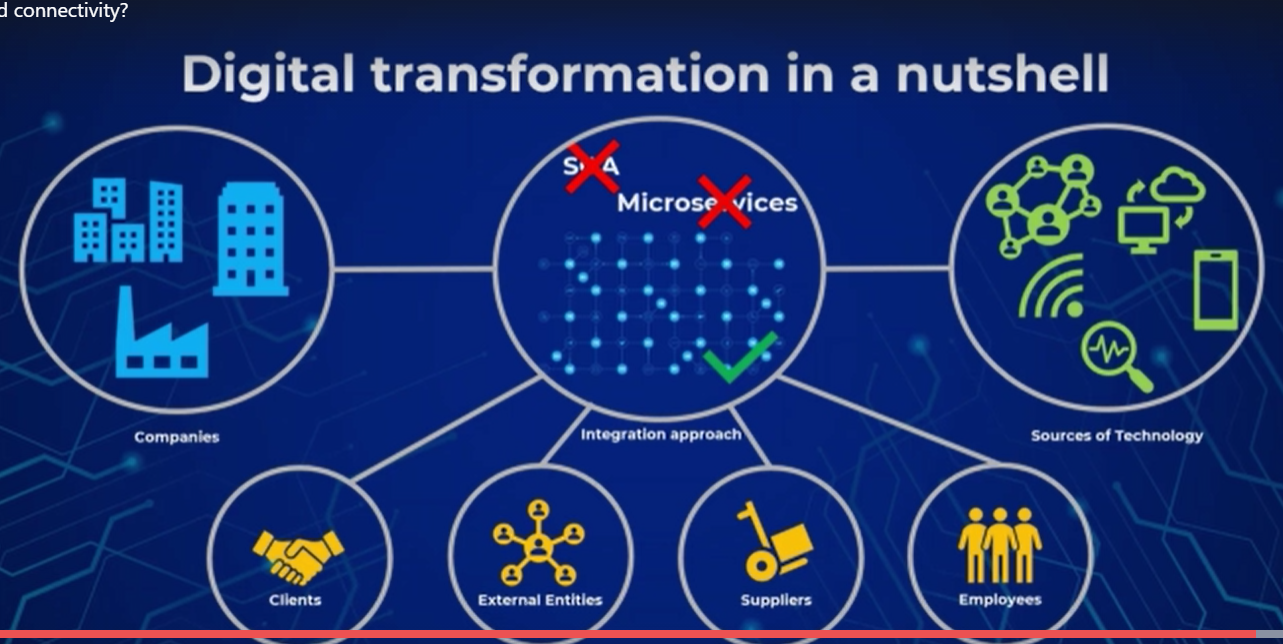
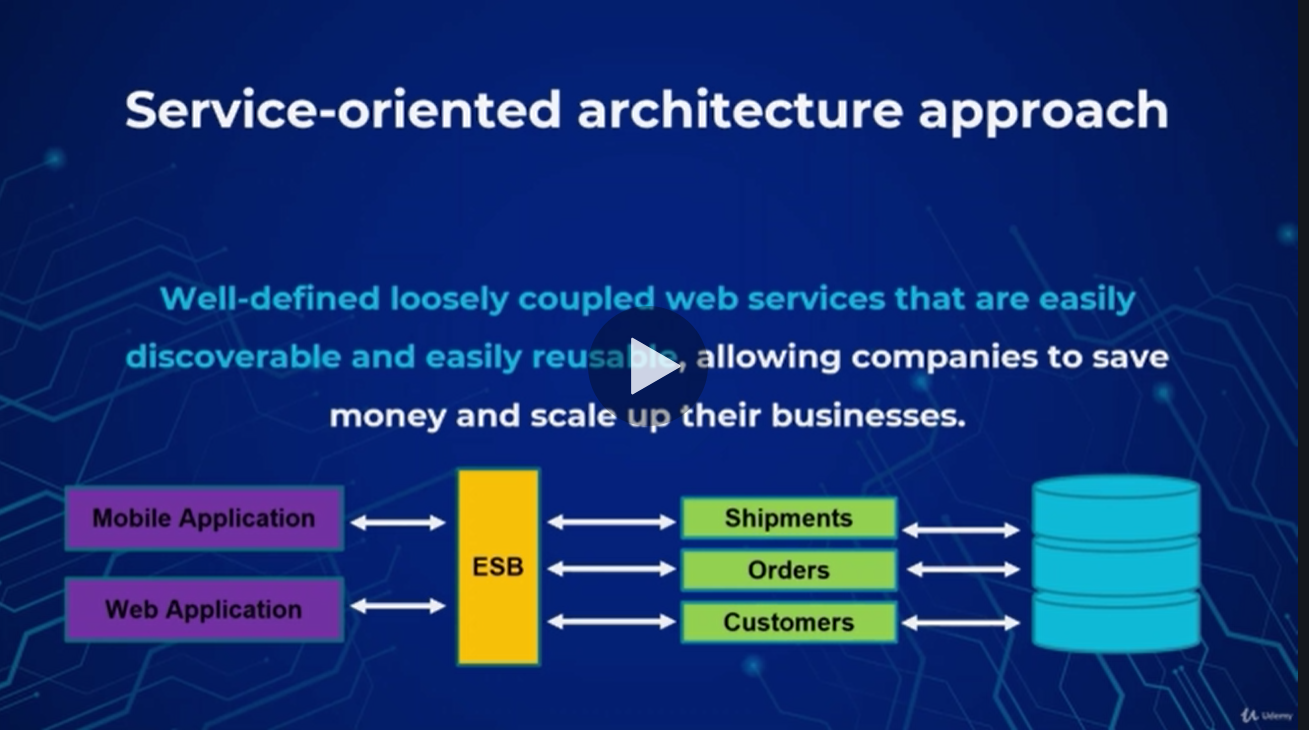
