**1) Where: WEBLOG.XML**

**Code:**  <context-root>/authn</context-root>

This is the website address: ie: localhost:7001/authn

**2) Where: WEB.XML (Always the Start)**

**Code:** <welcome-file-list> <welcome-file>**login.jsp**</welcome-file> </welcome-file-list>

The welcome-file-list element of web-app, is used to define a list of welcome files. Its sub element is

welcome-file that is used to define the welcome file.

A welcome file is the file that is invoked automatically by the server, if you don't specify any file name.

By default server looks for the welcome file in following order:

**3) Where: Login.jsp**

**Code:** <form action="**AuthServlet**" method="POST">

**AuthServlet.java** will be called from **login.jsp** with a request and respond via HTTPServletRequest.

**4) Where: AuthServlet.java +** *(AuthBean injected in)*

**Code:** processRequest(HttpServletRequest request, HttpServletResponse response)

abean.checkIfLoggedIn(**Authbean**) called from **AuthServlet.**java. This bean will check the status of login

whether as Admin or Non-Admin users and responds Back to **login.jsp**

**5) Where: AuthBean.java** *(injected into 3) AuthServlet.java)*

**Code:** check if user is already logged in or not using the inuse indicator.

If already logged in, close connection and print already login, reset and back to login.jsp

If not logged in, update inuse to 1 in database.

**6) Where: AuthBean.java and AuthServlet.java**

**Code:** If user not logged in already and user is not admin,

AuthBean return false back to AuthServlet (refer no 3.) : function 🡺 checkIfLoggedIn in AuthBean.java

called by AuthServlet.

**7) Where: AuthServlet.java**

**Code:** AuthServlet will then responds back to Login.asp (refer no.2) if invalid logged in. So login appears again.

If valid logged in, AuthServlet will call home.jsp to decide whether is admin or guest log in.

**8) Where: home.jsp**

**Code:** Get Data from database.

If user = non-admin, home.jpg will be displayed.

If user = admin, UserRegJSP.jsp will be called for admin to add/delete records and unlock users

in database and wait for user to click on logout by sending request to AuthServlet to

perform logout.

**MODIFICATION to store admin rights in DATABASE**

**1) Database** added one extra field: ADMININD – Number of length 1,

0: Non-admin; 1: admin

1. **Authbean.java**

Add one function:

*public boolean checkIfAdmin(String userid) throws SQLException {*

*Connection con = ds.getConnection();*

*String stmt = "SELECT ADMININD from HR.AUTH\_TBLIND where USERID='" + userid + "'";*

*// System.out.println(" CheckIf Logged In :: " + stmt);*

*Statement st = con.createStatement();*

*ResultSet rs = st.executeQuery(stmt);*

*if (rs.next()) {*

*if (rs.getInt(1) == 1) {*

*System.out.println(" User is admin :: " + rs.getInt(1));*

*con.close();*

*return true;*

*}*

*}*

*con.close();*

*return false;*

*}*

1. **Authservlet.java**

Added one logic

……

else { if (abean.checkIfAdmin(userid)) {

abean.setLoggedInStatus(userid);

sess.setAttribute("user", username);

request.getRequestDispatcher("home.jsp").forward(request, response);}

else {abean.setLoggedInStatus(userid);

sess.setAttribute("user", username);

request.getRequestDispatcher("home\_nonadmin.jsp").forward(request, response);}

}

1. Split HOME.JSP into HOME.JSP and HOME\_NONADMIN.JSP

1. Modified HTML to include Admin toggle button.