

**LAPORAN HASIL PRATIKUM
PEMPROGRAMAN WEB & MOBILE I**



NAMA : YUDHA ARTHA NUGRAHA
NIM : 1193030503045
KELAS : A
MODUL : II (Form Handling)

**JURUSAN TEKNIK INFORMATIKA
FAKULTAS TEKNIK
UNIVERSITAS PALANGKA RAYA
2021**

BAB I

TUJUAN DAN 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

Variabel superglobal PHP `$_GET` dan `$_POST` digunakan untuk mengumpulkan data-form. Contoh berikut menunjukkan form HTML sederhana dengan dua field input dan tombol submit:

```
<html>
    <body>
        <form action="welcome.php"
            method="post"> Name:
            <input type="text"
                name="name"><br>E-
            mail: <input type="text"
                name="email"><br>
```

Ketika user mengisi form, dan menekan tombol click, data form dikirim untuk memproses file PHP dengan nama “welcome.php”. Data form dikirimkan dengan method HTTP POST. Untuk menampilkan data yang sudah disubmit bisa dilakukan dengan mencetak data tersebut menggunakan perintah `echo`. File “welcome.php” adalah sebagai berikut:

```
<html>

    <body>

        Welcome <?php echo $_POST["name"];
        ?><br> Your email address is: <?php
        echo $_POST["email"];
```

Jika field nama diinputkan dengan Tono dan email diinputkan dengan tono@mail.com maka output yang akan tampil adalah sebagai berikut:

Welcome Budi

Your email address is tono@mail.com

Hasil yang sama juga akan tampil dengan menggunakan method get sebagai berikut:

```
<html>

    <body>

        <form action="welcome_get.php"
            method="get"> Name: <input
            type="text" name="name"><br>
            E-mail: <input type="text"
            name="email"><br>
```

dengan file “welcome_get.php” sebagai berikut:

```
<html>

    <body>

        Welcome <?php echo $_GET["name"];
        ?><br> Your email address is: <?php
        echo $_GET["email"];
```

1.2.1. GET vs. POST

GET dan POST membuat sebuah array (contoh array(kunci => nilai, kunci2 => nilai2, kunci3 => nilai3, ...)). Array ini menyimpan pasangan kunci/nilai, dimana kunci- kunci adalah nama-nama dari form control dan nilai-nilai adalah data input dari user. Method GET diakses menggunakan \$_GET dan method POST diakses menggunakan.

\$_POST. Kedua variabel ini adalah variabel superglobal, yang selalu bisa diakses, tanpa memperhatikan lingkup dan bisa diakses dari fungsi, class atau file yang berbeda tanpa harus melakukan teknik khusus. \$_GET adalah sebuah array dari variabel yang dikirimkan ke skrip melalui parameter URL. \$_POST adalah sebuah array dari variabel yang dikirimkan ke skrip melalui method HTTP POST.

Kapan sebaiknya menggunakan GET?

Informasi dikirim dari sebuah form dengan method GET bisa dilihat oleh semua orang (semua nama dan nilai variabel ditampilkan di URL). GET juga memiliki batas pada jumlah informasi yang dikirim. Batasannya adalah sekitar 2000 karakter. Namun, karena variabel ditunjukkan di URL, ia memungkinkan untuk dilakukan bookmark halaman. Dalam beberapa kasus, hal ini sangat bermanfaat. GET bisa digunakan untuk mengirimkan data yang tidak sensitif.

Ingat! GET tidak boleh digunakan untuk mengirimkan password atau informasi sensitif lainnya!

Kapan menggunakan POST?

Informasi yang dikirim dari sebuah form dengan method POST tidak bisa dilihat oleh siapapun (semua nama-nama atau nilai-nilai tertanam didalam body request HTTP) dan tidak memiliki batasan

jumlah informasi yang akan dikirim. POST juga mendukung fungsionalitas lanjutan seperti dukungan untuk input biner multi-part ketika sedang melakukan upload file ke server. Namun, karena variabel tidak ditampilkan di URL, tidak mungkin untuk dilakukan bookmark halaman (data tidak ter-bookmark). Developer lebih baik menggunakan POST untuk mengirimkan data form.

Validasi Form PHP

Pertimbangkan keamanan ketika memproses form PHP!

PHP Form Validation Example

* required field.

