

**LAPORAN PRAKTIKUM  
PEMROGRAMAN WEB & APLIKASI I**



**NAMA : Muhammad Ilyas Abdullah**  
**NIM : 185110500111013**  
**KELAS : C (Permata Merdeka)**  
**MODUL : 2**

**JURUSAN TEKNIK INFORMATIKA  
FAKULTAS TEKNIK  
UNIVERSITAS PALANGKARAYA  
2021**

## BAB 1

### 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

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

Welcome Budi

Your email address is tonos@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

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 1.4 HTML

## 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.

PS: 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 1.5 HTML

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

Gambar 1.6 HTML

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="E-mail">

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

Comment: <textarea name="comment" row="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"]);? >">
```

## 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/%3cnama\\_folder%3e/test\\_form.php/%22%3E%3Cscript%3Ealert\('hacked'\)%3C/script](http://localhost/%3cnama_folder%3e/test_form.php/%22%3E%3Cscript%3Ealert('hacked')%3C/script)

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>`!

## 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;/scr  
ipt&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 1.7 HTML

Ingat bahwa pada permulaan skrip, adalah pemeriksaan apakah form sudah di submit 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 {
        $comment = test_input($_POST["comment"]);
    }

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

```

Gambar 1.8 HTML

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

```

```

☐* <?php echo $genderErr;?></span>
<br><br>


```

Gambar 1.9 HTML

### 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 1.10 HTML

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 1.11 HTML

## 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 1.12 HTML

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 dan tag . 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 1.13 HTML

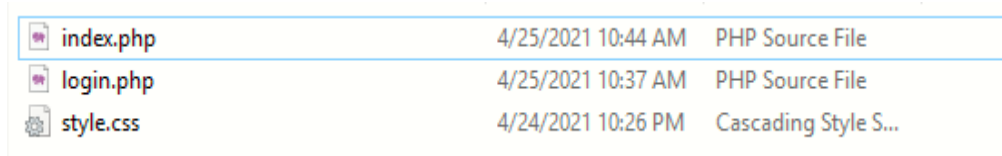
## BAB 2




### 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.

Dalam proses untuk menghasilkan program di atas, penulis membuat 3 file dalam pembuatannya, yang pertama adalah file `index.php`, `style.css` dan `login.php`. Ketiga file ini memiliki fungsinya masing-masing. Untuk file `index.php` digunakan sebagai tempat untuk pembuatan html, kemudian `style.css` digunakan untuk pembuatan layout dari tampilan login dan `login.php` adalah file untuk memproses variabel sehingga dapat menghasilkan hasil yang sesuai dengan kriteria yang disebutkan di atas.



 <code>index.php</code>	4/25/2021 10:44 AM	PHP Source File
 <code>login.php</code>	4/25/2021 10:37 AM	PHP Source File
 <code>style.css</code>	4/24/2021 10:26 PM	Cascading Style S...

*Gambar 2.1. File yang digunakan dalam penyelesaian masalah yang ada di atas*

Adapun untuk tampilan hasil code yang akan digunakan di dalam file tersebut antara lain:

```

1 <!DOCTYPE html>
2 <html>
3 <head>
4   <title>MK Pemrograman dan Web App I</title>
5   <link rel="stylesheet" type="text/css" href="style.css">
6   <meta charset="UTF-8">
7   <meta name="viewport" content="width=device-width, initial-scale=1.0">
8 </head>
9 <body>
10  <form action="login.php" method="post">
11    <h2>Login</h2>
12    <?php if (isset($_GET['error'])) { ?>
13      <p class="error"><?php echo $_GET['error']; ?></p>
14    <?php } ?>
15    <label>User Name</label>
16    <input type="text" name="uname" placeholder="user name"><br>
17
18    <label>password</label>
19    <input type="password" name="password" placeholder="uppercase, lowercase, number and special character is a must"><br>
20
21    <button type="submit">Login</button>
22  </form>
23 </body>
24 </html>

```

*Gambar 2.2. Souce code untuk file index.php*

Melalui file yang ada di atas, penulis menggunakan metode POST yang mana metode ini digunakan untuk mengumpulkan data form yang ada. Disini dapat dilihat penggunaan fungsi if untuk dapat memanggil fungsi post yang ada di file index. Pada bagian input password, ada penambahan placeholder=uppercase, lowercase, number and special character is a must untuk memudahkan pengguna untuk mengetahui jenis password yang harus dimasukan. Kemudian selanjutnya ada file style.css yang mana file digunakan untuk memberikan sedikit tampilan dari login form yang digunakan. Untuk kode dari file css dapat dilihat dalam gambar dibawah ini:

```

body {
  background: #1690A7;
  display: flex;
  justify-content: center;
  align-items: center;
  height: 100vh;
}

* {
  font-family: sans-serif;
  box-sizing: border-box;
}

form {
  width: 500px;
  border: 2px solid #ccc;
  padding: 30px;
  background: #fff;
  border-radius: 15px;
}

h2 {
  text-align: center;
  margin-bottom: 40px;
}

input {
  display: block;
  border: 2px solid;
  width: 95%;
  padding: 10px;
  margin: 10px auto;
  border-radius: 5px;
}

label {
  color: #888;
  font-size: 18px;
  padding: 10px;
}

```

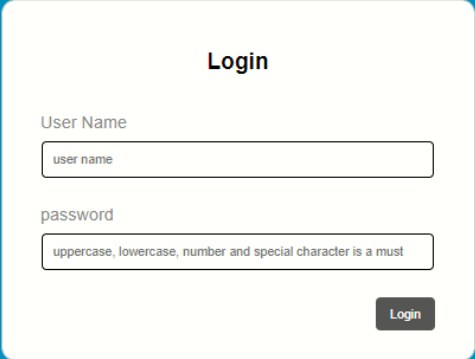
```

42
43 ▼ button {
44     float: right;
45     background: #555;
46     padding: 10px 15px;
47     color: #fff;
48     border-radius: 5px;
49     margin-right: 10px;
50     border: none;
51 }
52
53 button:hover{
54     opacity: .7;
55 }
56
57 ▼ .error {
58     background-color: #F2DEDE;
59     color: #A94442;
60     padding: 10px;
61     width: 95%;
62     border-radius: 5px;
63     margin: 20px auto;
64 }

```

*Gambar 2.3. Source code untuk file css*

Dalam file css tersebut, kita dapat melihat beberapa hal yang di layoutkan seperti body, form, h2, input, label. Button, button hover dan error. Dalam pembuatan login form menggunakan form sans-serif. Adapun untuk hasil implementasi dari file index.php serta file style.css adalah sebagai berikut:



**Login**

User Name

user name

password

uppercase, lowercase, number and special character is a must

Login

*Gambar 2.4. Tampilan layout dari login form yang dibuat*

Kemudian untuk file login.php sebagai tempat variabel serta fungsi yang mana dapat dilihat pada gambar dibawah ini:

```

1  <?php
2  session_start();
3
4  if (isset($_POST['uname']) && isset($_POST['password'])) {
5      function validate($data){
6          $data = trim($data);
7          $data = stripslashes($data);
8          $data = htmlspecialchars($data);
9          return $data;
10     }
11     $uname = validate($_POST['uname']);
12     $pass = validate($_POST['password']);
13     $user = strlen($uname);
14     $passwords = strlen($pass);
15     if (empty($uname)) {
16         header("Location: index.php?error=user name is required");
17         exit();
18     }else if (empty($pass)){
19         header("Location: index.php?error=password is required");
20         exit();
21     }else if ($user>7){
22         header("Location: index.php?error=username must consist of seven characters!");
23         exit();
24     }else if (!preg_match("/[A-Z]/", $pass)){
25         header("Location: index.php?error=UPPERCASE is a must!");
26         exit();
27     }else if (!preg_match("/[a-z]/", $pass)){
28         header("Location: index.php?error=lowercase is a must!");
29         exit();
30     }else if (!preg_match("/^[a-zA-Z\d]/", $pass)){
31         header("Location: index.php?error=special characters is a must!");
32         exit();
33     }else if (!preg_match("/[0-9]/", $pass)){
34         header("Location: index.php?error=Number! dont forget to put number ffs!");
35         exit();
36     }else if ($passwords<10){
37         header("Location: index.php?error=do you want to get hacked? 10 or more characters for password!");
38         exit();
39     }else{
40         echo "login success! welcome to our website $uname!";
41     }
42
43     }else{
44         header("Location: index.php");
45         exit();
46     }
47
48     ?>

```

Gambar 2.5. Tampilan file login.php

Untuk menyelesaikan penugasan yang ada, yang pertama adalah pembuatan username yang tidak boleh lebih dari 7 karakter. Dalam pembuatannya kita dapat menggunakan fungsi if ... elseif ... else dapat kita lihat di bawah ini

```

else if ($user>7){
header("Location: index.php?error=username must consist of seven characters!");
exit();
}

```

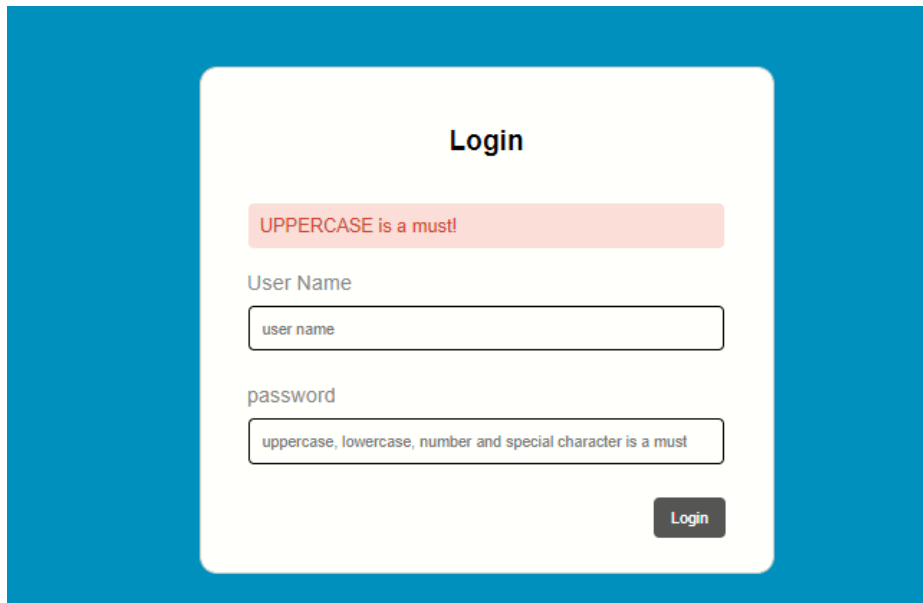
Jika variabel \$user diinputkan oleh user melebihi 7 karakter maka header akan memunculkan error yang berbunyi “username must consist of seven characters!” sehingga user tidak dapat masuk.



Kemudian untuk penugasan yang kedua, password yang diinputkan harus terdiri dari huruf kapital, huruf kecil, angka dan karakter khusus. Kita dapat menggunakan fungsi if ... elseif ... else yang mana masih dalam satu root

