UNIVERSITI MALAYSIA TERENGGANU

CSM3023

BACHELOR OF COMPUTER SCIENCE (MOBILE COMPUTING) WITH HONORS

LAB 2

SEMESTER II 2023/2024

Prepared for:

DR. MOHAMMAD NOR BIN HASSAN

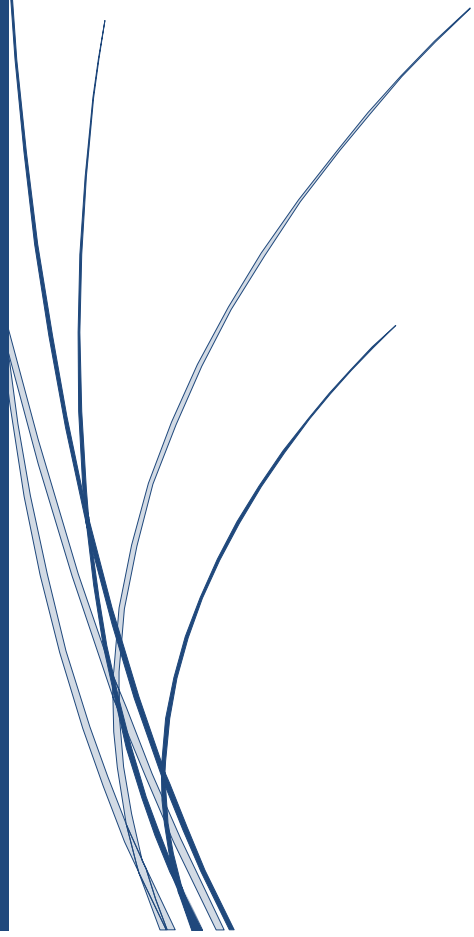
Prepared by:

HUSNA ZAHIRA BINTI RUZELI (S67554)
(K1)

Week 2

Servlet: Data Sharing and Database Management

Web Programming 2



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

Revision History

Revision Date	Previous Revision Date	Summary of Changes	Changes Marked
		First Issue	Mohamad Nor Hassan
		Second Issue	Dr Rabiei Mamat Dr Faizah Aplop Dr Fouad Ts Dr Rosmayati Mohemad Fakhrul Adli Mohd Zaki
13/03/2019	21/02/2019	Addition of Revision History, Table of Contents, Formatting Cover Page	Fakhrul Adli Mohd Zaki
15/03/2019		Add Task1, Task 2, Task 3 and Task 4	Fakhrul Adli Mohd Zaki

Table of Contents

Task 1: Data Sharing in Servlet 5

Task 2: Creating A Table in MySQL Database 15

Task 3: Setting the Environment of Web Application for Database Connection 20

Task 4: Using Servlets for Database CRUD Operations.....23

Arahan:

Manual makmal ini adalah untuk kegunaan pelajar-pelajar Pusat Pengajian Informatik dan Matematik Gunaan (PPIMG), Universiti Malaysia Terengganu (UMT) sahaja. Tidak dibenarkan mencetak dan mengedar manual ini tanpa kebenaran rasmi daripada penulis.

Sila ikuti langkah demi langkah sebagaimana yang dinyatakan di dalam manual. Tandakan (√) setiap langkah yang telah selesai dibuat dan tulis kesimpulan bagi setiap aktiviti yang telah selesai dijalankan.

Instruction:

This laboratory manual is for use by the students of the School of Informatics and Applied Mathematics (PPIMG), Universiti Malaysia Terengganu (UMT) only. It is not permissible to print and distribute this manual without the official authorisation of the author.

Please follow step by step as described in the manual. Tick (ã) each step completed and write the conclusions for each completed activity.

Task 1: Data Sharing in Servlet

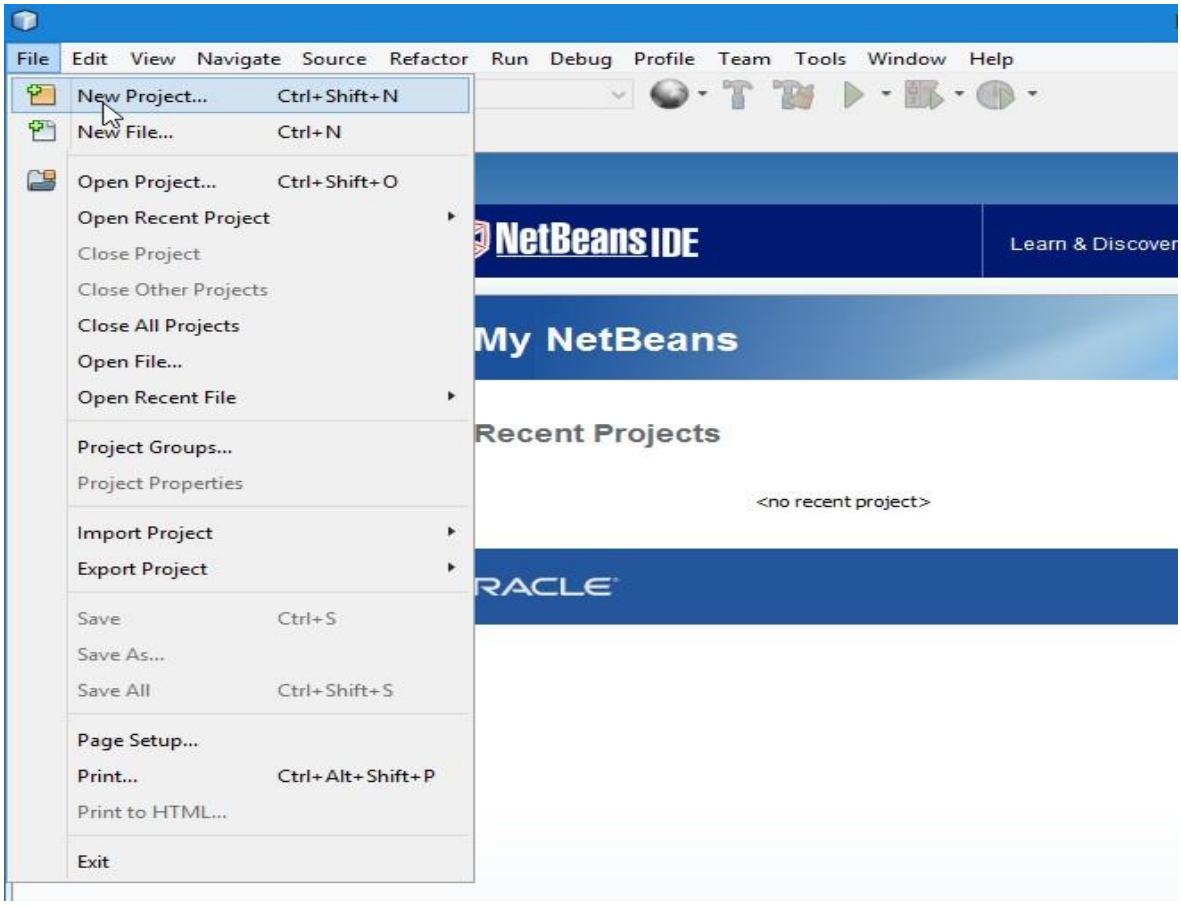
Objective:	To use servlet for request forwarding and data sharing
Problem Description:	Write a login form and a servlet to authenticate a user.
Estimated time:	30 minutes

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

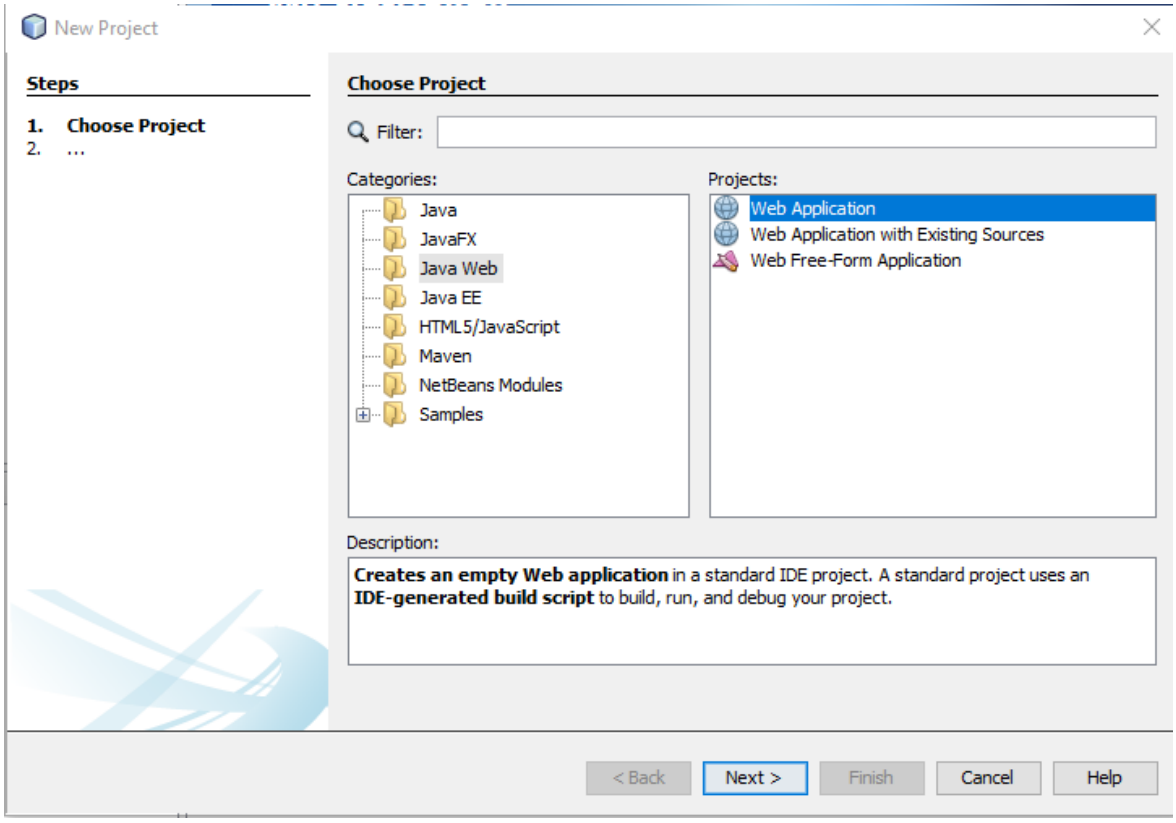
2. Go to F:\CSF3107-SXXXXX\ directory and create sub-directory and name it as *Lab 2*.

3. Open your NetBeans IDE.

4. Go to File -> New Project

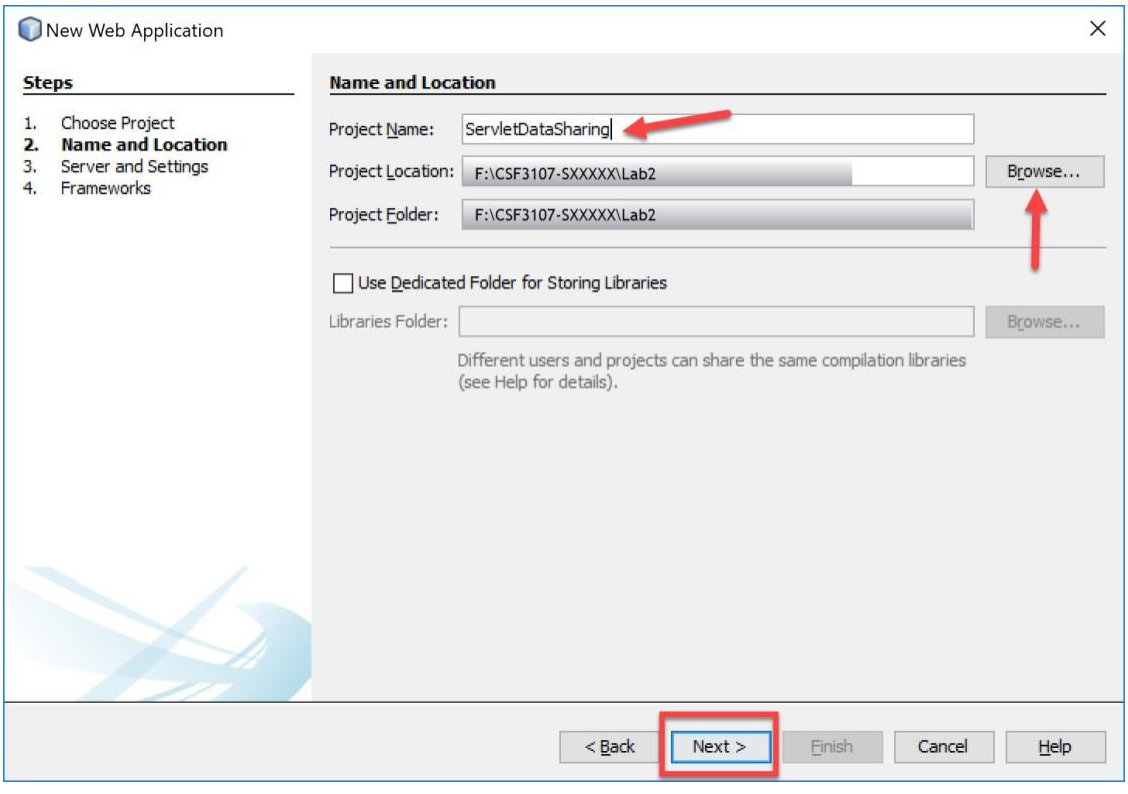


5. Select Java Web -> Web Application and click Next.

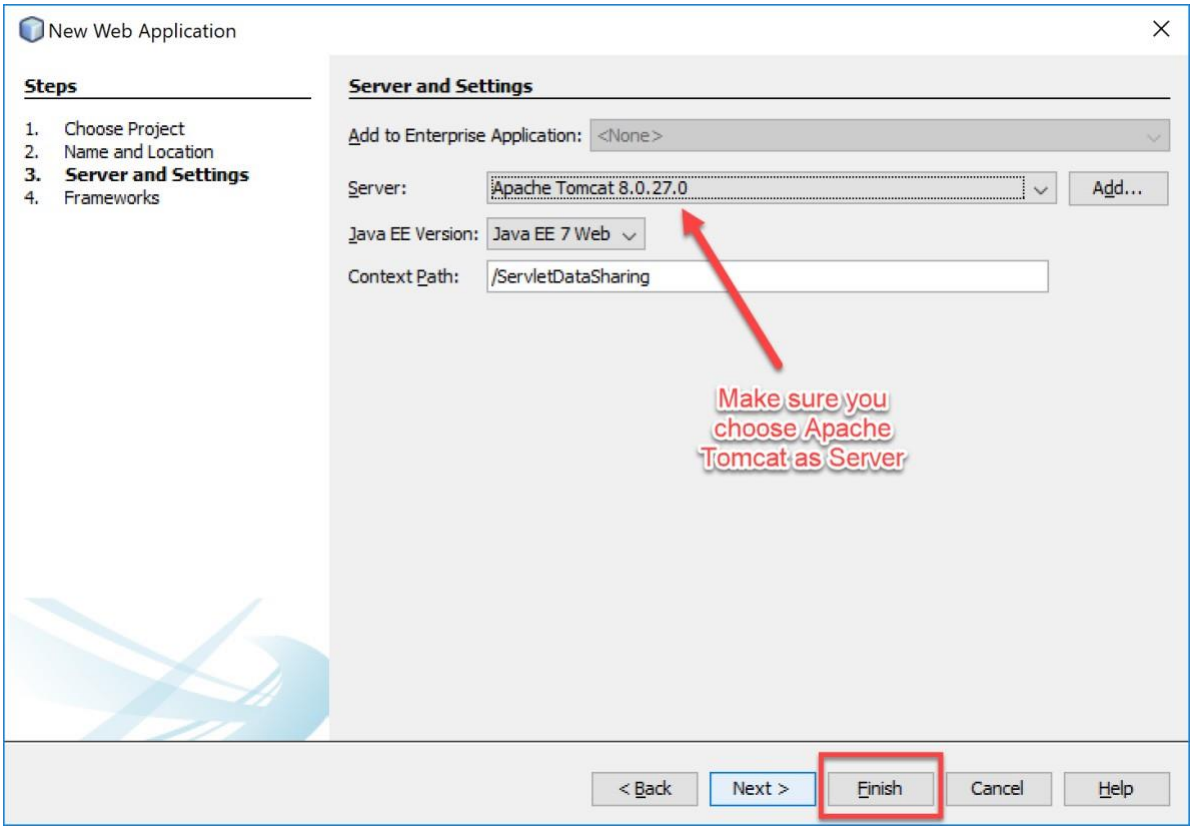


6. Type Project Name: *ServletDataSharing*.

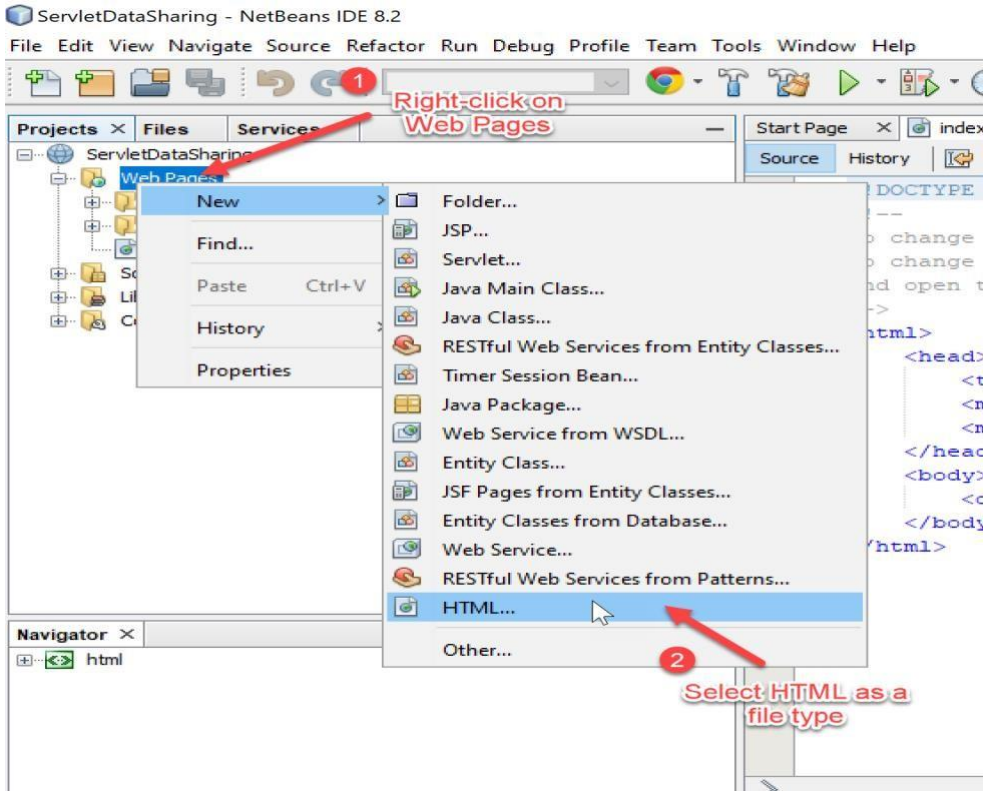
7. Click Browse and choose Project Location: F:\CSF3107-SXXXXX\Lab2. Then click the Next button.



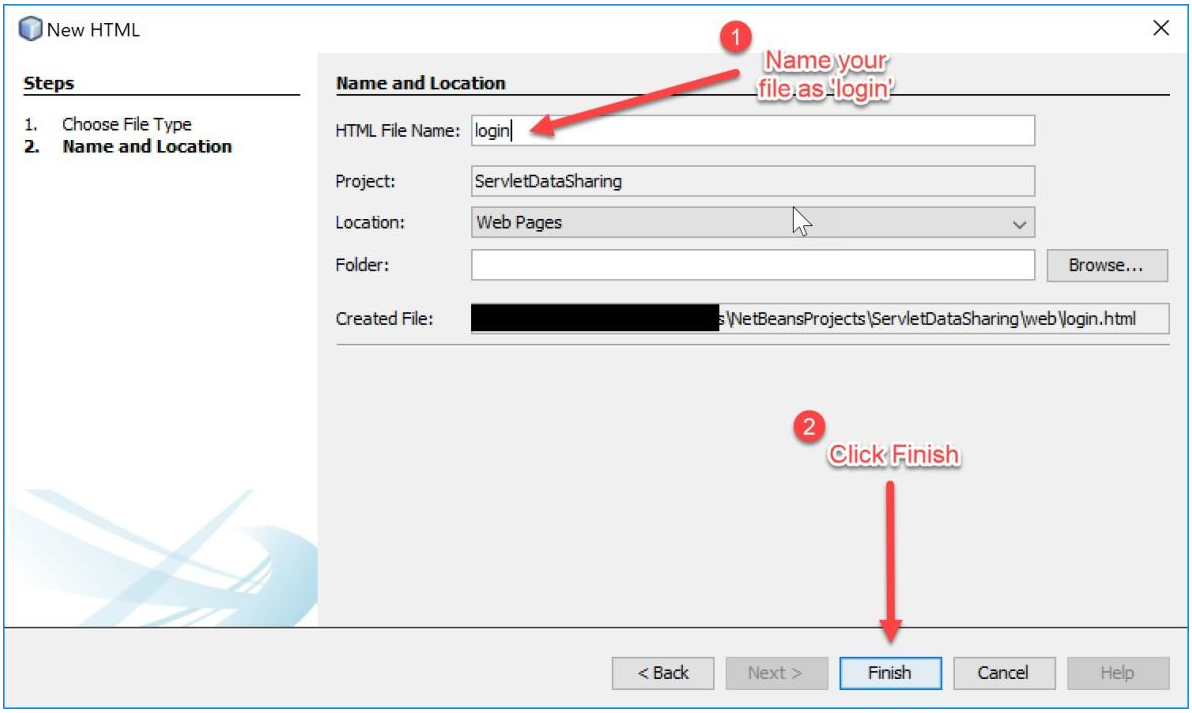
8. Then, choose Apache Tomcat as the server and click Finish.



