

**LAPORAN HASIL PRAKTIKUM  
PEMROGRAMAN WEB DAN MOBILE I**



**Nama : Brian Agustian Kristianto**  
**NIM : 193030503066**

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

# **BAB I**

## **TUJUAN DAN LANDASAN TEORI**

### **1.1. TUJUAN**

- 1) Mahasiswa mampu membuat handling yang mampu mengolah data dari form HTML.
- 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>
<input type="submit">
</form>
</body>
</html>
```

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>
<form action="welcome.php" method="post">
Name: <input type="text" name="name"><br> E-mail: <input <html>
<body>
Welcome <?php echo $_POST["name"]; ?><br>
Your email address is: <?php echo $_POST["email"];
?>
</body>
</html>type="text" name="email"><br>
<input type="submit">
</form>
</body>
</html>

```

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

```

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>

```

### 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.

#### a. 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.

<b>Ingat!</b> GET tidak boleh digunakan untuk mengirimkan password atau informasi sensitif lainnya!
---

b. 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.

### 1.2.2. Validasi Form PHP

Pertimbangkan keamanan ketika memproses form PHP!

#### PHP Form Validation Example

\* required field.

Name:  \*

E-mail:  \*

Website:

Comment:

Gender: ☐ Female ☐ Male \*

**Gambar 1.1** form

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

a. 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>

b. 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
```

c. 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.

### 1.2.3. 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 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/"><script>alert('hacked')
</script>">
```

Dengan cara ini, percobaan penyalahgunaan akan gagal.

#### 1.2.4. 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;
}
?>
```

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.

#### 1.2.5. 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 {
        $comment = test_input($_POST["comment"]);
    }
    if (empty($_POST["gender"])) {
        $genderErr = "Gender is required";
    } else {
        $gender = test_input($_POST["gender"]);
    }
}
?>
```

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>

```

#### 1.2.6. 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.

### 1.2.7. 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";
}
```

### 1.2.8. 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.

### **1.3. 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 diinput harus terdiri dari huruf capital, huruf kecil, angka dan karakter khusus.
3. Jumlah karakter password tidak boleh kurang dari sepuluh karakter.

## BAB II

### PEMBAHASAN

Berdasarkan tugas yang diberikan pada bagian bab pertama, yakni membuat program website menggunakan form, berikut ini adalah penjelasan tiap nomor. Berikut ini adalah script untuk tugas kali ini.

```
<html>
<head>
<title>MODUL 2</title>
</head>
<body>
<form action="" method="POST">
  <input type="text" name="name" value="" placeholder="Masukan Userna
me">
  <br>
  <input type="password" name="pass" value="" placeholder="Masukan pa
ssword">
  <br>
  <input type="password" name="cpass" value="" placeholder="Masukan k
embali password">
  <br>
  <input type="submit" name="submit" value="Validasi">
  <input type="reset" value="Reset Form">
</form>
<?php
if(isset($_POST['submit'])){
  $n = $_POST['name'];
  $p = $_POST['pass'];
  $pv = $_POST['cpass'];
  if(empty($n)){
    echo "Tolong isikan Nama Anda!";
  }
  if(strlen($n)>7){
    echo "Username tidak boleh lebih dari tujuh karakter!<br>";
  }
}
```

```

        if(!preg_match("#[0-9]+#", $p)){
            echo "Harap dalam password anda harus mengandung
angka!<br>";
        }
        if(!preg_match("#[a-z]+#", $p)){
            echo "Harap dalam password anda harus mengandung huruf
kecil!<br>";
        }
        if(!preg_match("#[A-Z]+#", $p)){
            echo "Harap dalam password anda harus mengandung huruf
besar/kapital!<br>";
        }
        if(!preg_match("/[\\'$%&*()]{#~?><>,|=_-}/", $p)){
            echo "Harap dalam password anda harus mengandung karakter
unik!<br>";
        }
        if($p != $pv){
            echo "Kedua password tersebut tidak cocok!, harap isi ulang
lagi.<br>";
        }
    }
?>
</body>
</html>

```

```

1  <html>
2  <head>
3  <title>MODUL 2</title>
4  </head>
5  <body>
6  <form action="" method="POST">
7      <input type="text" name="name" value="" placeholder="Masukan Username">
8      <br>
9      <input type="text" name="pass" value="" placeholder="Masukan password">
10     <br>
11     <input type="text" name="cpass" value="" placeholder="Masukan kembali password">
12     <br>
13
14     <input type="submit" name="submit" value="Validasi">
15     <input type="reset" value="Reset Form">
16 </form>

