

Project Report

Product Name	Advanced Certificate in Web Development
Qualification Name (NICF)	NICF-Advanced Certificate in Infocomm Technology (Software & Applications)
Product Name	Web Development Foundations
Module Name (NICF)	NICF -Web Development Foundations

Student name		Assessor name	
Date issued	Completion date		Submitted on

Project title	Design, Develop, Implement & Document Community Portal Website
----------------------	---

Learner declaration	
I certify that the work submitted for this assignment is my own and research sources are fully acknowledged.	
Student signature:	Date:

Content

1. Project background	
2. Project Objectives	
3. Project Requirement Specification	
4. Task 1	
5. Task 2	
6. Task 3	
7. Task 4	
8. Task 5	
9. Task 6	
10. Task 7	
11. Task 8	
12. Task 9	
13. Task 10	
14. Task 11	

1. Project Background

We have been approached by ABC Jobs Pte Ltd to develop a community portal for software developer. This project will be continued through Module 3, Module 4, Module 5 and Capstone projects. For this module the scope is Designing, Developing, Implementing & Documenting a Spring Website. Project Scope is to design a Community Portal Similar to LinkedIn.com. Users will be able to register on the portal using the Registration Page. Users of the portal can search for other users using various parameters like First Name, Last Name, Company Name, City. Users will be able to see the Public Profile of the user after searching for it. Portal allow user to login, request forgotten password and Update their profile information Project scope in this module is for Spring Website development.

2. Project Objective

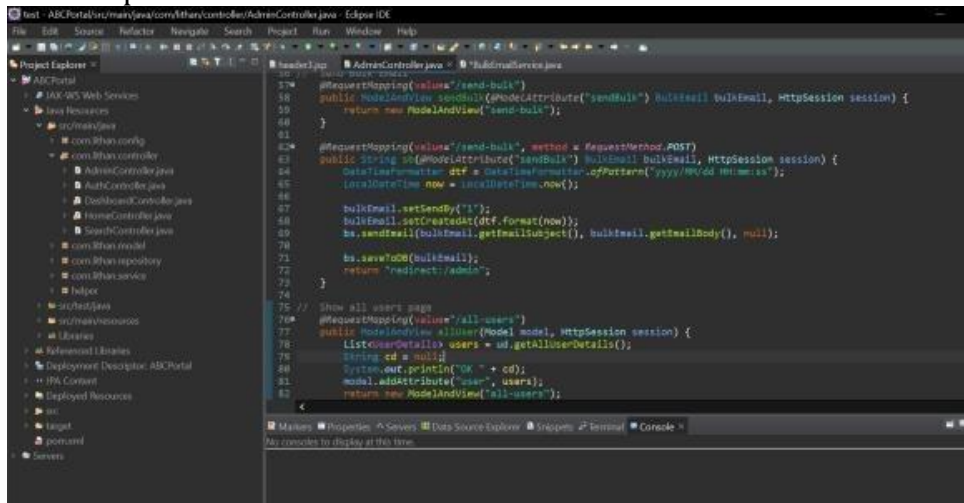
We need to design, develop, implement, test and document for ABC Jobs community Portal using spring framework.

- To develop and design the community portal website functional requirements using Spring MVC framework
- To design the webpages using JSP
- To implement a java-based Spring website using the Eclipse
- To document the develop website using a document generator, Javadoc

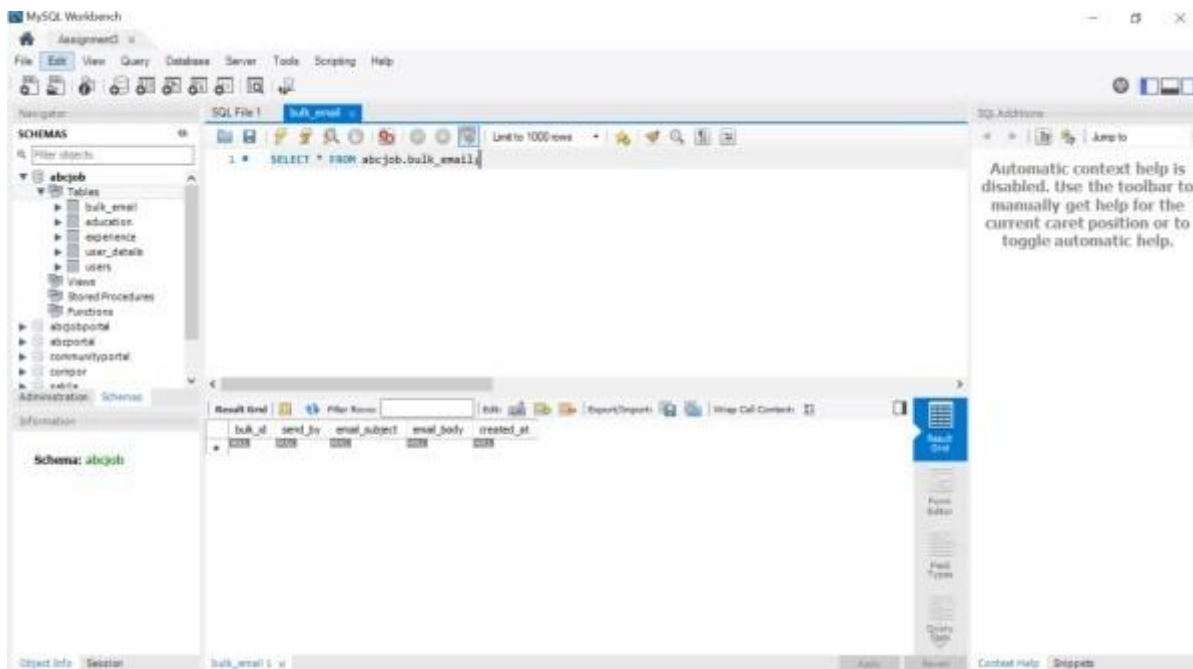
3. Project Requirements Specifications

< Screenshot of all the tools used in implementation and documentation>

1. Eclipse



2. SQL Workbench



3. Apache Tomcat 9.0

Programming > apache-tomcat-9.0.71 >				Search apache-tomcat-9.0.71	
selected folder in library				Give access to	
				Burn	
				New folder	
Name	Date modified	Type	Size		
bin	09/01/2023 22:33	File folder			
conf	09/01/2023 22:33	File folder			
lib	09/01/2023 22:33	File folder			
logs	09/01/2023 22:33	File folder			
temp	09/01/2023 22:33	File folder			
webapps	09/01/2023 22:33	File folder			
work	09/01/2023 22:33	File folder			
BUILDING.txt	09/01/2023 22:33	Text Document	21		
CONTRIBUTING.md	09/01/2023 22:33	Markdown Source...	7		
LICENSE	09/01/2023 22:33	File	57		
NOTICE	09/01/2023 22:33	File	3		
README.md	09/01/2023 22:33	Markdown Source...	4		
RELEASE-NOTES	09/01/2023 22:33	File	7		
RUNNING.txt	09/01/2023 22:33	Text Document	17		

4. Google Chrome



Google Chrome

App

5. Microsoft Word



Word

App

6. Microsoft PowerPoint Presentation



PowerPoint

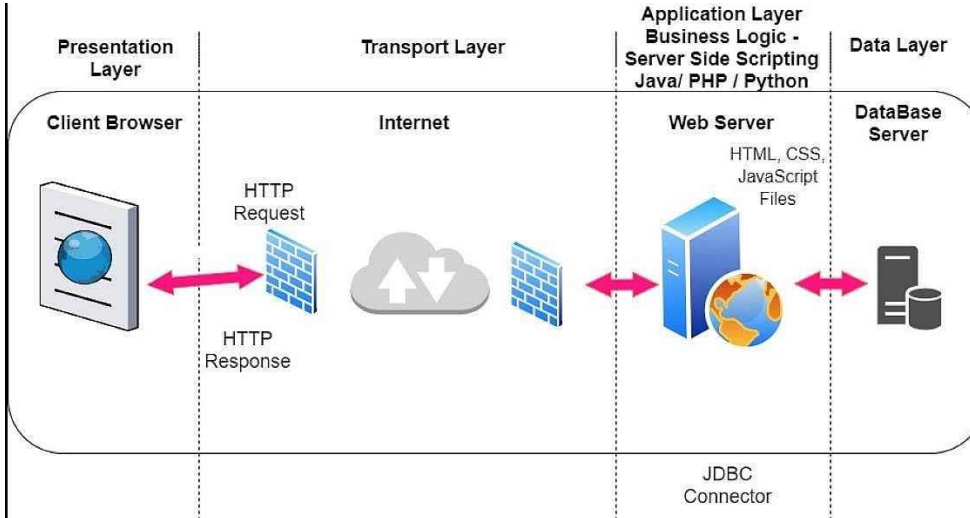
App

7. Etc

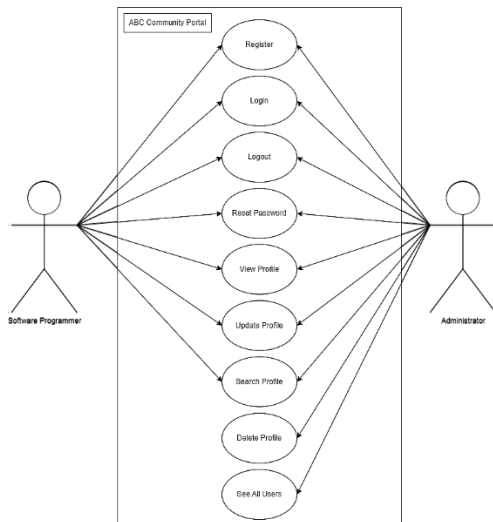
4. Task 1 Technical design

1. Task Statement (a): Block diagram of the system

Solution:



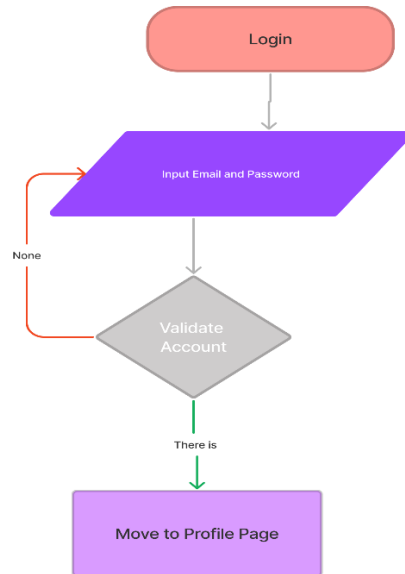
<Add Use Case diagram for describing overall requirements>>



<<Design Activity Diagram/use case/flowcharts for each scenario>>

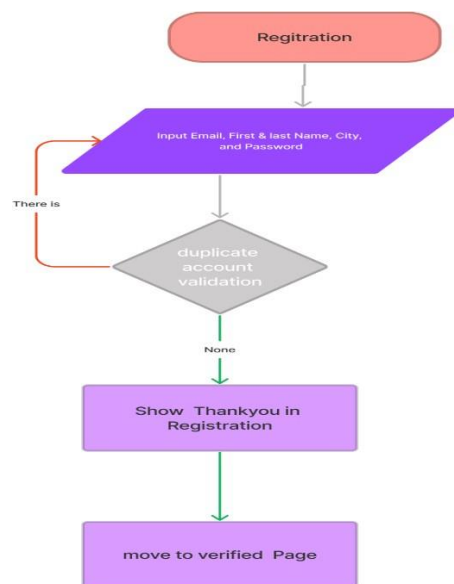
Flowchart	Steps
-----------	-------

Login



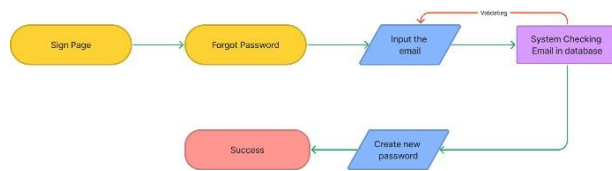
1. User goes to Sign in Page.
2. User enter the email and the password
3. The system will check the data after the user clicks a sign-in button if the valid user continues. If false or different user will re-enter the email and password again.
4. Next Move to Profile Page

