

LAPORAN PRAKTIKUM CRUD DAN LOGIN PHP NATIVE



Nama Anggota :

Dimas Prasetyo (08)

Muhammad Farrel Hakim (20)

Muhammad Nawfal Aryan S (21)

Titis Viqnatan Candra Sally (32)

REKAYASA PERANGKAT LUNAK

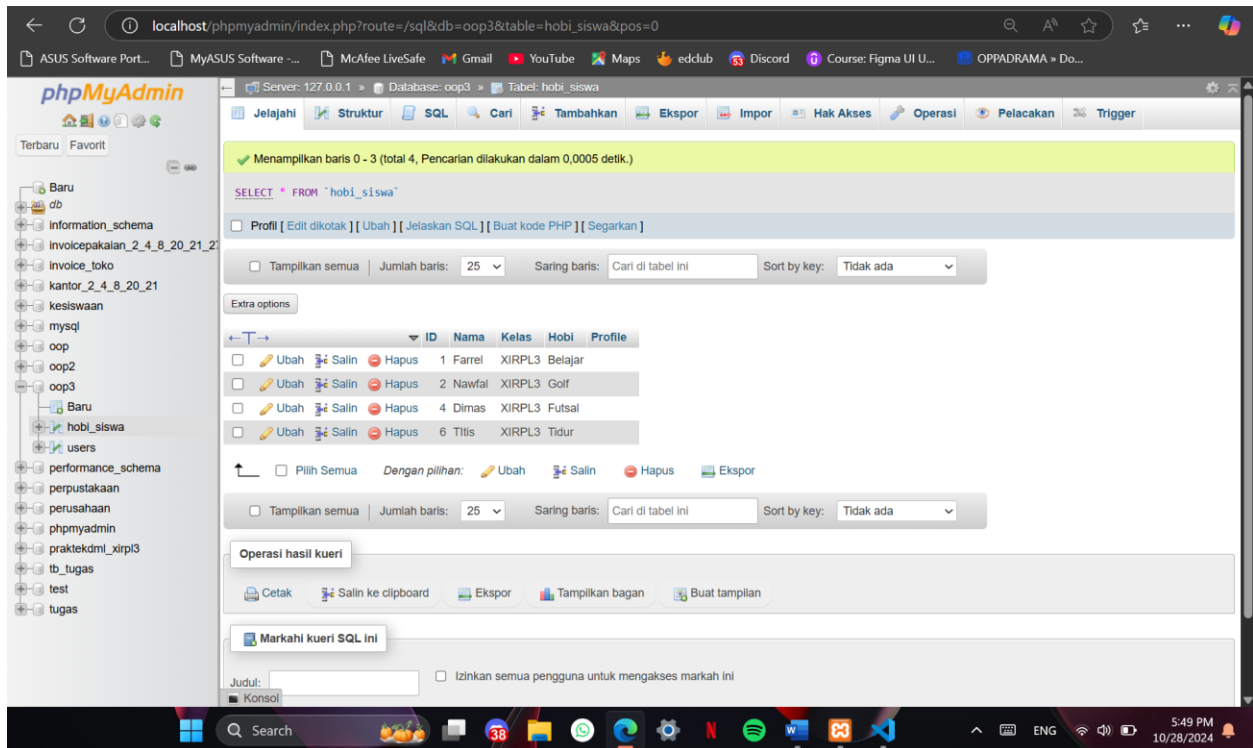
SMK TEXMACO SEMARANG

2024

TAHAPAN

I. hobi_siswa

A. Membuat Database dan Tabel



Langkah pertama untuk menjalankan program PHP ini adalah membuat database bernama `oop3` dan tabel `hobi_siswa` di dalamnya. Tabel `hobi_siswa` memiliki kolom `ID`, `Nama`, `Kelas`, `Hobi`, dan `Profile`, yang masing-masing difungsikan untuk menyimpan informasi siswa, termasuk ID unik sebagai kunci utama (`AUTO_INCREMENT`), nama siswa (`VARCHAR(50)`), kelas (`VARCHAR(10)`), hobi (`VARCHAR(100)`), dan profil tambahan (`TEXT`). Setelah tabel dibuat, satu data dapat dimasukkan menggunakan perintah `INSERT`, contohnya: `INSERT INTO hobi_siswa (Nama, Kelas, Hobi, Profile) VALUES ('Nawfal', 'XIRPL3', 'Golf', 'Link foto yang akan ditambahkan');`. Data ini akan memberikan ID otomatis pada entri pertama dan menyimpan informasi lengkap siswa bernama Nawfal, yang berada di kelas XIRPL3 dengan hobi Golf. Struktur tabel ini siap untuk dikelola oleh program PHP, yang akan memungkinkan pembaruan, penambahan, atau penghapusan data secara dinamis.

B. Membuat koneksi dengan nama file connect.php

```
<?php
class Database {
private $host = "localhost";
private $db_name = "oop3";
private $username = "root";
private $password = "";

public $conn;
public function getConnection() {
$this->conn = null;
try {
$this->conn = new PDO("mysql:host=" . $this->host . ";dbname="
. $this->db_name, $this->username, $this->password);
$this->conn->exec("set names utf8");
}catch(PDOException $exception) {
echo "Connection error:" . $exception->getMessage();
}
return $this->conn;
}
}

?>
```

Program PHP ini mendefinisikan kelas `Database` untuk mengelola koneksi ke database menggunakan PDO. Kelas ini memiliki beberapa properti, termasuk `\$host`, `\$db_name`, `\$username`, dan `\$password`, yang digunakan untuk menyimpan informasi koneksi database seperti host (`localhost`), nama database (`oop3`), nama pengguna (`root`), dan kata sandi (kosong). Fungsi `getConnection()` bertugas untuk membuat koneksi ke database: awalnya, ia menginisialisasi properti `\$conn` sebagai `null` dan kemudian mencoba membuat objek PDO untuk koneksi ke database MySQL berdasarkan informasi yang diberikan. Pengaturan `utf8` digunakan untuk memastikan karakter ditampilkan dengan benar. Jika koneksi berhasil, objek PDO disimpan di `\$conn` dan dikembalikan; jika terjadi kesalahan, fungsi menangkap pengecualian dan menampilkan pesan error. Kelas ini memungkinkan bagian lain dari program untuk mengakses database `oop3` dengan mudah dan menangani koneksi secara terpusat.

