LAB- Consuming Restful Services

In this lab you will understand

a) How to consume a restful webservice

b) How to extract uri params from http request and pass uri params to a restful service

STEP 1

1)Create a project with name consumerest

Execute **rest.bat** given to you in lab-docs folder . This will start a rest service which starts listening on port **7070** and base path is **/rest**

NOTE: We need JDK 11 to run this rest service with out any problem. If u have any other version also, it is a problem. Even mule server and studio needs the same version. So, make sure that u have JDK 11 installed.

2)

Give requests to http://localhost:7070/rest/products and observe that you get all products in Json format

Give requests to **http://localhost:7070/rest/products/Mac** will give give you products whose name contains Mac. Try the same for

http://localhost:7070/rest/products/Hp and **http://localhost:7070/rest/products/Moto**

Open POSTMAN and give a POST request to http://localhost:7070/rest/products and pass the following product Json in the body :

```
{
    "name": "Sony   VAIO",
    "description": "SONY Laptop ",
    "originalPrice": 4000,
    "offer": {
      "offerPrice": 2000,
      "offerValidUntil": 1466098094993
    },
    "brandName": "Sony",
    "images": [
      "image15.jpeg",
      "image16.jpeg",
      "image17.jpeg"
    ]
  }
```

Observe that the product   is created successfully.

Test by giving GET request to http://localhost:7070/rest/products. You should see 7 products

5) Now we want to consume this rest service using mule

Create a new configuration file with name   consumerest.xml in   src/main/app

Configure a flow with   http listener at port 8081 and path /products

Rename this flow as “mainflow”

Drag Http Request Component from   HttpModule and drop it   outside of existing flow so that a new flow will be created with “Request” component in it.

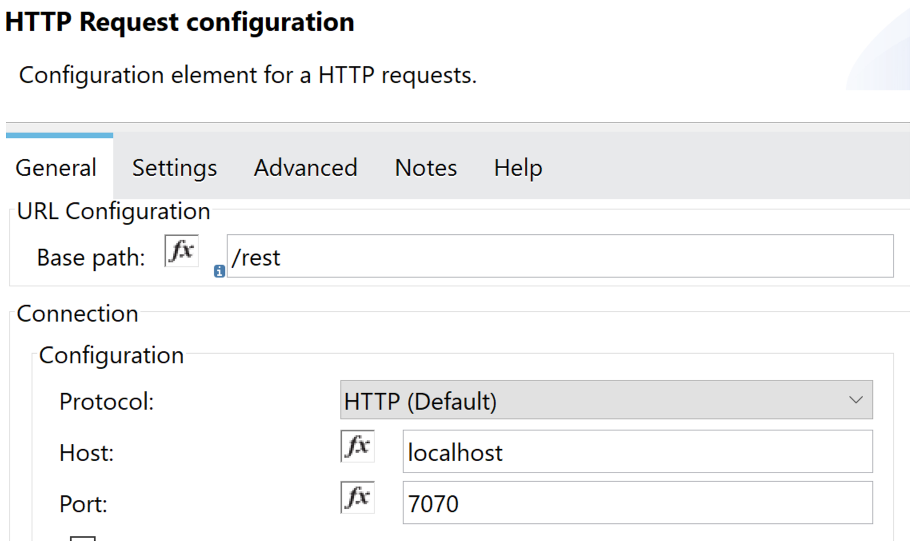
This flow doesnot contain any component in the source part. Such a flow is called “Private Flow”.

Now there should 2 flows in same xml.

Click on the flow, Rename the flow with “Http Request” component as “getallproducts”

Configure the Http Request connector configuration as shown below :

Make sure u give the base path as /rest



HTTP Request configuration

Configuration element for a HTTP requests.

