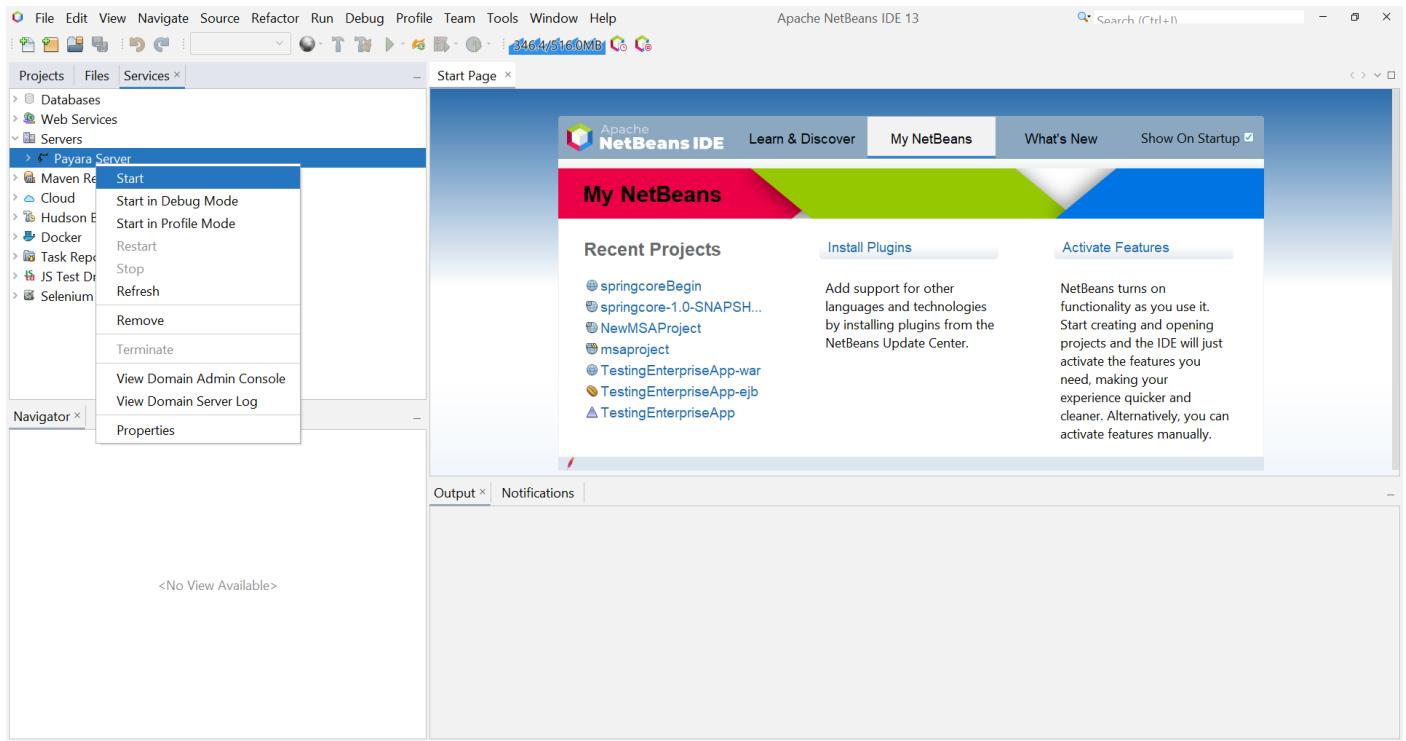


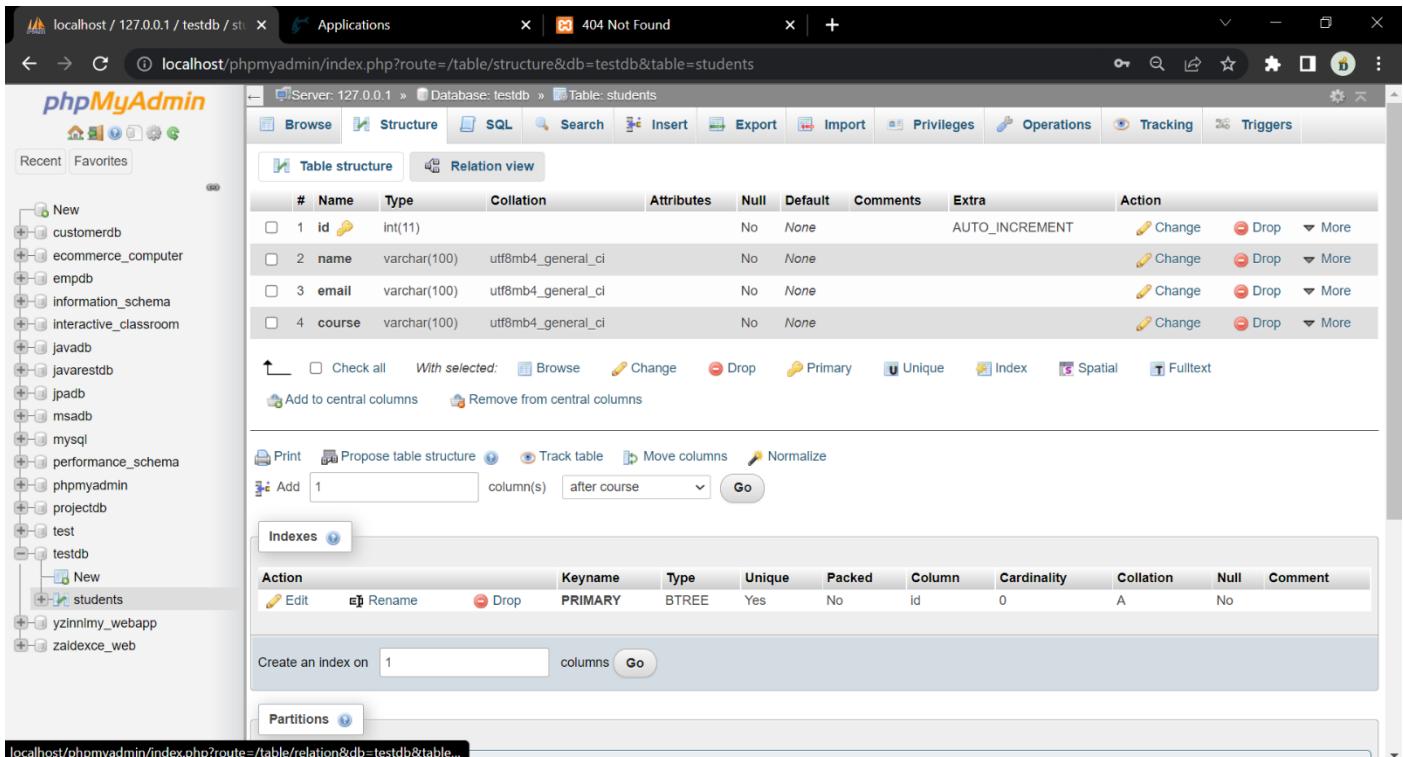
Today I'll show you How to run 2 Microservice application in Payara microprofile. Before moving on I want to you to have following files in your machine.

- 1) mysql-connector-java-5.1.23-bin or mysql-connector-java-5.1.30.jar
- 2) payara-micro-5.2022.2.jar
- 3) domain.xml
- 4) jwttenizr.jar

1) Open Netbeans 13 or later and start payara server

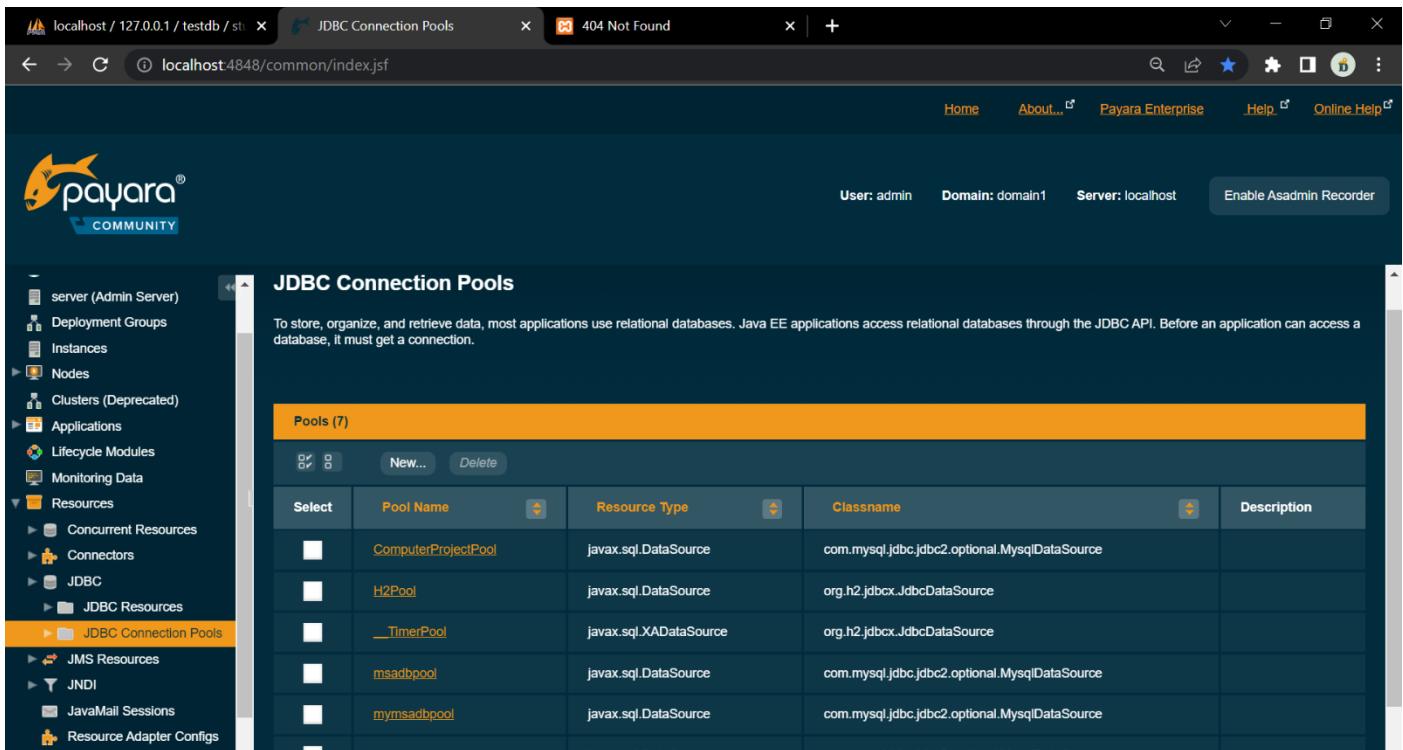


- 2) Turn on your xampp/wamp server and create one database with tables in  
<http://localhost/phpmyadmin/index.php>



The screenshot shows the phpMyAdmin interface for the 'testdb' database. On the left, a tree view lists various databases and their tables. The 'students' table under the 'testdb' database is selected. The main panel displays the 'Table structure' tab for the 'students' table. The table has four columns: 'id', 'name', 'email', and 'course'. The 'id' column is defined as int(11), 'name' as varchar(100), 'email' as varchar(100), and 'course' as varchar(100). The 'Collation' for all columns is utf8mb4\_general\_ci. The 'Attributes' column shows 'No' for all columns. The 'Default' column shows 'None' for all columns. The 'Comments' column is empty. The 'Extra' column shows 'AUTO\_INCREMENT' for the 'id' column. Action buttons for each column include 'Change', 'Drop', and 'More'. Below the table structure, there are buttons for 'Check all', 'With selected:', 'Browse', 'Change', 'Drop', 'Primary', 'Unique', 'Index', 'Spatial', and 'Fulltext'. There are also buttons for 'Add to central columns' and 'Remove from central columns'. Below the table structure, there are buttons for 'Print', 'Propose table structure', 'Track table', 'Move columns', and 'Normalize'. A search bar allows adding 1 or more columns after 'course'. Below the indexes section, there is a button for 'Create an index on 1 columns'. The 'Indexes' section shows one primary index named 'PRIMARY' of type BTREE with 'Keyname' set to 'PRIMARY', 'Type' set to 'BTREE', 'Unique' set to 'Yes', 'Packed' set to 'No', 'Column' set to 'id', 'Cardinality' set to '0', 'Collation' set to 'A', and 'Null' set to 'No'. The 'Partitions' section is currently empty.

- 3) I've created database with named testdb and created one table named students. Now add Url  
<http://localhost:4848/common/index.jsf> in your browser and Go to jdbc Connection Pool



The screenshot shows the Payara Admin Console interface. The top navigation bar includes links for Home, About..., Payara Enterprise, Help, and Online Help. The main content area is titled 'JDBC Connection Pools'. On the left, a sidebar navigation menu lists various administrative sections such as server (Admin Server), Deployment Groups, Instances, Nodes, Clusters (Deprecated), Applications, Lifecycle Modules, and Monitoring Data. Under the 'Resources' section, 'JDBC Resources' and 'JDBC Connection Pools' are selected. The main panel displays a table titled 'Pools (7)' with columns for Select, Pool Name, Resource Type, Classname, and Description. The table lists seven connection pools: 'ComputerProjectPool' (Resource Type: javax.sql.DataSource, Classname: com.mysql.jdbc.jdbc2.optional.MysqlDataSource), 'H2Pool' (Resource Type: javax.sql.DataSource, Classname: org.h2.jdbcx.JdbcDataSource), 'TimerPool' (Resource Type: javax.sql.XADataSource, Classname: org.h2.jdbcx.JdbcDataSource), 'msadbpool' (Resource Type: javax.sql.DataSource, Classname: com.mysql.jdbc.jdbc2.optional.MysqlDataSource), 'mymyisadbpool' (Resource Type: javax.sql.DataSource, Classname: com.mysql.jdbc.jdbc2.optional.MysqlDataSource), and two unnamed entries (Resource Type: javax.sql.DataSource, Classname: com.mysql.jdbc.jdbc2.optional.MysqlDataSource).

- 4) Create new connection pool, Here in my case I've given the pool name testdbpool and select the following options mentioned in ss and then click next.

New JDBC Connection Pool (Step 1 of 2)

Identify the general settings for the connection pool.

\* Indicates required field

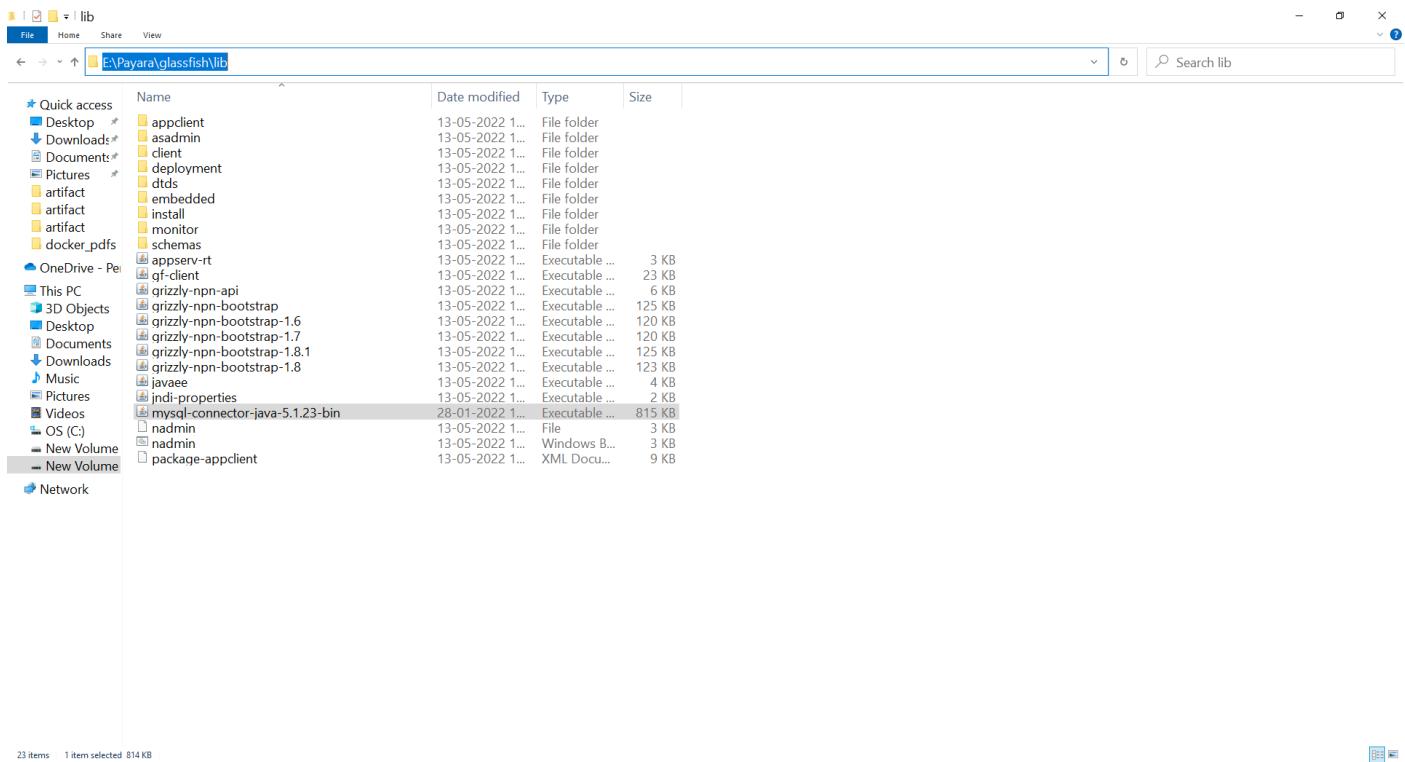
**General Settings**

Pool Name:	<input type="text" value="testdbpool"/>
Resource Type:	<input type="text" value="javax.sql.DataSource"/>
Database Driver Vendor:	<input type="text" value="MySQL"/>
Introspect:	<input checked="" type="checkbox"/> Enabled

- 5) Scroll down little bit and add the following properties, Your machine might contained more properties than properties mentioned in following ss. Search or Add the properties and Leave extra properties as it is and click next

URL	<input type="text" value="jdbc:mysql://localhost:3306/testdb?useSSL=false"/>
url	<input type="text" value="jdbc:mysql://localhost:3306/testdb?useSSL=false"/>
driverClass	<input type="text" value="com.mysql.jdbc.Driver"/>
driver	<input type="text" value="com.mysql.jdbc.Driver"/>
password	<input type="text" value="root"/>
databaseName	<input type="text"/>
roleName	<input type="text"/>
serverName	<input type="text" value="localhost"/>
datasourceName	<input type="text"/>
user	<input type="text" value="root"/>
networkProtocol	<input type="text"/>
portNumber	<input type="text" value="3306"/>

- 6) Before moving on I want you check the weather mysql-connector-java-5.1.23-bin or mysql-connector-java-5.1.30.jar is existed in your payara server location which is mentioned in below ss. If not then add over there



- 7) Back to the admin console of payara in your browser and click on JDBC Resources click new add new one

JDBC resources provide applications with a means to connect to a database.

