

Laporan Praktikum CRUD Dan Login PHP Native



Nama Anggota Kelompok :

- Galih Rahdatu Ananta (12)
- M Nazwa Arya (16)
- M. Darun Naim (18)
- Nur Cahyono (25)

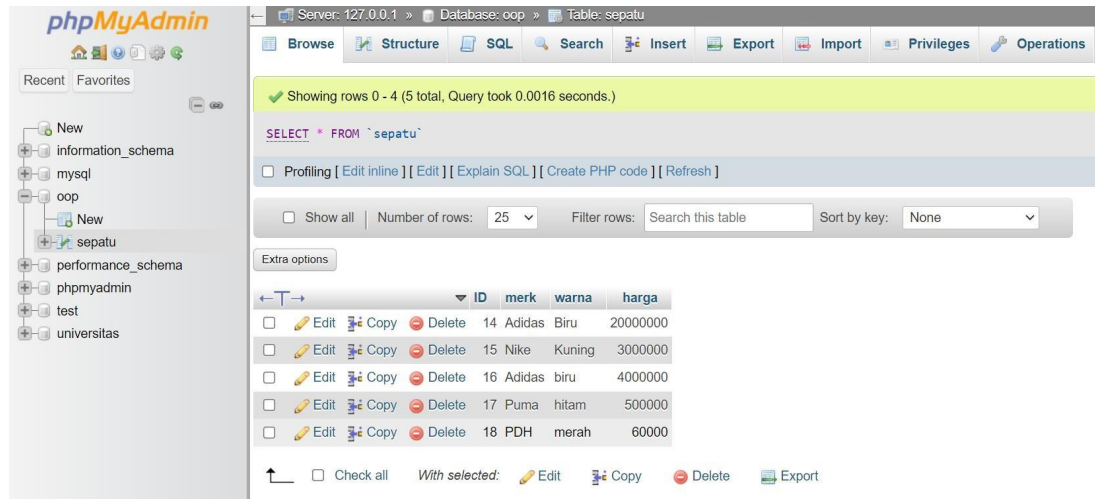
REKAYASA PERANGKAT LUNAK

SMK TEXMACO SEMARANG

2024

I. Tahapan

A. Membuat database & tabel



B. Membuat koneksi

```
<?php
class oop {
    private $host =
        "localhost"; private
    $db_name = "oop"; private
    $username = "root";
    private $password = "";
    public $conn;

    public function getConnection() {
        $this->conn = null;
        try{
```

```

    } catch (PDOException $expection) {
        echo "Connection error:" . $expection->getMessage();
    }
    return $this->conn;
}
?>

```

PENJELASAN : Kode PHP di atas mendefinisikan kelas `oop`, yang dirancang untuk mengelola koneksi ke database MySQL menggunakan PDO (PHP Data Objects). Kelas ini memiliki beberapa properti privat untuk menyimpan informasi koneksi, termasuk host, nama database, nama pengguna, dan kata sandi, serta satu properti publik `$conn` untuk menyimpan objek koneksi. Metode `getConnection()` berfungsi untuk membuat koneksi ke database, namun saat ini kode tersebut belum lengkap karena pernyataan untuk menginisialisasi objek PDO dengan string koneksi masih terputus. Jika koneksi berhasil, objek PDO akan disimpan dalam `$conn`; jika gagal, exception akan ditangkap dan pesan kesalahan akan ditampilkan. Dengan demikian, kelas ini memberikan kerangka kerja dasar untuk menghubungkan aplikasi PHP ke database, meskipun memerlukan perbaikan pada bagian yang hilang.

4o mini

C. Menampilkan Data

```

    echo"<td><button onclick=\"if (confirm('yakin hapus ini?'))
window.location.href = './delete_action.php?ID=" . $ID . "';\"
type='button' class='red'>Delete</button></td>";
    echo"</tr>";
}
?>

</body>
</html>

```

PENJELASAN : Kode PHP di atas menghasilkan sebuah elemen tabel HTML yang berisi tombol "Delete". Tombol ini dilengkapi dengan fungsi JavaScript yang memunculkan kotak konfirmasi ketika diklik, menanyakan pengguna apakah mereka yakin ingin menghapus entri terkait. Jika pengguna mengklik "OK", halaman akan dialihkan ke `delete_action.php` dengan parameter query `ID`, yang berisi ID entri yang ingin dihapus. Jika pengguna mengklik "Batal", tidak ada tindakan lebih lanjut yang dilakukan. Dengan demikian, kode ini memberikan cara interaktif untuk menghapus data sambil memastikan bahwa pengguna tidak melakukan tindakan tersebut secara tidak sengaja.

