Andiek Suncahyo

SID - Surabaya

**PHP DARI NOL PANDUAN PEMULA MENUJU MASTER CODING**

KATA PENGANTAR

Puji syukur penulis panjatkan ke hadirat Allah SWT, karena berkat rahmat dan karunia-Nya buku berjudul **“Panduan Belajar PHP untuk Pemula”** ini dapat diselesaikan.

Buku ini disusun dengan tujuan memberikan gambaran yang sederhana, sistematis, dan mudah dipahami bagi para pembaca yang baru pertama kali mengenal bahasa pemrograman PHP. Materi yang disajikan berfokus pada konsep dasar, mulai dari instalasi lingkungan kerja dengan XAMPP, sintaks dasar PHP, pengelolaan variabel, operator, struktur kontrol, perulangan, array, fungsi, hingga pemrograman berbasis objek.

Selain itu, buku ini juga membahas penggunaan **form, session, cookies**, serta pengelolaan **database MySQL** dengan contoh nyata pembuatan aplikasi sederhana. Dengan pendekatan belajar sambil praktik (*learning by doing*), diharapkan pembaca dapat lebih mudah memahami setiap konsep dan langsung mengaplikasikannya.

Penulis menyadari bahwa buku ini masih jauh dari sempurna. Oleh karena itu, kritik dan saran yang membangun dari pembaca sangat diharapkan demi penyempurnaan isi buku di masa mendatang.

Akhir kata, semoga buku ini bermanfaat bagi mahasiswa, pelajar, praktisi, maupun siapa saja yang ingin memulai perjalanan belajar pemrograman web dengan PHP.

Sidoarjo, September 2025

**Andiek Suncahyo**

DAFTAR ISI

Contents

[Install PHP 1](#_Toc210127254)

[Install XAMPP 1](#_Toc210127255)

[Memeriksa Port yang digunakan oleh PC 1](#_Toc210127256)

[Mematikan port yang ada di dalam PC Running 2](#_Toc210127257)

[Mengganti Port Confiq XAMPP 2](#_Toc210127258)

[Running XAMPP 2](#_Toc210127259)

[Cara membuat XAMPP auto-run setiap kali Windows dinyalakan: 3](#_Toc210127260)

[PHP PROGRAMING 4](#_Toc210127261)

[Pembuatan file pertama PHP 4](#_Toc210127262)

[PHP Variables and Echo Function 5](#_Toc210127263)

[Concatenation Operator, a href, BackSlash /Escape sequences 6](#_Toc210127264)

[Arithmetic operation 6](#_Toc210127265)

[If, else and nested if else Statements 7](#_Toc210127266)

[Comparison operators and Logical operators 7](#_Toc210127267)

[Arrays in PHP 10](#_Toc210127268)

[While and Do While Loop in PHP 14](#_Toc210127269)

[While Loop 14](#_Toc210127270)

[Do While Loop 15](#_Toc210127271)

[For Loop in PHP 16](#_Toc210127272)

[Struktur Dasar For Loop 16](#_Toc210127273)

[Contoh Penggunaan For Loop 16](#_Toc210127274)

[Kesimpulan 17](#_Toc210127275)

[Foreach Loop di PHP 18](#_Toc210127276)

[Apa Itu Foreach Loop 18](#_Toc210127277)

[Contoh Foreach dengan Array Sederhana 18](#_Toc210127278)

[Contoh Foreach dengan Key dan Value 19](#_Toc210127279)

[Contoh Foreach dengan Array Multidimensi 19](#_Toc210127280)

[Contoh Foreach dengan Objek 20](#_Toc210127281)

[Pembuatan Object Class 20](#_Toc210127282)

[Penggunaan Fungsi SWITCH di dalam PHP 22](#_Toc210127283)

[GET Variables in PHP 25](#_Toc210127284)

[Apa Itu GET Variables 25](#_Toc210127285)

[Cara Menggunakan GET Variables di PHP 25](#_Toc210127286)

[Menangani Error dengan isset() 25](#_Toc210127287)

[Menggunakan GET dengan Form HTML 26](#_Toc210127288)

[Perbedaan GET dan POST 27](#_Toc210127289)

[Keamanan GET Variables 27](#_Toc210127290)

[Penggunaan Metode $\_POST di dalam PHP 29](#_Toc210127291)

[Basic Functions 31](#_Toc210127292)

[Date and Time Functions 33](#_Toc210127293)

[Global Variables and Functions 40](#_Toc210127294)

[include() and require() functions 41](#_Toc210127295)

[PHP Session dan Setting PHP Session 44](#_Toc210127296)

[Contoh: Simulasi Login dan Logout dengan Session 47](#_Toc210127297)

[Penggunaan Cookies di dalam PHP 49](#_Toc210127298)

[Creating First MySQL Database With phpMyAdmin 53](#_Toc210127299)

[How to connect to MySQL database using PHP 54](#_Toc210127300)

[Getting Data from MySQL Database using PHP 55](#_Toc210127301)

[Create Update Delete (CRUD) 56](#_Toc210127302)

[Apa Itu Prepare() Param() Execute() 61](#_Toc210127303)

[Fungsi Explode() Dan Str\_Pad() 75](#_Toc210127304)

[Perbedaan = ( Assign) dan .= (Concatenate) di PHP 76](#_Toc210127305)

[GITHUB.COM CREATE UPLOAD DAN DOWNLOAD 86](#_Toc210127306)

[Persiapan Awal 86](#_Toc210127307)

[Urutan langkah-langkah lengkap untuk Pertama kali GIT: 87](#_Toc210127308)

[Tabel catatan perintah Git 87](#_Toc210127309)

[Cara download / ambil aplikasi dari GitHub ke laptop 88](#_Toc210127310)

[Sisipkan PHP pada Body HTML 89](#_Toc210127311)

# Install PHP

Download PHP pada Link di bawah ini:

<https://www.php.net/>

A screenshot of a computer

AI-generated content may be incorrect.

# Install XAMPP

Download XAMPP pada Link di bawah ini:

<https://www.apachefriends.org/de/index.html>

A screenshot of a computer

AI-generated content may be incorrect.

## **Memeriksa Port yang digunakan oleh PC**

Open CMD ketikan : *C:\Users\Andiek>netstat -ano*

**A screenshot of a computer

AI-generated content may be incorrect.**

Mematikan port yang ada di dalam PC Running:  
Open CMD ketikan: *C:\Users\Andiek>taskkill /pid <Number Port Yang Akan Di Matikan>*

A black background with white text

AI-generated content may be incorrect.

Mengganti Port Confiq XAMPP

Edit file ***httpd.conf*** dengan menggunakan notepad

A screenshot of a computer

AI-generated content may be incorrect.

Ganti Port 80 dengan port yang kosong semisal dengan port 8080

A screenshot of a computer program

AI-generated content may be incorrect.

Running XAMPP

A screenshot of a computer

AI-generated content may be incorrect.

Lakukan pengecekan masuk ke dalam explorer ketikan: [*http://localhost:80*](http://localhost:80)

A screenshot of a computer

AI-generated content may be incorrect.

Masuk ke dalam folder XAMPP dan temukan folder **htdocs,** folder tersebut adalah main GATEWAY dari XAMPP yang anda install di dalamnya terdapat semua aplikasi yang bisa dibuka melalui browser.

A screenshot of a computer

AI-generated content may be incorrect. A screenshot of a computer

AI-generated content may be incorrect.

Cara membuat XAMPP auto-run setiap kali Windows dinyalakan:

**1. Aktifkan Apache & MySQL dari XAMPP Control Panel**

1. Buka **XAMPP Control Panel** (klik icon XAMPP).
2. Klik tombol **Config** (di kanan atas).
3. Centang **Apache** dan **MySQL** pada bagian **Autostart of modules**.
4. Klik **Save** → Tutup dan buka ulang XAMPP Control Panel untuk cek apakah setting tersimpan.  
   (Sekarang setiap kali XAMPP Control Panel dibuka, Apache & MySQL otomatis start.)

**Catatan:** Langkah ini hanya meng-auto-start service saat XAMPP Control Panel aktif. Agar Control Panel-nya otomatis terbuka saat Windows startup, lanjut ke langkah 2.

**2. Tambahkan Shortcut ke Startup Folder**

1. Tekan **Windows + R** → ketik shell:startup → tekan Enter.  
   (Ini membuka folder Startup Windows.)
2. Buka lokasi file xampp-control.exe (biasanya di C:\xampp\xampp-control.exe).
3. Klik kanan file itu → pilih **Create shortcut**.
4. Pindahkan shortcut tersebut ke folder Startup yang tadi dibuka.  
   ✅ Sekarang setiap kali Windows dinyalakan, XAMPP Control Panel otomatis terbuka dan Apache/MySQL akan langsung jalan.

**3. (Opsional) Jalankan Sebagai Service**

Jika ingin Apache & MySQL berjalan **tanpa buka Control Panel**, bisa install sebagai service:

1. Buka XAMPP Control Panel.
2. Klik checkbox **Svc** di sebelah Apache → klik OK jika ada konfirmasi.
3. Lakukan hal yang sama untuk MySQL.
4. Restart PC → Apache & MySQL akan berjalan di background langsung tanpa harus buka panel.

# PHP PROGRAMING

Pembuatan file pertama PHP

Untuk membuat file PHP bisa menggunakan aplikasi **Visual Studio Code** pada link berikut <https://code.visualstudio.com/download> kemudian install sampai dengan sukses.

Buat satu folder dengan nama aplikasi disini kita kasih contoh nama folder “youtube” di dalam htdocs folder.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

Kemudian tuliskan code seperti pada gambar di atas untuk membuat file yang berisikan kata “Hello Word”

Disetiap penulisan code PHP pasti akan di mulai dengan **<?php** dan di akhiri dengan **?>** seperti contoh yang ada pada gambar code di atas.

Untuk membuat remark di dalam php bisa menggunakan tanda **(//), (#),(/\*..\*/)** dengan demikian semua tulisan di dalam tanda tersebut tidak akan dibaca oleh PHP.

Save file dan silahkan open link http://localhost/youtube/test.php maka akan tampil halaman hello word seperti tampak di bawah ini.

A screenshot of a computer

AI-generated content may be incorrect.

PHP Variables and Echo Function

Perhatikan code di bawah ini

A screenshot of a computer program

AI-generated content may be incorrect.

Dalam perbuatan variable di dalam PHP harus menggunakan tanda **“$”** seperti contoh yang ada di code samping ada pembuatan variable: **value** dan **Name.**

Variable di PHP ini sangat sensitive terhadap besar kecilnya huruf sebagai contoh:  
**Case Sensitive ($Boo!=$boo!=$Boo)**

Contoh diatas menunjukkan ada tiga variable yang berbeda.

**Nama variabel tertentu yang disediakan oleh PHP**  
- Variable **Form** ($ \_post, $ \_get) dan - Variable **Server**  ($ \_server) - dll

Untuk menampilkan hasil variable ke dalam tampilan layar bisa menggunakan **echo** seperti yang dituliskan ke di dalam code menggunakan (**“)** atau menggunakan (**‘).**

Bila menggunakan “ maka isi variable akan di tampilkan

Bila menggunakan ‘ maka apa yang ada di dalamnya akan ditampilkan seperti apa adanya seperti contoh yang ada di dalam code, di bawah ini adalah hasil code di atas yang akan ditampilkan di dalam layar.

A screenshot of a computer

AI-generated content may be incorrect.

Concatenation Operator, a href, BackSlash /Escape sequences

A computer screen shot of a program code

AI-generated content may be incorrect.

Di bab ini dijelaskan penggunaan simbol **Concat (.)** yaitu untuk menggabungkan string yang akan ditampilakan di layar oleh echo dengan variabel tertentu seperti contoh di atas.

Gunakan tanda **Back Slash (\)** di dalam echo string untuk menampilkan beberapa simbol yang digunakan PHP semisal mau menampilkan **(“)** seperti contoh di atas.

A screenshot of a computer

AI-generated content may be incorrect.

Arithmetic operation

A screenshot of a computer program

AI-generated content may be incorrect.

Dalam memasukkan perhitungan dengan menggunakan code PHP dengan contoh di samping.  
  
ADD (Tambah) = ( + )

SUBSTRACT (Pengurangan)=( - )

MULTIPLE (Perkalian) = ( \* )

DEVIDE (Pembagian) = ( / )

If, else and nested if else Statements

A screenshot of a computer program

AI-generated content may be incorrect.

Dari Gambar di atas adalah contoh untuk penggunaan **(IF)** di mana ada **tiga** option:

1. Apabila variable number1 sama dengan variable number 2 maka akan mengeluarkan hasil **“Dua number adalah sama”**
2. Apabila variable number 1 lebih kecil dari variable number 2 maka akan mengeluarkan hasil **“Number satu lebih kecil dari number dua”**
3. Option terakhir adalah apa bila keadaan di luar dari dua option yang sudah dituliskan pada code sebelumnya sehingga akan menghasilkan **“Dua number tidak ada dalam Option 1 dan 2”**

Comparison operators and Logical operators

A screenshot of a computer

AI-generated content may be incorrect.

Operator perbandingan ini sangat berguna dalam **pengkondisian** (if, switch), **looping** (while, for), dan berbagai **logika dalam PHP**. 🚀

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer program

AI-generated content may be incorrect.

A computer screen with text

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

Arrays in PHP

**Pengertian Array dalam Pemrograman PHP**

**Array** adalah struktur data dalam PHP yang digunakan untuk menyimpan **banyak nilai dalam satu variabel**. Dengan array, kita bisa menyimpan dan mengakses sekumpulan data tanpa harus membuat banyak variabel satu per satu.

<?php

// Cara lama (PHP 5 dan sebelumnya)

$buah = array("Apel", "Mangga", "Jeruk");

// Cara baru (PHP 5.4+)

$buah = ["Apel", "Mangga", "Jeruk"];

?>

**Jenis-Jenis Array di PHP**

1. **Indexed Array** (Array Berindeks Numerik)
2. **Associative Array** (Array Asosiatif)
3. **Multidimensional Array** (Array Multi Dimensi)

**1. Indexed Array (Array Berindeks Numerik)**

Indexed array menggunakan indeks numerik (dimulai dari **0**).

**A computer screen shot of text

AI-generated content may be incorrect.**

**2. Associative Array (Array Asosiatif)**

Associative array menggunakan **kunci (key)** sebagai indeks, bukan angka.

A screen shot of a computer

AI-generated content may be incorrect.

**3. Multidimensional Array (Array Multi Dimensi)**

Array multidimensi adalah **array di dalam array**, digunakan untuk menyimpan data yang lebih kompleks.

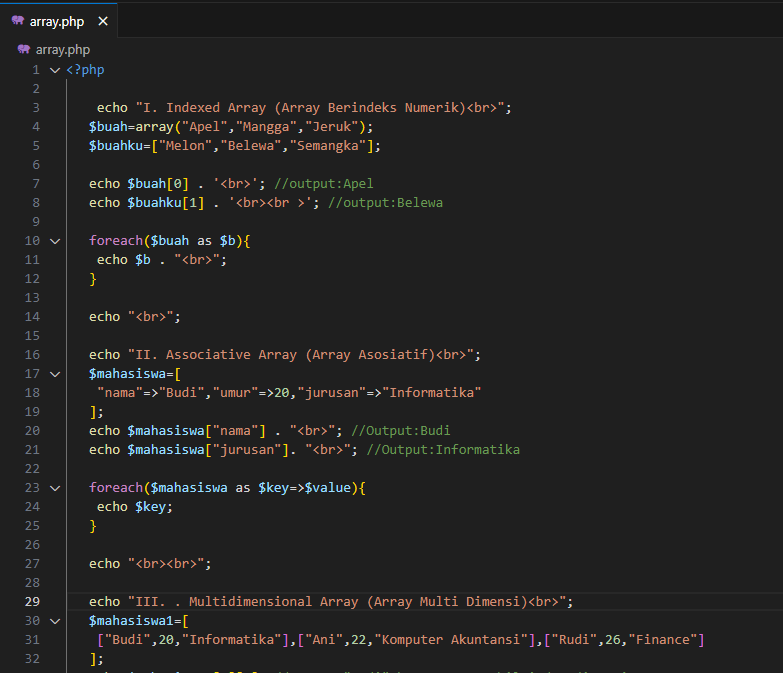
A screen shot of a computer

AI-generated content may be incorrect.

**Fungsi Array dalam PHP**

PHP menyediakan banyak fungsi untuk manipulasi array, seperti:

| **Fungsi** | **Deskripsi** |
| --- | --- |
| count($arr) | Menghitung jumlah elemen dalam array |
| array\_push($arr, $val) | Menambahkan elemen ke akhir array |
| array\_pop($arr) | Menghapus elemen terakhir array |
| array\_shift($arr) | Menghapus elemen pertama array |
| array\_unshift($arr, $val) | Menambahkan elemen di awal array |
| array\_merge($arr1, $arr2) | Menggabungkan dua array |
| array\_keys($arr) | Mengambil semua key dalam array asosiatif |
| array\_values($arr) | Mengambil semua nilai dalam array |
| in\_array($val, $arr) | Memeriksa apakah sebuah nilai ada di dalam array |
| sort($arr) | Mengurutkan array secara ascending |
| rsort($arr) | Mengurutkan array secara descending |

****

**A screen shot of a computer

AI-generated content may be incorrect.**

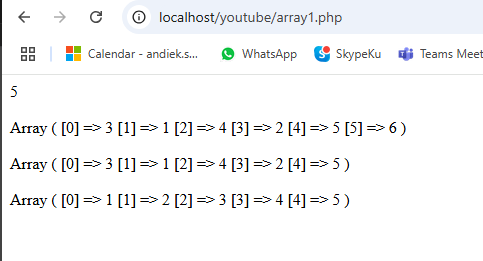
**A screenshot of a computer

AI-generated content may be incorrect.**

**Contoh Penggunaan Fungsi Array**

A screen shot of a computer program

AI-generated content may be incorrect.



**Kesimpulan**

* **Array** adalah variabel yang dapat menyimpan banyak nilai sekaligus.
* **Tiga jenis array utama** di PHP:
  1. **Indexed Array** → menggunakan angka sebagai indeks.
  2. **Associative Array** → menggunakan string sebagai indeks.
  3. **Multidimensional Array** → array dalam array.
* **Fungsi array** di PHP mempermudah manipulasi data.

While and Do While Loop in PHP

Dalam PHP, terdapat dua jenis perulangan yang sering digunakan untuk menjalankan kode secara berulang berdasarkan suatu kondisi, yaitu while dan do-while.

### While Loop

🔹 **Cara kerja:**

* Cek kondisi **terlebih dahulu** sebelum menjalankan kode di dalamnya.
* Jika kondisi **benar (true),** maka kode akan dijalankan.
* Setelah itu, kondisi dicek lagi sebelum mengulang prosesnya.
* Jika kondisi sudah **salah (false),** perulangan berhenti.

**Contoh While Loop:**

A screen shot of a computer code

AI-generated content may be incorrect.

**Hasil Output:**

A screenshot of a phone

AI-generated content may be incorrect.

📌 **Catatan:**

* Jika kondisi awalnya sudah **false,** maka kode dalam while **tidak akan dieksekusi sama sekali**.

