

Apa itu web dinamis dan PHP

Web dinamis adalah jenis situs web yang menggunakan konten yang dapat berubah secara dinamis berdasarkan interaksi pengguna, data yang disimpan dalam database, atau logika bisnis tertentu. Berbeda dengan situs web statis yang memiliki konten tetap, situs web dinamis memungkinkan pengguna untuk berinteraksi dengan konten yang berubah sesuai dengan kebutuhan atau preferensi mereka.

PHP adalah bahasa pemrograman server-side yang sering digunakan untuk mengembangkan aplikasi web dinamis. Singkatan PHP adalah "Hypertext Preprocessor". PHP bekerja di sisi server, yang berarti kode PHP dijalankan di server web sebelum hasilnya dikirim ke browser pengguna. PHP dapat digunakan untuk melakukan berbagai tugas di situs web, termasuk mengambil dan menyimpan data dari database, menghasilkan halaman web secara dinamis, menangani formulir, dan banyak lagi. Ini adalah salah satu bahasa pemrograman paling populer yang digunakan untuk pengembangan web, terutama untuk situs web dinamis.

Echo & commentar

Echo merujuk pada tindakan mereproduksi atau membagikan kembali konten yang sudah ada kepada pengikut atau audiens Anda. Misalnya, jika seseorang memposting tweet atau kiriman di platform sosial lainnya, dan Anda memutuskan untuk membagikan kembali (retweet, share, dll.) konten tersebut kepada pengikut Anda, itu disebut sebagai "echo". Praktik ini memungkinkan untuk menyebarkan informasi lebih luas atau memberi penghargaan pada konten yang dianggap relevan atau menarik.

Komentar, di sisi lain, adalah respons atau tanggapan yang diberikan oleh pengguna terhadap konten yang diposting oleh pengguna lain. Ini bisa berupa pertanyaan, pendapat, dukungan, kritik, atau reaksi lainnya terhadap apa yang telah dibagikan. Komentar memungkinkan untuk berinteraksi dengan konten dan pengguna lainnya, membuka diskusi, atau memberikan umpan balik.

Struktur Dasar PHP

```
<?php

$variable = "value";
echo "menampilkan text dan variable dari $variable";

?>
```

menampilkan text dan variable value

Program pertama

Langkah - Langkah Menghubungkan Apache dengan Text Editor(Web server)

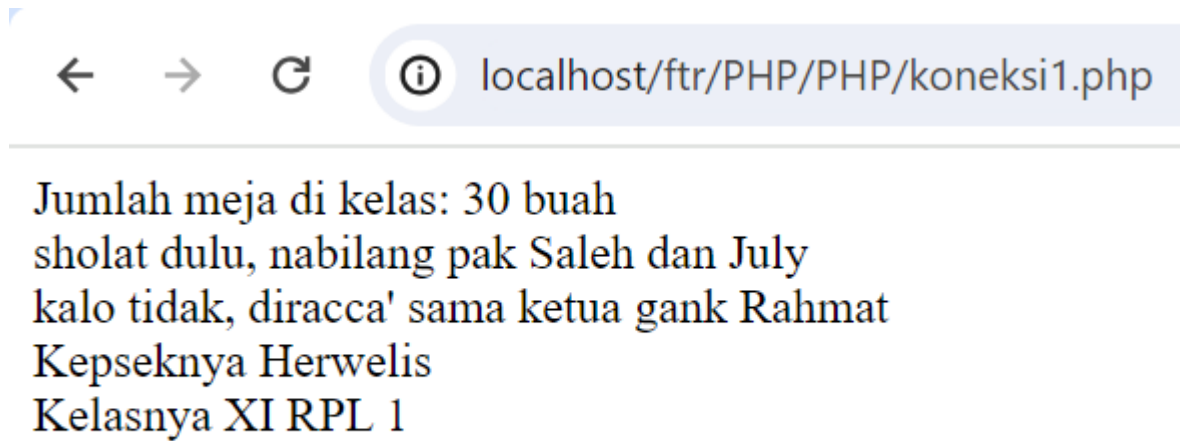
1. Open XAMPP app
2. Start Apache in XAMPP
3. Open Fila Manager
4. Open Drive C
5. Open File XAMPP
6. Open htdocs
7. Create New Folder(Folder_name)
8. Open Text Editor app
9. Open Folder_name
10. Add New File PHP in text editor(File.php)S
11. Done

Kode Program

```
<?php
// Komentar satu baris
/* Komentar
banyak
baris
*/
//Variebel
$meja = 30;
$tk_kelas = "XI";
$ketua_kelas = "July";
$wali_kelas = "Saleh";
$ketua_gank = "Rahmat"; //Pengubahan nilai
//Konstanta
const KEPSEK = "Herwelis";
define('Kelas', 'RPL 1');
/*Kutip satu hanya membaca STRING, variabel dan string
dipisahkan dengan tanda titik*/
echo 'Jumlah meja di kelas: ' . $meja . ' buah';
echo "<br>";
//Kutip dua bisa membaca nilai dari sebuah variabel
    echo "sholat dulu, nabilang pak $wali_kelas dan $ketua_kelas";
echo "<br>";
```

```
//Kutip satu dibaca string disini
echo "kalo tidak, diracca' sama ketua gank $ketua_gank";
echo "<br>";
//Pemanggilan konstanta
echo 'Kepseknya ' . KEPSEK;
echo "<br>";
echo 'Kelasnya ' . $tk_kelas . ' '. Kelas;
```

Hasil



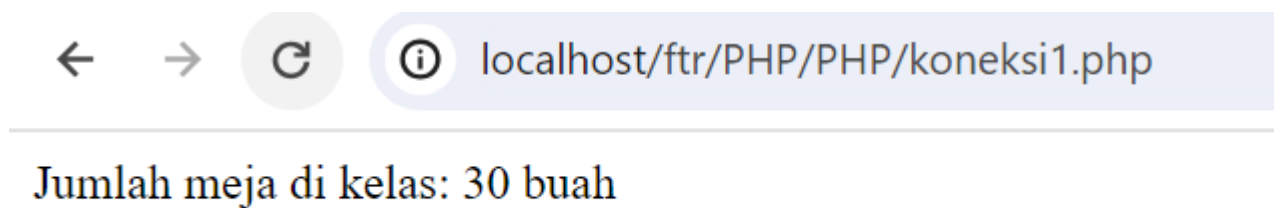
Echo

Kutip satu

Kode Program

```
$meja = 30;
echo 'Jumlah meja di kelas: ' . $meja . ' buah';
```

Hasil



Penjelasan

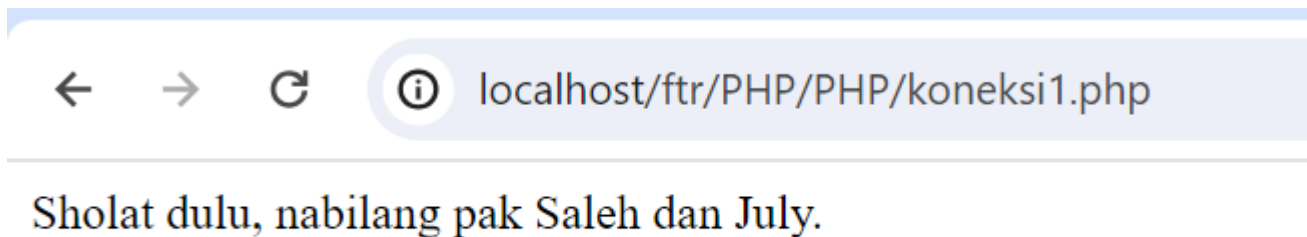
- `$meja = 30;` - Baris ini menginisialisasi sebuah variabel `$meja` dan memberinya nilai 30.
- `echo 'Jumlah meja di kelas: ' . $meja . ' buah';` - Baris ini menampilkan teks yang mencakup nilai dari variabel `$meja`. Operator `.` digunakan untuk menggabungkan string dalam PHP.

kutip dua

Kode Program

```
$ketua_kelas = "July";  
$wali_kelas = "Saleh";  
echo "sholat dulu, nabilang pak $wali_kelas dan $ketua_kelas";
```

Hasil



Penjelasan

- **Inisialisasi Variabel:** Pada baris pertama dan kedua, Anda mendefinisikan dua variabel:
 - `$ketua_kelas` diinisialisasi dengan nilai "July".
 - `$wali_kelas` diinisialisasi dengan nilai "Saleh".
- **Pemanggilan echo:** Pada baris ketiga, Anda menggunakan perintah `echo` untuk mencetak sebuah string ke layar.
- **String Interpolation:** String yang Anda cetak ("sholat dulu, nabilang pak \$wali_kelas dan \$ketua_kelas") menggunakan fitur interpolasi string PHP. Ini berarti nilai dari variabel `$wali_kelas` dan `$ketua_kelas` akan dimasukkan langsung ke dalam string pada saat eksekusi.
 - `$wali_kelas` akan digantikan dengan nilai "Saleh".
 - `$ketua_kelas` akan digantikan dengan nilai "July".
 - Bagian "sholat dulu, nabilang pak " adalah bagian statis dari string.

- Bagian `$wali_kelas` diwakili sebagai "Saleh" karena itu nilai dari variabel `$wali_kelas`.
- Bagian `$ketua_kelas` diwakili sebagai "July" karena itu nilai dari variabel `$ketua_kelas`.
- **Penutupan PHP:** Jangan lupa bahwa kode PHP diakhiri dengan `?>` , meskipun Anda tidak menemukannya dalam contoh yang Anda berikan.

komentar

single line

Kode Program

```
//ini digunakan untuk ketika mau membuat satu baris komentar
```

Penjelasan

Digunakan untuk menambahkan komentar pada satu baris saja. Komentar ini dimulai dengan tanda double slash (`//`) dan berlaku hingga akhir baris.

multiple line

Kode Program

```
/*ini digunakan  
untuk ketika  
mau membuat  
lebih dari  
satu baris komentar*/
```

Penjelasan

Digunakan untuk menambahkan komentar yang lebih dari satu baris. Komentar ini dimulai dengan `/*` dan diakhiri dengan `*/` , dan dapat mencakup beberapa baris kode.

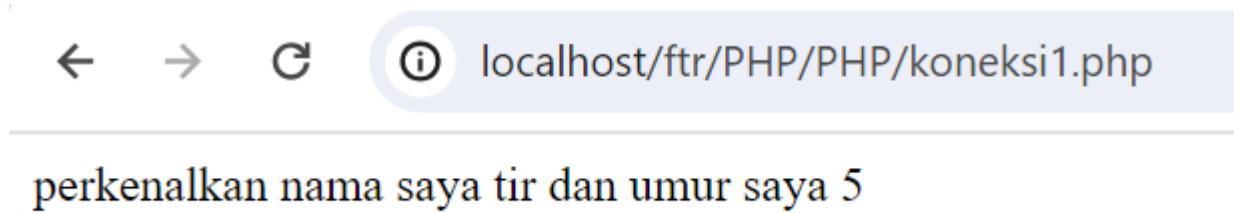
Variabel, Konstanta, operator

Variabel

Kode Program

```
$nama = "tir"; $umur = 5;  
echo "perkenalkan nama saya $nama dan umur saya $umur"
```

Hasil



Penjelasan

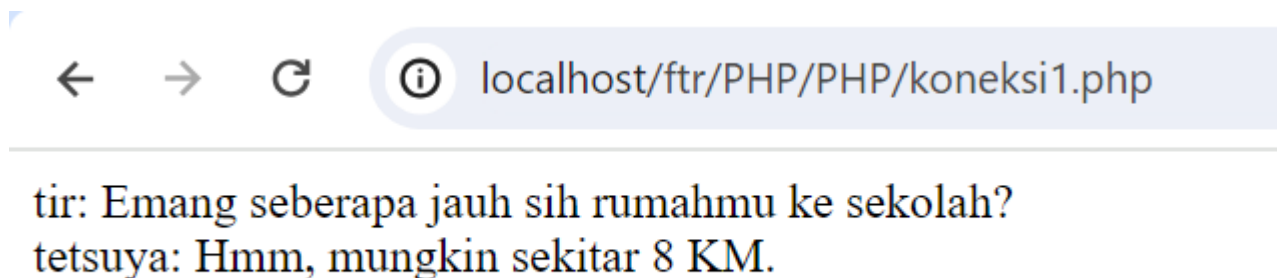
Variabel adalah tempat untuk menyimpan data yang dapat berubah nilainya selama eksekusi program. Anda bisa mendeklarasikan variabel dengan menggunakan tanda dollar (\$) diikuti dengan nama variabelnya.

