

## **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

<b>Introduction .....</b>	<b>3</b>
Storyline.....	3
Architecture .....	5
Products used & Versions .....	5
<b>Lab Instructions (PLEASE READ BEFORE PROCEEDING):.....</b>	<b>5</b>
<b>OIC Integration with Marketing Cloud for Contact Synch.....</b>	<b>6</b>
Environment Configuration .....	6
Create OIC Connections .....	7
Create REST Adapter based Connection .....	7
Create Eloqua Adapter based Connection.....	12
Create OIC – Eloqua Integration .....	15
Configure and Map Endpoints .....	15
Create and configure SynchContacts Integration .....	15
Configure GETContact.....	23
Map GETContact.....	28
Map SynchContacts endpoint .....	33
Configure Swtiching Rules .....	35
Configure endpoints based on rules.....	42
Configure Mapping for Create/Update actions .....	51
Add mapping to get update results.....	54
Add Tracking.....	58
Testing.....	63
Create and configure Synch Contacts Integration.....	63

# 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.

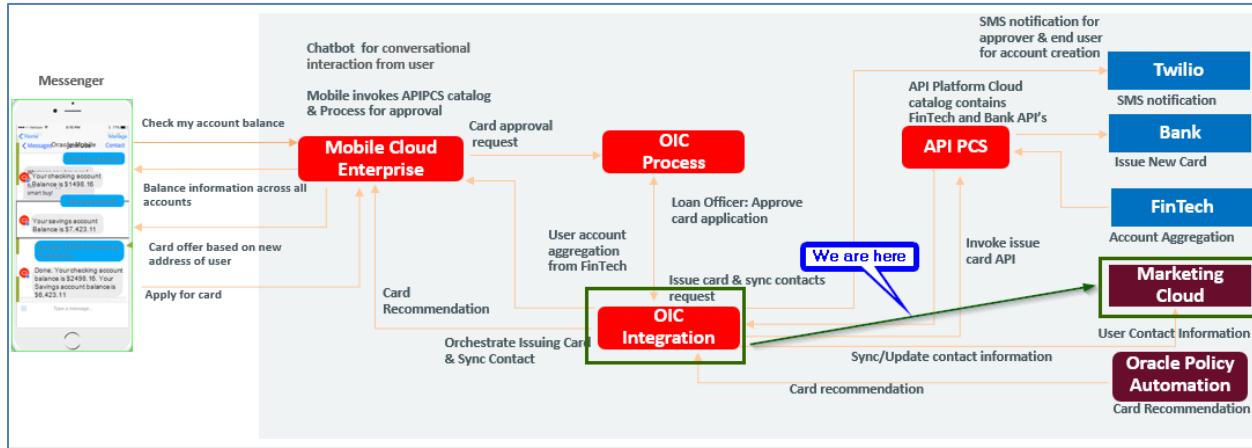
If Lisa decides to apply for the card, we gather basic information including Lisa address which might need update and send it for approval.

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 Notes**" 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/integrationCloudServiceApp.htm)

[http://docs.oracle.com/cloud/latest/marketingcs\\_gs/OMCAA/Help/Apps/IntegrationCloudService/Tasks/InstallingOIC.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.

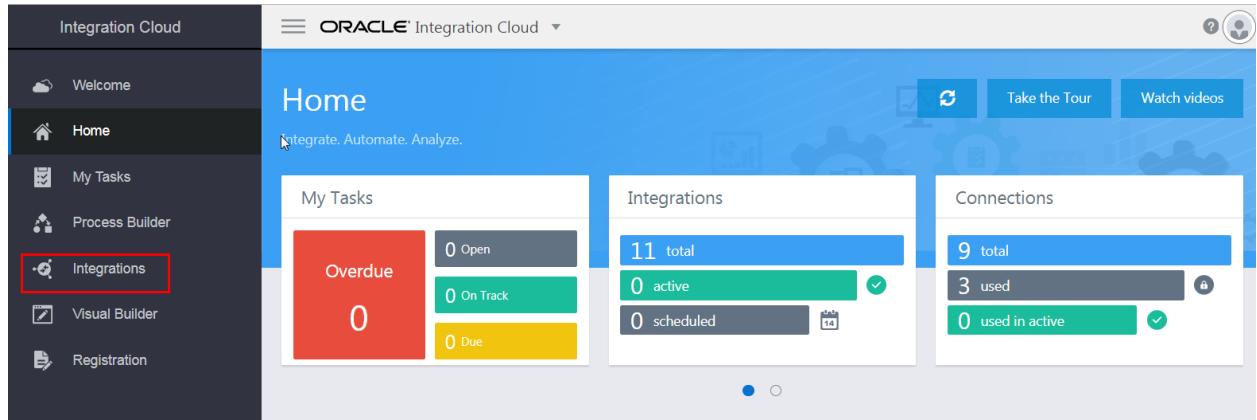
### Create REST Adapter based Connection

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

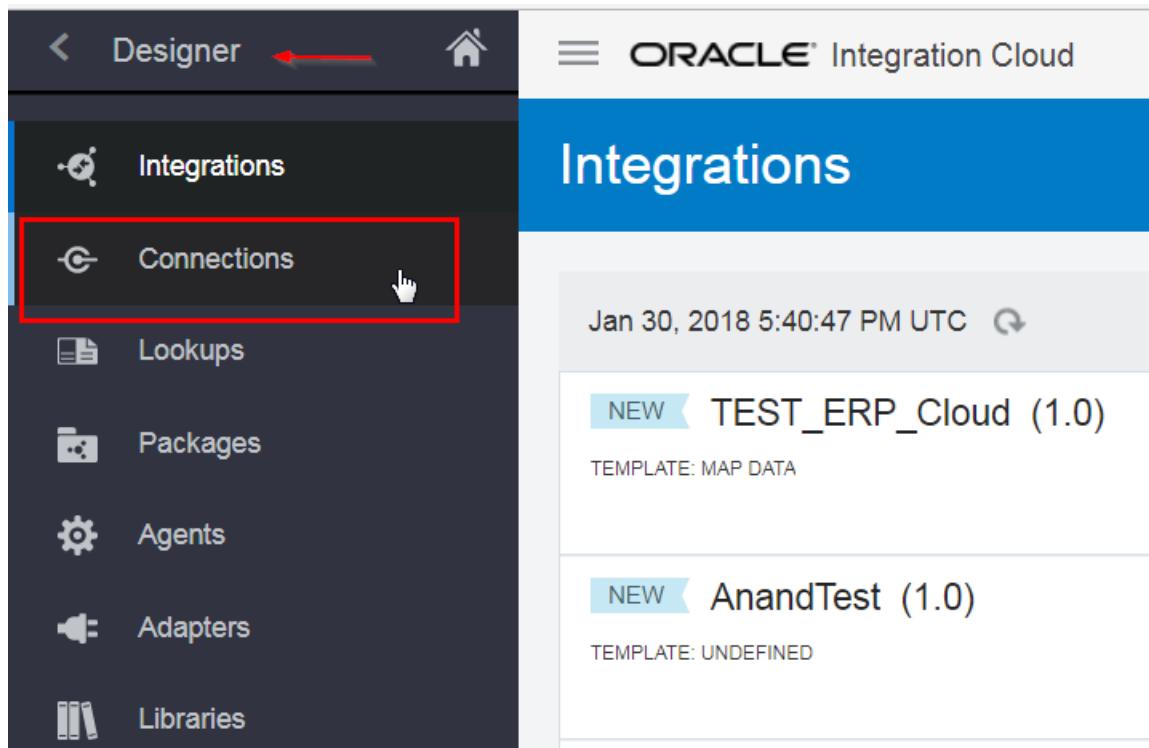
**Refer to the OIC URL& credentials provided to you.**

**Example: OIC instance URL** <https://<hostname>.oraclecloud.com/ic/home/>

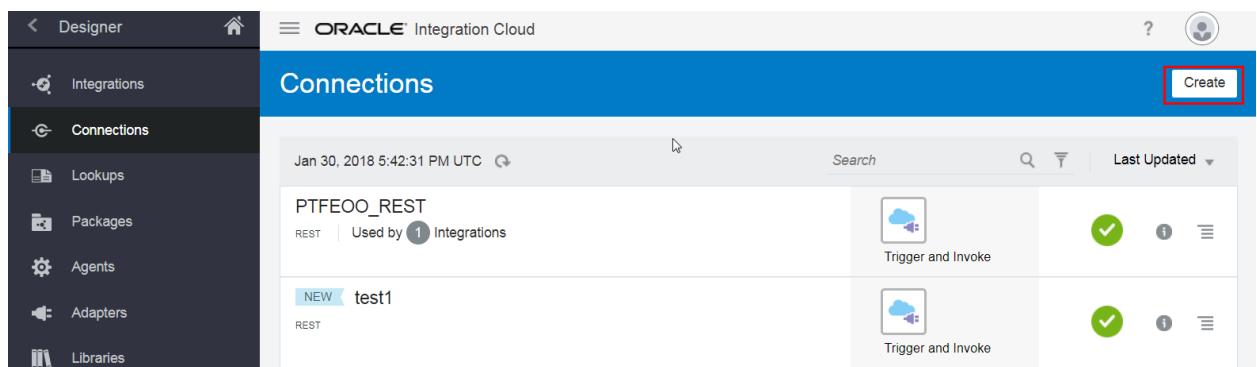
2. Click “**Got it**” if you see a welcome message. Else, proceed to next step.
3. Click on **Integrations** from the Menu on the left



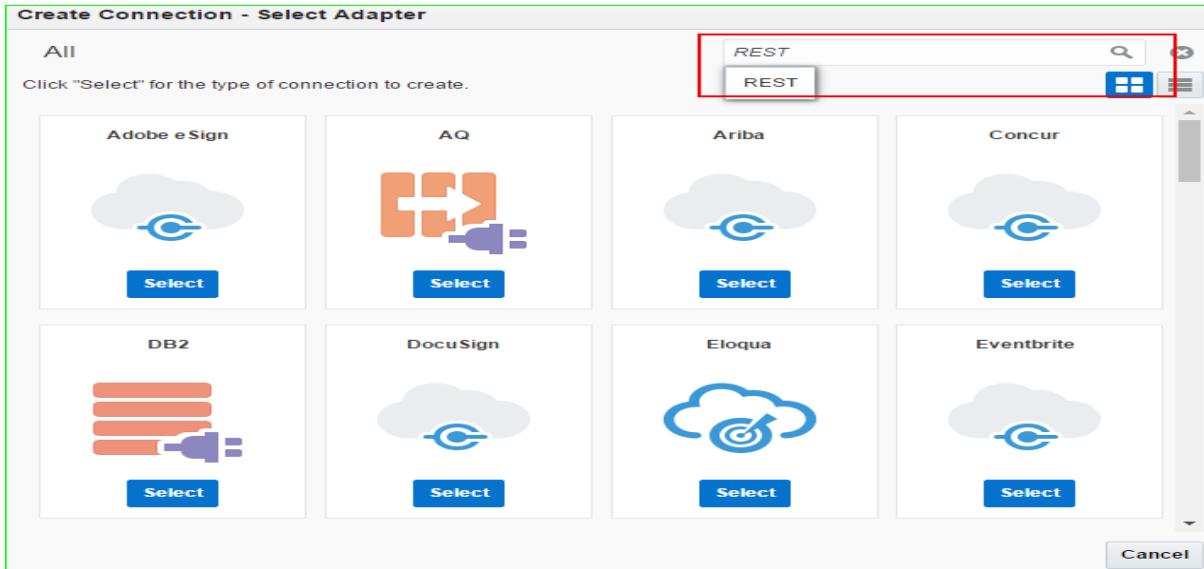
4. In “**Designer**” page, Click on **Connections**



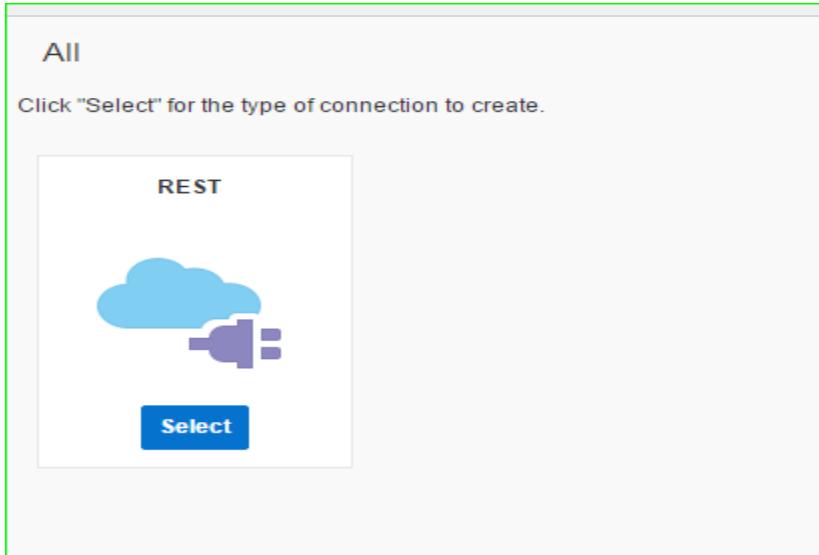
5. Click “Create” on top right.



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.

**New Connection - Information**

Enter information that describes the connection. Use a meaningful name and description to help others find your connection when they create their own integrations. The Identifier must be unique and can be set only when the connection is created.

* Name	FSDP_Eloqua_RESTadp_USERNAME
* Identifier	FSDP_ELOQUA_RESTADP_USERNAME
Role	Trigger and Invoke
Description	Enter a brief description...

**Create** **Cancel**

10. Now click on **Configure Connectivity**

Email Address

Connection Properties  
Click Configure Connectivity to specify information to connect to your application/endpoint and process requests.

Connection Type

TLS Version

Connection URL

**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/worksheets>

**Connection Properties**

Enter information so we can connect to your application/endpoint and process requests.

Property Name	Property Value
* Connection Type	REST API Base URL
TLS Version	TLSv1.1
* Connection URL	https:// i/REST/1.0/data

**OK** **Cancel**

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”

**Credentials**

You can configure the Security Policy for this connection. Please select the Security Policy.

**Security Policy** **Basic Authentication**

Your application/endpoint requires that users and services provide security credentials for access. Specify the login credentials below.

Property Name	Property Value
* Username	OracleSESandbox015\marion.may
* Password	*****
* Confirm Password	*****

**OK** **Cancel**

14. Click Test at the top to test the connection

FSDP\_Eloqua\_RESTadp\_USERNAME

**Test** **Close** **Save**

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

Connection FSDP\_Eloqua\_RESTadp\_USERNAME was tested successfully. 1

FSDP\_Eloqua\_RESTadp\_USERNAME

Test Close Save

REST Connection 2 100% Last Saved: 3 minutes ago

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.

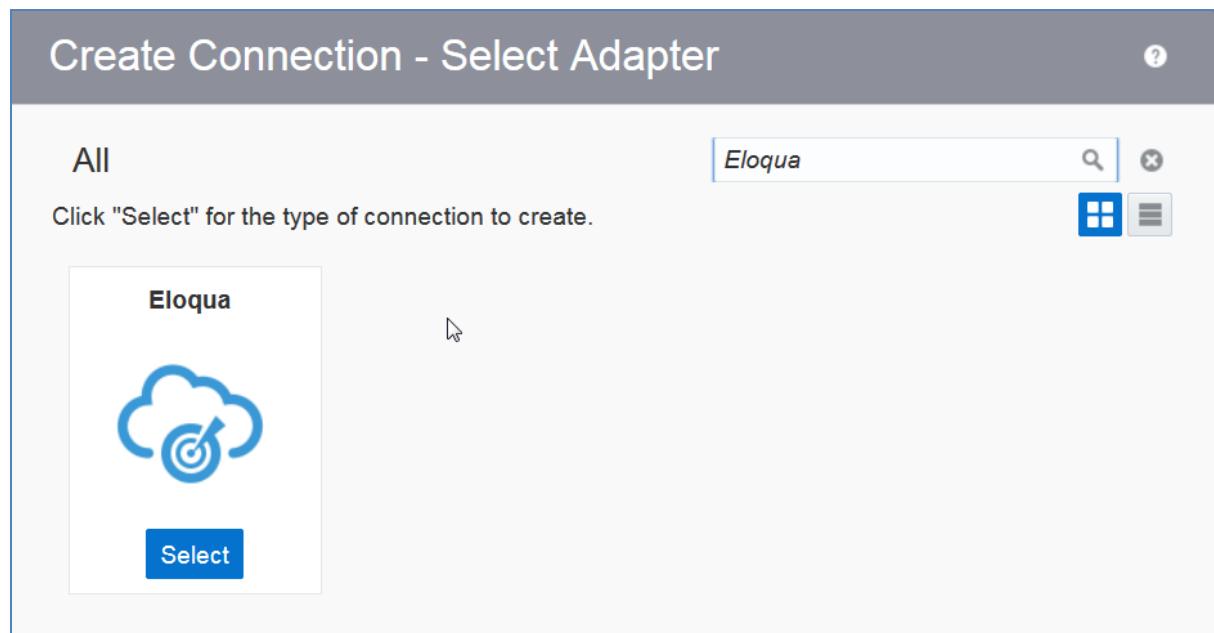
## Create Eloqua Adapter based Connection

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

**New Connection - Information**

Enter information that describes the connection. Use a meaningful name and description to help others find your connection when they create their own integrations. The Identifier must be unique and can be set only when the connection is created.

* Name	FSDP_Eloqua_Eloquaadp_USERNAME
* Identifier	FSDP_ELOQUA_ELOQUAAD_USERNAME
Role	Trigger and Invoke
Description	Enter a brief description...

**Create** **Cancel**

**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/IntegrationCloudServiceApp.htm)

[http://docs.oracle.com/cloud/latest/marketingcs\\_gs/OMCAA/Help/Apps/IntegrationCloudService/Tasks/InstallingOIC.htm](http://docs.oracle.com/cloud/latest/marketingcs_gs/OMCAA/Help/Apps/IntegrationCloudService/Tasks/InstallingOIC.htm)

- 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>**

**Credentials**

You can configure the Security Policy for this connection. Please select the Security Policy.

