.response;
    List<Contact> li = [select Id, Account.Name, LastName, Phone from Contact where LastName =:IName];
    System.debug('List of Contact==>' + li);
    return li;
}
```

3) Request body:

Parameter Passing:- (JSON Format)

Target:-

```
String accName = 'Forbes';
```

```
request.setEndpoint('callout:AccessToken/services/apexrest/ContactFetcher/');  
request.setMethod('POST');
```

```
request.setHeader('Content-Type', 'application/JSON');  
request.setHeader('Content-Length', '0');
```

```
MyPayload c = new MyPayload();  
c.LName = accName;  
request.setBody(JSON.serialize(c));
```

```
public class MyPayload {  
    public String LName;  
}
```

Source:-

```
@HttpPost  
global static List<Contact> childClassMethod2(String LName){  
    List<Contact> li = [select Id, Account.Name, LastName, Phone from Contact where LastName =:LName];  
    System.debug('List of Contact==>' + li);  
    return li;  
}
```

//Other way to pass the data from target:-

```
request.setBody('{ "LName": "Forbes" }');
```

```
String accName = "Forbes";  
request.setBody('{ "LName": ' + accName + ' }');
```

DATA SECURITY: (SAML, JSON Web token)

Data Transformation:

XML and JSON format the data

String encryption and data sending part:-

Object to string --> JSON, Serialize

String

Blob --> Binary => (0,1) =>(by using ascii)(bit, byte measurement unit)

Encoding

Decoding

Blob

string

String to object--> JSON Deserialize

JSON.stringify() ::: object to string :: server always understand string format

JSON.serialize() ::: object to string

JSON.parse() ::: string to Object

JSON.deserialize() ::: string to object

Example:

Parameter Passing:- (JSON Format) (Encoded format)

Target:-

```
String accName = 'Forbes';
```

```
request.setEndpoint('callout:AccessToken/services/apexrest/ContactFetcher/');
```

```
request.setMethod('POST');
```

```
request.setHeader('Content-Type', 'application/JSON');
```

```
request.setHeader('Content-Length', '0');
```

```
MyPayload c = new MyPayload();
```

```
c.LName = accName;
```

```
String paramvalue = EncodingUtil.base64encode(blob.valueOf(JSON.serialize(c)));
```

```
request.setBody(paramvalue);
```

```
HttpResponse response = http.send(request);
```

```
public class MyPayload {  
    public String LName;  
}
```

Source:-

```
@HttpPost
global static List<Contact> childClassMethod2(){
    RestRequest restReq = RestContext.request;
    RestResponse restRes = RestContext.response;

    String jsonString = RestContext.request.requestBody.toString();
    System.debug('jsonString ==>' + jsonString);

    blob paramvalue2= EncodingUtil.base64Decode(jsonString);
    String LName2 = paramvalue2.toString();

    MyPayload mp = (MyPayload )System.JSON.deserialize(LName2,MyPayload.class );
    System.debug(mp.LName);