Registration

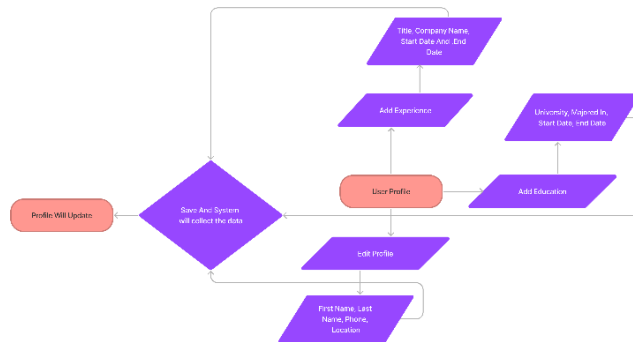


1. User goes to Sign Up Page.
2. User enters their first name,last name, city, email, and password, confirms the password, and clicks sign up.
3. The user will see an error state above the button if entries are invalid.
4. If valid, a verification will be sent on the next to Thankyou in registration

Forgot Password



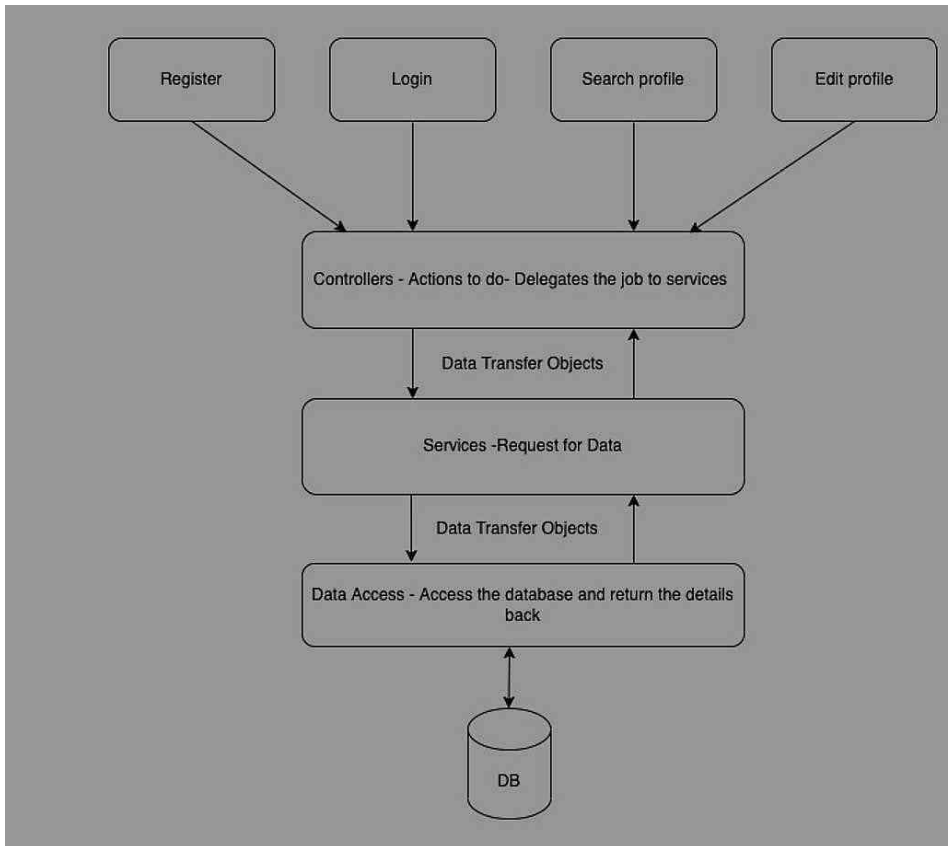
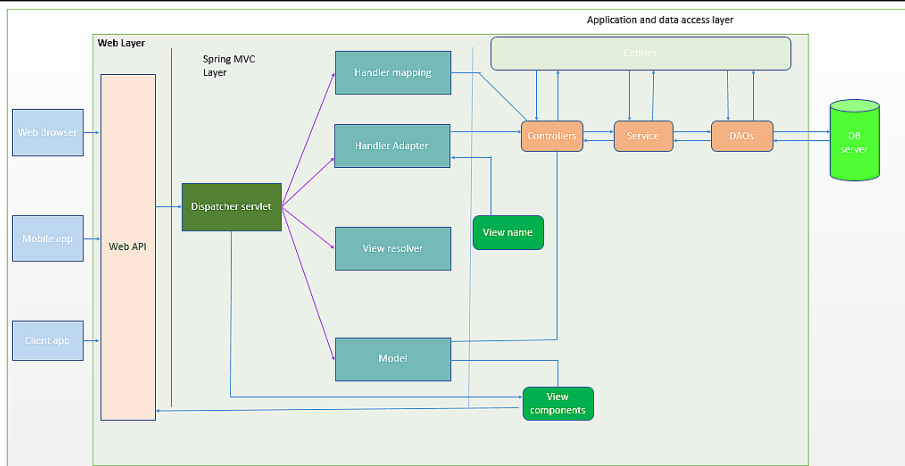
1. User at the sign-in page and click the forgot password
2. User enters the email that is used for sign in
3. If the email in database the process will continue, if not will show the alert
4. User enters a new password and confirms the password
5. When success directed to sign in page



1. Go to the user profile
2. Click edit profile
3. We can change the first name, last name, phone number, and city. User have another option for editing the education and experience
4. After we complete editing the profile, we can click save
5. System will collect and update the data

Task Statement (b): Architecture of the system

Solution:



<Explain the flow of the application clearly>

((Consider one scenario as an example and explain the request flow from controller->service->DAO->DB clearly))

Process

Description

--	--

	<p>After user already registered on the website and done the verification, User will direct into the login page and the system save the data user into database. Using</p>	
<pre>// login controller @RequestMapping(value="/login", method = RequestMethod.GET) public ModelAndView login(HttpSession session) throws Exception { return new ModelAndView("login"); }</pre>	<p>The first method is a GET method with a request mapping value of "/login". This method takes a HttpSession object as a parameter and returns a new ModelAndView object with a view name of "login". This method is likely used to display the login form.</p>	
<pre>@RequestMapping(value="/login", method = RequestMethod.POST) // login process public ModelAndView login(@ModelAttribute("login") Users user, Model model, HttpServletRequest req, HttpServletResponse res) { HttpSession session = req.getSession(); Users isLogin = us.login(user); if(isLogin != null) { session.setAttribute("email", isLogin.getEmail()); session.setAttribute("userId", isLogin.getUserId()); session.setAttribute("roleId", isLogin.getRoleId()); session.setAttribute("isLogin", true); return new ModelAndView("redirect:/profile"); } String msg = "Credentials is incorrect !"; model.addAttribute("message", msg); return new ModelAndView("login"); }</pre>	<p>The second method is a POST method with a request mapping value of "/login". This method takes several parameters, including a Users object that is annotated with "@ModelAttribute". This means that the method expects form data to be submitted with the request that can be used to create a new Users object. The method also takes a Model object, which can be used to add attributes to the view, and HttpServletRequest and HttpServletResponse objects for handling the request and response, respectively.</p>	
<pre>@ModelAttribute("login") Users user, Model model, HttpServletRequest req, HttpServletResponse res) {</pre>	<p>The method first gets the current session object using the HttpServletRequest object. This is done in order to store information about the current user in the session.</p>	
<pre>HttpSession session = req.getSession(); Users isLogin = us.login(user);</pre>	<p>The method then calls a method called "login" on a Users service object called "us" and passes in the Users object created from the form data. This method is likely used to authenticate the user based on their email and password.</p>	
<pre>if(isLogin != null) { session.setAttribute("email", isLogin.getEmail()); session.setAttribute("userId", isLogin.getUserId()); session.setAttribute("roleId", isLogin.getRoleId()); session.setAttribute("isLogin", true); return new ModelAndView("redirect:/profile"); }</pre>	<p>If the login is successful (the "isLogin" object is not null), the method sets several attributes on the session object, including the user's email, userId, roleId, and a boolean value indicating that the user is logged in. The method then returns a new ModelAndView object with a view name of "redirect:/Profile". This will redirect the user to the dashboard page.</p>	
<pre>String msg = "Credentials is incorrect !"; model.addAttribute("message", msg); return new ModelAndView("login");</pre>	<p>If the login is unsuccessful, the method sets a message indicating that the credentials are incorrect as an attribute on the model object. The method then returns a new ModelAndView object with a view name of "login". This will display the login form again with an error message indicating that the login was unsuccessful.</p>	

2. System

a. List down all the modules to be developed

Private Module - Administrator - Manage user Data and Bulk Emailing.

Public Module - Software Programmers –

- A. Registration
- B. Login
- C. Forgot Password
- D. Search and Search Result
- E. Update Profile

b. List down all the pages in each of the modules and briefly describe the purpose

The answer is on C

c. Note down which are Create, Edit & View pages.

Insert - Create Page for users to register and data are inserted into database

Select - View Page for the users can view the others profile page

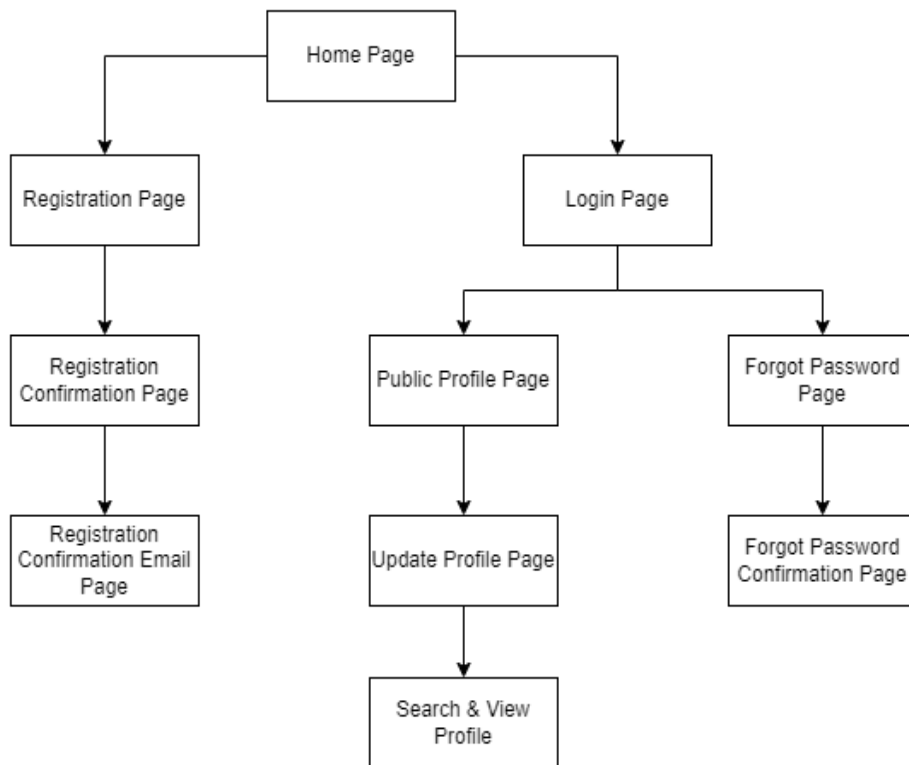
Update - Update Page for the user to edit and update their profile page

