PaaS4SaaS & X-PaaS Hands on Use Cases

2.1 FSDP-OIC-Oracle Marketing Cloud Integration

Disclaimer:

The following is intended to outline our general product direction. It is intended for information purposes only, and may not be incorporated into any contract. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. The development, release, and timing of any features or functionality described for Oracle's products remains at the sole discretion of Oracle.

Table of Contents

Introduction

In this lab, we will build integration to Oracle Marketing Cloud (Eloqua). This integration would be used to GET/CREATE/UPDATE contact details from Eloqua.
This integration will be called from an OIC Process to create contact and/or update address of contact in Oracle Marketing Cloud (Eloqua).

Storyline

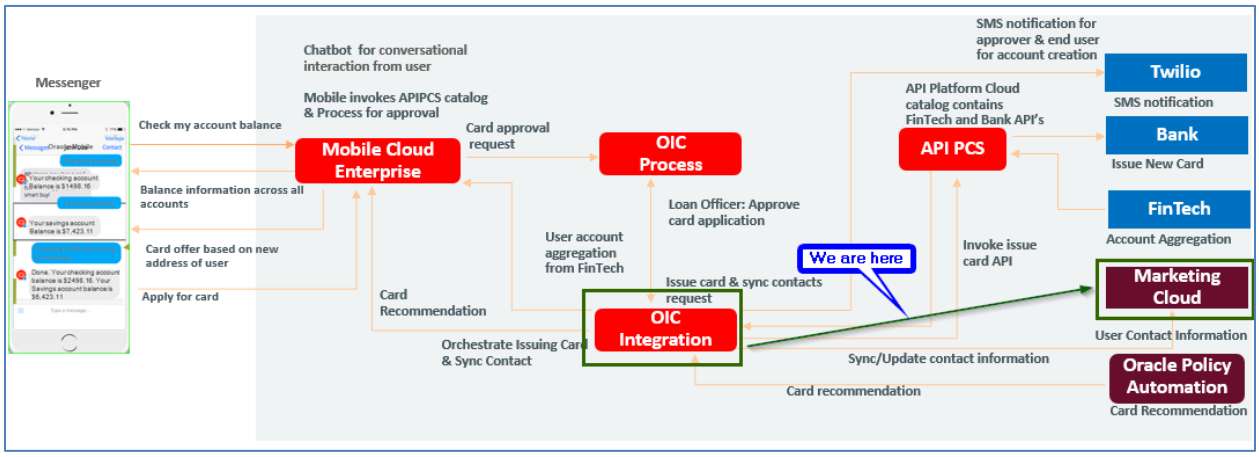
In this session we will be using Oracle PaaS components namely **Oracle Integration Cloud (OIC), Oracle Mobile Cloud Enterprise (OMCe)** and **Oracle API Platform Cloud** along with SaaS components **Oracle Marketing Cloud & Oracle Policy Automation** to build an end to end solution which will also access Bank and FinTech API's.

In this use case we have 3 personas.
1) Lisa who is the end user prefers digital interactions, she wants her financial institution (Call it Zip Bank here) to **know her better and make proactive offers**. Zip bank has partnership with **Fintech who offers account aggregation** not only for accounts she has with Zip Bank but across several banks in which Lisa might have account being able to show in once place.
As part of the end user interaction we are building a chat bot, which is a feature of Oracle Mobile Cloud Enterprise (OMCe), for Lisa's interaction with her zip bank for checking her balances across accounts, pay card balance etc.
The balances across multiple accounts are consolidated using Fintech whose API endpoints are defined & secured in API Platform Cloud Service.
As Lisa interacts with bank using chatbot, bank makes proactive targeted card offer for Lisa. This offer is coming from recommendation engine Oracle Policy Automation(OPA). Oracle Integration Cloud is used to retrieve the offer from recommendation engine (Using SOAP adapter in OIC) and provide the offer to Lisa in chat bot as part of her interaction.
If Lisa decides to apply for the card, we gather basic information including Lisa address which might need update and send it for approval.
2) Mike – Mike is a loan officer at the retail banking division of zip bank. He is responsible for loan approvals including new cards. We are building a Process using Oracle Integration Cloud for Mike's approval of new card request. Also, when Mike approves the card applications we are using Oracle Integration Cloud to call banks APIs to request for new cards. The Bank's API endpoints are defined, secured & managed using API Platform Cloud Service. Integration cloud is using REST adapter to talk to bank API's published/defined at API Platform Cloud Service which will send the request to back end bank API's to issue card.

3) Frank – Frank is an IT analyst responsible for monitoring API connectivity usage etc. Frank monitors the API's using API Platform Cloud Service. The monitor capacity lets frank look at API's calls being made including successful & failed calls.

Architecture

The high-level architecture diagram for the integration performed in this lab is the following:



Products used & Versions

You would need the following products for this use case

- Oracle Integration Cloud Service (OIC) 18.1.3
- Oracle Marketing Cloud (Eloqua): 11.1.11

Lab Instructions (PLEASE READ BEFORE PROCEEDING):

- For all the artifacts you will create as part of the lab, please suffix the username assigned to you (you will find that in the spreadsheet provided by the instructor). E.g PTFE01, PTFE02 etc. Further instructions are provided in the respective sections.
  - Please **DO NOT** create/modify as **_USERNAME** or any other name since that may lead to errors. This is to have a unique name to your artifacts and avoid any conflict with other users who may be using the same environment.
- You will find "**Note**" and "**Tech Notes**" across this document. While "**Note**" section is a mandatory read mostly containing the lab instructors, you may read "**Tech Note**s" for getting more info about that step. You may safely ignore "**Tech Notes**" should you repeat this lab or have time constraints.

OIC Integration with Marketing Cloud for Contact Synch

Pre-Requisites:

- Running instances of Oracle Integration Cloud.
- REST endpoint URL for Eloqua Cloud and its credentials. This would be provided by the instructor.
- **FS Digital Platform Information.xlsx** excel sheet with all the references of artifacts/endpoints. This excel sheet would be used to refer all the URLs/Credentials across this document. Please keep the excel open and use the information from the specific sheets accordingly.
- Postman REST client for Chrome browser (or any equivalent).

Environment Configuration

Nothing specific needed.

Note: We have already completed the pre-requisite of installing OIC for Eloqua app in Oracle Market place to link this OIC instance to Eloqua system.

Tech Note: It is mandatory to have the "OIC for Eloqua" app installed and configured in your Oracle Market Place prior to testing an Eloqua connection using Eloqua adapter as "Trigger and Invoke", in OIC.

Below are references for further understanding.

http://docs.oracle.com/cloud/latest/marketingcs_gs/OMCAA/Help/Apps/IntegrationCloudService/IntegrationCloudServiceApp.htm

http://docs.oracle.com/cloud/latest/marketingcs_gs/OMCAA/Help/Apps/IntegrationCloudService/Tasks/InstallingOIC.htm

Create OIC Connections

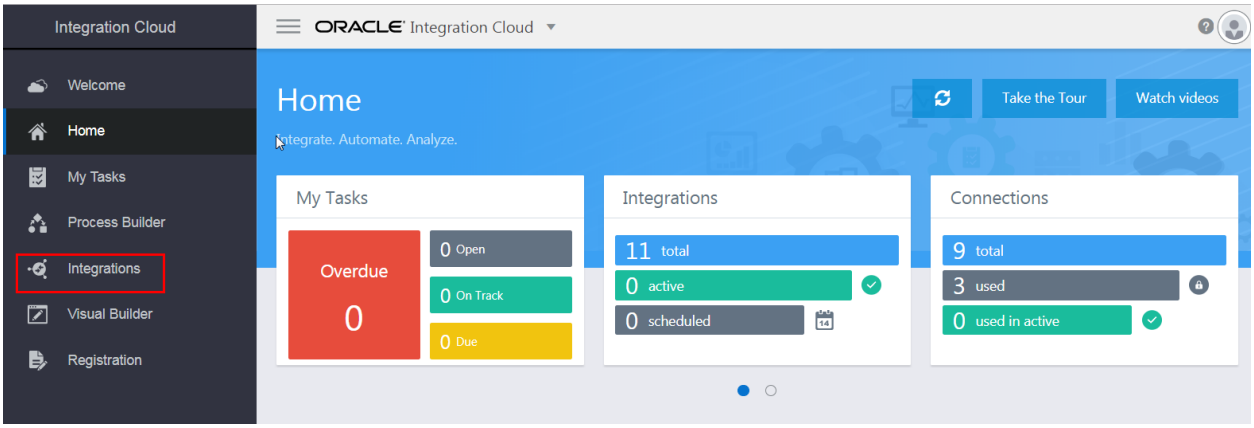
We need to create 2 type of connections to Eloqua system. One using OIC REST Connector and another one using OIC Eloqua connector.

Create REST Adapter based Connection

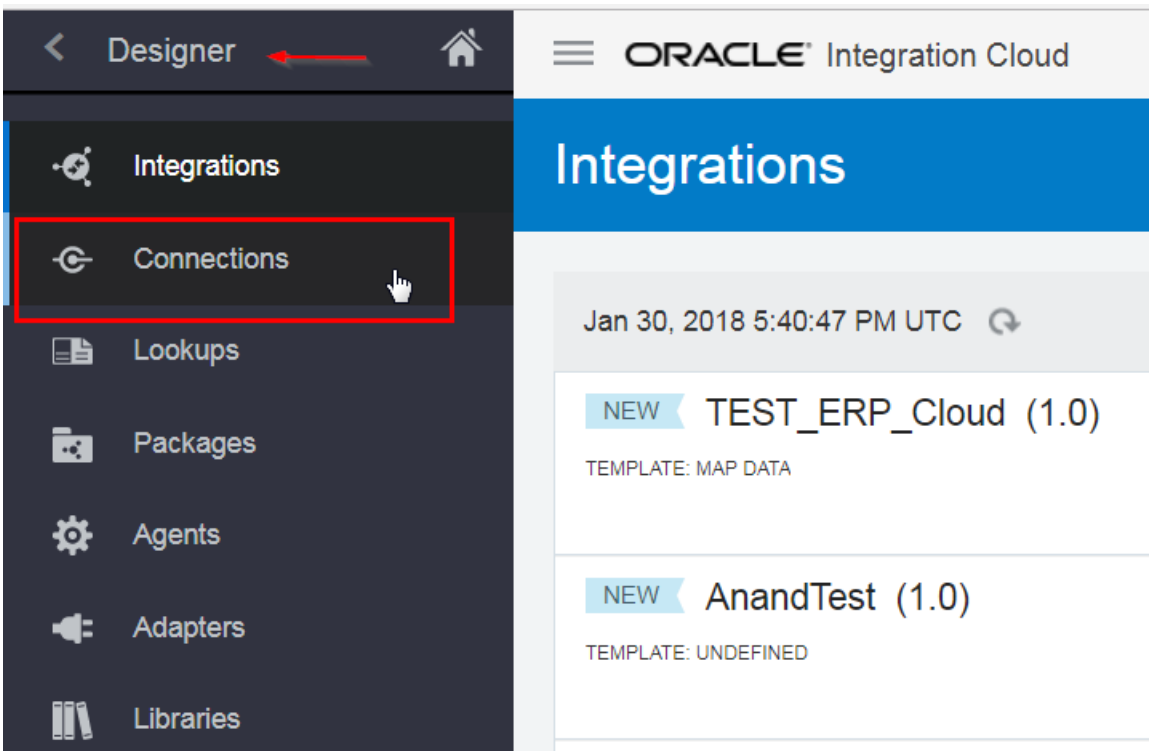
1. Login to your allocated OIC instance using the credentials provided.

Refer to the OIC URL& credentials provided to you. **Example: OIC instance URL** https://<hostname>.oraclecloud.com/ic/home/

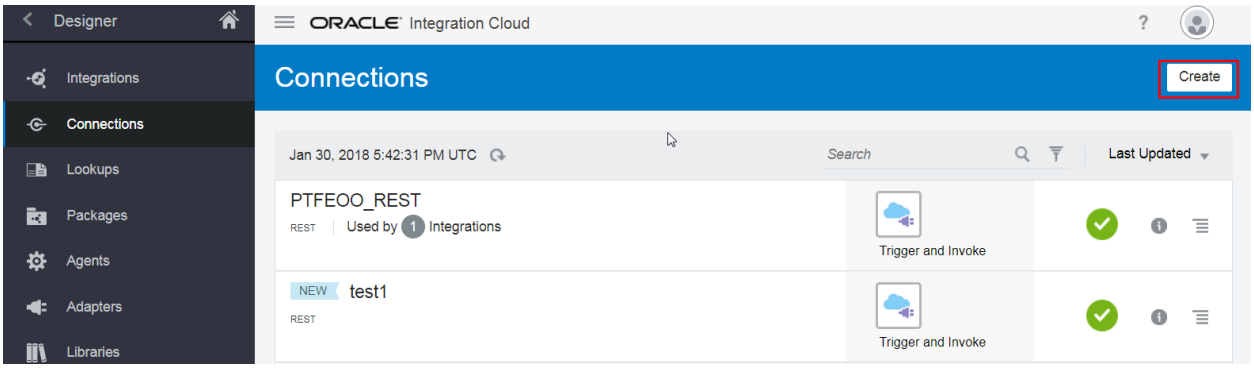
2. Click "**Got it**" if you see a welcome message. Else, proceed to next step.
3. Click on **Integrations** from the Menu on the left



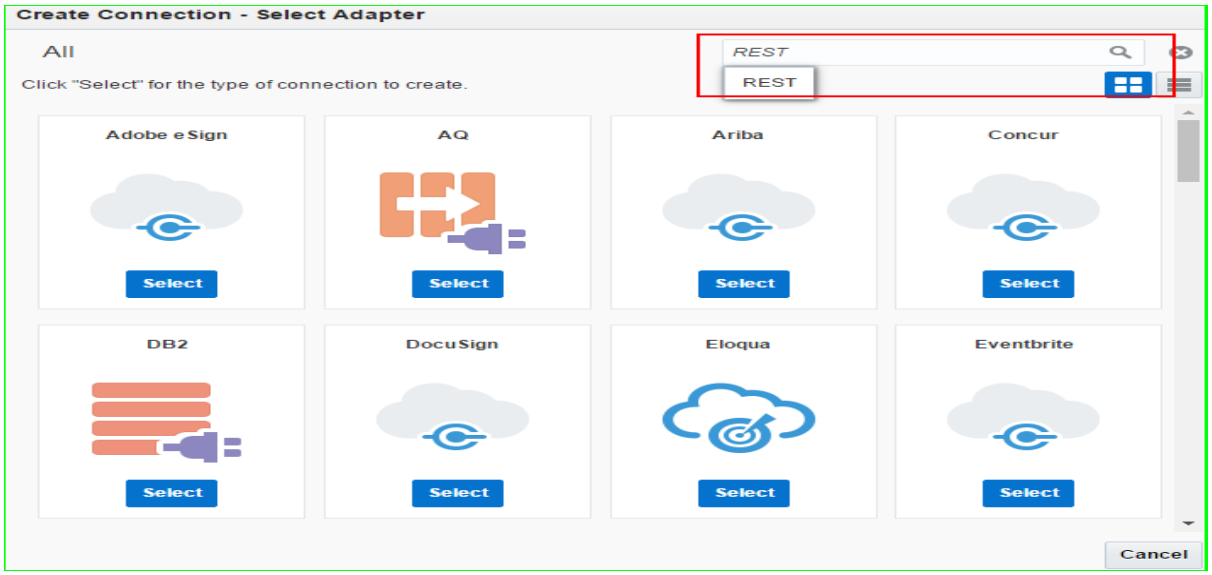
4. In "**Designer**" page, Click on **Connections**



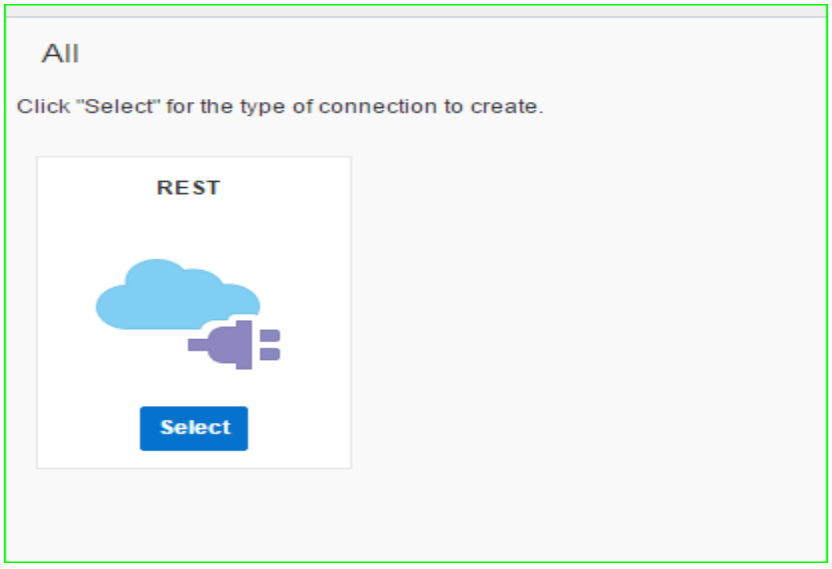
5. Click "**Create**" on top right.



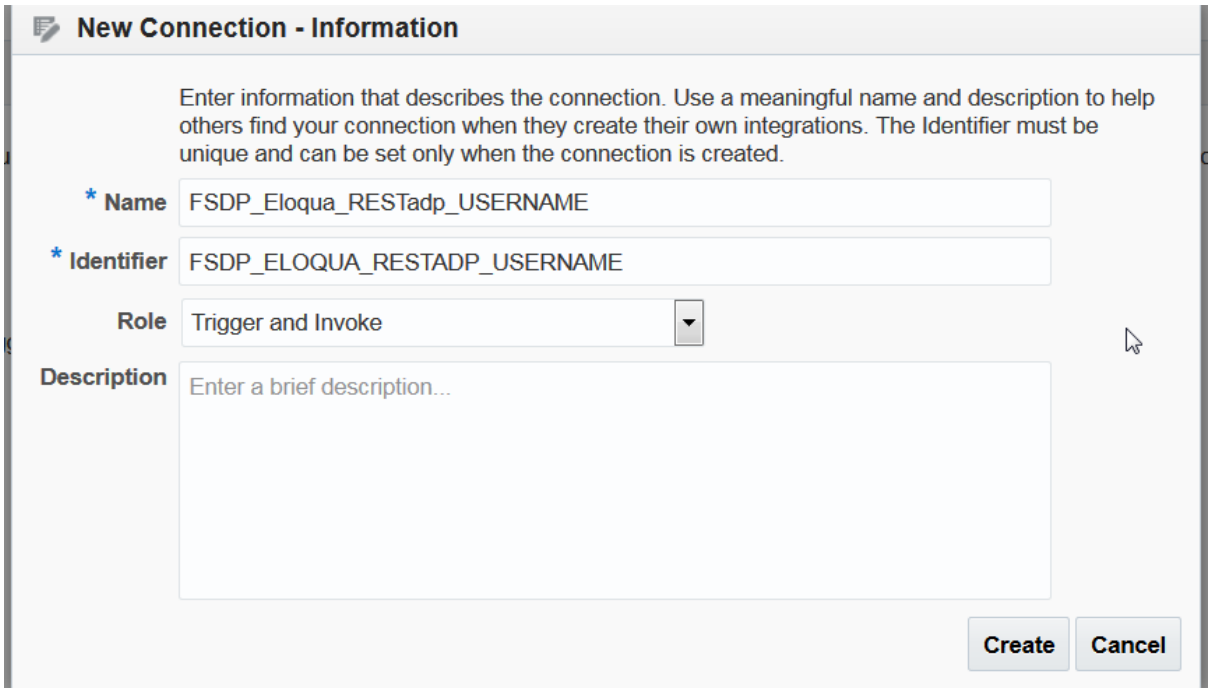
6. Now type **REST** and click on Search to narrow down the selection



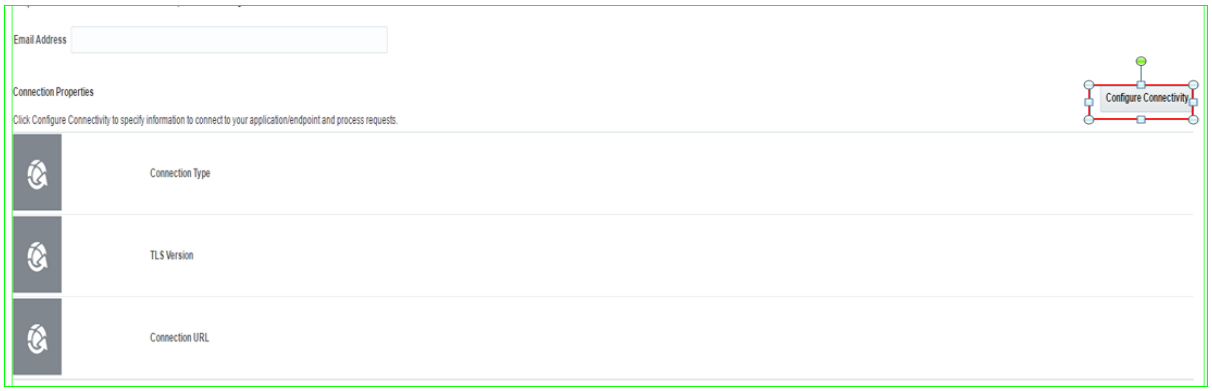
7. Select the **REST** connection



8. Specify the name as **FSDP_Eloqua_RESTadp_USERNAME** (Replace **USERNAME** with your allocated username). Make sure that the connection type is **Trigger and Invoke.**
9. Click on **Create** to create the Connection.

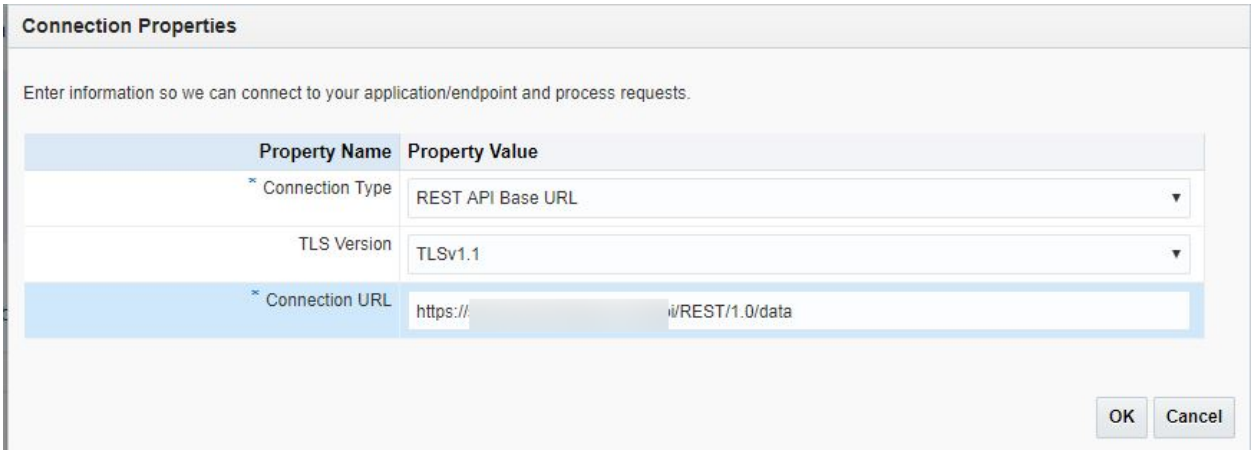


10. Now click on **Configure Connectivity**

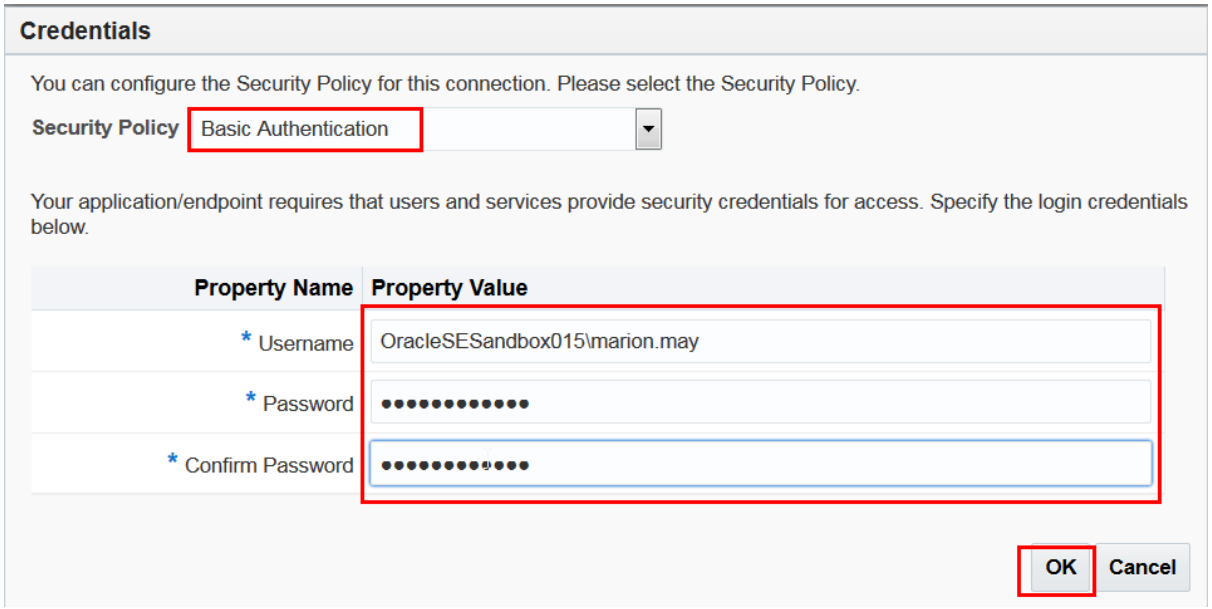


11. Provide the following values and click **Ok** to save.

Connection Type: REST API Base URL
TLS Version : TLSV1.1
Connection URL : <Use the Eloqua REST endpoint provided by instructor/worksheet>

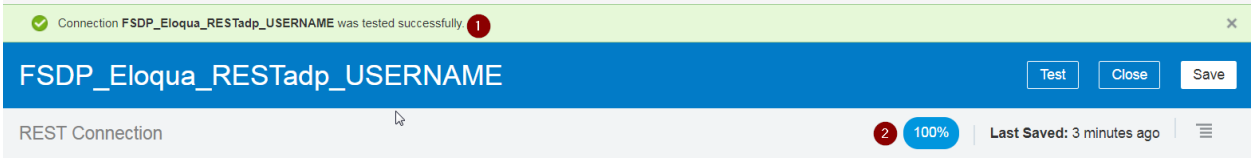


12. Now click on **Configure Security**
13. Choose the security policy as **Basic Authentication**. Provide the credentials. Click **Ok**
    Note: <Use the Eloqua REST endpoint credentials provided by instructor/worksheet> and verify right format <Eloqua company name\first.last> in field "Username"



14. Click Test at the top to test the connection

15. After Test is successful, you should now see 100%.



16. Click on **Save** to save the connection. You should see a message now at the top

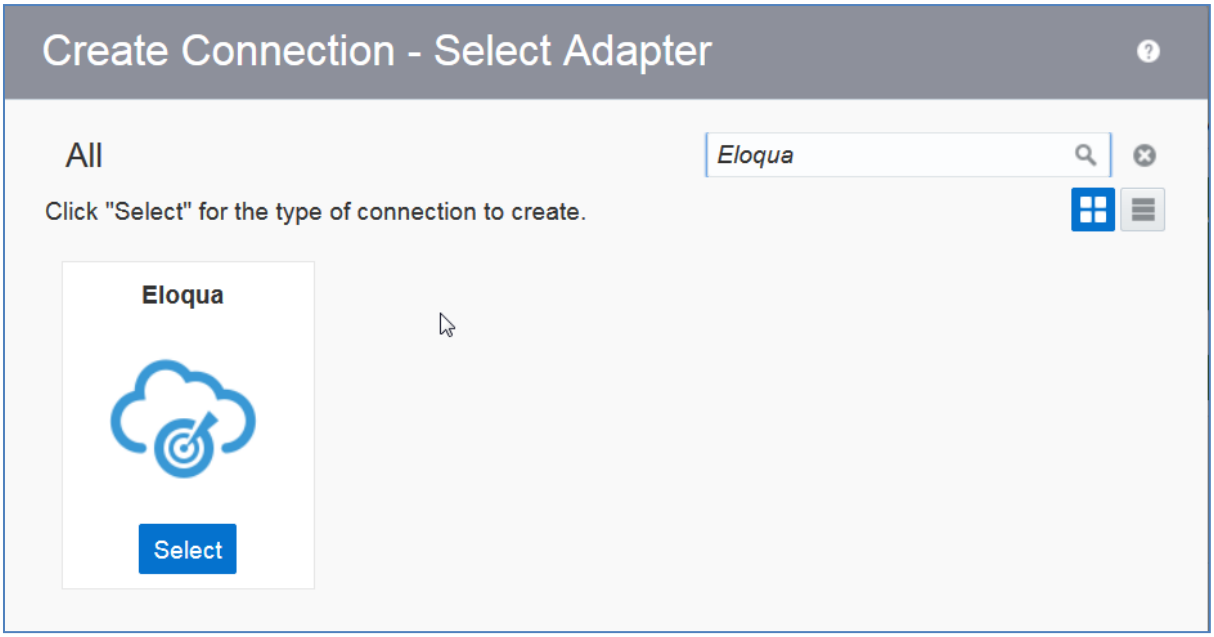
17. Click on **Close** to return back to main page.