Select	JNDI Name	Logical JNDI Name	Enabled	Connection Pool	Description
<input type="checkbox"/>	jdbc/ComputerProjectPool		✓	ComputerProjectPool	
<input type="checkbox"/>	jdbc/_TimerPool		✓	_TimerPool	
<input type="checkbox"/>	jdbc/__default	java:comp/DefaultDataSource	✓	H2Pool	
<input type="checkbox"/>	jdbc/msadb		✓	msadbpool	
<input type="checkbox"/>	jdbc/mymssql		✓	mymssqlpool	
<input type="checkbox"/>	jdbc/restdbpool		✓	restdbpool	
<input type="checkbox"/>	jdbc/testdbpool		✓	testdbpool	

User: admin Domain: domain1 Server: localhost Enable Asadmin Recorder

localhost:4848/common/index.jsf

- 8) Click on New button and new jdbc resource and then click on ok button, See the below ss.

JNDI Name: \* jdbo/testpool

Pool Name: testdbpool

Description:

Status:  Enabled

Additional Properties (0)

Select	Name	Value	Description
No items found.			

- 9) Again to Go to JDBC Connection pools and select the connection pool that you've created recently and click on ping button , You must get ping successed message.

Ping Succeeded

Edit JDBC Connection Pool

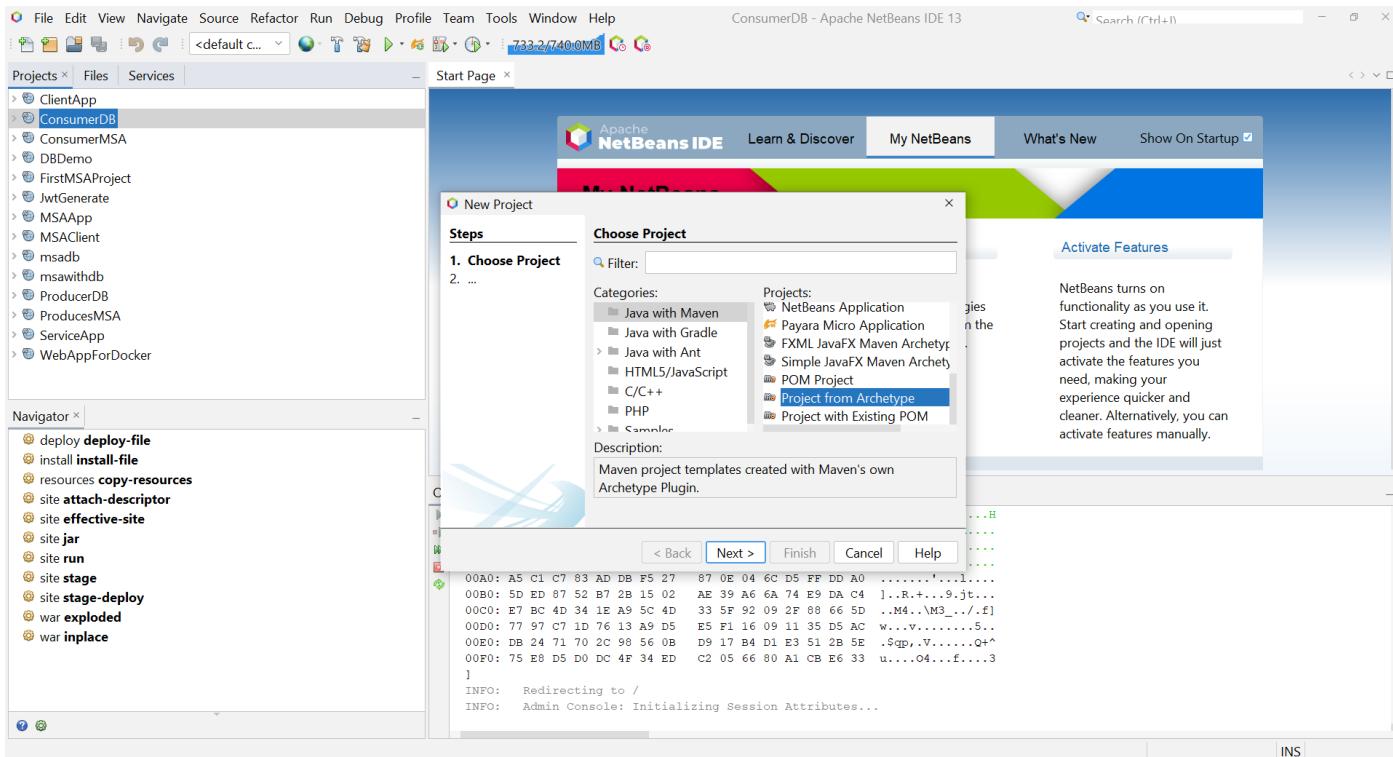
General Settings

Pool Name: testdbpool

Resource Type: javax.sql.DataSource

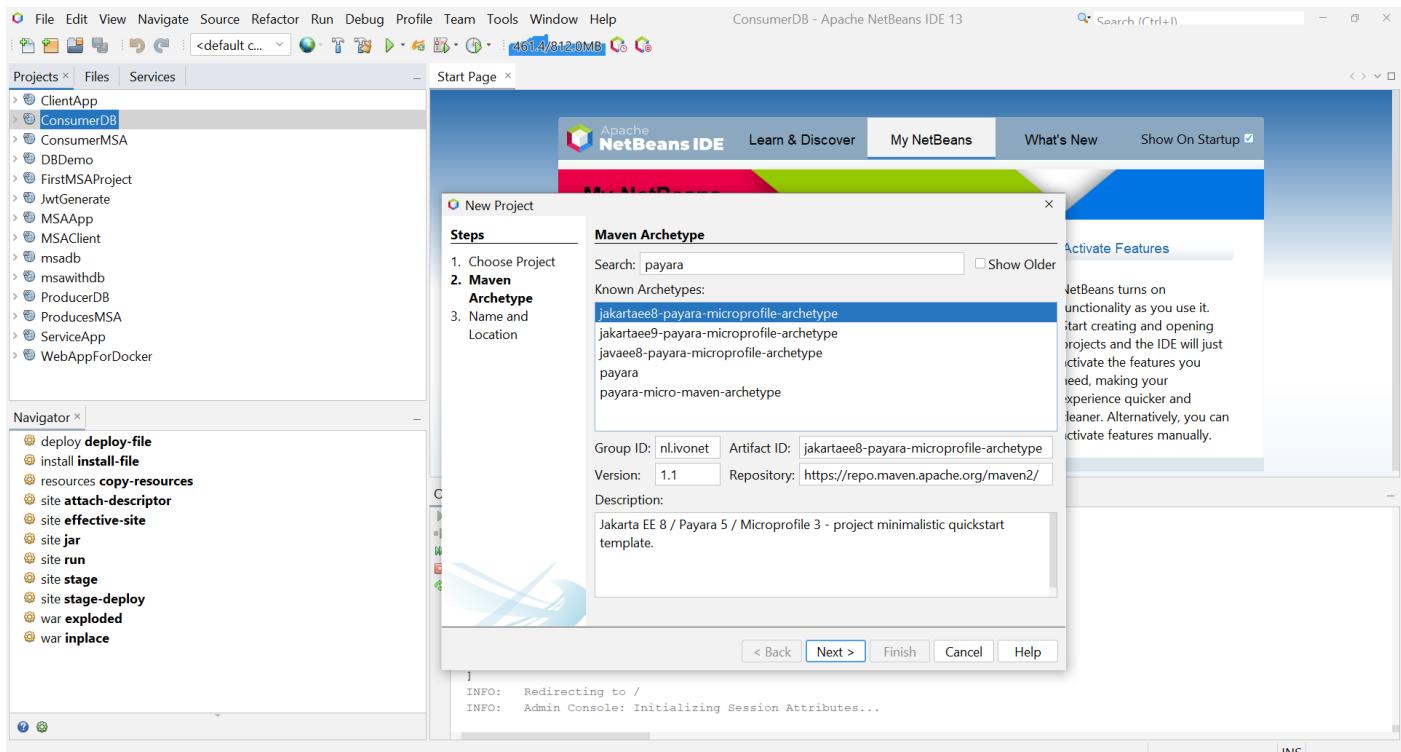
Alright Your database part is done, Now move on Application creation part

10) Create new maven project with type Project from Archetype and click next

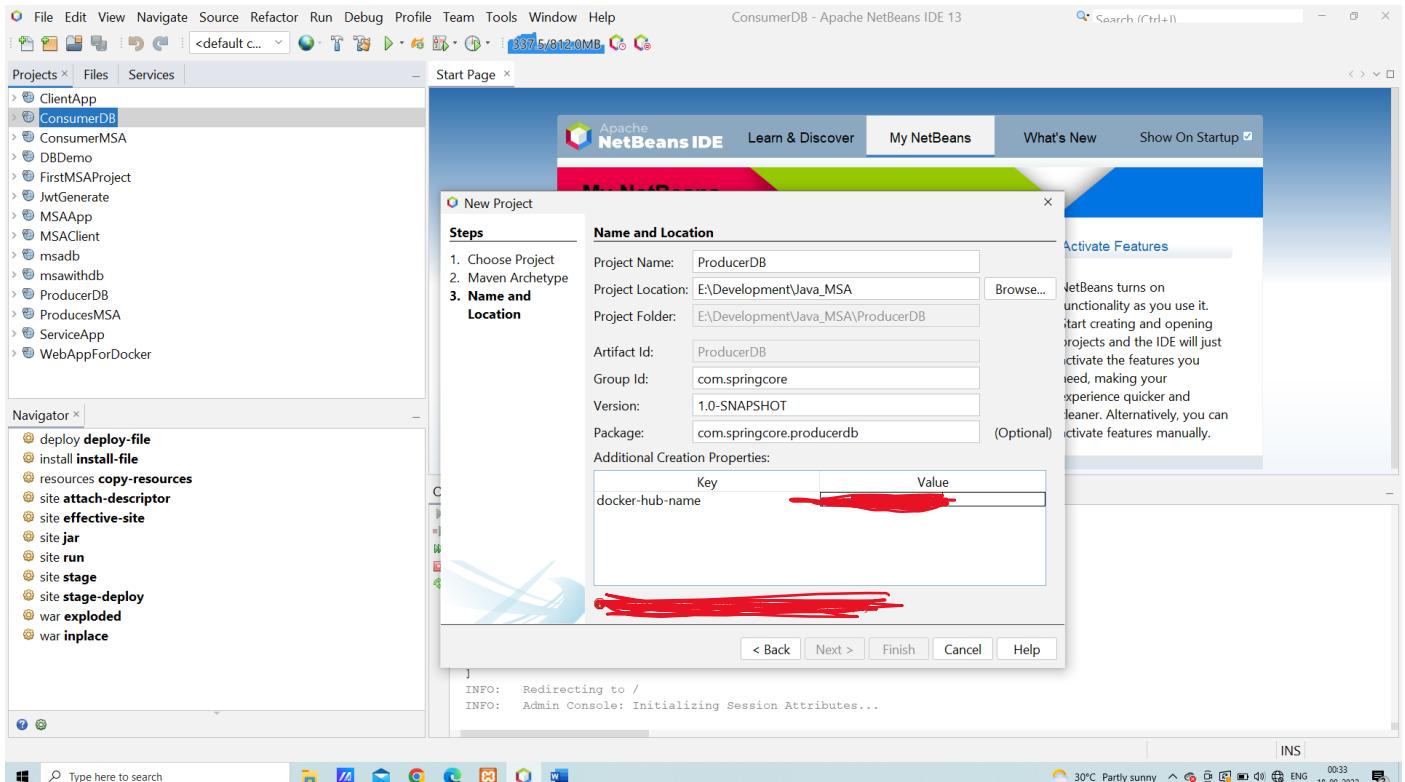


11) Search jakartaee8-payara-microprofile-archetype and click next

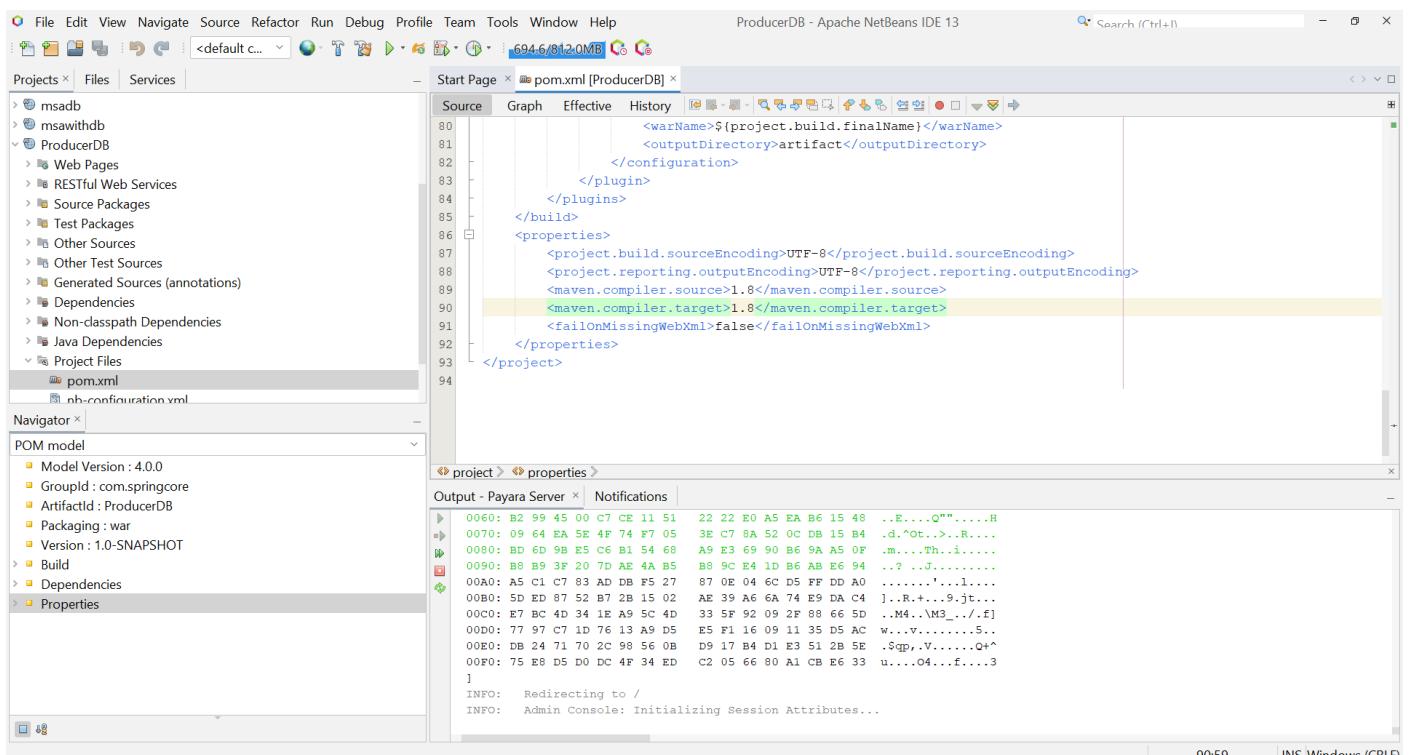
(Note : Currently we are creating Producer App)



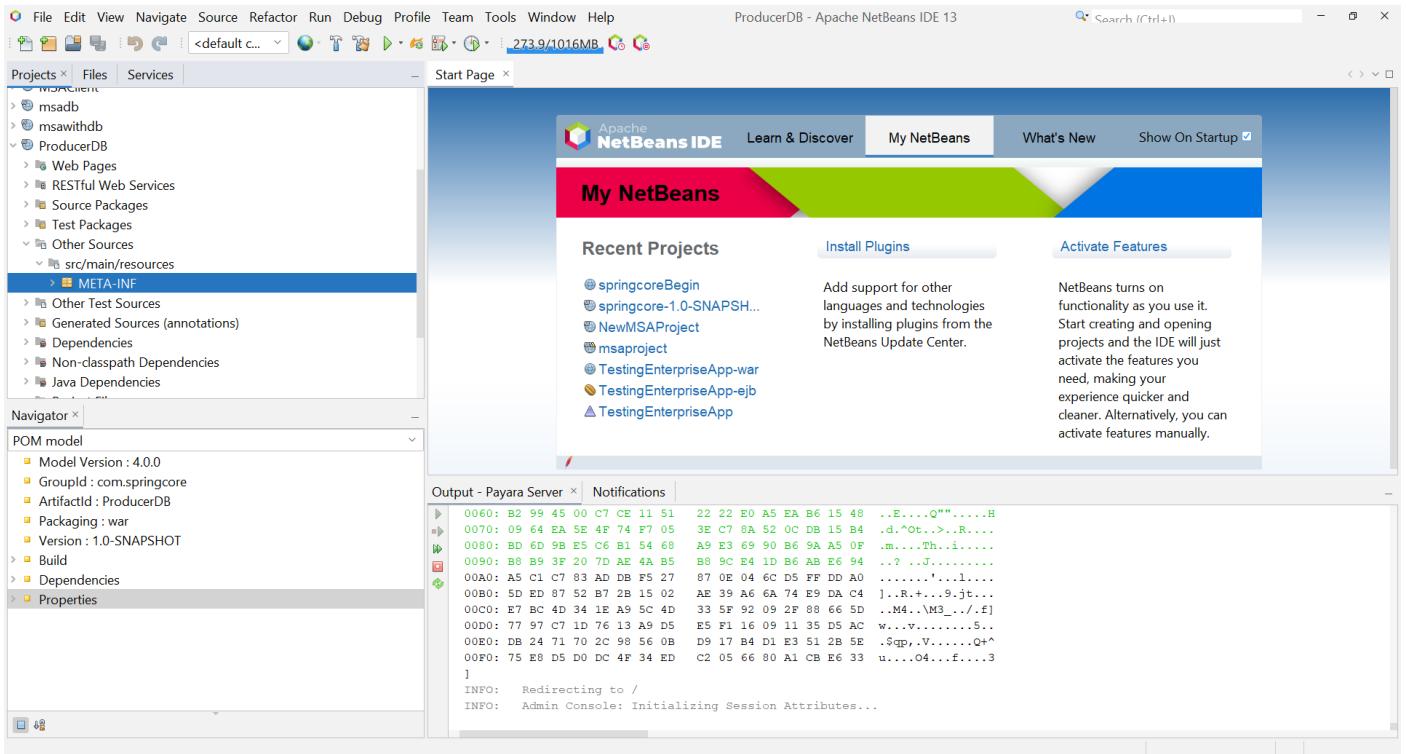
12) Enter the project Name and enter the docker-hub-name (optional) , Here in my case I've given ProducerDB and click next



