LAPORAN HASIL PRAKTIKUM
PEMROGRAMAN WEB & MOBILE I

NAMA : **ELSAN PENTANUGRAHA**

NIM : **193030503047**

KELAS : **A**

MODUL : **II (FORM HANDLING)**

JURUSAN TEKNIK INFORMATIKA

FAKULTAS TEKNIK

UNIVERSITAS PALANGKA RAYA

2021

BAB I
LANDASAN TEORI

1.1 Tujuan

1.1.1. Mahasiswa mampu membuat handling yang mampu mengolah data dari form HTML.

1.1.2. Mahasiswa mampu membuat batasan-batasan untuk menangani inputan dari form HTML.

1.2 Landasan Teori

1.2.1. Variabel superglobal

Variabel super global di PHP adalah variabel bawaan yang bersifat global. Variabel bawaan yang dimaksud adalah: variabel yang sudah otomatis ada tanpa perlu kita definisikan sendiri. Dan ia bersifat global dalam artian bisa kita akses dari mana pun dan kapan pun.[1]

Variabel super global menyimpan banyak sekali data penting dan juga bermanfaat yang bisa kita gunakan dalam menyelesaikan projek yang sedang kita kerjakan.[1]

1. Variabel $_SERVER

   Variabel yang pertama dan utama adalah variabel $_SERVER. Ia adalah sebuah array asosiatif yang menyediakan berbagai macam informasi tentang request yang ditangkap oleh server. Data yang dimuat berupa headers, paths, lokasi skrip, dan sebagainya.[1]

2. Variabel $_GET

   Variabel $_GET adalah array asosiatif yang berisi nilai dari query string.[1]

3. Variabel $_POST

   Variabel $_POST mirip dengan variabel $_GET. Hanya saja data yang di-passing tidaklah melalui query string pada URL, akan tetapi pada body request. Dan request method yang dilakukan haruslah dengan metode POST.[1]

4. Variabel $_SESSION

   Variabel $_SESSION merupakan array asosiatif yang menyimpan data sesi pengguna.Variabel ini bisa kita gunakan untuk menyimpan user yang login pada satu sesi tertentu. Atau juga bisa digunakan untuk menyimpan data cart pada toko online. Secara default, umur sesi pada PHP adalah 1440 detik atau 24 menit.[1]

5. Variabel $_REQUEST

   Variabel $_REQUEST adalah array asosiatif yang menyimpan gabungan nilai dari variabel $_GET, $_POST, dan $_COOKIE yang kesemuanya berhubungan dengan data yang dikirim bersamaan dengan request user.[1]

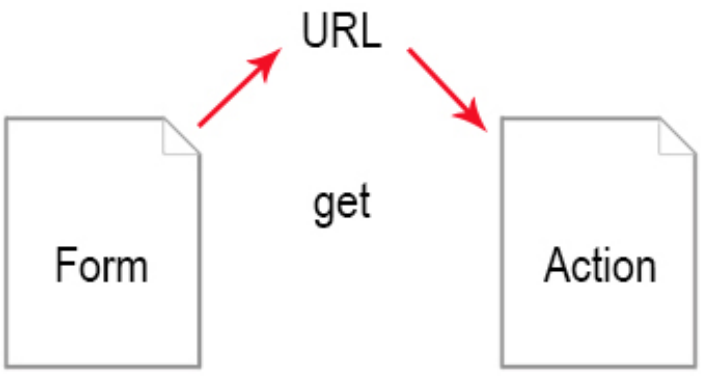
6. Variabel $GLOBALS

   Variabel $GLOBALS adalah array asosiatif yang menyimpan semua variabel global yang didefinisikan saat program dijalankan. Kita telah membahas tentang variabel ini dalam pembahasan ruang lingkup variabel.[1]

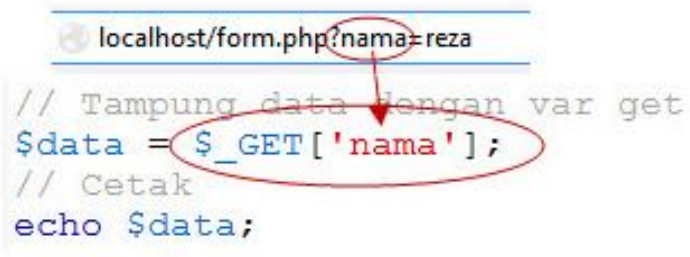
1.2.2. GET & POST

1. Metode GET

   Pada metode ini data yang dikirim adalah URL yang berupa rangkaian pasangan nama dan nilai yang dipisahkan oleh ampersand (&).[2]



Gambar 1.2.2.1 Contoh metode GET.



Gambar 1.2.2.2 Contoh syntax metode GET.

Keuntungan dan Kerugian Menggunakan Metode GET

1) Karena data yang dikirim oleh metode GET ditampilkan dalam URL, dimungkinkan untuk menandai halaman dengan nilai string kueri tertentu.
2) Metode ini tidak pas untuk menyampaikan informasi sensitif misalnya nama pengguna dan kata sandi, karena ini sepenuhnya terlihat dalam string kueri URL serta berpotensi disimpan dalam memori browser klien sebagai halaman yang dikunjungi.
3) Karena metode GET memberikan data ke variabel lingkungan server, panjang URL terbatas. Jadi, ada batasan untuk total data yang akan dikirim.

PHP menyediakan variabel superglobal $ _GET untuk mengakses semua informasi yang dikirim baik melalui URL atau dikirimkan melalui formulir HTML menggunakan metode = “get”.[2]

2. Metode POST

Dalam metode ini data yang dikirim kepada server sebagai paket yang dalamnya berisi komunikasi terpisah dengan skrip pemrosesan. Proses data yang dikirim tidak akan tampak pada URL.[2]

Gambar 1.2.2.3 Contoh metode POST.

```
// Tampung data dengan var post
$data = $_POST['nama'];
// Cetak
echo $data;
```

Gambar 1.2.2.4 Contoh syntax metode POST.

