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

Exercise06: SAP Cloud Platform Integration Components Used:

- AS2 Sender Adapter
- SFTP Receiver Adapter
- EDI Splitter
- Router
- EDI to XML Convertor
- XSLT Mapping
- Content Modifier
- Groovy Script
- Mail Receiver Adapter
- AS2 Receiver Adapter

Exercise06: Prerequisites

- Install and Configure SAP Cloud Connector

INTEGRATION SCENARIO

Retail customer has its own EDI Infrastructure (Mendelson Server) and would send purchase order to the partner (Sap Cloud Platform Integration) using ANSI X12 format and AS2 connectivity. Partner has SAP Cloud Platform Integration for EDI Infrastructure and AS2 connectivity. Partner will process EDI document, create a sales order in SAP ERP system using Standard IDOC ORDERS05 and simultaneously send 997 functional acknowledgements to the customer

Use Case: In this scenario, there are 2 Parties/Entities involved

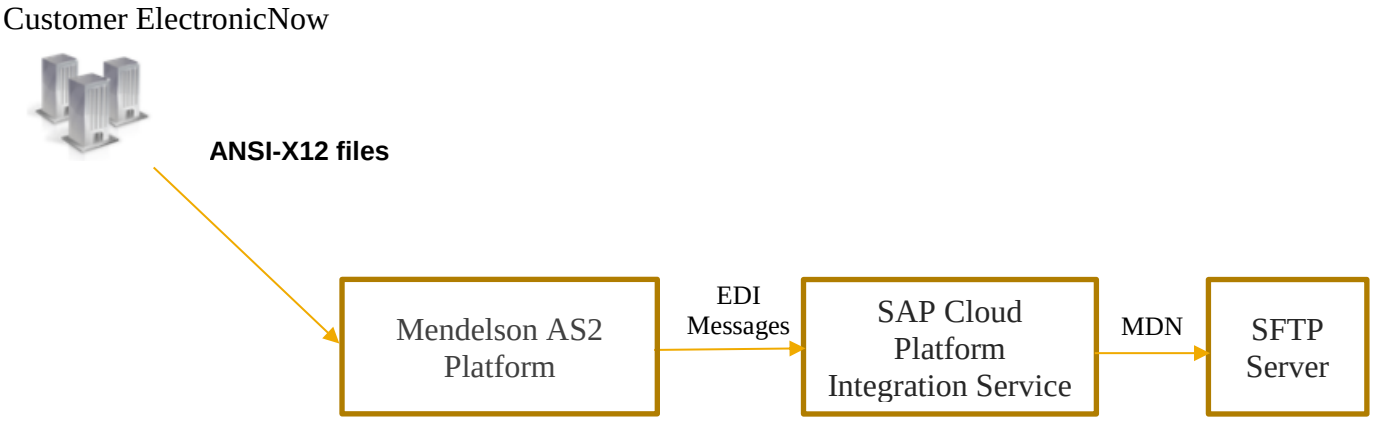
- One Retail customer - ElectronicNow(sender) which hosts its own EDI Mandelson platform, where he sends Purchase orders to its partner “SAP” in EDI-X12 format through AS2-Connection to partner’s Sap Cloud Platform Integration system and receives the generated functional acknowledgement 997 via AS2.
- Partner’s Sap Cloud Platform Integration system(receiver) receives the orders from Retail customer and processes these orders in the SAP CPI Backend and generate IDOC ORDERS05 sales orders.

Also, ElectronicNow Sender compresses, signs (Senders’ private key) and encrypts (Receivers’ public key) while sending the EDI message. At Receiver’s side, the receiver decrypts (Receivers’ private key), verifies the signature (Senders’ public key) and finally decompresses the EDI message.

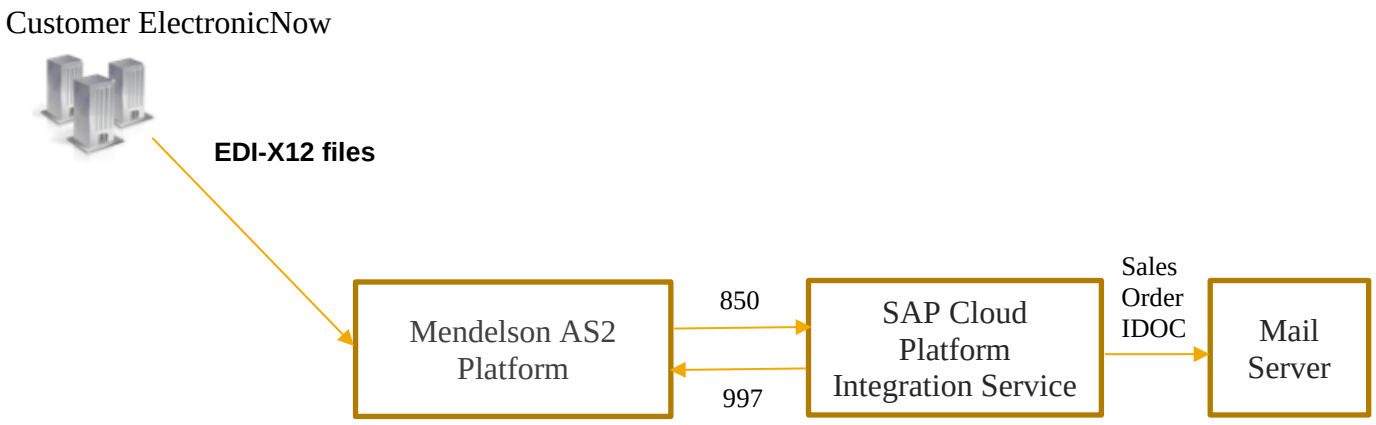
Here we are using private key **Key2** of AS2 Mendelson for signing and a public certificate from SAP Cloud Integration tenant for encryption purpose when we send EDI messages to partner

For the AS2 secure message exchange to happen between Mendelson and SAP Cloud Platform Integration tenant, we need to exchange the public keys between these two applications.

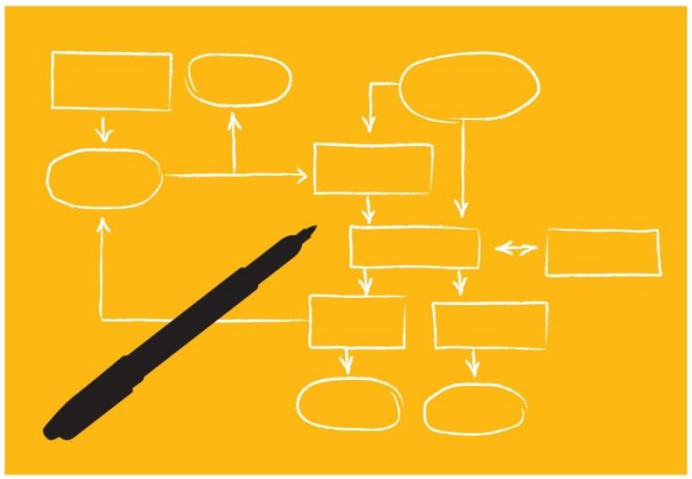
Exercise 06a ElectronicNow Customer will send ANSI X12 850 file from Mendelson AS2 Simulator to its partner SAP Cloud Platform Integration tenant as EDI messages using AS2 connection and SAP CPI system will process the EDI messages and generate MDN attachments



Exercise 06b ElectronicNow Customer will send ANSI X12 850 file to Mendelson AS2 platform, which will send it to SAP Backend CPI System as 850 EDI messages with Compression, Signing, and Encryption, SAP CPI service will process it and creates IDOC Sales orders and send the customer with functional acknowledgement of 997 message. Here we showcase both Synchronous MDN and Asynchronous MDN with Sign



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

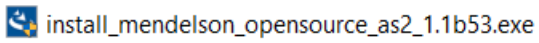
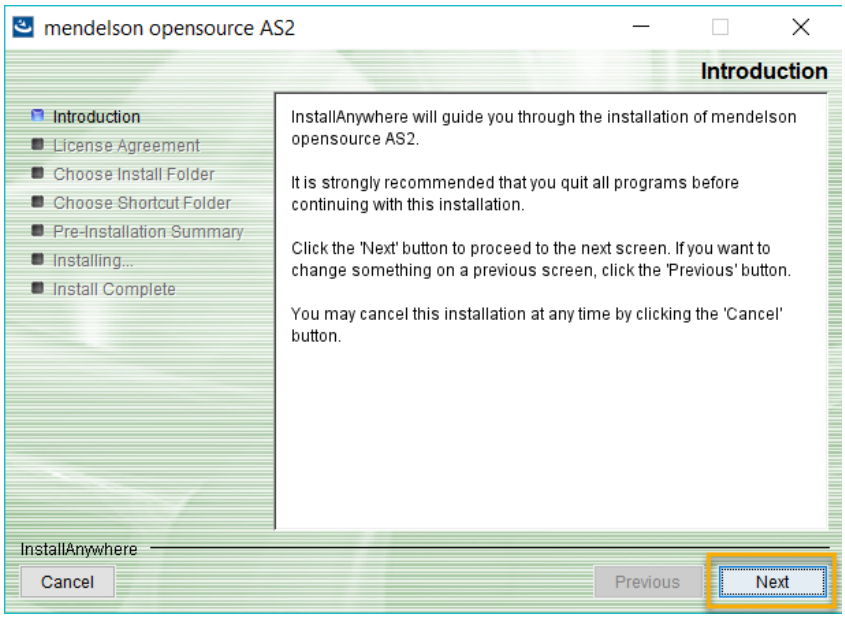
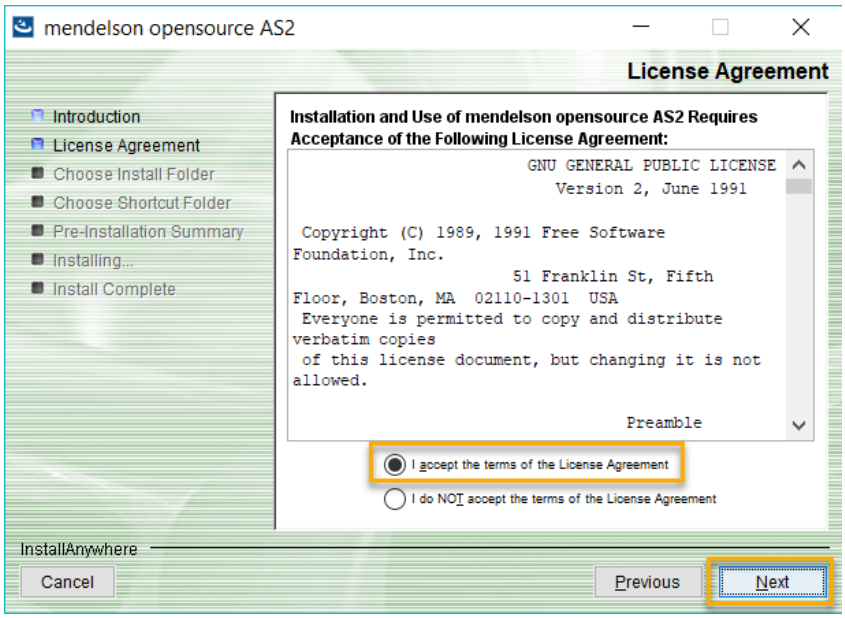
Exercise Files

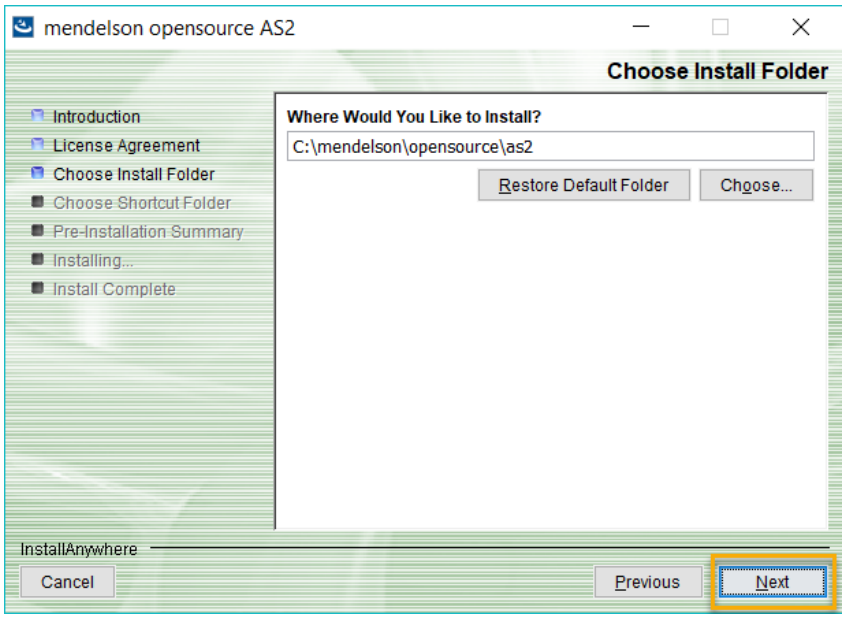
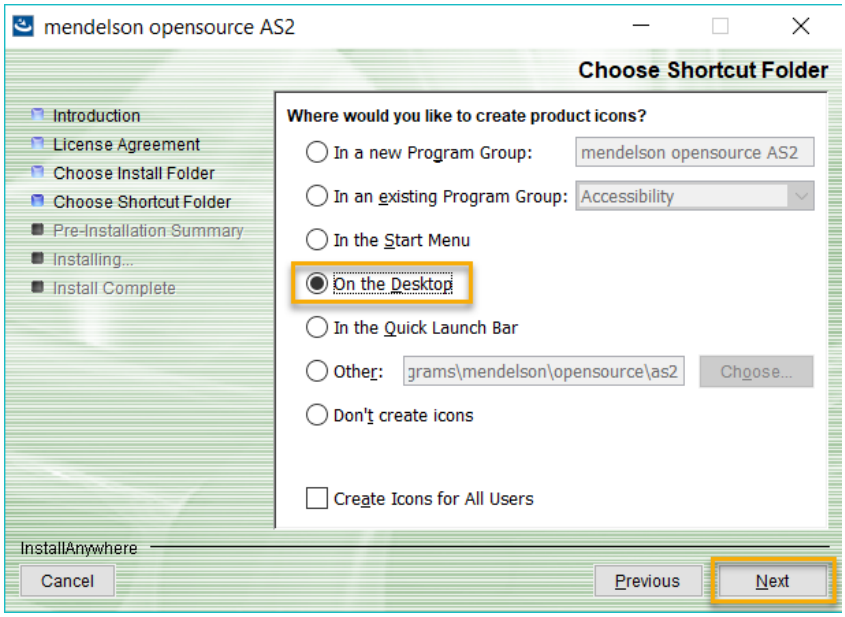
Exercise Files are provided by the session moderator. Download and Extract them into one of your local folders for use later in your exercises and setup:

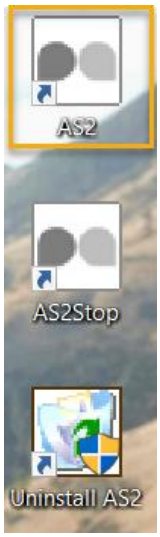
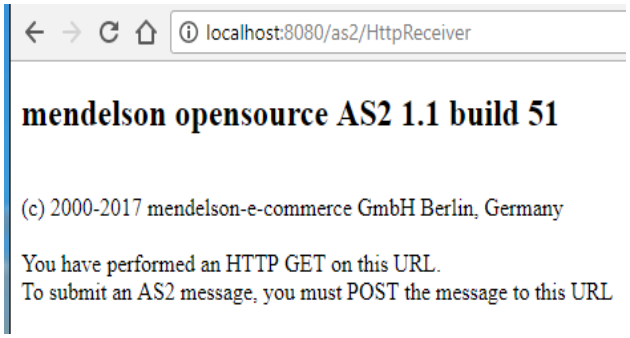
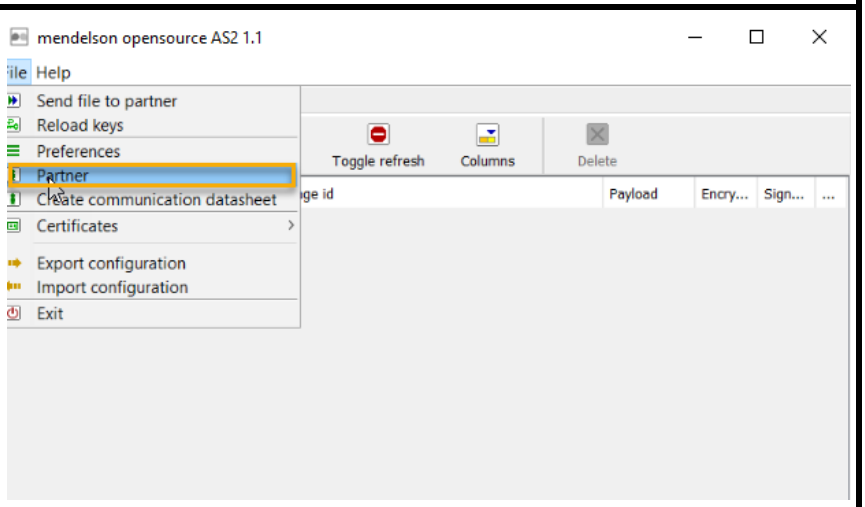
Note

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 ***rktw002/rktw003*** 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 06A PREREQUSITE (MENDELSON AS2 SETUP)

Explanation	Screenshot
Follow these steps to install Mendelson AS2 tool and configure customer and partners	
1) Run the **install_mendelson_opensource_as2_1.1b53.exe** from file system share or from http://mendelson-e-c.com/as2/	
2) Click on **Next**	
3) Accept the license agreement and click on **Next**	

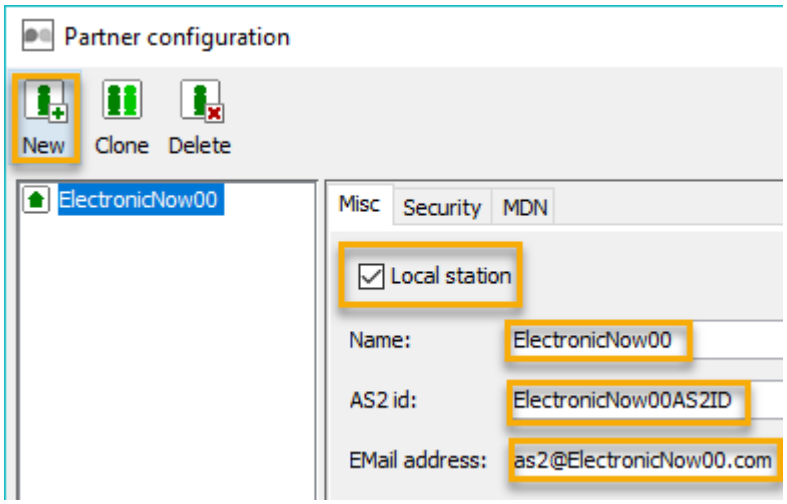
4) Select the install location and click **Next**	
5) Select **"on the Desktop"** option for create product icons and click on **Next** and complete the installation	

6) After Installation, you should see **AS2** and **AS2Stop** shortcuts created on the desktop. Launch **AS2** server by clicking on ***Run as administrator***	
7) Enter http://localhost:8080/as2/HttpReceiver in your browser and you should see the below message.	
8) Let's configure the AS2 partner i.e. ElectronicNow who act as initiator to post AS2 messages to its partner SAP. Select **File -> Partner**	

9) Click on **New** and select **Local Station** and configure the following values:

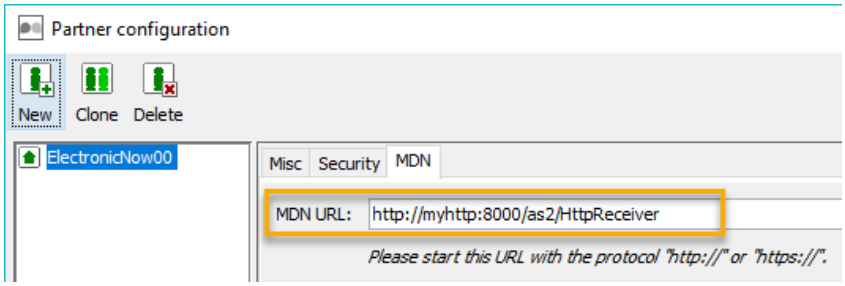
a. **Name:** ElectronicNowXX
b. **AS2 id:** ElectronicNowXXAS2ID
c. **Email address:** as2@ElectronicNowXX.com

Note: Replace **XX** with your participant id

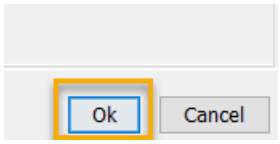


10) Choose the **MDN** tab and enter the MDN URL as http://<Virtual Host>:<Virtual Port>/as2/HttpReceiver

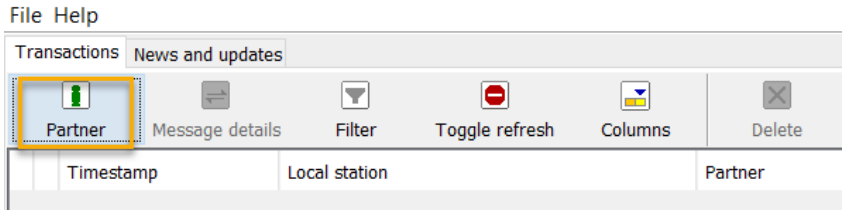
Note: Here **Virtual Host** and **Virtual Port** are the virtual host and port addresses which you have entered in Cloud Connector.



