

Lesson 02 Demo 01

Session Tracking in Servlets

Objective: To demonstrate the implementation of session tracking in servlets using cookies

Tools Required: Eclipse IDE

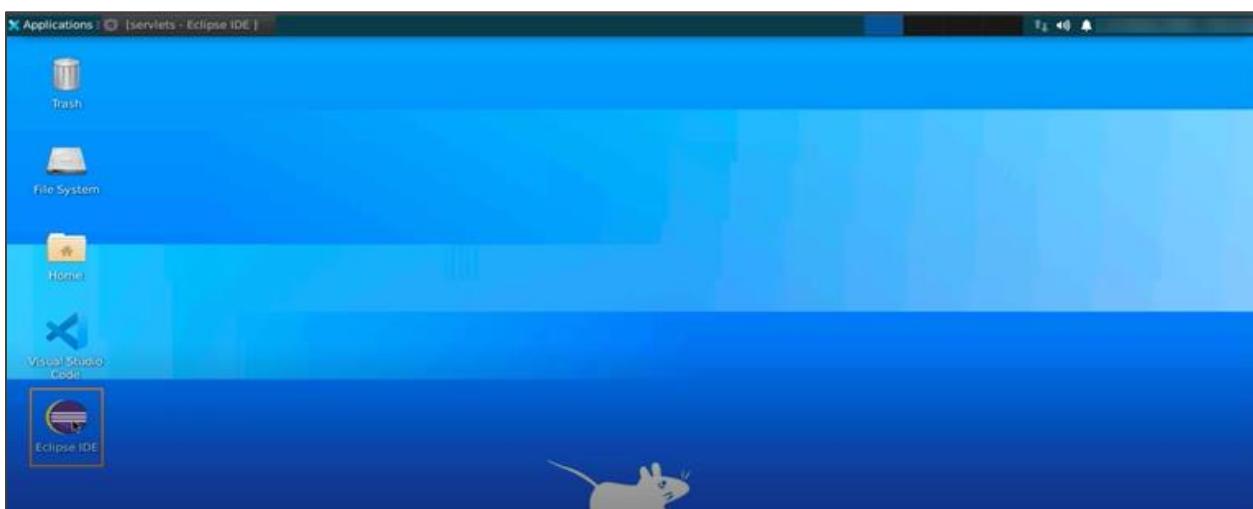
Prerequisites: None

Steps to be followed:

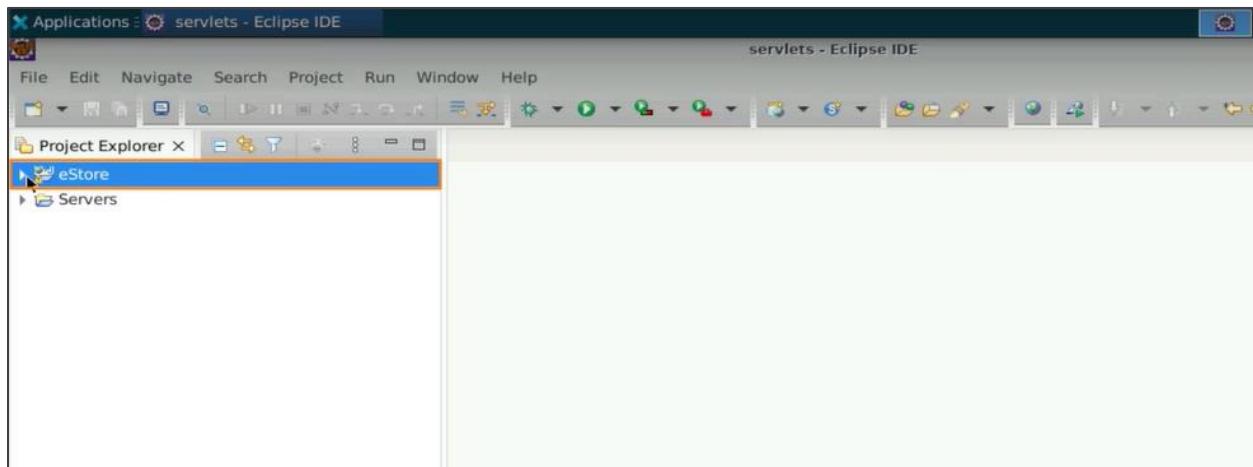
1. Set up and authenticate the user
2. Implement session tracking with cookies