Name: *

E-mail: *

Website:

Comment:

Gender: ☐ Female ☐ Male *

Form HTML yang akan kita gunakan pada modul ini, mengandung bermacam- macam field input, misalnya text field yang harus diisi dan text field yang opsional, tombol pilihan (radio button), dan tombol submit. Rule atau aturan validasi untuk form diatas adalah sebagai berikut:

Field	Rule Validasi
Name	Dibutuhkan. + Harus hanya mengandung huruf dan spasi
E-mail	Dibutuhkan. + Harus mengandung sebuah alamat email yang valid dengan

	@ dan .
Website	Opsional. Jika ada, harus mengandung URL yang valid.
Comment	Opsional. Field input multi-line (text area).
Gender	Dibutuhkan. Harus memilih salah satu

Kode HTML untuk membentuk Form tersebut adalah sebagai berikut

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="na
me">E-
```

mail:

```
<input
type="text"
"
```

```
name="e
mail">
```

Website: <input type="text" name="website">

Comment: <textarea name="comment" rows="5" cols="40"></textarea>

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
```

Form Element

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

```
<form  
method="post"  
action="<?php echo  
htmlspecialchars($_  
SERVER["PHP_SE  
LF"]);?>">
```

Ketika form disubmit, data pada form dikirim dengan method “post”.

- 1.2.2. **`$_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.

Catatan Penting pada Keamanan Form PHP

Variabel `$_SERVER["PHP_SELF"]` bisa digunakan oleh hacker! Jika `PHP_SELF` digunakan pada halaman web, user bisa memasukkan skrip dengan terlebih dahulu memasukkan garis miring (`/`) kemudian beberapa perintah Cross Site Scripting (XSS) untuk dieksekusi. XSS adalah tipe kelemahan keamanan komputer yang secara tipikal ditemukan dalam aplikasi web.

Asumsikan kita memiliki halaman web

dengan nama “test_form.php”, dan form hanya kita deklarasikan sebagai berikut:

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

Kemudian user memasukkan URL pada address bar dengan alamat sebagai berikut:

[http://localhost/<nama_folder>/test_form.php/%22%3E%3Cscript%3Ealert\('hacked'\)%3C/script%3E](http://localhost/<nama_folder>/test_form.php/%22%3E%3Cscript%3Ealert('hacked')%3C/script%3E)

yang jika ditranslasikan akan menjadi:

```
<form method="post"  
action="test_form.php/"><script>alert('hacked')</script>
```

Kode ini menambah tag script dan perintah alert atau peringatan, ketika halaman dibuka, kode javascript tersebut akan dieksekusi, maka user akan melihat kotak peringatan dengan tulisan “hacked”.

Berhati-hatilah dengan kemungkinan penambahan kode javascript pada tag <script>!

Hacker bisa mengarahkan user ke file pada server yang lain, dan file itu bisa mengandung kode yang bisa merubah variabel global atau melakukan submit

Bagaimana menghindari penyalahgunaan \$_SERVER[“PHP_SELF”]?

Caranya adalah dengan menggunakan fungsi htmlspecialchars(). Fungsi tersebut akan mengkonversikan karakter khusus ke entitas HTML. Ketika user memasukkan URL dengan tag script seperti contoh sebelumnya, maka akan ditranslasikan sebagai berikut:

```
<form method="post"  
action="test_form.php/&quot;&gt;&lt;script&gt;  
alert('hacked')&lt;/script&gt;">
```


dengan cara ini, percobaan penyalahgunaan akan gagal.

Memvalidasi data Form dengan PHP

Hal pertama yang akan kita lakukan adalah memasukkan semua variabel melalui fungsi htmlspecialchars(). Kemudian ada juga dua hal ketika user melakukan submit form:

1. Membuang karakter-karakter yang tidak dibutuhkan (seperti spasi extra, tab extra, dan baris baru yang ekstra) dari data input user (dengan fungsi trim()).
2. Membuang backslash (\) satu garis miring dari data input user (dengan fungsi stripslashes()).

