Lesson 01 Demo 02

Create a RESTful Web Service

Objective: To create and consume a RESTful web service with Jersey

Tool Required: Eclipse IDE

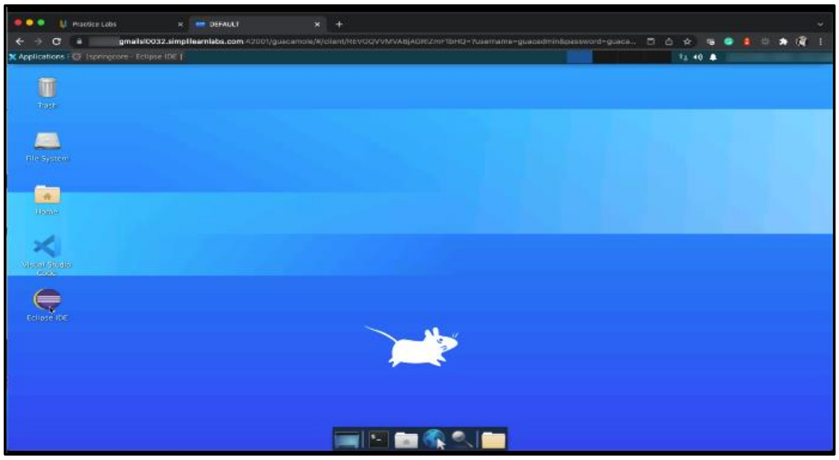
Pre-requisites: None

Steps to be followed:

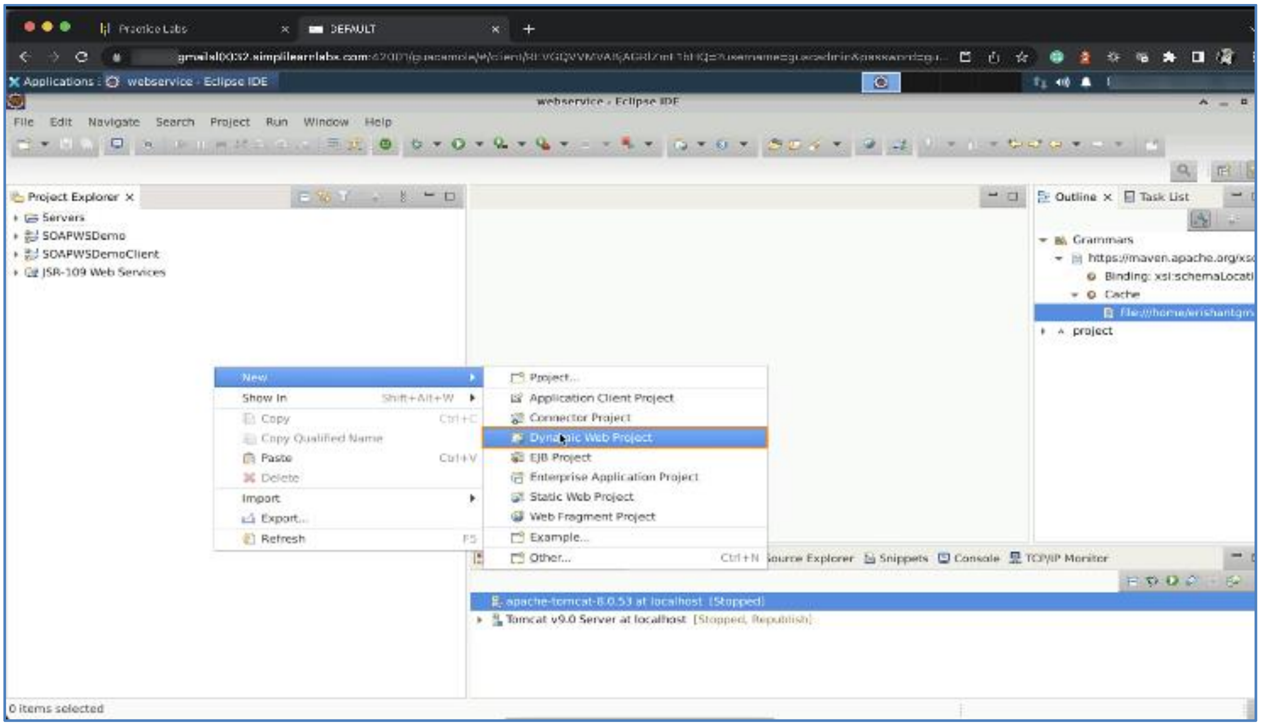
1. Creating a dynamic project
2. Setting up pom.xml and creating the service class
3. Creating an HTML file and mapping a Servlet in web.xml
4. Creating a JSP file and mapping it to a Java client file

Step 1: Creating a dynamic project

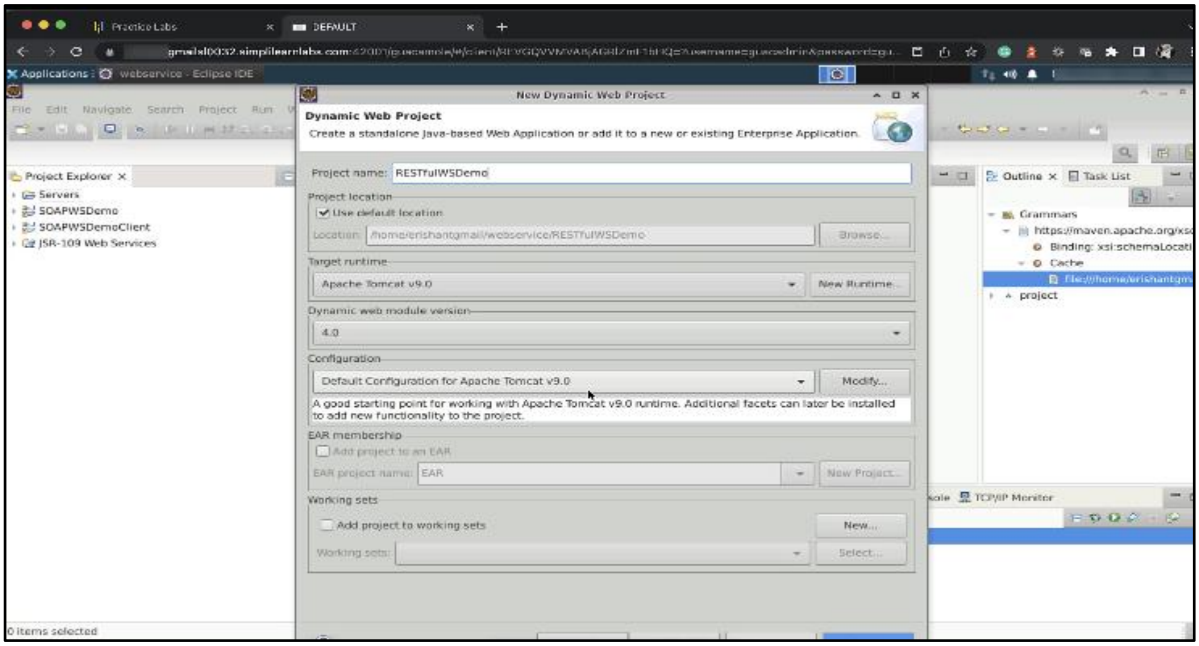
1.1 Open **Eclipse IDE** from your desktop



1.2 Right-click on **Project Explorer** and select **New -> Dynamic Web Project**



1.3 Provide the name to the project as **RESTfulWSDemo**


New Dynamic Web Project

Dynamic Web Project

Create a standalone Java-based Web Application or add it to a new or existing Enterprise Application.

Project name: RESTfulWSDemo

Project location

☑ Use default location

Location: /home/erishantgmail/webservice/RESTfulWSDemo

Target runtime: Apache Tomcat v9.0

Dynamic web module version: 4.0

Configuration: Default Configuration for Apache Tomcat v9.0

A good starting point for working with Apache Tomcat v9.0 runtime. Additional facets can later be installed to add new functionality to the project.

EAR membership

☐ Add project to an EAR

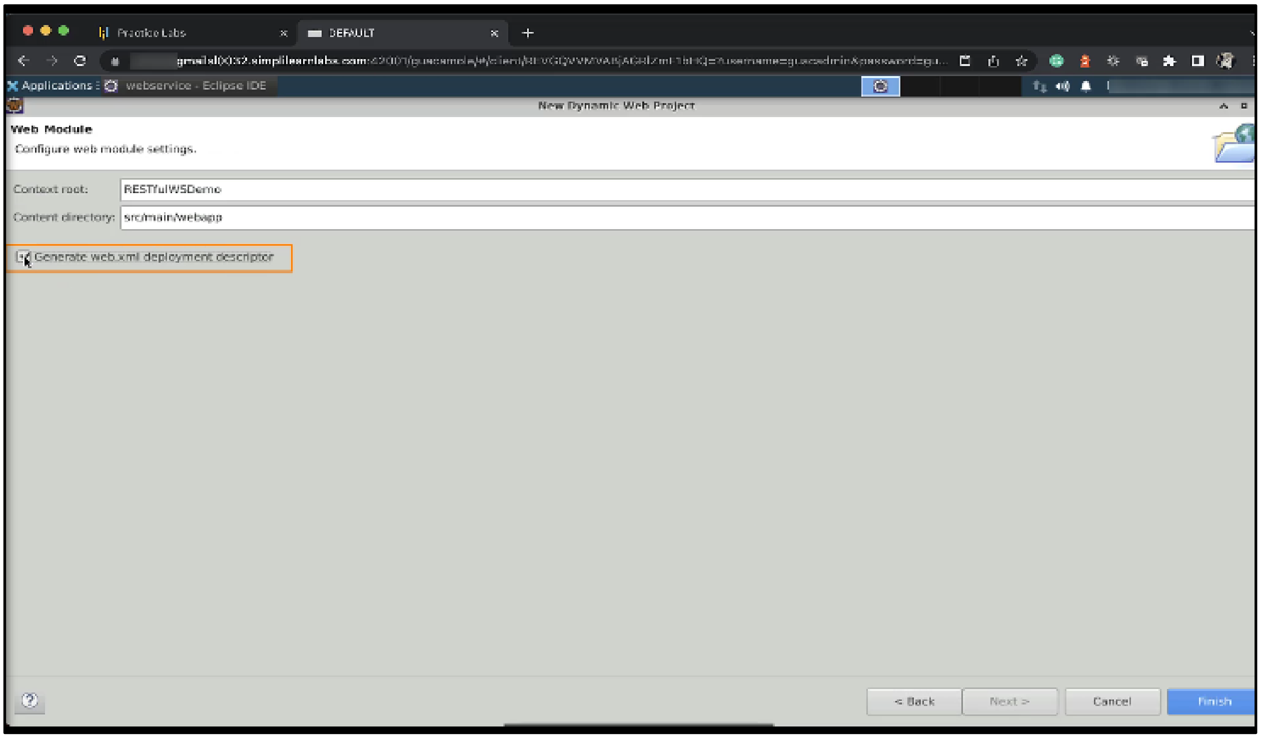
EAR project name: EAR

Working sets

☐ Add project to working sets

Working sets:

1.4 Check on **Generate web.xml deploymeny descriptor** and click on **Finish**



Web Module

Configure web module settings.

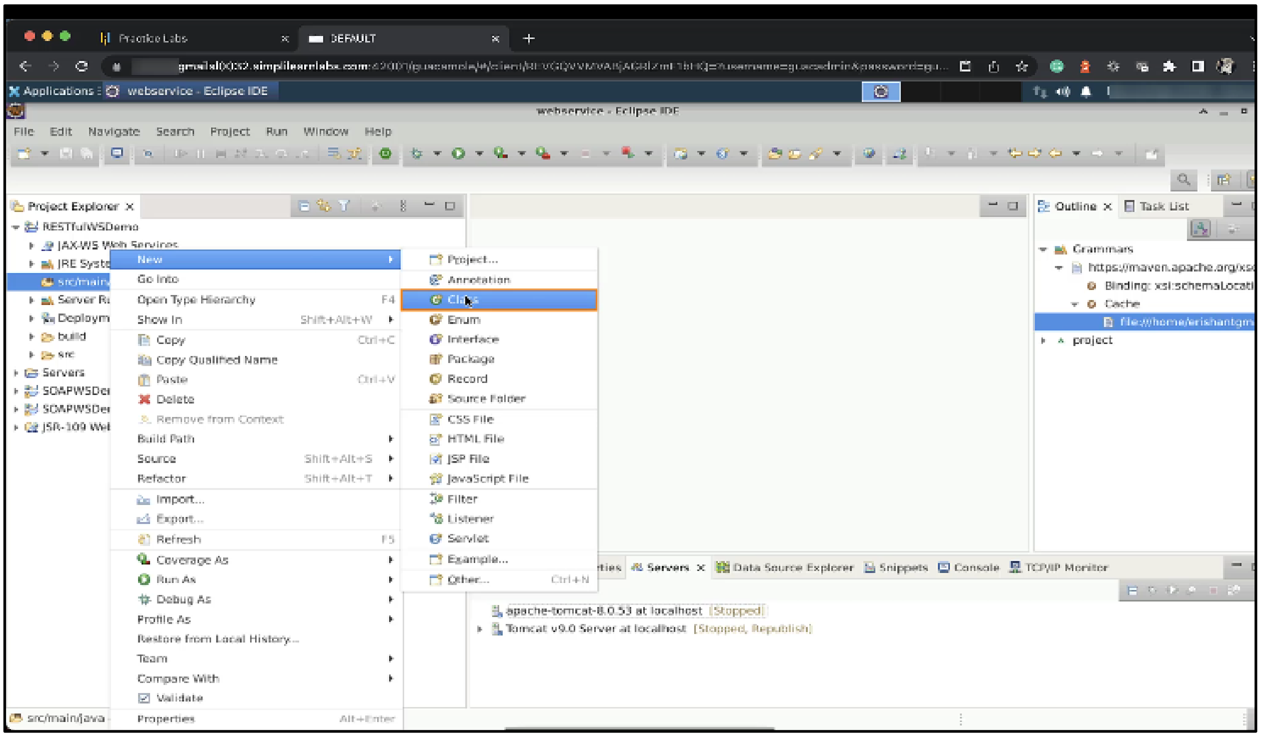
Context root: RESTfulWSDemo

Content directory: src/main/webapp

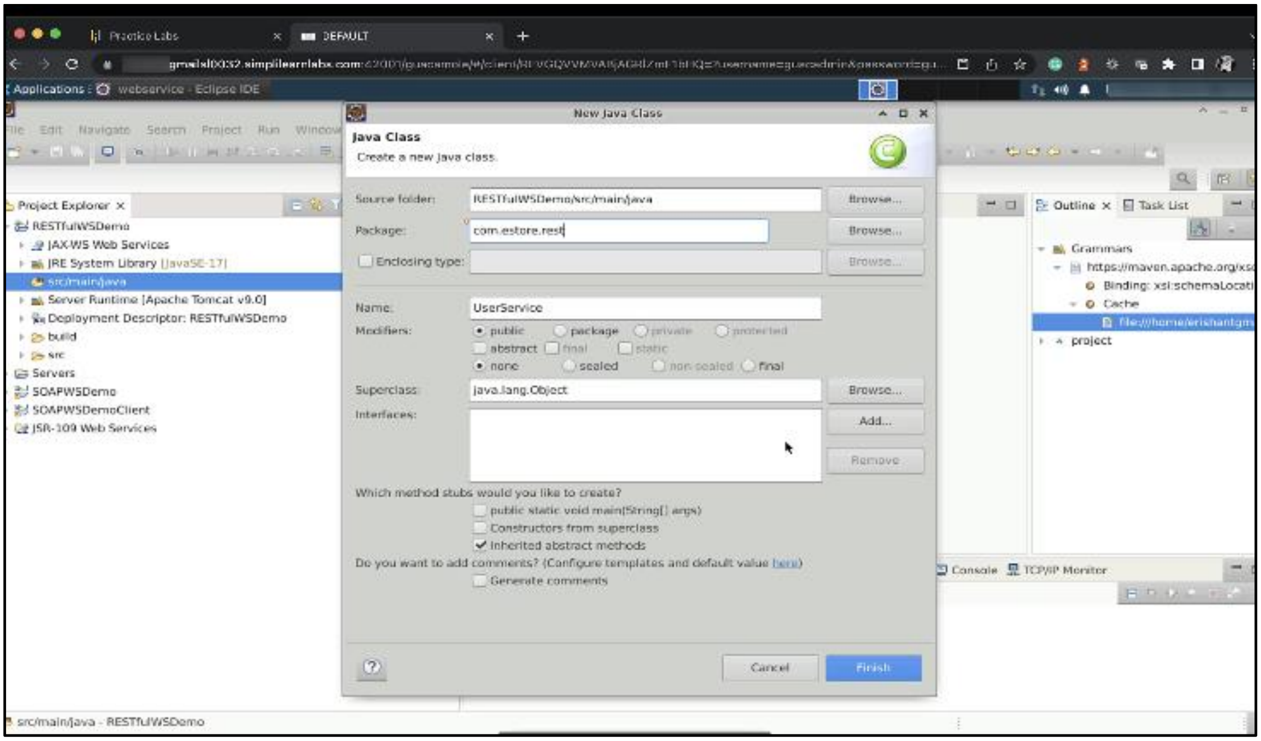
☑ Generate web.xml deployment descriptor

Step 2: Setting up pom.xml and creating the service class

2.1 Right-click on the **src** folder and select **New -> Class**



2.2 Provide a class name as **UserService** and package name as **com.estore.rest** and click on **Finish**


Practice Labs
DEFAULT
Applications : webservice - Eclipse IDE
File Edit Navigate Search Project Run Window
Project Explorer
RESTfulWSDemo
JAX-WS Web Services
JRE System Library [JavaSE-17]
src/main/java
Server Runtime [Apache Tomcat v9.0]
Deployment Descriptor: RESTfulWSDemo
build
src
Servers
SOAPWSDemo
SOAPWSDemoClient
JSR-109 Web Services
Outline
Task List
Grammars
https://maven.apache.org/xsd
Binding: xsi:schemaLocation
Cache
file://home/
project
Console
TCP/IP Monitor
src/main/java - RESTfulWSDemo
Browse...
Add...
Remove
Cancel
Finish

New Java Class

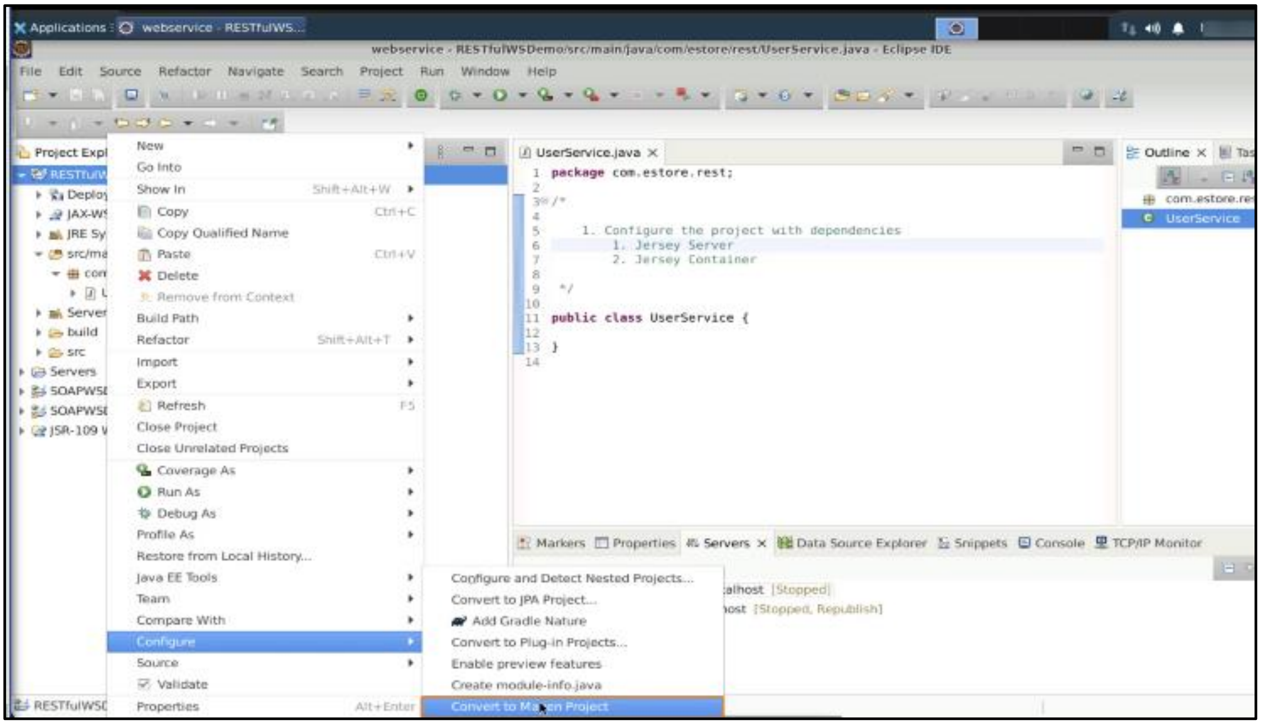
Java Class
Create a new Java class.