Step 1: Set up and authenticating the user

1.1 Open Eclipse IDE

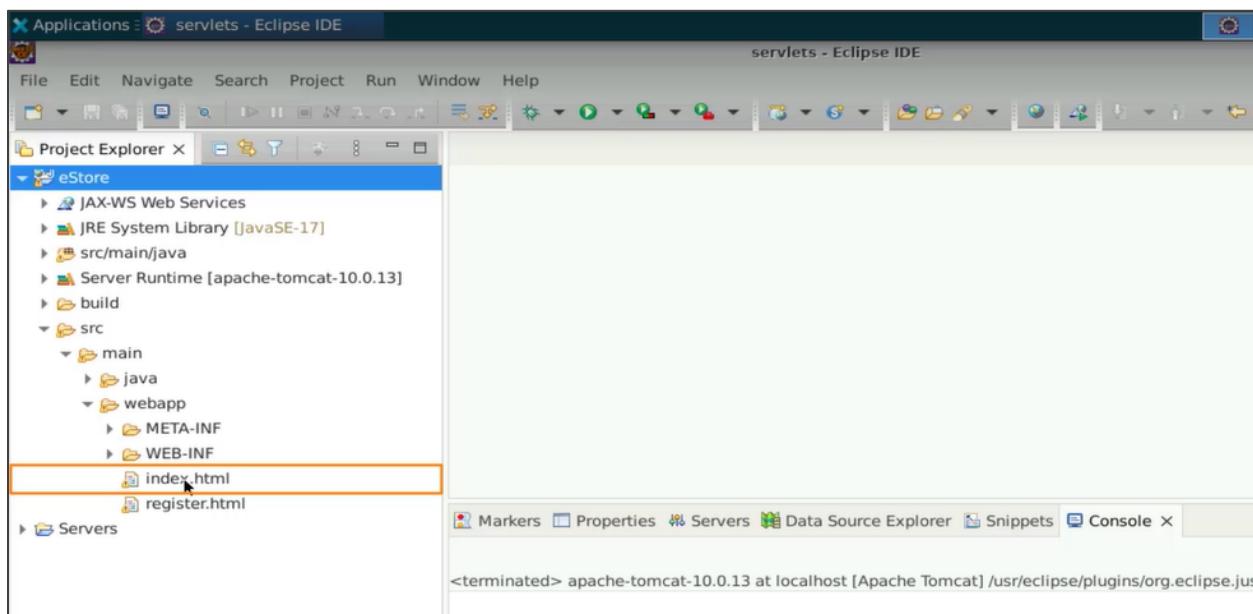


1.2 Open the eStore project



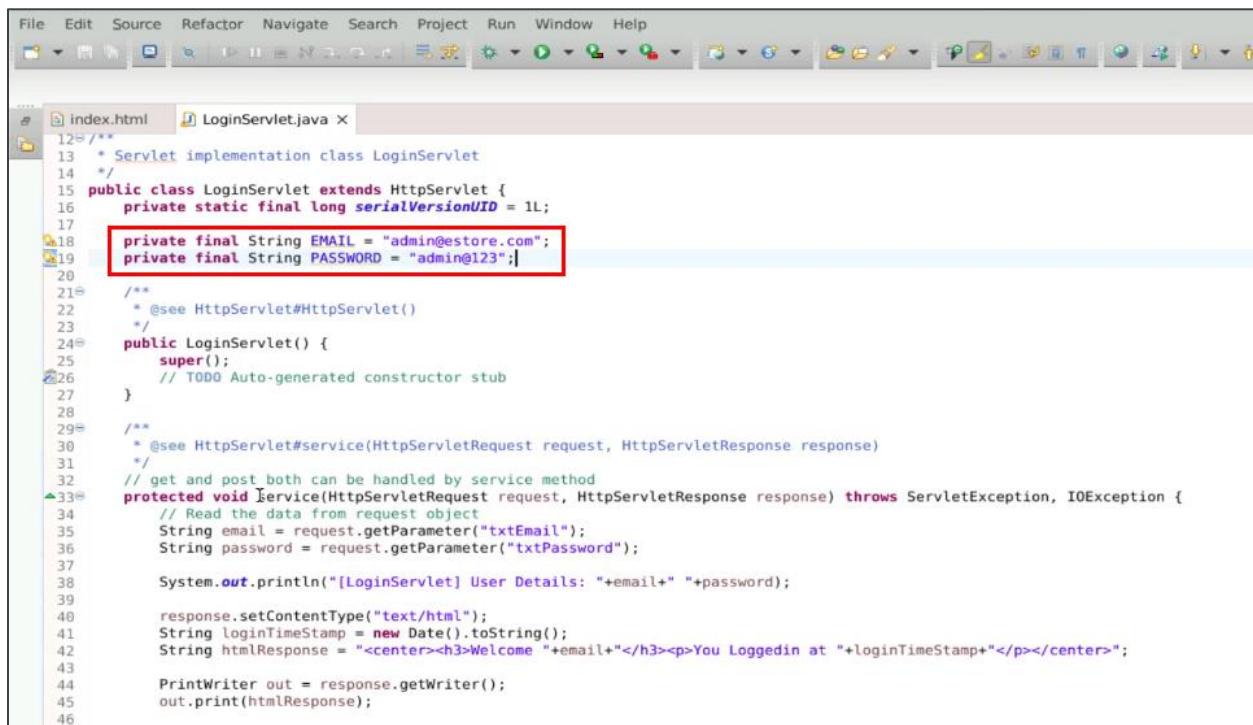
Note: Please refer to the previous demo on how to create the **eStore** project

1.3 Navigate to the **index.html** page under the **webapp** folder, which serves as the registration page



Note: In the **index.html** file, there is a login form as **Login** for the user with the action set and a link to navigate to **register.html** for account registration.

1.4 In the **LoginServlet.java**, globally define the attributes such as **EMAIL** and **PASSWORD** as private and final



The screenshot shows a Java code editor with the file **LoginServlet.java** open. The code defines a servlet class **LoginServlet** that extends **HttpServlet**. It includes two private final static fields: **EMAIL** and **PASSWORD**, both set to "admin@store.com". The code also contains a constructor, a service method, and a main block that reads parameters from the request, prints them to the console, sets the response content type to text/html, and prints a welcome message with the user's email and the current timestamp.

```
File Edit Source Refactor Navigate Search Project Run Window Help
Index.html LoginServlet.java X
12 /**
13  * Servlet implementation class LoginServlet
14 */
15 public class LoginServlet extends HttpServlet {
16     private static final long serialVersionUID = 1L;
17
18     private final String EMAIL = "admin@store.com";
19     private final String PASSWORD = "admin@123";
20
21     /**
22      * @see HttpServlet#HttpServlet()
23      */
24     public LoginServlet() {
25         super();
26         // TODO Auto-generated constructor stub
27     }
28
29     /**
30      * @see HttpServlet#service(HttpServletRequest request, HttpServletResponse response)
31      */
32     // get and post both can be handled by service method
33     protected void service(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
34         // Read the data from request object
35         String email = request.getParameter("txtEmail");
36         String password = request.getParameter("txtPassword");
37
38         System.out.println("[LoginServlet] User Details: "+email+" "+password);
39
40         response.setContentType("text/html");
41         String loginTimeStamp = new Date().toString();
42         String htmlResponse = "<center><h3>Welcome "+email+"</h3><p>You Loggedin at "+loginTimeStamp+"</p></center>";
43
44         PrintWriter out = response.getWriter();
45         out.print(htmlResponse);
46     }
}
```

1.5 Comment the **doGet** and **doPost** methods in the **LoginServlet.java** file

```

File Edit Source Refactor Navigate Search Project Run Window Help
index.html *LoginServlet.java X
52     // Read the data from request object
53     String email = request.getParameter("txtEmail");
54     String password = request.getParameter("txtPassword");
55
56     System.out.println("[LoginServlet] doGet User Details: "+email+" "+password);
57
58     response.setContentType("text/html");
59     String loginTimeStamp = new Date().toString();
60     String htmlResponse = "<center><h3>Welcome "+email+"</h3><p>You Loggedin at "+loginTimeStamp+"<br> Handled by [doGet] </p></center>";
61
62     PrintWriter out = response.getWriter();
63     out.print(htmlResponse);
64
65 /**
66 */
67 /*
68     @Override
69     protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
70         // TODO Auto-generated method stub
71         super.doPost(req, resp);
72
73         String email = request.getParameter("txtEmail");
74         String password = request.getParameter("txtPassword");
75
76         System.out.println("[LoginServlet] doPost User Details: "+email+" "+password);
77
78         response.setContentType("text/html");
79         String loginTimeStamp = new Date().toString();
80         String htmlResponse = "<center><h3>Welcome "+email+"</h3><p>You Loggedin at "+loginTimeStamp+"<br> Handled by [doPost] </p></center>";
81
82         PrintWriter out = response.getWriter();
83         out.print(htmlResponse);
84     }
85 */
86

```

1.6 In the service method of **LoginServlet.java**, add an if-else condition to validate if the email and password match. If they match, show the welcome page with a button to navigate to the home page. Otherwise, display a message saying **Invalid Username or Password** along with the timestamp of the login.

