Assignment 2 - matelook

Aims

This assignment aims to give you:

- experience in constructing a CGI script and Perl/Python 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/Python etc. Being able to search documentation efficiently for the information you need is a *very* useful skill for any kind of computing work.

Introduction

Andrew has decided he will make himself rich exploiting COMP[29]041 students' coding skills and then give up lecturing. Andrew's plan is to have COMP[29]041 students create a social media platform called *matelook* for UNSW students. Andrew is unaware of any other social media platforms so he thinks *matelook* will become very popular and he will become rich
matelook allows users post messages and comments on these messges and replies to these comments to be added.

matelook allows users to indicate other users are their *mates*

Your task is to produce a CGI script `matelook.cgi` which provides the core features of *matelook*.

In other words your task is to implement a simple but fully functional social media 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 Sets

You have been provided with 3 synthetic datasets containing the details of *matelook* users & their postings:

- small (dataset-small) (4Mb unpacked) zip (dataset-small.zip) (0.4Mb) tar.xz (dataset-small.tar.xz) (0.1Mb) - 10 users, 42 posts, 209 comments, 163 replies
- medium (dataset-medium) (44Mb unpacked) zip (dataset-medium.zip) (4Mb) tar.xz (dataset-medium.tar.xz) (0.8Mb) - 42 users, 420 posts, 2520 comments, 1749 replies
- large (dataset-large) (340Mb unpacked) zip (dataset-large.zip) (28Mb) tar.xz (dataset-large.tar.xz) (8Mb) - 420 users, 3525 posts, 19080 comments, 13277 replies

I expect most people will work with *medium* or *large* datasets. During debugging you may find the *small* dataset useful.

The information for each user is in a separate directory named with their zid name For example in the *medium* dataset UNSW student James Franco has zid z5098340 so his information is in the directory: `dataset-medium/z3275760/` (dataset-medium/z3275760/)

Each *matelook* user's directory contains a file named `user.txt` containing relevant information that the *matelook* user has supplied.

For example dataset-medium/z3275760/user.txt (dataset-medium/z3275760/user.txt) contains James's details:

```
email=J.Franco@unsw.edu.au
program=Engineering/Commerce
zid=z3275760
full_name=James Franco
courses=[2014 S1 CHEM1011, 2014 S1 ENGG1000, 2014 S1 MATH1131, 2014 S1 PHYS1121, 2014 S2 CVEN1300, 2014 S2
password=debbie
birthday=1998-03-25
home_suburb=University Of New South Wales
mates=[z3485885, z5063045, z5013363, z5014335]
```

Note James has supplied the suburb where he lives and the coordinates of his home address. Notice also the list of James' mates. Most *matelook* users will also have an image present in the same directory named `profile.jpg`. For example `dataset-medium/z3275760/profile.jpg` (dataset-medium/z3275760/profile.jpg) contains James's image.

Note some details may be missing for some *matelook* users. This is deliberate, it indicates the *matelook* user 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.

Also present in a *matelook* users's directory may be directory named `posts` containing posts they've made in numbered sub-directories.

For example dataset-large/z5017894/posts/5/post.txt (dataset-large/z5017894/posts/5/post.txt) contains:

```
from=z5017894
time=2013-11-03T05:54:30+0000
latitude=-33.9138
message=To the shy boy in electrical engineering, Adrian, I've had a crush on you for years. You're everyt
longitude=151.2400
```

Post directories can contain a directory named `comments` . It will contain, in numbered sub-directories comments made on the post. For example dataset-small/z5099187/posts/2/comments/2/comment.txt (dataset-small/z5099187/posts/2/comments/2/comment.txt) contains:

```
from=z3462191
time=2016-04-26T07:55:01+0000
message=HAHAHAHAHAHA okay which one of you was this z5063045 z5099187 z3493921 z5059413
```

Note this mentions 4 other *matelook* users using the *matelook* convention of referencing other users with their zid.
Comment directories can contain a directory named `replies` . It will contain, in numbered sub-directories replies made to that comment.

