2170 ASSIGNMENT 3

***Due date.*** **Tuesday November 10, 2020 (11:00pm)**

**Purpose:** This is the second in a set of assignments that will lead you to develop a lightweight Content Management System (CMS) that is similar to Instagram – in this case called ***Picturegram.*** *Picturegram* will capture moments using mainly photos with small posts, that other users can comment. As well, Picturegram will have the following functionality/pages: search posts (in page and within all posts), and other administrative duties (e.g., add a new post, sign in).

The purpose of this assignment is to build on the codebase you created in Assignments 1 and 2. **All assignments are to be your own work.**

**In this assignment, you will do the following:**

1. Create a relational database and tables, using MySQL (you can use PHPMYAdmin), to provide functionality for specified aspects of your CMS ("Picturgram").
2. Add additional functionality to the homepage (index.php), posts (post.php), the about page (about.php).
   1. Using the database to save, retrieve and update content to display on the listed pages
   2. Ensuring links are active and functional
3. Implement the addPost.php page – users will be able to add text for a post and an image file in a form. The information will be used to create a new Post stored in the database.
4. Ensure that you add functionality noted as missing in Assignment 2 (the markers will list items that were missing in the last assignment and these need to be addressed in this assignment – in the assignment documentation).

Note that if you decide to make small changes to your templates you designed in Assignment 1 or 2, that is fine but these changes must be described in your documentation.

**Image and content considerations:**

I have supplied the text files and images that you will use for this assignment (description of the files is at the end of the assignment).

**Assignment Information:**

***Grades.*** This assignment will be graded out of 60 marks and then scaled to 10 points.

***Submission.*** All deliverables must be submitted on Brightspace: <https://dal.brightspace.com>

***Late submission policy.*** **There is a two-day late policy** (one day late: 10% deduction, second day late: 15% deduction. Max of 25% deduction). No assignments will be accepted after the 2-day late deadline.

***Academic Integrity.*** Dalhousie academic integrity policy applies to all submissions in this course. You are expected to submit your own work. Please refer to and understand the academic integrity policy, available at: <https://www.dal.ca/academicintegrity>

***Content for the website.*** Use the supplied content which is mostly based on the Lorem Ipsum generator to generate paragraphs of dummy text: <http://www.lipsum.com> and images that are mine (I give you permission to use for the assignment/s).

***Deliverables.*** Follow instructions of this document to submit all deliverables of this assignment.

**General Instructions**

1. Create a folder named A3.
2. Copy all your scripts, images, folders and other files from your A2 submission and paste them into A3. Make sure updated and additional files are also included (including extra files you downloaded for the templates from Bootstrap).
3. For this assignment, you will need to use MySQL.

Note: if you are using a local stack that come with MariaDB or another DBMS which is not MySQL it is your responsibility to ensure/verify that your PHP-MySQL scripts work with MySQL as well. We will be using MySQL for marking. *[MariaDB is referred to as a “drop-in” replacement to MySQL, and therefore, it should work as a direct replacement. It should work with the same queries that you had used for MySQL – but still make sure it works]*

1. Start working on this assignment by creating the database and tables. *NOTE: We will not be implementing the "Login" for this assignment, but it should be included.*

**Database Use**

1. You will be connecting the front-end templates that you created in A2/A1 to a database. You will apply your knowledge of creating databases and tables, setting primary keys for a table, etc.  
   **Note:** *you do not need to submit a “database dump” along with your submission. However, you must create* ***the table structure exactly*** *as specified in this section, because we will be using a database with the specified structure to verify your code. You can use the data from the files to fill the database.*
2. Create a database named **Picturegram**. In this database, create three tables as described next.  
   **Note:** *In your DB setup, you must set the username to be* ***root*** *and password to be* ***root****.*

**Table1: Users**

This table will hold the name of all the users of the system (this would include "you" as the main user of your Picturegram and all other users of the application). About is a short description of the user. **There will only be one user (owner) as in the user table for now, since there is no login to determine who is the user of the account.**

