UNIVERSITI MALAYSIA TERENGGANU

CSM3023 WEB-BASED APPLICATION DEVELOPMENT (K1)

BACHELOR OF COMPUTER SCIENCE (MOBILE COMPUTING) WITH HONORS

LAB 2 - SERVLET: DATA SHARING AND DATABASE MANAGEMENT

SEMESTER II 2023/2024

Prepared for:

DR. MOHAMAD NOR HASSAN

Prepared by:

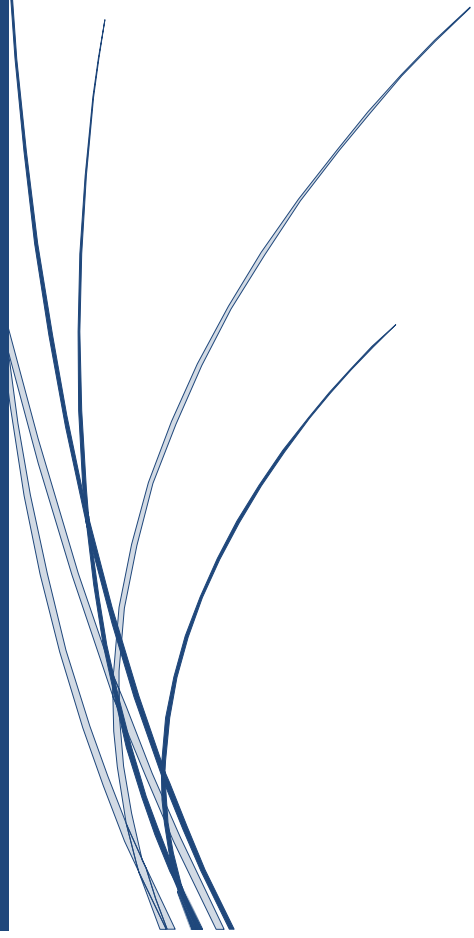
LUQMAN HAKIM BIN AZIZ

(S66292)

Week 2

Servlet: Data Sharing and Database Management

Web Programming 2



Lecturers

PUSAT PENGAJIAN INFORMATIK DAN MATEMATIK GUNAAN (PPIMG), UNIVERSITI MALAYSIA TERENGGANU (UMT)

Table of Contents

Task 1: Data Sharing in Servlet

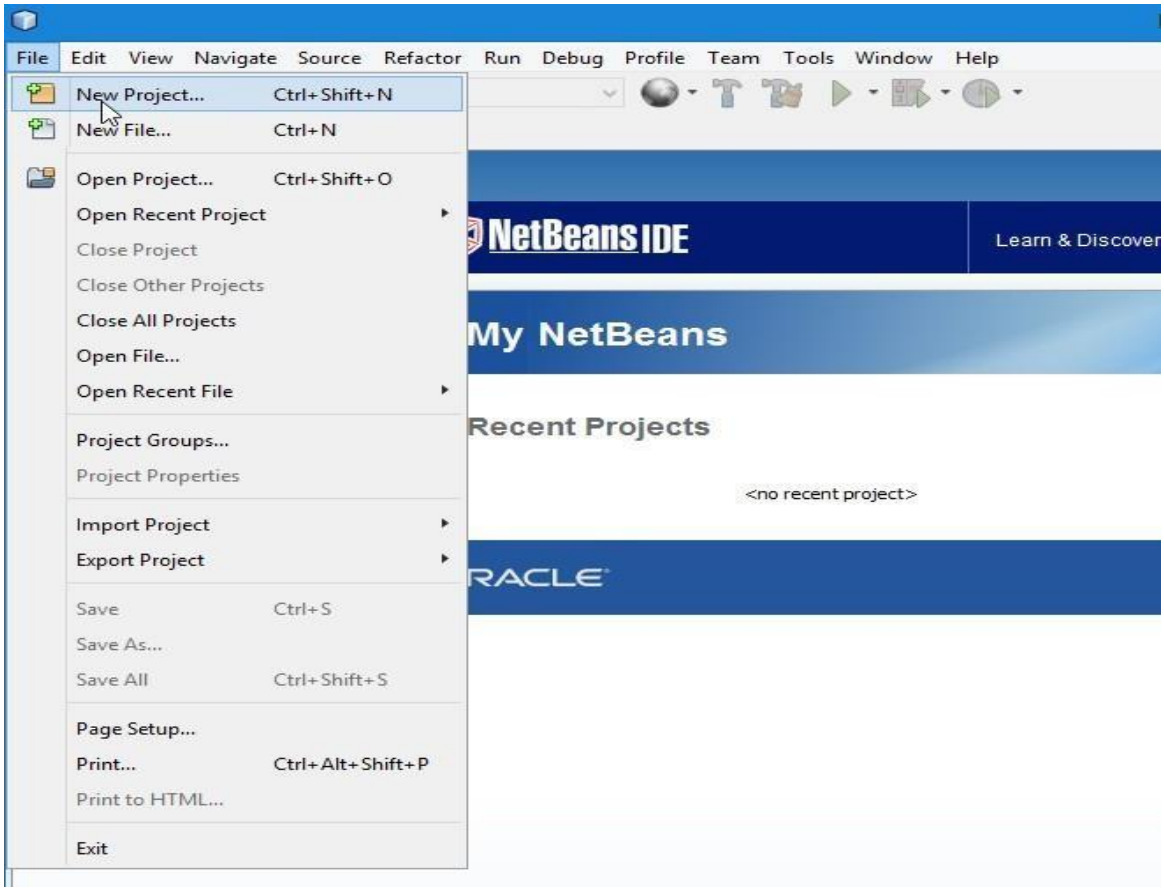
Objective:	To use servlet for request forwarding and data sharing
Problem Description:	Write a login form and a servlet to authenticate a user.
Estimated time:	30 minutes

1. Create a directory F:\CSF3107-SXXXXX (only if you have not created it yet). Note: Replace *XXXXX* with your matric number.

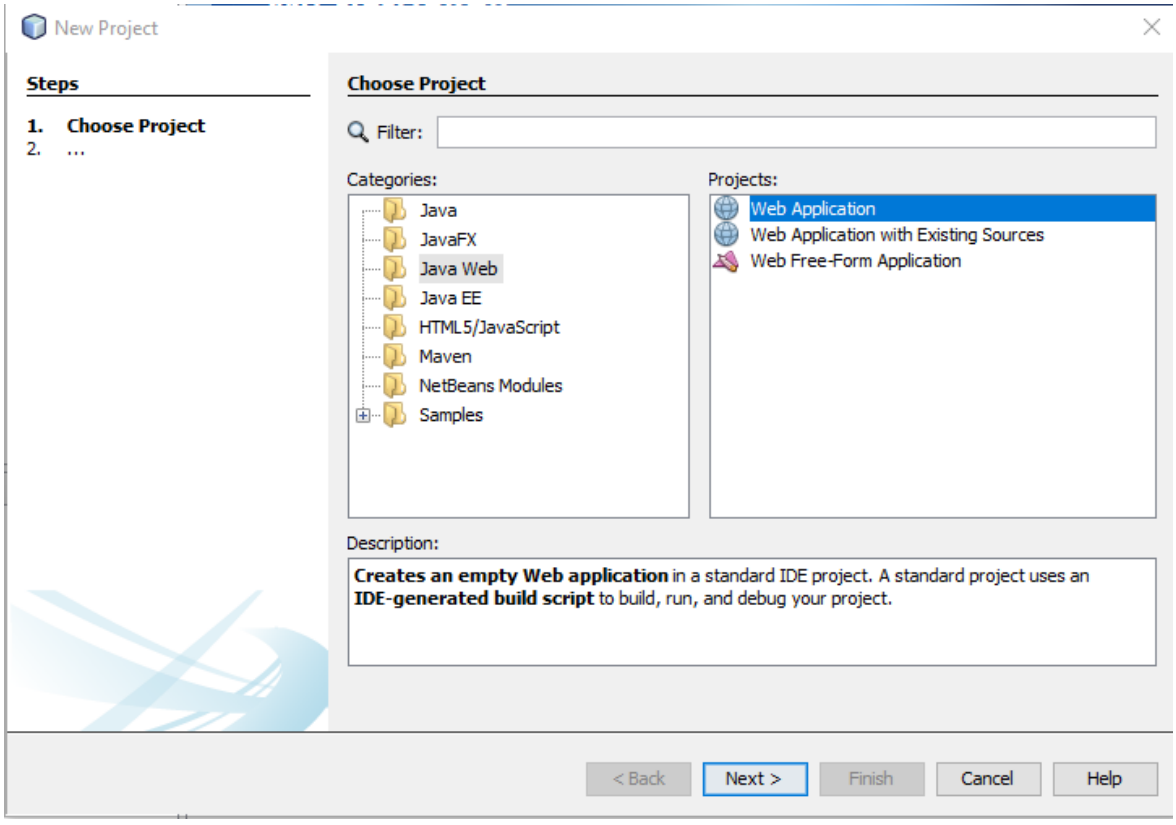
2. Go to F:\CSF3107-SXXXXX\ directory and create sub-directory and name it as *Lab 2*.

3. Open your NetBeans IDE.

4. Go to File -> New Project

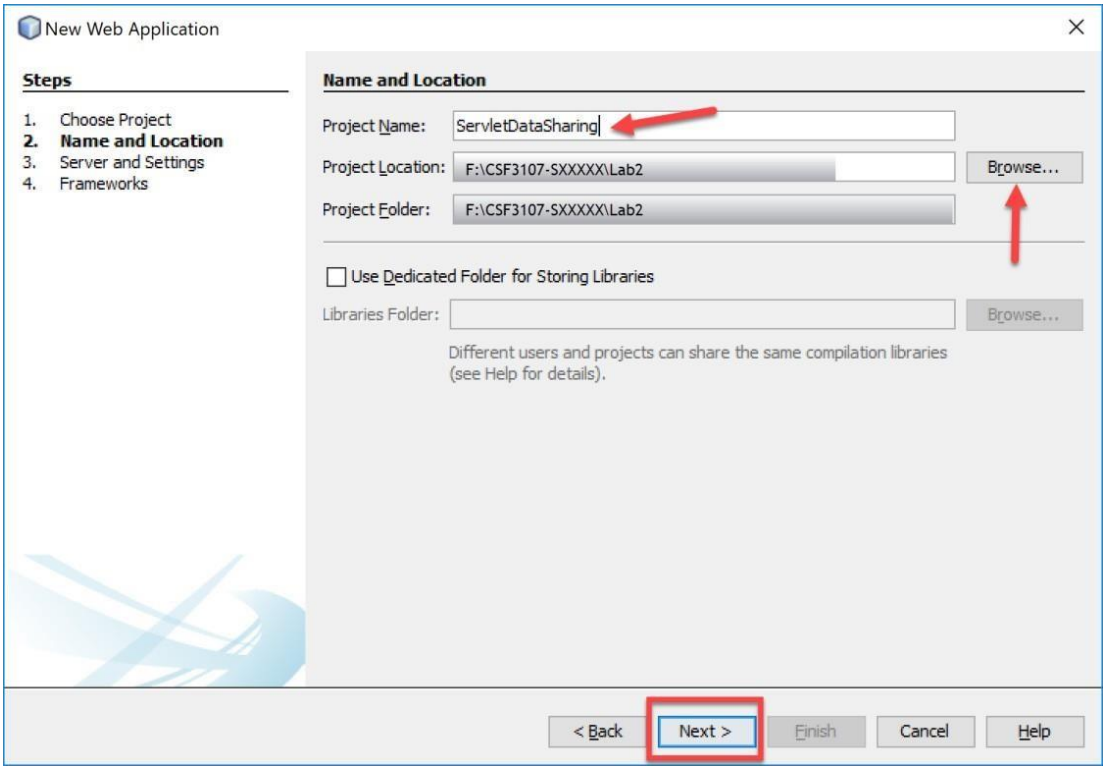


5. Select Java Web -> Web Application and click Next.

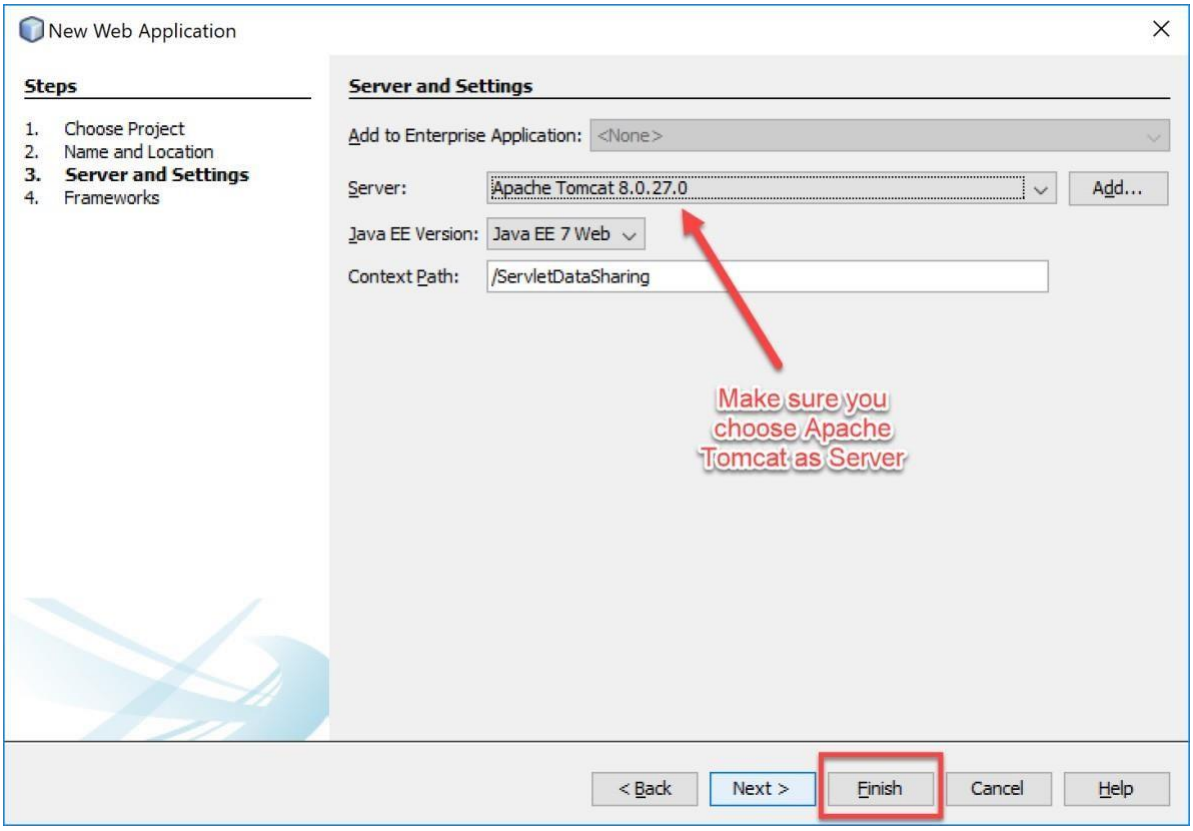


6. Type Project Name: *ServletDataSharing*.

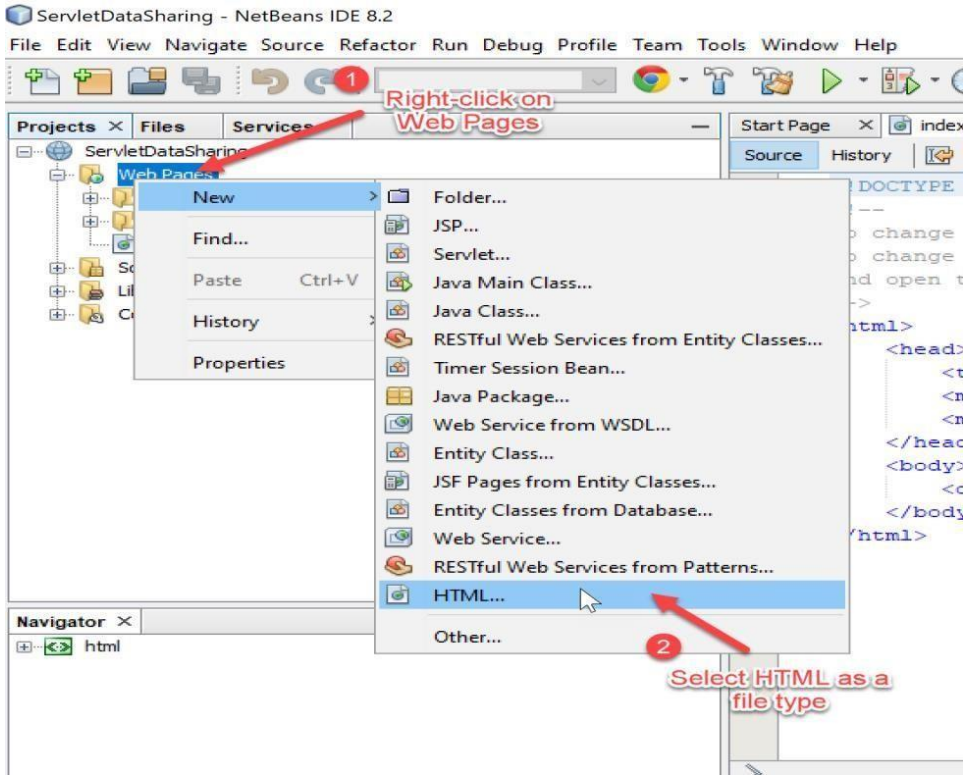
7. Click Browse and choose Project Location: F:\CSF3107-SXXXXX\Lab2. Then click the Next button.



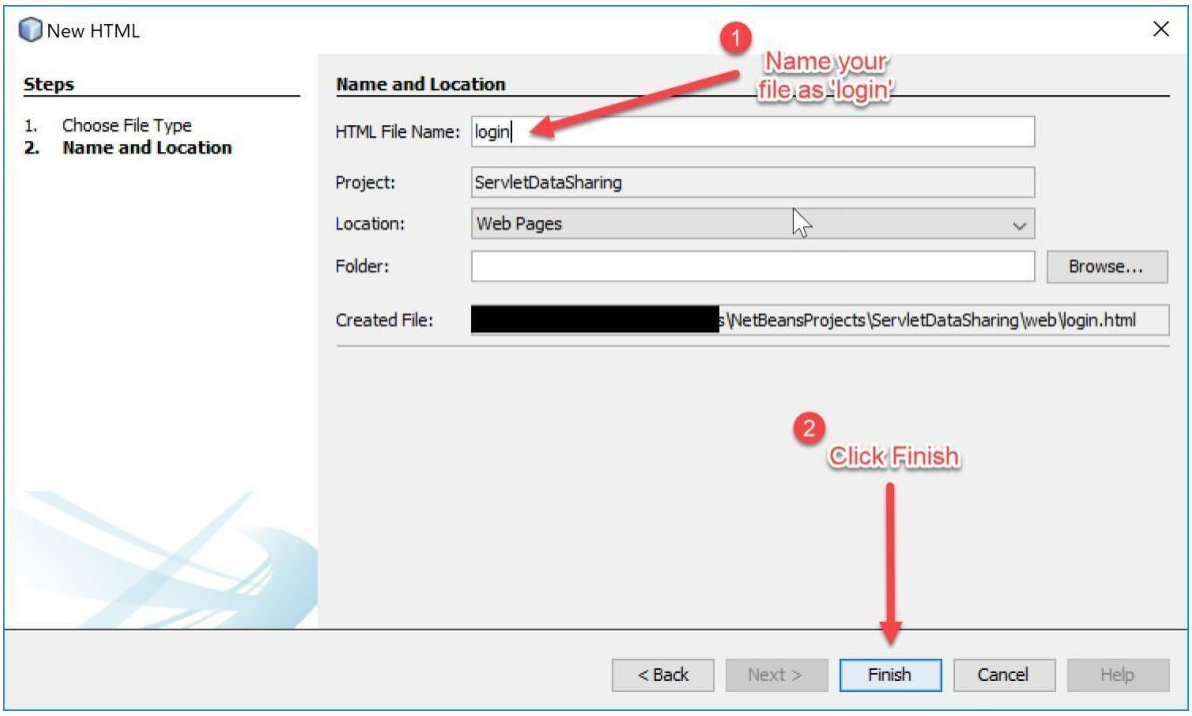
8. Then, choose Apache Tomcat as the server and click Finish.



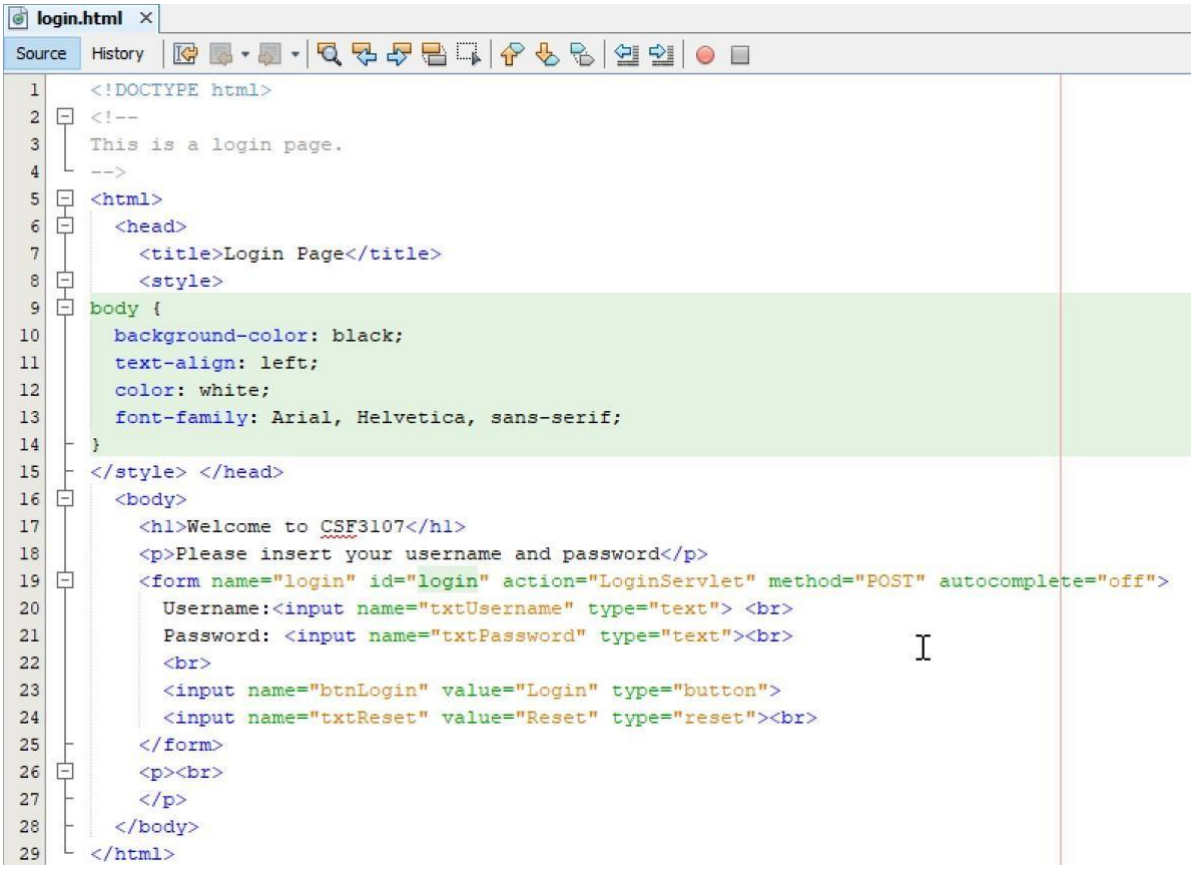
9. You will see your project structure on the left side of your Netbeans editor. Next, you will create an HTML file for a login page. Follow the instructions as shown on the following screenshot.