```
}else if (!preg_match("/[A-Z]/", $pass)){
    header("Location: index.php?error=UPPERCASE is a must!");
    exit();
}else if (!preg_match("/[a-z]/", $pass)){
    header("Location: index.php?error=lowercase is a must!");
    exit();
}else if (!preg_match("/[^a-zA-Z\d]/", $pass)){
    header("Location: index.php?error=special characters is a must!");
    exit();
}else if (!preg_match("/[0-9]/", $pass)){
    header("Location: index.php?error=Number! dont forget to put number
ffs!");
    exit();
}
```

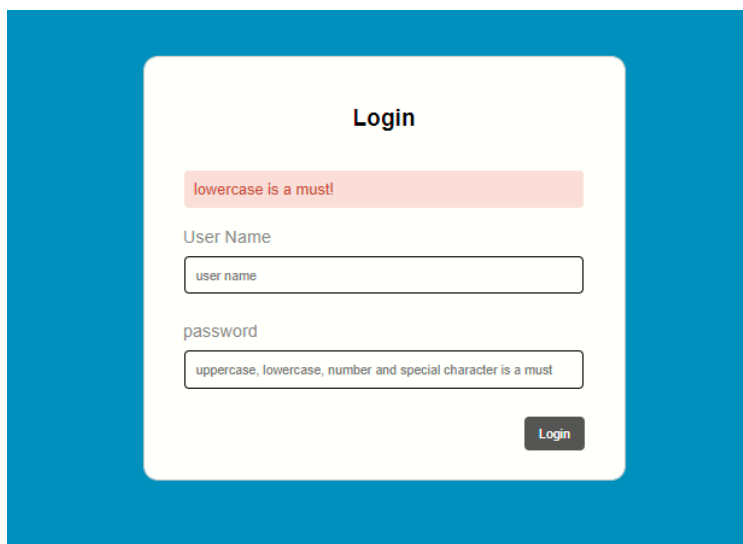
Disini kita menggunakan fungsi built-in dari PHP yaitu preg\_match. Preg\_match sendiri memiliki fungsi untuk mencari string berdasarkan pola, mengembalikan nilai true jika polanya ada, false jika polanya tidak ada. Untuk penulisan huruf kapital pada password fungsi Preg\_match akan mengidentifikasi string jika kemudian ada yang tidak memasukan huruf kapital sehingga akan memunculkan header berwarna merah yang akan menginstruksikan user untuk memasukan huruf kapital ke dalam tag input seperti yang ada di bawah ini:



The image shows a login form titled "Login" on a blue background. The form has two input fields: "User Name" and "password". The "User Name" field contains the text "user name". The "password" field contains the text "uppercase, lowercase, number and special character is a must". Above the "password" field, there is a red error message that says "UPPERCASE is a must!". A "Login" button is located at the bottom right of the form.

*Gambar 2.6. Peringatan huruf kapital*

Untuk huruf kecil, angka dan karakter khusus juga menggunakan fungsi if elseif else dengan built in Preg\_match yang memiliki fungsi serupa pada fungsi untuk memasukan huruf kapital. Sehingga ketika user memasukan password yang tidak ada huruf kecilnya maka header akan keluar dan meminta user untuk memasukan huruf kecil ke dalam form password begitu juga dengan angka serta karakter khusus. Untuk output dari masing-masing fungsi dapat dilihat pada gambar di bawah ini:



The image shows a login form titled "Login" on a blue background. The form has two input fields: "User Name" and "password". The "User Name" field contains the text "user name". The "password" field contains the text "uppercase, lowercase, number and special character is a must". Above the "password" field, there is a red error message that says "lowercase is a must!". A "Login" button is located at the bottom right of the form.

*Gambar 2.7. Header jika user tidak memasukan huruf kecil ke dalam form password*

The image shows a login form titled "Login" on a blue background. The form has two input fields: "User Name" and "password". The "User Name" field contains the text "user name". The "password" field contains the text "uppercase, lowercase, number and special character is a must". Above the "password" field, there is a red error message box that says "Number! dont forget to put number ffs!". A "Login" button is located at the bottom right of the form.

*Gambar 2.8. Header jika user tidak memasukan angka ke dalam form password*

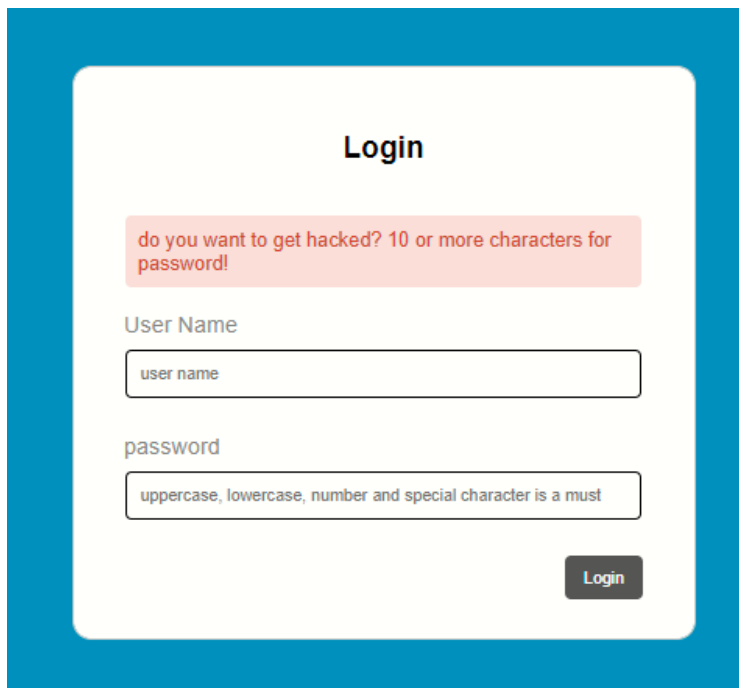
The image shows a login form titled "Login" on a blue background. The form has two input fields: "User Name" and "password". The "User Name" field contains the text "user name". The "password" field contains the text "uppercase, lowercase, number and special character is a must". Above the "password" field, there is a red error message box that says "special characters is a must!". A "Login" button is located at the bottom right of the form.

*Gambar 2.9. Header jika user tidak memasukan karakter spesial ke dalam form password*

Untuk tugas yang terakhir adalah untuk membuat password yang dimasukan ke dalam form tidak melebihi dari 10 password. Disini kita menggunakan fungsi yang sama yang juga digunakan pada form username. Adapun untuk kode yang digunakan dapat dilihat pada kode yang ada di bawah ini:

```
}else if ($passwords<10){  
    header("Location: index.php?error=do you want to get hacked? 10 or  
more characters for password!");  
    exit();
```

Ketika user memasukan password kurang dari 10 karakter maka yang akan keluar notice pada header yang menginstruksikan untuk memasukan karakter lebih dari 10 output dari header dapat dilihat pada gambar di bawah ini:

The image shows a web page with a blue background. In the center is a white rounded rectangle containing a login form. At the top of the form is the title "Login". Below the title is a red error message box that says "do you want to get hacked? 10 or more characters for password!". Underneath the message are two input fields. The first is labeled "User Name" and contains the text "user name". The second is labeled "password" and contains the text "uppercase, lowercase, number and special character is a must". At the bottom right of the form is a dark grey button labeled "Login".

*Gambar 2.10. Header jika user memasukan hanya 10*

Jika kemudian user dapat memasukan semua form yang ada (user name dan password) dengan benar, maka output yang akan dikeluarkan adalah seperti pada gambar di bawah ini:

---

```
login success! welcome to our website ilyas!
```

*Gambar 2.11. Output jika user dapat memasukan semua form dengan sukses*

Untuk form yang dimasukan akan menyesuaikan dengan inputan dari user. Kode yang digunakan dapat dilihat sebagai berikut:

```
}else{  
    echo "login success! welcome to our website $uname!";  
}
```

### **BAB III**




#### **KESIMPULAN**

Form handling dapat digunakan untuk mengambil data yang sudah diinputkan ke dalam form yang kemudian dapat di *run* dengan program yang ada. Dalam pembuatan form handling dapat dilakukan dengan dua cara yaitu dengan metode post dan get. Pada dua metode tersebut memiliki kekurangan serta kelebihan masing-masing. Dalam hal ini, pembuatan form handling menggunakan metode post sendiri memiliki keunggulan yang mana seluruh data yang diinput dikirimkan melalui body request HTTP serta tidak memiliki batasan pengiriman jumlah informasi selain itu metode post juga mendukung fungsi lanjutan seperti untuk input biner multipart jika kemudian ingin di upload ke dalam server. Kelemahan dari form handling menggunakan metode post adalah data yang tidak dapat terbookmark.

## **DAFTAR PUSTAKA**

Praktikum, K. (n.d.). MODUL PRAKTIKUM PEMROGRAMAN WEB I *Jurusan Teknik Informatika Fakultas Teknik Universitas Palangka Raya*.

## LAMPIRAN

 index.php	4/25/2021 10:44 AM	PHP Source File
 login.php	4/25/2021 10:37 AM	PHP Source File
 style.css	4/24/2021 10:26 PM	Cascading Style S...

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <title>MK Pemrograman dan Web App I</title>
5   <link rel="stylesheet" type="text/css" href="style.css">
6   <meta charset="UTF-8">
7   <meta name="viewport" content="width=device-width, initial-scale=1.0">
8 </head>
9 <body>
10   <form action="login.php" method="post">
11     <h2>Login</h2>
12     <?php if (isset($_GET['error'])) { ?>
13       <p class="error"><?php echo $_GET['error']; ?></p>
14     <?php ?>
15     <label>User Name</label>
16     <input type="text" name="uname" placeholder="user name"><br>
17
18     <label>password</label>
19     <input type="password" name="password" placeholder="uppercase, lowercase, number and special character is a must"><br>
20
21     <button type="submit">Login</button>
22   </form>
23 </body>
24 </html>
```



```

body {
  background: #1690A7;
  display: flex;
  justify-content: center;
  align-items: center;
  height: 100vh;
}

* {
  font-family: sans-serif;
  box-sizing: border-box;
}

form {
  width: 500px;
  border: 2px solid #ccc;
  padding: 30px;
  background: #fff;
  border-radius: 15px;
}

h2 {
  text-align: center;
  margin-bottom: 40px;
}

input {
  display: block;
  border: 2px solid;
  width: 95%;
  padding: 10px;
  margin: 10px auto;
  border-radius: 5px;
}

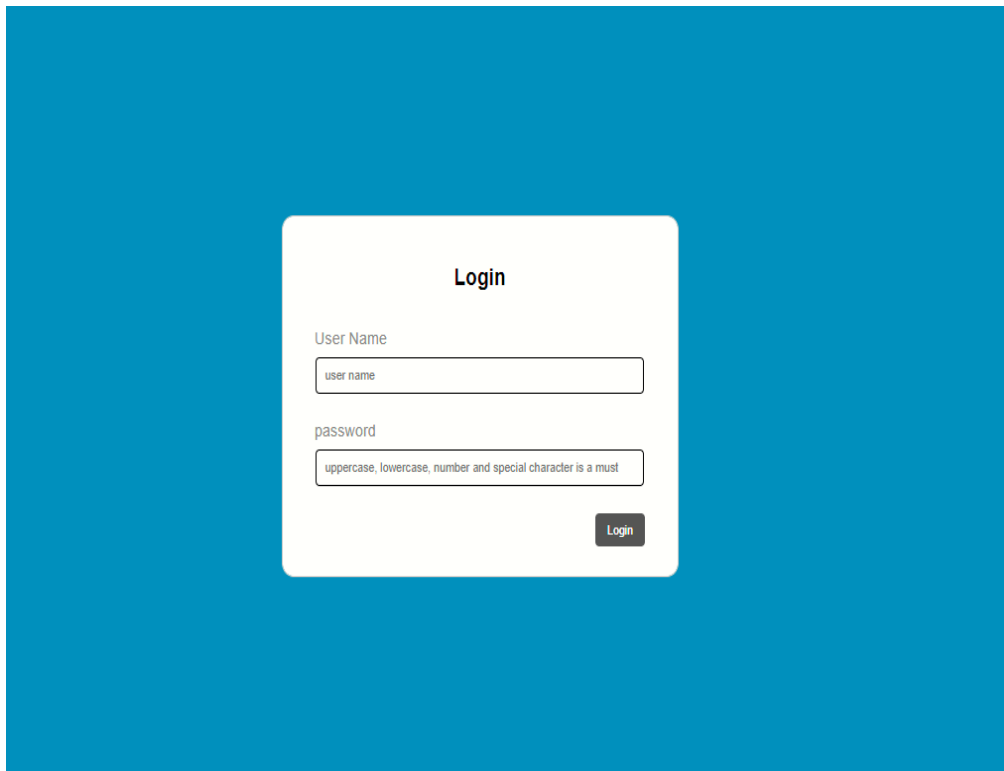
label {
  color: #888;
  font-size: 18px;
  padding: 10px;
}

