Lesson 01 Demo 03

Service doGet() and doPost() in Servlet

Objective: To explore the doGet() and doPost() services in Servlet

Tools Required: Eclipse IDE

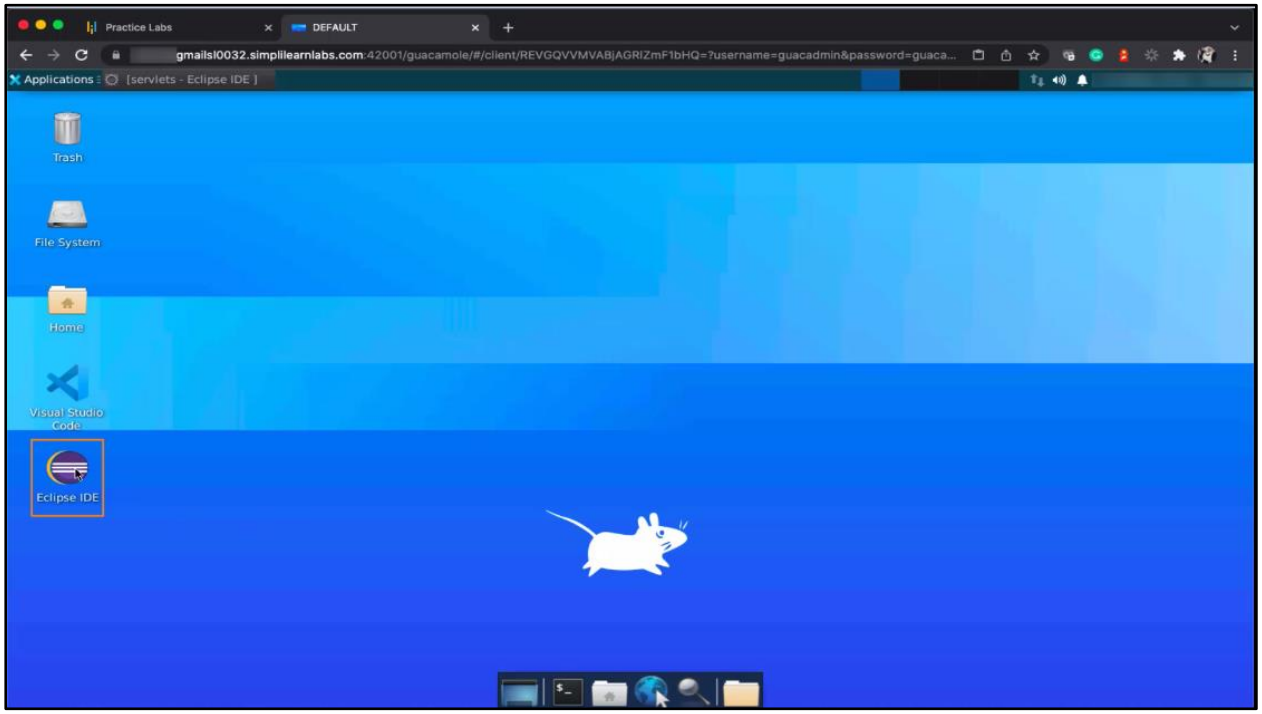
Prerequisites: None

Steps to be followed:

1. Perform the HTTP POST method
2. Perform the HTTP GET method
3. Create a doGet() method
4. Create a doPost() method

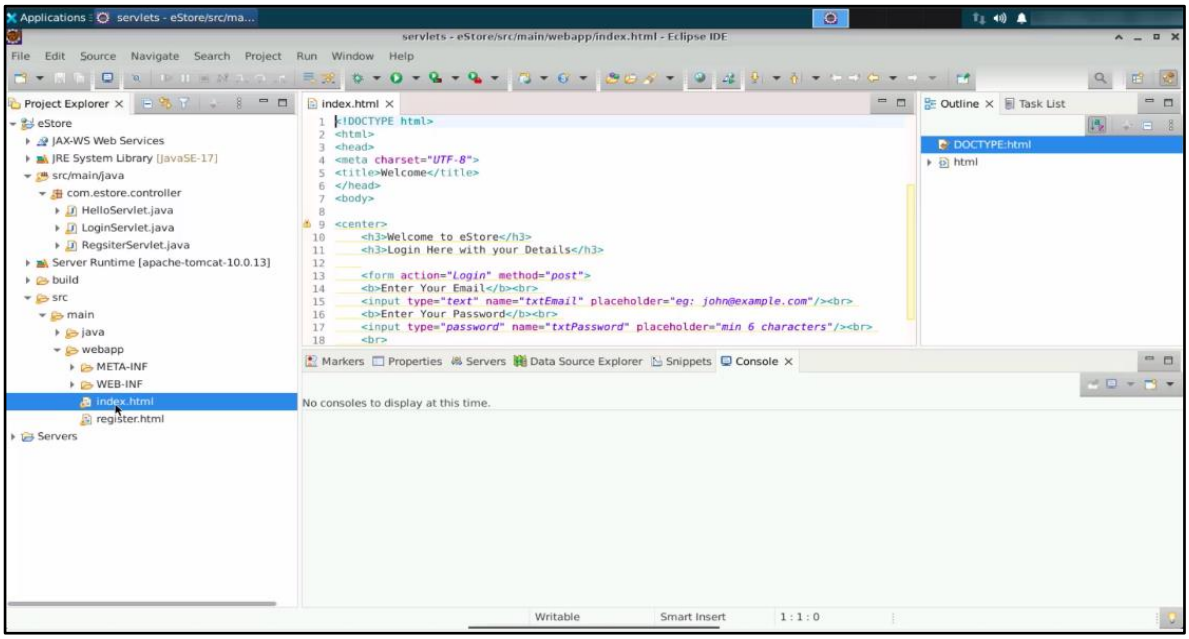
Step 1: Perform the HTTP POST method

1.1 Open **Eclipse IDE**

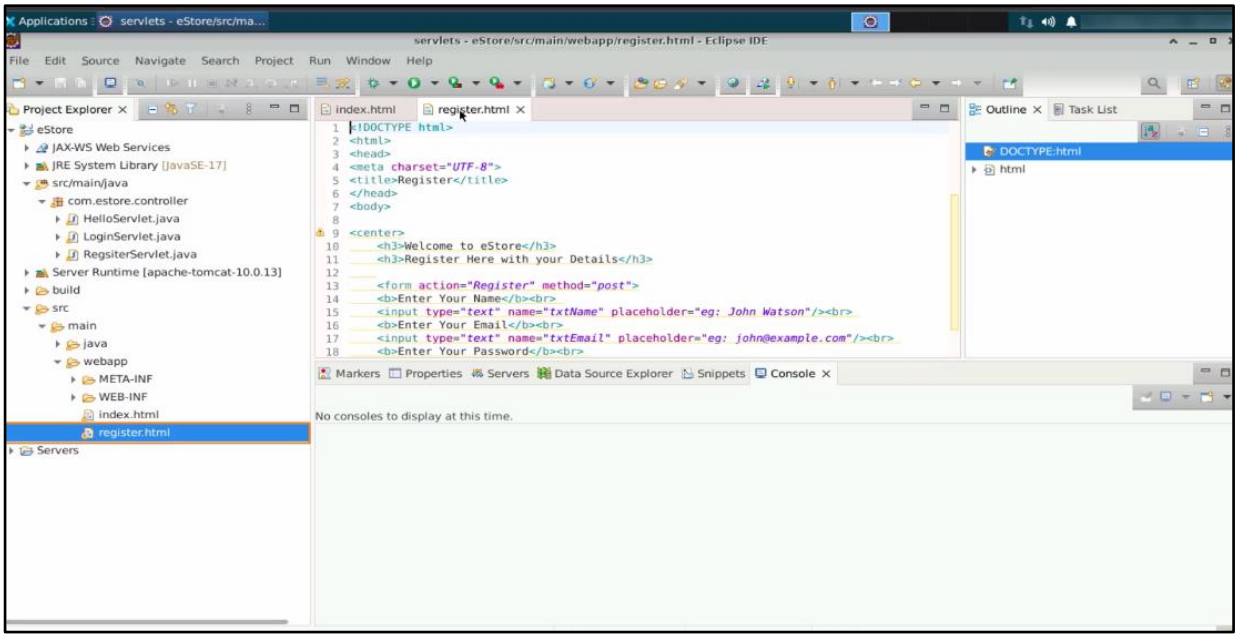


1.2 Open the **index.html** page inside the **webapp** folder.

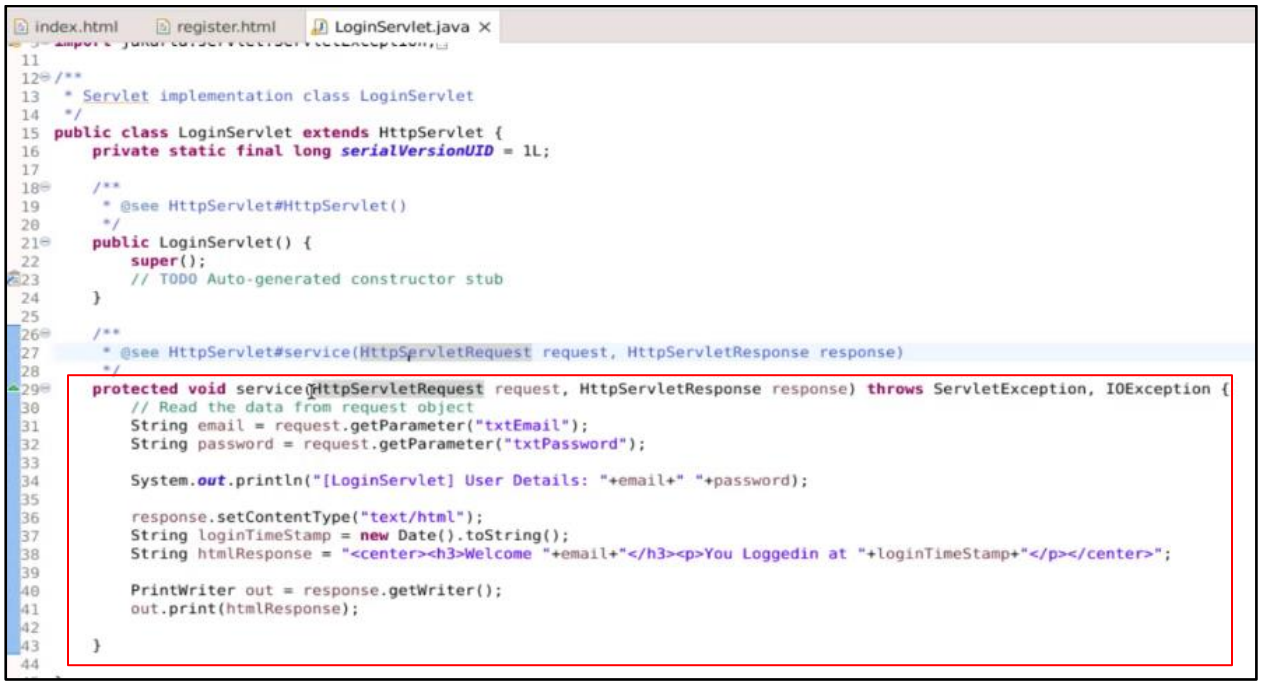
Note: eStore is a dynamic web project that has already been created in **Demo 02**.