4o mini

Hasil :

Data Sepatu				Add Data Logout	
ID	MERK	WARNA	HARGA		
1	NIKE	HITAM	Rp.100,000,000	Edit	Delete
2	ADIDAS	BIRU MERAH	Rp.2,147,483,647	Edit	Delete
3	JORDAN	PUTIH	Rp.30,000,000	Edit	Delete
4	ATT	HITAM PUTIH	Rp.2,147,483,647	Edit	Delete
5	PDH	COKLAT	Rp.500,000,000	Edit	Delete

D. Menambah Data

Create :

```
<!DOCTYPE html>
<html>
  <head>
    <title>Tambah Sepatu</title>
    <link rel="stylesheet" href="form.css">
  </head>
  <body>
    <h1>Tambah Sepatu </h1>
    <form action="create_action.php"
    method="post"> Merk:<input type="text"
    name="merk"><br> Warna:<input type="text"
    name="warna"><br> Harga:<input type="number"
    name="harga"><br>
    <input type="submit" value="Simpan">
```

PENJELASAN : Kode HTML di atas mendefinisikan sebuah halaman untuk menambahkan sepatu dengan judul "Tambah Sepatu". Halaman ini berisi sebuah formulir yang meminta pengguna untuk mengisi informasi mengenai sepatu, termasuk merk, warna, dan harga, menggunakan elemen input yang sesuai. Metode pengiriman formulir ditetapkan sebagai 'POST', sehingga data akan dikirim ke skrip 'create_action.php' saat tombol "Simpan" ditekan. Terdapat juga penghubungan ke file CSS eksternal 'form.css' untuk penataan gaya halaman. Secara keseluruhan, halaman ini dirancang untuk memudahkan pengguna dalam memasukkan data sepatu baru ke dalam sistem.

Create Action:

```
<?php
include_once 'config.php';
```

```
include_once 'sepatu.php';

$database = new oop();
$db = $database->getConnection();
```

