

**LAPORAN PRAKTIKUM**  
**PEMROGRAMAN WEB & MOBILE I**



**NAMA : RANI MELIYANA PUTRI**

**NIM : 11191062**

**KELAS : C**

**MODUL II**

**JURUSAN TEKNIK INFORMATIKA**

**FAKULTAS TEKNIK**

**UNIVERSITAS PALANGKA RAYA**

**2021**

## BAB I

### TUJUAN DAN LANDASAN TEORI

#### 1. Tujuan Praktikum

**1.1.** Mahasiswa mampu membuat handling yang mampu mengolah data dari form HTML.

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

#### 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>
      <input type="submit">
    </form>
  </body>
</html>
```

**Gambar 1.2.1**

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"];
    ?> </body>
  </html>

```

**Gambar 1.2.2**

Jika field nama diinputkan dengan Tono dan email diinputkan dengan [tono@mail.com](mailto: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>
      <input type="submit">
    </form>
  </body>
</html>

```

**Gambar 1.2.3**

dengan file “welcome\_get.php” sebagai berikut:

```

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

```

**Gambar 1.2.4**

### 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 \*

**Gambar 1.2.5**

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="name">

E-mail: <input type="text" name="email">

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_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 &lt; dan &gt;. 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
```

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 form pada alamat web yang berbeda untuk mencuri data user.

### **Bagaimana cara 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($data) {
    $data = trim($data);
    $data = stripslashes($data);
    $data = htmlspecialchars($data);
    return $data;
}

?>
```

**Gambar 1.2.6**

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 = "";
    }

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

    if (empty($_POST["gender"])) {
        $genderErr = "Gender is required";
    } else {
        $gender = test_input($_POST["gender"]);
    }
}
?>
```

**Gambar 1.2.7**

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;?></span> <br><br>  
    E-mail:  
    <input type="text" name="email">  
    <span class="error">* <?php echo $emailErr;?></span>  
    <br><br>  
    Website:  
    <input type="text" name="website">  
    <span class="error"><?php echo $websiteErr;?></span>  
    <br><br>  
    Comment: <textarea name="comment" rows="5" cols="40"></textarea>  
    <br><br>  
    Gender:  
    <input type="radio" name="gender" value="female">Female  
  
    <input type="radio" name="gender" value="male">Male  
    <span class="error">* <?php echo $genderErr;?></span>  
    <br><br>  
    <input type="submit" name="submit" value="Submit">  
  
</form>
```

**Gambar 1.2.8**

### ***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 1.2.9**

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";  
}
```

**Gambar 1.2.10**

### ***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):VV|www.)[-a-z0-9+&@#V%?=-_!:.~]*[-a-z0-9+&@#V %=-_~_]/i",$website)) {  
    $websiteErr = "Invalid URL";  
}
```

**Gambar 1.2.11**

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:
<input type="radio" name="gender"
<?php if (isset($gender) && $gender=="female") echo
"checked";?> value="female">Female
<input type="radio" name="gender"
<?php if (isset($gender) && $gender=="male") echo
"checked";?> value="male">Male
```

**Gambar 1.2.12**

## BAB II

### PEMBAHASAN

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, angka dan karakter khusus.
3. Jumlah karakter password tidak boleh kurang dari sepuluh karakter.

#### Source Code

1	<html>
2	<head>
3	<title>Modul 2 Form Handling</title>
4	</head>
5	<body>
6	<h3>Form Handling</h3>
7	<?php
8	// Membuat variabel
9	\$username="";
10	\$pass = "";
11	\$usernameErr = "";
12	\$passErr = "";
13	\$username_valid = true;
14	\$username_valid_msg = "";
15	\$valid_jenis_pass = true;
16	\$valid_jenis_pass_msg = "";
17	
18	// Cek form sudah di klik submit/belum
19	if(isset(\$_POST['submit'])){
20	\$username = trim(\$_POST['username']);
21	\$pass = trim(\$_POST['pass']);
22	

```

23      // Cek input kosong
24      if(empty($username)){
25          $usernameErr = "Username masih kosong.<br>";
26      }
27      if(empty($pass)){
28          $passErr = "Password masih kosong.<br>";
29      }
30
31      // validasi username
32      if(!preg_match("/^[a-zA-Z][a-zA-Z0-9-
33      _\.\]{0,6}$/",$username)) {
34          $username_valid = false;
35          $username_valid_msg = "Username yang dimasukkan
36          tidak boleh lebih dari tujuh karakter. <br>";
37      }
38      // validasi password
39      if (!preg_match("/(?!.*\d)(?!.*[a-z])(?!.*[A-
40      Z])(?!.*[\^\'\$%&*()]{@:\'\#~?><>,;\@|\|-=\-_\+|-
41      -\\`\\})).{10,}/",$pass)){
42          $valid_jenis_pass = false;
43          $valid_jenis_pass_msg = "password yang diinputkan
44          harus terdiri dari huruf kapital, huruf kecil, angka,
45          karakter khusus dan tidak boleh kurang dari 10
46          karakter.<br>";
47      }
48  }
49  }
50  ?>
51  <form action="form-handling.php" method="post">
52      Username :
53      <input type="text" name="username" value="<?php echo
54      $username; ?>"><br>
55      <?php echo $usernameErr.$username_valid_msg; ?>

