Batch: E-2 **Roll No.: 16010123325**

Experiment No. 09

Grade: AA / AB / BB / BC / CC / CD /DD

Signature of the Staff In-charge with date

TITLE: Design and Demonstrate database Connectivity between PHP & MYSQL

AIM: Design and Demonstrate database Connectivity between PHP & MYSQL

Expected OUTCOME of Experiment:

CO5: Apply database operations by integrating SQL queries and session variables.

Books/ Journals/ Websites referred:

1. Head First HTML5 Programming published by Shroff/O'Reilly in 2011.
2. HTML, XHTML, and CSS Bible, 5th Edition By Steven Schafe.
3. https://developer.mozilla.org/en-US/docs/Web/Guide/HTML/HTML5
4. https://www.w3schools.com/php/php_mysql_connect.asp

Pre Lab/ Prior Concepts:

PHP

The PHP Hypertext Preprocessor (PHP) is a programming language that allows web developers to create dynamic content that interacts with databases. PHP is basically used for developing web based software applications. This tutorial helps you to build your base with PHP.

The different server side technologies should be considered in the design of web pages.

The technology used can be PHP, ASP, JSP, ASP.NET etc.

PHP started out as a small open source project that evolved as more and more people found out how useful it was. Rasmus Lerdorf unleashed the first version of PHP way back in 1994.

- PHP is a recursive acronym for "PHP: Hypertext Preprocessor".
- PHP is a server side scripting language that is embedded in HTML. It is used to manage dynamic content, databases, session tracking, even build entire e-commerce sites.
- It is integrated with a number of popular databases, including MySQL, PostgreSQL, Oracle, Sybase, Informix, and Microsoft SQL Server.
- PHP is pleasingly zippy in its execution, especially when compiled as an Apache module on the Unix side. The MySQL server, once started, executes even very complex queries with huge result sets in record-setting time.
- PHP supports a large number of major protocols such as POP3, IMAP, and LDAP. PHP4 added support for Java and distributed object architectures (COM and CORBA), making n-tier development a possibility for the first time.
- PHP is forgiving: PHP language tries to be as forgiving as possible.
- PHP Syntax is C-Like.

Common uses of PHP

- PHP performs system functions, i.e. from files on a system it can create, open, read, write, and close them.
- PHP can handle forms, i.e. gather data from files, save data to a file, through email you can send data, return data to the user.
- You add, delete, modify elements within your database through PHP.
- Access cookies variables and set cookies.
- Using PHP, you can restrict users to access some pages of your website.
- It can encrypt data.

Characteristics of PHP

Five important characteristics make PHP's practical nature possible –

- Simplicity
- Efficiency
- Security
- Flexibility
- Familiarity

"Hello World" Script in PHP

To get a feel for PHP, first start with simple PHP scripts. Since "Hello, World!" is an essential example, first we will create a friendly little "Hello, World!" script.

As mentioned earlier, PHP is embedded in HTML. That means that in amongst your normal HTML (or XHTML if you're cutting-edge) you'll have PHP statements like this -

```
<html>
   <head>
   <title>Hello World</title>
  </head>
   <body>
   <?php echo "Hello, World!";?>
  </body>
</html>
```

It will produce following result -

```
Hello, World!
```

If you examine the HTML output of the above example, you'll notice that the PHP code is not present in the file sent from the server to your Web browser. All of the PHP present in the Web page is processed and stripped from the page; the only thing returned to the client from the Web server is pure HTML output.

All PHP code must be included inside one of the three special markup tags ATE are recognised by the PHP Parser.

```
<?php PHP code goes here ?>
<?    PHP code goes here ?>
```

```
<script language="php"> PHP code goes here </script>
```

A most common tag is the <?php...?> and we will also use the same tag in our tutorial.

From the next chapter we will start with PHP Environment Setup on your machine and then we will dig out almost all concepts related to PHP to make you comfortable with the PHP language.

Installation of Php:

In order to develop and run PHP Web pages three vital components need to be installed on your computer system.

- Web Server – PHP will work with virtually all Web Server software, including Microsoft's Internet Information Server (IIS) but then most often used is freely available Apache Server. Download Apache for free here – https://httpd.apache.org/download.cgi
- Database – PHP will work with virtually all database software, including Oracle and Sybase but most commonly used is freely available MySQL database. Download MySQL for free here – https://www.mysql.com/downloads/
- PHP Parser – In order to process PHP script instructions a parser must be installed to generate HTML output that can be sent to the Web Browser. This tutorial will guide you how to install PHP parser on your computer.