C. Menampilkan Data

```
<?php
include './connect.php';
include './hobi.php';
$db = new Database();
$hobi = new Hobi($db->getConnection());

session_start();
if (!isset($_SESSION['username'])) {
    echo "<script>alert('Anda belum login, silahkan login terlebih dahulu.');
```

```
    text-align: center;
    text-transform: uppercase;
    letter-spacing: 1.5px;
    margin-bottom: 20px;
}

h3 {
    font-size: 1.4rem;
    color: #555;
    text-align: center;
    margin-bottom: 20px;
}

a {
    display: inline-block;
    background-color: #6c5ce7;
    color: #fff;
    padding: 10px 18px;
    text-decoration: none;
    border-radius: 4px;
    font-weight: 500;
    margin-bottom: 20px;
    transition: background-color 0.3s ease;
}

a:hover {
    background-color: #574b90;
}

.container {
    max-width: 900px;
    margin: 0 auto;
}

table {
    width: 100%;
    border-collapse: collapse;
    background-color: #fff;
    border-radius: 8px;
    overflow: hidden;
    box-shadow: 0 2px 8px rgba(0, 0, 0, 0.1);
}

th, td {
    padding: 12px 16px;
```

```
        text-align: center;
    }

    th {
        background-color: #6c5ce7;
        color: white;
        font-weight: 600;
        text-transform: uppercase;
        letter-spacing: 0.5px;
    }

    td {
        font-size: 0.95rem;
        color: #333;
    }

    tr:nth-child(even) {
        background-color: #f7f7f7;
    }

    tr:hover {
        background-color: #ececec;
        transition: background-color 0.3s ease;
    }

    .btn {
        padding: 8px 12px;
        border: none;
        border-radius: 4px;
        color: white;
        font-size: 0.85rem;
        text-transform: uppercase;
        cursor: pointer;
        transition: background-color 0.3s ease;
    }

    .btn-edit {
        background-color: #00b894;
    }

    .btn-delete {
        background-color: #d63031;
    }

    .btn-out {
```

```
        background-color: #d63031;
    }

    .btn-edit:hover {
        background-color: #009975;
    }

    .btn-delete:hover {
        background-color: #b82d2a;
    }

    .btn-out:hover {
        background-color: #b82d2a;
    }

    .btn-add {
        display: block;
        margin: 0 auto 20px;
        text-align: center;
    }

    .btn-out {
        display: flex;
        margin: 10px;
        width: 80px;
        text-align: center;
        position: absolute;
        right: 270px;
        top: 105px;
        color: white;
        padding: 8px 8px;
        border: none;
        border-radius: 4px;
        font-size: 0.85rem;
        text-transform: uppercase;
        cursor: pointer;
        transition: background-color 0.3s ease;
    }

    @media (max-width: 768px) {
        table, th, td {
            font-size: 0.85rem;
        }

        .btn {
            font-size: 0.75rem;
        }
    }
}
```

```

        padding: 6px 10px;
    }
}

</style>
</head>
<body>

<div class="container">
    <h1>CRUD OOP PHP</h1>
    <h3>Data Hobi Siswa</h3>
    <button class="btn-out" onclick="if(confirm('Apakah anda yakin
ingin log out?'))
window.location='../Login/dashboard.php?logout';">Log Out</button>
    <a href="./create.php" class="btn btn-add">Tambahkan Data</a>

    <table>
        <tr>
            <th>ID</th>
            <th>Profile</th>
            <th>Nama</th>
            <th>Kelas</th>
            <th>Hobi</th>
            <th colspan="2">Aksi</th>
        </tr>
        <?php
        foreach($hobi->read() as $x){
            $id = $x["ID"];
            $nama = $x["Nama"];
            $kelas = $x["Kelas"];
            $hobi_siswa = $x["Hobi"];

            echo "
            <tr>
                <td>" . $id . "</td>
                <td><img alt='" . $nama . "' src='" . $x["Profile"] . "'
class='img-fluid' style='width: 56px; border-radius: 200px;'/></td>
                <td>" . $nama . "</td>
                <td>" . $kelas . "</td>
                <td>" . $hobi_siswa . "</td>
                <td><a href='./edit.php?ID=" . $id . "&Nama=" . $nama
. "&Kelas=" . $kelas . "&Hobi=" . $hobi_siswa . "\" class='btn btn-
edit'>Edit</a></td>

```



```

                <td><a onclick=\"if (confirm('Yakin hapus ini?'))
window.location.href = './delete_action.php?ID=' . $id . ''\"
class='btn btn-delete'>Delete</a></td>
            </tr>";
        }
    ?>
</table>
</div>

</body>
</html>

```






Program PHP ini menampilkan antarmuka web untuk mengelola data hobi siswa menggunakan konsep CRUD (Create, Read, Update, Delete) dalam pola OOP. Program pertama-tama memuat file koneksi ('connect.php') dan file kelas 'Hobi' ('hobi.php'), kemudian membuat objek 'Database' dan 'Hobi' untuk terhubung ke database dan mengambil data. Program memeriksa sesi login pengguna, dan jika belum login, mengarahkan pengguna ke halaman login. Halaman HTML ini menampilkan judul, tabel data hobi siswa yang diperoleh melalui metode 'read()' dari objek 'Hobi', dan tombol untuk menambah data baru atau mengedit dan menghapus data yang ada. Tabel menunjukkan kolom 'ID', 'Profile', 'Nama', 'Kelas', dan 'Hobi' untuk setiap siswa, serta menyediakan tombol "Edit" dan "Delete" untuk mengelola setiap entri. Terdapat juga tombol log out yang menampilkan konfirmasi sebelum pengguna diarahkan ke halaman login saat memilih log out. Desain antarmuka dibuat responsif dan estetis dengan gaya CSS yang mendukung tampilan modern. Hasilnya akan seperti dibawah ini.