```

48	 
49	Password :
50	<input type="password" name="pass" id="myInput" value="<?php echo \$pass; ?>"> 
51	<?php echo \$passErr.\$valid_jenis_pass_msg; ?> 
52	<input type="checkbox" onclick="myFunction()">Show Password  
53	<input type="submit" name="submit" value="Submit">
54	</form>
55	<?php
56	// Cek semua input sudah diisi apa belum
57	if(!empty(\$username) and !empty(\$pass) and \$username_valid and \$valid_jenis_pass){
58	echo "Selamat semua input sudah diisi dengan benar. ";
59	}
60	?>
61	</body>
62	<script>
63	function myFunction() {
64	var x = document.getElementById("myInput");
65	if (x.type === "password") {
66	x.type = "text";
67	} else {
68	x.type = "password";
69	}
70	}
71	</script>
72	</html>

## Penjelasan

Sebelum membuat validasi pada username dan password, maka buat dulu variabel yang akan digunakan seperti berikut. Karena hanya diperlukan form yang

berisi username dan password maka variabel yang dibuat hanya yang berkaitan dengan username dan password.

```
// Membuat variabel
$username="";
$pass = "";
$usernameErr = "";
$passErr = "";
$username_valid = true;
$username_valid_msg = "";
$valid_jenis_pass = true;
$valid_jenis_pass_msg = "";
```

Untuk mengecek form yang sudah diklik submit atau belum maka menggunakan method \$\_POST seperti kode di bawah. Untuk membuang karakter-karakter yang tidak dibutuhkan (seperti spasi extra, tab extra, dan baris baru yang ekstra) dari data input user maka menggunakan fungsi trim(). Penggunaan method \$\_POST dikarenakan form yang dibuat adalah form untuk login, yang dimana informasi yang dimasukkan bersifat rahasia dan 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.

```
// Cek form sudah di klik submit/belum
if(isset($_POST['submit'])){
    $username = trim($_POST['username']);
    $pass = trim($_POST['pass']);
```

Untuk melakukan pengecekan dari input yang masih kosong maka menggunakan fungsi empty() seperti pada kode berikut yang dimana jika input masih kosong maka akan memunculkan kalimat berupa “Username masih kosong” atau “Password masih kosong”.

```
// Cek input kosong
if(empty($username)){
    $usernameErr = "Username masih kosong.<br>";
}
if(empty($pass)){
    $passErr = "Password masih kosong.<br>";
}
```



Selanjutnya, melakukan validasi untuk username seperti pada soal nomor 1 yang dimana username yang diinputkan tidak boleh lebih dari 7 karakter atau maksimal berjumlah 7 karakter.

```
if(!preg_match("/^[a-zA-Z][a-zA-Z0-9-_.]{0,6}$/", $username)) {  
    $username_valid = false;  
    $username_valid_msg = "Username yang dimasukkan tidak boleh lebih dari tujuh karakter. <br>";  
}
```

Disini menggunakan fungsi `preg_match()` yang di dalamnya berisi huruf kapital, huruf kecil, angka, dan karakter serta maksimal input adalah 7 karakter. Disini dituliskan `{0,6}` karena perhitungan index dimulai dari 0 bukan 1, sehingga jika ingin maksimal karakter nya 7, maka dituliskan 6. Di dalam fungsi `preg_match()` juga memanggil variabel `$username` yang berarti fungsi tersebut akan dijalankan pada variabel `$username`. Selanjutnya, di dalam kondisi tersebut, terdapat 2 variabel yaitu `$username_valid` dan `$username_valid_msg` yang dimana kondisi tersebut akan dijalankan jika username yang diinputkan oleh pengguna tidak sesuai dengan ketentuan maka akan menampilkan kalimat "Username yang dimasukkan tidak boleh lebih dari tujuh karakter."

