Dosen : Mujiyanto, M.Kom

Praktikum 8: Membuat Sistem Login dan CRUD User Sederhana

Tutorial ini akan memandu Anda langkah demi langkah untuk membuat aplikasi web sederhana menggunakan PHP native (tanpa framework) yang memiliki fitur registrasi, login, logout, dan operasi CRUD (Create, Read, Update, Delete) untuk data pengguna. Kita akan menggunakan database MySQL/MariaDB dan ekstensi PDO untuk interaksi database yang aman.

⚙️ Prasyarat

Sebelum memulai, pastikan Anda memiliki:

1. **Web Server Lokal**: XAMPP, WAMP, MAMP, atau sejenisnya yang sudah terinstal dan berjalan. Ini akan menyediakan Apache (web server), PHP, dan MySQL/MariaDB.
2. **Text Editor atau IDE**: Seperti Visual Studio Code, Sublime Text, Notepad++, atau PhpStorm.
3. **Browser Web**: Google Chrome, Firefox, atau sejenisnya.
4. **Pemahaman Dasar**:
    - HTML dan CSS (untuk tampilan).
    - PHP (sintaks dasar, variabel, array, fungsi, superglobals seperti `$_POST`, `$_GET`, `$_SESSION`).
    - SQL (dasar query SELECT, INSERT, UPDATE, DELETE).

📂 Struktur Folder Proyek

Buatlah struktur folder untuk proyek Anda seperti berikut:

```
sistem_login_crud/
├── config/
│   └── database.php         # File konfigurasi koneksi database
├── auth/
│   ├── login.php            # Halaman dan proses login
│   ├── register.php         # Halaman dan proses registrasi
│   └── logout.php           # Proses logout
├── users/
│   ├── index.php            # Halaman utama CRUD (menampilkan semua user)
│   ├── create.php           # Halaman dan proses tambah user baru
│   ├── edit.php             # Halaman dan proses edit user
│   ├── delete.php           # Proses hapus user
│   └── auth_check.php       # Script untuk memeriksa status login
├── public/
│   └── css/
│       └── style.css        # File CSS untuk styling (opsional)
```

```
└── index.php                # Halaman utama aplikasi (landing page)
```

Langkah 1: Membuat Database dan Tabel `users`

1. Buka phpMyAdmin (biasanya dapat diakses melalui `http://localhost/phpmyadmin`).
2. Buat database baru. Misalnya, beri nama `db_app_user`.
3. Setelah memilih database `db_app_user`, jalankan query SQL berikut untuk membuat tabel `users`:

```
4. CREATE TABLE users (
5.     id INT AUTO_INCREMENT PRIMARY KEY,
6.     username VARCHAR(50) NOT NULL UNIQUE,
7.     password VARCHAR(255) NOT NULL,
8.     nama_lengkap VARCHAR(100) NOT NULL,
9.     email VARCHAR(100) NOT NULL UNIQUE,
10.      role ENUM('admin', 'user') DEFAULT 'user', -- Opsional: untuk
   peran pengguna
11.      created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
12.      updated_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP ON UPDATE
   CURRENT_TIMESTAMP
13.  );
14.
```

Penjelasan Kolom:

- `id`: Primary key, akan bertambah otomatis.
- `username`: Nama pengguna untuk login, harus unik.
- `password`: Kata sandi pengguna, akan disimpan dalam bentuk hash.
- `nama_lengkap`: Nama lengkap pengguna.
- `email`: Alamat email pengguna, harus unik.
- `role`: (Opsional) Peran pengguna, bisa 'admin' atau 'user'. Defaultnya 'user'.
- `created_at`: Timestamp kapan data pengguna dibuat.
- `updated_at`: Timestamp kapan data pengguna terakhir diubah.

Langkah 2: Konfigurasi Koneksi Database (`config/database.php`)

File ini akan berisi detail koneksi ke database MySQL Anda.

```
<?php
// config/database.php

$host = "localhost";        // Atau alamat IP server database Anda
$db_name = "db_app_user";   // Nama database yang telah Anda buat
$username_db = "root";      // Username database Anda (default XAMPP adalah
"root")
$password_db = "";          // Password database Anda (default XAMPP kosong)

try {
```

```
    $conn = new PDO("mysql:host={$host};dbname={$db_name}", $username_db,
$password_db);
    // Set mode error PDO ke exception
    $conn->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
    // echo "Koneksi ke database berhasil!"; // Hapus atau beri komentar
setelah tes
} catch(PDOException $exception) {
    // Tampilkan pesan error jika koneksi gagal
    die("Koneksi error: " . $exception->getMessage());
}
?>
```

Penting:

- Sesuaikan nilai `$host`, `$db_name`, `$username_db`, dan `$password_db` dengan konfigurasi server database Anda.
- Kita menggunakan **PDO (PHP Data Objects)** karena lebih aman (membantu mencegah SQL Injection melalui prepared statements) dan mendukung berbagai jenis database.

