CONFIGURING, DEPLOYING AND MONITORING WEBUI BASED SCENARIO USING SAP CLOUD PLATFORM INTEGRATION

Exercise00: SAP Cloud Platform Integration Components Used:

- Timer
- Content Modifier
- Request-Reply
- SOAP Receiver Adapter
- Router
- Data Store Operations - Write
- Groovy script

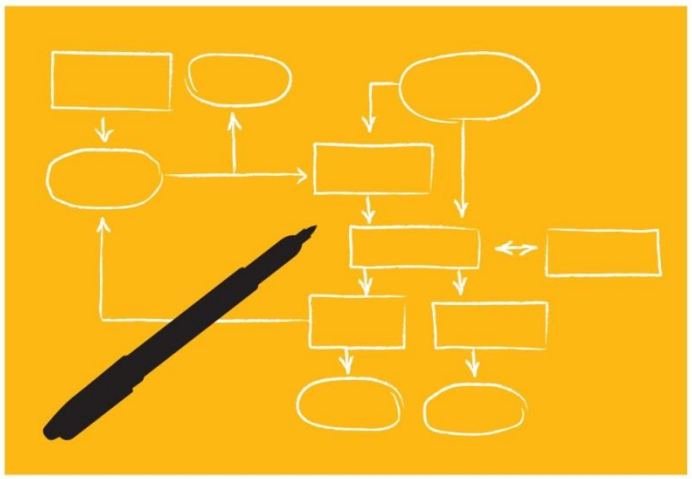
INTEGRATION SCENARIO

This initial exercise is used to perform the smoke test for SAP Cloud Platform Integration Web UI.

Use Case: In this scenario we will call a third-party SOAP based free web service to get the weather data of a city.
Based on the fact whether we get the valid response or not, message routes to two different paths, one would be default and one for valid response.
We also write the weather service response to Data Store and log the response as attachment as well.

In case, third party service is not available we need to change the service endpoint to mocked weather service which is deployed on the CPI tenant for you.

Welcome to the Integration Flow challenge!



In this exercise, your aim is to create an integration flow that solves a challenge *(described in the Integration scenario)*.

And when you work your way through the exercise, our aim is that you learn:

1. How to access and work with the Cloud Platform Integration Web tooling
2. How to speed up your integration project by leveraging reference integration flows
3. How to customize a reference integration flow by configuring its connectivity and flow steps
4. Basic monitoring of an integration flow
5. Using Open-source tools to test your integration flow

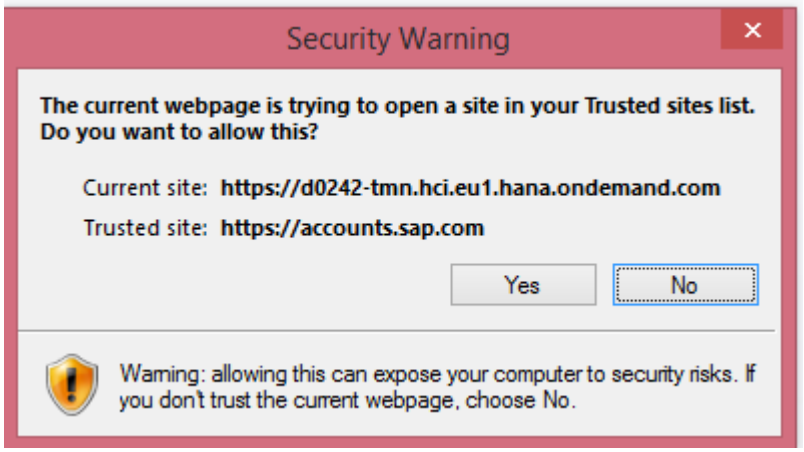
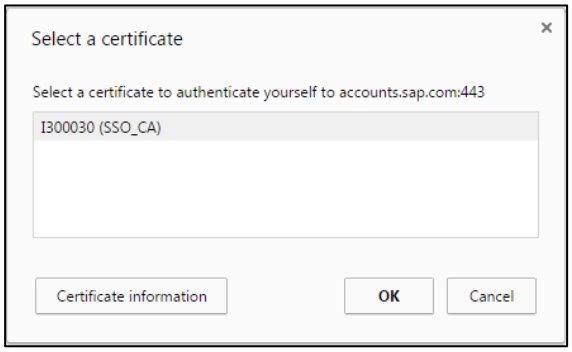
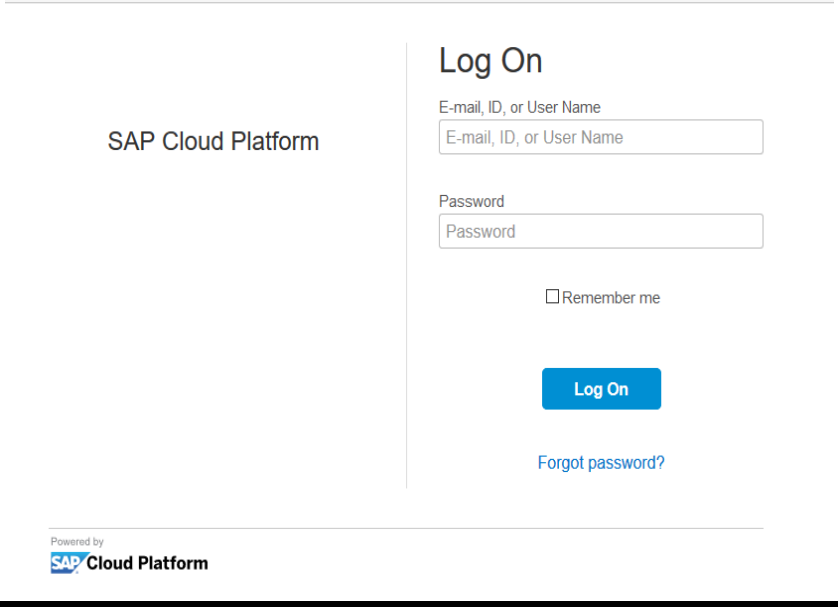
Exercise Files

Exercise Files are provided by the session moderator. Download and Extract them into one of your local folders for use later in your exercises and setup:

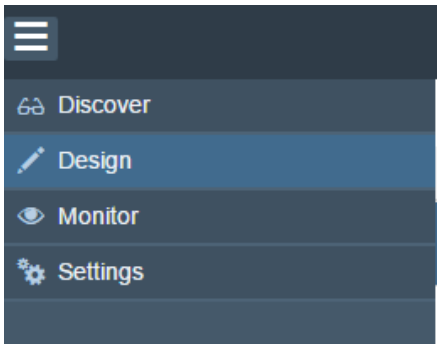
Note

1. In the exercise, we have used the notation XX to refer to the content created by you or your group uniquely. The session moderator shall assist you with assigning you/group the unique ID that you can use to replace XX.
2. In the exercise, we have used ***d0242/d0243*** as the example tenants. The final tenant details shall be provided by the session moderator.
3. Please note that colours and other visual appearance might differ slightly from the screenshot screens, as the CPI editor might have received feature upgrades since production of this content.

