**Database II GROUP 1**

Head: Apol

Members:

Osmar

Brian

Apol worked on XAAMP Question 4

& MongoDB Question 2.

Osmar worked on XAAMP Question 1.

Brian worked on MongoDB Question 2.

***Answers down below***

XAAMP Question 1: Osmar’s Answer

**1.** A dynamic application needs a Web server, MySQL, and PHP. For Web server Apache is preferred since it meets all the needs of the Next Generation of Genealogy Sitebuilding(TNG). TNG stores your data in databases located on the server which is then rendered into “dynamic” Web pages when prompted by a site visitor. As a visitor’s requests change, TNG locates the appropriate data from the database and renders it in HTML format for viewing in their browser.

A user login process is needed to run an application. For this purpose, sessions will be used.

A session is just the time duration of a user for the activities he performs on the web. For e.g.: when a user opens Facebook and login to view his/her Facebook profile and after completing the activities he/she logout. The time between login and log out is called a session.

It is needed to create an application for login logout operations using sessions.

This application will have two parts. The part1 includes the registration of the users (id, uname, upassw). The part2 contains the login, logout operations including the use of sessions.

The login operation will ask from the user to insert the name (uname) and the password (upassw) with the “reset” choice. If the user has already registered, then the application will answer: Welcome (the name of user) (i.e. Welcome Paul) otherwise a message “Invalid username or password”.

The logout operation happens after the end of the login and the user will come back to the initial status for waiting to insert new input (uname, upassw).

Register.php

<!DOCTYPE html>

<html>

    <head>

        <title>User Registration</title>

    </head>

    <body>

        <h2>User Registration</h2>

        <form action="register.php" method="post" autocomplete = "off">

            <label for="uname">Username:</label><br>

            <input type="text" id="uname" name="uname"><br><br>

            <label for="upassw">Password:</label><br>

            <input type="password" id="upassw" name="upassw"><br><br>

            <input type="submit" value="Submit">

        </form>

    </body>

    <?php

    $servername = "localhost:3307";

    $username = "root";

    $password = "";

    $database = "page";

    $conn = new *mysqli*($servername, $username, $password, $database);

    if ($conn->connect\_error) {

        die("Connection failed: " . $conn->connect\_error);

    }

    if ($\_SERVER["REQUEST\_METHOD"] == "POST") {

        $uname = ($\_POST['uname']);

        $upassw = ($\_POST['upassw']);

        $duplicate = mysqli\_query($conn, "SELECT \* FROM users WHERE uname = '$uname'");

        if (mysqli\_num\_rows($duplicate) > 0) {

            echo "This username has already been taken";

        } else {

            $sql = "INSERT INTO users (uname, upassw) VALUES ('$uname', '$upassw')";

            if ($conn->query($sql) === TRUE) {

                echo "New record created successfully";

            } else {

                echo "Error: " . $sql . "<br>" . $conn->error;

            }

        }

    }

    // Close connection

    $conn->close();

    ?>

    <p>Already have an account?</p>

    <form action="log in-out.php" method="get">

                <input type="submit" value="Log in">

    </form>

</html>

Log.php

<!DOCTYPE html>

<html>

<head>

    <title>User Login</title>

</head>

<body>

    <h2>User Login</h2>

    <form method="post" action="" autocomplete="off">

        Username: <br><input type="text" name="uname"><br><br>

        Password: <br><input type="password" name="upassw"><br>

        <input type="submit" name="login" value="Login">

        <input type="submit" name="reset" value="Reset">

    </form>

    <p>Want to make an account?</p>

    <form action="register.php" method="get">

        <input type="submit" value="Register">

    </form>

</body>

</html>

<?php

session\_start();

$servername = "localhost:3307";

$username = "root";

$password = "";

$dbname = "page";

$conn = mysqli\_connect($servername, $username, $password, $dbname);

// Check connection

if (!$conn) {

    die("Connection failed: " . mysqli\_connect\_error());

}

// Check if form is submitted for login

if ($\_SERVER["REQUEST\_METHOD"] == "POST" && isset($\_POST['login'])) {

    $uname = ($\_POST['uname']);

