

Lesson 03 Demo 03

Servlet Configuration

Objective: To demonstrate the configuration of Servlets both individually and collectively, including the setup of init and context parameters.

Tools Required: Eclipse IDE

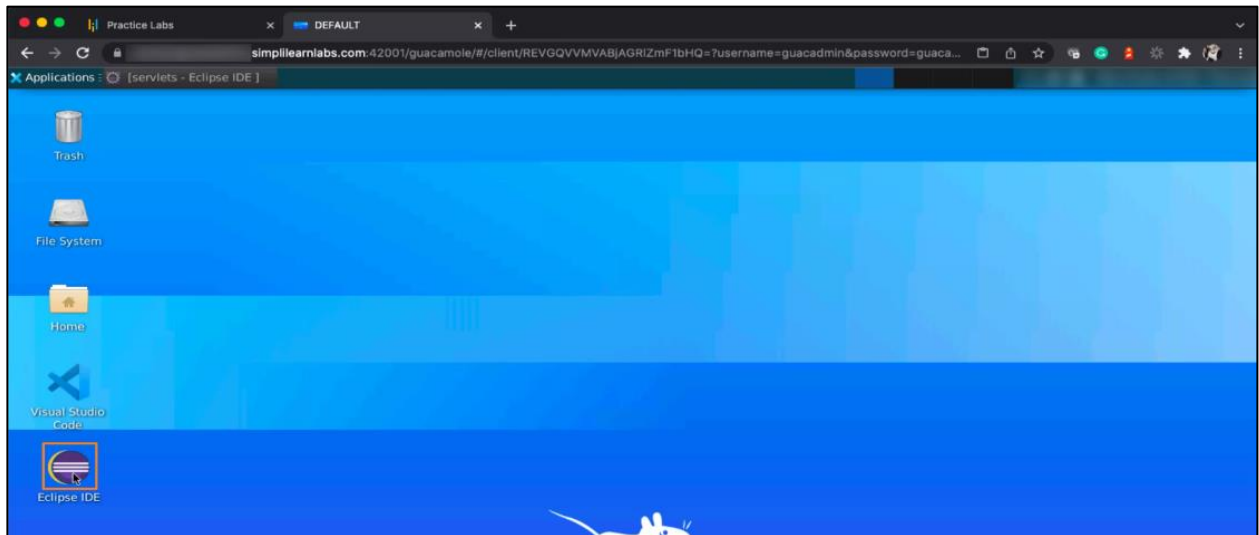
Prerequisites: None

Steps to be followed:

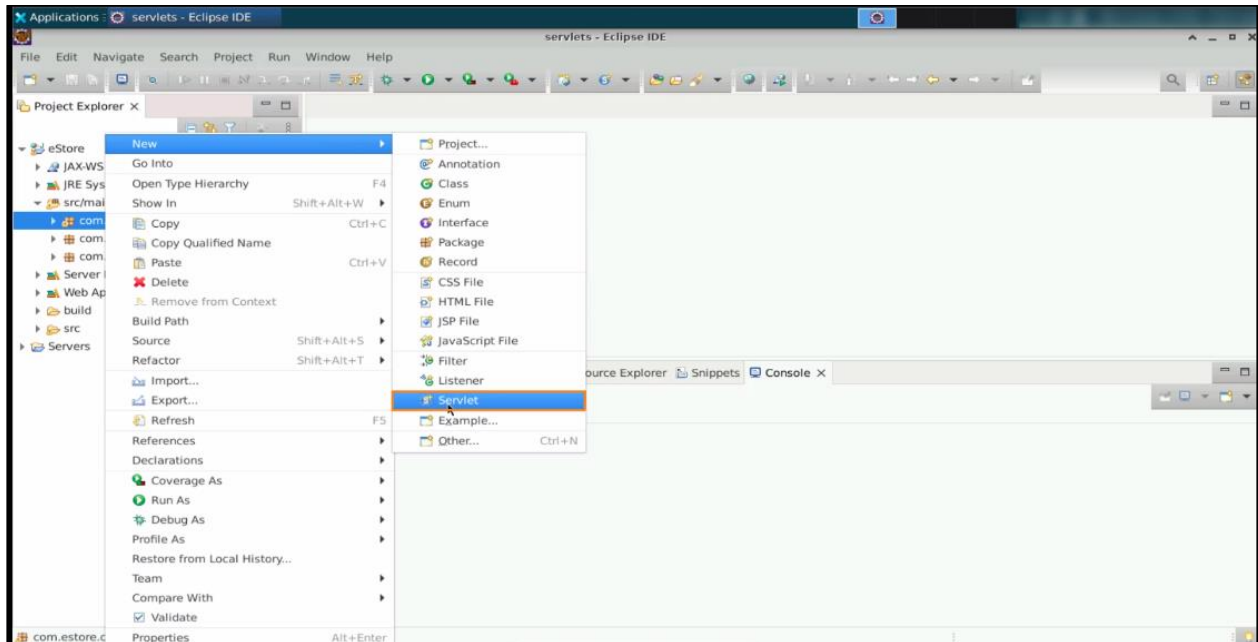
1. Create a new servlet
2. Create a hyperlink and run an application to check the output
3. Create a `getServerConfig` method
4. Create and extract parameters
5. Create multiple parameters and methods

Step 1: Create a new servlet

1.1 Open Eclipse IDE

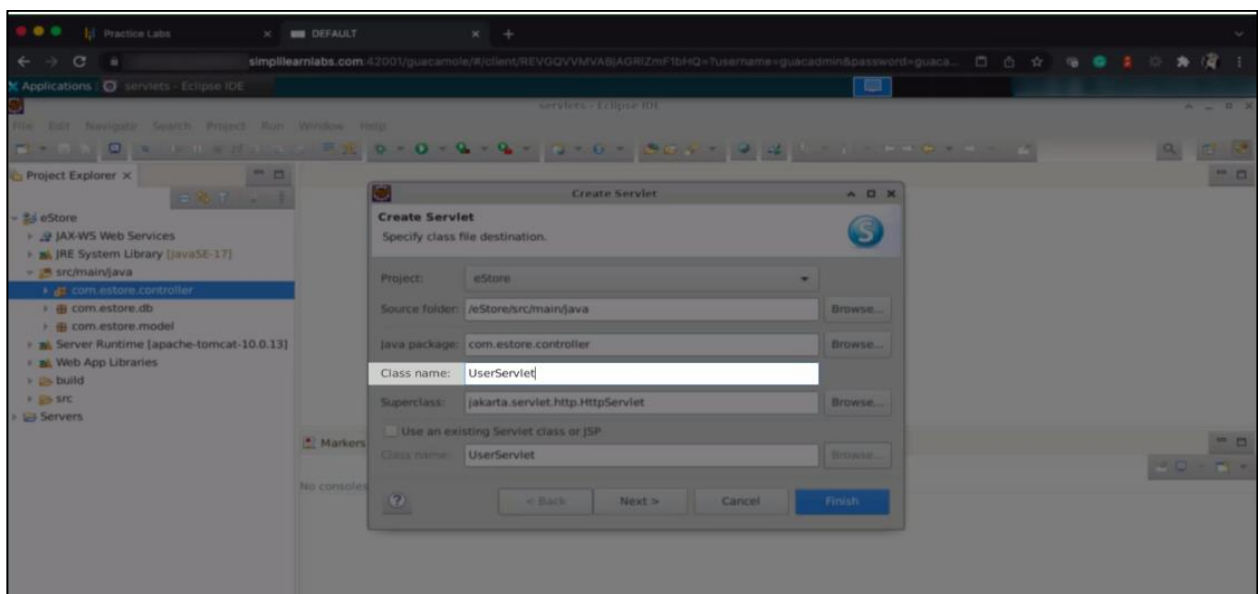


1.2 Go to the **estore** project. Go to **src/main** and then **com.estore.controller**. Right-click on it, select **New**, and click on **servlet**

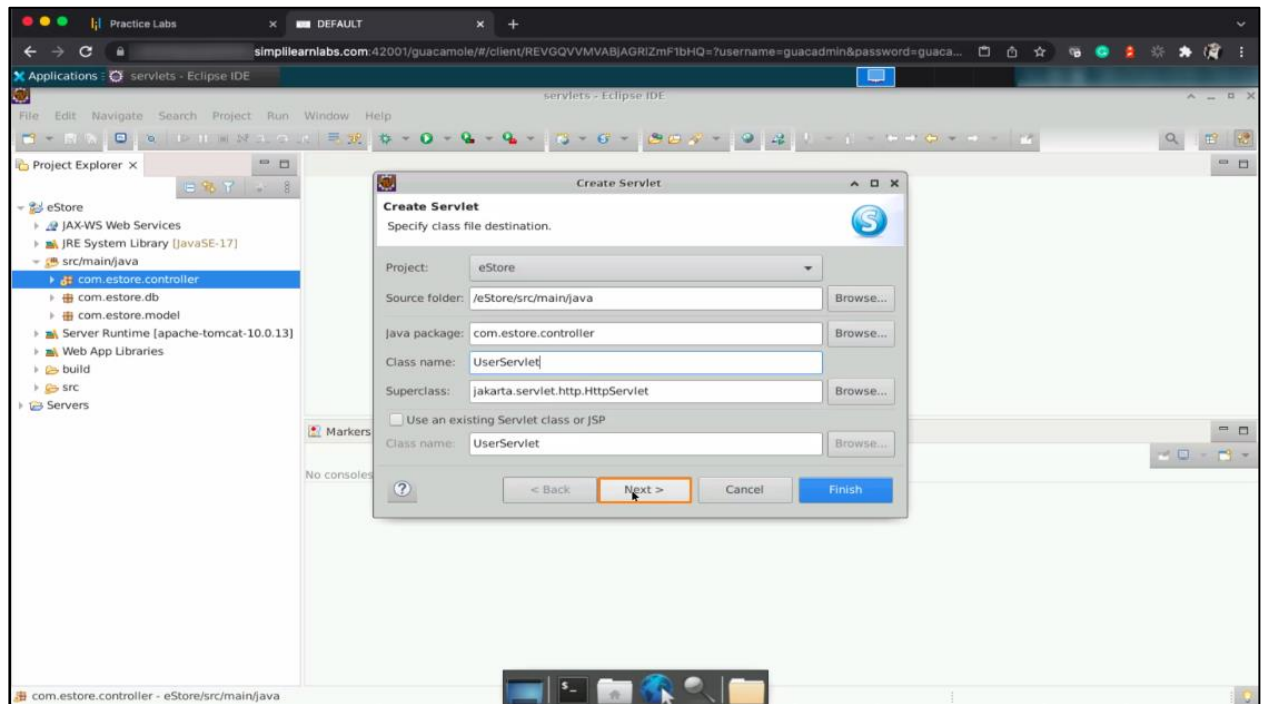


Note: Please refer to the previous demo on how to create the **eStore** project.

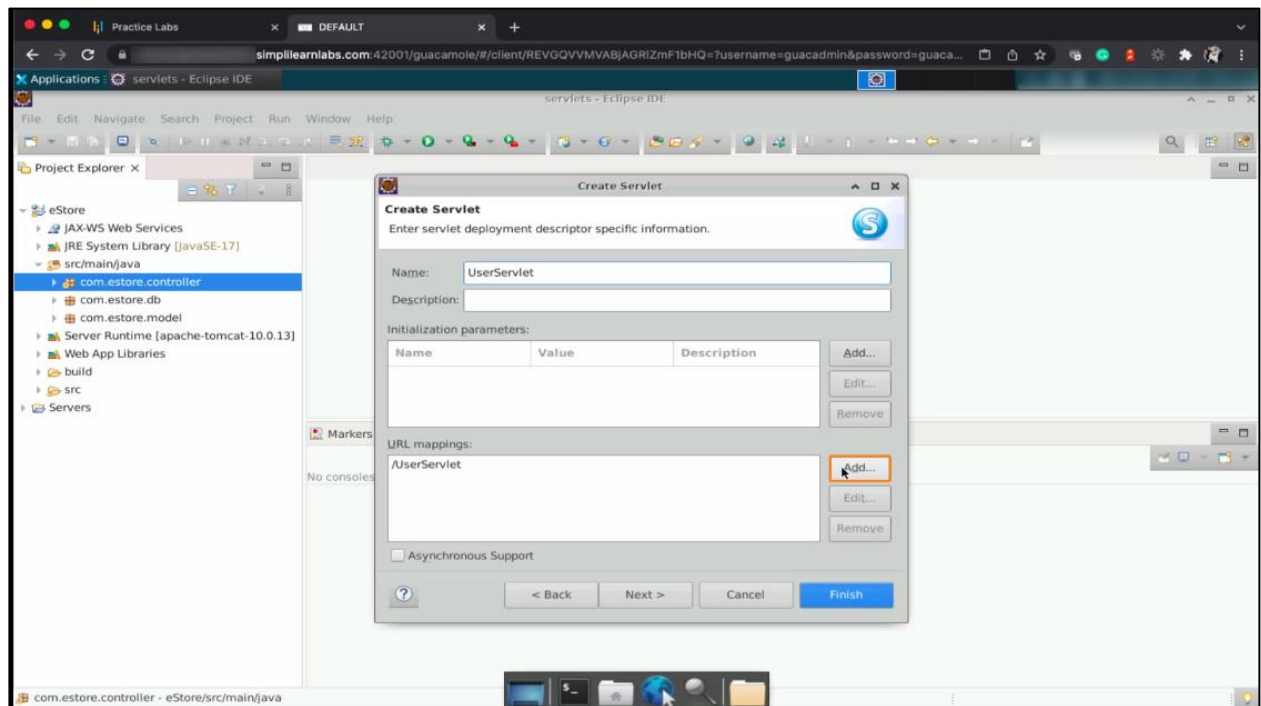
1.3 Provide the name of the class as **UserServlet**