Keuntungan dan Kerugian Menggunakan Metode POST

1. Ini lebih aman daripada GET karena informasi yang dimasukkan pengguna tidak pernah terlihat dalam string kueri URL atau dalam log server.
2. Ada batasan yang jauh lebih besar pada jumlah data yang dapat dilewati dan seseorang dapat mengirim data teks serta data biner (mengunggah file) menggunakan POST.
3. Karena metode ini pada proses data yang dikirim tidak akan tampak pada URL, maka tidak mungkin untuk mem-bookmark halaman dengan permintaan tertentu.

Misalnya metode $ _GET, PHP menyediakan juga variabel yang superglobal lain $ _POST untuk mengakses semua informasi yang dikirim melalui metode posting atau dikirimkan melalui formulir HTML menggunakan metode = “post”.[2]

1.2.3. Validasi Form PHP

Form merupakan component pada Website yang sangat sering digunakan, apalagi pada sebuah sistem informasi, karena digunakan untuk menginput data. Namun agar data yang diinput Valid atau sesuai keinginan kita, maka kita perlu membuat Validasi data yang diinput pada Form tersebut.[3]

Kode HTML untuk membentuk Form tersebut adalah sebagai berikut:

1. Text Field

   Field nama, email dan website adalah elemen-elemen text input, dan field komentar adalah textarea yaitu sebagai berikut:

```
Name: <input type="text" name="name">
Password: <input type="password" name="password">
```

2. Radio Button

   Field jenis kelamin adalah radio button yaitu sebagai berikut:

   Gender:

```
<input type="radio" name="gender" value="female">Female
<input type="radio" name="gender" value="male">Male
```

3. Form Element

   Kode HTML untuk membentuk form pada gambar diatas adalah sebagai berikut:

```
<form method="post" action="<?php echo
htmlspecialchars($_SERVER["PHP_SELF"]);? >">
```

Ketika form disubmit, data pada form dikirim dengan method “post”. $_SERVER[“PHP_SELF”] adalah variabel super global yang mengembalikan nama file dari skrip yang sedang dieksekusi. Sehingga kode form diatas mengirim data pada form ke halaman itu sendiri. Sedangkan fungsi htmlspecialchars() adalah fungsi yang mengkonversikan karakter-karakter spesial ke entitas HTML. Sebagai contoh, fungsi tersebut akan mengkonversikan karakter < dan > menjadi < dan >. Fungsi ini mencegah injeksi yang bisa

dilakukan dengan HTML atau javascript (Cross-site Scripting Attack) pada form tersebut.[4]

Validasi Nama

Kode berikut menunjukkan cara sederhana untuk memeriksa apakah field nama hanya mengandung huruf dan spasi. Jika nilai dari nama tidak valid, maka pesan error akan disimpan didalam variabel $nameErr:

```
$name = test_input($_POST["name"]);
if (!preg_match("/^[a-zA-Z ]*$/",$name)) {
        $nameErr = "Only letters and white space allowed";
}
```

Gambar 1.2.3.1 contoh validasi nama.

Fungsi preg_match() mencari string berdasarkan pola, mengembalikan nilai true jika polanya ada, false jika polanya tidak ada.[4]

Pentingnya Melakukan Validasi Nilai Form

Nilai yang telah diinput oleh user atau pengunjung web, tidak bisa begitu saja di simpan langsung ke dalam database. Karena kita tidak tahu apakah nilai tersebut telah sesuai dengan nilai yang kita kehendaki. Misalkan apakah nilai tersebut harus berupa angka, atau hanya bisa berupa huruf, atau apakah hanya bisa diinput dalam range tertentu saja.[5]

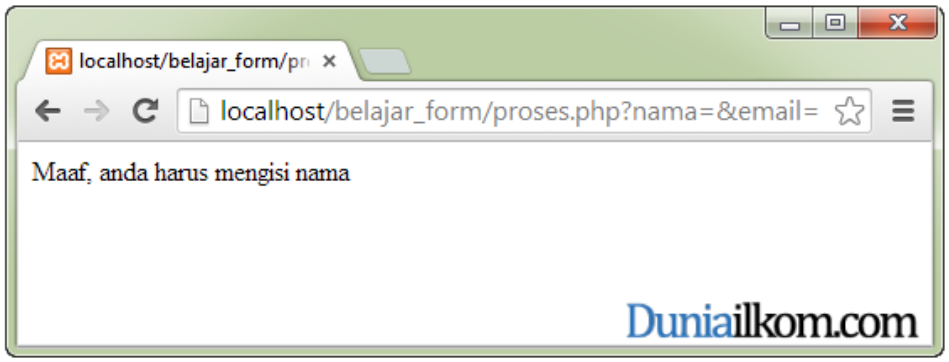
Dalam kasus yang ekstrim, seorang user bisa saja memasukkan kode script atau tag HTML yang bisa merusak situs kita, hal ini dikenal dengan Cross-site Scripting. Sebuah proses validasi nilai merupakan hal yang sangat penting dalam merancang form. Khusus untuk validasi mencegah Cross-site Scripting dan juga HTML injection ini akan saya bahas pada tutorial form PHP berikutnya.[5]

```php
<?php
if (isset($_GET['nama']) AND isset($_GET['email']))
{
   $nama=$_GET['nama'];
   $email=$_GET['email'];
}
else
{
   die("Maaf, anda harus mengakses halaman ini dari form.html");
}

if (!empty($nama))
{
   echo "Nama: $nama <br /> Email: $email";
}
else
{
   die("Maaf, anda harus mengisi nama");
}
?>
```

Gambar 1.2.3.2 Contoh syntax validasi.