Langkah 3: Membuat Halaman Utama (`index.php` di root)

Ini adalah halaman pertama yang akan dilihat pengguna.

```
<?php
session_start(); // Mulai session di setiap halaman yang membutuhkannya
?>
<!DOCTYPE html>
<html lang="id">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Selamat Datang</title>
    <link rel="stylesheet" href="public/css/style.css"> </head>
<body>
    <div class="container">
        <h1>Selamat Datang di Aplikasi Manajemen User</h1>
        <nav>
            <ul>
                <?php if (isset($_SESSION['user_id'])): ?>
                    <li>Halo, <?php echo
htmlspecialchars($_SESSION['nama_lengkap']); ?>!</li>
                    <li><a href="users/index.php">Manajemen User
(CRUD)</a></li>
                    <li><a href="auth/logout.php">Logout</a></li>
                <?php else: ?>
                    <li><a href="auth/login.php">Login</a></li>
                    <li><a href="auth/register.php">Registrasi</a></li>
                <?php endif; ?>
            </ul>
        </nav>
```

```
        <p>Ini adalah halaman utama. Silakan login atau registrasi untuk
melanjutkan.</p>
    </div>
</body>
</html>
```

Langkah 4: Sistem Registrasi Pengguna (`auth/register.php`)

Halaman ini memungkinkan pengguna baru untuk mendaftar.

```
<?php
session_start();
require_once '../config/database.php'; // Hubungkan ke database

$errors = [];
$username = $nama_lengkap = $email = ""; // Inisialisasi variabel

if ($_SERVER["REQUEST_METHOD"] == "POST") {
    $username = trim($_POST['username']);
    $password = $_POST['password'];
    $konfirmasi_password = $_POST['konfirmasi_password'];
    $nama_lengkap = trim($_POST['nama_lengkap']);
    $email = trim($_POST['email']);
    $role = $_POST['role'] ?? 'user'; // Default ke 'user' jika tidak ada

    // --- Validasi Input Sederhana ---
    if (empty($username)) { $errors[] = "Username wajib diisi."; }
    if (strlen($username) < 4) { $errors[] = "Username minimal 4 karakter.";
}
    if (empty($password)) { $errors[] = "Password wajib diisi."; }
    if (strlen($password) < 6) { $errors[] = "Password minimal 6 karakter.";
}
    if ($password !== $konfirmasi_password) { $errors[] = "Konfirmasi
password tidak cocok."; }
    if (empty($nama_lengkap)) { $errors[] = "Nama lengkap wajib diisi."; }
    if (empty($email) || !filter_var($email, FILTER_VALIDATE_EMAIL)) {
        $errors[] = "Email tidak valid atau wajib diisi.";
    }
    if (!in_array($role, ['admin', 'user'])) { $errors[] = "Role tidak
valid."; }


    // --- Cek apakah username atau email sudah ada ---
    if (empty($errors)) {
        try {
            $stmt = $conn->prepare("SELECT id FROM users WHERE username =
:username OR email = :email LIMIT 1");
            $stmt->bindParam(':username', $username);
            $stmt->bindParam(':email', $email);
            $stmt->execute();
            if ($stmt->rowCount() > 0) {
                $errors[] = "Username atau Email sudah terdaftar. Silakan
gunakan yang lain.";
```

```
            }
        } catch (PDOException $e) {
            $errors[] = "Error saat memeriksa data: " . $e->getMessage();
        }
    }

    // --- Jika tidak ada error, simpan ke database ---
    if (empty($errors)) {
        $hashed_password = password_hash($password, PASSWORD_BCRYPT); // Hash
password

        try {
            $stmt = $conn->prepare("INSERT INTO users (username, password,
nama_lengkap, email, role) VALUES (:username, :password, :nama_lengkap,
:email, :role)");
            $stmt->bindParam(':username', $username);
            $stmt->bindParam(':password', $hashed_password);
            $stmt->bindParam(':nama_lengkap', $nama_lengkap);
            $stmt->bindParam(':email', $email);
            $stmt->bindParam(':role', $role);

            if ($stmt->execute()) {
                $_SESSION['success_message'] = "Registrasi berhasil! Silakan
login.";
                header("Location: login.php"); // Arahkan ke halaman login
                exit();
            } else {
                $errors[] = "Registrasi gagal. Silakan coba lagi.";
            }
        } catch (PDOException $e) {
            $errors[] = "Error database: " . $e->getMessage();
        }
    }
}
?>
<!DOCTYPE html>
<html lang="id">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Registrasi User</title>
    <link rel="stylesheet" href="../public/css/style.css">
</head>
<body>
    <div class="container">
        <h2>Registrasi User Baru</h2>

        <?php if (!empty($errors)): ?>
            <div class="errors">
                <?php foreach ($errors as $error): ?>
                    <p><?php echo htmlspecialchars($error); ?></p>
                <?php endforeach; ?>
            </div>
        <?php endif; ?>

        <form action="register.php" method="post">
            <div>
```

```
                <label for="username">Username:</label>
                <input type="text" id="username" name="username" value="<?php
echo htmlspecialchars($username); ?>" required>
            </div>
            <div>
                <label for="password">Password:</label>
                <input type="password" id="password" name="password"
required>
            </div>
            <div>
                <label for="konfirmasi_password">Konfirmasi Password:</label>
                <input type="password" id="konfirmasi_password"
name="konfirmasi_password" required>
            </div>
            <div>
                <label for="nama_lengkap">Nama Lengkap:</label>
                <input type="text" id="nama_lengkap" name="nama_lengkap"
value="<?php echo htmlspecialchars($nama_lengkap); ?>" required>
            </div>
            <div>
                <label for="email">Email:</label>
                <input type="email" id="email" name="email" value="<?php echo
htmlspecialchars($email); ?>" required>
            </div>
            <div>
                <label for="role">Role:</label>
                <select id="role" name="role">
                    <option value="user" selected>User</option>
                    <option value="admin">Admin</option>
                </select>
            </div>
            <div>
                <button type="submit">Daftar</button>
            </div>
            <p>Sudah punya akun? <a href="login.php">Login di sini</a></p>
        </form>
    </div>
</body>
</html>
```

Penjelasan `register.php`:

- **`session_start()`**: Memulai session.
- **Validasi**: Melakukan validasi dasar pada input.
- **Cek Duplikasi**: Memastikan username dan email belum terdaftar.
- **`password_hash()`**: Menggunakan fungsi ini untuk mengenkripsi password sebelum disimpan. Ini **sangat penting** untuk keamanan. `PASSWORD_BCRYPT` adalah algoritma hashing yang kuat.
- **`htmlspecialchars()`**: Digunakan saat menampilkan kembali input pengguna (misalnya jika ada error dan form diisi ulang) untuk mencegah serangan XSS.