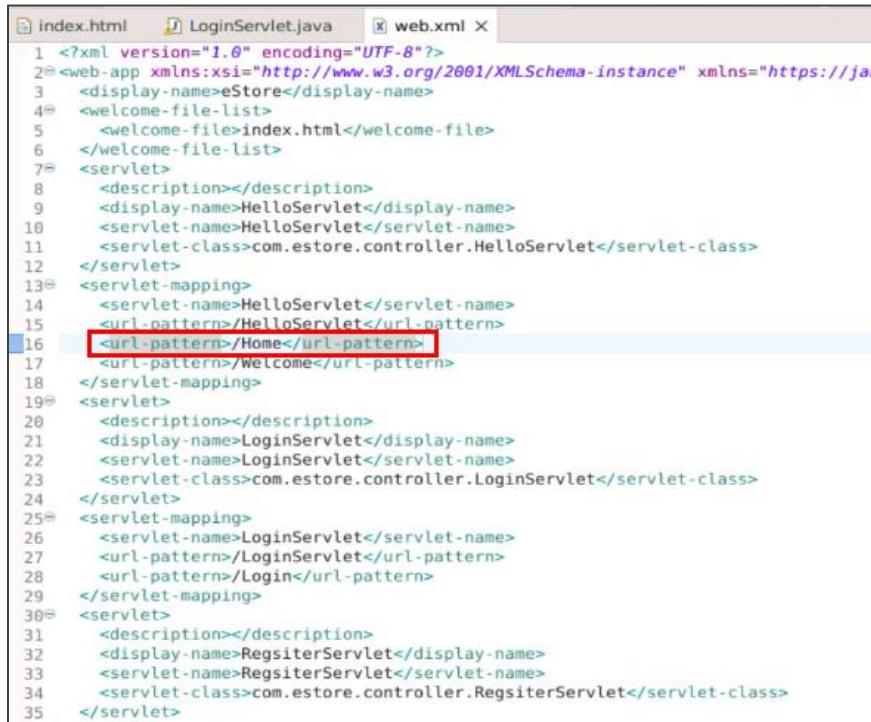
```

index.html LoginServlet.java X
29 /**
30     * @see HttpServlet#service(HttpServletRequest request, HttpServletResponse response)
31     */
32     // get and post both can be handled by service method
33     protected void service(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
34         // Read the data from request object
35         String email = request.getParameter("txtEmail");
36         String password = request.getParameter("txtPassword");
37
38         //System.out.println("[LoginServlet] User Details: "+email+" "+password);
39
40         response.setContentType("text/html");
41         String message = "";
42
43         if(email.equals(EMAIL) && password.equals(PASSWORD)) {
44             message += "<p>Welcome to Home<br><a href='Home'>Click to navigate to Home</a></p>";
45         }else {
46             message = "<b>Inavlid Username or Password</b>";
47         }
48
49
50         String loginTimeStamp = new Date().toString();
51         String htmlResponse = "<center><h3>Welcome "+email
52             + "</h3><p>You Attempted LogIn at "
53             + loginTimeStamp+"</p><br><br>"
54             +message+"</center>";
55
56         PrintWriter out = response.getWriter();
57         out.print(htmlResponse);
58     }
59
60

```

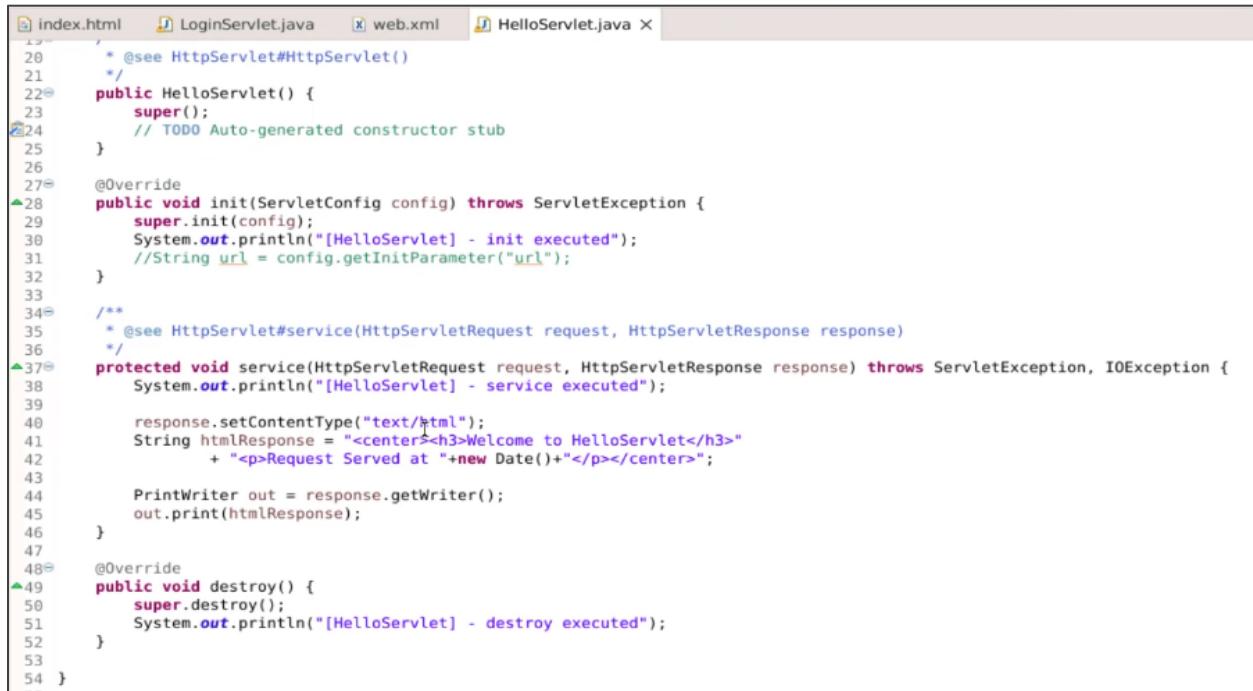
Observation: From the index page, a request will be sent to the **LoginServlet**, which will capture the email and password from the request.