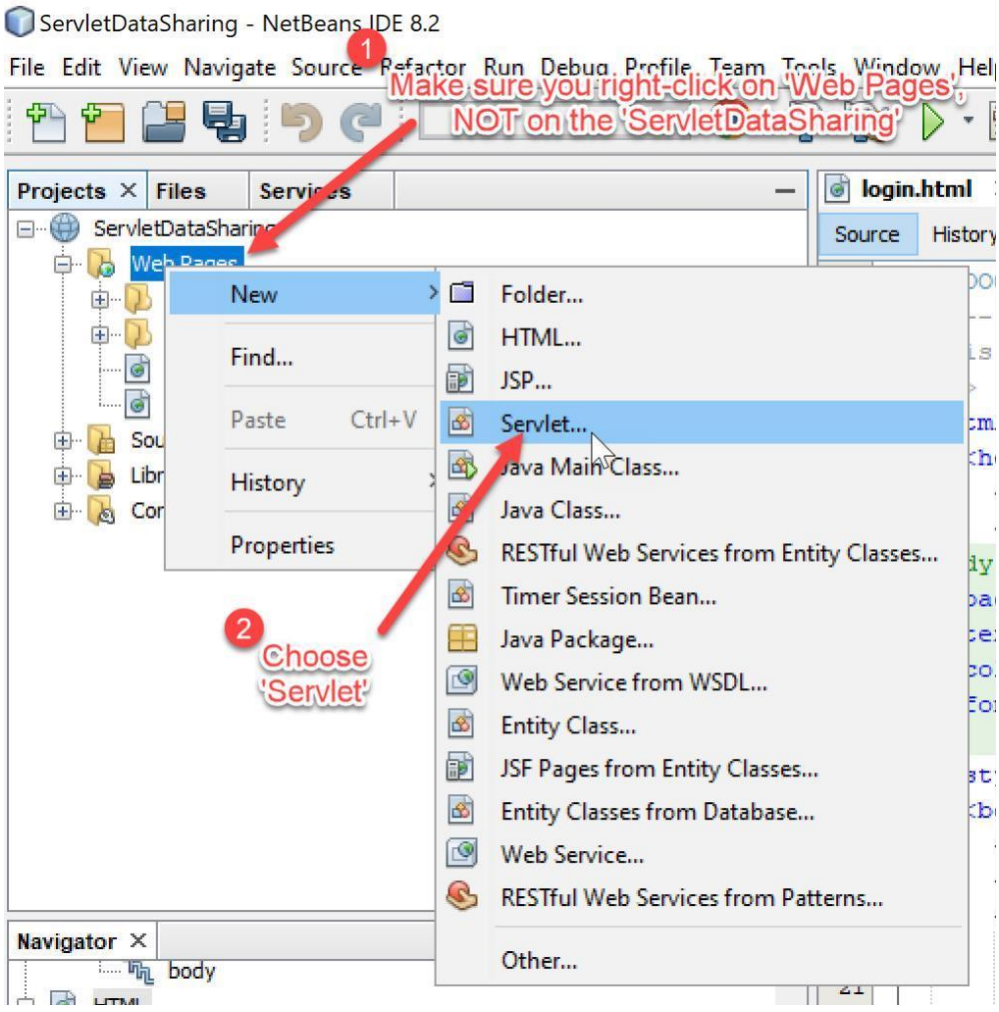
10. Name your HTML file as ‘login’, then click Finish.

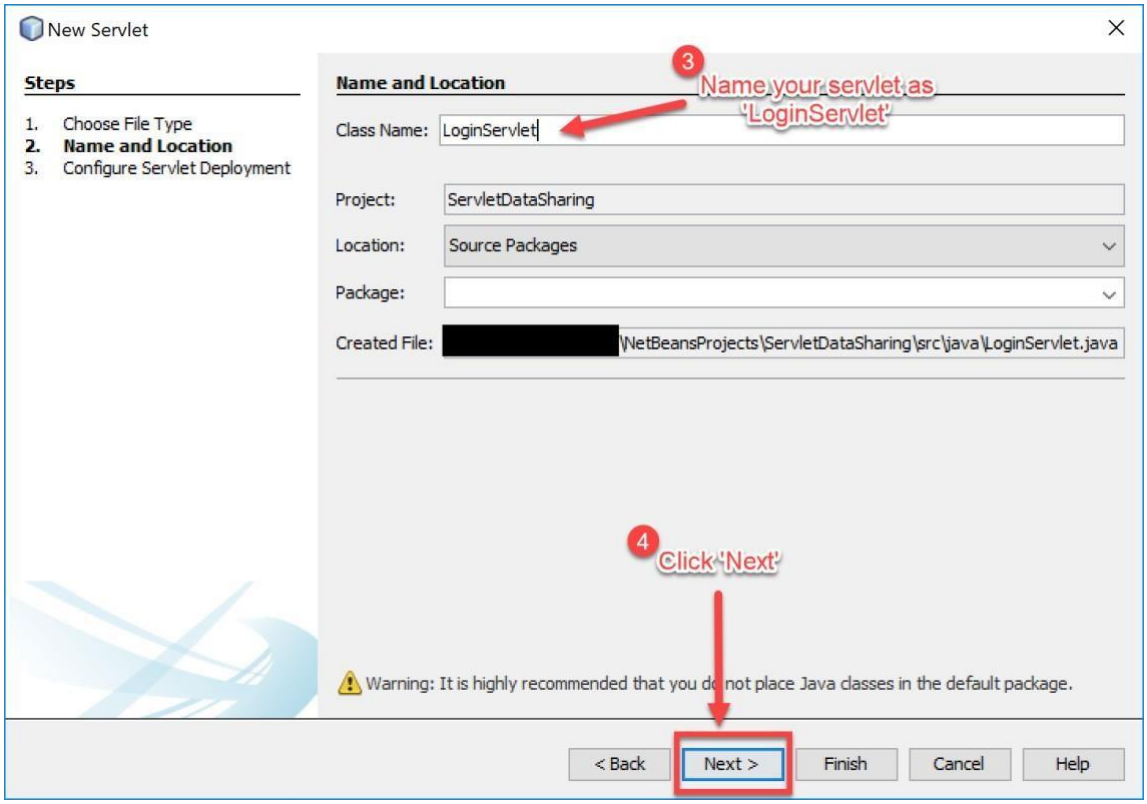


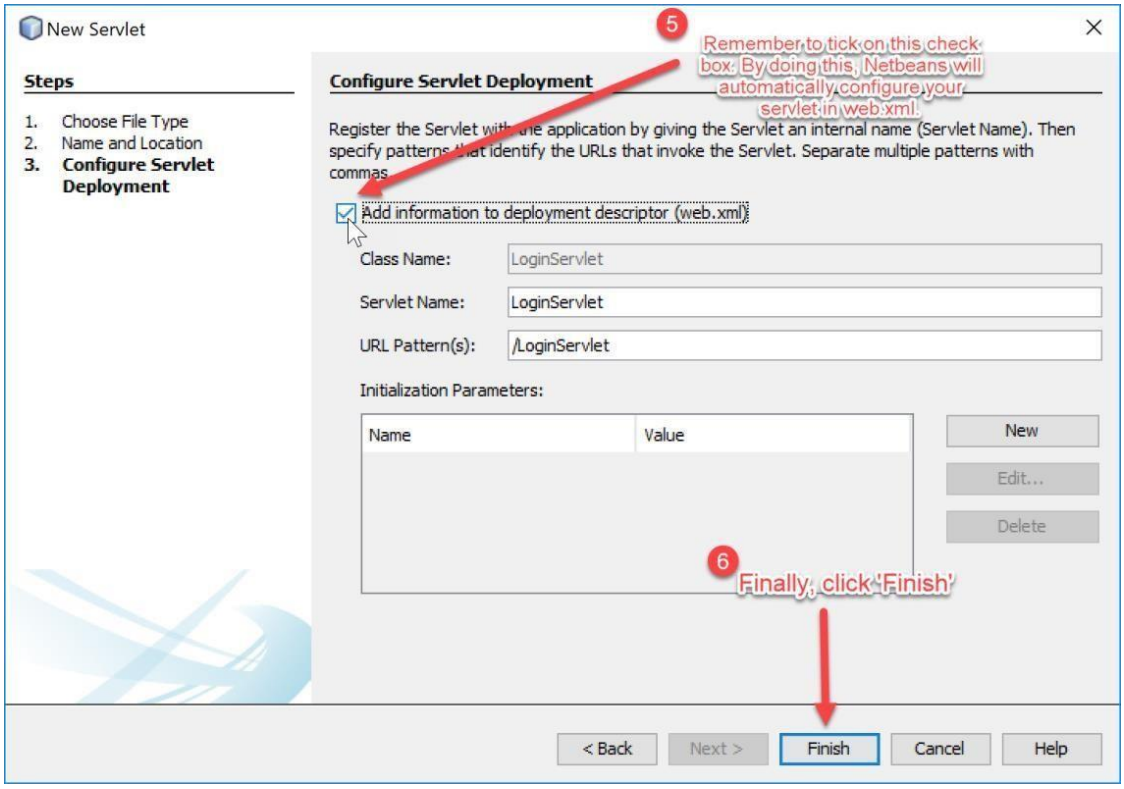
11. Type the following HTML markups into your login.html.

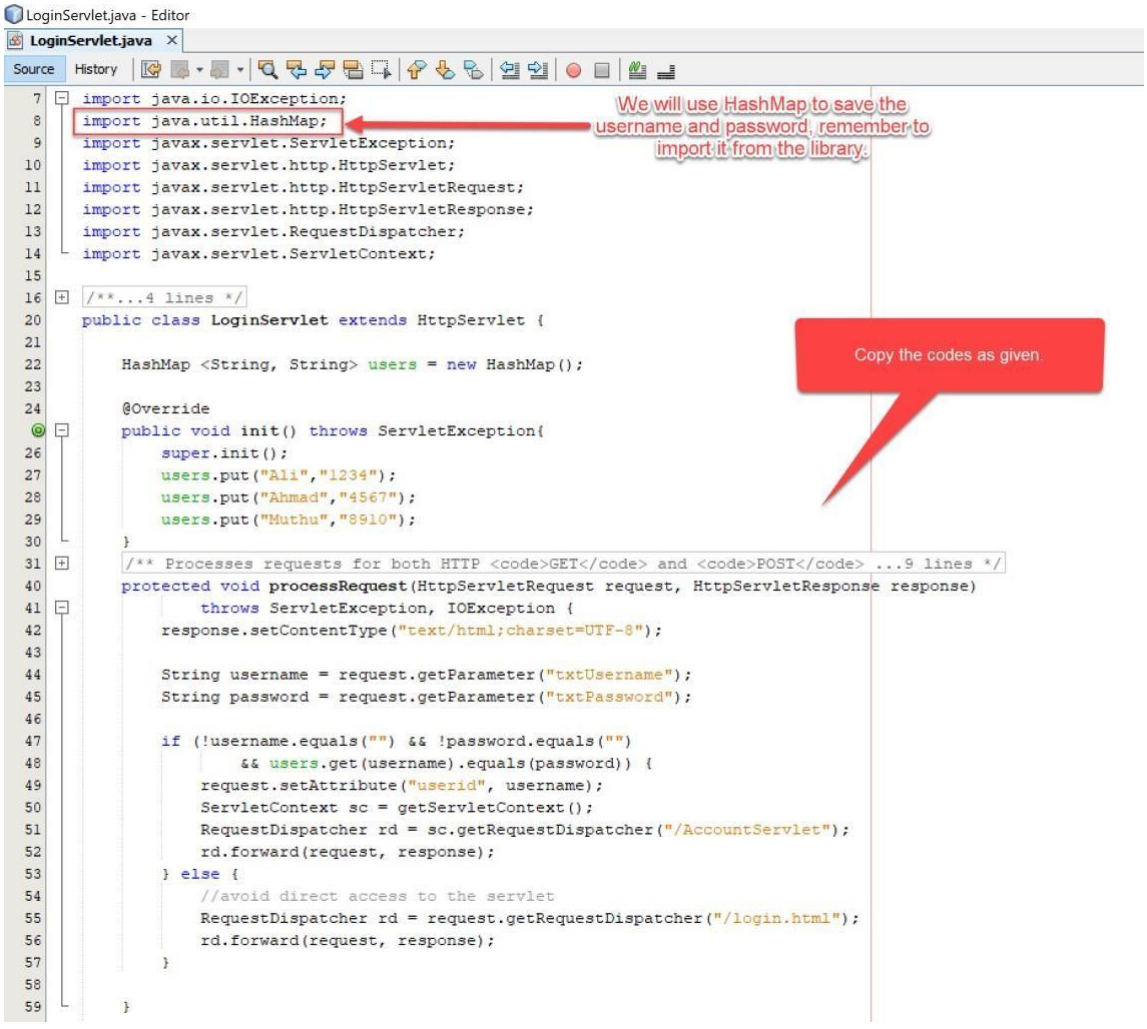


12. Next, we are going to create a servlet to process the username and password insert by the user on the login form. We start it by creating a new file for our servlet. Follow the steps below:



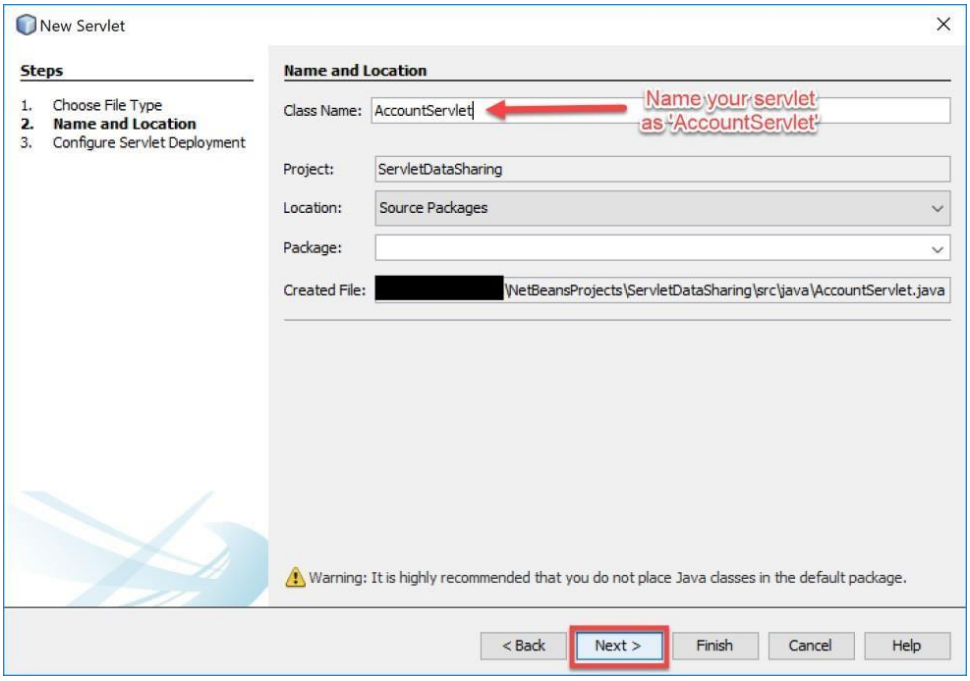

New Servlet
Steps
1. Choose File Type
2. Name and Location
3. Configure Servlet Deployment
Name and Location
3
Name your servlet as 'LoginServlet'
Class Name: LoginServlet
Project: ServletDataSharing
Location: Source Packages
Package:
Created File: \NetBeansProjects\ServletDataSharing\src\java\LoginServlet.java
4
Click 'Next'
Warning: It is highly recommended that you do not place Java classes in the default package.
< Back
Next >
Finish
Cancel
Help

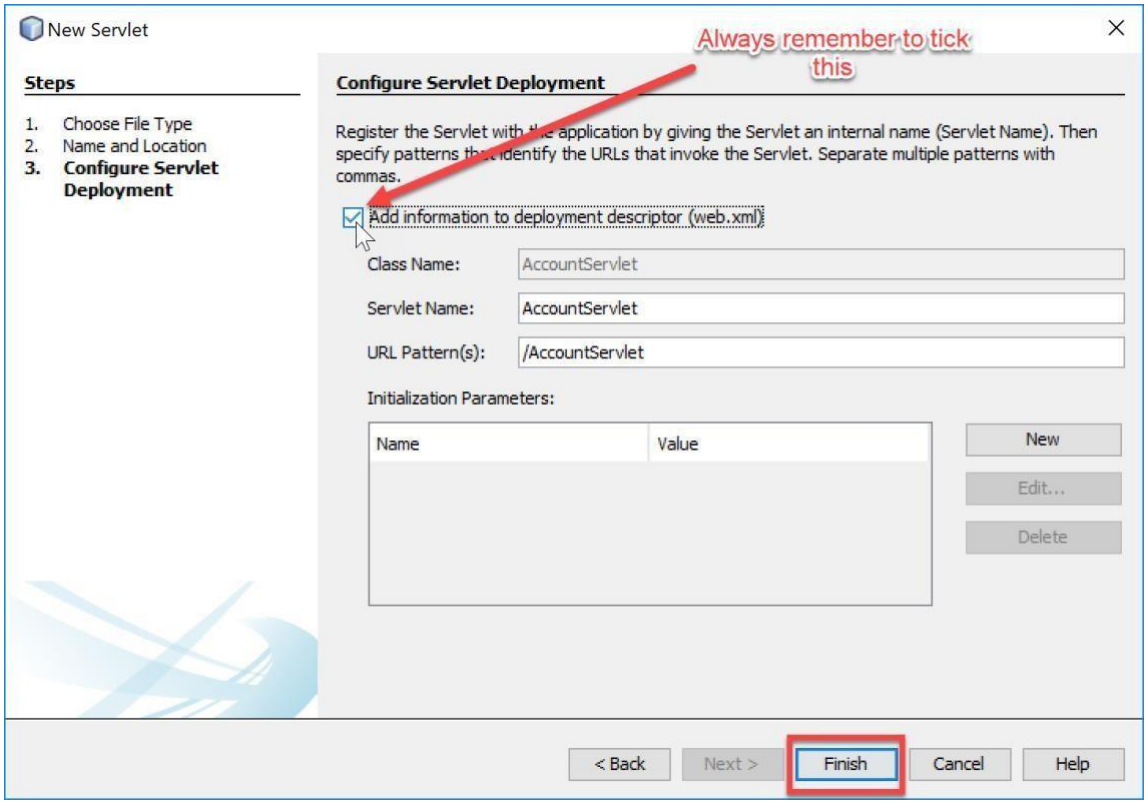

New Servlet
Steps
1. Choose File Type
2. Name and Location
3. Configure Servlet Deployment
5
Remember to tick on this check box. By doing this, Netbeans will automatically configure your servlet in web.xml
Configure Servlet Deployment
Register the Servlet with the application by giving the Servlet an internal name (Servlet Name). Then specify patterns that identify the URLs that invoke the Servlet. Separate multiple patterns with commas.
Add information to deployment descriptor (web.xml)
Class Name: LoginServlet
Servlet Name: LoginServlet
URL Pattern(s): /LoginServlet
Initialization Parameters:
Name
Value
New
Edit...
Delete
6
Finally, click 'Finish'
< Back
Next >
Finish
Cancel
Help

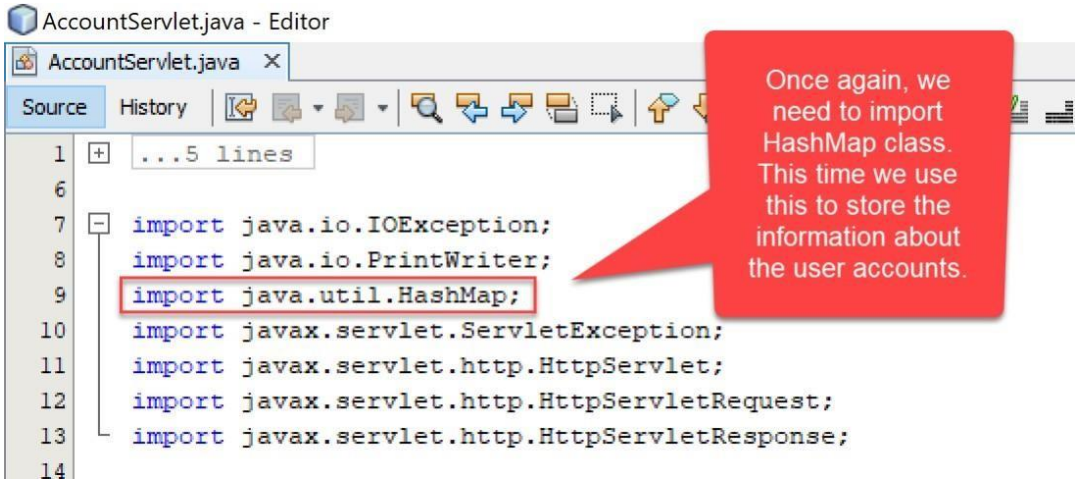


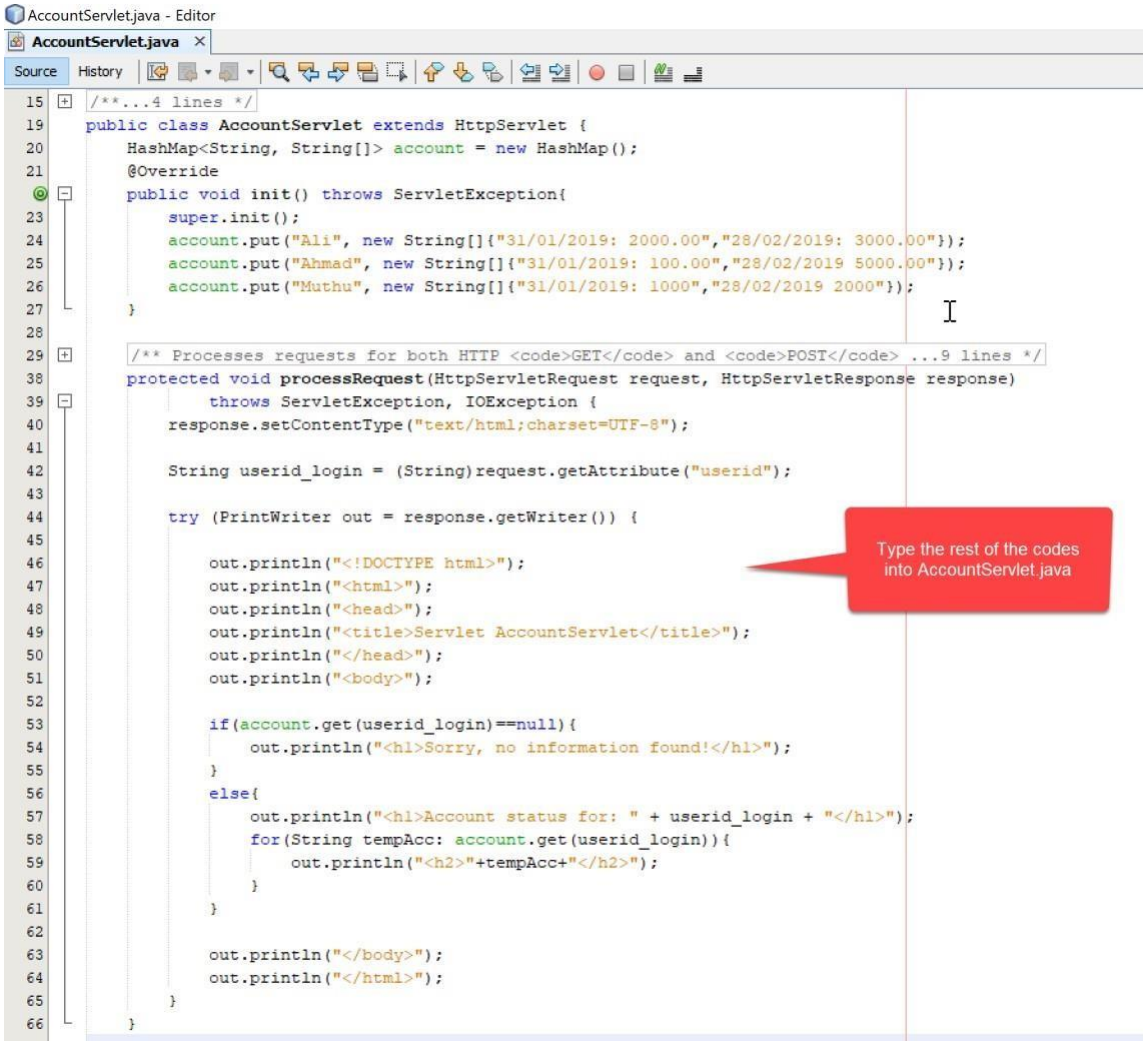

```
import java.io.IOException;
import java.util.HashMap;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.RequestDispatcher;
import javax.servlet.ServletContext;

/**...4 lines */
public class LoginServlet extends HttpServlet {

    HashMap <String, String> users = new HashMap();

    @Override
    public void init() throws ServletException{
        super.init();
        users.put("Ali","1234");
        users.put("Ahmad","4567");
        users.put("Muthu","8910");
    }
    /** Processes requests for both HTTP <code>GET</code> and <code>POST</code> ...9 lines */
    protected void processRequest(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
        response.setContentType("text/html;charset=UTF-8");

        String username = request.getParameter("txtUsername");
        String password = request.getParameter("txtPassword");

        if (!username.equals("") && !password.equals("")
                && users.get(username).equals(password)) {
            request.setAttribute("userid", username);
            ServletContext sc = getServletContext();
            RequestDispatcher rd = sc.getRequestDispatcher("/AccountServlet");
            rd.forward(request, response);
        } else {
            //avoid direct access to the servlet
            RequestDispatcher rd = request.getRequestDispatcher("/login.html");
            rd.forward(request, response);
        }

    }
```