Constanta

Kode Program

```
const jarak  
define(8, "KM");  
  
echo "tir: Emang Seberapa jauh sih rumahmu ke sekolah"  
echo "tetsuya: Hmm, mungkin Sekitar" jarak
```

Hasil



Penjelasan

Konstanta adalah seperti variabel, tetapi nilainya tidak dapat diubah selama eksekusi program. Anda mendeklarasikan konstanta menggunakan fungsi define()

Operator

Aritmatika

Penjelasan

Operator aritmatika digunakan dalam PHP untuk melakukan operasi matematika pada bilangan. PHP mendukung operator aritmatika standar seperti penjumlahan, pengurangan, perkalian, pembagian, dan modulus.

- Penjumlahan: Menggunakan operator `+` untuk menambahkan dua bilangan.
- Pengurangan: Menggunakan operator `-` untuk mengurangi dua bilangan.
- Perkalian: Menggunakan operator `*` untuk mengalikan dua bilangan.
- Pembagian: Menggunakan operator `/` untuk membagi dua bilangan.
- Modulus: Menggunakan operator `%` untuk mendapatkan sisa dari pembagian dua bilangan.

Program

```
<?php
// Program PHP untuk demonstrasi operator aritmatika

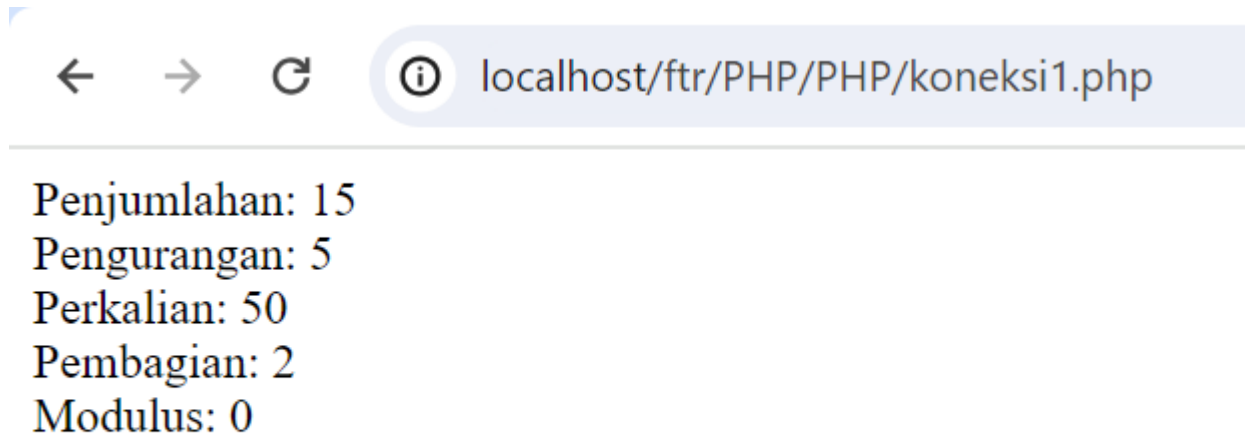
// Deklarasi variabel
$angka1 = 10;
$angka2 = 5;

// Operasi aritmatika
$penjumlahan = $angka1 + $angka2;
$pengurangan = $angka1 - $angka2;
$perkalian = $angka1 * $angka2;
$pembagian = $angka1 / $angka2;
$modulus = $angka1 % $angka2;

// Menampilkan hasil
echo "Penjumlahan: " . $penjumlahan . "<br>";
echo "Pengurangan: " . $pengurangan . "<br>";
echo "Perkalian: " . $perkalian . "<br>";
echo "Pembagian: " . $pembagian . "<br>";
echo "Modulus: " . $modulus . "<br>";
```

?>

Hasil



Analisis

Program di atas mendemonstrasikan penggunaan operator aritmatika dalam PHP untuk melakukan operasi matematika sederhana antara dua bilangan. Setiap operasi aritmatika dihitung dan hasilnya ditampilkan.

Kesimpulan

operator aritmatika di PHP adalah alat yang kuat untuk melakukan operasi matematika dan merupakan bagian penting dari pengembangan web dengan PHP.

Perbandingan

Penjelasan

Operator perbandingan digunakan untuk membandingkan dua nilai dan menghasilkan hasil boolean (true atau false) berdasarkan hasil perbandingan tersebut. PHP mendukung operator perbandingan standar seperti `<` (kurang dari), `>` (lebih dari), `≤` (kurang dari atau sama dengan), `≥` (lebih dari atau sama dengan), `==` (sama dengan), dan `≠` (tidak sama dengan).

Program

```
<?php
// Program PHP untuk demonstrasi operator perbandingan
```



```
// Deklarasi variabel
$angka1 = 10;
$angka2 = 5;

// Operasi perbandingan
$kurang_dari = $angka1 < $angka2;
$lebih_dari = $angka1 > $angka2;
$kurang_dari_sama_dengan = $angka1 ≤ $angka2;
$lebih_dari_sama_dengan = $angka1 ≥ $angka2;
$sama_dengan = $angka1 == $angka2;
$tidak_sama_dengan = $angka1 ≠ $angka2;

// Menampilkan hasil
echo "Apakah angka1 kurang dari angka2? " . ($kurang_dari ? "Ya" : "Tidak") .
"<br>";
echo "Apakah angka1 lebih dari angka2? " . ($lebih_dari ? "Ya" : "Tidak") . "
<br>";
echo "Apakah angka1 kurang dari atau sama dengan angka2? " .
($kurang_dari_sama_dengan ? "Ya" : "Tidak") . "<br>";
echo "Apakah angka1 lebih dari atau sama dengan angka2? " .
($lebih_dari_sama_dengan ? "Ya" : "Tidak") . "<br>";
echo "Apakah angka1 sama dengan angka2? " . ($sama_dengan ? "Ya" : "Tidak") .
"<br>";
echo "Apakah angka1 tidak sama dengan angka2? " . ($tidak_sama_dengan ? "Ya" :
"Tidak") . "<br>";
?>
```

Hasil

← → ↻ ⓘ localhost/fttr/PHP/PHP/koneksi1.php

Apakah angka1 kurang dari angka2? Tidak
 Apakah angka1 lebih dari angka2? Ya
 Apakah angka1 kurang dari atau sama dengan angka2? Tidak
 Apakah angka1 lebih dari atau sama dengan angka2? Ya
 Apakah angka1 sama dengan angka2? Tidak
 Apakah angka1 tidak sama dengan angka2? Ya

Analisis

Program di atas mengilustrasikan penggunaan operator perbandingan dalam PHP untuk membandingkan dua nilai numerik. Setiap operasi perbandingan dievaluasi dan hasilnya ditampilkan dalam bentuk teks (Ya atau Tidak) berdasarkan hasil perbandingan.

Kesimpulan

Dengan menggunakan operator perbandingan di PHP, kita dapat membandingkan dua nilai dengan mudah dan menghasilkan hasil boolean berdasarkan hasil perbandingan tersebut. Ini memungkinkan pengembang untuk membuat keputusan dalam kode berdasarkan hubungan antara nilai-nilai tersebut.

Logika

Penjelasan

Operator logika digunakan untuk menggabungkan beberapa kondisi boolean dan menghasilkan nilai boolean baru. PHP mendukung operator logika standar seperti `&&` (AND logika), `||` (OR logika), dan `!` (NOT logika).

- AND Logika (`&&`): Menghasilkan `true` jika kedua kondisi adalah `true`.
- OR Logika (`||`): Menghasilkan `true` jika salah satu kondisi atau kedua kondisi adalah `true`.
- NOT Logika (`!`): Menghasilkan kebalikan nilai dari kondisi yang diberikan.

Program

```
<?php
// Program PHP untuk demonstrasi operator logika

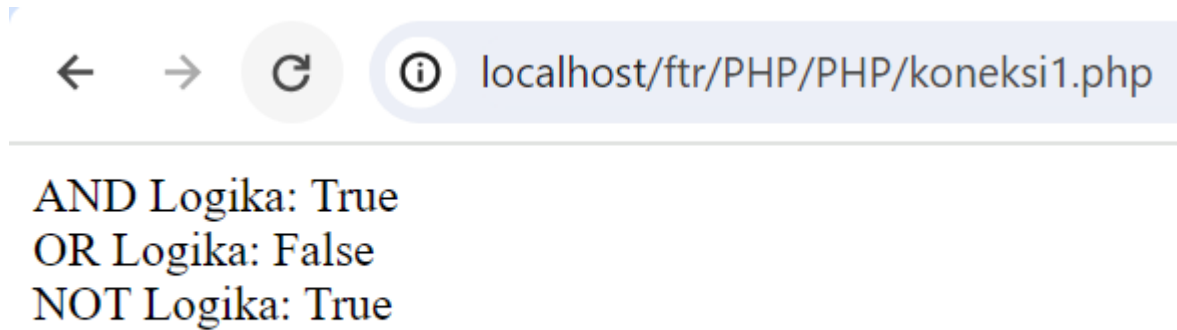
// Deklarasi variabel
$nilai1 = 10;
$nilai2 = 5;

// Operasi logika
$and = ($nilai1 > 5) && ($nilai2 < 10);
$or = ($nilai1 < 5) || ($nilai2 > 10);
$not = !($nilai1 == $nilai2);

// Menampilkan hasil
echo "AND Logika: " . ($and ? "True" : "False") . "<br>";
echo "OR Logika: " . ($or ? "True" : "False") . "<br>";
echo "NOT Logika: " . ($not ? "True" : "False") . "<br>";
```

?>

Hasil



Analisis

Program di atas menggambarkan penggunaan operator logika dalam PHP untuk mengevaluasi beberapa kondisi boolean dan menghasilkan hasil logika berdasarkan operasi yang dilakukan. Setiap operasi logika dievaluasi dan hasilnya ditampilkan dalam bentuk teks (True atau False) berdasarkan hasil logika.

Kesimpulan

operator logika di PHP adalah alat yang penting dalam pengembangan web untuk membuat keputusan yang bergantung pada kondisi-kondisi yang diberikan.

Conditional Statement

IF

Penjelasan

Kondisional statement **IF** digunakan untuk melakukan pemeriksaan kondisi tertentu. Jika kondisi tersebut benar (true), maka blok kode di dalam IF akan dieksekusi. Jika kondisi salah (false), maka blok kode tersebut tidak dieksekusi.

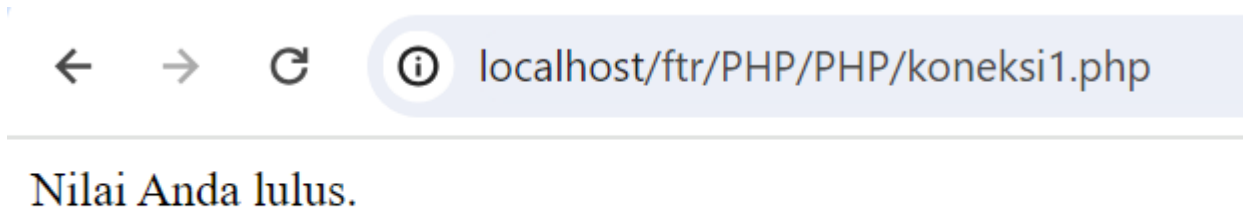
Struktur Program

```
if (kondisi) {  
    // blok kode yang akan dieksekusi jika kondisi benar  
}
```

Kode Program

```
$nilai = 80;
if ($nilai ≥ 70) {
    echo "Nilai Anda lulus.";
}
```

Hasil



Analisis

- Baris 1: Sebuah variabel `$nilai` diinisialisasi dengan nilai 80.
- Baris 2: Kondisi dievaluasi, yaitu apakah nilai `$nilai` lebih besar atau sama dengan 70.
- Baris 3: Karena kondisi benar (true), maka blok kode di dalam IF dieksekusi.
- Baris 4: Mencetak teks "Nilai Anda lulus." karena nilai `$nilai` memenuhi kondisi.

