

Modul Praktikum
S1-TEKNIK INFORMATIKA

PEMROGRAMAN
WEB



SEKOLAH TINGGI MANAJEMEN INFORMATIKA DAN KOMPUTER
WIDYA CIPTA DHARMA
2025

	STMIK WIDYA CIPTA DHARMA	S1–Teknik Informatika
IFT4234 Pemrograman Web	Pengenalan PHP	Labsheet 01
<i>Semester 4</i>		Dosen: Pitrasacha Adytia,M.T. Email: pitra@wicida.ac.id

1. Tujuan

Modul ini disusun sebagai panduan bagi dosen dalam merancang dan melaksanakan kegiatan praktikum. Tujuannya adalah untuk memastikan bahwa mahasiswa dapat:

- Memahami sintaks dasar PHP.
- Mampu membuat dan menjalankan skrip PHP sederhana.
- Memahami cara mengintegrasikan PHP dengan HTML.
- Mengenali global variabel server di PHP.
- Menggunakan query parameter dalam URL

2. Dasar Teori

2.1 Menampilkan "Hello, World!" di PHP

Buat file baru dengan nama `hello.php` dan isi dengan kode berikut:

```
<?php
    echo "Hello, World!";
?>
```

Langkah-langkah:

- Simpan file sebagai `hello.php` dalam direktori server (`htdocs` untuk XAMPP atau `www` untuk LAMP).
- Buka browser dan akses <http://localhost/hello.php>.

2.2 Menampilkan Informasi PHP

Buat file baru dengan nama `phpinfo.php` dan isi dengan kode berikut:

```
<?php
    phpinfo();
```

?>

Langkah-langkah:

1. Simpan file sebagai `phpinfo.php` dalam direktori server.
2. Buka browser dan akses `http://localhost/phpinfo.php`.
3. Perhatikan informasi yang ditampilkan mengenai versi PHP dan konfigurasi server.

2.3 Integrasi PHP dengan HTML

PHP dapat disisipkan dalam HTML untuk menghasilkan halaman web dinamis. Buat file `index.php` dan isi dengan kode berikut:

```
<!DOCTYPE html>
<html>
<head>
    <title>PHP dan HTML</title>
</head>
<body>
    <h1>Selamat Datang di PHP</h1>
    <p>Hari ini adalah: <?php echo date("l, d F Y"); ?></p>
</body>
</html>
```

Langkah-langkah:

1. Simpan file sebagai `index.php`.
2. Jalankan di browser dengan `http://localhost/index.php`.

2.4 Global Variabel `$_SERVER`

PHP memiliki variabel global `$_SERVER` yang menyimpan informasi tentang server dan request. Buat file `server_info.php` dan isi dengan kode berikut:

```
<?php
echo "Nama Server: " . $_SERVER['SERVER_NAME'] . "<br>";
echo "Alamat IP Server: " . $_SERVER['SERVER_ADDR'] .
"<br>";
echo "Metode Request: " . $_SERVER['REQUEST_METHOD'] .
"<br>";
?>
```

Langkah-langkah:

1. Simpan file sebagai `server_info.php`.
2. Jalankan di browser dengan `http://localhost/server_info.php`.
3. Perhatikan informasi yang ditampilkan.

2.5 Menggunakan Query Parameter

Query parameter digunakan untuk mengirimkan data melalui URL. Buat file `query.php` dan isi dengan kode berikut:

```
<?php
    $name = isset($_GET['name']) ? $_GET['name'] : 'Guest';
    echo "Hello, " . htmlspecialchars($name) . "!";
?>
```

Langkah-langkah:

1. Simpan file sebagai `query.php`.
2. Jalankan di browser dengan `http://localhost/query.php?name=John`.
3. Ganti nilai `name` di URL dan lihat perubahannya.

