PHP GET AND POST

What is PHP Form Handling?

- PHP is widely used for handling HTML forms.
- Forms collect data from users and send it to the server for processing.
- PHP offers two main methods to handle form data: **GET** and **POST**.
- Both methods are used in conjunction with HTML forms and determine how the data is sent to the server.
- **Purpose of form handling:** To capture user input and use it for tasks such as logging in, searching, etc.

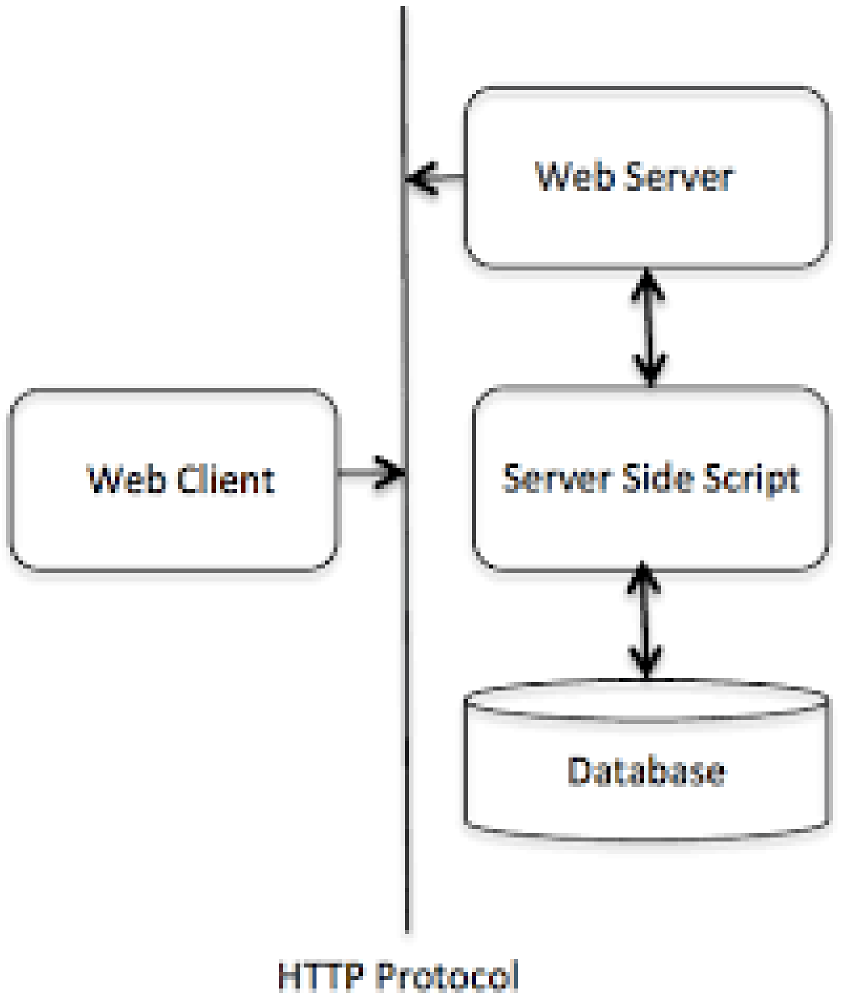
- When working with web forms, PHP provides two primary methods for submitting data to a server: **GET** and **POST**. These methods are part of the HTTP protocol and serve different purposes depending on how data is submitted.

✓**GET**: Used to retrieve data from the server.

✓**POST**: Used to send data to the server for processing.

Both methods are part of how a browser communicates with a web server, and understanding their differences is essential for working with forms and handling data in web applications.

How HTTP protocol works ?



GET vs POST Overview

- **GET** and **POST** are two HTTP methods used to send form data to the server.
- **GET**: Appends form data to the URL (visible in the browser's address bar).
- **POST**: Sends form data in the request body (not visible in the URL).

- When to use GET: For retrieving data (e.g., search forms).
- When to use POST: For sending sensitive or large amounts of data (e.g., login forms).

Form Structure in HTML

```
<form action="process.php" method="get">
    <input type="text" name="username" >
    <input type="submit" value="Submit">
</form>
```

The <form> element defines how the data will be sent (GET/POST) and the action URL (where the data is sent).

PHP Form Handling with GET and POST

- After submitting a form, PHP can access the form data using two superglobal arrays: $_POST and $_GET.
- **GET method** uses $_GET[].
- **POST method** uses $_POST[].

Introduction to the GET Method

- The **GET** method is used when you want to retrieve data from the server. When a user submits a form using GET, the form data is appended to the URL in the form of a query string.
- This data is collected by the predefined **$_GET variable** for processing.
- The information sent from an HTML form using the GET method is visible to everyone in the browser's address bar, which means that all the variable names and their values will be displayed in the URL. Therefore, the get method is not secured to send sensitive information.
- Example http://example.com/form.php?name=John&age=25

Characteristics of “GET” method

- **Data Visibility**: The submitted data is visible in the URL.
- **URL Length Limitation**: Some browsers and servers have limits on the length of URLs, typically around 2000 characters.
- **Caching**: GET requests are cached by default. As a result, browsers may store GET requests, leading to improved performance when retrieving the same resource multiple times.
- **Bookmarking**: URLs with GET data can be bookmarked, making GET suitable for search or filter requests.

What is $_GET?

