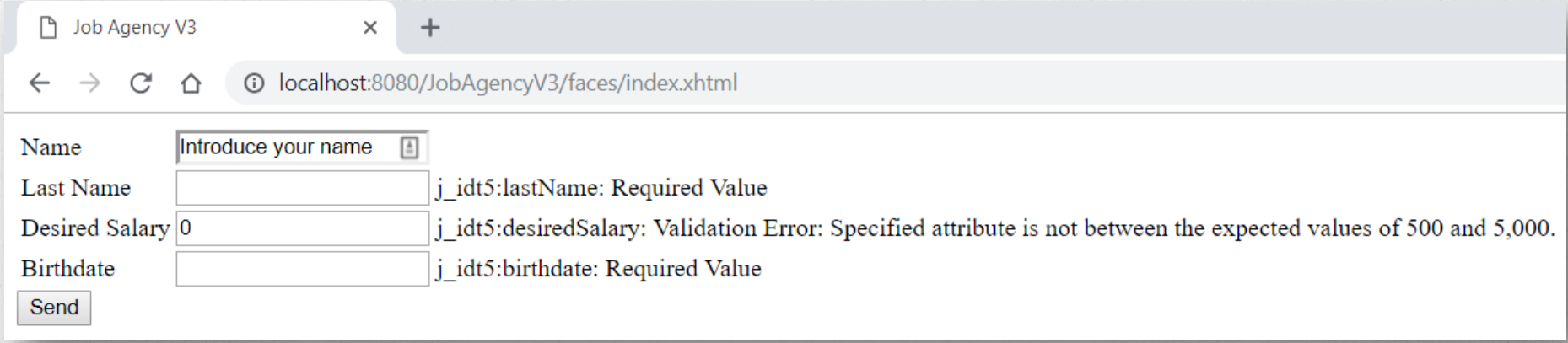
EXERCISE

HANDLING OF MESSAGES WITH JSF

EXERCISE OBJECTIVE

- The objective of the exercise is to modify our project to add the handling of messages in JSF. The result should be similar to the following:



EXERCISE OBJECTIVE

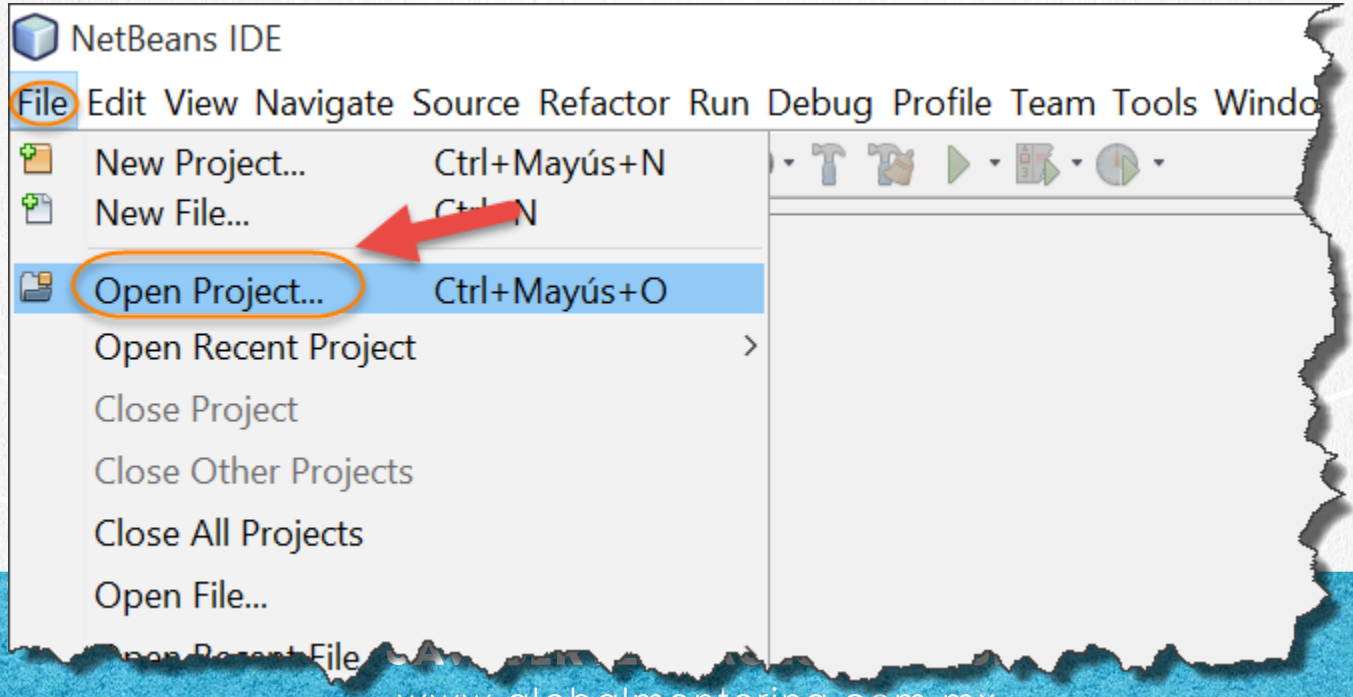
The objective of this exercise is to put into practice the concept of Message Management in JSF.

We will achieve this by changing each of the tags of type h: outputLabel by tags that will come from a properties file.

The result of the exercise visually does not have any change, it only changes the origin of the labels (labels) that are shown.