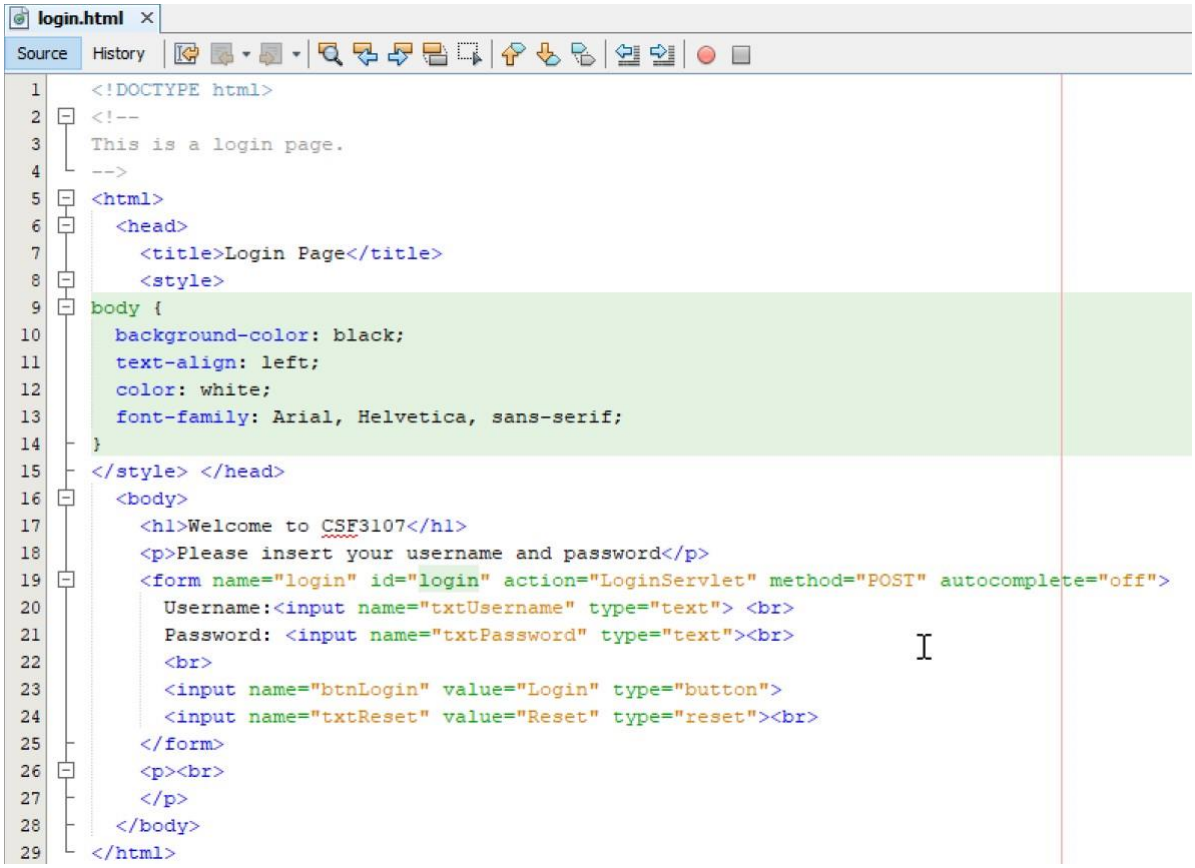
9. You will see your project structure on the left side of your Netbeans editor. Next, you will create an HTML file for a login page. Follow the instructions as shown on the following screenshot.



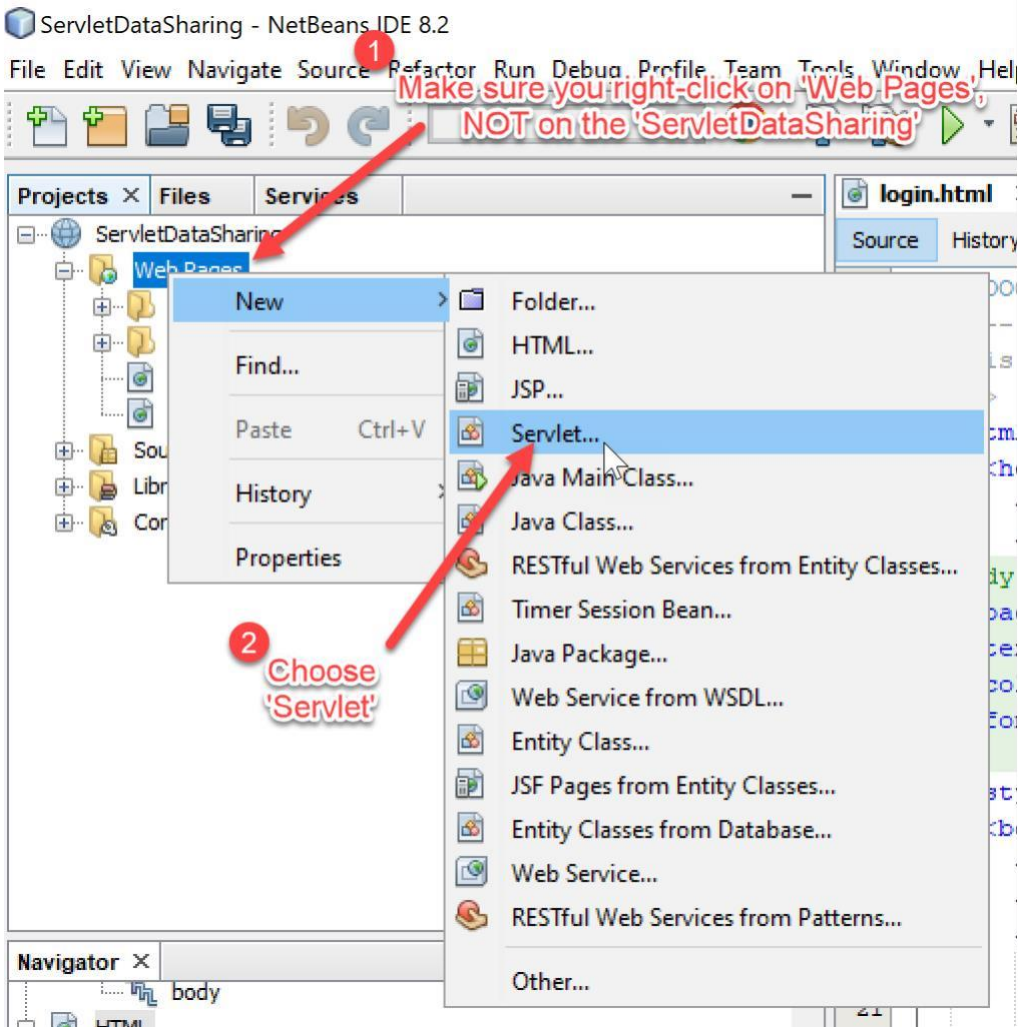
10. Name your HTML file as ‘ l o g then click Finish.

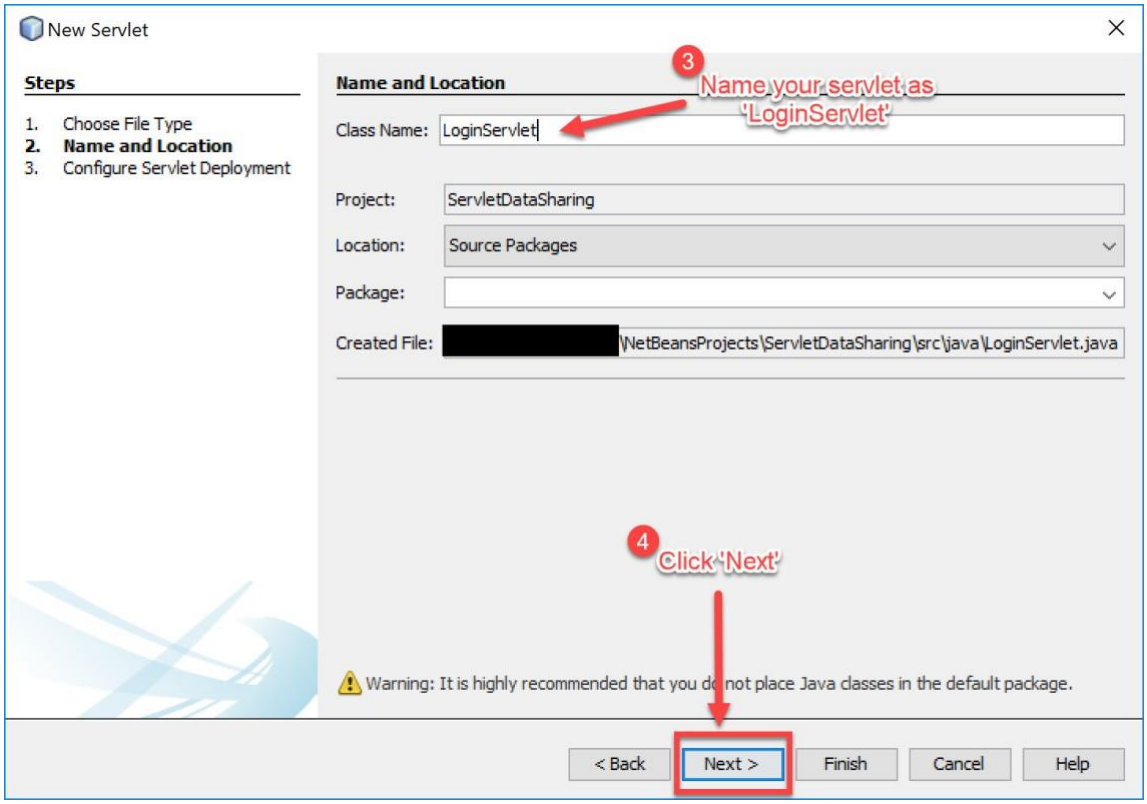


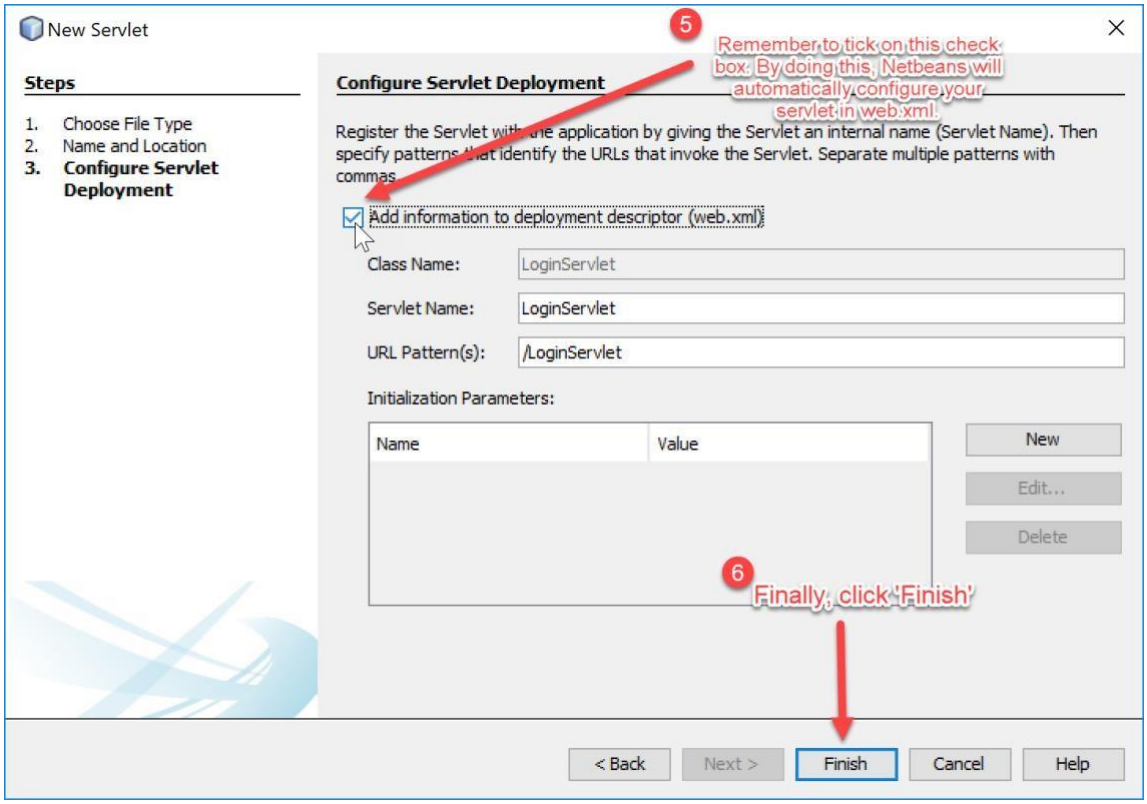
11. Type the following HTML markups into your login.html.

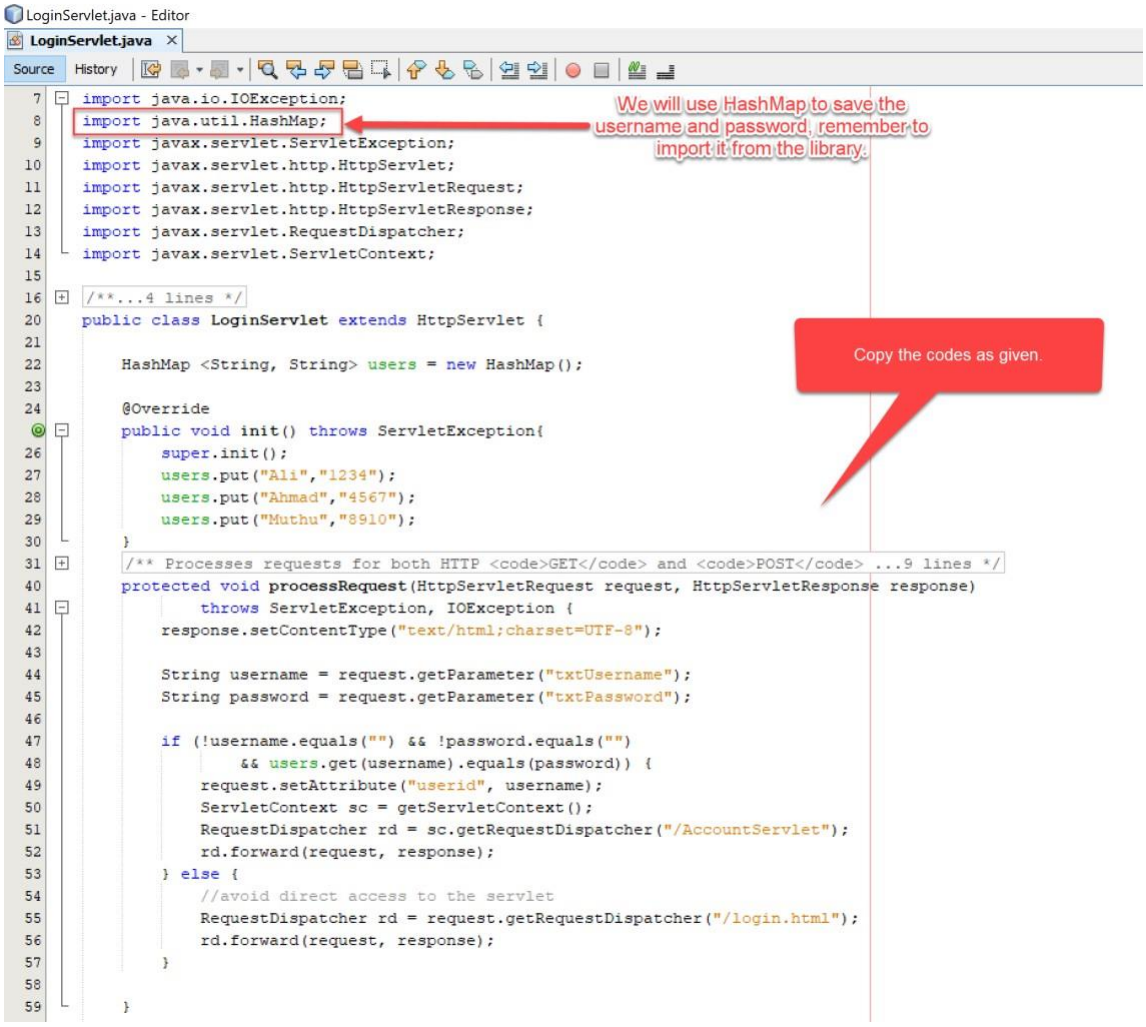


12. Next, we are going to create a servlet to process the username and password insert by the user on the login form. We start it by creating a new file for our servlet. Follow the steps below:



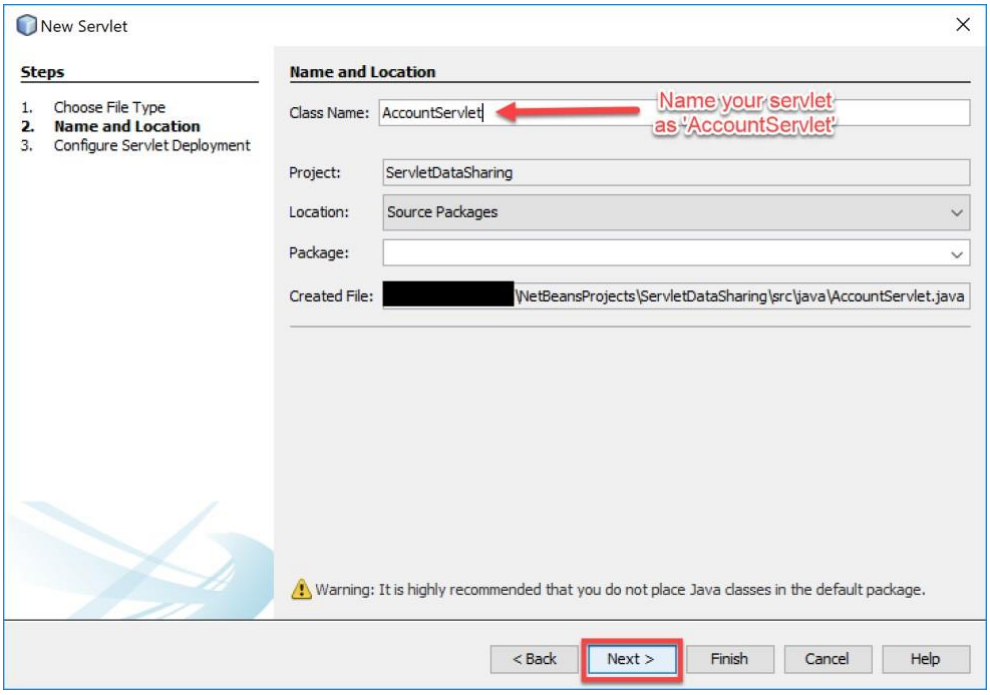

New Servlet
Steps
1. Choose File Type
2. Name and Location
3. Configure Servlet Deployment
Name and Location
3
Name your servlet as 'LoginServlet'
Class Name: LoginServlet
Project: ServletDataSharing
Location: Source Packages
Package:
Created File: [redacted]\NetBeansProjects\ServletDataSharing\src\java\LoginServlet.java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

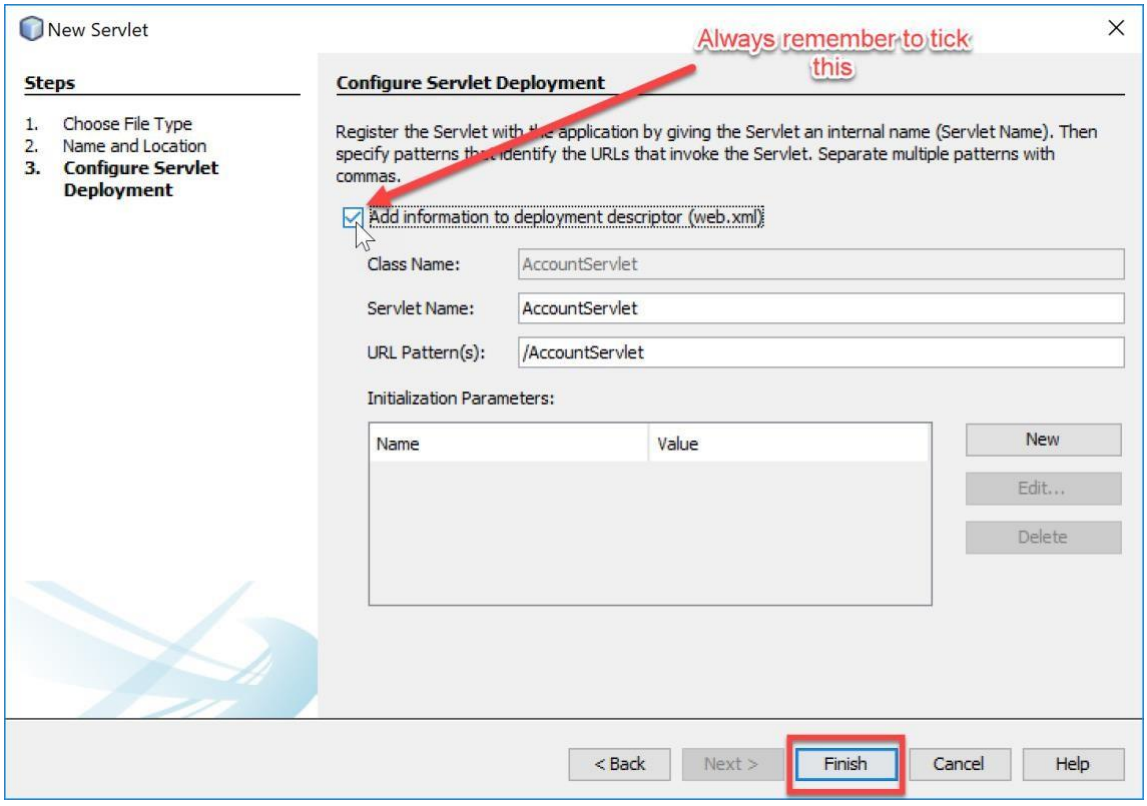

New Servlet
Steps
1. Choose File Type
2. Name and Location
3. Configure Servlet Deployment
5
Remember to tick on this check box. By doing this, Netbeans will automatically configure your servlet in web.xml
Configure Servlet Deployment
Register the Servlet with the application by giving the Servlet an 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