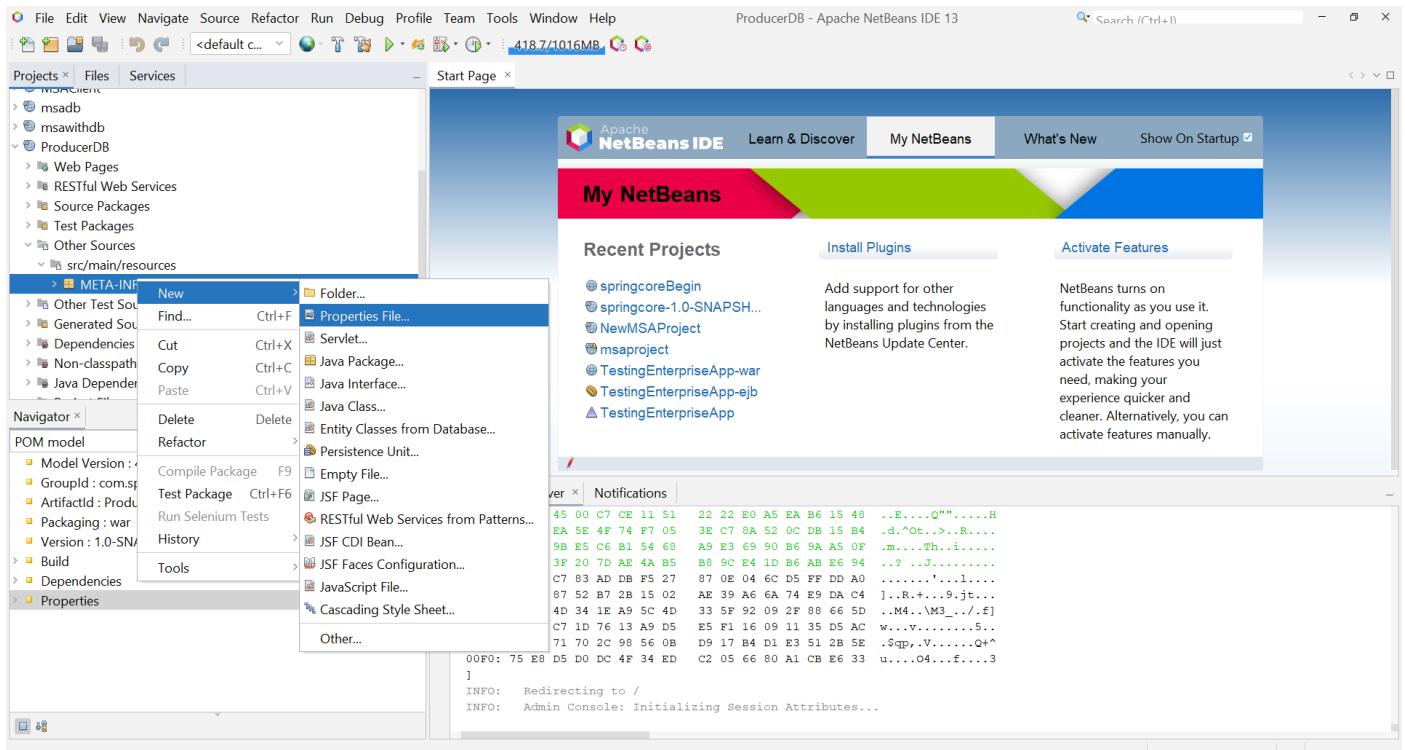
13) Go to Project and select Project Files and Select pom.xml and make some changes mentioned in ss.



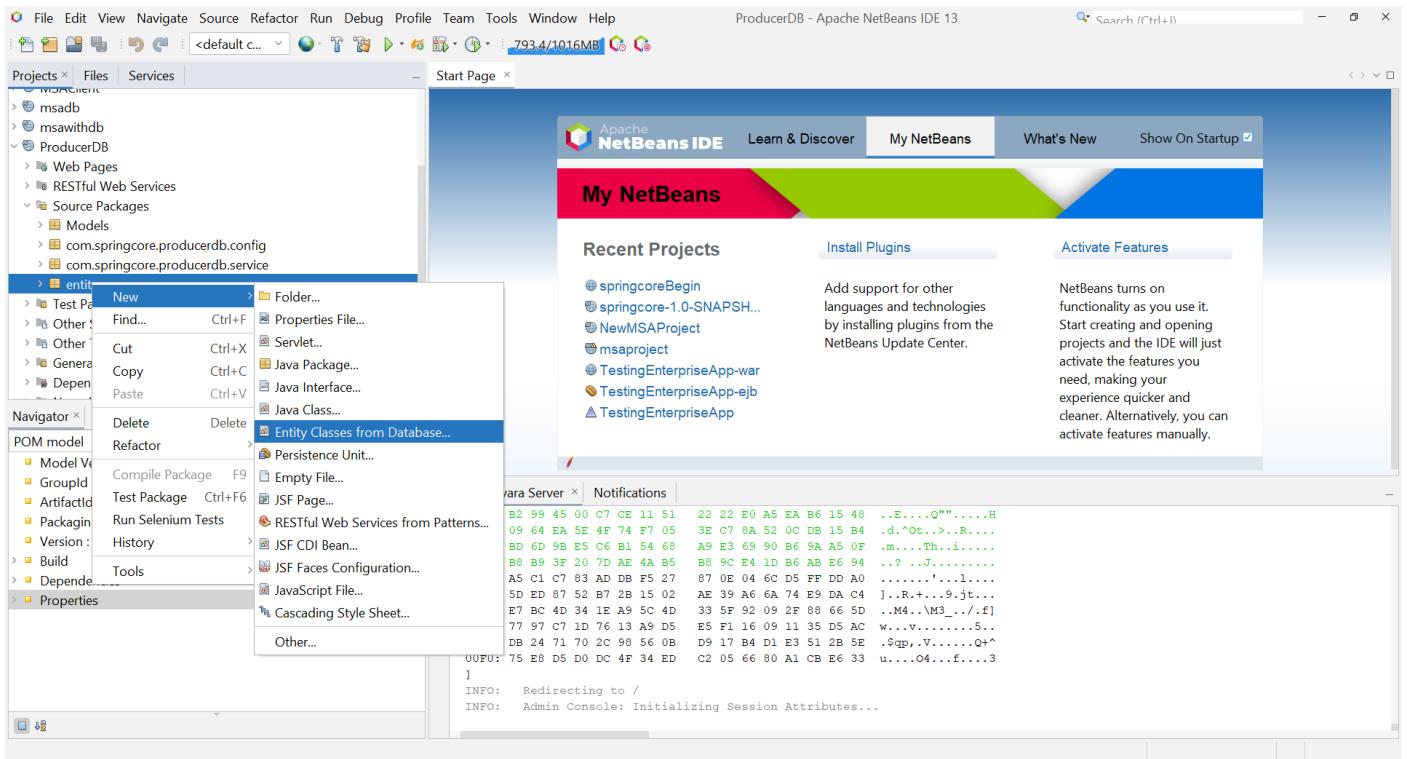
14) Go to Other Sources -> src/main/resources and create folder named META-INF (name is mandatory)



15) Right click on that META-INF and create one properties file named micrprofile-config

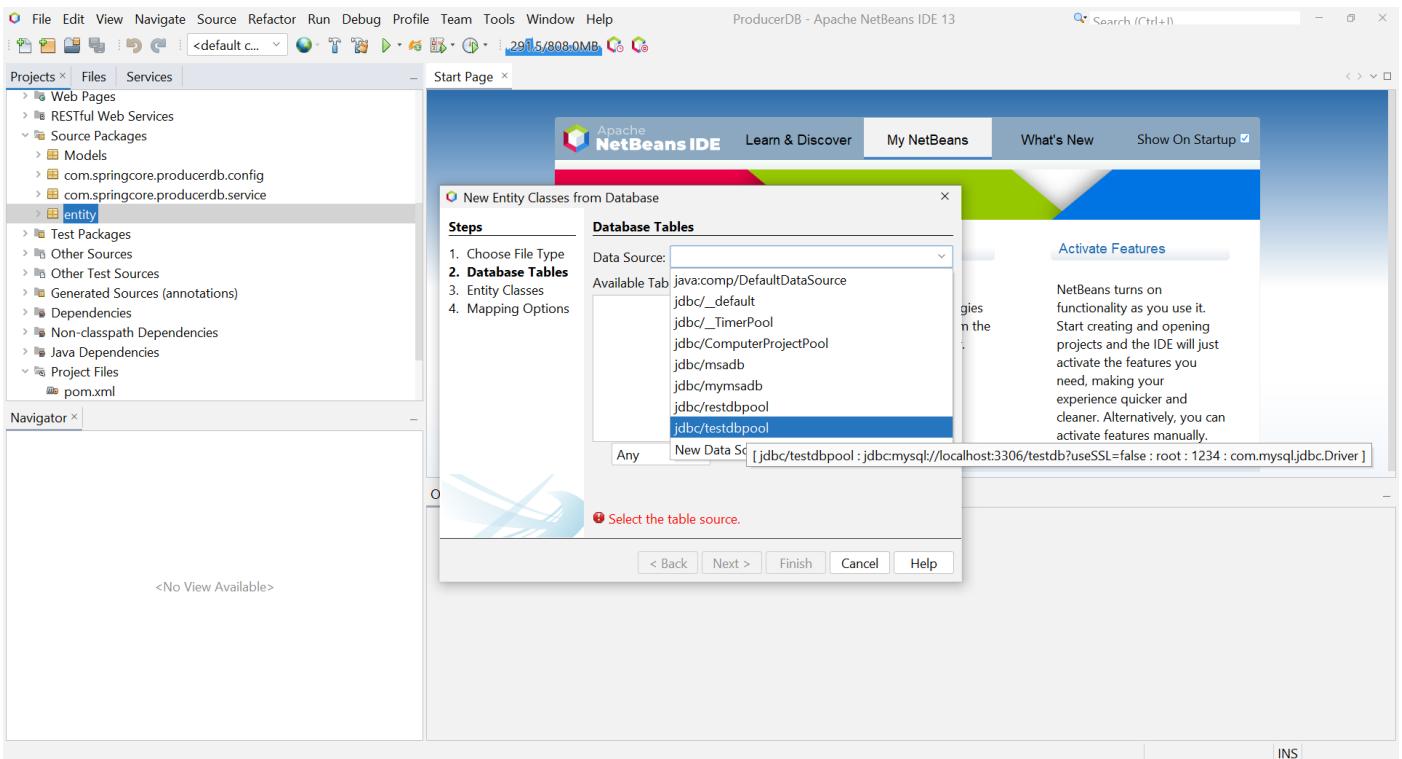


- 16) Go to source packages and create one package called entity, Right click on that package and select new->entity classes from database

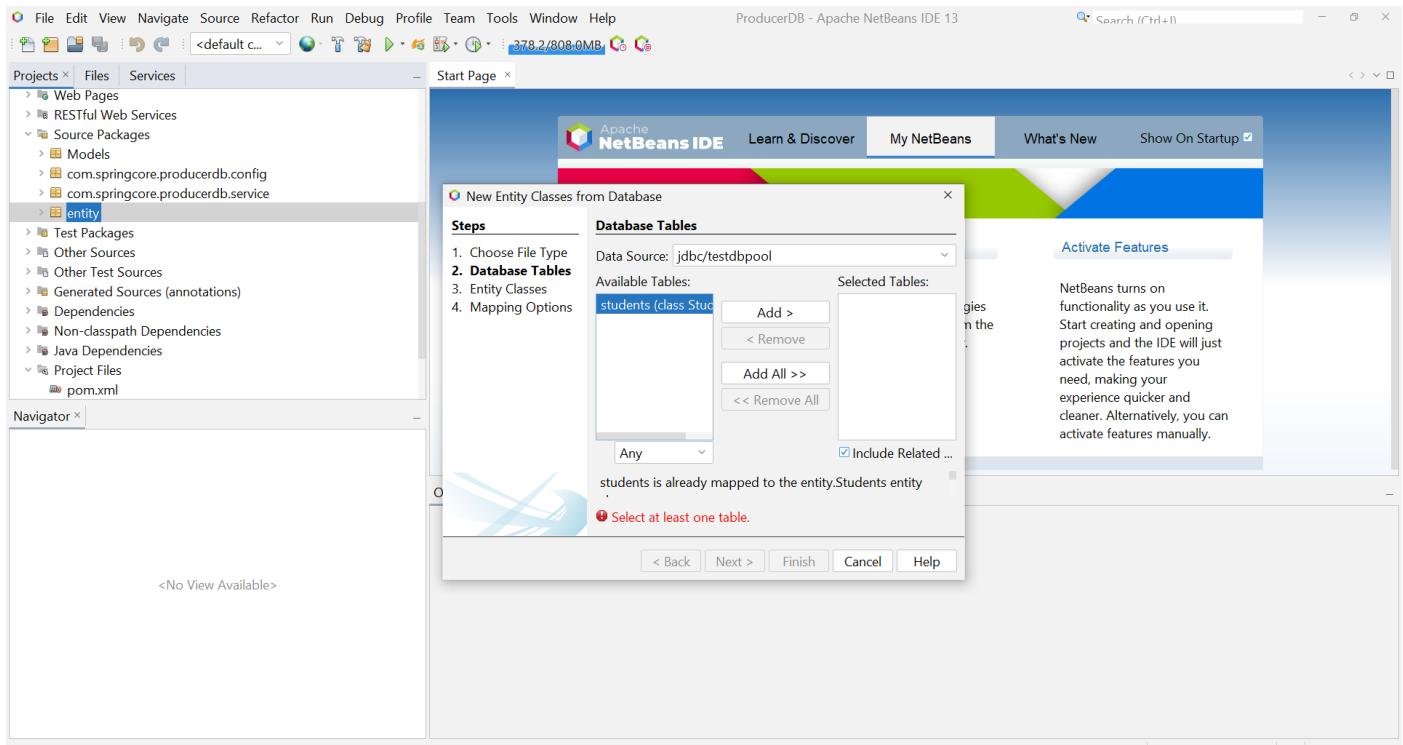


- 17) Fetch from server , Choose your server and then click next.

