

---

## CSCI 4000 Assignment 3

**Total estimated time** for this assignment: **12 hours** (if you are a good programmer)

The main programming language for this assignment should be **PHP** (mixed with HTML and CSS). If you do not use **PHP**, you will get **0 points**. 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.

When you see “Richard Ricardo” or “richardricardo” in the examples and screen captures, change it to **<your name>**.

When you see “Richard” or “richard” in the example screen captures, change it to **<your first name>**.

If you do not put **<your name>** / **<your first name>** in the above mentioned fields, you will get **0 points** for the question(s).

**No two students** should submit webpages with exactly the same code, content, layout, or color combination. If found, **both** students will get **0 points**.

Please change the provided example’s layout and color combination (color scheme). **If you use (copy) ANY of the provide example’s color combination, 10 points will be deducted (-10 points).**

Create a folder on your hard disk, name the folder **lastname\_firstname\_assignment3**. Save all the files from this assignment in this folder.

Use XAMPP web server solution stack package to help debugging PHP code. It will make your debugging process easier. All php files must not produce any error, or any warning (**-2 points for each error, each warning**). Your program must run. A program that does not run will get at most **50% of the possible points**. All files must begin with a **comment** that identifies the author, the course code, and the program date (**- 2 points each question** if found missing). All html, css and php files must be clearly **documented (commented)**. Points will be taken off (**-2 points each question**) for insufficient comments (`<!-- -->`, `/* */`, `//`).

- When you view page source in a web browser, **<!DOCTYPE html>** must be at the top of every page. In other words, all pages must be written in HTML5. (**-20 points** if not)
  - You **can** put php code before **<!DOCTYPE html>**.
  - You **cannot** put html code before **<!DOCTYPE html>**.
- Before adding PHP code, all html files must pass html validation at <http://validator.w3.org/> without any **error** (and with only 1 warning).
- After adding PHP code, the generated html code (Firefox web browser > right-click > view page source) must also pass html validation at <http://validator.w3.org/> without any **error** (and with only 1 warning).
- All css files must pass css validation at <http://jigsaw.w3.org/css-validator/> without any **error**. (**-2 points for each error/warning**, only 1 warning is allowed for html validator)

**Question 1 – Database:** PHP Chapter 4, eg008 and knowledge of SQL (20 points) **Estimated time: 3 hours**

- Save question 1 files in folder “**lastname\_firstname\_assignment3**”: (2 points)
  - **create\_db.sql**
- Create a text file **create\_db.sql**, write sql statements in the file to
  - Create a MySQL database **richard\_ricardo\_assignment\_db**. (2 points)
  - In the database, create 1 table.
    - student (2 points)
  - Create the following fields (columns) for the table (refer to examples below for details).
    - student table: studentID, name, email (4 points)
    - studentID is the primary key of the student table (2 points)
  - Insert test records to the student table. (4 points)
  - Create a MySQL database username **richardweb** with password **richardchocolate**, with data privileges (select, insert, update, delete) for the **richard\_ricardo\_assignment\_db** database. (4 points)
  - All above must be done by SQL statements in the text file create\_db.sql. (0 points if not)
- Load **create\_db.sql** in XAMPP > phpMyAdmin to create the above mentioned database.
- Note: In the real world, do NOT put sql files in a website folder. Keep it offline and safe.

Example: “richard ricardo assignment db” database and the table inside

The screenshot shows the phpMyAdmin interface for the 'richard\_ricardo\_assignment\_db' database. The 'Structure' tab is active, displaying a table named 'student' with 11 rows. The table structure is as follows:

Table	Action	Rows	Type	Collation	Size	Overhead
student	Browse Structure Search Insert Empty Drop	11	InnoDB	latin1_swedish_ci	16 KiB	-
<b>1 table</b>	<b>Sum</b>	<b>11</b>	<b>InnoDB</b>	<b>latin1_swedish_ci</b>	<b>16 KiB</b>	<b>0 B</b>

Below the table structure, the 'Create table' form is visible, showing the table name 'student' and the number of columns set to 4.