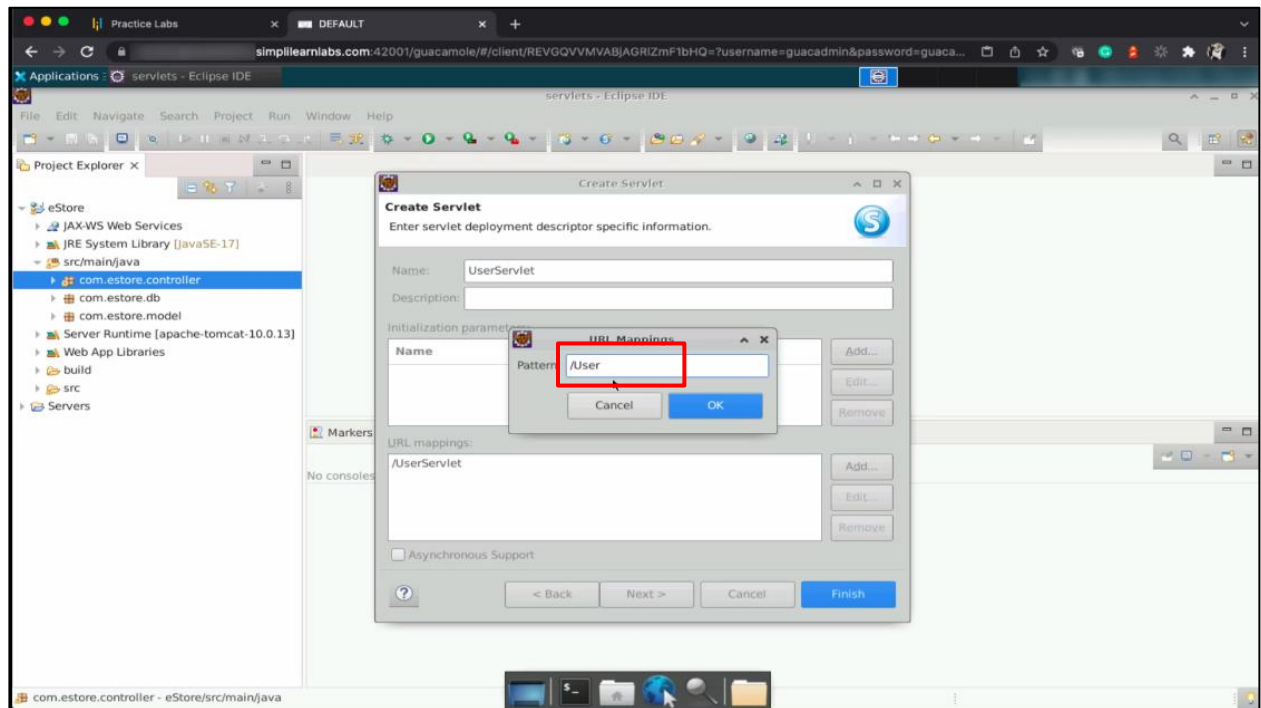
1.4 Click on Next



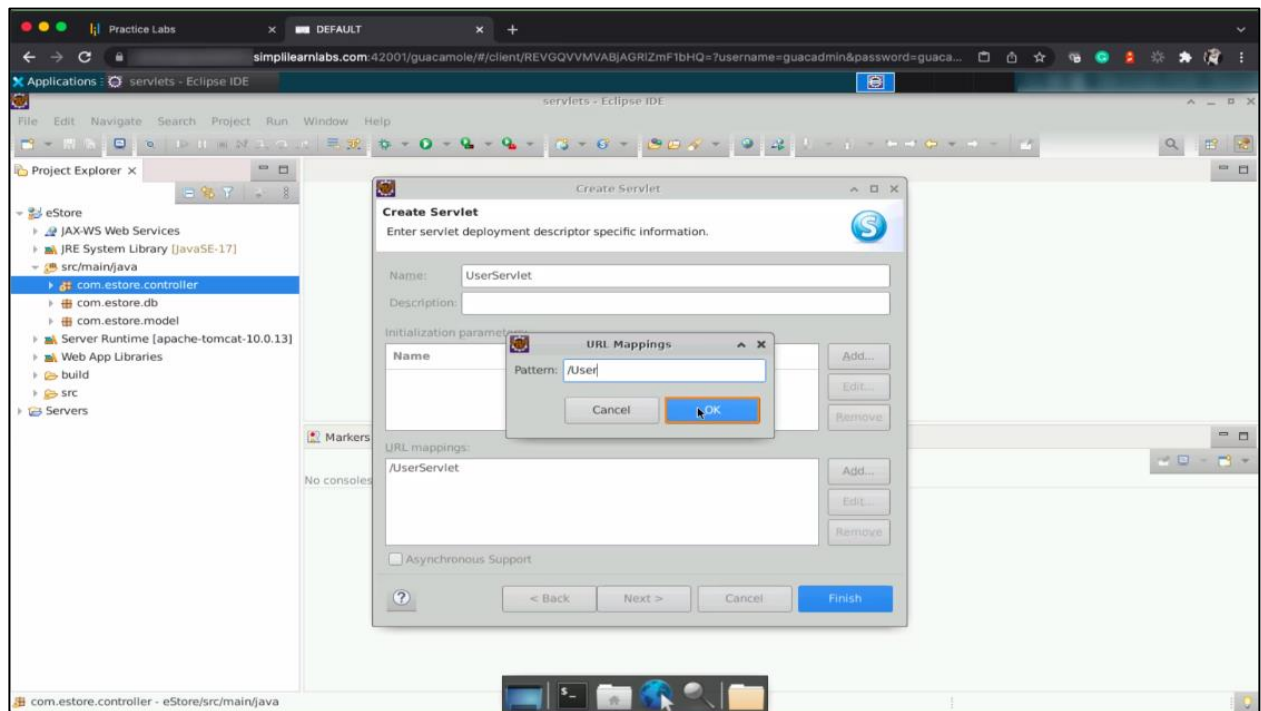
1.5 Click on Add



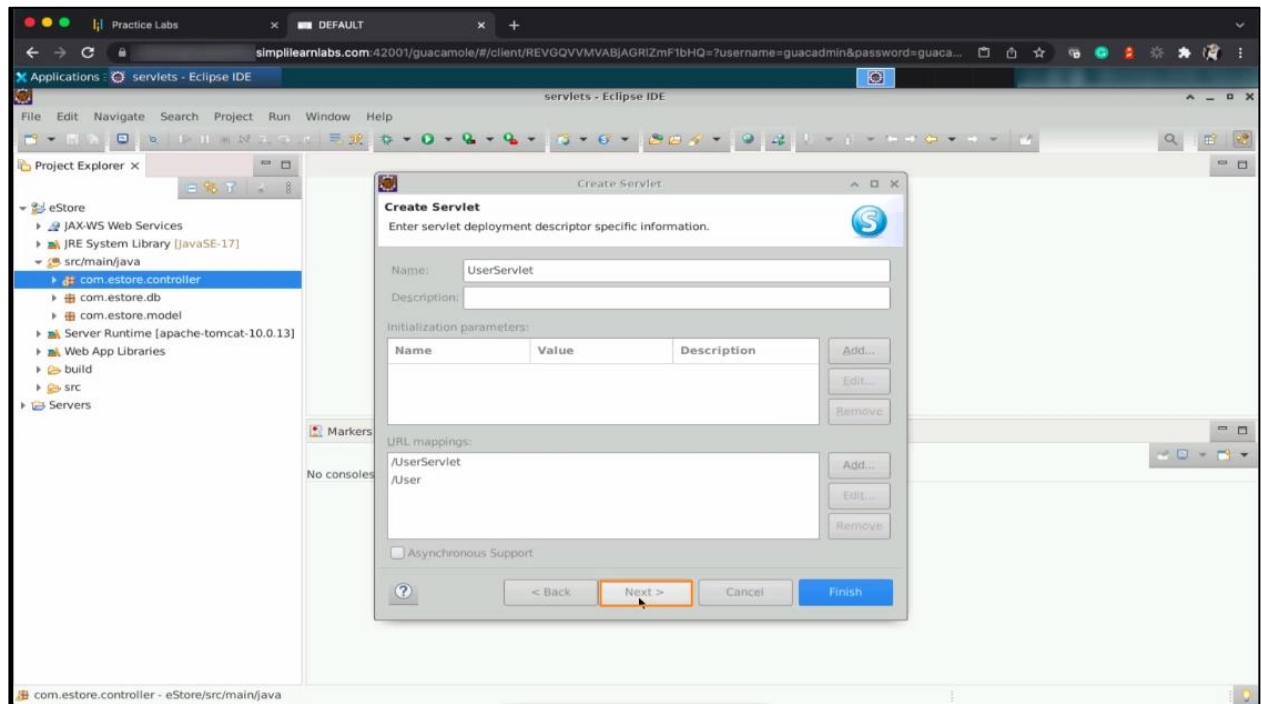
1.6 Map the URL by writing the pattern as /User



