Software

To finish the lab, you may need the following software:

1. NetBeans IDE version 12.2 or version 12.5
2. JDK version 1.8.0 (jdk1.8.0_202, or later)
3. GlassFish Server Open Source Edition version 5.1.0

Aim

Programming a simple Java EE Web Application using the following technologies

1. JSF
2. CDI Beans (better than Managed Beans)
3. Stateless Session Bean (SLSB)
4. Entity Class and its DTO

Overview of the Java EE application

In this Lab, we extend the ED-JEE project done in Lab 04 so that it can now add a new user via a Web Application. We are going to extend the application by

1. Programming a set of JSF pages so as to provide one of the CRUD operations over the web
   a. A JSF page that displays the main menu for the CRUD options, `mainmenu.xhtml`
   b. A JSF page that supports adding a new user to the database, `addUser.xhtml`
   c. A JSF page that displays the successful response after adding a new user to the database, `addUserSuccess.xhtml`
   d. A JSF page that displays a failure message of not being able to add a new user to the database, `addUserFailure.xhtml`
   e. … (others are omitted for your additional work) …

   Note: Both (c) and (d) can be done by using one JSF page that displays either the successful response or the failure response of the add user option, `addUserResponse.xhtml`. In this lab, we are doing this in two JSF pages. This is just an arbitrary choice made by me. If you want to implement this in just one JSF page, try it yourself as an additional exercise.
2. Programing the managed bean, "MyuserManagedBean.java", that stores the information for a user and talks to the MyuserFacade stateless session bean to create the actual Myuser data in the database
3. Defining the page navigation rules in the application configuration file, faces-config.xml
4. Setting up the managed bean information in the application configuration file, faces-config.xml

PreLab Tasks

Make sure you have completed Lab_02, giving you "ED-JDBC", and Lab_04, giving you "ED-JEE-DTO". Please copy the entire "ED-JEE-DTO" **to a different folder.** Otherwise, this lab will destroy your previous work. The application already has a stateless session bean MyuserFacade, an entity class Myuser, a DTO MyuserDTO and a database table MYUSER that stores the permanent Myuser data. Also, if you have changed the MYUSER database table, you can rerun the "ED-JDBC" project to recreate the table and reload some data into the table.

Overview of Lab Tasks

In this Lab, you will learn to perform the following tasks in programming those enterprise application components as described in Overview of the Java EE application in this lab above.

LT1. Open an existing Enterprise Application project using NetBeans (Recap)
LT2. Create a web application and add it to the existing Enterprise Application project
LT3. Program a Managed Bean component for use in the JSF technologies
LT4. Program a set of JSF pages
LT5. Define the page navigation rules in the application configuration file
LT6. Deploy the entire Enterprise Application using NetBeans (Recap)
LT7. Run the Enterprise Web Application using NetBeans

Lab Tasks

This Lab should be run on MS Windows Platform

LT1. Open an existing NetBeans project called "ED-JEE-DTO"

LT1.1 Select “File” > “Open Project”
LT1.2 Browse to the folder where you stored “ED-JEE-DTO”
LT1.3 Choose “ED-JEE-DTO”
LT1.4 Make sure the following checked boxes are checked
a. “Open as Main Project”
b. “Open Required Projects”
LT1.5 Click “Open Project”
*Note: There are four newly open "projects" in the "Projects" Windows. They are
a. ED-JEE-DTO (Enterprise Application Project)
b. ED-JEE-DTO-ejb (EJB Module Project)
c. ED-JEE-DTO-RI (EJB Remote Interface Project)
d. ED-JEE-DTO-war (Web Application Project)

LT2. In this Lab task, you will learn how to create a web application that uses the JSF framework.
LT2.1 Select “ED-JEE-DTO-war” in the Project tab
Note: The “index.xhtml” page is opened in the editor window for you. We will not need it. But, there is no harm to leave it there.

LT3. In this Lab task, we need program a Managed Bean component, MyuserManagedBean.java that stores the information of a user (entered by someone through a web browser) and talks to the MyuserFacade SLSB on the EJB tier to create the actual user data in the database. Remember, all such communications are through the MyuserDTO. Please note that this Managed Bean is sitting on the web server.