3. Latihan / Tugas

- a. Modifikasi `index.php` agar menampilkan waktu saat ini selain tanggal.
- b. Buat halaman `greeting.php` yang menerima parameter `name` dan `age` dari URL, kemudian menampilkan pesan seperti `Hello, John! You are 25 years old.`

	STMIK WIDYA CIPTA DHARMA	S1–Teknik Informatika
IFT4234 Pemrograman Web	Website Form	Labsheet 02
<i>Semester 4</i>		Dosen: Pitrasacha Adytia, M.T. Email: pitra@wicida.ac.id

BAB 1: TUJUAN

Modul ini bertujuan untuk:

1. Memahami dan mencegah serangan **XSS (Cross-Site Scripting)**.
2. Membuat dan mengelola **form dengan metode POST** yang berisi berbagai elemen input.
3. Menggunakan **header()** dalam **PHP** untuk manipulasi HTTP request.
4. Melakukan **redirect** menggunakan **PHP**.
5. Mengatur **HTTP Response Code** sesuai kebutuhan dalam aplikasi web.

BAB 2: MATERI

2.1 XSS (Cross-Site Scripting)

Penjelasan

XSS adalah serangan keamanan yang memungkinkan penyerang menyisipkan kode JavaScript berbahaya ke dalam halaman web.

Contoh Kode Rentan

```
<?php
if (isset($_GET['name'])) {
    echo "Hello, " . $_GET['name'];
}
?>
```

Jika pengguna memasukkan `?name=<script>alert('XSS!')</script>`, maka skrip akan dieksekusi di browser.

Cara Mencegah XSS

Gunakan `htmlspecialchars()` untuk menghindari eksekusi skrip:

```
<?php
if (isset($_GET['name'])) {
    echo "Hello, " . htmlspecialchars($_GET['name'], ENT_QUOTES, 'UTF-8');
```

```
}  
?>
```

2.2 Form POST dengan Berbagai Elemen

Penjelasan

Metode **POST** digunakan untuk mengirim data tanpa menampilkannya di URL.

Contoh Implementasi Form POST

```
<!DOCTYPE html>  
<html>  
<head>  
  <title>Form POST</title>  
</head>  
<body>  
  <form action="process.php" method="POST">  
    <label for="name">Nama:</label>  
    <input type="text" id="name" name="name" required><br><br>  
  
    <label>Jenis Kelamin:</label>  
    <input type="radio" name="gender" value="Laki-laki"> Laki-laki  
    <input type="radio" name="gender" value="Perempuan"> Perempuan <br><br>  
  
    <label>Hobi:</label>  
    <input type="checkbox" name="hobby[]" value="Membaca"> Membaca  
    <input type="checkbox" name="hobby[]" value="Olahraga"> Olahraga  
    <input type="checkbox" name="hobby[]" value="Musik"> Musik <br><br>  
  
    <button type="submit">Kirim</button>  
  </form>  
</body>  
</html>
```

File process.php untuk Memproses Data:

```
<?php  
if ($_SERVER["REQUEST_METHOD"] == "POST") {  
  echo "Nama: " . htmlspecialchars($_POST['name']) . "<br>";  
  echo "Jenis Kelamin: " . htmlspecialchars($_POST['gender']) . "<br>";  
  
  if (isset($_POST['hobby'])) {  
    echo "Hobi: " . implode(", ", $_POST['hobby']) . "<br>";  
  }  
}
```

?>

2.3 Header dalam PHP

Penjelasan

Fungsi header() digunakan untuk mengontrol informasi HTTP dalam respons.

Contoh: Menentukan Jenis Konten

```
<?php
header("Content-Type: application/json");
$data = ["message" => "Hello, world!"];
echo json_encode($data);
?>
```

2.4 Redirect dengan Header

Penjelasan

Redirect digunakan untuk mengarahkan pengguna ke halaman lain.