```

```

42
43 ▼ button {
44   float: right;
45   background: #555;
46   padding: 10px 15px;
47   color: #fff;
48   border-radius: 5px;
49   margin-right: 10px;
50   border: none;
51 }
52
53 button:hover{
54   opacity: .7;
55 }
56
57 ▼ .error {
58   background-color: #F2DEDE;
59   color: #A94442;
60   padding: 10px;
61   width: 95%;
62   border-radius: 5px;
63   margin: 20px auto;
64 }

```



```
1 <?php
2 session_start();
3
4 if (isset($_POST['uname']) && isset($_POST['password'])) {
5     function validate($data){
6         $data = trim($data);
7         $data = stripslashes($data);
8         $data = htmlspecialchars($data);
9         return $data;
10    }
11    $uname = validate($_POST['uname']);
12    $pass = validate($_POST['password']);
13    $user = strlen($uname);
14    $passwords = strlen($pass);
15    if (empty($uname)) {
16        header("Location: index.php?error=user name is required");
17        exit();
18    }else if (empty($pass)){
19        header("Location: index.php?error=password is required");
20        exit();
21    }else if ($user>7){
22        header("Location: index.php?error=username must consist of seven characters!");
23        exit();
24    }else if (!preg_match("/[A-Z]/", $pass)){
25        header("Location: index.php?error=UPPERCASE is a must!");
26        exit();
27    }else if (!preg_match("/[a-z]/", $pass)){
28        header("Location: index.php?error=lowercase is a must!");
29        exit();
30    }else if (!preg_match("/[^\a-zA-Z\d]/", $pass)){
31        header("Location: index.php?error=special characters is a must!");
32        exit();
33    }else if (!preg_match("/[0-9]/", $pass)){
34        header("Location: index.php?error=Number! dont forget to put number ffs!");
35        exit();
36    }else if ($passwords<10){
37        header("Location: index.php?error=do you want to get hacked? 10 or more characters for password!");
38        exit();
39    }else{
40        echo "login success! welcome to our website $uname!";
41    }
42
43    }else{
44        header("Location: index.php");
45        exit();
46    }
47
48    ?>
```

Login

UPPERCASE is a must!

User Name

user name

password

uppercase, lowercase, number and special character is a must

Login

Login

lowercase is a must!

User Name

user name

password

uppercase, lowercase, number and special character is a must

Login

## Login

Number! dont forget to put number ffs!

User Name

user name

password

uppercase, lowercase, number and special character is a must

Login

## Login

Number! dont forget to put number ffs!

User Name

user name

password

uppercase, lowercase, number and special character is a must

Login

Login

special characters is a must!

User Name

user name

password

uppercase, lowercase, number and special character is a must

Login

Login

do you want to get hacked? 10 or more characters for password!

User Name

user name

password

uppercase, lowercase, number and special character is a must

Login

---

login success! welcome to our website ilyas!