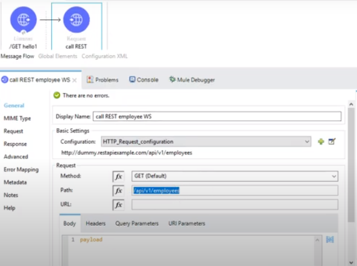
**1) Types of Integration:**

* File based Integration pattern
* JDBC Integration pattern
* Batch Processing and scheduled based Integration pattern
* Publish Subscribe pattern
* Polling Consumer pattern
* Point to Point Integration pattern
* Guaranteed Delivery message pattern
* Operating system Command Execution pattern
* Webservice Synchronous Consumer pattern
* Webservice Provider Pass-through pattern

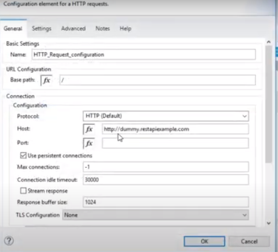
**2) Components for REST vs SOAP**

* For REST API, use HTTP 'Request'.
* For SOAP API, use 'Web Service Consumer' component.

**For REST API, Below is the flow**



Edit the connection wizard like below with the REST API url:



**3) Scheduler Component**

For a Scheduler based event trigger, use 'Scheduler' component in a Flow so that the scheduler will get triggered based on the CRON time that we mentioned and it can trigger other components/Flows

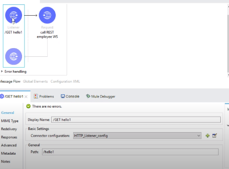
* Under the 'Scheduling Strategy' , set the frequency as either 'Fixed Frequency' or 'CRON' and other parameters like Daily, Hourly etc

**4) To Create SOAP API**

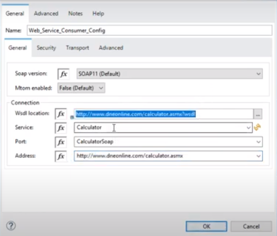
Under the 'Add Modules’, Add 'Web Service Consumer' component by dragging and dropping it. Now use the consume method inside it and configure it.

Now once this is added into the Module, The relevant Jar files will be added inside the project.

* Add the HTTP listener to invoke the consumer
* Configure the wsdl parameter to point the wsdl url from where you want to consume the data like below



Below are the wsdl information from the sample example from website:



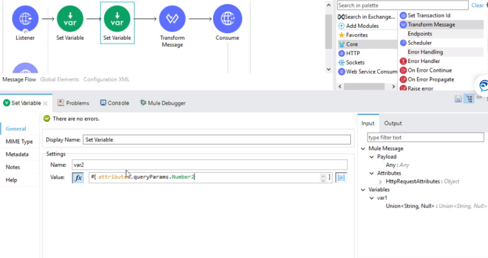
Now, we can pass the numbers from the postman rather than hardcoded in ‘set variable’

We are passing the value parameters from postman:

Add ‘Transform Message’ component to map the input variables to the output variables by mapping the variables in the mapping area:

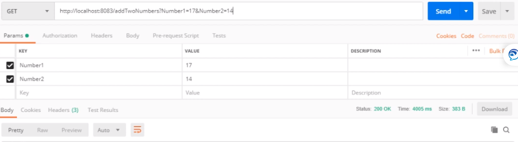
In the ‘Set Variable’ settings value, do below:

fx #[attributes.queryParams.Number1



The parameters are passed as:

http://localhost:port#/addTwoNumbers/Number1=4&Number2=5



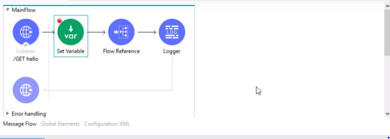
In order to get the output in the ‘Logger’ component, give below expression:

Message: fx : #[payload.body.AddResponse.AddResult

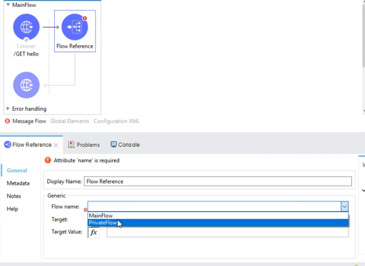
**5) Main Flow, Private Flow and Subflow :**

Here we can create a Main flow and then we can use a ‘Flow Reference’ component inside that to reference to either a **Private Flow** (it’s the same Flow but just renamed as Private Flow) or a ‘Subflow’ component’.