Example: “student” table structure

The screenshot shows the detailed structure of the 'student' table in phpMyAdmin. The table has three columns:

#	Name	Type	Collation	Attributes	Null	Default	Extra	Action
1	studentID	int(11)			No	None	AUTO_INCREMENT	Change Drop Primary More
2	name	varchar(255)	latin1_swedish_ci		No	None		Change Drop Primary More
3	email	varchar(255)	latin1_swedish_ci		No	None		Change Drop Primary More

The primary key is set to 'studentID'. Below the table structure, the 'Information' tab is visible, showing space usage and row statistics:

Space usage		Row statistics	
Data	16 KiB	Format	Compact
Index	0 B	Collation	latin1_swedish_ci
Total	16 KiB	Next autoindex	12
		Creation	Feb 24, 2015 at 08:08 PM

## Example: "records" in student table

The screenshot shows the phpMyAdmin interface for the 'richard\_ricardo\_assignment\_db' database. The 'student' table is selected, and the records are displayed in a table format. The table has three columns: 'studentID', 'name', and 'email'. There are 11 rows of data, each with a checkbox for selection and buttons for 'Edit', 'Copy', and 'Delete'.

studentID	name	email
1	PO BLACK	poblack@gmail.com
2	SHIFU HOFFMAN	shifuhoffman@gmail.com
3	TIGRESS JOLIE	tigressjolie@gmail.com
4	JENNIFER YUH	jennifersyuh@gmail.com
5	OX STORMING	oxstorming@gmail.com
6	MONKEY CHAN	monkeychan@gmail.com
7	VIPER LIU	viperliu@gmail.com
8	MANTIS ROGEN	mantisrogen@gmail.com
9	CRANE CROSS	cranecross@gmail.com
10	OOGWAY KIM	oogway@gmail.com
11	PING HONG	pinghong@gmail.com

Example: data privileges for user **richardweb**

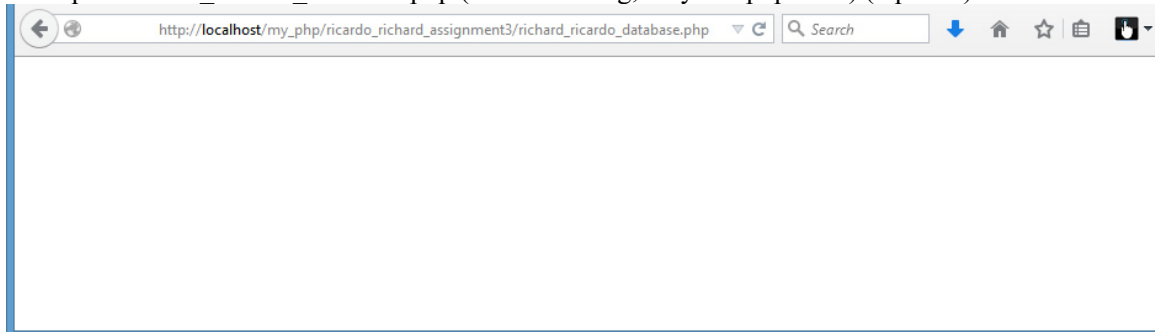
The screenshot shows the 'Edit Privileges' page for the user 'richardweb'@'localhost'. The page is titled 'Edit Privileges: User 'richardweb'@'localhost'' and shows the database-specific privileges for the 'richard\_ricardo\_assignment\_db' database. The privileges listed are SELECT, INSERT, UPDATE, and DELETE, with a 'Grant' of 'No' and 'Table-specific privileges' of 'No'. There are buttons for 'Edit Privileges' and 'Revoke'. Below the table, there is a section for adding privileges on the following database, with a dropdown menu set to 'Use text field' and a 'Go' button.

