WebToDo List – Java EE

QUIBEL Florian

Summary :

1. MySql

In order to interact with MySql database, we should use the jdbc lib for java.

A class named MySqlConnector has been created. This one is connecting to the db using the db's URL and the user info. Once connected, we can add some functions to the class, that we can send request to the db.

Using jdbc lib, we send request by using 3 commands,

- Connection (connect to the db)
- Statement (Request interface for MySql, where is stored the query)
- ResultSet (List of returned rows, used only for select statement)

And we don't forget to close all those connection after using them.

So we created add, get, update, and remove function for User and ToDo tables.

2. Login

To log in a user, we need to have his username and his password. We invite him to fill those in the html form.

The Login Servlet is splitted in 2 parts, get and post function.

The get function will show up the JSP et the post function will compute response filled. We use the MySql class we've just coded to check if the username exists, then check if the password is correct.

Once logged, username and password are saved in cookies, and username and role in the current session.

Get cookies : Cookie[] cookies = request.getCookies();

Create cookie : Cookie c_uname = new Cookie("username", username);

Get current Session : HttpSession session = request.getSession();

Create session : session.setAttribute("role", role.toString());

After the first login, the cookie are used to prefill username and password for next sessions opened. Those cookie are stored in the server during a time we set.

After Login, the user is redirected to the ToDo page. If the user is a student, he will just have the list of todos. In contrary, the instructor will have access to add a todo and edit, update each todo.

To solve this differential accessibility, we use the JSP lib. This lib provides some method, as the <c:if test=””/>. So we check the user's role before putting the buttons in the page.

3. ToDo

a. Add

To add a todo, we use the function in the MySqlConnector. For simplicity, a new servlet is routed to control the add statement. Add.JSP is a simple form that register todo's description. On clicked, the add button automatically send a request to insert it inside the MySql db using the post method of the servlet.

b. Edit

Edit a todo is solved by the same way as adding. This differs by passing the id parameter inside the URL. Then we can get the full todo in MySql and edit it with an update MySql statement.

c. Remove

Remove is exactely the same a editing, we just change to a remove MySql statement.

d. List

In order to list all the todos, we get them first from MySql db. So now, we do have access to them inside the ResultSet, which we parse to an ArrayList.

We can display them with the JSP lib which provides the <c:forEach .../> method. This method takes the list as arguments, and each item in the list is generated with the proper html block inside the <c:forEach ../>. In our case, each item have his description, an edit, and remove button (if the user is an instructor).