Assignment 2 - LOVE2041

- Thu Sep 25 23:00 Version 0.1 - initial release

Aims

This assignment aims to give you:

- experience in constructing a CGI script and Perl programming generally,
- practice in producing a complete CGI-based web site,
- and an introduction to the issues involved in programming for the web.

Note: the material in the lecture notes will not be sufficient by itself to allow you to complete this assignment. You may need to search on-line documentation for CGI, Perl etc. Being able to search documentation efficiently for the information you need is a *very* useful skill for any kind of computing work.

Introduction

You should construct a CGI script `love2041.cgi` in Perl which provides the core features of a dating web site called LOVE2041 for UNSW students.

Popular dating web sites (http://en.wikipedia.org/wiki/Comparison_of_online_dating_websites) implement a large set of features.

Your goal in this assignment is to implement a simpler but fully functional dating web site.

But don't panic, the assessment for this assignment (see below) will allow you to obtain a reasonable mark if you successfully complete some basic features.

Data Set

You have been provided a synthetic dataset (students) containing the dating profiles for 380 UNSW students. It also available as a zip file (students.zip). During debugging you may find it convenient to use a smaller dataset such as this subset of 30 dating profiles (students_30.zip).
The information for each student is in a separate directory named with their user name For example UNSW student Emma Watson has chosen the username *GeekGirl42* so her dating profile is in the directory: `students/GeekGirl42/` (students/GeekGirl42/)

Each students's directory contains a file named `profile.txt` containing relevant information that the student has supplied.

For example `students/GeekGirl42/profile.txt` (students/GeekGirl42/profile.txt) contains Emma's details:

```
name:
        Emma Watson
degree:
        Software Engineering
password:
        hogwarts4ever
height:
        1.60m
birthdate:
```

```
        1990/04/15
favourite_hobbies:
        Tai Chi
        Magic
        Equestrianism
        Meteorology
        Soapmaking
        Football
weight:
        62kg
favourite_TV_shows:
        Dexter
        Breaking Bad
        True Blood
        Under the Dome
        Utopia
        Teen Wolf
        The Blacklist
favourite_movies:
        Harry Potter and the Philosopher's Stone
        Harry Potter and the Chamber of Secrets
        Harry Potter and the Prisoner of Azkaban
        Harry Potter and the Goblet of Fire
        Harry Potter and the Order of the Phoenix
        Harry Potter and the Half-Blood Prince
        Harry Potter and the Deathly Hallows Part 1
        Harry Potter and the Deathly Hallows Part 2
email:
        E.Watson@ugrad.unsw.edu.au
favourite_bands:
        London Grammar
courses:
        2012 S1 COMP1917
        2012 S1 MATH1141
        2012 S2 COMP1927
        2012 S2 MATH1241
        2012 S2 MATH3411
        2013 S1 COMP2121
        2013 S1 COMP2911
        2013 S1 COMP3331
        2013 S1 MATH2111
        2013 S2 COMP2041
        2013 S2 COMP9444
        2014 S1 COMP2911
        2014 S1 COMP3411
        2014 S1 MATH2801
        2014 S2 COMP2041
        2014 S2 COMP3421
        2014 S2 COMP6771
gender:
        female
hair_colour:
        brown
favourite_books:
        Harry Potter and the Philosopher's Stone
        Harry Potter and the Chamber of Secrets
        Harry Potter and the Prisoner of Azkaban
        Harry Potter and the Goblet of Fire
        Harry Potter and the Order of the Phoenix
        Harry Potter and the Half-Blood Prince
        Harry Potter and the Deathly Hallows
username:
        GeekGirl42
```

Each student's directory also contains a file named `preferences.txt` containing information about their dating preferences.
For example `students/GeekGirl42/preferences.txt` (students/GeekGirl42/preferences.txt) contains Emma's preferences:

```
age:
        min:
                18
        max:
                22
gender:
        male
hair_colours:
        blonde
        black
height:
        min:
                1.75m
        max:
                1.85m
```

This indicates indicates Emma is looking to meet a man 18-22 years old, between 1.75 and 1.85 metres in height and with blonde or black hair.

Most students will also have some images present in the same directory. A user's primary image will be named `profile.jpg` . This is Emma's primary image:

For example `students/GeekGirl42/profile.jpg` (students/GeekGirl42/profile.jpg) contains Emma's primary image.