Create Eloqua Adapter based Connection

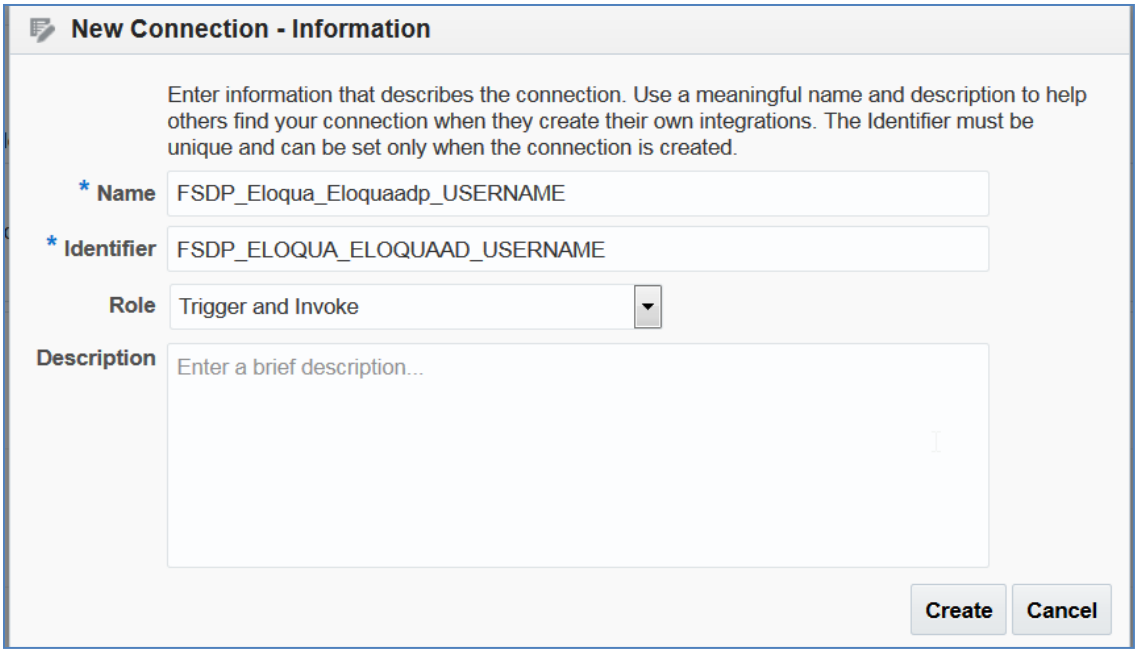
Now we can create the second connection to ELOQUA cloud using Eloqua Adapter. You are on the main connections page now.

1. Click on **Create**

2. In the search window specify **Eloqua** and search. You should get the Eloqua connection**. Select** the **Eloqua** connection**.**



3. Specify the following and click **Create** to create new connection
   Connection Name: FSDP_Eloqua_Eloquaadp_USERNAME (**replace the USERNAME with the user allocated to you**)
   Identifier : Auto Populated
   Role: Trigger and Invoke



Note: We have already completed the pre-requisite of installing OIC for Eloqua app in Oracle Market place to link this OIC instance to Eloqua system.
Tech Note: It is mandatory to have the "OIC for Eloqua" app installed and configured in your Oracle Market Place prior to testing an Eloqua connection using this adapter as "Trigger and Invoke". Below are references for further understanding.
http://docs.oracle.com/cloud/latest/marketingcs_gs/OMCAA/Help/Apps/IntegrationCloudService/IntegrationCloudServiceApp.htm
http://docs.oracle.com/cloud/latest/marketingcs_gs/OMCAA/Help/Apps/IntegrationCloudService/Tasks/InstallingOIC.htm

4. Click on **Configure Security** and provide following values. When done click **Ok**

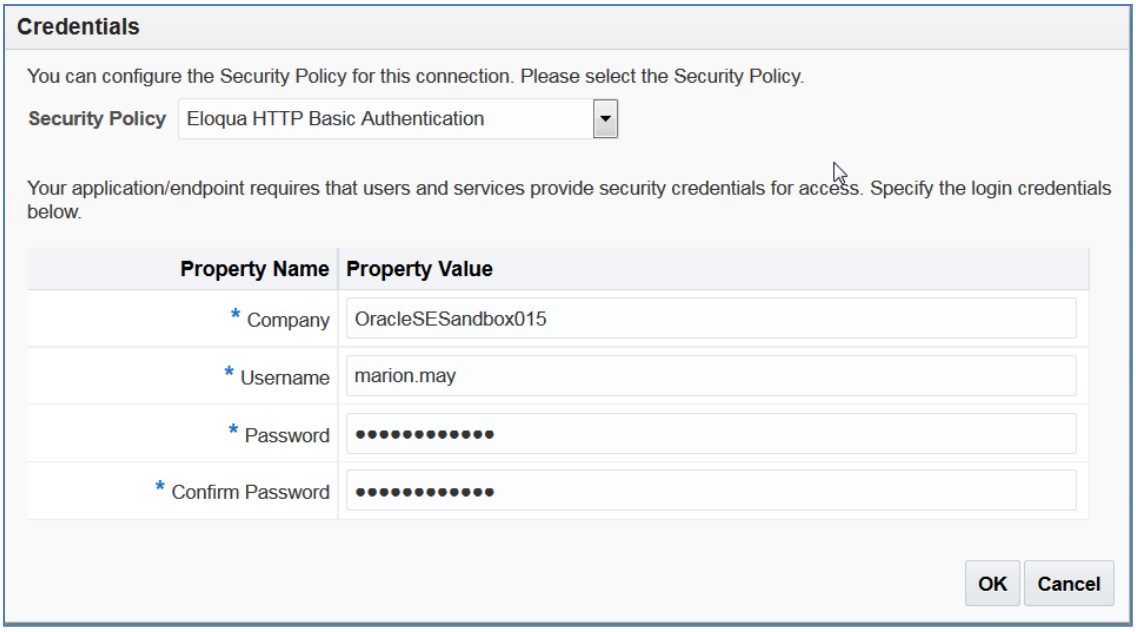
Security Policy : Auto Selected

Company : CompanyName provided to you

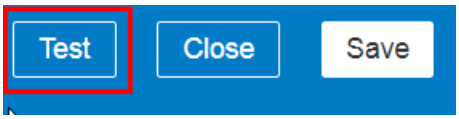
User Name : User provided to you

Password: password provided to you

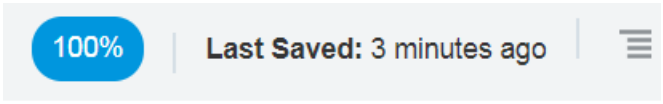
Note: <Use the Eloqua REST endpoint credentials provided by instructor/worksheet>



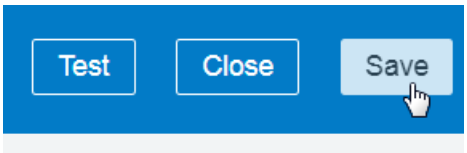
5. Click Test at the top to test the connection



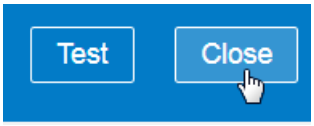
6. After Test is successful, you should now see 100%.



7. Click on **Save** to save the connection. You should see a message now at the top



8. Click on **Close** to return back to main page.

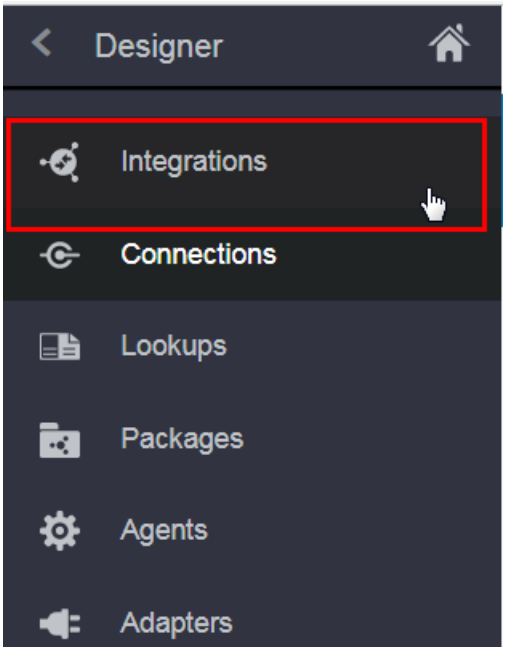


Create OIC – Eloqua Integration

Configure and Map Endpoints

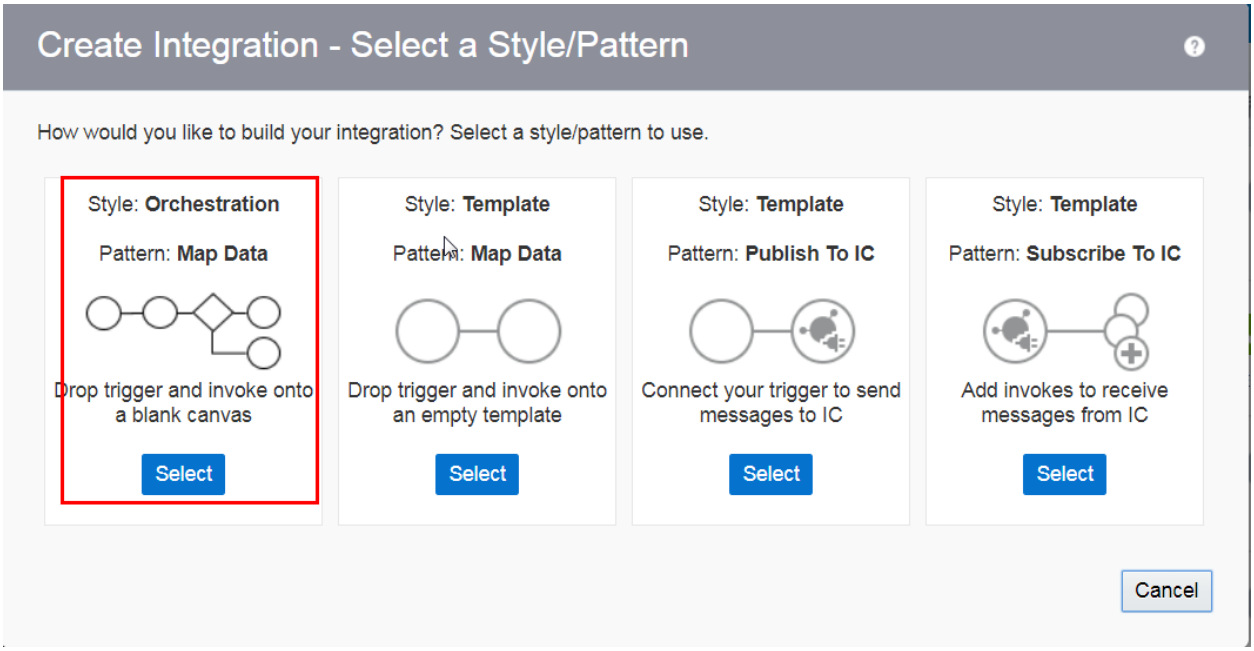
Create and configure SynchContacts Integration

1. You are on OIC home-> Designer. Click on **Integrations** on the left

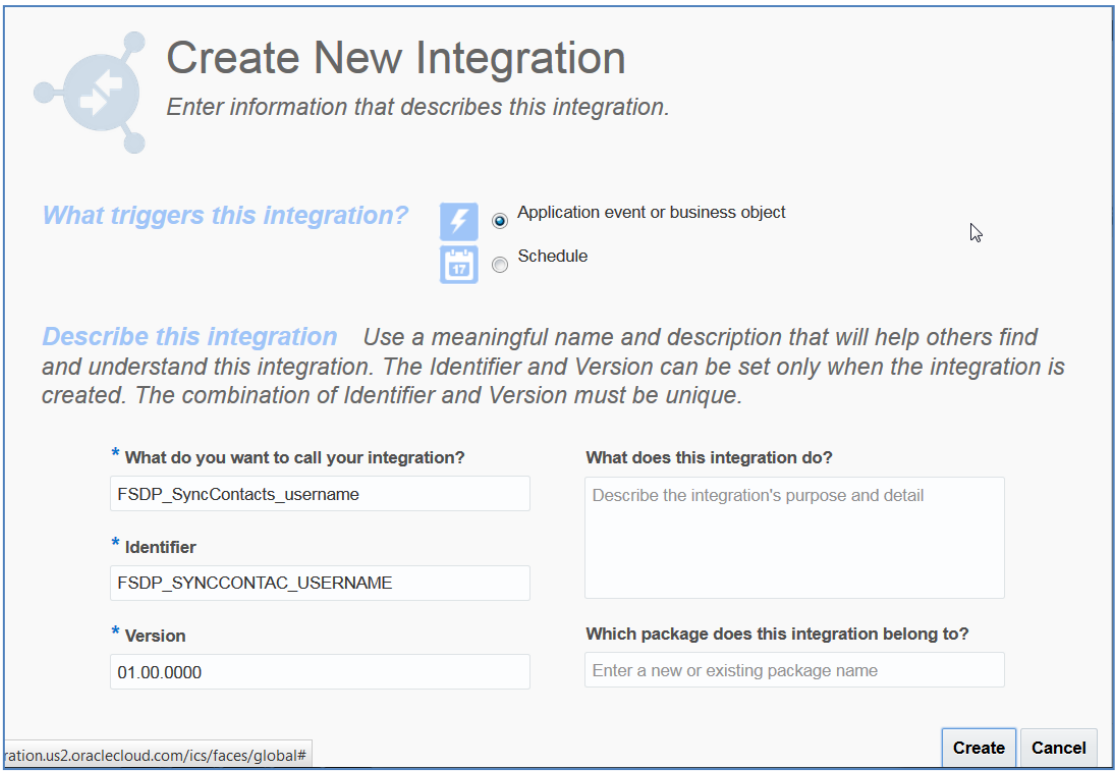


2. Now click on **Create** to create the integration

3. Select "**Orchestration**".

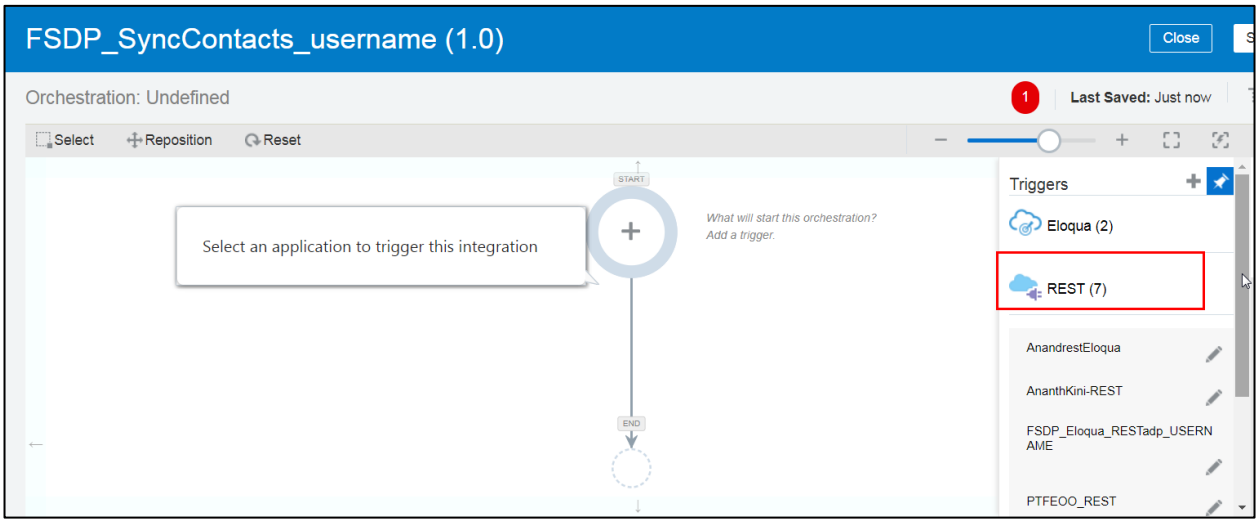


4. Name it as "**FSDP_SyncContacts_username**" (**replace _username** with the login name assigned to you) and leave the other fields to defaults

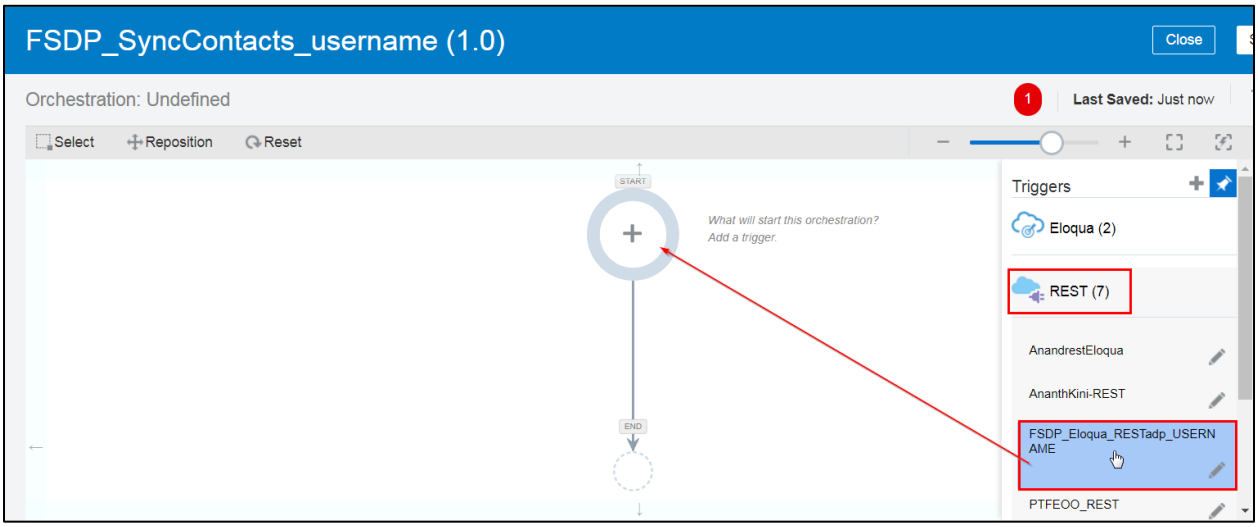


5. Click "**Create".** Integration will be created and you will be launched on to it in Edit mode.
6. On the right, under the list of connectors, select "**REST**" -> "**FSDP_Eloqua_RESTadp_username**" (replace **_username** with the login name assigned to you)

Note: This is the one that you created in previous section.



7. Drag and drop it on the circle which will start the REST Endpoint Configuration Wizard.



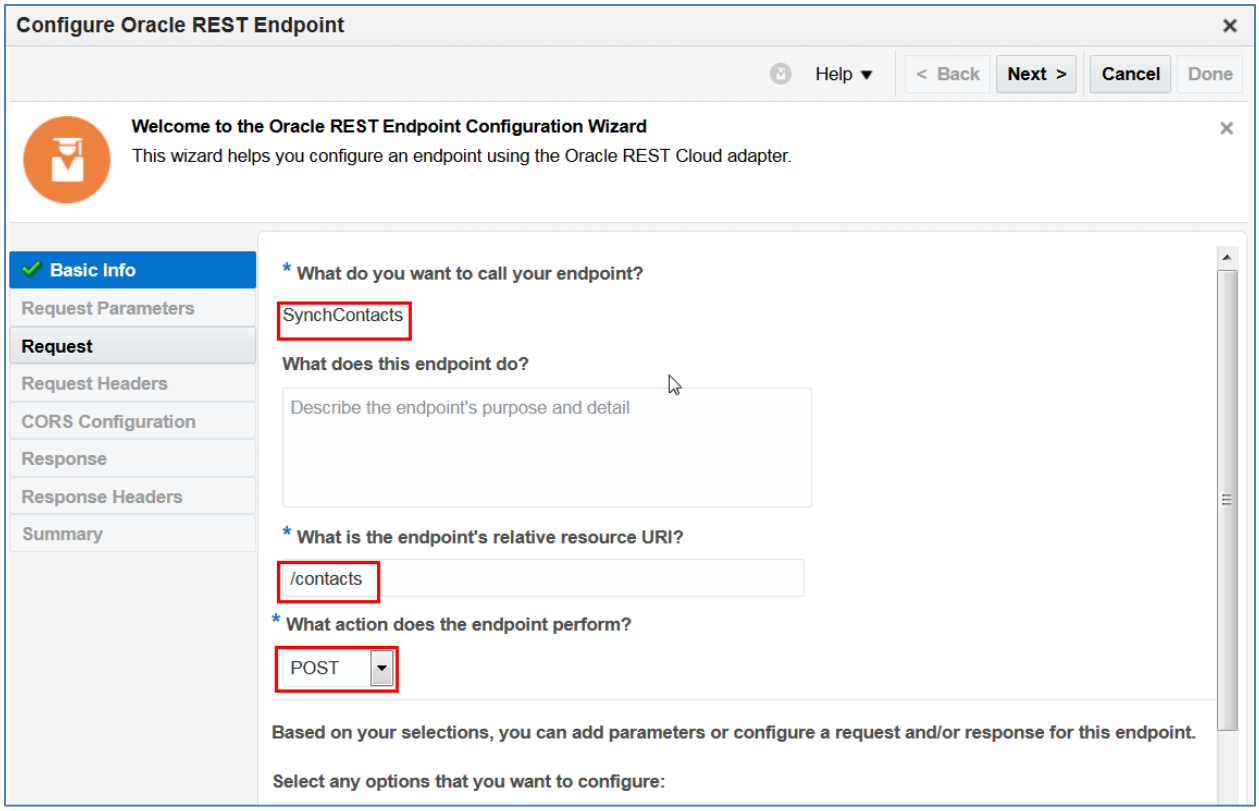
8. Provide the following values

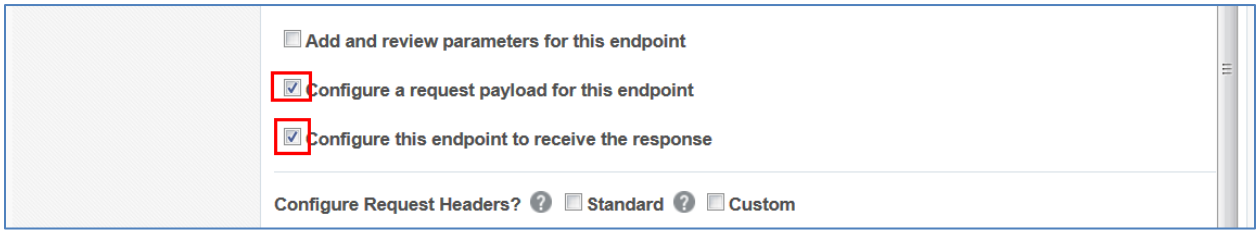
What do you want to call your endpoint : SynchContacts

What is the endpoints resource URI: /contacts

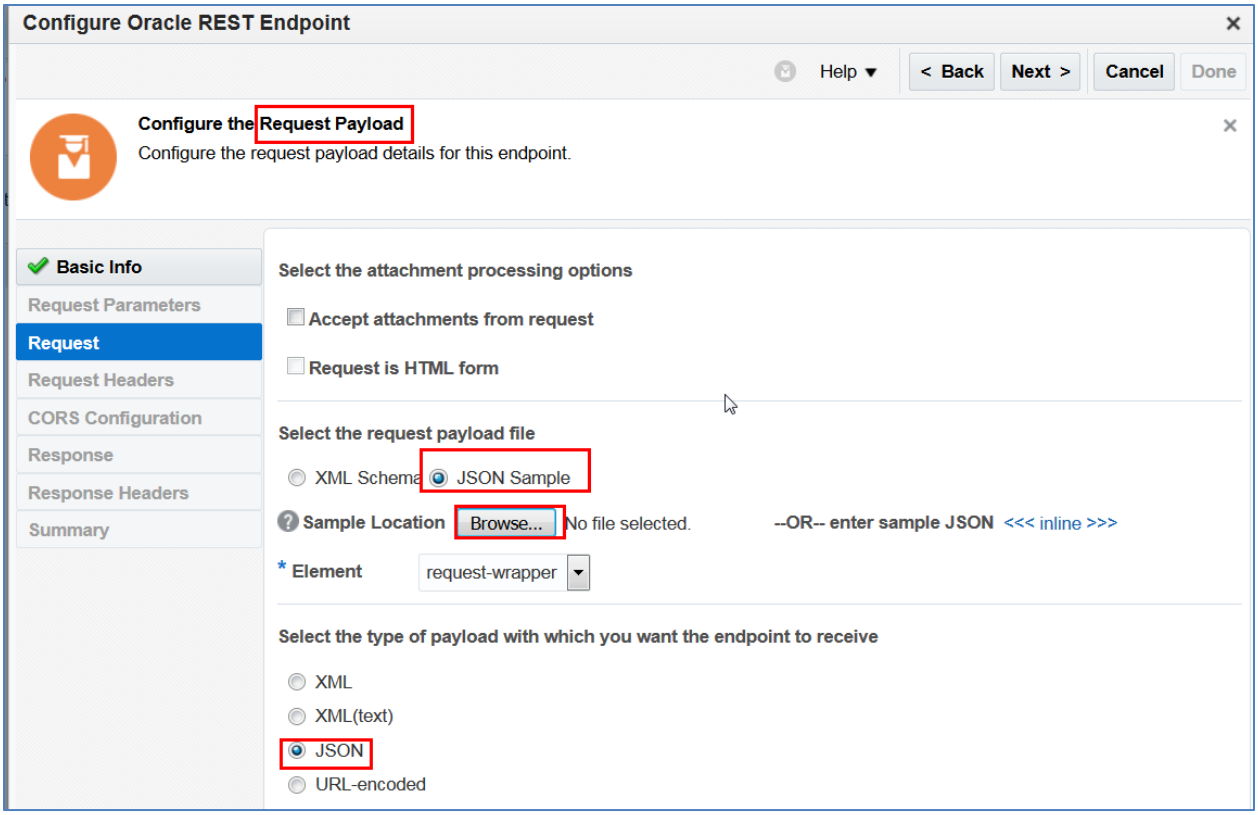
What action does the endpoint perform : POST

Select the "Configure Request and Response" payload checkboxes.

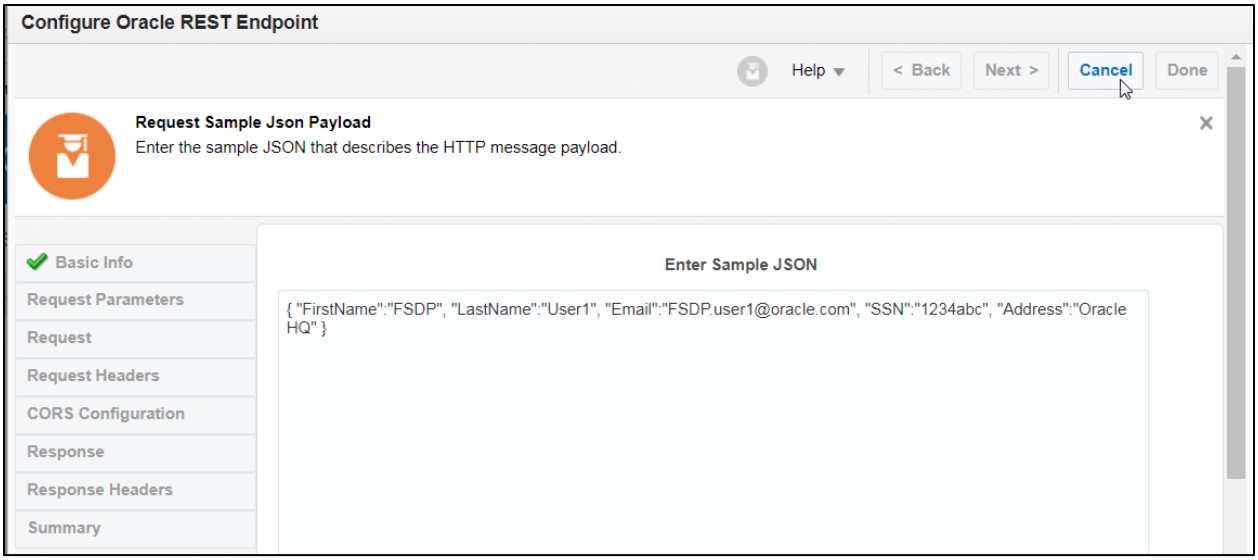




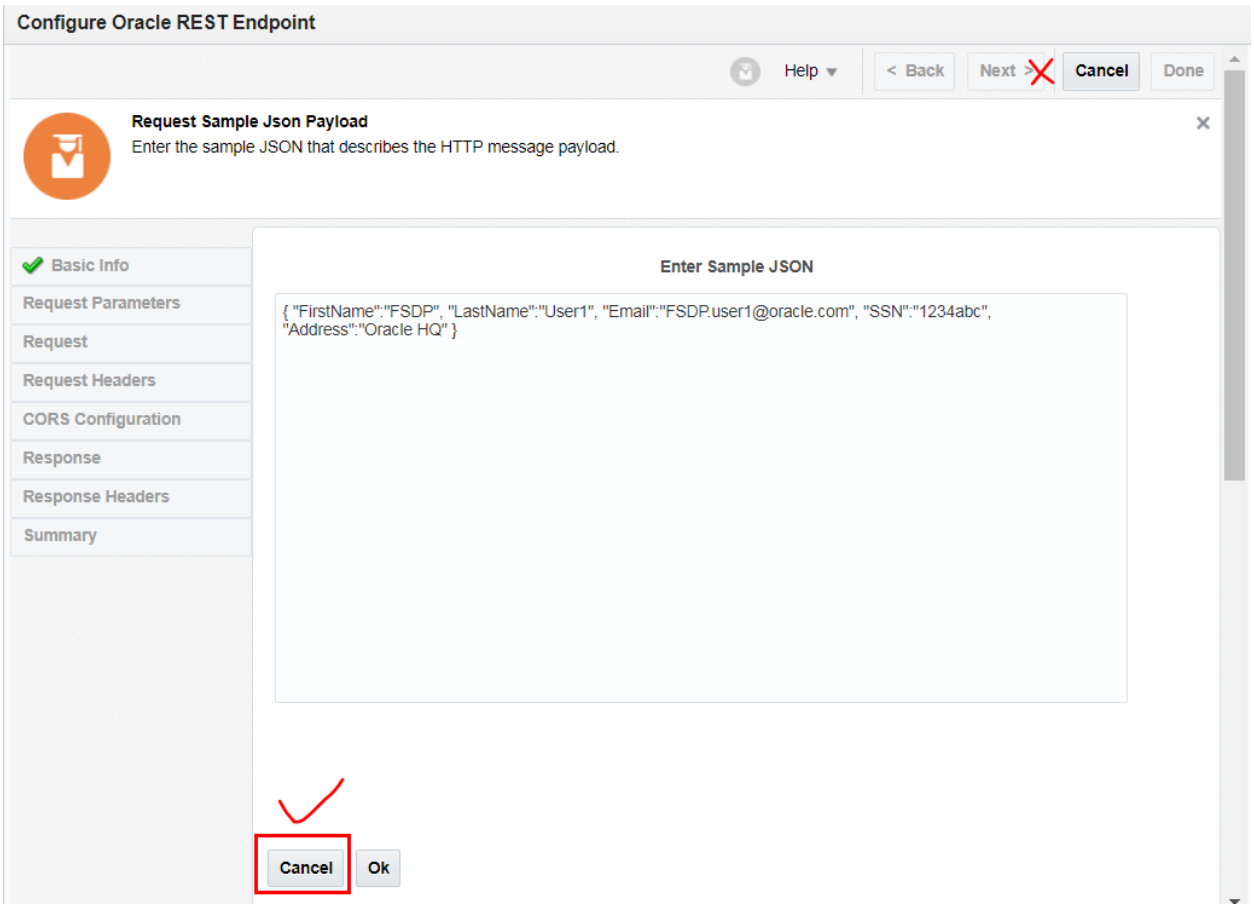
9. Click **Next.**
10. Select "**JSON Sample"** as the REQUEST payload file and "**Browse"** and select "**SynchContactsRequest.json**" provided to you by the instructor.