- 18) Select your jdbc resource , Here in my it would be jdbc/testpooledb and click next.  
 (If next dialogue window ask you password then enter mysql password)

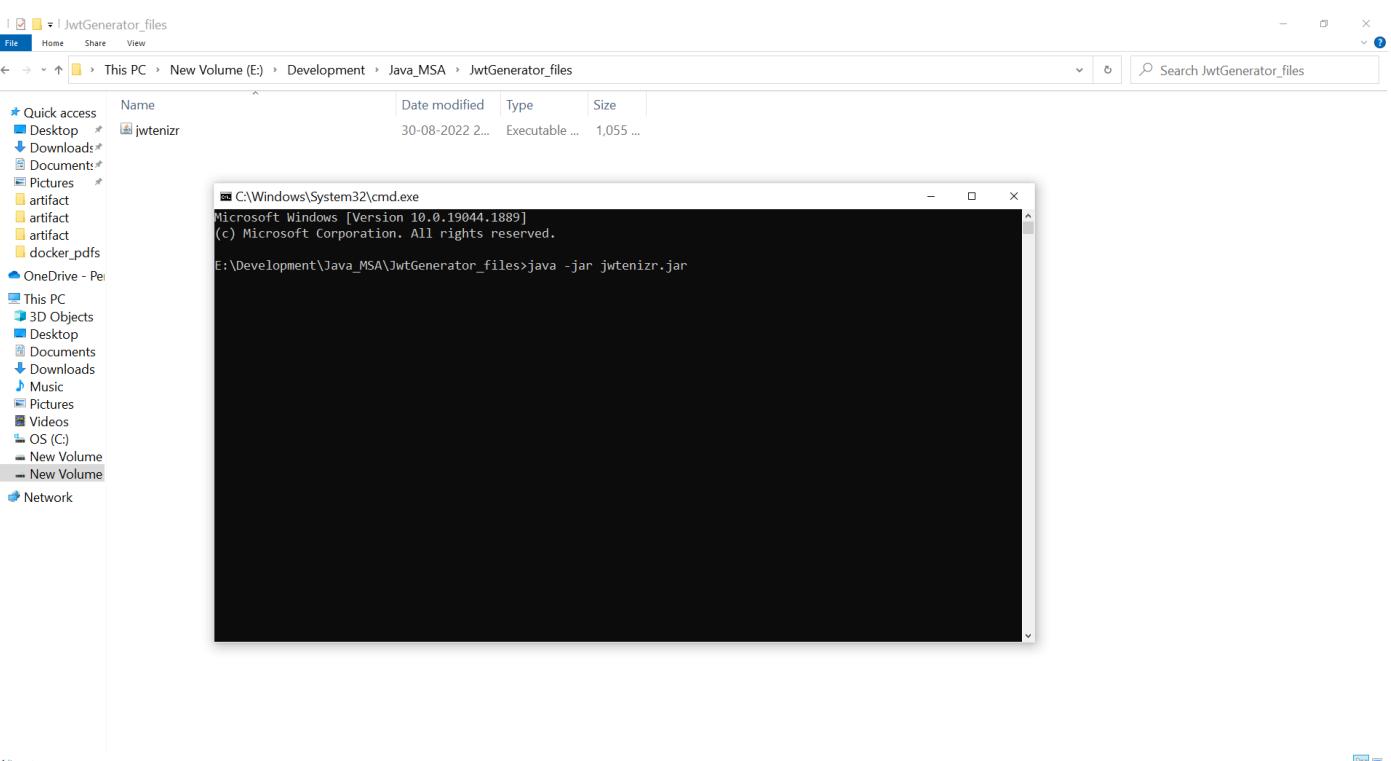
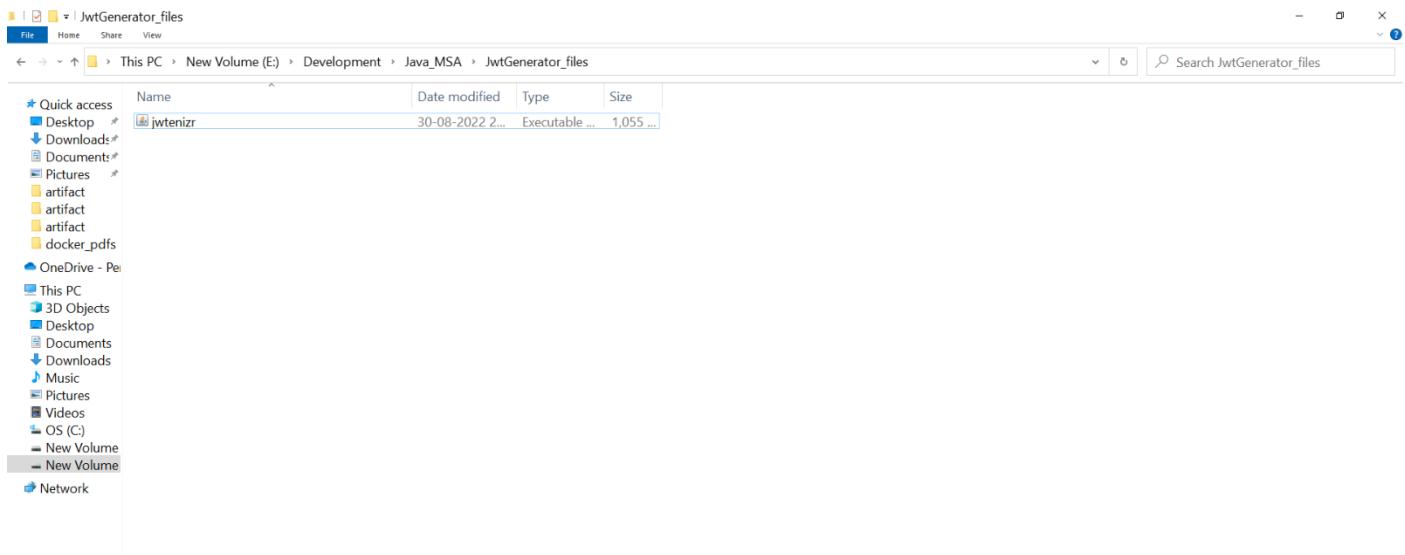


- 19) Add the tables and click next, next till finish don't make any changes



Pefect! Let's create JWT token and issuer , For that you need jwt jar file, Put jwt jar file in one blank folder

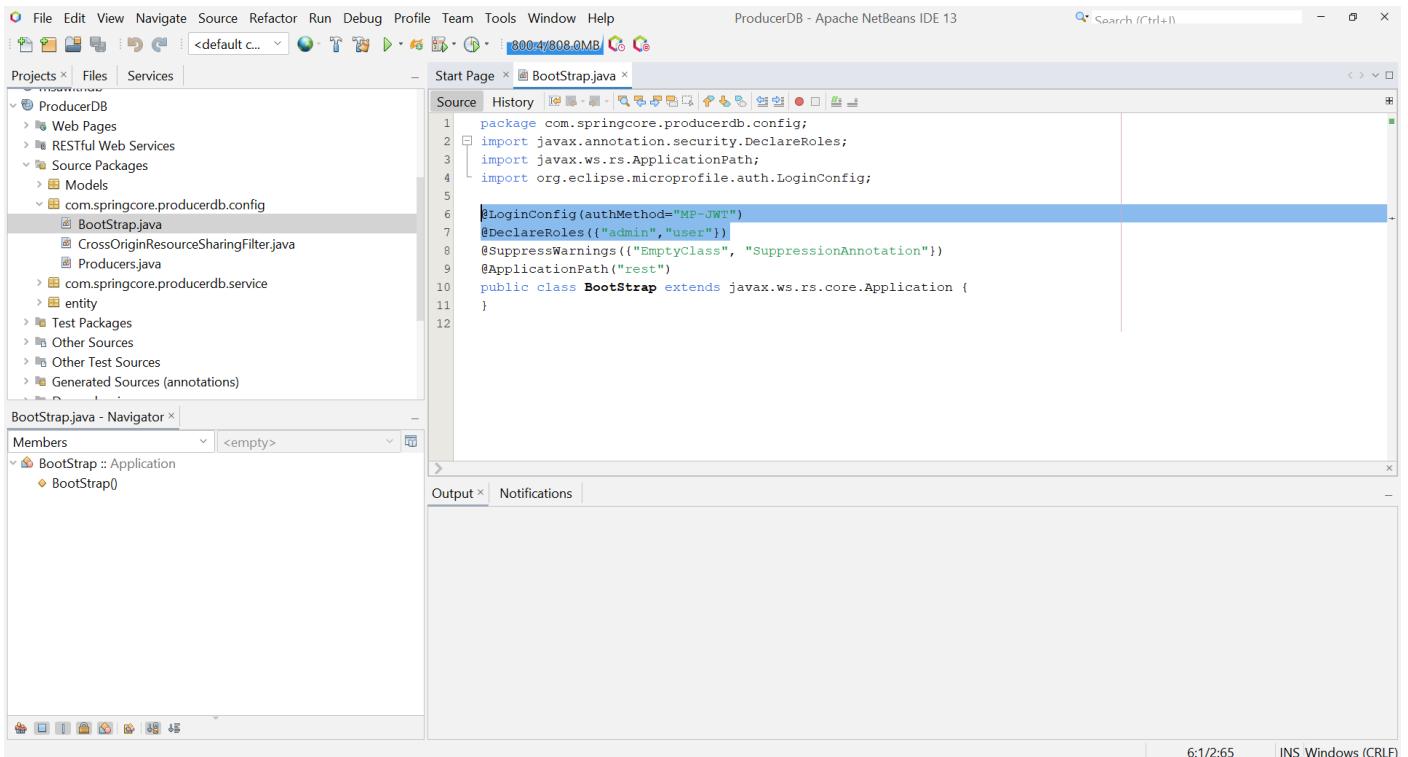
20) Open command prompt/terminal over there and add the command



21) Some file would be generated and then open jwt-token file in your favourite text editor

and make some changes like groups set to admin and user and then save that file , again fire the command using command prompt or terminal . See that command in step no 20. Don't forget this step. Otherwise your app will give unexpected behaviour

22) Go to your project go inside the package com.springcore.projectname.config and open Bootstrap.java and add some annotations highlighted in ss.



The screenshot shows the Apache NetBeans IDE 13 interface. The top menu bar includes File, Edit, View, Navigate, Source, Refactor, Run, Debug, Profile, Team, Tools, Window, and Help. The title bar says "ProducerDB - Apache NetBeans IDE 13". The left sidebar has sections for Projects, Files, and Services, with "ProducerDB" selected. Under "ProducerDB", there are Web Pages, RESTful Web Services, Source Packages (Models, com.springcore.producerdb.config), Test Packages, Other Sources, Other Test Sources, and Generated Sources (annotations). The main editor window displays the code for "BootStrap.java". The code is as follows:

```
1 package com.springcore.producerdb.config;
2 import javax.annotation.security.DeclareRoles;
3 import javax.ws.rs.ApplicationPath;
4 import org.eclipse.microprofile.auth.LoginConfig;
5
6 @LoginConfig(authMethod="MP-JWT")
7 @DeclareRoles({"admin","user"})
8 @SuppressWarnings({"EmptyClass", "SuppressionAnnotation"})
9 @ApplicationPath("rest")
10 public class BootStrap extends javax.ws.rs.core.Application {
11 }
12
```

The Navigator panel on the left shows the members of the "BootStrap :: Application" class, which includes "BootStrap0". The bottom right corner of the interface shows the status "6:1:2:65" and "INS Windows (CRLF)".

23) Create new package Model and create new Java class file , In my case I've created Models package and DataModel.java file in that package and add the code

```

1 package Models;
2 import entity.Students;
3 import java.util.Collection;
4 import javax.ejb.Stateless;
5 import javax.persistence.EntityManager;
6 import javax.persistence.PersistenceContext;
7
8 @Stateless
9 public class DataModel
10 {
11     @PersistenceContext(unitName = "com.springcore_ProducerDB_war_1.0-SNAPSHOTPU")
12     EntityManager em;
13
14     public Collection<Students> getStudents()
15     {
16         return em.createNamedQuery("Students.findAll").getResultList();
17     }
18 }

```

24) Replace unitName with your unitName available on persistence.xml file ,  
(Other sources->src/main/resources/META-INF)

```

<!-- version="1.0" encoding="UTF-8"-->
<!-- persistence version="2.1", xmlns="http://xmlns.jcp.org/xml/ns/persi
<!-- <persistence-unit name="com.springcore_ProducerDB_war_1.0-SN

```

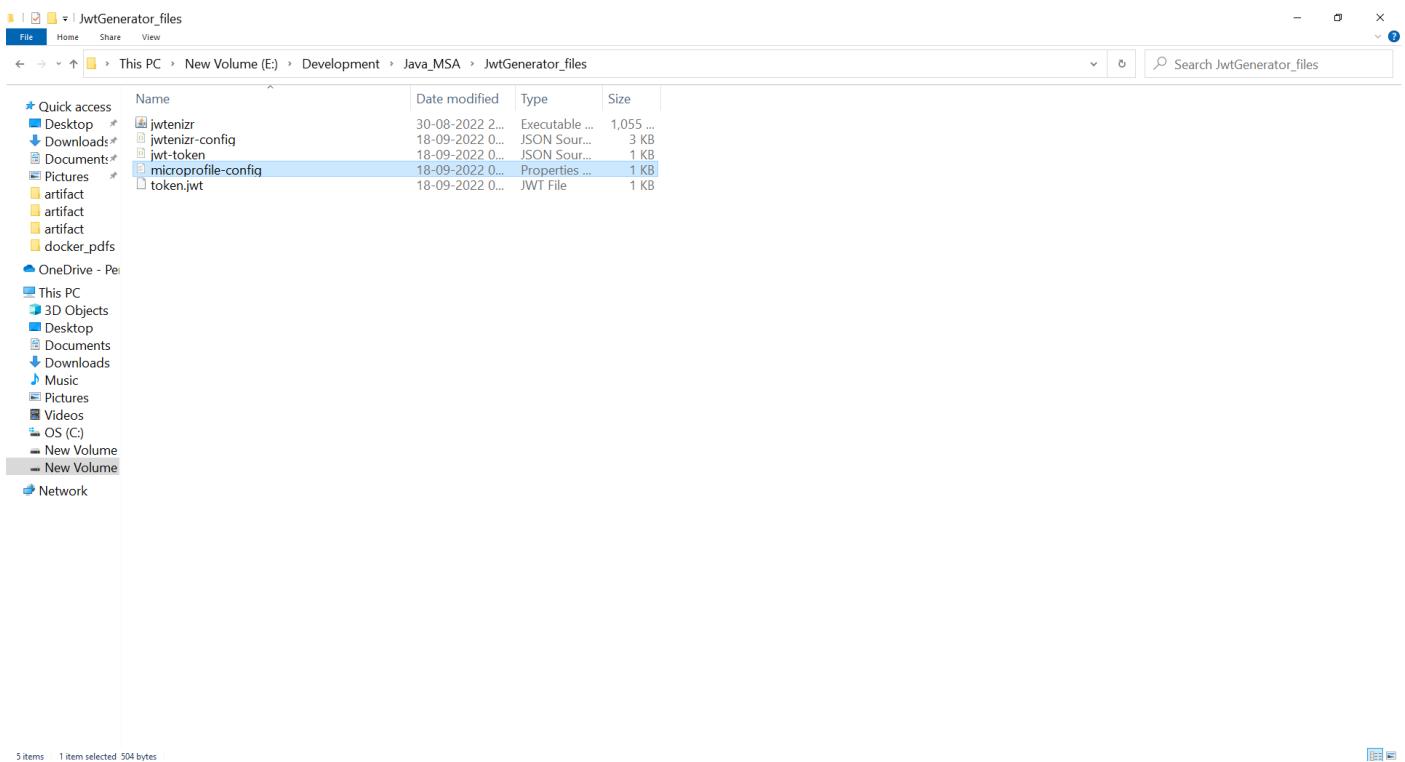
Output - Payara Server | Notifications |

```

Instances: {
    DataGrid: development Name: server Lite: false This: true UUID: 1d9b79d2-cb25-41ed-a728-aa49f1c95d11 Address: /192.168.43.172:
}
INFO: Payara Notification Service bootstrapped.
INFO: Bootstrapping Monitoring Console Runtime
INFO: Starting monitoring data collection for server
INFO: Starting monitoring watch collection for server
INFO: Network Listener JMS_PROXY_default_JMS_host started in: 9ms - bound to [/0.0.0.0:7676]
INFO: JMXStartupService has started JMXConnector on JMXService URL service:jmx:rmi://0.0.0.0:8686/jndi/xmi://0.0.0.0:8686/jmx:
INFO: Skipping registration of inhabitant for service reference [org.osgi.service.metatype.MetaTypeProvider] as the service o
INFO: Loading application _adminui done in 1,272 ms
INFO: Initializing Mojarra [version.string] for context ''
INFO: Loading application [_adminui] at [/]

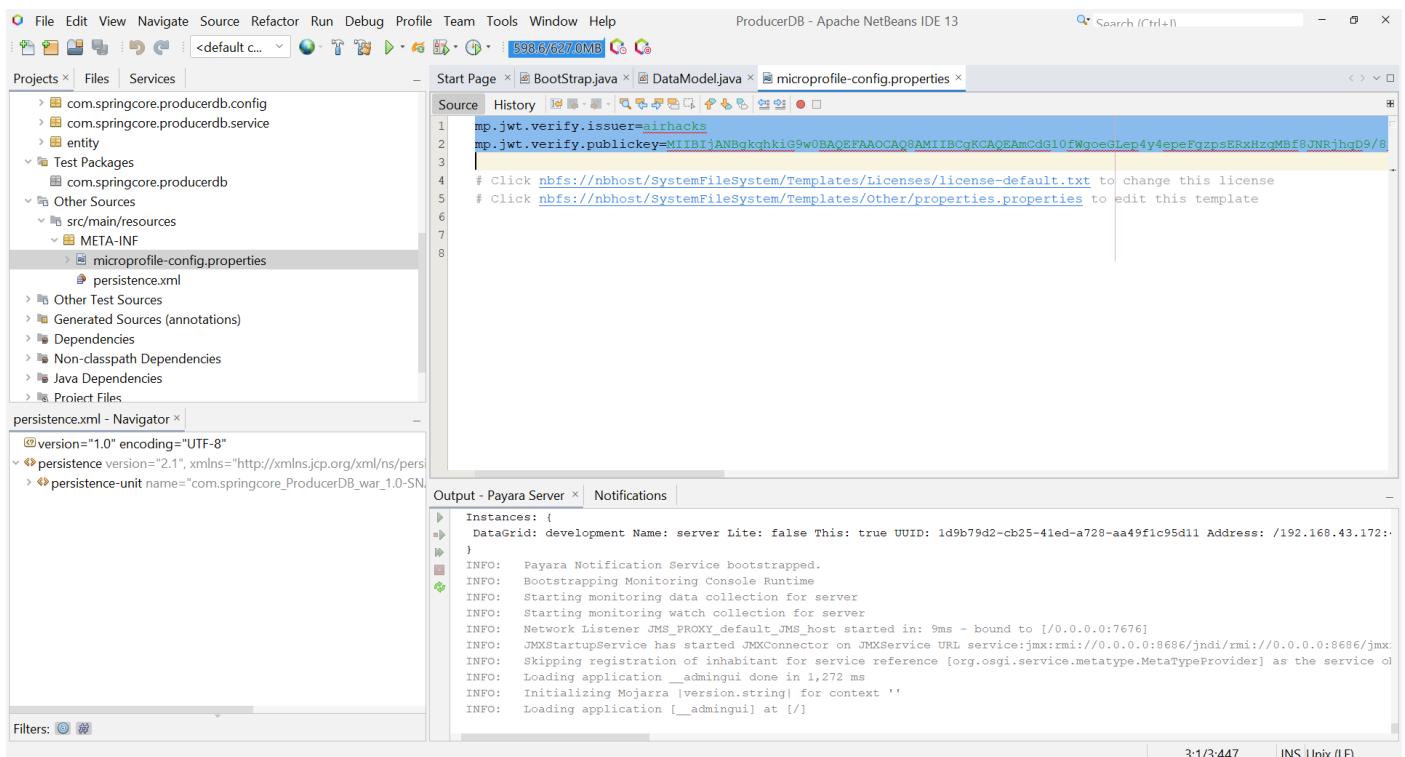
```

