Assignment 2 - UNSWtalk

Software Construction (http://www.cse.unsw.edu.au/~cs2041/17s2/)

Aims

This assignment aims to give you:

- an introduction to the issues involved in programming for the web.
- experience in constructing a web application.
- Perl/Python programming generally.

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 . Andrew's plan is to have COMP[29]041 students create a social media platform for UNSW students called *UNSWtalk* Because Andrew is unaware of any other competing social media platforms, he thinks *UNSWtalk* will become very popular and he will become rich.
He wants *UNSWtalk* to allow students to post messages, other students to comment on these messges and replies to these comments to be added.

He wants *UNSWtalk* to allows students to indicate other students are their *friends*.

Your task is to produce either Python or Perl code which provides the core features of *UNSWtalk*.

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 datasets containing the *UNSWtalk* details of (synthetic) UNSW students & their postings:

- small (dataset-small) (4Mb unpacked) zip (dataset-small.zip) (0.4Mb) tar.xz (dataset-small.tar.xz) (0.1Mb) - 10 students, 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 students, 420 posts, 2520 comments, 1749 replies
- large (dataset-large) (340Mb unpacked) zip (dataset-large.zip) (28Mb) tar.xz (dataset-large.tar.xz) (8Mb) - 420 students, 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.

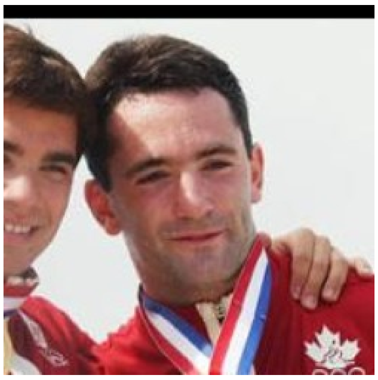
The information for each student is in a separate directory named with their zid name For example in the *medium* dataset UNSW student Alexandre Despatie has zid z5192930 so his information is in the directory: `dataset-medium/z5198807/` (dataset-medium/z5198807/)

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

For example dataset-medium/z5198807/student.txt (dataset-medium/z5198807/student.txt) contains Alexandre's details:

```
home_latitude: -33.7492
courses: (2017 S1 ACCT1501, 2017 S1 COMP1511, 2017 S1 ECON1101, 2017 S1 MATH1131, 2017 S2 COMP1531, 2017 S2 C
home_longitude: 151.1093
program: Commerce / Computer Science
zid: z5198807
email: z5198807@unsw.edu.au
birthday: 1999-05-29
friends: (z5194082, z5190129, z5196684, z5191841, z5197178, z5191824, z5193757, z5194685)
password: cream
full_name: Alexandre Despatie
```

Note Alexandre has supplied the suburb where he lives and the coordinates of his home address. Notice also the list of Alexandre' friends.
Most *UNSWtalk* students will also have an image present in the same directory named `img.jpg`. For example `dataset-medium/z5198807/img.jpg` (dataset-medium/z5198807/img.jpg) contains Alexandre's image.



Note some details may be missing for some *UNSWtalk* students. This is deliberate, it indicates the *UNSWtalk* student has chosen not to supply this information and your web site should handle this sensibly. Also images might not be present for all students. Again your web site should handle this sensibly.

Also present in a students's *UNSWtalk* directory may be files containing posts they've made in numbered sub-directories.

For example dataset-large/z5196613/4.txt (dataset-large/z5196613/4.txt) contains:

```
message: To the cute indian guy (I think your name is Arian..?)\nYou looked like a total hottie in the Elec tu
from: z5196613
longitude: 151.2261
time: 2016-04-27T12:52:42+0000
latitude: -33.9081
```

Also in students's *UNSWtalk* directory may be files with `comments` on posts.
For example dataset-large/z5196613/4-2.txt (dataset-large/z5196613/4-2.txt) contains:

```
time: 2016-04-26T07:55:01+0000
from: z5193182
message: HAHAHAHAHAHA okay which one of you was this z5192183 z5192541 z5196164 z5197797
```

Note this mentions 4 other *UNSWtalk* students using the *UNSWtalk* convention of referencing other students with their zid.
Note this is a comment on the post dataset-large/z5196613/4.txt (dataset-large/z5196613/4.txt)

Also in students's *UNSWtalk* directory may be files with `replies` to comments.