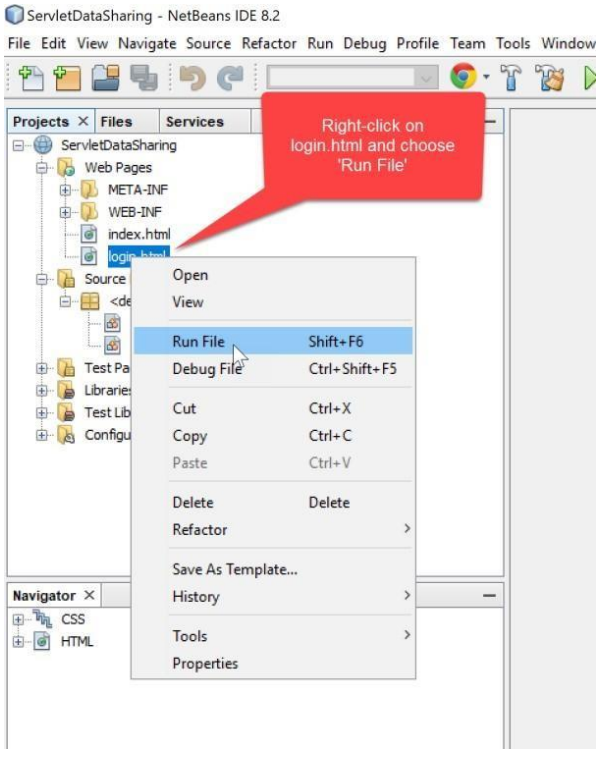
13. Next, we need a servlet to return the information about users' accounts and display it on the browser. Repeat previous step for creating a new servlet and fill the details as shown on the screenshots below.



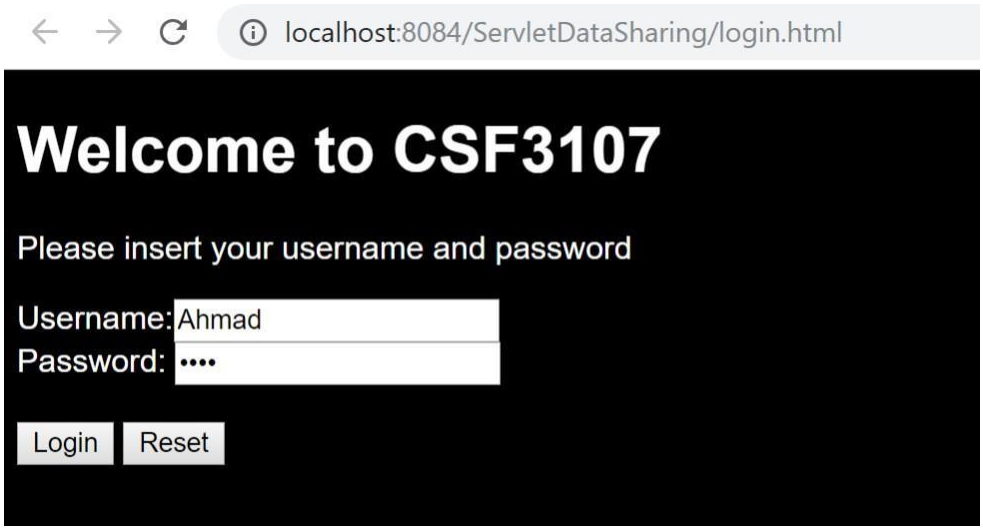

New Servlet
Always remember to tick this
Steps
1. Choose File Type
2. Name and Location
3. Configure Servlet Deployment
Configure Servlet Deployment
Register the Servlet with the application by giving the Servlet an internal name (Servlet Name). Then specify patterns that identify the URLs that invoke the Servlet. Separate multiple patterns with commas.
Add information to deployment descriptor (web.xml)
Class Name: AccountServlet
Servlet Name: AccountServlet
URL Pattern(s): /AccountServlet
Initialization Parameters:
Name
Value
New
Edit...
Delete
< Back
Next >
Finish
Cancel
Help


AccountServlet.java - Editor
AccountServlet.java
Source
History
...5 lines
import java.io.IOException;
import java.io.PrintWriter;
import java.util.HashMap;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
Once again, we need to import HashMap class. This time we use this to store the information about the user accounts.

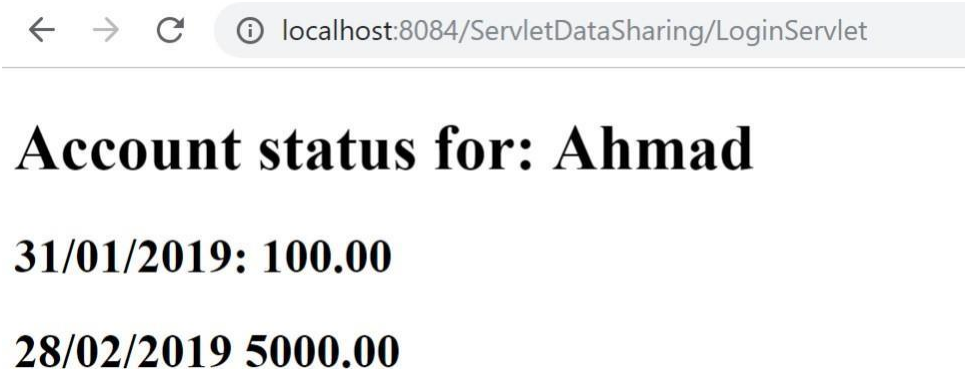

AccountServlet.java - Editor
AccountServlet.java
Source
History
/**...4 lines */
public class AccountServlet extends HttpServlet {
HashMap<String, String[]> account = new HashMap();
@Override
public void init() throws ServletException{
super.init();
account.put("Ali", new String[]{"31/01/2019: 2000.00","28/02/2019: 3000.00"});
account.put("Ahmad", new String[]{"31/01/2019: 100.00","28/02/2019 5000.00"});
account.put("Muthu", new String[]{"31/01/2019: 1000","28/02/2019 2000"});
}
/** Processes requests for both HTTP <code>GET</code> and <code>POST</code> ...9 lines */
protected void processRequest(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {
response.setContentType("text/html;charset=UTF-8");
String userid_login = (String)request.getAttribute("userid");
try (PrintWriter out = response.getWriter()) {
out.println("<!DOCTYPE html>");
out.println("<html>");
out.println("<head>");
out.println("<title>Servlet AccountServlet</title>");
out.println("</head>");
out.println("<body>");
if(account.get(userid_login)==null){
out.println("<h1>Sorry, no information found!</h1>");
}
else{
out.println("<h1>Account status for: " + userid_login + "</h1>");
for(String tempAcc: account.get(userid_login)){
out.println("<h2>"+tempAcc+"</h2>");
}
}
out.println("</body>");
out.println("</html>");
}
}
Type the rest of the codes into AccountServlet.java


ServletDataSharing - NetBeans IDE 8.2
File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window
Projects
Files
Services
ServletDataSharing
Web Pages
META-INF
WEB-INF
index.html
login.html
Source
Right-click on login.html and choose 'Run File'
Open
View
Run File Shift+F6
Debug File Ctrl+Shift+F5
Cut Ctrl+X
Copy Ctrl+C
Paste Ctrl+V
Delete Delete
Refactor
Save As Template...
History
Tools
Properties
Navigator
CSS
HTML

14. The Output will appear in a web browser. Put a Username as ‘Ahmad’ and Password as ‘4567’, then click *Login*.



15. If you have followed all the previous steps correctly, you will see the account information of a user named ‘Ahmad’. Repeat step 14 with other users as input and see the result.



Reflections:

1. What have you you learnt from this exercise?

- Servlets have a initialization (init()), handling requests (doGet(), doPost()), and finally being destroyed (destroy()). This knowledge helps in managing resources and performing necessary actions at each stage of the servlet's life.

2. What are the common methods used in Java Servlet?

- void init(): This method is called by the servlet container to initialize the servlet.

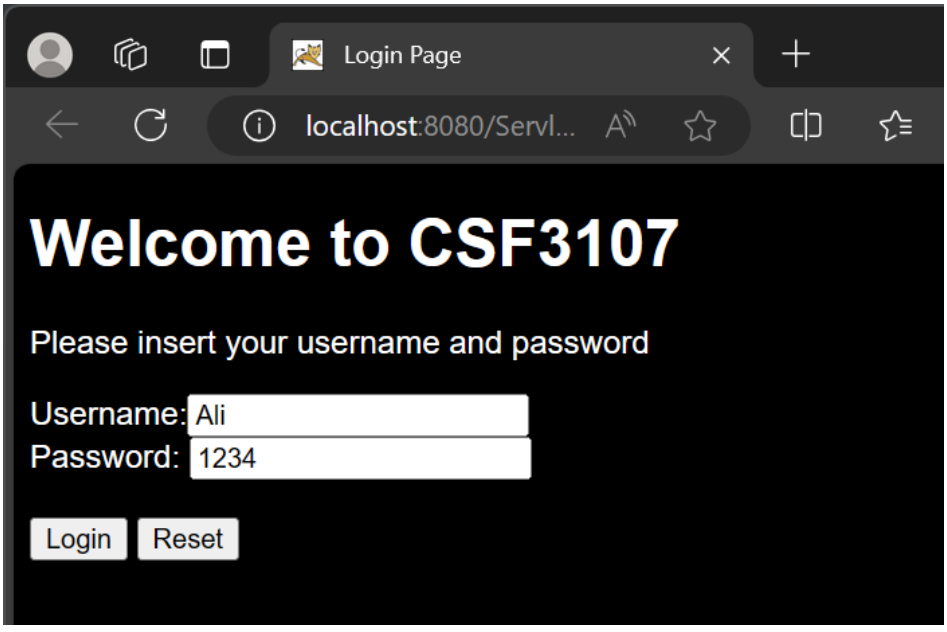
- void doGet()This method is invoked by the servlet container to handle HTTP GET

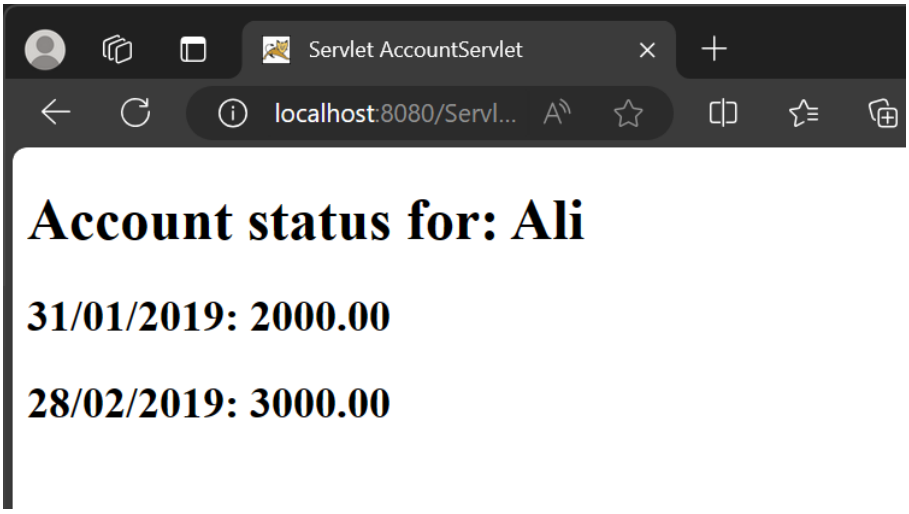
requests.

- void doPost(): This method is invoked by the servlet container to handle HTTP POST requests.

- void destroy(): This method is called by the servlet container to indicate that the servlet is being taken out of service.

OUTPUT

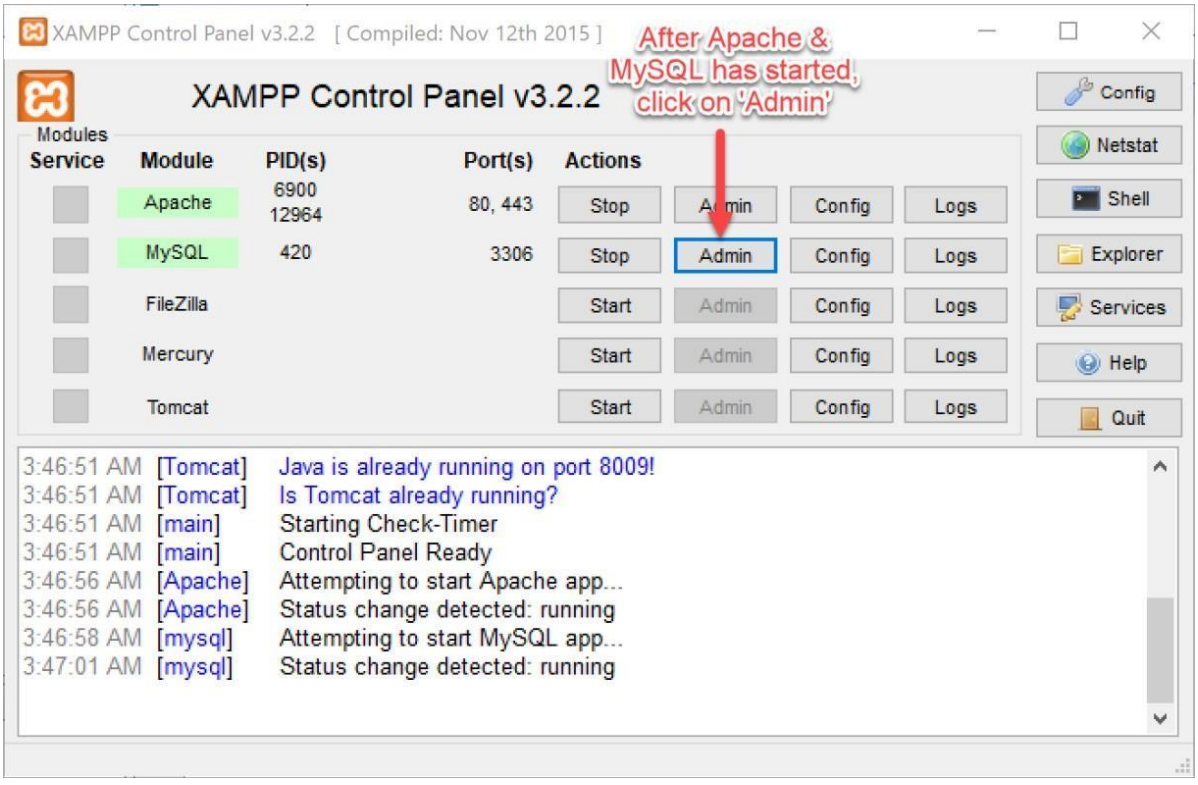




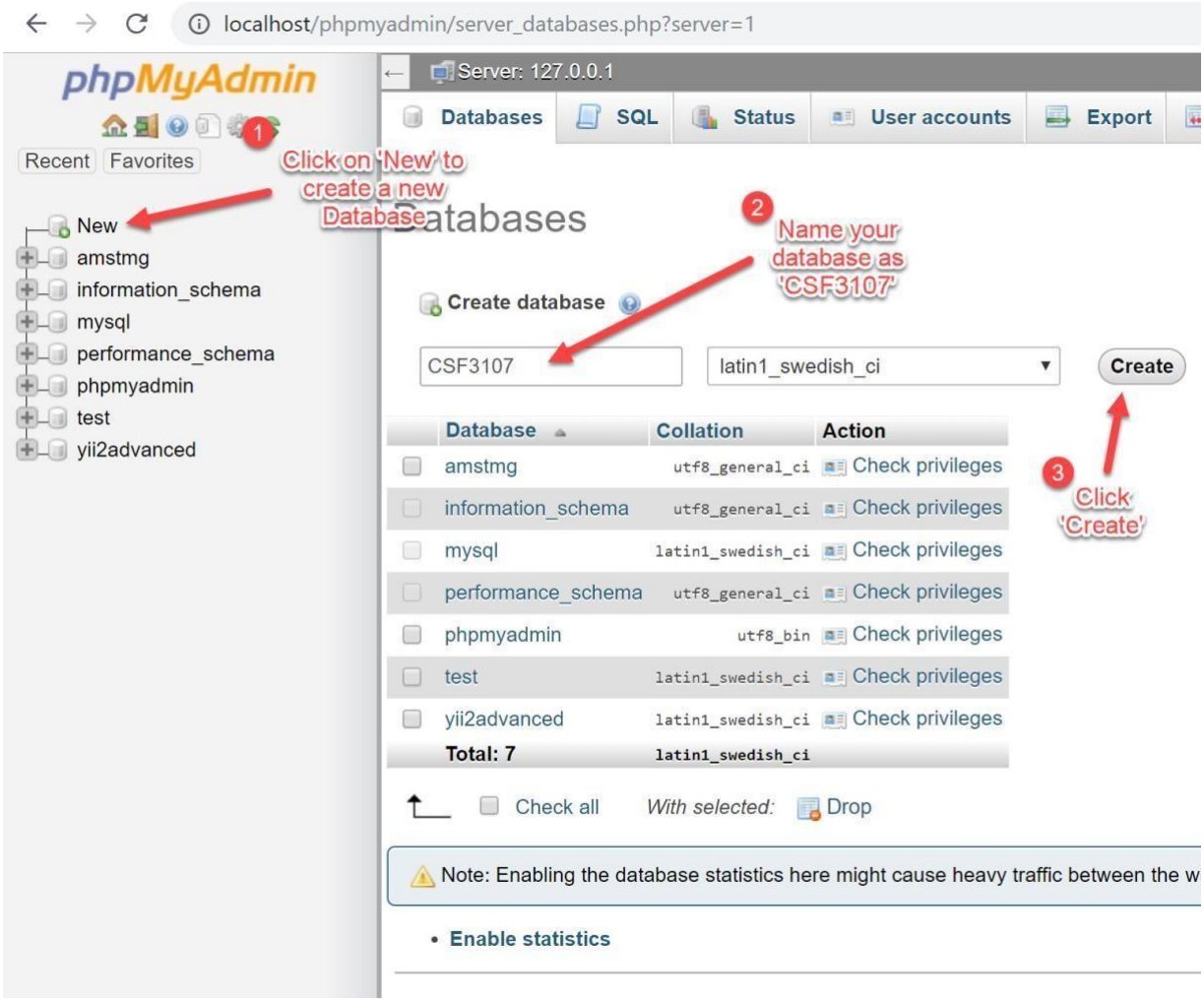
Task 2: Creating A Table in MySQL Database

Objective:	To create a MySQL table to store user credentials
Problem Description:	Prepare a user table to be used in Web Application
Estimated time:	30 minutes

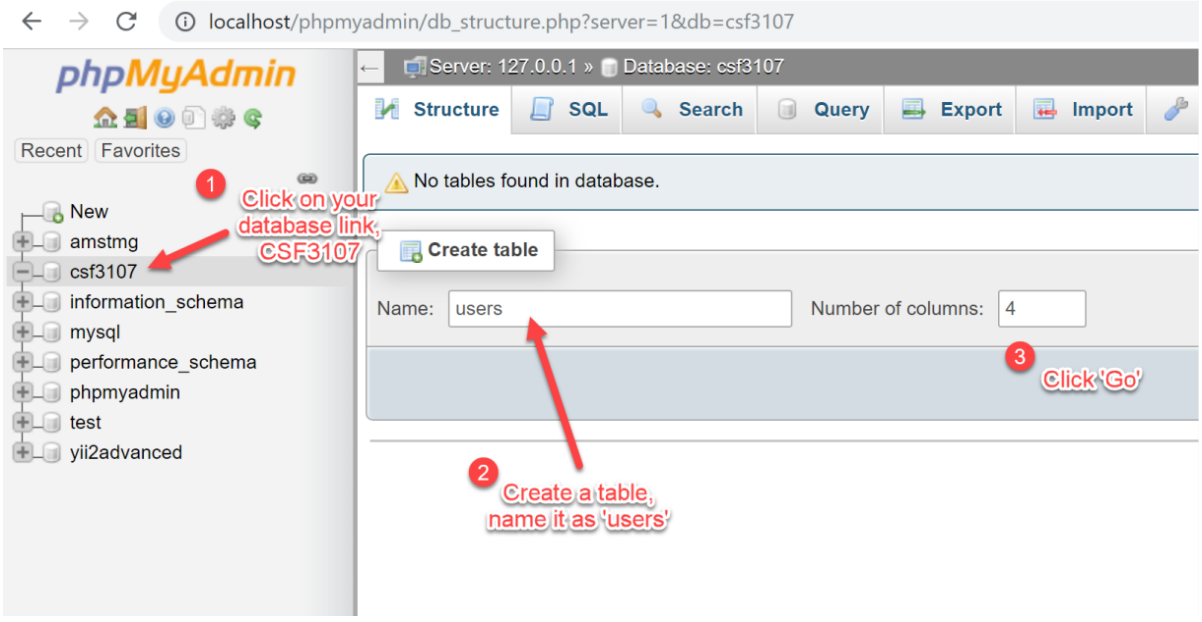
1. Start Apache and MySQL.



