SYSC4504

Fundamentals of Web Development (Winter 2024)

Assignment 3

SYSCBOOK – Part 3

	Name	Email	Responsibility
TA:	Amir Ghorab	amirhoseeinghorab@cmail.carleton.ca	Marking
	Omar Imran	OmarImran@cmail.carleton.ca	

Table of Contents

I. Introduction

In assignment 01 and assignment 02, you started working on creating multiple pages (client-side) and the starting point server-side of a local version of X (Twitter) for SYSC students. In this assignment, you will work on the final and last step of the website that will create a login system and allow multiple users to use the website.

II. Objective

Apply the theory that was learned during lectures about PHP, JavaScript and MySQL in creating a login system part of the website.
NOTE: It is recommended to go over chapters 8, 9, 12, 14 and 15 from the textbook before attempting the assignment.

III. Project Preparation

This assignment, will build up on your previous solution:

1. Download your solution from assignment 02.
2. Extract the solution and move the folder into SYSC4504_Labs folder you created in Lab 04.
3. Rename A02's folder to "`firstname_lastname_a03`". Replace `firstname` and `lastname` with your information.
   - **IMPORTANT:** the folder name should follow this format. Failure to name your folder as above will result in a zero on this assignment.
4. Open the folder in VSCode as usual and you can start implementing your project.

IV. Project Implementation

The Project is divided into two parts:

1. Updating assignment 02's files
2. Adding new features

It is recommended to read both parts as some changes in the current files require some of the newly added features. Below is the explanation for each part:

1. Updating assignment 02's files (4 points)

In this part of the assignment, you will work on updating and adding some new features to the previously created files in a02 to accommodate the login system you will create in the next section.

a. Switch all the MySQL statements to prepared statements.

- **NOTE:** Refer to topic 12's slides for more details.

b. Changes to index.php page

- The page should check if the user is logged in (`student_id` in `$_SESSION` is set)
  - If the user is **not logged** in, the page should transfer to login.php page

- **HINT:** the header function can transfer to a different page.

- If the user is **logged in**, the page should display the last 10 posts on from the database, ordered from **newest** to **oldest**.
  - The posts should be from **all users** not just the user that is currently logged in.

c. Changes to profile.php page

- The page should check if the user is logged in (`student_id` in `$_SESSION` is set)
  - If the user is **not logged** in, the page should transfer to login.php page
  - If the user is **logged in**, the page should retrieve the user's information and display it in the fields (similar to A02)

d. Changes to register.php page

- Add two fields named "`password`" and "`confirm password`" to get the user's password.
  - If the two passwords don't match, display an error message next to the password label.
  - This should be checked using **JavaScript** and before the form is submitted.
  - **NOTE:** Refer to the `users_passwords` table in the next section for more details.
- When the student registers, check if the email address exists in the database.
  - if it **does**, display an error message and ask them to enter a new email address.
    - **NOTE:** The user's information shouldn't be added to the database unless the user (email address) doesn't exist.
  - If the user **doesn't** exist in the database, continue as in A02.
    - Use `password_hash()` php function to create a hashed value of the password before adding it to the users_passwords table
    - **Example:** `$pass = password_hash("SYSC4504", PASSWORD_BCRYPT);`
- The user should be prompted with an option (a link) to login if they don't have an account.
  - The link should direct them to the login.php page.

e. Changes to user info section

- If the user is **not logged in**, the user info side bar should remain empty.
- If a regular user is **logged in**, user info side bar should display the user's information (first and last name, avatar, email address and program).

f. Changes to the navigation menu

- If the user is **not logged in**, they should only see links to index.php, lgoin.php and register.php in the menu.
- If a regular user is **logged in**, they should see links to index.php, profile.php and logout.php.
- An admin should be able to see what a regular user sees plus a link for user_list.php.
  - **NOTE:** Refer to user_list.php page in the next section for more details.

2. Adding new features (8 points)

In this part, you will work on adding new features to the website, mainly a login system.

a. Create a new table and call it `users_passwords`

- The table has two columns `student_id` (primary key, integer length 10) and `password` (`VARCHAR` length 255 characters)

b. Create a login page (login.php) (email and password)

- If the user logs in, the login page should check if the user's email address and password (using `password_verify()`) match in the database then login and update the `$_SESSION`
- To check if the retrieved hashed password from the database is correct, use `password_verify()` function.
- Example:

```
if (password_verify("SYSC4504", $pass)){
    echo "they are the same";
}
```

  - If the email and password match the database, the user should be directed to index.php
- The user should be prompted with an option (a link) to register if they don't have an account.
  - The link should direct them to the register.php page.
- For any pages other than register.php or login.php, if the user tries to access them through the URL directly without logging in, they should automatically be transferred to the login page.
- When the user logs in, there should be different contents based on their account type.
  - **HINT:** Add to the user's account type to the session
  - In the user_list.php page, check if the account type is an admin, if not, display an error message "`Permission denied`" and display a link to allow the user to go to index.php page.

