University of Washington, CSE 190 M
Homework Assignment 4: NerdLuv
EXTRA FEATURES

The following are **three extra features** you can complete to add new functionality to your program. Each optional feature you complete is worth +1 point, redeemable on HW4 only. In other words, you can't get more than 20/20 points on HW4, but a properly implemented extra feature will "make up" for any one -1 deduction you would have otherwise earned on this assignment. Think of them as insurance. If the sum of your extra features and points earned exceeds 20/20, each additional extra feature you completed will earn you **+1 free late day**.

If you choose to do any of these extra features, in your file's comment header, indicate which one(s) you did.

Extra Feature #1. Robust page with form validation:

If the user submits invalid data that doesn't match the above criteria to **signup.php**, or if the user submits an empty name to **matches.php**, do not show any matches or sign up the user. Instead display an error message of your choice, such as the one shown at left.

Add PHP code in **signup.php** and **matches.php** that tests all query parameters submitted for validity. Use PHP **regular expressions** to do this. Specifically, you must check the following aspects of each query parameter that is submitted using proper strict regular expressions that allow only these patterns:

- The **name** must not be blank (both pages).

- The **age** submitted must be an integer and must be between 0 and 99 inclusive.

- The **gender** submitted must be male or female. (No other values such as "robot" are allowed.)

- The Keirsey **personality type** must be a 4-letter string whose letters come from the Keirsey personality dimensions: I or E for the 1st letter, N or S 2nd, F or T 3rd, and J or P 4th.

- The favorite **operating system** must come from the choices provided.

- The **seeking min/max ages** submitted must be integers, must be between 0 and 99 inclusive, and the minimum seeking age must be less than or equal to the maximum.

Also make your pages robust against the following simple basic input errors:

- Re-submitting to **signup.php** a person who is already in the file.

- Trying to view matches on **matches.php** for a person who isn't in the file.

Note: It may help you to know that PHP has a function `exit` that you can call to immediately stop the currently running script. This function can be useful if you encounter an error and don't want to display any more of the HTML that is to come later in your code. You can display an appropriate error, end the page, and `exit` immediately.

Extra feature #2. User photos:

Modify the **matches.php** page to display photos for each user.
There is a set of images in the following folder on the course web site for every user that is initially in the input file:

http://www.cs.washington.edu/education/courses/cse190m/10su/homework/4/images/

Matches for Bjarne Stroustroup

Leia Organa
gender: F
age: 53
type: ISFJ
OS: Windows

Mary Lou Jepsen
gender: F
age: 45
type: ISTJ
OS: Windows

Roberta Williams
gender: F
age: 57
type: ENFP

When the user views his/her matches, instead of displaying the default image of **user.jpg**, instead display the proper image for each individual user, as shown at left. (See the course web site for screenshots.) The file names exactly match the users' names, except that the file names are all lowercase and with spaces replaced by dashes.   For example, there are image files named **marty_stepp.jpg** and **angry_video_game_nerd.jpg**.

But the system won't have photos for new users that sign up. So you should also modify **signup.php** to allow each new applicant to submit a photo. To handle uploading of photos, add a file input box to **signup.php** that allows the user to browse for a file to upload. Use an `input` tag with `size` of 18 characters wide. (See screenshot below.) When **signup.php** receives a POST, it will save this photo onto the server into a subdirectory named **images/** using the naming system described (such as **dick_cheney.jpg**) and show it when subsequent users search for matches. You may assume that every user submits a valid JPEG image file.

New User Signup:

Name:
Gender: ○ Male ● Female
Age:
Personality type: (Don't know your type?)
Favorite OS: Windows ▾
Seeking age: to
Photo: Browse…
Sign Up

You will also need to make one small modification to your code for **matches.php**. When displaying images for each user, some of the images will come from the web URL above (the initially provided users), and some will come from your own local server (any newly signed-up users). To be able to tell which is which, you should insert a test in your code for each match. Test whether the image file exists on the server in the **images/** directory; if so, display it from there. If not, assume that it exists on the course web site in the previously named directory and display it from there.

Note: When saving a file into a directory like **images/**, you may get permission issues. Make sure that your **images/** folder on the server has both "Write" and "Execute" permissions enabled for all users. You may also want to add the line below to your code after saving any user's new image, so that when you save a new image file to Webster, it is given full permission so that you can move/delete it. Without this line, only the `apache` user that PHP runs as will be able to delete any newly added user image files.

```
chmod("images/WHATEVER_IMAGE_FILE.jpg", 0777);   # full permission to all users
```

Extra feature #3. Gay/lesbian/bisexual matches:

Our current specification for the assignment is exclusionary because it doesn't allow for male/male or female/female relationships. For this extra feature, we'll add several small modifications to allow a user to sign up as gay or bisexual if so desired, and to show suitable matches for that person.

To do this, we'll add a new dimension to our user data. In the **singles.txt** file we will now store a new piece of data that we'll call the "seek gender(s)" at the end of each line; this piece of data represents the gender(s) that the person is willing to date. The value will be either M (male), F (female), or MF (male and female). For example, if Oldspice Guy is a straight male, Alan Turing is a gay male, Nostalgia Critic is a bisexual male, and Rosie O Donnell is a lesbian female, then their lines in the input file would look like this *(notice the very last value on each line)*:
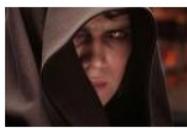
```
Oldspice Guy,M,36,ENTJ,Windows,1,99,F
Alan Turing,M,41,ESTP,Mac OS X,31,50,M
Nostalgia Critic,M,28,ENTJ,Linux,12,79,MF
Rosie O Donnell,F,46,ENFJ,Windows,30,50,F
```

We have provided a new version of the data file **singles.txt** on the course web site that includes a "seek gender" for every existing user. If you want to complete this extra feature, go get this new file and use it in your program.

Modify your **signup.php** by adding a new line for **Seek Gender(s):** to the form, so that new users can sign up as straight, gay, or bisexual. (See screenshot at left. The screenshot also shows the Photo upload box from extra feature #2, but you don't have to complete extra feature #2 to complete this feature.) Display **two checkboxes** that allow the user to select genders he/she is interested in meeting: male and/or female. It is legal for an applicant to check either box alone or both boxes. When the form first displays, the Male box is checked. When the user clicks the text label next to either checkbox, the nearby box should check or uncheck.

Also modify your **matches.php** code to only consider a pair a "match" if their "seek genders" are compatible. In other words, only consider two people to be a match if Person 1's seek gender(s) includes the gender of Person 2 and Person 2's seek gender(s) includes the gender of Person 1. The example screenshots below show some of the matches for Nostalgia Critic (left) and Rosie O Donnell (right). Nostalgia Critic (bisexual) now matches some men, including Anakin Skywalker and Rasmus Lerdorf, while Rosie (lesbian) matches other lesbian/bisexual women.