

**LAPORAN HASIL AKHIR PRAKTIKUM
PEMROGRAMAN WEB I**



NAMA : MICHAEL ISACHAR

NIM : 193030503043

KELAS : A

MODUL : II (Form Handling)

**JURUSAN TEKNIK INFORMATIKA
FAKULTAS TEKNIK
UNIVERSITAS PALANGKA RAYA**

2021

BAB I

TUJUAN DAN LANDASAN TEORI

1.1 Tujuan

- Mahasiswa mampu membuat handling yang mampu mengolah data dari form HTML.
- Mahasiswa mampu membuat batasan-batasan untuk menangani inputan dari form HTML.

1.2 Landasan Teori

1.2.1 Form Handling

PHP Form Handling merupakan hal penting dalam sebuah website, form handling digunakan untuk mengambil data dari inputan user melalui form HTML.

Contoh

```
<html>
<body>

<form action="tampil.php" method="post">
    Masukan Nama : <input type="text" name="nama" />
    <input type="submit" value="Submit" />
</form>

</body>
```

```
</html>
```

Kemudian untuk menampilkan datanya buat satu file lagi dengan nama tampil.php

```
<html>
```

```
<body>
```

```
Selamat datang <?php echo $_POST["nama"]; ?>!<br />
```

```
</body>
```

```
</html>
```

Form HTML dibentuk menggunakan tag, dan tag form memiliki attribute method yang digunakan untuk menentukan bagaimana cara form mengirim data ke server (data dikirim ke halaman yang diset di dalam attribute action).

\$_POST termasuk dalam kategori Predefined Variable yang disediakan oleh library PHP. Kegunaan dari **\$_POST** hampir sama dengan **\$_GET** yaitu digunakan untuk mengambil suatu nilai yang dikirimkan oleh form bedanya **\$_POST** mengambil nilai yang dikirim oleh form dengan method="post".

Semua nilai / informasi yang dikirimkan oleh form yang menggunakan method POST tidak akan terlihat oleh user yang mengakses, dikarenakan informasi yang dikirim akan tidak ditampilkan di Address Bar Web Browser. Selain **\$_POST** juga tidak memiliki batasan pada jumlah informasi yang dikirim.

1.2.2 HTML (Hypertext Markup Language)

HTML adalah singkatan dari **Hypertext Markup Language**. HTML memungkinkan seorang user untuk membuat dan menyusun bagian paragraf, heading, link atau tautan, dan blockquote untuk halaman web dan aplikasi.

HTML bukanlah bahasa pemrograman, dan itu berarti HTML tidak punya kemampuan untuk membuat fungsionalitas yang dinamis. Sebagai gantinya, HTML memungkinkan user untuk mengorganisir dan memformat dokumen, sama seperti Microsoft Word.

Ketika bekerja dengan HTML, Anda menggunakan struktur kode yang sederhana (tag dan *attribute*) untuk *mark up* halaman website. Misalnya, Anda membuat sebuah paragraf dengan menempatkan *enclosed text* di antara tag pembuka `<p>` dan tag penutup `</p>`.

```
<p>This is how you add a paragraph in HTML.</p>
```

```
<p>You can have more than one!</p>
```

Kesimpulannya, pengertian HTML sebagai bahasa mark up sangatlah mudah untuk dipahami bahkan bagi webmaster pemula di bidang web development sekalipun.

1.2.3 Cara Kerja HTML

Dokumen HTML adalah file yang diakhiri dengan ekstensi **.html** atau **.htm**. Ekstensi file ini bisa dilihat dengan menggunakan web browser apa pun (seperti Google Chrome, Safari, atau Mozilla Firefox). Browser tersebut membaca file HTML dan *me-render* kontennya sehingga user internet bisa melihat dan membacanya.

Biasanya, rata-rata situs web menyertakan sejumlah halaman HTML yang berbeda-beda. Contohnya, beranda utama, halaman ‘tentang kami’, halaman kontak yang semuanya memiliki dokumen HTML terpisah.