11) Click **Ok** to save the configuration



12) Now let's configure ElectronicNow's AS2 partner i.e. SAPCompany. Select **Partner** option again

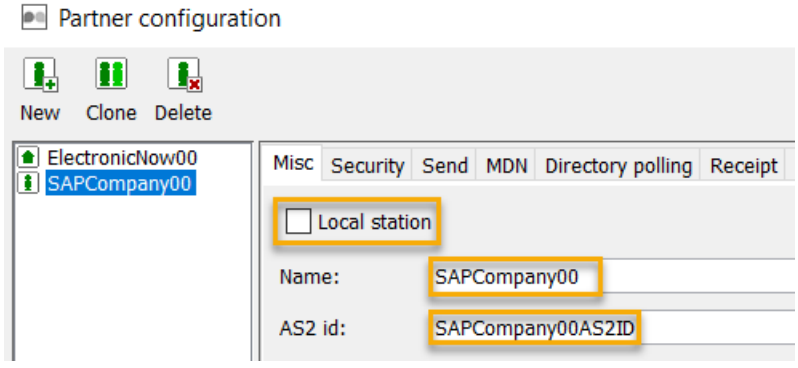


13) Click on **New** and configure the following values:

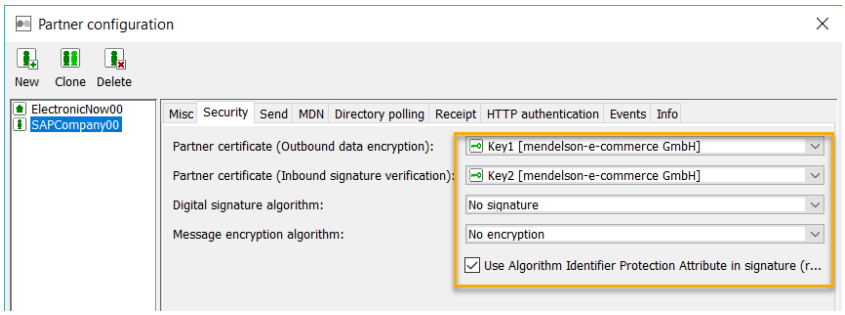
a. **Name:** SAPCompanyXX
b. **AS2 id:** SAPCompanyXXAS2ID

Make sure **Local Station** is unchecked

Note: Replace **XX** with your participant id



14) Choose the **Security** tab and configure security settings as shown in the screenshot.



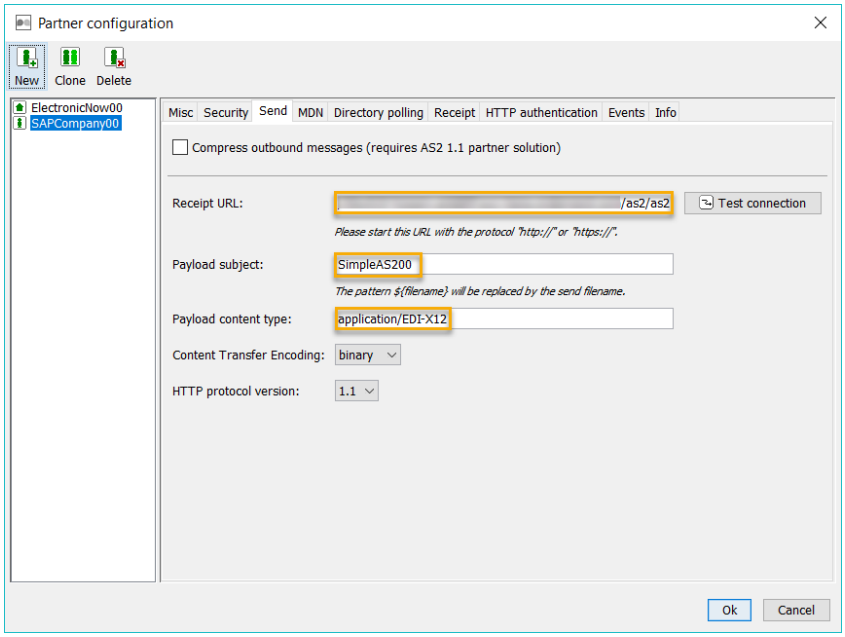
15) Choose the **Send** tab and configure settings as given below:

a. **Receipt URL:** <Runtime URL of the SAP Cloud Integration tenant>/as2/as2

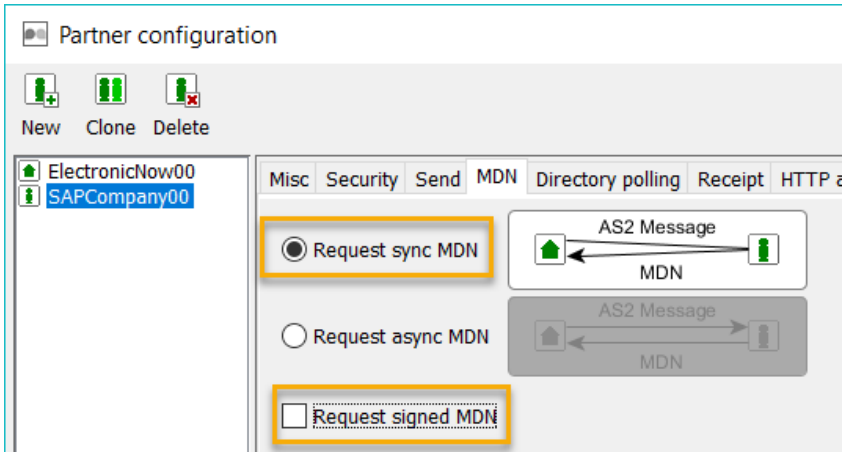
b. **Payload subject:** SimpleAS2XX

c. **Payload content type:** application/EDI-X12

Note: Replace **XX** with your participant id

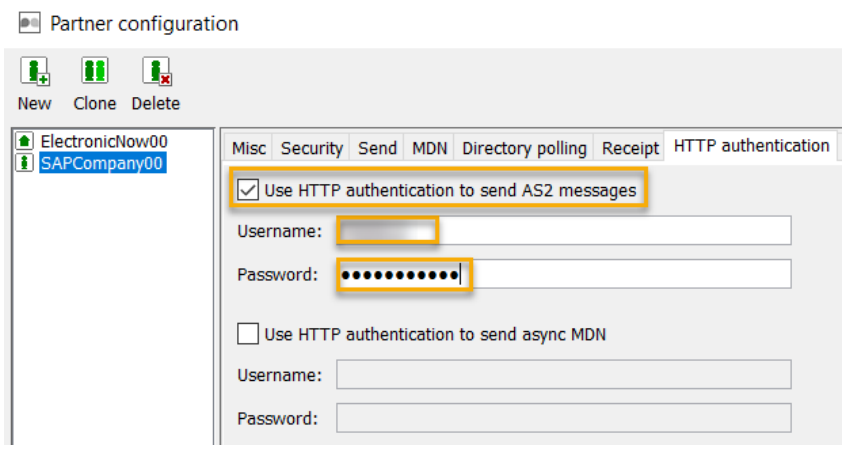


16) Choose the **MDN** tab and Configure MDN settings as shown in the screenshot.
Uncheck **"Request signed MDN"**

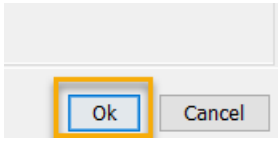


17) Choose the **HTTP authentication** tab and configure HTTP authentication to send AS2 messages to Cloud Platform Integration tenant as shown in the screenshot.

Where Username and Password is your Cloud Platform Integration tenant login user credentials.



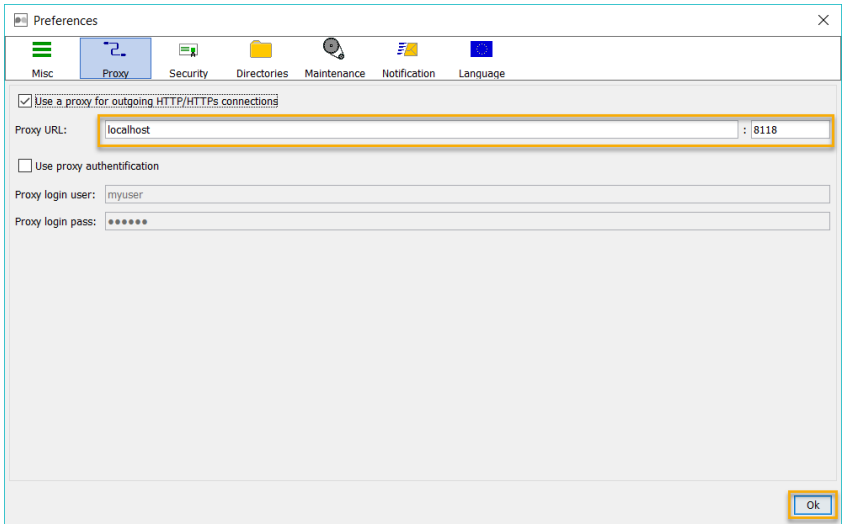
18) Click **Ok** to save the configuration



19) Configure proxy settings (Optional)

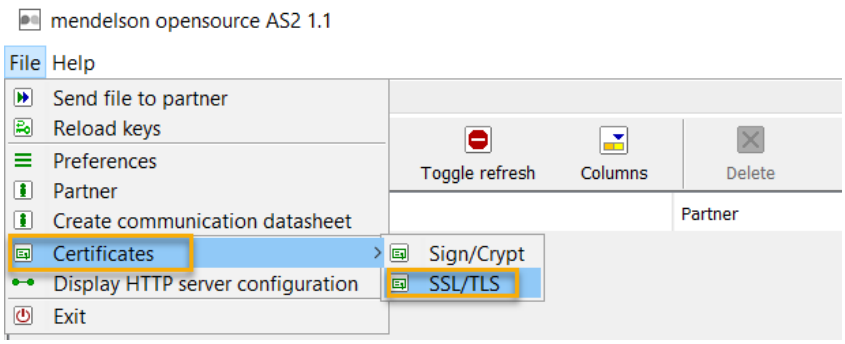
Navigate to **File -> Preferences -> Proxy** and set the proxy value.

If you are in internet, you may leave these settings blank. If your organizations require it, then you can mention proxy settings as per your organization.

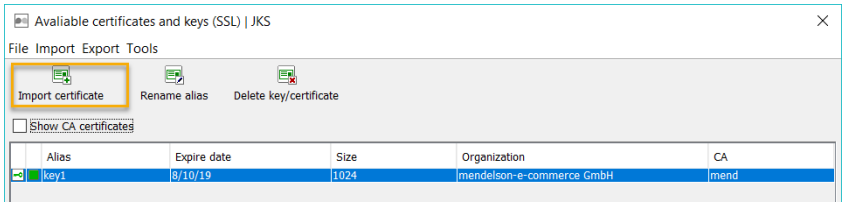


20) Import Cloud Platform Integration root certificate on to the Mandelson tool

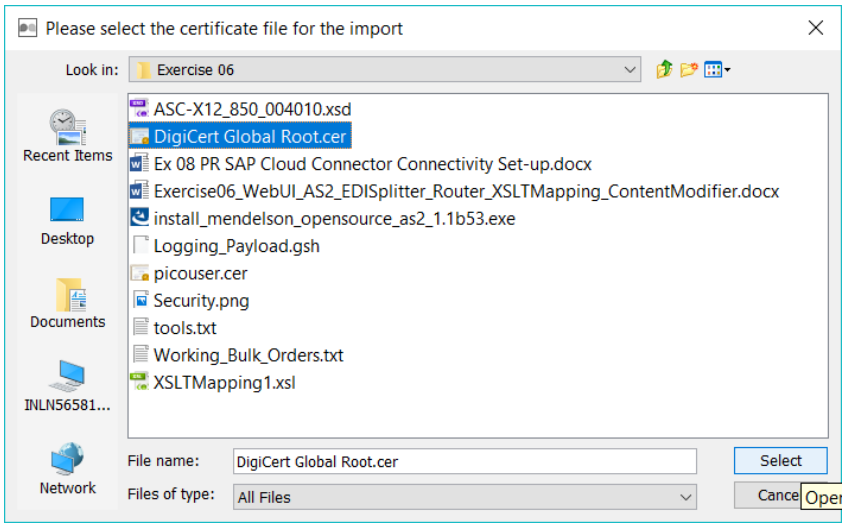
a. In the Mandelson tool navigate to **File -> Certificates -> SSL/TLS**



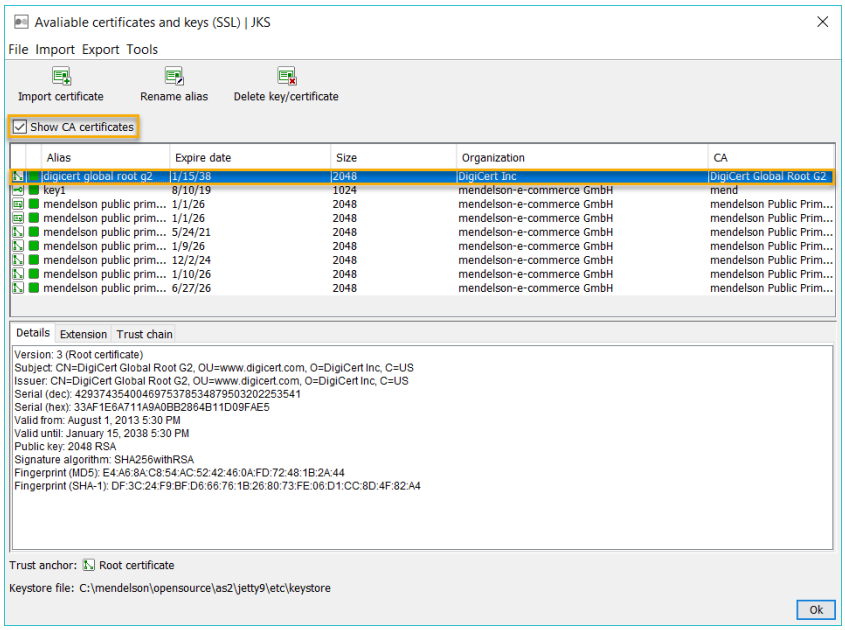
b. Click on import certificate



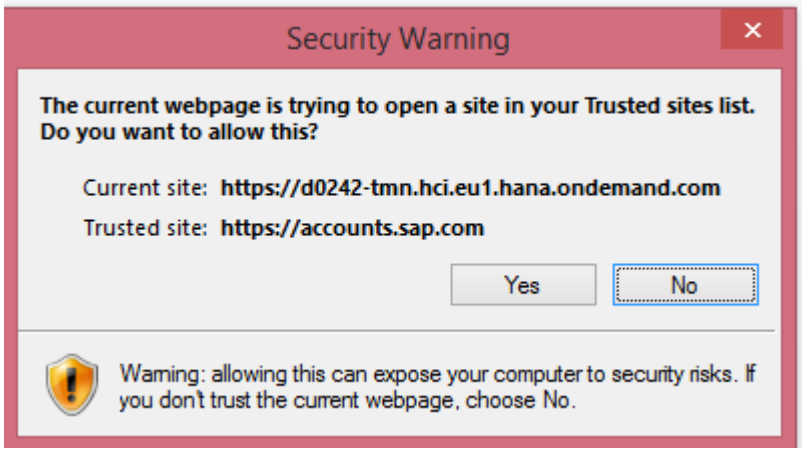
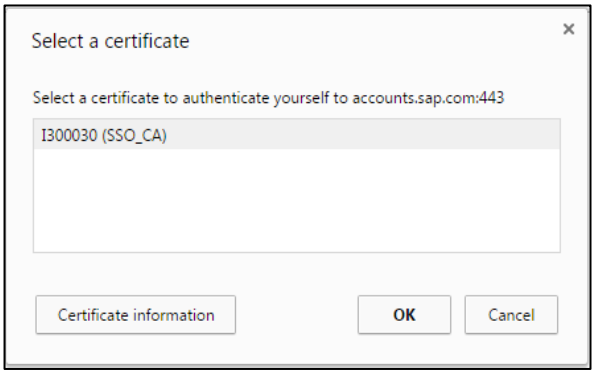
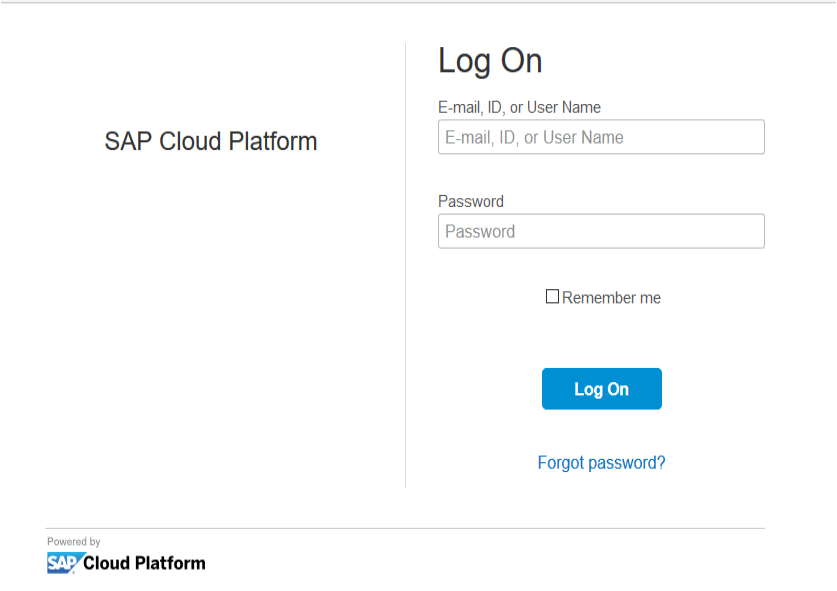
c. Click on Browse and from the Exercise folder, import the certificate **DigiCert Global Root.cer** file.



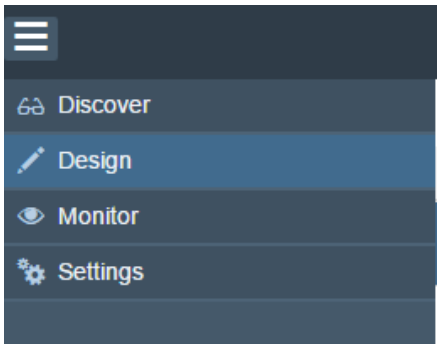
21) Check "**Show CA certificates**" to view Root certificate. SAP Cloud Platform Integration Root certificate should be listed.