1.7 Open the **web.xml** file and update the **url-pattern** of HelloServlet from **/Hello** to **/Home**



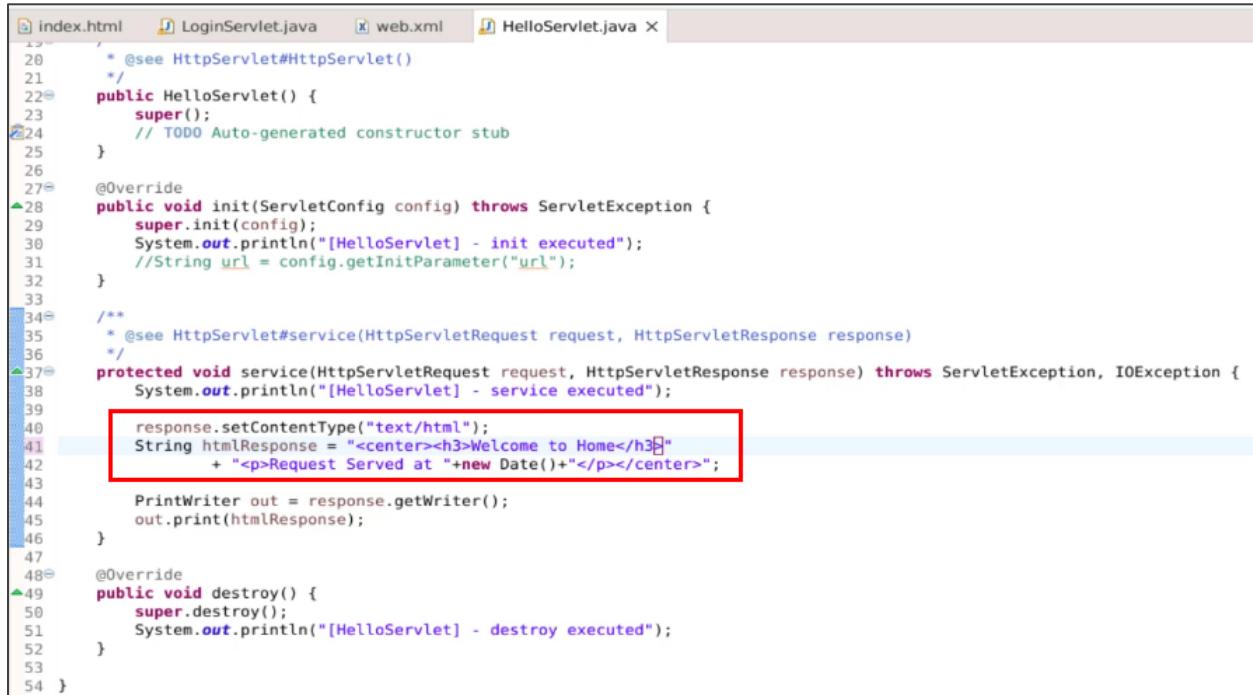
```
index.html LoginServlet.java web.xml
1 <?xml version="1.0" encoding="UTF-8"?>
2 <web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="https://jakarta.ee/xml/ns/jakarta-web-apis/2.0">
3   <display-name>eStore</display-name>
4   <welcome-file-list>
5     <welcome-file>index.html</welcome-file>
6   </welcome-file-list>
7   <servlet>
8     <description></description>
9     <display-name>HelloServlet</display-name>
10    <servlet-name>HelloServlet</servlet-name>
11    <servlet-class>com.estore.controller.HelloServlet</servlet-class>
12  </servlet>
13  <servlet-mapping>
14    <servlet-name>HelloServlet</servlet-name>
15    <url-pattern>/HelloServlet</url-pattern>
16    <url-pattern>/Home</url-pattern>[Red box highlights this line]
17    <url-pattern>/Welcome</url-pattern>
18  </servlet-mapping>
19  <servlet>
20    <description></description>
21    <display-name>LoginServlet</display-name>
22    <servlet-name>LoginServlet</servlet-name>
23    <servlet-class>com.estore.controller.LoginServlet</servlet-class>
24  </servlet>
25  <servlet-mapping>
26    <servlet-name>LoginServlet</servlet-name>
27    <url-pattern>/LoginServlet</url-pattern>
28    <url-pattern>/Login</url-pattern>
29  </servlet-mapping>
30  <servlet>
31    <description></description>
32    <display-name>RegisterServlet</display-name>
33    <servlet-name>RegisterServlet</servlet-name>
34    <servlet-class>com.estore.controller.RegisterServlet</servlet-class>
35  </servlet>
```

1.8 Open the **HelloServlet.java** servlet file where the init and destroy methods were previously configured



```
index.html LoginServlet.java web.xml HelloServlet.java X
1  /*
2  * @see HttpServlet#HttpServlet()
3  */
4  public HelloServlet() {
5      super();
6      // TODO Auto-generated constructor stub
7  }
8
9  @Override
10 public void init(ServletConfig config) throws ServletException {
11     super.init(config);
12     System.out.println("[HelloServlet] - init executed");
13     //String url = config.getInitParameter("url");
14 }
15
16 /**
17 * @see HttpServlet#service(HttpServletRequest request, HttpServletResponse response)
18 */
19 protected void service(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
20     System.out.println("[HelloServlet] - service executed");
21
22     response.setContentType("text/html");
23     String htmlResponse = "<center><h3>Welcome to HelloServlet</h3>" +
24         "<p>Request Served at " +new Date()+"</p></center>";
25
26     PrintWriter out = response.getWriter();
27     out.print(htmlResponse);
28 }
29
30 @Override
31 public void destroy() {
32     super.destroy();
33     System.out.println("[HelloServlet] - destroy executed");
34 }
35
36 }
```