For example dataset-medium/z5076002/posts/9/comments/1/replies/1/reply.txt (dataset-medium/z5076002/posts/9/comments/1/replies/1/reply.txt) contains:

```
message=This was def u z5079464 hahahah
from=z5062399
time=2016-06-12T01:44:26+0000
```

You are free to modify the dataset and the data format in any way you choose. Your code should still assume details may be absent from *matelook* user details posts and comments because *matelook* users 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 large dataset into this format and have it available when you demo your web site so searches can be conducted using a familiar set of *matelook* users.

Before choosing to use a database to store *matelook* user 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 working (regardless of subset & order).

Display User Information & Posts (Level 0)

The starting-point script you've been given (see below) displays user information in raw form and does not display their image or posts.
Your web site should display user information in nicely formatted HTML with appropriate accompanying text. It should display the user's image if there is one.

Private information such e-mail, password, lat, long and courses should not be displayed.

The user's posts should be displayed in reverse chronological order.

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 something other than matelook but the primary CGI script has to be `matelook.cgi`.

Mate list (Level 1)

When a matelook page is shown a list of the names of that user's mates should be displayed.
The list should include a thumbnail image of the mate (if they have a profile image).

Clicking on the image and/or name should take you to that matelook page.

Search for Names (Level 1)

Your web site should provide searching for a user whose name containing a specified substring. Search results should include the matching name and a thumbnail image. Clicking on the image and/or name should take you to that matelook page.

Logging In & Out (Level 1)

Users should have to login to use matelook.
Their password should be checked when they attempt to log in.

Users should also be able to logout.

Displaying Posts (Level 2)

When a user logs in they should see their recent posts, any posts from their mates and any posts which contain their zid in the post, comments or replies.
Comments and replies should be shown appropriately with posts.

When displaying posts zids should be replaced with a link to the user's matelook page. The link text should be the user's full name.

Making Posts(Level 2)

Users should be able to make posts.

Searching Posts (Level 2)

It should be possible to search for posts containing particular words.

Commenting on Post and replying to Comments (Level 2)

When viewing a post, it should be possible to click on a link and create a comment on that post. When viewing a comment , it should be possible to click on a link and create a reply to that comment.

Mate/Unmate Users (Level 3)

A user should be able to add & delete users from their mate list.

Pagination of Posts & Search Results (Level 3)

When searching for users or posts and when viewing posts the users be shown the first *n* (e.g n == 16) results. They should be able then view the next *n* and the next *n* and so on.

User Account Creation (Level 3)

Your web site should allow users to create accounts with a zid, password and e-mail address. You should of course check that an account for this zid 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.
You should confirm e-mail address are valid and owned by the *matelook* user 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 zid and then add a appropriate `user.txt` containing their details.

Profile Text (Level 3)

A *matelook* user should be able to add to some text to their details , probably describing their interests, which is displayed with their user details.

```
My interests are long walks on the beach and Python programming.
I plan to use what I've learnt COMP2041 to cure the world of all known diseases.
```

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.

You can choose to store this text in the `user.txt` file or elsewhere,

Mate Requests (Level 3)

A user, when viewing a matelook page, should be able to send a *mate request* to the owner of that matelook page. The other user should be notified by an e-mail. The e-mail should containing an appropriate link to the web site which will allow them to accept or reject the mate request.

Mate Suggestions (Level 3)

Your web site should be able to provide a list of likely mate suggestions.
Your web site should display matelook users sorted on how likely the user is to know them. It should display a set (say 10) of matelook users. It should allow the next best-matching set of potential mates or the previous set of potential mates users to be viewed.

The user should be able to click on a potential mate , see their matelook page (where there will be able to send them a mate request).

Your matching algorithm should assume that people who have taken the same course in the same session may know each other.

For example Ralph Firman (dataset-medium/z5017258/user.txt) and Sheryl Crow (dataset-medium/z5024986/user.txt) are both taking *2016 S2 PSYC1011* which should cause your algorithm to make Ralph a more likely mate suggestion for Sheryl (and vice-versa).