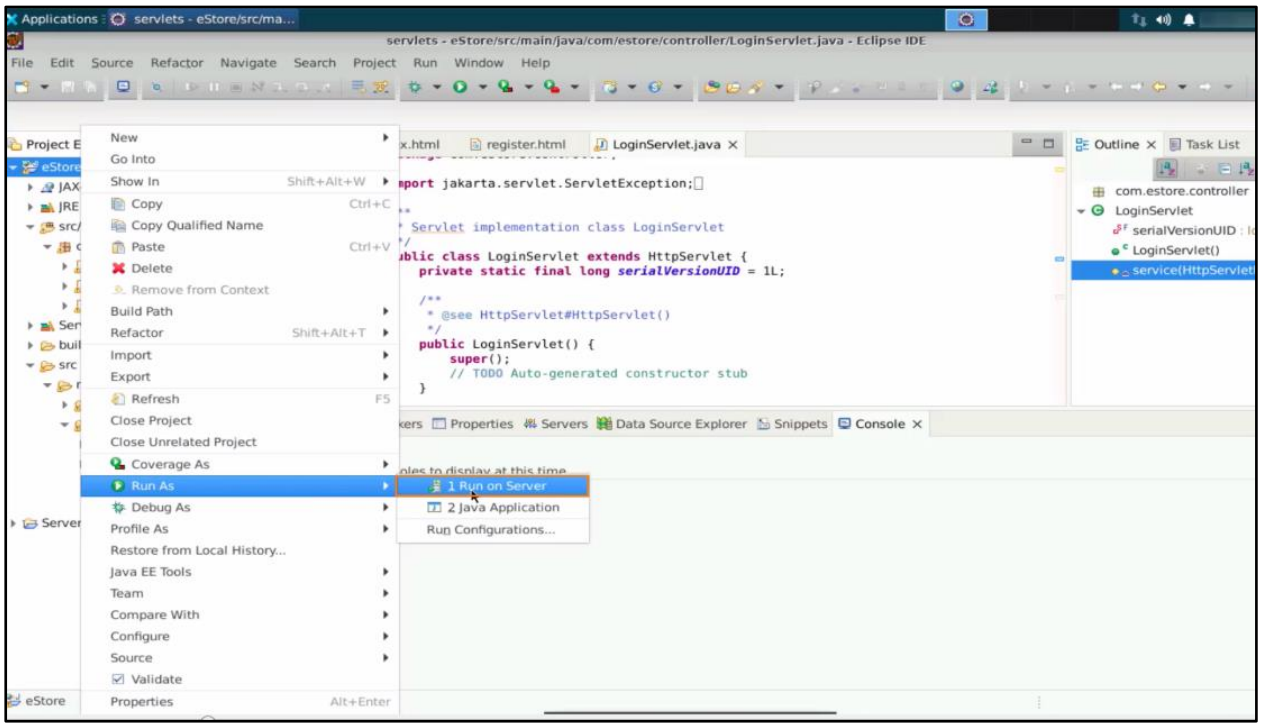
1.3 Open the **register.html** file and find a registration form that takes the register action**.** It is like a Servlet which sends an email and the password.



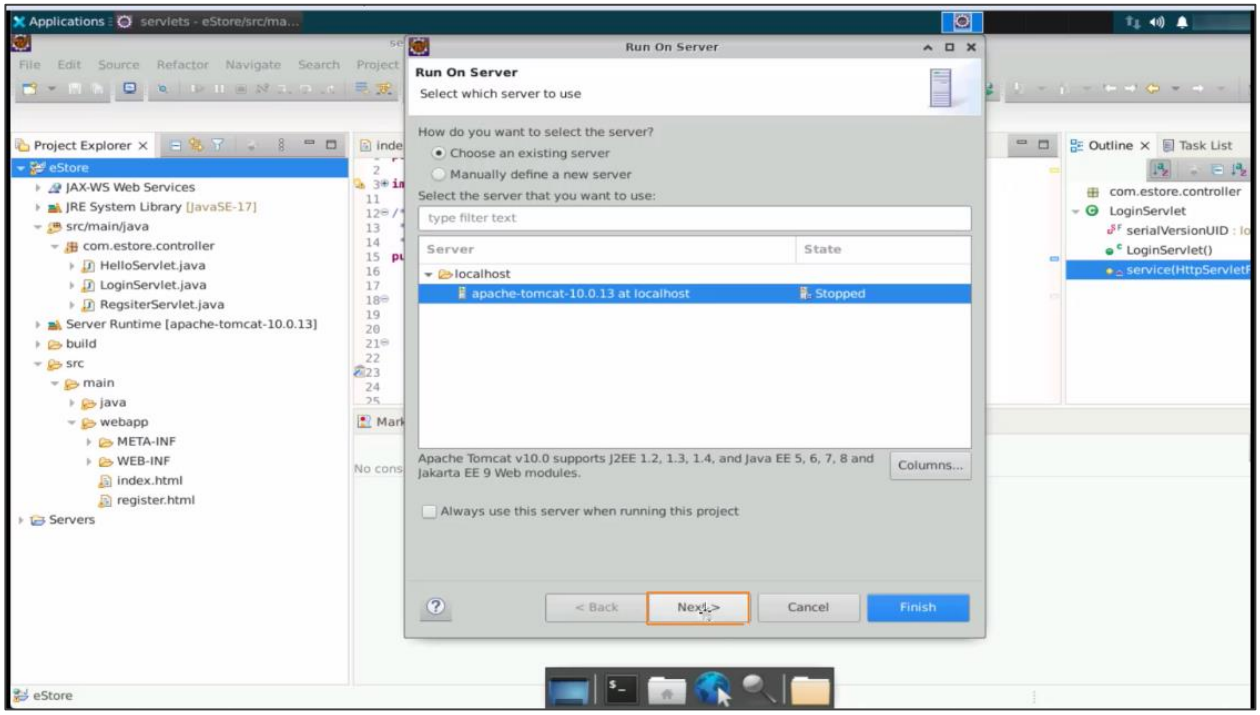
1.4 Now, open the **LoginServlet.java** file and search for the **service** method. A Service method can handle **GET** and **POST** requests.



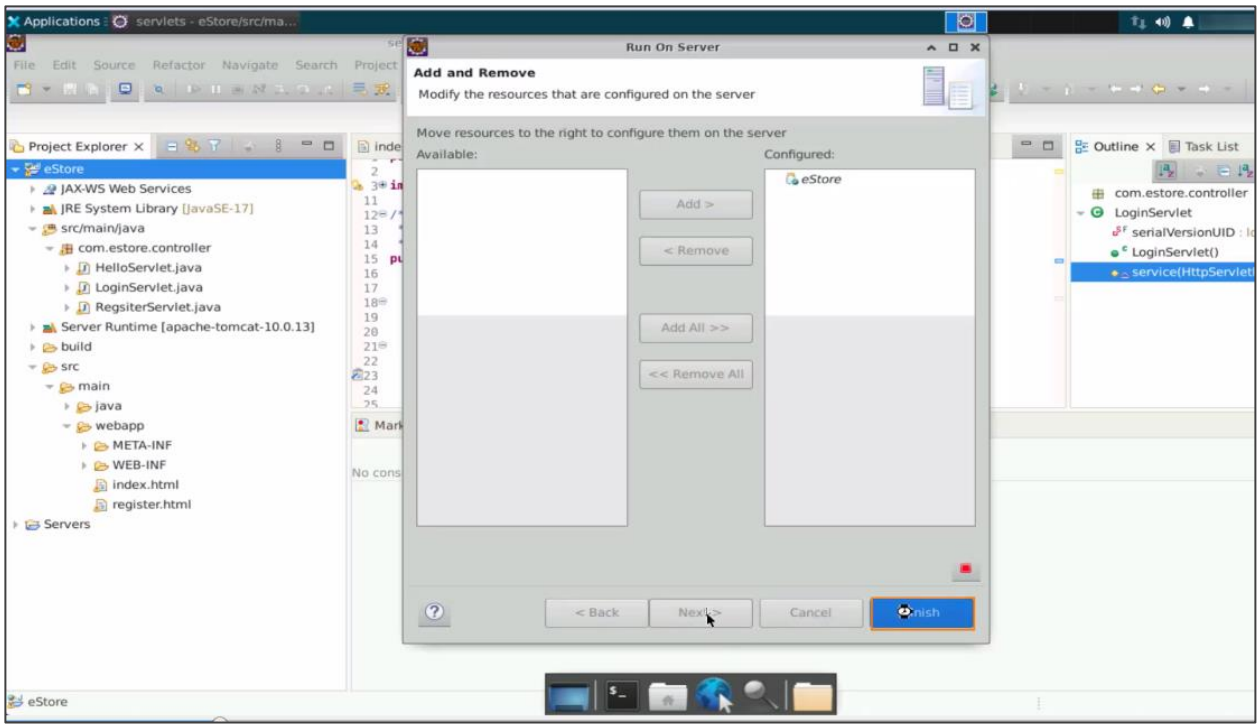
1.5 Run the **estore** project. Right-click on **eStore**, select **Run As,** and click on **Run on Server**



1.6 Choose the server **apache-tomcat-10.0.13** and click **Next**

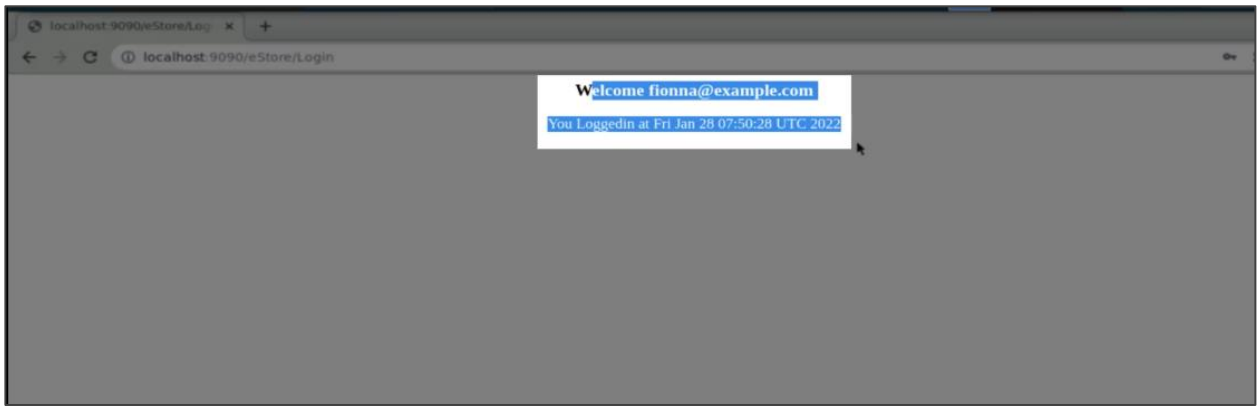


1.7 Click **Finish**



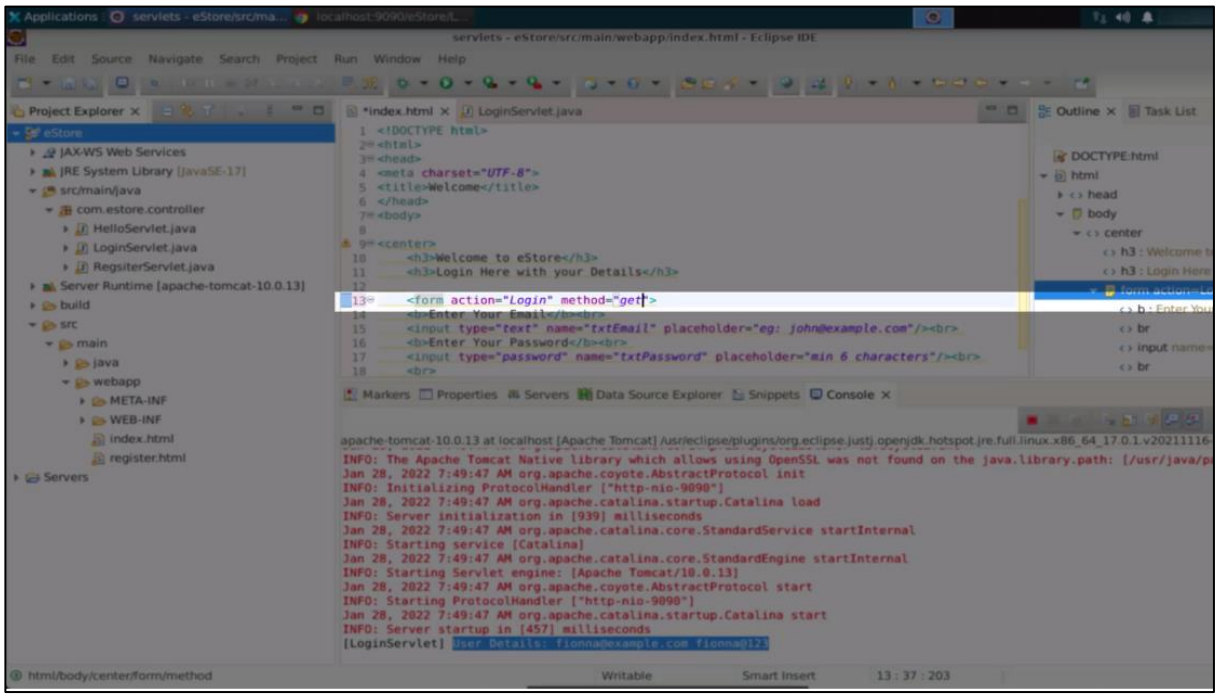
1.8 The login page would appear in the browser. Enter the information as required and click the **LOGIN** button

