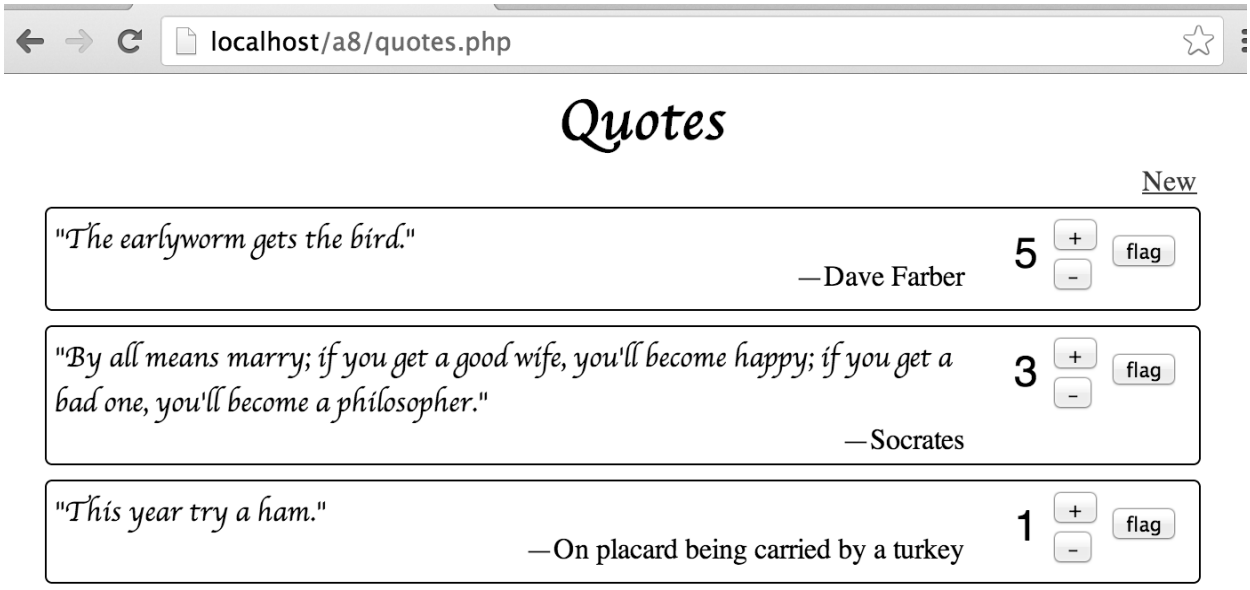
CSC 337, Fall 2013
Assignment 8
Due: Wednesday, November 20 at 22:00:00

Problem 1. (35 points) **`quotes.php`**

In this problem you are to create a PHP web application that maintains a MySQL table of quotations. Quotes can added, voted up or down, and flagged. Here's what the user might see:

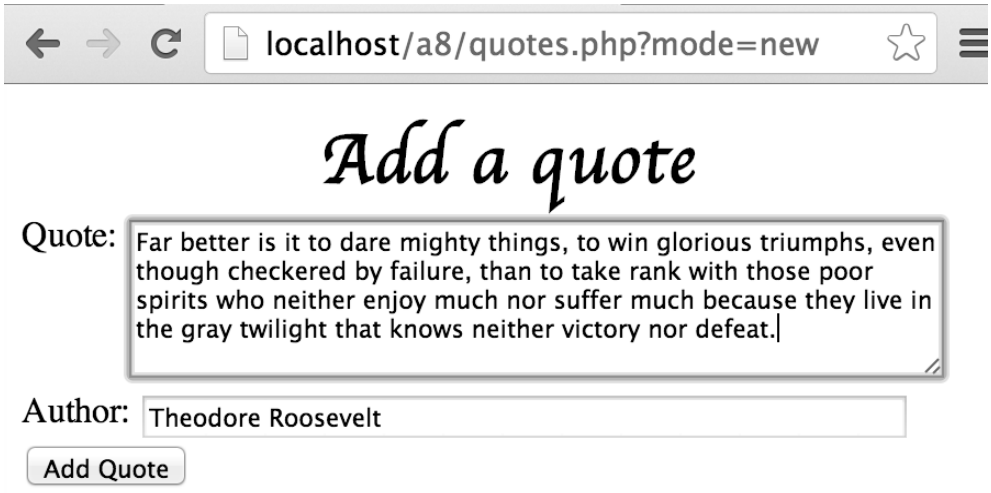


Quotes are displayed in descending order by rating, which is displayed in a larger font to the right of the quote. If two quotes have the same rating, it doesn't matter which one is displayed first.