```

**Gambar 2.1** form

Dalam bagian body website ini, terdapat element form yaitu input. Elemen input memiliki beberapa tipe, berikut ini adalah tipe yang dideklarasikan di dalam elemen input:

a. Text

Ketika Anda ingin memasukkan data dalam bentuk tertulis, angka atau simbol, jenis teks yang akan digunakan. Elemen tipe teks

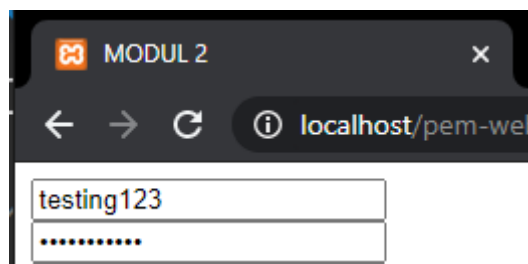


biasanya digunakan untuk mengisi data nama lengkap. Dalam tugas ini ada beberapa tipe text yaitu

```
<input type="text" name="name" value="" placeholder="Masukan Username">  
<br>  
<input type="password" name="pass" value="" placeholder="Masukan password">  
<br>  
<input type="password" name="cpass" value="" placeholder="Masukan kembali password">  
<br>
```

**Gambar 2.2** text

Input text nantinya ketika diinputkan akan berupa teks, sedangkan input type password akan berupa titik hitam. Placeholder adalah teks yang ada dalam form yang menunjukkan mana yang isian username dan mana yang isian password.



**Gambar 2.3** text dan password

b. Tombol submit

Tombol "Kirim" digunakan sebagai tombol. Mengklik tombol akan menginstruksikan pemrosesan formulir untuk mengirimkan data ke halaman target. Secara default, teks yang tertulis di tombol "submit" adalah kata "submit". Berikut ini adalah gambar untuk tombol submit.

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

**Gambar 2.4** submit

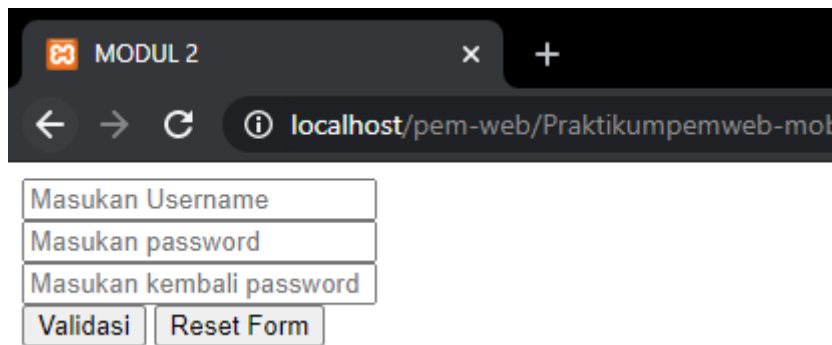
c. Tombol reset

Tombol reset digunakan untuk mereset form yang diisi (kosong). Sama seperti tombol submit, nilai properti ini dapat digunakan untuk mengubah teks yang tertulis pada tombol reset. Label yang digunakan adalah:

```
<input type="reset" value="Reset Form">
```

**Gambar 2.5** reset

Ini adalah bentuk web yang telah di buat.



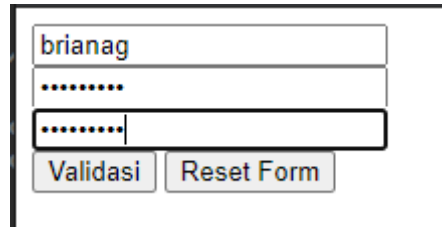
**Gambar 2.6** website form

Lanjut pada bagian validasi, pada bagian ini bertujuan untuk mengecek apakah username atau password yang diberikan sesuai dengan kriteria atau tidak. Pada gambar di bawah ini adalah variable untuk username, password, dan confirm password.

```
<?php
if(isset($_POST['submit'])){
    $n = $_POST['name'];
    $p = $_POST['pass'];
    $pv = $_POST['cpass'];
```

