Extra Credit Lab

Due: 11:00pm on December 7, 2014

20 points

This lab is an all or nothing lab. If you complete this lab, your lowest lab score will be turned into a 20. You will be adding features to your completed Lab 8 for this lab. If you have not completed lab 8, you must complete it before starting this lab. All requirements for lab 8 apply to this lab. They are listed below for reference.

You will be adding to and modifying the login website we created in class. In other words, make sure you have a login page and an "index" page to land on once you are logged in. This index page should not be accessible if you are not logged in. Be sure to follow the directions below. In order to find your file on the server, make sure you specify the URL to access your file in your d2l submission as a comment. I would create a separate folder for this lab as there will be several files. It should be stored at something like http://webdev.cs.uwosh.edu/userID/Labs/Login/index.php. A link from the comments in your d2l submission makes your page much easier to find.

1. Create a registration page that allows the user to sign up for your website. The registration page must accept a username, password and date of birth.

2. Once the user has entered a username and has left the username textbox (lose focus), the username is checked against usernames already in the database. You must accomplish this using AJAX. If the username already exists in the database, the username textbox turns red and a line of text underneath the textbox appears telling the user that particular username has already been taken. The submit button must be disabled if a valid username has not been chosen. It is only enabled if a valid username has been entered.

3. Your registration page must have two password textboxes as the user must reenter the password. The textboxes should not display the password but only display *'s.

4. You must disable the second password textbox until the user has entered a valid password in the first textbox. A valid password is one that has a number, a letter and a character that is neither a number or a letter. The password must also be 8 or more characters. Once the user has satisfied those requirements, the second password textbox is enabled. If the user modifies a valid password in the first box and it no longer meets the password requirements, the second password box is disabled, the text

is cleared out and the background color is cleared out. As a hint, think about the onkeyup event.

5. When the second password textbox is enabled, the background of the textbox should be red to indicate the passwords do not match. The textbox should remain red until the user matches the first password textbox. When the password is entered the same as the first textbox, the textbox background immediately turns green. If the user makes a change to either password textbox and the passwords differ again and the password in the first textbox is still valid, the textbox background should change back to red. If the password of the first box became invalid, the action at the end of step 4 should occur.

6. The submit button should be disabled if the passwords are not valid or are not the same.

7. The user must have entered a valid date. A date is valid if the user is more than 18 years old and less than 120 years old. Do not hardcode values in for what is 18 years old. Make sure to use the current date and time so your site isn't static. This check must occur when the user clicks the submit button. If the date was invalid, an error message pops up in some dialog box. If the date were valid and everything else was filled in correctly, the form submits as usual. This task is nontrivial and will take a bit of work to figure out how to do correctly.

8. Once the data has been submitted, you must ensure the username has not already been taken using php. If the username were taken already, you should post this error message and post a link back to the registration page.

9. Once the data has been submitted and the username is unique, your php file must do all of the validations again. You must validate the password has a letter, number and other character and must ensure the date is valid. It is not hard to bypass the form's validations and submit faulty data. Assuming the data is correct, you must insert the username, password and date of birth into your database. How you store the information is up to you. The only requirement is that you can't store the passwords as plaintext. You must use some kind of hash to encrypt the password. You can read about password hashing here: http://php.net/manual/en/faq.passwords.php

10. Modify the login procedure to query the database to ensure the user exists and the password is correct. If correct, the user is logged in. If incorrect, the same error message is shown.

11. A user should not be able to register if the user is already logged in.

12. You must create a logout link that logs the user out. After the user is logged out, the user is automatically redirected back to the login page.

13. On the index page that the user sees after logging in, the user should see his or her birthday as entered on the registration page. The user should be able to update the birthday without having to submit anything. This must be accomplished using AJAX. The new date being entered in must pass all validations as before. If the date entered doesn't pass the previous validation for dates, the original date is put back.