Langkah 5: Sistem Login Pengguna (`auth/login.php`)

Halaman ini digunakan pengguna untuk masuk ke sistem.

```
<?php
session_start();
require_once '../config/database.php';

// Jika sudah login, arahkan ke halaman dashboard user
if (isset($_SESSION['user_id'])) {
    header("Location: ../users/index.php");
    exit();
}

$errors = [];
$username = "";

// Tampilkan pesan sukses dari registrasi jika ada
if (isset($_SESSION['success_message'])) {
    $success_message = $_SESSION['success_message'];
    unset($_SESSION['success_message']); // Hapus pesan setelah ditampilkan
}

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

    if (empty($username)) { $errors[] = "Username wajib diisi."; }
    if (empty($password)) { $errors[] = "Password wajib diisi."; }

    if (empty($errors)) {
        try {
            $stmt = $conn->prepare("SELECT id, username, password,
nama_lengkap, role FROM users WHERE username = :username LIMIT 1");
            $stmt->bindParam(':username', $username);
            $stmt->execute();

            if ($stmt->rowCount() == 1) {
                $user = $stmt->fetch(PDO::FETCH_ASSOC);
                // Verifikasi password
                if (password_verify($password, $user['password'])) {
                    // Password cocok, simpan informasi user ke session
                    $_SESSION['user_id'] = $user['id'];
                    $_SESSION['username'] = $user['username'];
                    $_SESSION['nama_lengkap'] = $user['nama_lengkap'];
                    $_SESSION['role'] = $user['role']; // Simpan role

                    header("Location: ../users/index.php"); // Arahkan ke
halaman dashboard user
                    exit();
                } else {
                    $errors[] = "Username atau password salah.";
                }
            } else {
                $errors[] = "Username atau password salah.";
```

```
            }
        } catch (PDOException $e) {
            $errors[] = "Error database: " . $e->getMessage();
        }
    }
}
?>
<!DOCTYPE html>
<html lang="id">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Login User</title>
    <link rel="stylesheet" href="../public/css/style.css">
</head>
<body>
    <div class="container">
        <h2>Login User</h2>

        <?php if (isset($success_message)): ?>
            <div class="success">
                <p><?php echo htmlspecialchars($success_message); ?></p>
            </div>
        <?php endif; ?>

        <?php if (!empty($errors)): ?>
            <div class="errors">
                <?php foreach ($errors as $error): ?>
                    <p><?php echo htmlspecialchars($error); ?></p>
                <?php endforeach; ?>
            </div>
        <?php endif; ?>

        <form action="login.php" method="post">
            <div>
                <label for="username">Username:</label>
                <input type="text" id="username" name="username" value="<?php
echo htmlspecialchars($username); ?>" required>
            </div>
            <div>
                <label for="password">Password:</label>
                <input type="password" id="password" name="password"
required>
            </div>
            <div>
                <button type="submit">Login</button>
            </div>
            <p>Belum punya akun? <a href="register.php">Daftar di
sini</a></p>
        </form>
    </div>
</body>
</html>
```

Penjelasan `login.php`:

- **Cek Session**: Jika `$_SESSION['user_id']` sudah ada, berarti pengguna sudah login, jadi langsung arahkan.
- **`password_verify()`**: Fungsi ini digunakan untuk membandingkan password yang diinput pengguna dengan hash password yang tersimpan di database.
- **Menyimpan Session**: Jika login berhasil, informasi penting pengguna (seperti `id`, `username`, `nama_lengkap`, `role`) disimpan ke dalam `$_SESSION`.

Langkah 6: Sistem Logout Pengguna (`auth/logout.php`)

Skrip ini akan menghapus semua data session dan mengarahkan pengguna ke halaman login.

```
<?php
// auth/logout.php
session_start();

// Hapus semua variabel session
$_SESSION = array();

// Jika ingin menghancurkan session, juga hapus cookie session.
// Catatan: Ini akan menghancurkan session, dan bukan hanya data session!
if (ini_get("session.use_cookies")) {
    $params = session_get_cookie_params();
    setcookie(session_name(), '', time() - 42000,
        $params["path"], $params["domain"],
        $params["secure"], $params["httponly"]
    );
}

// Akhirnya, hancurkan session.
session_destroy();

// Arahkan ke halaman login atau halaman utama
header("Location: login.php");
exit();
?>
```

Langkah 7: Pemeriksaan Otentikasi (`users/auth_check.php`)

Skrip ini akan disertakan di awal setiap halaman dalam folder `users/` untuk memastikan hanya pengguna yang sudah login yang dapat mengaksesnya.

```
<?php
// users/auth_check.php
if (session_status() == PHP_SESSION_NONE) { // Mulai session jika belum aktif
    session_start();
}
```

```
// Jika user belum login (tidak ada user_id di session), arahkan ke halaman
login
if (!isset($_SESSION['user_id'])) {
    $_SESSION['error_message'] = "Anda harus login untuk mengakses halaman
ini.";
    header("Location: ../auth/login.php"); // Sesuaikan path jika file ini
dipindah
    exit();
}

// Opsional: Pemeriksaan role jika diperlukan
// function isAdmin() {
//     return isset($_SESSION['role']) && $_SESSION['role'] === 'admin';
// }

// function requireAdmin() {
//     if (!isAdmin()) {
//         $_SESSION['error_message'] = "Anda tidak memiliki hak akses
admin.";
//         header("Location: index.php"); // Arahkan ke halaman user biasa
atau dashboard
//         exit();
//     }
// }
?>
```

Anda bisa mengaktifkan dan menggunakan fungsi `isAdmin()` dan `requireAdmin()` jika ingin membatasi akses halaman tertentu hanya untuk admin.

