

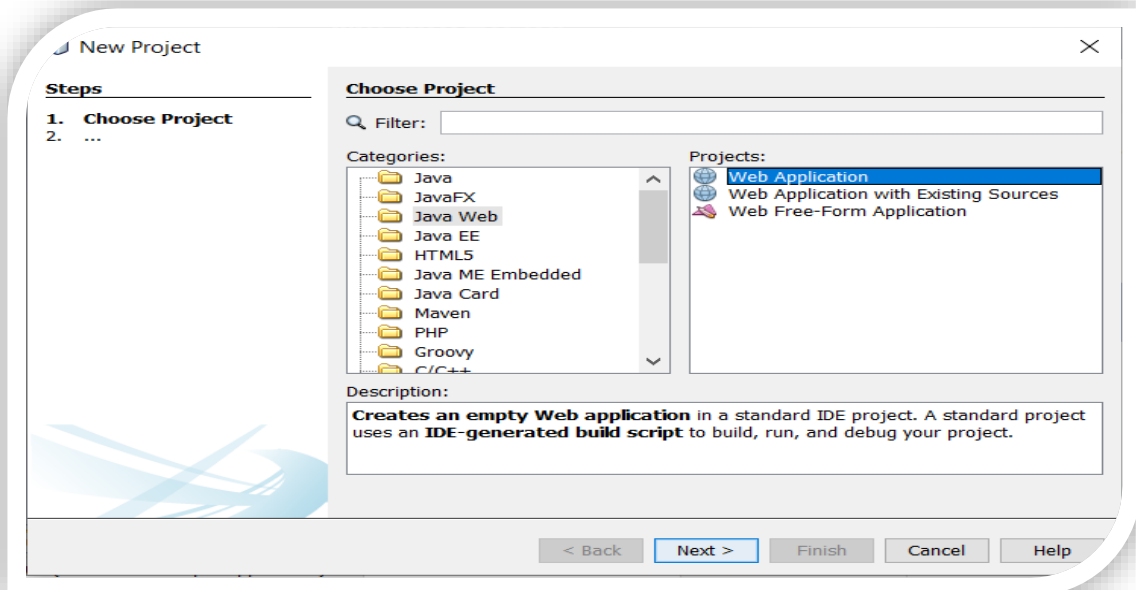
ATELIER 2 : JavaServer Faces 2.2

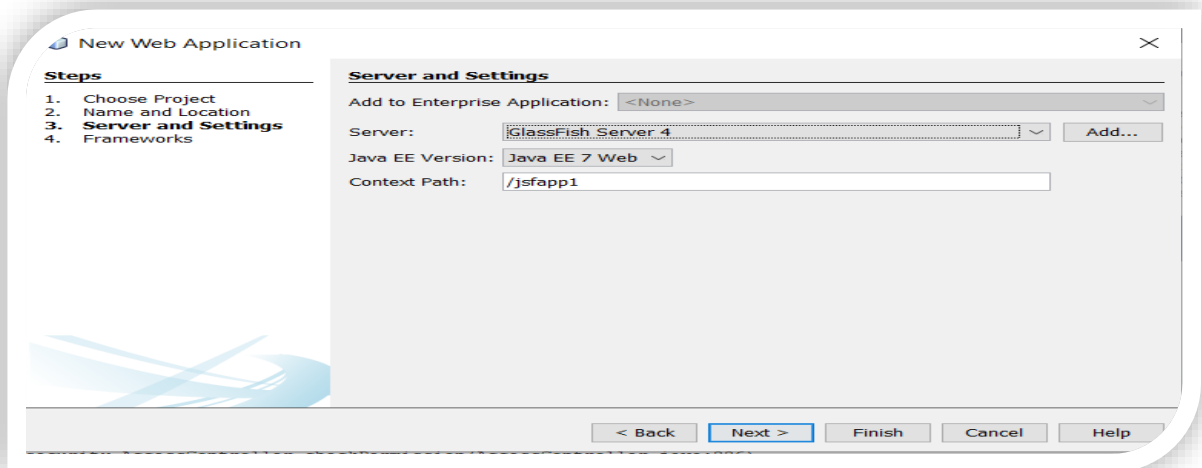
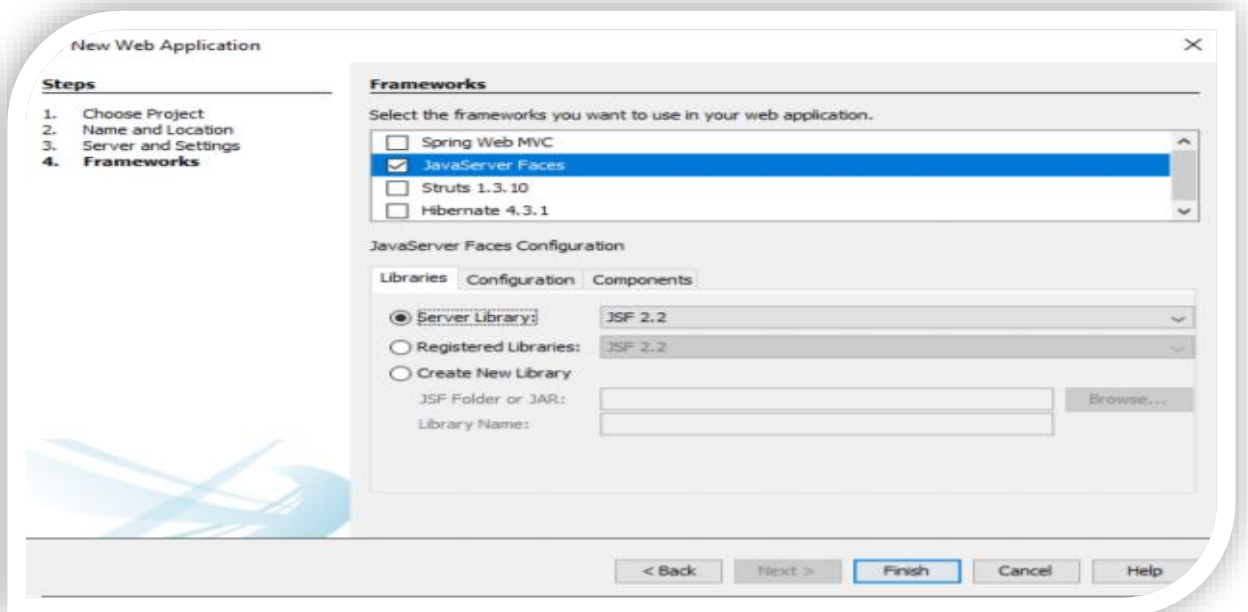
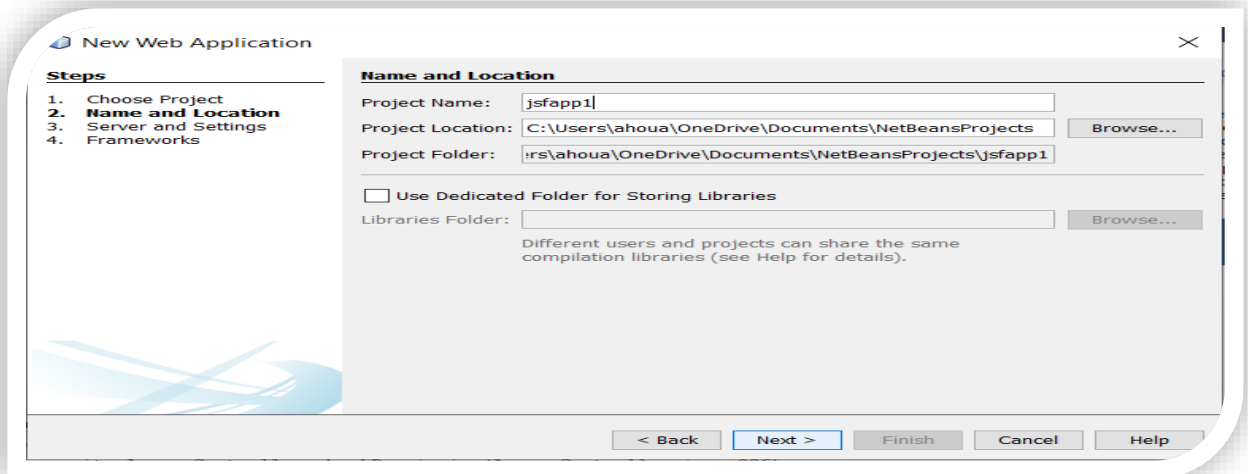
Objectif : création d'une application web JSF en utilisant les nouvelles fonctionnalités de JSF 2.2 Template de JSF et les composants composite....