Masing-masing halaman HTML terdiri atas seperangkat **tags** (bisa disebut juga **elements**), yang mengacu pada *building block* halaman

website. Tag tersebut membuat hirarki yang menyusun konten hingga menjadi bagian, paragraf, heading, dan *block* konten lainnya.

Sebagian besar element HTML memiliki tag pembuka dan penutup yang menggunakan syntax `<tag></tag>`.

1.2.3 GET

Dalam bahasa inggris kita akrab dengan istilah GETting, dari istilah tersebut dapat diartikan bahwa metode GET pada HTTP ditujukan untuk mengambil (get) data dari server.

Pada metode ini umumnya data berbentuk query string yang dikirim via url, data tersebut berupa pasangan `key=value` yang dipisahkan dengan tanda `&`. Data tersebut digabung dengan url utama yang dipisahkan dengan tanda `?`

Sebelum dikirim, terlebih dahulu data diproses sehingga memenuhi standar format URL. URL hanya boleh memuat **huruf** (besar dan kecil), **angka**, dan beberapa karakter lain dalam ASCII Character Set seperti (`“.-_~`), karakter di luar itu akan diubah ke format tertentu yang diawali tanda `%` kemudian diikuti dengan 2 digit hexadesimal.

Variabel `$_GET` pada PHP berbentuk associative array. Variabel ini bentuknya sama seperti variabel pada umumnya, bedanya `$_GET` ini merupakan variabel global sehingga bisa diakses dimana saja.

Karena bentuknya sama dengan yang lain, variabel ini dapat kita manipulasi sebagaimana kita memanipulasi variabel array lainnya, misal dengan menambahkan nilainya: `$_GET['status'] = 'aktif'` atau menghapusnya `unset($_GET['nama'])`.

1.2.3.1 Kelebihan dan Kekurangan

Terdapat beberapa kelebihan penggunaan metode GET, diantaranya adalah:

1. Sempel, dan data mudah diedit, misal untuk menuju halaman 5 dari suatu website, kita tinggal mengganti urlnya.
2. Halaman dapat dibookmark dan disimpan pada history browser sehingga mudah untuk diakses kembali.
3. Dapat kembali ke halaman sebelumnya dengan mudah (dengan mengklik tombol Back pada browser).
4. Dapat direfresh dengan mudah.
5. Dapat di distribusikan/dishare.

Meskipun banyak kelebihannya, penggunaan metode ini memiliki beberapa kelemahan yaitu:

1. Panjang data terbatas hanya 2kb – 8kb (tergantung browsernya), jika melebihi batas tersebut akan muncul pesan error 414 Request-URI Too Long, sehingga tidak dapat digunakan untuk mengirim data dalam jumlah besar.
2. Hanya dapat mengirim data jenis teks, jenis lainnya seperti: gambar, file zip, dll tidak dapat dikirim.
3. Karena data dikirim via URL, data tersebut mudah terekspose.

1.2.4 POST

Pada protokol HTTP, metode POST dapat dikirim baik melalui query string maupun body, seperti pada GET, data yang dikirim melalui query string akan ditampilkan pada URL dan sedangkan yang dikirim melalui body tidak terlihat oleh user.

Pada PHP, data POST yang dikirim melalui query string disimpan pada variabel `$_GET` (seperti metode GET) sedangkan yang dikirim melalui body disimpan pada variabel `$_POST`.

Sama seperti `$_GET`, variabel `$_POST` juga berbentuk associative array dan bersifat global yang artinya dapat diakses dimana saja, selain itu juga dapat dilakukan manipulasi sebagaimana variabel array lainnya. Penggunaan metode POST sering kita jumpai terutama pada saat pengiriman data menggunakan form html.

1.2.4.1 Kelebihan dan kekurangan

Pengiriman data menggunakan metode POST memiliki beberapa kelebihan diantaranya:

1. Lebih aman dari pada metode GET karena data yang dikirim tidak terlihat, serta parameter yang dikirim tidak disimpan pada history browser/log browser.
2. Dapat mengirim data dalam jumlah besar.
3. Dapat mengirim berbagai jenis data seperti gambar, file, dll, tidak harus teks.