Security Policy Eloqua HTTP Basic Authentication ▾

Your application/endpoint requires that users and services provide security credentials for access. Specify the login credentials below.

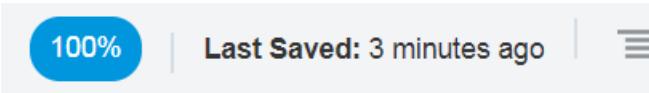
Property Name	Property Value
* Company	OracleSESandbox015
* Username	marion.may
* Password	*****
* Confirm Password	*****

OK Cancel

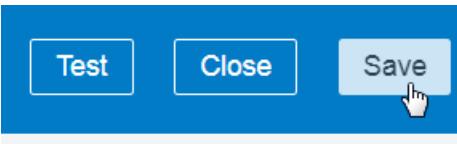
- Click Test at the top to test the connection



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



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



- Click on Close to return back to main page.

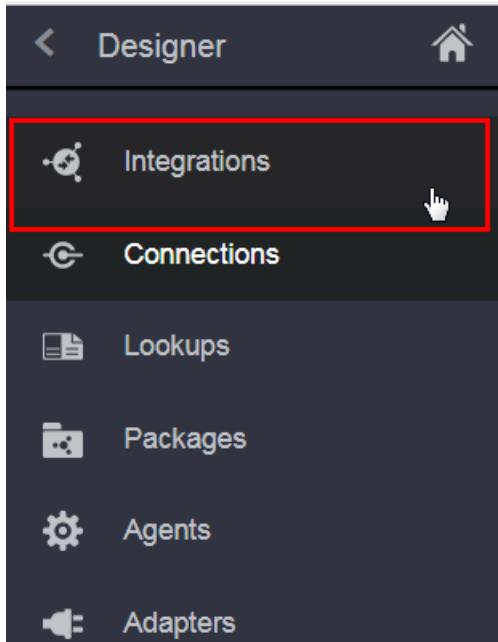


## Create OIC – Eloqua Integration

### 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

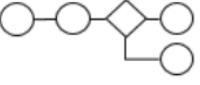


3. Select “**Orchestration**”.

## Create Integration - Select a Style/Pattern

How would you like to build your integration? Select a style/pattern to use.

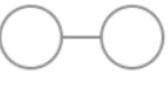
**Style: Orchestration**  
**Pattern: Map Data**



Drop trigger and invoke onto a blank canvas

**Select**

**Style: Template**  
**Pattern: Map Data**



Drop trigger and invoke onto an empty template

**Select**

**Style: Template**  
**Pattern: Publish To IC**



Connect your trigger to send messages to IC

**Select**

**Style: Template**  
**Pattern: Subscribe To IC**



Add invokes to receive messages from IC

**Select**

**Cancel**

- Name it as “**FSDP\_SyncContacts\_username**” (replace **\_username** with the login name assigned to you) and leave the other fields to defaults

### Create New Integration

Enter information that describes this integration.

**What triggers this integration?**

Application event or business object  
 Schedule

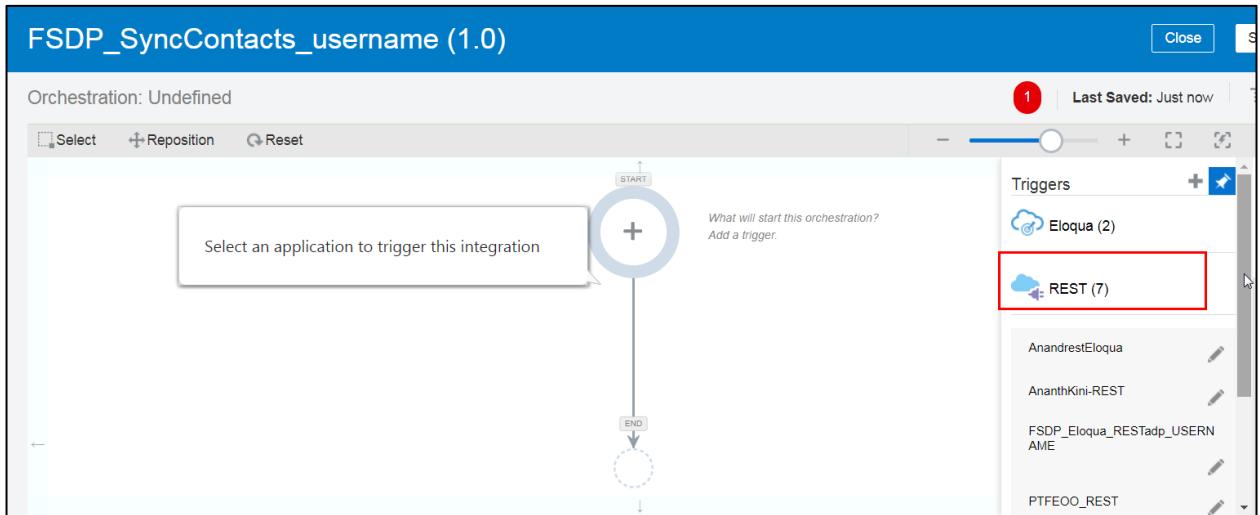
**Describe this integration** Use a meaningful name and description that will help others find and understand this integration. The Identifier and Version can be set only when the integration is created. The combination of Identifier and Version must be unique.

<p>* <b>What do you want to call your integration?</b>  <input type="text" value="FSDP_SyncContacts_username"/></p> <p>* <b>Identifier</b>  <input type="text" value="FSDP_SYNCCONTAC_USERNAME"/></p> <p>* <b>Version</b>  <input type="text" value="01.00.0000"/></p>	<p><b>What does this integration do?</b>  <input type="text" value="Describe the integration's purpose and detail"/></p> <p><b>Which package does this integration belong to?</b>  <input type="text" value="Enter a new or existing package name"/></p>
--	--

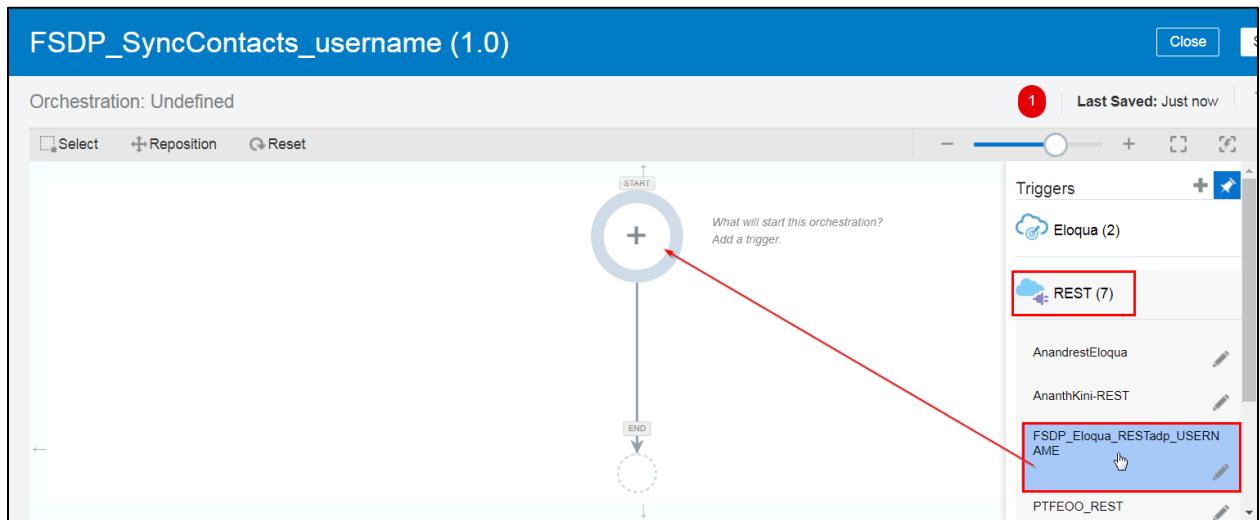
[ration.us2.oraclecloud.com/ics/faces/global#](#) **Create** **Cancel**

- Click “**Create**”. Integration will be created and you will be launched on to it in Edit mode.
- 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.**

**Configure Oracle REST Endpoint**

Welcome to the Oracle REST Endpoint Configuration Wizard

This wizard helps you configure an endpoint using the Oracle REST Cloud adapter.

**Basic Info**

\* What do you want to call your endpoint?  
SynchContacts

What does this endpoint do?  
Describe the endpoint's purpose and detail

\* What is the endpoint's relative resource URI?  
/contacts

\* What action does the endpoint perform?  
POST

Based on your selections, you can add parameters or configure a request and/or response for this endpoint.

Select any options that you want to configure:

	<input type="checkbox"/> Add and review parameters for this endpoint <input checked="" type="checkbox"/> Configure a request payload for this endpoint <input checked="" type="checkbox"/> Configure this endpoint to receive the response  Configure Request Headers? <input type="checkbox"/> Standard <input type="checkbox"/> Custom
--	--

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

**Configure Oracle REST Endpoint**

Configure the **Request Payload**  
Configure the request payload details for this endpoint.

**Basic Info**

**Request** (selected)

**Request Parameters**

**Request Headers**

**CORS Configuration**

**Response**

**Response Headers**

**Summary**

Select the attachment processing options

Accept attachments from request

Request is HTML form

Select the request payload file

XML Schema  JSON Sample

? Sample Location  No file selected. --OR-- enter sample JSON <<< inline >>>

\* Element

Select the type of payload with which you want the endpoint to receive

XML

XML(text)

JSON

URL-encoded

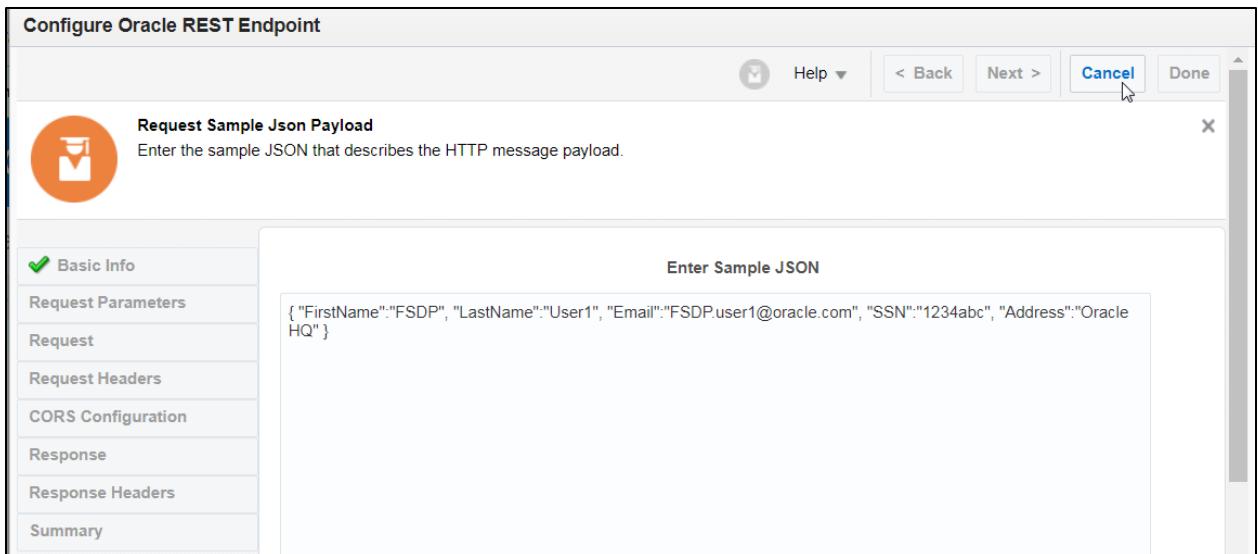
11. Click “<<<inline>>>” to view the uploaded json. The contents should look like below.

**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

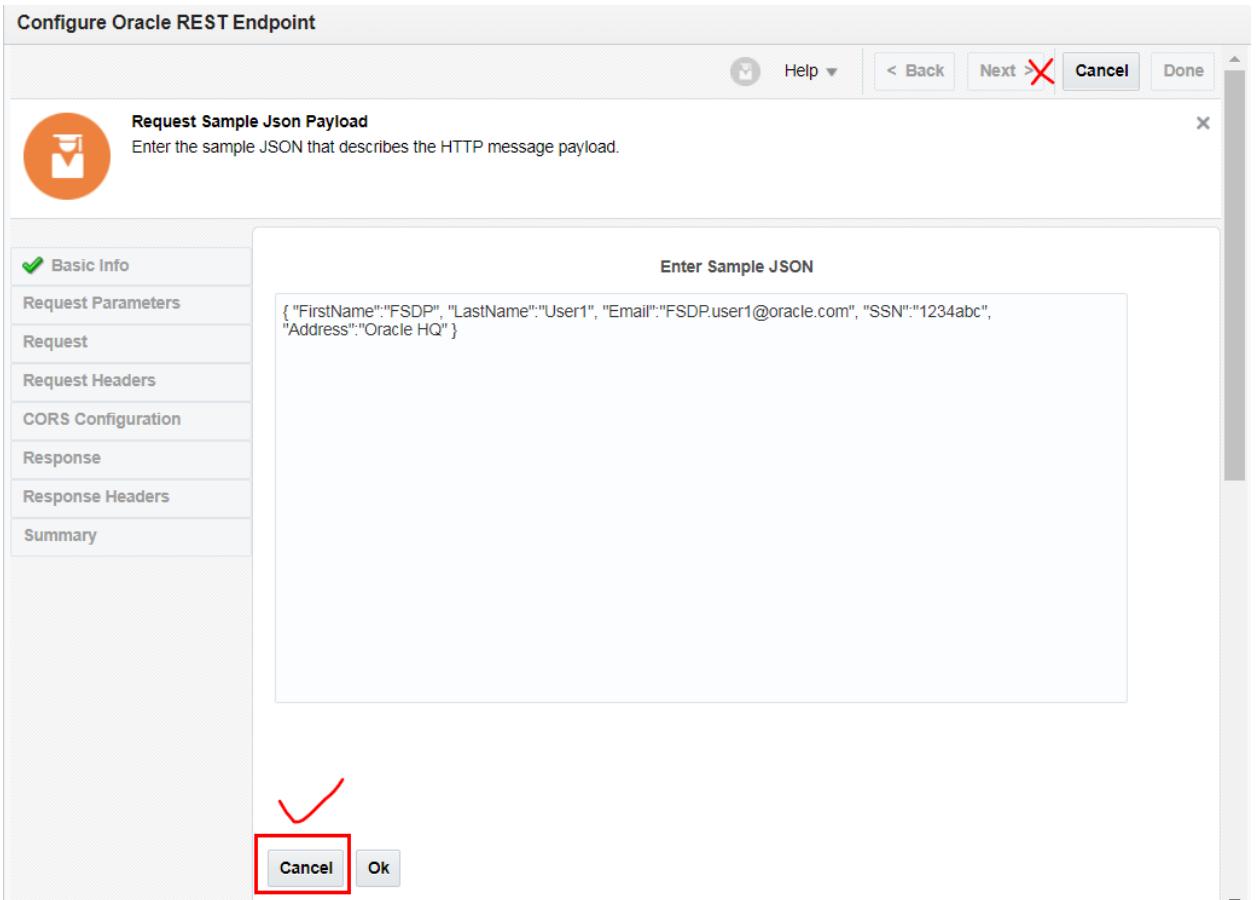
XML Schema  JSON Sample

? Sample Location  No file chosen --OR-- enter sample JSON <<< inline >>>

\* Element



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.

 Configure the Response Payload  
Specify the response payload details for this integration.

Basic Info

Request Parameters

Request

Request Headers

CORS Configuration

**Response**

Response Headers

Summary

Select the attachment processing options

Accept attachments from response

Response is HTML form

Select the response payload file

XML Schema  JSON Sample

Sample Location  No file selected. --OR-- enter sample JSON <<< inline >>>

\* Element

Select the type of payload with which you want the endpoint to reply

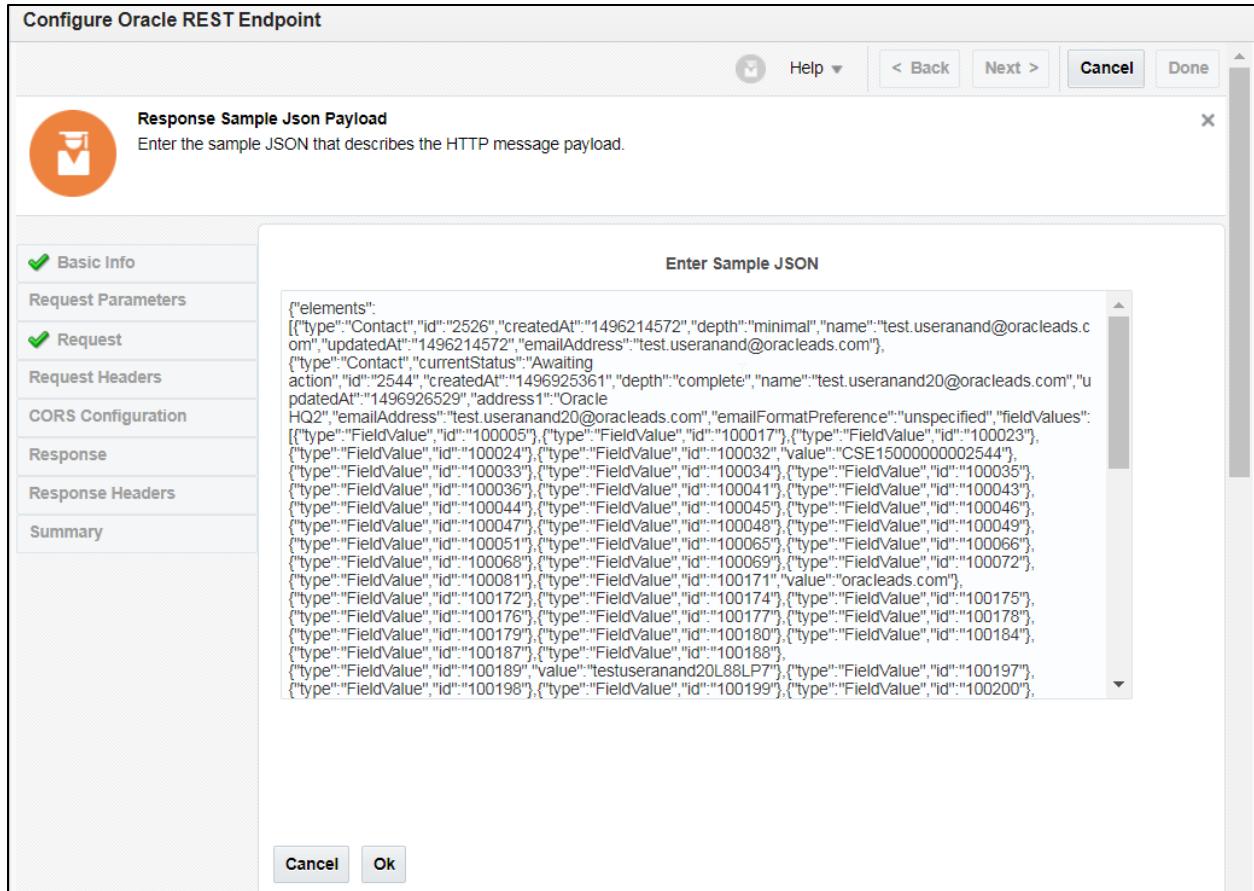
XML

XML(text)