There may also be some other images present named `photo00.jpg` , `photo01.jpg` , ...

Note some details or preferences may not be present in all profiles. This is deliberate, it indicates the student has chosen not to supply this information and your web site should handle this sensibly. Also images might not be present for all users. Again your web site should handle this sensibly.

You are free to modify the data set and the data format in any way you choose. Your code should still assume details or preferences may be absent from profiles because students choose not to supply them.

While you do not have to use this format to store data but I expect most students will do so and unless you are very confident it is recommended you do so.

If you use another data format you should import the synthetic dataset into this format and have it available when you demo your web site so searches can be conducted using a familiar set of students.

Before choosing to use a database to store student data, note it can be tricky getting full-fledged database systems such as mysql set up on CSE systems and Andrew (& tutors) won't be able to offer any help. If you do choose to use a database sqlite is recommended because its embedded, and hence much simpler to setup, but again Andrew (& tutors) won't be able to help.

Subsets

To assist you tackling the assignments requirements have been broken into several levels in approximate order of difficulty. You do not have to follow this implementation order but unless you are confident I'd recommend following this order. You will receive marks for whatever features you have working or partially workining (regardless of subset & order).

Display Dating Profiles (Level 0)

The starting-point script you've been given (see below) displays dating profiles in raw form and does not display images. Your web site should display dating profiles in nicely formatted HTML with appropriate accompanying text. It should display the image accompanying a profile (if there is one).
Private parts of the profile should not be displayed (real name, e-mail, password, courses taken).

Interface (Level 0)

Your web site must generate nicely formatted convenient-to-use web pages including appropriate navigation facilities and instructions for naive users. Although this is not a graphic design exercise you should produce pages with a common and somewhat distinctive look-and-feel. You may find CSS useful for this.
As part of your personal design you may change the name of the website to soemthing other than LOVE2041 but the primary CGI script has to be `love2041.cgi`.

Browsing Dating Profiles (Level 1)

The starting-point script you've been given (see below) displays dating profiles only one-at-a-time. When browsing your web site should display a set (say 10) of dating profiles. It should allow the next set or the previous set of dating profiles to be viewed. The user should be able to select a single profile and see more information.

Logging In & Out (Level 1)

Students should have to login before browsing or searching for dating profiles.
Their password should be checked when they attempt to log in.

Students should also be able to logout.

Search for Users By Name(Level 1)

Your web site should provide searching for a name containing a specified substring. Search results should include the matching name and important details from their profile.

Matching (Level 2)

After logging in your web site should allow students to search for matching dating profiles.
Your web site should display dating profiles sorted on how well they match. It should display a set (say 10) of dating profiles. It should allow the next best-matching set or the previous set of dating profiles to be viewed. The user should be able to select a single profile and see more information.

Your matching algorithm should take into account the degree of mismatches. A student looking for someone 21-24 may be interested in a 25-year old, they are unlikely to be interested in a 60 year old.

In the absence of other information you should assumes students are a better match if they are moe similar. So even if a 20 year old expresses no age preference they are more likely to be interested in someone who is 22 year old rather than someone who is 42 years old. This should be given less weight than where a preference is expressed.

Your matching algorithm should weight the importance of similarities differently. For example, having the same favourite movies should be much less important than being of the same age. And if a user expresses a preference for particular hair colour this should be weighted much more heavily then having taken the some of the same courses.

Your matching must make some use of the non-numeric data (favourite books etc) even if it gives it low weight.

There are many possible choices in this matching algorithm and no single right answer. It is fine (and expected) if your results differ from other students. Be prepared to explain why your matching algorithm is sensible and how it works during your assignment demo. You may chose to have a development mode available which when turned on displays extra information regarding the matching.

User Account Creation (Level 3)

Your web site should allow students to create accounts with a user name, password and e-mail address. You should only accept a restricted syntax (certain characters) for this username and of course check that an account for this user name does not exist already. It should be compulsory that a valid e-mail-address is associated with an account but the e-mail address need not be a UNSW address. Students may prefer all their LOVE2041 e-mail goes to an external address (e.g gmail).
You should confirm e-mail address are valid and owned by the student creating the account by e-mailing them a link necessary to complete account creation.