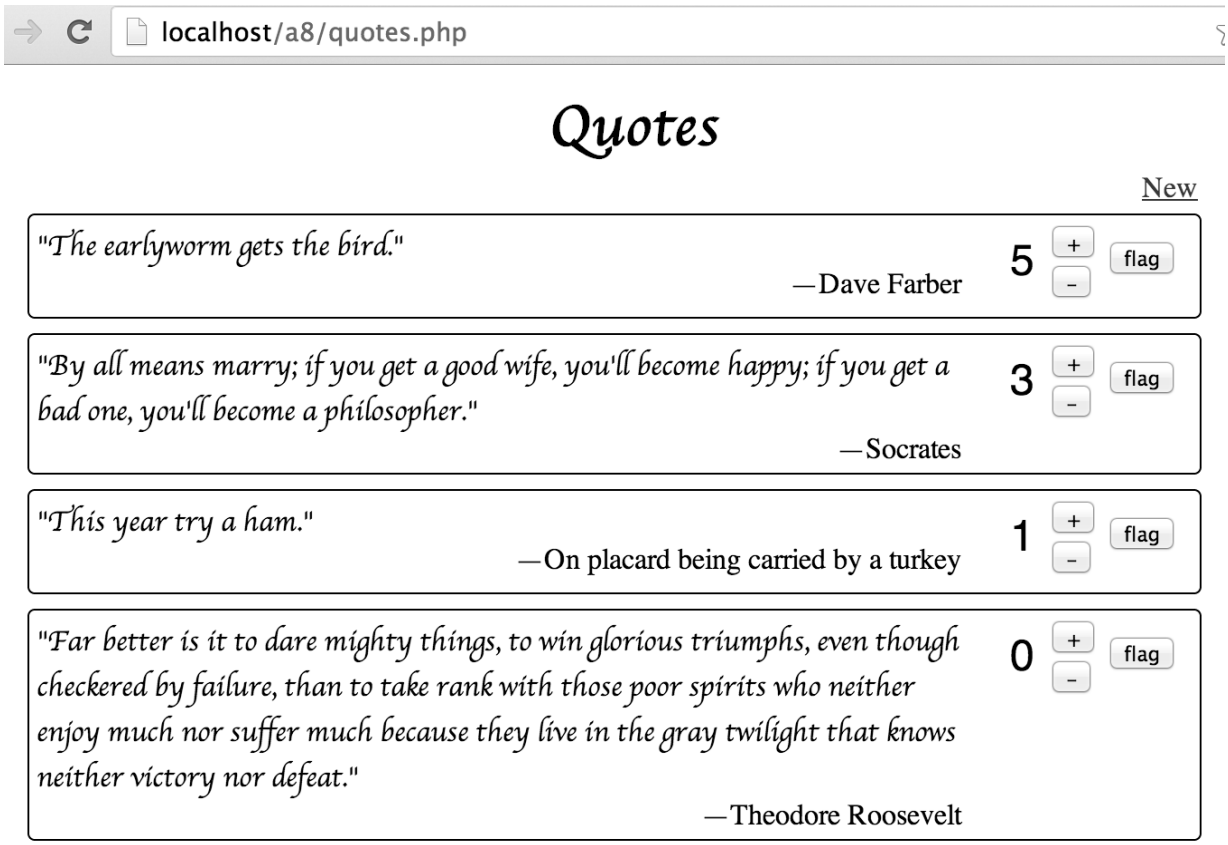
Clicking the plus and minus buttons raise or lower the rating by 1. Ratings may go negative.

If a quote is flagged, it disappears from view but is not deleted from the database.

Clicking on the New link brings up a simple form to enter a new quote:



Upon clicking Add Quote the full list of quotes, including the new one, is displayed. New quotes start with a rating of zero.



Here's a link for a video demo: http://screencast.com/t/AiY3oTkjKhrq

You can see that there's some styling but not much. The important part is the behavior, so put your time into that and do some styling only if time permits.

The quotes and associated data are held in a single MySQL table; the data is **not** represented as cookies, hidden values, etc.

The above is the description of the problem itself. What follows below are a number of technical details you'll want to know.