    List<Contact> li = [select Id, Account.Name, LastName, Phone from Contact where LastName = :mp.LName ];
    System.debug('List of Contact==>' + li);
    return li;
}
public class MyPayload {
    public String LName;
}
```

DEBUGGING:-**Tool Useful in development and simulation :**

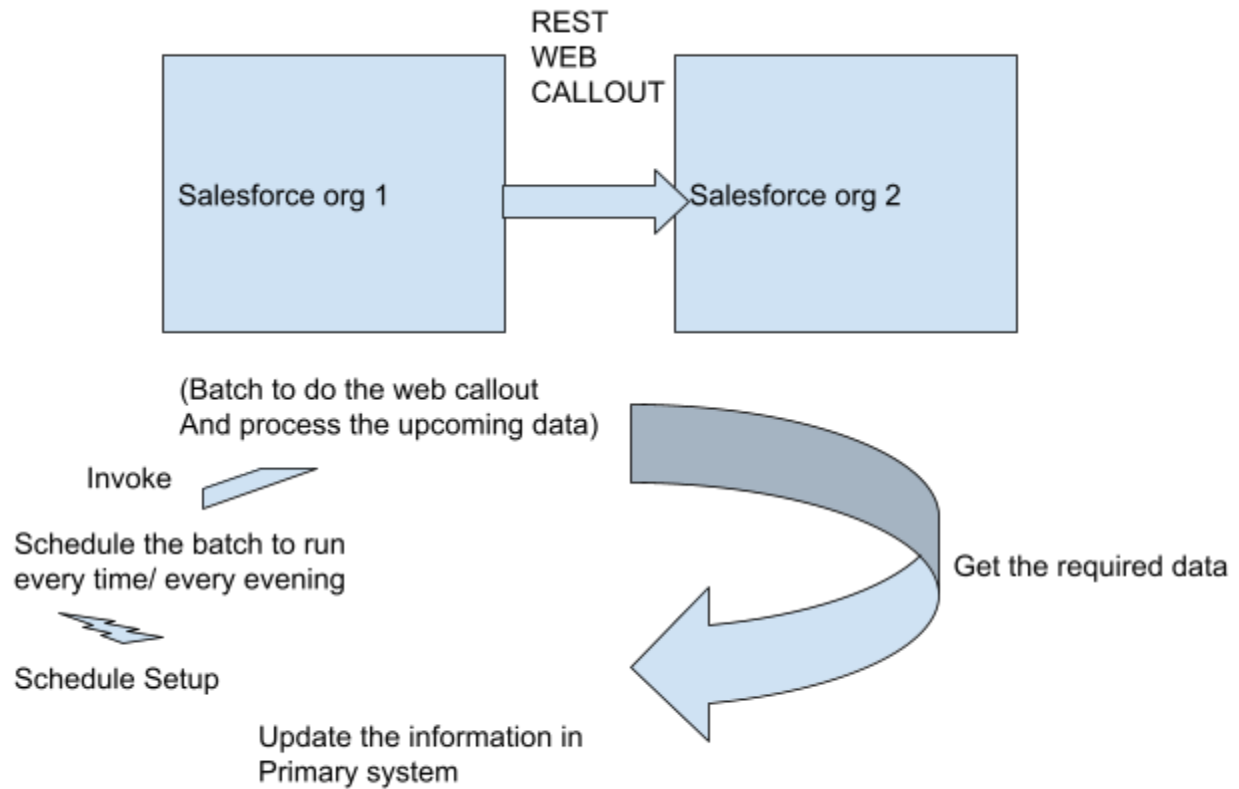
- Workbench
- SOAP UI
- Postman

Troubleshooting:-

- ping/ tracert/ nslookup/ netstat

FULL FLEDGED PROJECT:-

Project of API (including Schedulable, Batch, State, REST Web Callout)



Business UseCase:-

In one org, we have some contact data

In second org, we have some contact data

Need to update in the primary system whose contact matches with secondary org. Just update one custom field call indicator with text "this one already exist in secondary org"

Solution Outline:-

- 1) Web callout configuration**
- 2) Web callout class**
- 3) Response Mock Up Class and Test Web Callout Class**

- 4) Batch class**
- 5) Test Batch class**

- 6) Schedule the class, Test schedule class**

- 7) Setup the Schedule Class**
- 8) Manage/ Delete the schedule setup**

- 9) Monitor the batch job**
- 10) Delete the batch job**

1) Web Callout Configuration:-FOR Automatic OAUTH Configuration:-

Target Org Configuration:- Connected APP

SETUP

Manage Connected Apps

Connected App Name
AutomaticAuthenticationProcess


Help for this Page

Back to List: Custom Apps

Edit

Delete

Manage



Version	1.0
API Name	AutomaticAuthenticationProcess
Created Date	11/9/2021, 4:23 AM
By	By: Shubham Chandra
Contact Email	shamuk004@gmail.com
Contact Phone	
Last Modified Date	11/9/2021, 6:57 AM
By	By: Shubham Chandra
Description	
Info URL	

▼ API (Enable OAuth Settings)

Consumer Key	3MVG8G9p2CUskz2CTumTFvSnXVOD)b6m6X_pj6__W4KMnZy6OuVqph6 aBcWWZPz_OT0hasOJ__F2_K5d	Consumer Secret	2TFTF8Q2B1BC059604A8ACAF8F45AE5302143476006CAF688031B8812FC0DC
Selected OAuth scopes	<div>Copy</div> <div>Access the identity (URL, service (id, profile, email, address, phone)) Manage user data via API (api) Manage user data via Web (web) Full access (full) Access Connect REST API (resources (chatter_api)) Access Visualforce applications (visualforce) Perform requests at any time (refresh, token, offline_access) Access unique user identifiers (openid) Access custom permissions (custom_permissions) Access Analytics REST API (resources (analytics_api)) Access Analytics REST API Churn (Securities (securities_api)) Manage Partner services (partner_api) Access Lightning Applications (lightning) Access content resources (content) Manage Salesforce CDP ingestion API data (cdp_ingest_api) Manage Salesforce CDP profile data (cdp_profile_api) Perform ANSI SQL queries on Salesforce CDP data (cdp_query_api)</div>		
Enable for Device Flow	<input checked="" type="checkbox"/>	Require Secret for Web Server Flow	<input type="checkbox"/>
Require Secret for Refresh Token Flow	<input type="checkbox"/>	Introspect ID Tokens	<input type="checkbox"/>
Token valid for	0 Hour(s)	Include Custom Attributes	<input type="checkbox"/>
Include Custom Permissions	<input type="checkbox"/>	Enable Single Logout	Single Logout disabled

▼ Initial Access Token for Dynamic Client Registration

Initial Access Token:

Generate

▼ Custom Connected App Handler

Apex Plugin Class	
Run As	

Source Org Configuration:- Auth Provider:-

SETUP

Auth. Providers

Auth. Provider

Help for this Page

Auth. Provider Detail

Edit

Delete

Clone

Auth. Provider ID	0505f000000U1xe
Provider Type	Salesforce
Name	AutomaticAuth
URL Suffix	AutomaticAuth
Consumer Key	3MVG8G9p2CUskz2CTumTFvSnXVOD)b6m6X_pj6__W4KMnZy6OuVqph6 aBcWWZPz_OT0hasO9__F2_K5d