I expect (and recommend) most students to use the data format of the data set you've been supplied. If so for a new user you would need create a directory named with their username and then add appropriate `profile.txt` and `preferences.txt` files containing their details & preferences.

Profile Text (Level 3)

A student should be able to add to their profile some text, probably describing their interests and desires, e.g.

```
I love cute puppies & kitten. I'm looking for a wonderful Python programmer
who enjoys long walks on the beach at sunset and will hold my
hand while we bring about world peace together.
```

This text should be displayed when other students view their profile. It should be possible to use some simple (safe) HTML tags, and only these tags, in this text. The data set you've been given doesn't any include any examples of such text.
If you are using the data format of the data set you have been supplied you would probably store this text in the `profile.txt` file.

Password Recovery (Level 3)

Users should be able to recover/change lost passwords via e-mail

Uploading& Deleting Images (Level 3)

A user should be able to upload an image of themself (see lecture examples). A user should be able to delete particular images.

Editing Profiles (Level 3)

A user should be able to edit all the elements of their dating profile including details, preferences & description.

Suspending/Deleting Dating Profiles (Level 3)

A student not currently interested in dating should be able to suspend their profile. When they are interested in dating again they should be able to make their profile visible again.
A student who has found their perfect match should be able to delete their account.

Messaging (Level 3)

A user, when view a dating profile, should be able to send message via the web site to the owner of the dating profile. The message should be delivered from the web site to the e-mail the owner of the dating profile. It should include a link to reply via the web site. Neither user's e-mail or real name should be revealed to the other. Users can of course exchange this information if they wish.

Advanced Features (Level 4)

If you wish to obtain over 90% you should consider providing adding features beyond those above. marks will be available for extra features. Some possibilitites are listed below.

- User-generated questions - better matching might be obtained if extra questions might be added to the existing questions (favourite books, movie, TV, ...). Students could suggests question that think think would assist in matching - , e.g. "Who is your favourite Star Trek Captain?"
- More options to be added/suggested

Getting Started

Here is the source (love2041.cgi) to a CGI script (http://www.cse.unsw.edu.au/~cs2041cgi/assignments/LOVE2041/love2041.cgi) with crude partial implementation of Level 0.
You may use this scripts as a starting point for your assignment or you may choose to ignore it.

Here is the commands you can use to try out this script (replace `YOUR_ACCOUNT` with your CSE account name):

```
% cd
% mkdir -p public_html/ass2
% priv webonly public_html/ass2
% cd public_html/ass2
% git init
Initialized empty Git repository in ...
% cp /home/cs2041/public_html/assignments/LOVE2041/love2041.cgi .
% rsync -a /home/cs2041/public_html/assignments/LOVE2041/students
/ students/
% chmod 755 love2041.cgi
% git add love2041.cgi
% echo /students >.gitignore
% git commit -a -m 'starting point'
....
% firefox http://cgi.cse.unsw.edu.au/~YOUR_ACCOUNT/ass2/love2041.c
gi
% vi love2041.cgi
% git commit -a -m 'added code for basic formatting'
....
% ....
% ....
% ....
% git commit -a -m 'tidied up assignment for submission'
% tar zcf git.tar .git
% give cs2041 ass2 love2041.cgi diary.txt git.tar
```

Assumptions/Clarifications

I expect almost all students will use Perl to complete this assignment but you are permitted to use Python. You may use Javascript for part of the assignment. A high mark for the assignment can be obtained without Javascript.

You may use any Perl or Python package which is installed on CSE's system.

You may use general purpose, publicly available (open source) Javascript libraries (e.g. JQuery) ,CSS (e.g.Bootstrap) or HTML - much sure use of other people's work is clearly acknowledged and distinguished from your own work.

You can not otherwise use code written by another person.

You may submit multiple CGI files but the primary file must be named love2041.cgi You may submit other files used by your CGI script(s). If you have need to submit many other files or a directory hierarchy, submit them as a tar file named files.tar.

Make sure you submit all files and in the appropriate hierarchy. If for example yoiu do URL rewriting in a `.htaccess` file (I'm not expecting or recommending this). make sure you submit the .htaccess file.