Berikutnya, untuk soal nomor 2 dan 3 untuk melakukan validasi pada password yang dimana password yang diinputkan harus terdiri dari huruf kapital, huruf kecil, angka dan karakter khusus serta jumlah karakter password tidak boleh kurang dari 10 karakter atau minimal 10 karakter.

```
// validasi password  
if (!preg_match("/(?!.*\d)(?!.*[a-z])(?!.*[A-Z])(?!.*[\^\'£$%^&*()]{0,1}){0,1}[@:\'#\~?><>,;\@|\|-=\_+\\-\\`\\`\\`]{10,}/", $pass)){  
    $valid_jenis_pass = false;  
    $valid_jenis_pass_msg = "password yang diinputkan harus terdiri dari huruf kapital, huruf kecil, angka, karakter khusus dan tidak boleh kurang dari 10 karakter.<br>";  
}
```

Disini juga menggunakan fungsi `preg_match()` yang di dalamnya berisi huruf kapital, huruf kecil, angka, dan karakter serta minimal input adalah 10 karakter. Disini dituliskan `{10,}` karena minimal karakter nya adalah 10. Dalam

fungsi `preg_match()` ini juga memanggil variabel `$pass` yang berarti fungsi tersebut akan dijalankan pada variabel `$pass`. Selanjutnya, di dalam kondisi tersebut, terdapat 2 variabel yaitu `$valid_jenis_pass` dan `$valid_jenis_pass_msg` yang dimana kondisi tersebut akan dijalankan jika username yang diinputkan oleh pengguna tidak sesuai dengan ketentuan maka akan menampilkan kalimat “password yang diinputkan harus terdiri dari huruf kapital, huruf kecil, angka dan karakter khusus serta jumlah karakter password tidak boleh kurang dari 10 karakter atau minimal 10 karakter.”

Selanjutnya, pembuatan form html nya dimana form action nya adalah nama file itu sendiri dengan method post. Lalu untuk form username digunakan value berupa pemanggilan variabel `$username` serta `$usernameErr` dan `$username_valid_msg` untuk memvalidasi username yang diinputkan oleh pengguna apakah sudah sesuai atau belum. Dan untuk form password juga digunakan value berupa pemanggilan variabel `$pass` serta `$passErr` dan `$valid_jenis_pass_msg` untuk memvalidasi password yang diinputkan oleh pengguna apakah sesuai atau belum.

```
<form action="form-handling.php" method="post">
    Username :
    <input type="text" name="username" value="<?php echo $username
; ?>"><br>
    <?php echo $usernameErr.$username_valid_msg; ?>
    <br>
    Password :
    <input type="password" name="pass" id="myInput" value="<?php e
cho $pass; ?>"><br>
    <?php echo $passErr.$valid_jenis_pass_msg; ?><br>
    <input type="checkbox" onclick="myFunction()">Show Password<br>
><br>
    <input type="submit" name="submit" value="Submit">
</form>
```

Pada kode di atas juga digunakan javascript `myFunction()` yang digunakan untuk menampilkan password yang dimasukan pengguna jika diinginkan. Berikut isi dari javascript tersebut.