Source folder: RESTfulWSDemo/src/main/java
Package: com.estore.rest
□ Enclosing type:

Name: UserService
Modifiers: ● public ○ package ○ private ○ protected
□ abstract □ final □ static
● none ○ sealed ○ non-sealed ○ final
Superclass: java.lang.Object
Interfaces:

Which method stubs would you like to create?
□ public static void main(String[] args)
□ Constructors from superclass
☑ Inherited abstract methods
Do you want to add comments? (Configure templates and default value here)
□ Generate comments

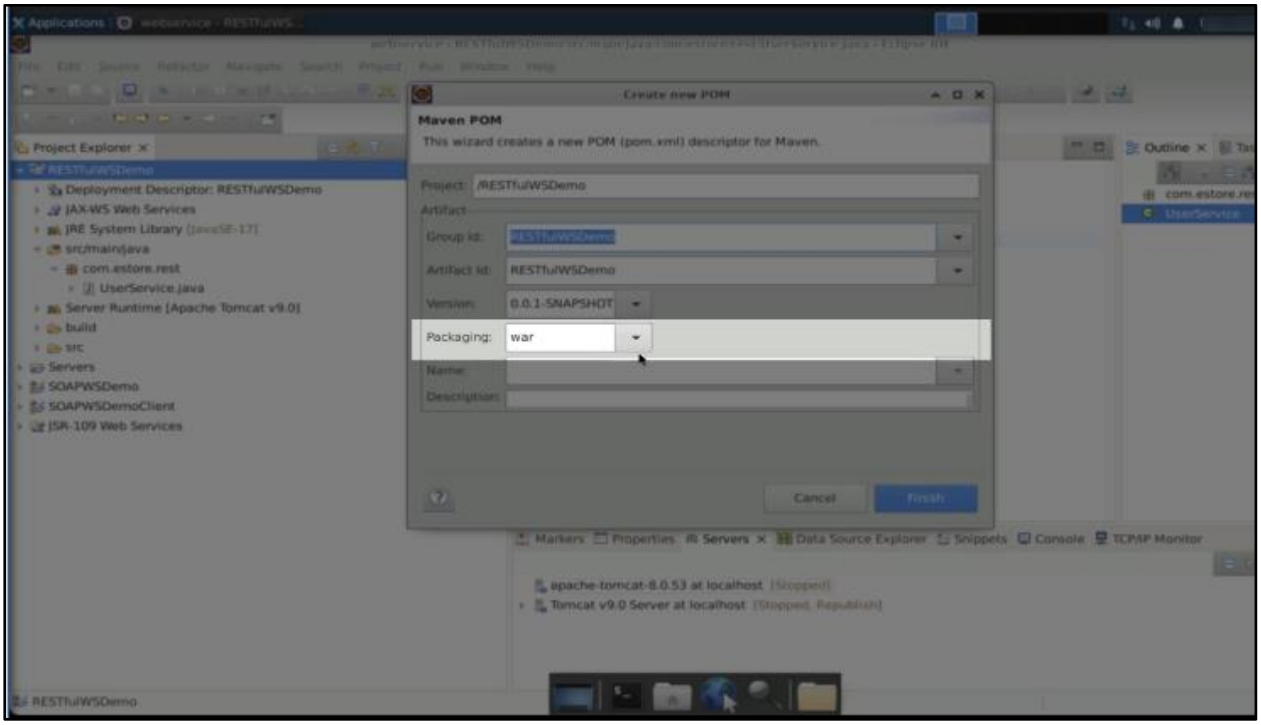
2.3 Right-click on project name and select **Configure -> Convert to Maven Project**


webservice - RESTfulWSDemo/src/main/java/com/estore/rest/UserService.java - Eclipse IDE

UserService.java

```
package com.estore.rest;

/*

    1. Configure the project with dependencies
        1. Jersey Server
        2. Jersey Container

 */

public class UserService {

}
```

2.4 In the **Create new** POM dialogue box, select **Packaging** as **war** and click **Finish**


Create new POM

Maven POM

This wizard creates a new POM (pom.xml) descriptor for Maven.

Project: /RESTfulWSDemo

Artifact

Group Id: RESTfulWSDemo

Artifact Id: RESTfulWSDemo

Version: 0.0.1-SNAPSHOT

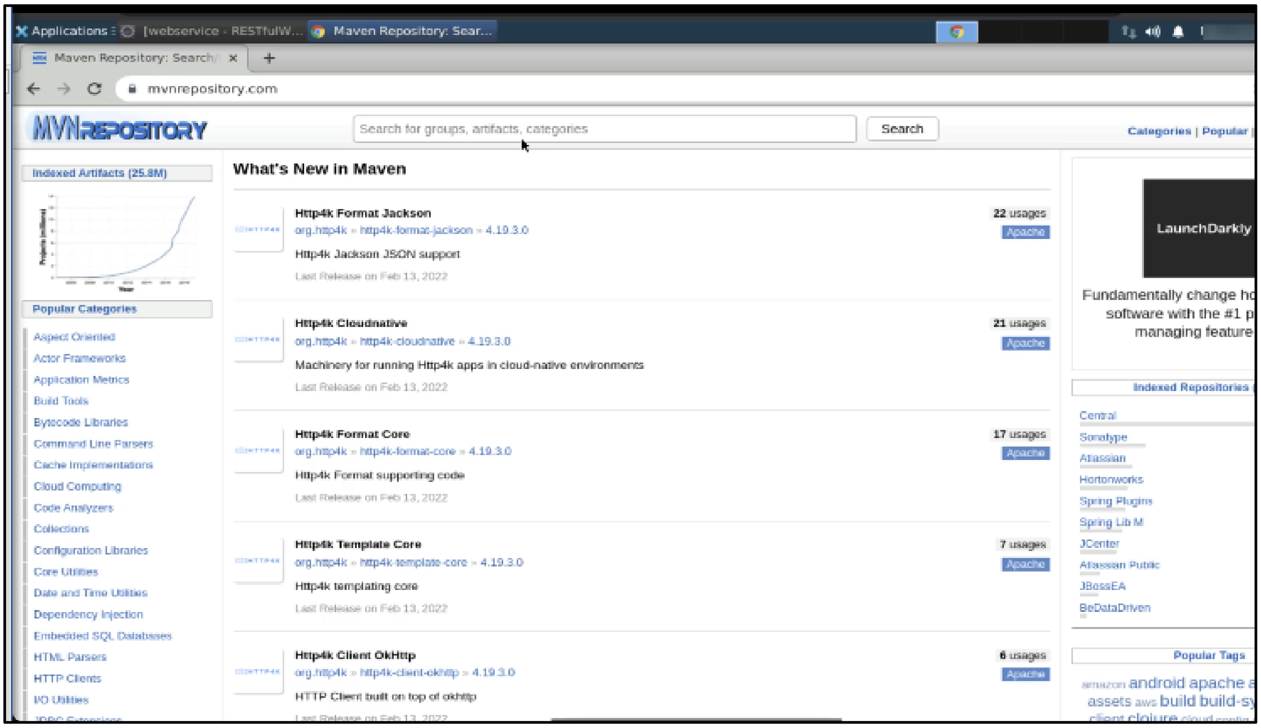
Packaging: war

Name:

Description:

Cancel
Finish
Project Explorer
RESTfulWSDemo
Deployment Descriptor: RESTfulWSDemo
JAX-WS Web Services
JRE System Library [JavaSE-17]
src/main/java
com.estore.rest
UserService.java
Server Runtime [Apache Tomcat v9.0]
build
src
Servers
SOAPWSDemo
SOAPWSDemoClient
JSR-109 Web Services
apache-tomcat-8.0.53 at localhost [Stopped]
Tomcat v9.0 Server at localhost [Stopped, Republish]

2.5 Open the **mavenrepository.com** in your browser



What's New in Maven

Http4k Format Jackson — 22 usages — Apache
org.http4k » http4k-format-jackson » 4.19.3.0
Http4k Jackson JSON support
Last Release on Feb 13, 2022

Http4k Cloudnative — 21 usages — Apache
org.http4k » http4k-cloudnative » 4.19.3.0
Machinery for running Http4k apps in cloud-native environments
Last Release on Feb 13, 2022

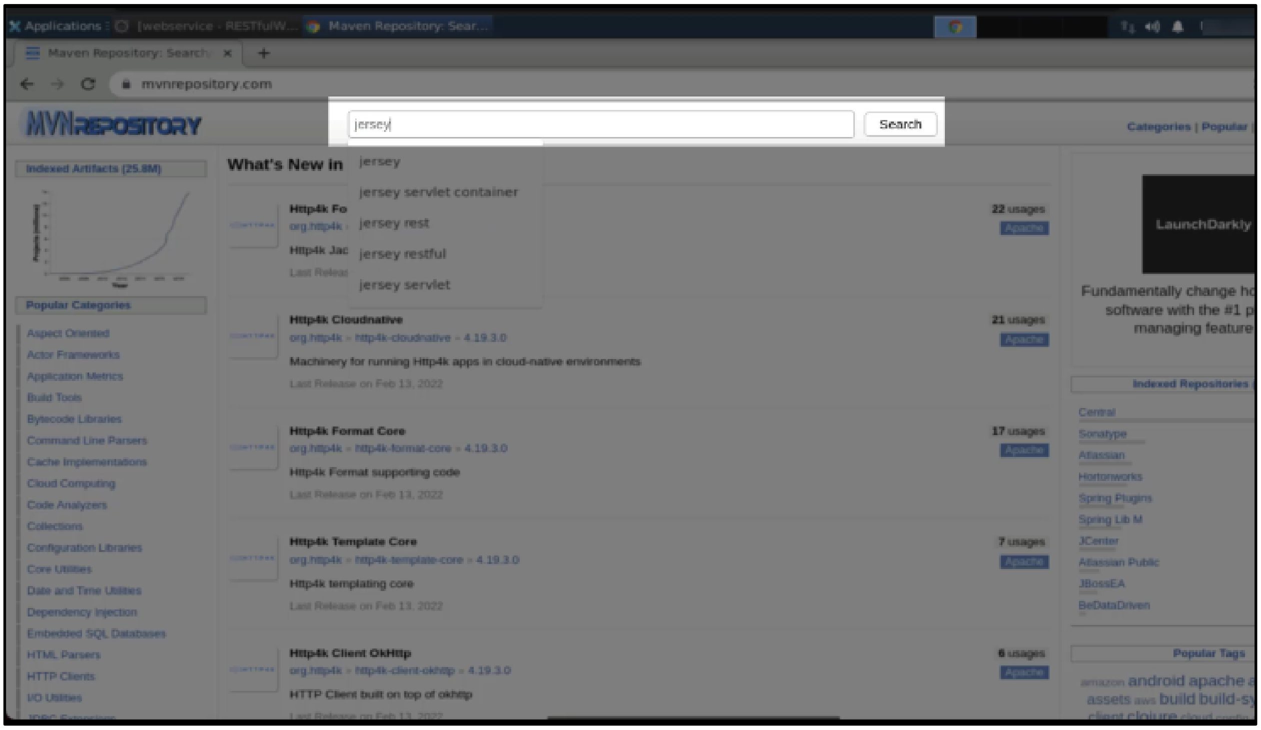
Http4k Format Core — 17 usages — Apache
org.http4k » http4k-format-core » 4.19.3.0
Http4k Format supporting code
Last Release on Feb 13, 2022

Http4k Template Core — 7 usages — Apache
org.http4k » http4k-template-core » 4.19.3.0
Http4k templating core
Last Release on Feb 13, 2022

Http4k Client OkHttp — 6 usages — Apache
org.http4k » http4k-client-okhttp » 4.19.3.0
HTTP Client built on top of okhttp
Last Release on Feb 13, 2022

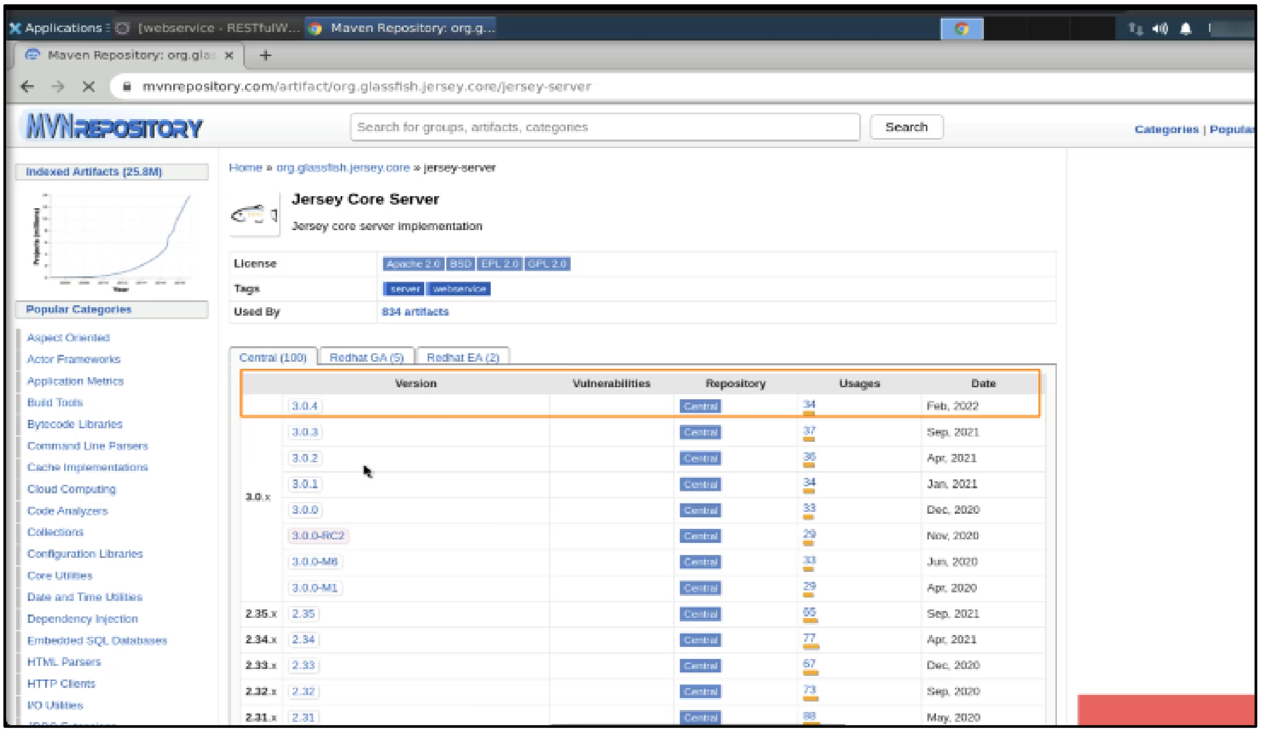
Fundamentally change how you deliver software with the #1 platform for managing feature

2.6 Search for **Jersey** in the search bar as shown below:



Search field: jersey

2.7 Select the version from the **Jersey Core Server** as **3.0.4** as shown below:



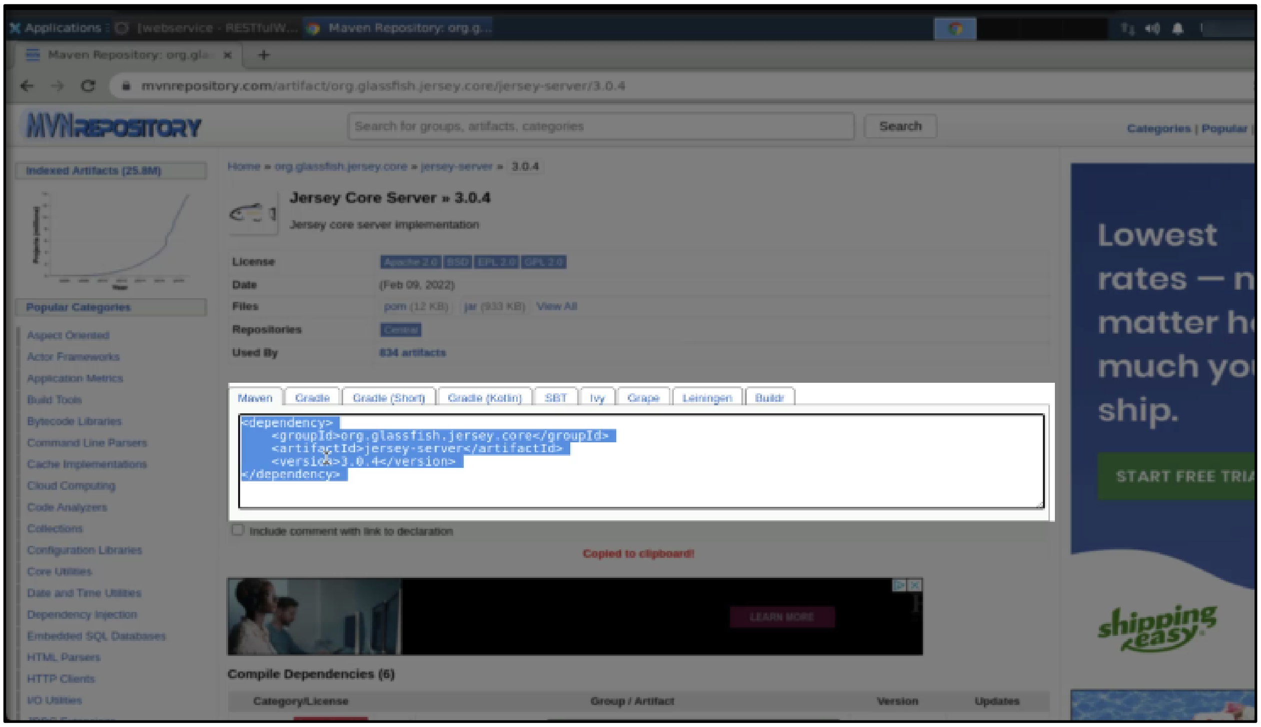
Jersey Core Server

Jersey core server implementation

License	Apache 2.0 BSD EPL 2.0 GPL 2.0
Tags	server webservice
Used By	834 artifacts