The below data is being shown, which says **Welcome fionna@example.com.**

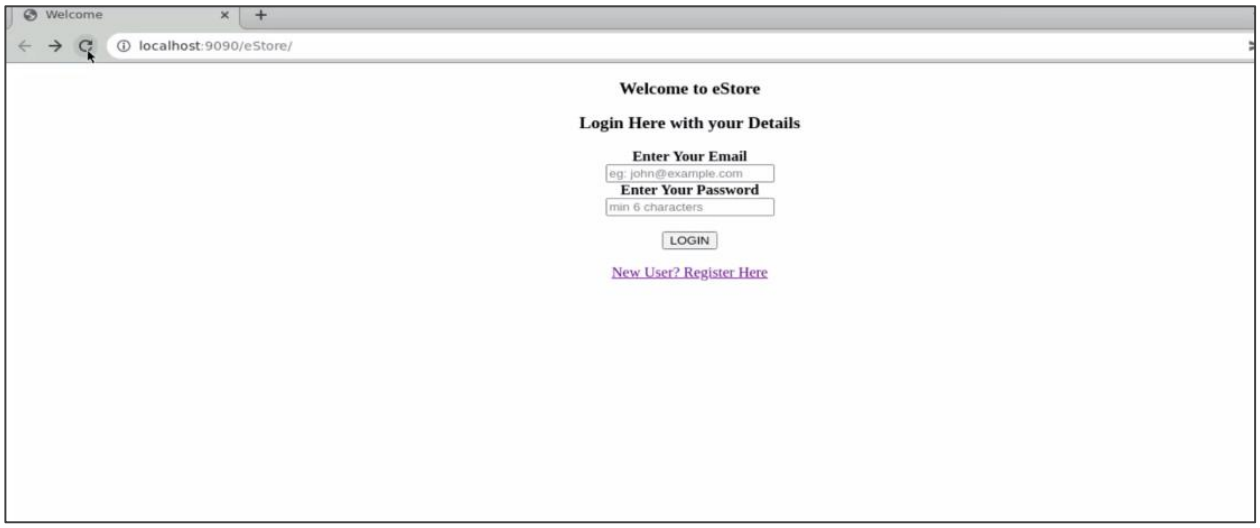


Step 2: Performing the HTTP GET method

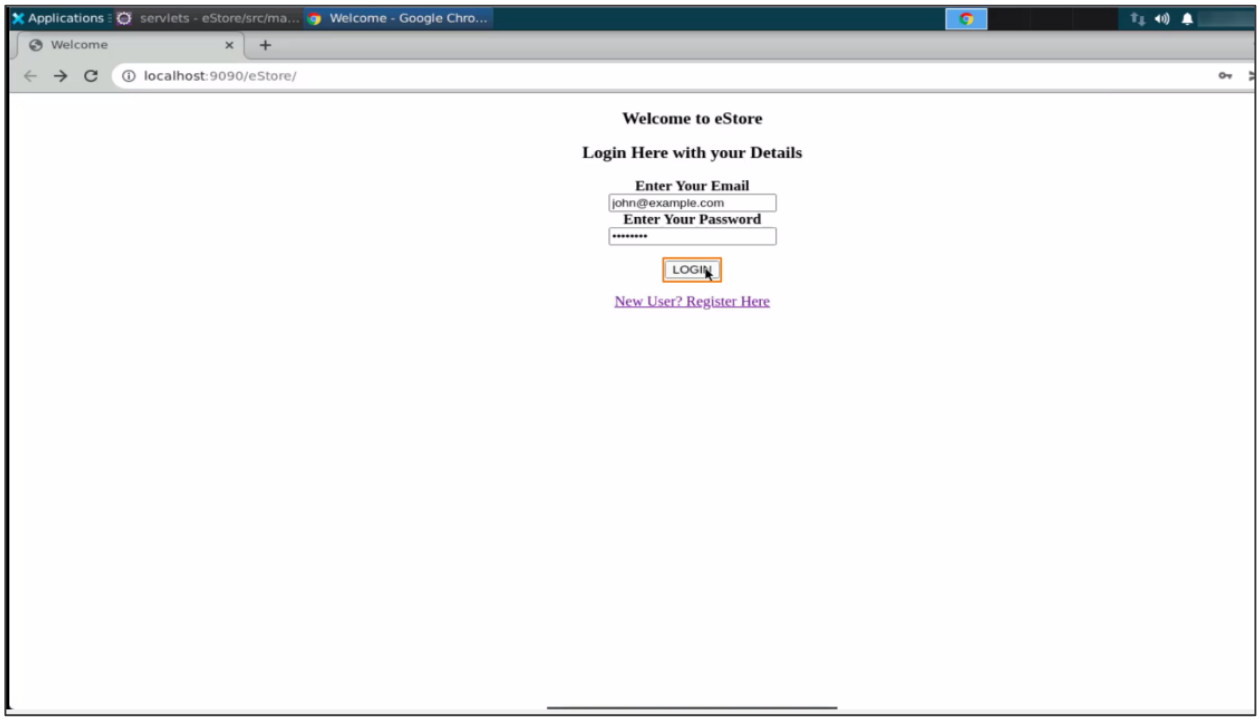
2.1 Go to the **index.html** page, change the method to **get,** and save it



2.2 Again, go to the browser and refresh the page

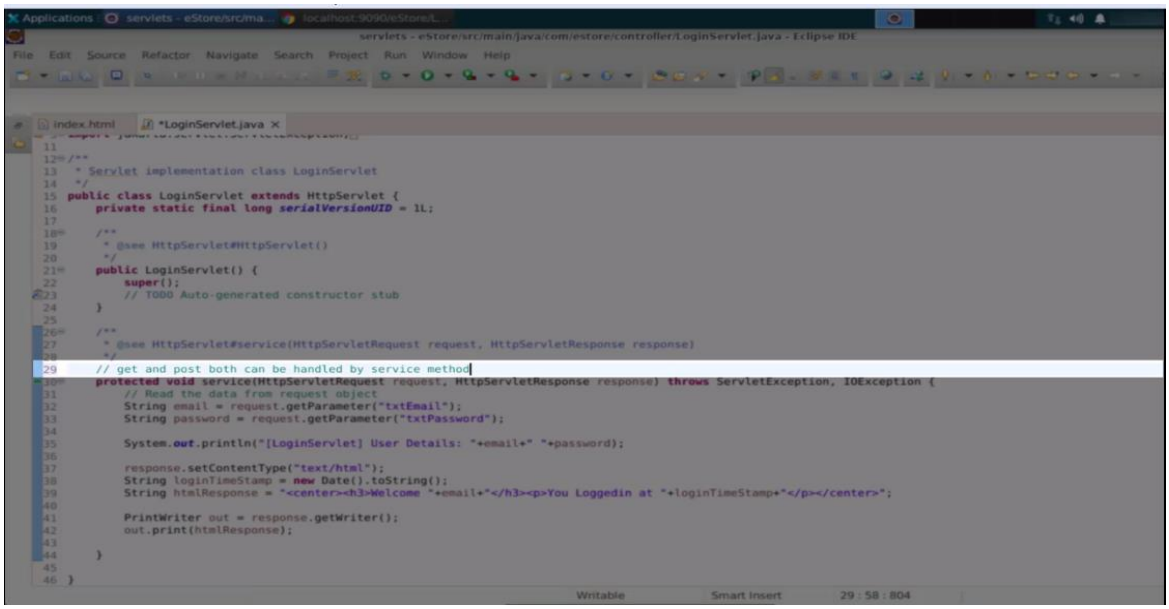


2.3 Enter the username as **john@example.com**, the password as **john@123**, and click **LOGIN**



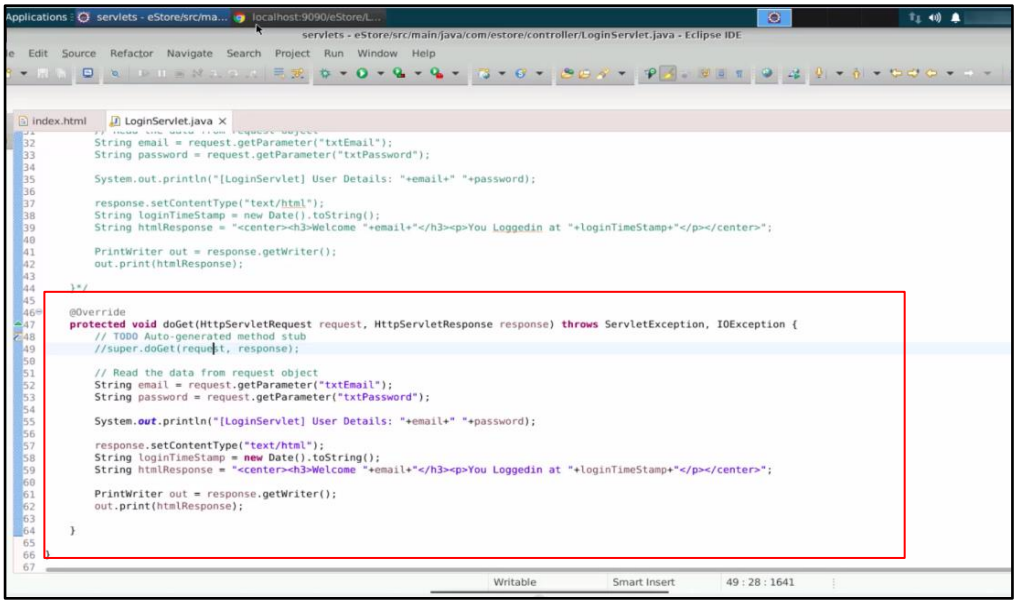
We will now be able to view the URL data. Thus, a GET request is created, and the data becomes part of the URL. It is not recommended to use this for sensitive information.

2.4 To handle both the **GET** and the **POST** requests from the client, go to the **LoginServlet.java** page and add a comment for them

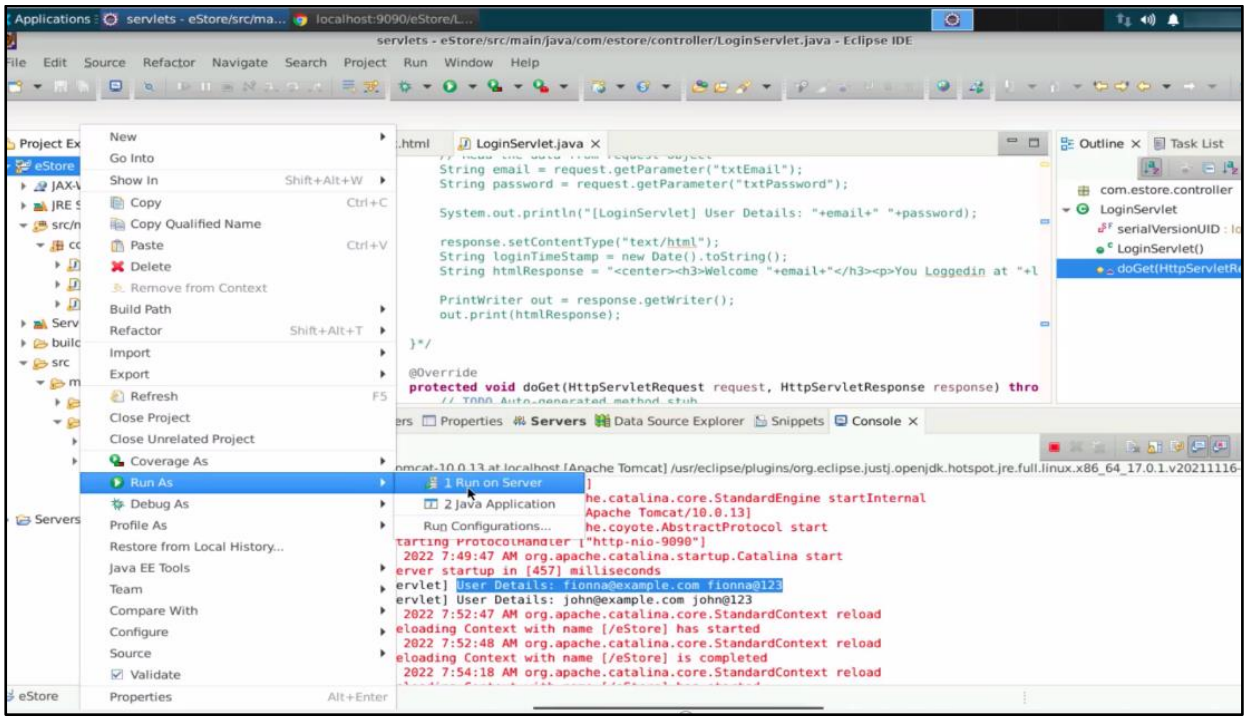


Note: As the service method does not allow the addition of restrictions, users need to comment on the method to add restrictions.

