[IS113] Extra Exercises - Week 12 - Session Management & Authentication

Objectives

- To master the concepts of session management in PHP
- To master other selected topics, e.g., password hashing, etc.

Instructions

- Questions with no asterisk mark are easy peasy.
- Questions marked with * are slightly challenging.
- Questions marked with ** are challenging.
- Questions marked with *** are very challenging.

Download

- **Resources**: Click here

NOTE: If you spot any mistakes/errors in the questions, please contact your instructors by email and state the issues. We will try to address it as soon as possible.

Database Connection (from inside PHP code)

1) **WAMP Users**
    a) Upon WAMP installation, if you have not changed your MySQL login info will be:
        i) **Username**: root
        ii) **Password**: <left empty>
2) **MAMP Users**
    a) For most students we have assisted, it appears that the default MySQL login info is:
        i) **Username**: root
        ii) **Password**: root
    b) Additionally, your **MySQL port** appears to be **3306** (*please verify this on your own laptop computer and **remember to note it down***).
        i) You will have to specify **port** in `ConnectionManager.php`.
        ii) Please remember to configure `ConnectionManager.php` on your own in all Extra Exercises as well as in Lab Test 2 questions **on your own** (as we instructional staff do NOT provide a separate `ConnectionManager.php` file for non-WAMP users).

Question 1: Find Oldest Person (*)

Given:

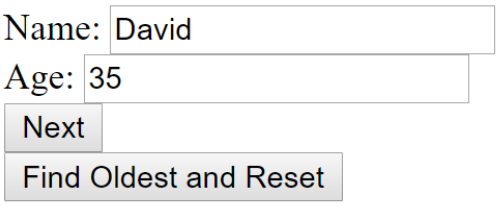
- `find_oldest_person/`
  - `Person.php` (**complete**)
  - `input.php` (**complete**)
  - `process.php` (***partial***)

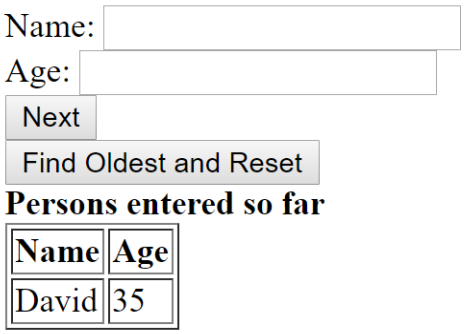
How data flow across pages:

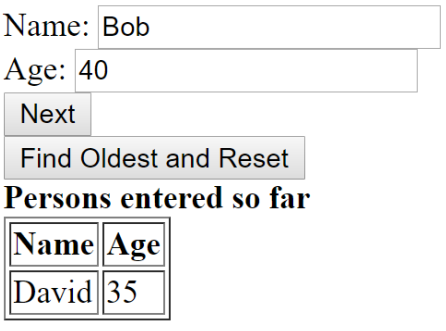
- **`input.php -> process.php -> [input.php -> process.php]*`**

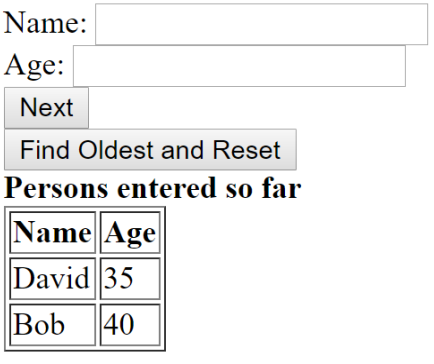
Part A: Add one person

Complete **`process_one_person`** function of **`process.php`** such that it creates a new `Person` object based on data entered by user inputs (name and age) and stores it in the session. The session maintains an array of `Person` objects based on person details entered so far. Read comments in the PHP file for detailed instructions. You can assume that users always enter correct data. If Part A is completed well, the following would be the expected behavior:

<table>
<tr>
<td>(1) New person details are entered:
 

input.php</td>
<td>(2) After “Next” button is clicked:
 

input.php</td>
</tr>
<tr>
<td>(3) New person details are entered:
 

input.php</td>
<td>(4) After “Next” button is clicked:
 

input.php</td>
</tr>
</table>

Part B: Find Oldest

Complete **`find_oldest_and_reset`** function of **`process.php`** such that it finds and displays the oldest persons (it could be one or more) that have been entered through the form so far, and resets the array of `Person` objects stored in the session. Read comments in the PHP file for detailed instructions. If Part B is completed well, the following would be the expected behavior:

(1) *After multiple persons are entered:*

Name:
Age:
Next
Find Oldest and Reset

Persons entered so far

Name	Age
David	35
Bob	40
Ann	40

input.php

(2) *After "Find Oldest and Reset" button is clicked:*

Oldest persons entered:
Bob (40)
Ann (40)
Start Again

process.php

(3) After "Start Again" link is clicked

Name:
Age:
Next
Find Oldest and Reset

input.php

Question 2: Shopping Cart (*/**)

Given:

- `shopping_cart/`
  - `shop.php` ***(partial)***
  - `process.php` **(partial)**
  - `autoload.php` (complete)
  - `model/ConnectionManager.php` (complete)
  - `model/Item.php` (complete)
  - `model/ItemDAO.php` (complete)
  - `model/week12extra.sql` (complete)

How data flow across pages:

- **`shop.php -> process_shop.php -> [shop.php -> process_shop.php]*`**

Import `week12extra.sql` into MySQL database

Part A: Display Items from Shopping Cart (Difficulty Level: *)

Complete **`shop.php`** such that it displays, in a table, the details (item name, price, quantity) of the `Item` objects stored in the session. Also compute and display the total price of all the items in the shopping cart. Read comments in the PHP file for detailed instructions.

Refer to the figure below for the expected behavior.

Part B: Add Items into Shopping Cart (Difficulty Level: **)

Complete **`addToCart`** function of **`process_shop.php`** such that it creates a new `Item` object for each item quantity entered by the user in `shop.php` and stores it in the session. For example, if the user enters Quantity = 2 for the item **iPhoneXs** in `shop.php`, upon clicking "Add to Cart" button, `process_shop.php` should create an `Item` object with properties: `name=iPhoneXs, price=2088, quantity=2`. Likewise for all the other item quantities the user entered.
Create an array of those `Item` objects and store the array in the session. That is, the session maintains an array of `Item` objects based on the item quantities entered so far.

Read comments in the PHP file for detailed instructions and refer to the figure below. You can assume that users always enter correct data.

Part C: Clear Shopping Cart (Difficulty Level: *)

Complete **`clearCart`** function of **`process_shop.php`** such that it clears all the item objects stored in the session.

If Part A, Part B, and Part C are completed well, the following would be the expected behavior:

(1) *Item quantities are entered:*

Item Name	Price (S$)	Quantity
HuaweiMate20Pro	1080	
HuaweiP30	1880	
iPhoneXs	2088	2
iPhoneXsMax	2388	1
LGG7Thinq	589	
LGV40Thinq	1098	
SamsungS10	1228	
SamsungS10e	988	
SamsungS10plus	1428	

Add to Cart
Clear Cart

shop.php

(2) *After "Add to Cart" button is clicked:*

Shopping Items added into session. Please continue to shop!
Continue Shopping!

process_shop.php

(3) *When "Continue Shopping!" button is clicked:*

(4) *User adds new items:*

Item Name	Price (S$)	Quantity
HuaweiMate20Pro	1080	
HuaweiP30	1880	
iPhoneXs	2088	
iPhoneXsMax	2388	
LGG7Thinq	589	
LGV40Thinq	1098	
SamsungS10	1228	
SamsungS10e	988	
SamsungS10plus	1428	

Add to Cart
Clear Cart

Your Current Shopping Items:

Item Name	Quantity	Price (S$)
iPhoneXs	2	2088
iPhoneXsMax	1	2388
Total		6564

shop.php

Item Name	Price (S$)	Quantity
HuaweiMate20Pro	1080	
HuaweiP30	1880	
iPhoneXs	2088	
iPhoneXsMax	2388	
LGG7Thinq	589	2
LGV40Thinq	1098	
SamsungS10	1228	1
SamsungS10e	988	1
SamsungS10plus	1428	

Add to Cart
Clear Cart

Your Current Shopping Items:

Item Name	Quantity	Price (S$)
iPhoneXs	2	2088
iPhoneXsMax	1	2388
Total		6564

shop.php

(5) *After "Add to Cart" button in shop.php and "Continue Shopping" button in process_shop.php are clicked:*

Item Name	Price (S$)	Quantity
HuaweiMate20Pro	1080	
HuaweiP30	1880	
iPhoneXs	2088	
iPhoneXsMax	2388	
LGG7Thinq	589	
LGV40Thinq	1098	
SamsungS10	1228	
SamsungS10e	988	
SamsungS10plus	1428	

Add to Cart
Clear Cart

Your Current Shopping Items:

Item Name	Quantity	Price (S$)
LGG7Thinq	2	589
SamsungS10	1	1228
SamsungS10e	1	988
iPhoneXs	2	2088
iPhoneXsMax	1	2388
Total		9958

shop.php

(6) *When "Clear Cart" button in shop.php is clicked:*

Shopping Cart Cleared. Please continue to Shop!
Continue Shopping!

process_shop.php

(7) *When "Continue Shopping" button is clicked:*

Item Name	Price (S$)	Quantity
HuaweiMate20Pro	1080	
HuaweiP30	1880	
iPhoneXs	2088	
iPhoneXsMax	2388	
LGG7Thinq	589	
LGV40Thinq	1098	
SamsungS10	1228	
SamsungS10e	988	
SamsungS10plus	1428	

Add to Cart
Clear Cart

shop.php

Question 3: Login Pages (**)

Given:

- `login_pages/model`
  - `ConnectionManager.php, User.php` (**complete**)
  - `UserDAO.php` ***(complete)***
- `login_pages/`
  - `common.php` (**complete**)
  - `register.php` (**complete**)
  - `process_register.php` ***(partial)***
  - `login.php` (***partial***)
  - `process_login.php` ***(partial)***
  - `change_password.php` (***partial***)
  - `process_change_password.php` ***(partial)***
  - `logout.php` ***(partial)***
  - `welcome.php` ***(complete)***
  - `database.sql` ***(run this before you start)***

This exercise allows you to get familiar with typical user login and authentication functions, with the encrypted password stored in the database. It makes use of simple session management variables to manage the passing of information. It covers the following functions:

- Register a user
- Login as a user
- Change password
- Logout

Note: This exercise does not cover all possible scenarios of the login and authentication processes that typical businesses need.

List of password setup in the database.

username	**password_hash**	**Actual password (not a database field)**
zack	Hash value of the actual password; not shown here	zackp
yew		yewp
wong		wongp
tan		tanp

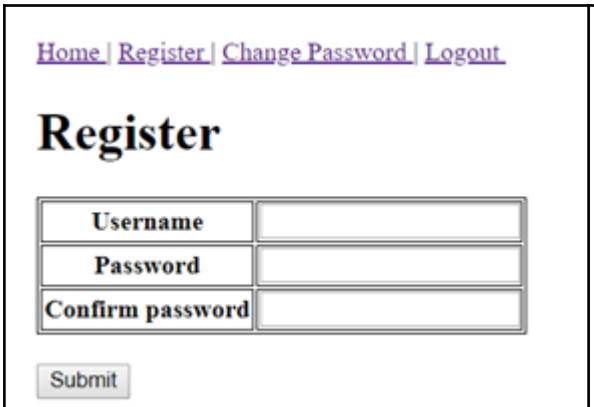
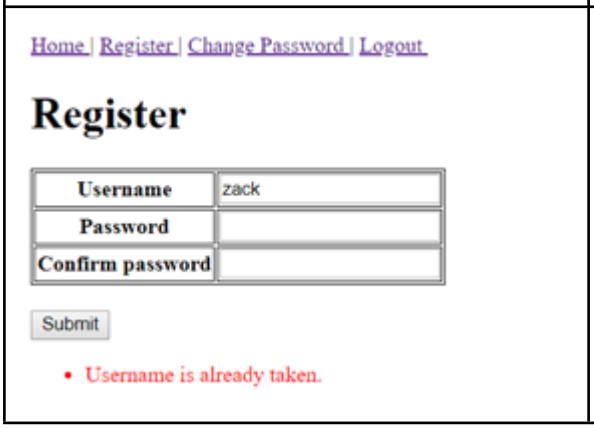
The following shows one of the ways to handle session management variable in the application.