1.9 Update the **htmlResponse** message in the service method of **HelloServlet.java** to display **Welcome to Home** along with the current date and timestamp



```
index.html LoginServlet.java web.xml HelloServlet.java X
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
```

```
    * @see HttpServlet#HttpServlet()
    */
public HelloServlet() {
    super();
    // TODO Auto-generated constructor stub
}

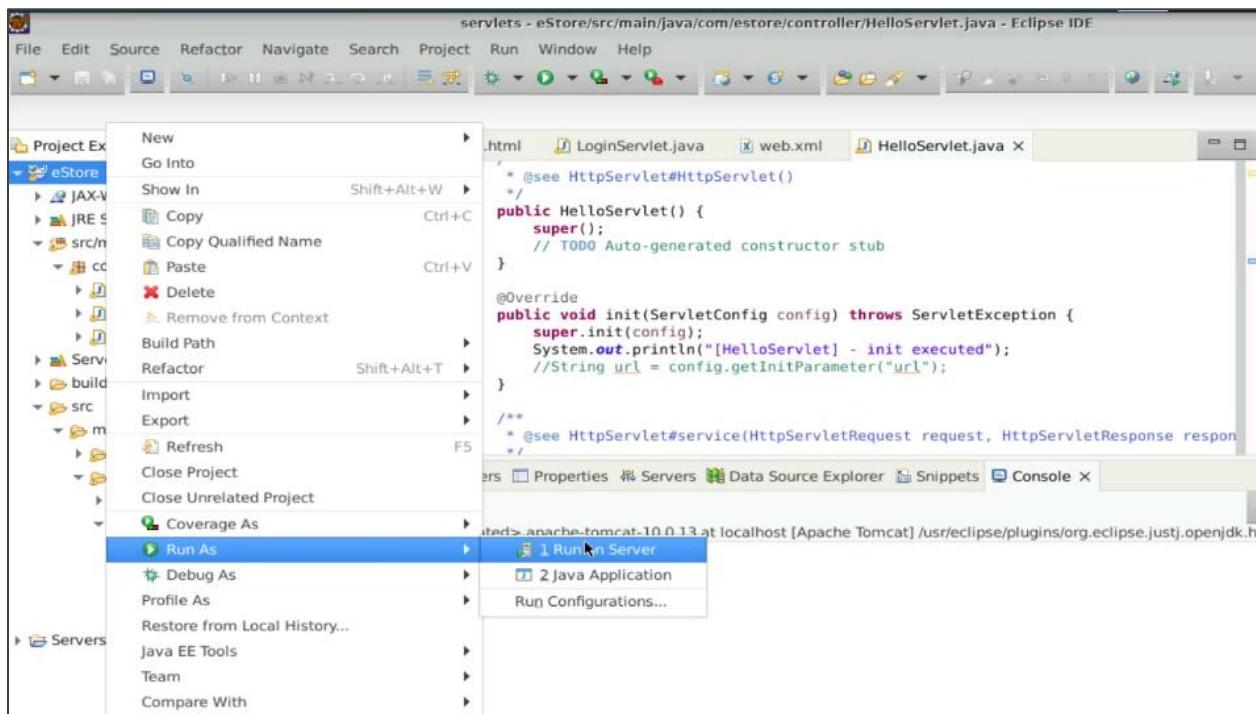
@Override
public void init(ServletConfig config) throws ServletException {
    super.init(config);
    System.out.println("[HelloServlet] - init executed");
    //String url = config.getInitParameter("url");
}

/**
 * @see HttpServlet#service(HttpServletRequest request, HttpServletResponse response)
 */
protected void service(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
    System.out.println("[HelloServlet] - service executed");

    response.setContentType("text/html");
    String htmlResponse = "<center><h3>Welcome to Home</h3>
        + "<p>Request Served at "+new Date()+"</p></center>";
    PrintWriter out = response.getWriter();
    out.print(htmlResponse);
}

@Override
public void destroy() {
    super.destroy();
    System.out.println("[HelloServlet] - destroy executed");
}
```

1.10 Save and run the code by right clicking the **eStore** project, selecting **Run As**, and selecting **Run on Server**



1.11 Enter invalid credentials and click the **LOGIN** button





Observation: You will observe the output displaying **Welcome username@example.com** with the attempted login timestamp. This information indicates a successful login attempt.

In case of an invalid username or password, you will receive the message **Invalid Username or Password**. Additionally, you can view the GET request sent during the login process in the URL.

1.12 Go to the **index.html** file and change the method to **POST** to ensure that the data is not present in the URL

```

<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>Welcome</title>
</head>
<body>
<center>
<h3>Welcome to eStore</h3>
<h3>Login Here with your Details</h3>
<form action="Login" method="post">
<br><b>Enter Your Email</b><br>
<input type="text" name="txtEmail" placeholder="eg: john@example.com"/><br>
<b>Enter Your Password</b><br>
<input type="password" name="txtPassword" placeholder="min 6 characters"/><br>
<br>

```

1.13 Enter the correct credentials for the admin and click on the **LOGIN** button



Observation: Now you will observe that the URL doesn't contain any data and the message **Welcome to Home** is shown.

1.14 Click on the **Click to navigate to Home** link





The result will be displayed as **Welcome to Home**, indicating that you have successfully navigated to the home URL.

1.15 Add a private final String in the **LoginServlet.java** file for the variables **NAME** and **TOTAL_SALES**, with the values set as **John** and **3000**, respectively

```

index.html *LoginServlet.java x web.xml HelloServlet.java
1 package com.estore.controller;
2
3 import jakarta.servlet.ServletException;
4
5 /**
6  * Servlet implementation class LoginServlet
7 */
8 public class LoginServlet extends HttpServlet {
9     private static final long serialVersionUID = 1L;
10
11     private final String EMAIL = "admin@estore.com";
12     private final String PASSWORD = "admin@123";
13
14     private final String NAME = "John";
15     private final int TOTAL_SALES = 30000;
16
17     /**
18      * @see HttpServlet#HttpServlet()
19      */
20     public LoginServlet() {
21         super();
22         // TODO Auto-generated constructor stub
23     }
24
25     /**
26      * @see HttpServlet#service(HttpServletRequest request, HttpServletResponse response)
27      */
28     // get and post both can be handled by service method
29     protected void service(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
30         // Read the data from request object
31         String email = request.getParameter("txtEmail");
32         String password = request.getParameter("txtPassword");
33
34         //System.out.println("[LoginServlet] User Details: "+email+" "+password);
35     }
36 }
37
38
39
40
41
42

```

Step 2: Implement session tracking with cookies

2.1 Go to the `LoginServlet.java` file to implement session tracking with cookies

The screenshot shows the Eclipse IDE interface. The top menu bar includes File, Edit, Source, Refactor, Navigate, Search, Project, Run, Window, and Help. Below the menu is a toolbar with various icons for file operations like Open, Save, Cut, Copy, Paste, Find, and others. The left side features the Project Explorer view, which lists the project structure: eStore (containing JAX-WS Web Services, JRE System Library [JavaSE-17], src/main/java, and src). The src/main/java folder contains com.estore.controller (HelloServlet.java, LoginServlet.java, RegisterServlet.java), build, and src (main and webapp). The webapp folder contains META-INF, WEB-INF, index.html, and register.html. The right side shows the Java Editor with the code for LoginServlet.java:

```
1 package com.estore.controller;
2
3 * import jakarta.servlet.ServletException;
4 /**
5  * Servlet implementation class LoginServlet
6  */
7 public class LoginServlet extends HttpServlet {
8     private static final long serialVersionUID = 1L;
9
10    /**
11     * @see HttpServlet#HttpServlet()
12     */
13    public LoginServlet() {
14        super();
15        // TODO Auto-generated constructor stub
16    }
17}
```