Meskipun terdapat kelebihan, penggunaan metode ini juga memiliki beberapa kelemahan, walaupun sebenarnya bukan kelemahan melainkan memang menjadi karakteristik dari metode ini:

1. Data tidak disimpan pada history browser.
2. Data tidak dapat dibookmark.
3. Karena dianggap sebagai data sensitif, maka ketika kita meresh browser, akan muncul konfirmasi pengiriman ulang data, demikian juga ketika kita tekan tombol back.

BAB II

PEMBAHASAN

2.1 Pembahasan Tugas Praktikum

Pada tugas dimodul 2 ini diminta untuk membuat program web yang dapat menginputkan username dan password menggunakan form dan penanganan input data. Langkah pertama terlebih dahulu membuat sebuah database agar ketika menginputkan username dan passwordnya telah terdaftar atau data tersebut sudah ada di dalam database. Buka localhost menggunakan aplikasi xampp, setelah itu akan terhubung ke phpmyadmin. Buat database baru dan beri nama fakultas, selanjutnya buat satu tabel dengan nama admin. Isi tabel tersebut dengan 3 kolom dengan judul id, username, dan password pada tiap kolom. Id merupakan primary key. Setelah itu input datanya seperti gambar di bawah ini.



Gambar 2.1 Database fakultas

Pada gambar di atas terlihat usernamnya adalah ‘mikelis’ dan passwordnya adalah ‘Mikel011’. Setelah membuat database, buat file phpnya dengan nama koneksi agar ketika login dapat terhubung ke database. Adapun kode programnya seperti di bawah ini.

```
<?php  
$koneksi =
```

```

mysqli_connect("localhost","root","","fakultas");

if (mysqli_connect_errno()){

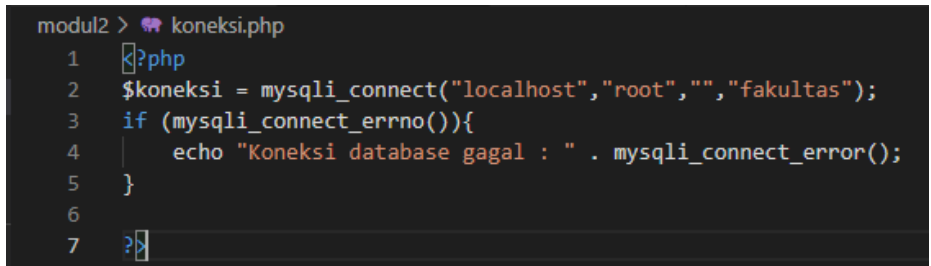
    echo "Koneksi database gagal : " .
mysqli_connect_error();

}

?>

```

Adapun tampilannya seperti di bawah ini.



```

modul2 > koneksi.php
1  <?php
2  $koneksi = mysqli_connect("localhost","root","","fakultas");
3  if (mysqli_connect_errno()){
4      echo "Koneksi database gagal : " . mysqli_connect_error();
5  }
6
7  ?>

```

Gambar 2.2 Tampilan koneksi.php

Pada tahap selanjutnya buat lagi file PHP dengan judul index.php. Dalam file ini merupakan bagian membuat halaman login sederhana untuk admin. Kode program index.php seperti di bawah ini.

```

<!DOCTYPE html>

<html>

<head>

    <title>Halaman Login fakultas</title>

</head>

<body>