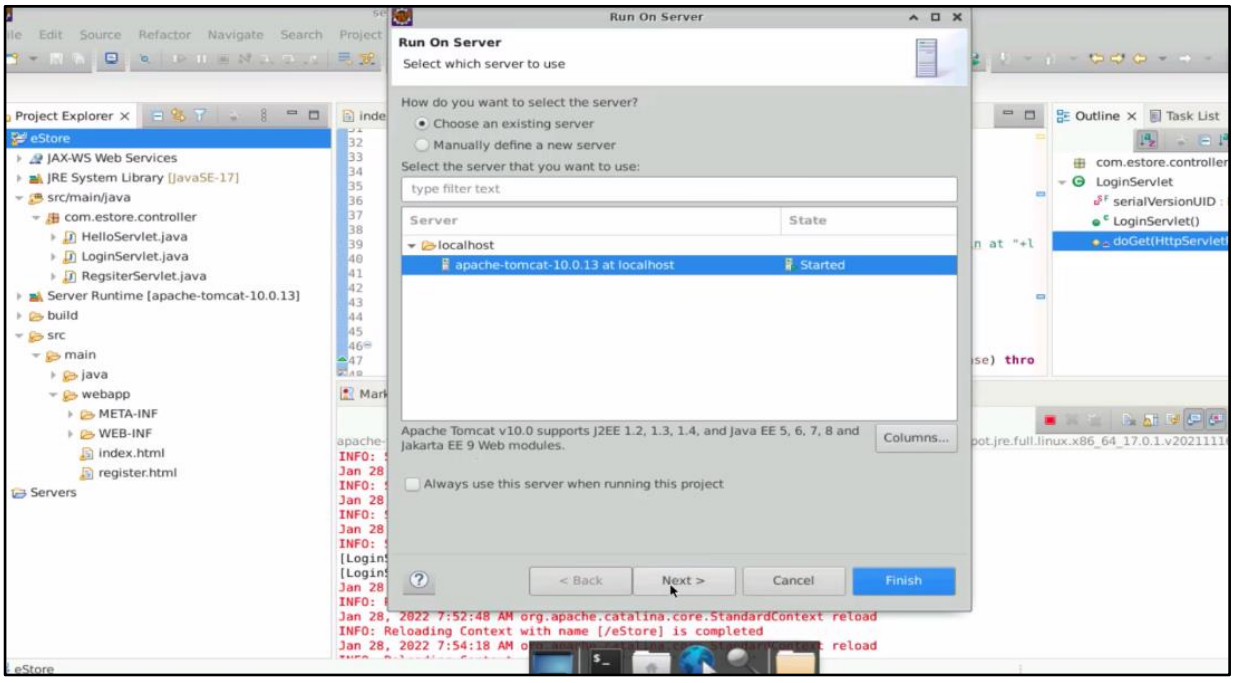
2.5 Replace the service method with doGet() and copy the complete doGet() code



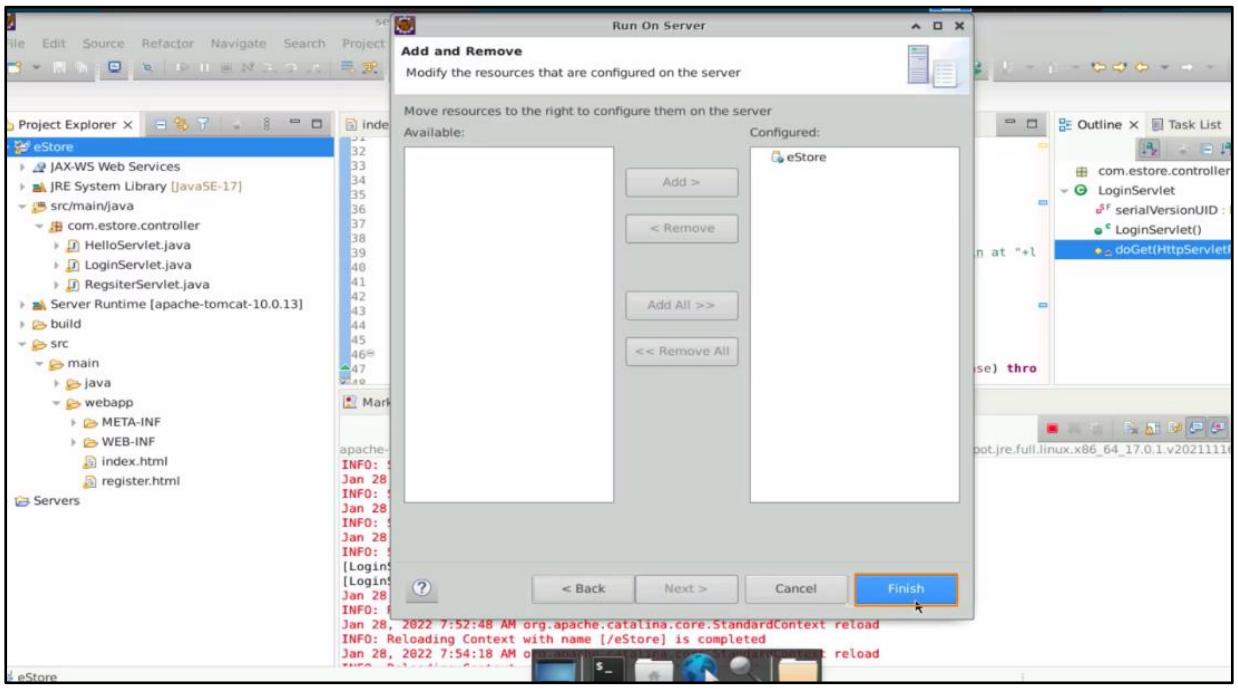
2.6 Since the server is modified, recompile the entire project. Right-click on **eStore**, select **Run As,** and click **Run on Server**



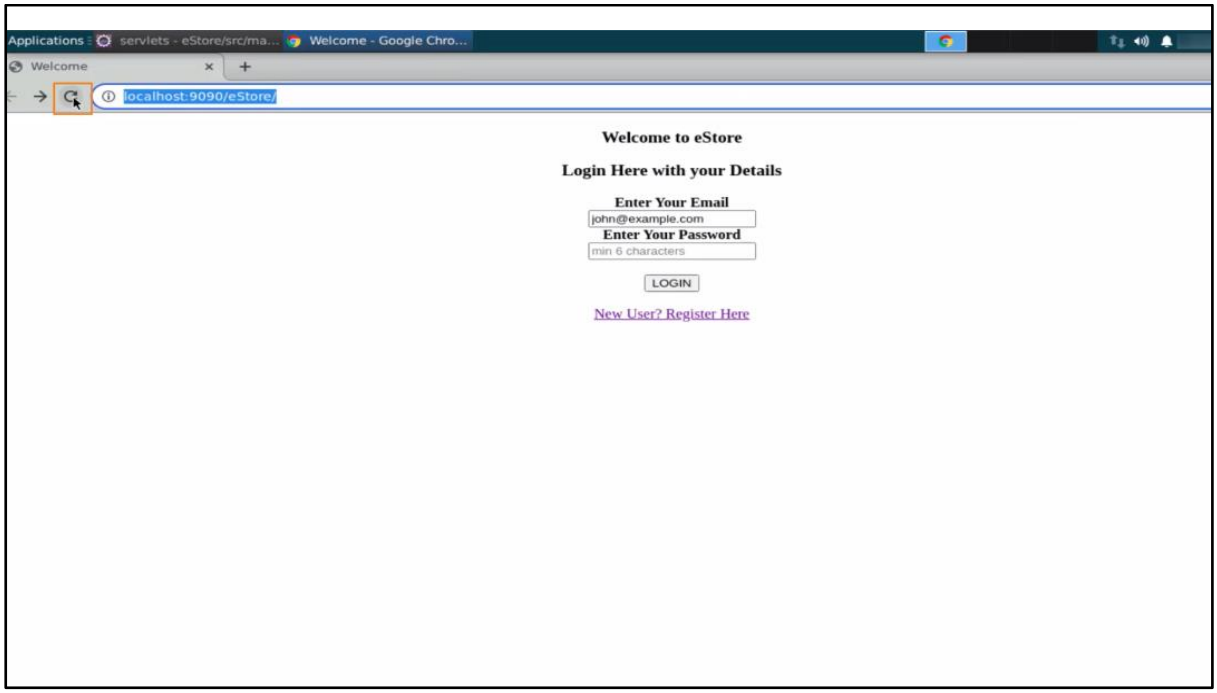
2.7 Select the tomcat server as **apache-tomcat-10.0.13** and click **Next**