`$a8/quotes.sql`

`$a8/quotes.sql` has SQL statements to create a table named `quote` and then insert some rows in it. At the moment it looks like this:

```
drop table if exists quote;
create table quote
(
 id         bigint not null auto_increment primary key,
 added      datetime,
 quote      varchar(2000) not null,
 author     varchar(100) not null,
 rating     int not null,
 flagged    char(1) not null
 );
```

```
insert into quote(added, quote, author, rating, flagged)
     values(now(), 'The earlyworm gets the bird.', 'Dave
     Farber', 0, 'f');
insert into quote(added, quote, author, rating, flagged)
     values(now(), 'By all means marry; if you get a good wife,
     you''ll become happy; if you get a bad one, you''ll become
     a philosopher.', 'Socrates', 0, 'f');
insert into quote(added, quote, author, rating, flagged)
     values(now(), 'This year try a ham.', 'On placard being
     carried by a turkey', 0, 'f');
```

Note that it starts by dropping the `quote` table. This is common practice in a database initialization script—drop the table, create the table, insert rows. Repeat for each table but here we've only got one table.

Note the uses of two consecutive apostrophes in the quotation from Socrates:

```
...you''ll become happy; if you get a bad one, you''ll become...
```

That's a SQL way of escaping an apostrophe: specify two apostrophes! Some older programming languages use that same mechanism, too. With MySQL you can use `\'`, too.

Note the second column in the quote table:

```
create table quote
(
 ...
 added      datetime,
 ...
 );
```

The column is named `added` and its type is `datetime`. It represents a precise moment in time—a date and a time on the date, including a timezone. The `insert` statements in `quotes.sql` look like this:

```
insert into quote(added, ...) values(now(), ...);
```

Rather than specifying a literal value for the date and time, we simply call MySQL's `now()` function to produce the current time. **You'll want to do the same thing with the `insert(...)` statement created by your program.** This version of `quotes.php` doesn't make any use of `added`.

You need a database to put the `quote` table in!

You'll need a MySQL database to hold the `quote` table. If you've got XAMPP installed on your machine you can certainly use XAMPP's MySQL. If you're working on lectura, you can make a database on `mysql.cs.arizona.edu` that can then be accessed on the CS machines. Both options are described next, in turn.

Making and using a `quotes` database with XAMPP's MySQL

If you're using XAMPP's MySQL, use the XAMPP control panel to be sure the MySQL server is running. Then use the `mysql` command to create a database named `quotes` and process the SQL in `quotes.sql`, assuming you've downloaded `quotes.sql` from `$a8`. Here are the commands to do that:

```
Z:\whm\337\a8>c:\xampp\mysql\bin\mysql -u root
```

```
Welcome to the MySQL monitor.  Commands end with ; or \g.
...
mysql> create database quotes;
Query OK, 1 row affected (0.00 sec)

mysql> use quotes;
Database changed

mysql> source quotes.sql;
Query OK, 0 rows affected, 1 warning (0.00 sec)

Query OK, 0 rows affected (0.02 sec)

Query OK, 1 row affected (0.00 sec)

Query OK, 1 row affected (0.00 sec)

Query OK, 1 row affected (0.00 sec)

mysql> select * from quote;
[should display three rows, albeit with lots of wrapping]
3 rows in set (0.02 sec)
```

Once the database has been created, you can name it on the `mysql` command line instead of having to type "`use quotes;`" at the `mysql>` prompt.

```
Z:\whm\337\a8> c:\xampp\mysql\bin\mysql -u root quotes

mysql> select count(*) from quote;
+----------+
| count(*) |
+----------+
|        3 |
+----------+
1 row in set (0.00 sec)
```

If you want to "reset" your database, you can do this: (there's no output unless there's an error)

```
Z:\> c:\xampp\mysql\bin\mysql -u root quotes < quotes.sql
```

If you want to make a backup of your database, do this: (note redirection of output to `myquotes.sql` and use of `mysql`**`dump`**)

```
Z:\> c:\xampp\mysql\bin\mysqldump -u root quotes > myquotes.sql
```

To reload from that backup, do this:

```
Z:\> c:\xampp\mysql\bin\mysql -u root quotes < myquotes.sql
```

Note that in each case the database name, `quotes`, is specified on the command line.