<you may answer this way>

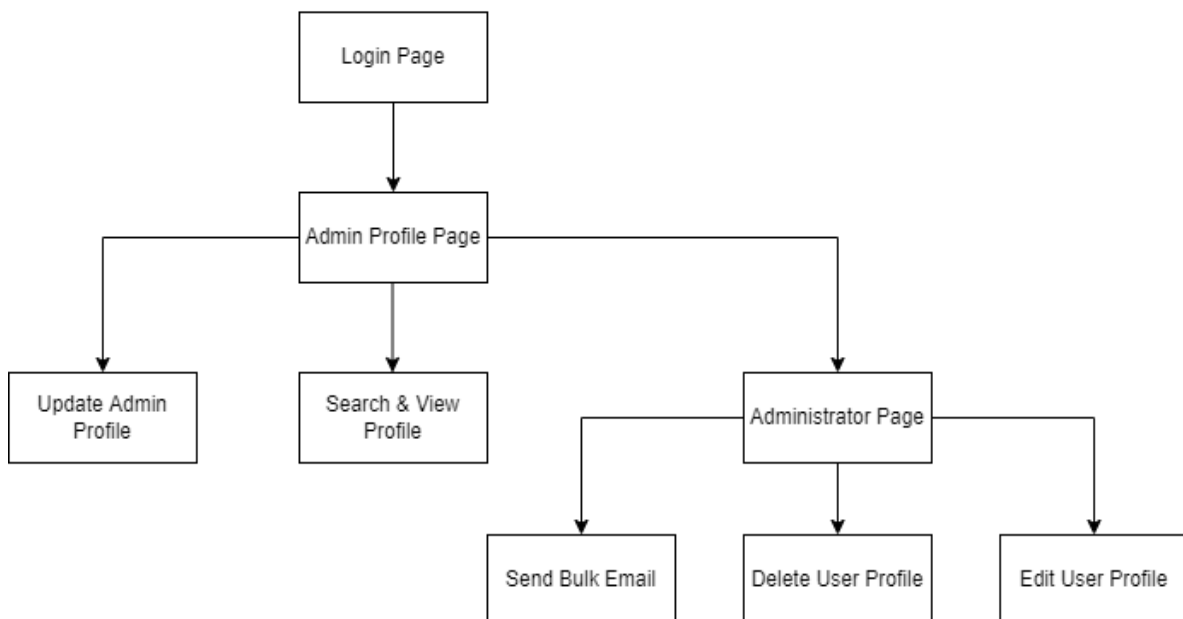
Pages (.jsp)	Brief description.	Operations.
Header1	Navbar, ABC Logo, button Dashboard, sign up, and sign in	View
HeaderNav	Navbar, ABC Logo Button Profile, Search, LogOut, Admin	View
Header3	Navbar, ABC Logo Button Profile, Search, LogOut, Admin	View
Index	Landing page	View
Registration	users can create a new account in the portal	Insert
Thankyou	Appreciate new user	View
Verified	Verifying the account	View
Login	Users can enter the portal by logging in with a username and password	Select
Forgotpass	Verifying the user registered email before the password can be changed	Select
Resetpass	Allows users to change password to new password	Update
Profile	Show user profile and data	View and Update
Search	Users can search other user	View and Select
Result	See a list of search results viewing other profiles	View and Select
Footer2	Copyright, company name	View
Footer3	Copyright, company name	View
Admin/index	A simple page for the admin to select options for managing users	View
All-user	View all the users and can view details, delete	View, Select, Update
Send-bulk	Send bulk email to user	Select

d. Create the Sitemap of Public & Private Site

Public Site



Private Site



3. Technical Environment Requirements

Hardware Requirements

- Database server – MySQL server
- Firewall settings - Port 9091 -- Http
- Port 3306 – Database

Software Requirements

Development environment-	JDK 11.0.3
Framework	Spring framework 5.2.10
Database server-	MySQL Server 8.0.22

4. System Integration Requirements

JDBC - JDBC Connector 8.0.22

5. Portability Requirements

Multi OS-	Windows, Linux, and Mac
Multiple Browsers -	Chrome, Safari
Multiple Devices -	Desktop, Tablet, Mobile

6. Maintainability Requirements

- Backup the database for every 6 hours.
- Vulnerability check – periodical scanning.
- Each request should be processed with 10 seconds.

7. Performance Requirements

Concurrent number of users- 100

Loading time – 15secs

8. Security Requirements

a. List down the Security Access for the different kind of users

Pages	Access based on User Role	
	Software Programmer	Administrator
Community portal home page	View	View
Registration page	View/Create	View/Create
Thankyou page	View	View
Verified email page	View	View
Login page	View/Login	View/Login
Forgot Password page	View	View
Forgot password confirmation page	View/Edit	View/Edit
Public Profile page	View	View
Update profile page	View/Edit	View/Edit
Bulk Email	N/A	View/Edit
Administration page	N/A	View

Functions	Access based on User Role	
	Software Programmer	Administrator
Register	Yes	Yes
Login	Yes	Yes
Change password	Yes	Yes
Update profile	Yes	Yes
Search others	Yes	Yes
View other user profiles	Yes	Yes
Update other user profiles	No	Yes
Delete other user profiles	No	Yes
Send bulk email	No	Yes

b. Briefly explain Login & Logout mechanism

When a registered user logs into the website by entering their username and password, the user is allowed access or authenticated to their user profile or home page.

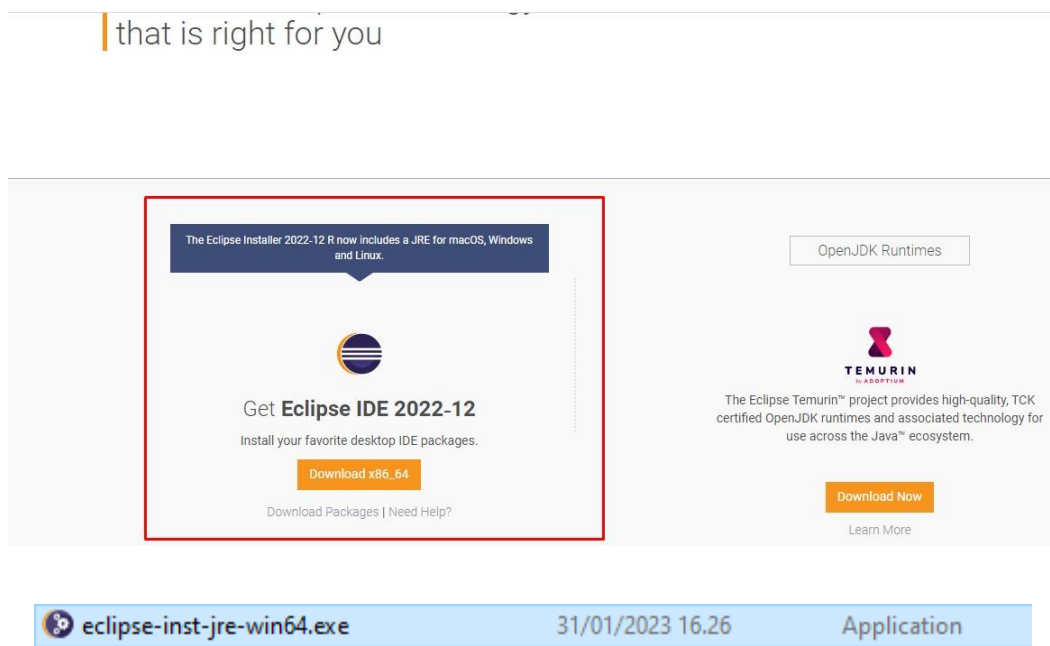
Logout is done when the user clicks on the logout button which redirects the user to the login page. Alternatively, the website can automatically log itself out if the user leaves the page. Logout helps prevent unauthorized users from accessing other people's accounts and is therefore an important part of security.

Task 2

Task Statement:

1. **Install Eclipse in your laptop**
2. Download eclipse on eclipse.org/downloads/
3. Open the download folder and double-click on the eclipse installer.
4. Choose the second option because we want to develop Web Development.
5. Last Thing is to click on install and wait for several moments. After that, you can run it on your desktop.

2. Provide Screen capture of eclipse IDE





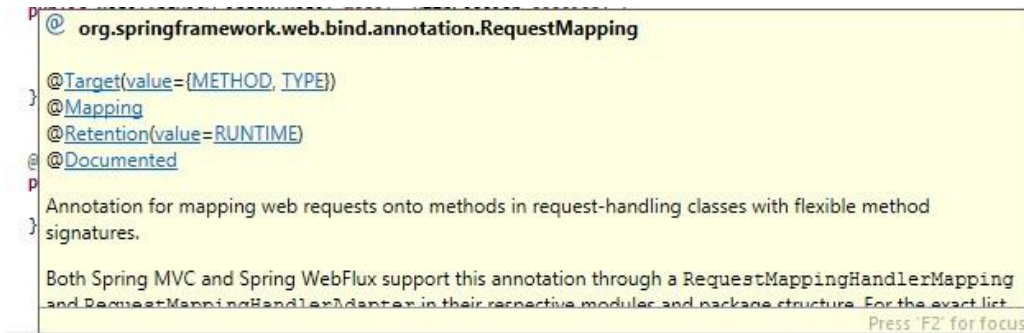
5. Task 3

Task Statement:

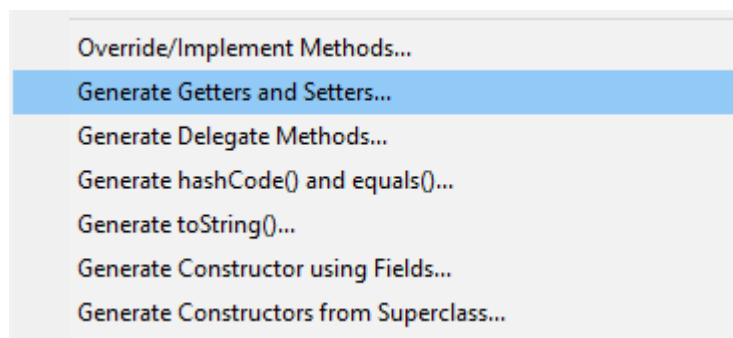
1. Provide description of 3 features in IDE along with the screen capture of their use.

