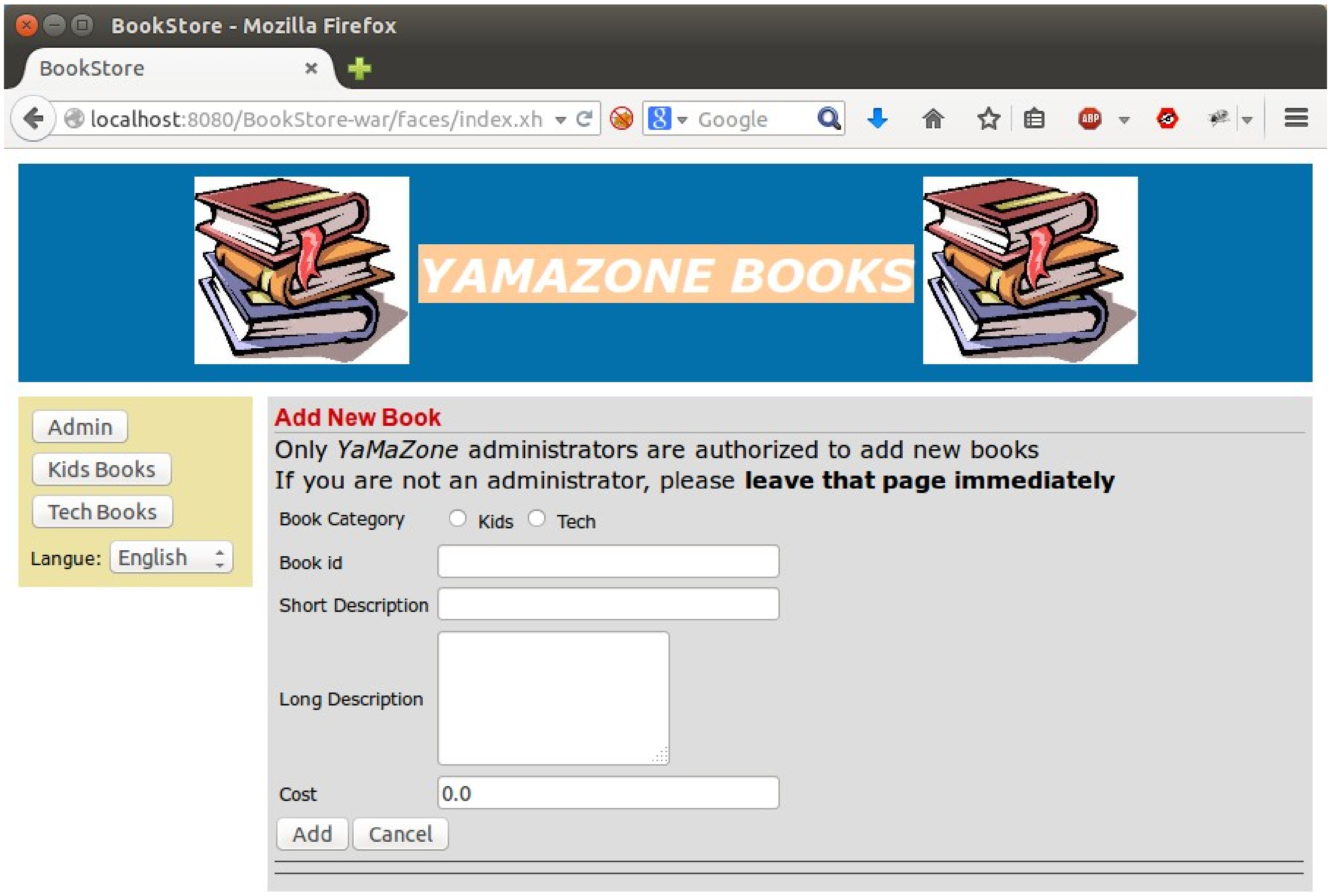
SEG3102 – Lab8

Authentication

Objective

In this lab, you will learn about design/implementation of authentication mechanisms for securing Java EE applications. You will use the Java EE Container provided Form based authentication/authorization mechanism with Glassfish and look at an example of Application provided authentication.