CRUD OOP PHP

Data Hobi Siswa

LOG OUT

TAMBAHKAN DATA

ID	PROFILE	NAMA	KELAS	HOBI	AKSI	
1		Noval	XIRPL3	Melihat Kecoak	EDIT	DELETE
2		Nawfal	XIRPL3	Golf	EDIT	DELETE
4		Dimas	XIRPL3	Futsal	EDIT	DELETE
6		Titis	XIRPL3	Tidur	EDIT	DELETE
8		Rayhan	XITB2	Tidur	EDIT	DELETE

D. Tambah Data

- create.php

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-
scale=1.0">
  <title>Tambah Hobi Siswa</title>
  <style>
    @import
url('https://fonts.googleapis.com/css2?family=Poppins:wght@300;400;600
&display=swap');

    body {
      font-family: 'Poppins', sans-serif;
      background: linear-gradient(135deg, #e0eafc, #cfdef3);
      display: flex;
      justify-content: center;
      align-items: center;
      height: 100vh;
      margin: 0;
    }

    h1 {
      text-align: center;
      color: #34495e;
      font-size: 2rem;
      text-transform: uppercase;
      letter-spacing: 1px;
      margin-bottom: 20px;
    }

    form {
      background-color: #fff;
      padding: 30px;
      border-radius: 10px;
      box-shadow: 0 4px 12px rgba(0, 0, 0, 0.1);
      width: 300px;
    }

    label {
      font-size: 1rem;
      color: #333;
```

```

        display: block;
        margin-bottom: 10px;
    }

    input[type="text"], input[type="submit"] {
        width: 100%;
        padding: 10px;
        margin-bottom: 15px;
        border-radius: 5px;
        border: 1px solid #ddd;
        font-size: 1rem;
    }

    input[type="text"]:focus {
        border-color: #2980b9;
        outline: none;
    }

    input[type="submit"] {
        background-color: #2980b9;
        color: white;
        font-size: 1rem;
        cursor: pointer;
        border: none;
        transition: background-color 0.3s ease;
    }

    input[type="submit"]:hover {
        background-color: #1f5a80;
    }

    .form-group {
        margin-bottom: 20px;
    }

    .form-group:last-of-type {
        margin-bottom: 0;
    }
</style>
</head>
<body>
    <form action="create_action.php" method="post">
        <h1>Tambah Data Hobi Siswa</h1>
        <div class="form-group">

```

```

        <label for="nama">Nama</label>
        <input type="text" name="nama" id="nama"
placeholder="Masukkan nama">
    </div>
    <div class="form-group">
        <label for="posisi">Kelas</label>
        <input type="text" name="kelas" id="kelas"
placeholder="Masukkan kelas">
    </div>
    <div class="form-group">
        <label for="hobi">Hobi</label>
        <input type="text" name="hobi" id="hobi"
placeholder="Masukkan hobi">
    </div>
    <div class="form-group">
        <label for="profile">Profile URL</label>
        <input type="text" name="profile" id="profile"
placeholder="Masukkan profile url">
    </div>
    <input type="submit" value="Simpan">
</form>
</body>
</html>

```

Program HTML ini menampilkan antarmuka form untuk menambahkan data hobi siswa ke database. Formulir ini dirancang dengan HTML dan CSS, serta mengarahkan data yang dimasukkan ke file `create_action.php` untuk diproses lebih lanjut melalui metode POST. Di dalam form, pengguna diminta untuk mengisi empat bidang input: `Nama`, `Kelas`, `Hobi`, dan `Profile URL`, dengan masing-masing memiliki label yang sesuai dan kolom teks yang siap diisi. Gaya desainnya modern, menggunakan font `Poppins` dan background gradien biru muda, sementara form berada di tengah layar dengan gaya kotak putih yang dilengkapi bayangan lembut, memberi kesan estetis dan minimalis. Input untuk `nama`, `kelas`, `hobi`, dan `profile` diberi batasan dan efek fokus untuk menunjukkan interaksi pengguna. Tombol submit memiliki warna biru yang berubah saat di-hover, memberikan pengalaman pengguna yang dinamis dan responsif.

- create_action.php

```
<?php
include_once 'connect.php';
include_once 'hobi.php';

$database = new Database();
$db = $database->getConnection();

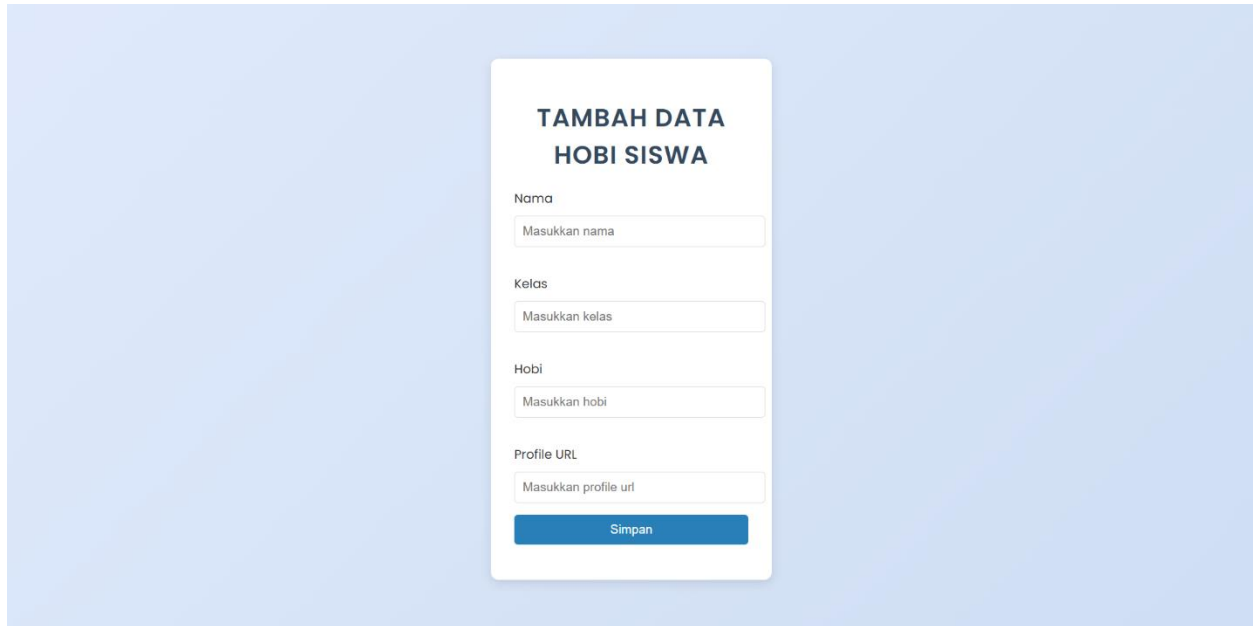
$hobi = new hobi($db);
$hobi->nama = $_POST['nama'];
$hobi->kelas = $_POST['kelas'];
$hobi->hobi_siswa = $_POST['hobi'];
$hobi->profile = $_POST['profile'];

