**updateUser.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-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml"

xmlns:h="http://xmlns.jcp.org/jsf/html">

<h:body>

<h1>

Edit User

</h1>

<h2>

<p>Please Update the user's details below</p>

</h2>

<h3>

<h:form>

<h:outputText value="User ID "/>

<h:inputText id="userid" value="#{myuserManagedBean.userid}"

required="true"

requiredMessage="The user field cannot be empty!" size="30" readonly ="true" />

<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.updateUser}" />

</h:form>

</h3>

</h:body>

</html>

**updateUserFailure.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-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml"

xmlns:h="http://xmlns.jcp.org/jsf/html">

<h:head>

<title>Edit User Details</title>

</h:head>

<h:body>

<p>User update - Failure</p>

<p>Error updating this user ID</p>

<h:form>

<h:outputText value="User ID: "/><h:outputText value="#{myuserManagedBean.userid}"/><p/>

<h:commandButton value="Try Again" action="updateUser.xhtml"/>

</h:form>

</h:body>

</html>

**updateUserSuccess.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-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml"

xmlns:h="http://xmlns.jcp.org/jsf/html">

<h:head>

<title>Edit User Details</title>

</h:head>

<h:body>

<p>User Update - Success</p>

<h:outputText value="#{myuserManagedBean.name}"/>, your details have been updated!

<h:form>

<h:panelGrid columns="2">

<h:outputText value="User Id: "/><h:outputText value="#{myuserManagedBean.userid}"/>

<h:outputText value="Name: "/><h:outputText value="#{myuserManagedBean.name}"/>

<h:outputText value="Password "/><h:outputText value="#{myuserManagedBean.password}"/>

<h:outputText value="Confirm Password "/><h:outputText value="#{myuserManagedBean.cPassword}"/>

<h:outputText value="Email: "/><h:outputText value="#{myuserManagedBean.email}"/>

<h:outputText value="Telephone: "/><h:outputText value="#{myuserManagedBean.phone}"/>

<h:outputText value="Address: "/><h:outputText value="#{myuserManagedBean.address}"/>

<h:outputText value="Security Question: "/><h:outputText value="#{myuserManagedBean.secQn}"/>

<h:outputText value="Security Answer: "/><h:outputText value="#{myuserManagedBean.secAns}"/>

</h:panelGrid>

<p></p>

<h:commandButton id="return" value="Return" action="mainmenu.xhtml"/>

</h:form>

<h:outputText value="Confirmation email sent, please check your email: #{myuserManagedBean.email}."/>

</h:body>

</html>

**checkUser.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-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml"

xmlns:h="http://xmlns.jcp.org/jsf/html">

<h:body>

<h1>

Check User

</h1>

<h2>

<p>Please enter the user's id</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:panelGrid> <p></p>

<h:commandButton id="check" value="Submit" action="#{myuserManagedBean.getRecord}" />

</h:form>

</h3>

</h:body>

</html>

**mainmenu.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-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml"

xmlns:h="http://xmlns.jcp.org/jsf/html">

<h:head>

<title>Main Menu</title>

</h:head>

<h:body>

<h:form>

<h1>

<h:outputText value="Welcome to Myuser WebApplication!"/>

</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="checkUser.xhtml">Edit a user</a></li>

<li><a href="undercon.html">Delete a user</a></li>

</ol>

</h3>

</h:form>

</h:body>

</html>

**Faces-config.xml**

<navigation-rule>

<from-view-id>/checkUser.xhtml</from-view-id>

<navigation-case>

<from-action>#{myuserManagedBean.getRecord}</from-action>

<from-outcome>success</from-outcome>

<to-view-id>/updateUser.xhtml</to-view-id>

</navigation-case>

</navigation-rule>

<navigation-rule>

<from-view-id>/updateUser.xhtml</from-view-id>

<navigation-case>

<from-action>#{myuserManagedBean.updateUser}</from-action>

<from-outcome>success</from-outcome>

<to-view-id>/updateUserSuccess.xhtml</to-view-id>

</navigation-case>

<navigation-case>

<from-action>#{myuserManagedBean.updateUser}</from-action>

<from-outcome>failure</from-outcome>

<to-view-id>/updateUserFailure.xhtml</to-view-id>

</navigation-case>

</navigation-rule>

**MailCheck.java**

package web;

import java.util.Date;

import java.util.Properties;

import javax.mail.Message;

import javax.mail.Session;

import javax.mail.Transport;

import javax.mail.internet.InternetAddress;

import javax.mail.internet.MimeMessage;

public class MailCheck {

public static void sendMail(String recipient) {

String smtpServer = "smtp.gmail.com";

String from = "labtaskswinburne@gmail.com";

String to = recipient;

String subject = "Details Updated";

String body = "Hi, your details for this test website have been updated.\n"

+ "If this was not you, please contact us immediately at labtaskswinburne@gmail.com ...";

String emailUser = from;

