**CollabTaskTrackerTool**

Attachments:

1. CollabTaskTrackingTool.war (it is an export of dynamic web project and includes all source code and library jars.)

2. collabTool\_db.sql (it has the DB (collabTool\_db) MySQL dump and inline are user creation commands (collabTool\_user).

CREATE USER 'collabTool\_user'@'localhost' IDENTIFIED BY 'Aastha13#';

GRANT ALL PRIVILEGES ON \* . \* TO 'collabTool\_user'@'localhost';

FLUSH PRIVILEGES;

Deployment:

* Deploy this .war file in your web container (tested in tomcat 7 with java 7 compliance)
* Import DB dump
* Create the DB user

DB schema:

It has 3 tables:

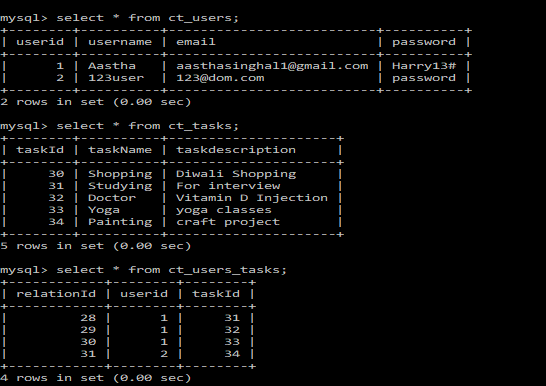
1. ct\_users -> has 2 users who can logIn

2. ct\_tasks-> tasks created by users

3. ct\_users\_tasks-> relation between tasks and users through their ids

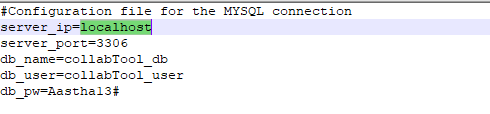
(It maintains 2 foreign keys to both task and user table and populates them as a new task gets added,

Task and user table have an auto-generating primary key.)



Assumptions:

* MySQL server is on localhost where application is deployed if not , change localhost to DB server IP in "DBParameter.conf" file loacted in ..webapps\CollabTaskTrackingTool\DB\_Configurations



* Request URL is launched from browser local to where application is deployed. Some javascript files would need a change to accommodate Application server IP if this is not the case.

Functionality and Workflow:

* User 1st request to the application goes to the login page.
* When he fills credentials, an ajax request is made to the application to validate the username/password and
  + On success -> a session gets created with user object and he is redirected to his home page
  + On failure -> failure message appears on his screen (ajax call, page is not reloaded).
* On the home page:
  + A welcome message appears with user name
  + Logout button exits to invalidate the user session
  + A Table of all tasks (created by any user is loaded)
  + There is a Refresh button: which with an ajax call gets him the all the tasks again. Tasks are sent as JSON to the callback function in the JavaScript file from where the table is regenerated dynamically.
  + There is an Add New Task button: which has some JavaScript to convert table rows into input fields.
  + A submit button: to insert the newly added task to task table in DB and a new relation for user who created task and new task in relation DB.
  + A little bit of javaScript is present here to make only one button ADD/Submit visible at a time.
  + There are Remove buttons on each task table row, they are removing rows from the frontend only, task still remains intact in DB.