Dalam kode PHP diatas, saya memodifikasi beberapa bagian kode program. Pada logika IF pertama, saya melakukan pengecekan apakah variabel $_GET[‘nama’] dan $_GET[‘email’] tersedia atau tidak. Jika tersedia maka pindahkan nilainya ke variabel $nama dan $email agar lebih mudah untuk diproses. Namun jika tidak, fungsi die() akan menghentikan proses dan menampilkan pesan kesalahan. [5]

Pada logika IF kedua, saya memeriksa apakah variabel $nama kosong atau tidak dengan fungsi !empty(). Fungsi !empty($nama) akan menghasilkan nilai true hanya jika variabel $nama tidak kosong (perhatikan tanda ! sebagai pembalik logika empty()). Namun jika $nama ternyata kosong (tidak diisi), maka tampilkan pesan kesalahan.[5]



Gambar 1.2.3.3 Contoh pesan yang akan ditampilkan jika data telah divalidasi.

1.3 Tugas Praktikum

Buatlah program web untuk menginputkan username dan password menggunakan form dan penanganan input data dengan kriteria sebagai berikut:

1.3.1. Username yang dinputkan tidak boleh lebih dari tujuh karakter.

1.3.2. Password yang dinputkan harus terdiri dari huruf kapital, huruf kecil, angka dan karakter khusus.

1.3.3. Jumlah karakter password tidak boleh kurang dari sepuluh karakter.

BAB II
PEMBAHASAN

Untuk menyelesaikan tugas yang ada pada modul 2 ini aplikasi yang digunakan adalah visual studio code yang digunakan sebagai code editor untuk membuat program login form dengan bahasa PHP. Kemudian XAMPP yang digunakan sebagai server yang berdiri sendiri (localhost) dan juga web browser Chrome yang digunakan untuk menjalankan program web login form yang telah dibuat.

Pada soal terdapat syarat yang harus dibuat sebagai handling pada login form. Syarat atau kriteria ini dibuat untuk melakukan validasi data username dan password yang telah diinputkan. Berikut adalah pembahasan dari kriteria login yang telah dibuat.

2.1. Pembuatan Form pada login.php

Yang pertama dilakukan adalah pembuatan form, berikut adalah hasil tangkapan layar baris kode program pada pembuatan login form.

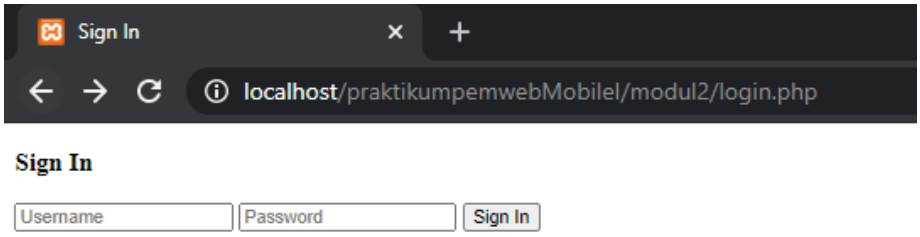
login.php

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Sign In</title>
</head>
<body>
    <form class="boxes" method="post" action="welcome.php">
    <h3>Sign In</h3>
    <input name="uname" type="text" placeholder="Username">
    <input name="pwd" type="pwd" placeholder="Password">
    <input name="Submit" type="submit" value="Sign In">
    </form>
</body>
</html>
```

Gambar 2.1.1 baris program login form.

Dari baris kode program diatas, pada tag title dibuat text “Sign In” yang nantinya akan muncul pada bagian tab pada web browser. Kemudian didalam tag body dideklarasikan kelas dari form yang bernama *boxes* dengan *method=“post”* dan juga *action=”welcome.php”* yang nantinya welcome.php akan digunakan untuk mengisi baris kode program yang akan melakukan handling. Kemudian juga

dideklarasikan header dengan teks “Sign In”. Untuk membuat kolom input data username dan password dibuat dengan baris kode program *<input name="uname" type="text" placeholder="Username">* yang akan digunakan untuk membuat kolom input username. Lalu *<input name="pwd" type="pwd" placeholder="Password">* yang akan digunakan untuk membuat kolom input password. Dan *<input name="Submit" type="submit" value="Sign In">* yang digunakan untuk membuat tombol Sign In yang akan melakkukan submit untuk melakukan validasi data yang telah diinputkan. Berikut adalah hasil tangkapan layar dari baris program yang dijalankan melalui web browser.



Gambar 2.1.2 hasil dari pembuatan login form.

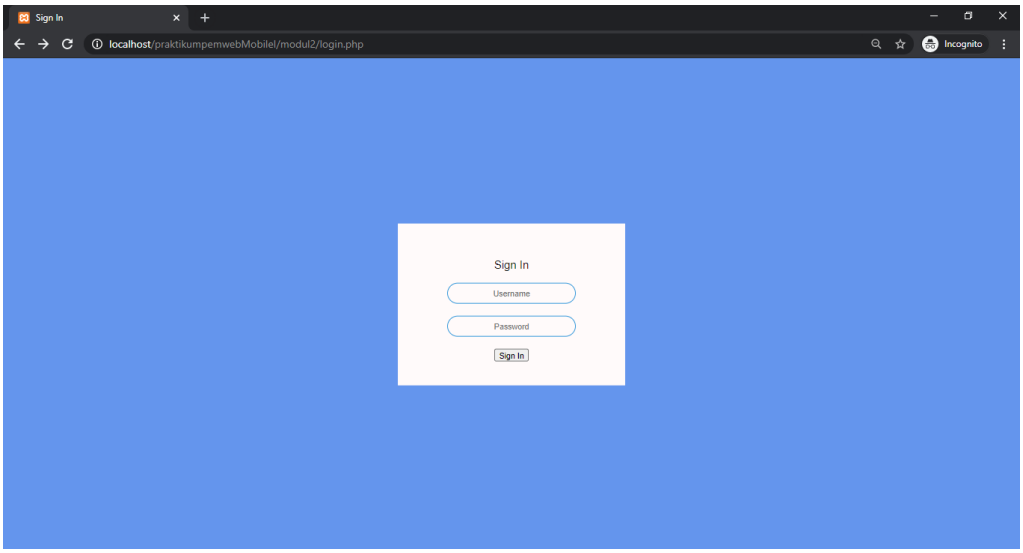
Pada bagian head dari login.php saya buat tag <style> yang didalam nya digunakan untuk mengatur tampilan dari form login yang dibuat. Tag ini ditempatkan pada bagian tag <head>. Berikut adalah hasil tangkapan layar baris kode program yang mengatur tampilan login form.

Gambar 2.1.3 baris program internal CSS.

