JAVA EE COURSE

SECURING THE WEB LAYER WITH JAVA EE

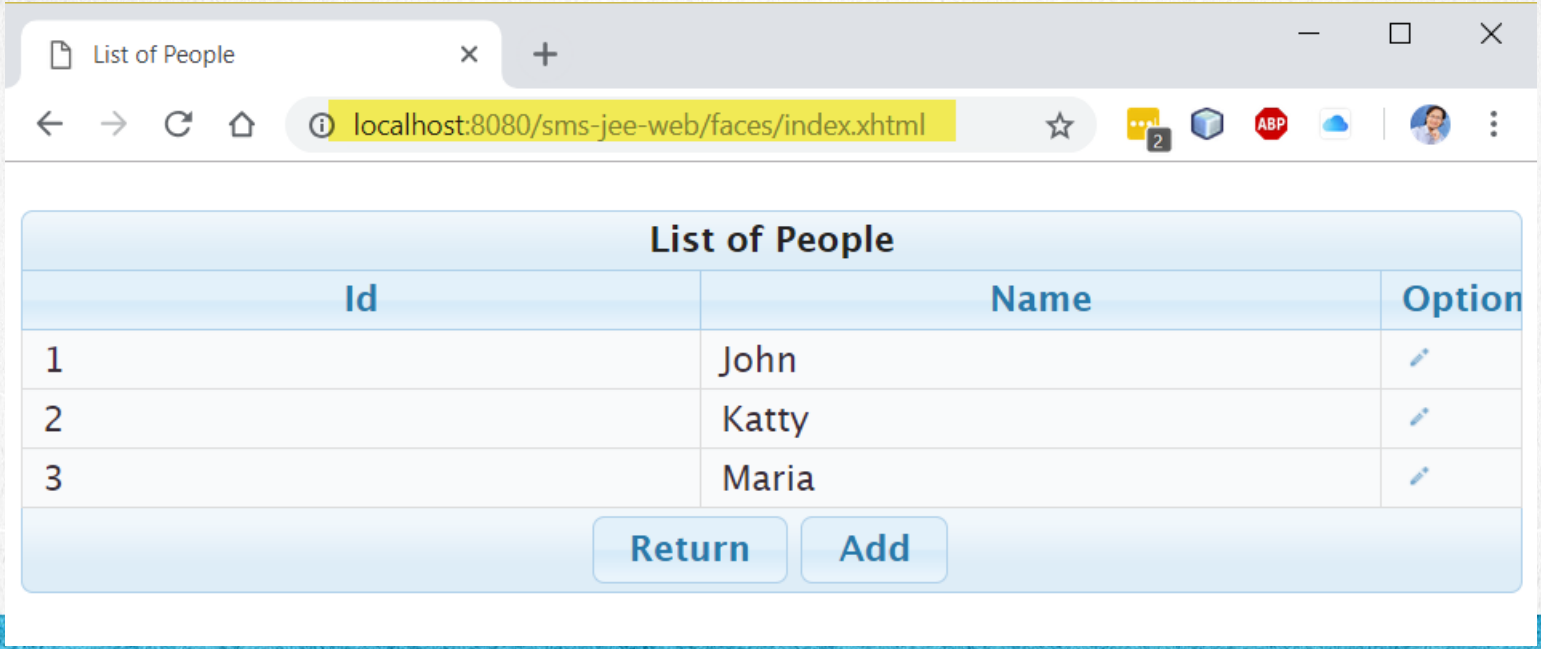
By the expert: Eng. Ubaldo Acosta

Global Mentoring

JAVA ONLINE COURSES UNIVERSITY

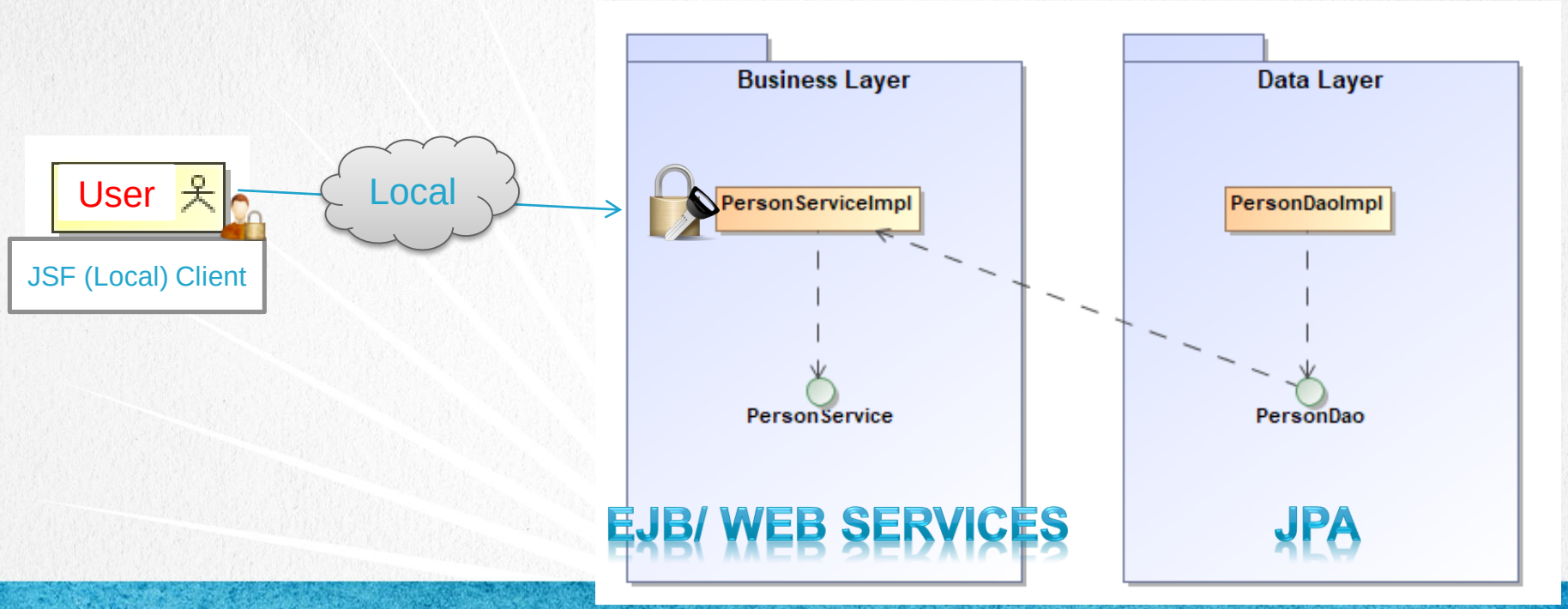
EXERCISE OBJECTIVE

•The objective of the exercise is to ensure the Web layer of the SMS project. The result is shown below (data may vary). This is the continuation of the previous exercise:



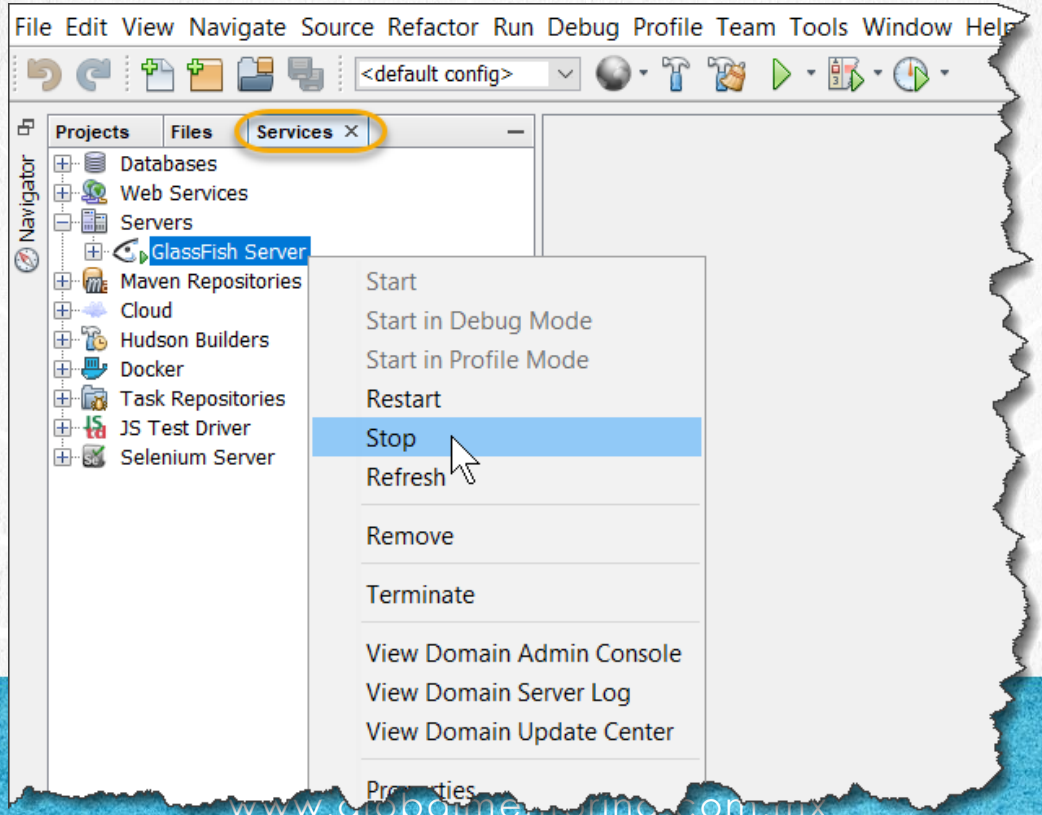
ARCHITECTURE WITH SECURE WEB LAYER

This is the Exercise Class Diagram, where you can see the Architecture of our System:



STOP GLASSFISH

Stop the Glassfish server:



1. MODIFY THE WEB CLIENT

We modified the web.xml file. We add security restrictions to be able to login from the web client:

```
<login-config>
   <auth-method>BASIC</auth-method>
   <realm-name>Login in</realm-name>
</login-config>

<security-constraint>
   <web-resource-collection>
      <web-resource-name>JSF Web Application</web-resource-name>
      <url-pattern>/*</url-pattern>
   </web-resource-collection>

   <auth-constraint>
      <role-name>ROLE_ADMIN</role-name>
      <role-name>ROLE_USER</role-name>
      <role-name>ROLE_GUEST</role-name>
   </auth-constraint>
</security-constraint>
```

1. MODIFY THE FILE

web.xml:

Click to download

```xml
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns="http://xmlns.jcp.org/xml/ns/javaee"
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/javaee http://xmlns.jcp.org/xml/ns/javaee/web-app_4_0.xsd"
         version="4.0">
    <context-param>
        <param-name>primefaces.THEME</param-name>
        <param-value>cupertino</param-value>
    </context-param>
    <welcome-file-list>
        <welcome-file>faces/index.xhtml</welcome-file>
    </welcome-file-list>
    <servlet>
        <servlet-name>JerseyWebApplication</servlet-name>
        <servlet-class>
            org.glassfish.jersey.servlet.ServletContainer
        </servlet-class>
        <init-param>
            <param-name>jersey.config.server.provider.packages</param-name>
            <param-value>sms.service.rest</param-value>
        </init-param>
        <load-on-startup>1</load-on-startup>
    </servlet>
    <servlet-mapping>
        <servlet-name>JerseyWebApplication</servlet-name>
        <url-pattern>/webservice/*</url-pattern>
    </servlet-mapping>
```

1. MODIFY THE FILE

web.xml:

Click to download

```
    <login-config>
        <auth-method>BASIC</auth-method>
        <realm-name>Login in</realm-name>
    </login-config>
    <security-constraint>
        <web-resource-collection>
            <web-resource-name>JSF Web Application</web-resource-name>
            <url-pattern>/*</url-pattern>
        </web-resource-collection>
        <auth-constraint>
            <role-name>ROLE_ADMIN</role-name>
            <role-name>ROLE_USER</role-name>
            <role-name>ROLE_GUEST</role-name>
        </auth-constraint>
    </security-constraint>
</web-app>
```

2. MODIFY THE INDEX.XHTML FILE

Modify the index.xhtml file to add the security tags of primefaces:

```
Welcome: <h:outputText value="#{p:userPrincipal()}"/>


<h:panelGrid columns="1" rendered="#{p:ifGranted('ROLE_ADMIN')||p:ifGranted('ROLE_USER')}">
    <h:commandButton value="List of People" action="listPeople" />
</h:panelGrid>
```

When executing the application we will be able to observe the name of the user that was authenticated, and if the user has the role of ROLE_ADMIN or ROLE_USER, he will be able to see the List People button, otherwise this button will not be displayed.