EXERCISE 00

Explanation	Screenshot
Follow these steps for integration flow creation in Web UI.	
1) Access the Web application using Google Chrome (or IE10 Browser) a. Open Google Chrome b. Copy the application URL or **The URL that you can use will be assigned to you by the instructor.** c. You will get a security warning if you want to open this site in your trusted sites list. Accept it with 'Yes'.	
2) If you get displayed your Certificate List, click on 'Cancel'	
3) You will now see the log-on screen for the SAP ID Service/SAP Cloud Platform. Enter the following credentials: *User:* <Provided by instructor> *Password*: <Provided by instructor>	

4) Now you will have access to the Web application. Familiarize yourself with the environment and choose the ***Design*** area.
Here, you will create an Integration Package to store your iFlows for different Exercises

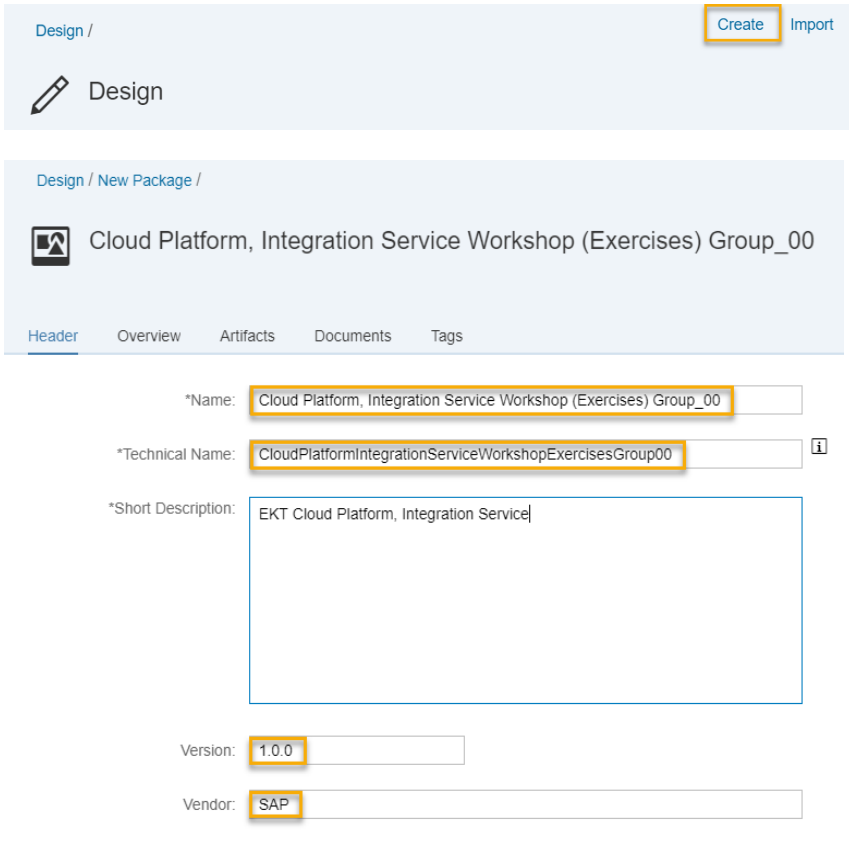


In the left-hand navigation, you can switch between the following sections of the Web application:

- **Discover** → SAP's Reference Catalog
- **Design** → Your customer Workspace (Design Time Content)
- **Monitor** → Message Monitoring and deployed integration flows

Hint: Use icon ☰ to hide the first level navigation menu

Click on **Create**



Enter Details

a) **Name:** Cloud Platform, Integration Service Workshop (Exercises) Group_XX

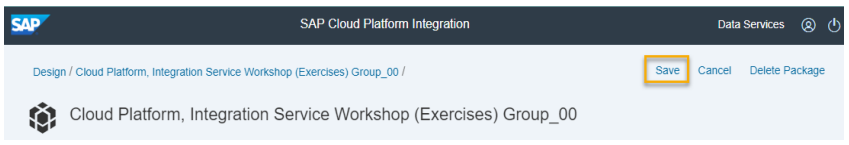
NOTE: Replace XX with the group number provided by the instructor

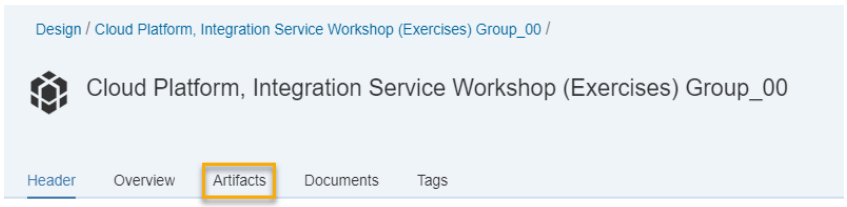
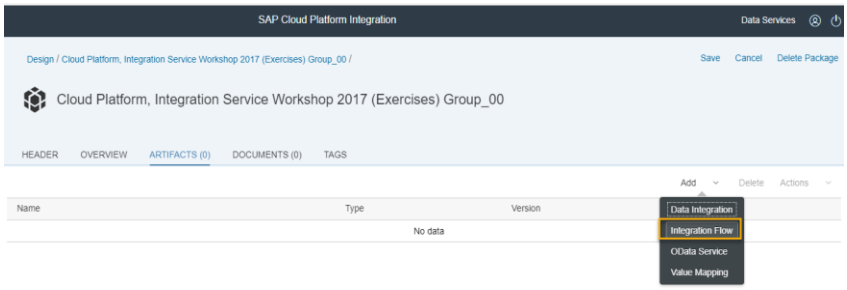
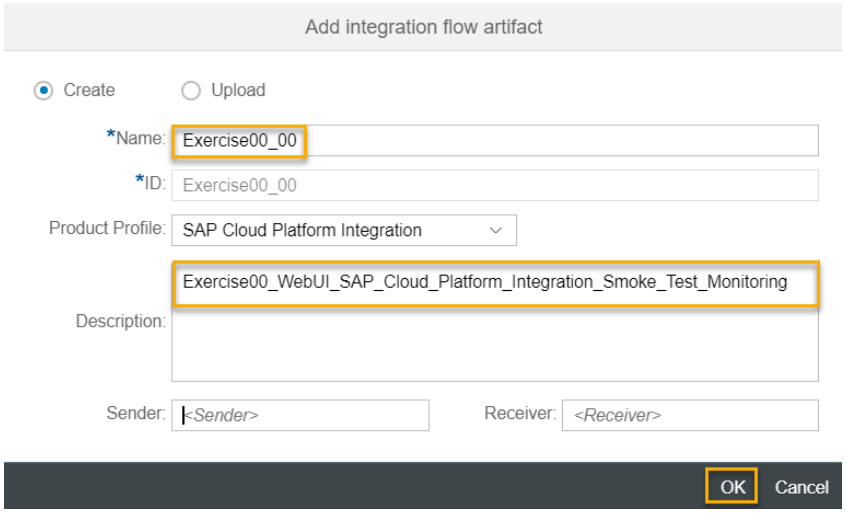
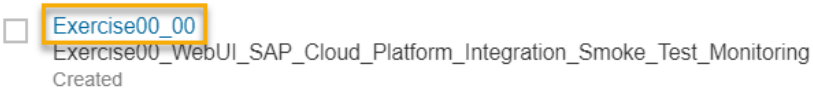
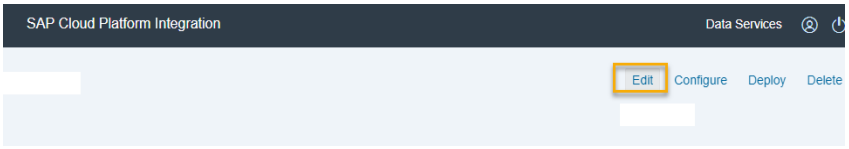
b) **Short Description:** EKT Cloud Platform, Integration Service