ETAPE 1 : Création du nouveau projet.

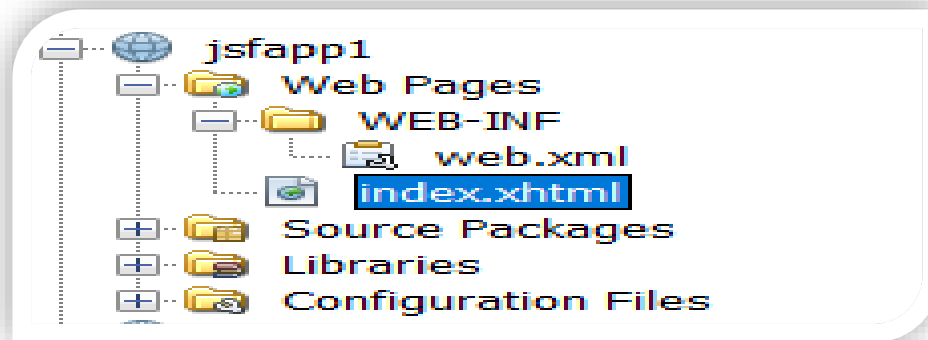
- 1- Pour créer un nouveau projet JSF, on doit aller dans **Fichier | Nouveau projet**, sélectionnez la catégorie **Java Web**, et **Web application** comme type de projet.

Les photos ci-dessous illustrent cette étape :





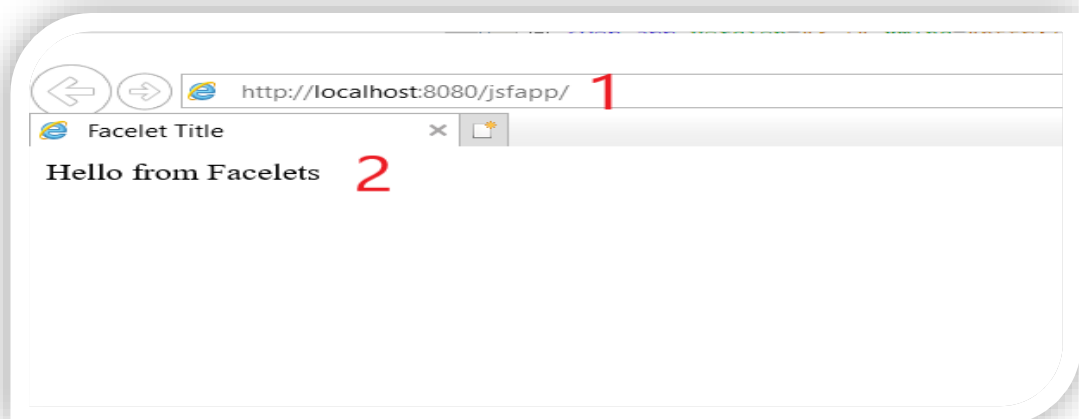
2- La structure finale de notre projet est la suivante.



3- A gauche sur NetBeans sur la section des projets un clique droit sur notre projet suivi d'une clique sur RUN pour démarrer et lancer notre application.

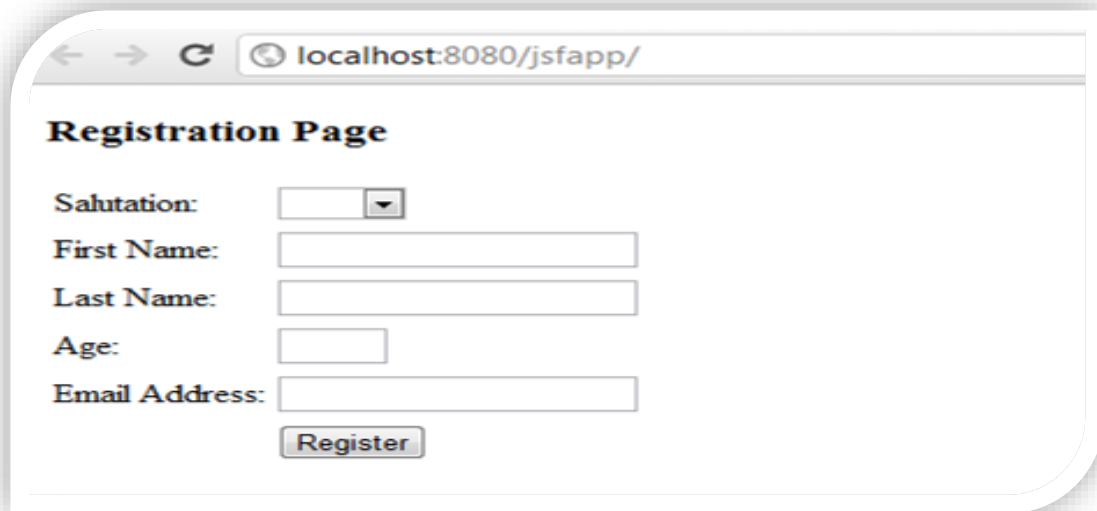
4- Sur le navigateur web notre application est lancer sous le lien :

[Localhost :8080/jsfapp](http://localhost:8080/jsfapp)



5- On modifier le code source du fichier index.xhtml sous le répertoire web-page, le nouveau code sera pour cree un formulaire pour récupérer les données des utilisateurs.

6- On actualise la page sur le navigateur et notre formulaire est afficher comme suit



The screenshot shows a web browser window with the address bar displaying "localhost:8080/jsfapp/". The page content is titled "Registration Page" in bold. Below the title, there is a registration form with the following fields and labels:

- Salutation:** A dropdown menu.
- First Name:** A text input field.
- Last Name:** A text input field.
- Age:** A text input field.
- Email Address:** A text input field.