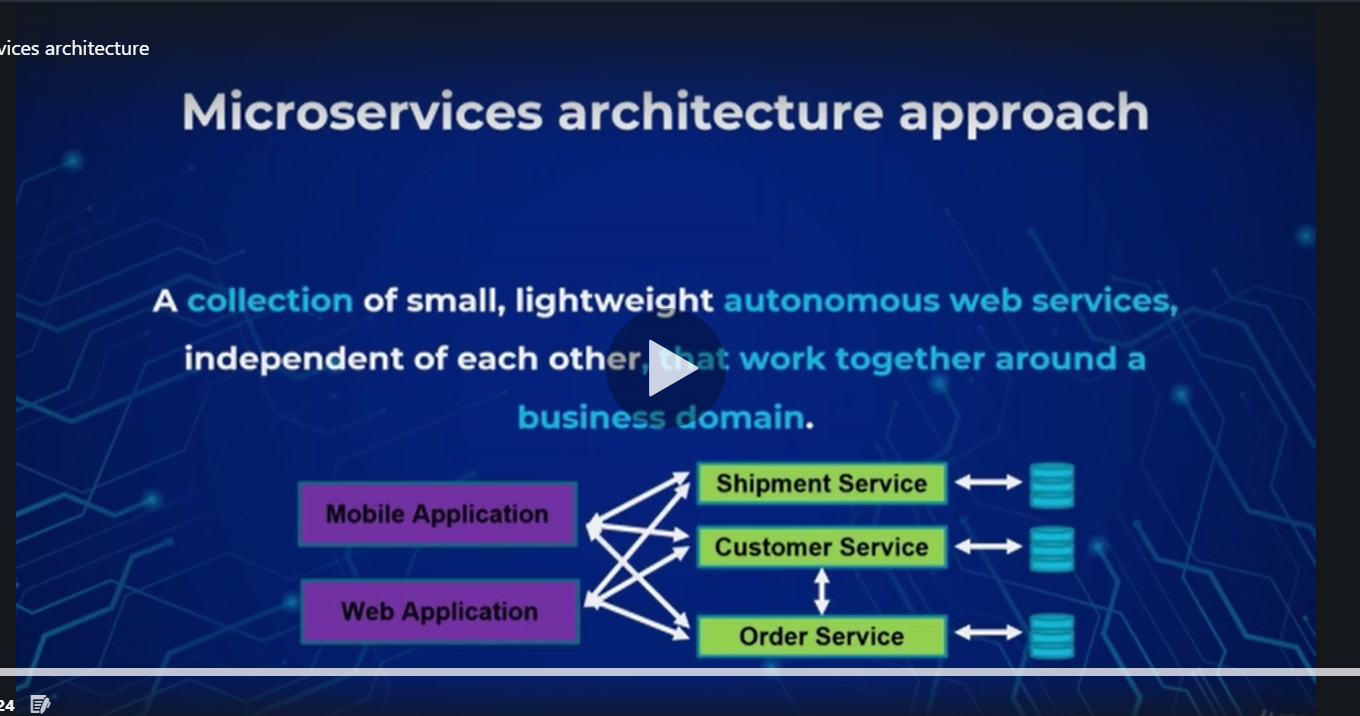