Three IDE features are

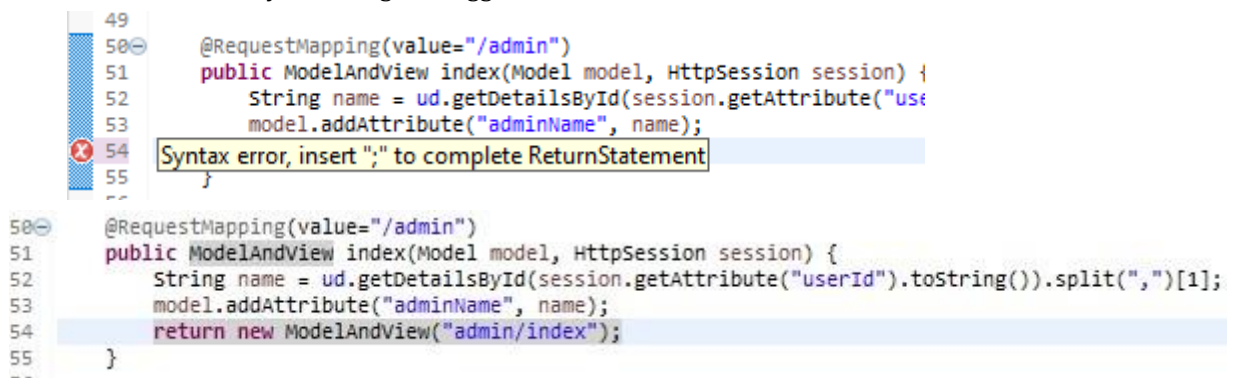
- a. Give Us Information when we point the cursor into a code that we dont understand



- b. Has automatic Source Generator



- c. Can fix the error by following the suggestion



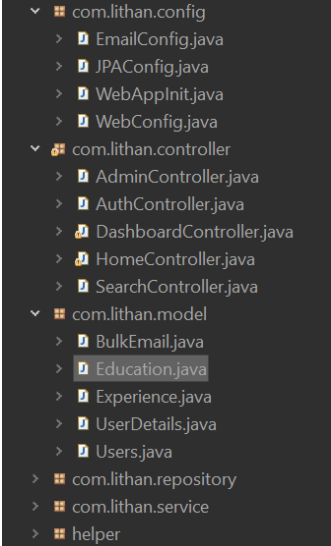
2. Include it as part of Project Presentation

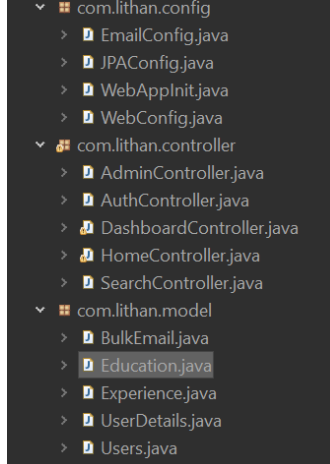
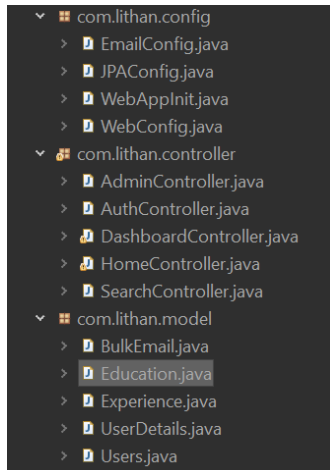
6. Task 4

Task Statement

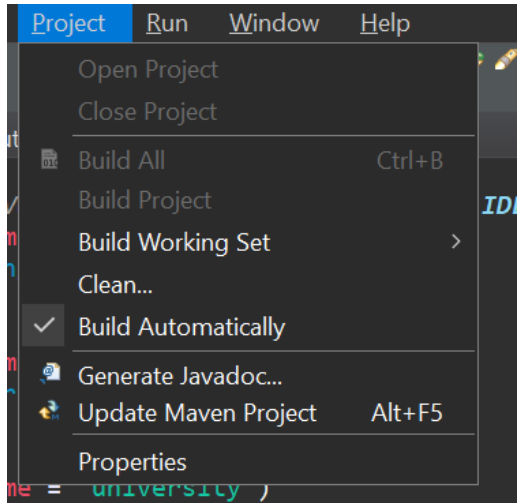
Create the following items in “Coding & Documentation Standards” Section in **Project Report**

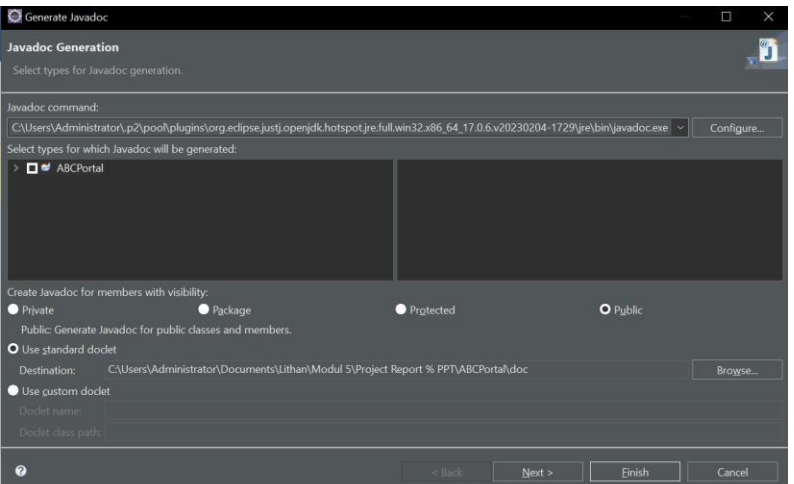
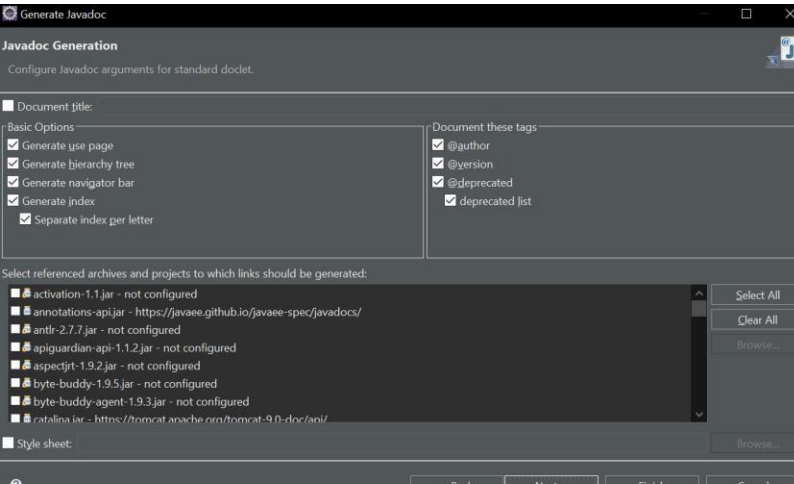
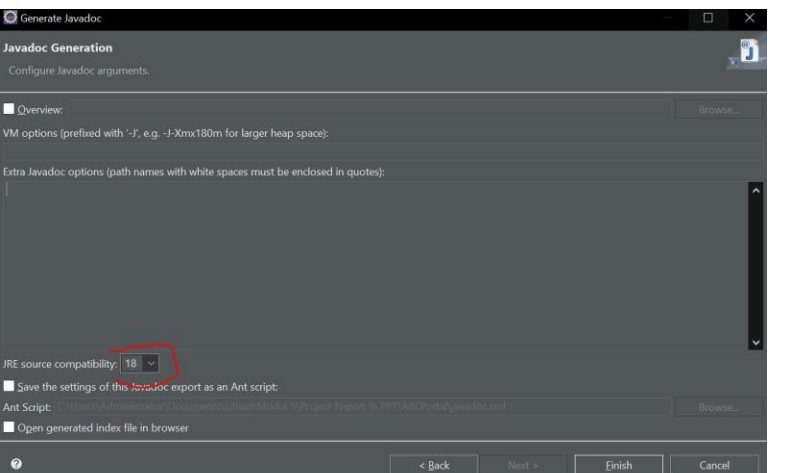
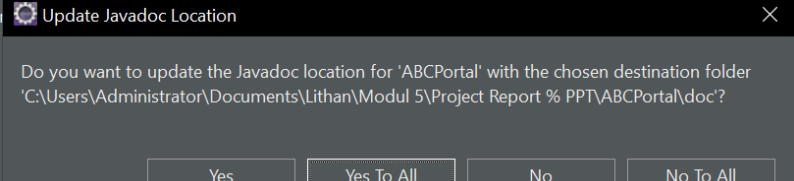
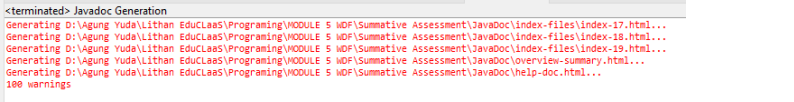
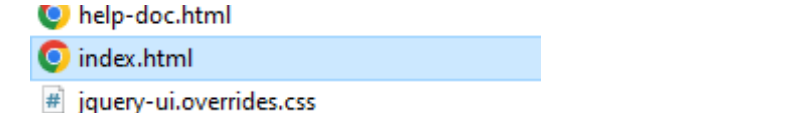
1. List 5 Coding Standards & Conventions to be used and explain shortly.


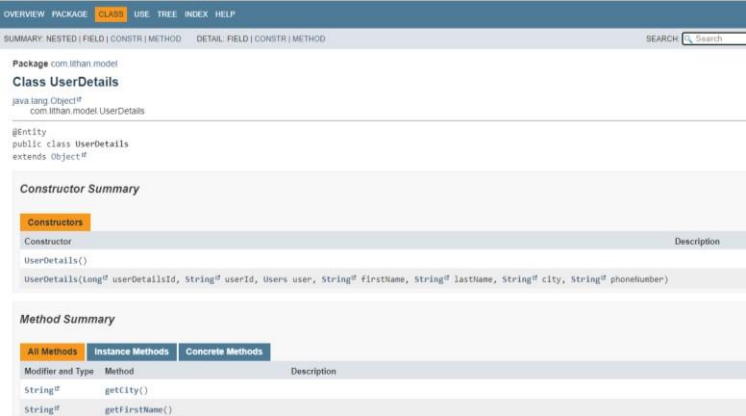
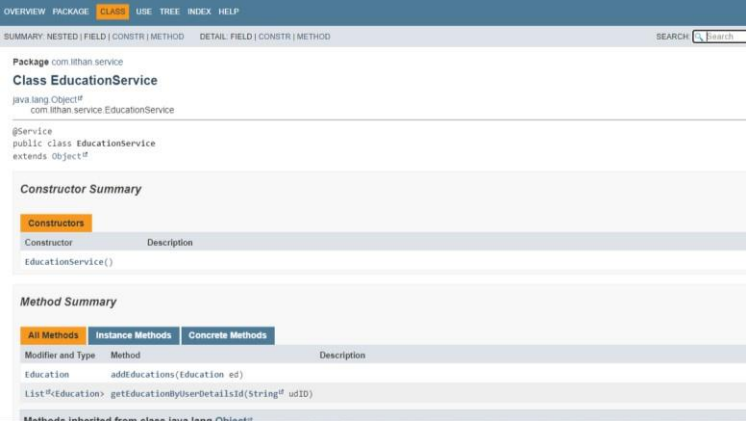
<pre>@Column(name = "user_details_id") private String userDetailsId; @Column(name = "university") private String university; @Column(name = "majored") private String majored; @Column(name = "ed_start_date") private String ed_start_date; @Column(name = "ed_end_date") private String ed_end_date;</pre>	Accessing class members privately Class fields should be inaccessible to protect the fields using a private access modifier. It is best practice to maintain encapsulation or hiding the data in preventing the information from being compromised.
<pre>// login controller @RequestMapping(value="/login", method = RequestMethod.GET) public ModelAndView login(HttpSession session) throws Exception { return new ModelAndView("login"); }</pre>	Appropriate commenting Commenting, such as a single line comment, ensures readability and describes how the code functions. This gives information that the code alone cannot perceive, allowing different people to comprehend the Java code.
	Organizing a Project The Spring MVC framework should be reflected in the project's structure by adding packages on the layer with corresponding MVC classes or interfaces. The controller, dao, service, and model layers of the project are shown in detail in a separate package.
	Class Declaration Class names should be nouns, in mixed case, with the first letter of each internal word capitalized.