Part A: Complete "`process_register.php`"

Complete `process_register.php` to perform the following:

- Retrieve data from `register.php`. Check that they are not blank.
- Check that the Username does not already exist in the database.
- Check that the Password and Confirm Password are the same.
- If any errors, `register.php` will be reloaded, with username displayed in the Username field and display error messages.
- If there are no errors, hashed the Password and create a `User` object and add the record to the database. Redirect page to `Login.php`. The username should be displayed in the Username.

The following shows the flows of the web pages.

	When the page is first loaded.
	Display of errors.
	Check if username is taken.

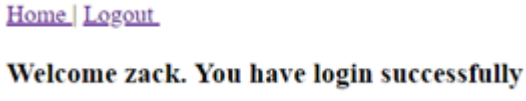
Home | Register | Change Password | Logout **Welcome ! Login Page** **Username** orange **Password** Login	Successful registered and redirected to login.php.

Part B: Complete "`process_login.php`"

Complete functions in `process_login.php` to perform the following:

- Retrieve data from `login.php`. Check that data are entered.
- Check that the Username is valid.
- Verify the password against the hashed password in the database.
- If any errors, `login.php` will be reloaded, with username displayed in the Username field and display error messages.
- If login is successful. Redirect the page to `welcome.php`

The following shows the flows of the web pages.

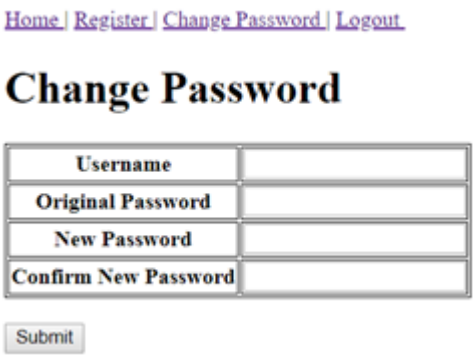
	When the page is first loaded.
	Enter usename = zack1. Error to display username does not exist.
	Enter usename = zack, password = 'abc' Error to display error.
	Enter usename = zack, password as 'zackp' Redirect to welcome.php

Part C: Complete "`process_change_password.php`"

Complete functions in `process_change_password.php` to perform the following:

- Retrieve data from `change_password.php`. Check that data are entered.
- Check that the Password and Confirm Password are the same.
- Check that the Username exist in the database.
- Check if the Username, Original Password pair is valid against the data in the database.
- If any errors, `change_password.php` will be reloaded, with username displayed in the Username field and display error messages.
- If there are no errors, the new password is hashed and updated in the database. Redirect page to `Login.php`. The username should be displayed in the Username.

The following shows the flows of the web pages.

<table>
<tr>
<td> 
</td>
<td>When the page is first loaded.</td>
</tr>
<tr>
<td> 
</td>
<td>Enter username = zack and the rest of the fields empty.

Errors are displayed.</td>
</tr>
</table>

Home | Register | Change Password | Logout **Change Password** Username: zack1 Original Password: New Password: Confirm New Password: Submit • Username is invalid.	Enter username = zack1, Original password = zackp, New Password = newzack and Confirm New Password = newzack.
Home | Register | Change Password | Logout **Change Password** Username: zack Original Password: New Password: Confirm New Password: Submit • Existing password invalid	Enter Username = zack, Original password = zack, New Password = newzack and Confirm New Password = newzack.
Home | Register | Change Password | Logout **Change Password** Username: zack Original Password: New Password: Confirm New Password: Submit • The NEW passwords are different.	Enter Username = zack, Original password = zackp, New Password = new and Confirm New Password = newzack.

Home | Register | Change Password | Logout **Welcome ! Login Page** Username: zack Password: Login	Enter Username = zack, Original password = zackp, New Password = newzack and Confirm New Password = newzack. Successful registered and redirected to login.php

Part D: Complete "`logout.php`"

- Check that if no user is not authenticated and login successfully, it will redirect to login.php page.
- If there is a user that is successfully login to the application. The following message will be shown.

Thank you zack for visiting

Home

- Unset all session variables used.