Consumer Secret	Click to reveal
Authorize Endpoint URL	https://login.salesforce.com/services/oauth2/authorize
Token Endpoint URL	https://login.salesforce.com/services/oauth2/token
Default Scopes	
Include Consumer Secret in API Responses	<input checked="" type="checkbox"/>
Custom Error URL	
Custom Logout URL	
Registration Handler	
Execute Registration As	
Portal	
Icon URL	

Salesforce Configuration

Test-Only Initialization URL	https://d5f000002q7eeau-dev-ed.my.salesforce.com/services/auth/test/AutomaticAuth
Existing User Linking URL	https://d5f000002q7eeau-dev-ed.my.salesforce.com/services/auth/link/AutomaticAuth
OAuth-Only Initialization URL	https://d5f000002q7eeau-dev-ed.my.salesforce.com/services/auth/oauth/AutomaticAuth
Callback URL	https://d5f000002q7eeau-dev-ed.my.salesforce.com/services/auth/callback/AutomaticAuth
Single Logout URL	https://d5f000002q7eeau-dev-ed.my.salesforce.com/services/auth/oidc/logout

Edit

Delete

Clone

Named Credential:-

SETUP

Named Credentials

Named Credential: AccessToken

Specify the callout endpoint's URL and the authentication settings that are required for Salesforce to make callouts to the remote system.

[Back to Named Credentials](#)

EditDelete

Label

AccessToken

Name

AccessToken

URL

https://shubhamtheboss-dev-ed.my.salesforce.com

Authentication

Certificate

Identity Type

Named Principal

Authentication Protocol

OAuth 2.0

Authentication Provider

AutomaticAuth

Scope

refresh_token full

Authentication Status

Authenticated as shubhamdhanuka11@gmail.com

Callout Options

Generate Authorization Header

✓

Allow Merge Fields in HTTP Header

✓

Allow Merge Fields in HTTP Body

✓

Outbound Network Connection

Remote site settings:-

SETUP

Remote Site Settings

All Remote Sites

Below is the list of Web addresses that your organization can invoke from salesforce.com. To add another Web address, click New Remote Site.

View: All Remote Sites Create New View

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z Other All

Action	Remote Site Name ↑	Namespace Prefix	Remote Site URL	Active	Created By	Created Date	Last Modified By	Last Modified Date
Edit Del	AcenDevNet	--	http://www.acendevnet.com	✓	Dhanuka_Shubham	9/14/2021, 7:37 PM	Dhanuka_Shubham	9/14/2021, 7:37 PM
Edit Del	EnableVICPreview	--	https://d5e00000k9seas-dev-ed-vlocity-ins-na162.visual.force.com	✓	Dhanuka_Shubham	9/14/2021, 7:37 PM	Dhanuka_Shubham	9/14/2021, 7:37 PM
Edit Del	EnableVICPreviewLightning	--	https://d5e00000k9seas-dev-ed-lightning.force.com	✓	Dhanuka_Shubham	9/14/2021, 7:37 PM	Dhanuka_Shubham	9/14/2021, 7:37 PM
Edit Del	SFOC	vlocity_ins	https://example.com	✓	Dhanuka_Shubham	9/14/2021, 7:37 PM	Dhanuka_Shubham	9/14/2021, 7:37 PM
Edit Del	VlocityDocs	vlocity_ins	https://docs-vlocity-help-center.s3.amazonaws.com	✓	Dhanuka_Shubham	9/14/2021, 7:37 PM	Dhanuka_Shubham	9/14/2021, 7:37 PM
Edit Del	VlocityDocsHqas	vlocity_ins	https://docs-vlocity-help-center.s3.amazonaws.com	✓	Dhanuka_Shubham	9/14/2021, 7:37 PM	Dhanuka_Shubham	9/14/2021, 7:37 PM
Edit Del	VlocityLibrary	vlocity_ins	https://vlocity.com	✓	Dhanuka_Shubham	9/14/2021, 7:37 PM	Dhanuka_Shubham	9/14/2021, 7:37 PM
Edit Del	VlocityAuth	--	https://shubhamtheboss-dev-ed.my.salesforce.com	✓	Dhanuka_Shubham	11/8/2021, 5:03 AM	Dhanuka_Shubham	11/8/2021, 5:03 AM
Edit Del	VlocityAuth2	--	https://open.salesforce.com	✓	Dhanuka_Shubham	11/8/2021, 5:06 AM	Dhanuka_Shubham	11/8/2021, 5:06 AM

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z Other All

2) Web callout class:-

1) Target Org Class:-

Rest Class:

```
@RestResource(urlMapping='/ContactFetcher/*')
```

```
global class childClass {
//information processing
    @HttpGet
    global static List<Contact> childClassMethod(){
        List<Contact> li = [SELECT Id, Name, Phone, Account.Name FROM Contact];
        System.debug('List of Contact==>' + li);
        return li;
    }

//Data Creating
    /* @HttpPost
    global static List<Contact> childClassMethod2(){
        List<Contact> li = [SELECT Id, Name, Phone, Account.Name FROM Contact];
        System.debug('List of Contact==>' + li);
        return li;
    }*/
}
```

2) Source Org Class:-

WebCallout Class:

```
global class Webcallout {
    global static List<String> WebcalloutMethod(){
        List<String> li = new List<String>();

        Http http = new Http();
        HttpRequest request = new HttpRequest();

        request.setEndpoint('callout:AccessToken/services/apexrest/ContactFetcher/');
        request.setMethod('GET');

        HttpResponse response = http.send(request);
        while (response.getStatusCode() == 302) {
            request.setEndpoint(response.getHeader('Location'));
            response = new Http().send(request);
        }

        System.debug('response' + response);
        System.debug('respose is ==> '+response.getBody());
    }
}
```

```

List<JSON2Apex> resultMap = (List<JSON2Apex> ) System.JSON.deserialize(response.getBody(),
List<JSON2Apex>.class);
    System.debug('list iteration ==>'+resultMap[0]);

    for (JSON2Apex obj : resultMap){
        //JSON2Apex obj1 = (JSON2Apex) System.JSON.deserializeUnTyped(obj, JSON2Apex.class);
        //System.debug('list iteration ==>'+obj.Name);
        li.add(obj.Name);
    }
    System.debug('size of coming records:-' + li.size());
    System.debug('item in coming list:-' + li);

    return li;
}
}

```

JSON to Apex Class :-

```

public class JSON2Apex {

    public class Account {
        public Attributes attributes;
    }

    public Attributes attributes;
    public String Id;
    public String Name;
    public String Phone;
    public String AccountId;
    public Account Account;

    public class Attributes {
        public String type;
        public String url;
    }

    public static JSON2Apex parse(String json) {
        return (JSON2Apex) System.JSON.deserialize(json, JSON2Apex.class);
    }
}

```

JSON to Apex Test Class:-

@IsTest