- $_GET is a superglobal array in PHP that is used to collect data sent through the URL using the HTTP GET method.
- When data is passed via the URL or query string (e.g., through a form submission with method="GET" or by appending data to the URL), it becomes accessible in PHP via $_GET.
- Use $_GET when the data being transmitted is not sensitive and should be visible.
- Syntax : **$_GET['parameter_name'];**

Characteristics of $_GET

- **Associative Array**: $_GET is an associative array where the keys are the names of the parameters passed via the URL, and the values are the corresponding data.
- **Accessible Globally**: Since $_GET is a superglobal, it can be accessed from anywhere within the PHP script, without needing to use global.
- **Used for Form Handling**: Commonly used to collect data from HTML forms when the method="GET" is specified
- **Used for Query Strings**: When data is sent as part of the URL (after a '?' symbol), it can be retrieved with $_GET.

Advantages of the GET Method

- **Bookmarkable URLs**: Users can bookmark or share URLs with parameters.
- **Simple to use**: Easy for small amounts of data.
- **Debugging**: Data is visible in the URL, making it easier to debug.
- Ideal for non-sensitive data (search queries)

Disadvantages of the GET Method

- **Limited data size**: URL length is limited, restricting the amount of data.
- **Insecure**: Sensitive data like passwords or personal info should not be sent via GET.
- Data is cached by browsers, making it potentially vulnerable.

Introduction to the POST Method

- The **POST** method is also used to submit the HTML form data. But the data submitted by this method is collected by the predefined superglobal variable **$_POST** instead of $_GET.

- Unlike the GET method, it does not have a limit on the amount of information to be sent. The information sent from an HTML form using the POST method is not visible to anyone.

- **POST** method sends data in the request body (not appended to the URL). Data is not visible in the address bar.

Characteristics of POST

- **Data Visibility**: Data is sent in the body of the request, so it’s not visible in the URL.
- **No Size Limitation**: While GET is limited by URL length, POST can handle much larger amounts of data.
- **No Caching**: POST requests are not cached by browsers.
- **No Bookmarking**: Since the data is not part of the URL, POST requests cannot be bookmarked.

What is $_POST?

- $_POST is a PHP superglobal variable used to collect form data sent through the HTTP POST method.

- It is an associative array where keys are the names of form fields, and the values are the data submitted by the user.

- Use $_POST when transmitting sensitive information such as login credentials, or when submitting large amounts of data (e.g., file uploads, form submissions with a lot of fields).

- Syntax: **$_POST['field_name'];**

Characteristics of $_POST

- **Associative Array**: $_POST stores data in key-value pairs, where the keys represent the form field names and the values are the user inputs.
- **Invisible Data Transmission**: Unlike $_GET, data sent via $_POST is not visible in the URL, making it more secure for sending sensitive information like passwords.
- **Used for Form Handling**: Commonly used to collect data from HTML forms when the method="POST" is specified.
- **No Data Size Limitation**: While there is technically a limit on data size set by the server, it is much larger than the limit for GET requests.

Advantages of the POST Method

- **No size limitation**: Data can be larger than GET.
- **More secure**: Data is sent in the body, not visible in the URL.
- **No caching**: Data is not cached by the browser.

$_REQUEST and $_SERVER variable

- The **$_REQUEST** variable is a **superglobal variable**, which can hold the content of both $_GET and $_POST variable.

- In other words, the PHP $_REQUEST variable is used to collect the form data sent by either GET or POST methods.

- **$_SERVER** in PHP is a superglobal array that holds information about headers, paths, and script locations on the server. It provides access to various server and execution environment-related data. Since it's a superglobal, it is accessible throughout the script, regardless of scope.

Example 1

Create a file get.php

```
<html>
  <body>

     <form action = "gettest.php" method = "GET">
        Username: <input type = "text" name = "username" />
<br>
        Blood Group: <input type = "text" name = "bloodgroup" />
<br>
        <input type = "submit" />
     </form>

  </body>
</html>
```

Create a file gettest.php

```
<html>
   <body>

      Welcome <?php echo $_GET["username"]; ?> </br>
      Your blood group is: <?php echo $_GET["bloodgroup"];
?>

   </body>
</html>
```

Execute get.php

OUTPUT:

```
Welcome Navneet
Your blood group is: A+
```

Example 2

Create a file post.php

```
<html>
   <body>

      <form action = "posttest.php" method = "post">
         Username: <input type = "text" name = "username" /> <br>
         Blood Group: <input type = "text" name = "bloodgroup" />
<br>
         <input type = "submit" />
      </form>

   </body>
</html>
```

Create a file posttest.php

```
<html>
   <body>

     Welcome
<?php
     echo $_POST["username"]; ?> </br>
     Your blood group is: <?php echo _POST["bloodgroup"];
 ?>

   </body>
</html>
```

Execute post.php

OUTPUT:

```
Welcome Navneet
Your blood group is: A+
```

Example 3

Create a file get-method.php

```
<!DOCTYPE html>
<html lang="en">
<head>
    <title>Example of PHP GET method</title>
</head>
<body>
<?php
if(isset($_GET["name"])){
    echo "<p>Hi, " . $_GET["name"] . "</p>";
}
?>
<form method="get" action="<?php echo $_SERVER["PHP_SELF"];?>">
    <label for="inputName">Name:</label>
    <input type="text" name="name" id="inputName">
    <input type="submit" value="Submit">
</form>
</body>
```

The PHP_SELF variable is used to get the name and path of the current file

Execute get-method.php

OUTPUT:

```
Hi, navneet kaur
```

Example 4

Create a file post-method.php

```
<!DOCTYPE html>
<html lang="en">
<head>
    <title>Example of PHP POST method</title>
</head>
<body>
<?php
if(isset($_POST["name"])){
    echo "<p>Hi, " . $_POST["name"] . "</p>";
}
?>
<form method="post" action="<?php echo $_SERVER["PHP_SELF"];?>">
    <label for="inputName">Name:</label>
    <input type="text" name="name" id="inputName">
    <input type="submit" value="Submit">
</form>
</body>
```

Execute post-method.php

OUTPUT:

```
Hi, navneet kaur
```