COMP-206
Software Systems
Assignment #4

Due: April 14, 2015 at 23:30 on myCourses

Facebook

We would be infringing on copy-writes to duplicate Facebook, so your "Facebook" will be themed based or interest based. It can be anything: the McGill muggle quidditch team, quantum cryptography, anime, hacking, photography, selling professors off to the highest bidder... what ever you like, as long as it is legal. Make sure to register your team and its theme on the discussion board dedicated to this announcement. Include your home page URL in case other students would like to visit your site and post messages. It will also help us find it for grading. Pick a team name.

Team Selection

You may work alone or in a team, I suggest a team. If you are working in a team then 3 students is the maximum. Select a team name and use that name for your website.

Web Hosting

The following resources are available for you:

- McGill Set-up personal web page: http://socsinfo.cs.mcgill.ca/wiki/Webserver_FAQ
- Seeing your web server errors: http://cgi.cs.mcgill.ca/cgi-bin/geterrors.cgi
- Web Tutorial: http://www.cs.tut.fi/~jkorpela/forms/cgic.html#get

SETUP

To get this assignment functional you must first set-up your McGill web page directory. I suggest you try out the sample tutorials supplied above to help you further. But, in general you need to do the following:

FIRST: Each person in your group should create a public_html directory in their home directory to practise web development, but select one team member who will host the "official" web site. That official web site will be the one that will be graded.

The public_html directory needs to be chmod'ed as a public directory. All files within this directory that you want accessible by the outside world should also be chmod'ed as public. Test that this works by creating a simple index.html or default.htm file that says "HELLO WORLD". Then use the browser on the computer of **another** team member to see if that page can be viewed. If it shows up then you have set things up correctly. Note that your home directory may also need to be chmod'ed to public (but not the files within it!).

SECOND: Divide the work evenly among yourselves. Make sure you understand all parts of this assignment even though someone else might have programmed it.

THIRD: Enjoy this assignment, be creative. You are permitted to go beyond the scope of the assignment but you will ONLY be graded on what was asked for. The additional elements are for glory, not grades. Do not replace requirements that give you grades for glory features because you will lose points.

The Web Site Front End

Facebook is a social media website that allows members to "follow" each other. Facebook requires you to first become a member, then to select other members as friends. You then get to see the posts of only those people you are friends with.

Our Facebook variant will be theme based. This means that the conversations on your Facebook variant will centre around your theme.

Our Facebook variant will be limited but don't let that stop you from doing something interesting. Your Facebook variant will have the following pages: Welcome/Login page, Become a member page, and the information feed page. Your web site should have a common theme, style and colour that will be consistently displayed across every web page. You need to determine that theme with your team members. You can search the Internet for an example web site to pattern your theme on, but do not use their code. **Do not use template sights**. For this assignment you must create everything by hand using Vi. **You cannot use authoring tools**. You will loose a lot of points if you try. You cannot use CSS or JavaScript for this assignment.

WELCOME and LOGIN PAGE

The first page is the welcome page and should be programmed in HTML. It will introduce your Facebook variant using a flashy theme based design – you **cannot** use a web-authoring tool to write the code. This home page will welcome users, provide information about your site for prospective new comers, display a graphic image related to your theme, a hyper link to the become and member page, and a CGI form asking the user for their user name and password and a submit button that says Login.

The welcome page must display the following information: Your team name, your web site theme name, at least one theme related graphic and at least one paragraph describing the Facebook variant theme and purpose.

BECOME A MEMBER PAGE

This page displays a form asking the user for their name, user name and password. A submit button labelled Register is also present. There is also a hyper link back to the welcome page if the user changes their mind.

THE FEED PAGE (the topics update page)

This page first displays a form asking the user to enter a topic update (this is like a status update but restricted to the site's theme). A submit button is beside this text box to record this update into the site's

database. After this the 10 most recent posts are displayed from your friends (but at this point is should display nothing since no one has become a member yet). Finally, to make things shorter for you, all registered user names are displayed. At the end is a text box where you can type in one of those user names, press the submit button, and they become your friend automatically without any need for the other user to accept your friendship. It is not a friendship request. You start following them immediately.

The Back-End

Your website front end has a Welcome/Login page, a Registration page, and the Topic Update page. Now we will build the back end. The back end is the part of the website that people do not see but does most of the work and stores most of the data. Your back end will use two CSV files as databases.

There is the members.csv file recording the name, user name, password, and friends of all registered users. Each member is separated from the next member by a carriage return. You may use the space character as the separator between name, user name, password, and list of friends. The information is not encrypted. [For glory the information should be encrypted]. For example:

bob jbob abc mary sam
mary mj ddd bob

The above example shows that bob's friends are mary and sam. Mary's friend is only bob. Bob's user name is jbob and his password is abc. Mary's user name is mj and her password is ddd.

There is the topic.csv file. Each entry in this file uses two lines. The first line records the user name of the individual who wrote the second line. This file contains a mixture of all the topic updates from all the users. The updates are recorded in the order they were appended to the file. This means that the items at the end of the file are the most recent. For example the file looks like this:

bob
The McGill Beater took down her opponent with s single hit.
mary
Seekers should sport a nicer outfit.

