Modul Ajar

Pemrograman Web

Pertemuan VI Form dan PHP

6.1 Form dan PHP

Form pada PHP merupakan salah satu komponen penting dalam pengembangan aplikasi web karena memungkinkan interaksi antara pengguna dan server. Dengan menggunakan elemen form HTML seperti <form>, <input>, <textarea>, dan <select>, pengguna dapat mengirimkan data seperti nama, email, komentar, atau pilihan tertentu. Data ini kemudian dikirim ke server melalui atribut method, biasanya menggunakan GET atau POST, dan diproses oleh file PHP yang ditentukan dalam atribut action.

Di sisi server, PHP menangani data yang dikirim melalui form dengan menggunakan variabel superglobal seperti $_GET atau $_POST. Setelah data diterima, PHP dapat memprosesnya lebih lanjut, seperti menyimpan ke database, melakukan validasi input, atau menampilkan pesan balasan kepada pengguna. Untuk menjaga keamanan, pengembang juga disarankan menggunakan fungsi seperti htmlspecialchars() untuk mencegah serangan seperti XSS (Cross-site Scripting). Dengan memanfaatkan form dan PHP secara tepat, aplikasi web dapat menjadi lebih dinamis dan responsif terhadap input dari pengguna.

6.2 Pengenalan Form HTML

Form digunakan untuk mengumpulkan data dari pengguna. Contoh kode

```
1  <form action="proses.php" method="post">
2    <label>Nama:</label>
3    <input type="text" name="nama">
4    <input type="submit" value="Kirim">
5  </form>
```

Gambar 6.1 Kode untuk Form

Attribut penting:

action : file Dimana tujuan saat form dikirim

method: metode pengiriman data (GET atau POST)

6.3 Menangani data form dengan PHP

Metode **POST** dan **GET** adalah dua cara utama dalam mengirimkan data dari form HTML ke server menggunakan PHP. Metode **POST** digunakan ketika data yang dikirim bersifat sensitif, besar, atau tidak ingin ditampilkan di URL. Data yang dikirim melalui POST tidak akan terlihat di address bar browser, sehingga lebih aman untuk digunakan dalam pengiriman informasi pribadi seperti password atau data formulir panjang. Di sisi PHP, data POST dapat diakses melalui variabel superglobal $_POST.

Sementara itu, metode **GET** mengirimkan data melalui URL, dengan menambahkan parameter setelah tanda tanya (?). Misalnya, form.php?nama=Ali. Metode ini cocok untuk permintaan yang tidak sensitif, seperti pencarian atau filter data, karena pengguna bisa menyimpan atau membagikan URL tersebut. Data GET dapat diakses menggunakan $_GET pada PHP.

Perbedaan utama antara POST dan GET terletak pada **cara pengiriman data dan keamanannya**. GET menampilkan data langsung di URL dan memiliki batas panjang karakter, sehingga kurang aman untuk data pribadi. Sedangkan POST menyembunyikan data dari URL, tidak memiliki batas panjang yang signifikan, dan lebih cocok untuk data yang kompleks atau rahasia. Pemilihan metode tergantung pada kebutuhan dan tingkat keamanan yang diperlukan oleh aplikasi web.

a. Metode POST

```php
<?php
// proses.php
$nama = $_POST['nama'];
echo "Halo, $nama!";
?>
```

Gambar 6.2 Metode POST

b. Metode GET

```html
<form action="proses.php" method="get">
  <input type="text" name="nama">
  <input type="submit" value="Kirim">
</form>
```

```php
<?php
$nama = $_GET['nama'];
echo "Halo, $nama!";
?>
```

Gambar 6.3 Metode GET

6.4 Validasi Form Sederhana

a. validasi form

Fungsi dari validasi form sederhana adalah untuk memastikan bahwa data yang dimasukkan oleh pengguna sesuai dengan aturan yang ditentukan sebelum data tersebut diproses lebih lanjut oleh server. Validasi ini berfungsi sebagai langkah awal untuk mencegah kesalahan input, seperti kolom yang kosong, format email yang tidak valid, atau data yang tidak sesuai tipe yang diharapkan. Dengan validasi, sistem dapat memberikan umpan balik langsung kepada pengguna agar mereka memperbaiki input yang salah atau kurang.

```php
<?php
if ($_SERVER["REQUEST_METHOD"] == "POST") {
  if (empty($_POST["nama"])) {
    echo "Nama wajib diisi!";
  } else {
    $nama = htmlspecialchars($_POST["nama"]);
    echo "Halo, $nama!";
  }
}
?>
```