	
	<p>Naming patterns</p> <p>Each class, interface, method, variable, and other name conventions should follow the correct naming conventions; for example, methods should use the PascalCase convention while variables should follow the camelCase style.</p>

2. Describe briefly documentation tool, you will be using and its configuration.

	<p>Click on Project and Generate Javadoc...</p>
---	---

	<p>Choose your project and the destination folder click next.</p>
	<p>Click next on this part.</p>
	<p>Choose JRE source compability based on your own.</p>
	<p>Click Yes to All.</p>
	<p>Wait until done</p>
	<p>Open Index.html in javadoc folder</p>
	<p>And you can see the result</p>

7. Task 5 (Views)

Task Statement

- Develop the following pages in HTML (Developed in Module 3) (source code)

- Home Page


```

1 <%@ page language="java" contentType="text/html; charset=UTF-8"
2   pageEncoding="UTF-8"%>
3
4 <%@ include file="header1.jsp" %>
5
6 <style>
7
8 body{
9   background-color: rgba(129, 212, 248, 0.692);
10 }
11   button {
12     background-color: rgba(255, 255, 255, 0.692);
13
14     border: 1px rgba(129, 212, 248, 0.692);
15     padding: 10px 40px;
16     text-align: center;
17     display: inline-block;
18     box-shadow: 0px 10px 10px rgba(0, 0, 0, 0.25);
19   }
20 </style>
21 <h1 style="margin-left: 20px;">Welcome to your <br> professional community</h1>
22 <i style="margin-left: 20px;">learn - make friends - find work</i>
23
24
25 <div class="p-5 d-flex m-5">
26
27   <div class="card-body">
28     <br><br><br><br><a href="registration"> <button style="margin-right: 90px;">SignUp Now!</button></a>
29     <a href="#about"> <button>Learn More</button></a>
30   </div>
31   <div class="card-body">
32     
33   </div></div>
34   <div class="container" style="background-color:smoke
35     ">
36
37 <h1 id="about"> About Us</h1>
38   <div class="p-5 d-flex m-5" >

```

B. Profile Page

```

1 <%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
2 <%@ include file="header3.jsp" %>
3 <style>body {
4   background-color: rgba(129, 212, 248, 0.692);
5 }</style>
6 <section class="profile pt-5 pb-5" style="background-color:rgba(129, 212, 248, 0.692);">
7
8   <div class="container">
9     <div class="row">
10      <div class="alert alert-success alert-dismissible fade show my-3 ${ message == null ? " d-none" : "d-block"}>
11        ${ message }
12        <button type="button" class="btn-close" data-bs-dismiss="alert" aria-label="Close"></button>
13      </div>
14
15      <div class="profile col-lg-12">
16        <div class="card">
17          <div class="img-fluid rounded-3 text-light d-flex flex-row">
18            style="height: 400px; background-image: url(img/Capture.PNG); background-repeat: no-repeat;
19            <div class="ms-4 mt-5 d-flex flex-column" style="width: 150px;">
20              
23            </div>
24          </div>
25          <div class="p-4 text-black bg-light rounded-3">
26            <div class="d-flex align-items-center justify-content-between">
27              <div class="myName" style="margin-left: 165px;">
28                <h2 class="fw-bold">${fullName}</h2>
29              </div>
30            </div>
31            <div>
32              <button class="btn btn-primary h5 rounded-pill px-5 py-2 " data-bs-toggle="modal"
33                data-bs-target="#editProfileModal">Edit
34                Profile</button>
35            </div>
36          </div>
37        </div>
38      </div>

```

C. Result Page

```

1 <%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
2 <jsp:include page="header3.jsp">
3   <jsp:param value="result.jsp" name="HTMLtitle" />
4 </jsp:include>
5
6 <div class="container">
7   <div class="alert alert-success alert-dismissible fade show ${message == null ? "d-none" : "d-block"} role="alert">
8     ${message}
9     <button type="button" class="btn-close" data-bs-dismiss="alert" aria-label="Close"></button>
10   </div>
11   <div class="profile col-lg-12">
12     <div class="row border rounded-3 p-3 mb-4">
13       <div>
14         class="col-2 border-none rounded-circle align-self-stretch text-center fs-1 d-flex align-items-center justify-content-center"
15         <span>${f}</span>
16         <span>${l}</span>
17       </div>
18       <div class="col-8 p-5">
19         <h2>${fullName}</h2>
20         <p>${city}</p>
21       </div>
22       <div class="col-2 align-self-center">
23         <button class="btn btn-success" id="follow">Follow</button>
24       </div>
25     </div>
26     <div class="col-lg-12 d-flex mt-5 bg-light rounded-3">
27       <div class="col-lg-5">
28         <div class="container">
29           <div class="row">
30             <h1>Status</h1>
31             <div class="col-lg-12">
32               <div class="card mb-4">
33                 <div class="card-body">
34                   <div class="row">
35                     <div class="col-sm-12">
36                       <div class="d-flex justify-content-between align-items-start">
37                         <h5 class="text-white bg-dark p-2" style="width: max-content"
38
39

```


D. Search Page

```

1 <% taglib prefix = "c" uri = "http://java.sun.com/jsp/jstl/core" %>
2 <jsp:include page="header3.jsp">
3   <jsp:param value="Search" name="HTMLtitle" />
4 </jsp:include>
5 <div class="container">
6   <form action="" method="get" class="mb-4">
7     <h1>Search Other</h1>
8     <input type="text" class="form-control" name="q" placeholder="Search Other" value="<%= request.getParameter("q") %">
9     <div id="searchHelp" class="form-text">Press enter to search</div>
10   </form>
11
12 <div>
13   <h1>${notFound == true ? "Not Found" : ""}</h1>
14   <c:forEach var="s" items="${selected}">
15     <div class="d-flex align-items-center border mb-3 rounded p-5 shadow-sm">
16       <div>
17         <h2>${s.getFirstName()} ${s.getLastName()}</h2>
18         <p>${s.getCity()}</p>
19       </div>
20       <form action="result" method="post" class="ms-auto">
21         <input type="hidden" name="uId" value="${s.getUserId()}">
22         <button type="submit" class="btn btn-outline-info ms-auto">View Profile</button>
23       </form>
24     </div>
25   </c:forEach>
26 </div>
27 </div>
28 <jsp:include page="footer3.jsp"></jsp:include>

```

E. Login Page

```

1 <jsp:include page="header.jsp">
2 <jsp:param value="Login" name="HTMLtitle" />
3 </jsp:include>
4
5 <style>
6 body{
7     background-color: rgba(129, 212, 248, 0.692);
8 }</style>
9 <div class="container text-center content-side shadow">
10 <div class="row d-flex align-items-center">
11 <div class="col-lg-7 p-3 mt-3">
12 
13
14 </div>
15 <div class="col-lg-5 p-3">
16 <h1 class="mb-2">Sign In</h1>
17 <div class="alert alert-danger alert-dismissible fade show my-3" ${ message == null ? "d-none" : "d-block"}>
18     ${ message }
19     <button type="button" class="btn-close" data-bs-dismiss="alert" aria-label="Close"></button>
20 </div>
21 <div class="alert alert-success alert-dismissible fade show my-3" ${ scMessage == null ? "d-none" : "d-block"}>
22     ${ scMessage }
23     <button type="button" class="btn-close" data-bs-dismiss="alert" aria-label="Close"></button>
24 </div>
25 <form action="login" method="post" class="needs-validation" novalidate>
26 <div class="form-group mt-3 px-3 py-1">
27 <input type="email" class="form-control" placeholder="Email Address" name="email" maxlength="50"
28     required>
29 <div class="invalid-feedback">
30     Email address should be have @ and .
31 </div>
32 </div>
33 <div class="form-group mt-3 px-3 py-1">
34 <input type="password" class="form-control" placeholder="Password" name="password"
35     maxlength="15" required>
36 <div class="invalid-feedback">
37     Password required & must be match
38 </div>
39 </div>

```

F. Bulk- Email Page

```
1 <%@ taglib prefix = "c" uri = "http://java.sun.com/jsp/jstl/core" %>
2 <%@ taglib prefix="form" uri="http://www.springframework.org/tags/form"%>
3 <jsp:include page="header3.jsp">
4     <jsp:param value="Send Bulk Email" name="HTMLtitle" />
5     <jsp:param value="css/style.css" name="isNested" />
6 </jsp:include>
7
8 <div class="container">
9     <form:form action="send-bulk" method="post" modelAttribute="sendBulk">
10         <div class="form-floating mb-3">
11             <input type="text" class="form-control" id="subject" name="emailSubject">
12             <label for="subject">Subject</label>
13         </div>
14         <div class="form-floating mb-3">
15             <textarea class="form-control" id="body" name="emailBody"></textarea>
16             <label for="body">Body</label>
17         </div>
18         <button type="submit" class="btn btn-primary">Send Message to All</button>
19     </form:form>
20 </div>
21
22 <jsp:include page="footer3.jsp"></jsp:include>
```

2. Template them by separating Headers/ Footers and including them using Server Side includes.
 - Header

```

1<!-- NO page language="en" contentType="text/html; charset=utf-8"
2pageEncoding="utf-8">
3<!--<?xml
4html lang="en"
5-->
6<head>
7<meta charset="UTF-8" />
8<meta http-equiv="X-UA-Compatible" content="IE=edge" />
9<meta name="viewport" content="width=device-width, initial-scale=1.0" />
10<title>ABC Job Community Portal</title>
11<link rel="stylesheet" href="/css/homepage.css" />
12<link rel="stylesheet" href="/css/reset.css" />
13<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-alpha1/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-qLpDa6qd1603aPMuqQq697m12137/35r66G6q10fw/mHdM"
14-->
15</head>
16<body>
17<div class="navbar navbar-expand-lg p-2 shadow mb-3" style="background-color: #f8d7da; border: 1px solid #f5c6cb;">
18<div class="container">
19<div class="btn-group navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-target="#navbarNav"
20aria-controls="navbarNav" data-bs-expanded="false" aria-label="Toggle navigation">
21<span class="navbar-toggler-icon"/>
22</div>
23<div class="collapse navbar-collapse justify-content-end" id="navbarNav">
24<li class="navbar-nav me-auto mb-2 mb-lg-0">
25<ul class="list-unstyled">
26<li class="btn btn-outline-primary me-4 my-2 rounded-pill px-5 p-5" href="#home" style="color: #000000;">Home</li>
27<li class="btn btn-outline-primary me-4 my-2 rounded-pill px-5 p-5" href="#login" style="color: #000000;">Login</li>
28<li class="btn btn-outline-primary me-4 my-2 rounded-pill px-5 p-5" href="#register" style="color: #000000;">Sign Up</li>
29</ul>
30</div>
31</div>
32</div>
33</body>
34</html>

```

[illegible]

- Footer

```

1 <header class="text-center at-3 pt-1">
2   <div style="background-color: rgb(129, 212, 248, 0.542);">
3     <div class="container">
4       <p>© 2023 ABC 100s Pte Ltd</p>
5     </div>
6   </header>
7
8 <script>
9   function validateForm() {
10     const password = document.getElementById('password').value;
11     const confpass = document.getElementById('confpass').value;
12     if (password != confpass) {
13       alert('Password Does Not Match!');
14       return false; // mencegah form untuk melakukan submit
15     } else {
16       return true; // Form akan melakukan submit jika password dan confpass sama
17     }
18   }
19 </script>
20
21 <script src="https://cdn.jsdelivr.net/npm/bootstrap@3.0-alpha/dist/js/bootstrap.bundle.min.js" integrity="sha384-w6Ad4Wl3x1hUyWNt98NADQtdmb3N3x08Y4QdKyIsg6Yb4gzI/qxZcwi6Tb63X" crossorigin="anonymous"></script>
22 </body>
23 </html>

```

```

10 <header class="text-center pt-3 pt-3 id=youish">
11 <div class="container">
12 <p>© 2023 ABC Jobs Etc Ltd/</p>
13 </div>
14 </header>
15
16 <script>
17 // Sample starter JavaScript for disabling form submissions if there are invalid fields
18 (() => {
19   'use strict'
20
21   // Fetch all the forms we want to apply custom Bootstrap validation styles to
22   const forms = document.querySelectorAll('.needs-validation')
23
24   // Loop over them and prevent submission
25   Array.from(forms).forEach(form => {
26     form.addEventListener('submit', event => {
27       if (!form.checkValidity()) {
28         event.preventDefault()
29         event.stopPropagation()
30       }
31       form.classList.add('was-validated')
32     }, false)
33   })
34 })()
35 </script>
36 <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-alpha1/dist/js/bootstrap.bundle.min.js" integrity="sha384-w76AqP9Wv3E1iqgZjZAGC+nL8vWZv3Nv3L3i60V86C10XupzKwMvC1Z6fBDKC" crossorigin="anonymous"></script>
37 </body>
38 </html>