At the bottom of the form, there is a "Register" button.

ETAPE 2 : sur cette étape on va configurer la navigation entre les pages et la récupération des données remplis sur le formulaire à l'aide d'un backing Bean et le valider à l'aide de validateur JSF et retourner une page de confirmation.

- 1- On commence par la création par faire des changement sur notre fichier index.xhtml.

```
Page x index.xhtml x web.xml x styles.css x index.xhtml x
<?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:h="http://xmlns.jcp.org/jsf/html"
xmlns:f="http://xmlns.jcp.org/jsf/core">
<h:head>
<title>Registration</title>
<h:outputStylesheet library="css" name="styles.css"/>
</h:head>
<h:body>
<h3>Registration Page</h3>
<h:form>
<h:panelGrid columns="3"

columnClasses="rightalign, leftalign, leftalign">
<h:outputLabel value="Salutation: " for="salutation"/>
<h:selectOneMenu id="salutation" label="Salutation"
value="#{registrationBean.salutation}" >
<f:selectItem itemLabel="" itemValue="" />
<f:selectItem itemLabel="Mr." itemValue="MR"/>
<f:selectItem itemLabel="Mrs." itemValue="MRS"/>

<f:selectItem itemLabel="Miss" itemValue="MISS"/>
<f:selectItem itemLabel="Ms" itemValue="MS"/>
<f:selectItem itemLabel="Dr." itemValue="DR"/>
</h:selectOneMenu>
<h:message for="salutation"/>

<h:outputLabel value="First Name:" for="firstName"/>
<h:inputText id="firstName" label="First Name"
required="true"
value="#{registrationBean.firstName}" />
<h:message for="firstName" />
<h:outputLabel value="Last Name:" for="lastName"/>
<h:inputText id="lastName" label="Last Name"
required="true"
value="#{registrationBean.lastName}" />
<h:message for="lastName" />

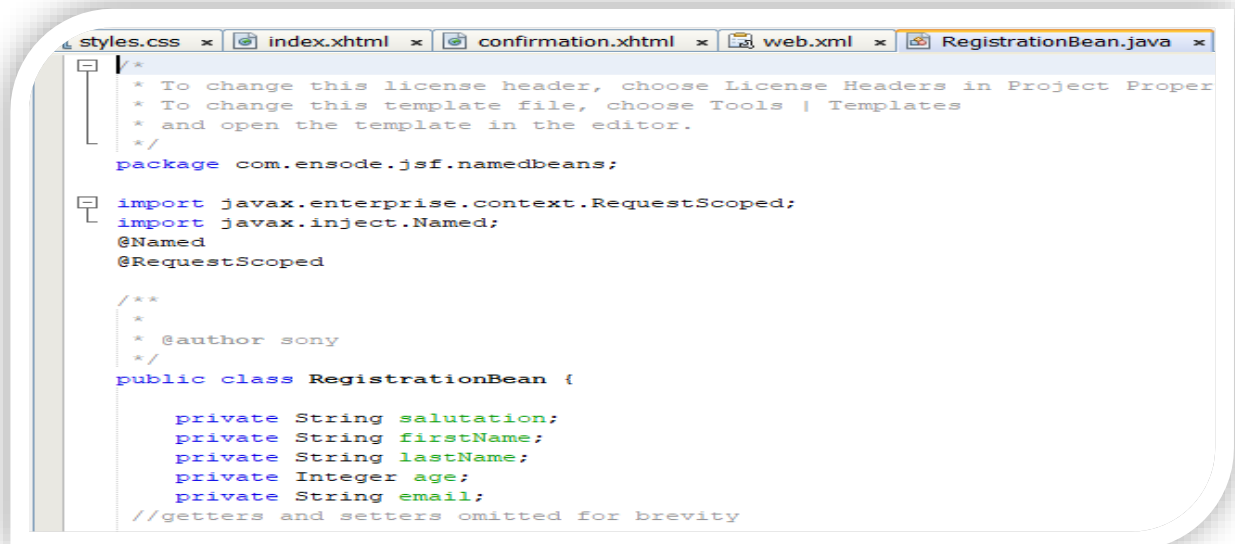
<h:outputLabel value="Email Address:" for="email"/>
<h:inputText id="email" label="Email Address"
required="true" value="#{registrationBean.email}">
<f:validator validatorId="emailValidator"/>
</h:inputText>
<h:message for="email" />

<h:panelGroup/>
<h:commandButton id="register" value="Register"
action="confirmation" />
</h:panelGrid>
</h:form>
</h:body>
</html>
```

```
<h:outputLabel value="Email Address:" for="email"/>
<h:inputText id="email" label="Email Address"
required="true" value="#{registrationBean.email}">
<f:validator validatorId="emailValidator"/>
</h:inputText>
<h:message for="email" />

<h:panelGroup/>
<h:commandButton id="register" value="Register"
action="confirmation" />
</h:panelGrid>
</h:form>
</h:body>
</html>
```