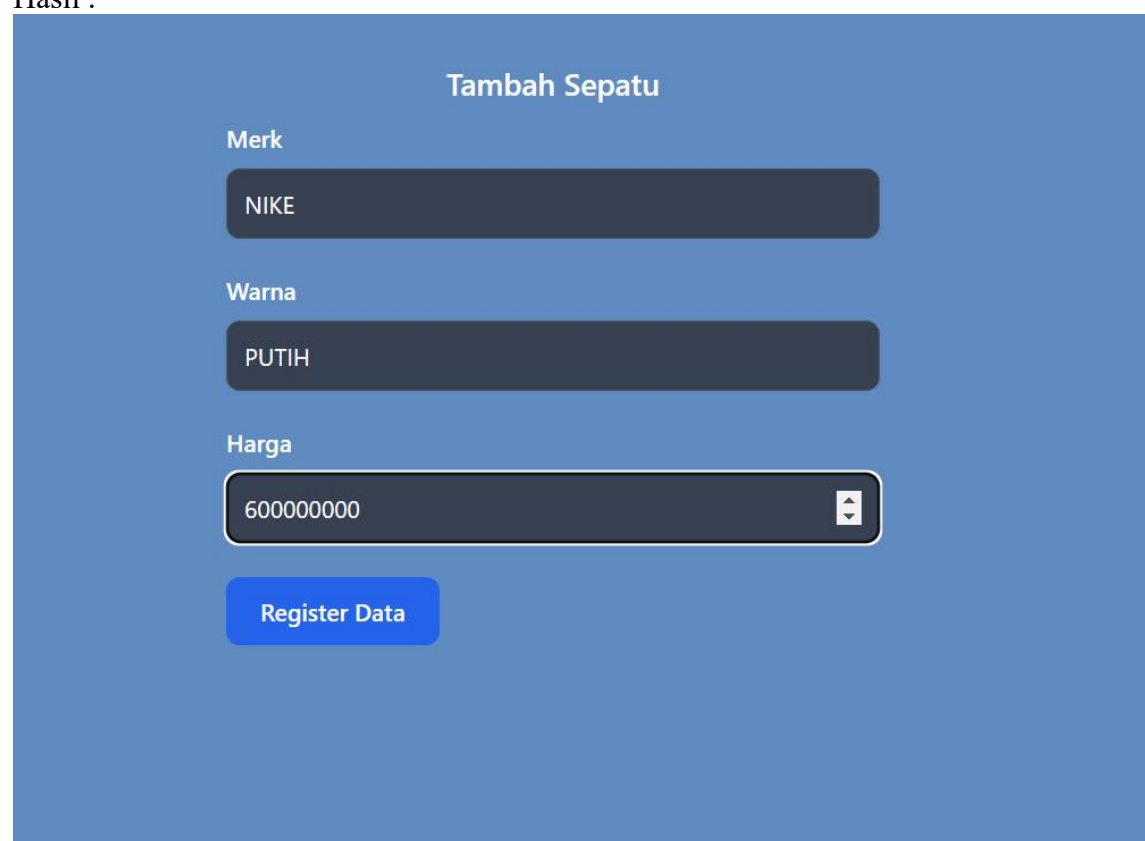
```
$sepatu = new Sepatu($db);
```

```
$sepatu->merk = $_POST['merk'];
$sepatu->warna = $_POST['warna'];
$sepatu->harga = $_POST['harga'];
```

```
if($sepatu->create()) {
    echo "Sepatu berhasil ditambahkan.";
} else {
    echo "Gagal menambahkan sepatu.";
}
?>
<br>
<a href="index.php">Kembali</a>
```

PENJELASAN : Kode PHP di atas mengelola proses penambahan sepatu ke dalam database setelah formulir diisi. Pertama, skrip ini menyertakan dua file penting: `config.php` untuk konfigurasi koneksi database dan `sepatu.php`, yang mungkin mendefinisikan kelas `Sepatu`. Setelah membuat objek dari kelas `oop` untuk mendapatkan koneksi database, objek `Sepatu` juga dibuat menggunakan koneksi tersebut. Kemudian, atribut objek `Sepatu` diisi dengan data yang diterima dari formulir (merk, warna, dan harga) melalui metode POST. Selanjutnya, skrip mencoba untuk menyimpan data sepatu ke dalam database dengan memanggil metode `create()`. Jika berhasil, pesan "Sepatu berhasil ditambahkan." akan ditampilkan; jika gagal, pesan "Gagal menambahkan sepatu." akan muncul. Terakhir, terdapat tautan untuk kembali ke halaman utama (`index.php`). Kode ini secara keseluruhan memungkinkan pengguna untuk menyimpan data sepatu baru secara efektif.

Hasil :



Tambah Sepatu

Merk

NIKE

Warna

PUTIH

Harga

600000000

Register Data

E. Edit Data

```
<?php
include_once 'config.php';
include_once 'sepatu.php';

$database = new oop();
$db = $database->getConnection();
```

```
// Periksa apakah ada ID yang dikirim
if (isset($_GET['id'])) {
    $sepatu->ID = $_GET['id'];
    $sepatu->readOne();// Ambil data berdasarkan ID untuk diisi di form
}
```

```

if ($_POST) {
    // Set data sepatu dari form
    $sepatu->merk = $_POST['merk'];
    $sepatu->warna = $_POST['warna'];
    $sepatu->harga = $_POST['harga'];

    // Jalankan update
    if ($sepatu->update()) {
        echo "<div>Data berhasil diubah!</div>";
    } else {
        echo "<div>Gagal mengubah data.</div>";
    }
}
?>