General | Settings | Advanced | Notes | Help

URL Configuration

Base path: /rest

Connection

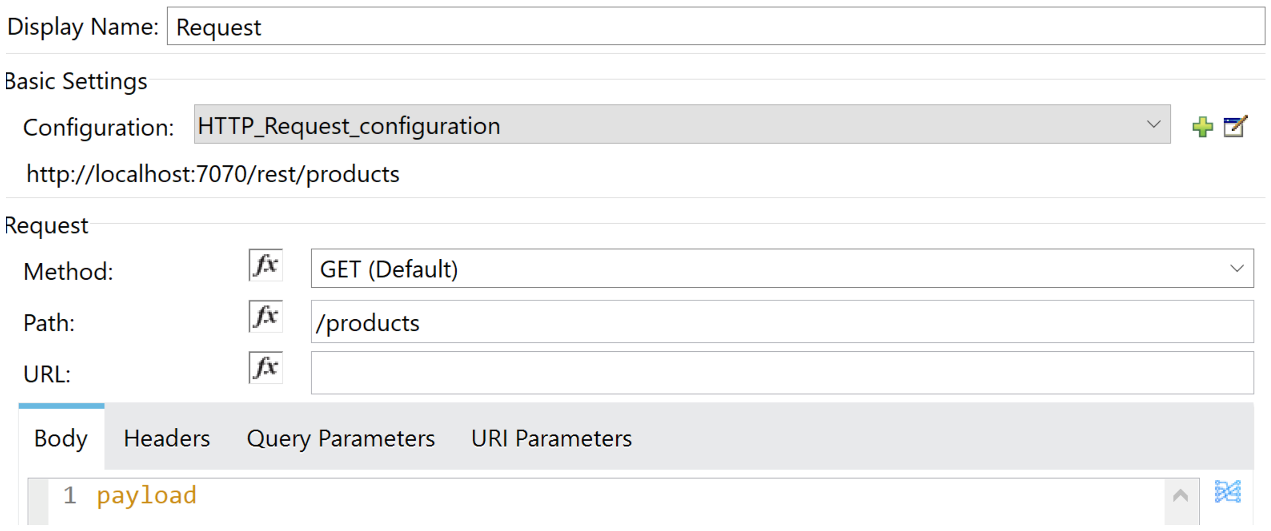
Configuration

Protocol: HTTP (Default)

Host: localhost

Port: 7070

Configure the path of http outbound endpoint as /products and Method as GET



Display Name: Request

Basic Settings

Configuration: HTTP_Request_configuration

http://localhost:7070/rest/products

Request

Method: GET (Default)

Path: /products

URL:

Body | Headers | Query Parameters | URI Parameters

```
payload
```

Now from the mainflow use FlowReference to refer to “getallproducts” flow.

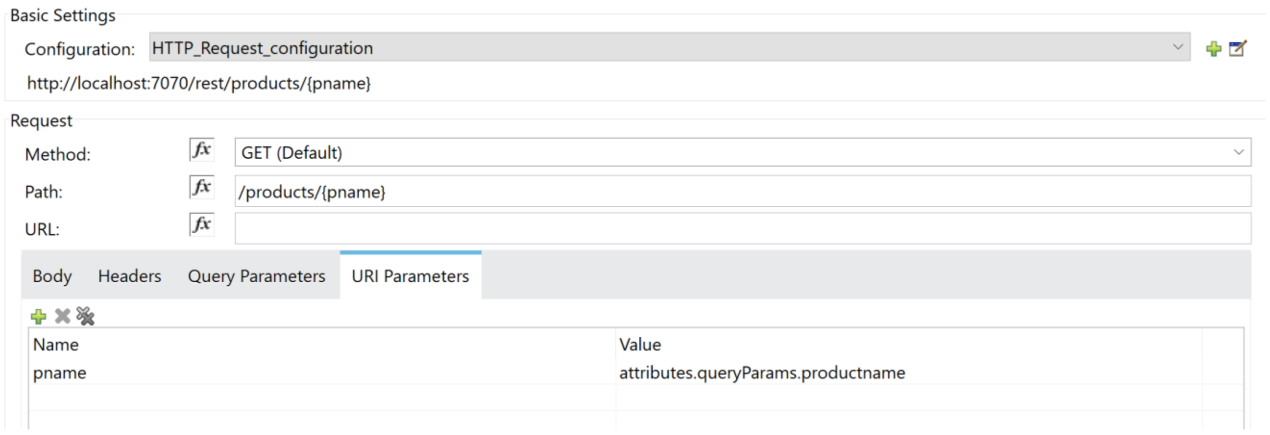
Now run the application and give a request to http://localhost:8081/products. You should see all the products

6) Drag   another “Http Request” endpoint into a new flow in the same xml.
Rename the flow as “getproductsbyname”.

Let this Http Requestor also refer to same global “Http Request Configuration” as we want to consume same rest api

Now modify the path of http Request endpoint such that it will get the products whose name matches the name passed as **query** parameter **"productname"**

Hint : configure the path as /products/{pname}] and set URI parameters as below



Basic Settings

Configuration: HTTP_Request_configuration

http://localhost:7070/rest/products/{pname}

Request

Method: GET (Default)

Path: /products/{pname}

URL:

Name	Value
pname	attributes.queryParams.productname

Now we want to link 2 existing flows (getallproductsflow and getproductsbynameflow) with main flow and conditionally route request to getproductsbyname flow if there is a query parameter with name "productname"

if there is no queryparameter with name productname, we want to route to getallproducts flow.

if there is queryparameter with name productname, we want to route to getproductsbyname flow.

Drag a choice router after “Http Listener”.