Your matching algorithm should also assume that people may know mates of their mates.

You may choose to have your matching algorithm use other part of user details.

Note there are many choices in this matching algorithm so your results will differ from other students. Be prepared to explain how & why your matching algorithm works during your assignment demo. You may chose to have a development mode available which when turned on displays extra information regarding the matching.

Password Recovery (Level 3)

Users should be able to recover/change lost passwords via having an e-mail sent to them.

Uploading & Deleting Images (Level 3)

In addition to their profile image users should also be allowed to add a background image. A user should be able to upload/change/delete both background & profile images. The lecture CGI examples include one for uploading a file.

Editing Information (Level 3)

A user should be able to edit their details and change their profile images.

Deleting Posts (Level 3)

A *matelook* user should also be able to delete any of their posts, comments or replies,

Suspending/Deleting matelook Account (Level 3)

A *matelook* user not currently interested in matelook should be able to suspend their account. A suspended account is not visible to other users.
A *matelook* user should also be able to delete their account completely.

Notifications (Level 3)

A user should be able to indicate they wish to be notified by e-mail in one of these events:

- their zid is mentioned in a post, comment or reply
- they get a mate request

Including Links, Images & Videos (Level 3)

A user should be able to include links, images and videos in a post, comment or reply. These should be appropriately displayed when the item is viewed.

Privacy(Level 3)

A user should be able to make part or all of the content of their matelook page visible only to their mates.

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.

Getting Started

Here is the source (matelook.cgi.txt) to a Perl script (matelook.cgi) with crude partial implementation of Level 0. You may choose to use this script as a starting point for your assignment.
The same Perl but using `cgi.pm` shortcuts is also available (matelook.cgipm.cgi.txt). You may choose to start with instead this version if you prefer this style of coding.

A Python version of the same code is also available (matelook.py.cgi.txt). You may choose to start with this code if you prefer to the the assignment in Python. It is strongly recommended you use Python 3.5. The versions of Python 2 on CSE's CGI server are inconsistent with those on the lab machines and may be missing some packages.

However you start the assignment, make sure you name your script `matelook.cgi`.

All the above CGI scripts refer to a CSS file named matelook.css (matelook.css). It contains some simple CSS you can use as a starting point, but don't spend much time on CSS - almost all the marks are for coding!

You should use the gitlab.cse.unsw.edu.au repository you've been using for the lab CGI exercises for this assignment. You can do this be using the commands below.

```
$ cd
$ mkdir -p public_html/ass2
$ priv webonly public_html/ass2
$ cd public_html/ass2
$ git clone gitlab@gitlab.cse.unsw.EDU.AU:z5555555/16s2-comp2041-ass2 .
Cloning into '.'...
$ unzip /home/cs2041/public_html/assignments/matelook/dataset-medium.zip
....
$ chmod 700 matelook.cgi
$ chmod 644 matelook.css
$ git add matelook.cgi matelook.css
....
$ firefox http://cgi.cse.unsw.edu.au/~z5555555/ass2/matelook.cgi
$ vi matelook.cgi
$ vi diary.txt
$ git add diary.txt
$ git commit -a -m 'added code for basic formatting'
$ git push
....
$ git commit -a -m 'tidied up assignment for submission'
$ give cs2041 ass2 matelook.cgi diary.txt
$ git push
....
```

I expect most students will just work in their CSE account and push work to gitlab.cse.unsw.edu.au from CSE, but you can try setting up a git repository on your home machine and pushing work to gitlab.cse.unsw.edu.au from there.
If you do this you'll want to use git's pull command to update the repository in your CSE account.

```
$ git pull
Unpacking objects: 100% (3/3), done.
From gitlab@gitlab.cse.unsw.EDU.AU/z5555555/16s2-comp2041-ass2.git
   226cddf..e64fee9  master     -> origin/master
Updating 226cddf..e64fee9
Fast-forward
 matebook.cgi |    1 +
 1 file changed, 1 insertion(+)
```

You can complete this assignment just by running CGI scripts from CSE account. Some students find it convenient to install a webserver on their personal machine so they can run CGI scripts locally. There are multiple ways to this all of which depend on the software available on your machine. Here are instructions for running a webserver using Docker (docker.html).

