

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



**Nama : Oscar Oktorian Almando**  
**NIM : 193020503037**  
**Kelas : A**  
**Modul : II (Form Handling)**

**JURUSAN/PROGRAM STUDI TEKNIK INFORMATIKA**  
**FAKULTAS TEKNIK**  
**UNIVERSITAS PALANGKA RAYA**  
**2021**

## **BAB I**

### **TUJUAN DAN LANDASAN TEORI**

#### **1.1 Tujuan**

- 1.1.1 Mahasiswa mampu membuat handling yang mampu mengolah data dari form HTML.
- 1.1.2 Mahasiswa mampu membuat batasan-batasan untuk menangani inputan dari form HTML.

#### **1.2 Landasan Teori**

##### **1.2.1 PHP Form Handling**

PHP Form Handling merupakan hal penting dalam sebuah website, form handling digunakan untuk mengambil data dari inputan user melalui form HTML (Adhitya 2012).

Vairabel `$_POST` merupakan *Predefined Variable* yang disediakan library PHP. Variabel ini pada umumnya digunakan dalam mengambil suatu nilai yang dikirimkan oleh form dengan atribut `method="post"`.

Semua nilai / informasi yang dikirimkan oleh form yang menggunakan method POST tidak akan terlihat oleh user yang mengakases, dikarenakan informasi yang dikirim akan tidak ditampilkan di Address Bar Web Browser (Adhitya 2012). Selain `$_POST` juga tidak memiliki batasan pada jumlah informasi yang dikirim.

##### **1.2.2 GET vs. POST**

GET dan POST membuat sebuah array (contoh array (kunci => nilai, kunci2 => nilai2, kunci3 => nilai3, ...)) (Jurusan Teknik Informatika 2021). 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 variable yang

dikirimkan ke skrip melalui parameter URL. \$\_POST adalah sebuah array dari variabel yang dikirimkan ke skrip melalui method HTTP POST.

Informasi dikirim dari sebuah form dengan method GET bisa dilihat oleh semua orang (semua nama dan nilai variabel ditampilkan di URL) (Jurusan Teknik Informatika 2021). 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.

### 1.2.3 Membuat Form di HTML

Untuk membuat form di HTML diperlukan tag <form> yang mana pada tag ini memerlukan dua atribut khusus dan wajib digunakan (Neko 2020). Atribut yang disebutkan antara lain :

- a) **Action** – berfungsi untuk menentukan aksi atau tindakan yang akan dilakukan ketika data dikirim. Atribut ini akan mengirim data form ke alamat URL yang telah didefinisikan pada *value* atribut ini. Jadi nilai dari atribut action merupakan alamat URL dari suatu file yang akan memproses data form tersebut.
- b) **method** – digunakan untuk menentukan metode apa yang akan digunakan untuk mengirim data form. Terdapat dua nilai yang perlu diingat untuk menggunakan atribut ini, yaitu POST dan GET.

### 1.2.4 Mengenal Field di HTML

Field merupakan ruas atau bidang tempat yang ada di dalam tag <form> untuk mengisi data (Neko 2020). Salah satu tag yang sering digunakan untuk membuat field di HTML adalah tag <input>.

#### 1.2.4.1 Atribut Type

Berikut ini nilai-nilai yang sering digunakan untuk membuat form, antara lain :

- a) **text** – menghasilkan field untuk mengisi data berupa teks
- b) **email** – menghasilkan field untuk mengisi data khusus email.

- c) password – menghasilkan field untuk mengisi data khusus password atau kata sandi, field ini nantinya hanya akan menampilkan bintang atau bulatan yang berfungsi untuk menutupi nilai asli.
- d) radio – menghasilkan radio button, dengan menggunakan input type ini pengguna akan memilih pilihan yang telah ditentukan yang telah dimasukkan ke input tipe nilai radio.
- e) checkbox – menghasilkan checkbox yang membuat pengguna dapat memilih lebih dari satu pilihan dari banyak daftar pilihan yang disediakan.
- f) number – menghasilkan field untuk mengisi data khusus angka.
- g) date – menghasilkan field untuk mengisi data khusus tanggal.
- h) file – menghasilkan field untuk upload file.
- i) Submit – menghasilkan tombol submit.