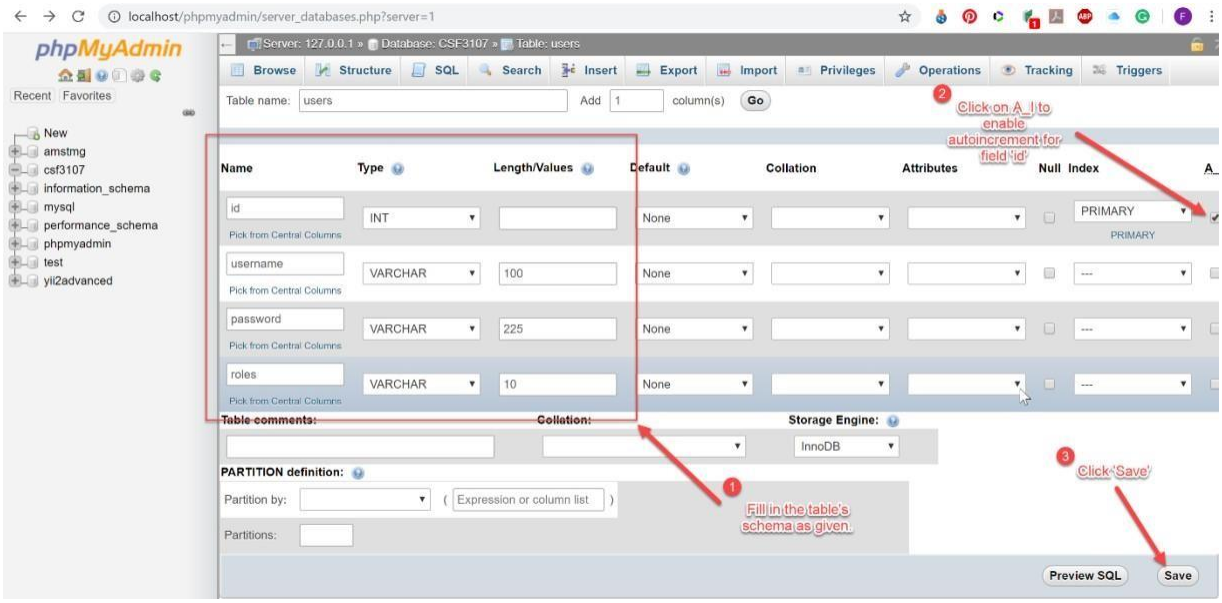
2. Create a new database.



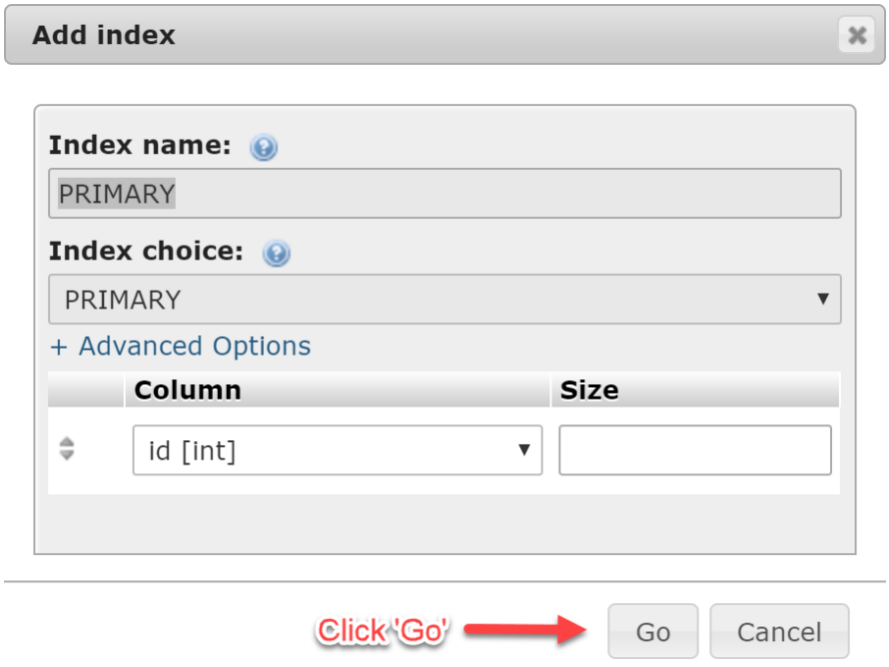
3. After the database has been created, now we are going the create a table named ‘users’. This table has four columns.



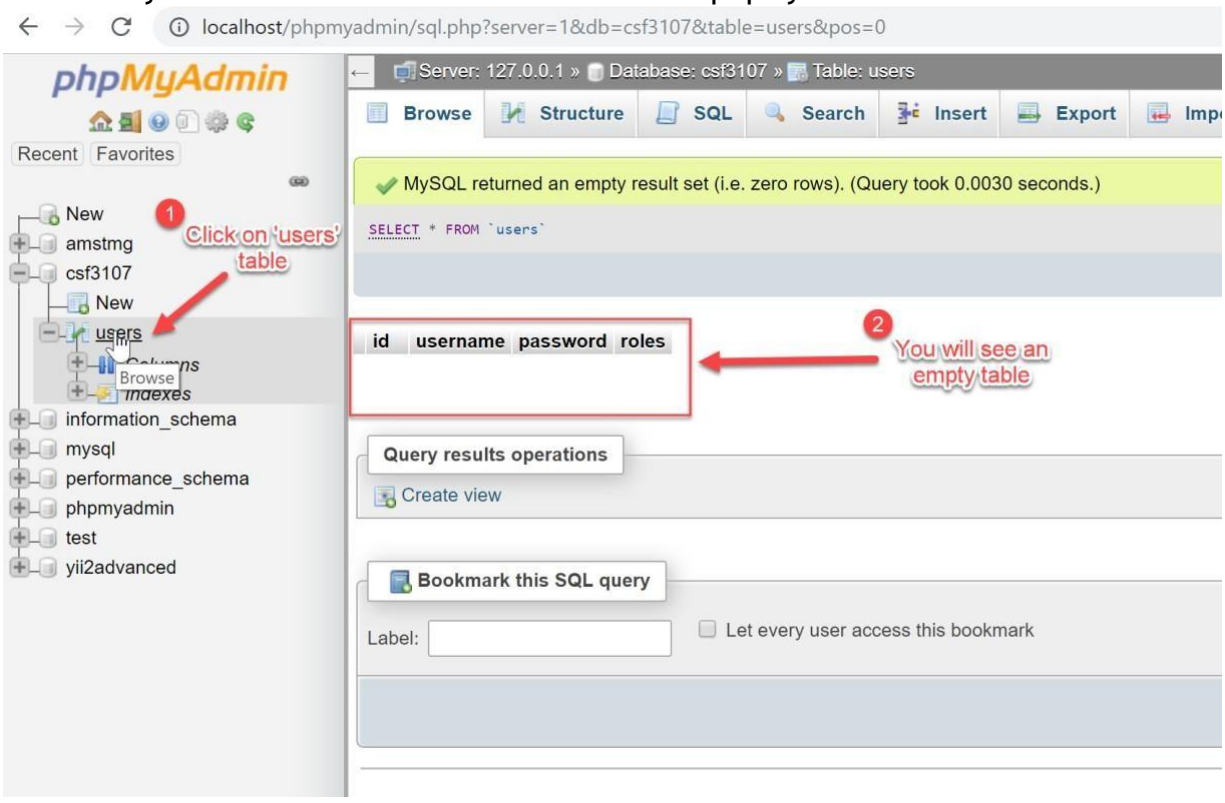
4. Setup the table ‘users’ as below:



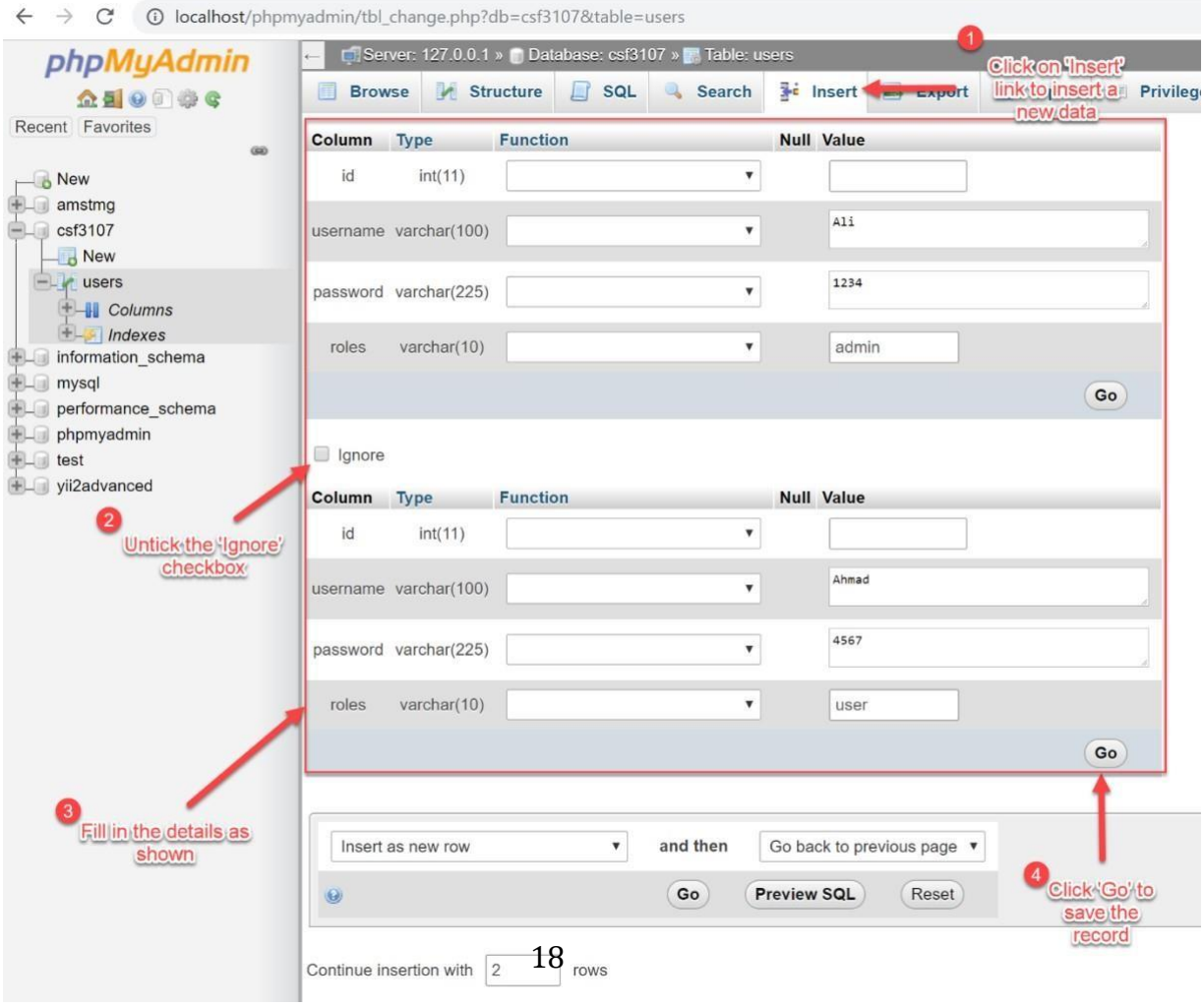
5. After you tick for auto-increment, this dialogue box will appear, leave the settings as it is and click ‘Go’.



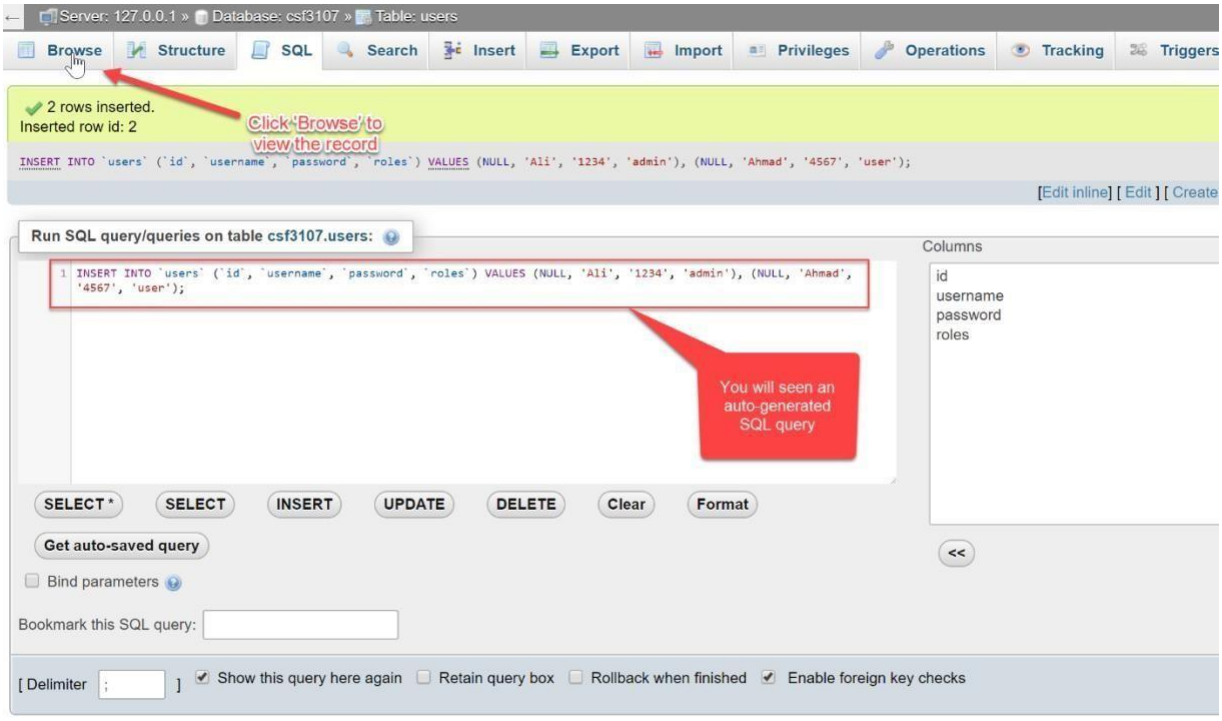
6. Your newly created table can be seen on the left phpMyAdmin.



7. You may insert some data into the table by following the steps on the screenshot.

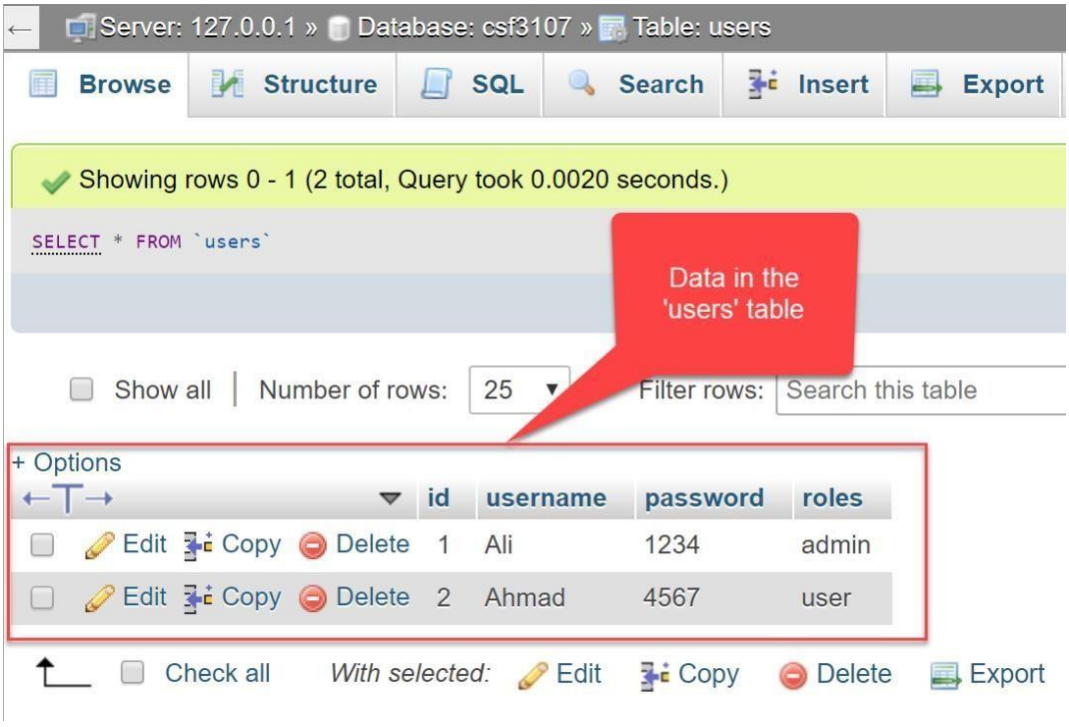


8. After you have clicked the ‘Go’ button in the previous step, you will see an autogenerated SQL query on the phpMyAdmin page. To view the data, click Browse.

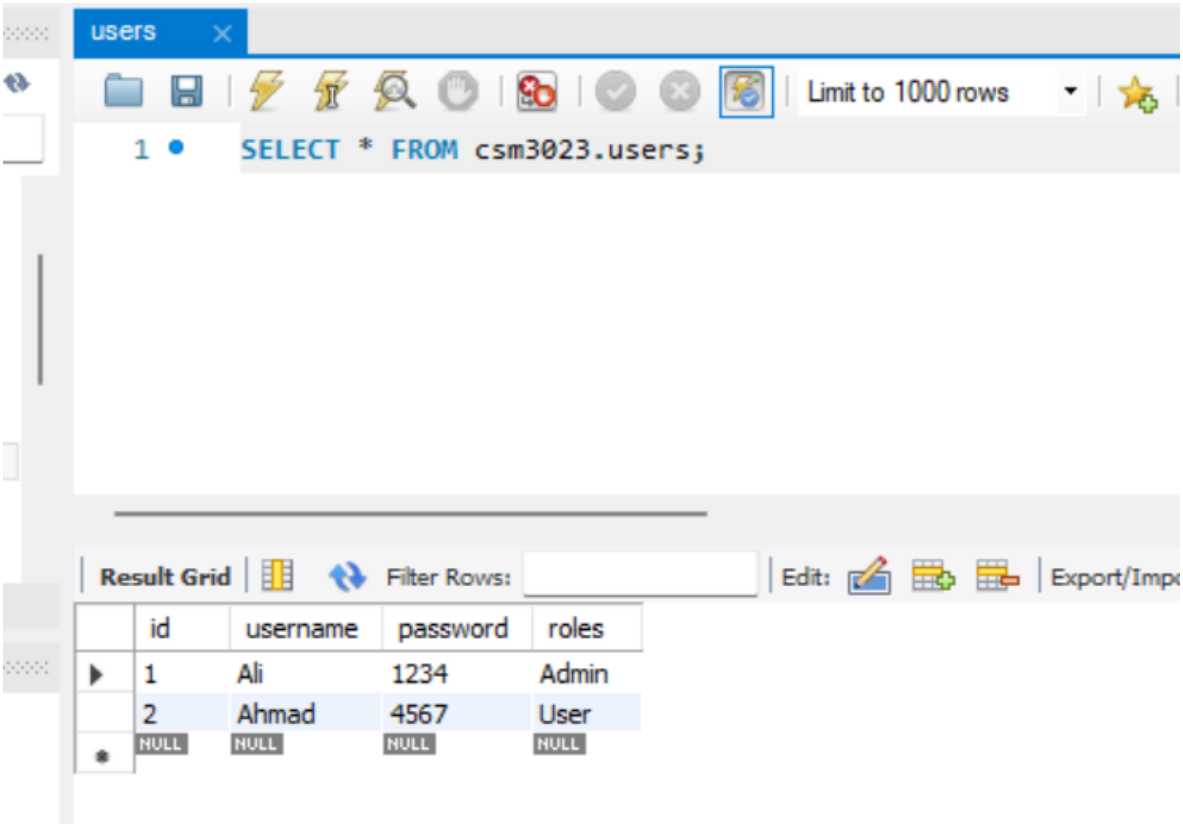


9. You will see the data of the users. Now, we have finished this task.

Note: In a real implementation, you should encrypt the password and never display the original form of it.