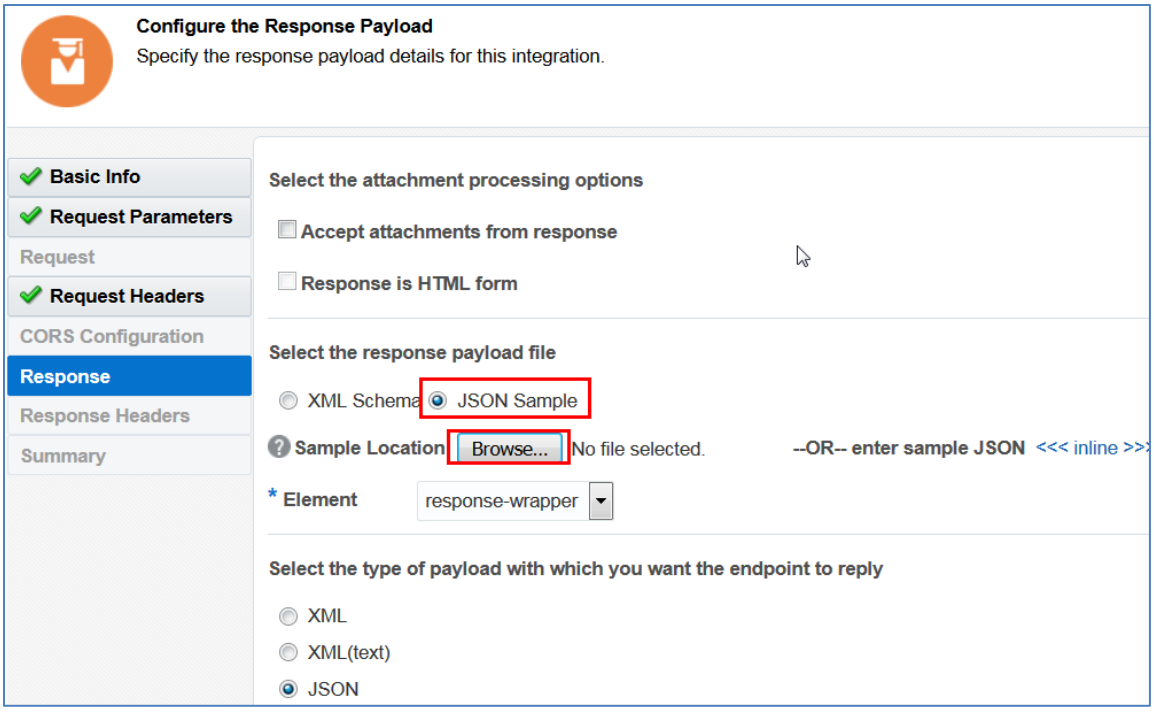
11. Click **"<<<inline>>>"** to view the uploaded json. The contents should look like below.
    Note: (If you don't see any values, either **re-select** the file or click "update" button you may see). DO NOT proceed unless you see values



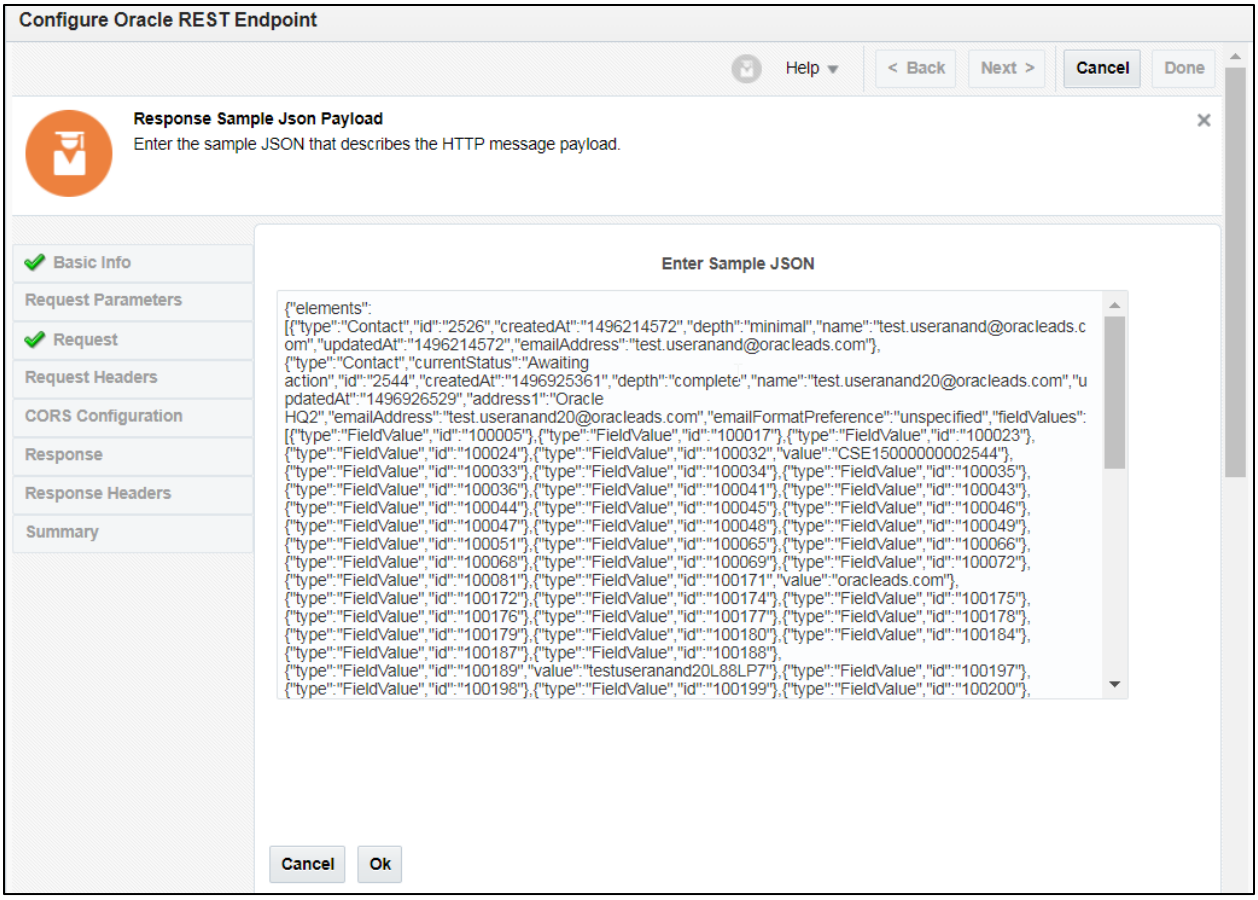
12. Click "**Cancel**" on the bottom. DO NOT click the one on top.



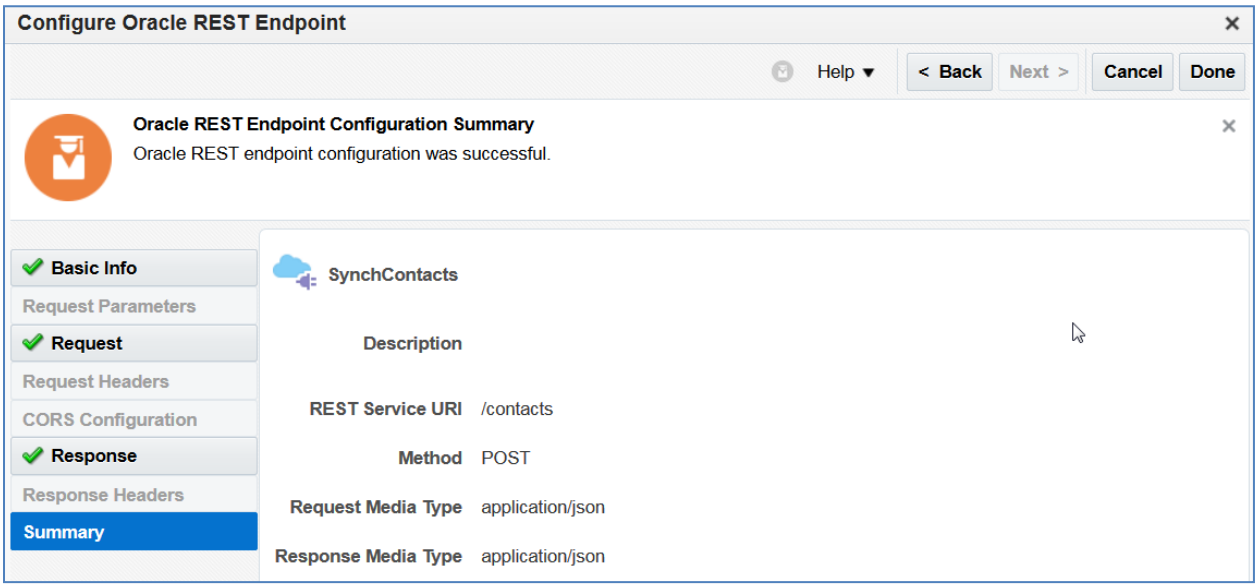
13. Click **Next.**
14. Select "**JSON"** as the RESPONSE payload file and "**Browse"** and select "**SynchContactsResponse.json**" provided to you by the instructor.

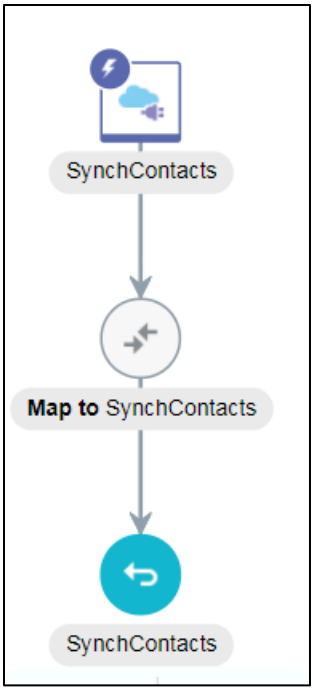


15. Click **"<<<inline>>>"** to view the uploaded json. The contents should look like below.
    Note: (If you don't see any values, either **re-select** the file or click "update" button you may see). DO NOT proceed unless you see values



16. Click "**Cancel**" on the bottom. DO NOT click the one on top.
17. Click "**Next"**
18. Review summary and click "**Done" .** You will see the Orchestration created

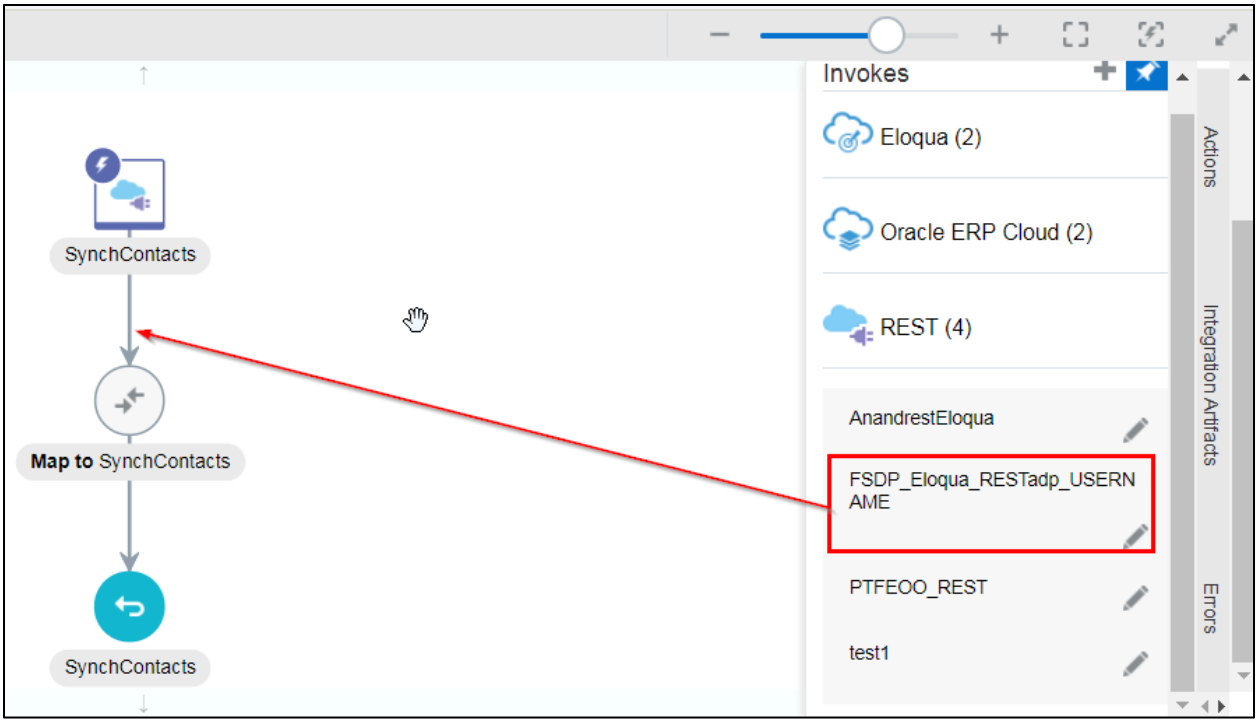


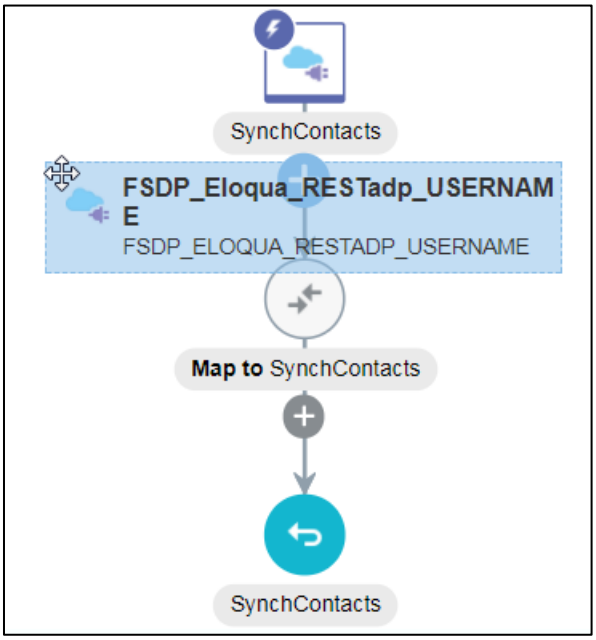


19. **"Save"** the integration.

Configure GETContact

1. On the right, under **INVOKES**, click "**REST**" -> "**FSDP_Eloqua_RESTadp_username**" (NOT **_username** but the one which you created with the login name assigned to you in previous section)
2. Drag and drop it in between SynchContacts **Trigger** and **Map (on the EXACT spot as shown below)**





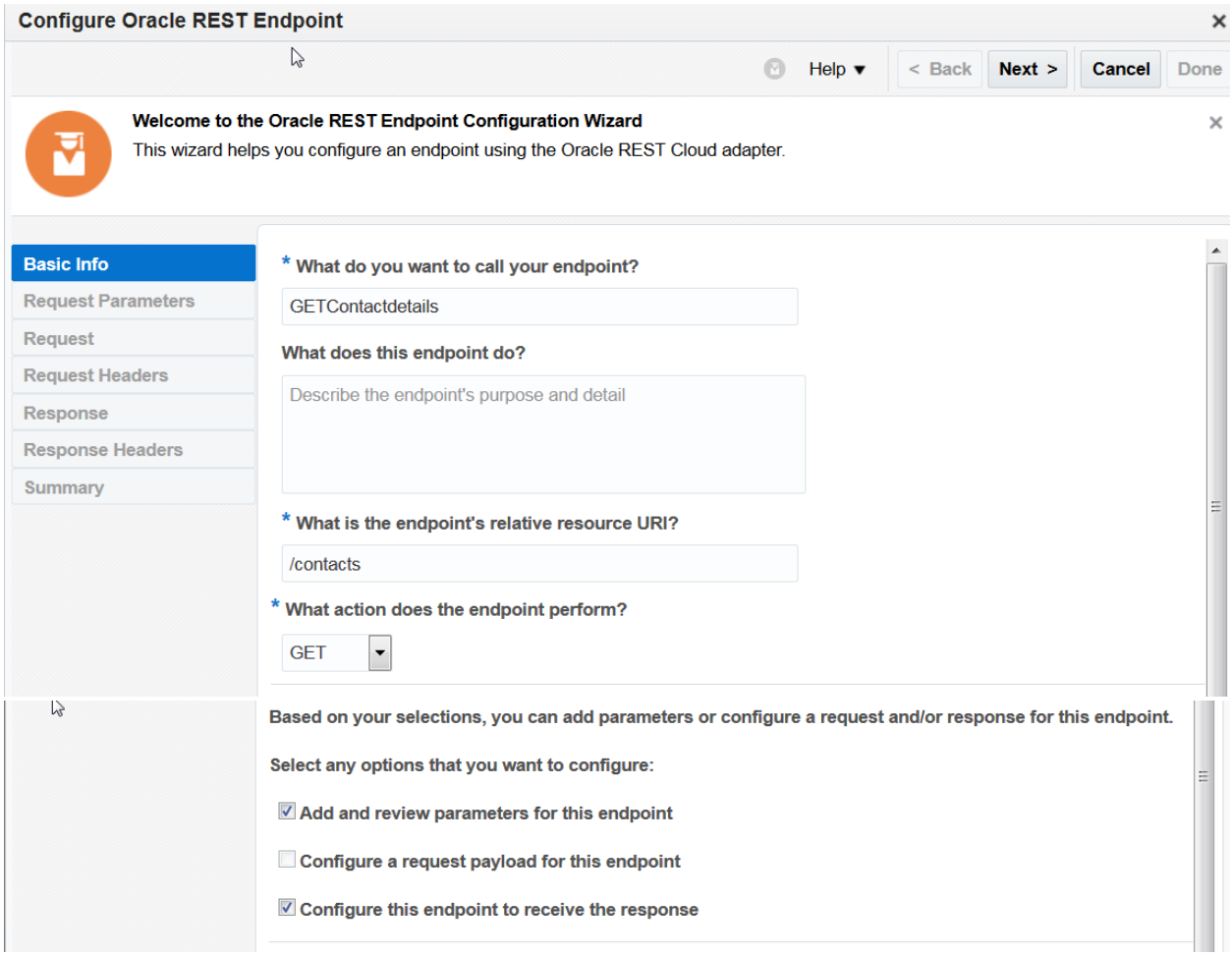
3. Provide the following details

What do you want to call your endpoint : GETContactdetails

What is the endpoints resource URI: /contacts

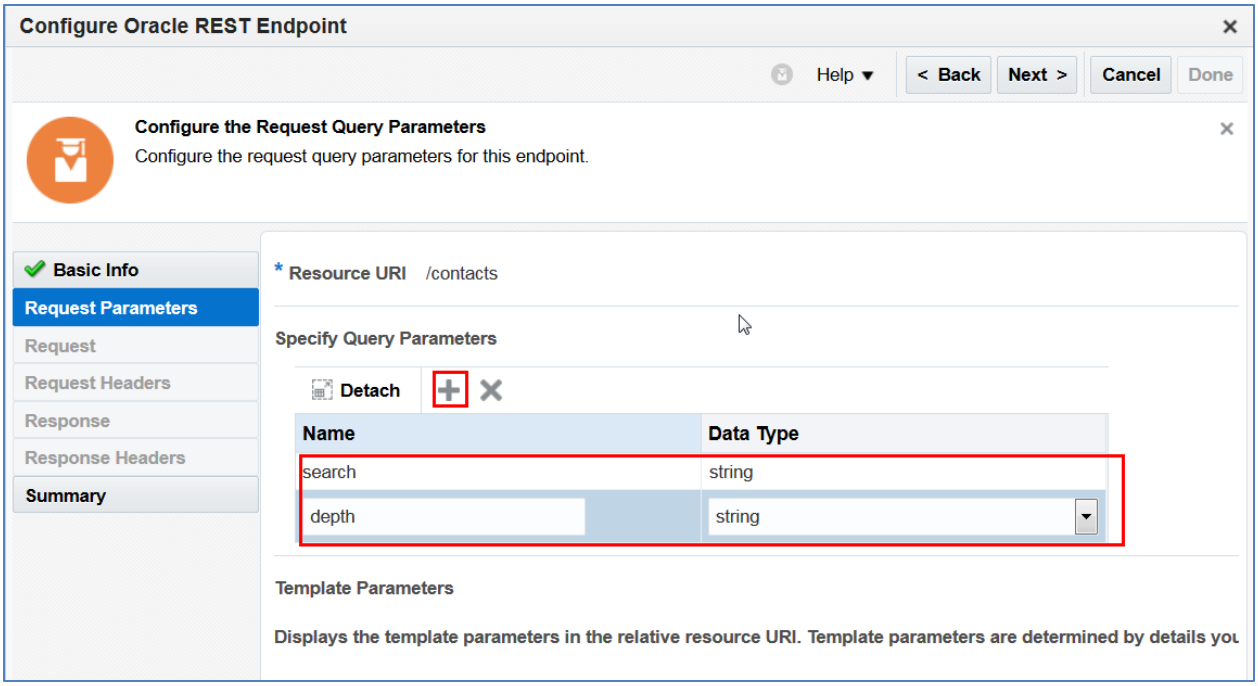
What action does the endpoint perform : GET

Select the "Add Review Parameters" and "Configure Response" payload checkboxes.

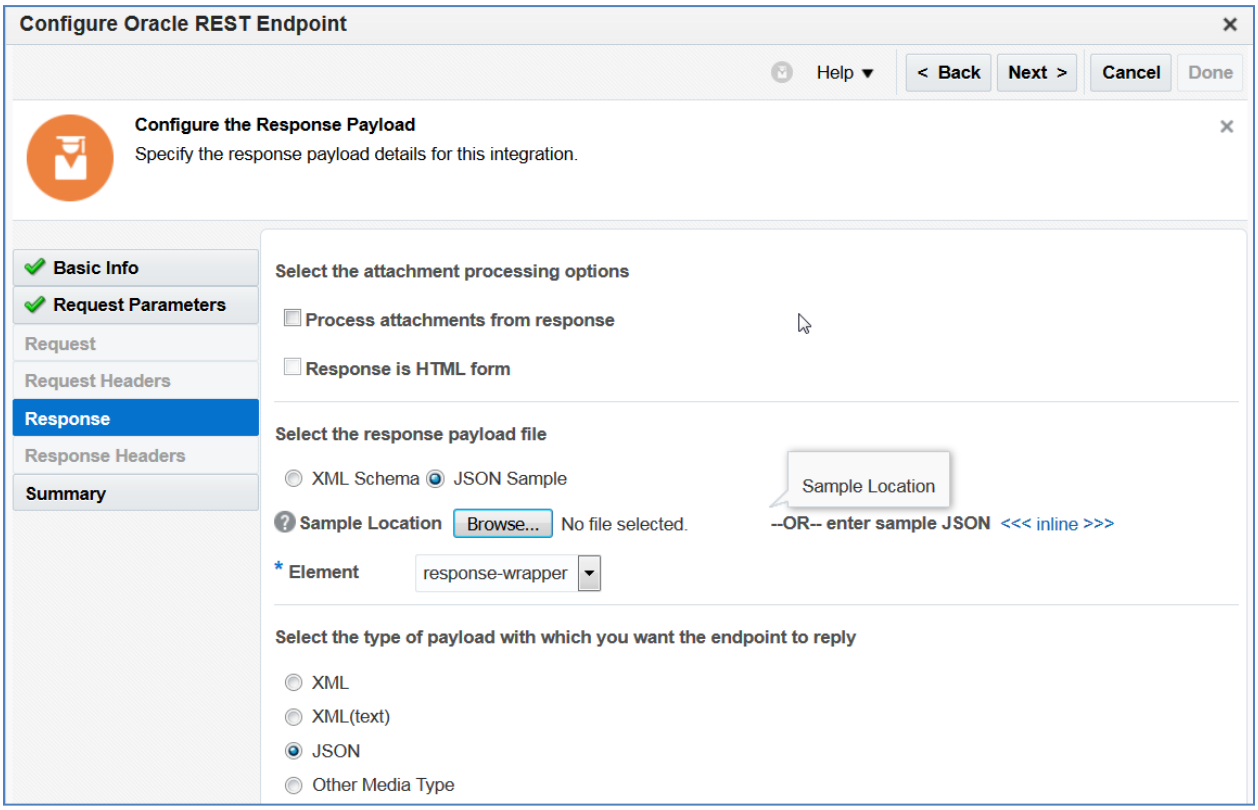


4. Click "**Next".** Add below request parameters by clicking "**+**" button

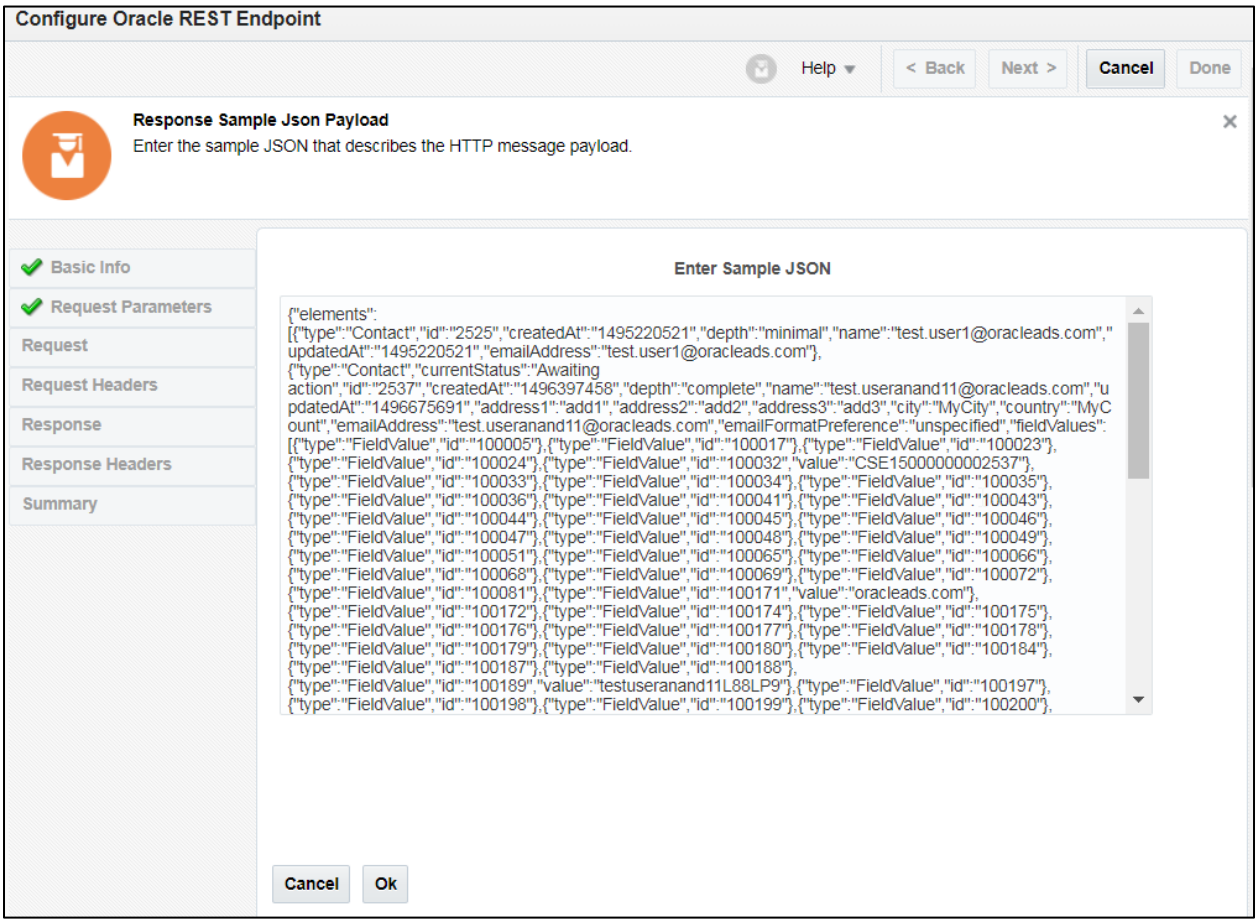
Name	DataType
search	string
depth	string



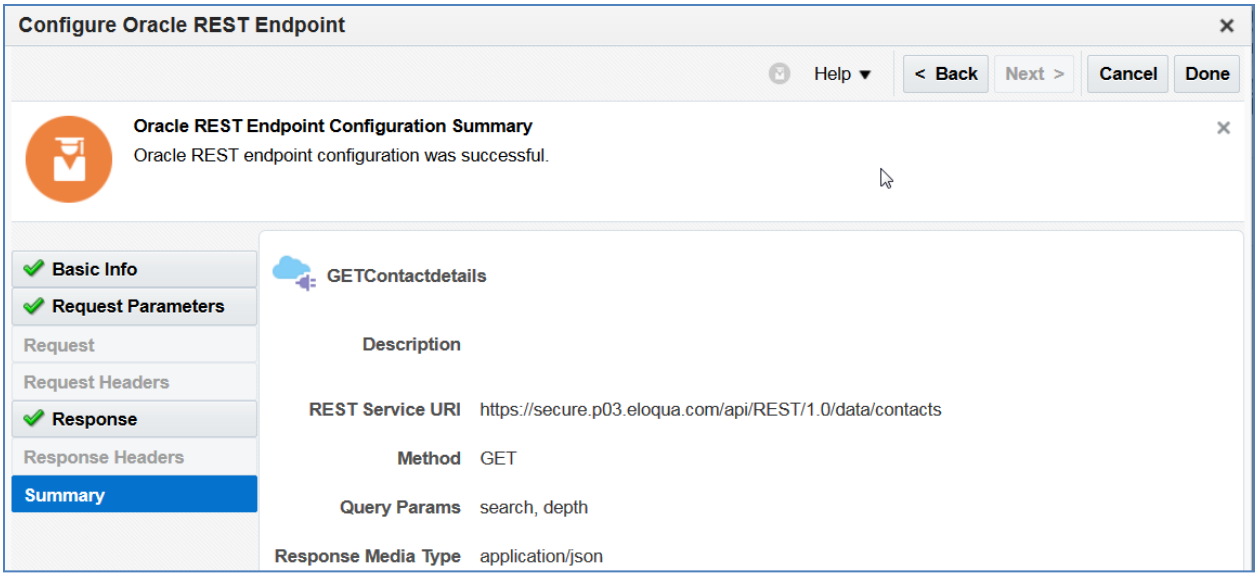
5. Click "**Next".**
6. In the RESPONSE config page, Select **JSON Sample** type and choose "**GETContactDetailsResponse.json**" file given by the instructor.