Kesimpulan Program

Program akan mencetak "Nilai Anda lulus." karena nilai `$nilai` adalah 80, yang lebih besar dari atau sama dengan 70.

Else IF

Penjelasan

Kondisional statement `Else IF` digunakan untuk menambahkan kondisi tambahan setelah kondisi IF. Jika kondisi IF tidak terpenuhi (false), maka kondisi Else IF akan dicek. Jika kondisi Else IF benar (true), maka blok kode di dalam Else IF akan dieksekusi.

Struktur program

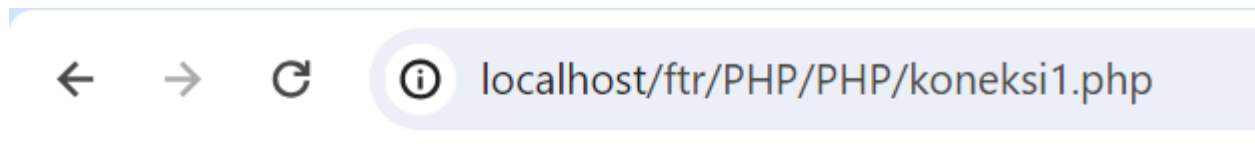
```
if (kondisi1) {
    // blok kode yang akan dieksekusi jika kondisi1 benar
} else if (kondisi2) {
```

```
// blok kode yang akan dieksekusi jika kondisi2 benar  
}
```

Kode Program

```
$nilai = 55;  
if ($nilai ≥ 70) {  
    echo "Nilai Anda lulus."  
} else if ($nilai ≥ 60) {  
    echo "Anda dapat remedial."  
} else {  
    echo "Anda tidak lulus."  
}
```

Hasil



Analisis

- Baris 1: Sebuah variabel `$nilai` diinisialisasi dengan nilai 55.
- Baris 2: Kondisi pertama dievaluasi, yaitu apakah nilai `$nilai` lebih besar atau sama dengan 70. Kondisi ini tidak terpenuhi (false).
- Baris 3: Kondisi kedua dievaluasi, yaitu apakah nilai `$nilai` lebih besar atau sama dengan 60. Kondisi ini juga tidak terpenuhi (false).
- Baris 4: Karena kedua kondisi sebelumnya tidak terpenuhi, maka blok kode di dalam ELSE dieksekusi.
- Baris 5: Mencetak teks "Anda tidak lulus." karena nilai `$nilai` kurang dari 60.

Kesimpulan Program

Program akan mencetak "Anda tidak lulus." karena nilai `$nilai` adalah 55, yang kurang dari 60.

Else

Penjelasan

Kondisional statement `Else` digunakan untuk mengeksekusi blok kode tertentu jika kondisi IF sebelumnya tidak terpenuhi (false).

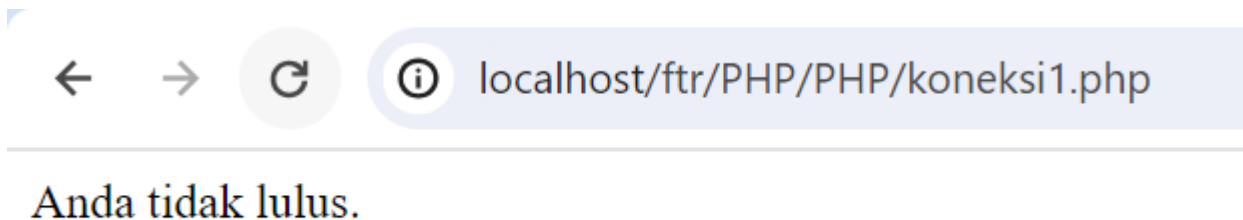
Struktur program

```
if (kondisi){  
    // blok kode yang akan dieksekusi jika kondisi benar  
} else {  
    // blok kode yang akan dieksekusi jika kondisi salah  
}
```

Kode Program

```
$nilai = 55;  
if ($nilai ≥ 60)  
{  
    echo "Nilai Anda lulus.";  
} else {  
    echo "Anda tidak lulus.";  
}
```

Hasil



Analisis

- Baris 1: Sebuah variabel `$nilai` diinisialisasi dengan nilai 55.
- Baris 2: Kondisi dievaluasi, yaitu apakah nilai `$nilai` lebih besar atau sama dengan 60. Kondisi ini tidak terpenuhi (false).
- Baris 3: Karena kondisi sebelumnya tidak terpenuhi, maka blok kode di dalam ELSE dieksekusi.

- Baris 4: Mencetak teks "Anda tidak lulus." karena nilai `$nilai` kurang dari 60.

Kesimpulan Program

Program akan mencetak "Anda tidak lulus." karena nilai `$nilai` adalah 55, yang kurang dari 60.

if-else-if-else

Penjelasan

Pernyataan if-else-if-else digunakan ketika terdapat beberapa kondisi yang harus diuji secara berurutan.

Struktur

```
if (kondisi1) {  
    // Blok kode yang dijalankan jika kondisi1 benar  
} elseif (kondisi2) {  
    // Blok kode yang dijalankan jika kondisi2 benar  
} else {  
    // Blok kode yang dijalankan jika semua kondisi salah  
}
```

Kode Program

```
$nilai = 75;  
  
if ($nilai ≥ 80) {  
    echo "Selamat, Anda mendapatkan nilai A!";  
} elseif ($nilai ≥ 70) {  
    echo "Anda mendapatkan nilai B.";  
} elseif ($nilai ≥ 60) {  
    echo "Anda mendapatkan nilai C.";  
} else {  
    echo
```

Hasil



localhost/ftr/PHP/PHP/koneksi1.php

Anda mendapatkan nilai B.

Analisis:

Variabel \$nilai diberikan nilai 75.

Pernyataan if pertama digunakan untuk menguji apakah nilai \$nilai lebih besar atau sama dengan 80.

Karena nilai \$nilai tidak mencapai 80, maka blok kode di dalam if pertama tidak akan dieksekusi.

Pernyataan elseif pertama digunakan untuk menguji apakah nilai \$nilai lebih besar atau sama dengan 70.

Karena nilai \$nilai adalah 75 dan memenuhi kondisi \$nilai >= 70, blok kode di dalam elseif pertama akan dieksekusi.

Blok kode di dalam elseif pertama mencetak pesan "Anda mendapatkan nilai B." menggunakan pernyataan echo.

Setelah blok kode di dalam elseif pertama dieksekusi, program akan keluar dari struktur kondisional dan berakhir.

Kesimpulan:

Jika variabel \$nilai memiliki nilai 75, maka kondisi \$nilai >= 80 akan salah, namun kondisi \$nilai >= 70 akan benar. Oleh karena itu, pesan "Anda mendapatkan nilai B." akan ditampilkan

Switch Case

Penjelasan

Kondisional statement `Switch Case` digunakan untuk membandingkan nilai ekspresi dengan beberapa nilai yang mungkin. Jika nilai ekspresi cocok dengan salah satu nilai `case`, maka blok kode di dalam `case` tersebut akan dieksekusi.

Struktur program

```
switch (ekspresi) {  
  case nilai1:  
    // blok kode yang akan dieksekusi jika ekspresi sama dengan nilai1  
    break;  
  case nilai2:
```

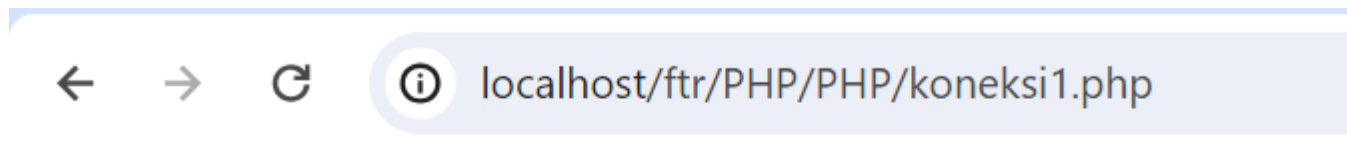


```
// blok kode yang akan dieksekusi jika ekspresi sama dengan nilai2
break;
default:
// blok kode yang akan dieksekusi jika tidak ada case yang cocok
}
```

Kode Program

```
$nilai = 'B'; switch ($nilai) {
case 'A':
echo "Anda sangat baik.";
break;
case 'B':
echo "Anda baik.";
break;
case 'C':
echo "Anda cukup.";
break;
default:
echo "Anda tidak lulus.";
}
```

Hasil



Anda baik.

Analisis

- Baris 1: Sebuah variabel `$nilai` diinisialisasi dengan nilai 'B'.
- Baris 2: Nilai `$nilai` dibandingkan dengan nilai-nilai case di dalam switch.
- Baris 3-5: Karena nilai `$nilai` sama dengan 'B', maka blok kode di dalam case 'B' dieksekusi.
- Baris 6: Mencetak teks "Anda baik." karena nilai `$nilai` sama dengan 'B'.

Kesimpulan Program

Program akan mencetak "Anda baik." karena nilai `$nilai` adalah 'B', yang cocok dengan case 'B'.

Array

Array 1 Dimensi

Penjelasan

Array 1 dimensi adalah struktur data yang dapat menyimpan beberapa nilai dalam satu variabel. Setiap nilai dalam array memiliki indeks numerik yang dimulai dari 0.

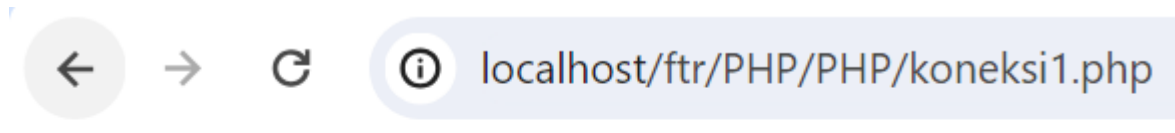
Struktur program

```
$nama_array = array(nilai1, nilai2, nilai3, ... );
```

Kode Program

```
$buah = array(
    "Apel", "Jeruk", "Mangga", "Anggur"
);
echo "Saya suka makan " . $buah[0];
```

Hasil



Analisis

- Baris 1: Membuat array `$buah` dengan nilai "Apel", "Jeruk", "Mangga", dan "Anggur".
- Baris 2: Mencetak teks "Saya suka makan Apel" karena nilai dengan indeks 0 dalam array adalah "Apel".

Kesimpulan Program

Program akan mencetak "Saya suka makan Apel" karena nilai dengan indeks 0 dalam array `$buah` adalah "Apel".

Array Asosiatif

Penjelasan

Array asosiatif adalah struktur data yang menggunakan nama kunci (key) untuk setiap nilai dalam array. Ini memungkinkan kita untuk mengakses nilai berdasarkan kunci yang ditetapkan.

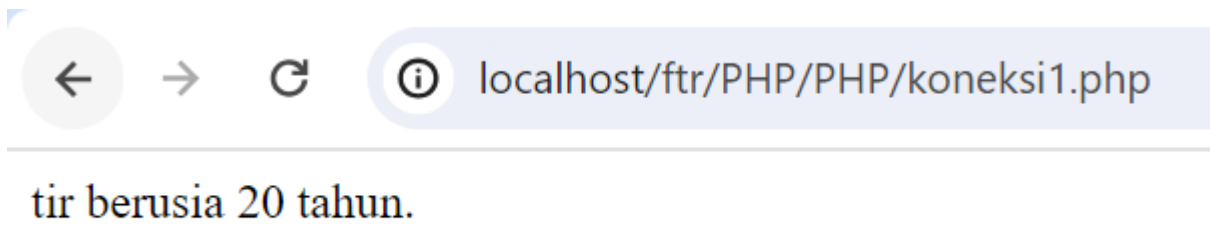
Struktur program

```
$nama_array = array("kunci1" => nilai1, "kunci2" => nilai2, "kunci3" => nilai3, ... );
```

Kode Program

```
$siswa = array("nama" => "tir", "umur" => 20, "kelas" => "XII");  
echo $siswa["nama"] . " berusia " . $siswa["umur"] . " tahun.";
```

Hasil



Analisis

- Baris 1: Membuat array `$siswa` dengan kunci "nama", "umur", dan "kelas".
- Baris 2: Menggunakan kunci "nama" dan "umur" untuk mengakses nilai dalam array `$siswa`.
- Baris 2: Mencetak teks "John berusia 20 tahun."