Drag one “Flow Reference” into When and make it to reference getallproductsflow. Change the display name of flow reference as “getallproductsflow”

Drag one more “Flow Reference” just below “when” and make it to reference getproductsbynameflow.

Change the display name of flow reference as “getproductsbynameflow”

Now Click the first “when” and give the condition as

`#[ attributes.queryParams.productname == null]`

Click on the second “when ” and give the condition as
`#[ attributes.queryParams.productname != null]`

In the default block, drag a logger and configure it to log “Invalid Request”

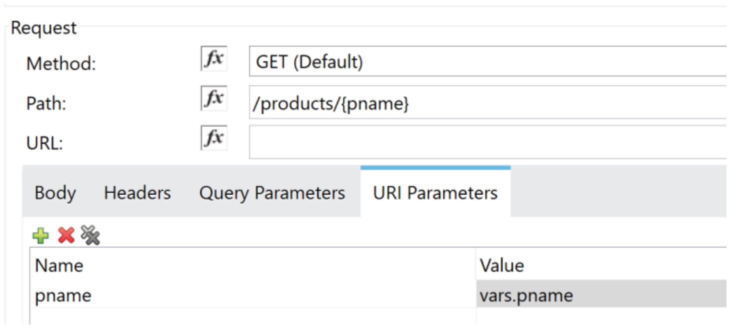
Deploy the application give request to **http://localhost:8081/products?productname=Mac** and observe that you will get all the product whose name contains Mac

If you give a request to **http://localhost:8081/products** with out passing query parameter, you should get all products

7) In getproductsbynameflow, you are expecting a http queryparameter with name productname. That means your private flow is not completely reusable or independent. You private flow is expecting that it will be called from a flow with Http Inbound endpoint. This is not a good practice.

To make this private flow reusable, we can make it to use a variable in expression instead of using attributes.queryParams.productname

Now, Click on “Http Request ” component in getproductsbynameflow, click on uriparameters tab and modify the value for pname as vars.pname as shown below:



Request

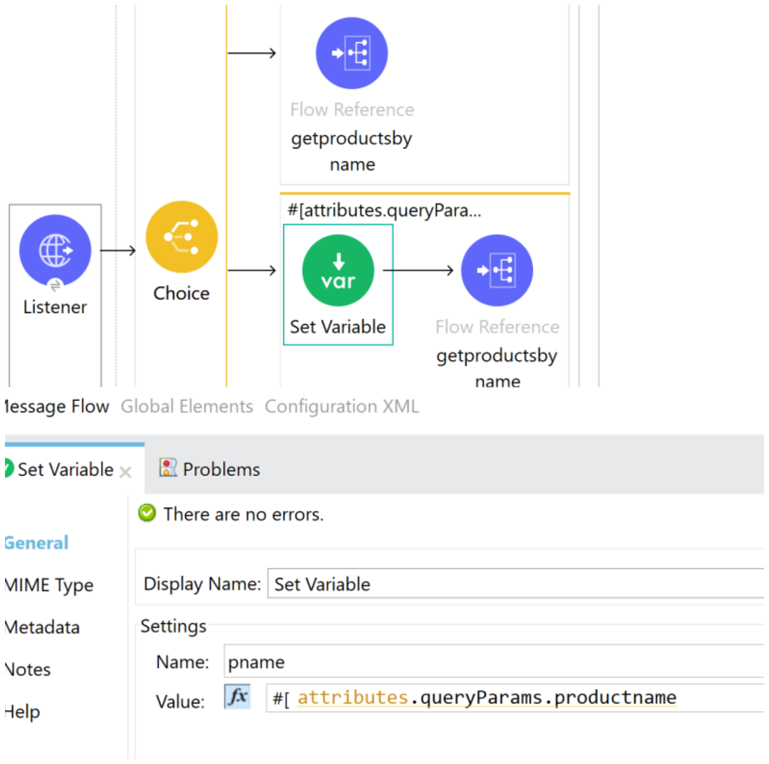
Method: GET (Default)

Path: /products/{pname}

URL:

Name	Value
pname	vars.pname

Now Drag a “Set variable” transformer before flow reference to getproductsbynameflow inside second “when ” of the main flow configure a variable with name "pname" value as /#[attributes.queryParams.productname] as shown below:



There are no errors.

Display Name: Set Variable

Settings

Name: pname

Value: #[attributes.queryParams.productname

Deploy the application give request to **http://localhost:8081/products?productname=Mac** and observe that you will get all the product whose name contains Mac

9)

Don’t do the below step. This is just for understanding how to extract uri params in the request.

If the path of “Http Listener” endpoint is as /products/{productname} then {productname} is the uri parameter

How do you access this uriParam ?

[Hint : #[`attributes.uriParams.productname`]]

This is the end of the Exercise