25) Open microprofile-config file that you've generated in your favourite text editor

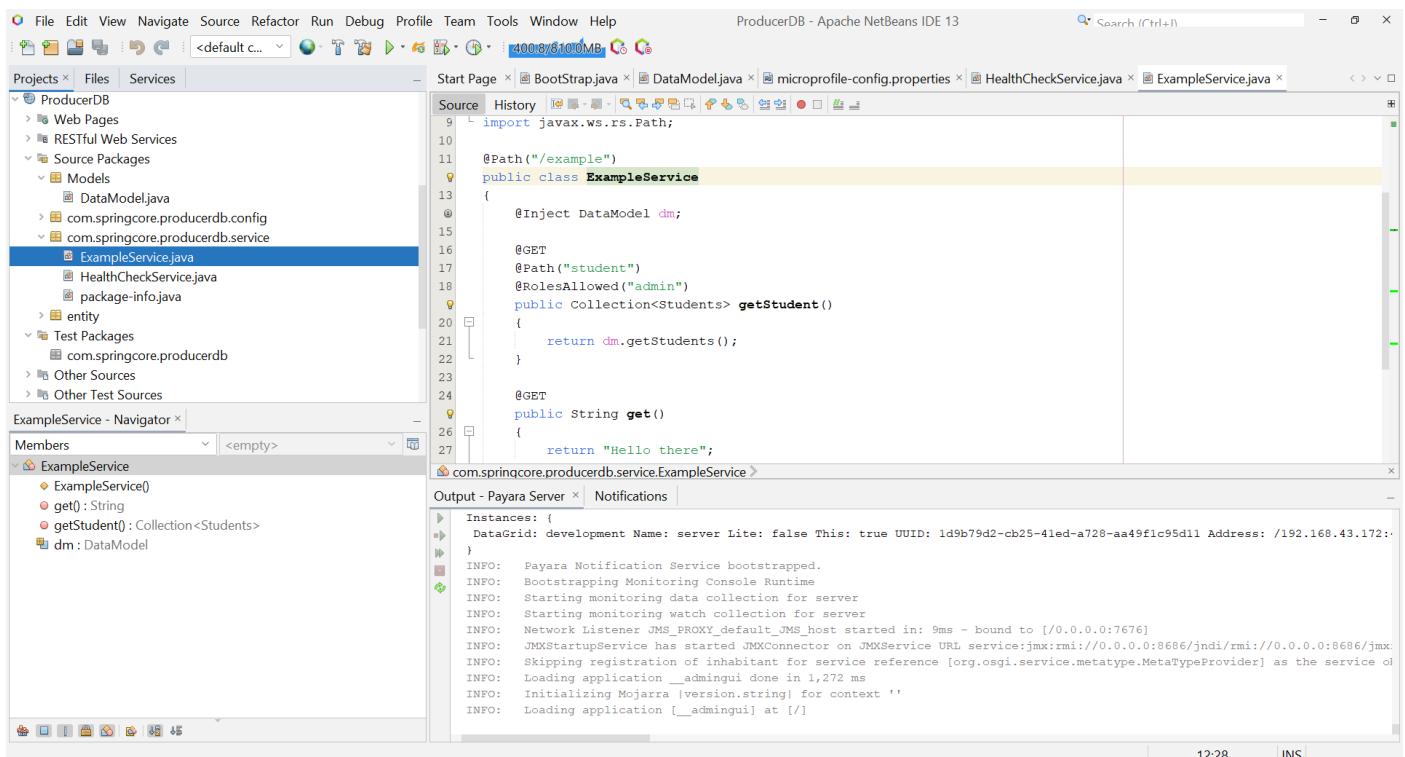


26) Copy the the below content and paste on your microprofile-config properties file

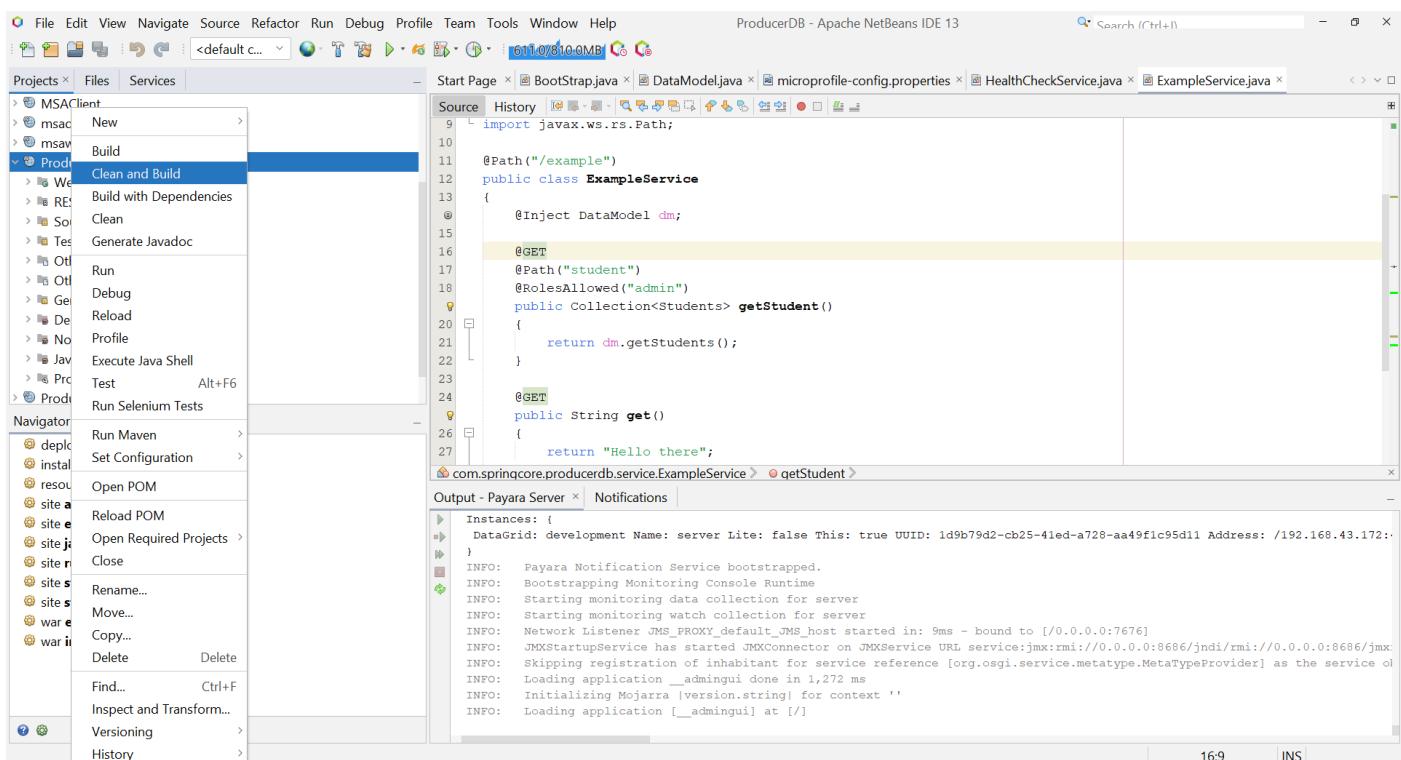
```
#generated by jwttenizr
#Sun Sep 18 01:01:24 IST 2022
mp.jwt.verify.issuer=airhacks
mp.jwt.verify.publickey=MIIBIjANBgkqhkiG9w0BAQEFAOAQ8AMIIIBCgKCAQEAhGGit5maOsraxLW2zHp5HPUiX//
```



27) Open ExampleService.java (source package/com.springcore.projectname.service / ExampleService.java) and add the code

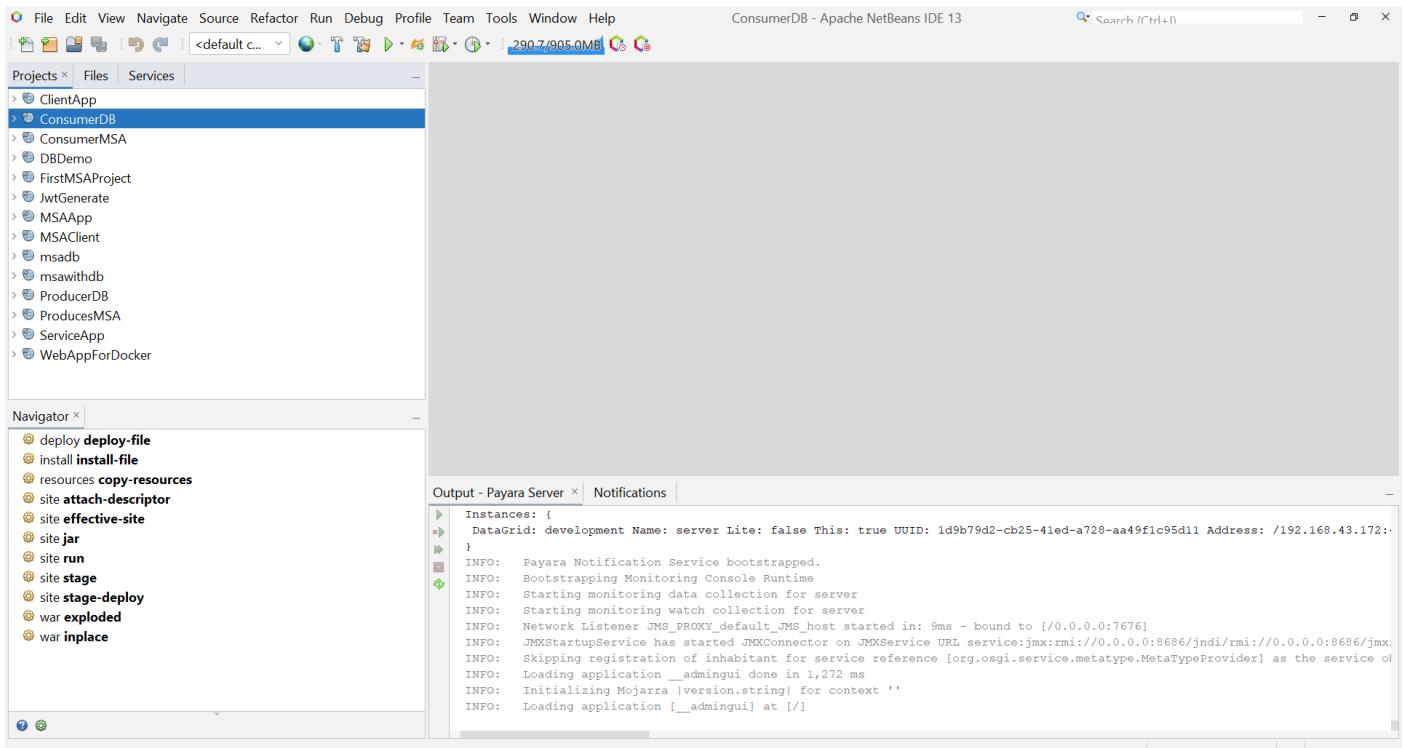


28) Right click on the project and click on clean and build

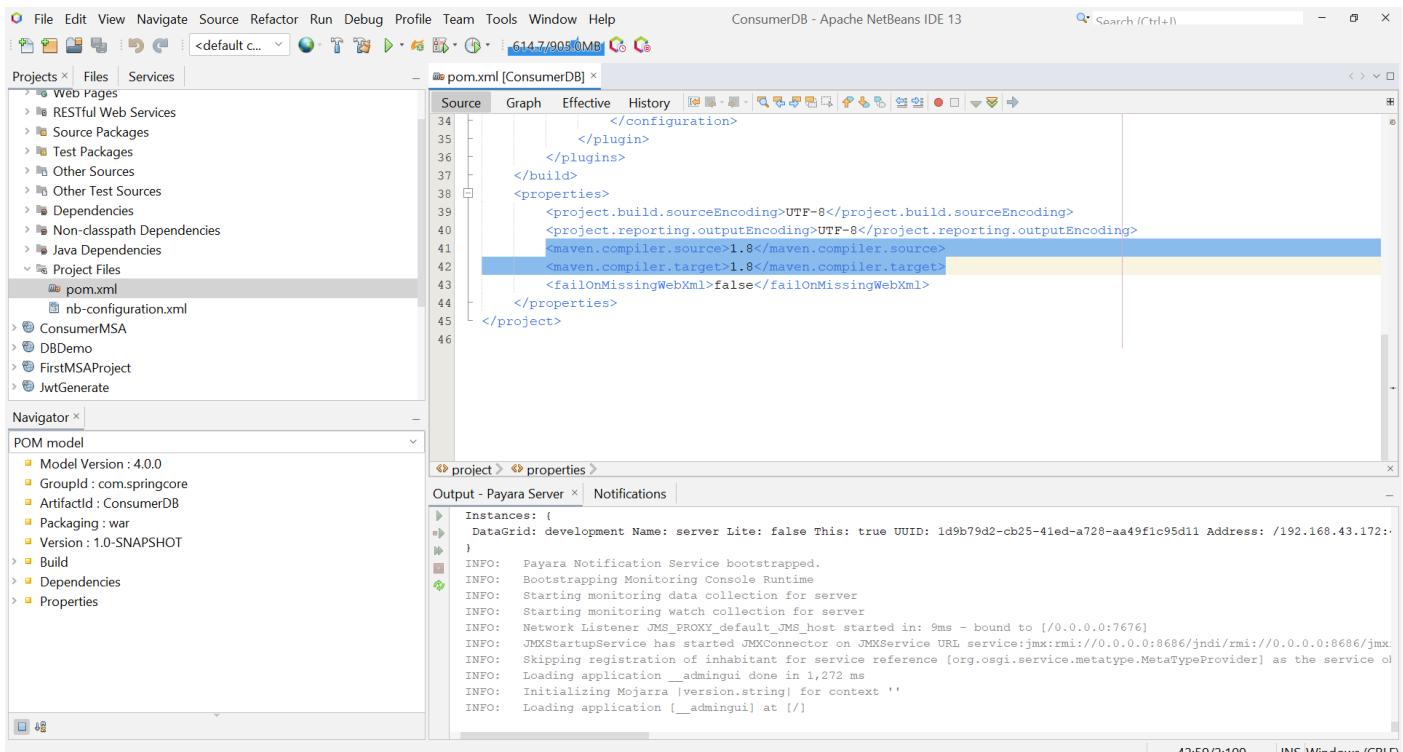


Alright we have done till upto Producer App Let's move on to create Consumer App which fetch the data from Producer App.

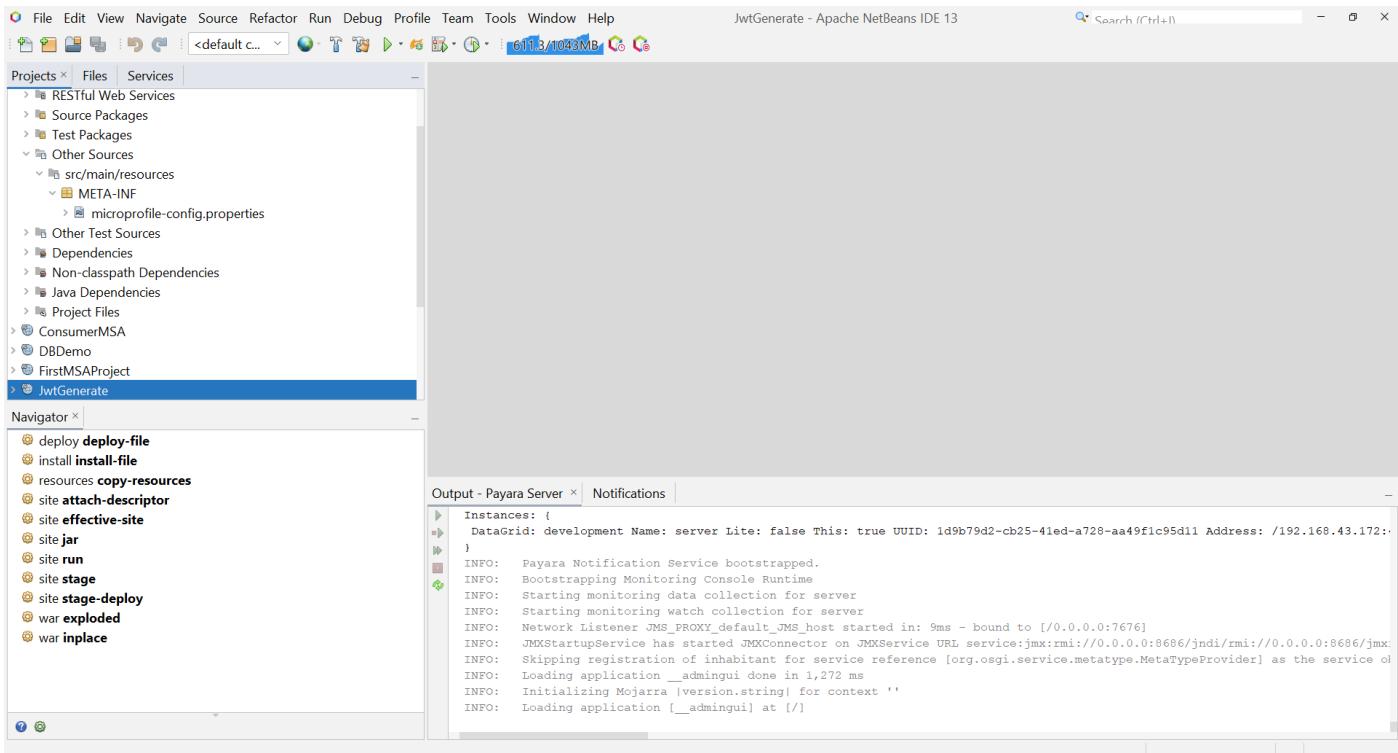
29) Create new project same as earlier , In my case I've given the project name ConsumerDB.