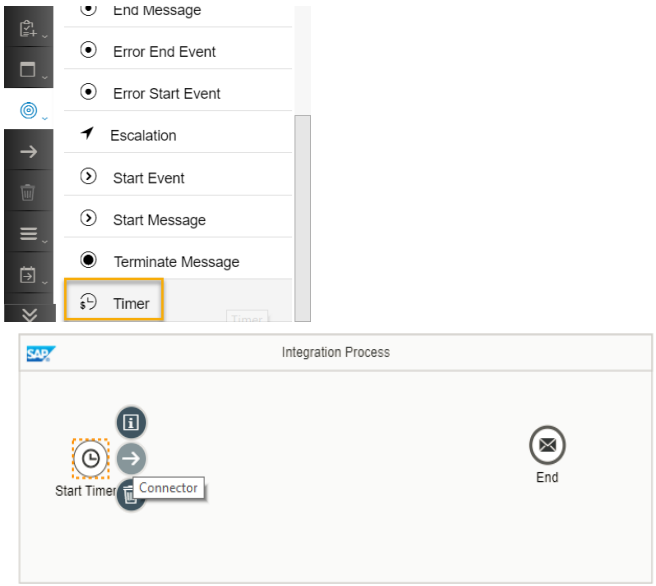
c) **Version:** 1.0.0

d) **Vendor:** SAP

5) Click on Save

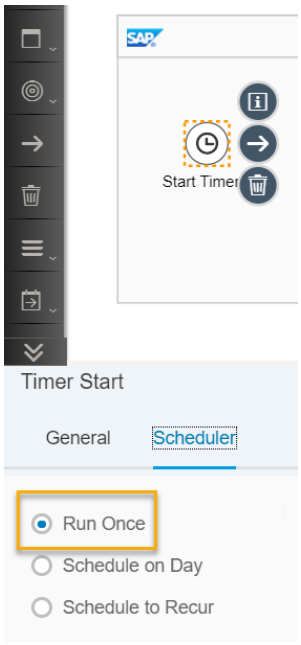


6) Navigate to **Artifacts** Tab	
7) Click on ***Add-> Integration Flow***	
8) Select ***Create*** and enter the following details: *a.* Name: ***Exercise00_XX*** *b.* Description: ***Exercise00_WebUI_SAP_Cloud_Platform_Integration_Smoke_Test_Monitoring*** c. Click on ***OK*** **Note:** Please remember to save after every action	
9) Click on ***Exercise00_XX***	
Define and edit integration flow	**Follow steps to edit the integration flow**
10) Click on ***Edit*** button on the upper corner in the right-hand side.	

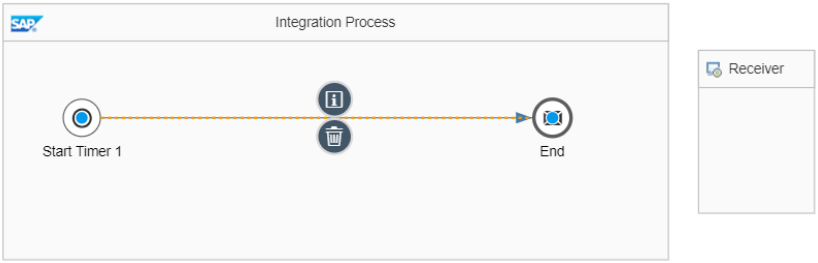
11) Click on Sender system and click the ***Delete*** icon to delete the system. Similarly, click on ***Start*** message and click ***Delete*** icon.	
12) Integration Flow should appear like this	
13) From the palette, select ***Events -> Timer*** and drop it to Integration Process	

14) Switch to ***Scheduler*** tab and set it to **Run Once.**

This would ensure that iFlow is executed as soon as it is deployed



15) Connect ***Timer*** to ***End Message*** event

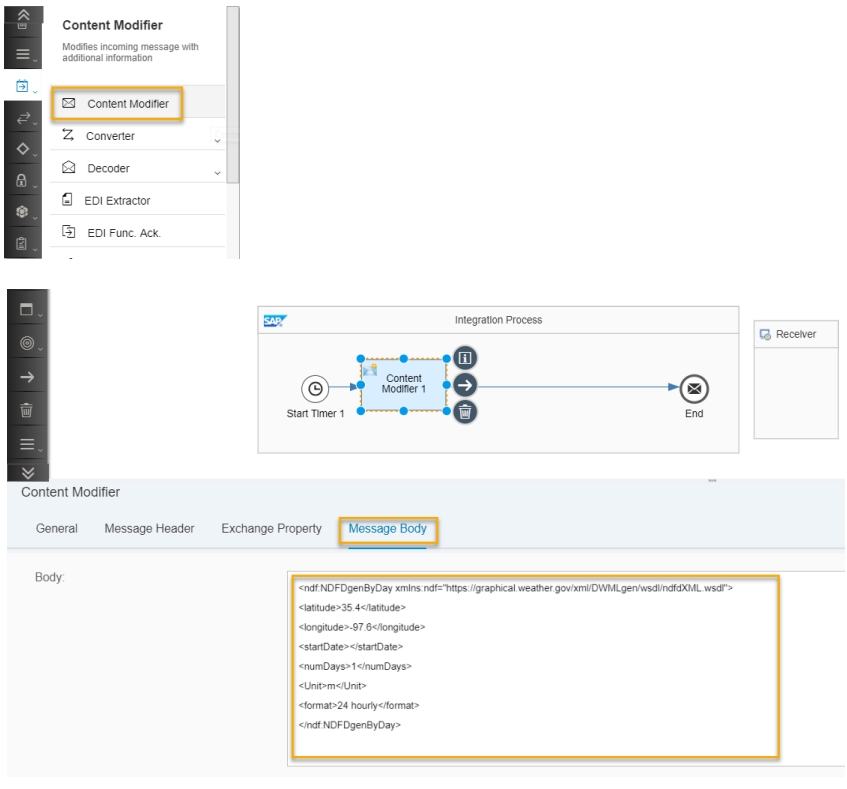


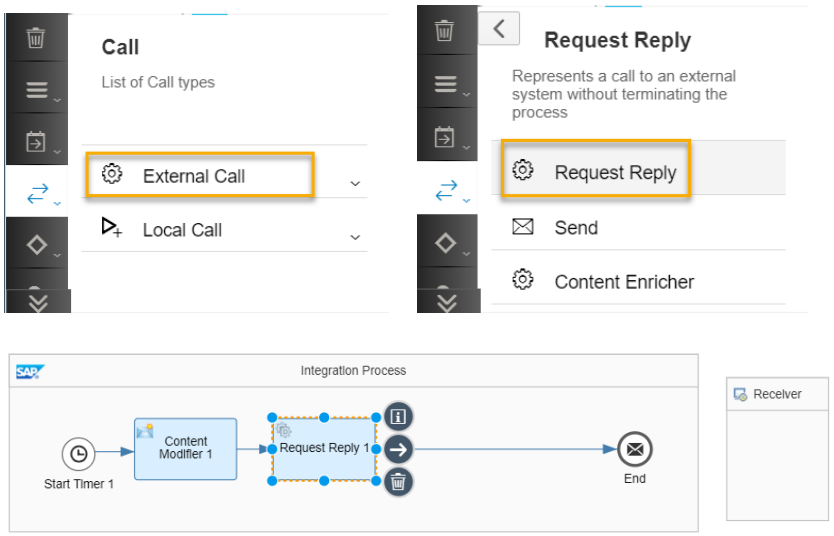
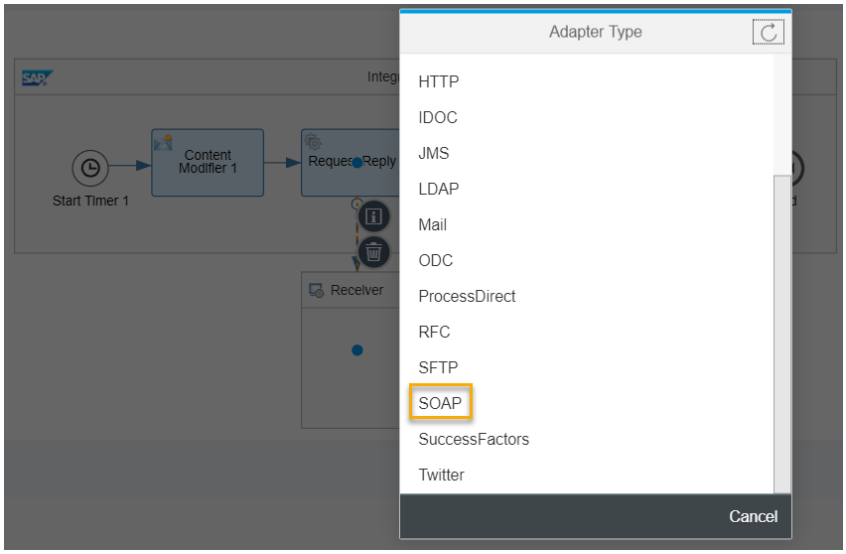
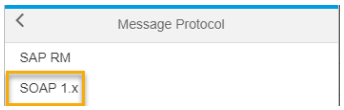
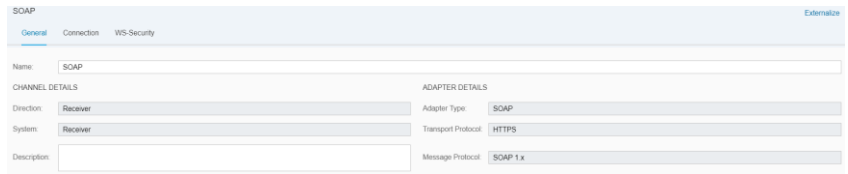
16) From the palette, select ***Message Transformers -> Content Modifier*** and drop it on the connection between ***Timer*** and ***End*** message in the Integration flow. This would automatically create the connections.

