DEPARTMENT OF COMPUTER STUDIES

**(Applications Development and Emerging Technologies)**

**PRE-SUMMATIVE ASSESSMENT**

**7**

**USER PRIVILEGES, MANAGEMENT AND FILE UPLOAD**

|  |  |
| --- | --- |
| **Student Name / Group Name:** |  |
| **Members (if Group):** | |  |  | | --- | --- | | **Name** | **Role** | |  |  | |  |  | |  |  | |
| **Section:** |  |
| **Professor:** |  |

1. **PROGRAM OUTCOME/S (PO) ADDRESSED BY THE LABORATORY EXERCISE**

* Design, implement and evaluate computer-based systems or applications to meet desired needs and requirements.

1. **COURSE LEARNING OUTCOME/S (CLO) ADDRESSED BY THE LABORATORY EXERCISE**

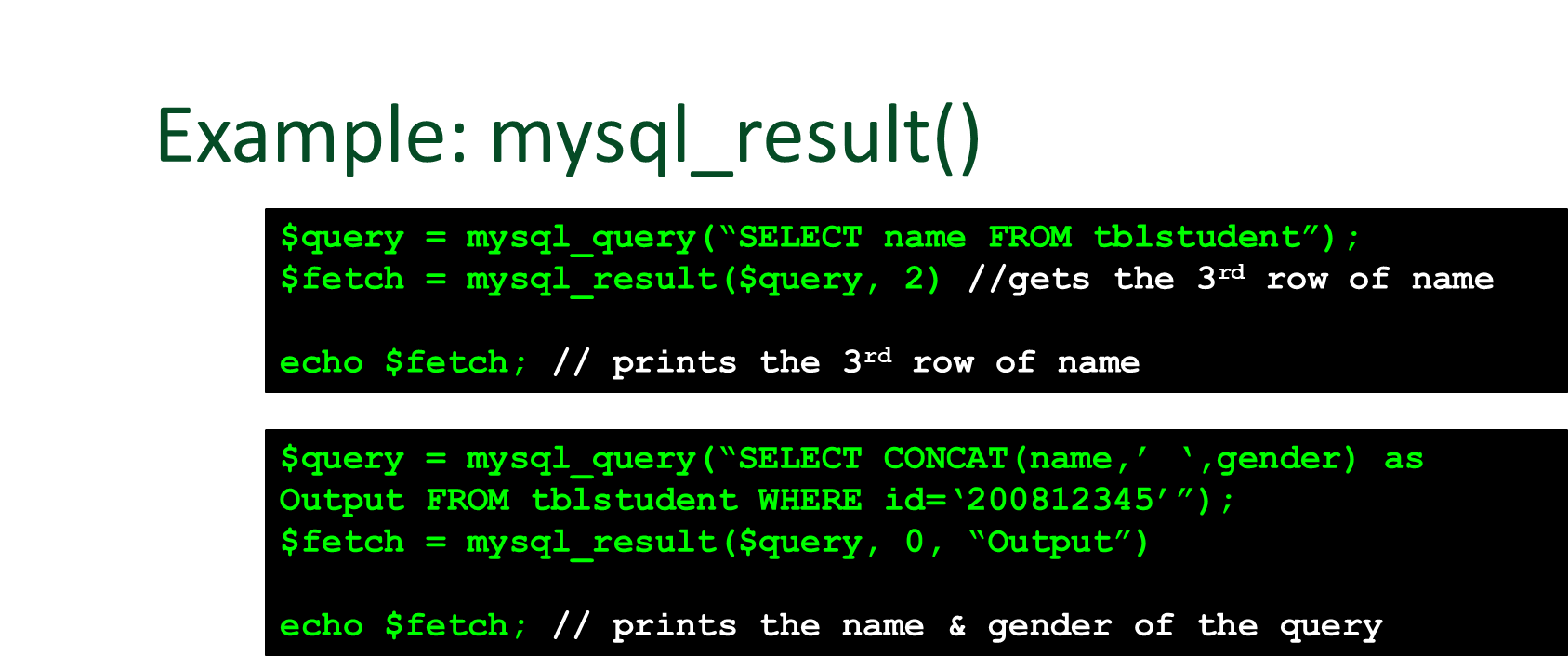
* Understand and apply best practices and standards in the development of website.

1. **INTENDED LEARNING OUTCOME/S (ILO) OF THE LABORATORY EXERCISE**

At the end of this exercise, students must be able to:

* To provide a good background of Relational Database using MySQL.
* To know the importance of Database in Web Application using MySQL.
* To Identify the importance of Database Structure in constructing tables.
* To be familiar with the syntax in managing users and database.
* To define a good structure of tables in a given database for data storage.
* To be familiar in the common syntax of creating database and tables and the correct data type to be used for each field.