$hasil = $hobi->create();

if($hasil) {
    echo "Data berhasil ditambahkan.";
} else {
    echo "Gagal menambahkan data.";
}
echo "<BR> <a href='./'>Back</a>"
?>
```

Program PHP ini bertugas untuk menangani penambahan data hobi siswa ke dalam database. Pertama, program mengimpor file koneksi ('connect.php') dan file kelas 'Hobi' ('hobi.php'), kemudian membuat objek 'Database' untuk mendapatkan koneksi ke database. Setelah itu, objek 'Hobi' diinisialisasi dengan koneksi database yang sudah dibuat. Program mengambil data dari formulir yang dikirim melalui metode POST—yaitu 'nama', 'kelas', 'hobi', dan 'profile'—lalu menetakannya ke properti yang sesuai dalam objek 'Hobi'. Selanjutnya, metode 'create()' dari objek 'Hobi' dipanggil untuk menyimpan data tersebut ke dalam tabel database. Jika penyimpanan berhasil, program akan menampilkan pesan "Data berhasil ditambahkan." Sebaliknya, jika gagal, pesan "Gagal menambahkan data." akan ditampilkan. Terakhir, terdapat tautan yang mengarahkan kembali pengguna ke halaman sebelumnya untuk melanjutkan interaksi. Program ini merupakan bagian dari sistem manajemen data siswa yang lebih besar, yang memanfaatkan pemrograman berorientasi objek untuk pengelolaan data.

-Tampilan Tambah Data



The image shows a web form titled "TAMBAH DATA HOBI SISWA" (Add Student Hobby Data). The form is centered on a light blue background. It contains four input fields, each with a label above it: "Nama" (Name), "Kelas" (Class), "Hobi" (Hobby), and "Profile URL". Each input field has a placeholder text "Masukkan [field name]" (Enter [field name]). Below the input fields is a blue button labeled "Simpan" (Save).

E. Edit Data

- edit.php

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<title>Edit</title>
<style>
/* Styling untuk body */
body {
font-family: Arial, sans-serif;
background-color: #f0f0f5;
margin: 0;
padding: 20px;
display: flex;
justify-content: center;
align-items: center;
height: 100vh;
}
/* Container form */
form {
```

```
background-color: #fff;
border-radius: 10px;
box-shadow: 0 0 15px rgba(0, 0, 0, 0.2);
padding: 30px;
width: 350px;
}
/* Judul */
h1 {
text-align: center;
color: #333;
margin-bottom: 40px;
width: 420px;
}
/* Style untuk label input */
label {
font-size: 16px;
color: #555;
}
/* Style untuk input field */
input[type="number"], input[type="text"] {
width: 100%;

padding: 8px;
margin: 10px 0 20px;
border: 1px solid #ccc;
border-radius: 5px;
}
/* Style untuk tombol kirim */
input[type="submit"] {
width: 100%;
background-color: #2068E4;
color: white;
padding: 10px;
border: none;
border-radius: 5px;
cursor: pointer;
font-size: 16px;
}
/* Hover efek untuk tombol */
input[type="submit"]:hover {
background-color: #0A285A;
}
</style>
</head>
<body>
```

```

<h1>Edit Data Hobi Siswa</h1>
<form action="./edit_action.php" method="POST">
Id: <input type="number" name="id" id="id" min="0" <?php $x =
$_GET["ID"];if ($x !== null) echo "value='$x'"; ?>><br>
Nama: <input type="text" name="nama" id="nama" <?php $x =
$_GET["Nama"];if ($x !== null) echo "value='$x'"; ?>><br>
Posisi: <input type="text" name="kelas" id="kelas" <?php $x =
$_GET["Kelas"];if ($x !== null) echo "value='$x'"; ?>><br>
Hobi: <input type="text" name="hobi" id="hobi" <?php $x =
$_GET["Hobi"];if ($x !== null) echo "value='$x'"; ?>><br>
Profile: <input type="text" name="profile" id="profile" <?php $x =
$_GET["Profile"] ?? "";if ($x !== null) echo "value='$x'"; ?>><br>
<input type="submit" value="Kirim">
</form>
</body>
</html>

```

Program PHP ini digunakan untuk mengedit data hobi siswa yang tersimpan dalam database. Program ini menyertakan dua file eksternal, yaitu `connect.php` untuk mengatur koneksi database dan `hobi.php` yang berisi kelas `hobi` dengan metode `edit()` untuk memperbarui data. Setelah koneksi database diinisialisasi melalui objek `\$database`, dibuat objek `\$hobi` dengan parameter koneksi tersebut. Data siswa yang akan diperbarui (seperti `id`, `nama`, `kelas`, `hobi_siswa`, dan `profile`) diambil dari form HTML menggunakan metode POST dan disimpan dalam properti objek `\$hobi`. Program kemudian memanggil metode `\$hobi->edit()`, yang akan mengembalikan nilai `true` jika berhasil atau `false` jika gagal, dan hasil ini disimpan dalam variabel `\$hasil`. Berdasarkan nilai `\$hasil`, program menampilkan pesan "data berhasil di edit." atau "data tidak bisa di edit." untuk memberi tahu pengguna tentang status pembaruan data. Di akhir, program juga menampilkan tautan kembali ke halaman utama ("Back Home") untuk memudahkan navigasi pengguna.

