In a web application, a web server may be responding to several clients at the same time. *Session tracking* is a way by which the server can identify the client. Since the HTTP protocol is stateless, the client needs to open a separate connection every time it interacts with the server, and the server treats each request as a new request.

In order to identify the client, the server needs to maintain the client's state. There are several session tracking techniques. For the purpose of this project we will focus on the Session Object. (Object <--> think Java object, sort of)

Session Object

A Session object is the representation of one user session. A user's Session starts when the user opens a browser and sends the first request to the server. The Session object is available in all the requests (in the entire user session) so attributes stored in the Http session will be available in any jsp (which means in any page of the web application).

When a session is created, the server generates a unique ID and attaches that ID to the session. The server sends back this ID to the client and from there on, the browser sends back this ID with every request of that user to the server; the server then identifies the client using this.

- **How to get/create a Session Object**
  - By calling the getSession() method on the HttpServletRequest object (remember this is an implicitly available object)
  - **`HttpSession session = request.getSession()`**
- **How to destroy a SessionObject**
  - This is used to kill the user session, especially used when the end-user logs off. To invalidate the session use:
  - **`session.invalidate();`**

Other important methods defined for class Session:

· **`void setAttribute(String attributeName, Object value)`**- this method is used to store an attribute+value in session. This method takes two arguments- one is the attribute name , and the other is the value.

· **`Object getAttribute(String attributeName)`** – this method is used to get the value stored in a session under the attribute name. Remember the return type is Object. That means you can store any type of object in the session.

· **void removeAttribute(String attributeName)**- this method is used to remove the attribute from session.

· **public boolean isNew()-** This method returns true if server does not find any state of the client.

Note: Browser session and server sessions are different. Browser session is client session which starts when you open the browser and gets destroyed when closing the browser, whereas the server session is maintained at the server end.

EXAMPLE
-Create a user session when the user logs in and invalidate the session when the user logs out.

Steps:
a) create a login.jsp to grab username and password of user.
b) checkLoginDetails.jsp which will check the username and password are correct. If they are correct it will store the username in session and redirect to success.jsp.
c) success.jsp will print the username of the user stored in the session.
d) logout.jsp will call session.invalidate() to kill the server session.
e) will try to access session object after invalidate which will throw an error.

a) login.jsp

```
<!DOCTYPE html>
<html>
   <head>
      <title>Login Form</title>
   </head>
   <body>
     <form action="displayLoginDetails.jsp" method="POST">
       Username: <input type="text" name="username"/> <br/>
       Password:<input type="password" name="password"/> <br/>
       <input type="submit" value="Submit"/>
     </form>
   </body>
</html>
```

b) checkLoginDetails.jsp

```
<%@ page import ="java.sql.*" %>
<%
    String userid = request.getParameter("username");
    String pwd = request.getParameter("password");
    Class.forName("com.mysql.jdbc.Driver");
    Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/dbname","root",
"dbpass");
    Statement st = con.createStatement();
    ResultSet rs;
    rs = st.executeQuery("select * from users where username='" + userid + "' and password='" + pwd
+ "'");
    if (rs.next()) {
        session.setAttribute("user", userid); // the username will be stored in the session
        out.println("welcome " + userid);
        out.println("<a href='logout.jsp'>Log out</a>");
        response.sendRedirect("success.jsp");
    } else {
        out.println("Invalid password <a href='login.jsp'>try again</a>");
    }
%>
```

c) success.jsp

```
<%
    if ((session.getAttribute("user") == null)) {
%>
You are not logged in<br/>
<a href="login.jsp">Please Login</a>
<%} else {
%>
Welcome <%=session.getAttribute("user")%>  //this will display the username that is stored in the
session.
<a href='logout.jsp'>Log out</a>
<%
    }
%>
```

d) logout.jsp

```
<%
```

```
session.invalidate();
session.getAttribute("user");   //this will throw an error
response.sendRedirect("login.jsp");

%>
```