FWD PHP PROJECT

OBJECTIVES

Write a secure database administration interface.

DUE DATE

Assigned during session11, due 11:59pm the night **after** session12, or as otherwise specified by your instructor.

REQUIREMENTS

You may complete this assignment with a partner or on your own. If you team up with someone, there must be a prominent heading on the main page that includes both of your names.

You will write a web application that allows a user to administer the students table on the **bcit** database. The user will need to be able to view the current state of the table, as well as add, delete or edit a record.

Scripts should use **dbinfo.php** credentials anytime they need to connect to the database.

index.php: Write a main page PHP page that displays the entire students table as an HTML table, one record per row. Include an **'Add Student'** link somewhere near the table. Include two additional data cells per row, one containing a **'Delete'** and the other an **'Update'** link. The screenshots with this assignment provide an example of how it might look.

The **Update** and **Delete** links should send record-specific information via a $_GET query string to the script(s) responsible for updating and deleting from the database. See the **sort-records-after.php** file from session 11 for an example of this technique.

Database Administration Functionality

Write script(s) for handling each basic database operation:

Add Student: Display a form with inputs for Student Number, First and Last names, and a Submit button.

Delete: Display the record information of the record they chose to delete. Give them one last chance to change their mind, eg: ask the user if they are sure they want to delete the record, and display a form with a radio button option of ‘yes’ and ‘no’.

Update: Display a form with fields for Student Number, First and Last names. Pre-populate these fields with the current record data. Add a Submit button.

Script Behavior

Regardless of the database operation performed, be sure to forward the user back to the page that displays all table data, so that they can see the latest table status.

Always provide feedback to the user as they interact with your scripts. Each time the user sees the **index.php** page, display positive feedback, such as: 'A new record has been added to the table' or, when appropriate, display descriptive error messages, eg: 'The record could not be updated as requested.'

Thoroughly validate all form data and protect against SQL injection attacks.

HINTS

Build your scripts slowly and test thoroughly as you go. Try to break the problem down into smaller parts, and solve each part before proceeding to the next. For example, first deal with displaying the student records in a table, after completing that functionality, deal with one of the operations, eg: ‘Add a student’ and complete that before dealing with the ‘Delete’ or ‘Update’ features.

Use the **php_project_flowchart.jpeg** flowchart file as a guide for how to split up the tasks and sequencing.

Use sessions wherever possible to remember important data across multiple pages.

SUBMISSION

Before the due date, compress all assignment resources (HTML, CSS, PHP, images, folders, etc) into a single **.ZIP**. Upload **.ZIP** to

learn.bcit.ca > Web Scripting 2 > Content > Session12 > Project Dropbox

If completed with a partner, each of you should upload a copy of the code to the Dropbox.