- **PHP and MySQL with HTML** are used together to create an interactive and dynamic webpage.
- First, we need a local server such as **XAMPP**. After installing it, we start **MySql** , in addition to **Apache** which is a web server software that acts as an intermediary between the user and the server. It receives the user's request, forwards it to PHP for processing (such as calculations or database operations), and then sends the result back to the browser so it can be displayed on the screen.
- We need to run the PHP and HTML files through our local server.
- In the browser's address bar, type the file path using localhost, for example:
  localhost/Myproject/first.html
  localhost/Myproject/first.php
- (Make sure the folder name matches the one where you saved your files.)

---

html file (Receive the user's name and print it with PHP.)

```
<!DOCTYPE html>
<html>
  <body>
    <!--create a form that sends the user's data to first.php using the POST method when the form is submitted..-->
    <form action="first.php" method="post">
      <label for="fname"> my name </label>
      <!-- The name attribute is used to access this input value in PHP -->
      <input type="text" id="fname" name="myname">
      <input type="submit">
    </form>
  </body>
```

```
</html>
```

php file

```
<?php
$n1=$_POST['myname'];
echo "my first name is ". $n1;
?>
```

---

html file (Receive the user's quantity and print the total price with PHP.)

```
<!DOCTYPE html>
<html>
  <body>
    <!--create a form that sends the user's data to first.php using the POST method when the form is submitted..-->
    <form action="first.php" method="post">
      <label for="quant"> quantity </label>
      <!-- The name attribute is used to access this input value in PHP -->
      <input type="text" id="quant" name="myquant">
      <input type="submit">
    </form>
  </body>
  </html>
```

php file

```
<?php
```

```
$q1=$_POST['myquant'];

echo "  my total price is ". ($q1*10);

?>
```

Notes::

- Put calculations inside **parentheses** if using them in echo.
- PHP statements must end with **; (semicolon).**
- Inside echo, separate strings and variables with **. (dot operator)**.
- <**form**> → This tag creates a form on the webpage where users can **enter data.**
- **action**="first.php" → Specifies where the form data will be sent when the user clicks submit. In this case, the **data goes to the file first.php for processing**.
- method="**post**" → Defines how the data is **sent**:
- $q1 → a **PHP variable** used to store a value.
- $_POST → a **superglobal array** in PHP that stores data sent from a form using **POST method**.
- 'myquant' → the **name of the input field** in the HTML form.
- = → **assignment operator**, used to store a value in a variable.

To insert the user variables inside database follow these steps:

Open the localhost database through a link

localhost / 127.0.0.1 | phpMyAdmin 5.2.1

1- Go to **Databases** → enter the name you want (e.g., cat) → click **Create**.
2- **Create a Table**
3- Open your new database → click **Create table** → enter table name (e.g., myorder)
4- Specify the number of columns you need (e.g., 3) → click **Go**.
5- Add column names and types,
6- Then write html and php files as follows::

html file (Receive the user's orders then compute the total and print them with PHP.)

```
<!DOCTYPE html>
<html>
  <body>
    <!--create a form that sends the user's data to first.php using the POST method when the form is submitted..-->
    <form action="first.php" method="post">
      <!-- The name attribute is used to access this input value in PHP -->
      <table>
        <tr>
          <td>Item</td>
          <td>quantity</td>
        </tr>
        <tr>
          <td>apple</td>
          <td><input type="text" name="ap"></td>
        </tr>
        <tr>
          <td>banana</td>
          <td><input type="text" name="ba"></td>
        </tr>
```

```
        </table>
        <input type="submit">
      </form>
   </body>
   </html>
```

Php file

```
<?php
//define user values
$apq=$_POST['ap'];
$baq=$_POST['ba'];
$totalq=$apq*20+$baq*15;
// 1. Set up database connection details
$servername="localhost";    // server name (localhost for local server)
$username="root";          // database username (default is 'root' for XAMPP)
$password="";              // database password (empty by default in XAMPP)
$dbname="cat";             // name of the database you created
// 2. Create a new connection object to connect PHP with MySQL
$conn=new mysqli($servername,$username,$password,$dbname);
// 3. Prepare an SQL query to insert user data into 'myorder' table
// 'ap_q', 'ba_q', 'total_q' are column names in the table
// $apq, $baq, $totalq are PHP variables storing user input
$sql="INSERT INTO myorder(ap_q,ba_q,total_q)VALUES($apq, $baq, $totalq)";
// 4. Execute the SQL query using the connection object
```

```
$conn->query($sql);

// if you want(5. Display a confirmation message and show the order details)

echo"<h3>order processed succssefully</h3>";

echo "<p> apple order : $apq </p>";

echo "<p> banana order: $baq </p>";

echo "<p> total order: $totalq</p>";

?>
```

Notes::

1- -> is the object operator in PHP
2- $conn → is the **object** representing your database connection.
3- ->query() → is **calling the query method** of that object to run the SQL command (which is insert in this code)
4- query() is a method that can run (insert, delete, select , or update)
5- query() takes the text ($sql) from php to mysql to execute it as an insert command.
6- new mysqli() create a new object from the php built in class mysqli