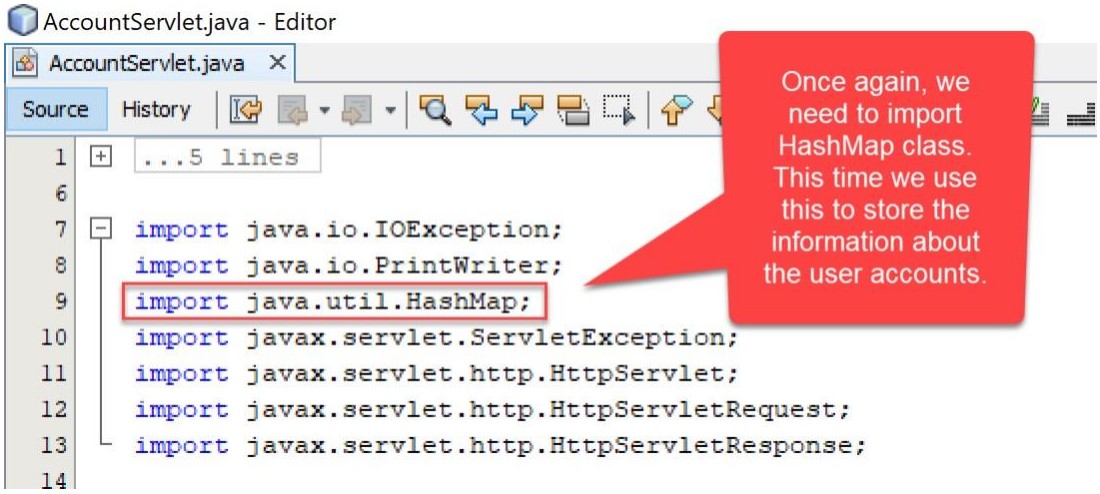



```
import java.io.IOException;
import java.util.HashMap;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.RequestDispatcher;
import javax.servlet.ServletContext;

/**...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 informa 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