### Do While Loop

🔹 **Cara kerja:**

* Kode **dijalankan dulu sekali**, baru kondisi dicek setelahnya.
* Jika kondisi **benar (true),** maka kode akan diulang.
* Jika kondisi **salah (false),** perulangan berhenti.

**Contoh Do While Loop:**

A computer screen shot of a number and numbers

AI-generated content may be incorrect.

**Hasil Output:**

A black text on a white background

AI-generated content may be incorrect.

📌 **Perbedaan utama dengan while loop**:

* do while **pasti menjalankan kode setidaknya sekali**, meskipun kondisi awalnya **false**.

**Contoh Jika Kondisi Awalnya False**

**A screenshot of a computer program

AI-generated content may be incorrect.**

**Hasil Output:**



**Kesimpulan**

| **Perbedaan** | **While Loop** | **Do While Loop** |
| --- | --- | --- |
| **Pengecekan kondisi** | Sebelum menjalankan kode | Setelah menjalankan kode sekali |
| **Jika kondisi awalnya false** | Tidak akan berjalan | Tetap dijalankan sekali |
| **Kapan digunakan?** | Jika ingin memastikan hanya menjalankan saat kondisi benar | Jika ingin kode dijalankan minimal sekali |

For Loop in PHP

### Struktur Dasar For Loop

For loop dalam PHP memiliki sintaks dasar seperti berikut:

**for (inisialisasi; kondisi; perubahan) {**

**// Blok kode yang akan dieksekusi**

**}**

**Penjelasan Setiap Bagian:**

* **Inisialisasi** → Bagian ini digunakan untuk mendeklarasikan dan memberikan nilai awal pada variabel penghitung. Biasanya berupa $i = 0;.
* **Kondisi** → Selama kondisi bernilai true, perulangan akan terus berjalan. Jika false, perulangan berhenti.
* **Perubahan** → Setelah setiap iterasi, variabel penghitung diubah, biasanya dengan i++ (menambah 1).

### Contoh Penggunaan For Loop

**Contoh 1: Menampilkan Angka 1 sampai 5**

**A computer screen shot of a black screen with white text

AI-generated content may be incorrect.**

* **Inisialisasi:** $i = 1;
* **Kondisi:** $i <= 5;
* **Perubahan:** $i++ (menambah 1 setiap iterasi)

**Hasil Output:**

A screenshot of a computer

AI-generated content may be incorrect.

**Contoh 2: Loop dengan Langkah Berbeda**

**A black background with white text

AI-generated content may be incorrect.**

**Hasil Output:**

A screenshot of a computer screen

AI-generated content may be incorrect.

* **Perubahan $i += 2** membuat angka bertambah 2 setiap iterasi.

**Contoh 3: Menggunakan for Loop untuk Menampilkan Array**

A black background with white text

AI-generated content may be incorrect.

**Hasil Output:**

A black text on a white background

AI-generated content may be incorrect.

* **count($buah)** digunakan untuk mendapatkan jumlah elemen dalam array.
* **Perulangan berjalan dari 0 sampai jumlah elemen array - 1.**

### Kesimpulan

* **For loop** digunakan untuk melakukan perulangan dengan jumlah yang sudah ditentukan.
* Struktur terdiri dari **Inisialisasi**, **Kondisi**, dan **Perubahan**.
* Bisa digunakan untuk menampilkan angka, mengelola array, atau proses lain yang membutuhkan perulangan.

Foreach Loop di PHP

### Apa Itu Foreach Loop

Foreach adalah salah satu perulangan (loop) di PHP yang digunakan khusus untuk mengiterasi (mengulang) elemen dalam array atau objek. Perulangan ini sangat berguna ketika kita ingin memproses setiap elemen dalam array tanpa harus mengetahui jumlah total elemen di dalamnya.

**Sintaks Dasar Foreach di PHP**

foreach ($array as $nilai) {

// Blok kode yang akan dijalankan

}

Atau jika kita ingin mendapatkan **kunci (key) dan nilai (value)**:

foreach ($array as $kunci => $nilai) {

// Blok kode yang akan dijalankan

}

### Contoh Foreach dengan Array Sederhana

A screen shot of a computer code

AI-generated content may be incorrect.

**Hasil Output:**

A screenshot of a computer

AI-generated content may be incorrect.

**Penjelasan:**

* Setiap elemen dalam array $buah akan disimpan sementara dalam variabel $item setiap kali perulangan berjalan.
* Kemudian, $item dicetak menggunakan echo.

### Contoh Foreach dengan Key dan Value

**A screenshot of a computer program

AI-generated content may be incorrect.**

**Hasil Output:**

A black text on a white background

AI-generated content may be incorrect.

Penjelasan:

* $nama menyimpan **key** (Andi, Budi, Citra).
* $usia menyimpan **value** (25, 30, 28).
* Loop akan berjalan sebanyak jumlah elemen dalam array.

### Contoh Foreach dengan Array Multidimensi

A computer code on a black background

AI-generated content may be incorrect.

**Hasil Output:**

A black text on a white background

AI-generated content may be incorrect.

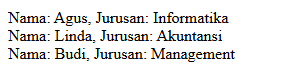
**Penjelasan:**

* foreach ($karyawan as $data) berarti setiap elemen dalam array $karyawan akan disimpan sementara dalam $data.
* $data["nama"], $data["posisi"], dan $data["gaji"] digunakan untuk mengakses nilai dalam array asosiatif.

### Contoh Foreach dengan Objek

****

**Hasil Output:**



**Penjelasan:**

* Objek dari class Mahasiswa dibuat dan disimpan dalam array $mahasiswa.
* Dalam foreach, setiap elemen objek bisa diakses dengan $mhs->nama dan $mhs->jurusan.

**Kesimpulan**

* Foreach digunakan untuk mengulang array dan objek dengan cara yang lebih mudah dibandingkan for atau while.
* Bisa digunakan untuk **array sederhana, array asosiatif, array multidimensi, dan objek**.
* Sangat berguna ketika kita ingin membaca data tanpa harus menggunakan **counter/index manual**.

Pembuatan Object Class

Dalam pemrograman berbasis objek (OOP), kelas berfungsi sebagai cetakan (template) untuk membuat objek.

A computer screen shot of a program code

AI-generated content may be incorrect.

**Hasil Output**

A screenshot of a computer

AI-generated content may be incorrect.

Menggunakan **kelas (class) Mahasiswa** dalam PHP memiliki beberapa keuntungan, terutama dalam konteks **pemrograman berorientasi objek (OOP)**. Berikut adalah beberapa **keuntungan utama** dari penggunaan kelas ini:

**1. Reusability (Dapat Digunakan Kembali)**

* Dengan menggunakan kelas Mahasiswa, kita bisa membuat banyak objek mahasiswa tanpa perlu menulis ulang struktur data atau fungsi.

**2. Encapsulation (Pengelompokan Data dan Fungsi)**

* **Data dan logika dalam satu tempat**:
  + Semua informasi terkait mahasiswa ($nama, $jurusan) disimpan di dalam kelas.
  + Setiap objek menyimpan datanya masing-masing.
* **Bisa menambahkan validasi di dalam kelas**:  
  Misalnya, kita bisa memastikan bahwa nama tidak kosong:

**3. Maintainability (Mudah Dikelola)**

* Jika ada perubahan dalam struktur data mahasiswa, kita hanya perlu **mengubah kelas**, bukan semua bagian kode yang menggunakannya.
* Contoh: Jika kita ingin menambahkan atribut NIM, kita cukup mengedit kelas saja.

**4. Scalability (Dapat Diperluas dengan Mudah)**

* Bisa menambahkan **method (fungsi) tambahan** ke dalam kelas.
* Misalnya, kita bisa menambahkan fungsi untuk menampilkan data mahasiswa:
* Dengan cara ini, **kode tetap rapi dan mudah dikembangkan**.

**5. Abstraction (Menyembunyikan Detail yang Tidak Perlu)**

* Kelas menyembunyikan detail implementasi dari pengguna kelas.
* Pengguna cukup tahu bahwa **cukup membuat objek dan mengisinya**, tanpa perlu tahu bagaimana data dikelola di dalamnya.

**6. Object-Oriented Principles (Mengikuti Prinsip OOP)**

* **Encapsulation**: Data disimpan dalam satu tempat.
* **Reusability**: Kelas dapat digunakan berkali-kali.
* **Maintainability**: Mudah diperbaiki dan dikembangkan.
* **Scalability**: Bisa ditambahkan metode baru tanpa merusak kode lama.

**Kesimpulan**

Menggunakan kelas dalam PHP memiliki **banyak keuntungan**, seperti:

✅ **Kode lebih rapi dan terorganisir**  
✅ **Bisa digunakan berulang kali (reusability)**  
✅ **Mudah dikembangkan dan diperbarui (scalability & maintainability)**  
✅ **Menyembunyikan detail teknis dari pengguna (abstraction)**

Dengan konsep ini, kode lebih **efisien, modular, dan fleksibel** untuk pengembangan lebih lanjut!

## Penggunaan Fungsi SWITCH di dalam PHP Bottom of Form

**Apa itu switch dalam PHP**

switch adalah salah satu struktur kendali dalam PHP yang digunakan untuk menggantikan banyak if...else if...else. Ini sangat berguna ketika kita ingin membandingkan satu variabel dengan banyak kemungkinan nilai.

**Sintaks Dasar switch**

switch ($variabel) {

case nilai1:

// kode yang akan dijalankan jika $variabel == nilai1

break;

case nilai2:

// kode yang akan dijalankan jika $variabel == nilai2

break;

default:

// kode yang akan dijalankan jika tidak ada yang cocok

}

**Cara Kerja switch**

1. **Memeriksa nilai variabel** – PHP akan membandingkan nilai dari variabel dengan setiap case yang ada.
2. **Menjalankan kode yang sesuai** – Jika ada nilai yang cocok, maka kode dalam case tersebut akan dieksekusi.
3. **Menggunakan break** – Setiap case biasanya diakhiri dengan break; untuk mencegah eksekusi ke case berikutnya.
4. **default (Opsional)** – Jika tidak ada nilai yang cocok, bagian default akan dieksekusi.

**Contoh Penggunaan switch**

A screenshot of a computer program

AI-generated content may be incorrect.

✅ **Output Yang Dihasilkan**

**A screenshot of a chat

AI-generated content may be incorrect.**

**2. Menggunakan Beberapa case Bersamaan**

Kadang kita ingin beberapa case memiliki hasil yang sama. Kita bisa menuliskannya tanpa break; di antara mereka.

A computer screen with text

AI-generated content may be incorrect.

✅ **Output yang Dihasilkan**

**A screenshot of a phone chat

AI-generated content may be incorrect.**

**3. Menggunakan default**

Jika tidak ada case yang cocok, maka blok default akan dijalankan.

A computer screen shot of text

AI-generated content may be incorrect.

✅ **Output Yang Di hasilkan**

A screenshot of a phone chat

AI-generated content may be incorrect.

**4. Contoh switch Tanpa break**

Jika kita tidak menulis break, maka PHP akan terus menjalankan semua case di bawahnya, meskipun nilai sudah cocok.

A screenshot of a computer program

AI-generated content may be incorrect.

✅ **Output Yang Di Hasilkan**

A screenshot of a computer

AI-generated content may be incorrect.

⚠ **Kenapa?** Karena tidak ada break;, maka PHP menjalankan semua case setelah menemukan kecocokan.

**Kapan Menggunakan switch vs if...else?**

| **Kondisi** | **Gunakan** |
| --- | --- |
| Jika hanya ada 2-3 pilihan sederhana | if...else |
| Jika ada banyak kemungkinan dengan perbandingan yang sederhana | switch |
| Jika ada kondisi kompleks (seperti perbandingan <, >, !=) | if...else |

🔹 **switch lebih rapi dan lebih mudah dibaca** dibandingkan dengan banyak if...else if...else.

**Kesimpulan**

* switch digunakan untuk menggantikan banyak if...else if...else.
* Gunakan case untuk menentukan kondisi.
* Gunakan break; agar tidak mengeksekusi case berikutnya.
* default digunakan jika tidak ada yang cocok.
* switch lebih bersih dibandingkan banyak if...else, tetapi hanya bisa digunakan untuk perbandingan **sama dengan (==)**.

## GET Variables in PHP

### Apa Itu GET Variables

GET Variables di PHP adalah cara untuk mengambil data dari URL menggunakan metode HTTP GET. Data yang dikirim melalui metode GET akan ditambahkan ke URL dalam bentuk **query string**.

**Contoh format URL dengan metode GET:**

http://example.com/halaman.php?nama=Andi&usia=25

Pada URL di atas:

* ?nama=Andi&usia=25 adalah **query string**
* nama=Andi berarti variabel nama memiliki nilai "Andi"
* usia=25 berarti variabel usia memiliki nilai 25

### Cara Menggunakan GET Variables di PHP

Di PHP, kita dapat mengambil data dari URL menggunakan array **superglobal $\_GET**. Array ini berisi pasangan **key-value** dari parameter yang dikirim melalui URL.

**Contoh Dasar**

Misalnya, kita memiliki halaman PHP bernama **get.php**:

A screen shot of a computer program

AI-generated content may be incorrect.

**Jika kita mengakses:**

*http://localhost/youtube/get.php?nama=Andi&usia=25*

A screenshot of a computer

AI-generated content may be incorrect.

### Menangani Error dengan isset()

Jika kita mencoba mengakses variabel yang tidak ada di URL, PHP akan menampilkan error. Untuk menghindari error, gunakan fungsi isset() untuk mengecek apakah variabel tersedia.

A computer code on a black background

AI-generated content may be incorrect.

Jika kita mengakses **isset\_get.php** tanpa parameter GET, misalnya:

*http://localhost/youtube/isset\_get.php*

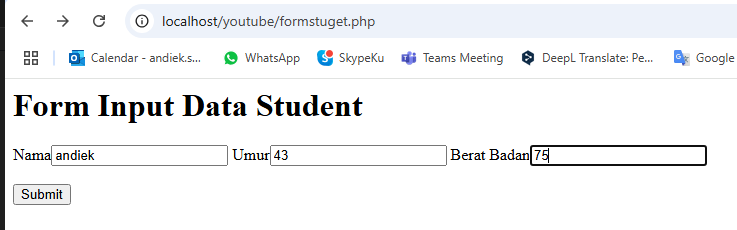
Maka output yang muncul:

A screenshot of a chat

AI-generated content may be incorrect.

A screen shot of a computer code

AI-generated content may be incorrect.



A screenshot of a computer

AI-generated content may be incorrect. A screenshot of a computer

AI-generated content may be incorrect.

### Menggunakan GET dengan Form HTML

Selain dari URL langsung, kita juga bisa mengirim data menggunakan **form HTML** dengan metode GET.

**Contoh formget**

A screen shot of a computer program

AI-generated content may be incorrect.

Jika pengguna mengisi nama dan usia, lalu mengklik "Kirim", browser akan mengarahkan ke URL

A screenshot of a computer

AI-generated content may be incorrect. A screenshot of a computer

AI-generated content may be incorrect.

### Perbedaan GET dan POST

| **Fitur** | **GET** | **POST** |
| --- | --- | --- |
| Data dikirim melalui | URL (query string) | Body request |
| Batasan panjang data | Ada (sekitar 2048 karakter) | Tidak terbatas |
| Keamanan | Kurang aman (data terlihat di URL) | Lebih aman (data tidak terlihat) |
| Cocok untuk | Pencarian, navigasi, berbagi link | Formulir login, data sensitif |

### Keamanan GET Variables

Karena data GET terlihat di URL, ada beberapa hal yang perlu diperhatikan:

1. **Hindari mengirimkan informasi sensitif** seperti password.
2. Gunakan **htmlspecialchars()** untuk menghindari serangan **XSS (Cross-Site Scripting)**.

**Parameter:**

1. $type → Tipe sumber input:
   * INPUT\_GET → Data dari URL ($\_GET)
   * INPUT\_POST → Data dari formulir ($\_POST)
   * INPUT\_COOKIE → Data dari cookie ($\_COOKIE)
   * INPUT\_SERVER → Data dari server ($\_SERVER)
   * INPUT\_ENV → Data dari environment ($\_ENV)
2. $variable\_name → Nama variabel yang ingin diambil.
3. $filter (Opsional) → Filter yang digunakan. Defaultnya FILTER\_DEFAULT (tidak memfilter).
   * Contoh: FILTER\_SANITIZE\_STRING, FILTER\_VALIDATE\_EMAIL, FILTER\_VALIDATE\_INT, dll.
4. $options (Opsional) → Parameter tambahan untuk filter.

A screen shot of a computer code

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

1. **Gunakan filter\_input()** untuk validasi input. **(Sudah tidak dianjurkan lagi untuk PHP di atas PHP 8)**

**A black screen with white text

AI-generated content may be incorrect.**

**A screenshot of a computer

AI-generated content may be incorrect.**

**Kesimpulan**

* $\_GET digunakan untuk mengambil data dari URL.
* Data dikirim dalam bentuk **query string**.
* Gunakan isset() untuk mengecek apakah variabel tersedia.
* Hindari mengirim informasi sensitif dengan GET.
* Gunakan htmlspecialchars() atau filter\_input() untuk keamanan.

## Penggunaan Metode $\_POST di dalam PHP

**📌 Apa Itu $\_POST?**

$\_POST adalah **variabel superglobal** di PHP yang digunakan untuk mengambil data yang dikirim dari **form HTML** dengan **metode POST**.

Artinya, saat pengguna mengisi form (misalnya nama dan email), lalu menekan tombol "submit", data itu bisa diambil di PHP menggunakan $\_POST.

**📝 Contoh Form HTML**

A screen shot of a computer program

AI-generated content may be incorrect.

* action="proses.php" artinya data akan dikirim ke file proses.php.
* method="POST" artinya data dikirim menggunakan metode POST.

**🧠 Cara Mengambil Data di PHP**

Sekarang kita ambil data yang dikirim tadi di file proses.php:

A computer screen shot of text

AI-generated content may be incorrect.

**✅ Kenapa Pakai POST?**

* Data tidak terlihat di URL (lebih **aman** dari GET).
* Bisa kirim **data lebih banyak**, cocok untuk form panjang.
* Cocok untuk form dengan **password**, **login**, dan sebagainya.

**⚠️ Tips Keamanan**

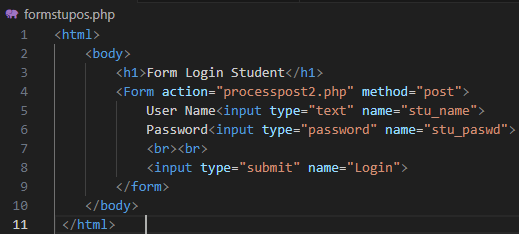
1. **Selalu cek apakah data ada** sebelum dipakai:

A computer screen shot of text

AI-generated content may be incorrect.

1. Contoh form Input Studies

Di bawah ini form dan process PHP yang menggunakan metode “POS” dengan memastikan text harus dipastikan tidak kosong.