```
25          .boxes input[type="text"],.boxes input[type="pwd"]{
26          border:0;
27          background:none;
28          display:block;
29          margin:20px auto;
30          text-align:center;
31          border:2px solid #3498db;
32          padding:8px 5px;
33          width:200px;
34          outline:none;
35          border-radius:24px;
36          transition: 0.25s;
37          }
38      </style>
39      <meta charset="UTF-8">
40      <meta http-equiv="X-UA-Compatible" content="IE=edge">
41      <meta name="viewport" content="width=device-width, initial-scale=1.0">
42      <title>Sign In</title>
43  </head>
```

Gambar 2.1.4 baris program internal CSS.

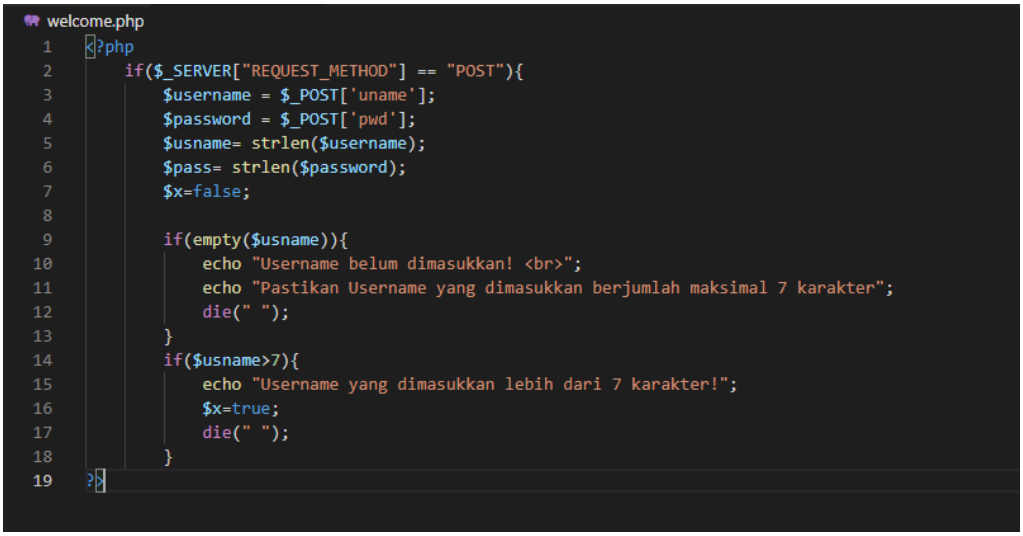
Ketika program dijalankan, maka tampilan login form akan berubah. Berikut adalah hasil tangkapan layar ketika program dijalankan pada web browser.



Gambar 2.1.5 login form yang telah diubah menggunakan internal CSS.

2.2. Handling username yang dinputkan tidak boleh lebih dari tujuh karakter

Pada pembuatan handling untuk melakukan validasi username, baris kode program dibuat pada file “welcome.php”. berikut adalah tangkapan layar pembuatan handling welcome.php.



Gambar 2.2.1 pembuatan handling username.

Dari gambar diatas, baris program diatas yaitu:

```
if($_SERVER["REQUEST_METHOD"] == "POST"){
    $username = $_POST['uname'];
    $password = $_POST['pwd'];
    $usname= strlen($username);
    $pass= strlen($password);
    $x=false;
```

Baris program ini digunakan untuk mendeklarasikan variabel-variabel yang akan menyimpan nilai dari inputan username dan password. Variabel usname dan pass digunakan untuk menghitung pannjang nilai yang dimasukkan pada kolom input username dan password. Dan variabel x menyatakan nilai false.

Kemudian pada baris program:

```
if(empty($usname)){
    echo "Username belum dimasukkan! <br>";
    echo "Pastikan Username yang dimasukkan berjumlah maksimal 7 karakter";
    die(" ");
}
```

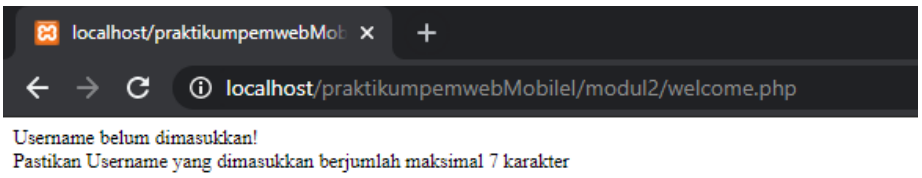
Digunakan untuk melakukan pengecekan apakah username telah dimasukkan atau belum. Dengan kondisi jika data pada variabel usname kosong maka akan menampilkan pesan “username belum dimasukan!”.

Kemudian pada baris program:

```
if($usname>7){
    echo "Username yang dimasukkan lebih dari 7 karakter!";
    $x=true;
    die(" ");
}
```

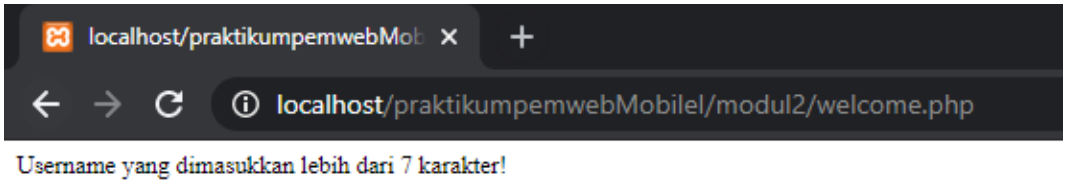
Digunakan untuk melakukan pengecekan apakah username yang dimasukkan tidak melebihi tujuh karakter, jika username yang dimasukan melebihi tujuh karakter maka program akan menampilkan pesan “*Username yang dimasukkan lebih dari 7 karakter!*”.

Berikut adalah hasil tangkapan layar program handling ketika telah selesai melakukan validasi.



Gambar 2.2.2 pesan ketika melakukan submit tanpa menginputkan username.

Gambar 2.2.3 percobaan melakukan login dengan username lebih dari 7 karakter.



Gambar 2.2.4 pesan ketika menginputkan username lebih dari 7 karakter.

2.3. Handling password yang dinputkan harus terdiri dari huruf kapital, huruf kecil, angka dan karakter khusus

Pada pembuatan handling untuk melakukan validasi password, baris kode program dibuat pada file "welcome.php". Berikut adalah tangkapan layar pembuatan handling welcome.php.