```

```

<!DOCTYPE html>
<html>
<head>
    <title>Update Data Sepatu</title>
    <link rel="stylesheet" href="form.css">
</head>
<body>
    <h1>Update Data Sepatu</h1>
    <form action="<?php echo htmlspecialchars($_SERVER["PHP_SELF"]) .
    "?id=" . $sepatu->ID); ?>" method="post">
        <label for="merk">Merk:</label>
        <input type="text" name="merk" value="<?php echo $sepatu-
>merk; ?>" required><br>

```

```

        <label for="warna">Warna:</label>
        <input type="text" name="warna" value="<?php echo $sepatu-
>warna; ?>" required><br>

```

```

        <label for="harga">Harga:</label>
        <input type="number" name="harga" value="<?php echo $sepatu-
>harga; ?>" required><br>

```

```

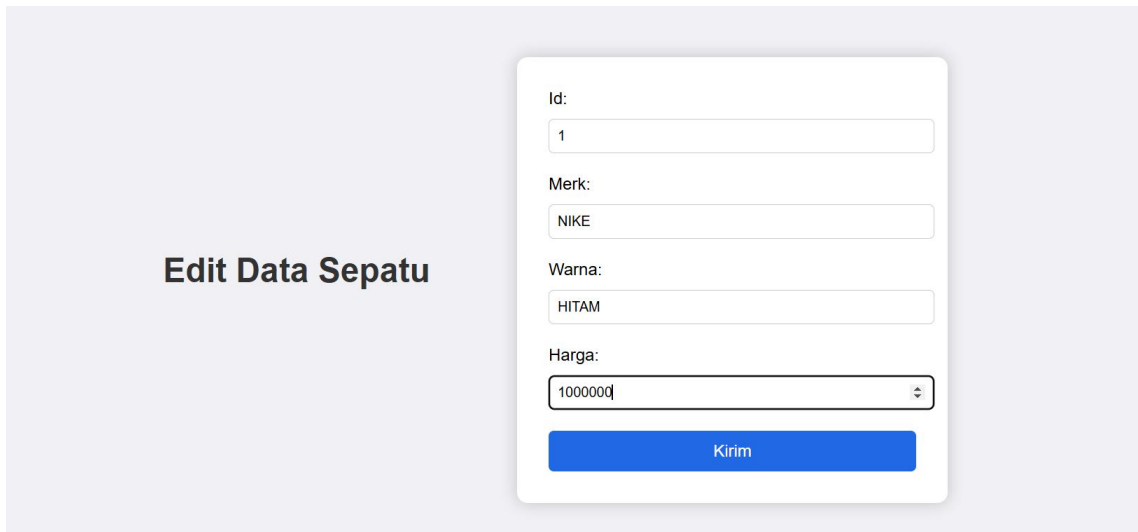
        <input type="submit" value="Update">
    </form>
    <br>
</body>
</html>

```

PENJELASAN : Kode PHP dan HTML di atas berfungsi untuk memperbarui data sepatu dalam aplikasi. Pertama, skrip PHP menyertakan file konfigurasi dan definisi kelas 'Sepatu', kemudian membuat koneksi ke database. Jika parameter 'id' ada di URL, data sepatu yang relevan diambil menggunakan metode 'readOne()' dan disiapkan untuk diisi dalam formulir. Dalam bagian HTML, formulir disusun untuk menerima input dari pengguna mengenai merk, warna, dan harga sepatu. Setiap input diisi dengan nilai yang ada saat ini untuk memudahkan pengguna dalam melakukan pembaruan. Formulir dikirim ke halaman yang sama dengan menambahkan parameter 'id' pada URL, dan setelah data dikirim melalui metode POST, perubahan akan disimpan ke database. Jika pembaruan berhasil, pesan keberhasilan ditampilkan, dan jika gagal, pesan kesalahan

muncul. Halaman ini secara keseluruhan menyediakan antarmuka yang interaktif dan aman bagi pengguna untuk memperbarui informasi sepatu

Hasil :



Edit Data Sepatu

Id:

Merk:

Warna:

Harga:

F. Delete Data

Delete :

```
<?php
include_once 'config.php';
include_once 'sepatu.php';

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

// Periksa apakah ada ID yang dikirim untuk dihapus
if (isset($_GET['id'])) {
    $sepatu->ID = $_GET['id'];

    if ($sepatu->delete()) {
        echo "<div>Data berhasil dihapus!</div>";
    } else {
        echo "<div>Gagal menghapus data.</div>";
    }
}
?>
<a href="index.php">Kembali</a>
```