Database	Privileges	Grant	Table-specific privileges	Action
richard_ricardo_assignment_db	SELECT, INSERT, UPDATE, DELETE	No	No	Edit Privileges Revoke

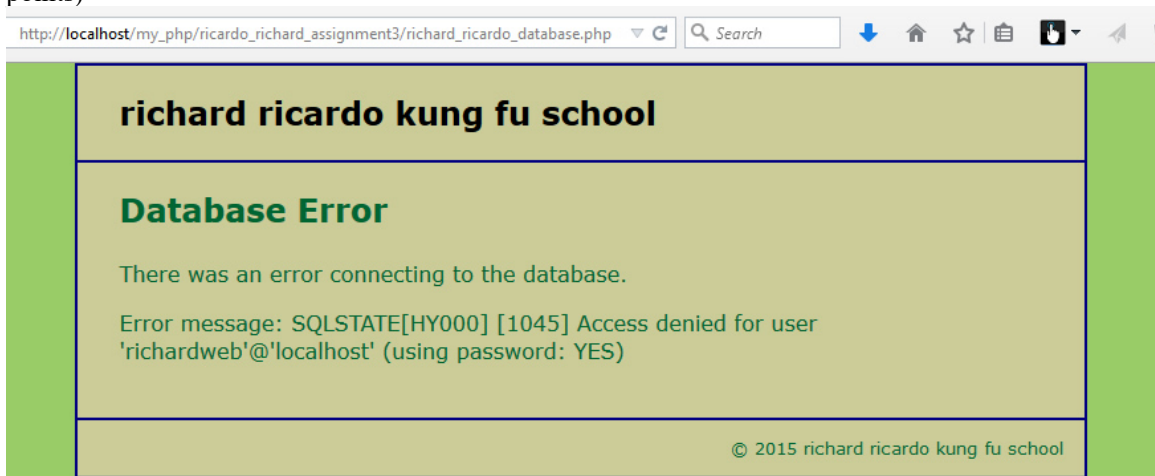
**Question 2 – Display:** PHP Chapter 4, eg008/9 and knowledge of SQL (40 points) **Estimated time: 4 hours**

- Save question 2 files in “**lastname\_firstname\_assignment3**”: (4 points)
  - **index.php**
  - **main.css**
  - **richard\_ricardo\_database.php**
  - **richard\_ricardo\_database\_error.php**
- Create web page(s) that displays the default information (example shown).
- The initial page and related outputs should look like the examples shown below.
- Create your page(s) using “richard ricardo’s kung fu school” as the page title(s) (<title> tag). (2 points)
- Save the first page as index.php. (2 points)
- Create file richard\_ricardo\_database.php (php code only, no html code) to connect to database
  - Use PDO (PHP Data Object) to interface with your MySQL database.
  - Connect to the **richard\_ricardo\_assignment\_db** MySQL database (created in Q1), using username **richardweb** with password **richardchocolate**.
  - Use try {} catch () {} exception handling to detect if the connection is successful.
  - If the connection is not successful, show an error page **richard\_ricardo\_database\_error.php**.
- Create a css file named **main.css** to format all pages by creating your own layout (no two students should have the same layout). You should use **the same css file** to format all questions. (6 points)
- index.php
  - Uses **richard\_ricardo\_database.php** to connect to the database.
  - This page lists the students from the “student” table (created in Q1).