- 2- On cree un backing bean sous le packages namedbens, les backing beans sont des simples classe java avec des annotations.
Les backing beans sont utile sur étape pour récupération des donnees entrer par l'utilisateur.



```
/*
 * To change this license header, choose License Headers in Project Properties
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package com.ensode.jsf.namedbens;

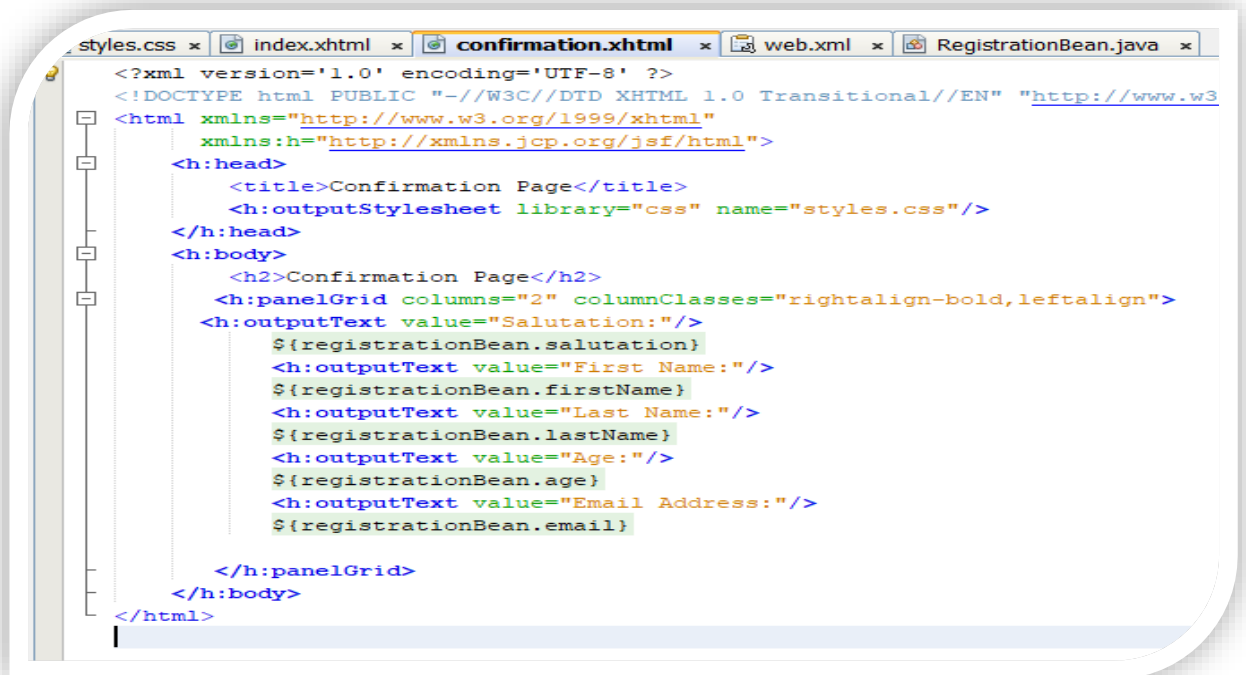
import javax.enterprise.context.RequestScoped;
import javax.inject.Named;
@Named
@RequestScoped

/**
 *
 * @author sony
 */
public class RegistrationBean {

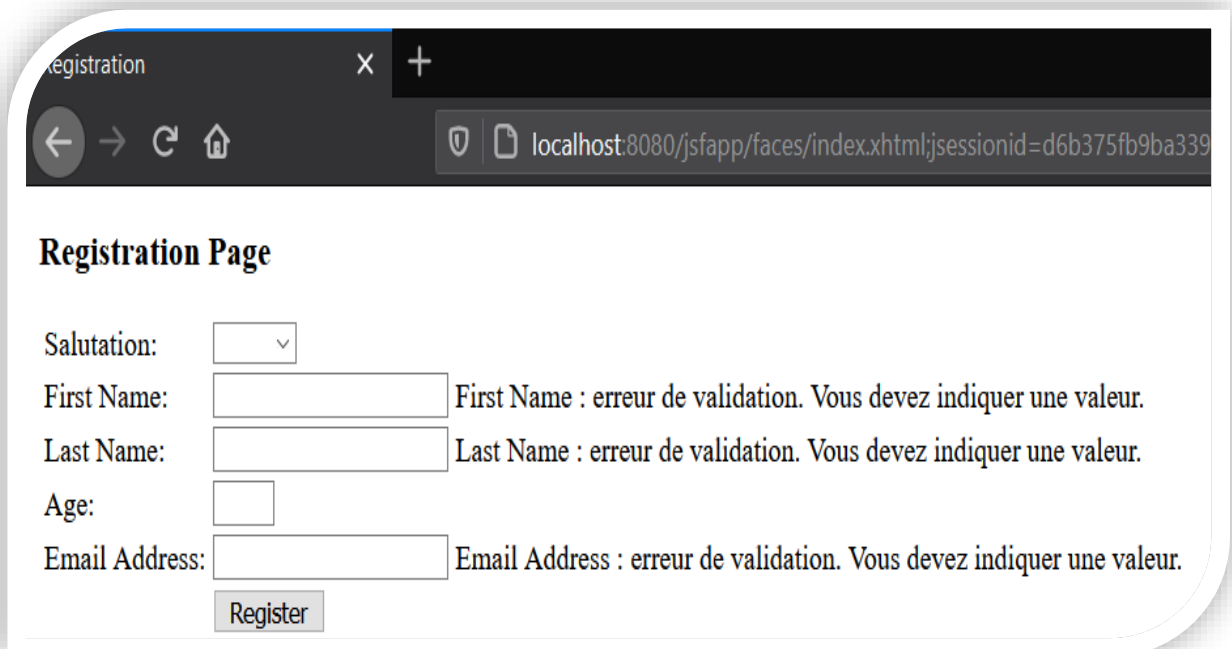
    private String salutation;
    private String firstName;
    private String lastName;
    private Integer age;
    private String email;
    //getters and setters omitted for brevity
}
```

- 3- Apres que le validateur de jsf fait sont travaille si les donnees entre par l'utilisateur respecte la forme du validateur on fait une redirection vers la page confirmation sinon on demande à l'utilisateur de remplir le formulaire en respectant les normes a l'aide des message générer par le validateur jsf.

4- Creation de la page confirmatio



```
<?xml version='1.0' encoding='UTF-8' ?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3
<html xmlns="http://www.w3.org/1999/xhtml"
      xmlns:h="http://xmlns.jcp.org/jsf/html">
  <h:head>
    <title>Confirmation Page</title>
    <h:outputStylesheet library="css" name="styles.css"/>
  </h:head>
  <h:body>
    <h2>Confirmation Page</h2>
    <h:panelGrid columns="2" columnClasses="rightalign-bold,leftalign">
      <h:outputText value="Salutation:"/>
        ${registrationBean.salutation}
      <h:outputText value="First Name:"/>
        ${registrationBean.firstName}
      <h:outputText value="Last Name:"/>
        ${registrationBean.lastName}
      <h:outputText value="Age:"/>
        ${registrationBean.age}
      <h:outputText value="Email Address:"/>
        ${registrationBean.email}
    </h:panelGrid>
  </h:body>
</html>
```



Registration

localhost:8080/jsfapp/faces/index.xhtml;jsessionid=d6b375fb9ba339

Registration Page

Salutation:

First Name: First Name : erreur de validation. Vous devez indiquer une valeur.

Last Name: Last Name : erreur de validation. Vous devez indiquer une valeur.

Age:

Email Address: Email Address : erreur de validation. Vous devez indiquer une valeur.