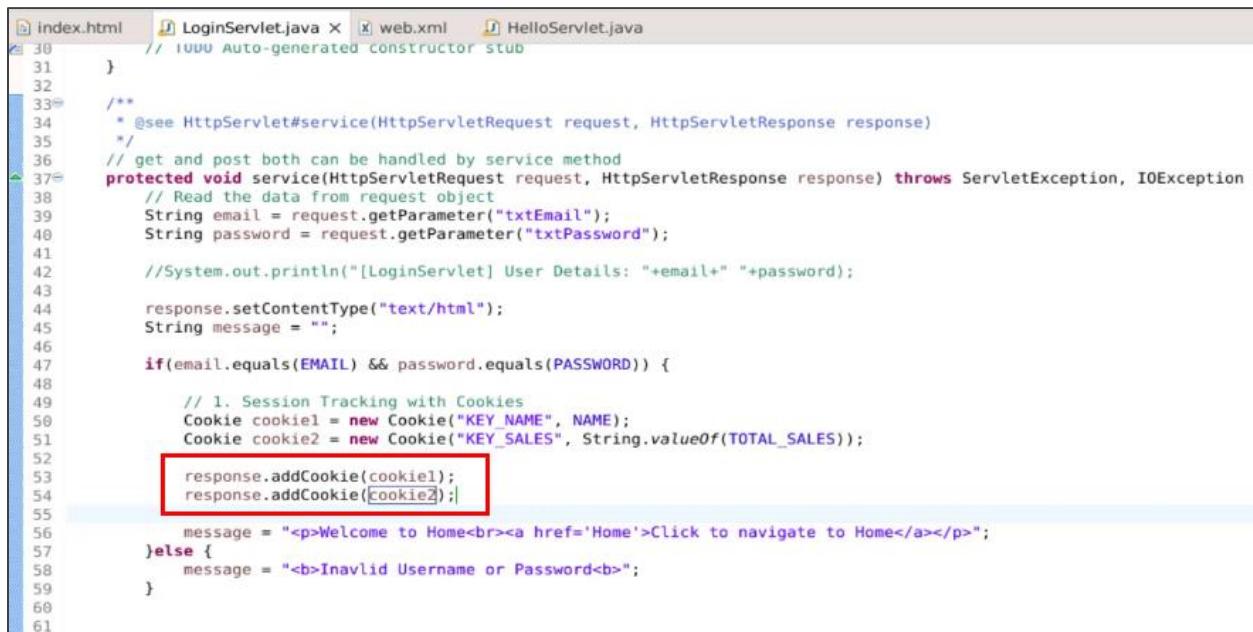
Below the editor are several tabs: Markers, Properties, Servers, Data Source Explorer, Snippets, and Console. The Console tab shows the output: <terminated> apache-tomcat-10.0.13 at localhost [Apache Tomcat] /usr/eclipse/plugins/org.eclipse.justj.openjdk.h

2.2 Under the if clause in the **LoginServlet**, create Cookie objects named **cookie1** and **cookie2** to store the values **NAME (John)** and **TOTAL_SALES (3000)**

```
index.html LoginServlet.java X web.xml HelloServlet.java
30 // TODO Auto-generated constructor stub
31
32
33 /**
34 * @see HttpServlet#service(HttpServletRequest request, HttpServletResponse response)
35 */
36 // get and post both can be handled by service method
37 protected void service(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
38     // Read the data from request object
39     String email = request.getParameter("txtEmail");
40     String password = request.getParameter("txtPassword");
41
42     //System.out.println("[LoginServlet] User Details: "+email+" "+password);
43
44     response.setContentType("text/html");
45     String message = "";
46
47     if(email.equals(EMAIL) && password.equals(PASSWORD)) {
48
49         // 1. Session Tracking with Cookies
50         Cookie cookie1 = new Cookie("KEY_NAME", NAME);
51         Cookie cookie2 = new Cookie("KEY_SALES", String.valueOf(TOTAL_SALES));
52
53         message = "<p>Welcome to Home<br><a href='Home'>Click to navigate to Home</a></p>";
54     }else {
55         message = "<b>Inavlid Username or Password</b>";
56     }
57
58
59     String loginTimeStamp = new Date().toString();
60     String htmlResponse = "<center><h3>Welcome " +email
61             +"</h3><p>You Attempted LogIn at "
62             +loginTimeStamp+"</p><br><br>"
63             +message+"</center>";
64 }
```

Since cookies cannot store integer values, use the **String.valueOf()** method to convert the value of **TOTAL_SALES** into a string.

2.3 Add the cookie objects as responses using the **response.addCookie()** function



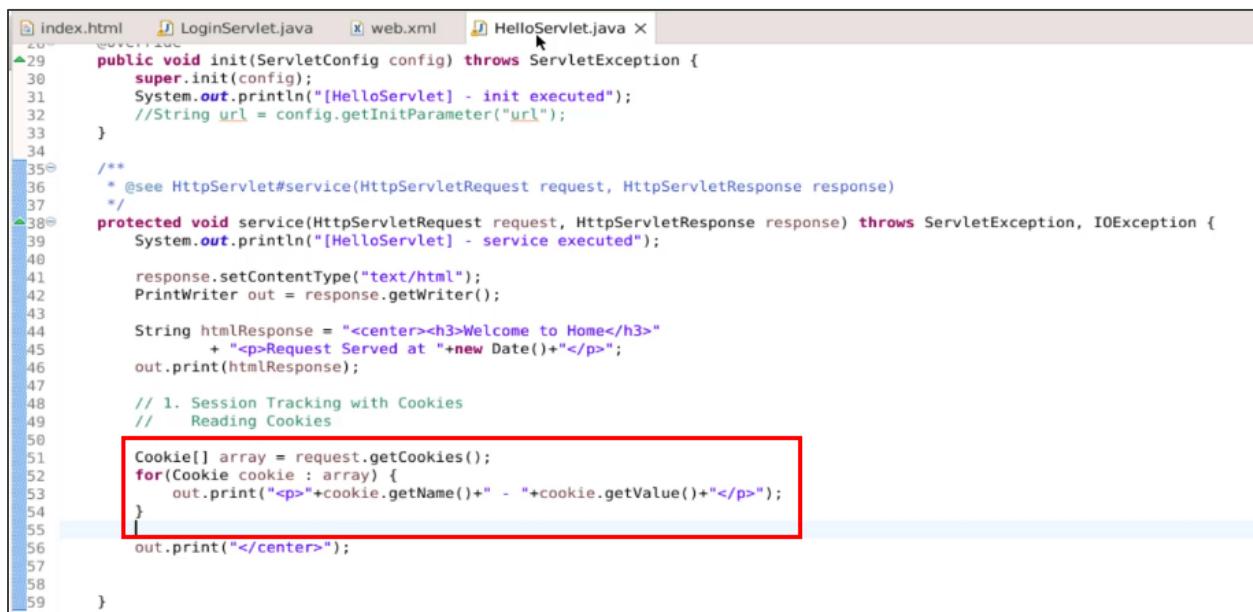
```

index.html LoginServlet.java X web.xml HelloServlet.java
30     }
31 }
32 }
33 /**
34 * @see HttpServlet#service(HttpServletRequest request, HttpServletResponse response)
35 */
36 // get and post both can be handled by service method
37 protected void service(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
38     // Read the data from request object
39     String email = request.getParameter("txtEmail");
40     String password = request.getParameter("txtPassword");
41
42     //System.out.println("[LoginServlet] User Details: "+email+" "+password);
43
44     response.setContentType("text/html");
45     String message = "";
46
47     if(email.equals(EMAIL) && password.equals(PASSWORD)) {
48
49         // 1. Session Tracking with Cookies
50         Cookie cookie1 = new Cookie("KEY_NAME", NAME);
51         Cookie cookie2 = new Cookie("KEY_SALES", String.valueOf(TOTAL_SALES));
52
53         response.addCookie(cookie1);
54         response.addCookie(cookie2);
55
56         message = "<p>Welcome to Home<br><a href='Home'>Click to navigate to Home</a></p>";
57     }else {
58         message = "<b>Invalid Username or Password</b>";
59     }
60
61