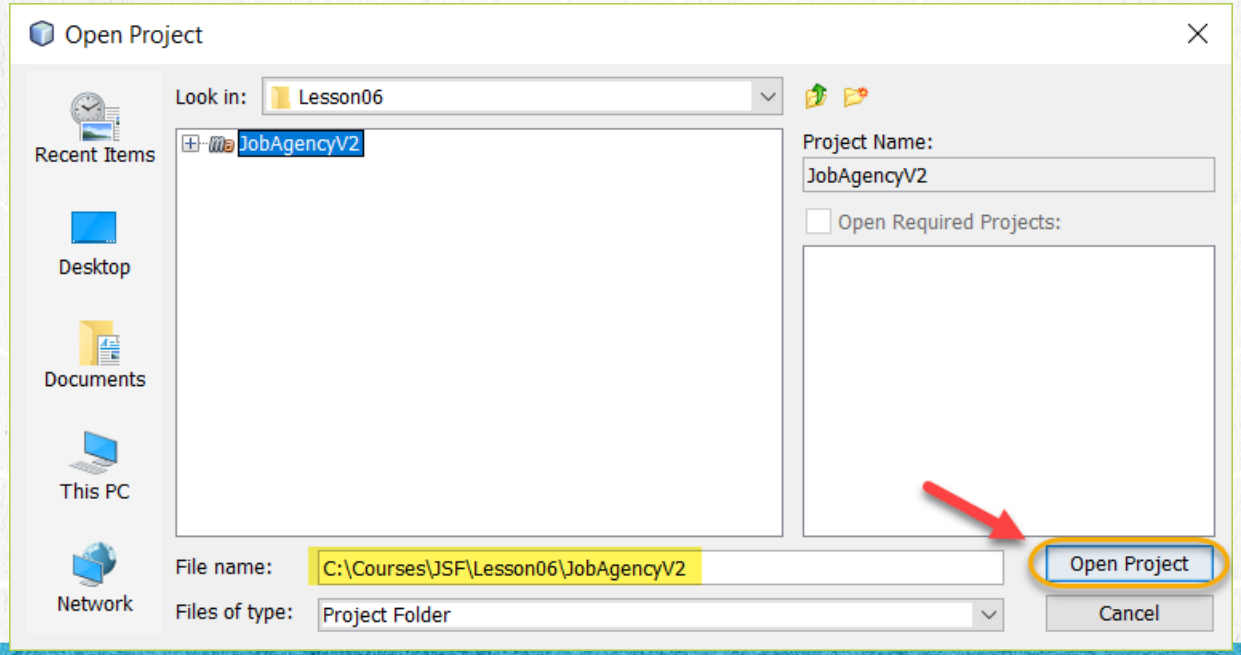
1. OPEN THE PROJECT

We open the JobAgencyV2 project, just in case we do not have it open:



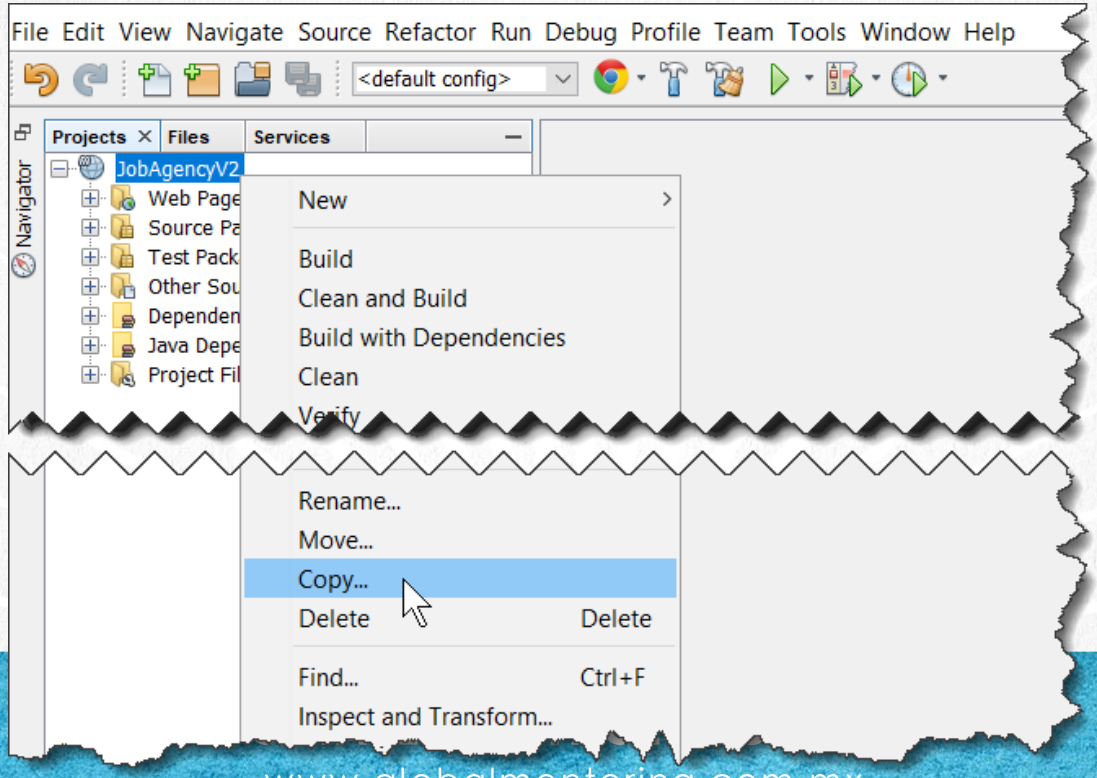
1. OPEN THE PROJECT

We open the JobAgencyV2 project, just in case we do not have it open:



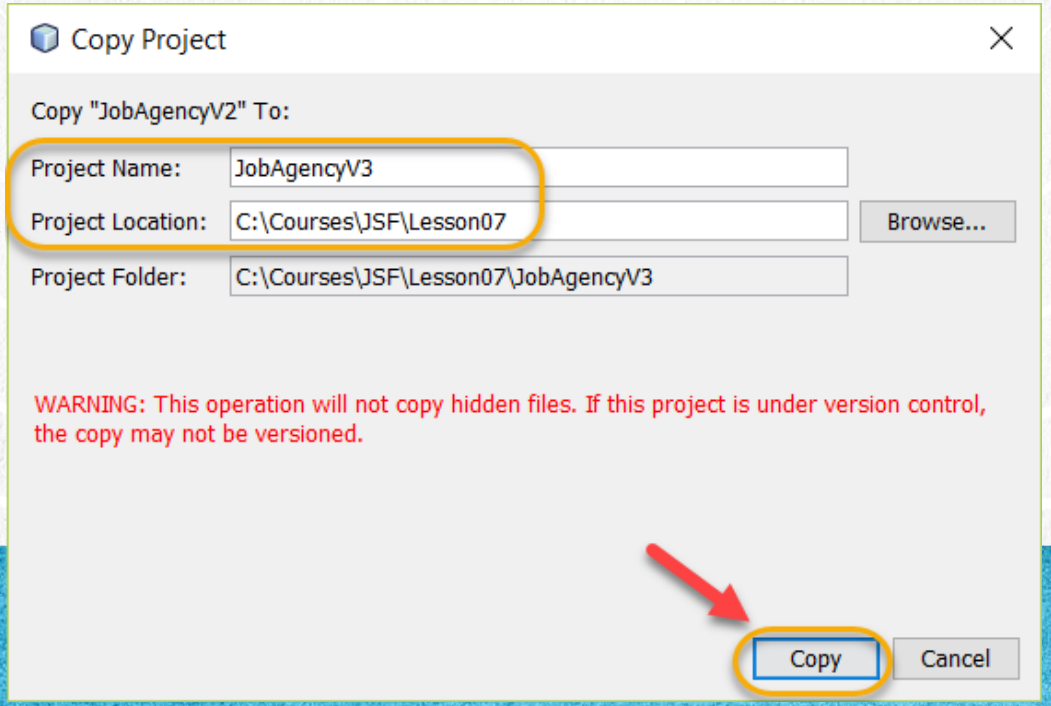
2. COPY THE PROJECT

We copy the JobAgencyV2 project:



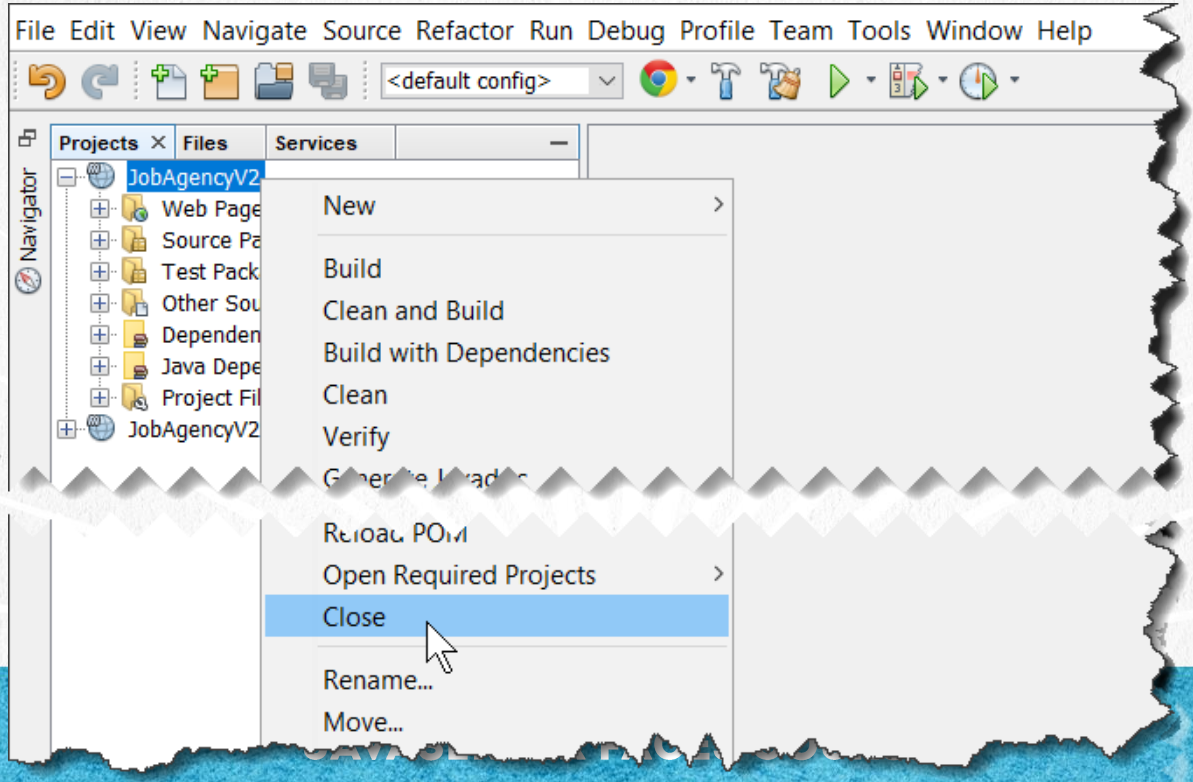
2. COPY THE PROJECT

We copy the JobAgencyV2 project, rename it to JobAgencyV3 and deposit it in the new path shown:



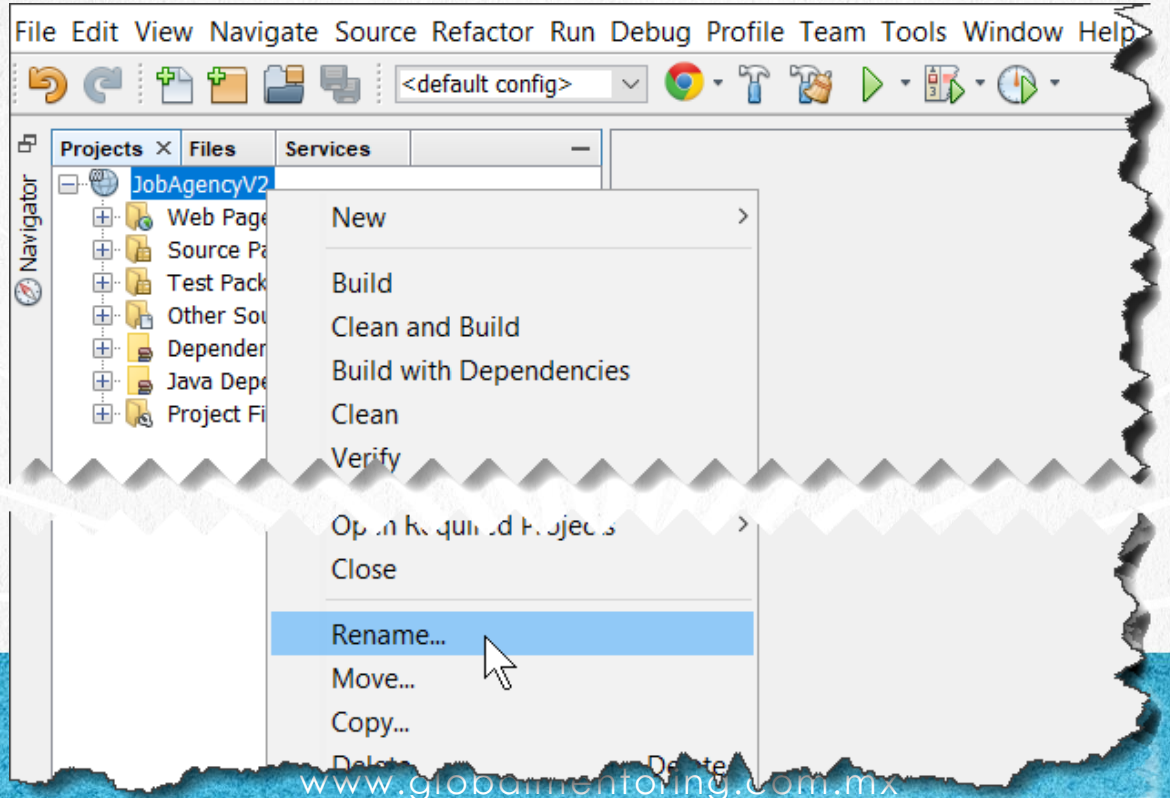
3. CLOSE THE PREVIOUS PROJECT

We closed the previous project, and left only the new open :



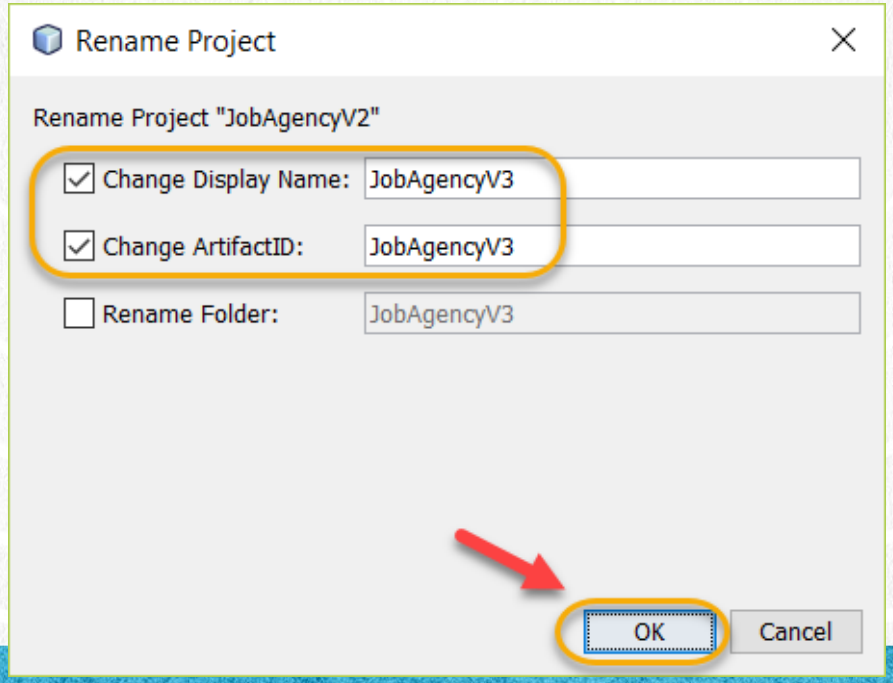
4. RENAME THE PROJECT

We changed the name of the project to JobAgencyV3:



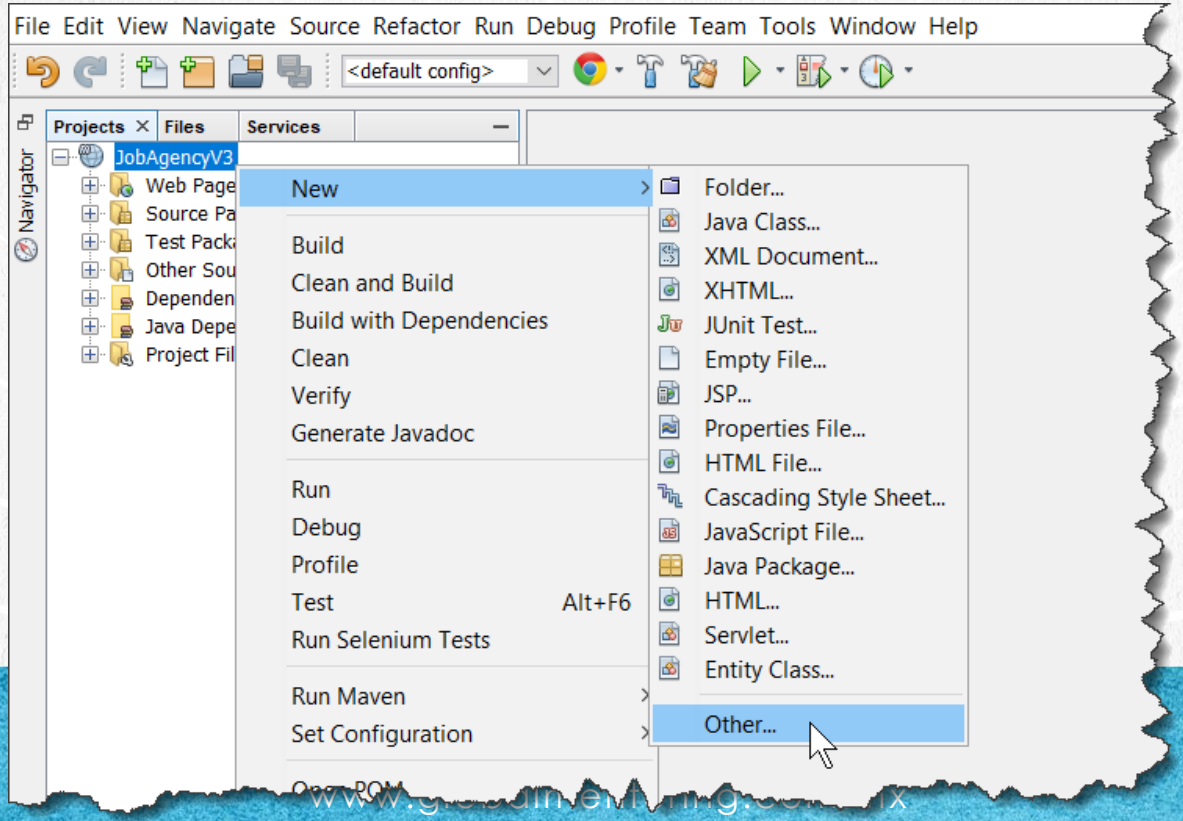
4. RENAME THE PROJECT

We changed the name of the project to JobAgencyV3:



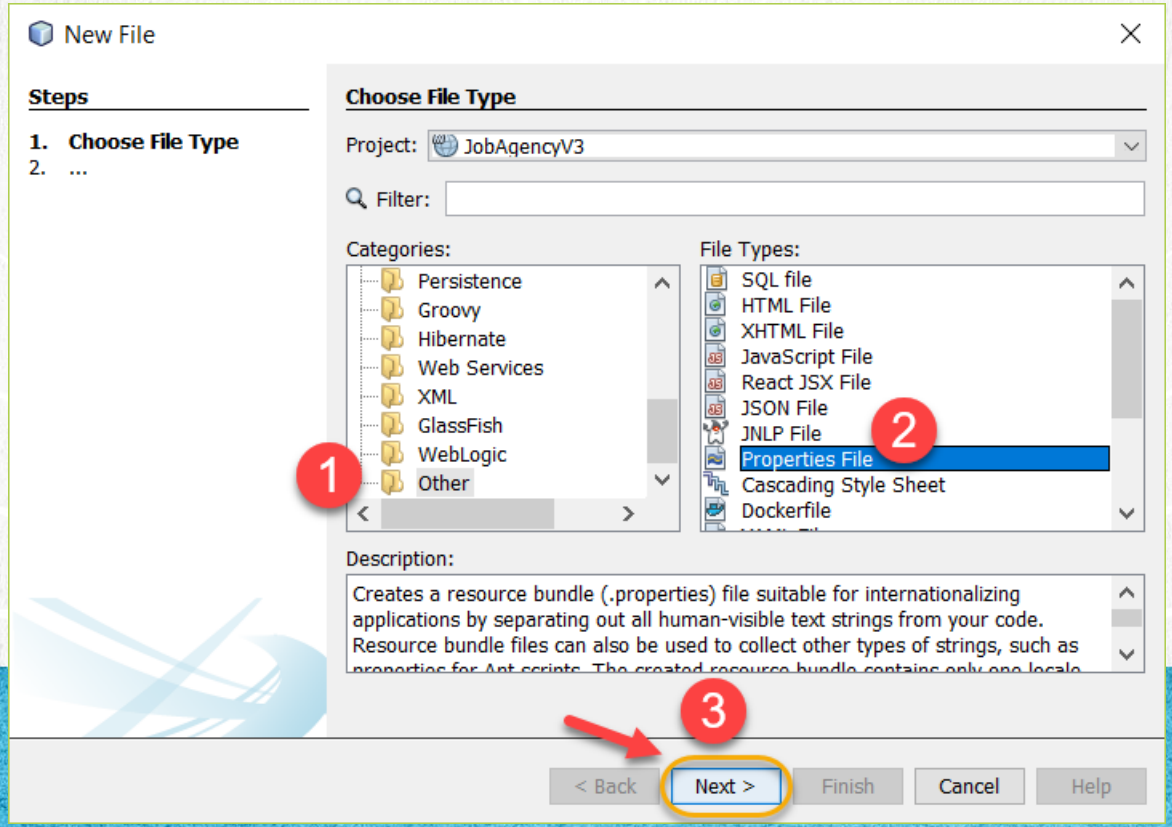
5. CREATE A PROPERTIES FILE

We create a properties file called messages.properties:



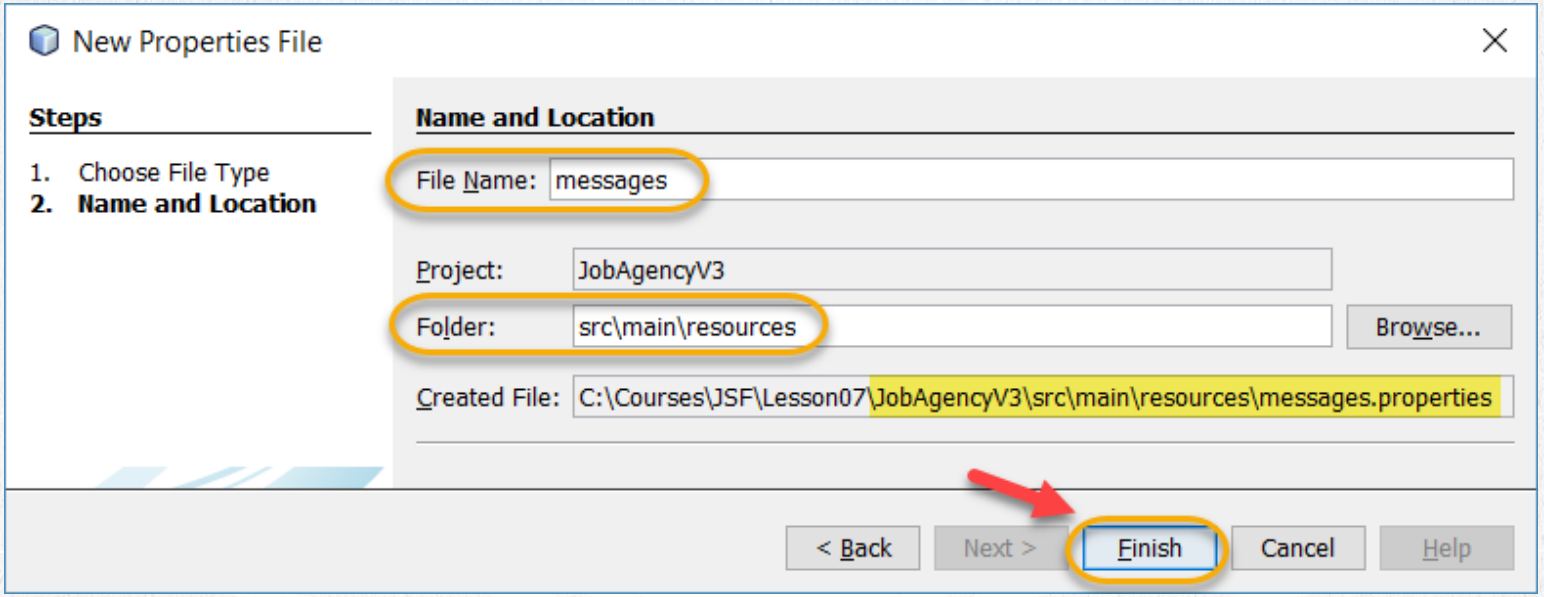
5. CREATE A PROPERTIES FILE

We create a properties file called messages.properties:



5. CREATE A PROPERTIES FILE

We create a properties file called messages.properties:



6. MODIFY THE CODE

messages.properties:

Click to download

```
vacantForm.name=Name
vacantForm.lastName=Last Name
vacantForm.desiredSalary=Desired Salary
vacantForm.birthdate=Birthdate
vacantForm.send=Send
```

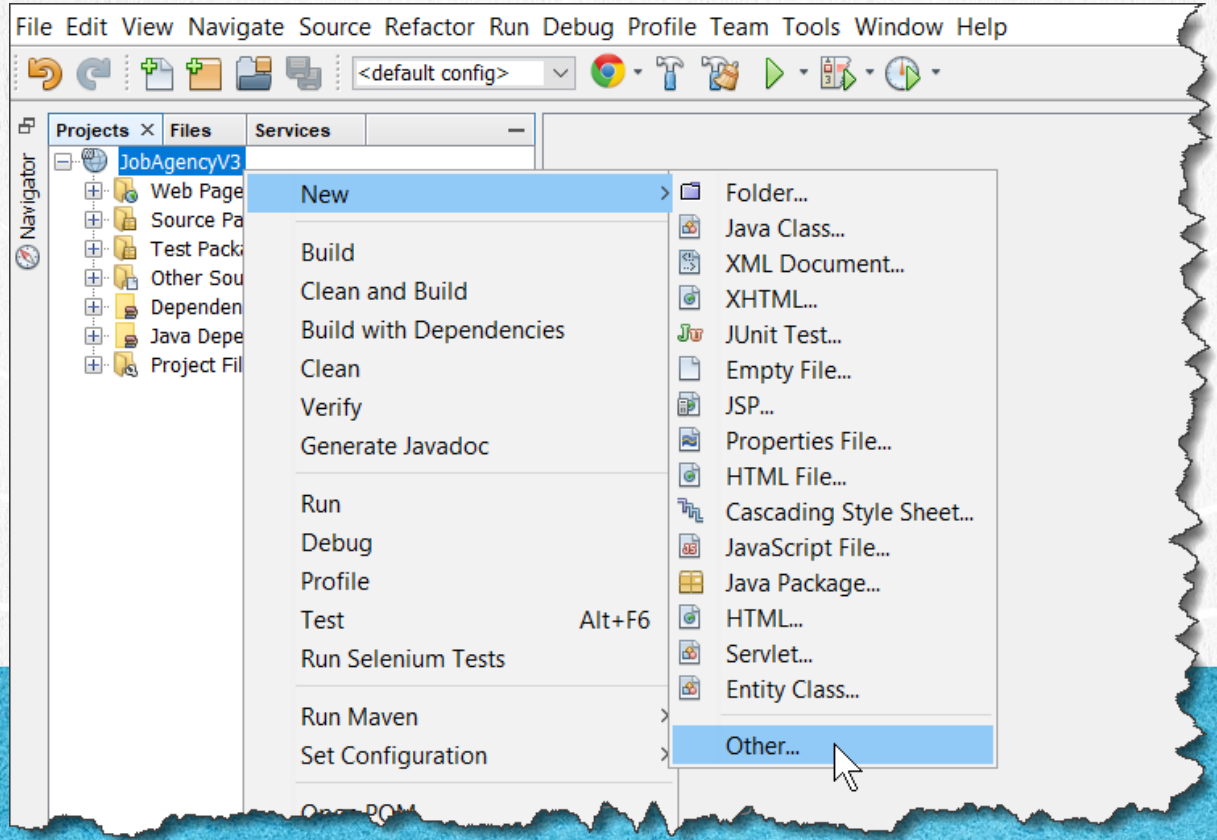
7. CREATE A PROPERTIES FILE

Sometimes it is convenient to change the error messages that JSF has by default.

In order to change them, we will follow a similar procedure to the previous one, creating a jsf.properties file, however the name can be any other.