5- On peut créer nos propres validateurs et les utilisés sur nos formulaire.



```

/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package com.ensode.jsf.validators;
import java.util.regex.Matcher;
import java.util.regex.Pattern;
import javax.faces.application.FacesMessage;
import javax.faces.component.UIComponent;
import javax.faces.component.html.HtmlInputText;
import javax.faces.context.FacesContext;
import javax.faces.validator.Validator;
import javax.faces.validator.ValidatorException;
import javax.faces.validator.FacesValidator;
@FacesValidator(value="emailValidator")
public class EmailValidator implements Validator {

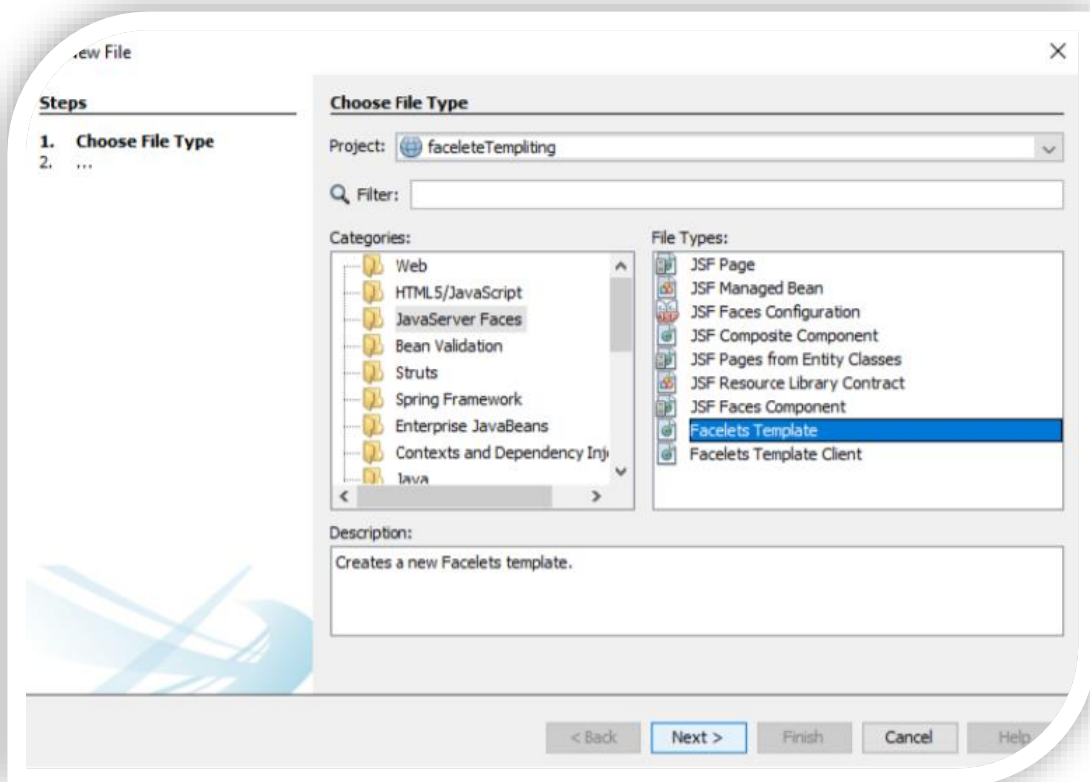
    /**
     * @param facesContext
     * @param uiComponent
     * @param value
     * @throws ValidatorException
     */
    @Override
    public void validate(FacesContext facesContext, UIComponent uiComponent, Object value) throws ValidatorException {
        Pattern pattern = Pattern.compile("\\w+@\\w+\\.\\w+");
        Matcher matcher = pattern.matcher((CharSequence) value);
        HtmlInputText htmlInputText = (HtmlInputText) uiComponent;
        String label;
        if (htmlInputText.getLabel() == null ||
            htmlInputText.getLabel().trim().equals("")) {
            label = htmlInputText.getId();
        } else {
            label = htmlInputText.getLabel();
        }
        if (!matcher.matches()) {
            FacesMessage facesMessage =
                new FacesMessage(label + ": not a valid email address");
            throw new ValidatorException(facesMessage);
        }
    }
}

```

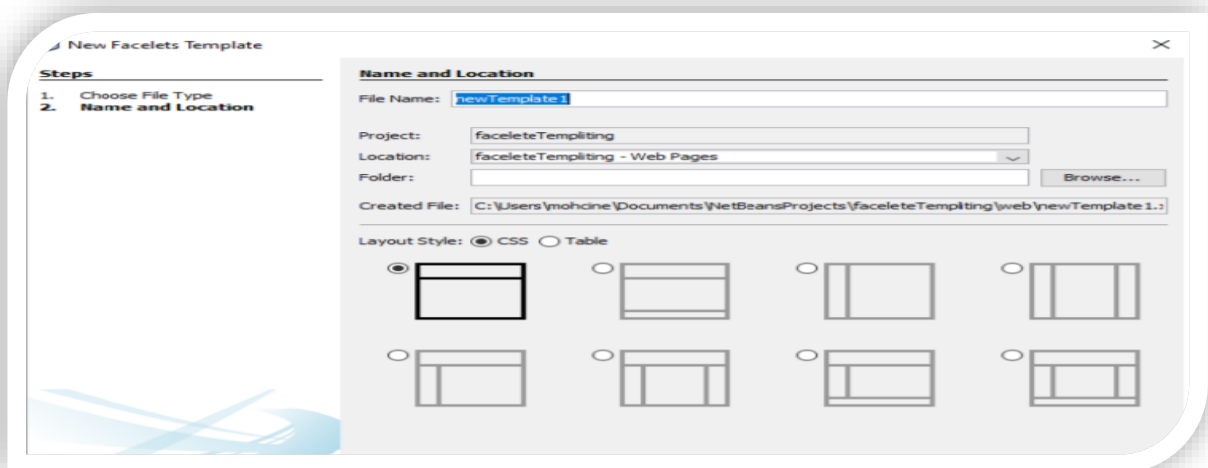
ETAPE 3 : utilisation des Templates des facelates jsf.

Facelets est officiellement devenu le Framework officiel de création de modèles JSF dans JSF 2.0. Nous définissons des modèles dans notre application Web, puis les intégrons dans notre page Web. Le modèle contient des espaces réservés dans lesquels nous allons pousser le contenu de notre page.

