**High Level Description**

1. The source file primarily contains the following two files:

welcome.jsp:

A jsp page file for the site look with embedded fields for dynamic html generation

hello\_user.java:

the “EnterServlet” java servlet for processing doGet and doPost request, and generate corresponding response page according to request type. Simply open the project folder and run this servlet in Eclipse will start the application.

2. In order to run the application with .war, we upload/deploy the .war file (project1a.war) to the tomcat server. This can be either EC2 tomcat or local tomcat. After uploaded the .war file. We can then access the site from the URL “**xxxxxxxxx:8080/project1a/EnterServlet**”, where “xxxx” can be either your EC2 public DNS number if on hosted on EC2 or localhost if hosted on local machine.

**Cookie**

Cookie consists of three fields: session ID, version number, and metadata. They are separated using “:::” in between (i.e. Session ID:::version number:::metadata).

(note: metadata is not used so it is stored as string “metaData” in the cookie value string)

Example cookie string value: hdjkashdhk:::2:::metaData.

**Session State Table**

The table is implemented as HashTable in Java (synchronized preventing critical section problem)

Key (String): session ID

Value (session state object/class): session state information

The session state object/class is an inner class that encapsulates session state information including session ID, version number, message, and expiration time.

Session ID:

String type and generated by UUID class, which is global unique id that identifies a session

Version Number:

Integer number that increments every time from each request

Message:

String type and it stores the message needs to be shown in the front end. Default as “Hello User!” and then it is whatever typed by the user and replaced by the replace button.

Expiration Time:

Timestamp type which stores expiration time for server to check if session expired or not.

Each request will then generate a new expiration time (1 minute from current time) if session hasn’t expired.

**Garbage Collecting Time-out Session**

Cleaner Thread:

This thread is created when the servlet is constructed (in the constructer). It checks expired session state in the hash table (session state table) every 10 sec and delete all the sessions that have expired by checking the expiration time.

Remove in request:

In each request, the server reads the session ID from the cookie string and lookup to see if the session has expired or not (checking expiration time). If expired, it deletes from the table and creates a new page with new cookie. If not, it would process the request as normal.