```
15  /**...4 lines */
19  public class AccountServlet extends HttpServlet {
20      HashMap<String, String[]> account = new HashMap();
21      @Override
23      public void init() throws ServletException{
24          super.init();
25          account.put("Ali", new String[]{"31/01/2019: 2000.00","28/02/2019: 3000.00"});
26          account.put("Ahmad", new String[]{"31/01/2019: 100.00","28/02/2019 5000.00"});
27          account.put("Muthu", new String[]{"31/01/2019: 1000","28/02/2019 2000"});
28      }
29      /** Processes requests for both HTTP <code>GET</code> and <code>POST</code> ...9 lines */
38      protected void processRequest(HttpServletRequest request, HttpServletResponse response)
39              throws ServletException, IOException {
40          response.setContentType("text/html;charset=UTF-8");
41
42          String userid_login = (String)request.getAttribute("userid");
43
44          try (PrintWriter out = response.getWriter()) {
45
46              out.println("<!DOCTYPE html>");
47              out.println("<html>");
48              out.println("<head>");
49              out.println("<title>Servlet AccountServlet</title>");
50              out.println("</head>");
51              out.println("<body>");
52
53              if(account.get(userid_login)==null){
54                  out.println("<h1>Sorry, no information found!</h1>");
55              }
56              else{
57                  out.println("<h1>Account status for: " + userid_login + "</h1>");
58                  for(String tempAcc: account.get(userid_login)){
59                      out.println("<h2>"+tempAcc+"</h2>");
60                  }
61              }
62
63              out.println("</body>");
64              out.println("</html>");
65          }
66      }
```

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

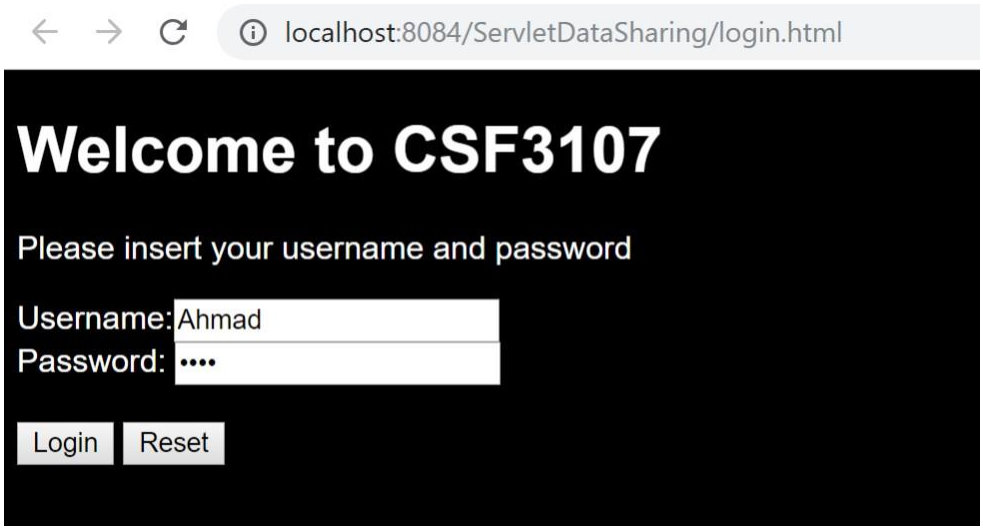
ServletDataSharing
- Web Pages
  - META-INF
  - WEB-INF
  - index.html
  - login.html
- Source
- Test Pa
- Librarie
- Test Lib
- Configu

Navigator
- CSS
- HTML

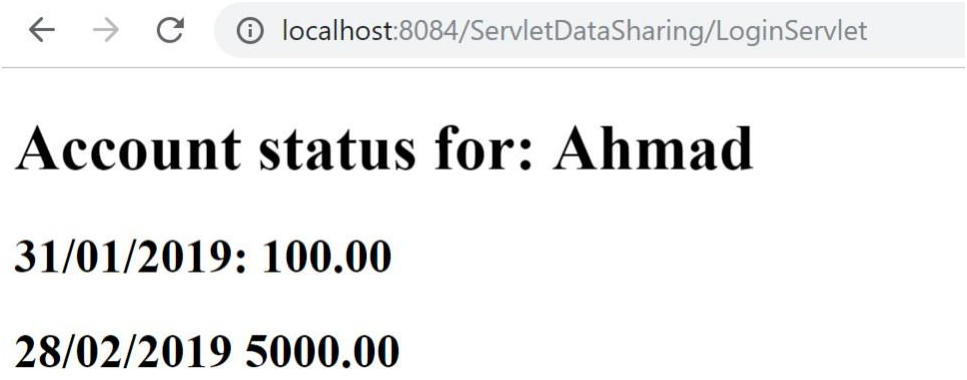
Right-click on login.html and choose 'Run File'

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

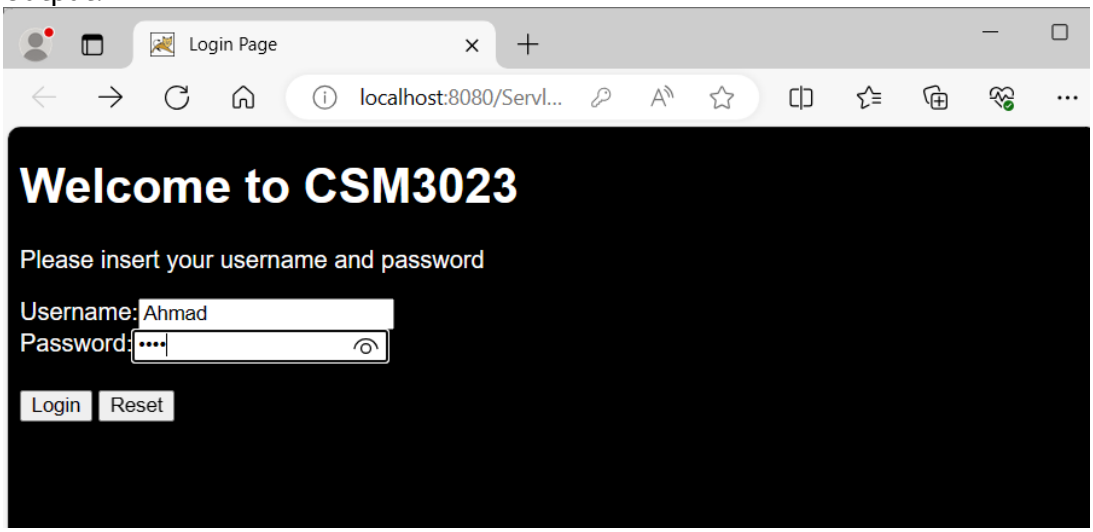
14. The Output will appear in a web browser. Put a Username as ‘ A h m and ’ Password a s ‘ 4 then Click, *Login*.

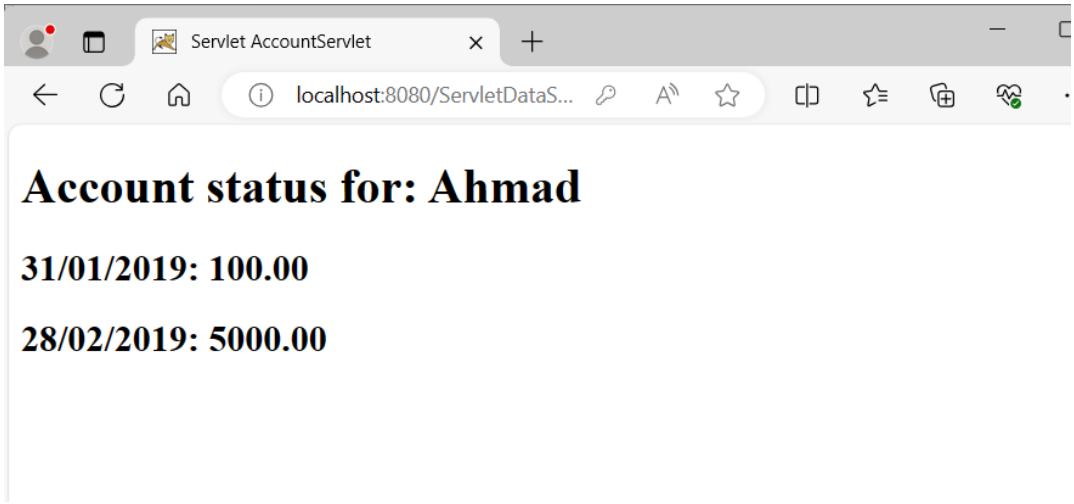


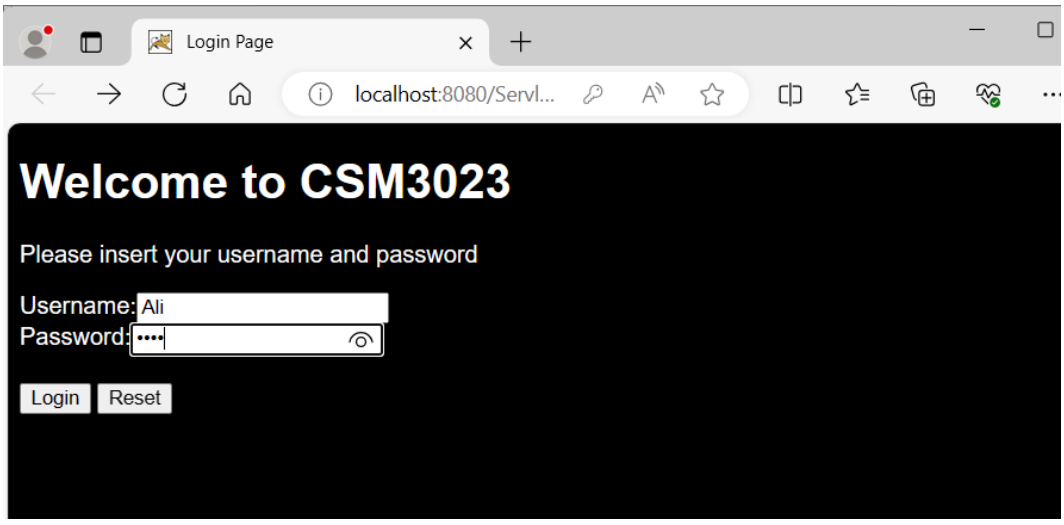
15. If you have followed all the previous steps correctly, you will see the account information of a user named ‘ A h m a Repeat step 14 with other users as input and see the result.

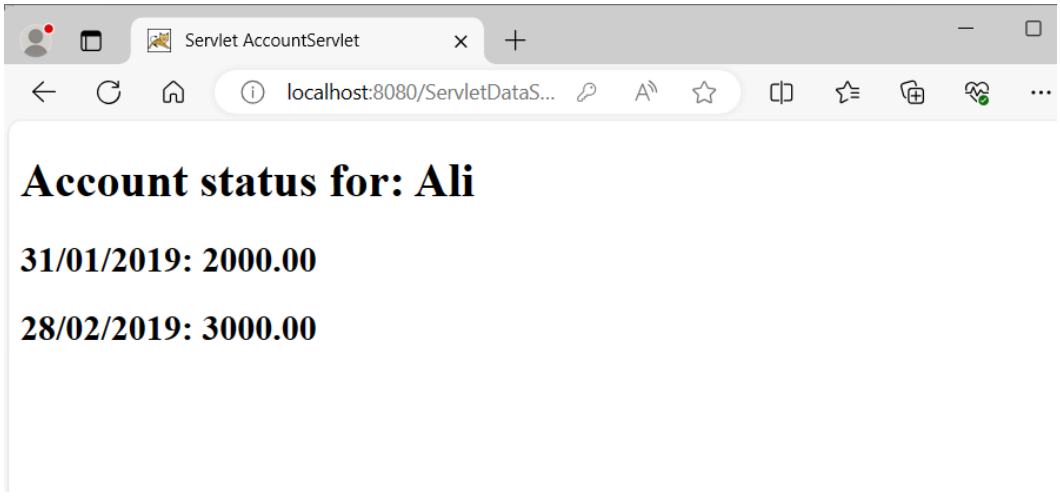


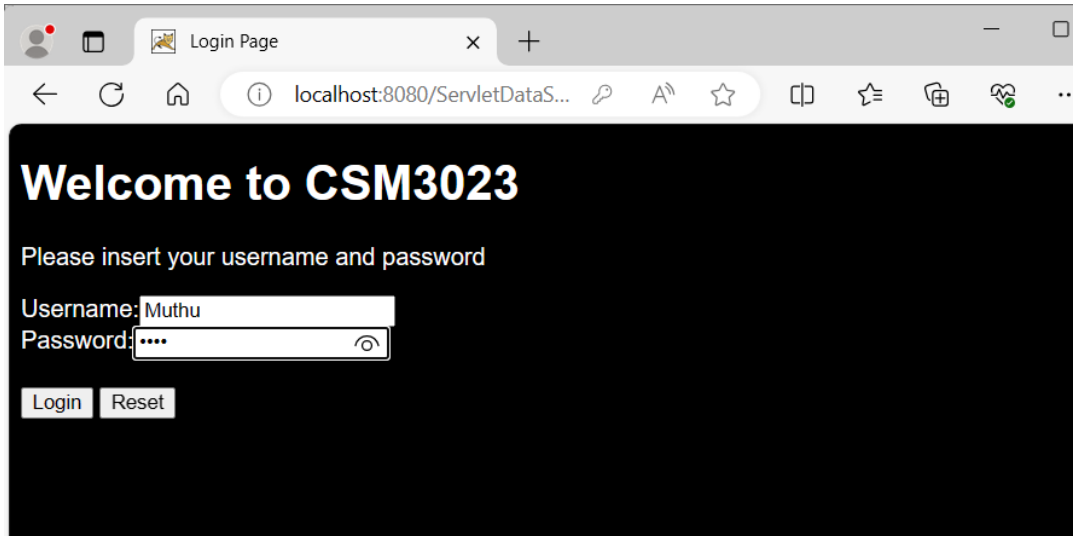
Output:

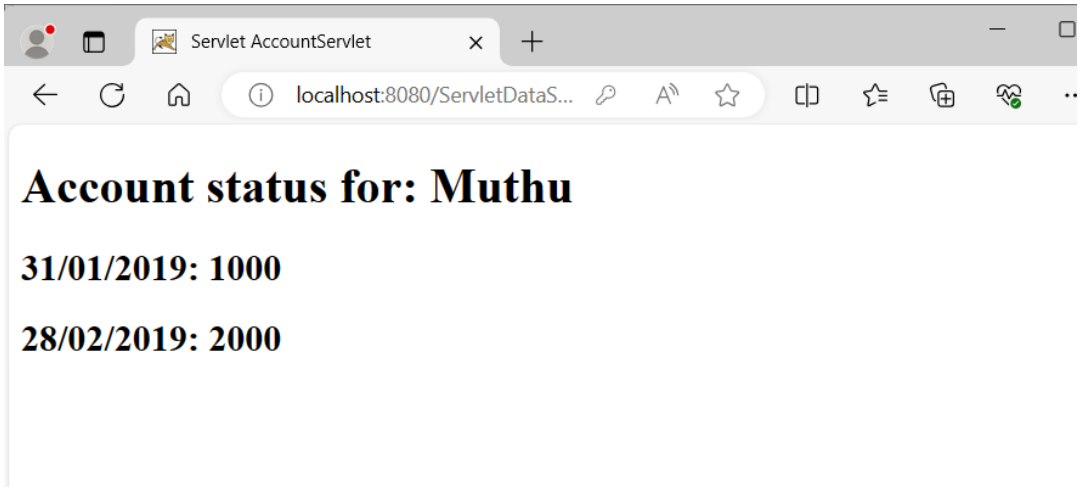



Servlet AccountServlet
localhost:8080/ServletDataS...
Account status for: Ahmad
31/01/2019: 100.00
28/02/2019: 5000.00


Login Page
localhost:8080/Servl...
Welcome to CSM3023
Please insert your username and password
Username: Ali
Password:
Login
Reset


Servlet AccountServlet
localhost:8080/ServletDataS...
Account status for: Ali
31/01/2019: 2000.00
28/02/2019: 3000.00





Reflections:

1. What have you you learnt from this exercise?

   Answer:

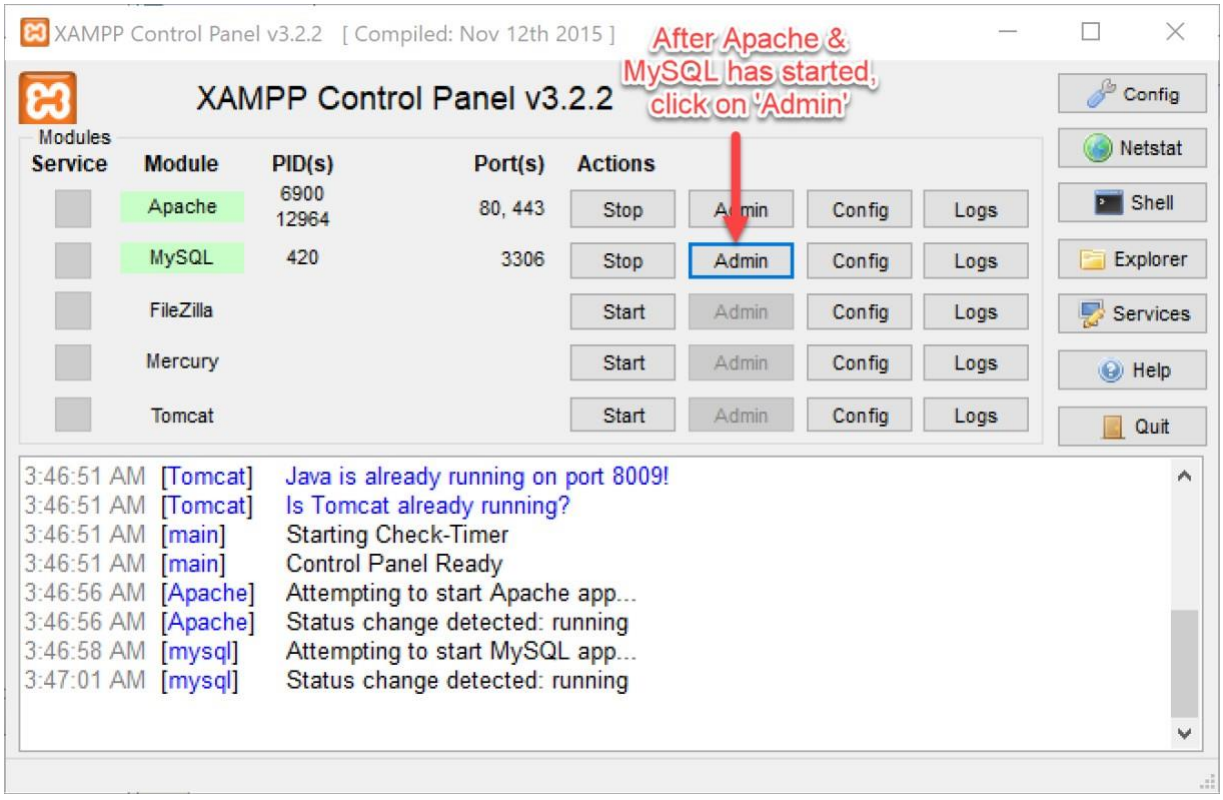
   To make data sharing through using servlet.

2. What are the common methods used in Java Servlet?
   Answer:
   doGet() and doPost() methods.

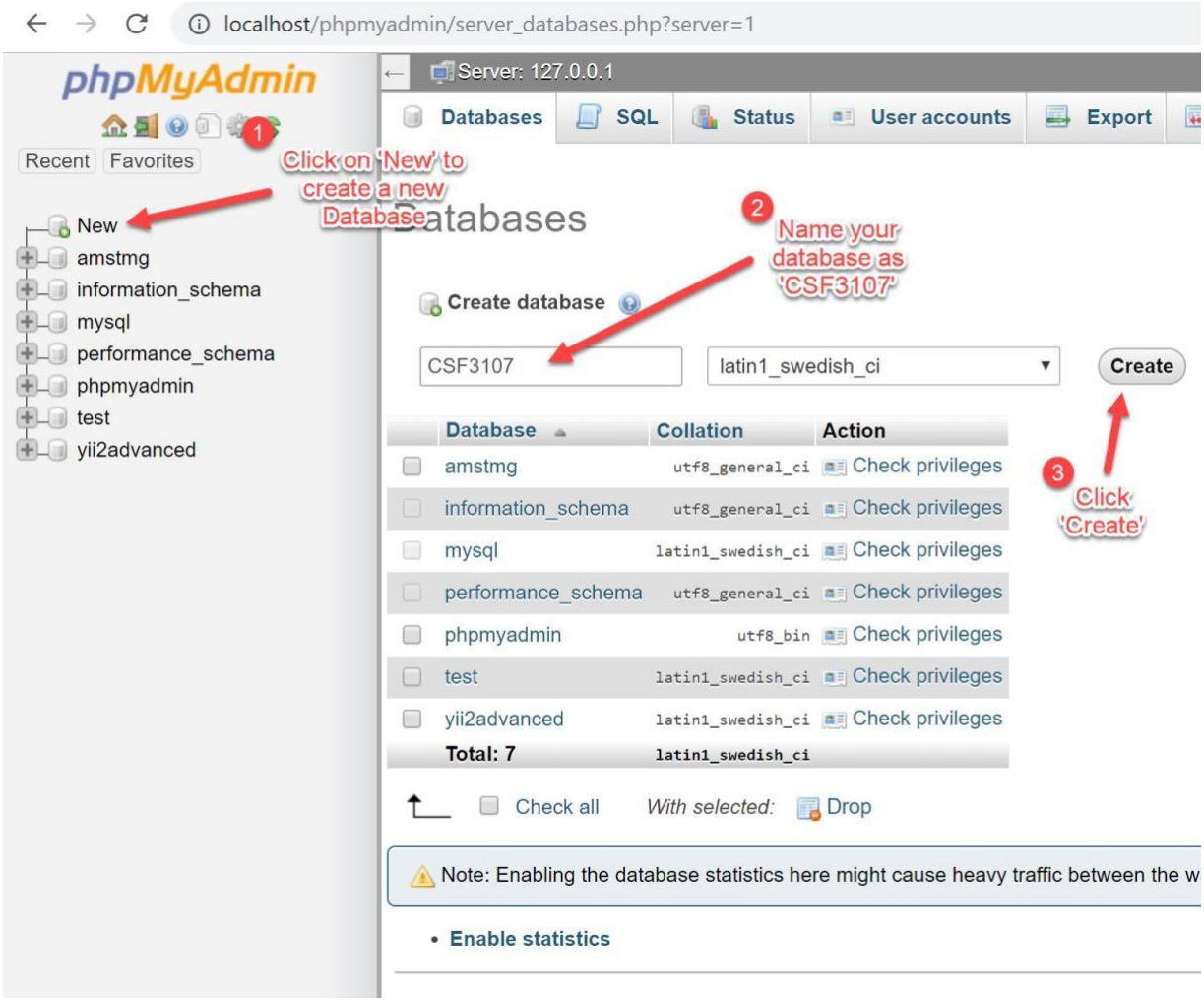
Task 2: Creating A Table in MySQLDatabase