```
public class JSON2Apex_Test {
```

```
    static testMethod void testParse() {
```

```
        String json = '{'+
```

```
,
```

```
\"attributes\":{\"type\":\"Contact\",\"url\":\"/services/data/v53.0/subjects/Contact/0032v000037Ps3mAAC\"},'+
```

```
    \"Id\":\"0032v000037Ps3mAAC\",'+
```

```
    \"Name\":\"Sean Forbes\",'+
```

```
    \"Phone\":\"(512) 757-6000\",'+
```

```
    \"AccountId\":\"0012v00002c0vPdAAI\",'+
```

```
,
```

```
\"Account\":{\"attributes\":{\"type\":\"Account\",\"url\":\"/services/data/v53.0/subjects/Account/0012v00\"}}}}';
```

```
        JSON2Apex obj = JSON2Apex.parse(json);
```

```
        System.assert(obj != null);
```

```
    }
```

```
}
```

3) Response Mock Up Class and Test Web Callout Class

Mockup

@isTest

```
global class WebCalloutMock implements HttpCalloutMock {
```

```
    String json = '{'+
```

```
,
```

```
\"attributes\":{\"type\":\"Contact\",\"url\":\"/services/data/v53.0/subjects/Contact/0032v000037Ps3mAAC\"},'+
```

```
    \"Id\":\"0032v000037Ps3mAAC\",'+
```

```
    \"Name\":\"test\",'+
```

```
    \"Phone\":\"9828185204\",'+
```

```
    \"AccountId\":\"0012v00002c0vPdAAI\",'+
```

```
,
```

```
\"Account\":{\"attributes\":{\"type\":\"Account\",\"url\":\"/services/data/v53.0/subjects/Account/0012v00\"}}}}';
```

```
global HTTPResponse respond(HTTPRequest request) {
```

```
    HTTPResponse response = new HTTPResponse();
```

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

```
    response.setBody(json);
```

```
    response.setStatusCode(200);
```

```
    return response;
```

```
}
```

```
}
```

Test Web Callout Class:-

```
@istest
public class WebcalloutTest {
    @istest
    static void main(){
        Test.setMock(HttpCalloutMock.class, new WebCalloutMock());
        List<String> strResp = Webcallout.WebcalloutMethod();
        System.debug('List of mock response' + strResp);
    }
}
```

4) Batch class

```
global class FristBathc implements Database.Batchable<Subject>, Database.AllowsCallouts, Database.Stateful{
    // variable to add the id
    global List<String> ExternalList = new List<String>();

    //Constructor for webcallout
    global FristBathc(){
        ExternalList = Webcallout.WebcalloutMethod();
        System.debug('here is the execution');
        System.debug('here is returned data in batch' + ExternalList);
    }
    // start Method
    global Database.QueryLocator start (Database.BatchableContext BC){
        //get list of customer record
        return Database.getQueryLocator('select id, Name, Indicator__c from Contact where Name in :ExternalList ');
    }
    //execute method
    global void execute(Database.BatchableContext BC, List<subject> scope) {
        //if customer status is active, update customer status and customer discription
        //either you can define it as customer list or you can cast it in iteration: to case Customer c =
        (customer__c)scope;
        List<Contact> li = scope;
        System.debug('scope list' + li);
        List<Contact> lic = new List<Contact>();

        for(Contact ac: li){
            ac.Indicator__c = 'yes this one also existing in secondary system';
            lic.add(ac);
        }
        if(lic.size() != 0){
            Database.update(lic);
        }
    }
    //finish method
    global void finish(Database.BatchableContext BC) {
        //send an email to customer
    }
}
```

5) Test Batch class:-

```
@istest
public class firstbatchtest {
    //Apex provides the built-in WebServiceMock interface and the Test.setMock method.
    @istest
    static void main(){
        Test.setMock(HttpCalloutMock.class, new WebCalloutMock());
        List<String> strResp = Webcallout.WebcalloutMethod();
        System.debug('response in test' + strResp);

        Contact con = new Contact();
        con.LastName = 'test';
        insert con;
        List <Contact> ua = [select Id, Phone, Indicator__c from Contact where Name in :strResp
LIMIT 1];
        System.debug('resonse going' + ua);

        Test.startTest();

        FristBathc BD = new FristBathc();
        BD.start(null);
        BD.execute(null, ua);
        BD.finish(null);
        //Database.executeBatch(BD);
        Test.stopTest();

        //System.assertEquals('7230882125', ua.Phone);