Switch to ***Message Body*** tab, paste the content given below:

<ndf:NDFDgenByDay xmlns:ndf="https://graphical.weather.gov/xml/DWMLgen/wsdl/ndfdXML.wsdl">
<latitude>35.4</latitude>
<longitude>-97.6</longitude>
<startDate></startDate>
<numDays>1</numDays>
<Unit>m</Unit>
<format>24 hourly</format>
</ndf:NDFDgenByDay>

Save



Note: This latitude and longitude is of **Oklahoma City**.	
17) From the palette, select ***Call -> External Call -> Request Reply*** and drop it on the connection between ***Content Modifier*** and ***End*** message in the Integration flow.	
18) Click on ***Request Reply*** step, select ***Connector*** and drag it to ***Receiver*** system. (You might drag Receiver under Request-Reply for better readability) This would open the receiver adapter list. Select ***SOAP*** adapter	
Select ***SOAP 1.x*** as Message Protocol	 

19) Switch to ***Connection*** Tab, enter the following details:

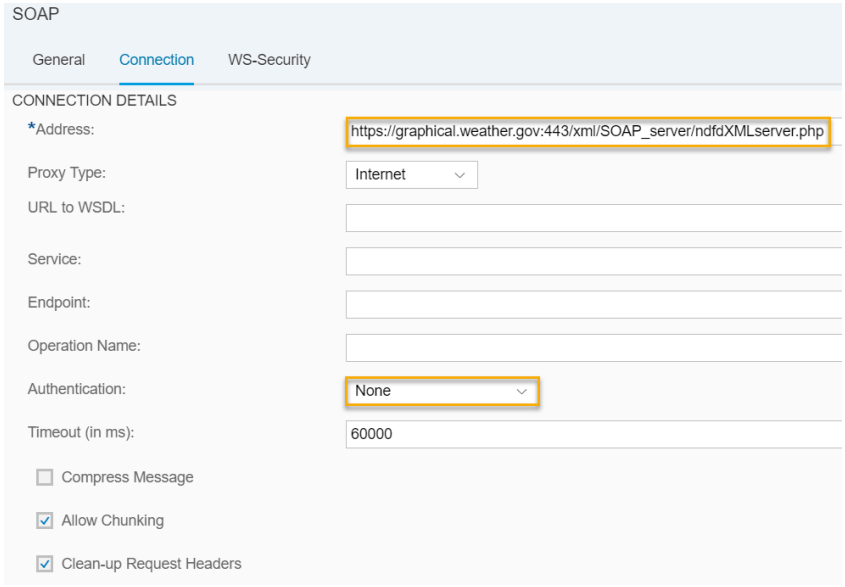
a. Address: ***https://graphical.weather.gov/xml/SOAP_server/ndfdXMLserver.php***
b. Select Authentication as **"None"**

Keep all other parameters as it is.

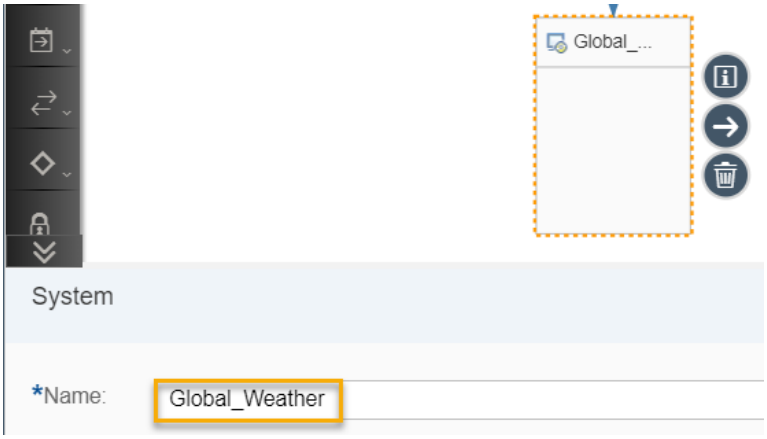
Save

Note: In case, this service is not available we need to change the service endpoint to mocked weather service which is deployed on the CPI tenant for you. Also, you need to change the Authentication to Basic

Instructor will provide you the SOAP Endpoint of mocked weather service.



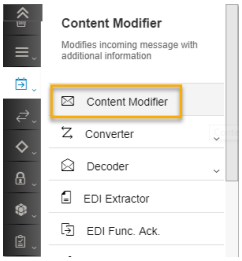
20) Click on Receiver system and set Name as ***Global_Weather***