Kesimpulan Program

Program akan mencetak "John berusia 20 tahun." karena nilai dengan kunci "nama" adalah "tir" dan nilai dengan kunci "umur" adalah 20 dalam array `$siswa`.

Array Multidimensi

Penjelasan

Array multidimensi adalah array yang memiliki satu atau lebih array di dalamnya. Hal ini memungkinkan kita untuk membuat struktur data yang kompleks, seperti matriks.

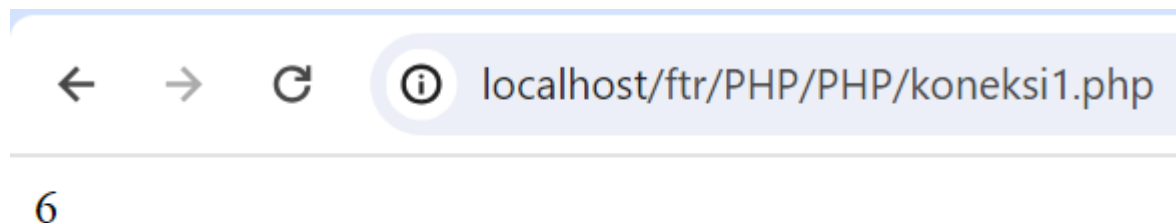
Struktur program

```
$nama_array = array(  
    array(nilai1, nilai2, nilai3, ...),  
    array(nilai1, nilai2, nilai3, ...),  
    ... );
```

Kode Program

```
$matriks = array(  
    array(1, 2, 3),  
    array(4, 5, 6),  
    array(7, 8, 9) );  
echo $matriks[1][2]; // Mengakses nilai 6
```

Hasil



Analisis

- Baris 1-3: Membuat array `$matriks` yang berisi array-array dengan nilai tertentu.
- Baris 4: Menggunakan indeks 1 dan 2 untuk mengakses nilai 6 dalam array `$matriks`.
- Baris 4: Mencetak nilai 6.

Kesimpulan Program

Program akan mencetak nilai 6 karena nilai tersebut berada di indeks 1 dan 2 dalam array `$matriks`.

Var_Dump

Penjelasan

ungsi `var_dump()` di PHP. Fungsi ini digunakan untuk menampilkan informasi rinci tentang tipe dan nilai dari satu atau lebih variabel, termasuk struktur dan ukuran dari array dan objek.

Struktur

1. **Penggunaan `var_dump()`** : Memanggil fungsi `var_dump()` untuk menampilkan informasi tentang satu atau lebih variabel.
2. **Output**: Menampilkan informasi rinci tentang tipe dan nilai dari variabel yang diberikan.

Program

```
<?php
// Variabel-variabel contoh
$name = "John Doe";
$umur = 30;
$tinggi = 175.5;
$hobi = array("Membaca", "Bersepeda", "Memasak");

// Menampilkan informasi tentang variabel menggunakan var_dump()
echo "Informasi tentang variabel \ $name:\n";
var_dump($name);

echo "\nInformasi tentang variabel \ $umur:\n";
var_dump($umur);

echo "\nInformasi tentang variabel \ $tinggi:\n";
var_dump($tinggi);

echo "\nInformasi tentang variabel \ $hobi:\n";
var_dump($hobi);
?>
```

Hasil

Analisis

1. Program memanggil fungsi `var_dump()` untuk menampilkan informasi rinci tentang beberapa variabel yang berbeda.
2. Setiap variabel dipass sebagai argumen ke fungsi `var_dump()` , dan informasi rinci tentang tipe dan nilai dari variabel tersebut ditampilkan sebagai output.
3. Output dari `var_dump()` juga akan menunjukkan struktur dan ukuran dari array dan objek, jika variabel yang diberikan merupakan array atau objek.

Kesimpulan

Fungsi `var_dump()` sangat berguna untuk debugging dan pemahaman mendalam tentang variabel dalam kode PHP. Dengan menggunakan `var_dump()` , kita dapat melihat tipe data dari variabel, nilai-nilai yang disimpan di dalamnya, serta struktur dan ukuran dari array dan objek. Ini membantu kita dalam mengidentifikasi masalah dan memahami bagaimana data disimpan dan diakses dalam program kita. Dalam konteks debugging, `var_dump()` adalah alat yang sangat berguna untuk menganalisis variabel dengan lebih mendalam.

Looping (Perulangan)

FOR

Penjelasan

penggunaan struktur perulangan `for` di PHP. Perulangan `for` digunakan untuk mengeksekusi blok kode berulang kali selama kondisi tertentu terpenuhi.

Struktur

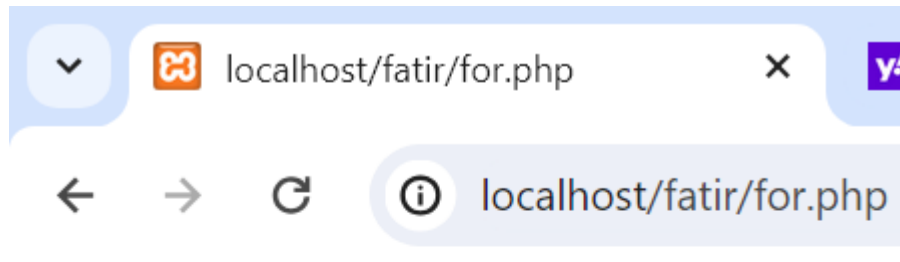
```
for (inisialisasi; kondisi; perubahan) {  
    // blok kode yang akan diulang  
}
```

Program

```
<?php  
echo "Menampilkan angka 1 sampai 5: <br>";
```

```
for ($i = 1; $i ≤ 5; $i++) {  
    echo $i . "<br>";  
}  
?>
```

Hasil



Menampilkan angka 1 sampai 5:

1
2
3
4
5

Analisis

1. Program menggunakan perulangan `for` untuk menampilkan angka dari 1 sampai 5.
2. Variabel `$i` digunakan sebagai variabel kontrol perulangan yang diinisialisasi dengan nilai 1.
3. Perulangan akan terus berlangsung selama nilai `$i` kurang dari atau sama dengan 5.
4. Setiap iterasi, nilai `$i` akan ditampilkan, kemudian nilai `$i` akan bertambah satu setelah setiap iterasi.

Kesimpulan

Perulangan `for` adalah alat yang kuat dalam pemrograman untuk mengeksekusi serangkaian instruksi berulang kali.

While

Penjelasan

penggunaan struktur perulangan `while` di PHP. Perulangan `while` digunakan untuk mengeksekusi blok kode selama kondisi yang diberikan benar.

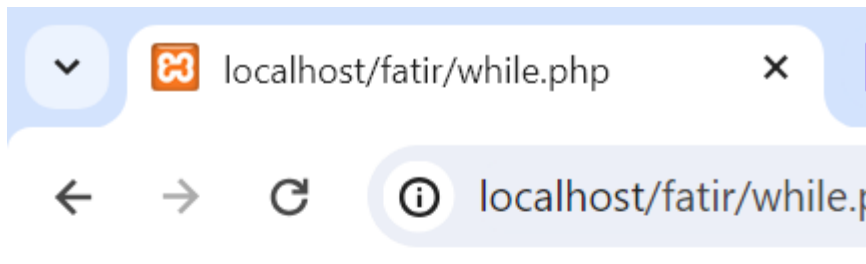
Struktur

```
while (kondisi) {  
    // blok kode yang akan diulang  
    // perubahan kondisi harus dilakukan di dalam blok kode  
}
```

Program

```
<?php  
  
$counter = 1;  
  
echo "Menampilkan angka 1 sampai 10: <br>";  
while ($counter ≤ 10) {  
    echo $counter . "<br>";  
    $counter++;  
}  
?>
```

Hasil



Menampilkan angka 1 sampai 10:

1
2
3
4
5
6
7
8
9
10

Analisis

1. Program menggunakan perulangan `while` untuk menampilkan angka dari 1 sampai 10.
2. Variabel `$counter` diinisialisasi dengan nilai 1 di luar perulangan.
3. Perulangan akan terus berlangsung selama nilai `$counter` kurang dari atau sama dengan 10.
4. Setiap iterasi, nilai `$counter` akan ditampilkan, kemudian nilai `$counter` akan ditambah satu untuk menghindari perulangan tak terbatas.

Kesimpulan

Perulangan `while` adalah cara yang efektif untuk mengeksekusi serangkaian instruksi selama kondisi tertentu terpenuhi

Do While

Penjelasan

penggunaan struktur perulangan `do-while` di PHP. Perulangan `do-while` mirip dengan perulangan `while`, namun blok kode di dalamnya akan dieksekusi setidaknya satu kali, bahkan jika kondisi awalnya salah.

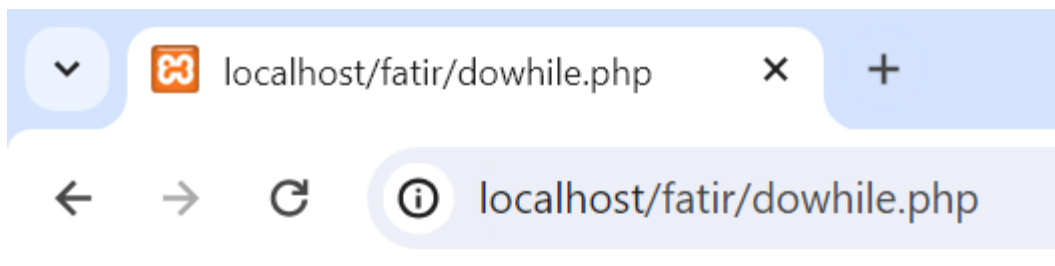
Struktur

```
do {  
    // blok kode yang akan diulang  
    // perubahan kondisi harus dilakukan di dalam blok kode  
} while (kondisi);
```

Program

```
<?php  
  
$counter = 1;  
  
echo "Menampilkan angka 1 sampai 15: <br>";  
do {  
    echo $counter . "<br>";  
    $counter++;  
} while ($counter ≤ 15);  
?>
```

Hasil



Menampilkan angka 1 sampai 15:

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15

Analisis

1. Program menggunakan perulangan `do-while` untuk menampilkan angka dari 1 sampai 5.
2. Variabel `$counter` diinisialisasi dengan nilai 1 sebelum perulangan dimulai.
3. Blok kode di dalam perulangan akan dieksekusi setidaknya satu kali sebelum kondisi di dalam `while` dievaluasi.
4. Setelah setiap iterasi, nilai `$counter` akan diperbarui dan kondisi akan dievaluasi kembali. Perulangan akan berlanjut selama kondisi yang ditentukan benar.

Kesimpulan

Perulangan `do-while` berguna ketika kita ingin menjalankan suatu blok kode setidaknya sekali, bahkan jika kondisi awalnya salah.

Foreach

Penjelasan

penggunaan struktur perulangan `foreach` di PHP. Perulangan `foreach` digunakan untuk mengulang setiap elemen dalam array.