Langkah 8: CRUD - Read/Menampilkan Data User (`users/index.php`)

Halaman ini akan menampilkan daftar semua pengguna yang terdaftar dan menyediakan link untuk operasi CRUD lainnya.

```
<?php
require_once 'auth_check.php'; // Wajib login
require_once '../config/database.php';

// Ambil semua data user, diurutkan berdasarkan tanggal dibuat
try {
    $stmt = $conn->prepare("SELECT id, username, nama_lengkap, email, role,
created_at FROM users ORDER BY created_at DESC");
    $stmt->execute();
    $users = $stmt->fetchAll(PDO::FETCH_ASSOC);
} catch (PDOException $e) {
    // Idealnya, log error ini daripada menampilkannya langsung
    $page_error = "Error mengambil data user: " . $e->getMessage();
    $users = []; // Kosongkan jika ada error
}
?>
```

```
<!DOCTYPE html>
<html lang="id">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Manajemen User</title>
    <link rel="stylesheet" href="../public/css/style.css">
</head>
<body>
    <div class="container">
        <div class="header-nav">
            <h2>Manajemen User</h2>
            <div>
                <span>Halo, <?php echo
htmlspecialchars($_SESSION['nama_lengkap']); ?> (<?php echo
htmlspecialchars($_SESSION['role']); ?>) | </span>
                <a href="../index.php">Halaman Utama</a> |
                <a href="../auth/logout.php">Logout</a>
            </div>
        </div>

        <?php if (isset($_SESSION['message'])): ?>
            <div class="message <?php echo isset($_SESSION['message_type']) ?
$_SESSION['message_type'] : 'success'; ?>">
                <p><?php echo htmlspecialchars($_SESSION['message']); ?></p>
            </div>
            <?php
                unset($_SESSION['message']);
                unset($_SESSION['message_type']);
            ?>
        <?php endif; ?>

        <?php if (isset($page_error)): ?>
            <div class="errors"><p><?php echo htmlspecialchars($page_error);
?></p></div>
        <?php endif; ?>

        <p><a href="create.php" class="btn">Tambah User Baru</a></p>

        <?php if (count($users) > 0): ?>
            <table>
                <thead>
                    <tr>
                        <th>ID</th>
                        <th>Username</th>
                        <th>Nama Lengkap</th>
                        <th>Email</th>
                        <th>Role</th>
                        <th>Tanggal Daftar</th>
                        <th>Aksi</th>
                    </tr>
                </thead>
                <tbody>
                    <?php foreach ($users as $user): ?>
                        <tr>
                            <td><?php echo htmlspecialchars($user['id']);
?></td>
```

```
                            <td><?php echo
htmlspecialchars($user['username']); ?></td>
                            <td><?php echo
htmlspecialchars($user['nama_lengkap']); ?></td>
                            <td><?php echo htmlspecialchars($user['email']);
?></td>
                            <td><?php echo
htmlspecialchars(ucfirst($user['role'])); ?></td>
                            <td><?php echo htmlspecialchars(date('d M Y,
H:i', strtotime($user['created_at']))); ?></td>
                            <td>
                                <a href="edit.php?id=<?php echo $user['id'];
?>" class="btn-edit">Edit</a>
                                <?php if ($_SESSION['user_id'] !=
$user['id']): // Jangan biarkan user hapus diri sendiri dari sini ?>
                                <a href="delete.php?id=<?php echo
$user['id']; ?>" class="btn-delete" onclick="return confirm('Apakah Anda
yakin ingin menghapus user ini?');">Hapus</a>
                                <?php endif; ?>
                            </td>
                        </tr>
                    <?php endforeach; ?>
                </tbody>
            </table>
        <?php else: ?>
            <p>Belum ada user terdaftar.</p>
        <?php endif; ?>
    </div>
</body>
</html>
```

Langkah 9: CRUD - Create/Tambah User Baru (`users/create.php`)