1. **BACKGROUND INFORMATION**

****

PHP FILE UPLOAD

With PHP, it is easy to upload files to the server.

However, with ease comes danger, so always be careful when allowing file uploads!

Configure The "php.ini" File

First, ensure that PHP is configured to allow file uploads.

In your "php.ini" file, search for the file\_uploads directive, and set it to On:

file\_uploads = On

Create The HTML Form

Next, create an HTML form that allow users to choose the image file they want to upload:

<!DOCTYPE html>  
<html>  
<body>  
  
<form action="upload.php" method="post" enctype="multipart/form-data">  
  Select image to upload:  
  <input type="file" name="fileToUpload" id="fileToUpload">  
  <input type="submit" value="Upload Image" name="submit">  
</form>  
  
</body>  
</html>

Some rules to follow for the HTML form above:

* Make sure that the form uses method="post"
* The form also needs the following attribute: enctype="multipart/form-data". It specifies which content-type to use when submitting the form

Without the requirements above, the file upload will not work.

Other things to notice:

* The type="file" attribute of the <input> tag shows the input field as a file-select control, with a "Browse" button next to the input control

The form above sends data to a file called "upload.php", which we will create next.

Create The Upload File PHP Script

The "upload.php" file contains the code for uploading a file:

<?php  
$target\_dir = "uploads/";  
$target\_file = $target\_dir . basename($\_FILES["fileToUpload"]["name"]);  
$uploadOk = 1;  
$imageFileType = strtolower(pathinfo($target\_file,PATHINFO\_EXTENSION));  
// Check if image file is a actual image or fake image  
if(isset($\_POST["submit"])) {  
  $check = getimagesize($\_FILES["fileToUpload"]["tmp\_name"]);  
  if($check !== false) {  
    echo "File is an image - " . $check["mime"] . ".";  
    $uploadOk = 1;  
  } else {  
    echo "File is not an image.";  
    $uploadOk = 0;  
  }  
}  
?>

PHP script explained:

* $target\_dir = "uploads/" - specifies the directory where the file is going to be placed
* $target\_file specifies the path of the file to be uploaded
* $uploadOk=1 is not used yet (will be used later)
* $imageFileType holds the file extension of the file (in lower case)
* Next, check if the image file is an actual image or a fake image

**Note:** You will need to create a new directory called "uploads" in the directory where "upload.php" file resides. The uploaded files will be saved there.

Check if File Already Exists

Now we can add some restrictions.

First, we will check if the file already exists in the "uploads" folder. If it does, an error message is displayed, and $uploadOk is set to 0:

// Check if file already exists  
if (file\_exists($target\_file)) {  
  echo "Sorry, file already exists.";  
  $uploadOk = 0;  
}

Limit File Size

The file input field in our HTML form above is named "fileToUpload".

Now, we want to check the size of the file. If the file is larger than 500KB, an error message is displayed, and $uploadOk is set to 0:

// Check file size  
if ($\_FILES["fileToUpload"]["size"] > 500000) {  
  echo "Sorry, your file is too large.";  
  $uploadOk = 0;  
}

Limit File Type

The code below only allows users to upload JPG, JPEG, PNG, and GIF files. All other file types gives an error message before setting $uploadOk to 0:

// Allow certain file formats  
if($imageFileType != "jpg" && $imageFileType != "png" && $imageFileType != "jpeg"  
&& $imageFileType != "gif" ) {  
  echo "Sorry, only JPG, JPEG, PNG & GIF files are allowed.";  
  $uploadOk = 0;  
}

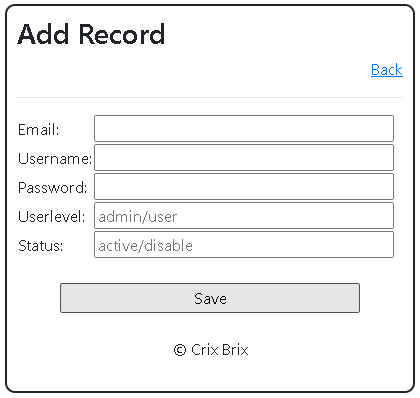
Complete Upload File PHP Script

The complete "upload.php" file now looks like this:

<?php  
$target\_dir = "uploads/";  
$target\_file = $target\_dir . basename($\_FILES["fileToUpload"]["name"]);  
$uploadOk = 1;  
$imageFileType = strtolower(pathinfo($target\_file,PATHINFO\_EXTENSION));  
  
// Check if image file is a actual image or fake image  
if(isset($\_POST["submit"])) {  
  $check = getimagesize($\_FILES["fileToUpload"]["tmp\_name"]);  
  if($check !== false) {  
    echo "File is an image - " . $check["mime"] . ".";  
    $uploadOk = 1;  
  } else {  
    echo "File is not an image.";  
    $uploadOk = 0;  
  }  
}  
  