A screen shot of a computer program

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect. A screenshot of a chat

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

**🔚 Kesimpulan**

$\_POST digunakan untuk menangkap data dari form HTML yang dikirim dengan metode POST. Ini sangat berguna untuk membuat form input pengguna seperti login, pendaftaran, komentar, dll.

## Basic Functions

**🔹** Apa itu Function di PHP?

**Function** adalah sekumpulan kode yang bisa digunakan berulang-ulang untuk melakukan suatu tugas tertentu. Jadi, daripada nulis kode yang sama berkali-kali, kamu cukup membuat satu function dan memanggilnya kapan saja kamu butuh.

**🔹** Kenapa Harus Pakai Function?

* ✅ Biar kode lebih rapi dan mudah dibaca.
* ✅ Menghindari pengulangan kode (DRY: Don’t Repeat Yourself).
* ✅ Memudahkan debugging dan pemeliharaan program.

**🔹** Cara Membuat Function di PHP

Format dasarnya seperti ini:

**function namaFunction() {**

*// kode yang akan dijalankan*

**}**

**Contoh sederhana:**

A screen shot of a computer program

AI-generated content may be incorrect.

A screenshot of a phone call

AI-generated content may be incorrect.

**🔹** Function dengan Parameter

Parameter adalah nilai yang bisa dikirim ke function agar hasilnya bisa berbeda-beda.

A screenshot of a phone chat

AI-generated content may be incorrect.A screen shot of a computer program

AI-generated content may be incorrect.

**🔹** Function dengan Return Value

Function juga bisa **mengembalikan nilai** ke pemanggilnya menggunakan return.

A screen shot of a computer program

AI-generated content may be incorrect.

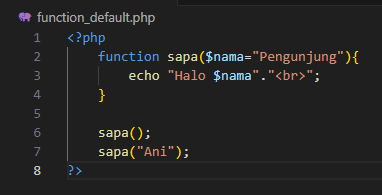
A screenshot of a phone

AI-generated content may be incorrect.

**JADI RETURN** bukan untuk menampilkan ke layar, tapi untuk mengembalikan hasil ke kode yang memanggil function tersebut.

**🔹** Function dengan Nilai Default

Kamu bisa kasih **nilai default** ke parameter, jadi kalau tidak dikirim nilai, dia pakai default-nya.

A screenshot of a phone

AI-generated content may be incorrect.

**🔹** Function Bawaan vs. Function Buatan Sendiri

* **Function Bawaan (Built-in)**: PHP sudah menyediakan banyak function, seperti:
  + strlen("Halo") → Menghitung panjang string.

A screen shot of a computer code

AI-generated content may be incorrect.

**Output :** “Jumlah String adalah 6”

* + date("Y-m-d") → Menampilkan tanggal hari ini.

A computer code with colorful text

AI-generated content may be incorrect.

**Output :** “Tanggal hari ini adalah *<Tanggal Komputer Sekarang>*”

* **Function Buatan Sendiri**: Kamu yang membuatnya sesuai kebutuhan.

**🔹 Kesimpulan**

| **Konsep** | **Penjelasan Singkat** |
| --- | --- |
| Function | Kumpulan kode untuk tugas tertentu |
| Parameter | Nilai yang dikirim ke function |
| Return | Nilai yang dikembalikan dari function |
| Default Parameter | Nilai bawaan kalau tidak dikirim parameter |
| Built-in vs Custom | Bawaan PHP vs buatan sendiri |

## Date and Time Functions

**📅 Apa Itu Date and Time Functions?**

PHP menyediakan **fungsi bawaan** untuk bekerja dengan **tanggal dan waktu**, seperti:

* Menampilkan tanggal sekarang
* Mengatur format tanggal
* Menghitung selisih waktu
* Menambah atau mengurangi hari
* Dll

**🔹 Fungsi date()**

Fungsi yang paling sering dipakai untuk menampilkan tanggal dan waktu.

A black screen with white text

AI-generated content may be incorrect. A screenshot of a phone call

AI-generated content may be incorrect.

**Penjelasan format:**

* Y = Tahun 4 digit (contoh: 2025)
* y = Tahun 2 digit (contoh: 25)
* m = Bulan (01 sampai 12)
* d = Hari (01 sampai 31)
* H = Jam (24 jam)
* h = Jam (12 jam)
* i = Menit
* s = Detik
* l (huruf L kecil) = Nama hari (Monday, Tuesday, dst.)
* D = Hari singkat (Mon, Tue, dst.)
* F = Nama bulan (January, February, dst.)

**Contoh:**

A black background with white text

AI-generated content may be incorrect. A black text on a white background

AI-generated content may be incorrect.

**🔹 Fungsi time()**

time() menghasilkan jumlah **detik sejak 1 Januari 1970 (Unix timestamp)**.

php

CopyEdit

echo time(); // Contoh: 1743562123

Gunanya untuk:

* Menyimpan waktu di database
* Menghitung selisih waktu
* Konversi waktu

**🔹 Fungsi mktime()**

Digunakan untuk membuat timestamp dari tanggal dan waktu tertentu.

A screen shot of a computer

AI-generated content may be incorrect. **Output:** 23-3-2025

Format **mktime**(jam, menit, detik, bulan, hari, tahun)

**🔹 Fungsi strtotime()**

Mengubah teks tanggal ke timestamp.

A black background with white text

AI-generated content may be incorrect.

**Output**: Senin, 31-03-2025 (jika hari ini Sabtu, 29 Maret 2025)

Contoh input:

* "now"
* "tomorrow"
* "+1 week"
* "last Sunday"
* "2025-04-01"

**🔹 Fungsi date\_default\_timezone\_set()**

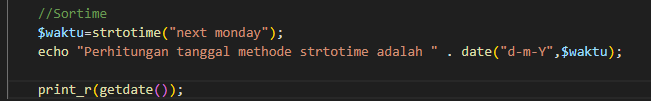
PHP secara default pakai zona waktu tertentu (kadang bukan WIB), jadi **penting** untuk atur sesuai zona kita.

A screen shot of a computer code

AI-generated content may be incorrect.

**🔹 Fungsi getdate()**

Memberikan informasi tanggal secara lengkap dalam bentuk array.

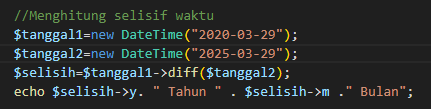
Contoh hasil:

Array ( [seconds] => 45 [minutes] => 30 [hours] => 14 [mday] => 29 [mon] => 3 [year] => 2025 [weekday] => Saturda ...)



**🔹 Fungsi date\_diff() (untuk menghitung selisih waktu)**

Khususnya pakai **DateTime object**.

**📌 Tips dan Catatan:**

* Selalu set timezone ke Asia/Jakarta biar waktu sesuai.
* Simpan waktu dalam bentuk timestamp atau format ISO (Y-m-d H:i:s) di database.
* Untuk aplikasi produksi, disarankan gunakan **DateTime class** karena lebih fleksibel.

**🚀 Contoh Kasus Sederhana**

**Menampilkan tanggal cetak invoice:**

**php**

date\_default\_timezone\_set("Asia/Jakarta");

echo "Tanggal Cetak: " . date("d-m-Y H:i");

**Hitung umur pelanggan:**

A screen shot of a computer code

AI-generated content may be incorrect. **Output:** Umur Sekarang 42 tahun

**🎯 Contoh Kasus: Hitung Estimasi Tanggal Selesai Produksi**

**📝 Skenario:**

Ingin menampilkan:

* Tanggal order masuk
* Estimasi selesai produksi (misalnya 3 hari kerja setelah order)
* Tanggal kirim

**✅ Contoh Kode PHP-nya:**

A computer screen shot of a program code

AI-generated content may be incorrect.   
**Output** *Semisal Tgl saat ini 6 April 2025*

A black text on a white background

AI-generated content may be incorrect.

**💡 Tips Tambahan:**

Kalau kamu mau **lewatkan hari Sabtu dan Minggu**, kita bisa pakai logika tambahan buat cari **hari kerja** saja.

Ide skenario untuk case di atas:

* Jadwal reminder otomatis
* Hitung deadline bayar DP
* Laporan bulanan pengiriman

**Contoh Format ISO-8601:**

| **Format** | **Arti** |
| --- | --- |
| 2025-04-06 | Tanggal (YYYY-MM-DD) → 6 April 2025 |
| 14:30:00 | Waktu (HH:MM:SS) → jam 2 siang lewat 30 menit |
| 2025-04-06T14:30:00Z | Tanggal dan waktu dalam format penuh (T = Time, Z = UTC) |
| N | Nomor hari (1 = Senin, 7 = Minggu) — seperti di kode kamu tadi |

**Menghitung estimasi selesai produksi + pengiriman hanya pada hari kerja (Senin–Jumat)**, **melewatkan Sabtu dan Minggu**.

**🎯 Skenario:**

* Order masuk hari apa pun
* Produksi butuh 3 hari kerja (bukan kalender)
* Tambah 1 hari kerja lagi untuk pengiriman
* Hasil: tampilkan tanggal order, selesai, dan kirim

**✅ Contoh Kode PHP:**

A screen shot of a computer program

AI-generated content may be incorrect.

**Output dengan estimation tanggal masuk mengikuti komputer saat ini (07-04-2025):**

A close up of text

AI-generated content may be incorrect.

**🧪 Contoh 2:** Misal hari ini **Sabtu, 29 Maret 2025**:

A black text on a white background

AI-generated content may be incorrect.

Karena 29 Maret itu Sabtu, proses mulai Senin (31 Maret), lalu hitung 3 hari kerja.

**Kita bisa lanjutkan fitur kayak:**

* Hitung deadline bayar DP maksimal 2 hari kerja
* Kirim notifikasi jika sudah 3 hari tapi belum dibayar
* Export jadwal produksi ke PDF atau Excel

**🎯 Skenario Nyata: Deadline Pembayaran DP Otomatis + Notifikasi**

**Alur:**

1. Pelanggan order hari ini (order date).
2. Sistem kasih waktu **maksimal 2 hari kerja** untuk bayar DP.
3. Kalau lewat 2 hari kerja, tampilkan **status: “Belum Bayar (Terlambat)”**.
4. Bisa ditambahkan fitur kirim notifikasi (misal ke WhatsApp atau email, nanti).

**✅ Contoh Kode PHP:**

A screen shot of a computer program

AI-generated content may be incorrect.

**🧪 Contoh Output Hari Ini (Selasa, 8 April 2025)**

A close up of numbers

AI-generated content may be incorrect.

**🧪 Contoh Output Hari Ini (Selasa, 9 April 2025)**

A close up of a phone

AI-generated content may be incorrect.

**🔧 Mau Lebih Otomatis?**

Bisa:

* Ambil data order dari database (MySQL)
* Simpan status ke dalam kolom status pembayaran
* Kirim notifikasi WA otomatis pakai API (kayak Wablas, Twilio, atau WhatsApp Business API)
* Buat dashboard admin untuk melihat order yang "Terlambat Bayar"

**Ayok Kita Buat:**

1. **Versi terhubung ke database**
2. **Form order + tracking status**
3. **Template notifikasi otomatis**

## Global Variables and Functions

**🌀 Apa itu Global Variables di PHP?**

Global variables (variabel global) adalah **variabel yang bisa diakses dari mana saja dalam script**, **terutama dari luar fungsi**. Tapi, secara **default**, variabel global **tidak langsung bisa diakses dari dalam fungsi**. Harus ada cara khusus untuk mengaksesnya.

**🔸 Contoh sederhana:**



**Output:** Halo Budi

🧠 **Catatan:** global memberitahu PHP bahwa kita ingin pakai variabel yang dideklarasikan di luar fungsi.

**🔄 Cara lain: $GLOBALS Array**

$GLOBALS adalah array khusus yang menyimpan **semua variabel global**. Kita bisa mengakses variabel global pakai nama variabel sebagai key-nya.

**Contoh:**

A screen shot of a computer code

AI-generated content may be incorrect.

**Output**: Umurnya: 25

**🧰 Fungsi (Functions) di PHP**

Function atau fungsi di PHP adalah **blok kode yang bisa digunakan berulang kali**. Fungsinya seperti alat: kita kasih input (jika perlu), lalu dapat hasilnya.

**🔧 Contoh Fungsi Sederhana:**

php

function salam($nama) {

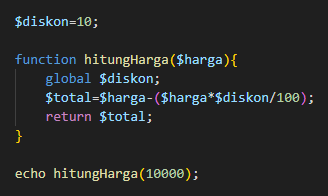
return "Halo, $nama!";

}

echo salam("Ani"); // Output: Halo, Ani!

**✨ Global Variables + Functions**

Kadang kita gabungkan variabel global dan fungsi, misalnya:



**Output**: 90000

**💡 Tips:**

* Gunakan **global variables dengan bijak**. Terlalu banyak variabel global bisa bikin program sulit dibaca dan debug.
* Kalau bisa, **pakai parameter** untuk mengirim data ke fungsi, lebih aman dan rapi.

## include() and require() functions

**🔹 Apa itu include() dan require() dalam PHP?**

include() dan require() adalah **fungsi bawaan PHP** yang digunakan untuk **memasukkan file PHP lain ke dalam file utama**.

Misalnya, kalau kamu punya file berisi header atau konfigurasi database, kamu bisa menyisipkannya di banyak file PHP tanpa harus menuliskannya berulang kali.

**🔹 Contoh Penggunaan Dasar**

Misal kamu punya file bernama **include\_header.php**:

**A screen shot of a computer

AI-generated content may be incorrect.**

**A screenshot of a computer

AI-generated content may be incorrect.**

**A screen shot of a computer

AI-generated content may be incorrect.**

**A screenshot of a computer

AI-generated content may be incorrect.**

**A screen shot of a computer

AI-generated content may be incorrect.**

**A screenshot of a computer

AI-generated content may be incorrect.**

**🔹 Perbedaan Antara include() dan require()**

| **Fungsi** | **Jika File Tidak Ditemukan** | **Lanjut Eksekusi?** |
| --- | --- | --- |
| include() | Akan menampilkan **peringatan (warning)** | Ya, skrip tetap lanjut |
| require() | Akan menampilkan **fatal error** | Tidak, skrip berhenti |

**✳ Contoh:**

A screen shot of a computer

AI-generated content may be incorrect.

A screenshot of a computer error

AI-generated content may be incorrect.

Terlihat scrip **echo “Script baris ini tidak akan dijalankan”** tidak di execution pada script di atas.

**🔹 Kapan Menggunakan include() dan require()?**

* Gunakan **require()** jika file yang disisipkan **sangat penting** untuk aplikasi, misalnya file konfigurasi, koneksi database, dll.
* Gunakan **include()** jika file tambahan tersebut **tidak wajib** dan aplikasi masih bisa berjalan tanpanya, seperti widget, footer, dll.

**🔹 Tips Tambahan**

1. **Gunakan include\_once dan require\_once** jika kamu ingin memastikan file hanya dimuat **satu kali**, bahkan jika dipanggil berkali-kali.

php

include\_once("config.php");

require\_once("header.php");

1. Ini sangat berguna untuk **menghindari error duplikasi fungsi** atau variable.

**🔹 Kesimpulan Singkat**

| **Fungsi** | **Lanjut jika error?** | **Digunakan untuk** |
| --- | --- | --- |
| include() | Ya (Warning) | File tambahan (tidak wajib) |
| require() | Tidak (Fatal Error) | File penting (wajib ada) |
|  |  |  |

## PHP Session dan Setting PHP Session

**📌 Apa itu PHP Sessions?**

PHP **session** adalah cara untuk menyimpan data (dalam bentuk variabel) yang bisa digunakan di **banyak halaman web**.

Contoh situasi:

* Kamu login di website → sistem menyimpan data siapa kamu (misal: nama atau user ID) supaya tidak perlu login lagi di setiap halaman.
* Kamu belanja online → sistem menyimpan daftar belanja kamu sampai checkout.

**Tanpa session**, data hanya tersimpan di satu halaman dan hilang begitu kamu pindah ke halaman lain.

**📌 Cara Kerja Session di PHP (Gambaran Sederhana)**

1. Ketika user pertama kali mengunjungi website, PHP membuat **session ID** unik untuk user itu.
2. Session ID ini biasanya disimpan di **cookie** di browser user.
3. Data session-nya sendiri disimpan di server.
4. Selama user terus mengakses website, PHP akan membaca session ID dan menghubungkannya dengan data di server.

**📌 Cara Menggunakan dan Mengatur PHP Sessions**

Berikut langkah-langkah dasar:

✅ **1. Memulai session**

Sebelum menyimpan atau mengakses data session, kamu **harus memulai session** dengan:



Ini biasanya ditaruh di **paling atas** file PHP sebelum kode HTML atau echo apapun.

✅ **2. Menyimpan data ke session**

Setelah session\_start(), kamu bisa menyimpan data seperti ini:

A black screen with white text

AI-generated content may be incorrect.

Sekarang ada dua data di session:

A screenshot of a chat

AI-generated content may be incorrect.

✅ **3. Mengakses data dari session**

Di halaman manapun, kamu bisa akses datanya (jangan lupa session\_start() dulu):

echo $\_SESSION['nama']; // Hasil: Andiek

echo $\_SESSION['usia']; // Hasil: 43

✅ **4. Menghapus data session tertentu**

Kalau mau hapus satu data saja:

A black background with white text

AI-generated content may be incorrect.

A black text on a white background

AI-generated content may be incorrect.

✅ **5. Menghancurkan semua session (logout misalnya)**

Kalau mau menghapus semua data session (misalnya pas user logout):



Biasanya, ini dilakukan setelah session\_start().

**📌 Contoh Lengkap**

**File 1: simpan\_session.php**

**A screen shot of a computer program

AI-generated content may be incorrect.**

**A screenshot of a chat

AI-generated content may be incorrect.**

**File 2: cek\_session.php**

**A screen shot of a computer code

AI-generated content may be incorrect.**

**A screenshot of a chat

AI-generated content may be incorrect.**

**File 3: hapus\_session.php**

A screen shot of a computer program

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

**⚠️ Catatan penting**

* Selalu panggil session\_start() di **atas** sebelum mengakses session.
* Jangan menaruh spasi atau output apapun sebelum session\_start(), karena akan memunculkan error.
* Session hanya hidup selama browser belum ditutup atau sampai server menghapusnya (bisa diatur dalam konfigurasi PHP).

**📌 Kenapa Session Penting?**

* Menjaga **status login**
* Menyimpan **keranjang belanja**
* Menyimpan **preferensi user**
* Menyimpan data antar halaman **tanpa perlu disimpan di URL**

## Contoh: Simulasi Login dan Logout dengan Session

Bayangkan kamu punya 3 file:

1. login.php → untuk menyimpan data ke session (anggap sebagai proses login)
2. dashboard.php → untuk mengecek apakah user sudah login
3. logout.php → untuk menghapus session (logout) menggunakan session\_destroy()

✅ **1. login.php**  
Simulasi login, menyimpan data ke session.

A screen shot of a computer code

AI-generated content may be incorrect.

A screenshot of a chat

AI-generated content may be incorrect.

✅ **2. dashboard.php**  
Mengecek apakah user sudah login.

A computer screen shot of a program code

AI-generated content may be incorrect.

A screenshot of a chat

AI-generated content may be incorrect.