Contoh Redirect ke Halaman Lain

```
<?php
header("Location: https://example.com");
exit();
?>
```

Redirect Setelah Login

```
<?php
if ($_SERVER["REQUEST_METHOD"] == "POST") {
    if ($_POST['username'] == 'admin' && $_POST['password'] == '12345') {
        header("Location: dashboard.php");
        exit();
    } else {
        header("Location: login.php?error=1");
        exit();
    }
}
?>
```

2.5 Response Code dalam PHP

Penjelasan

Kode status HTTP memberi tahu klien tentang status permintaan yang dikirim ke server.

Kode	Status	Deskripsi
200	OK	Permintaan berhasil
301	Moved Permanently	Redirect permanen
302	Found	Redirect sementara
400	Bad Request	Permintaan tidak valid
401	Unauthorized	Tidak memiliki izin akses
403	Forbidden	Akses ditolak
404	Not Found	Halaman tidak ditemukan
500	Internal Server Error	Kesalahan server

Contoh Implementasi Response Code

```

<?php
if (isset($_GET['code'])) {
    http_response_code($_GET['code']);
    switch ($_GET['code']) {
        case 404:
            echo "Halaman tidak ditemukan.";
            break;
        case 500:
            echo "Terjadi kesalahan server.";
            break;
        default:
            echo "Kode tidak dikenali.";
    }
} else {
    http_response_code(200);
    echo "Halaman normal.";
}
?>

```


BAB 3: TUGAS

Tugas 1: Mencegah XSS

Buat file `xss_test.php` yang menerima input dari GET. Terapkan `htmlspecialchars()` dan bandingkan hasilnya dengan tanpa filter.

Tugas 2: Form POST

Buat form di `form_post.php` yang memiliki input teks, radio button, checkbox, dropdown, dan textarea. Tampilkan hasil input di `process.php`.

Tugas 3: Menggunakan Header

Buat file `header_example.php` yang mengatur Content-Type: `application/json` dan menampilkan data dalam format JSON.

Tugas 4: Redirect

Buat sistem login sederhana dengan `login.php` dan redirect ke `dashboard.php` jika login berhasil.

Tugas 5: Response Code

Buat file `response_code.php` yang menampilkan pesan berdasarkan kode HTTP yang diberikan sebagai parameter di URL.

	STMIK WIDYA CIPTA DHARMA	S1–Teknik Informatika
IFT4234 Pemrograman Web	Website Dinamis	Labsheet 03
<i>Semester 4</i>		Dosen: Pitrasacha Adytia, M.T. Email: pitra@wicida.ac.id

BAB 1: TUJUAN

Modul ini bertujuan untuk:

1. Memahami penggunaan **Session** dan **Cookie** dalam PHP.
2. Mampu melakukan **upload dan download file** menggunakan PHP.
3. Mengenali **tipe data** dan penggunaan **operator** dalam PHP.
4. Menggunakan **percabangan dan perulangan** untuk pengambilan keputusan.
5. Mengelola **array** dan **manipulasi string** dalam PHP.

BAB 2: MATERI

2.1 SESSION

Session digunakan untuk menyimpan data yang bisa digunakan di berbagai halaman selama sesi masih aktif.

Langkah-langkah:

1. **Membuka session** pada halaman PHP menggunakan `session_start()`.
2. **Menyimpan data ke dalam session** menggunakan `$_SESSION`.
3. **Mengakses data session** dari halaman lain.
4. **Menghapus session** jika tidak diperlukan.

Contoh Implementasi

```
php
CopyEdit
<?php
// Mulai session
session_start();
$_SESSION['username'] = "admin";
echo "Session username telah disimpan.";
?>
```

Mengakses Session di Halaman Lain

```
php
CopyEdit
<?php
session_start();
echo "Username: " . $_SESSION['username'];
?>
```

Menghapus Session

```
php
CopyEdit
<?php
session_start();
session_destroy();
echo "Session telah dihapus.";
?>
```

2.2 COOKIE

Cookie digunakan untuk menyimpan data di browser pengguna.

Langkah-langkah:

1. **Membuat cookie** menggunakan `setcookie()`.
2. **Mengakses cookie** dengan `$_COOKIE`.
3. **Menghapus cookie** dengan mengatur waktu kedaluwarsa di masa lalu.