* Subflow are mainly a proven working model and can be reused/referenced in any flow, It will not have any error handler and the main flow’s error handler will be handling it.
* Based on conditions we can either invoke a Private Flow or a Subflow.



We can call the private flow or the Subflow from the ‘Flow Reference’s ‘Flow Name’ drop down list.



**6) Publish and Subscribe through VMConnector:**

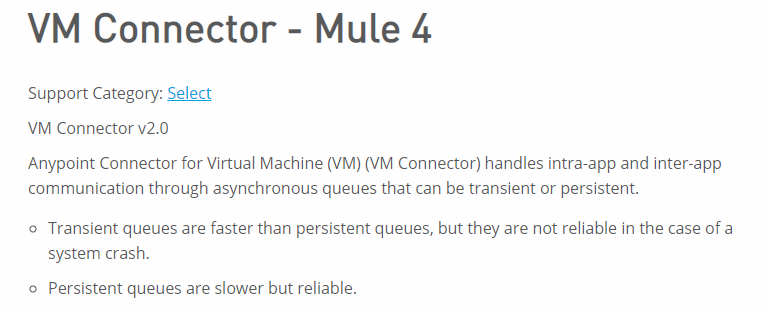
This Integration is used when you have to invoke from one mule app to another or to distribute one message to multiple recipients, we can use publish method and then the clients/consumers can subscribe to it to get the messages.

Refer to below Mule documentation for this:

<https://docs.mulesoft.com/connectors/vm/vm-connector>

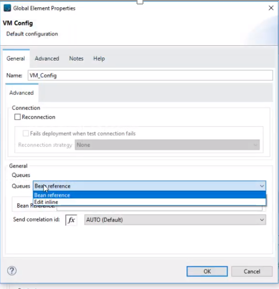
The messages can kept as Transient or persistent depending up the requirement.

This works in Asynchronous way.

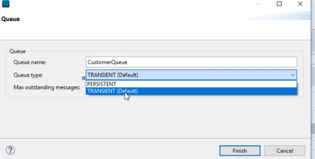


Below we can change the config properties to point ot ‘Edit Inline’ for Queues.

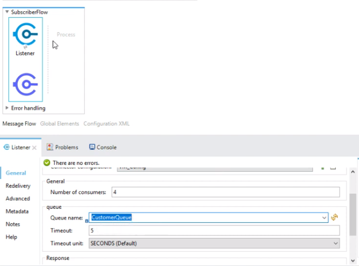
This will popup the QueueName and Type wizard and choose either ‘Persistent’ or ‘Transient’



Choose the Queue type:



Now let’s create subscriberFlow:



* Publish and Subscribe will happen in an Asynchronous way.
* PublishConsume will happen in a Synchronous way

Below we changed publish to publishConsume and have this consumed in our Subscribe Flow so that the control from publish flow will end only after it is consumed and thus happens in a Synchronous way.



**7) Scatter\_Gather Flow:**

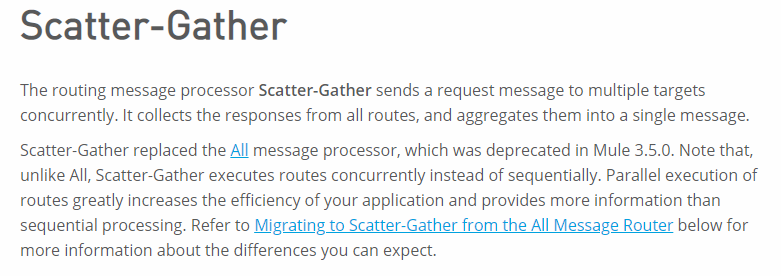
**Business Use case: Suppose you have data coming from 3 different systems like product from one system, price from another system and inventory from another source, they can all be fetched in parallel by using scatter gather, let’s say if each takes 3 seconds then total fetch time is still 3 seconds instead of 9 seconds if they run in sequence. So the main purpose is to increase speed and performance.**

**Below is the Link for Mule Documentation:**

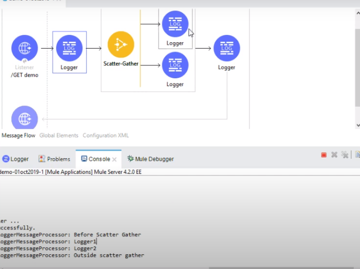
<https://docs.mulesoft.com/mule-runtime/3.9/scatter-gather>

Scatter and Gather should have at least more than one flow coming out of it.