The INSERT INTO statement is used to insert new rows in a database table.

Syntax Insert Data Into MySQL

1. The SQL query must be quoted in PHP
2. String values inside the SQL query must be quoted
3. Numeric values must not be quoted
4. The word NULL must not be quoted

The INSERT INTO statement is used to add new records to a MySQL table:

INSERT INTO table_name (column1, column2, column3,...)
VALUES (value1, value2, value3,...)

Inserting Data into a MySQL Database Table

Now that you've understood how to create database and tables in MySQL. In this experiment you will learn how to execute SQL query to insert records into a table.

The INSERT INTO statement is used to insert new rows in a database table.

Let's make a SQL query using the INSERT INTO statement with appropriate values, after that we will execute this insert query through passing it to the PHP mysqli_query () function to insert data in table.

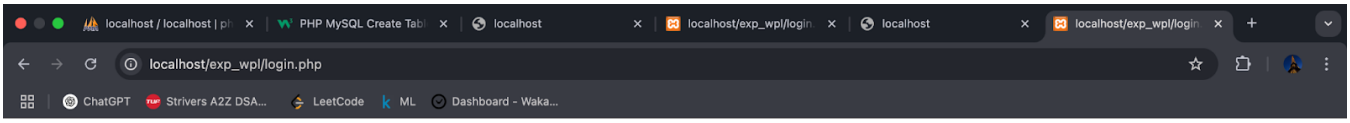
Implementation Details: Code

In this section students need to specify how the details are applied for server side programming in their application. for the following

Design and Demonstrate database Connectivity between PHP & MYSQL
a) Write a PHP code to create a database & table in Mysql.
b) Write a PHP code to insert, delete, select and Update the data from the database.
c) Create Dynamic, Interactive and database -Driven web application Individually using php & mysql.

Connection created

```php
<?php
$host = 'localhost';
$dbname = 'user_registration';
$username = 'root';
$password = '';

try {
    $pdo = new PDO("mysql:host=$host;dbname=$dbname", $username, $password);
    $pdo->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
} catch(PDOException $e) {
    die("Connection failed: " . $e->getMessage());
}
?>
```

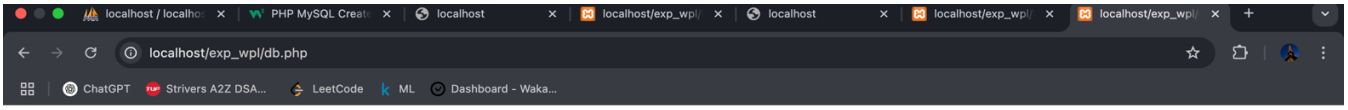


Connected successfully

Database created

```php
<?php
$servername = "localhost";
$username = "root";
$password = "";

// Create connection
$conn = new mysqli($servername, $username, $password);
// Check connection
if ($conn->connect_error) {
```

```
  die("Connection failed: " . $conn->connect_error);
}

// Create database
$sql = "CREATE DATABASE db_login";
if ($conn->query($sql) === TRUE) {
  echo "Database created successfully";
} else {
  echo "Error creating database: " . $conn->error;
}

$conn->close();
?>
```



Database created successfully

Table created

```php
<?php
$servername = "localhost";
$username = "root";
$password = "";
$dbname = "db_login";

// Create connection
$conn = new mysqli($servername, $username, $password, $dbname);
// Check connection
if ($conn->connect_error) {
  die("Connection failed: " . $conn->connect_error);
}

// sql to create table
$sql = "CREATE TABLE ecologin(
id INT(6) UNSIGNED AUTO_INCREMENT PRIMARY KEY,
firstname VARCHAR(30) NOT NULL,
lastname VARCHAR(30) NOT NULL,
email VARCHAR(50),
reg_date TIMESTAMP DEFAULT CURRENT_TIMESTAMP ON UPDATE CURRENT_TIMESTAMP
)";

if ($conn->query($sql) === TRUE) {
  echo "Table ecologin created successfully";
} else {
  echo "Error creating table: " . $conn->error;
}
$conn->close();
?>
```