- edit_action.php

```
<?php
include_once 'connect.php';
include_once 'hobi.php';

$database = new Database();
$db = $database->getConnection();
$hobi = new hobi($db);
$hobi->id = $_POST['id'];
$hobi->nama = $_POST['nama'];
$hobi->kelas = $_POST['kelas'];
$hobi->hobi_siswa = $_POST['hobi'];
$hobi->profile = $_POST['profile'];
$hasil = $hobi->edit();
echo "<h1>";
if($hasil) {
echo "data berhasil di edit.";
} else {
echo "data tidak bisa di edit.";
}
echo "</h1>";
echo " - <a href=\"../\">Back Home</a>";
?>
```

Program PHP ini bertanggung jawab untuk mengedit data hobi siswa yang sudah ada di dalam database. Pertama, program mengimpor file koneksi ('connect.php') dan file kelas 'Hobi' ('hobi.php'), kemudian membuat objek 'Database' untuk mendapatkan koneksi ke database. Selanjutnya, objek 'Hobi' diinisialisasi dengan koneksi tersebut. Program kemudian mengambil data yang dikirim melalui metode POST, termasuk 'id', 'nama', 'kelas', 'hobi', dan 'profile', dan menentukannya ke properti yang sesuai pada objek 'Hobi'. Setelah data ditetapkan, metode 'edit()' dipanggil untuk memperbarui data siswa di dalam tabel database. Program kemudian menampilkan hasil proses pengeditan; jika berhasil, akan muncul pesan "data berhasil di edit.", dan jika gagal, akan muncul pesan "data tidak bisa di edit." Akhirnya, terdapat tautan yang memungkinkan pengguna untuk kembali ke halaman utama. Program ini merupakan bagian penting dari sistem manajemen data siswa, memungkinkan pengguna untuk memperbarui informasi siswa secara efisien.

- Tampilan edit data

[illegible]

F. Delete Data

- delete.php

```
<?php
include_once 'connect.php';
include_once 'hobi.php';

$databse = new Database();
$db = $databse->getConnection();

$hobi = new hobi($db);
$hobi->id = $_GET["ID"];

$hasil = $hobi->delete();

if ($hasil) header("Location: ./");
else echo "<h1>Data Gagal Dihapus<h1><br> - <a href=\"./\">Back
Home</a>";
?>
```

Program PHP ini berfungsi untuk menghapus data hobi siswa dari database. Pertama, program mengimpor file koneksi ('connect.php') dan file kelas 'Hobi' ('hobi.php'), lalu membuat objek 'Database' untuk mendapatkan koneksi ke database. Setelah itu, objek 'Hobi' diinisialisasi dengan koneksi yang diperoleh. Program mengambil parameter 'ID' dari URL yang dikirim menggunakan metode GET, yang menunjukkan ID data hobi siswa yang akan dihapus, dan menentukannya pada properti 'id' objek 'Hobi'. Metode 'delete()' kemudian dipanggil untuk menghapus data yang sesuai dari tabel database. Jika penghapusan berhasil, program akan mengalihkan pengguna kembali ke halaman utama menggunakan header "Location: ./". Namun, jika penghapusan gagal, program akan menampilkan pesan "Data Gagal Dihapus" diikuti dengan tautan untuk kembali ke halaman utama. Dengan demikian, program ini menyediakan fungsionalitas penting dalam sistem manajemen data siswa, memungkinkan pengguna untuk menghapus data siswa dengan mudah.

- delete_action.php

```
<?php
include_once 'connect.php';
include_once 'hobi.php';

$databse = new Database();
$db = $databse->getConnection();

$hobi = new hobi($db);
$hobi->id = $_GET['ID'];

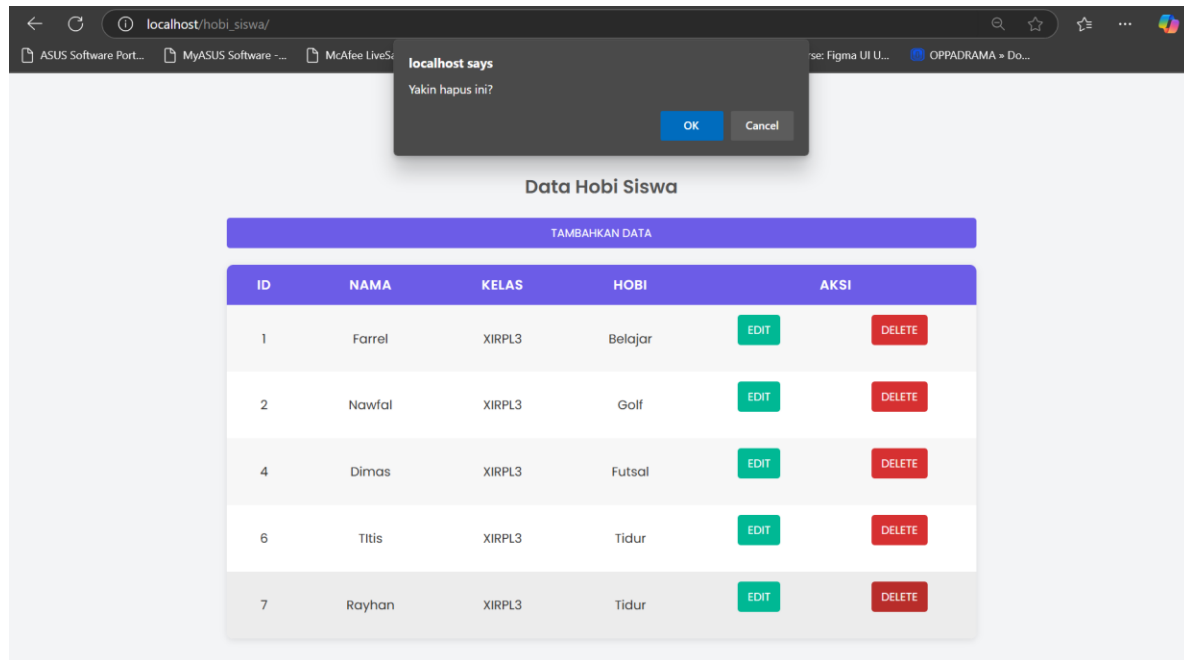
$hasil = $hobi->delete();

if($hasil) {
    header('location: ./');
} else {
    echo "<h1>";
    echo "data tidak bisa di hapus.";
    echo "</h1>";
    echo " - <a href=\"./\">Back Home</a>";
}

?>
```