**What is Scatter-Gather.**



**Below it shows how the logger has been invoked showing the flow of Scatter\_Gather**



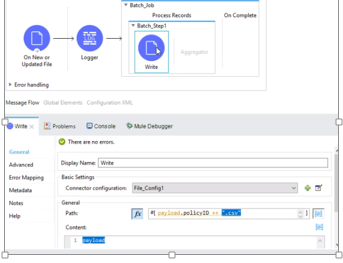
**8) BATCH PROCESSING (File processing):**

Mulesoft can do ETL like work by looking out for a file and then process them in a batch process for a huge file.

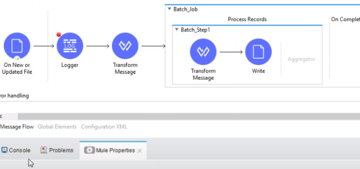
‘BatchJob’ component and ‘File write’ components can be used to read a bulk file and process it and output into an output file.

A Batch steps process 1600 records at a time with 100 records processed/thread in parallel at a time.

So total is 16 threads.



Below file is transformed from csv to Java using ‘Transform message’ and then again converted to java to csv after processing and before writing it into output file



**9) Invoke Static component:**

A file can be split up into multiple files by using a Java class and that java class can be invoked by ‘Invoke Static’ component

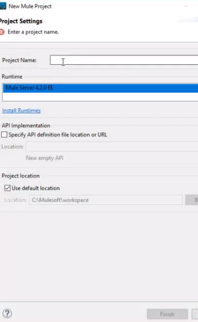
**10) How to import a SOAP wsdl into Mule Project**

Below click the option of ‘Specify API definition file location or URL’ so that we can import the wsdl.

The Mule project will be created with the SOAP components needed for the project.

We can create a wsdl using ‘Java Eclipse’ or any other Online SOAP development kit.

Then we can test the SOAP based mule project through the SOAPUI online otol.

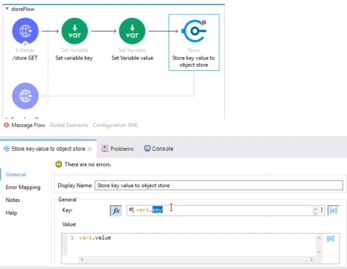




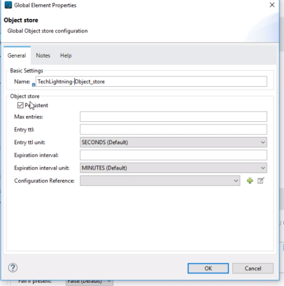
We can see where the SOAP API is imported into Mule project.

**11) ObjectStore:**

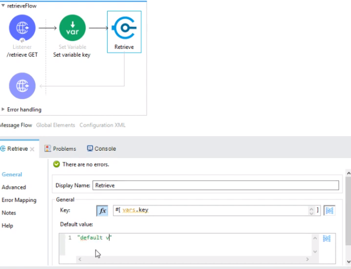
If we want to store and retrieve the parameters which have been passed on to the API end point then we can use ‘object store’ and ‘retrieve connecters to store the data



We need to choose ‘persistent’ option so that the value is stored on Disk.



Below you can retrieve the data using the retrieve component



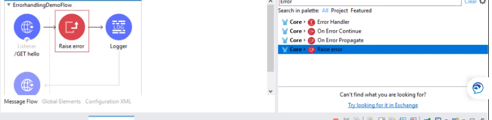
**12) Error Handling:**

There are 4 types of errors in Mulesoft

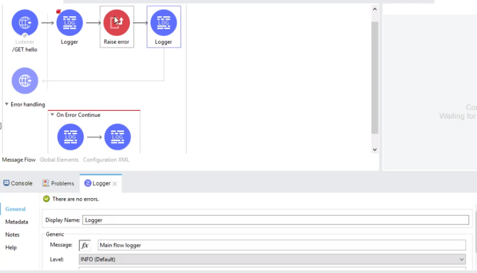
* Error Handler
* On Error Continue
* On Error Propagate
* Raise Error