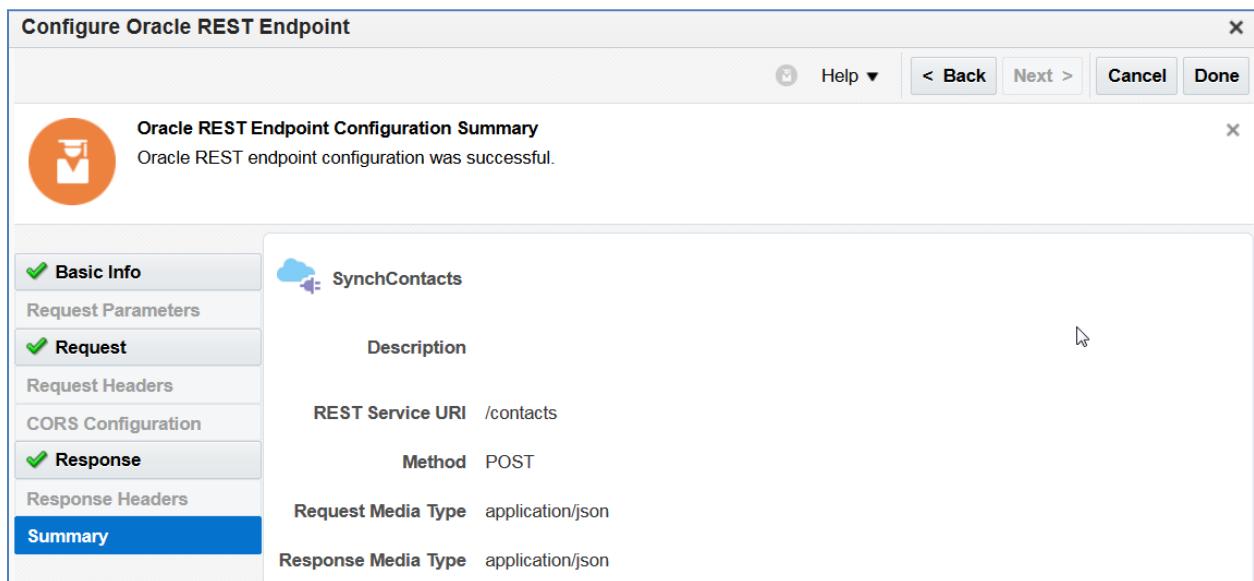
JSON

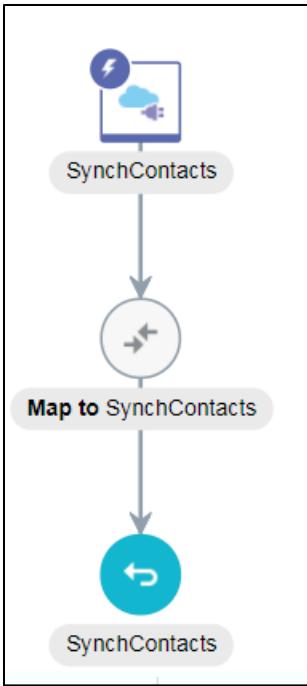
15. Click “<<<inline>>>” to view the uploaded json. The contents should look like below.

**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

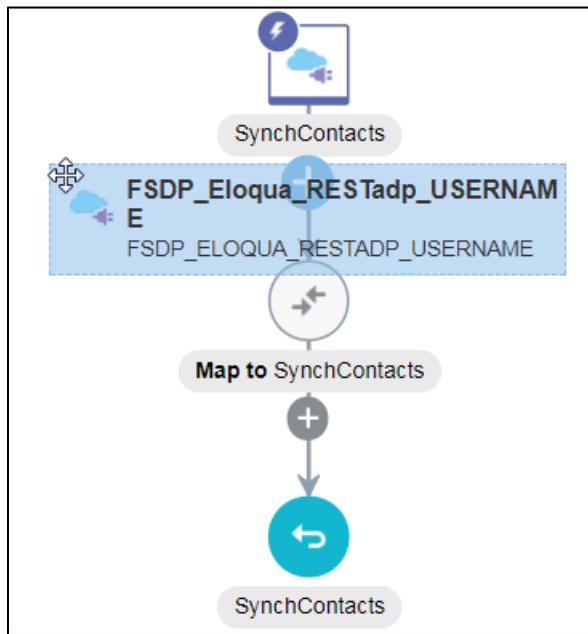
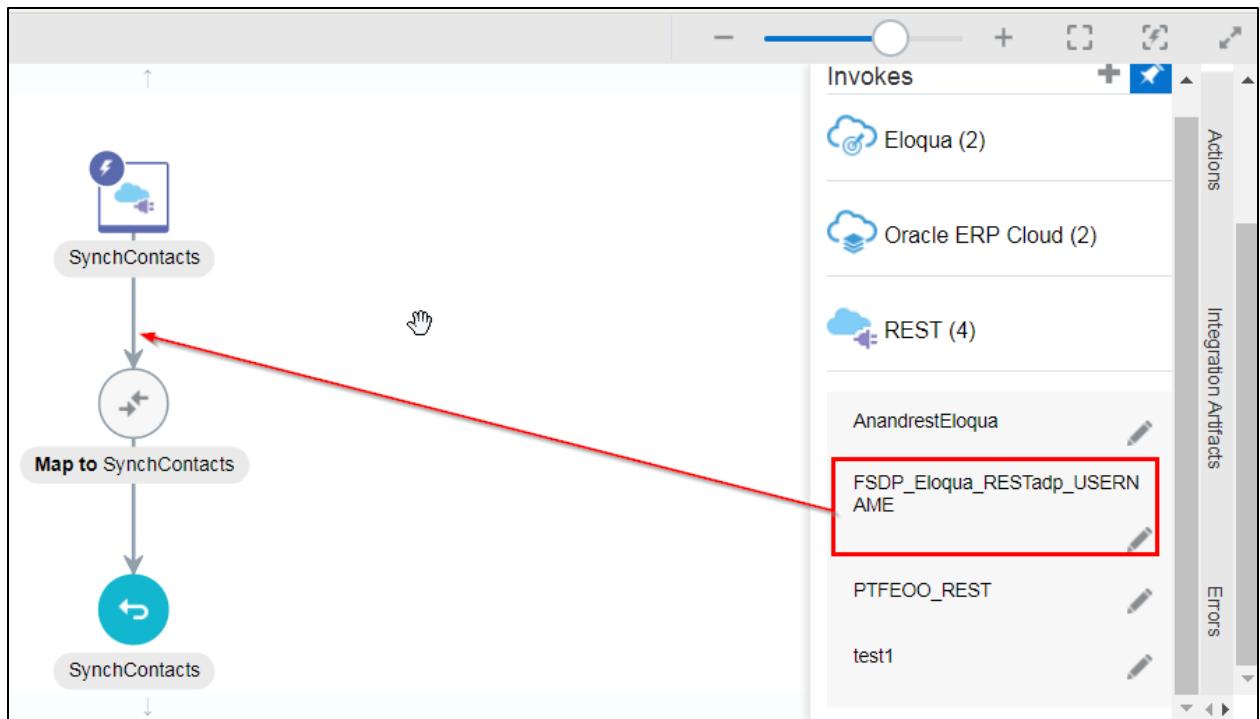




19. “Save” the integration.

### *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.

Configure Oracle REST Endpoint

Welcome to the Oracle REST Endpoint Configuration Wizard  
This wizard helps you configure an endpoint using the Oracle REST Cloud adapter.

**Basic Info**

\* What do you want to call your endpoint?  
GETContactdetails

What does this endpoint do?  
Describe the endpoint's purpose and detail

\* What is the endpoint's relative resource URI?  
/contacts

\* What action does the endpoint perform?  
GET

Based on your selections, you can add parameters or configure a request and/or response for this endpoint.

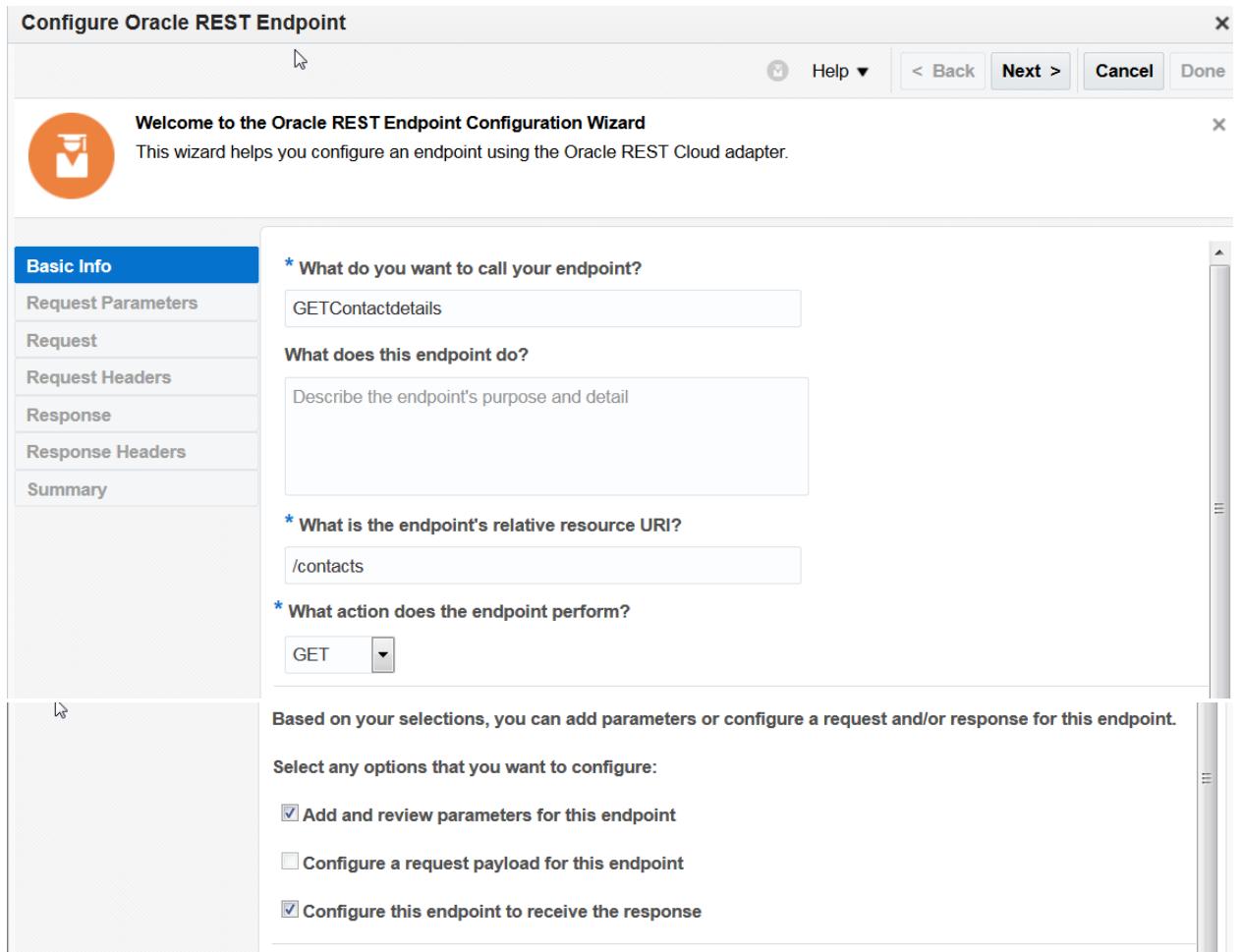
Select any options that you want to configure:

Add and review parameters for this endpoint

Configure a request payload for this endpoint

Configure this endpoint to receive the response

< Back Next > Cancel Done



4. Click “Next”. Add below request parameters by clicking “+” button

Name	DataType
search	string
depth	string

**Configure Oracle REST Endpoint**

**Configure the Request Query Parameters**  
Configure the request query parameters for this endpoint.

**Basic Info**

**Request Parameters**

**Resource URI** /contacts

**Specify Query Parameters**

Name	Data Type
search	string
depth	string

**Template Parameters**  
Displays the template parameters in the relative resource URI. Template parameters are determined by details you

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

**Configure Oracle REST Endpoint**

**Configure the Response Payload**  
Specify the response payload details for this integration.

**Basic Info**

**Request Parameters**

**Response**

**Select the attachment processing options**

Process attachments from response

Response is HTML form

**Select the response payload file**

XML Schema  JSON Sample

**Sample Location**  No file selected.

--OR-- enter sample JSON <<< inline >>>

**\* Element** response-wrapper

**Select the type of payload with which you want the endpoint to reply**

XML

XML(text)

JSON

Other Media Type

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...

Configure Oracle REST Endpoint

Help < Back Next > Cancel Done

Response Sample Json Payload

Enter the sample JSON that describes the HTTP message payload.

Basic Info

Request Parameters

Request

Request Headers

Response

Response Headers

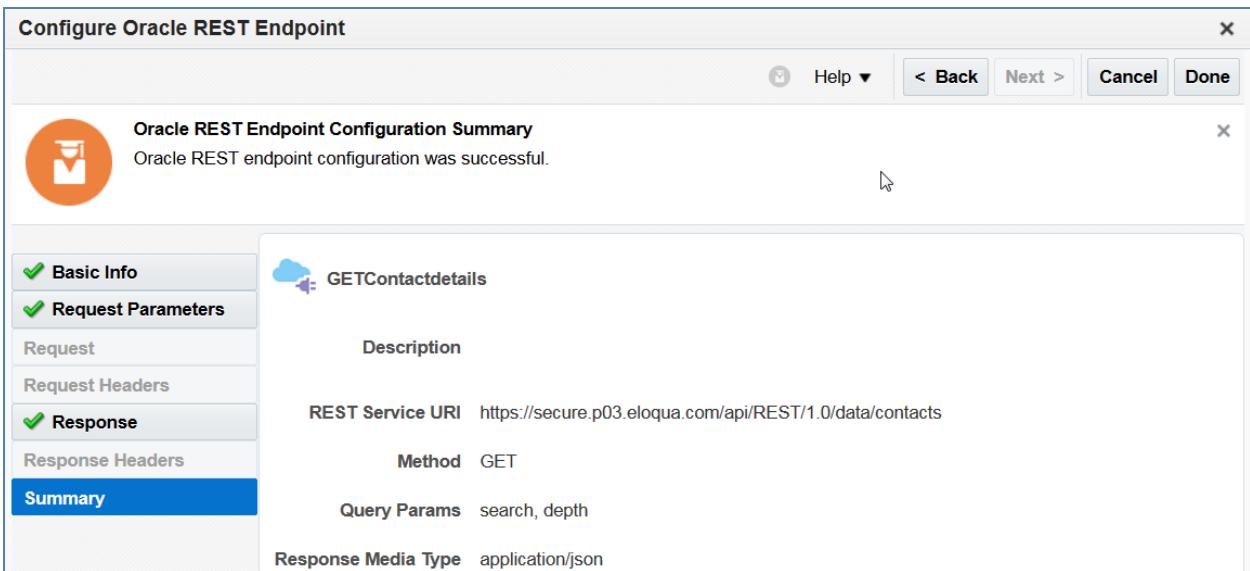
Summary

Enter Sample JSON

```
{"elements": [{"type": "Contact", "id": "2525", "createdAt": "1495220521", "depth": "minimal", "name": "test.user1@oracleleads.com", "updatedAt": "1495220521", "emailAddress": "test.user1@oracleleads.com"}, {"type": "Contact", "currentStatus": "Awaiting", "action": "id": "2537", "createdAt": "1496397458", "depth": "complete", "name": "test.useranand11@oracleleads.com", "updatedAt": "1496675691", "address1": "add1", "address2": "add2", "address3": "add3", "city": "MyCity", "country": "MyCountry", "emailAddress": "test.useranand11@oracleleads.com", "emailFormatPreference": "unspecified", "fieldValues": [{"type": "FieldValue", "id": "100005"}, {"type": "FieldValue", "id": "100017"}, {"type": "FieldValue", "id": "100023"}, {"type": "FieldValue", "id": "100024"}, {"type": "FieldValue", "id": "100032", "value": "CSE1500000000002537"}, {"type": "FieldValue", "id": "100033"}, {"type": "FieldValue", "id": "100034"}, {"type": "FieldValue", "id": "100035"}, {"type": "FieldValue", "id": "100036"}, {"type": "FieldValue", "id": "100041"}, {"type": "FieldValue", "id": "100043"}, {"type": "FieldValue", "id": "100044"}, {"type": "FieldValue", "id": "100045"}, {"type": "FieldValue", "id": "100046"}, {"type": "FieldValue", "id": "100047"}, {"type": "FieldValue", "id": "100048"}, {"type": "FieldValue", "id": "100049"}, {"type": "FieldValue", "id": "100051"}, {"type": "FieldValue", "id": "100065"}, {"type": "FieldValue", "id": "100066"}, {"type": "FieldValue", "id": "100068"}, {"type": "FieldValue", "id": "100069"}, {"type": "FieldValue", "id": "100072"}, {"type": "FieldValue", "id": "100081"}, {"type": "FieldValue", "id": "100171", "value": "oracleleads.com"}, {"type": "FieldValue", "id": "100172"}, {"type": "FieldValue", "id": "100174"}, {"type": "FieldValue", "id": "100175"}, {"type": "FieldValue", "id": "100176"}, {"type": "FieldValue", "id": "100177"}, {"type": "FieldValue", "id": "100178"}, {"type": "FieldValue", "id": "100179"}, {"type": "FieldValue", "id": "100180"}, {"type": "FieldValue", "id": "100184"}, {"type": "FieldValue", "id": "100187"}, {"type": "FieldValue", "id": "100188"}, {"type": "FieldValue", "id": "100189"}, {"type": "FieldValue", "id": "100189", "value": "testuseranand11.88LP9"}, {"type": "FieldValue", "id": "100197"}, {"type": "FieldValue", "id": "100198"}, {"type": "FieldValue", "id": "100199"}, {"type": "FieldValue", "id": "100200"}]}
```