```

- Include

```

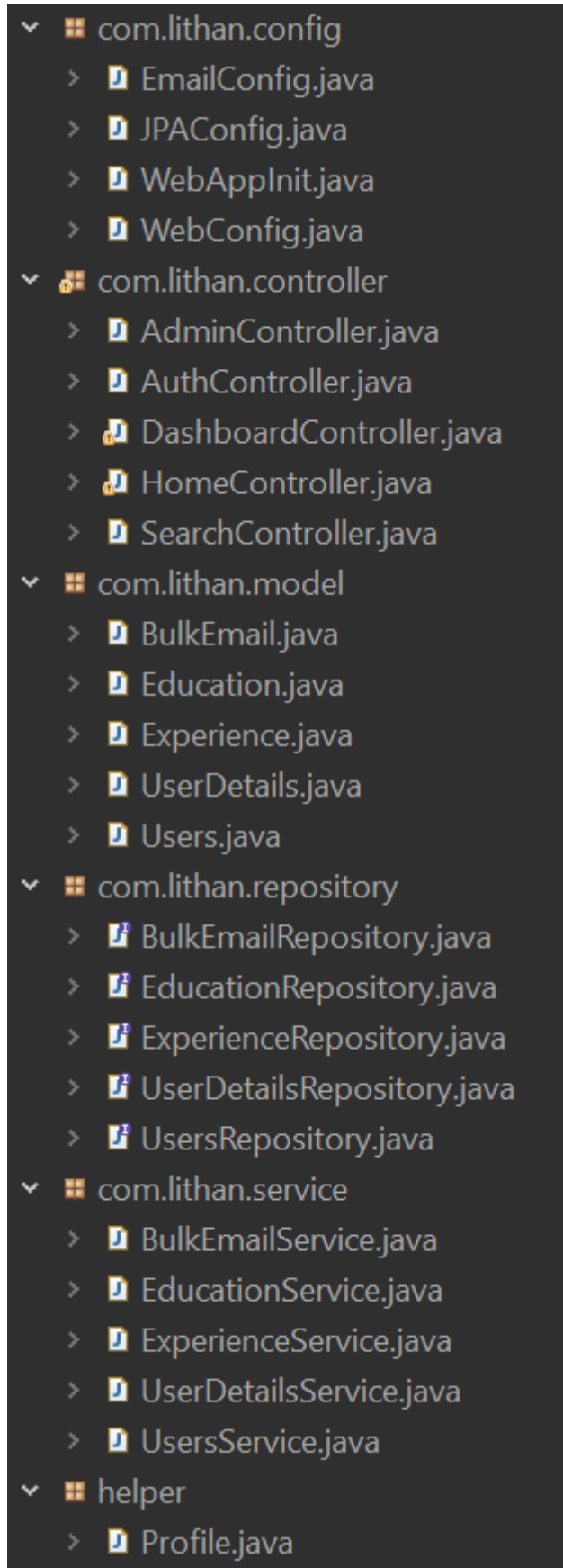
1 <%@ taglib prefix="form" uri="http://www.springframework.org/tags/form"%>
2 <%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
3 <%@ include file="headerNav.jsp" %>
4 <link rel="stylesheet" href="css/bootstrap.css" %>

```

3. Provide the screen capture in Project Presentation

8. Task 6 (java classes)

1. Package wise



2.

MainCotroller.java

Source code

Task Statement:

1. Create the required classes in Java

1.1 UsersDetails.java

```

1 package com.lithan.model;
2
3 import javax.persistence.Column;
4
5 @Entity
6 @Table(name = "user_details")
7 public class UserDetails {
8     @Id
9     @GeneratedValue(strategy = GenerationType.IDENTITY)
10    @Column(name = "user_details_id")
11    private Long userDetailsId;
12
13    @Column(name = "user_id")
14    private String userId;
15
16    @OneToOne(optional=false)
17    @JoinColumn(name = "user_id", referencedColumnName = "user_id", insertable=false, updatable=false)
18    private Users user;
19
20    @Column(name = "first_name")
21    private String firstName;
22
23    @Column(name = "last_name")
24    private String lastName;
25
26    @Column(name = "city")
27    private String city;
28
29    @Column(name = "phoneNumber")
30    private String phoneNumber;
31
32    public UserDetails() {}
33
34    public UserDetails(Long userDetailsId, String userId, Users user, String firstName, String lastName, String city, String phoneNumber) {
35        super();
36        this.userDetailsId = userDetailsId;
37        this.userId = userId;
38        this.user = user;
39        this.firstName = firstName;
40        this.lastName = lastName;
41        this.city = city;
42        this.phoneNumber = phoneNumber;
43    }
44
45    public Long getUserDetailsId() {
46        return userDetailsId;
47    }
48
49    public void setUserDetailsId(Long userDetailsId) {
50        this.userDetailsId = userDetailsId;
51    }
52
53    public String getUserId() {
54        return userId;
55    }
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71

```

1.2 UserDetailsService.java

```

1 package com.lithan.service;
2
3 import java.util.List;
4
5 @Service
6 @Transactional
7 public class UserDetailsService {
8     @Autowired
9     UserDetailsRepository repo;
10
11     public UserDetails register(UserDetails userDetails) {
12         return repo.save(userDetails);
13     }
14
15     public String getDetailsById(String userId) {
16         return repo.getDetailsById(userId);
17     }
18
19     public UserDetails editprofile(Long userDetailsId, UserDetails ud) {
20         UserDetails userDetails = repo.findById(userDetailsId).get();
21
22         // update
23         userDetails.setCity(ud.getCity());
24         userDetails.setFirstName(ud.getFirstName());
25         userDetails.setLastName(ud.getLastName());
26         userDetails.setPhoneNumber(ud.getPhoneNumber());
27
28         // save
29         return repo.save(userDetails);
30     }
31
32     public List<UserDetails> searchByKey(String key) {
33         return repo.searchByKey(key);
34     }
35
36     public List<UserDetails> getAllUserDetails() {
37         return repo.findAll();
38     }
39
40     public UserDetails getDetailsById(Long id) {
41         return repo.findById(id).get();
42     }
43 }
44
45
46
47
48
49
50
51
52
53

```

1.3 Users.java


```

1 package com.lithan.model;
2
3 import javax.persistence.Column;
4
5 @Entity
6 @Table(name = "users")
7 public class Users {
8
9     @Id
10    @GeneratedValue(strategy = GenerationType.IDENTITY)
11    @Column(name = "user_id")
12    private Long userId;
13
14    @Column(name = "role_id")
15    private String roleId;
16
17    @Column(name = "email")
18    private String email;
19
20    @Column(name = "password")
21    private String password;
22
23    @Column(name = "created_at")
24    private String createdAt;
25
26    public Users() {}
27
28    public Users(Long userId, String roleId, String email, String password, String createdAt) {
29        super();
30        this.userId = userId;
31        this.roleId = roleId;
32        this.email = email;
33        this.password = password;
34        this.createdAt = createdAt;
35    }
36
37    public Long getUserId() {
38        return userId;
39    }
40
41    public void setUserId(Long userId) {
42        this.userId = userId;
43    }
44
45    public String getRoleId() {
46        return roleId;
47    }
48
49    public void setRoleId(String roleId) {
50        this.roleId = roleId;
51    }
52
53    public String getEmail() {
54        return email;
55    }
56
57    }
58
59

```

2. Include the Data Access Layer classes

2.1 UsersRepository.java

```

1 package com.lithan.repository;
2
3 import org.springframework.data.repository.query.Param;
4
5 @Repository
6 public interface UsersRepository extends JpaRepository<Users, Long> {
7
8     @Query(value = "SELECT * FROM users ORDER BY user_id DESC LIMIT 1", nativeQuery = true)
9     String getLastUser();
10
11     @Query(value = "SELECT * FROM users WHERE :email = email AND :password = password", nativeQuery = true)
12     Users login(@Param("email") String email, @Param("password") String password);
13
14     @Query(value = "SELECT * FROM users WHERE email = :email", nativeQuery = true)
15     String checkEmail(@Param("email") String email);
16
17 }

```

2.2 UserDetailsRepository.java

```

1 package com.lithan.repository;
2
3 import java.util.List;
4
5 @Repository
6 public interface UserDetailsRepository extends JpaRepository<UserDetails, Long> {
7
8     @Query(value = "SELECT user_details_id, first_name, last_name, city, phoneNumber FROM user_details"
9         + " JOIN users ON user_details.user_id = users.user_id"
10         + " WHERE users.user_id = :userId", nativeQuery = true)
11     String getDetailsById(@Param("userId") String userId);
12
13     @Query(value = "SELECT * FROM user_details"
14         + " WHERE first_name LIKE %:key%"
15         + " OR last_name LIKE %:key%"
16         , nativeQuery = true)
17     public List<UserDetails> searchByKey(@Param("key") String key);
18
19 }

```

2.3 EducationRepository.java

```

1 package com.lithan.repository;
2
3 import java.util.List;
4
5 @Repository
6 public interface EducationRepository extends JpaRepository<Education, Long> {
7
8     @Query(value = "SELECT * FROM education WHERE user_details_id = :udID", nativeQuery = true)
9     public List<Education> getEducationByUserDetailsId(@Param("udID") String udID);
10
11 }

```

2.4 ExperienceRepository.java

```

1 package com.lithan.repository;
2
3 import java.util.List;
4
5
6
7
8
9
10
11
12 @Repository
13 public interface ExperienceRepository extends JpaRepository<Experience, Long> {
14
15     @Query(value = "SELECT * FROM experience WHERE user_details_id = :udID", nativeQuery = true)
16     public List<Experience> getExperienceByUserDetailsId(@Param("udID") String udID);
17
18 }

```

3. Include samples as part of Project Report

4. Task 7

(screen capture of your working website – web pages)

Task Statement:

Provide screen capture of developed pages and hosted application in Project Presentation

1. Use Queries, Procs Developed in Module 4

Registration	<pre> public Users register(Users user) { return repo.save(user); } </pre>
Viewprofile other user	<pre> @Query(value = "SELECT user_details_id, first_name, last_name, city, phoneNumber FROM user_details" + " JOIN users ON user_details.user_id = users.user_id" + " WHERE users.user_id = :userId", nativeQuery = true) </pre>
search	<pre> @Query(value = "SELECT * FROM user_details" + " WHERE first_name LIKE %:key%" + " OR last_name LIKE %:key%" + " OR city LIKE %:key%", nativeQuery = true) </pre>
Login	<pre> public Users login(Users user) { try { Users login = repo.login(user.getEmail(), user.getPassword()); return login; } catch (Exception e) { System.out.println("Credential is null " + e); } return null; } </pre>
Update Prorfile	<pre> public UserDetails editprofile(Long userDetailsId, UserDetails ud) { UserDetails userDetails = repo.findById(userDetailsId).get(); // update userDetails.setCity(ud.getCity()); userDetails.setFirstName(ud.getFirstName()); userDetails.setLastName(ud.getLastName()); userDetails.setPhoneNumber(ud.getPhoneNumber()); // save return repo.save(userDetails); } </pre>

Forget Password

```
public Users updatePassword(String password, String email) {  
    Long userId = Long.parseLong(repo.checkEmail(email).split(",")[0]);  
    Users user = repo.findById(userId).get();  
    // update  
    user.setPassword(password);  
  
    // save  
    return repo.save(user);  
}
```

Please refer to report- Task 1, 2. (c)

2. Develop the Models, Views, Controller (at least for one of the functionality, better if you can use for all the functionality)