```

For session tracking, it is important to read cookies. First, we create the cookies, write them, and then read them.

2.4 Read the cookies from the request using the **Cookie[]** array and the **getCookies()** method. Iterate within the array using an enhanced for loop to print the name and value of each cookie

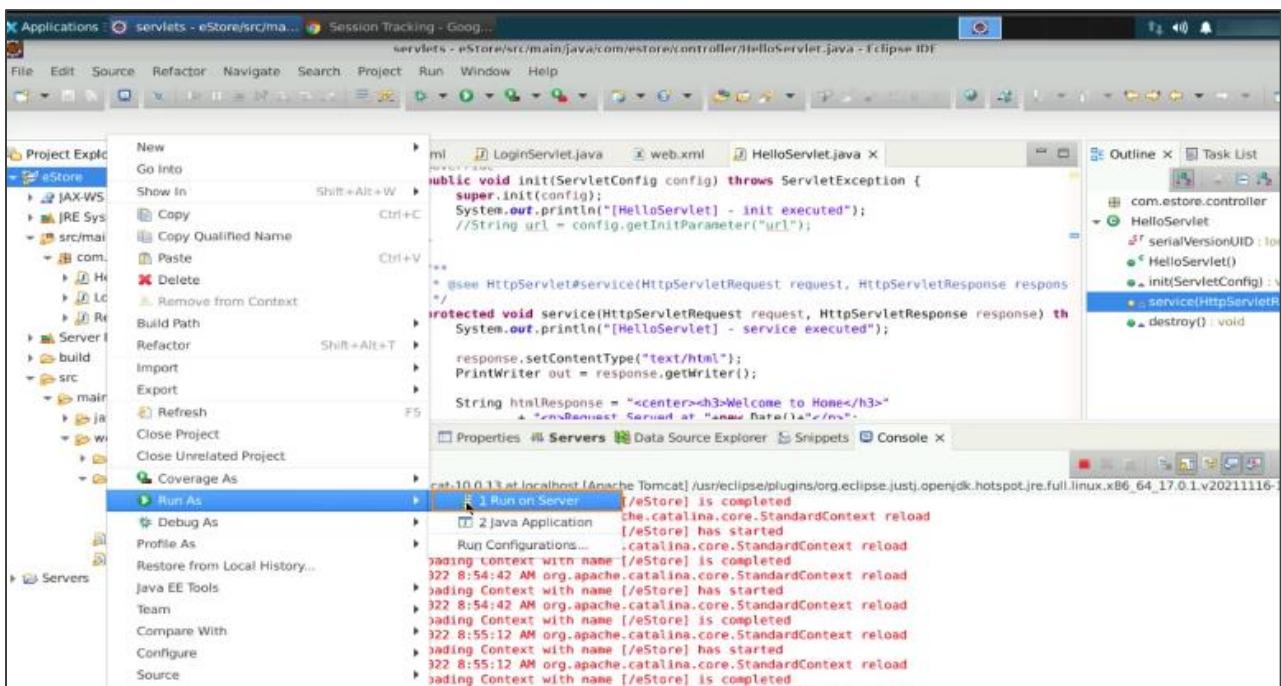


```

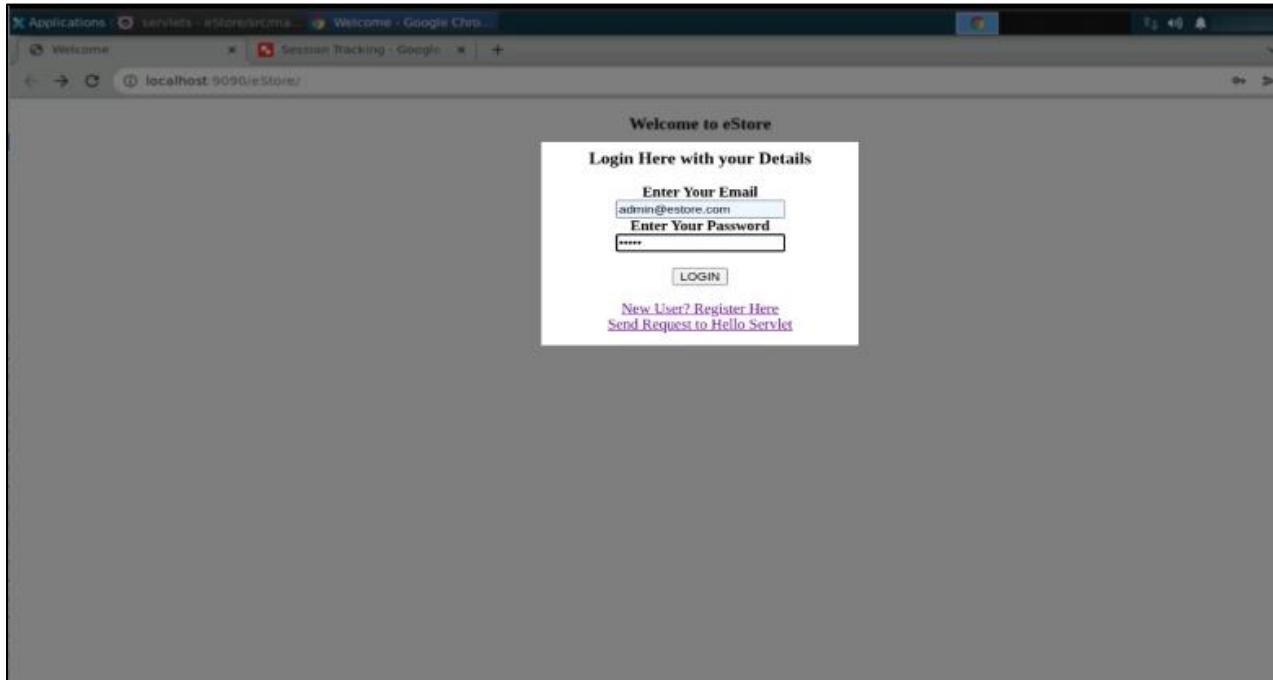
29  public void init(ServletConfig config) throws ServletException {
30      super.init(config);
31      System.out.println("[HelloServlet] - init executed");
32      //String url = config.getInitParameter("url");
33  }
34
35  /**
36   * @see HttpServlet#service(HttpServletRequest request, HttpServletResponse response)
37   */
38  protected void service(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
39      System.out.println("[HelloServlet] - service executed");
40
41      response.setContentType("text/html");
42      PrintWriter out = response.getWriter();
43
44      String htmlResponse = "<center><h3>Welcome to Home</h3>" +
45          "<p>Request Served at " + new Date() + "</p>";
46      out.print(htmlResponse);
47
48      // 1. Session Tracking with Cookies
49      // Reading Cookies
50
51      Cookie[] array = request.getCookies();
52      for(Cookie cookie : array) {
53          out.print("<p>" + cookie.getName() + " - " + cookie.getValue() + "</p>");
54      }
55
56      out.print("</center>");
57
58  }
59

