Les noms du groupe:

Fode sylla
Oulimata laye ndiaye
Aida Diop
Babacar ndiour
Aboubakrine thiam

Projet Frameword JSF:

création des éléments métiers module EJB :

```
import javax.ejb.Stateless;

@Stateless
public class BusinessElementBean implements BusinessElementLocal {

    @Override
    public void doSomething() {
        // Implémentation de la logique métier
    }

    // Autres méthodes métier...
}

import javax.ejb.Local;

@Local
public interface BusinessElementLocal {
    void doSomething();
    // Autres méthodes métier...
}
```

Création d'une « entité bean » provenant de
la base de données:

```
import javax.persistence.Entity;
import javax.persistence.Id;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
```

```java
@Entity
public class EntiteBean {

    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;

    private String nom;
    private String description;

    // Constructeurs, getters et setters...

    // Exemple de constructeur sans arguments
    public EntiteBean() {
    }

    // Exemple de constructeur avec des arguments
    public EntiteBean(String nom, String description) {
        this.nom = nom;
        this.description = description;
    }

    // Getters et setters pour les propriétés

    public Long getId() {
        return id;
    }

    public void setId(Long id) {
        this.id = id;
    }

    public String getNom() {
        return nom;
    }

    public void setNom(String nom) {
        this.nom = nom;
    }
```

```
    public String getDescription() {
        return description;
    }

    public void setDescription(String description) {
        this.description = description;
    }
}
```

code création d’un
ManagedBean:

```
import javax.faces.bean.ManagedBean;
import javax.faces.bean.SessionScoped;

@ManagedBean(name = "exempleBean")
@SessionScoped
public class ExempleBean implements Serializable {

    private String donnee;

    // Constructeur sans arguments
    public ExempleBean() {
    }

    // Getter et setter pour la propriété donnee
    public String getDonnee() {
        return donnee;
    }

    public void setDonnee(String donnee) {
        this.donnee = donnee;
    }

    // Autres méthodes de gestion des actions liées à ce bean...
}
```

Créer une JSF index.xhtml pour afficher tous
les étudiants de la base données:

```
<!-- index.xhtml -->
<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml"
      xmlns:h="http://java.sun.com/jsf/html"
      xmlns:f="http://java.sun.com/jsf/core">

<h:head>
    <title>Liste des Étudiants</title>
</h:head>

<h:body>
    <h2>Liste des Étudiants</h2>

    <h:dataTable value="#{studentBean.allStudents}" var="student">
        <h:column>
            <f:facet name="header">ID</f:facet>
            #{student.id}
        </h:column>
        <h:column>
            <f:facet name="header">Nom</f:facet>
            #{student.nom}
        </h:column>
        <h:column>
            <f:facet name="header">Prénom</f:facet>
            #{student.prenom}
        </h:column>
    </h:dataTable>

</h:body>
</html>

@ManagedBean(name = "studentBean")
@ViewScoped
public class StudentBean implements Serializable {

    // Méthode pour récupérer tous les étudiants de la base de données
    public List<Student> getAllStudents() {
        // Implémentation pour récupérer la liste des étudiants depuis la base de données
        // Assurez-vous d'ajuster cela en fonction de votre logique de récupération de données
        return studentService.getAllStudents();
    }
```

```
    // Autres méthodes et propriétés du bean...
}
```

Insérer une « JSF Data Table » à partir de de l'entité « etudiant »:

```
<!-- students.xhtml -->
<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml"
      xmlns:h="http://java.sun.com/jsf/html"
      xmlns:f="http://java.sun.com/jsf/core">

<h:head>
    <title>Liste des Étudiants</title>
</h:head>

<h:body>
    <h2>Liste des Étudiants</h2>

    <h:dataTable value="#{studentBean.allStudents}" var="etudiant">
        <h:column>
            <f:facet name="header">ID</f:facet>
            #{etudiant.id}
        </h:column>
        <h:column>
            <f:facet name="header">Nom</f:facet>
            #{etudiant.nom}
        </h:column>
        <h:column>
            <f:facet name="header">Prénom</f:facet>
            #{etudiant.prenom}
        </h:column>
        <!-- Ajoutez d'autres colonnes selon les propriétés de votre entité "etudiant" -->
    </h:dataTable>

</h:body>
</html>
```