✅ **3. logout.php**  
Menghapus session menggunakan session\_destroy().

A screen shot of a computer program

AI-generated content may be incorrect.

A screenshot of a chat

AI-generated content may be incorrect.

**📌 Penjelasan bagaimana session\_destroy() bekerja:**

* Saat kamu membuka logout.php, semua data session akan dihapus.
* Kalau kamu coba buka dashboard.php setelah logout, akan muncul pesan **“Kamu belum login”**.
* Kalau kamu buka login.php, session baru akan dibuat lagi.

**⚠️ Catatan penting**

* **session\_destroy()** hanya menghapus data di server, tapi kadang browser masih menyimpan session ID.  
  Untuk benar-benar membersihkan semua:

## Penggunaan Cookies di dalam PHP

**Apa itu Cookies di PHP?**

**Cookies** adalah data kecil yang disimpan di komputer/laptop browser pengguna (misalnya Chrome, Firefox).  
Data ini dikirim oleh server (PHP) dan bisa dipakai kembali ketika pengguna membuka halaman lain.

Contoh kegunaan cookies:

* Menyimpan nama pengguna supaya tidak perlu login lagi.
* Menyimpan preferensi seperti bahasa atau tema website.
* Menyimpan item keranjang belanja di toko online.

**Cara Membuat Cookies di PHP**

Gunakan fungsi setcookie().

<?php

A computer screen shot of text

AI-generated content may be incorrect.

?>

👉 Cookies ini akan tersimpan di browser pengguna selama **1 jam**.

**Cara Membaca Cookies di PHP**

Setelah cookies dibuat, kita bisa membaca isinya dengan $\_COOKIE.

<?php

A black screen with text on it

AI-generated content may be incorrect.

?>

**Cara Menghapus Cookies**

Untuk menghapus cookies, kita buat cookies dengan waktu yang sudah kadaluarsa:

A black screen with white text

AI-generated content may be incorrect.

**Contoh Lengkap**

<?php

// 1. Membuat cookies

setcookie("user", "Budi", time() + 3600);

// 2. Menampilkan cookies

if(isset($\_COOKIE["user"])) {

echo "Selamat datang kembali, " . $\_COOKIE["user"];

} else {

echo "Halo, ini pertama kali kamu berkunjung!";

}

// 3. Hapus cookies (jika perlu)

// setcookie("user", "", time() - 3600);

?>

**Intinya**

1. **Membuat cookies** → setcookie("nama", "isi", waktu\_kadaluarsa);
2. **Membaca cookies** → $\_COOKIE["nama"];
3. **Menghapus cookies** → set waktu kadaluarsa ke masa lalu.

**📝 Alur Sederhana**

1. User mengisi **username** lewat form.
2. Jika login sukses, server akan menyimpan username ke **cookies**.
3. Ketika buka halaman lain, website bisa membaca cookies dan otomatis mengenali user tanpa login lagi.
4. User juga bisa **logout** → cookies dihapus.

**1. Form Login (login.php)**

<!DOCTYPE html>

<html>

<head>

<title>Login dengan Cookies</title>

</head>

<body>

<h2>Login</h2>

<form method="post" action="proses\_login.php">

Username: <input type="text" name="username" required>

<br><br>

<button type="submit">Login</button>

</form>

</body>

</html>

**2. Proses Login (proses\_login.php)**

<?php

// Ambil input dari form

$username = $\_POST['username'];

// Simpan ke cookies selama 1 jam

setcookie("username", $username, time() + 3600);

// Redirect ke halaman welcome

header("Location: welcome.php");

exit();

?>

**3. Halaman Welcome (welcome.php)**

<?php

if(isset($\_COOKIE['username'])) {

echo "Selamat datang kembali, " . $\_COOKIE['username'] . "!<br>";

echo "<a href='logout.php'>Logout</a>";

} else {

echo "Kamu belum login. <a href='login.php'>Login dulu</a>";

}

?>

**4. Logout (logout.php)**

<?php

// Hapus cookies

setcookie("username", "", time() - 3600);

// Kembali ke halaman login

header("Location: login.php");

exit();

?>

**🖥️ Cara Coba**

1. Buka login.php, masukkan username.
2. Setelah login, akan diarahkan ke welcome.php → cookies sudah tersimpan.
3. Kalau browser ditutup lalu dibuka lagi (dalam 1 jam), user masih dianggap login.
4. Klik **Logout** → cookies dihapus, user harus login ulang.

**📝 Alur**

1. User isi **username** + **password**.
2. Server cek dengan data yang sudah ditentukan (contoh: username = admin, password = 12345).
3. Kalau benar → simpan cookies.
4. Kalau salah → tampilkan pesan error.
5. Ada juga tombol **logout** untuk hapus cookies.

**1. Form Login (login.php)**

<!DOCTYPE html>

<html>

<head>

<title>Login dengan Cookies</title>

</head>

<body>

<h2>Login</h2>

<form method="post" action="proses\_login.php">

Username: <input type="text" name="username" required><br><br>

Password: <input type="password" name="password" required><br><br>

<button type="submit">Login</button>

</form>

</body>

</html>

**PH2. Proses Login (proses\_login.php)**

<?php

// Data username & password yang diijinkan (hardcode untuk contoh)

$valid\_user = "admin";

$valid\_pass = "12345";

// Ambil data dari form

$username = $\_POST['username'];

$password = $\_POST['password'];

// Cek login

if($username === $valid\_user && $password === $valid\_pass) {

// Simpan ke cookies (berlaku 1 jam)

setcookie("username", $username, time() + 3600);

// Redirect ke halaman welcome

header("Location: welcome.php");

exit();

} else {

echo "Username atau Password salah! <a href='login.php'>Coba lagi</a>";

}

?>

**3. Halaman Welcome (welcome.php)**

<?php

if(isset($\_COOKIE['username'])) {

echo "Selamat datang, " . $\_COOKIE['username'] . "!<br>";

echo "<a href='logout.php'>Logout</a>";

} else {

echo "Kamu belum login. <a href='login.php'>Login dulu</a>";

}

?>

**4. Logout (logout.php)**

<?php

// Hapus cookies

setcookie("username", "", time() - 3600);

// Redirect ke halaman login

header("Location: login.php");

exit();

?>

**🚀 Cara Coba**

1. Buka login.php.
2. Masukkan **username: admin** dan **password: 12345**.  
   → Kalau benar, masuk ke welcome.php.  
   → Kalau salah, muncul pesan error.
3. Kalau cookies masih aktif (1 jam), bisa akses langsung welcome.php tanpa login.
4. Klik **Logout** → cookies terhapus, harus login ulang.

## Creating First MySQL Database With phpMyAdmin

**1. Apa itu phpMyAdmin?**

* **phpMyAdmin** adalah aplikasi berbasis web (dibuka melalui browser) yang digunakan untuk mengelola database MySQL/MariaDB.
* Dengan phpMyAdmin, kita bisa membuat database, tabel, mengatur data, hingga menjalankan query SQL tanpa harus mengetikkan perintah manual di terminal/command line.
* Biasanya sudah tersedia otomatis jika kita menginstall paket **XAMPP, Laragon, WAMP, atau LAMP**.

**2. Cara Membuka phpMyAdmin**

1. Pastikan **XAMPP/Laragon** sudah berjalan, terutama service **Apache** dan **MySQL**.
2. Buka browser, lalu ketik:
3. http://localhost/phpmyadmin
4. Jika berhasil, akan muncul halaman utama phpMyAdmin.

**3. Membuat Database Baru**

1. Setelah masuk phpMyAdmin, lihat menu di sebelah kiri atas.
2. Klik **"Database"**.
3. Akan muncul form untuk membuat database baru:
   * Masukkan **nama database** (contoh: belajar\_php).
   * Pilih **collation** (umumnya pilih utf8mb4\_general\_ci supaya mendukung karakter internasional termasuk emoji).
4. Klik tombol **Create**.
5. Selesai 🎉, database baru sudah dibuat.

**4. Membuat Tabel di dalam Database**

Database ibarat **lemari**, sedangkan tabel ibarat **laci** di dalam lemari.

1. Setelah database berhasil dibuat, klik nama database tersebut (contoh: belajar\_php).
2. Akan muncul menu untuk membuat tabel.
3. Masukkan:
   * **Nama tabel** (contoh: users)
   * **Jumlah kolom** (misalnya 4 kolom: id, nama, email, password).
4. Klik **Go**.

**5. Menentukan Struktur Kolom**

Di tahap ini, kita menentukan isi tabel (field/kolom):

* id → tipe **INT**, pilih **AUTO\_INCREMENT**, dan centang **PRIMARY KEY**.  
  (artinya ID akan otomatis bertambah setiap ada data baru).
* nama → tipe **VARCHAR**, panjang 100 (untuk nama maksimal 100 karakter).
* email → tipe **VARCHAR**, panjang 100.
* password → tipe **VARCHAR**, panjang 255 (karena biasanya password yang di-hash panjang).

Klik **Save**, maka tabel users sudah siap.

**6. Mengisi Data ke dalam Tabel**

1. Klik tabel users.
2. Pilih tab **Insert**.
3. Isi data contoh:
   * nama: Andi
   * email: andi@example.com
   * password: 123456
4. Klik **Go** → Data berhasil disimpan.
5. Untuk melihat hasil, klik tab **Browse**.

**7. Menjalankan Query SQL (Opsional)**

Jika ingin coba mengetikkan SQL manual:

1. Klik tab **SQL**.
2. Ketik perintah:
3. SELECT \* FROM users;
4. Klik **Go**, maka akan muncul semua data dari tabel users.

**8. Ringkasan**

* **Database** = lemari penyimpanan data.
* **Tabel** = laci di dalam database.
* **Kolom/Field** = kategori data (id, nama, email).
* **Row/Record** = isi datanya (contohnya: Andi, andi@example.com).
* **phpMyAdmin** mempermudah pemula agar tidak perlu menulis SQL panjang, cukup klik-klik saja.

A diagram of a database

AI-generated content may be incorrect.

## How to connect to MySQL database using PHP

**1. Cara Koneksi PHP ke MySQL**

Ada dua cara umum:

1. **mysqli (MySQL Improved Extension)** → lebih sering dipakai, mudah.
2. **PDO (PHP Data Objects)** → lebih fleksibel, bisa untuk banyak jenis database.

Untuk pemula, saya sarankan pakai **mysqli** dulu.

**2. Contoh Koneksi Menggunakan mysqli**

<?php

$servername = "localhost"; // biasanya localhost

$username = "root"; // default user XAMPP/Laragon

$password = ""; // default kosong (""), bisa diubah

$dbname = "belajar\_php"; // nama database yang sudah dibuat

// Membuat koneksi

$conn = new mysqli($servername, $username, $password, $dbname);

// Mengecek koneksi

if ($conn->connect\_error) {

die("Koneksi gagal: " . $conn->connect\_error);

}

else{ echo "Koneksi berhasil!";}

?>

✅ Kalau sukses → akan muncul tulisan **"Koneksi berhasil!"**  
❌ Kalau gagal → akan muncul error "Koneksi gagal: ..."

**3. Contoh Koneksi Menggunakan PDO**

<?php

$servername = "localhost";

$username = "root";

$password = "";

$dbname = "belajar\_php";

try {

$conn = new PDO("mysql:host=$servername;dbname=$dbname", $username, $password);

// Set error mode

$conn->setAttribute(PDO::ATTR\_ERRMODE, PDO::ERRMODE\_EXCEPTION);

echo "Koneksi berhasil!";

  //menutup koneksi database dengan metode PDO

$conn=null;

} catch(PDOException $e) {

echo "Koneksi gagal: " . $e->getMessage();

}

?>

Kelebihan PDO: bisa dipakai bukan hanya untuk MySQL, tapi juga PostgreSQL, SQLite, Oracle, dll.

**4. Struktur Folder**

* Simpan file PHP di dalam folder:
  + htdocs (jika pakai XAMPP)
  + www (jika pakai WAMP/Laragon)

**Contoh**:  
C:\xampp\htdocs\koneksi.php

Lalu akses lewat browser:

http://localhost/koneksi.php

**5. Tips untuk Pemula**

* Jika error: pastikan **Apache & MySQL** sudah aktif di XAMPP Control Panel.
* Nama database ($dbname) harus sama persis dengan yang dibuat di **phpMyAdmin**.
* Kalau password MySQL Anda tidak kosong, ganti bagian $password.

## Getting Data from MySQL Database using PHP

**1. Koneksi ke Database**

Pertama kita butuh koneksi ke database (seperti yang sudah saya jelaskan sebelumnya). Misalnya:

<?php

    //Deklarasi variable server

    $servername="localhost";

    $username="root";

    $password="root";

    $dbname="belajar\_php";

    mysqli\_report(MYSQLI\_REPORT\_ERROR | MYSQLI\_REPORT\_STRICT);

    //membuat Koneksi

    try{

        $conn=new mysqli($servername,$username,$password,$dbname);

**2. Mengambil Data dengan Query**

Misalkan kita punya tabel users dengan kolom: id, nama, email.

<?php

$sql = "SELECT id, nama, email FROM users";

$result = $conn->query($sql);

if ($result->num\_rows > 0) {

// Menampilkan setiap baris data

while($row = $result->fetch\_assoc()) {

echo "ID: " . $row["id"]. " - Nama: " . $row["nama"]. " - Email: " . $row["email"]. "<br>";

}

} else {

echo "Tidak ada data";

}

$conn->close();

?>

**3. Menampilkan Data dalam Bentuk Tabel HTML**

Supaya lebih rapi, kita bisa tampilkan hasil query dalam tabel:

<?php

$sql = "SELECT id, nama, email FROM users";

$result = $conn->query($sql);

if ($result->num\_rows > 0) {

echo "<table border='1' cellpadding='10'>";

echo "<tr><th>ID</th><th>Nama</th><th>Email</th></tr>";

while($row = $result->fetch\_assoc()) {

echo "<tr>";

echo "<td>" . $row["id"] . "</td>";

echo "<td>" . $row["nama"] . "</td>";

echo "<td>" . $row["email"] . "</td>";

echo "</tr>";

}

echo "</table>";

} else {

echo "Tidak ada data";

}

$conn->close();

?>

## Create Update Delete (CRUD)

contoh CRUD dasar yang simpel dan langsung bisa dipakai di XAMPP. Struktur filenya:

* db.php → koneksi database (mysqli + try/catch)
* create\_table.sql → buat tabel contoh
* index.php → list data + form tambah
* edit.php → form edit + update
* delete.php → hapus data

Asumsi: MySQL XAMPP (user root tanpa password), database belajar\_php.

**1) db.php – koneksi sekali, tinggal include**

<?php

// db.php

$servername = "localhost";

$username = "root";

$password = ""; // default XAMPP: kosong

$dbname = "belajar\_php";

mysqli\_report(MYSQLI\_REPORT\_ERROR | MYSQLI\_REPORT\_STRICT);

try {

$conn = new mysqli($servername, $username, $password, $dbname);

$conn->set\_charset("utf8mb4");

} catch (mysqli\_sql\_exception $e) {

die("Koneksi gagal: " . $e->getMessage());

}  
?>

**2) create\_table.sql – jalankan sekali di phpMyAdmin**

CREATE DATABASE IF NOT EXISTS belajar\_php;

USE belajar\_php;

CREATE TABLE IF NOT EXISTS mahasiswa (

id INT AUTO\_INCREMENT PRIMARY KEY,

nama VARCHAR(100) NOT NULL,

nim VARCHAR(30) NOT NULL UNIQUE,

jurusan VARCHAR(100) NOT NULL,

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP

);

**3) index.php – tampilkan data + form tambah**

<?php

require\_once "db.php";

// Handle Tambah (POST)

if ($\_SERVER['REQUEST\_METHOD'] === 'POST' && isset($\_POST['aksi']) && $\_POST['aksi'] === 'tambah') {

$nama = trim($\_POST['nama'] ?? '');

$nim = trim($\_POST['nim'] ?? '');

$jurusan = trim($\_POST['jurusan'] ?? '');

if ($nama === '' || $nim === '' || $jurusan === '') {

$err = "Semua field wajib diisi.";

} else {

try {

$stmt = $conn->prepare("INSERT INTO mahasiswa (nama, nim, jurusan) VALUES (?, ?, ?)");

$stmt->bind\_param("sss", $nama, $nim, $jurusan);

$stmt->execute();

header("Location: index.php?msg=Data+berhasil+ditambahkan");

exit;

} catch (mysqli\_sql\_exception $e) {

$err = "Gagal menambah data: " . $e->getMessage();

}

}

}

// Ambil semua data

$result = $conn->query("SELECT id, nama, nim, jurusan, created\_at FROM mahasiswa ORDER BY id DESC");

$rows = $result->fetch\_all(MYSQLI\_ASSOC);

?>

<!doctype html>

<html lang="id">

<head>

<meta charset="utf-8">

<title>CRUD Mahasiswa</title>

<meta name="viewport" content="width=device-width, initial-scale=1">

<style>

body{font-family:system-ui,-apple-system,Segoe UI,Roboto,Arial,sans-serif;max-width:960px;margin:24px auto;padding:0 16px}

table{border-collapse:collapse;width:100%}

th,td{border:1px solid #ddd;padding:8px}

th{background:#f5f5f5;text-align:left}

.row{display:grid;grid-template-columns:1fr 1fr;gap:24px}

.card{border:1px solid #e5e5e5;border-radius:12px;padding:16px}

input,button{padding:10px;border:1px solid #ccc;border-radius:8px;width:100%}

button{cursor:pointer}

.msg{background:#e9f7ef;border:1px solid #b6e1c7;padding:10px;border-radius:8px;margin-bottom:12px}

.err{background:#fdecea;border:1px solid #f5c6cb}

.actions a{margin-right:8px}

</style>

</head>

<body>

<h1>Data Mahasiswa</h1>

<?php if (!empty($\_GET['msg'])): ?>

<div class="msg"><?= htmlspecialchars($\_GET['msg']) ?></div>

<?php endif; ?>

<?php if (!empty($err)): ?>

<div class="msg err"><?= htmlspecialchars($err) ?></div>

<?php endif; ?>

<div class="row">

<div class="card">

<h2>List Data</h2>

<table>

<thead>

<tr>

<th>#</th>

<th>Nama</th>

<th>NIM</th>

<th>Jurusan</th>

<th>Dibuat</th>

<th>Aksi</th>

</tr>

</thead>

<tbody>

<?php if (count($rows) === 0): ?>

<tr><td colspan="6" style="text-align:center">Belum ada data</td></tr>

<?php else: ?>

<?php foreach ($rows as $i => $r): ?>

<tr>

<td><?= $i+1 ?></td>

<td><?= htmlspecialchars($r['nama']) ?></td>

<td><?= htmlspecialchars($r['nim']) ?></td>

<td><?= htmlspecialchars($r['jurusan']) ?></td>

<td><?= htmlspecialchars($r['created\_at']) ?></td>

<td class="actions">

<a href="edit.php?id=<?= urlencode($r['id']) ?>">Edit</a>

<a href="delete.php?id=<?= urlencode($r['id']) ?>" onclick="return confirm('Hapus data ini?')">Hapus</a>