If we test with the guest user, we will not see the button to list people, and even if we could see it or execute directly the URL http://localhost:8080/sms-jee-web/faces/listPeople.xhtml we could not execute any EJB method because he does not have permission for it either.

2. MODIFY THE FILE

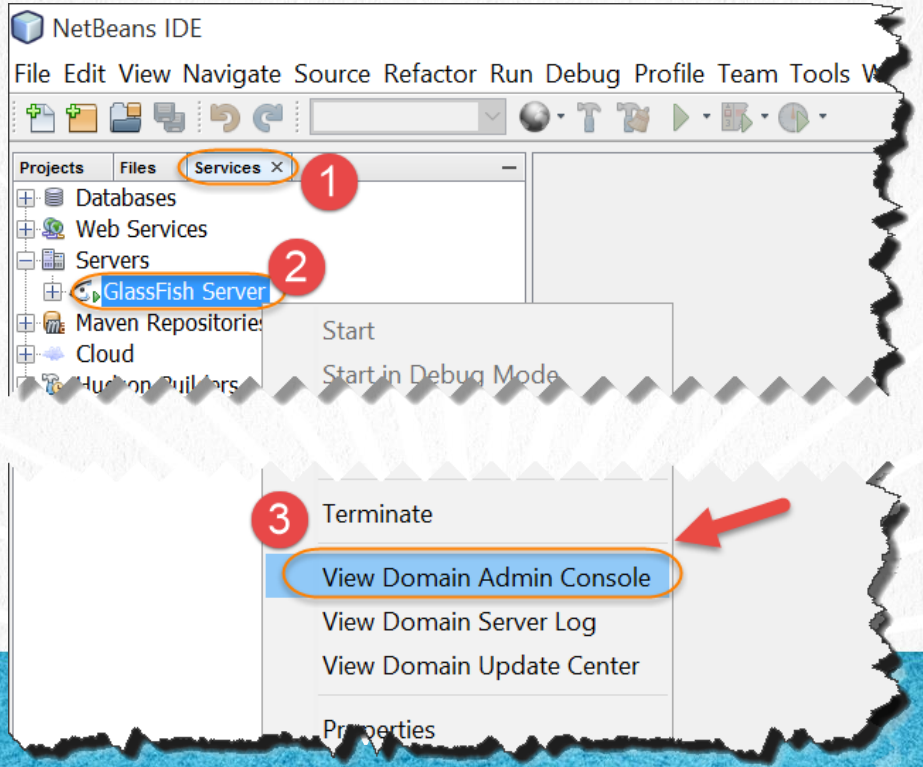
index.xhtml:

Click to download

```xhtml
<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml"
      xmlns:h="http://java.sun.com/jsf/html"
      xmlns:p="http://primefaces.org/ui">
    <h:head>
        <title>SMS System</title>
    </h:head>
    <h:body>
        <h2>SMS System</h2>
        Welcome: <h:outputText value="#{p:userPrincipal()}"/>
        <h:form>
            <p:messages />
            <h:panelGrid columns="1" rendered="#{p:ifGranted('ROLE_ADMIN')||p:ifGranted('ROLE_USER')}">
                <h:commandButton value="List People" action="listPeople" />
            </h:panelGrid>
        </h:form>
    </h:body>
</html>
```

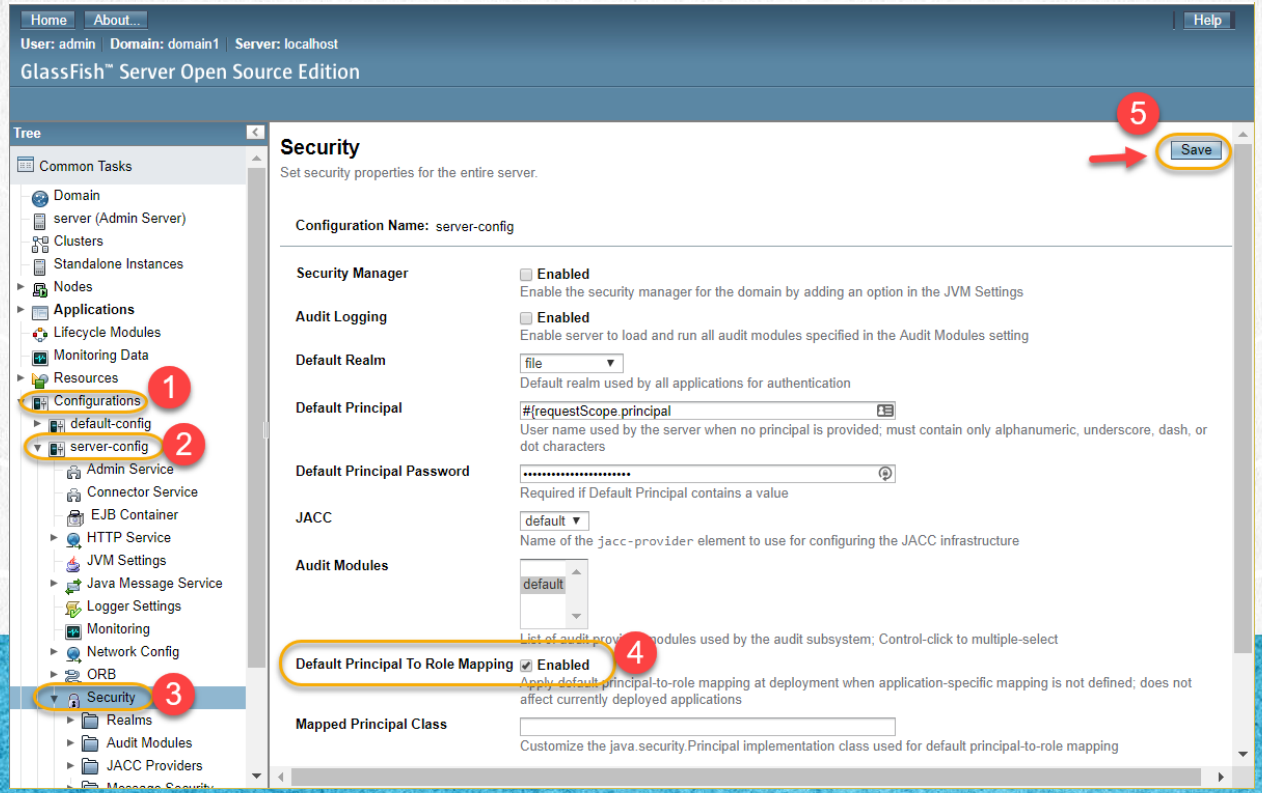
3. MODIFY THE WEB CLIENT

Access to the Glassfish console, the server must already be started:



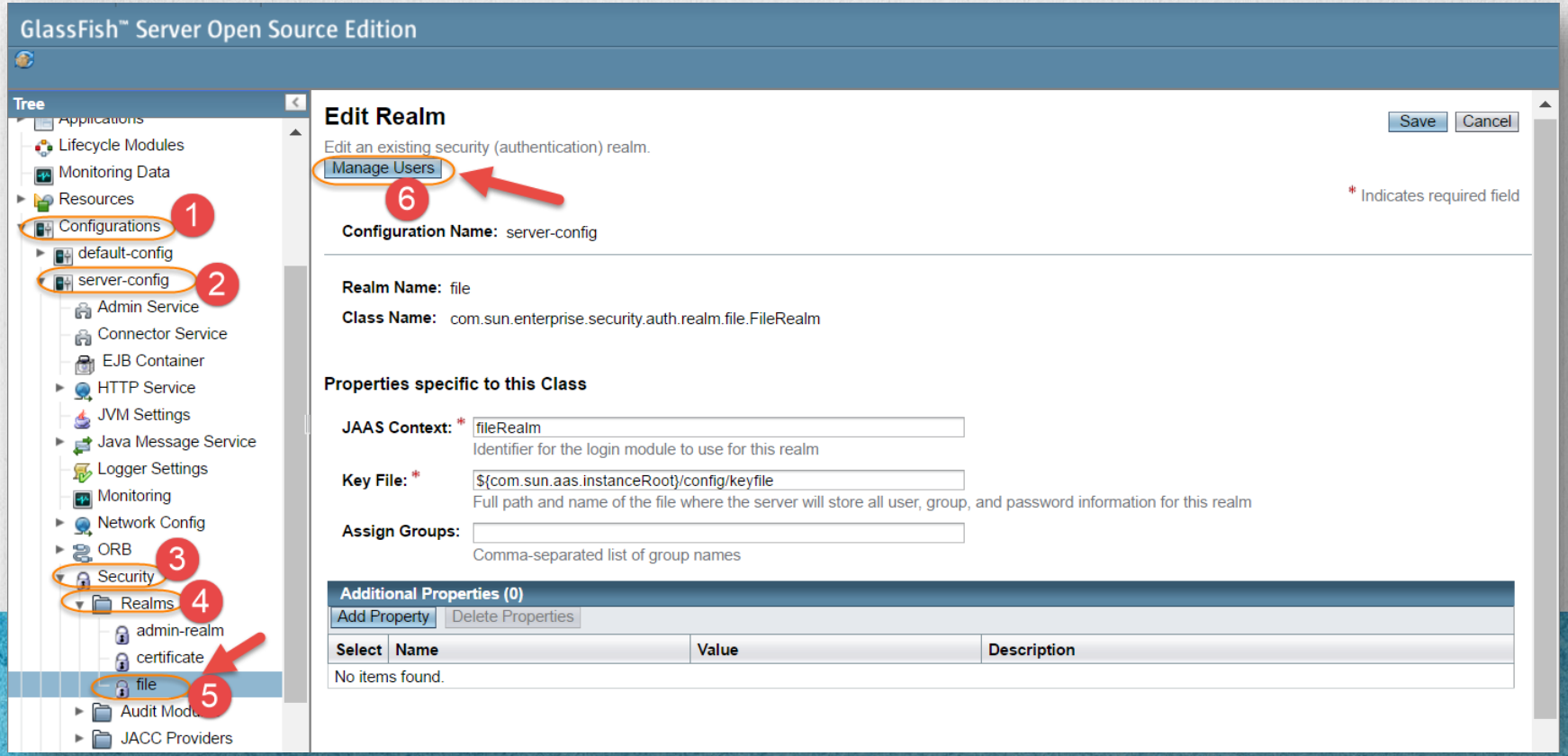
3. MODIFY THE WEB CLIENT

Configure File Realms in GlassFish. We enter the GlassFish console -> Configurations -> server-config -> Security. We enable *Default Principal To Role Mapping*. This option indicates that the name of the group is the same as that of the role, and therefore it is no longer necessary to add the file glassfish-web.xml to the project.