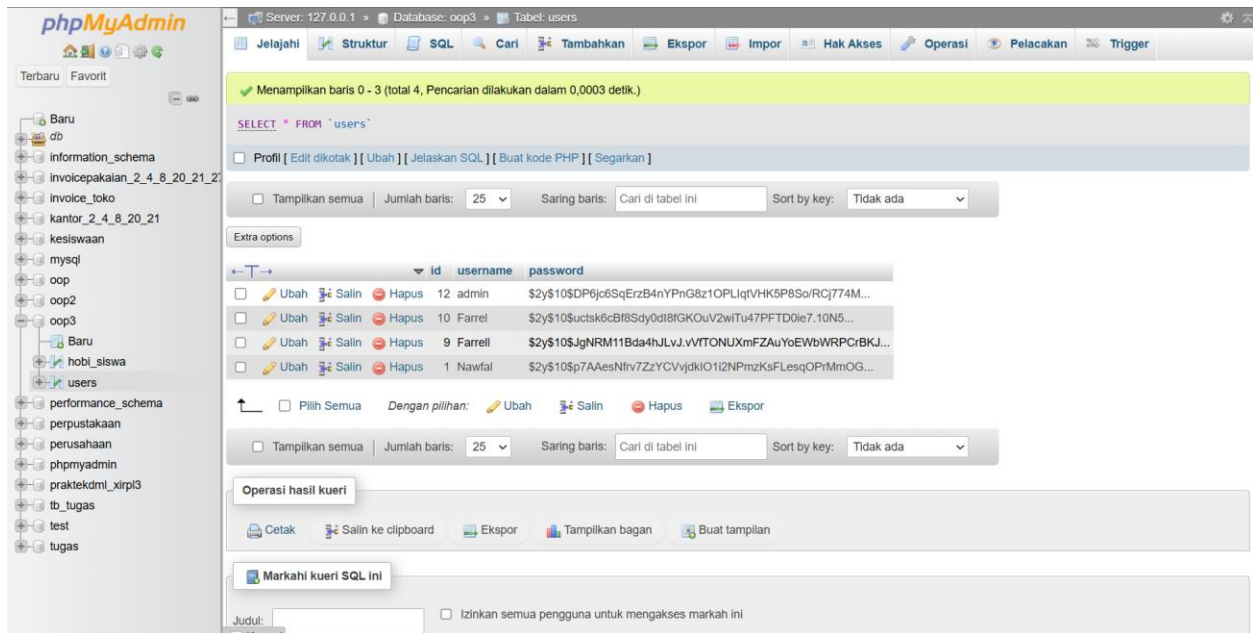
Program PHP ini dirancang untuk menghapus data hobi siswa dari database. Pertama, program mengimpor file koneksi ('connect.php') dan file kelas 'Hobi' ('hobi.php'), kemudian membuat objek 'Database' untuk mendapatkan koneksi ke database. Selanjutnya, objek 'Hobi' diinisialisasi dengan koneksi yang diperoleh, dan ID data yang ingin dihapus diambil dari parameter URL menggunakan metode GET. ID tersebut kemudian disimpan ke dalam properti 'id' dari objek 'Hobi'. Setelah itu, metode 'delete()' dipanggil untuk melakukan penghapusan data yang bersangkutan dari tabel database. Jika penghapusan berhasil, pengguna akan dialihkan kembali ke halaman utama dengan perintah 'header('location: ./')'. Namun, jika penghapusan gagal, program akan menampilkan pesan "data tidak bisa di hapus." diikuti dengan tautan untuk kembali ke halaman utama. Program ini merupakan bagian penting dari sistem manajemen data siswa, memungkinkan pengguna untuk menghapus informasi siswa dengan mudah dan efektif.

- Tampilan menghapus data



II. Login

A. Membuat tabel baru dengan nama users



Untuk meningkatkan keamanan dan manajemen akses dalam aplikasi, perlu dibuat tabel baru di dalam database `oop3` dengan nama `users`, yang akan berfungsi untuk menyimpan data login pengguna. Tabel ini akan memiliki struktur yang terdiri dari empat kolom: `id`, yang berfungsi sebagai kunci utama untuk mengidentifikasi setiap pengguna secara unik; `username`, yang menyimpan nama pengguna yang digunakan untuk login; dan `password`, yang menyimpan kata sandi pengguna dalam bentuk terenkripsi untuk melindungi informasi sensitif. Dengan adanya tabel `users`, aplikasi dapat mengelola otentikasi pengguna dengan lebih baik, memastikan hanya pengguna yang terdaftar yang dapat mengakses sistem, serta memberikan lapisan keamanan tambahan terhadap data yang dikelola.

B. Buat folder baru dengan nama Login di httdocs

Name	Date modified	Type	Size
dashboard	8/5/2024 7:20 AM	File folder	
hobi_siswa	10/29/2024 5:43 PM	File folder	
img	8/5/2024 7:20 AM	File folder	
LatihanPhp	10/4/2024 8:26 AM	File folder	
LatihanPHP2	9/30/2024 2:41 PM	File folder	
Login	10/30/2024 8:24 AM	File folder	
webalizer	8/5/2024 7:20 AM	File folder	
xampp	8/5/2024 7:20 AM	File folder	
applications	6/15/2022 11:07 PM	Microsoft Edge HTM...	4 KB
bitnami	6/15/2022 11:07 PM	CSS Source File	1 KB
Catatan hari kamis 5 september	9/5/2024 8:14 AM	Text Document	1 KB
favicon	7/16/2015 10:32 PM	ICO File	31 KB
form	8/11/2024 8:55 AM	Microsoft Edge HTM...	2 KB

