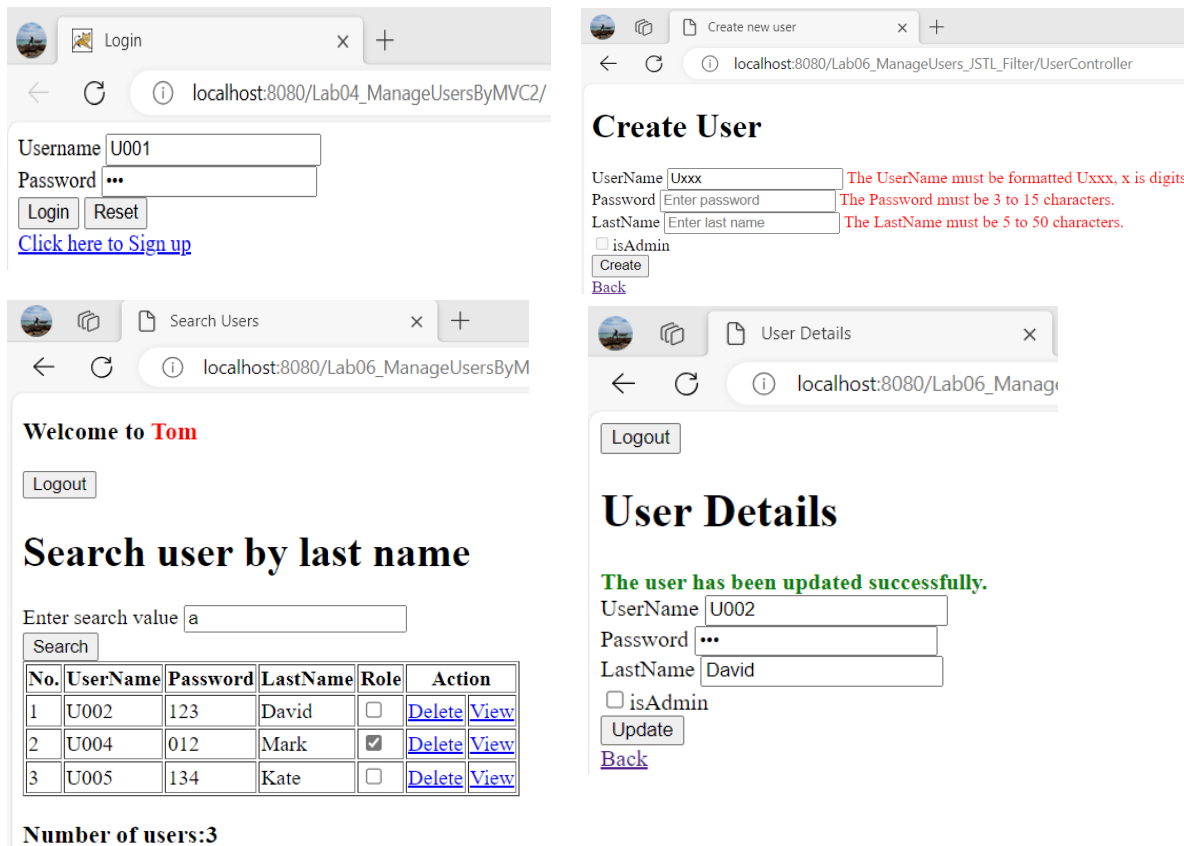


Building a Java Web Application

Using MVC2, JSP, JSTL & Filter

Requirements

Create a Java web application (using HTML, Servlets, JSP, EL, JSTL, Filter, and JDBC) by MVC2 Model to perform functions as manage users.



The screenshots show a web application running on localhost:8080. The first screenshot shows the 'Login' page with fields for Username (U001) and Password, and buttons for Login, Reset, and a link to Sign up. The second screenshot shows the 'Create new user' page with fields for Username (Uxxx), Password, and LastName, with validation messages. The third screenshot shows the 'Search Users' page with a search bar and a table of users. The fourth screenshot shows the 'User Details' page with fields for Username (U002), Password, and LastName, and buttons for Logout, Update, and Back.

Welcome to Tom

[Logout](#)

Search user by last name

Enter search value

[Search](#)

No.	UserName	Password	LastName	Role	Action
1	U002	123	David	<input type="checkbox"/>	Delete View
2	U004	012	Mark	<input checked="" type="checkbox"/>	Delete View
3	U005	134	Kate	<input type="checkbox"/>	Delete View

Number of users:3

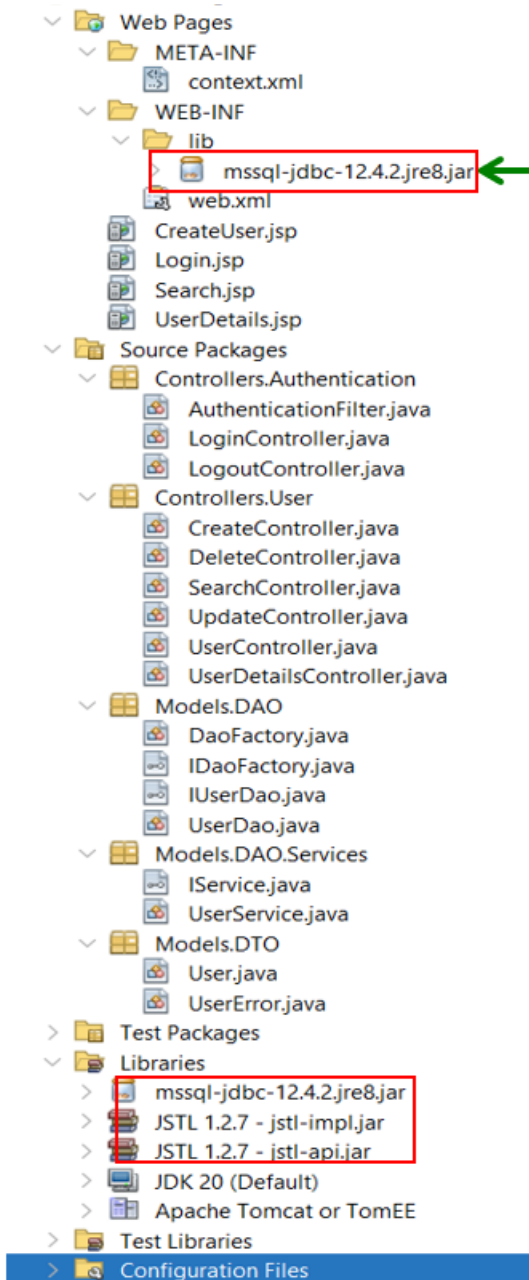
Lab Objectives

In this lab, you will be:

- Create Java web project by MVC2 Model
- Create the web pages by HTML, and JSP (JSTL & EL)
- Apply passing data from HTML to Servlet by GET and POST method
- **Apply Filter and Session for the Authentication**
- Apply Error handling in the web app

- Apply design pattern: Repository and Factory pattern
- Apply RequestDispatcher to pass data between Servlets
- Performs CRUD functions by SqlJDBC driver and SQL Server
- Create a dynamic connection to the SQL Server
- Run the project and test the functions of the project

Step 01. Create a Java Web App named **Lab07ManageUsers_JSTL_Filter** is structured as follows:



- On the project, right-click on Libraries | Add JAR/Folder then select **sqljdbc(sqljdbc42.jar)** driver that **matches the SQL Server version** on the machine.

-Open SQL Server Management Studio and create a database named **SampleDB** has a table named **Registration** as follows:

Column Name	Data Type	Allow Nulls
UserName	varchar(30)	<input type="checkbox"/>
Password	varchar(30)	<input type="checkbox"/>
LastName	nvarchar(50)	<input type="checkbox"/>
isAdmin	bit	<input type="checkbox"/>

UserName	Password	LastName	isAdmin
U001	123	Tom	True
U002	456	David	False
U003	789	John	False
U004	012	Mark	True
U005	134	Kate	False

-Open the **Context.xml** in the META-INF folder and config for the connection string as follows:

```

1 <?xml version="1.0" encoding="UTF-8"?>
2 <Context path="/Lab06_ManageUsers_JSTL_Filter">
3   <Resource auth="Container"
4     driverClassName="com.microsoft.sqlserver.jdbc.SQLServerDriver"
5     name="SampleDB"
6     password="123"
7     type="javax.sql.DataSource"
8     url="jdbc:sqlserver://localhost:1433;databaseName=SampleDB;userName=sa"/>
9 </Context>

```

-Open the **web.xml** in the WEB-INF folder then update as follows:

```

1 <?xml version="1.0" encoding="UTF-8"?>
2 <web-app version="3.1" xmlns="http://xmlns.jcp.org/xml/ns/javaee" xmlns:
3   <servlet>
4     <servlet-name>UserController</servlet-name>
5     <servlet-class>Controllers.User.UserController</servlet-class>
6   </servlet>
7   <servlet-mapping>
8     <servlet-name>UserController</servlet-name>
9     <url-pattern>/UserController</url-pattern>
10  </servlet-mapping>
11  <session-config>
12    <session-timeout>
13      30
14    </session-timeout>
15  </session-config>
16  <welcome-file-list>
17    <welcome-file>Login.jsp</welcome-file>
18  </welcome-file-list>
19  <resource-ref>
20    <res-ref-name>DBCon</res-ref-name>
21    <res-type>javax.sql.DataSource</res-type>
22    <res-auth>Container</res-auth>
23    <res-sharing-scope>Shareable</res-sharing-scope>
24  </resource-ref>
25 </web-app>

```

Step 02. Write codes for the classes in the **Models.DTO**, **Models.DAO** and **Models.DAO.Services** packages are as follows:

2.1. Models.DTO package

- User.java**

```

1  package Models.DTO;
2  import java.io.Serializable;
3  /**
4   * @author SwordLake
5   */
6  public class User implements Serializable{
7      private String userName;
8      private String password;
9      private String lastName;
10     private boolean isAdmin;
11
12     public User() {
13         this.userName = null;
14         this.password = null;
15         this.lastName = null;
16         this.isAdmin = false;
17     }
18     public User(String userName, String password,
19         String lastName, boolean isAdmin) {...6 lines }
20
21     public String getUserName() {...3 lines }
22     public void setUserName(String userName) {...3 lines }
23     public String getPassword() {...3 lines }
24     public void setPassword(String password) {...3 lines }
25     public String getLastName() {...3 lines }
26     public void setLastName(String lastName) {...3 lines }
27     public boolean isIsAdmin() {...3 lines }
28     public void setIsAdmin(boolean isAdmin) {...3 lines }
29
30     @Override
31     public String toString() {
32         return String.format("UserName:%s, Password:%s, LastName:%s, isAdmin:%b",
33             userName, password, lastName, isAdmin);
34     }
35 }

```

- UserError.java**

```

UserError.java x
Source History
1 package Models.DTO;
2
3 /**
4  * * @author SwordLake
5  */
6 public class UserError {
7     private String userNameError;
8     private String lastNameError;
9     private String passwordError;
10    private String duplicateUserName;
11    public UserError() { }
12    public UserError(String userNameError, String lastNameError,
13                     String passwordError, String duplicateUserName) {
14        this.userNameError = userNameError;
15        this.lastNameError = lastNameError;
16        this.passwordError = passwordError;
17        this.duplicateUserName = duplicateUserName;
18    }
19    public String getUserNameError() {...3 lines }
22    public void setUserNameError(String userNameError) {...3 lines }
25    public String getLastNameError() {...3 lines }
28    public void setLastNameError(String lastNameError) {...3 lines }
31    public String getPasswordError() {...3 lines }
34    public void setPasswordError(String passwordError) {...3 lines }
37    public String getDuplicateUserName() {...3 lines }
40    public void setDuplicateUserName(String duplicateUserName) {...3 lines }
43 }
  
```

2.2. Models.DAO package

• IUserDao.java

```

IUserDao.java x
Source History
1 package Models.DAO;
2
3 import Models.DTO.User;
4 import java.util.List;
5 /**
6  * @author SwordLake
7  */
8 public interface IUserDao {
9     User login(String userName, String password) throws Exception ;
10    User getUserByUserName(String userName) throws Exception ;
11    List<User> searchUserByLastName(String searchValue) throws Exception ;
12    boolean addUser(User user) throws Exception ;
13    boolean deleteUser(String userName) throws Exception ;
14    boolean updateUser(User user) throws Exception ;
15 }
  
```

• UserDao.java

UserDao.java x

Source History

```

7   import java.sql.SQLException;
8   import java.util.ArrayList;
9   import java.util.List;
10  import javax.naming.Context;
11  import javax.naming.InitialContext;
12  import javax.sql.DataSource;
13  /**
14   * @author SwordLake
15   */
16  public class UserDao implements IUserDao{
17      // <editor-fold defaultstate="collapsed" desc="getConnection Method">
18      public static Connection getConnection() throws Exception {
19          Connection cnn;
20          try {
21              //Using context (Dynamic connection)
22              //1. get current context
23              Context currentContext = new InitialContext();
24              //2. Look up tomcat context
25              Context tomcatContext = (Context) currentContext.lookup("java:comp/env");
26              //3. Look up our datasource
27              DataSource ds = (DataSource) tomcatContext.lookup("SampleDB");
28              //4. Open connection from DS
29              cnn = ds.getConnection();
30              return cnn;
31          } catch (SQLException ex) {
32              throw ex;
33          }
34      }
35      // </editor-fold>
36
37      // <editor-fold defaultstate="collapsed" desc="login Method">
38      public User login(String userName, String password) throws Exception {
39          User user = null;
40          Connection cnn = null;
41          PreparedStatement preStm = null;
42          ResultSet rs = null;
43          String lastName;
44          boolean isAdmin;
45          try {
46              cnn = getConnection();
47              String sql =
48                  "select LastName,isAdmin from Registration where [UserName]=? and [Password]=?";
49              preStm = cnn.prepareStatement(sql);
50              preStm.setString(1, userName);
51              preStm.setString(2, password);
52              rs = preStm.executeQuery();
53              while (rs.next()) {
54                  lastName = rs.getString(1);
55                  isAdmin = rs.getBoolean(2);
56                  user = new User(userName, password, lastName, isAdmin);
57              } //end while
58          } catch (Exception ex) {
59              throw ex;
60          }
61      }
62      // </editor-fold>

```

context.xml x

Source History

```

1   <?xml version="1.0" encoding="UTF-8"?>
2   <Context path="/Lab06_ManageUsers_JSTL_Filter"
3       <Resource auth="Container" driverClassName=
4           name="SampleDB"
5           password="123"
6           type="javax.sql.DataSource"
7           url="jdbc:sqlserver://localhost:1
8       </Context>

```

```

53         } finally {
54             if (rs != null) {
55                 rs.close();
56             }
57             if (preStm != null) {
58                 preStm.close();
59             }
60             if (cnn != null) {
61                 cnn.close();
62             }
63         }
64         return user;
65     } //end login
66 // </editor-fold>

68 // <editor-fold defaultstate="collapsed" desc="getUserByUserName Method">
69 public User getUserByUserName(String userName) throws Exception {
70     User user = null;
71     Connection cnn = null;
72     PreparedStatement preStm = null;
73     ResultSet rs = null;
74     String lastName, password;
75     boolean isAdmin;
76     try {
77         cnn = getConnection();
78         String sql =
79             "select Password, LastName, isAdmin from Registration where [UserName]=?";
80         preStm = cnn.prepareStatement(sql);
81         preStm.setString(1, userName);
82         rs = preStm.executeQuery();

83         while (rs.next()) {
84             password = rs.getString(1);
85             lastName = rs.getString(2);
86             isAdmin = rs.getBoolean(3);
87             user = new User(userName, password, lastName, isAdmin);
88         } //end while
89     } catch (Exception ex) {
90         throw ex;
91     } finally {
92         if (rs != null) {
93             rs.close();
94         }
95         if (preStm != null) {
96             preStm.close();
97         }
98         if (cnn != null) {
99             cnn.close();
100         }
101     }
102     return user;
103 } //end getUserByUserName
104 // </editor-fold>

```

```

106 // <editor-fold defaultstate="collapsed" desc="searchUserByLastName Method">
107 public List<User> searchUserByLastName(String searchValue) throws Exception {
108     String userName, password, lastName;
109     boolean isAdmin;
110     Connection cnn = null;
111     PreparedStatement preStm = null;
112     ResultSet rs = null;
113     List<User> userList = new ArrayList();
114     try {
115         cnn = getConnection();
116         String sql = "select UserName,Password,LastName,isAdmin from Registration "
117             + " where LastName like ?";
118         preStm = cnn.prepareStatement(sql);
119         preStm.setString(1, "%" + searchValue + "%");
120         rs = preStm.executeQuery();
121         while (rs.next()) {
122             userName = rs.getString(1);
123             password = rs.getString(2);
124             lastName = rs.getString(3);
125             isAdmin = rs.getBoolean(4);
126             User user = new User(userName, password, lastName, isAdmin);
127             userList.add(user);
128         } //end while
129     } catch (Exception ex) {
130         throw ex;
131     } finally {
132         if (rs != null) {
133             rs.close();
134         }
135         if (preStm != null) {
136             preStm.close();
137         }
138         if (cnn != null) {
139             cnn.close();
140         }
141     }
142     if (userList.isEmpty()) {
143         return null;
144     }
145     return userList;
146 } //end searchUserByLastName
147 // </editor-fold>

```



```

149 // <editor-fold defaultstate="collapsed" desc="addUser Method">
150 public boolean addUser(User user) throws Exception {
151     PreparedStatement preStm = null;
152     Connection cnn = null;
153     try {
154         cnn = getConnection();
155         String sql = "Insert Registration values(?,?,?,?)";
156         preStm = cnn.prepareStatement(sql);
157         preStm.setString(1, user.getUserName());
158         preStm.setString(2, user.getPassword());
159         preStm.setString(3, user.getLastname());
160         preStm.setBoolean(4, user.isIsAdmin());
161         return preStm.executeUpdate() > 0;
162     } catch (Exception ex) {
163         throw ex;
164     } finally {
165         if (preStm != null) {
166             preStm.close();
167         }
168         if (cnn != null) {
169             cnn.close();
170         }
171     }
172 } //end addUser

175 // <editor-fold defaultstate="collapsed" desc="deleteUser Method">
176 public boolean deleteUser(String userName) throws Exception {
177     PreparedStatement preStm = null;
178     Connection cnn = null;
179     try {
180         cnn = getConnection();
181         String sql = "delete Registration Where UserName=?";
182         preStm = cnn.prepareStatement(sql);
183         preStm.setString(1, userName);
184         return preStm.executeUpdate() > 0;
185     } catch (Exception ex) {
186         throw ex;
187     } finally {
188         if (preStm != null) {
189             preStm.close();
190         }
191         if (cnn != null) {
192             cnn.close();
193         }
194     }
195 } //end deleteUser
196 // </editor-fold>

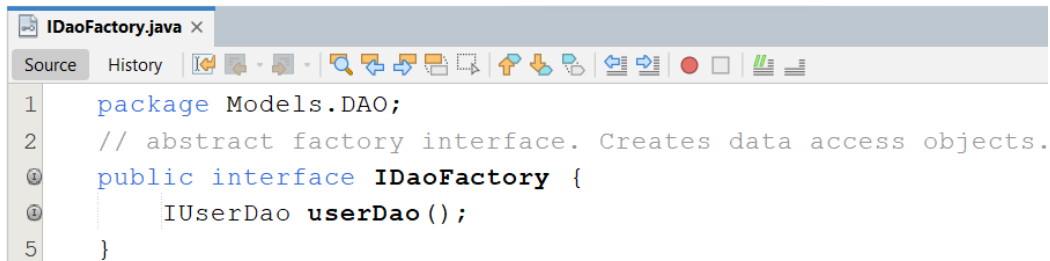
```

```

198 // <editor-fold defaultstate="collapsed" desc="updateUser Method">
199 public boolean updateUser(User user) throws Exception {
200     PreparedStatement preStm = null;
201     Connection cnn = null;
202     try {
203         cnn = getConnection();
204         String sql="Update Registration Set Password=?,LastName=?,isAdmin=? Where UserName=?";
205         preStm = cnn.prepareStatement(sql);
206         preStm.setString(1, user.getPassword());
207         preStm.setString(2, user.getLastname());
208         preStm.setBoolean(3, user.isIsAdmin());
209         preStm.setString(4, user.getUserName());
210         return preStm.executeUpdate() > 0;
211     } catch (Exception ex) {
212         throw ex;
213     } finally {
214         if (preStm != null) {
215             preStm.close();
216         }
217         if (cnn != null) {
218             cnn.close();
219         }
220     }
221 } //end updateUser
222 // </editor-fold>