	Version	Vulnerabilities	Repository	Usages	Date
	3.0.4		Central	34	Feb, 2022
3.0.x	3.0.3		Central	37	Sep, 2021
	3.0.2		Central	36	Apr, 2021
	3.0.1		Central	34	Jan, 2021
	3.0.0		Central	33	Dec, 2020
	3.0.0-RC2		Central	29	Nov, 2020
	3.0.0-M6		Central	33	Jun, 2020
	3.0.0-M1		Central	29	Apr, 2020
2.35.x	2.35		Central	65	Sep, 2021
2.34.x	2.34		Central	77	Apr, 2021
2.33.x	2.33		Central	67	Dec, 2020
2.32.x	2.32		Central	73	Sep, 2020
2.31.x	2.31		Central	88	May, 2020

2.8 Copy the dependency script



Jersey Core Server » 3.0.4

Jersey core server implementation

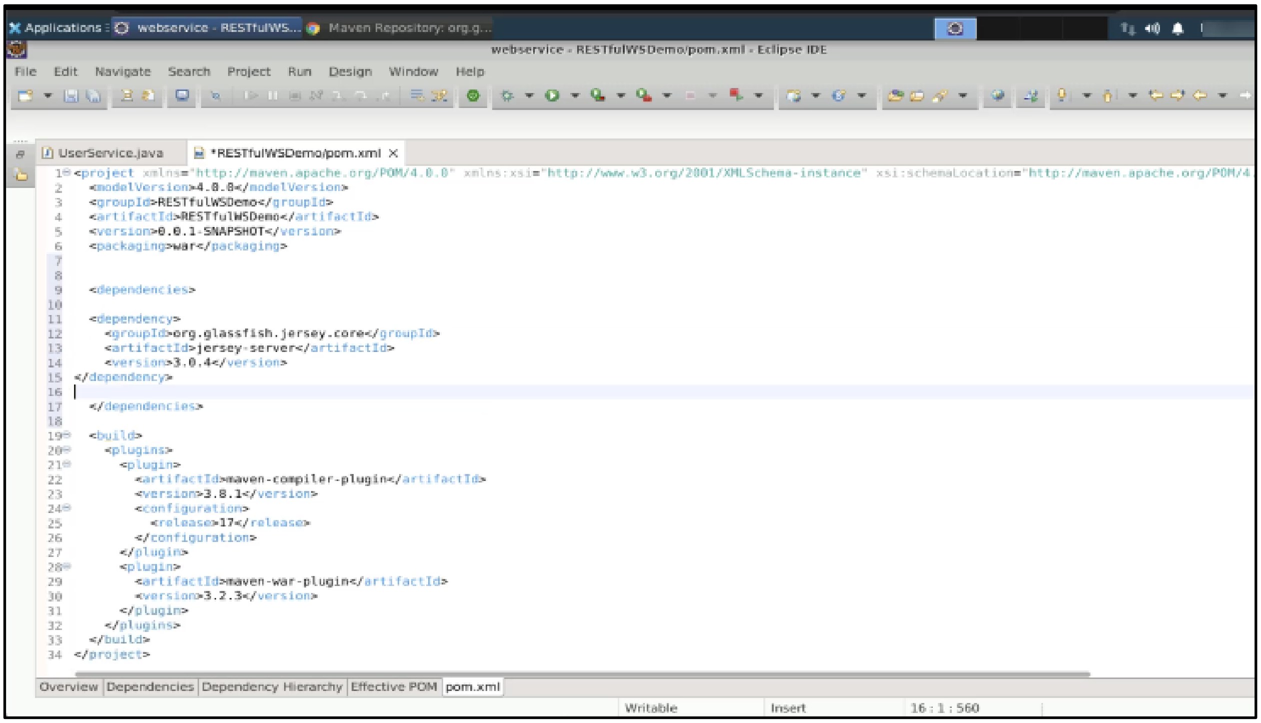
License	Apache 2.0 BSD EPL 2.0 GPL 2.0
Date	(Feb 09, 2022)
Files	pom (12 KB) jar (933 KB) View All
Repositories	Central
Used By	834 artifacts

```
<dependency>
    <groupId>org.glassfish.jersey.core</groupId>
    <artifactId>jersey-server</artifactId>
    <version>3.0.4</version>
</dependency>
```

☐ Include comment with link to declaration

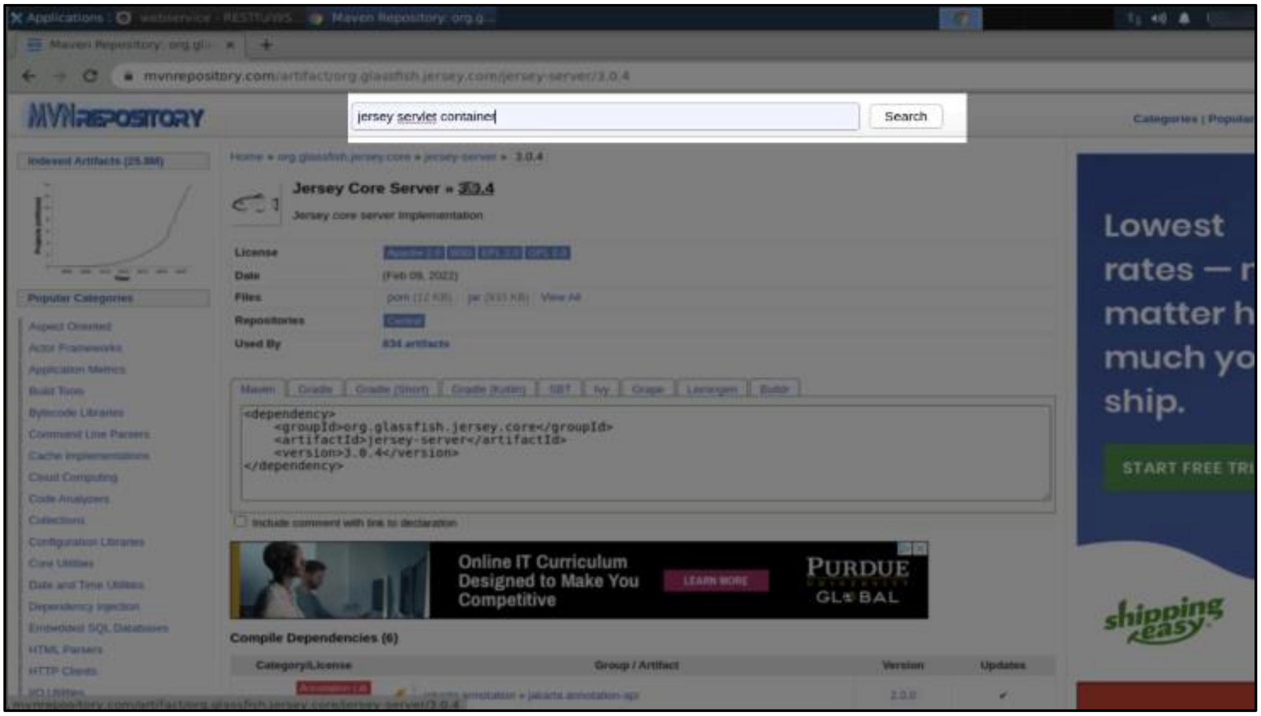
Copied to clipboard!

2.9 Paste it in the **pom.xml** file within the **dependencies** tag


webservice - RESTfulWSDemo/pom.xml - Eclipse IDE
File Edit Navigate Search Project Run Design Window Help
UserService.java
*RESTfulWSDemo/pom.xml
Overview Dependencies Dependency Hierarchy Effective POM pom.xml
Writable Insert 16 : 1 : 560

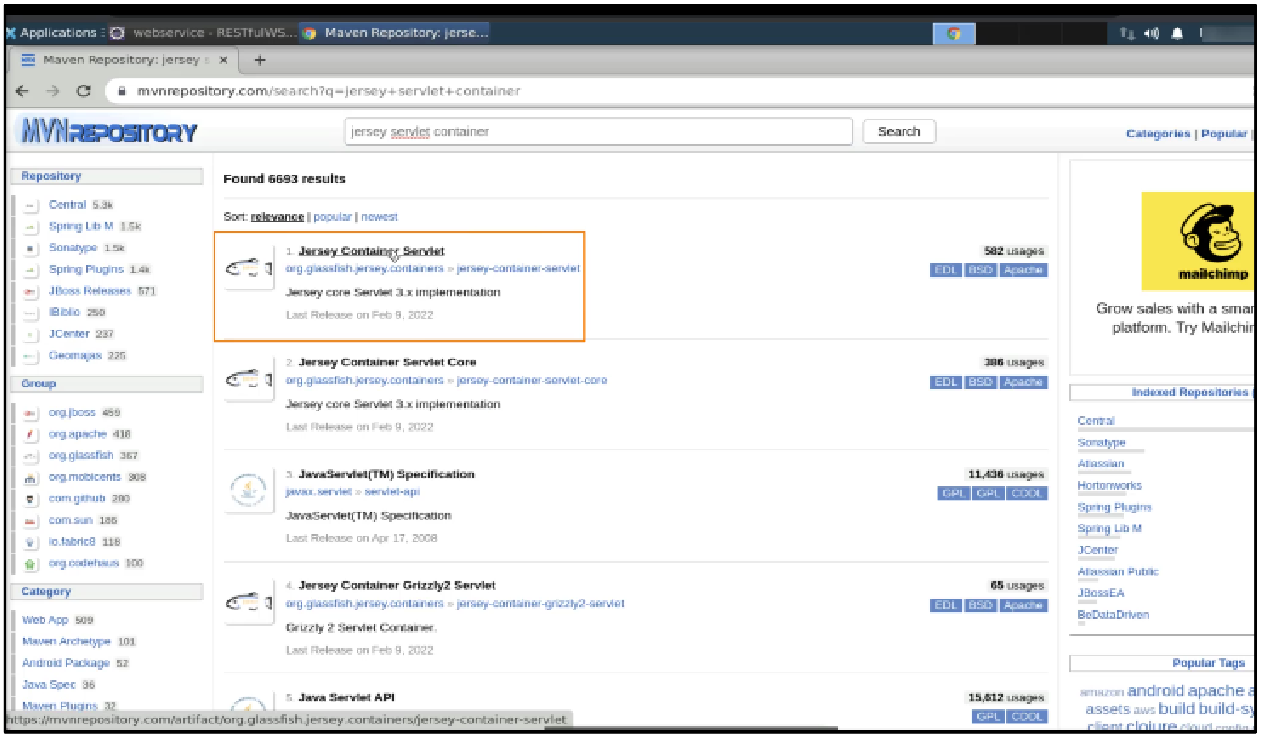
```
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http://maven.apache.org/POM/4
  <modelVersion>4.0.0</modelVersion>
  <groupId>RESTfulWSDemo</groupId>
  <artifactId>RESTfulWSDemo</artifactId>
  <version>0.0.1-SNAPSHOT</version>
  <packaging>war</packaging>


  <dependencies>

  <dependency>
    <groupId>org.glassfish.jersey.core</groupId>
    <artifactId>jersey-server</artifactId>
    <version>3.0.4</version>
</dependency>

   </dependencies>

  <build>
    <plugins>
      <plugin>
        <artifactId>maven-compiler-plugin</artifactId>
        <version>3.8.1</version>
        <configuration>
          <release>17</release>
        </configuration>
      </plugin>
      <plugin>
        <artifactId>maven-war-plugin</artifactId>
        <version>3.2.3</version>
      </plugin>
    </plugins>
  </build>
</project>
```

2.10 Search for **Jersey Servlet Container** on **mvnrepository.com** website


mvnrepository.com/artifact/org.glassfish.jersey.core/jersey-server/3.0.4
MVNREPOSITORY
jersey servlet container
Search
Jersey Core Server » 3.0.4
Maven
Include comment with link to declaration
Online IT Curriculum Designed to Make You Competitive
LEARN MORE
PURDUE GLOBAL
Lowest rates — matter how much you ship.
START FREE TRI
shipping easy
Compile Dependencies (6)

```
<dependency>
    <groupId>org.glassfish.jersey.core</groupId>
    <artifactId>jersey-server</artifactId>
    <version>3.0.4</version>
</dependency>
```

2.11Select **Jersey Containing Servlet** dependency



Found 6693 results

Sort: relevance | popular | newest

1. Jersey Container Servlet — 582 usages — EDL BSD Apache
org.glassfish.jersey.containers » jersey-container-servlet
Jersey core Servlet 3.x implementation
Last Release on Feb 9, 2022

2. Jersey Container Servlet Core — 386 usages — EDL BSD Apache
org.glassfish.jersey.containers » jersey-container-servlet-core
Jersey core Servlet 3.x implementation
Last Release on Feb 9, 2022

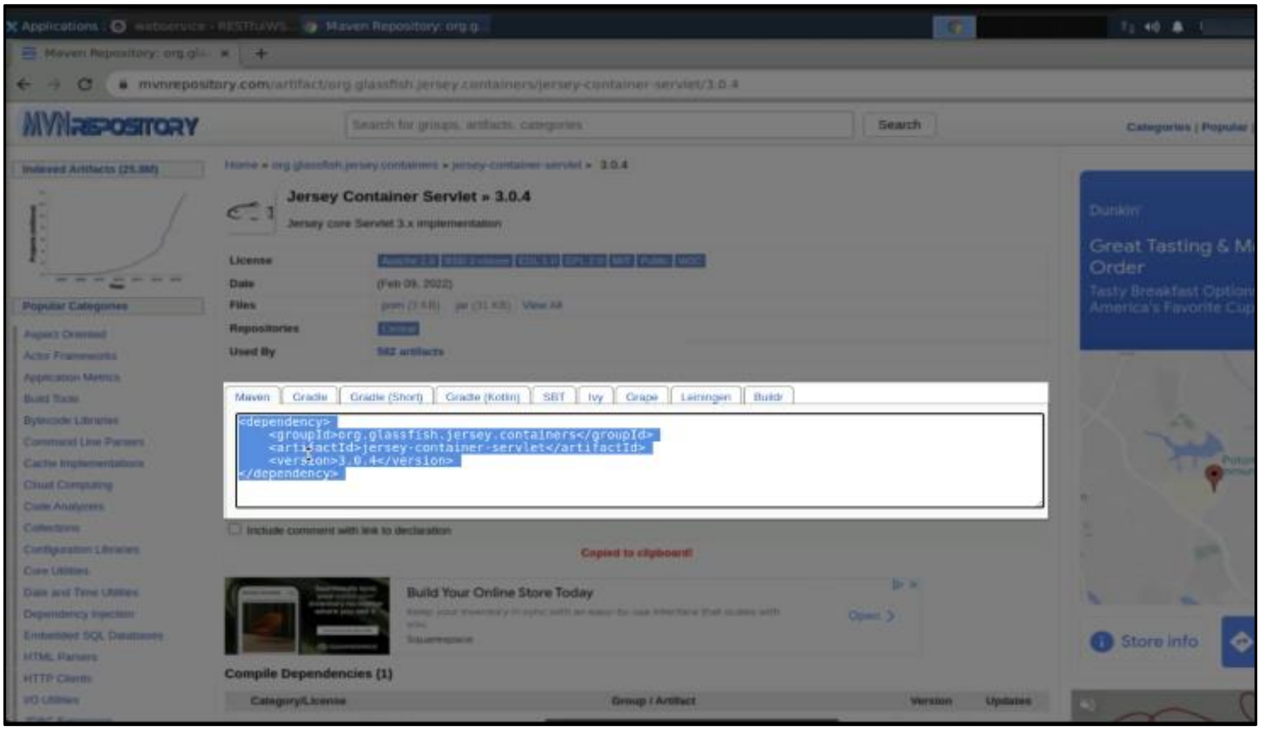
3. JavaServlet(TM) Specification — 11,438 usages — GPL GPL CDDL
javax.servlet » servlet-api
JavaServlet(TM) Specification
Last Release on Apr 17, 2008

4. Jersey Container Grizzly2 Servlet — 65 usages — EDL BSD Apache
org.glassfish.jersey.containers » jersey-container-grizzly2-servlet
Grizzly 2 Servlet Container.
Last Release on Feb 9, 2022

5. Java Servlet API — 15,812 usages — GPL CDDL

https://mvnrepository.com/artifact/org.glassfish.jersey.containers/jersey-container-servlet

2.12 Copy the dependency script


Applications
webservice - RESTfulWS...
Maven Repository: org.g...
mvnrepository.com/artifact/org.glassfish.jersey.containers/jersey-container-servlet/3.0.4
MVNREPOSITORY
Search for groups, artifacts, categories
Search
Categories | Popular
Indexed Artifacts (25.3M)
Popular Categories
Maven
Gradle
Gradle (Short)
Gradle (Kotlin)
SBT
Ivy
Grape
Leiningen
Buildr
Build Your Online Store Today
Open
Store info

Home » org.glassfish.jersey.containers » jersey-container-servlet » 3.0.4

Jersey Container Servlet » 3.0.4

Jersey core Servlet 3.x implementation

Date	(Feb 09, 2022)
Used By	582 artifacts

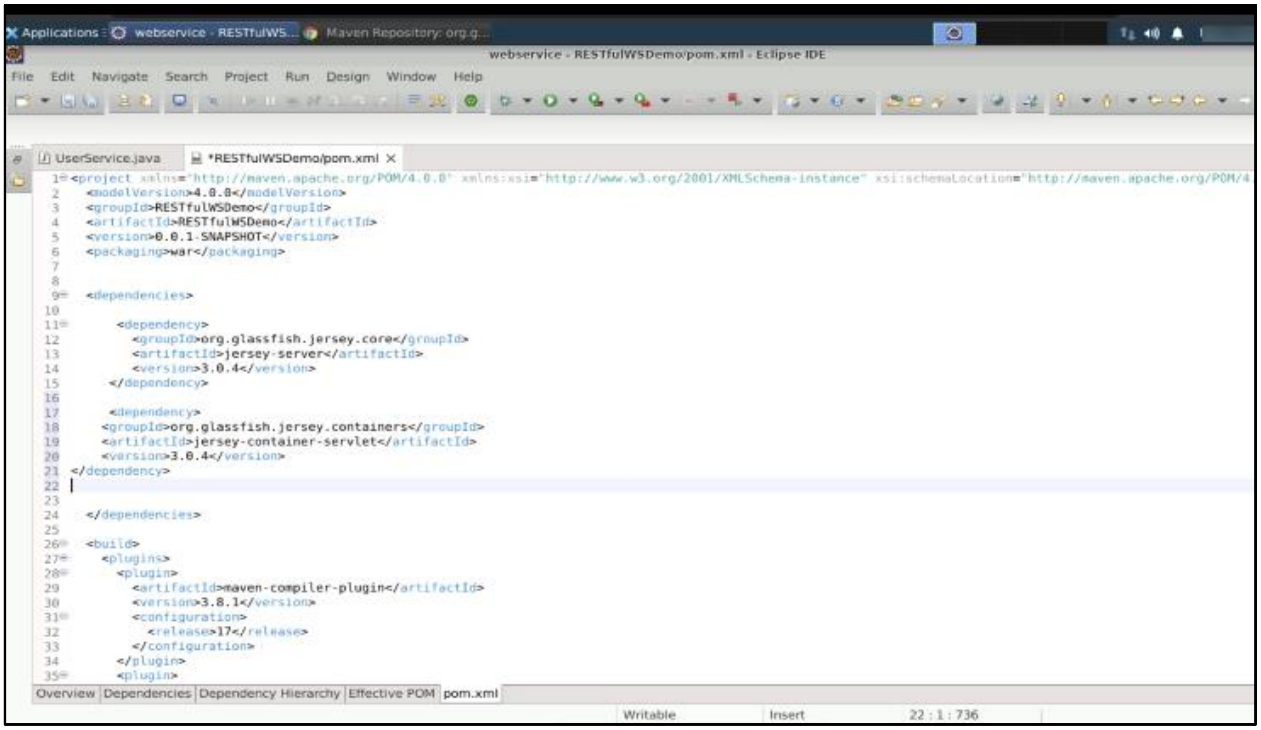
```
<dependency>
    <groupId>org.glassfish.jersey.containers</groupId>
    <artifactId>jersey-container-servlet</artifactId>
    <version>3.0.4</version>
</dependency>
```

□ Include comment with link to declaration

Copied to clipboard!