// Check if file already exists  
if (file\_exists($target\_file)) {  
  echo "Sorry, file already exists.";  
  $uploadOk = 0;  
}  
// Check file size  
if ($\_FILES["fileToUpload"]["size"] > 500000) {  
  echo "Sorry, your file is too large.";  
  $uploadOk = 0;  
}  
  
// Allow certain file formats  
if($imageFileType != "jpg" && $imageFileType != "png" && $imageFileType != "jpeg"  
&& $imageFileType != "gif" ) {  
  echo "Sorry, only JPG, JPEG, PNG & GIF files are allowed.";  
  $uploadOk = 0;  
}  
// Check if $uploadOk is set to 0 by an error  
if ($uploadOk == 0) {  
  echo "Sorry, your file was not uploaded.";  
// if everything is ok, try to upload file  
} else {  
  if (move\_uploaded\_file($\_FILES["fileToUpload"]["tmp\_name"], $target\_file)) {  
    echo "The file ". basename( $\_FILES["fileToUpload"]["name"]). " has been uploaded.";  
  } else {  
    echo "Sorry, there was an error uploading your file.";  
  }  
}  
?>

1. **GRADING SYSTEM / RUBRIC (please see separate sheet)**
2. **LABORATORY ACTIVITY**
3. Create database with 10 different records use the following fields [id(auto increment),email, username, password, userlevel(admin/user), status, image]. Use the webpage example below to add the records.

Example (admin\_add.php)



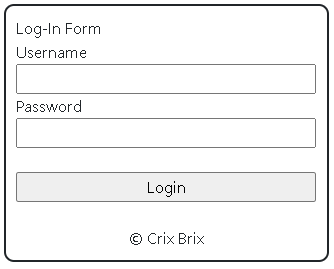
Note: Back link will redirect to admin.php

1. Create a Login form using the information from the created database.

Note: depends on the user level where the webpage will redirect

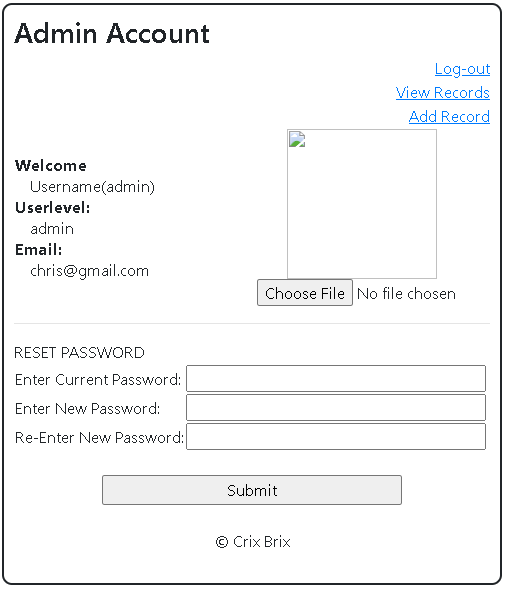
admin -> admin.php

user -> user.php



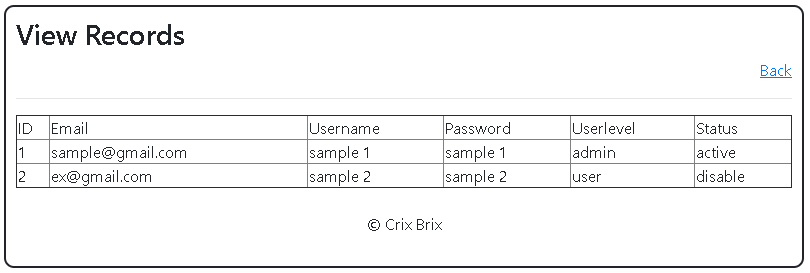
1. On the admin.php create a link where can show all the records of the database. Please see the example below

Example (admin.php)



Note: click the View Records link to proceed viewusers.php

Example (viewusers.php)