</td>

</tr>

<?php endforeach; ?>

<?php endif; ?>

</tbody>

</table>

</div>

<div class="card">

<h2>Tambah Data</h2>

<form method="post">

<input type="hidden" name="aksi" value="tambah">

<p>

<label>Nama</label><br>

<input type="text" name="nama" required>

</p>

<p>

<label>NIM</label><br>

<input type="text" name="nim" required>

</p>

<p>

<label>Jurusan</label><br>

<input type="text" name="jurusan" required>

</p>

<p><button type="submit">Simpan</button></p>

</form>

</div>

</div>

</body>

</html>

**4) edit.php – form edit + update**

<?php

require\_once "db.php";

// Ambil ID

$id = isset($\_GET['id']) ? (int) $\_GET['id'] : 0;

if ($id <= 0) {

die("ID tidak valid");

}

// Handle Update (POST)

if ($\_SERVER['REQUEST\_METHOD'] === 'POST' && isset($\_POST['aksi']) && $\_POST['aksi'] === 'update') {

$nama = trim($\_POST['nama'] ?? '');

$nim = trim($\_POST['nim'] ?? '');

$jurusan = trim($\_POST['jurusan'] ?? '');

if ($nama === '' || $nim === '' || $jurusan === '') {

$err = "Semua field wajib diisi.";

} else {

try {

$stmt = $conn->prepare("UPDATE mahasiswa SET nama=?, nim=?, jurusan=? WHERE id=?");

$stmt->bind\_param("sssi", $nama, $nim, $jurusan, $id);

$stmt->execute();

header("Location: index.php?msg=Data+berhasil+diupdate");

exit;

} catch (mysqli\_sql\_exception $e) {

$err = "Gagal update: " . $e->getMessage();

}

}

}

// Ambil data lama

$stmt = $conn->prepare("SELECT id, nama, nim, jurusan FROM mahasiswa WHERE id=?");

$stmt->bind\_param("i", $id);

$stmt->execute();

$hasil = $stmt->get\_result()->fetch\_assoc();

if (!$hasil) {

die("Data tidak ditemukan");

}

?>

<!doctype html>

<html lang="id">

<head>

<meta charset="utf-8">

<title>Edit Mahasiswa</title>

<style>

body{font-family:system-ui,-apple-system,Segoe UI,Roboto,Arial,sans-serif;max-width:720px;margin:24px auto;padding:0 16px}

input,button{padding:10px;border:1px solid #ccc;border-radius:8px;width:100%}

button{cursor:pointer}

.msg{background:#fdecea;border:1px solid #f5c6cb;padding:10px;border-radius:8px;margin-bottom:12px}

</style>

</head>

<body>

<h1>Edit Data</h1>

<?php if (!empty($err)): ?>

<div class="msg"><?= htmlspecialchars($err) ?></div>

<?php endif; ?>

<form method="post">

<input type="hidden" name="aksi" value="update">

<p>

<label>Nama</label><br>

<input type="text" name="nama" value="<?= htmlspecialchars($hasil['nama']) ?>" required>

</p>

<p>

<label>NIM</label><br>

<input type="text" name="nim" value="<?= htmlspecialchars($hasil['nim']) ?>" required>

</p>

<p>

<label>Jurusan</label><br>

<input type="text" name="jurusan" value="<?= htmlspecialchars($hasil['jurusan']) ?>" required>

</p>

<p>

<button type="submit">Update</button>

<a href="index.php" style="margin-left:8px">Kembali</a>

</p>

</form>

</body>

</html>

**5) delete.php – hapus data**

<?php

require\_once "db.php";

$id = isset($\_GET['id']) ? (int) $\_GET['id'] : 0;

if ($id <= 0) {

die("ID tidak valid");

}

try {

$stmt = $conn->prepare("DELETE FROM mahasiswa WHERE id=?");

$stmt->bind\_param("i", $id);

$stmt->execute();

header("Location: index.php?msg=Data+berhasil+dihapus");

exit;

} catch (mysqli\_sql\_exception $e) {

die("Gagal hapus: " . $e->getMessage());

}

## Apa Itu Prepare() Param() Execute()

$conn->prepare($sql) adalah cara membuat **prepared statement** di MySQLi.  
Prepared statement = query SQL yang **dipersiapkan lebih dulu**, lalu **diisi datanya belakangan**.

Kenapa dipakai?

* **Lebih aman** → terhindar dari SQL Injection (kalau kita langsung tempel variabel ke query rawan disusupi karakter berbahaya).
* **Lebih efisien** → query bisa dipakai berulang dengan nilai berbeda tanpa disusun ulang.

**Alurnya bagaimana?**

Contoh query biasa (tidak aman):

$sql = "SELECT \* FROM users WHERE id = $id";

$result = $conn->query($sql);

👉 Kalau $id diisi '1 OR 1=1' bisa bobol semua data.

Kalau pakai **prepare + bind\_param**:

$stmt = $conn->prepare("SELECT \* FROM users WHERE id = ?");

$stmt->bind\_param("i", $id); // "i" = integer

$stmt->execute();

$result = $stmt->get\_result();

Di sini:

* ? = **placeholder** (tempat sementara)
* bind\_param("i", $id) = isi placeholder ? dengan nilai dari $id
* execute() = jalankan query dengan nilai yang sudah diikat

👉 Jadi MySQL hanya melihat query SELECT \* FROM users WHERE id = ?  
Lalu angka dimasukkan **terpisah**, sehingga lebih aman.

**Balik ke kode**

$stmt\_last = $conn->prepare($sql\_last);

$stmt\_last->bind\_param("ss", $bulan, $tahun);

$stmt\_last->execute();

$result\_last = $stmt\_last->get\_result();

Artinya:

1. **prepare($sql\_last)** → siapkan query yang punya ?
2. **bind\_param("ss", $bulan, $tahun)** → isi ? pertama dengan $bulan dan ? kedua dengan $tahun.
   * "ss" = keduanya dianggap string (s = string).
   * Kalau integer → pakai "i".
3. **execute()** → jalankan query dengan nilai yang sudah dimasukkan.
4. **get\_result()** → ambil hasil query.

**🔹 4. Ilustrasi sederhana**

Misal query:

SELECT \* FROM anakyatim WHERE bulan=? AND tahun=?

Kalau $bulan = 9 dan $tahun = 2025 → hasil akhirnya jadi:

SELECT \* FROM anakyatim WHERE bulan='9' AND tahun='2025'

🔑 **Kesimpulan singkat**

* prepare() = siapkan query dengan ? (placeholder).
* bind\_param() = isi nilai ke dalam ?.
* execute() = jalankan query.

Contoh **mini script insert data** pakai prepare(), biar langsung bisa dipahami bedanya dengan query biasa.

**🔹 Contoh tabel**

Misalnya kita punya tabel sederhana:

CREATE TABLE anakyatim (

id INT AUTO\_INCREMENT PRIMARY KEY,

nama VARCHAR(100),

alamat VARCHAR(200),

tgl\_lahir DATE

);

**🔹 Contoh Insert TANPA prepare() (raw query)**

<?php

require "db.php";

$nama = $\_POST['nama'];

$alamat = $\_POST['alamat'];

$tgl\_lahir= $\_POST['tgl\_lahir'];

$sql = "INSERT INTO anakyatim (nama, alamat, tgl\_lahir)

VALUES ('$nama', '$alamat', '$tgl\_lahir')";

if ($conn->query($sql)) {

echo "Data berhasil disimpan";

} else {

echo "Error: " . $conn->error;

}

?>

**🔹 Contoh Insert DENGAN prepare() param() execute()**

<?php

require "db.php";

$nama = $\_POST['nama'];

$alamat = $\_POST['alamat'];

$tgl\_lahir= $\_POST['tgl\_lahir'];

// 1. Siapkan query dengan tanda tanya (?) sebagai placeholder

$stmt = $conn->prepare("INSERT INTO anakyatim (nama, alamat, tgl\_lahir) VALUES (?, ?, ?)");

// 2. Isi placeholder dengan nilai variabel

// "sss" artinya 3 parameter, semuanya tipe string

$stmt->bind\_param("sss", $nama, $alamat, $tgl\_lahir);

// 3. Jalankan query

if ($stmt->execute()) {

echo "Data berhasil disimpan";

} else {

echo "Error: " . $stmt->error;

}

// 4. Tutup statement

$stmt->close();

$conn->close();

?>

**🔹 Penjelasan alurnya**

1. **prepare()** → buat template query dengan ?
2. INSERT INTO anakyatim (nama, alamat, tgl\_lahir) VALUES (?, ?, ?)
3. **bind\_param("sss", $nama, $alamat, $tgl\_lahir)**
   * "sss" = tiga parameter bertipe **string** (s)
   * kalau integer → "i", double → "d", blob → "b".
4. **execute()** → jalankan query dengan nilai yang sudah diikat.

👉 Aman dari SQL Injection, karena MySQL hanya menerima data murni, tidak langsung menempel di query.

contoh **SELECT data** dengan prepare().  
Bedanya sama INSERT tadi, di sini kita ambil data dari tabel pakai kondisi WHERE.

**🔹 Misal tabelnya**

CREATE TABLE anakyatim (

id INT AUTO\_INCREMENT PRIMARY KEY,

nama VARCHAR(100),

alamat VARCHAR(200),

tgl\_lahir DATE

);

**🔹 Contoh SELECT TANPA prepare()**

<?php

require "db.php";

$id = $\_GET['id']; // data dari URL, misal detail.php?id=5

$sql = "SELECT \* FROM anakyatim WHERE id = $id";

$result = $conn->query($sql);

if ($result && $result->num\_rows > 0) {

$row = $result->fetch\_assoc();

echo "Nama: " . $row['nama'] . "<br>";

echo "Alamat: " . $row['alamat'] . "<br>";

} else {

echo "Data tidak ditemukan";

}

?>

👉 Masalah: kalau ada yang nakal kasih URL detail.php?id=5 OR 1=1 bisa keluar semua data (SQL Injection).

**🔹 Contoh SELECT DENGAN prepare()**

<?php

require "db.php";

$id = $\_GET['id']; // contoh input dari URL

// 1. Siapkan query dengan placeholder (?)

$stmt = $conn->prepare("SELECT \* FROM anakyatim WHERE id = ?");

// 2. Ikat parameter ke placeholder

// "i" = integer (karena id berupa angka)

$stmt->bind\_param("i", $id);

// 3. Jalankan query

$stmt->execute();

// 4. Ambil hasil

$result = $stmt->get\_result();

if ($result && $result->num\_rows > 0) {

$row = $result->fetch\_assoc();

echo "Nama: " . $row['nama'] . "<br>";

echo "Alamat: " . $row['alamat'] . "<br>";

} else {

echo "Data tidak ditemukan";

}

// 5. Tutup statement

$stmt->close();

$conn->close();

?>

**🔹 Penjelasan**

1. prepare("SELECT \* FROM anakyatim WHERE id = ?")  
   → bikin query dengan tanda ? (placeholder).
2. bind\_param("i", $id)
   * "i" = integer (kalau string → "s").
   * isi ? dengan nilai $id.
3. execute()  
   → jalankan query.
4. get\_result()  
   → ambil hasil query dalam bentuk **mysqli\_result**.

⚡ Jadi kalau user nakal kasih input "5 OR 1=1", MySQL tetap baca sebagai angka biasa, bukan bagian dari query. **Inilah kelebihan prepared statement.**

**UPDATE** dan **DELETE** dengan prepare() juga. Jadi nanti Bapak punya contoh komplit: **INSERT – SELECT – UPDATE – DELETE** semuanya pakai prepared statement.

**🔹 UPDATE data**

Misalnya kita mau update nama & alamat anak yatim berdasarkan id:

<?php

require "db.php";

$id = $\_POST['id'];

$nama = $\_POST['nama'];

$alamat = $\_POST['alamat'];

// 1. Siapkan query

$stmt = $conn->prepare("UPDATE anakyatim SET nama = ?, alamat = ? WHERE id = ?");

// 2. Ikat parameter ("ssi" = string, string, integer)

$stmt->bind\_param("ssi", $nama, $alamat, $id);

// 3. Eksekusi

if ($stmt->execute()) {

echo "Data berhasil diupdate";

} else {

echo "Error: " . $stmt->error;

}

$stmt->close();

$conn->close();

?>

👉 Kalau ada anak dengan id=5, maka akan diupdate sesuai input.

**🔹 DELETE data**

Misalnya kita mau hapus anak yatim berdasarkan id:

<?php

require "db.php";

$id = $\_GET['id']; // contoh ambil dari URL delete.php?id=5

// 1. Siapkan query

$stmt = $conn->prepare("DELETE FROM anakyatim WHERE id = ?");

// 2. Ikat parameter ("i" = integer)

$stmt->bind\_param("i", $id);

// 3. Eksekusi

if ($stmt->execute()) {

echo "Data berhasil dihapus";

} else {

echo "Error: " . $stmt->error;

}

$stmt->close();

$conn->close();

?>

👉 Kalau buka delete.php?id=5, maka data dengan id=5 akan terhapus.

**🔹 Kesimpulan CRUD dengan prepare()**

* **INSERT** → bind\_param("sss", ...) sesuai jumlah field.
* **SELECT** → bind\_param("i", $id) untuk ambil detail.
* **UPDATE** → bind\_param("ssi", ...) campuran string & integer.
* **DELETE** → bind\_param("i", $id) untuk hapus by ID.

Semua lebih aman dari **SQL Injection**, karena nilai tidak langsung ditempel di query, tapi dikirim terpisah ke MySQL.

Contoh **CRUD lengkap dalam 1 file** (pakai **MySQLi + prepared statement**) untuk tabel anakyatim. Tinggal sesuaikan db config di bagian atas. Sudah ada:

* List data
* Tambah (INSERT)
* Edit (SELECT + UPDATE)
* Hapus (DELETE)
* Validasi ringan & sanitasi output
* Bootstrap biar rapi

**1) Contoh struktur tabel**

Jalankan sekali di MySQL:

CREATE TABLE anakyatim (

id INT AUTO\_INCREMENT PRIMARY KEY,

nama VARCHAR(100) NOT NULL,

alamat VARCHAR(200) NOT NULL,

tgl\_lahir DATE NOT NULL

);

**2) Satu file: anakyatim.php**

Simpan sebagai anakyatim.php, lalu akses di browser.

<?php

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* anakyatim.php - CRUD 1 file (MySQLi)

\* by: Cak Andieks &

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

// === CONFIG KONEKSI ===

$DB\_HOST = "localhost";

$DB\_USER = "root";

$DB\_PASS = "";

$DB\_NAME = "dbname\_anda"; // ganti sesuai database Anda

$conn = new mysqli($DB\_HOST, $DB\_USER, $DB\_PASS, $DB\_NAME);

if ($conn->connect\_error) {

die("Koneksi gagal: " . $conn->connect\_error);

}

$conn->set\_charset("utf8mb4");

// Helper escape output HTML

function e($s) { return htmlspecialchars((string)$s, ENT\_QUOTES, 'UTF-8'); }

// Ambil action

$action = $\_GET['action'] ?? 'index';

$id = isset($\_GET['id']) ? (int)$\_GET['id'] : 0;

// Pesan notifikasi sederhana via query string

$flash = $\_GET['msg'] ?? "";

// ========================

// Aksi CREATE (simpan data)

// ========================

if ($action === 'store' && $\_SERVER['REQUEST\_METHOD'] === 'POST') {

$nama = trim($\_POST['nama'] ?? "");

$alamat = trim($\_POST['alamat'] ?? "");

$tglLahir = trim($\_POST['tgl\_lahir'] ?? "");

// Validasi sederhana

$errors = [];

if ($nama === "") $errors[] = "Nama wajib diisi.";

if ($alamat === "") $errors[] = "Alamat wajib diisi.";

if ($tglLahir === "") $errors[] = "Tanggal lahir wajib diisi.";

if (empty($errors)) {

$stmt = $conn->prepare("INSERT INTO anakyatim (nama, alamat, tgl\_lahir) VALUES (?, ?, ?)");

$stmt->bind\_param("sss", $nama, $alamat, $tglLahir);

if ($stmt->execute()) {

header("Location: anakyatim.php?msg=" . urlencode("Berhasil menambah data."));

exit;

} else {

$errors[] = "Gagal simpan: " . $stmt->error;

}

$stmt->close();

}

// Jika gagal, tampilkan form tambah lagi dengan error

$action = 'create';

}

// ========================

// Aksi UPDATE (simpan edit)

// ========================

if ($action === 'update' && $\_SERVER['REQUEST\_METHOD'] === 'POST') {

$id = (int)($\_POST['id'] ?? 0);

$nama = trim($\_POST['nama'] ?? "");

$alamat = trim($\_POST['alamat'] ?? "");

$tglLahir = trim($\_POST['tgl\_lahir'] ?? "");

$errors = [];

if ($id <= 0) $errors[] = "ID tidak valid.";

if ($nama === "") $errors[] = "Nama wajib diisi.";

if ($alamat === "") $errors[] = "Alamat wajib diisi.";

if ($tglLahir === "") $errors[] = "Tanggal lahir wajib diisi.";

if (empty($errors)) {

$stmt = $conn->prepare("UPDATE anakyatim SET nama=?, alamat=?, tgl\_lahir=? WHERE id=?");

$stmt->bind\_param("sssi", $nama, $alamat, $tglLahir, $id);

if ($stmt->execute()) {

header("Location: anakyatim.php?msg=" . urlencode("Berhasil mengubah data."));

exit;

} else {

$errors[] = "Gagal update: " . $stmt->error;

}

$stmt->close();

}

// Jika gagal, kembalikan ke form edit dengan error

$action = 'edit';

}

// ========================

// Aksi DELETE

// ========================

if ($action === 'delete' && $id > 0) {

$stmt = $conn->prepare("DELETE FROM anakyatim WHERE id=?");

$stmt->bind\_param("i", $id);

if ($stmt->execute()) {

header("Location: anakyatim.php?msg=" . urlencode("Berhasil menghapus data."));

exit;

} else {

$flash = "Gagal hapus: " . $stmt->error; // tampilkan di halaman index

}

$stmt->close();

$action = 'index';

}

// ========================

// Ambil data utk edit (jika action=edit)

// ========================

$editRow = null;

if ($action === 'edit' && $id > 0) {

$stmt = $conn->prepare("SELECT id, nama, alamat, tgl\_lahir FROM anakyatim WHERE id=?");

$stmt->bind\_param("i", $id);

$stmt->execute();

$res = $stmt->get\_result();

$editRow = $res->fetch\_assoc() ?: null;

$stmt->close();

if (!$editRow) {

$flash = "Data tidak ditemukan.";

$action = 'index';