    }
}
```

6) Schedule the class and test schedule class:-

Class:-

```
global class scheduleClass implements schedulable{
    global void execute(SchedulableContext SC){
        FristBathc BD = new FristBathc();
        Database.executeBatch(BD);
    }
}
```

TestClass:-

```
@isTest
public class scheduleClassTest {
    @isTest
    static void main (){
        Test.StartTest();
        scheduleClass sc = new scheduleClass();
        //sc.execute();
        String sch = '0 0 23 * * ?';
        system.schedule('Test', sch, sc);

        Test.stopTest();
    }
}
```


7) Setup the schedule:

The screenshot shows the 'Apex Classes' setup page. On the left, a sidebar contains navigation links: Email, Custom Code, and Data Classification. The main content area is titled 'Apex Classes' and includes a status bar indicating 'Percent of Apex Used: 0.08%'. Below this, there are links for 'Estimate your organization's code coverage' and 'Compress all classes'. A table lists the Apex classes, with columns for Action, Name, Namespace Prefix, API Version, Status, Size Without Comments, Last Modified By, and Has Trace Flags. The table shows three classes: 'AccountInfo', 'AccountCreate', and 'AccountTransferTest', all with a status of 'Active' and a size of 253, 206, and 10,224 respectively.

Action	Name	Namespace Prefix	API Version	Status	Size Without Comments	Last Modified By	Has Trace Flags
Edit Security	AccountInfo	vtocdy_jns	31.0	Active	253	Shubham Chandra	<input type="checkbox"/>
Edit Security	AccountCreate	vtocdy_jns	31.0	Active	206	Shubham Chandra	<input type="checkbox"/>
Edit Security	AccountTransferTest	vtocdy_jns	30.0	Active	10,157	Shubham Chandra	<input type="checkbox"/>
Edit	AccountTransferTest	vtocdy_jns	30.0	Active	10,224	Shubham Chandra	<input type="checkbox"/>

The screenshot shows the 'Schedule Apex' page. It includes a 'Job Name' field with the value 'TestOfScheduleClass' and an 'Apex Class' field with the value 'scheduleClass'. The 'Frequency' is set to 'Weekly'. A dropdown menu for 'Recurs every week on' shows the days of the week, with 'Saturday' selected. The 'Start' date is '11/17/2021', the 'End' date is '12/17/2021', and the 'Preferred Start Time' is '12:00 AM'. The page also includes 'Save' and 'Cancel' buttons.

Job Name: TestOfScheduleClass
Apex Class: scheduleClass

Frequency: ☒ Weekly ☐ Monthly

Recurs every week on:
☐ Sunday
☐ Monday
☐ Tuesday
☐ Wednesday
☐ Thursday
☒ Friday
☐ Saturday

Start: 11/17/2021
End: 12/17/2021
Preferred Start Time: 12:00 AM

8) Manage/ Delete the schedule setup:-

The screenshot shows the 'All Scheduled Jobs' page. It includes a 'View' dropdown menu set to 'All Scheduled Jobs' and a 'Create New View' link. A table lists the scheduled jobs, with columns for Action, Job Name, Submitted By, Submitted, Started, Next Scheduled Run, and Type. The table shows one job: 'TestOfScheduleClass', submitted by 'Dhanuka Shubham' on '11/17/2021, 11:49 PM', with a 'Next Scheduled Run' of '11/20/2021, 12:00 AM' and a 'Type' of 'Scheduled Apex'.

Action	Job Name	Submitted By	Submitted	Started	Next Scheduled Run	Type
Manage Del	TestOfScheduleClass	Dhanuka Shubham	11/17/2021, 11:49 PM		11/20/2021, 12:00 AM	Scheduled Apex

9) Monitor the batch job:-

Click here to go to the new batch jobs page

Apex Jobs

Monitor the status of all Apex jobs, and optionally, abort jobs that are in progress.

View: [All] Create New View

Action	Submitted Date	Job Type	Status	Status Detail	Total Batches	Batches Processed	Failures	Submitted By	Completion Date	Apex Class	Apex Method	Apex Job ID
	11/17/2021, 11:31 PM	Batch Apex	Completed		1	1	0	Dhanuka_Shukham	11/17/2021, 11:31 PM	FinalBatch		707590000PUNVCz
	11/15/2021, 9:27 AM	Batch Apex	Completed		1	1	0	Dhanuka_Shukham	11/15/2021, 9:27 AM	FinalBatch		707590000Hoc2w
	11/15/2021, 7:24 AM	Batch Apex	Completed		1	1	0	Dhanuka_Shukham	11/15/2021, 7:24 AM	FinalBatch		707590000H0p1
	11/15/2021, 7:21 AM	Batch Apex	Completed		1	1	0	Dhanuka_Shukham	11/15/2021, 7:21 AM	FinalBatch		707590000H0p1
	11/12/2021, 12:00 AM	Batch Apex	Completed		1	1	0	Dhanuka_Shukham	11/12/2021, 12:00 AM	FinalBatch		707590000H0p0
	11/5/2021, 9:20 PM	Scheduled Apex	Aborted		0	0	0	Dhanuka_Shukham		ScheduleClass		707590000Kp9Ue

10) Delete the batch job:-

Query:-

SELECT

ApexClassId,CompletedDate,CreatedById,CreatedDate,ExtendedStatus,Id,JobItemsProcessed,Job Type,LastProcessed,LastProcessedOffset,MethodName,NumberOfErrors,ParentJobId,Status,Total JobItems FROM AsyncApexJob where status = 'queued'

System.abortJob('JobID');

Few practical exam of parsing :-

// Online Javascript Editor for free

// Write, Edit and Run your Javascript code using JS Online Compiler