Halaman formulir untuk menambahkan pengguna baru oleh admin atau pengguna yang memiliki hak akses.

```
<?php
require_once 'auth_check.php';
// requireAdmin(); // Aktifkan jika hanya admin yang boleh menambah user
require_once '../config/database.php';

$errors = [];
$username = $nama_lengkap = $email = $role = ""; // Inisialisasi

if ($_SERVER["REQUEST_METHOD"] == "POST") {
    $username = trim($_POST['username']);
    $password = $_POST['password'];
    $konfirmasi_password = $_POST['konfirmasi_password'];
    $nama_lengkap = trim($_POST['nama_lengkap']);
    $email = trim($_POST['email']);
    $role = $_POST['role'] ?? 'user';
```

```
    // Validasi serupa dengan register.php
    if (empty($username)) { $errors[] = "Username wajib diisi."; }
    if (empty($password)) { $errors[] = "Password wajib diisi."; }
    if ($password !== $konfirmasi_password) { $errors[] = "Konfirmasi
password tidak cocok."; }
    if (empty($nama_lengkap)) { $errors[] = "Nama lengkap wajib diisi."; }
    if (empty($email) || !filter_var($email, FILTER_VALIDATE_EMAIL)) {
$errors[] = "Email tidak valid."; }
    if (!in_array($role, ['admin', 'user'])) { $errors[] = "Role tidak
valid."; }

    // Cek duplikasi username/email
    if (empty($errors)) {
        try {
            $stmt_check = $conn->prepare("SELECT id FROM users WHERE username
= :username OR email = :email LIMIT 1");
            $stmt_check->bindParam(':username', $username);
            $stmt_check->bindParam(':email', $email);
            $stmt_check->execute();
            if ($stmt_check->rowCount() > 0) {
                $errors[] = "Username atau Email sudah terdaftar.";
            }
        } catch (PDOException $e) {
            $errors[] = "Error saat memeriksa data: " . $e->getMessage();
        }
    }

    if (empty($errors)) {
        $hashed_password = password_hash($password, PASSWORD_BCRYPT);
        try {
            $stmt = $conn->prepare("INSERT INTO users (username, password,
nama_lengkap, email, role) VALUES (:username, :password, :nama_lengkap,
:email, :role)");
            $stmt->bindParam(':username', $username);
            $stmt->bindParam(':password', $hashed_password);
            $stmt->bindParam(':nama_lengkap', $nama_lengkap);
            $stmt->bindParam(':email', $email);
            $stmt->bindParam(':role', $role);

            if ($stmt->execute()) {
                $_SESSION['message'] = "User baru berhasil ditambahkan!";
                $_SESSION['message_type'] = "success";
                header("Location: index.php");
                exit();
            } else {
                $errors[] = "Gagal menambahkan user.";
            }
        } catch (PDOException $e) {
            $errors[] = "Error database: " . $e->getMessage();
        }
    }
}
?>
<!DOCTYPE html>
<html lang="id">
<head>
```

```
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Tambah User Baru</title>
    <link rel="stylesheet" href="../public/css/style.css">
</head>
<body>
    <div class="container">
        <div class="header-nav">
            <h2>Tambah User Baru</h2>
            <div>
                <a href="index.php">Kembali ke Daftar User</a> |
                <a href="../auth/logout.php">Logout</a>
            </div>
        </div>

        <?php if (!empty($errors)): ?>
            <div class="errors">
                <?php foreach ($errors as $error): ?>
                    <p><?php echo htmlspecialchars($error); ?></p>
                <?php endforeach; ?>
            </div>
        <?php endif; ?>

        <form action="create.php" method="post">
            <div>
                <label for="username">Username:</label>
                <input type="text" id="username" name="username" value="<?php
echo htmlspecialchars($username); ?>" required>
            </div>
            <div>
                <label for="password">Password:</label>
                <input type="password" id="password" name="password"
required>
            </div>
             <div>
                <label for="konfirmasi_password">Konfirmasi Password:</label>
                <input type="password" id="konfirmasi_password"
name="konfirmasi_password" required>
            </div>
            <div>
                <label for="nama_lengkap">Nama Lengkap:</label>
                <input type="text" id="nama_lengkap" name="nama_lengkap"
value="<?php echo htmlspecialchars($nama_lengkap); ?>" required>
            </div>
            <div>
                <label for="email">Email:</label>
                <input type="email" id="email" name="email" value="<?php echo
htmlspecialchars($email); ?>" required>
            </div>
            <div>
                <label for="role">Role:</label>
                <select id="role" name="role">
                    <option value="user" <?php echo ($role === 'user' ?
'selected' : ''); ?>>User</option>
                    <option value="admin" <?php echo ($role === 'admin' ?
'selected' : ''); ?>>Admin</option>
                </select>
```

```
            </div>
            <div>
                <button type="submit">Tambah User</button>
            </div>
        </form>
    </div>
</body>
</html>
```

Langkah 10: CRUD - Update/Edit User (`users/edit.php`)