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

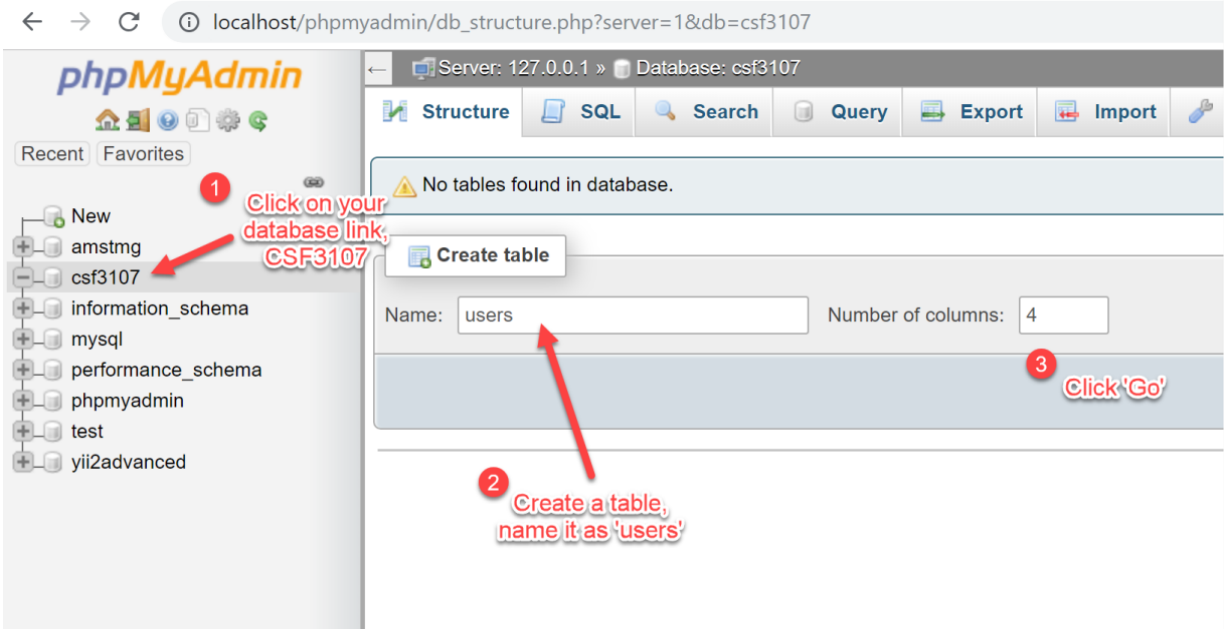
1. Start Apache and MySQL.



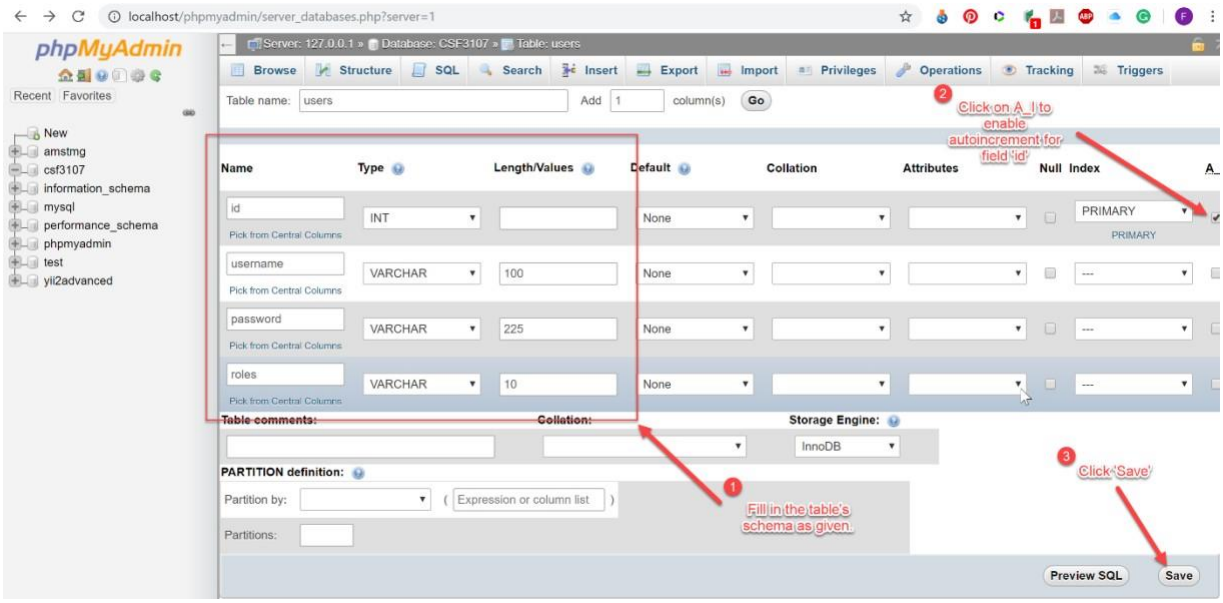
2. Create a new database.



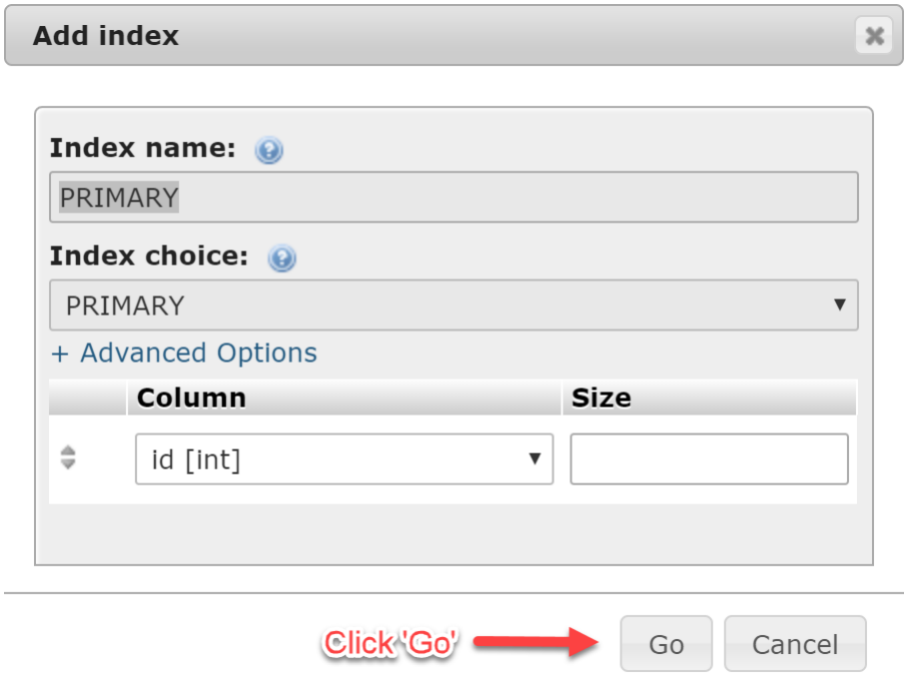
3. After the database has been created, now we are going the create a table named ‘ u s e This table has four columns.



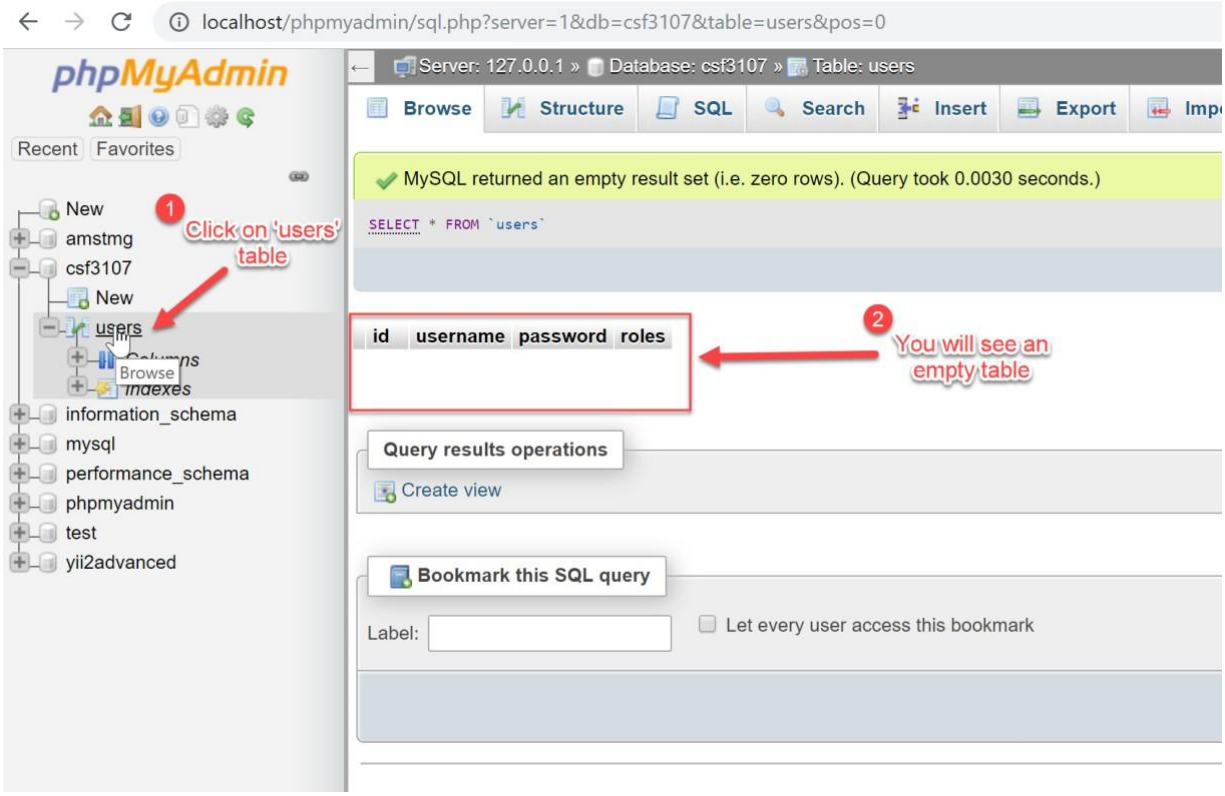
4. Setup the table ‘ u s e r b s elow: a s



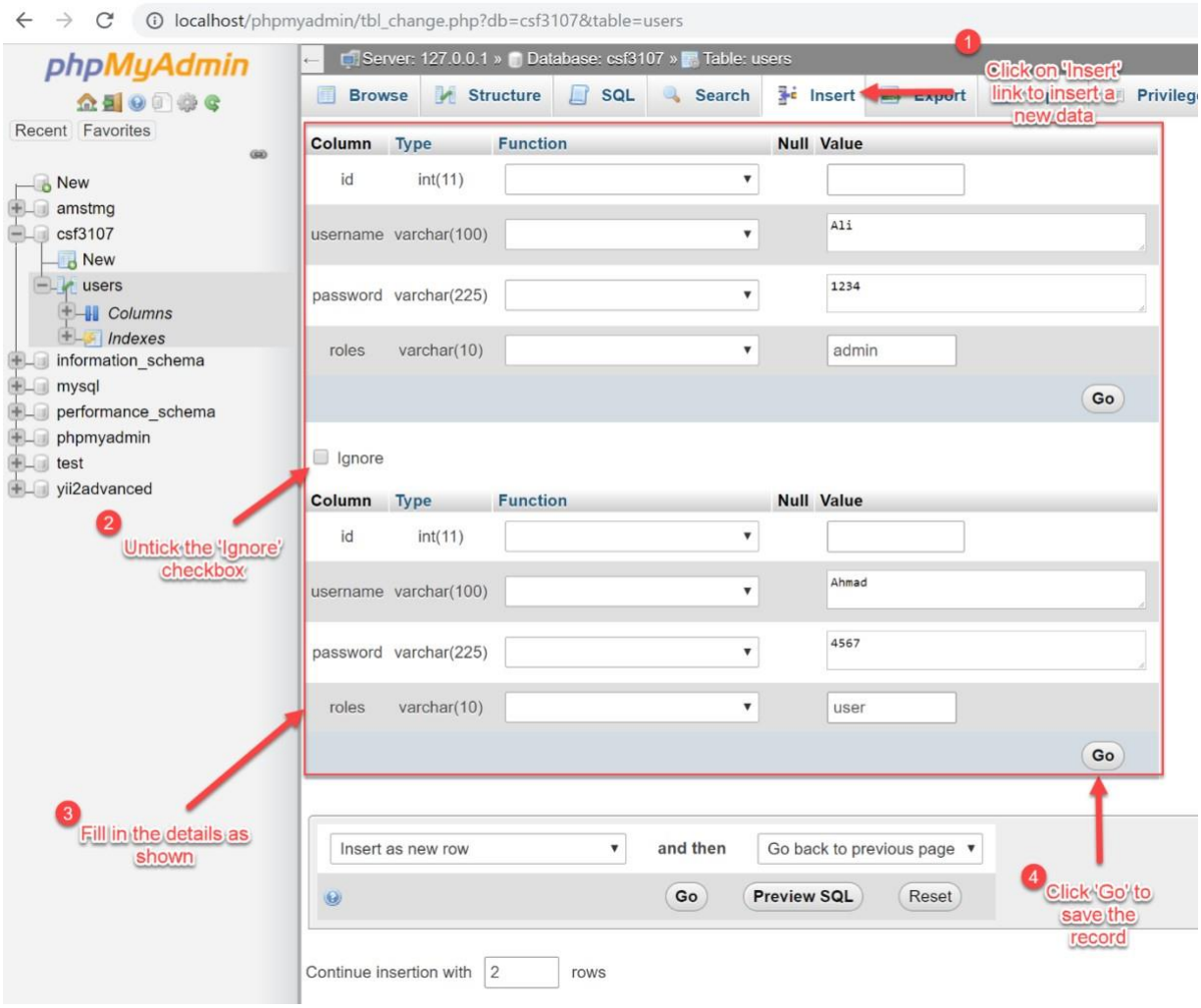
5. After you tick for auto-increment, this dialogue box will appear, leave the settings as it is and click ‘ G o ’ .



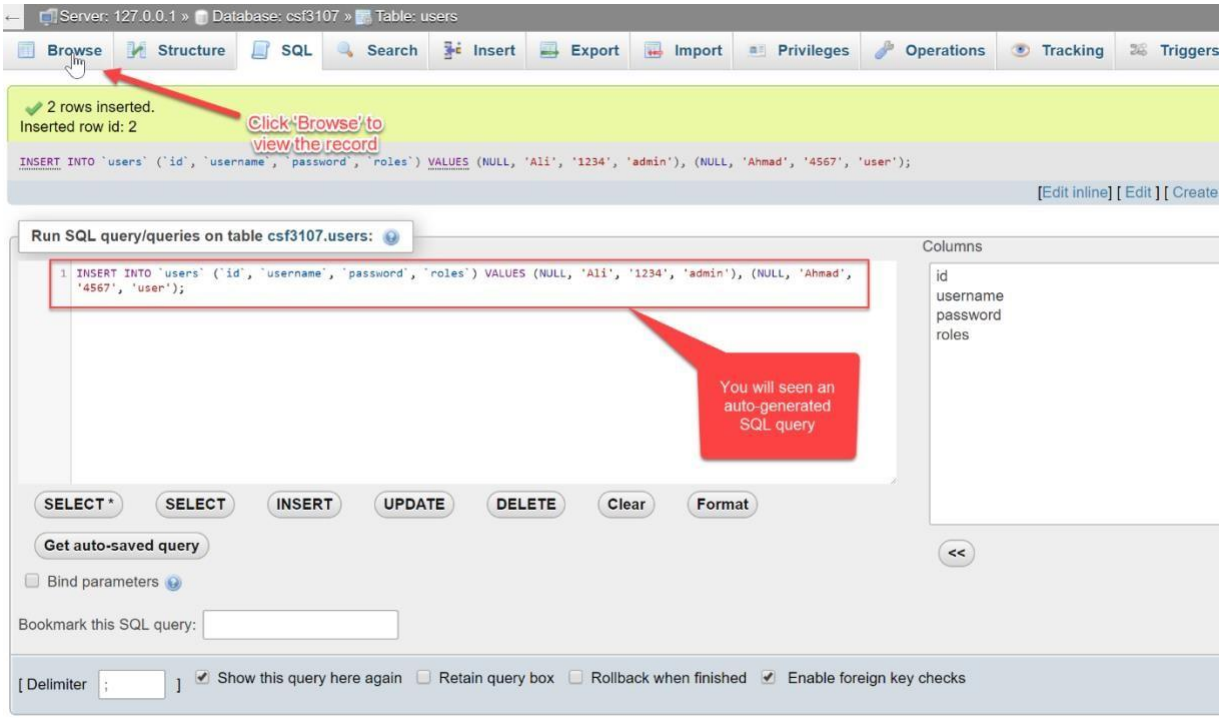
6. Your newly created table can be seen on the left phpMyAdmin.



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

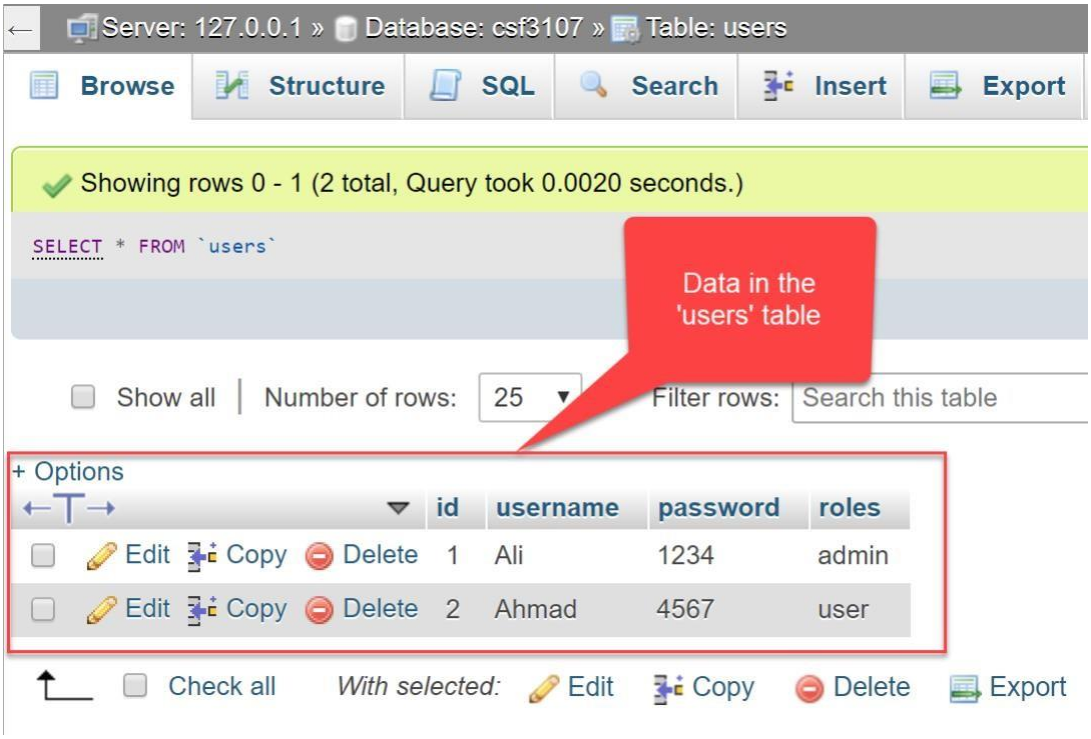


8. After you have clicked the 'Go' button in autogenerated SQL query on the phpMyAdmin page. To view the data, click Browse.

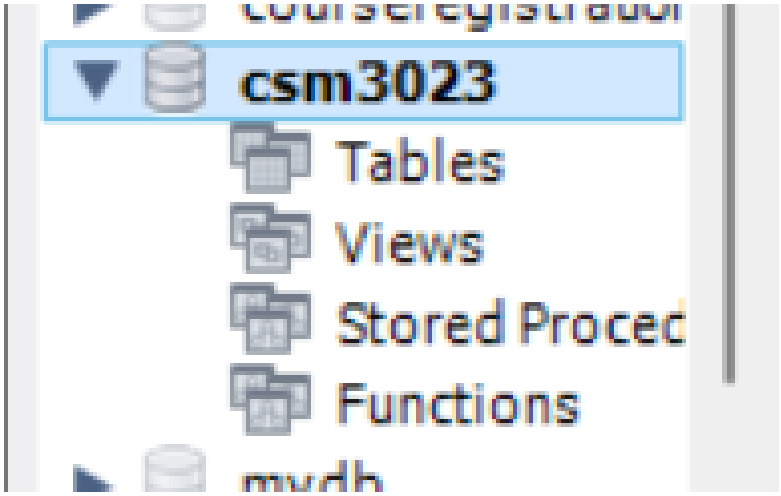


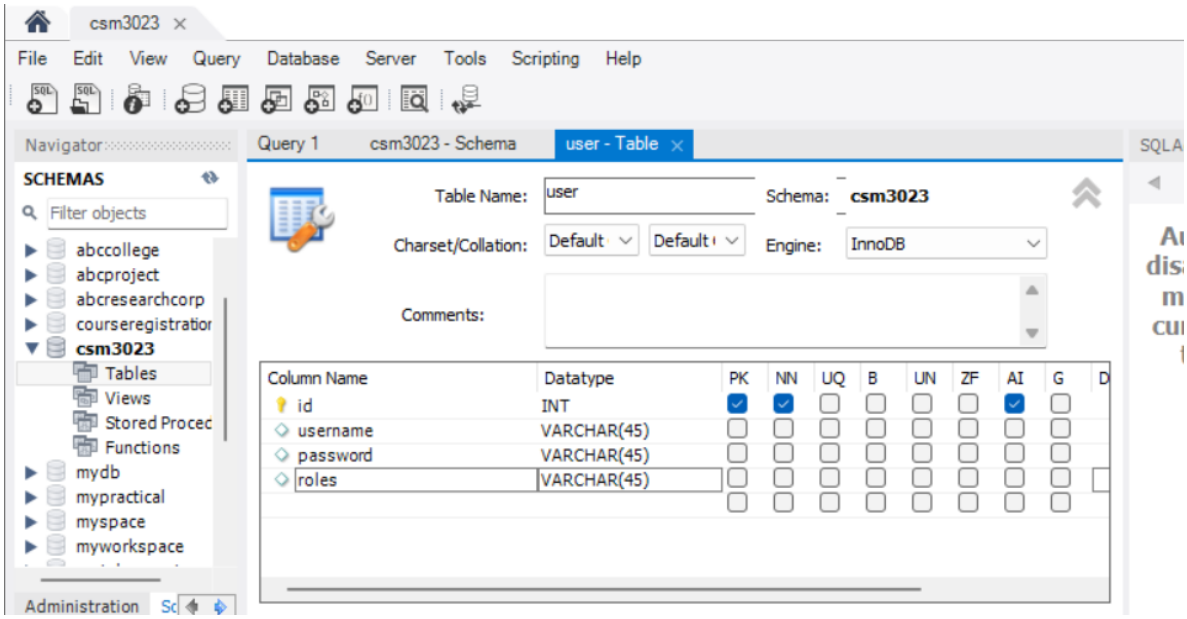
9. You will see the data of the users. Now, we have finished this task.

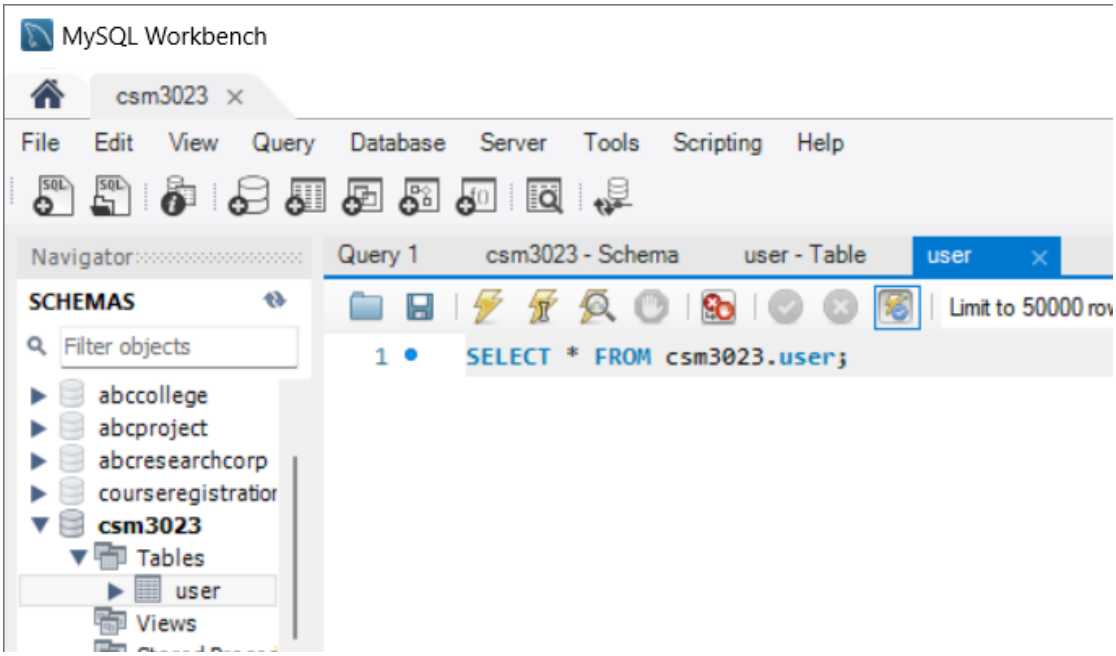
Note: In a real implementation, you should encrypt the password and never display the original form of it.

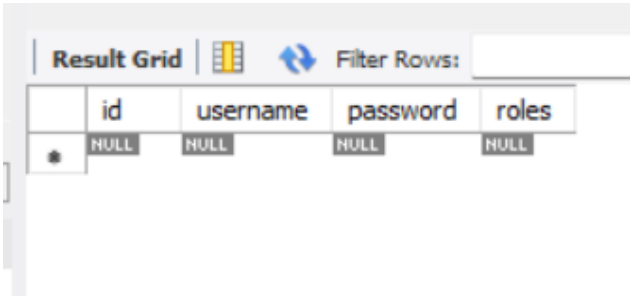


Output:


csm3023
Tables
Views
Stored Procec
Functions


csm3023
File Edit View Query Database Server Tools Scripting Help
Navigator
SCHEMAS
Filter objects
abccollege
abcproject
abcresearchcorp
courseregistratior
csm3023
Tables
Views
Stored Procec
Functions
mydb
mypractical
myspace
myworkspace
Administration
Query 1
csm3023 - Schema
user - Table
Table Name: user
Schema: csm3023
Charset/Collation: Default Default
Engine: InnoDB
Comments:
Column Name
Datatype
PK NN UQ B UN ZF AI G
id INT
username VARCHAR(45)
password VARCHAR(45)
roles VARCHAR(45)
SQLA

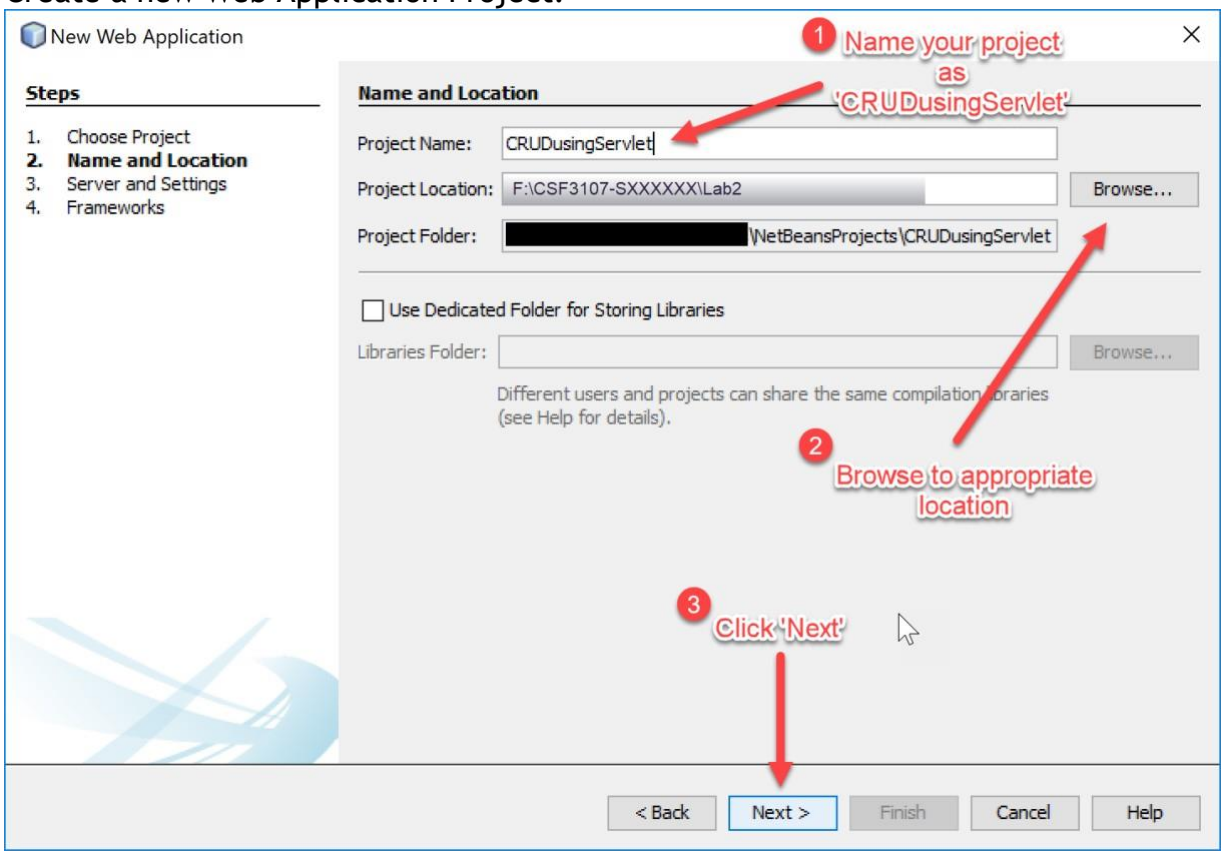

MySQL Workbench
csm3023
File Edit View Query Database Server Tools Scripting Help
Navigator
SCHEMAS
Filter objects
abccollege
abcproject
abcresearchcorp
courseregistration
csm3023
Tables
user
Views
Query 1
csm3023 - Schema
user - Table
user
Limit to 50000 rov
1
SELECT * FROM csm3023.user;


Result Grid
Filter Rows:
id
username
password
roles
NULL
NULL
NULL
NULL

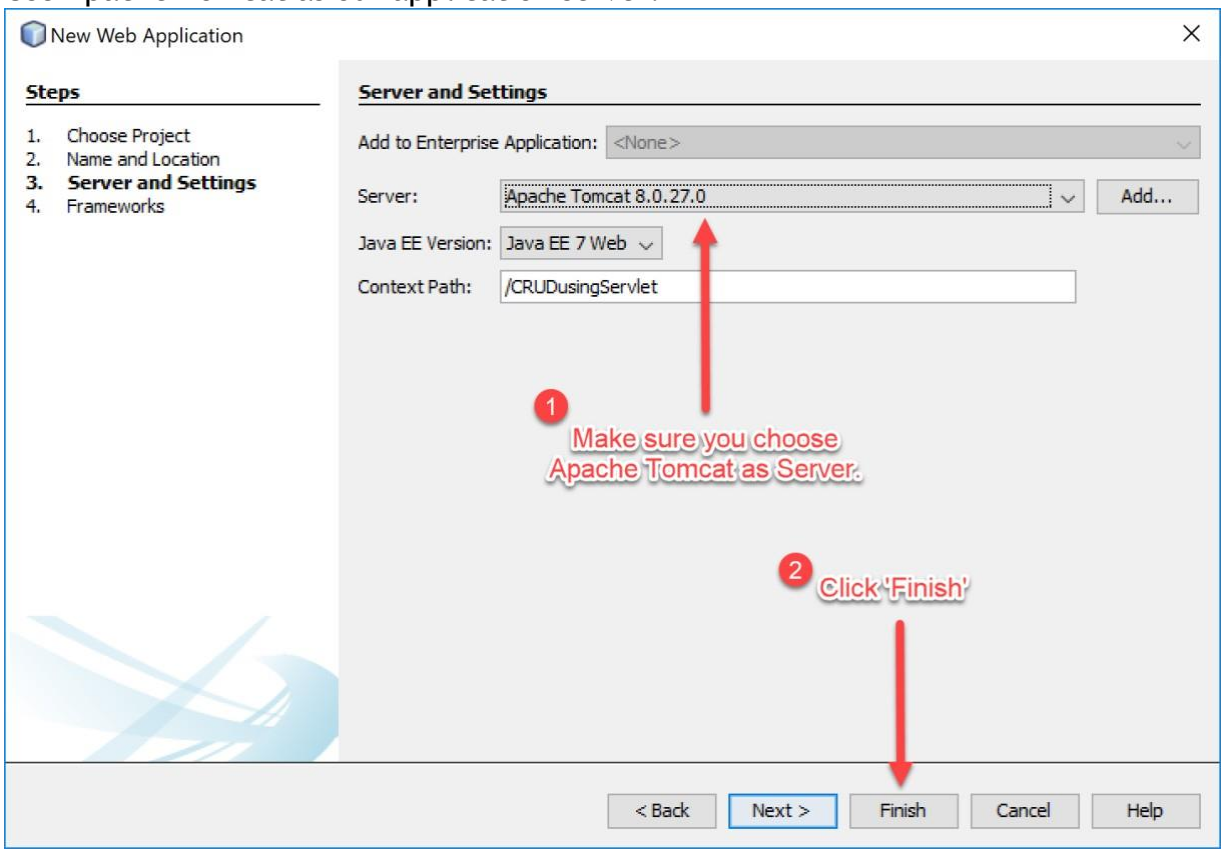
Task 3: Setting the Environment of Web Application for Database Connection

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

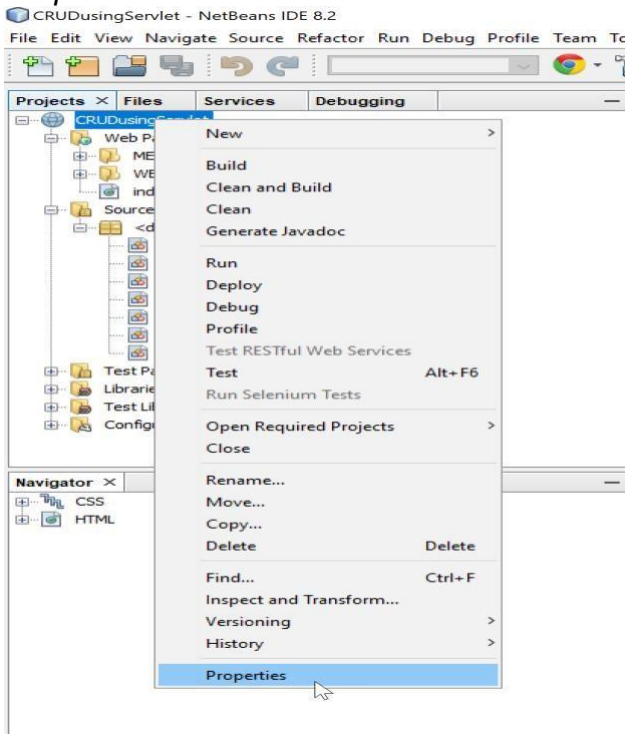
1. Create a new Web Application Project.



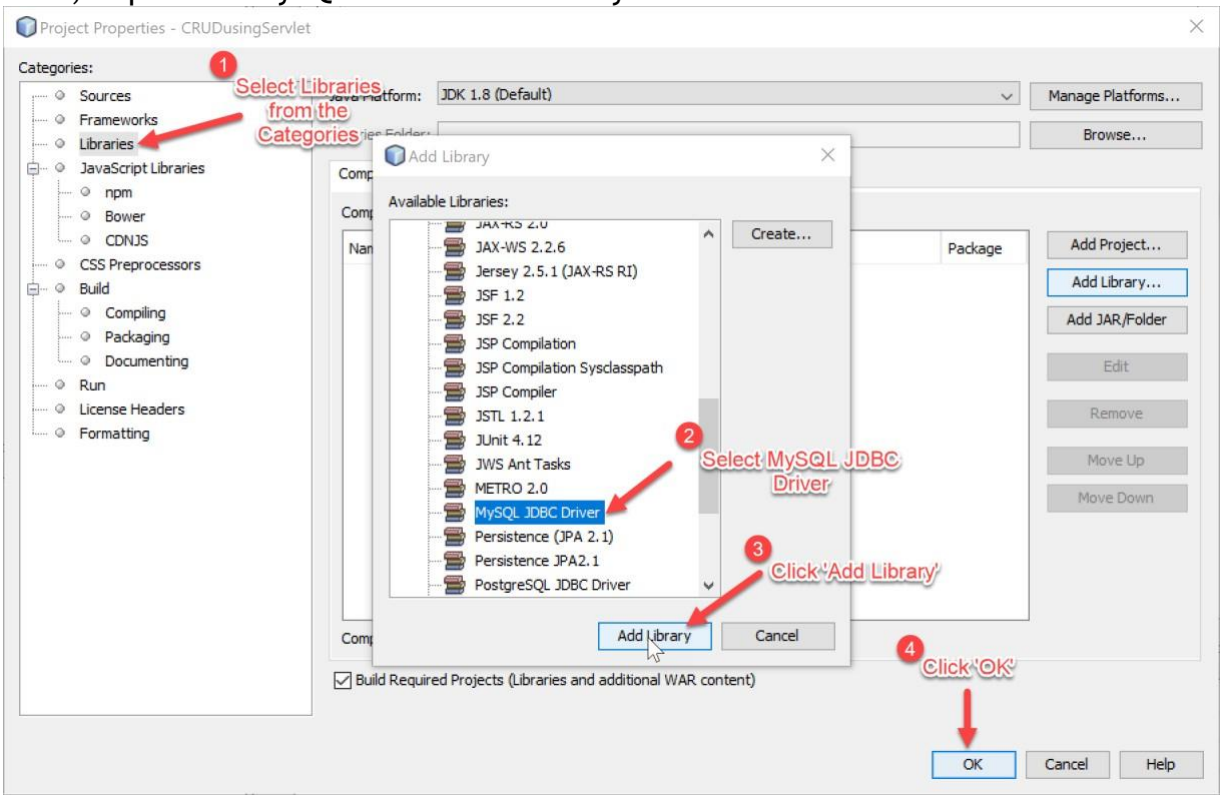
2. Use Apache Tomcat as our application server.



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

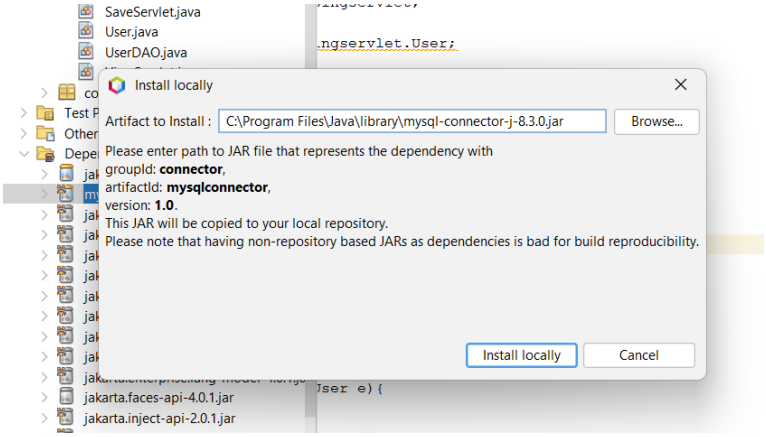


4. Next, import the MySQL JDBC Driver library.



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

   Output:

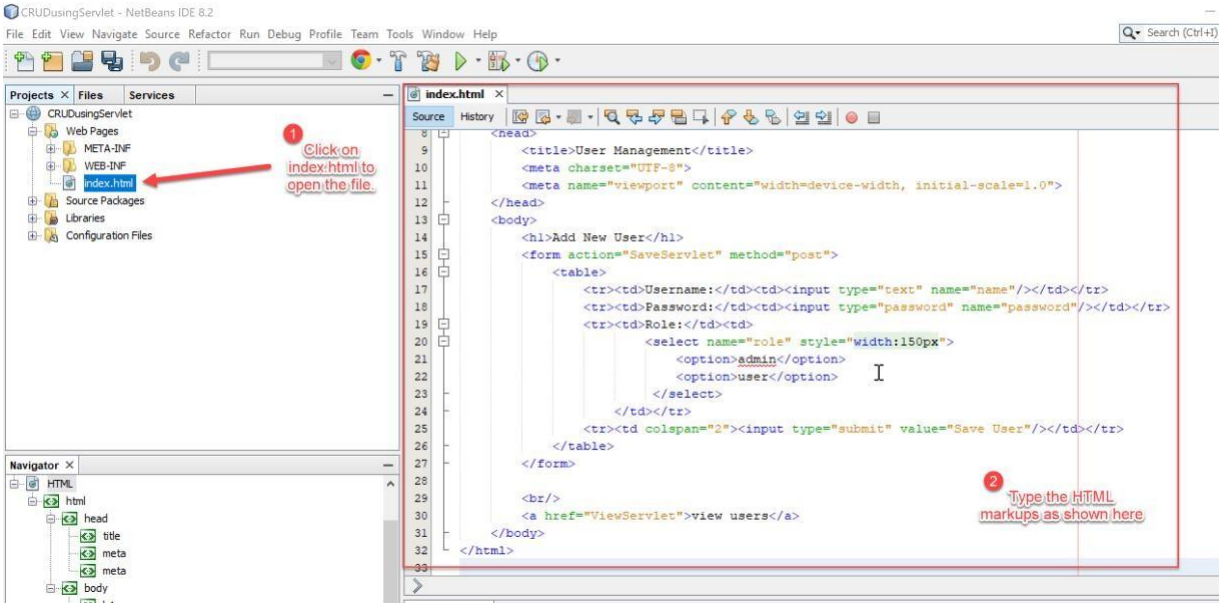


Task 4: Using Servlets for Database CRUDOperations

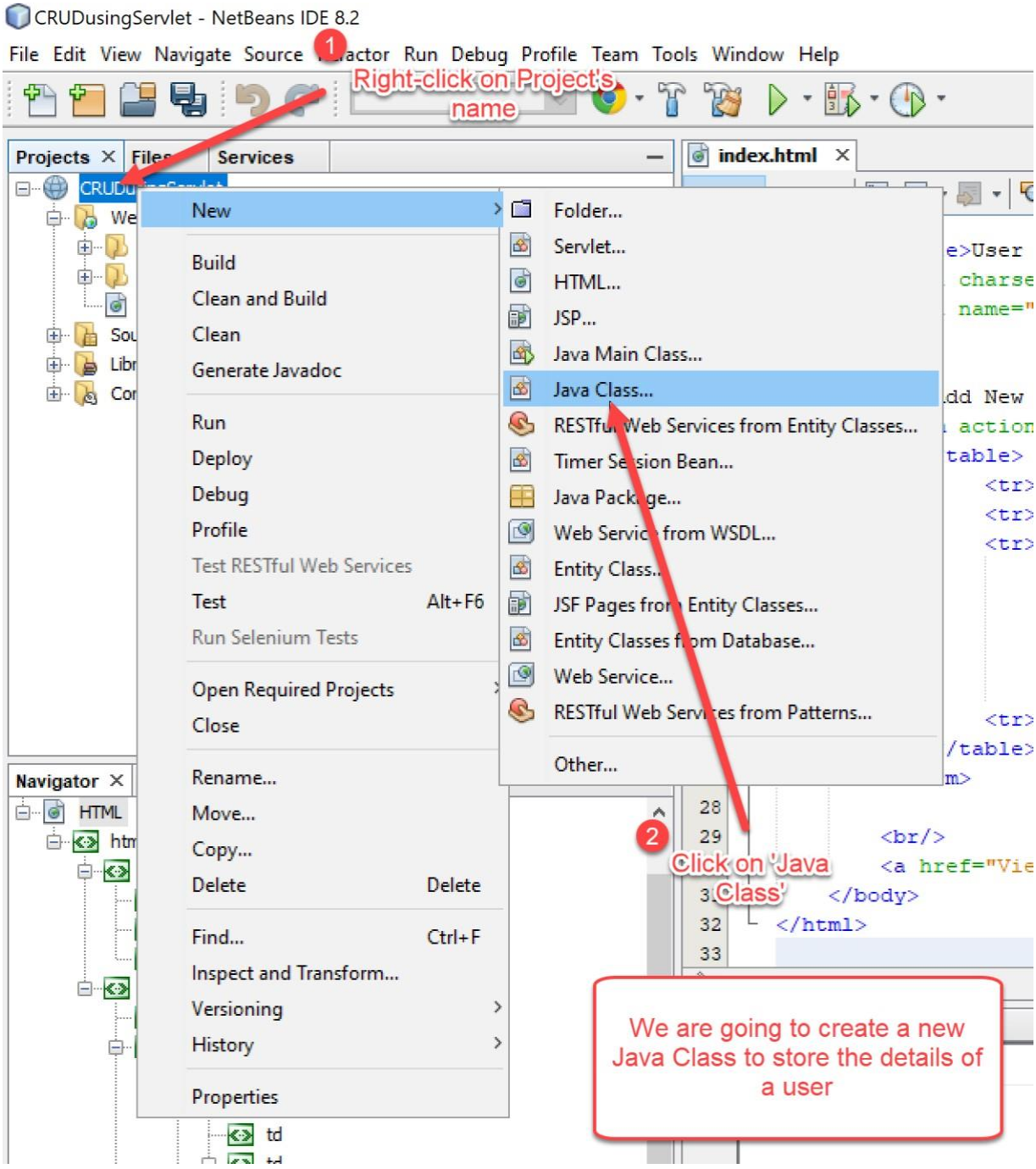
Objective:	To program multiple servlets for manipulating the database

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

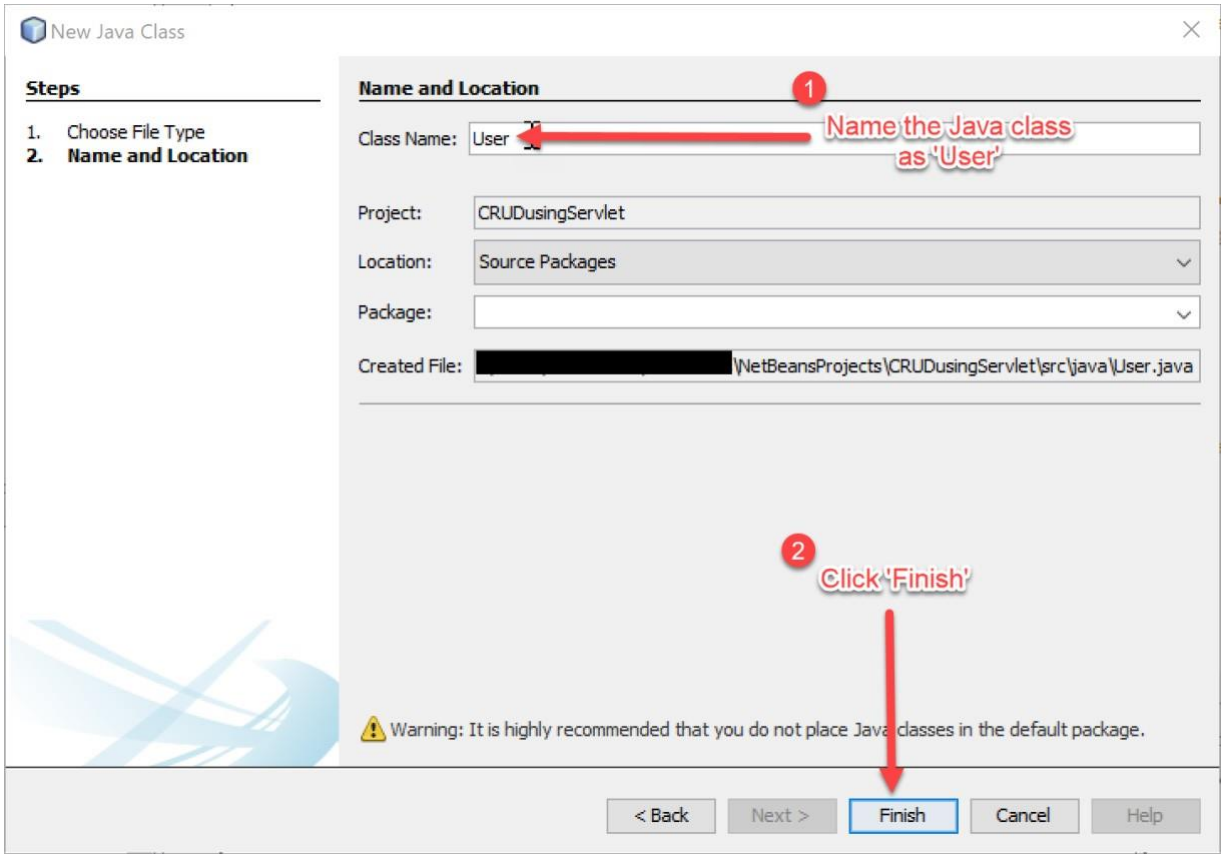
1. Open the Web Application Project *(CRUDusingServlet)*created in Task 3.
2. Open the index.html file and edit as follows.



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.

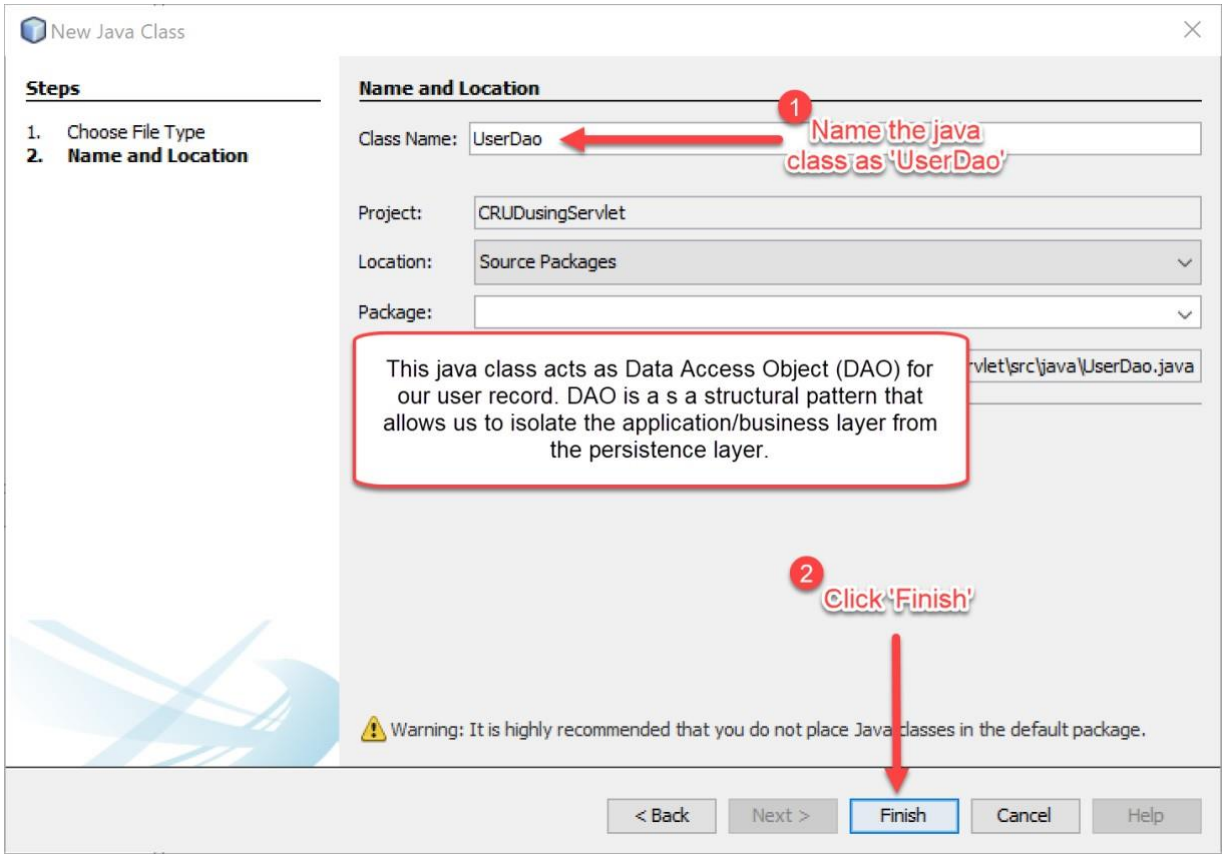
User.java - Editor

User.java

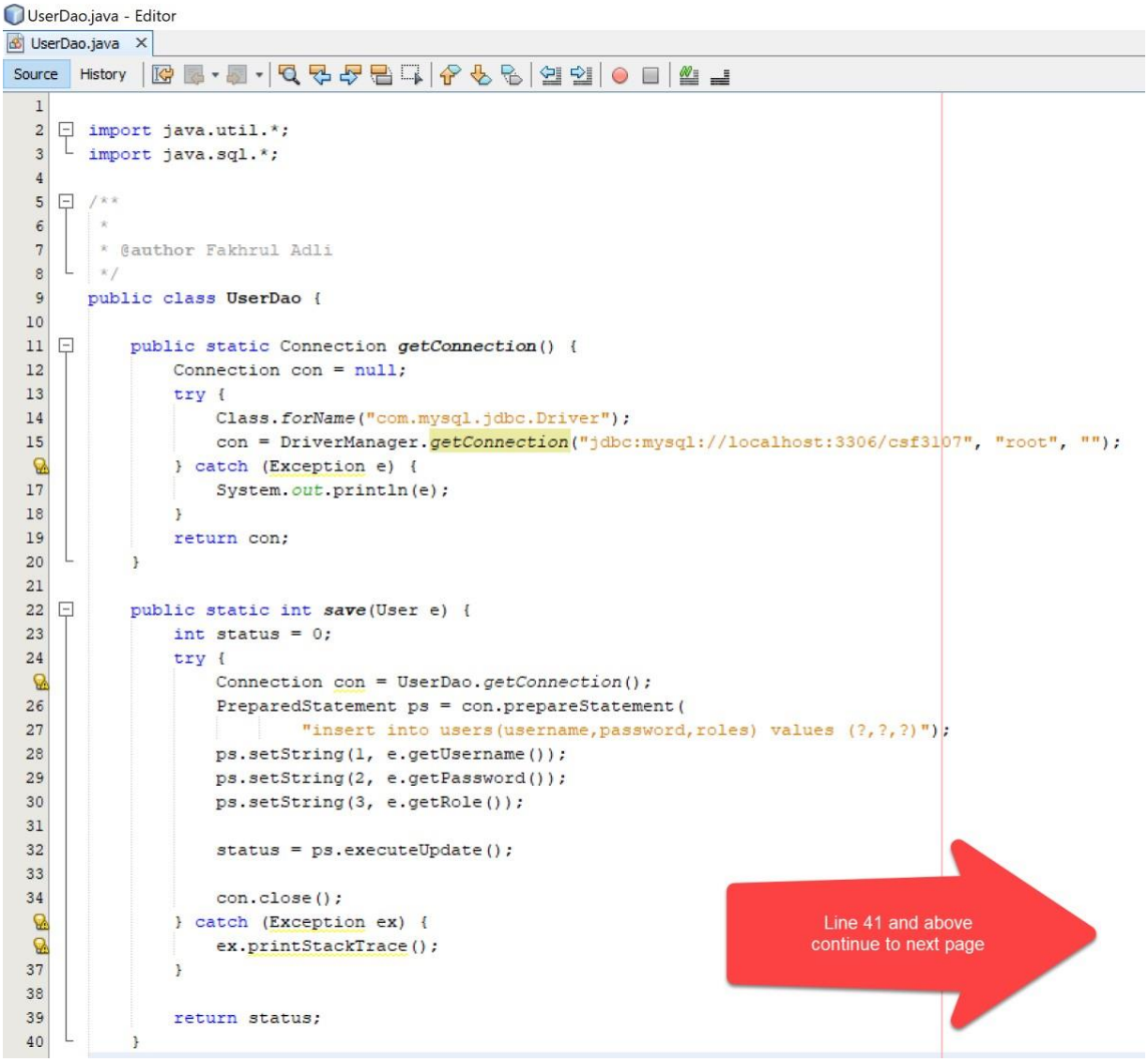
Source History