First, we need to store the following properties in the Managed Beans
String userid
String name
String password
String email
String phone
String address
String secQn
String secAns

Second, we decided to call the “createRecord()” method in the MyuserFacade SLSB, which returns a boolean type to indicate whether the “create” operation is successful or not.

Third, the method “addUser()” in the MyuserManagedBean is responsible for making the request to “create a Myuser record” in the MyuserFacade SLSB. This method returns the String “success” to indicate that the “create record” operation in the EJB container is a success, or the String “failure” if the operation in EJB container fails.

We are now doing all these. Before that, we need to add the "ED-JEE-DTO-RI" project to "ED-JEE-DTO-war" project
LT3.1 Add the "ED-JEE-DTO-RI" project to ED-JEE-DTO-war project (Recap)
a. Expand the "ED-JEE-DTO-war" project and select Libraries
b. Right click the mouse and select "Add Project..."
c. In the Add Project dialog box, select “ED-JEE-DTO-RI” project
d. Click "Add Project JAR Files"
LT3.2 Select ED-JEE-DTO-war project and right click the mouse and select “New” > “JSF CDI Bean…” [In case, you cannot find the option, select “New” > “Other…” > “JavaServer Faces” > “JSF CDI Bean” and click “Next”]
LT3.3 In the “New JSF CDI Bean” window
a. Enter “MyuserManagedBean” in the “Class Name” field
b. Enter “web” in the Package field

c. Make sure “myuserManagedBean” in the “Name” field starts with a lower case “m”
d. Select “request” in the “Scope” combo box
e. Click “Finish”

Note: NetBeans creates the file “MyuserManagedBean.java”.

LT3.4 In this Lab task, we are going to put some code in the “MyuserManagedBean.java” class

a. Put your mouse within the “MyuserManagedBean” class
b. Right click your mouse, select “Insert Code…” > “Call Enterprise Bean…”
c. In the “Call Enterprise Bean” dialog window, expand on “ED-JEE-DTO-ejb” node and select “MyuserFacade”
d. Make sure “Remote” is selected in the “Referenced Interface”
e. Click “OK”

Note: NetBeans generates the following code segment for you

```
@EJB private MyuserFacadeRemote myuserFacade;
```

LT3.5 Add the "Java EE 8 API Library" to the "Libraries"
(Hope you remember how to do it)

LT3.6 Copy and paste the following code segment in MyuserManagedBean.java

```
private String userid;
private String name;
private String password;
private String cPassword;  // for confirmed password field
private String email;
private String phone;
private String address;
private String secQn;
private String secAns;

public MyuserManagedBean() {
}
```

LT3.7 Generate the getters and setters for all instance variables
(Hope you remember what to do)

LT3.8 Copy and paste the following code segment in MyuserManagedBean.java

```
/*
 * add a user to the database
 * @return "success" if the add operation is successful
 *         "failure" otherwise
 */
public String addUser() {

    String result = "failure";

    /*
     * are all data entered valid?
     * and password the same as cPassword (case sensitive)
     * before calling the façade's createRecord() method
     */
    if (isValidUserid(userid) && isValidName(name)
        && isValidPassword(password) && isValidPassword(cPassword)
        && isValidEmail(email) && isValidPhone(phone)
        && isValidAddress(address) && isValidSecQn(secQn)
        && isValidSecAns(secAns) && password.equals(cPassword)) {

        MyuserDTO myuserDTO = new MyuserDTO(userid, name,
            password, email, phone, address, secQn, secAns);

        if (myuserFacade.createRecord(myuserDTO)) {
            result = "success";
        }
    }

    return result;
}

/* Some basic checking, complicated checking can be done later
```

```
 * not a good way of doing this
 * Should use JSF's validator method to do this - left as C task
 */
public boolean isValidUserid(String userid) {
    return (userid != null);
}

public boolean isValidName(String name) {
    return (name != null);
}

public boolean isValidPassword(String password) {
    return (password != null);
}

public boolean isValidEmail(String email) {
    return (email != null);
}

public boolean isValidPhone(String phone) {
    return (phone != null);
}

public boolean isValidAddress (String address) {
        return (address != null);
}

public boolean isValidSecQn(String secQn) {
    return (secQn != null);
}

public boolean isValidSecAns(String secAns) {
        return (secAns != null);
}
```