7. Ensure that you see “<<<inline>>> as below.
   Note: (If you don’t see any values, either **re-select** the file or click “update” button you may see). DO NOT proceed unless you see values…



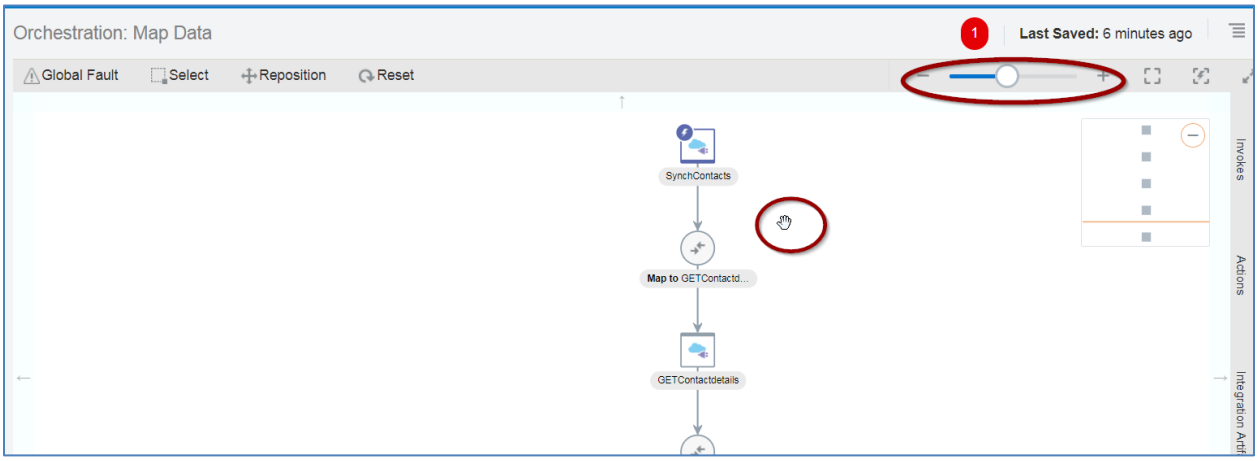
8. Click “**Cancel**” on the bottom. DO NOT click the one on top.
9. Click “**Next”** and review configuration summary.



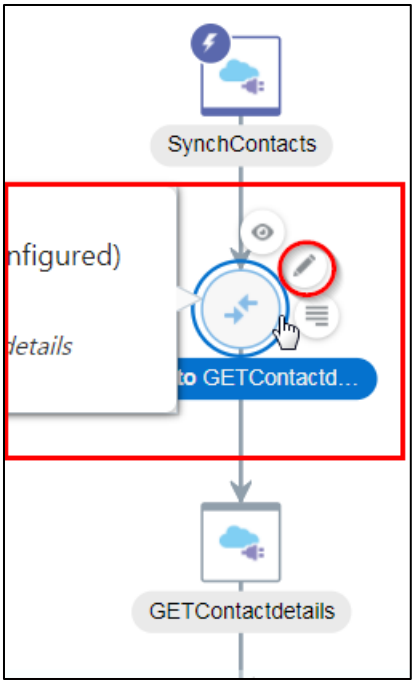
10. Click **Done** and "**Save"** the integration.

Map GETContact

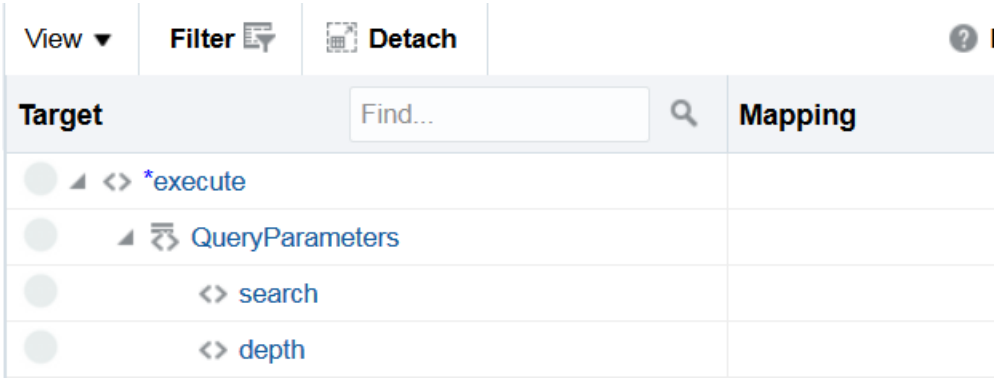
Note: You may **Zoom-in or Zoom**-out the Orchestration to get greater visibility of the integration either by **scrolling** mouse over Orchestration page or using the **-,+** buttons.



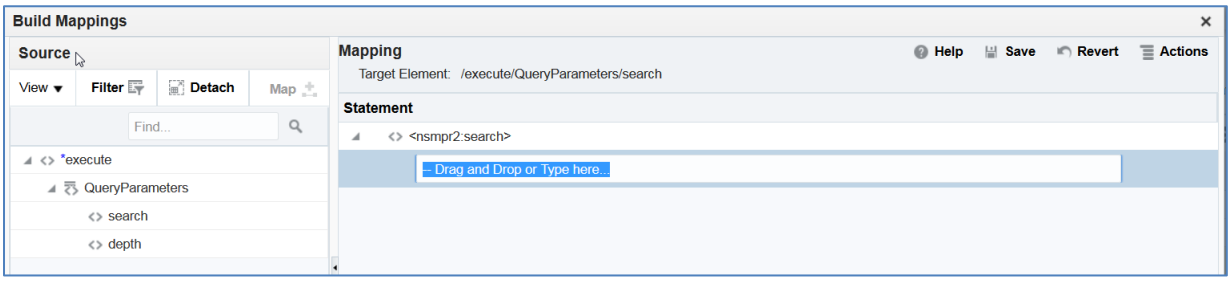
1. Click on "**Map to GETContactDetails"** MAP -> "EDIT"



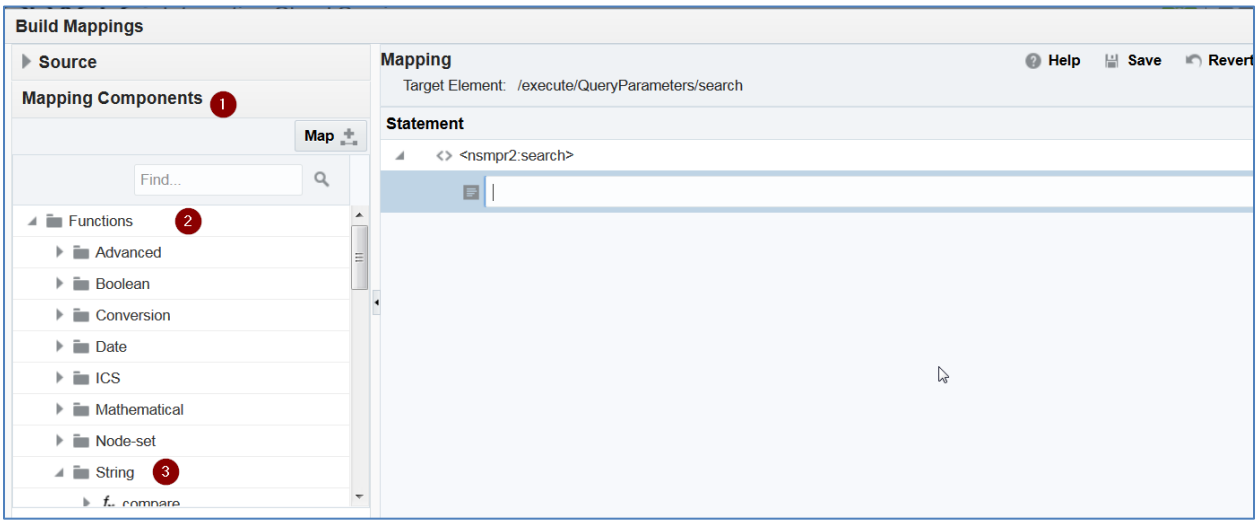
2. Click on "**search"** target on the RIGHT side.

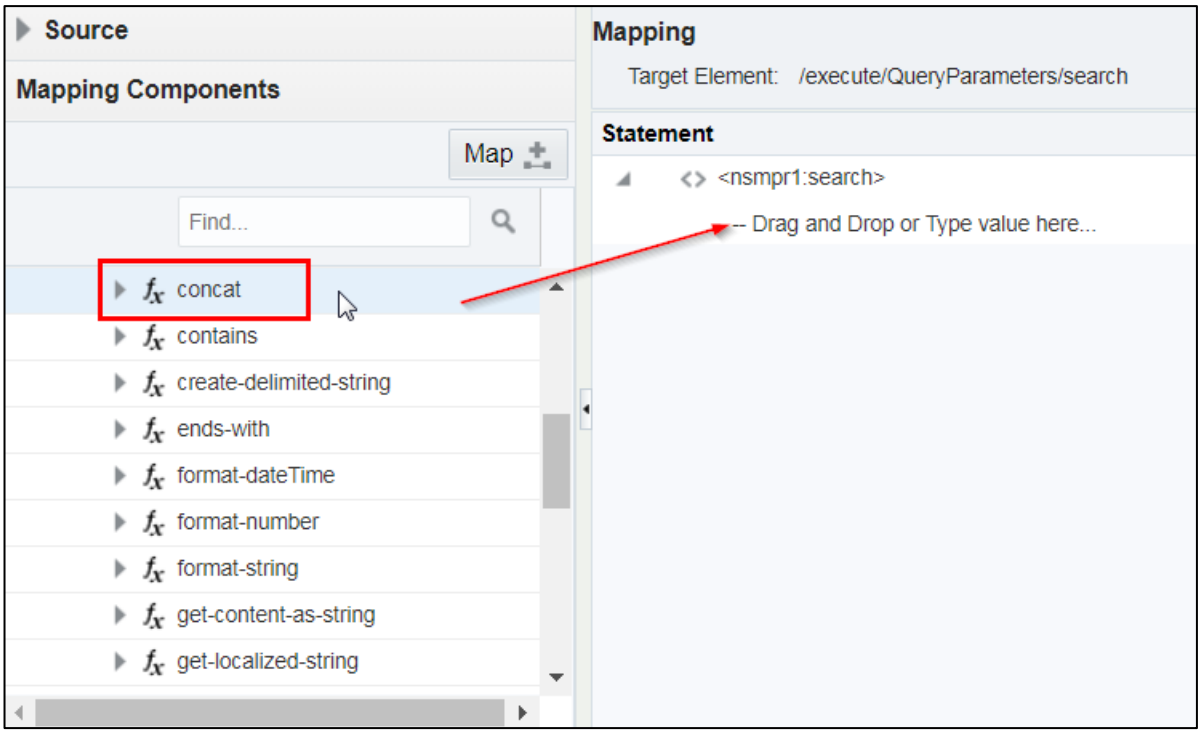


3. In the Mappings Wizard, **click** on "Drag and Drop here…"



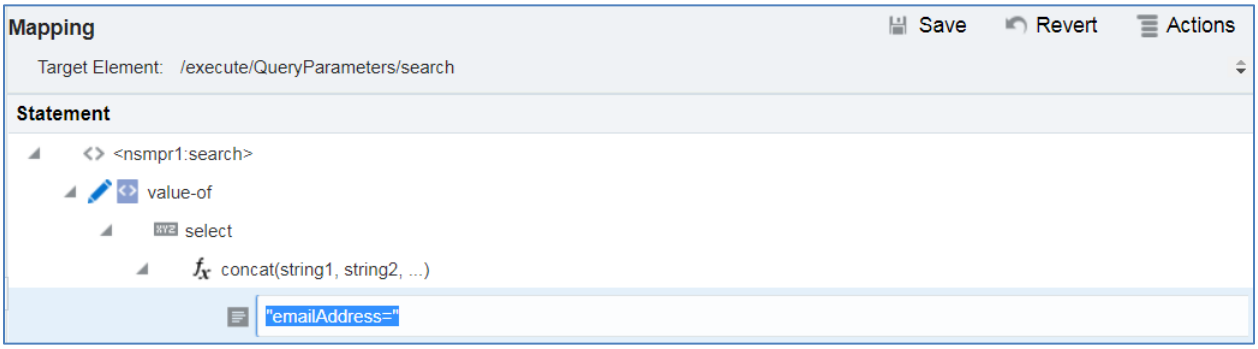
4. On the left hand side, expand “**Mapping Components”->Functions->String**, and select “fx **concat”** and drag/drop it to the Statement field on the right.



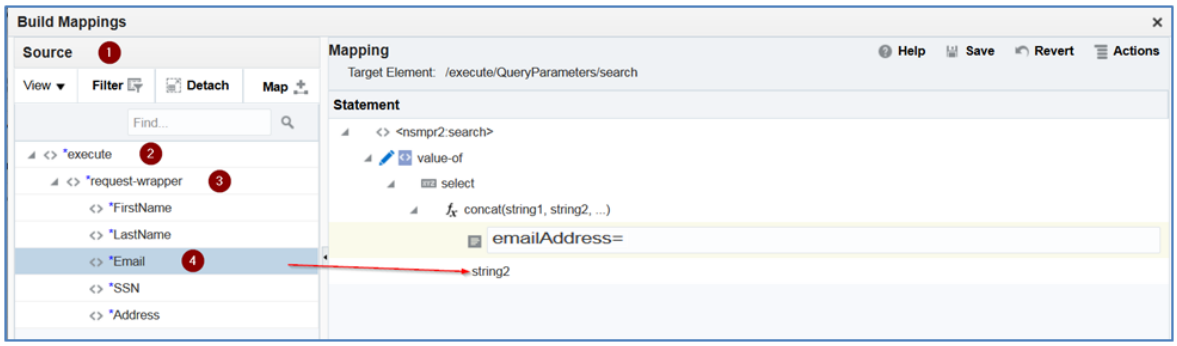


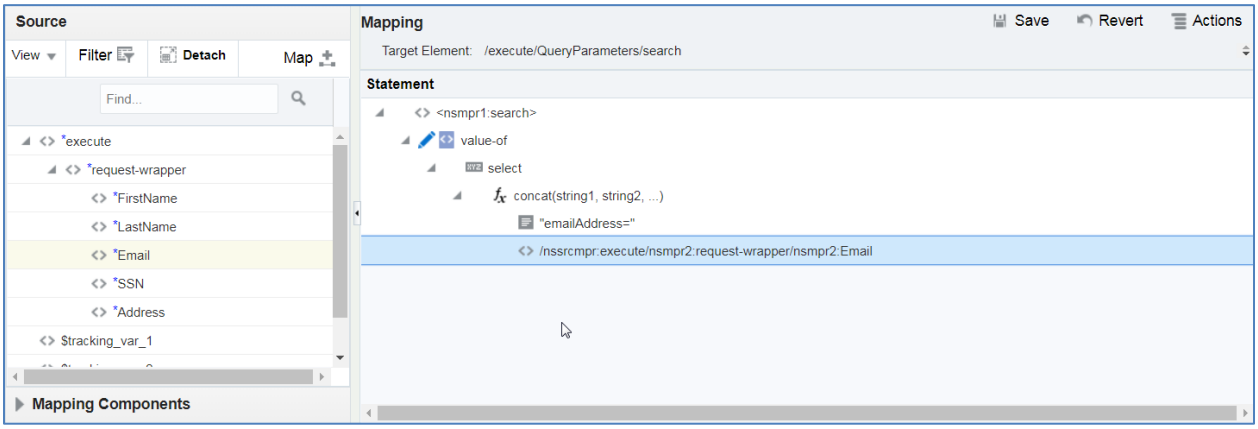
5. Click on “**String1**” and type “**emailAddress=”. (including double-quotes)**

Note: Please copy to a notepad and then to OIC. Do not copy/paste directly from here.



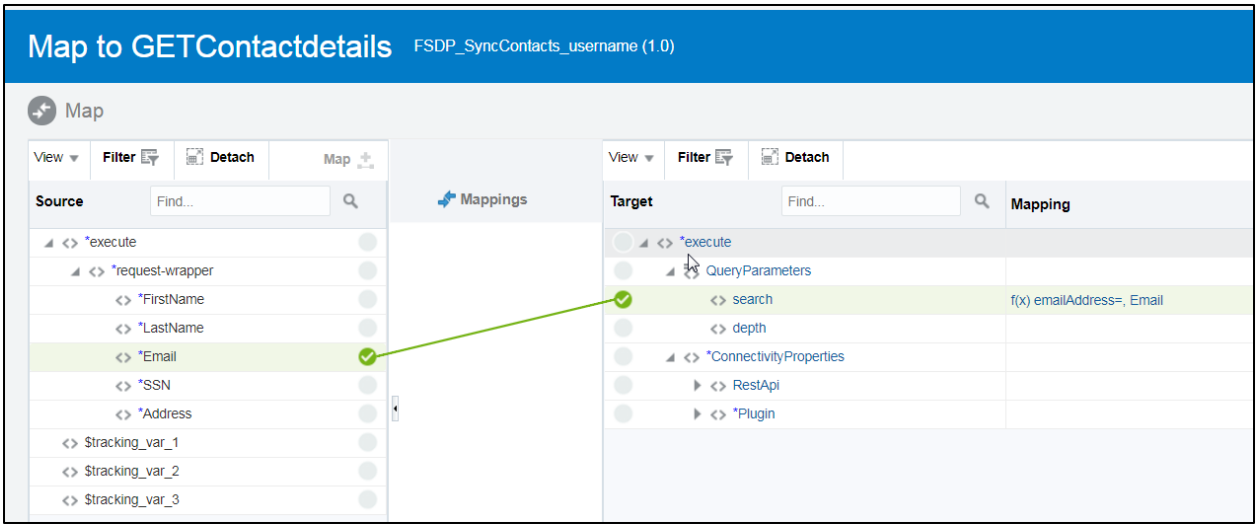
6. For **String2** drag&drop (from the left hand side) **Source->execute->request-wrapper->Email**



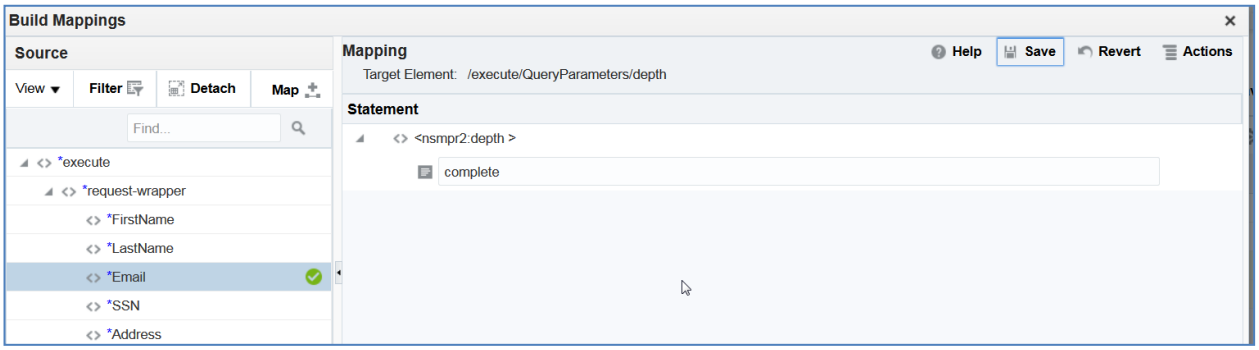


7. Click "**Save".**

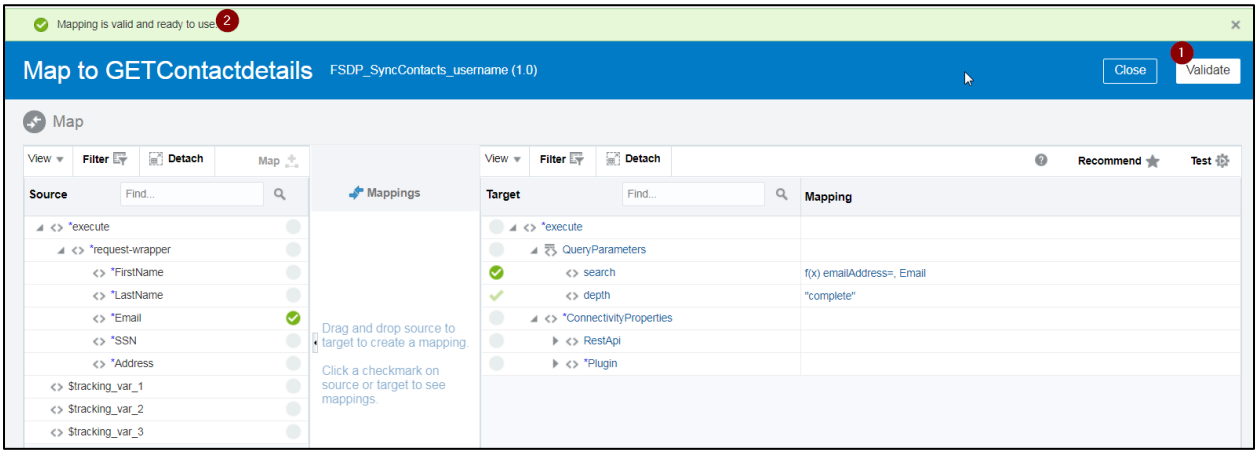
8. Click "**Close"** (on bottom right)**.** Now, it should look like below.



9. Now, click on the target "**depth**"
10. In the "-- Drag and Drop or Type here..." **click** and type "**complete**" (double quotes NOT required).



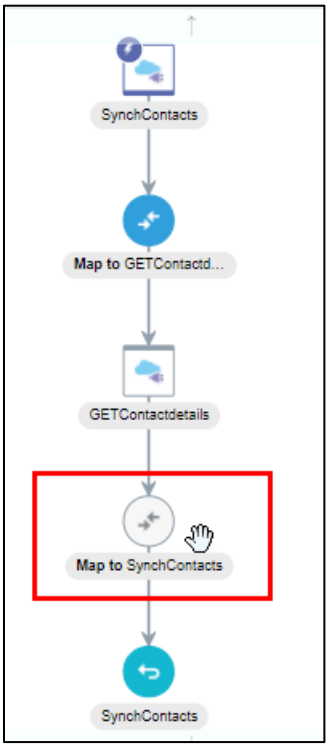
11. Click "**Save"** and then "**Close".**
12. On the main mapping page, **verify** the mapping as below. Click **"Validate" .**You should see a success message.



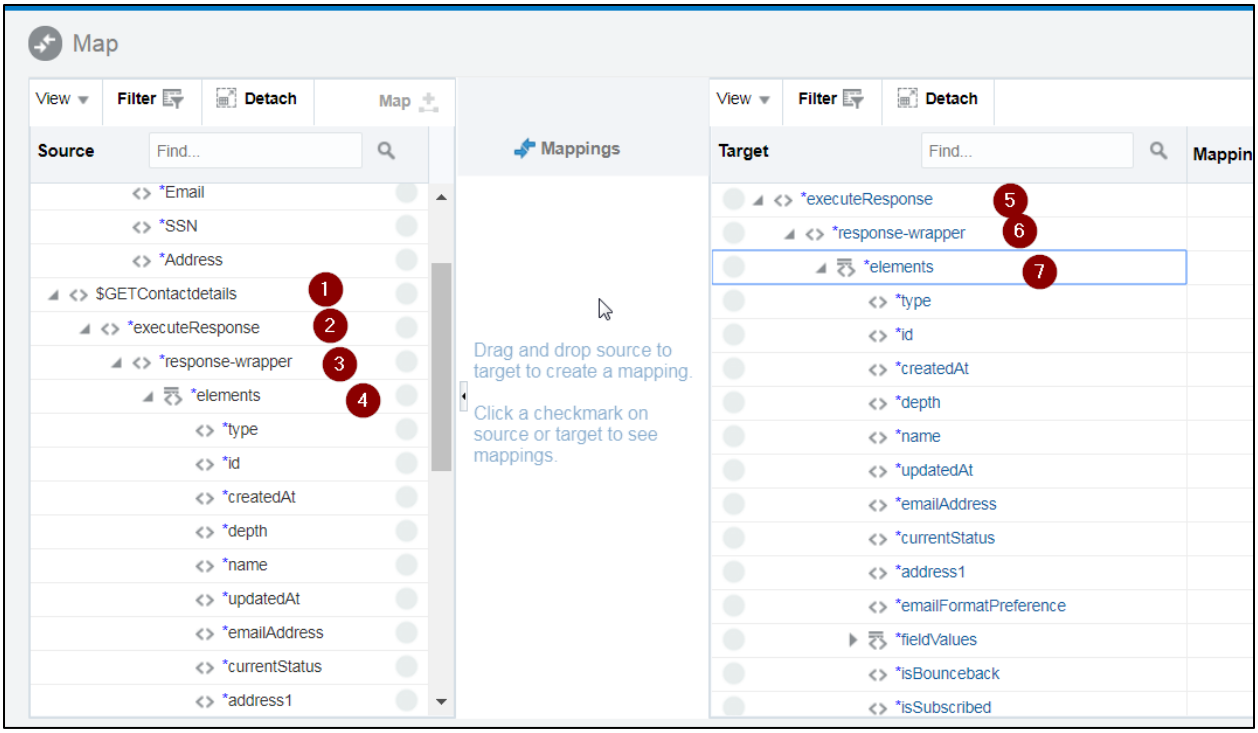
13. Click "**Close**"
14. "**Save**" the main integration again.

Map SynchContacts endpoint

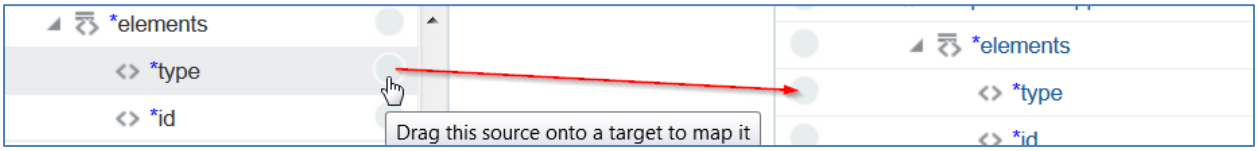
1. Click on **"Map to SynchContacts**" mapping and click "**edit". (Scroll your mouse on Orchestration designer** if you are unable to see the entire Orchestration)



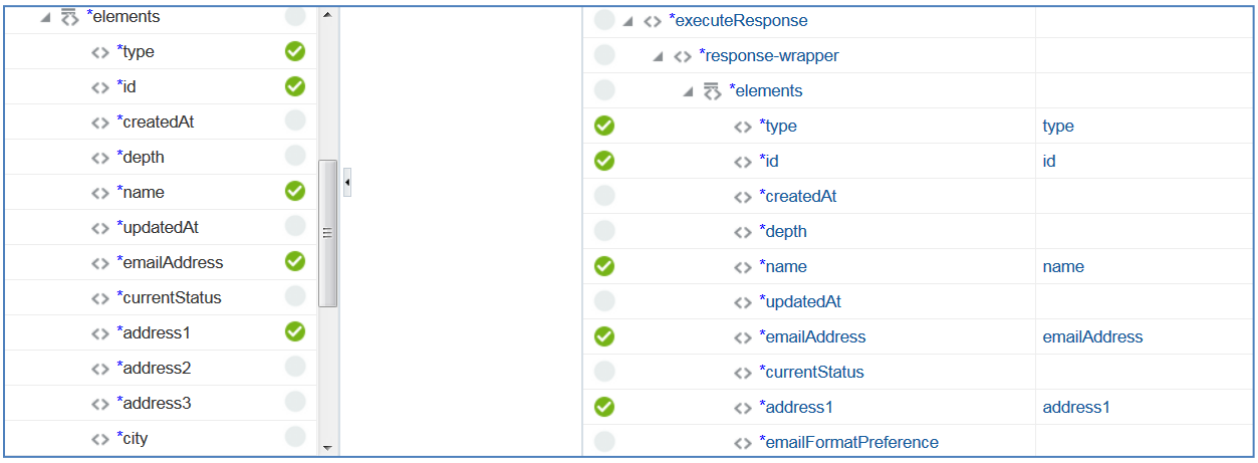
2. Expand the **Source** elements under "**$GetContactdetails"** on the left and **Target** elements on the right.