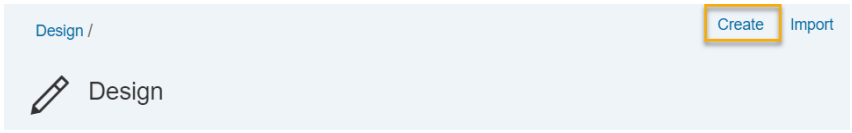
EXERCISE 06A

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 Internet Explorer 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



Click on **Create**



Enter Details

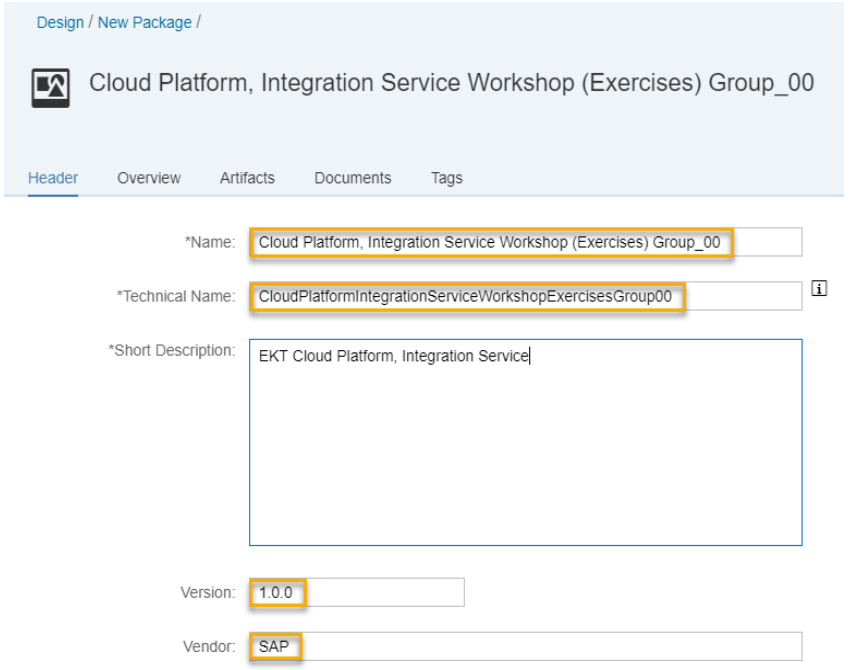
a) **Name**: Cloud Platform, Integration Service Workshop (Exercises) Group_XX

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

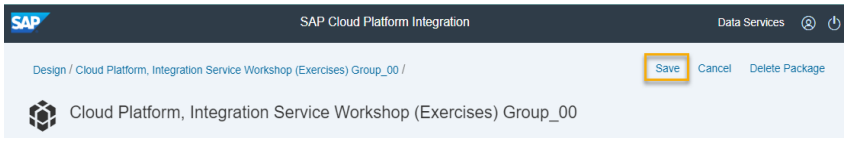
b) **Short Description**: EKT 2018 Cloud Platform, Integration Service

c) **Version:** 1.0.0

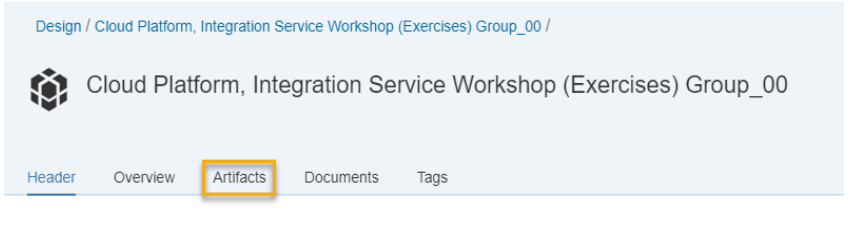
d) **Vendor:** SAP

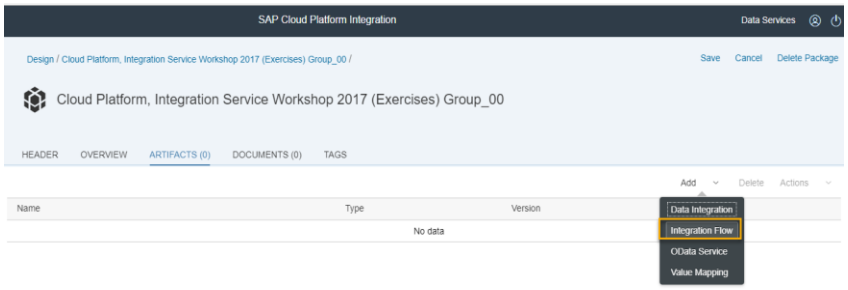
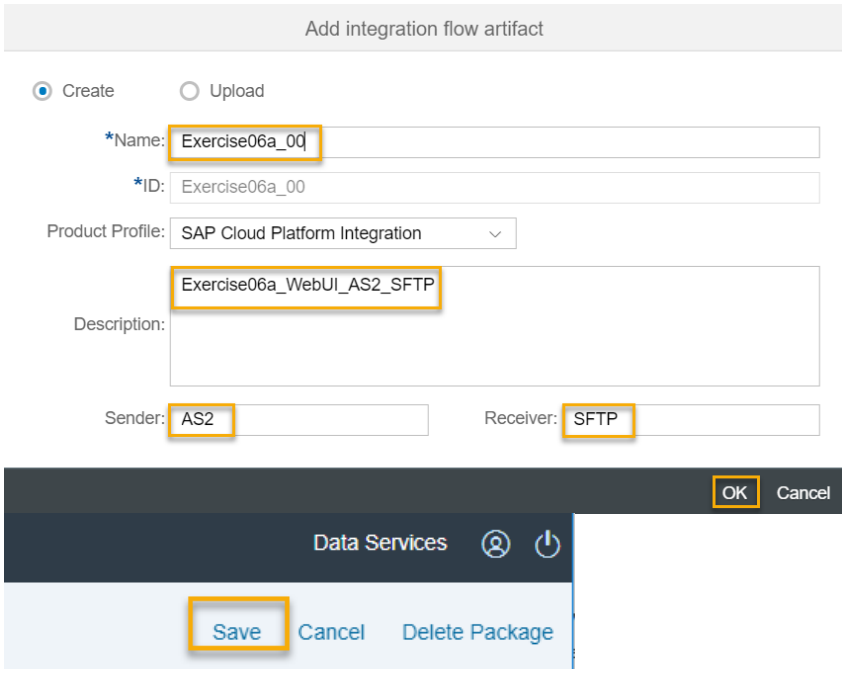
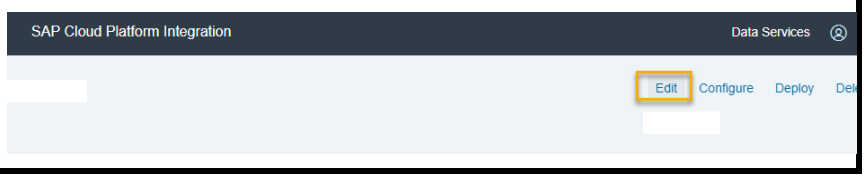


5) Click on Save

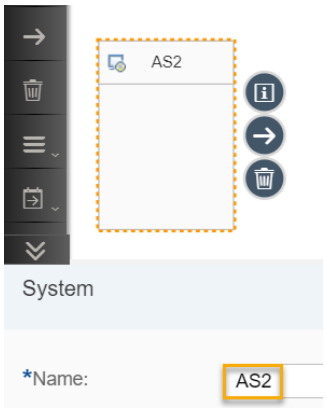


6) Navigate to **Artifacts** Tab

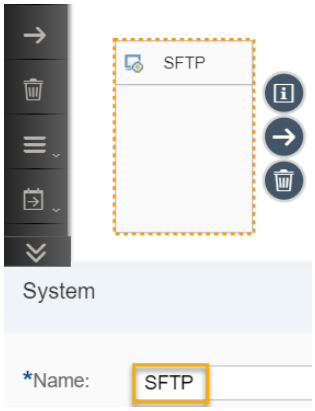


7) Click on ***Add-> Integration Flow***	
8) Select ***Create*** and enter following details: a. Name: ***Exercise06a_XX*** b. Description: ***Exercise06a_WebUI_AS2_SFTP*** c. Sender: ***AS2*** d. Receiver: ***SFTP*** e. Click on ***OK*** **Note:** Please remember to save after every action	
9) Click on **Excercise06a_XX**	
Define and edit integration flow	**Follow steps to edit the integration flow**
10) Click on ***Edit*** button on the upper corner on right hand side.	

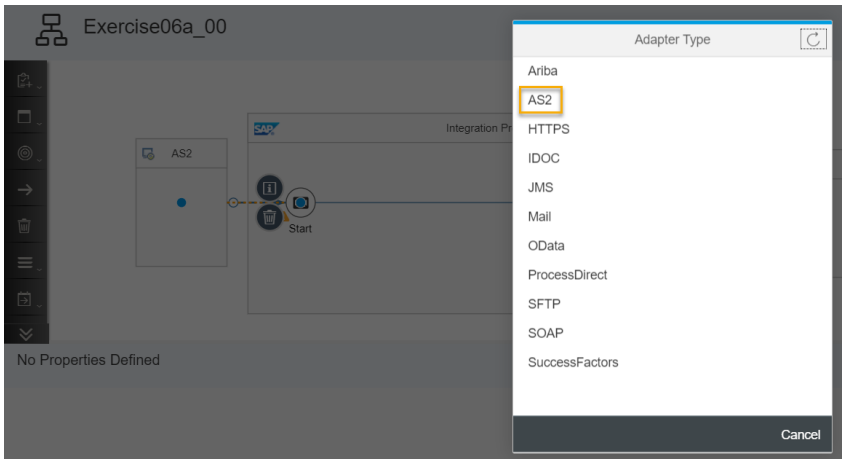
11) Click on Sender system and set Name as ***AS2***



12) Click on Receiver System and set Name as ***SFTP***

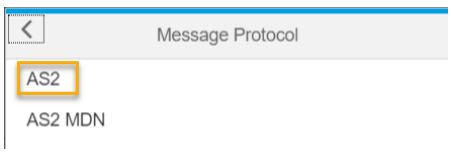


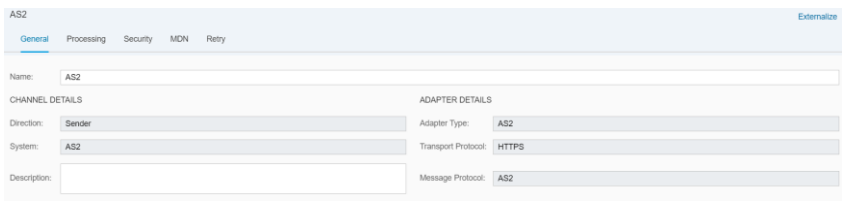
13) Click on ***AS2*** sender system, select ***Connector*** and drag it to ***Start*** message step.
This would open the sender adapter list.



Select ***AS2*** adapter

Select ***AS2*** as Message Protocol





14) Switch to ***Processing*** tab, enter the following details:

a. Message ID Left Part: **.***

b. Message ID Right Part: **.***

c. Partner AS2 ID: **ElectronicNowXXAS2ID**

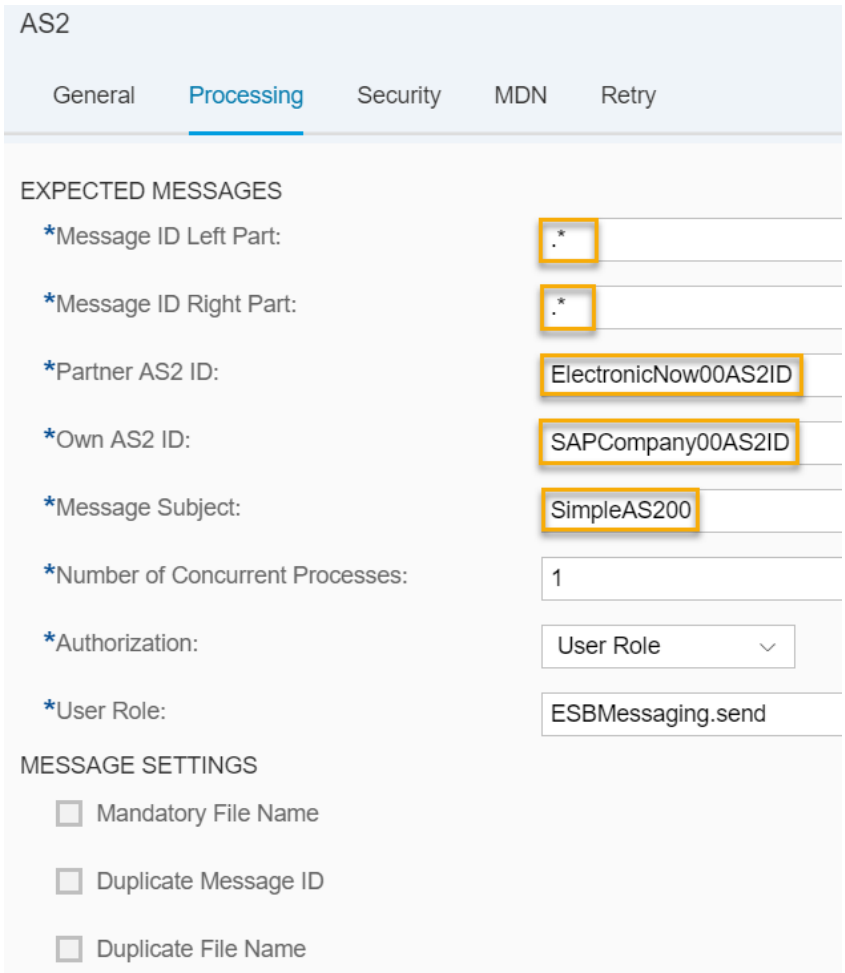
d. Own AS2 ID: **SAPCompanyXXAS2ID**

e. Message Subject: **SimpleAS2XX**

Keep other properties related to Security, MDN and Retry to the default values

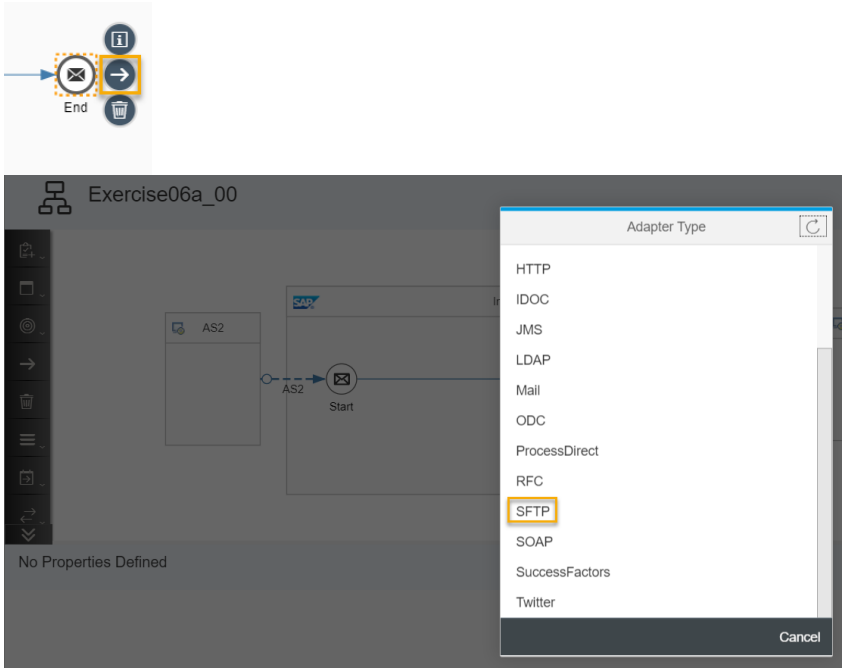
Note: Replace **XX** with your participant id

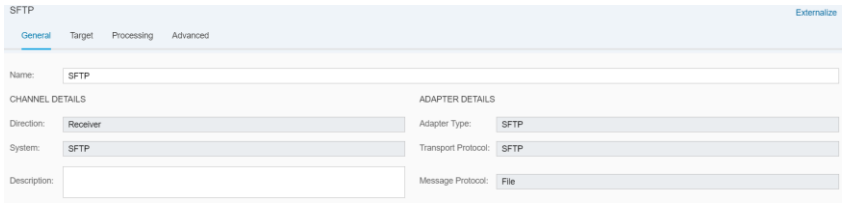
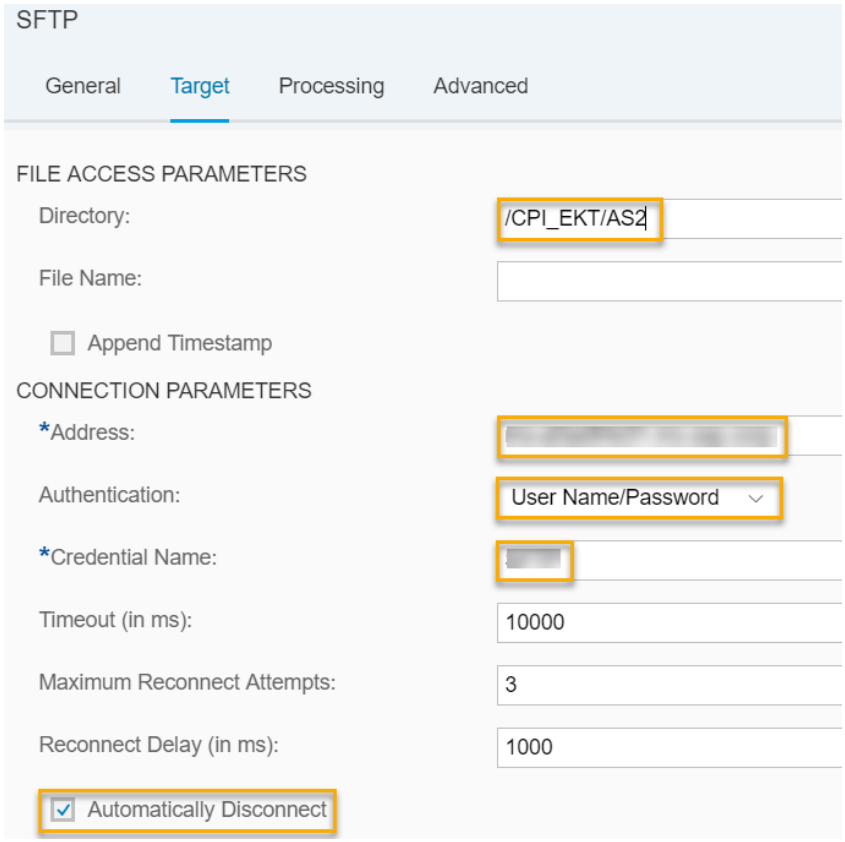
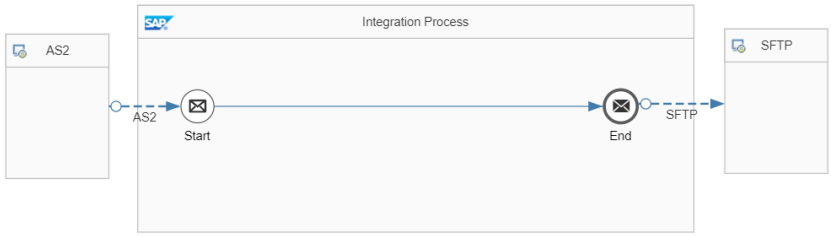
Save



15) Click on ***End*** message step, select ***Connector*** and drag it to ***SFTP*** receiver system.
This would open the receiver adapter list.