Remember to fix the import and format of the code, then **save** the file

LT4. Add the beans.xml config file in ED-JEE-DTO-war project to activate CDI Bean Manager on Java EE server

a. Select the ED-JEE-DTO-war project
b. Right click your mouse on ED-EMS-SLSB-war
c. Select "New" > "beans.xml (CDI Configuration File)…" [In case, you cannot find the option, select "New" > "Other…" > "Contexts and Dependency Injection" > "beans.xml (CDI Configuration File)"" and click "Next"]
d. In the "New beans.xml (CDI Configuration File)" window, do the following
   i. Enter beans in the "File Name" text field if NetBeans has not pre-filled the information for you
   ii. Enter WEB-INF in the "Folder" text field if NetBeans has not pre-filled the information for you
   iii. Click the **Finish** button

NetBeans will create the beans.xml page and open it in an editor tag. Please see below for an example. If the content is different, I suggest you use the following. Remember to save the file.

beans.xml [code segment]

```
<?xml version="1.0" encoding="UTF-8"?>
<beans 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/beans_1_1.xsd"
       bean-discovery-mode="annotated">
</beans>
```

If you cannot see the file, you probably need to manually create it via your own text editor or the NetBeans editor (Create a new plain text file using the "Files" tab, navigate to the right location WEB-INF and save the file as beans.xml).

LT5. In this Lab task, you will program several JSF pages. They are

(1) `mainmenu.xhtml` that displays the CRUD options

(2) `addUser.xhtml` that presents a form allowing user to add details of a user

(3) `addUserSuccess.xhtml` that displays a successful message for adding a user

(4) `addUserFailure.xhtml` that displays a failure message for adding a user

(5) `undercon.html` that displays unknown request / unimplemented selection

LT5.1 Now, create a "JSF page" called mainmenu.xhtml in ED-JEE-DTO-war project

a. Select the ED-JEE-DTO-war project

b. Right click the mouse and select "New" > "JSF Page…"
[In case, you could not find the option, select "New" > "Other…" > "Web" > "JSF Page", then click "Next"]

c. Enter "mainmenu" in the "File Name" field (NetBeans will add the file extension)

d. Select "Facelets" in the "Options" field

e. Click "Finish"

f. Copy and paste the following code segment to mainmenu.xhtml

```
<h:head>
  <title>Main Menu</title>
</h:head>
<h:body>
  <h:form>
    <h1>
      <h:outputText value="Welcome to Myuser Web Application!"/>
    </h1>

    <h2>
      <h:outputText value="Please select one of the following options"/>
    </h2>

    <h3>
      <ol>
        <li><a href="addUser.xhtml">Add a new user</a></li>
        <li><a href="undercon.html">Display a user</a></li>
        <li><a href="undercon.html">Edit a user</a></li>
        <li><a href="undercon.html">Delete a user</a></li>
      </ol>
    </h3>
  </h:form>
</h:body>
```

Remember to reformat and save the file

LT5.2 Create "addUser.xhtml" in ED-JEE-DTO-war project

a. Now, repeat the steps in LT5.1(a) – (e) above to create "addUser.xhtml"
[Remember to use the "Facelets" option]