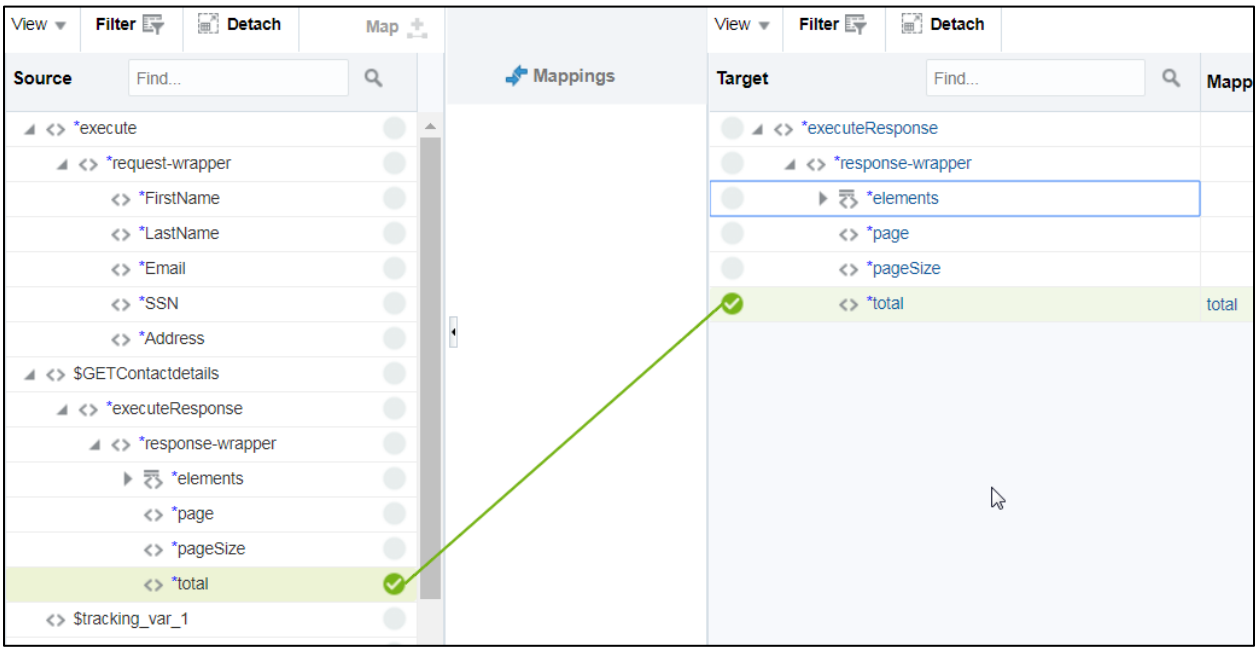


3. Drag and drop the below elements from the respective **source** to **target (you need to drop into the "ball")**



Source	**Target**
type	type
Id	Id
Name	Name
emailAddress	emailAddress
address1	address1
total (**this on level of "elements". So you need to collapse "elements" or scroll down). See the screenshots below**	Total (**this on level of "elements". So you need to collapse "elements" or scroll down). See the screenshots below**





4. Click "**Validate" and "Close"**
5. Click "**Save"** integration again

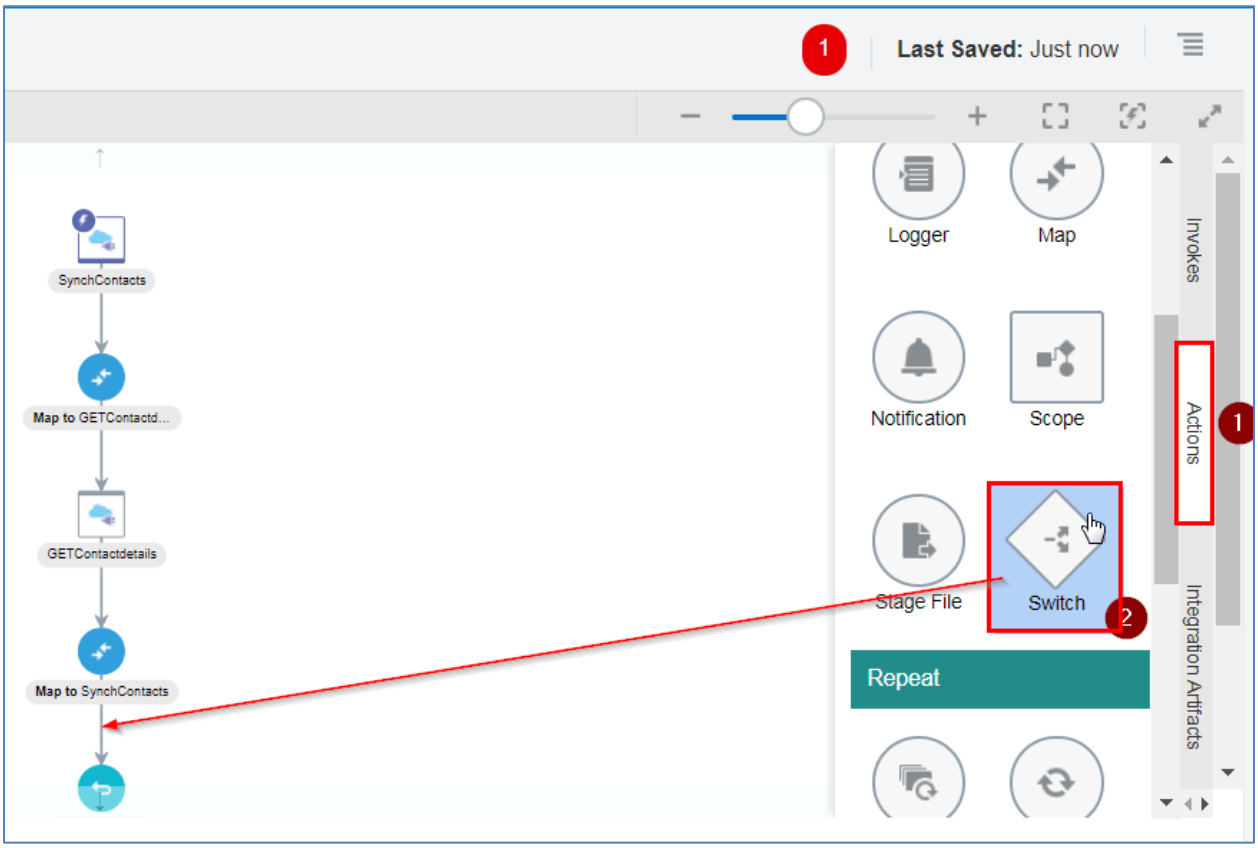
Configure Swtiching Rules

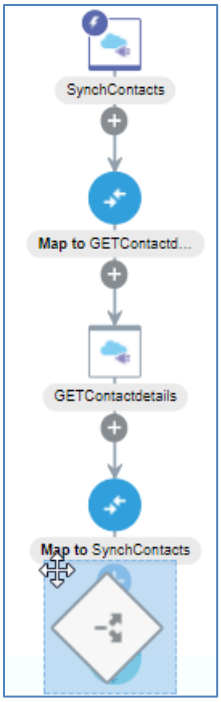
Now, we need to configure the Switch based on the results of GETContacts. The flow would look like below…

##################
if contact exists{

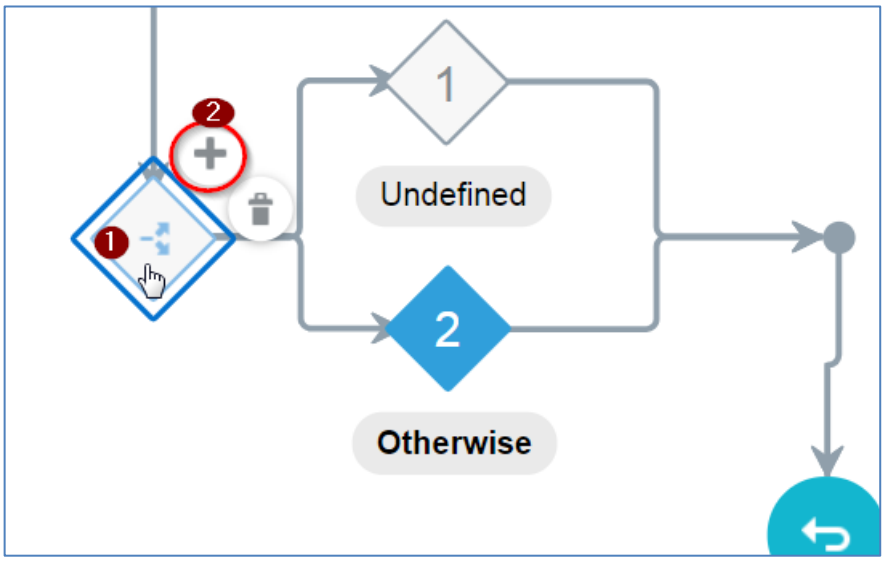
```
validate address
        if address same {
                return results
        }
        else {
                update address in Eloqua
            }
        }
else {
                Create Contact in Eloqua
        }
###################
```

1. On the main Orchestration designer page, Click on "**Actions**" (on the right).
2. Choose **"Switch"** and drop it under "**Map to SynchContacts**" mapping**.**

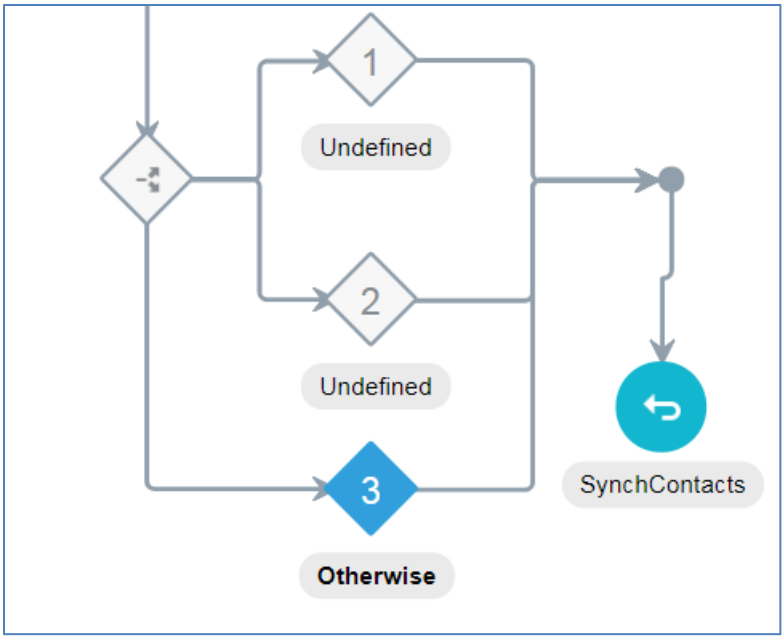




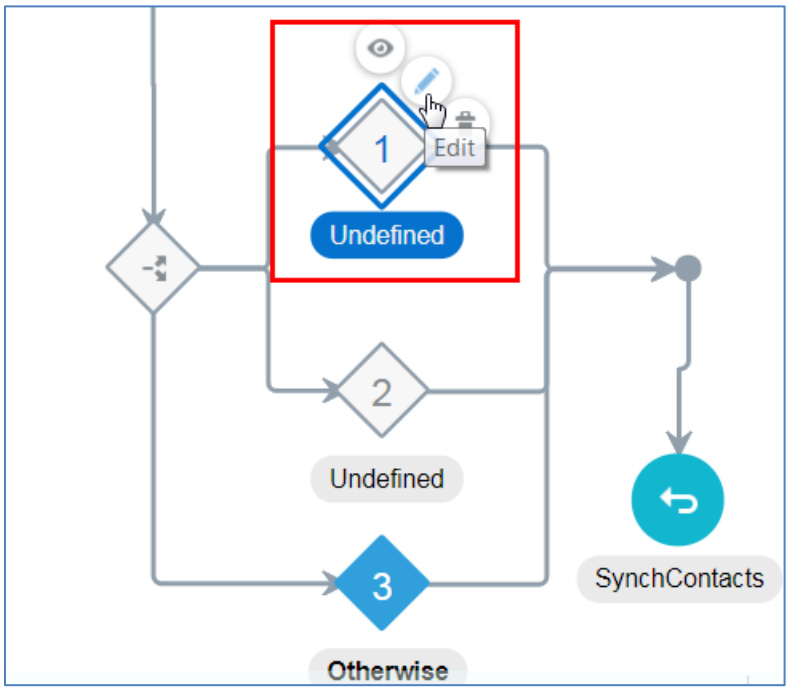
3. It should create a Switch with 2 routes. We need to add more routes.
4. **Click** on the **Switch** and click **"+"**



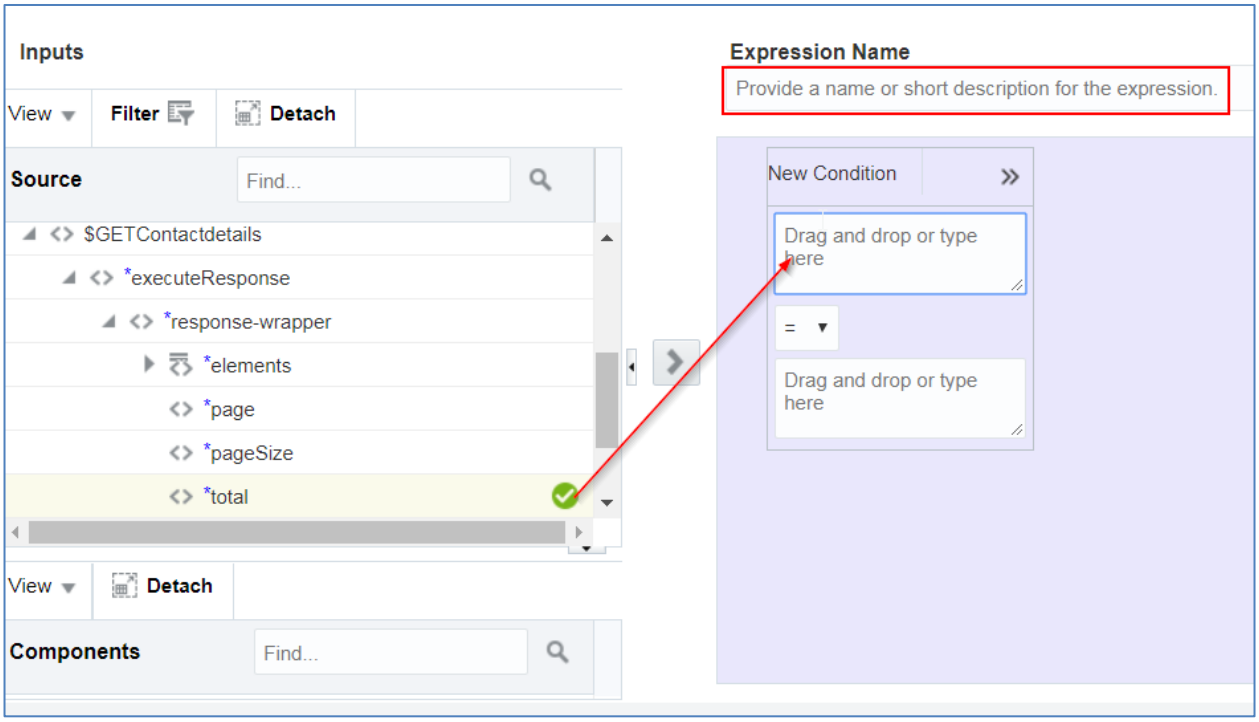
5. It should look like below



6. Click on the first “Undefined” rule to edit it.

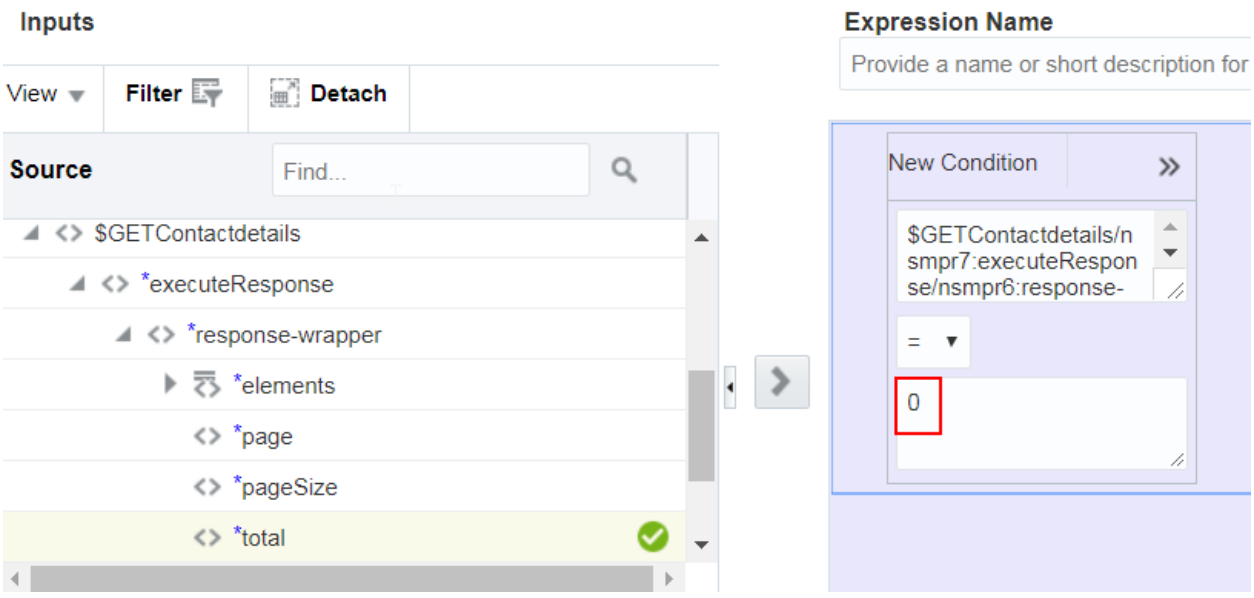


7. On the **Expression Builder Page** , Under “**Source”** drag **$GETContactdetails->executeResponse->responsewrapper->total** on to the right.

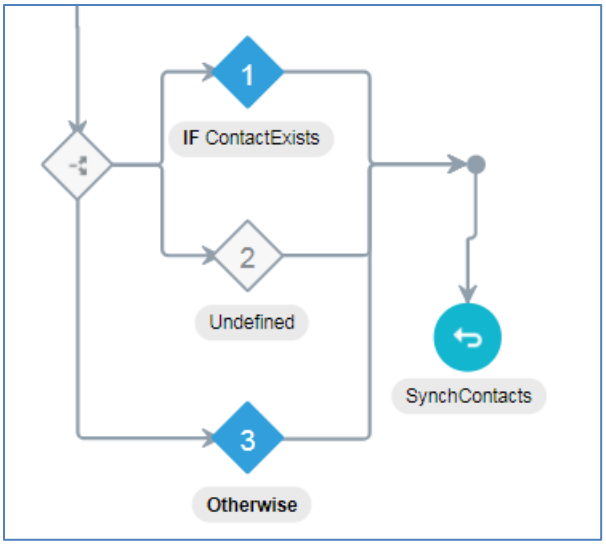


Note: (provide a name to the expression to reflect business condition. If you don't OIC will take care of it.) Lets give the name as "**ContactExists**"

8. On the bottom field of condition, enter "**0**".

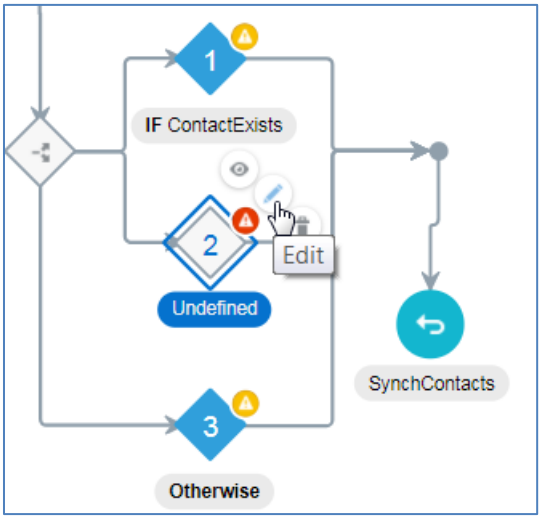


9. Click "**Validate**" and "**Close**"
10. It should look below

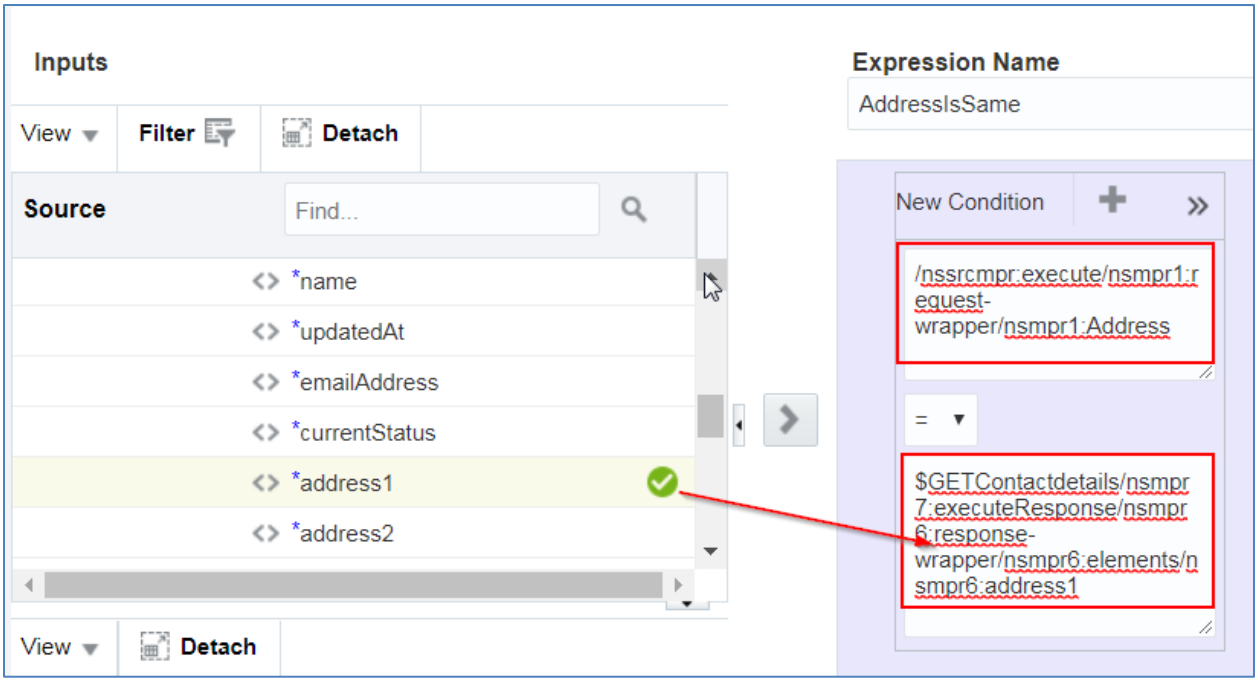


Note: Notice the name of the expression (if you haven't provided one).

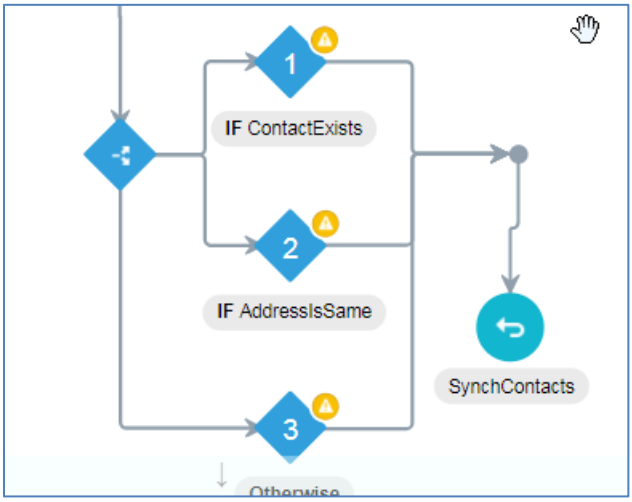
11. Now let's create 2nd rule. Click on second "**Undefined**" to edit it



12. Create a rule which checks if "**SOURCE->execute -> request-wrapper->Address**" = "**$GetContactdetails->executeResponse->response-wrapper->elements->address1"** (Drag & drop the fields into the condition box as we did in previous rule)



13. Ensure that you see the expressions as highlighted above on the right.
14. Click "**Validate**" and "**Close"**
15. Click **Save** again



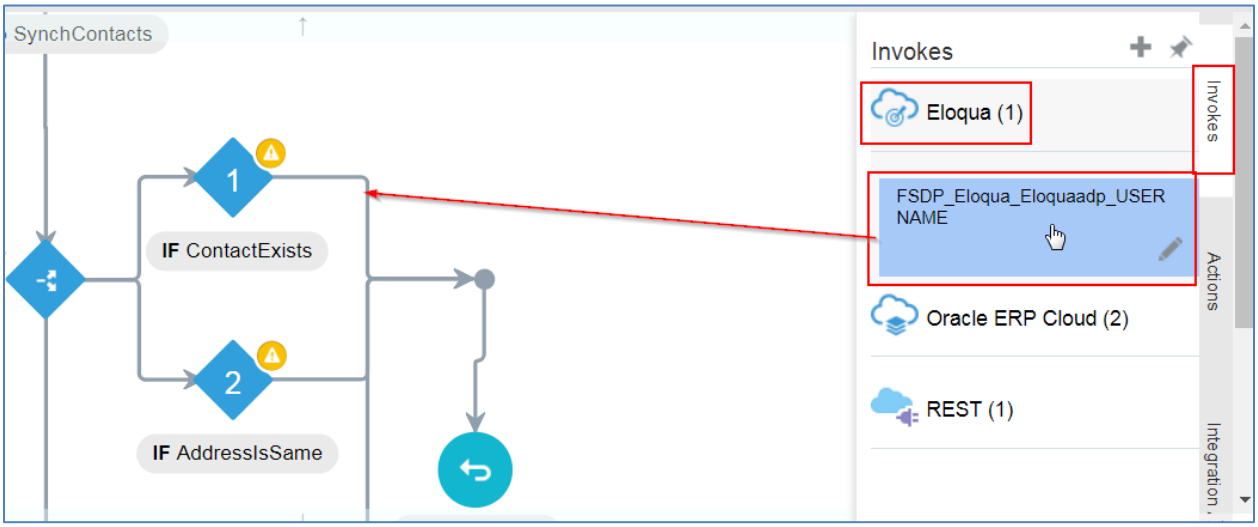
Note: No worries if you don't see the name of expression as "**AddressIsSame**" (you might see as "IF Address = address1"). We omitted that step intentionally to highlight OIC auto naming… Just proceed.

