

**LAPORAN PRATIKUM**  
**PEMROGRAMAN WEB DAN MOBILE I**



**NAMA : Rakhel Cakra Kusumadinata Sera**  
**NIM : 193030503062**  
**MODUL : II**  
**KELAS : A**

**JURUSAN/PROGRAM STUDI 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 1.1 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>
    Welcome <?php echo $_POST["name"]; ?><br>
    Your email address is: <?php echo $_POST["email"];
    ?> </body>
</html>
```

**Gambar 1.2 HTML dan PHP**

Jika field nama diinputkan dengan Tono dan email diinputkan dengan ton@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 1.3 HTML dan PHP**

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 1.6 HTML dan PHP**

### **GET vs. POST**

GET dan POST membuat sebuah array (contoh array(kunci => nilai, kunci2 => nilai2, kunci3 => nilai3, ...)). Array ini menyimpan pasangan kunci/nilai, dimana kuncikunci 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 1.7 VALIDASI**

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>
```

**Gambar 1.8** Text Field

### 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
```

**Gambar 1.9** Text Radio Button

### Form Element

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

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

**Gambar 1.10** From Element

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"];?>">
```

### **Gambar 1.11 PHP**

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

```
http://localhost/test_form.php/%22%3E%3Cscript%3Ealert('hacked')%3C/scr ipt %3E
```

yang jika ditranslasikan akan menjadi:

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

### **Gambar 1.12 PHP**

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**

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/&quot;&gt;&lt;script&gt;alert('hacked')&lt;/script&gt;">
```

**Gambar 1.13** Script

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 1.14 PHP**

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 = "";
    }
}

```

**Gambar 1.15 PHP**

```

    } 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"]);
    }
}
?>

```

**Gambar 1.16 PHP**

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

```

Gambar 1.17 PHP

```

<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 1.18 PHP

### 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 1.19 Validasi Nama

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 1.20** Validasi Email

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

**Gambar 1.21** Validasi 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>`.

**Gambar 1.22** Validasi URL

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 1.23** Validasi URL

## BAB II

### PEMBAHASAN

#### Tugas Praktikum

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.

#### Pengerjaan Praktikum

```
<link rel="stylesheet" href="atur.css">
<form action="modul2.php" method="POST">
  <fieldset>
    <legend>Login</legend>
    <p>
      <label>Username:</label>
      <input type="text" name="username">
    </p>
    <p>
      <label>Password:</label>
      <input type="password" name="password">
    </p>
    <p>
      <button type="submit">Login</button>
```

```
</p>

</fieldset>

</form>
```

Yang pertama, harus membuat html untuk membuat login form. Untuk style css diatur pada file *atur.css* yang dimana disambungkan dengan tag `<link></link>`. Setelah itu membuat tag `<form></form>` yang berfungsi sebagai untuk mengumpulkan informasi melalui inputan pengguna. Attribute Action pada yang berfungsi untuk menentukan dimana data yang diinput oleh form akan diproses. Dapat dilihat isi attribute adalah *modul2.php*. attribute selanjutnya yaitu Method yang dimana isi attribute adalah *POST* yang berfungsi untuk mengirimkan informasi melalui method HTTP POST.

Lalu tag fieldset disini untuk mengelompokkan beberapa objek form menjadi sebuah kelompok, sedangkan tag legend berfungsi untuk menampilkan judul dari kelompok objek form yang dimana kodingan diatas adalah “Login”.

Tag label disini berfungsi untuk memudahkan penggunaan dalam memasukkan data (membuat form menjadi user friendly). Tag label diisi oleh “Username” dan “Password”. Lalu diikuti input dengan attribute name adalah “username” dan attribute type adalah “text”. Selanjutnya ada attribute name adalah “password” dengan type -nya adalah “password”. Dan ketika semua objek form telah diisi, maka tag button *Login* akan mengirimkan informasi .

```
<?php

if ($_SERVER["REQUEST_METHOD"] == "POST") {

    $username = $_REQUEST["username"];

    $password = $_REQUEST["password"];

    $user = strlen($username);

    $pass = strlen($password);

    $x = false;
```