    $upassw = ($\_POST['upassw']);

    // Check if user exists in database

    $sql = "SELECT \* FROM users WHERE uname = '$uname' AND upassw = '$upassw'";

    $result = mysqli\_query($conn, $sql);

    if (mysqli\_num\_rows($result) == 1) {

        $\_SESSION['uname'] = $uname;

        header("Location: home.php"); // Redirect to home page

        exit();

    } else {

        echo "Invalid username or password.";

    }

}

mysqli\_close($conn);

?>

Home.php

<?php

session\_start();

// Check if user is logged in

if (!isset($\_SESSION['uname'])) {

    header("Location: index.php");

    exit();

}

?>

<!DOCTYPE html>

<html>

<head>

    <title>Welcome</title>

</head>

<body>

    <h2>Welcome, <?php echo $\_SESSION['uname']; ?>!</h2>

    <form action="logout.php" method="post">

        <input type="submit" name="logout" value="Logout">

    </form>

</body>

</html>

A screenshot of a login form

Description automatically generated

A screenshot of a computer program

Description automatically generated

Username cannot be taken twice.

A screenshot of a registration form

Description automatically generated

Logged in with johnt username and password

A screenshot of a login form

Description automatically generated

A screenshot of a login screen

Description automatically generated

Did not let a person with no account log in

A white background with black text

Description automatically generated

Welcome screen when a user logs in.

XAAMP Question 4: Apol’s Answer

**4. Consider** a database “mydb” with a table “messages” (id,name,message,date).

The table has the data:

|  |  |  |
| --- | --- | --- |
| id | name | message |
| 1 | peter | Hello all |
| 2 | paul | Hello |
| 3 | melissa | Hi |

The “date” col will be created with the data insertion.

*Output:*

Prepare a set of PHP scripts so that giving a name (i.e. peter) can recognize that it is in the table.

*Process:*

The name will be inserted by a form having the choices: “login”, “reset”.

In case that the name (i.e. ann) is not defined in the table a message (“not in”) would appear.

When you insert various names that are not in the list the answer (“not in”) will appear until you will insert an existing name (i.e. paul) and a message (welcome paul) will come up.

After that a choice of “logout” will terminate the process. Using session will facilitate the process.

Before start is better to define the scripts with the form that will be used considering also the session case.

**DATABASE SCRIPT USED IN BOTH METHODS:**

<?php

    //connect to myPHPadmin SQL Server

    $dbc = mysqli\_connect('localhost', 'root', '');

    //query to create DATABASE booksa

    $query = "CREATE DATABASE IF NOT EXISTS mydb";

    if (!mysqli\_query($dbc, $query))

        echo "Error creating database: " . mysqli\_error($dbc);

    //select mydb to use

    $query = "USE mydb";

    if (!mysqli\_query($dbc, $query))

        echo "Error selecting database: " . mysqli\_error($dbc);

    //query to create TABLE mydb

    $query = "CREATE TABLE IF NOT EXISTS messages(

        id INT AUTO\_INCREMENT,

        name VARCHAR(255) NOT NULL,

        message TEXT NOT NULL,

        date DATE NOT NULL,

        PRIMARY KEY (id)

        )";

    if (!mysqli\_query($dbc, $query))

        echo "Error creating  table: " . mysqli\_error($dbc);

    //Check for peter's entry

    $query = "SELECT name FROM messages WHERE name='peter'";

    $result = @mysqli\_query($dbc, $query);

    if (mysqli\_num\_rows($result) == 0) {

        $query = "INSERT INTO messages VALUES

        (DEFAULT, 'peter', 'Hello all', NOW())";

        if (!mysqli\_query($dbc, $query))

            echo "Error inserting data into messages table: " . mysqli\_error($dbc);

    }

    //Check for paul's entry

    $query = "SELECT name FROM messages WHERE name='paul'";

