**CST8257 Web Application Development**

Lab 7

# Objective

1. Accessing database in PHP.

# Due Date

See Canvas post for the due of this lab. To earn 10 points, you are required:

1. Complete the lab as required.
2. Submit your lab work to Canvas before the due date.

1. Demonstrate your lab work during the following week’s lab session.

# Requirements

1. Follow the Power Point Slide, **4. Manage MySQL Database.ppt** to create a Database, **CST8257**, on your MySQL Data Server.
2. Create the following tables in database CST8257:

**Student**

StudentId, varchar(16), NOT NULL, PRIMARY KEY

Name, varchar(256), NOT NULL

Phone, varchar(16)

Password, varchar(256)

**Course**

CourseCode, varchar(10), NOT NULL, PRIMARY KEY

Title, varchar(256), NOT NULL

WeeklyHours, int, NOT NULL

**Semester**

SemesterCode, varchar(10), NOT NULL, PRIMARY KEY

Term, varchar(10), NOT NULL

Year, int, NOT NULL

**CourseOffer**

CourseCode, varchar(10), NOT NULL, PRIMARY KEY

SemesterCode, varchar(10), NOT NULL, PRIMARY KEY

FOREIGN KEY (SemesterCode) REFERENCES Semester(SemesterCode),

FOREIGN KEY (CourseCode) REFERENCES Course(CourseCode)

**Registration**

StudentId, varchar(16), NOT NULL, PRIMARY KEY

CourseCode, varchar(10), NOT NULL, PRIMARY KEY

SemesterCode, varchar(10), NOT NULL

FOREIGN KEY (StudentId) REFERENCES Student(StudentId),

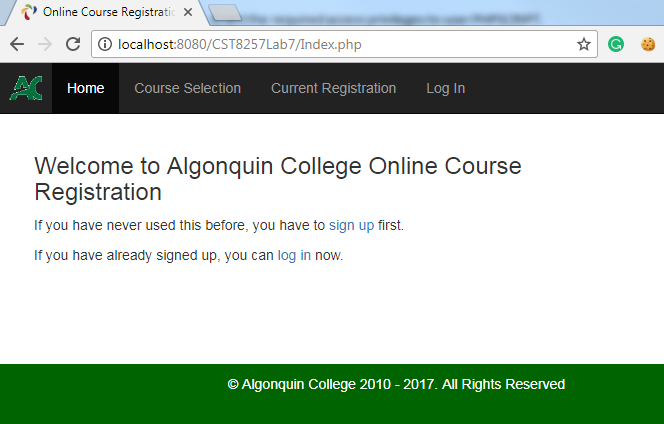
FOREIGN KEY (CourseCode) REFERENCES CourseOffer (CourseCode)

FOREIGN KEY (SemesterCode) REFERENCES CourseOffer (SemesterCode)

1. Run SQL statements in file “**Script for Populating Lab 7 Database.sql**” (downloadable from Canvas) against database CST8257 to populate Course table and CourseOffer table.
2. Create a new user “PHPSCRIPT” with password “1234” for use by PHP scripts to access CST8257 database.
3. Grant the required access privileges to user PHPSCRIPT.
4. Develop an online course registration web application which is supported at the backend by a MySQL database. The application should have following pages and functionality.

* **Landing (or Default) Page**

Create a **Index.php** page as shown below:



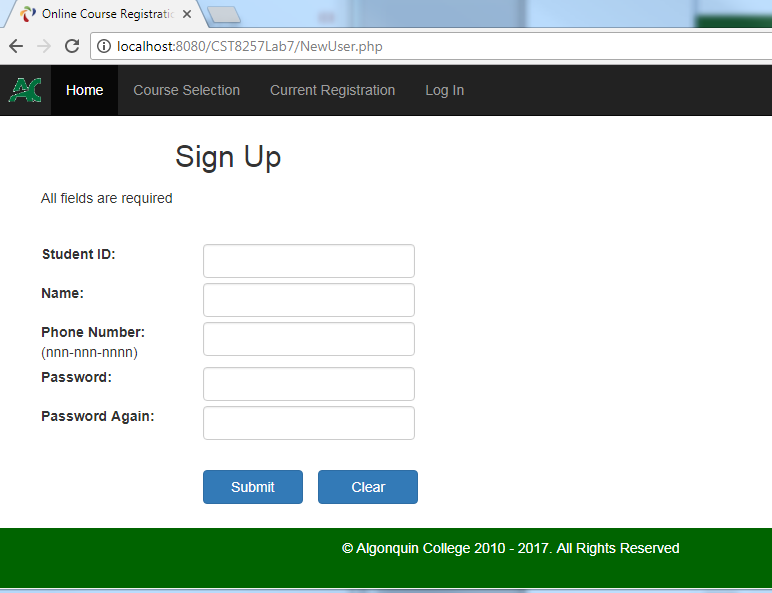
This page should contain two links:

sign up – On click, brings the user to the **NewUser.php** page

log in – On click, brings the user to the Login.php page

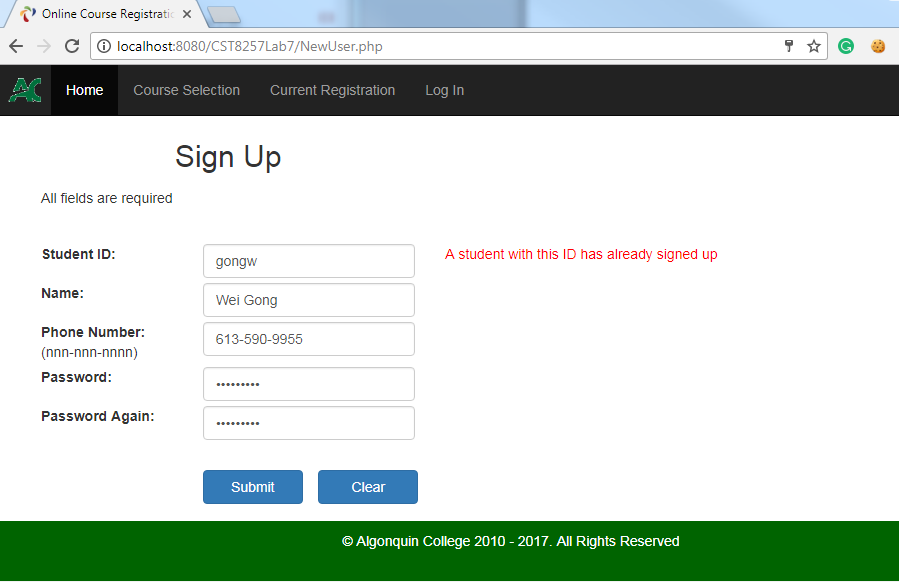
* **NewUser.php**

For students never used this application, this page is for them to sign up:



After the user clicks **Submit** button, the Web App should perform following tasks:

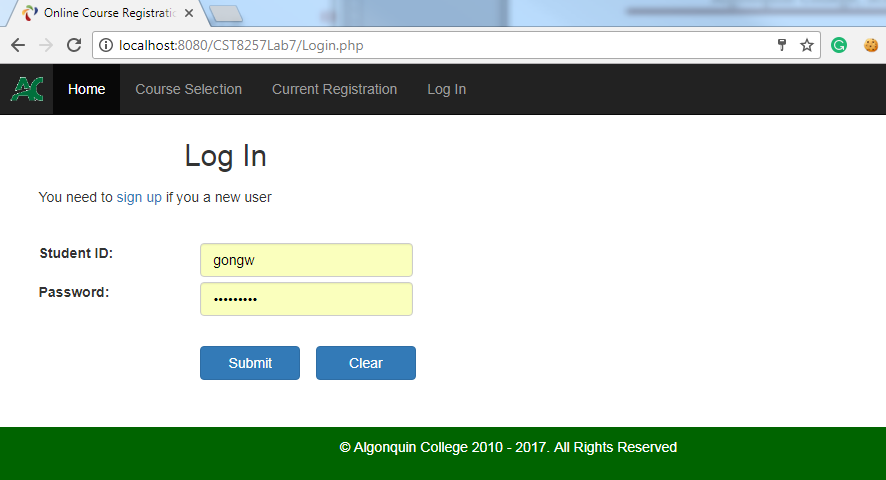
1. Validate that data entered satisfy the following requirements:
   1. Student ID is not blank.
   2. Name is not blank.
   3. Phone Number is in the format of nnn-nnn-nnnn
   4. Password is at least 6 characters long, contains at least one upper case, one lowercase and one digit.
2. If there is any field with invalid data, display an error message on the page next to the data field. The user entered data should be preserved.
3. If the entered Student ID already exists in the system, display an error message next to the Student ID field, as shown below:



1. If all data entered are valid, save the data into the **Student** table.
2. Redirection the user to the **CourseSelection.php** page.