Insérer une « JSF Data Table » à partir de de

l'entité « etudiant » :

```
<!-- students.xhtml -->
<!DOCTYPE html>
<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>Liste des Étudiants</title>
</h:head>

<h:body>
   <h2>Liste des Étudiants</h2>

   <h:dataTable value="#{studentBean.allStudents}" var="etudiant">
      <h:column>
         <f:facet name="header">ID</f:facet>
         #{etudiant.id}
      </h:column>
      <h:column>
         <f:facet name="header">Nom</f:facet>
         #{etudiant.nom}
      </h:column>
      <h:column>
         <f:facet name="header">Prénom</f:facet>
         #{etudiant.prenom}
      </h:column>
      <!-- Ajoutez d'autres colonnes selon les propriétés de votre entité "etudiant" -->
   </h:dataTable>

</h:body>
</html>
```

création des vues Insérer un formulaire pour insérer un étudiant:

```
<!-- addStudent.xhtml -->
<!DOCTYPE html>
<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>Ajouter un Étudiant</title>
</h:head>

<h:body>
  <h2>Ajouter un Étudiant</h2>

  <h:form>
    <h:panelGrid columns="2">
      <h:outputLabel for="nom">Nom :</h:outputLabel>
      <h:inputText id="nom" value="#{studentBean.newStudent.nom}" required="true" />

      <h:outputLabel for="prenom">Prénom :</h:outputLabel>
      <h:inputText id="prenom" value="#{studentBean.newStudent.prenom}" required="true" />

      <!-- Ajoutez d'autres champs du formulaire en fonction des propriétés de votre entité "etudiant" -->

      <h:commandButton value="Ajouter" action="#{studentBean.addStudent}" />
    </h:panelGrid>
  </h:form>

</h:body>
</html>
```

Compléter le formulaire en ajouter le bouton de commande qui appel la méthode permettant d'ajouter l'étudiant dont les informations viennent d'être saisies et ensuite recharger la page index:

```
<!-- addStudent.xhtml -->
<!DOCTYPE html>
<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>Ajouter un Étudiant</title>
</h:head>

<h:body>
    <h2>Ajouter un Étudiant</h2>

    <h:form>
        <h:panelGrid columns="2">
            <h:outputLabel for="nom">Nom :</h:outputLabel>
            <h:inputText id="nom" value="#{studentBean.newStudent.nom}" required="true" />

            <h:outputLabel for="prenom">Prénom :</h:outputLabel>
            <h:inputText id="prenom" value="#{studentBean.newStudent.prenom}" required="true" /
>

            <!-- Ajoutez d'autres champs du formulaire en fonction des propriétés de votre entité
"etudiant" -->

            <h:commandButton value="Ajouter" action="#{studentBean.addStudent}" />
            <h:commandButton value="Recharger la Page" action="index?faces-redirect=true" />
        </h:panelGrid>
    </h:form>

</h:body>
</html>
```

Ajouter au formulaire une nouvelle colonne
pour les opérations
modification:

```
<!-- index.xhtml -->
<!DOCTYPE html>
<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>Liste des Étudiants</title>
</h:head>

<h:body>
```

```
    <h2>Liste des Étudiants</h2>

    <h:dataTable value="#{studentBean.allStudents}" var="etudiant">
        <h:column>
            <f:facet name="header">ID</f:facet>
            #{etudiant.id}
        </h:column>
        <h:column>
            <f:facet name="header">Nom</f:facet>
            #{etudiant.nom}
        </h:column>
        <h:column>
            <f:facet name="header">Prénom</f:facet>
            #{etudiant.prenom}
        </h:column>
        <!-- Ajoutez d'autres colonnes selon les propriétés de votre entité "etudiant" -->
        <h:column>
            <f:facet name="header">Opérations</f:facet>
            <h:commandButton value="Modifier" action="#{studentBean.editStudent(etudiant)}" />
        </h:column>
    </h:dataTable>

</h:body>
</html>
```