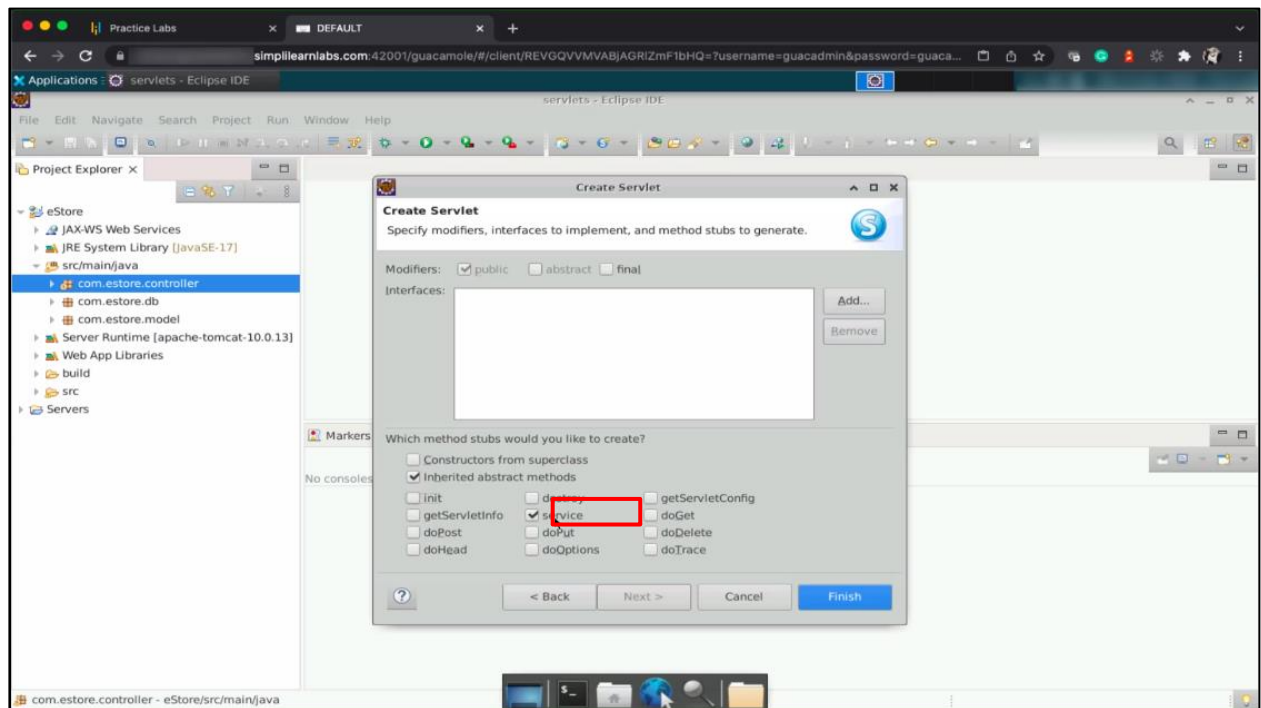
1.7 Click on OK



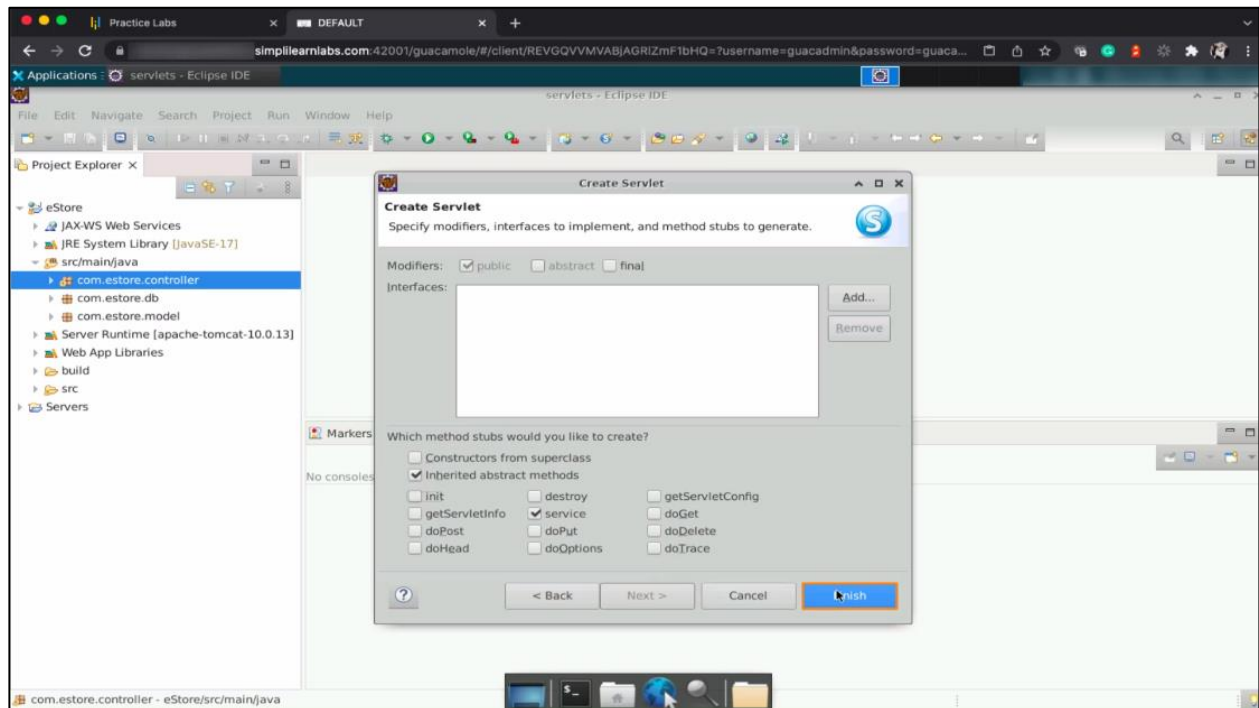
1.8 Click on Next



1.9 Check the service method box

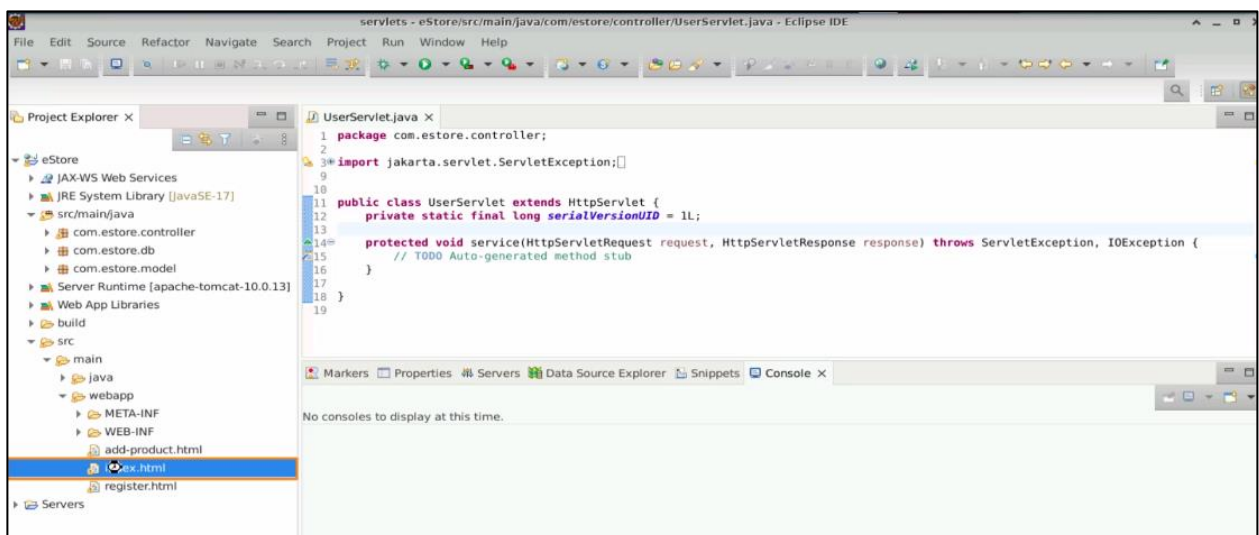


1.10 Click on **Finish**

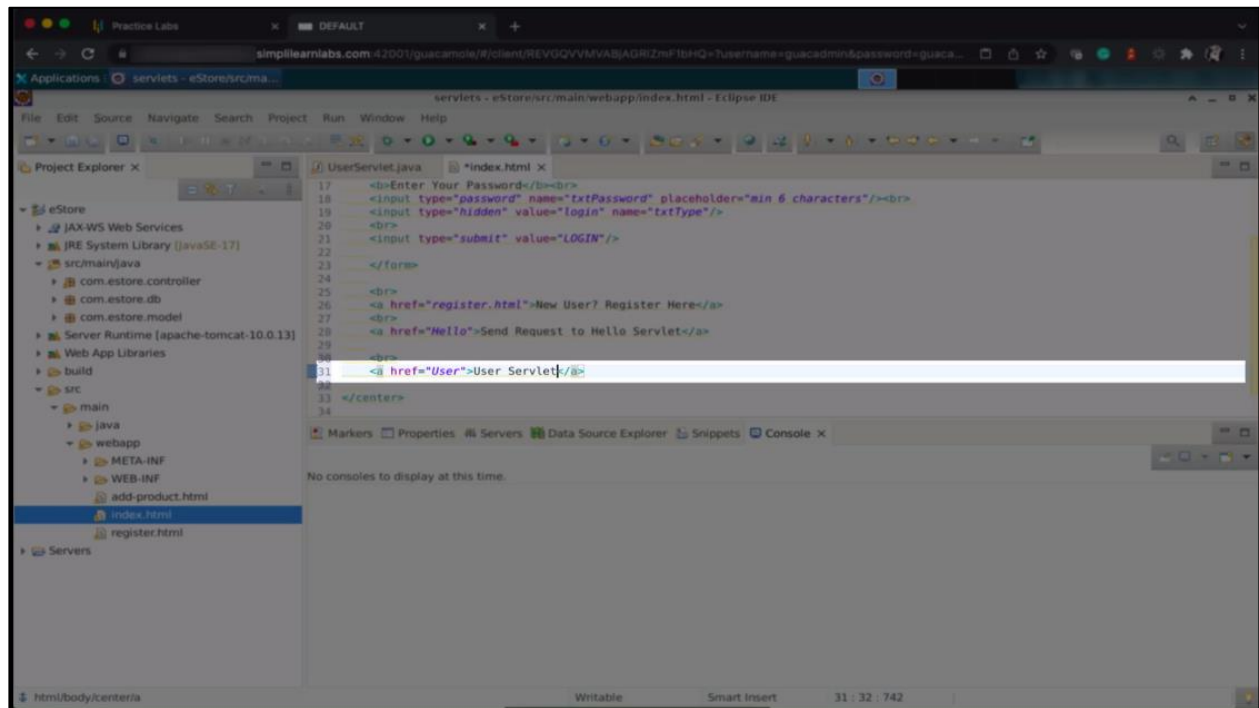


Step 2: Create a hyperlink and run the application to check the output

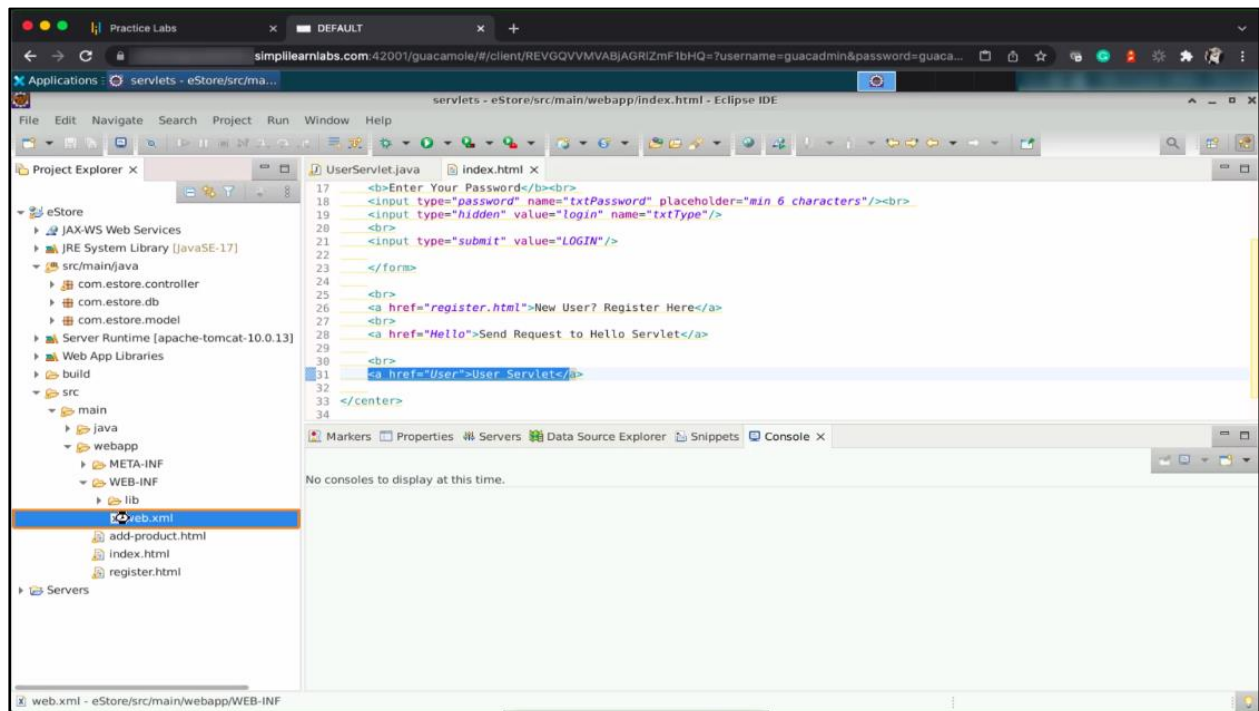
2.1 Open the **index.html** file

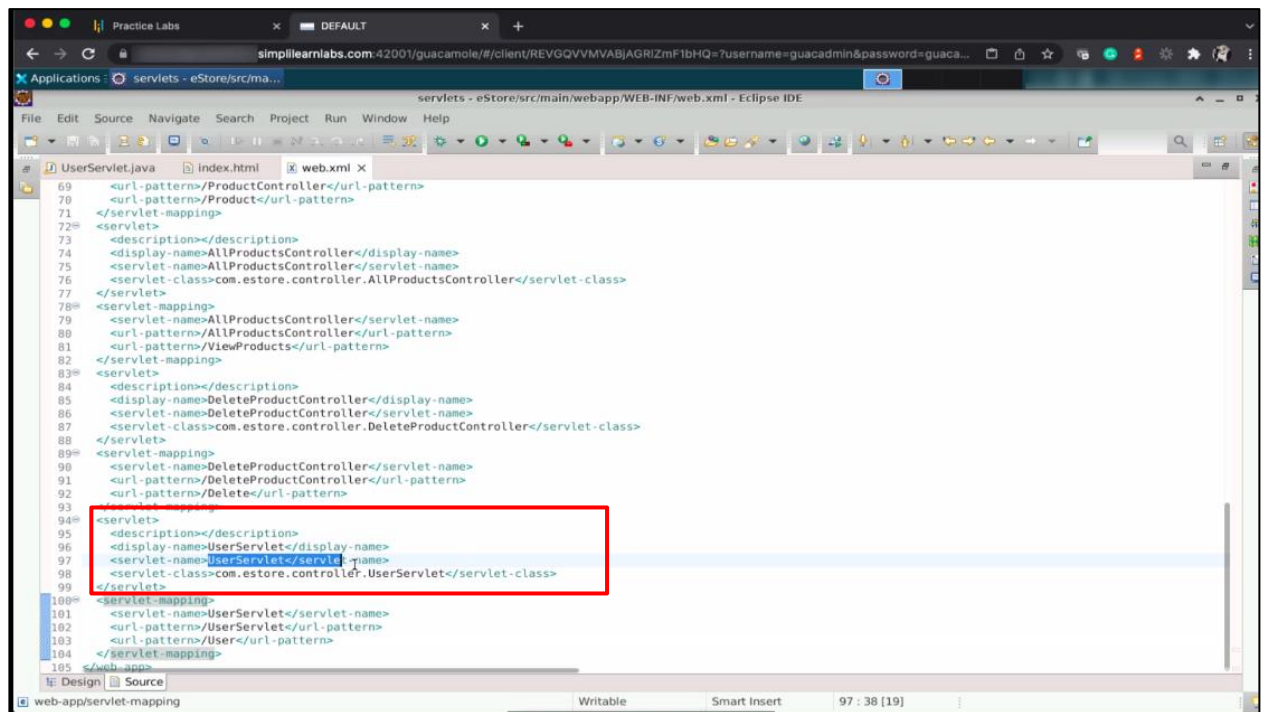


2.2 Create a **hyperlink** for requesting **User servlet**



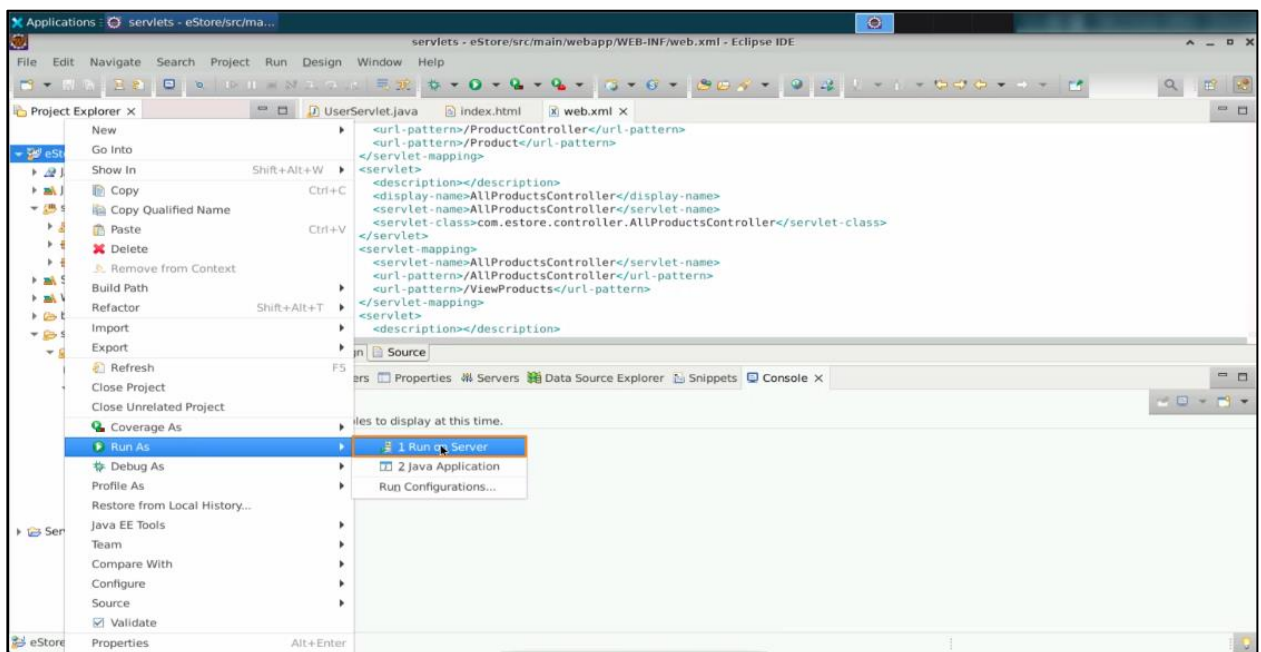
2.3 Open the **web.xml** file



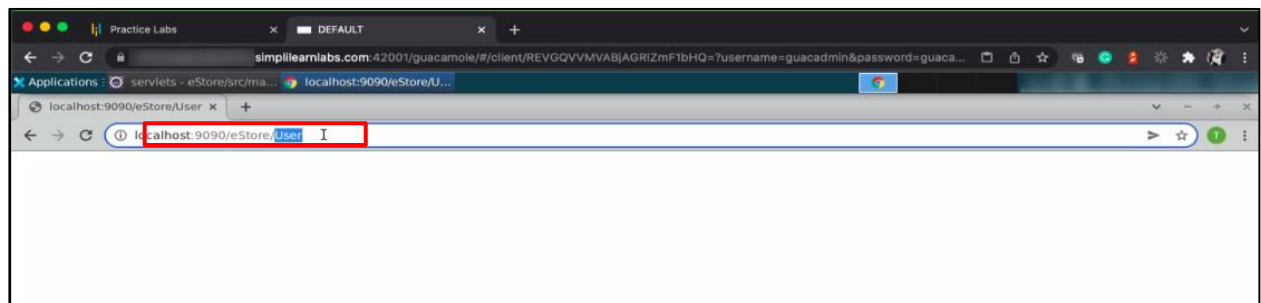
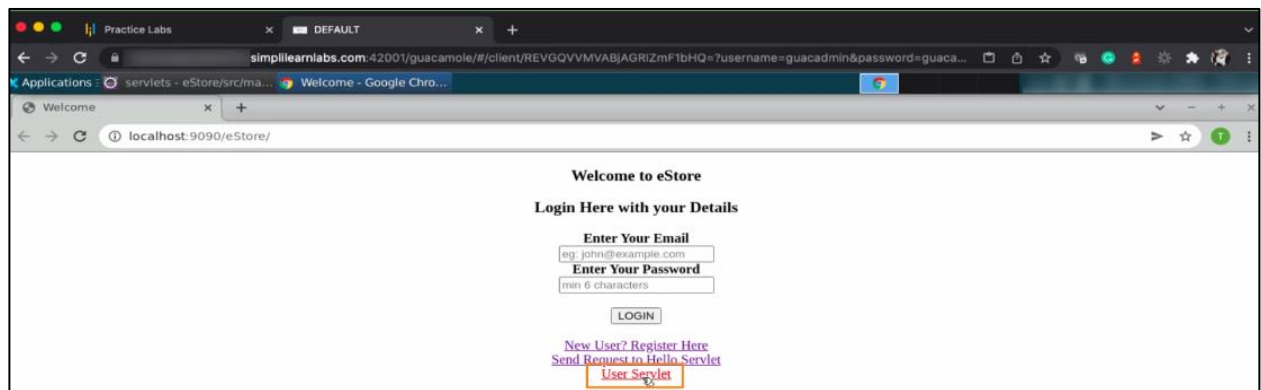


You can see the **UserServlet** tag with two URL patterns.

2.4 Run the project on the server. Right-click on **estore**, select **Run As**, and then select **Run on Server**



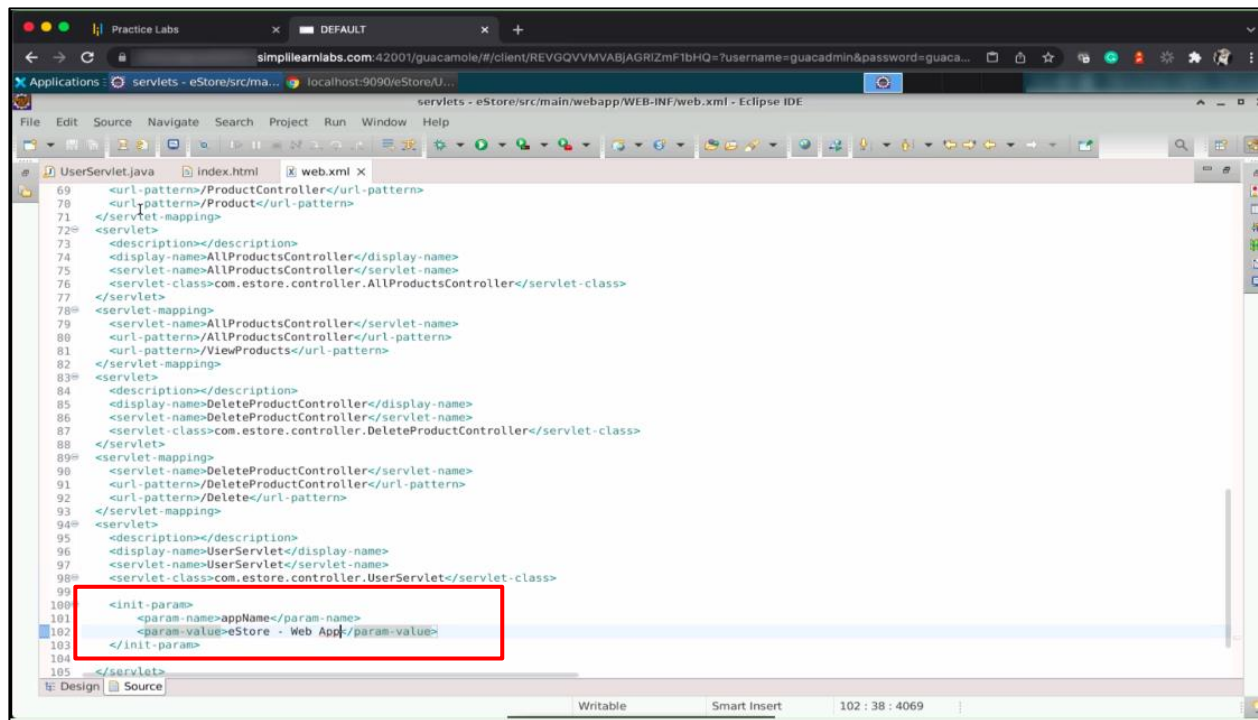
2.5 Click on User servlet



There is no response from the server for now.

