**Assisted Practice: 1.9 Session Tracking Using HTTP Session**

This section will guide you to:

* Create an HTML page to login with and without HTTPSession
* Create servlets to handle each request
* Show the output of the servlet showing whether the session was handled with HTTPSession or not

**Development Environment**

* Eclipse IDE for Enterprise Java Developers v2019-03 (4.11.0)
* Apache Tomcat Server v9.0
* JRE: OpenJDK Runtime Environment 11.0.2

This lab has ten subsections, namely:

* + 1. Creating a dynamic web project
    2. Creating an HTML page
    3. Creating a LoginServlet servlet
    4. Creating a Dashboard servlet
    5. Configuring web.xml
    6. Checking for servlet-api.jar
    7. Building the project
    8. Publishing and starting the project
    9. Running the project
    10. Pushing the code to your GitHub repositories

**Step 1.9.1:** Creating a dynamic web project

* Open Eclipse
* Go the **File** menu. Choose **New->Dynamic Web Project**
* Enter the project name as **HTTPSessionDemo**. Click on **Next**
* Enter nothing in the next screen and click on **Next**
* Check the checkbox **Generate web.xml deployment descriptor** and click on **Finish**
* This will create the project files in the Project Explorer

**Step 1.9.2:** Creating an HTML page

* In the Project Explorer, expand the project **HTTPSessionDemo**
* Expand **WebContent**. Right click on **WebContent**. Choose **New->HTML File**
* Enter the filename as **index.html** and click on **Finish**
* Enter the following code:

<!DOCTYPE html>

<**html**>

<**head**>

<**meta** charset="UTF-8">

<**title**>HTTPSession Demo</**title**>

</**head**>

<**body**>

<**a** href="login?userid=admin">Dashboard with Session based login</**a**><**br**>

<**a** href="dashboard">Dashboard without Session based login</**a**>

</**body**>

</**html**>

* Click on the **Save** icon

**Step 1.9.3:** Creating a LoginServlet servlet

* In the Project Explorer, expand **HTTPSessionDemo->Java Resources**
* Right click on **src** and choose **New->Servlet**
* In **Class Name,** enter **LoginServlet** and click on **Finish**
* Enter the following code:

**import java.io.\*;**

**import** javax.servlet.**\***;

**import** javax.servlet.annotation.**\***;

**import** javax.servlet.http.**\***;

/\*\*

**\*** Servlet implementation class LoginServlet

\*/

**@WebServlet("/LoginServlet")**

**public** **class** LoginServlet **extends** HttpServlet {

**private** **static** **final** long serialVersionUID = 1L;

/\*\*

**\*** **@see** HttpServlet**#**HttpServlet()

\*/

**public** LoginServlet() {

**super**();

// TODO Auto-generated constructor stub

}

/\*\*

**\*** **@see** HttpServlet**#**doGet(HttpServletRequest request**,** HttpServletResponse response)

\*/

**protected** void doGet(HttpServletRequest request, HttpServletResponse response) **throws** ServletException, **IOException** {

// TODO Auto-generated method stub

**String** userId = request.getParameter("userid");

HttpSession session=request.getSession();

session.setAttribute("userid", userId);

response.sendRedirect("dashboard");

}

/\*\*

**\*** **@see** HttpServlet**#**doPost(HttpServletRequest request**,** HttpServletResponse response)

\*/

**protected** void doPost(HttpServletRequest request, HttpServletResponse response) **throws** ServletException, **IOException** {

// TODO Auto-generated method stub

doGet(request, response);

}

}

**Step 1.9.4:** Creating a Dashboard servlet

* In the Project Explorer, expand **HTTPSessionDemo->Java Resources**
* Right click on **src** and choose **New->Servlet**
* In **Class Name,** enter **Dashboard** and click on **Finish**
* Enter the following code:

**import java.io.\*;**

**import** javax.servlet.**\***;

**import** javax.servlet.annotation.**\***;

**import** javax.servlet.http.**\***;

/\*\*

**\*** Servlet implementation class Dashboard

\*/

**@WebServlet("/Dashboard")**

**public** **class** Dashboard **extends** HttpServlet {

**private** **static** **final** long serialVersionUID = 1L;

/\*\*

**\*** **@see** HttpServlet**#**HttpServlet()

\*/

**public** Dashboard() {

**super**();

// TODO Auto-generated constructor stub

}

/\*\*

**\*** **@see** HttpServlet**#**doGet(HttpServletRequest request**,** HttpServletResponse response)

\*/

**protected** void doGet(HttpServletRequest request, HttpServletResponse response) **throws** ServletException, **IOException** {