Cancel Ok

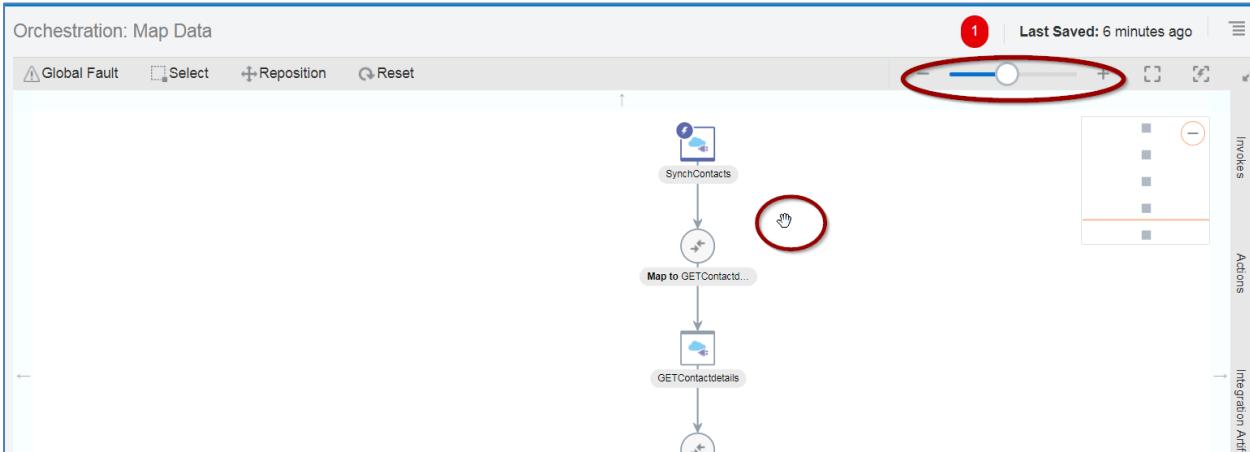
8. Click “**Cancel**” on the bottom. **DO NOT** click the one on top.
9. Click “**Next**” and review configuration summary.



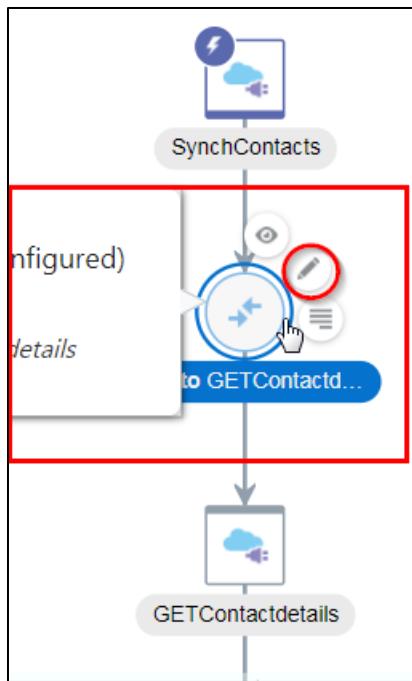
10. Click **Done** and “Save” the integration.

### 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.



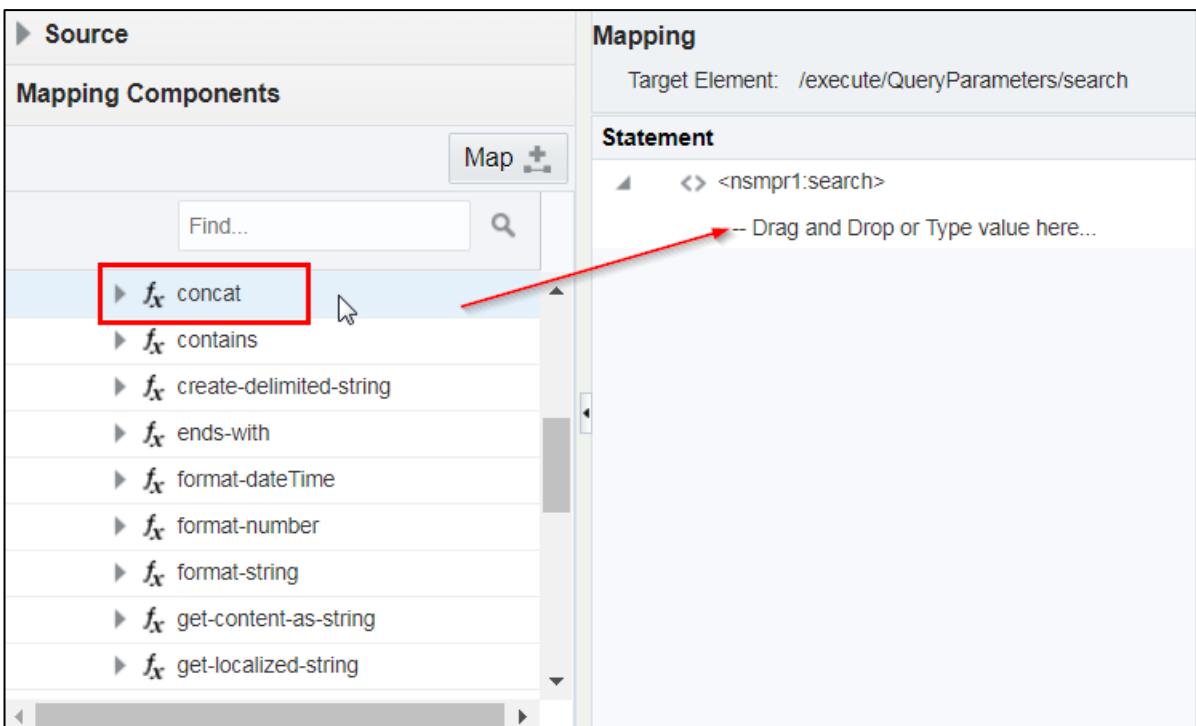
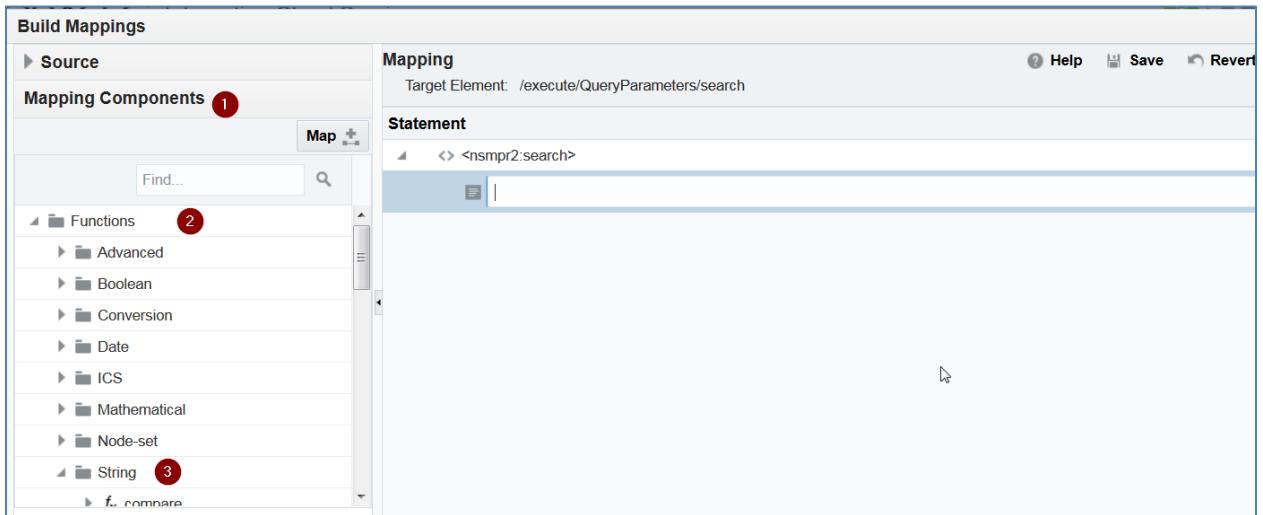
1. Click on “**Map to GETContactDetails**” MAP -> “**EDIT**”



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.

Mapping

Target Element: /execute/QueryParameters/search

Statement

```

<> <nsmpr1:search>
  ↳ value-of
    ↳ select
      ↳ concat(string1, string2, ...)
        ↳ "emailAddress="
  
```

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

Build Mappings

Source

Mapping

Target Element: /execute/QueryParameters/search

Statement

```

<> <nsmpr1:search>
  ↳ value-of
    ↳ select
      ↳ concat(string1, string2, ...)
        ↳ emailAddress=
          ↳ string2
  
```

Source

Mapping

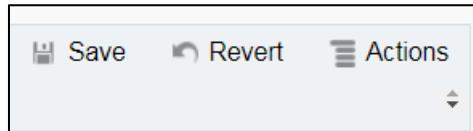
Target Element: /execute/QueryParameters/search

Statement

```

<> <nsmpr1:search>
  ↳ value-of
    ↳ select
      ↳ concat(string1, string2, ...)
        ↳ "emailAddress="
        ↳ </nssrcmpr.execute/nsmp2:request-wrapper/nsmp2 Email>
  
```

7. Click “Save”.



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

The screenshot shows the 'Map' section of the 'Map to GETContactdetails' configuration page. The Source pane on the left lists fields like \*execute, \*request-wrapper, FirstName, LastName, Email, SSN, Address, \$tracking\_var\_1, \$tracking\_var\_2, and \$tracking\_var\_3. The Target pane on the right lists \*execute, QueryParameters, search, depth, ConnectivityProperties, RestApi, and Plugin. A green line connects the 'Email' field in the Source pane to the 'depth' field in the Target pane, indicating a successful mapping.

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

The screenshot shows the 'Build Mappings' dialog. The Source pane lists fields including Email, which is selected and has a green checkmark. The Mapping section shows the Target Element as /execute/QueryParameters/depth. The Statement pane contains the mapping rule <nsmpr2:depth> complete. The 'Save' button is highlighted.

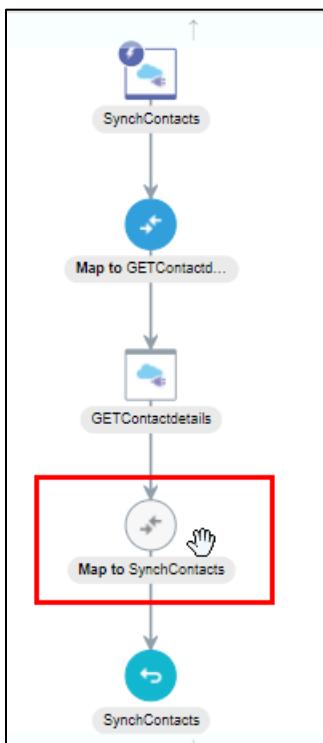
11. Click “**Save**” and then “**Close**”.
12. On the main mapping page, **verify** the mapping as below. Click “**Validate**”. You should see a success message.

The screenshot shows the main 'Map to GETContactdetails' configuration page. The Source and Target panes are identical to the previous screenshots. A message in the center states "Mapping is valid and ready to use". The 'Validate' button in the top right corner has a red notification badge with the number '1'. The 'Test' button is also visible.

13. Click “**Close**”
14. “**Save**” the main integration again.

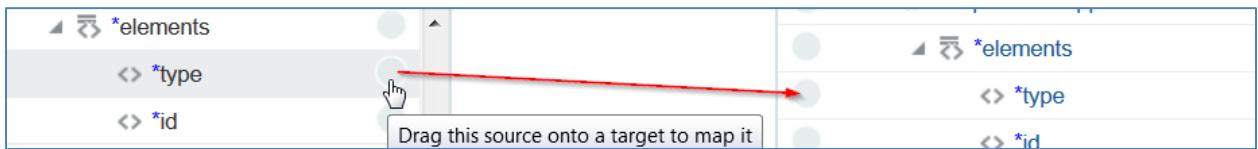
### *Map SynchContacts endpoint*

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)



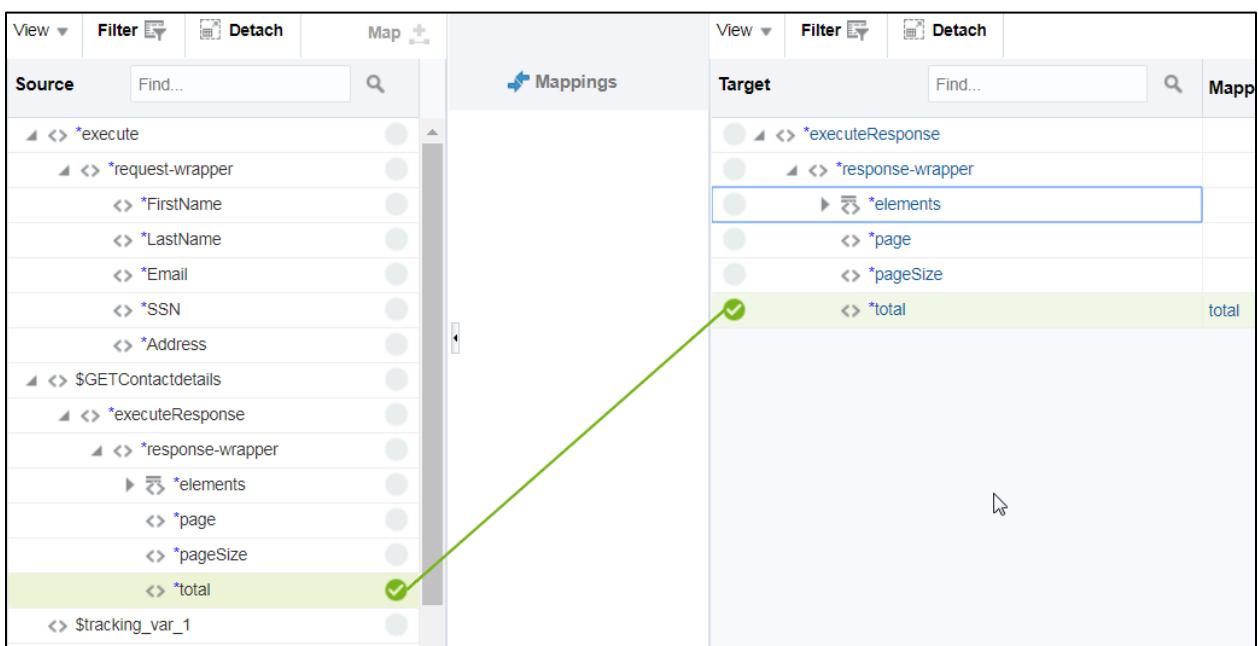
2. Expand the **Source** elements under “\$GetContactdetails” on the left and **Target** elements on the right.