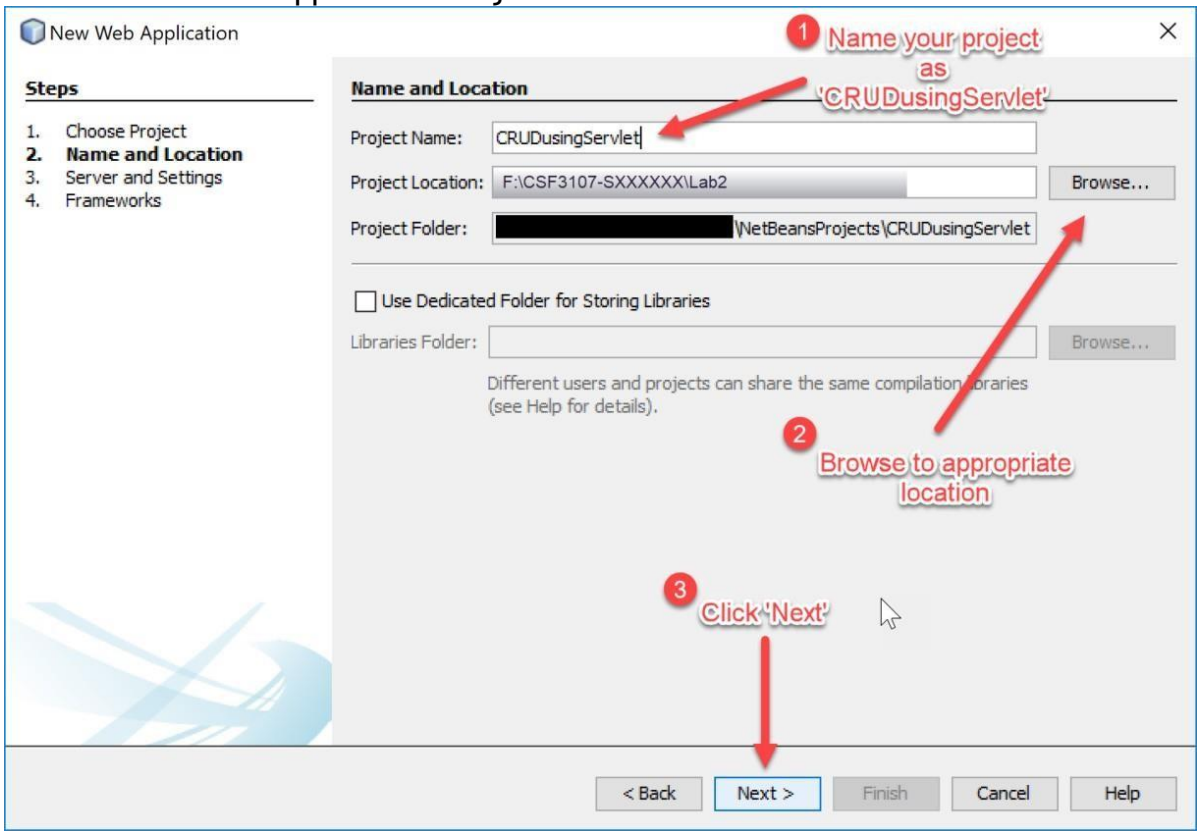
OUTPUT


users
Limit to 1000 rows
1
SELECT * FROM csm3023.users;
Result Grid
Filter Rows:
Edit:
Export/Impo
id
username
password
roles
1
Ali
1234
Admin
2
Ahmad
4567
User
NULL
NULL
NULL
NULL

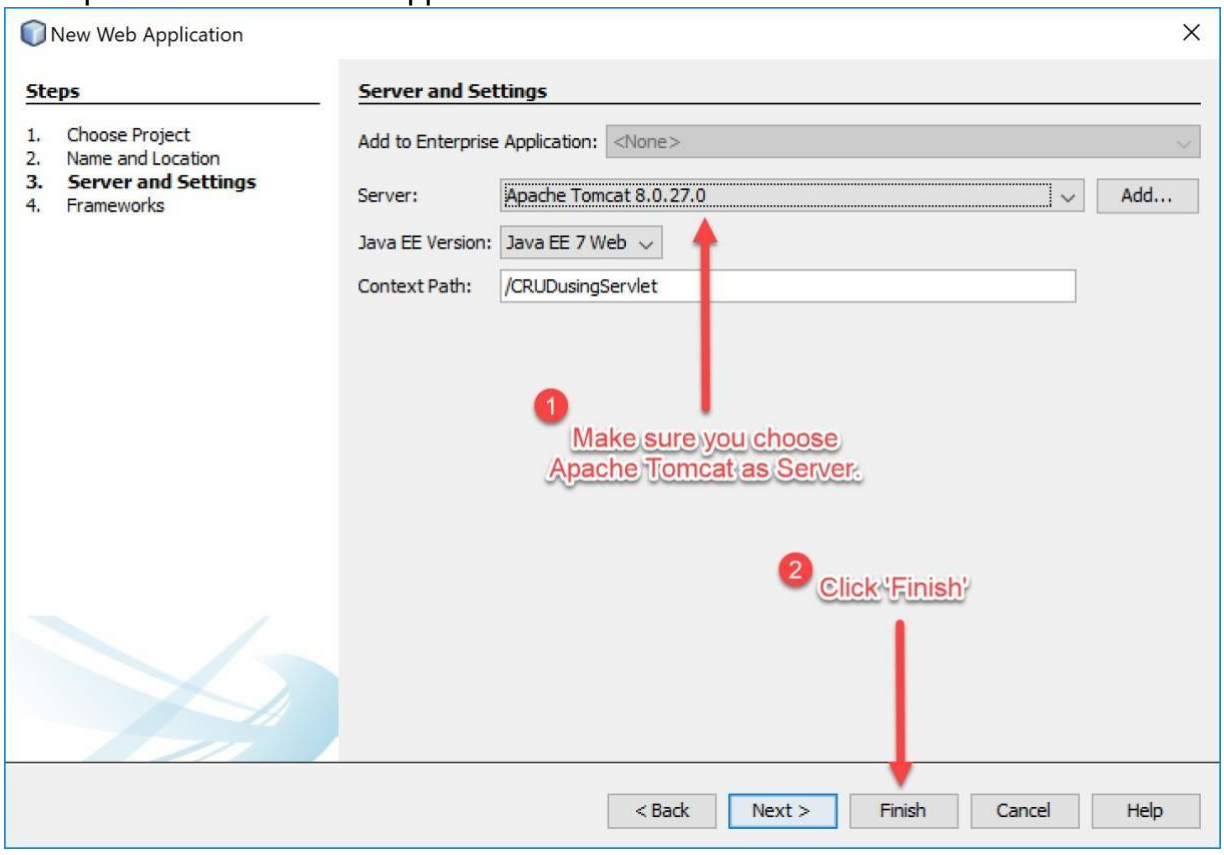
Task 3: Setting the Environment of Web Application for Database Connection

Objective:	To set up a proper environment for integrating web application to the database
Problem Description:	Import MySQL JDBC Library to an existing project
Estimated time:	5 minutes

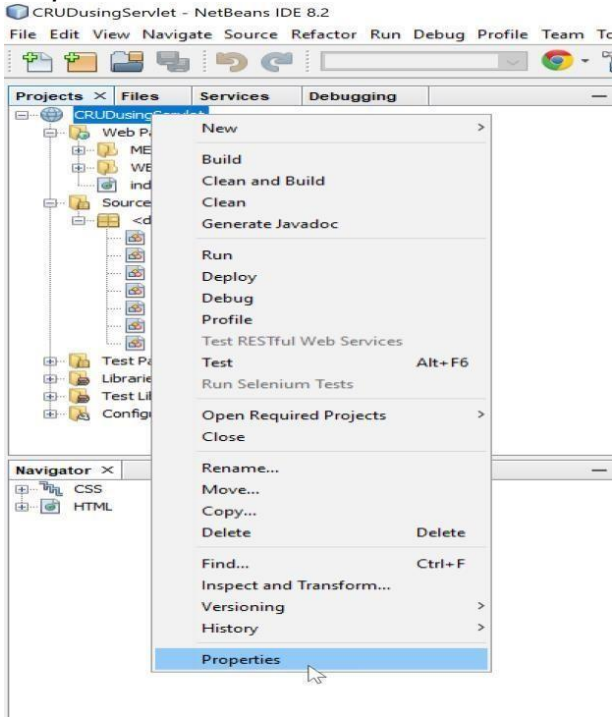
1. Create a new Web Application Project.



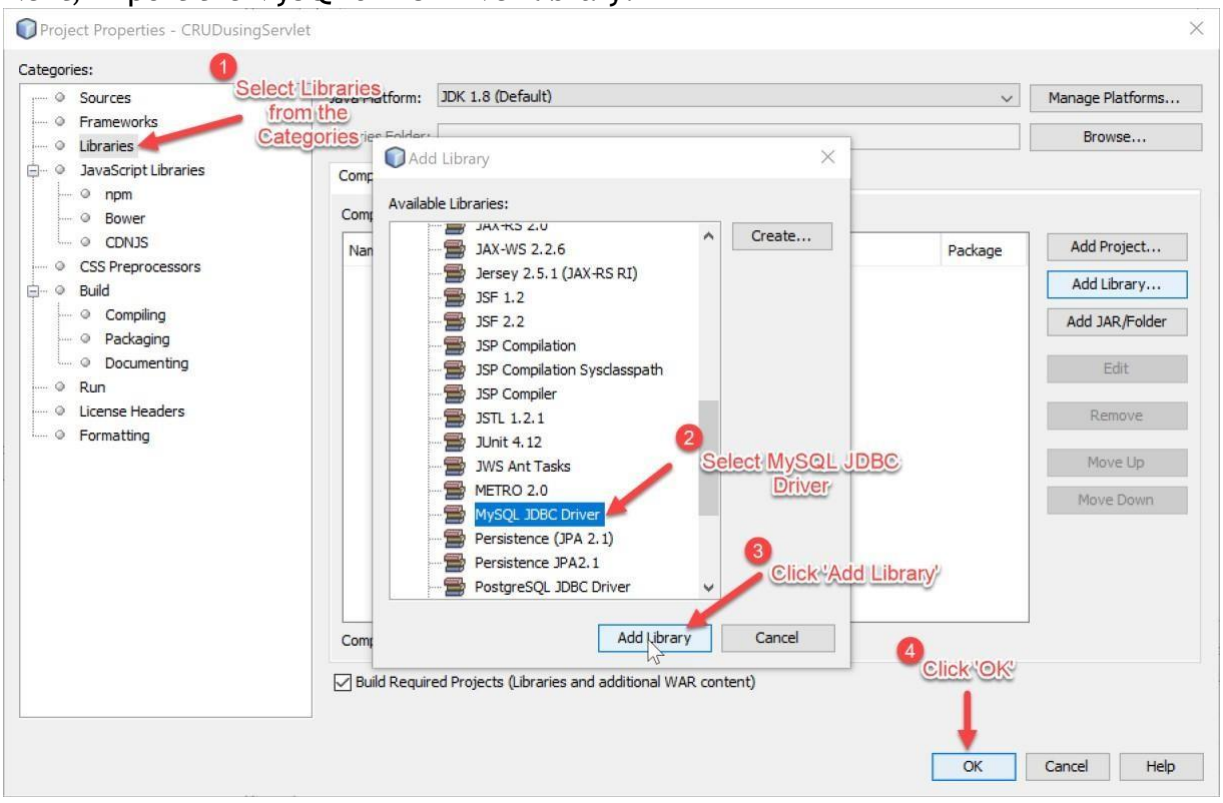
2. Use Apache Tomcat as our application server.



3. Once finished, right-click on Project's name *(CRUDusingServlet)* and choose *Properties*.



4. Next, import the MySQL JDBC Driver library.

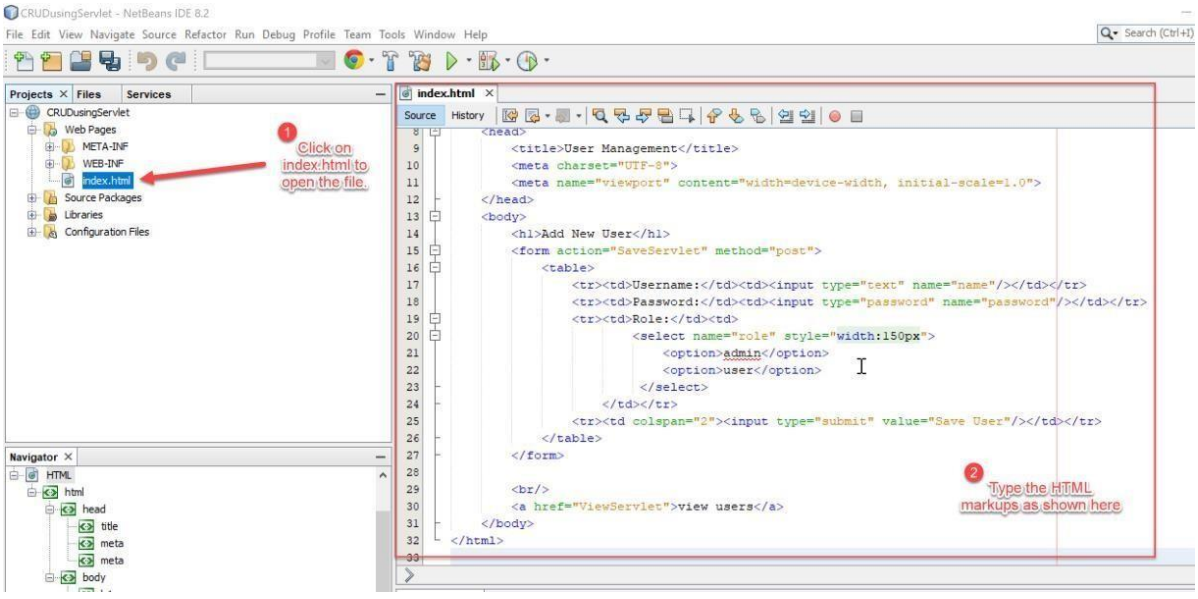


5. Now your application is ready to connect to MySQL database and execute a SQL query.

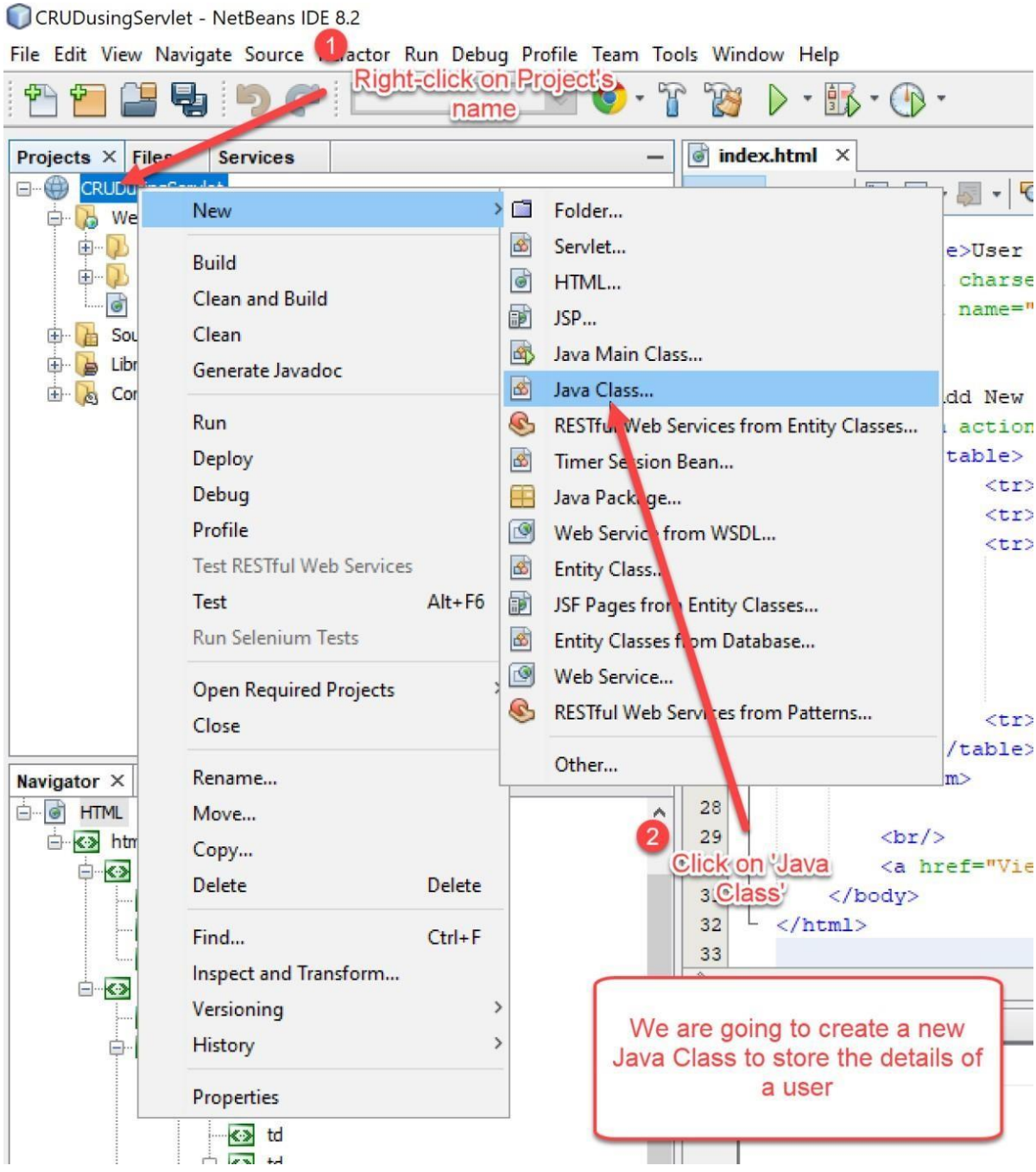
Task 4: Using Servlets for Database CRUD Operations

Objective:	To program multiple servlets for manipulating the database
Problem Description:	Program five different servlets to handle database operations such as insert, update and delete. i. SaveServlet.java: to save data into the database ii. ViewServlet.java: to view data retrieved from database iii. EditServlet.java & EditServlet2.java: to edit existing data iv. DeleteServlet.java: to delete existing data Apart from the servlets, we are going to develop two custom Java class know as JavaBeans and Data Access Object (DAO).
Estimated time:	90 minutes

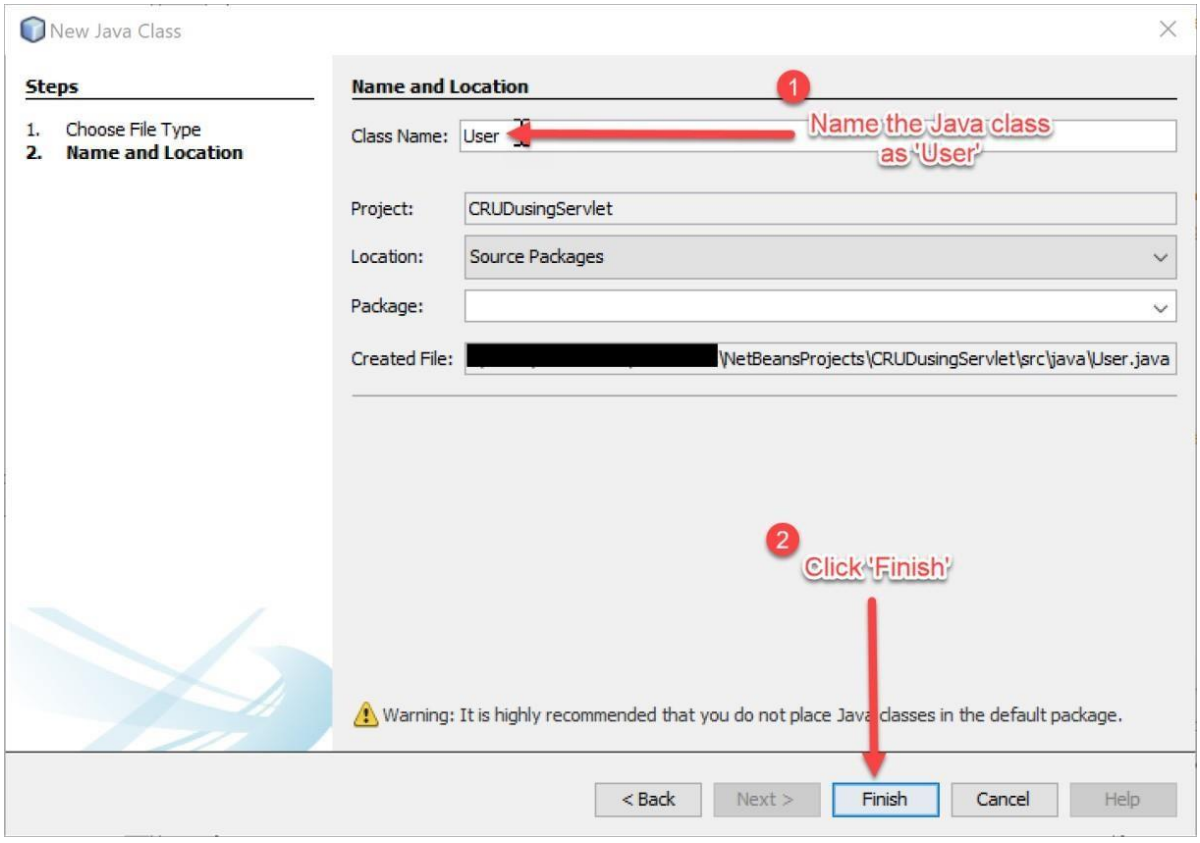
1. Open the Web Application Project *(CRUDusingServlet)* created in Task 3.
2. Open the index.html file and edit as follows.



3. Now we are going to create a Java class which is specially used to store the user data.



4. Name the Java class as *User*.



5. Type the following codes in User.java file.

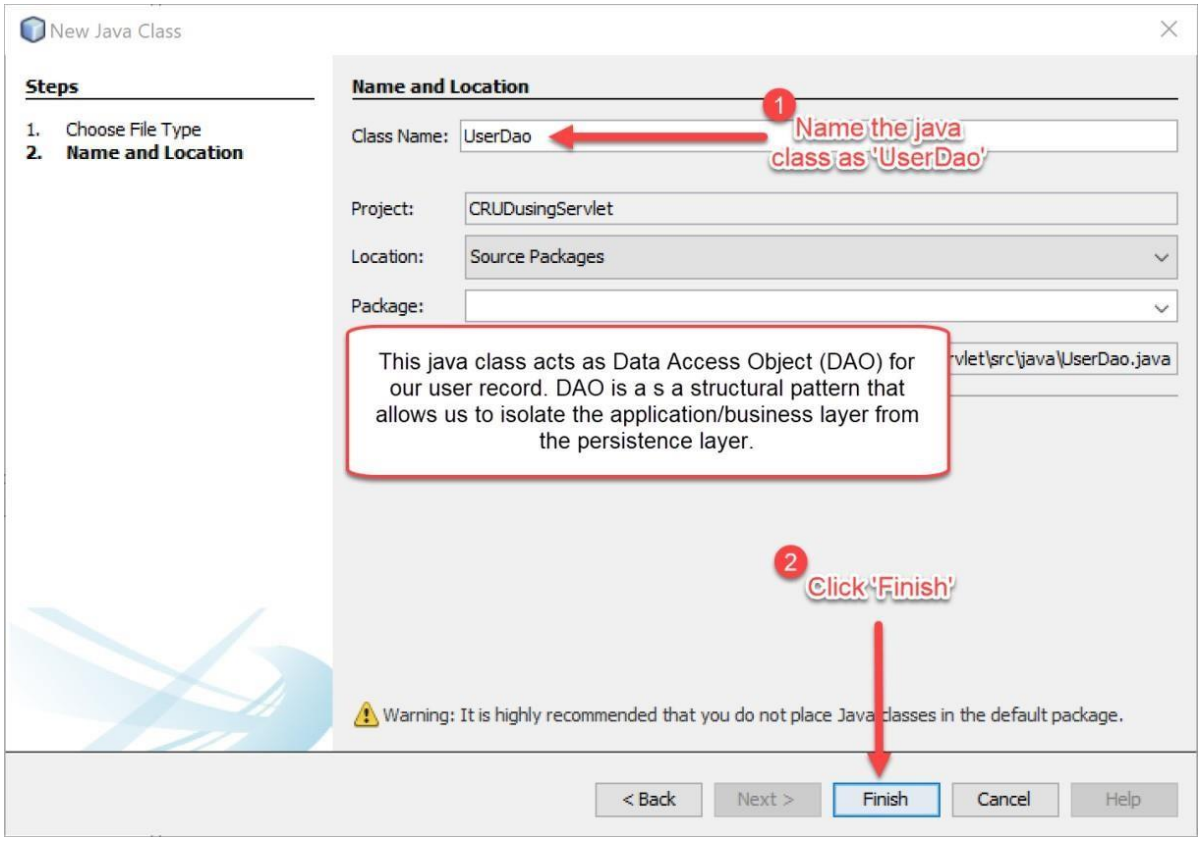
User.java - Editor

User.java

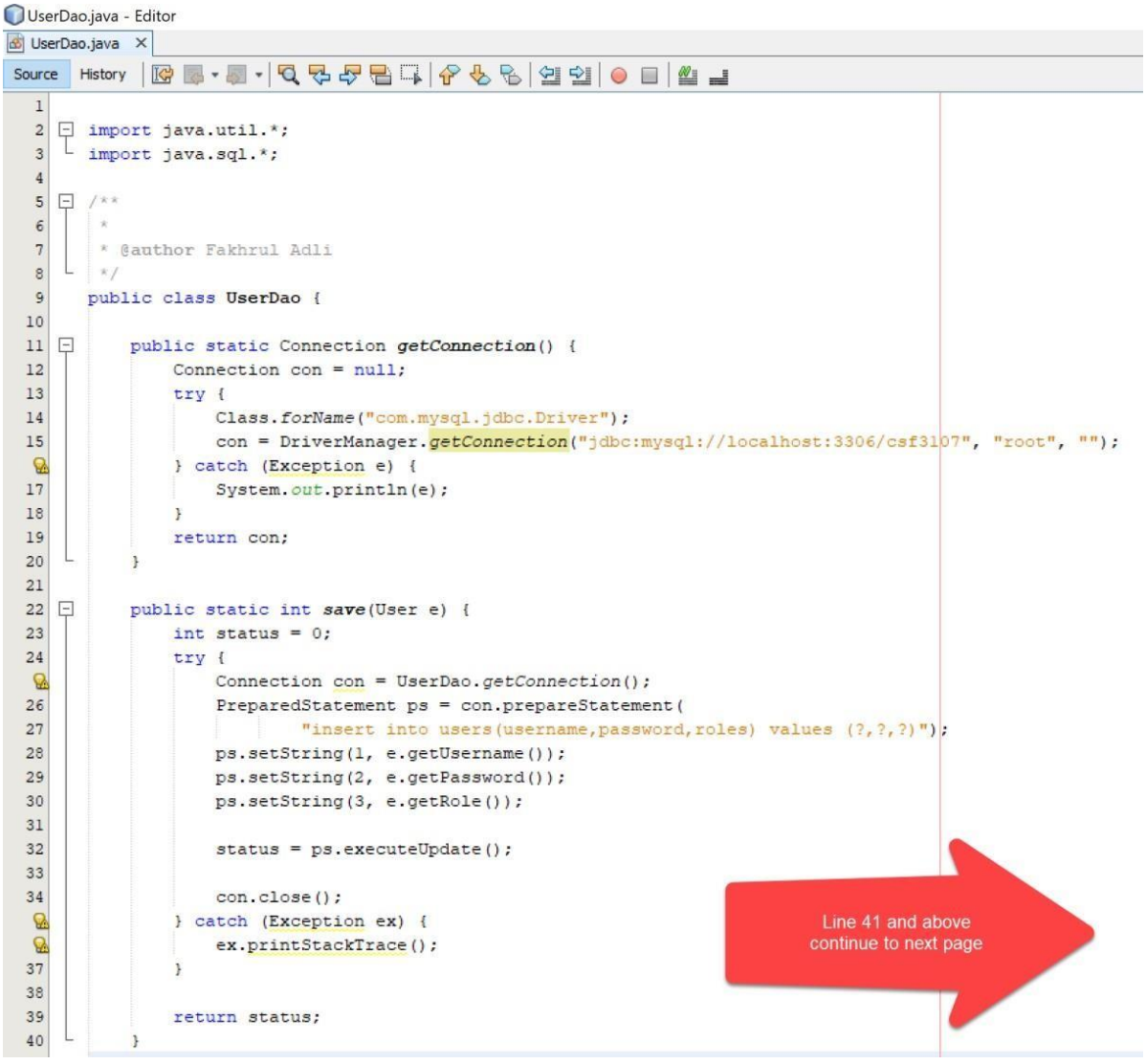
Source History

```
10  */
11  public class User {
12
13      private int id;
14      private String username, password, role;
15
16      public int getId() {
17          return id;
18      }
19
20      public void setId(int id) {
21          this.id = id;
22      }
23
24      public String getUsername() {
25          return username;
26      }
27
28      public void setUsername(String username) {
29          this.username = username;
30      }
31
32      public String getPassword() {
33          return password;
34      }
35
36      public void setPassword(String password) {
37          this.password = password;
38      }
39
40      public String getRole() {
41          return role;
42      }
43
44      public void setRole(String role) {
45          this.role = role;
46      }
47
48  }
```

6. Next, we need to program the Data Access Object (DAO) class which is used for managing database connection and SQL operations.