Your CGI script must run on CSE's system. It will be run from a class account so you must submit all necessary files and do not hard code absolute URLs or pathnames in your code. In other words don't put embed URLs like `http://www.cse.unsw.edu.au/~abc1234/ass2/subdir/background.html` in your code. Use relative URLs like `"subdir/background.html"` . Similarly don't put absolute pathnames such as `/home/abc123/public_html/ass2/subdir/datafile"` . Use relative pathname such as `"subdir/datafile"`

We will use firefox(iceweasel) on CSE lab machines for the demo session. Your code should be sufficiently portable to work on wiuth that version of firefox but you will be allowed to demo on Chrome instead but again on a CSE lab machine.

You should avoid running external programs (via system, back quotes or open). where an equivalent operation could be performed simply in Perl. You may be penalized in the handmarking if you do so.

You are permitted to run an external program to send e-mail, although this is possible from perl.

You are only required to provide basic security - using a hidden field to store user's password in plaintext is OK. More sophisticated securiy may qualify as an extra feature for subset 4.

You should follow discussion about the assignment in the course forum (https://piazza.com/class/hyuqwpte53neu). All questions about the assignment should be posted there unless they concern your private circumstances. This allows all students to see all answers to questions.

Version History & Diary

You should keep a version history as you develop - as you did with assignment 1. Submit this in a file named git.tar.
You must keep a record of your work on this assignment. The entries should include date, starting&finishing time, and a brief (one or two line) description of the work carried out. For example:

Date	Start	Stop	Activity	Comments
29/10	16:00	17:30	coding	implemented creation of user accounts
30/10	20:00	10:30	debugging	found bug in handling of hair colours

Include this diary in the files you submit as diary.txt.

Assessment

Assignment 2 will contribute 15 marks to your final COMP2041/9041 mark
20% of the marks for assignment 2 1 will come from hand marking by your tutor. These marks will be awarded on the basis of clarity, commenting, elegance and style. In other words, your tutor will assess how easy it is for a human to read and understand your program.

80% of the marks for assignment 2 will come from a peer assessment session where your submitted CGI script is tested against a specified set of operations and assessed by fellow students. You will be present to assist in determining what features are and are not working, you will also be able to indicate any extra

features you have implemented. Details of the peer assessment sessions will be announced in week 13.

Here is an indicative marking scheme .

100%	Elegant Perl with an excellent implementation of levels 0-3 and some optional (level 4) features
90+%	Very well written Perl which implements of levels 0-3 successfully
85%	Well written Perl which implements most of levels 0-3 successfully
75+%	Readable Perl which implements of levels 0-2 successfully
65%	Reasonably readable code which implements level 0-1 successfully
55%	Reasonably readable code which implements level 0 successfully
45%	Major progress on the assignment with some things working/almost working
-70%	Knowingly supplying work to any other person which is subsequently submitted by another student.
0 FL for COMP2041/9041	Submitting any other person's work with their consent. This includes joint work.
academic misconduct	Submitting another person's work without their consent.

Originality of Work

The work you submit must be your own work. Submission of work partially or completely derived from any other person or jointly written with any other person is not permitted. The penalties for such an offence may include negative marks, automatic failure of the course and possibly other academic discipline. Assignment submissions will be examined both automatically and manually for such submissions. Relevant scholarship authorities will be informed if students holding scholarships are involved in an incident of plagiarism or other misconduct.

Do not provide or show your assignment work to any other person - apart from the teaching staff of COMP2041/9041. If you knowingly provide or show your assignment work to another person for any reason, and work derived from it is submitted you may be penalized, even if the work was submitted without your knowledge or consent. This may apply even if your work is submitted by a third party unknown to you.

Note, you will not be penalized if your work is taken without your consent or knowledge.

Submission

This assignment is due at 23:59pm Sunday November 2 Submit the assignment using this *give* command:

```
% give cs2041 love2041.cgi diary.txt git.tar [files.tar] [any-other-files]
```

If your assignment is submitted after this date, each day it is late reduces the maximum mark it can achieve by 5% for the first day, 10% for the second day and 20% per day after that. For example if an assignment

worth 76% was submitted one day late, the late submission would have no effect. If the same assignment was submitted 3 days late it would be awarded 65%, the maximum mark it can achieve at that date.