Form Based Authentication



Add New Book

Only *YaMaZone* administrators are authorized to add new books
If you are not an administrator, please **leave that page immediately**

Book Category	Kids Tech
Book id	
Short Description	
Long Description	
Cost	0.0

The YamaZone application doesn't provide a secure access to the Admin page. This is clearly a problem for a realistic application.Import the project from the ***bookStore.zip*** archive. If you want to use a version of the application that you already have installed from a previous lab, **make sure to replace the following line** in ***menu.xhtml***

```
<h:commandButton action="admin" value="#{msg['admin']}"/>
```

with

```
<h:button outcome="admin" value="#{msg['admin']}"/>
```

Suppose now that we want to restrict access to the Admin page so that only authenticated administrators can use it. The Java EE platform provides various authentication mechanisms that can be used :

- Basic Authentication - a browser pop up window is displayed asking the user to enter his credentials. This approach is the easiest to implement. However, passwords are not encrypted by default and the login page can not be customized.
- Digest Authentication - works like basic authentication, with the exception that passwords are encrypted when sent to the server.
- Certificate based Authentication – based on certificates issued by certificate authorities such as Verisign or Thawte .
- Form-based authentication - we need to develop a Web page used to collect user credentials. This is the most commonly used approach. The advantages include the ability to customize login pages; additionally, the user name and password can be encrypted by setting up the page to use the HTTPS (HTTP over SSL) .

Steps for implementing form-base authentication

1. Create a login page.
2. Create a login error page that will be displayed when a user enters incorrect credentials.
3. Configure the web application to use a security realm for authentication.
4. Configure the application server to define a users.

Login Page

The login page must contain

- an HTML form with a method of POST and an action of **j_security_check**.
- The form must contain a text field named **j_username**, and a password field named **j_password**.

Create a **JSF Template Client** from ***template.xhtml*** called ***loginPage.xhtml*** in ***BookStore-war*** and edit so that the contents are as follow :

```
<?xml version='1.0' encoding='UTF-8' ?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml"
      xmlns:ui="http://xmlns.jcp.org/jsf/facelets">
    <body>
        <ui:composition template="./template.xhtml">
            <ui:define name="content">
          <p>Please enter your username and password to use that function</p>
        <form method="POST" action="j_security_check">
            <table cellpadding="0" cellspacing="0" border="0">
                <tr>
                    <td align="right">Username:&nbsp;</td>
                    <td>
                        <input type="text" name="j_username"/>
                    </td>
                </tr>
                <tr>
                    <td align="right">Password:&nbsp;</td>
                    <td>
                        <input type="password" name="j_password"/>
                    </td>
                </tr>
                <tr>
                    <td></td>
                    <td><input type="submit" value="Login"/></td>
                </tr>
            </table>
        </form>
            </ui:define>
        </ui:composition>
    </body>
</html>
```
Note that the Facelet uses a HTML Form rather than a JSF Form.

Login Error Page

The login error page is displayed when the user credentials are incorrect.

Create a **JSF Template Client** from ***template.xhtml*** called ***loginErrorPage.xhtml*** in ***BookStore-war*** and edit so that the contents are as follow :

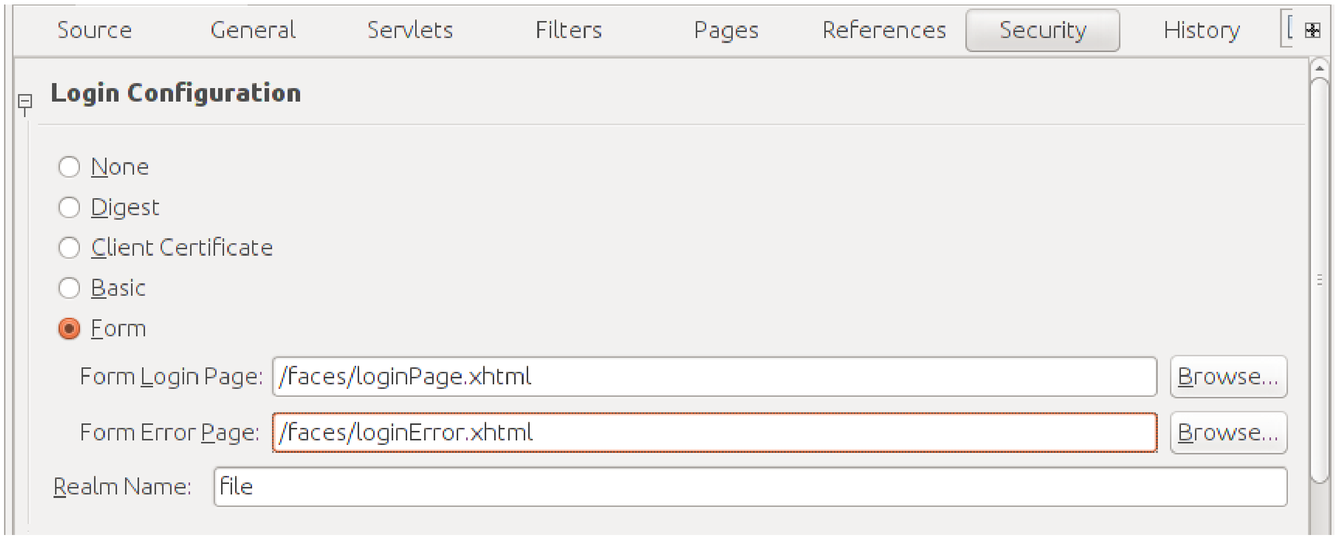
```
<?xml version='1.0' encoding='UTF-8' ?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml"
      xmlns:ui="http://xmlns.jcp.org/jsf/facelets">
    <body>
        <ui:composition template="./template.xhtml">
            <ui:define name="content">
                Wrong username or password. Please try again..
            </ui:define>
        </ui:composition>
    </body>
</html>
```