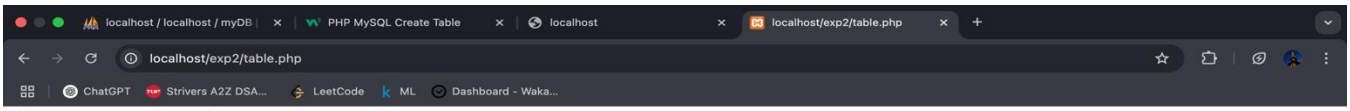


Table MyGuests created successfully

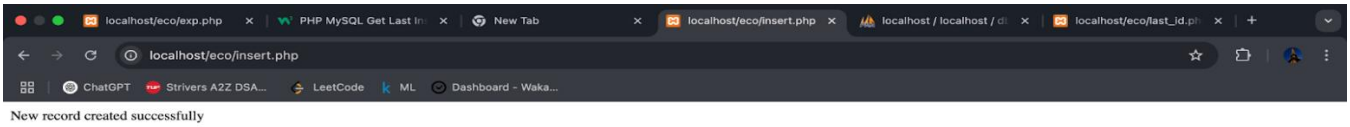
Insert/ Insert Multiple/Select/Last id/Where/Order /Delete /Update

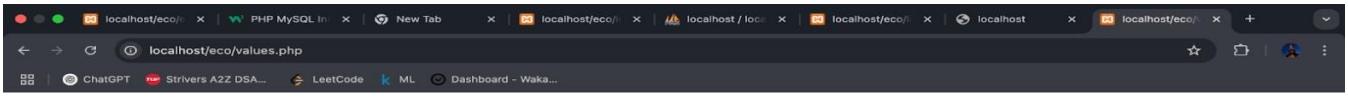
```php
<?php
require_once 'db_connect.php';

function insertTestUser($pdo) {
    try {
        $stmt = $pdo->prepare("INSERT INTO users (username, email, password, phone,
gender, terms)
                               VALUES (?, ?, ?, ?, ?, ?)");

        $password = password_hash("test123", PASSWORD_DEFAULT);

        $success = $stmt->execute([
            "testuser",
            "test@example.com",
            $password,
            "1234567890",
            "male",
            1
        ]);

        echo "Insert Test: " . ($success ? "Success" : "Failed") . "<br>";
    } catch(PDOException $e) {
        echo "Insert Error: " . $e->getMessage() . "<br>";
    }
}

function selectAllUsers($pdo) {
    try {
        $stmt = $pdo->query("SELECT * FROM users");
        echo "<h3>All Users:</h3>";
        while($row = $stmt->fetch(PDO::FETCH_ASSOC)) {
            echo "Username: " . $row['username'] . " | Email: " . $row['email'] .
"<br>";
        }
    } catch(PDOException $e) {
        echo "Select Error: " . $e->getMessage() . "<br>";
    }
}
```

```php
function selectWithWhere($pdo) {
    try {
        $stmt = $pdo->prepare("SELECT * FROM users WHERE gender = ?");
        $stmt->execute(['male']);
        echo "<h3>Male Users:</h3>";
        while($row = $stmt->fetch(PDO::FETCH_ASSOC)) {
            echo "Username: " . $row['username'] . " | Email: " . $row['email'] .
"<br>";
        }
    } catch(PDOException $e) {
        echo "Where Error: " . $e->getMessage() . "<br>";
    }
}

function selectWithOrder($pdo) {
    try {
        $stmt = $pdo->query("SELECT * FROM users ORDER BY username ASC");
        echo "<h3>Users Ordered by Username:</h3>";
        while($row = $stmt->fetch(PDO::FETCH_ASSOC)) {
            echo "Username: " . $row['username'] . " | Email: " . $row['email'] .
"<br>";
        }
    } catch(PDOException $e) {
        echo "Order Error: " . $e->getMessage() . "<br>";
    }
}

function deleteUser($pdo, $email) {
    try {
        $stmt = $pdo->prepare("DELETE FROM users WHERE email = ?");
        $success = $stmt->execute([$email]);
        echo "Delete Test: " . ($success ? "Success" : "Failed") . "<br>";
    } catch(PDOException $e) {
        echo "Delete Error: " . $e->getMessage() . "<br>";
    }
}

// Run all tests
echo "<h2>Database Operation Tests</h2>";
```

```php
// Insert test
insertTestUser($pdo);

// Select all test
selectAllUsers($pdo);

// Where clause test
selectWithWhere($pdo);

// Order by test
selectWithOrder($pdo);

// Delete test
deleteUser($pdo, "test@example.com");

// Show final results
echo "<h3>Final Results after Delete:</h3>";
selectAllUsers($pdo);
?>
```