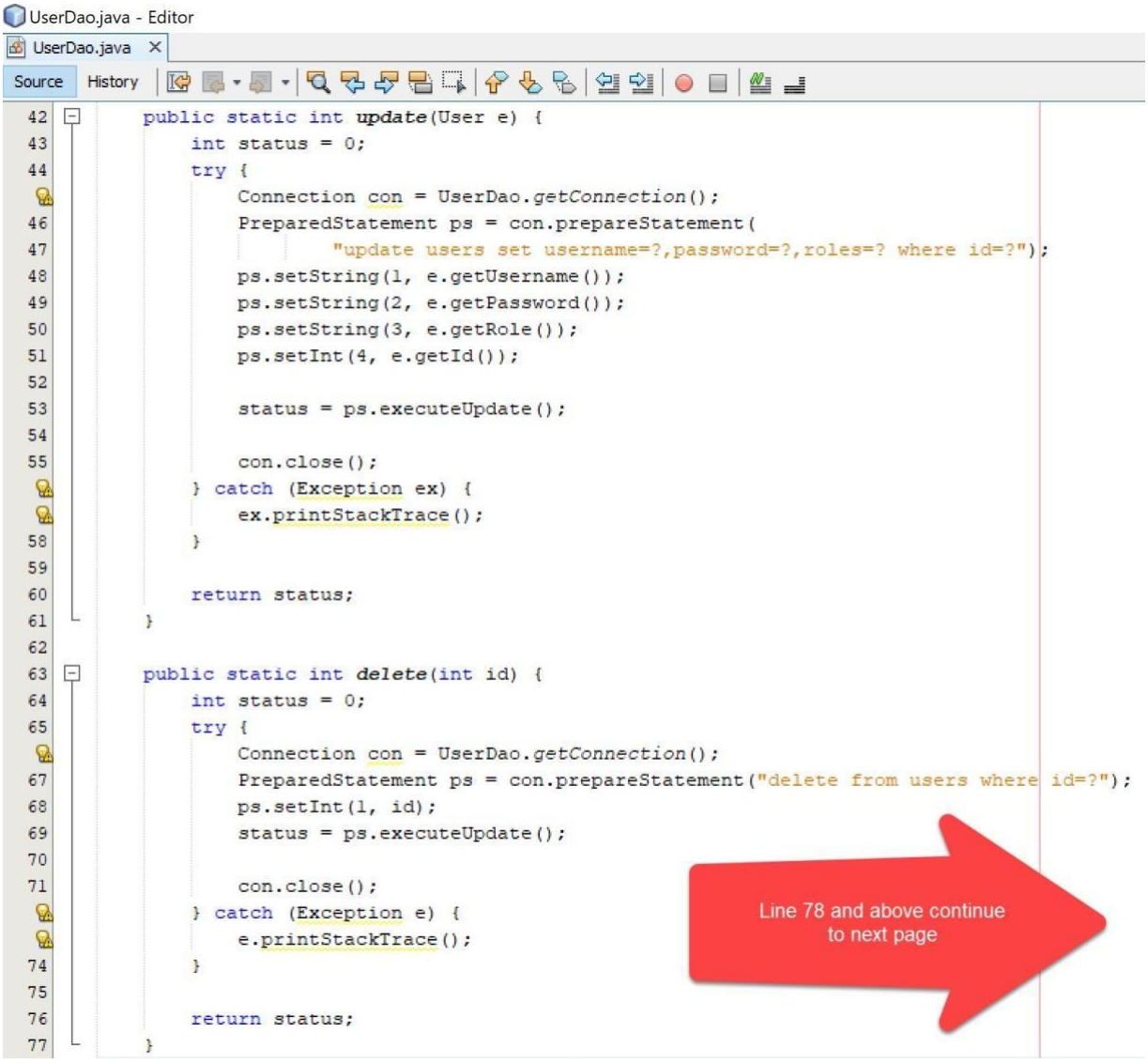


7. Put the following codes in UserDao.java file. Make sure your database URL is correct, and database user credentials are also valid.

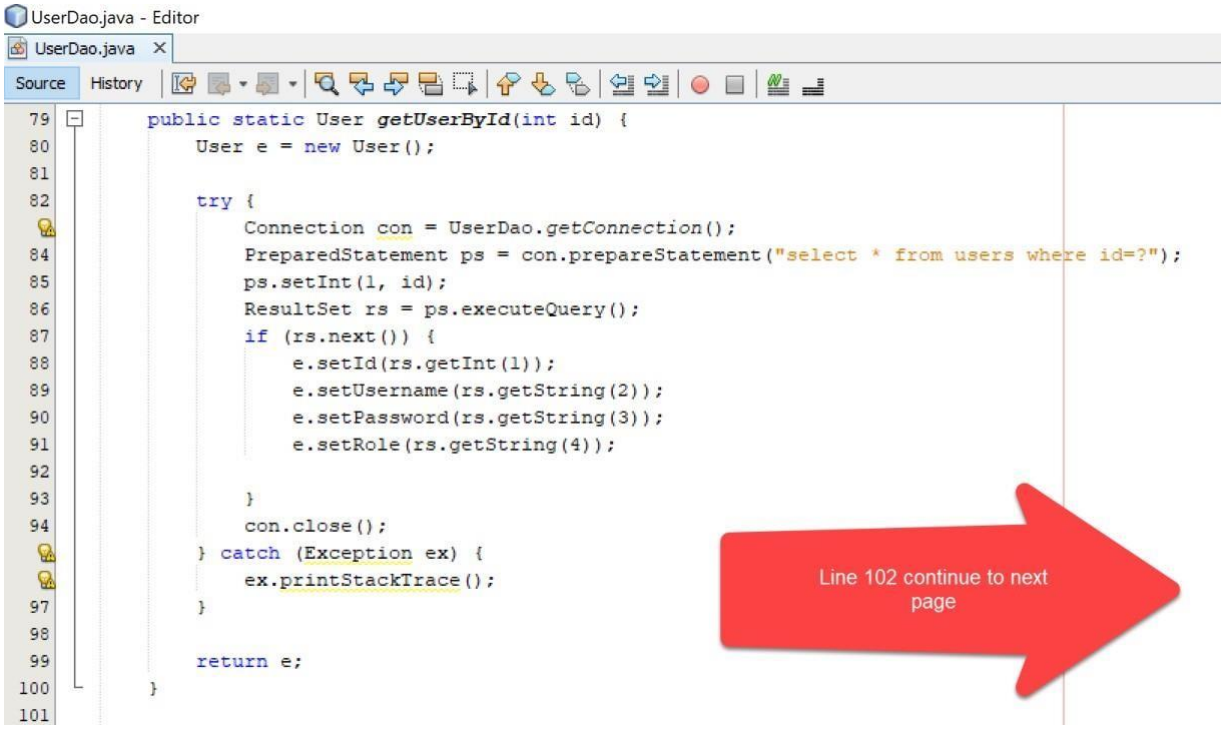



```
1
2   import java.util.*;
3   import java.sql.*;
4
5   /**
6    *
7    * @author Fakhrul Adli
8    */
9   public class UserDao {
10
11      public static Connection getConnection() {
12          Connection con = null;
13          try {
14              Class.forName("com.mysql.jdbc.Driver");
15              con = DriverManager.getConnection("jdbc:mysql://localhost:3306/csf3107", "root", "");
17          } catch (Exception e) {
18              System.out.println(e);
19          }
20          return con;
21      }
22
23      public static int save(User e) {
24          int status = 0;
26          try {
27              Connection con = UserDao.getConnection();
28              PreparedStatement ps = con.prepareStatement(
29                      "insert into users(username,password,roles) values (?,?,?)");
30              ps.setString(1, e.getUsername());
31              ps.setString(2, e.getPassword());
32              ps.setString(3, e.getRole());
33
34              status = ps.executeUpdate();
37              con.close();
38          } catch (Exception ex) {
39              ex.printStackTrace();
40          }
41
42          return status;
43      }
```

8. …continued.


UserDao.java - Editor
UserDao.java
Source
History
42 public static int update(User e) {
43 int status = 0;
44 try {
46 Connection con = UserDao.getConnection();
47 PreparedStatement ps = con.prepareStatement(
48 "update users set username=?,password=?,roles=? where id=?");
49 ps.setString(1, e.getUsername());
50 ps.setString(2, e.getPassword());
51 ps.setString(3, e.getRole());
52 ps.setInt(4, e.getId());
53
54 status = ps.executeUpdate();
55
58 con.close();
59 } catch (Exception ex) {
60 ex.printStackTrace();
61 }
62
63 return status;
64 }
65
67 public static int delete(int id) {
68 int status = 0;
69 try {
70 Connection con = UserDao.getConnection();
71 PreparedStatement ps = con.prepareStatement("delete from users where id=?");
74 ps.setInt(1, id);
75 status = ps.executeUpdate();
76
77 con.close();
} catch (Exception e) {
e.printStackTrace();
}
return status;
}
Line 78 and above continue
to next page

9. …continued.

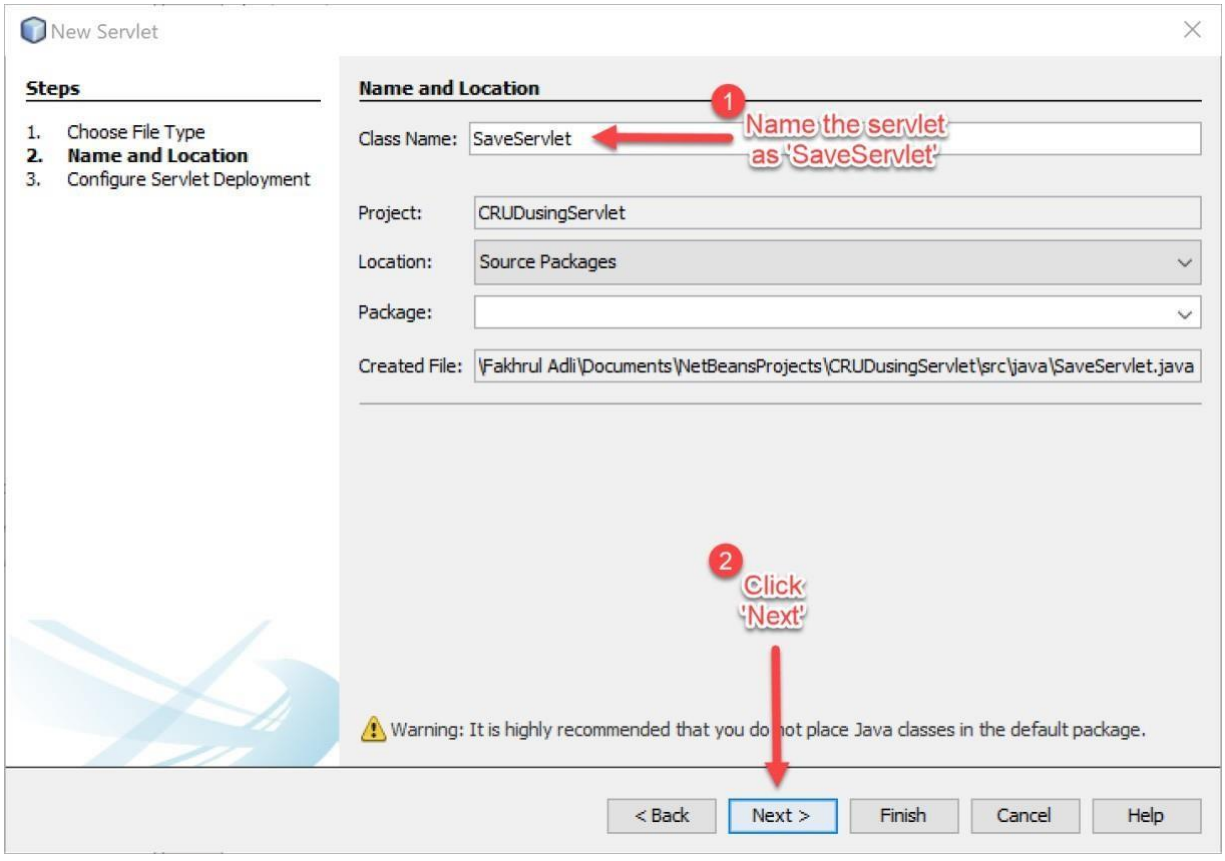



```
79      public static User getUserById(int id) {
80          User e = new User();
81
82          try {
84              Connection con = UserDao.getConnection();
85              PreparedStatement ps = con.prepareStatement("select * from users where id=?");
86              ps.setInt(1, id);
87              ResultSet rs = ps.executeQuery();
88              if (rs.next()) {
89                  e.setId(rs.getInt(1));
90                  e.setUsername(rs.getString(2));
91                  e.setPassword(rs.getString(3));
92                  e.setRole(rs.getString(4));
93
94              }
97              con.close();
98          } catch (Exception ex) {
99              ex.printStackTrace();
100         }
101
102         return e;
123     }
```

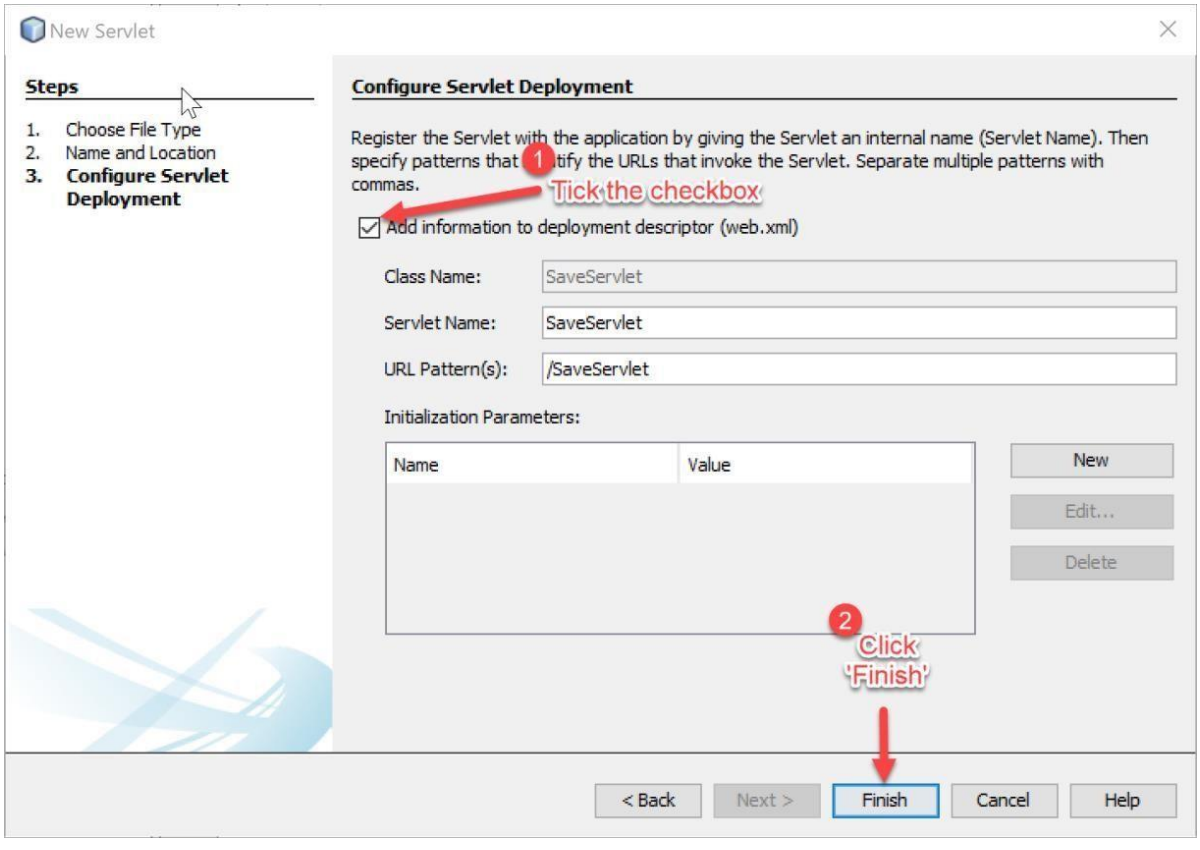
10. Save (CTRL+s) UserDao.java file.


```
101
102     public static List<User> getAllUsers() {
104         List<User> list = new ArrayList<User>();
105
107         try {
108             Connection con = UserDao.getConnection();
109             PreparedStatement ps = con.prepareStatement("select * from users");
110             ResultSet rs = ps.executeQuery();
111             while (rs.next()) {
112                 User e = new User();
113                 e.setId(rs.getInt(1));
114                 e.setUsername(rs.getString(2));
115                 e.setPassword(rs.getString(3));
116                 e.setRole(rs.getString(4));
117                 list.add(e);
120             }
121             con.close();
122         } catch (Exception e) {
123             e.printStackTrace();
124         }
125
            return list;
        }
    }
```

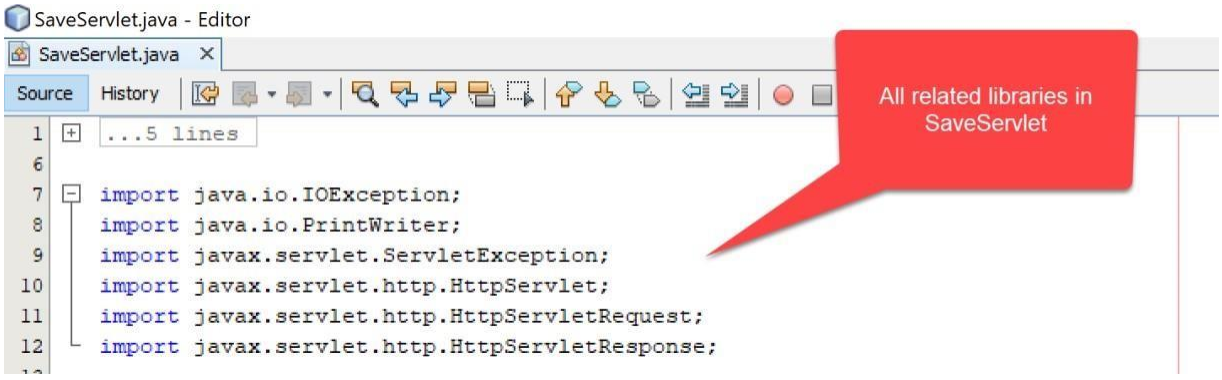
11. After finish coding the UserDao.java, now we are to code the servlet for saving data into the database.



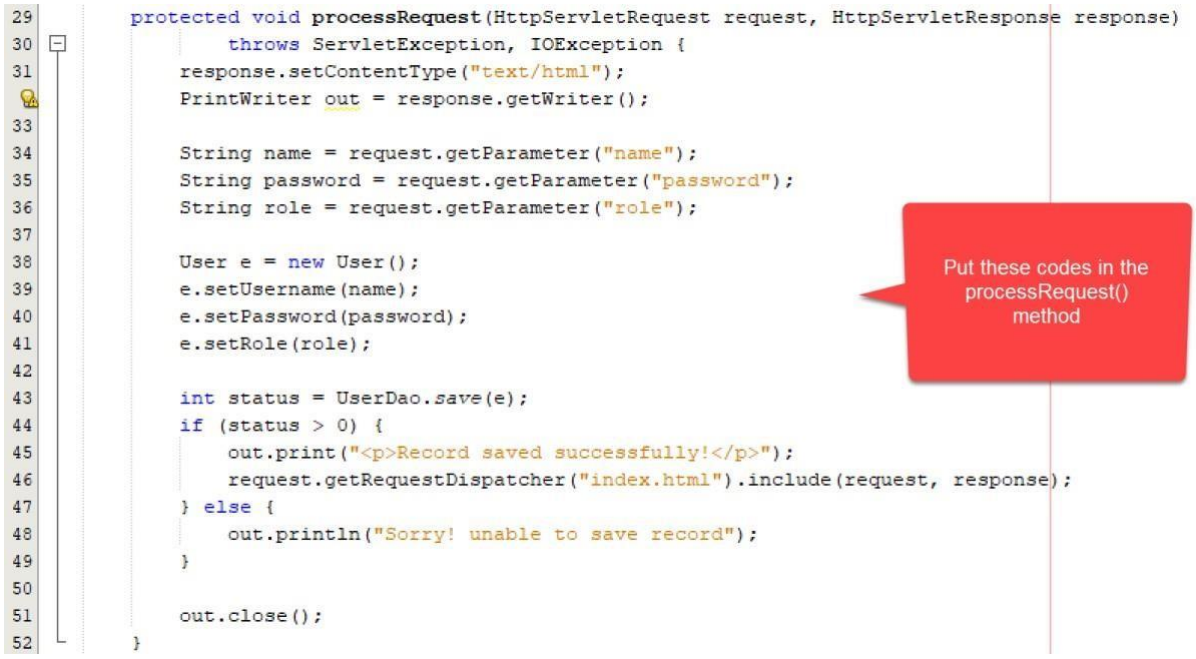
12. Remember to tick the checkbox every time you create a new servlet.