    $result = @mysqli\_query($dbc, $query);

    if (mysqli\_num\_rows($result) == 0) {

        $query = "INSERT INTO messages VALUES

        (DEFAULT, 'paul', 'Hello', NOW())";

        if (!mysqli\_query($dbc, $query))

            echo "Error inserting data into messages table: " . mysqli\_error($dbc);

    }

    //Check for melissa's entry

    $query = "SELECT name FROM messages WHERE name='melissa'";

    $result = @mysqli\_query($dbc, $query);

    if (mysqli\_num\_rows($result) == 0) {

        $query = "INSERT INTO messages VALUES

        (DEFAULT, 'melissa', 'Hi', NOW())";

        if (!mysqli\_query($dbc, $query))

            echo "Error inserting data into messages table: " . mysqli\_error($dbc);

    }

    mysqli\_close($dbc);

?>

**WITHOUT SESSIONS: CODE:**

login.php:

<!DOCTYPE HTML>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <title>Login</title>

</head>

<body>

<?php

require 'createDatabase.php';

$errors = []; // Initialize an empty array to store errors

// Create the databse connection

$dbc = @mysqli\_connect("localhost", 'root', '', 'mydb') OR die('Could not connect MYSQL: '. mysqli\_connect\_error());

// If form is submitted

if ($\_SERVER["REQUEST\_METHOD"] == "POST")

{

    // Check if empty of each part of form

    if (empty($\_POST['name']) && !isset($\_POST['name']))

    {

        $errors[] = "Name is required";

    }

    else

    {

        $name = mysqli\_real\_escape\_string($dbc, strtolower(trim($\_POST['name'])));

    }

    // If there are no errors, proceed with insertion

    if (empty($errors))

    {

        // Make the query:

        $query = "SELECT name FROM messages WHERE name='$name'";

        $result = @mysqli\_query($dbc, $query);

        if (!mysqli\_query($dbc, $query))

            echo "Error inserting data into messages table: " . mysqli\_error($dbc);

        if (mysqli\_num\_rows($result) <= 0)

        {

            echo '<h1>NOT IN</h1>';

        }

        else

        {

            //user will be logged in to the website

            header("Location: loggedin.php?name=$name");

        }

    }

    else

    { //Report the errors.

        echo "<h3>Error List:</h3>";

        echo "<ul>";

        foreach ($errors as $error)

        {

            echo "<li>$error</li>";

        }

        echo "</ul>";

    }

}

// Close connection

mysqli\_close($dbc);

?>

<h1>Login</h1>

<form action="login.php" method="post">

<p>Name: <input type="text" id="name" name="name" size="15" maxlength="20" autocomplete="off"></p>

<p><input type="submit" name="Submit" value="login"> <input type="reset" name="Reset" value="reset"></p>

</form>

</body>

</html>

Loggedin.php

<!DOCTYPE HTML>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <title>Logged In</title>

</head>

<body>

<?php

session\_start();

// Check for a valid user ID, through GET or POST:

if ((isset($\_SESSION['name'])) && (!empty($\_SESSION['name'])))

{

    $name = $\_SESSION['name'];

}

else

{

    echo '<p class="error">This page has been accessed in error.</p>';

    exit();

}

echo "<h1 align=\"center\">Welcome $name</h1>";

?>

<form action="login.php" method="get">

<p align="center"><input type="submit" name="Submit" value="logout"></p>

</form>

<?php

$\_SESSION = [];

session\_destroy();

setcookie('PHPSESSID', '', time()-3600, '/', '', 0, 0);

?>

</body>

</html>

**WITH SESSION: CODE:**

login.php:

<!DOCTYPE HTML>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <title>Login</title>

</head>

<body>

<?php

require 'createDatabase.php';

$errors = []; // Initialize an empty array to store errors

// Create the databse connection

$dbc = @mysqli\_connect("localhost", 'root', '', 'mydb') OR die('Could not connect MYSQL: '. mysqli\_connect\_error());

// If form is submitted

if ($\_SERVER["REQUEST\_METHOD"] == "POST")

{

    // Check if empty of each part of form

    if (empty($\_POST['name']) && !isset($\_POST['name']))

    {

        $errors[] = "Name is required";