```

```
<h2>Login</h2>

<br/>

<?php

if(isset($_GET['pesan'])) {

    if($_GET['pesan'] == "gagal") {

        echo "Login gagal! username dan
password salah!";

    }else if($_GET['pesan'] == "logout") {

        echo "Anda telah berhasil logout";

    }else if($_GET['pesan'] == "belum_login") {

        echo "Anda harus login untuk mengakses
halaman admin";

    }

}

?>

<br/>

<br/>

<form method="post" action="login.php">

    <table>

        <tr>

            <td>Username</td>

            <td>:</td>

            <td><input type="text"
name="username" placeholder="Masukkan username"></td>

        </tr>

    </table>

</form>
```

```

        <tr>

            <td>Password</td>

            <td>:</td>

            <td><input type="password"
name="password" placeholder="Masukkan password"></td>

        </tr>

        <tr>

            <td></td>

            <td></td>

            <td><input type="submit"
value="LOGIN"></td>

        </tr>

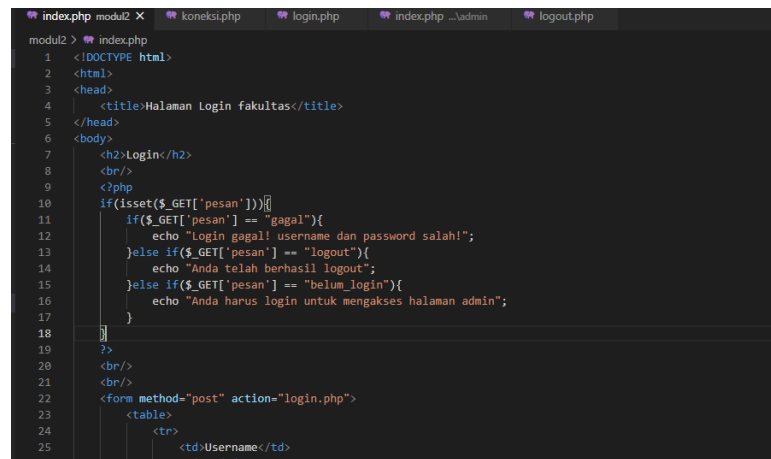
    </table>

</form>

</body>

</html>
```

Adapun tampilannya seperti di bawah ini.



```
modul2 > index.php
1 <!DOCTYPE html>
2 <html>
3 <head>
4 <title>Halaman Login fakultas</title>
5 </head>
6 <body>
7 <h2>Login</h2>
8 <br/>
9 <?php
10 if(isset($_GET['pesan'])) {
11     if($_GET['pesan'] == "gagal") {
12         echo "Login gagal! username dan password salah!";
13     } else if($_GET['pesan'] == "logout") {
14         echo "Anda telah berhasil logout";
15     } else if($_GET['pesan'] == "belum_login") {
16         echo "Anda harus login untuk mengakses halaman admin";
17     }
18 }
19 ?>
20 <br/>
21 <br/>
22 <form method="post" action="login.php">
23 <table>
24 <tr>
25 <td>Username</td>
```

Gambar 2.3 Tampilan file index

Tahap berikutnya lagi buat file php dengan nama login, dalam file ini yang merupakan aksi dari file index.php di atas. Pada bagian ini menggunakan POST yang merupakan metode pertukaran data pada protokol HTTP. Adapun kode programnya seperti di bawah ini.

```
<?php

session_start();

include 'koneksi.php';

$username = $_POST['username'];

$password = $_POST['password'];

$data = mysqli_query($koneksi,"select * from admin
where username='$username' and
password='$password'");

$cek = mysqli_num_rows($data);

if($cek > 0) {

    $_SESSION['username'] = $username;
```

```

        $_SESSION['status'] = "login";

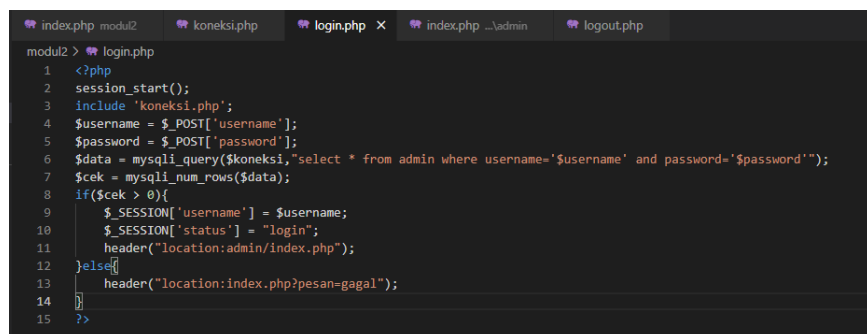
        header("location:admin/index.php");
    }else{

        header("location:index.php?pesan=gagal");
    }

?>