#### **1.2.4.2 Atribut Name**

Atribut ini digunakan sebagai pengenalan untuk nama elemen terkait. Karena digunakan sebagai pengenalan suatu elemen maka ia tidak akan menampilkan apapun ketika dijalankan di browser.

#### **1.2.4.3 Atribut Value**

Atribut ini berfungsi untuk memberikan nilai default untuk elemen.

#### **1.2.4.4 Atribut Placeholder**

Atribut placeholder berfungsi untuk memberikan teks bantuan di dalam field. Teks ini akan muncul ketika field dalam keadaan kosong. Jika field mulai diisi maka placeholder akan otomatis disembunyikan, namun akan tampil kembali bila kosong.

#### **1.2.4.5 Atribut Required**

Atribut yang akan menghasilkan syarat supaya elemen terkait wajib diisi atau dilarang untuk dikosongkan.

### **1.2.5 Regular Expression PHP**

Regular Expression, atau biasa disebut Regex, adalah sebuah metode untuk mencari suatu pola dalam sebuah string (Prihandaya 2016). Fungsi yang paling sering digunakan untuk Regex dalam PHP adalah *preg\_match(\$regex, \$string)*, yang mana *\$regex* adalah pola yang akan dicari dan *\$string* adalah variabel yang akan dicari apakah ada pola *\$regex* di dalamnya.

## BAB II PEMBAHASAN

Dalam praktikum modul 2 ini, aplikasi yang digunakan untuk menuliskan sintaks PHP dan HTML adalah aplikasi Komodo Edit. Sintaks yang telah dibuat akan disimpan ke dalam format \*.php dan dieksekusi untuk ditampilkan hasilnya menggunakan aplikasi XAMPP (untuk memproses PHP) dan browser untuk menampilkannya. File yang dibuat adalah tentunya satu file berekstensi .php yang merupakan jawaban tugas praktikum modul 2 ini, dan satu file berekstensi .css yang digunakan untuk mengubah tampilan yang ada pada file .php.

Pembahasan sintaks pada file PHP akan dibahas per bagian, dan berikut ini bagian-bagian sintaks yang telah dibuat.

```
<!DOCTYPE html>
<?php
    Bagian 1
    Bagian 2
    Bagian 3
?>
<html>
    <head>
        Bagian 4
    </head>
    <body>
        <div class="login">
            Bagian 5
        </div>
        <div class="tengah">
            Bagian 6
        </div>
    </body>
</html>
```

## 2.1 Sintaks Bagian 1

Berikut ini merupakan sintaks yang ada pada bagian 1.

```
if($_SERVER["REQUEST_METHOD"]=="POST"){  
    $username = $_POST['username'];  
    $passwd = $_POST['password'];  
} elseif($_SERVER["REQUEST_METHOD"]=="GET"){  
    $username = null;  
    $passwd = null;  
} else {  
    $username = null;  
    $passwd = null;  
}
```

Sintaks di atas merupakan sintaks untuk memeriksa *request method* yang telah digunakan atau pernah diterapkan pada web tersebut. Sintaks untuk memeriksanya menggunakan percabangan *if*, yang mana pada sintaks tersebut dapat dilihat terdapat tiga kondisi.

Kondisi pertama yaitu jika web server mendeteksi *request method*-nya adalah method *post*, maka variabel yang bernama *\$username* akan diisi nilainya berdasarkan dari field yang memiliki atribut nama *username* yang telah dilakukan submit dengan method *post* pada sesi sebelumnya. Kemudian di baris selanjutnya variabel yang bernama *\$passwd* juga akan berisi nilai yang ditampung oleh method *post* dengan atribut nama *password* yang telah dilakukan submit pada sesi sebelumnya. Jadi pada kondisi pertama akan mengeksekusi sintaks jika *method-request* adalah *post*.