Contoh Implementasi

```
php
CopyEdit
<?php
// Menyimpan cookie selama 1 jam
setcookie("user", "admin", time() + 3600);
echo "Cookie telah disimpan.";
?>
```

Mengakses Cookie

```
php
CopyEdit
<?php
if (isset($_COOKIE['user'])) {
    echo "User: " . $_COOKIE['user'];
}
?>
```

Menghapus Cookie

```
php
CopyEdit
<?php
setcookie("user", "", time() - 3600);
echo "Cookie telah dihapus.";
?>
```

2.3 UPLOAD FILE

Menggunakan form dengan `enctype="multipart/form-data"` untuk menangani upload file.

Langkah-langkah:

1. Buat form HTML dengan input file.
2. Periksa apakah ada file yang diunggah.
3. Simpan file ke dalam folder tertentu.

Contoh Form Upload File

```
php
CopyEdit
<!DOCTYPE html>
<html>
<body>
    <form action="upload.php" method="POST" enctype="multipart/form-data">
        <input type="file" name="fileToUpload">
        <input type="submit" value="Upload">
    </form>
</body>
</html>
```

File `upload.php` untuk Memproses Upload

```
php
CopyEdit
<?php
if ($_FILES["fileToUpload"]["error"] == 0) {
    move_uploaded_file($_FILES["fileToUpload"]["tmp_name"], "uploads/" .
$_FILES["fileToUpload"]["name"]);
    echo "File berhasil diupload.";
} else {
    echo "Gagal upload.";
}
?>
```

2.4 DOWNLOAD FILE

Langkah-langkah:

1. Tentukan file yang akan diunduh.
2. Gunakan header `Content-Disposition` untuk mendownload file.
3. Gunakan `readfile()` untuk membaca file.

Kode untuk Mengunduh File

```
php
CopyEdit
<?php
$file = "path/to/file.pdf";
header("Content-Disposition: attachment; filename=" . basename($file));
readfile($file);
?>
```

2.5 TIPE DATA DI PHP

Tipe Data	Contoh
String	"Hello World"
Integer	123
Float	3.14
Boolean	true / false
Array	["Apple", "Banana"]
Object	<code>\$obj = new ClassName();</code>
Null	null

2.6 OPERATOR DI PHP

Langkah-langkah:

1. Gunakan **operator aritmatika** untuk perhitungan.
2. Gunakan **operator perbandingan** untuk mengevaluasi nilai.
3. Gunakan **operator logika** untuk kondisi kompleks.

Contoh:

```
php
CopyEdit
$a = 10; $b = 5;
echo $a + $b; // 15
var_dump($a == $b); // false
var_dump($a > 5 && $b < 10); // true
```

2.7 PERCABANGAN

Langkah-langkah:

1. Gunakan `if-else` untuk kondisi sederhana.
2. Gunakan `switch-case` untuk banyak kondisi.

```
php
CopyEdit
$nilai = 80;
if ($nilai >= 75) {
    echo "Lulus";
} else {
    echo "Tidak Lulus";
}
```

2.8 PERULANGAN

Langkah-langkah:

1. Gunakan **for** untuk perulangan dengan batas tetap.
2. Gunakan **while** untuk perulangan dengan kondisi.

```
php
CopyEdit
for ($i = 1; $i <= 5; $i++) {
    echo "Angka: $i <br>";
}
```

2.9 ARRAY

Langkah-langkah:

1. Gunakan **array sederhana** untuk menyimpan daftar nilai.
2. Gunakan **array asosiatif** untuk pasangan kunci-nilai.

```
php
CopyEdit
$buah = ["Apel", "Jeruk", "Mangga"];
echo $buah[1]; // Jeruk
```

2.10 MANIPULASI STRING

Langkah-langkah:

1. Gunakan `strlen()` untuk menghitung panjang string.
2. Gunakan `strtoupper()` untuk membuat teks kapital.

```
php
CopyEdit
$str = "Hello PHP";
echo strlen($str); // Panjang string
echo strtoupper($str); // Kapital semua huruf
```

2.11 FUNCTION


Langkah-langkah:

1. Gunakan `function` untuk mengelompokkan kode yang bisa digunakan kembali.
2. Panggil fungsi dengan parameter jika diperlukan.

```
php
CopyEdit
function sapa($nama) {
    return "Halo, " . $nama;
}
echo sapa("Andi");
```

BAB 3: TUGAS

1. **Session & Cookie:**
 - a. Buat halaman login dengan **session** yang menyimpan username dan logout yang menghapus session.
 - b. Tambahkan **cookie** yang menyimpan username selama 1 hari.
2. **Upload & Download File:**
 - a. Buat sistem upload file yang hanya menerima **.jpg, .png** dan ukuran maksimum **1MB**.
 - b. Tambahkan fitur **download file** dari server.
3. **Tipe Data & Operator:** Buat program yang menampilkan **tipe data** dan menggunakan operator aritmatika & perbandingan.
4. **Percabangan & Perulangan:**
 - a. Buat program yang meminta **nilai ujian** dan menampilkan status lulus/tidak lulus dengan **if-else**.
 - b. Buat perulangan **for** yang mencetak angka 1-10.
5. **Array & String Manipulation:**
 - a. Buat array daftar nama mahasiswa dan cetak nama mereka dengan **foreach**.
 - b. Buat string manipulasi untuk mengubah teks ke huruf besar dan mengganti kata.
6. **Function:** Buat fungsi **luasLingkaran(\$radius)** untuk menghitung luas lingkaran.

	STMIK WIDYA CIPTA DHARMA	S1–Teknik Informatika
IFT4234 Pemrograman Web	PHP Object Oriented	Labsheet 04
<i>Semester 4</i>		Dosen: Pitrasacha Adytia, M.T. Email: pitra@wicida.ac.id

BAB 1: TUJUAN

1. Memahami konsep dasar **Object-Oriented Programming (OOP)** dalam PHP.
2. Membuat dan menggunakan **class, object, properties, dan function** dalam OOP.
3. Menggunakan **this keyword** dan **constant** dalam class PHP.
4. Mengimplementasikan **constructor dan destructor** dalam PHP OOP.
5. Menerapkan konsep **inheritance** untuk pewarisan class.
6. Menerapkan **polymorphism** untuk fleksibilitas dalam pemrograman OOP.

BAB 2: MATERI

2.1 PENGENALAN OOP

Object-Oriented Programming (OOP) adalah paradigma pemrograman yang berfokus pada penggunaan **object** dan **class** untuk membangun aplikasi.

Langkah-langkah:

1. Buat class dengan **properties** dan **function**.
2. Buat object berdasarkan class tersebut.
3. Akses properties dan function dari object.

Contoh Implementasi

```

php
CopyEdit
<?php
class Mobil {
    public $merk;
    public $warna;

    public function tampilkanInfo() {
        return "Mobil $this->merk berwarna $this->warna.";
    }
}

```



```
// Membuat object dari class Mobil
$mobil1 = new Mobil();
$mobil1->merk = "Toyota";
$mobil1->warna = "Merah";

echo $mobil1->tampilkanInfo();
?>
```

2.2 CLASS, OBJECT, PROPERTIES, FUNCTION

Langkah-langkah:

1. Buat class baru dengan beberapa **properties**.
2. Tambahkan function untuk mengembalikan nilai dari properties.
3. Buat object berdasarkan class tersebut.

Contoh Implementasi

```
php
CopyEdit
<?php
class Laptop {
    public $merek;
    public $ram;

    public function tampilkanSpesifikasi() {
        return "Laptop $this->merek dengan RAM $this->ram GB.";
    }
}

// Membuat object dari class Laptop
$laptop1 = new Laptop();
$laptop1->merek = "Asus";
$laptop1->ram = 8;

echo $laptop1->tampilkanSpesifikasi();
?>
```

2.3 THIS KEYWORD & CONSTANT

Langkah-langkah:

1. Gunakan `this` untuk mengacu pada properties atau function dalam class.
2. Gunakan `const` untuk mendefinisikan **constant** di dalam class.