Application Configuration

The application configuration is done in the application deployment descriptor *web.xml* in folder **Configuration Files.**

- Open *web.xml* form folder **Configuration Files.**
- Click on the *Security* button in the toolbar. In the *Login Configuration* section, choose the type of authentication (***Form***) and indicate the login and login error pages.

  Specify ***file*** as **Realm Name**.



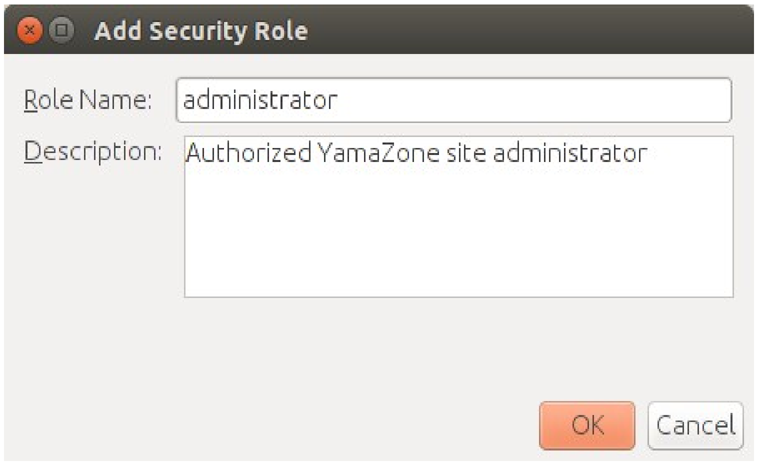
Login Configuration

None [✓ Form]
Digest
Client Certificate
Basic

Form Login Page: /faces/loginPage.xhtml
Form Error Page: /faces/loginError.xhtml
Realm Name: file

Make sure that **Form Login Page** is ***/faces/loginPage.xhtml*** and **Form Error Page** ***/faces/loginError.xhtml***.

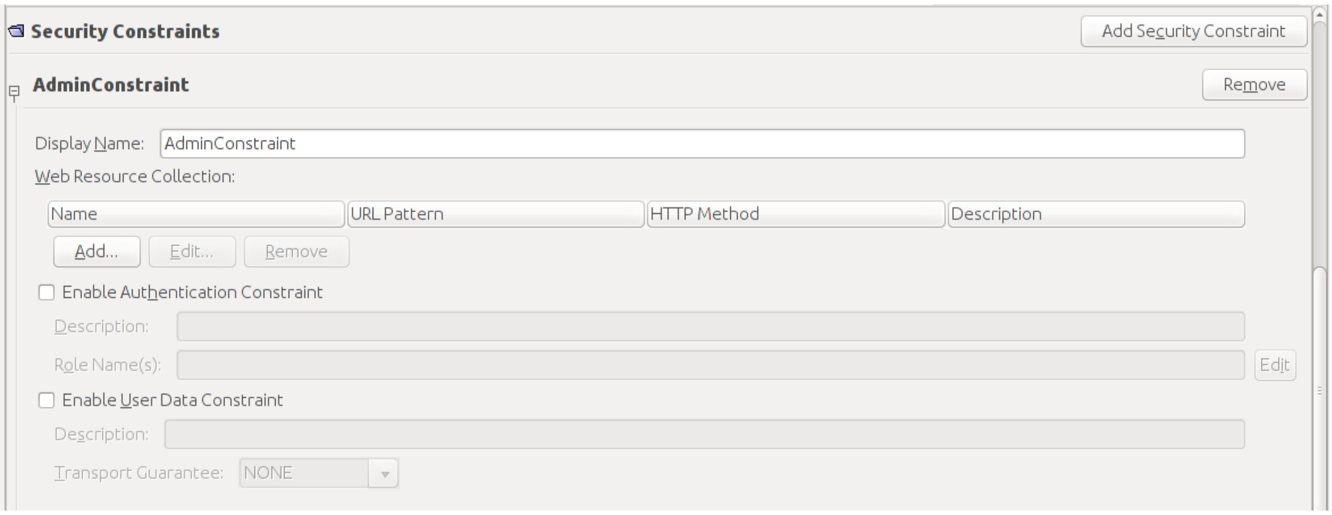
- Click on **Add..** in the *Security Roles* section. Define a role *administrator.*