Compile Dependencies (1)

2.13 Paste the copied dependencies in the **pom.xml** file under the **dependencies** tag



Applications
webservice - RESTfulWS...
Maven Repository: org.g...
webservice - RESTfulWSDemo/pom.xml - Eclipse IDE
File Edit Navigate Search Project Run Design Window Help
UserService.java
*RESTfulWSDemo/pom.xml

```
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http://maven.apache.org/POM/4
  <modelVersion>4.0.0</modelVersion>
  <groupId>RESTfulWSDemo</groupId>
  <artifactId>RESTfulWSDemo</artifactId>
  <version>0.0.1-SNAPSHOT</version>
  <packaging>war</packaging>


  <dependencies>

      <dependency>
        <groupId>org.glassfish.jersey.core</groupId>
        <artifactId>jersey-server</artifactId>
        <version>3.0.4</version>
     </dependency>

     <dependency>
    <groupId>org.glassfish.jersey.containers</groupId>
    <artifactId>jersey-container-servlet</artifactId>
    <version>3.0.4</version>
</dependency>


  </dependencies>

  <build>
    <plugins>
      <plugin>
        <artifactId>maven-compiler-plugin</artifactId>
        <version>3.8.1</version>
        <configuration>
          <release>17</release>
        </configuration>
      </plugin>
      <plugin>
```

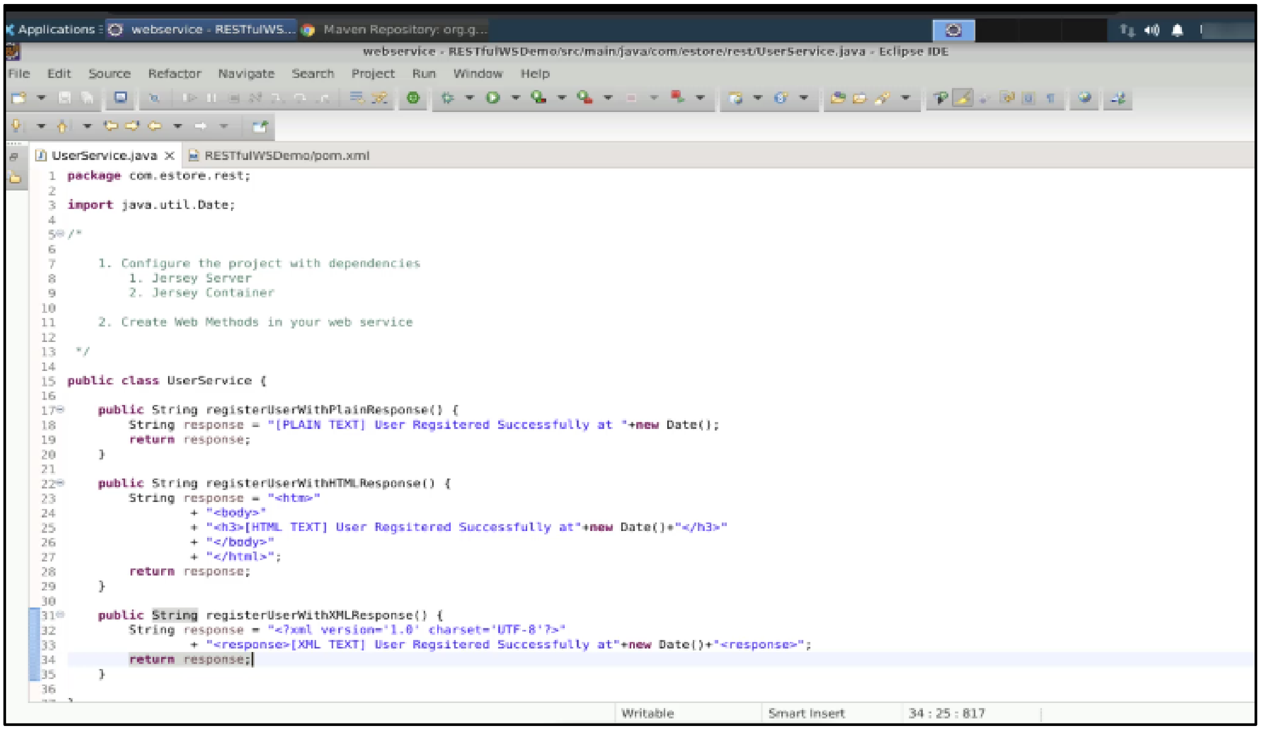
Overview
Dependencies
Dependency Hierarchy
Effective POM
pom.xml
Writable
Insert
22 : 1 : 736

2.14 Add the code given below in **UserService.java:**

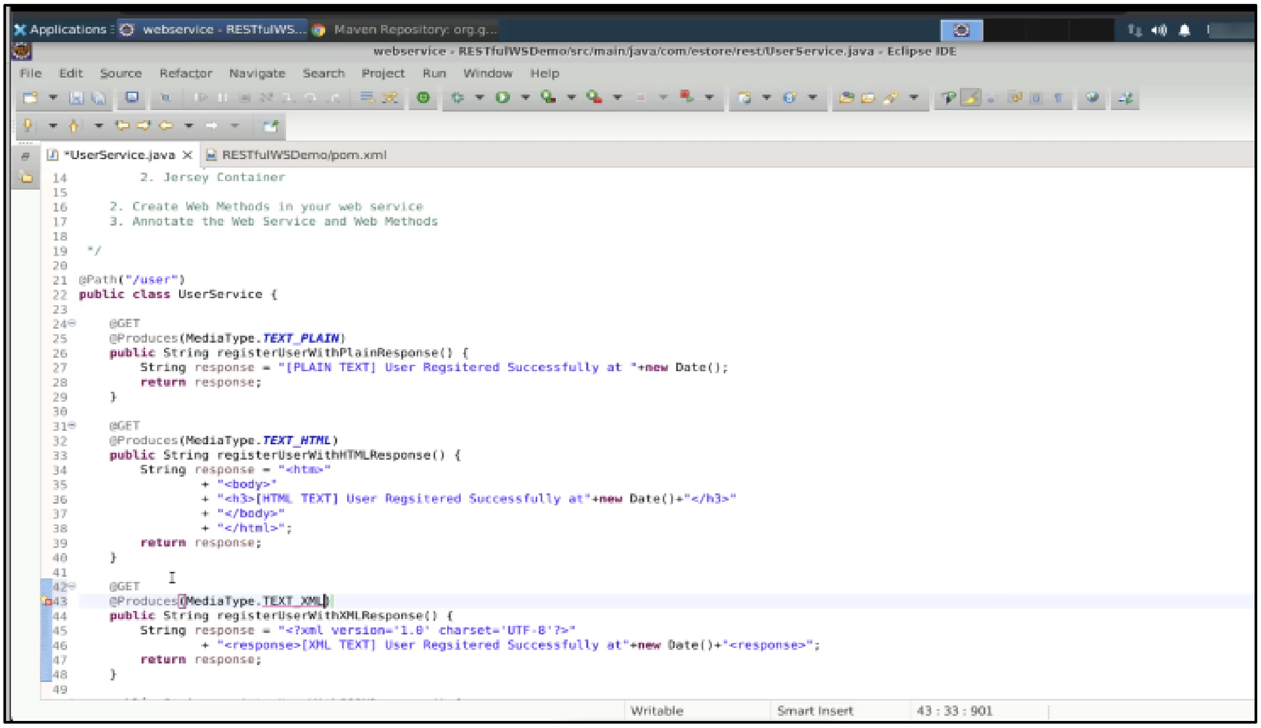
```
package com.estore.rest;

import java.util.Date;

//import org.glassfish.jersey.servlet.ServletContainer;

import jakarta.ws.rs.GET;
import jakarta.ws.rs.Path;
import jakarta.ws.rs.Produces;
import jakarta.ws.rs.core.MediaType;

/*

        1. Configure the project with dependencies
                1. Jersey Server
```

```
                    2. Jersey Container

          2. Create Web Methods in your web service
          3. Annotate the Web Service and Web Methods
          4. Configure ServletContainer from the Jersey in web.xml
[org.glassfish.jersey.servlet.ServletContainer]

 */

@Path("/user")
public class UserService {

          @GET
          @Produces(MediaType.TEXT_PLAIN)
          public String registerUserWithPlainResponse() {
                    String response = "[PLAIN TEXT] User Regsitered Successfully at "+new
Date();
                    return response;
          }

          @GET
          @Produces(MediaType.TEXT_HTML)
          public String registerUserWithHTMLResponse() {
                    String response = "<htm>"
                                        + "<body>"
                                        + "<h3>[HTML TEXT] User Regsitered Successfully
at"+new Date()+"</h3>"
                                        + "</body>"
                                        + "</html>";
                    return response;
          }

          @GET
          @Produces(MediaType.TEXT_XML)
          public String registerUserWithXMLResponse() {
                    String response = "<?xml version='1.0' charset='UTF-8'?>"
                                        + "<response>[XML TEXT] User Regsitered Successfully
at"+new Date()+"<response>";
```

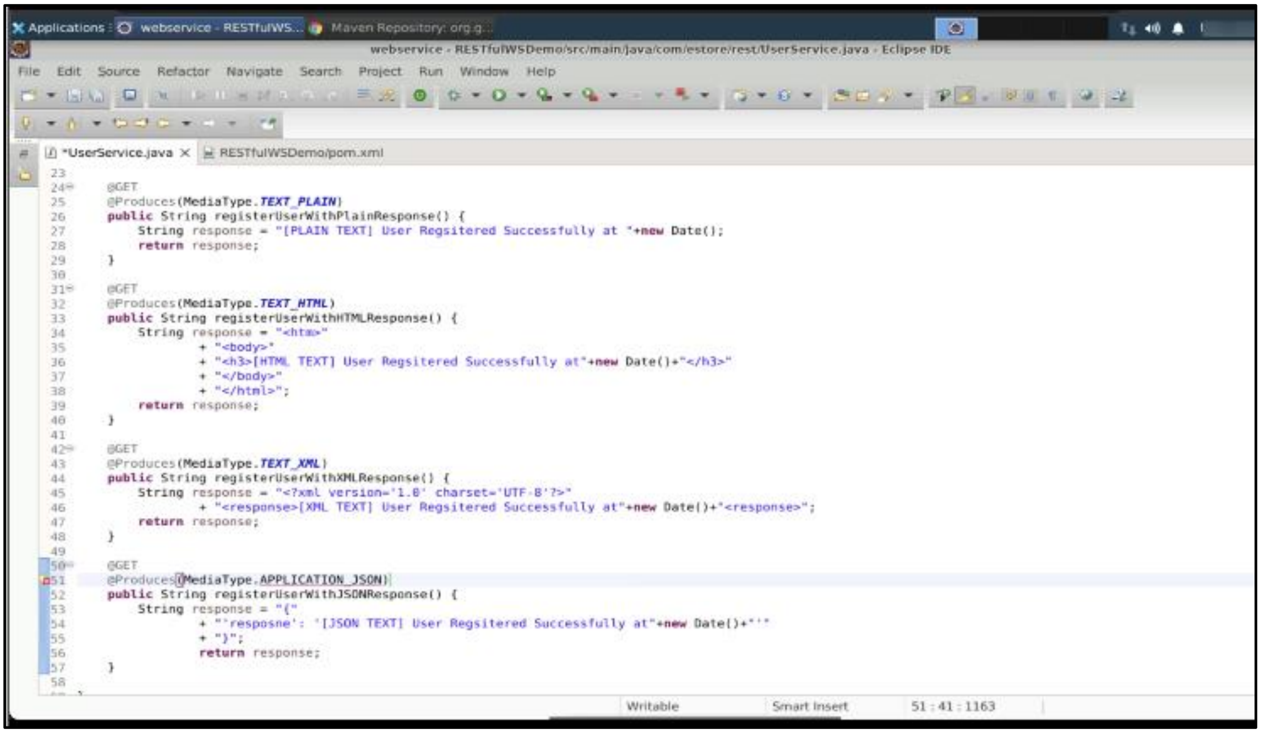
```
                return response;
        }

        @GET
        @Produces(MediaType.APPLICATION_JSON)
        public String registerUserWithJSONResponse() {
                String response = "{"
                                        + "'resposne': '[JSON TEXT] User Regsitered Successfully
at"+new Date()+"'"
                                        + "}";
                                        return response;
        }

}
```


Applications
webservice - RESTfulWS...
Maven Repository: org.g...
webservice - RESTfulWSDemo/src/main/java/com/estore/rest/UserService.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
UserService.java
RESTfulWSDemo/pom.xml
Writable
Smart Insert
34 : 25 : 817

```
package com.estore.rest;

import java.util.Date;

/*

    1. Configure the project with dependencies
        1. Jersey Server
        2. Jersey Container

    2. Create Web Methods in your web service

 */

public class UserService {

    public String registerUserWithPlainResponse() {
        String response = "[PLAIN TEXT] User Regsitered Successfully at "+new Date();
        return response;
    }

    public String registerUserWithHTMLResponse() {
        String response = "<htm>"
                + "<body>"
                + "<h3>[HTML TEXT] User Regsitered Successfully at"+new Date()+"</h3>"
                + "</body>"
                + "</html>";
        return response;
    }

    public String registerUserWithXMLResponse() {
        String response = "<?xml version='1.0' charset='UTF-8'?>"
                + "<response>[XML TEXT] User Regsitered Successfully at"+new Date()+"<response>";
        return response;
    }

```


Applications
webservice - RESTfulWS...
Maven Repository: org.g...
webservice - RESTfulWSDemo/src/main/java/com/estore/rest/UserService.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
*UserService.java
RESTfulWSDemo/pom.xml
Writable
Smart Insert
43 : 33 : 901

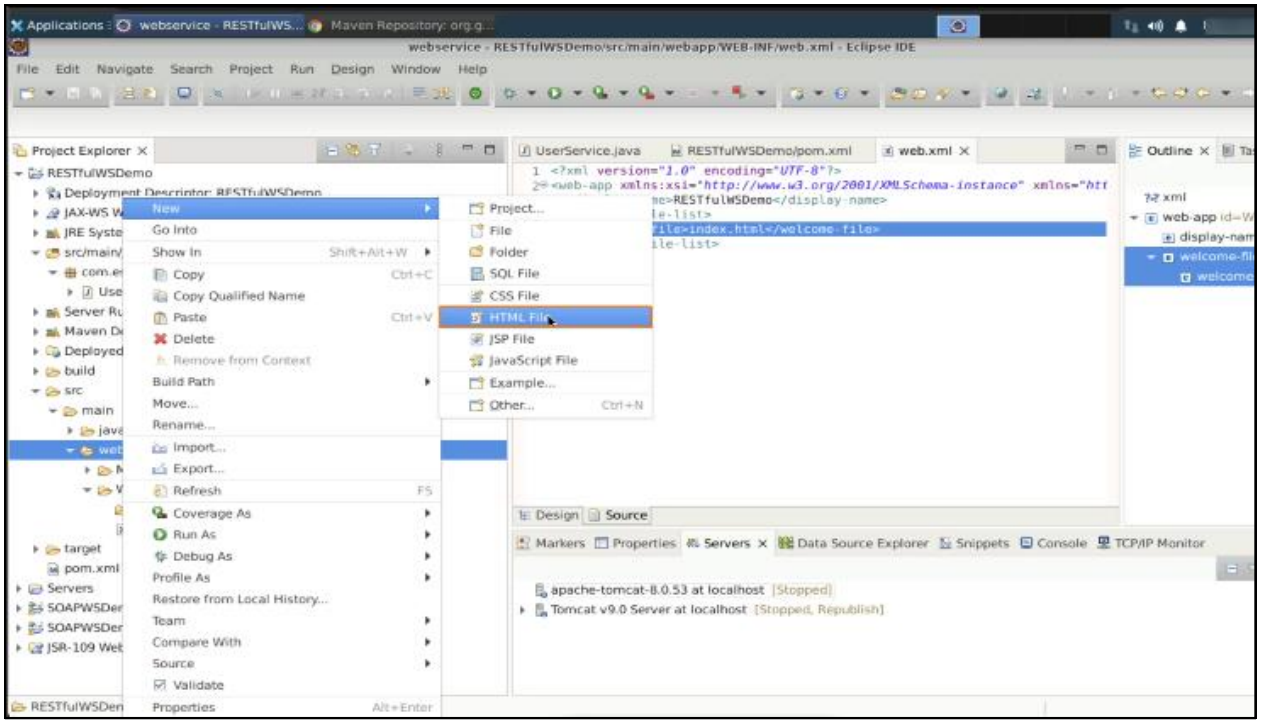
```
        2. Jersey Container

    2. Create Web Methods in your web service
    3. Annotate the Web Service and Web Methods

 */

@Path("/user")
public class UserService {

    @GET
    @Produces(MediaType.TEXT_PLAIN)
    public String registerUserWithPlainResponse() {
        String response = "[PLAIN TEXT] User Regsitered Successfully at "+new Date();
        return response;
    }

    @GET
    @Produces(MediaType.TEXT_HTML)
    public String registerUserWithHTMLResponse() {
        String response = "<html>"
                + "<body>"
                + "<h3>[HTML TEXT] User Regsitered Successfully at"+new Date()+"</h3>"
                + "</body>"
                + "</html>";
        return response;
    }

    @GET
    @Produces(MediaType.TEXT_XML)
    public String registerUserWithXMLResponse() {
        String response = "<?xml version='1.0' charset='UTF-8'?>"
                + "<response>[XML TEXT] User Regsitered Successfully at"+new Date()+"<response>";
        return response;
    }

```


Applications
webservice - RESTfulWS...
Maven Repository: org.g...
webservice - RESTfulWSDemo/src/main/java/com/estore/rest/UserService.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
*UserService.java
RESTfulWSDemo/pom.xml
Writable
Smart Insert
51 : 41 : 1163

```

    @GET
    @Produces(MediaType.TEXT_PLAIN)
    public String registerUserWithPlainResponse() {
        String response = "[PLAIN TEXT] User Regsitered Successfully at "+new Date();
        return response;
    }

    @GET
    @Produces(MediaType.TEXT_HTML)
    public String registerUserWithHTMLResponse() {
        String response = "<html>"
                + "<body>"
                + "<h3>[HTML TEXT] User Regsitered Successfully at"+new Date()+"</h3>"
                + "</body>"
                + "</html>";
        return response;
    }

    @GET
    @Produces(MediaType.TEXT_XML)
    public String registerUserWithXMLResponse() {
        String response = "<?xml version='1.0' charset='UTF-8'?>"
                + "<response>[XML TEXT] User Regsitered Successfully at"+new Date()+"<response>";
        return response;
    }

    @GET
    @Produces(MediaType.APPLICATION_JSON)
    public String registerUserWithJSONResponse() {
        String response = "{"
                + "'resposne': '[JSON TEXT] User Regsitered Successfully at"+new Date()+"'"
                + "}";
                return response;
    }