Example: richard\_ricardo\_database.php (shows nothing, only has php code) (5 points)



Example: richard\_ricardo\_database\_error.php (shows this if something is wrong with database connection) (5 points)



Example: index.php (16 points)

**Richard Ricardo Kung Fu School - Students**

**Student List**

Student ID	Name	Email	
1	PO BLACK	poblack@gmail.com	Delete
2	SHIFU HOFFMAN	shifuhoffman@gmail.com	Delete
3	TIGRESS JOLIE	tigressjolie@gmail.com	Delete
4	JENNIFER YUH	jenniferyuh@gmail.com	Delete
5	OX STORMING	oxstorming@gmail.com	Delete
6	MONKEY CHAN	monkeychan@gmail.com	Delete
7	VIPER LIU	viperliu@gmail.com	Delete
8	MANTIS ROGEN	mantisrogen@gmail.com	Delete
9	CRANE CROSS	cranecross@gmail.com	Delete
10	OOGWAY KIM	oogway@gmail.com	Delete
11	PING HONG	pinghong@gmail.com	Delete

[Add Student](#)

© 2015 richard ricardo kung fu school

**Question 3 – Delete Student:** PHP Chapter 4, eg008/9 and SQL (20 points) **Estimated time: 2 hours**

- Save question 3 files in folder “**lastname\_firstname\_assignment3**”. (2 points)
  - **richard\_ricardo\_delete\_student.php**
- The initial page and related outputs should look like the examples shown below.
- Create your page(s) using “richard ricardo’s kung fu school” as the page title(s) (<title> tag). (2 points)
- When a user click on a “Delete” link on index.php, (16 points)
  - the user should be directed to **richard\_ricardo\_delete\_student.php**.
  - richard\_ricardo\_delete\_student.php uses **richard\_ricardo\_database.php** to connect to the database.
  - the selected student record will be deleted from the database table “student”.
  - index.php will be shown again with updated records.

Example: index.php, before deletion, click on the delete button on Student ID 4

http://localhost/my\_php/ricardo\_richard\_assignment3/

## Richard Ricardo Kung Fu School - Students

### Student List

Student ID	Name	Email	
1	PO BLACK	poblack@gmail.com	Delete
2	SHIFU HOFFMAN	shifuhoffman@gmail.com	Delete
3	TIGRESS JOLIE	tigressjolie@gmail.com	Delete
4	JENNIFER YUH	jenniferyuh@gmail.com	Delete
5	OX STORMING	oxstorming@gmail.com	Delete
6	MONKEY CHAN	monkeychan@gmail.com	Delete
7	VIPER LIU	viperliu@gmail.com	Delete
8	MANTIS ROGEN	mantisrogen@gmail.com	Delete
9	CRANE CROSS	cranecross@gmail.com	Delete
10	OOGWAY KIM	oogway@gmail.com	Delete
11	PING HONG	pinghong@gmail.com	Delete

[Add Student](#)

© 2015 richard ricardo kung fu school

Example: After deletion, student ID 4 was deleted. Now click on the delete button on Student ID 11.

http://localhost/my\_php/ricardo\_richard\_assignment3/ricardo\_ricardo\_delete\_student

## Richard Ricardo Kung Fu School - Students

### Student List

Student ID	Name	Email	
1	PO BLACK	poblack@gmail.com	Delete
2	SHIFU HOFFMAN	shifuhoffman@gmail.com	Delete
3	TIGRESS JOLIE	tigressjolie@gmail.com	Delete
5	OX STORMING	oxstorming@gmail.com	Delete
6	MONKEY CHAN	monkeychan@gmail.com	Delete
7	VIPER LIU	viperliu@gmail.com	Delete
8	MANTIS ROGEN	mantisrogen@gmail.com	Delete
9	CRANE CROSS	cranecross@gmail.com	Delete
10	OOGWAY KIM	oogway@gmail.com	Delete
11	PING HONG	pinghong@gmail.com	Delete

[Add Student](#)