**Gambar 2.7** variabel

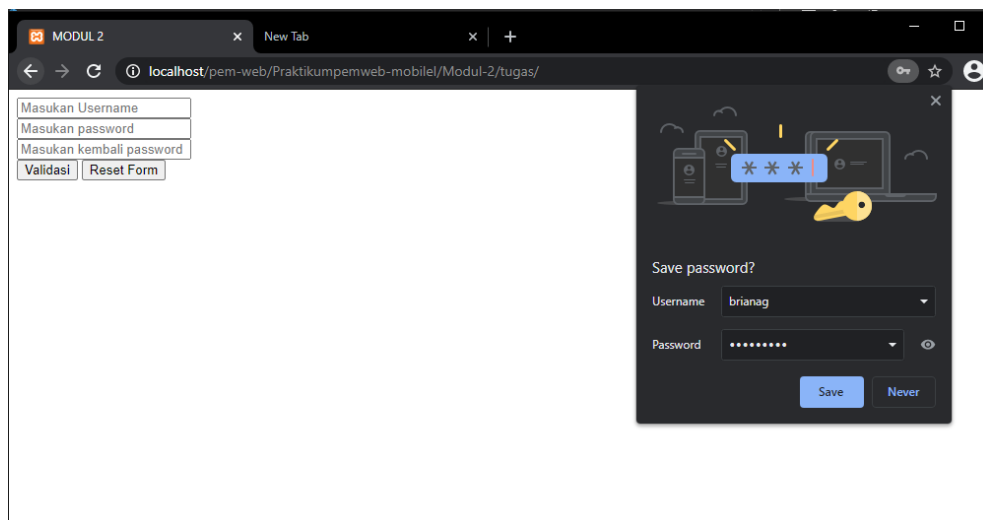
Pada nomor satu, salah satu kriteria untuk username adalah username tidak boleh lebih dari tujuh. Maka dari itu digunakanlah `strlen($n)<7`. `$n` merupakan variable untuk name (username). Untuk melihat apakah kode ini berjalan atau tidak, berikut ini adalah gambarnya.



A screenshot of a web form. The first input field contains the text 'brianag'. Below it are two more input fields, both filled with dots representing masked passwords. At the bottom of the form are two buttons: 'Validasi' and 'Reset Form'.

**Gambar 2.8** username berjumlah tujuh karakter

Apabila divalidasi, website tersebut akan menyimpan username dan password yang berarti username tersebut memenuhi kriteria nomor satu. Berikut ini adalah gambarnya.

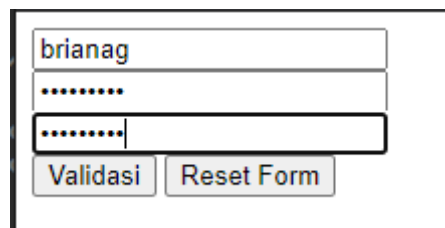


A screenshot of a web browser window. The address bar shows 'localhost/pem-web/Praktikumpemweb-mobile/Modul-2/tugas/'. The page content shows a form with three input fields: 'Masukan Username', 'Masukan password', and 'Masukan kembali password'. Below these fields are 'Validasi' and 'Reset Form' buttons. A modal dialog box is open on the right side of the screen, titled 'Save password?'. It contains a 'Username' field with the value 'brianag' and a 'Password' field with masked characters. At the bottom of the dialog are 'Save' and 'Never' buttons.

**Gambar 2.9** validasi berhasil

Kemudian untuk nomor dua dan tiga, terdapat beberapa kriteria, yakni: password yang diinputkan harus terdiri dari huruf kapital, huruf kecil, angka dan karakter khusus dan jumlah karakter password tidak boleh kurang dari sepuluh karakter. Khusus untuk nomor dua ini menggunakan `preg_match`. Fungsi `preg_match ()` mengembalikan apakah kecocokan

ditemukan dalam string. Ada empat `preg_match` yang dipakai dalam tugas ini, diantaranya `preg_match("#[0-9]+#", $p)`, `preg_match("#[a-z]+#", $p)`, `preg_match("#[A-Z]+#", $p)`, dan `preg_match("/[\\'\"^$%&*()}{#~?><>,|_=-]/", $p)`. Masing-masing dari `preg_match` tersebut berfungsi agar password harus mengandung karakter huruf besar dan kecil, karakter unik/special, dan angka. Lanjut ke bagian nomor 3, kriteria untuk nomor tiga ini adalah karakter dalam password ini harus berjumlah minimal sepuluh karakter. Sama seperti nomor satu sebelumnya, untuk membuat batas minimal jumlah karakter ini tetap menggunakan `strlen($p)<10`. Berikut ini adalah contohnya.