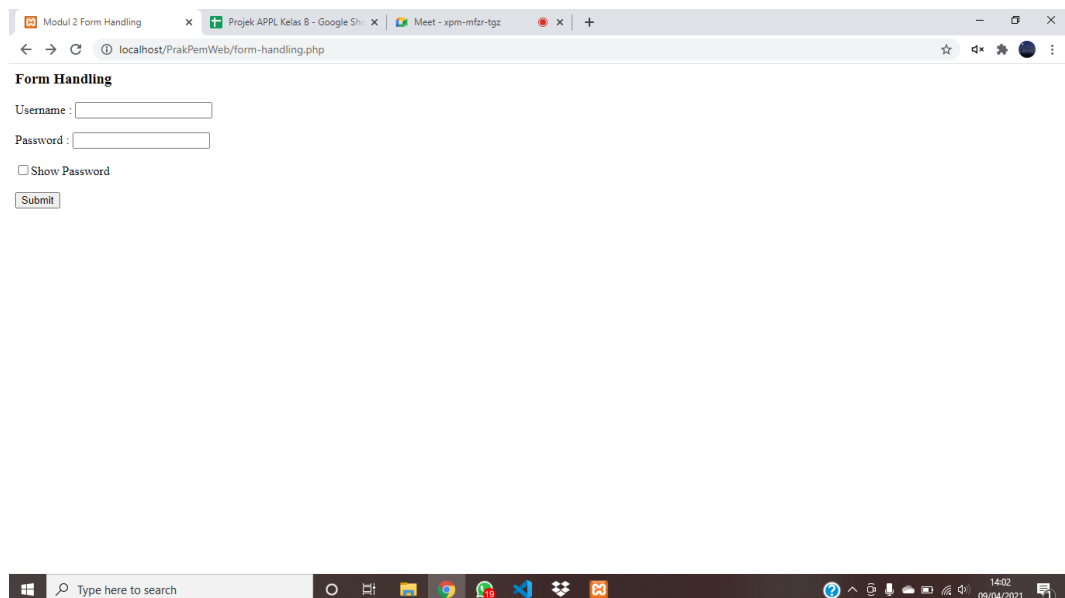
```
<script>
```

```
function myFunction() {
    var x = document.getElementById("myInput");
    if (x.type === "password") {
        x.type = "text";
    } else {
        x.type = "password";
    }
}
</script>
```

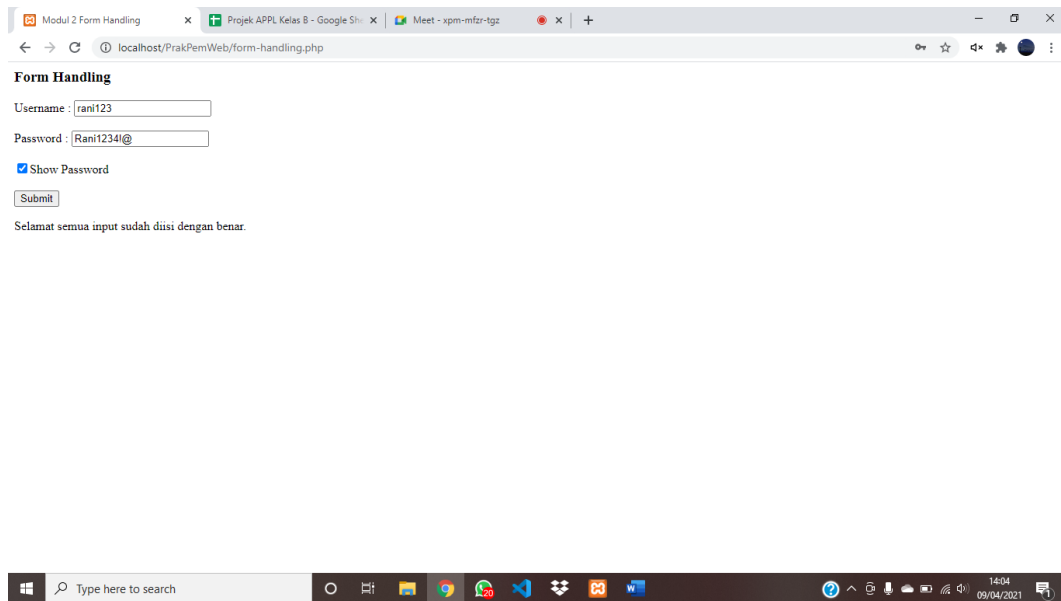
Dan terakhir, untuk melakukan pengecekan apakah semua input sudah diisi atau belum maka digunakan php sebagai berikut. Yang dimana jika semua input sudah diisi dan sesuai dengan yang diminta maka akan menampilkan kalimat “Selamat semua input sudah diisi dengan benar.”

```
<?php
    // Cek semua input sudah diisi apa belum
    if(!empty($username) and !empty($pass) and $username_valid and
    $valid_jenis_pass){
        echo "Selamat semua input sudah diisi dengan benar.<br>";
    }
?>
```