Langkah berikutnya adalah membuat fungsi yang akan melakukan pemeriksaan kebenaran data yang diinputkan oleh user. Contohnya adalah sebagai berikut:

```
<?php

// define variables and set to empty values
$name = $email = $gender = $comment = $website = "";

if ($_SERVER["REQUEST_METHOD"] == "POST") {
    $name = test_input($_POST["name"]);
    $email = test_input($_POST["email"]);
    $website = test_input($_POST["website"]);
    $comment = test_input($_POST["comment"]);
    $gender = test_input($_POST["gender"]);
}

function
test_input($da
    ta) {
```

```

        $data =
        trim($data);

        $data = stripslashes($data);
        $data =
        htmlspecialchars(
        $data);return
        $data;

    }
?>

```

Ingat bahwa pada permulaan skrip, adalah pemeriksaan apakah form sudah disubmit menggunakan `$_SERVER["REQUEST_METHOD"]`. Jika `REQUEST_METHOD` adalah POST, maka form telah disubmit dan seharusnya tervalidasi. Jika belum tersubmit, lewati langkah validasi dan tampilkan form kosong. Namun pada contoh diatas semua field input adalah opsional. Skrip bekerja baik bahkan jika user tidak melakukan entri data.

Field yang Dibutuhkan

Kode program berikut terdapat tambahan variabel baru yaitu: `$nameErr`, `$emailErr`, `$genderErr`. Variabel-variabel error ini akan menangani pesan error untuk field yang dibutuhkan. Percabangan dengan `if else` juga akan ditambahkan untuk setiap variabel `$_POST`. Fungsinya untuk memeriksa apakah variabel `$_POST` kosong, hal ini dilakukan dengan menggunakan fungsi `empty()`. Jika kosong, maka pesan error disimpan dalam variabel error yang berbeda, dan jika tidak kosong, ia akan mengirim data input user melalui fungsi `test_input()`:

```
<?php
```

```
// define variables and set to empty values

$nameErr = $emailErr = $genderErr = $websiteErr = "";
$name = $email = $gender = $comment = $website = "";

if ($_SERVER["REQUEST_METHOD"] == "POST")
{
    if (empty($_POST["name"])) {
        $nameErr = "Name is required";
    } else {
        $name = test_input($_POST["name"]);
    }

    if (empty($_POST["email"])) {
        $emailErr = "Email is required";
    } else {
        $email = test_input($_POST["email"]);
    }

    if (empty($_POST["website"])) {
        $website = "";
    }
}
```

```
    } else {  
        $website =  
        test_input($_POST["website"]);  
    }  
  
    if (empty($_POST["comment"])) {  
        $comment = "";  
    } else {  
        $comment =  
    }  
?>
```

Setelah kode diatas ditambahkan, beberapa skrip ditambahkan pada setiap field yang dibutuhkan pada form, fungsinya untuk menampilkan pesan error jika field yang dibutuhkan tidak diisi. Form HTMLnya adalah sebagai berikut:

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

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

```
<span class="error">* <?php echo
```

```
$nameErr;?></s
```

```
pan><br><br>E-
```

```
mail:
```

```
<input type="text" name="email">
```

```
<span class="error">* <?php echo $emailErr;?></span>
```

```
<
```

```
b
```

```
r
```

```
>
```

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

```
<span class="error">* <?php echo $genderErr;?></span>
```

```
<br><br>
```

```
<input type="submit" name="submit" value="Submit">
```

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";
```

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

Validasi Email

Cara paling mudah dan paling aman untuk memeriksa apakah sebuah alamat email memiliki pola yang sesuai adalah dengan menggunakan fungsi `filter_var()`. Kode dibawah memeriksa apakah alamat email yang dimasukkan menggunakan pola yang sesuai atau tidak, jika tidak, maka pesan error akan disimpan kedalam variabel

`$emailErr`:

```
$email = test_input($_POST["email"]);  
if (!filter_var($email, FILTER_VALIDATE_EMAIL))  
    { $emailErr = "Invalid email format";
```

Validasi URL

Kode program berikut menunjukkan cara untuk memeriksa apakah sintaks alamat URL valid atau tidak. Ekspresi reguler ini mengizinkan keberadaan tanda pisah pada URL. Jika sintaks alamat URL tidak valid, maka pesan error akan disimpan kedalam variabel `$websiteErr`:

```

$website = test_input($_POST["website"]);
if (!preg_match("/^b(?:(:https?|ftp):\\\/|www\\.)[-a-z0-9+&@#\/%?=_!:,.;]*[-a-z0-9+&@#\/%?=_!:,.;]/i",$website)) {
    $websiteErr = "Invalid URL";
}

```

Biasanya, jika user salah menginputkan nilai, maka halaman yang tampil adalah halaman yang sama dengan field yang sudah terisi dengan nilai field yang sudah diinput sebelumnya. Untuk menunjukkan nilai dalam field input setelah user menekan tombol submit, ada beberapa skrip PHP yang perlu ditambahkan didalam atribut value pada field input name, email, dan website. Khusus untuk field textarea, akan skrip tersebut akan ditambahkan antara tag <textarea> dan tag </textarea>. Skrip yang singkat akan mengeluarkan nilai dari variabel \$name, \$email, \$website dan \$comment. Untuk radio button atau tombol radio, akan ditambahkan kode yang membuat salah satu pilihan terpilih.

```

Name: <input type="text" name="name" value="<?php echo
$name;?>">E-mail: <input type="text" name="email"
value="<?php echo $email;?>">

Website: <input type="text" name="website" value="<?php echo $website;?>">

Comment: <textarea name="comment" rows="5" cols="40"><?php echo
$comment;? ></textarea>

Gender:

```

TUGAS

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

1. username yang diinputkan tidak boleh lebih dari tujuh karakter.
2. password yang diinputkan harus terdiri dari huruf kapital, huruf kecil, angkadan karakter khusus.
3. Jumlah karakter password tidak boleh kurang dari sepuluh karakter.

BAB II

PEMBAHASAN



```
1
2 <!DOCTYPE html>
3 <html>
4 <head>
5   <title>Modul 2</title>
6 </head>
7 <body>
8
9   <form
10     action="output.php" method="post">
11     <ul>
12       <li>
13         <label for="user">Username<br></label>
14         <input type="text" name="user" id="user">
15       </li>
16       <li>
17         <label for="pass">Password<br></label>
18         <input type="Password" name="pass" id="pass">
19       </li>
20       <li>
21         <button type="submit"> submit </button>
22       </li>
23     </ul>
24   </form>
25
26
27 </body>
28 </html>
```

Gambar 2.1 source modul2_193030503045

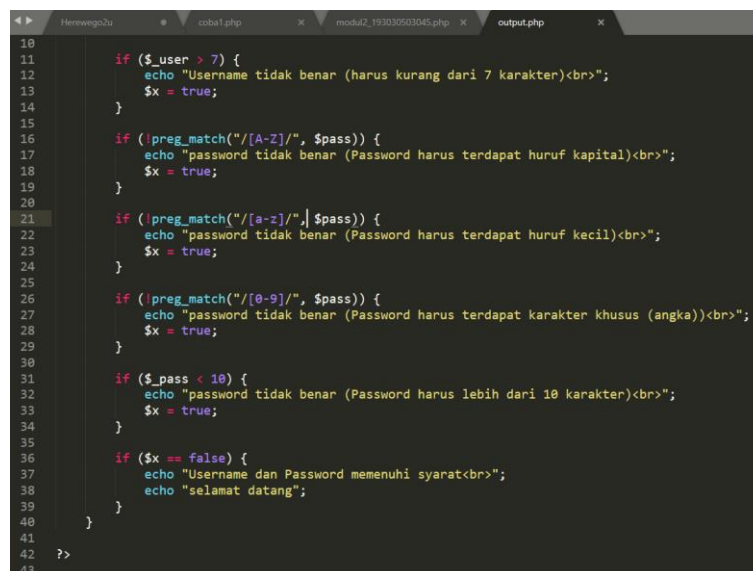
Pada source code modul2_193030503045.php berisikan coding yang digunakan untuk tampilan login dan untuk hyperlink ke source code output.php , pada baris 2 digunakan untuk menggunakan html versi ke 5, pada baris ke 5 digunakan untuk memberi nama halaman web, pada baris ke 9-10 digunakan untuk mengatur tujuan hyperlink dari form(web login) dengan method post, pada baris 13-14 digunakan untuk membuat textfield username, pada baris ke 17-18 digunakan untuk membuat textfield yang bertipe password sehingga tidak ditampilkan karakter pada text fieldnya namun hanya berupa titik, pada baris 21 digunakan untuk melakukan submit agar hyperlink dapat dilakukan, pada baris 27-28 merupakan tanda akhir dari program.

A screenshot of a code editor with a dark theme. The editor has four tabs at the top: 'Herewego2u', 'coba1.php', 'modul2_193030503045.php', and 'output.php'. The 'modul2_193030503045.php' tab is active. The code is as follows:

```
1 <?php
2
3
4 if ($_SERVER["REQUEST_METHOD"] == "POST") {
5     $user = $_REQUEST["user"];
6     $pass = $_POST["pass"];
7     $user = strlen($user);
8     $pass = strlen($pass);
9     $x = false;
10
```

Gambar 2.2 source code output variable user&pass

pada bagian ini digunakan untuk mengatur user dan pass dari source code modul2_193030503045 agar dapat mengambil data dari output.php.

A screenshot of a code editor with a dark theme. The editor has four tabs at the top: 'Herewego2u', 'coba1.php', 'modul2_193030503045.php', and 'output.php'. The 'modul2_193030503045.php' tab is active. The code is as follows:

```
10
11 if ($user > 7) {
12     echo "Username tidak benar (harus kurang dari 7 karakter)<br>";
13     $x = true;
14 }
15
16 if (preg_match("/[A-Z]/", $pass)) {
17     echo "password tidak benar (Password harus terdapat huruf kapital)<br>";
18     $x = true;
19 }
20
21 if (preg_match("/[a-z]/", $pass)) {
22     echo "password tidak benar (Password harus terdapat huruf kecil)<br>";
23     $x = true;
24 }
25
26 if (preg_match("/[0-9]/", $pass)) {
27     echo "password tidak benar (Password harus terdapat karakter khusus (angka))<br>";
28     $x = true;
29 }
30
31 if ($pass < 10) {
32     echo "password tidak benar (Password harus lebih dari 10 karakter)<br>";
33     $x = true;
34 }
35
36 if ($x == false) {
37     echo "Username dan Password memenuhi syarat<br>";
38     echo "selamat datang";
39 }
40
41
42 ?>
43
```

Gambar 2.3 source code if tertentu

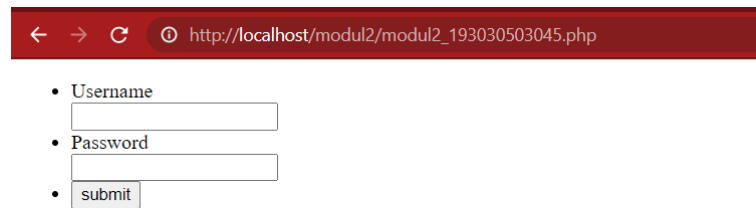
Pada bagian ini digunakan untuk memenuhi syarat syarat yang diinginkan dengan fungsi if, pada baris 11-13 digunakan untuk mengatur agar jika usernamemelebihi 8 karakter akan dilemparkan ke echo, pada baris ke 16-17 digunakan untuk mengatur password dimana pada password harus digunakan huruf capital jika tidak akan di lempar ke echo baris17, pada baris ke 21-23 memiliki fungsi yang sama seperti baris 16-18 namun dengan syarat huruf kecil atau bukan capital, pada baris 26-28 digunakan untuk memberi peringatan jika di password tidak ada angka dikarenakan pada password harus ada angka agar password dapat diterima, pada baris ke 31-34 digunkanan untuk memberi peringatan jika password kurang dari 10 karakter, pada baris 36-39 digunakan untuk jika semua syarat terpenuhi dan berhasil masuk.

A screenshot of a code editor with a dark background. The editor has several tabs at the top: 'Herewego2u', 'coba1.php', 'modul2_193030503045.php', and 'output.php'. The 'modul2_193030503045.php' tab is active. The code is written in HTML and is as follows:

```
43
44
45 <!DOCTYPE html>
46 <html>
47 <head>
48   <title></title>
49 </head>
50 <body>
51   <br>
52   <a href="modul2_193030503045.php">KEMBALI KE MENU LOGIN
53 </body>
54 </html>
```

Gambar 2.4 hyperlink modul2_193030500345

pada bagian ini digunakan untuk hyperlink ke source code modul2_193030503045.php, pada baris ke 52 digunakan fungsi a href dimana merupakan hyperlink, pada hyperlink ini di kasih kata kata kembali ke login.



A screenshot of a web browser window. The address bar shows the URL `http://localhost/modul2/modul2_193030503045.php`. Below the address bar, there is a list of form fields:

- Username:
- Password:
- submit:

Gambar 2.5 tampilan login

Di atas merupakan tampilan login dari source code yang telah di buat di dalam web login dapat di masukan username dan password



A screenshot of a web browser window. The address bar shows the URL `http://localhost/modul2/output.php`. Below the address bar, the text reads:

Username dan Password memenuhi syarat
selamat datang
[KEMBALI KE MENU LOGIN](#)

Gambar 2.6 tampilan web jika sesuai ketentuan

Pada bagian ini merupakan tampilan website jika saat memasukan username dan password tepat seperti yang diminta



Gambar 2.7 tampilan web jika terjadi kesalahan

Pada bagian halaman web ini digunakan untuk menampilkan kesalahan kesalahan ketika input username dan password

BAB III

KESIMPULAN

Superglobals variabel pada PHP yaitu variabel \$ _GET dan variabel \$ _POST digunakan untuk mengumpulkan data formulir. Kita dapat membuat dan menggunakan formulir di PHP. Untuk mendapatkan data formulir, kita perlu menggunakan variabel PHP superglobals \$ _GET dan \$ _POST.

Permintaan formulir atau form dapat berupa get atau post. Untuk mengambil data dari get request, kita perlu menggunakan \$ _GET, untuk post request harus menggunakan \$ _POST. Situs web saat ini telah menyediakan banyak fungsionalitas yang dapat digunakan untuk menyimpan, memperbarui, mengambil, dan menghapus data dalam database.

Pada laporan di atas merupakan salah satu contoh dari superglobals yang telah ada di php

DAFTAR PUSTAKA

Bagus Dharma Iswara. Form Handling PHP: Contoh dan Source Code.

DosenIT.com. Published October 28, 2020. Accessed April 4, 2021.

<https://dosenit.com/php/form-handling-php-contoh-dan-source-code>

SeventhQueen. Lab-Informatika | PHP Form Handling. Lab-Informatika.


Published 2012. Accessed April 4, 2021. [https://www.lab-](https://www.lab-informatika.com/php-form-handling)

[informatika.com/php-form-handling](https://www.lab-informatika.com/php-form-handling)

Modul Praktikum Pemrograman Web & Mobile I. Jurusan Teknik Informatika.

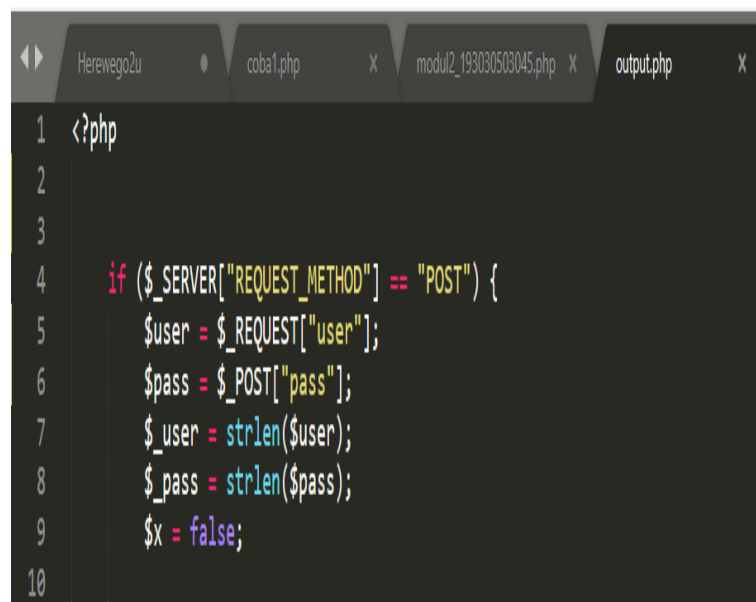
Fakultas Teknik. Universitas Palangka Raya. 2021.

LAMPIRAN



```
1
2 <!DOCTYPE html>
3 <html>
4 <head>
5   <title>Modul 2</title>
6 </head>
7 <body>
8
9   <form
10     action="output.php" method="post">
11     <ul>
12       <li>
13         <label for="user">Username<br></label>
14         <input type="text" name="user" id="user">
15       </li>
16       <li>
17         <label for="pass">Password<br></label>
18         <input type="Password" name="pass" id="pass">
19       </li>
20       <li>
21         <button type="submit"> submit </button>
22       </li>
23     </ul>
24   </form>
25
26
27 </body>
28 </html>
```

Gambar 2.1 source modul2_193030503045