1-



2- on choisit notre model

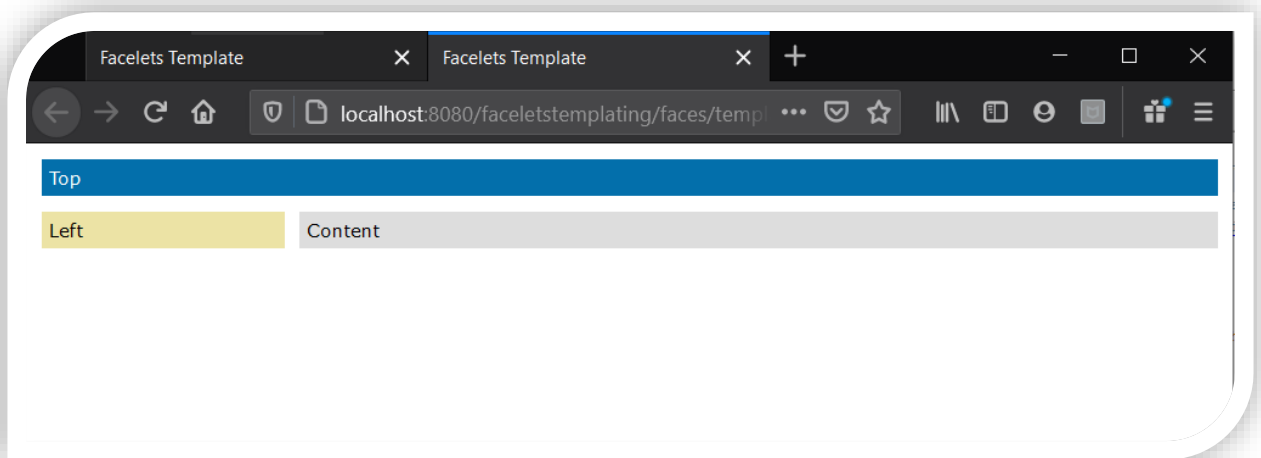


3-

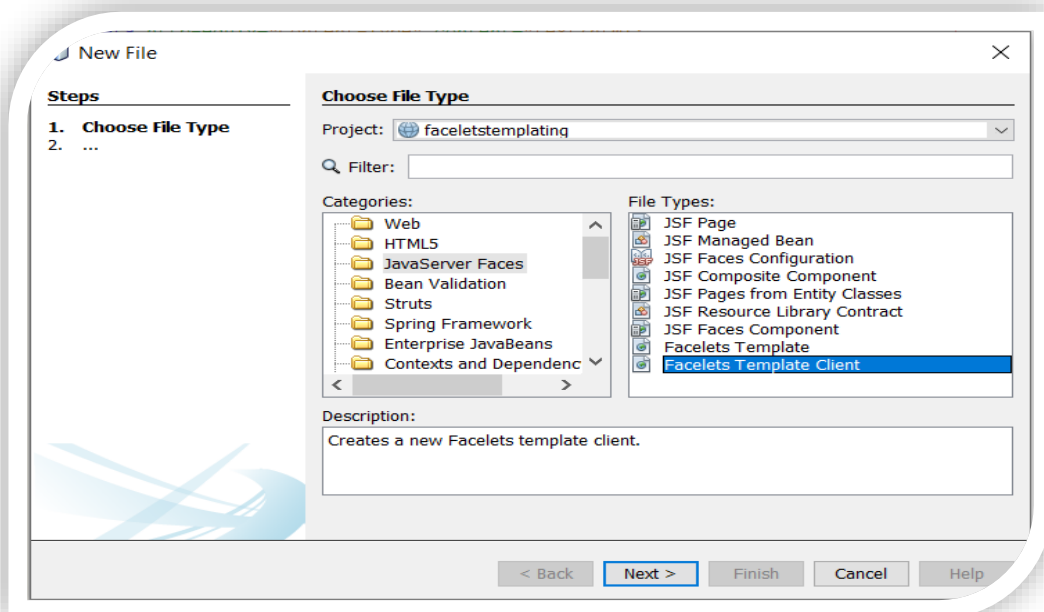
Il s'agit d'une page JSF assez standard qui a les éléments habituels tels qu'un corps et un en-tête avec des feuilles de style. Il a également plusieurs ui:insertbalises qui

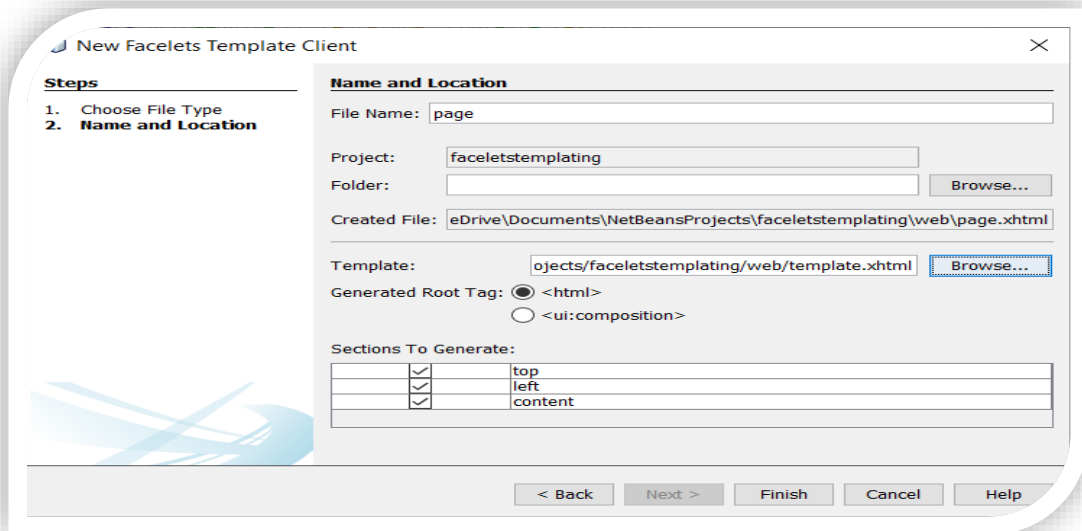
créent des espaces réservés dans notre modèle qui correspondent aux sections que nous avons sélectionnées dans la boîte de dialogue du modèle. Le nom sur les espaces réservés est utilisé pour pousser notre contenu dans les espaces réservés par `#{na}`, `#{e}`. Tout contenu dans les espaces réservés du modèle sera écrasé par le contenu client du modèle inséré dans l'espace réservé. Cependant, placer le contenu dans le modèle ici vous donne la possibilité d'avoir du contenu par défaut au cas où la page client du modèle ne l'utilise pas.