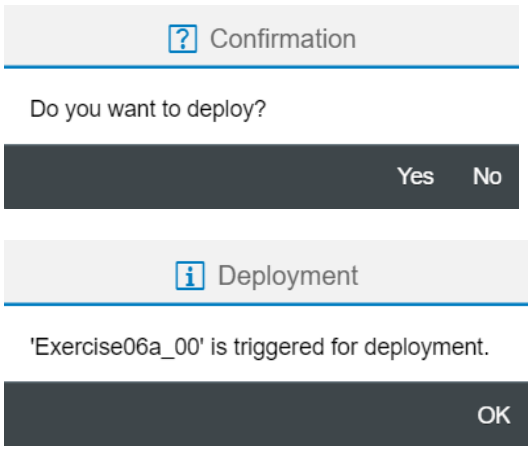
Select **SFTP** adapter



	
16) Switch to ***Target*** tab, enter the following details: a. ***Directory:*** /CPI_EKT/AS2 b. ***Address:*** <provided by the Instructor> c. ***Authentication:*** User Name/Password d. ***Credential Name:*** Name of credential object which holds SFTP user and password, deployed on the tenant. e. ***Automatically Disconnect:*** checked Save	
17) Your integration flow should look like this.	
Deploy Integration Project on tenant:	**Follow this step to deploy Integration Project on tenant.**
18) Press **Deploy** in the Integration Flow Task Bar.	

19) You will receive a confirmation. Click on ***Yes***.

Once the deployment is completed successfully you will receive a 2nd notification.

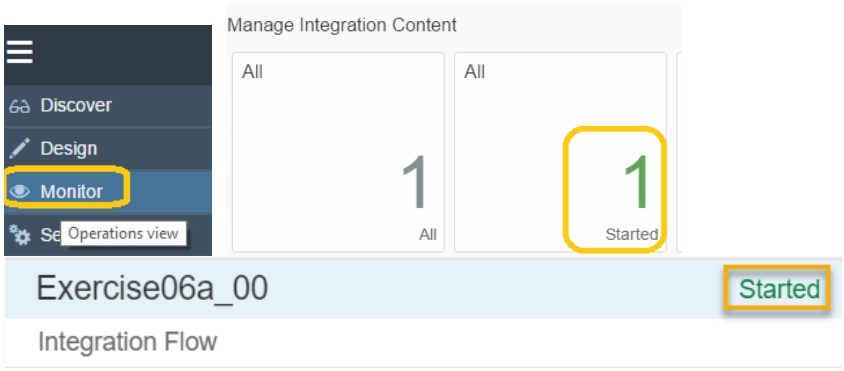


20) Verify if the deployment is successful:

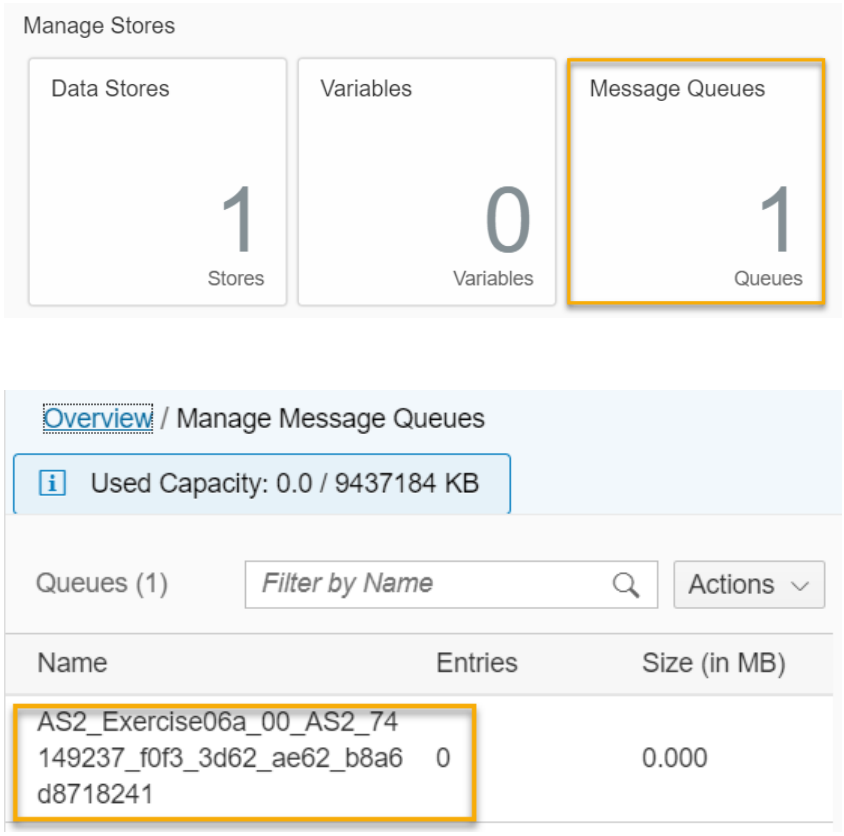
In the first level Menu Bar switch to Section ***Monitor*** and then click on ***All*** in ***Manage Integration Content***.

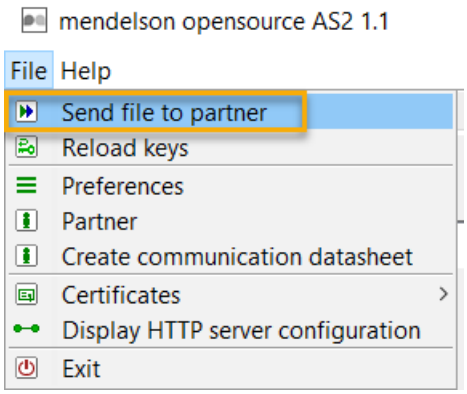
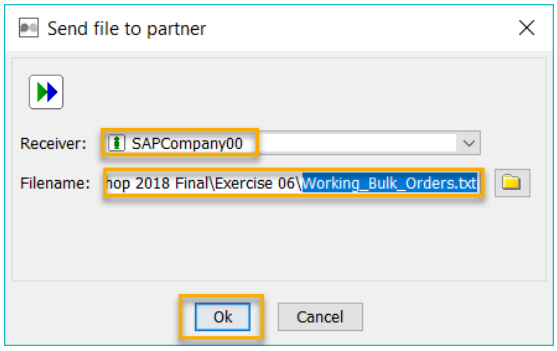
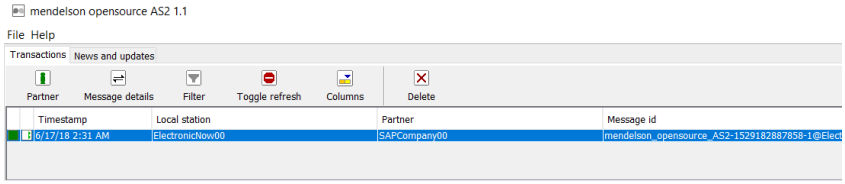
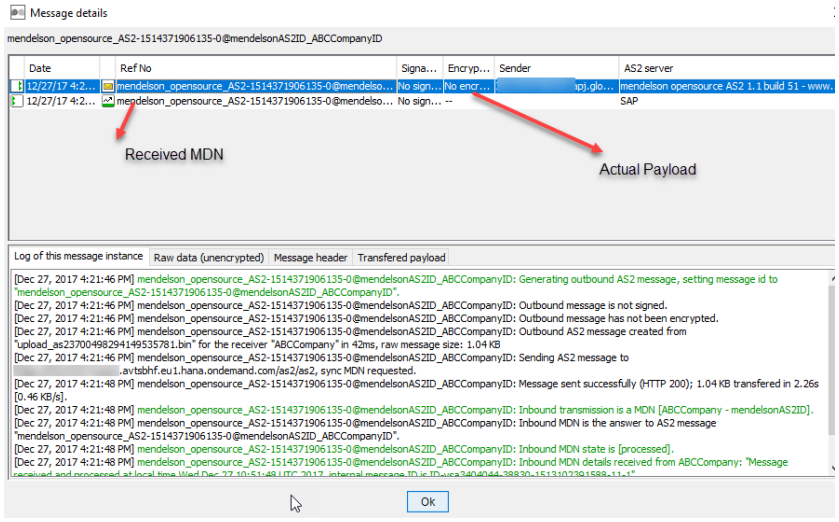
You should see an entry with your integration flow.

Check the 'status'. It should be in status ***Started***.

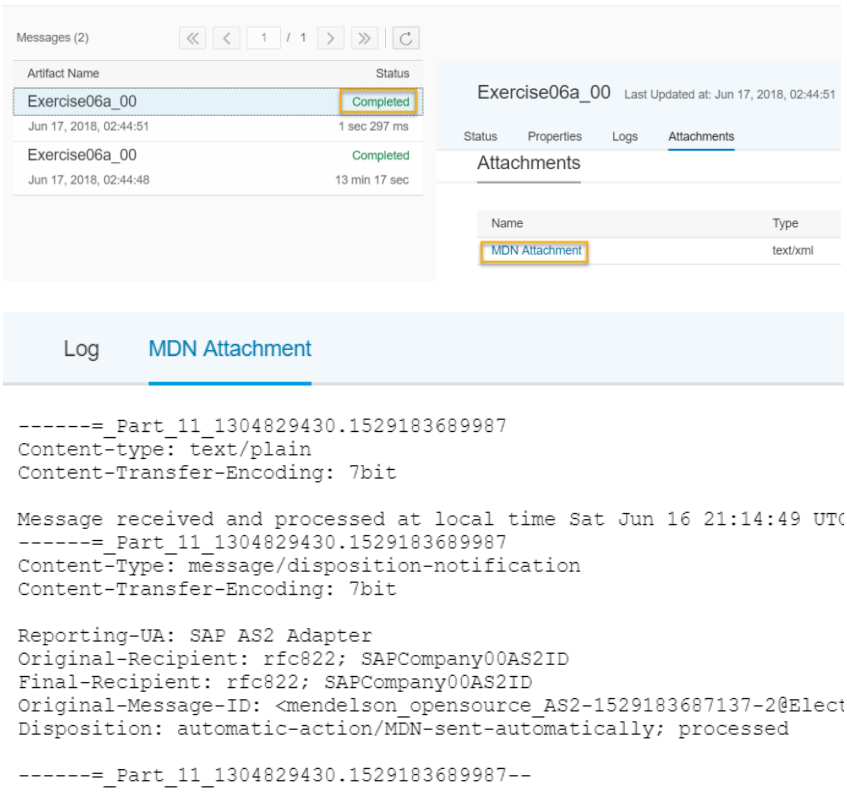


21) A Message queue named after the integration flow is created under **Monitoring -> Manage Stores -> Message Queues**



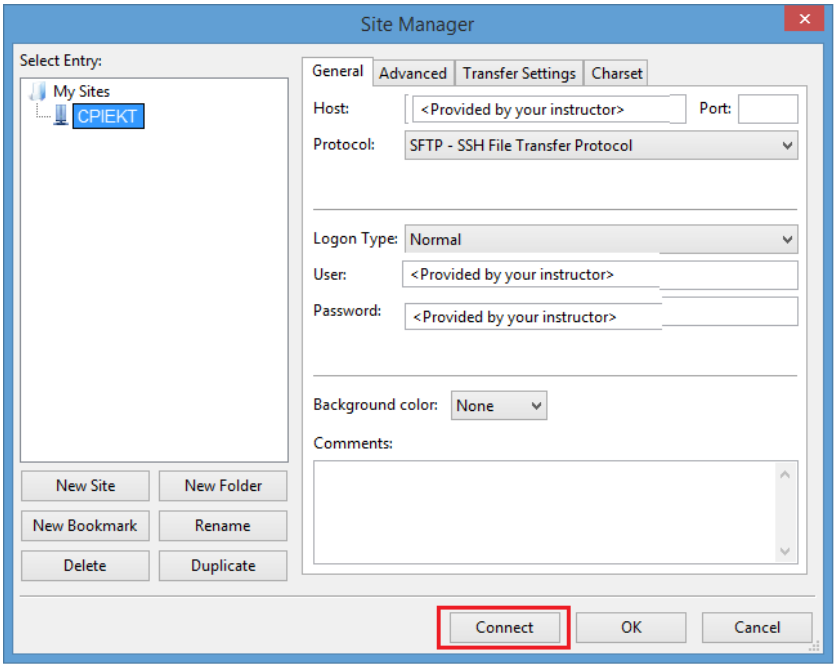
Execute end to end scenario:	Follow steps to execute end to end scenario.
22) Let's test a simple AS2 message post from Mendelson to SAP Cloud Platform Integration Tenant In the Mendelson tool, Choose **File -> Send file to partner**	
23) Choose **SAPCompanyXX** from the Receiver dropdown and select **Working_Bulk_Orders.txt** file and click Ok	
24) You should see a successful message in the Mandelson tool as shown below	
25) Double click on the message to see the message that was sent and the MDN received back	

26) On the CPI tenant, Navigate to **Monitor Message processing** to see a Completed message with an MPL attachments (**MDN Attachment**)

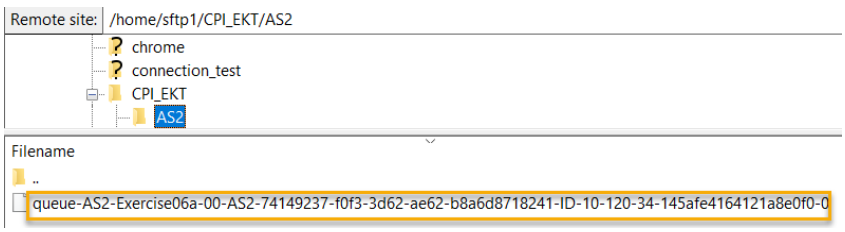


Check generated file with FileZilla	**Follow these steps to check the file in the SFTP folders**

27) Open FileZilla and connect to the **SFTP Server** using the credentials provided by your instructor



28) Check the generated file in the AS2 folder



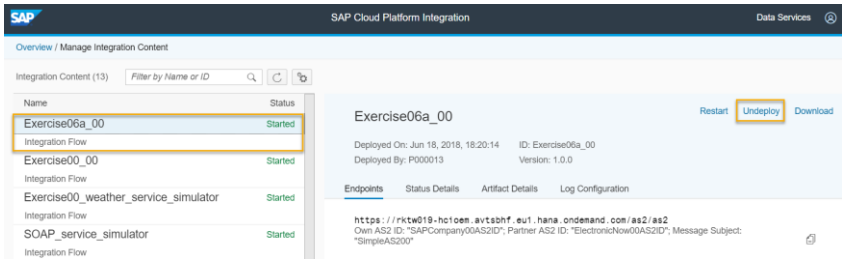
29) Finally, **Undeploy** the integration flow.

a. Navigate to **Monitoring -> Manage Integration Content**

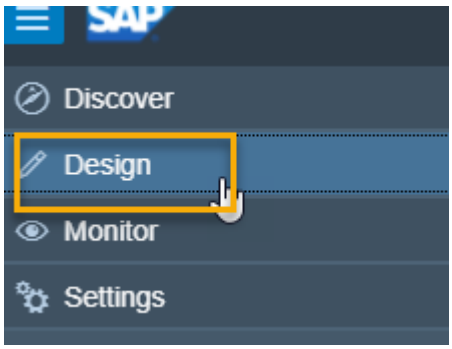
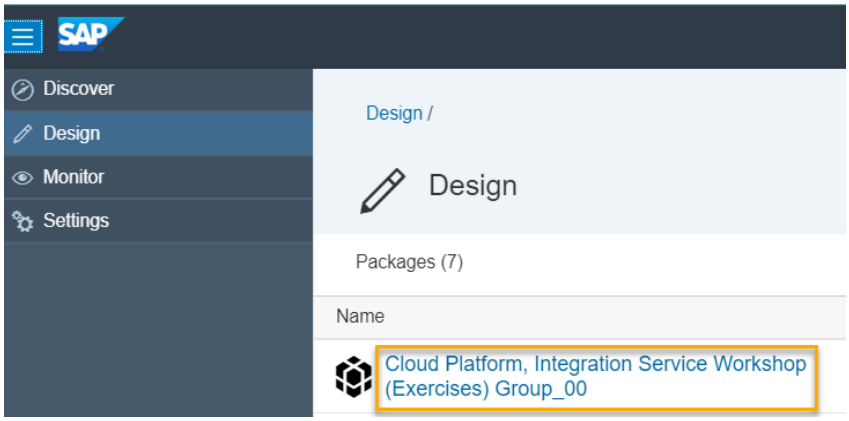
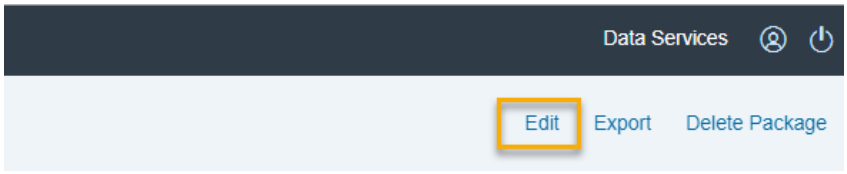
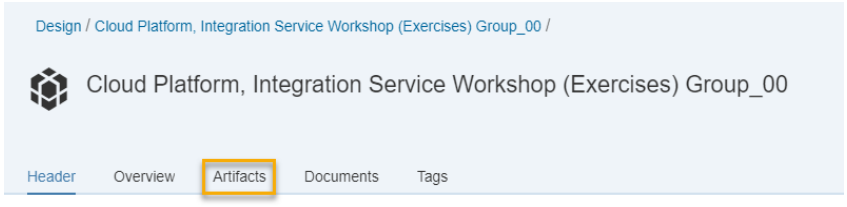
b. Select **Exercise06a_XX** integration flow.

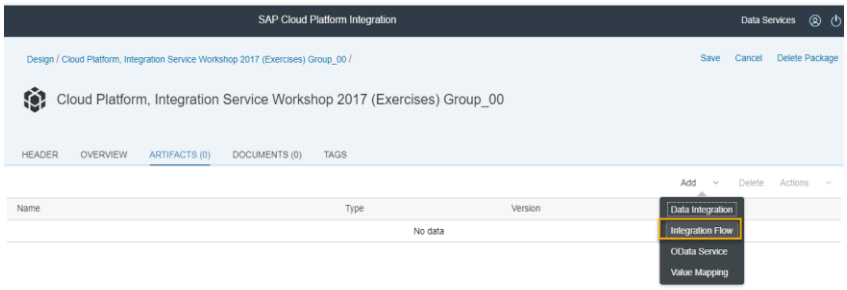
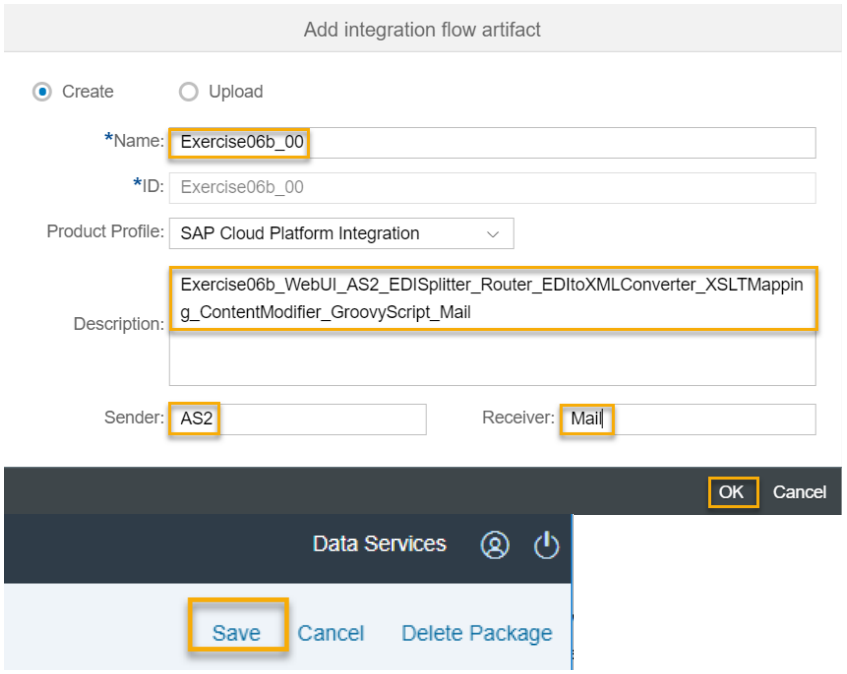
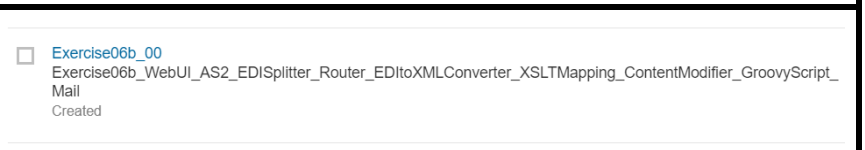
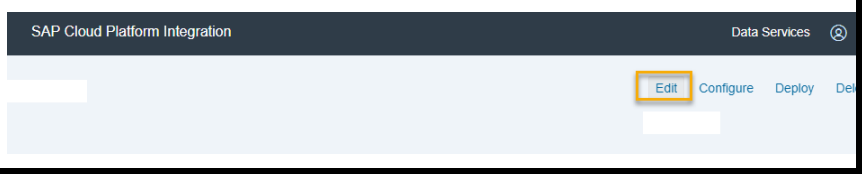
c. Click **Undeploy** button on the top right section.