// TODO Auto-generated method stub

**PrintWriter** out = response.getWriter();

out.println("<html><body>");

HttpSession session=request.getSession(**false**);

**String** userId = **null**;

**if** (session.getAttribute("userid") != **null**)

userId =(**String**)session.getAttribute("userid");

**if** (userId == **null** ) {

out.println("No UserId was found in session.<br>");

} **else** {

out.println("UserId obtained from session :" + userId + "<br>");

}

out.println("</body></html>");

}

/\*\*

**\*** **@see** HttpServlet**#**doPost(HttpServletRequest request**,** HttpServletResponse response)

\*/

**protected** void doPost(HttpServletRequest request, HttpServletResponse response) **throws** ServletException, **IOException** {

// TODO Auto-generated method stub

doGet(request, response);

}

}

**Step 1.9.5:** Configuring web.xml

* In the Project Explorer, expand **HTTPSessionDemo->WebContent->WEB-INF**
* Double click on **web.xml** to open it in the editor
* Enter the following script:

<?xml version="1.0" encoding="UTF-8"?>

<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="http://xmlns.jcp.org/xml/ns/javaee" xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/javaee http://xmlns.jcp.org/xml/ns/javaee/web-app\_4\_0.xsd" id="WebApp\_ID" version="4.0">

<display-name>HTTPSessionDemo</display-name>

<welcome-file-list>

<welcome-file>index.html</welcome-file>

<welcome-file>index.htm</welcome-file>

<welcome-file>index.jsp</welcome-file>

<welcome-file>default.html</welcome-file>

<welcome-file>default.htm</welcome-file>

<welcome-file>default.jsp</welcome-file>

</welcome-file-list>

<servlet>

<servlet-name>LoginServlet</servlet-name>

<servlet-class>LoginServlet</servlet-class>

</servlet>

<servlet>

<servlet-name>Dashboard</servlet-name>

<servlet-class>Dashboard</servlet-class>

</servlet>

<servlet-mapping>

<servlet-name>Dashboard</servlet-name>

<url-pattern>/dashboard</url-pattern>

</servlet-mapping>

<servlet-mapping>

<servlet-name>LoginServlet</servlet-name>

<url-pattern>/login</url-pattern>

</servlet-mapping>

</web-app>

</servlet-mapping>

<servlet-mapping>

<servlet-name>LoginServlet</servlet-name>

<url-pattern>/login</url-pattern>

</servlet-mapping>

</web-app>

**Step 1.9.6:** Checking for servlet-api.jar

* Before building the project, we need to add **servlet-api.jar** to the project
* Servlet-api.jar file is already present in your practice lab. (Refer FSD: Lab Guide - Phase 2)
* To add it to the project, follow the below mentioned steps:
  + In the Project Explorer, right click on the **HTTPSessionDemo** and choose **Properties**
  + Select **Java Build Path** from the options on the left
  + Click on **Libraries** tab on the right
  + Under **ClassPath,** expand the node that says **Apache Tomcat**
  + If there is an existing entry for the **servlet-api.jar,** then click on **Cancel** and exit the window
  + If it is not there, then click on **Classpath** entry and click on **Add External JARs** button on the right
  + From the file list, select the **servlet-api.jar** file and click on **Ok**
  + Click on **Apply and Close**

**Step 1.9.7:** Building the Project

* From the **Project** menu at the top, click on **Build**
* If any compile errors are shown, fix them as required

**Step 1.9.8:** Publishing and starting the project

* If you do not see the **Servers** tab near the bottom of the IDE, go to **Window** menu and click on **Show View->Servers**
* Right click on the **Server** entry and choose **Add and Remove**
* Click the **Add** button to move **FormFieldsTracking** from the **Available** list to the **Configured** list
* Click on **Finish**
* Right click on the **Server** entry and click on **Publish**
* Right click on the **Server** entry and click on **Start**
* This will start the server

**Step 1.9.9:** Running the project

* To run the project, open a web browser and type: [**http://localhost:8080/**](http://localhost:8080/ServletConcept)**HTTPSessionDemo**

**Step 1.9.10:** Pushing the code to your GitHub repositories

* Open your command prompt and navigate to the folder where you have created your files.

**cd <folder path>**

* Initialize your repository using the following command:

**git init**

* Add all the files to your git repository using the following command:

**git add .**

* Commit the changes using the following command:

**git commit . -m “Changes have been committed.”**

* Push the files to the folder you initially created using the following command:

**git push -u origin master**