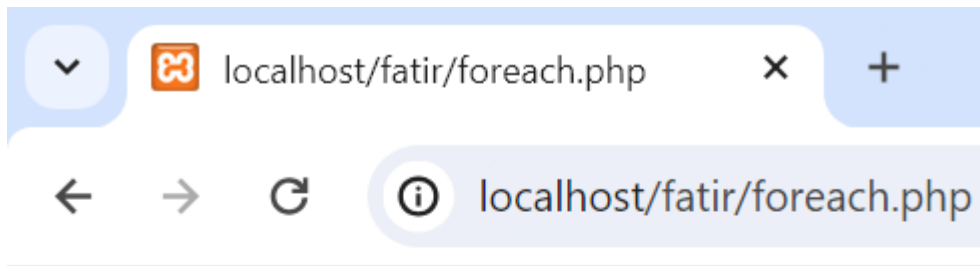
Struktur

```
foreach ($array as $nilai) {  
    // blok kode yang akan diulang  
}
```

Program

```
<?php  
  
$buah = array("Apel", "Pisang", "Jeruk", "Mangga", "Anggur");  
  
echo "Menampilkan nama-nama buah: <br>";  
foreach ($buah as $nama_buah) {  
    echo $nama_buah . "<br>";  
}  
?>
```

Hasil



Menampilkan nama-nama buah:

Apel

Pisang

Jeruk

Mangga

Anggur

Analisis

1. Program menggunakan perulangan `foreach` untuk mengulang setiap elemen dalam array `$buah`.
2. Variabel `$nama_buah` digunakan untuk menyimpan nilai setiap elemen dalam setiap iterasi perulangan.
3. Blok kode di dalam perulangan akan dieksekusi sekali untuk setiap elemen dalam array, dan nilai elemen tersebut akan ditampilkan.

Kesimpulan

Perulangan `foreach` adalah alat yang sangat berguna untuk mengulang setiap elemen dalam array tanpa perlu mengkhawatirkan tentang indeks atau panjang array.

Function

Penjelasan

Program ini adalah contoh sederhana dari penggunaan fungsi di PHP. Fungsi digunakan untuk mengelompokkan serangkaian pernyataan ke dalam blok tunggal, sehingga dapat digunakan kembali dan dipanggil berkali-kali di berbagai bagian program.

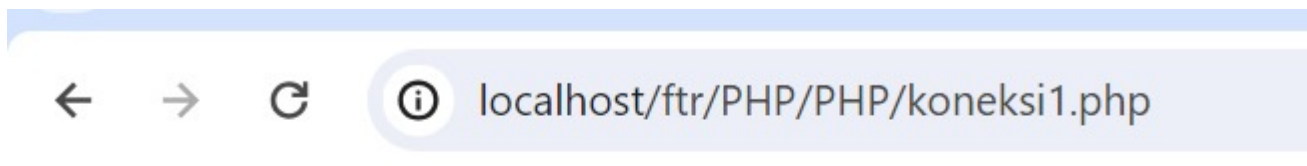
Struktur

```
function nama_function(parameter1, parameter2, ... ) {  
    // Blok kode yang dijalankan ketika function dipanggil  
    // ...  
    return nilai;  
}
```

Program

```
<?php  
// Deklarasi fungsi untuk menghitung luas segitiga  
function hitungLuasSegitiga($alas, $tinggi) {  
    $luas = 0.5 * $alas * $tinggi;  
    return $luas;  
}  
  
// Memanggil fungsi dan menampilkan hasilnya  
$luas_segitiga = hitungLuasSegitiga(6, 8);  
echo "Luas segitiga: " . $luas_segitiga;  
?>
```

Hasil



Analisis

1. Program mendefinisikan fungsi `hitungLuasSegitiga()` yang menerima dua parameter, yaitu alas dan tinggi segitiga.
2. Di dalam fungsi, luas segitiga dihitung menggunakan formula matematika yang tepat, dan kemudian nilai luas tersebut dikembalikan menggunakan pernyataan `return`.
3. Fungsi dipanggil dengan menyediakan argumen sesuai dengan parameter yang diterima, dan hasilnya ditampung dalam variabel `$luas_segitiga`.
4. Hasil perhitungan luas segitiga kemudian ditampilkan menggunakan pernyataan `echo`.

Kesimpulan

Fungsi adalah alat yang sangat berguna dalam pemrograman PHP untuk mengorganisir dan mengelompokkan kode ke dalam unit yang dapat digunakan kembali.

PHP From

Get Method

Penjelasan

Metode GET adalah salah satu metode pengiriman data dari client ke server dalam protokol HTTP. Dalam PHP, kita dapat menggunakan metode GET untuk mengirim data melalui URL.

Struktur

```
<?php

// Dapatkan nilai parameter
$q = $_GET['q'];

// Lakukan sesuatu dengan data yang dikirimkan
echo "Anda mencari: $q<br>";

?>
```

Code

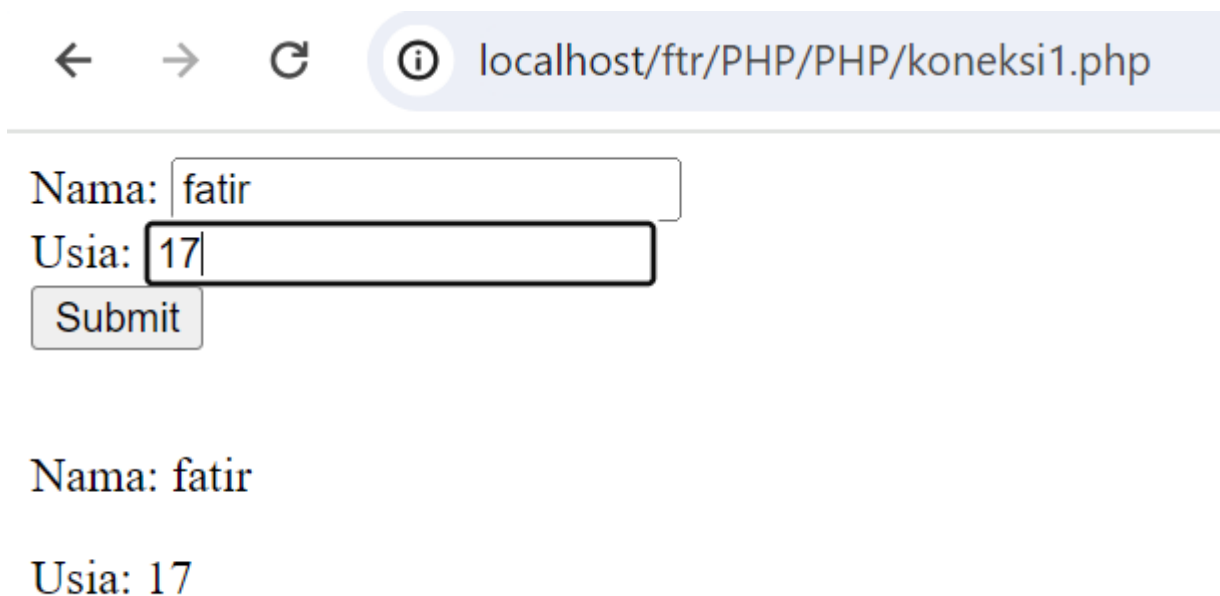
```
<!DOCTYPE html>
<html>
<head>
    <title>Form GET</title>
</head>
<body>
    <form method="GET" action="test2.php">
        <label for="name">Nama:</label>
        <input type="text" name="name" id="name">
        <br>
        <label for="age">Usia:</label>
        <input type="text" name="age" id="age">
        <br>
        <input type="submit" value="Submit">
    </form>
```

```
</body>
</html>
```

```
<?php
if (isset($_GET['name']) && isset($_GET['age'])) {
    $name = $_GET['name'];
    $age = $_GET['age'];

    echo "Nama: " . $name . "<br>";
    echo "Usia: " . $age;
}
?>
```

Hasil



The screenshot shows a web browser window with the address bar displaying 'localhost/fttr/PHP/PHP/koneksi1.php'. Below the address bar, there is a form with two input fields: 'Nama: fatir' and 'Usia: 17'. A 'Submit' button is located below the 'Usia' field. Below the form, the output of the PHP script is displayed: 'Nama: fatir' followed by a line break, and 'Usia: 17'.

Aanlisis:

Bagian HTML:

- Program HTML ini menampilkan sebuah form dengan metode GET yang mengirimkan data ke file "test2.php" saat tombol "Submit" ditekan.
- Form memiliki dua input field yaitu "Nama" dan "Usia" yang masing-masing memiliki atribut name yang sesuai.
- Ketika tombol "Submit" ditekan, data yang diisi oleh pengguna akan dikirimkan ke file "test2.php" untuk diproses.

Bagian PHP:

- Program PHP ini berada di bawah tag penutup HTML (=) dan akan dieksekusi saat form dikirimkan.</li- Program menggunakan isset() untuk memeriksa apakah parameter name dan age telah dikirim melalui metode GET.
- Jika parameter tersebut ada dalam URL, program akan mengambil nilai-nilainya menggunakan \$_GET['name'] dan \$_GET['age'].
- Selanjutnya, program akan menampilkan nilai-nilai tersebut dengan menggunakan echo untuk mencetak pesan "Nama: [nama]" dan "Usia: [usia]" di halaman.

Kesimpulan:

Program tersebut adalah contoh sederhana yang menggunakan metode GET dalam PHP untuk mengambil data yang dikirim melalui form HTML. Ketika form dikirimkan, data yang diisi oleh pengguna akan dikirim melalui URL sebagai parameter dan nilai yang dapat diakses melalui \$_GET di file "test2.php". Program kemudian mengambil nilai-nilai tersebut dan menampilkannya di halaman dengan menggunakan echo. Pastikan file "test2.php" tersedia dan dapat diakses dengan benar serta server web Anda telah dikonfigurasi untuk menjalankan PHP. Setelah form dikirimkan, Anda akan melihat pesan "Nama: [nama]" dan "Usia: [usia]" di halaman "test2.php" sesuai dengan data yang diisi oleh pengguna.

Post Method

Penjelasan:

Metode POST digunakan untuk mengirimkan data dari client ke server. Data dikirimkan dalam tubuh permintaan HTTP dan tidak terlihat dalam URL seperti metode GET. Metode POST lebih aman untuk mengirim data sensitif seperti kata sandi. Dalam PHP, data yang dikirim melalui metode POST dapat diakses menggunakan variabel global \$_POST.

Struktur:

```
<?php

// Periksa apakah formulir telah dikirimkan
if ($_SERVER['REQUEST_METHOD'] === 'POST') {

    // Dapatkan nilai field formulir
    $name = $_POST['name'];
    $email = $_POST['email'];

    // Lakukan sesuatu dengan data yang dikirimkan
    echo "Nama: $name<br>";
```

```
    echo "Email: $email<br>";  
}  
  
?>
```

Code

```
<html lang="en">  
<head>  
    <title>Document</title>  
</head>  
<body>  
    <!-- Pada atribut action, kalian tuliskan nama file php yang bertugas  
    untuk mengelola atau menangkap data dari form tersebut. -->  
    <form action="proses_post.php" method="POST">  
        <input type="text" name="nama_lengkap" placeholder="Masukkan nama">  
        <input type="number" name="umur" placeholder="Masukkan umur">  
        <input type="password" name="password" placeholder="Masukkan  
password"><br>  
        <button type="submit">Kirim</button>  
    </form>  
</body>  
</html>
```

<?php

```
// Key dari array-nya, sesuai dengan nama dari atribut name di setiap input-  
nya
```

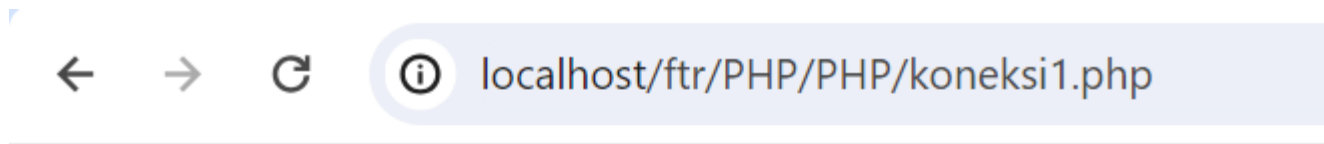
```
$nama = $_POST["nama"];  
$umur = $_POST["umur"];
```

```
var_dump($_POST);  
?>
```

```
<!DOCTYPE html>  
<html lang="en">  
<head>  
    <title> XI RPL 1 - POST</title>  
</head>  
<body>  
    <p>Nama anda <?= $_POST["nama_lengkap"] ?></p>  
    <p>Umur anda <?= $umur ?> tahun</p>  
    <p>Password anda aman!</p>
```

```
</body>  
</html>
```

Hasil



Formulir Data

Nama:

Umur:

Password:

Nama: fatir

Umur: 17 tahun

Password anda aman!