```
20      $upperc = preg_match('@[A-Z]@', $password);
21      $lowerc = preg_match('@[a-z]@', $password);
22      $angka    = preg_match('@[0-9]@', $password);
23      $specialc = preg_match('@[^\w]@', $password);
24
25      if(!$upperc || !$lowerc || !$angka || !$specialc ){
26          echo 'Password lemah';
27      }
```

Gambar 2.3.1 baris program pengecekan password.

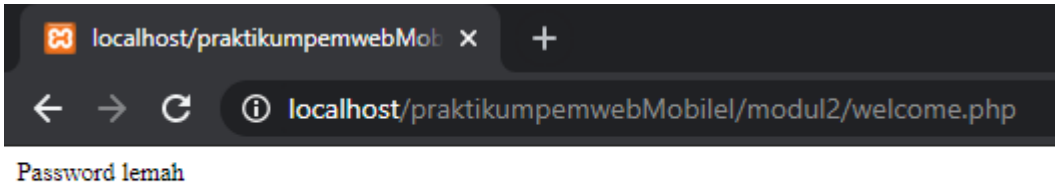
Dari baris program diatas variabel upperc digunakan untuk melakukan pengecekan apakah password yangdiinputkan telah memiliki huruf kapital. Variabel lowerc digunakan untuk melakukan pengecekan apakan password yang diinputkan memiliki huruf kecil. Varibael angka digunakan untuk melakukan pengecekan apakah password yang telah diinputkan memiliki angka. Dan variabel yang terakhir adalah sepcialc yang digunakan untuk melakukan pengecekan apakah password telah memiliki karakter khusus.

Kemudian dideklarasikan kondisi if berikut:

```
if(!$upperc || !$lowerc || !$angka || !$specialc ){
        echo 'Password lemah';
}
```

Pendeklarasian kondisi ini digunakan untuk melakukan pengecekan password, dengan kondisi apakah password tidak memiliki huruf kapital atau tidak memiliki huruf kecil atau tidak memiliki angka ataupun juga tidak memiliki karakter khusus maka program akan menampilkan pesan "*Passowrd lemah*". Berikut adalah hasil tangkapan layar pengujian dengan menginputkan password tanpa adanya karakter khusus.

Gambar 2.3.2 pengujian input password tanpa karakter spesial.



Gambar 2.3.3 pesan yang ditampilkan ketika password tidak memenuhi kriteria.

Pesan yang ditampilkan adalah password lemah karena password tidak memenuhi kriteria yang ditentukan. Jadi password yang dimasukkan harus memenuhi semua kriteria yang ada yaitu huruf kapital, huruf kecil, angka, dan karakter khusus.

2.4. Handling jumlah karakter password tidak boleh kurang dari sepuluh karakter

Pada pembuatan handling untuk melakukan validasi password, baris kode program dibuat pada file “welcome.php”. Berikut adalah tangkapan layar pembuatan handling welcome.php.

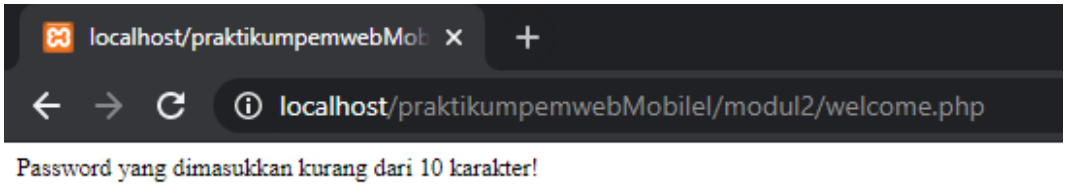
```php
if(empty($pass)){
    echo "Password belum dimasukkan!";
    echo "<br> Password harus memiliki Huruf kapital, huruf kecil, angka & karakter khusus.";
    die(" ");
}
if($pass<10){
    echo "Password yang dimasukkan kurang dari 10 karakter!";
    $x=true;
    die(" ");
}
```

Gambar 2.4.1 baris program pengecekan password.

Dari baris program diatas kondisi *if(empty($pass))* digunakan untuk melakukan pengecekan pada variabel pass apakah password telah diinputkan atau belum. Jika belum maka pesan yang akan muncul adalah “*Password belum dimasukkan! Password harus memiliki Huruf kapital, huruf kecil, angka & karakter khusus.*”. Pendeklarasian kondisi selanjutnya adalah if($pass<10) yang digunakan untuk melakukan pengecekan pada variabel pass yang telah dimasukan apakah telah melebihi sepuluh karakter, jika password yang diinputkan belum berjumlah sepuluh karakter makan pesan yang akan muncul adalah “*Password yang dimasukkan kurang dari 10 karakter!*”.

Berikut adalah hasil tangkapan layar pengujian dengan menginputkan password yang berjumlah kurang dari sepuluh karakter.

Gambar 2.4.2 pengujian input password kurang dari 10 karakter.



Gambar 2.4.3 pesan yang ditampilkan ketika password belum berjumlah 10 karakter.

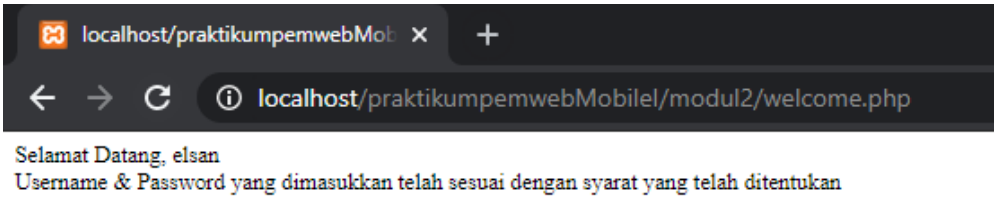
Ketika seluruh kriteria telah dibuat, selanjutnya adalah membuat pesan ketika username dan password yang dimasukan telah memenuhi kriteria. Berikut adalah hasil tangkapan layar dari baris program.