There are 4 attributes: UserID, Name, About, AboutImage (the file name of the image for the About page). Note, store the image file name and then upload the image file (see Figure 1 for the setup of this Table).

|  |  |
| --- | --- |
| 1. UserID    * Type: integer (11)    * Set up as the primary key    * Must be set to auto increment starting at 1    * Must not be null 2. Name    * Must be VARCHAR (100)    * Must not be null. | 1. About    * Must be text    * Must not be null 2. About Image    * Must be VARCHAR(50)    * Must not be null |

Figure 1a) Set up of the Users Table (PHPMyAdmim)

|  |
| --- |
|  |

Figure 1b) Sample input in the Users Table (note, only UserID 1 will be used in this assignment)

|  |
| --- |
|  |

**Table 2: Posts**

This table will hold information related to the posts that users make, including a PostID, the UserID of who made the post, the PostImage file name, the date of the post (using a timestamp), the text of the post (Post). (see Figure 2 for an example of the setup of this Table and with a few sample entries).

There are 6 attributes: PostID, UserID, Title, ImageFName, Post, and Date

|  |  |
| --- | --- |
| 1. PostID    * Type: integer (11)    * Set up as the primary key    * Must be set to auto increment starting at 1    * Must not be null 2. UserID    * Type: integer (11)    * Set up as foreign key - links to UserID in Users Table. index the attribute and then you can add the foreign key (if using PHPMyAdmin)    * Must not be null 3. PostImage    * Must be VARCHAR(50)    * Must not be null | 1. Post    * Must be Type: Text    * Must not be null 2. Date    * Type: timestamp    * Must not be null    * Default: current\_Timestamp    * Should update when create a post automatically (this is in PHPMyAdmin under extra - set to updateCurrent\_timestamp    * Note, this sets it UTF – so translate to the local timezone[date\_default\_timezone\_set('America/Halifax');] in your PHP code when displaying on the appropriate page. |

Figure 2a) Set up of the Posts Table (PHPMyAdmim)

|  |
| --- |
| Table  Description automatically generated |

Figure 2b) Sample input in the Posts Table (Use the text from files from A2 to fill in Posts (as test data).

|  |
| --- |
| Graphical user interface, text, application  Description automatically generated |

**Table 3: Comments**

The Comments table records the Comments made on each Post (as well as the User who made it although for this assignment this will always be set to UserID 1 – since we are not dealing with the multiple users yet). There are five attributes: CommentID, UserID, PostD, Comment, and Date (see Figure 3a and Figure 3b for an example of the setup of this Table and with sample entries. Again, you can use the files from A2 to fill this table).

|  |  |
| --- | --- |
| 1. CommentID    * Type: integer (11)    * Set up as the primary key    * Must be set to auto increment starting at 1    * Must not be null 2. UserID    * Type: integer (11)    * Set up as a foreign key (that links to UserID in Users Table) – you will need to index the attribute and then you can add the foreign key (if using PHPMyAdmin)    * Must not be null 3. PostID    * Type: integer (11)    * Set up as a foreign key (link to PostID in Posts Table) – index the attribute and then you can add the foreign key (if using PHPMyAdmin)    * Must not be null | 1. Comment    * Must be Type: Text    * Must not be null 2. Date    * Type: timestamp    * Must not be null    * Default: current\_Timestamp    * Should update when create a post automatically (in PHPMyAdmin under extra it should be set to updateCurrent\_timestamp    * Note, this sets it UTF – so you will need to translate to the local timezone [date\_default\_timezone\_set('America/Halifax');] in your PHP code when displaying on the appropriate page. |

Figure 3a) Set up of the Comments Table (PHPMyAdmim)

|  |
| --- |
| Table  Description automatically generated |

Figure 3b) Sample input in the Comments Table (Use the files from A2 to fill the comments for each post. Note, you will need to open each comment file and manually (add copy/paste) each comment in the file to the table).

|  |
| --- |
| Table  Description automatically generated |

DB3. Familiarize yourself with the **phpMyAdmin** interface available in MAMP or other local server environments

that you may be using for your work. This will be useful when you want to create and manage your database tables. You should be able to access this interface on your web browser by visiting the following link: **http://localhost/phpMyAdmin**

**Functionality/Additional Functionality for Assignment 3**

1. Wherever you got data/saved data from files to populate the templates (Home, Post, and About) in Assignment 2, you will retrieve/save data from the Picturegram database. We still won't be implementing Login Page for this assignment (but the login.php must still be included).
2. You will ensure when a new comment is made for a Post it is updated and visible on the Post page
3. All posts and comments should be sorted by showing newest (most recent) first.
4. Update the addPost page so that a user can add a new post.