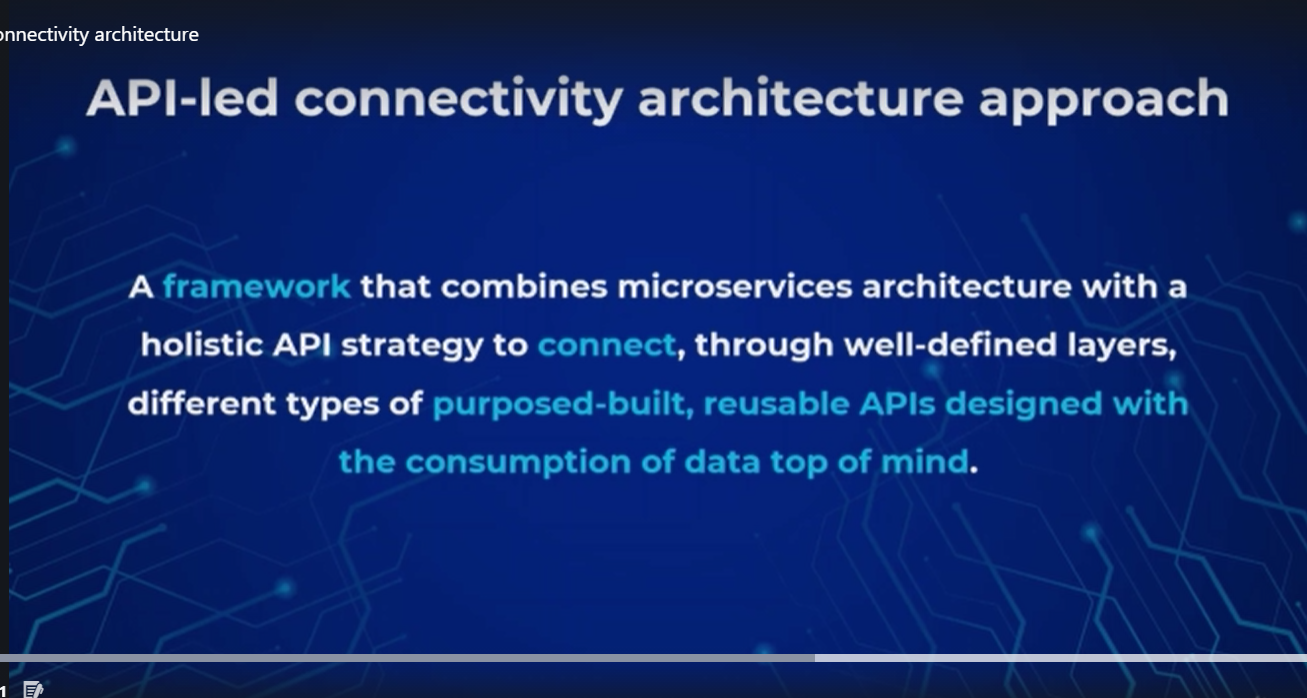
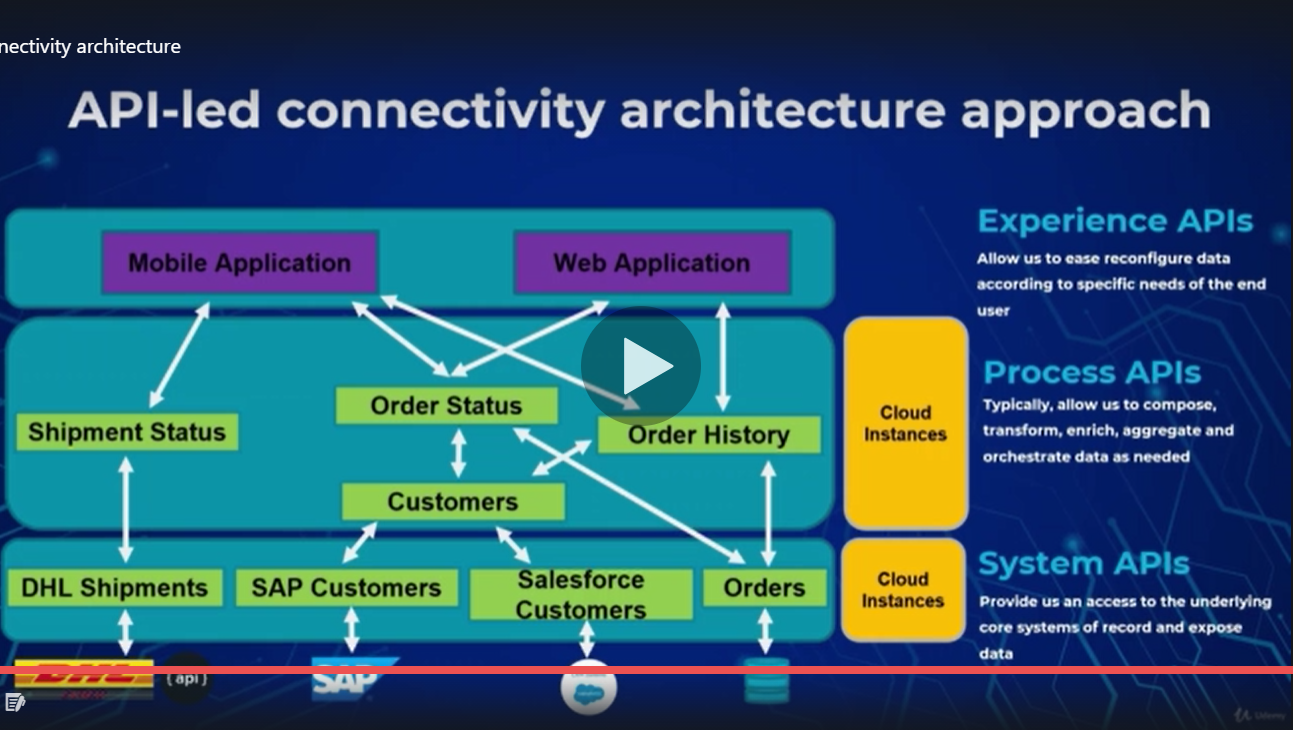
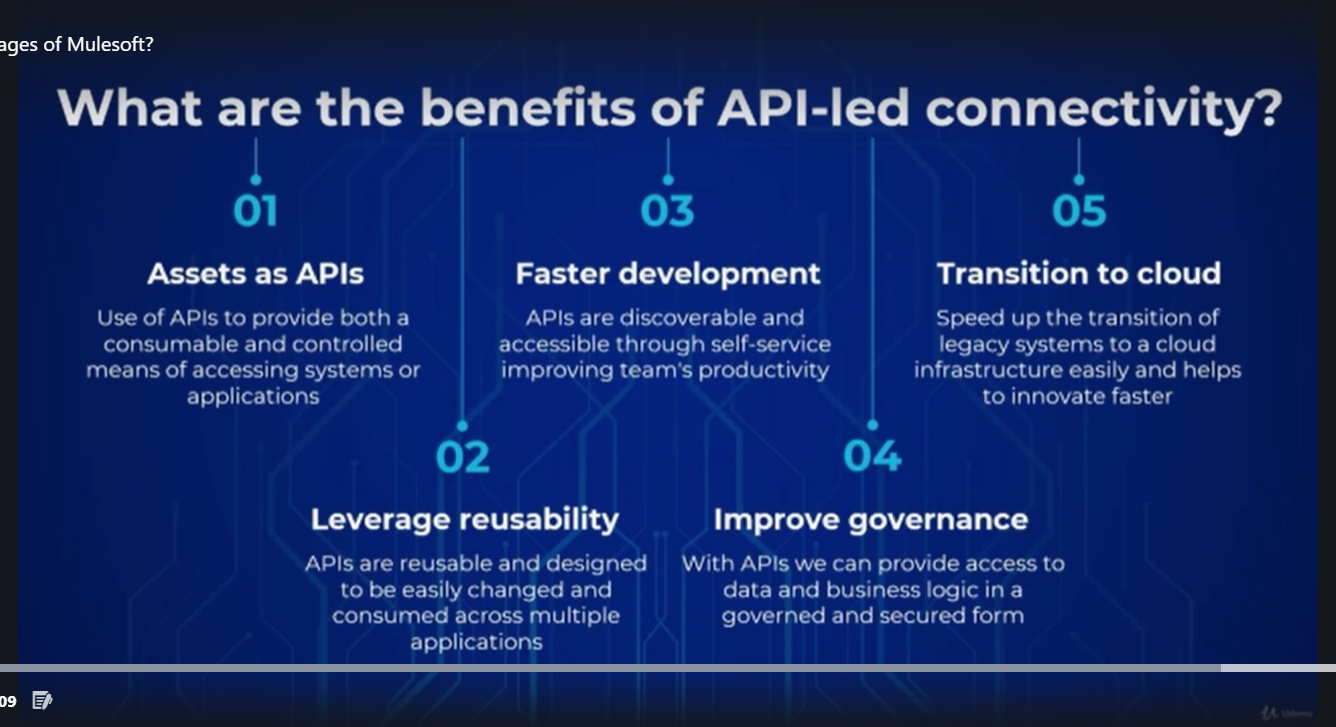
In order to understand Api-Led connectivity approach,let say as innovations are happening in digital world like SAAS,IOT,AI ML TECHNOLGIES the digital data n system were also got innovated and with the traditional integration approach like SOA,Microservice were not fulfilling the new digital requirement of business so that’s y Api led approach came into existence. Which extends soa principles and microservice approach