```

Step 3: Creating an HTML file and mapping a Servlet in web.xml

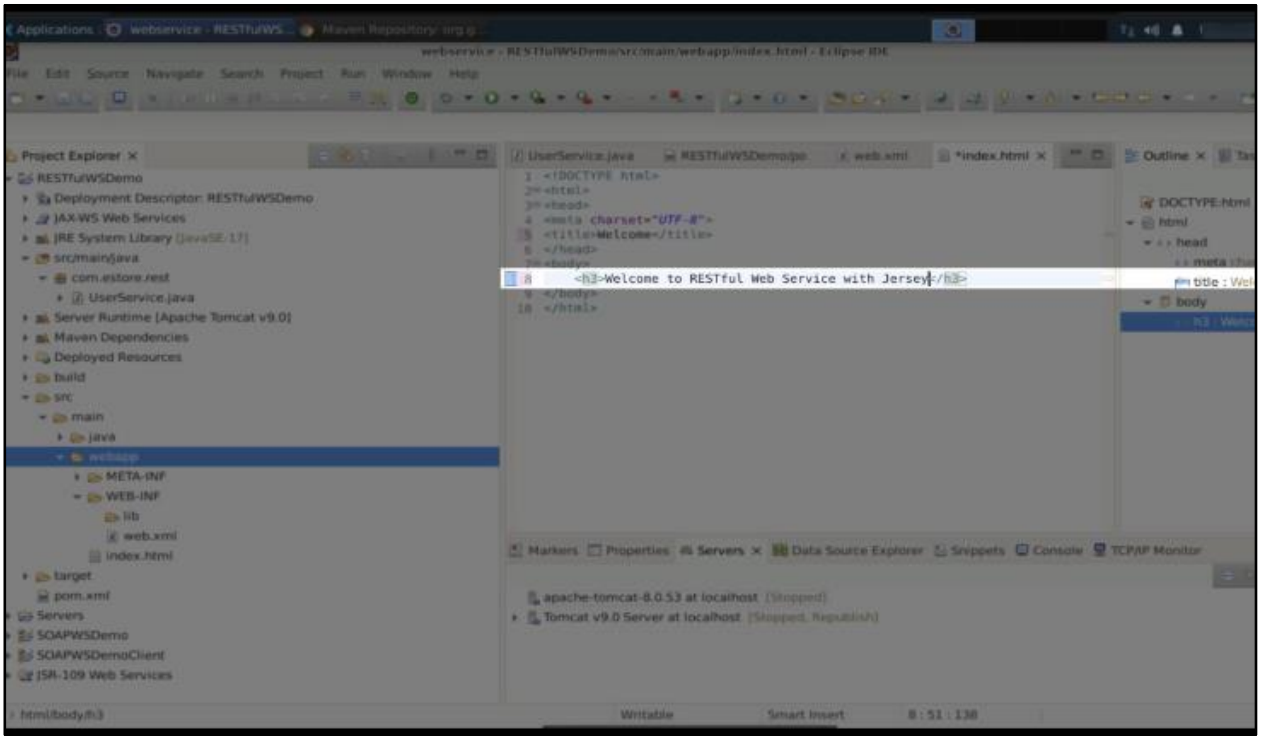
3.1 Right-click on the **web-inf** folder and select **New -> HTML File** and give the name of the file as **index**



```
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="htt
ne>RESTfulWSDemo</display-name>
le-list>
file>index.html</welcome-file>
ile-list>
```

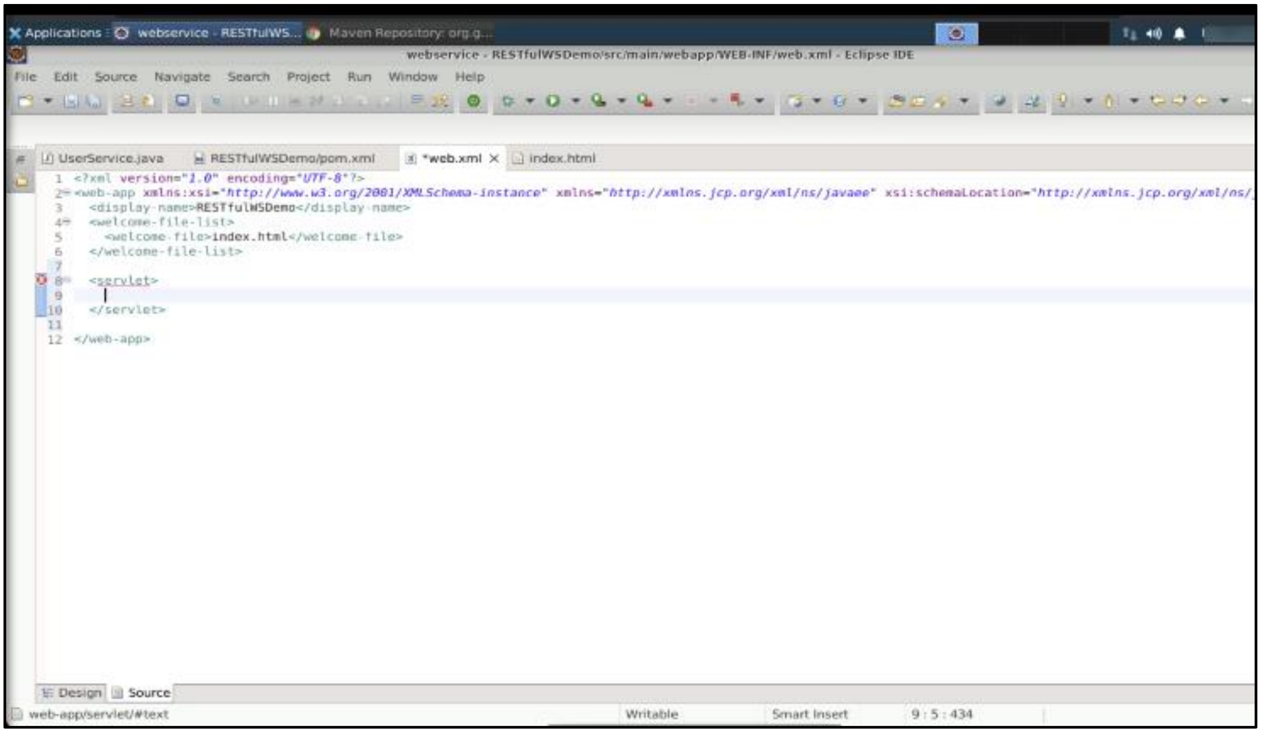
3.2 Add the code given below in the HTML file:

```
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>Welcome</title>
</head>
<body>
        <h3>Welcome to RESTful Web Service with Jersey</h3>
        <a href="MyRestClient.jsp">Test The Web Service with Client</a>
</body>
</html>
```



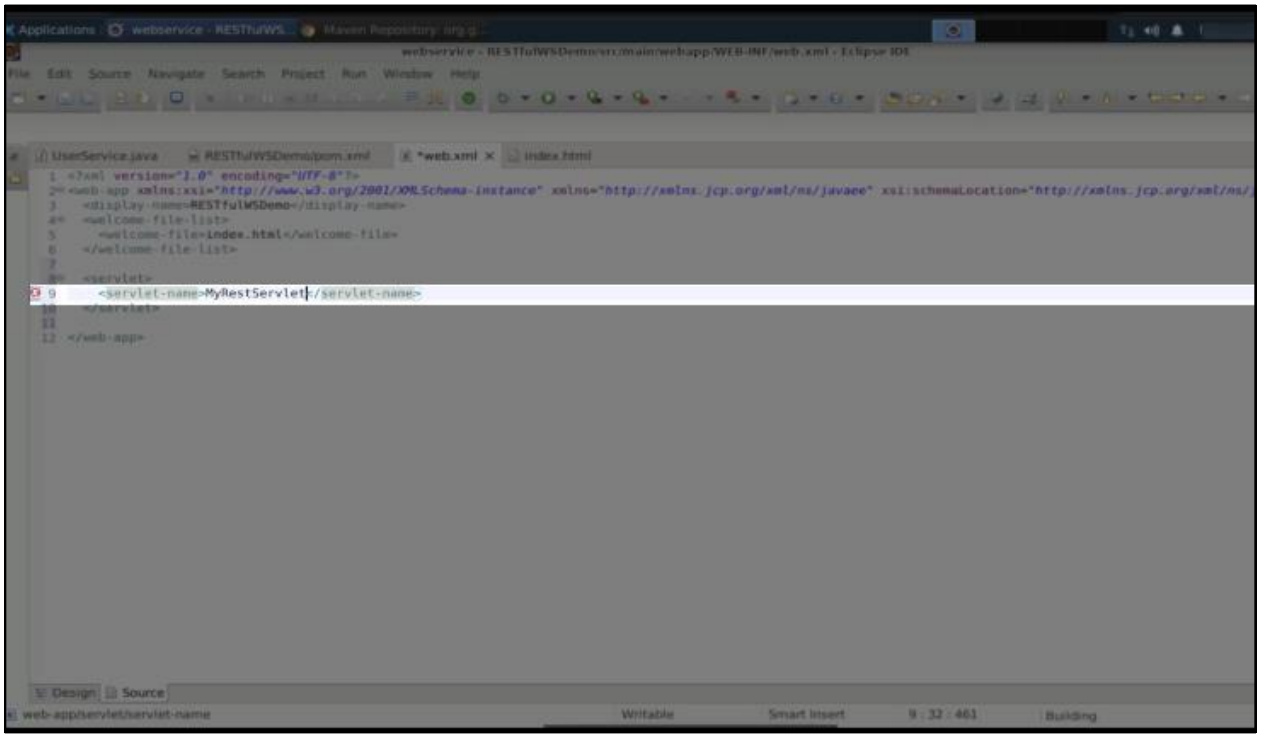
```
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>Welcome</title>
</head>
<body>
    <h3>Welcome to RESTful Web Service with Jersey</h3>
</body>
</html>
```

3.3 Open the **web.xml** file and add a servlet tag in the **web.xml** file



```
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="http://xmlns.jcp.org/xml/ns/javaee" xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/
  <display-name>RESTfulWSDemo</display-name>
  <welcome-file-list>
    <welcome-file>index.html</welcome-file>
  </welcome-file-list>

  <servlet>

  </servlet>

</web-app>
```

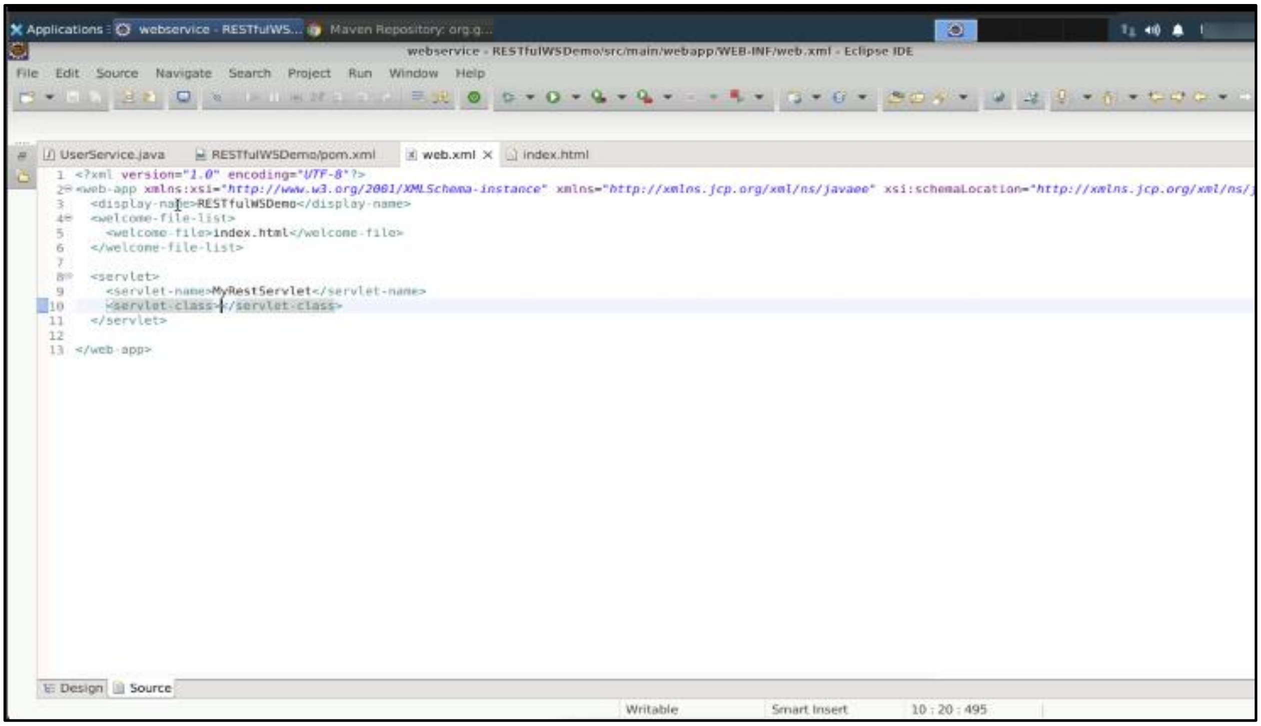
3.4 Provide the name of the servlet as **MyRestServlet**



```
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="http://xmlns.jcp.org/xml/ns/javaee" xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/j
  <display-name>RESTfulWSDemo</display-name>
  <welcome-file-list>
    <welcome-file>index.html</welcome-file>
  </welcome-file-list>

  <servlet>
    <servlet-name>MyRestServlet</servlet-name>
  </servlet>

</web-app>
```

webservice - RESTfulWSDemo/src/main/webapp/WEB-INF/web.xml - Eclipse IDE
File Edit Source Navigate Search Project Run Window Help
UserService.java
RESTfulWSDemo/pom.xml
*web.xml
index.html
Design
Source
web-app/servlet/servlet-name
Writable
Smart Insert
9 : 32 : 461
Building

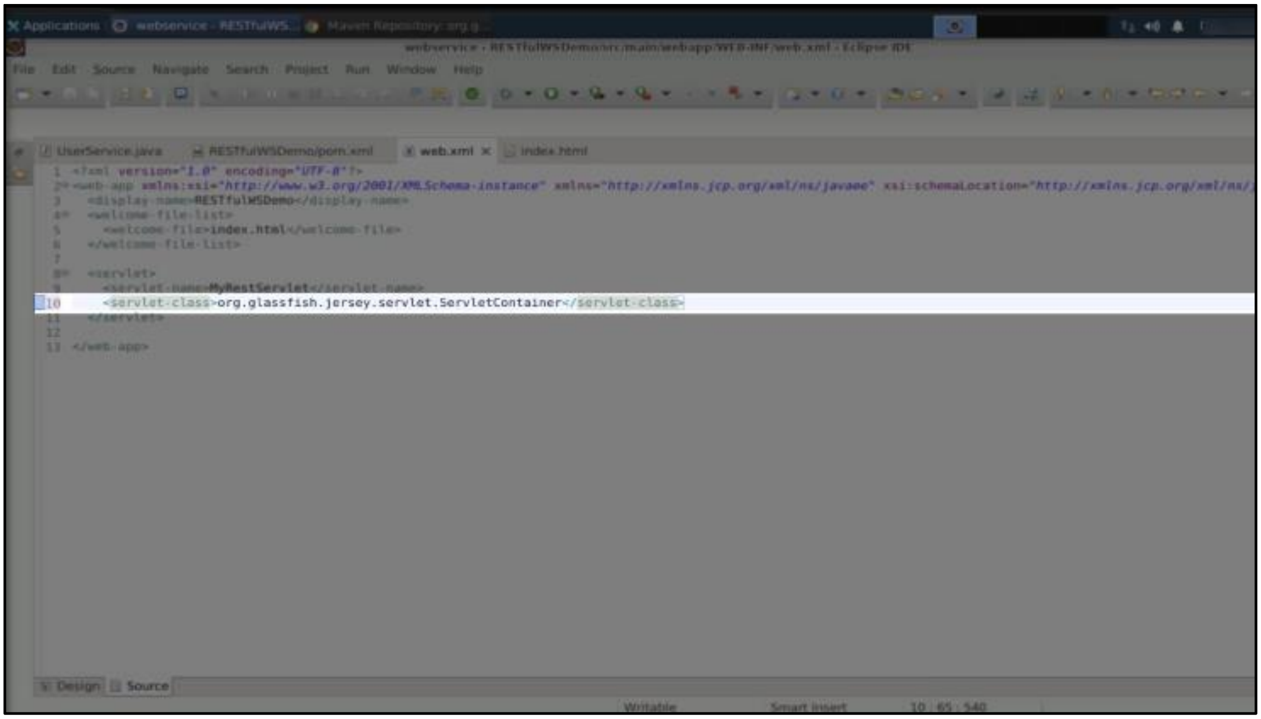
3.5 Add a class for the servlet



```
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="http://xmlns.jcp.org/xml/ns/javaee" xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/j
  <display-name>RESTfulWSDemo</display-name>
  <welcome-file-list>
    <welcome-file>index.html</welcome-file>
  </welcome-file-list>

  <servlet>
    <servlet-name>MyRestServlet</servlet-name>
    <servlet-class></servlet-class>
  </servlet>

</web-app>
```

Applications
webservice - RESTfulWS...
Maven Repository: org.g...
webservice - RESTfulWSDemo/src/main/webapp/WEB-INF/web.xml - Eclipse IDE
File Edit Source Navigate Search Project Run Window Help
UserService.java
RESTfulWSDemo/pom.xml
web.xml
index.html
Design
Source
Writable
Smart Insert
10 : 20 : 495