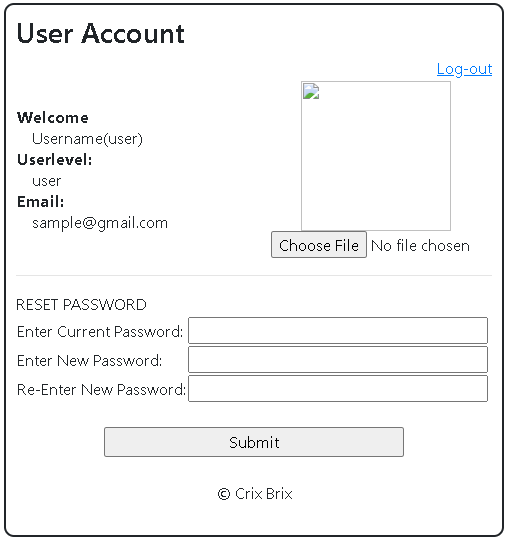


Note: Back link will go back to admin.php

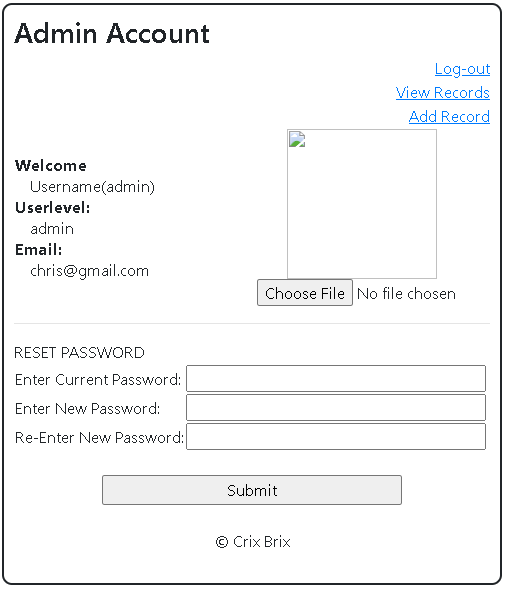
1. Create file function to upload image for the admin/user

See the example below:

Example (user.php)



Example(admin.php)



1. Screenshot your database from XAMPP.

***Snip and paste your source codes here. Snip it directly from the IDE so that colors of the codes are preserved for readability. Include additional pages if necessary.***

**VII. QUESTION AND ANSWER**

1. What are the steps to connect a database?
2. Give the MySQL command that you used in your webpage and describe each
3. What is the importance of file functions?
4. What are the different file functions?

**VIII. REFERENCES**

1. <https://www.w3schools.com/css/>
2. <https://www.w3schools.com/html/>
3. <https://www.w3schools.com/php/php_variables.asp>
4. <https://www.w3schools.com/php/php_mysql_intro.asp>
5. <https://www.w3schools.com/php/php_mysql_connect.asp>
6. <https://www.w3schools.com/php/php_mysql_create.asp>
7. <https://www.w3schools.com/php/php_mysql_create_table.asp>
8. <https://www.w3schools.com/php/php_mysql_insert.asp>
9. <https://www.w3schools.com/php/php_mysql_insert_multiple.asp>
10. <https://www.w3schools.com/php/php_mysql_select.asp>
11. <https://www.w3schools.com/php/php_mysql_select_where.asp>
12. <https://www.w3schools.com/php/php_mysql_select_orderby.asp>
13. <https://www.w3schools.com/php/php_mysql_delete.asp>
14. <https://www.w3schools.com/php/php_mysql_update.asp>
15. <https://skillforge.com/how-to-create-a-database-using-phpmyadmin-xampp/>
16. <https://www.w3schools.com/php/php_file_upload.asp>

**Note: The following rubrics/metrics will be used to grade students’ output.**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Program (100 pts.)** | **(Excellent)** | **(Good)** | **(Fair)** | **(Poor)** |
| **Program execution (20pts)** | Program executes correctly with no syntax or runtime errors **(18-20pts)** | Program executes with less than 3 errors **(15-17pts)** | Program executes with more than 3 errors **(12-14pts)** | Program does not execute **(10-11pts)** |
| **Correct output**  **(20pts)** | Program displays correct output with no errors **(18-20pts)** | Output has minor errors **(15-17pts)** | Output has multiple errors **(12-14pts)** | Output is incorrect **(10-11pts)** |
| **Design of output**  **(10pts)** | Program displays more than expected **(10pts)** | Program displays minimally expected output **(8-9pts)** | Program does not display the required output (**6-7pts)** | Output is poorly designed **(5pts)** |
| **Design of logic**  **(20pts)** | Program is logically well designed **(18-20pts)** | Program has slight logic errors that do no significantly affect the results **(15-17pts)** | Program has significant logic errors **(3-5pts)** | Program is incorrect **(10-11pts)** |
| **Standards**  **(20pts)** | Program code is stylistically well designed **(18-20pts)** | Few inappropriate design choices (i.e. poor variable names, improper indentation) **(15-17pts)** | Several inappropriate design choices (i.e. poor variable names, improper indentation) **(12-14pts)** | Program is poorly written **(10-11pts)** |
| **Delivery**  **(10pts)** | The program was delivered on time. **(10pts)** | The program was delivered a day after the deadline. **(8-9pts)** | The program was delivered two days after the deadline. **(6-7pts)** | The program was delivered more than two days after the deadline. **(5pts)** |