Payara micro with two application showing data and then add jwt

You Need 4 files Initially

* payara-micro-5.2022.2.jar
* mysql-connector-java-5.1.47-bin.jar
* jwtenizr.jar
* domain.xml

1>

Create New Project (MSAApp) -> Java with maven -> project from Archetype

Select : jakartaee8-payara-microprofile-archetype

Dockerhub:value ok

Do changes in pom.xml

<dependency>

<groupId>org.eclipse.microprofile</groupId>

<artifactId>microprofile</artifactId>

<version>3.2</version>

<type>pom</type>

<scope>provided</scope>

</dependency>

<maven.compiler.source>1.8</maven.compiler.source>

<maven.compiler.target>1.8</maven.compiler.target>

2>

* Create persistence unit

(in tager error go to properties->run->select payara server)

* Create Entity Classes from database. (Model)
* Create JAVA class named DataModel for business logic with entityManager. (Beans)

public class DataModel {

@PersistenceContext(unitName = "punit")

EntityManager em;

public Collection<Students> students()

{

return em.createNamedQuery("Students.findAll").getResultList();

}

}

3).

* Make Changes in Example Service means Add your rest resources in that file

package com.mycompany.demo.service;

import Entity.Students;

import Model.DataModel;

import java.util.Collection;

import javax.annotation.security.RolesAllowed;

import javax.inject.Inject;

import javax.ws.rs.GET;

import javax.ws.rs.Path;

@Path("/example")

public class ExampleService {

@Inject DataModel dm;

@Path("/students")

// @RolesAllowed("admin")

@RolesAllowed({"admin","user"})

@GET

// @Produces(MediaType.APPLICATION\_JSON)

public Collection<Students> students() {

return dm.students();

}

}

4).

Create another app client like u create uper app

Create New Project (MSAClirnt) -> Java with maven -> project from Archetype

Select : jakartaee8-payara-microprofile-archetype

Dockerhub:value ok

Do changes in pom.xml

<dependency>

<groupId>org.eclipse.microprofile</groupId>

<artifactId>microprofile</artifactId>

<version>3.2</version>

<type>pom</type>

<scope>provided</scope>

</dependency>

<maven.compiler.source>1.8</maven.compiler.source>

<maven.compiler.target>1.8</maven.compiler.target>

5) Paste whole folder of entity in this app to have all table entities

6).Create MSAClient INTERFACE to call the other app api s

package Client;

import Entity.Students;

import java.util.Collection;

import javax.annotation.security.RolesAllowed;

import javax.enterprise.context.ApplicationScoped;

import javax.ws.rs.GET;

import javax.ws.rs.Path;

import javax.ws.rs.Produces;

import javax.ws.rs.core.MediaType;

import org.eclipse.microprofile.config.Config;

import org.eclipse.microprofile.config.ConfigProvider;

import org.eclipse.microprofile.rest.client.annotation.ClientHeaderParam;

import org.eclipse.microprofile.rest.client.inject.RegisterRestClient;

/\*\*

\*

\* @author radhika

\*/

//@RegisterRestClient(configKey = "myclient")

@RegisterRestClient(baseUri = "http://localhost:8085/demo/rest/example")

@ApplicationScoped

//@Named

public interface MSAClient {

@ClientHeaderParam(name="authorization",value="{generateJWTToken}")

@GET

@Path("/students")

@RolesAllowed("admin")

@Produces(MediaType.APPLICATION\_JSON)

public Collection<Students> students();

default String generateJWTToken()

{

Config config = ConfigProvider.getConfig();

String jwt = config.getValue("jwt-string", String.class);

String authtoken = "Bearer "+jwt;

return authtoken;

}

}

7).add MSAApp url into MSACLient App in microprofile-config.properties on top

myclient/mp-rest/url=http://localhost:8085/demo/rest

* Create servlet(do some changes like)

public class TestClientServlet extends HttpServlet {

@Inject @RestClient MSAClient msacl;

Collection<Students> docApps;

* Add this inside body tag

try{

docApps=msacl.students();

out.println("<h1>Available Doctor's</h1><br/><br/><br/>");

out.println("<table><tr><td>DocName</td><td>Specialist</td><td>Address</td><td>FromTime</td><td>ToTime</td></tr>");

for(Students dapp:docApps){

out.println("<tr><td>"+dapp.getName()+"</td><td>"+dapp.getEmail()+"</td><td>"+dapp.getCourse()+"</td></tr>");

}

out.println("<table>");

}catch(Exception e){

out.println("Error Error Error");

}

—-----------------------------------------------------------------------------------ADD JWT

Run jwtenizr.jar by typing command java -jar jwtenizr.jar in CMD.