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 Switching Rules

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

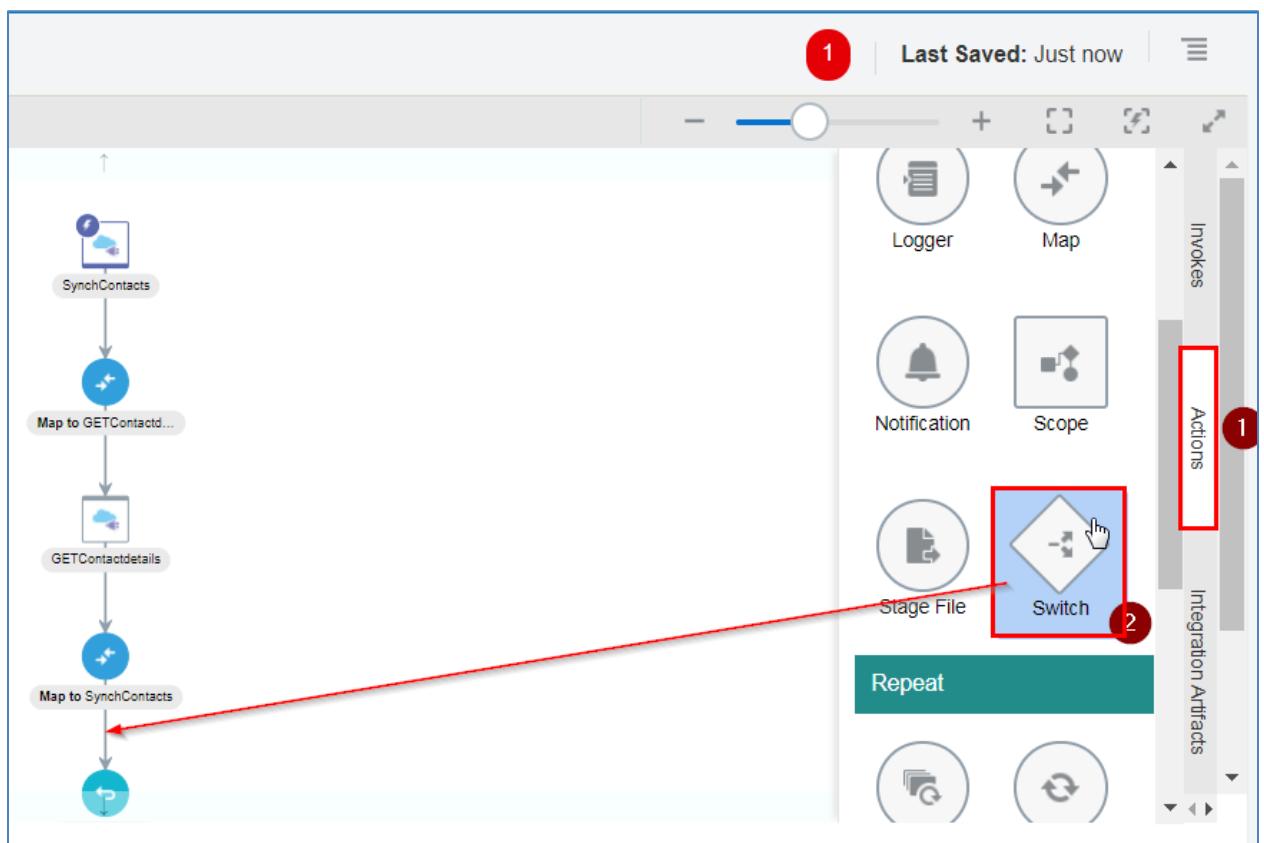
```
#####
if contact exists{
```

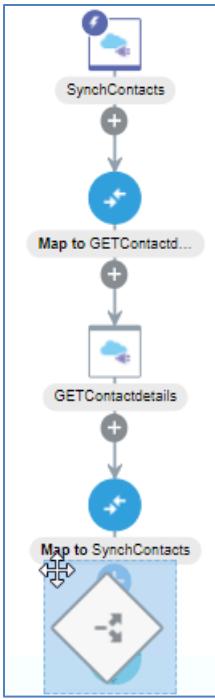
```

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

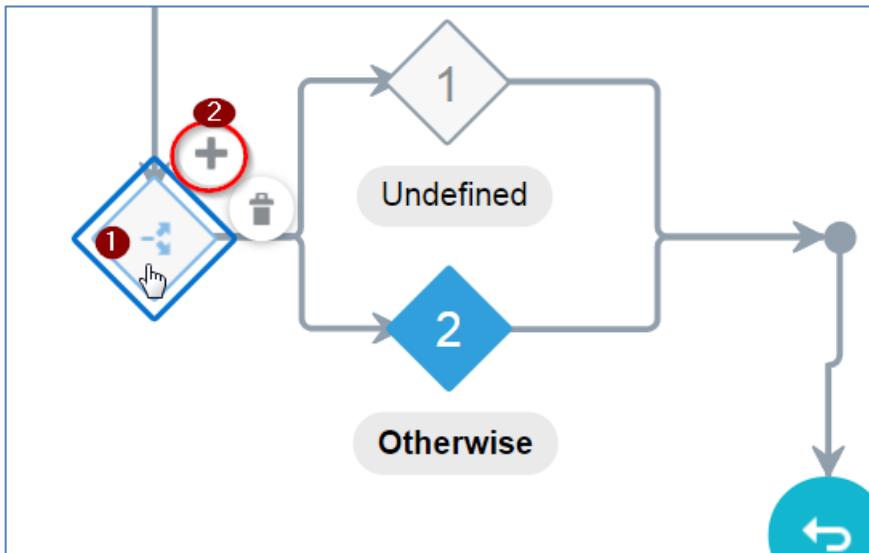
```

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

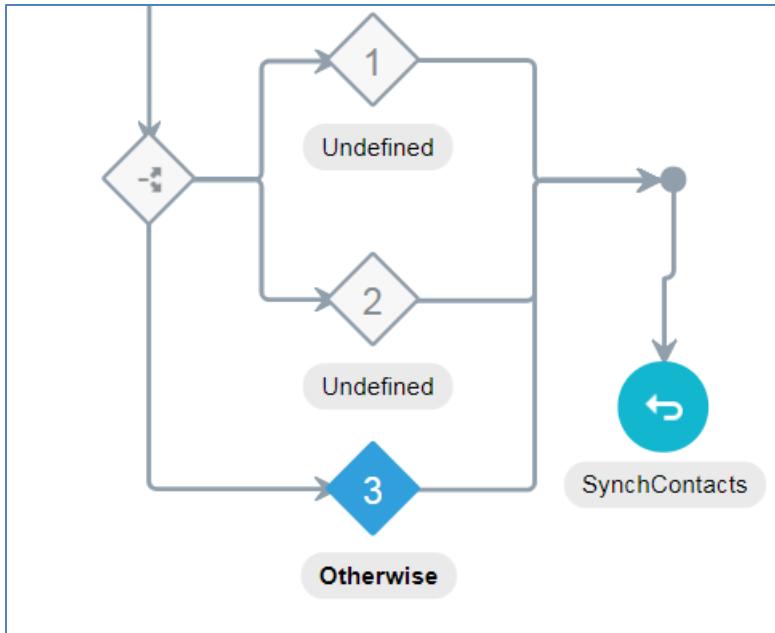




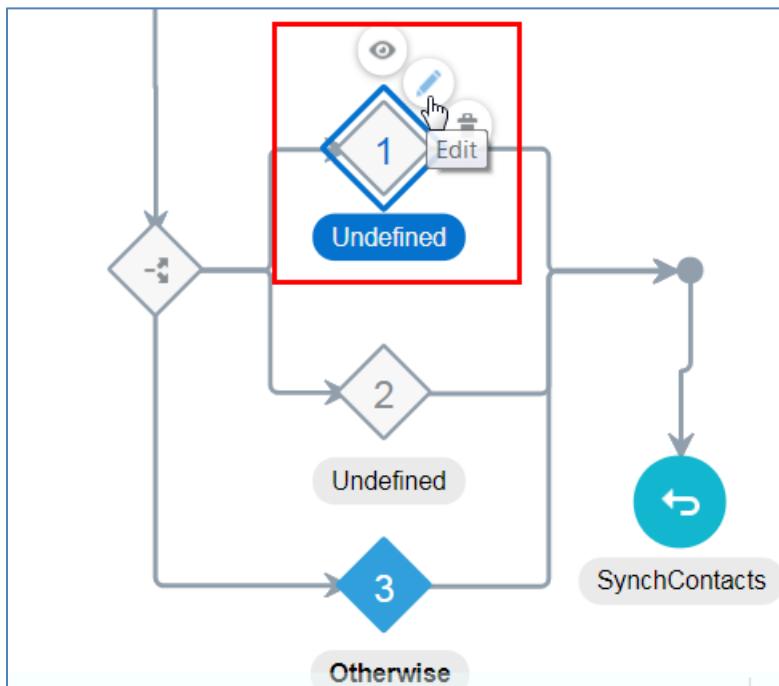
3. It should create a Switch with 2 routes. We need to add more routes.
4. Click on the **Switch** and click “+”



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.

The screenshot shows the Oracle Integration Configuration interface. On the left, there are two panels: 'Inputs' and 'Components'. The 'Inputs' panel has tabs for 'View', 'Filter', and 'Detach'. Below these are sections for 'Source' and 'Components', each with a 'Find...' search bar and a magnifying glass icon. In the 'Source' section, a tree view shows nodes like '\$GETContactdetails', '\*executeResponse', '\*response-wrapper', '\*elements', '\*page', '\*pageSize', and '\*total'. The '\*total' node is highlighted with a yellow background and a green checkmark. A red arrow points from this checkmark to the 'Drag and drop or type here' field in the 'Expression Name' dialog box. The 'Expression Name' dialog box has a red border around its input field. It contains a 'New Condition' section with a 'Drag and drop or type here' field and a dropdown menu with '=' selected.

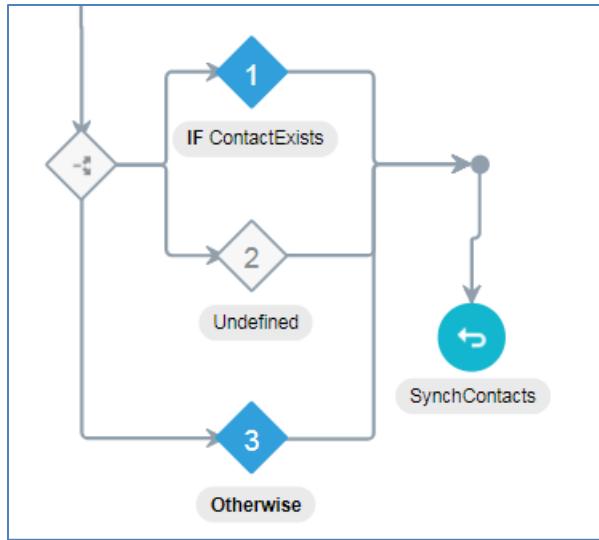
**Note:** (provide a name to the expression to reflect business condition. If you don't OIC will take care of it.) Lets give the name as "**ContactExists**"

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

This screenshot shows the same interface after step 8. The 'Inputs' panel is identical. In the 'Expression Name' dialog box, the 'New Condition' section now contains the path '\$GETContactdetails/n smpr7:executeRespon se/nsmpr6:response-' followed by a dropdown menu with '=' selected. Below this, the bottom field of the condition is highlighted with a red border and contains the value '0'.

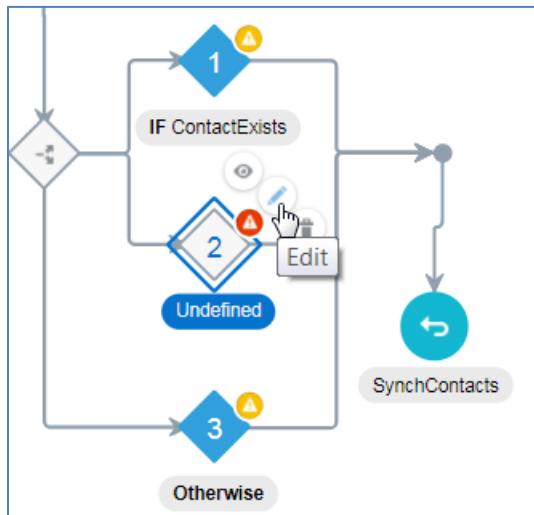
9. Click "**Validate**" and "**Close**"

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)

**Inputs**

View ▾ Filter Detach

**Source**

Find...

<> *name
<> *updatedAt
<> *emailAddress
<> *currentStatus
<> *address1
<> *address2

Expression Name  
AddressIsSame

New Condition

```
/nssrcmp:execute/nsmpr1:request-wrapper/nsmpr1:Address
```

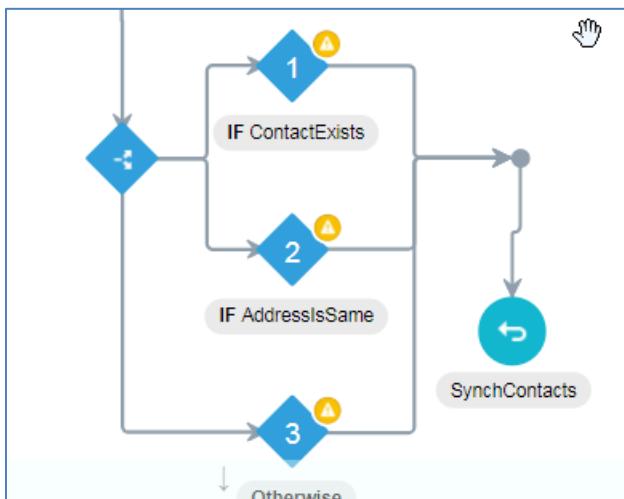
=

```
$GETContactdetails/nsmpr7:executeResponse/nsmpr6:response-wrapper/nsmpr6:elements/nsmpr6:address1
```

13. Ensure that you see the expressions as highlighted above on the right.

14. Click “Validate” and “Close”

15. Click **Save** again



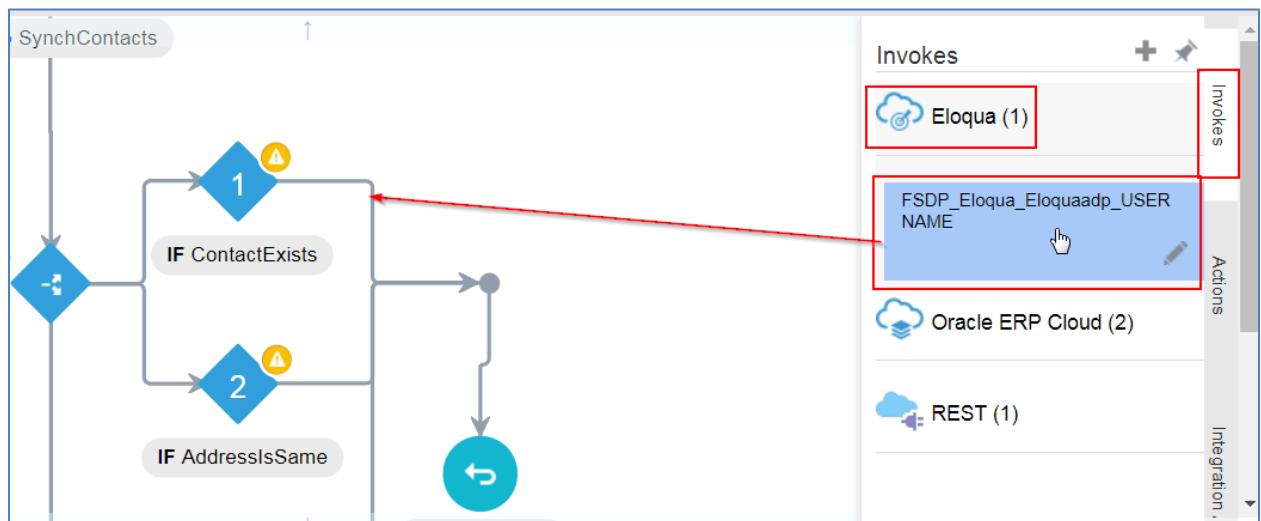
**Note:** No worries if you don't see the name of expression as “**AddressIsSame**” (you might see as “IF Address = address1”). We omitted that step intentionally to highlight OIC auto naming... Just proceed.

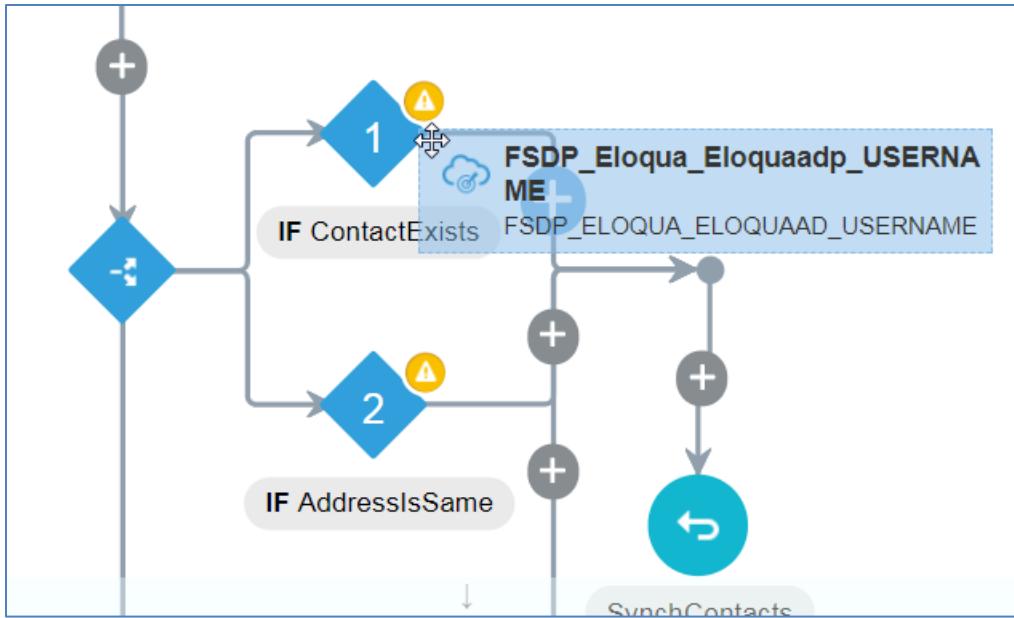
## Configure endpoints based on rules

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

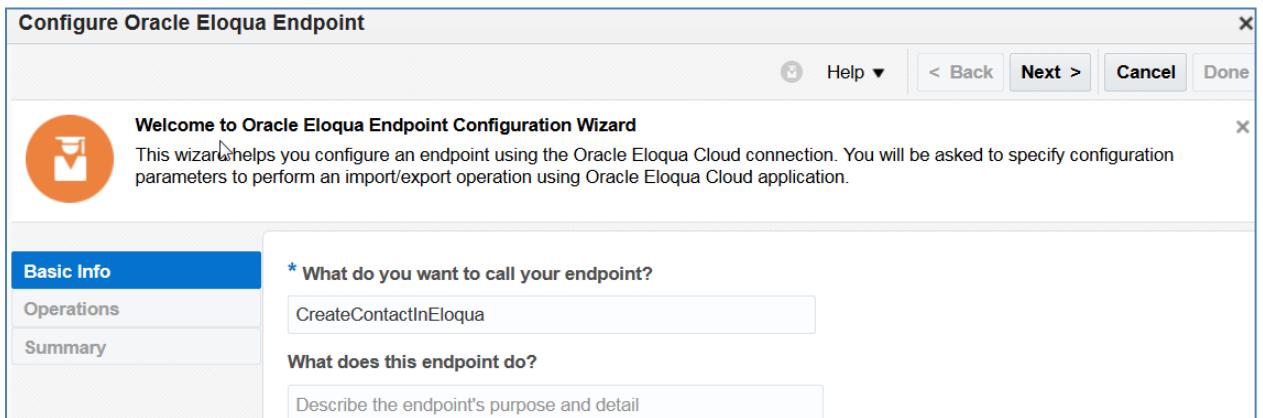
1. On the integration Orchestration designer page, from the “**Invokes**” menu on the right, Select “**Eloqua**=> “**FSDP\_Eloqua\_Eloquaadp\_USERNAME**” (NOT **username** but the one you created earlier), **drag&drop** it next to **first rule**.