Step 3: Create the getServerConfig method

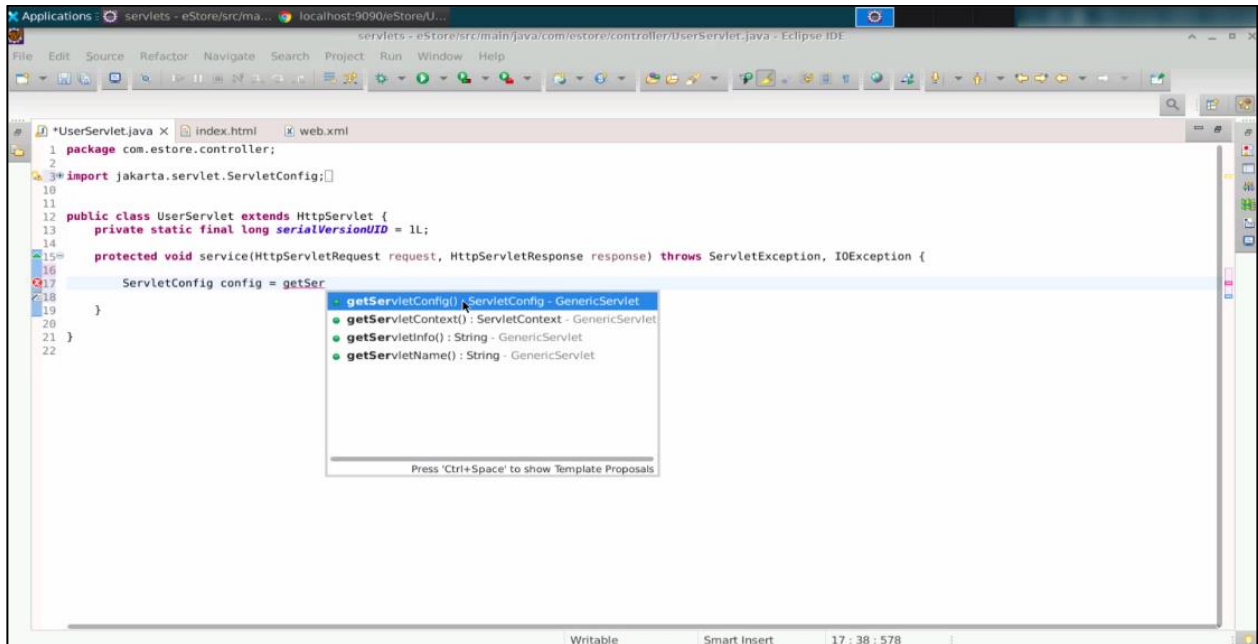
3.1 Come back to the **web.xml** file and create the **init-param** sub-tag in the **servlet** tag for the configuration of the servlet. Also, create **param-name** and **param-value**, two more subtags in the **init-param** tag



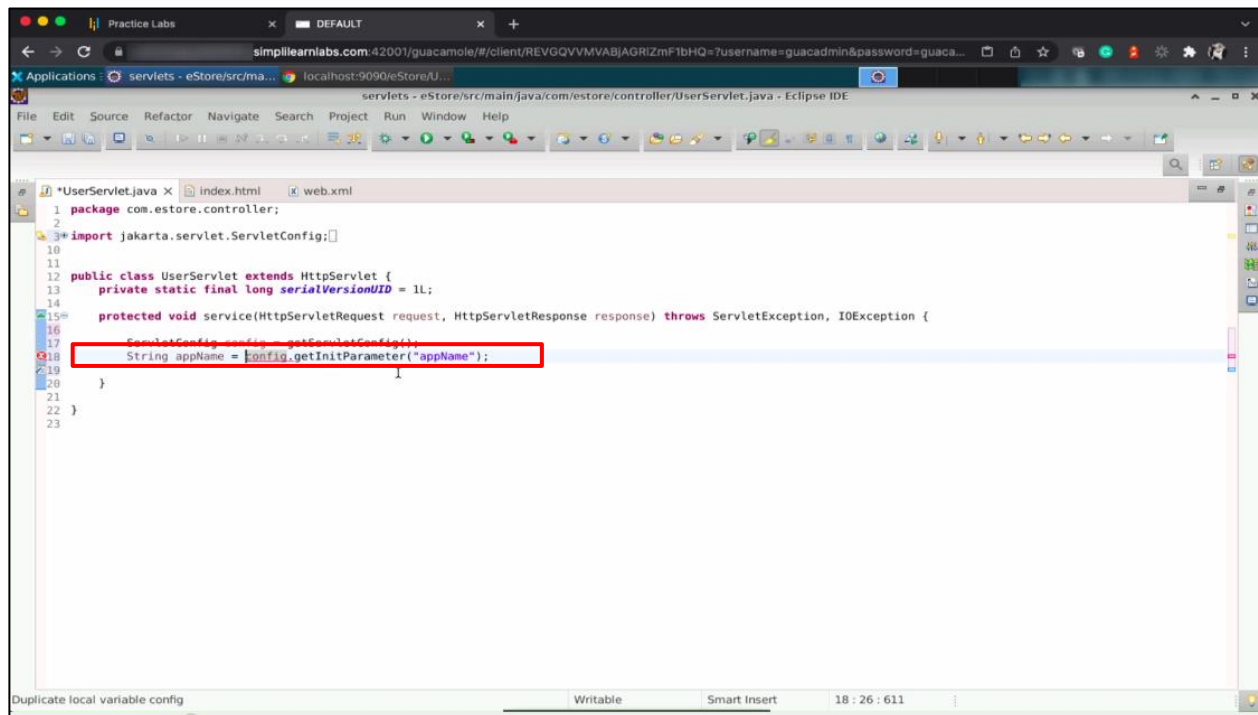
The screenshot shows the Eclipse IDE with the `web.xml` file open. The file contains several servlet mappings. The `<init-param>` tag is highlighted with a red box, and its contents are as follows:

```
100 <init-param>
101 <param-name>appName</param-name>
102 <param-value>eStore - Web App</param-value>
103 </init-param>
```

3.2 Go to the **UserServlet.java** file and inherit the **getServerConfig** method, which is already created and can be used because of the parent relationship with the HTTP servlet



3.3 Add an **init** parameter where you must get the **name** of **init-param**

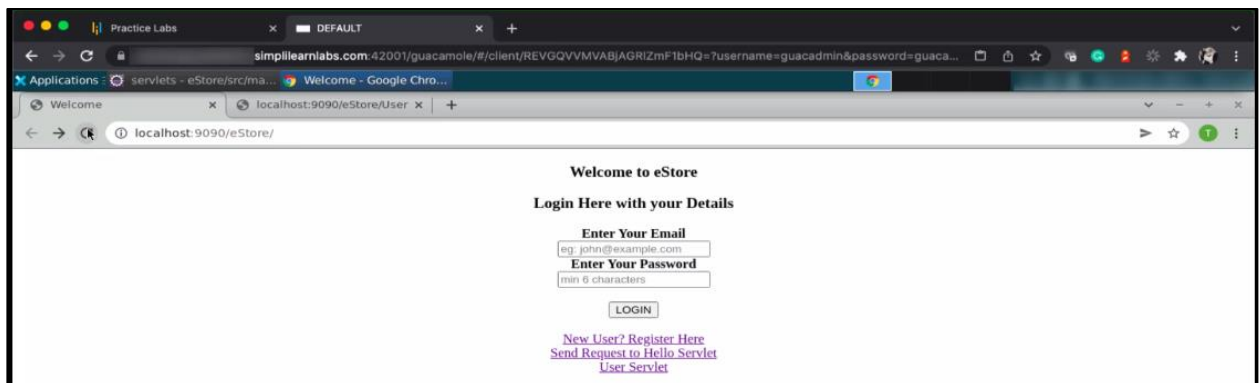


3.4 Set the **responses** to be shared on UI and **print** them back to the client

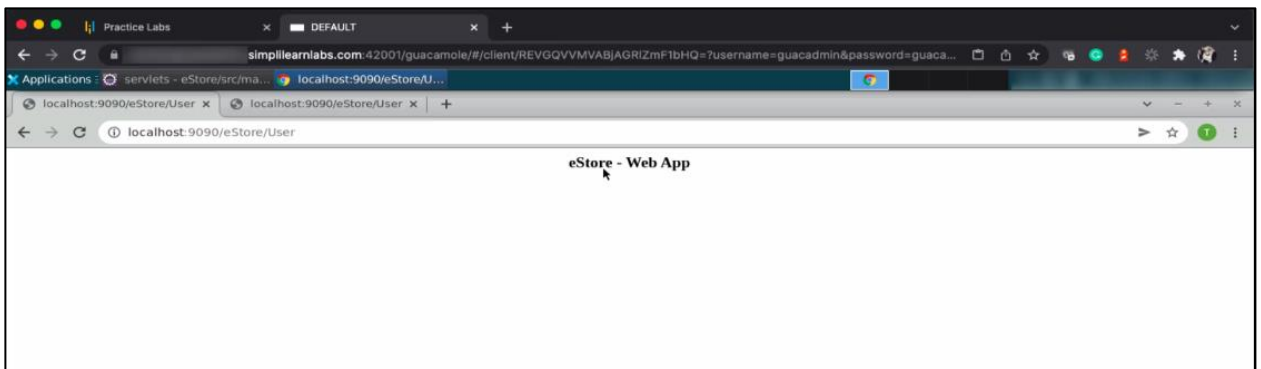
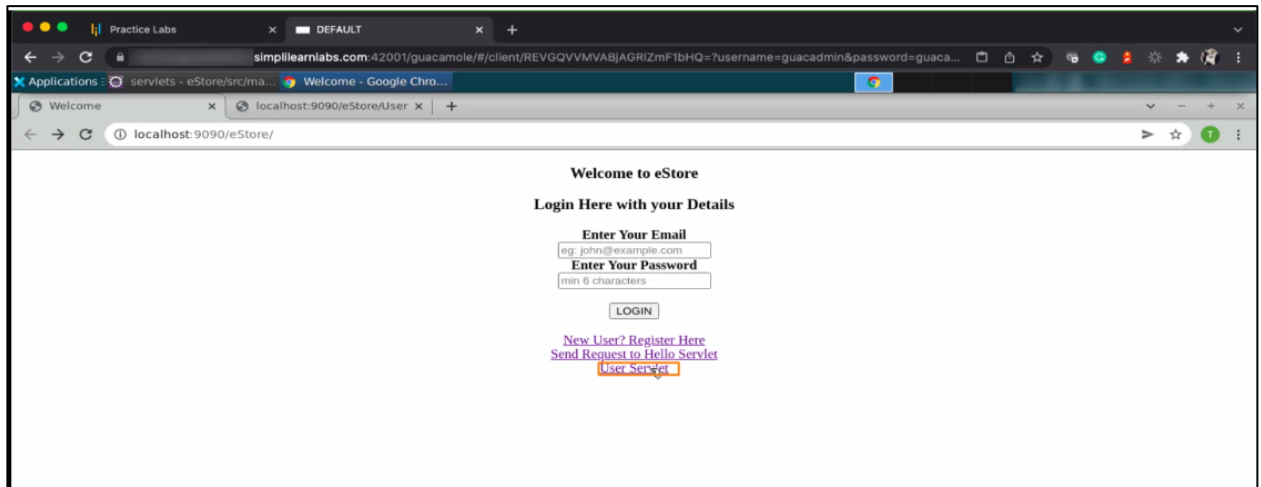
```

1 package com.estore.controller;
2
3 import jakarta.servlet.ServletException;
4
11
12
13 public class UserServlet extends HttpServlet {
14     private static final long serialVersionUID = 1L;
15
16     protected void service(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
17         response.setContentType("text/html");
18         PrintWriter out = response.getWriter();
19
20         ServletConfig config = getServletConfig();
21         String appName = config.getInitParameter("appName");
22
23         StringBuffer responseText = new StringBuffer();
24         responseText.append("<center>");
25         responseText.append("<h3>" + appName + "</h3>");
26         responseText.append("</center>");
27
28         out.print(responseText.toString());
29
30     }
31
32 }
33
34
  
```

3.5 Go back to the browser and refresh the screen



3.6 Click on User servlet



You can see the output as **eStore - Web App** in response.

Step 4: Create and extract parameters

4.1 Come back to the **web.xml** file and create three different **init** parameters

```

85 <display-name>DeleteProductController</display-name>
86 <servlet-name>DeleteProductController</servlet-name>
87 <servlet-class>com.estore.controller.DeleteProductController</servlet-class>
88 </servlet>
89 <servlet-mapping>
90 <servlet-name>DeleteProductController</servlet-name>
91 <url-pattern>/DeleteProductController</url-pattern>
92 </servlet-mapping>
93 </servlet>
94 <servlet>
95 <description></description>
96 <display-name>UserServicelet</display-name>
97 <servlet-name>UserServicelet</servlet-name>
98 <servlet-class>com.estore.controller.UserServicelet</servlet-class>
99
100 <init-param>
101 <param-name>appName</param-name>
102 <param-value>eStore - Web App</param-value>
103 </init-param>
104
105 <init-param>
106 <param-name>accessToken</param-name>
107 <param-value>1200AB9A</param-value>
108 </init-param>
109
110 <init-param>
111 <param-name>memoryUsageThreshold</param-name>
112 <param-value>16</param-value>
113 </init-param>
114
115 </servlet>
116 <servlet-mapping>
117 <servlet-name>UserServicelet</servlet-name>
118 <url-pattern>/UserServicelet</url-pattern>
119 <url-pattern>/User</url-pattern>
120 </servlet-mapping>
121 </web-app>

```

4.2 Come back to the **UserServicelet.java** file, and you can read these parameters one by one. This is one of the ways you extract the parameter.

```

1 package com.estore.controller;
2
3 import jakarta.servlet.ServletException;
4
5 public class UserServicelet extends HttpServlet {
6     private static final long serialVersionUID = 1L;
7
8     protected void service(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
9         response.setContentType("text/html");
10        PrintWriter out = response.getWriter();
11
12        ServletConfig config = getServletConfig();
13        String appName = config.getInitParameter("appName");
14        String memoryUsageThreshold = config.getInitParameter("memoryUsageThreshold");
15
16        StringBuffer responseText = new StringBuffer();
17        responseText.append("<center>");
18        responseText.append("<h3>" + appName + "</h3>");
19        responseText.append("</center>");
20
21        out.print(responseText.toString());
22    }
23 }

```


4.3 Mark the following line as a comment:

```

1 package com.estore.controller;
2
3 import jakarta.servlet.ServletConfig;
4
5 public class UserServicelet extends HttpServlet {
6     private static final long serialVersionUID = 1L;
7
8     protected void service(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
9         response.setContentType("text/html");
10        PrintWriter out = response.getWriter();
11
12        ServletConfig config = getServletConfig();
13        String appName = config.getInitParameter("appName");
14        //String memoryUsageThreshold = config.getInitParameter("memoryUsageThreshold");
15
16        StringBuffer responseText = new StringBuffer();
17        responseText.append("<center>");
18        responseText.append("<h3>"+appName+"</h3>");
19        responseText.append("</center>");
20
21        out.print(responseText.toString());
22    }
23 }

```

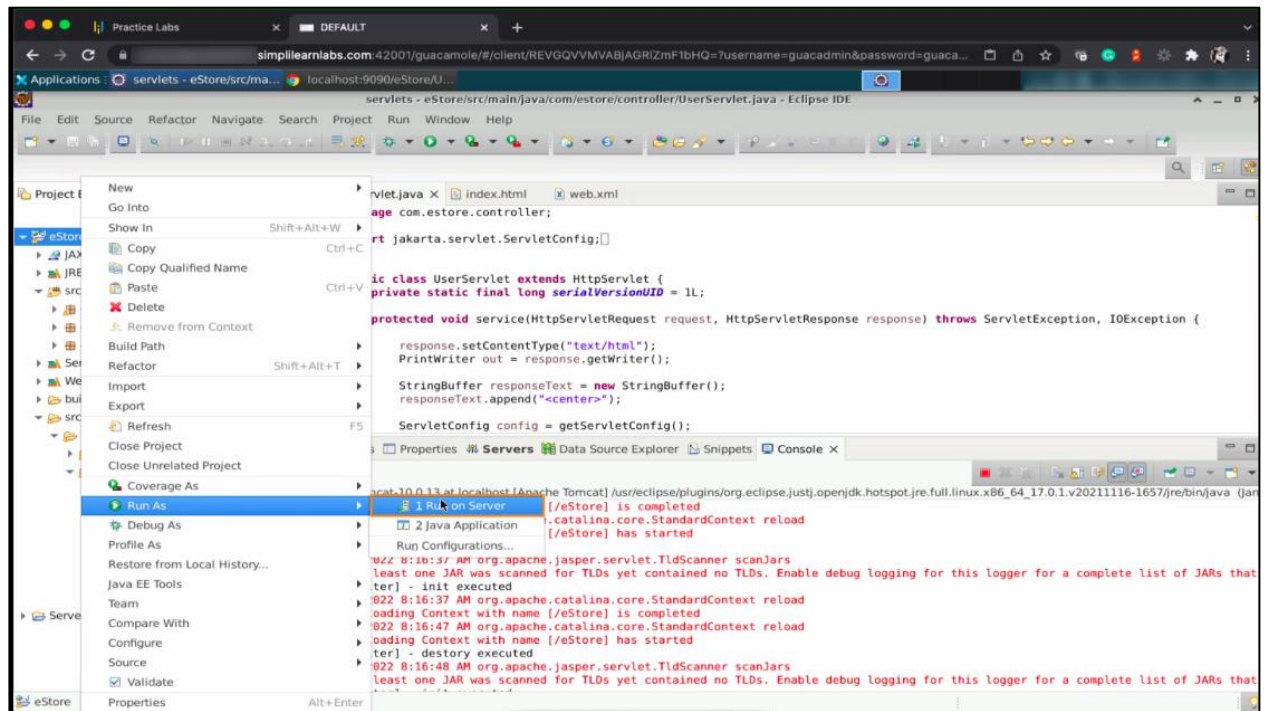
4.4 Capture all parameters using Enumeration, then iterate using a while loop through these parameters. This is another way how you extract the parameter.

```

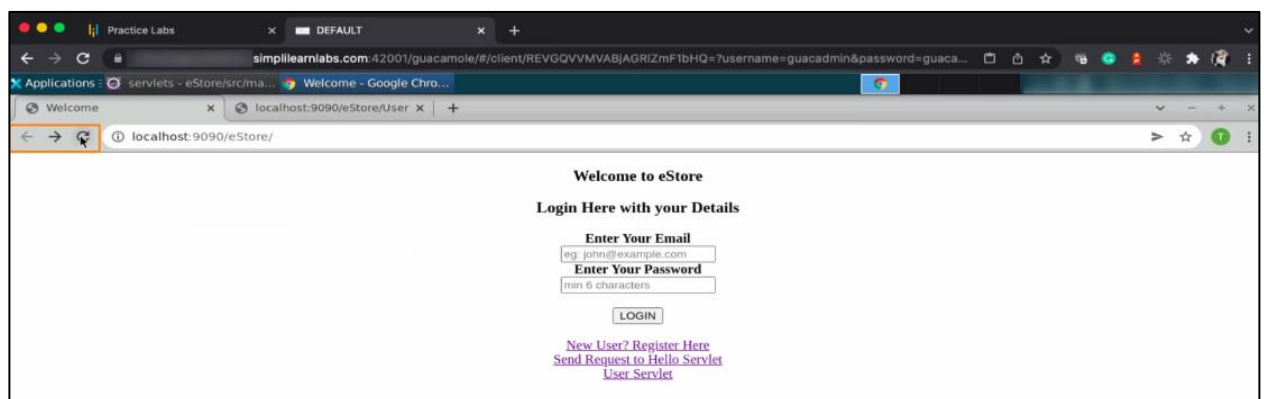
1 package com.estore.controller;
2
3 import jakarta.servlet.ServletConfig;
4
5 public class UserServicelet extends HttpServlet {
6     private static final long serialVersionUID = 1L;
7
8     protected void service(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
9         response.setContentType("text/html");
10        PrintWriter out = response.getWriter();
11
12        StringBuffer responseText = new StringBuffer();
13        responseText.append("<center>");
14
15        ServletConfig config = getServletConfig();
16        String appName = config.getInitParameter("appName");
17        //String memoryUsageThreshold = config.getInitParameter("memoryUsageThreshold");
18
19        responseText.append("<h3>"+appName+"</h3>");
20
21        Enumeration<String> paramNames = config.getInitParameterNames();
22        while(paramNames.hasMoreElements()) {
23            String name = paramNames.nextElement(); // parameter name
24            String value = config.getInitParameter(name);
25            responseText.append("<p>"+name+" "+value+"</p>");
26        }
27
28        responseText.append("</center>");
29
30        out.print(responseText.toString());
31    }
32 }