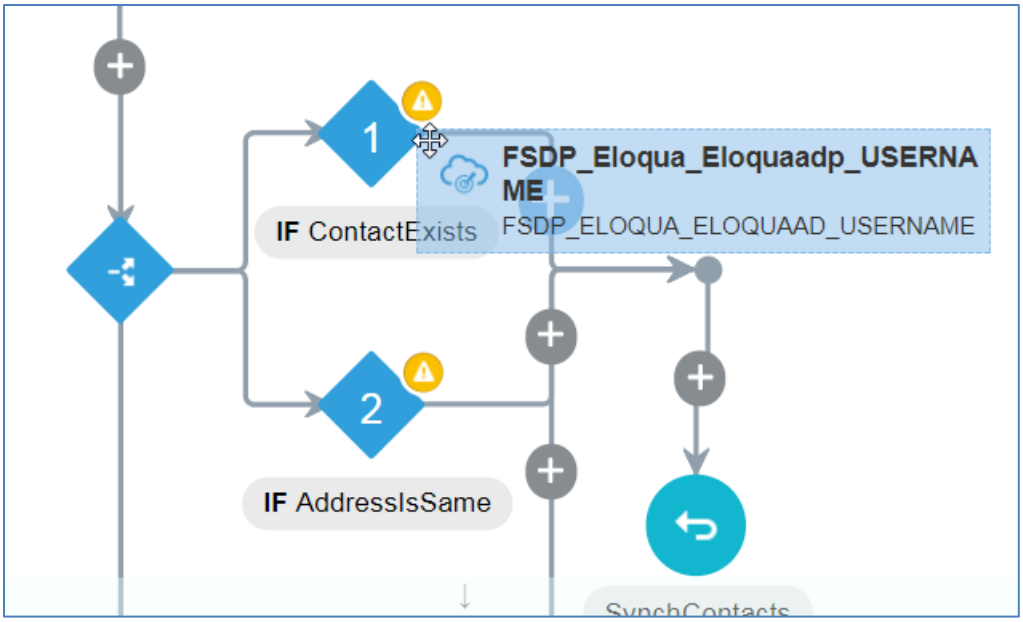
Configure endpoints based on rules

Let's further configure to invoke respective adapters to trigger Eloqua Create/Update

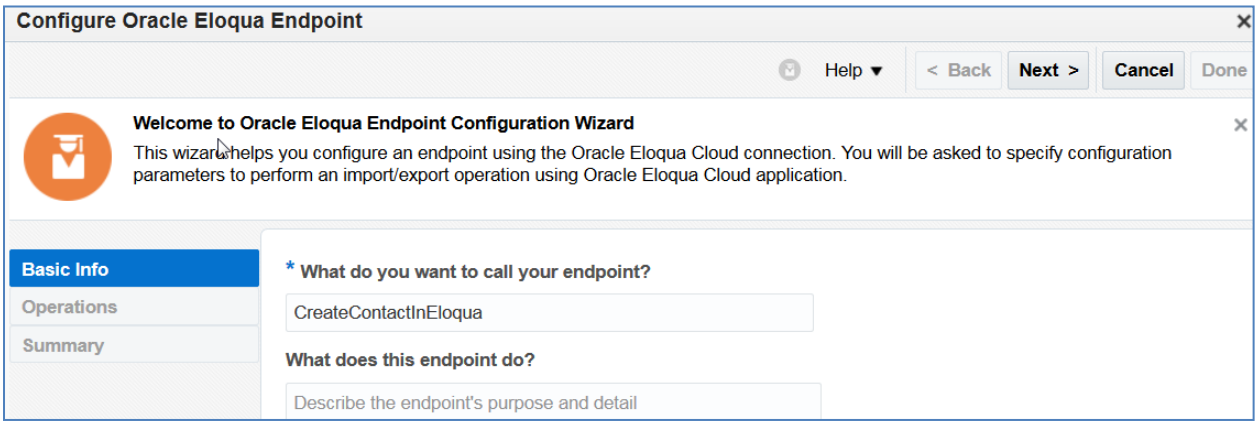
1. On the integration Orchestration designer page, from the "**Invokes**" menu on the right, Select "**Eloqua"=>** "**FSDP_Eloqua_Eloquaadp_USERNAME"** (NOT **username** but the one you created earlier), **drag&drop** it next to **first rule.**

   Note: Ensure that you drop it at the right place**. +** symbol may prop at many places but you should drop this near to the first rule.

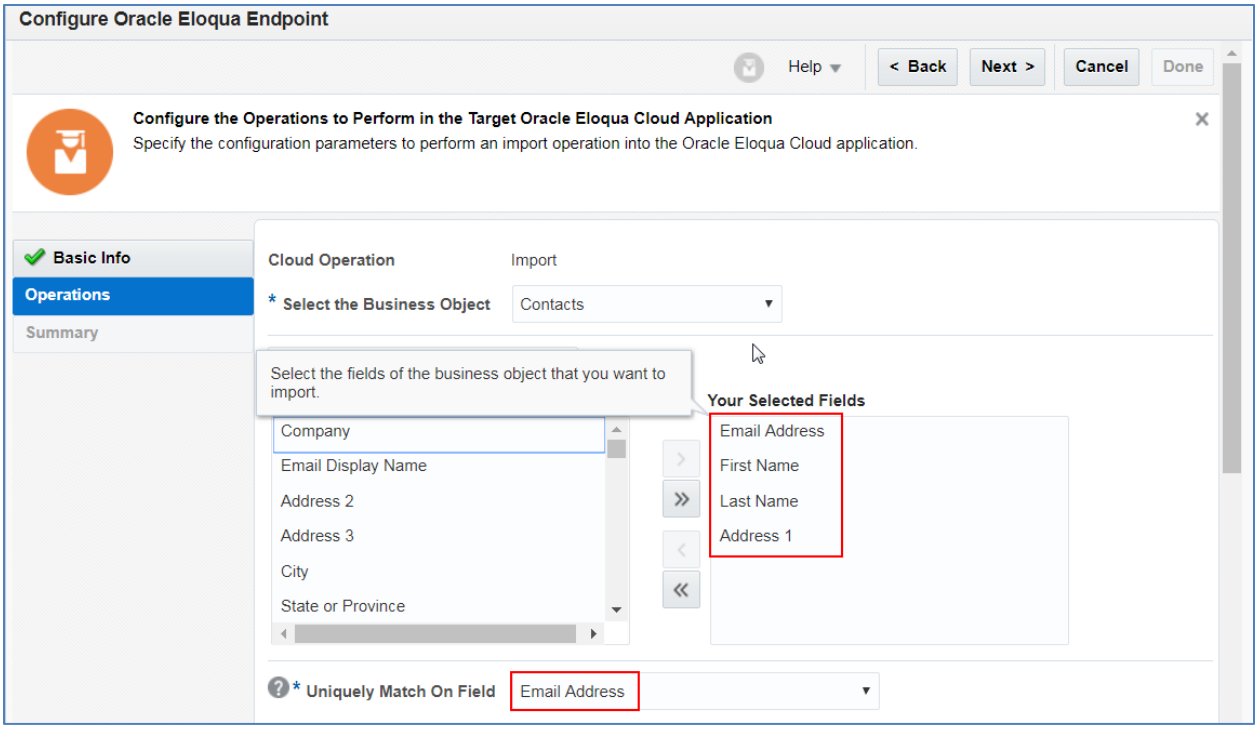




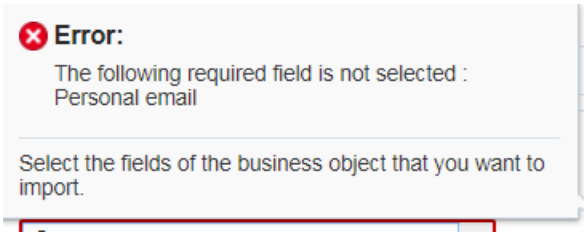
2. In the Eloqua Endpoint Configuration Wizard name it as "**CreateContactInEloqua**". Click **next**



3. Select **"Contacts"** as "**Business Object".**
4. Select **EmailAddress**, **FirstName**, **LastName** & **Address1** fields. (If any of these fields are pre-selected, you can ignore those).
5. Uniquely match with **"EmailAddress"**.

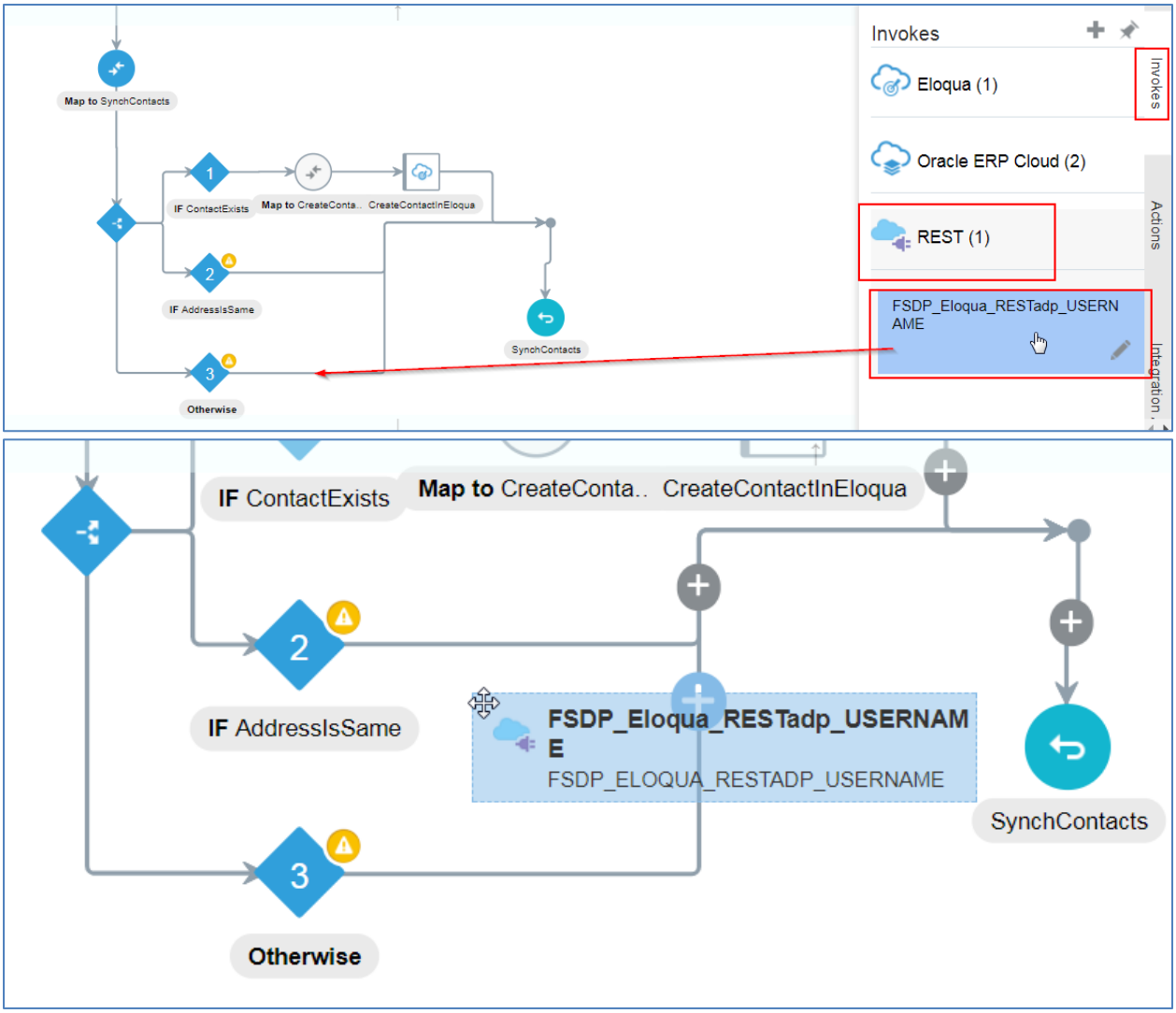


Note: You MAY see an error message similar to the one below (it may contain different field name). Just select whichever fields it complains (in this case it's "Personal email") to the mapping and move on.

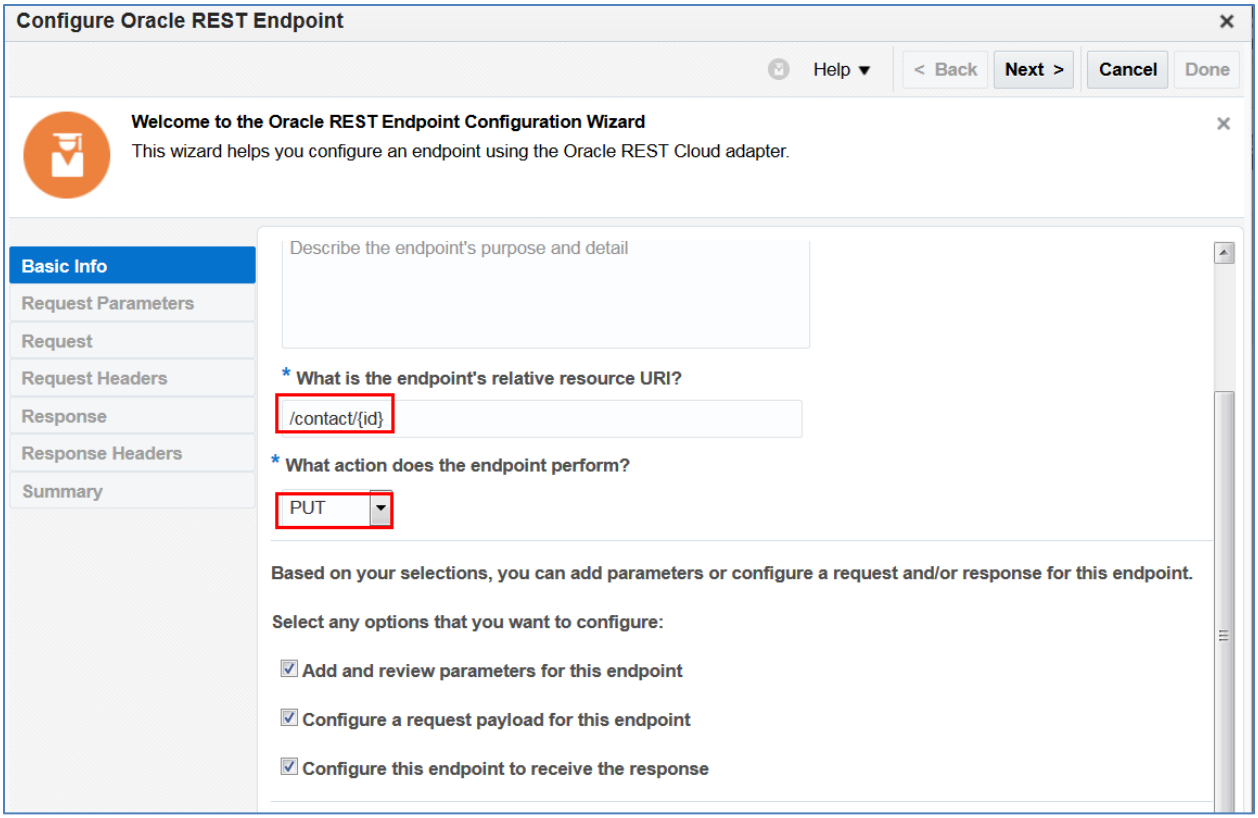


6. Click **Next** and review Summary**.** Click **Done.**

7. Now, let's configure "**OTHERWISE**" condition to update the address.
8. From the "**Invokes**" menu, choose **REST=>FSDP_Eloqua_RESTadp_USERNAME (NOT username but the one which you created earlier)** and drop it next to "**3 Otherwise**" rule

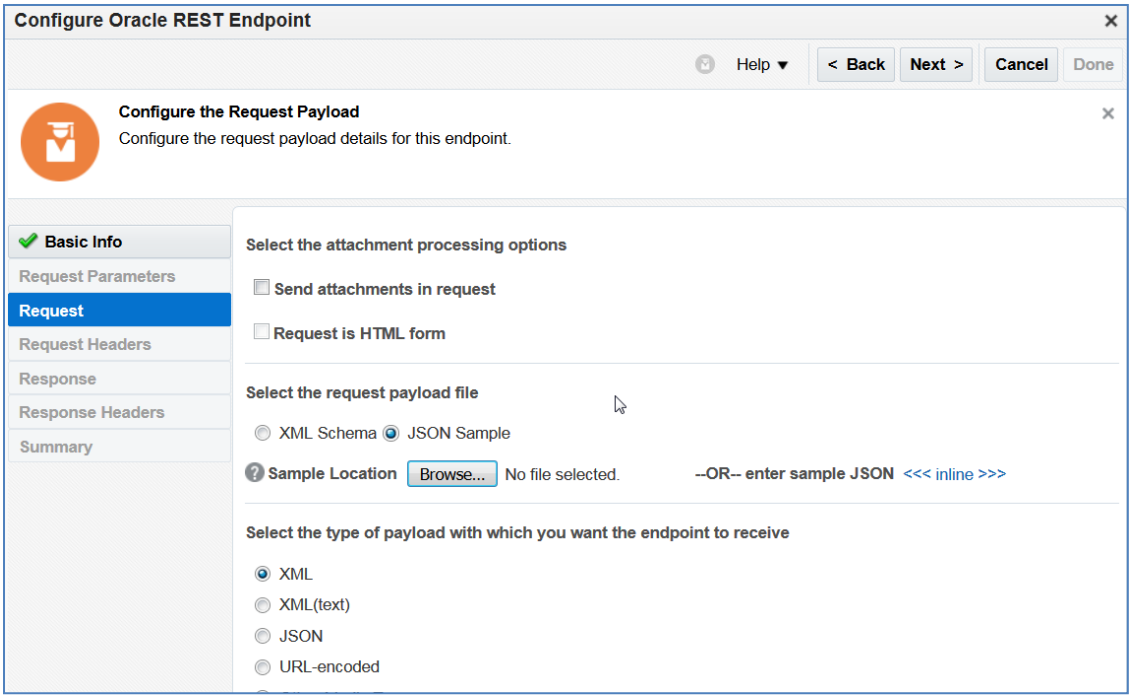


9. In the config wizard, name it as "**UpdateEloquaAddress**", use **"/contact/{id}"** as URI, select **PUT,** and select **Configure REQUEST and RESPONSE** option.



10. Click **Next.**

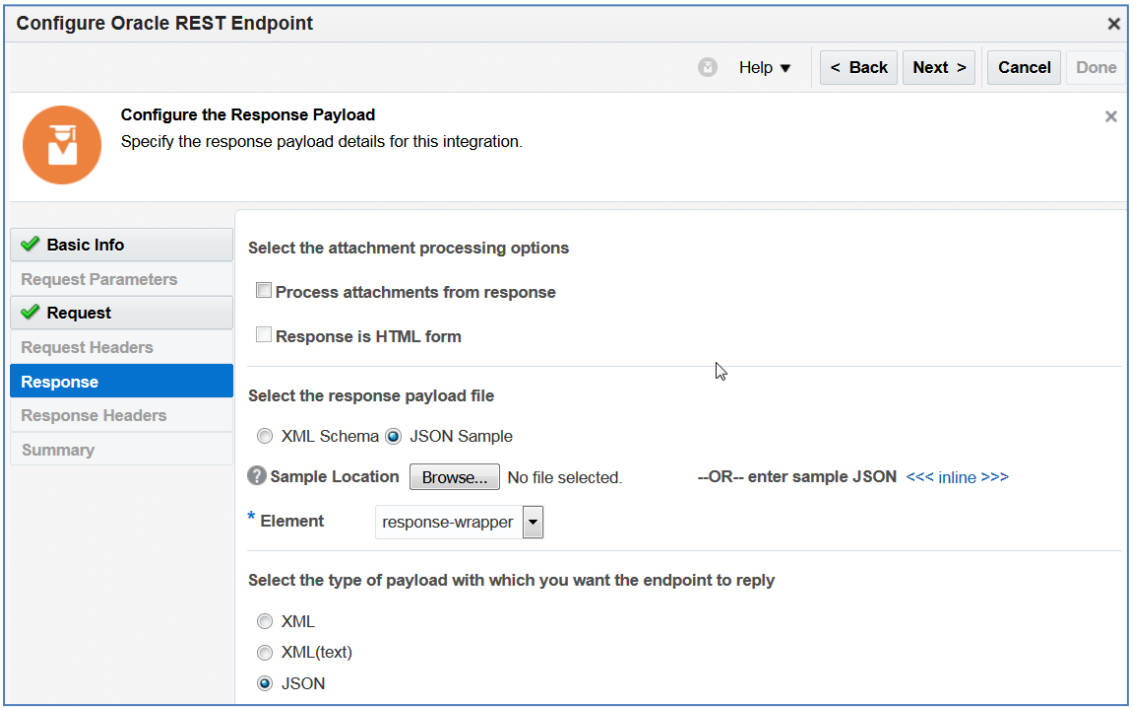
11. Click **Next** again for **Request Payload configuration**
12. Select **JSON Sample** and select the file **UpdateEloquaAddressREQUEST.JSON** (provided by the instructor).



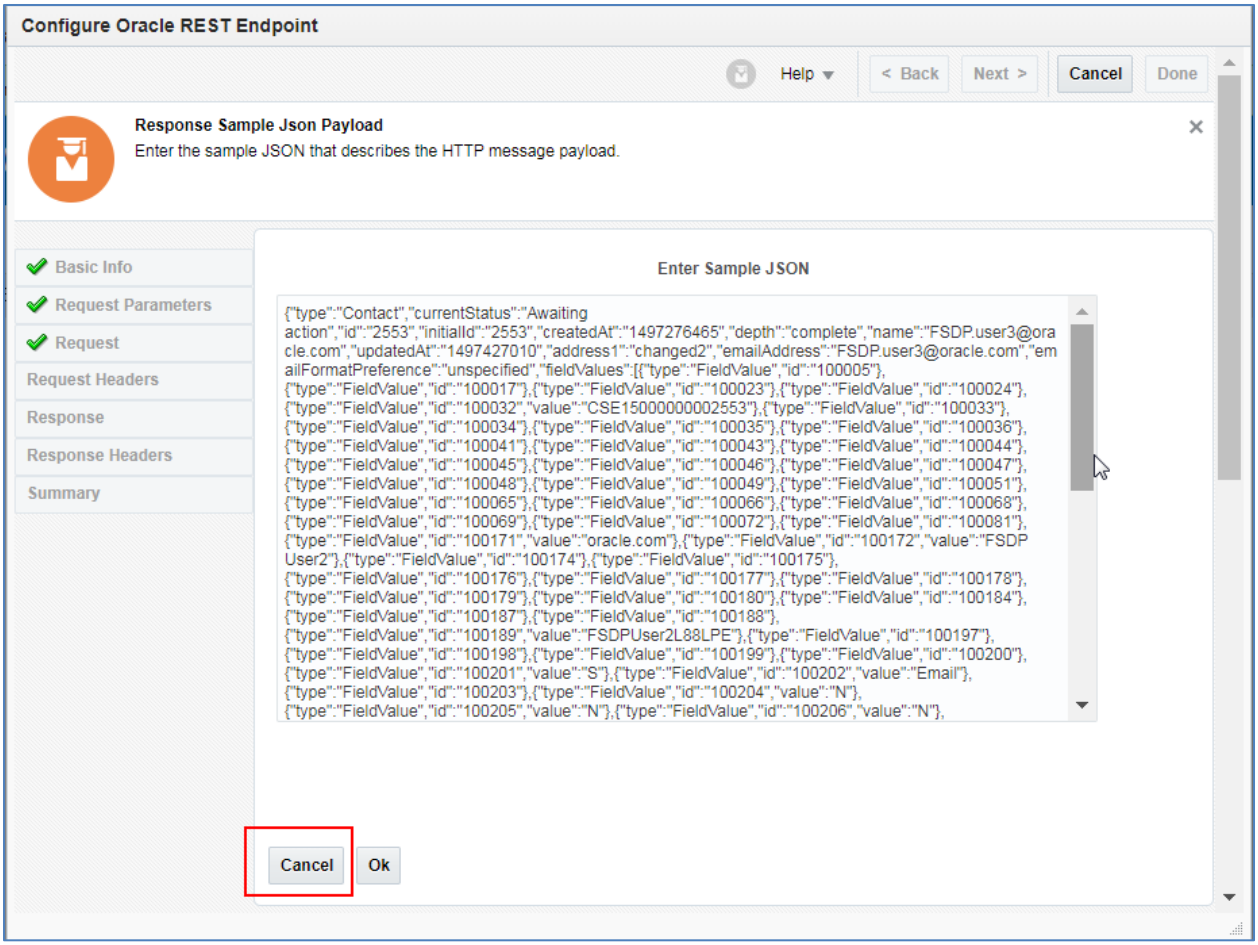
13. Click <<<inline >>> and ensure that you see the below payload. Click **cancel** on the bottom.

14. Click **Next** for **RESPONSE payload config.**

15. Select **JSON Sample** and select the file **UpdateEloquaAddressRESPONSE.JSON** (provided by the instructor).

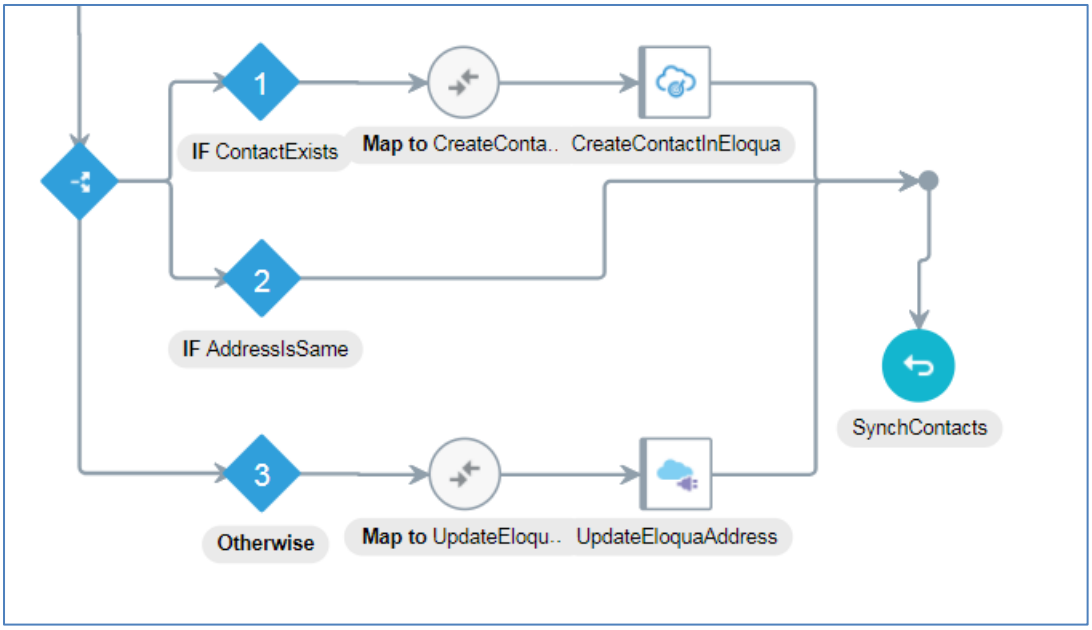


16. Click <<<inline >>> and ensure that you see the below payload. Click **cancel** on the bottom.



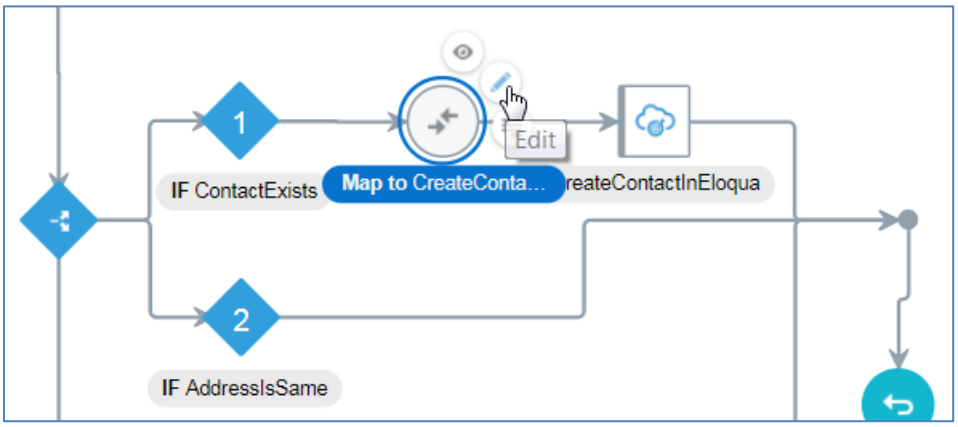
17. Click **Next** and review the summary. Click **Done**

18. **Save** the integration! Your integration should look like below.

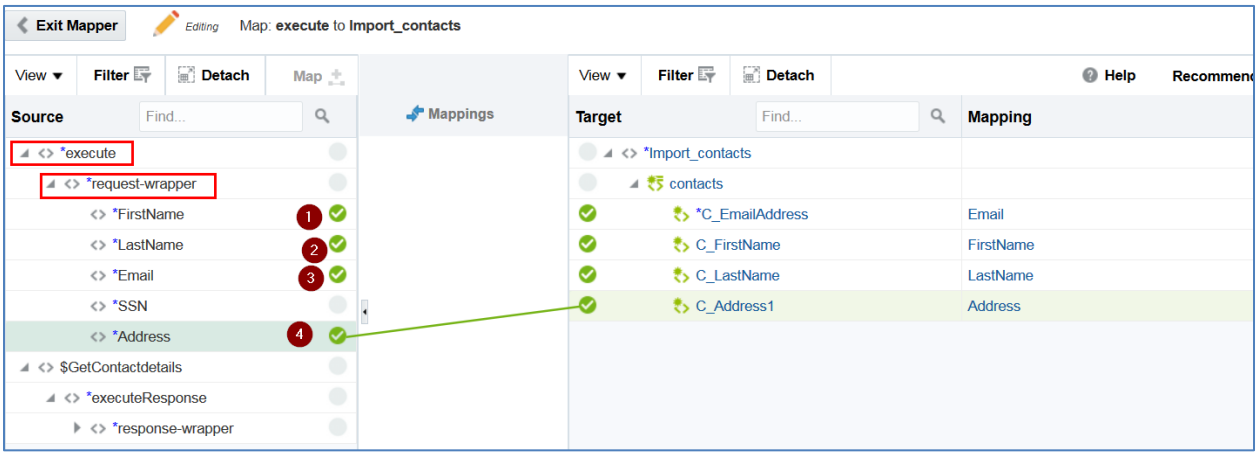

1
IF ContactExists
Map to CreateConta..
CreateContactInEloqua
2
IF AddressIsSame
SynchContacts
3
Otherwise
Map to UpdateEloqu..
UpdateEloquaAddress

Configure Mapping for Create/Update actions

1. Click on “Map to CreateContactInEloqua” map and choose EDIT.



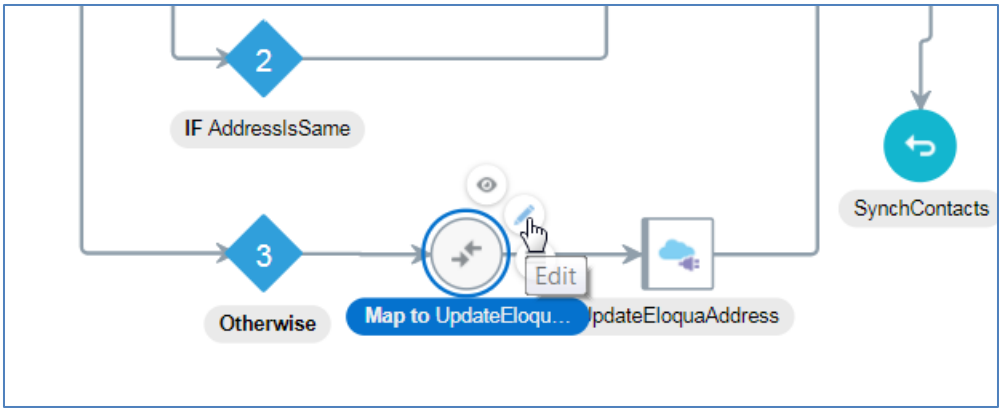
2. Map fields as below from execute.