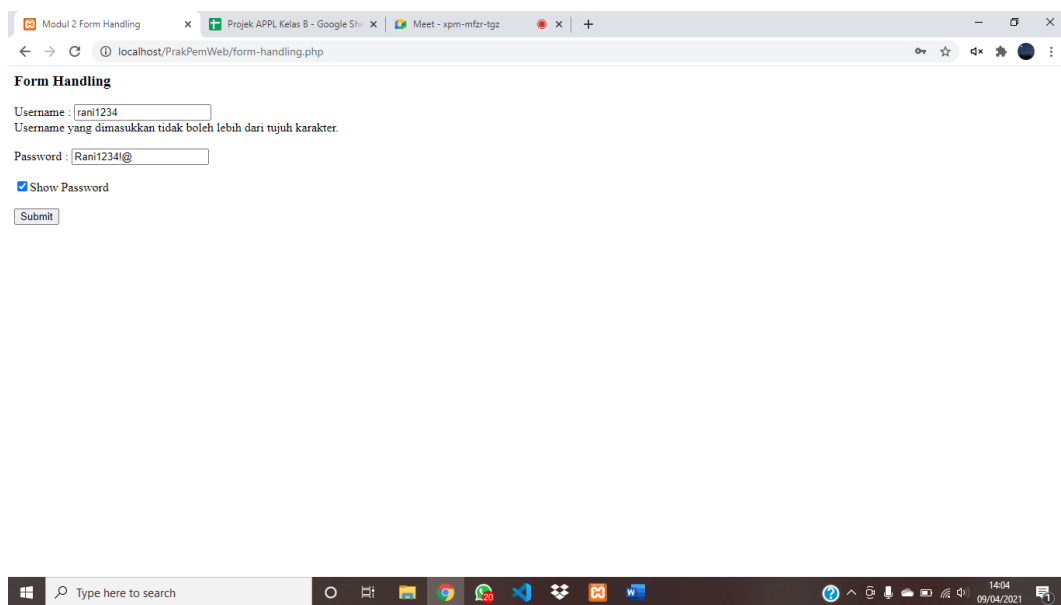
Berikut hasil program yang sudah dibuat.