}

}

// ========================

// View HTML

// ========================

?>

<!doctype html>

<html lang="id">

<head>

<meta charset="utf-8">

<title>CRUD Anak Yatim (MySQLi + Prepared)</title>

<meta name="viewport" content="width=device-width, initial-scale=1">

<link

href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.min.css"

rel="stylesheet">

</head>

<body class="bg-light">

<div class="container py-4">

<div class="d-flex justify-content-between align-items-center mb-3">

<h1 class="h3 m-0">Data Anak Yatim</h1>

<div>

<?php if ($action === 'index'): ?>

<a href="anakyatim.php?action=create" class="btn btn-success">+ Tambah</a>

<?php else: ?>

<a href="anakyatim.php" class="btn btn-secondary">← Kembali</a>

<?php endif; ?>

</div>

</div>

<?php if (!empty($flash)): ?>

<div class="alert alert-info"><?= e($flash) ?></div>

<?php endif; ?>

<?php if (!empty($errors ?? [])): ?>

<div class="alert alert-danger">

<ul class="mb-0">

<?php foreach ($errors as $er): ?>

<li><?= e($er) ?></li>

<?php endforeach; ?>

</ul>

</div>

<?php endif; ?>

<?php if ($action === 'index'): ?>

<?php

// Tampilkan list data

$rows = [];

$result = $conn->query("SELECT id, nama, alamat, tgl\_lahir FROM anakyatim ORDER BY id DESC");

if ($result && $result->num\_rows > 0) {

$rows = $result->fetch\_all(MYSQLI\_ASSOC);

}

?>

<div class="card">

<div class="card-body">

<div class="table-responsive">

<table class="table table-striped align-middle">

<thead>

<tr>

<th style="width:6rem;">ID</th>

<th>Nama</th>

<th>Alamat</th>

<th>Tgl Lahir</th>

<th style="width:10rem;">Aksi</th>

</tr>

</thead>

<tbody>

<?php if (empty($rows)): ?>

<tr>

<td colspan="5" class="text-center text-muted">Belum ada data.</td>

</tr>

<?php else: ?>

<?php foreach ($rows as $r): ?>

<tr>

<td><?= e($r['id']) ?></td>

<td><?= e($r['nama']) ?></td>

<td><?= e($r['alamat']) ?></td>

<td><?= e($r['tgl\_lahir']) ?></td>

<td>

<a href="anakyatim.php?action=edit&id=<?= (int)$r['id'] ?>" class="btn btn-sm btn-primary">Edit</a>

<a href="anakyatim.php?action=delete&id=<?= (int)$r['id'] ?>"

class="btn btn-sm btn-danger"

onclick="return confirm('Yakin data akan dihapus?');">

Hapus

</a>

</td>

</tr>

<?php endforeach; ?>

<?php endif; ?>

</tbody>

</table>

</div>

</div>

</div>

<?php elseif ($action === 'create'): ?>

<div class="card">

<div class="card-header">Tambah Data</div>

<div class="card-body">

<form method="post" action="anakyatim.php?action=store" novalidate>

<div class="mb-3">

<label class="form-label">Nama</label>

<input type="text" name="nama" class="form-control" required value="<?= e($\_POST['nama'] ?? '') ?>">

</div>

<div class="mb-3">

<label class="form-label">Alamat</label>

<textarea name="alamat" class="form-control" rows="2" required><?= e($\_POST['alamat'] ?? '') ?></textarea>

</div>

<div class="mb-3">

<label class="form-label">Tanggal Lahir</label>

<input type="date" name="tgl\_lahir" class="form-control" required value="<?= e($\_POST['tgl\_lahir'] ?? '') ?>">

</div>

<button class="btn btn-success">Simpan</button>

<a href="anakyatim.php" class="btn btn-secondary">Batal</a>

</form>

</div>

</div>

<?php elseif ($action === 'edit' && $editRow): ?>

<div class="card">

<div class="card-header">Edit Data</div>

<div class="card-body">

<form method="post" action="anakyatim.php?action=update" novalidate>

<input type="hidden" name="id" value="<?= (int)$editRow['id'] ?>">

<div class="mb-3">

<label class="form-label">Nama</label>

<input type="text" name="nama" class="form-control" required value="<?= e($\_POST['nama'] ?? $editRow['nama']) ?>">

</div>

<div class="mb-3">

<label class="form-label">Alamat</label>

<textarea name="alamat" class="form-control" rows="2" required><?= e($\_POST['alamat'] ?? $editRow['alamat']) ?></textarea>

</div>

<div class="mb-3">

<label class="form-label">Tanggal Lahir</label>

<input type="date" name="tgl\_lahir" class="form-control" required value="<?= e($\_POST['tgl\_lahir'] ?? $editRow['tgl\_lahir']) ?>">

</div>

<button class="btn btn-primary">Update</button>

<a href="anakyatim.php" class="btn btn-secondary">Batal</a>

</form>

</div>

</div>

<?php endif; ?>

<p class="text-muted small mt-3 mb-0">

Demo ini memakai <strong>prepared statements</strong> untuk semua operasi (INSERT, SELECT by id, UPDATE, DELETE) agar aman dari SQL Injection.

</p>

</div>

</body>

</html>

**Catatan penting**

* Pastikan mengganti dbname\_anda di config koneksi.
* Tombol **Hapus** sudah pakai onclick="return confirm(...)"
* Jika ingin menambahkan **ID khusus** seperti IDA-00001, kita bisa tambahkan field user\_id lalu buatkan generator saat INSERT. Nanti saya bisa sisipkan blok generator itu di file ini kalau Bapak mau.

**Kelebihan fetch\_assoc()**

✅ Lebih mudah dibaca karena pakai nama kolom.  
✅ Lebih aman dan jelas saat ada perubahan urutan kolom di tabel.  
✅ Menghindari kebingungan antara index angka.

**1) Tambah form filter (di bagian if ($action === 'index') sebelum query list)**

Letakkan **SEBELUM** kode yang mengambil data ($result = $conn->query(...)), ganti bagian index view seperti ini:

<?php if ($action === 'index'): ?>

<!-- FORM FILTER -->

<div class="card mb-3">

<div class="card-body">

<form method="get" class="row g-2">

<input type="hidden" name="action" value="index">

<div class="col-md-4">

<label class="form-label mb-1">Kata Kunci (Nama/Alamat)</label>

<input type="text" name="q" class="form-control" placeholder="mis: Andi / Sidoarjo"

value="<?= e($\_GET['q'] ?? '') ?>">

</div>

<div class="col-md-3">

<label class="form-label mb-1">Tgl Lahir (Dari)</label>

<input type="date" name="dari" class="form-control"

value="<?= e($\_GET['dari'] ?? '') ?>">

</div>

<div class="col-md-3">

<label class="form-label mb-1">Tgl Lahir (Sampai)</label>

<input type="date" name="sampai" class="form-control"

value="<?= e($\_GET['sampai'] ?? '') ?>">

</div>

<div class="col-md-2 d-flex align-items-end">

<button class="btn btn-primary w-100">Filter</button>

</div>

</form>

</div>

</div>

<?php

// ====== BUILD QUERY DENGAN FILTER (prepared) ======

$q = trim($\_GET['q'] ?? '');

$dari = trim($\_GET['dari'] ?? '');

$sampai = trim($\_GET['sampai'] ?? '');

$sql = "SELECT id, nama, alamat, tgl\_lahir FROM anakyatim WHERE 1=1";

$types = "";

$params = [];

// filter kata kunci ke nama/alamat

if ($q !== "") {

$sql .= " AND (nama LIKE ? OR alamat LIKE ?)";

$like = "%$q%";

$types .= "ss";

$params[] = &$like;

$params[] = &$like;

}

// filter tanggal lahir dari

if ($dari !== "") {

$sql .= " AND tgl\_lahir >= ?";

$types .= "s";

$params[] = &$dari;

}

// filter tanggal lahir sampai

if ($sampai !== "") {

$sql .= " AND tgl\_lahir <= ?";

$types .= "s";

$params[] = &$sampai;

}

$sql .= " ORDER BY id DESC";

$stmtList = $conn->prepare($sql);

if ($types !== "") {

// bind variadik

$bindArgs = array\_merge([$types], $params);

// php < 8: gunakan call\_user\_func\_array

call\_user\_func\_array([$stmtList, 'bind\_param'], $bindArgs);

}

$stmtList->execute();

$res = $stmtList->get\_result();

$rows = $res ? $res->fetch\_all(MYSQLI\_ASSOC) : [];

$stmtList->close();

?>

<div class="card">

<div class="card-body">

<div class="table-responsive">

<table class="table table-striped align-middle">

<thead>

<tr>

<th style="width:6rem;">ID</th>

<th>Nama</th>

<th>Alamat</th>

<th>Tgl Lahir</th>

<th style="width:10rem;">Aksi</th>

</tr>

</thead>

<tbody>

<?php if (empty($rows)): ?>

<tr>

<td colspan="5" class="text-center text-muted">Tidak ada data yang cocok.</td>

</tr>

<?php else: ?>

<?php foreach ($rows as $r): ?>

<tr>

<td><?= e($r['id']) ?></td>

<td><?= e($r['nama']) ?></td>

<td><?= e($r['alamat']) ?></td>

<td><?= e($r['tgl\_lahir']) ?></td>

<td>

<a href="anakyatim.php?action=edit&id=<?= (int)$r['id'] ?>" class="btn btn-sm btn-primary">Edit</a>

<a href="anakyatim.php?action=delete&id=<?= (int)$r['id'] ?>"

class="btn btn-sm btn-danger"

onclick="return confirm('Yakin data akan dihapus?');">Hapus</a>

</td>

</tr>

<?php endforeach; ?>

<?php endif; ?>

</tbody>

</table>

</div>

</div>

</div>

<?php elseif ($action === 'create'): ?>

Catatan: di atas saya **mengganti** blok pengambilan list yang sebelumnya pakai $conn->query(...) menjadi **prepared + filter dinamis**.

**2) (Opsional) Tambah index untuk performa**

Supaya pencarian cepat:

CREATE INDEX idx\_anakyatim\_nama ON anakyatim (nama);

CREATE INDEX idx\_anakyatim\_alamat ON anakyatim (alamat);

CREATE INDEX idx\_anakyatim\_tgl ON anakyatim (tgl\_lahir);

**3) Kalau nanti ada kolom pic & status**

Tinggal tambah di form + query:

// FORM tambahan

<select name="status" class="form-select">

<option value="">-- Semua Status --</option>

<option value="aktif" <?= (($\_GET['status']??'')==='aktif'?'selected':'') ?>>Aktif</option>

<option value="nonaktif" <?= (($\_GET['status']??'')==='nonaktif'?'selected':'') ?>>Nonaktif</option>

</select>

<input type="text" name="pic" class="form-control" value="<?= e($\_GET['pic'] ?? '') ?>" placeholder="PIC">

// QUERY tambahan

if (!empty($\_GET['status'])) {

$sql .= " AND status = ?";

$types .= "s";

$status = $\_GET['status'];

$params[] = &$status;

}

if (!empty($\_GET['pic'])) {

$sql .= " AND pic LIKE ?";

$types .= "s";

$picLike = "%".$\_GET['pic']."%";

$params[] = &$picLike;

}

## Fungsi Explode() Dan Str\_Pad()

**1. explode()**

👉 Fungsinya: **memecah string menjadi array** berdasarkan pemisah tertentu.

Dalam kode:

$parts = explode('/', $row\_last['id']);

Misalnya data terakhir di database adalah:

IDB/9/2025/00001

Maka:

$parts = ["IDB", "9", "2025", "00001"];

Lalu:

$last\_number = (int) end($parts);

* end($parts) → ambil elemen terakhir array = "00001"
* (int) → ubah string "00001" jadi angka 1.

📌 Jadi explode() dipakai untuk **mengambil bagian nomor urut** dari ID terakhir.

**2. str\_pad()**

👉 Fungsinya: **menambahkan padding (karakter tambahan) di depan atau belakang string**, sampai panjang tertentu.

Dalam kode:

$nomor\_urut = str\_pad($next\_number, 5, "0", STR\_PAD\_LEFT);

* $next\_number misalnya = 2
* str\_pad(2, 5, "0", STR\_PAD\_LEFT) artinya:
  + Panjang total harus 5
  + Isi dengan "0" di sebelah kiri (STR\_PAD\_LEFT)
  + Hasilnya: "00002"

📌 Jadi str\_pad() dipakai supaya format nomor urut **selalu 5 digit** (00001, 00002, 00003, dst).

**3. Hasil akhir kode**

Misalnya data terakhir = IDB/9/2025/00001, maka:

* explode() → ambil angka 1
* Tambah 1 → jadi 2
* str\_pad() → format jadi 00002
* Hasil akhir ID baru: IDB/9/2025/00002

## Perbedaan = ( Assign) dan .= (Concatenate) di PHP

**📌 Perbedaan = dan .= di PHP**

* = → **assign** (mengisi ulang nilai variabel, yang lama hilang/diganti)
* .= → **concatenate** (menambahkan teks di **belakang** nilai variabel yang sudah ada)

**🔧 Dalam kasusmu:**

$sql .= "WHERE no\_kwitansi LIKE ? OR nama\_pelanggan LIKE ? OR status\_bayar LIKE ? ";

Artinya:

* Ambil isi variabel $sql yang **sudah ada** (misalnya "SELECT \* FROM kwitansi ").
* Tambahkan string baru **di belakangnya**.
* Jadi nilai $sql berubah, tapi tidak menghapus yang sebelumnya.

**📝 Contoh sederhana**

$sql = "SELECT \* FROM kwitansi "; // isi awal

$sql .= "WHERE status\_bayar='Lunas' "; // ditambah

echo $sql;

**Output:**

SELECT \* FROM kwitansi WHERE status\_bayar='Lunas'

Kalau pakai = biasa:

$sql = "SELECT \* FROM kwitansi ";

$sql = "WHERE status\_bayar='Lunas' "; // isi awal ketimpa

echo $sql;

**Output:**

WHERE status\_bayar='Lunas'

➡ Bagian SELECT \* FROM kwitansi hilang.

**💡 Kesimpulan**

$sql .= ... digunakan karena kamu ingin **menyambung** query, bukan menimpa yang sudah ada.  
Ini biasa dipakai untuk membuat query dinamis, misalnya kalau ada pencarian (WHERE ...), filter tambahan, atau ORDER BY/LIMIT.

**contoh real biar gampang kebayang.**

Misalkan kode awal kamu begini:

$sql = "SELECT \* FROM kwitansi "; // query dasar

$search = "lunas"; // user ketik kata pencarian di form

if ($search !== "") {

$sql .= "WHERE no\_kwitansi LIKE ? OR nama\_pelanggan LIKE ? OR status\_bayar LIKE ? ";

$params = ["%$search%", "%$search%", "%$search%"];

$types = "sss";

}

**🔎 Kasus 1: $search ADA isinya (lunas)**

* Isi awal $sql → SELECT \* FROM kwitansi
* Setelah .= →

SELECT \* FROM kwitansi

WHERE no\_kwitansi LIKE ?

OR nama\_pelanggan LIKE ?

OR status\_bayar LIKE ?

Kemudian PHP akan **binding parameter** dari $params, sehingga nilai akhirnya kira-kira akan dieksekusi seperti ini:

SELECT \* FROM kwitansi

WHERE no\_kwitansi LIKE '%lunas%'

OR nama\_pelanggan LIKE '%lunas%'

OR status\_bayar LIKE '%lunas%'

Hasilnya → semua baris yang ada kata **lunas** di salah satu kolom akan diambil.

**🔎 Kasus 2: $search KOSONG**

$search = ""; // user tidak isi pencarian

Karena $search !== "" → **false**, maka blok if tidak dijalankan.

Maka isi $sql tetap:

SELECT \* FROM kwitansi

Artinya → ambil **semua data** tanpa filter.

**📝 Intinya:**

* **Tanpa .=** → query kamu akan hilang bagian awalnya (SELECT \* FROM kwitansi).
* **Dengan .=** → query awal tetap dipertahankan, lalu ditambahkan kondisi WHERE ... kalau ada pencarian.

**contoh lengkap query dinamis dengan WHERE, ORDER BY, dan LIMIT supaya kamu lihat alur penyambungan query pakai .= sampai tuntas.**

**🔧 Contoh Kode Lengkap**

$sql = "SELECT \* FROM kwitansi "; // query dasar

// ---- Filter Pencarian ----

if ($search !== "") {

$sql .= "WHERE no\_kwitansi LIKE ? OR nama\_pelanggan LIKE ? OR status\_bayar LIKE ? ";

$params = ["%$search%", "%$search%", "%$search%"];

$types = "sss";

}

// ---- Tambah Sorting ----

$sql .= "ORDER BY tanggal DESC "; // urutkan dari tanggal terbaru

// ---- Tambah Pagination ----

$sql .= "LIMIT ? OFFSET ? "; // batasi jumlah baris per halaman

$params[] = $limit; // jumlah baris per halaman

$params[] = $offset; // mulai dari baris ke berapa

$types .= "ii"; // i = integer untuk bind\_param

**🧠 Ilustrasi Hasil Query**

Misalkan:

* $search = "lunas"
* $limit = 10
* $offset = 20 (halaman ke-3 misalnya)

Maka query hasil akhirnya yang akan dieksekusi:

SELECT \* FROM kwitansi

WHERE no\_kwitansi LIKE '%lunas%'

OR nama\_pelanggan LIKE '%lunas%'

OR status\_bayar LIKE '%lunas%'

ORDER BY tanggal DESC

LIMIT 10 OFFSET 20

📌 Artinya:

* Cari semua data yang ada kata **lunas**.
* Urutkan dari tanggal terbaru.
* Ambil **10 data saja**, mulai dari data ke-21 (offset 20).

**🖼 Visual Alur Query**

| **Tahap** | **Isi $sql** |
| --- | --- |
| Awal | SELECT \* FROM kwitansi |
| Setelah WHERE | SELECT \* FROM kwitansi WHERE no\_kwitansi LIKE ? ... |
| Setelah ORDER BY | SELECT \* FROM kwitansi WHERE ... ORDER BY tanggal DESC |
| Setelah LIMIT/OFFSET | SELECT \* FROM kwitansi WHERE ... ORDER BY tanggal DESC LIMIT ? OFFSET ? |

Dengan cara ini, query kamu **fleksibel**:

* Kalau tidak ada $search, bagian WHERE tidak ditambahkan.
* Kamu bisa tambahkan filter tambahan lagi misalnya AND status\_bayar = 'Lunas' cukup pakai .= lagi.
* Pagination bisa diubah hanya dengan mengubah $limit dan $offset.