```
if($usname && $upperc && $lowerc && $angka && $specialc){
    echo "Selamat Datang, ".$username;
    echo"<br>";
    echo "Username & Password yang dimasukkan telah sesuai dengan syarat yang telah ditentukan";
    $x=false;
    die(" ");
}
```

Gambar 2.4.4 baris program selamat datang ketika telah memenuhi kriteria.

Dari baris program diatas dideklarasikan kondisi if, dengan kondisi jika variabel *usname, upperc, lowerc, angka, specialc* telah memenuhi kriteria maka akan menampilkan pesan "*selamat datang,* (username yang dimasukan)" kemudian memanggil variabel username yang telah diinputkan. Dan juga pesan "*Username & Password yang dimasukkan telah sesuai dengan syarat yang telah ditentukan*".

Berikut adalah hasil tangkapan layar yang menampilkan pesan selamat datang ketika usename dan password yang dimasukkan telah memenuhi kriteria.



Gambar 2.4.5 pesan yang ditampilkan ketika username dan password telah memenuhi kriteria.

KESIMPULAN

Kesimpulan yang didapat dari praktikum modul satu ini adalah handling pada PHP dibuat untuk melakukan validasi. Data diinputkan kedalam variabel melalui form yang kemudian akan dilakukan pengecekan apakah telah memenuhi kriteria atau belum melalui variabel yang telah dideklarasikan yang digunakan untuk menyimpan data yang diinputkan.

Pada PHP terdapat variabel superglobal, contoh nya GET dan POST. Pada metode GET data yang dikirim adalah URL yang berupa rangkaian pasangan nama dan nilai yang dipisahkan oleh ampersand(&). Sedangkan Dalam metode POST data yang dikirim kepada server sebagai paket yang dalamnya berisi komunikasi terpisah dengan skrip pemrosesan. Proses data yang dikirim tidak akan tampak pada URL.

DAFTAR PUSTAKA

[1] "PHP: Variabel Super Global | Jago Ngoding." https://jagongoding.com/web/php/menengah/variabel-super-global/ (accessed Apr. 03, 2021).

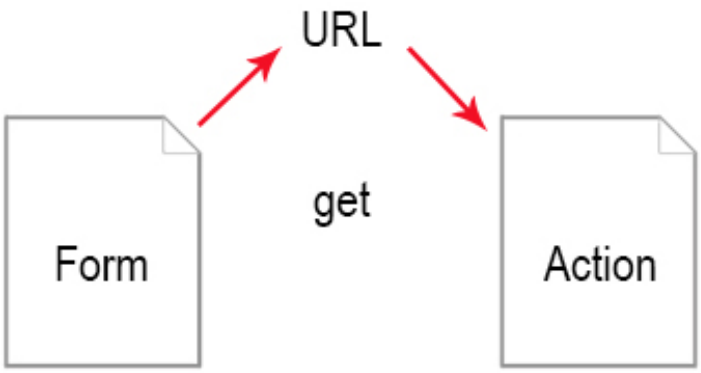
[2] "Metode GET and POST Pada PHP - KODING ASYIK." http://kodingasyik.com/metode-get-and-post-pada-php/ (accessed Apr. 03, 2021).

[3] "Validasi Form Dengan PHP - WebHozz Blog." https://www.webhozz.com/blog/validasi-form-dengan-php/ (accessed Apr. 03, 2021).

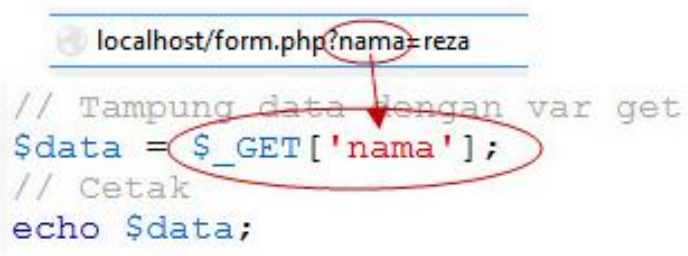
[4] K. Praktikum, "MODUL PRAKTIKUM PEMROGRAMAN WEB I Jurusan Teknik Informatika Fakultas Teknik Universitas Palangka Raya."

[5] "Tutorial PHP: Cara Membuat Validasi Form PHP (fungsi isset dan empty) | Duniailkom." https://www.duniailkom.com/tutorial-form-php-cara-membuat-validasi-form-php-fungsi-isset-dan-empty/ (accessed Apr. 03, 2021).

LAMPIRAN



Gambar 1.2.2.1 Contoh metode GET.



Gambar 1.2.2.2 Contoh syntax metode GET.

Gambar 1.2.2.3 Contoh metode POST.

