Principles of Web Development

Mini Assignment 9
Due: November 7, 2020 on MyCourses at 23:59

In the course outline is says we have 10 mini assignments but in the grading scheme it says we have only 9 mini assignments. Therefore, this is our FINAL mini assignment. After you finish this assignment, you will only need to hand in your project. We will still have in-class sessions to practice new material.

In this mini assignment:

- You do **not** need to make your webpage:
  - Pretty
  - Interactive
  - Responsive
- You **must** do:
  - Asynchronous <form>
  - Calls a Python program
  - That uses an CSV text file as a database
  - Runs in the public_html folder in the SOCS server under your account

Please do the following:

Your job is to redo mini 8 but this time use an asynchronous AJAX call to invoke a Python program.

There is also an optional glory question.

Your job is to (non-glory question):

1. If you have not already created your public_html directory in the SOCS server then please do so. You may need to do Exercise 1 from Lab 1.
2. Login to your account using ssh or putty
3. Now cd into public_html
4. Next vi loginAsyc.html, and create a simple form with the following: <h1>Please Login</h1> then a programmatic **form (but not a <form>)** that displays one text input and one password input. The text input displays "Username:" and the password input displays "Password:" (without the quotes). More specifically, the word Username: is followed by a textbox then a new line and the word Password: with a password input box following. Below these two input boxes display a submit <button> tag. The submit button displays "Login" instead of Submit. The async call invokes a program named checkpass.py using the method "post".
5. Now, create a CSV text file as the database called users.csv. The CSV file is formatted in the following way: first word is the username followed by a comma and then the password as the second word. Below is an example:

   bob,12345
   Yuki,abc123

Maria,jklmnop

6. The program checkpass.py will open the CSV file and search for the existence of username and password (entered from the form) with the valid usernames and passwords in the CSV file. If the username and password match, then the user sees "Your Password Matches" otherwise the user sees "Wrong username or password".
7. Program ends.

GLORY QUESTION (optional)

You get no additional points for doing the glory question. This is only for bragging rights. However, the TA will look at your work and comment on it.

The glory question is about password security. In professional websites the passwords are not stored in readable form in the database. Instead they are encrypted.

Create another Python program called Caesar.py that can encrypt and decrypt a string using command-line arguments. Specifically: Caesar -e abc123 5 will encrypt abc123 with the key 5 and print the result to the screen. Caesar -d jklihj 7 will decrypt jklihi with the key 7 and print the result to the screen. The algorithm is Caesar cipher, a very simple character shifting cipher (you can look it up if your are not familiar with it).

The checkpass.py program would need to decrypt the password before comparing. To make this easy, assume the passwords in the CSV file are already encrypted. Your cgi program will only need to decrypt the password. Use Caesar.c program to help you. (You can make the glory question less glorious by writing the Caesar cipher as a function within your cgi program, instead of it being a secondary program).

WHAT TO HAND IN

- Make sure the program works in your public_html SOCS directory
- Submit a readme.html file with your name and an <a>URL</a> to your login.html file on SOCS
- ZIP loginAsync.html, checkpass.py, users.csv and upload that to the MyCourses assignment box for the TA to look at. Make sure you coded this by hand.
- If you did the glory question, then add the Caesar.py program in the zip file.

HOW IT WILL BE GRADED

- Maximum points 20
- 5 points for loginAsync.html
- 15 points for checkpass.py
    - 2.5 CGI input from the packet
    - 2.5 Printing the resultant web page
    - 5.0 Password checking logic
    - 5.0 points reading the csv text file database