Contoh Implementasi

```
php
```

```

CopyEdit
<?php
class Komputer {
    public $prosesor;
    const GARANSI = "2 Tahun";

    public function tampilkanInfo() {
        return "Komputer ini menggunakan prosesor $this->prosesor dan
memiliki garansi " . self::GARANSI;
    }
}

// Membuat object
$pc = new Komputer();
$pc->prosesor = "Intel Core i7";
echo $pc->tampilkanInfo();
?>

```

2.4 CONSTRUCTOR & DESTRUCTOR

Langkah-langkah:

1. Gunakan **constructor** untuk mengatur nilai awal saat object dibuat.
2. Gunakan **destructor** untuk membersihkan atau menutup koneksi setelah object selesai digunakan.

Contoh Implementasi

```

php
CopyEdit
<?php
class Produk {
    public $nama;

    // Constructor
    public function __construct($nama) {
        $this->nama = $nama;
        echo "Produk $this->nama telah dibuat.<br>";
    }

    // Destructor
    public function __destruct() {
        echo "Produk $this->nama dihapus.<br>";
    }
}

// Membuat object
$produk1 = new Produk("Smartphone");
?>

```

2.5 INHERITANCE (PEWARISAN)

Langkah-langkah:

1. Gunakan **extends** untuk membuat class anak yang mewarisi class induk.
2. Tambahkan function baru di class anak.
3. Gunakan **parent::__construct()** jika class anak memiliki constructor.

Contoh Implementasi

```
php
CopyEdit
<?php
// Class Induk
class Kendaraan {
    public $merk;

    public function __construct($merk) {
        $this->merk = $merk;
    }

    public function info() {
        return "Ini adalah kendaraan merk $this->merk.";
    }
}

// Class Anak
class Motor extends Kendaraan {
    public $cc;

    public function __construct($merk, $cc) {
        parent::__construct($merk);
        $this->cc = $cc;
    }

    public function info() {
        return "Motor merk $this->merk dengan kapasitas $this->cc cc.";
    }
}

// Membuat object dari class anak
$motor1 = new Motor("Honda", 150);
echo $motor1->info();
?>
```

2.6 POLYMORPHISM

Langkah-langkah:

1. Gunakan **inheritance** untuk membuat class anak.
2. Override function di class anak untuk memberikan perilaku yang berbeda.

Contoh Implementasi

```

php
CopyEdit
<?php
class Hewan {
    public function bersuara() {
        return "Hewan ini bersuara.";
    }
}

class Kucing extends Hewan {
    public function bersuara() {
        return "Meong!";
    }
}

class Anjing extends Hewan {
    public function bersuara() {
        return "Guk guk!";
    }
}

// Membuat object
$hewan1 = new Kucing();
echo $hewan1->bersuara(); // Meong!

$hewan2 = new Anjing();
echo $hewan2->bersuara(); // Guk guk!
?>

```

BAB 3: TUGAS

Tugas 1: Membuat Class dan Object

Buat **class Mahasiswa** dengan properties:

- nama
- nim
- jurusan

Tambahkan function `tampilkanInfo()` untuk menampilkan data mahasiswa dalam format berikut:

"Nama: [nama], NIM: [nim], Jurusan: [jurusan]"

Buat object dari class tersebut dan tampilkan hasilnya.

Tugas 2: Constructor dan Destructor

Buat class **Pegawai** dengan properties:

- nama
- jabatan

Gunakan **constructor** untuk mengatur nilai saat object dibuat dan **destructor** untuk menampilkan pesan saat object dihapus.