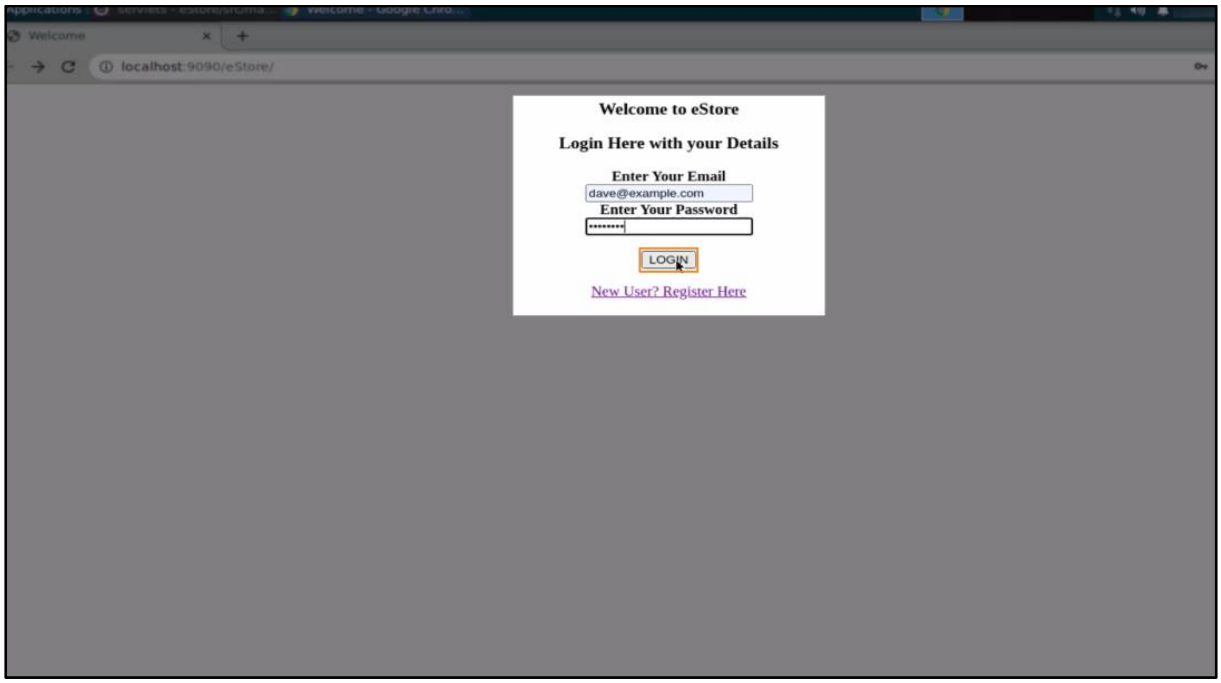
2.8 Click **Finish**



2.9 Refresh the browser



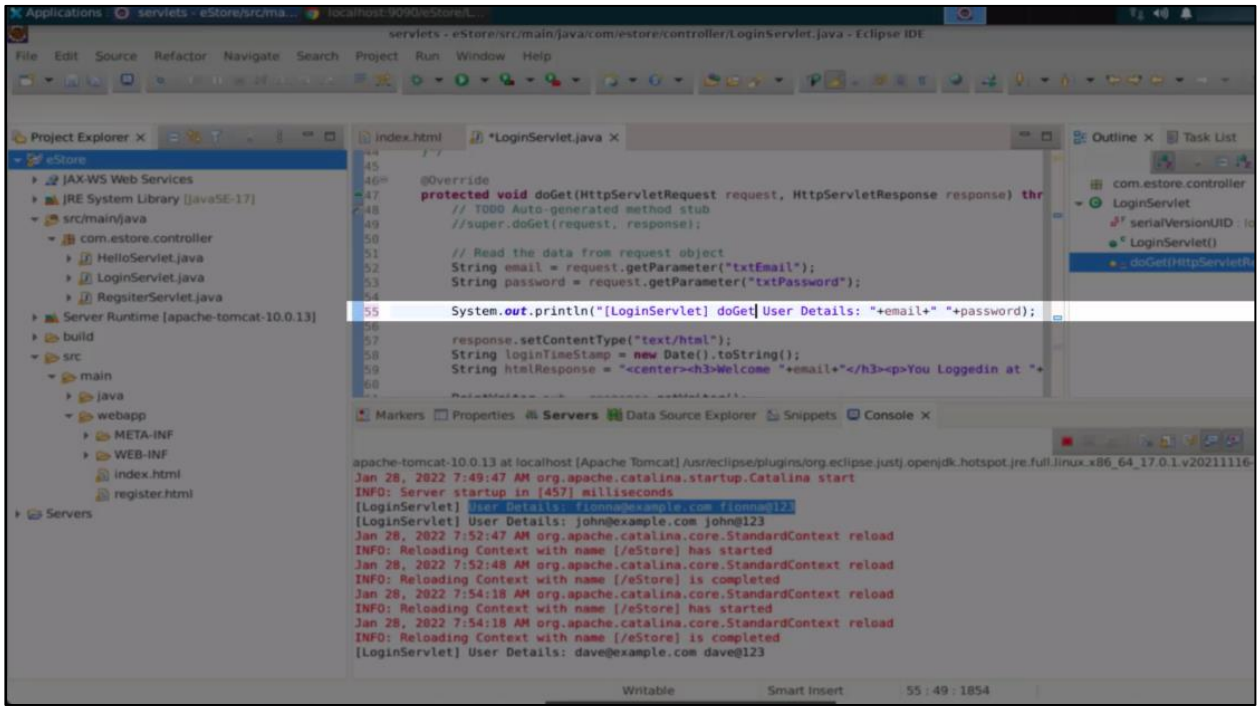
2.10 Enter the username **dave@example.com** and the password **dave@123**, and click on **LOGIN**



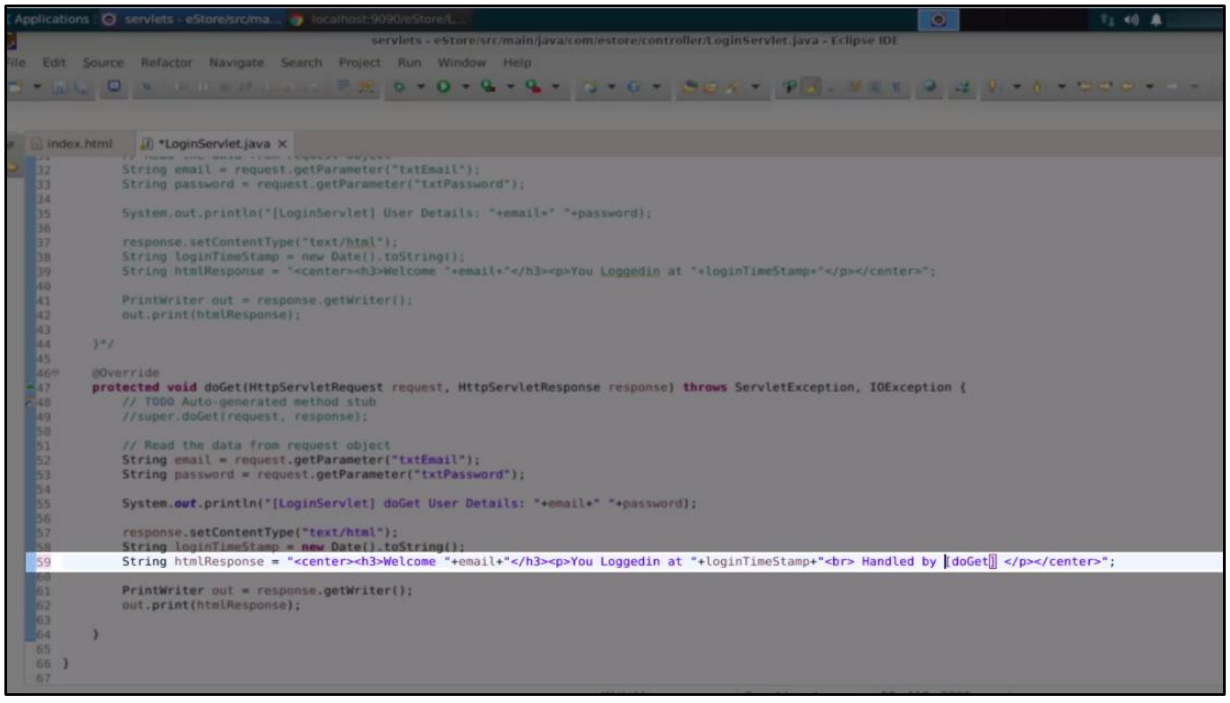
The response displayed will be as shown in the screenshot:

Step 3: Creating a doGet() method

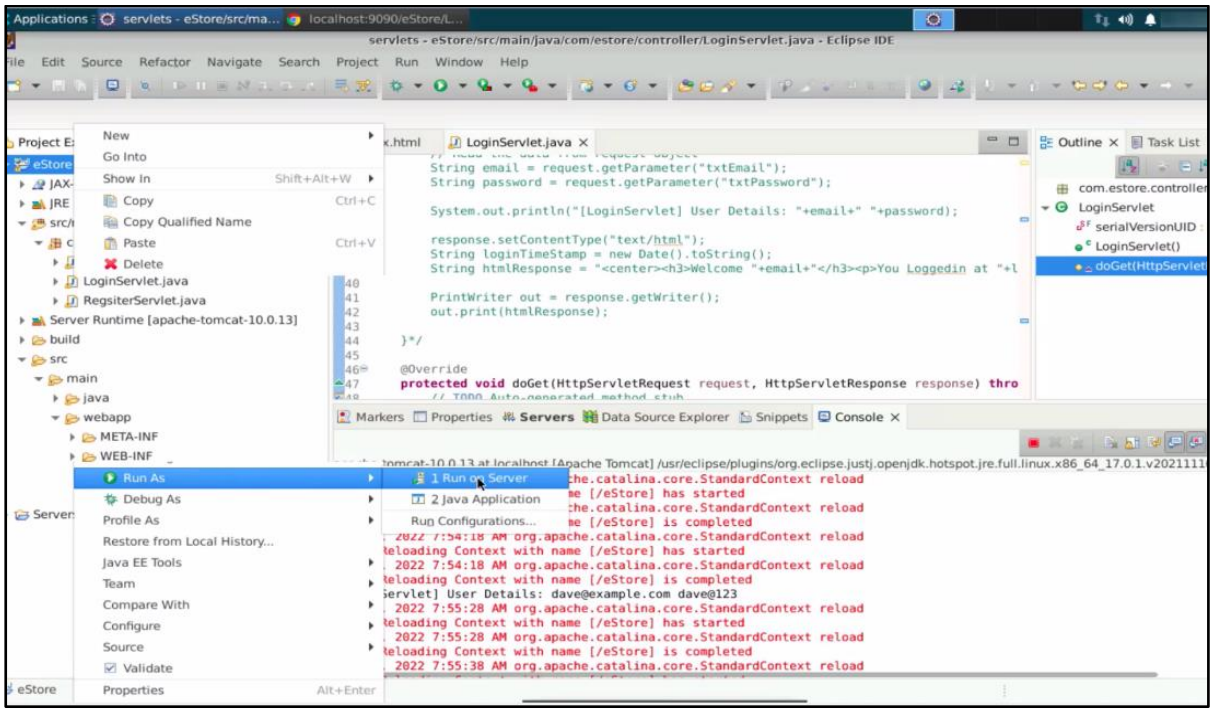
3.1 To manipulate the response, go to the **LoginServlet.java** file and write the **doGet()** method



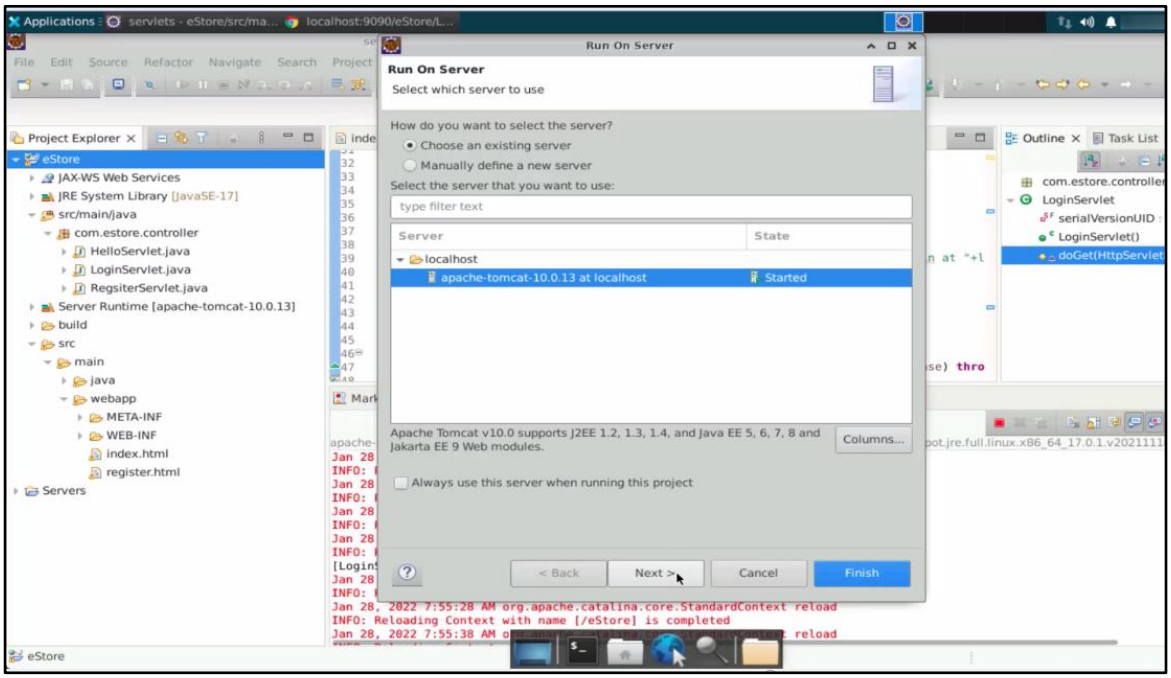
3.2 Add the **doGet()** parameter in the HTML response as **Handled by [doGet]** and save it