```
 */
public class User {

    private int id;
    private String username, password, role;

    public int getId() {
        return id;
    }

    public void setId(int id) {
        this.id = id;
    }

    public String getUsername() {
        return username;
    }

    public void setUsername(String username) {
        this.username = username;
    }

    public String getPassword() {
        return password;
    }

    public void setPassword(String password) {
        this.password = password;
    }

    public String getRole() {
        return role;
    }

    public void setRole(String role) {
        this.role = role;
    }

}
```

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

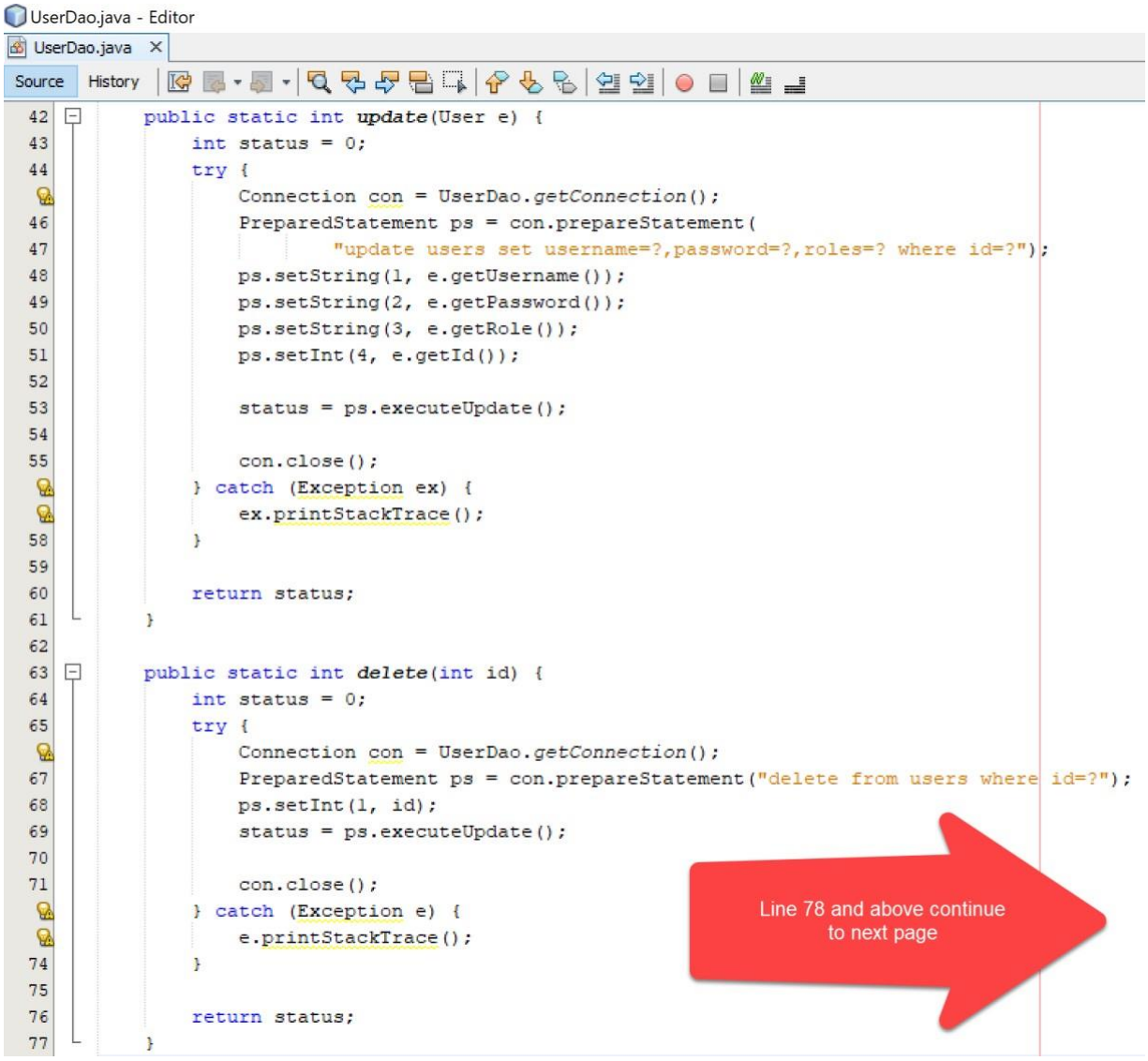


7. Put the following codes in UserDao.java file. Make sure your database URL is correct, and database user credentials are also valid.



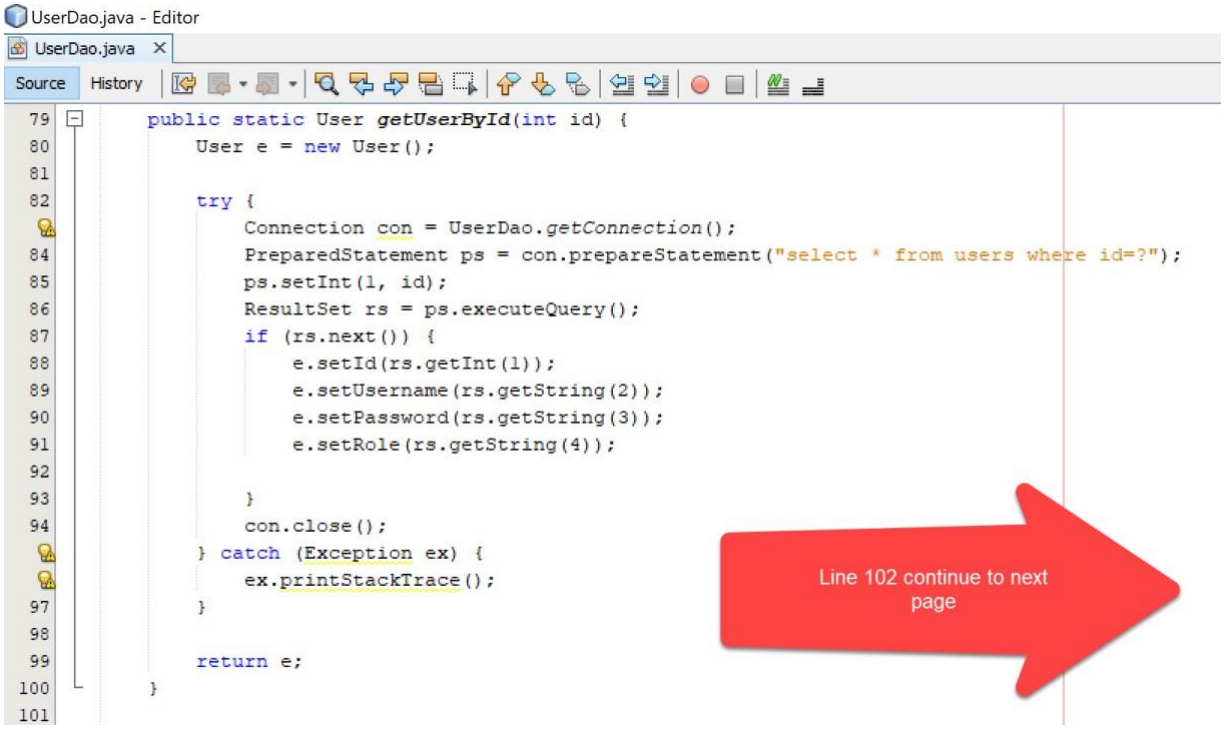

```
1
2   import java.util.*;
3   import java.sql.*;
4
5   /**
6    *
7    * @author Fakhrul Adli
8    */
9   public class UserDao {
10
11      public static Connection getConnection() {
12          Connection con = null;
13          try {
14              Class.forName("com.mysql.jdbc.Driver");
15  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

8. …c o n t i n u e d .