```

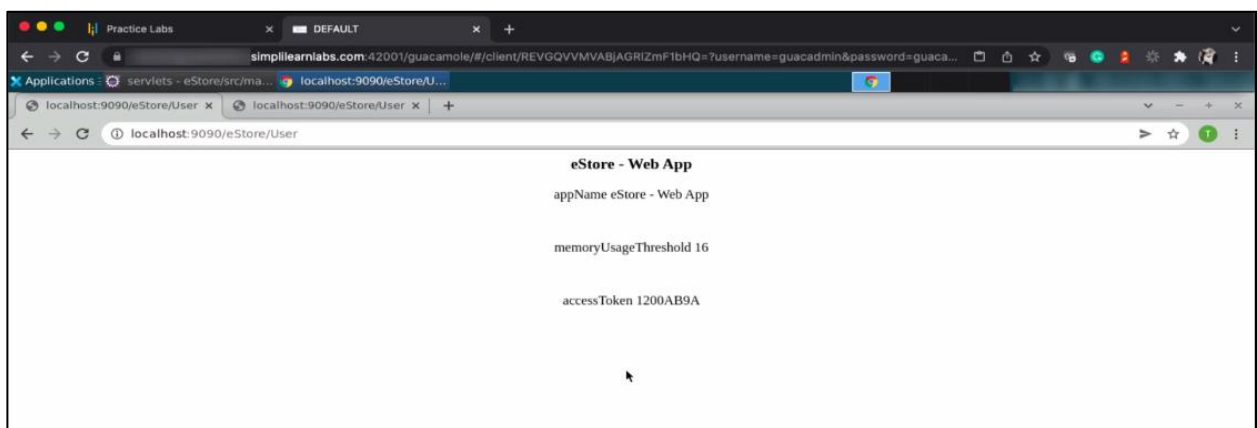
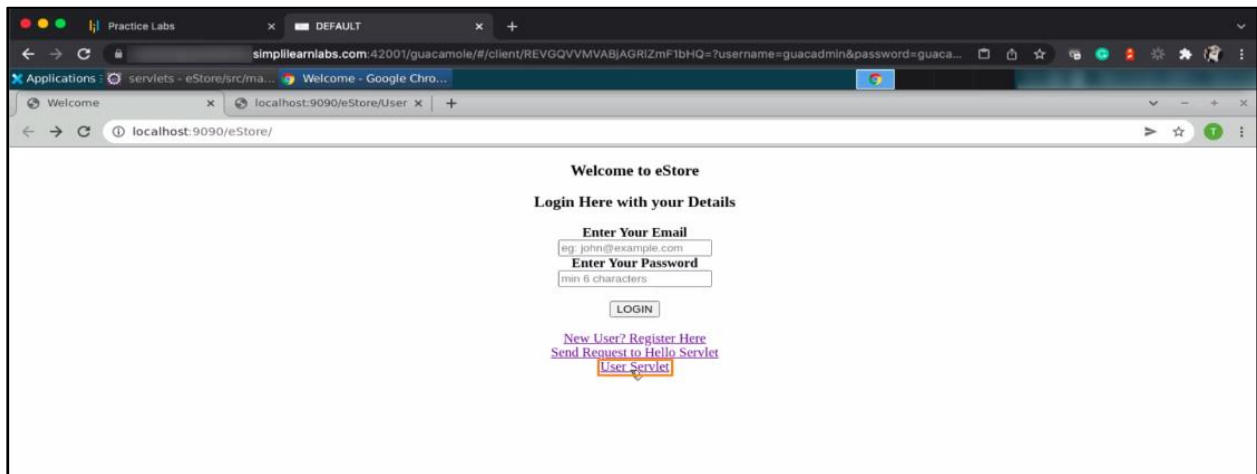
4.5 Run the project on the server by right-clicking on it, selecting **Run As**, and then **Run on Server**



4.6 Go back to the browser and refresh the screen



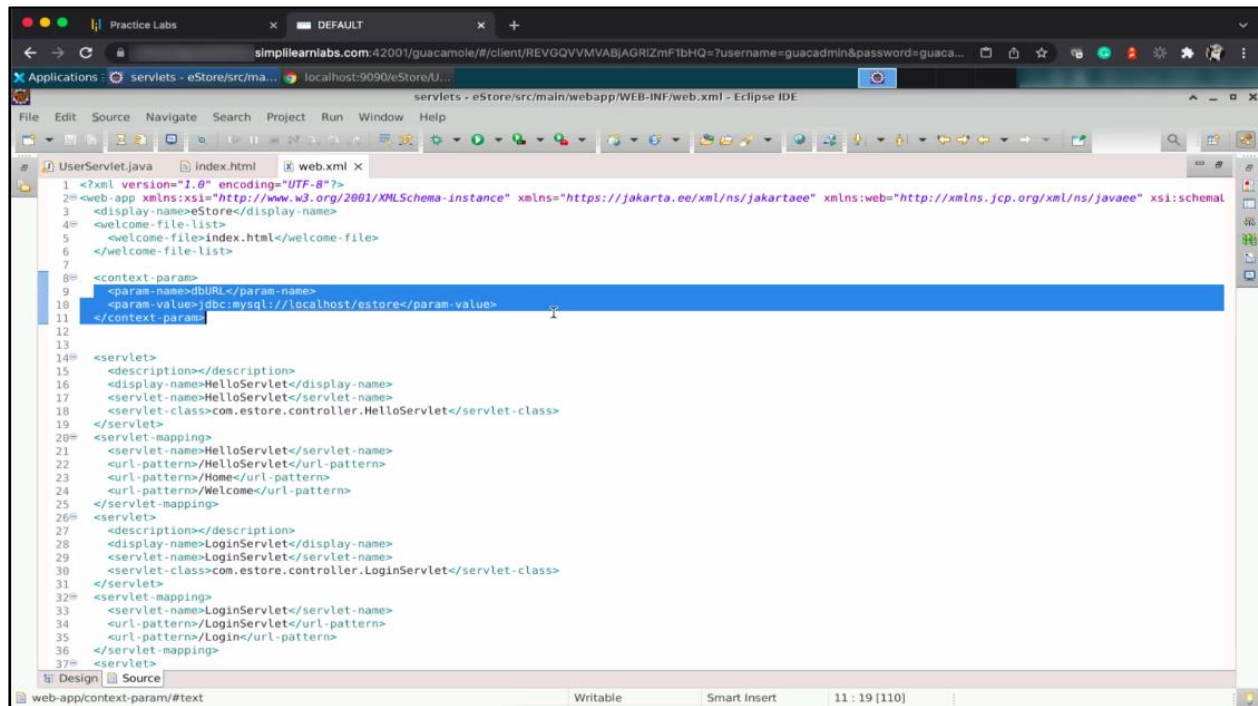
4.7 Click on the link **User servlet**



You can see all **init** parameters in the output, but they are disordered.

Step 5: Create multiple parameters and methods

5.1 Come back to **web.xml** and create **context-param**, which is available throughout the web application, and any servlet can use it

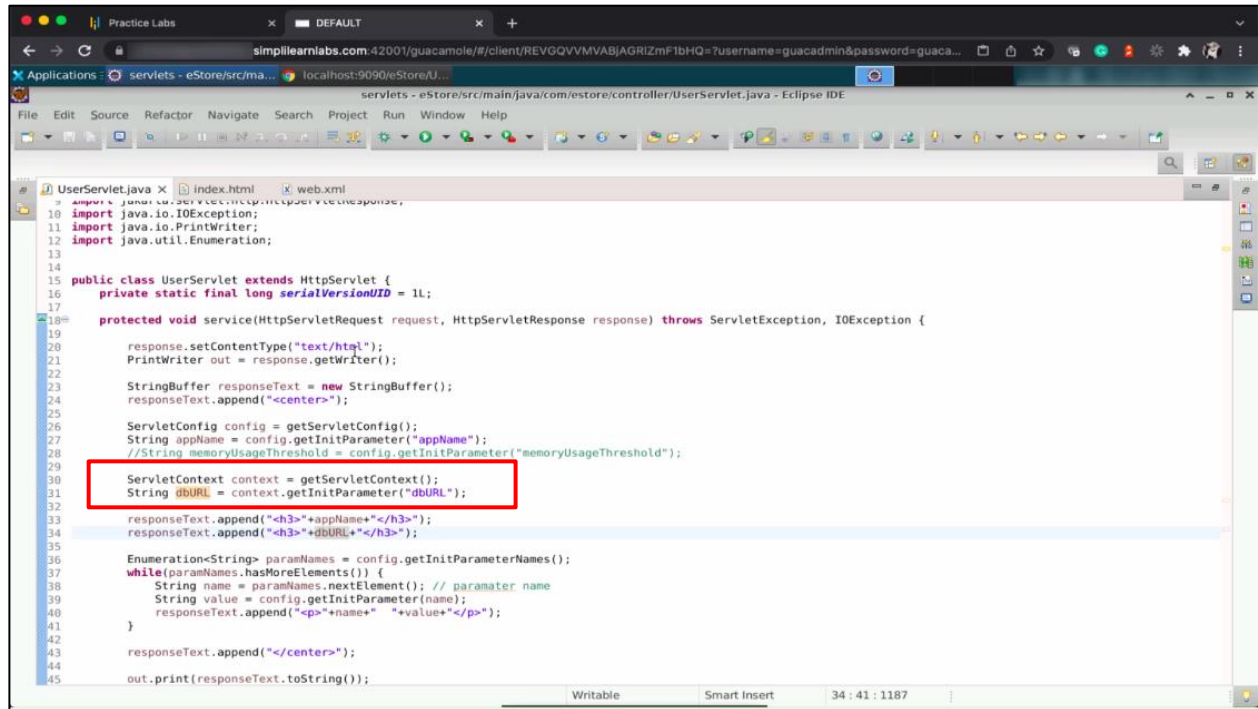


```

1 <?xml version="1.0" encoding="UTF-8"?>
2 <web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="https://jakarta.ee/xml/ns/jakartaee" xmlns:web="http://xmlns.jcp.org/xml/ns/javaee" xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/javaee web-app_3_1.xsd" >
3   <display-name>eStore</display-name>
4   <welcome-file-list>
5     <welcome-file>index.html</welcome-file>
6   </welcome-file-list>
7
8   <context-param>
9     <param-name>dbURL</param-name>
10    <param-value>jdbc:mysql://localhost/estore/param.value</param-value>
11  </context-param>
12
13
14  <servlet>
15    <description></description>
16    <display-name>HelloServlet</display-name>
17    <servlet-name>HelloServlet</servlet-name>
18    <servlet-class>com.estore.controller.HelloServlet</servlet-class>
19  </servlet>
20  <servlet-mapping>
21    <servlet-name>HelloServlet</servlet-name>
22    <url-pattern>/HelloServlet</url-pattern>
23    <url-pattern>/Home</url-pattern>
24    <url-pattern>/Welcome</url-pattern>
25  </servlet-mapping>
26  <servlet>
27    <description></description>
28    <display-name>LoginServlet</display-name>
29    <servlet-name>LoginServlet</servlet-name>
30    <servlet-class>com.estore.controller.LoginServlet</servlet-class>
31  </servlet>
32  <servlet-mapping>
33    <servlet-name>LoginServlet</servlet-name>
34    <url-pattern>/LoginServlet</url-pattern>
35    <url-pattern>/Login</url-pattern>
36  </servlet-mapping>
37  </web-app>