**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.

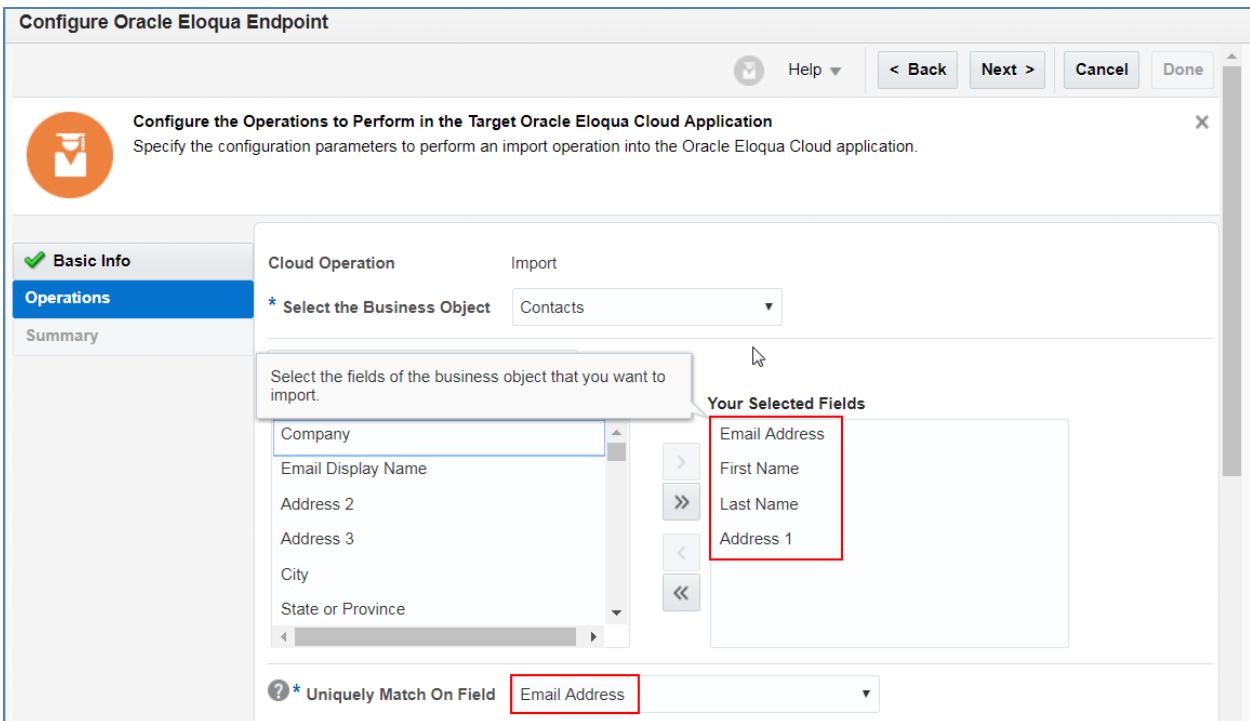




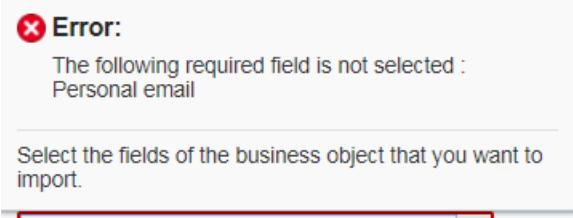
2. In the Eloqua Endpoint Configuration Wizard name it as “**CreateContactInEloqua**”. Click next



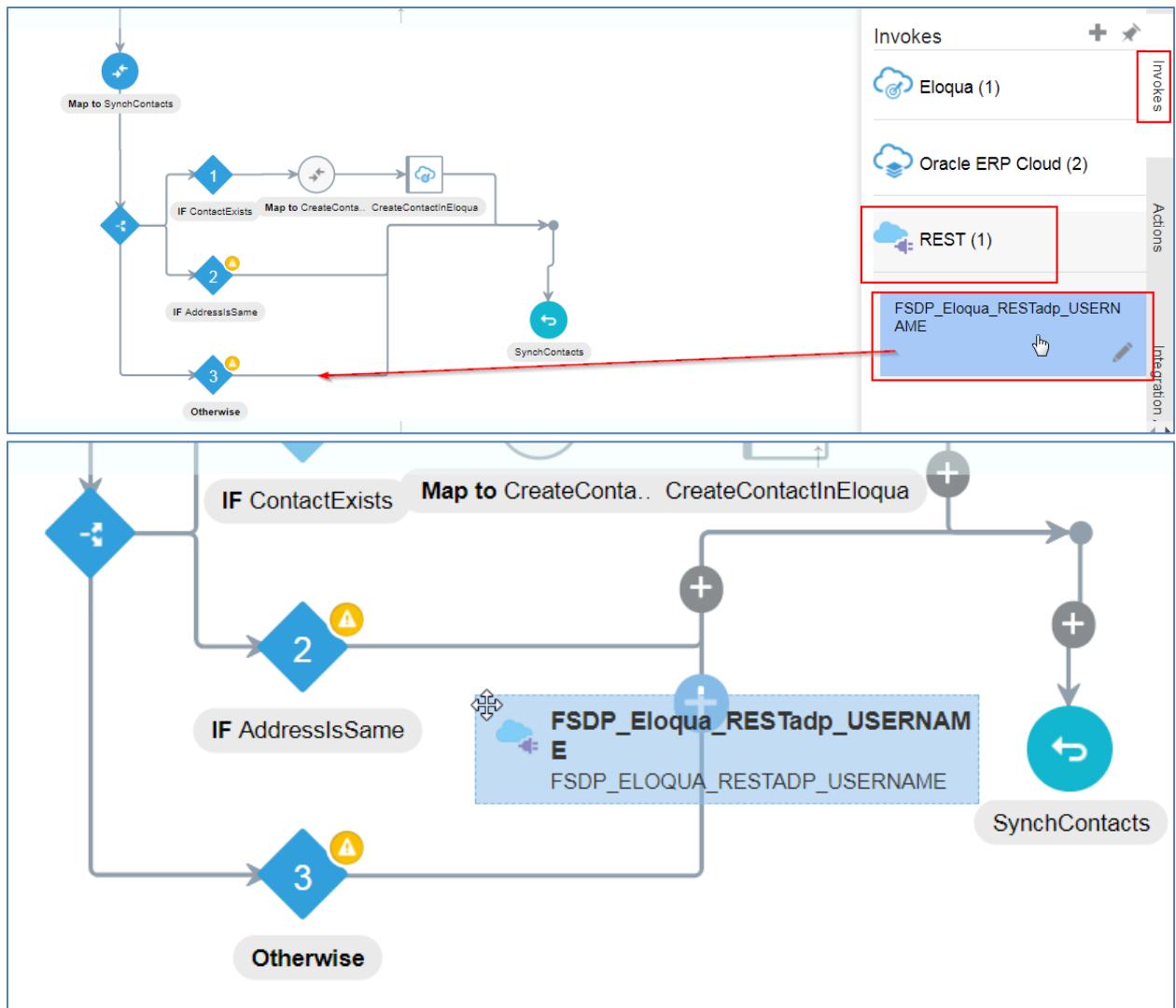
3. Select “**Contacts**” as “**Business Object**”.
4. Select **EmailAddress**, **FirstName**, **LastName** & **Address1** fields. (If any of these fields are pre-selected, you can ignore those).
5. Uniquely match with “**EmailAddress**”.



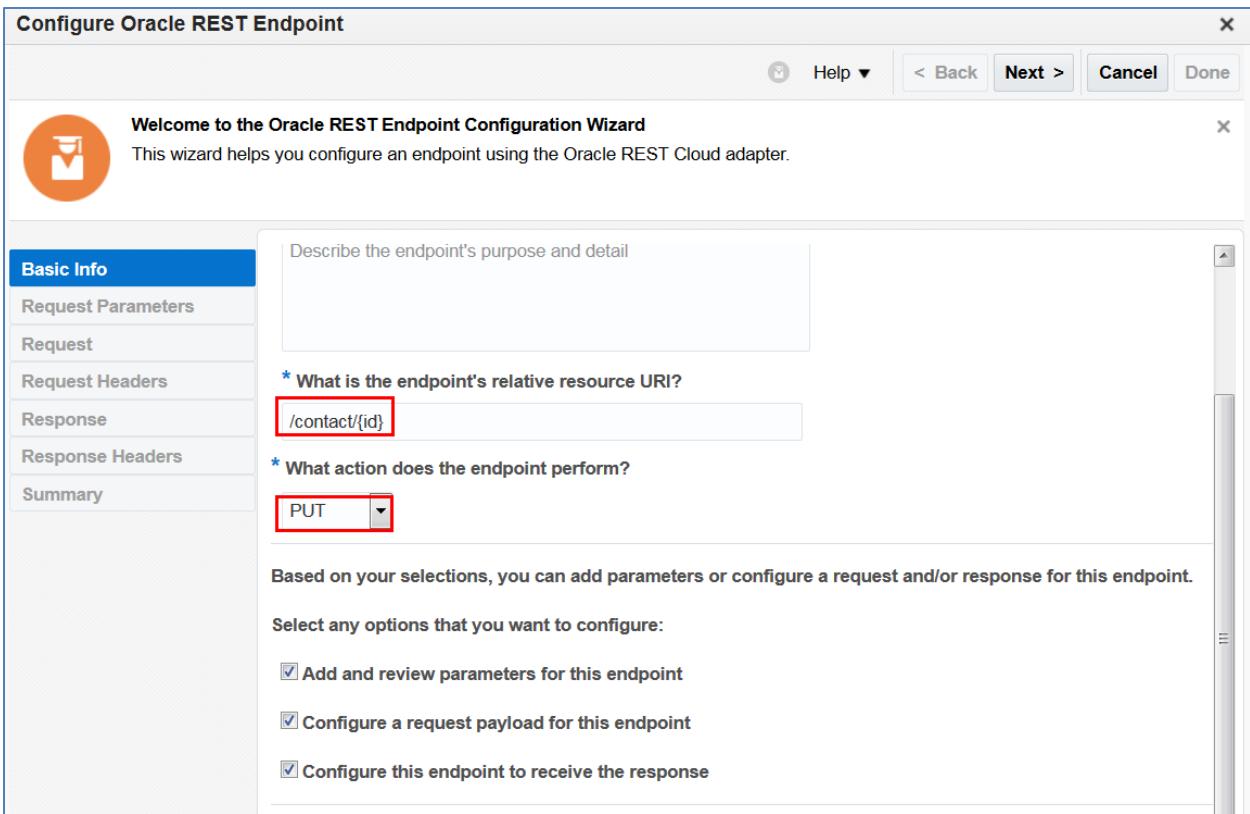
**Note:** You MAY see an error message similar to the one below (it may contain different field name). Just select whichever fields it complains (in this case it's "Personal email") to the mapping and move on.



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



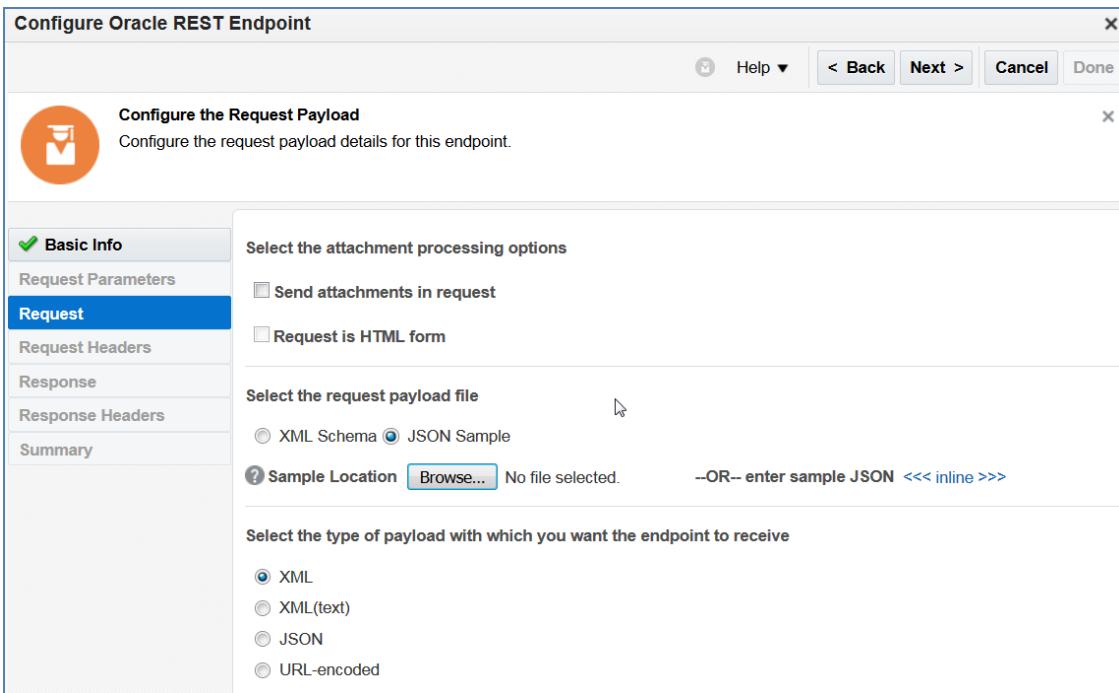
- In the config wizard, name it as “**UpdateEloquaAddress**”, use “**/contact/{id}**” as URI, select **PUT**, and select **Configure REQUEST and RESPONSE** option.



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).

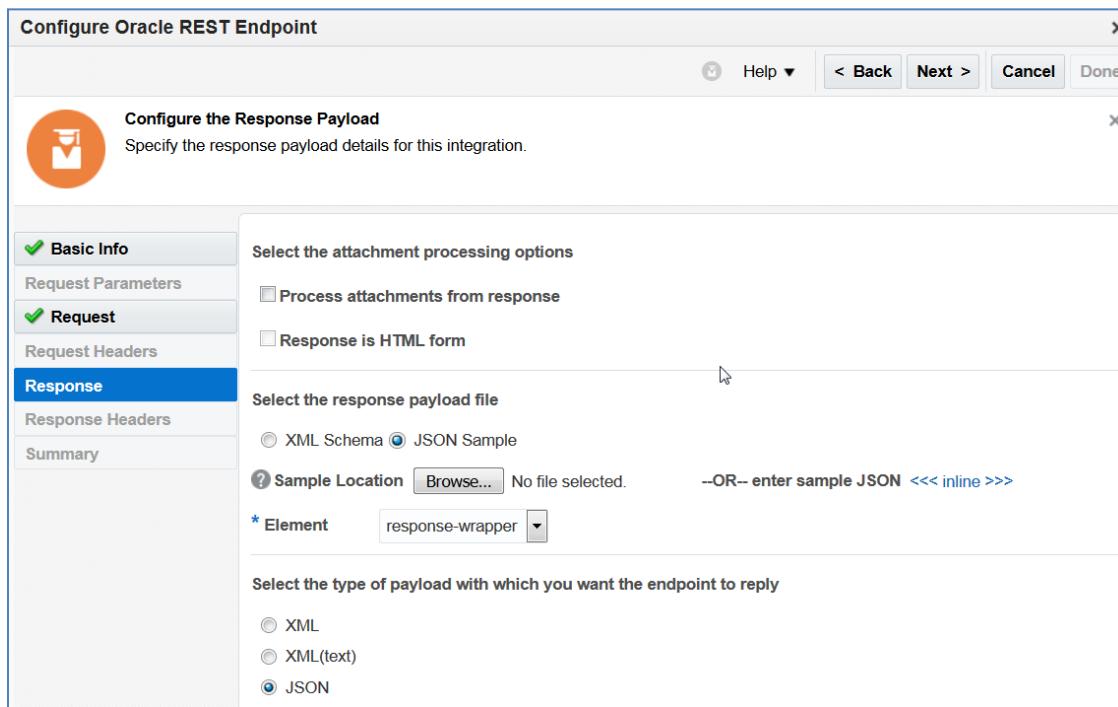


13. Click <<<inline >>> and ensure that you see the below payload. Click **cancel** on the bottom.

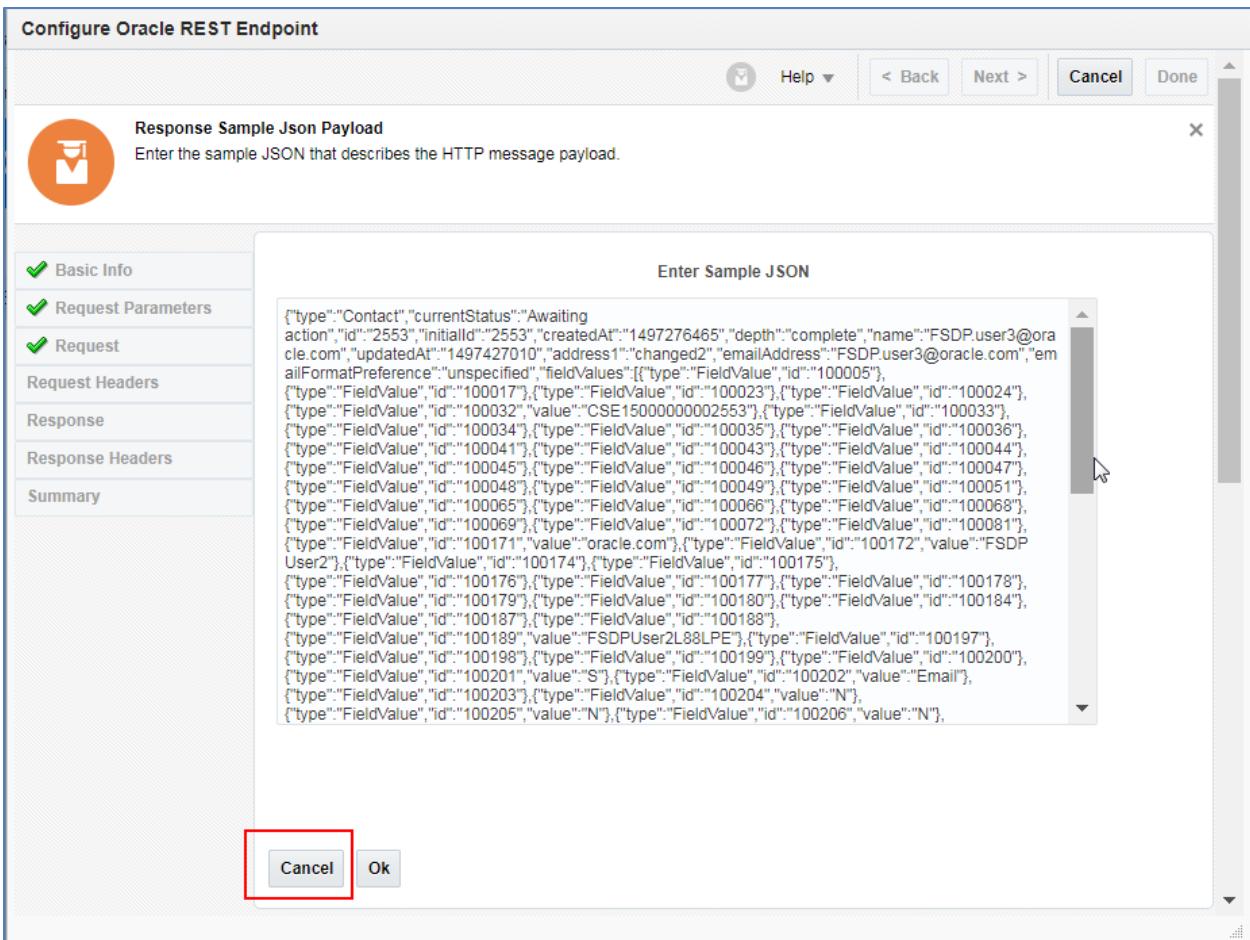
A screenshot of a modal dialog box. The main content area contains the JSON payload: { "emailAddress": "FSDP.user1@oracle.com", "id": "1234", "address1": "Oracle HQ" }. At the bottom of the dialog are two buttons: 'Cancel' and 'Ok'.