**Contoh Lengkap & Siap Pakai: Pencarian (Q), Pagination (Ps, Page), COUNT(\*) Untuk Total Data, ORDER BY, LIMIT/OFFSET, Plus Output Tabel & Navigasi Halaman. Struktur Mengikuti Gaya Kode Kamu Sebelumnya.**

<?php

require\_once "db.php";

// --- Konfigurasi paging ---

$allowed\_sizes = [5,10,20,50];

$ps = (int)($\_GET['ps'] ?? 10);

$limit = in\_array($ps, $allowed\_sizes) ? $ps : 10;

$page = max(1, (int)($\_GET['page'] ?? 1));

$offset = ($page - 1) \* $limit;

// --- Pencarian ---

$search = trim($\_GET['q'] ?? "");

// --- Susun query dasar ---

// (Gunakan SELECT \* agar preg\_replace COUNT(\*) di bawah tetap cocok)

$sql = "SELECT \* FROM kwitansi ";

$params = [];

$types = "";

// --- WHERE dinamis jika ada pencarian ---

if ($search !== "") {

$sql .= "WHERE no\_kwitansi LIKE ? OR nama\_pelanggan LIKE ? OR status\_bayar LIKE ? ";

$params = ["%{$search}%", "%{$search}%", "%{$search}%"];

$types = "sss";

}

// --- Hitung total baris untuk pagination ---

// Ubah 'SELECT \* FROM' menjadi 'SELECT COUNT(\*) as jml FROM' sekali saja (kemunculan pertama)

$sql\_count = preg\_replace('/SELECT \\* FROM/i', 'SELECT COUNT(\*) as jml FROM', $sql, 1);

$stmt = $conn->prepare($sql\_count);

if ($types) { $stmt->bind\_param($types, ...$params); }

$stmt->execute();

$count = $stmt->get\_result()->fetch\_assoc()['jml'] ?? 0;

$stmt->close();

$total\_pages = max(1, (int)ceil($count / $limit));

// --- Sorting + Pagination ---

$sql .= "ORDER BY tanggal DESC LIMIT ? OFFSET ? ";

$params[] = $limit;

$params[] = $offset;

$types .= "ii";

// --- Eksekusi query data ---

$stmt = $conn->prepare($sql);

$stmt->bind\_param($types, ...$params);

$stmt->execute();

$res = $stmt->get\_result();

$rows = $res ? $res->fetch\_all(MYSQLI\_ASSOC) : [];

$stmt->close();

// --- Helper untuk aman di HTML/URL ---

function h($s){ return htmlspecialchars((string)$s, ENT\_QUOTES, 'UTF-8'); }

function u($s){ return urlencode((string)$s); }

?>

<!doctype html>

<html lang="id">

<head>

<meta charset="utf-8">

<title>List Kwitansi</title>

<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.min.css" rel="stylesheet">

</head>

<body class="bg-light">

<div class="container py-4">

<h1 class="h4 mb-3">List Kwitansi</h1>

<!-- Form Pencarian & Page Size -->

<form class="row g-2 mb-3" method="get">

<div class="col-md-6">

<input type="text" class="form-control" name="q" placeholder="Cari no kwitansi/nama/status..."

value="<?= h($search) ?>">

</div>

<div class="col-auto">

<select name="ps" class="form-select" onchange="this.form.submit()">

<?php foreach ($allowed\_sizes as $s): ?>

<option value="<?= $s ?>" <?= $s===$limit?'selected':'' ?>><?= $s ?>/hal</option>

<?php endforeach; ?>

</select>

</div>

<div class="col-auto">

<button class="btn btn-primary">Cari</button>

<?php if ($search !== ""): ?>

<a class="btn btn-outline-secondary" href="?ps=<?= (int)$limit ?>">Reset</a>

<?php endif; ?>

</div>

</form>

<!-- Info ringkas -->

<div class="mb-2 small text-muted">

Total data: <strong><?= (int)$count ?></strong> • Halaman: <strong><?= (int)$page ?>/<?= (int)$total\_pages ?></strong>

</div>

<!-- Tabel Data -->

<div class="table-responsive">

<table class="table table-sm table-striped align-middle">

<thead class="table-light">

<tr>

<th>No</th>

<th>No Kwitansi</th>

<th>Nama Pelanggan</th>

<th>Tanggal</th>

<th>Status Bayar</th>

<th class="text-end">Total</th>

</tr>

</thead>

<tbody>

<?php if (!$rows): ?>

<tr><td colspan="6" class="text-center text-muted">Tidak ada data.</td></tr>

<?php else:

$no\_awal = $offset + 1;

foreach ($rows as $i => $r): ?>

<tr>

<td><?= (int)$no\_awal + $i ?></td>

<td><?= h($r['no\_kwitansi'] ?? '') ?></td>

<td><?= h($r['nama\_pelanggan'] ?? '') ?></td>

<td><?= h($r['tanggal'] ?? '') ?></td>

<td><?= h($r['status\_bayar'] ?? '') ?></td>

<td class="text-end"><?= number\_format((float)($r['total'] ?? 0), 0, ',', '.') ?></td>

</tr>

<?php endforeach; endif; ?>

</tbody>

</table>

</div>

<!-- Pagination -->

<nav aria-label="Page navigation">

<ul class="pagination pagination-sm">

<?php

// helper buat bikin URL dengan mempertahankan q & ps

$base = '?ps='. (int)$limit . '&q=' . u($search) . '&page=';

$prev = max(1, $page - 1);

$next = min($total\_pages, $page + 1);

?>

<li class="page-item <?= $page<=1?'disabled':'' ?>">

<a class="page-link" href="<?= $base . '1' ?>">« First</a>

</li>

<li class="page-item <?= $page<=1?'disabled':'' ?>">

<a class="page-link" href="<?= $base . $prev ?>">‹ Prev</a>

</li>

<?php

// tampilkan range halaman kecil (mis: current ±2)

$start = max(1, $page - 2);

$end = min($total\_pages, $page + 2);

for ($p=$start; $p<=$end; $p++): ?>

<li class="page-item <?= $p===$page?'active':'' ?>">

<a class="page-link" href="<?= $base . $p ?>"><?= $p ?></a>

</li>

<?php endfor; ?>

<li class="page-item <?= $page>=$total\_pages?'disabled':'' ?>">

<a class="page-link" href="<?= $base . $next ?>">Next ›</a>

</li>

<li class="page-item <?= $page>=$total\_pages?'disabled':'' ?>">

<a class="page-link" href="<?= $base . $total\_pages ?>">Last »</a>

</li>

</ul>

</nav>

</div>

</body>

</html>

**Catatan penting**

* **Kenapa SELECT \*?** Supaya regex preg\_replace('/SELECT \\* FROM/i', ...) bisa mengganti dengan aman ke COUNT(\*). (Kalau kamu memilih (SELECT id, tanggal ...), ubah regex-nya jadi lebih umum).
* **Keamanan & performa:** pakai prepare + bind\_param untuk cegah SQL injection; paginasi pakai LIMIT/OFFSET.
* **Regex case-insensitive:** /i agar tetap cocok meski kamu menulis select \* from.

Ini versi yang **lebih fleksibel**, bisa menangani SELECT kolom1, kolom2 FROM ... juga, tidak hanya SELECT \* FROM ....   
Kuncinya kita ganti **bagian SELECT ... FROM** dengan SELECT COUNT(\*) as jml FROM menggunakan regex yang lebih umum.

**🔧 Contoh Kode**

Ganti bagian preg\_replace di kode sebelumnya dengan ini:

// --- Hitung total baris untuk pagination ---

// Ganti apapun setelah SELECT sampai FROM dengan COUNT(\*)

$sql\_count = preg\_replace('/^SELECT\s.+?\sFROM/i', 'SELECT COUNT(\*) as jml FROM', $sql, 1);

**🧠 Penjelasan Regex**

* ^SELECT → harus mulai dari kata SELECT
* \s → spasi setelah SELECT
* .+? → ambil semua karakter **seperlunya saja** (lazy match)
* \sFROM → sampai ketemu kata FROM
* /i → **case-insensitive**, jadi cocok walaupun kamu tulis select kecil

**🧪 Contoh Kasus**

**1) Kalau $sql seperti ini:**

SELECT \* FROM kwitansi WHERE status\_bayar = 'Lunas'

Hasil $sql\_count jadi:

SELECT COUNT(\*) as jml FROM kwitansi WHERE status\_bayar = 'Lunas'

**2) Kalau $sql seperti ini:**

SELECT id, nama\_pelanggan, total FROM kwitansi

Hasil $sql\_count jadi:

SELECT COUNT(\*) as jml FROM kwitansi

✅ Jadi **tidak masalah** meskipun kamu memilih kolom tertentu, tetap bisa dihitung totalnya.

**Keuntungan Cara Ini**

✔ **Fleksibel** – bisa menangani SELECT \* maupun SELECT kolom1, kolom2  
✔ **Dinamis** – tetap mempertahankan WHERE ..., JOIN ..., dll  
✔ **Aman** – hanya ganti bagian SELECT, tidak mengganggu query lainnya

Mau saya satukan versi regex baru ini ke kode yang sebelumnya saya buatkan (yang sudah lengkap dengan pencarian, sorting, pagination) supaya kamu bisa langsung copy-paste tanpa ubah banyak?

**You said:**

boleh bosku

**ChatGPT said:**

Mantap bosku! 🔥  
Ini saya satukan semua jadi satu file **lengkap**, sudah pakai regex fleksibel yang bisa handle SELECT \* atau SELECT kolom1, kolom2 ... FROM ....  
Langsung siap pakai di halaman list kwitansi.

<?php

require\_once "db.php";

// --- Konfigurasi paging ---

$allowed\_sizes = [5,10,20,50];

$ps = (int)($\_GET['ps'] ?? 10);

$limit = in\_array($ps, $allowed\_sizes) ? $ps : 10;

$page = max(1, (int)($\_GET['page'] ?? 1));

$offset = ($page - 1) \* $limit;

// --- Pencarian ---

$search = trim($\_GET['q'] ?? "");

// --- Susun query dasar ---

$sql = "SELECT id, no\_kwitansi, nama\_pelanggan, tanggal, status\_bayar, total FROM kwitansi ";

$params = [];

$types = "";

// --- WHERE dinamis jika ada pencarian ---

if ($search !== "") {

$sql .= "WHERE no\_kwitansi LIKE ? OR nama\_pelanggan LIKE ? OR status\_bayar LIKE ? ";

$params = ["%{$search}%", "%{$search}%", "%{$search}%"];

$types = "sss";

}

// --- Hitung total baris untuk pagination ---

// Regex lebih fleksibel: ganti apapun setelah SELECT ... sampai FROM

$sql\_count = preg\_replace('/^SELECT\s.+?\sFROM/i', 'SELECT COUNT(\*) as jml FROM', $sql, 1);

$stmt = $conn->prepare($sql\_count);

if ($types) { $stmt->bind\_param($types, ...$params); }

$stmt->execute();

$count = $stmt->get\_result()->fetch\_assoc()['jml'] ?? 0;

$stmt->close();

$total\_pages = max(1, (int)ceil($count / $limit));

// --- Sorting + Pagination ---

$sql .= "ORDER BY tanggal DESC LIMIT ? OFFSET ? ";

$params[] = $limit;

$params[] = $offset;

$types .= "ii";

// --- Eksekusi query data ---

$stmt = $conn->prepare($sql);

$stmt->bind\_param($types, ...$params);

$stmt->execute();

$res = $stmt->get\_result();

$rows = $res ? $res->fetch\_all(MYSQLI\_ASSOC) : [];

$stmt->close();

// --- Helper untuk aman di HTML/URL ---

function h($s){ return htmlspecialchars((string)$s, ENT\_QUOTES, 'UTF-8'); }

function u($s){ return urlencode((string)$s); }

?>

<!doctype html>

<html lang="id">

<head>

<meta charset="utf-8">

<title>List Kwitansi</title>

<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.min.css" rel="stylesheet">

</head>

<body class="bg-light">

<div class="container py-4">

<h1 class="h4 mb-3">List Kwitansi</h1>

<!-- Form Pencarian & Page Size -->

<form class="row g-2 mb-3" method="get">

<div class="col-md-6">

<input type="text" class="form-control" name="q" placeholder="Cari no kwitansi/nama/status..."

value="<?= h($search) ?>">

</div>

<div class="col-auto">

<select name="ps" class="form-select" onchange="this.form.submit()">

<?php foreach ($allowed\_sizes as $s): ?>

<option value="<?= $s ?>" <?= $s===$limit?'selected':'' ?>><?= $s ?>/hal</option>

<?php endforeach; ?>

</select>

</div>

<div class="col-auto">

<button class="btn btn-primary">Cari</button>

<?php if ($search !== ""): ?>

<a class="btn btn-outline-secondary" href="?ps=<?= (int)$limit ?>">Reset</a>

<?php endif; ?>

</div>

</form>

<!-- Info ringkas -->

<div class="mb-2 small text-muted">

Total data: <strong><?= (int)$count ?></strong> • Halaman: <strong><?= (int)$page ?>/<?= (int)$total\_pages ?></strong>

</div>

<!-- Tabel Data -->

<div class="table-responsive">

<table class="table table-sm table-striped align-middle">

<thead class="table-light">

<tr>

<th>No</th>

<th>No Kwitansi</th>

<th>Nama Pelanggan</th>

<th>Tanggal</th>

<th>Status Bayar</th>

<th class="text-end">Total</th>

</tr>

</thead>

<tbody>

<?php if (!$rows): ?>

<tr><td colspan="6" class="text-center text-muted">Tidak ada data.</td></tr>

<?php else:

$no\_awal = $offset + 1;

foreach ($rows as $i => $r): ?>

<tr>

<td><?= (int)$no\_awal + $i ?></td>

<td><?= h($r['no\_kwitansi'] ?? '') ?></td>

<td><?= h($r['nama\_pelanggan'] ?? '') ?></td>

<td><?= h($r['tanggal'] ?? '') ?></td>

<td><?= h($r['status\_bayar'] ?? '') ?></td>

<td class="text-end"><?= number\_format((float)($r['total'] ?? 0), 0, ',', '.') ?></td>

</tr>

<?php endforeach; endif; ?>

</tbody>

</table>

</div>

<!-- Pagination -->

<nav aria-label="Page navigation">

<ul class="pagination pagination-sm">

<?php

$base = '?ps='. (int)$limit . '&q=' . u($search) . '&page=';

$prev = max(1, $page - 1);

$next = min($total\_pages, $page + 1);

?>

<li class="page-item <?= $page<=1?'disabled':'' ?>">

<a class="page-link" href="<?= $base . '1' ?>">« First</a>

</li>

<li class="page-item <?= $page<=1?'disabled':'' ?>">

<a class="page-link" href="<?= $base . $prev ?>">‹ Prev</a>

</li>

<?php

$start = max(1, $page - 2);

$end = min($total\_pages, $page + 2);

for ($p=$start; $p<=$end; $p++): ?>

<li class="page-item <?= $p===$page?'active':'' ?>">

<a class="page-link" href="<?= $base . $p ?>"><?= $p ?></a>

</li>

<?php endfor; ?>

<li class="page-item <?= $page>=$total\_pages?'disabled':'' ?>">

<a class="page-link" href="<?= $base . $next ?>">Next ›</a>

</li>

<li class="page-item <?= $page>=$total\_pages?'disabled':'' ?>">

<a class="page-link" href="<?= $base . $total\_pages ?>">Last »</a>

</li>

</ul>

</nav>

</div>

</body>

</html>

**🔑 Bedanya dengan Versi Sebelumnya**

✅ **Regex lebih aman & fleksibel** – bisa ganti SELECT kolom1, kolom2 FROM ... jadi COUNT(\*) dengan benar  
✅ **Masih mempertahankan WHERE, JOIN, GROUP BY** – tidak terpotong  
✅ **Pagination, filter, dan order by sudah siap pakai**

**Biar makin lancar, ini catatan ringkas yang sering kepakai:**

* .=  
  → menyambung string (bukan menimpa). Cocok buat bangun query dinamis.
* preg\_replace('/^SELECT\s.+?\sFROM/i', 'SELECT COUNT(\*) as jml FROM', $sql, 1)  
  → ubah bagian SELECT ... FROM jadi COUNT(\*) untuk hitung total data.  
  Jika di $sql ada spasi/komentar di depan, pakai: '/^\s\*SELECT\s.+?\sFROM/i'.
* bind\_param($types, ...$params)  
  → pastikan tipe tepat. Misal LIMIT/OFFSET = i (integer).  
  Contoh: $types .= 'ii'; $params[] = $limit; $params[] = $offset;
* get\_result()  
  → butuh ekstensi **mysqlnd**. Kalau server nggak support, gunakan bind\_result() + fetch().
* Urutan aman:
  1. Susun $sql dasar (+ WHERE jika ada)
  2. Buat $sql\_count (jangan ada ORDER BY/LIMIT di sini)
  3. Tambah ORDER BY + LIMIT/OFFSET ke $sql data
  4. prepare/bind/execute untuk count dan data
* Index yang disarankan (biar pencarian kenceng):
  1. no\_kwitansi, nama\_pelanggan, status\_bayar, tanggal.
* Catatan COUNT vs GROUP BY/DISTINCT:  
  Kalau nanti ada GROUP BY/DISTINCT, hasil COUNT(\*) mungkin beda makna.  
  Solusi amannya: bungkus query jadi subquery:

SELECT COUNT(\*) AS jml FROM ( <query\_tanpa\_ORDER\_LIMIT> ) AS t

# GITHUB.COM CREATE UPLOAD DAN DOWNLOAD

Untuk meng-upload code ke GitHub, kamu perlu beberapa langkah. Berikut panduan step-by-step yang bisa kamu ikuti, mulai dari awal sampai code kamu berhasil masuk repository GitHub.

## Persiapan Awal

**1. Persiapan Awal**

1. **Buat akun GitHub** (jika belum):  
   Daftar di <https://github.com>.
2. **Install Git di komputer**
   * Download dari <https://git-scm.com>.
   * Install dengan pilihan default (biarkan setting bawaannya).
3. **Buat repository baru di GitHub**
   * Klik tombol **New Repository** di GitHub.
   * Beri nama repository (misalnya belajar-php).
   * Pilih **Public** (atau Private jika tidak ingin publik bisa lihat).
   * Klik **Create repository**.

## Urutan langkah-langkah lengkap untuk Pertama kali GIT:

1. **Upload Pertama Kali** (ketika repository masih kosong)
2. **Update Berikutnya** (setelah ada perubahan di project saat development)

Saya susun supaya bisa jadi SOP pribadi kamu.

**📌 Urutan Langkah Upload ke GitHub (Pertama Kali)**

| **Langkah** | **Perintah di Console** | **Keterangan** |
| --- | --- | --- |
| 1 | git init | Membuat repository Git di folder project. |
| 2 | git add . | Tambahkan semua file ke staging area. |
| 3 | git commit -m "First commit" | Buat commit pertama dengan pesan. |
| 4 | git branch -M main | (Opsional) Ganti nama branch jadi main agar sama dengan GitHub. |
| 5 | git remote add origin https://github.com/username/nama-repo.git | Hubungkan repository lokal ke repository GitHub. |
| 6 | git push -u origin main | Upload (push) semua file ke GitHub, sekaligus menghubungkan branch lokal dengan branch di GitHub. |

**🔄 Urutan Langkah Update Setelah Ada Perubahan (Development)**

| **Langkah** | **Perintah di Console** | **Keterangan** |
| --- | --- | --- |
| 1 | git status | Cek file yang berubah. |
| 2 | git add . | Masukkan semua perubahan ke staging area. (Atau gunakan git add <file> kalau hanya file tertentu). |
| 3 | git commit -m "Deskripsi perubahan" | Simpan perubahan ke repository lokal. |
| 4 | git push | Upload commit terbaru ke GitHub. |

**🏷️ Tips Tambahan**

* **Selalu cek status** dengan git status sebelum commit, supaya tahu file mana saja yang akan ikut.
* **Deskripsi commit harus jelas** (contoh: "Tambah fitur cetak kwitansi", "Perbaiki bug di save\_kwitansi.php").
* Jika ada orang lain yang kerja bareng di repo yang sama, lakukan git pull sebelum mulai coding supaya repo lokalmu up-to-date.
* Kalau pindah laptop atau komputer baru, cukup git clone <url-repo> lalu langsung mulai kerja.

