IMY 220
Assignment 2:
PHP File Upload

Due: Wednesday 17 September at 13:00.

The submission instructions are available on ClickUP. Any deviation from these instructions will cause a 10% deduction from your mark.

Instructions

- Use the provided registration and login functionality and add functionality for a system that displays upcoming events. The focus of this assignment will be PHP and MySQL but you will be making use of some bootstrap.
- Use MySQL to add images to a gallery and display them when the user is logged in.

Part 1: SQL Database

You will be required to create a database with three tables in it using PHPmyAdmin.
- The database must be called *dbuser*
- The tables must be called *tbusers*, *tbevents* and *tbgallery*

tbusers must contain the following fields:
- *user_id* – integer, primary key, auto increment
- *name* – varchar (100)
- *surname* – varchar (100)
- *email* – varchar (100)
- *password* – varchar (100)
- *birthday* – Date

tbevents must contain the following fields:
- *event_id*: integer, primary key, auto increment
- *user_id*: integer
- *name*: varchar (100)
- *description*: varchar (100)
- *date*: Date

tbgallery must contain the following fields:
- *image_id*: integer, primary key, auto increment
- *event_id*: integer
- *image_name*: varchar(50)

Part 2: login.php – image upload

Download the files (*index.html*, *style.css, login.php, and register.php*) from ClickUP. These files contain the functionality for creating and logging in as a new user using the database structure. It also contains an event creation form on the login page and additional CSS for displaying images correctly. You will only need to edit ***login.php*** to complete this assignment.

Inside *login.php*, when a user selects an image with the file input, and types in all the text inputs, both the image, name, description and event date need to be uploaded and be displayed inside the user's upcoming events gallery. This must work as follows:

- When a user creates a new event (by selecting a valid file with the file input, typing in a description, name, date, and submitting the form), the form should submit to itself (to *login.php*) and should get and upload the file, name, description, and date that has been selected using the file input and text input. Use the POST method for this.
- To do this, you will need to send through the user's email and password again. Use **hidden inputs** to do this.
- Note that an event should **not** be created/added to the database if the user has not submitted the **name, description, date,** and **image file.** Thus, you should check that user has submitted all the associated information and the image file.
- You should then check if the uploaded file is a jpg/jpeg image and is smaller than 1MB (you may assume that it is otherwise safe to upload). Only if the selected file satisfies these requirements should the file be uploaded to the directory called "gallery" (which you will also need to create yourself).
- A new row should also be added to the *tbevents* table (when the user uploads) with the *user_id* of the current user, name of the event, description, and date. A new row should be added to *tbgallery* table with the new event's *event_id* and the filename that the file was uploaded with. The filename, name and description will be used to fetch the images and descriptions again for the event gallery.
- The page must be error-free at all times.
- The result of this should be that a user is able to log in using *index.html* and use the file and text inputs on *login.php* to create events. A user should be able to create as many events as they want (one at a time).
- You do not have to cater for users uploading files with the same name. If they do, the new file will simply overwrite the old one.

Part 3: login.php – Displaying the events

Display all the events a user has created inside a gallery in **reverse chronological order**. This has to work as follows:

- Beneath the table with the user's details and the form for creating an event, add a heading and a Bootstrap row. Give the Bootstrap row an additional class of "*eventsGallery*" In order to display the images neatly (and have a constant size). Tip: if you get stuck as to where this should be placed, look in *styles.css* to see how it was structured.
- Fetch all the events that the user has uploaded, in other words, all the events (images, descriptions, name and date) that have a *user_id* value that is the same as the current user's id. Fetch the images that correspond to the respective *event_id* value of each event belonging to the current user.

- Display each of these inside Bootstrap cards. More information on bootstrap cards can be found on their website.
- The event name, description and date should all be displayed in the bootstrap card, and they should resemble the screenshot below.

Note that *style.css* includes styling rules for displaying images inside the bootstrap cards neatly, so it is important that you add the correct class to the row. The result should be that if a user has created an event, their login page should display these in a gallery (including directly after they created the events). For example, if a user has created 5 events, their page might look like this:

Bonus

Allow users to delete an event. You can do this by adding a button to the event that says "delete" and if pressed, it deletes the event from all the appropriate tables. You will need to create an additional form to do this. Make sure it deletes everywhere, otherwise you won't receive the marks.

Additional Information

- Refer to the slides and online resources for help

Submit only the following file(s) according to the submission instructions.

- *login.php*