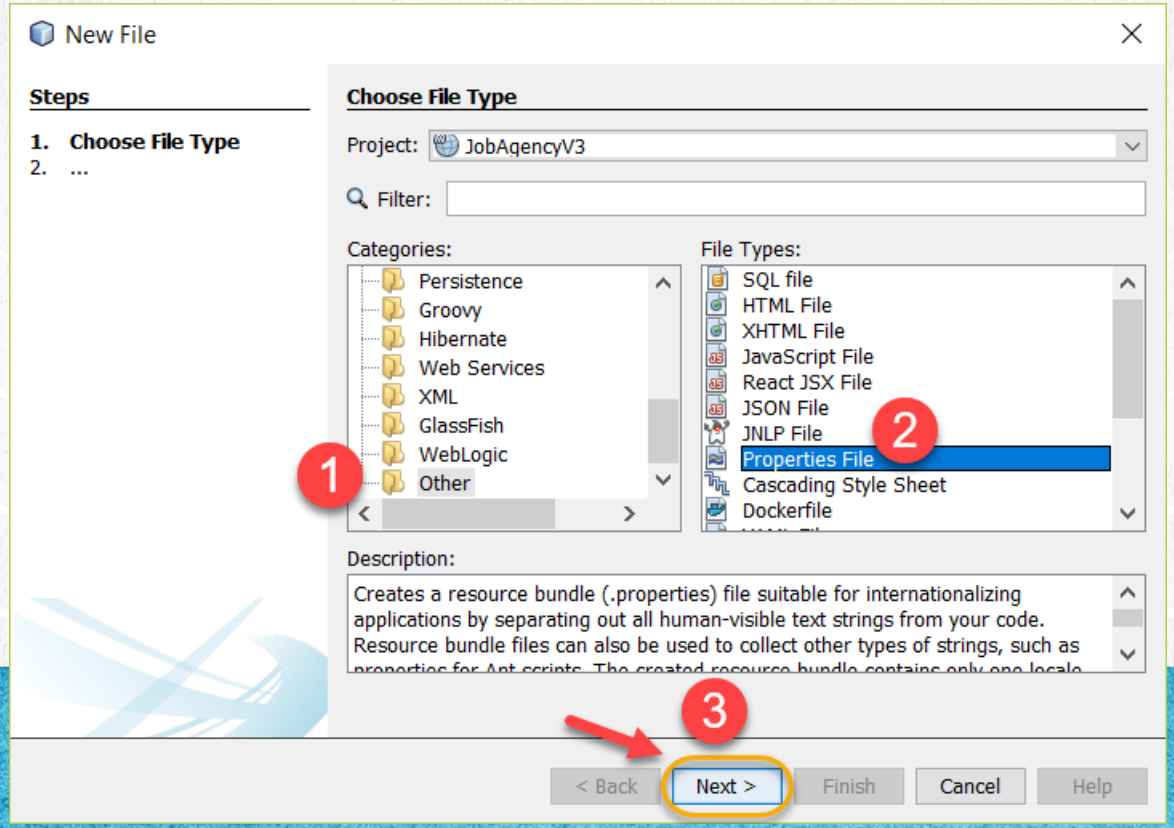
7. CREATE A PROPERTIES FILE

We create a properties file called jsf.properties:



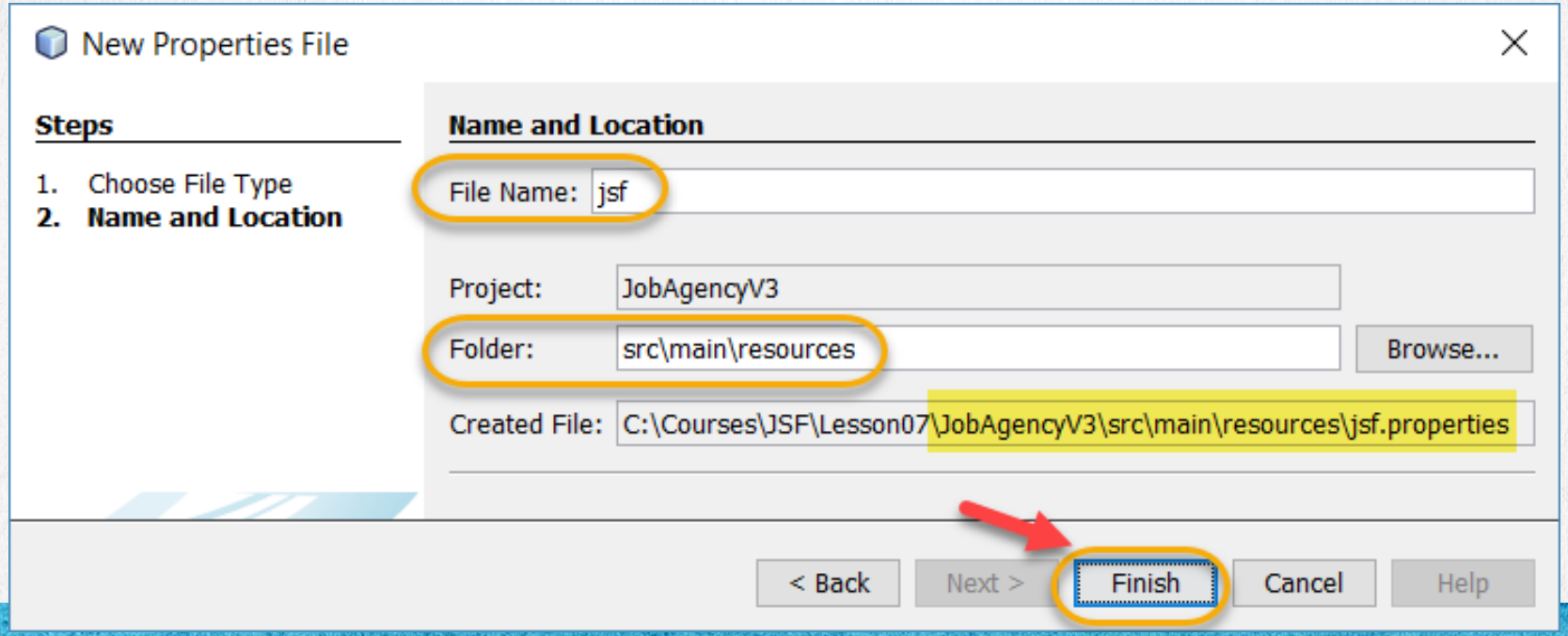
7. CREATE A PROPERTIES FILE

We create a properties file called jsf.properties:



7. CREATE A PROPERTIES FILE

We create a properties file called jsf.properties:



8. MODIFY THE FILE

jsf.properties:

Click to download

```
javax.faces.component.UIInput.REQUIRED={0}: Required Value
```

9. MODIFY THE XML FILE

We modified the faces-config.xml file. We add the configuration of the message management (resource bundle), the handling of JSF messages and we specify the language that we will handle by default in the JSF application.

