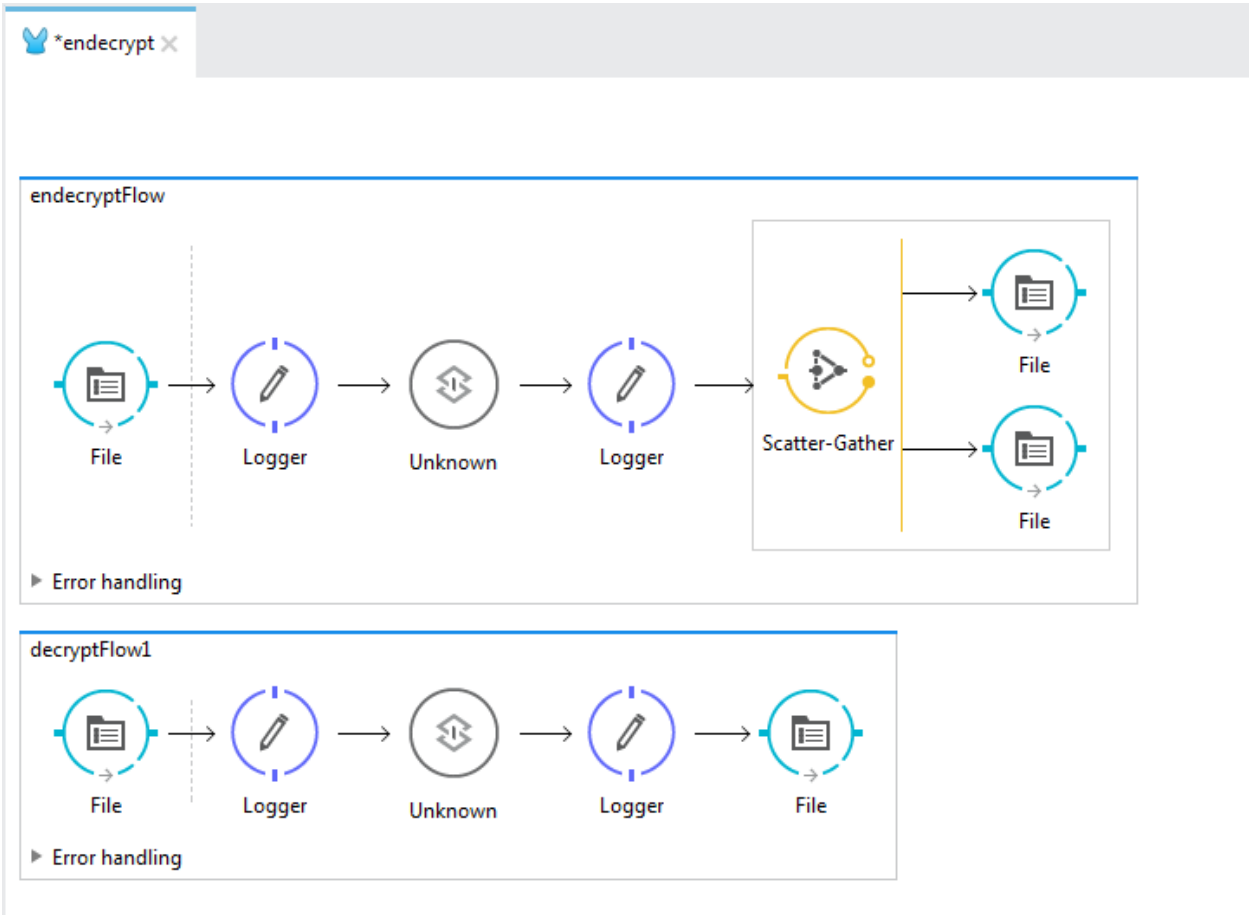
LAB – Scatter gather & base64

Encrypt and decrypt a file using Scatter and Gather.

Steps:

- Open AnyPoint Studio
- File → New → Mule Project
- Give the project name as endecrypt as mentioned below. And then finish.
- Create the below flow to encrypt and decrypt file content.



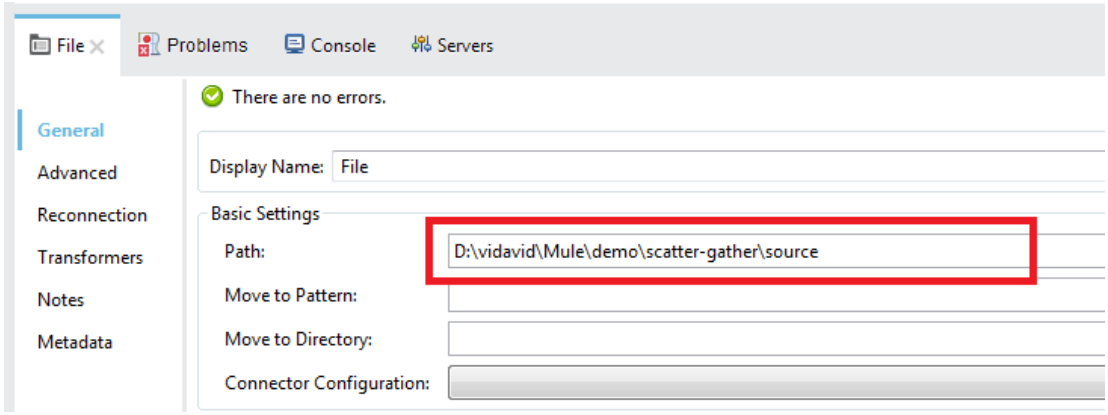
- The equivalent configuration file also mentioned below:

configuration.xml

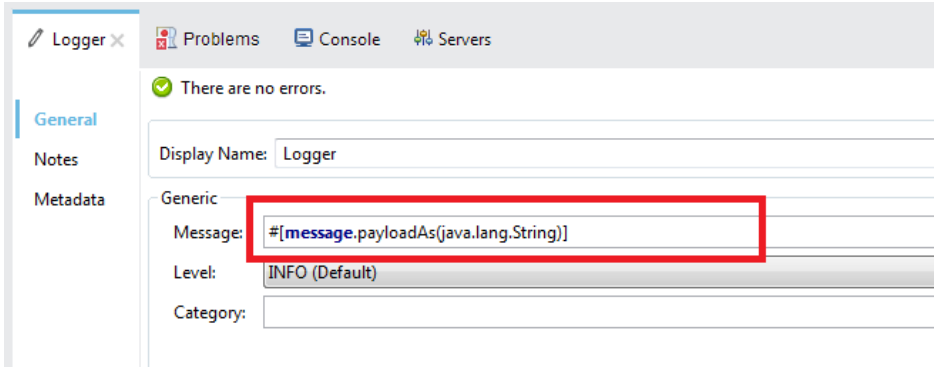
```
<?xml version="1.0" encoding="UTF-8"?>

<mule xmlns:tracking="http://www.mulesoft.org/schema/mule/ee/tracking"
xmlns:file="http://www.mulesoft.org/schema/mule/file" xmlns="http://www.mulesoft.org/schema/mule/core"
xmlns:doc="http://www.mulesoft.org/schema/mule/documentation"
        xmlns:spring="http://www.springframework.org/schema/beans"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:schemaLocation="http://www.springframework.org/schema/beans
http://www.springframework.org/schema/beans/spring-beans-current.xsd
http://www.mulesoft.org/schema/mule/core http://www.mulesoft.org/schema/mule/core/current/mule.xsd
http://www.mulesoft.org/schema/mule/file http://www.mulesoft.org/schema/mule/file/current/mule-file.xsd
```

```
http://www.mulesoft.org/schema/mule/ee/tracking
http://www.mulesoft.org/schema/mule/ee/tracking/current/mule-tracking-ee.xsd">
    <flow name="endecryptFlow">
        <file:inbound-endpoint responseTimeout="10000" doc:name="File" path="D:\vidavid\Mule\demo\scatter-
gather\source"/>
        <logger message="#[message.payloadAs(java.lang.String)]" level="INFO" doc:name="Logger"/>
          <base64-encoder-transformer  xmlns="http://www.mulesoft.org/schema/mule/core" encoding="utf8"/>
        <logger level="INFO" doc:name="Logger" message="#[message.payloadAs(java.lang.String)]"/>
        <scatter-gather doc:name="Scatter-Gather">
            <file:outbound-endpoint path="D:\vidavid\Mule\demo\scatter-gather\Encoded"
responseTimeout="10000" doc:name="File"/>
            <file:outbound-endpoint path="D:\vidavid\Mule\demo\scatter-gather\Encoded - Copy"
responseTimeout="10000" doc:name="File"/>
        </scatter-gather>
    </flow>
    <flow name="decryptFlow1">
        <file:inbound-endpoint responseTimeout="10000" doc:name="File" path="D:\vidavid\Mule\demo\scatter-
gather\Encoded"/>
        <logger level="INFO" doc:name="Logger" message="#[message.payloadAs(java.lang.String)]"/>
        <base64-decoder-transformer xmlns="http://www.mulesoft.org/schema/mule/core" encoding="utf8"/>

        <logger level="INFO" doc:name="Logger" message="#[message.payloadAs(java.lang.String)]"/>
        <file:outbound-endpoint responseTimeout="10000" doc:name="File"
path="D:\vidavid\Mule\demo\scatter-gather\Decoded"/>
    </flow>
</mule>
```

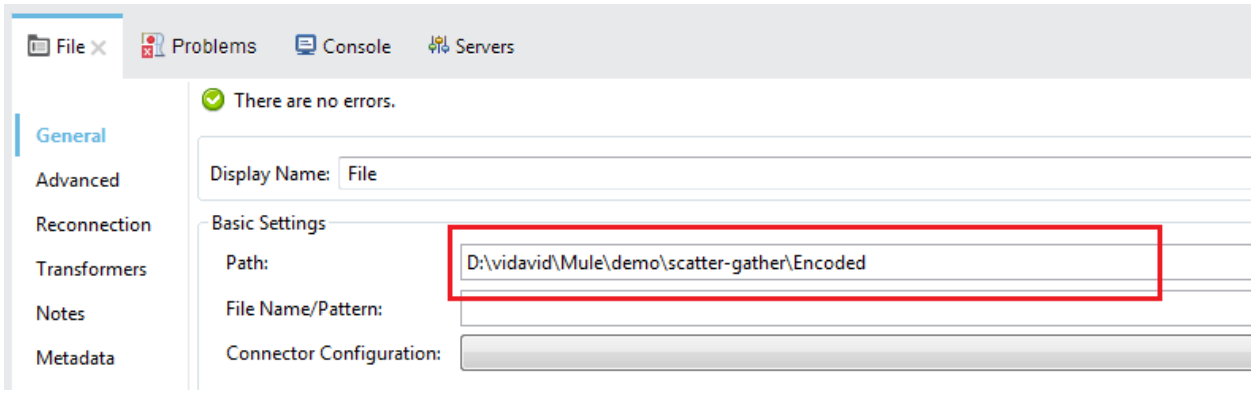
- In endecrypt flow mention the file path, from where you want to load the file.

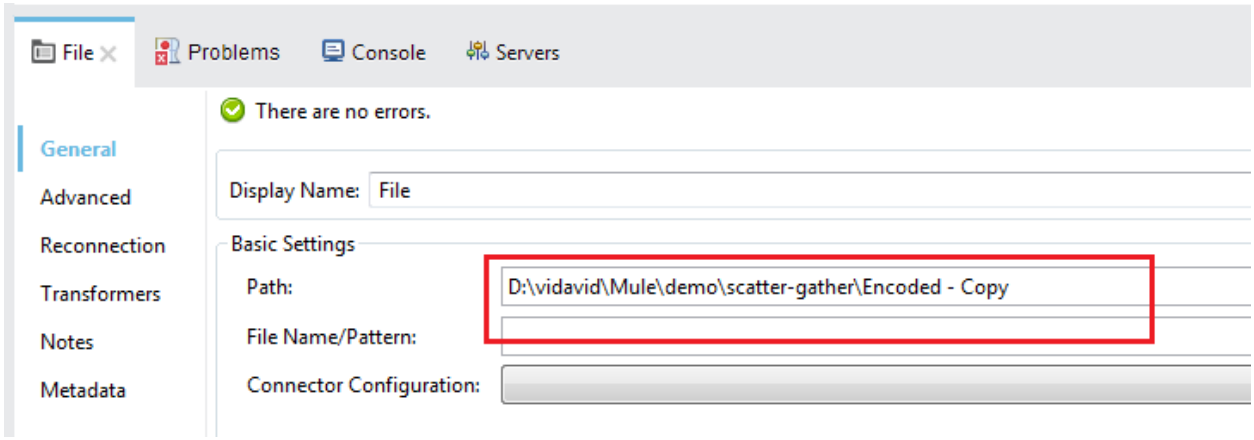


- In both logger property mention the message as highlighted below:

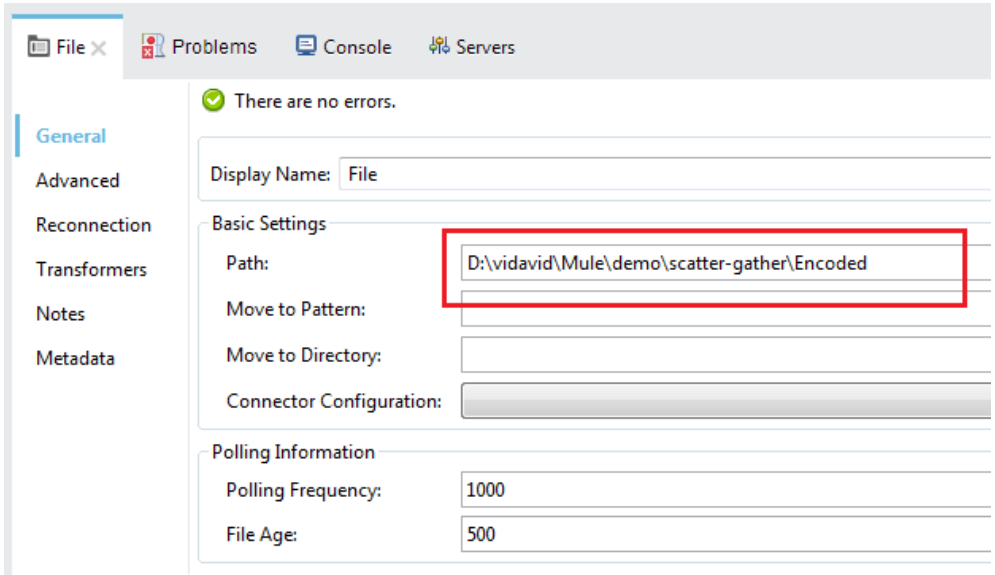


- Under the scatter and gather we configured the couple of file. The file path should be the location of the file where you want to decrypt the file.

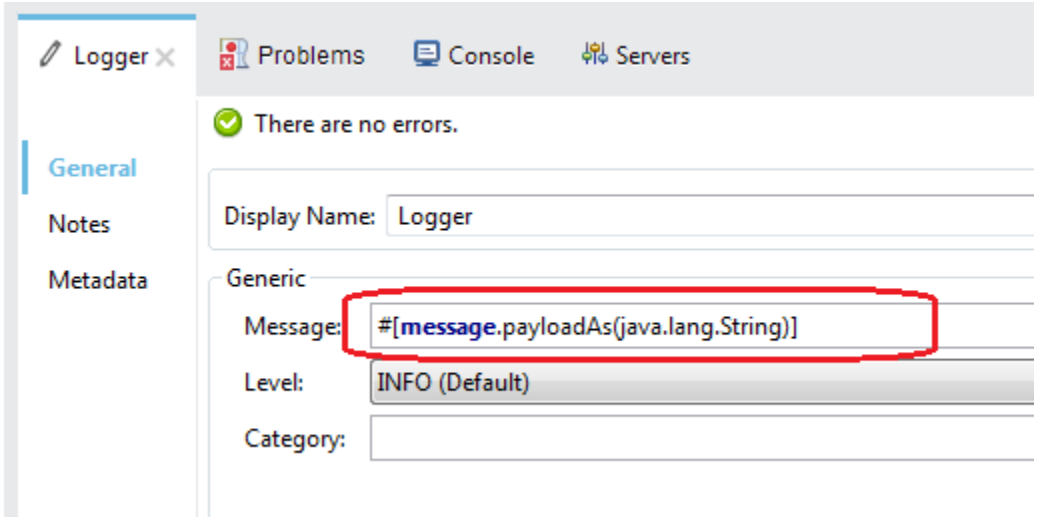




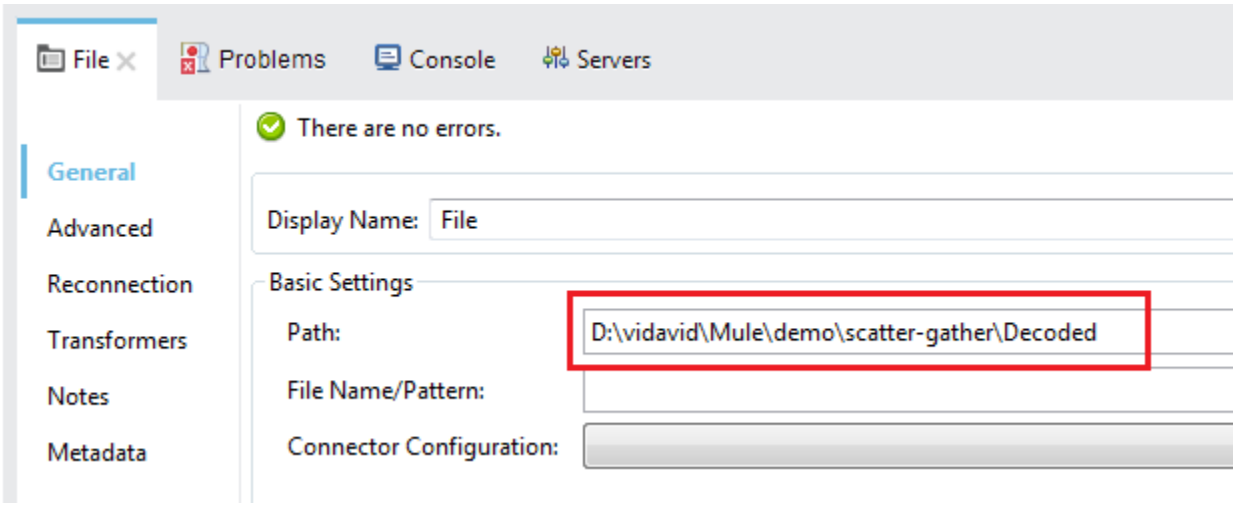
- In the decryptflow1 our file property path is one of the file paths of the Scatter and Gather component. It will provide the encoded file location to apply decryption.



- Mention BOTH logger message as mentioned BELOW:



- Here file property path is where you want to store decoded file.



- Right Click project → Run As→ Mule Application.
- Once Application started. Open postman plug-in to test the application.
- You will be getting the below log message in the console screen.

```
**********************************************************************
* Application: endecrypt                                             *
* OS encoding: \, Mule encoding: UTF-8                               *
*                                                                    *
* Agents Running:                                                    *
*   JMX Agent                                                        *
*   Batch module default engine                                      *
*   DevKit Extension Information                                     *
*   Wrapper Manager                                                  *
**********************************************************************
INFO  2016-10-07 18:49:16,970 [main] org.mule.module.launcher.MuleDeploymentService:
**********************************************************************
* Started app 'endecrypt'                                            *
**********************************************************************
INFO  2016-10-07 18:49:17,094 [main] org.mule.module.launcher.DeploymentDirectoryWatcher:
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
+ Mule is up and kicking (every 5000ms)                              +
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
INFO  2016-10-07 18:49:17,172 [main] org.mule.module.launcher.StartupSummaryDeploymentListener:
**********************************************************************
*            - - + DOMAIN + - -               * - - + STATUS + - - *
**********************************************************************
* default                                     * DEPLOYED             *
**********************************************************************

********************************************************************************************************
*           - - + APPLICATION + - -           *       - - + DOMAIN + - -       * - - + STATUS + - - *
********************************************************************************************************
* endecrypt                                   * default                        * DEPLOYED           *
********************************************************************************************************
```

- As soon as when you keep the file under source directory. Our file will be decoded immediately

Learning:

- From the example we learnt when to use scatter and gather flow control.

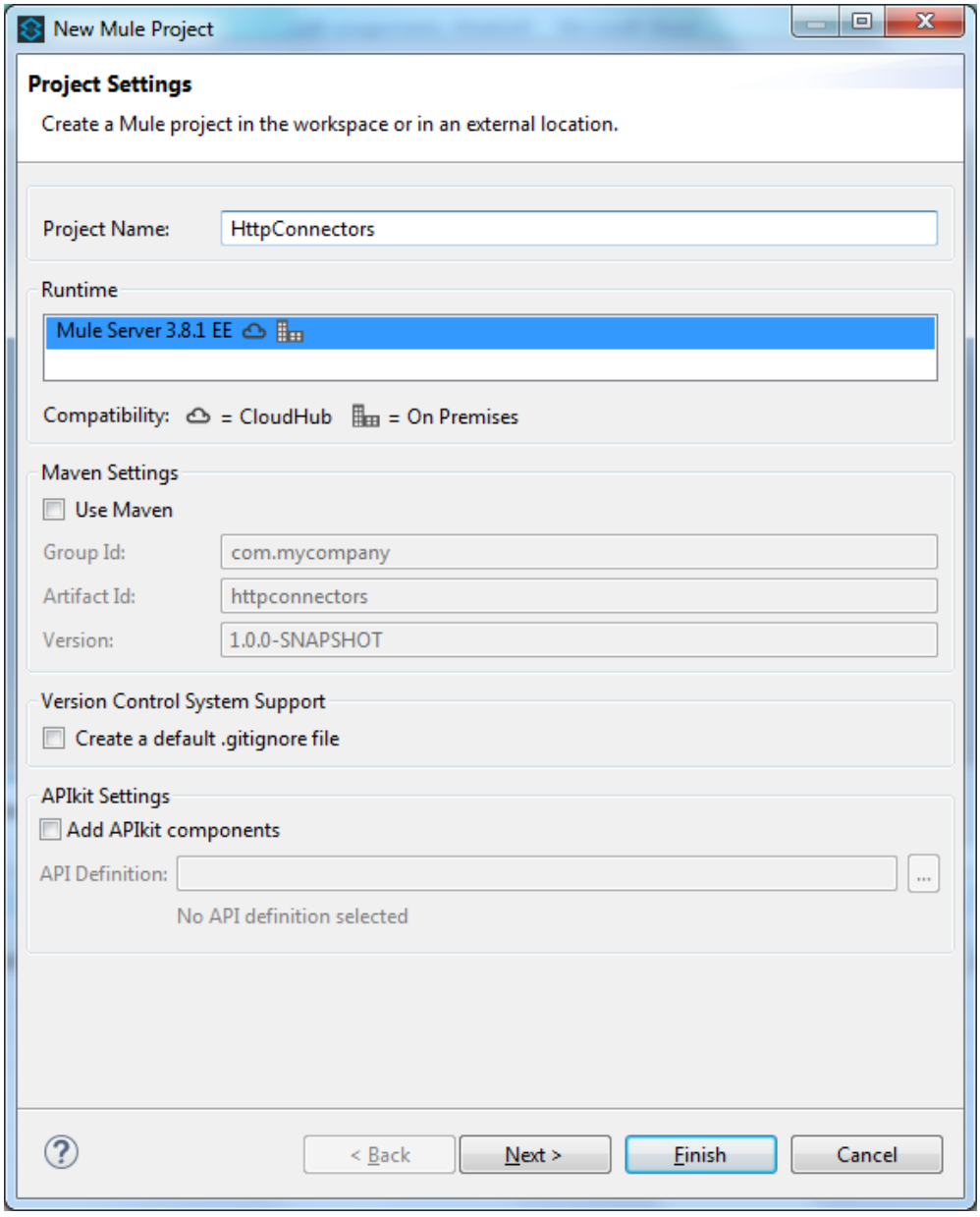
You have successfully completed this lab!