Role Name: administrator

Description: Authorized YamaZone site administrator

- Click on the **Add Security Constraint** button to specify an access policy. Specify ***Display Name*** as *AdminConstraint.*



Display Name: AdminConstraint

Web Resource Collection:

Enable Authentication Constraint

Description:

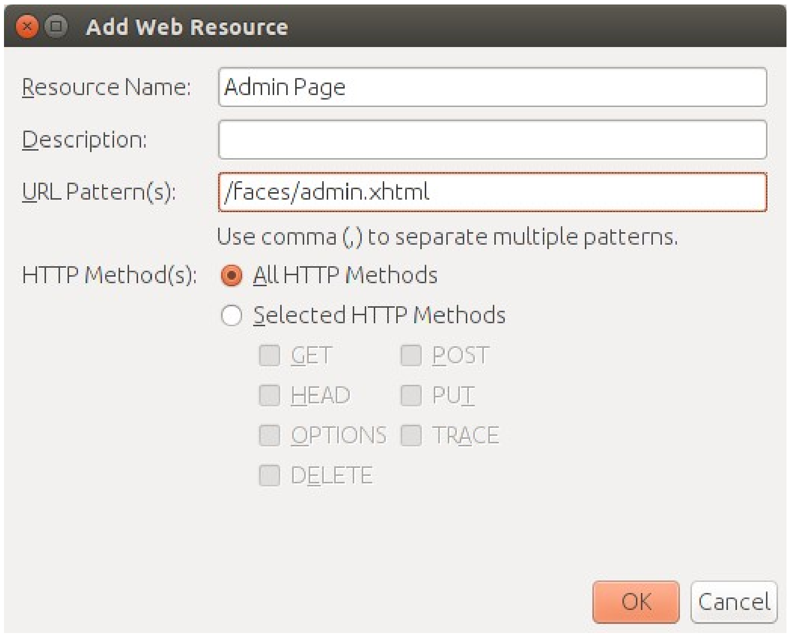
Role Name(s):

Enable User Data Constraint

Description:

Transport Guarantee: NONE

- Click on the **Add...** button under the **Web Resource Collection** section. Provide a Resource Name, an optional Description and URL Pattern(s) for the pages belonging to the security constraint.



Resource Name: Admin Page

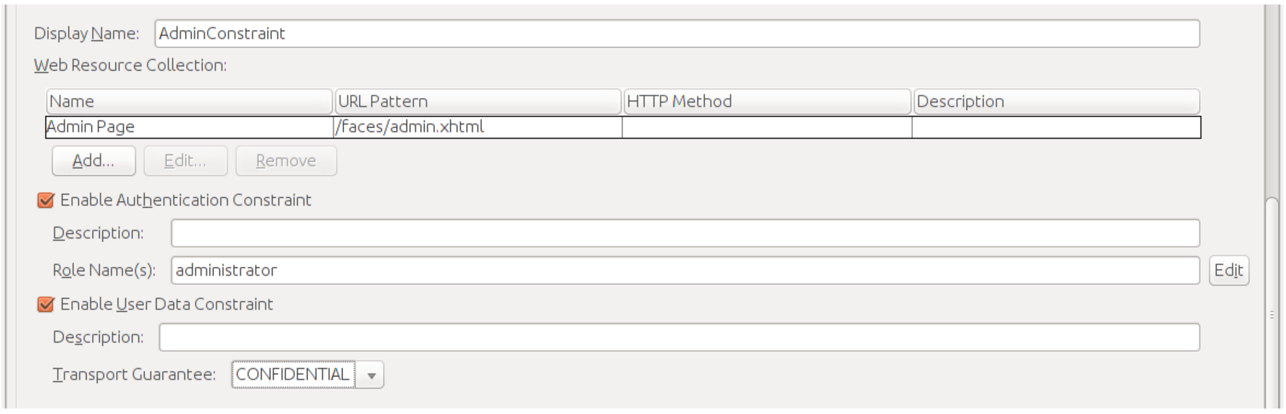
URL Pattern(s): /faces/admin.xhtml

Use comma (,) to separate multiple patterns.