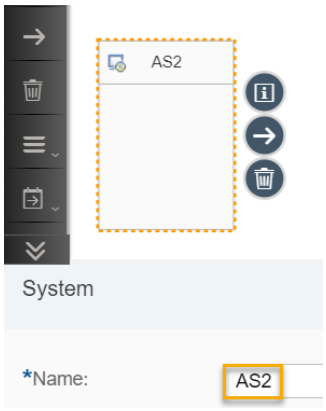
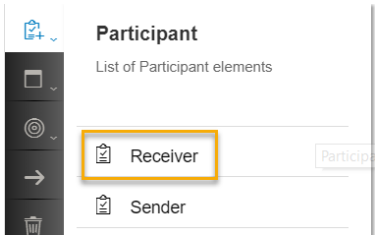
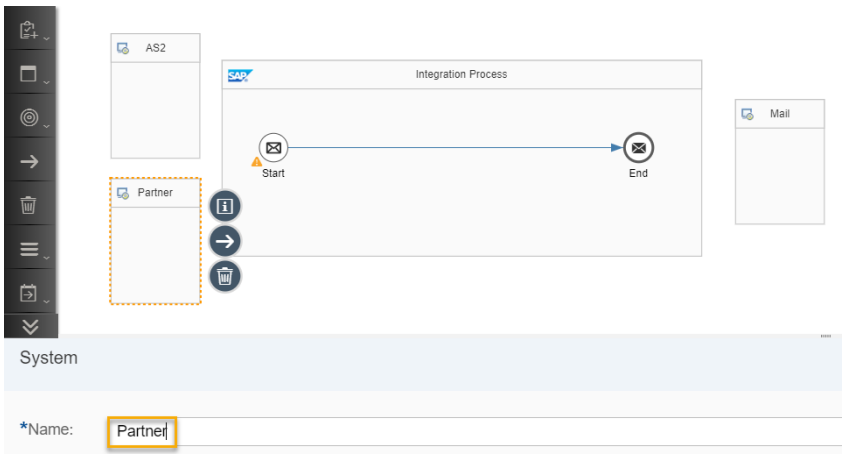
Note: This will help us to deploy the next part of this exercise without any issue. As next part of this exercise is having the same inbound message properties.



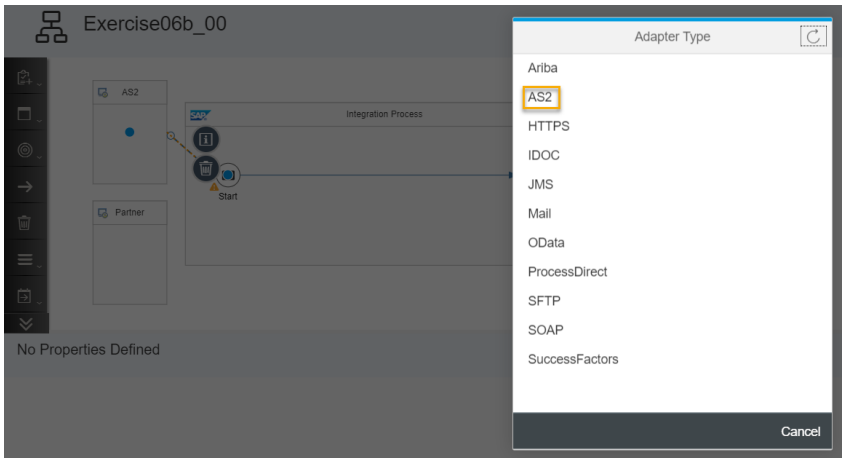
EXERCISE 06B

Explanation	Screenshot
Follow these steps for integration flow creation in Web UI.	
1) Navigate to **Design** tab	
2) Select the package	
3) Click on **Edit**	
4) Navigate to **Artifacts** Tab	

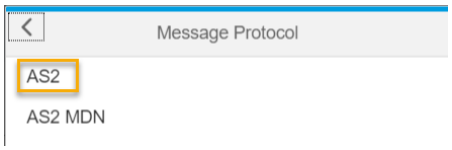
5) Click on ***Add-> Integration Flow***	
6) Select ***Create*** and enter following details: a. Name: ***Exercise06b_XX*** b. Description: ***Exercise06b_WebUI_AS2_EDISplitter_Router_EDItoXMLConverter_XSLTMapping_ContentModifier_GroovyScript_Mail*** c. Sender: ***AS2*** d. Receiver: ***Mail*** e. Click on ***OK*** **Note:** Please remember to save after every action	
7) Click on **Exercise06b_XX**	
Define and edit integration flow	**Follow steps to edit the integration flow**
8) Click on ***Edit*** button on the upper corner on right hand side.	

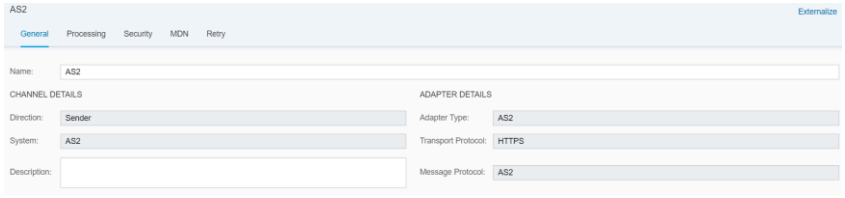
9) Click on Sender system and set Name as ***AS2***	
10) Click on Receiver System and set Name as ***Mail***	
11) From the palette, select ***Participants -> Receiver*** and drop it on iFlow editor window outside the Integration Process This receiver will route the 997 functional acknowledgments to the partner configured in the Mendelson platform	
12) Click on newly added Receiver system and set Name as ***Partner***	

13) Click on ***AS2*** sender system, select ***Connector*** and drag it to ***Start*** message step.
This would open the sender adapter list.



Select ***AS2*** adapter





14) Switch to ***Processing*** tab, enter the following details:

a. Message ID Left Part: **.***

b. Message ID Right Part: **.***

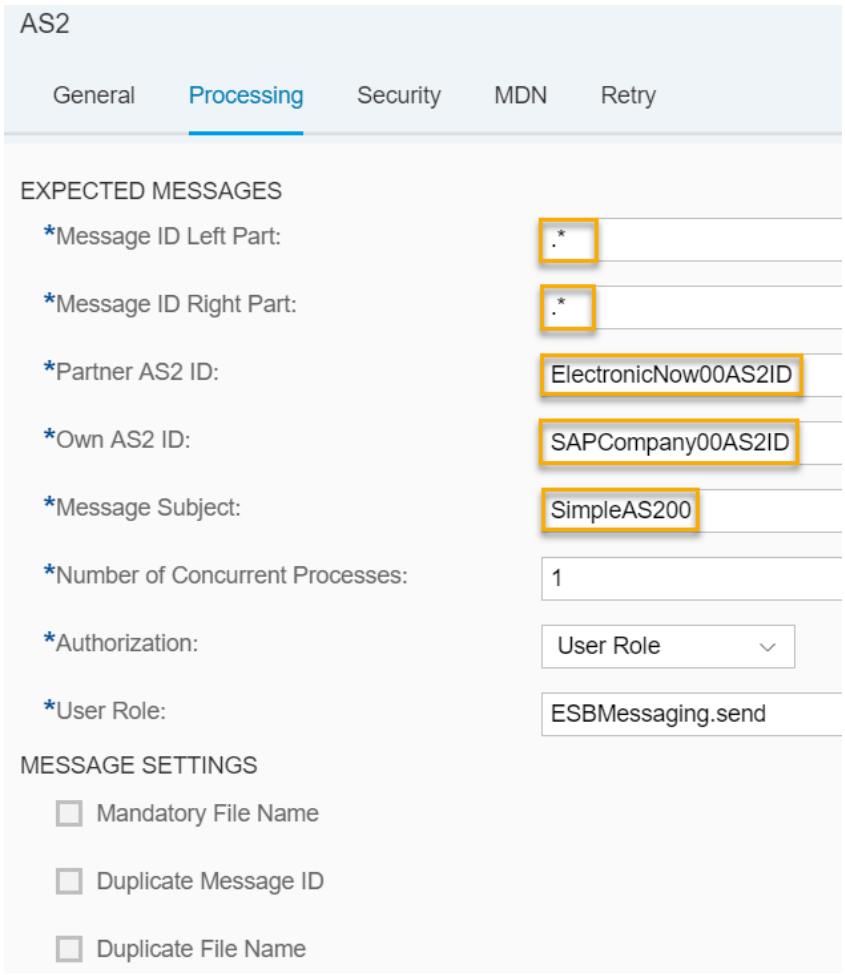
c. Partner AS2 ID: **ElectronicNowXXAS2ID**

d. Own AS2 ID: **SAPCompanyXXAS2ID**

e. Message Subject: **SimpleAS2XX**

Note: Replace **XX** with your participant id

Save



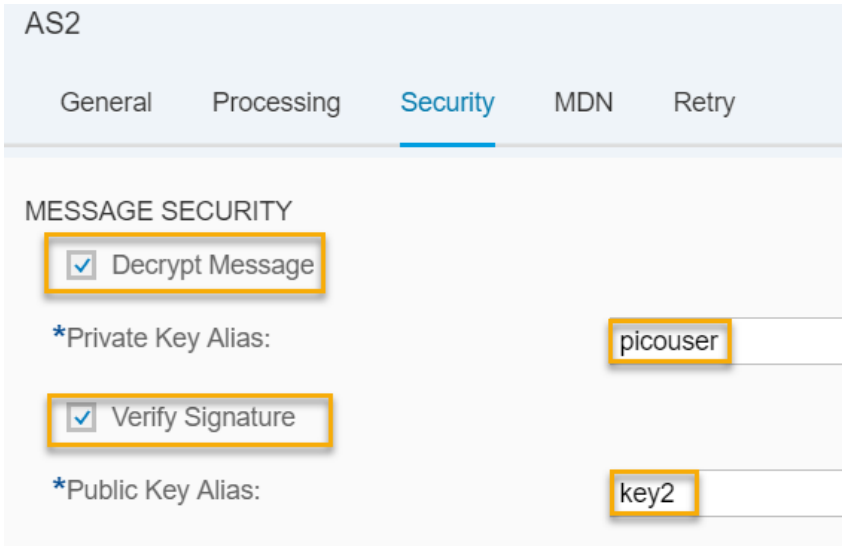
15) Switch to ***Security*** tab, enter the following details:

a. Decrypt Message: **checked**

b. Private Key Alias: **picouser**

c. Verify Signature: **checked**

d. Public Key Alias: **key2**



16) Switch to ***MDN*** tab, enter the following details:

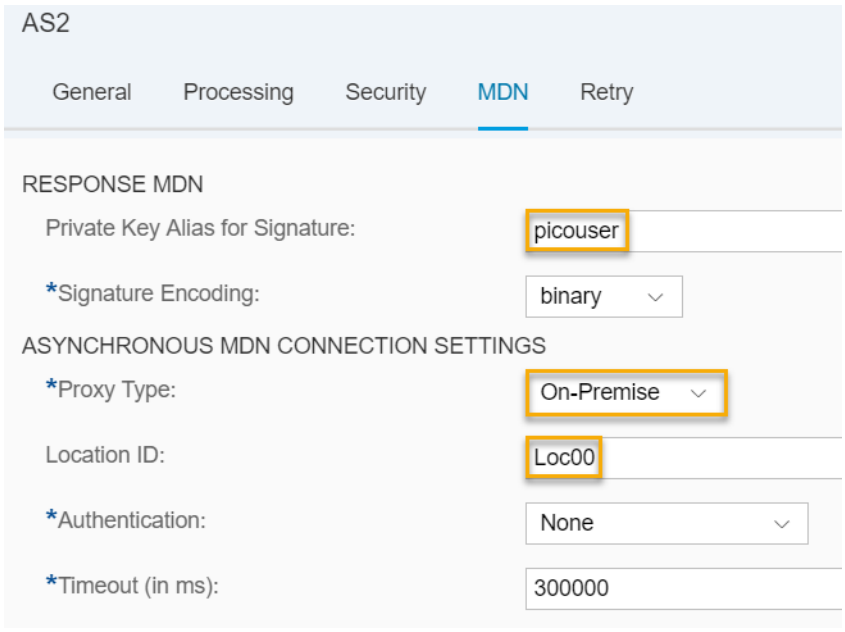
a. Private Key Alias for Signature: **picouser**

b. Proxy Type: **On-Premise**

c. Location ID: **LocXX**

Note: Replace **XX** with your participant id

Save



17) A groovy script is provided which writes AS2 EDI messages in MPL which is quite useful for debugging purpose. It need to be added after **Start** message step in iFlow window using the following steps:

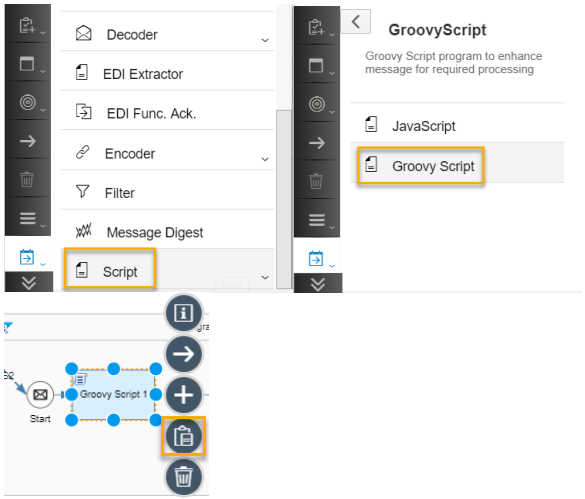
a. From palette, select **Message Transformers** -> **Script** -> **Groovy Script** and drop it on the connection between ***Start*** message and ***End*** message in the integration flow. This would automatically create connections.

b. Click on ***Assign*** icon .

c. Upload ***Log_Payload.gsh*** file from file system provided by the instructor.

d. Rename the script as **Log AS2 Message**

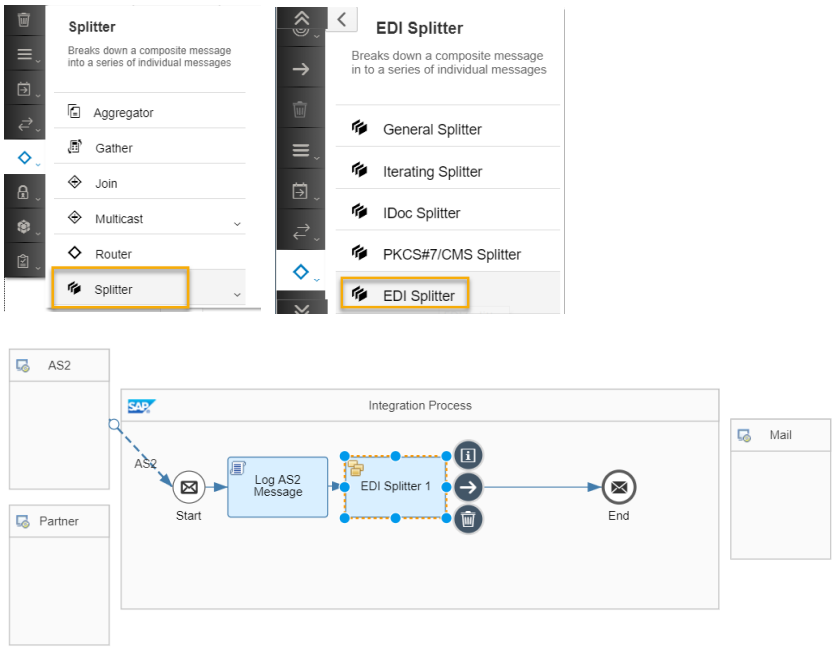
Save



Log_Payload.gsh

```
1  import com.sap.gateway.ip.core.customdev.util.Message;
2  import java.util.HashMap;
3
4  def Message processData(Message message) {
5      def body = message.getBody(java.lang.String) as String;
6      def messageLog = messageLogFactory.getMessageLog(message);
7      messageLog.addAttachmentAsString("Payload", body, "text/plain");
8
9      return message;
10 }
```

18) From the palette, select ***Message Routing -> Splitter -> EDI Splitter*** and drop it on the connection between ***Groovy Script*** and ***End*** message in the integration flow.



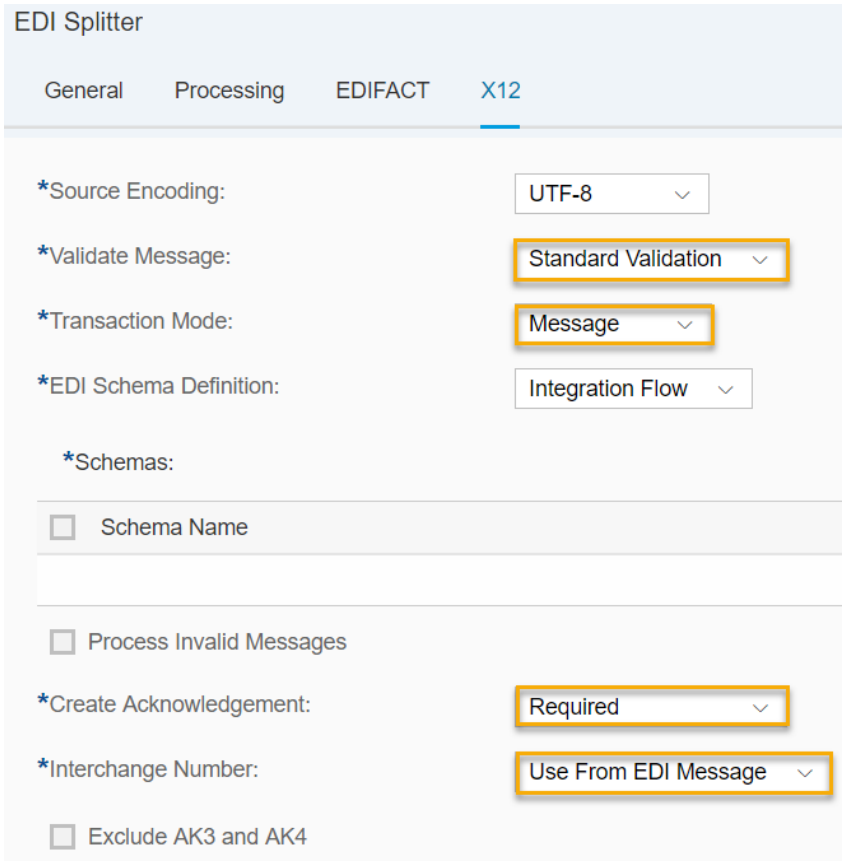
19) Switch to EDI Splitter ***X12*** tab, enter the following details:

a. Validate Message: **Standard Validation**

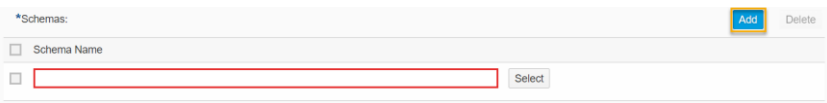
b. Transaction Mode: **Message**

c. Create Acknowledgement: **Required**

d. Interchange Number: **Use From EDI Message**



20) Under the same EDI Splitter ***X12*** tab, provide the **Schema** configuration by clicking on **Add** button



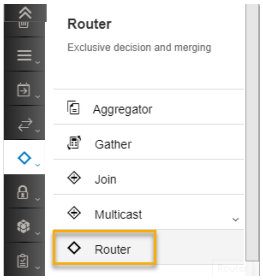
Click on **Select -> Upload from File System** and browse for the ***ASC-X12_850_004010.xsd*** file which is available as a part of the exercise folder

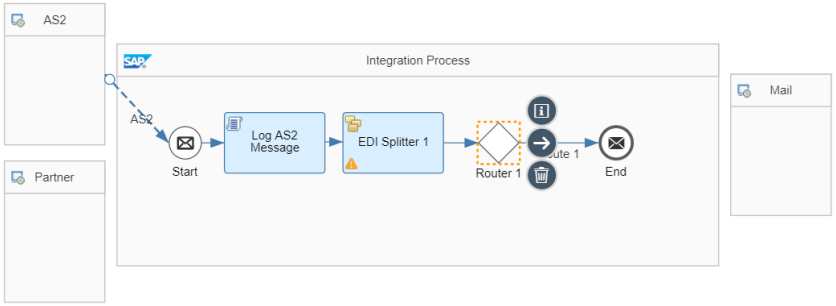
You can see the uploaded XSD file under Schemas

Save

21) From the palette, select ***Message Routing -> Router*** and drop it on the connection between ***EDI Splitter*** and ***End*** message in the integration flow.