```

• IDaoFactory.java

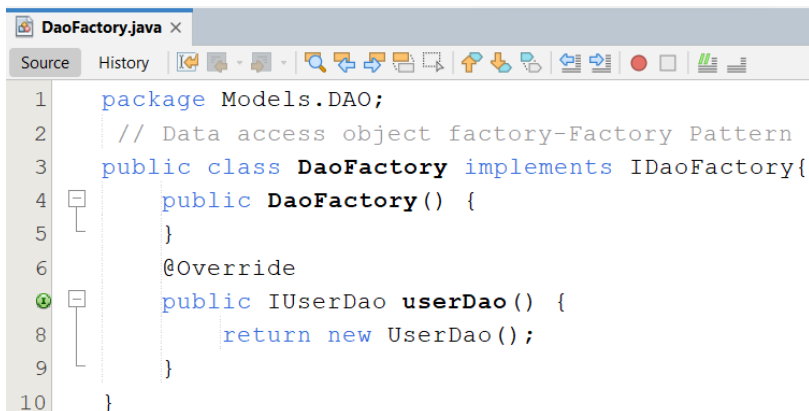


```

1 package Models.DAO;
2 // abstract factory interface. Creates data access objects.
3 public interface IDaoFactory {
4     IUserDao userDao();
5 }

```

• DaoFactory.java



```

1 package Models.DAO;
2 // Data access object factory-Factory Pattern
3 public class DaoFactory implements IDaoFactory{
4     public DaoFactory() {
5     }
6     @Override
7     public IUserDao userDao() {
8         return new UserDao();
9     }
10 }

```

2.3. Models. DAO.Services package

• IService.java

```

1  package Models.DAO.Services;
2
3  import Models.DTO.User;
4  import java.util.List;
5
6  //Single interface to all 'repositories'
7  public interface IService<T> {
8
9      //Repository
10     User checkAccount(String userName, String password) throws Exception ;
11     User getUserByUserName(String userName) throws Exception ;
12     List<User> getUsersByKeywordOfLastName(String value) throws Exception ;
13     boolean addNewUser(User user) throws Exception ;
14     boolean removeUser(String userName) throws Exception ;
15     boolean updateUser(User user) throws Exception;
16 }

```

• UserService.java

```

1  package Models.DAO.Services;
2
3  import Models.DAO.DaoFactory;
4  import Models.DAO.IUserDao;
5  import Models.DTO.User;
6  import java.util.List;
7
8  /**
9   * @author SwordLake
10  */
11 public class UserService implements IService<User>{
12     IUserDao userDao;
13
14     public UserService() {
15         this.userDao = new DaoFactory().userDao();
16     }
17
18     @Override
19     public User checkAccount(String userName, String password) throws Exception {
20         return userDao.login(userName, password);
21     }
22
23     @Override
24     public User getUserByUserName(String userName) throws Exception {
25         return userDao.getUserByUserName(userName);
26     }
27
28     @Override
29     public List<User> getUsersByKeywordOfLastName(String value) throws Exception {
30         return userDao.searchUserByLastName(value);
31     }
32 }

```

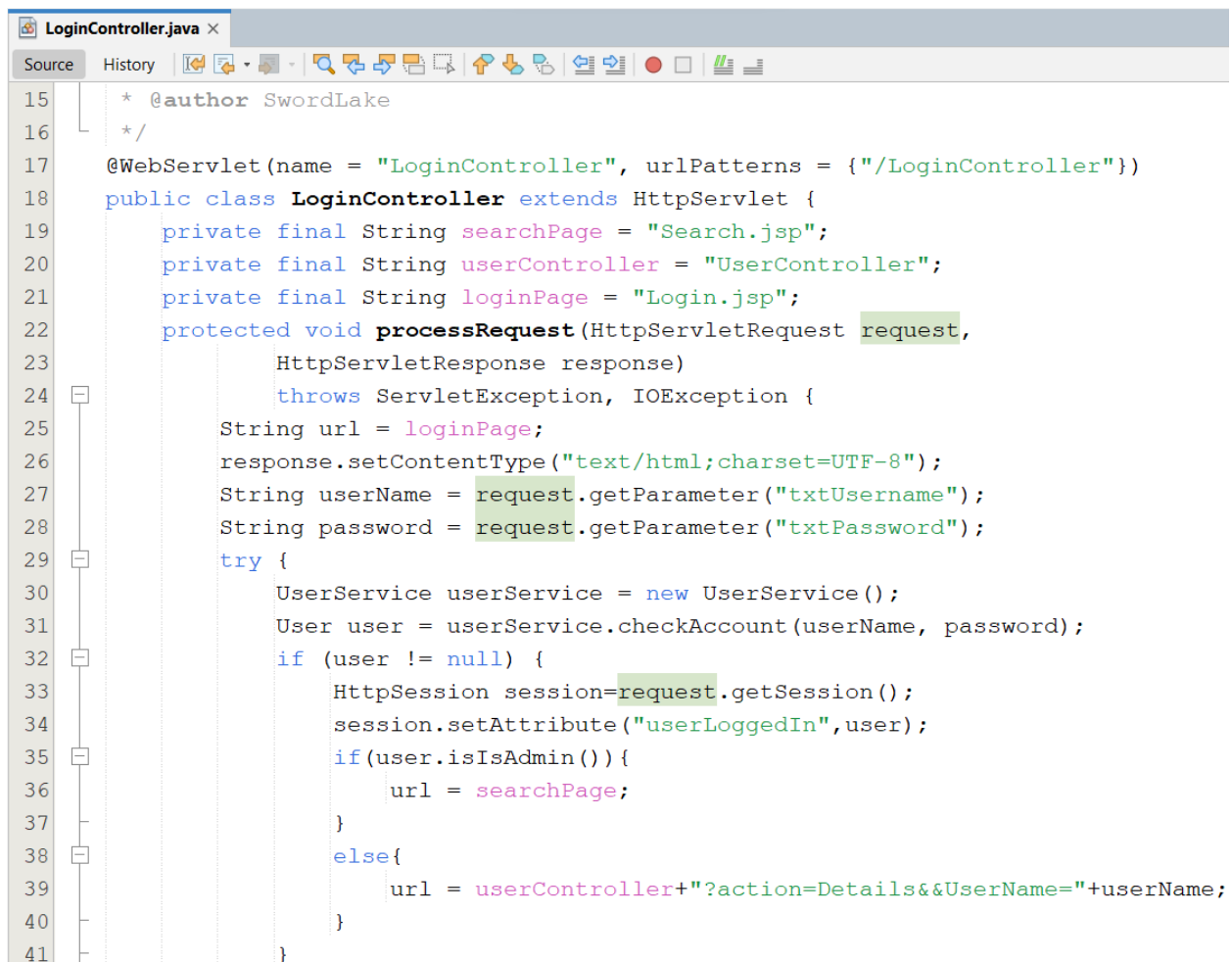
```

27     @Override
28     public boolean addNewUser(User user) throws Exception {
29         return userDao.addUser(user);
30     }
31     @Override
32     public boolean removeUser(String userName) throws Exception {
33         return userDao.deleteUser(userName);
34     }
35     @Override
36     public boolean updateUser(User user) throws Exception {
37         return userDao.updateUser(user);
38     }
39 }

```

Step 03. Write codes for the classes in the **Controllers.Authentication** package is as follows:

- **LoginController.java**



```

15  * @author SwordLake
16  */
17  @WebServlet(name = "LoginController", urlPatterns = {"/LoginController"})
18  public class LoginController extends HttpServlet {
19      private final String searchPage = "Search.jsp";
20      private final String userController = "UserController";
21      private final String loginPage = "Login.jsp";
22      protected void processRequest(HttpServletRequest request,
23                                  HttpServletResponse response)
24          throws ServletException, IOException {
25          String url = loginPage;
26          response.setContentType("text/html;charset=UTF-8");
27          String userName = request.getParameter("txtUsername");
28          String password = request.getParameter("txtPassword");
29          try {
30              UserService userService = new UserService();
31              User user = userService.checkAccount(userName, password);
32              if (user != null) {
33                  HttpSession session = request.getSession();
34                  session.setAttribute("userLoggedIn", user);
35                  if (user.isIsAdmin()) {
36                      url = searchPage;
37                  }
38                  else {
39                      url = userController + "?action=Details&&UserName=" + userName;
40                  }
41              }

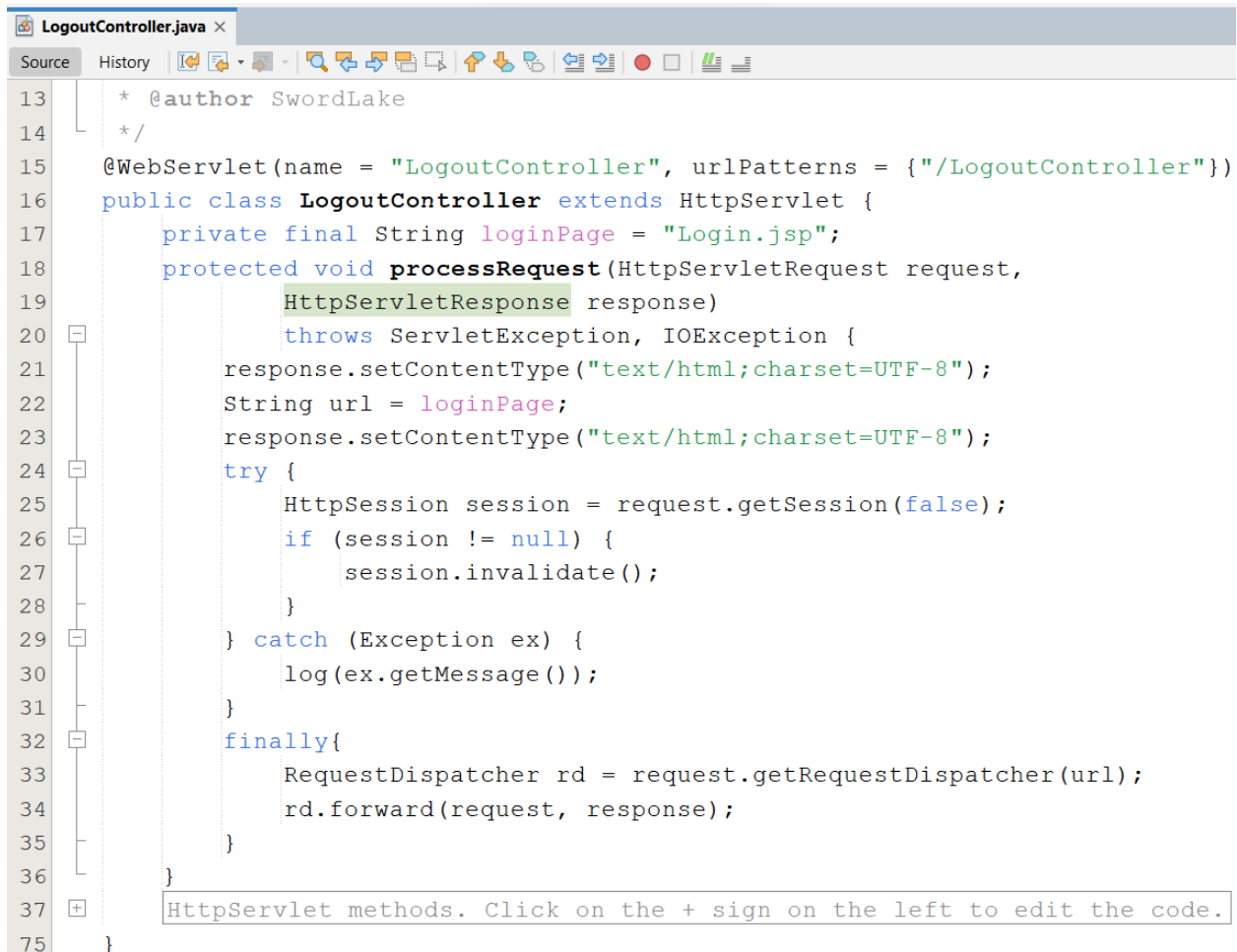
```

```

42         else{
43             request.setAttribute("message", "The user name or password is invalid");
44         }
45     } catch (Exception ex) {
46         log(ex.getMessage());
47     }
48     finally
49     {
50         RequestDispatcher rd = request.getRequestDispatcher(url);
51         rd.forward(request, response);
52     }
53 }
54
55 HttpServlet methods. Click on the + sign on the left to edit the code.
56
57 }

```

• LogoutController.java

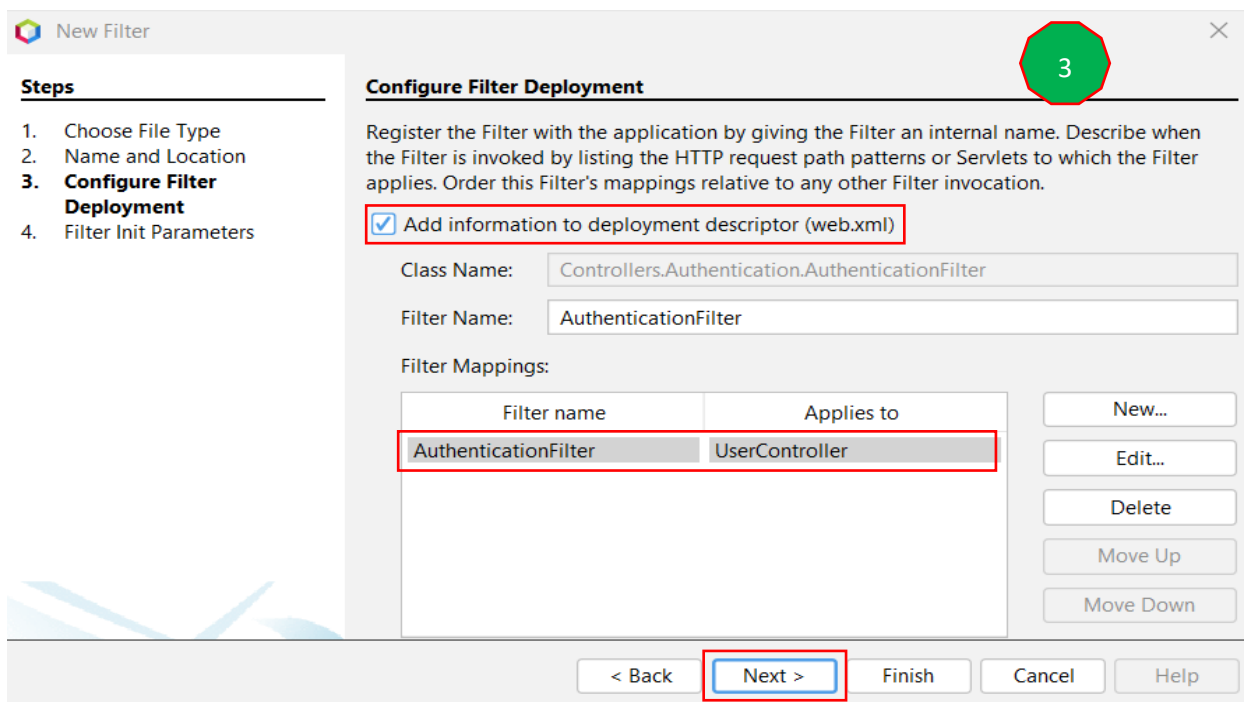
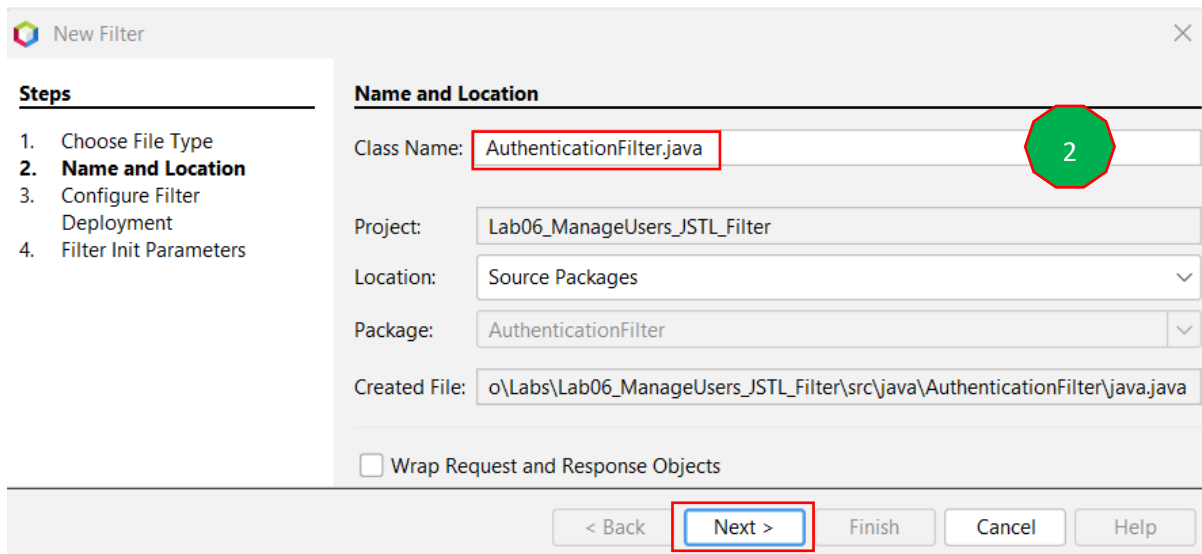
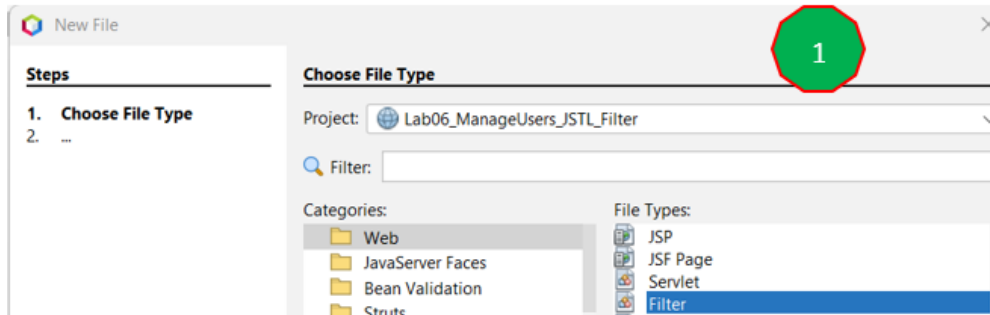


```

13  * @author SwordLake
14  */
15  @WebServlet(name = "LogoutController", urlPatterns = {"/LogoutController"})
16  public class LogoutController extends HttpServlet {
17      private final String loginPage = "Login.jsp";
18      protected void processRequest(HttpServletRequest request,
19                                  HttpServletResponse response)
20          throws ServletException, IOException {
21          response.setContentType("text/html; charset=UTF-8");
22          String url = loginPage;
23          response.setContentType("text/html; charset=UTF-8");
24          try {
25              HttpSession session = request.getSession(false);
26              if (session != null) {
27                  session.invalidate();
28              }
29          } catch (Exception ex) {
30              log(ex.getMessage());
31          }
32          finally{
33              RequestDispatcher rd = request.getRequestDispatcher(url);
34              rd.forward(request, response);
35          }
36      }
37
38      HttpServlet methods. Click on the + sign on the left to edit the code.
39
40  }