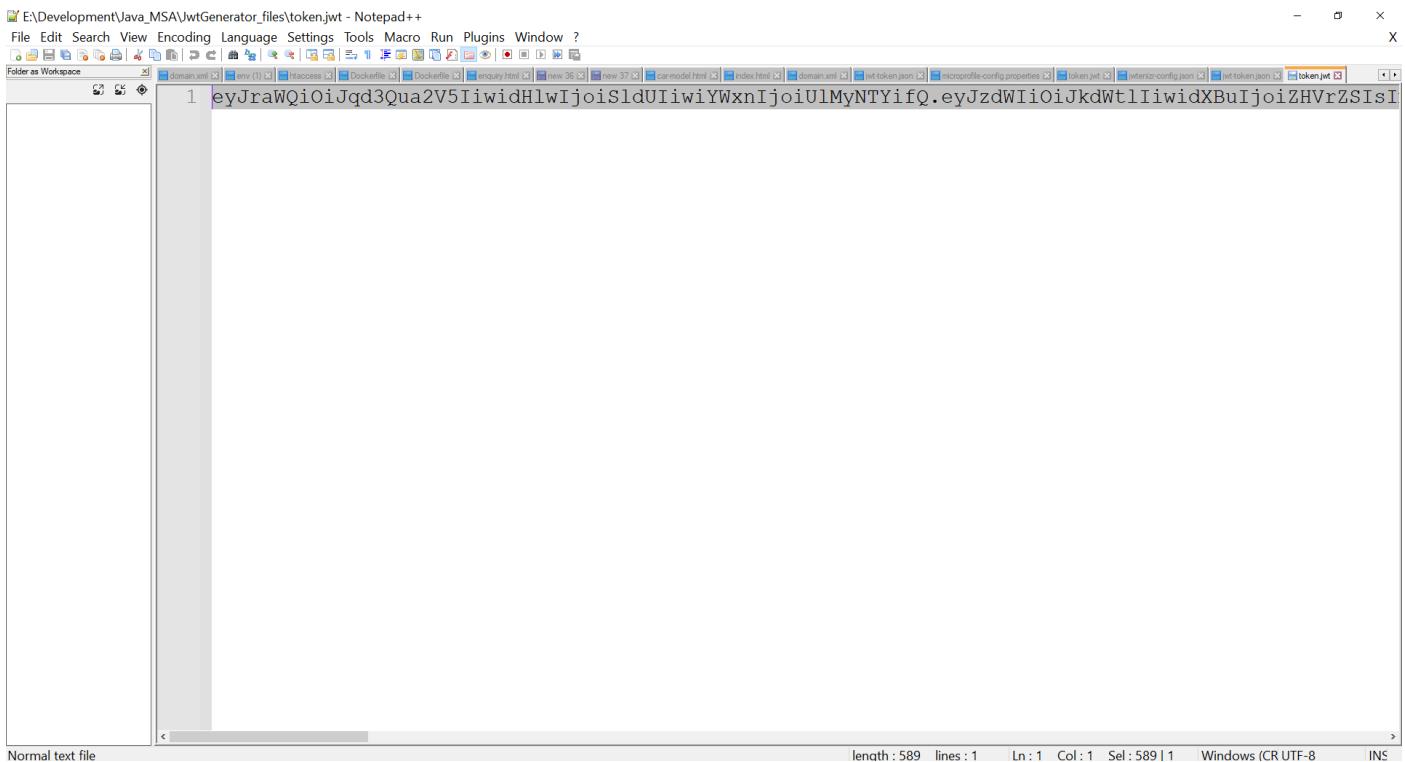
30) As we did earlier make some changes in pom.xml file



31) Go to Other Sources->src/main/resources and create new folder (Not the package) and name should be META-INF and then create one property file name should microprofile-config in Consumer App as we did same as earlier.

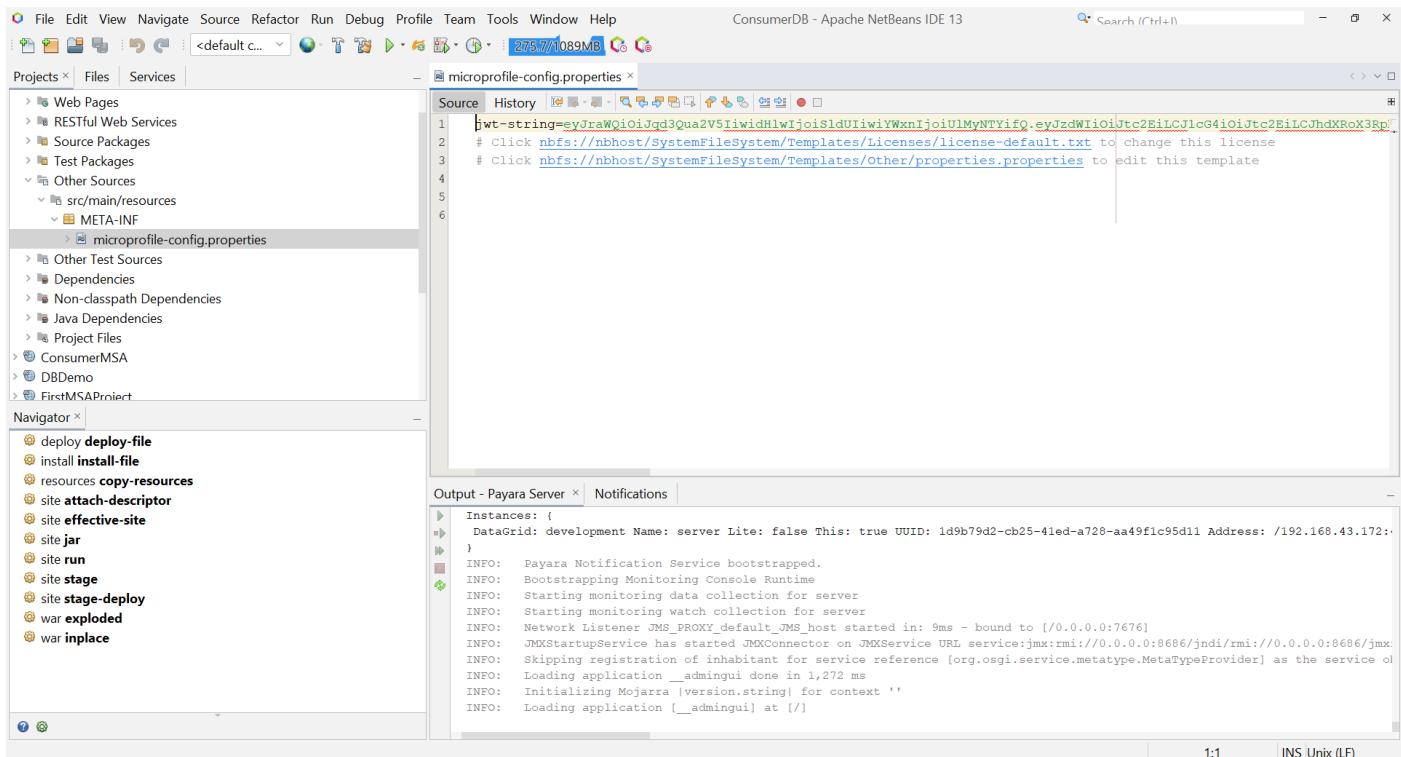


32) Now Open jwt.token file that you've generated in your favourite text editor copy the whole string.



33) Paste that string in your microprofile-config property file of Consumer App

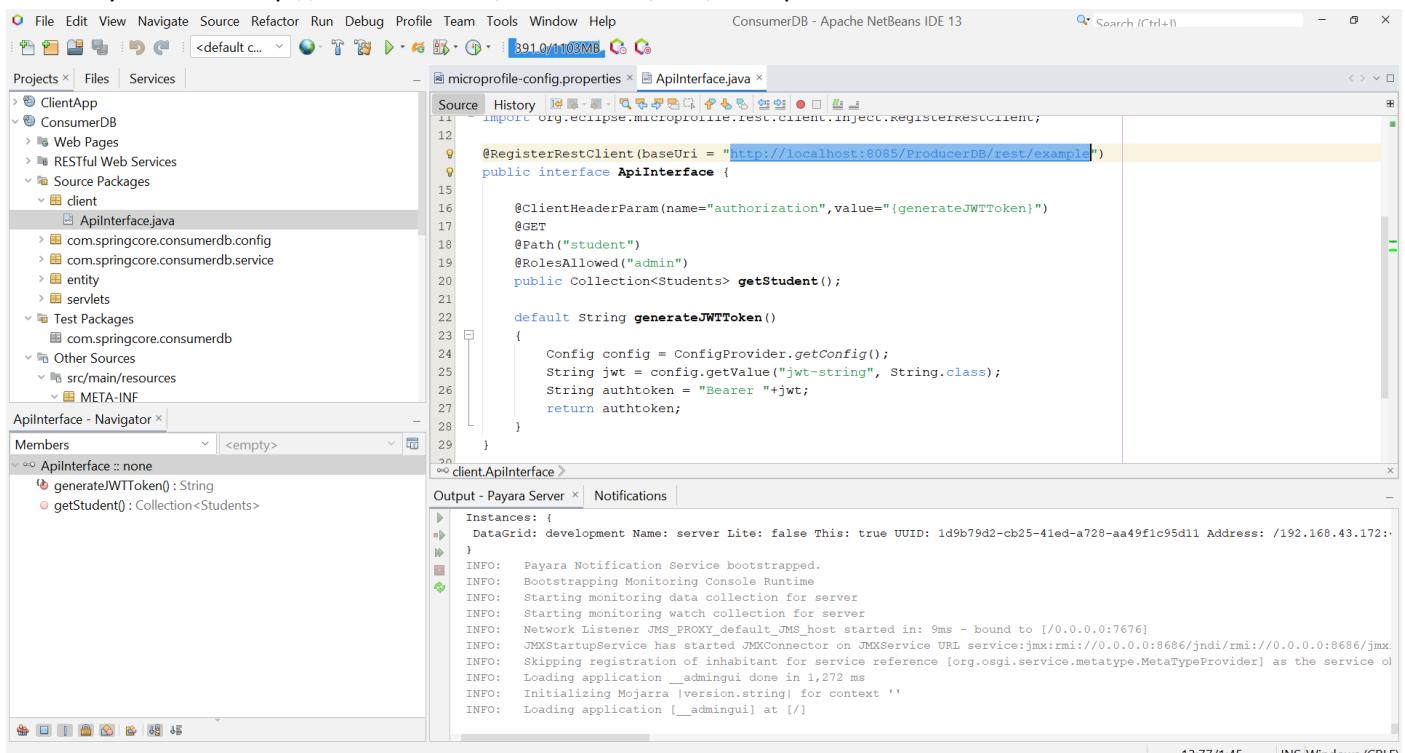
Format should be **jwt-string=paste\_your\_jwt\_string**



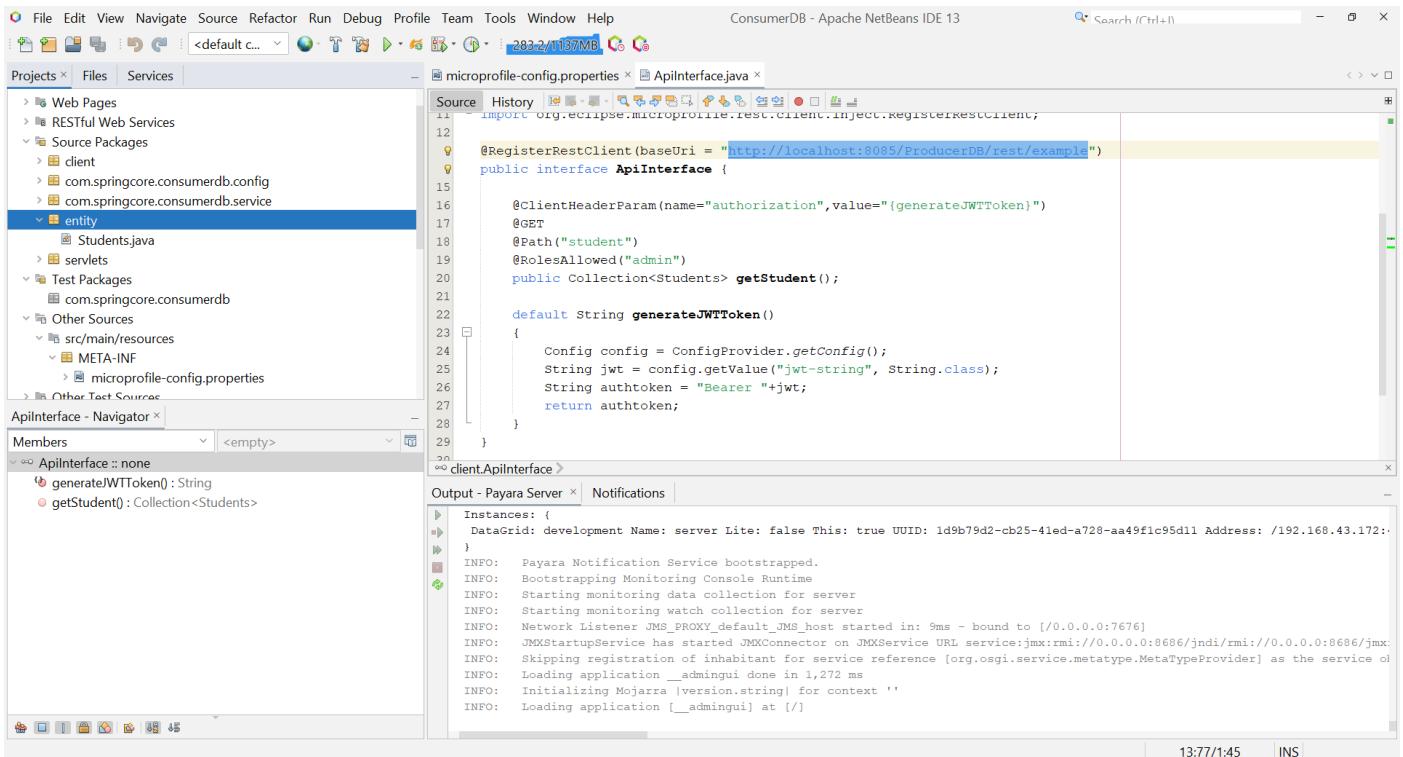
34) Create one package and in that package create one Interface. In my Case I've Created ApiInterface.java in client package and add the code

(Note : your base url should be your Producer app Url along with port number in which you gonna run your app in payara micro server)

In my case url : <http://localhost:8085/ProducerDB/rest/example>

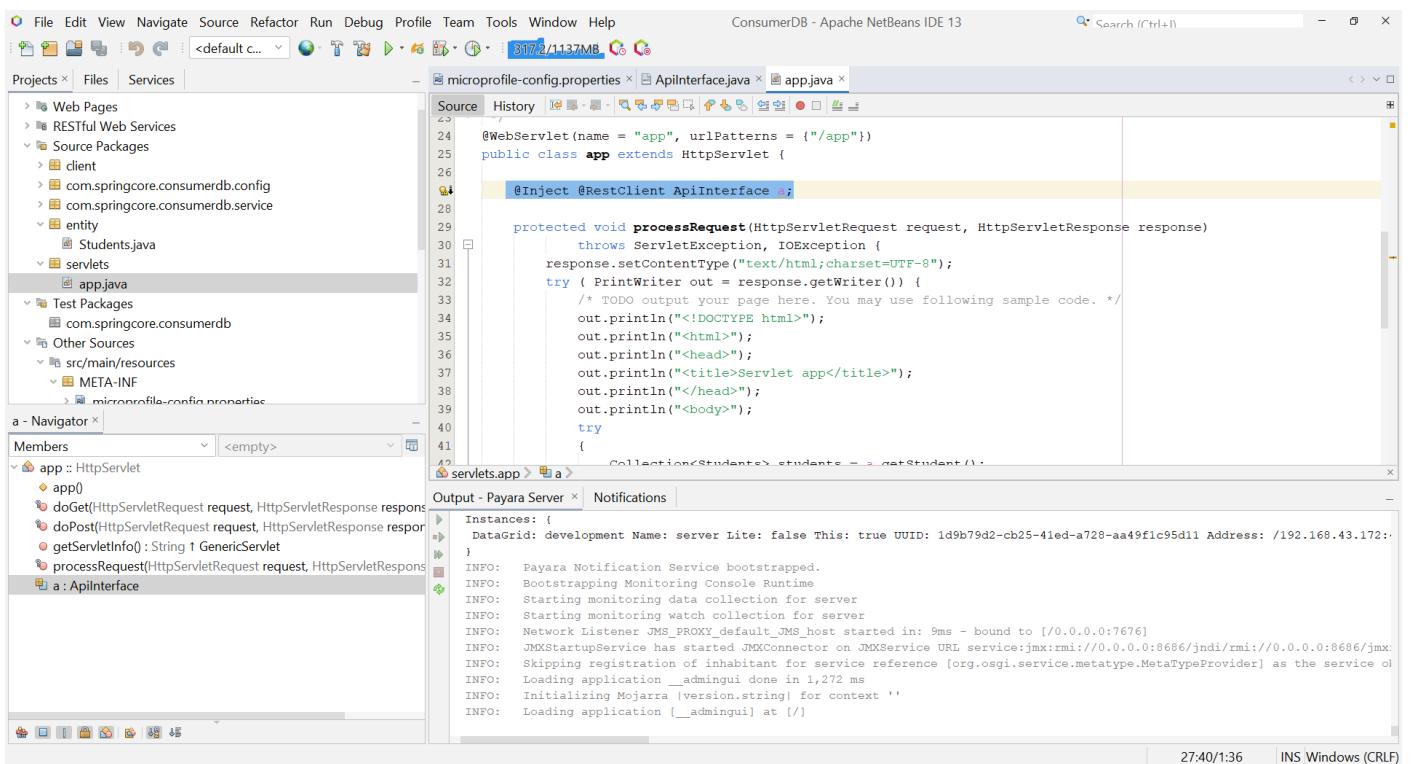


**35) Copy the entity package from Producer App and paste that package in your source package of Consumer App.**



**36) Create another package and add one Servlet. Write the Code mentioned in the below ss.**

In my case , I've create app.java servlet in servlets package.