PENJELASAN : Kode PHP di atas bertujuan untuk menghapus data sepatu dari database. Pertama, skrip ini menyertakan file konfigurasi dan definisi kelas `Sepatu`, serta membuat koneksi ke database menggunakan kelas `oop`. Selanjutnya, kode memeriksa apakah parameter `id` ada dalam URL. Jika ada, nilai `id` tersebut diassign ke atribut `ID` dari objek `sepatu`. Kemudian, metode `delete()` dipanggil untuk menghapus data sepatu yang bersangkutan. Jika penghapusan berhasil, pesan "Data berhasil dihapus!" ditampilkan; jika tidak, pesan "Gagal menghapus data." muncul sebagai umpan balik kepada pengguna. Terakhir, terdapat tautan untuk kembali ke halaman utama (`index.php`), memungkinkan navigasi yang mudah bagi pengguna. Dengan demikian, kode ini memberikan fungsi yang jelas dan langsung untuk menghapus data sepatu dari sistem.

Delete action :

```
<?php
include_once 'config.php';
include_once 'sepatu.php';

$databse = new oop();
$db = $databse->getConnection();
```

```
$sepatu = new Sepatu($db);
$sepatu->ID = $_GET['ID'];
```

```
$hasil = $sepatu->delete();
```

```

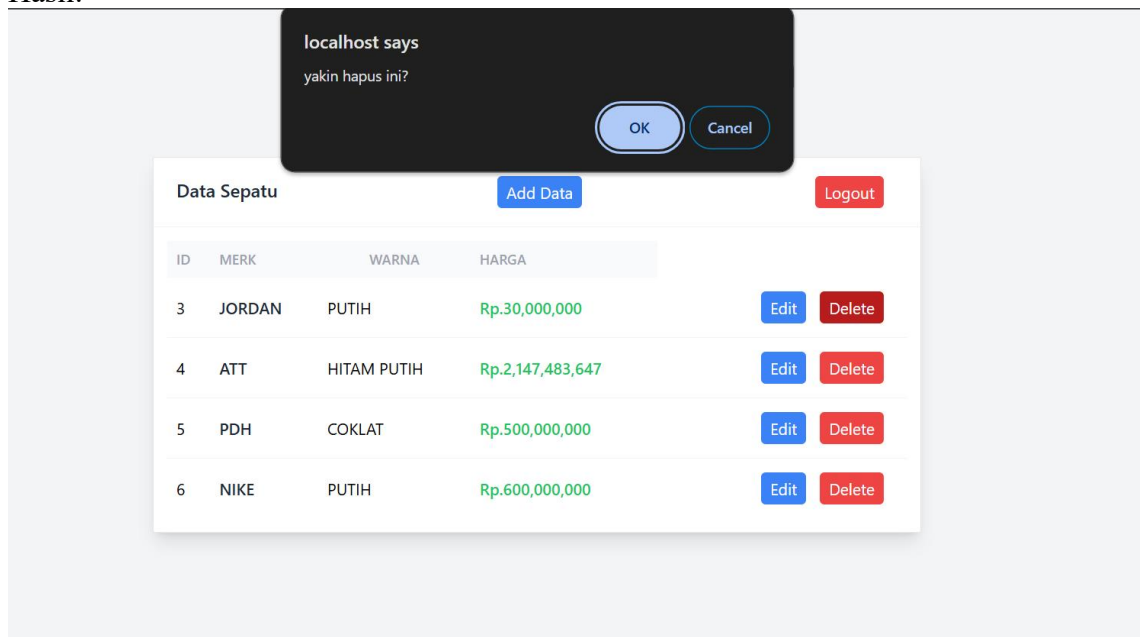
if($hasil) {
    header('location: ./');
} else {
    echo "<h1>";
    echo "data tidak bisa di hapus.";
    echo "</h1>";
    echo " - <a href=\"./\">Back Home</a>";
}