Gambar 6.4 Metode Validasi Form

b. Menghindari XSS (Cross-site Scripting)

Cross-Site Scripting (XSS) adalah salah satu jenis serangan keamanan pada aplikasi web, di mana penyerang menyisipkan skrip berbahaya (biasanya JavaScript) ke dalam halaman web yang nantinya akan dijalankan oleh browser pengguna lain. Serangan ini terjadi ketika aplikasi web menerima input dari pengguna dan menampilkannya kembali ke halaman web tanpa

melakukan penyaringan atau validasi yang benar. Akibatnya, skrip berbahaya tersebut bisa dijalankan secara otomatis oleh browser pengguna tanpa sepengetahuan mereka.

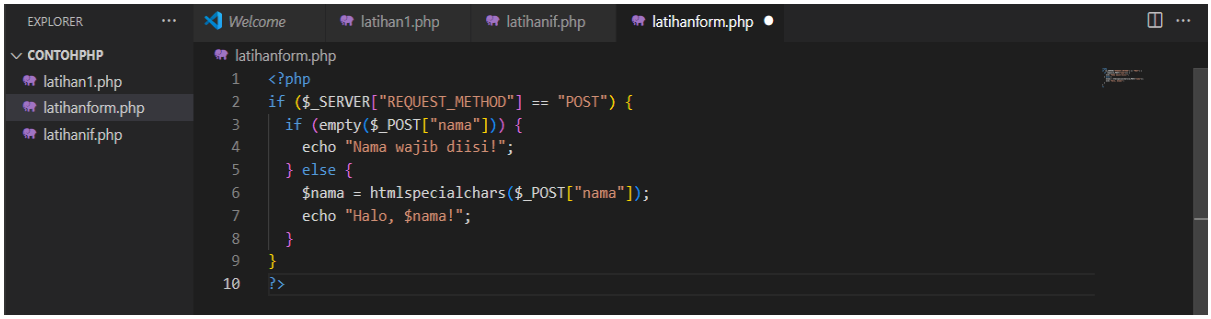
Tujuan utama dari serangan XSS adalah untuk mencuri informasi penting dari pengguna, seperti cookie, data sesi login, atau mengarahkan pengguna ke situs palsu. Misalnya, jika pengguna memasukkan <script>alert('XSS');</script> ke dalam form komentar, dan aplikasi tidak memfilter input tersebut, maka setiap orang yang membuka halaman komentar itu akan melihat popup yang dijalankan oleh skrip tersebut. Untuk mencegah XSS, developer harus selalu memvalidasi dan membersihkan input pengguna menggunakan fungsi seperti htmlspecialchars() atau library filtering lainnya agar karakter berbahaya seperti <, >, dan " tidak bisa dijalankan sebagai skrip.

```php
$nama = htmlspecialchars($_POST["nama"]);
echo "Halo, $nama!";
```

Gambar 6.5 Menghindari XSS

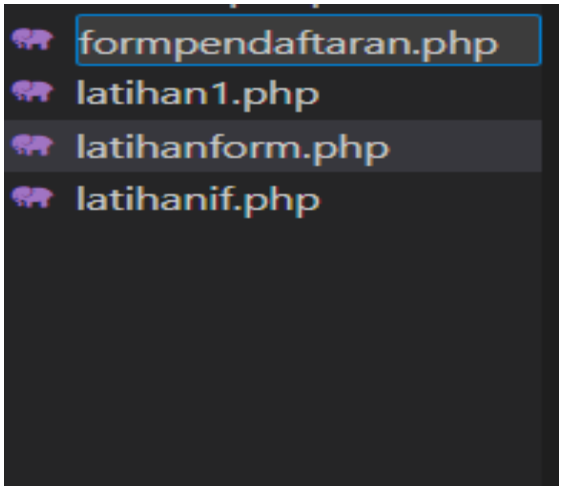
6.5 Studi Kasus Sederhana

Langkah 1. Pastikan folder yang ada di visual studio code adalah folder C:/xampp/htdocs/contohphp