```php
// Tampung data dengan var post
$data = $_POST['nama'];
// Cetak
echo $data;
```

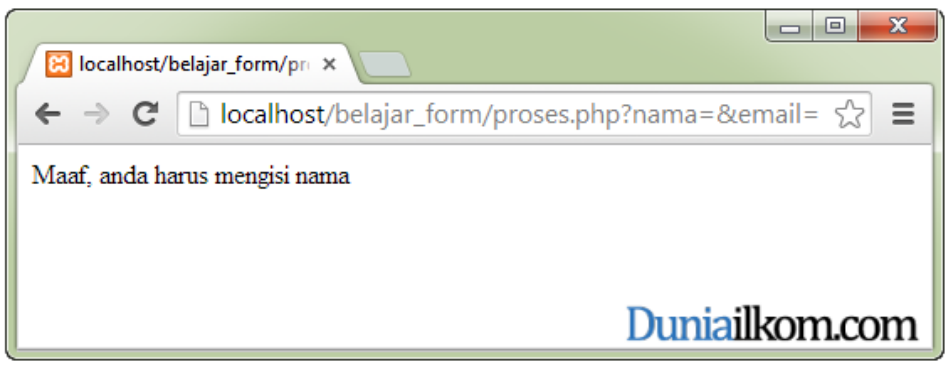
Gambar 1.2.2.4 Contoh syntax metode POST.

```php
$name = test_input($_POST["name"]);
if (!preg_match("/^[a-zA-Z ]*$/",$name)) {
        $nameErr = "Only letters and white space allowed";
}
```

Gambar 1.2.3.1 contoh validasi nama.

```php
<?php
if (isset($_GET['nama']) AND isset($_GET['email']))
{
   $nama=$_GET['nama'];
   $email=$_GET['email'];
}
else
{
   die("Maaf, anda harus mengakses halaman ini dari form.html");
}

if (!empty($nama))
{
   echo "Nama: $nama <br /> Email: $email";
}
else
{
   die("Maaf, anda harus mengisi nama");
}
?>
```

Gambar 1.2.3.2 Contoh syntax validasi.

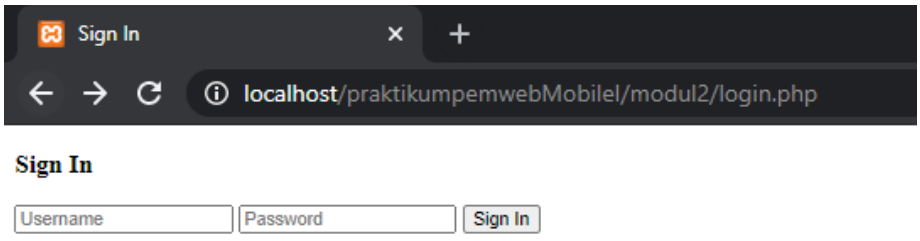


Gambar 1.2.3.3 Contoh pesan yang akan ditampilkan jika data telah divalidasi.

login.php

```php
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Sign In</title>
</head>
<body>
    <form class="boxes" method="post" action="welcome.php">
    <h3>Sign In</h3>
    <input name="uname" type="text" placeholder="Username">
    <input name="pwd" type="pwd" placeholder="Password">
    <input name="Submit" type="submit" value="Sign In">
    </form>
</body>
</html>
```

Gambar 2.1.1 baris program login form.



Gambar 2.1.2 hasil dari pembuatan login form.

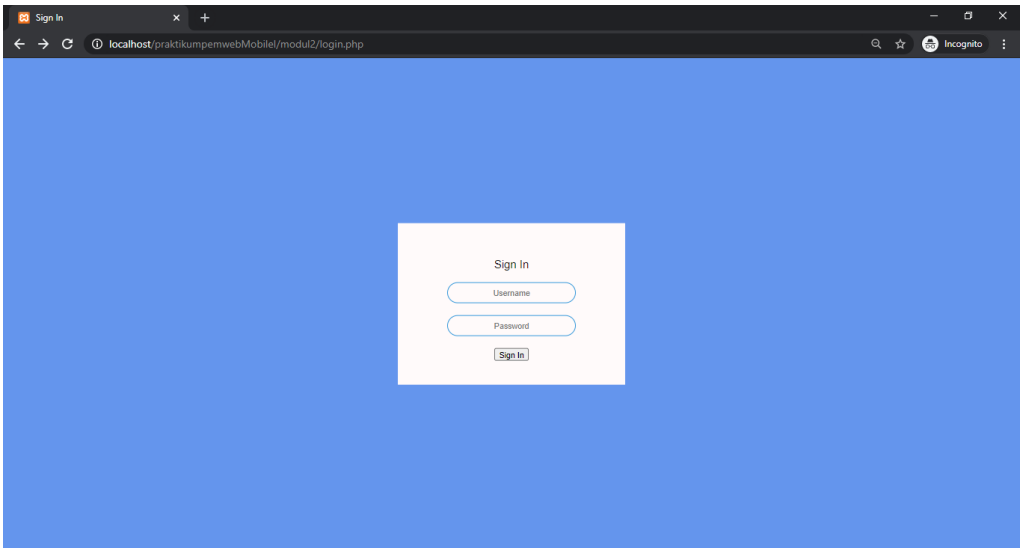
login.php

```php
<!DOCTYPE html>
<html lang="en">
<head>
    <style>
        body{
            margin: 0;
            padding: 0;
            font-family: sans-serif;
            background: #6495ED;
        }
        .boxes{
            width:300px;
            padding:40px;
            position:absolute;
            top:50%;
            left:50%;
            background:#FFFAFA;
            transform: translate(-50%,-50%);
            text-align:center;
        }
        .boxes h3{
            color:#191919;
            font-weight:300;
        }
```

Gambar 2.1.3 baris program internal CSS.