Pada kondisi if else, dilakukan pengecekan data. Menggunakan `$_SERVER['REQUEST_METHOD']` untuk mengecek apakah `$_SERVER['REQUEST_METHOD']` sama dengan *POST*. Lalu variable `$username` akan berisi username yang dikirim menggunakan form tadi dan begitu juga dengan `$password`. Untuk variable `$user` akan menghitung panjang `$username` dan variable `$pass` yang menghitung panjang `$password`.

```
if ($user > 7) {  
    echo "username lebih dari 7";  
    $x = true;  
}  
  
if (!preg_match("/[A-Z]/", $password)) {  
    echo "password kapital\n";  
    $x = true;  
}  
  
if (!preg_match("/[a-z]/", $password)) {  
    echo "password kecil\n";  
    $x = true;  
}  
  
if (!preg_match("/^[a-zA-Z\d]/", $password)) {  
    echo "password special character\n";  
    $x = true;  
}
```

```

if (!preg_match("/[0-9]/", $password)) {

    echo "password digit\n";

    $x = true;

}

if ($pass > 10) {

    echo "password lebih dari 10";

    $x = true;

}

if ($x == false) {

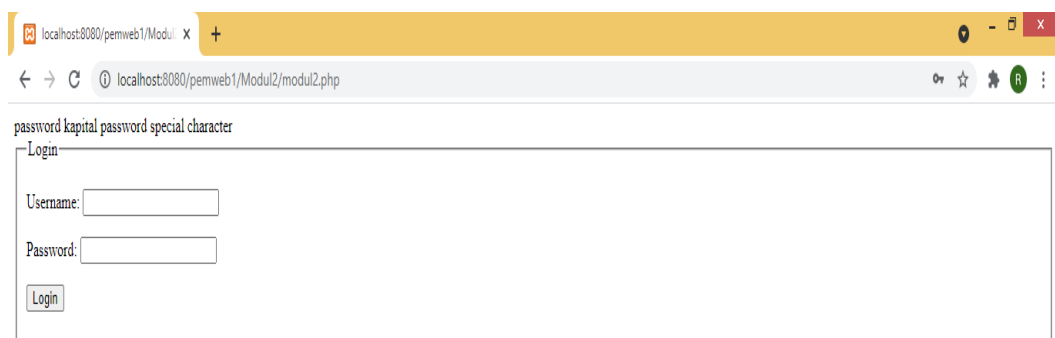
    echo "berhasil";

}

```

Lalu menggunakan if else, akan melakukan pengecekan sesuai dengan permasalahan yang telah diberikan. *preg\_match()* disini berfungsi untuk mengecek apakah string yang ditentukan sesuai dengan pola yang diberi. Bila tidak, maka program akan memberikan peringatan sesuai permasalahan yang ada. Dan variable *\$x* akan bernilai *TRUE* yang berarti program akan kembali meminta memasukkan form hingga persyaratannya terpenuhi.

Output yang diberikan:



The screenshot shows a web browser window with the address bar displaying 'localhost:8080/pemweb1/Modul2/modul2.php'. The page content includes a login form with the following elements:

- Header text: "password kapital password special character"
- Form title: "Login"
- Username input field: "Username:
- Password input field: "Password:
- Login button: "Login"

**Gambar 2. 1** Output gagal



berhasil

Login

Username:

Password:

**Gambar 2. 2** Output berhasil

### **BAB III**

#### **KESIMPULAN**

Penggunaan form pada php saat memudahkan dalam penginputan data. Lalu method juga akan menyesuaikan kondisi yang diberikan pada PHP. Misalnya pada kasus kali ini menggunakan method POST dan action dilakukan file itu sendiri. Lalu penggunaan fungsi seperti `$_SERVER[REQUEST_METHOD]` untuk pengecekan suatu kondisi dan `$preg_match` yang berfungsi untuk melakukan melihat pola string yang sesuai dengan yang diminta.