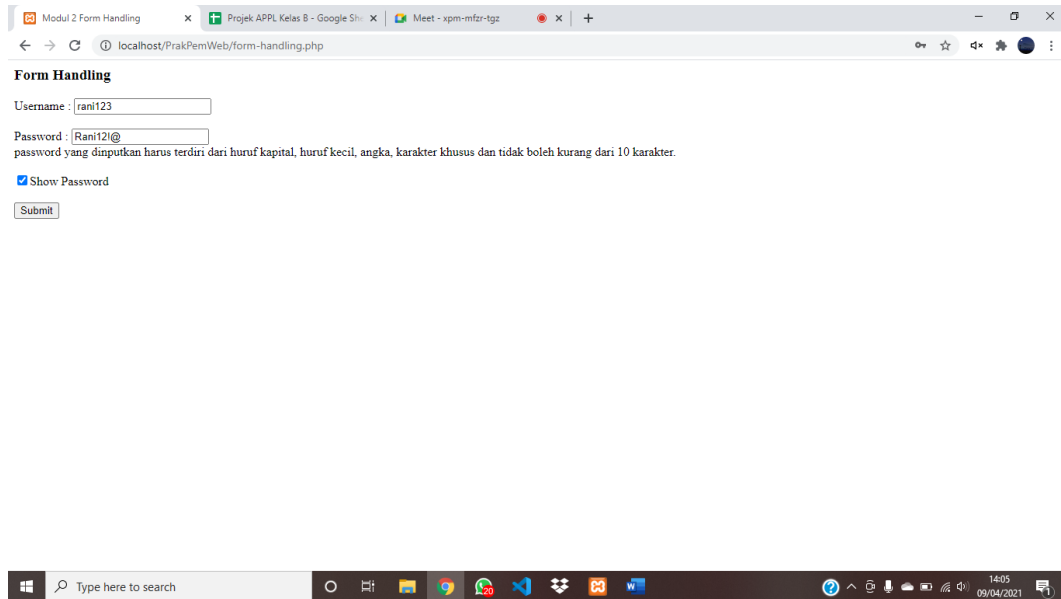
**Gambar 2.1 Tampilan awal**



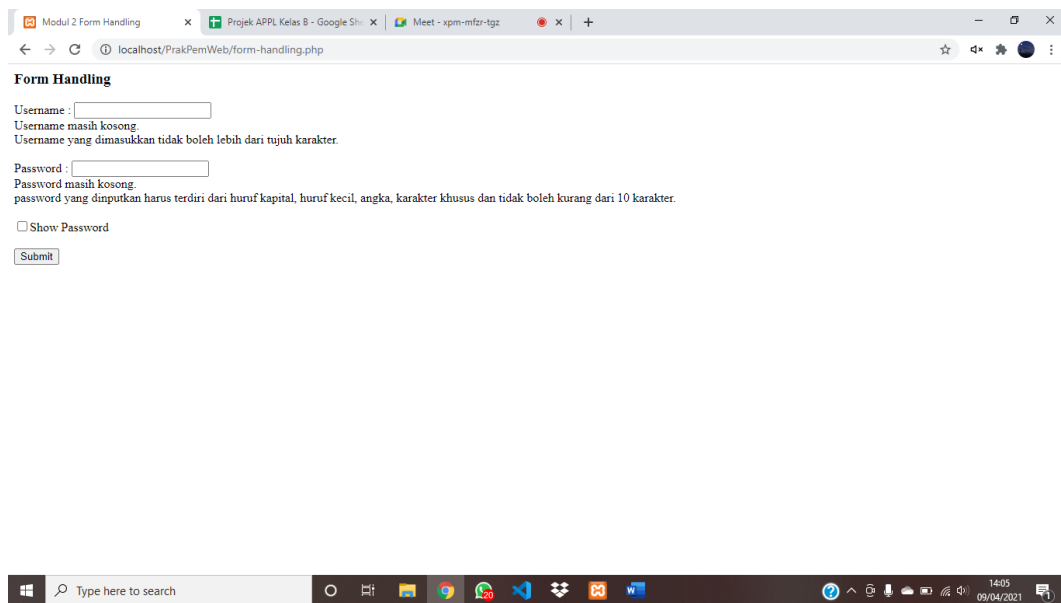
**Gambar 2.2 Tampilan setelah mengisi semua form dengan benar**



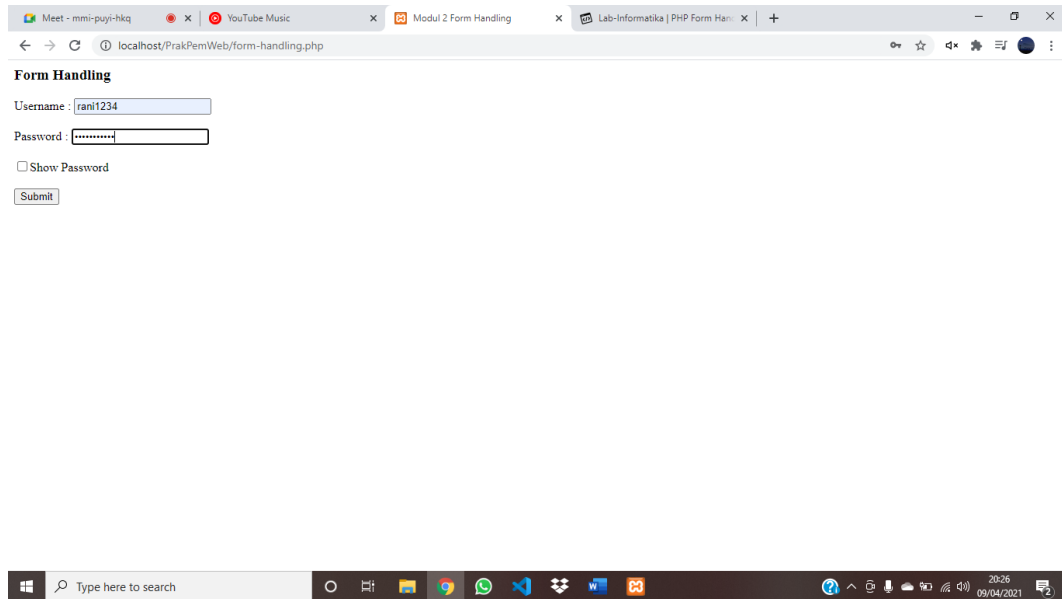
**Gambar 2.3 Tampilan jika username yang dimasukkan tidak sesuai**