Assumptions/Clarifications

It is a requirement of the assignment that when you work the assignment for more than a few minutes you push the work to `gitlab.cse,unsw.edu` (see above).
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 matelook.cgi You may submit other files used by your CGI script(s).

If you submit an executable named `init`, it will be run once before before your assignment, is in the same directory as your assignment. This is provide the possibility of set up code for complex assignments. I expect only a few student will need this.

Make sure you submit all files and in the appropriate hierarchy. If for example you 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.

Do not use a URL like `http://www.cse.unsw.edu.au/~z5555555/ass2/subdir/background.html` in your code - use a relative URL like `"subdir/background.html"`.

Similarly don't put a pathnames such as `/home/z5555555/public_html/ass2/subdir/datafile"` in your code - use a relative pathname such as `"subdir/datafile"`

You scripts will accessed via a **https** URL at the demo session so check it works with https, e.g. when you access it as `http://www.cse.unsw.edu.au/~z5555555/ass2/matelook.cgi`

For this reason do not use **http** URLs for external resources (e.g. Bootstrap) use a **https** URL.

We will use firefox(iceweasel) on CSE lab machines for the demo session. Your code should be sufficiently portable to work on with 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, subprocess, back quotes or open). where an equivalent operation could be performed simply in Perl/Python. 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 & Python.

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

You should follow discussion about the assignment in the course forum (https://piazza.com/unsw.edu.au/fall2016/comp2041comp9041/home). 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.

Diary

You must keep a record of your work on this assignment as you did for assignment in an ASCII file 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 zids

Include these notes in the files you submit as an ASCII file named diary.txt.
Some students choose to store this information in git commit messages and use a script to generate `diary.txt` from `git log` before they submit. You are welcome to do this.

Assessment

Assignment 2 will contribute 15 marks to your final COMP2041/9041 mark
Assignment 2 marking occurs in peer assessment sessions. Details of the peer assessment sessions will be announced in week 13. Your must attend one peer assessment sessions.

80% of the marks for assignment 2 will come frome your submitted CGI script being 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.

20% of the marks for assignment 2 will be awarded on the basis of clarity, commenting, elegance and style of your code.. In other words, your fellow students will assess how easy it is for a human to read and understand your program.

Here is an indicative marking scheme .

100%	Elegant Perl/Python with an excellent implementation of levels 0-3 and some optional (level 4) features
90+%	Very well written Perl/Python which implements levels 0-3 successfully
85%	Well written Perl/Python which implements most of levels 0-3 successfully
75+%	Readable Perl/Python 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.

The lecturer may vary the assessment scheme after inspecting the assignment submissions but its likely to be broadly similar to the above.

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 suspension from UNSW. 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.

Plagiarism or other misconduct can also result in loss of student visas.

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 October 30 Submit the assignment using this *give* command:

```
$ give cs2041 ass2 matelook.cgi matelook.css diary.txt [files.tar] [any-other-files]
```

If you have need to submit many other files, files in a sub-directory or a directory hierarchy, submit them as a tar file named files.tar. For example if you have subdirectories named *css*, *js* and *images*, this will submit all the files in them (including directories they contain).
Do not submit the datasets unless you have changed them.

```
$ tar -Jcf files.tar css js images
$ give cs2041 ass2 matelook.cgi diary.txt files.tar
```

If your assignment is submitted after this date, each hour it is late reduces the maximum mark it can achieve by 1%. For example if an assignment worth 76% was submitted 5 hours late, the late submission would have no effect. If the same assignment was submitted 30 hours late it would be awarded 70%, the maximum mark it can achieve at that time.