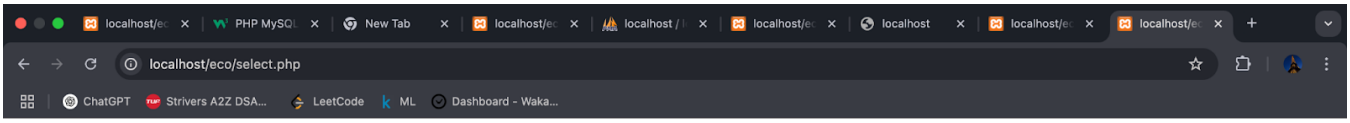
Insert multiple & single





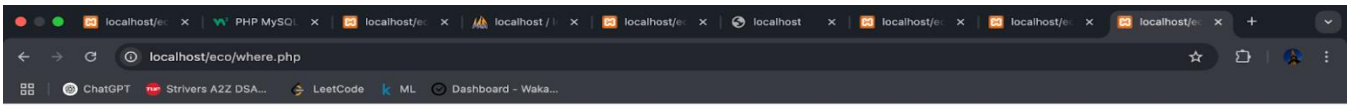
New records created successfully

Select



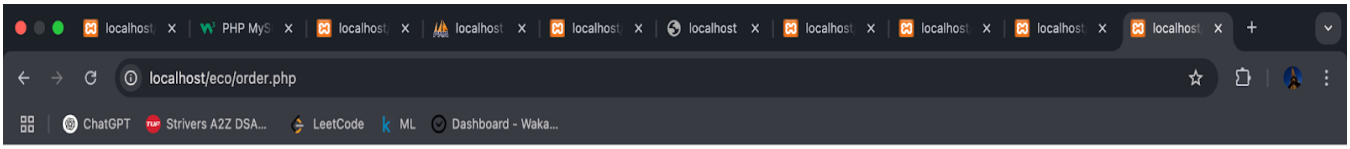
id: 1 - Name: John Doe
id: 2 - Name: Shreya Menon
id: 3 - Name: Mary Moe
id: 4 - Name: Julie Dooley

Where



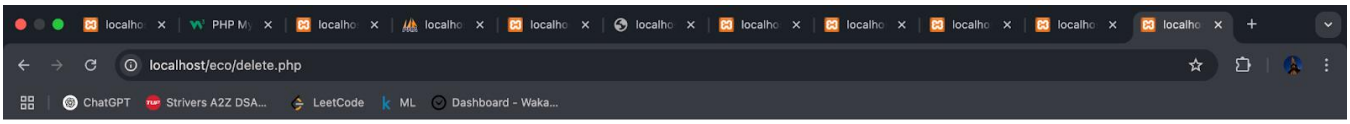
id: 2 - Name: Shreya Menon

Order by



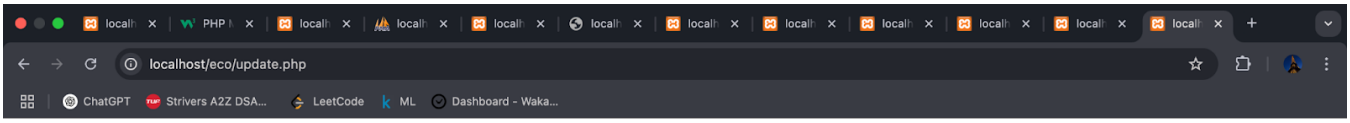
id: 1 - Name: John Doe
id: 4 - Name: Julie Dooley
id: 2 - Name: Shreya Menon
id: 3 - Name: Mary Moe