10. MODIFY THE CODE

faces-config.xml:

Click to download

```
<?xml version="1.0" encoding="UTF-8"?>
<faces-config
    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-facesconfig_2_3.xsd"
    version="2.3">
    <lifecycle>
        <phase-listener>
            beans.lifecycle.DebuggerListener
        </phase-listener>
    </lifecycle>
    <application>
        <locale-config>
            <default-locale>en</default-locale>
        </locale-config>
        <!--form labels-->
        <resource-bundle>
            <!--No need to add the file extension-->
            <base-name>messages</base-name>
            <var>msgs</var>
        </resource-bundle>
        <!--change validator values-->
        <message-bundle>jsf</message-bundle>
    </application>
</faces-config>
```

8. MODIFY THE XHTML FILE

We modified the file index.xhtml so that now we use the names of the labels, no longer in hard code, but those defined in the file of mensaje.properties.

Let's see how the index.xhtml file is already modified:

Experiencia y Conocimiento para tu vida

8. MODIFY THE CODE

index.xhtml:

Click to download

```
<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml"
      xmlns:h="http://java.sun.com/jsf/html"
      xmlns:f="http://xmlns.jcp.org/jsf/core">
    <h:head>
        <title>Job Agency V3</title>
    </h:head>
    <h:body>
        <h:form>

            <h:messages globalOnly="true" ></h:messages>

            <table>
                <tr>
                    <td><h:outputLabel for="name" value="#{msgs['vacantForm.name']}" /></td>
                    <td><h:inputText id="name" required="true" value="#{candidate.name}" /></td>
                    <td><h:message for="name" /></td>
                </tr>
                <tr>
                    <td><h:outputLabel for="lastName" value="#{msgs['vacantForm.lastName']}" /></td>
                    <td><h:inputText id="lastName" required="true" value="#{candidate.lastName}" /></td>
                    <td><h:message for="lastName" /></td>
                </tr>
```

8. MODIFY THE CODE

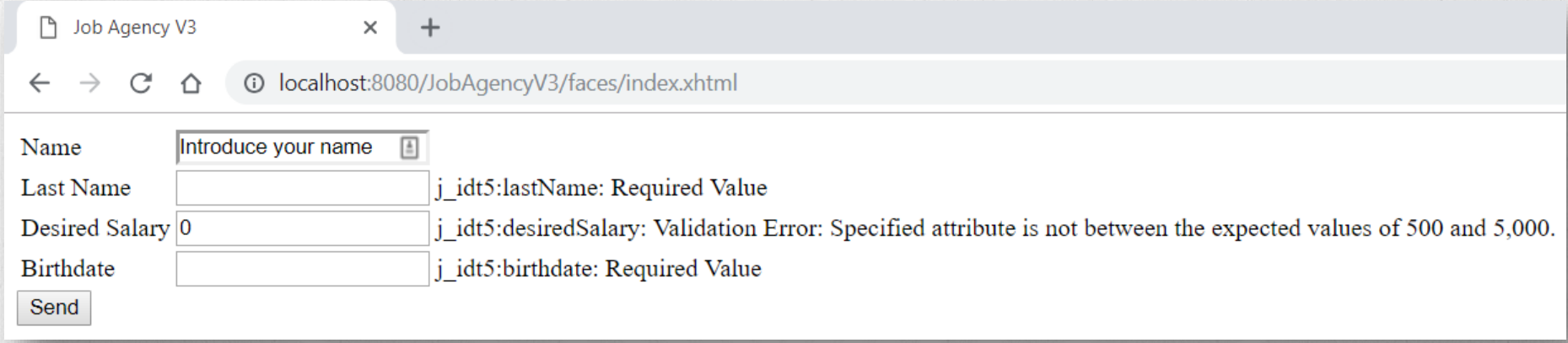
index.xhtml:

Click to download

```
                <tr>
                    <td><h:outputLabel for="desiredSalary" value="#{msgs['vacantForm.desiredSalary']}"  /></td>
                    <td><h:inputText id="desiredSalary" required="true"
                                     value="#{candidate.desiredSalary}">
                            <f:validateLongRange minimum="500" maximum="5000"/>
                        </h:inputText></td>
                    <td><h:message for="desiredSalary" /></td>
                </tr>
                <tr>
                    <td><h:outputLabel for="birthdate" value="#{msgs['vacantForm.birthdate']}" /></td>
                    <td><h:inputText id="birthdate" required="true"
                                     value="#{candidate.birthdate}">
                            <f:convertDateTime pattern="MM/dd/yyyy" />
                        </h:inputText></td>
                    <td><h:message for="birthdate" /></td>
                </tr>
            </table>
            <h:commandButton action="#{vacantForm.send}" value="#{msgs['vacantForm.send']}" />
        </h:form>
    </h:body>
</html>
```

9. EXECUTE THE PROJECT

Execute the Project:



EXERCISE CONCLUSION

With this exercise we put into practice the management concept of Messages with JSF. We customize both the messages of the JSF form, as well as the error or validation messages that we could have in our JSF application.

In the following exercise we will see the concept of Internationalization, which will allow us to add several languages to our Web application with JSF.

ONLINE COURSE

JAVASERVER FACES (JSF)

By: Eng. Ubaldo Acosta