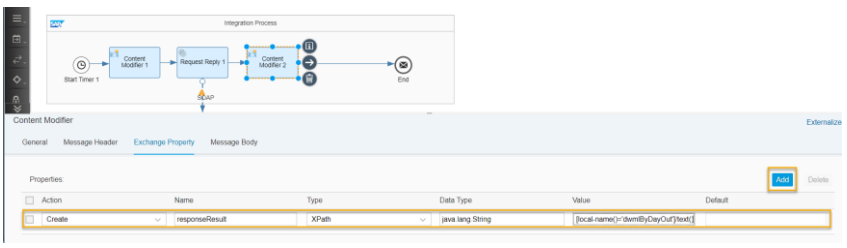


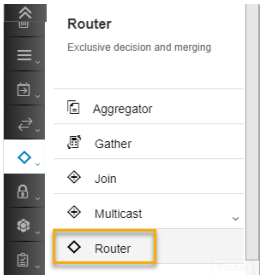
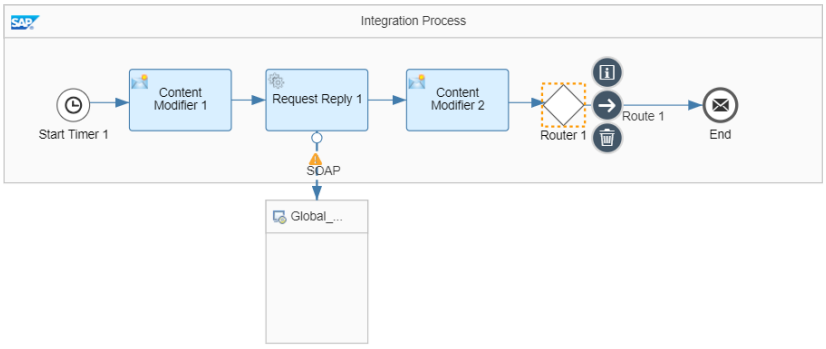
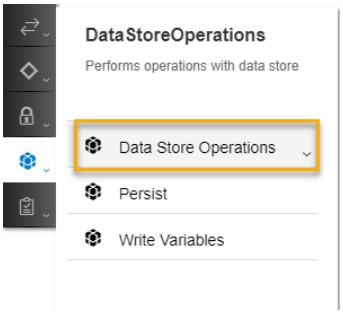
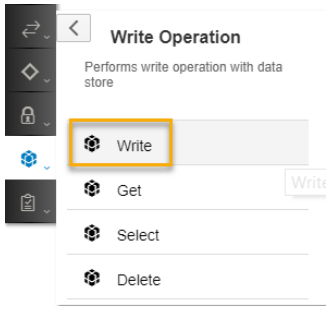
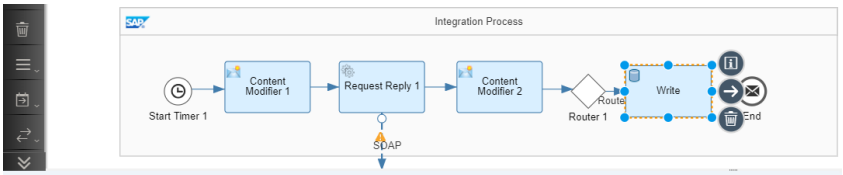
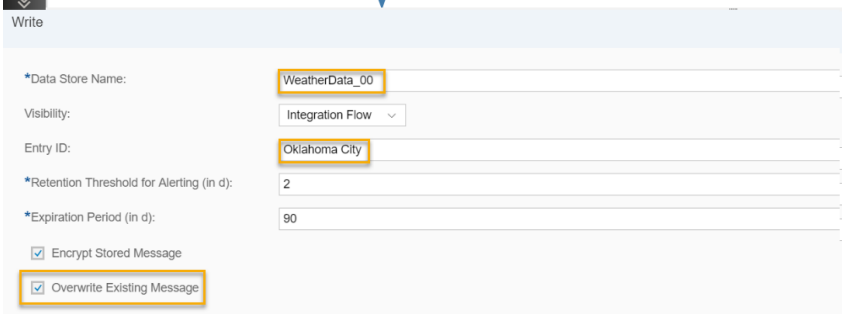
21) From the palette, select ***Message Transformers -> Content Modifier*** and drop it on the connection between ***Request Reply*** and ***End*** message in the integration flow.

Switch to ***Exchange Property*** tab, click on ***Add*** and add following properties:

1. Action: ***Create***
   Name: ***responseResult***
   Type: ***XPath***
   Data Type: ***java.lang.String***
   Value: ***//* [local-name()='dwmlByDayOut']/text()***





Save	
22) Select ***Router*** from the palette, drag and drop it on the connection between ***Content Modifier*** and ***End*** message	 
23) From the palette, select ***Persistence > Data Store Operations > Write*** and drop it on the connection between ***Router*** and ***End*** message in the integration flow. Provide the following properties: a. Data Store Name: ***WeatherData_XX*** b. Entry ID: ***Oklahoma City*** c. Overwrite Existing Message: ***Checked*** ***Note:*** Replace XX with your Group ID Save and ignore all the errors.	   

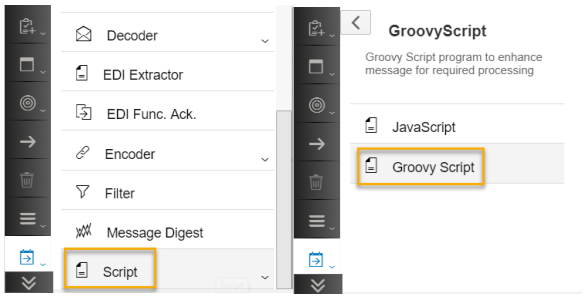
24) From the palette, select **Script > Groovy Script** and drop it on the connection between ***Write*** data store flow step and ***End*** message in the integration flow.

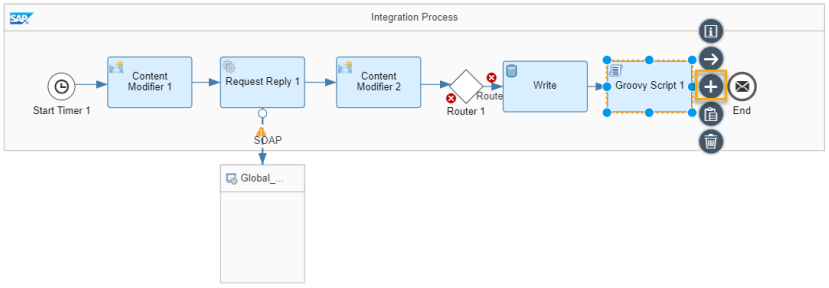
Click on ***Create*** icon ⊕

Replace the script default code with the following code:

```
import
com.sap.gateway.ip.core.customdev.util.Mes
sage;
import java.util.HashMap;

def Message processData(Message
message) {
    def body =
message.getBody(java.lang.String) as String;
    def messageLog =
messageLogFactory.getMessageLog(messa
ge);
    if(messageLog != null){

messageLog.addAttachmentAsString("Log
Current Payload:", body, "text/plain");
  }
    return message;
}
```

Click on **OK**





script1.groovy

```groovy
import com.sap.gateway.ip.core.customdev.util.Message;
import java.util.HashMap;

def Message processData(Message message) {
    def body = message.getBody(java.lang.String) as String;
    def messageLog = messageLogFactory.getMessageLog(message);
    if(messageLog != null){
        messageLog.addAttachmentAsString("Log current Payload:", body, "text/plain");
    }
    return message;
}
```

25) From the palette, add another ***Groovy Script***

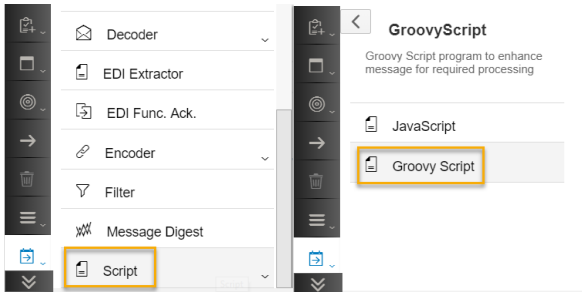
Place it as shown in the integration flow

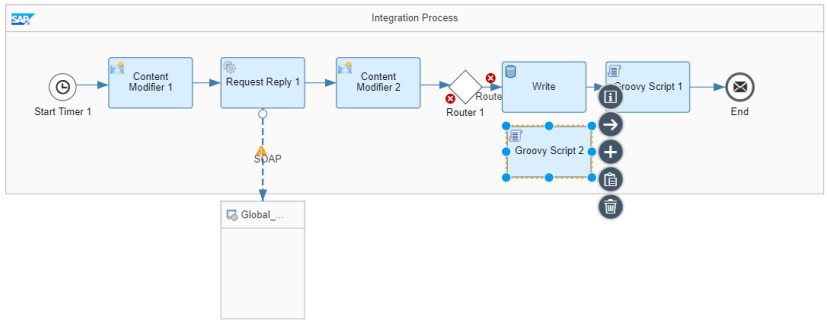
Click on ***Create*** icon ⊕

Replace the script default code with the following code:

```
import
com.sap.gateway.ip.core.customdev.util.Mes
sage;
import java.util.HashMap;

def Message processData(Message
message) {
    def messageLog =
messageLogFactory.getMessageLog(messa
ge);
    if(messageLog != null){

messageLog.addAttachmentAsString("Log
Current Payload:", "No valid response found
from the webservice", "text/plain");
  }
    return message;
}
```