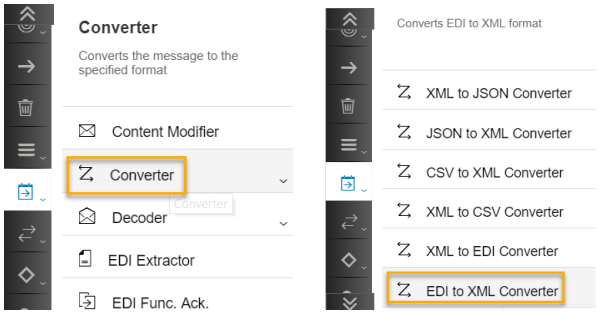
This is used to create a route for Acknowledgment and EDI valid messages.

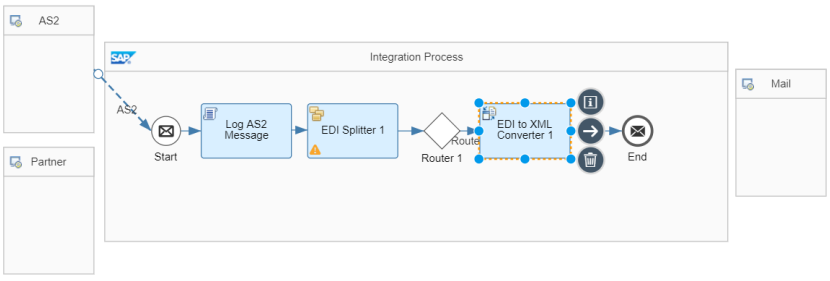




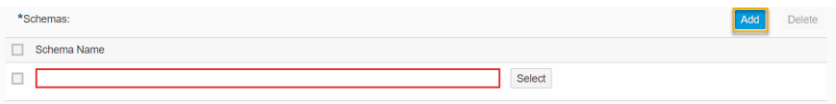
22) From the palette, select ***Message Transformers -> Converter -> EDI to XML Converter*** and drop it on the connection between ***Router*** and ***End*** message in the integration flow.

This is used to transform EDI message to XML format.





23) Switch to EDI to XML Converter ***X12*** tab, provide the **Schema**



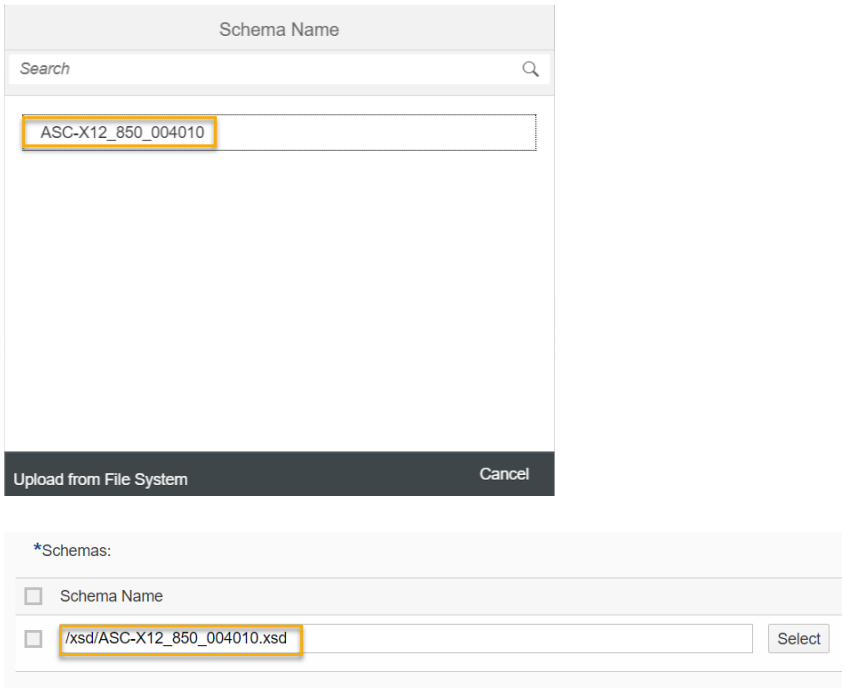
configuration by clicking on **Add** button.

Mainly we need to provide XSD for EDI message.

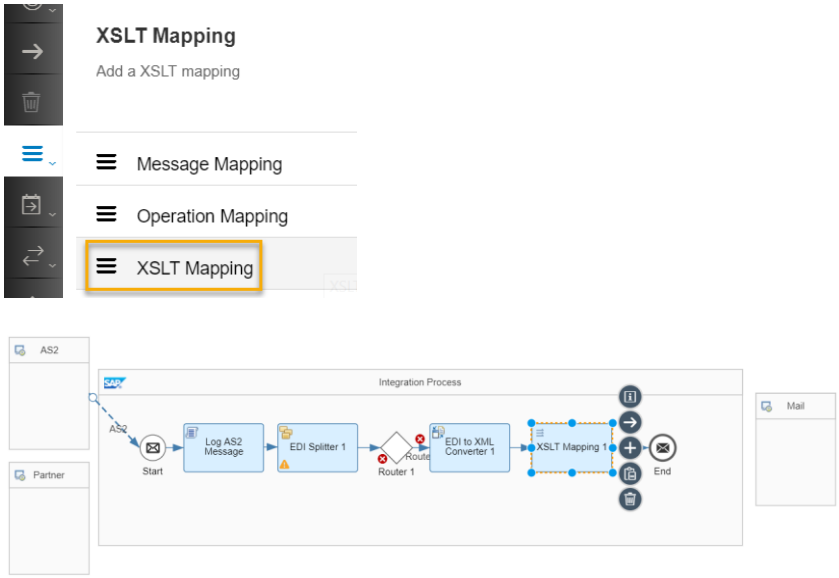
Click on **Select -> *ASC-X12_850_004010*** which is already referenced for EDI Splitter Schema mapping.

Save

Note: Ignore the errors on Integration Flow after save



24) From the palette, select ***Mapping -> XSLT Mapping*** and drop it on the connection between ***EDI to XML Converter*** and ***End*** message in the integration flow.

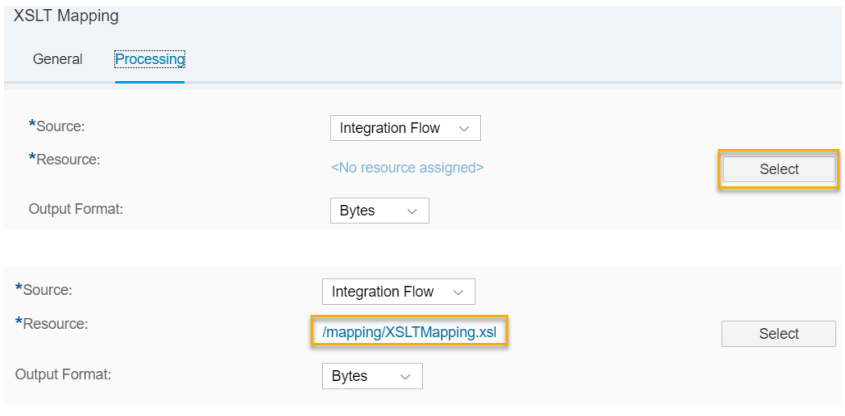


25) Switch to ***Processing*** tab, provide the **Resource** configuration.

Click on **Select -> Upload from File System** and browse for the ***XSLTMapping.xsl*** file which is available as a part of the exercise folder

You can see the uploaded XSL file under Mappings

Save



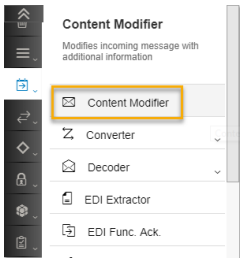
26) From the palette, select ***Message Transformers -> Content Modifier*** and drop it on the connection between ***XSLT Mapping*** and ***End*** message in the integration flow.

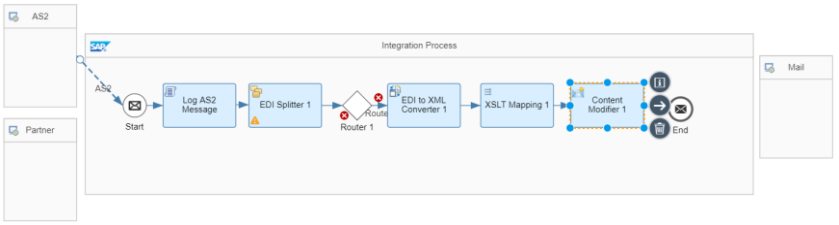
Switch to ***Message Body*** tab, enter the following text as Body:

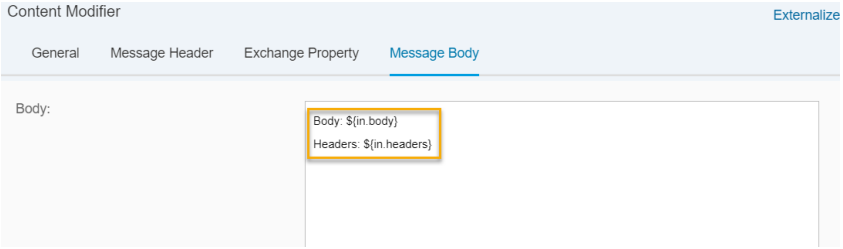
Body:
${in.body}

Headers:
${in.headers}

Save







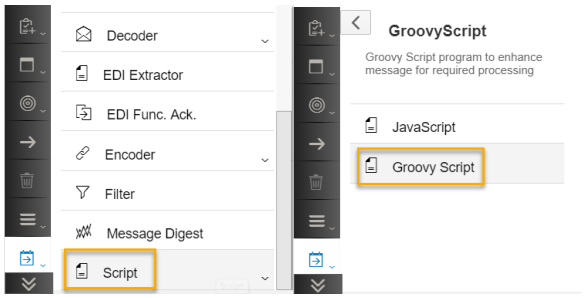
27) A groovy script is provided which writes IDOC Orders in MPL which is quite useful for debugging purpose. It need to be added after **Content Modifier** message step in iFlow window using the following steps:

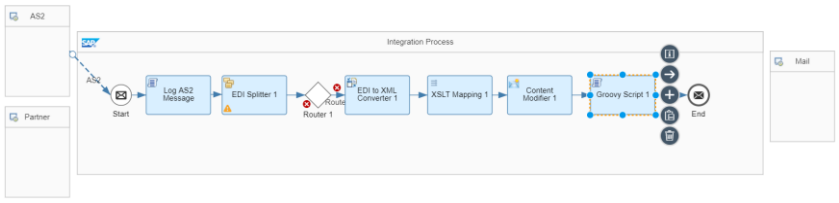
a. From palette, select **Message Transformers** -> **Script** -> **Groovy Script** and drop it on the connection between ***Content Modifier*** and ***End*** message in the Integration flow.

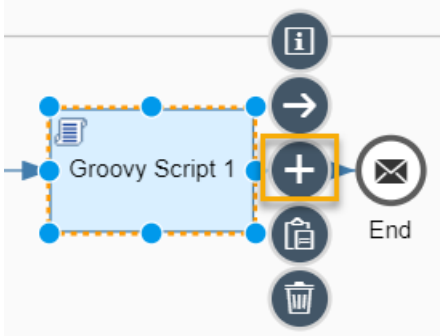
b. Click on ***Create*** icon ⊕.

c. Replace the code with following code:

```
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);
```







messageLog.addAttachmentAsString(" Log IDoc Orders", body, "text/plain");

return message;
}

d. Click on **OK**

e. Rename the script as **Log IDOC Orders**

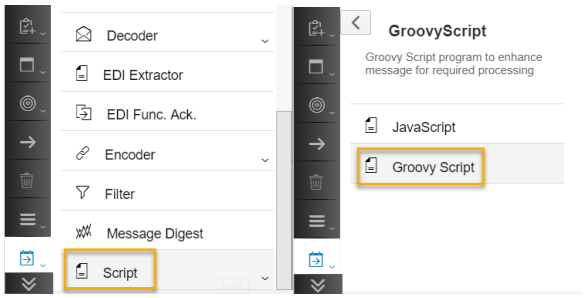
Save

script1.groovy

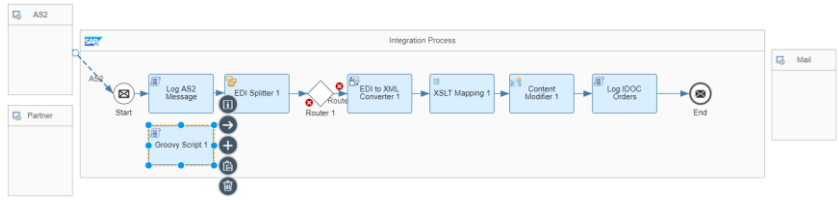
```groovy
import com.sap.gateway.ip.core.customdev.util.Message;
import java.util.HashMap;

def Message processData(Message message) {
    def body = message.getBody(java.lang.String) as String;
    def messageLog = messageLogFactory.getMessageLog(message);
    messageLog.addAttachmentAsString(" Log IDoc Orders", body, "text/plain");

    return message;
}
```

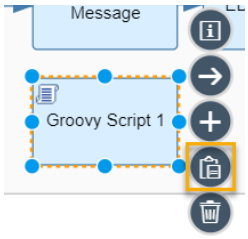
28) A groovy script is provided which writes AS2 997 Acknowledgment messages in MPL which is quite useful for debugging purpose. It need to be added in Acknowledgment route which we will create.
Add groovy script in iFlow window using the following steps:

a. From palette, select **Message Transformers** -> **Script** -> **Groovy Script** and drop it in the integration flow below the **Log AS2 Message** groovy script flow step.



b. Click on ***Assign*** icon .





c. Upload ***Log_Functional_Acknowledgement.gsh*** file from file system provided by the instructor.

d. Rename the script as **Log Functional Acknowledgement**

Save

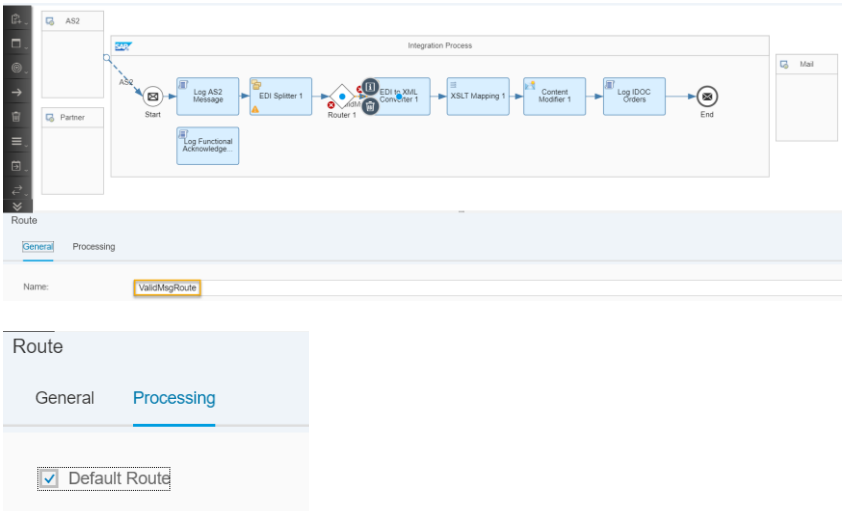
Log_Functional_Acknowledgement.gsh

```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);
    messageLog.addAttachmentAsString("Log Functional Acknowledgement", body, "text/plain");

    return message;
}
```

29) Select connection between ***Router*** and ***EDI to XML Converter*** and configure the route with following values:

a. Name: ***ValidMsgRoute***
b. Check ***Default Route***

Save

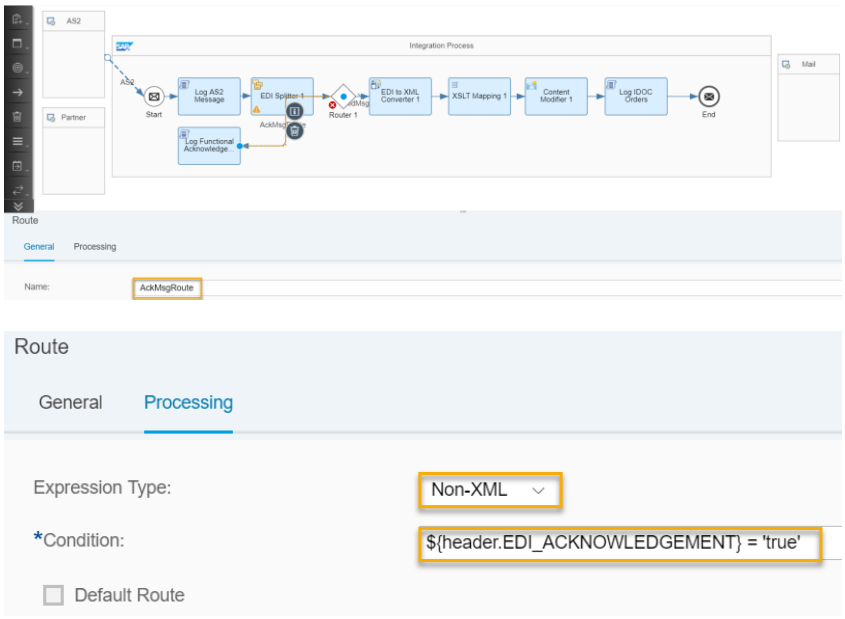


30) Now create an acknowledgement message route for EDI Splitter to send 997 functional acknowledgment to customer for processed EDI message by connecting Router's non- Default route to Log Functional Acknowledgment script.