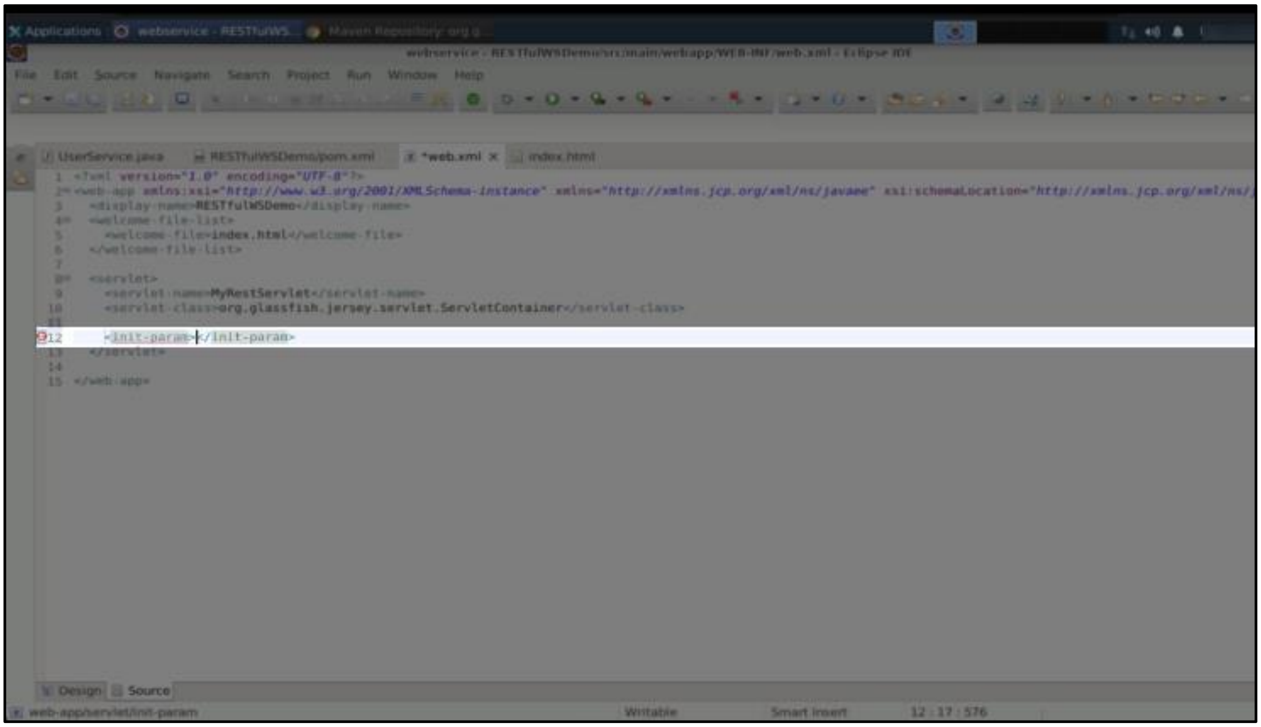
3.6 Add the package name for the servlet class in **web.xml** as **org.glassfish.jersey.servlet.ServletContainer,** as shown below:



```
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="http://xmlns.jcp.org/xml/ns/javaee" xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/j
  <display-name>RESTfulWSDemo</display-name>
  <welcome-file-list>
    <welcome-file>index.html</welcome-file>
  </welcome-file-list>

  <servlet>
    <servlet-name>MyRestServlet</servlet-name>
    <servlet-class>org.glassfish.jersey.servlet.ServletContainer</servlet-class>
  </servlet>

</web-app>
```

Applications
webservice - RESTfulWS...
Maven Repository: org.g...
webservice - RESTfulWSDemo/src/main/webapp/WEB-INF/web.xml - Eclipse IDE
File Edit Source Navigate Search Project Run Window Help
UserService.java
RESTfulWSDemo/pom.xml
web.xml
index.html
Design
Source
Writable
Smart Insert
10 : 65 : 540

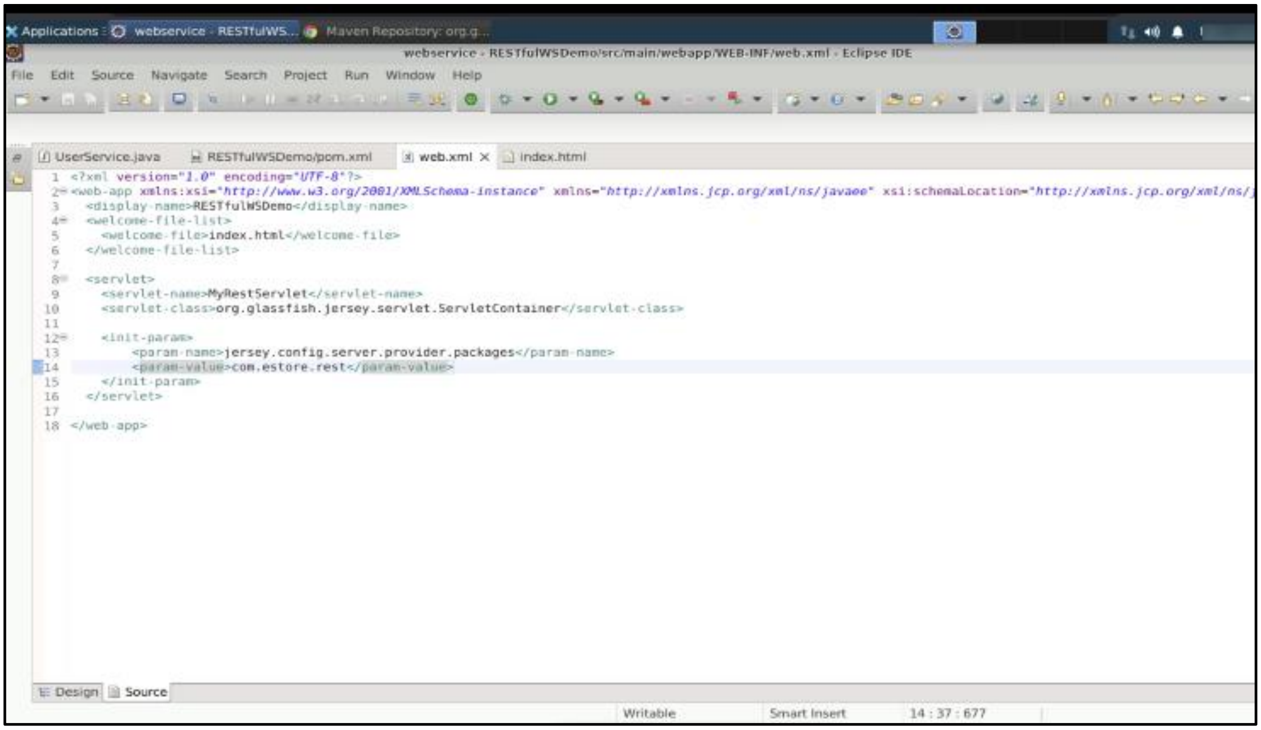
3.7 Pass an initializer parameter



webservice - RESTfulWSDemo/src/main/webapp/WEB-INF/web.xml - Eclipse IDE
File Edit Source Navigate Search Project Run Window Help
UserService.java
RESTfulWSDemo/pom.xml
*web.xml
index.html
Design
Source
web-app/servlet/init-param
Writable
Smart Insert
12 : 17 : 576

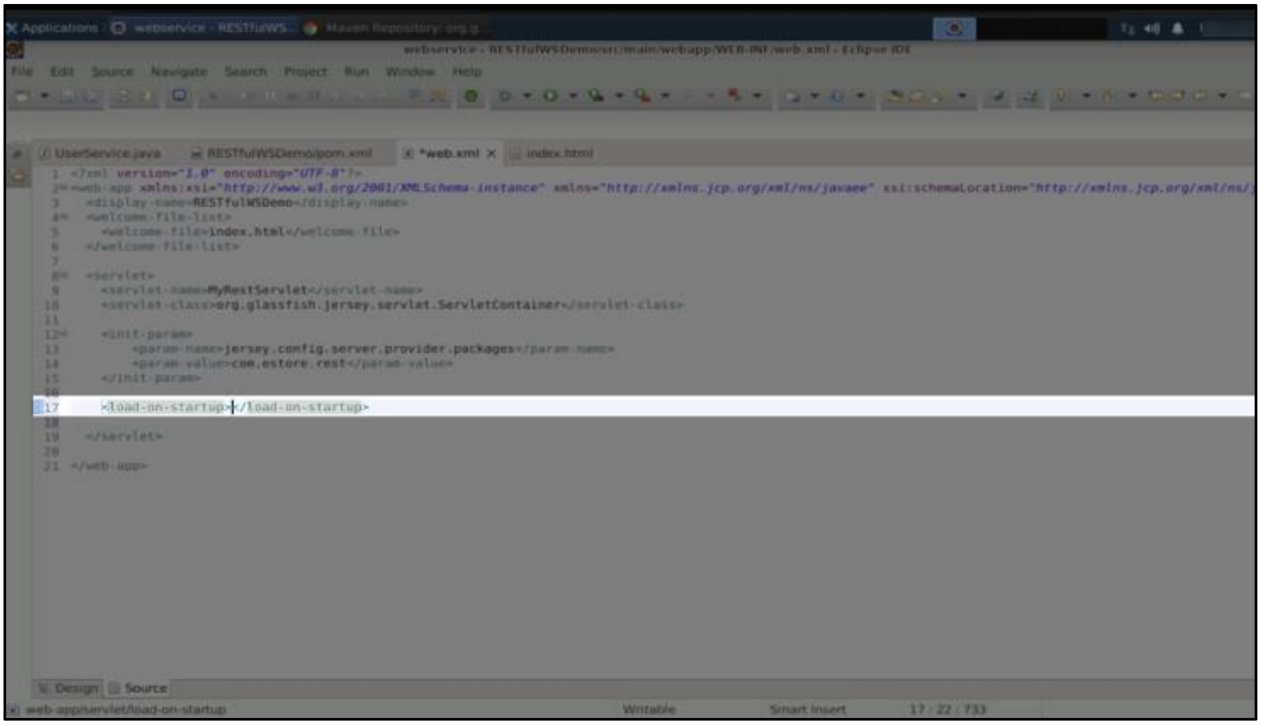
```
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="http://xmlns.jcp.org/xml/ns/javaee" xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/j
  <display-name>RESTfulWSDemo</display-name>
  <welcome-file-list>
    <welcome-file>index.html</welcome-file>
  </welcome-file-list>

  <servlet>
    <servlet-name>MyRestServlet</servlet-name>
    <servlet-class>org.glassfish.jersey.servlet.ServletContainer</servlet-class>

    <init-param></init-param>
  </servlet>

</web-app>
```

3.8 Add **param-values** and **load-on-startup** tags as shown below:



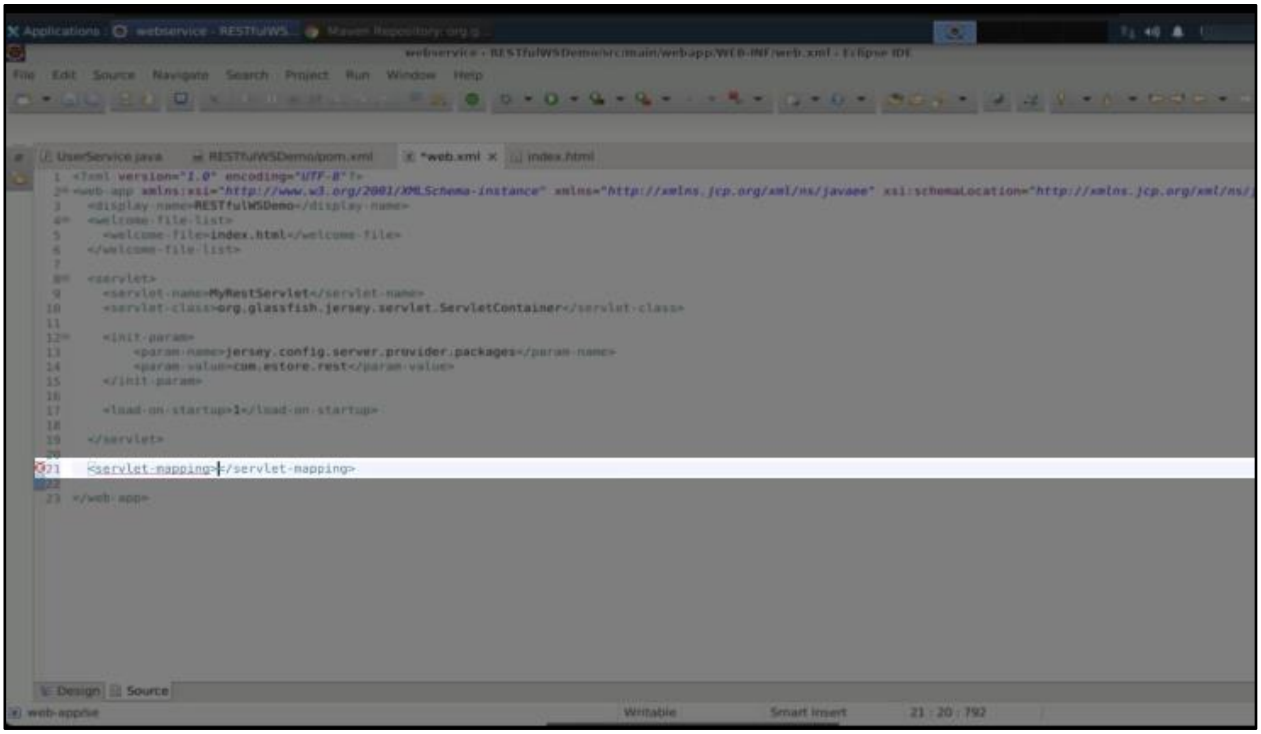
webservice - RESTfulWSDemo/src/main/webapp/WEB-INF/web.xml - Eclipse IDE
File Edit Source Navigate Search Project Run Window Help
UserService.java
RESTfulWSDemo/pom.xml
web.xml
index.html
Design
Source
Writable
Smart Insert
14 : 37 : 677

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="http://xmlns.jcp.org/xml/ns/javaee" xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/j
  <display-name>RESTfulWSDemo</display-name>
  <welcome-file-list>
    <welcome-file>index.html</welcome-file>
  </welcome-file-list>

  <servlet>
    <servlet-name>MyRestServlet</servlet-name>
    <servlet-class>org.glassfish.jersey.servlet.ServletContainer</servlet-class>

    <init-param>
        <param-name>jersey.config.server.provider.packages</param-name>
        <param-value>com.estore.rest</param-value>
    </init-param>
  </servlet>

</web-app>
```


Applications
webservice - RESTfulWS...
Maven Repository: org.g...
webservice - RESTfulWSDemo/src/main/webapp/WEB-INF/web.xml - Eclipse IDE
File Edit Source Navigate Search Project Run Window Help
UserService.java
RESTfulWSDemo/pom.xml
*web.xml
index.html
Design
Source
web-app/servlet/load-on-startup
Writable
Smart Insert
17 : 22 : 733

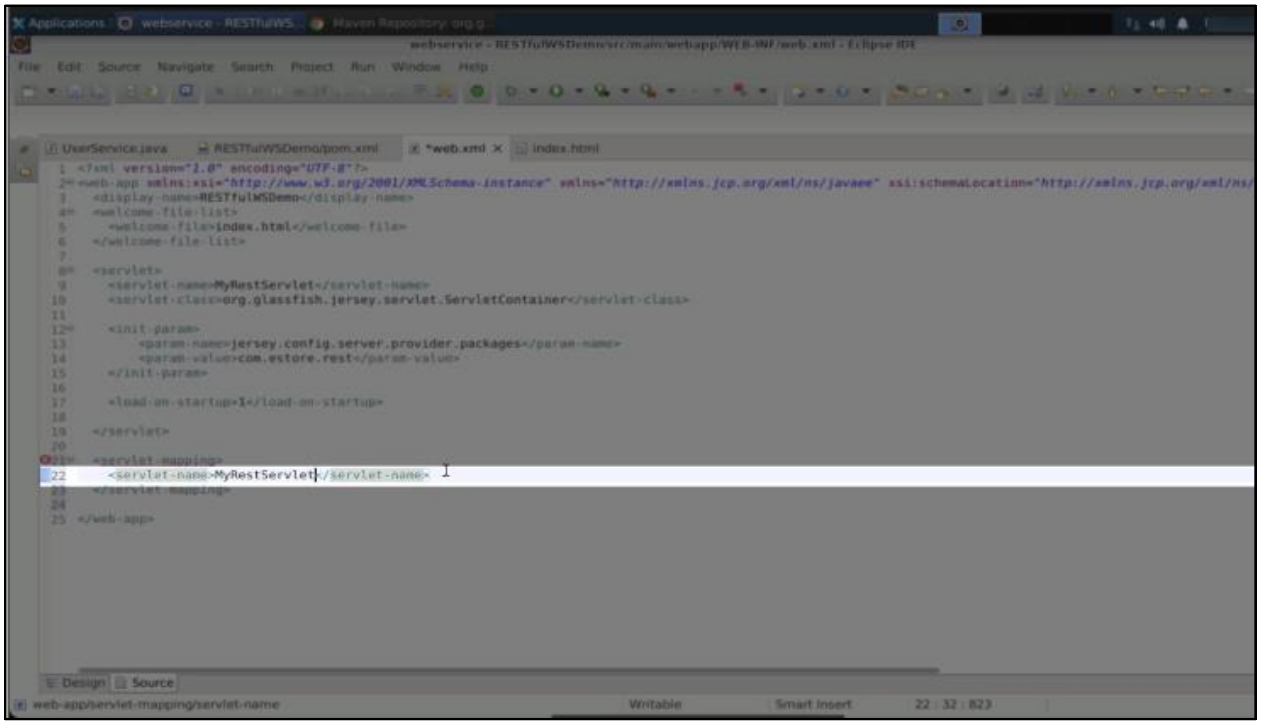
```
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="http://xmlns.jcp.org/xml/ns/javaee" xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/
  <display-name>RESTfulWSDemo</display-name>
  <welcome-file-list>
    <welcome-file>index.html</welcome-file>
  </welcome-file-list>

  <servlet>
    <servlet-name>MyRestServlet</servlet-name>
    <servlet-class>org.glassfish.jersey.servlet.ServletContainer</servlet-class>

    <init-param>
        <param-name>jersey.config.server.provider.packages</param-name>
        <param-value>com.estore.rest</param-value>
    </init-param>

    <load-on-startup></load-on-startup>

  </servlet>

</web-app>
```

3.9 Add **servlet-mapping** tag


Applications
webservice - RESTfulWS...
Maven Repository: org.g...
webservice - RESTfulWSDemo/src/main/webapp/WEB-INF/web.xml - Eclipse IDE
File Edit Source Navigate Search Project Run Window Help
UserService.java
RESTfulWSDemo/pom.xml
*web.xml
index.html
Design
Source
web-app/se
Writable
Smart Insert
21 : 20 : 792

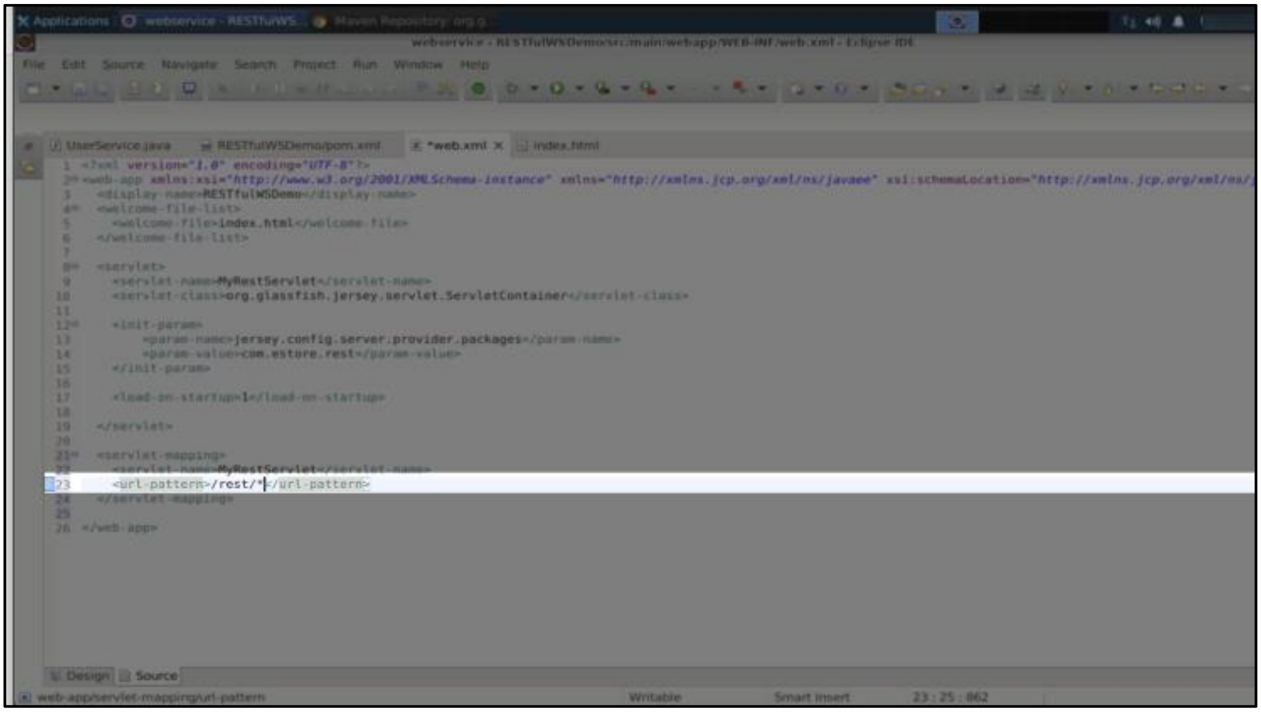
```
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="http://xmlns.jcp.org/xml/ns/javaee" xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/
  <display-name>RESTfulWSDemo</display-name>
  <welcome-file-list>
    <welcome-file>index.html</welcome-file>
  </welcome-file-list>