```

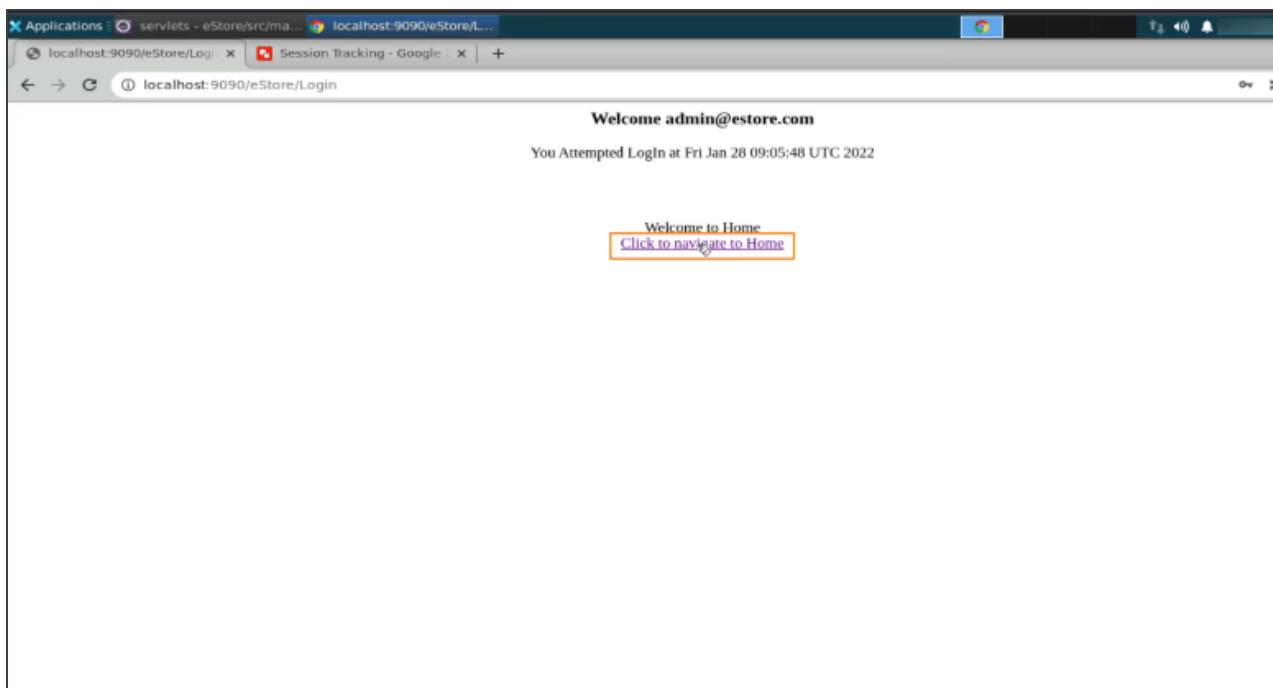
2.5 Save and rerun the code on the server

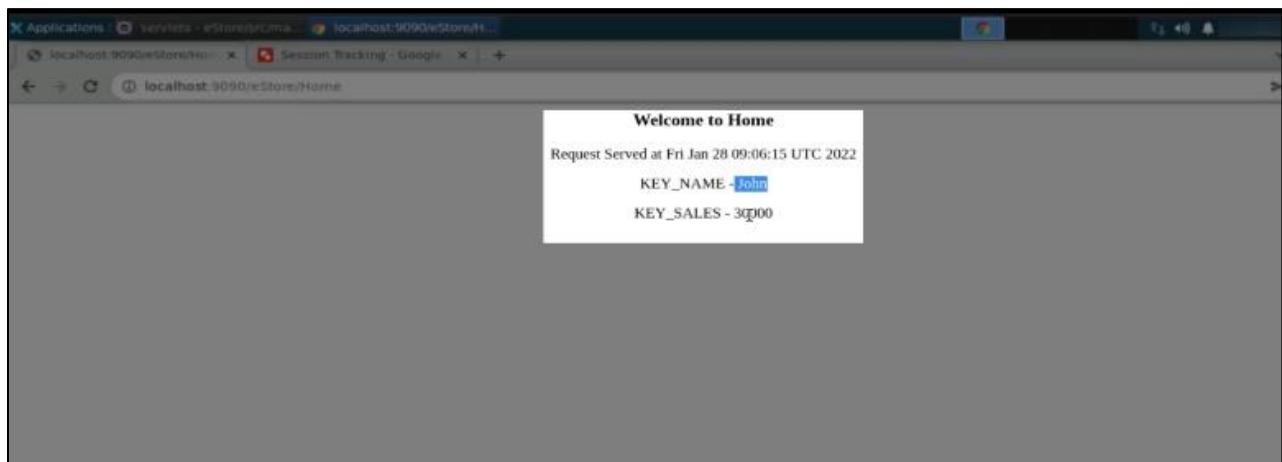


2.6 Return to the browser, log in again with the admin email and password, and click on the **LOGIN** button



2.7 Click on the **Click to navigate to Home** link





Observation: The result can be seen when it prints the **KEY_NAME** and **KEY_SALES** values. This is the first way to work with the session tracking technique. However, please note that this method may not work if cookies are disabled.

Following these steps, you have successfully implemented session tracking in servlets using cookies.