Halaman formulir untuk mengubah data pengguna yang sudah ada.

```
<?php
require_once 'auth_check.php';
// requireAdmin(); // Aktifkan jika hanya admin yang boleh mengedit user
require_once '../config/database.php';

$errors = [];
$user_id = $_GET['id'] ?? null;

if (!$user_id || !filter_var($user_id, FILTER_VALIDATE_INT)) {
    $_SESSION['message'] = "ID User tidak valid.";
    $_SESSION['message_type'] = "error";
    header("Location: index.php");
    exit();
}

// Ambil data user yang akan diedit
try {
    $stmt = $conn->prepare("SELECT id, username, nama_lengkap, email, role
FROM users WHERE id = :id");
    $stmt->bindParam(':id', $user_id, PDO::PARAM_INT);
    $stmt->execute();
    $user = $stmt->fetch(PDO::FETCH_ASSOC);

    if (!$user) {
        $_SESSION['message'] = "User tidak ditemukan.";
        $_SESSION['message_type'] = "error";
        header("Location: index.php");
        exit();
    }
} catch (PDOException $e) {
    $_SESSION['message'] = "Error mengambil data user: " . $e->getMessage();
    $_SESSION['message_type'] = "error";
    header("Location: index.php");
    exit();
}

// Inisialisasi variabel form dengan data yang ada
$username_form = $user['username'];
$nama_lengkap_form = $user['nama_lengkap'];
$email_form = $user['email'];
```

```
$role_form = $user['role'];

if ($_SERVER["REQUEST_METHOD"] == "POST") {
    $username_form = trim($_POST['username']);
    $password_new = $_POST['password']; // Password baru, opsional
    $konfirmasi_password_new = $_POST['konfirmasi_password'];
    $nama_lengkap_form = trim($_POST['nama_lengkap']);
    $email_form = trim($_POST['email']);
    $role_form = $_POST['role'] ?? 'user';

    // Validasi
    if (empty($username_form)) { $errors[] = "Username wajib diisi."; }
    if (empty($nama_lengkap_form)) { $errors[] = "Nama lengkap wajib diisi.";
}
    if (empty($email_form) || !filter_var($email_form,
FILTER_VALIDATE_EMAIL)) { $errors[] = "Email tidak valid."; }
    if (!empty($password_new) && $password_new !== $konfirmasi_password_new)
{
        $errors[] = "Konfirmasi password baru tidak cocok.";
    }
     if (!empty($password_new) && strlen($password_new) < 6) { $errors[] =
"Password baru minimal 6 karakter."; }
    if (!in_array($role_form, ['admin', 'user'])) { $errors[] = "Role tidak
valid."; }


    // Cek duplikasi username/email (kecuali untuk user ini sendiri)
    if (empty($errors)) {
        try {
            $stmt_check = $conn->prepare("SELECT id FROM users WHERE
(username = :username OR email = :email) AND id != :id_current LIMIT 1");
            $stmt_check->bindParam(':username', $username_form);
            $stmt_check->bindParam(':email', $email_form);
            $stmt_check->bindParam(':id_current', $user_id, PDO::PARAM_INT);
            $stmt_check->execute();
            if ($stmt_check->rowCount() > 0) {
                $errors[] = "Username atau Email baru sudah digunakan oleh
user lain.";
            }
        } catch (PDOException $e) {
            $errors[] = "Error saat memeriksa duplikasi: " . $e-
>getMessage();
        }
    }

    if (empty($errors)) {
        try {
            // Bangun query update
            $sql_update = "UPDATE users SET username = :username,
nama_lengkap = :nama_lengkap, email = :email, role = :role";
            $params_update = [
                ':username' => $username_form,
                ':nama_lengkap' => $nama_lengkap_form,
                ':email' => $email_form,
                ':role' => $role_form,
                ':id' => $user_id
            ];
```

```
            // Jika password baru diisi, update juga passwordnya
            if (!empty($password_new)) {
                $hashed_password_new = password_hash($password_new,
PASSWORD_BCRYPT);
                $sql_update .= ", password = :password";
                $params_update[':password'] = $hashed_password_new;
            }

            $sql_update .= " WHERE id = :id";
            $stmt_update = $conn->prepare($sql_update);

            if ($stmt_update->execute($params_update)) {
                $_SESSION['message'] = "Data user berhasil diperbarui!";
                $_SESSION['message_type'] = "success";
                // Jika user yang sedang login mengedit profilnya sendiri,
update session
                if ($_SESSION['user_id'] == $user_id) {
                    $_SESSION['username'] = $username_form;
                    $_SESSION['nama_lengkap'] = $nama_lengkap_form;
                    $_SESSION['role'] = $role_form;
                }
                header("Location: index.php");
                exit();
            } else {
                $errors[] = "Gagal memperbarui data user.";
            }
        } catch (PDOException $e) {
            $errors[] = "Error database: " . $e->getMessage();
        }
    }
}
?>
<!DOCTYPE html>
<html lang="id">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Edit User: <?php echo htmlspecialchars($user['username']);
?></title>
    <link rel="stylesheet" href="../public/css/style.css">
</head>
<body>
    <div class="container">
        <div class="header-nav">
            <h2>Edit User: <?php echo htmlspecialchars($user['username']);
?></h2>
             <div>
                <a href="index.php">Kembali ke Daftar User</a> |
                <a href="../auth/logout.php">Logout</a>
            </div>
        </div>

        <?php if (!empty($errors)): ?>
            <div class="errors">
                <?php foreach ($errors as $error): ?>
                    <p><?php echo htmlspecialchars($error); ?></p>
```

```
                <?php endforeach; ?>
            </div>
        <?php endif; ?>

        <form action="edit.php?id=<?php echo $user_id; ?>" method="post">
            <div>
                <label for="username">Username:</label>
                <input type="text" id="username" name="username" value="<?php
echo htmlspecialchars($username_form); ?>" required>
            </div>
            <div>
                <label for="password">Password Baru (Opsional):</label>
                <input type="password" id="password" name="password">
                <small>Kosongkan jika tidak ingin mengubah password.</small>
            </div>
            <div>
                <label for="konfirmasi_password">Konfirmasi Password Baru
(Opsional):</label>
                <input type="password" id="konfirmasi_password"
name="konfirmasi_password">
            </div>
            <div>
                <label for="nama_lengkap">Nama Lengkap:</label>
                <input type="text" id="nama_lengkap" name="nama_lengkap"
value="<?php echo htmlspecialchars($nama_lengkap_form); ?>" required>
            </div>
            <div>
                <label for="email">Email:</label>
                <input type="email" id="email" name="email" value="<?php echo
htmlspecialchars($email_form); ?>" required>
            </div>
            <div>
                <label for="role">Role:</label>
                <select id="role" name="role">
                    <option value="user" <?php echo ($role_form === 'user' ?
'selected' : ''); ?>>User</option>
                    <option value="admin" <?php echo ($role_form === 'admin'
? 'selected' : ''); ?>>Admin</option>
                </select>
            </div>
            <div>
                <button type="submit">Simpan Perubahan</button>
            </div>
        </form>
    </div>
</body>
</html>
```

Langkah 11: CRUD - Delete/Hapus User (`users/delete.php`)