```

?>

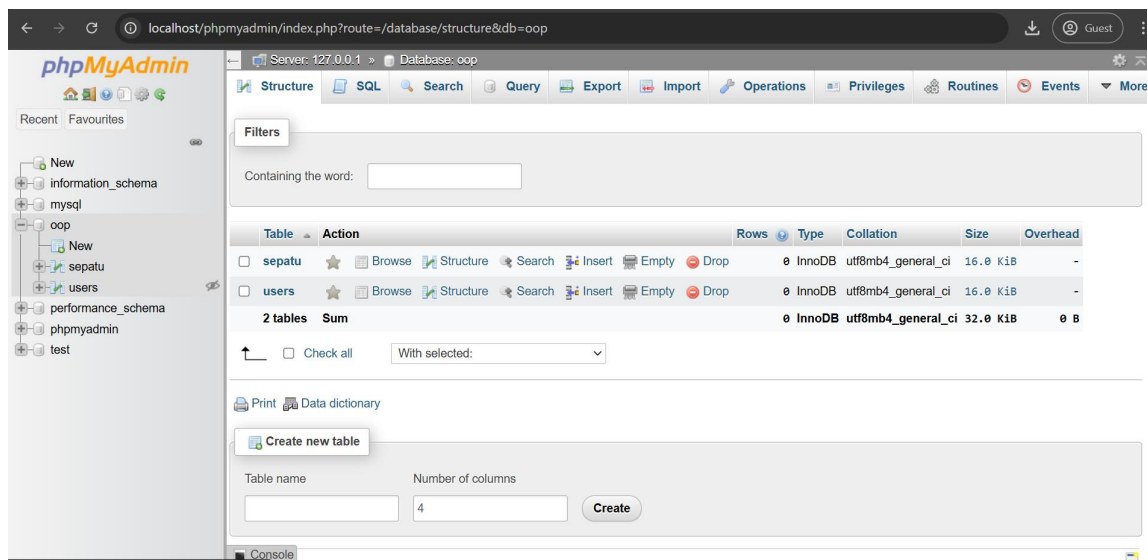
PENJELASAN : Kode PHP di atas bertujuan untuk menghapus data sepatu dari database berdasarkan ID yang diterima melalui parameter URL. Skrip ini dimulai dengan menyertakan file konfigurasi dan definisi kelas `Sepatu`, lalu membuat objek `oop` untuk mendapatkan koneksi database. Selanjutnya, objek `Sepatu` dibuat dengan koneksi yang sama, dan ID yang diterima dari URL diset ke atribut `ID` objek tersebut. Kemudian, metode `delete()` dipanggil untuk menghapus data sepatu yang bersangkutan. Jika penghapusan berhasil, pengguna akan dialihkan kembali ke halaman utama menggunakan `header('location: ./');`. Jika gagal, pesan "data tidak bisa di hapus." ditampilkan, bersama dengan tautan untuk kembali ke halaman utama. Dengan demikian, kode ini memberikan fungsi penghapusan yang efisien dan umpan balik yang jelas kepada pengguna.

Hasil:



II. Login

1. Membuat database di phpMyAdmin/xampp dengan nama oop yang berisi tabel bernama sepatu dan users. Contoh gambar dibawah ini



- Menyambungkan PHP ke database dengan membuat file bernama connect.php seperti berikut

```
<?php
class Database {
    private $host = "localhost";
    private $db_name = "oop";
    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;
    }
}
```

PENJELASAN : Kode PHP di atas mendefinisikan kelas 'Database' yang bertujuan untuk mengelola koneksi ke database MySQL menggunakan PDO (PHP Data Objects).

Kelas ini memiliki empat properti privat yang menyimpan informasi koneksi seperti host, nama database, nama pengguna, dan kata sandi. Metode `getConnection()` berfungsi untuk membuat koneksi ke database; jika berhasil, koneksi disimpan dalam properti publik `\$conn`, dan karakter set diatur ke UTF-8 untuk mendukung berbagai karakter. Jika terjadi kesalahan saat mencoba menghubungkan, pesan kesalahan ditangkap dan ditampilkan. Secara keseluruhan, kelas ini menyediakan cara yang terstruktur dan aman untuk menghubungkan aplikasi PHP ke database.