For example dataset-large/z5196613/4-2-1.txt (dataset-large/z5196613/4-2-1.txt) contains:

```
message: omg i really hope this is legit
from: z5192183
time: 2016-04-26T08:10:31+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 *UNSWtalk* student details, posts and comments because *UNSWtalk* students choose not to supply them.
While you do not have to use this format to store data, 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 *UNSWtalk* students.

Before choosing to use a database to store *UNSWtalk* 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 assignment, 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 Student Information & Posts (Level 0)

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

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

The student'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 students. 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 UNSWtalk but the primary CGI script has to be `UNSWtalk.cgi`.

Friend list (Level 1)

When a UNSWtalk page is shown a list of the names of that student's friends should be displayed.
The list should include a thumbnail image of the friend (if they have a profile image).

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

Search for Names (Level 1)

Your web site should provide searching for a student whose name contains 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 UNSWtalk page.

Logging In & Out (Level 1)

Students should have to login to use UNSWtalk.
Their password should be checked when they attempt to log in.

Students should also be able to logout.

Displaying Posts (Level 2)

When a student logs in they should see their recent posts, any posts from their friends 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 student's UNSWtalk page. The link text should be the student's full name.

Making Posts (Level 2)

Students 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.

Friend/Unfriend Students (Level 3)

A student should be able to add & delete students from their friend list.

Pagination of Posts & Search Results (Level 3)

When searching for students or posts and when viewing posts the students 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.

Student Account Creation (Level 3)

Your web site should allow students 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 the e-mail address is valid and owned by the *UNSWtalk* 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 student you would need create a directory named with their zid and then add a appropriate `student.txt` containing their details.

Profile Text (Level 3)

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

```
My interests are long walks on the beach and Python programming.
I plan to use what I've learnt COMP9041 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 include any examples of such text.
You can choose to store this text in the `student.txt` file or elsewhere.

Friend Requests (Level 3)

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

Friend Suggestions (Level 3)

Your web site should be able to provide a list of likely friend suggestions.
Your web site should display UNSWtalk students sorted on how likely the student is to know them. It should display a set (say 10) of UNSWtalk students. It should allow the next best-matching set of potential friends or the previous set of potential friends to be viewed.

The student should be able to click on a potential friend, see their UNSWtalk page (where they will be able to send them a friend request).

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

For example Reese Witherspoon (dataset-medium/z5198410/student.txt) and James Franco (dataset-medium/z5198491/student.txt) are both taking *2017 S2 MATH1231* which should cause your algorithm to make Reese a more likely friend suggestion for James (and vice-versa).

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

You may choose to have your matching algorithm use other parts of student 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)

Students 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 jpg image students should also be allowed to add a background image. A student 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 student should be able to edit their details and change their profile images.

Deleting Posts (Level 3)

A *UNSWtalk* student should also be able to delete any of their posts, comments or replies.

Suspending/Deleting UNSWtalk Account (Level 3)

A *UNSWtalk* student not currently interested in UNSWtalk should be able to suspend their account. A suspended account is not visible to other students.
A *UNSWtalk* student should also be able to delete their account completely.

Notifications (Level 3)

A student 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 friend request

Including Links, Images & Videos (Level 3)

A student 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 student should be able to make part or all of the content of their UNSWtalk page visible only to their friends.

Advanced Features (Level 4)

If you wish to obtain over 90% you should consider adding features beyond those above. Marks will be available for extra features.

Getting Started

Setting Up Your Git Repository

You should first clone the gitlab.cse.unsw.edu.au repository for this assignment

```
$ cd
$ mkdir -p public_html/ass2
$ priv webonly public_html/ass2
$ cd public_html/ass2
$ git clone gitlab@gitlab.cse.unsw.EDU.AU:z5555555/17s2-comp2041-ass2 .
Cloning into '.'...
$ chmod 644 .htaccess
$ unzip /web/cs2041/17s2/assignments/UNSWtalk/dataset-medium.zip
....
```

If the git clone above fails - redo the instructions in assignment 1 for adding your ssh key to gitlab.
And if that fail try with a https url instead (again replacing 5555555 with your zid) - and supplying your zid/zpass when requested:

```
$ git clone https://gitlab.cse.unsw.edu.au/z5555555/17s2-comp2041-ass2.git .
Username for 'https://gitlab.cse.unsw.edu.au': z5555555
Password for 'https://z5080336@gitlab.cse.unsw.edu.au':
```

Starting Code

Here is the source (UNSWtalk.cgi.txt) to a Perl script with crude partial implementation of Level 0. You may choose to use this script as a starting point for your assignment.
This CGI scripts refers to a CSS file named UNSWtalk.css (UNSWtalk.css). It contains some simple CSS you can use as a starting point, but don't spend too much time on CSS - almost all the marks are for coding!