LAB – HTTP Connectors

Testing HTTP Connector service using POSTMAN chrome plugin.

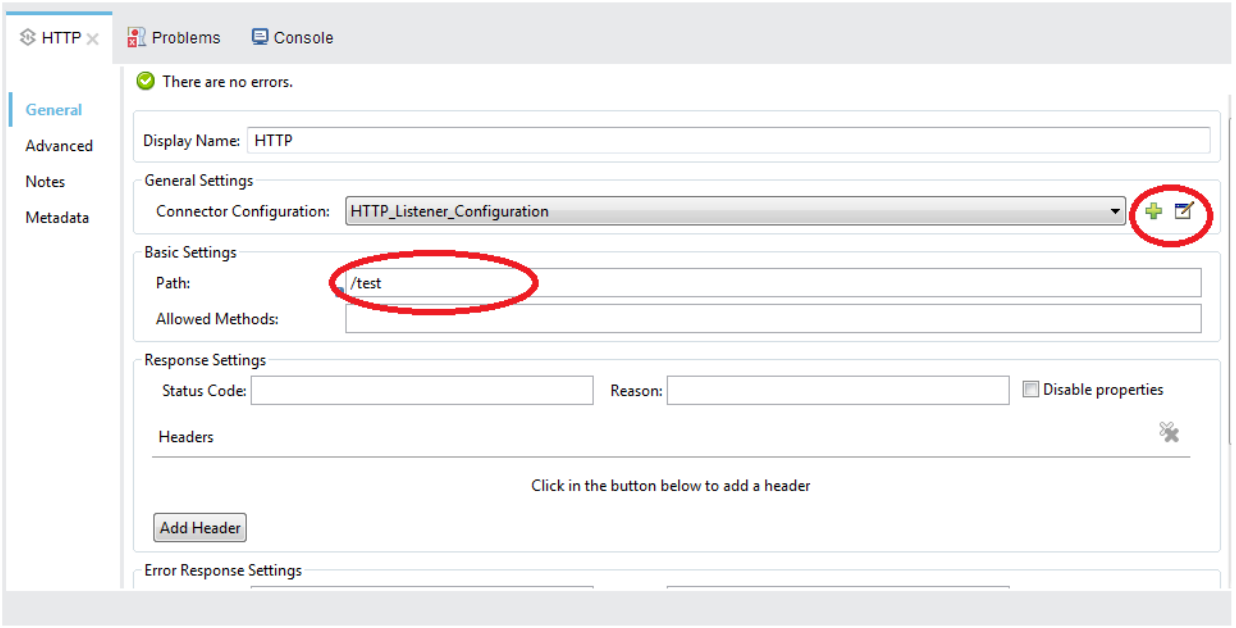
Steps:

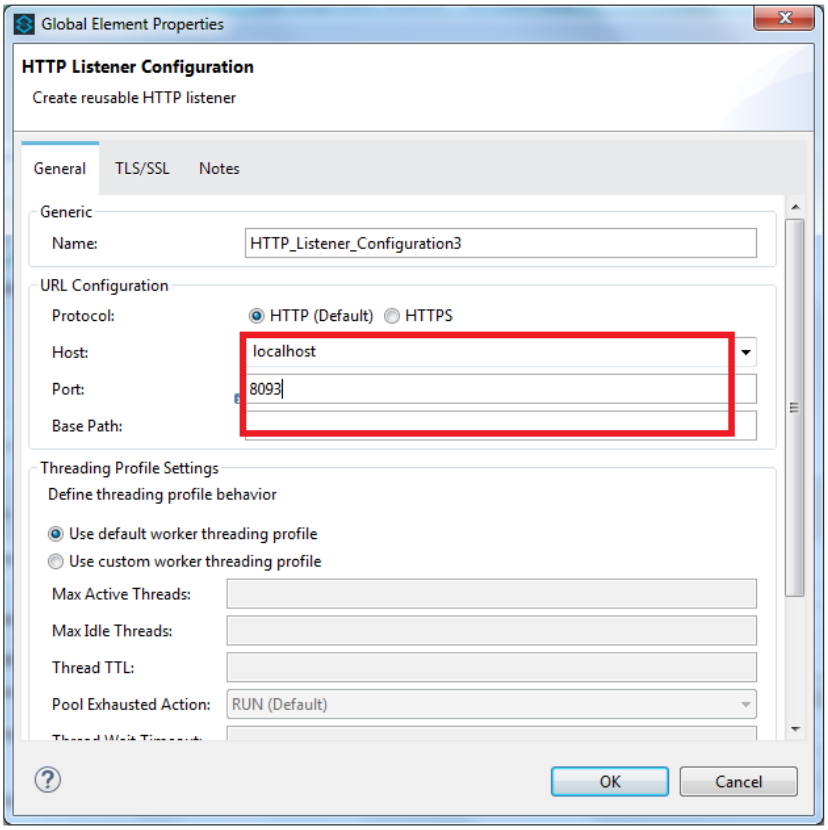
- Open AnyPoint Studio
- File → New → Mule Project
- Give the project name as HttpConnectors as mentioned below. And then finish.



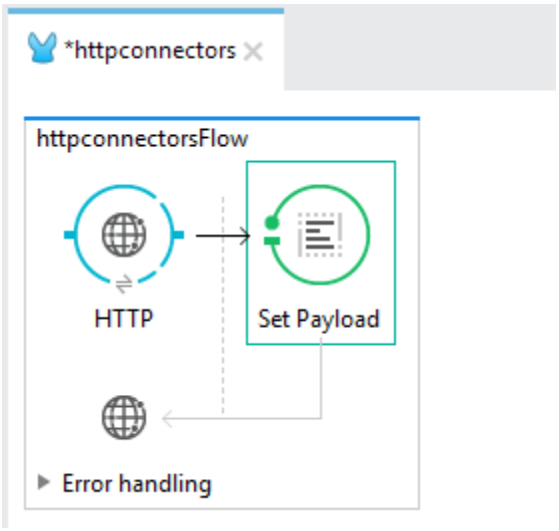
- You will be getting one new mule project.

- Open httpconnectors.xml file, enable message flow tab.
- Choose Http (click and drag to the place) from Mule Palette.
- Click the plus sign

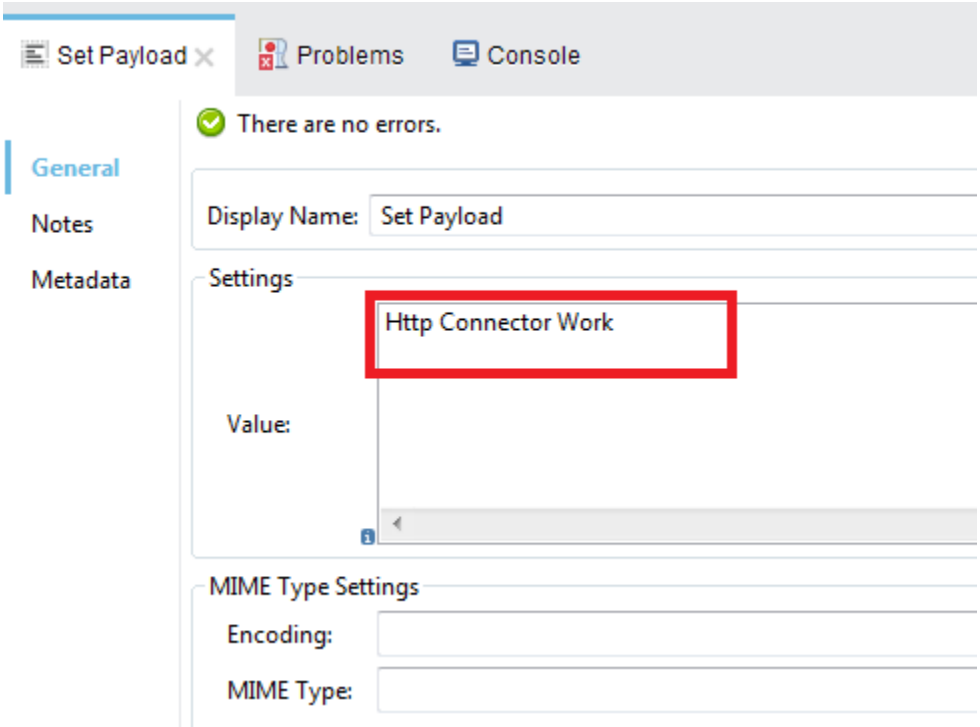




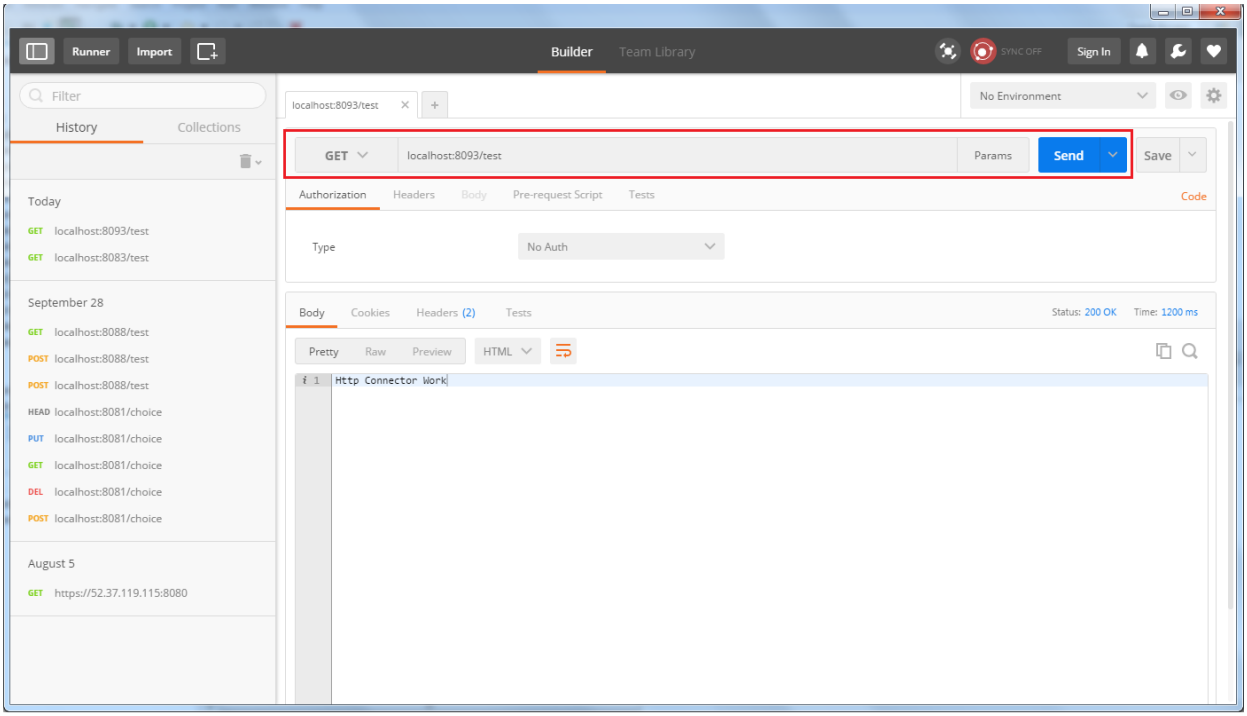
- Enter "/test" as path.
- Add setpayload from Mule palette. Your xml file should look like the below:



- Select Set Payload and mention the value as below:



- Right Click project → Run As→ Mule Application.
- Once Application started. Open postman plug-in to test the application.
- In the postman application enter the url, you will get the message as Set Payload value.



Learning:

- From the above lab we learnt how to use HttpConnector services. Along with this you should know the below options as well.
- Note:
  - Understand the “Allowed Method” Options also. (Used to specify very particular HTTP method such as POST, PUT, GET ,ect…)
  - Understand “Response Settings” options also.

  - “Add Headers” to add additional response along with header.

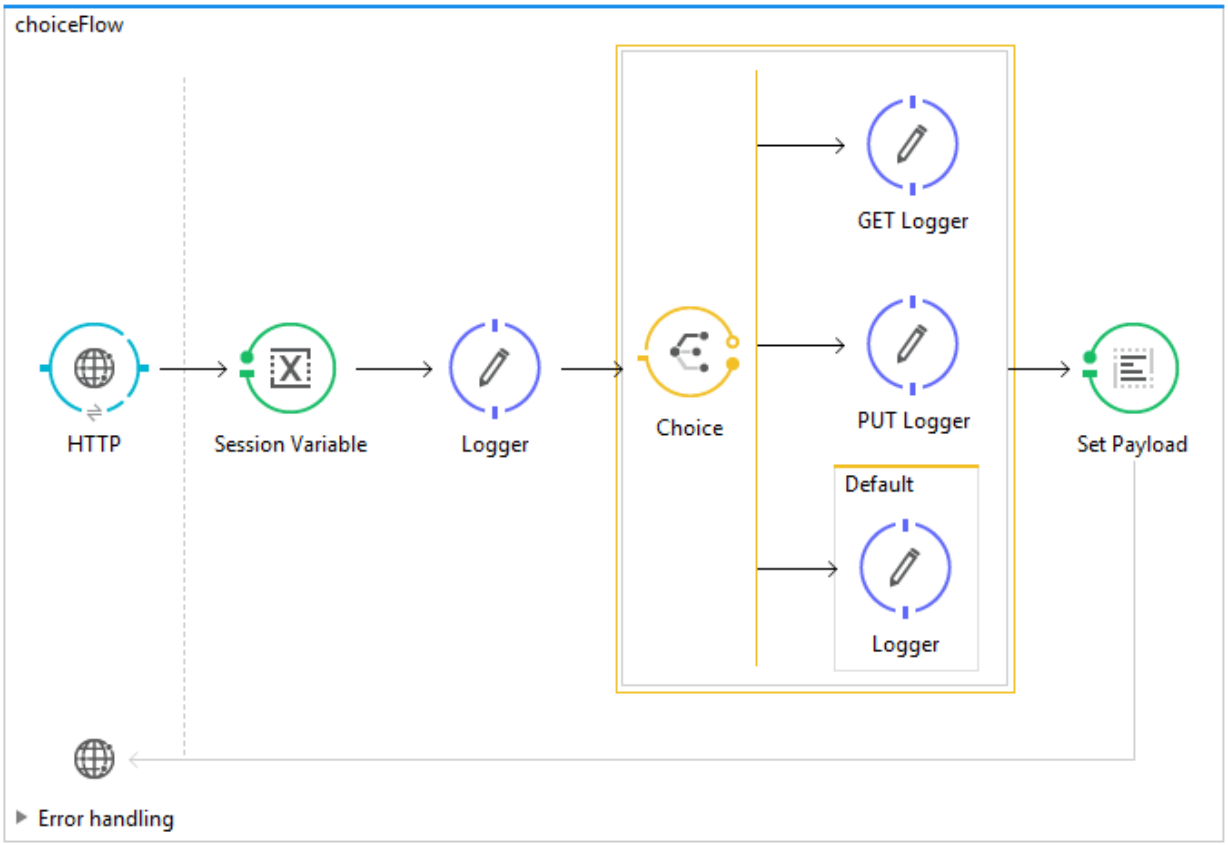
You have successfully completed this lab!

LAB – Choice Router and Flow Reference

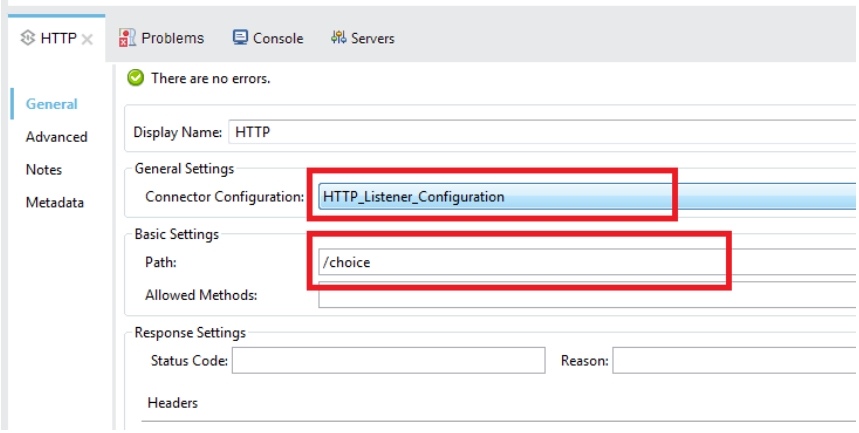
This demo will brief about how to use Choice Router and Flow References.

Steps:

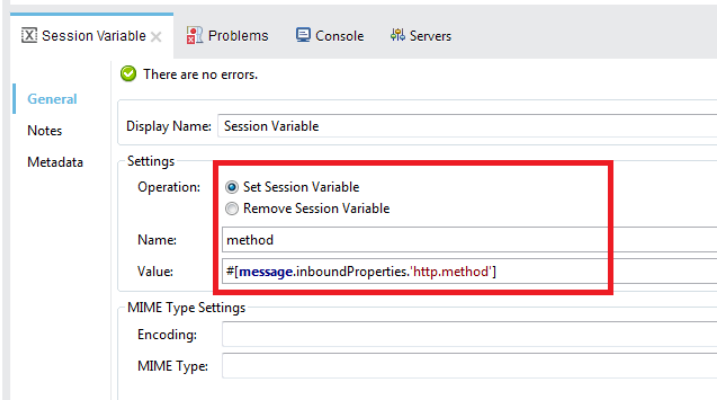
- Open AnyPoint Studio
- File → New → Mule Project
- Give the project name as choice as mentioned below. And then finish.
- Create the below flow with choice.



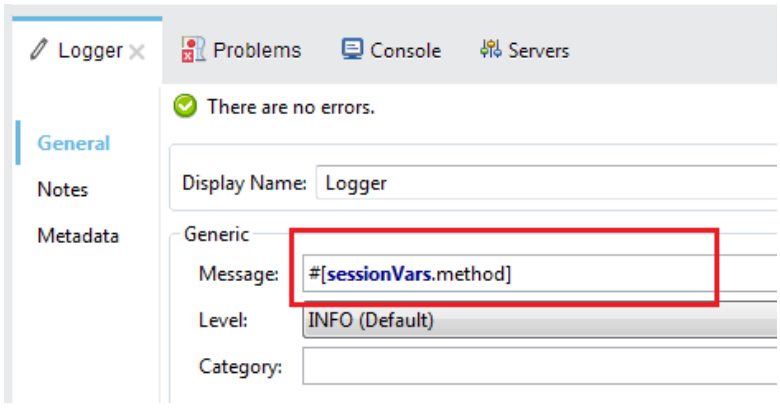
- HTTP Connector property should as mentioned below.
  - In Connector Configuration select **localhost** and change port as **8081**
  - In path enter **"/choice"**



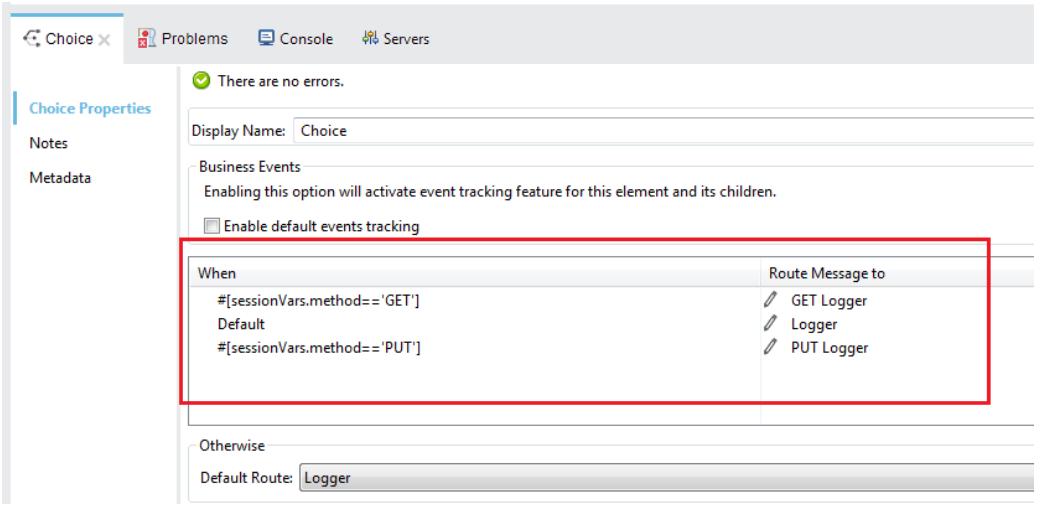
- In the session variable select operation as **Set Session variable**, name as **method** and Value as **#[message.inboundProperties.'http.method']**



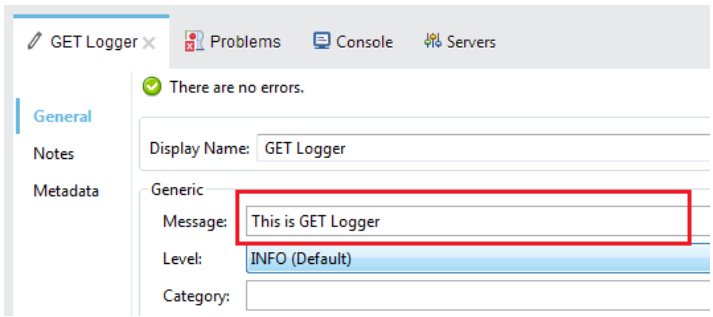
- Set the logger property as highlighted below:



- In the choice property configure the below:

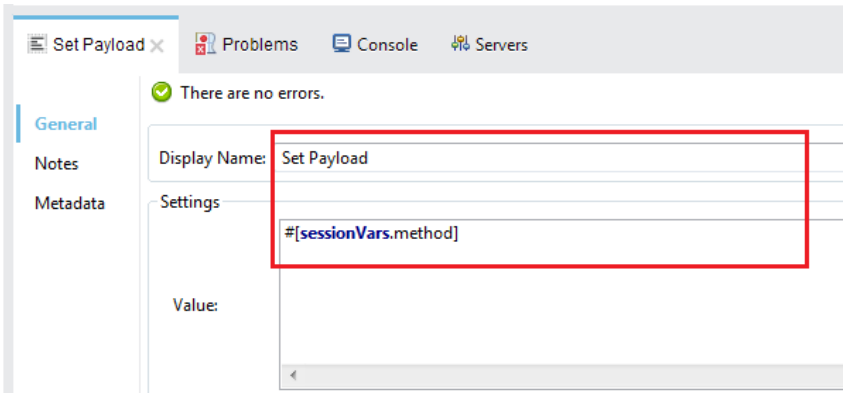


- GET logger, PUT Logger and Default Logger configuration as mentioned below:



Set the message property for GET Logger as "This is GET Logger", PUT Logger as " This is PUT Logger" and Default Logger as "The method something other than GET and PUT"

- Mention Set Payload property as highlighted below:

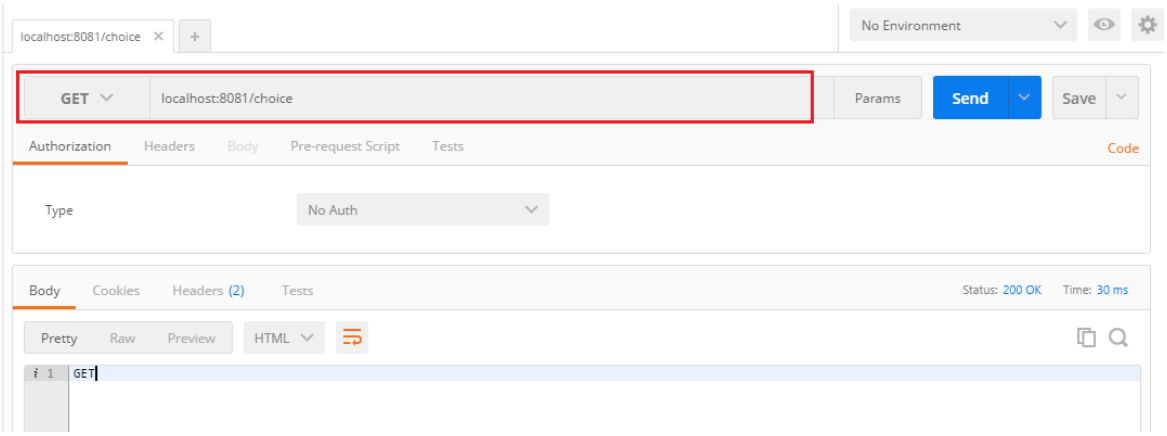


- For the above configuration the Choice.xml file should be as like below:

```
<?xml version="1.0" encoding="UTF-8"?>
<mule xmlns:http="http://www.mulesoft.org/schema/mule/http"
xmlns:tracking="http://www.mulesoft.org/schema/mule/ee/tracking"
```

```
xmlns="http://www.mulesoft.org/schema/mule/core"
xmlns:doc="http://www.mulesoft.org/schema/mule/documentation"
       xmlns:spring="http://www.springframework.org/schema/beans"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xsi:schemaLocation="http://www.springframework.org/schema/beans
http://www.springframework.org/schema/beans/spring-beans-current.xsd
http://www.mulesoft.org/schema/mule/core
http://www.mulesoft.org/schema/mule/core/current/mule.xsd
http://www.mulesoft.org/schema/mule/http
http://www.mulesoft.org/schema/mule/http/current/mule-http.xsd
http://www.mulesoft.org/schema/mule/ee/tracking
http://www.mulesoft.org/schema/mule/ee/tracking/current/mule-tracking-ee.xsd">
    <http:listener-config name="HTTP_Listener_Configuration" host="localhost" port="8081"
doc:name="HTTP Listener Configuration"/>
    <flow name="choiceFlow">
        <http:listener config-ref="HTTP_Listener_Configuration" path="/choice"
doc:name="HTTP"/>
        <set-session-variable variableName="method"
value="#[message.inboundProperties.'http.method']" doc:name="Session Variable"/>
        <logger message="#[sessionVars.method]" level="INFO" doc:name="Logger"/>
        <choice doc:name="Choice">
            <when expression="#[sessionVars.method=='GET']">
                <logger message="This is GET Logger" level="INFO" doc:name="GET Logger"/>
            </when>
            <when expression="#[sessionVars.method=='PUT']">
                <logger message="This is PUT Logger" level="INFO" doc:name="PUT Logger"/>
            </when>
            <otherwise>
                <logger message="The method something other than GET and PUT" level="INFO"
doc:name="Logger"/>
            </otherwise>
        </choice>
        <set-payload value="#[sessionVars.method]" doc:name="Set Payload"/>
    </flow>

</mule>
```

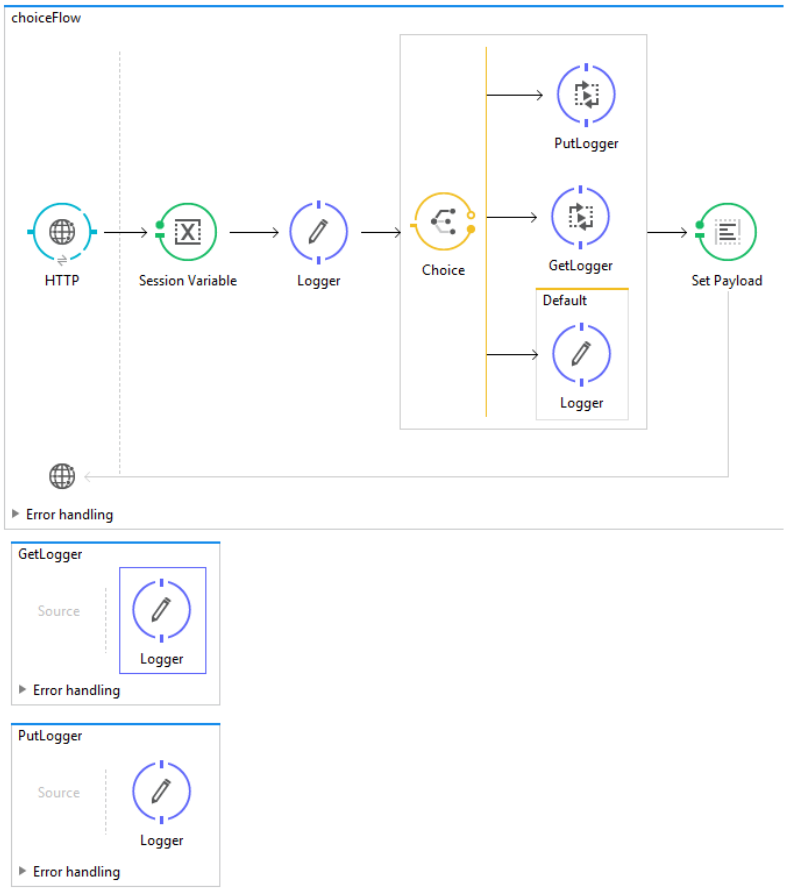
- Right Click project → Run As→ Mule Application.
- Open POSTMAN application enter the URL with GET method



- Now check your console log , you will be getting the below log details
- Change the method as GET, PUT and other. If you absorb the log details, we will be getting the complete details about the choice.

```
INFO  2016-10-10 09:20:05,318 [Mule.app.deployer.monitor.1.thread.1] org.mule.module.management.agent.WrapperManagerAgent: This JVM hasn't
been launched by the wrapper, the agent will not run.
INFO  2016-10-10 09:20:05,341 [Mule.app.deployer.monitor.1.thread.1] org.mule.DefaultMuleContext:
**********************************************************************
* Application: choice                                                *
* OS encoding: \, Mule encoding: UTF-8                               *
*                                                                    *
* Agents Running:                                                    *
*   JMX Agent                                                        *
*   Batch module default engine                                      *
*   DevKit Extension Information                                     *
*   Wrapper Manager                                                  *
**********************************************************************
INFO  2016-10-10 09:20:05,341 [Mule.app.deployer.monitor.1.thread.1] org.mule.module.launcher.MuleDeploymentService:
**********************************************************************
* Started app 'choice'                                               *
**********************************************************************
INFO  2016-10-10 09:20:23,273 [[choice].HTTP_Listener_Configuration.worker.01]
org.mule.api.processor.LoggerMessageProcessor: PATCH
INFO  2016-10-10 09:20:23,281 [[choice].HTTP_Listener_Configuration.worker.01]
org.mule.api.processor.LoggerMessageProcessor: The method something other than GET and PUT
INFO  2016-10-10 09:20:27,311 [[choice].HTTP_Listener_Configuration.worker.01]
org.mule.api.processor.LoggerMessageProcessor: GET
INFO  2016-10-10 09:20:27,312 [[choice].HTTP_Listener_Configuration.worker.01]
org.mule.api.processor.LoggerMessageProcessor: This is GET Logger
```

- Now change the logger into flow reference as mentioned below:



Learning:

- From the above demo we understand how to use Choice component.

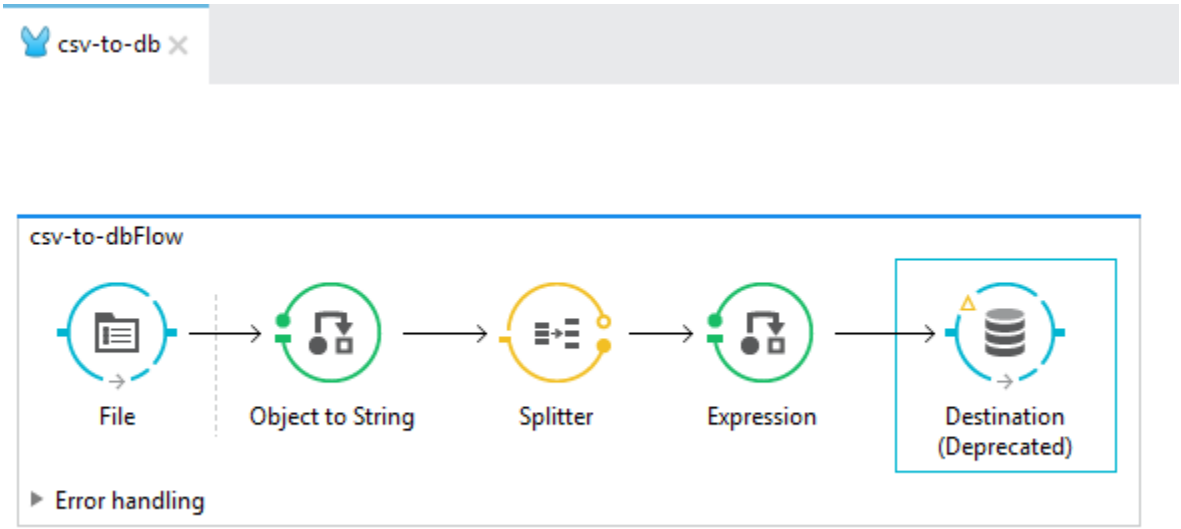
You have successfully completed this lab!

LAB – Import CSV into MYSQL database

Create one flow to import the CSV file into MYSQL database.

Steps:

- Open AnyPoint Studio
- File → New → Mule Project
- Give the project name as csv-to-db and then finish.
- Create the following flow:



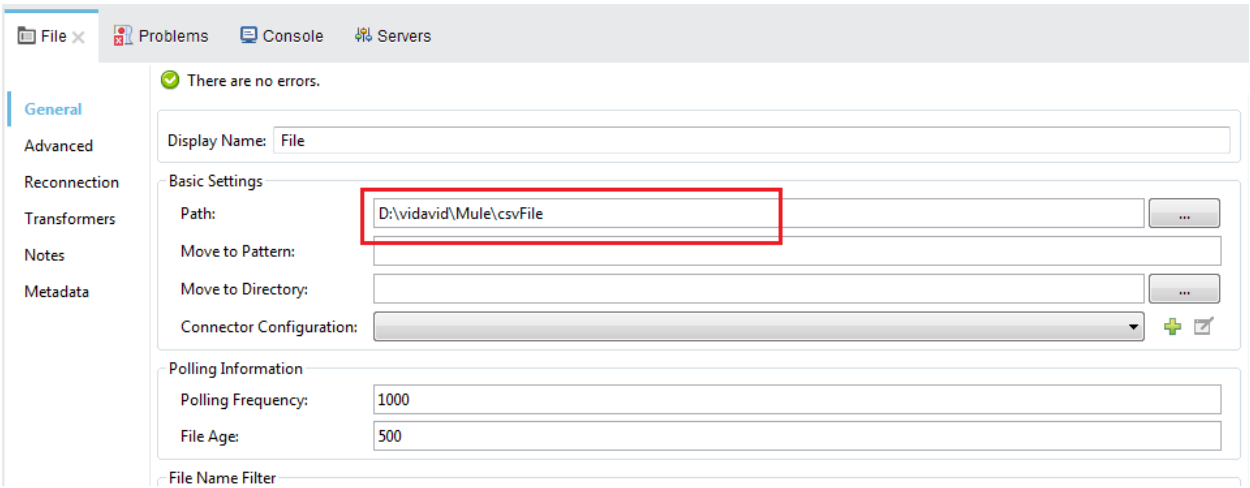
- If you look at the configuration.xml file, it will look as below for the above flow:

```
<?xml version="1.0" encoding="UTF-8"?>

<mule xmlns:jdbc-ee="http://www.mulesoft.org/schema/mule/ee/jdbc"
xmlns:jdbc="http://www.mulesoft.org/schema/mule/jdbc"
      xmlns:file="http://www.mulesoft.org/schema/mule/file"
xmlns="http://www.mulesoft.org/schema/mule/core"
xmlns:doc="http://www.mulesoft.org/schema/mule/documentation"
      xmlns:spring="http://www.springframework.org/schema/beans"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xsi:schemaLocation="http://www.mulesoft.org/schema/mule/ee/jdbc
http://www.mulesoft.org/schema/mule/ee/jdbc/current/mule-jdbc-ee.xsd
http://www.springframework.org/schema/beans
http://www.springframework.org/schema/beans/spring-beans-current.xsd
http://www.mulesoft.org/schema/mule/core
http://www.mulesoft.org/schema/mule/core/current/mule.xsd
```

```
http://www.mulesoft.org/schema/mule/file
http://www.mulesoft.org/schema/mule/file/current/mule-file.xsd">
    <configuration doc:name="Configuration">
        <expression-language autoResolveVariables="true">
            <import class="org.mule.util.StringUtils"></import>
        </expression-language>
    </configuration>
     <jdbc-ee:mysql-data-source name="MySQL_Daata_Base" user="root"
password="admin" url="jdbc:mysql://localhost:3306/demodb"
transactionIsolation="UNSPECIFIED" doc:name="MySQL Data Source"/>

    <jdbc-ee:connector name="MS_SQL_DB_Connector" dataSource-
ref="MySQL_Daata_Base" validateConnections="true" queryTimeout="-1"
pollingFrequency="0" doc:name="Database">
        <jdbc-ee:query key="insertRecord" value="INSERT INTO
userdetails VALUES
(#[message.payload[0]],#[message.payload[1]],#[message.payload[2]]);"/
>
    </jdbc-ee:connector>
    <flow name="csv-to-dbFlow">
        <file:inbound-endpoint responseTimeout="10000" doc:name="File"
path="D:\vidavid\Mule\csvFile"></file:inbound-endpoint>

        <!-- Step 2: Convert between object arrays and strings -->
        <object-to-string-transformer doc:name="Object to
String"></object-to-string-transformer>

        <!-- Step 3: Split each row -->
        <splitter expression="#[StringUtils.split(message.payload,
'\n\r')]" doc:name="Splitter"></splitter>

        <!-- Step 4: Transform CSV row in array -->
        <expression-transformer
expression="#[StringUtils.split(message.payload, ',')]"
doc:name="Expression"></expression-transformer>

        <!-- Step 5: Dump into the destination Database -->
        <jdbc-ee:outbound-endpoint exchange-pattern="one-way"
queryTimeout="-1" doc:name="Destination" connector-
ref="MS_SQL_DB_Connector" queryKey="insertRecord"></jdbc-ee:outbound-
endpoint>
    </flow>
</mule>
```

File:

- In the file Mule properties mention the path locating the source folder which carries .csv file to upload into database.

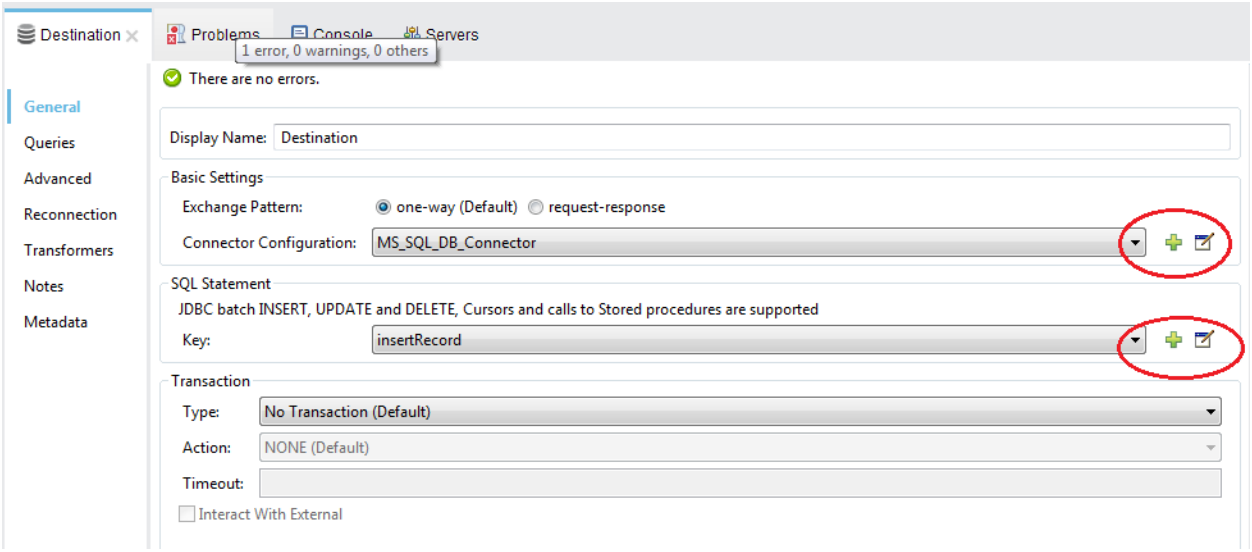


Database:

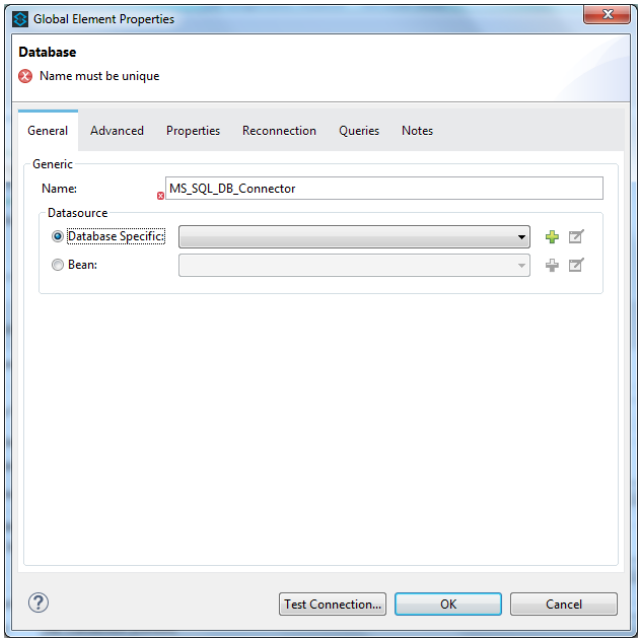
- Goto mysql database and create the table as below:

```
mysql> use demodb;
Database changed
mysql> create table userdetails(userid int primary key,firstName varchar(15),las
tName varchar(15));
Query OK, 0 rows affected (1.36 sec)
```

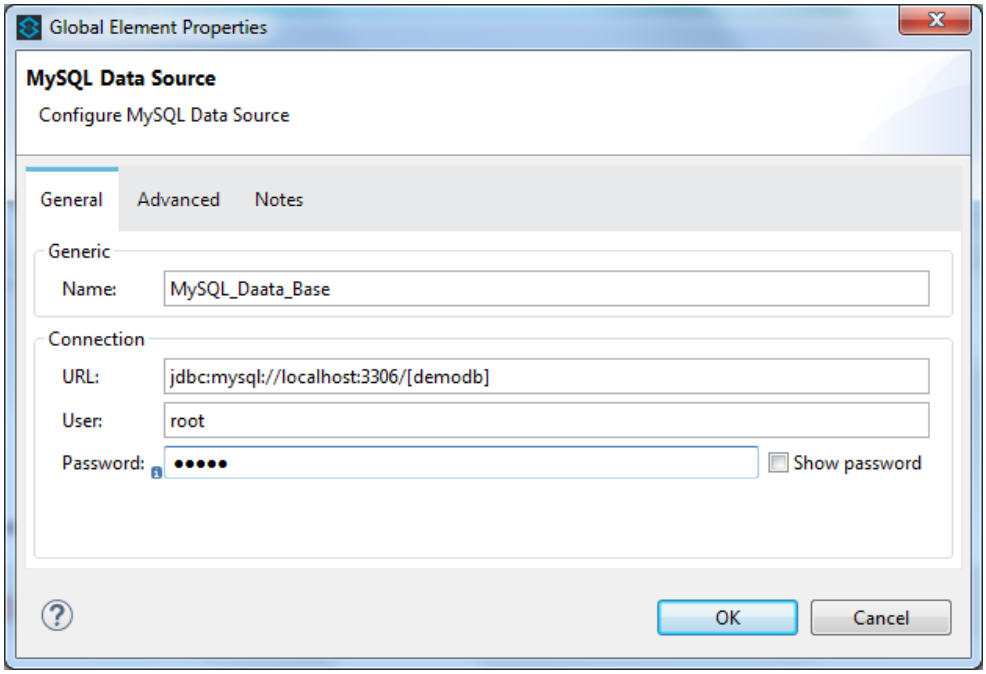
- Add "mysql-connector-java-5.1.26-bin.jar" by using Right Click Project → build Path → AddExternal Archives..
- In the database connector add the below configuration:



- Click plus sign in Connector Configuration

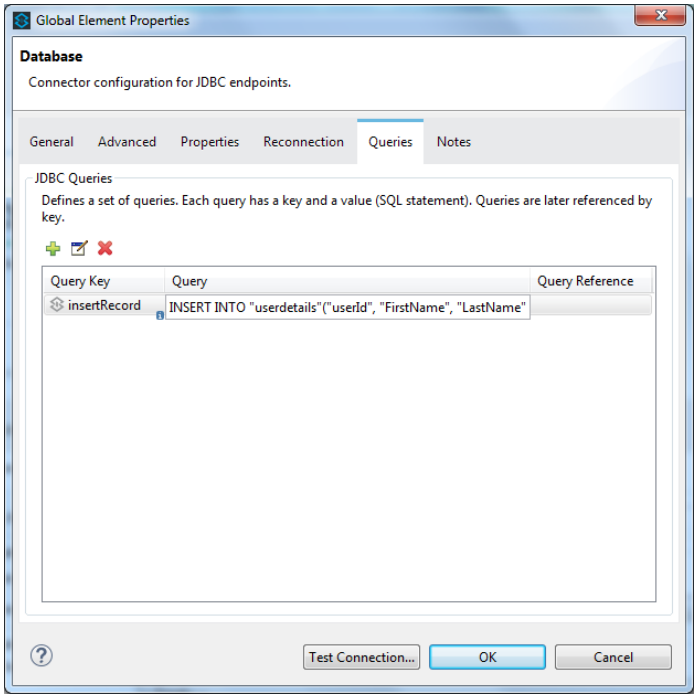


- Click Database Specific, and add MYSQL database



- Under Queries Tab mention the below query with key call “insertRecord”

  INSERT INTO userdetails("userId", "FirstName", "LastName") VALUES (#[**message.payload**[0]],#[**message.payload**[1]],#[**message.payload**[2]])



- Right Click project → Run As→ Mule Application.