Main focus was on discovery n consumption and to use soa org needs to have huge investment + all serviceswere deployed onto ESB so there was scalability issue also in SOA.



It is recent approach to overcome soa disadvantages like scalability,project delivery time reusability etc. which divided the large esb into into isolated or independent component but here also we had a issue like it provides complexity if we have no of small services created which can result into redundant work .

Tools required :-

1)Java JDK 8 (to run anypoint studio) from oracle ---required becoz anypoint studiois java based app n mule apps uses maven to manage the project so that’s y.install it n then copy the jdk folder path n create system variable in environment variable like %JAVA\_HOME%= (GIVE DIRECTORY PATH)

N THEN IN PATH SYSTEM variable add %JAVA\_HOME%/bin

To check if installed correctly:-

Cmd open kro->java -version n then echo %JAVA\_HOME% N echo %Path%

2)Maven ( not required) download from maven apache site .zip file n extract it to ur local then decalre two variables in System variables in environment variables like %M2\_Home% and %Maven\_Home% =maven directory path n include the these variables in Path system variabl like %Maven\_Home%/bin and %M2\_Home%/bin

To check if installed correctly:-

Cmd open kro->mvn -version n then echo %M2\_HOME% N echo %Path%

N echo %MAVEN\_HOME%

3)Git(not required) download it fromgit site once installed open in git bash n then follow with below cmds

Pwd,cd desktop/,mkdir mulesoft\_course,cd mulesoft\_course/,git init

Once we done with above cmd we have to configure the server where our files will be stored

For that go to github create account.Add repository with somename

Go back to local mulesoft\_course directory create a readmr.md file then on git bash follow command

Git status,git add .,git commit -m ‘create readme.md file’,git status,

To push those new files to github server, copy the command from recently created repository Command like git remote add origin ‘path of git repository’ and then git status,git push -u origin master,git status,ls n then check on github website whether you got the readme.md file on github server

4)postman (important to test our code as well as others code it is an http client interface used for making http request)

5)MariaDb(not imp but to use db connector) is opensource mysql client with cli as well as gui called heidisql

6)(Anypoint studio:important develops mule apps for choosing workspace choose mulesoft course created earlier so that all mule app can be on gitlab as well.It is eclipse based Integration Development TOOL.Used for building and testing Mule app.