Kondisi kedua yaitu jika web server mendeteksi *request method*-nya adalah method *get*, maka variabel *\$username* dan *\$passwd* akan diisi dengan nilai *null*. Kondisi kedua ini nantinya digunakan untuk melakukan ‘refresh’, yaitu mengosongkan data yang diinput tanpa mengakibatkan error dalam penginputan data.

Kondisi ketiga yaitu jika web server tidak mendeteksi *request method* jenis *get* maupun *post*, maka variabel *\$username* dan *\$passw* akan diisi dengan nilai *null*. Kondisi ketiga ini berguna pada saat pertama kali membuka file *.php* yang bersangkutan, agar tidak terjadi error pada bagian form yang nantinya memuat variabel *\$username* dan *\$passw*.

Variabel *\$username* digunakan untuk menampung nilai username, nantinya variabel ini akan diperiksa apakah nilainya telah sesuai kriteria pada tugas praktikum atau tidak.

Variabel *\$passw* digunakan untuk menampung nilai password, nantinya variabel ini akan diperiksa apakah nantinya password yang diinput telah sesuai kriteria atau tidak.

## 2.2 Sintaks Bagian 2

Berikut ini merupakan sintaks yang ada pada bagian 2.

```
function cekPass($passw){
    $cek = true;
    $uppercase = preg_match("/[A-Z]/", $passw);
    $lowercase = preg_match("/[a-z]/", $passw);
    $number = preg_match("/[0-9]/", $passw);
    $symbol = preg_match("/[@ ! # $ % ^ & * ( ? ) < > _ \ | \ ] /", $passw);
    if(!(strlen($passw)<10)){
        if(!$uppercase || !$lowercase || !$number || !$symbol){
            echo "password tidak memenuhi kriteria! <br>";
            if(!$uppercase){
                echo "- Anda perlu menambahkan huruf kapital <br>";
            }
            if(!$lowercase){
                echo "- Anda perlu menambahkan huruf kecil <br>";
            }
            if(!$number){
                echo "- Anda perlu menambahkan karakter angka <br>";
            }
            if(!$symbol){
```



```
        echo "- Anda perlu menambahkan karakter simbol <br>";
    }
    $cek = false;
}
} else {
    echo "Karakter password kurang dari 10! <br>";
    $cek = false;
}
return $cek;
}
```

Sintaks di atas merupakan sintaks dari fungsi *cekPass(\$passw)* yang berfungsi untuk memeriksa apakah nilai yang nantinya diisi pada parameter (yakni password) sudah sesuai kriteria pada tugas praktikum atau tidak.

Adapun kriteria yang harus dimiliki password yakni password harus berjumlah lebih dari 10 karakter, dan password tersebut harus memiliki kombinasi huruf kapital, huruf kecil, angka dan simbol.

Pada bagian baris awal di dalam fungsi terdapat 5 variabel yakni *\$cek* untuk menampung nilai yang akan dikembalikan fungsi (bertipe boolean), *\$suppercase*, *\$lowercase*, *\$number* dan *\$symbol*. Selain variabel *\$cek*, keempat variabel lain menggunakan fungsi *regular expression* yakni *preg\_match(\$regex, \$passw)*.

Setiap variabel akan memeriksa salah satu kriteria khusus yang harus dimiliki variabel parameter *\$passw*. Variabel *\$suppercase* menampung nilai kembalian boolean dari *preg\_match()* untuk memeriksa apakah terdapat karakter huruf kapital di dalam variabel *\$passw* atau tidak.

Variabel *\$lowerrcase* menampung nilai kembalian boolean dari *preg\_match()* untuk memeriksa apakah terdapat karakter huruf kecil di dalam variabel *\$passw* atau tidak.

Variabel *\$number* menampung nilai kembalian boolean dari *preg\_match()* untuk memeriksa apakah terdapat karakter angka di dalam variabel *\$passw* atau tidak.