```
25          .boxes input[type="text"],.boxes input[type="pwd"]{
26          border:0;
27          background:none;
28          display:block;
29          margin:20px auto;
30          text-align:center;
31          border:2px solid #3498db;
32          padding:8px 5px;
33          width:200px;
34          outline:none;
35          border-radius:24px;
36          transition: 0.25s;
37          }
38      </style>
39      <meta charset="UTF-8">
40      <meta http-equiv="X-UA-Compatible" content="IE=edge">
41      <meta name="viewport" content="width=device-width, initial-scale=1.0">
42      <title>Sign In</title>
43  </head>
```

Gambar 2.1.4 baris program internal CSS.

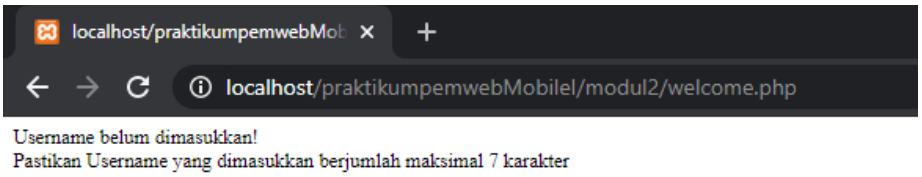


Gambar 2.1.5 login form yang telah diubah menggunakan internal CSS.

welcome.php

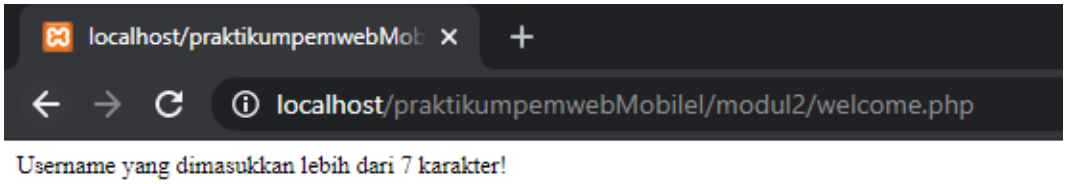
```php
<?php
    if($_SERVER["REQUEST_METHOD"] == "POST"){
        $username = $_POST['uname'];
        $password = $_POST['pwd'];
        $usname= strlen($username);
        $pass= strlen($password);
        $x=false;

        if(empty($usname)){
            echo "Username belum dimasukkan! <br>";
            echo "Pastikan Username yang dimasukkan berjumlah maksimal 7 karakter";
            die(" ");
        }
        if($usname>7){
            echo "Username yang dimasukkan lebih dari 7 karakter!";
            $x=true;
            die(" ");
        }
?>
```

Gambar 2.2.1 pembuatan handling username.



Gambar 2.2.2 pesan ketika melakukan submit tanpa menginputkan username.

Gambar 2.2.3 percobaan melakukan login dengan username lebih dari 7 karakter.

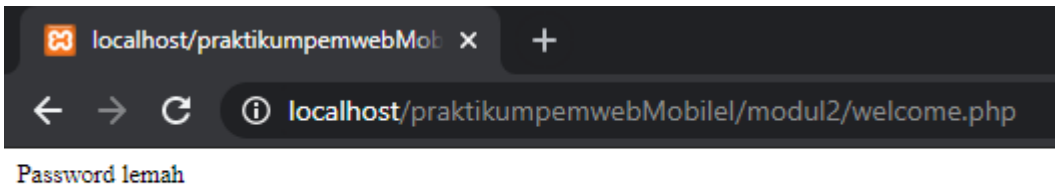


Gambar 2.2.4 pesan ketika menginputkan username lebih dari 7 karakter.

```
20      $upperc = preg_match('@[A-Z]@', $password);
21      $lowerc = preg_match('@[a-z]@', $password);
22      $angka  = preg_match('@[0-9]@', $password);
23      $specialc = preg_match('@[^\w]@', $password);
24
25      if(!$upperc || !$lowerc || !$angka || !$specialc ){
26          echo 'Password lemah';
27      }
```

Gambar 2.3.1 baris program pengecekan password.

Gambar 2.3.2 pengujian input password tanpa karakter spesial.

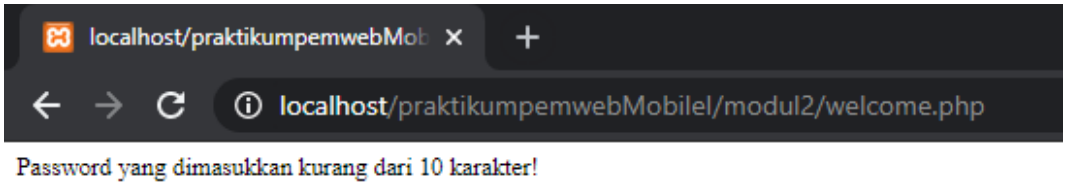


Gambar 2.3.3 pesan yang ditampilkan ketika password tidak memenuhi kriteria.

```php
if(empty($pass)){
    echo "Password belum dimasukkan!";
    echo "<br> Password harus memiliki Huruf kapital, huruf kecil, angka & karakter khusus.";
    die(" ");
}
if($pass<10){
    echo "Password yang dimasukkan kurang dari 10 karakter!";
    $x=true;
    die(" ");
}
```

Gambar 2.4.1 baris program pengecekan password.

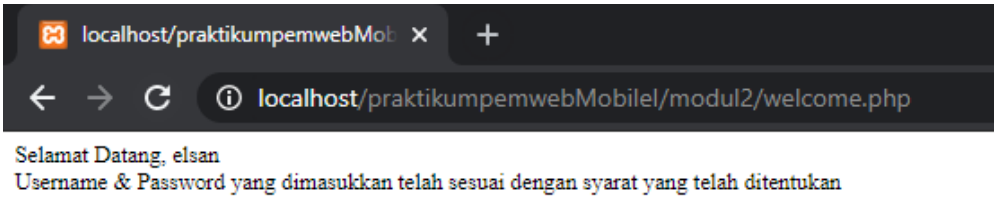
Gambar 2.4.2 pengujian input password kurang dari 10 karakter.



Gambar 2.4.3 pesan yang ditampilkan ketika password belum berjumlah 10 karakter.

```php
if($usname && $upperc && $lowerc && $angka && $specialc){
    echo "Selamat Datang, ".$username;
    echo"<br>";
    echo "Username & Password yang dimasukkan telah sesuai dengan syarat yang telah ditentukan";
    $x=false;
    die(" ");
}
```

Gambar 2.4.4 baris program selamat datang ketika telah memenuhi kriteria.



Gambar 2.4.5 pesan yang ditampilkan ketika username dan password telah memenuhi kriteria.