Basic dw functions like

In transform msg we can use dw basic fun like

%dw 2.0

**output** **application/json**

var surya={

newstring:"surya"++"kant",

sum:sum([1,2,4]),

avg:avg([122,23,20]),

iseven:isEven(14),

upper:upper("suryakant"),

lower:lower("ASDF"),

}

---

surya

To read json file from dw

Use transform msg component n

Add below

%dw 2.0

**output** **application/dw**

output application/dw or application/json or application/java or application/xml

---

readUrl("classpath://surya.json")

To add map function which takes an input as array n returns as an aaray we mapping elements from an array into object.]

%dw 2.0

**output** **application/dw**

var customearray=["mulesoft","raj","tiger"]

---

customearray map ((item, index) ->

index:item ++ "\_" ++ index

)

To pluck objects from an object into an array

%dw 2.0

**var** customobject= {

"0": "mulesoft",

"1": "surya"

}

**output** **application/json**

---

customobject pluck ((value, key, index) ->

value

)

Filter an array and return only matching values

%dw 2.0

**output** **application/json**

**var** customarray=[1,2,3,4,5,6]

---

{

collection:customarray filter ((item, index) ->

item<4

)

}

Filter an object n return matching objects

%dw 2.0

**output** **application/json**

**var** customobj={

"0":"mulesoft",

"1":"suryakant"

}

---

customobj filterObject ((value, key, index) ->

(key contains 1) **and** sizeOf(value)<15

)

Selectors over an object

1)Single value selector:selects only one value that matches

%dw 2.0

**output** **application/json**

**var** customobj={

Peoples:

{

people:

{

name:"surya"

},

people:

{

name:"raj"

},

people:

{

name:"shant"

},

}

}

---

customobj.Peoples.people//returns only one first matching value

2)multi value selector:return multiople values that are matching represented by \*

%dw 2.0

**output** **application/json**

**var** customobj={

Peoples:

{

people:

{

name:"surya"

},

people:

{

name:"raj"

},

people:

{

name:"shant"

},

}

}

---

customobj.Peoples.\*people

3)descesdent value selector:retrive all values matching with key name here we use to match key name with .. operator

%dw 2.0

**output** **application/json**

**var** customobj={

Peoples:

{

people:

{

name:"surya",

natak:{

name:1

}

},

people:

{

name:"raj",natak:{

name:2

}

},

people:

{

name:"shant",

natak:{

name:3

}

},

}

}

---

customobj.Peoples..name

4)Key value selector:retrieves key n value symbol &

%dw 2.0

**output** **application/json**

**var** customobj={

Peoples:

{

people:

{

name:"surya"

},

people:

{

name:"raj"

},

people:

{

name:"shant"

},

}

}

---

customobj.Peoples.\*people.&name

5)Index value selector:-selects the objects from index perspective starts with 0

%dw 2.0

**output** **application/json**

**var** customobj={

Peoples:

{

people:

{

name:"surya"

},

people:

{

name:"raj"

},

people:

{

name:"shant"

},

}

}

---

customobj.Peoples[2]

Selectors over strings

1)index selector:returns specific leeter from string

%dw 2.0

**output** **application/json**

**var** customstring="suryakant"

---

customstring[4]

2)range selector over an string

%dw 2.0

**output** **application/json**

**var** customstring="suryakant"

---

customstring[0 to 3]

%dw 2.0

**output** **application/json**

**var** customstring="suryakant"

---

customstring[6 to 3]

config file .yaml or .properties

.yaml

http:

host: ”localhost”

port: “8081”

path: “/env”