```

5.2 Go back to the **UserServlet.java** file and create the **ServletContext** method from the HTTP generic method

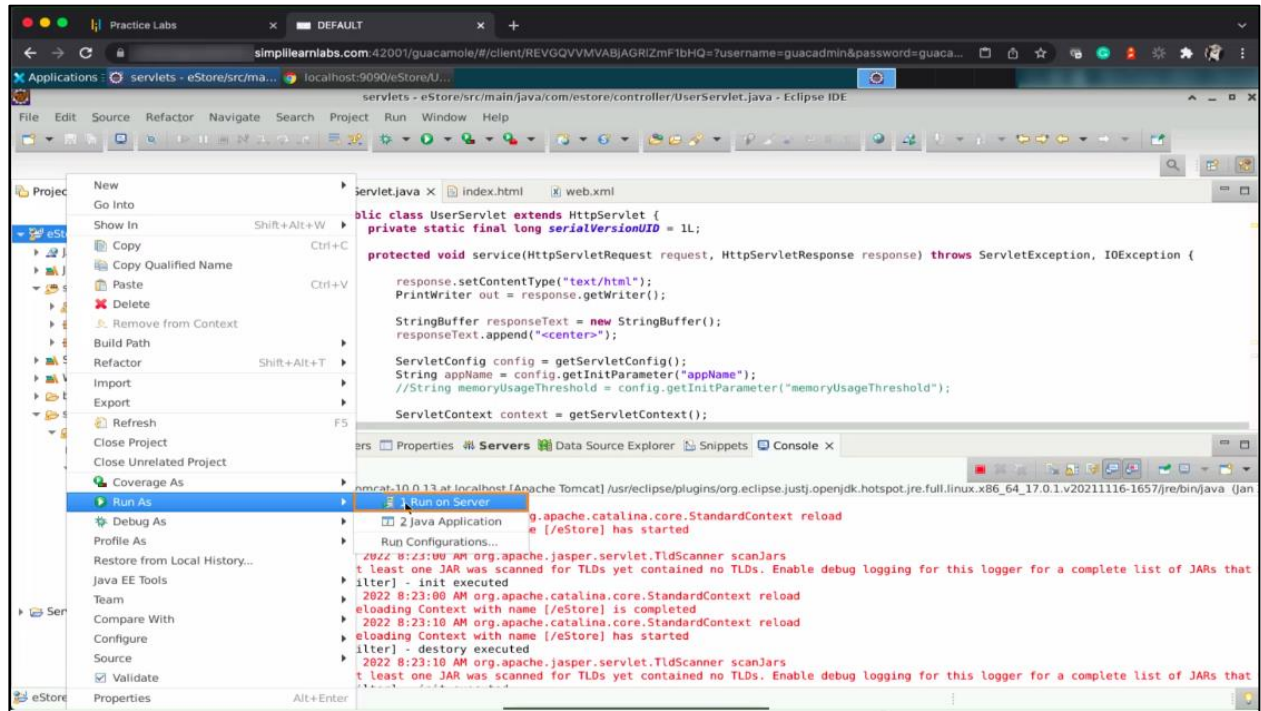


```

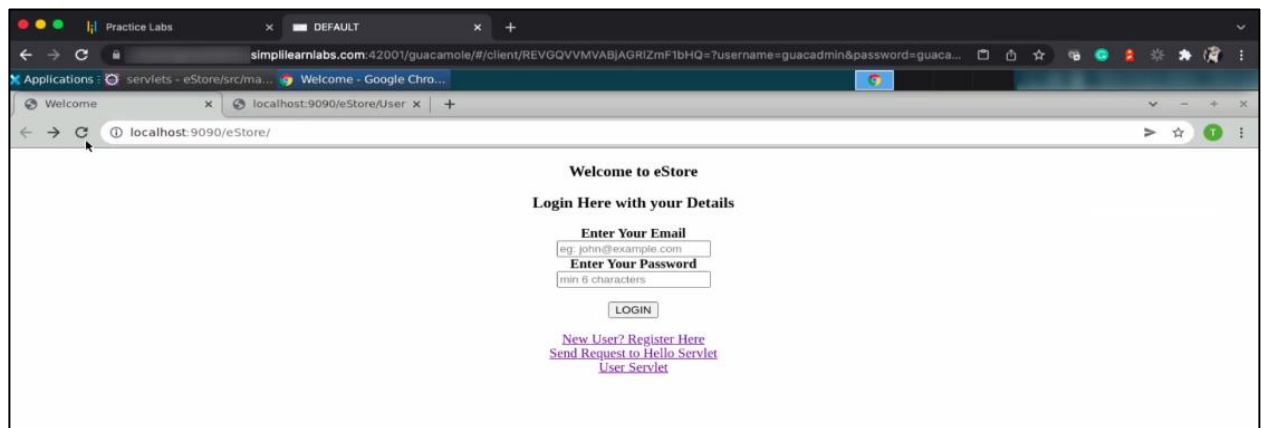
10 import java.io.IOException;
11 import java.io.PrintWriter;
12 import java.util.Enumeration;
13
14
15 public class UserServlet extends HttpServlet {
16     private static final long serialVersionUID = 1L;
17
18     protected void service(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
19
20         response.setContentType("text/html");
21         PrintWriter out = response.getWriter();
22
23         StringBuffer responseText = new StringBuffer();
24         responseText.append("<center>");
25
26         ServletConfig config = getServletConfig();
27         String appName = config.getInitParameter("appName");
28         //String memoryUsageThreshold = config.getInitParameter("memoryUsageThreshold");
29
30         ServletContext context = getServletContext();
31         String dbURL = context.getInitParameter("dbURL");
32
33         responseText.append("<h3>" + appName + "</h3>");
34         responseText.append("<h3>" + dbURL + "</h3>");
35
36         Enumeration<String> paramNames = config.getInitParameterNames();
37         while(paramNames.hasMoreElements()) {
38             String name = paramNames.nextElement(); // parameter name
39             String value = config.getInitParameter(name);
40             responseText.append("<p>" + name + " " + value + "</p>");
41         }
42
43         responseText.append("</center>");
44
45         out.print(responseText.toString());

```

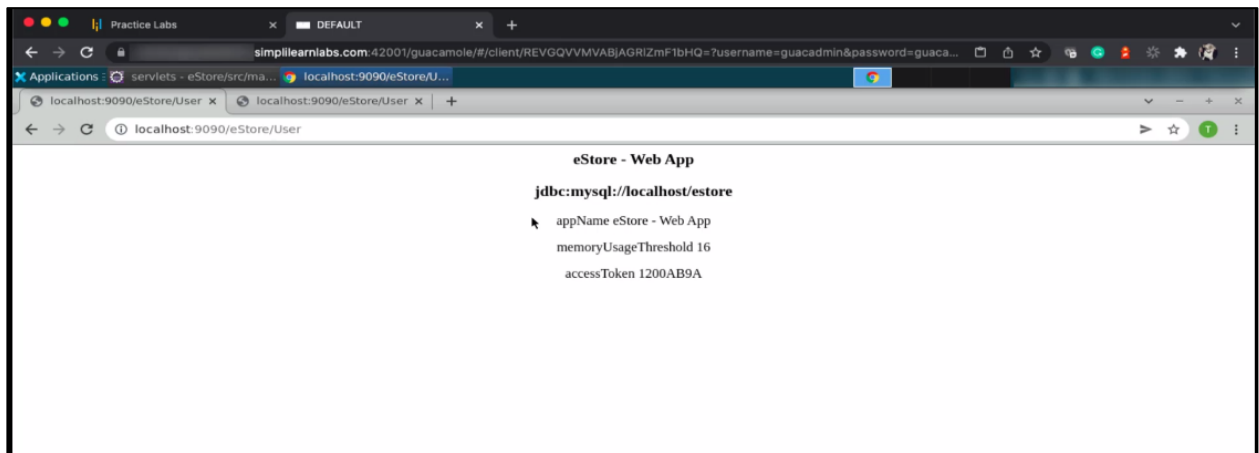
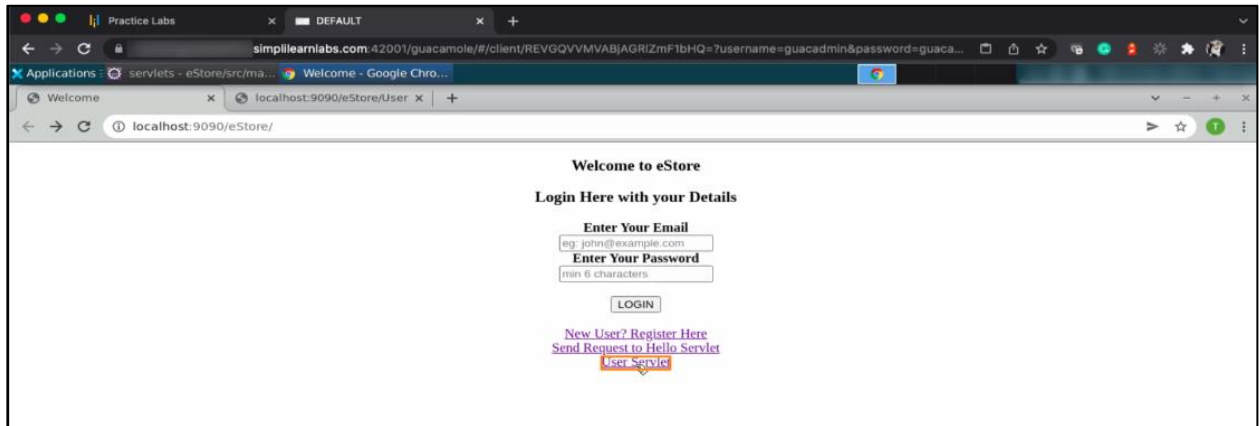
5.3 Click on the server. Right-click on the project, select **Run As**, and then select **Run on Server**



5.4 Go back to the browser and refresh the screen

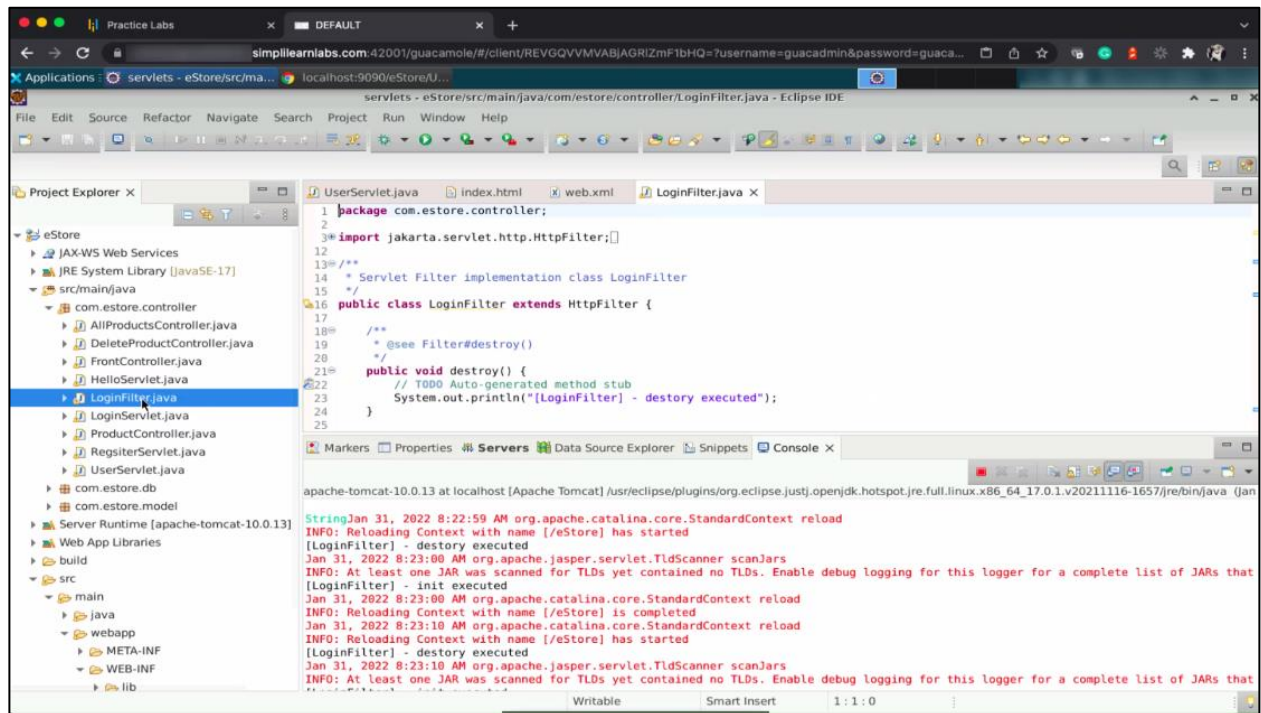


5.5 Click on the link **User servlet**

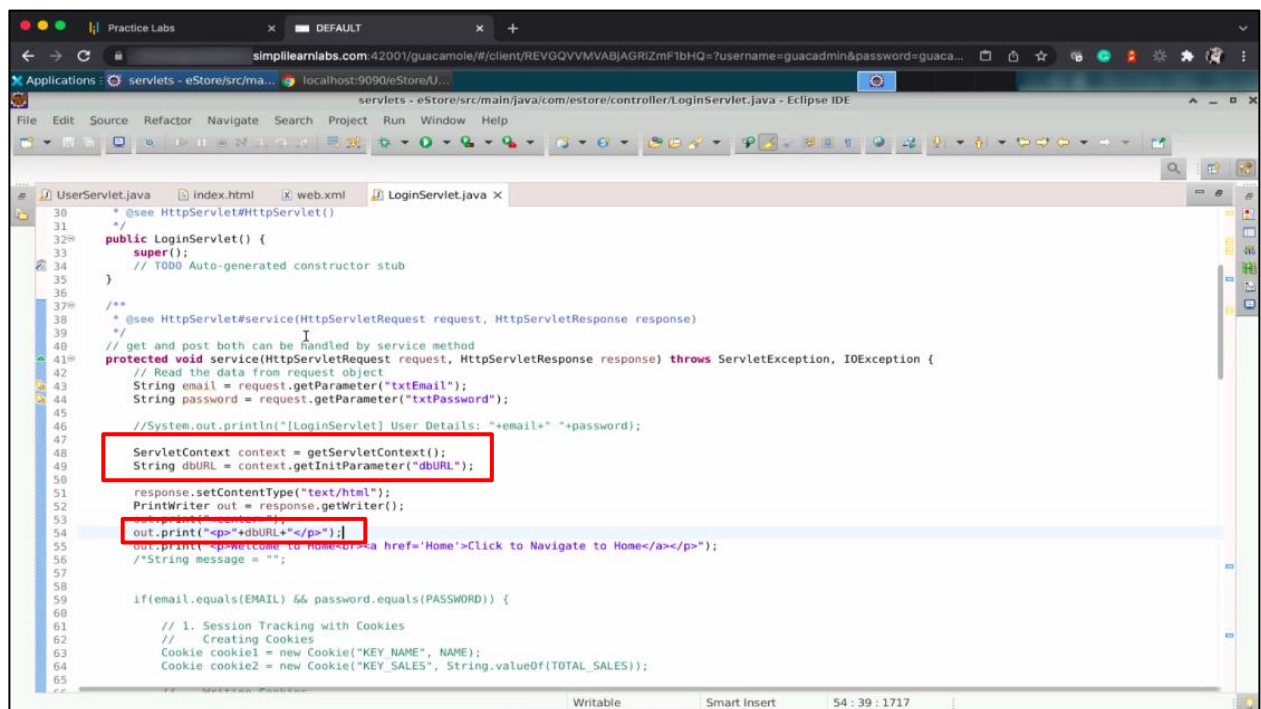


You can see **dbURL** in the output, which is **context-param**.

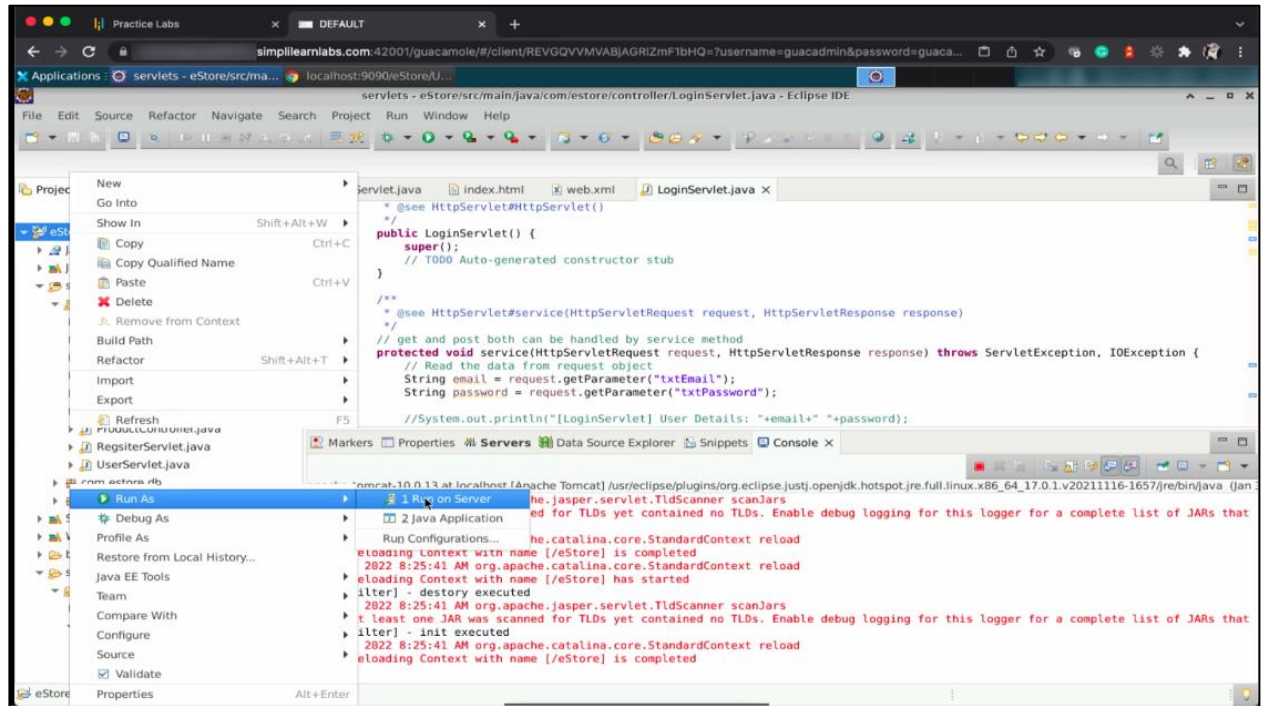
5.6 Click on **com.estore.controller** and open the **LoginFilter.java** file