  <servlet>
    <servlet-name>MyRestServlet</servlet-name>
    <servlet-class>org.glassfish.jersey.servlet.ServletContainer</servlet-class>

    <init-param>
        <param-name>jersey.config.server.provider.packages</param-name>
        <param-value>com.estore.rest</param-value>
    </init-param>

    <load-on-startup>1</load-on-startup>

  </servlet>

  <servlet-mapping></servlet-mapping>

</web-app>
```

3.10 Insert the name of your servlet class as shown below:


Applications
webservice - RESTfulWS...
Maven Repository: org.g...
webservice - RESTfulWSDemo/src/main/webapp/WEB-INF/web.xml - Eclipse IDE
File Edit Source Navigate Search Project Run Window Help
UserService.java
RESTfulWSDemo/pom.xml
*web.xml
index.html
Design
Source
web-app/servlet-mapping/servlet-name
Writable
Smart Insert
22 : 32 : 823

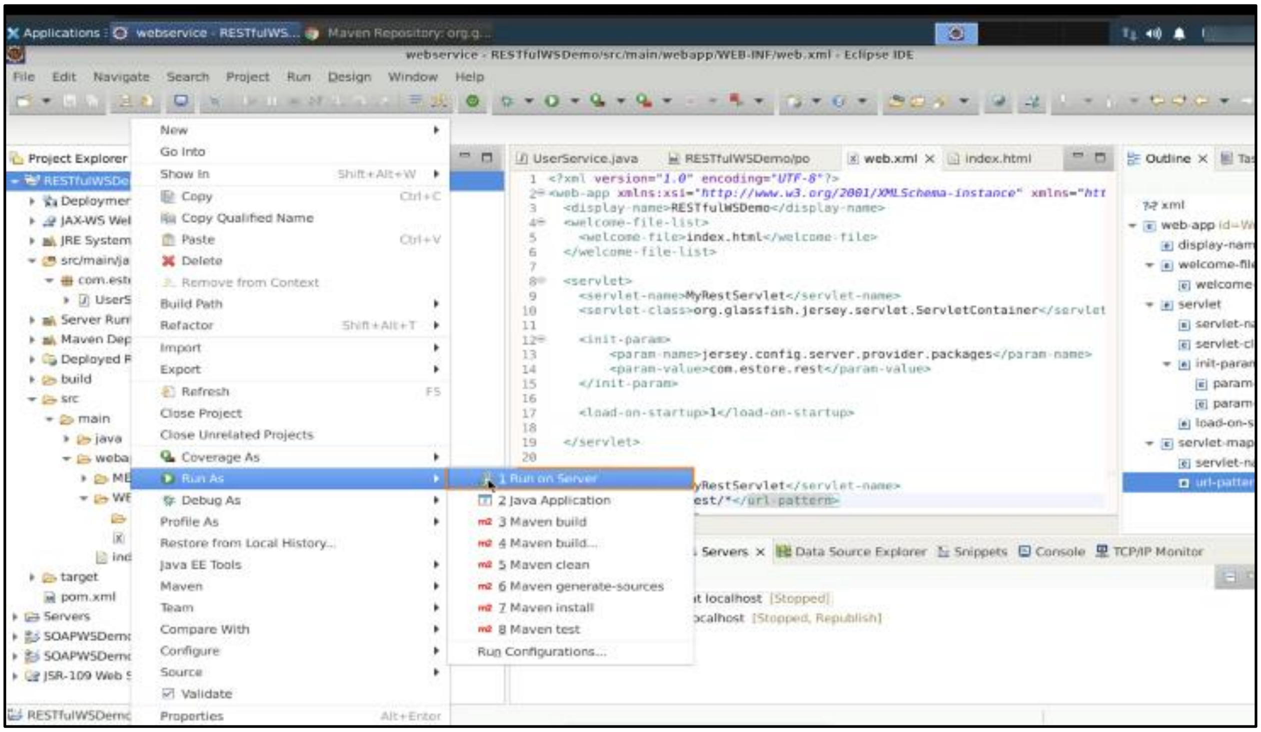
```
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="http://xmlns.jcp.org/xml/ns/javaee" xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/
  <display-name>RESTfulWSDemo</display-name>
  <welcome-file-list>
    <welcome-file>index.html</welcome-file>
  </welcome-file-list>

  <servlet>
    <servlet-name>MyRestServlet</servlet-name>
    <servlet-class>org.glassfish.jersey.servlet.ServletContainer</servlet-class>

    <init-param>
        <param-name>jersey.config.server.provider.packages</param-name>
        <param-value>com.estore.rest</param-value>
    </init-param>

    <load-on-startup>1</load-on-startup>

  </servlet>

  <servlet-mapping>
    <servlet-name>MyRestServlet</servlet-name>
  </servlet-mapping>

</web-app>
```

3.11 Add the URL pattern as **/rest/***


Applications
webservice - RESTfulWS...
Maven Repository: org.g...
webservice - RESTfulWSDemo/src/main/webapp/WEB-INF/web.xml - Eclipse IDE
File Edit Source Navigate Search Project Run Window Help
UserService.java
RESTfulWSDemo/pom.xml
*web.xml
index.html
Design
Source
web-app/servlet-mapping/url-pattern
Writable
Smart Insert
23 : 25 : 862

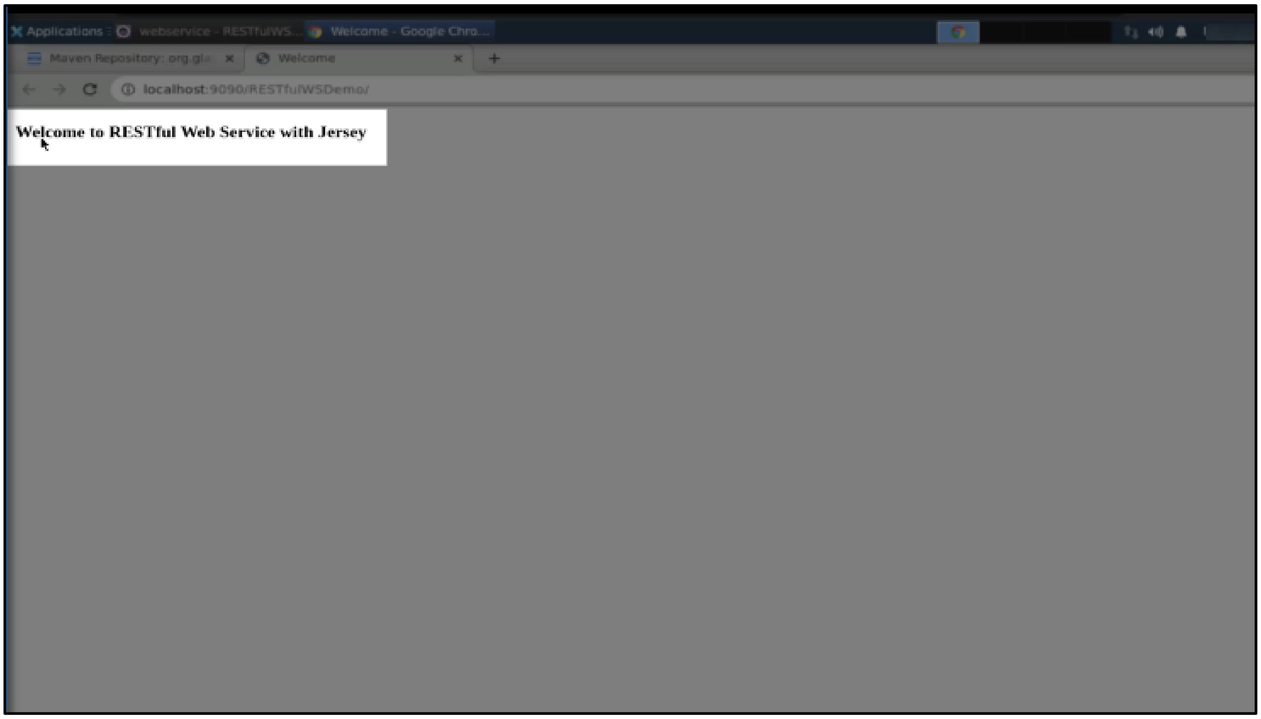
```
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="http://xmlns.jcp.org/xml/ns/javaee" xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/
  <display-name>RESTfulWSDemo</display-name>
  <welcome-file-list>
    <welcome-file>index.html</welcome-file>
  </welcome-file-list>

  <servlet>
    <servlet-name>MyRestServlet</servlet-name>
    <servlet-class>org.glassfish.jersey.servlet.ServletContainer</servlet-class>

    <init-param>
        <param-name>jersey.config.server.provider.packages</param-name>
        <param-value>com.estore.rest</param-value>
    </init-param>

    <load-on-startup>1</load-on-startup>

  </servlet>

  <servlet-mapping>
    <servlet-name>MyRestServlet</servlet-name>
    <url-pattern>/rest/*</url-pattern>
  </servlet-mapping>

</web-app>
```

3.12 Right-click on the project, click **Run As,** and select **Run on Server**



```
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="htt
  <display-name>RESTfulWSDemo</display-name>
  <welcome-file-list>
    <welcome-file>index.html</welcome-file>
  </welcome-file-list>

  <servlet>
    <servlet-name>MyRestServlet</servlet-name>
    <servlet-class>org.glassfish.jersey.servlet.ServletContainer</servlet

    <init-param>
        <param-name>jersey.config.server.provider.packages</param-name>
        <param-value>com.estore.rest</param-value>
    </init-param>

    <load-on-startup>1</load-on-startup>

  </servlet>

yRestServlet</servlet-name>
est/*</url-pattern>
```

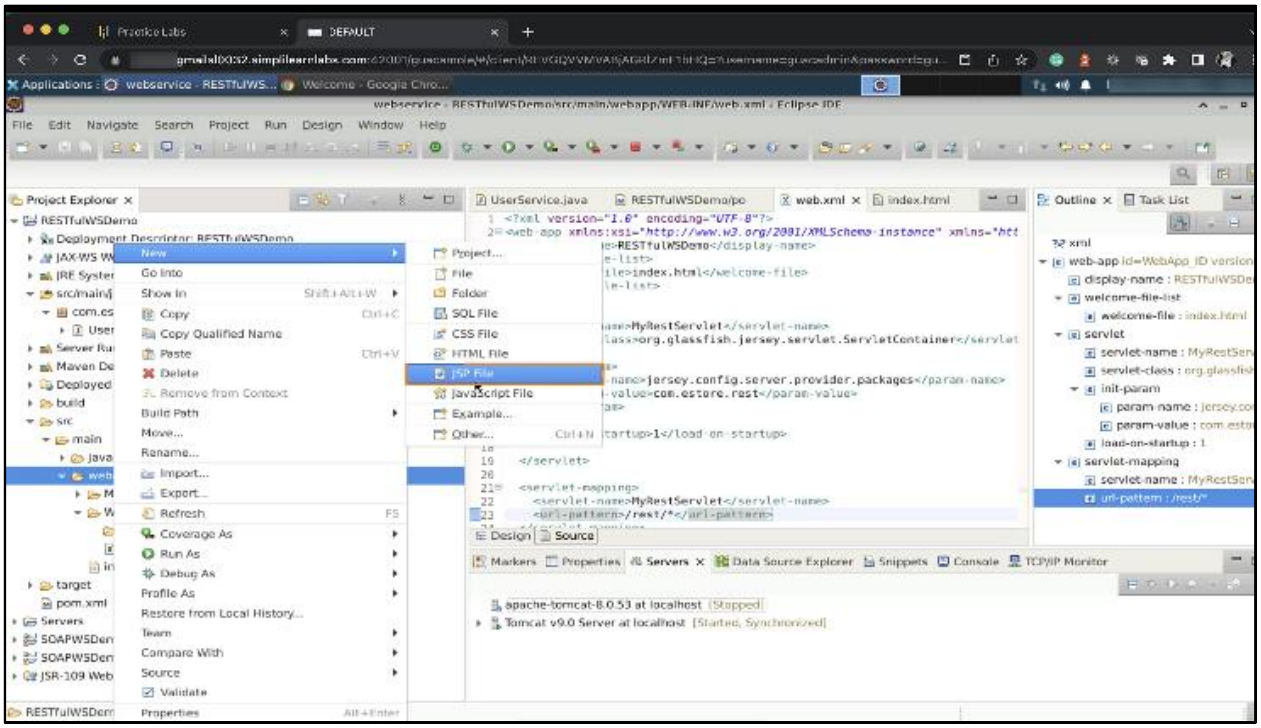
The output will be displayed as shown below:



Welcome to RESTful Web Service with Jersey

Step 4: Creating a JSP file and mapping it to a Java client file

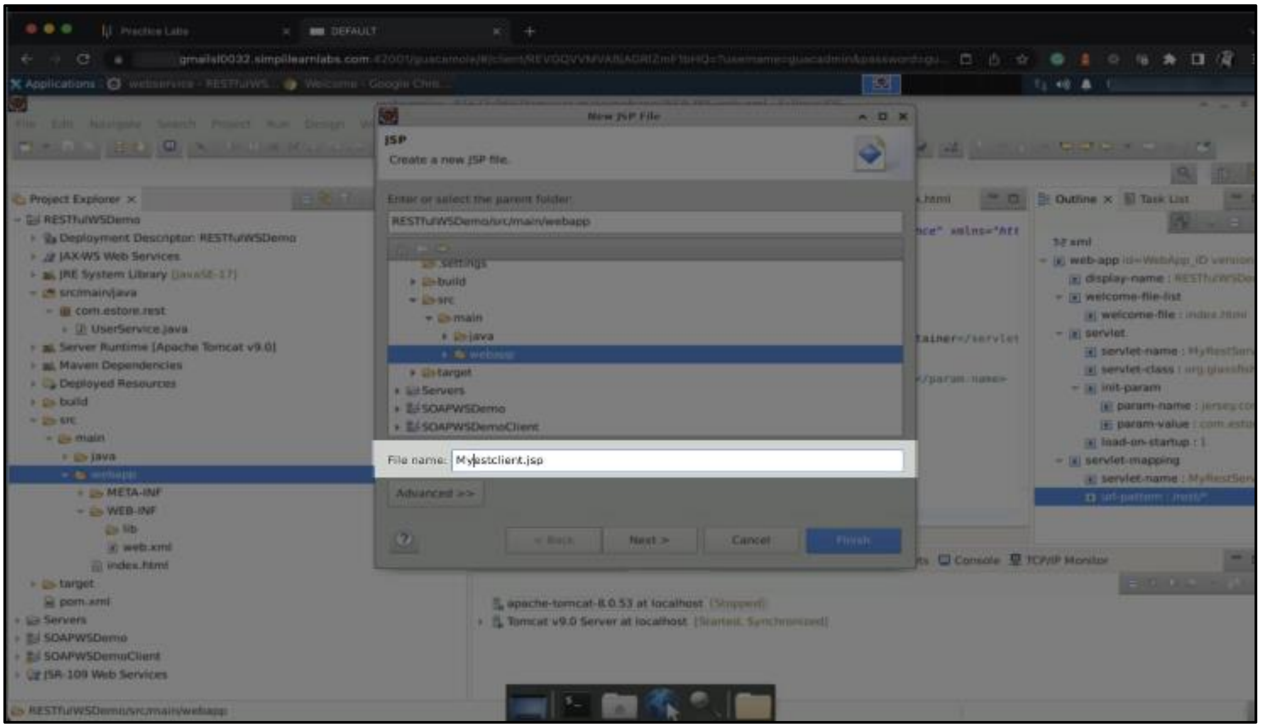
4.1 Create a JSP file under the web app



```
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="htt
e>RESTfulWSDemo</display-name>
e-list>
ile>index.html</welcome-file>
le-list>

ame>MyRestServlet</servlet-name>
lass>org.glassfish.jersey.servlet.ServletContainer</servlet

name>jersey.config.server.provider.packages</param-name>
-value>com.estore.rest</param-value>

tartup>1</load-on-startup>

  </servlet>

  <servlet-mapping>
    <servlet-name>MyRestServlet</servlet-name>
    <url-pattern>/rest/*</url-pattern>
```

```
apache-tomcat-8.0.53 at localhost [Stopped]
Tomcat v9.0 Server at localhost [Started, Synchronized]
```

4.2 Name the file as **MyRestClient** and click **Next**



Enter or select the parent folder: RESTfulWSDemo/src/main/webapp

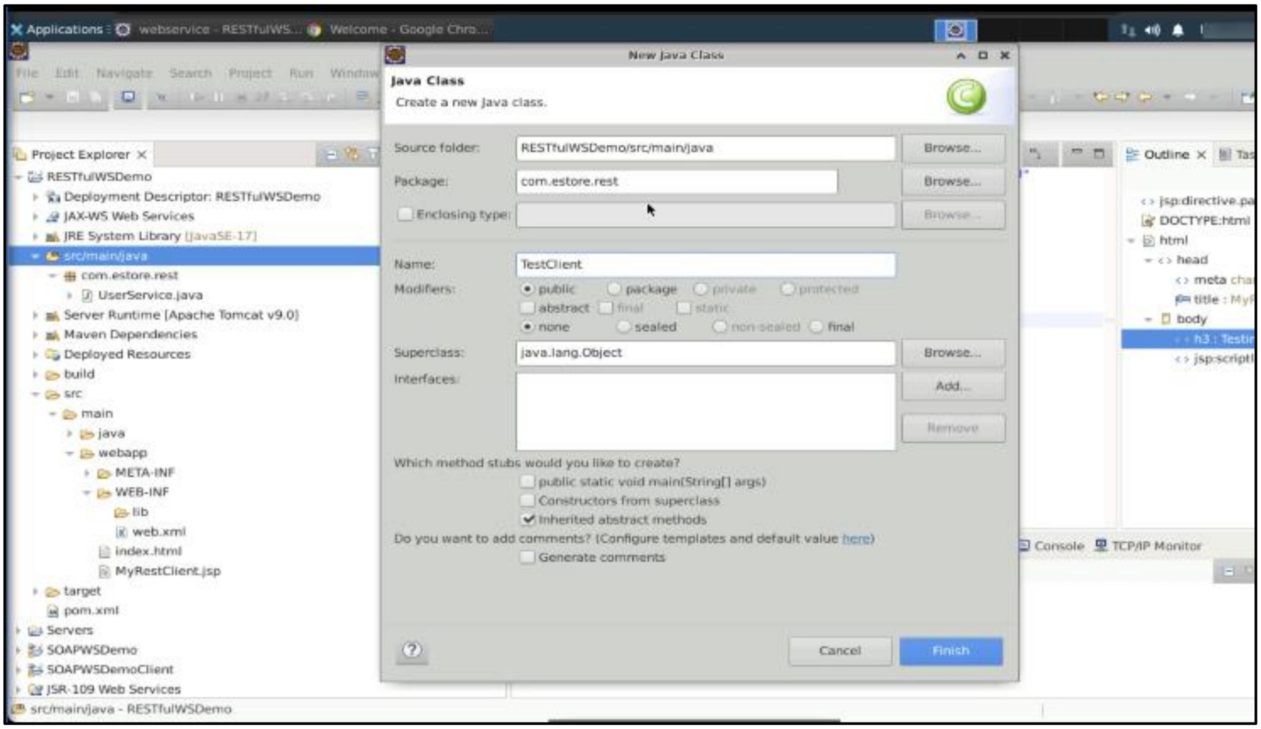
File name: Myestclient.jsp

4.3 Add the code given below in the **MyRestClient** file:

```
<%@page import="java.net.URI"%>
<%@page import="jakarta.ws.rs.client.WebTarget"%>
<%@page import="jakarta.ws.rs.core.UriBuilder"%>
<%@page import="jakarta.ws.rs.client.ClientBuilder"%>
<%@page import="jakarta.ws.rs.client.Client"%>
<%@page import="org.glassfish.jersey.client.ClientConfig"%>
<%@page import="com.estore.test.TestClient"%>
<%@ page language="java" contentType="text/html; charset=UTF-8"
   pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>MyRestClient</title>
</head>
<body>
```

```
	<h3>Testing Restful Service:</h3>
	<%!
	