C. Buat file koneksi dengan nama connect.php

```
<?php
class Database {
    private $host = "localhost";
    private $db_name = "oop3";
    private $username = "root";
    private $password = "";

    public $conn;
    public function getConnection() {
        $this->conn = null;
        try {
            $this->conn = new PDO("mysql:host=" . $this->host . ";dbname="
            . $this->db_name, $this->username, $this->password);
            $this->conn->exec("set names utf8");
            $this->conn->setAttribute(PDO::ATTR_ERRMODE,
PDO::ERRMODE_EXCEPTION);
        } catch(PDOException $exception) {
            echo "Connection error:" . $exception->getMessage();
        }
        return $this->conn;
    }
}

?>
```

Program PHP ini mendefinisikan kelas `Database`, yang bertanggung jawab untuk mengelola koneksi ke database MySQL. Kelas ini memiliki beberapa properti privat, termasuk `host`, `db_name`, `username`, dan `password`, yang menyimpan informasi penting untuk menghubungkan ke database `oop3`. Metode `getConnection()` digunakan untuk membuat dan mengembalikan koneksi ke database. Di dalam metode ini, objek PDO diinisialisasi untuk mengatur koneksi, dan jika koneksi berhasil, pengkodean karakter diatur ke UTF-8 untuk mendukung karakter internasional. Selain itu, pengaturan atribut PDO `ATTR_ERRMODE` diatur ke `ERRMODE_EXCEPTION`, yang memastikan bahwa kesalahan koneksi akan dilempar sebagai pengecualian, memungkinkan penanganan kesalahan yang lebih baik. Jika terjadi kesalahan saat menghubungkan, pesan kesalahan akan ditampilkan kepada pengguna. Kelas ini merupakan komponen penting dalam arsitektur aplikasi berbasis PHP yang berorientasi objek, memfasilitasi interaksi antara aplikasi dan database.

D. Buat file dengan nama dashboard.php

```
<?php
//index.php
require_once 'config/connect.php';
require_once 'classes/user.php';

$database = new Database();
$db = $database->getConnection();
$user = new User($db);

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

    if($user->login($username, $password)) {
        header("Location: /hobi_siswa");
        exit();
    } else {
        $error_message = "Username atau password salah!";
    }
}
?>
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    <title>Login</title>
    <link
href="https://cdn.jsdelivrivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.
min.css" rel="stylesheet" integrity="sha384-
QWTKZyjpPEjISv5WaRU90FeRpok6YctnYmDr5pNlyT2bRjXh0JMhjY6hW+ALEwIH"
crossorigin="anonymous">
</head>
<body class=" ">
    <div class="container mt-5">
        <div class="row justify-content-center">
            <div class="col-md-4">
                <h3 class="text-center">Login</h3>
                <?php if(isset($error_message)): ?>
                    <div class="alert alert-danger"><?php echo
$error_message; ?></div>
```



```

        <?php endif; ?>
        <form action="" method="POST" class="">
            <div class="form-group">
                <label>Username</label>
                <input type="text" name="username"
class="form-control" required>
            </div>
            <div class="form-group">
                <label>Password</label>
                <input type="password" name="password"
class="form-control" required>
            </div><br>
            <div class="d-flex justify-content-sm-between">
                <button type="submit" class="btn btn-primary
btn-block align-middle">Login</button>
                <a href="forgot.php">Forgot Password</a>
            </div>
            <div>
                <a>Don't have an account?</a><a
href="register.php"> register now</a>
            </div>
        </form>
</body>
</html>

```

Program PHP ini adalah halaman login yang menggunakan arsitektur berbasis objek untuk mengotentikasi pengguna. Pada awalnya, program mengimpor file konfigurasi koneksi database (`connect.php`) dan kelas `User` yang berisi logika terkait pengguna. Koneksi ke database dibuat dengan memanggil metode `getConnection()` dari objek `Database`, dan objek `User` diinisialisasi menggunakan koneksi tersebut. Jika permintaan yang diterima adalah metode POST, program mengambil nilai `username` dan `password` dari formulir yang diisi. Kemudian, metode `login()` dari objek `User` dipanggil untuk memeriksa kredensial pengguna. Jika login berhasil, pengguna akan dialihkan ke halaman utama aplikasi menggunakan perintah `header("Location: /hobi_siswa")`. Jika login gagal, pesan kesalahan ditampilkan untuk memberi tahu pengguna bahwa username atau password salah. Halaman HTML dirancang dengan Bootstrap untuk memastikan antarmuka pengguna yang responsif dan menarik. Formulir login terdiri dari dua input untuk username dan password, serta tautan untuk reset password dan pendaftaran akun baru, menciptakan pengalaman pengguna yang lengkap dan fungsional.

F. buat file user.php

```
<?php
//clases/user.php
require_once 'config/connect.php';

class User{
    private $conn;
    private $table_name = "users";

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

    /**
     * mengecek kredensial login
     * @param string $username
     * @param string $password
     * @return boolean
     */

    public function login($username, $password) {
        //Query untuk mengambil data user yang sesuai dengan username
        yang diinput
        $query = "SELECT * FROM " . $this->table_name . " WHERE
        username = :username LIMIT 1";
        $stmt = $this->conn->prepare($query);
        $stmt->bindParam(":username", $username);
        $stmt->execute();

        //Jika ada data user yang sesuai dengan username yang diinput
        if($stmt->rowCount() > 0){
            //ambil data user yang sesuai dengan username yang diinput
            $row = $stmt->fetch(PDO::FETCH_ASSOC);
            //Jika password yang diinput sesuai dengan password yang
            tersimpan di database
            if(password_verify($password, $row['password'])){
                //mulai sesi dan simpan username
                session_start();
                $_SESSION['username'] = $username;
                //kembalikan true
                return true;
            }
        }
    }
}
```

```

        //kembalikan false jika tidak ada data user yang sesuai dengan
        username yang diinput atau password yang diinput tidak sesuai dengan
        password yang tersimpan di database
        return false;
    }

    //mfungsi menambah user baru
    public function register($username, $password) {
        $query = "INSERT INTO " . $this->table_name . " (username,
password) VALUES (:username, :password)";
        $stmt = $this->conn->prepare($query);

        //enkripsi password sebelum disimpan
        $hashed_password = password_hash($password, PASSWORD_DEFAULT);

        $stmt->bindParam(":username", $username);
        $stmt->bindParam(":password", $hashed_password);

        try {
            if($stmt->execute()) {
                return true;
            }
        } catch(PDOException $exception) {
            return false;
        }

        return false;
    }
}