HTTP Method(s): [✓ All HTTP Methods] Selected HTTP Methods

- Check the **Enable Authentication Constraint** checkbox and enter the role(s) that are authorized to view the page in the Role Name(s) field. Click on Edit beside Role Name(s) field and select *administrator*.

  Check the **Enable User Data Constraint** checkbox and select CONFIDENTIAL as **Transport Guarantee**. This will ensure user credential as sent on an encrypted connection.



Display Name: AdminConstraint

Web Resource Collection:

Name	URL Pattern	HTTP Method	Description
Admin Page	/faces/admin.xhtml		

✓ Enable Authentication Constraint

Description:

Role Name(s): administrator

✓ Enable User Data Constraint

Description:

Transport Guarantee: CONFIDENTIAL

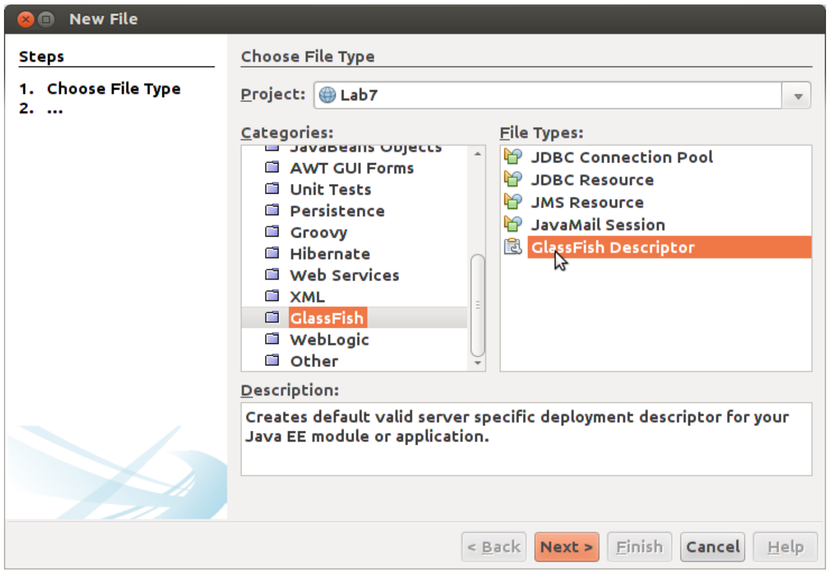
- Make sure to save *web.xml*.

Server Configuration

When deploying to GlassFish, we need to modify *glassfish -web. xml*, a GlassFish-specific deployment descriptor.

- Create *glassfish -web. xml* by Right-clicking on project ***BookStore-war*** and selecting

File -> New -> Other, selecting the **GlassFish** category and the **GlassFish Descriptor** file type.



