CS 346 – Team Project

Due: Sunday 12/10/2017 at 8:00PM

100 points [counting for 25% of your course grade]

*This is a **team** project*

The goal of this team project is to build a fully functional, secure, and appealing WebCLICKER app. Here, as a reminder, is a list of the required functional features of the web app that you must build over the next five weeks.

Student side:

- Password-protected log in and access to all (and only) student functions; student-initiated password change; student-initiated password reset used when the current password is forgotten.
- Request the active question; display the active question (or an error message); submit student's answer to the server.
- Search for past (i.e., already deactivated) questions; this feature must include an advanced search feature that allows students to filter questions using all of the question characteristics (see below) in all acceptable combinations (e.g., all questions about CSS that were worth more than 2 points and for which partial but not full credit was received).
- Explicit log out (to clear session data).

Instructor side:

- Password-protected log in and access to all (and only) instructor functions; instructor-initiated password change; instructor-initiated password reset used when the current password is forgotten.
- Insert a new question into the database (see below for a list of required question features).
- Edit and save any one or more features of an existing question.
- Delete an existing question.
- Select and activate one question at a time; collect students' answers and display them (to the instructor only) as they come in using a bar chart, as well as a timer for the activation time.
- Deactivate the current question; store statistics about the question (date, start time, end time, average score, number of correct answers) in the database.
- Deactivate all questions (to deal with questions that were left active by mistake or that were activated too early).
- Display the scores for all students and all questions asked on a given day.
- Explicit log out (to clear session data).

Note that the list above does not include the ability for the instructor to update his or her class roster. You can assume that the student database will be created/updated automatically for the instructor at the beginning of the semester.

In order to help you implement the required forms and the back-end database tables, here are the minimum lists of data items that you will have to store:

- **student**: unique ID, username, first and last names, email address, hashed password, number of password changes, day and time of last login and logout
- **instructor**: same as above

- **question**: id, status (e.g., already activated or not, draft mode, etc.), question statement (including text box for the answer, selectable options, etc.), correct answer, number of points, textual description of the question topic, topic keywords, section number of topic in textbook, PHP grader code, number of correct answers, average points earned by the class, start and end time/day of the question's activation interval

You will also need to store, for each question and for each student, his or her answer and number of points earned on each question.

Here is a suggested breakdown of the project into tasks and a reasonable order for them:

1. Complete the design stages for your database, as discussed in class and in Appendix B of our textbook.
2. Finalize your schema and implement it in your *team<i>* database on the MySQL server running on *webdev*.
3. Implement the PHP code that performs the processing for all of the forms needed for this app, which you built in the last component of the individual project.
4. Finalize your user interface and connect it to your PHP code, using responsive design to style your app for two screen sizes, namely wide screens and narrow screens (say, those fewer than 700 pixels).
5. Add session management and authentication to your app, making sure that your app is secure (e.g, students must not be able to access any instructor features; people other than students and the instructor must not be able to go past the login page).
6. Use Ajax and the canvas API to implement the dynamic bar chart of student answers during question-activation intervals.
7. Perform in-depth integration testing at this stage, following the functional testing that you did continuously at the previous stages.

You are expected to make progress on this project each and every week starting now (see below for assessment details). At this point in the semester, we have covered all that you need to complete tasks 1 through 4 above, except for the responsive design component in task 4. We will cover this topic as well as what you need for the following tasks starting next week. You are expected to keep your progress on this project in sync with the material discussed in class each week. This way you will not have to cram a lot of work into the last week. In fact, the week of December 4 should be almost entirely devoted to task 7.

Assessment

This project will be assessed along two dimensions. First, a portion of your grade will be identical for all team members and will be assigned based on the usability, functionality, and security characteristics of the finished product as determined by the final report produced by the team, the quality of the demo given to the class, and the outcome of the code review and extensive testing performed by the instructor. Second, the rest (and a significant portion) of your grade will be assigned individually based on your performance as a team member, as determined by your self-evaluations, evaluations of your team mates, and the consolidation of these two factors performed by the instructor, e.g., as a function of the (mis)match between your self-evaluation and the evaluations given by your team mates. All team members must contribute significantly to the project each week. More precisely, you will be required to submit a self-evaluation and an evaluation of each one of your team mates through a form that will be made available to you later this week, before 8:00PM on the following Sundays: 11/12, 11/19, 12/03, and 12/10. Failure to submit a full set of evaluations on each one of these dates will be penalized in the individual component of your grade.

Your goal for this project is to carefully design and implement your app. Of course, you must make it as usable as possible, in the precise sense that we discussed in class. Here are some additional requirements for this team project:

- You must follow the principles for good web design discussed in class and in our textbook. For example, the “look and feel” of your app must be consistent throughout. At the same time, there should be a clearly visible distinction (your choice!) between the student and instructor sides.