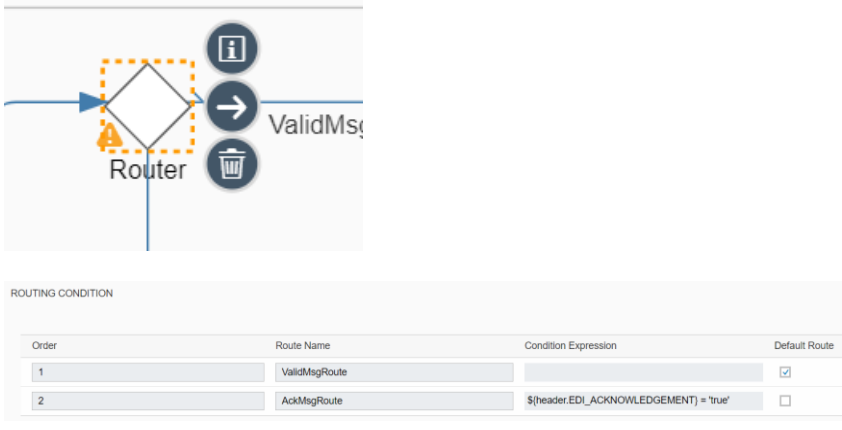
Draw the connection from ***Router*** to ***Log Functional Acknowledgment*** and configure the route with following values:

a. Name: ***AckMsgRoute***
b. Expression Type: ***Non-XML***
c. Condition: ***${header.EDI_ACKNOWLEDGEMENT} = 'true'***

Save

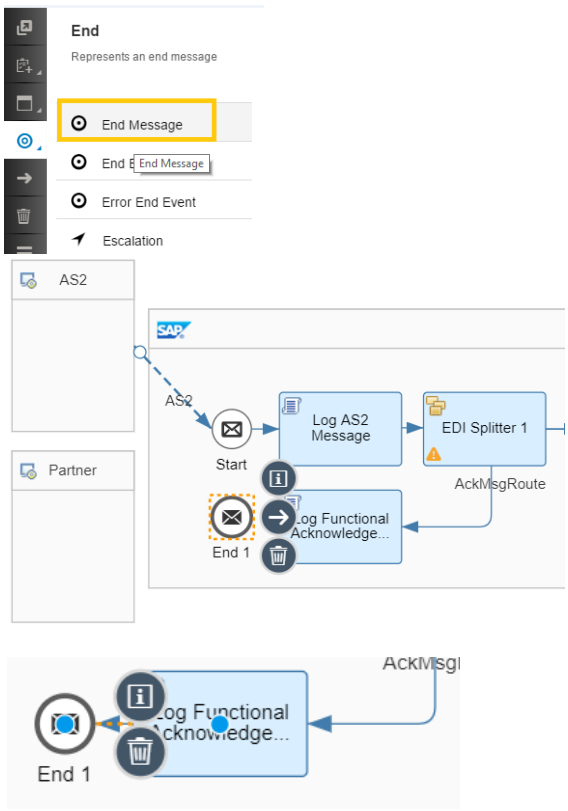


31) Now we can see two routing conditions for the router.



32) Click ***Events***, select ***End Message*** and drop it on the integration flow

Connect ***Log Functional Acknowledgment*** groovy script to ***End*** Message event.



33) Click on ***End 1*** message step, select ***Connector*** and drag it to ***Partner*** receiver system.
This would open the receiver adapter list.

Select **AS2** adapter

34) Switch to ***Connection*** tab, enter the following details:

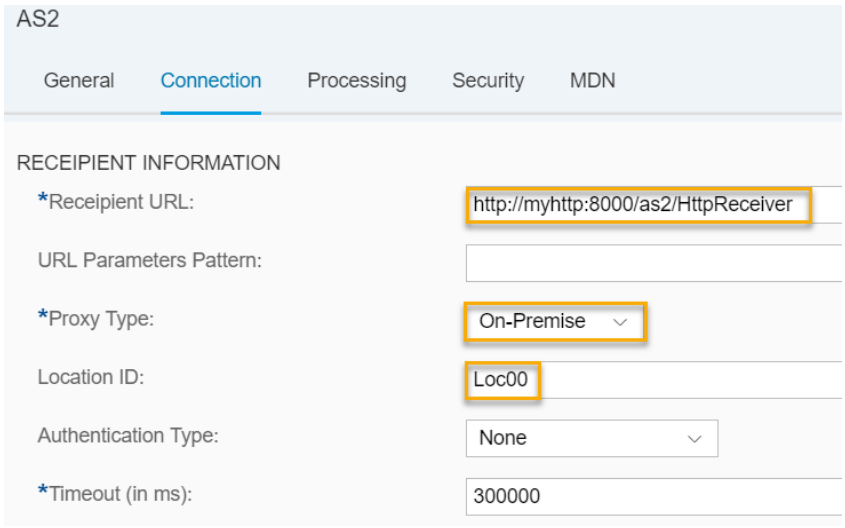
a. Recipient URL: http://<Virtual Host>:<Virtual Port>/as2/HttpReceiver
E.g. http://myhttp:8000/as2/HttpReceiver

b. Proxy Type: **On-Premise**

c. Location ID: **LocXX**

Note: Here **Virtual Host** and **Virtual Port** are the virtual host and port addresses which you have entered in Cloud Connector.

Replace **XX** with your participant id



35) Switch to ***Processing*** tab, enter the following details:

a. Own AS2 ID: **SAPCompanyXXAS2ID**

b. Partner AS2 ID: **ElectronicNowXXAS2ID**

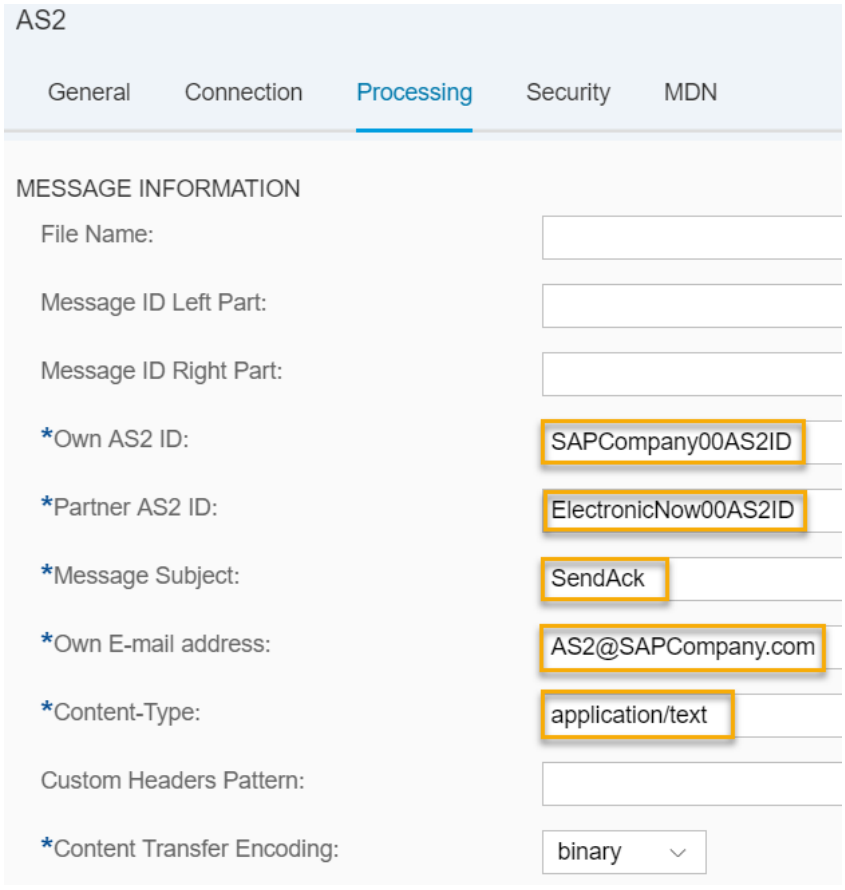
c. Message Subject: **SendAck**

d. Own E-mail address: **AS2@SAPCompany.com**

e. Content-Type: **application/text**

Note: Replace **XX** with your participant id

Save



36) Switch to ***Security*** tab, enter the following details:

a. Compress Message: **checked**

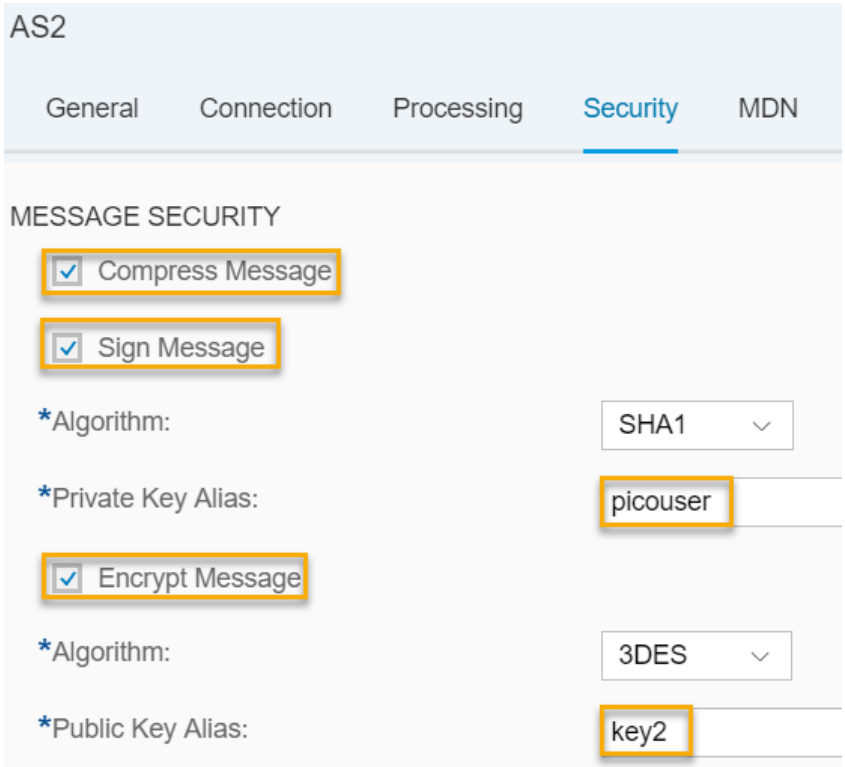
b. Sign Message: **checked**

c. Private Key Alias: **picouser**

d. Encrypt Message: **checked**

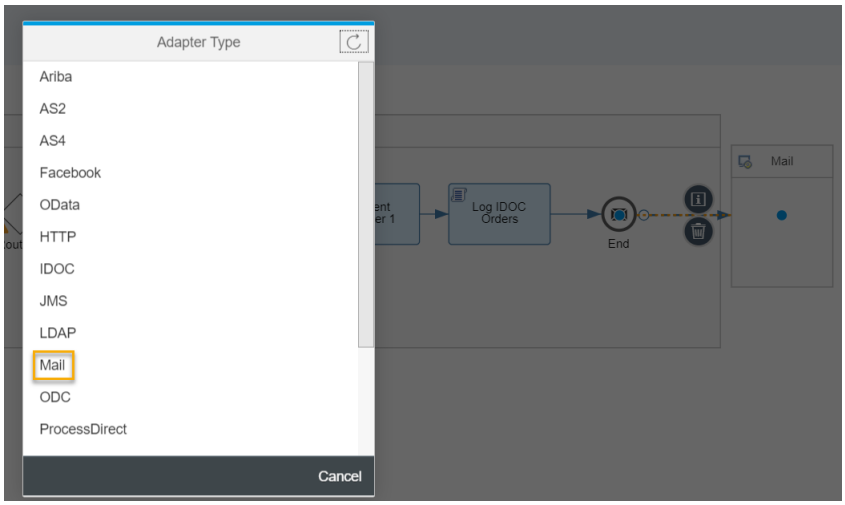
e. Public Key Alias: **key2**

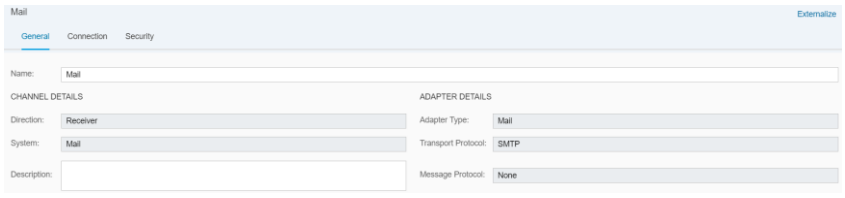
Save

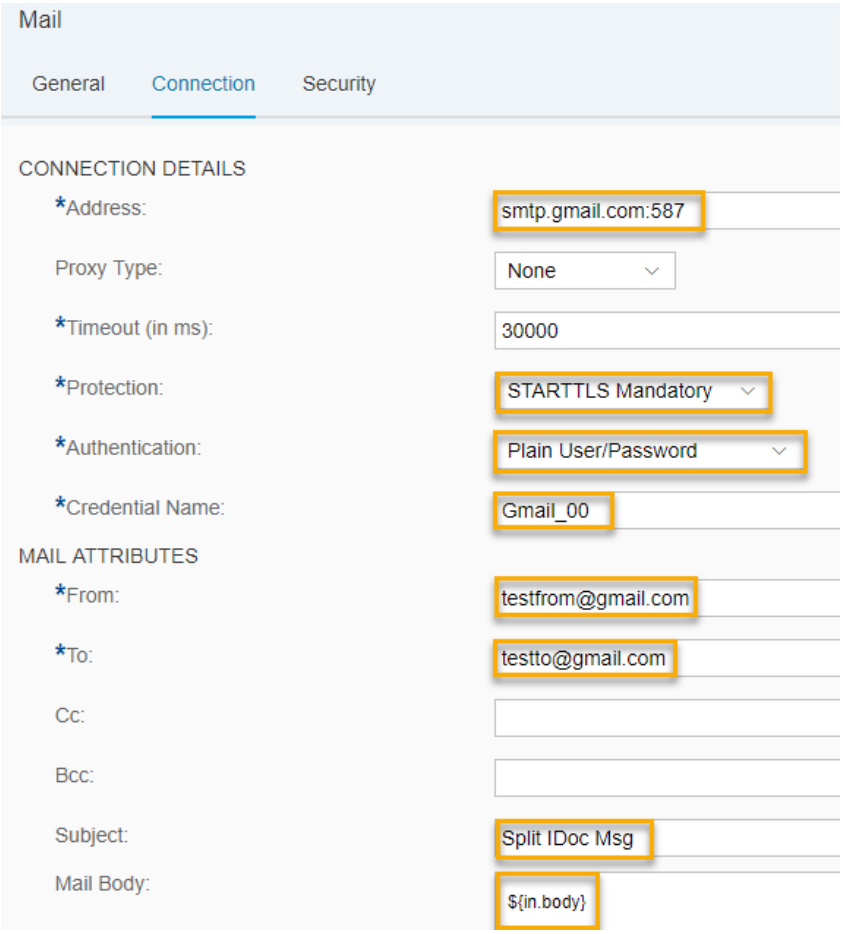
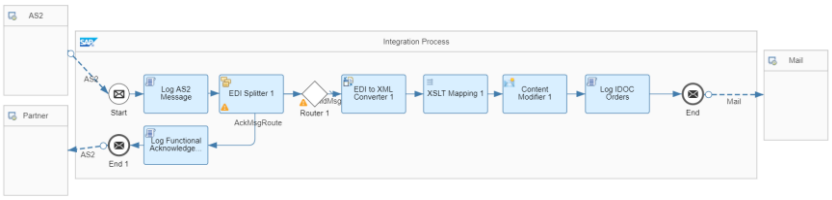
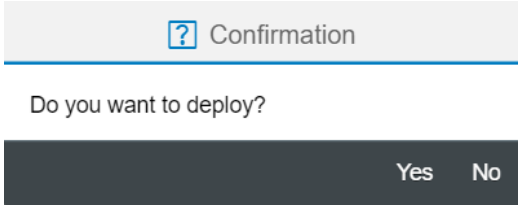


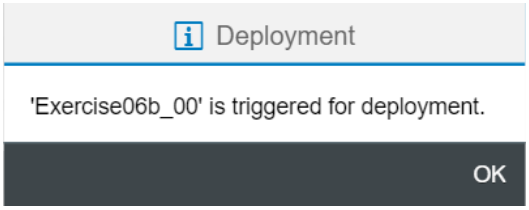
37) Click on ***End*** message step, select ***Connector*** and drag it to ***Mail*** receiver system.
This would open the receiver adapter list.

Select **Mail** adapter





38) Switch to ***Connection*** tab, enter the following details: a. Address: ***smtp.gmail.com:587*** b. Protection: ***STARTTLS Mandatory*** *c.* Authentication: ***Plain User/Password*** d. Credential Name: ***Gmail_XX*** *e. From: Mail ID from where you want to send the mail. (eg. testfrom@gmail.com)* *f.* To: *Mail ID where you want to receive the mail. (eg. testto@gmail.com)* *g.* Subject: ***Split IDoc Msg*** Save	
39) Your integration flow should look like this.	
Deploy Integration Project on tenant:	**Follow this step to deploy Integration Project on tenant.**
40) Press **Deploy** in the Integration Flow Task Bar.	
41) You will receive a confirmation. Click on ***Yes***. Once the deployment is completed successfully you will receive a 2nd notification.	

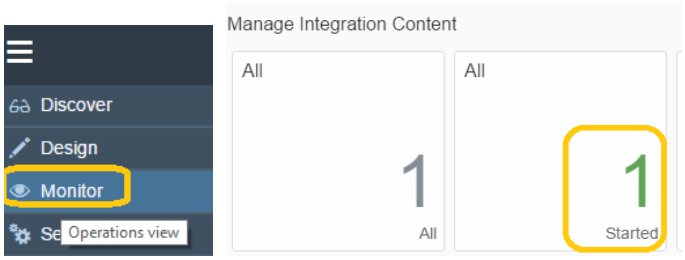


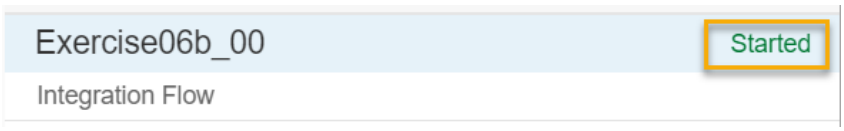
42) Verify if the deployment is successful:

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.

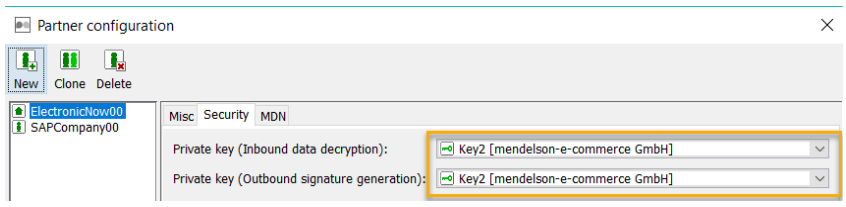
Check the 'status'. It should be in status ***Started***.





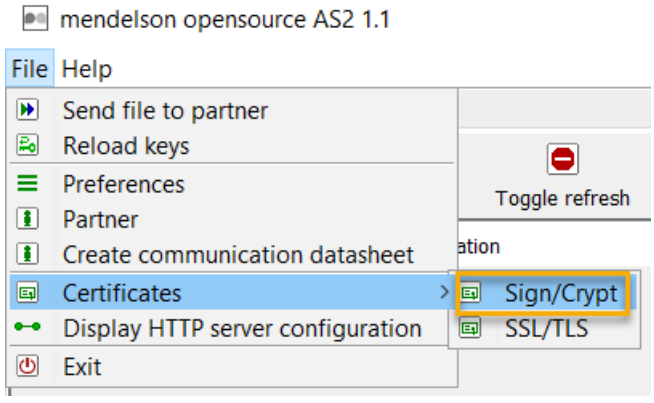
Execute end to end scenario:	**Follow steps to execute end to end scenario.**

43) Modify the Mendelson AS2 local station Customer **ElectronicNowXX** security configuration and select **Key2** for both private key inbound/outbound security configuration.