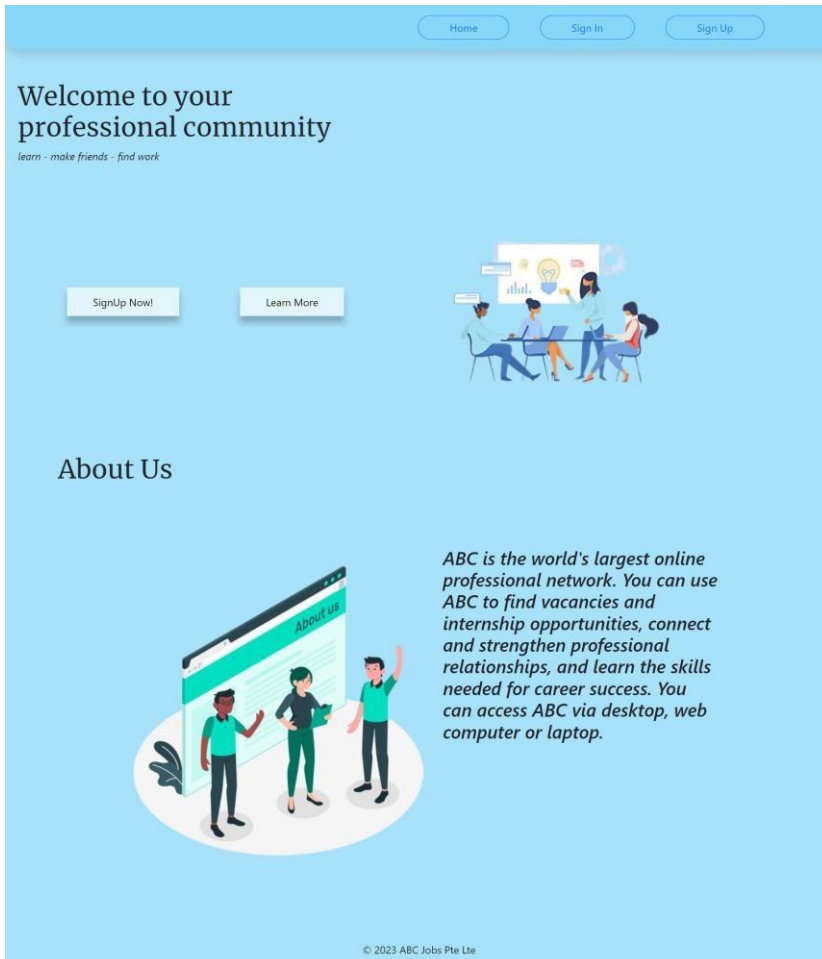
```
registration  
@RequestMapping(value="/registration")  
public ModelAndView registration(HttpSession session) throws Exception {  
    return new ModelAndView("registration");  
}  
  
@RequestMapping(value="/registration", method = RequestMethod.POST)  
public String registration(  
    @RequestParam("email") String email,  
    @RequestParam("password") String password,  
    @RequestParam("firstName") String firstName,  
    @RequestParam("lastName") String lastName,  
    @RequestParam("city") String city,  
    Users user, UserDetails userDetails, Model model) throws Exception {  
    try {  
        user.setEmail(email);  
        user.setPassword(password);  
        user.setRoleId("2");  
        user.setCreatedAt(java.time.LocalDate.now().toString());  
        us.register(user);  
  
        userDetails.setUserId(us.getUserId().split(",")[0]);  
        userDetails.setFirstName(firstName);  
        userDetails.setLastName(lastName);  
        userDetails.setCity(city);  
        ud.register(userDetails);  
  
        model.addAttribute("email", user.getEmail());  
        return "thankyou";  
    } catch (Exception e) {  
        System.out.println(e);  
    }  
    model.addAttribute("errMsg", "Email is currently in use!");  
    return "registration";  
}
```

The Java Spring Controller defined by this code manages user registration requests. The controller displays an empty registration form when a user reaches the "/registration" URL. The controller processes the data once the user accepts the registration form, creates a new user account, and stores the user data in a database. If the registration process is successful, a "thank you" page is displayed to the user. An error warning appears on the registration page if the user's email address has already been registered with the system.

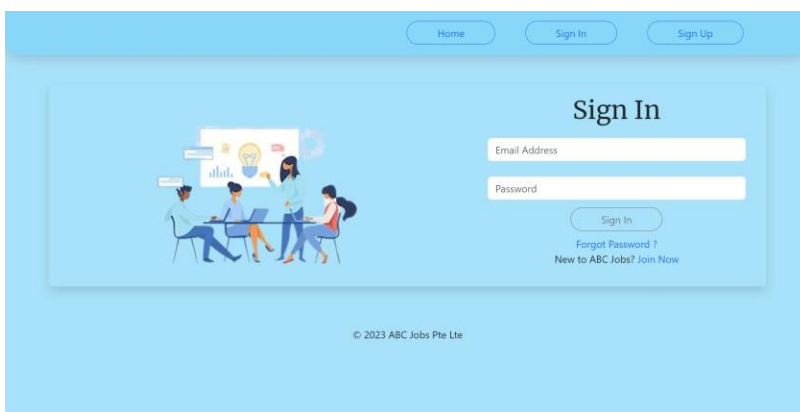
Two @RequestMapping methods are used in the code: one handles GET requests to display the registration form and the other handles POST requests to handle the form's input data.

3. Develop Home Page & Integrate with Login System

3.1 Home Page



3.2 Login



4. Develop Registration Form & Registration Thank You

4.1 Registration Form

HomeSign InSign Up

Sign Up

vyhe@mailinator.com

Mechelle

Coffey

Quo perferendis dolores sit qui quo asperiores cu

Agree and Join

Already on ABC Jobs? [Sign In](#)

© 2023 ABC Jobs Pte Ltd

4.2 Thank You

HomeSign InSign Up

Thank You

vyhe@mailinator.com

Your registration has been received and we appreciate your time.

Login

© 2023 ABC Jobs Pte Ltd

4.3 Verified Email

HomeSign InSign Up

Verified

Your account has been verified

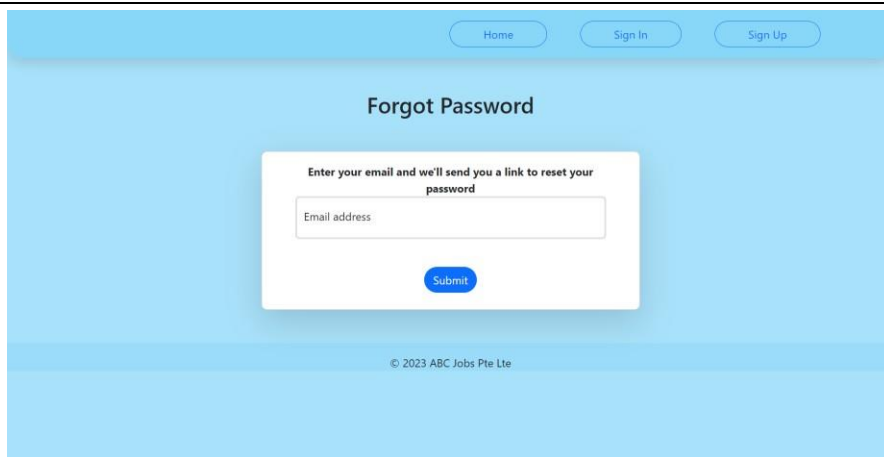
Please click below to Continue

Login

© 2023 ABC Jobs Pte Ltd

5. Develop Forget Password mechanism

5.1 Forget Password



Home Sign In Sign Up

Forgot Password

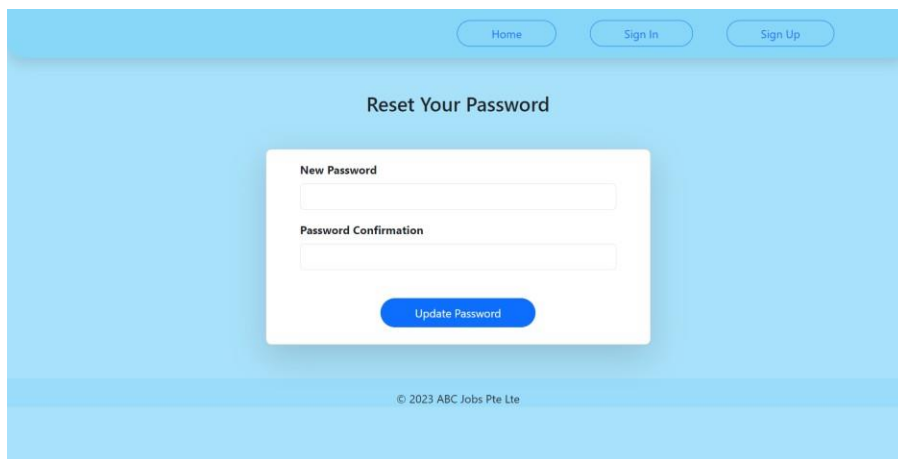
Enter your email and we'll send you a link to reset your password

Email address

Submit

© 2023 ABC Jobs Pte Ltd

5.2 Reset Password



Home Sign In Sign Up

Reset Your Password

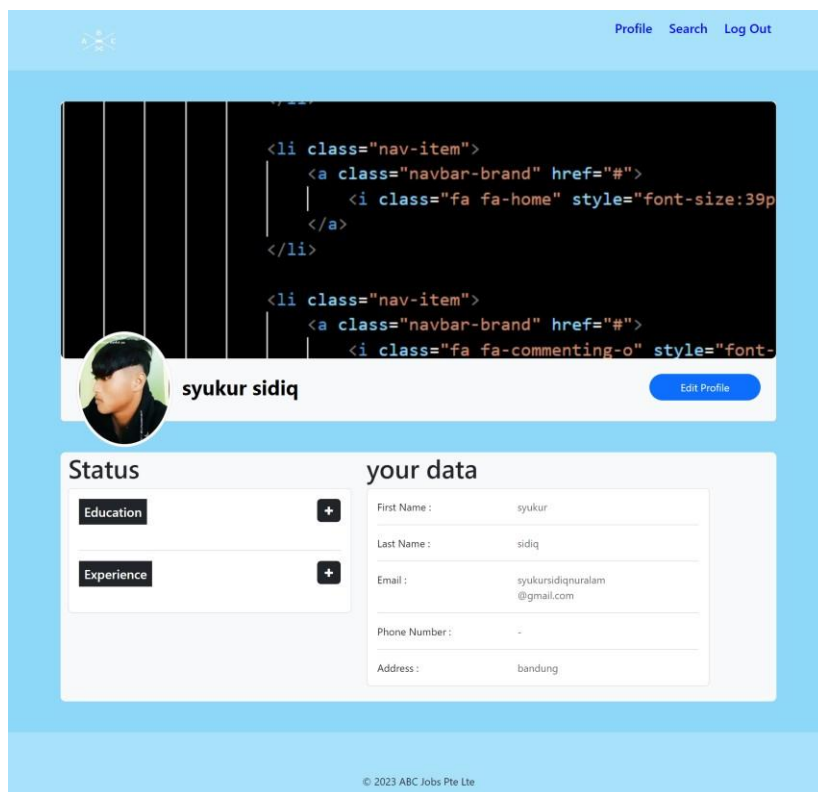
New Password

Password Confirmation

Update Password

© 2023 ABC Jobs Pte Ltd

6. Develop Landing Page post login



Profile Search Log Out

syukur sidiq Edit Profile

Status

Education +

Experience +

your data

First Name : syukur

Last Name : sidiq

Email : syukursidignur@gmail.com

Phone Number : -

Address : bandung

© 2023 ABC Jobs Pte Ltd

7. Develop Search Box for Users



Search Other

Press enter to search

8. Develop View Search Results



Search Other

Press enter to search

Nita Mcconnell

Nihil beatae nemo lo

[View Profile](#)