Learning:

- From the above case study, we learnt how to import csv file into database. We also understand how to use splitter and expression-transformer.

```
<!-- Step 2: Convert between object arrays and strings -->
        <object-to-string-transformer        doc:name="Object        to
String"></object-to-string-transformer>

<!-- Step 3: Split each row -->
        <splitter      expression="#[StringUtils.split(message.payload,
'\n\r')]" doc:name="Splitter"></splitter>

<!-- Step 4: Transform CSV row in array -->
        <expression-transformer
expression="#[StringUtils.split(message.payload,                ',')]"
doc:name="Expression"></expression-transformer>
```

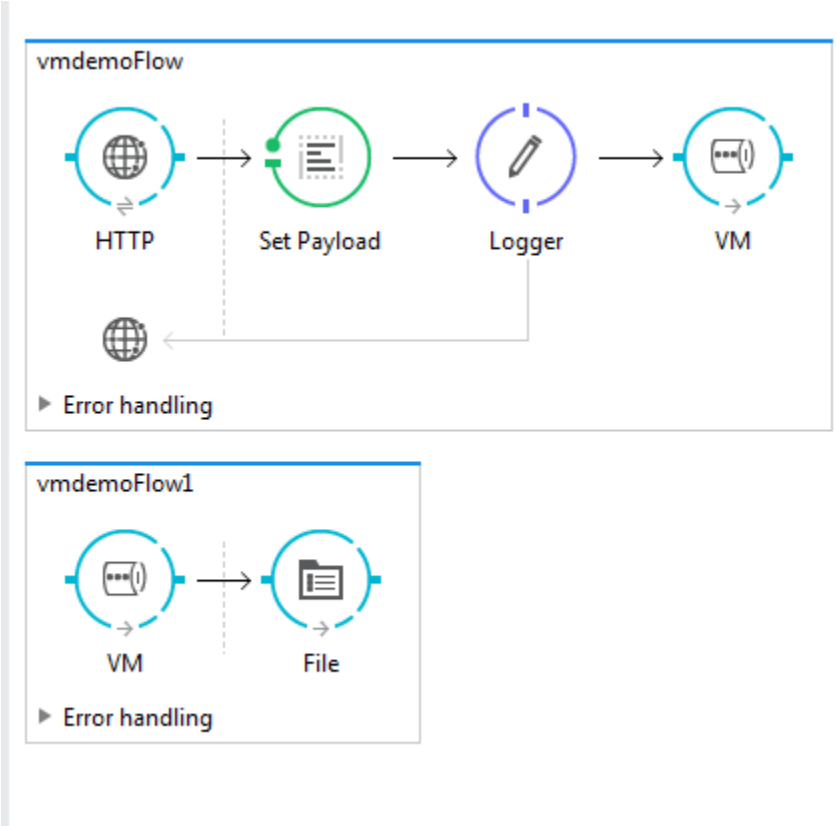
You have successfully completed this lab!

LAB VM

An overview about VM and demo on how it works.

Steps:

- Open AnyPoint Studio
- File → New → Mule Project
- Give the project name as vmdemo and then finish.
- Create the following flow:



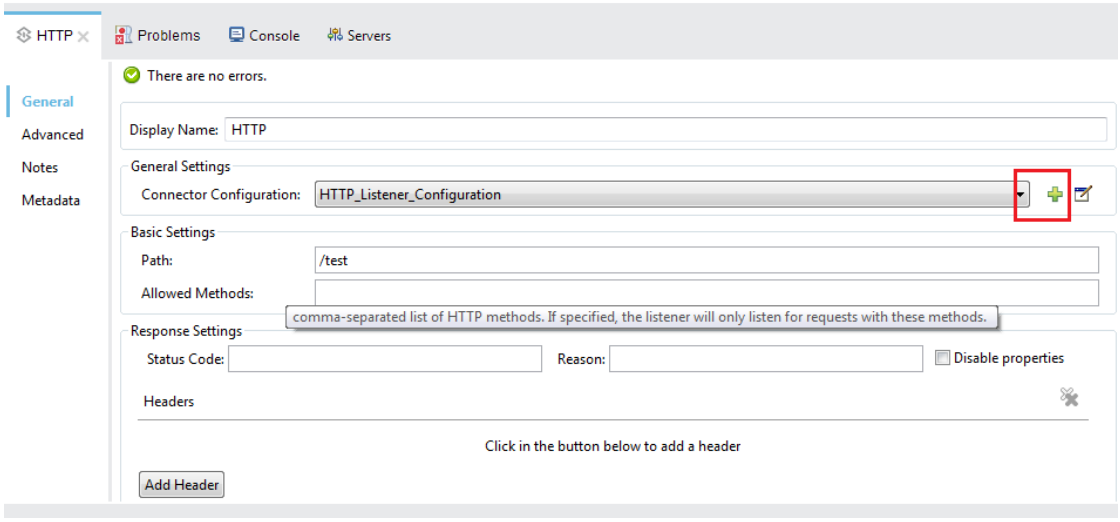
- For the above flow below is the configuration details

configuration.xml

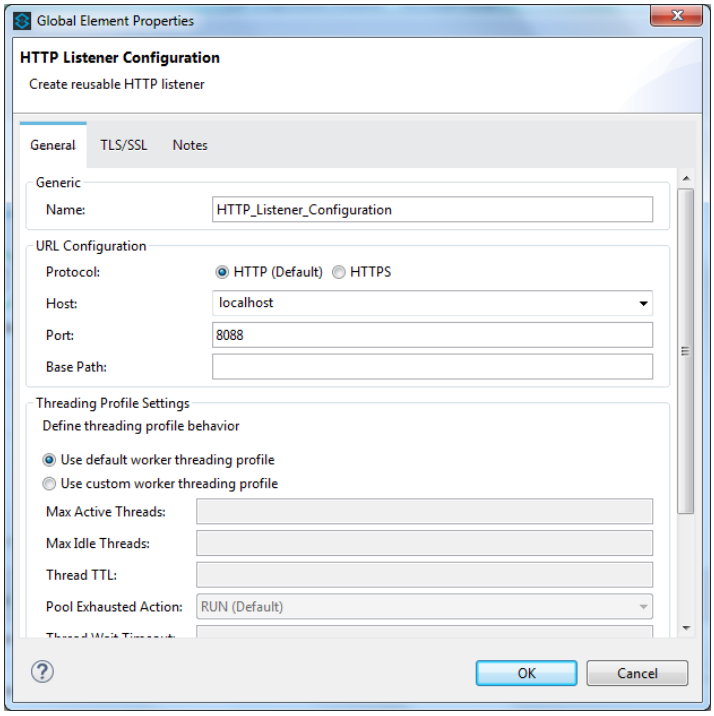
```
<?xml version="1.0" encoding="UTF-8"?>

<mule xmlns:file="http://www.mulesoft.org/schema/mule/file"
xmlns:vm="http://www.mulesoft.org/schema/mule/vm"
xmlns:http="http://www.mulesoft.org/schema/mule/http"
xmlns="http://www.mulesoft.org/schema/mule/core"
xmlns:doc="http://www.mulesoft.org/schema/mule/documentation"
       xmlns:spring="http://www.springframework.org/schema/beans"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
```

```
       xsi:schemaLocation="http://www.springframework.org/schema/beans
http://www.springframework.org/schema/beans/spring-beans-current.xsd
http://www.mulesoft.org/schema/mule/core
http://www.mulesoft.org/schema/mule/core/current/mule.xsd
http://www.mulesoft.org/schema/mule/http
http://www.mulesoft.org/schema/mule/http/current/mule-http.xsd
http://www.mulesoft.org/schema/mule/file
http://www.mulesoft.org/schema/mule/file/current/mule-file.xsd
http://www.mulesoft.org/schema/mule/vm http://www.mulesoft.org/schema/mule/vm/current/mule-
vm.xsd">
    <http:listener-config name="HTTP_Listener_Configuration" host="localhost" port="8088"
doc:name="HTTP Listener Configuration"/>
    <flow name="vmdemoFlow">
        <http:listener config-ref="HTTP_Listener_Configuration" path="/test" doc:name="HTTP"/>
        <set-payload value="The message in VM queue" doc:name="Set Payload"/>
        <logger message="#[payload]" level="INFO" doc:name="Logger"/>
        <vm:outbound-endpoint exchange-pattern="one-way" path="vmq" doc:name="VM"/>
    </flow>
    <flow name="vmdemoFlow1">
        <vm:inbound-endpoint exchange-pattern="one-way" path="vmq" doc:name="VM"/>
        <file:outbound-endpoint path="D:\vidavid\Mule\demo\vm-demo" outputPattern="output.txt"
responseTimeout="10000" doc:name="File"/>
    </flow>
</mule>
```

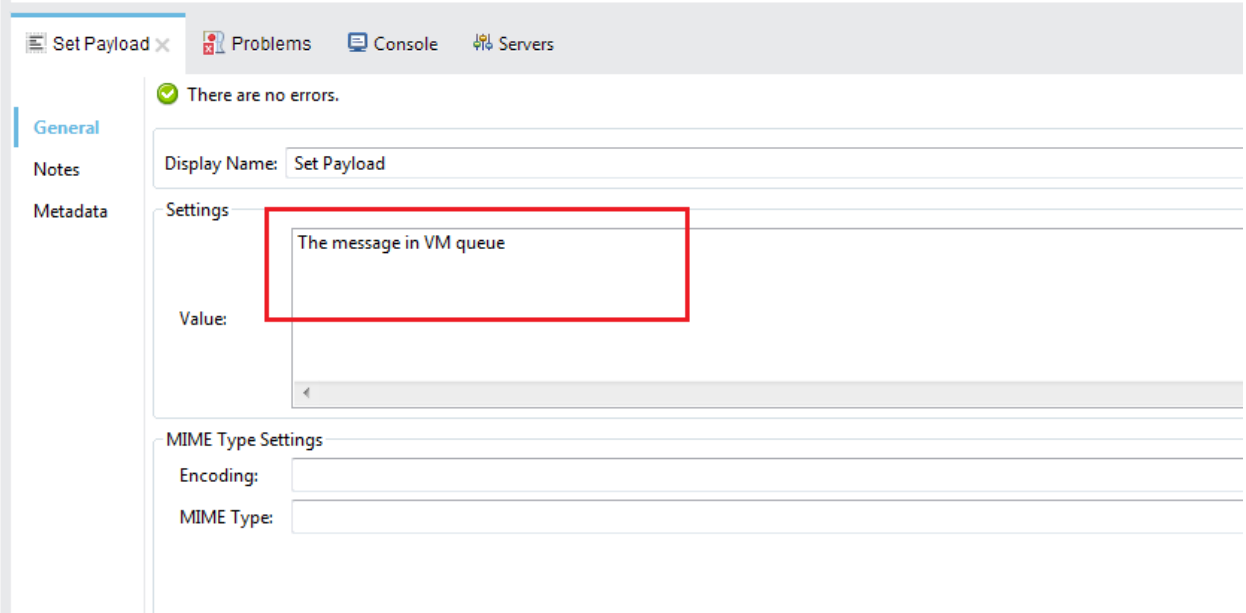
- HTTP connector properties , under Connector configuration click plus sign which is highlighted below.



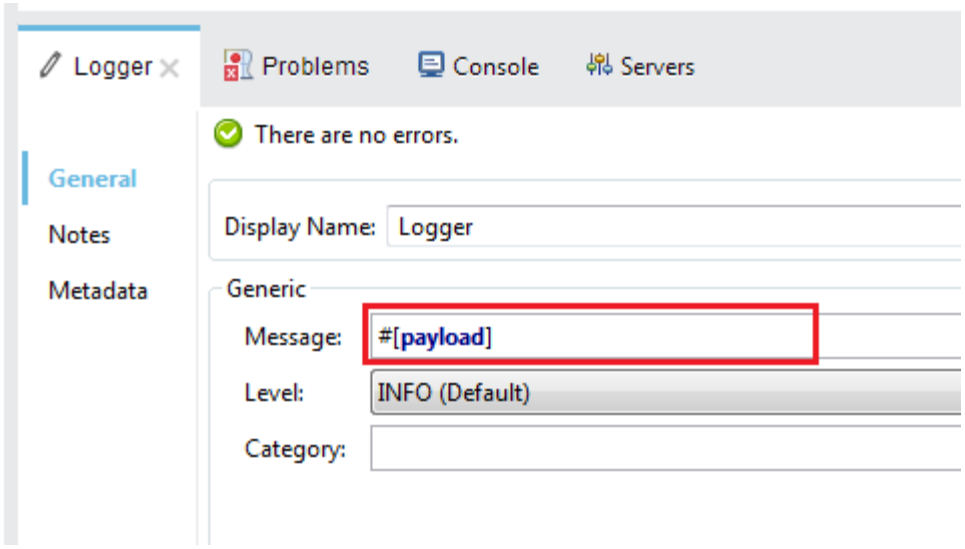
- It will open HTTP Listener Configuration. In this window mention host as localhost and port as 8088.



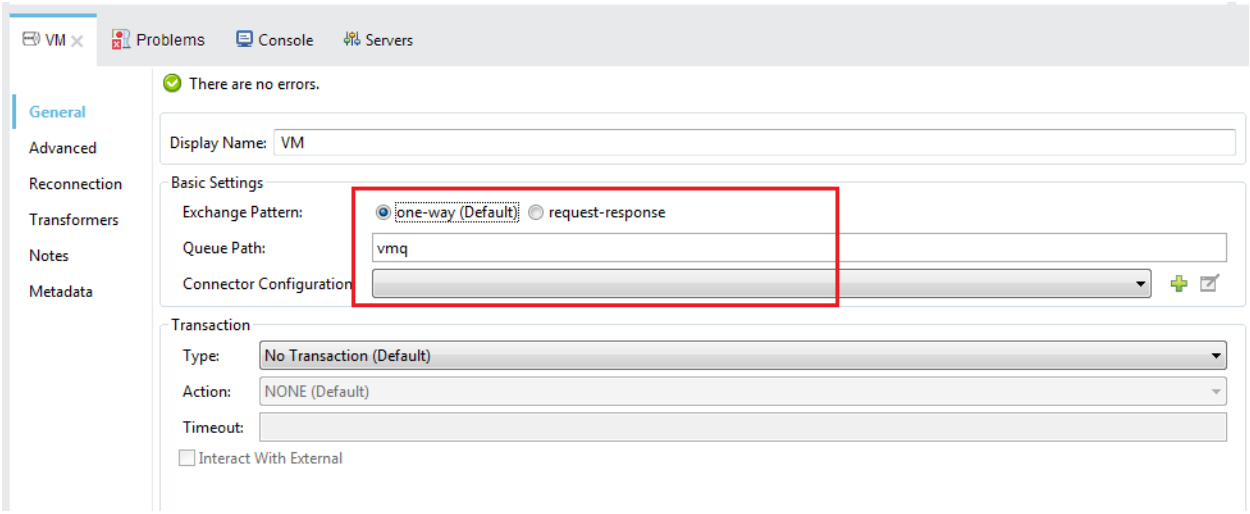
- Set payload value as below:



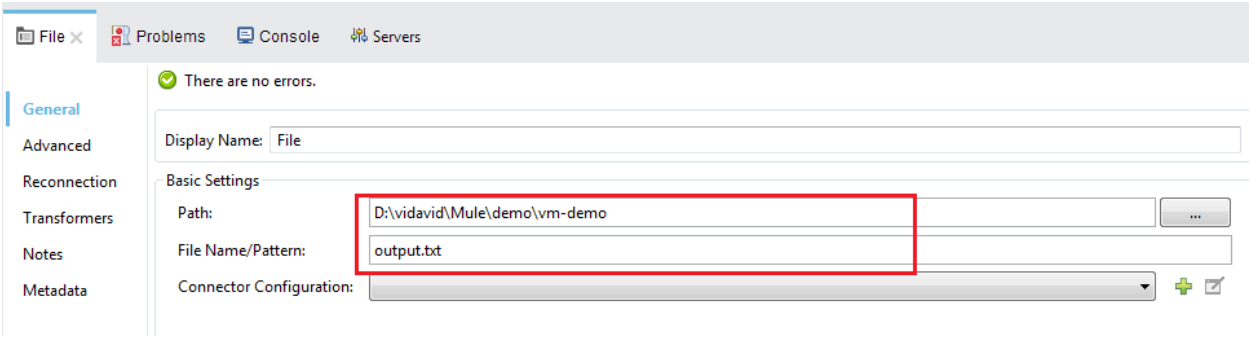
- Logger messages need to be configured.



- In VM properties select exchange pattern as **One-way (Default)** and specify Queue path name as **vmq.**



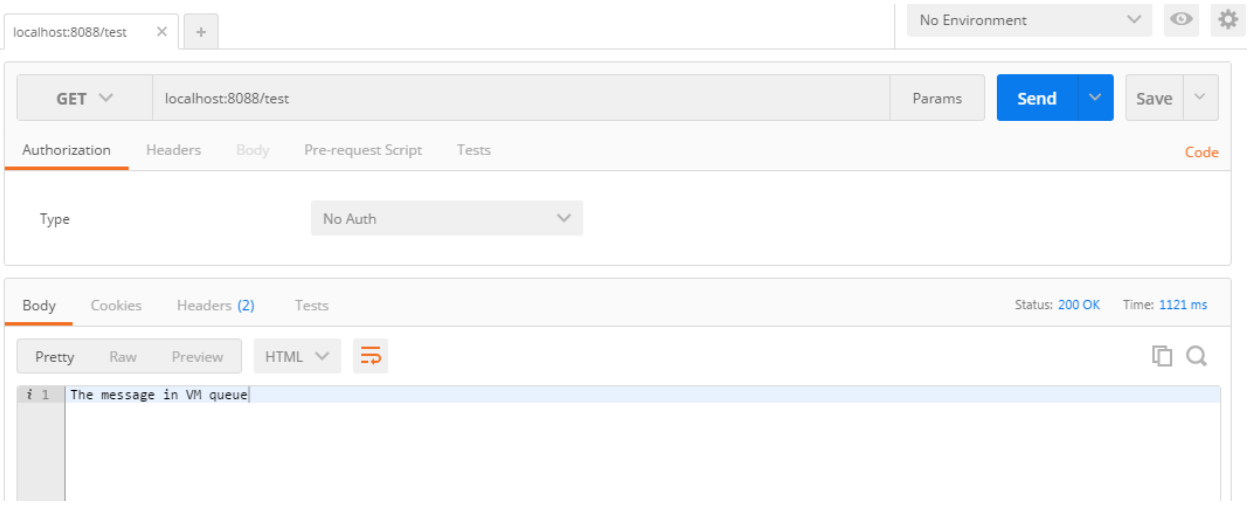
- VM which is in "**Vmdemoflow1"**, Queue path should be same "**vmdemoflow"**. In our case it is **vmq.**
- In the file property specify path and filename as below:



- Right Click project → Run As→ Mule Application.
- Once the application started you will be getting the below log messages in the console log:

```
INFO  2016-10-06 14:20:30,909 [main] org.mule.DefaultMuleContext:
**********************************************************************
* Application: vmdemo                                                *
* OS encoding: \, Mule encoding: UTF-8                               *
*                                                                    *
* Agents Running:                                                    *
*   JMX Agent                                                        *
*   Batch module default engine                                      *
*   DevKit Extension Information                                     *
*   Wrapper Manager                                                  *
**********************************************************************
INFO  2016-10-06 14:20:30,910 [main] org.mule.module.launcher.MuleDeploymentService:
**********************************************************************
* Started app 'vmdemo'                                               *
**********************************************************************
INFO  2016-10-06 14:20:30,922 [main] org.mule.module.launcher.DeploymentDirectoryWatcher:
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
+ Mule is up and kicking (every 5000ms)                             +
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
INFO  2016-10-06 14:20:30,977 [main] org.mule.module.launcher.StartupSummaryDeploymentListener:
**********************************************************************
*              - - + DOMAIN + - -               * - - + STATUS + - - *
**********************************************************************
* default                                       * DEPLOYED           *
**********************************************************************

************************************************************************************************************
*            - - + APPLICATION + - -            *       - - + DOMAIN + - -       * - - + STATUS + - - *
************************************************************************************************************
* vmdemo                                        * default                        * DEPLOYED           *
************************************************************************************************************
```

- Now open postman application and enter the URL “localhost:8088/test”, you will be getting the message from payload.



Learning:

- From the above exercise we learnt how to use VM and when to use VM. Now you can change exchange pattern as “request-response” and see the difference.