* **Login.php**

For those students who have been signed up, you should provide a login page:

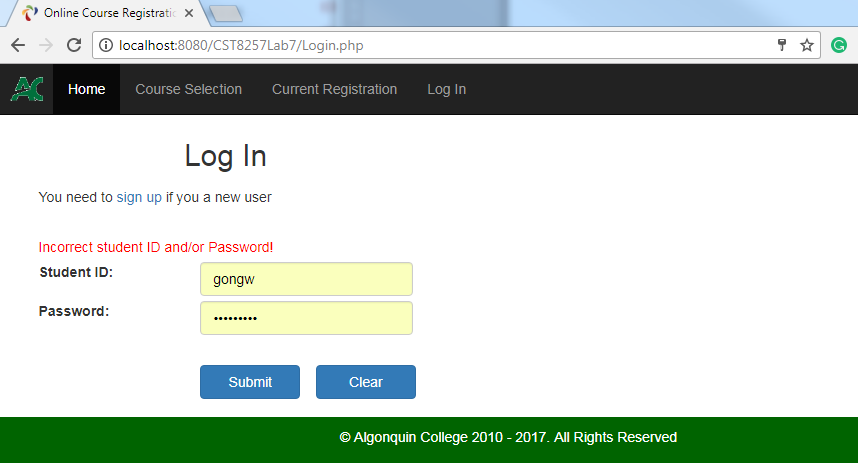


On this page, a user will log in by entering his student ID and the Password he/she created during signing up.

1. After the user clicks “Login” button, the Application validate that
2. Student ID is not blank.
3. Password is not blank.

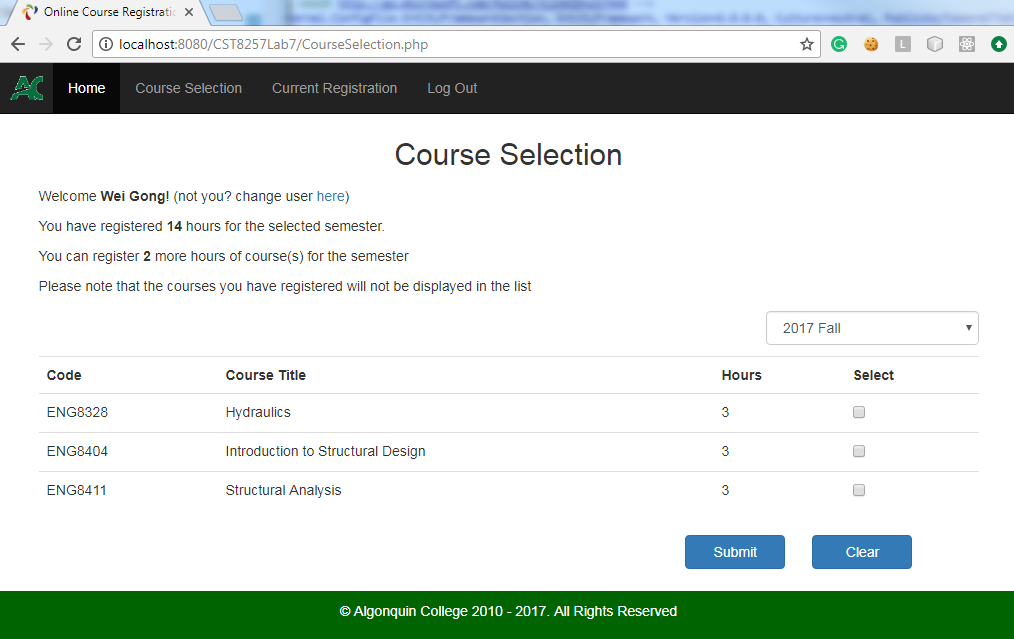
If a data entered fails the validation, an error should be displayed beside the field.

1. If entered Student ID and Password pass the validation, the application checks the entered Student ID and Password pair against the records in the **Student**. If no match found, an error message should be displayed on the page:



* **Course Selection Page**

After successful login, the user will be presented with a page listing all courses offer in the semester specified by a dropdown list as shown below:



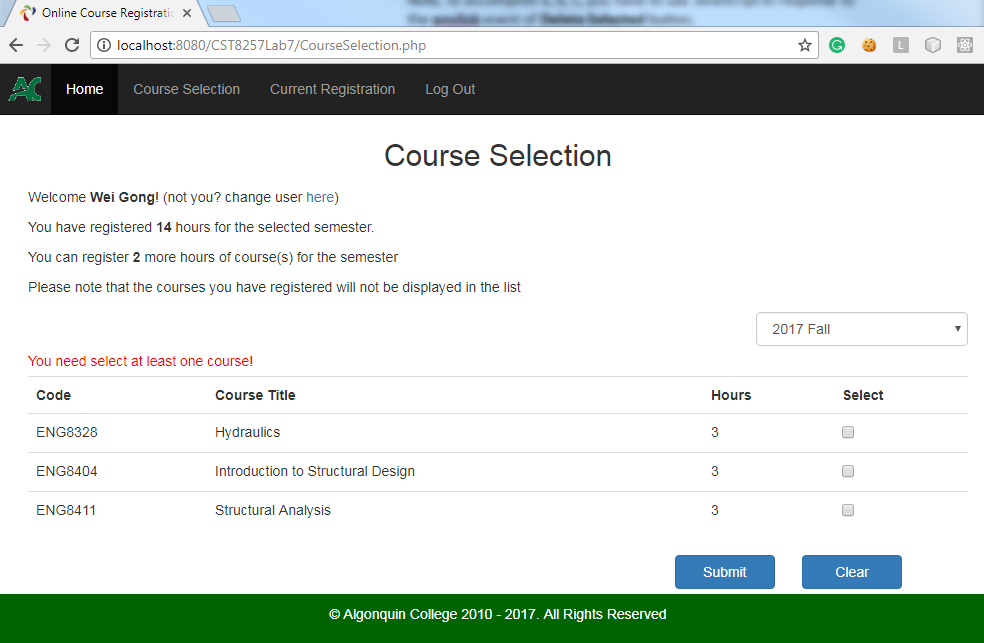
1. The Semester in dropdown list on the page should be obtained from the **Semester** database table.

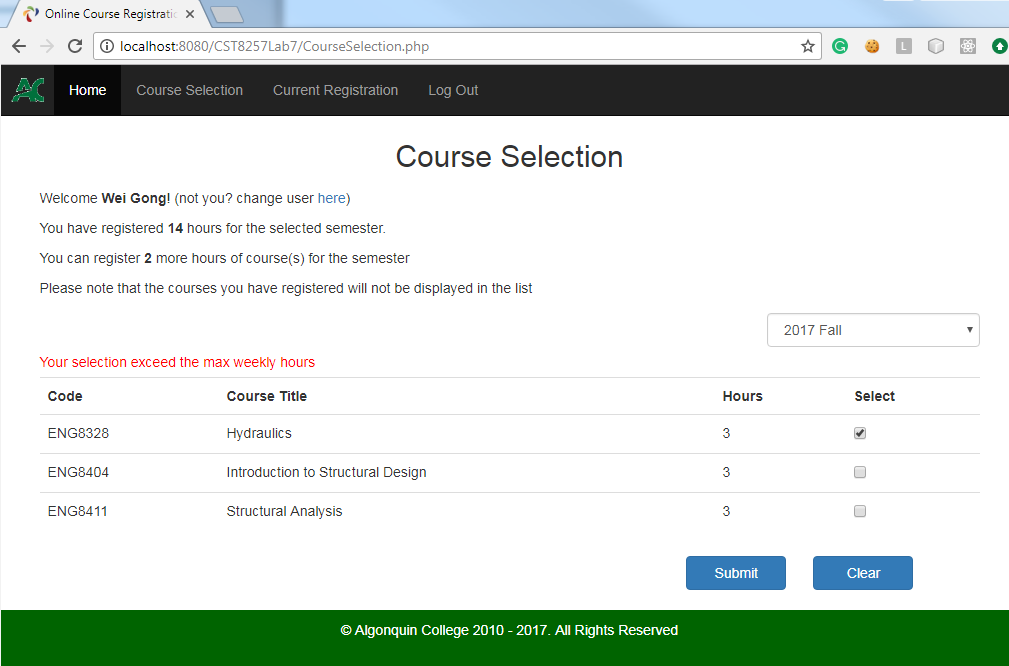
1. The course **Course** list on the page for the selected semester should be obtained from the and **CourseOffer** table.
2. The displayed course list should exclude those courses which the user has already registered.
3. When the user selects a different semester in the dropdown list, the course list will be updated with the courses offered in the selected semester.

Note: You will need to use JavaScript to response to the **onchange** event of the dropdown list and submit the form.

1. The page should always display the number of weekly hours the user has registered for the semester and the remaining number of weekly hours the user can register for the semester.
2. Upon the **Submit** button clicked, the application validates:
   1. The user selected at least one course.
   2. The total number of weekly hours of the registered courses for the semester does not exceed 16 hours.

If the validation fails, an appropriate error message should be displayed on the page.