Anlisis:

Pada bagian HTML:

- Formulir menggunakan metode POST (`method="POST"`), yang cocok untuk mengirim data sensitif seperti password.
- Atribut `action="proses_post.php"` menunjukkan bahwa data dari formulir akan dikirim ke skrip PHP bernama `proses_post.php` untuk diproses.

- Setiap input memiliki atribut `name` yang akan digunakan untuk mengidentifikasi nilai yang dikirimkan ke skrip PHP.
- Input untuk `nama_lengkap` (tipe `text`), `umur` (tipe `number`), dan `password` (tipe `password`) disertakan, dengan masing-masing menyertakan placeholder untuk memberi petunjuk kepada pengguna.

Pada bagian PHP

- `$umur = $_POST["umur"];` : Mengambil nilai dari input dengan nama `umur`.
- `htmlspecialchars($nama_lengkap)` : Digunakan untuk menghindari serangan XSS (Cross-Site Scripting) dengan mengonversi karakter khusus HTML menjadi entitas HTML.
- Pesan "Password anda aman!" ditampilkan secara statis untuk memberikan umpan balik yang aman kepada pengguna.

Kesimpulan

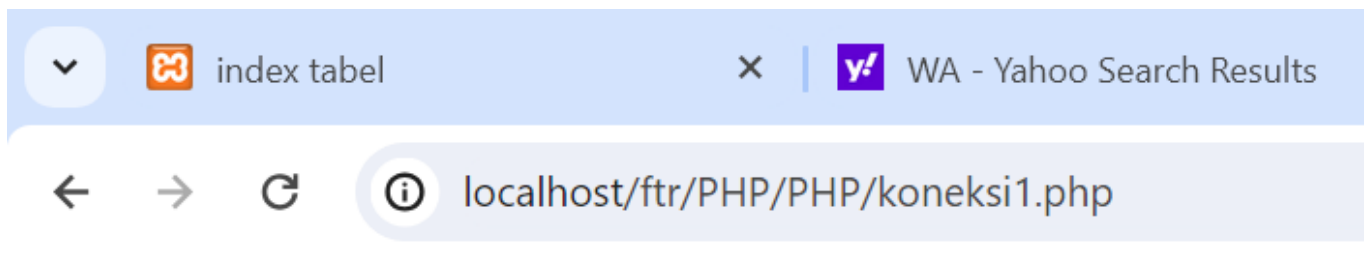
Program ini adalah sebuah halaman HTML dengan form yang mengirim data menggunakan metode POST. Data yang dikirim kemudian ditangkap di file "proses_post.php" menggunakan variabel `$_POST`. Namun, ada ketidaksesuaian antara atribut `name` pada input field dengan variabel yang digunakan di PHP. Sehingga, program ini perlu diperbaiki dengan mengganti atribut `name="nama_lengkap"` menjadi `name="nama"` agar sesuai dengan variabel yang digunakan di PHP. Selain itu, program juga bisa ditingkatkan dengan penanganan data yang lebih baik, seperti validasi input, sanitasi data, dan penggunaan metode keamanan yang lebih baik untuk mengelola password.

Koneksi Database

Kode Program

```
<?php
//koneksi ke database
$koneksi = mysqli_connect('localhost', 'root', '', 'rental_fatir');
// Memeriksa koneksi
if (!$koneksi) {
    die("Error, tidak bisa koneksi ke database: " . mysqli_connect_error());
} else {
    echo "<br> Koneksi aman <br>";
}
?>
```

Hasil



Koneksi aman

Analisis

- Fungsi `mysqli_connect` digunakan untuk membuat koneksi ke database MySQL.
- Parameter pertama adalah nama host, dalam hal ini 'localhost'.
- Parameter kedua adalah nama pengguna database, dalam hal ini 'root'.
- Parameter ketiga adalah kata sandi pengguna database, dalam hal ini kosong ('').
- Parameter keempat adalah nama database yang akan digunakan, dalam hal ini 'rental_angga'.
- Menggunakan fungsi `mysqli_connect_error` untuk menampilkan pesan error jika koneksi gagal.
- Jika koneksi berhasil, pesan "Koneksi aman" akan ditampilkan.

Kesimpulan

Kode tersebut berguna untuk memastikan bahwa koneksi ke database berhasil dilakukan sebelum menjalankan operasi-operasi database lainnya. Jika koneksi gagal, maka akan menampilkan pesan error dan menghentikan eksekusi script dengan `die()`.

Tampilkan Data

Kode Program

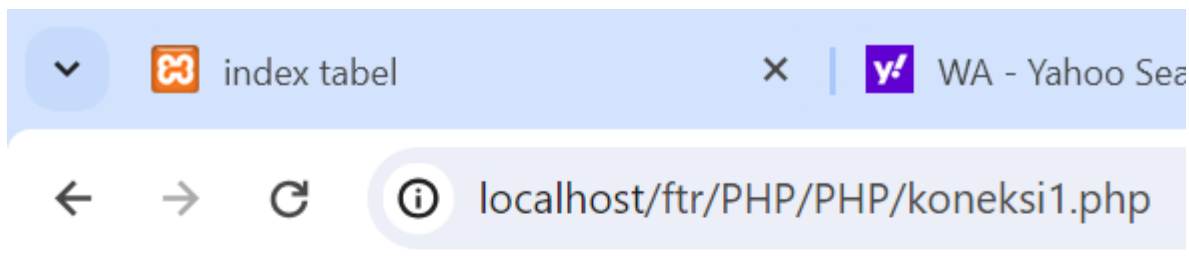
```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>index tabel</title>
```

```

</head>
<body>
  <h2>Data Siswa </h2>
  <table border="5">
    <tr>
      <th>id_siswa</th>
      <th>nama</th>
      <th>email</th>
      <th>jenis_kelamin</th>
      <th>alamat</th>
    </tr>
    <?php
    include "koneksi.php";
    $i = 1;
    $query = mysqli_query($koneksi, "SELECT * FROM siswa");
    while ($data = mysqli_fetch_array($query)) {
      ?>
      <tr>
        <td><?php echo $i; ?></td>
        <td><?php echo $data['nama']; ?></td>
        <td><?php echo $data['email']; ?></td>
        <td><?php echo $data['jenis_kelamin']; ?></td>
        <td><?php echo $data['alamat']; ?></td>
      </tr>
      <?php
      $i++;
    }
    ?>
  </table>
</body>
</html>

```

Hasil



Data Siswa

id_siswa	nama	email	jenis_kelamin	alamat
1	fatir	tir@gmail.com	Laki-laki	mars

Analisis

- Baris 1-8: Membuat struktur dasar dokumen HTML dengan pengaturan charset UTF-8 dan viewport untuk memastikan kompatibilitas dengan berbagai perangkat. Judul halaman ditetapkan sebagai "index tabel".
- Baris 9-16: Membuat tabel dengan border 5 dan menambahkan header kolom untuk `id_siswa`, `nama`, `email`, `jenis_kelamin`, dan `alamat`.
- Baris 17-32: Mengimpor file koneksi database (`koneksi.php`). Menginisialisasi variabel `$i` untuk penomoran baris. Mengambil data dari tabel `siswa` di database dan menampilkan setiap baris data dalam tabel HTML. Setiap iterasi dari loop `while` menambahkan satu baris baru ke tabel HTML dengan data siswa yang diambil dari database. Penomoran baris diatur menggunakan variabel `$i` yang diincrement setiap kali data ditampilkan.
- Baris 33-35: Menutup tag tabel dan dokumen HTML.

Kesimpulan

- Membuat struktur dasar dokumen HTML dengan pengaturan charset dan viewport.
- Membuat tabel HTML dengan header kolom untuk `id_siswa`, `nama`, `email`, `jenis_kelamin`, dan `alamat`.
- Mengimpor koneksi database.
- Mengambil data dari tabel `siswa` di database dan menampilkan setiap baris data dalam tabel HTML.

- Menampilkan setiap baris data dengan penomoran yang diatur menggunakan variabel \$i.

Tambahkan Data

Kode Program

```
<!DOCTYPE html>
<html lang="en">
<head>
    <title>Document</title>
</head>
<body>
    <h2>Tambah Data</h2>
    <?php
    include "koneksi.php";
    if (isset($_POST['simpan'])) {
        $nama = $_POST['nama'];
        $email = $_POST['email'];
        $jenis_kelamin = $_POST['jenis_kelamin'];
        $alamat = $_POST['alamat'];
        $query = mysqli_query($koneksi, "INSERT into
siswa(nama,email,jenis_kelamin,alamat)
values ('$nama','$email','$jenis_kelamin','$alamat')");
        if ($query == true) {
            echo "<script>
            alert('Tambah data Berhasil')
            window.location.href='table.php'
            </script>";
        } else {
            echo '<script>alert("Tambah data gagal")</script>';
        }
    }
    ?>
    <form method="post">
        <table>
            <tr>
                <td>Nama</td>
                <td><input type="text" name="nama"></td>
            </tr>
            <tr>
                <td>Email</td>
                <td><input type="text" name="email"></td>
            </tr>
            <tr>
```



```

        <td>Jenis Kelamin</td>
        <td>>
            <select name="jenis_kelamin">
                <option>Laki-laki</option>
                <option>Perempuan</option>
            </select>
        </td>
    </tr>
    <tr>
        <td>Alamat</td>
        <td><input type="text" name="alamat"></td>
    </tr>
    <tr>
        <td></td>
        <td>
            <button name="simpan" type="submit">Simpan</button>
            <button type="reset">Reset</button>
            <a href="table.php">Kembali</a>
        </td>
    </tr>
</table>
</form>
</body>
</html>

```

Hasil

Before

id_siswa	Gambar	nama	email	jenis_kelamin	alamat	Aksi
1		fatir	tir@gmail.com	Laki-laki	mars	Ubah Hapus

After

id_siswa	Gambar	nama	email	jenis_kelamin	alamat	Aksi
1		fatir	tir@gmail.com	Laki-laki	mars	Ubah Hapus
2		fatir	ftir@gmail.com	Laki-laki	jupiter	Ubah Hapus

Analisis

- Baris 1-8: Membuat struktur dasar dokumen HTML dengan bahasa yang ditetapkan ke bahasa Inggris dan judul halaman "Document".
- Baris 9-26: Mengimpor file koneksi database (`koneksi.php`). Memeriksa apakah tombol "simpan" telah ditekan. Jika ya, mengambil data dari form dan menyimpan data baru siswa ke dalam database. Jika berhasil, menampilkan pesan sukses dan mengarahkan pengguna ke `table.php` . Jika gagal, menampilkan pesan error.
- Baris 27-55: Membuat formulir HTML untuk menambahkan data siswa baru. Terdapat input untuk mengisi nama, email, jenis kelamin, dan alamat siswa. Terdapat juga tombol untuk menyimpan data baru, mereset form, dan kembali ke halaman `table.php` .

Kesimpulan

1. Mengimpor koneksi database.
2. Memeriksa apakah formulir telah disubmit dan menyimpan data baru jika ya.
3. Menampilkan formulir HTML untuk mengisi data siswa baru.

Jika penambahan data berhasil dilakukan, pengguna akan diarahkan kembali ke halaman `table.php` dengan pesan sukses. Jika gagal, pengguna akan melihat pesan error. Formulir ini juga menyediakan opsi untuk mereset data yang telah diinput dan kembali ke halaman daftar siswa.