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

9. …continued.

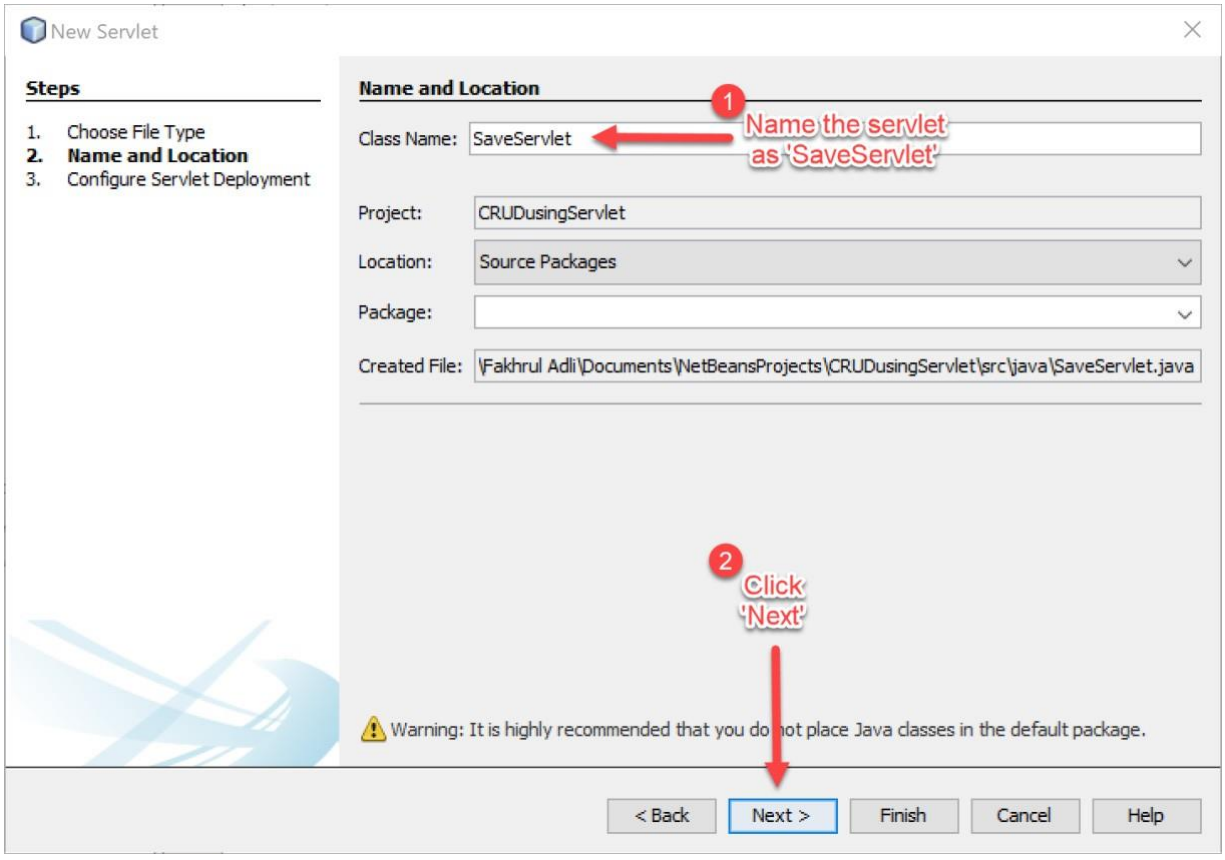



```
79      public static User getUserById(int id) {
80          User e = new User();
81
82          try {
84              Connection con = UserDao.getConnection();
85              PreparedStatement ps = con.prepareStatement("select * from users where id=?");
86              ps.setInt(1, id);
87              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;
103     }
```

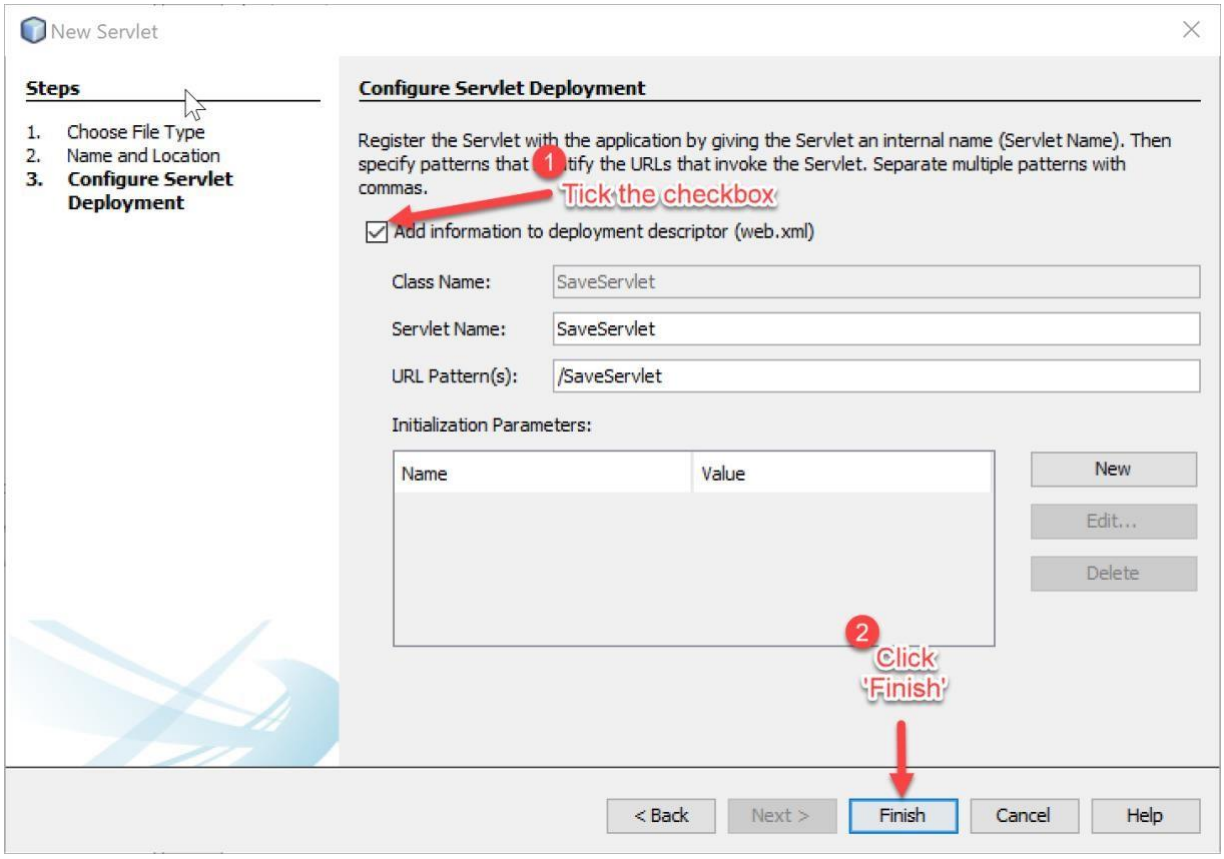
10. Save (CTRL+s) UserDao.java file.


```
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
126         return list;
127     }
128 }
```

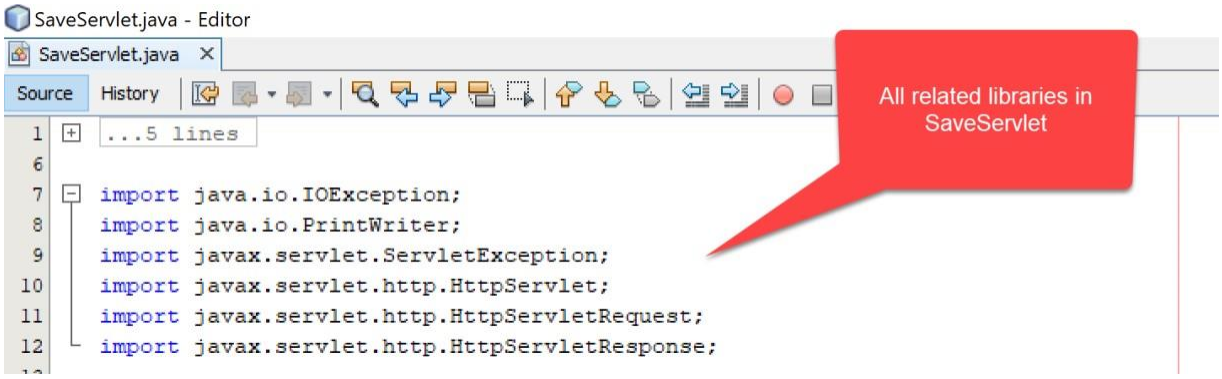
11. After finish coding the UserDao.java, now we are to code the servlet for saving data into the database.



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.

ViewServlet.java - Editor

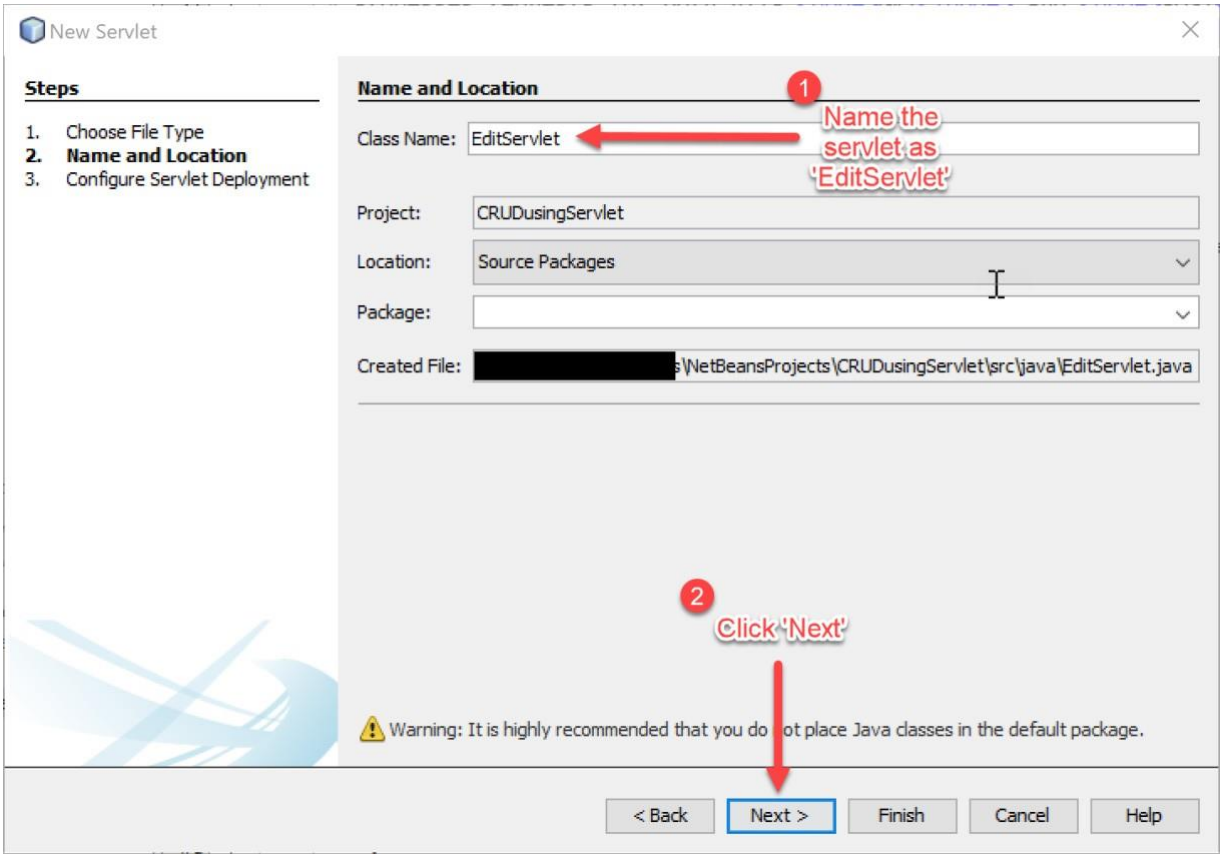
```
7  import java.io.IOException;
8  import java.io.PrintWriter;
9  import javax.servlet.ServletException;
10 import javax.servlet.http.HttpServlet;
11 import javax.servlet.http.HttpServletRequest;
12 import javax.servlet.http.HttpServletResponse;
13
14 import java.util.List;
```

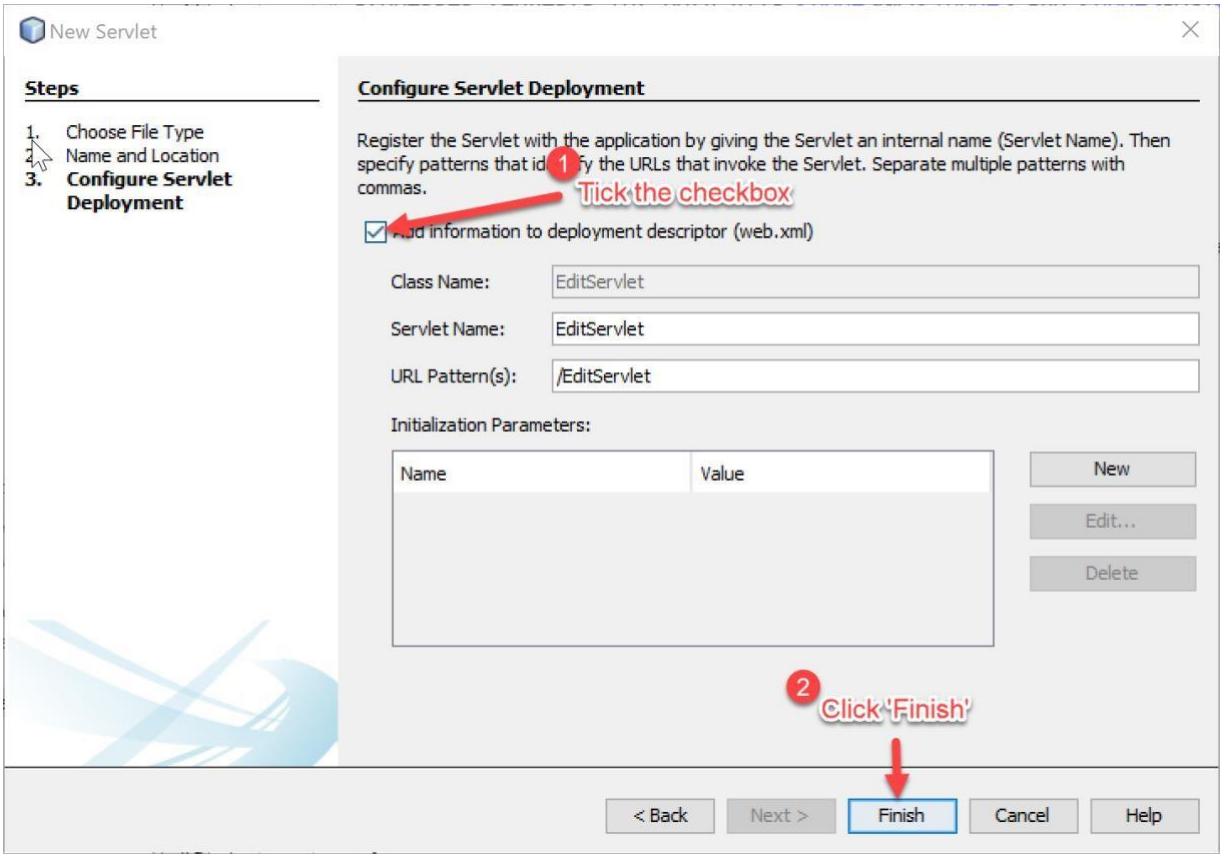
Import all related libraries

```
31     protected void processRequest(HttpServletRequest request, HttpServletResponse response)
32             throws ServletException, IOException {
33         response.setContentType("text/html");
           PrintWriter out=response.getWriter();
35         out.println("<a href='index.html'>Add New User</a>");
36         out.println("<h1>User List</h1>");
37
38         List<User> list=UserDao.getAllUsers();
39
40         out.print("<table border='1' width='100%'");
41         out.print("<tr><th>Id</th><th>Name</th><th>Password</th><th>Role</th>"
42                 + "<th>Edit</th><th>Delete</th></tr>");
           for(User e:list){
44          out.print("<tr><td>"+e.getId()+"</td><td>"+e.getUsername()+"</td><td>"
45                  +e.getPassword()+"</td><td>"+e.getRole()+"</td><td><a href='EditServlet?id="
46                  +e.getId()+"'>edit</a></td> <td><a href='DeleteServlet?id="
47                  +e.getId()+"'>delete</a></td></tr>");
48         }
49         out.print("</table>");
50
51         out.close();
52     }
```

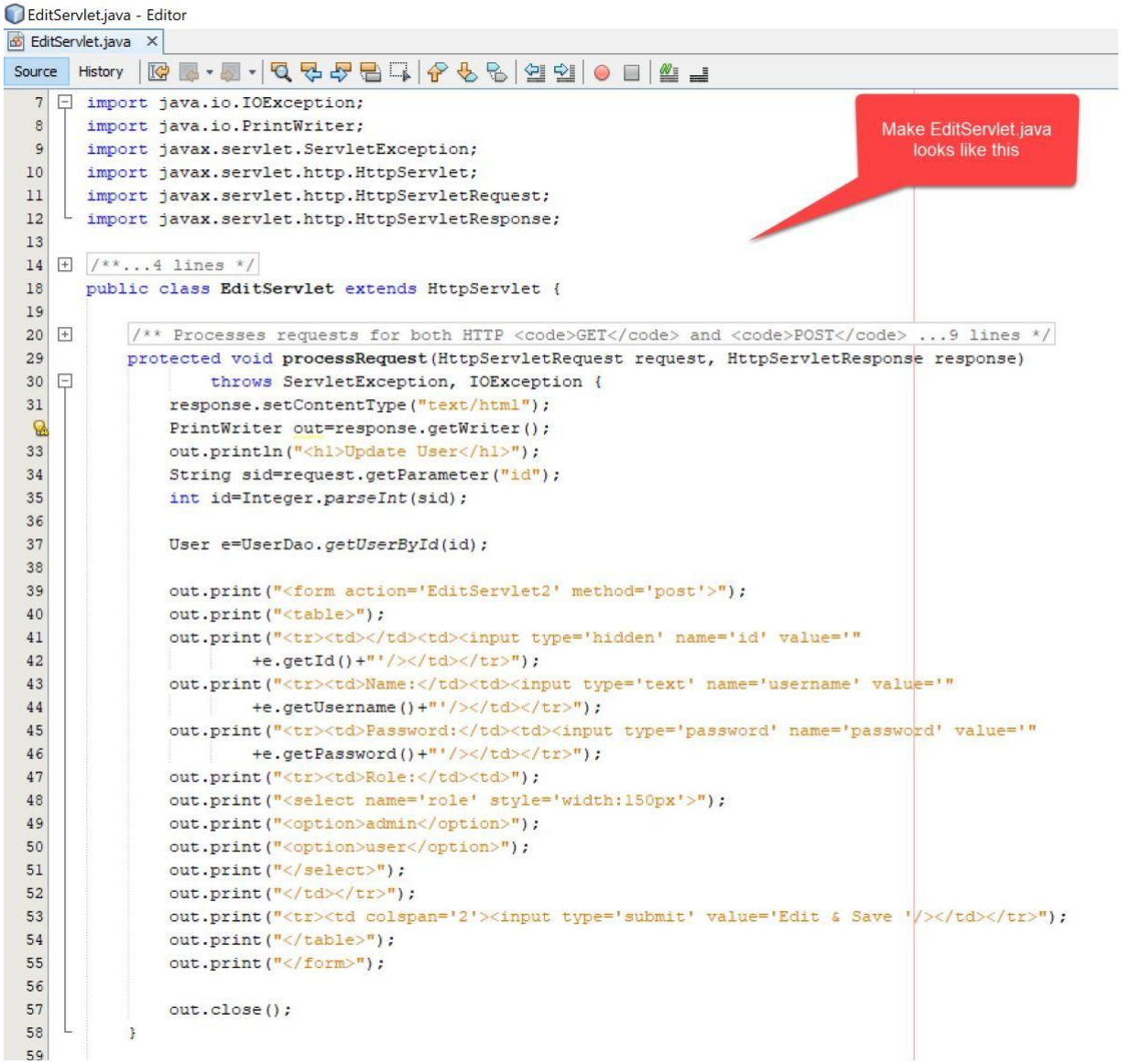
Put these codes in processRequest method

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.