Difference between ‘On Error Propagate’ and ‘On Error Continue’

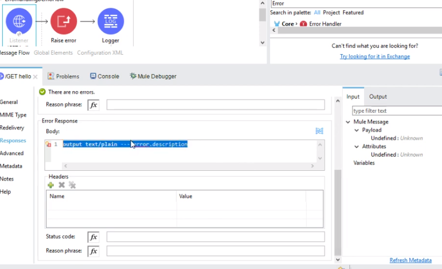
* On Error Continue will be exiting with Success
* On Error Propagate will be exiting with Failure



Below we can see how On Error Continue is placed inside the Error Handling component.



We can see below an Error response in displayed as Http listener response is configured with default error response.



**13) Mule Domain Project:**

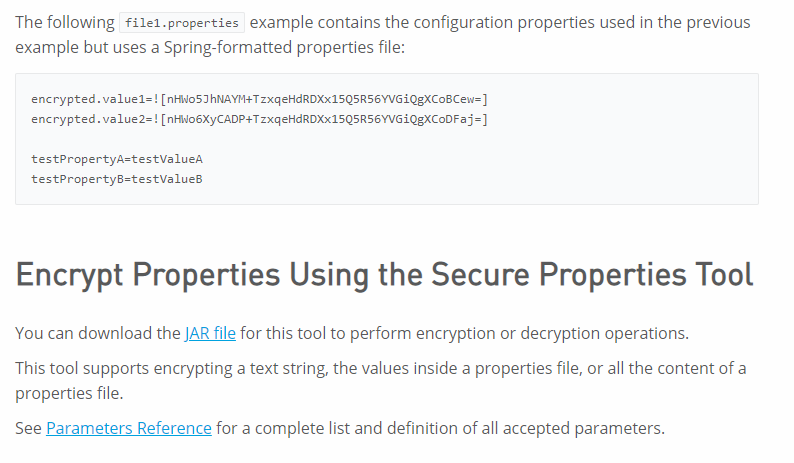
A Mule domain project is created for Mule run time applications to share their resources.

Like all the configurations from global components can be stored here and individual mule application’s properties can be mapped to this mule domain project so that the confogurations can be shared/reused by all the applications pointed to this ‘Mule Domain project’.

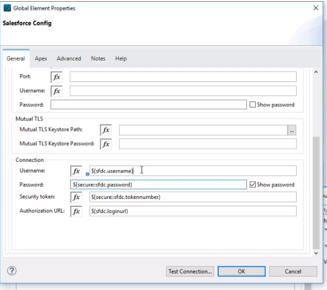
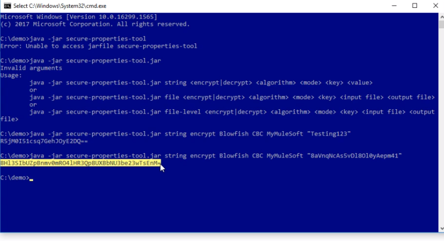
**14) Secure Configuration:**

Download the Jar file from below link and encrypt the credentials using this jar file and create the encrypted token and use that in connection properties for the component.

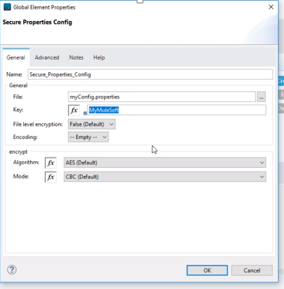
<https://docs.mulesoft.com/mule-runtime/4.2/secure-configuration-properties>



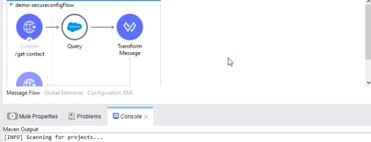
Below is how we encrypt using CLI



Now add the secure properties config from the Exchange into Global elements file and then point this secure config file like below:



Below is a typical flow how we configured to connect to salesforce database using query method and by using seco=ure config parameters:



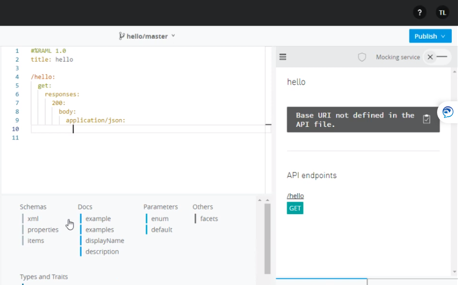
**15) Anypoint.mulesoft.com:**

Using Anypoint Cloud Portal we can do the following:

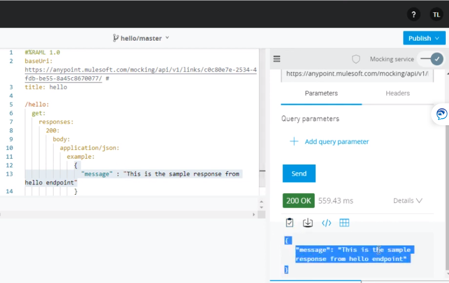
* We can create RAML in Design Center
* Publish RAML in Exchange
* Download RAML into Any point Studio
* Rn the Application
* Create API in API Manager
* Create a Policy
* Run the application

From the Design center, we can create a ‘New’ ‘API Specification’ to create an API.

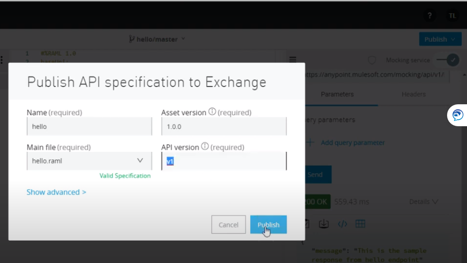
Create a RAML in API Specification.



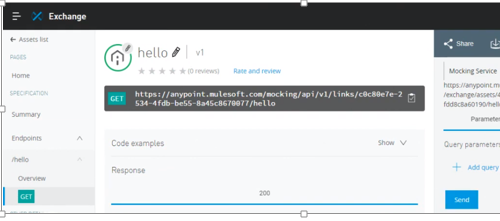
We can test it using Mock services, test url like below:



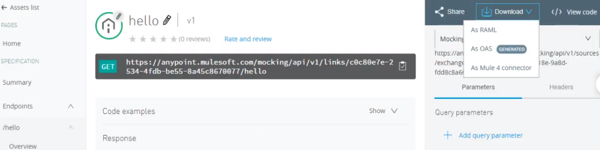
Once tested and everything looks Ok we can go ahead and publish it to Anypoint Exchange.



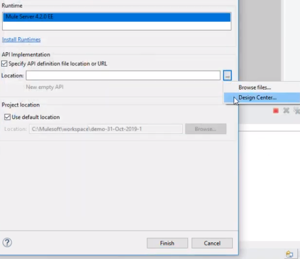
Now we can see the same in the exchange and start using it.



We can download it as RAML by downloading it from Exchange and use it your local as well:

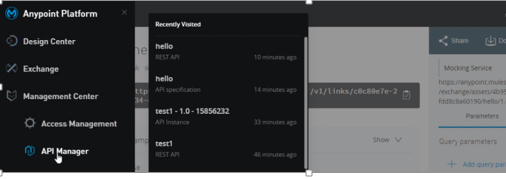


Now we can consume this RAML file from either using the local path where we stored the RAML File or form the design center like below:

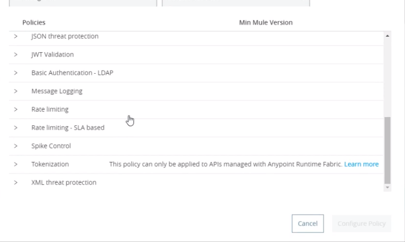


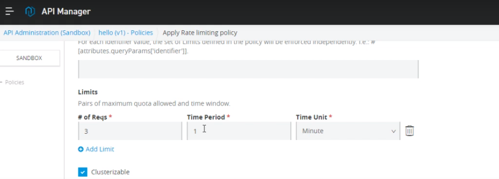
Now we can run the application in Anypoint studio.