You have successfully completed this lab!

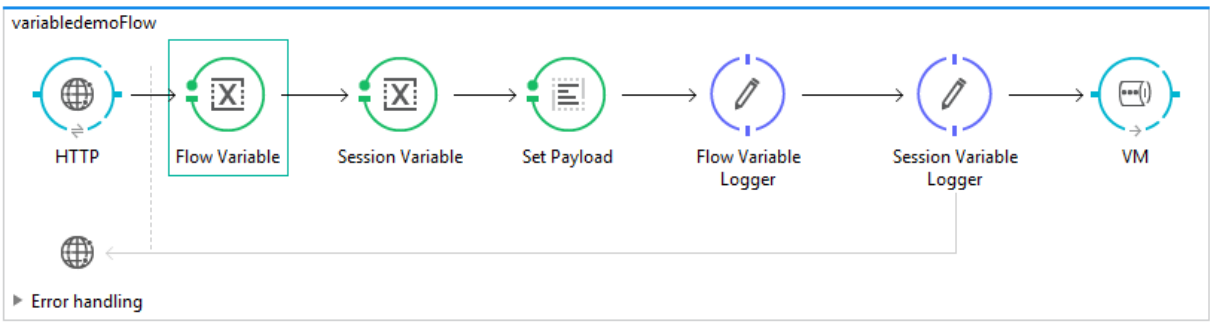
LAB – Variables in Mule

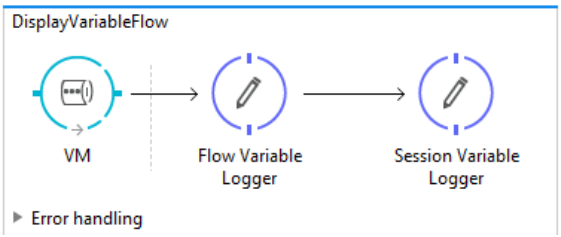
An overview about different types of variables in Mule.

- *Flow Variable*
- *Session Variable*

Steps:

- Open AnyPoint Studio
- File → New → Mule Project
- Give the project name as variabledemo and then finish.
- Create the following flow:



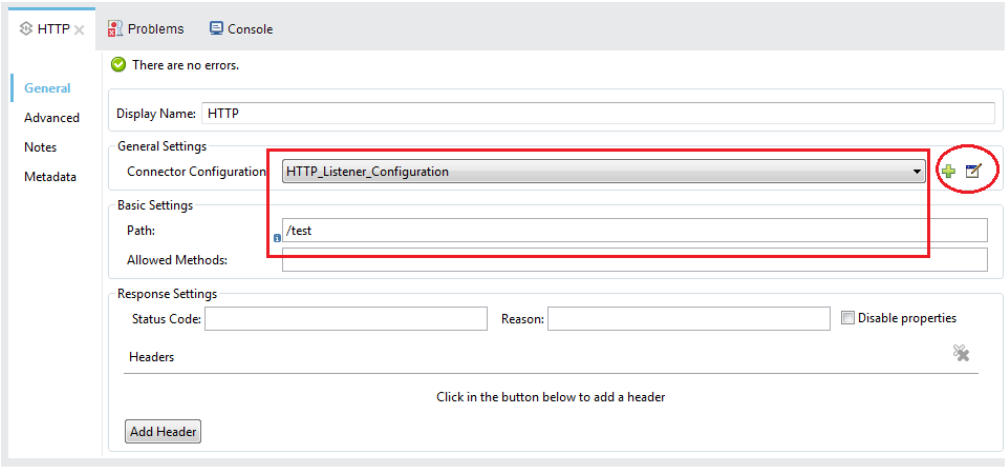


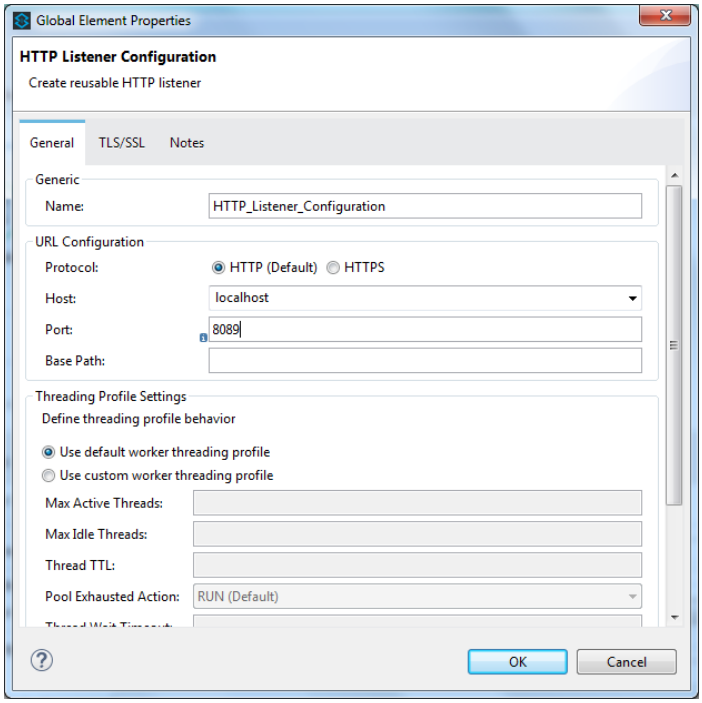
- For the above flow configuration XML file is below:
  configuration XML

```
<?xml version="1.0" encoding="UTF-8"?>

<mule xmlns:vm="http://www.mulesoft.org/schema/mule/vm"
xmlns:http="http://www.mulesoft.org/schema/mule/http"
xmlns:tracking="http://www.mulesoft.org/schema/mule/ee/tracking"
xmlns="http://www.mulesoft.org/schema/mule/core"
xmlns:doc="http://www.mulesoft.org/schema/mule/documentation"
       xmlns:spring="http://www.springframework.org/schema/beans"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xsi:schemaLocation="http://www.springframework.org/schema/beans
http://www.springframework.org/schema/beans/spring-beans-current.xsd
http://www.mulesoft.org/schema/mule/core
http://www.mulesoft.org/schema/mule/core/current/mule.xsd
http://www.mulesoft.org/schema/mule/http
http://www.mulesoft.org/schema/mule/http/current/mule-http.xsd
```

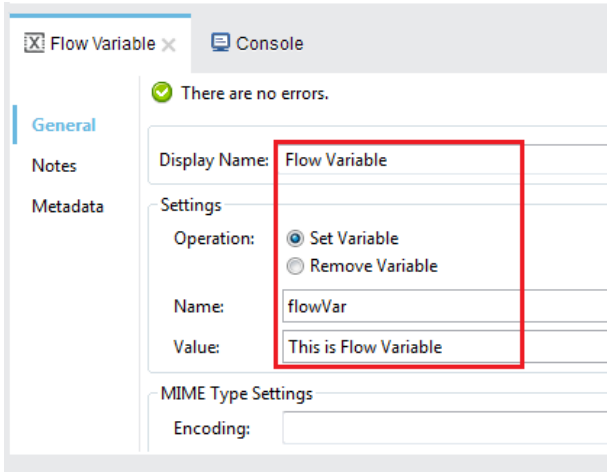
```
http://www.mulesoft.org/schema/mule/vm http://www.mulesoft.org/schema/mule/vm/current/mule-vm.xsd
http://www.mulesoft.org/schema/mule/ee/tracking
http://www.mulesoft.org/schema/mule/ee/tracking/current/mule-tracking-ee.xsd">
    <http:listener-config name="HTTP_Listener_Configuration" host="localhost" port="8089" doc:name="HTTP Listener Configuration"/>
    <flow name="variabledemoFlow">
        <http:listener config-ref="HTTP_Listener_Configuration" path="/test" doc:name="HTTP"/>
        <set-variable variableName="flowVar" value="This is Flow Variable " doc:name="Flow Variable"/>
        <set-session-variable variableName="sessionVar" value="This is Session Variable." doc:name="Session Variable"/>
        <set-payload value="Success Message." doc:name="Set Payload"/>
        <logger message="#[flowVars.flowVar]" level="INFO" doc:name="Flow Variable Logger"/>
        <logger message="#[sessionVars.sessionVar]" level="INFO" doc:name="Session Variable Logger"/>
        <vm:outbound-endpoint exchange-pattern="one-way" path="vmqvar" doc:name="VM"/>
    </flow>
    <flow name="DisplayVariableFlow">
        <vm:inbound-endpoint exchange-pattern="one-way" path="vmqvar" doc:name="VM"/>
        <logger message="#[flowVars.flowVar]" level="INFO" doc:name="Flow Variable Logger"/>
        <logger message="#[sessionVars.sessionVar]" level="INFO" doc:name="Session Variable Logger"/>
    </flow>
</mule>
```

- Configure HttpConnector as mentioned below, choose localhost with port 8089.

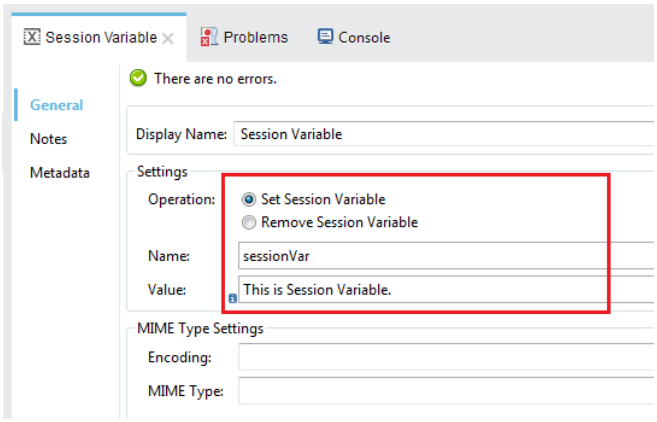




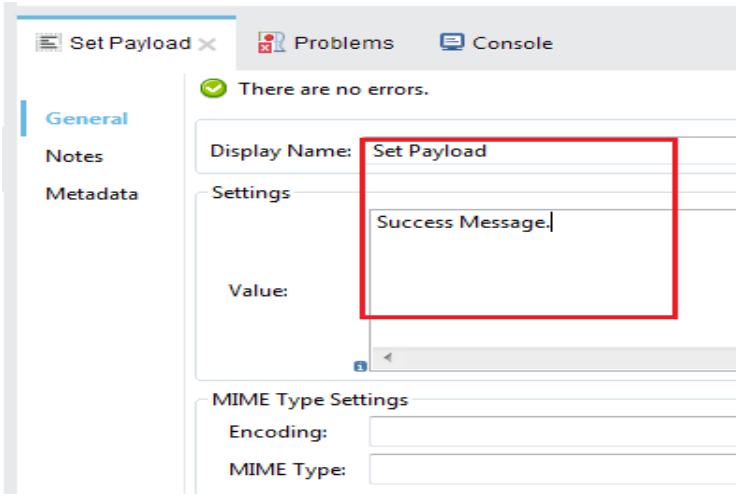
- **Flow variable property should be changed as mentioned below:**



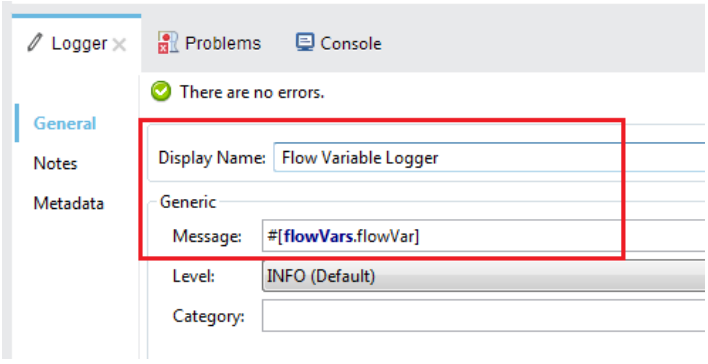
- **Session variable property should be changed as mentioned below:**



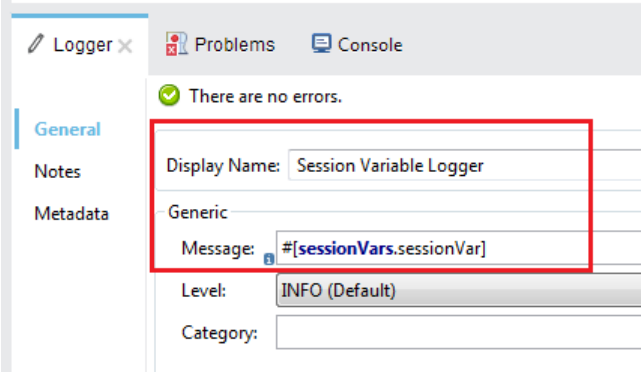
- Set the payload property as mentioned below:



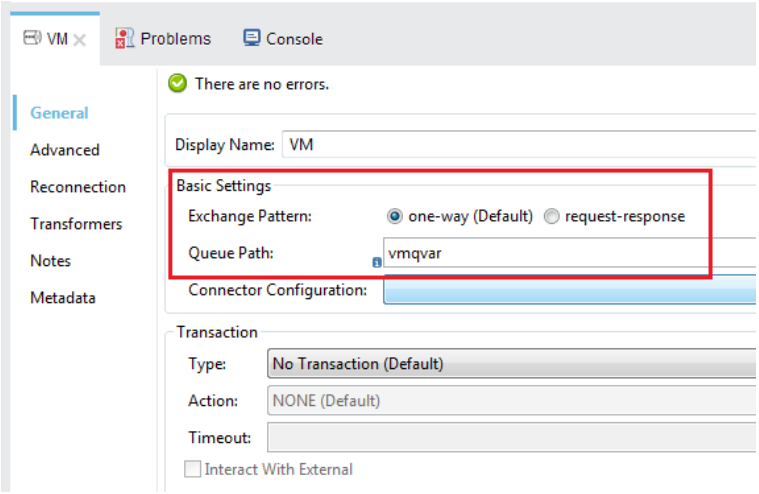
- Bring the flow variable in flow variable logger:



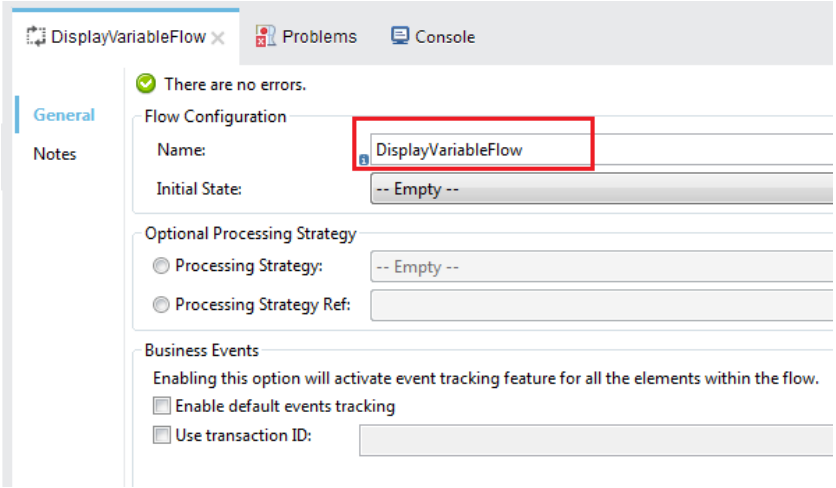
- **Bring the session variable in Session variable logger.**



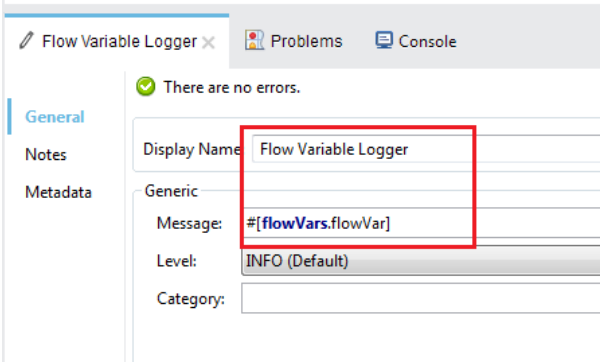
- **Configure VM message queue to continue the flow in the subsequent flow.**

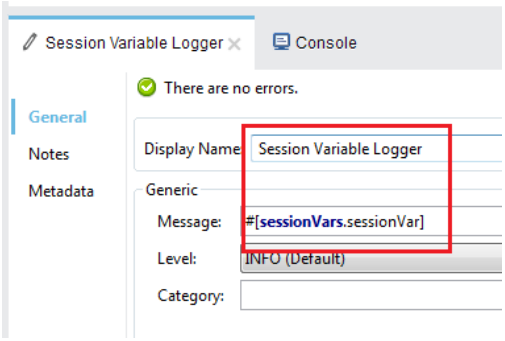


- Create one new flow with **Display Variable Flow** name:

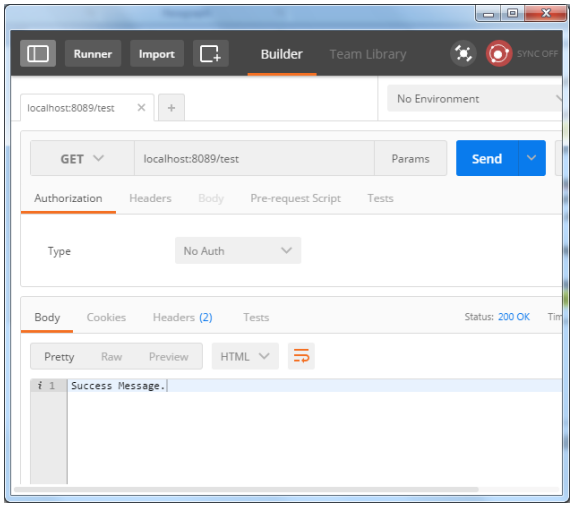


- Bring flow variable and session variable in the relevant logger created in the Display variable flow, as mentioned below:





- Now execute the Mule application.
- Right Click project → Run As→ Mule Application.
- Once the application started you will be getting the below log messages in the console log:
- Open your postman application and enter the URL as
  - Localhost:8089/test
- You will be get the pay load message.



- Console log details will be similar like the below:

```
INFO  2016-10-20 09:53:47,263 [main] org.mule.module.launcher.MuleDeploymentService:
**********************************************************************
* Started app 'variabledemo'                                         *
**********************************************************************
INFO  2016-10-20 09:53:47,296 [main] org.mule.module.launcher.DeploymentDirectoryWatcher:
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
+ Mule is up and kicking (every 5000ms)                              +
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
INFO  2016-10-20 09:53:47,306 [main] org.mule.module.launcher.StartupSummaryDeploymentListener:
**********************************************************************
*              - - + DOMAIN + - -               * - - + STATUS + - - *
**********************************************************************
* default                                       * DEPLOYED           *
**********************************************************************

************************************************************************************************************
*            - - + APPLICATION + - -            *       - - + DOMAIN + - -       * - - + STATUS + - - *
************************************************************************************************************
* variabledemo                                  * default                        * DEPLOYED           *
************************************************************************************************************

INFO  2016-10-20 09:56:56,778 [[variabledemo].HTTP_Listener_Configuration.worker.01]
org.mule.api.processor.LoggerMessageProcessor: This is Flow Variable
INFO  2016-10-20 09:56:56,849 [[variabledemo].HTTP_Listener_Configuration.worker.01]
org.mule.api.processor.LoggerMessageProcessor: This is Session Variable.
INFO  2016-10-20 09:56:57,420 [[variabledemo].HTTP_Listener_Configuration.worker.01]
org.mule.lifecycle.AbstractLifecycleManager: Initialising:
'connector.VM.mule.default.dispatcher.2023829251'. Object is: VMMessageDispatcher
INFO  2016-10-20 09:56:57,420 [[variabledemo].HTTP_Listener_Configuration.worker.01]
org.mule.lifecycle.AbstractLifecycleManager: Starting: 'connector.VM.mule.default.dispatcher.2023829251'.
Object is: VMMessageDispatcher
INFO  2016-10-20 09:56:57,717 [[variabledemo].DisplayVariableFlow.stage1.02]
org.mule.api.processor.LoggerMessageProcessor: null
INFO  2016-10-20 09:56:57,717 [[variabledemo].DisplayVariableFlow.stage1.02]
org.mule.api.processor.LoggerMessageProcessor: This is Session Variable.
```

Learning:

- From the above exercise we learnt different type of variables such as Flow variable and Session variable.

You have successfully completed this lab!

LAB – Idempotent Filter

An overview of Idempotent Filter to avoid duplicate messages.