Tugas 3: Inheritance

Buat class **Karyawan** sebagai parent class dengan properties:

- nama
- gaji


Buat class **Manager** yang mewarisi class **Karyawan** dan memiliki tambahan property tunjangan.

Buat function `hitungTotalGaji()` yang menghitung total gaji (gaji pokok + tunjangan).

Tugas 4: Polymorphism

Buat class **Hewan** dengan function `suara()`.

Buat class **Kucing** dan **Anjing** yang mewarisi **Hewan** dan override function `suara()` sesuai jenis hewan.

	STMIK WIDYA CIPTA DHARMA	S1–Teknik Informatika
IFT4234 Pemrograman Web	PHP Object Oriented	Labsheet 05
<i>Semester 4</i>		Dosen: Pitrasacha Adytia, M.T. Email: pitra@wicida.ac.id

BAB 1: TUJUAN

Setelah menyelesaikan modul ini, mahasiswa diharapkan dapat:

1. Membuat **website sederhana** menggunakan PHP
2. Menggunakan **Form Handling (GET & POST)** dan mencegah **XSS**.
3. Mengelola **session dan cookie** untuk autentikasi pengguna.
4. Menggunakan **upload dan download file** dalam PHP.
5. Menerapkan **OOP (Object-Oriented Programming)** dalam pengelolaan data.
6. Menggunakan konsep **inheritance & polymorphism** dalam sistem.

BAB 2: MATERI

Mahasiswa akan membangun **Sistem Manajemen Data Mahasiswa** yang memiliki fitur:

1. **Autentikasi pengguna (Login & Logout menggunakan Session & Cookie)**
2. **CRUD (Create, Read, Update, Delete) data mahasiswa menggunakan file TXT sebagai penyimpanan sementara**
3. **Upload & Download file dokumen mahasiswa**
4. **Menggunakan OOP untuk mengelola data mahasiswa**

□ Struktur Folder Proyek

```
bash
CopyEdit
/website-sederhana
├── /uploads
├── /assets
├── index.php
├── login.php
├── logout.php
├── tambah.php
└── edit.php
```

```
— hapus.php
— upload.php
— download.php
— data_mahasiswa.txt
— style.css
— README.md
```

BAB 3: MEMBUAT SISTEM LOGIN SEDERHANA DENGAN SESSION & COOKIE

3.1 Form Login (login.php)

```
php
CopyEdit
<?php
session_start();
$error = "";

if ($_SERVER["REQUEST_METHOD"] == "POST") {
    $username = $_POST["username"];
    $password = $_POST["password"];

    if ($username == "admin" && $password == "12345") {
        $_SESSION["user"] = $username;
        setcookie("user", $username, time() + (86400 * 1), "/"); // Cookie
        berlaku 1 hari
        header("Location: index.php");
        exit();
    } else {
        $error = "Username atau Password salah!";
    }
}

?>

<!DOCTYPE html>
<html>
<head>
    <title>Login</title>
</head>
<body>
    <h2>Login</h2>
    <form method="POST">
        <label>Username:</label>
        <input type="text" name="username" required><br>
        <label>Password:</label>
        <input type="password" name="password" required><br>
        <button type="submit">Login</button>
    </form>
    <p style="color:red;"><?php echo $error; ?></p>
</body>
</html>
```

3.2 Logout (logout.php)

```

php
CopyEdit
<?php
session_start();
session_destroy();
setcookie("user", "", time() - 3600, "/");
header("Location: login.php");
exit();
?>

```

BAB 4: MEMBUAT SISTEM CRUD TANPA DATABASE (MENGUNAKAN FILE TXT)