Delete



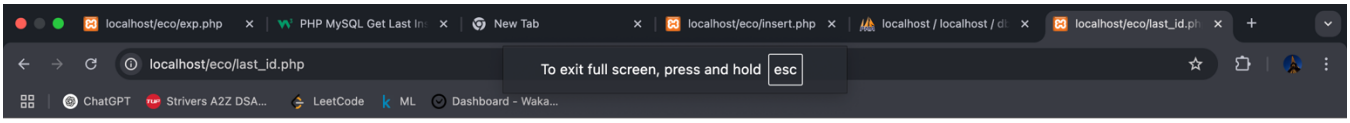
Record deleted successfully

Update



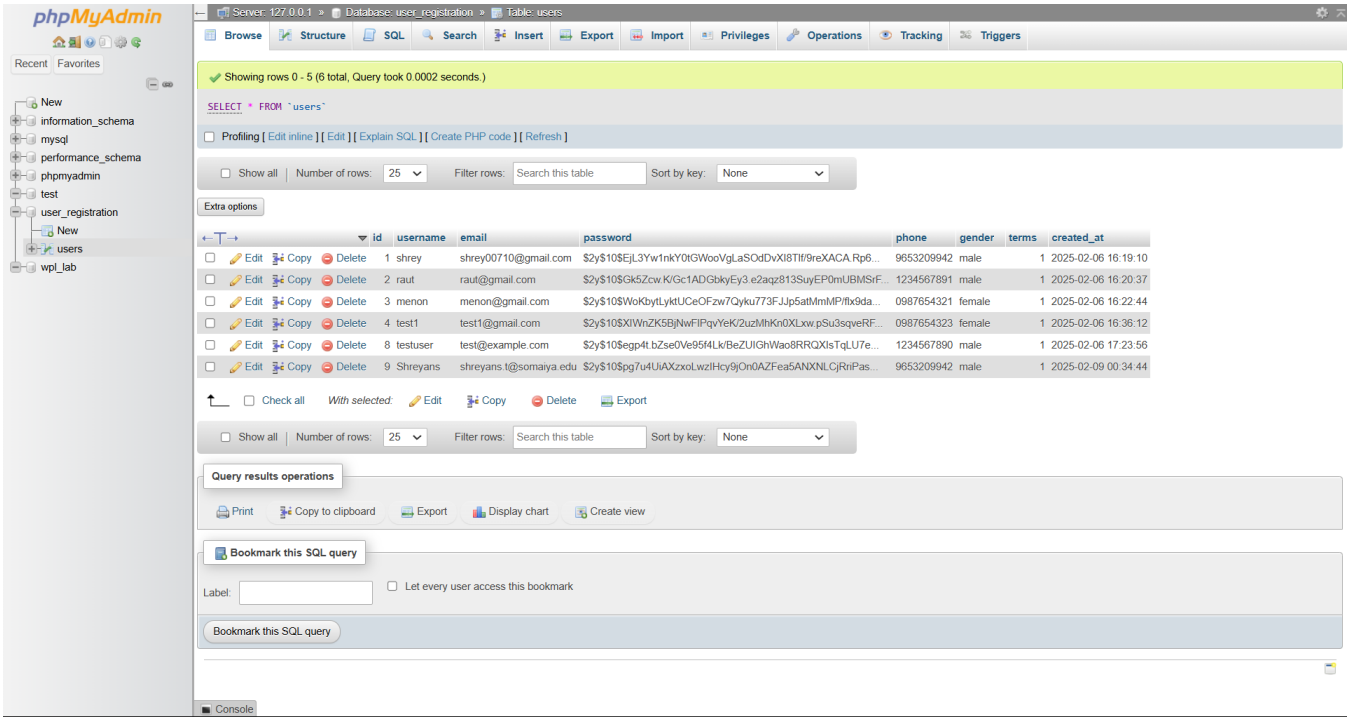
Record updated successfully

Last id



New record created successfully. Last inserted ID is: 2

Database



Steps for execution of the code

1. **Establish Database Connection** – Create a connection between PHP and MySQL using mysqli
2. **Create Database** – Use a PHP script to create a database if it does not already exist
3. **Create Table** – Define and create a table within the database
4. **Insert Data** – Add single and multiple records into the table
5. **Retrieve Data** – Select and display records using various queries (WHERE, ORDER BY, SELECT)
6. **Update Records** – Modify existing records in the database
7. **Delete Records** – Remove specific records based on a condition
8. **Fetch Last Inserted ID** – Retrieve the ID of the last inserted record

Conclusion:

The experiment successfully demonstrates PHP-MySQL connectivity, enabling essential database operations like creation, insertion, retrieval, updating, and deletion. It provides hands-on experience in server-side programming **and** database management, essential for dynamic web development.

Post Lab Descriptive Questions (Add questions from examination point view)

1) What are the advantages of server side technologies that you used?

Advantages of Server-Side Technologies Used (PHP) :

- **Security**: Sensitive operations are processed on the server, keeping data secure
- **Dynamic Content**: Generates customized responses for users
- **Database Interaction**: Enables seamless data storage, retrieval, and manipulation
- **Reduced Client Load**: Offloads processing from the client to the server, improving performance

2) Which function is used for database connectivity in PHP?

Procedural Approach – mysqli_connect()

- This function establishes a connection to the MySQL database using a simple function call.

Object-Oriented Approach – new mysqli()

- This method creates an instance of the mysqli class to connect to the database.

3) How would you redirect the page in PHP?

In PHP, you can redirect users to another page using the header() function. This function sends a raw HTTP header to the browser, instructing it to navigate to a different URL.

Syntax-

```
header("Location: newpage.php");

exit();
```

Date: **Signature of faculty in-charge**