b. Copy and paste the following code segment in addUser.xhtml

```
<h:head>
  <title>Add a User Page</title>
</h:head>
<h:body>
  <h1>
    Add a User
  </h1>

  <h2>
    <p>Please enter the user's details below</p>
  </h2>

  <h3>
    <h:form>
      <h:panelGrid columns="2">
```



```
        <h:outputText value="User Id: "/>
        <h:inputText id="userid" value="#{myuserManagedBean.userid}"
            required="true"
            requiredMessage="The userid field cannot be empty!"
            size="6" />

        <h:outputText value="Name: "/>
        <h:inputText id="name" value="#{myuserManagedBean.name}"
            required="true"
            requiredMessage="The name field cannot be empty!"
            size="30"/>

        <h:outputText value="Password "/>
        <h:inputText id="password" value="#{myuserManagedBean.password}"
            required="true"
            requiredMessage="The password field cannot be empty!"
            size="6"/>

        <h:outputText value="Confirm Password "/>
        <h:inputText id="cPassword" value="#{myuserManagedBean.cPassword}"
            required="true"
            requiredMessage="The confirm password field cannot be empty!"
            size="6"/>

        <h:outputText value="Email: "/>
        <h:inputText id="email" value="#{myuserManagedBean.email}"
            required="true"
            requiredMessage="The email field cannot be empty!"
            size="30" />

        <h:outputText value="Telephone: "/>
        <h:inputText id="phone" value="#{myuserManagedBean.phone}"
            required="true"
            requiredMessage="The telephone field cannot be empty!"
            size="10" />

        <h:outputText value="Address: "/>
        <h:inputText id="address" value="#{myuserManagedBean.address}"
            required="true"
            requiredMessage="The email field cannot be empty!"
            size="30" />

        <h:outputText value="Security Question: "/>
        <h:inputText id="secQn" value="#{myuserManagedBean.secQn}"
            required="true"
            requiredMessage="The security question field cannot be empty!"
            size="60"/>

        <h:outputText value="Security Answer: "/>
        <h:inputText id="secAns" value="#{myuserManagedBean.secAns}"
            required="true"
            requiredMessage="The security answer field cannot be empty!"
            size="60"/>
      </h:panelGrid>
      <p></p>

      <h:commandButton id="submit" value="Submit"
          action="#{myuserManagedBean.addUser}" />
    </h:form>
  </h3>
</h:body>
```

Remember to reformat and save the file.

LT5.3 Create the "addUserSuccess.xhtml" page in ED-JEE-DTO-war project

a. Now, repeat the steps in LT5.1(a)-(e) to create the "addUserSuccess.xhtml" [Remember to use the "Facelets" option]

b. Copy and paste the following code segment to addUserSuccess.xhtml

```
<h:head>
```

```
  <title>User Added</title>
</h:head>

<h:body>
  <h:form>
    <h1>
    User Added - Success
    </h1>

    <h2>
        User whose userid is
        <h:outputText value="#{myuserManagedBean.userid}"/>
        has been added to the system.
    </h2>
    <p></p>

    <h3>
      Back to <h:commandButton value="Main Menu" action="mainmenu.xhtml"/>!
    </h3>
  </h:form>
</h:body>
```

Remember to save the file

LT5.4 Create "addUserFailure.xhtml" in ED-JEE-DTO-war project

a. Now, repeat steps in LT5.1(a) – (e), to create the addUserFailure.xhtml [Remember to use the "Facelets" option]

b. Copy and paste the following code segment to addUserFailure.xhtml

```
<h:head>
    <title>User Not Added</title>
</h:head>
<h:body>
    <h:form>
    <h1>
      User Added - Failure
    </h1>

    <h2>
    User whose userid is
    <h:outputText value="#{myuserManagedBean.userid}"/>
    cannot be added to the system.
   </h2>
    <p></p>

    <h3>
    Possibly there is an existing user with the same userid.
    </h3>
    <p></p>

    Back to <h:commandButton value="Main Menu" action="mainmenu.xhtml"/>
  </h:form>
</h:body>
```

Remember to reformat and save the file

LT5.5 Create the "undercon.html" dummy page

a. Now, create the "undercon.html" page to handle requests that have not been implemented in ED-JEE-DTO-war [As a reminder, Select "New" > "HTML…"]

b. Copy and paste the following code segment to "undercon.html"

```
<head>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8"/>
    <title>Under Construction Page</title>
</head>
<body>
    <h1>
        This page is still under construction!
    </h1>

    Back to <a href="mainmenu.xhtml">Main Menu</a>menu.
</body>
```

There is nothing interesting here. All we need is a pointer so that we can go back to the main menu. Remember to save the file.

LT6. Now, we can define the page navigation rules in the application configuration file "faces-config.xml". We need to create the file first and then add the rules in the file.

Note: The rules are as follows:

(1) from addUser.xhtml – success → addUserSuccess.xhtml

(2) from addUser.xhtml – failure → addUserFailure.xhtml

LT6.1 Select the "ED-JEE-DTO-war" project

LT6.2 Right click the mouse and select "New" > "JSF Faces Configuration…" [In case, you cannot find the option, select "New" > "Other…" > "JavaServer Faces" > "JSF Faces Configuration" and click "Next"]

LT6.3 Make sure "faces-config" is in "File Name" field

LT6.4 Click "Finish"

Note: NetBeans will create and open the file "faces-config.xml" automatically. At the moment, there are no navigations rules.