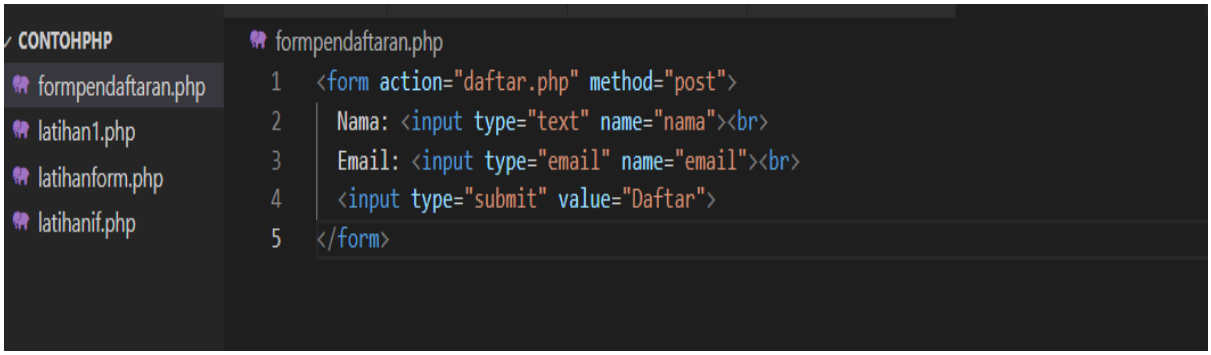
Gambar 6.6 Visual Studio Code

Langkah 2. Buat file baru dengan nama formpendaftaran.php



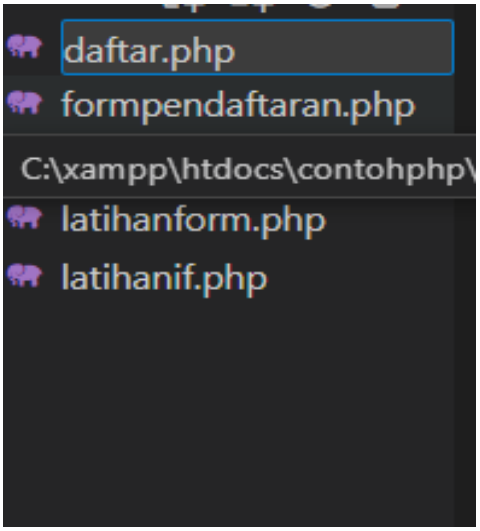
Gambar 6.7 Buat file baru formpendaftaran.php

Langkah 3. Buat kode berikut ini



Gambar 6.8 kode formpendaftaran.php

Langkah 4. Buat file baru dengan nama daftar.php



Gambar 6.9 file daftar.php

Langkah 5. Buat kode untuk file daftar.php

```php
<?php
if ($_SERVER["REQUEST_METHOD"] == "POST") {
  $nama = htmlspecialchars($_POST["nama"]);
  $email = htmlspecialchars($_POST["email"]);

  if (empty($nama) || empty($email)) {
    echo "Semua field wajib diisi!";
  } else {
    echo "Terima kasih sudah mendaftar, $nama ($email)";
  }
}
?>
```

Gambar 6.10 Kode untuk daftar.php

Langkah 6. Buka web browser, lalu arahkan addressbar ke Alamat **localhost/contohphp/formpendaftaran.php**

Gambar 6.11 Hasil form pendaftaran

Gambar 6.12 Hasil Ketika form pendaftaran berhasil di input

Tugas Mandiri

1. Buat form buku tamu yang bisa di input

   data inputan :

   a. Nama

   b. Jurusan

c. Komentar

2. Buat form sederhana untuk menghitung umur Contoh

Gambar 6.13 Tugas Form Umur

Gambar 6.14 Hasil Form Umur