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

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

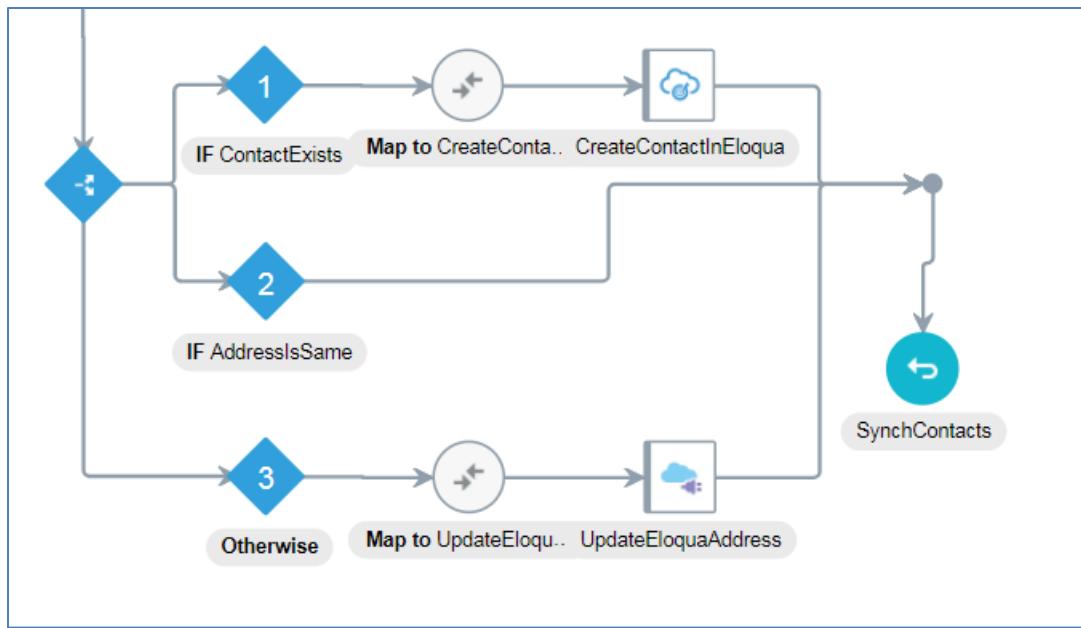


16. Click <<<inline >>> and ensure that you see the below payload. Click **cancel** on the bottom.



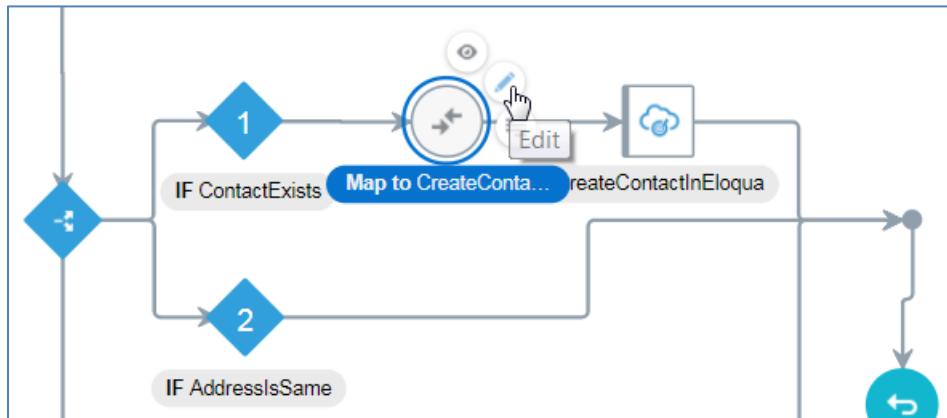
17. Click **Next** and review the summary. Click **Done**

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



## Configure Mapping for Create/Update actions

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

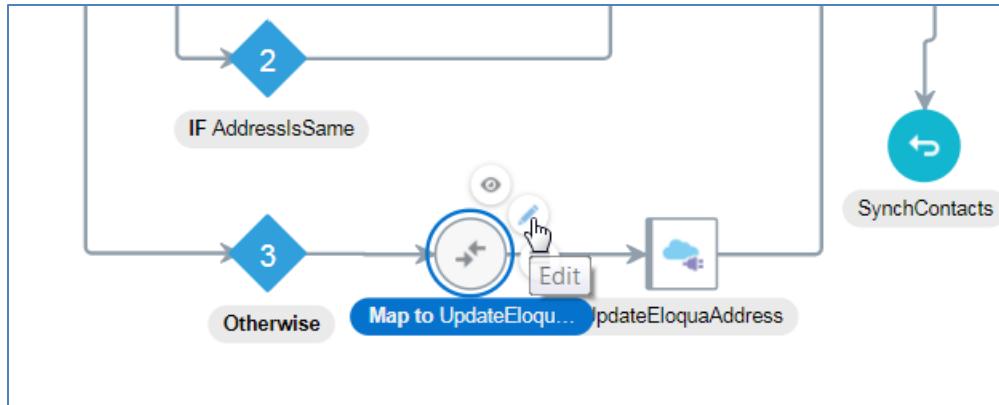


2. Map fields as below from execute.

Source	Target
execute	
*request-wrapper	
↳ *FirstName	↳ C_EmailAddress
↳ *LastName	↳ C_FirstName
↳ *Email	↳ C_LastName
↳ *SSN	
↳ *Address	↳ C_Address1
\$GetContactdetails	
*executeResponse	
*response-wrapper	

**Note:** Ignore any other fields.

3. Click **Validate** and **Close**
4. Map the update process  
Click “Map to **UpdateEloquaAddress**” mapping to edit it.



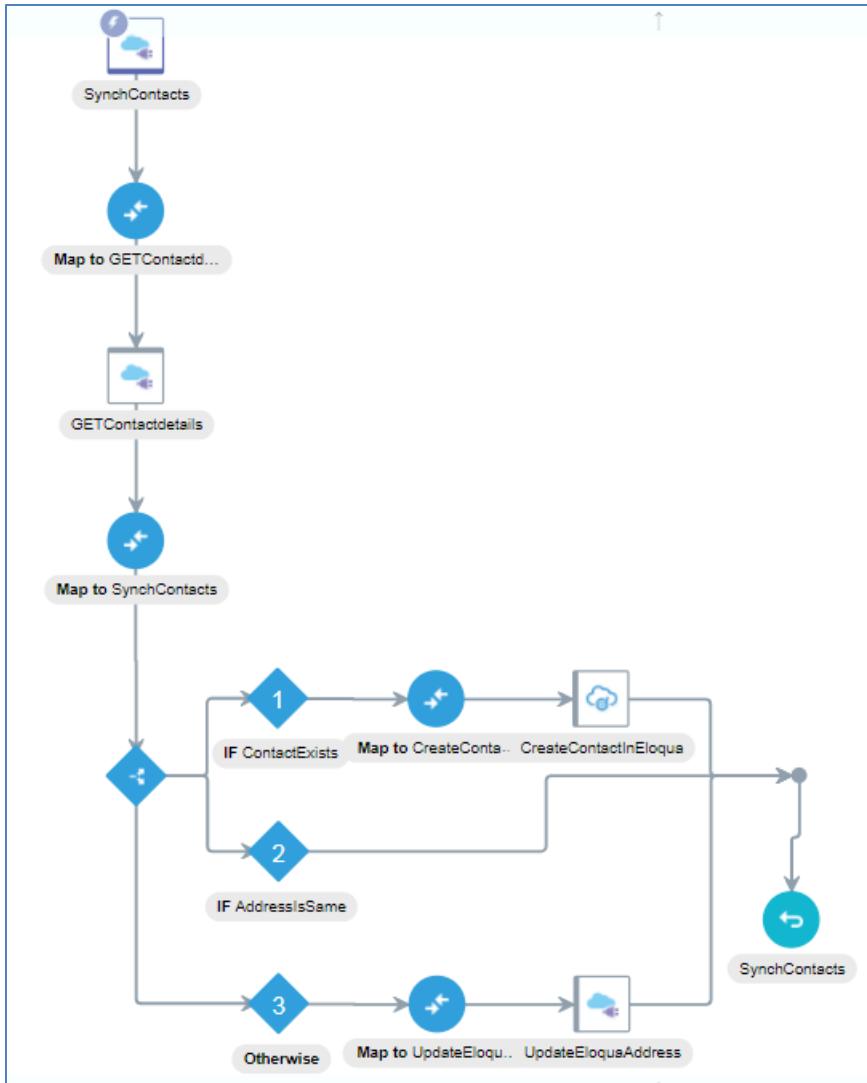
5. And do the mapping as shown below.

Source	Mappings	Target	Mapping
↳ *execute			
↳ *request-wrapper			
↳ *FirstName		↳ *TemplateParameters	
↳ *LastName		↳ *id	id
↳ *Email	✓	↳ *emailAddress	Email
↳ *SSN		↳ *id	id
↳ *Address	✓	↳ *address1	Address
↳ *elements		↳ *ConnectivityProperties	
↳ *type		↳ RestApi	
↳ *id	✓	↳ *Plugin	
↳ *createdAt			
↳ *depth			
↳ *name			
↳ *updatedAt			

**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.**

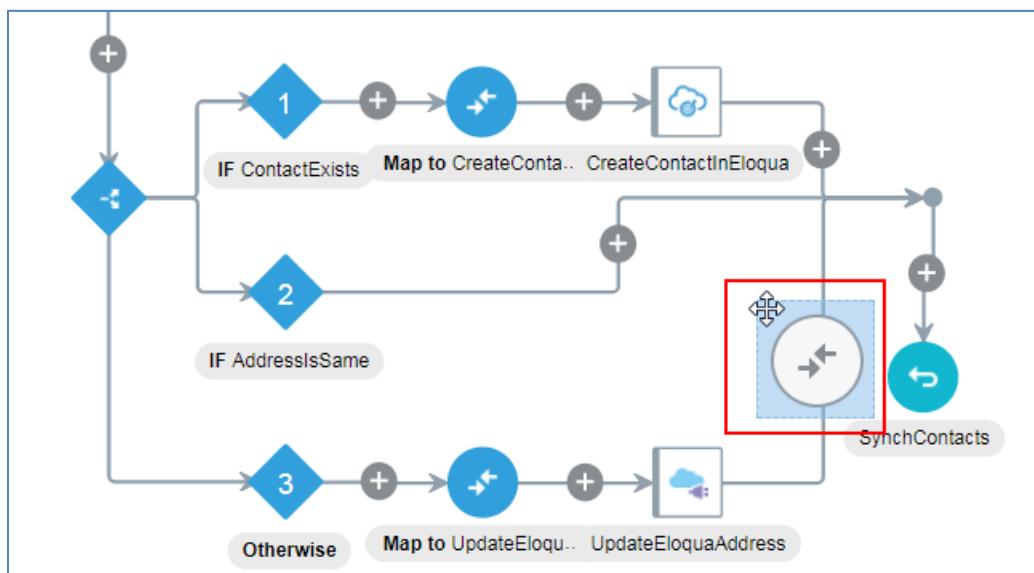
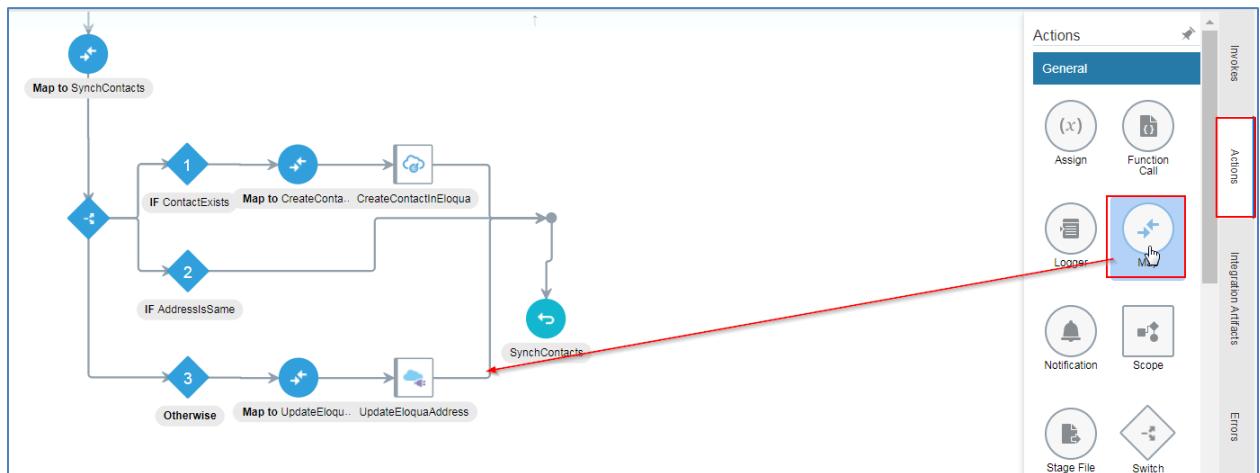
6. **Validate and Close**  
 7. **Save** the integration.  
 The Integration should look like below



## 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.

- 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)

The screenshot shows the MuleSoft Anypoint Studio Mapper interface with two panes: Source and Target. The Source pane lists fields like \$UpdateEloquaAddress, executeResponse, response-wrapper, type, currentStatus, id, initialId, createdAt, depth, name, updatedAt, address1, and emailAddress. The Target pane lists executeResponse, response-wrapper, elements, id, emailAddress, and address1. A tooltip indicates: "Drag and drop source to target to create a mapping." and "Click a checkbox on source or target to see mappings." Red arrows point from the Source fields (id, address1, emailAddress) to their corresponding Target field checkboxes (id, address1, emailAddress).

This screenshot shows the same Mapper interface as above, but with a new row in the Target pane: a checkbox labeled "total". A red box highlights this "total" checkbox. A green arrow points from the "address1" field in the Source pane to the "total" checkbox in the Target pane, indicating it is being mapped to the total field.

## Build Mappings

**Source**

View ▾ Filter Detach Map

Find...

- ▲ <> \*execute
  - ▲ <> \*request-wrapper
    - <> \*FirstName
    - <> \*LastName
    - <> \*Email
    - <> \*SSN
    - <> \*Address
  - ▲ <> \$GETContactdetails
    - ▲ <> \*executeResponse
      - ▶ <> \*response-wrapper
    - ▲ <> \$UpdateEloquaAddress
      - ▲ <> \*executeResponse
        - ▶ <> \*response-wrapper
  - <> \$tracking\_var\_1
  - <> \$tracking\_var\_2
- ▶ Mapping Components

**Mapping**

Target Element: /executeResponse/response-wrapper/total

**Statement**

▲ <> <nsmpr4:total>

100

View ▾ Filter Detach

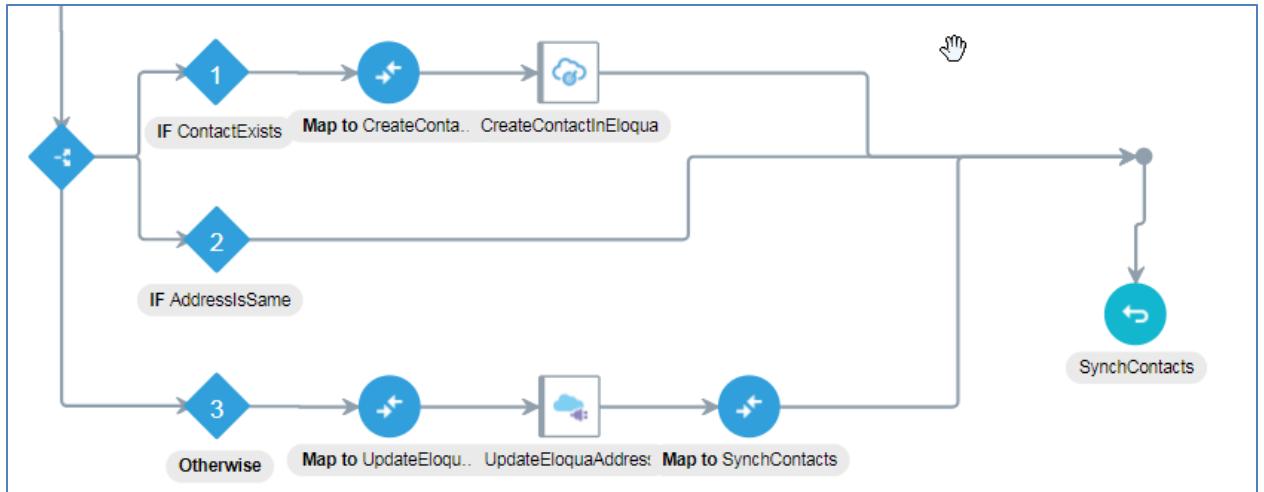
**Target**

Find...