Note: The file is opened in XML format.

Now, we are going to add rule (1) in the file. This was split into two actions: "Add Navigation Rule" and then "Add Navigation Case".

LT6.5 We are going to add navigation rule.

a. In the editor window of "faces-config.xml", right click the mouse

b. Select "Insert" and then "Navigation Rule…"

c. In the "Add Navigation Rule" window

1. Select "Browse…"
2. Expand on "Web Pages" node in the "Browse Files" window
3. Select "addUser.xhtml"
4. Click "Select File"

Note: You are now back to the "Add Navigation Rule" window

5. Click "Add"

Note: A navigation rule has now been added to the file. It looks like the following

```
<navigation-rule>
    <from-view-id>/addUser.xhtml</from-view-id>
</navigation-rule>
```

LT6.6 We are now going to add a navigation case under a navigation rule.

a. In the editor window of "faces-config.xml", right click the mouse again

b. Select "Insert" > "Add Navigation Case…"

c. In the "Add Navigation Case" window, the "/addUser.xhtml" has been selected for you in the "From View" field. We now need to define the "From Outcome" and "To View" fields

1. Enter "#{myuserManagedBean.addUser}" in the "From Action" field
2. Enter "success" in the "From Outcome" field
3. Select "Browse…" button next to the "To View" field
4. Expand on "Web Pages" node in the "Browse Files" window
5. Select "addUserSuccess.xhtml"
6. Click "Select File"

Note: You are then back to the "Add Navigation Case" window

7. Click "Add"

Note: A navigation case has now been added to the file. The entire navigation rule looks like the following

```
<navigation-rule>
  <from-view-id>/addUser.xhtml</from-view-id>
  <navigation-case>
    <from-action>#{myuserManagedBean.addUser}</from-action>
    <from-outcome>success</from-outcome>
    <to-view-id>/addUserSuccess.xhtml</to-view-id>
  </navigation-case>
</navigation-rule>
```


LT6.7 Repeat steps in LT6.6 to add the remaining rule (2) to the application configuration file, faces-config.xml

Note: Once you finished adding the rules, the navigation rules in the faces-config.xml file should look like the following

```
<navigation-rule>
    <from-view-id>/addUser.xhtml</from-view-id>
    <navigation-case>
        <from-action>#{myuserManagedBean.addUser}</from-action>
        <from-outcome>success</from-outcome>
        <to-view-id>/addUserSuccess.xhtml</to-view-id>
    </navigation-case>
    <navigation-case>
        <from-action>#{myuserManagedBean.addUser}</from-action>
        <from-outcome>failure</from-outcome>
        <to-view-id>/addUserFailure.xhtml</to-view-id>
    </navigation-case>
</navigation-rule>
```

Remember to save the file. Alternatively, you can copy and paste these to the faces-config.xml file.

LT7. Deploy the enterprise application "ED-JEE-DTO" project (Recap)

LT7.1 Select "ED-JEE-DTO" enterprise application project in the "Projects" window

LT7.2 Right click the mouse and select "Deploy" (You may want to select "Clean and Build" first to clean up any mess that has been done.)

LT8. Setup to run the enterprise web application in "ED-JEE-DTO" project

LT8.1 Select "ED-JEE-DTO" enterprise application project in the "Projects" window

LT8.2 Right click the mouse and select "Properties" > "Run"

LT8.3 Select "ED-JEE-DTO-war" in the "Client Module" field

LT8.4 Enter "faces/mainmenu.xhtml" in the "Relative URL" field

LT8.5 Click "OK"

LT8.6 Right click the mouse and select "Run"

Note: Alternatively, you can open your browser and type the url http://localhost:8080/ED-JEE-DTO-war/faces/mainmenu.xhtml in your browser

Finally, take several deep breaths and relax.

Questions:

What if you run the "ED-JEE-DTO-war" project instead?

What if you open a browser and type the url as http://localhost:8080/ED-JEE-DTO/faces/mainmenu.xhtml?

What if the url http://localhost:8080/ED-JEE-DTO-war/mainmenu.xhtml?

What if the url is http://localhost:8080/ED-JEE-DTO-war/faces/mainmenu.xhtml?

Further Tasks

Extend the capability of the enterprise application to support other CRUD operations