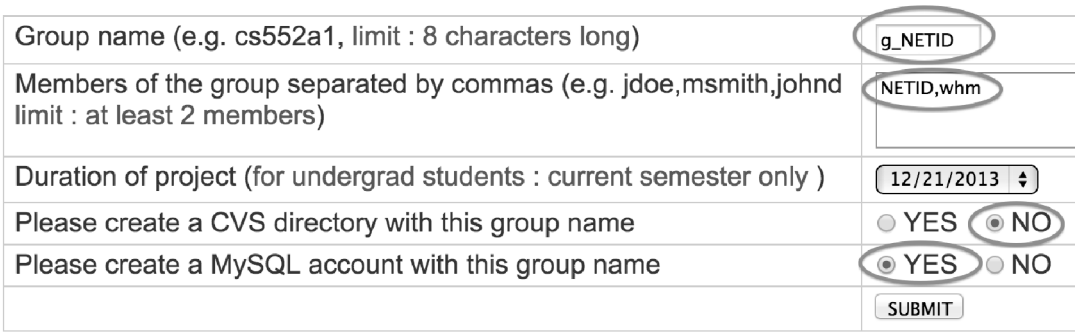
Making and using a database with MySQL on the CS machines

Things are more complicated if you're working with MySQL on the CS machines. The most important thing to understand is that if you're using XAMPP's MySQL, **you** are the database administrator . You can pick the database name and create the database The database name used above is "quotes". On the CS machines you do not have that administrative capability. Instead, you need to request that a database be

created for you. Make that request by hitting this page:

```
http://cgi.cs.arizona.edu/~apply/services/group_create.php
```

You'll see a form like this:



You might be inclined to make the group name be your NetID but that alone won't work. Maybe prefix it with "`g_`", but you can try other things as well. You'll be notified immediately if it's already in use.

The group needs two members, so name `whm` as the second member.

You don't need CVS, so click the "NO" radio button for it.

The whole purpose of this is to create a MySQL database you can use, so click "YES" for a MySQL account, which includes a database.

Click SUBMIT, of course. (Look! It's a PHP program that you're using to request a database!)

Database creation is not instantaneous but within maybe 15 minutes you should receive an email message with the subject "MySQL database (...) created".

Our TA Jackson followed the instructions above and used `g_jcw21` for a group name. The database-created message he got included this:

> To access this database you can use a CS linux box with mysql installed and enter commands such as:
>
> /usr/bin/mysql -h mysql.cs.arizona.edu -p -u g_jcw21 jcw21_g_jcw21
> Enter password: secret

We can see the user name is g_jcw21 and the database name is jcw21_g_jcw21. (Yes, it's repetitious!)

(Again, those using XAMPP can create and use a database named `quotes` but those using MySQL on the CS machines need to use the generated database name. In this case that name is `jcw21_g_jcw21`, not `quotes`!)

To create the `quote` table in the database `jcw21_g_jcw21`, do this on lectura:

```
mysql -h mysql -p -u g_jcw21 jcw21_g_jcw21 < /cs/www/classes/cs337/fall13/a8/quotes.sql
```

It will prompt for the password sent in the mail message.

Ouch! That's a lot to type and you might need to do it again from time to time. I recommend setting up `bash` aliases for `msq` and `msd`, and an `a8` variable. Sadly, it's not simple to do that because depending

on when your CS account was created you might have any one of several configurations for bash start-up files. In my 352 UNIX slides, which are on the Piazza Resources page, slides 110-111 talk about `bash` start-up files and the various possibilities you may see. A "shotgun" approach is to add the same three lines at the end of each of the four files `.bash_profile`, `.bash_login`, `.profile`, and `.bashrc` in your home directory. Here are those three lines:

```
alias msq="mysql -h mysql -p -u DBLOGIN DBNAME"
alias msd="mysqldump -h mysql -p -u DBLOGIN DBNAME"
a8=/cs/www/classes/cs337/fall13/a8
```

In the example above, `DBLOGIN` would be `g_jcw21` and `DBNAME` would be `jcw21_g_jcw21`.

Match the quotes and spacing exactly!

If any of those four files don't exist, skip it. **When done, log out and back in.** You should then be able to see the alias and variable like this:

```
% alias msq
alias msq='mysql -h mysql -p -u g_jcw21 jcw21_g_jcw21'
% alias msd
alias msd='mysqldump -h mysql -p -u g_jcw21 jcw21_g_jcw21'
% echo $a8
/cs/www/classes/cs337/fall13/a8
```

After doing all that once you'll then able to create/reload the quote table like this:

```
% msq < $a8/quotes.sql
Enter password:
% (back to shell prompt)
```

You can do queries like this:

```
% msq
Enter password:
mysql> select * from quote;
[should display three rows, albeit with lots of wrapping]
3 rows in set (0.00 sec)
```

If you want to make a backup of your database, do this: (note redirection of output to `myquotes.sql` and use of `msd`, not `msq`)

```
% msd > myquotes.sql
Enter password:
```

To reload from that backup, do this:

```
% msq < myquotes.sql
Enter password:
```

Miscellaneous

If you want to keep your quotes but reset their ratings and clear flaggings, you can use SQL's `update` command:

```
mysql> update quote set flagged='f', rating=0;
Query OK, 4 rows affected (0.03 sec)
```

```
Rows matched: 4  Changed: 4  Warnings: 0
```

If you want to change your database password, do this:

```
mysql> set password for 'g_jcw21'@'%' = password('newpw');
Query OK, 0 rows affected (0.00 sec)
```

DON'T use a valuable password, like your NetID password, for your MySQL database password!