Click on **OK**

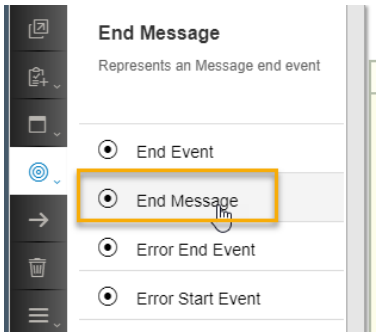


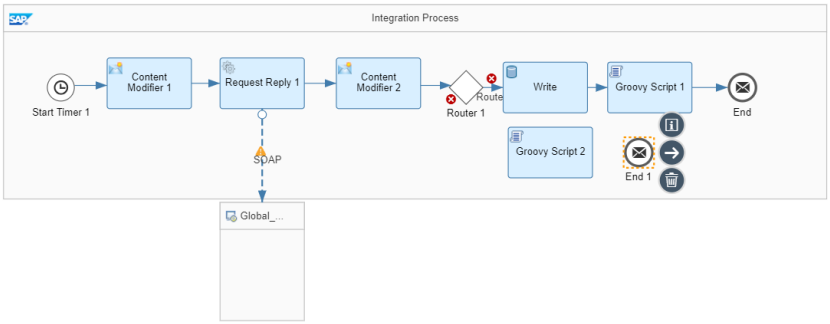


script2.groovy

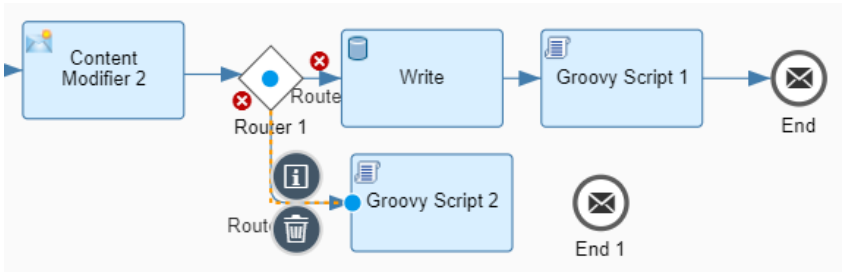
```groovy
import com.sap.gateway.ip.core.customdev.util.Message;
import java.util.HashMap;

def Message processData(Message message) {
    def messageLog = messageLogFactory.getMessageLog(message);
    if(messageLog != null){
        messageLog.addAttachmentAsString("Log current Payload:", "No valid response found from the webservice", "text/plain");
    }
    return message;
}
```

26) From the palette, Choose ***End Message*** and place it after the second Groovy Script.

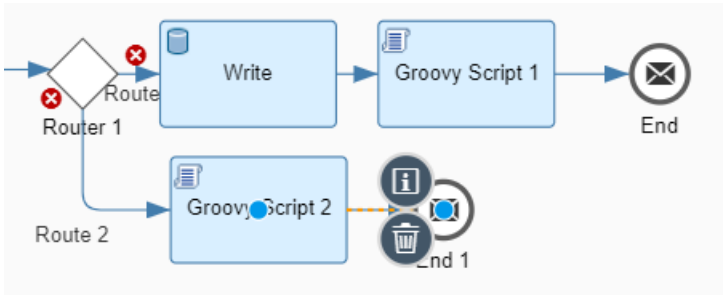




27) Connect ***Router*** to the ***Groovy Script***

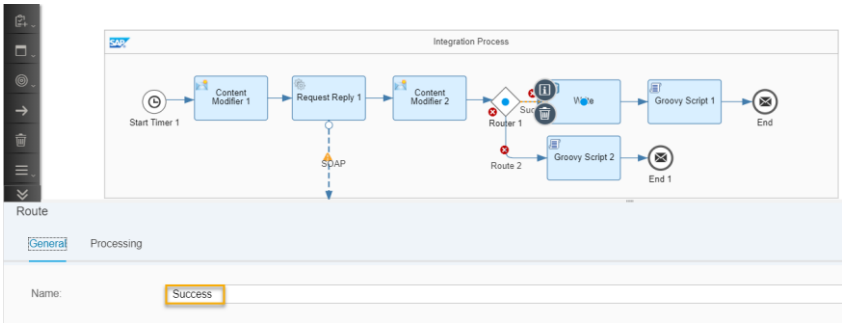


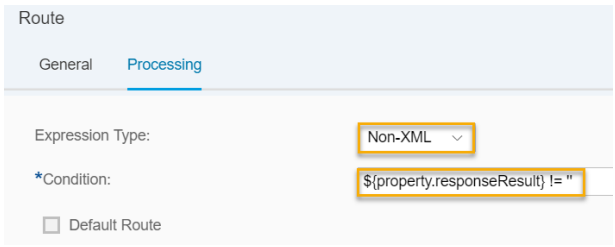
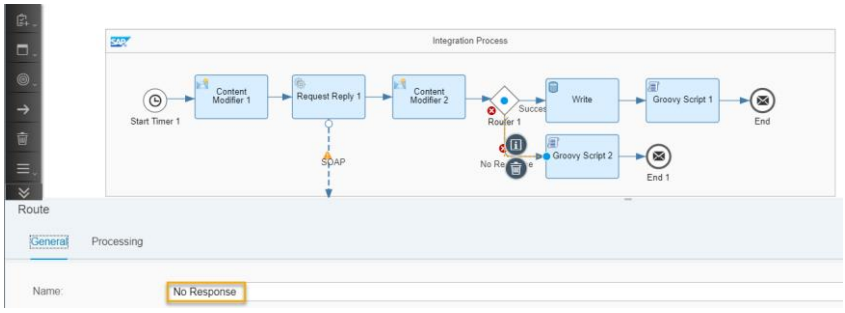
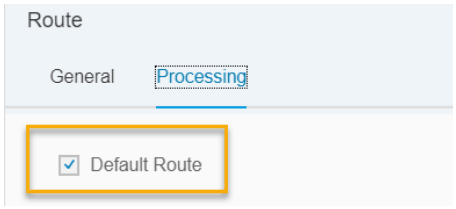
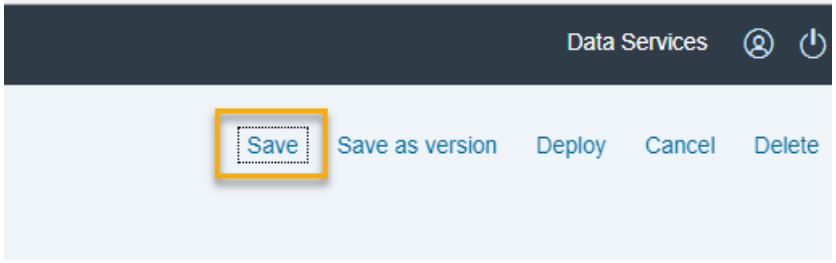
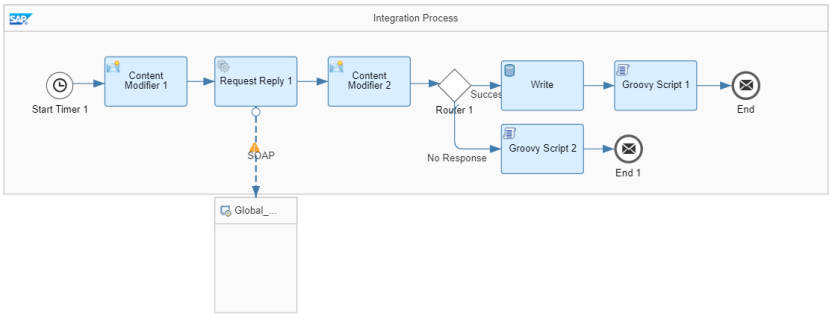
28) Connect ***Groovy Script*** to the ***End 1*** Message