Skrip ini akan memproses penghapusan data pengguna. Ini adalah skrip backend, jadi tidak ada output HTML langsung.

```php
<?php
require_once 'auth_check.php';
// requireAdmin(); // Aktifkan jika hanya admin yang boleh menghapus
require_once '../config/database.php';

$user_id_to_delete = $_GET['id'] ?? null;

if (!$user_id_to_delete || !filter_var($user_id_to_delete,
FILTER_VALIDATE_INT)) {
    $_SESSION['message'] = "ID User tidak valid untuk dihapus.";
    $_SESSION['message_type'] = "error";
    header("Location: index.php");
    exit();
}

// Sangat penting: Jangan biarkan user menghapus dirinya sendiri!
if (isset($_SESSION['user_id']) && $_SESSION['user_id'] ==
$user_id_to_delete) {
    $_SESSION['message'] = "Anda tidak dapat menghapus akun Anda sendiri.";
    $_SESSION['message_type'] = "warning"; // atau "error"
    header("Location: index.php");
    exit();
}

// Opsional: Jika ada aturan lain, misalnya admin terakhir tidak boleh
dihapus.
// try {
//     $stmt_count_admin = $conn->prepare("SELECT COUNT(*) as admin_count
FROM users WHERE role = 'admin'");
//     $stmt_count_admin->execute();
//     $admin_info = $stmt_count_admin->fetch(PDO::FETCH_ASSOC);

//     $stmt_user_to_delete_role = $conn->prepare("SELECT role FROM users
WHERE id = :id");
//     $stmt_user_to_delete_role->bindParam(':id', $user_id_to_delete,
PDO::PARAM_INT);
//     $stmt_user_to_delete_role->execute();
//     $user_to_delete_info = $stmt_user_to_delete_role-
>fetch(PDO::FETCH_ASSOC);

//     if ($user_to_delete_info && $user_to_delete_info['role'] === 'admin'
&& $admin_info['admin_count'] <= 1) {
//         $_SESSION['message'] = "Tidak dapat menghapus admin terakhir.";
//         $_SESSION['message_type'] = "error";
//         header("Location: index.php");
//         exit();
//     }
// } catch (PDOException $e) {
//     $_SESSION['message'] = "Error saat memeriksa status admin: " . $e-
>getMessage();
//     $_SESSION['message_type'] = "error";
//     header("Location: index.php");
//     exit();
// }


try {
```

```
    $stmt = $conn->prepare("DELETE FROM users WHERE id = :id");
    $stmt->bindParam(':id', $user_id_to_delete, PDO::PARAM_INT);

    if ($stmt->execute()) {
        if ($stmt->rowCount() > 0) {
            $_SESSION['message'] = "User berhasil dihapus!";
            $_SESSION['message_type'] = "success";
        } else {
            $_SESSION['message'] = "User tidak ditemukan atau sudah
dihapus.";
            $_SESSION['message_type'] = "warning";
        }
    } else {
        $_SESSION['message'] = "Gagal menghapus user.";
        $_SESSION['message_type'] = "error";
    }
} catch (PDOException $e) {
    $_SESSION['message'] = "Error database: " . $e->getMessage();
    $_SESSION['message_type'] = "error";
}

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

Langkah 12: Styling (Opsional - `public/css/style.css`)

Buat file `style.css` di dalam folder `public/css/` untuk memberikan tampilan dasar pada aplikasi Anda.

```
/* public/css/style.css */
body {
    font-family: 'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;
    line-height: 1.6;
    margin: 0;
    padding: 0;
    background-color: #f9f9f9;
    color: #333;
}

.container {
    width: 85%;
    max-width: 960px;
    margin: 30px auto;
    overflow: hidden;
    padding: 25px;
    background-color: #fff;
    box-shadow: 0 0 15px rgba(0,0,0,0.1);
    border-radius: 8px;
}

h1, h2 {
    color: #333;
```

```
    text-align: center;
    margin-bottom: 25px;
}

.header-nav {
    display: flex;
    justify-content: space-between;
    align-items: center;
    margin-bottom: 20px;
    padding-bottom: 15px;
    border-bottom: 1px solid #eee;
}

.header-nav h2 {
    margin: 0;
    text-align: left;
    font-size: 1.8em;
}

.header-nav div {
    font-size: 0.95em;
}
.header-nav div span {
    font-weight: bold;
}
.header-nav div a {
    margin-left: 10px;
}


nav ul {
    padding: 0;
    list-style: none;
    text-align: center;
    margin-bottom: 20px;
}

nav ul li {
    display: inline;
    margin-right: 15px;
}

nav ul li a {
    color: #007bff;
    text-decoration: none;
    font-weight: bold;
}

nav ul li a:hover {
    text-decoration: underline;
}

form div {
    margin-bottom: 18px;
}

form label {
```

```
    display: block;
    margin-bottom: 6px;
    font-weight: bold;
    color: #555;
}

form input[type="text"],
form input[type="password"],
form input[type="email"],
form select {
    width: 100%;
    padding: 12px;
    border: 1px solid #ddd;
    border-radius: 5px;
    box-sizing: border-box;
    font-size: 1em;
}

form input:focus, form select:focus {
    border-color: #007bff;
    outline: none;
    box-shadow: 0 0 5px rgba(0,123,255,0.25);
}

form small {
    font-size: 0.85em;
    color: #777;
    display: block;
    margin-top: 5px;
}

button, .btn, .btn-edit, .btn-delete {
    display: inline-block;
    background-color: #007bff;
    color: white;
    padding: 10px 18px;
    border: none;
    border-radius: 5px;
    cursor: pointer;
    text-decoration: none;
    font-size: 1em;
    transition: background-color 0.2s ease-in-out;
}

button:hover, .btn:hover {
    background-color: #0056b3;
}

.btn-edit {
    background-color: #ffc107; /* Kuning */
    color: #212529;
}
.btn-edit:hover {
    background-color: #e0a800;
}