Choose File Type

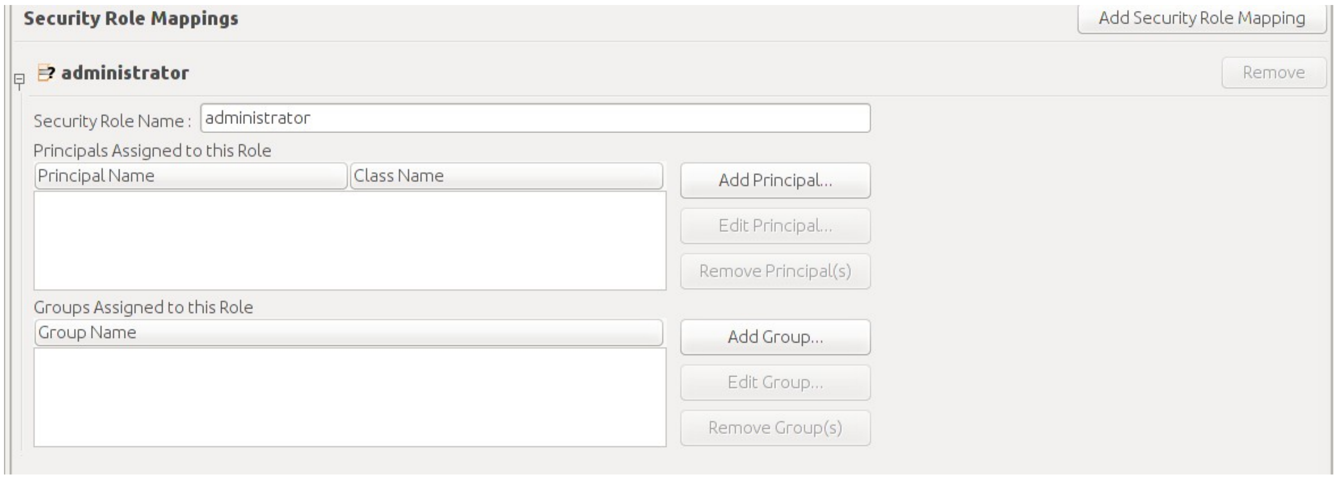
Project: Lab7

Categories: JavaBeans Objects, AWT GUI Forms, Unit Tests, Persistence, Groovy, Hibernate, Web Services, XML, GlassFish, WebLogic, Other

File Types: JDBC Connection Pool, JDBC Resource, JMS Resource, JavaMail Session, GlassFish Descriptor

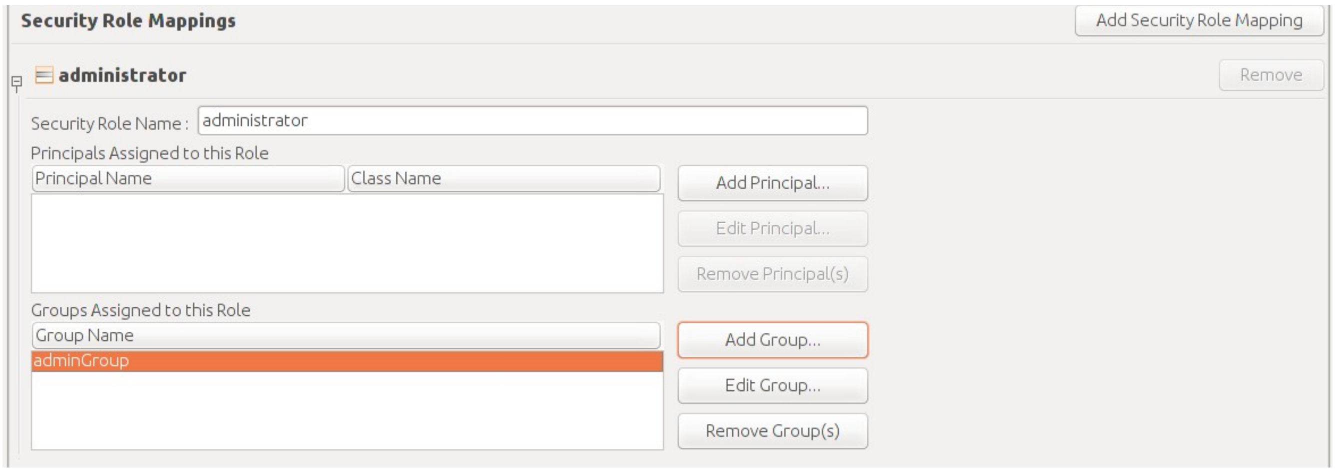
Description:
Creates default valid server specific deployment descriptor for your Java EE module or application.

- Click **Next** > then **Finish**. The IDE should open the *glassfish -web. xml* descriptor in a visual editor. If not click to open from **Configuration Files** folder.
- Click on the **Security** tab and expand the *administrator* role.



Security Role Name : administrator

- We need to enter one or more groups to be assigned to the role by clicking the **Add Group...** button and entering a group name ***adminGroup***.