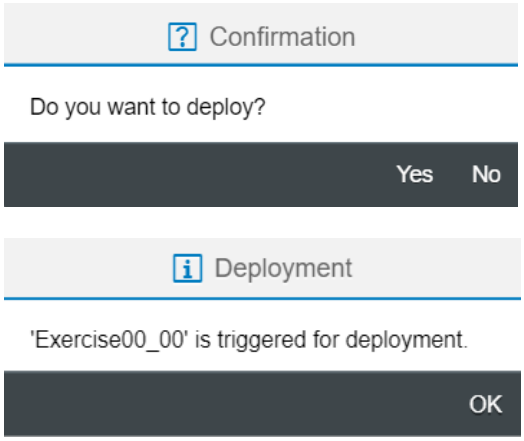
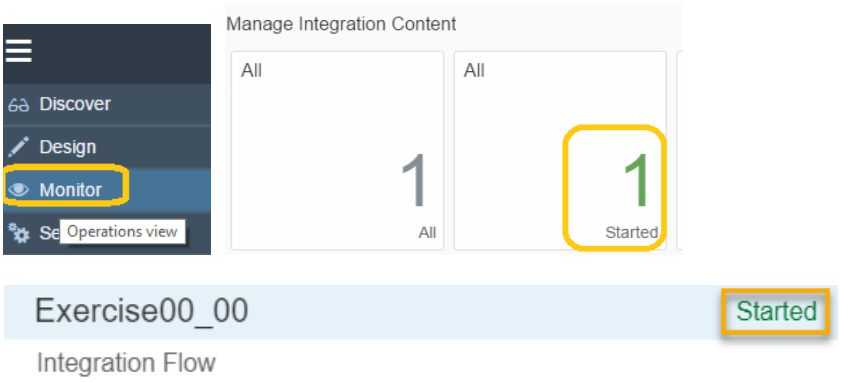
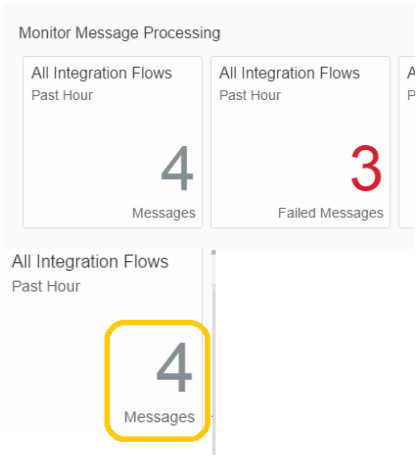


29) Select connection between ***Router*** and ***Write*** step and configure the route with following values:

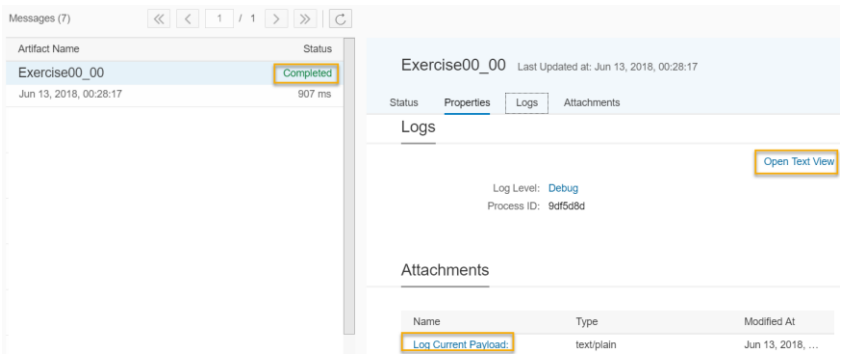
a. Name: ***Success***
b. Expression Type: ***Non-XML***
c. Condition: ***${property.responseResult} != ''***



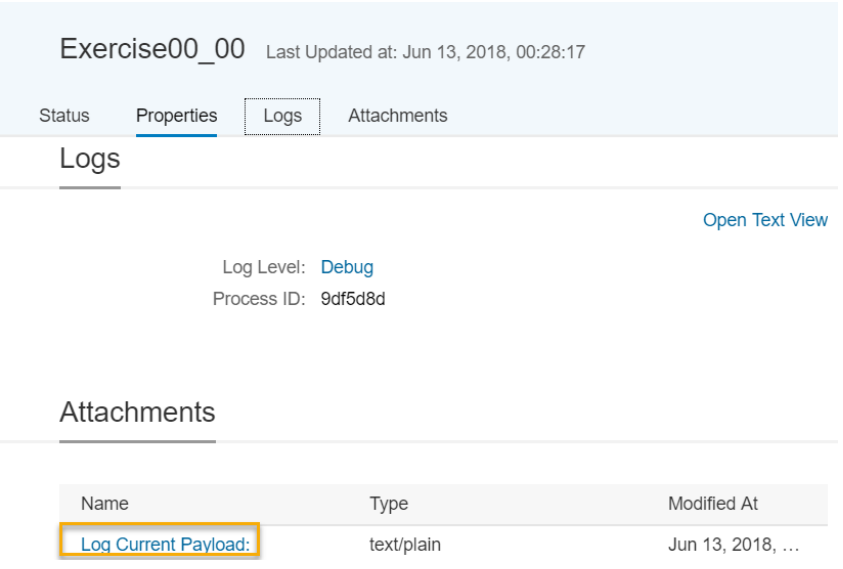
	
30) Select connection between ***Router*** and ***Groovy Script*** step and configure the route with following values: a. Name: ***No Response*** b. Default Route: ***Checked***	 
31) Save your changes	
32) Your integration flow should look like this.	
Deploy Integration Project on tenant:	**Follow this step to deploy Integration Project on tenant.**
33) Press **Deploy** in the Integration Flow Task Bar.	

34) You will receive a confirmation. Click on ***Yes***. Once the deployment is completed successfully you will receive a 2nd notification.	
35) Verify if the deployment is successful: In the first level Menu Bar switch to Section ***Monitor*** and then click on ***All*** in ***Manage Integration Content***. You should see an entry with your integration flow. Check the 'status'. It should be in status ***Started***.	
Execute end to end scenario:	**Follow steps to execute end to end scenario.**
36) As this is a Timer triggered iFlow and we have set schedule as Run Once, it would be executed immediately after the deployment.	
37) Switch to Section ***Monitor*** -> Select **Messages** On this dashboard you will find all the messages processed as per the status. If there is any error, you will find the processing/ error log in the Error Messages section on the dash board. Click on All messages	