The image shows a web form with three input fields stacked vertically. The first field contains the text 'brianag'. The second field contains ten dots, representing a password. The third field contains seven dots, also representing a password. Below the input fields are two buttons: 'Validasi' and 'Reset Form'.

**Gambar 2.10** password

Password di atas adalah *Maxdc191*<. Apabila divalidasi maka website tersebut juga akan menyimpan username dan password yang berarti berhasil.

### **BAB III**

### **KESIMPULAN**

Melalui bab ini, mahasiswa dapat membuat form dalam html atau php. Karena form sangat sering digunakan bahkan paling penting dalam sebuah website, terutama form login dalam setiap website sosial media seperti facebook, twitter, Instagram, dll. Pada bab ini juga dibahas bagaimana cara membuat kriteria-kriteria terhadap form yang akan dibuat, seperti jumlah karakter, huruf kecil atau besar, dll.

## DAFTAR PUSTAKA

(Informatika, 2021) Informatika, J. T. (2021). *Modul Praktikum Basis Data I*.

W3schools.com. (n.d.). *PHP preg\_match() Function*.

[https://www.w3schools.com/php/func\\_regex\\_preg\\_match.asp](https://www.w3schools.com/php/func_regex_preg_match.asp)

## LAMPIRAN

```
<html>
<head>
<title>MODUL 2</title>
</head>
<body>
<form action="" method="POST">
  <input type="text" name="name" value="" placeholder="Masukan Userna
me">
  <br>
  <input type="password" name="pass" value="" placeholder="Masukan pa
ssword">
  <br>
  <input type="password" name="cpass" value="" placeholder="Masukan k
embali password">
  <br>
  <input type="submit" name="submit" value="Validasi">
  <input type="reset" value="Reset Form">
</form>
<?php
  if(isset($_POST['submit'])){
    $n = $_POST['name'];
    $p = $_POST['pass'];
    $pv = $_POST['cpass'];
    if(empty($n)){
      echo "Tolong isikan Nama Anda!";
    }
    if(strlen($n)>7){
      echo "Username tidak boleh lebih dari tujuh karakter!<br>";
    }
  }
```

```

        if(!preg_match("#[0-9]+#", $p)){
            echo "Harap dalam password anda harus mengandung
angka!<br>";
        }
        if(!preg_match("#[a-z]+#", $p)){
            echo "Harap dalam password anda harus mengandung huruf
kecil!<br>";
        }
        if(!preg_match("#[A-Z]+#", $p)){
            echo "Harap dalam password anda harus mengandung huruf
besar/kapital!<br>";
        }
        if(!preg_match("/[!@#$%^&*(){}#~?><>,|=_-]/", $p)){
            echo "Harap dalam password anda harus mengandung karakter
unik!<br>";
        }
        if($p != $pv){
            echo "Kedua password tersebut tidak cocok!, harap isi ulang
lagi.<br>";
        }
    }
?>
</body>
</html>

```

```

1  <html>
2  <head>
3  <title>MODUL 2</title>
4  </head>
5  <body>
6  <form action="" method="POST">
7      <input type="text" name="name" value="" placeholder="Masukan Username">
8      <br>
9      <input type="text" name="pass" value="" placeholder="Masukan password">
10     <br>
11     <input type="text" name="cpass" value="" placeholder="Masukan kembali password">
12     <br>
13
14     <input type="submit" name="submit" value="Validasi">
15     <input type="reset" value="Reset Form">
16 </form>

```

**Gambar 2.1** form

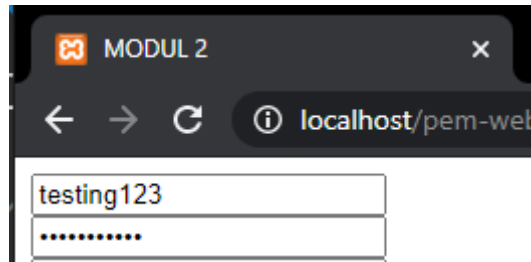
```

<input type="text" name="name" value="" placeholder="Masukan Username">
<br>
<input type="password" name="pass" value="" placeholder="Masukan password">
<br>
<input type="password" name="cpass" value="" placeholder="Masukan kembali password">
<br>