		/*ClientConfig configuration = new ClientConfig();
		Client client = ClientBuilder.newClient(configuration);
		URI uri =
UriBuilder.fromUri("http://localhost:9090/RESTfulWSDemo").build();
		WebTarget target = client.target(uri);*/
		TestClient client = new TestClient();
	%>
	Plain Response: <%= client.getPlainResponse() %>
	<br>
	HTML Response: <%= client.getHTMLResponse() %>
	<br>
	XML Response: <%= client.getXMLResponse() %>
	<br>
	JSON Response: <%= client.getJSONResponse() %>
	<br>

</body>
</html>
```

4.4 Create a new Java class and name the class as **TestClient**



New Java Class

Java Class

Create a new Java class.

Source folder: RESTfulWSDemo/src/main/java

Package: com.estore.rest

□ Enclosing type:

Name: TestClient

Modifiers: ● public ○ package ○ private ○ protected

□ abstract □ final □ static

● none ○ sealed ○ non-sealed ○ final

Superclass: java.lang.Object

Interfaces:

Which method stubs would you like to create?

□ public static void main(String[] args)

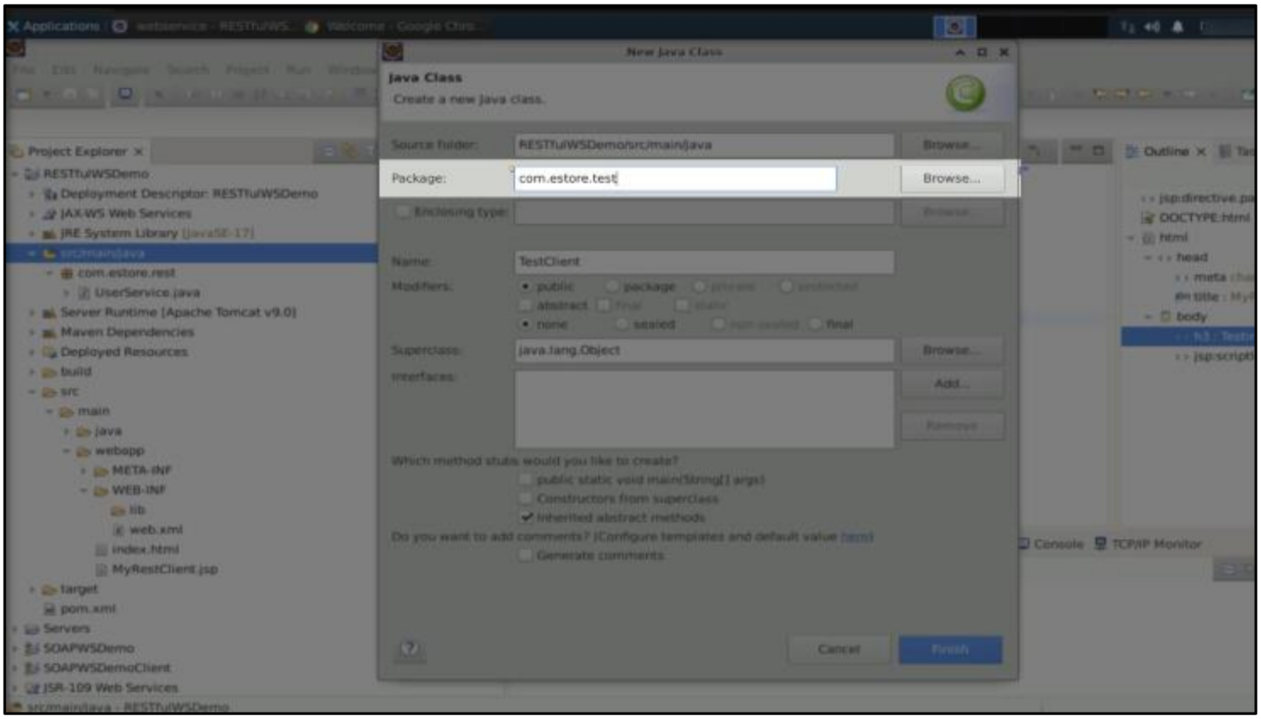
□ Constructors from superclass

☑ Inherited abstract methods

Do you want to add comments? (Configure templates and default value here)

□ Generate comments

4.5 Add the package name as **com.estore.test**



New Java Class

Java Class

Create a new Java class.

Source folder: RESTfulWSDemo/src/main/java

Package: com.estore.test

□ Enclosing type:

Name: TestClient

Modifiers: ● public ○ package ○ private ○ protected

□ abstract □ final □ static

● none ○ sealed ○ non-sealed ○ final

Superclass: java.lang.Object

Interfaces:

Which method stubs would you like to create?

□ public static void main(String[] args)

□ Constructors from superclass

☑ Inherited abstract methods

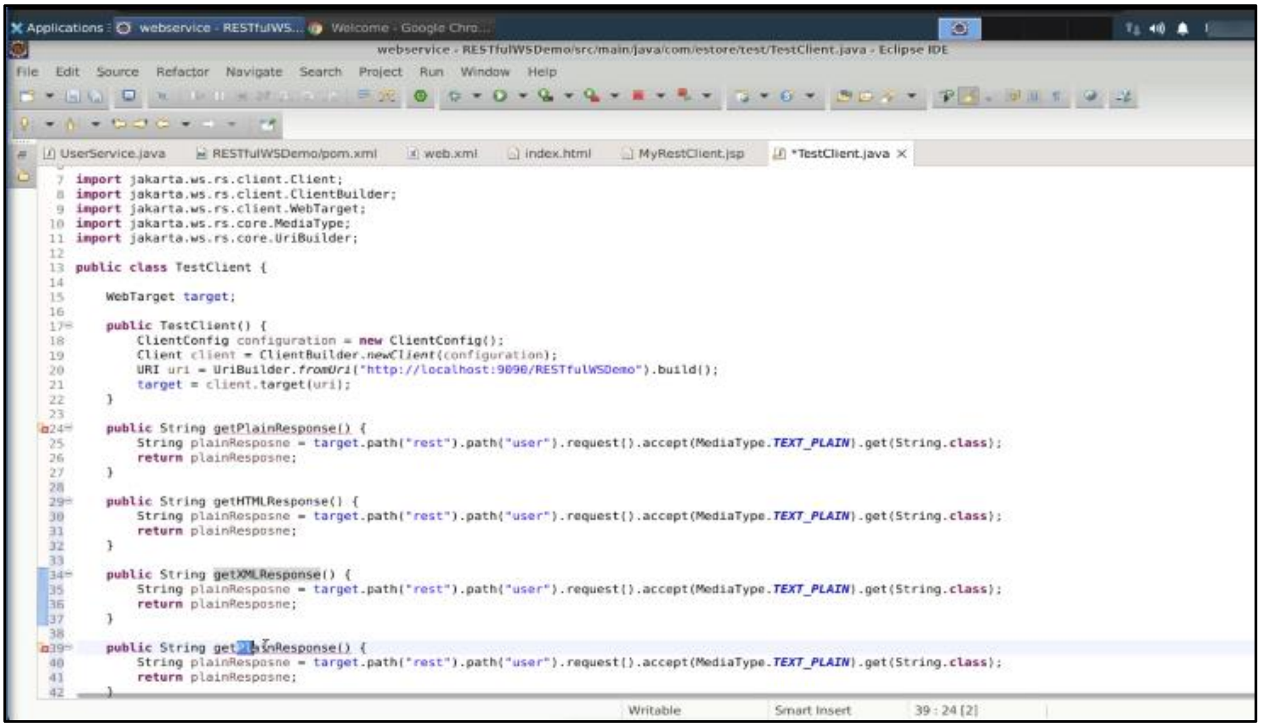
Do you want to add comments? (Configure templates and default value here)

□ Generate comments

4.6 Add the code given below in **TestClient:**

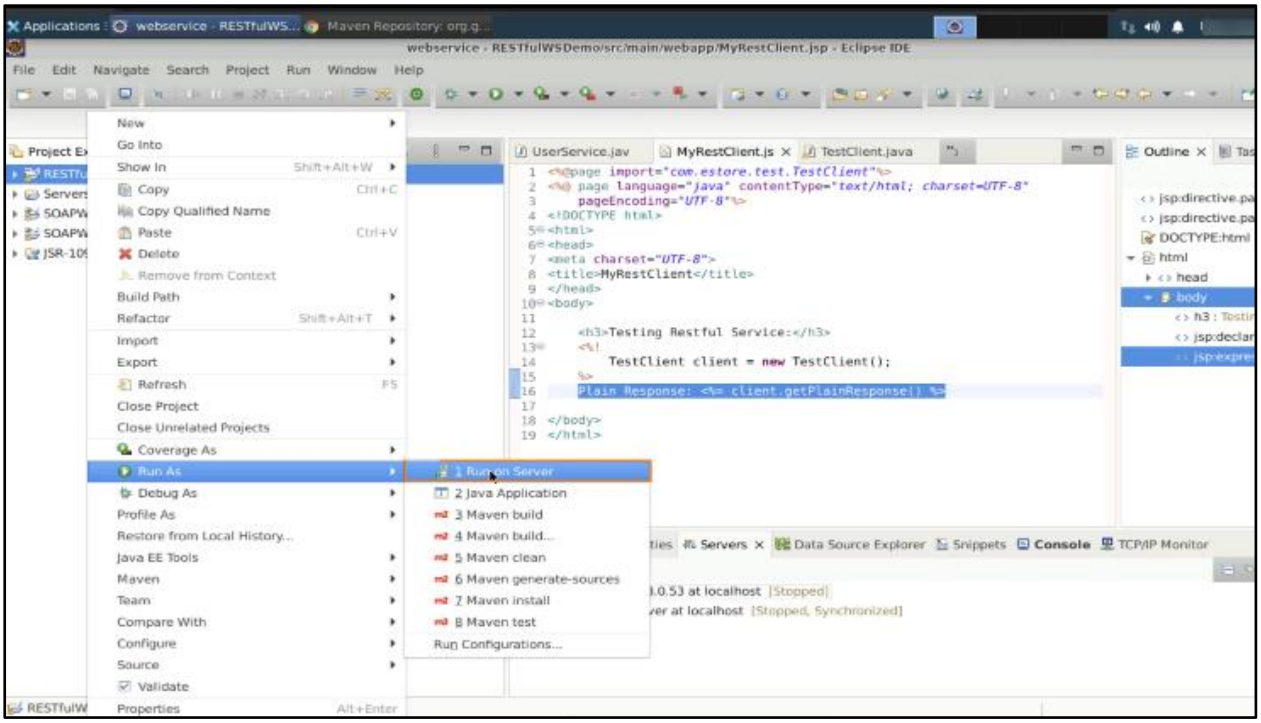
```
package com.estore.test;

import java.net.URI;
import java.util.Date;

import javax.print.attribute.standard.Media;

import org.glassfish.jersey.client.ClientConfig;

import jakarta.ws.rs.client.Client;
import jakarta.ws.rs.client.ClientBuilder;
import jakarta.ws.rs.client.WebTarget;
import jakarta.ws.rs.core.MediaType;
import jakarta.ws.rs.core.UriBuilder;

public class TestClient {

        WebTarget target;

        public TestClient() {
                ClientConfig configuration = new ClientConfig();
                Client client = ClientBuilder.newClient(configuration);
                URI uri =
UriBuilder.fromUri("http://localhost:9090/RESTfulWSDemo").build();
                target = client.target(uri);
        }

        public String getPlainResponse() {
                String response =
target.path("rest").path("user").request().accept(MediaType.TEXT_PLAIN).get(String.
class);
                return response;
        }

        public String getHTMLResponse() {
```

```
		String response =
target.path("rest").path("user").request().accept(MediaType.TEXT_HTML).get(String.class);
		return response;
	}

	public String getXMLResponse() {
		String response =
target.path("rest").path("user").request().accept(MediaType.TEXT_XML).get(String.class);
		return response;
	}

	public String getJSONResponse() {
		String response =
target.path("rest").path("user").request().accept(MediaType.APPLICATION_JSON).get(String.class);
		return response;
	}

}
```


Applications
webservice - RESTfulWS...
Welcome - Google Chro...
webservice - RESTfulWSDemo/src/main/java/com/estore/test/TestClient.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
UserService.java
RESTfulWSDemo/pom.xml
web.xml
index.html
MyRestClient.jsp
*TestClient.java

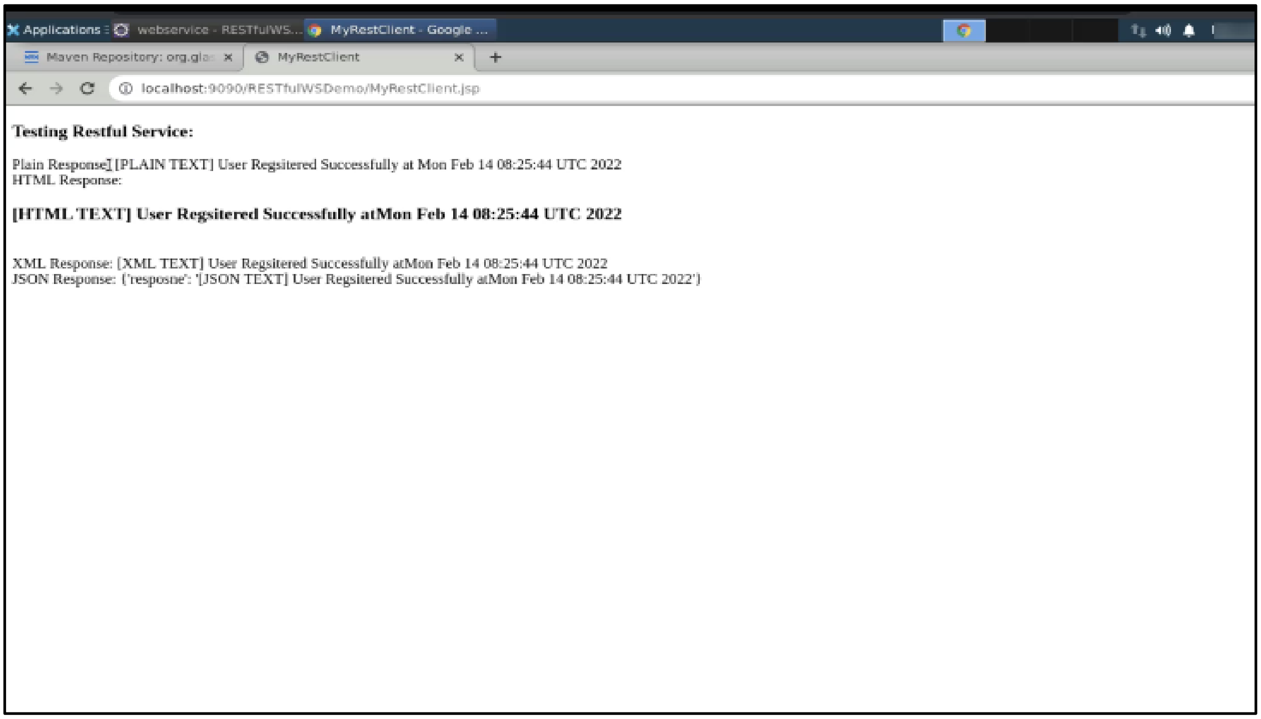
```
import jakarta.ws.rs.client.Client;
import jakarta.ws.rs.client.ClientBuilder;
import jakarta.ws.rs.client.WebTarget;
import jakarta.ws.rs.core.MediaType;
import jakarta.ws.rs.core.UriBuilder;

public class TestClient {

    WebTarget target;

    public TestClient() {
        ClientConfig configuration = new ClientConfig();
        Client client = ClientBuilder.newClient(configuration);
        URI uri = UriBuilder.fromUri("http://localhost:9090/RESTfulWSDemo").build();
        target = client.target(uri);
    }

    public String getPlainResponse() {
        String plainResposne = target.path("rest").path("user").request().accept(MediaType.TEXT_PLAIN).get(String.class);
        return plainResposne;
    }

    public String getHTMLResponse() {
        String plainResposne = target.path("rest").path("user").request().accept(MediaType.TEXT_PLAIN).get(String.class);
        return plainResposne;
    }

    public String getXMLResponse() {
        String plainResposne = target.path("rest").path("user").request().accept(MediaType.TEXT_PLAIN).get(String.class);
        return plainResposne;
    }

    public String getPlainResponse() {
        String plainResposne = target.path("rest").path("user").request().accept(MediaType.TEXT_PLAIN).get(String.class);
        return plainResposne;
    }
```

4.7 Run the code on the server



```
<%@page import="com.estore.test.TestClient"%>
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>MyRestClient</title>
</head>
<body>

    <h3>Testing Restful Service:</h3>
    <%!
        TestClient client = new TestClient();
    %>
    Plain Response: <%= client.getPlainResponse() %>

</body>
</html>
```

The output will show the successful addition of the RESTful service:



localhost:9090/RESTfulWSDemo/MyRestClient.jsp

```
Testing Restful Service:

Plain Response] [PLAIN TEXT] User Regsitered Successfully at Mon Feb 14 08:25:44 UTC 2022
HTML Response:

[HTML TEXT] User Regsitered Successfully atMon Feb 14 08:25:44 UTC 2022


XML Response: [XML TEXT] User Regsitered Successfully atMon Feb 14 08:25:44 UTC 2022
JSON Response: {'resposne': '[JSON TEXT] User Regsitered Successfully atMon Feb 14 08:25:44 UTC 2022'}
```

Applications
webservice - RESTfulWS...
MyRestClient - Google ...
Maven Repository: org.gla
MyRestClient