- There is no minimum or maximum number of pages for your app, as long as all required features are present.
- Your database must contain at least seven questions. If you pick questions that were submitted as part of the individual project, make sure to fix all of the errors that were present then (and do not introduce any other ones!).
- Your database must contain one instructor account and at least two student accounts with usernames and passwords that are included at the very top of your project report (for a total of three sets of credentials). Make sure that the first student has scores recorded for all questions but that the second student has not answered any of the questions yet. Choose your questions and student answers judiciously so that you can execute an interesting/complex search query during the in-class demo. I will use the credentials of the second student during my testing of your app.
- Your site must only contain valid HTML (both statically and dynamically generated) markup, valid CSS rules, and professional quality PHP and JavaScript code. All pages must contain two self-verifying links to the W3C validators for HTML5 and CSS. You may not use any client-side libraries or frameworks unless we have discussed them in class (e.g., jQuery will be allowed if we get to it before the due date for this project). You may not use the Prototype or Scriptaculous libraries that the book discusses.
- Your app must have an appropriate navigation structure, with headers, footers, and a navigation “bar”.
- There must be a single entry point to your web app to be used by all users (students or instructor).
- Your code must be structured in a well-designed directory structure. Your submission may NOT include unused directories, files, or lines of code. Delete everything that does not belong in a professional-quality deliverable.
- Make sure that your web site works as expected and looks as you want it to look in the latest version of the Chrome browser in which all grading will be done. Your web app must work on the *webdev* server. Make sure to test it fully there before submitting it.

In-class Demo

Each team will have 15 in-class minutes on Monday 12/11/17 or Wednesday 12/13/17 to demo their app to me and the rest of the class. Since we are all working on the same project, you will not need to introduce the app itself; the whole time slot should be devoted to a detailed demo of your app during which you must illustrate (at least), and in this order, the following use cases:

1. The instructor tries to log in but enters the wrong username and/or password.
2. The instructor successfully logs in, changes his or her password, and logs out.
3. The instructor uses the new password to log in, then creates a new question, and inserts it into the database.
4. The instructor modifies the previously created question and then activates it (keeping it activated for a forthcoming student use case).
5. A student tries to log in but enters the wrong username and/or password.
6. The same student successfully logs in, changes his or her password, and logs out.
7. The same student uses the new password to log in, then asks for the currently active question, submits his or her INcorrect answer to the question, and finally receives feedback on it from the automated grader.
8. The instructor observes the new answer coming in (in the bar chart), then deactivates the question.

9. The same student tries to activate the next question but is told that no question is currently active.
10. The student performs an interesting search query and then logs out.

Make sure to fully prepare your demo and to rehearse it under time constraints so that you are able to complete it in no more than 15 minutes. You will be cut off after 15 minutes even if you are not done with the demo. Incomplete demos will be penalized in the overall grade. All team members must present parts of the demo of approximately equal durations.

Submission procedure

1. Remember to fill out the evaluation form before 8:00PM on the four Sundays listed above.

2. Use your team account to upload your app to the *webdev* server and make sure that it works properly for any user who is simply given the URL of its entry point.

3. Before 8:00PM on the due date, submit to the D2L dropbox for *Pteam* a zip file called `team`*`i`*`.zip` (where *`i`* is replaced with your team number, between 1 and 6) containing the entirety of your source code (EXACTLY as it appears on the *webdev* server), including all `.html`, `.css`, `.js`, and `.php` files as well as all needed assets (images, etc.), if any. I must be able to unzip this file, copy it to my local web server, and run it as is in my browser. Make sure to include, at the top level of your submission directory, a TEXT file called `webclicker.mysql` (generated by *phpmyadmin* and then renamed by you) that contains a script for recreating your entire, populated database, in case of server crash as well as for deployment/grading purposes. ***Do not forget to include in your dropbox submission comment a working link to the entry point of your web app on webdev***.

4. Before 8:00PM on the due date, send me (as an attachment) a project report (as a single Word, OpenOffice or PDF document) that describes in no more than 3 pages, the main challenges you encountered during this project, how you met these challenges as a team, as well as the parts of your deliverable that you are the most proud of (and why) and the known bugs or issues that remain. If you know of no bugs or issues in your app, please say so. It is to your advantage to be honest about this, since explicitly acknowledged bugs will cost you fewer points than issues I discover on my own during testing.

If any one of the components (the app, the zip file, the report, the MySQL script, the demo, etc.) is late, then the late-submission penalty will apply to the overall team project grade for all team members.

Reminder: Failure to meet the requirements stated in this document will result in point deductions.