**Updates to the Homepage Template**

**For this assignment, there will be one row in the User Table (we will add more users and login functionality in A4]) While all information should be from the query you can hard code the UserID 1 anytime you need to reference the user.**

You will use the supplied text and images files that you will add to your database (Picturegram to fill the content for the for the home page. The posts will be displayed by date (the most recent will be first). The format will always be Image, post text, authors name and the date and time that the post was made (the most recent will be displayed first).

When a user clicks the author's name, the user should be taken to the About page which will populate the content on the page using the database.

When a user clicks the text or image of a post, the user should be taken to the Post page (post.php) that will dynamically display the selected post/image (as well as the comments that other users make on the post).

Use Figure 4 to help you implement this page.

|  |  |
| --- | --- |
| A picture containing photo, different, view, television  Description automatically generated | NOTE: the *name* (e.g.,"Lorem Nullan") on the navigation bar, will be based on UserID#1. The image and title/subtitle on this page can be hardcoded.  Use the tables in the database Picituregram, to populate the page.  Use the Post Table to populate the homepage will all of UserID #1's posts.  All the posts (images and text) are listed in order (most recent to oldest). Note, the first post is dated August 22, 2020 and last post is dated June 3, 2020).  The author's name should be the one listed in the User Table that corresponds to UserID #1.  When a user selects an image or text of a post, it should take them to the post.php page which will be updated based on selected post.  HINT: Use a querystring with GET method and pass related information (e.g., UserID and PostID) in the URL that the post.php page can then use to create its page). |
| Figure 4. Homepage Template | |

**Updates to the Post Template**

The Post page contains a single post with an image and text of the original post and a list of all the comments made on the post. It should also have a form to add a new comment. All the data from this page will come from the database (you will need to access the Users, Posts and Comments table). You can use the information from the querystring to query the database.

When a user adds a comment, it should write the comment to the comments table in the database with the current timestamp and update the display of the comments with the new comment added. Note, the author of the comment should also be added after the date (see figure 5d).

Use Figure 5 to help you further implement this page.

|  |  |
| --- | --- |
| Graphical user interface, website  Description automatically generated | Graphical user interface, website  Description automatically generated |
| Figure 5a) Step 1. User selects a post from index.php (homepage) | Figure 5b) Step 2. post.php page – loaded with appropriated info and comments if there are any |
| Graphical user interface, website  Description automatically generated | Graphical user interface, website  Description automatically generated |
| Fig. 5c) Step 3. User makes a new comment and presses submit (comment is saved to comment.txt file) | Figure 5d) Step 4. post.php updates with new comment as first. Note, the author of the comment is added after the date the comment was made. Comments should be in order of newest to oldest. |

**Updates to the About Template**

This page contains information about the author of the blog. It has an image and a description of the author that will downloaded from the database. The user should be taken to this page anytime they click an author's name in a post or when they click the About on the navigation bar.

Use Figure 6 to help you further implement this page.

|  |
| --- |
| Text  Description automatically generated |
| Figure 6. About template (with data added) |

**Updates to the addPost Template**

Adapt the form that you created for addPost.php in Assignment 1. The form will allow users to add new posts (Figure 7). For a new post, the user will the post text and the image file name (assume that the server already has the image and store it in you’re A3/files directory).

When you insert the new post, remember that you don't need to include the PostID since it is an auto-incremented primary key or the date field since this automatically added. Also, assume that the UserID is 1.

