Summary of this week's projects

Exercise 1. Guest Book using MySQL.

1. Form for signing the guestbook
2. Form handler for entering guest book information into the database
3. Display the contents of the Guest Book

Exercise 2. Random Proverb.

1. Display a random proverb in the footer of the Project web site.

Exercise 3. Add Proverb.

1. Provide a form and database code to let the user add a new proverb to your database.

Exercise 4. Final Project checkpoint.

1. Create a file called forms.php.
2. Add the addproverbs file to the Project.
3. Add the guest book files to the Project.
4. Add the alphabetize files to the Project.

Exercise 5. Link everything to your home page.

Exercise 1. Guest Book using MySQL.

Like the project from last week, this guest book will consist of 3 files:

[1] Form to sign the guest book;
[2] Form handler to write to the guest book file;
[3] Script to show the contents of the guest book.

File 1. Form.

1. Create a folder and call it Chapter10. Your work will go in this folder.
2. Create an HTML file. Save it as `guestbook.html`.
3. Put this form in the `<body>` section of the `guestbook.html` file:

```
<p>Enter your name to sign the guest book.</p>

<form method='post' action='signguestbook.php'>

<p>First name: <input type='text' name='firstname' /></p>
<p>Last name: <input type='text' name='lastname' /></p>
<p><input type='submit' value='Add to Guest Book' /></p>

</form>

<a href='showguestbook.php'>Show guest book</a>
```

continued on next page →

File 2. Form handler.

1. Put your `login.php` file from last Thursday night (or see textbook page 234) in a place where you can include it, either in the root level of your web site (`htdocs`, `www`, or `public_html`) or in the `Chapter10` folder.
2. Create a PHP file. Save it as `signguestbook.php`.
3. This file should be a complete HTML file with DOCTYPE, head, body, etc.
4. Add this code in a PHP script section in the body. You may have to modify the `login.php` line depending on exactly where you put this file and what you named it.

```
include "../login.php";
$ok = true;

if (empty($_POST['firstname']) || empty($_POST['lastname'])) {
  echo "<p>You must enter your first and last name.</p>\n";
  echo "<p><a href='guestbook.html'>Go back to guest book</a></p>\n";
  $ok = false;
}

if ($ok) {
  $conn = new mysqli ($hn, $un, $pw, $db);
  if ($conn->connect_error) {
    echo "<p>Unable to connect to database. {$conn->connect_error}</p>\n";
    $ok = false;
  }
}
```

5. Then, add this code that will create the guestbook table in your database. So, you won't need to log in and create this table from the command line. This PHP module will create the table if it does not already exist. This is not a typical practice. But it is helpful because once your code works properly on your home server, it will also install this table on the Mission College server.

```
if ($ok) {
  $tablename = 'visitors';
  $query = "SHOW TABLES LIKE '$tablename'";
  $result = $conn->query($query);
  if ($result->num_rows == 0) {
    $query = "CREATE TABLE $tablename (id INT NOT NULL AUTO_INCREMENT
         PRIMARY KEY, lastname VARCHAR(64), firstname VARCHAR(64))";
    $result = $conn->query($query);
    if ($result === FALSE) {
      echo "<p>Unable to create the table. {$conn-> connect_error}</p>\n";
      $ok = false;
    }
  }
}
```

continued on next page →

6. This last section will insert the new entry into the table, disconnect from the server, and print a link back to the guest book form.

```
if ($ok) {
  $lastname = stripslashes ($_POST['lastname']);
  $firstname = stripslashes ($_POST['firstname']);
  $query = "INSERT INTO $tablename VALUES(NULL, '$lastname',
      '$firstname')";
  $result = $conn->query($query);
  if ($result === FALSE) {
    echo "<p>Unable to add to guest book. {$conn-> connect_error}</p>\n";
    $ok = false;
  }
}

if ($ok) {
  echo "<p>Thank you for signing our guest book, $firstname
$lastname!</p>\n";
  $conn->close();
}

echo "<p><a href='guestbook.html'>Go back to guest book</a></p>\n";
```

File 3. Show Guest Book.

1. Create a new PHP file. Save it as `showguestbook.php`.
2. This file should be a complete HTML file with DOCTYPE, head, body, etc.
3. Put this code in a PHP code block. This first section checks that the table exists.

```
include "../login.php";
$ok = true;

$conn = new mysqli ($hn, $un, $pw, $db);
if ($conn->connect_error) {
  echo "<p>Unable to connect to database. {$conn->connect_error}</p>\n";
  $ok = false;
}

if ($ok) {
  $tablename = 'visitors';
  $query = "SHOW TABLES LIKE '$tablename'";
  $result = $conn->query($query);
  if ($result->num_rows == 0) {
    echo "<p>There are no entries in the guest book.</p>\n";
    $ok = false;
  }
}
```

continued on next page →

4. This next section checks to see if there are any entries in the table.

```
if ($ok) {
  $query = "SELECT * FROM $tablename";
  $result = $conn->query($query);
  if ($result->num_rows < 1) {
    echo "<p>There are no entries in the guest book.</p>\n";
    $ok = false;
  }
}
```