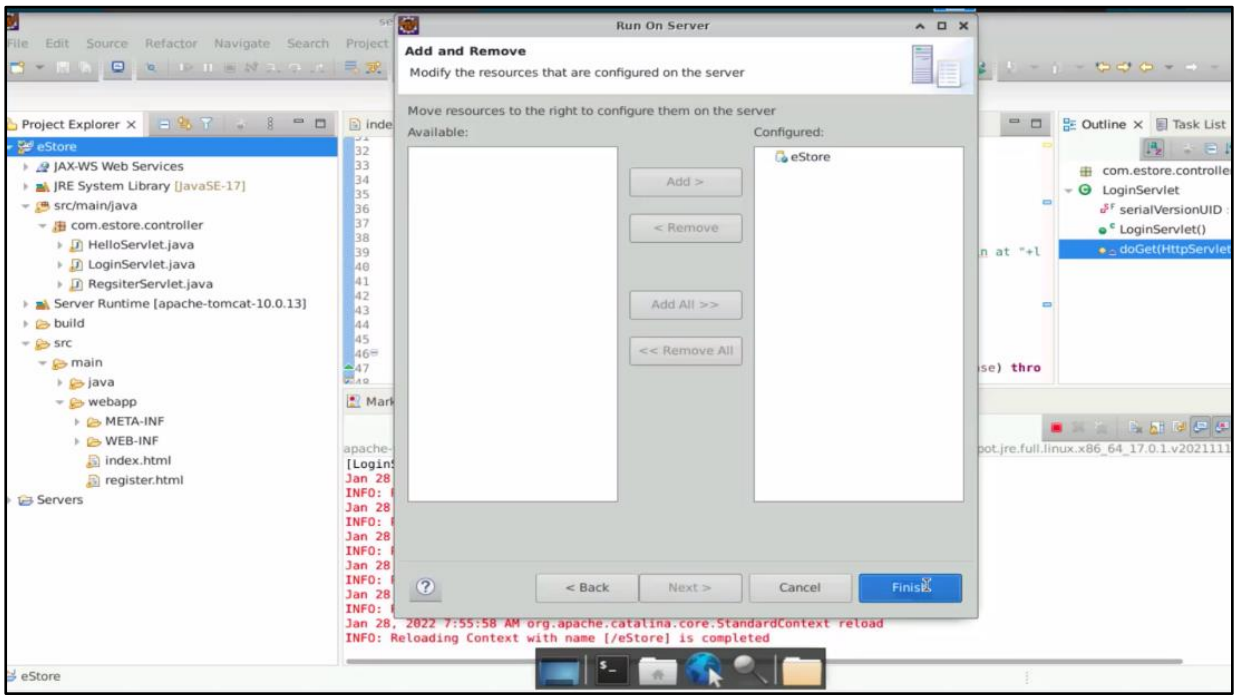
3.3 Re-run the project on the server by right-clicking on **eStore**, selecting **Run As**, and then selecting **Run on Server**



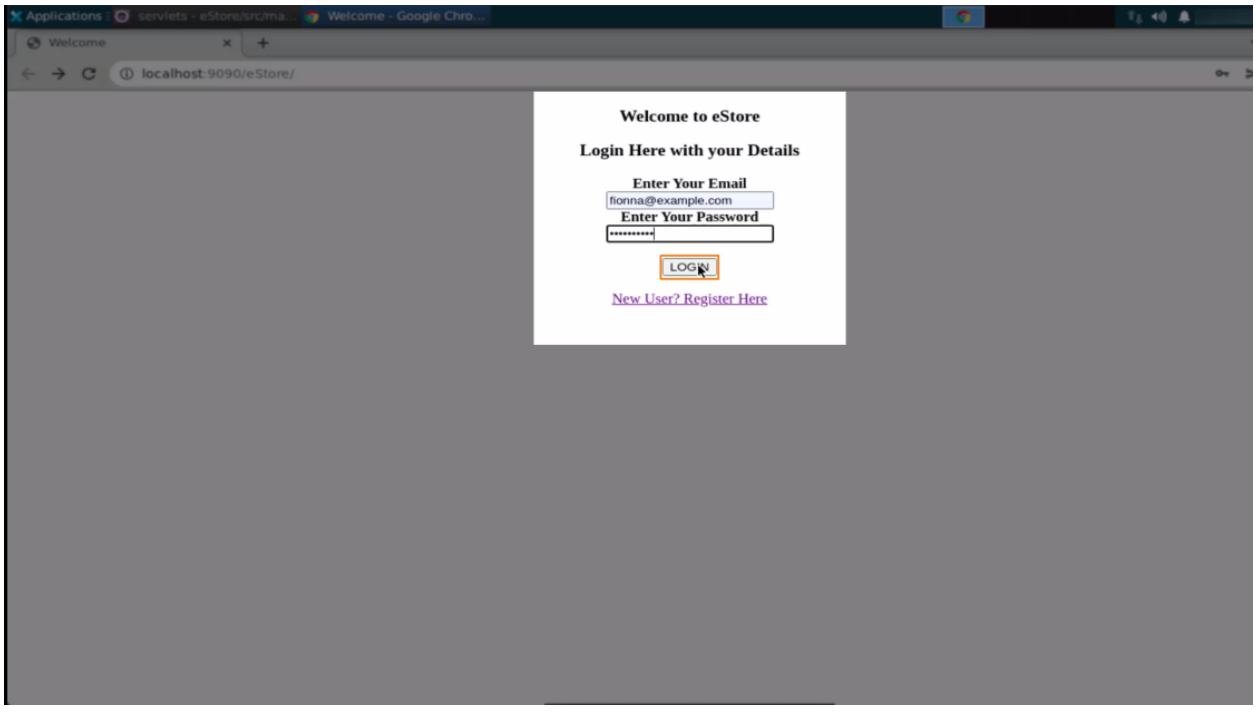
3.4 Select Apache Tomcat as **apache-tomcat-10.0.13 at localhost** and click **Next**



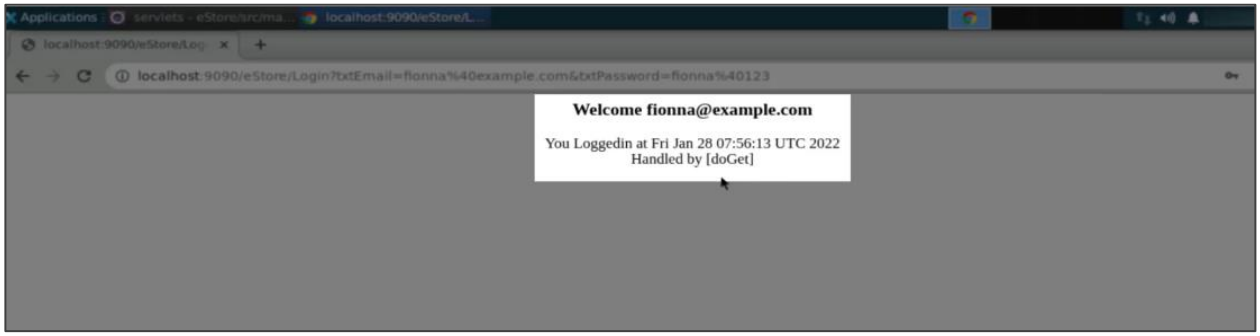
3.5 Click **Finish**



3.6 Refresh the browser, enter the username and password, and click **LOGIN**

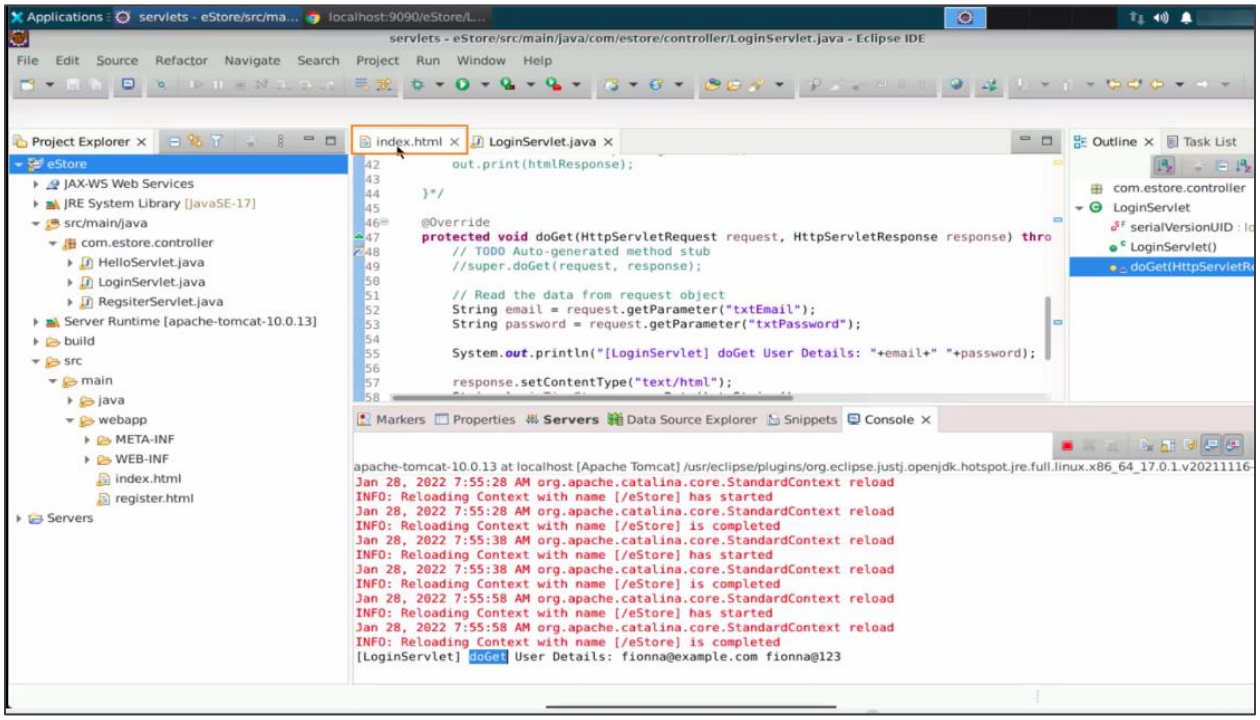


The output appears as shown below:

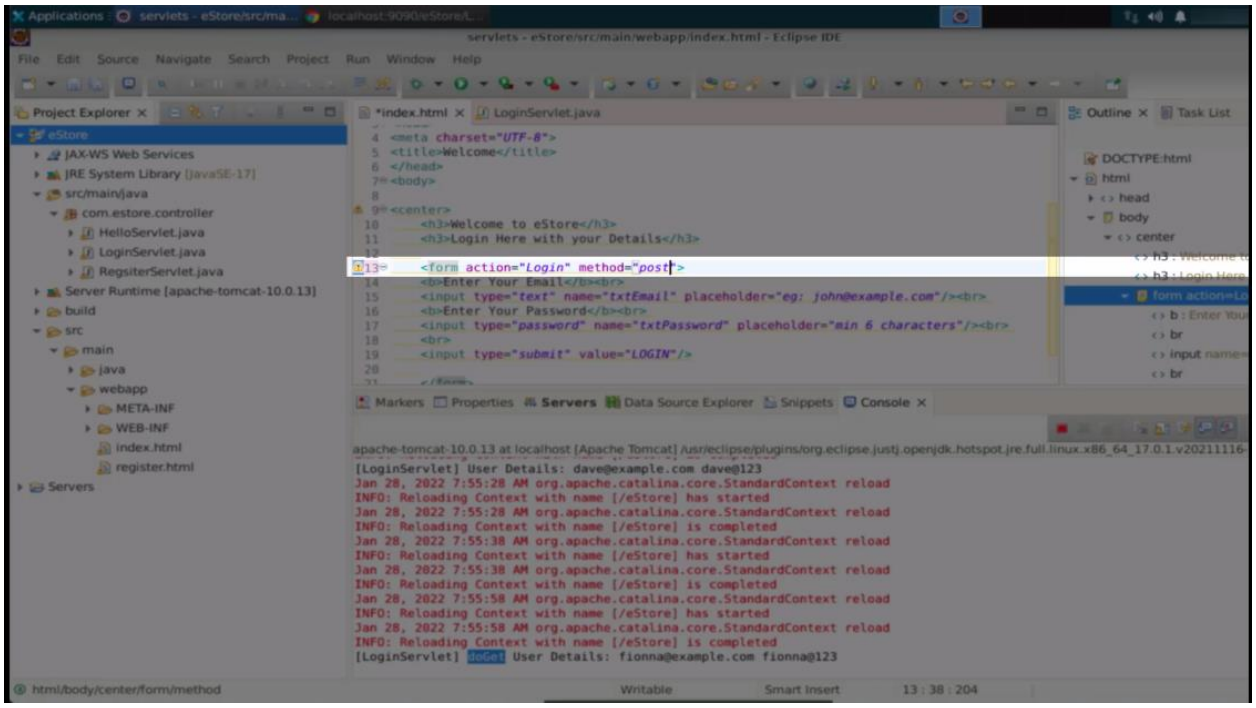


Step 4: Creating a doPost() method

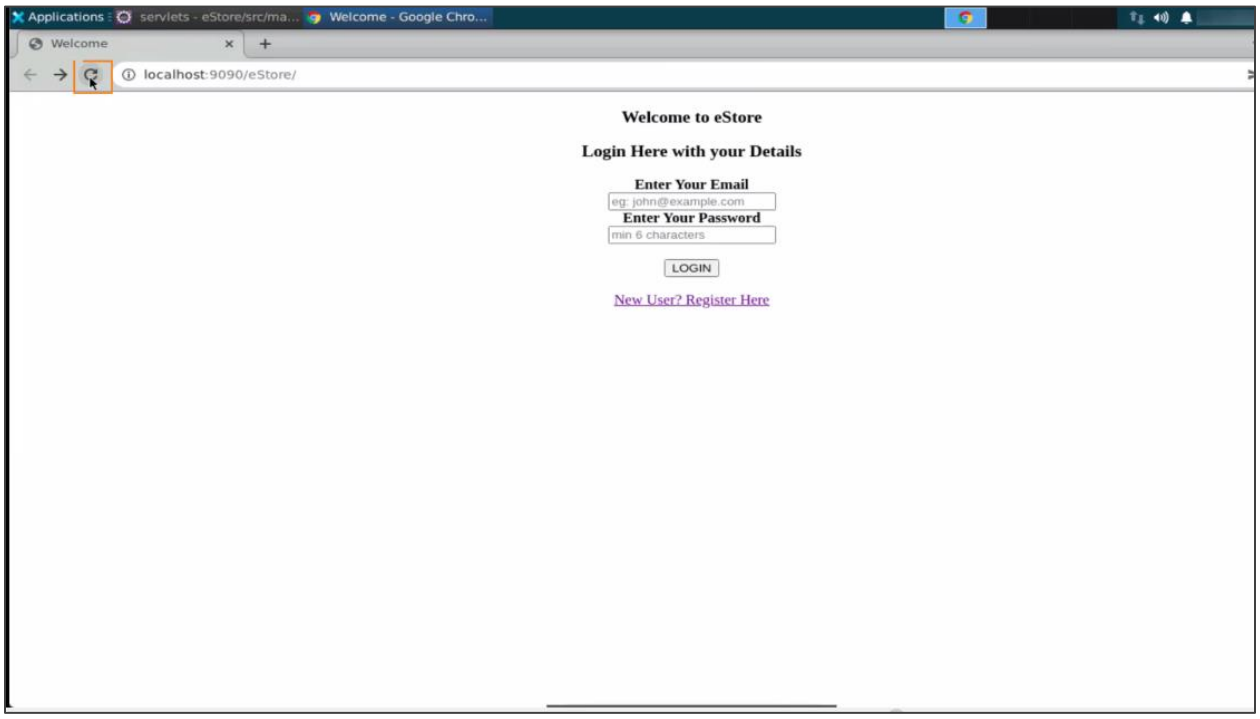
4.1 Go to the **index.html** page



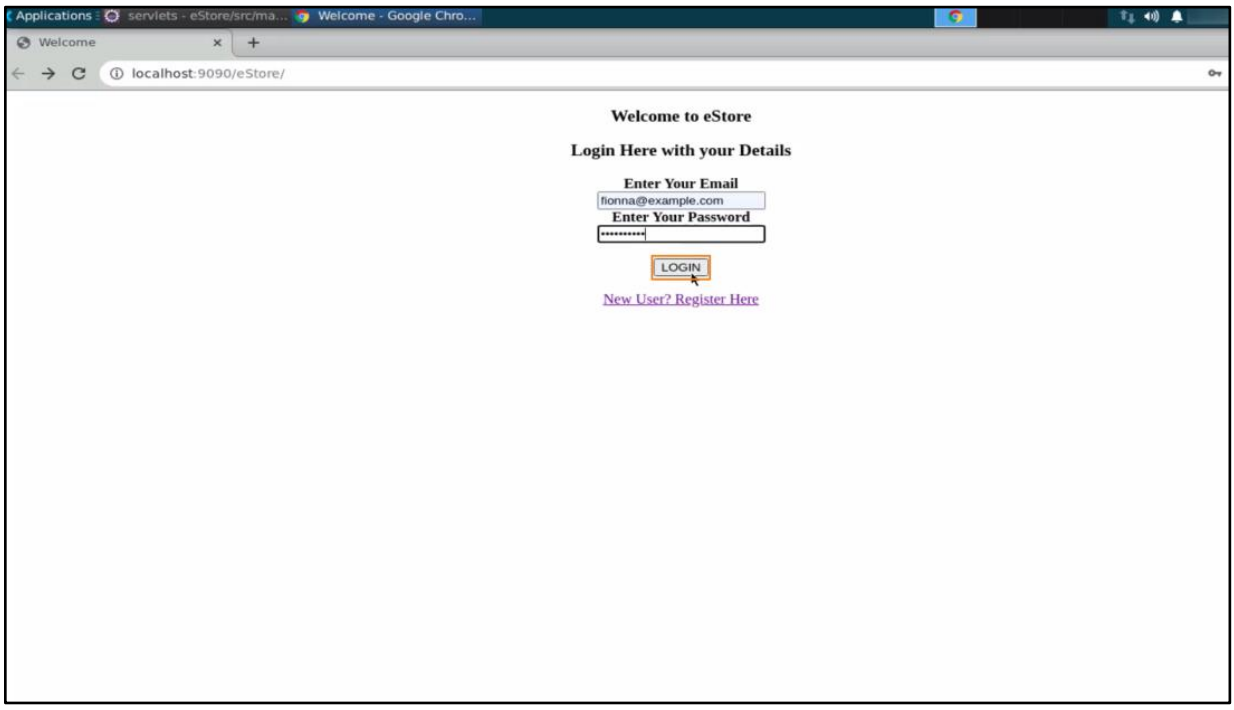
4.2 Change the method to **doPost()**



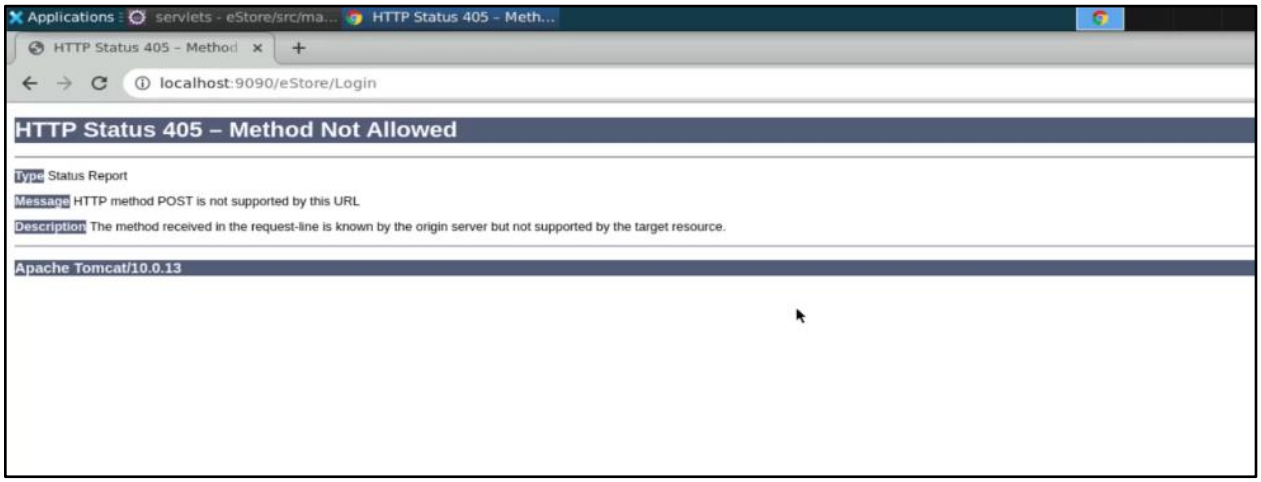
4.3 Refresh the browser



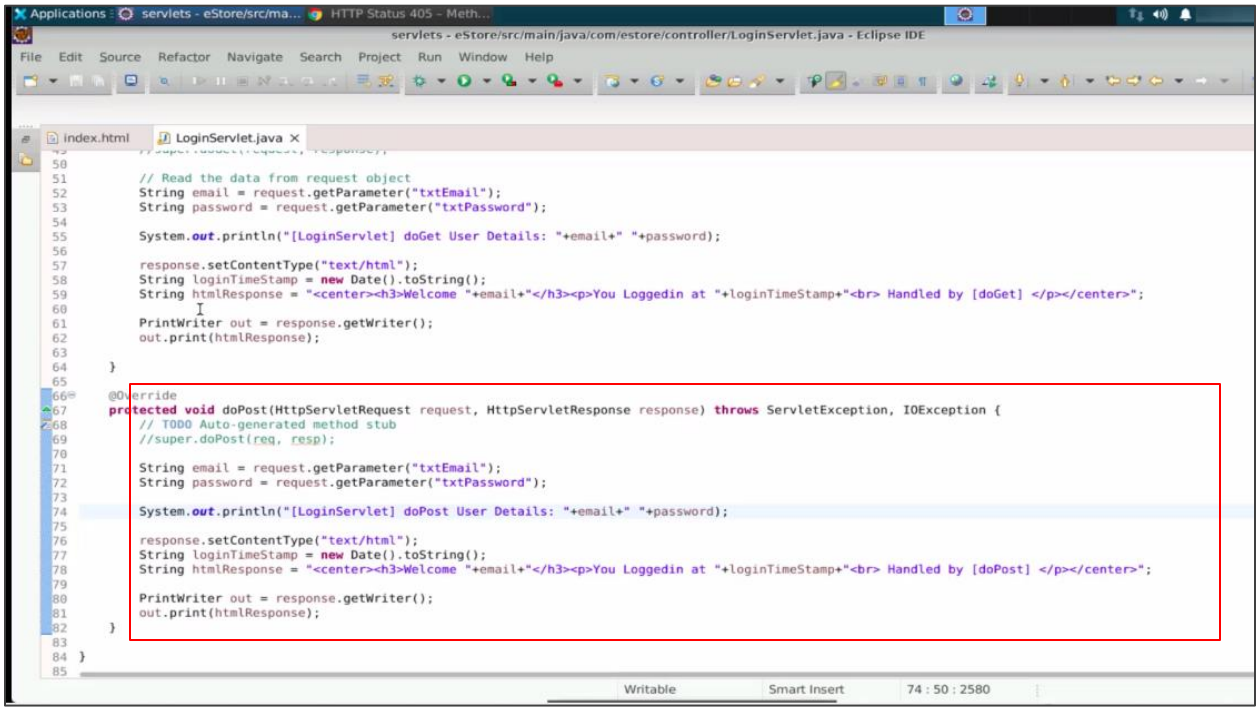
4.4 Enter the username and password and click **LOGIN**



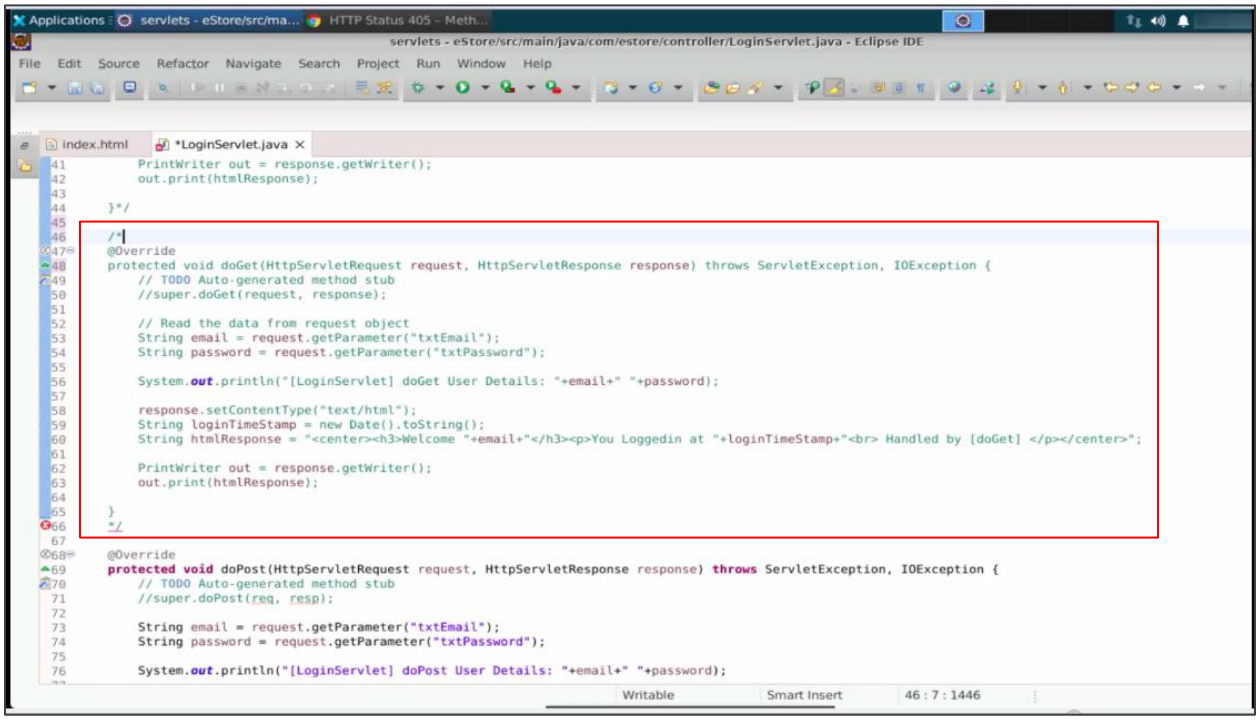
Observation: The server throws an exception, which indicates that the login servlet is not capable of entertaining the request in the form of a post. This means that the URL does not support the HTTP method POST.