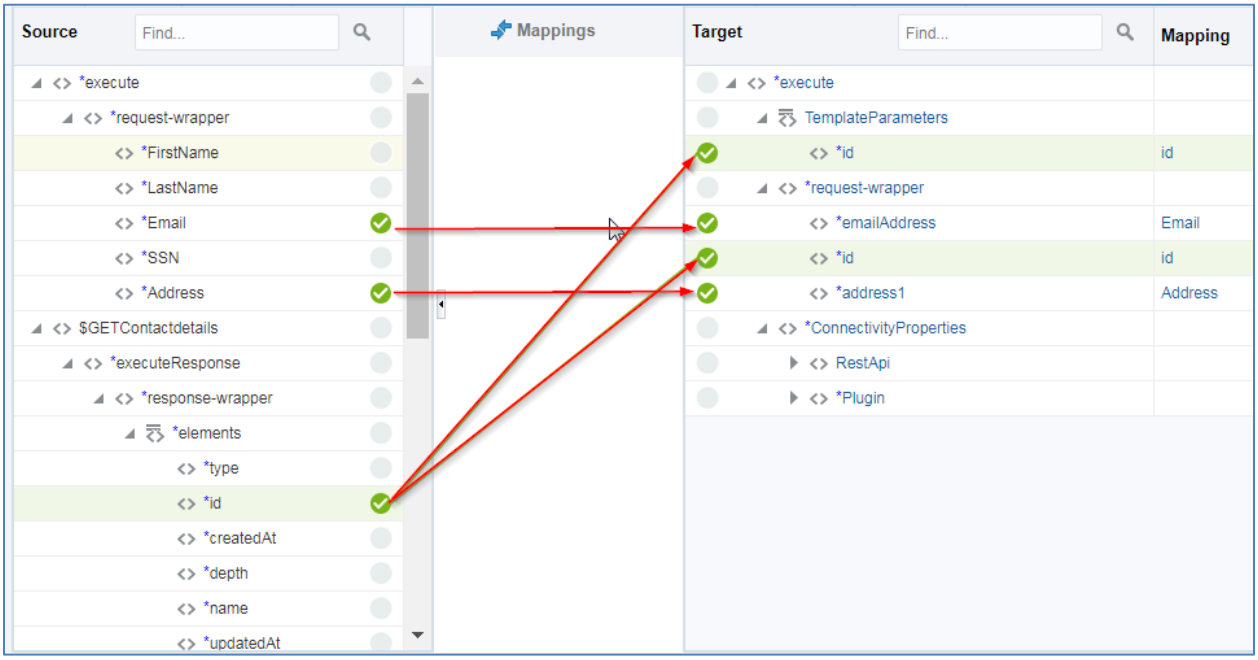
Note: Ignore any other fields.

3. Click **Validate** and **Close**

4. Map the update process
   Click “Map to **UpdateEloquaAddress**” mapping to edit it.

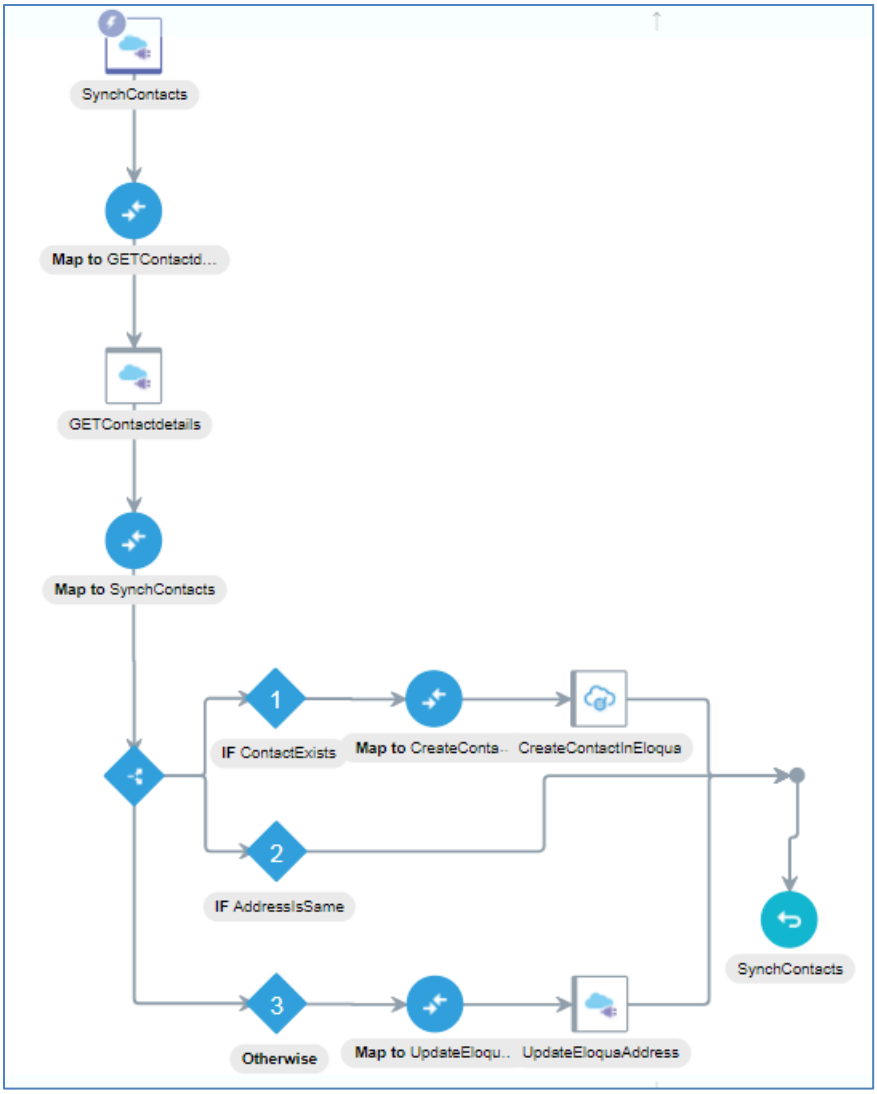


5. And do the mapping as shown below.



Note: Please be mindful that you are selecting fields from 2 DIFFERENT sources this time. One from **execute**->**request**-wrapper and other from $**GetContactdetails**->**response**-wrapper. **This way we are using the result data of a previous endpoint request, in the next endpoint's request URI.**

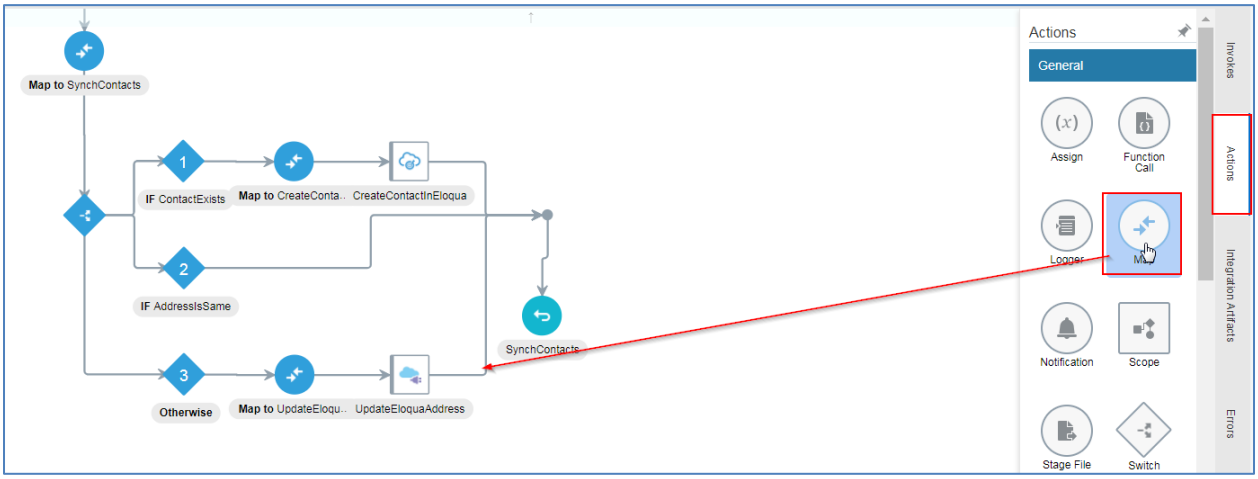
6. **Validate** and **Close**
7. **Save** the integration.
   The Integration should look like below

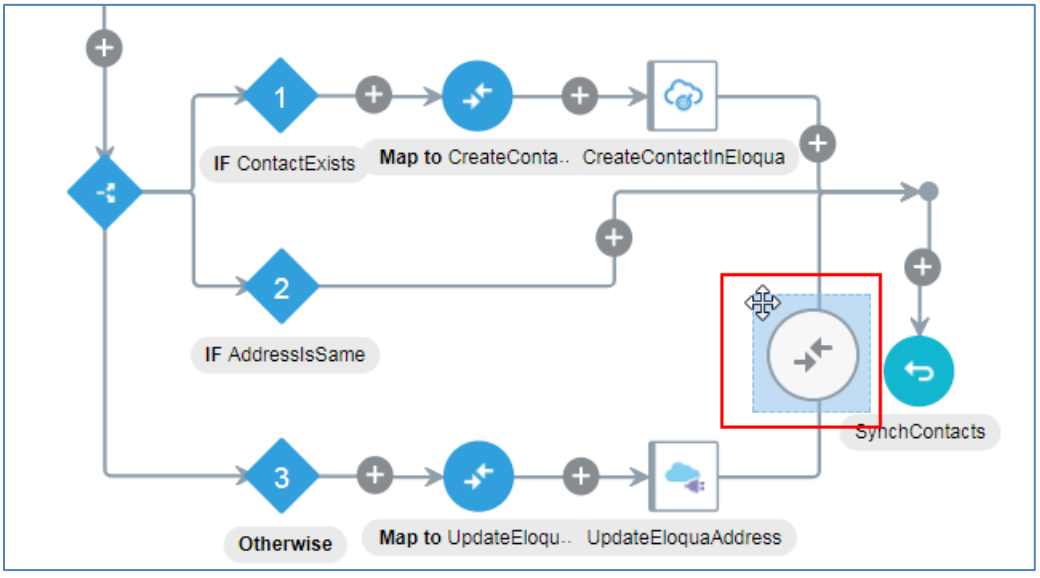

SynchContacts
Map to GETContactd...
GETContactdetails
Map to SynchContacts
1
IF ContactExists
Map to CreateConta...
CreateContactInEloqua
2
IF AddressIsSame
SynchContacts
3
Otherwise
Map to UpdateEloqu...
UpdateEloquaAddress

Add mapping to get update results

Let's add a mapping so that we can map the response for update activity from Eloqua with the final return.

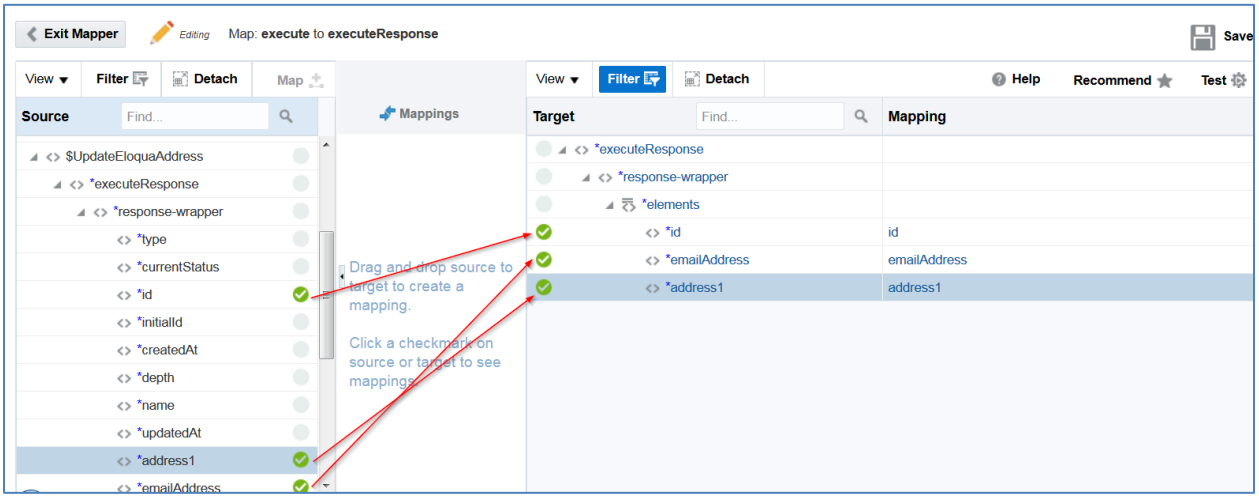
1. From the **Actions** menu on left drag and drop a **map** after **UpdateEloquaAddress** as shown below.

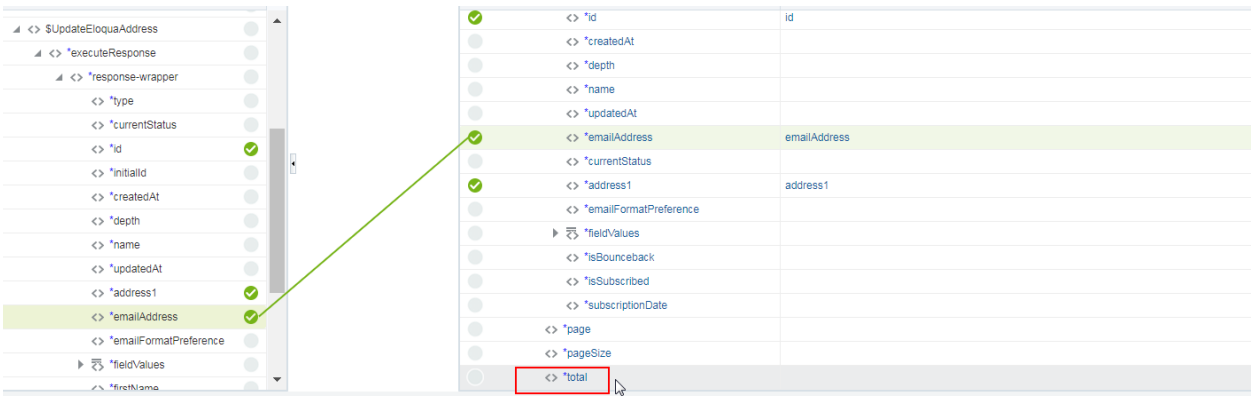


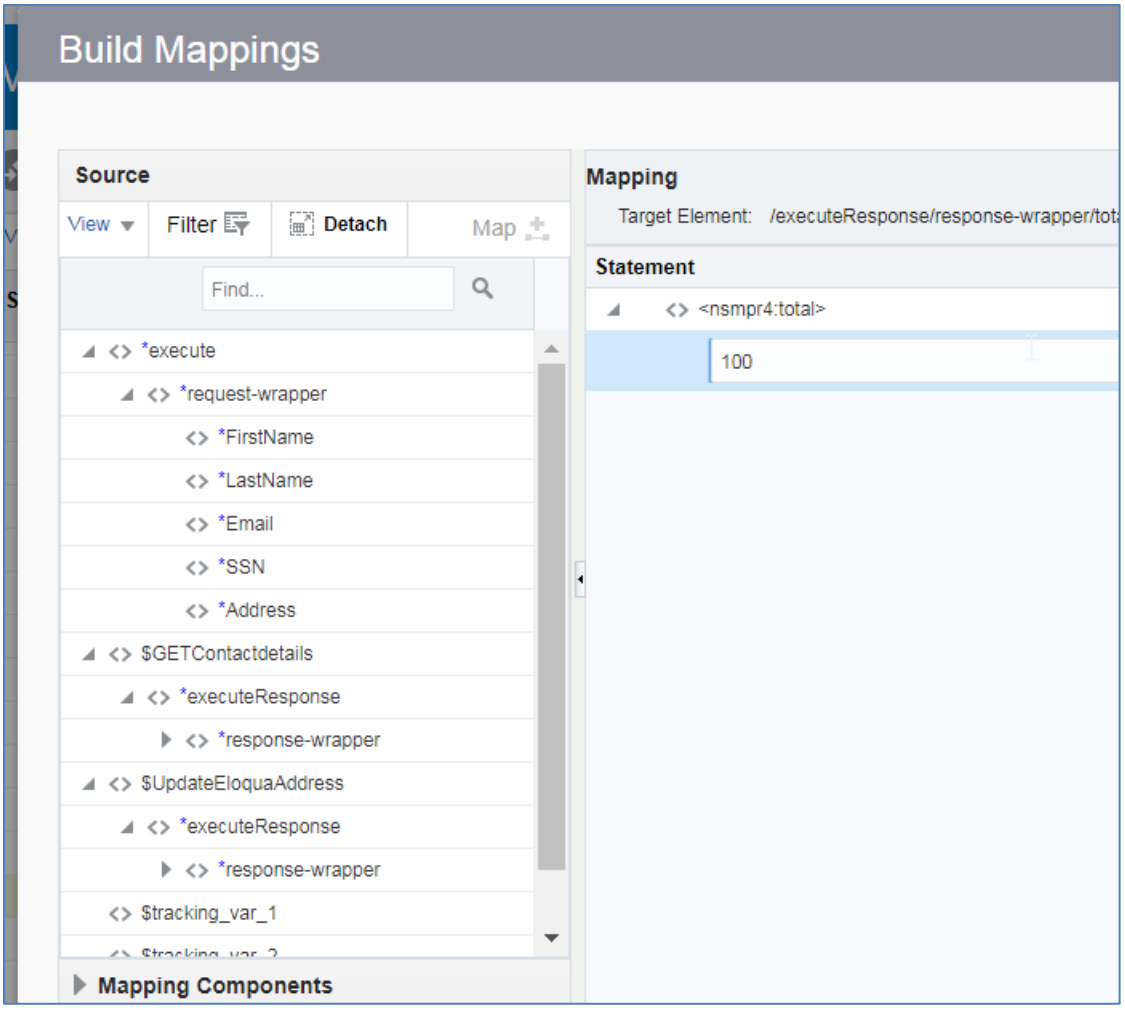


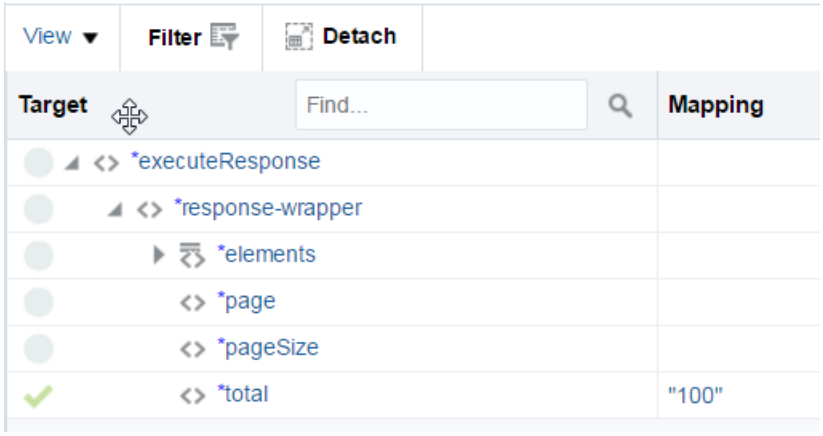
2. Do the mapping as shown below.

Source	Target
/$UpdateEloquaAddress/executeResponse/response-wrapper/emailAddress	/executeResponse/response-wrapper/elements/emailAddress
/$UpdateEloquaAddress/executeResponse/response-wrapper/address1	/executeResponse/response-wrapper/elements/address1
/$UpdateEloquaAddress/executeResponse/response-wrapper/id	/executeResponse/response-wrapper/elements/id
	/executeResponse/response-wrapper/total Type "**100**" at the "Drag and Drop" after clicking the total (on **Target not Source**)

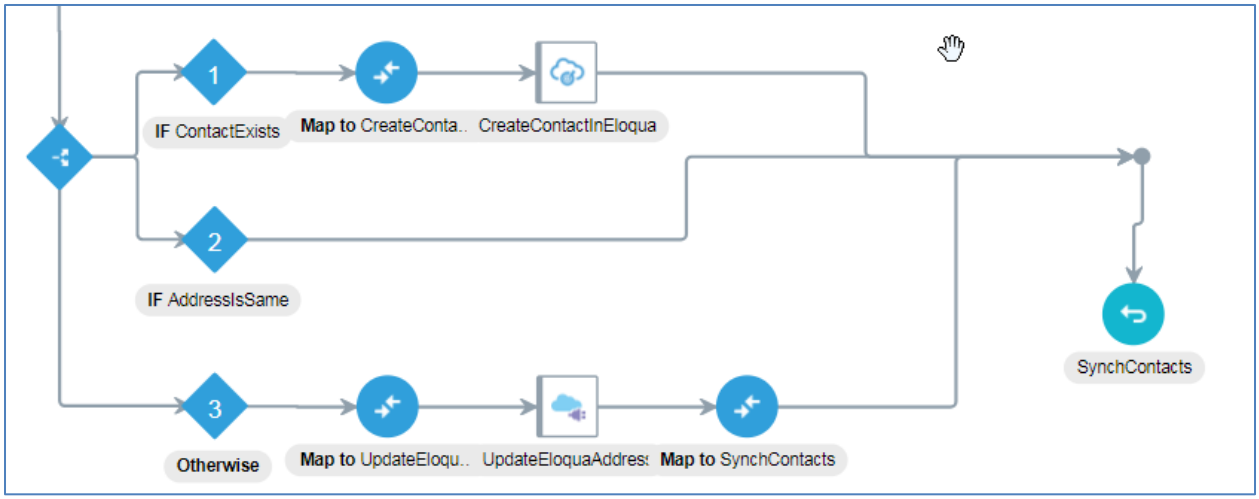








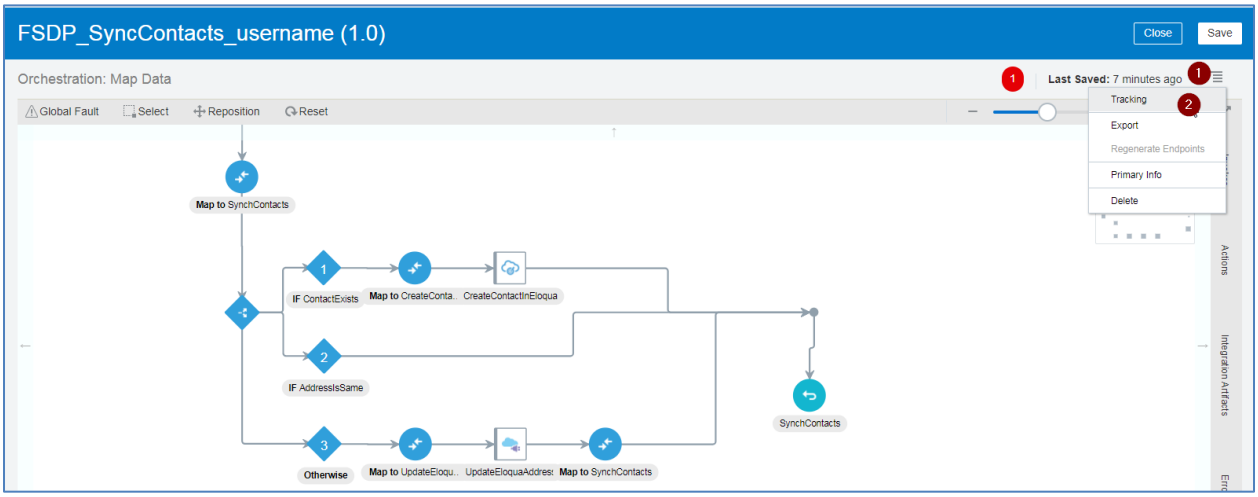
3. Click "**Validate**" and "**Close**". Now, Integration should look like below.


1
IF ContactExists
Map to CreateConta..
CreateContactInEloqua
2
IF AddressIsSame
SynchContacts
3
Otherwise
Map to UpdateEloqu..
UpdateEloquaAddres
Map to SynchContacts

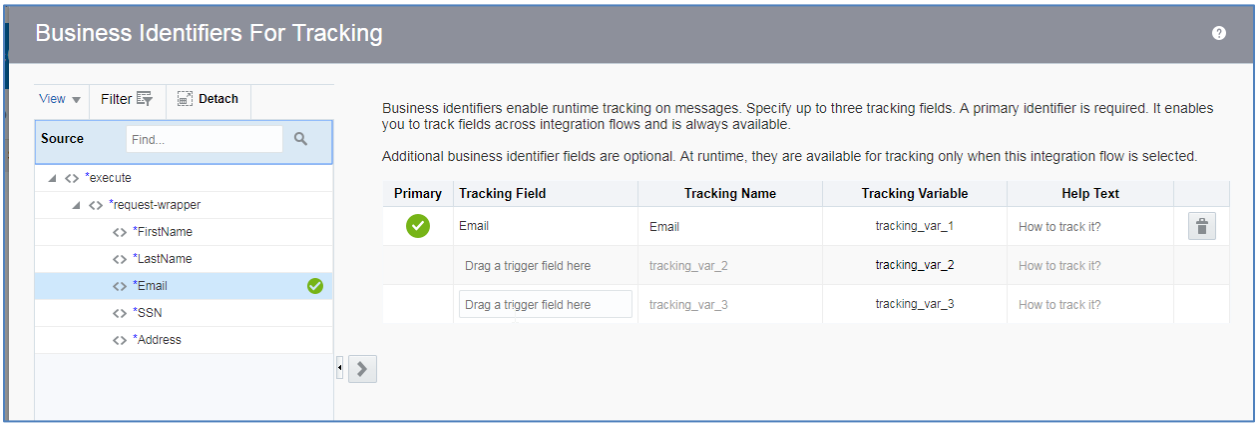
Click "**Save**".

Add Tracking

1. On the Top-right of designer, Click "**Tracking**" button on Actions menu.

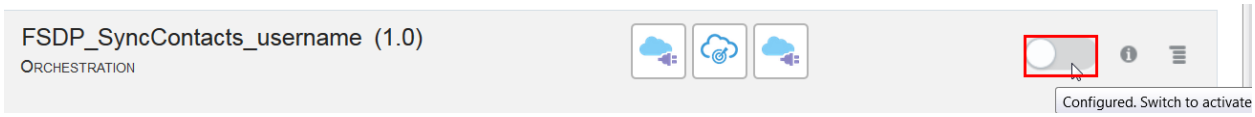


2. Drag **Email** to the Right and drop it on first **Tracking Field**.

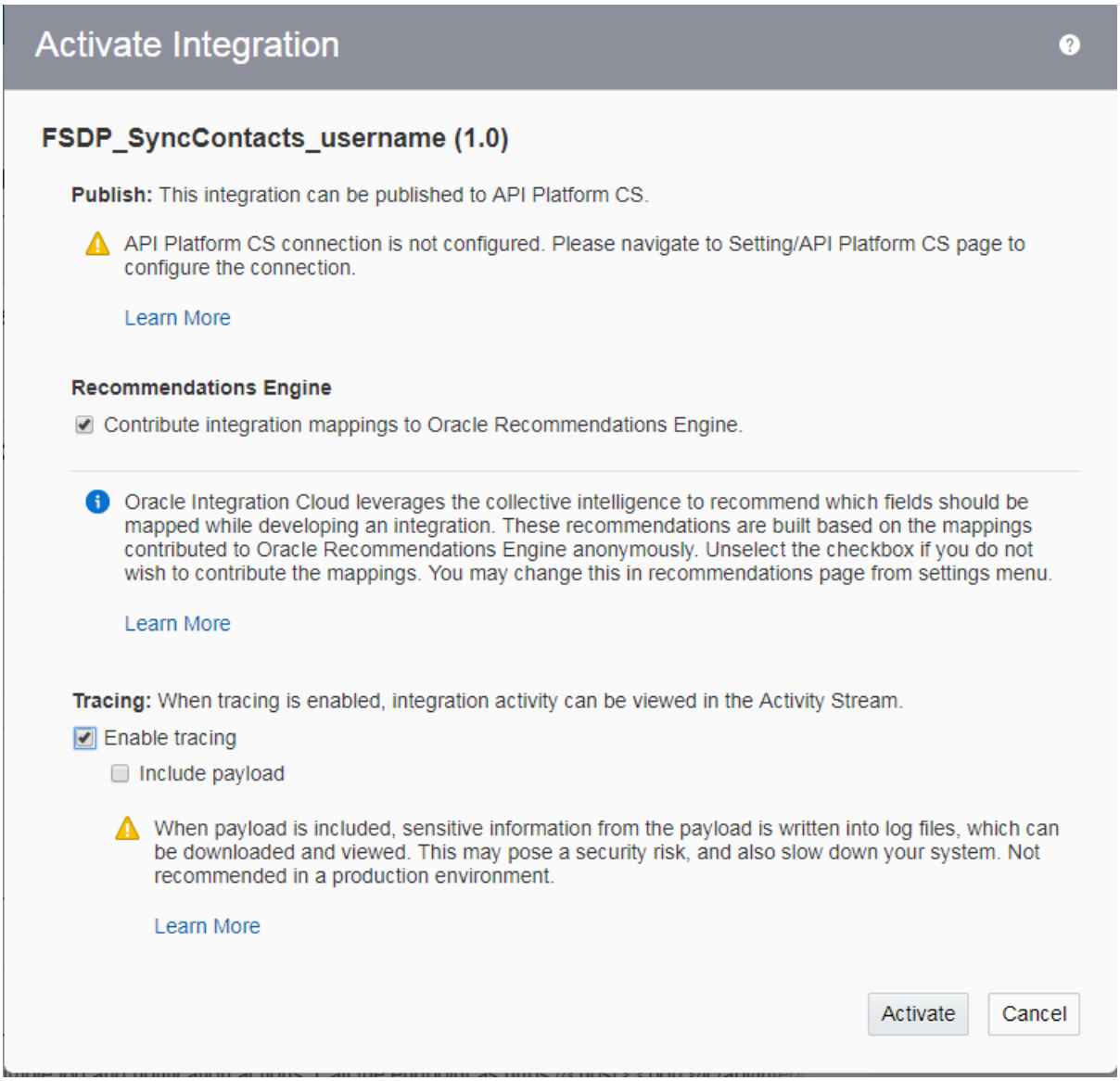


3. Click "**Save" (at the bottom)**
4. **"Save"** the integration (one last time, hopefully ☺)
5. Click "**Close" !**

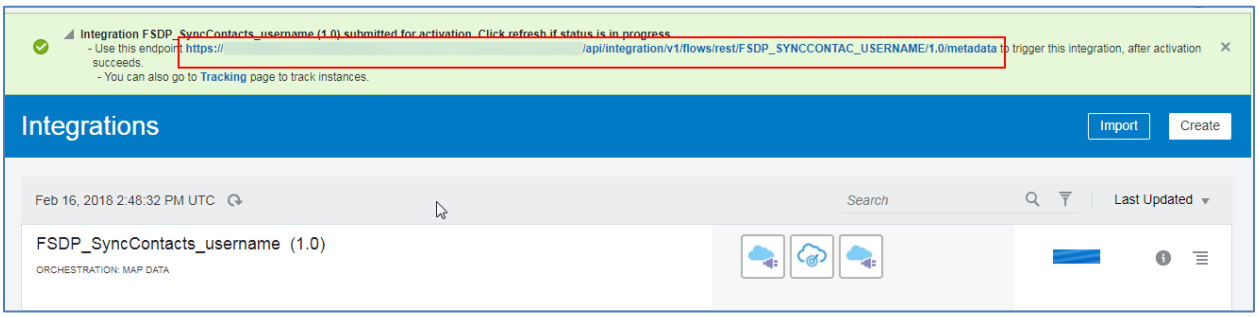
6. Click on the switch to **Activate** the integration.