```
console.log("Welcome to Programiz!");
```

```
let columns = [  
  { label: 'Contact Name', fieldName: 'ContactName' },  
  { label: 'Contact Phone', fieldName: 'ContactPhone', type: 'phone' },  
  { label: 'Contact Email', fieldName: 'ContactEmail' },  
  { label: 'Contact Language', fieldName: 'ContactLanguage' },  
];  
console.log(JSON.stringify(columns));  
column = JSON.stringify(columns);  
console.log(JSON.parse(column));  
console.log('<----->');
```

```
headers=[];  
parms_headers = 'Name, Email';  
parms_headers.split(',').forEach((item,i)=>{  
  var x = {};  
  x.label =i;  
  x.fieldName = item.trim();  
  headers.push(x);  
});  
console.log('Headers:'+JSON.stringify(headers));  
console.log('<----->');
```

```

headers=[];
parms_label = 'Contact Name, Contact Phone';
parms_fieldName = 'ContactName, ContactPhone';
array_label = parms_label.split(",");
array_fieldName = parms_fieldName.split(",");
console.log(array_label);
console.log(array_fieldName);

```

```

parms_label.split(',').forEach((item,i)=>{
    var x = {};
    x.label =array_label [i].trim();
    x.fieldName = array_fieldName[i].trim();
    headers.push(x);
});
console.log('Headers:'+ JSON.stringify(headers));
head= JSON.stringify(headers);
console.log(JSON.parse(head));

```

```

node /tmp/PfLrputo2o.js
Welcome to Programiz!
[{"label":"Contact Name","fieldName":"ContactName"}, {"label":"Contact Phone","fieldName":"ContactPhone",
  "type":"phone"}, {"label":"Contact Email","fieldName":"ContactEmail"}, {"label":"Contact Language",
  "fieldName":"ContactLanguage"}]
[ { label: 'Contact Name', fieldName: 'ContactName' },
  { label: 'Contact Phone',
    fieldName: 'ContactPhone',
    type: 'phone' },
  { label: 'Contact Email', fieldName: 'ContactEmail' },
  { label: 'Contact Language', fieldName: 'ContactLanguage' } ]
<----->>
Headers:[{"label":0,"fieldName":"Name"}, {"label":1,"fieldName":"Email"}]
<----->>
[ 'Contact Name', ' Contact Phone' ]
[ 'ContactName', ' ContactPhone' ]
Headers:[{"label":"Contact Name","fieldName":"ContactName"}, {"label":"Contact Phone","fieldName":
  "ContactPhone"}]
[ { label: 'Contact Name', fieldName: 'ContactName' },
  { label: 'Contact Phone', fieldName: 'ContactPhone' } ]

```

Some practical exam to learn and work:-

How to do json parsing using json parser class and wrapper class:-

Main class:-