    }

    else

    {

        $name = mysqli\_real\_escape\_string($dbc, strtolower(trim($\_POST['name'])));

    }

    // If there are no errors, proceed with insertion

    if (empty($errors))

    {

        // Make the query:

        $query = "SELECT name FROM messages WHERE name='$name'";

        $result = @mysqli\_query($dbc, $query);

        if (!mysqli\_query($dbc, $query))

            echo "Error inserting data into messages table: " . mysqli\_error($dbc);

        if (mysqli\_num\_rows($result) <= 0)

        {

            echo '<h1>NOT IN</h1>';

        }

        else

        {

            session\_start();

            $\_SESSION['name'] = $name;

            //user will be logged in to the website

            header("Location: loggedin.php");

        }

    }

    else

    { //Report the errors.

        echo "<h3>Error List:</h3>";

        echo "<ul>";

        foreach ($errors as $error)

        {

            echo "<li>$error</li>";

        }

        echo "</ul>";

    }

}

// Close connection

mysqli\_close($dbc);

?>

<h1>Login</h1>

<form action="login.php" method="post">

<p>Name: <input type="text" id="name" name="name" size="15" maxlength="20" autocomplete="off"></p>

<p><input type="submit" name="Submit" value="login"> <input type="reset" name="Reset" value="reset"></p>

</form>

</body>

</html>

Loggedin.php

<!DOCTYPE HTML>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <title>Logged In</title>

</head>

<body>

<?php

session\_start();

// Check for a valid user ID, through GET or POST:

if ((isset($\_SESSION['name'])) && (!empty($\_SESSION['name'])))

{

    $name = $\_SESSION['name'];

}

else

{

    echo '<p class="error">This page has been accessed in error.</p>';

    exit();

}

echo "<h1 align=\"center\">Welcome $name</h1>";

?>

<form action="login.php" method="get">

<p align="center"><input type="submit" name="Submit" value="logout"></p>

</form>

<?php

$\_SESSION = [];

session\_destroy();

setcookie('PHPSESSID', '', time()-3600, '/', '', 0, 0);

?>

</body>

</html>

**XAAMP QUESTION 4 OUTPUT: (same for both)**

**A screenshot of a computer

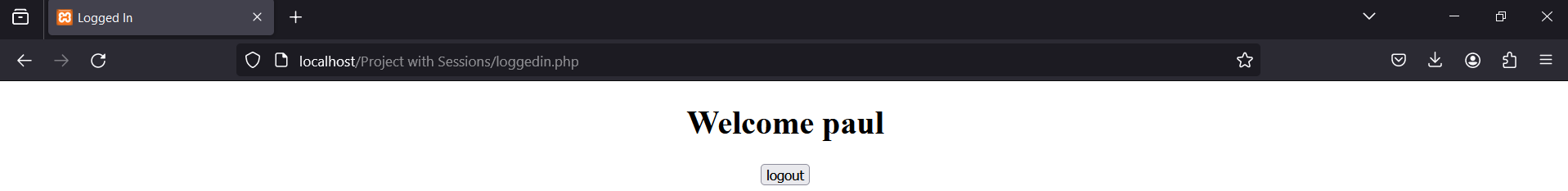
Description automatically generated**

**A screenshot of a computer

Description automatically generated**

A screenshot of a computer

Description automatically generated

MongoDB Question 1: Apol’s Answer

**1. Games.** Consider the following game table of scores:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| name | age | Score 1 | Score 2 | Score 3 |
| peter | 20 | 4 | 6 | 5 |
| paul | 22 | 10 | 4 | 6 |
| jim | 25 | 5 | 10 | 7 |
| george | 19 | 3 | 9 | 8 |

a.Create a collection “games”, and find the players that have ages between 20 and 25

use ProjQ1;

db.createCollection("games");

db.games.insertMany([{name:"peter", age:20, Score1: 4, Score2: 6, Score3: 5},

{name:"paul", age:22, Score1: 10, Score2: 4, Score3: 6},

{name:"jim", age:25, Score1: 5, Score2: 10, Score3: 7},

{name:"george", age:19, Score1: 3, Score2: 9, Score3: 8}]);

db.games.find( {$and : [{age : {$gte : 20}},{age : {$lte : 25}}]});