```

-Right-click on the Controllers.Authentication package then add a filter controller named **AuthenticationFilter.java** as follows:



-Open the **web.xml** in the WEB-INF folder to view configuration information of the **AuthenticationFilter** and update if not available as follows:

```

1  <?xml version="1.0" encoding="UTF-8"?>
2  <web-app version="3.1" xmlns="http://xmlns.jcp.org/xml/ns/javaee" xmlns:xsi="http://
3  <filter>
4      <filter-name>AuthenticationFilter</filter-name>
5      <filter-class>Controllers.Authentication.AuthenticationFilter</filter-class>
6  </filter>
7  <filter-mapping>
8      <filter-name>AuthenticationFilter</filter-name>
9      <servlet-name>UserController</servlet-name>
10 </filter-mapping>
11 <servlet>
12     <servlet-name>UserController</servlet-name>
13     <servlet-class>Controllers.User.UserController</servlet-class>
14 </servlet>
15 <servlet-mapping>
16     <servlet-name>UserController</servlet-name>
17     <url-pattern>/UserController</url-pattern>
18 </servlet-mapping>
19 <session-config>
20     <session-timeout>30</session-timeout>
21 </session-config>
22 <welcome-file-list>
23     <welcome-file>index.html</welcome-file>
24 </welcome-file-list>
25 <resource-ref>
26     <res-ref-name>jsp/servlet/</res-ref-name>
27     <res-class>javax.servlet.Servlet</res-class>
28     <res-url>/servlet/</res-url>
29 </resource-ref>
30 </web-app>

```

New Filter

Steps

1. Choose File Type
2. Name and Location
3. Configure Filter Deployment
4. **Filter Init Parameters**

Filter Init Parameters

Specify any init parameters for the Filter.

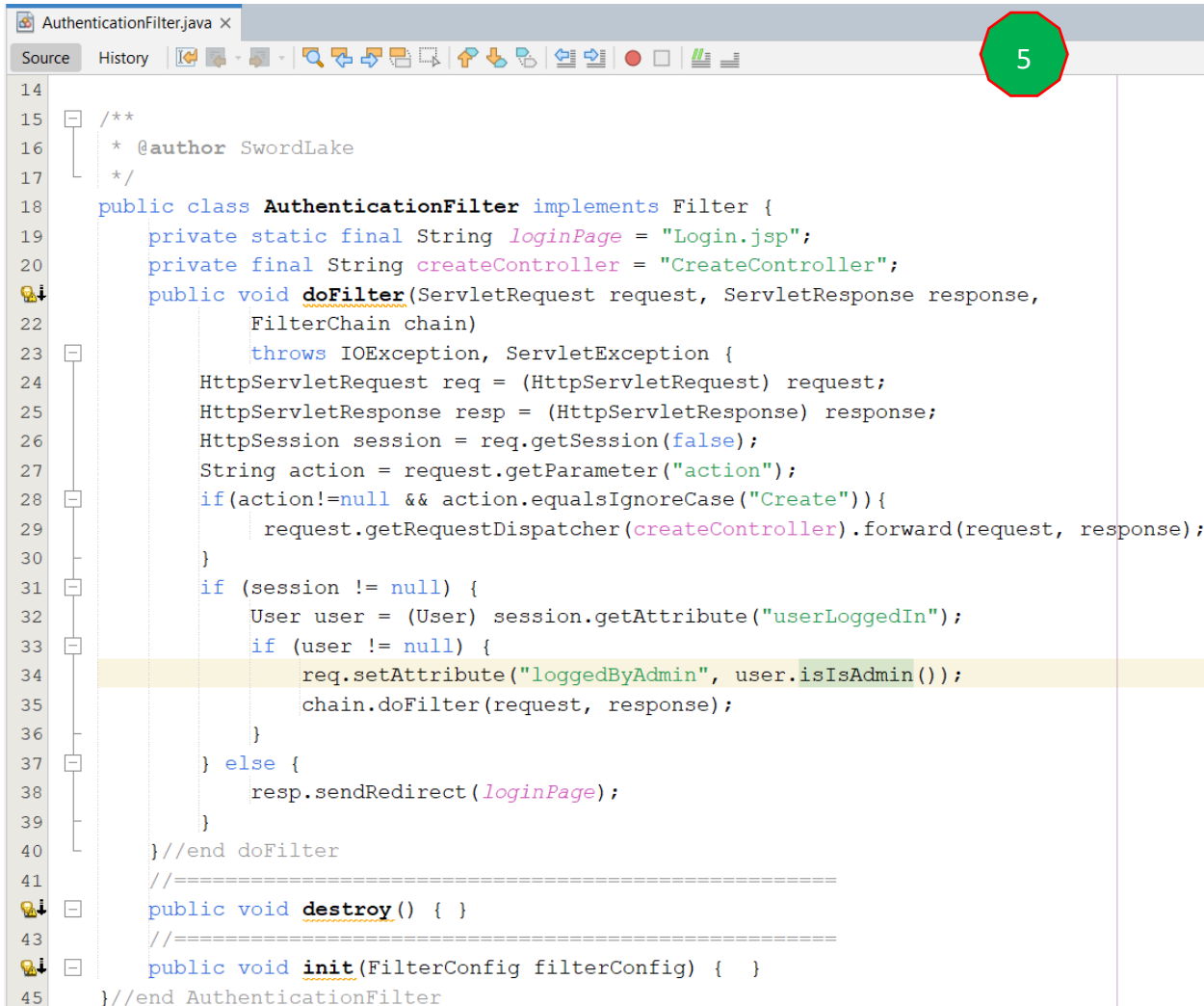
Initialization Parameters:

Name	Value

New Edit... Delete

< Back Next > **Finish** Cancel Help

- Write codes for **AuthenticationFilter.java** as follows:



```

14
15 /**
16  * @author SwordLake
17  */
18 public class AuthenticationFilter implements Filter {
19     private static final String loginPage = "Login.jsp";
20     private final String createController = "CreateController";
21     public void doFilter(ServletRequest request, ServletResponse response,
22         FilterChain chain)
23         throws IOException, ServletException {
24         HttpServletRequest req = (HttpServletRequest) request;
25         HttpServletResponse resp = (HttpServletResponse) response;
26         HttpSession session = req.getSession(false);
27         String action = request.getParameter("action");
28         if(action!=null && action.equalsIgnoreCase("Create")){
29             request.getRequestDispatcher(createController).forward(request, response);
30         }
31         if (session != null) {
32             User user = (User) session.getAttribute("userLoggedIn");
33             if (user != null) {
34                 req.setAttribute("loggedByAdmin", user.isIsAdmin());
35                 chain.doFilter(request, response);
36             }
37         } else {
38             resp.sendRedirect(loginPage);
39         }
40     } //end doFilter
41     //=====
42     public void destroy() { }
43     //=====
44     public void init(FilterConfig filterConfig) { }
45 } //end AuthenticationFilter
    
```

Step 04. Write codes for the classes in the **Controllers.User** package is as follows:

- **CreateController.java**


```

CreateController.java x
Source History
14  /**
15   * @author SwordLake
16   */
17  @WebServlet(name = "CreateController", urlPatterns = {"/CreateController"})
18  public class CreateController extends HttpServlet {
19      private final String ceateUserPage = "CreateUser.jsp";
20
21      protected void processRequest(HttpServletRequest request, HttpServletResponse response)
22          throws ServletException, IOException {
23          String userName, password, lastName;
24          boolean isAdmin = true, isError = false;
25          String message = null;
26          response.setContentType("text/html;charset=UTF-8");
27          String url = ceateUserPage;
28          try {
29              UserService userService = new UserService();
30              userName = request.getParameter("txtUserName");
31              password = request.getParameter("txtPassword");
32              lastName = request.getParameter("txtLastName");
33              String admin = request.getParameter("chkIsAdmin");
34              UserError userError = new UserError();
35              if (userName.matches("U\\d{3}") == false) {
36                  userError.setUserNameError("The UserName must be formatted Uxxx, x is digits.");
37                  isError = true;
38              } //end check userName
39              if (password.matches("(.){3,15}") == false) {
40                  userError.setPasswordError("The Password must be 3 to 15 characters.");
41                  isError = true;
42              } //end check Password
43              if (lastName.matches("(.){5,50}") == false) {
44                  userError.setLastNameError("The LastName must be 5 to 50 characters.");
45                  isError = true;
46              } //end check Password
47              if (userService.getUserByUserName(userName) != null) {
48                  userError.setDuplicateUserName("The UserName already exists.");
49                  isError = true;
50              }
51              if (isError == false) {
52                  if (admin == null) {
53                      isAdmin = false;
54                  }
55                  User user = new User(userName, password, lastName, isAdmin);
56                  if (userService.addNewUser(user) == true) {
57                      message = " <b style='color: green'>The user has been created successfully</b>";
58                  }
59                  else {
60                      message = " <b style='color: red'>Something went wrong.</b>";
61                  }
62              } //end isError
63              else {
64                  request.setAttribute("ErrorDetails", userError);
65              }

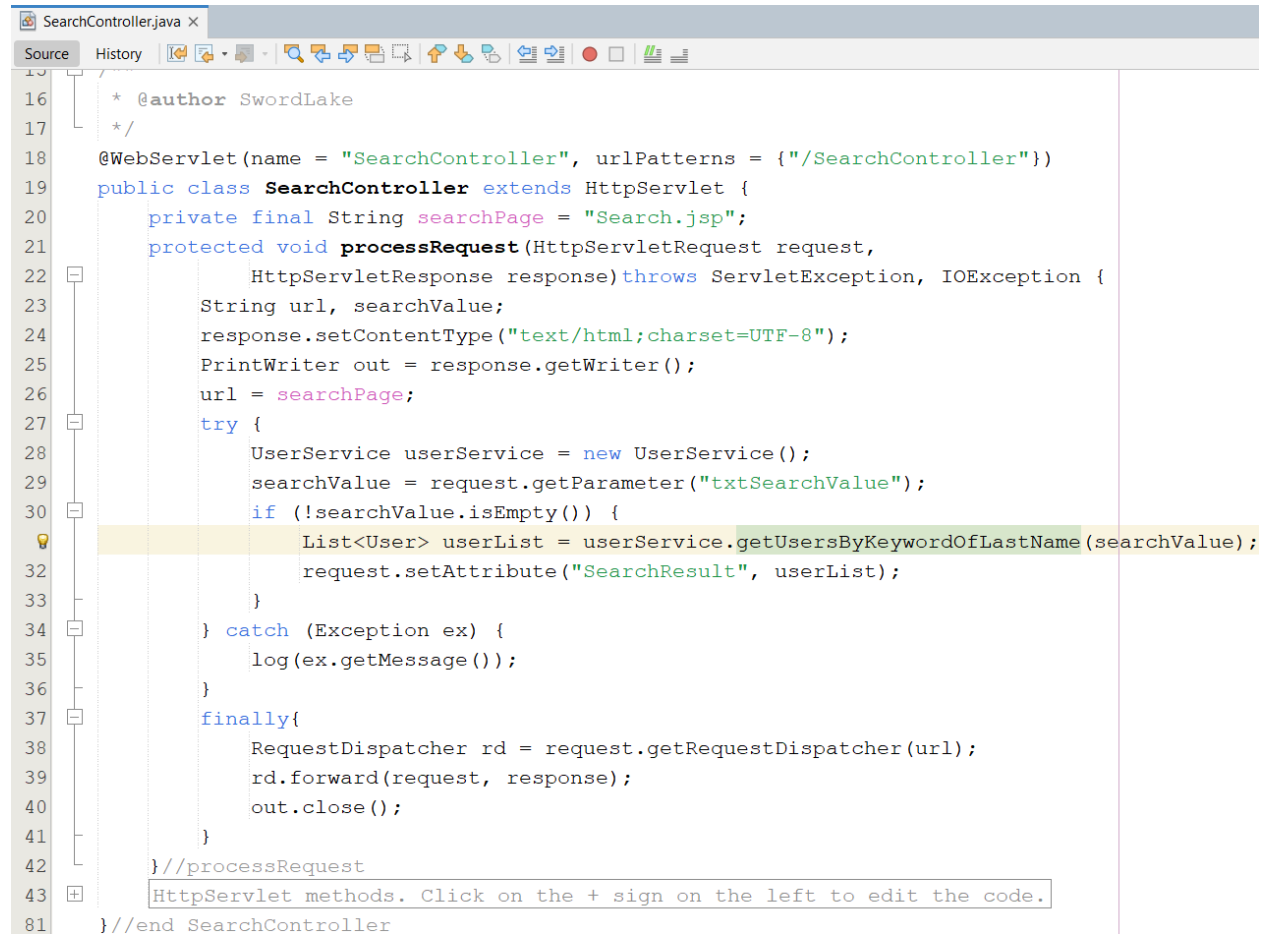
```

```

66         } catch (Exception ex) {
67             log(ex.getMessage());
68         } finally {
69             request.setAttribute("message", message);
70             RequestDispatcher rd = request.getRequestDispatcher(url);
71             rd.forward(request, response);
72         }
73     } //end processRequest
74     HttpServlet methods. Click on the + sign on the left to edit the code.
113 } //end CreateController

```

• SearchController.java



```

15  * @author SwordLake
16  */
17
18  @WebServlet(name = "SearchController", urlPatterns = {"/SearchController"})
19  public class SearchController extends HttpServlet {
20      private final String searchPage = "Search.jsp";
21      protected void processRequest(HttpServletRequest request,
22          HttpServletResponse response) throws ServletException, IOException {
23          String url, searchValue;
24          response.setContentType("text/html;charset=UTF-8");
25          PrintWriter out = response.getWriter();
26          url = searchPage;
27          try {
28              UserService userService = new UserService();
29              searchValue = request.getParameter("txtSearchValue");
30              if (!searchValue.isEmpty()) {
31                  List<User> userList = userService.getUsersByKeywordOfLastName(searchValue);
32                  request.setAttribute("SearchResult", userList);
33              }
34          } catch (Exception ex) {
35              log(ex.getMessage());
36          }
37          finally{
38              RequestDispatcher rd = request.getRequestDispatcher(url);
39              rd.forward(request, response);
40              out.close();
41          }
42      } //processRequest
43      HttpServlet methods. Click on the + sign on the left to edit the code.
81  } //end SearchController

```

• DeleteController.java

```

DeleteController.java x
Source History
14  /* @author SwordLake */
15  @WebServlet(name = "DeleteController", urlPatterns = {"/DeleteController"})
16  public class DeleteController extends HttpServlet {
17      private final String userController = "UserController";
18      protected void processRequest(HttpServletRequest request,
19          HttpServletResponse response) throws ServletException, IOException {
20          response.setContentType("text/html;charset=UTF-8");
21          PrintWriter out = response.getWriter();
22          String url = userController;
23          String message = null;
24          try {
25              UserService userService = new UserService();
26              HttpSession session = request.getSession();
27              User userLoggedIn = (User) session.getAttribute("userLoggedIn");
28              String userName = request.getParameter("UserName");
29              String searchValue = request.getParameter("txtSearchValue");
30              if (userName.equals(userLoggedIn.getUserName())) {
31                  message = "<b style='color: red'>This user logged in, can not delete.</b>";
32              } else {
33                  if (!userName.isEmpty()) {
34                      if (userService.removeUser(userName) == true) {
35                          message = "<b style='color: green'>The user has been deleted successfully.</b>";
36                      } else {
37                          message = "<b style='color: red'>Something went wrong.</b>";
38                      }
39                  }
40              }
41              url = userController+"?action=Search&txtSearchValue="+searchValue;
42          } catch (Exception ex) {
43              log(ex.getMessage());
44          } finally {
45              request.setAttribute("message", message);
46              RequestDispatcher rd = request.getRequestDispatcher(url);
47              rd.forward(request, response);
48              out.close();
49          }
50      } //end processRequest
51      //HttpServlet methods. Click on the + sign on the left to edit the code.
90  } //end DeleteController