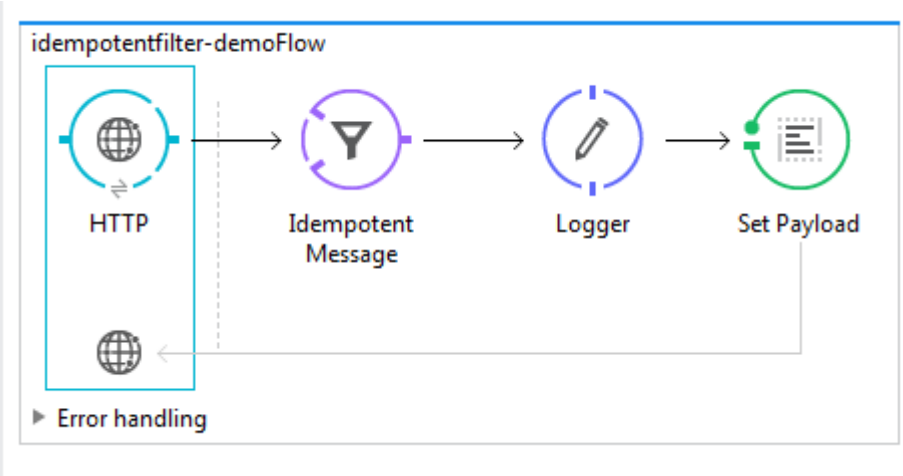
Notes:

Install SOAP UI by using below link to access REST web services.

https://www.soapui.org/downloads/thank-you-for-downloading-soapui.html

Steps:

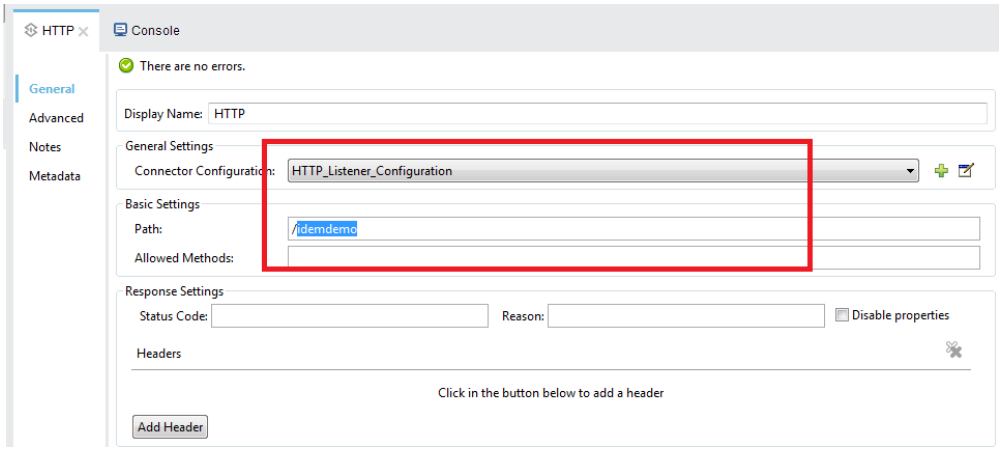
- Open AnyPoint Studio
- File → New → Mule Project
- Give the project name as idempotentfilter-demo and then finish.
- Create the following flow:

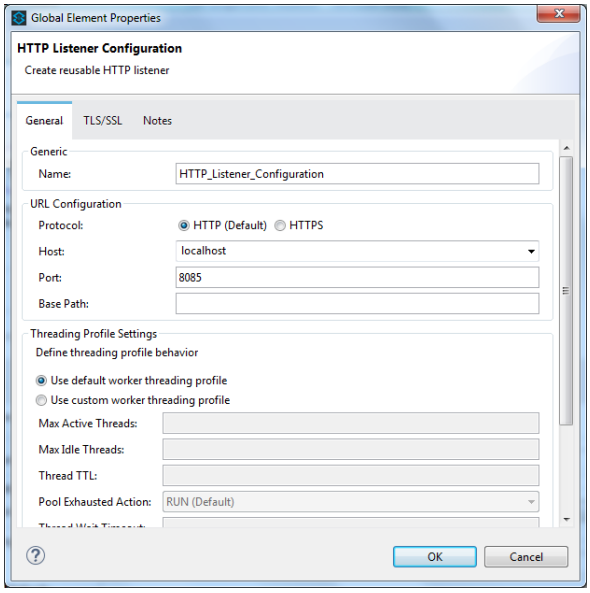


- For the above flow you can look at the Configuration XML as mentioned below:

```
<?xml version="1.0" encoding="UTF-8"?>

<mule xmlns:http="http://www.mulesoft.org/schema/mule/http"
xmlns="http://www.mulesoft.org/schema/mule/core"
xmlns:doc="http://www.mulesoft.org/schema/mule/documentation"
        xmlns:spring="http://www.springframework.org/schema/beans"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:schemaLocation="http://www.springframework.org/schema/beans
http://www.springframework.org/schema/beans/spring-beans-current.xsd
http://www.mulesoft.org/schema/mule/core http://www.mulesoft.org/schema/mule/core/current/mule.xsd
http://www.mulesoft.org/schema/mule/http http://www.mulesoft.org/schema/mule/http/current/mule-http.xsd">
    <http:listener-config name="HTTP_Listener_Configuration" host="localhost" port="8085" doc:name="HTTP
Listener Configuration"/>
    <flow name="idempotentfilter-demoFlow">
        <http:listener config-ref="HTTP_Listener_Configuration" path="/idemdemo" doc:name="HTTP"/>
        <idempotent-message-filter idExpression="#[xpath3('/messagebody/header/msgid')]"
doc:name="Idempotent Message"/>
        <logger message="Passed" level="INFO" doc:name="Logger"/>
        <set-payload value="Passed" doc:name="Set Payload"/>
    </flow>
</mule>
```

- Configure Http Connector using **localhost** with post **8085** and path as "**/idemdemo**".

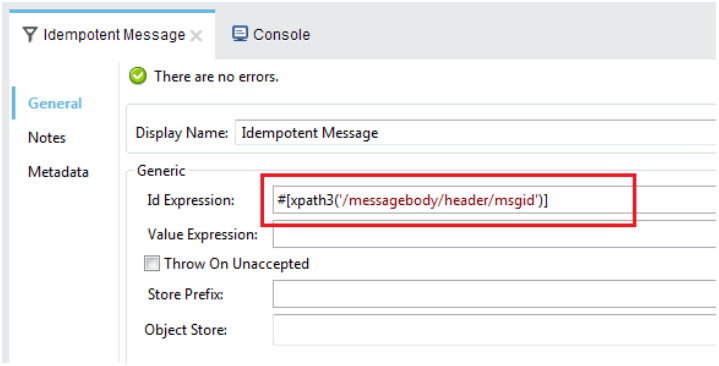




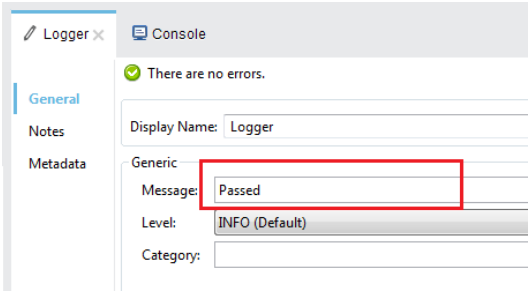
- Consider the below XML file called "**FoodMenu.xml**"

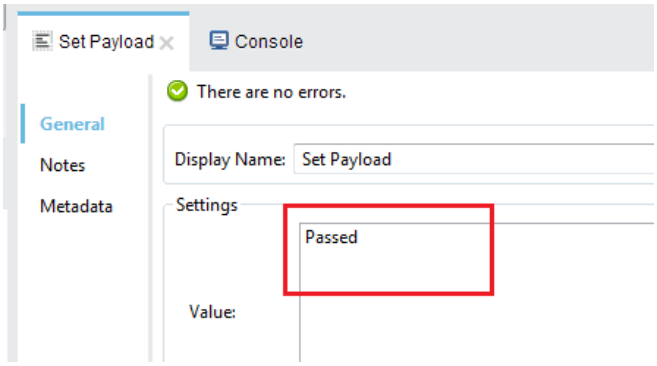
```
<?xml version="1.0" encoding="UTF-8"?>
<messagebody>
<header>
<msgid>001</msgid>
<msgdesc>food menu</msgdesc>
</header>
<breakfast_menu>
<food>
        <name>Belgian Waffles</name>
        <price>$91.45</price>
        <description>Two of our famous Belgian Waffles with plenty of real maple syrup</description>
        <calories>350</calories>
</food>
<food>
        <name>Strawberry Belgian Waffles</name>
        <price>$7.45</price>
        <description>Light Belgian Waffles covered with strawberries and whipped cream </description>
        <calories>900</calories>
</food>

</breakfast_menu>
</messagebody>
```

- To avoid multiple message id mention the Id Expression as **"#[xpath3('/messagebody/header/msgid')]"** in the Idempotent Filter component.



- In logger and set payload mention "passed" as message and value, as given below:



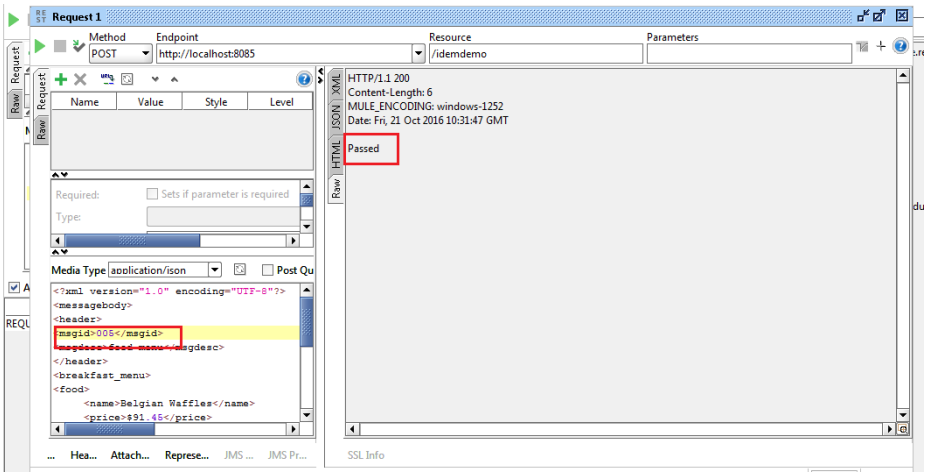


- Now run the application. You will be getting the following log in console once you enter the URL http://localhost:8085/idemdemo

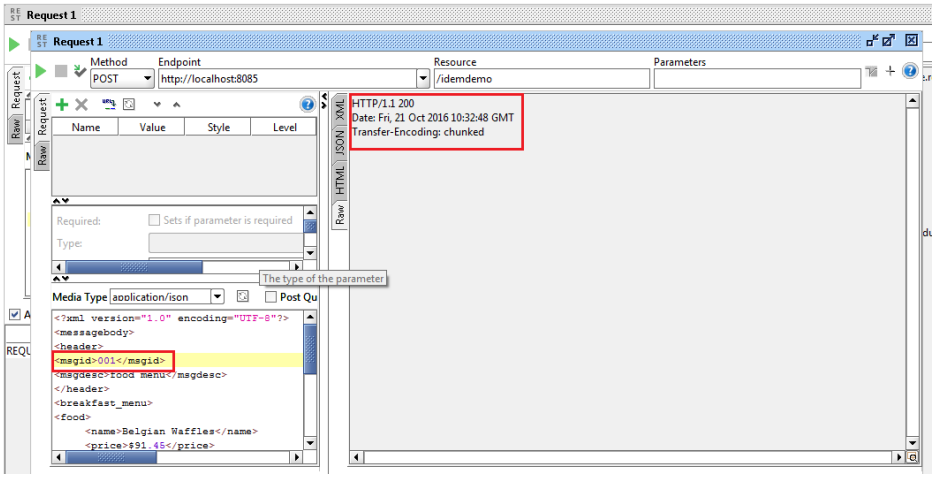
```
**********************************************************************
* Application: idempotentfilter-demo                                 *
* OS encoding: \, Mule encoding: UTF-8                               *
*                                                                    *
* Agents Running:                                                    *
*   JMX Agent                                                        *
*   Batch module default engine                                      *
*   DevKit Extension Information                                     *
*   Wrapper Manager                                                  *
**********************************************************************
```

```
INFO  2016-10-21 16:01:47,498 [[idempotentfilter-demo].HTTP_Listener_Configuration.worker.01]
org.mule.api.processor.LoggerMessageProcessor: Passed
INFO  2016-10-21 16:02:44,245 [[idempotentfilter-demo].HTTP_Listener_Configuration.worker.01]
org.mule.api.processor.LoggerMessageProcessor: Passed
```

- Now open soap UI create one REST web services for the URL http://localhost:8085/idemdemo and pass FoodMenu.xml file via post request. If message id is unique you will get Passed as message.



- If message id is duplicated, you will get below error message.



Learning:

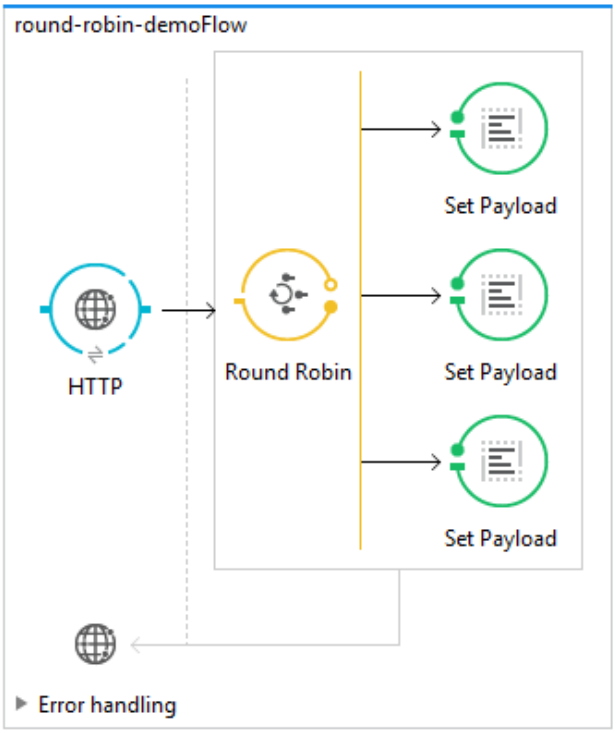
- From the above exercise we learnt how to avoid duplicate message using Idempotent Filter.

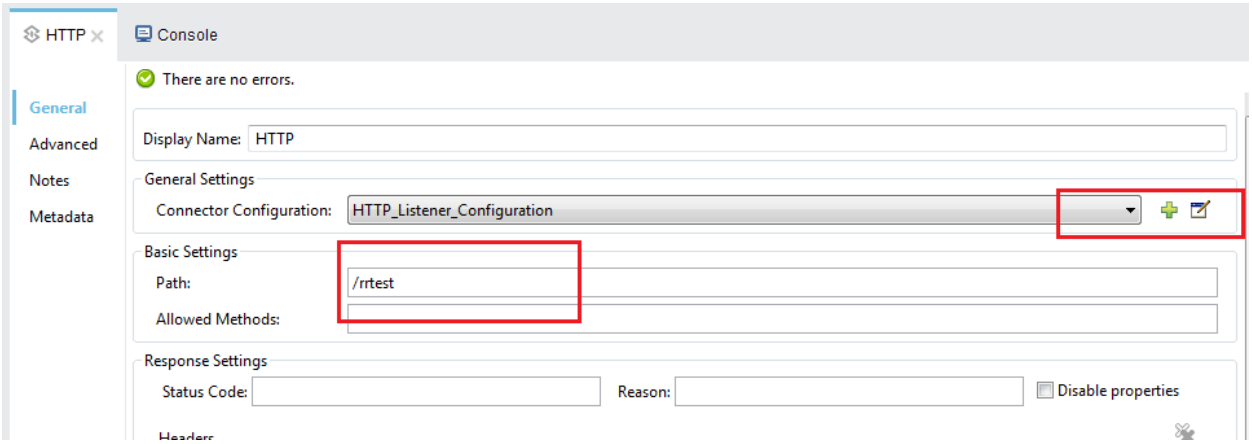
You have successfully completed this lab!

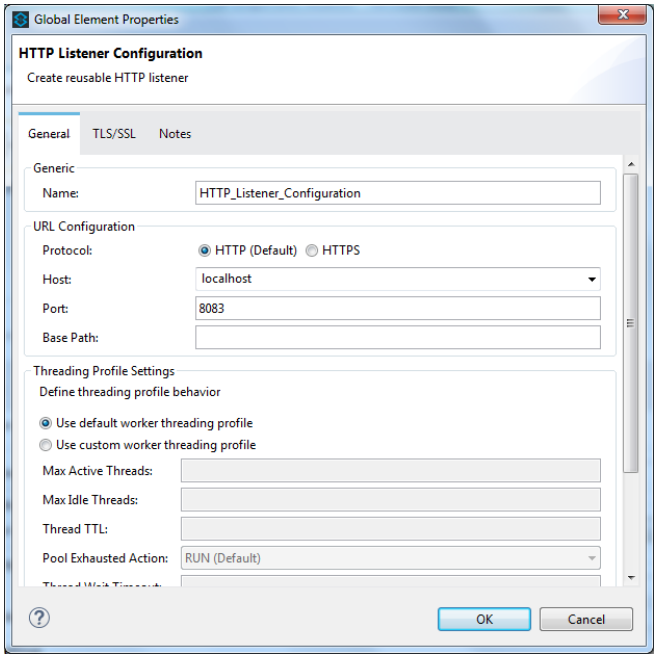
LAB Round Robin

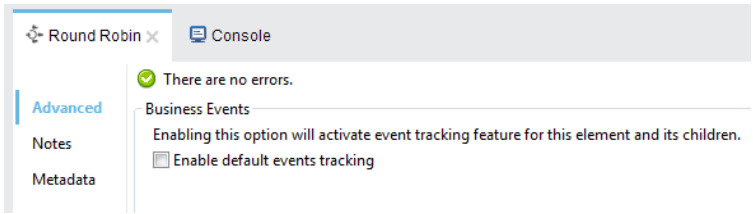
Give an overview about Round Robin.

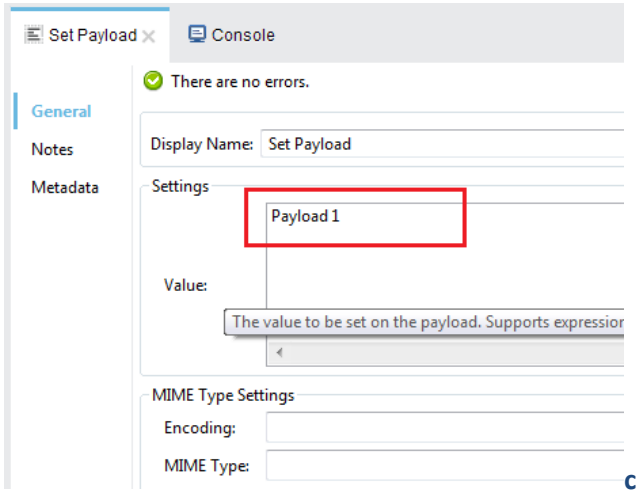
Steps:



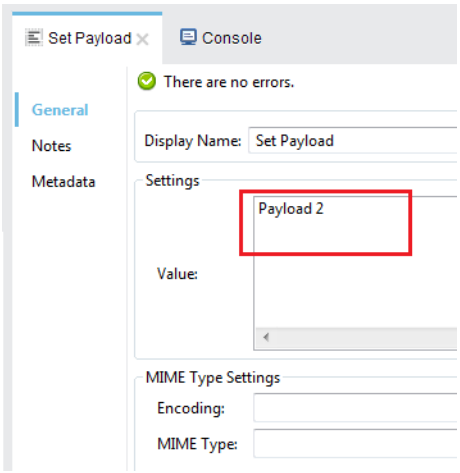



Global Element Properties
HTTP Listener Configuration
Create reusable HTTP listener
General
TLS/SSL
Notes
Generic
Name:
HTTP_Listener_Configuration
URL Configuration
Protocol:
HTTP (Default)
HTTPS
Host:
localhost
Port:
8083
Base Path:
Threading Profile Settings
Define threading profile behavior
Use default worker threading profile
Use custom worker threading profile
Max Active Threads:
Max Idle Threads:
Thread TTL:
Pool Exhausted Action:
RUN (Default)
OK
Cancel

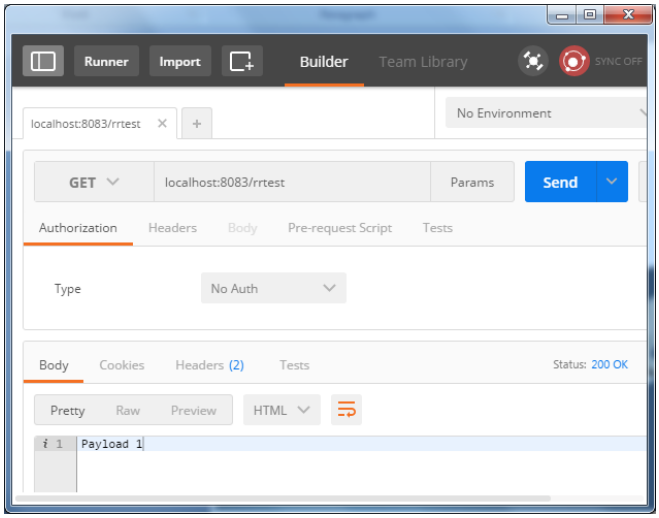

Round Robin
Console
There are no errors.
Advanced
Notes
Metadata
Business Events
Enabling this option will activate event tracking feature for this element and its children.
Enable default events tracking

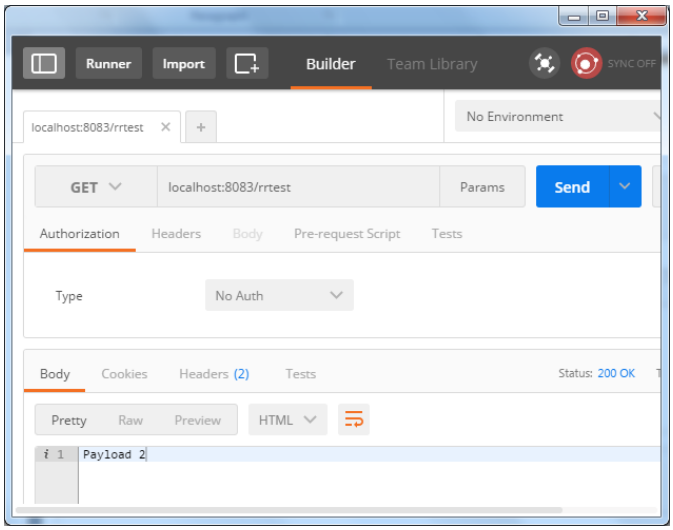

Set Payload
Console
There are no errors.
General
Notes
Metadata
Display Name:
Set Payload
Settings
Payload 1
Value:
The value to be set on the payload. Supports expressior
MIME Type Settings
Encoding:
MIME Type:

c




```
INFO  2016-10-21 17:01:47,833 [main] org.mule.module.management.agent.WrapperManagerAgent: This JVM hasn't been launched by the wrapper, the
agent will not run.
INFO  2016-10-21 17:01:47,863 [main] org.mule.DefaultMuleContext:
**********************************************************************
* Application: round-robin-demo                                      *
* OS encoding: \, Mule encoding: UTF-8                               *
*                                                                    *
* Agents Running:                                                    *
*   JMX Agent                                                        *
*   Batch module default engine                                      *
*   DevKit Extension Information                                     *
*   Wrapper Manager                                                  *
**********************************************************************
INFO  2016-10-21 17:01:47,863 [main] org.mule.module.launcher.MuleDeploymentService:
**********************************************************************
* Started app 'round-robin-demo'                                     *
**********************************************************************
INFO  2016-10-21 17:01:47,933 [main] org.mule.module.launcher.DeploymentDirectoryWatcher:
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
+ Mule is up and kicking (every 5000ms)                             +
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
INFO  2016-10-21 17:01:47,953 [main] org.mule.module.launcher.StartupSummaryDeploymentListener:
**********************************************************************
*              - - + DOMAIN + - -               * - - + STATUS + - - *
**********************************************************************
* default                                       * DEPLOYED           *
**********************************************************************

********************************************************************************************************************
*            - - + APPLICATION + - -            *       - - + DOMAIN + - -       * - - + STATUS + - - *
********************************************************************************************************************
* round-robin-demo                              * default                        * DEPLOYED           *
********************************************************************************************************************
```





Learning:

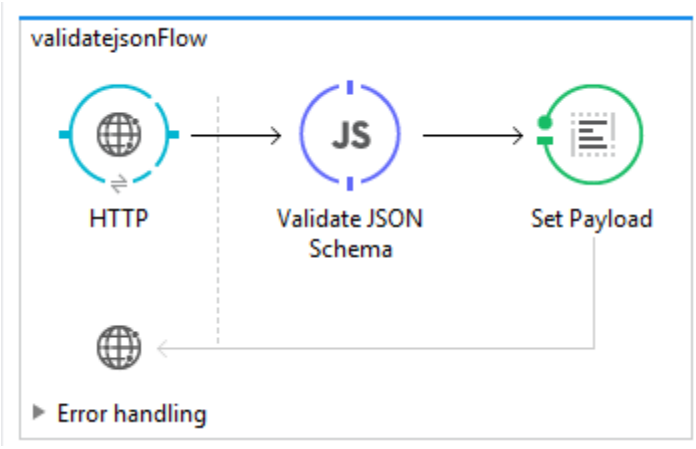
- From the above exercise we learnt how to use VM and when to use VM. Now you can change exchange pattern as “request-response” and see the difference.