3. Membuat file dengan nama dashboard.php seperti berikut

```
<?php
include './connect.php';
include './sepatu.php';
$db = new Database();
$sepatu = new Sepatu ($db->getConnection());
?>
<script src="https://cdn.tailwindcss.com"></script>
<section class="antialiased bg-gray-100 text-gray-600 h-screen
px-4">
    <div class="flex flex-col justify-center h-full">
        <!-- Table -->
        <div class="w-full max-w-2xl mx-auto bg-white shadow-lg
rounded-sm border border-gray-200">
            <header class="px-5 py-4 border-b border-gray-100
flex justify-between">
                <h2 class="font-semibold text-gray-800">Data
Karyawan</h2>
                <a href="./create.php" class='mr-3 text-sm bg-
blue-500 hover:bg-blue-700 text-white py-1 px-2 rounded
focus:outline-none focus:shadow-outline'>Add Data</a>
            </header>
            <div class="p-3">
                <div class="overflow-x-auto">
                    <table class="table-auto w-full">
                        <thead class="text-xs font-semibold
uppercase text-gray-400 bg-gray-50">
                            <tr>
                                <th class="p-2 whitespace-
nowrap">
                                    <div class="font-semibold
text-left">ID</div>
                                </th>
                                <th class="p-2 whitespace-
nowrap">
                                    <div class="font-semibold
text-left">Name</div>
                                </th>
                                <th class="p-2 whitespace-
nowrap">
                                    <div class="font-semibold
text-center">Posisi</div>
                                </th>
```

```

        <th class="p-2 whitespace-
nowrap">
        <div class="font-semibold
text-left">Gaji</div>
        </th>
    </tr>
</thead>
<tbody class="text-sm divide-y divide-
gray-100">
    <?php
    $no = 1;
    foreach($sepatu->read() as $x){
        $id = $x["ID"];
        $merk = $x["merk"];
        $warna = $x["warna"];
        $harga = $x["harga"];
        $harga =
number_format($x["harga"]);
        $jenis = ["sepatu1.jpeg",
"sepatu2.jpeg", "sepatu3.jpeg", "sepatu4.jpeg", "sepatu5.jpeg"];
        $random =
$jenis[array_rand($jenis)];

        echo "
        <tr>
            <td class='p-2 whitespace-
nowrap'>
                <div class='text-
left'>$id</div>
            </td>
            <td class='p-2 whitespace-
nowrap'>
                <div class='flex items-
center'>
                    <div class='w-10 h-10
flex-shrink-0 mr-2 sm:mr-3'><img class='rounded-full'
src='https://github.com/alexarnoldrodrygo1426/Produktif/upload/m
ain/pict/$jenis' width='40' height='40' alt='$merk'></div>
                    <div class='font-
medium text-gray-800'>$merk</div>
                </div>
            </td>
            <td class='p-2 whitespace-
nowrap'>
                <div class='text-
left'>$warna</div>
            </td>
            <td class='p-2 whitespace-
nowrap'>
                <div class='text-left
font-medium text-green-500'>Rp.$harga</div>
            </td>

```

```
 >    PENJELASAN : Kode PHP dan HTML di atas membangun antarmuka untuk menampilkan data sepatu dalam bentuk tabel, menggunakan framework CSS Tailwind untuk styling. Pertama, skrip ini mengimpor file koneksi database dan kelas `Sepatu`, lalu membuat objek untuk mengakses data sepatu. Tabel di dalam elemen ` ` menampilkan informasi setiap sepatu, termasuk ID, merk, warna, dan harga, dengan gambar sepatu yang diambil secara acak dari daftar gambar. Pengguna dapat melihat semua data sepatu yang ada, serta memiliki opsi untuk menambah data baru dengan tautan "Add Data". Setiap baris dalam tabel dilengkapi dengan tombol "Edit" untuk mengubah informasi sepatu dan tombol "Delete" yang memunculkan konfirmasi sebelum menghapus data. Ini memberikan antarmuka yang interaktif dan mudah digunakan untuk mengelola data sepatu dalam aplikasi.  4. Membuat file dengan nama user.php seperti berikut  ```  <?php require_once './config/connect.php';  class User{     private $conn;     private $table_name = "users";      public function __construct($db) {         $this->conn = $db;     } }  ``` |
```

```
}
```

```
public function login($username, $password) {  
    $query = "SELECT * FROM " . $this->table_name . "WHERE  
username = :username LIMIT 1";
```

```
    $stmt = $this->conn->prepare($query);  
    $stmt->bindParam(":username", $username);  
    $stmt->execute();
```

```
    if($stmt->rowCount() > 0) {  
        $row = $stmt->fetch(PDO::FETCH_ASSOC);  
        if(password_verify($password, $row['password'])) {  
            return true;  
        }  
    }  
    return false;  
}
```

```
?>
```