```

**Gambar 2.2** text





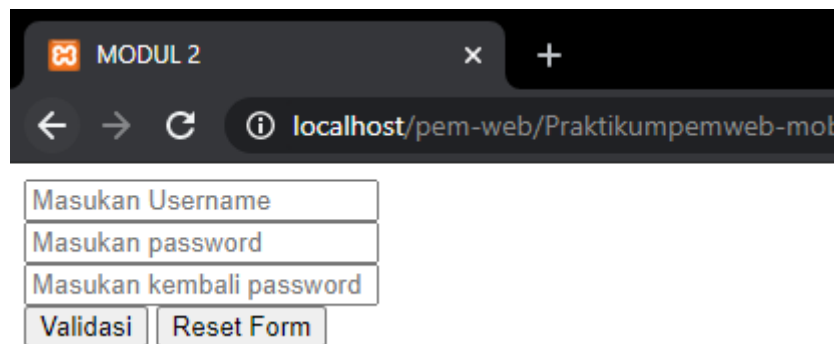
**Gambar 2.3** text dan password

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

**Gambar 2.4** submit

```
<input type="reset" value="Reset Form">
```

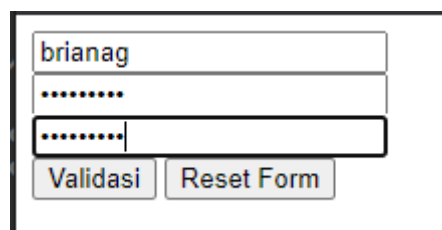
**Gambar 2.5** reset



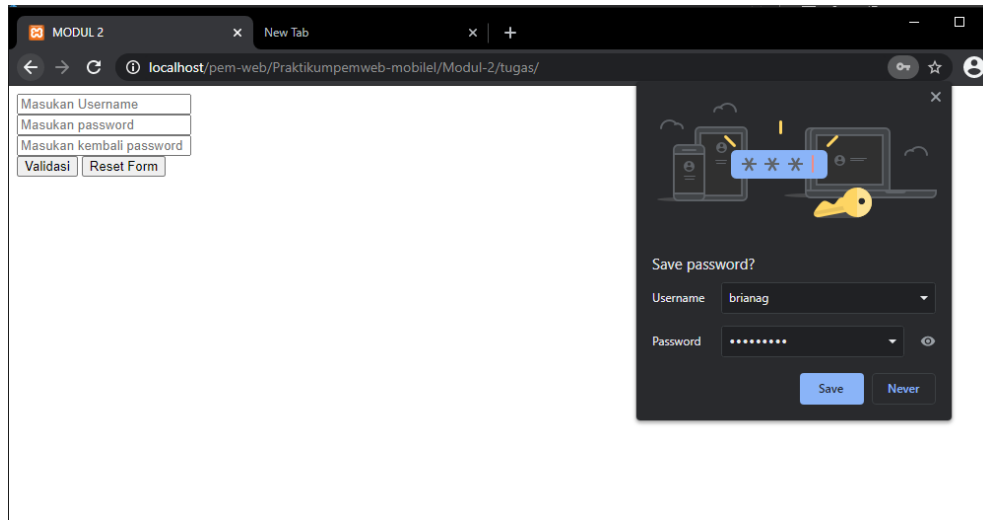
**Gambar 2.6** website form

```
<?php
if(isset($_POST['submit'])){
    $n = $_POST['name'];
    $p = $_POST['pass'];
    $pv = $_POST['cpass'];
```

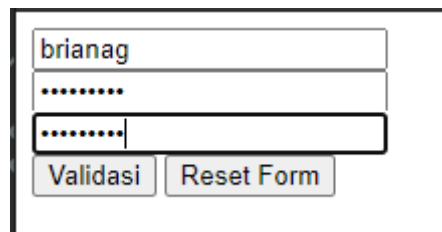
**Gambar 2.7** variabel



**Gambar 2.8** username berjumlah tujuh karakter



**Gambar 2.9** validasi berhasil



**Gambar 2.10** password