You have successfully completed this lab!

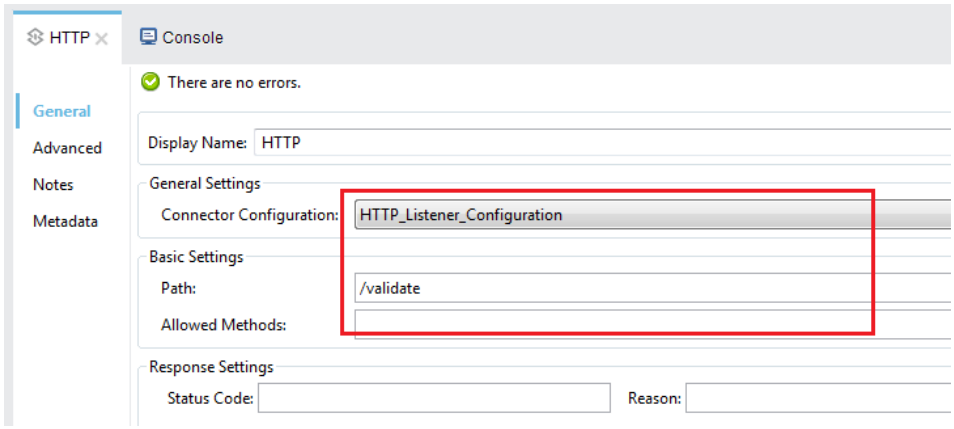
LAB Validate JSON Schema

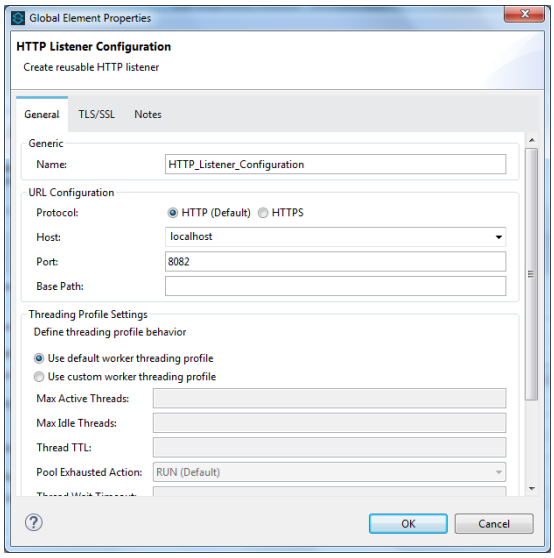
Validate JSON file with JSON Schema.

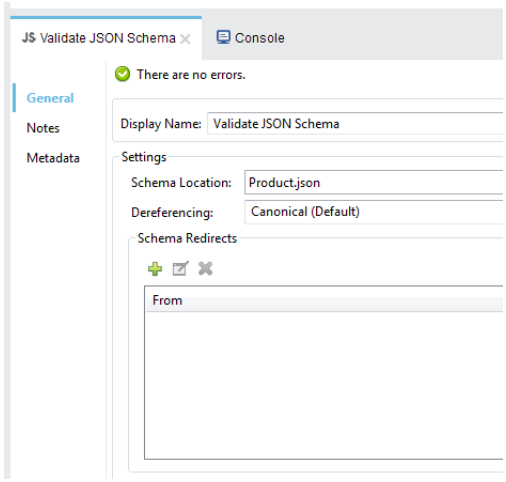
Steps:




```
<?xml version="1.0" encoding="UTF-8"?>

<mule xmlns:json="http://www.mulesoft.org/schema/mule/json"
xmlns:http="http://www.mulesoft.org/schema/mule/http"
xmlns="http://www.mulesoft.org/schema/mule/core"
xmlns:doc="http://www.mulesoft.org/schema/mule/documentation"
        xmlns:spring="http://www.springframework.org/schema/beans"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:schemaLocation="http://www.springframework.org/schema/beans
http://www.springframework.org/schema/beans/spring-beans-current.xsd
http://www.mulesoft.org/schema/mule/core
http://www.mulesoft.org/schema/mule/core/current/mule.xsd
http://www.mulesoft.org/schema/mule/json
http://www.mulesoft.org/schema/mule/json/current/mule-json.xsd
http://www.mulesoft.org/schema/mule/http
http://www.mulesoft.org/schema/mule/http/current/mule-http.xsd">
    <http:listener-config name="HTTP_Listener_Configuration" host="localhost" port="8082"
doc:name="HTTP Listener Configuration"/>
    <flow name="validatejsonFlow">
        <http:listener config-ref="HTTP_Listener_Configuration" path="/validate"
doc:name="HTTP"/>
        <json:validate-schema schemaLocation="Product.json" doc:name="Validate JSON Schema"/>
        <set-payload value="Passed" doc:name="Set Payload"/>
    </flow>
</mule>
```


HTTP
Console
There are no errors.
General
Advanced
Notes
Metadata
Display Name: HTTP
General Settings
Connector Configuration: HTTP_Listener_Configuration
Basic Settings
Path: /validate
Allowed Methods:
Response Settings
Status Code:
Reason:


Global Element Properties
HTTP Listener Configuration
Create reusable HTTP listener
General
TLS/SSL
Notes
Generic
Name: HTTP_Listener_Configuration
URL Configuration
Protocol: HTTP (Default) HTTPS
Host: localhost
Port: 8082
Base Path:
Threading Profile Settings
Define threading profile behavior
Use default worker threading profile
Use custom worker threading profile
Max Active Threads:
Max Idle Threads:
Thread TTL:
Pool Exhausted Action: RUN (Default)
OK
Cancel


JS Validate JSON Schema
Console
There are no errors.
General
Notes
Metadata
Display Name: Validate JSON Schema
Settings
Schema Location: Product.json
Dereferencing: Canonical (Default)
Schema Redirects
From

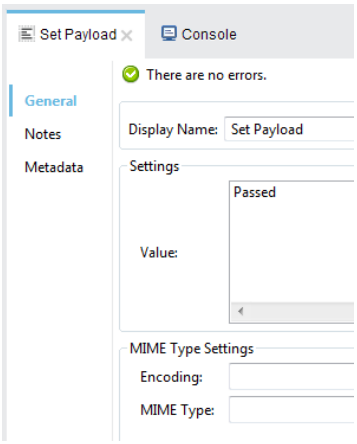
Product.json (schema)

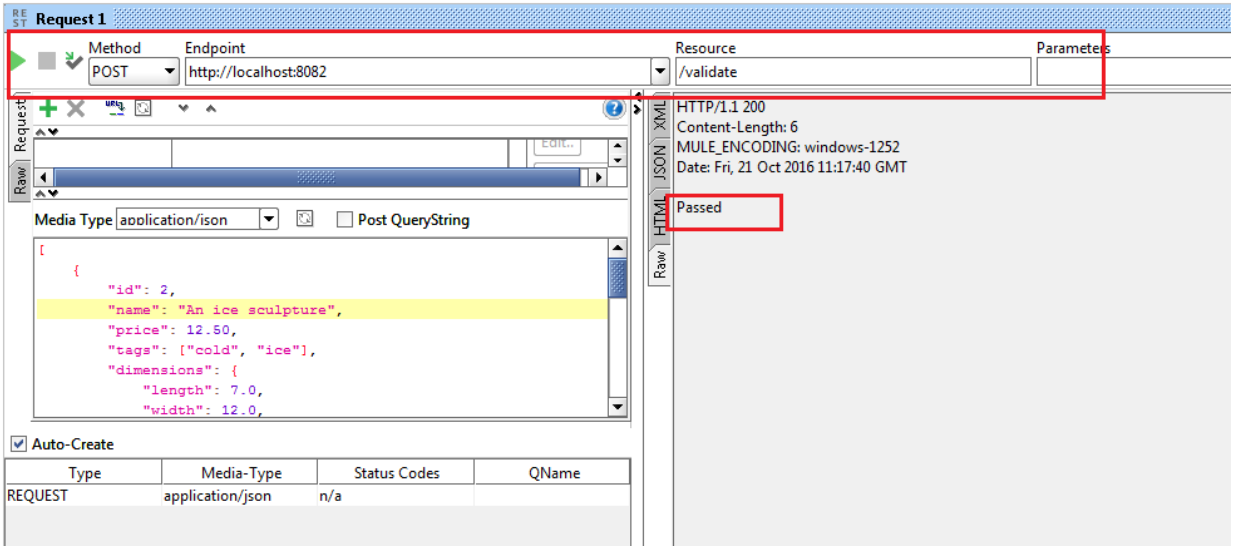
```
{
    "$schema": "http://json-schema.org/draft-04/schema#",
    "title": "Product set",
    "type": "array",
    "items": {
        "title": "Product",
        "type": "object",
        "properties": {
            "id": {
                "description": "The unique identifier for a product",
                "type": "number"
            },
            "name": {
                "type": "string"
            },
            "price": {
                "type": "number",
                "minimum": 0,
                "exclusiveMinimum": true
            },
            "tags": {
                "type": "array",
                "items": {
                    "type": "string"
                },
                "minItems": 1,
                "uniqueItems": true
            },
            "dimensions": {
                "type": "object",
                "properties": {
                    "length": {"type": "number"},
                    "width": {"type": "number"},
                    "height": {"type": "number"}
                },
                "required": ["length", "width", "height"]
            },
            "warehouseLocation": {
                "description": "Coordinates of the warehouse with the product",
                "$ref": "http://json-schema.org/geo"
            }
        },
        "required": ["id", "name", "price"]
    }
}
```

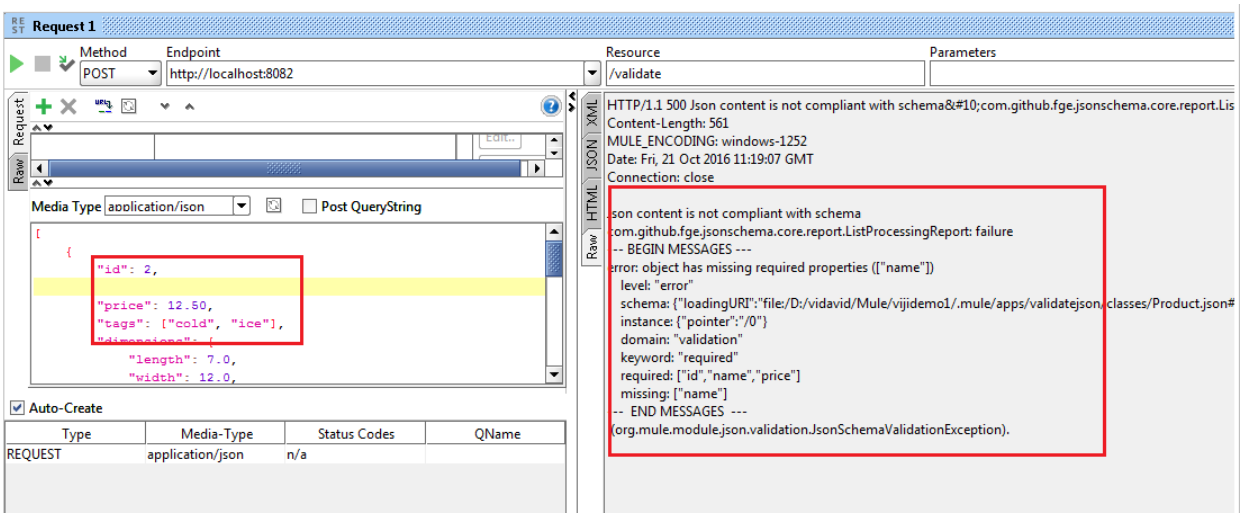
Product.json

```
[
    {
        "id": 2,
        "name": "An ice sculpture",
        "price": 12.50,
        "tags": ["cold", "ice"],
        "dimensions": {
```

```
            "length": 7.0,
            "width": 12.0,
            "height": 9.5
        },
        "warehouseLocation": {
            "latitude": -78.75,
            "longitude": 20.4
        }
    },
    {
        "id": 3,
        "name": "A blue mouse",
        "price": 25.50,
            "dimensions": {
            "length": 3.1,
            "width": 1.0,
            "height": 1.0
        },
        "warehouseLocation": {
            "latitude": 54.4,
            "longitude": -32.7
        }
    }
]
```





Learning:

- From the above exercise we learnt how to use VM and when to use VM. Now you can change exchange pattern as “request-response” and see the difference.

You have successfully completed this lab!

LAB Composite Source

Demo for Composite Source.

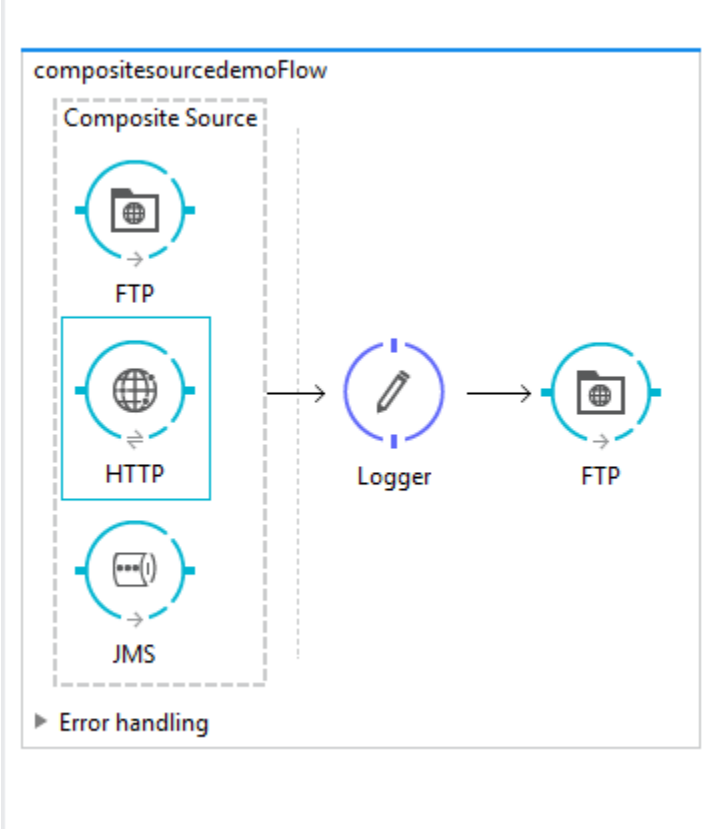
Pre-requisite:

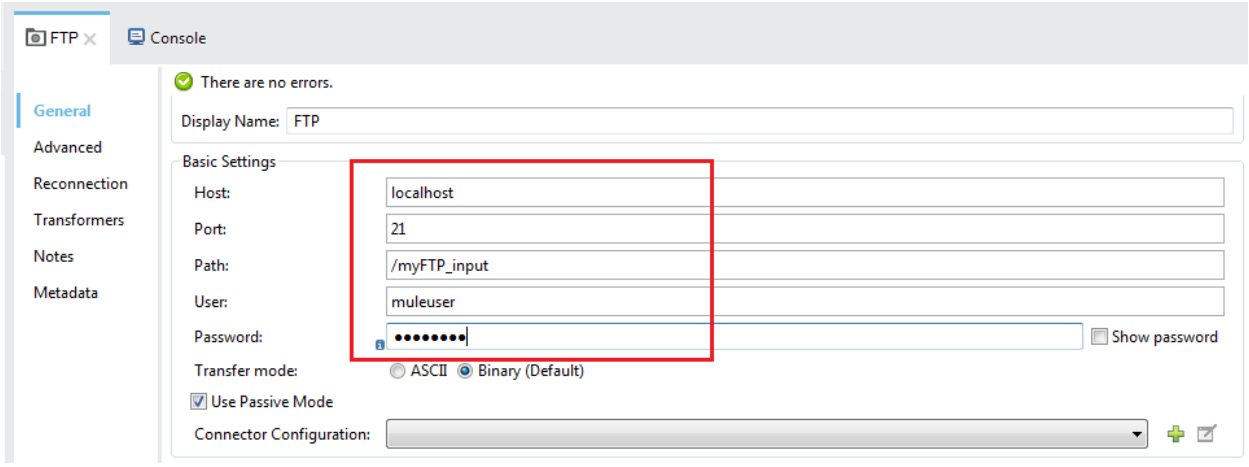
Download FileZilla Server and Client. Install both server and Client to access share folder under FTP.

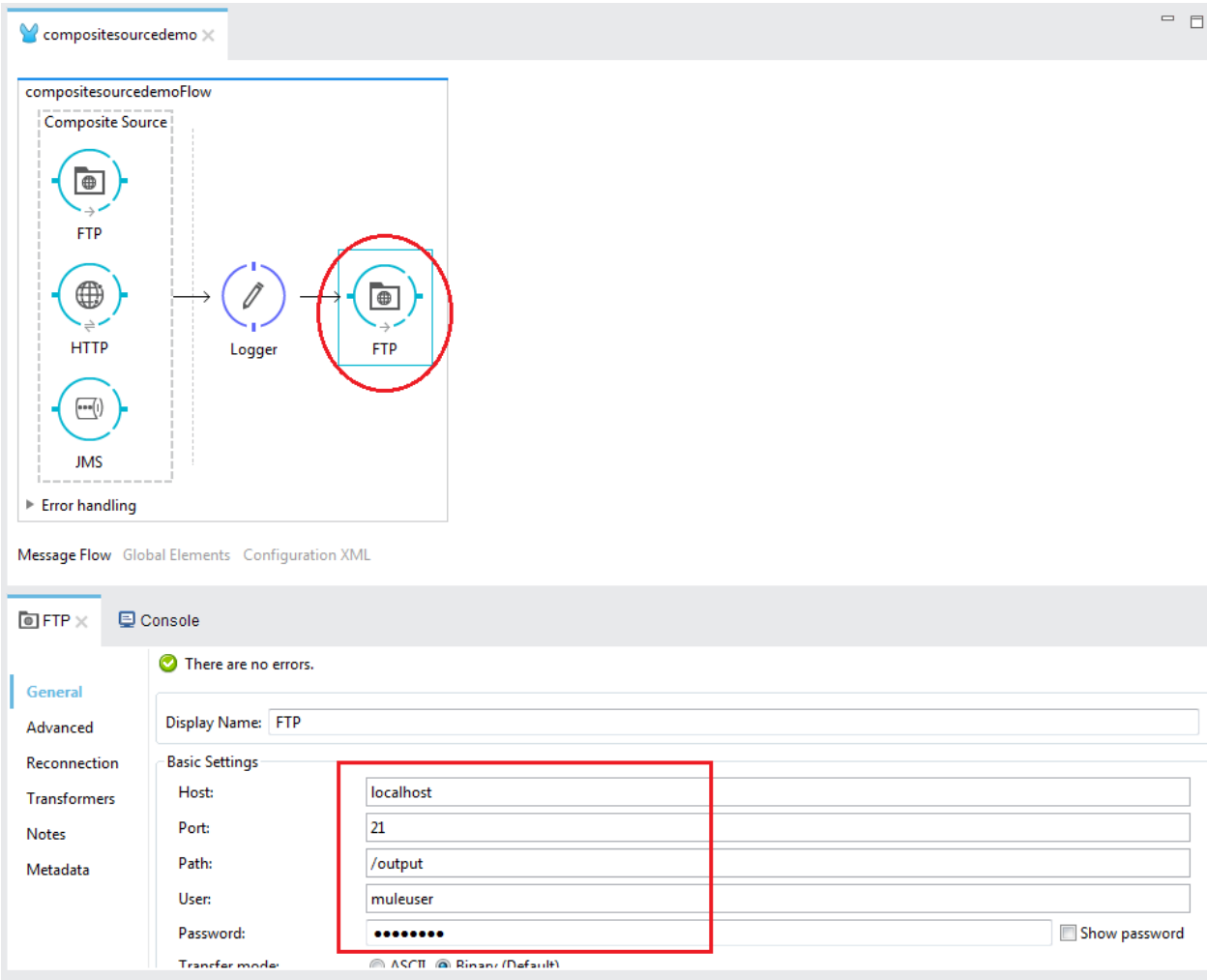
https://filezilla-project.org/download.php?type=client

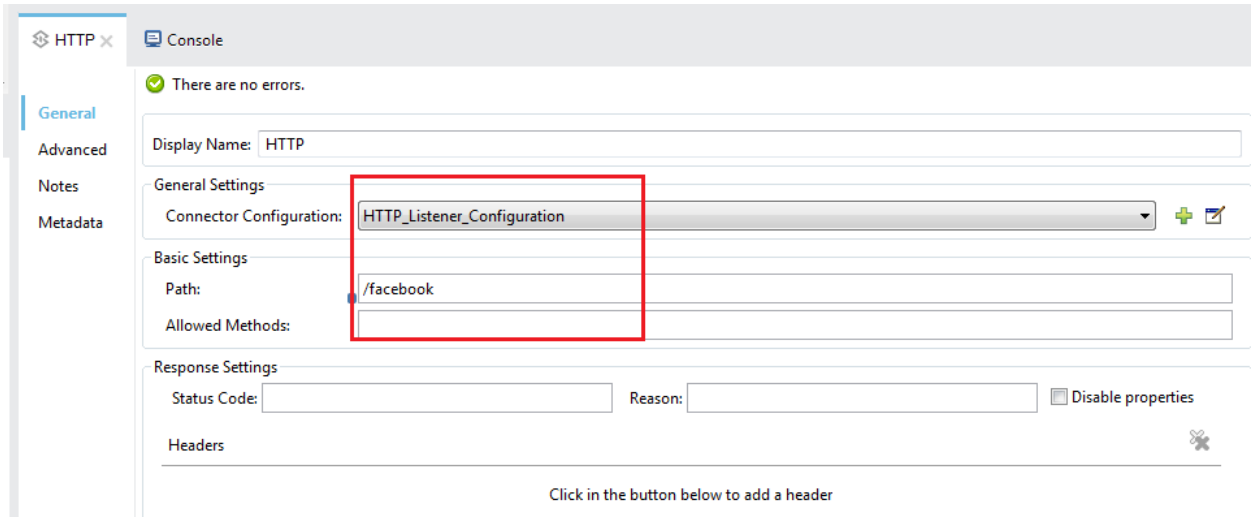
Steps:

- Open AnyPoint Studio
- File → New → Mule Project
- Give the project name as compositesourcedemo as mentioned below. And then finish.
- Create the below flow with compositesource.