Ubah Data

Kode Program

```

<!DOCTYPE html>
<html lang="en">
<head>
    <title>Document</title>
</head>
<body>
    <h2>Ubah Data</h2>
    <?php
    include "koneksi.php";
    $id = $_GET['id_siswa'];
    if (isset($_POST['simpan'])) {
        $nama = $_POST['nama'];
        $email = $_POST['email'];
        $jenis_kelamin = $_POST['jenis_kelamin'];
        $alamat = $_POST['alamat'];
        $query = mysqli_query($koneksi, "UPDATE siswa SET
                                           nama='$nama',
                                           email='$email',
                                           jenis_kelamin='$jenis_kelamin',
                                           alamat='$alamat'
                                           WHERE id_siswa=$id");

        if ($query) {
            echo "<script>
                alert('ubah data Berhasil')
                window.location.href='table.php'
            </script>";
        } else {
            echo '<script>alert("ubah data gagal")</script>';
        }
    }
    $query = mysqli_query($koneksi, "SELECT * FROM siswa where id_siswa=$id");
    $data = mysqli_fetch_array($query);
    if ($data == "") {
        die('Data tidak ada');
    }
    ?>
    <form method="post">
        <table>
            <tr>
                <td>Nama</td>
                <td><input type="text" value="<?= $data['nama'] ?>"
name="nama"></td>
            </tr>
            <tr>
                <td>Email</td>
                <td><input type="text" value="<?= $data['email'] ?>"

```

```

name="email"></td>
</tr>
<tr>
<td>Jenis Kelamin</td>
<td>>
<select name="jenis_kelamin">
<option <?php if ($data['jenis_kelamin'] == "laki-
laki")
echo 'selected'; ?>>Laki-laki</option>
<option <?php if ($data['jenis_kelamin'] ==
"perempuan")
echo 'selected'; ?>>Perempuan</option>
</select>
</td>
</tr>
<tr>
<td>Alamat</td>
<td><input type="text" value="<?= $data['alamat'] ?>"
name="alamat"></td>
</tr>
<tr>
<td></td>
<td>
<button name="simpan" type="submit">Ubah</button>
<button type="reset">Reset</button>
<a href="table.php">Kembali</a>
</td>
</tr>
</table>
</form>
</body>
</html>

```

Hasil

Before

id_siswa	Gambar	nama	email	jenis_kelamin	alamat	Aksi
1		fatir	tir@gmail.com	Laki-laki	mars	Ubah Hapus
2		fatir	fi@gmail.com	Laki-laki	jupiter	Ubah Hapus

After

id_siswa	Gambar	nama	email	jenis_kelamin	alamat	Aksi
1		fatir	tir@gmail.com	Laki-laki	mars	Ubah Hapus
2		nur rezki alfatin	fr@gmail.com	Laki-laki	jupiter	Ubah Hapus

Analisis

- Baris 1-8: Membuat struktur dasar dokumen HTML dengan bahasa yang ditetapkan ke bahasa Inggris dan judul halaman "Document".
- Baris 9-11: Mengimpor file koneksi database (`koneksi.php`) dan mengambil parameter `id_siswa` dari URL.
- Baris 12-26: Memeriksa apakah tombol "simpan" telah ditekan. Jika ya, mengambil data dari form dan mengupdate data siswa di database. Jika berhasil, menampilkan pesan sukses dan mengarahkan pengguna ke `table.php` . Jika gagal, menampilkan pesan error.
- Baris 27-32: Mengambil data siswa dari database berdasarkan `id_siswa` . Jika data tidak ditemukan, menghentikan eksekusi dengan pesan "Data tidak ada".
- Baris 33-64: Membuat formulir HTML untuk mengubah data siswa dengan mengisi nilai awal berdasarkan data yang diambil dari database. Terdapat pilihan untuk mengubah

nama, email, jenis kelamin, dan alamat siswa. Terdapat juga tombol untuk menyimpan perubahan, mereset form, dan kembali ke halaman `table.php`.

Kasimpulan

1. Mengimpor koneksi database.
2. Mengambil `id_siswa` dari URL.
3. Memeriksa apakah formulir telah disubmit dan melakukan update data jika ya.
4. Mengambil data siswa dari database untuk mengisi nilai awal form.
5. Menampilkan formulir HTML dengan data yang bisa diubah.

Jika perubahan data berhasil dilakukan, pengguna akan diarahkan kembali ke halaman `table.php` dengan pesan sukses. Jika gagal, pengguna akan melihat pesan error. Formulir ini juga menyediakan opsi untuk mereset data yang telah diinput dan kembali ke halaman daftar siswa.

Hapus Data

Kode Program

```
<?php
include('koneksi.php');
if(isset($_GET['id'])){
    $id = $_GET['id'];
    $query = mysqli_query($koneksi, "DELETE FROM siswa WHERE id_siswa = $id");
    if($query) {
        echo "<script>
        alert('Hapus data Berhasil')
        window.location.href='table.php'
        </script>";
    }else {
        echo '<script>alert("Hapus data gagal")</script>';
    }
}
?>
```

Hasil

Before

id_siswa	Gambar	nama	email	jenis_kelamin	alamat	Aksi
1		fatir	tir@gmail.com	Laki-laki	mars	Ubah Hapus
2		nur rezki alfatr	ftir@gmail.com	Laki-laki	jupiter	Ubah Hapus
3		tetsuya	tetsuya@gmail.com	Laki-laki	bumi	Ubah Hapus

After

id_siswa	Gambar	nama	email	jenis_kelamin	alamat	Aksi
1		fatir	tir@gmail.com	Laki-laki	mars	Ubah Hapus
2		fatir	ftir@gmail.com	Laki-laki	jupiter	Ubah Hapus

Analisis

- Mengimpor file koneksi database (koneksi.php).
- Memeriksa apakah parameter ID (\$_GET['id']) telah diberikan melalui URL.
- Jika ID telah diberikan, maka ID tersebut disimpan dalam variabel \$id .
- Selanjutnya, melakukan query DELETE untuk menghapus data siswa dari tabel siswa berdasarkan ID siswa yang diberikan.
- Jika query DELETE berhasil dilakukan, maka akan menampilkan pesan sukses menggunakan JavaScript alert dan mengarahkan pengguna kembali ke halaman table.php .

- Jika query DELETE gagal, akan menampilkan pesan error menggunakan JavaScript `alert`.

Kesimpulan

- **Koneksi Database:** Kode tersebut memastikan terjadi koneksi yang sukses ke database sebelum melakukan operasi penghapusan data.
- **Penghapusan Data:** Jika parameter ID (`$_GET['id']`) tersedia dalam URL, maka data siswa dengan ID tersebut akan dihapus dari tabel `siswa` menggunakan perintah SQL `DELETE`.
- **Feedback Pengguna:** Setelah operasi penghapusan selesai, pengguna akan diberikan feedback berupa pesan sukses jika penghapusan berhasil dilakukan, atau pesan gagal jika terjadi masalah dalam penghapusan data.
- **Redirect:** Setelah pesan sukses ditampilkan, pengguna akan diarahkan kembali ke halaman `table.php` agar dapat melihat perubahan yang telah terjadi.

Session/Login

Kode Program

```
<?php
session_start();
if (isset($_POST['submit'])) {
    $username = $_POST['username'];
    $password = $_POST['password'];
    $koneksi = mysqli_connect('localhost', 'root', '', 'basis_data') or
die('error koneksi');
    $result = mysqli_query($koneksi, "SELECT * FROM user WHERE username =
'$username' AND password = '$password'");
    $data = mysqli_fetch_assoc($result);
    if (isset($data)) {
        $_SESSION['username'] = $data['username'];
        $_SESSION['nama'] = $data['nama'];
        $_SESSION['status'] = 'login';
        header('Location: user.php');
    } else {
        echo "Username dan Password Salah";
    }
}
?>
<!DOCTYPE html>
<html>
<head>
```



```

    <title>Login Session</title>
</head>
<body>
    <form method="post">
        <label>Username</label>
        <input type="text" name="username">
        <br>
        <label>Password</label>
        <input type="password" name="password">
        <br>
        <button type="submit" name="submit">Login</button>
    </form>
</body>
</html>

```

```

<?php

session_start();

if ($_SESSION['status'] == 'login' && $_SESSION['username'] == 'admin') {
    header("Location: admin.php");
}
if ($_SESSION['status'] ≠ 'login') {
    header('Location: session.php');
}

?>
<!DOCTYPE html>
<html lang="en">

<head>
    <title>Document</title>
</head>
<body>
    <h1>Halaman User</h1>
    <h1>Halo, <?= $_SESSION['nama'] ?></h1>
    <a href="logout.php">Logout</a>
</body>
</html>

```

```

<?php

if ($_SESSION['status'] == 'login' && $_SESSION['username'] ≠ 'admin') {
    header("Location: user.php");
}

```

```

        exit();
    } else if ($_SESSION['status'] == 'login' && $_SESSION['username'] == 'admin')
    {
        header("Location: admin.php");
    } else {
        header("Location: session.php");
    }
}

```

```
<?php
```

```
session_start();
```

```

if ($_SESSION['status'] == 'login' && $_SESSION['username'] ≠ 'admin') {
    header("Location: user.php");
}

```

```

if ($_SESSION['status'] ≠ 'login') {
    header('Location: session.php');
}

```

```
?>
```

```
<!DOCTYPE html>
```

```
<html lang="en">
```

```
<head>
```

```
    <title>Document</title>
```

```
</head>
```

```
<body>
```

```
    <h1>Halaman Admin</h1>
```

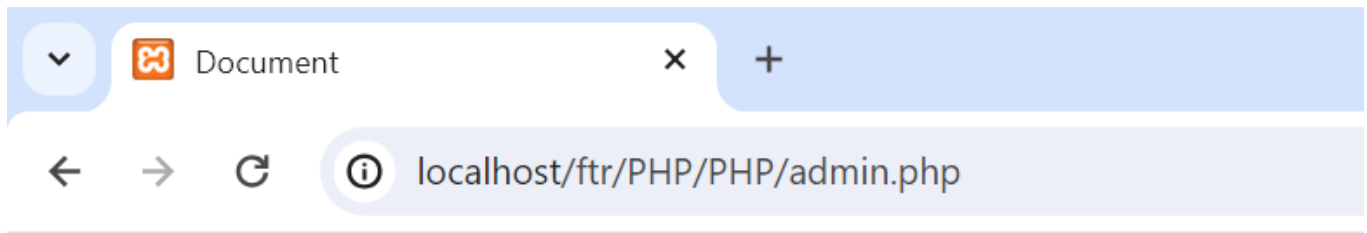
```
    <h1>Halo, <?= $_SESSION['nama'] ?></h1>
```

```
    <a href="logout.php">Logout</a>
```

```
</body>
```

```
</html>
```

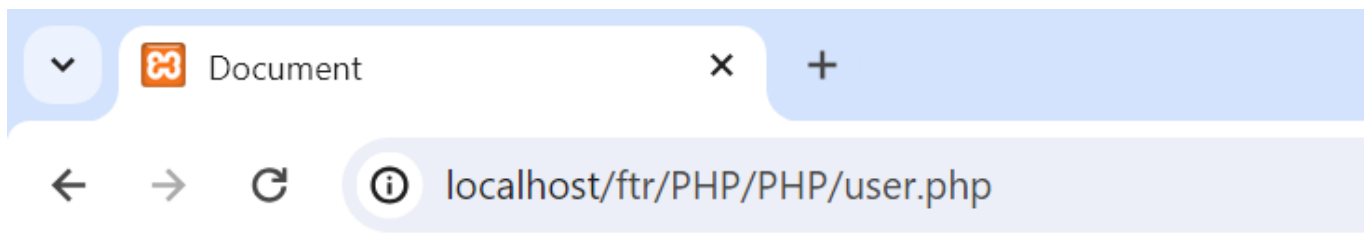
Hasil



Halaman Admin

Halo, fatir

[Logout](#)



Halaman User

Halo, tir

[Logout](#)

Analisis

- Fungsi `session_start()` digunakan untuk memulai session di PHP. Ini harus dipanggil di setiap halaman yang menggunakan session.