4 – sur le navigateur la vue du model est comme suit sur l'image ci-dessous

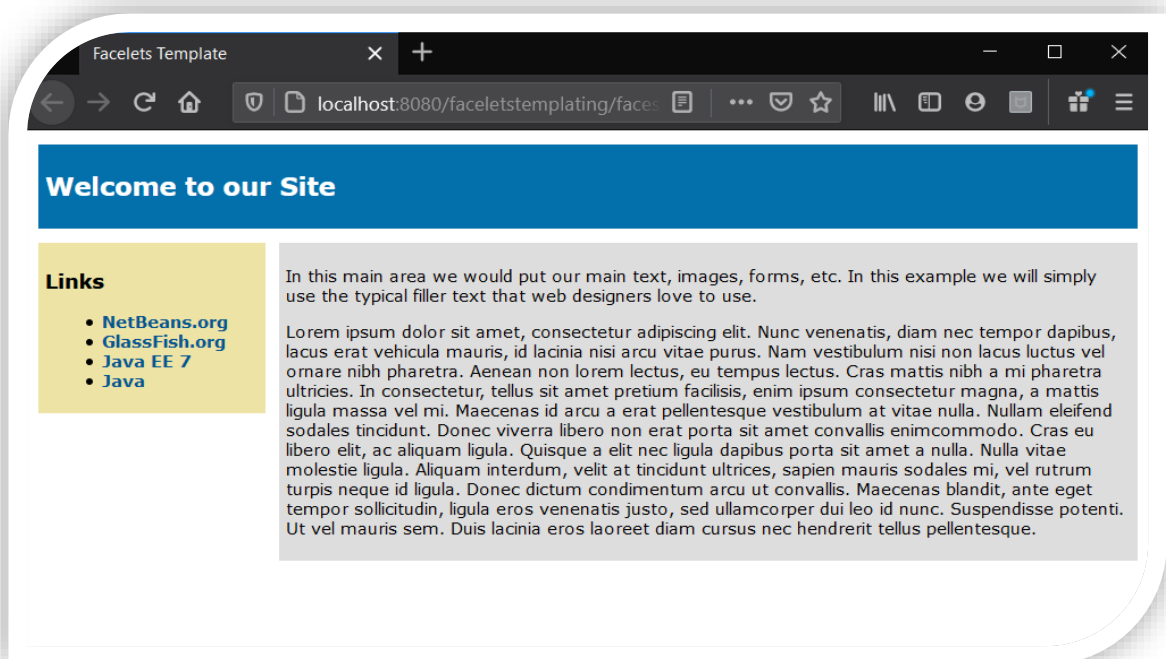


6- Creation du client de notre model



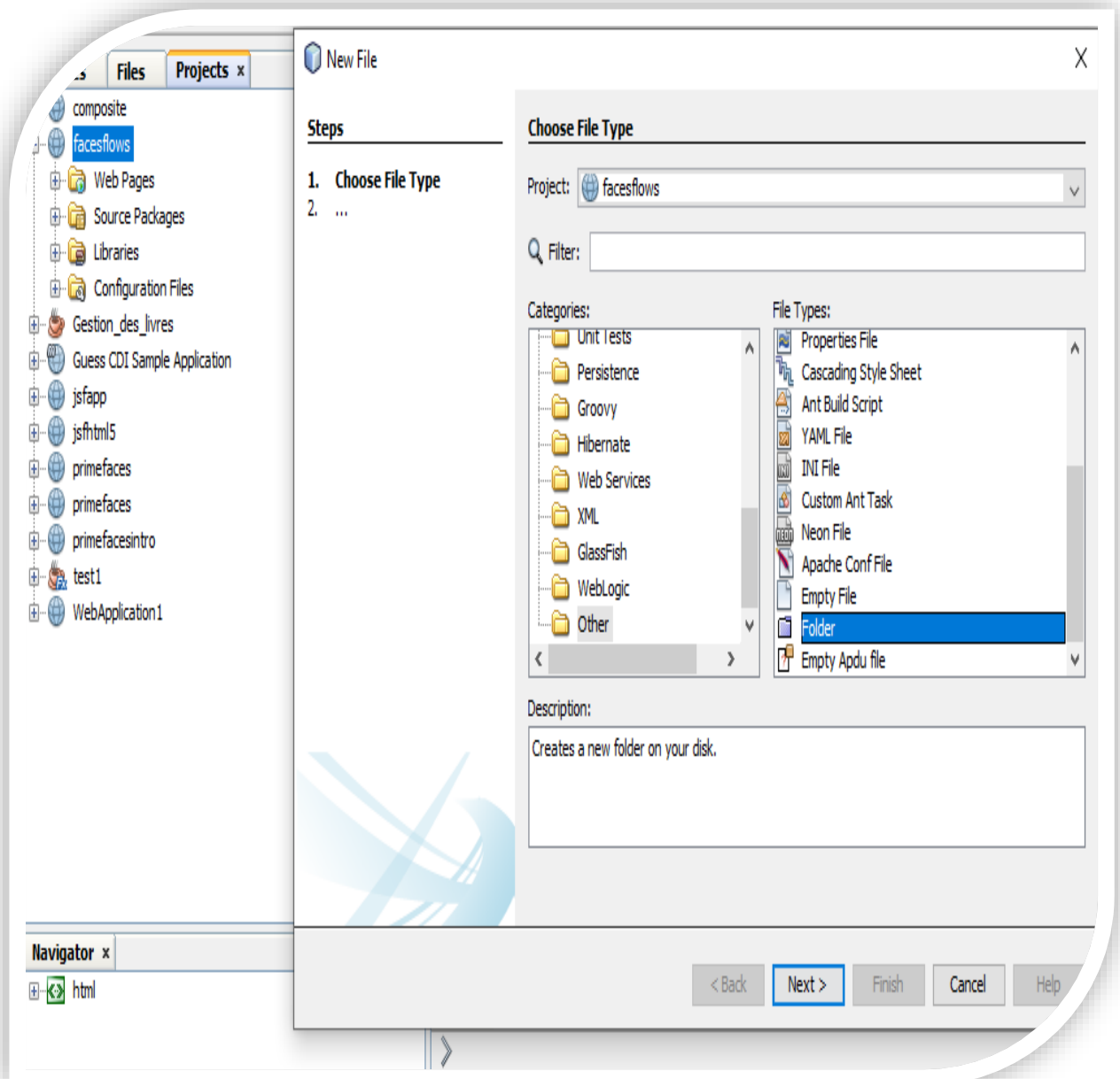


7- Sur le navigateur la vue de notre page client est comme ci-dessous

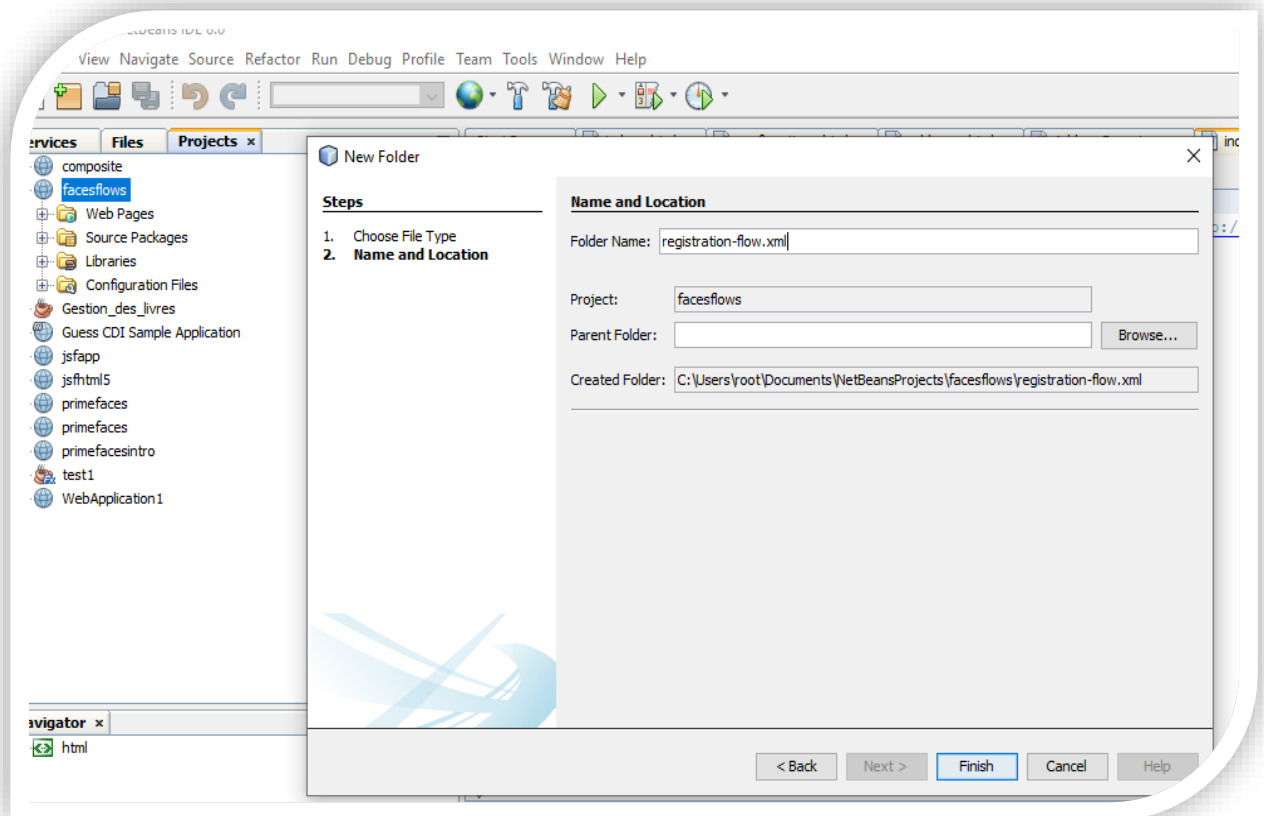


ETAPE 3 : utilisation des Flows faces.

Faces Flow a été inspiré par le framework Spring Flow populaire et en tant que tel, il n'est pas destiné à remplacer le système de navigation d'application Web; Plutôt, Faces Flow peut être utilisé pour encapsuler un ensemble d'étapes guidant l'utilisateur tout au long de l'exécution d'une tâche métier.