```
1 <?php
2
3
4 if ($_SERVER["REQUEST_METHOD"] == "POST") {
5   $user = $_REQUEST["user"];
6   $pass = $_POST["pass"];
7   $user = strlen($user);
8   $pass = strlen($pass);
9   $x = false;
10
```

Gambar 2.2 source code output variable user&pass


```

10
11     if ($_user > 7) {
12         echo "Username tidak benar (harus kurang dari 7 karakter)<br>";
13         $x = true;
14     }
15
16     if (!preg_match("/[A-Z]/", $pass)) {
17         echo "password tidak benar (Password harus terdapat huruf kapital)<br>";
18         $x = true;
19     }
20
21     if (!preg_match("/[a-z]/", $pass)) {
22         echo "password tidak benar (Password harus terdapat huruf kecil)<br>";
23         $x = true;
24     }
25
26     if (!preg_match("/[0-9]/", $pass)) {
27         echo "password tidak benar (Password harus terdapat karakter khusus (angka))<br>";
28         $x = true;
29     }
30
31     if ($_pass < 10) {
32         echo "password tidak benar (Password harus lebih dari 10 karakter)<br>";
33         $x = true;
34     }
35
36     if ($x == false) {
37         echo "Username dan Password memenuhi syarat<br>";
38         echo "selamat datang";
39     }
40 }
41
42 ?>
43

```

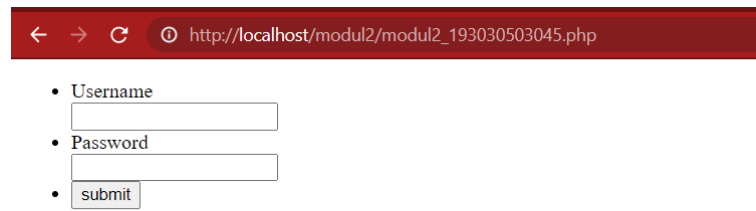
Gambar 2.3 source code if tertentu

```

43
44
45 <!DOCTYPE html>
46 <html>
47 <head>
48     <title></title>
49 </head>
50 <body>
51     <br>
52     <a href="modul2_193030503045.php">KEMBALI KE MENU LOGIN
53 </body>
54 </html>

```

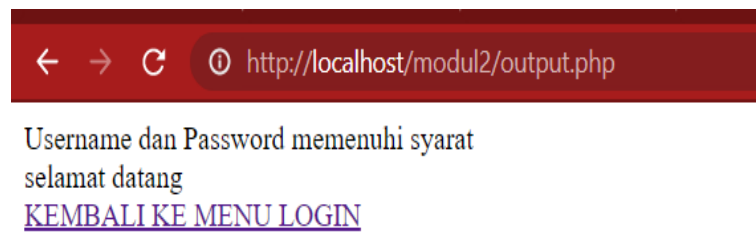
Gambar 2.4 hyperlink modul2_193030500345



A screenshot of a web browser window. The address bar shows the URL `http://localhost/modul2/modul2_193030503045.php`. Below the address bar, there is a list of form elements:

- Username: A text input field.
- Password: A text input field.
- submit: A button.

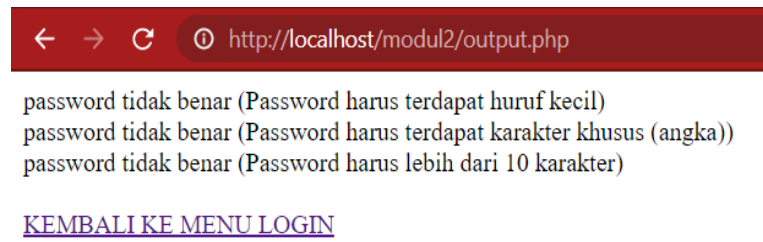
Gambar 2.5 tampilan login



A screenshot of a web browser window. The address bar shows the URL `http://localhost/modul2/output.php`. Below the address bar, the text reads:

Username dan Password memenuhi syarat
selamat datang
[KEMBALI KE MENU LOGIN](#)

Gambar 2.6 tampilan web jika sesuai ketentuan



Gambar 2.7 tampilan web jika terjadi kesalahan