Now we can manage this in API Manager in ‘Anypoint Platform’

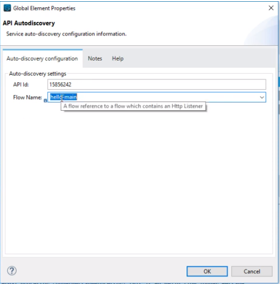


Now we can manage our API like setting the API Policy by setting the rate limit etc.





Now go the ‘Global elements’ in your Mule runtime and configure the apiid from the apimanager to the mule run tike apirouterkit



Through API Auto-Discovery method we can locate our API from the API Manager and then can link it to our runtime Mule application.

Through API Manager we can do below 2 things:

* Design Governance 🡪 like setting resources,methods etc
* Runtime Governance 🡪 like setting and managing policies like rats,ip blacklisting,white listing etc

**16) Mulesoft Choice Router:**

Refer to below link:

<https://docs.mulesoft.com/mule-runtime/4.2/choice-router-concept>

Based on the content in the payload we can make a conditional routing called ‘Choice Router’ which can call s specific flow based on the condition defined on the choice Router else the default choice floe will be used.

Example : If there are 2 flights companies like Delta and American airlines merged together based on the payload content you can route the incoming traffic to either delta system or AA system based on choice router conditions.

**17) Mulesoft Cluster setup:**

Download Mulesoft runtime client from below link:

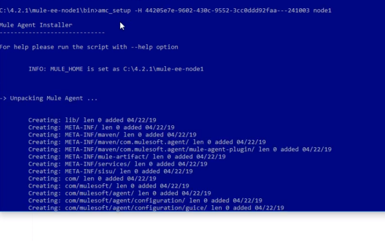
<https://www.mulesoft.com/lp/dl/mule-esb-enterprise>

You can copy the same and keep them as 2 separate nodes for a cluster.

Go to Anypoint Studio’s Runtime Manager and click Add servers.

Once you copied the agentid from add new server, go the node1 bin in server 1 and run the agent and do the same in the node2 as well.

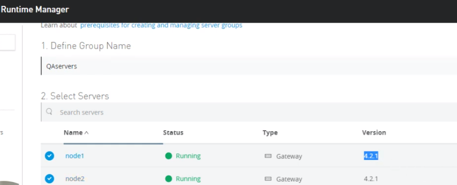
Invoke the mule in the 2 servers with 2 diff port#



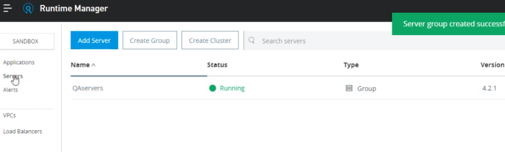
After Agent is installed, invoke Mule in that server node1



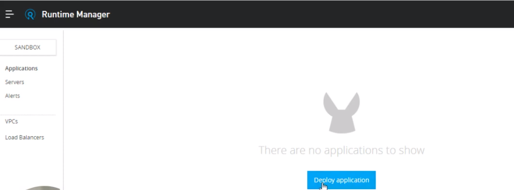
Now we can see the node1 and node2 are in running status.



Now we can see sever is running, earlier it was not running.

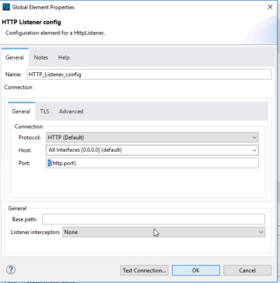


Now let’s deploy our application in this server:



Now we can create a mule project in our anypoint studio and then we can deploy that project’s jar file in this cluster:

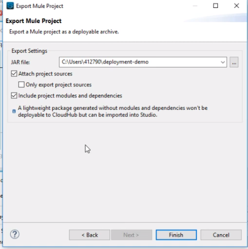
Instead of hardcoding the port#, we need to pull it from the config file.



By giving port# variable above

Now let’s export the mule project as a jar file by right clicking the mule project and choose export.

Now deploy this in the mule Runtimemnagaer inside the Qaservers.



Now instead of Groups, you can deploy the same in Cluster as well.

Click Create cluster, then add the node1 and node2 inside that and it will create that new Cluster.

**SOAP Vs REST**

The Business Use case,Constraints,Requirements,security,network bandwidth and environment decides whether we need to use SOAP or REST as the Webservice