## Tabel catatan perintah Git

yang paling sering dipakai untuk kerja sehari-hari. Kamu bisa simpan ini sebagai catatan, print, atau jadikan file .md / .txt supaya gampang dicari.

**📌 Tabel Catatan Perintah Git (Console)**

| **Perintah** | **Fungsi / Keterangan** |
| --- | --- |
| git init | Membuat repository Git baru di folder lokal (hanya sekali di awal project). |
| git status | Melihat status repository: file baru, file berubah, file siap di-commit. |
| git add . | Menambahkan semua perubahan ke staging area (siap di-commit). |
| git add <nama\_file> | Menambahkan hanya 1 file tertentu ke staging area. |
| git commit -m "Pesan commit" | Menyimpan perubahan ke repository lokal dengan pesan commit. |
| git branch | Melihat daftar branch dan branch yang aktif sekarang. |
| git branch -M main | Mengganti nama branch aktif menjadi main (dipakai di awal kalau perlu). |
| git remote add origin <url-repo> | Menghubungkan repository lokal dengan repository GitHub (hanya sekali). |
| git push -u origin main | Upload (push) branch main pertama kali ke GitHub. |
| git push | Upload commit terbaru ke GitHub (setelah -u pertama kali, cukup ini saja). |
| git pull | Mengambil update dari GitHub dan menggabungkannya ke branch lokal. |
| git log --oneline | Melihat riwayat commit dalam bentuk singkat. |
| git diff | Melihat detail perbedaan file yang sudah diubah tapi belum di-add. |
| git clone <url-repo> | Mengunduh repository dari GitHub ke komputer lokal. |
| git checkout -b <nama\_branch> | Membuat dan langsung pindah ke branch baru. |
| git checkout <nama\_branch> | Pindah ke branch lain. |
| git merge <nama\_branch> | Menggabungkan branch tertentu ke branch aktif sekarang. |
| git rm <nama\_file> | Menghapus file dari repo (lalu commit dan push). |
| git reset --hard | Mengembalikan semua file ke kondisi commit terakhir (hati-hati, perubahan hilang!). |

**🔄 Alur Upload Perubahan ke GitHub**

[Ubah File di Laptop]

│

▼

git status

(cek file yang berubah)

│

▼

git add .

(masukkan file ke staging area)

│

▼

git commit -m "pesan perubahan"

(simpan ke repo lokal)

│

▼

git push

(upload ke GitHub)

│

▼

[Selesai → File muncul di GitHub]

**📝 Analogi Biar Gampang**

* **Edit file** → seperti **ambil barang dari rak**.
* **git add .** → seperti **masukkan barang ke keranjang belanja**.
* **git commit -m "pesan"** → seperti **bayar di kasir (simpan transaksi ke catatan toko)**.
* **git push** → seperti **kirim barang ke gudang pusat (GitHub)**.

## Cara download / ambil aplikasi dari GitHub ke laptop

istilahnya di Git adalah **clone repository**.  
Dengan clone, kamu dapat salinan penuh dari repository (beserta history commit-nya) ke laptopmu.

Berikut langkah-langkahnya step-by-step:

**🖥️ Cara Download (Clone) Repository dari GitHub**

| **Langkah** | **Perintah di Console** | **Keterangan** |
| --- | --- | --- |
| 1 | **Copy URL repository dari GitHub** | Klik tombol **Code** di halaman repo → pilih HTTPS (contoh: https://github.com/user/nama-repo.git). |
| 2 | Buka **CMD / Terminal** di folder tujuan | Pindah ke folder tempat kamu mau menyimpan project (contoh: D:\xampp\htdocs\). |
| 3 | Jalankan perintah clone | bash<br>git clone https://github.com/user/nama-repo.git<br> |
| 4 | Masuk ke folder project | bash<br>cd nama-repo<br> |
| 5 | Jalankan / buka project sesuai kebutuhan | Misalnya kalau itu aplikasi PHP, taruh di htdocs XAMPP lalu akses via browser. |

**📝 Contoh Nyata**

Misalnya kamu mau clone repo kwitansiku kamu sendiri:

cd D:\xampp\htdocs

git clone https://github.com/andiekqtm2010-hash/kwitansiku.git

cd kwitansiku

Setelah itu kamu akan melihat semua file (index.php, db.php, dll) ada di folder kwitansiku.

**🔄 Update Jika Repo Ada Perubahan**

Kalau kamu sudah clone sebelumnya, tidak perlu clone ulang.  
Cukup masuk ke folder project, lalu jalankan:

git pull

Ini akan **download perubahan terbaru saja**, bukan semua file dari awal.

**📌 Catatan**

* **Clone hanya sekali**. Selanjutnya cukup git pull untuk update.
* Kalau repo **private**, pastikan kamu sudah login GitHub di laptop (atau masukkan username & password / token saat diminta).
* Pastikan Git sudah terinstall di laptop (cek dengan git --version di CMD).

# Sisipkan PHP pada Body HTML

Kalau mau **nyelipin kode PHP** ke dalam bagian <body> HTML, ada beberapa hal penting yang perlu diperhatikan biar rapi dan aman.

**🔹 1. Penempatan Tag PHP**

* Gunakan tag PHP <?php ... ?> untuk menyisipkan kode.
* Contoh di dalam <body>:

<body>

<h1>Daftar Kwitansi</h1>

<?php

echo "<p>Hari ini: " . date("d-m-Y") . "</p>";

?>

</body>

👉 Aturan: Semua kode PHP harus ditaruh **di dalam <?php ... ?>**.

**🔹 2. Jangan Campur Echo Berlebihan**

Kalau output HTML banyak, lebih baik **tutup tag PHP** dulu daripada echo panjang-panjang.

❌ Kurang rapi:

<?php

echo "<table>";

echo "<tr><td>Nama</td></tr>";

echo "</table>";

?>

✅ Lebih rapi:

<table>

<tr><td><?php echo $nama; ?></td></tr>

</table>

**🔹 3. Pastikan Variabel Sudah Ada**

Kalau kamu pakai variabel di dalam <body>, pastikan nilainya **sudah didefinisikan sebelumnya**.  
Misalnya:

<?php $nama = "Andiek"; ?>

<body>

<p>Halo, <?php echo $nama; ?></p>

</body>

Kalau variabel belum ada → bisa muncul **Notice: Undefined variable**.

**🔹 4. Perhatikan Mixing PHP & HTML**

Kalau datanya dalam bentuk **loop**, lebih enak pakai cara campur PHP–HTML:

<table border="1">

<tr><th>No</th><th>Nama</th></tr>

<?php

$no = 1;

foreach ($pelanggan as $p): ?>

<tr>

<td><?= $no++; ?></td>

<td><?= htmlspecialchars($p); ?></td>

</tr>

<?php endforeach; ?>

</table>

**🔹 5. Gunakan htmlspecialchars() untuk Data User**

Supaya aman dari XSS (script disuntikkan lewat input user).

<td><?= htmlspecialchars($nama\_pelanggan); ?></td>

**🔹 6. Struktur Lebih Rapi dengan Include**

Kalau <body> panjang banget, biasanya dipisah pakai include atau require.

<body>

<?php include "header.php"; ?>

<div class="content">

<?php include "content.php"; ?>

</div>

<?php include "footer.php"; ?>

</body>

👉 Jadi intinya, yang perlu diperhatikan:

1. Tag PHP ditutup–dibuka dengan benar.
2. Variabel sudah ada sebelum dipakai.
3. Jangan echo HTML panjang-panjang → lebih baik campur HTML–PHP.
4. Sanitasi output (htmlspecialchars).
5. Atur struktur (pakai include) kalau kode panjang.

Contoh Kode langsung:  
contoh **1 file kwitansi.php** yang lengkap dan rapi: ada form cari (q), page size (ps), paging, tabel hasil, dan **SUM total** sesuai filter. Sudah pakai **prepared statement**, htmlspecialchars(), dan pola campur PHP–HTML di <body> biar enak dibaca.

<?php

// kwitansi.php

// --- Setup dasar ---

date\_default\_timezone\_set('Asia/Jakarta');

require\_once "db.php"; // pastikan $conn = new mysqli(...)

// --- Page size dropdown ---

$allowed\_sizes = [5,10,20,50];

$ps = (int)($\_GET['ps'] ?? 10);

$limit = in\_array($ps, $allowed\_sizes) ? $ps : 10;

// --- Paging ---

$page = max(1, (int)($\_GET['page'] ?? 1));

$offset = ($page - 1) \* $limit;

// --- Pencarian ---

$search\_raw = $\_GET['q'] ?? "";

$search = trim($search\_raw);

// --- Susun WHERE dinamis ---

$where = [];

$params = [];

$types = "";

if ($search !== "") {

$where[] = "(no\_kwitansi LIKE ? OR nama\_pelanggan LIKE ? OR status\_bayar LIKE ?)";

$like = "%{$search}%";

$params = [$like, $like, $like];

$types = "sss";

}

$where\_sql = $where ? ("WHERE " . implode(" AND ", $where)) : "";

// --- Query data (paged) ---

$sql\_data = "SELECT id, no\_kwitansi, nama\_pelanggan, tanggal, total, status\_bayar

FROM kwitansi

$where\_sql

ORDER BY tanggal DESC, id DESC

LIMIT ? OFFSET ?";

$stmt = $conn->prepare($sql\_data);

// binding dinamis: tambahkan 2 integer di akhir untuk limit & offset

if ($types) {

$types\_data = $types . "ii";

$params\_data = [...$params, $limit, $offset];

$stmt->bind\_param($types\_data, ...$params\_data);

} else {

$stmt->bind\_param("ii", $limit, $offset);

}

$stmt->execute();

$result = $stmt->get\_result();

$rows = $result->fetch\_all(MYSQLI\_ASSOC);

$stmt->close();

// --- Hitung total baris (untuk paging) ---

$sql\_count = "SELECT COUNT(\*) AS jml FROM kwitansi $where\_sql";

$stmt = $conn->prepare($sql\_count);

if ($types) { $stmt->bind\_param($types, ...$params); }

$stmt->execute();

$count = $stmt->get\_result()->fetch\_assoc()['jml'] ?? 0;

$stmt->close();

$total\_pages = max(1, (int)ceil($count / $limit));

// --- Grand total sesuai filter (SUM) ---

$sql\_sum = "SELECT SUM(total) AS gtotal FROM kwitansi $where\_sql";

$stmt = $conn->prepare($sql\_sum);

if ($types) { $stmt->bind\_param($types, ...$params); }

$stmt->execute();

$grand\_total = (float)($stmt->get\_result()->fetch\_assoc()['gtotal'] ?? 0);

$stmt->close();

// --- Helper: bikin URL paging sambil bawa q & ps ---

function build\_url($page, $ps, $q) {

$qs = [

'page' => $page,

'ps' => $ps,

];

if (trim($q) !== "") $qs['q'] = $q;

return '?' . http\_build\_query($qs);

}

?>

<!doctype html>

<html lang="id">

<head>

<meta charset="utf-8">

<title>Daftar Kwitansi</title>

<meta name="viewport" content="width=device-width,initial-scale=1">

<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.min.css" rel="stylesheet">

</head>

<body class="bg-light">

<div class="container py-4">

<div class="d-flex justify-content-between align-items-center mb-3">

<h3 class="m-0">Daftar Kwitansi</h3>

<small class="text-muted">Hari ini: <?= date('d-m-Y H:i') ?> WIB</small>

</div>

<!-- Form filter & page size -->

<form class="row g-2 align-items-end mb-3" method="get" action="">

<div class="col-sm-6 col-md-5">

<label class="form-label">Cari (No, Nama, Status)</label>

<input type="text" name="q" class="form-control"

value="<?= htmlspecialchars($search\_raw) ?>"

placeholder="mis. Budi / Lunas / KW-001">

</div>

<div class="col-sm-3 col-md-2">

<label class="form-label">Baris/halaman</label>

<select name="ps" class="form-select">

<?php foreach ($allowed\_sizes as $size): ?>

<option value="<?= $size ?>" <?= $ps===$size?'selected':'' ?>><?= $size ?></option>

<?php endforeach; ?>

</select>

</div>

<div class="col-sm-3 col-md-2">

<button class="btn btn-primary w-100">Terapkan</button>

</div>

<?php if ($search !== ""): ?>

<div class="col-12">

<a class="small link-secondary" href="<?= build\_url(1, $ps, "") ?>">Reset filter</a>

</div>

<?php endif; ?>

</form>

<!-- Info ringkas -->

<div class="d-flex justify-content-between align-items-center mb-2">

<div>

<span class="badge bg-secondary">Total data: <?= number\_format($count,0,',','.') ?></span>

<span class="badge bg-info text-dark">Hal: <?= $page ?>/<?= $total\_pages ?></span>

</div>

<div class="fw-semibold">

Grand Total (sesuai filter): Rp <?= number\_format($grand\_total,0,',','.') ?>

</div>

</div>

<!-- Tabel hasil -->

<div class="card shadow-sm">

<div class="table-responsive">

<table class="table table-striped table-hover mb-0 align-middle">

<thead class="table-light">

<tr>

<th style="width:60px">#</th>

<th>No Kwitansi</th>

<th>Nama Pelanggan</th>

<th>Tanggal</th>

<th class="text-end">Total</th>

<th>Status</th>

</tr>

</thead>

<tbody>

<?php if (!$rows): ?>

<tr><td colspan="6" class="text-center text-muted py-4">Tidak ada data</td></tr>

<?php else: ?>

<?php

$no = $offset + 1;

$subtotal\_page = 0;

foreach ($rows as $r):

$subtotal\_page += (float)$r['total'];

?>

<tr>

<td><?= $no++ ?></td>

<td><?= htmlspecialchars($r['no\_kwitansi']) ?></td>

<td><?= htmlspecialchars($r['nama\_pelanggan']) ?></td>

<td><?= htmlspecialchars(date('d-m-Y', strtotime($r['tanggal']))) ?></td>

<td class="text-end">Rp <?= number\_format($r['total'],0,',','.') ?></td>

<td>

<span class="badge <?= strtolower($r['status\_bayar'])==='lunas'?'bg-success':'bg-warning text-dark' ?>">

<?= htmlspecialchars($r['status\_bayar']) ?>

</span>

</td>

</tr>

<?php endforeach; ?>

<!-- subtotal halaman -->

<tr class="table-light">

<td colspan="4" class="text-end fw-semibold">Subtotal halaman ini</td>

<td class="text-end fw-semibold">Rp <?= number\_format($subtotal\_page,0,',','.') ?></td>

<td></td>

</tr>

<?php endif; ?>

</tbody>

</table>

</div>

</div>

<!-- Pagination -->

<nav class="mt-3">

<ul class="pagination pagination-sm mb-0">

<li class="page-item <?= $page<=1?'disabled':'' ?>">

<a class="page-link" href="<?= build\_url(max(1,$page-1), $ps, $search\_raw) ?>">« Prev</a>

</li>

<?php

// tampilkan range halaman yang wajar

$start = max(1, $page - 2);

$end = min($total\_pages, $page + 2);

for ($i=$start; $i<=$end; $i++):

?>

<li class="page-item <?= $i===$page?'active':'' ?>">

<a class="page-link" href="<?= build\_url($i, $ps, $search\_raw) ?>"><?= $i ?></a>

</li>

<?php endfor; ?>

<li class="page-item <?= $page>=$total\_pages?'disabled':'' ?>">

<a class="page-link" href="<?= build\_url(min($total\_pages,$page+1), $ps, $search\_raw) ?>">Next »</a>

</li>

</ul>

</nav>

</div>

</body>

</html>

**Catatan cepat:**

* **Campur PHP–HTML di body**: loop tabel pakai <?php foreach ... ?> lalu echo kolom dengan <?= ... ?>.
* **Aman XSS**: semua teks dari DB/GET di-output lewat htmlspecialchars(...).
* **Prepared statements**: semua filter LIKE via bind param (%{$search}%).
* **Paging rapi**: COUNT(\*) buat total baris, lalu LIMIT/OFFSET untuk data.
* **Grand Total**: SUM(total) mengikuti filter yang sama.

# COALESCE (SQL)

**Apa itu?**  
Fungsi SQL yang mengembalikan **nilai pertama yang bukan NULL** dari daftar argumen.

**Bentuk umum:**

COALESCE(expr1, expr2, expr3, ...)

**Contoh di kasusmu (SUM bisa NULL kalau tidak ada baris cocok):**

SELECT COALESCE(SUM(total), 0) AS gtotal

FROM kwitansi

WHERE nama\_pelanggan LIKE ?;

* Jika SUM(total) hasilnya NULL → COALESCE memulangkannya jadi 0.

**Bedanya dengan IFNULL (MySQL):**

* IFNULL(a, b) hanya 2 argumen dan khusus MySQL.
* COALESCE(a, b, c, ...) standar SQL dan bisa banyak argumen.  
  Misal: COALESCE(email, phone, '-') → ambil yang pertama tidak NULL.

**Terkait operator PHP ?? (null coalescing):**

* Di PHP, ($x ?? 0) artinya: kalau $x ada & bukan null, pakai $x, kalau tidak, pakai 0.
* Konsepnya mirip COALESCE di SQL, tapi ini **di PHP**, bukan di database.

**Ringkasan “kapan pakai apa”**

* **preg\_replace** → butuh **mengedit string** pakai regex (misal memodifikasi SQL kamu dari SELECT \* ke COUNT(\*)).
* **prepare** → **eksekusi query** yang aman dengan **placeholder** + bind\_param. Wajib untuk input user.
* **COALESCE** → di **SQL**, untuk pastikan hasil agregat/kolom **tidak NULL**, misal COALESCE(SUM(total),0).

PENUTUP

Alhamdulillah, buku **“Panduan Belajar PHP untuk Pemula”** ini akhirnya dapat diselesaikan. Dalam buku ini telah dibahas berbagai materi dasar pemrograman PHP, mulai dari instalasi XAMPP, sintaks dasar, variabel, operator, percabangan, perulangan, array, fungsi, objek, hingga penggunaan form, session, cookies, dan pengelolaan database MySQL.

Penulis berharap buku ini dapat menjadi pegangan praktis bagi pembaca yang baru memulai belajar pemrograman web. Dengan memahami dasar-dasar PHP, diharapkan pembaca mampu mengembangkan aplikasi web sederhana, serta memiliki fondasi yang kuat untuk mempelajari konsep pemrograman yang lebih lanjut maupun framework modern seperti Laravel.

Penulis menyadari bahwa ilmu pengetahuan terus berkembang, begitu pula teknologi di bidang pemrograman web. Oleh karena itu, pembaca diharapkan terus berlatih, mencari referensi tambahan, dan mengikuti perkembangan teknologi agar tidak tertinggal.

Akhir kata, semoga buku ini dapat memberikan manfaat dan menjadi salah satu pintu masuk untuk menumbuhkan minat serta kemampuan dalam dunia pemrograman web.

Surabaya, September 2025

**Penulis**