Security Role Mappings

administrator

Security Role Name : administrator

Principals Assigned to this Role

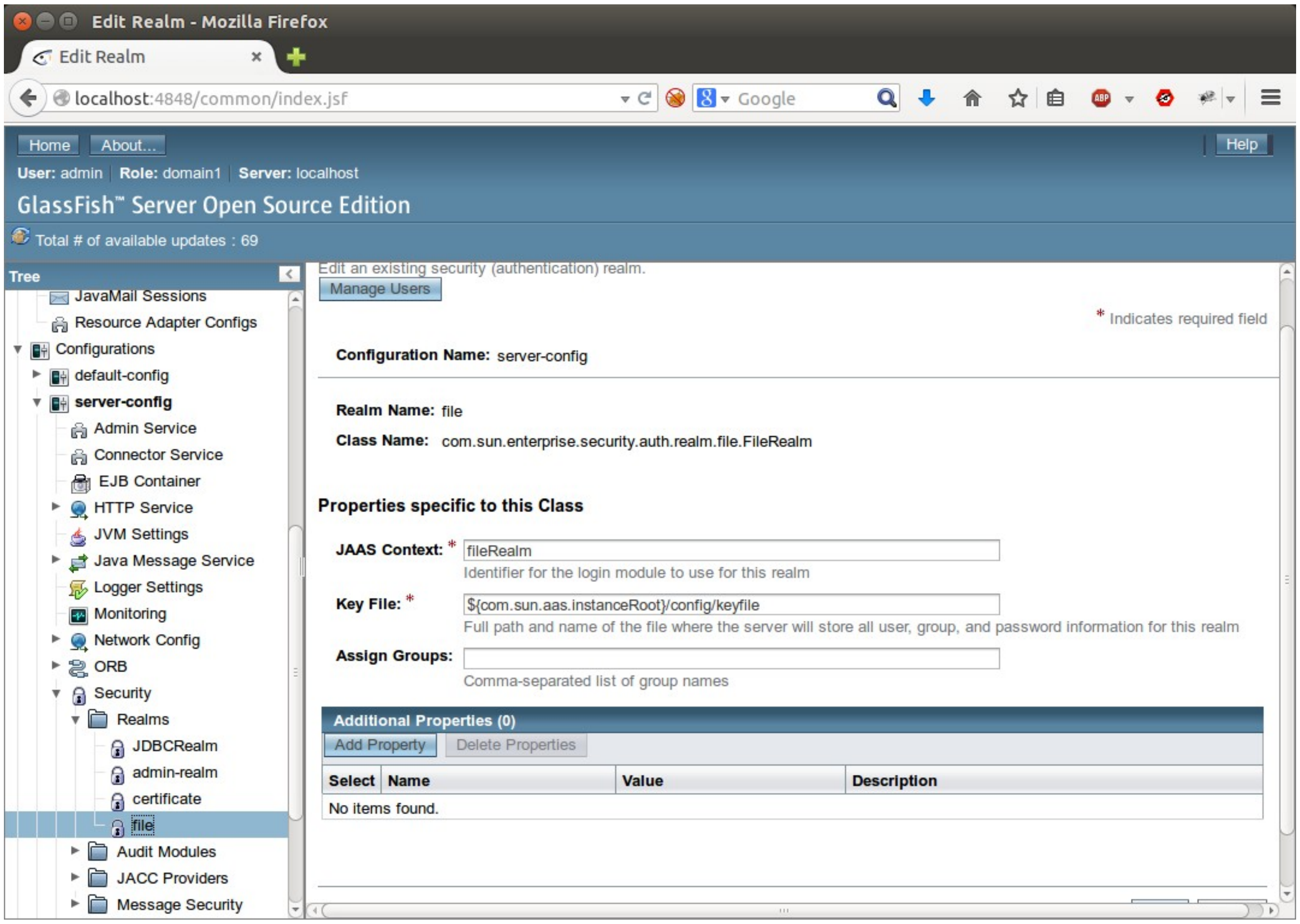
Principal Name	Class Name

Groups Assigned to this Role

Group Name
adminGroup

- Open the GlassFish admin console by going to the **Services** window, expanding the **Servers** node, then right- clicking on the **GlassFish Server**, and selecting **View Domain Admin Console**.

  In the GlassFish console, expand the **Configuration** node, followed by **Security**, followed by **Realms**, then click on the **file** realm.



User: admin Role: domain1 Server: localhost

GlassFish™ Server Open Source Edition

Total # of available updates : 69

Edit an existing security (authentication) realm.