Variabel *\$number* menampung nilai kembalian boolean dari *preg\_match()* untuk memeriksa apakah terdapat karakter simbol di dalam variabel *\$passw* atau tidak.

Setelah mendeklarasikan variabel beserta nilainya yang akan ditampung, maka selanjutnya terdapat sintaks melakukan pemeriksaan dengan menggunakan percabangan *if*.

Hal yang pertama untuk diperiksa adalah panjang karakter dari parameter *\$passw*, jika memiliki panjang kurang dari 10 karakter maka sintaks akan langsung mengeksekusi baris *else* yang ada pada bagian bawah yang memberikan nilai variabel *\$cek = false*. Jika sesuai, maka di dalam percabangan itu terdapat percabangan lagi untuk memeriksa apakah keempat kriteria (huruf kapital, huruf kecil, angka dan simbol) dimiliki oleh parameter *\$passw* atau tidak. Jika memiliki semua kriteria tersebut, maka variabel *\$cek* akan tetap bernilai *true*. Jika tidak memiliki salah satu kriteria tersebut, maka akan dieksekusi sintaks untuk memeriksa masing-masing kriteria untuk memberikan pesan bahwa kriteria tersebut tidak dimiliki parameter *\$passw*.

Setelah semua dilakukan pemeriksaan, maka baris selanjutnya akan mengembalikan *\$cek* kepada fungsi, sehingga fungsi memiliki nilai *\$cek* dari hasil pemeriksaan *\$passw*.

### 2.3 Sintaks Bagian 3

Berikut ini merupakan sintaks yang ada pada bagian 3.

```
function cekUser($usrname){  
    $cek = true;  
    if(strlen($usrname)>7){  
        echo "Panjang username lebih dari 7  
karakter!";  
        $cek = false;  
    }  
    return $cek;  
}
```

Sintaks di atas merupakan sintaks fungsi *cekUser(\$username)* yang sama seperti fungsi *cekPass(\$passw)* yang berfungsi untuk memeriksa apakah nilai yang nantinya diisi pada parameter (yakni username) sudah sesuai kriteria pada tugas praktikum atau tidak.

Adapun kriteria yang harus dimiliki username yakni harus berjumlah kurang dari 7 karakter. Variabel yang dimiliki dalam fungsi ini hanya *\$cek* yang menampung boolean yang bernilai *true*, yang nantinya akan dikembalikan pada fungsi. Selanjutnya pengecekan menggunakan percabangan *if*, akan diperiksa apakah username tersebut memiliki panjang lebih dari 7 karakter. Jika benar maka akan memunculkan pesan bahwa inputan memiliki panjang lebih dari 7 karakter, dan nilai *\$cek* akan diisi nilai *false*. Selanjutnya variabel *\$cek* akan dikembalikan pada fungsi.

## 2.4 Sintaks Bagian 4

Berikut ini merupakan sintaks yang ada pada bagian 4.

```
<title> Modul 2 | Oscar Oktorian Almando</title>
<link rel="stylesheet" type="text/css" href="style.css">
```

Sintaks ini berada pada bagian `<head>` di struktur HTML, yang mana berisi `<title>` untuk memberi nama atau judul pada bagian tab browser nantinya, dan tag `<link>` yang berisi atribut untuk menghubungkan file saat ini dengan file *style.css* yang bertipe *css*.

## 2.5 Sintaks Bagian 5

Berikut ini merupakan sintaks yang ada pada bagian 5.

```
<div class="login">
  <h1 class="tengah"> SIGN UP </h1>
  <form action="<?php echo $_SERVER['PHP_SELF']; ?>"
    method="post">
    <div>
      <div class="tengah"> Username </div>
      <div class="tengah">
```

```

        <input class="rad" name="username" type="text"
placeholder="Masukkan Username" value="<?php echo $username
?>" required>

    </div>

    <div class="tengah">

        <p class="warn">

            <?php

                if($username != null){

                    if(cekUser($username)){

                        echo "<p class=\"apr\">Input username berhasil </p>";

                    }

                }

            ?> </p>

        </div>

    </div>

    <div>

        <div class="tengah"> Password </div>

        <div class="tengah">

            <input class="rad" name="password" type="text"
placeholder="Masukkan Password" value="<?php echo $passw ?>"
required>

        </div>

        <div class="tengah">

            <p class="warn"> <?php

                if($passw != null){

                    if(cekPass($passw)){

                        echo "<p class=\"apr\">Input password
berhasil </p>";

                    }

                }

            ?> </p>