Ajouterau ManagerBean de la méthode
permettant de gérer l'opération de
suppression:

```
// StudentBean.java

import javax.faces.bean.ManagedBean;
import javax.faces.bean.ViewScoped;
import java.io.Serializable;
import java.util.List;

@ManagedBean(name = "studentBean")
@ViewScoped
public class StudentBean implements Serializable {
```

```
    private List<Student> allStudents;

    // Autres propriétés et méthodes...

    public List<Student> getAllStudents() {
        // Implémentation pour récupérer la liste des étudiants depuis la base de données
        // Assurez-vous d'ajuster cela en fonction de votre logique de récupération de données
        return studentService.getAllStudents();
    }

    public void deleteStudent(Student student) {
        // Implémentation pour supprimer l'étudiant de la base de données
        // Assurez-vous d'ajuster cela en fonction de votre logique de suppression
        studentService.deleteStudent(student);
    }

    // Autres méthodes...
}

<h:dataTable value="#{studentBean.allStudents}" var="etudiant">
    <!-- ... Colonnes existantes ... -->
    <h:column>
        <f:facet name="header">Opérations</f:facet>
        <h:commandButton value="Modifier" action="#{studentBean.editStudent(etudiant)}" />
        <h:commandButton value="Supprimer" action="#{studentBean.deleteStudent(etudiant)}" />
    </h:column>
</h:dataTable>
```

Ajoute des liens de suppression et modification :

```
<!-- index.xhtml -->

<h:dataTable value="#{studentBean.allStudents}" var="etudiant">
    <!-- ... Colonnes existantes ... -->
    <h:column>
        <f:facet name="header">Opérations</f:facet>
        <h:link value="Modifier" outcome="editStudent">
            <f:param name="studentId" value="#{etudiant.id}" />
        </h:link>
        <h:outputText value=" | " />
```

```
        <h:commandLink value="Supprimer" action="#{studentBean.deleteStudent(etudiant)}" />
    </h:column>
</h:dataTable>
```

Ajoute des liens de suppression et modification :

```
<!-- index.xhtml -->

<h:dataTable value="#{studentBean.allStudents}" var="etudiant">
    <h:column>
        <f:facet name="header">ID</f:facet>
        #{etudiant.id}
    </h:column>
    <h:column>
        <f:facet name="header">Nom</f:facet>
        #{etudiant.nom}
    </h:column>
    <h:column>
        <f:facet name="header">Prénom</f:facet>
        #{etudiant.prenom}
    </h:column>
    <h:column>
        <f:facet name="header">Date de Naissance</f:facet>
        #{etudiant.dateNaissance}
    </h:column>
    <h:column>
        <f:facet name="header">Opérations</f:facet>
        <h:link value="Modifier" outcome="editStudent">
            <f:param name="studentId" value="#{etudiant.id}" />
        </h:link>
        <h:outputText value=" | " />
        <h:commandLink value="Supprimer" action="#{studentBean.deleteStudent(etudiant)}" />
    </h:column>
</h:dataTable>
```

Ajoute des liens de suppression et modification:

```
<!-- index.xhtml -->

<h:dataTable value="#{studentBean.allStudents}" var="etudiant">
    <h:column>
```

```
        <f:facet name="header">ID</f:facet>
        #{etudiant.id}
    </h:column>
    <h:column>
        <f:facet name="header">Nom</f:facet>
        #{etudiant.nom}
    </h:column>
    <h:column>
        <f:facet name="header">Prénom</f:facet>
        #{etudiant.prenom}
    </h:column>
    <h:column>
        <f:facet name="header">Date de Naissance</f:facet>
        #{etudiant.dateNaissance}
    </h:column>
    <h:column>
        <f:facet name="header">Opérations</f:facet>
        <h:link value="Modifier" outcome="editStudent">
            <f:param name="studentId" value="#{etudiant.id}" />
        </h:link>
        <h:outputText value=" | " />
        <h:commandLink value="Supprimer" action="#{studentBean.deleteStudent(etudiant)}">
            <f:ajax execute="@form" render="@form" />
        </h:commandLink>
    </h:column>
</h:dataTable>
```