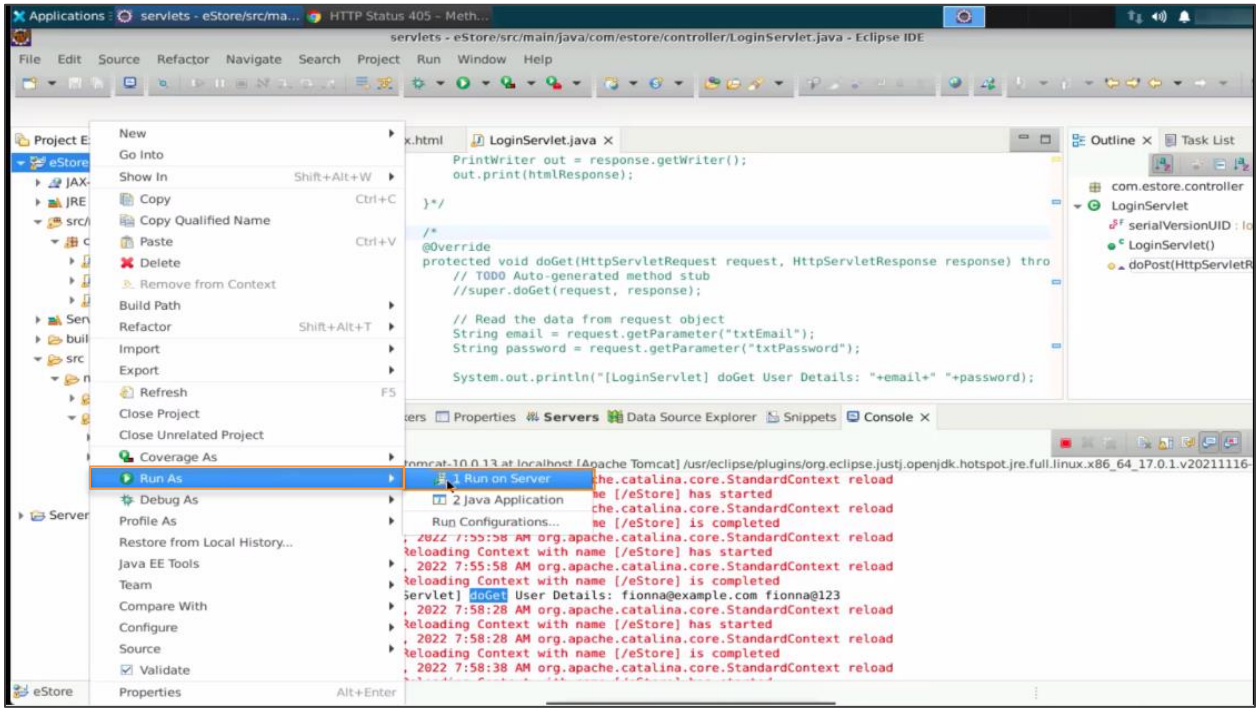
4.5 Go back to the **LoginServlet.java** file and override the **doPost()** method (lines 66 to 84). Change it to **Handled by [doPost]** and save it.



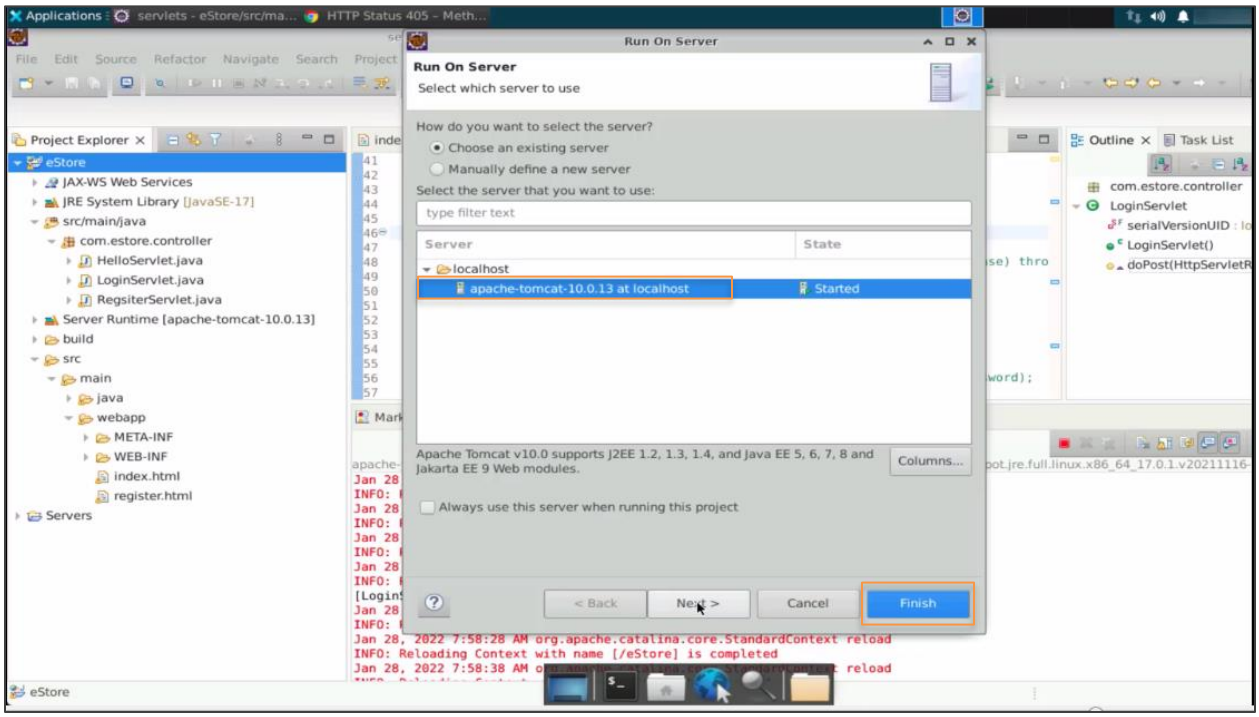
4.6 Comment the **doGet()** method and save it



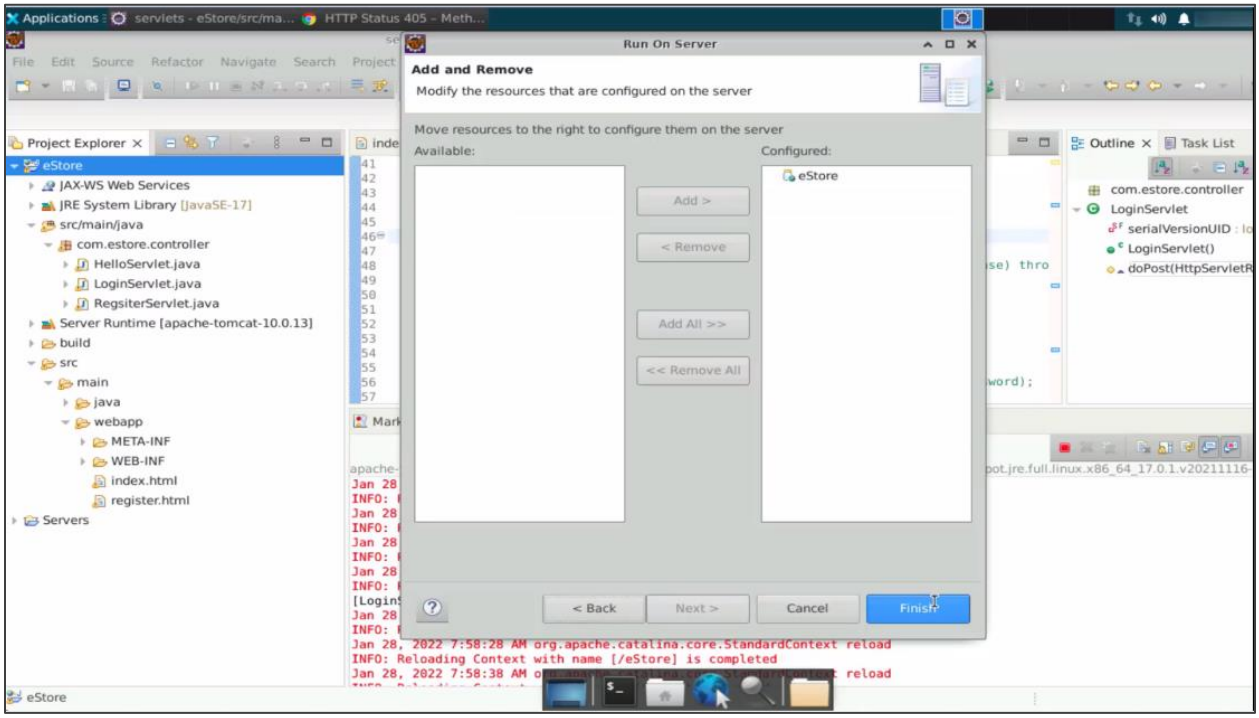
4.7 Run the code on the server by clicking **Run As** and **Run on Server**



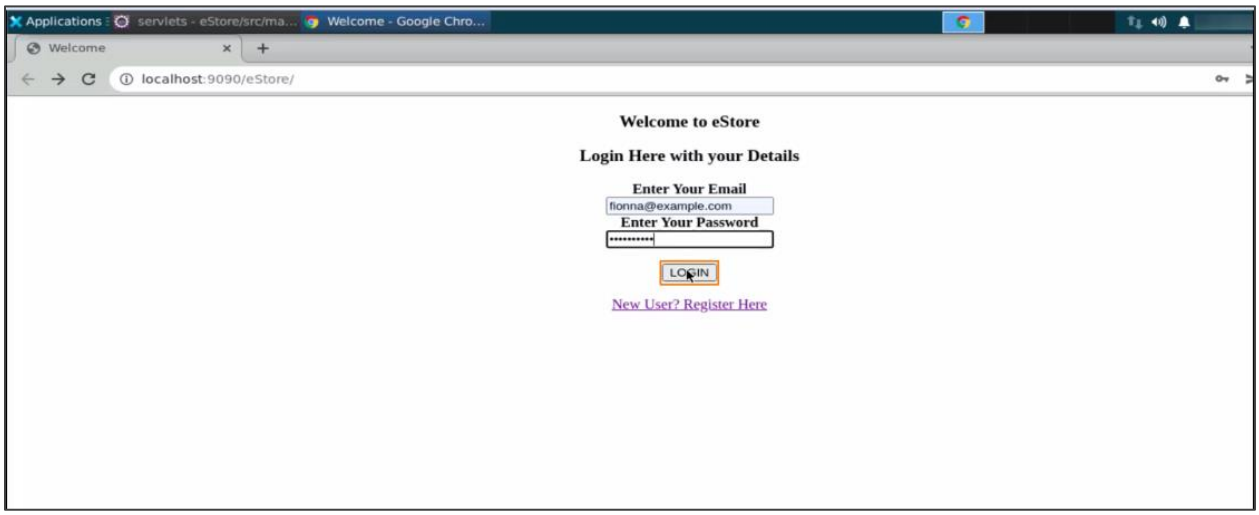
4.8 Select the **Apache Tomcat** server and click **Next**



4.9 Click **Finish**



4.10 Go to the browser and refresh the page. Enter the username and password for **Fionna**, and click **LOGIN**



You can see the output as **Handled by [doPost].** This means that the **doPost() method** is now working.

This indicates that the Servlet is handling the doGet and doPost methods per the requirements. The service method can also be used to address this issue. There are also other methods such as doPut and doDelete but in general, GET and POST are used.

Following these steps, you have successfully explored the doGet() and doPost() services in Servlet.