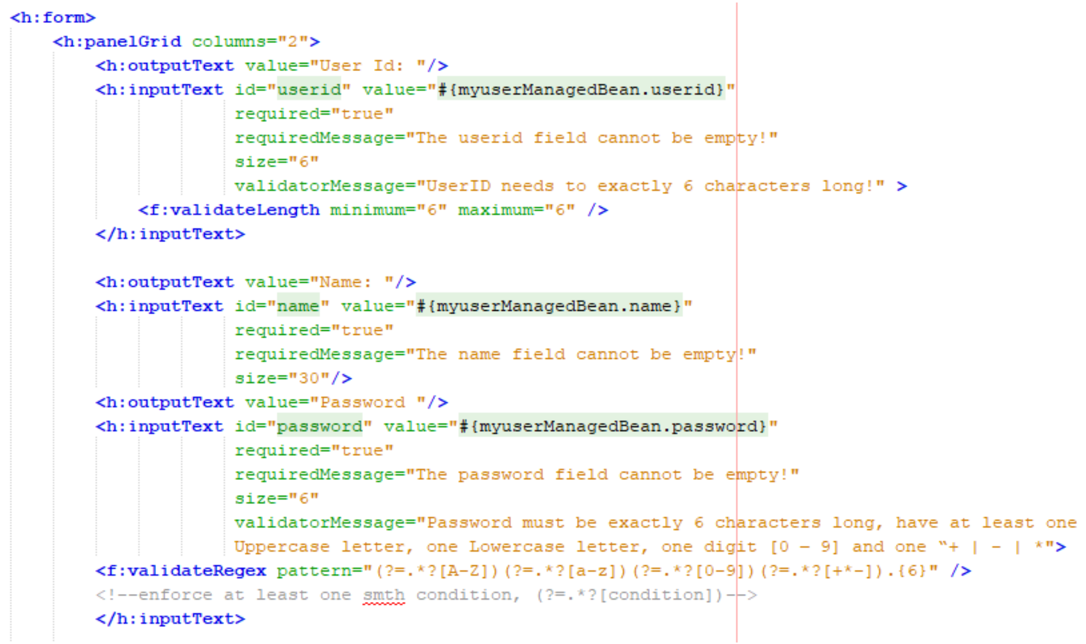
Name: S M Ragib Rezwan
ID: 103172423
Tutor: Wei Lai

Task1:

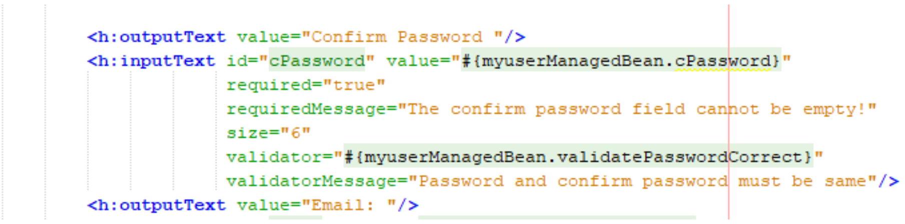
For a and b I have used standard validator tag as these could be simply done by using in built validators (like length limit and pattern regex)



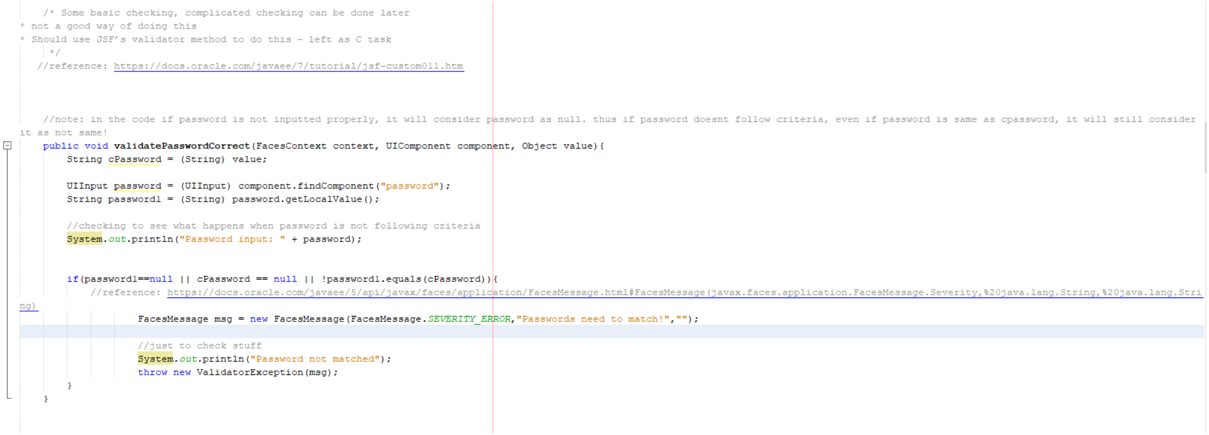
```
<h:form>
    <h:panelGrid columns="2">
        <h:outputText value="User Id: "/>
        <h:inputText id="userid" value="#{myuserManagedBean.userid}"
                     required="true"
                     requiredMessage="The userid field cannot be empty!"
                     size="6"
                     validatorMessage="UserID needs to exactly 6 characters long!" >
            <f:validateLength minimum="6" maximum="6" />
        </h:inputText>

        <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"
                     validatorMessage="Password must be exactly 6 characters long, have at least one
                     Uppercase letter, one Lowercase letter, one digit [0 – 9] and one "+ | - | *">
        <f:validateRegex pattern="(?=.*?[A-Z])(?=.*?[a-z])(?=.*?[0-9])(?=.*?[+*-]).{6}" />
        <!--enforce at least one smth condition, (?=.*?[condition])-->
        </h:inputText>
```