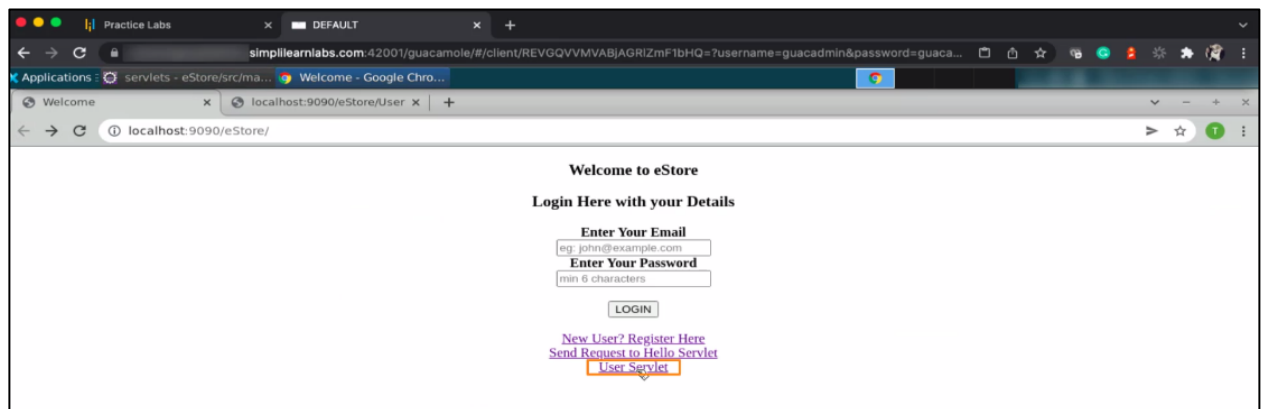
5.7 Create a similar **servletContext** method in the **LoginServlet.java** file

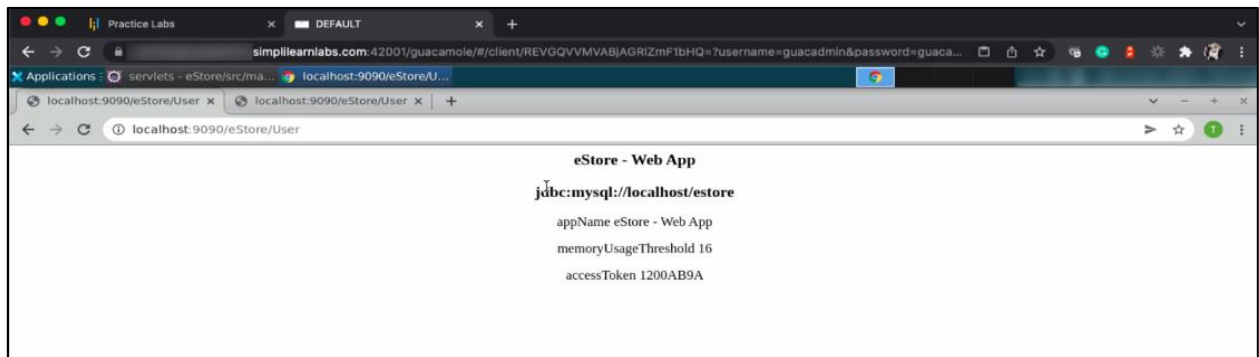


5.8 Run the project on the server. Right-click on the server, select **Run As**, and click **Run on Server**



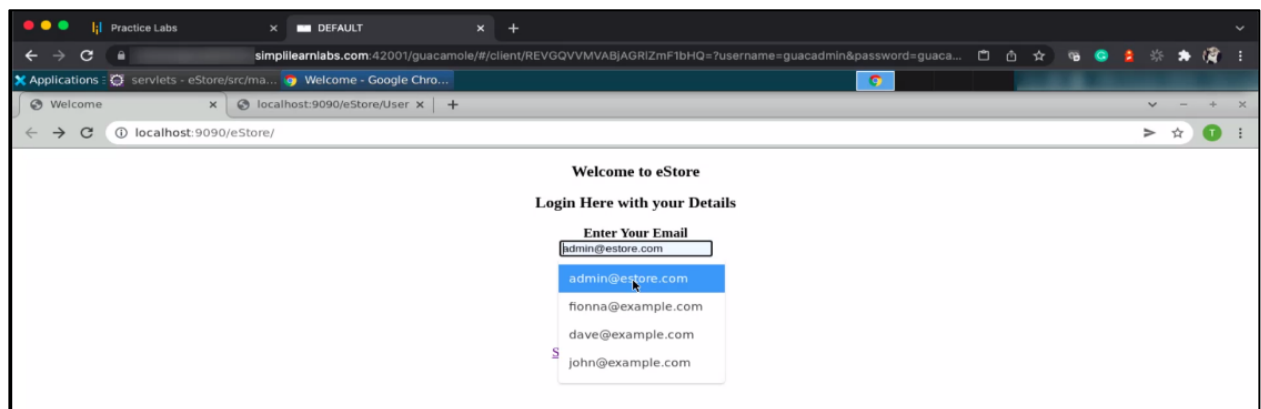
5.9 Click on the link **User servlet**



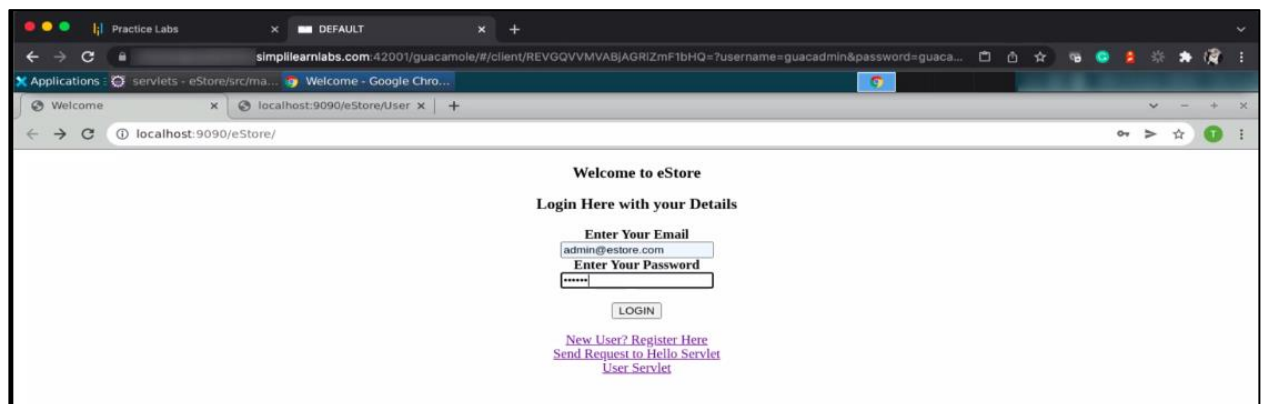


You can see that the output is the same as the previous output, where it prints the access tokens and the threshold value.

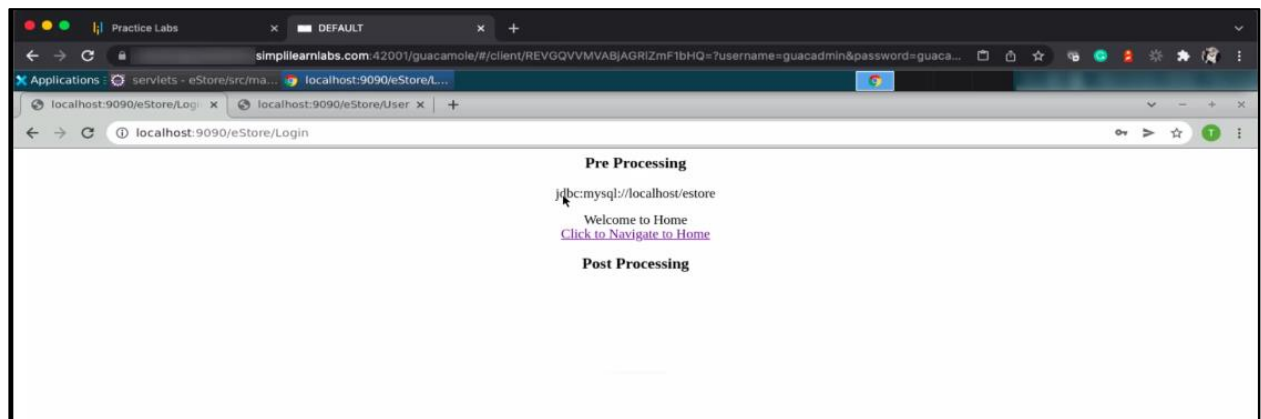
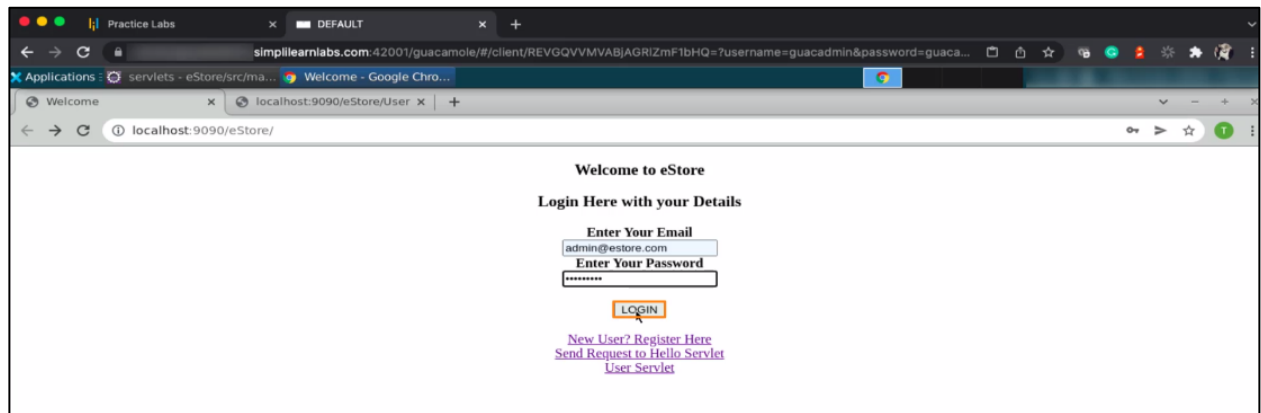
5.10 Go back and enter the email ID



5.11 Enter the password

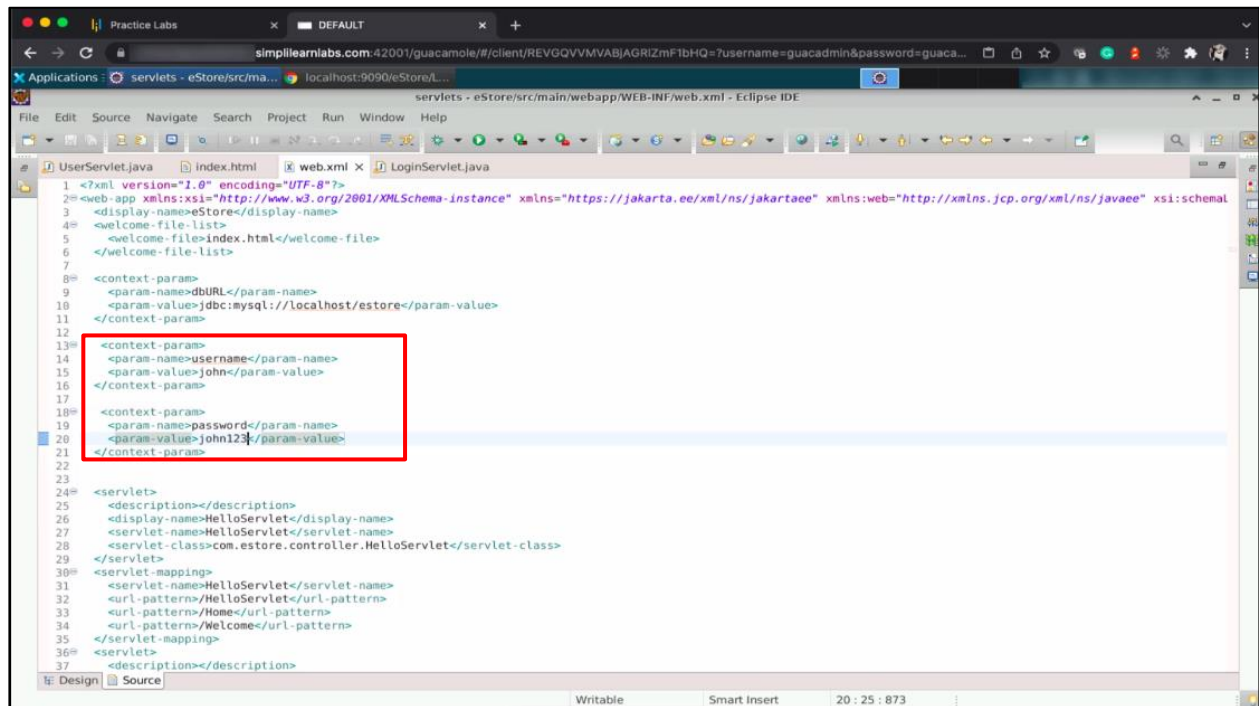


5.12 Click on LOGIN



You can see that the **dbURL** is accessible.

5.13 Go back to the **web.xml** file and create two more **context-params**

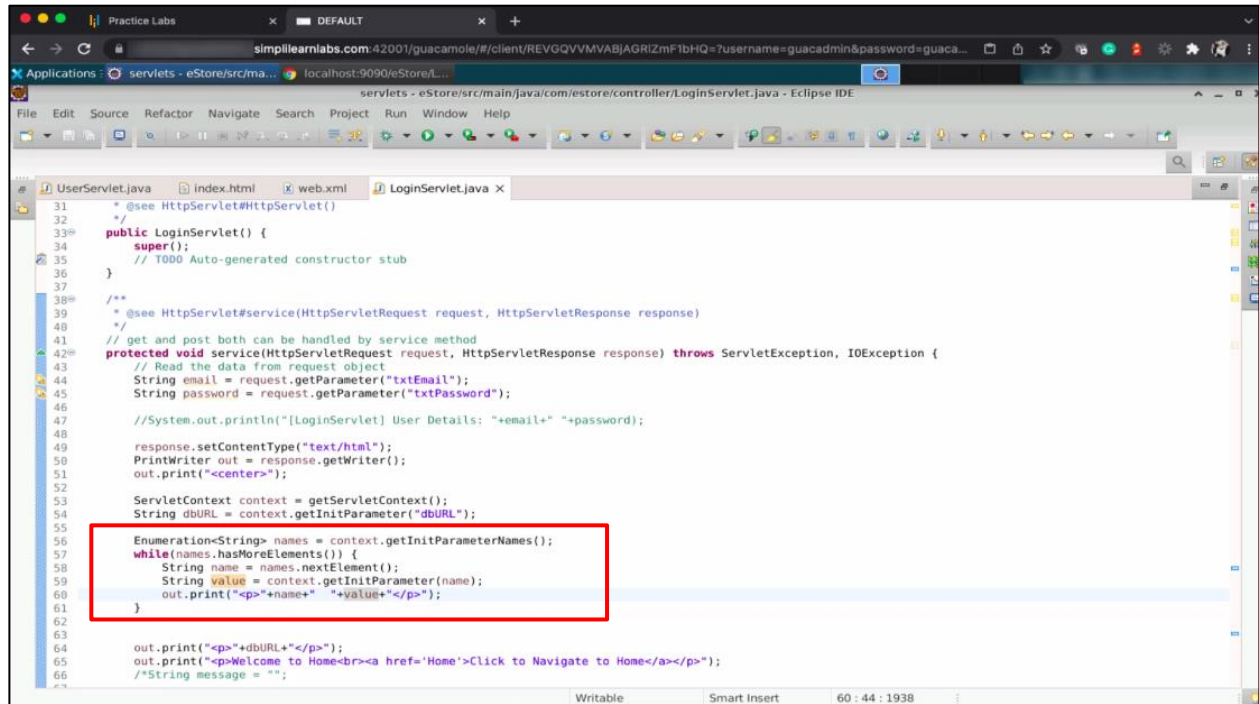


```

1 <?xml version="1.0" encoding="UTF-8"?>
2 <web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="https://jakarta.ee/xml/ns/jakartaee" xmlns:web="http://xmlns.jcp.org/xml/ns/javaee" xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/javaee web-app_3_1.xsd" >
3   <display-name>eStore</display-name>
4   <welcome-file-list>
5     <welcome-file>index.html</welcome-file>
6   </welcome-file-list>
7
8   <context-param>
9     <param-name>dbURL</param-name>
10    <param-value>jdbc:mysql://localhost/estore</param-value>
11  </context-param>
12
13  <context-param>
14    <param-name>username</param-name>
15    <param-value>john</param-value>
16  </context-param>
17
18  <context-param>
19    <param-name>password</param-name>
20    <param-value>john123</param-value>
21  </context-param>
22
23
24  <servlet>
25    <description></description>
26    <display-name>HelloServlet</display-name>
27    <servlet-name>HelloServlet</servlet-name>
28    <servlet-class>com.estore.controller.HelloServlet</servlet-class>
29  </servlet>
30  <servlet-mapping>
31    <servlet-name>HelloServlet</servlet-name>
32    <url-pattern>/HelloServlet</url-pattern>
33    <url-pattern>/Home</url-pattern>
34    <url-pattern>/Welcome</url-pattern>
35  </servlet-mapping>
36  <servlet>
37    <description></description>