Once the post is created and updated in the database, the next time the user visits "Home" the new post will show up first on their homepage (Figure 7).

|  |
| --- |
| Graphical user interface, website  Description automatically generated  When creating the post, remember the only user we have is UserID 1 (you can hard this into the SQL query or add as a hidden value in the form).  Edit the form so that the user can add a post and the name of an image to use. |
| Figure 7 a) The new Add Post template/page (addPost.php) |

|  |  |
| --- | --- |
| Graphical user interface, website  Description automatically generated | A picture containing table  Description automatically generated |
| Index.php after adding a new post (newest post at the top – added the beach post) | The database after adding a new post. |
| Figure 7b). The new addPost page (and how the new post is added to the home page) and database | |

**File Formats for Assignment 3**

Even though you will be using a database now to store data, there will still be some files needed (e.g., image files and *include* files that will contain the information needed for the database).

**Image files**

Use the same images that I included for Assignment 2 .

**Include files**

You should use *include* files in your code. For Assignment 3 you should have a serverlogin file (serverlogin.php) that contains the information you need to login into the database. Remember for your assignment the username and password need to be set to root. Your serverlogin file should look like this:

<?php // server login.php

$db\_hostname = 'localhost';

$db\_username = 'root'; //your username

$db\_password = 'root'; //your password

$db\_database = 'PictureGram'; //database name

?>

In addition, you should also add your file called functions.php that will hold functions that you call in your code that can 'included' in all your files.

**Submissions (DELIVERABLES) (will be graded for a total of 60 marks)**

1. Folder structure (5 marks)
   * Folder structure must be (note, you may have some additional files/folders from using Bootstraps templates that are not listed below. That is fine.):

**A3**

|---- index.php

|---- post.php

|---- about.php

|---- addPost.php

|---- login.php

|---- functions.php

|---- serverlogin.php

|---- **css**

| |---- CSS files as per framework[[1]](#footnote-1) (template)

| |---- CSS files that you might have created

|---- **fonts**

| |---- Font files as per framework

| |---- Other font files that you might have used

|---- **js**

| |---- JavaScript files as per framework1

| |---- Other JavaScript files that you might have created/used

|---- **misc**

| |---- <placeholder for future functionality – empty folder>

|---- **files**

| |---- all the files (text, csv, images) that you used

A2-----| -- include any files in this directory from A2 that weren't included in

the above directory.

1. Navigation and Overall look/functionality of site (5 marks)
   * Look and feel is consistent between all templates
   * Demonstrates that can load, update, data and display data in *order by* date
   * Navigation works as expected (i.e. proper pages load with up to date content).
2. Database and use of database information (10 marks)
   * Table creation and use (we need to be able to use a local database to grade so your database **must** match names and type exactly). Include the database template in your zipped file.
   * Wherever you used files to save/retrieve data in A2, this must be replaced with the database
3. Homepage template (10 marks)
   * Design, Functionality, Good programming style and commenting
   * Content is from the database which is queried and displayed properly. Your code should be able to handle any number of entries/rows of posts.
   * Dates and times are properly displayed
   * Links to post.php page
4. Post template (10 marks)
   * Design, Functionality, Good programming style and commenting
   * Content is from the database which is queried and displayed properly.
   * Data from index.php page is transferred (e.g., querystring) to post.php page
   * Comments are loaded properly in the correct order from the database
   * Dates and times are properly displayed
   * New comments are saved to database and after new comment is made, page is *automatically* refreshed with newest comment first
5. About (5 marks)
   * Design, Functionality, Good programming style and commenting
   * Content is from the database which is queried and displayed properly
6. AddPost (10 marks)
   * Design, Functionality, Good programming style and commenting
   * Data entered in the form is saved properly to the database, including the current date
   * The new post shows up when go back home in proper order
7. Documentation (framework.txt) (5 marks) – 1 to 2 pages max
   * Deal with any missing parts/parts for improvement listed from A1/A2 – list and describe how you fixed them.
   * Any updated elements from A2 (if you decided to change anything from A2) explain what you changed.
   * A description of the database
   * A description of how you implement the search
   * Any references

1. [↑](#footnote-ref-1)