Modifier l'index modification :

```
<!-- index.xhtml -->

<h:dataTable value="#{studentBean.allStudents}" var="etudiant">
    <h:column>
        <f:facet name="header">ID</f:facet>
        #{etudiant.id}
    </h:column>
    <h:column>
        <f:facet name="header">Nom</f:facet>
        #{etudiant.nom}
    </h:column>
    <h:column>
```

```
        <f:facet name="header">Prénom</f:facet>
        #{etudiant.prenom}
    </h:column>
    <h:column>
        <f:facet name="header">Date de Naissance</f:facet>
        #{etudiant.dateNaissance}
    </h:column>
    <h:column>
        <f:facet name="header">Opérations</f:facet>
        <h:link value="Modifier" outcome="editStudent">
            <f:param name="studentId" value="#{etudiant.id}" />
        </h:link>
        <h:outputText value=" | " />
        <h:commandLink value="Supprimer" action="#{studentBean.deleteStudent(etudiant)}">
            <f:ajax execute="@form" render="@form" />
        </h:commandLink>
    </h:column>
</h:dataTable>
```

Ajouter un bouton pour mettre à jour les informations du formulaire :

```
<!-- editStudent.xhtml -->

<h:body>
    <h:form>
        <h:outputLabel for="nom">Nom :</h:outputLabel>
        <h:inputText id="nom" value="#{studentBean.selectedStudent.nom}" required="true" />

        <h:outputLabel for="prenom">Prénom :</h:outputLabel>
        <h:inputText id="prenom" value="#{studentBean.selectedStudent.prenom}"
required="true" />

        <!-- Ajoutez d'autres champs du formulaire en fonction des propriétés de votre entité
"etudiant" -->

        <h:commandButton value="Mettre à Jour" action="#{studentBean.updateStudent}" />
    </h:form>
</h:body>

// StudentBean.java
```

```
@ManagedBean(name = "studentBean")
@ViewScoped
public class StudentBean implements Serializable {

    private Student selectedStudent;

    // Autres propriétés et méthodes...

    public void editStudent(Student student) {
        // Charger les informations de l'étudiant sélectionné dans selectedStudent
        this.selectedStudent = student;
        // Naviguer vers la page de modification

FacesContext.getCurrentInstance().getApplication().getNavigationHandler().handleNavigation(Fa
cesContext.getCurrentInstance(), null, "editStudent.xhtml?faces-redirect=true");
    }

    public void updateStudent() {
        // Implémentation pour mettre à jour les informations de l'étudiant dans la base de données
        // Assurez-vous d'ajuster cela en fonction de votre logique de mise à jour
        studentService.updateStudent(selectedStudent);
        // Naviguer vers la page d'index après la mise à jour

FacesContext.getCurrentInstance().getApplication().getNavigationHandler().handleNavigation(Fa
cesContext.getCurrentInstance(), null, "index.xhtml?faces-redirect=true");
    }

    // Autres méthodes...
}
```

Ajouter un attribut au managerBean afin de
ou savoir s'il faut afficher un bouton ajouter
modifier sur le formulaire :

```
// StudentBean.java

@ManagedBean(name = "studentBean")
@ViewScoped
public class StudentBean implements Serializable {

    private Student selectedStudent;
```

```
    private boolean isEditMode;

    // Autres propriétés et méthodes...

    public void editStudent(Student student) {
        this.selectedStudent = student;
        this.isEditMode = true;

FacesContext.getCurrentInstance().getApplication().getNavigationHandler().handleNavigation(Fa
cesContext.getCurrentInstance(), null, "editStudent.xhtml?faces-redirect=true");
    }

    public void addOrUpdateStudent() {
        if (isEditMode) {
            // Implémentation pour mettre à jour les informations de l'étudiant dans la base de
données
            // Assurez-vous d'ajuster cela en fonction de votre logique de mise à jour
            studentService.updateStudent(selectedStudent);
        } else {
            // Implémentation pour ajouter un nouvel étudiant dans la base de données
            // Assurez-vous d'ajuster cela en fonction de votre logique d'ajout
            studentService.addStudent(selectedStudent);
        }
        // Naviguer vers la page d'index après l'ajout ou la mise à jour

FacesContext.getCurrentInstance().getApplication().getNavigationHandler().handleNavigation(Fa
cesContext.getCurrentInstance(), null, "index.xhtml?faces-redirect=true");
    }

    // Autres méthodes...
}
```

Projet primeface:

Très Nombreux champs supplémentaires :
ColorPicker, Calendar, TagCloud…
Panneaux, graphiques, multimédia, gestion
de fichiers…
Gestion par JSF d’images dynamiques (en
JSF pures, gérées par… une servlet).
Extension de la spécification : langage plus
riche pour désigner des éléments.

```
<dependency>
   <groupId>org.primefaces</groupId>
   <artifactId>primefaces</artifactId>
   <version>YOUR_PRIMEFACES_VERSION</version>
</dependency>

<!-- Exemple de page avec des composants PrimeFaces -->

<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml"
      xmlns:h="http://xmlns.jcp.org/jsf/html"
      xmlns:f="http://xmlns.jcp.org/jsf/core"
      xmlns:p="http://primefaces.org/ui">

<h:head>
   <title>Exemple PrimeFaces</title>
</h:head>

<h:body>
   <h:form>
      <!-- ColorPicker -->
      <p:colorPicker value="#{yourManagedBean.color}" />

      <!-- Calendar -->
      <p:calendar value="#{yourManagedBean.date}" />

      <!-- TagCloud -->
      <p:tagCloud model="#{yourManagedBean.tagCloudModel}" />

      <!-- Panel -->
      <p:panel header="Panel Title">
         Contenu du panneau...
      </p:panel>

      <!-- Chart -->
      <p:chart type="line" model="#{yourManagedBean.chartModel}" />

      <!-- Multimédia -->
      <p:media value="/resources/media/sample.mp3" player="quicktime" width="300"
height="50" />
```

```
        <!-- Gestion de fichiers -->
        <p:fileUpload fileUploadListener="#{yourManagedBean.handleFileUpload}" />

        <!-- Gestion d'images dynamiques avec une servlet -->
        <p:graphicImage value="/yourDynamicImageServlet" />

        <!-- Langage riche pour désigner des éléments -->
        <p:inputText value="#{yourManagedBean.richText}" />

        <!-- Ajoutez d'autres composants PrimeFaces selon vos besoins -->
    </h:form>
</h:body>

</html>

// Exemple de servlet pour gérer des images dynamiques

import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import java.io.IOException;
import java.io.OutputStream;

@WebServlet("/yourDynamicImageServlet")
public class YourDynamicImageServlet extends HttpServlet {

    @Override
    protected void doGet(HttpServletRequest request, HttpServletResponse response) throws
ServletException, IOException {
        // Logique pour générer l'image dynamique
        // Exemple : Génération d'une image et écriture dans le flux de sortie
        byte[] dynamicImageData = YourImageGenerationLogic.generateImage();
        response.setContentType("image/png");
        response.setContentLength(dynamicImageData.length);
        OutputStream output = response.getOutputStream();
        output.write(dynamicImageData);
        output.close();
    }
```

```
}
```

Du calendrier du formulaire :

```
<!-- Exemple de page avec le composant p:calendar de PrimeFaces -->

<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml"
      xmlns:h="http://xmlns.jcp.org/jsf/html"
      xmlns:f="http://xmlns.jcp.org/jsf/core"
      xmlns:p="http://primefaces.org/ui">

<h:head>
   <title>Formulaire avec Calendrier PrimeFaces</title>
</h:head>

<h:body>
   <h:form>
      <h:panelGrid columns="2">
         <h:outputLabel for="dateNaissance">Date de Naissance :</h:outputLabel>
         <p:calendar id="dateNaissance" value="#{yourManagedBean.dateNaissance}"
pattern="dd/MM/yyyy" showOn="button" />

         <!-- Ajoutez d'autres champs du formulaire en fonction de vos besoins -->
         <!-- Par exemple, ColorPicker, TagCloud, etc. -->

         <h:commandButton value="Enregistrer" action="#{yourManagedBean.enregistrer}" />
      </h:panelGrid>
   </h:form>
</h:body>

</html>
```

Simple accés au web cam:

```
<!-- Exemple de page JSF avec PrimeFaces et accès à la webcam -->

<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml"
      xmlns:h="http://xmlns.jcp.org/jsf/html"
      xmlns:p="http://primefaces.org/ui">
```

```
<h:head>
   <title>Accès à la Webcam</title>
   <script>
      function startWebcam() {
         var video = document.getElementById('webcam');

         // Utiliser getUserMedia pour accéder à la webcam
         navigator.mediaDevices.getUserMedia({ video: true })
            .then(function (stream) {
               video.srcObject = stream;
            })
            .catch(function (error) {
               console.log('Erreur d\'accès à la webcam : ', error);
            });
      }
   </script>
</h:head>

<h:body>
   <h:form>
      <p:commandButton value="Démarrer la Webcam" onclick="startWebcam()" />

      <!-- Ajouter une balise video pour afficher le flux de la webcam -->
      <video id="webcam" width="640" height="480" autoplay></video>
   </h:form>
</h:body>

</html>
```