```

```
</div>  
</form>
```

Pada bagian 5 ini, berada di bagian `<body>` yang ada pada struktur HTML. Pertama sintaks bagian ini dituliskan di antara tag `<div>` yang memiliki atribut kelas “login”, yang mana nantinya mempunyai blok khusus agar tampilannya dapat diatur. Di dalamnya pada baris pertama terdapat baris header `<h1>` yang memunculkan tulisan “*SIGN UP*”

Kemudian dilanjutkan dengan penulisan *form* `<form action="<?php echo $_SERVER['PHP_SELF']; ?>" method="post">`. Form ini memiliki atribut *action* menuju dirinya sendiri (file saat ini), dan atribut *method* berjenis *post*.

Selanjutnya didalamnya dibentuk blok layout `<div>` lagi, yang mana di dalam `<div>` tersebut mempunyai blog layout `<div>` juga dengan atribut kelas=”tengah”. Terdapat 7 blok layout `<div>` dengan kelas yang sama, berikut ini penjelasannya secara berturun-turut.

1) Blok layout pertama

Blok layout `<div>` ini akan memuat tulisan “*Username*”.

2) Blok layout kedua

Blok layout `<div>` ini akan memunculkan tag *input* yaitu :

```
<input class="rad" name="username" type="text"  
placeholder="Masukkan Username" value="<?php echo  
$username ?>" required>.
```

Baris ini mempunyai atribut kelas yang bernama “rad”, atribut *name* dengan nilai “username”, *placeholder* dengan nilai “Masukkan Username”, dan *value* yang nantinya akan menampung variabel \$username. Baris ini nantinya yang akan diinput sebuah username, dan wajib untuk diisi karena telah diberi atribut *required*.

3) Blok layout ketiga

Blok layout ini berisi tag `<p>` yang memiliki kelas “warn”. Di dalam tag `<p>` sendiri berisi sintaks PHP, yang mana akan memanggil fungsi *cekUser()* untuk nantinya digunakan dan

menghasilkan pesan atau tulisan mengenai kesesuaian username dengan kriteria.

4) Blok layout keempat

Blok layout <div> ini akan memuat tulisan “*Password*”.

5) Blok layout kelima

Blok layout <div> ini akan memunculkan tag *input* yaitu :

```
<input class="rad" name="password" type="text"
placeholder="Masukkan Password" value="<?php echo
$password ?>" required>
```

Baris ini mempunyai atribut kelas yang bernama “*rad*”, atribut *name* dengan nilai “*password*”, *placeholder* dengan nilai “Masukkan Password”, dan *value* yang nantinya akan menampung variabel *\$password*. Baris ini nantinya yang akan diinput sebuah password, dan wajib untuk diisi karena telah diberi atribut *required*.

6) Blok layout keenam

Blok layout ini berisi tag <p> yang memiliki kelas “*warn*”. Di dalam tag <p> sendiri berisi sintaks PHP, yang mana akan memanggil fungsi *cekPass()* untuk nantinya digunakan dan menghasilkan pesan atau tulisan mengenai kesesuaian password yang diinput dengan kriteria yang telah ditentukan.

7) Blok layout ketujuh

Blok layout <div> ini akan memunculkan tag *input* yaitu :

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

Baris ini mempunyai atribut kelas yang bernama “*rad*”, atribut *name* dengan nilai “*password*”, atribut *type* bertipe “*submit*” dan *value* yang juga memiliki nilai “*Submit*”. Blok inilah yang nantinya akan mengirimkan informasi jika semua kolom telah terisi.