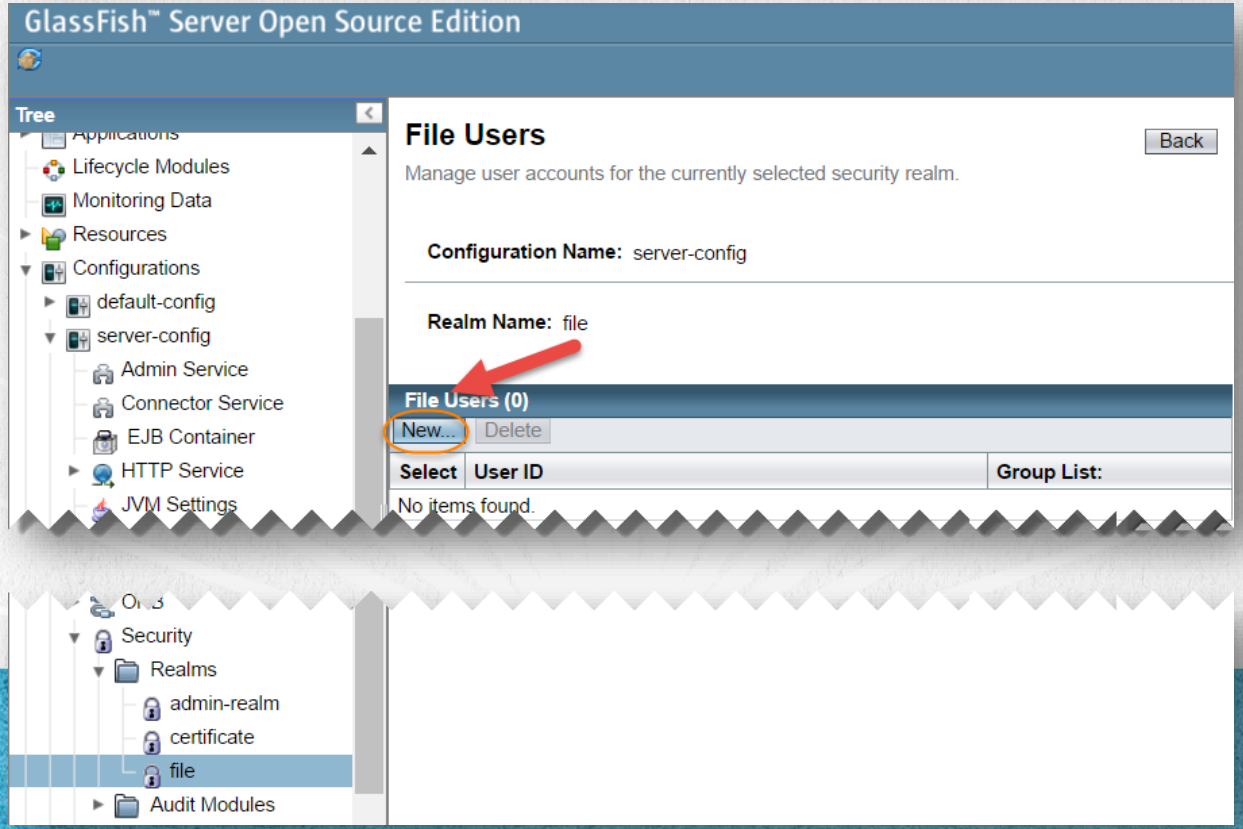
3. MODIFY THE WEB CLIENT

Now we configure the Security-> Realms -> file -> Manage Users option:



3. MODIFY THE WEB CLIENT

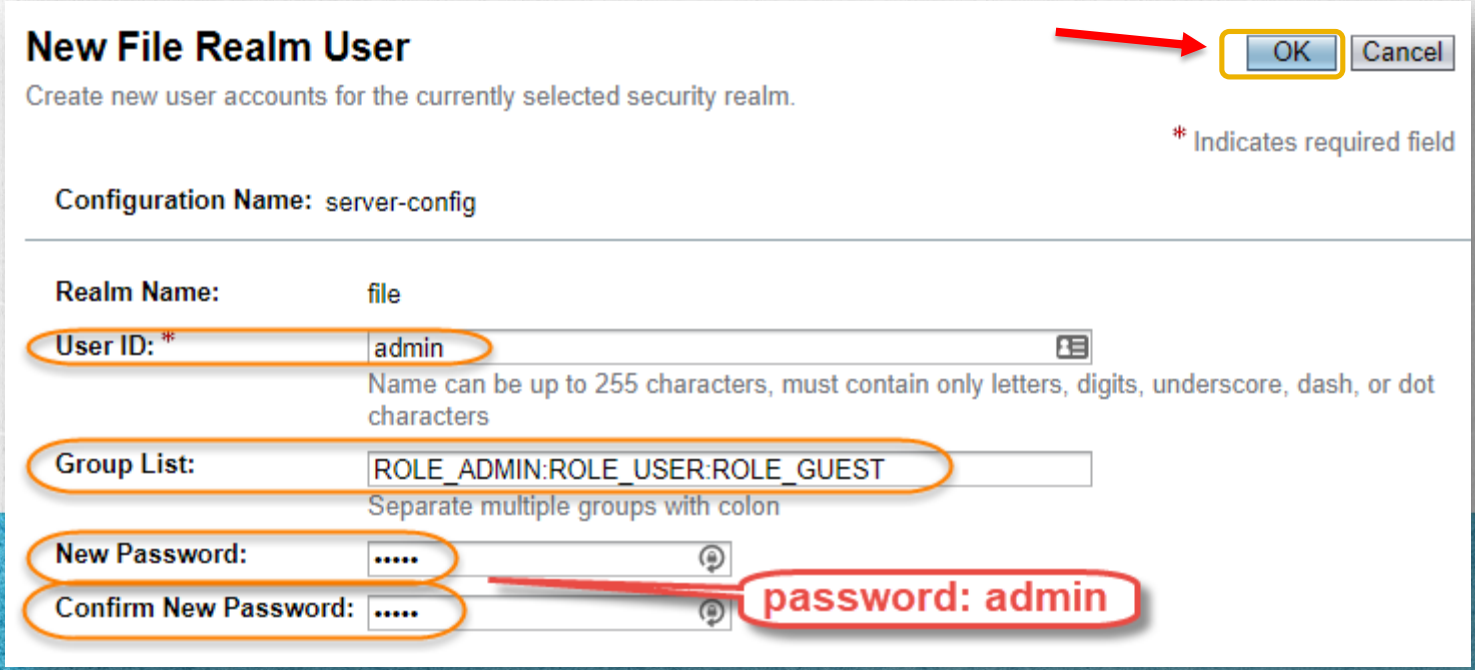
This is the user administration screen. We add a new one:



3. MODIFY THE WEB CLIENT

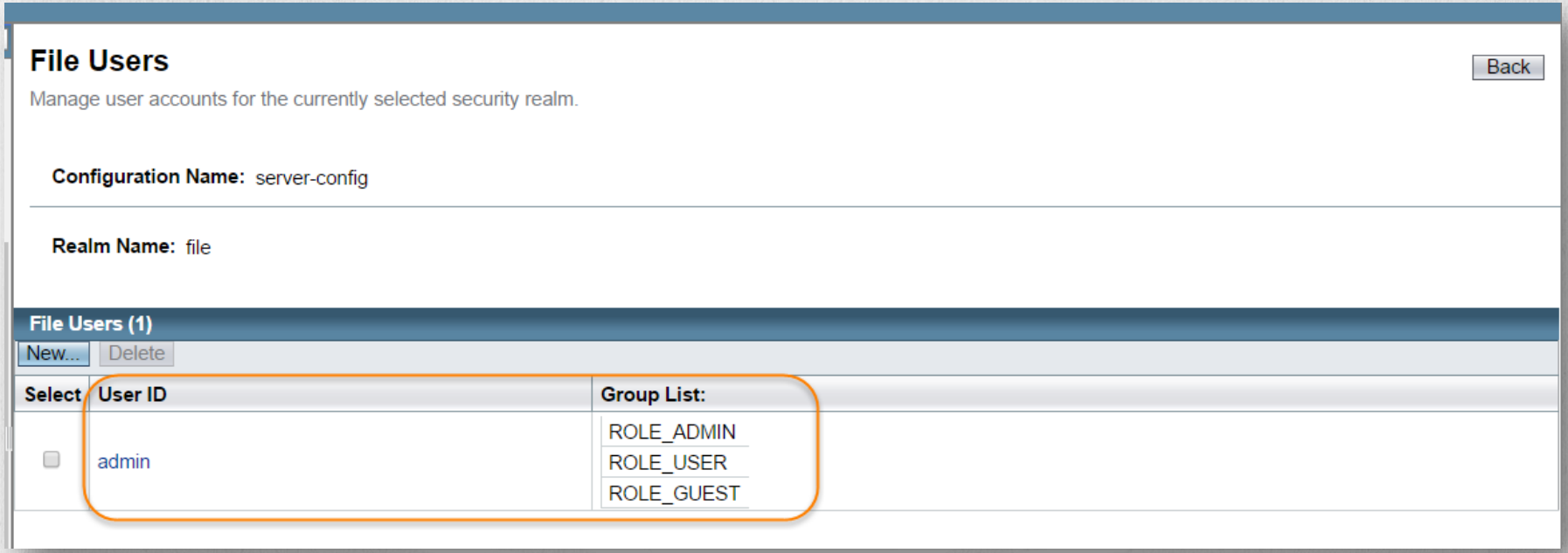
We add a user with the following data:

User Id: admin
Group List: ROLE_ADMIN:ROLE_USER:ROLE_GUEST
New Password: admin
Confirm New Password: admin



3. MODIFY THE WEB CLIENT

Result of adding the user admin:

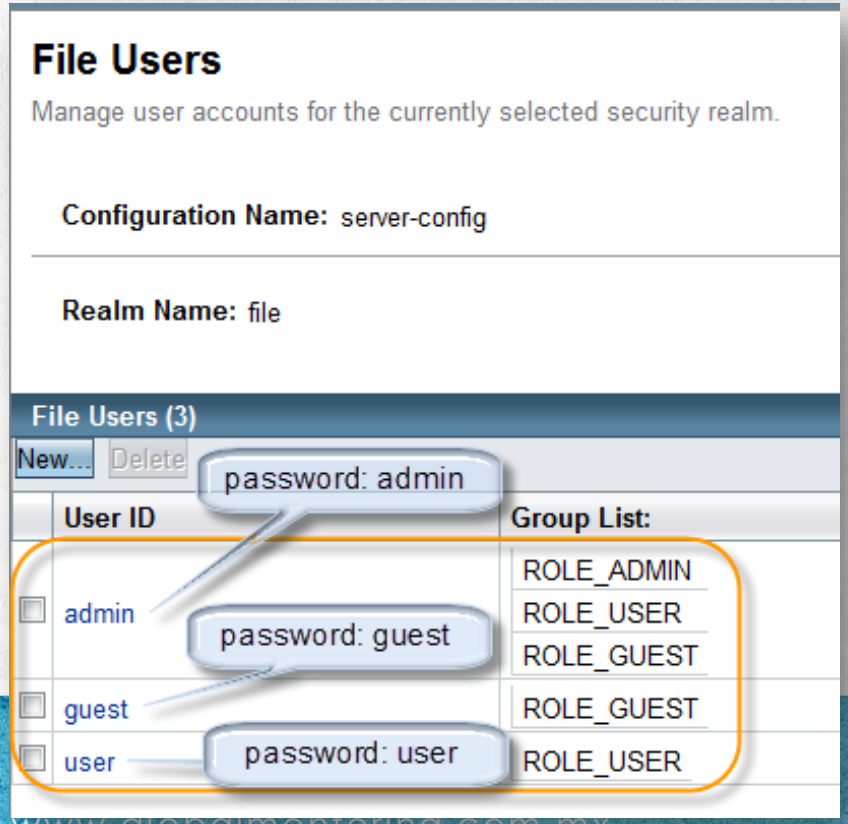


3. MODIFY THE WEB CLIENT

We repeat the previous process using the **New** button, until adding the following users, with their respective password and