Pour faire fonctionner notre flow, on doit ajouter un fichier de configuration xml dans notre dossier. Ce fichier doit porter le nom du flow et suffixé par -flow, dans notre cas le nom du fichier sera : registration-flow.xml. Avec NetBeans, on peut ajouter le fichier en cliquant droit sur le dossier flow (registration), et aller dans New|Other, et enfin sélectionner Empty File dans la catégorie Other.



Notre page index.xhtml doit être comme suivant :



Le fichier registraion-pg2.xhtml est :



```
<?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:h="http://xmlns.jcp.org/jsf/html">
  <h:head>
    <title>Registration</title>
  </h:head>
  <h:body>
    <h3>Registration Page (Continued)</h3>
    <h:form>
      <h:panelGrid columns="2">
        <h:outputLabel for="line1" value="Line 1"></h:outputLabel>
        <h:inputText id="line1" value="#{registrationBean.line1}"></h:inputText>
        <h:outputLabel for="line2" value="Line 2"></h:outputLabel>
        <h:inputText id="line2" value="#{registrationBean.line2}"></h:inputText>
        <h:outputLabel for="city" value="City"></h:outputLabel>
        <h:inputText id="city" value="#{registrationBean.city}"></h:inputText>
        <h:outputLabel for="state" value="State"></h:outputLabel>
        <h:inputText id="state" value="#{registrationBean.state}" size="2" maxlength="2"></h:inputText>
        <h:outputLabel for="zip" value="Zip"></h:outputLabel>
        <h:inputText id="zip" value="#{registrationBean.zip}"></h:inputText>
        <h:commandButton id="back" value="Go Back" action="registration"></h:commandButton>
        <h:commandButton id="continue" value="Continue" action="registration-confirmation"></h:commandButton>
      </h:panelGrid>
    </h:form>
  </h:body>
</html>
```

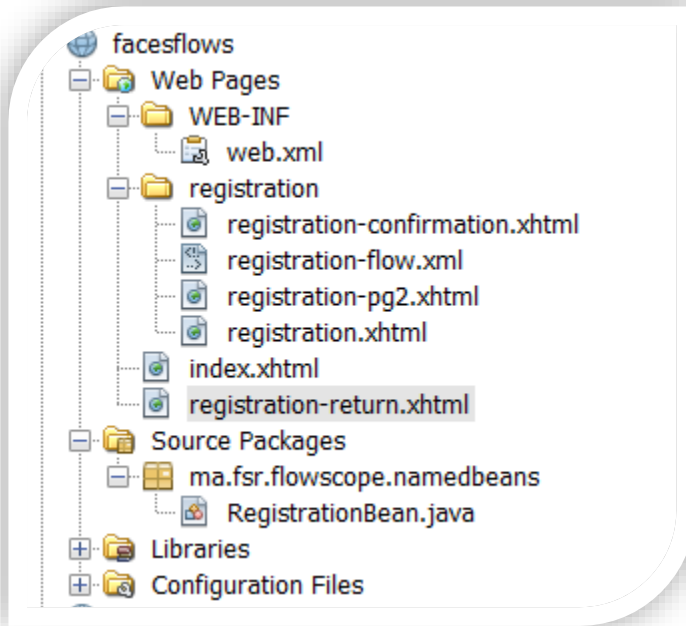
Le fichier registration-confirmation.xhtml

```
registration-pg2.xhtml | registration-confirmation.xhtml |
<?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-tr
<html xmlns="http://www.w3.org/1999/xhtml"
      xmlns:h="http://xmlns.jcp.org/jsf/html">
  <h:head>
    <title>Registration Confirmation</title>
  </h:head>
  <h:body>
    <h3>Registration Confirmation</h3>
    <h4>Personal Information</h4>
    #{registrationBean.salutation}<br/>
    #{registrationBean.firstName}<br/>
    #{registrationBean.lastName}<br/>
    #{registrationBean.age}<br/>
    #{registrationBean.email}<br/>
    <h4>Address Information</h4>
    #{registrationBean.line1}<br/>
    #{registrationBean.line2}<br/>
    #{registrationBean.city}<br/>
    #{registrationBean.state}<br/>
    #{registrationBean.zip}<br/>
    <h:form>
      <h:panelGroup>
        <h:commandButton value="Continue"
          action="registration-return"/>&#160;<h:commandButton value="Go Back" action="registration-pg2"/>
      </h:panelGroup>
    </h:form>
  </h:body>
</html>
```

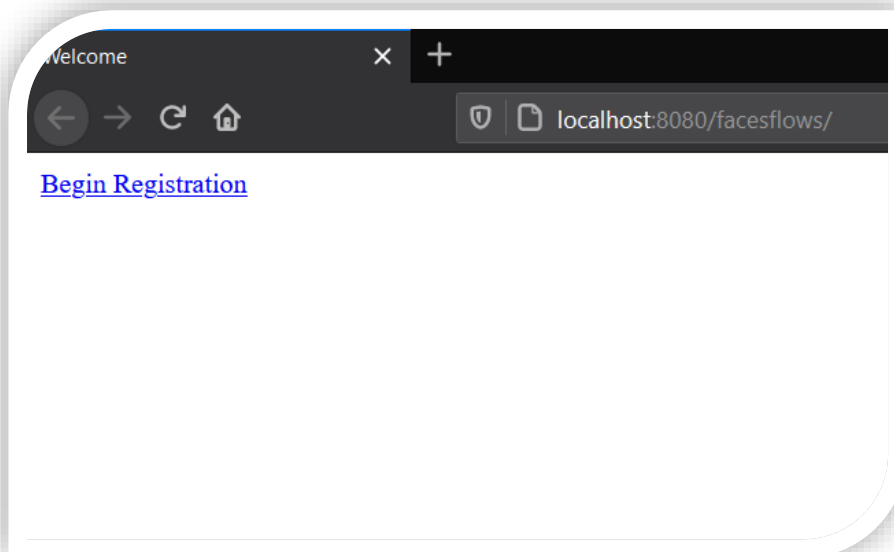
Le fichier registration-return.xhtml

```
registration-pg2.xhtml | registration-confirmation.xhtml |
<?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-tr
<html xmlns="http://www.w3.org/1999/xhtml"
      xmlns:h="http://xmlns.jcp.org/jsf/html">
  <h:head>
    <title>Registration Complete</title>
  </h:head>
  <h:body>
    Registration complete.
  </h:body>
</html>
```

Les fichiers nécessaires au projet sont organisés comme suit



Après la création des fichiers **registraion-pg2**, **registraion-confirmation**, **registraion-return** et le Bean **registraionBean** on exécute notre application.



Lorsque l'utilisateur clique sur le lien, la première page du flow est affichée.

Registration

← → ↻ 🏠 🔒 📄 localhost

Registration Page

Salutation:

First Name:

Last Name:

Age:

Email Address:

Après avoir rempli les données d'utilisateur :

On clic pour continue, On remarque que la navigation entre les pages marche bien.

Si vous voulez corriger vos informations on clic Go Back sinon Continue, il s'affiche une fenêtre d'affirmation que la registration complete.