```

Adapun tampilannya seperti di bawah ini.



```

modul2 > login.php
1  <?php
2  session_start();
3  include 'koneksi.php';
4  $username = $_POST['username'];
5  $password = $_POST['password'];
6  $data = mysqli_query($koneksi,"select * from admin where username='$username' and password='$password'");
7  $cek = mysqli_num_rows($data);
8  if($cek > 0){
9      $_SESSION['username'] = $username;
10     $_SESSION['status'] = "login";
11     header("location:admin/index.php");
12 }else{
13     header("location:index.php?pesan=gagal");
14 }
15 ?>

```

Gambar 2.4 Tampilan file login

Pada bagian ini terdapat ‘session_start()’ yang berfungsi sebagai mengeksekusi session pada server dan kemudian menyimpannya pada browser. Selanjutnya jangan lupa untuk menghubungkan file ini dengan file koneksi databasenya yaitu dengan menginputkan ‘include koneksi.php’. Terdapat perintah if yang didalam terdiri dari beberapa variabel, pada bagian ini berfungsi untuk mengecek apakah username dan password yang diinputkan telah terdaftar. Jika tidak ditemukan sistem akan mengirimkan pesan ‘gagal’. Adapun kode programnya seperti di bawah ini.

```
if($cek > 0){  
  
    $_SESSION['username'] = $username;  
  
    $_SESSION['status'] = "login";  
  
    header("location:admin/index.php");  
}  
else{  
  
    header("location:index.php?pesan=gagal");  
  
}
```

Tahap selanjutnya buat folder baru dan beri nama admin. Pada folder ini terdapat file index.php untuk admin dan file logout jika ingin keluar dari sistem. Kode program dari index.php untuk admin seperti di bawah ini.

```
<!DOCTYPE html>  
  
<html>  
  
<head>  
  
    <title>Halaman Login Fakultas</title>  
</head>  
  
<body>  
  
    <h2>Halaman Admin</h2>  
  
  
    <br/>  
  
  
    <?php  
  
    session_start();
```

```
        if($_SESSION['status']!="login"){

            header("location:../index.php?pesan=belum_login")
        ;

        }

        ?>

        <h4>Selamat datang, <?php echo
$_SESSION['username']; ?>! anda telah login.</h4>

        <br/>

        <br/>

        <a href="logout.php">LOGOUT</a>

    </body>
</html>
```


Adapun tampilannya seperti di bawah ini.

```
modul2 > admin > index.php
1  <!DOCTYPE html>
2  <html>
3  <head>
4      <title>Halaman Login Fakultas</title>
5  </head>
6  <body>
7      <h2>Halaman Admin</h2>
8
9      <br/>
10
11     <?php
12         session_start();
13         if($_SESSION['status']!="login"){
14             header("location:../index.php?pesan=belum_login");
15         }
16     >
17
18     <h4>Selamat datang, <?php echo $_SESSION['username']; >! anda telah login.</h4>
19
20     <br/>
21     <br/>
22
23     <a href="logout.php">LOGOUT</a>
24
25
26 </body>
27 </html>
```

Gambar 2.5 Tampilan file index dalam folder admin

Dari kode program atau gambar di atas terdapat pernyataan if else yang berisi

```
if($_SESSION['status']!="login"){

header("location:../index.php?pesan=belum_login");

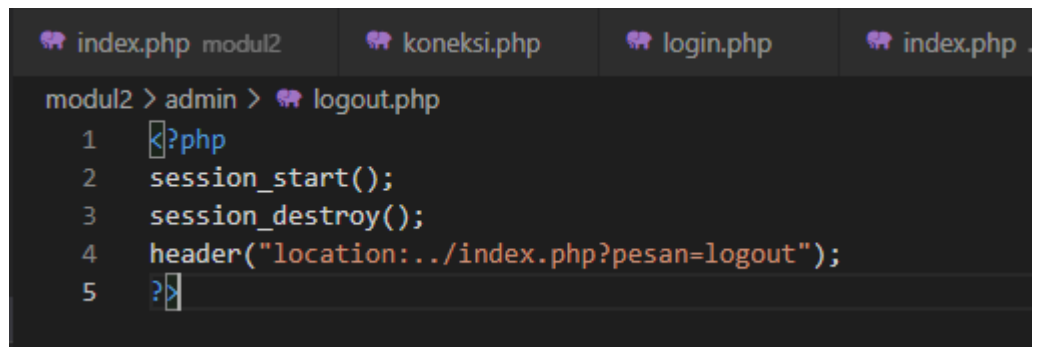
}
```