5. This last section prints the list of visitors in a table, disconnects from the database, and prints a link back to the guest book form.

```
if ($ok) {
  echo "<p>The following visitors have signed the guest book:</p>\n";
  echo "<table>\n";
  echo "<tr><th>First Name</th><th>Last Name</th></tr>\n";
  for ($i=0; $i<$result->num_rows; $i++) {
    $result->data_seek($i);
    $row = $result->fetch_assoc();
    echo "<tr>\n";
    echo "<td>{$row['firstname']}</td>\n";
    echo "<td>{$row['lastname']}</td>\n";
    echo "</tr>\n";
  }
  echo "</table>\n";
}

if ($ok) {
  $conn->close();
}

echo "<p><a href='guestbook.html'>Go back to guest book</a></p>\n";
```

Exercise 2. Random proverbs using MySQL.

1. Last week, you made a database table called `proverbs`. Now, we will display the contents of this proverb table.
2. If you haven't done so already, create a new folder, at the root level of your web site, and call it `Project` or `FinalProject`. *Do not* create this folder inside the `Chapter10` folder. It should be an entirely new folder. Your web site structure should look *something like* this:

continued on next page →

3. ▶ public_html
    a. ❑ index.php
    b. ❑ styles.css
    c. ▶ Chapter04
    d. ▶ Chapter05
    e. ▶ Chapter06
    f. ▶ Chapter07
    g. ▶ Chapter08
    h. ▶ Chapter10
    i. ▶ Chapter11
    j. ▶ FinalExam
    k. ▶ Midterm
    l. ▶ Project
4. Copy your project code from Assignment 11a, Exercise 4 into this new `Project` folder.
5. In `footer.php`, create a MySQL query to display a random proverb from the proverbs table.
    a. Use the MySQL `num_rows` property to find out how many proverbs are in the table.
    b. Use the PHP `rand()` function to choose a random proverb to display.
    c. Use the `data_seek()` method to choose that row out of the table.
6. Each time you retrieve a proverb from the table, update the displayed field for that record to show that it has been displayed one additional time. You can use a query something like this. (I have not tested this query, but it's in the textbook.) `$currentID` is the number of the proverb you chose to display.

```
$query = "UPDATE proverbs SET displayed = displayed + 1
      WHERE proverb = $currentID";
```

Exercise 3. Add proverbs to the database.

1. Create a form in an HTML and a form handler in PHP.
2. You may make an all-in-one form if you like, and call it `addproverb.php`. **Or** you may make a two-part form and call the form `addproverb.html` and the handler `addproverb.php`.
3. The form just has a single text input field (or textarea) that allows the user to enter a new proverb, and a submit button.
4. In the MySQL monitor program, add this line to make the `number` field in the `proverbs` table set to AUTO_INCREMENT:

```
ALTER TABLE proverbs MODIFY COLUMN number INT auto_increment;
```

5. Add the new proverb to the `proverbs` table. When adding, use `NULL` for the `number` field (it is set to `AUTO_INCREMENT` so it will automatically get the correct numerical value) and `0` for the `displayed` field, because it has never been displayed yet.

Exercise 4. Final Project Checkpoint.

Objective: create a file that has a list of forms.

1. Create a file called `forms.php`. In this file, create links to these other files:
    a. `addproverb.php` (from exercise 3, this week)
    b. `alphabetize.php` (from last week)
    c. `signguestbook.php` (from exercise 1, this week)
2. This new file, called `forms.php`, should use the template from your project. That is, it should have only the web form, and all the other features are from include files:
    a. header/banner
    b. text navigation
    c. button navigation
    d. footer
3. Copy the guestbook form and handlers from the Chapter10 folder into the Projects folder.
    a. Add the template to the form and the handlers.
4. Copy the addproverb form and handler from the Chapter10 folder into the Projects folder.
    a. Add the template to the form and the handlers.
5. Copy the alphabetize form and handler from the Chapter08 folder into the Projects folder.
    a. Add the template to the form and the handlers.

Exercise 5.

Link all these files to your home page so I can look at them quickly.

Looking forward

- The final project should have some **loops.** We didn't make any super interesting loops in class.
- The final project should have some string functions linked to that button. In the past, we used the **levenshtein** and **similar_text** functions. We may use something else here.
- We may add the **city mileage** project to the forms page.
- We probably will ***not*** link the **midterm** or final exam to the project, but you may do so if you'd like to make a nice complete portfolio of your work.
- The final project is going to display a **random proverb** and a **hit counter** in the footer. We haven't made the hit counter yet. That will be part of the Chapter 12 homework **(cookies)**.