c. Create a logout page (logout.php)

- When the user clicks on the logout link on the navigation menu, it should logs them out and transfer them to the login.php.

d. Create a new table and call it `users_permissions`

- The table has two columns `student_id` (primary key, integer length 10) and `account_type` (integer of length 1)
- The default value for `account_type` should be 1 (a regular user)
  - Admin (0) and regular user (1)
- Similar to A02, when you register a new user, make sure to create a record for the user in the `users_permissions` table.

e. Create a new page “user_list.php”

- The page will display a table that has a list of users and their information (“`student_id`, `First_name`, `Last_name`, `student_email`, `program`, and `account_type`”)
- This page is only accessible by users with `account_type` of Admin

V. Marking Scheme

The TA in charge of marking will import your database into their MySQL server. They should be able to test that you correctly performed all required tasks.

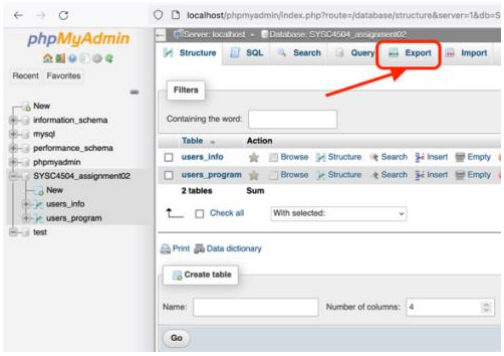
Mark division is noted next to each of the requirements listed above. The marks for each requirement will be based on:

- Completeness of your submission
- Correct solution to the problem
- Adhering to the submission requirements

VI. File Submission

After you finish all the requirements:

1. Export the database that was created as a “`.sql`” file

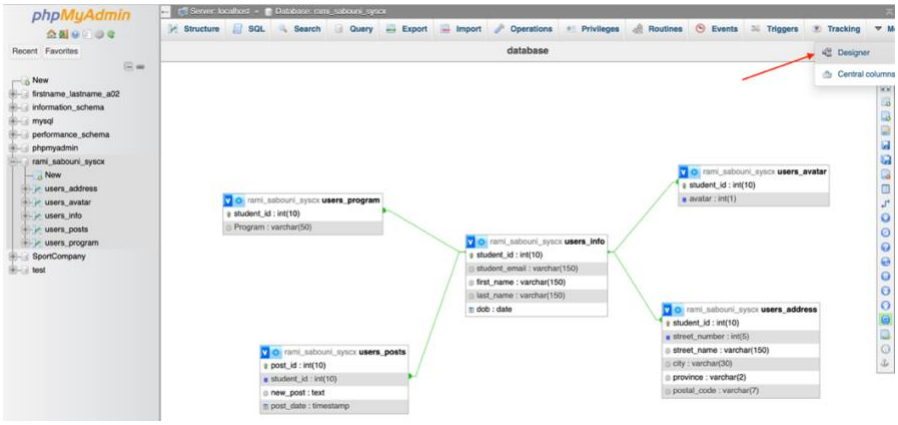


2. Name the file “`firstname_lastname_a03.sql`”. Replace `firstname` and `lastname` with your information.
3. Open the created .sql file and add the following statements at the top.

```
CREATE DATABASE IF NOT EXISTS firstname_lastname_syscx;
USE firstname_lastname_syscx;
```

- Replace `firstname` and `lastname` with your information.

4. Add the .sql file to your main assignment project folder. Should be in the same level as your `index.php`, `profile.php` and `register.php`.
5. Once you finish creating the database and the five required tables, click on the "`Designer`" tab and take a screenshot of similar to the one provided in the instructions above. Name the screenshot "`firstname_lastname_database`". The accepted extensions are `.png` or `.jpg`.
   - Replace `firstname` and `lastname` with your information.
   - **Note:** Depending on the width of your screen, you might find `Designer` tab under `More` drop down menu.



6. Add the screenshot to the assignment's folder.
7. List of files expected:
   - The assignment folder should be named: firstname_lastname_a03
   - The submitted format should be firstname_lastname_a03.zip
   - connection.php (if used)
   - index.php
   - register.php
   - profile.php
   - logout.php
   - login.php
   - user_list.php
   - Any assets (.js, images, css, …) needed by you website
   - firstname_lastname_a03.sql
   - Screenshot of the database (firstname_lastname_database.jpg or firstname_lastname_database.png)
     - Replace all firstname and lastname with your details.
   - **IMPORTANT: If you submit multiple files with the same name at different locations, or any extra files that are not needed by your website to function, you will receive a zero.**

List of files expected:

8. Compress your project's folder that you created at the beginning of the assignment.
   - **IMPORTANT:** The only allowed extension is: **.zip**
9. Submit your work on Brightspace.
   - No email submissions are allowed
   - Missed deadline will result in an automatic **zero**