.btn-delete {
```

```
    background-color: #dc3545; /* Merah */
}
.btn-delete:hover {
    background-color: #c82333;
}


.errors, .success, .message, .warning {
    padding: 12px 18px;
    margin-bottom: 20px;
    border: 1px solid transparent;
    border-radius: 5px;
    font-size: 0.95em;
}

.errors p, .success p, .message p, .warning p {
    margin: 0;
}

.errors {
    background-color: #f8d7da;
    color: #721c24;
    border-color: #f5c6cb;
}

.success, .message.success { /* .message.success untuk kompatibilitas */
    background-color: #d4edda;
    color: #155724;
    border-color: #c3e6cb;
}

.message.error { /* Jika menggunakan class 'message error' */
    background-color: #f8d7da;
    color: #721c24;
    border-color: #f5c6cb;
}
.warning, .message.warning {
    background-color: #fff3cd;
    color: #856404;
    border-color: #ffeeba;
}


table {
    width: 100%;
    border-collapse: collapse;
    margin-top: 25px;
    box-shadow: 0 0 10px rgba(0,0,0,0.05);
}

table th, table td {
    border: 1px solid #e0e0e0;
    padding: 12px;
    text-align: left;
    vertical-align: middle;
}
```

```
table th {
    background-color: #f2f2f2;
    font-weight: bold;
    color: #444;
}

table tr:nth-child(even) {
    background-color: #f9f9f9;
}
table tr:hover {
    background-color: #f1f1f1;
}

table td a {
    margin-right: 8px;
    padding: 6px 10px; /* Padding untuk tombol di tabel */
}
table td a:last-child {
    margin-right: 0;
}
```

Langkah 13: Cara Menjalankan Aplikasi

1. Letakkan seluruh folder `sistem_login_crud` Anda ke dalam direktori `htdocs` (jika menggunakan XAMPP/WAMP) atau `www` (jika menggunakan MAMP) di instalasi web server Anda.
2. Pastikan service Apache dan MySQL di XAMPP/WAMP/MAMP sudah berjalan.
3. Buka browser web Anda.
4. Akses aplikasi melalui URL: `http://localhost/sistem_login_crud/`
5. Anda akan diarahkan ke halaman `index.php` (halaman utama). Dari sana, Anda bisa mencoba fitur registrasi, login, dan setelah login, Anda akan bisa mengakses manajemen user (CRUD).

Poin Penting Keamanan dan Praktik Terbaik

- **SQL Injection**: Selalu gunakan *prepared statements* dengan PDO (menggunakan `bindParam()` atau melewatkan array parameter ke `execute()`). Ini adalah cara paling efektif untuk mencegah SQL Injection.
- **Password Hashing**:
    - Gunakan `password_hash()` untuk menyimpan password.
    - Gunakan `password_verify()` untuk memverifikasi password saat login.
    - Jangan pernah menyimpan password dalam bentuk teks biasa (plain text) atau menggunakan metode hashing yang sudah usang seperti MD5 atau SHA1.
- **XSS (Cross-Site Scripting)**: Selalu gunakan `htmlspecialchars()` saat menampilkan data apapun yang berasal dari input pengguna atau database ke halaman HTML. Ini akan mengkonversi karakter khusus HTML menjadi entitas HTML, sehingga mencegah eksekusi skrip berbahaya.

- **Session Security**:
    - Mulai session dengan `session_start()` di awal setiap skrip yang membutuhkan akses ke variabel session.
    - Hancurkan session secara menyeluruh saat logout (`session_destroy()` dan membersihkan cookie session).
    - Pertimbangkan untuk meregenerasi ID session secara periodik, terutama setelah login atau perubahan level hak akses (`session_regenerate_id(true)`), untuk membantu mencegah serangan *session fixation*.
- **Error Reporting**:
    - Selama pengembangan, aktifkan pelaporan error PHP untuk melihat semua jenis error:
    - `error_reporting(E_ALL);`
    - `ini_set('display_errors', 1);`
    -
    - Di server produksi, **matikan `display_errors`** (`ini_set('display_errors', 0);`) dan konfigurasikan PHP untuk mencatat error ke file log (`log_errors = On`, `error_log = /path/to/your/php-error.log`). Ini mencegah terungkapnya informasi sensitif melalui pesan error.
- **Validasi Input**: Lakukan validasi input baik di sisi klien (menggunakan atribut HTML5 seperti `required`, `type="email"`, `pattern`, atau JavaScript) maupun **(yang paling penting)** di sisi server (PHP). Validasi sisi server adalah keharusan karena validasi sisi klien dapat dengan mudah dilewati.
- **CSRF (Cross-Site Request Forgery)**: Untuk aplikasi yang lebih serius, implementasikan token CSRF untuk melindungi semua form yang melakukan perubahan data (POST, PUT, DELETE requests). Tutorial ini tidak mencakup CSRF untuk menjaga kesederhanaan, tetapi ini adalah aspek keamanan penting.
- **HTTPS**: Di lingkungan produksi, selalu gunakan HTTPS (SSL/TLS) untuk mengenkripsi semua komunikasi antara browser klien dan server web Anda.
- **Prinsip Hak Akses Terkecil (Principle of Least Privilege)**: Berikan pengguna hak akses hanya sebatas yang mereka butuhkan. Misalnya, tidak semua pengguna perlu akses ke fitur admin.