AccordionPanel:

```
<!-- Exemple de page JSF avec PrimeFaces et p:accordionPanel -->

<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml"
      xmlns:h="http://xmlns.jcp.org/jsf/html"
      xmlns:f="http://xmlns.jcp.org/jsf/core"
      xmlns:p="http://primefaces.org/ui">

<h:head>
```

```
    <title>Exemple d'AccordionPanel PrimeFaces</title>
</h:head>

<h:body>
    <h:form>
        <p:accordionPanel multiple="true" activeIndex="0,1">
            <!-- Onglet 1 -->
            <p:tab title="Onglet 1">
                <h:outputText value="Contenu de l'onglet 1" />
            </p:tab>

            <!-- Onglet 2 -->
            <p:tab title="Onglet 2">
                <h:outputText value="Contenu de l'onglet 2" />
            </p:tab>

            <!-- Onglet 3 -->
            <p:tab title="Onglet 3">
                <h:outputText value="Contenu de l'onglet 3" />
            </p:tab>
        </p:accordionPanel>
    </h:form>
</h:body>

</html>
```

Carousel:

```
<!-- Exemple de page JSF avec PrimeFaces et p:carousel -->

<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml"
      xmlns:h="http://xmlns.jcp.org/jsf/html"
      xmlns:f="http://xmlns.jcp.org/jsf/core"
      xmlns:p="http://primefaces.org/ui">

<h:head>
    <title>Exemple de Carousel PrimeFaces</title>
</h:head>

<h:body>
```

```
    <h:form>
        <p:carousel value="#{yourManagedBean.images}" var="image" numVisible="3"
responsive="true">
            <p:graphicImage name="/resources/images/#{image}" alt="Image" />
        </p:carousel>
    </h:form>
</h:body>

</html>
```

ContentFlow :

```
<!-- Exemple de galerie d'images avec PrimeFaces p:carousel -->

<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml"
      xmlns:h="http://xmlns.jcp.org/jsf/html"
      xmlns:f="http://xmlns.jcp.org/jsf/core"
      xmlns:p="http://primefaces.org/ui">

<h:head>
    <title>Exemple de Galerie d'Images PrimeFaces</title>
</h:head>

<h:body>
    <h:form>
        <p:carousel numVisible="3" responsive="true">
            <p:graphicImage name="/resources/images/image1.jpg" alt="Image 1" />
            <p:graphicImage name="/resources/images/image2.jpg" alt="Image 2" />
            <p:graphicImage name="/resources/images/image3.jpg" alt="Image 3" />
            <!-- Ajoutez d'autres images en fonction de vos besoins -->
        </p:carousel>
    </h:form>
</h:body>

</html>
```

Galleria exemple code primeface

```
<h:head>
    <title>Exemple de Galleria avec PrimeFaces</title>
```

```
    <h:outputScript library="primefaces" name="jquery/jquery.js" />
    <h:outputScript name="galleria/galleria-1.5.7.min.js" />
    <style type="text/css">
        @import url("galleria/themes/classic/galleria.classic.min.css");
    </style>
</h:head>

<h:body>
    <h:form>
        <div id="galleria">
            <g:graphicImage name="images/image1.jpg" alt="Image 1" />
            <g:graphicImage name="images/image2.jpg" alt="Image 2" />
            <g:graphicImage name="images/image3.jpg" alt="Image 3" />
            <!-- Ajoutez d'autres images en fonction de vos besoins -->
        </div>
    </h:form>

    <script>
        Galleria.loadTheme('galleria/themes/classic/galleria.classic.min.js');
        Galleria.run('#galleria');
    </script>
</h:body>
```