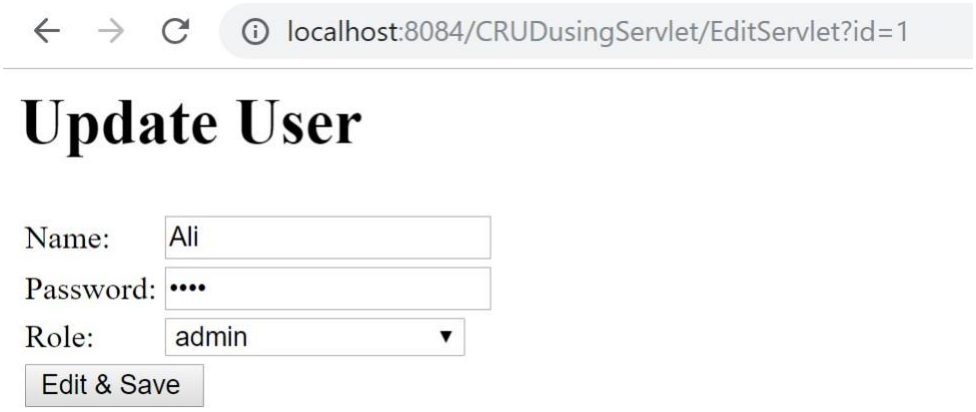


18. Type the following codes in the servlet you just created.

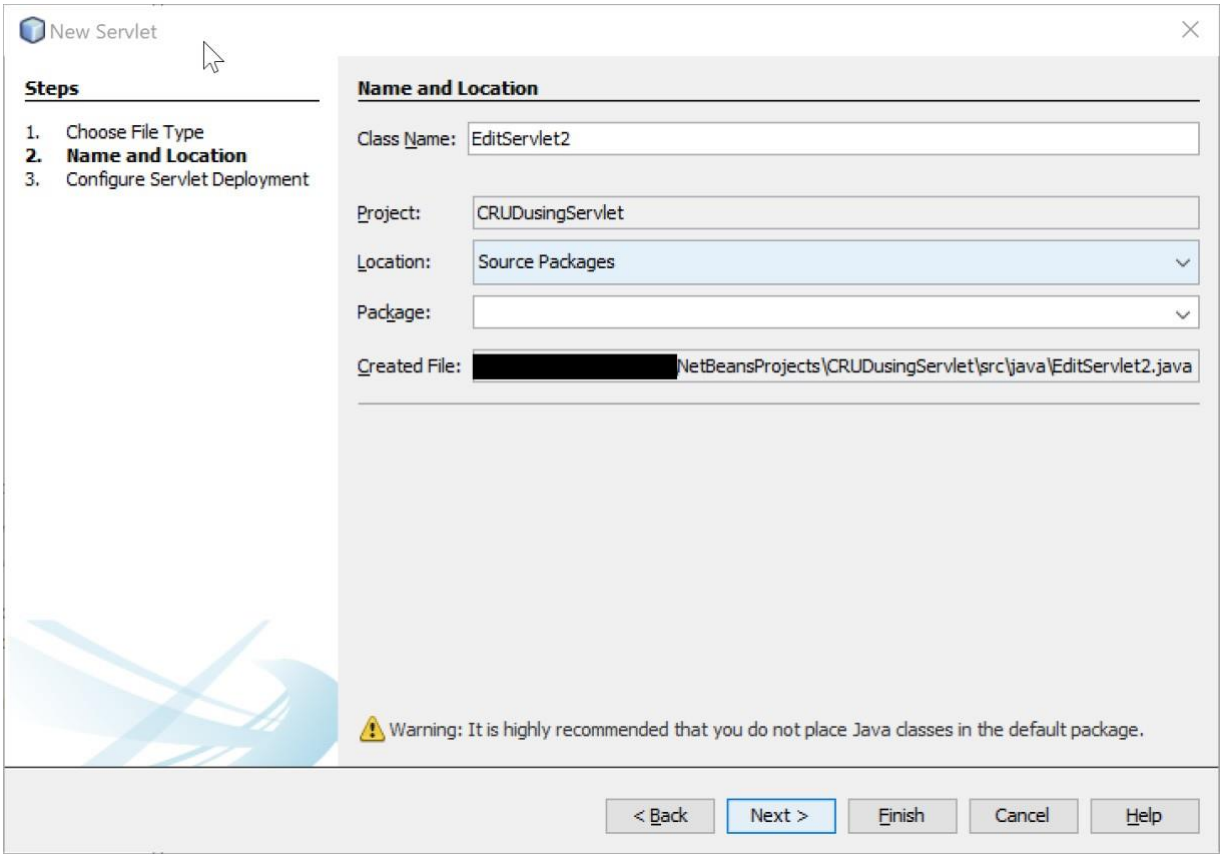



```
7  import java.io.IOException;
8  import java.io.PrintWriter;
9  import javax.servlet.ServletException;
10 import javax.servlet.http.HttpServlet;
11 import javax.servlet.http.HttpServletRequest;
12 import javax.servlet.http.HttpServletResponse;
13
14 /**...4 lines */
18 public class EditServlet extends 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");
           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:

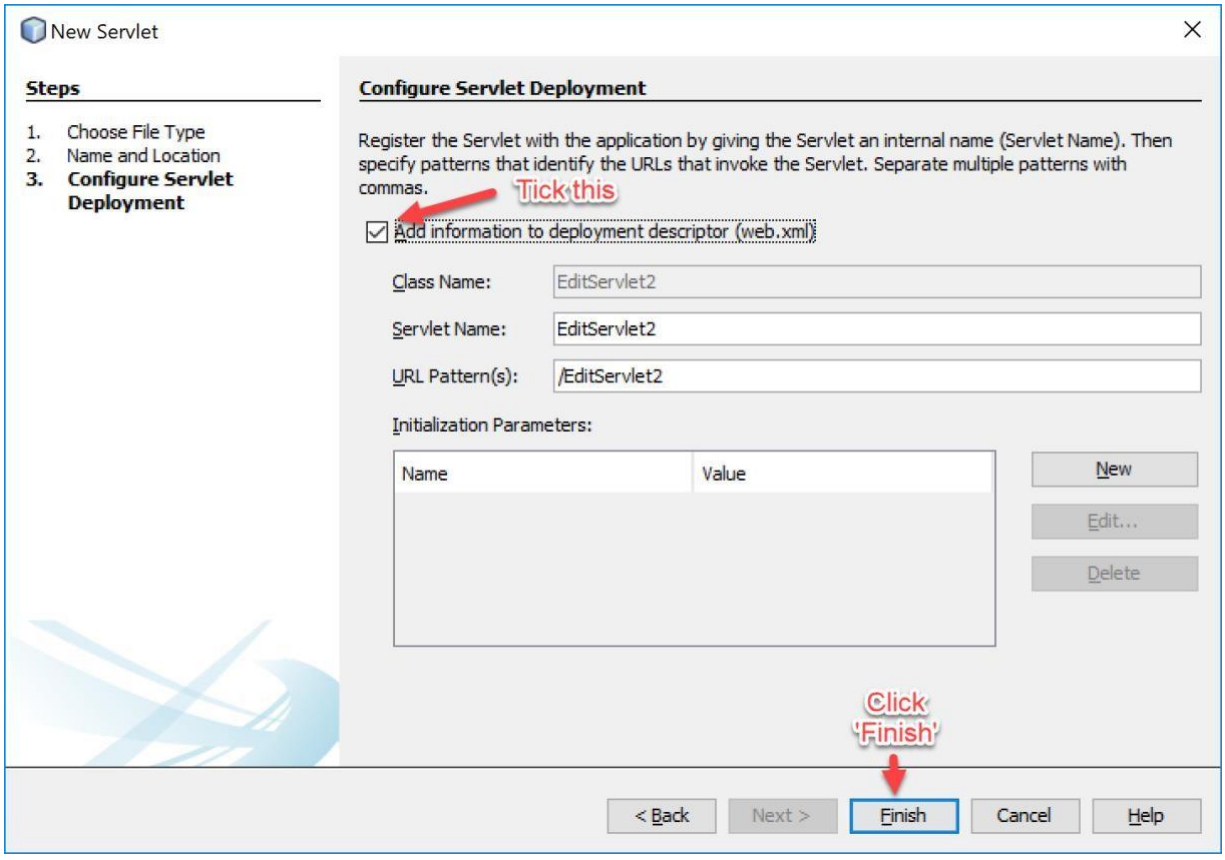


20. Now, we need another servlet to handle database processing.



21. Remember to tick 'Add information to deplo

Netbeans will configure the servlet in web.xml on your behalf.

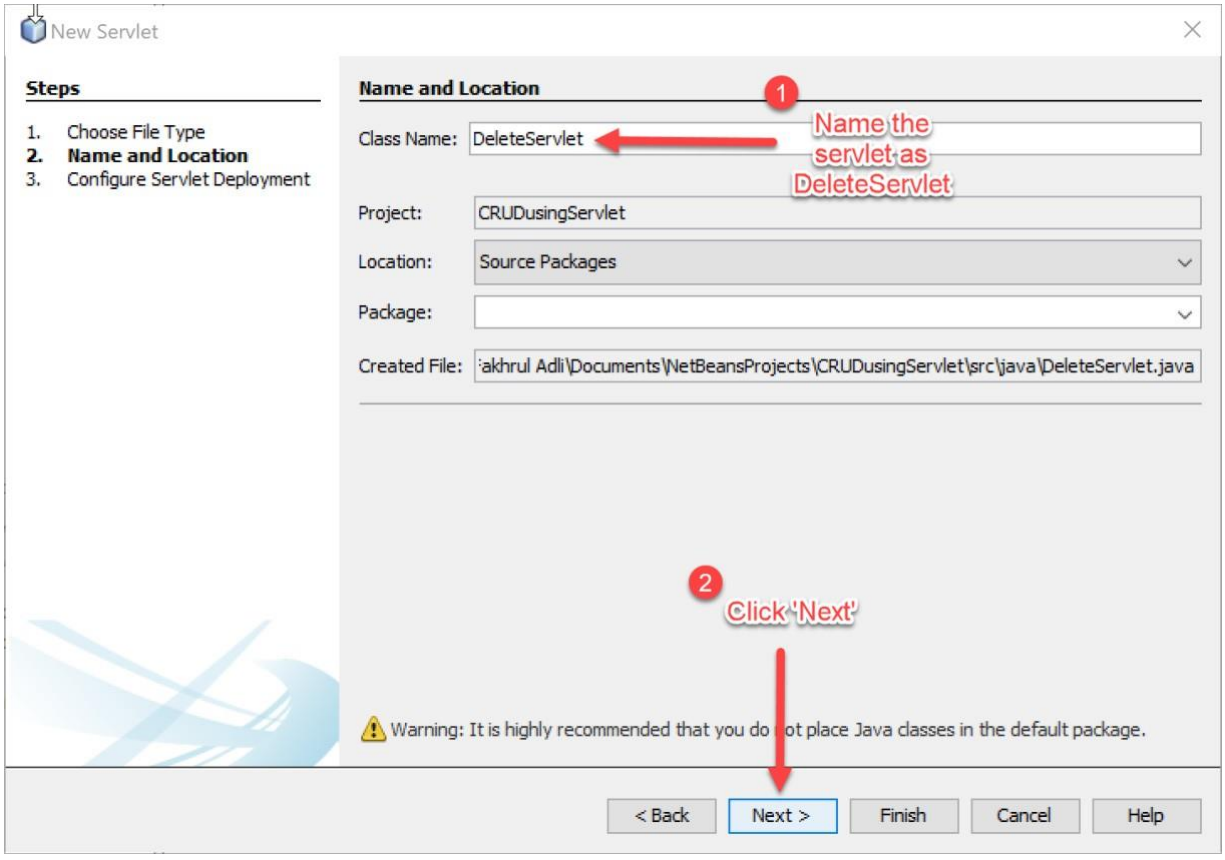


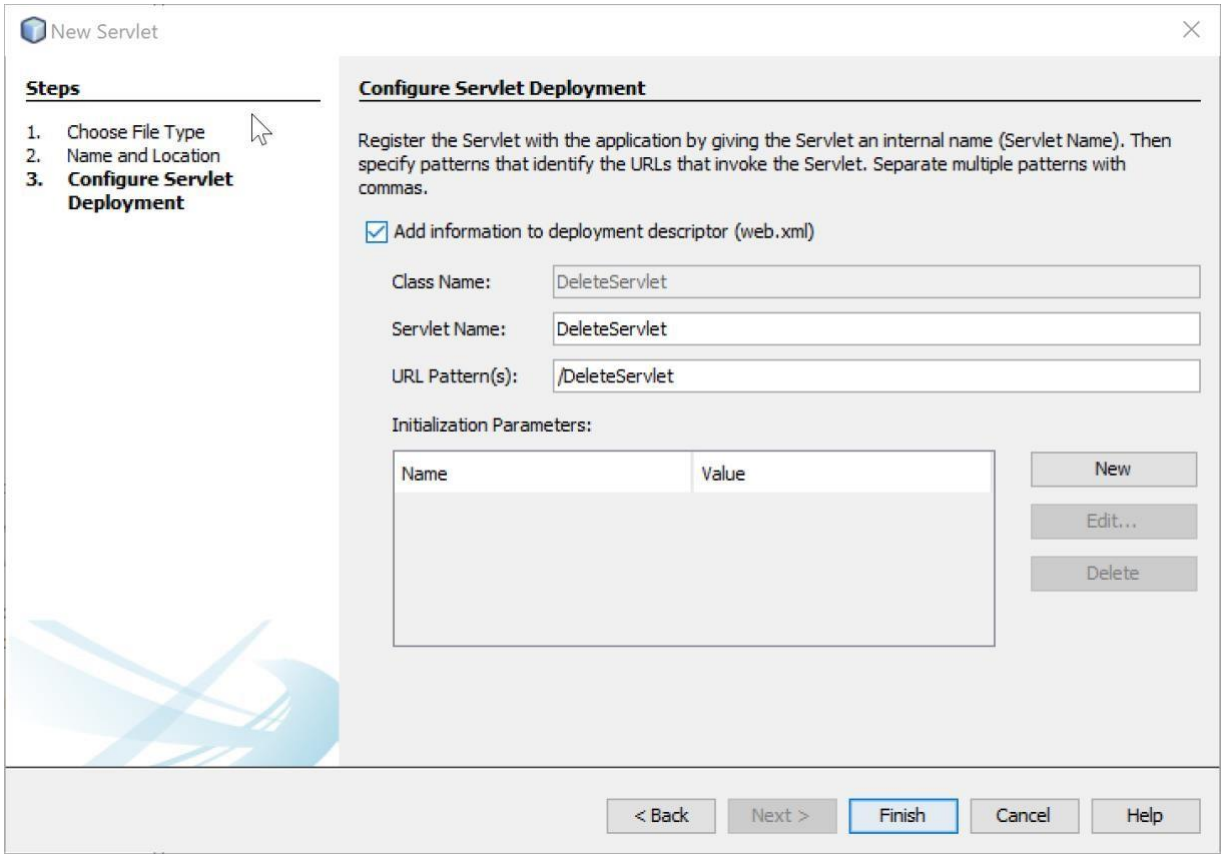
22. Type the following codes in the processRequest() method in EditServlet2.java. Save the file when you have finished.

```
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      }
```

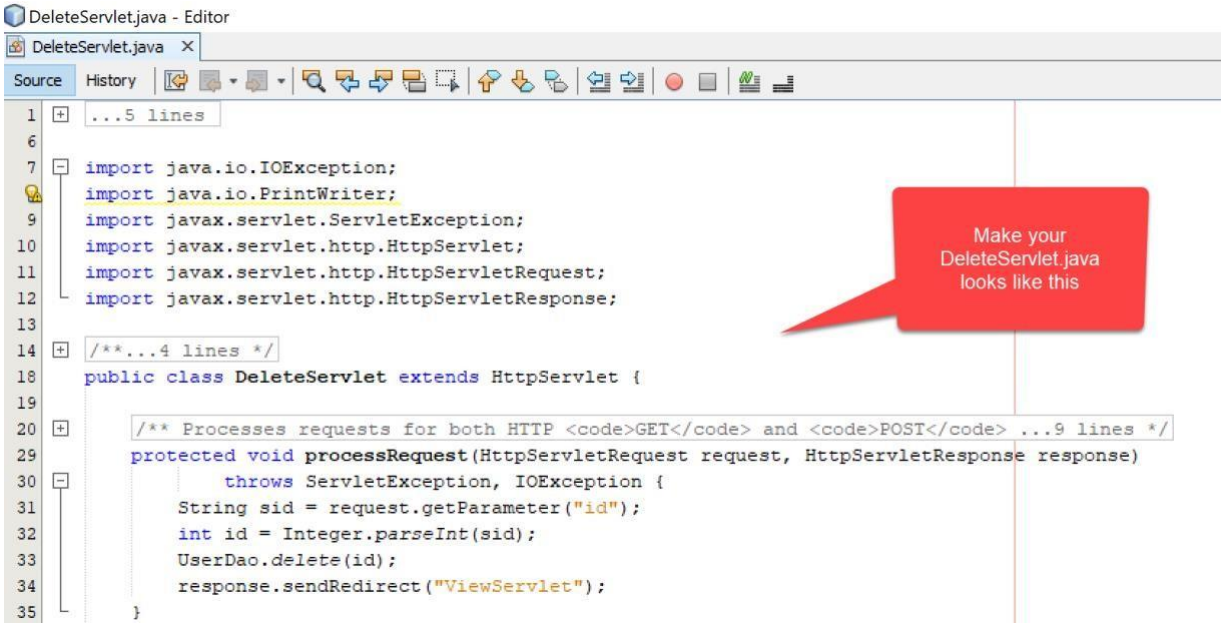
Put these codes in the processRequest() method

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.



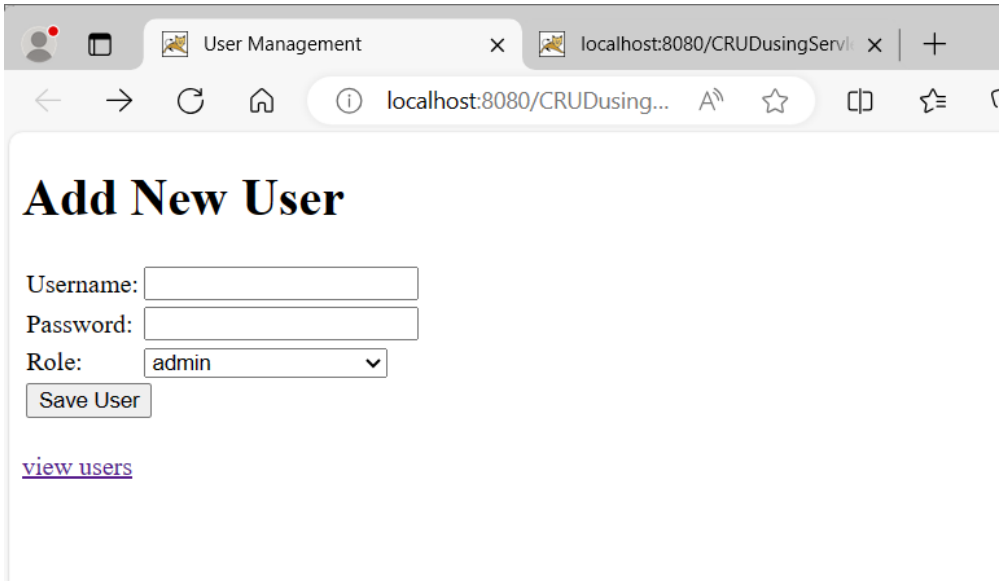


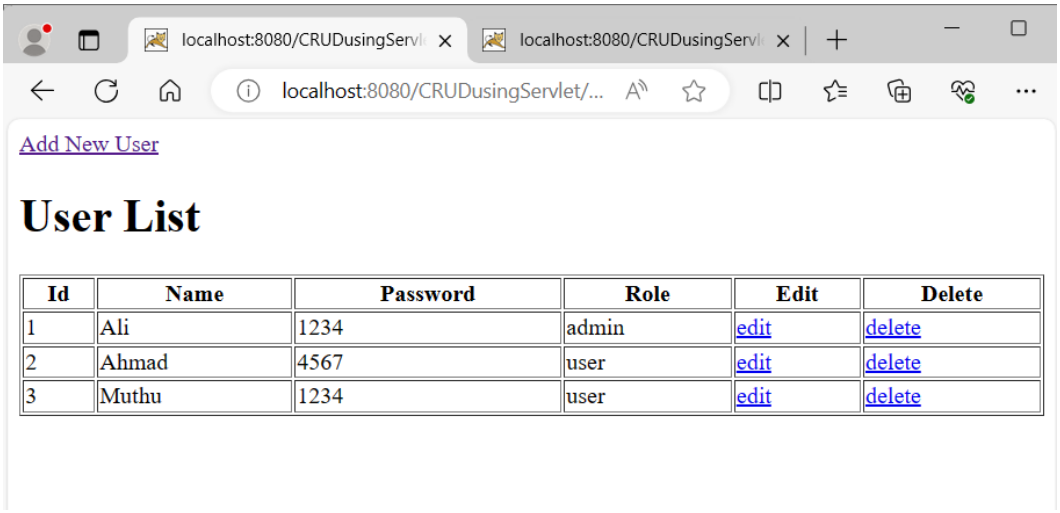
24. Write the codes below into your DeleteServlet.java file. Save the file when you have done.



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.

Output:





Reflections:

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

   Answer:

   MySQL connector

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.
   Answer:

   It was located in WEB-INF folder.

   Content:

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app version="6.0" xmlns="https://jakarta.ee/xml/ns/jakartaee"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="https://jakarta.ee/xml/ns/jakartaee
https://jakarta.ee/xml/ns/jakartaee/web-app_6_0.xsd">
    <servlet>
        <servlet-name>SaveServlet</servlet-name>
        <servlet-class>com.mycompany.crudusingservlet.SaveServlet</servlet-class>
    </servlet>
    <servlet>
        <servlet-name>ViewServlet</servlet-name>
        <servlet-class>com.mycompany.crudusingservlet.ViewServlet</servlet-class>
    </servlet>
    <servlet>
        <servlet-name>EditServlet</servlet-name>
        <servlet-class>com.mycompany.crudusingservlet.EditServlet</servlet-class>
    </servlet>
```

```
    <servlet>
        <servlet-name>EditServlet2</servlet-name>
        <servlet-class>com.mycompany.crudusingservlet.EditServlet2</servlet-class>
    </servlet>
    <servlet>
        <servlet-name>DeleteServlet</servlet-name>
        <servlet-class>com.mycompany.crudusingservlet.DeleteServlet</servlet-class>
    </servlet>
    <servlet-mapping>
        <servlet-name>SaveServlet</servlet-name>
        <url-pattern>/SaveServlet</url-pattern>
    </servlet-mapping>
    <servlet-mapping>
        <servlet-name>ViewServlet</servlet-name>
        <url-pattern>/ViewServlet</url-pattern>
    </servlet-mapping>
    <servlet-mapping>
        <servlet-name>EditServlet</servlet-name>
        <url-pattern>/EditServlet</url-pattern>
    </servlet-mapping>
    <servlet-mapping>
        <servlet-name>EditServlet2</servlet-name>
        <url-pattern>/EditServlet2</url-pattern>
    </servlet-mapping>
    <servlet-mapping>
        <servlet-name>DeleteServlet</servlet-name>
        <url-pattern>/DeleteServlet</url-pattern>
    </servlet-mapping>
    <session-config>
        <session-timeout>
            30
        </session-timeout>
    </session-config>
</web-app>
```

In the web.xml file, the <servlet-name> tag is used to specify the name of the servlet, while the <servlet-class> tag is used to specify the fully qualified class name od the servlet. The <servlet-mapping> tag is used to map a servlet to map a servlet 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?

   Answer:

   DAO is a way to separate the code that interacts with a database from the rest of the application. It helps in organizing and managing database operations, like CRUD method.