

**Date of submission:** 16/02/2025  
**Batch:** A1                      **Roll No.:** 16010123012  
**Div:** A  
**Student Name:** Aaryan Sharma  
**Experiment No:** 5  
**Staff In-charge:**

**TITLE: : Develop and Demonstrate the use of Form Handling and Validation in PHP**

**AIM:** To develop web forms using PHP form and Validation.

---

**Expected Outcome of Experiment:**

The expected outcomes aim to enhance understanding of the implications and trade-offs associated with different methods of form data handling in PHP.

Books/ Journals/ Websites referred:

Steve Prettyman, "Learn PHP 8 Using MySQL, JavaScript, CSS3, and HTML5",  
Apress 2nd / 2020 edition.

---

**Problem Statement:** Design and implement an application to demonstrate HTML form integration with PHP for data collection and processing.

Utilize the registration page designed in Experiment No. 1 and create PHP scripts to handle form submission and data processing as follows:

1. Create separate PHP scripts for handling form submissions using different methods:
  - **post\_registration.php:** Processes the registration form data using the `$_POST` method.
  - **get\_registration.php:** Handles the form data using the `$_GET` method.
  - **request\_registration.php:** Retrieves the submitted data using the `$_REQUEST` method.
2. Each script should validate the input fields (e.g., check for valid email format and ensure mandatory fields are filled) and display the submitted registration details in a structured format.

**Implementation and screenshots of output****HTML:**

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width, initial-
scale=1.0" />
    <title>Registration Form</title>
    <style>
      * {
        margin: 10px;
        text-align: center;
        font-family: Georgia, "Times New Roman", Times, serif;
        font-size: 20px;
      }
    </style>
  </head>
  <body>
    <h2>Registration Form</h2>
    <form action="post_registration.php" method="post">
      <label for="name">Name:</label>
      <input type="text" id="name" name="name" required /><br /><br />

      <label for="email">Email:</label>
      <input type="email" id="email" name="email" required /><br /><br />
    />

    <label for="phone">Phone:</label>
    <input type="text" id="phone" name="phone" required /><br /><br />

    <input type="submit" value="Register (POST)" />
  </form>

  <hr />

  <form action="get_registration.php" method="get">
    <label for="name">Name:</label>
    <input type="text" id="name" name="name" required /><br /><br />

    <label for="email">Email:</label>
    <input type="email" id="email" name="email" required /><br /><br />
  />
```

```
<label for="phone">Phone:</label>
<input type="text" id="phone" name="phone" required /><br /><br />

<input type="submit" value="Register (GET)" />
</form>

<hr />

<form action="request_registration.php" method="post">
  <label for="name">Name:</label>
  <input type="text" id="name" name="name" required /><br /><br />

  <label for="email">Email:</label>
  <input type="email" id="email" name="email" required /><br /><br />
/>

  <label for="phone">Phone:</label>
  <input type="text" id="phone" name="phone" required /><br /><br />

  <input type="submit" value="Register (REQUEST)" />
</form>
</body>
</html>
```

**POST:**

```
<?php
if ($_SERVER["REQUEST_METHOD"] == "POST") {
    $name = trim($_POST['name']);
    $email = trim($_POST['email']);
    $phone = trim($_POST['phone']);

    if (empty($name) || empty($email) || empty($phone)) {
        echo "All fields are required.";
        exit();
    }

    if (!filter_var($email, FILTER_VALIDATE_EMAIL)) {
        echo "Invalid email format.";
        exit();
    }

    echo "<h2>Registration Successful (POST Method)</h2>";
    echo "<p><strong>Name:</strong> $name</p>";
}
```

```
    echo "<p><strong>Email:</strong> $email</p>";
    echo "<p><strong>Phone:</strong> $phone</p>";
} else {
    echo "Invalid request method.";
}
```

**GET:**

```
<?php
if ($_SERVER["REQUEST_METHOD"] == "GET") {
    $name = trim($_GET['name']);
    $email = trim($_GET['email']);
    $phone = trim($_GET['phone']);

    if (empty($name) || empty($email) || empty($phone)) {
        echo "All fields are required.";
        exit();
    }

    if (!filter_var($email, FILTER_VALIDATE_EMAIL)) {
        echo "Invalid email format.";
        exit();
    }

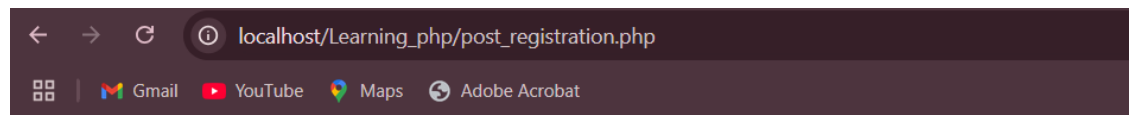
    echo "<h2>Registration Successful (GET Method)</h2>";
    echo "<p><strong>Name:</strong> $name</p>";
    echo "<p><strong>Email:</strong> $email</p>";
    echo "<p><strong>Phone:</strong> $phone</p>";
} else {
    echo "Invalid request method.";
}
```

**REQUEST:**

```
<?php
if ($_SERVER["REQUEST_METHOD"] == "POST" || $_SERVER["REQUEST_METHOD"]
== "GET") {
    $name = trim($_REQUEST['name']);
    $email = trim($_REQUEST['email']);
    $phone = trim($_REQUEST['phone']);

    if (empty($name) || empty($email) || empty($phone)) {
        echo "All fields are required.";
        exit();
    }
}
```