FTP
Console
There are no errors.
General
Advanced
Reconnection
Transformers
Notes
Metadata
Display Name: FTP
Basic Settings
Host: localhost
Port: 21
Path: /myFTP_input
User: muleuser
Password: ••••••••
Show password
Transfer mode: ASCII Binary (Default)
Use Passive Mode
Connector Configuration:


compositesourcedemo
compositesourcedemoFlow
Composite Source
FTP
HTTP
JMS
Logger
FTP
Error handling
Message Flow
Global Elements
Configuration XML
FTP
Console
There are no errors.
General
Advanced
Reconnection
Transformers
Notes
Metadata
Display Name: FTP
Basic Settings
Host: localhost
Port: 21
Path: /output
User: muleuser
Password: ••••••••
Show password



Learning:

- From the above exercise we learnt how to use Composite Source with various incoming channels.

You have successfully completed this lab!

LAB Active MQ Connector

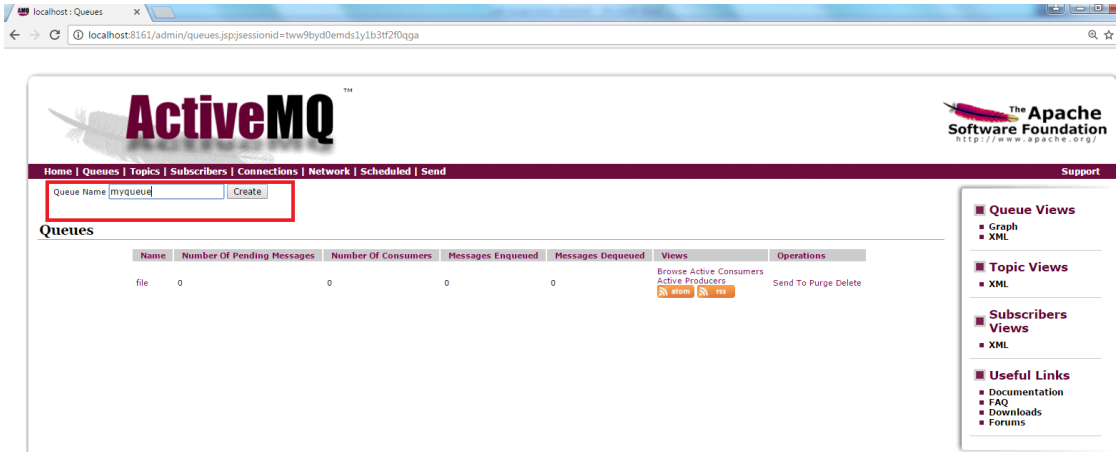
Give an example about Active MQ Connector.

Steps:

- **Download Activemq from the below link.**
  - **Select latest release and then download zip**

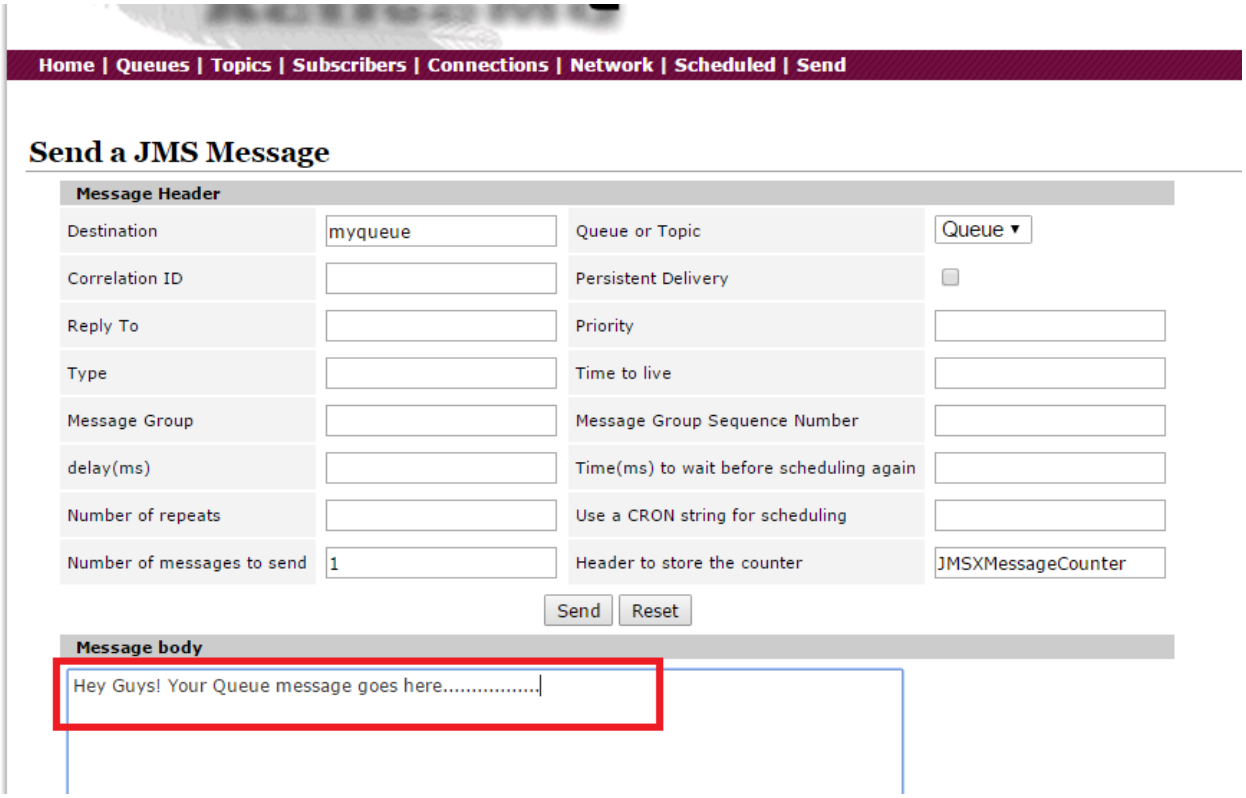
  http://activemq.apache.org/download.html
- Unzip the folder and then locate **bin** directory choose either **win32/win64**, it depends on your windows machine.
- Start **activemq** service.
- Once the service started, go to browser and then type http://localhost:8161/ it will open the Active MQ browser window.

- In this window click Manage ActiveMQ broker , provide username and password as admin
  - In this window click menu link called Queue to create new Queue.
  - In the Queue Name box enter “Queue Name” such as “myQueue”

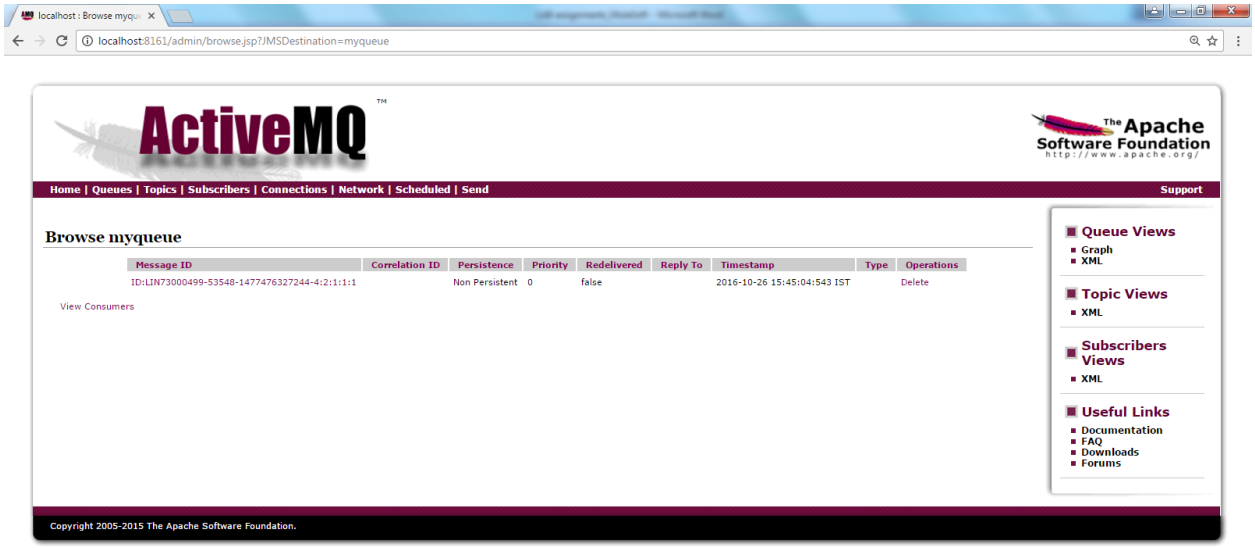


  - Now you could see the queue name in the list. Click send to link related to your queue name called ”myqueue”.

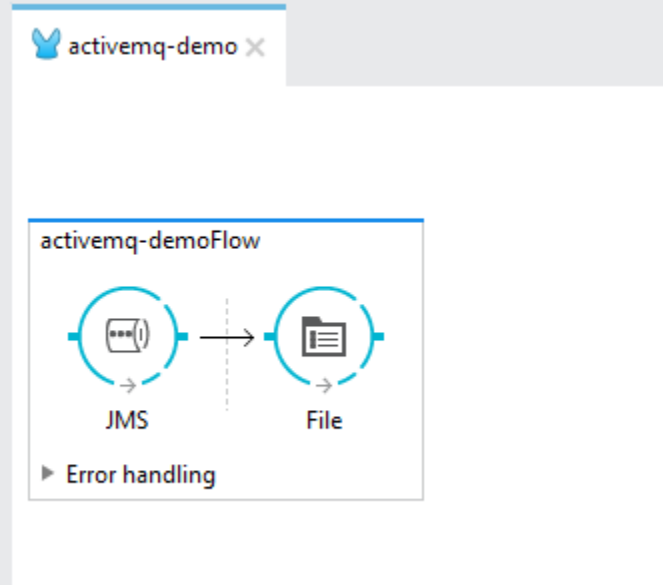
  - Now enter queue message as highlighted in the below. And then click send.



- Once you click the queue name in the list, you will be seeing the queue configuration details as mentioned below:



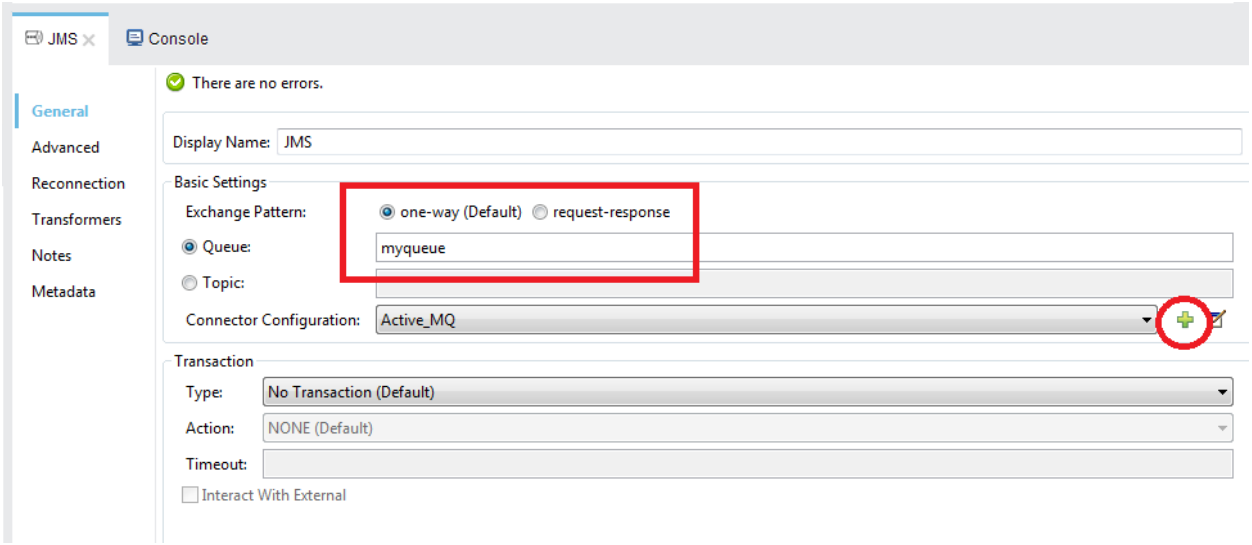
- Create the following flow:



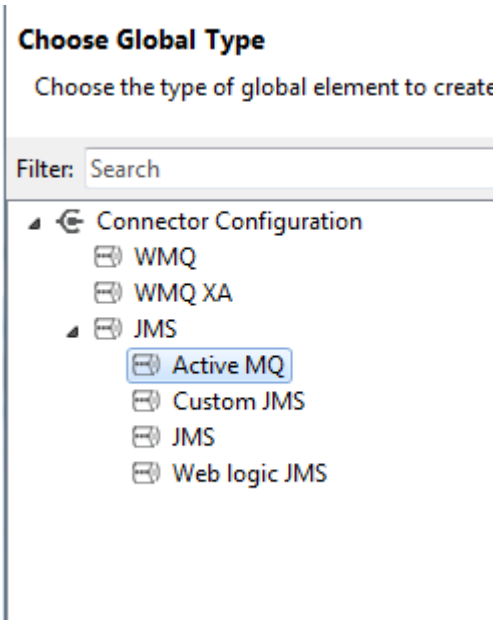
- For the above flow below is the configuration file.

```
<?xml version="1.0" encoding="UTF-8"?>

<mule xmlns:file="http://www.mulesoft.org/schema/mule/file"
xmlns:jms="http://www.mulesoft.org/schema/mule/jms"
xmlns:ftp="http://www.mulesoft.org/schema/mule/ee/ftp"
xmlns="http://www.mulesoft.org/schema/mule/core"
xmlns:doc="http://www.mulesoft.org/schema/mule/documentation"
       xmlns:spring="http://www.springframework.org/schema/beans"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xsi:schemaLocation="http://www.springframework.org/schema/beans
http://www.springframework.org/schema/beans/spring-beans-current.xsd
http://www.mulesoft.org/schema/mule/core
http://www.mulesoft.org/schema/mule/core/current/mule.xsd
http://www.mulesoft.org/schema/mule/ee/ftp
http://www.mulesoft.org/schema/mule/ee/ftp/current/mule-ftp-ee.xsd
http://www.mulesoft.org/schema/mule/jms http://www.mulesoft.org/schema/mule/jms/current/mule-
jms.xsd
http://www.mulesoft.org/schema/mule/file
http://www.mulesoft.org/schema/mule/file/current/mule-file.xsd">
    <jms:activemq-connector name="Active_MQ" brokerURL="tcp://localhost:61616"
validateConnections="true" doc:name="Active MQ"/>
    <flow name="activemq-demoFlow">
        <jms:inbound-endpoint queue="myqueue" connector-ref="Active_MQ" doc:name="JMS"/>
        <file:outbound-endpoint path="D:\vidavid\demomule\sample" responseTimeout="10000"
doc:name="File"/>
    </flow>
</mule>
```

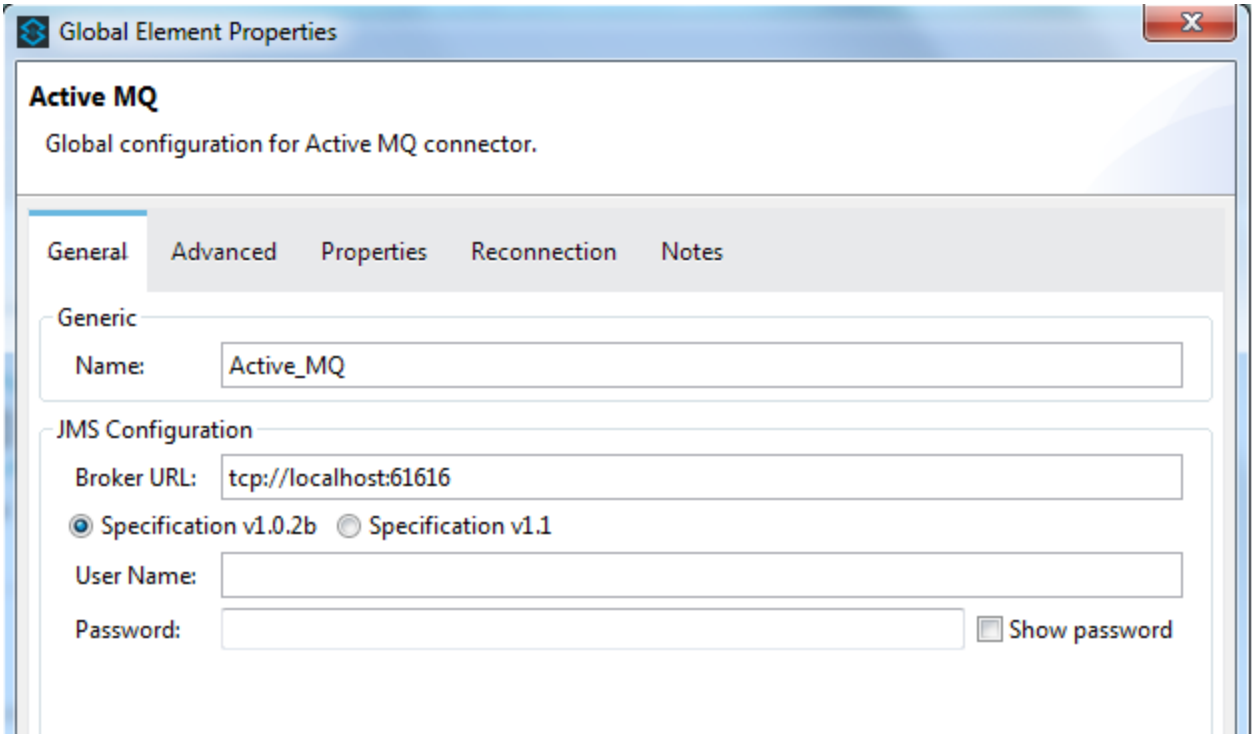
- Mention the below properties for JMS.



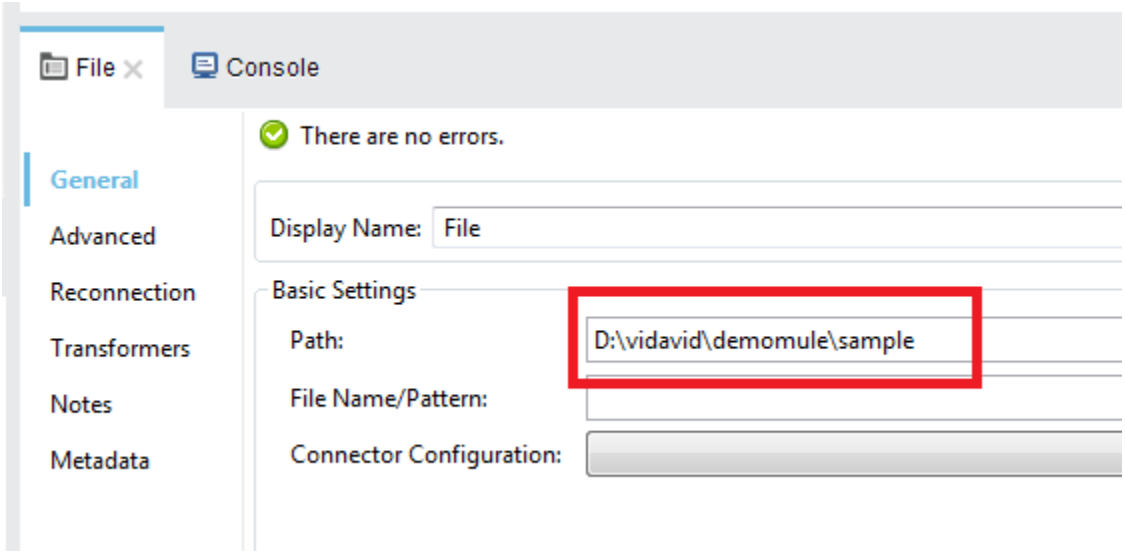
- Click plus sign which has been highlighted here.
- Select Active MQ configuration



- Click OK.
- And then click OK for the below screen.



- Configure the file property as mentioned below:



- Right Click project → Run As→ Mule Application.
- Once the application started you will be getting the below log messages in the console log:

```
* Started app 'activemq-demo'                                          *
* Application libraries:                                               *
*   - activemq-all-5.14.1.jar                                          *
**********************************************************************
INFO  2016-10-26 15:56:34,254 [main] org.mule.module.launcher.DeploymentDirectoryWatcher:
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
+ Mule is up and kicking (every 5000ms)                                +
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
INFO  2016-10-26 15:56:34,347 [main] org.mule.module.launcher.StartupSummaryDeploymentListener:
**********************************************************************
```

```
*              - - + DOMAIN + - -               * - - + STATUS + - - *
********************************************************************
* default                                      * DEPLOYED           *
********************************************************************

************************************************************************************************************
*            - - + APPLICATION + - -            *       - - + DOMAIN + - -       * - - + STATUS + - - *
************************************************************************************************************
* activemq-demo                                 * default                        * DEPLOYED           *
************************************************************************************************************

INFO  2016-10-26 15:57:16,623 [[activemq-demo].connector.file.mule.default.dispatcher.01]
org.mule.lifecycle.AbstractLifecycleManager: Initialising: 'connector.file.mule.default.dispatcher.195468091'. Object is:
FileMessageDispatcher
INFO  2016-10-26 15:57:16,727 [[activemq-demo].connector.file.mule.default.dispatcher.01]
org.mule.lifecycle.AbstractLifecycleManager: Starting: 'connector.file.mule.default.dispatcher.195468091'. Object is:
FileMessageDispatcher
INFO  2016-10-26 15:57:16,732 [[activemq-demo].connector.file.mule.default.dispatcher.01]
org.mule.transport.file.FileConnector: Writing file to: D:\vidavid\demomule\sample\c4837da0-9b66-11e6-9486-
9e6e20524153.dat
```

- Now you could check the file location. One new file will be created with the queue message.

Learning:

- From the above exercise we learnt how to configure Active MQ within JMS. Sending the messages to various component.

You have successfully completed this lab!