In the above example Bob's message is about Beaters and Mary's message is about Seekers.

WELCOME / LOGIN PAGE

Your welcome page not only introduces people to your website but is also lets people login or register. Your Login form will ask the user for their user name and password. After pressing the login button it will validate the login information with the members.csv file allowing only registered users access to your Facebook variant website. If login fails, then a page is generated indicating this failure and a hyper link that, if pressed, sends the user back to the Welcome/Login page [for Glory, prevent someone from accessing your topics update web page directly if they happen to know the address]. On successful login the user is directed to a webpage that hyper links to the topics webpage.

Welcome.html will send to the server the user name and password using the POST method. A compiled version of a C programmed called Login.c will be invoked by the “action” argument of your CGI form to validate the user. This C program will then either generate an error web page with a link back to the welcome page or it will display a success page with a link to the topics update page [for

Glory, send the user automatically to the welcome page or the topics update page]. The Login.c program will generate these redirection pages using printf.

THE REGISTRATION PAGE

The registration form only needs to ask the user to enter their name, user name and password.

The registration information will be validated against the user names from the members.csv file to make sure that no other user name exists that matches what was just entered. The password and name do not matter, only the user name must be unique. If there is a user name that matches then the registration fails and (1) nothing is appended to the members.csv file, and (2) an error message is displayed with a link back to the login page [for Glory, automatically display the error message on the login page in bold red large type font]. If the registration is successful then append the information to the CSV file and display a congratulations page that links the user back to the welcome page . The user will have to login.

You must create a Registration.pl Perl script that will carry out the registration process.

THE FEED PAGE (topics update page)

Your topics update page does not exist as an actual HTML page but instead as a Python program called MyFacebookPage.py. When invoked it will generate your topics update webpage customized for the user who logged in.

First, it might be worth building this page as an HTML document in order to work out all the layout bugs. Then convert it to a Python program. It is up to you. You will get partial points for the HTML version as well, in case you cannot get the Python generating version working.

You can freely design the look of your Topics Update page but it must have the following: (1) A large title with the name of the user who logged in. (2) A logout link that send the user back to your welcome/login page. (3) The status update text box and submit button permitted a member to post a message. (4) The 10 most recent messages from friends. [for glory, a way to page through all the user's friend messages] (5) A listing of all the current members. And (6) a text box that permits the user to input the user name of someone from the listing that they would like to make their friend.

Your webpage will be invoked multiple times, using the above scheme. We don't want to forget who the logged in user is. To work properly this Python program expects to receive the logged in user's name as a parameter. The best way to do this is to always invoke MyFacebookPage.py from a “form” that uses a “hidden” tag containing the logged in user's user name. When selecting the “form” submit button to invoke the Python program the “hidden” tag is treated as a variable=value combination. I suggest you use POST and the Python CGI library to access this variable. Here is an example:

<input type=”hidden” name=”username” value=””>

Put the logged in user's user name in the value attribute, for example: value=”bob”. Having the logged in user's user name makes everything else easy.

You must carry this "hidden" tag from page to page so that you never loose it. Remember that when you CGI to a new program you receive a new shell. That new shell's memory is empty and will only have what you defined from the "form" you last submitted. This means that after the user logs in then every "form" in your web page must have the above "hidden" tag with the logged in user's user name. Basically, you will "pass it on" from one page to the next.

Remember that any output from a C, Perl or Python program invoked by CGI will output back to the user's browser. If you output HTML/CGI code the browser will understand your output as a web page and display it that way. This means you need to write a Python program that outputs HTML/CGI code when it prints to the screen. The browser will do the rest automatically, making it look like an actual web page. This means that your web page can be generated by your program and populated with information from CSV files. As you write it out the browser will display it as a web page. Just make sure to output correct HTML/CGI.

I hope this all makes sense. I can also talk about it in class.

WHAT TO HAND IN

Hand in the following:

- Post your team information on myCourses.
- Submit a link.html file that will automatically take us to the assignment #4 home web page.
- Make sure your become and member page works so we can login.
- ZIP all the C, Python and HTML (plus any other files) into a single submission called Ass4.zip and upload that to myCourses.

HOW YOU WILL BE GRADED

- Your program must run in order for it to be graded. If your program does not run the TA does not need to go any further and may give you zero.
- Your program must run on the Trottier computers since this is a programming course for the Unix environment. The TA will not modify your text files in any way to get them to run under Trottier's operating system.
- All grades are awarded proportionally. In other words, if a question is graded out of 4 points and you got 75% of the program running then you will get 3 out of 4, etc.
- Grading portions of your code that do not run (assuming that your program runs at least minimally) will be graded proportionally as to how correct it is.
- Make sure your code is easy to read since this may impact the quality of your grade.

Your grade is distributed as follows:

- ➔ This assignment is worth a total of 20 points
- ➔ Everyone in your group will receive the same grade.
- ➔ +4 Login using the C language
- ➔ +4 Registration using Perl
- ➔ +12 Topics Update page (+2 hidden, +2 python, +2 update, +2 posts, +2 listing, +2 friend)