© 2015 richard ricardo kung fu school

Example: After deletion, student ID 11 was deleted

**Richard Ricardo Kung Fu School - Students**

**Student List**

Student ID	Name	Email	
1	PO BLACK	poblack@gmail.com	Delete
2	SHIFU HOFFMAN	shifuhoffman@gmail.com	Delete
3	TIGRESS JOLIE	tigressjolie@gmail.com	Delete
5	OX STORMING	oxstorming@gmail.com	Delete
6	MONKEY CHAN	monkeychan@gmail.com	Delete
7	VIPER LIU	viperliu@gmail.com	Delete
8	MANTIS ROGEN	mantisrogen@gmail.com	Delete
9	CRANE CROSS	cranecross@gmail.com	Delete
10	OOGWAY KIM	oogway@gmail.com	Delete

[Add Student](#)

© 2015 richard ricardo kung fu school

**Question 4 – Add Student:** PHP Chapter 4, eg008/9 and SQL (20 points)

**Estimated time: 3 hours**

- Save question 4 files in folder “**lastname\_firstname\_assignment3**”. (2 points)
  - **richard\_ricardo\_add\_student\_form.php**
  - **richard\_ricardo\_add\_student.php**
- The initial page and related outputs should look like the examples shown below.
- Create your page(s) using “richard ricardo’s kung fu school” as the page title(s) (<title> tag). (2 points)
- When a user click on “Add Student” link on index.php, (16 points)
  - the user should be directed to richard\_ricardo\_add\_student\_form.php.
  - a link “View All Students” giver the user option to go back to index.php.
  - richard\_ricardo\_add\_student\_form.php uses **richard\_ricardo\_database.php** to connect to the database.
  - After the user keyed in new student data, and press the submit (Add Students) button, richard\_ricardo\_add\_student.php will be used to add student record to the “student” database table, and index.php will be shown again with updated records.
    - richard\_ricardo\_add\_student.php uses **richard\_ricardo\_database.php** to connect to the database.

Example: richard\_ricardo\_add\_student\_form.php, user input

http://localhost/my\_php/ricardo\_richard\_assignment3/richard\_ricardo\_add\_student\_fc

## Richard Ricardo Kung Fu School - Add Student

### Add Student

Name:

Email:

[View All Students](#)

© 2015 richard richardo kung fu school

Example: after successfully adding student record

http://localhost/my\_php/ricardo\_richard\_assignment3/richard\_ricardo\_add\_student.pl

## Richard Ricardo Kung Fu School - Students

### Student List

Student ID	Name	Email	
1	PO BLACK	poblack@gmail.com	<input type="button" value="Delete"/>
2	SHIFU HOFFMAN	shifuhoffman@gmail.com	<input type="button" value="Delete"/>
3	TIGRESS JOLIE	tigressjolie@gmail.com	<input type="button" value="Delete"/>
5	OX STORMING	oxstorming@gmail.com	<input type="button" value="Delete"/>
6	MONKEY CHAN	monkeychan@gmail.com	<input type="button" value="Delete"/>
7	VIPER LIU	viperliu@gmail.com	<input type="button" value="Delete"/>
8	MANTIS ROGEN	mantisrogen@gmail.com	<input type="button" value="Delete"/>
9	CRANE CROSS	cranecross@gmail.com	<input type="button" value="Delete"/>
10	OOGWAY KIM	oogway@gmail.com	<input type="button" value="Delete"/>
12	RHINO GARBER	rhinogarber@gmail.com	<input type="button" value="Delete"/>

[Add Student](#)

© 2015 richard ricardo kung fu school



Example: richard\_ricardo\_add\_student\_form.php, user input

**Richard Richardo Kung Fu School - Add Student**

**Add Student**

Name:

Email:

[View All Students](#)