**Gambar 2.4 Tampilan jika password yang dimasukkan tidak sesuai**



**Gambar 2.5 Tampilan jika form tidak diisi**



**Gambar 2.6 Tampilan jika “show password” tidak dicentang**

## **BAB III**

### **KESIMPULAN**

**\$\_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.

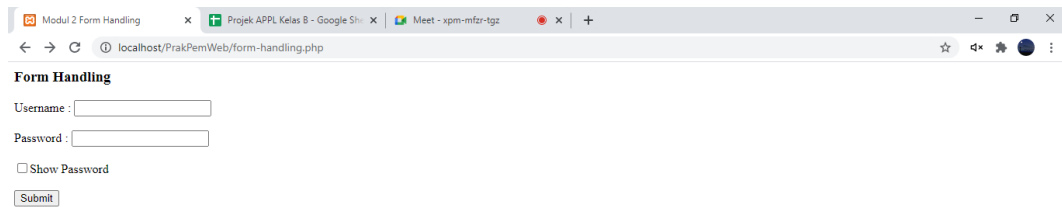
## **DAFTAR PUSTAKA**

Praktikum, K. (n.d.). *MODUL PRAKTIKUM PEMROGRAMAN WEB I Jurusan*

*Teknik Informatika Fakultas Teknik Universitas Palangka Raya.*



# LAMPIRAN



Modul 2 Form Handling x Projek APPL Kelas B - Google S... x Meet - xpm-mfzr-tgz x +

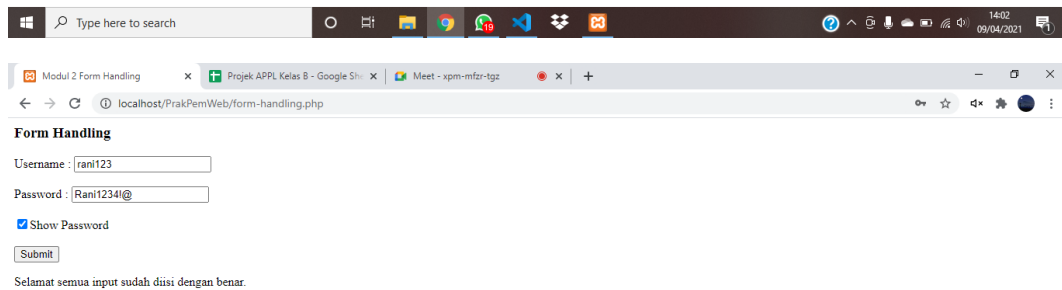
localhost/PrakPemWeb/form-handling.php

### Form Handling

Username :

Password :

☐ Show Password



Type here to search

Modul 2 Form Handling x Projek APPL Kelas B - Google S... x Meet - xpm-mfzr-tgz x +

localhost/PrakPemWeb/form-handling.php

### Form Handling

Username :

Password :

☒ Show Password

Selamat semua input sudah diisi dengan benar.



Type here to search

14:04 09/04/2021

Modul 2 Form Handling x Projek APPL Kelas B - Google Sh... x Meet - xpm-mfzr-tgz x +

localhost/PrakPemWeb/form-handling.php

### Form Handling

Username :

Username yang dimasukkan tidak boleh lebih dari tujuh karakter.

Password :

☒ Show Password

Type here to search

Modul 2 Form Handling x Projek APPL Kelas B - Google Sh... x Meet - xpm-mfzr-tgz x +

localhost/PrakPemWeb/form-handling.php

### Form Handling

Username :

Password :

password yang diinputkan harus terdiri dari huruf kapital, huruf kecil, angka, karakter khusus dan tidak boleh kurang dari 10 karakter.

☒ Show Password

Type here to search

14:05 09/04/2021

Modul 2 Form Handling x Projek APPL Kelas 8 - Google Sh... x Meet - xpm-mfzr-tgz x +

localhost/PrakPemWeb/form-handling.php

### Form Handling

Username :

Username masih kosong.  
Username yang dimasukkan tidak boleh lebih dari tujuh karakter.

Password :

Password masih kosong.  
password yang diinputkan harus terdiri dari huruf kapital, huruf kecil, angka, karakter khusus dan tidak boleh kurang dari 10 karakter.

☐ Show Password

Windows Search: Type here to search

Meet - mmi-puyi-hkq x YouTube Music x Modul 2 Form Handling x Lab-Informatika | PHP Form Han... x +

localhost/PrakPemWeb/form-handling.php

### Form Handling

Username :

Password :

☐ Show Password

Windows Search: Type here to search

20:28 09/04/2021