Homework 5: Databases

For this homework, you will practice working with databases by adding a database backend to the web app that you created in the last homework.

You will do this homework as a team; however, each member of your team will be responsible for the completion of a particular task.

Step 1. Add database backend to GameDen

You will continue working on the **GameDen** web app by adding MySQL-based persistence to the dynamic web pages.

Each team member must complete one of the tasks defined below.

NOTE: Keep in mind that this version of **GameDen** only supports one user; however, next homework will allow for multiple users.

For an example of a Java EE web app with a database backend, checkout this example:

https://utopia.cs.memphis.edu/course/comp7012-2013spring/examples/BookCatalog/trunk/

Don't forget about to add a **mysql-connector-java** jar file to your **WebContent/WEB-INF/lib/** folder.

Step 2. Submit (by tagging) your team's submission

The following instructions are essentially the same as last time; only the tag name has changed.

Attention! Before performing this step, you must make sure that all team members have committed their edits to the **trunk** in the repository.

Only one team member (the leader) performs the following.

First, you must fill out the **README.txt** file in your project's **trunk**. The file should list which team member performed each task (one team member per task).

To submit work in this course, you must tag it. Then, I will checkout the revision that you tagged and grade it. By tagging, you tell me that you are done, and this is the version you want me to grade.

The tag you must use for this homework is **hw5** (case sensitive, no spaces).

To tag the current revision of your trunk as **hw5**, do as follows:

1. Go to the **SVN Repository Exploring** perspective in Eclipse.
2. In the **SVN Repositories** view, find the **trunk** folder that you want to tag.

3. Right-click on the **trunk** folder, and click **Show History**. This should open the **History** view with a table listing the past commits to the **trunk**.
4. In the History table, right-click the newest revision (i.e., the one with the greatest revision number), and click **Tag from…** This should open a **Create Tag** dialog.
5. Enter **hw5** into the **Tag** field and optionally enter a log comment, then click **OK**. This should create the tag!

To verify that tagging was successful, open the following URL in a web browser (replacing *YOUR_TEAM* with the appropriate name):

https://utopia.cs.memphis.edu/course/comp7012-2013spring/teams/*YOUR_TEAM*/GameDen/tags/

You should see an **hw5** folder, and within that folder should be **src** and **WebContent** folders along with the **README.txt** file. Everyone's HTML files should be in the **WebContent** folder.

The Tasks

Task 1: User Info Form

- The dynamic web page must behave as specified in last homework, except it must store/retrieve all data from a database (and not from memory).

Task 2: Blackjack Game

- The dynamic web page must retrieve the user's name from the database and display it at the top of the page. If no name is defined, then print "Guest".
- The dynamic web page must store the number of Blackjack games that the user has won and lost in the database. It must display these stats at the top of the page next to the user's name.
- The dynamic web page must have a "Reset game data" button that clears the user's stored data for the game.

Task 3: Minefield Game

- The dynamic web page must retrieve the user's name from the database and display it at the top of the page. If no name is defined, then print "Guest".
- The dynamic web page must store the number of Minefield games that the user has won and lost in the database. It must display these stats at the top of the page next to the user's name.
- The dynamic web page must have a "Reset game data" button that clears the user's stored data for the game.

Task 4: Tic-Tac-Toe Game

- The dynamic web page must retrieve the user's name from the database and display it at the top of the page. If no name is defined, then print "Guest".
- The dynamic web page must store the number of Tic-Tac-Toe games that the user has won and lost in the database. It must display these stats at the top of the page next to the user's name.
- The dynamic web page must have a "Reset game data" button that clears the user's stored data for the game.

Task 5: Yahtzee Game

- The dynamic web page must retrieve the user's name from the database and display it at the top of the page. If no name is defined, then print "Guest".
- The dynamic web page must, after the user has rolled three times, print the score for the user's game at the bottom of the page. Score games as follows:
    - If all dice have the same number, score = 50 points.
    - If the dice produce either of these "straight" sequences, 1-2-3-4-5 or 2-3-4-5-6, score = 40 points.
    - If neither of the above is true, then score = sum of dice.
- The dynamic web page must store all the scores from the user's completed Yahtzee games in the database. It must display the user's high score and average score at the top of the page next to the user's name.
- The dynamic web page must have a "Reset game data" button that clears the user's stored data for the game.

Task 6: Battleship Game

- The dynamic web page must retrieve the user's name from the database and display it at the top of the page. If no name is defined, then print "Guest".
- For each completed game of Battleship, the dynamic web page must store the number of turns it took the user in the database. It must display the user's lowest turn count for a game and average turn count at the top of the page next to the user's name.
- The dynamic web page must have a "Reset game data" button that clears the user's stored data for the game.