CSCI 4000 Assignment 3

Important: For this assignment, you must use **PDO** (PHP Data Object) to interface with the MySQL database. If you do not use PDO, you will receive 0 points for almost all questions.

Q1: **Registration** [40 points] – PHP, MySQL, phpMyAdmin
Create a MySQL database.

- Create a MySQL database `<my name>AssignmentDB`.
- Create a database table `<my name>student` in the created database.
- Create the following fields (columns) for the created table: `name, email, major`. Choose the appropriate date types for the fields. (Note, you may also need to create a primary key field, e.g. `id`, for Q3 Delete)
- Create a MySQL database username "`<my name>web`" with password "`chocolate`", with data privileges (select, insert, update, delete, file).
- Do screen captures of the database structure, table structure and privileges for submission. (You may use phpMyAdmin or any other tool.)

Create a simple registration form, similar to what you might encounter when registering for an online Web site.

- In the form, add name, e-mail address and college major. Include default text in the name, e-mail and college major text boxes.
- Use html table (or css) to format the input fields (into rows and columns).
- Add Submit button. Save the document as `<my name>Registration.php` (page title `<my name>Registration`).
- Submit the form to the `<my name>RegnInsert.php` file (page title `<my name> RegnInsert`). `<my name>RegnInsert.php` should use the $_POST array to get all the form fields/inputs (name, e-mail, college major), and display them.
- `<my name>RegnInsert.php` should also connect to the `<my name>AssignmentDB` MySQL database, using the MySQL database username "`<my name>web`" and password "`chocolate`"; and insert the inputs to the `<my name>student` database table (`name, email, major` fields) mentioned above .
- Use an external *.css file to format both .php files. Create your own *.css file. Do NOT use the provided *.css file in D2L. No two students should use the same *.css file (**both will get 0 for this question**).

Important: If you do not put `<my name>` to the above mentioned fields (database related field, page title and filename), **you will get 0 point for the question.**

Estimated time: 4 hours

Q2: **Display** [20 points] - PHP, MySQL, phpMyAdmin
Write a PHP web page (program) that connect to the `<my name>AssignmentDB` MySQL database, using the MySQL database username "`<my name>web`" and password "`chocolate`", and retrieve all the data from the `<my name>student` database table (inserted in Q1).

- Use html table (or css) to format the output table (into rows and columns).
- Save the document as `<my name>Display.php` (page title `<my name> Display`).
- Use an external *.css file to format the .php file. Create your own *.css file. Do NOT use the provided *.css file in D2L. No two students should use the same *.css file (**both will get 0 for this question**).

Important: If you do not put `<my name>` to the above mentioned fields (database related field, page title and filename), **you will get 0 point for the question.**

Estimated time: 2 hours

Q3: **Delete** [30 points] - PHP, MySQL, phpMyAdmin
Write a PHP web page (program) that connect to the `<my name>AssignmentDB` MySQL database, using the MySQL database username "`<my name>web`" and password "`chocolate`", and retrieve all the data from the `<my name>student` database table (inserted in Q1).

- Display a "delete" link or button next to each data item (each row of data).
- Save the document as `<my name>Delete.php` (page title `<my name> Delete`).
- Use an external *.css file to format the .php file. Create your own *.css file. Do NOT use the provided *.css file in D2L. No two students should use the same *.css file (**both will get 0 for this question**).
- When the "delete" link or button is clicked, the corresponding row of data should be removed from the `<my name>student` database table. (Note, you would need to create additional *.php file(s) to do the deletion.)
- Automatically refresh the `<my name>Delete.php` page. The deleted data item (row of data) should disappear.
- Use html table (or css) to format the output table (into rows and columns).

Important: If you do not put `<my name>` to the above mentioned fields (database related field, page title and filename), **you will get 0 point for the question.**

Estimated time: 2 hours

Q4: **Home** [10 points] - HTML, CSS
Write a PHP web page (program).

- Save the document as `<my name>Home.php` (page title `<my name> Home`).
- Create links to the following pages created in Q1, 2 & 3.
  - `<my name>Registration.php`
  - `<my name>Display.php`
  - `<my name>Delete.php`
- Use an external *.css file to format the .php file. Create your own *.css file. Do NOT use the provided *.css file in D2L. No two students should use the same *.css file (**both will get 0 for this question**).

Submission instructions:
You need to test the above programs (questions) separately, and provide **two test cases** (if applicable) for each program (question). Do a screen capture of the input and related output for each test case. Use any graphic editing software (e.g. Microsoft Paint, Adobe Fireworks, GIMP) to cut out the program input and output (from the screen capture), paste them into a word document under a related question number, save the document as a pdf file.
You only need to do **one set** of screen captures of the **database structure, table structure and privileges**.

You need to submit the following:
1. A pdf file containing the screen captures of program input and output of all test cases, name the file **lastname_firstname_assignment03.pdf**.
2. All php files. Zip your files into a single zip file (or rar file) **lastname_firstname_assignment03.zip**. The zip file should contain:

- `<my name>Home.php`
- `<my name>Registration.php`
- `<my name>RegnInsert.php`
- `<my name>Display.php`
- `<my name>Delete.php` (and other related deletion files)
- The related *.css file(s)

Please submit electronic copy (the above mentioned **two files**) to D2L digital dropbox.

Grading guidelines (programming questions):
Your programs will be judged on several criteria, which are shown below.

- Correctness (50%): Does the program compile (run) correctly? Does the program do what it's supposed to do?
- Design (20%): Are operations broken down in a reasonable way (e.g. classes and methods)?
- Style (10%): Is the program **indented** properly? Do variables have **meaningful names**?
- Robustness (10%): Does the program handle erroneous or unexpected input gracefully?
- Documentation (10%): Do all program files begin with a **comment** that identifies the author, the course code, and the program date? Are all the classes, methods and data fields clearly **documented (comments)?** Are unclear parts of code **documented (comments)?** (Some items mentioned may not apply to some languages)

A program that does not compile (run) will get at most **50% of the possible points**.