**Mapping**

- ▲ <> \*executeResponse
- ▲ <> \*response-wrapper
- ▶ <> \*elements
- <> \*page
- <> \*pageSize
- ✓ <> \*total "100"

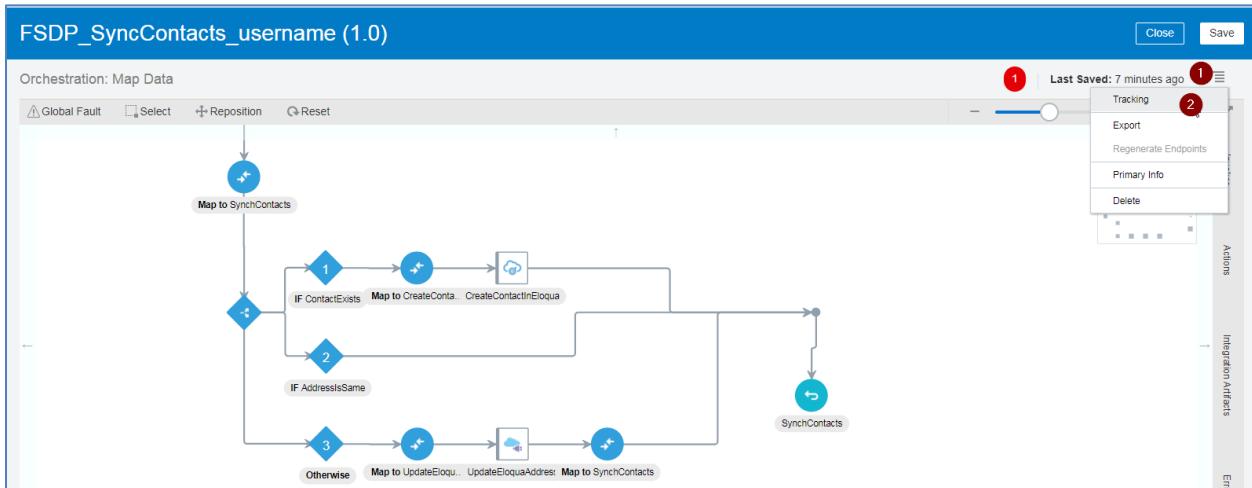
3. Click “**Validate**” and “**Close**”. Now, Integration should look like below.



Click “Save”.

## Add Tracking

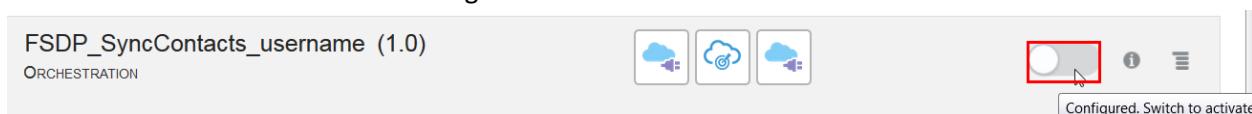
- On the Top-right of designer, Click “Tracking” button on Actions menu.



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

Primary	Tracking Field	Tracking Name	Tracking Variable	Help Text
<input checked="" type="checkbox"/>	Email	Email	tracking_var_1	How to track it?
	Drag a trigger field here	tracking_var_2	tracking_var_2	How to track it?
	Drag a trigger field here	tracking_var_3	tracking_var_3	How to track it?

- Click “Save” (at the bottom)
- “Save” the integration (one last time, hopefully ☺)
- Click “Close” !
- Click on the switch to Activate the integration.



7. Click the checkbox to enable tracing and click “Activate”.

**Activate Integration**

**FSDP\_SyncContacts\_username (1.0)**

**Publish:** This integration can be published to API Platform CS.

**⚠️** API Platform CS connection is not configured. Please navigate to Setting/API Platform CS page to configure the connection.

[Learn More](#)

**Recommendations Engine**

Contribute integration mappings to Oracle Recommendations Engine.

**ⓘ** Oracle Integration Cloud leverages the collective intelligence to recommend which fields should be mapped while developing an integration. These recommendations are built based on the mappings contributed to Oracle Recommendations Engine anonymously. Unselect the checkbox if you do not wish to contribute the mappings. You may change this in recommendations page from settings menu.

[Learn More](#)

**Tracing:** When tracing is enabled, integration activity can be viewed in the Activity Stream.

Enable tracing

Include payload

**⚠️** When payload is included, sensitive information from the payload is written into log files, which can be downloaded and viewed. This may pose a security risk, and also slow down your system. Not recommended in a production environment.

[Learn More](#)

[Activate](#) [Cancel](#)

8. You may see the below message.

Integration FSDP\_SyncContacts\_username (1.0) submitted for activation. Click refresh if status is in progress.  
- Use this endpoint <https://> to trigger this integration, after activation succeeds.  
- You can also go to [Tracking](#) page to track instances.

[Import](#) [Create](#)

**Integrations**

Feb 16, 2018 2:48:32 PM UTC	Search	Last Updated
FSDP_SyncContacts_username (1.0) ORCHESTRATION: MAP DATA		

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

FS Digital Platform Information_forTesting.xlsx - Excel	
File	Home
Paste	Calibri 11 A A Wrap Text
Clipboard	Font Alignment Number Styles Cells Editing
B6	A B
1	
2 IssueCard	Enter your OIC IssueCard Integration URL here
3 CardRecommendation	Enter your OIC CardRecommendation Integration URL here
4 SynchContacts	https://<OIC HOSTNAME>/ic/api/integration/v1/flows/rest/FSDP_SYNCCONTACTS_USERNAME1/1.0/metadata
5 ApprovalRequest	Enter your OIC Process Application WSDL URL here
6	
7	
8	
9	
10	
11	
12	
13	
SAMPLE Data (DC)	
https://<OIC HOSTNAME>	
https://<OIC HOSTNAME>	
https://<OIC HOSTNAME>	
https://<OIC hostnam	
(Ctrl) ▾	
Ready	
Environment Information   Lab 1.1-APIPCS   Lab2.3-IssueCard   <b>OIC Endpoints</b>   Yor ...	
Import Create	

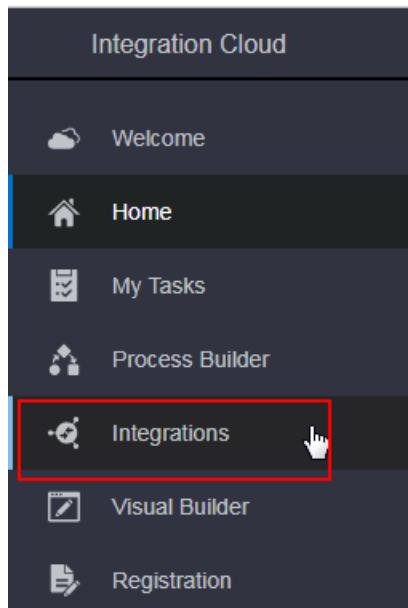
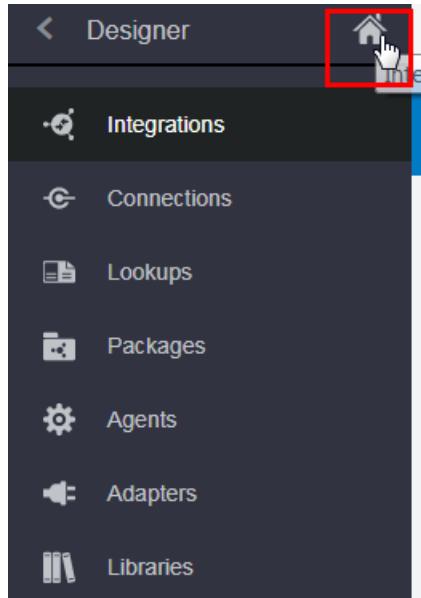
10. Wait until you see it activated.

The screenshot shows the 'Integrations' page in the Oracle Integration Cloud Service. It lists three active integrations:

- FSDP\_SyncContacts\_username (1.0) - Status: NEW, Template: MAP DATA. The 'TRACE ENABLED' switch is turned on (green).
- AnandTestEloqua (1.0) - Status: NEW, Template: UNDEFINED. The 'TRACE ENABLED' switch is off (grey).
- PTFE200\_ERPCLOUD\_CREATEORDER (1.0) - Status: NEW, Template: MAP DATA. The 'TRACE ENABLED' switch is off (grey).

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.



Integrations

Feb 16, 2018 3:02:11 PM UTC

Search Last Updated

FSDP\_SyncContacts\_username (1.0)  
ORCHESTRATION: MAP DATA

AnandTestEloqua (1.0)  
TEMPLATE: UNDEFINED

PTFE200\_ERPCLOUD\_CREATEORDER (1.0)  
TEMPLATE: MAP DATA

Import Create

TRACE ENABLED

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.

How to run

Endpoint URL:

https://1.oraclecloud.com:443/ic/api/integration/v1/flows/test/FSDP\_SYNCCONTACT\_USERNAME/1.0/metadata

How to run

Track Instances

Import

Last Upda

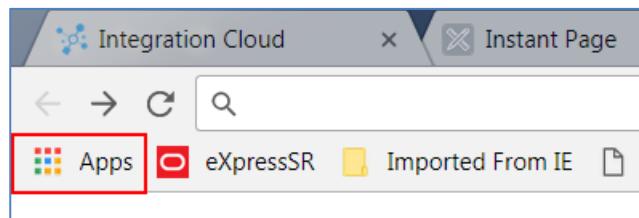
TRACE ENABLED

## Testing

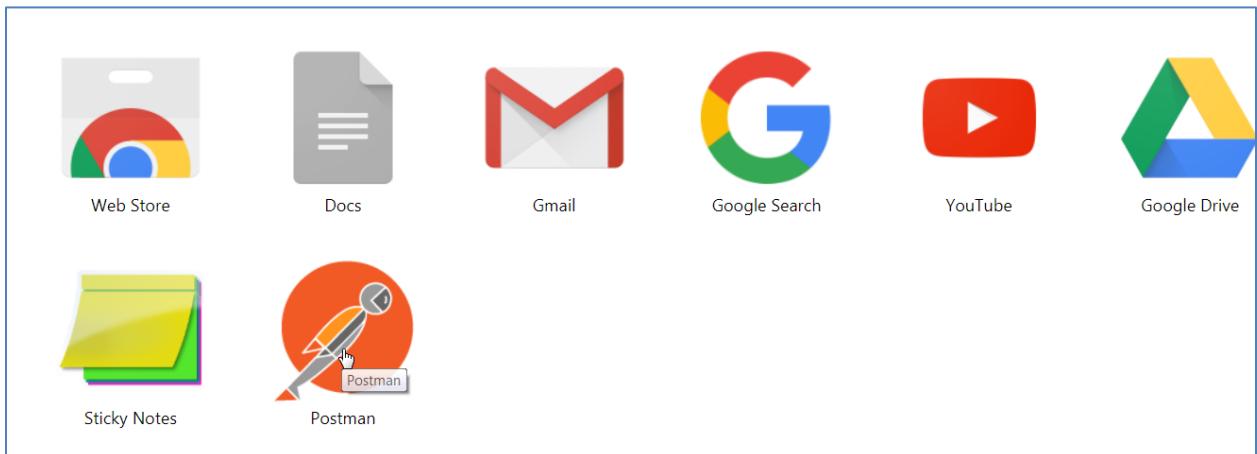
### *Create and configure Synch Contacts Integration*

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**.

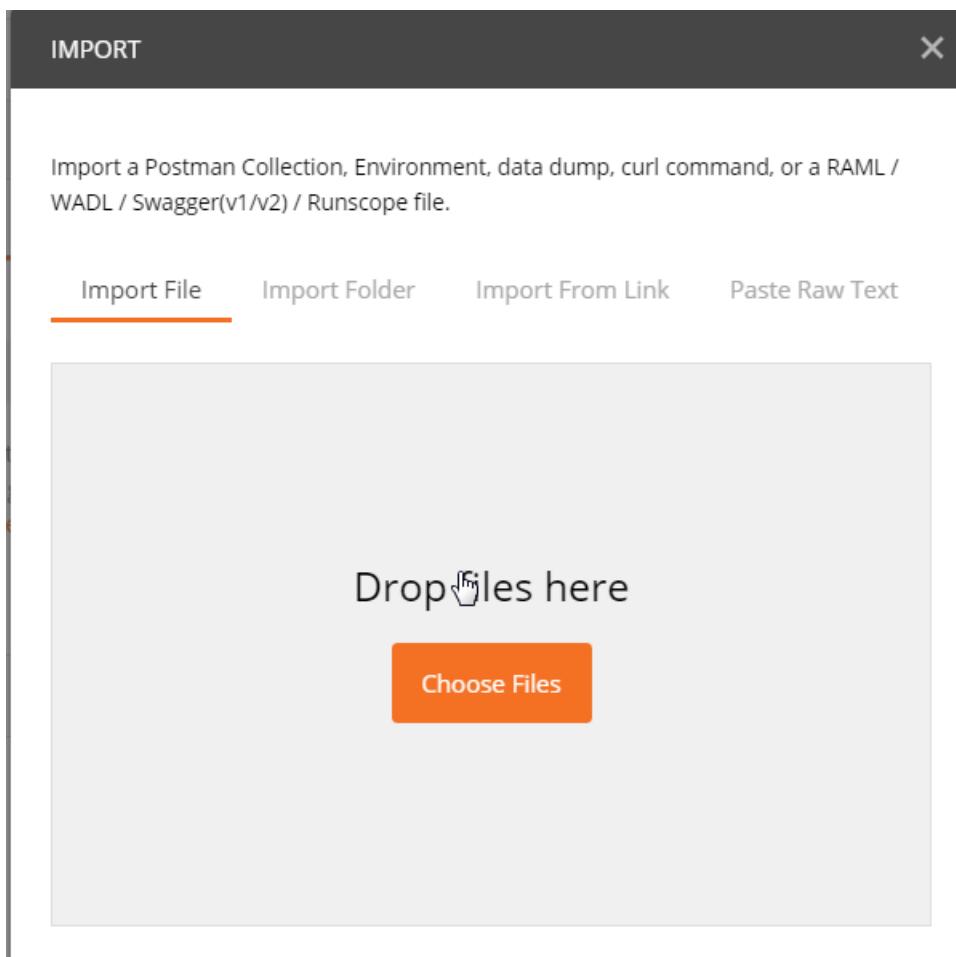
You may use POSTMAN standalone or choose your own... Below steps are for POSTMAN Chrome extension.



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. **Import**...



4. Choose “Digital Transformation Workshop.postman\_collection.json” file shared by instructor.

Collection Digital Transformation Workshop imported

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



**POST** Lab2.1-SYNCCONTACTS

6. Copy the **SynchContacts** integration URL from “**FS Digital Platform.xls**”->“**OIC Endpoints**”  
[https://<domain-name>/ic/api/integration/v1/flows/rest/FSDP\\_SYNCCONTACT\\_USERNAME/1.0/metadata](https://<domain-name>/ic/api/integration/v1/flows/rest/FSDP_SYNCCONTACT_USERNAME/1.0/metadata).

**Replace** the /metadata with /contacts in the URL

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

The screenshot shows the Postman interface with a collection named "SynchContacts". A POST request is selected with the URL `https://1.0.oraclecloud.com:443/ic/api/integration/v1/flows/rest/FSDP_SYNCCONTACT_USERNAME/1.0/contacts`. The "Authorization" tab is highlighted with a red box, and the "Basic Auth" option is selected within it.

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

The screenshot shows the "Lab2.1-SYNCCONTACTS" collection in Postman. The "Authorization" tab is selected with a red box. Under the "Type" dropdown, "Basic Auth" is selected. The "Username" and "Password" fields are also highlighted with red boxes.

8. Click “Body” and verify the details. Enter the below if required.  

```
{ "FirstName":"FSDP", "LastName":"User2", "Email":"FSDP.user30@oracleleads.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.

The screenshot shows the Postman interface with a POST request to `https://Integration-gse00011310.integration.us2.oraclecloud.com/integration/flowapi/rest/FSDP_SYNCCONTACTS`. The Authorization token is set to `_USERNAME/v01/contacts`. The request body is a JSON object:

```
{
  "FirstName": "FSDP",
  "LastName": "User30",
  "Email": "FSDPUser30@oracleleads.com",
  "SSN": "1234",
  "Address": "Oracle HQ"
}
```

The response status is 200 OK with a time of 6452 ms. The response body is:

```
{
  "elements": [
    {
      "type": "",
      "id": "",
      "name": "",
      "emailAddress": "",
      "address1": ""
    }
  ],
  "total": 0
}
```

- 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.

The screenshot shows the Postman interface with a GET request to the same endpoint. The Authorization token is set to `_USERNAME/v01/contacts`. The Headers tab shows two entries: `(2)` and `Body`. The Body tab contains the same JSON object as the previous screenshot.

The response status is 200 OK with a time of 2291 ms. The response body is:

```
{
  "elements": [
    {
      "type": "Contact",
      "id": "2558",
      "name": "FSDPUser30@oracleleads.com",
      "emailAddress": "FSDPUser30@oracleleads.com",
      "address1": "Oracle HQ"
    }
  ],
  "total": 1
}
```

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

The screenshot shows a Postman request configuration for a POST method to the URL <https://Integration-gse00011310.integration.us2.oraclecloud.com/integration/flowapi/rest/>. The 'Body' tab is selected, showing raw JSON data:

```
{ "FirstName": "FSDP", "LastName": "User30", "Email": "FSDPUser30@oracleleads.com", "SSN": "1234", "Address": "Oracle HQ-changed" }
```

The response status is 200 OK with a time of 4932 ms. The response body is:

```
{ "elements": [ { "id": "2558", "emailAddress": "FSDPUser30@oracleleads.com", "address1": "Oracle HQ-changed" } ], "total": 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.

The screenshot shows a Postman request configuration for a POST method to the same URL. The 'Body' tab is selected, showing raw JSON data with changes to FirstName and LastName:

```
{ "FirstName": "FSDP-changed", "LastName": "User-changed", "Email": "FSDPUser30@oracleleads.com", "SSN": "1234", "Address": "Oracle HQ-changed" }
```

The response status is 200 OK with a time of 2926 ms. The response body is:

```
{ "elements": [ { "type": "Contact", "id": "2558", "name": "FSDPUser30@oracleleads.com", "emailAddress": "FSDPUser30@oracleleads.com", "address1": "Oracle HQ-changed" } ], "total": 1 }
```