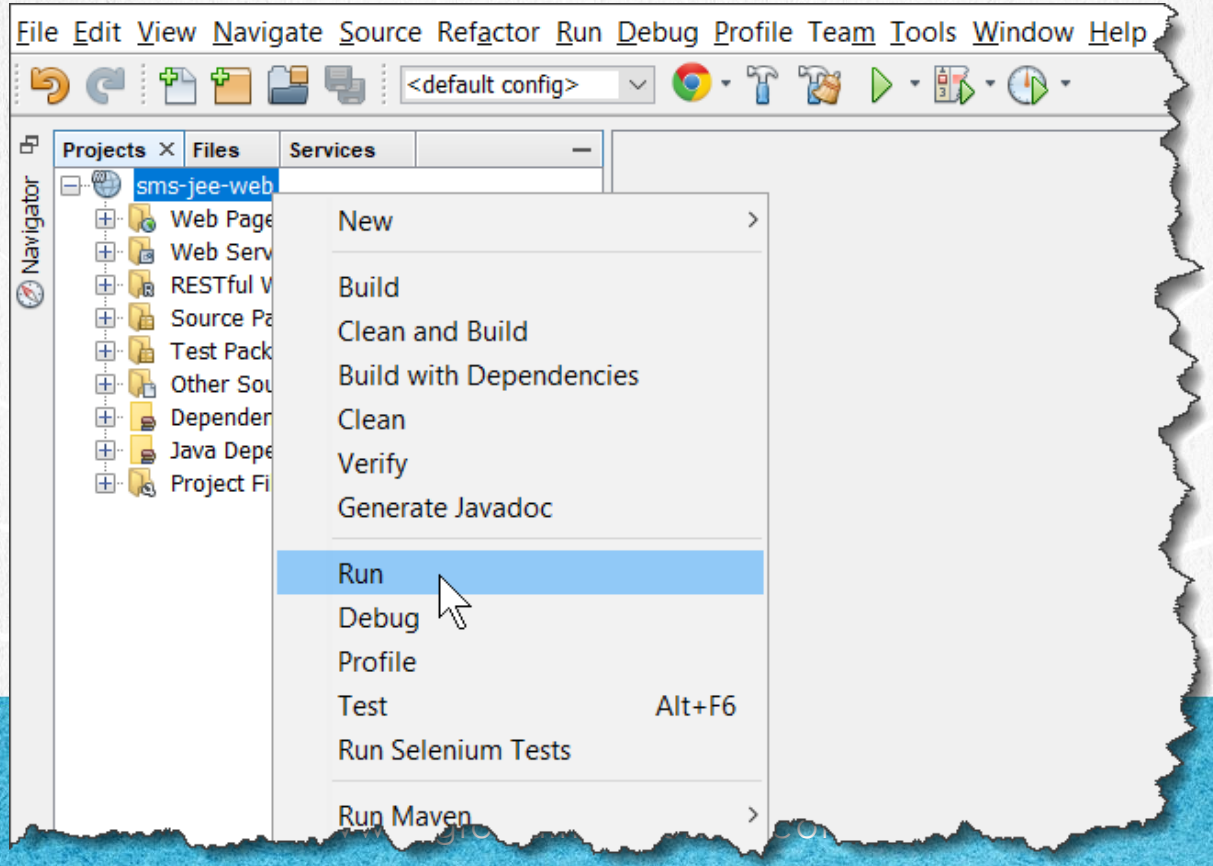
UserId: user
Group List: ROLE_USER
New Password: user
Confirm New Password: user

UserId: guest
Group List: ROLE_GUEST
New Password: guest
Confirm New Password: guest



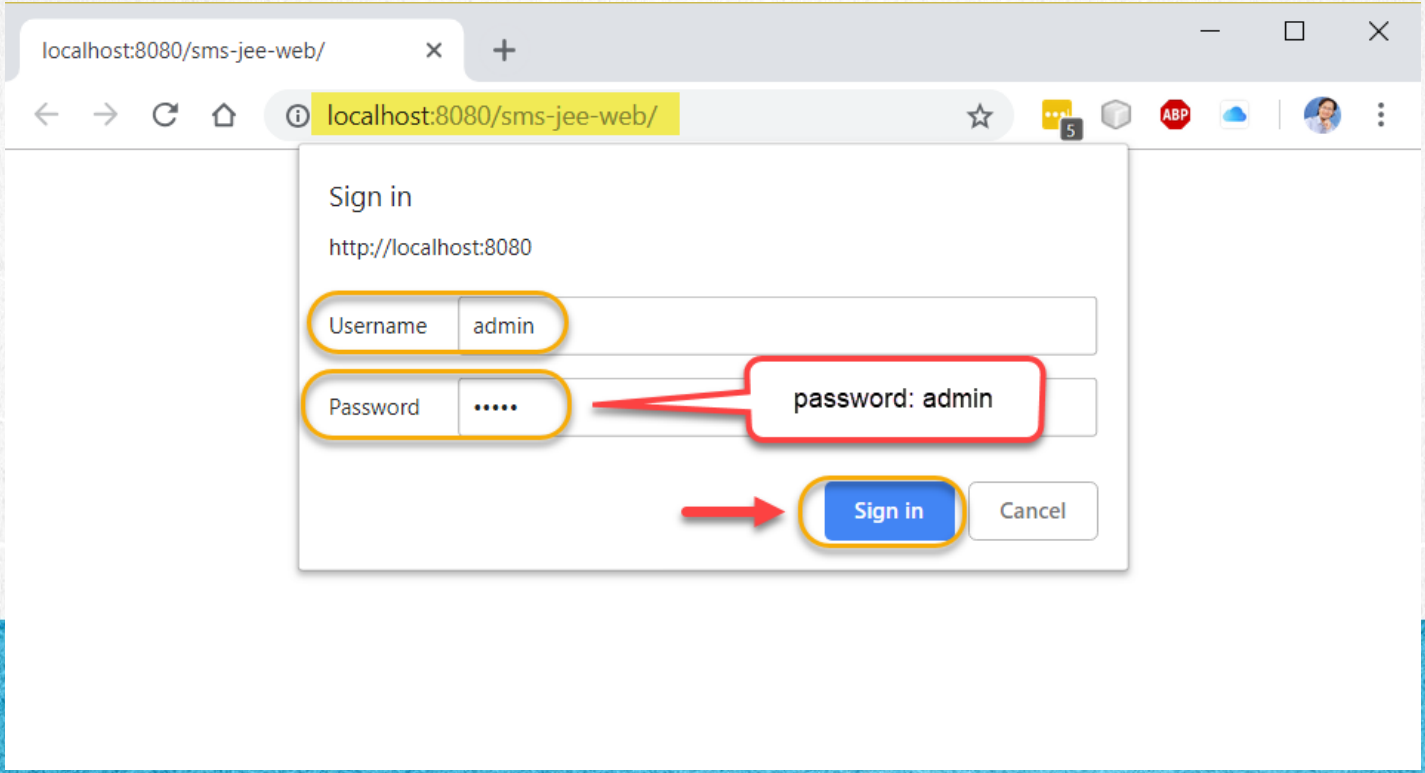
5. DEPLOY THE APPLICATION

We deploy the Web application again to see the changes:



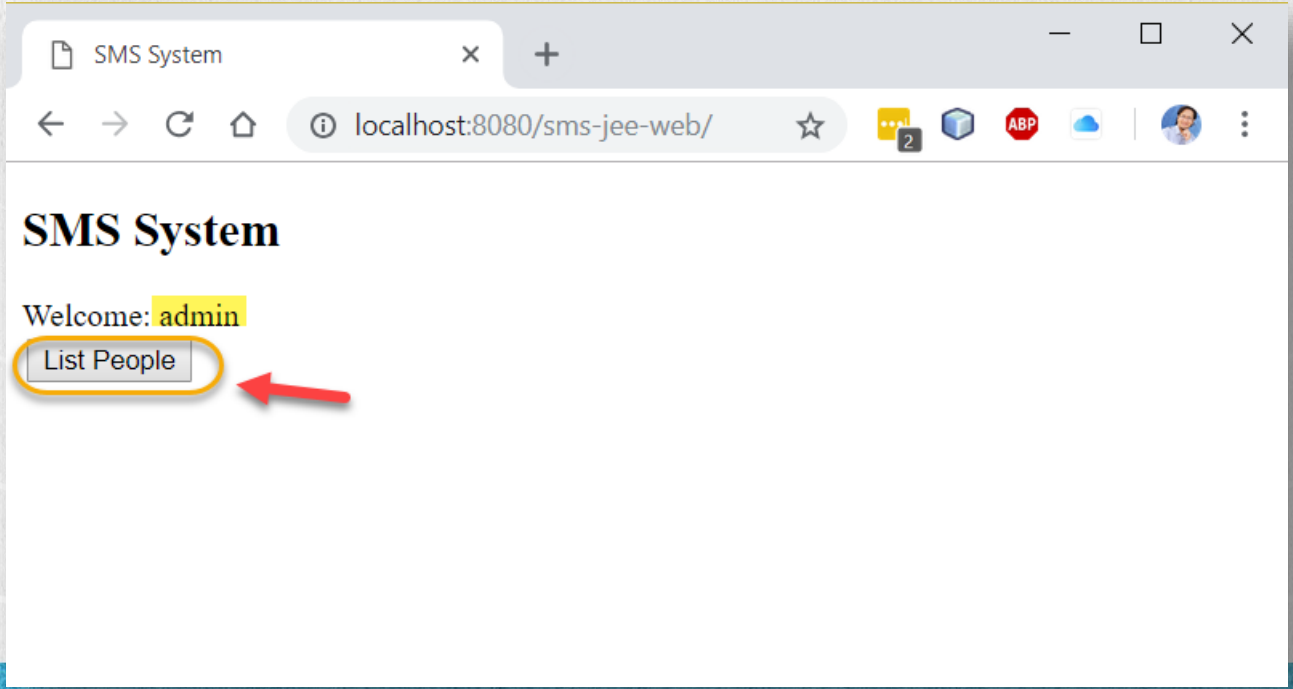
5. DEPLOY THE APPLICATION

We can test with erroneous values, and with correct values:
User: admin, Password: admin



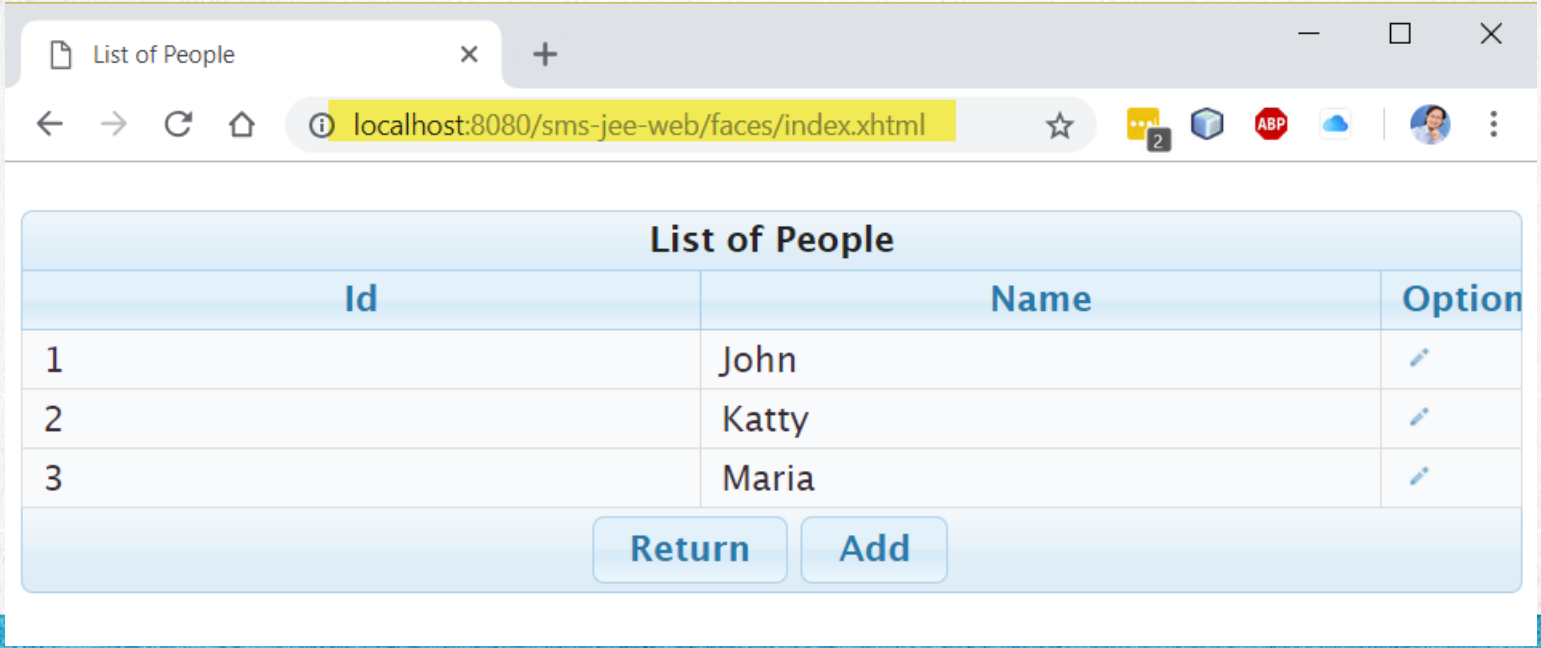
5. DEPLOY THE APPLICATION

We execute the operation of listing EJB people again and see if we already have access:



5. DEPLOY THE APPLICATION

As we can see we already have access to the EJB method, since we have provided the keys correctly and therefore we can see the list of objects of type Persona:



EXERCISE CONCLUSION

With this exercise we have added users and passwords, as well as the role assigned to the added users, in order to pass the validation requested by the respective EJB and the Web application.

We will continue with the other clients to continue validating access to the EJB method.

ONLINE COURSE

JAVA EE
JAKARTA EE

By: Eng. Ubaldo Acosta