But for c, I had to use a user defined validation method in in the myusermanagedbean class. That’s because here we need to compare two strings and thus it is bestter to do this by coding it customly instead of relying on their inbuilt standard validation methods.

(note I have given reference to websites that have helped me just above the lines of codes)



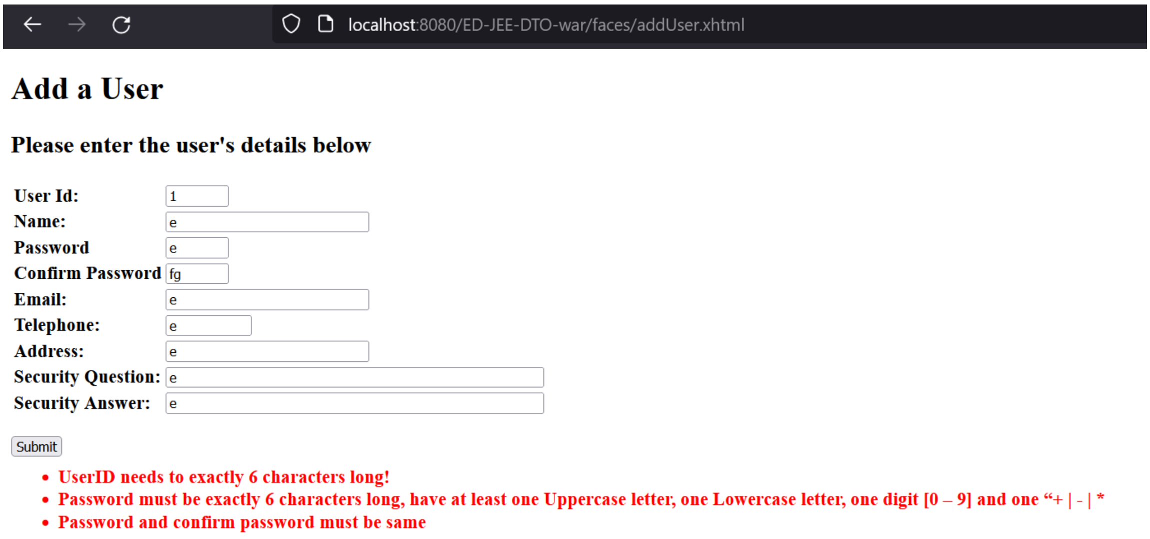
```
        <h:outputText value="Confirm Password "/>
        <h:inputText id="cPassword" value="#{myuserManagedBean.cPassword}"
                     required="true"
                     requiredMessage="The confirm password field cannot be empty!"
                     size="6"
                     validator="#{myuserManagedBean.validatePasswordCorrect}"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



```
    /* 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
     */
   //reference: https://docs.oracle.com/javaee/7/tutorial/jsf-custom011.htm



    //note: in the code if password is not inputted properly, it will consider password as null. thus if password doesnt follow criteria, even if password is same as cpassword, it will still consider
it as not same!
    public void validatePasswordCorrect(FacesContext context, UIComponent component, Object value){
        String cPassword = (String) value;

        UIInput password = (UIInput) component.findComponent("password");
        String password1 = (String) password.getLocalValue();