13. Make sure all related libraries are included in the SaveServlet.java file.

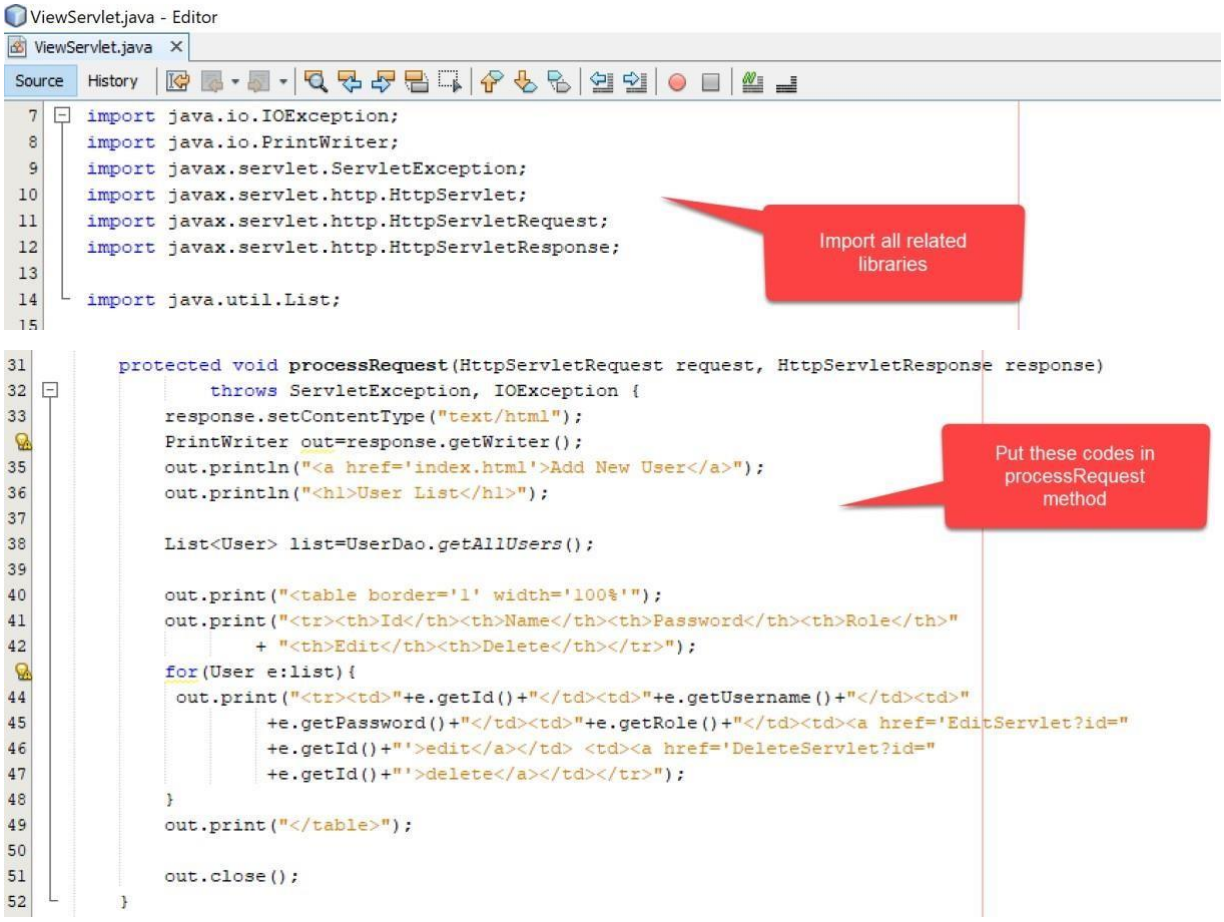


14. Type the following codes in processRequest() method.



15. Save the file after finished.

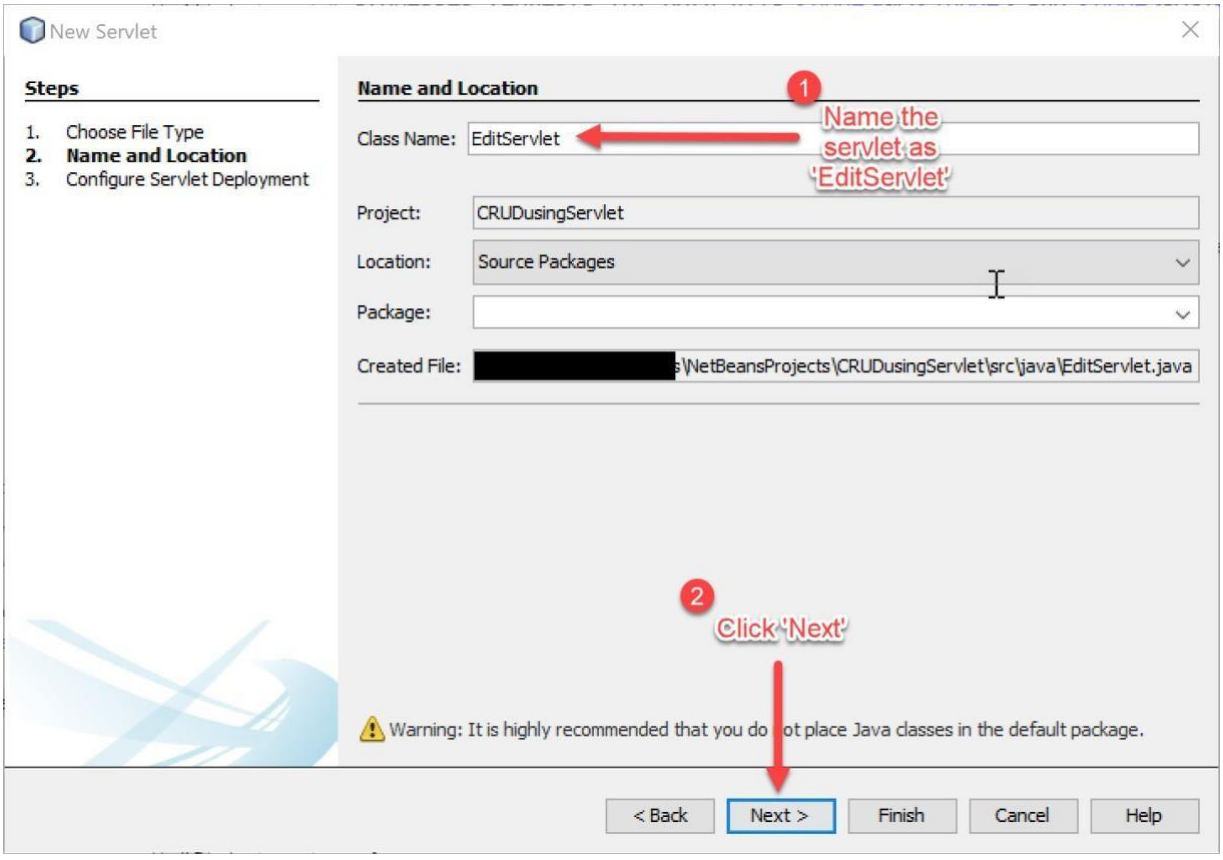
16. Next, we are going to create a servlet to view our data in database. Create a new servlet *ViewServlet*, tick 'Add information to deployment descriptor (web.xml)' and click finish. Insert the following codes to the servlet.

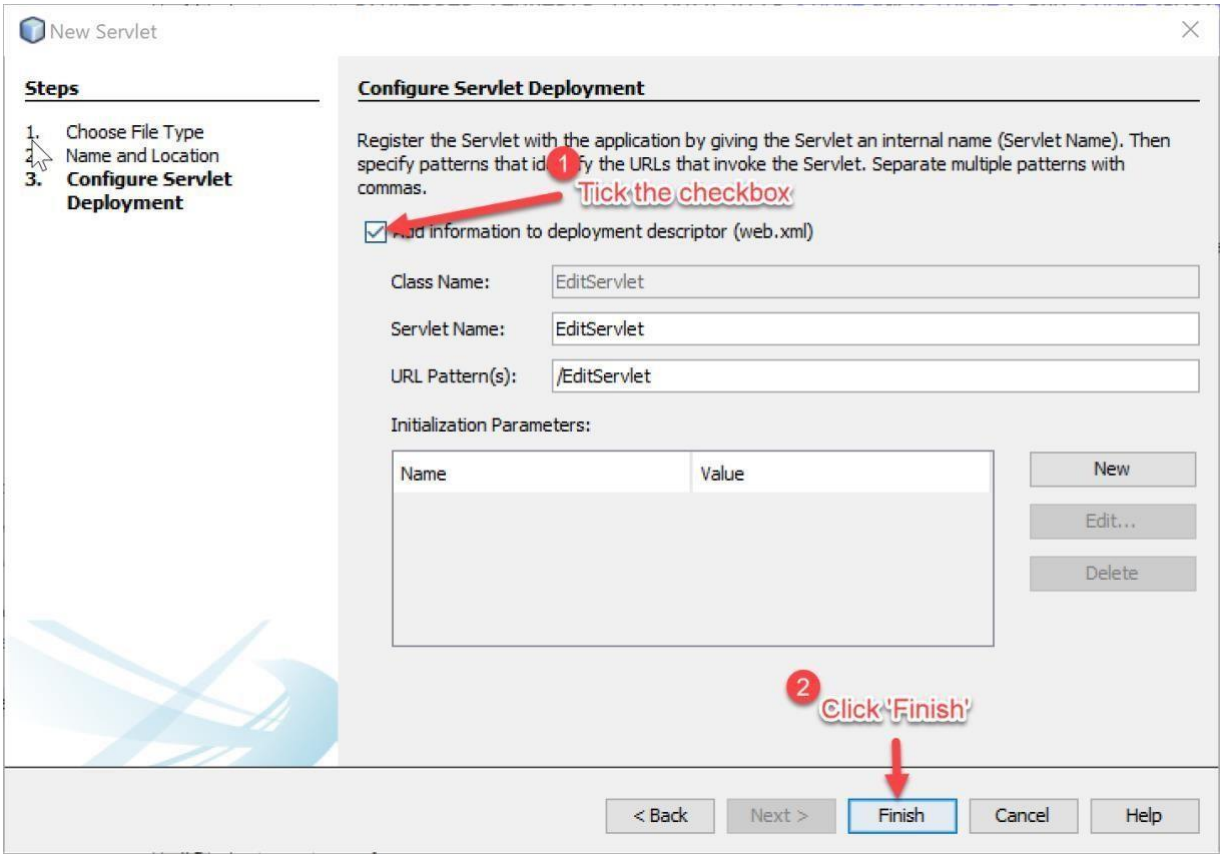



```
import java.io.IOException;
import java.io.PrintWriter;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

import java.util.List;
```

```
protected void processRequest(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {
    response.setContentType("text/html");
    PrintWriter out=response.getWriter();
    out.println("<a href='index.html'>Add New User</a>");
    out.println("<h1>User List</h1>");

    List<User> list=UserDao.getAllUsers();

    out.print("<table border='1' width='100%'");
    out.print("<tr><th>Id</th><th>Name</th><th>Password</th><th>Role</th>"
            + "<th>Edit</th><th>Delete</th></tr>");
    for(User e:list){
     out.print("<tr><td>"+e.getId()+"</td><td>"+e.getUsername()+"</td><td>"
             +e.getPassword()+"</td><td>"+e.getRole()+"</td><td><a href='EditServlet?id="
             +e.getId()+"'>edit</a></td> <td><a href='DeleteServlet?id="
             +e.getId()+"'>delete</a></td></tr>");
    }
    out.print("</table>");

    out.close();
}
```

17. Currently, we have finished creating a servlet for saving and viewing the data in the database. Now, we need a servlet to handle the process of editing the existing data stored in the table in the database.





18. Type the following codes in the servlet you just created.

EditServlet.java - Editor

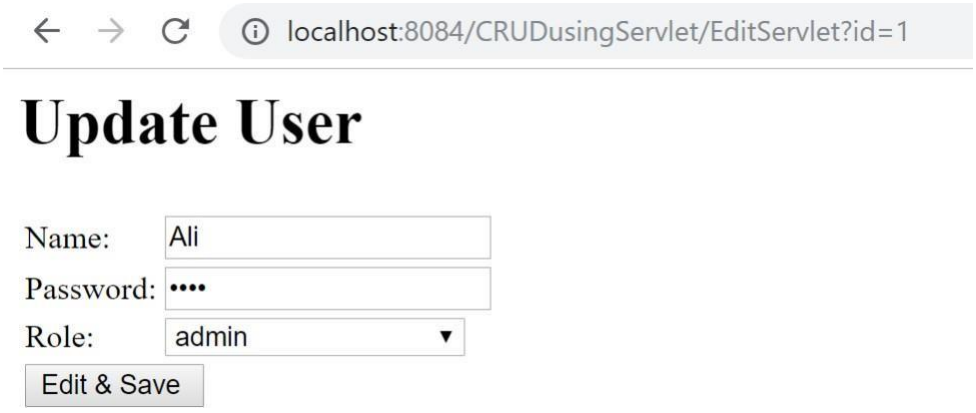
EditServlet.java

Source | History

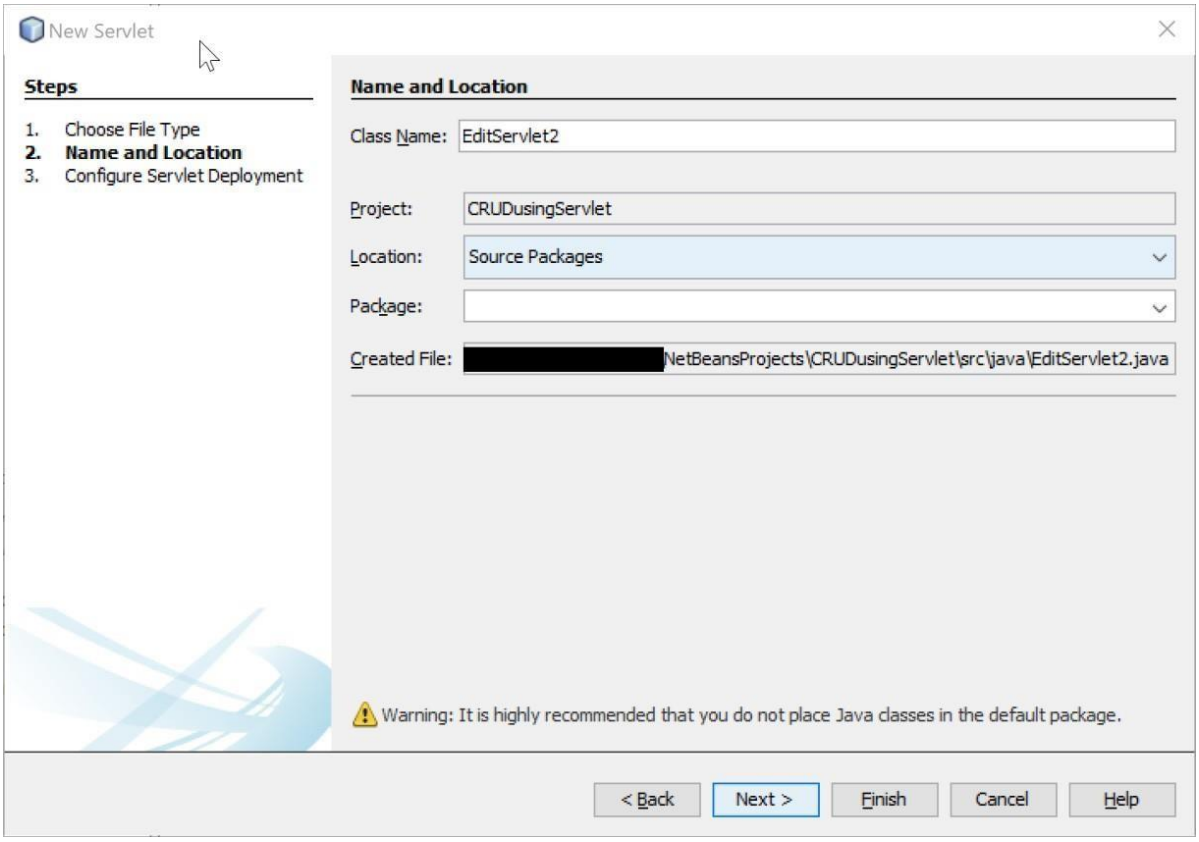
Make EditServlet.java looks like this

```
7  import java.io.IOException;
8  import java.io.PrintWriter;
9  import javax.servlet.ServletException;
10 import javax.servlet.http.HttpServlet;
11 import javax.servlet.http.HttpServletRequest;
12 import javax.servlet.http.HttpServletResponse;
13
14 /**...4 lines */
18 public class EditServlet extends HttpServlet {
19
20     /** Processes requests for both HTTP <code>GET</code> and <code>POST</code> ...9 lines */
29     protected void processRequest(HttpServletRequest request, HttpServletResponse response)
30             throws ServletException, IOException {
31         response.setContentType("text/html");
32         PrintWriter out=response.getWriter();
33         out.println("<h1>Update User</h1>");
34         String sid=request.getParameter("id");
35         int id=Integer.parseInt(sid);
36
37         User e=UserDao.getUserById(id);
38
39         out.print("<form action='EditServlet2' method='post'>");
40         out.print("<table>");
41         out.print("<tr><td></td><td><input type='hidden' name='id' value='"
42                 +e.getId()+"'/></td></tr>");
43         out.print("<tr><td>Name:</td><td><input type='text' name='username' value='"
44                 +e.getUsername()+"'/></td></tr>");
45         out.print("<tr><td>Password:</td><td><input type='password' name='password' value='"
46                 +e.getPassword()+"'/></td></tr>");
47         out.print("<tr><td>Role:</td><td>");
48         out.print("<select name='role' style='width:150px'>");
49         out.print("<option>admin</option>");
50         out.print("<option>user</option>");
51         out.print("</select>");
52         out.print("</td></tr>");
53         out.print("<tr><td colspan='2'><input type='submit' value='Edit & Save '/></td></tr>");
54         out.print("</table>");
55         out.print("</form>");
56
57         out.close();
58     }
59
```

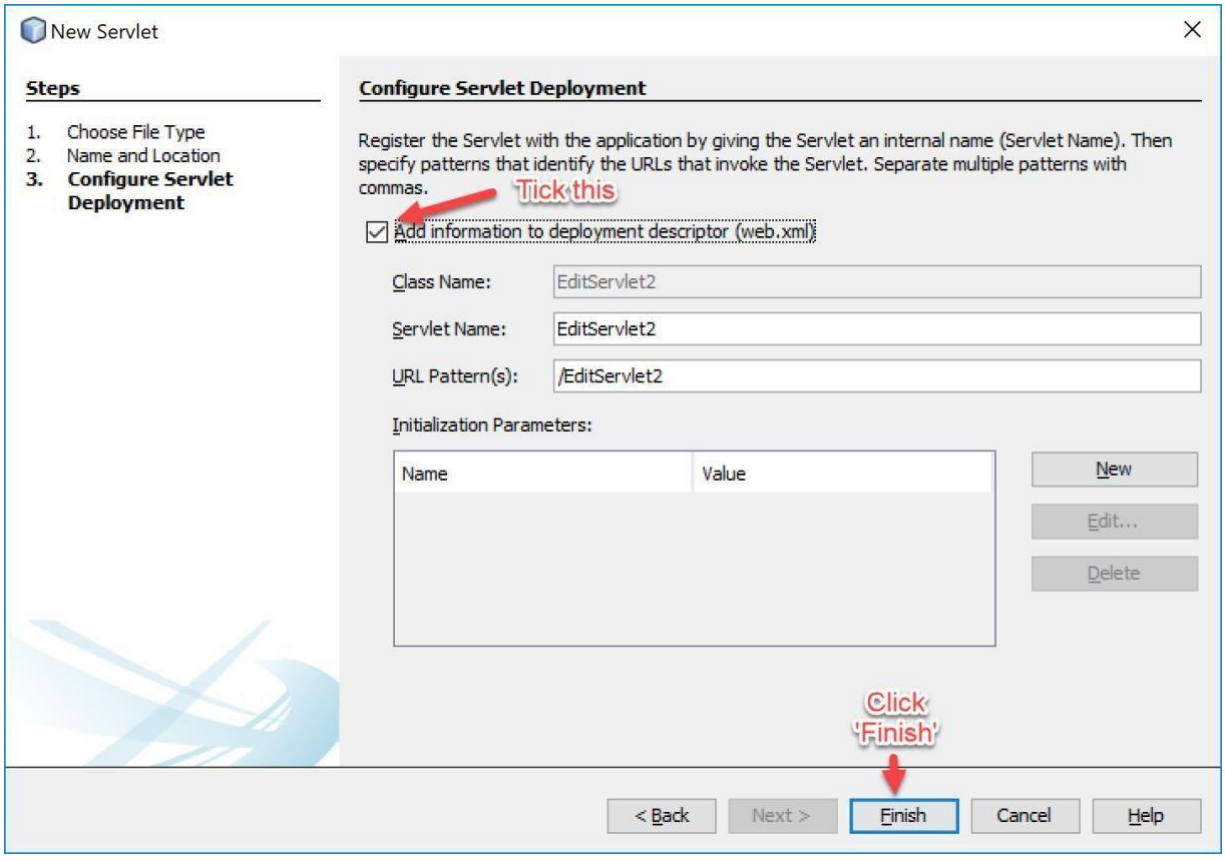
19. The previous servlet for editing data is for displaying the form as below:



20. Now, we need another servlet to handle database processing.



21. Remember to tick 'Add information to deployment descriptor (web.xml)'. Netbeans will configure the servlet in web.xml on your behalf.