4.1 Menampilkan Data Mahasiswa (index.php)

```

php
CopyEdit
<?php
session_start();
if (!isset($_SESSION["user"])) {
    header("Location: login.php");
    exit();
}

$data_mahasiswa = file("data_mahasiswa.txt", FILE_IGNORE_NEW_LINES);
?>

<!DOCTYPE html>
<html>
<head>
    <title>Data Mahasiswa</title>
</head>
<body>
    <h2>Daftar Mahasiswa</h2>
    <a href="tambah.php">Tambah Mahasiswa</a> | <a
href="logout.php">Logout</a>
    <table border="1">
        <tr>
            <th>Nama</th>
            <th>NIM</th>
            <th>Jurusan</th>
            <th>Aksi</th>
        </tr>
        <?php foreach ($data_mahasiswa as $data): ?>
            <?php list($nama, $nim, $jurusan) = explode("|", $data); ?>
            <tr>
                <td><?= $nama ?></td>
                <td><?= $nim ?></td>
                <td><?= $jurusan ?></td>
                <td>
                    <a href="edit.php?nim=<?= $nim ?>">Edit</a> |
                    <a href="hapus.php?nim=<?= $nim ?>">Hapus</a>
                </td>
            </tr>
        <?php endforeach; ?>
    </table>

```



```

    </table>
</body>
</html>

```

4.2 Menambahkan Data (tambah.php)

```

php
CopyEdit
<?php
if ($_SERVER["REQUEST_METHOD"] == "POST") {
    $nama = $_POST["nama"];
    $nim = $_POST["nim"];
    $jurusan = $_POST["jurusan"];

    $data = "$nama|$nim|$jurusan\n";
    file_put_contents("data_mahasiswa.txt", $data, FILE_APPEND);

    header("Location: index.php");
    exit();
}
?>

<!DOCTYPE html>
<html>
<head>
    <title>Tambah Mahasiswa</title>
</head>
<body>
    <h2>Tambah Data Mahasiswa</h2>
    <form method="POST">
        <label>Nama:</label>
        <input type="text" name="nama" required><br>
        <label>NIM:</label>
        <input type="text" name="nim" required><br>
        <label>Jurusan:</label>
        <input type="text" name="jurusan" required><br>
        <button type="submit">Simpan</button>
    </form>
</body>
</html>

```

BAB 5: UPLOAD & DOWNLOAD FILE

5.1 Upload File (upload.php)

```

php
CopyEdit
<?php
if ($_FILES["fileToUpload"]["error"] == 0) {
    move_uploaded_file($_FILES["fileToUpload"]["tmp_name"], "uploads/" .
$_FILES["fileToUpload"]["name"]);
    echo "File berhasil diupload.";
} else {

```

```

        echo "Gagal upload.";
    }
    ?>

<!DOCTYPE html>
<html>
<head>
    <title>Upload File</title>
</head>
<body>
    <h2>Upload File</h2>
    <form action="" method="POST" enctype="multipart/form-data">
        <input type="file" name="fileToUpload">
        <input type="submit" value="Upload">
    </form>
</body>
</html>

```

5.2 Download File (download.php)

```

php
CopyEdit
<?php
$file = "uploads/example.txt";
header("Content-Disposition: attachment; filename=" . basename($file));
readfile($file);
?>

```

BAB 6: PENERAPAN OOP

6.1 Membuat Class Mahasiswa (mahasiswa.php)

```

php
CopyEdit
<?php
class Mahasiswa {
    public $nama, $nim, $jurusan;

    public function __construct($nama, $nim, $jurusan) {
        $this->nama = $nama;
        $this->nim = $nim;
        $this->jurusan = $jurusan;
    }

    public function tampilkanInfo() {
        return "$this->nama ($this->nim) - $this->jurusan";
    }
}

$mahasiswa1 = new Mahasiswa("Andi", "21012345", "Teknik Informatika");
echo $mahasiswa1->tampilkanInfo();
?>

```

BAB 7: TUGAS PRAKTIKUM

1. **Selesaikan sistem login dengan validasi tambahan**
2. **Tambahkan fitur edit & hapus data mahasiswa**
3. **Tambahkan fitur pencarian mahasiswa berdasarkan NIM**
4. **Tambahkan fitur upload dan download file CV mahasiswa**
5. **Implementasikan OOP untuk menyimpan data mahasiswa dalam array**