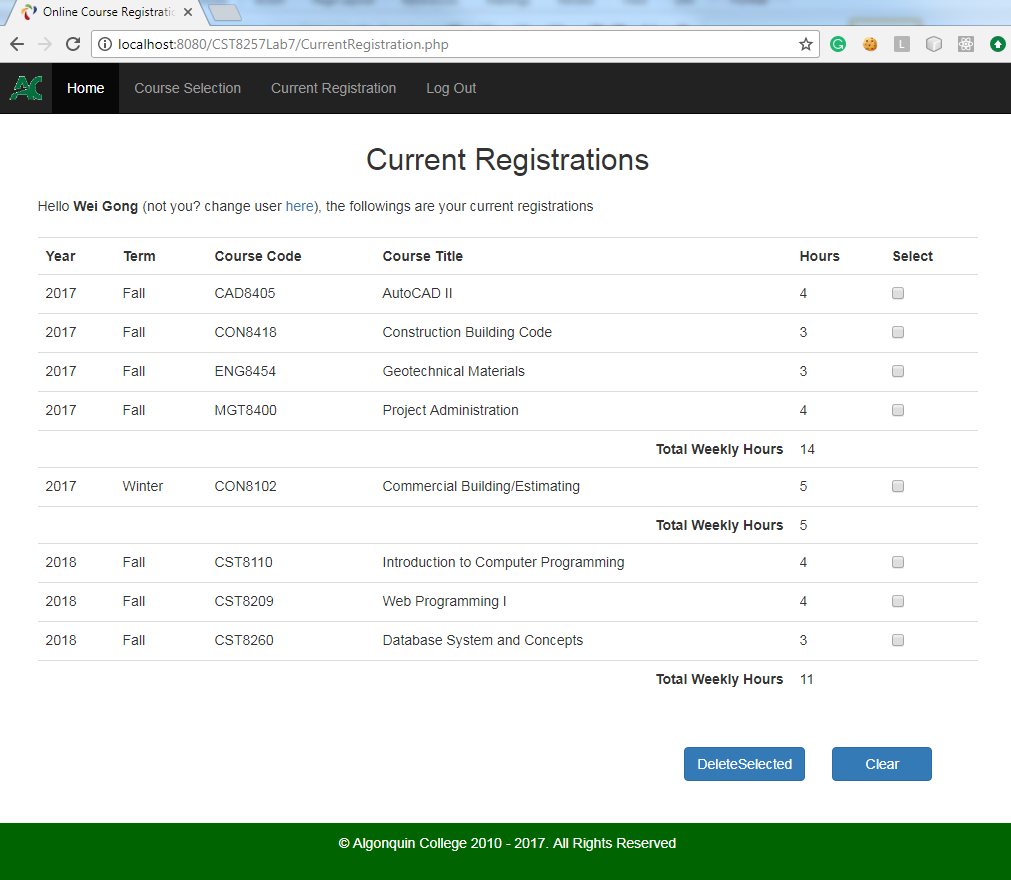




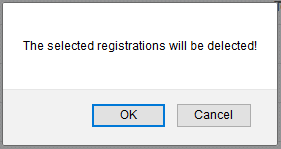
1. If the user selected course(s) pass the validation, application will:
   1. Add the course registration to the **Registration** table.
   2. Re-display the page with the newly registered course(s) removed from the available course selection list.
   3. Update the information on page showing the new total number of weekly hours and remaining number of hours.

* **Current Registration Page.**

This page, CurrentRegistration.php show all courses the user has registered as shown below:



1. On this page, the user can select one or more registered courses and click **Delete Selected** button to delete the registration. The application should:
   1. Prompt the user to confirm that the selected registrations will be deleted as:



* 1. If the user clicks **Cancel** button, the action will be canceled and no registration will be deleted.
  2. If the user click **OK** button, the selected registration will be deleted from the **Registration** table.

Note, to accomplish a, b, c, you have to use JavaScript to response to the **onclick** event of **Delete Selected** button.

* 1. The page will be displayed again showing the user’s remaining registrations
* **Page header**

As shown in the screen captures, all pages should have a standard page header which contains the following menu items:

* 1. **Home** – The link to **Index.php** page
  2. **Course Selection** – The link to **CourseSelection.php** page. If the user has not logged in yet when clicking on this link, he/she will be redirected to the login page for logging in. Once successfully authenticated, the user will be redirected to **CourseSelection.php.**
  3. **Current Registration** – The link to **CurrentRegistration.php** page. If the user has not logged in yet when clicking on this link, he/she will be redirected to the login page for logging in. Once successfully authenticated, the user will be redirected to **CurrentRegistration.php.**
  4. **Log In** / **Log Out** –If the user is currently not logged in, the menu is for logging in and the link to **Login.php** page. If the user is currently logged in, the menu is for logging out and the link to **logout.php**

The **logout.php** page should destroy the current session and then redirect the user to **Index.php** page.