Quail Dixon

Id esse sit aliquam et et quae a numquam consecte

[View Profile](#)

Lynn Cannon

Nesciunt aliquam accusamus ea nostrum architecto

[View Profile](#)

9. Develop View Other Profile Page



Nita McConnell

Nihil beatae nemo lo

Follow

Status

Education

Experiences

your data

First Name : Nita

Last Name : McConnell

Email : sykursidqnuralam@gmail.com

Phone Number : +1 (205) 937-6921

Address : Nihil beatae nemo lo

10. Develop Edit Profile Functionality

Add Experience



Title

Company name

Start Date



End Date



Close

Save changes

Edit Profile



First Name

Last Name

Phone

Location

Close

Save changes

Add Education



University

Majored In

Start Date



End Date



Close

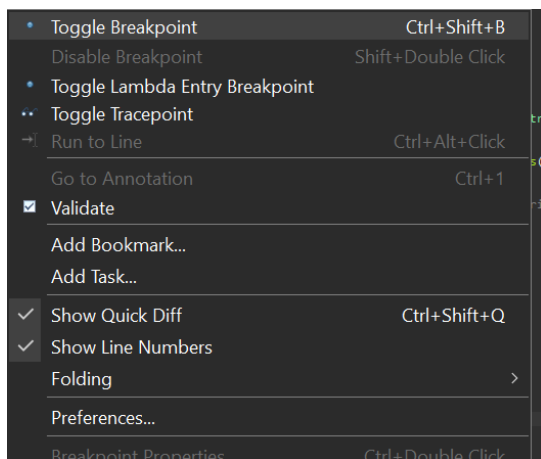
Save changes

8. Task 8

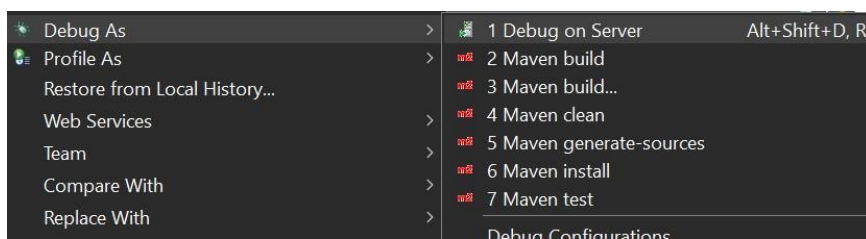
Task Statement:

1. Debug the application in Eclipse. Provide the screen capture of your application code in Eclipse Debug Mode include it as part of Project Presentation

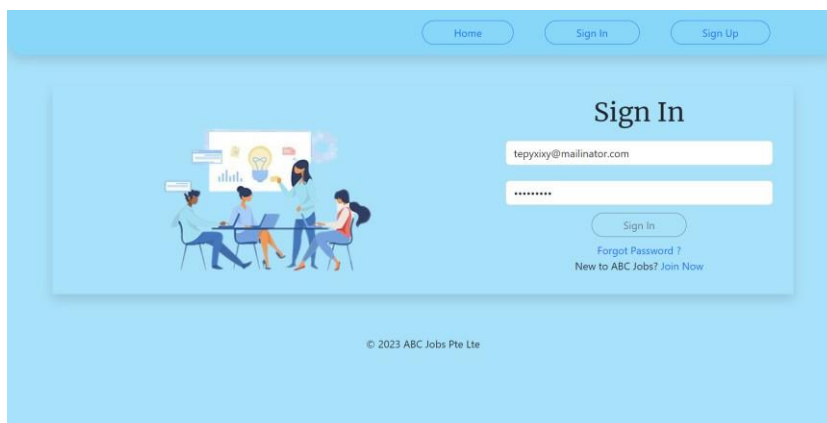
1.1 Set Toggle Breakpoint



1.2 Debug on server



1.3 Enter Wrong Credentials



1.4 In Debug perspective will show the result

The debug stop at the breakpoint because we input wrong credential

9. Task 9

Task Statement:

1. Provide screen capture of 3 Errors you encountered while developing the application.
2. Include it as part of Project Presentation

Error	Error handling (Solution)
	<ul style="list-style-type: none"> - If you run this program but this error is shown - Look at persistence.xml - Change the database name into your database - Change the password with your password
	<ul style="list-style-type: none"> - Look at search.jsp - Compare variable and parameter with search controller - Make sure you have the value, param same with controller, because if different the data will not show
	<ul style="list-style-type: none"> - Look in repository - Concern in search method - Do you have a title or not, if not change with other data that you have like city

10. Task 10

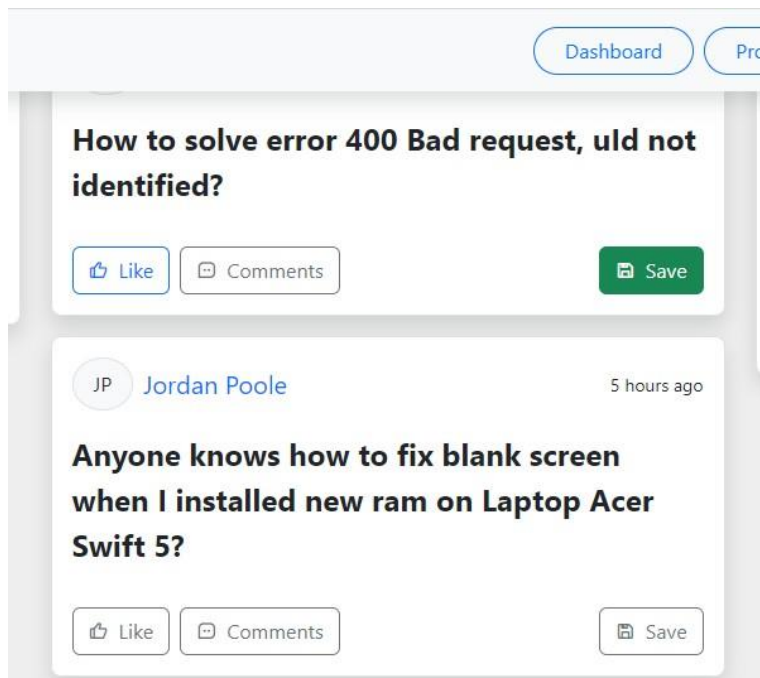
Task Statement:

1. Propose & Document a New feature to the application

1.1 Button to like, comment and save.

2. Develop the new feature and provide a report.

2.1 Button to like, comment and save



11. Task 11

Task Statement:

Provide Unit Testing code & screen capture of Eclipse IDE in debug mode.

- Test Scenario

Requirements	Test Scenario ID	Test Scenario	Number of test cases
Functional Testing	TS001-UAT testing	Verify that users can validate their email and register their information.	1
	TS002-UAT testing	Verify that after logging in, the user may search for, locate, and browse the profiles of other software engineers.	1
	TS003-UAT testing	Verify the website's ability to change passwords.	1

Test Scenario	Verify that users can validate their email and register their information.
TS001	
Test Cases	UAT Testing
UATC01	Verify that the signup and email confirmation are functional.

Scenario ID	Test Case ID	Test Case	Preconditions	Test Step	Test Data	Expected Results	Actual Results	Pass / Fail
TS001	UATC01	Verify that the signup and email confirmation are operational.	1. The website is designed and developed 2. Unit testing and system testing is done 3. All the major errors have been resolved and documented 4. The website is configured to Google Chrome by Apache Tomcat Server.	1. Go to Google Chrome. 2. Enter URL www.ABCPortal.com 3. Click "Sign Up" in website home page 4. Enter all fields and submit 5. Click confirmation	First Name= syukur Last Name= sidiq City = Seattle Email= syukur@gmail.com Password="qwerty123"	User should land to login page	As expected,	Pass

Home Sign In Sign Up

Sign Up

pihu@mailinator.com

Gregory

Hull

Voluptatem est adipiscing qui mollit vero maione

Password

Confirm Password

Agree and Join

Already on ABC Jobs? [Sign In](#)

© 2023 ABC Jobs Pte Ltd



13	2	pihu@mailinator.com	syukur123	2023-03-09
----	---	---------------------	-----------	------------



Home Sign In Sign Up

Thank You


pihu@mailinator.com

Your registration has been received and we appreciate your time.

Login

© 2023 ABC Jobs Pte Ltd





Successful


The link has been sent it to your email

Continue



Home Sign In Sign Up

Verified



Your account has been verified

Please click below to Continue

Login

© 2023 ABC Jobs Pte Ltd



Home Sign In Sign Up

Sign In

Email Address

Password

Sign In

Forgot Password?

How to ABC jobs? [Join Now](#)

© 2023 ABC Jobs Pte Ltd

Test Scenario	Verify that after logging in, the user may search for, locate, and browse the profiles of other software engineers.
TS002	
Test Cases	UAT Testing
UATC01	Verify that the functions for searching and browsing profiles are functional.

Scenario ID	Test Case ID	Test Case	Preconditions	Test Step	Test Data	Expected Results	Actual Results	Pass / Fail
TS002	UATC01	Check searching and viewing profile function is working correctly	1. The website is designed and developed 2. Unit testing and system testing is done 3. All the major errors have been resolved and documented 4. The website is configured to Google Chrome by Apache Tomcat Server. 5. User has successfully	1. Go to Google Chrome. 2. Enter URL www.ABCPortal.com 3. Click Login website home page 4. Login with registered credentials 5. Click "Search" on the navigation bar 6. Search for "Syukur" and click on the "View Profile" button	Search keyword = syukur	User should be able to view syukur profile	As expected,	Pass

			registered and login to the website					
--	--	--	---	--	--	--	--	--



Search Other

syukur

Press enter to search

syukur sidiq
bandung

View Profile



syukur sidiq
bandung

Follow

Status

Education

Experiences

your data

First Name : syukur

Last Name : sidiq

Email : syukur2@gmail.com

Phone Number : -

Address : bandung

Test Scenario	Check change password functionality in the website
TS003	
Test Cases	UAT Testing
UATC01	Check that password functionality is working correctly

Scenario ID	Test Case ID	Test Case	Preconditions	Test Step	Test Data	Expected Result
TS003	UATC01	Check that password functionality is working correctly	1. The website is designed and developed 2. Unit testing and system testing is done 3. All the major errors has resolved and documented	1. Go to Google Chrome. 1. Enter _____ URL www.ABCPortal.com 2. Click "Sign in" in website home page 3. Click "Forgot Password ?" in the Login Page 4. Enter registered email 5. Enter new password	Registered email = Syukur2@gmail.com Old password = "syukur123" New password = syukur	User should be able to login with the new password

- | | | | | | | |
|--|--|--|---|--|--|--|
| | | | <p>4. The website is configured to Google Chrome by Apache Tomcat Server.</p> <p>5. User has successfully registered and login to the website</p> | | | |
|--|--|--|---|--|--|--|

The screenshot shows a web browser window with a light blue background. At the top, there are three buttons: 'Home', 'Sign In', and 'Sign Up'. The main heading is 'Forgot Password'. Below it, a white box contains the text 'Enter your email and we'll send you a link to reset your password'. There is an input field for 'Email address' with the text 'syukur2@gmail.com' and a blue 'Submit' button. At the bottom, there is a copyright notice: '© 2023 ABC Jobs Pte Ltd'.



The screenshot shows a web browser window with a light blue background. At the top, there are three buttons: 'Home', 'Sign In', and 'Sign Up'. The main heading is 'Reset Your Password'. Below it, a white box contains two input fields: 'New Password' and 'Password Confirmation', both with masked text '*****'. There is a blue 'Update Password' button. At the bottom, there is a copyright notice: '© 2023 ABC Jobs Pte Ltd'.



Sign In

Password has been changed ×

Email Address

Password

Sign In

[Forgot Password ?](#)

New to ABC Jobs? [Join Now](#)