File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help ConsumerDB - Apache NetBeans IDE 13 Search (Ctrl+F) 466.3/1137MB

Projects Files Services microprofile-config.properties ApInterface.java app.java

```

35     out.println("<html>");
36     out.println("<head>");
37     out.println("<title>Servlet app</title>");
38     out.println("</head>");
39     out.println("<body>");
40     try
41     {
42         Collection<Students> students = a.getStudent();
43         for(Students s:students)
44         {
45             out.println(s.getName());
46         }
47     }
48     catch(Exception e)
49     {
50         out.println("Something went wrong");
51         out.println("<br>");
52         out.println(e);
53     }

```

Members app :: HttpServlet

- app()
  - doGet(HttpServletRequest request, HttpServletResponse response)
  - doPost(HttpServletRequest request, HttpServletResponse response)
  - getServletInfo() : String
  - GenericServlet
  - processRequest(HttpServletRequest request, HttpServletResponse response)
- a : ApInterface

Output - Payara Server Notifications

```

Instances: [
  DataGrid: development Name: server Lite: false This: true UUID: 1d9b79d2-cb25-41ed-a728-aa49f1c95d11 Address: /192.168.43.172:48481
]
INFO: Payara Notification Service bootstrapped.
INFO: Bootstrapping Monitoring Console Runtime
INFO: Starting monitoring data collection for server
INFO: Starting monitoring watch collection for server
INFO: Network Listener JMS_PROXY_default_JMS_host started in: 9ms - bound to [/0.0.0.0:7676]
INFO: JMXStartupService has started JMXConnector on JMXService URL service:jmx:rmi://0.0.0.0:8686/jndi/rmi://0.0.0.0:8686/jmx: Skipping registration of inhabitant for service reference [org.osgi.service.metatype.MetaTypeProvider] as the service of JMXConnector is already registered
INFO: Loading application __admingui done in 1,272 ms
INFO: Initializing Mojarra [version.string] for context ''
INFO: Loading application [__admingui] at [/]

```

48:31 INS Windows (CRLF)

### 37) Right click on that project click on Clean and Build

File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help ConsumerDB - Apache NetBeans IDE 13 Search (Ctrl+F) 972.7/1137MB

Projects Files Services microprofile-config.properties ApInterface.java app.java

ClientApp New Build Clean and Build Build with Dependencies Clean Generate Javadoc

Run Debug Reload Profile Execute Java Shell Test Alt+F6 Run Selenium Tests

Run Maven Set Configuration Open POM Reload POM Open Required Projects Close Rename... Move... Copy... Delete Delete Find... Ctrl+F Inspect and Transform... Versioning History

```

35     out.println("<html>");
36     out.println("<head>");
37     out.println("<title>Servlet app</title>");
38     out.println("</head>");
39     out.println("<body>");
40     try
41     {
42         Collection<Students> students = a.getStudent();
43         for(Students s:students)
44         {
45             out.println(s.getName());
46         }
47     }
48     catch(Exception e)
49     {
50         out.println("Something went wrong");
51         out.println("<br>");
52         out.println(e);
53     }

```

Output - Payara Server Notifications

```

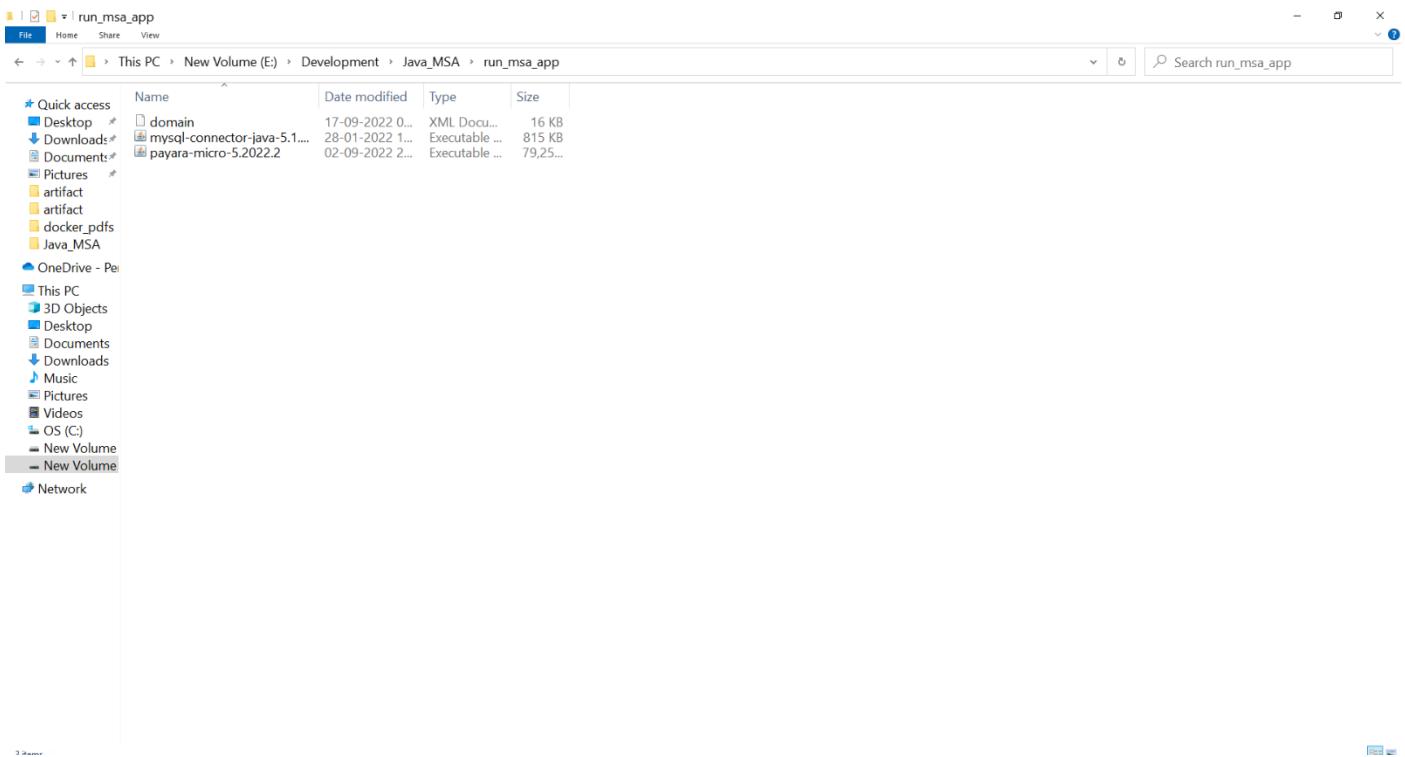
Instances: [
  DataGrid: development Name: server Lite: false This: true UUID: 1d9b79d2-cb25-41ed-a728-aa49f1c95d11 Address: /192.168.43.172:48481
]
INFO: Payara Notification Service bootstrapped.
INFO: Bootstrapping Monitoring Console Runtime
INFO: Starting monitoring data collection for server
INFO: Starting monitoring watch collection for server
INFO: Network Listener JMS_PROXY_default_JMS_host started in: 9ms - bound to [/0.0.0.0:7676]
INFO: JMXStartupService has started JMXConnector on JMXService URL service:jmx:rmi://0.0.0.0:8686/jndi/rmi://0.0.0.0:8686/jmx: Skipping registration of inhabitant for service reference [org.osgi.service.metatype.MetaTypeProvider] as the service of JMXConnector is already registered
INFO: Loading application __admingui done in 1,272 ms
INFO: Initializing Mojarra [version.string] for context ''
INFO: Loading application [__admingui] at [/]

```

48:31 INS

**Alright Fine! Let's move on deployment part, but before moving on deployment I want you to make some changes in domain.xml according to your jdbc connection**

- 38) Create one blank folder and put payara-microprofile.jar file and mysql-connector jar file along with domain.xml file



- 39) Open domain.xml file and make some changes denoted by red underline

\*E:\Development\Java\_MSA\run\_msa\_app\domain.xml - Notepad++

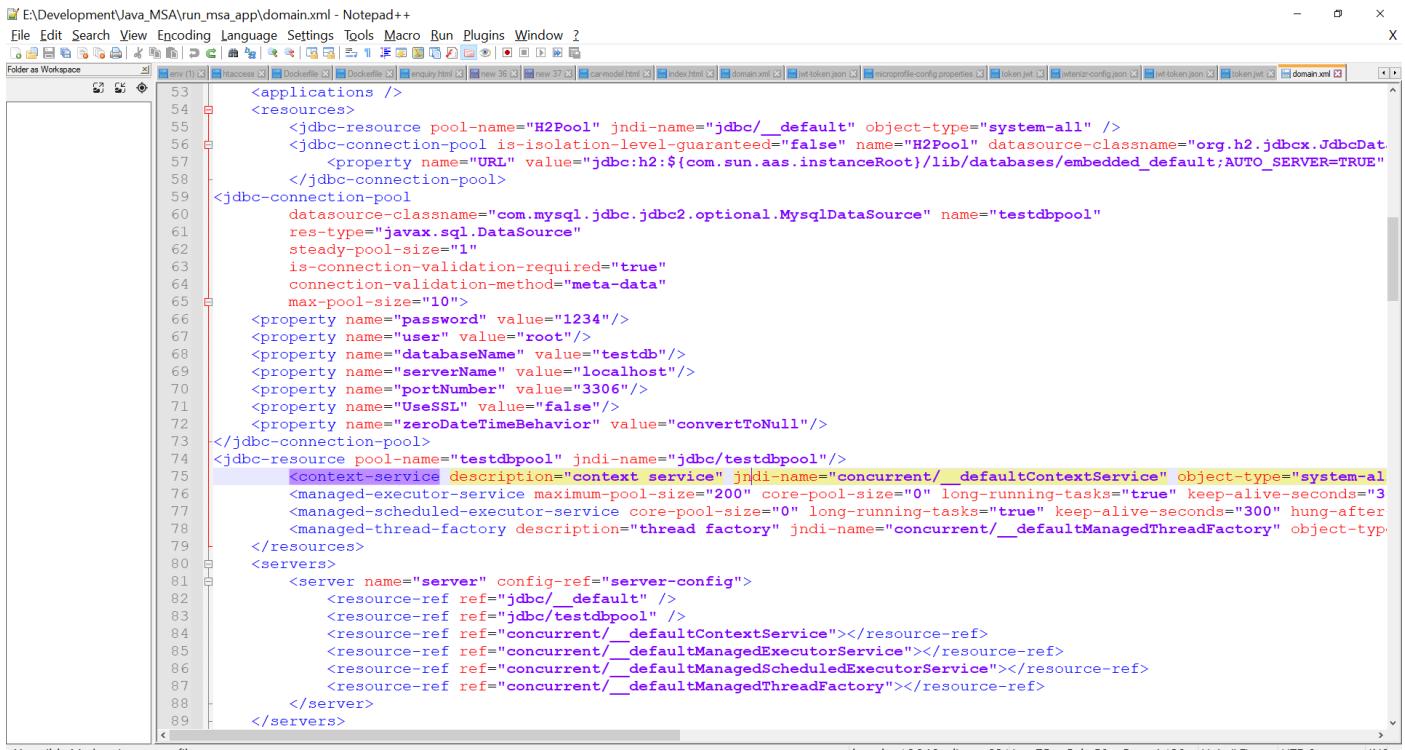
```

File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?
Folder as Workspace
domain.xml 1 access Dockfile Dockfile enquiry.html new 36 index.html domain.xml wallet.json microprofile-config.properties token.wt interzr-config.json token.json wallet.json domain.xml

53     <applications />
54   <resources>
55     <jdbc-resource pool-name="H2Pool" jndi-name="jdbc/_default" object-type="system-all" />
56     <jdbc-connection-pool is-isolation-level-guaranteed="false" name="H2pool" datasource-classname="org.h2.jdbc.JdbcDataSource" />
57       <property name="URL" value="jdbc:h2:${com.sun.aas.instanceRoot}/lib/databases/embedded_default;AUTO_SERVER=TRUE" />
58   </jdbc-connection-pool>
59   <jdbc-connection-pool
60     datasource-classname="com.mysql.jdbc.jdbc2.optional.MysqlDataSource" name="jdbc_connection_pool"
61     res-type="javax.sql.DataSource"
62     steady-pool-size="1"
63     is-connection-validation-required="true"
64     connection-validation-method="meta-data"
65     max-pool-size="10">
66       <property name="password" value="mysql_password"/>
67       <property name="user" value="mysql_username"/>
68       <property name="databaseName" value="databaseName"/>
69       <property name="serverName" value="localhost"/>
70       <property name="portNumber" value="3306"/>
71       <property name="useSSL" value="false"/>
72       <property name="zeroDateTimeBehavior" value="convertToNull"/>
73   </jdbc-connection-pool>
74   <jdbc-resource pool-name="jdbc_connection_pool" jndi-name="jdbc_resource_name" />
75     <context-service description="context service" jndi-name="concurrent/_defaultContextService" object-type="system-all" />
76     <managed-executor-service maximum-pool-size="200" core-pool-size="0" long-running-tasks="true" keep-alive-seconds="3" />
77     <managed-scheduled-executor-service core-pool-size="0" long-running-tasks="true" keep-alive-seconds="300" hung-after="0" />
78     <managed-thread-factory description="thread factory" jndi-name="concurrent/_defaultManagedThreadFactory" object-type="system-all" />
79   </resources>
80   <servers>
81     <server name="server" config-ref="server-config">
82       <resource-ref ref="jdbc/_default" />
83       <resource-ref ref="jdbc_resource_name" />
84       <resource-ref ref="concurrent/_defaultContextService"></resource-ref>
85       <resource-ref ref="concurrent/_defaultManagedExecutorService"></resource-ref>
86       <resource-ref ref="concurrent/_defaultManagedScheduledExecutorService"></resource-ref>
87       <resource-ref ref="concurrent/_defaultManagedThreadFactory"></resource-ref>
88     </server>
89   </servers>