```
if (!filter_var($email, FILTER_VALIDATE_EMAIL)) {  
    echo "Invalid email format.";   
    exit();  
}  
  
echo "<h2>Registration Successful (REQUEST Method)</h2>";  
echo "<p><strong>Name:</strong> $name</p>";  
echo "<p><strong>Email:</strong> $email</p>";  
echo "<p><strong>Phone:</strong> $phone</p>";  
} else {  
    echo "Invalid request method.";  
}
```

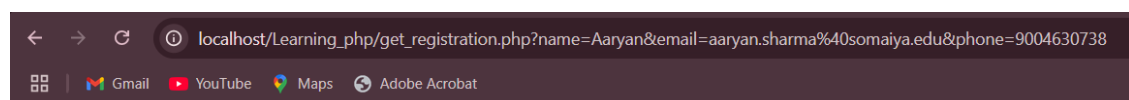


## Registration Successful (POST Method)

**Name:** Aaryan

**Email:** aaryan.sharma@somaiya.edu

**Phone:** 9004630738

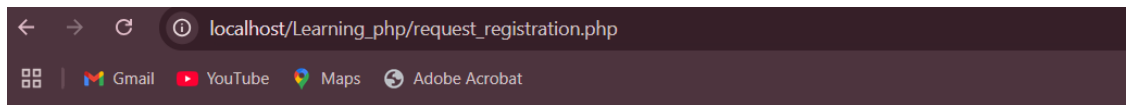


## Registration Successful (GET Method)

**Name:** Aaryan

**Email:** aaryan.sharma@somaiya.edu

**Phone:** 9004630738

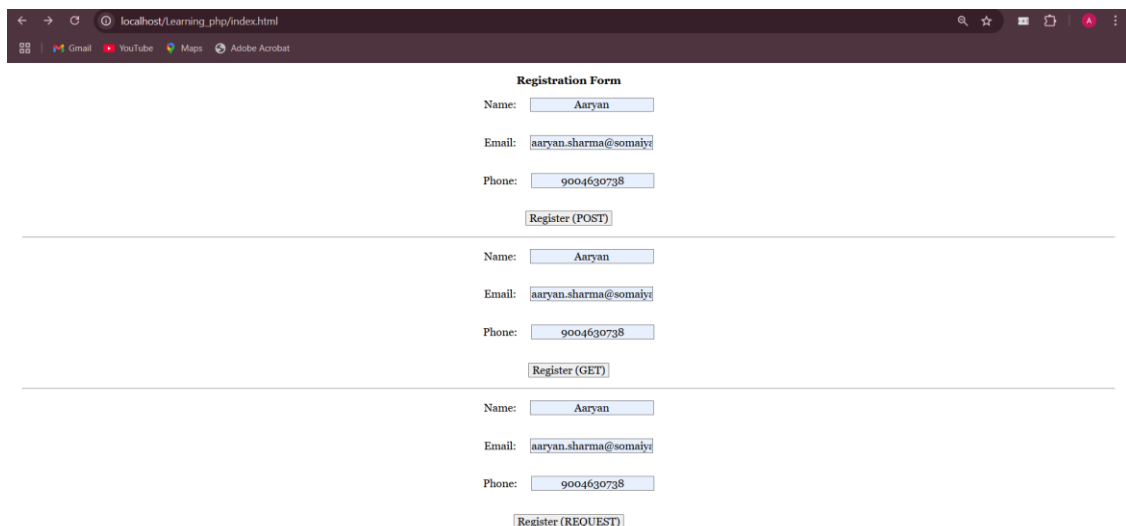


## Registration Successful (REQUEST Method)

**Name:** Aaryan

**Email:** aaryan.sharma@somaiya.edu

**Phone:** 9004630738



Registration Form

Name:

Email:

Phone:

---

Name:

Email:

Phone:

---

Name:

Email:

Phone:

### Conclusion:

I have completed this experiment and learned that `$_POST` is the most secure for handling sensitive data as it keeps information hidden from the URL. `$_GET` is the easiest to use for debugging and bookmarking but has security risks and URL length limitations. `$_REQUEST` can be unreliable since it merges data from multiple sources. This experiment has helped me understand when to use each method effectively in PHP form handling.

### Post Lab Objective with Answer:

**1. Which method (\$\_POST, \$\_GET, \$\_REQUEST) is the most secure, and why?**

\$\_POST is more secure because the data is sent in the request body, making it invisible in the URL and not stored in browser history or server logs. This prevents exposure to URL manipulation, bookmarking, and caching risks, making it the preferred method for handling sensitive user data like passwords and personal details.

**2. From a developer's perspective, which method (\$\_POST, \$\_GET, \$\_REQUEST) is easier to use, and why?**

\$\_GET is the easiest to use because debugging is simpler, as parameters are visible in the URL and can be modified directly in the browser. It allows easy bookmarking and sharing of results without requiring form submissions. Developers can test quickly without special tools. Additionally, \$\_GET works well for fetching non-sensitive data such as search queries, filters, and pagination without modifying server-side resources.

**3. How does the \$\_GET method handle data transmission, and what are its limitations?**

The \$\_GET method transmits data by appending key-value pairs to the URL, making it visible in the address bar. While this allows easy data retrieval and bookmarking, it has limitations. URL length is restricted (around 2048 characters), making it unsuitable for large data submissions. Since data is visible, it poses security risks such as exposure in browser history, logs, and potential manipulation. Additionally, \$\_GET is not ideal for sensitive data like passwords and cannot handle binary data such as file uploads, unlike \$\_POST.