String password = "labtask123";

try {

Properties props = System.getProperties();

props.put("mail.smtp.host", smtpServer);

props.put("mail.smtp.port", 587);

props.put("mail.smtp.auth", true);

props.put("mail.smtp.starttls.enable", true);

MyAuthenticator myPA = new MyAuthenticator(emailUser, password);

Session session = Session.getInstance(props, myPA);

Message msg = new MimeMessage(session);

msg.setFrom(new InternetAddress(from));

msg.setRecipients(Message.RecipientType.TO, InternetAddress.parse(to, false));

msg.setSubject(subject);

msg.setText(body);

msg.setHeader("X-Mailer", "Gmail");

msg.setSentDate(new Date());

Transport.send(msg, emailUser, password);

System.out.println("Message sent OK.");

} catch (Exception ex) {

ex.printStackTrace();

}

}

}

**MyAuthenticator.java**

package web;

import javax.mail.Authenticator;

import javax.mail.PasswordAuthentication;

public class MyAuthenticator extends Authenticator {

PasswordAuthentication mypa;

public MyAuthenticator(String username, String password) {

mypa = new PasswordAuthentication(username, password);

}

@Override

public PasswordAuthentication getPasswordAuthentication() {

return mypa;

}

}

**MyuserManagedBean.java**

public String getRecord() {

String result = "failure";

if (isValidUserid(userid)) {

MyuserDTO myuserDTO = myuserFacade.getRecord(userid);

if (myuserDTO != null) {

setUserid(myuserDTO.getUserid());

setAddress(myuserDTO.getAddress());

setEmail(myuserDTO.getEmail());

setName(myuserDTO.getName());

setPassword(myuserDTO.getPassword());

setPhone(myuserDTO.getPhone());

setSecAns(myuserDTO.getSecAns());

setSecQn(myuserDTO.getSecQn());

result = "success";

}

}

return result;

}

public String addUser() {

String result = "failure";

System.out.println("Add User Password: " + password);

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;

}

public String updateUser() {

String result = "failure";

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.updateRecord(myuserDTO)) {

result = "success";

}

System.out.println("Result: " + result);

}

if ("success".equals(result)) {

MailCheck.sendMail(getOldEmail());

}

return result;

}

A picture containing bird, flower, tree

Description automatically generatedA screenshot of a cell phone

Description automatically generatedA picture containing bird, tree, flower

Description automatically generatedA screenshot of a cell phone

Description automatically generatedA screenshot of a cell phone

Description automatically generated

A picture containing bird, tree

Description automatically generated

**4.1**

|  |  |  |
| --- | --- | --- |
| Client Tier   |  | | --- | | ED-JEE-DTO Application | | -MyuserAppClient | |
| Web Tier   |  | | --- | | ED-JEE-DTO War | | -JSF Pages  -MyuserManagedBean | |
| Business Tier   |  | | --- | | ED-JEE-DTO-ejb | | -MyuserFacade (Session Bean) |  |  | | --- | | ED-JEE-DTO-ejb | | -MyuserFacade (Session Bean) | |
| EIS Tier   |  | | --- | | sun-appserv-examples | | -Myuser database |  |  | | --- | | sun-appserv-examples | | -Myuser database | |

**4.2** Client tier component runs on the client machine. For this instance, the web clients contain dynamic web pages from HTML and XML generated by web components running on the web tier.

Web tier component runs on the Jave EE server. This web tier consists of web pages created using JSF.

Business tier component runs on the Java EE component. Business code/ logic receives data from client, processes it and sends it to the enterprise information system tier for storage, vice versa.

Enterise information system (EIS) runs on the EIS server tier handles EIS software and includes enterprise infrastructure systems.

**4.3** When designing this web page, I chose to follow the web design that previously implemented in previous labs. Add User and Edit User will be primarily implemented, with the others being under construction. By assigning each option an individual function and information, I can freely implement, design or upgrade one at a time that strictly follow OOP and UCD.

**5.1** If user chooses to edit their information, an old email will be used to receive confirmation notice. The old email codes will be implemented in MyuserManagedBean.java. When user chooses to go to the Edit User page, old email information will be retrieved and called in updateUser.java and updateUserSuccess.java. This action is implemented because as technology grows, credit stuffing attack (ATO) also grows. In an event of the account being hacked and a new email is updated to the user’s account, that email could be from the hacker. If the confirmation notice is sent to the new email, the hacker will then get access to the user’s account along with all credits, credentials, etc. Therefore, it should be sent to the old email that the user first created the account with. However, this process can be quite troublesome and not easily considered implementing.

**5.2** In order for this to happen, business policy needs to be changed. User needs to be warned in prior that the initial email is used to send any notification whether it’s an update, edit, or delete of account. Additionally, user can opt for different verification methods such as via phone number, nominated contact, or a security PIN. However, based on the importance or size of the project, these options seem to be out-op-scope due to their complexity and use of human power (coding etc). User can sometimes forget their initial information, hence not able to receive verification notice. The “new email” method seems to be the most reasonable, however extra security should be added.