```

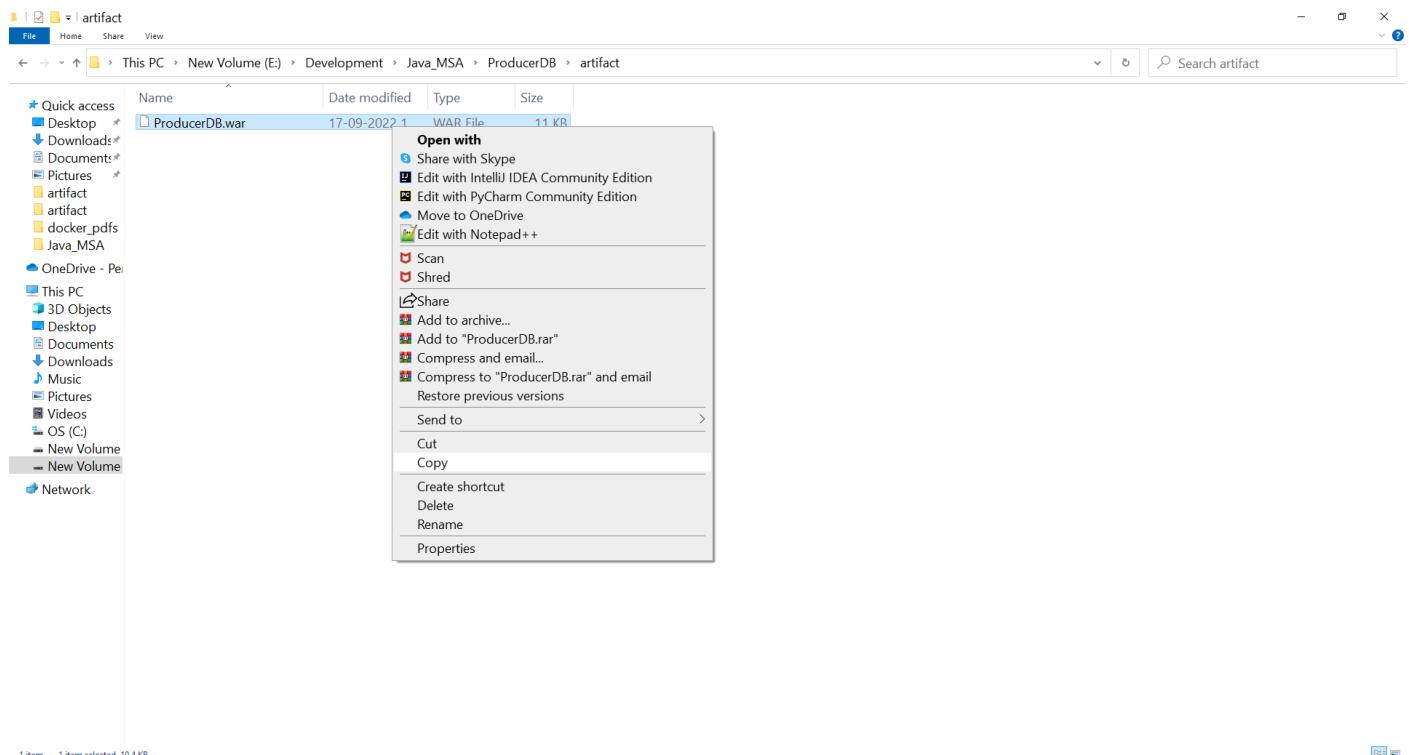
In my case output would be

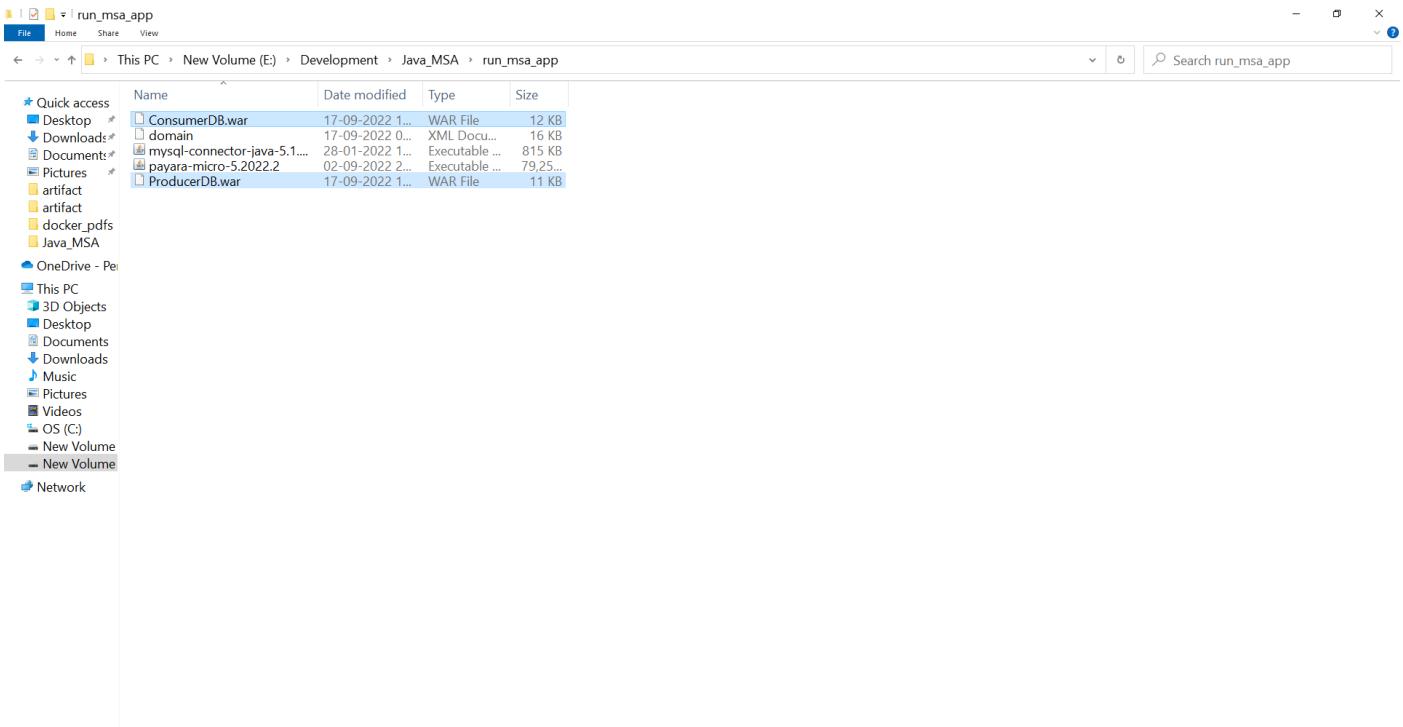
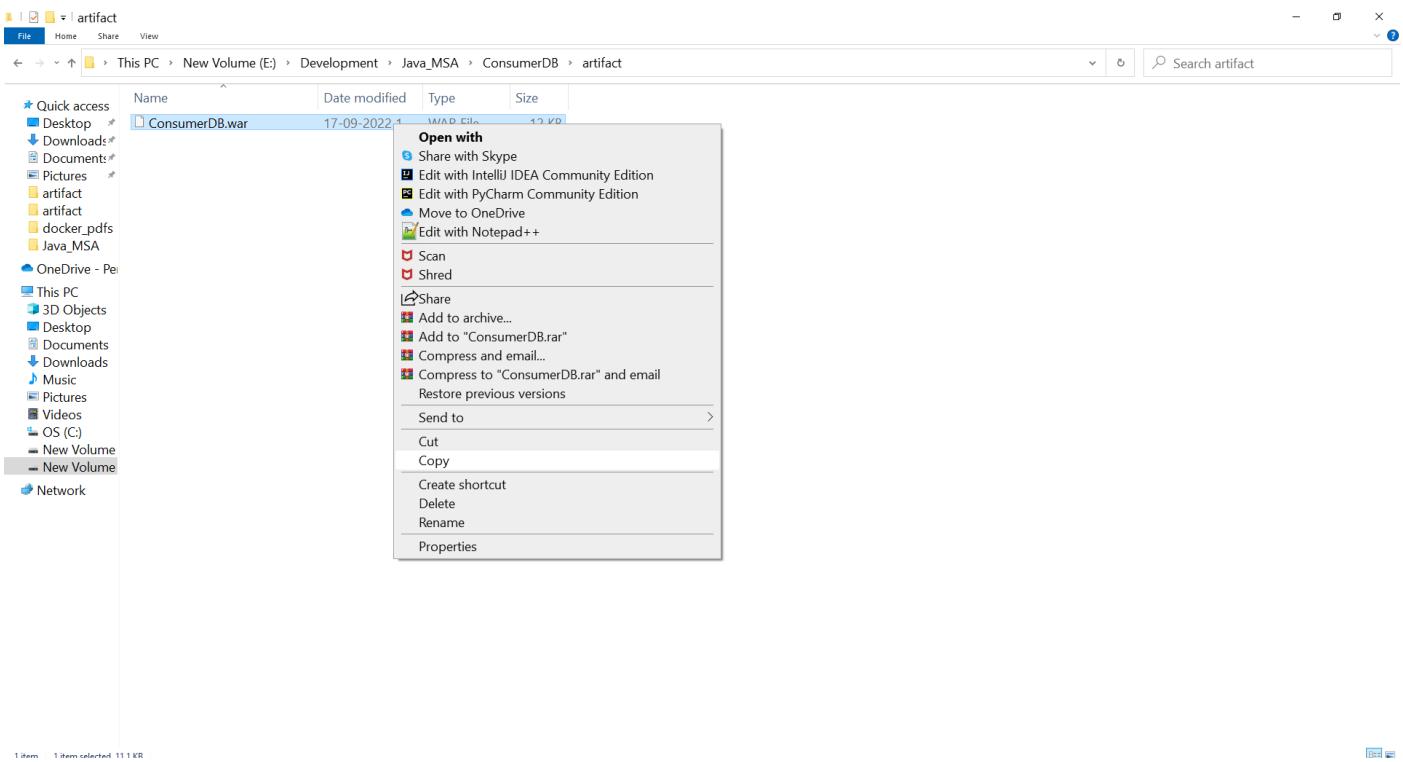


```
<applications />
<resources>
    <jdbc-resource pool-name="H2Pool" jndi-name="jdbc/_default" object-type="system-all" />
    <jdbc-connection-pool is-isolation-level-guaranteed="false" name="H2Pool" datasource-classname="org.h2.jdbcx.JdbcDataSource">
        <property name="URL" value="jdbc:h2:${com.sun.aas.instanceRoot}/lib/databases/embedded_default;AUTO_SERVER=TRUE" />
    </jdbc-connection-pool>
    <jdbc-connection-pool
        datasource-classname="com.mysql.jdbc.jdbc2.optional.MysqlDataSource" name="testdbpool"
        res-type="javax.sql.DataSource"
        steady-pool-size="1"
        is-connection-validation-required="true"
        connection-validation-method="meta-data"
        max-pool-size="10">
        <property name="password" value="1234"/>
        <property name="user" value="root"/>
        <property name="databaseName" value="testdb"/>
        <property name="serverName" value="localhost"/>
        <property name="portNumber" value="3306"/>
        <property name="UseSSL" value="false"/>
        <property name="zeroDateTimeBehavior" value="convertToNull"/>
    </jdbc-connection-pool>
    <jdbc-resource pool-name="testdbpool" jndi-name="jdbc/testdbpool"/>
        <context-service description="context service" jndi-name="concurrent/_defaultContextService" object-type="system-all">
            <managed-executor-service maximum-pool-size="200" core-pool-size="0" long-running-tasks="true" keep-alive-seconds="300" />
            <managed-scheduled-executor-service core-pool-size="0" long-running-tasks="true" keep-alive-seconds="300" hung-after="10000" />
            <managed-thread-factory description="thread factory" jndi-name="concurrent/_defaultManagedThreadFactory" object-type="system-all" />
        </context-service>
    </resources>
    <servers>
        <server name="server" config-ref="server-config">
            <resource-ref ref="jdbc/_default" />
            <resource-ref ref="jdbc/testdbpool" />
            <resource-ref ref="concurrent/_defaultContextService" />
            <resource-ref ref="concurrent/_defaultManagedExecutorService" />
            <resource-ref ref="concurrent/_defaultManagedScheduledExecutorService" />
            <resource-ref ref="concurrent/_defaultManagedThreadFactory" />
        </server>
    </servers>
</applications>
```

40) Go to your Producer App Directory->artifact and copy war file and paste that war file in the blank folder.

Perform same for Consumer App copy war file and paste that war file in the blank folder. Blank folder that you've created where you've domain.xml, mysql-connector and payara-microprofile jar file.



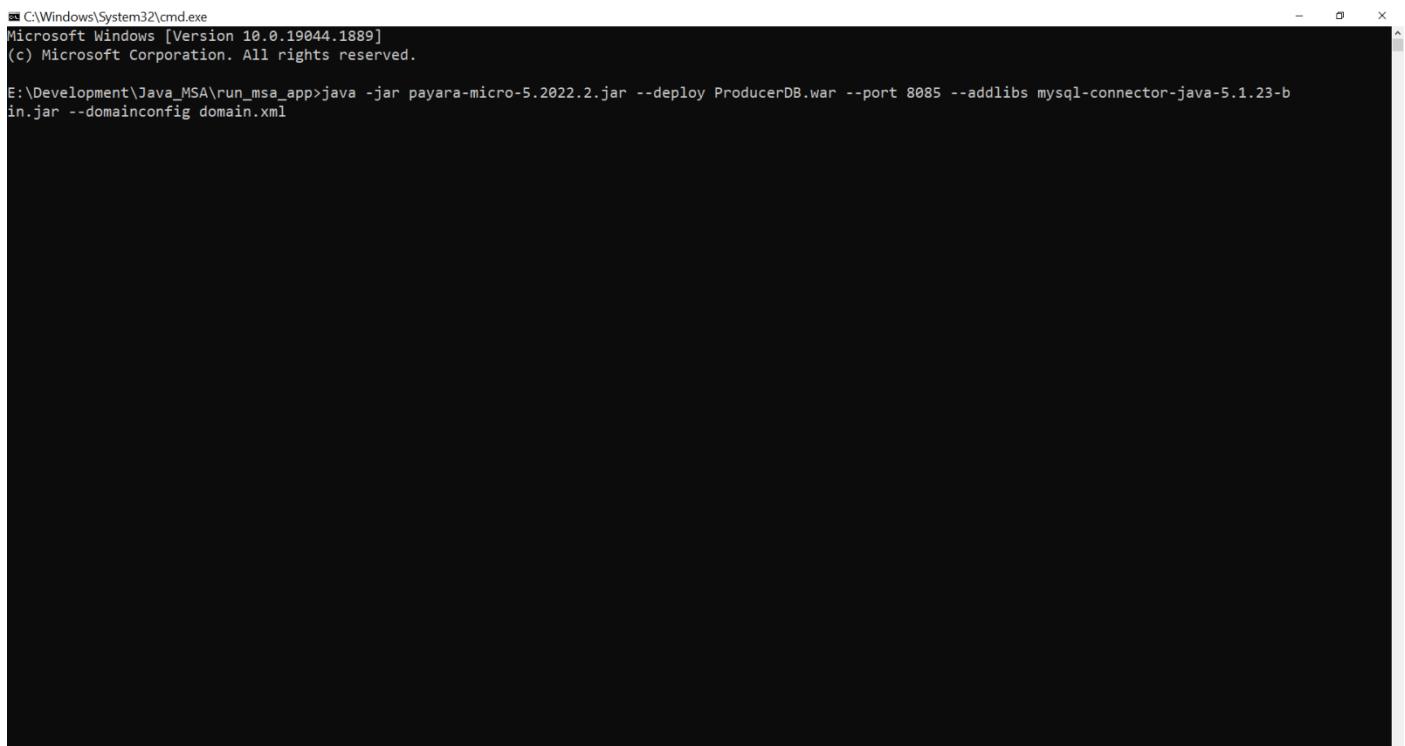


- 41) Open the terminal/Command Prompt on that location and Let's deploy the Producer App first.  
Before careful while writing the command.

In My case command :

```
java -jar payara-micro-5.2022.2.jar --deploy ProducerDB.war --port 8085 --addlibs mysql-connector-java-5.1.23-bin.jar --domainconfig domain.xml
```

I gonna run this Producer app on Port Number 8085, You have to run the Producer App on that port number which you mentioned in step no. 34



```
C:\Windows\System32\cmd.exe
Microsoft Windows [Version 10.0.19044.1889]
(c) Microsoft Corporation. All rights reserved.

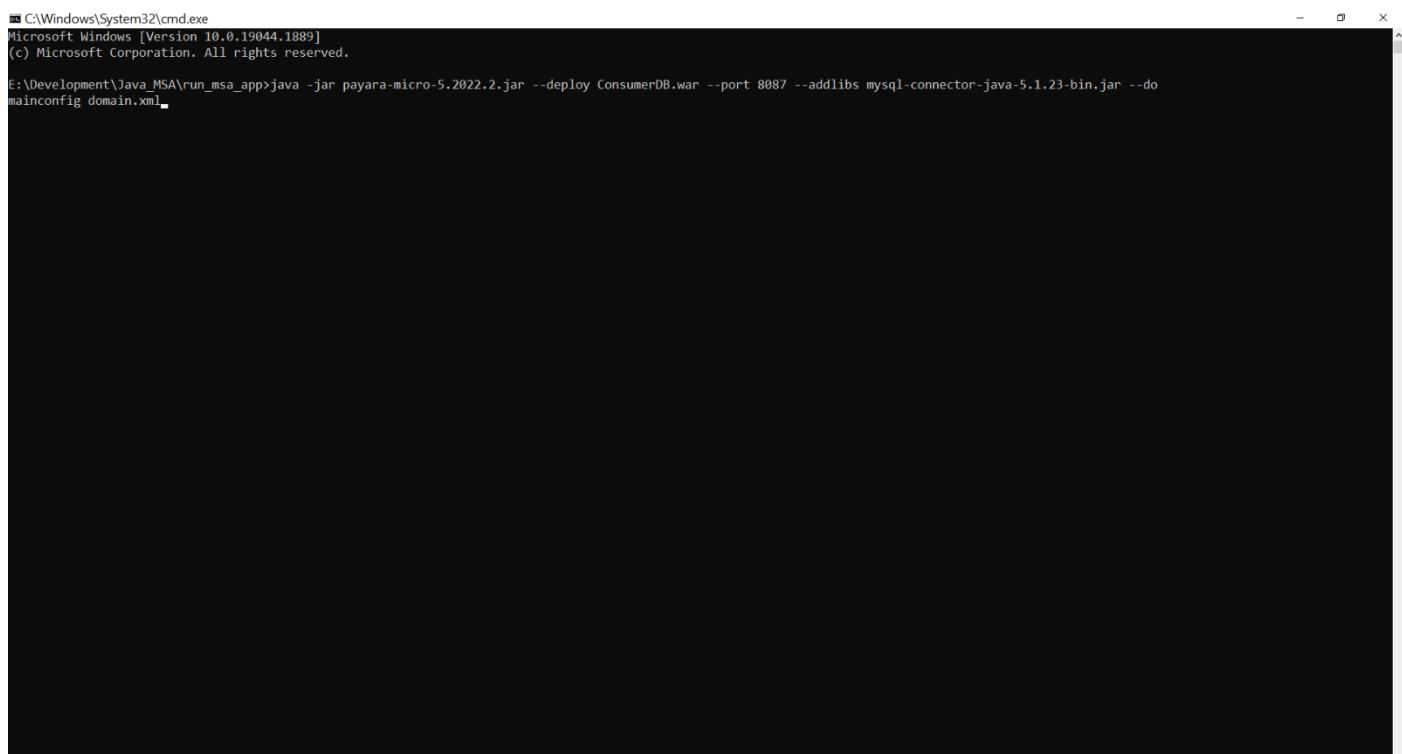
E:\Development\Java_MSA\run_msa_app>java -jar payara-micro-5.2022.2.jar --deploy ProducerDB.war --port 8085 --addlibs mysql-connector-java-5.1.23-bin.jar --domainconfig domain.xml
```

- 42) Wait and Watch, Open another terminal/Command prompt don't close earlier terminal/Command Prompt.  
And fire the same command again for consumer app

In my case Command would

```
java -jar payara-micro-5.2022.2.jar --deploy ConsumerDB.war --port 8087 --addlibs mysql-connector-java-5.1.23-bin.jar --domainconfig domain.xml
```

Here You can select any port number for Consumer App



A screenshot of a Windows Command Prompt window titled 'C:\Windows\System32\cmd.exe'. The window shows the following text:  
Microsoft Windows [Version 10.0.19044.1889]  
(c) Microsoft Corporation. All rights reserved.  
E:\Development\Java\_MSA\run\_msa\_app>java -jar payara-micro-5.2022.2.jar --deploy ConsumerDB.war --port 8087 --addlibs mysql-connector-java-5.1.23-bin.jar --domainconfig domain.xml

**Alright Both terminal should give some script as output when it is done Open the browser and Enter the following urls**

For Producer App my URL would : <http://localhost:8085/ProducerDB/rest/example/student>

It should give Unauthorized status

URL : http://localhost:port\_number/your\_project\_name/rest/example/your\_path

A screenshot of a web browser window. The title bar says "Payara Micro #badassfish - Error". The address bar shows "localhost:8085/ProducerDB/rest/example/student". The main content area displays an "HTTP Status 401 - Unauthorized" page. It includes fields for "Type" (Status report), "message" (Unauthorized), and "description" (This request requires HTTP authentication.). At the bottom of the page is a header "Payara Micro #badassfish".

For consumer Enter the URL : <http://localhost:8087/ConsumerDB/app>

URL : http://localhost:port\_number/project\_name/url\_of\_your\_servlet

A screenshot of a web browser window. The title bar says "Payara Micro #badassfish - Error" and "Servlet app". The address bar shows "localhost:8087/ConsumerDB/app". The main content area is completely blank. At the bottom left, it says "Dhawal Parmar".

*Happy Coding!*