## DAFTAR PUSTAKA

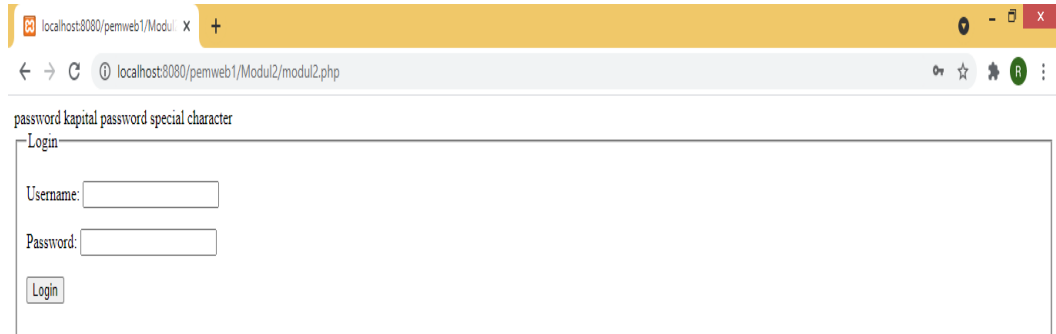
Dosen Teknik Informatika. (2021). MODUL PRAKTIKUM PEMROGRAMAN WEB I Jurusan Teknik Informatika Fakultas Teknik Universitas Palangka Raya. *MODUL PRAKTIKUM PEMROGRAMAN WEB I Jurusan Teknik Informatika Fakultas Teknik Universitas Palangka Raya*.

DUMET School. (2018). *Memahami PHP Request Methods*. Dumet School.

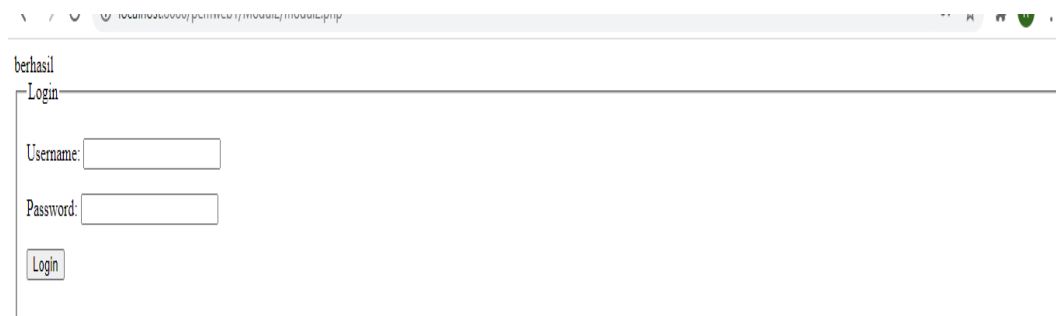
Muhardian, A. (2018). *Belajar HTML #11: Cara Membuat Form pada HTML*. Petani Kode. <https://www.petanikode.com/html-form/>

w3schools. (n.d.). *PHP substr\_count() Function*. W3schools. Retrieved March 29, 2021, from [https://www.w3schools.com/php/func\\_string\\_substr\\_count.asp](https://www.w3schools.com/php/func_string_substr_count.asp)

## LAMPIRAN



**Gambar 2. 3** Output gagal



**Gambar 2. 4** Output berhasil

### Source Code:

```
<?php
if ($_SERVER["REQUEST_METHOD"] == "POST") {
    $username = $_REQUEST["username"];
    $password = $_REQUEST["password"];
    $user = strlen($username);
    $pass = strlen($password);
    $x = false;
```

```

if ($user > 7) {
    echo "username lebih dari 7";
    $x = true;
}
if (!preg_match("/[A-Z]/", $password)) {
    echo "password kapital\n";
    $x = true;
}
if (!preg_match("/[a-z]/", $password)) {
    echo "password kecil\n";
    $x = true;
}

if (!preg_match("/[^a-zA-Z\d]/", $password)) {
    echo "password special character\n";
    $x = true;
}

if (!preg_match("/[0-9]/", $password)) {
    echo "password digit\n";
    $x = true;
}
if ($pass > 10) {
    echo "password lebih dari 10";
    $x = true;
}
if ($x == false) {
    echo "berhasil";
}
}
?>
<link rel="stylesheet" href="atur.css">
<form action="modul2.php" method="POST">
    <fieldset>
        <legend>Login</legend>
        <p>
            <label>Username:</label>
            <input type="text" name="username">
        </p>
        <p>
            <label>Password:</label>
            <input type="password" name="password">
        </p>
        <p>
            <button type="submit">Login</button>
        </p>
    </fieldset>
</form>

```