.properties

http.host=localhost

http.port=8081

http.path=/env

yaml files for diff environment

uat.yaml file

message:”uat”

prod.yaml file

message:”prod”

create common.yaml file for http coonection n create global config properties in global element of global.xml file

create another global config properties in global element of global.xml file n name it as ${environment}.yaml

create global property in global.xml for storing the environment variable as uat or prod

to log n set payload use

%dw 2.0

Output application/json

---

{envmessage:Mule::p(“message”)

}

If you run as it is then the value you congigured for environment value in global property that .yaml file will be used n will run

If you want then you can add argument in run config properties like

-denvironment=prod to run prod yaml file

Transit queue n persistent queue

Fast but not reliable in case of system crashes slow but reliable

Data Publish on Transient queue n if finds queue listener of transient then msg will be showed

Data Publish on persistent queue n if finds queue listener of persistent then msg will be showed

But if Data Publish on Transient queue n if it doesn’t finds queue listener of transient instead it finds listener of persistent then msg will be not be showed msg will get deleted from transient queue (we can modify the listener property to transient to check that we are not receiving the msg)

But if Data Publish on persistent queue n if it doesn’t finds queue listener of persistent instead it finds listener of transient then msg will be not be showed msg but the message still be there in persistent queue so once persistent queue is there it shows the msg we can check this by modifying the property of vm listener properties to persistent queue

Steps to create Api:-

Anypoint platform->Creat API Specification->Publish the API to exchange it will get published into private exchange n for taking the feedback from internal as well as external developers we need to share the api either to internal developer (using share option in share panel) or external developers (using public in share panel via public portal)

Once 1st phase completes that is approved api specification

Next is to implement the api specification to build n test API

For that we need to use anypoint studio

How to import api spec in anypoint studio is go to mule app right click then go to anypoint platform then go to configure anypoint platform with uname n pwd of urs uske bad once account added go to ur mule app anypoint platform n select import from anypoint platform design center.then select api spec name to be used for creating api n wait for some time after that api kit router will create an interface based on api specification(yu can change verion of component in pom.xml n do refresh of the mule app to take effect.)

Makethe implemenentation.xml file to add logic for each resource n once it is done n teted properly then we have third n final phase that is api deployment n governance.

Until the api is in anypoint studio,we use local runtime engine to run our mule app but what if we want to run the mule app on upper level environment like uat,prod.we have to deploy the mule app on one of the available mule runtime instance like mulesoft’s cloudhub cloud based aws managed instance or on premise i.e. customer hosted runtime environment or runtime facric.

To deploy the mule app,

Anypoint studio>Right click on the mule app want to deploy>select Anypoint platform>select Deploy to cloudhub>choose the runtime engine worker size or worker no n deply the app. Deployment time depends on mule project heaviness.We can check the progress in Runtime manager blade by clicking on the mule app name n selecting logs etc.once it is deployed we can test it through postman by using eg:- <http://Mule_app_cloudhub> <url:80/api/artist>

Please make a note that mule app is deployed in aws vm called worker where only one app can run.

To avoid this we can use on premise mule runtime

To use it we can downliad the runtime from <https://mulesoft.com/lp/dl/mule-esb-enterprise>. Download .zip file 300 mb file.Once downloaded extract the file and to manage that on Runtime manager of Anypoint platform go to Anypoint Platform>Run time manager>Server>Add server>Add unique name >copy command n then go to local file system where u have extracted the files n go inside the bin folder n then in the location path type cmd so it will open command prompt in same folder n then paste the command copied and in that modify the command like in starting we have ./ so remove that and add .bat for amc\_setup.bat and give enter. It will configutre the agent with Runtime manager and then type mule.bat install in next command in cmd to install the service of mule runtime engine on premise .open services n look for Mule enterprise engine it should be installed. N to start the mule runtime,go to cmd and type mule.bat start. Go to services n check Mule enterprise engine I running or not.

Once done then you can deploy any mule app on your on premise runtime by doing following

Goto runtime manager>Application>Deploy application>while selecting deployment target select on premise one n not clodhub n then attach jar file (which you can export from anypoint studio n then save it as .jar file) click on deploy.once deployed in your local file system in apps folder you can see the mule app files are located n in logs in logs folder