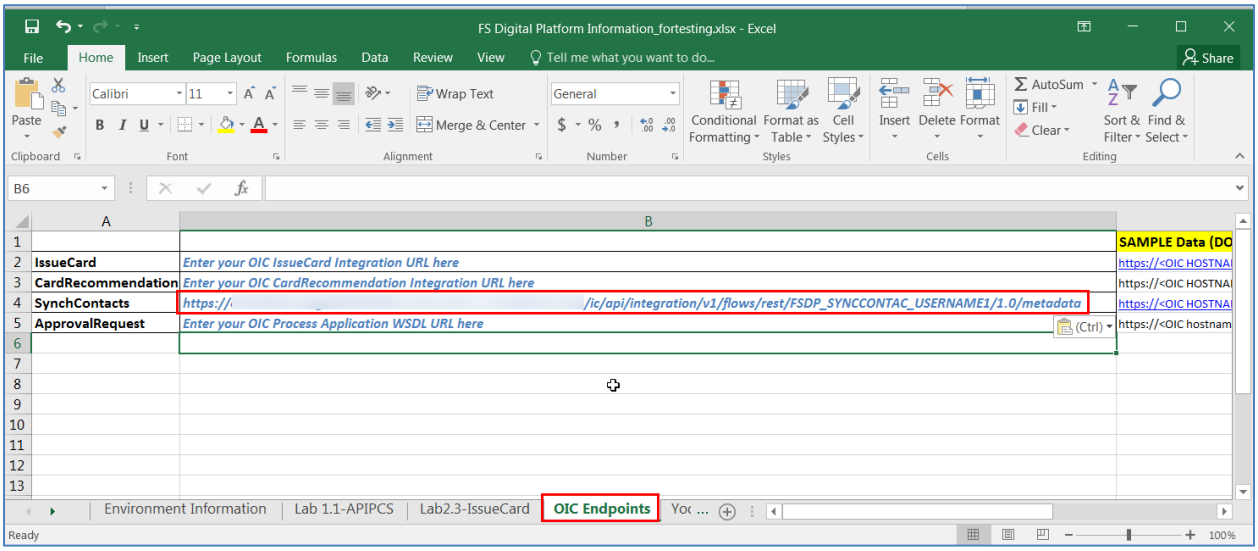
7. **Click** the checkbox to enable tracing and click "**Activate**".



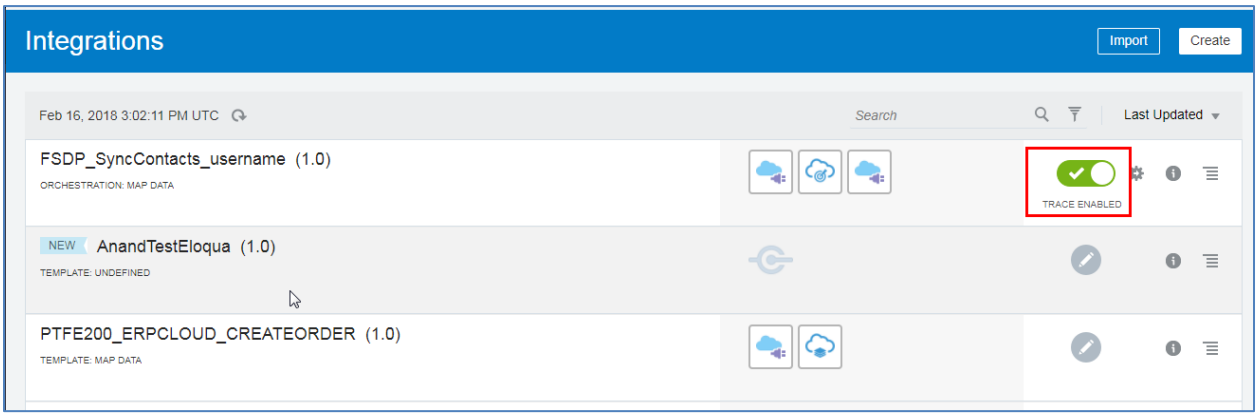
8. You may see the below message.



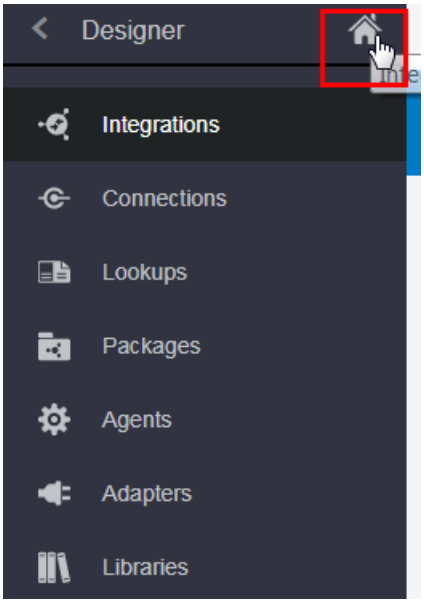
9. Make a note of the integration URL displayed on top to the **SynchContacts** field in **"FS Digital Platform.xls"->"OIC Endpoints"** sheet.

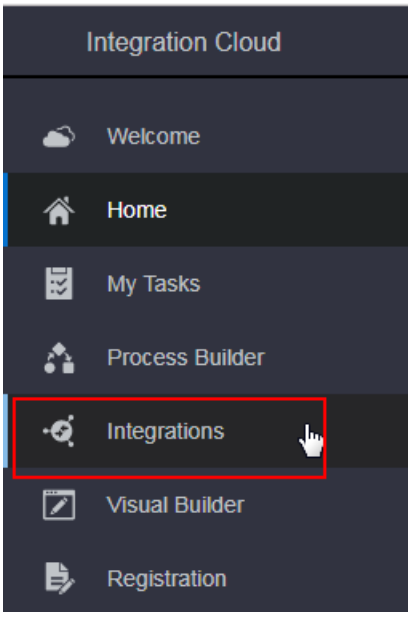


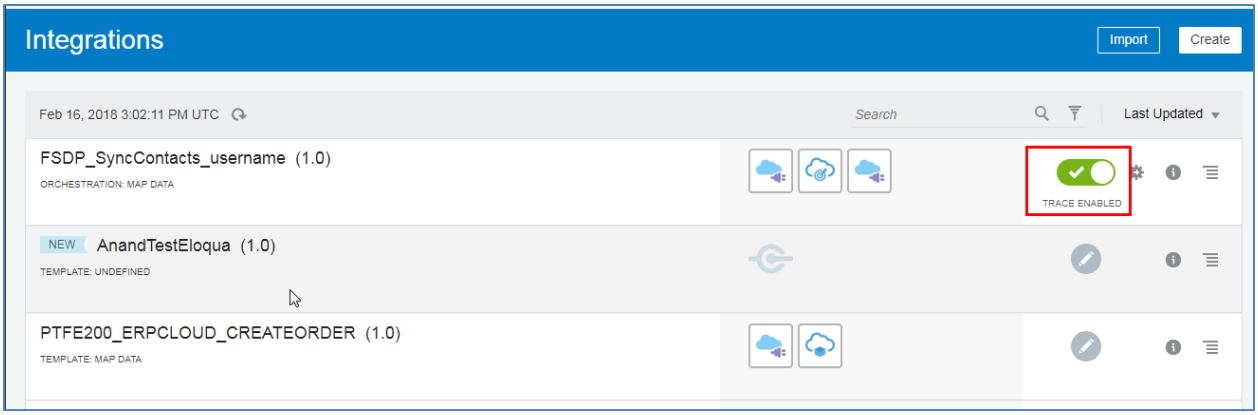
10. Wait until you see it activated.



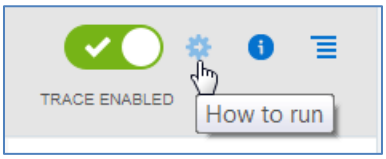
11. If you don't see it activated, refresh the page.
12. Still don't see it? You may have to go back to Integration home and come back to see the Activation status. Only then, follow the steps below.

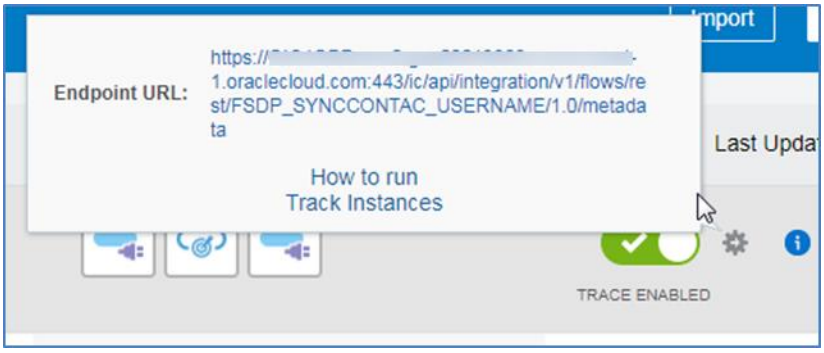

Designer
Integrations
Connections
Lookups
Packages
Agents
Adapters
Libraries


Integration Cloud
Welcome
Home
My Tasks
Process Builder
Integrations
Visual Builder
Registration



13. In case you missed to note down the Integration URL, you can get it later as well using the info button of the integration.





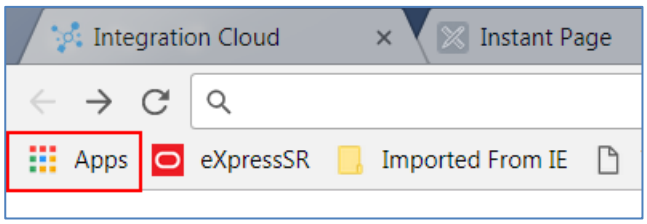
Testing

Create and configure Synch Contacts Integration

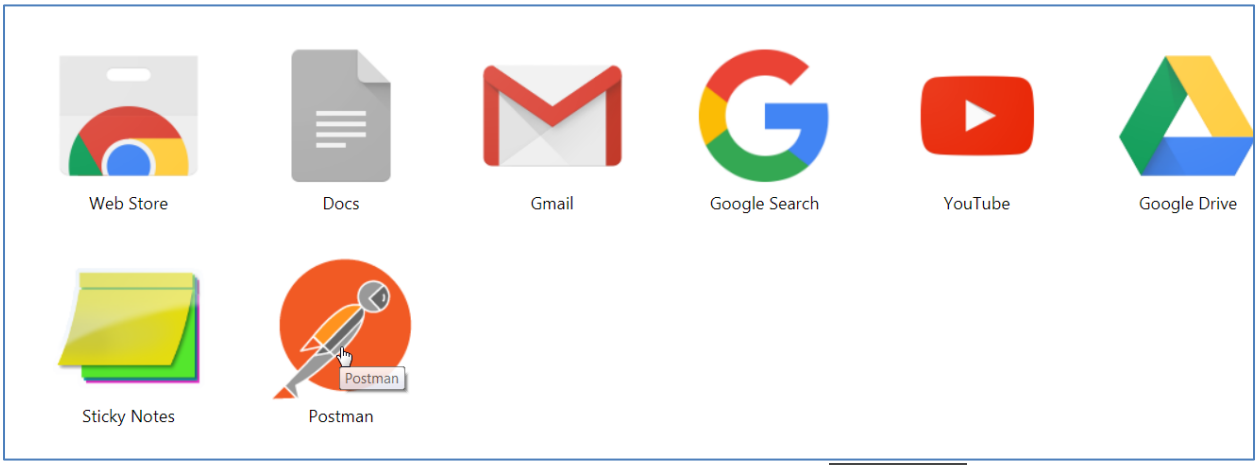
While this integration will be called from the OIC process (which you would be doing next), it is essential that we test this using some REST client. For this lab, we have used **POSTMAN extension in Chrome**.

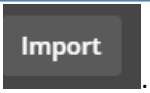
You may use POSTMAN standalone or choose your own… Below steps are for POSTMAN Chrome extension.

1. Open Chrome browser. Click on **Apps**.

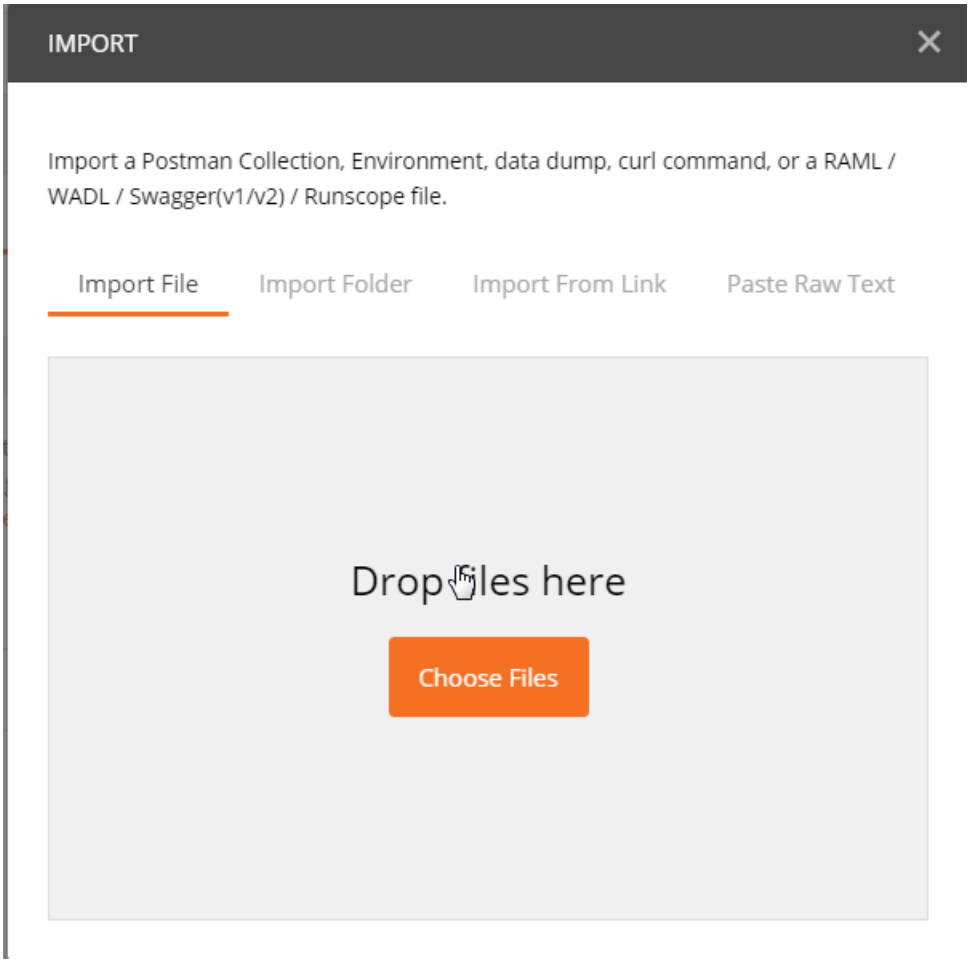


2. Click Postman extension. (If you don't see one here, you don't have Postman. Download it from google store.) Just google "Postman extension for Chrome browser" and add it to your extension.



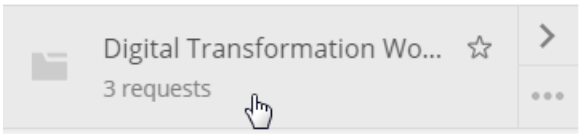
3. Import the Postman collection provided by the instructor for testing. 

..



4. Choose "Digital Transformation Workshop.postman_collection.json" file shared by instructor.

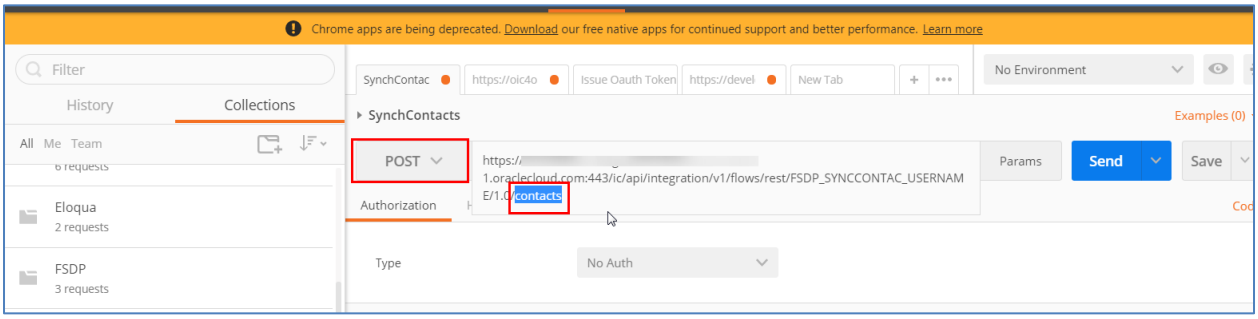
5. Click on "Digital Transformation Workshop" collection and click **Lab2.1-SYNCCONTACTS** collection



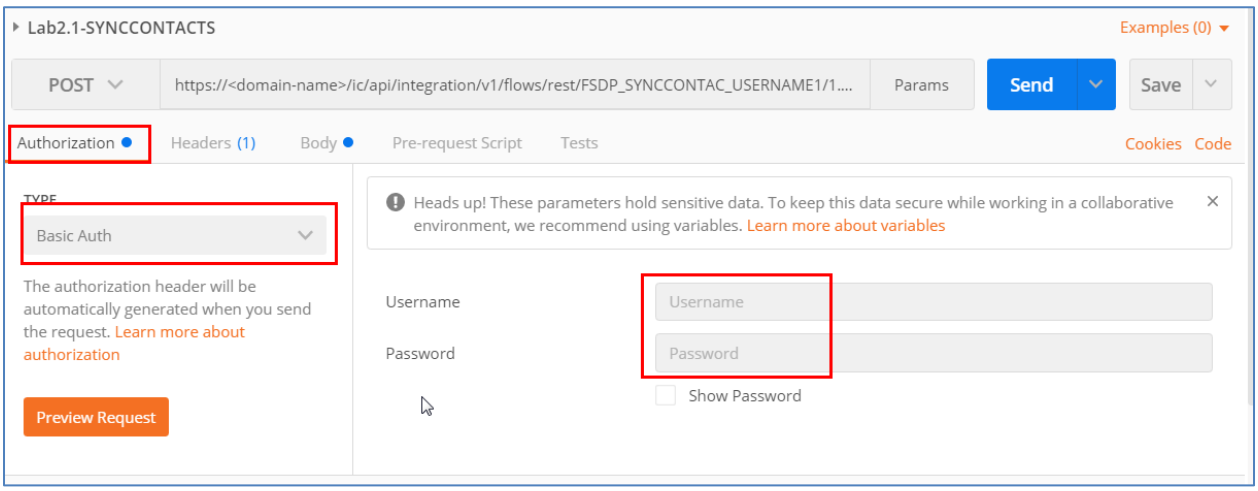
6. Copy the **SynchContacts** integration URL from **"FS Digital Platform.xls"->"OIC Endpoints"** https:// <domain-name>/ic/api/integration/v1/flows/rest/FSDP_SYNCCONTAC_USERNAME/1.0/metadata.

   Replace the /metadata with /contacts in the URL

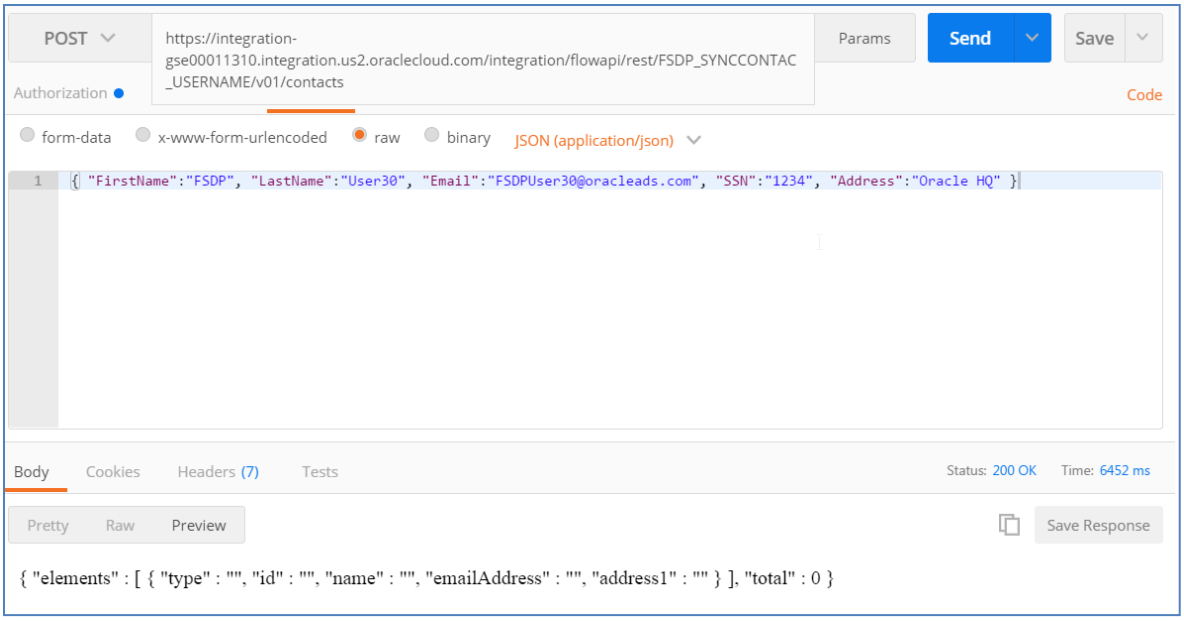
Example: https:// <domain-name>/ic/api/integration/v1/flows/rest/FSDP_SYNCCONTAC_USERNAME/1.0/contacts.



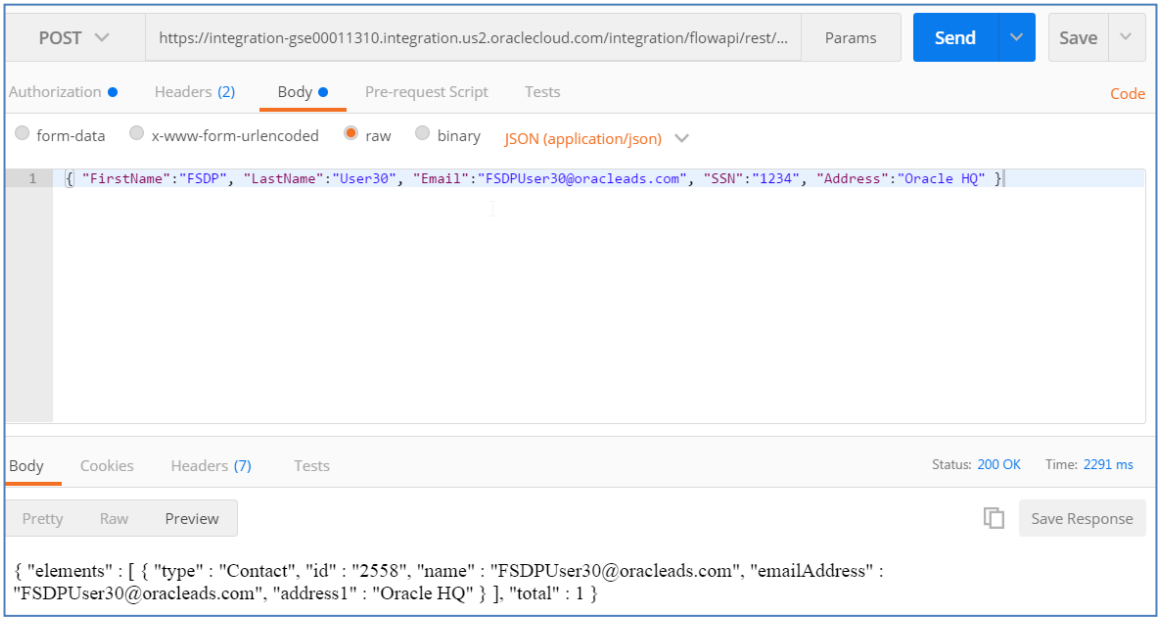
7. Update the credentials under "**Authorization**" tab using the **OIC** credentials given in the spreadsheet. Choose "**Basic Auth**" type



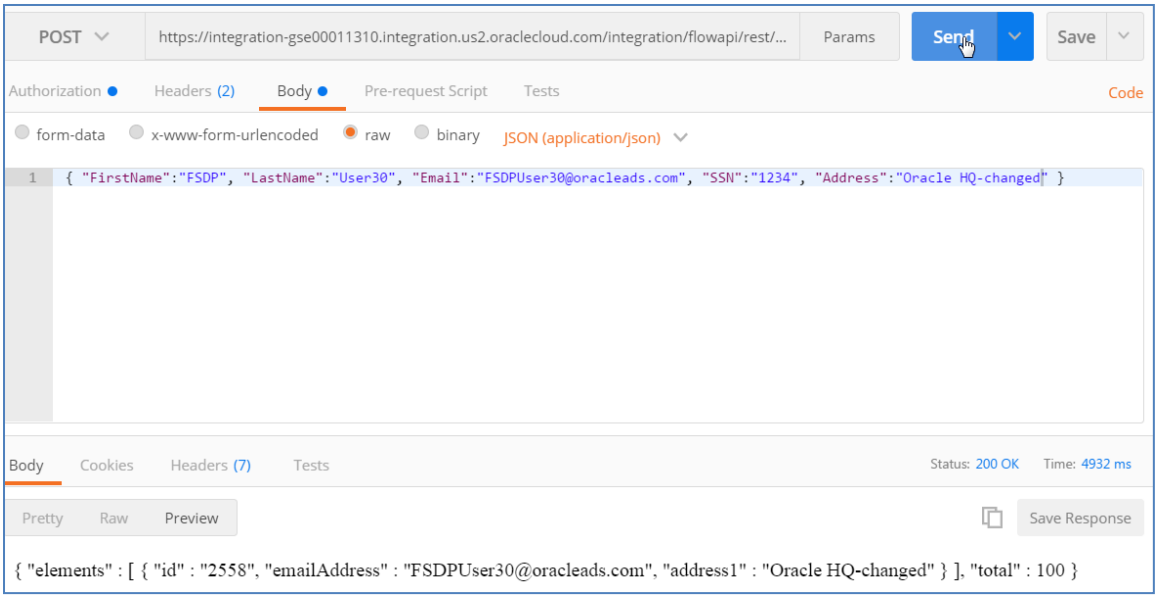
8. Click "**Body"** and verify the details. Enter the below if required.
   { "FirstName":"FSDP", "LastName":"User2", "Email":"FSDP.user30@oracleads.com", "SSN":"1234", "Address":"Oracle HQ" }
   Note: Feel free to use your own choice of email address. Since Email is the unique field in Eloqua, you cannot have 2 records with same email.
9. Click "**Update Request**".
10. Now click on **SEND** to send the request. After waiting for some time you should see the response like below at the bottom of your POSTMAN window.
    You should see total returned as "0". This implies that no record exists before and hence a creation.



11. Keep the body same and click “SEND” again -which means you are querying for the user (using GET request).
    You should see a total as “1” and the user data returned.



12. Now modify the address (try whatever you want). You should see the modified data returned from Eloqua with a response code “100”.



13. You may try different modifications and see how the integration returns. You would see that NO modifications apart from the address field initiate an UPDATE to Eloqua. The integration is configured that way.