        //checking to see what happens when password is not following criteria
        System.out.println("Password input: " + password);


        if(password1==null || cPassword == null || !password1.equals(cPassword)){
            //reference: https://docs.oracle.com/javaee/5/api/javax/faces/application/FacesMessage.html#FacesMessage(javax.faces.application.FacesMessage.Severity,%20java.lang.String,%20java.lang.Stri
ng)
                FacesMessage msg = new FacesMessage(FacesMessage.SEVERITY_ERROR,"Passwords need to match!","");

                //just to check stuff
                System.out.println("Password not matched");
                throw new ValidatorException(msg);
        }
    }
```

Name: S M Ragib Rezwan
ID: 103172423
Tutor: Wei Lai

Code outputs:

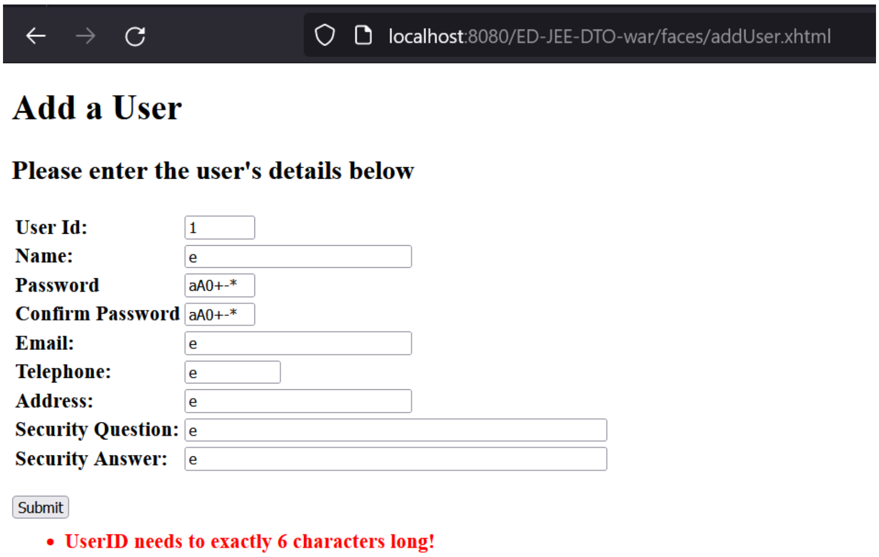


Add a User

Please enter the user's details below

User Id:	1
Name:	e
Password	e
Confirm Password	fg
Email:	e
Telephone:	e
Address:	e
Security Question:	e
Security Answer:	e

- UserID needs to exactly 6 characters long!
- Password must be exactly 6 characters long, have at least one Uppercase letter, one Lowercase letter, one digit [0 – 9] and one “+ | - | *
- Password and confirm password must be same



Add a User

Please enter the user's details below

User Id:	1
Name:	e
Password	aA0+-*
Confirm Password	aA0+-*
Email:	e
Telephone:	e
Address:	e
Security Question:	e
Security Answer:	e

- UserID needs to exactly 6 characters long!

Task4:

During this process, I had suffered a lot trying to overcome the problem with the email. Honestly, this is probably the first time for me where I could instantly finish a C level task but still remain stuck on that week’s P level task. In brief, the functionality of the software (take 5.1P) in sending an email to my gmail for promting of successful edit/ update had not worked at all. Instead errors like SSL Hello error, etc kept popping up due to either lack/ corruption of jar file (this was the conclusion that had been obtained after troubleshooting it with the course’s tutors). Thus it was quite irritated to say the least.

Name: S M Ragib Rezwan
ID: 103172423
Tutor: Wei Lai

But other than that, I did learn a lot in terms of how websites actually work and how to build up a smooth, efficient and scalable backend for the website itself. Furthermore, it has also allowed me to connect what I had learnt in my previous web application unit with OOP concepts, providing me access with a brand new and easier way to create a robust website (ie client tier, web tier, business tier, EIS tier and all other components and how they are linked)

Although I really liked the way hints were provided to help us search for and locate the two different ways of applying validation to forms, it did remind me of css and other aspects that websites normally have, which this website currently lacks completely. Thus, it made me feel a bit anxious thinking about all the different “internal server errors“ that I would have to face to add those parts , in order to make the websites presentation look more attractive (while ensuring those strange errors don’t pop up while trying to deploy or run it!)

In conclusion, although I have learnt lot of new things over these 5 weeks, I still have a long way to go and so I am currently looking forward to reaching that stage both with anticipation and worry.