A black screen with white text

Description automatically generated

A black screen with yellow text

Description automatically generated

A computer screen shot of a black screen

Description automatically generated

b. find the players with ages:

(1) 19,20

db.games.find( {$or : [{age : 19},{age : 20}]});

A computer screen shot of a code

Description automatically generated

(2) 22,23,25

db.games.find( {$or : [{age : 22},{age : 23}, {age:25}]});

A computer screen with numbers and letters

Description automatically generated

c. find players with scores one =5

db.games.find({Score1: 5});

A screenshot of a computer program

Description automatically generated

d. find players with scores two=9 and age=19

db.games.find( {$and : [{Score2 : 9},{age : 19}]});

A screenshot of a computer

Description automatically generated

e. find players with scores.two between 6,10

db.games.find( {$and : [{Score2 : {$gt : 6}},{Score2 : {$lt : 10}}]});

A screenshot of a computer program

Description automatically generated

f. find players with scores.three between 6,8 and age=25

db.games.find( {$and : [{Score3 : {$gt : 6}},{Score3 : {$lt : 10}}, {age : 25}]});

A screenshot of a computer program

Description automatically generated

MongoDB Question 2: Brian’s Answer

**2. Employees**. Consider the following employee table:

|  |  |  |  |
| --- | --- | --- | --- |
| num | name | sal | job |
| 1 | peter | 3000 | engineer |
| 2 | elia | 2000 | teacher |
| 3 | mary | 1500 | technician |
| 4 | joan | 1200 | artist |

a.insert data into collection “emlo”

use employees

db.createCollection("emlo")

db.emlo.insertMany([

{ num: 1, name: "Peter", sal: 3000, job: "Engineer" },

{ num: 2, name: "Elia", sal: 2000, job: "Teacher" },

{ num: 3, name: "Mary", sal: 1500, job: "Technician" },

{ num: 4, name: "Joan", sal: 1200, job: "Artist" }

]);

A screenshot of a computer code

Description automatically generated

b.display data(pretty)

db.emlo.find().pretty();

A screenshot of a computer screen

Description automatically generated

c. update salary for employee name: “elia” by more +4000. please verify your answer

db.emlo.updateOne({ name: "Elia" }, { $inc: { sal: 4000 }});

db.emlo.find({name: "Elia"})

A screenshot of a computer program

Description automatically generated

d. update all employee with increment 5000 each

db.emlo.updateMany({},{ $inc: {sal: 5000}});

db.emlo.find()

A screen shot of a computer program

Description automatically generated

A computer screen shot of a black background

Description automatically generated

e. change the “mary” job from technician to musician.

db.emlo.updateOne({name: "Mary"},{$set: {job: "Musician"}});

db.emlo.find({name: "Mary"})

A computer screen shot of a computer code

Description automatically generated

f. add a new employee: (without insert)

num:5,name: "dimos","sal":3000,"job":"clerk"

db.emlo.findAndModify({

query: {name: "Dimos"},

update: {$setOnInsert: {num: 5, name: "Dimos", "sal": 3000, "job": "clerk"}},

new: true,

upsert: true,

});

db.emlo.find()

A computer screen with green and yellow text

Description automatically generated

A screenshot of a computer program

Description automatically generated

g. update salary of "dimos" multiplied by 3

db.emlo.updateOne(

{name: "Dimos"},

{$mul: {sal: 3}}

);

db.emlo.find({name: "Dimos"})

A computer screen shot of a code

Description automatically generated

h. find document with name that start from "d"

db.emlo.find({name: {$regex: "^[dD]"}})

A computer screen shot of a code

Description automatically generated

i. find document with name that has "r" in any position

db.emlo.find({name: {$regex: "r"}})

A screen shot of a computer

Description automatically generated

j. find document with name that has "y" at the end of position

db.emlo.find({name: {$regex: "y$"}})

A computer screen with green and white text

Description automatically generated