```
public class JSONParsing {
    public static void responseDataParsing(){
        String str =
' [{"invoiceList": [{"totalPrice":5.5,"statementDate":"2011-10-04T16:58:54.858Z", "lineItems": [{"UnitPrice":1.0,"Quantity":5.0,"ProductName":"Pencil"}, {"UnitPrice":0.5,"Quantity":1.0,"ProductName":"Eraser"}], "invoiceNumber":1}, {"totalPrice":11.5,"statementDate":"2011-10-04T16:58:54.858Z", "lineItems": [{"UnitPrice":6.0,"Quantity":1.0,"ProductName":"Notebook"}, {"UnitPrice":2.5,"Quantity":1.0,"ProductName":"Ruler"}, {"UnitPrice":1.5,"Quantity":2.0,"ProductName":"Pen"}], "invoiceNumber":2}]]';
        JSONParser parser = JSON.createParser(str);
        System.debug('response' + parser);
        Double grandTotal = 0.0;
        while (parser.nextToken() != null) {
            if ((parser.getCurrentToken() == JSONTOKEN.FIELD_NAME) &&
                (parser.getText() == 'totalPrice')) {
                // Get the value.
                parser.nextToken();
                // Compute the grand total price for all invoices.
                grandTotal += parser.getDoubleValue();
            }
        }
        system.debug('Grand total=' + grandTotal);
        /*
JSON2Apex obj = JSON2Apex.parse(str);
System.debug('obj' + obj);
System.debug('obj' + obj.InvoiceList);
//System.assert(obj != null);
for (Object o : obj.InvoiceList){
System.debug('indivivual data' + o);
}*/
        Double sum = 0;
        List<JSON2Apex> resultMap =
(List<JSON2Apex>)System.JSON.deserialize(str, List<JSON2Apex>.class);
        for (JSON2Apex obj : resultMap){
            for(JSON2Apex.InvoiceList il : obj.InvoiceList ){
                //System.debug(''+ il.totalprice);
                sum = sum + il.totalprice;
            }
            //li.add(obj.Name);
        }
        System.debug('sum' + sum);
    }
}
```

Reference JSON2Apex class:-

```
public class JSON2Apex {
    public class LinelItems {
        public Double UnitPrice;
        public Double Quantity;
        public String ProductName;
    }

    public List<InvoiceList> invoiceList;

    public class InvoiceList {
        public Double totalPrice;
        public String statementDate;
        public List<LinelItems> linelItems;
        public Integer invoiceNumber;
    }

    public static JSON2Apex parse(String json) {
        return (JSON2Apex) System.JSON.deserialize(json, JSON2Apex.class);
    }
}
```

Encoded and decoded the value :-

```
public class LearningOfSpecificConcept {
    public static void main () {

        String str = String.valueOf(System.TODAY());
        System.debug('Date is ' + str);
        System.debug('Today' + System.TODAY());

        String str1 = 'b';
        Blob b = Blob.valueOf(str1);
        System.debug('blob is ' + b);

        String paramvalue = EncodingUtil.base64Encode(b);
        System.debug('Encoded ' + paramvalue);

        blob paramvalue2= EncodingUtil.base64Decode(paramvalue);
        System.debug('Decoded blob ' + paramvalue2);

        String str2 = paramvalue2.toString();
        System.debug('Decoded string ' + str2);
    }
}
```

**How to write JSON class and get the data from user and proceed
And pass the data back to request 🍌**

```
@RestResource(urlMapping='/ContactFetcher/*')

global class childClass {
//information processing
    @HttpGet
    global static List<Contact> childClassMethod(){
        List<Contact> li = [select Id, LastName, Phone from Contact];
        System.debug('List of Contact==>' + li);
        return li;
    }
    @HttpPost
    global static List<Contact> childClassMethod2(){
        RestRequest restReq = RestContext.request;
        RestResponse restRes = RestContext.response;
        //reading entire request body
        String jsonString = RestContext.request.requestBody.toString();
        System.debug('jsonString ==>' + jsonString);
        // Reading parametrs from URL
        //String Name= restReq.params.get('Name');
        blob paramvalue2= Encodingutil.base64Decode(jsonString);
        String LName2 = paramvalue2.toString();

        MyPayload mp = (MyPayload
)System.JSON.deserialize(LName2,MyPayload.class );
        System.debug(mp.LName);

        List<Contact> li = [select Id, Account.Name, LastName, Phone from
Contact where LastName = :mp.LName ];
        System.debug('List of Contact==>' + li);
        return li;
    }
    public class MyPayload {
        public String LName;
    }
}
```