HTTP redirects with PHP's `header()` function

Something we haven't yet talked about in class but that you might find handy is doing a "redirect" with PHP's `header(...)` function. Here's a trivial example: (`redirect1.php`)

```
<?php
header("Location: http://google.com");
```

If you hit `cgi.cs.arizona.edu/classes/cs337/fall13/redirect1.php` with your browser, you'll find yourself at `google.com`!

`curl` shows this:

```
% curl -i cgi.cs.arizona.edu/classes/cs337/fall13/redirect1.php
HTTP/1.1 302 Moved Temporarily
Date: Tue, 12 Nov 2013 23:25:06 GMT
Server: Apache/2.2.22 (Ubuntu)
X-Powered-By: PHP/5.3.10-1ubuntu3.8
Location: http://google.com
Vary: Accept-Encoding
Content-Length: 0
Content-Type: text/html
```

Note the server's response: `302 Moved Temporarily`. When the browser sees that 302 response it automatically hits the URL specified in the `Location:` header; no user interaction is required.

Try hitting `redirect1.php` with DevTools' Network tab active.

In `quotes.php` you might use `header("Location: quotes.php")` at the end of your functions to add quotes and change ratings. The above essentially says, "hit me", with the result that the app is run from scratch, as if you'd hit `http://localhost/a8/quotes.php`, for example.

Problem 2. (5 Project Points) `project3.html`

Sometimes the hardest step of a project is writing that first line of code. To get these five project points you need to do at least enough on your project to have something renderable with a browser.

In `project3.html` tell me what you've done thus far. Include one or more screen shots, too, naming them `project3a.*`, `project3b.*`, etc. Also turn in a snapshot of whatever HTML/CSS/PHP files you've developed thus far. They don't need to be error-free or work in any way, beyond satisfying the "something renderable" requirement above.

Problem 3. Extra Credit `observations.html`

Submit a styled-as-you-like HTML file named `observations.html` with...

(a) (1 point extra credit) An estimate of how long it took you to complete this assignment. To facilitate programmatic extraction of the hours from all submissions, please enclose the number of hours in a `span` with `class=hours`, like this:

```
I spent <span class=hours>8.5</span> hours on this assignment.
```

Other comments about the assignment are welcome, too. Was it too long, too hard, too detailed? Speak up! I appreciate all feedback, favorable or not.

(b) (1-3 points extra credit) Cite an interesting course-related observation (or observations) that you made while working on the assignment. The observation should have at least a little bit of depth. Think of me saying "Good!" as one point, "Interesting!" as two points, and "Wow!" as three points. I'm looking for quality, not quantity.

Turning in your work

Use the D2L Dropbox named `a8` to submit a zip file named `a8.zip` that contains all your work. If you submit more than one `a8.zip`, we'll grade your final submission. Here's the full list of deliverables:

`quotes.php`
`project3.html`, **and associated files**
`observations.html` (for extra credit)

Note that all characters in the file names are lowercase.

Have all the deliverables in the uppermost level of the zip. It's ok if your zip includes other files, too.

Miscellaneous

The HTML slides, the CSS slides, the PHP slides, the RDBMS slides through the Thanksgiving-in-a-Box example, in-class examples, and a maybe little reading on `php.net` about the PDO stuff should be all you need to complete this assignment.

Point values of problems correspond directly to assignment points in the syllabus.

`$a8` follows the convention of `$a`*N* on the previous assignments.

Remember that late assignments are not accepted and that there are no late days; but if circumstances beyond your control interfere with your work on this assignment, there may be grounds for an extension. See the syllabus for details.

My estimate is that it will take a typical CS junior from three to four hours to complete this assignment, not counting the project-related work

Keep in mind the point value of each problem; don't invest an inordinate amount of time in a problem or become incredibly frustrated before you ask for a hint or help. Remember that the purpose of the assignments is to build understanding of the course material by applying it to solve problems. If you reach the seven-hour mark, regardless of whether you have specific questions, it's probably time to touch base with us. Give us a chance to speed you up! **Our goal is that everybody gets 100% on this assignment AND gets it done in an amount of time that is reasonable for them.**