```

• UpdateController.java

```

UpdateController.java x
Source History
13  /**
14   * @author SwordLake
15   */
16  @WebServlet(name = "UpdateController", urlPatterns = {"/UpdateController"})
17  public class UpdateController extends HttpServlet {
18      private final String userController = "UserController";
19      protected void processRequest(HttpServletRequest request,
20          HttpServletResponse response) throws ServletException, IOException {
21          response.setContentType("text/html;charset=UTF-8");
22          String userName, password, lastName;
23          boolean isAdmin = true, isError = false;
24          String message = null;
25          response.setContentType("text/html;charset=UTF-8");
26          String url = userController;

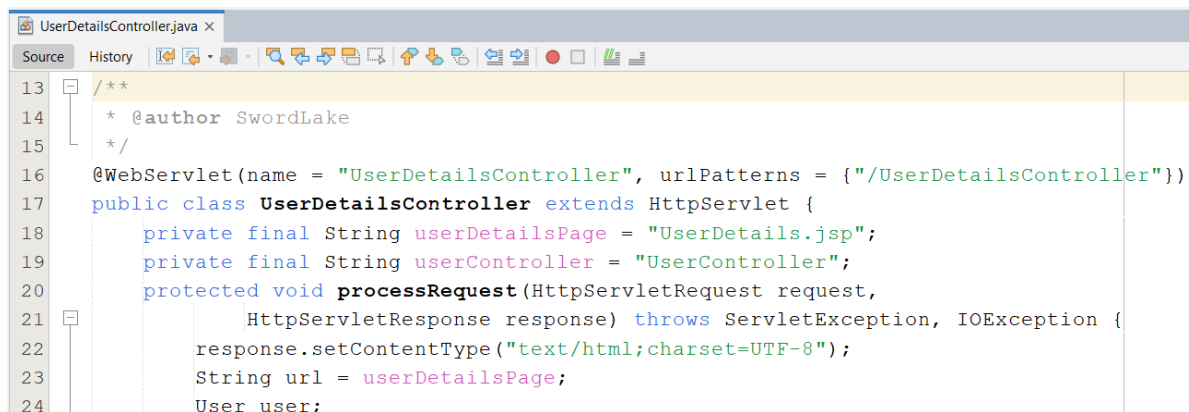
```

```

27     try {
28         UserService userService = new UserService();
29         userName = request.getParameter("txtUserName");
30         password = request.getParameter("txtPassword");
31         lastName = request.getParameter("txtLastName");
32         String admin = request.getParameter("chkIsAdmin");
33         UserError userError = new UserError();
34         if (userName.matches("U\\d{3}") == false) {
35             userError.setNameError("The UserName must be formatted Uxxx, x is digits.");
36             isError = true;
37         } //end check userName
38         if (password.matches("(.){3,15}") == false) {
39             userError.setPasswordError("The Password must be 3 to 15 characters.");
40             isError = true;
41         } //end check Password
42         if (lastName.matches("(.){5,50}") == false) {
43             userError.setLastNameError("The LastName must be 5 to 50 characters.");
44             isError = true;
45         } //end check LastName
46         if (isError == false) {
47             if (admin == null) {
48                 isAdmin = false;
49             }
50             User user = new User(userName, password, lastName, isAdmin);
51             if (userService.updateUser(user) == true) {
52                 message = "<b style='color: green'>The user has been updated successfully.</b>";
53             }
54             else{
55                 message = "<b style='color: red'>Something went wrong.</b>";
56             }
57         } //end isError
58         else {
59             request.setAttribute("ErrorDetails", userError);
60         }
61         url = userController+"?action=Details&UserName="+userName;
62     } catch (Exception ex) {
63         log(ex.getMessage());
64     }
65     finally{
66         request.setAttribute("message",message);
67         RequestDispatcher rd = request.getRequestDispatcher(url);
68         rd.forward(request, response);
69     }
70 } //processRequest
71 //HttpServlet methods. Click on the + sign on the left to edit the code.
109 } //end UpdateController

```

• UserDetailsController.java



```

13  /**
14   * @author SwordLake
15   */
16  @WebServlet(name = "UserDetailsController", urlPatterns = {"/UserDetailsController"})
17  public class UserDetailsController extends HttpServlet {
18      private final String userDetailsPage = "UserDetails.jsp";
19      private final String userController = "UserController";
20      protected void processRequest(HttpServletRequest request,
21      HttpServletResponse response) throws ServletException, IOException {
22          response.setContentType("text/html;charset=UTF-8");
23          String url = userDetailsPage;
24          User user;

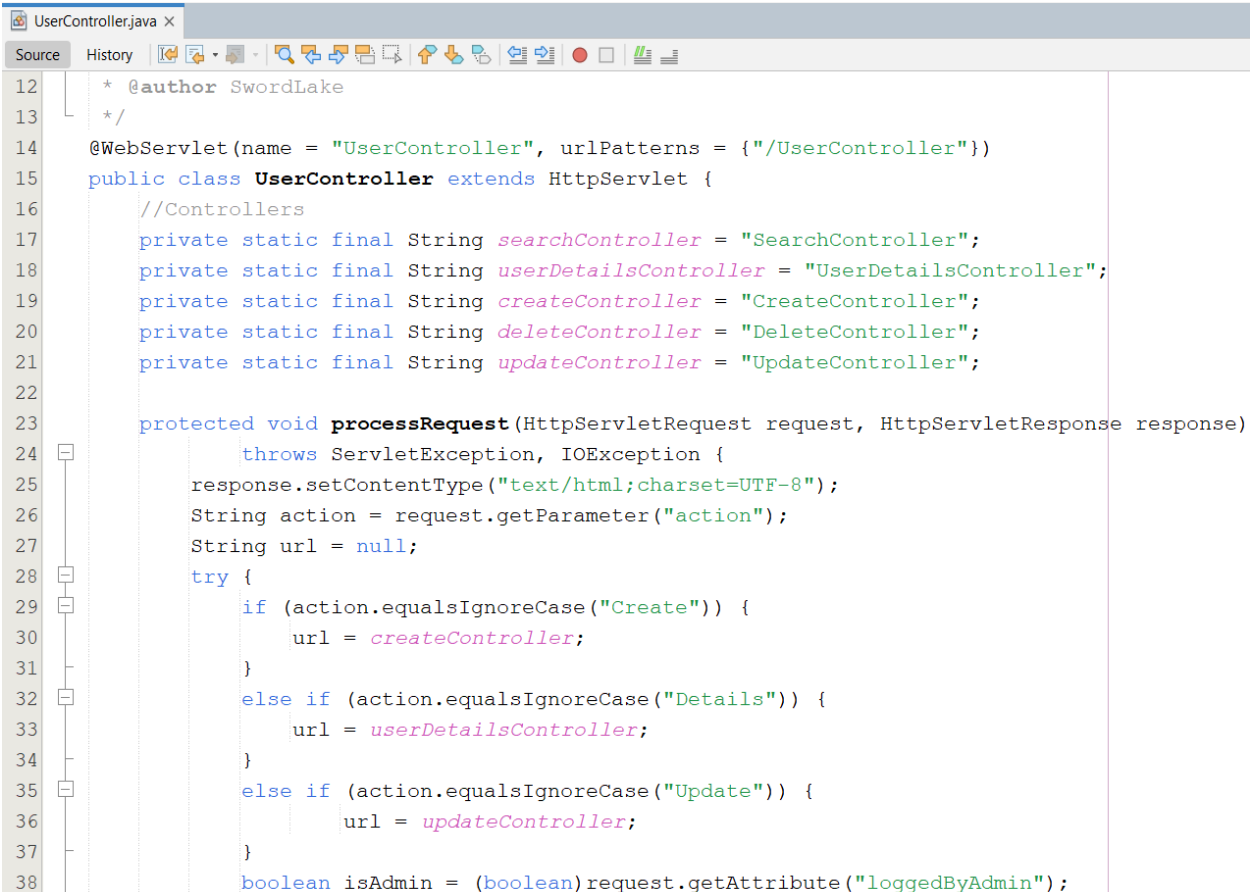
```

```

25     try {
26         String userName = request.getParameter("UserName");
27         UserService userService = new UserService();
28         if (!userName.isEmpty()) {
29             user = userService.getUserByUserName(userName);
30             if (user != null) {
31                 request.setAttribute("userDetails", user);
32             }
33             else {
34                 url = userController + "?action=Search";
35             }
36         }
37     } catch (Exception ex) {
38         log(ex.getMessage());
39     }
40     finally {
41         RequestDispatcher rd = request.getRequestDispatcher(url);
42         rd.forward(request, response);
43     }
44 } //processRequest
45 //HttpServlet methods. Click on the + sign on the left to edit the code.
83 } //end UserDetailsController

```

• UserController.java



```

UserController.java x
Source History
12  * @author SwordLake
13  */
14  @WebServlet(name = "UserController", urlPatterns = {"/UserController"})
15  public class UserController extends HttpServlet {
16      //Controllers
17      private static final String searchController = "SearchController";
18      private static final String userDetailsController = "UserDetailsController";
19      private static final String createController = "CreateController";
20      private static final String deleteController = "DeleteController";
21      private static final String updateController = "UpdateController";
22
23      protected void processRequest(HttpServletRequest request, HttpServletResponse response)
24          throws ServletException, IOException {
25          response.setContentType("text/html;charset=UTF-8");
26          String action = request.getParameter("action");
27          String url = null;
28          try {
29              if (action.equalsIgnoreCase("Create")) {
30                  url = createController;
31              }
32              else if (action.equalsIgnoreCase("Details")) {
33                  url = userDetailsController;
34              }
35              else if (action.equalsIgnoreCase("Update")) {
36                  url = updateController;
37              }
38              boolean isAdmin = (boolean) request.getAttribute("loggedByAdmin");