Color Picker exemple code primeface:

```
<!-- Exemple de page JSF avec PrimeFaces et p:colorPicker -->

<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml"
      xmlns:h="http://xmlns.jcp.org/jsf/html"
      xmlns:f="http://xmlns.jcp.org/jsf/core"
      xmlns:p="http://primefaces.org/ui">

<h:head>
    <title>Exemple de Color Picker PrimeFaces</title>
</h:head>

<h:body>
    <h:form>
        <p:outputLabel for="colorPicker" value="Sélectionnez une couleur : " />
        <p:colorPicker id="colorPicker" value="#{yourManagedBean.selectedColor}"
```

```
mode="popup" />

      <p:commandButton value="Enregistrer" action="#{yourManagedBean.enregistrer}" />
   </h:form>
</h:body>

</html>
```

Dock:

```
<!-- Exemple de page JSF avec PrimeFaces et p:menubar -->

<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml"
      xmlns:h="http://xmlns.jcp.org/jsf/html"
      xmlns:f="http://xmlns.jcp.org/jsf/core"
      xmlns:p="http://primefaces.org/ui">

<h:head>
   <title>Exemple de Menubar PrimeFaces</title>
</h:head>

<h:body>
   <h:form>
      <p:menubar>
         <p:submenu label="Fichier">
            <p:menuitem value="Nouveau" icon="pi pi-fw pi-plus" />
            <p:menuitem value="Ouvrir" icon="pi pi-fw pi-folder-open" />
            <p:menuitem value="Enregistrer" icon="pi pi-fw pi-save" />
         </p:submenu>

         <p:submenu label="Éditer">
            <p:menuitem value="Couper" icon="pi pi-fw pi-cut" />
            <p:menuitem value="Copier" icon="pi pi-fw pi-copy" />
            <p:menuitem value="Coller" icon="pi pi-fw pi-paste" />
         </p:submenu>

         <!-- Ajoutez d'autres sous-menus et éléments du menu en fonction de vos besoins -->
      </p:menubar>
   </h:form>
</h:body>
```

```
</html>
```

GMap:

```
<!-- Exemple de page JSF avec PrimeFaces et p:gmap -->

<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml"
      xmlns:h="http://xmlns.jcp.org/jsf/html"
      xmlns:f="http://xmlns.jcp.org/jsf/core"
      xmlns:p="http://primefaces.org/ui">

<h:head>
   <title>Exemple de GMap PrimeFaces</title>
</h:head>

<h:body>
   <h:form>
      <p:gmap center="41.850033, -87.6500523" zoom="10" style="width:100%;height:400px">
         <!-- Ajoutez des marqueurs sur la carte -->
         <p:gmapMarker position="41.850033, -87.6500523" title="Chicago" />
         <p:gmapMarker position="40.712776, -74.005974" title="New York" />
         <p:gmapMarker position="34.052235, -118.243683" title="Los Angeles" />

         <!-- Personnalisez davantage les marqueurs si nécessaire -->
         <!-- <p:gmapInfoWindow>
            Contenu de la fenêtre d'info
         </p:gmapInfoWindow> -->
      </p:gmap>
   </h:form>
</h:body>

</html>
```

Keyboard :

```
<!-- Exemple de page JSF avec des champs de saisie PrimeFaces -->

<!DOCTYPE html>
```

```
<html xmlns="http://www.w3.org/1999/xhtml"
      xmlns:h="http://xmlns.jcp.org/jsf/html"
      xmlns:f="http://xmlns.jcp.org/jsf/core"
      xmlns:p="http://primefaces.org/ui">

<h:head>
   <title>Exemple de Clavier Virtuel PrimeFaces</title>
</h:head>

<h:body>
   <h:form>
      <p:inputText id="username" value="#{yourManagedBean.username}" />
      <p:password id="password" value="#{yourManagedBean.password}" />

      <!-- Ajoutez d'autres champs de saisie selon vos besoins -->

      <p:commandButton value="Soumettre" action="#{yourManagedBean.submit}" />
   </h:form>
</h:body>

</html>
```

Ajouter le nom du thème approprié dans le
fichier « web.xml »

```
<!-- Exemple de configuration du thème PrimeFaces dans le fichier web.xml -->

<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_3_1.xsd"
         version="3.1">

   <!-- Autres configurations du fichier web.xml -->

   <context-param>
      <param-name>primefaces.THEME</param-name>
      <param-value>your_theme_name</param-value>
   </context-param>

</web-app>
```

<param-value>cupertino</param-value>