Bagian ini merupakan pengecekan apakah sudah login atau belum. Jika session status tidak sama dengan login maka halaman akan di alihkan ke halaman login lagi sambil mengirim pesan belum login. Pengecekan ini berfungsi untuk keamanan agar halaman admin tidak bisa diakses sebelum melakukan login.

Pada tahap terakhir buat file PHP dengan nama logout yang merupakan bagian dari sistem apabila ingin keluar dari halaman login. Adapun kode programnya seperti di bawah ini.

```
<?php  
  
session_start();  
  
session_destroy();  
  
header("location:../index.php?pesan=logout");  
  
?>
```

Adapun tampilan kode programnya seperti di bawah ini.

A screenshot of a code editor interface. At the top, there are four tabs: 'index.php modul2', 'koneksi.php', 'login.php', and 'index.php'. The active tab is 'logout.php'. The editor shows the following PHP code:

```
modul2 > admin > logout.php  
1  <?php  
2  session_start();  
3  session_destroy();  
4  header("location:../index.php?pesan=logout");  
5  ?>
```

Gambar 2.6 Tampilan file logout

File logout ini berfungsi sebagai menghapus semua session yang telah di buat saat login dan mengalihkan halamn kembali ke halaman login sambil mengirim pesan logout. Pada file ini terdapat session start() dan session_destroy().Kegunaan dari fungsi session_start(); adalah untuk memulai eksekusi session pada server dan kemudian menyimpannya pada browser. Sedangkan Pada bagian session_destroy() ini menjelesakan bahwa semua session yang telah di buat dan di inisialisasi akan di destroy atau di hancurkan.

BAB III

KESIMPULAN

Setelah saya mempelajari modul 2 ini tentang Form Handling dapat saya simpulkan yaitu :

PHP Form Handling merupakan hal penting dalam sebuah website, form handling digunakan untuk mengambil data dari inputan user melalui form HTML.

Form HTML dibentuk menggunakan tag, dan tag form memiliki attribute method yang digunakan untuk menentukan bagaimana cara form mengirim data ke server (data dikirim ke halaman yang diset di dalam attribute action).

HTML adalah singkatan dari Hypertext Markup Language. HTML memungkinkan seorang user untuk membuat dan menyusun bagian paragraf, heading, link atau tautan, dan blockquote untuk halaman web dan aplikasi.

\$_POST termasuk dalam kategori Predefined Variable yang disediakan oleh library PHP. Kegunaan dari **\$_POST** hampir sama dengan **\$_GET** yaitu digunakan untuk mengambil suatu nilai yang dikirimkan oleh form bedanya **\$_POST** mengambil nilai yang dikirim oleh form dengan method="post".

DAFTAR PUSTAKA

School, D. (2021). Fungsi dan Cara Menggunakan \$_SESSION PHP - Kursus Website Terbaik. Retrieved 4 April 2021, from <https://www.kursuswebsite.org/fungsi-dan-cara-menggunakan-session-php/>.

Perbedaan POST dan GET. (2021). Retrieved 4 April 2021, from <https://www.dumetschool.com/blog/perbedaan-post-dan-get>.

PHP Form Handling. (2021). Retrieved 4 April 2021, from <https://abangpreuner.wordpress.com/2014/02/04/976/>.