```

Kelas `User` dalam program PHP ini bertugas mengelola otentikasi dan pendaftaran pengguna dalam aplikasi berbasis database. Kelas ini memuat koneksi database melalui `connect.php` dan mendefinisikan nama tabel `users`. Konstruktor menyimpan objek database dalam properti privat `\$conn`. Metode `login()` memverifikasi kredensial pengguna dengan mencari username dan membandingkan password yang dimasukkan dengan yang tersimpan di database menggunakan `password_verify()`. Jika cocok, sesi dimulai dan username disimpan dalam session; jika tidak, metode ini mengembalikan false. Metode `register()` digunakan untuk menambah pengguna baru, dengan menyimpan username dan password yang telah di-hash ke dalam database. Jika penyimpanan berhasil, metode ini mengembalikan true; jika gagal, nilai false dikembalikan. Kelas ini mengintegrasikan keamanan dengan menggunakan hashing untuk password, memastikan pengelolaan data pengguna yang aman dalam aplikasi. G. buat file register.php

```

<?php
//register.php
require_once './config/connect.php';
require_once './classes/user.php';

$databse = new Database();
$db = $databse->getConnection();
$user = new User($db);

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

    if($user->register($username, $password)) {
        $sucsess_message = "Registrasi berhasil, Silahkan login.";
    } else {
        $error_message = "Registrasi gagal, Username mungkin sudah
digunakan.";
    }
}
?>

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    <title>Register</title>
    <link
href="https://cdn.jsdelivrivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.
min.css" rel="stylesheet" integrity="sha384-
QWTKZyjpPEjISv5WaRU90FeRpok6YctnYmDr5pNlyT2bRjXh0JMHjY6hW+ALEwIH"
crossorigin="anonymous">
</head>
<body>
    <div class="container mt-5">
        <div class="row justify-content-center">
            <div class="col-md-4">
                <h3 class="text-center">Registration</h3>
                <?php if(isset($error_message)): ?>
                    <div class="alert alert-danger"><?php echo
$error_message; ?></div>
                <?php endif; ?>
                <?php if(isset($sucsess_message)): ?>

```

```

        <div class="alert alert-success"><?php echo
$succsess_message; ?></div>
        <?php endif; ?>
        <form action="" method="POST" class="">
            <div class="form-group">
                <label>Username</label>
                <input type="text" name="username"
class="form-control" required>
            </div>
            <div class="form-group">
                <label>Password</label>
                <input type="password" name="password"
class="form-control" required>
            </div><br>
            <div>
                <button type="submit" class="btn btn-primary
btn-block align-middle">Register</button>
            </div>
            <div>
                <a>Already have an account?</a><a
href="dashboard.php"> sign in now</a>
            </div>
        </form>
    </body>
</html>

```

Halaman register.php bertujuan untuk mengelola proses pendaftaran pengguna baru dalam aplikasi berbasis PHP. Pertama, file ini menyertakan koneksi ke database dan kelas User, yang bertanggung jawab untuk interaksi pengguna. Ketika pengguna mengisi formulir pendaftaran dan mengirimkan data, server memeriksa metode permintaan dan mengambil username dan password dari input. Metode register() dalam kelas User kemudian dipanggil untuk menyimpan data pengguna ke dalam database. Jika pendaftaran berhasil, pesan sukses ditampilkan; jika gagal—misalnya, jika username sudah digunakan—pesan kesalahan ditampilkan. Antarmuka pengguna dibangun menggunakan Bootstrap untuk memastikan tampilan yang responsif dan menarik, dengan formulir yang mencakup kolom untuk username dan password, serta tombol untuk mengirimkan data. Terdapat juga tautan bagi pengguna yang sudah memiliki akun untuk langsung masuk ke halaman login.

III. Hasil

a. Dashboard

Login

Username

Password

Login

[Forgot Password](#)

Don't have an account? [register now](#)

b. Register

Registration

Username

Password

Register

Already have an account? [sign in now](#)

c. Registrasi berhasil

Registration

Registrasi berhasil, Silahkan login.

Username

Password

[Register](#)

Already have an account? [sign in now](#)

d. Registrasi gagal

Registration

Registrasi gagal, Username mungkin sudah digunakan.

Username

Password

[Register](#)

Already have an account? [sign in now](#)






e. Login berhasil

CRUD OOP PHP

Data Hobi Siswa

LOG OUT

TAMBAHKAN DATA

ID	PROFILE	NAMA	KELAS	HOBI	AKSI	
1		Noval	XIRPL3	Melihat Kecoak	EDIT	DELETE
2		Nawfal	XIRPL3	Golf	EDIT	DELETE
4		Dimas	XIRPL3	Futsal	EDIT	DELETE
6		Titis	XIRPL3	Tidur	EDIT	DELETE
8		Rayhan	XITB2	Tidur	EDIT	DELETE

f. Login gagal

Login

Username atau password salah!

Username

Password

Login

[Forgot Password](#)

Don't have an account? [register now](#)

g. Log out

The screenshot shows a web browser at `localhost/hobi_siswa/`. A modal dialog titled "localhost says" is displayed, asking "Apakah anda yakin ingin log out?" with "OK" and "Cancel" buttons. Below the dialog, the page title is "Data Hobi Siswa" with a "LOG OUT" button. A table titled "TAMBAHKAN DATA" contains student hobby data.

ID	PROFILE	NAMA	KELAS	HOBİ	AKSI
1		Noval	XIRPL3	Melihat Kecoak	EDIT DELETE
2		Nawfal	XIRPL3	Golf	EDIT DELETE
4		Dimas	XIRPL3	Futsal	EDIT DELETE
6		Titis	XIRPL3	Tidur	EDIT DELETE
8		Rayhan	XITB2	Tidur	EDIT DELETE

Login

Username

Password

[Login](#) [Forgot Password](#)

Don't have an account? [register now](#)