Change jwt-token.json and update by running again

(this will create 4 different files)

* Add 2 lines in bootstrap file of MSAApp

@LoginConfig(authMethod="MP-JWT")

@DeclareRoles({"admin","user"})

* IN example service assign role like

@Inject DataModel dm;

@GET

@Path("student")

@RolesAllowed("admin")

public Collection<Students> getStudent()

{

return dm.getStudents();

}

* In other sources/src/META-INF/microprofile-config add all fields of newly generated microprofile-config file(mp.jwt.verify.issuar and mp.jwt.verify.publickey)

2)CLIENT APP

Create JAVA INTERFACE inside client app named MSAClient inside that assign role and generate token

@RegisterRestClient(baseUri = "http://localhost:8085/ProducerDB/rest/example")

public interface ApiInterface {

@ClientHeaderParam(name="authorization",value="{generateJWTToken}")

@GET

@Path("student")

@RolesAllowed("admin")

public Collection<Students> getStudent();

default String generateJWTToken()

{

Config config = ConfigProvider.getConfig();

String jwt = config.getValue("jwt-string", String.class);

String authtoken = "Bearer "+jwt;

return authtoken;

}

}

* In servlet

public class app extends HttpServlet {

@Inject @RestClient ApiInterface a;

* Inside body tag

try

{

Collection<Students> students = a.getStudent();

for(Students s:students)

{

out.println(s.getName());

}

}

catch(Exception e)

{

out.println("Something went wrong");

out.println("<br>");

out.println(e);

}

* Inside microprofile-config addd (token file token)

jwt-string=eyJraWQiOiJqd3Qua2V5IiwidHlwIjoiSldUIiwiYWxnIjoi..

—-------------------------------------------------extra with parameter—---------------------------------------------

MSAAPP

* Producer BEan Class(Java class) method

public Collection<DoctorAppointment> getDoctorAvailability(String Specialization){

doctorAppointment=(Collection<DoctorAppointment>)em.createNamedQuery("DoctorAppointment.findAllBySpecialization").setParameter("specialization",Specialization ).getResultList();

return doctorAppointment;

}

* Exampleservice

@Path("/doctor\_avalilability/{specialization}")

//@Path("/example")

public class ExampleService {

@Inject Doc docService;

@RolesAllowed("Patient")

//@Path("/doctor\_avalilability/{specialization}")

@GET

@Produces(MediaType.APPLICATION\_JSON)

public Collection<DoctorAppointment> getDoctorAvailability(@PathParam("specialization") String Specialization){

return docService.getDoctorAvailability(Specialization);

}

MSAClient

* MsaClient.java (interface)

@RegisterRestClient(configKey = "myclient")

@ApplicationScoped

//@Named

@Path("/doctor\_avalilability/{specialization}")

public interface MSAClient {

@GET

@ClientHeaderParam(name="authorization", value="{generateJWTToken}")

@Produces(MediaType.APPLICATION\_JSON)

public Collection<DoctorAppointment> getDoctorAvailability(@PathParam("specialization") String Specialization);

* Servlet

try{

docApps=msacl.getDoctorAvailability("Cardiologist");

out.println("<h1>Available Doctor's</h1><br/><br/><br/>");

out.println("<table><tr><td>DocName</td><td>Specialist</td><td>Address</td><td>FromTime</td><td>ToTime</td></tr>");

for(DoctorAppointment dapp:docApps){

out.println("<tr><td>"+dapp.getDocName()+"</td><td>"+dapp.getSpecialization()+"</td><td>"+dapp.getAddress()+"</td><td>"+dapp.getFromTime()+"</td><td>"+dapp.getToTime()+"</td></tr>");

}

out.println("<table>");

}catch(Exception e){

out.println("Error Error Error");

}

—---------------------------------------------------add—----------------------------------------------------------

@Override

public void addCategory(String CategoryName, String Description,String Image) {

Category cat=new Category(CategoryName,Description,Image);

em.persist(cat);

}

@Path("addCity/{CityName}/{StateID}")

@POST

public void addCity(@PathParam("CityName") String CityName,@PathParam("StateID") Integer StateID) {

pbl.addCity(CityName, StateID);

}