```


5.14 Go back to **LoginServlet.java** and create a new method. Use **Enumeration** and **while** loop for reading all three **context-param** in one method



```

31  * @see HttpServlet#HttpServlet()
32  */
33  public LoginServlet() {
34      super();
35      // TODO Auto-generated constructor stub
36  }
37
38  /**
39   * @see HttpServlet#service(HttpServletRequest request, HttpServletResponse response)
40   */
41  // get and post both can be handled by service method
42  protected void service(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
43      // Read the data from request object
44      String email = request.getParameter("txtEmail");
45      String password = request.getParameter("txtPassword");
46
47      //System.out.println("[LoginServlet] User Details: "+email+" "+password);
48
49      response.setContentType("text/html");
50      PrintWriter out = response.getWriter();
51      out.print("<center>");
52
53      ServletContext context = getServletContext();
54      String dbURL = context.getInitParameter("dbURL");
55
56      Enumeration<String> names = context.getInitParameterNames();
57      while(names.hasMoreElements()) {
58          String name = names.nextElement();
59          String value = context.getInitParameter(name);
60          out.print("<p>"+name+" "+value+"</p>");
61      }
62
63      out.print("<p>"+dbURL+"</p>");
64      out.print("<p>Welcome to Home<br><a href='Home'>Click to Navigate to Home</a></p>");
65      //String message = "";
66

```

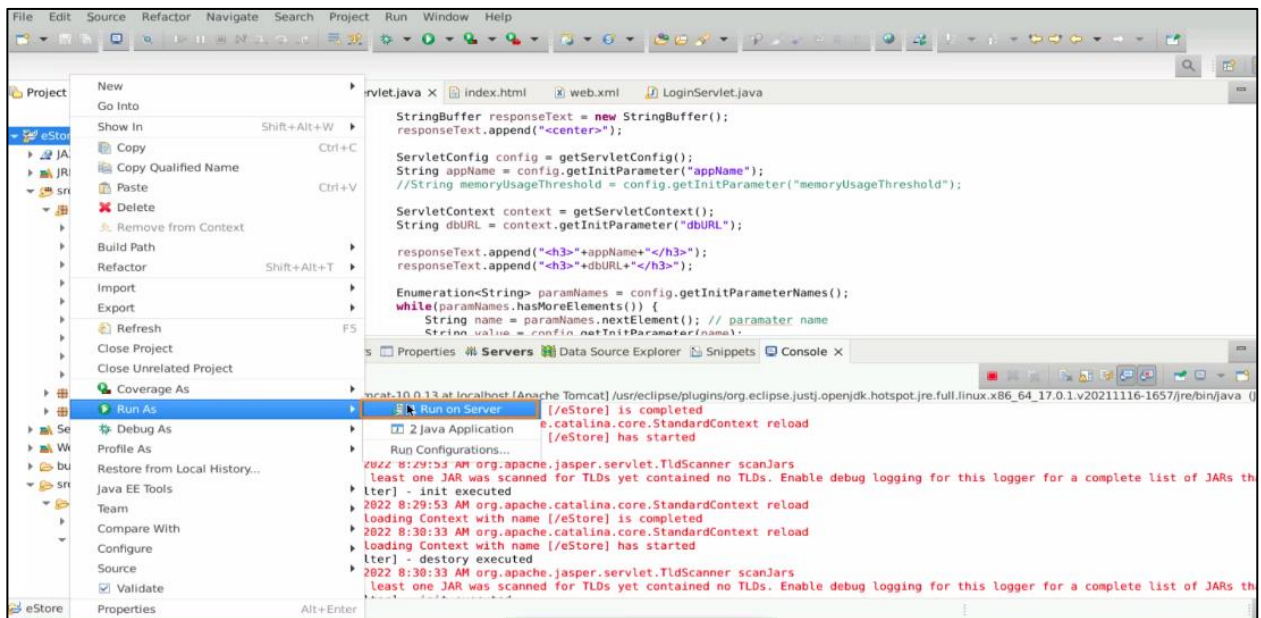
5.15 Go back to the **UserServlet.java** file and add the servlet context parameters to read the same content as **LoginServlet**

```

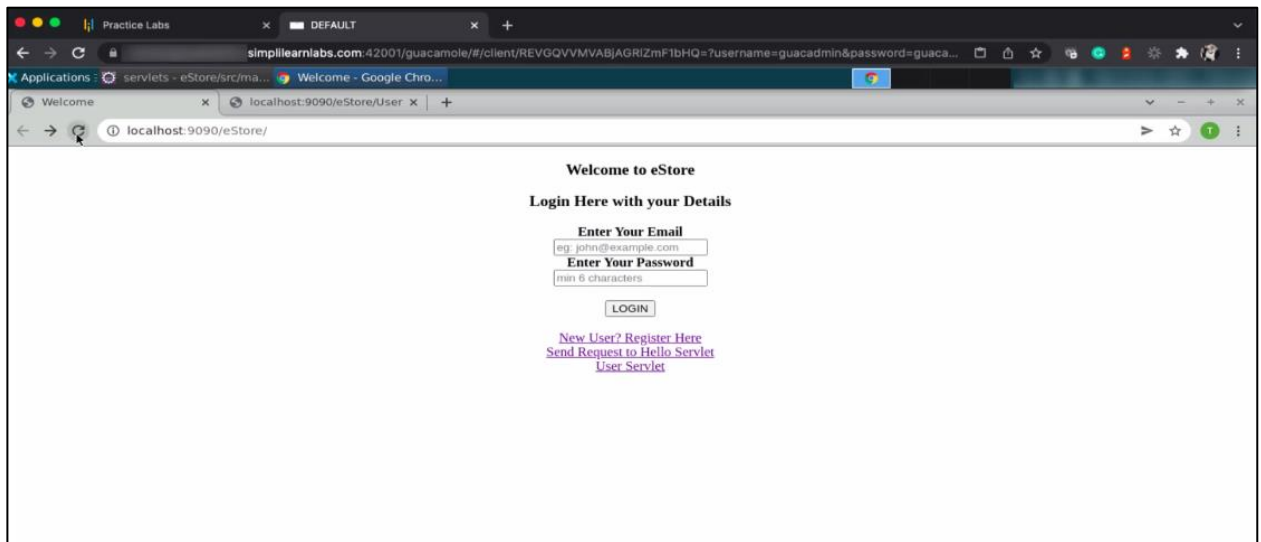
23  StringBuffer responseText = new StringBuffer();
24  responseText.append("<center>");
25
26  ServletConfig config = getServletConfig();
27  String appName = config.getInitParameter("appName");
28  //String memoryUsageThreshold = config.getInitParameter("memoryUsageThreshold");
29
30  ServletContext context = getServletContext();
31  String dbURL = context.getInitParameter("dbURL");
32
33  responseText.append("<h3>"+appName+"</h3>");
34  responseText.append("<h3>"+dbURL+"</h3>");
35
36  Enumeration<String> paramNames = config.getInitParameterNames();
37  while(paramNames.hasMoreElements()) {
38      String name = paramNames.nextElement(); // parameter name
39      String value = config.getInitParameter(name);
40      responseText.append("<p>"+name+" "+value+"</p>");
41  }
42
43  responseText.append("<h3>Servlet Context Parameters</h3>");
44  Enumeration<String> names = context.getInitParameterNames();
45  while(names.hasMoreElements()) {
46      String name = names.nextElement();
47      String value = context.getInitParameter(name);
48      responseText.append("<p>"+name+" "+value+"</p>");
49  }
50
51
52  responseText.append("</center>");
53
54  out.print(responseText.toString());
55
56
57  }
58

```

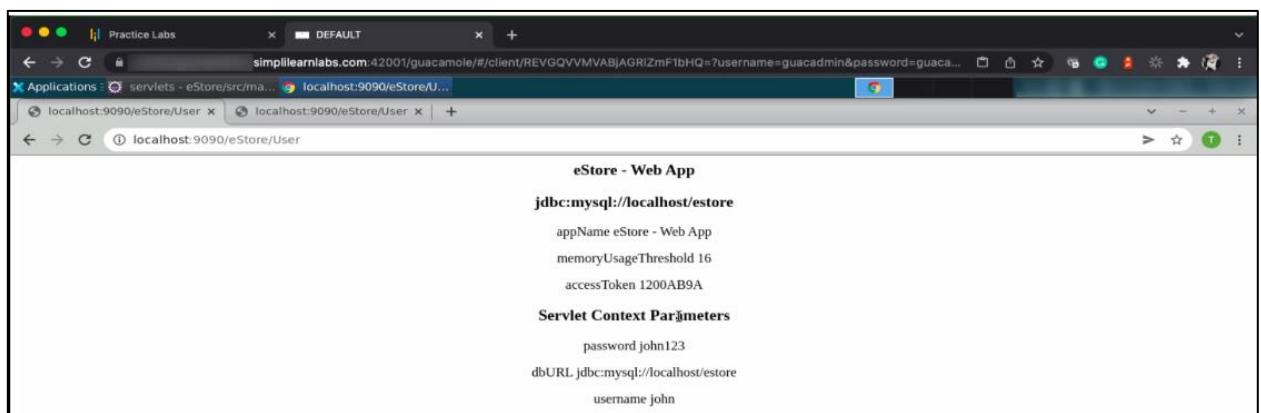
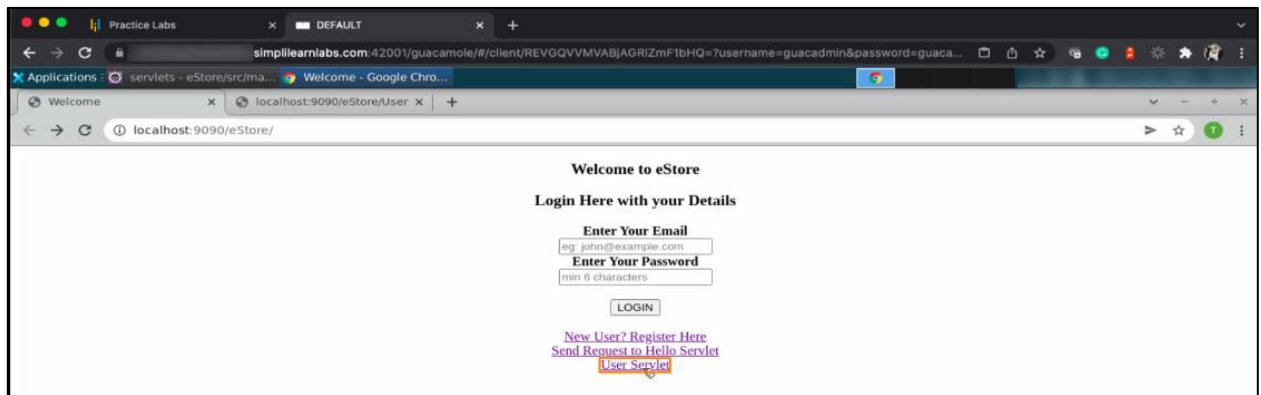
5.16 Run the server again by right-clicking on the project, selecting **Run As**, and then selecting **Run on Server**



5.17 Go back to the browser and refresh the screen

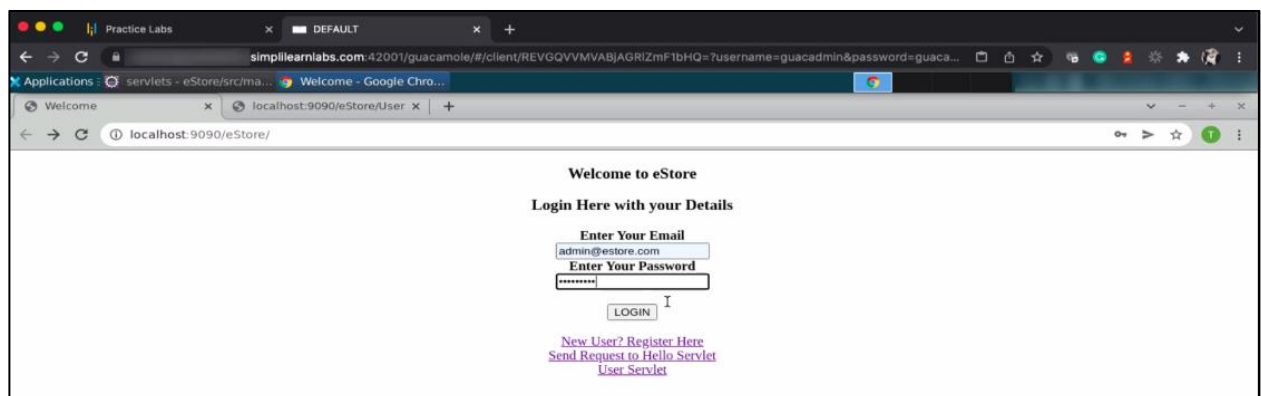


5.18 Click on the link **User servlet**

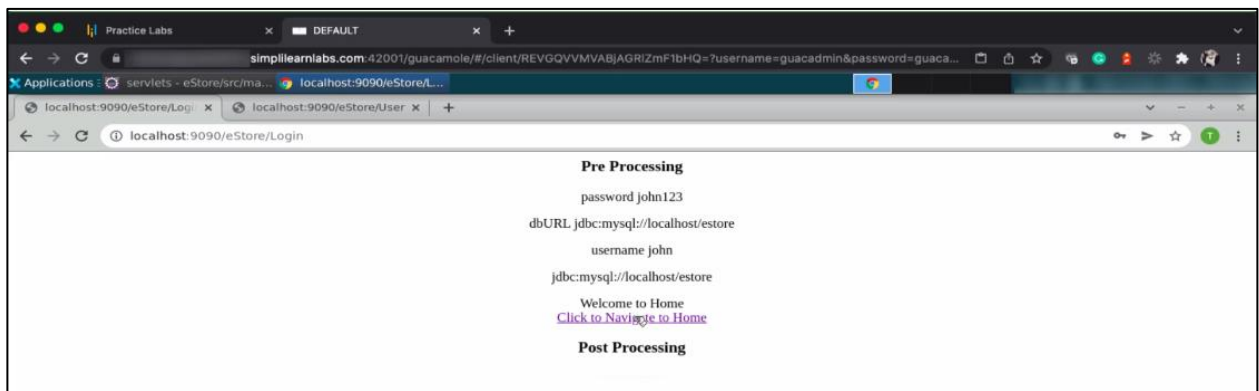
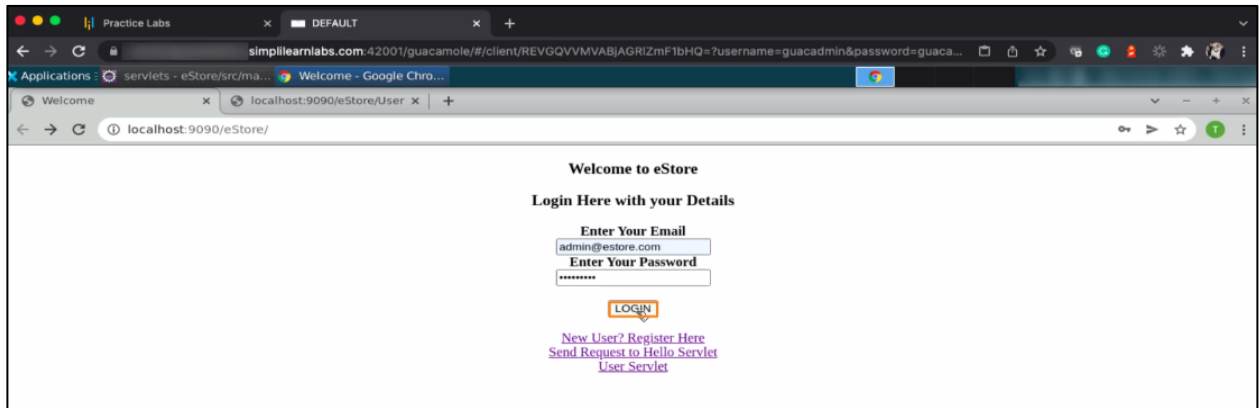


You can see all **context-params** and **init-params** in the output.

5.19 Go back to the browser and enter the **email** and **password**



5.20 Click on the **LOGIN** button



Observation: The login servlet reads all the context parameters.

Following these steps, you have successfully configured a servlet individually and collectively.