© 2015 richard richardo kung fu school

Example: after successfully adding student record

**Richard Ricardo Kung Fu School - Students**

**Student List**

Student ID	Name	Email	
1	PO BLACK	poblack@gmail.com	<input type="button" value="Delete"/>
2	SHIFU HOFFMAN	shifuhoffman@gmail.com	<input type="button" value="Delete"/>
3	TIGRESS JOLIE	tigressjolie@gmail.com	<input type="button" value="Delete"/>
5	OX STORMING	oxstorming@gmail.com	<input type="button" value="Delete"/>
6	MONKEY CHAN	monkeychan@gmail.com	<input type="button" value="Delete"/>
7	VIPER LIU	viperliu@gmail.com	<input type="button" value="Delete"/>
8	MANTIS ROGEN	mantisrogen@gmail.com	<input type="button" value="Delete"/>
9	CRANE CROSS	cranecross@gmail.com	<input type="button" value="Delete"/>
10	OOGWAY KIM	oogway@gmail.com	<input type="button" value="Delete"/>
12	RHINO GARBER	rhinogarber@gmail.com	<input type="button" value="Delete"/>
13	BOAR VERNON	boarvernon@gmail.com	<input type="button" value="Delete"/>

[Add Student](#)

© 2015 richard ricardo kung fu school

**Important:**

1. If you do not put `<your name>` / `<your first name>` in the above mentioned fields (as shown in the examples), you will get **0 points** for the question(s).
2. **No two students** should submit webpages with exactly the same code, content, layout, or color combination. If found, both students will get **0 points**.
3. When you view page source in a web browser, `<!DOCTYPE html>` must be at the top of every page. In other words, all pages must be written in HTML5. (**-20 points** if not)
  - You **can** put php code before `<!DOCTYPE html>`.
  - You **cannot** put html code before `<!DOCTYPE html>`.
4. Before adding PHP code, all html files must pass html validation at <http://validator.w3.org/> without any **error** (and with only 1 warning).
5. After adding PHP code, the generated html code (Firefox web browser > right-click > view page source) must also pass html validation at <http://validator.w3.org/> without any **error** (and with only 1 warning).
6. All css files must pass css validation at <http://jigsaw.w3.org/css-validator/> without any **error/warning**.
7. If your files do not pass the html and css validations, **2 points will be deducted for each html or css error/warning** found (1 warning allowed for html validator).
8. Document (comment) your HTML files (`<!-- -->`), CSS files (`/* */`), and PHP files (`/* */` OR `//`). **Points will be taken off** for insufficient comments (`<!-- -->`, `/* */`, `//`).

**Submission instructions:**

- You need to test all document(s).
- Do screen capture(s) of the **input** and the related **output(s)**. Use any graphic editing software (e.g. Microsoft Paint, Adobe Fireworks, GIMP, or Microsoft Expression Design etc) to cut out the browser output (from the screen capture), paste them into a word document.
- Provide **2 different test cases** for each question. In other words, for **each question**, you may need to have **2 input** screen captures and **2 related output** screen captures.
- Do NOT need to do screen capture(s) of html validation results and css validation results for this assignment.
- Save the word document as a pdf file.

You need to submit the following:

1. A pdf file containing the screen capture(s) of the web browser input and output pages, name the file **lastname\_firstname\_assignment3.pdf**.
2. All html file(s), php file(s), css file(s), and other related files (e.g. image files). Zip your file folder (**lastname\_firstname\_assignment3**) into a single zip file (or rar file) **lastname\_firstname\_assignment3.zip**. In the above example, the zip file should contain the following files and subfolders. If there is any image, there should be a `\images\` subfolder.

- lastname\_firstname\_assignment3\create\_db.sql
- lastname\_firstname\_assignment3\index.php
- lastname\_firstname\_assignment3\main.css
- lastname\_firstname\_assignment3\richard\_ricardo\_add\_student.php
- lastname\_firstname\_assignment3\richard\_ricardo\_add\_student\_form.php
- lastname\_firstname\_assignment3\richard\_ricardo\_database.php
- lastname\_firstname\_assignment3\richard\_ricardo\_database\_error.php
- lastname\_firstname\_assignment3\richard\_ricardo\_delete\_student.php

Please submit the above mentioned **two files** (.pdf and .zip) 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 (commented)**? Are unclear parts of code **documented (commented)**? (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**.