A Flask Python version of the same code is also available as a zip file (flask.zip). You may choose to start with this code if you prefer to do the assignment in Python. The zip file includes UNSWtalk.css (UNSWtalk.css).

Then if you want to use the starting code Python and Flask do this:

```
$ unzip /web/cs2041/17s2/assignments/UNSWtalk/flask.zip
...
$ git add static templates UNSWtalk.cgi UNSWtalk.py
```

If instead you want to use the starting code for Perl do this:

```
$ cp /web/cs2041/17s2/assignments/UNSWtalk/UNSWtalk.cgi .
$ cp /web/cs2041/17s2/assignments/UNSWtalk/UNSWtalk.css .
$ chmod 644 UNSWtalk.css
$ chmod 700 UNSWtalk.cgi
$ git add UNSWtalk.cgi UNSWtalk.css
```

In both cases should be able to run the starting code as a CGI script by replacing z5555555 in the url below with your zid:

```
$ firefox http://cgi.cse.unsw.edu.au/~z5555555/ass2/UNSWtalk.cgi &
```

If you are using Flask you run the starting code directly on a CSE lab machine:

```
$ ./UNSWtalk.py
 * Running on http://127.0.0.1:5000/
 * Restarting with reloader
```

```
$ firefox  http://127.0.0.1:5000/ &
```

This won't work on a CSE login server or Vlab.

Updates To Python starting Code

The first line of the Python starting point code has been updated to be `#!/web/cs2041/bin/python3.6.3` `/web/cs2041/bin/python3.6.3` a Python version with an upto date set of modules installed.

The script `UNSWtalk.cgi` allows `UNSWtalk.py` to be run as a CGI script has been updated since the assignment was released to give useful information that than just a 500 error when an exception occurs in `UNSWtalk.py`. You can get the latest version with this command:

```
$ cp -p /web/cs2041/17s2/assignments/UNSWtalk/flask/UNSWtalk.cgi  ~/public_html/ass2/UNSWtalk
```

DO NOT DO THIS if you are doing the assignment in Perl - it would overwrite your Perl in `UNSWtalk.cgi`.

Pushing to Gitlab

Every time you work on the assignment you must push work to gitlab

```
$ vi UNSWtalk.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'
$ git push
$ give cs2041 ass2 UNSWtalk.cgi diary.txt
....
```

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.
You will need to use a CSE vpn to access gitlab from outside UNSW.

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/z5555555/17s2-comp2041-ass2.git
   226cddf..e64fee9  master     -> origin/master
Updating 226cddf..e64fee9
Fast-forward
 UNSWtalk.cgi |    1 +
 1 file changed, 1 insertion(+)
```

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).
You are permitted to use Perl or Python to implement this assignment. You may not use other languages except 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 - make 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 UNSWtalk.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, 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 `https://www.cse.unsw.edu.au/~z5555555/ass2/UNSWtalk.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 student'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/class/j5ji4vjjra62a3). 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.

Here are two example functions that send email in Python (send_email.py.txt) and Perl (send_email.pl.txt)

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 student 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
HD++%	Very well written Perl/Python which implements levels 0-3 successfully
HD (85+)	Well written Perl/Python which implements most of levels 0-3 successfully
DN (75+)	Readable Perl/Python which implements of levels 0-2 successfully
CR (65+)	Reasonably readable code which implements level 0-1 successfully
PS (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 29. Submit the assignment using *give*.
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.

For example if you complete the assignment using Python and had files in the sub-directories `static` and `templates`, you might submit the assignment using these commands:

```
$ tar -Jcf files.tar static templates
$ give cs2041 ass2 UNSWtalk.cgi UNSWtalk.py diary.txt files.tar
```

For example if you complete the assignment using Perl and had files in the sub-directories `css`, `js` and `images`, you might submit the assignment uysing these commands:

```
$ tar -Jcf files.tar css js images
$ give cs2041 ass2 UNSWtalk.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 0.5% for the first 48 hours.
After the first 48 hours, each hour it is late reduces the maximum mark it can achieve by 4 marks an hour.

For example if an assignment worth 90% 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 85%, the maximum mark it can achieve at that time.

(flask)