* Indicates required field

Configuration Name: server-config

Realm Name: file

Class Name: com.sun.enterprise.security.auth.realm.file.FileRealm

Properties specific to this Class

JAAS Context: * fileRealm
Identifier for the login module to use for this realm

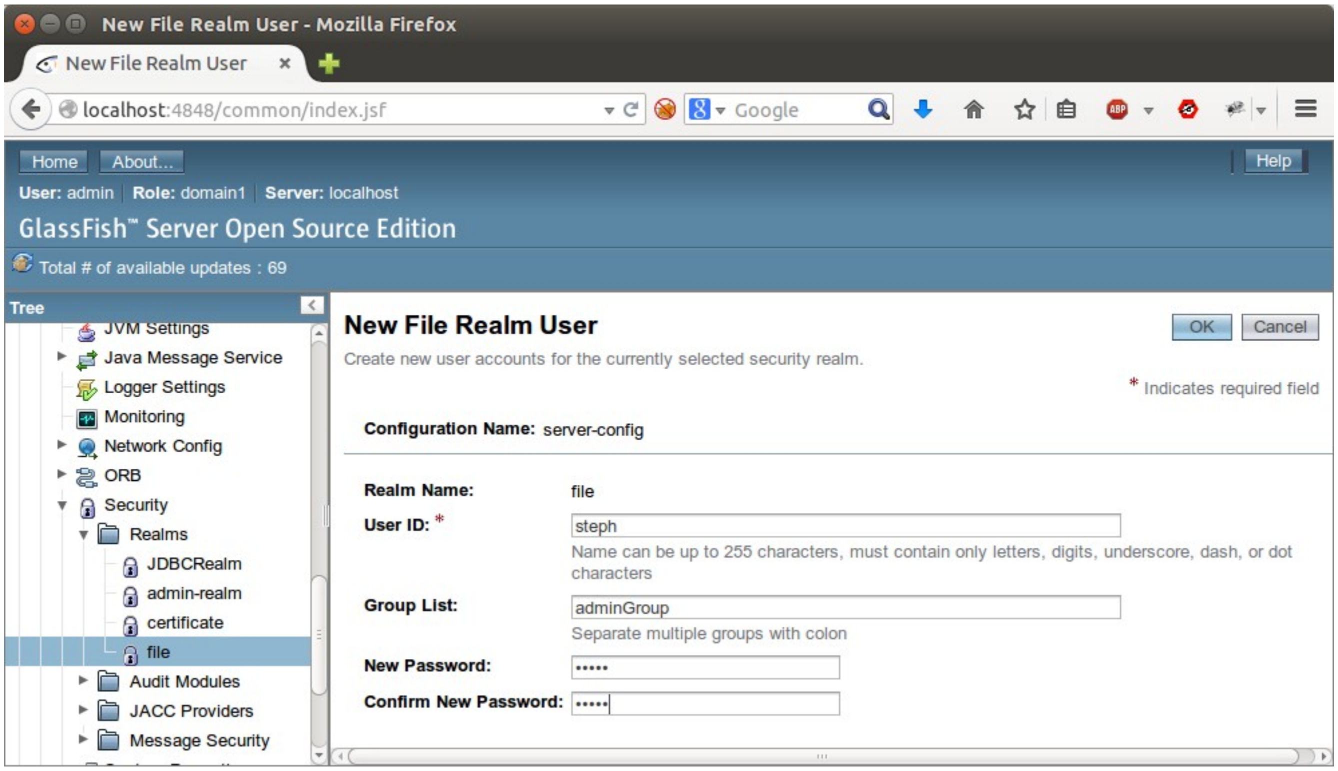
Key File: * ${com.sun.aas.instanceRoot}/config/keyfile
Full path and name of the file where the server will store all user, group, and password information for this realm

Assign Groups:
Comma-separated list of group names

Additional Properties (0)

Select	Name	Value	Description
No items found.			

- Click on the **Manage Users** button and then **New.** Specify a ***User ID, Group List*** (*adminGroup*) and ***Password*.**



New File Realm User

Create new user accounts for the currently selected security realm.

* Indicates required field

Configuration Name: server-config

Realm Name:	file
User ID: *	steph Name can be up to 255 characters, must contain only letters, digits, underscore, dash, or dot characters
Group List:	adminGroup Separate multiple groups with colon
New Password:	•••••
Confirm New Password:	•••••

Click **OK** button.

- Restart the Glassfish server by going to the Netbeans **Services** window, expanding the **Servers** node, then right- clicking on the **GlassFish Server**, and selecting **Restart**.
- Wait for the server to restart and run the project. The login form should display asking for username and password. Specify the User ID and Password entered above.
- **Note: it is normal to get a security warning the first time the application is run. This is due to the use of a secure connection. Just allow the exception and proceed.**

Application Provided Authentication

The archive ***UserAccountDemo.zip*** contains a zip export of a JSF project that demonstrates the implementation of Application Provided Authentication. Import the project, run it and explore the code.