- Saat form login disubmit (`isset($_POST['submit'])`), nilai dari input username dan password diambil.
- Dilakukan koneksi ke database dan query untuk mencari data pengguna dengan username dan password yang cocok.
- Jika data ditemukan, session untuk username, nama, dan status login diset, dan pengguna diarahkan ke halaman `user.php`.
- Jika data tidak ditemukan, akan ditampilkan pesan "Username dan Password Salah".
- Form ini berisi input untuk username dan password, serta tombol submit untuk mengirim data login.

Kesimpulan

1. **Session Management:** Kode menggunakan `session_start()` untuk memulai session, dan `$_SESSION` digunakan untuk menyimpan informasi tentang pengguna yang sedang login seperti username, nama, dan status.
2. **Form Login:** Form login menggunakan metode POST untuk mengirimkan data username dan password. Jika form disubmit, kode akan mencocokkan informasi login dengan data yang ada di database.
3. **Database Connection:** Kode menggunakan fungsi `mysqli_connect()` untuk melakukan koneksi ke database MySQL. Ini dilakukan sebelum proses pengecekan login dilakukan.
4. **Redirect:** Jika informasi login benar, pengguna diarahkan ke halaman `user.php`. Jika tidak, pesan error ditampilkan.
5. **Keamanan:** Kode ini hanya menyajikan contoh sederhana dan tidak menerapkan praktik keamanan yang lengkap, seperti penggunaan prepared statement untuk mencegah SQL injection atau hashing password untuk keamanan password.
6. **Kesimpulan:** Kode ini menunjukkan cara dasar mengelola proses login menggunakan session di PHP. Ini adalah fondasi yang baik untuk dikembangkan lebih lanjut dengan penanganan kesalahan yang lebih baik, keamanan yang lebih baik, dan fitur tambahan seperti logout.

Upload dan Download

Upload

Kode Program

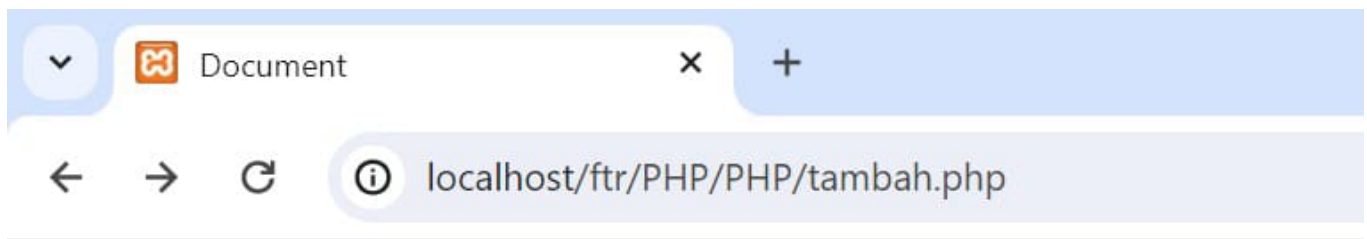
```
function upload(): string
{
    $nameImage = $_FILES['imageNew']['name'];
    $directoryFile = $_FILES['imageNew']['tmp_name'];
```

```

$errorImage = intval($_FILES['imageNew']['error']);
$sizeFile = $_FILES['imageNew']['size'];
// cek apakah gambar ada
if ($errorImage === 4) {
    echo "<script>alert('Anda Belum Upload Gambar')</script>";
    return false;
}
// mengambil ekstensi file
$validType = ['svg', 'jpg', 'png', 'jpeg', 'webp'];
$extensionFile = explode(".", $nameImage);
$extensionValid = strtolower(end($extensionFile));
// cek apakah yang diupload gambar atau bukan
if (!in_array($extensionValid, $validType)) {
    echo "<script>alert('yang anda Upload bukan gambar')</script>";
    return false;
}
// cek size file
if ($sizeFile > 3_000_000) {
    echo "<script>alert('Ukuran File Terlalu Besar!!(Maks 3MB)')
</script>";
    return false;
}
// upload file
$nameImage = uniqid() . "." . $extensionValid;
move_uploaded_file($directoryFile, "../assets/img/{$nameImage}");
// mengembalikan namafile yg sudah divalidasi
return $nameImage;
}

```

Hasil



Tambah Data

Nama

Email

Jenis Kelamin >

Alamat

Gambar beautiful-clouds-digital-art.jpg



Data Siswa Berprestasi

[+Tambah Data Baru](#)

[Export Excel](#)

id_siswa	Gambar	nama	email	jenis_kelamin	alamat	Aksi
1		fatir	tir@gmail.com	Laki-laki	mars	Ubah Hapus

Analisis

- **Variabel Input:** Fungsi ini mengambil input dari form dengan menggunakan variabel `$_FILES['imageNew']` untuk mendapatkan informasi tentang file gambar yang diunggah seperti nama file (`$nameImage`), lokasi sementara file (`$directoryFile`), error jika terjadi (`$errorImage`), dan ukuran file (`$sizeFile`).
- **Validasi:**
 - Fungsi ini melakukan beberapa validasi terhadap file gambar yang diunggah.
 - Pertama, dilakukan pengecekan apakah gambar sudah diunggah atau belum. Jika belum, akan muncul pesan peringatan dan fungsi akan mengembalikan `false`.
 - Selanjutnya, dilakukan pengecekan ekstensi file untuk memastikan bahwa yang diunggah adalah file gambar dengan ekstensi yang diizinkan (svg, jpg, png, jpeg, webp). Jika bukan gambar, akan muncul pesan peringatan dan fungsi akan mengembalikan `false`.
 - Kemudian, dilakukan pengecekan ukuran file. Jika ukurannya melebihi 3MB, akan muncul pesan peringatan dan fungsi akan mengembalikan `false`.
- **Proses Upload:** Jika semua validasi berhasil, maka file gambar akan diupload ke direktori yang ditentukan (`../assets/img/`) dengan nama file yang diacak (`$nameImage`).
- **Pengembalian Nama File:** Fungsi ini mengembalikan nama file gambar yang sudah divalidasi dan diupload agar dapat digunakan untuk menyimpan informasi file gambar tersebut di database atau di tempat lain yang diperlukan.
- **Pesan Peringatan:** Pesan peringatan ditampilkan menggunakan JavaScript `alert()` untuk memberi tahu pengguna tentang masalah yang terjadi saat upload file gambar.

Kesimpulan

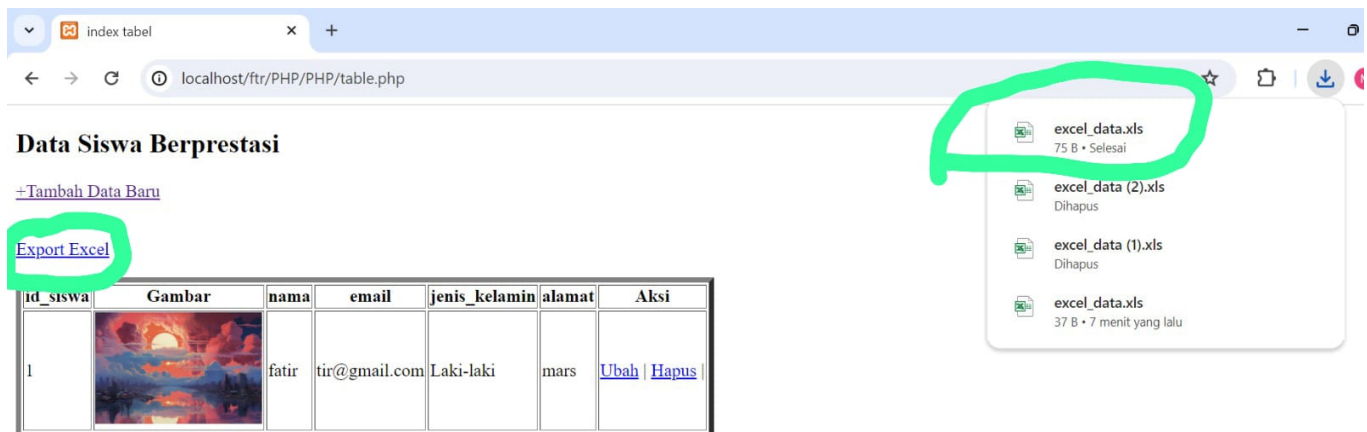
- **Validasi Input:** Fungsi ini melakukan validasi terhadap file gambar yang diunggah, termasuk pengecekan apakah file sudah diunggah, jenis file yang diunggah, dan ukuran file yang diunggah.
- **Penanganan Kesalahan:** Jika terjadi kesalahan dalam proses upload, seperti file belum diunggah, jenis file yang tidak valid, atau ukuran file yang terlalu besar, fungsi akan memberikan pesan peringatan dan mengembalikan `false`.
- **Proses Upload:** Jika tidak terjadi kesalahan, fungsi akan mengacak nama file gambar dan mengupload file gambar ke direktori yang ditentukan.
- **Pengembalian Nama File:** Fungsi akan mengembalikan nama file gambar yang sudah divalidasi dan diupload, sehingga dapat digunakan untuk keperluan selanjutnya, seperti menyimpan informasi file gambar tersebut di database.
- **Pesan Peringatan:** Pesan peringatan ditampilkan menggunakan JavaScript `alert()` untuk memberitahu pengguna tentang masalah yang terjadi saat upload file gambar.

Download


Kode Program

```
<?php
include "koneksi.php";
$query = mysqli_query($koneksi, 'SELECT * FROM siswa');
$data = [];
$data[] = ["ID", "Nama", "Email", "Jenis Kelamin", "Alamat"];
while ($row = mysqli_fetch_assoc($query)) {
    $data[] = [
        $row['id_siswa'],
        $row['nama'],
        $row['email'],
        $row['jenis_kelamin'],
        $row['alamat']
    ];
}
$namafile = "excel_data.xls";
header("Content-Type: application/vnd.ms-xls");
header("Content-Disposition: attachment;filename=\"$namafile\"");
header("Cache-Control: max-age=0");
$output = fopen("php://output", "w");
foreach ($data as $row) {
    fputcsv($output, $row, "\t");
}
fclose($output);
exit;
```

Hasil



The screenshot shows a web browser window with the address bar displaying 'localhost/fttr/PHP/PHP/table.php'. The page title is 'Data Siswa Berprestasi'. Below the title, there is a link '+Tambah Data Baru' and a link 'Export Excel' which is circled in green. Below the links is a table with the following data:

id_siswa	Gambar	nama	email	jenis_kelamin	alamat	Aksi
1		fatir	tir@gmail.com	Laki-laki	mars	Ubah Hapus

On the right side of the browser window, a download menu is open, showing a list of files. The first item, 'excel_data.xls' (75 B • Selesai), is circled in green.

Analisis

- **Koneksi Database:** Kode menggunakan `include "koneksi.php";` untuk mengimpor file

koneksi database yang menghubungkan ke database MySQL.

- **Query dan Pengambilan Data:** Dilakukan query `SELECT * FROM siswa` untuk mengambil semua data siswa dari tabel `siswa` dalam database. Data kemudian dimasukkan ke dalam array `$data` dalam format yang sesuai untuk diekspor ke file Excel.
- **Penyiapan File Excel:** Kode menggunakan header HTTP untuk mengatur tipe konten menjadi `application/vnd.ms-exls` agar browser menganggap file ini sebagai file Excel. Header `Content-Disposition` digunakan untuk memberi nama file Excel yang akan diunduh.
- **Penulisan Data ke File Excel:** Menggunakan `fputcsv()` untuk menulis data ke file Excel dengan delimiter tab (`"\t"`).
- **Penutup dan Keluar:** Setelah data ditulis ke file Excel, `fclose($output);` digunakan untuk menutup file dan `exit;` digunakan untuk mengakhiri skrip PHP.

Kesimpulan

Kode tersebut digunakan untuk mengambil data siswa dari database MySQL dan menyimpannya ke dalam file Excel yang kemudian diunduh oleh pengguna. Ini memanfaatkan kemampuan PHP untuk menghasilkan file Excel dengan menggunakan delimiter tab untuk memisahkan data. Kode ini memungkinkan pengguna untuk dengan mudah mengunduh data dalam format yang mudah dibaca dan dapat diedit di Excel.