PARAGRAF : Kode PHP di atas mendefinisikan kelas `User` yang bertanggung jawab untuk mengelola proses login pengguna. Kelas ini memiliki dua properti privat: `\$conn` untuk menyimpan koneksi database dan `\$table_name`, yang menunjuk ke tabel "users". Dalam konstruktor, koneksi database diinisialisasi. Metode `login()` menerima username dan password sebagai parameter, kemudian menjalankan query untuk mencari pengguna berdasarkan username yang diberikan. Jika pengguna ditemukan, password yang dimasukkan dibandingkan dengan password yang tersimpan di database menggunakan fungsi `password_verify()`. Jika verifikasi berhasil, metode ini mengembalikan nilai `true`, menandakan bahwa login berhasil; jika tidak, akan mengembalikan `false`. Secara keseluruhan, kode ini menyediakan mekanisme yang aman untuk autentikasi pengguna dalam aplikasi.

5. Membuat file dengan nama register.php seperti contoh berikut

```
<?php  
//register.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->register($username, $password)) {  
        $success_message = "Registrasi berhasil silahkan login";
```



```

    } else {
        $error_message = "Registrasi gagal. Username Mungkin
sudah di gunakan.";
    }
}

```

```
?>
```

```

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    <link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/css/boot
strap.min.css" rel="stylesheet" integrity="sha384-
EVSTQN3/azprG1Anm3QDgpJLIm9Nao0Yz1ztcQTWFspD3yD65VohhpUuCOmLASjC
" crossorigin="anonymous">
    <title>Register</title>
</head>
<body>
    <div class="container mt-5">
        <div class="row justify-content-center">
            <div class="col-md-4">
                <h3 class="text-center">Register</h3>
                <?php if (isset($error_message)): ?>
                    <div class="alert alert-danger">
                        <?php echo
htmlspecialchars($error_message); ?>
                    </div>
                <?php endif; ?>
                <form action="" method="POST">
                    <div class="form-group mb-3">
                        <label>Username</label>
                        <input type="text" name="username"
class="form-control" required>
                    </div>
                    <div class="form-group mb-3">
                        <label>Password</label>
                        <input type="password" name="password"
class="form-control" required>
                    </div>
                    <button type="submit" class="btn btn-primary
w-100">Daftar</button>
                    <p class="text-center mt-3">sudah punya akun?
                    <a href="index.php">login di sini</a>
                </form>
            </div>
        </div>
    </div>

```

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

PENJELASAN : Kode PHP dan HTML di atas menyediakan antarmuka untuk registrasi pengguna dalam aplikasi. Pertama, skrip ini mengimpor file konfigurasi dan kelas `User` untuk mengelola operasi terkait pengguna. Setelah mendapatkan koneksi database, objek `User` dibuat untuk menangani pendaftaran. Jika metode permintaan adalah POST, username dan password yang dimasukkan pengguna diambil dari formulir. Metode `register()` dipanggil untuk mencoba menyimpan informasi pengguna ke database; jika berhasil, pesan sukses ditampilkan, sedangkan jika gagal, pesan kesalahan menunjukkan bahwa username mungkin sudah digunakan. Halaman ini dirancang dengan Bootstrap untuk styling, menampilkan formulir pendaftaran yang responsif dengan input untuk username dan password, serta menyertakan tautan bagi pengguna yang sudah memiliki akun untuk login. Dengan demikian, kode ini memberikan pengalaman yang mudah dan intuitif untuk proses registrasi.