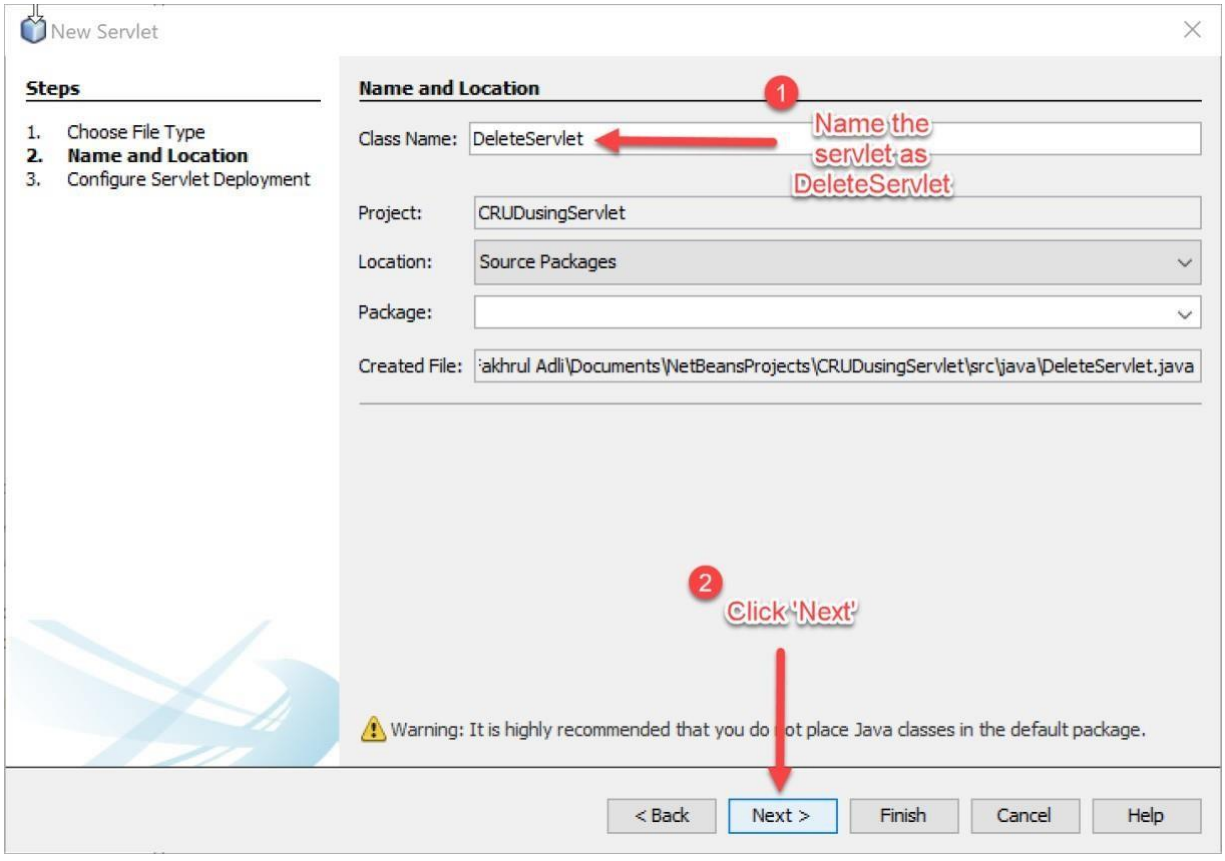


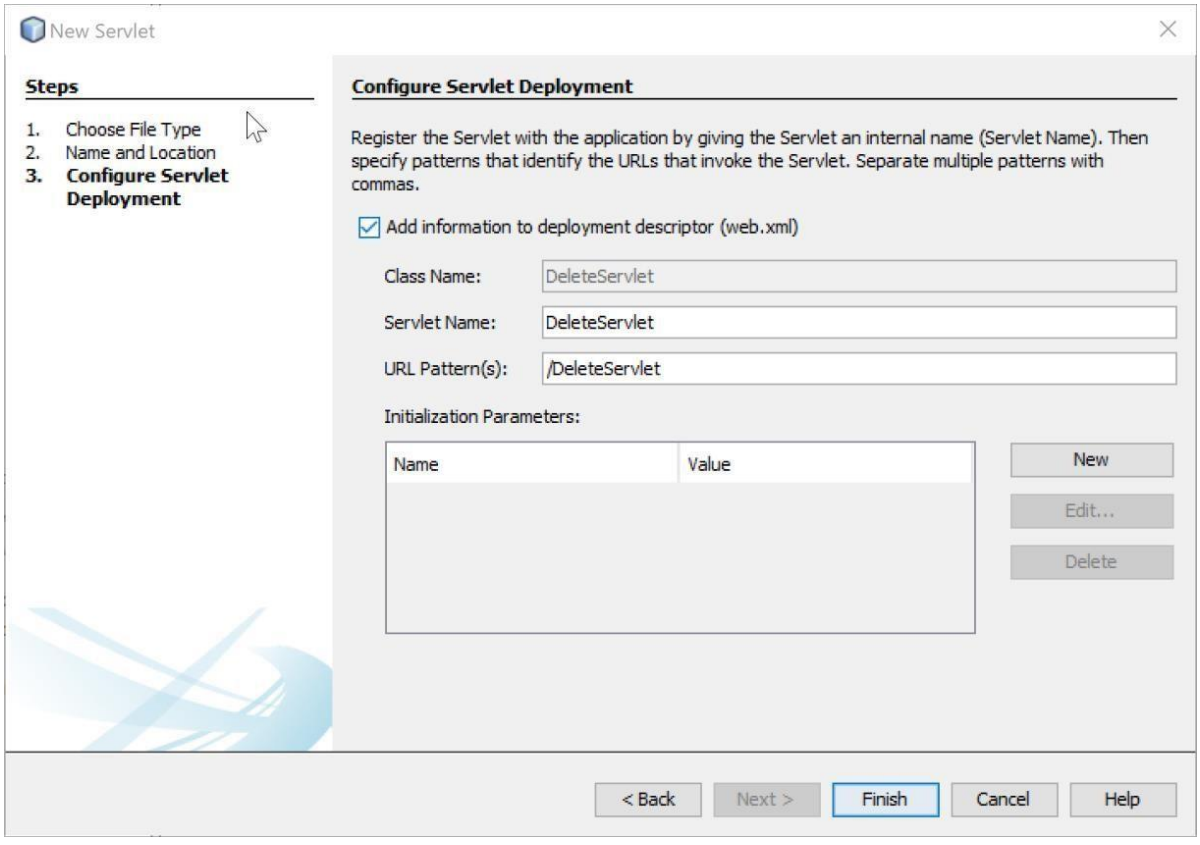
22. Type the following codes in the processRequest() method in EditServlet2.java. Save the file when you have finished.

```
29      protected void processRequest(HttpServletRequest request, HttpServletResponse response)
30              throws ServletException, IOException {
31          response.setContentType("text/html");
            PrintWriter out = response.getWriter();
33
34          String name = request.getParameter("name");
35          String password = request.getParameter("password");
36          String role = request.getParameter("role");
37
38          User e = new User();
39          e.setUsername(name);
40          e.setPassword(password);
41          e.setRole(role);
42
43          int status = UserDao.save(e);
44          if (status > 0) {
45              out.print("<p>Record saved successfully!</p>");
46              request.getRequestDispatcher("index.html").include(request, response);
47          } else {
48              out.println("Sorry! unable to save record");
49          }
50
51          out.close();
52      }
```

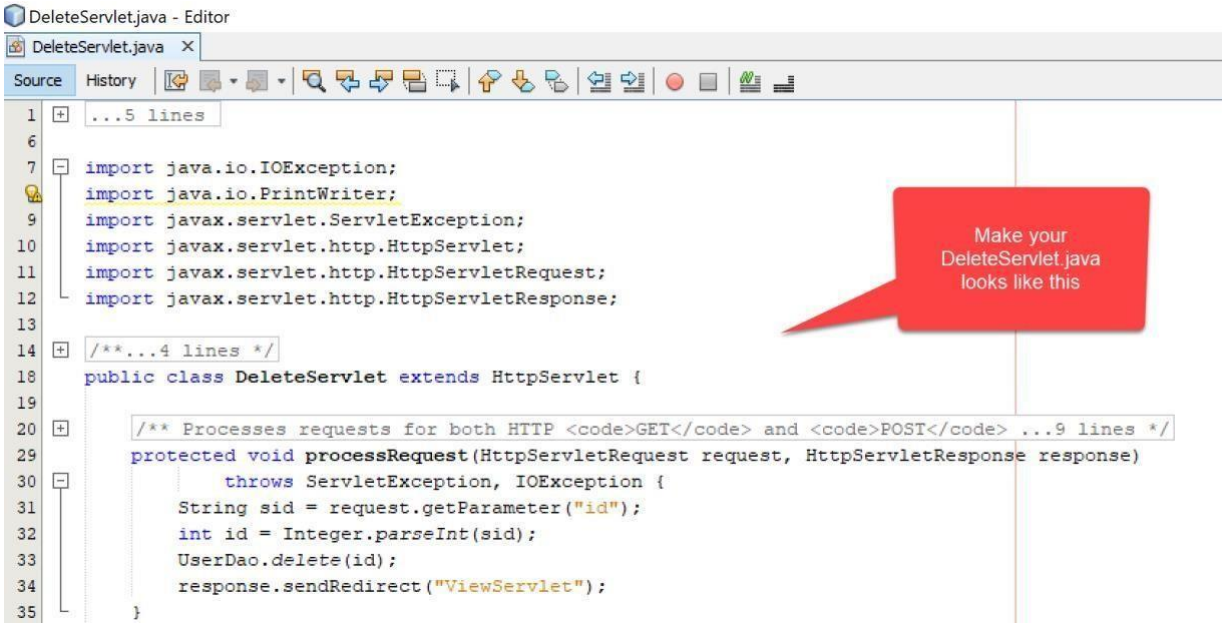
Put these codes in the processRequest() method

23. Now we already have servlets for edit and saving the data, so the last servlet we need to provide is a servlet for deleting the data.





24. Write the codes below into your DeleteServlet.java file. Save the file when you have done.



25. Run all the CRUD process of your servlet applications. In the end, you should be able to (C)reate, (R)etrieve, (U)pdate and (D)elete the data in the database.

Reflections:

1. What is the name of the Java Library that you need to import before coding the web application with database operations?

- JDBC (Java Database Connectivity)

2. Which folder keeps the web.xml file? Copy the contents of the file and explain in brief the tags included such as <servlet-name><servlet-class> <servlet-mapping>. etc.

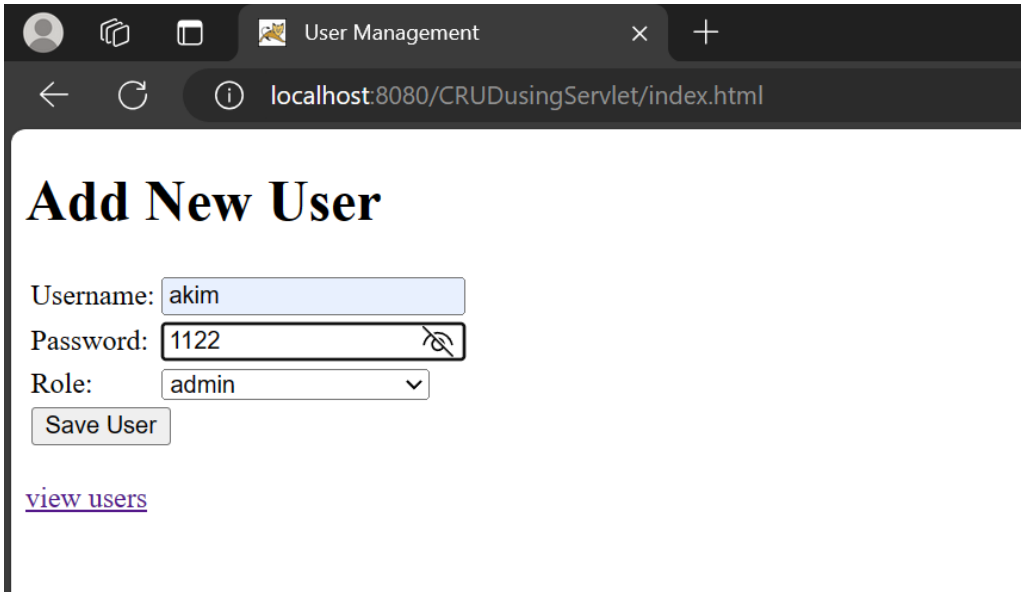
- in web inf folder
- <servlet-name>: Specifies a name for the servlet.
- <servlet-class>: Specifies the fully qualified class name of the servlet.
- <servlet-mapping>: Maps a servlet to a URL pattern.

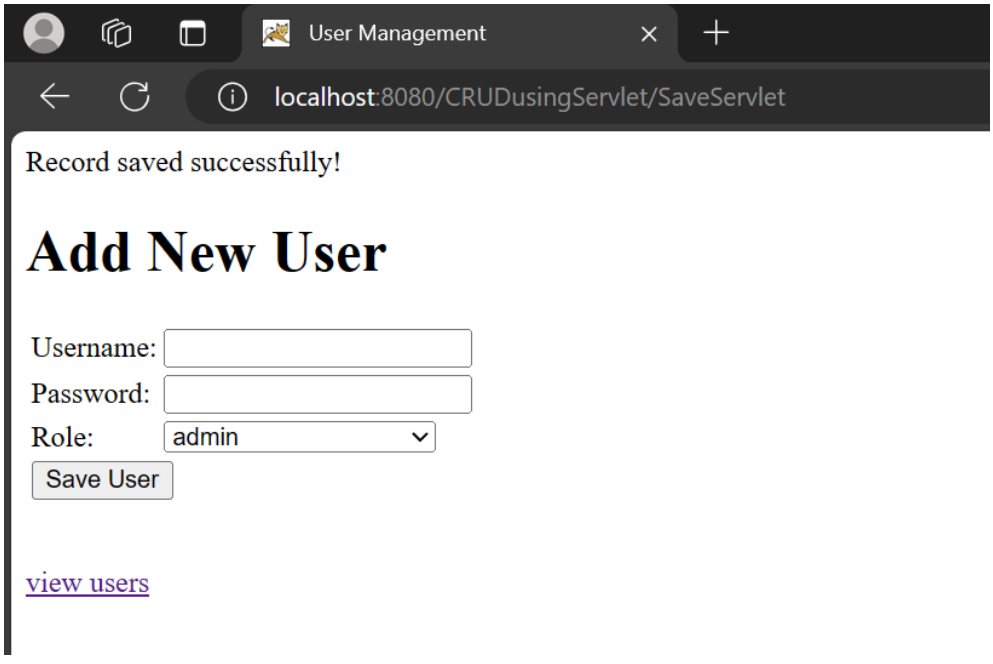
3. Define the usage of Data Access Object (DAO) servlet. How it ease the business process in your servlet-based web application?

- DAO servlet is a design pattern used to separate the business logic of an application from the data persistence logic. DAO servlet acts as an intermediary between the servlets and the database. It encapsulates all the database-related operations the querying, updating, inserting, and deleting data.

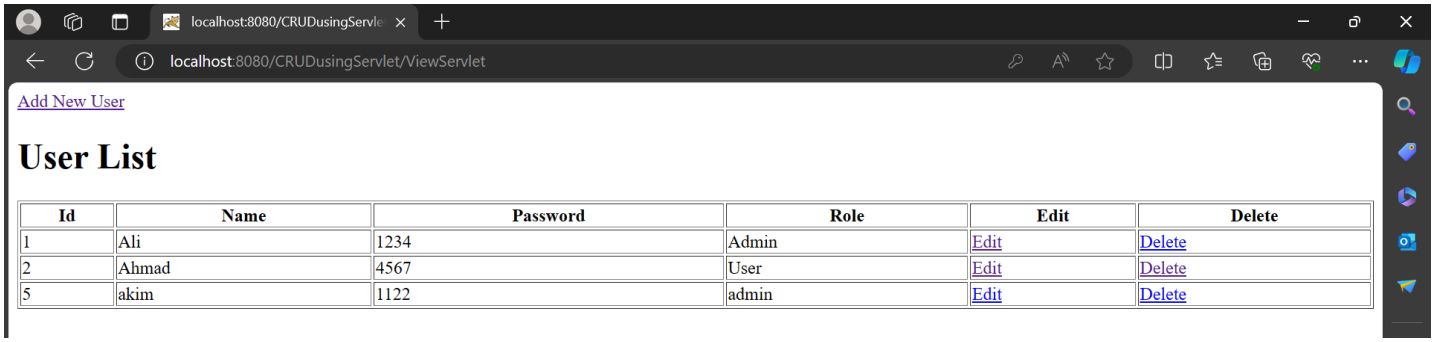
OUTPUT

Add user :

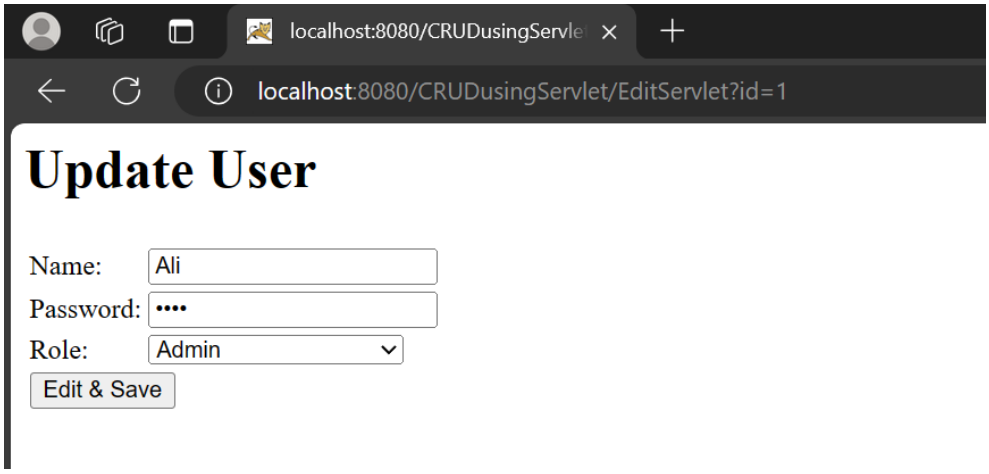


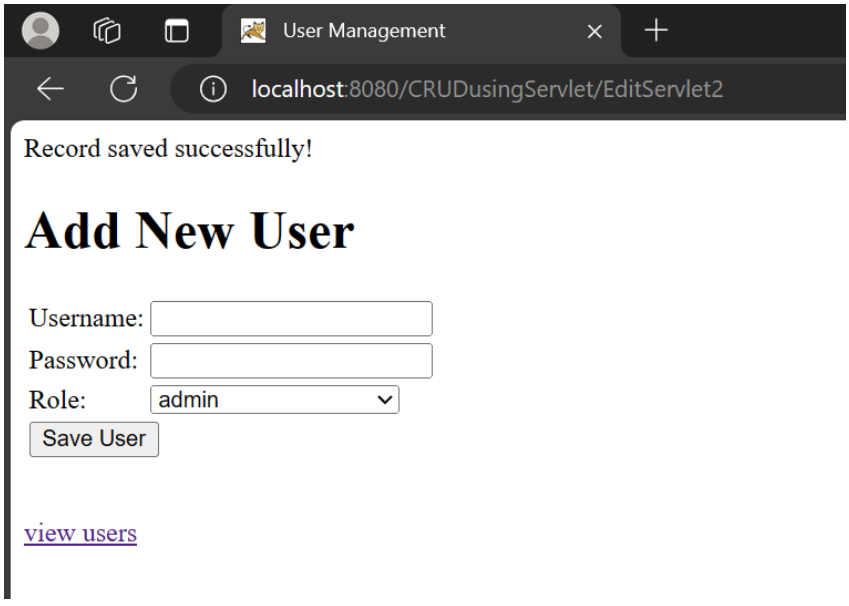


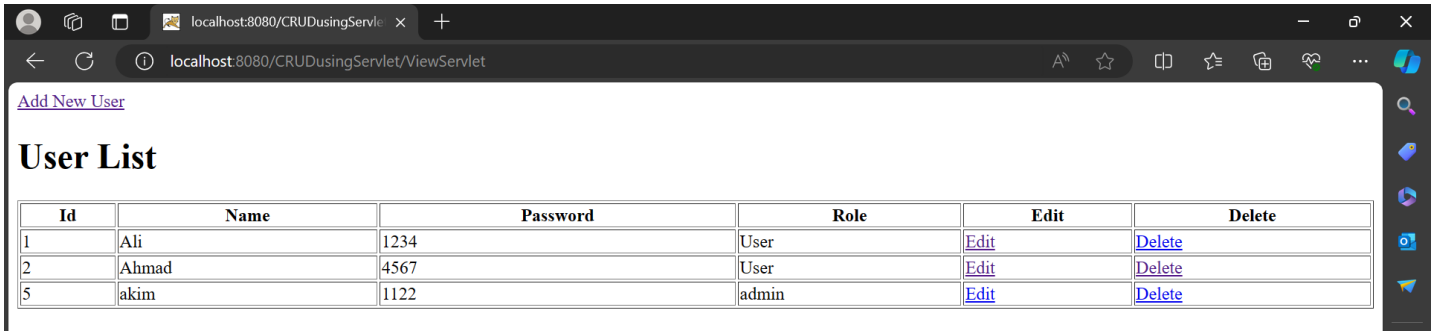
View user :



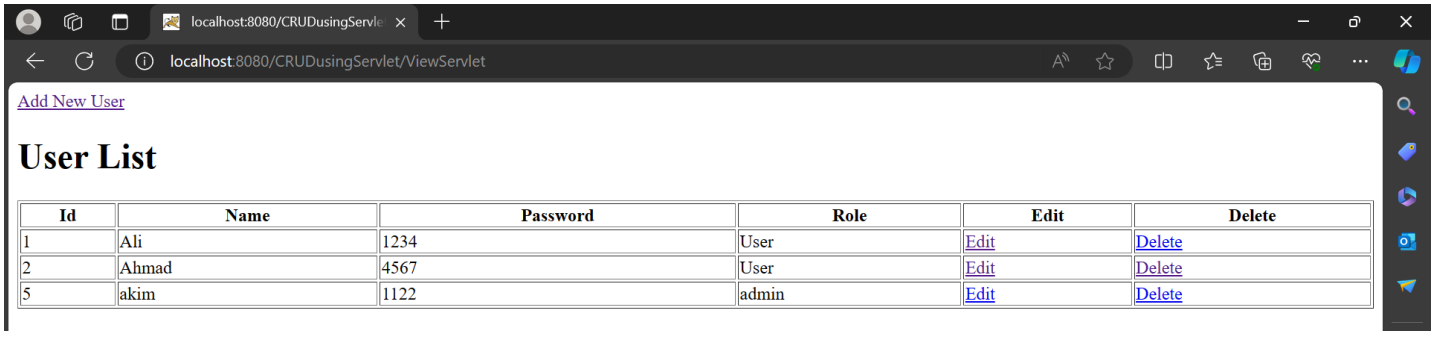
Edit user :

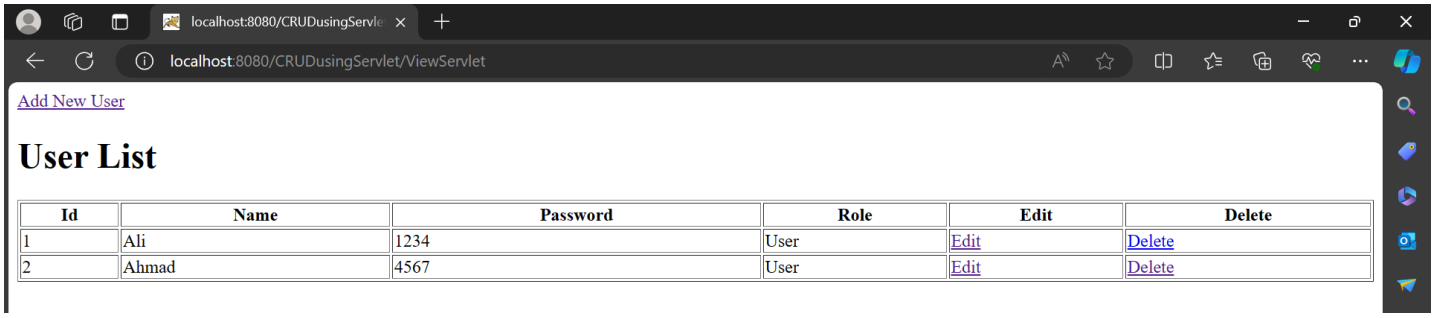





localhost:8080/CRUDusingServle
localhost:8080/CRUDusingServlet/ViewServlet
Add New User
User List
Id Name Password Role Edit Delete
1 Ali 1234 User Edit Delete
2 Ahmad 4567 User Edit Delete
5 akim 1122 admin Edit Delete

Delete user :


localhost:8080/CRUDusingServle
localhost:8080/CRUDusingServlet/ViewServlet
Add New User
User List
Id Name Password Role Edit Delete
1 Ali 1234 User Edit Delete
2 Ahmad 4567 User Edit Delete
5 akim 1122 admin Edit Delete


localhost:8080/CRUDusingServle
localhost:8080/CRUDusingServlet/ViewServlet
Add New User
User List
Id Name Password Role Edit Delete
1 Ali 1234 User Edit Delete
2 Ahmad 4567 User Edit Delete