**Daniel Williamson**

**Project #7**

**11/13/16**

**CS 212**

[**http://cefns.nau.edu/~daw277/index.html**](http://cefns.nau.edu/~daw277/index.html)

Project Overview

Purpose:

The purpose of this was rather difficult. A lot of reviewing, reading online and referring to classmates, namely Joseph Remy. The reason for this project was to create a login in page that connects the user to the database and logs them on. While being logged on the user then has the choice to logout or change their settings. They can delete their account and/or change their password from the settings page. It is also a place where they can view account activity.

Approach:

To begin, I knew I had to create some new pages, so I made a setting, a login, a register and a logout page. All for the user to access. To begin, the register page was created to do just that. To register the user and create an account. It prompts the user for some input, a username, password and to verify the password. It connects to the database and sends all the information to the databases arrays. I created 2 arrays into the database, a log\_users which holds a logID, which determines if the user logged on or not, a type to determine if the account is active or not, a date which logs times for whenever the user logs in, out, changes password, or deletes account and lastly a userID. The second table, named users, holds a userID, password which holds the users the password, email which holds the users email, username which holds the user’s username, isActive which determines if the user is active or not, and lastly isAdmin to check if the user has admin privileges. If for any reason the username the user selected is already being used or does not meet the requirements, it will notify them. If the email is being used or if the email is not valid, it will notify the user. If the passwords do not match or if it does not meet the requirements, it will notify the user. If all checks out, it will send all the information to the database, to be stored. It then notifies the user to go login.

Which leads us to the login page. Which, simple just asks the user to enter their username and password. Where it retrieves the information entered. If the username exists in our database and the password entered is correct it logs in the user. If the username does not exist it will display a vague message saying neither the password or username our correct. If the password was incorrect it does the same thing.

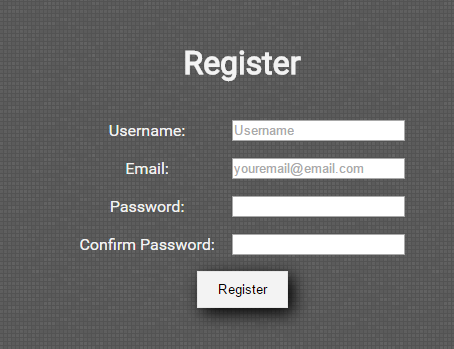
Since we have now created and logged in as a user, we now need a settings page. Which asks the user to input their current password to either change their password or to delete their account. On this page, it will also print out the user’s activity. To change the password, we just retrieve the information in our database, if the current password matches the user’s password it will allow the user to change their password to whatever the user entered into the new password. If the new password meets our requirements AND matches the verification password, it will replace the old password with the new one. If the user wants to delete their account. We simple toggle the isActive, from a 0 to a 1, to indicate, they are no longer active.

Lastly, we have a logout page. Which isn’t really much of a page anyway. When clicked, it terminates the session and directs them to the index page.

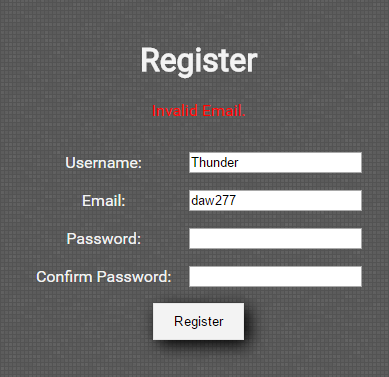
Results

These are my results:

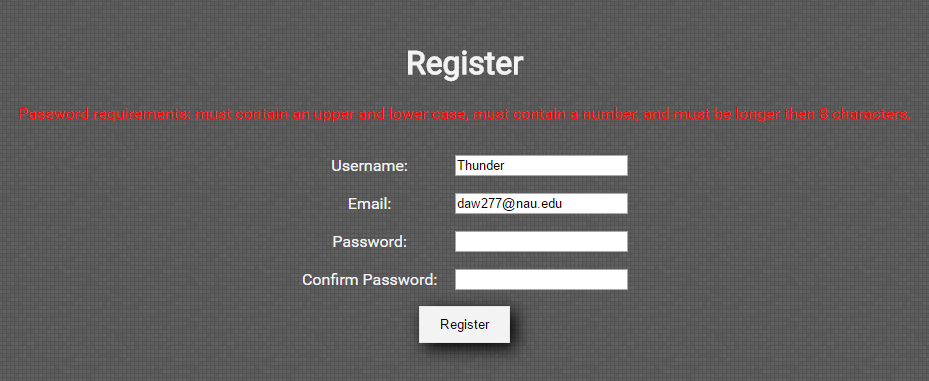
This is the interface for the register page:



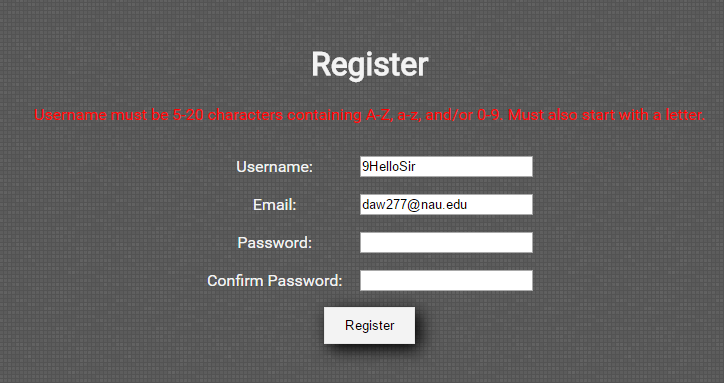
If email is not valid:



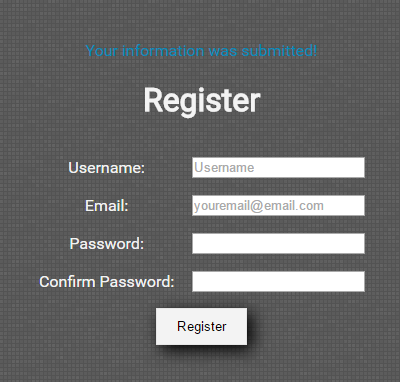
If password did not meet the requirements:



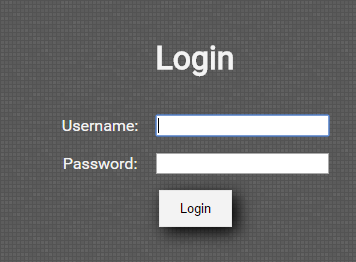
If user did not meet username requirements:



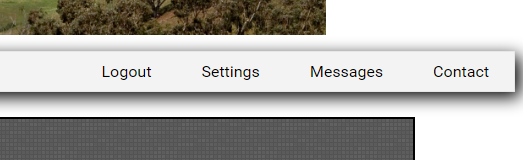
If all was good and you are successful.



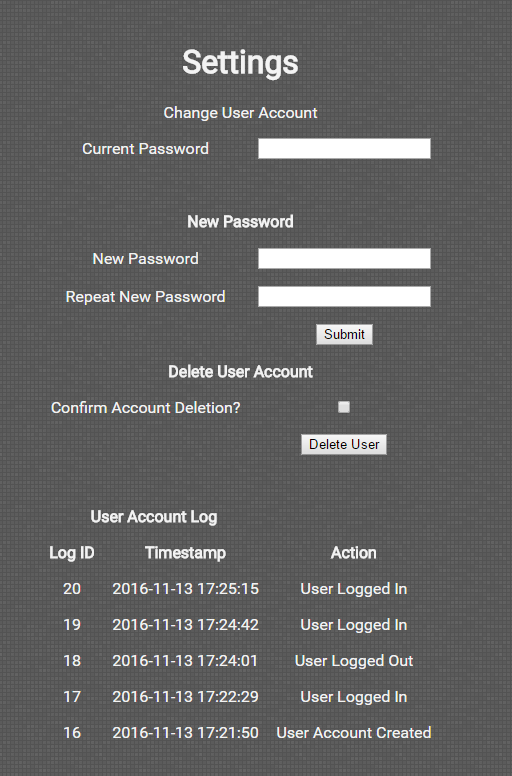
Login page interface:



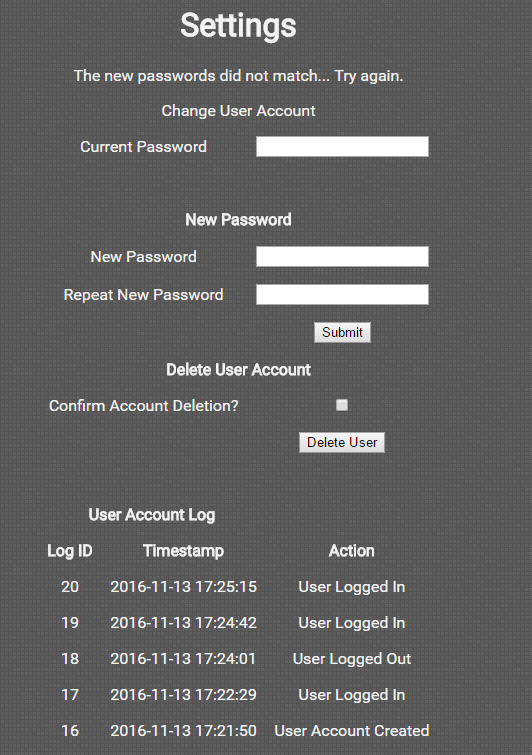
Once logged in, this is the new navigation bar:



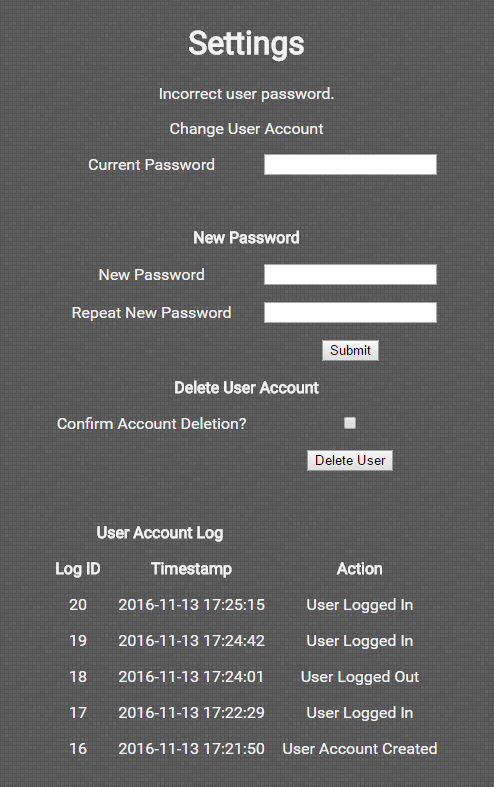
Settings page interface: Note the user account log.



If user did not match their new password:



If they did not enter the correct current password:



Conclusion

In conclusion, this was a very difficult lab. There was a lot of code involved. But I learned a lot about PHP. I even optimized my website by using the require\_once function that PHP offers. It allows me to have a single \_headers and \_footers page that I can change just once if I need to add something into my header.