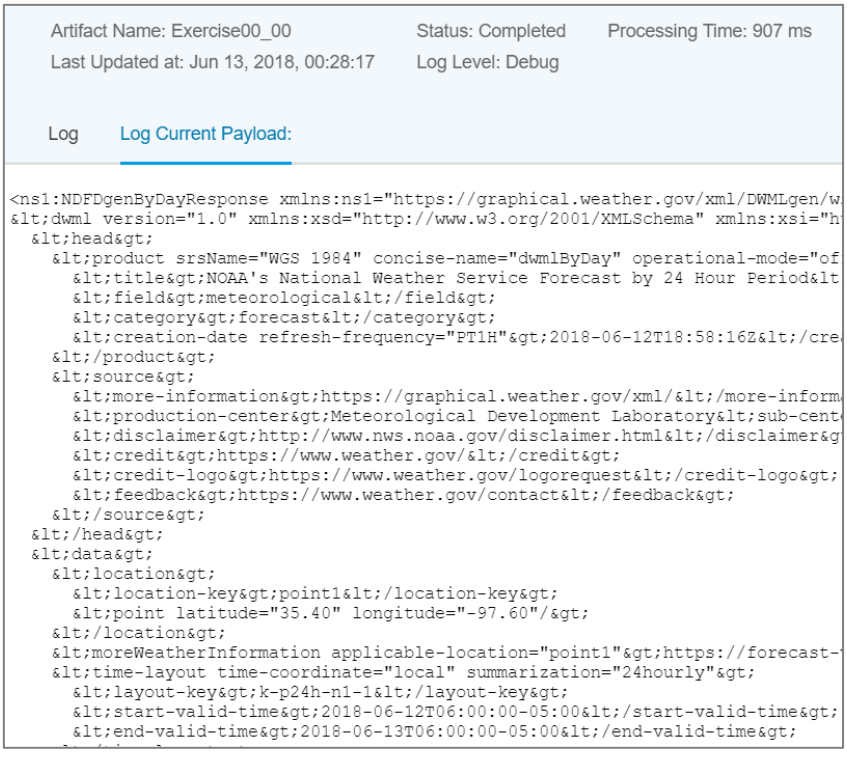
38) If message is successful, it can be seen with **Completed** status. Message Processing Log(MPL) and MPL attachments can be checked on clicking respective links on the right side.



39) Click on **Log Current Payload**

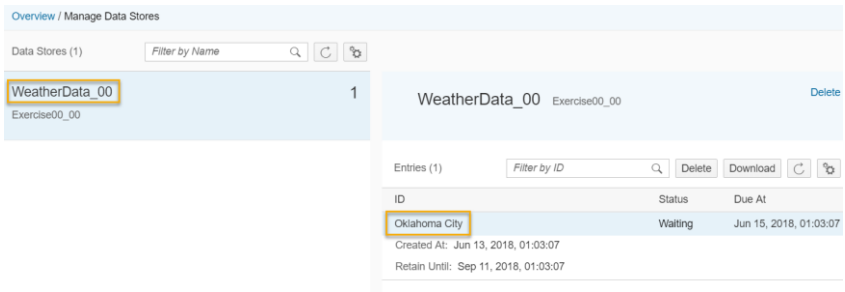


40) You should be able to see the response from the weather service.



41) Also go to the ***Operations View*** → ***Manage Stores*** → ***Data Stores***. You can see 1 new data stores is created and it has one entry with ID **Oklahoma City**.

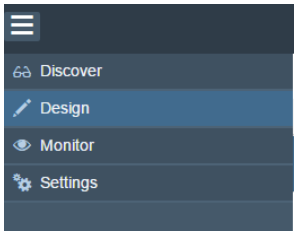
This concludes smoke test for the SAP Cloud Platform Integration Web UI and confirms that it is ready for productive usage

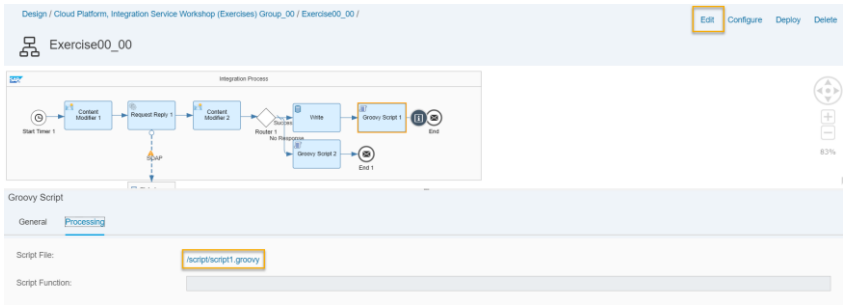


Checking Error details

42) Switch to ***Design*** section, locate your iFlow and select ***Groovy Script 1*** step

Click on **Edit** and then on **script1.groovy** to open the script.

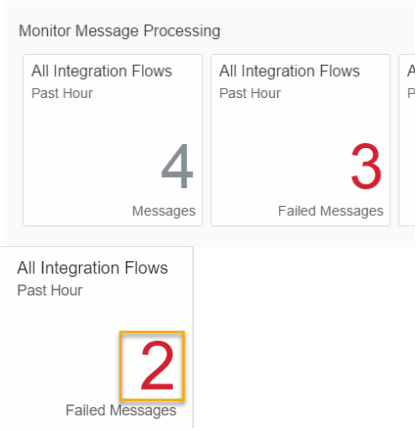
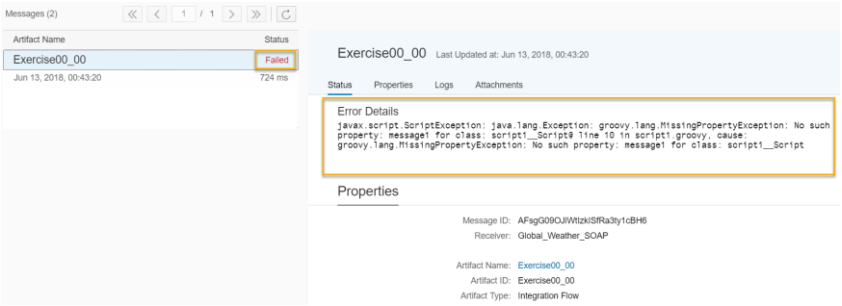
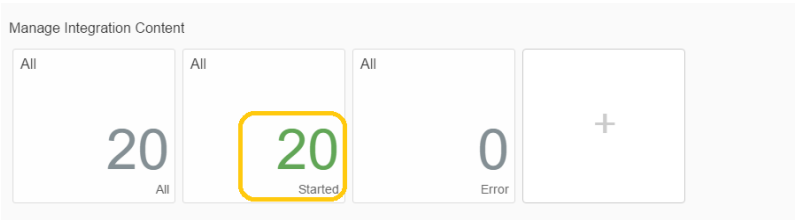
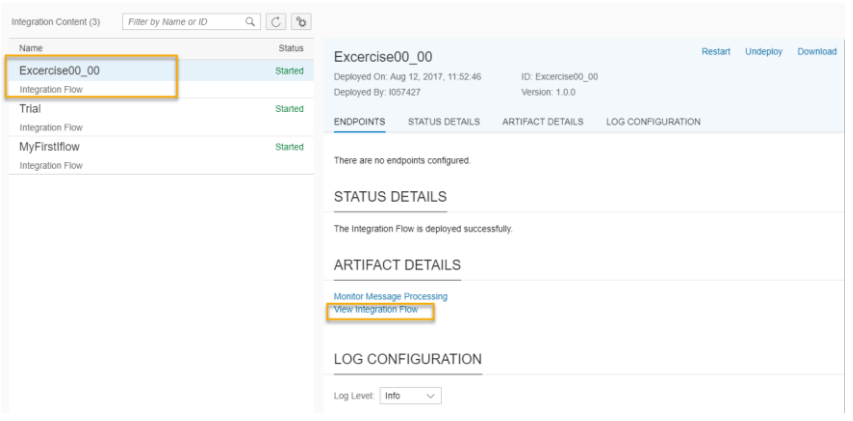




43) Change **return message** to **return message1**, click on **OK** button at top right corner.

This way we can introduce an error in the IFlow.

Save and Deploy

44) As this is a Timer triggered iFlow and we have set schedule as Run Once, it would be executed immediately after the deployment.	
45) Switch to Section ***Monitor*** -> Select **Messages** On this dashboard you will find all the messages processed as per the status. If there is any error, you will find the processing/ error log in the Error Messages section on the dash board. Click on **Failed** messages	
46) Error details are displayed on top of MPL (Message Processing Log)	
47) Go Back, under Manage Integration Content, click on number of **Started** Artifacts	
48) Select your iFlow and click on **View Integration Flow**	

49) This will open the iFlow you have deployed