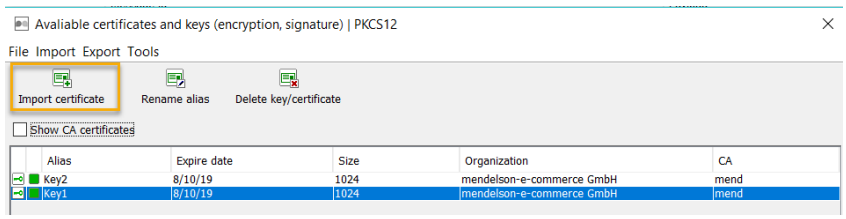


44) Import picouser certificate on to the Mendelson tool

a. In the Mandelson tool navigate to **File -> Certificates -> Sign/Crypt**

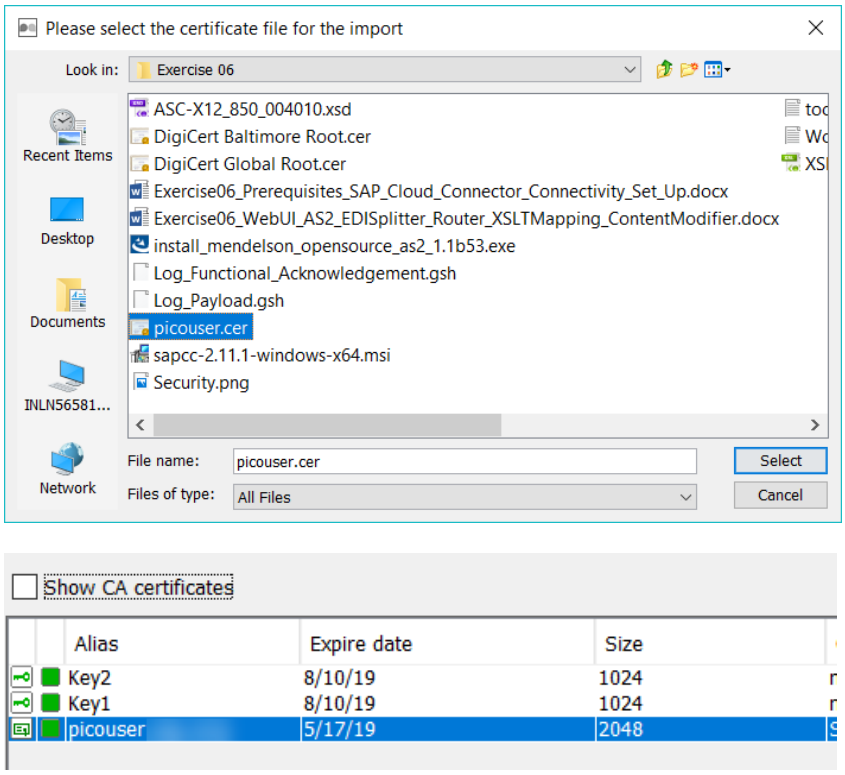


b. Click on import certificate

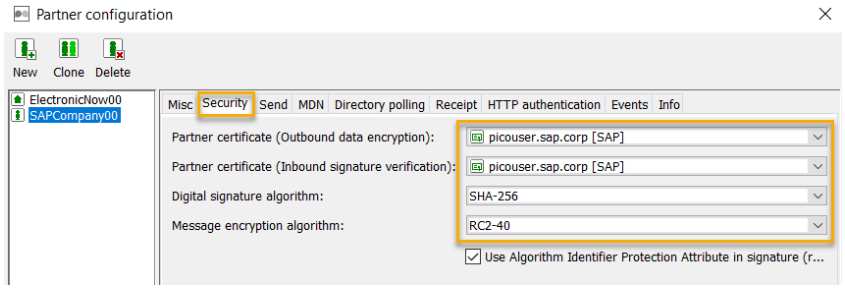


c. Click on Browse and from the Exercise folder, import the certificate **picouser.cer** file.

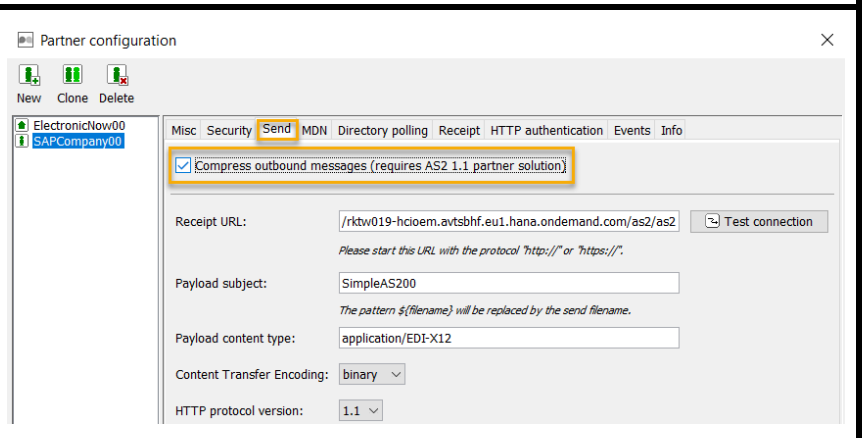
This import the SAP cloud platform integration certificate used for encryption and sign validation



45) Choose the above imported certificate (picouser) for encryption and signature certificates and algorithms for SAPCompanyXX partner security.

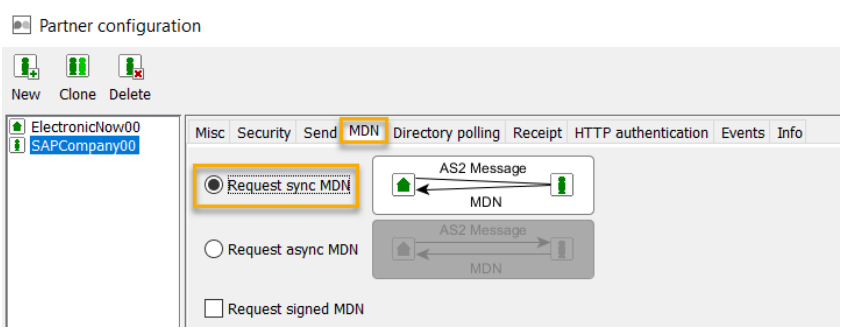


46) Select the Compress outbound messages in **Send** settings



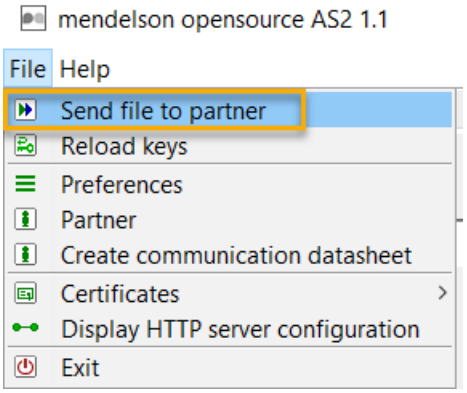
47) Request sync MDN option in **MDN** settings.

Click on **Ok**

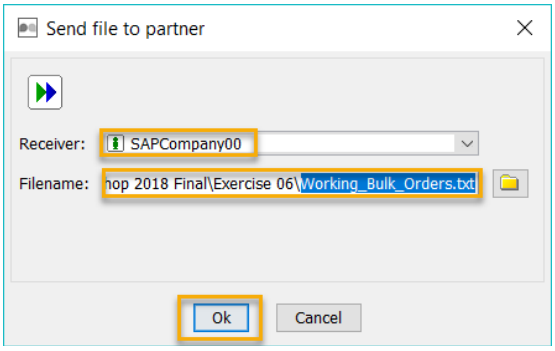


48) Let's test a simple AS2 message post from Mendelson to SAP Cloud Platform Integration Tenant

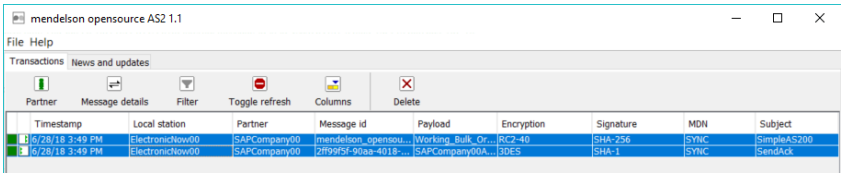
In the Mendelson tool, Choose **File -> Send file to partner**



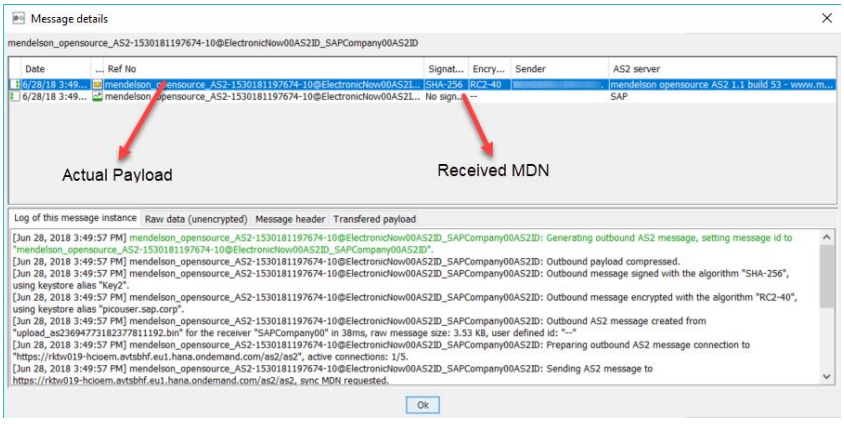
49) Choose **SAPCompanyXX** from the Receiver dropdown and select **Working_Bulk_Orders.txt** file and click Ok



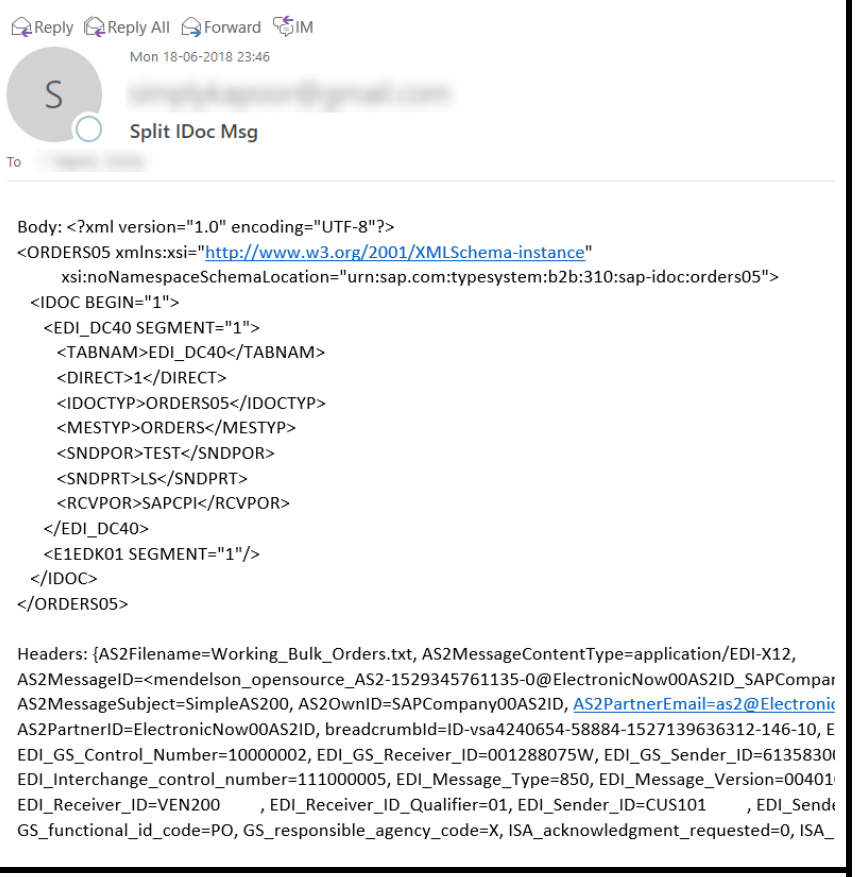
50) You should see two successful messages in the Mandelson tool as shown below



51) Double click on the message to see the message that was sent and the MDN received back

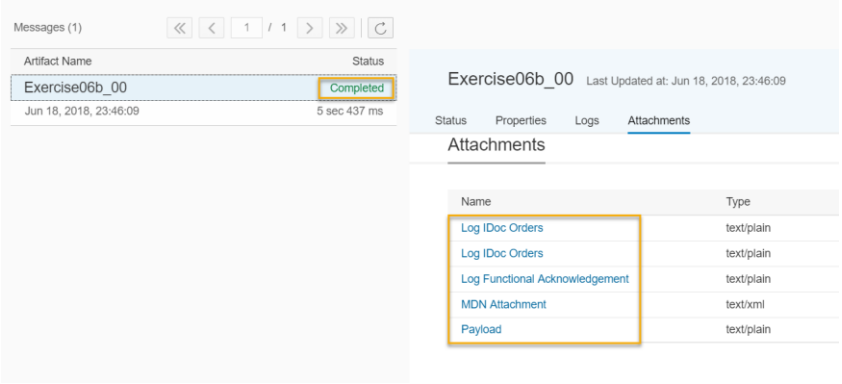


52) Check the Mail, you should receive a mail of IDOC SALES Orders in the configured mail account



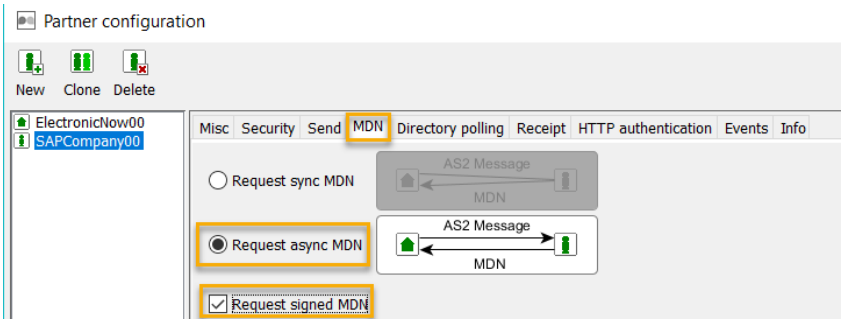
53) On the CPI tenant, Navigate to **Monitor Message processing** to see a Completed message with an MPL attachments.

Note: you can validate Logs for information on EDI messages being sent, IDOC Sales Orders created and Functional Acknowledgments delivered to partner



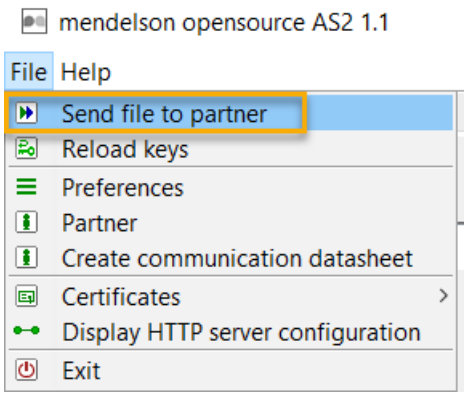
54) Request async MDN option in **MDN** settings.

Click on **Ok**

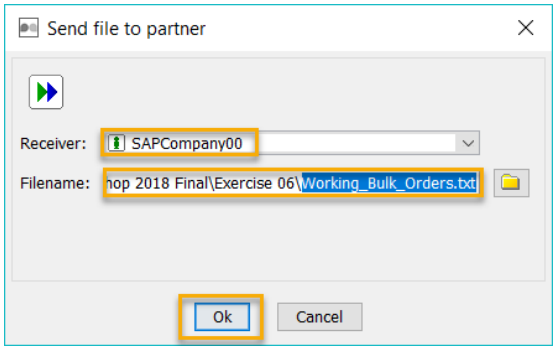


55) Let's test a simple AS2 message post from Mendelson to SAP Cloud Platform Integration Tenant

In the Mendelson tool, Choose **File -> Send file to partner**

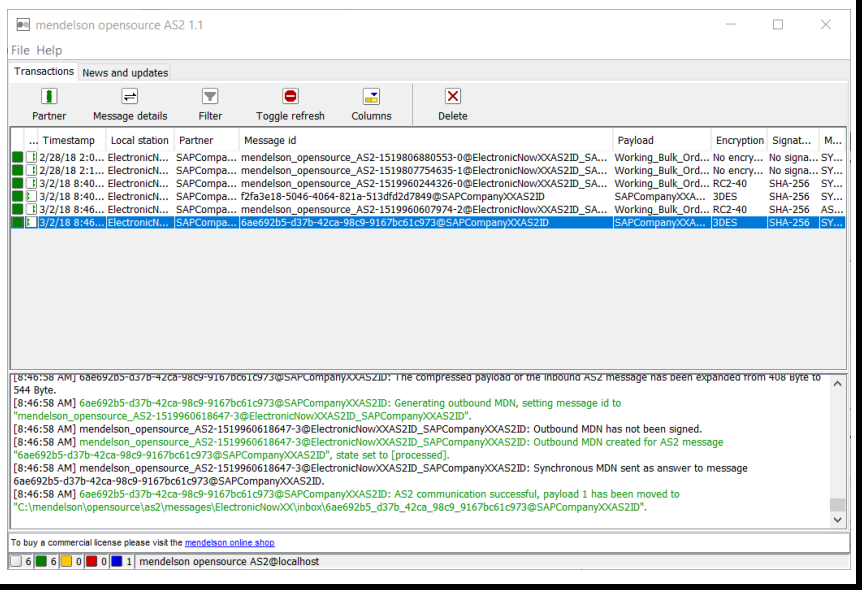


56) Choose **SAPCompanyXX** from the Receiver dropdown and select **Working_Bulk_Orders.txt** file and click Ok

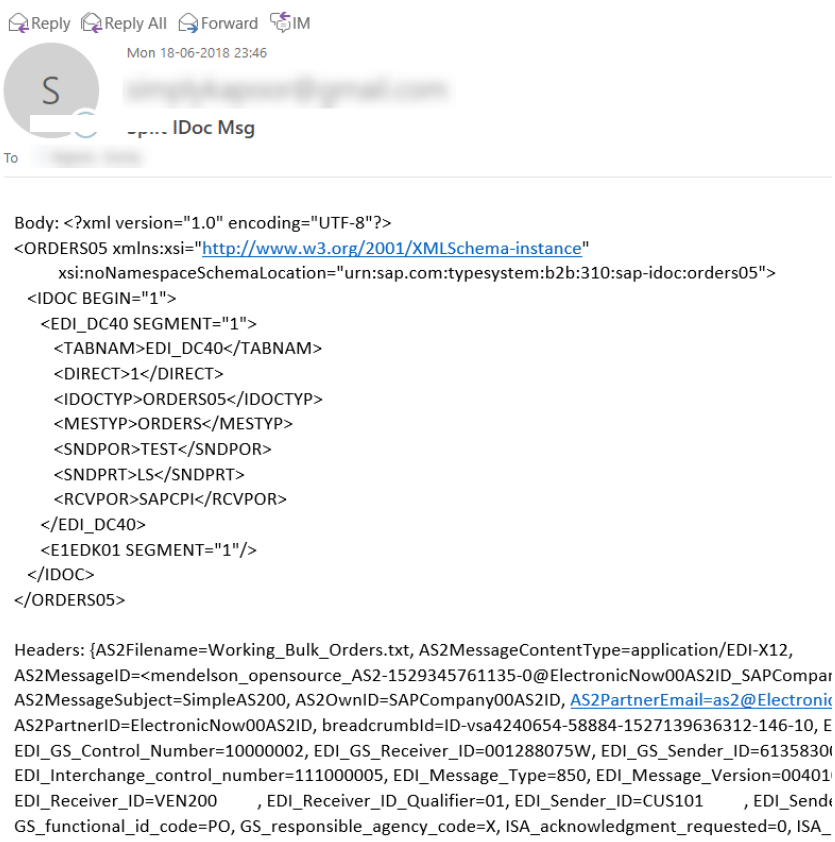


57) You should see a successful message in the Mandelson tool as shown below.

Note: Message in green indicates, successful post



58) Check the Mail, you should receive a mail of IDOC SALES Orders in the configured mail account in Asynchronous MDN use case as well.



59) On the CPI tenant, Navigate to **Monitor Message processing** to see a Completed message with an MPL attachments.

Note: you can validate Logs for information on EDI messages being sent, IDOC Sales Orders created and Functional Acknowledgments delivered to partner