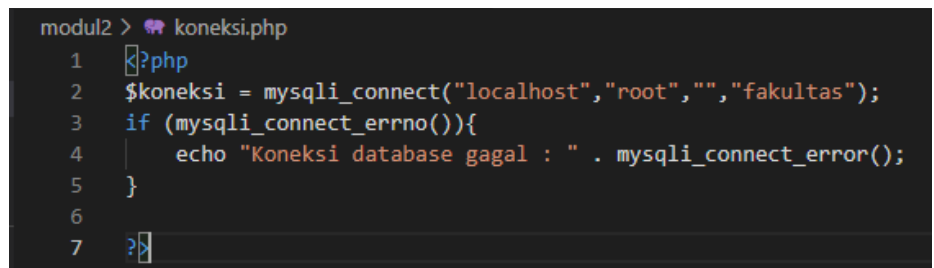
Belajar HTML #11: Cara Membuat Form pada HTML. (2021). Retrieved 4 April 2021, from <https://www.petanikode.com/html-form/>.

Hadi, A. (2021). Memahami GET dan POST Pada PHP dan HTTP | Jagowebdev. Retrieved 4 April 2021, from https://jagowebdev.com/memahami-get-dan-post-pada-php-dan-http/#:~:text=GET%20dan%20POST%20merupakan%20metode,disimpan%20pada%20variabel%20%24_GET.

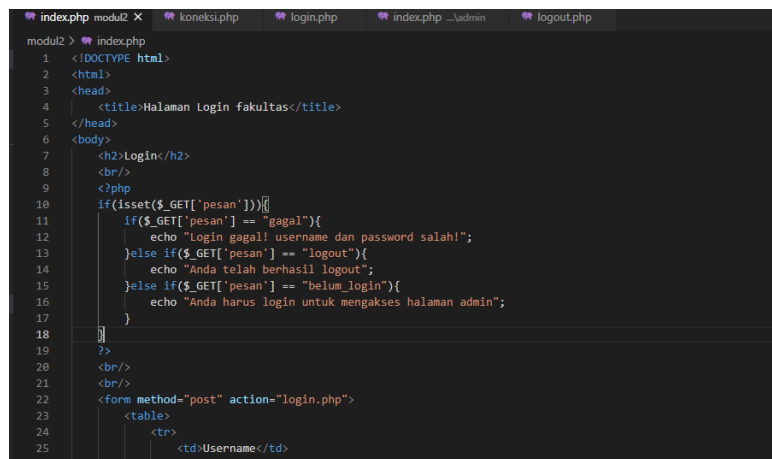
LAMPIRAN



Gambar 2.1 Database fakultas



Gambar 2.2 Tampilan koneksi.php



Gambar 2.3 Tampilan file index

```
modul2 > login.php
1  <?php
2  session_start();
3  include 'koneksi.php';
4  $username = $_POST['username'];
5  $password = $_POST['password'];
6  $data = mysqli_query($koneksi,"select * from admin where username='$username' and password='$password'");
7  $cek = mysqli_num_rows($data);
8  if($cek > 0){
9      $_SESSION['username'] = $username;
10     $_SESSION['status'] = "login";
11     header("location:admin/index.php");
12 }else{
13     header("location:index.php?pesan=gagal");
14 }
15 ?>
```

Gambar 2.4 Tampilan file login

```
modul2 > admin > index.php
1  <!DOCTYPE html>
2  <html>
3  <head>
4      <title>Halaman Login Fakultas</title>
5  </head>
6  <body>
7      <h2>Halaman Admin</h2>
8
9      <br/>
10
11     <?php
12     session_start();
13     if($_SESSION['status']!="login"){
14         header("location:../index.php?pesan=belum_login");
15     }
16     ?>
17
18     <h4>Selamat datang, <?php echo $_SESSION['username']; ?>! anda telah login.</h4>
19
20     <br/>
21     <br/>
22
23     <a href="logout.php">LOGOUT</a>
24
25
26 </body>
27 </html>
```

Gambar 2.5 Tampilan file index dalam folder admin

```
modul2 > admin > logout.php
1  <?php
2  session_start();
3  session_destroy();
4  header("location:../index.php?pesan=logout");
5  ?>
```

Gambar 2.6 Tampilan file logout