```

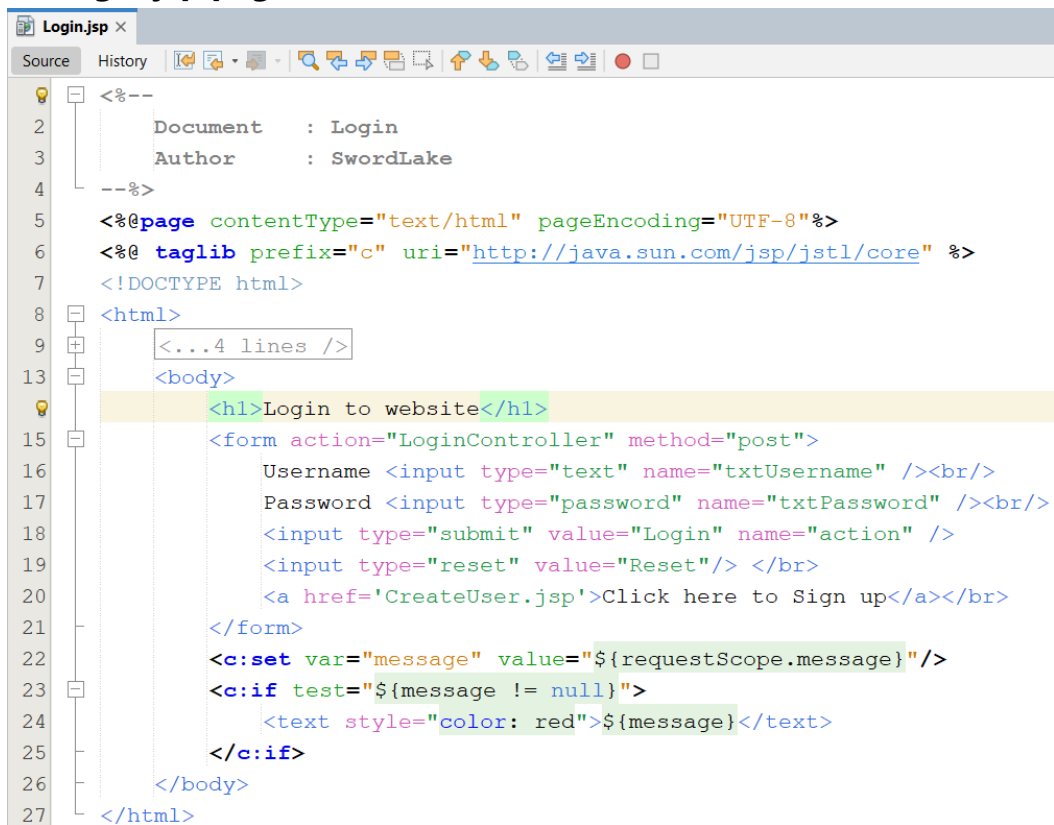
```

39         if(isAdmin){
40             if (action.equalsIgnoreCase("Delete")) {
41                 url = deleteController;
42             } else if (action.equalsIgnoreCase("Search")) {
43                 url = searchController;
44             }
45         }
46     } finally {
47         RequestDispatcher rd = request.getRequestDispatcher(url);
48         rd.forward(request, response);
49     }
50 } //end processRequest
51 //end UserController
90

```

Step 05. Write codes for the web pages as follows:

- **Login.jsp page**



```

1  <%--
2      Document    : Login
3      Author     : SwordLake
4  --%>
5  <%@page contentType="text/html" pageEncoding="UTF-8"%>
6  <%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
7  <!DOCTYPE html>
8  <html>
9      <...4 lines />
13 <body>
14     <h1>Login to website</h1>
15     <form action="LoginController" method="post">
16         Username <input type="text" name="txtUsername" /><br/>
17         Password <input type="password" name="txtPassword" /><br/>
18         <input type="submit" value="Login" name="action" />
19         <input type="reset" value="Reset" /> </br>
20         <a href='CreateUser.jsp'>Click here to Sign up</a><br>
21     </form>
22     <c:set var="message" value="{requestScope.message}" />
23     <c:if test="{message != null}">
24         <text style="color: red">{message}</text>
25     </c:if>
26 </body>
27 </html>

```

- **CreateUser.jsp page**

```

CreateUser.jsp x
Source History
7 <%@page contentType="text/html" pageEncoding="UTF-8"%>
8 <%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
9 <!DOCTYPE html>
10 <html>
11 <...4 lines />
12 <body>
13 <h1>Create User</h1>
14 <c:set var="message" value="${requestScope.message}"/>
15 <c:if test="${message != null}">
16     ${message}
17 </c:if>
18 <form action="UserController" method="post">
19 <c:set var="error" value="${requestScope.errorDetails}"/>
20 <c:if test="${not empty error.userNameError}">
21     <text style="color: red">${error.userNameError}</text>
22 </c:if>
23 <c:if test="${not empty error.duplicateUserName}">
24     <text style="color: red">${error.duplicateUserName}</text>
25 </c:if><br/>
26 Password <input type="password" placeholder="Enter password" name="txtPassword" />
27 <c:if test="${not empty error.passwordError}">
28     <text style="color: red">${error.passwordError}</text>
29 </c:if><br/>
30 LastName <input type="text" placeholder="Enter last name" name="txtLastName" />
31 <c:if test="${not empty error.lastNameError}">
32     <text style="color: red">${error.lastNameError}</text>
33 </c:if><br/>
34 <input type="checkbox" name="chkIsAdmin" disabled="true" /> isAdmin<br/>
35 <input type="submit" value="Create" name="action" /></br>
36 <a href='Login.jsp'>Back</a></br>
37 </form>
38 </body>
39 </html>

```

• Search.jsp page

```

Search.jsp x
Source History
1 <!--
2     Author      : SwordLake
3 -->
4 <%@page import="Models.DTO.User"%>
5 <%@page import="java.util.List"%>
6 <%@page contentType="text/html" pageEncoding="UTF-8"%>
7 <%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
8 <!DOCTYPE html>
9 <html>
10 <...4 lines />
11 <body>
12 <c:set var="userLoggedIn" value="${userLoggedIn}"/>
13 <c:if test="${userLoggedIn!=null}">
14     <c:set var="lastName" value="${userLoggedIn.lastName}"/>
15 </c:if>
16 <c:set var="searchValue" value="${param.txtSearchValue}"/>
17 <h3>Welcome to <text style="color: red">${lastName}</text></h3>

```

```

21 <form method="post">
22     <input type="submit" formaction="LogoutController" value="Logout"/></br>
23 </form>
24 <h1>Search user by last name</h1>
25 <form action="UserController" method="post">
26     Enter search value
27     <input type="text" name="txtSearchValue"
28         value="{searchValue!=null?searchValue:''}" /><br/>
29     <input type="submit" value="Search" name="action" />
30 </form>
31 <c:set var="userList" value="{requestScope.SearchResult}"/>
32 <c:set var="count" value="1"/>
33 <c:if test="{userList != null}">
34 <table border="1">
35 <thead>
36 <tr>
37 <th>No.</th>
38 <th>UserName</th>
39 <th>Password</th>
40 <th>LastName</th>
41 <th>Role</th>
42 <th colspan="2">Action</th>
43 </tr>
44 </thead>
45 <tbody>
46 <c:forEach var="user" items="{userList}">
47 <tr>
48 <td>{count}</td>
49 <td>{user.userName}</td>
50 <td>{user.password}</td>
51 <td>{user.lastName}</td>
52 <td><input type="checkbox" <c:if test="{user.isAdmin==true}">checked</c:if>/></td>
53 <td><a href="UserController?action=Delete&UserName={user.userName}&txtSearchValue={searchValue}">
54     Delete
55 </td>
56 <td><a href="UserController?action=Details&UserName={user.userName}&txtSearchValue={searchValue}">
57     View
58 </td>
59 </tr>
60 <c:set var="count" value="{count+1}"/>
61 </c:forEach>
62 </tbody>
63 </table>
64 <c:set var="message" value="{requestScope.message}"/>
65 <c:if test="{message != null}">
66     {message}
67 </c:if>
68 <h3>Number of users:{userList.size()}</h3>
69 </c:if>
70 <c:if test="{searchValue!= null}">
71 <c:if test="{userList.size()==0 || userList== null}">
72 <h3>No users were found.</h3>
73 </c:if>
74 </c:if>
75 </body>
76 </html>
  
```


• UserDetails.jsp page

```

UserDetails.jsp x
Source History
3      Author      : SwordLake
4      --%>
5      <%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
6      <%@page import="Models.DTO.User"%>
7      <%@page contentType="text/html" pageEncoding="UTF-8"%>
8      <!DOCTYPE html>
9      <html>
10     <...4 lines />
11     <body>
12         <form method="post">
13             <input type="submit" formaction="LogoutController" value="Logout"/></br>
14         </form>
15
16         <c:set var="userLoggedIn" value="${userLoggedIn}"/>
17         <c:if test="${userLoggedIn!=null}">
18             <c:set var="lastName" value="${userLoggedIn.lastName}"/>
19         </c:if>
20         <c:set var="searchValue" value="${param.txtSearchValue}"/>
21         <c:set var="user" value="${requestScope.userDetails}"/>
22         <c:if test = "${userDetails != null}">
23             <h1>User Details</h1>
24             <c:set var="message" value="${requestScope.message}"/>
25             <c:if test="${message != null}">
26                 ${message}
27             </c:if>
28             <form action="UserController" method="post">
29                 <c:set var="error" value="${requestScope.ErrorDetails}"/>
30                 UserName <input type="text" readonly="true" name="txtUserName" value='${user.userName}' />
31                 <c:if test="${not empty error.userNameError}">
32                     <text style="color: red">${error.userNameError}</text>
33                 </c:if><br/>
34                 Password <input type="password" name="txtPassword" value='${user.password}' />
35                 <c:if test="${not empty error.passwordError}">
36                     <text style="color: red">${error.passwordError}</text>
37                 </c:if><br/>
38                 LastName <input type="text" name="txtLastName" value='${user.lastName}' />
39                 <c:if test="${not empty error.lastNameError}">
40                     <text style="color: red">${error.lastNameError}</text>
41                 </c:if><br/>
42                 <input type="hidden" value='${searchValue}' name="txtSearchValue"/>
43                 <input type="checkbox" name="chkIsAdmin"
44                     <c:if test="${user.isAdmin==true}">checked</c:if>
45                     <c:if test="${userLoggedIn.isAdmin==false}">disabled</c:if>>isAdmin
46                 </input></br>
47                 <input type="submit" value="Update" name="action" /></br>
48             </form>
49             <c:if test="${userLoggedIn.isAdmin}" >
50                 <a href='UserController?action=Search&txtSearchValue=${searchValue}'>Back</a></br>
51             </c:if>
52             <c:if test="${!userLoggedIn.isAdmin}" >
53                 <a href='Login.jsp'>Back</a></br>
54             </c:if>
55         </c:if>
56     </body>
57 </html>

```

Step 06. Right-click on **Lab07_ManageUsers_JSTL_Filter** project, select Run to run the web app then test all functions.