Pada baris akhir di bagian 5 ini tersisa sintaks untuk menutup tag <div> dan <form> yang telah dibuat.

## 2.6 Sintaks Bagian 6

Berikut ini merupakan sintaks yang ada pada bagian 6.

```
<div class="tengah">

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

        <input class="rad" name="Submit" type="submit"
        value="Refresh">

    </form>

</div>
```

Sintaks ini berada pada bagian `<head>` di struktur HTML, dan berada tepat di bawah sintaks **bagian 5**. Sintaks ini terdiri atas tag `<div>` yang memiliki `class="tengah"`, yang mana di dalamnya berisi tag `<form>` dengan atribut `method` bernilai `get`. Di dalam `<form>` ini terdapat tag `<input>` yang mana memiliki atribut `type` bernilai `submit`, dan value bernilai `"Refresh"`. Sintaks **bagian 6** ini dibuat dengan tujuan untuk melakukan refresh, yang mana akan mengosongkan field yang telah diisi, dan menghilangkan pesan di bawah field tersebut.

Setelah semua bagian dijabarkan, berikut ini merupakan sintaks dari file .php untuk tugas praktikum secara keseluruhan.

```
<!DOCTYPE html>

<?php

    if($_SERVER["REQUEST_METHOD"]=="POST"){
        $username = $_POST['username'];
        $passw = $_POST['password'];
    } elseif($_SERVER["REQUEST_METHOD"]=="GET"){
        $username = null;
        $passw = null;
    } else {
        $username = null;
        $passw = null;
    }

    function cekPass($passw){
```

```

$cek = true;
$uppercase = preg_match("/[A-Z]/", $passw);
$lowercase = preg_match("/[a-z]/", $passw);
$number    = preg_match("/[0-9]/", $passw);
$symbol     = preg_match("/[@ ! # $ % ^ & * ( ? ) < > _ \ \ ] \[
]/", $passw);
if(!(strlen($passw)<10)){
    if(!$uppercase || !$lowercase || !$number || !$symbol){
        echo "password tidak memenuhi kriteria! <br>";
        if(!$uppercase){
            echo "- Anda perlu menambahkan huruf
kapital <br>";
        }
        if(!$lowercase){
            echo "- Anda perlu menambahkan huruf kecil
<br>";
        }
        if(!$number){
            echo "- Anda perlu menambahkan karakter
angka <br>";
        }
        if(!$symbol){
            echo "- Anda perlu menambahkan karakter
simbol <br>";
        }
        $cek = false;
    }
} else {
    echo "Karakter password kurang dari 10! <br>";
    $cek = false;
}
return $cek;
}

function cekUser($usname){

```



```

$ccek = true;
if(strlen($susname)>7){
    echo "Panjang username lebih dari 7 karakter!";
    $cek = false;
}
return $cek;
}
?>
<html>
<head>
<title> Modul 2 | Oscar Oktorian Almando</title>
<link rel="stylesheet" type="text/css" href="style.css">
</head>
<body>
<div class="login">
    <h1 class="tengah"> SIGN UP </h1>
    <form action="<?php echo $_SERVER['PHP_SELF']; ?>"
method="post">
        <div>
            <div class="tengah"> Username </div>
            <div class="tengah">
                <input class="rad" name="username"
type="text" placeholder="Masukkan Username" value="<?php echo $username ?>"
required>
            </div>
            <div class="tengah">
                <p class="warn">
                    <?php
                        if($username != null){
                            if(cekUser($username)){
                                echo "
class="\apr\">Input username berhasil </p>";
                            }
                        }
                    }
                </p>
            </div>
        </div>
    </form>
</div>
</body>
</html>

```

```

        ?> </p>

    </div>

</div>

<div>

    <div class="tengah"> Password </div>

    <div class="tengah">

        <input    class="rad"    name="password"
type="text" placeholder="Masukkan Password" value="<?php echo $passw ?>"
required>

    </div>

    <div class="tengah">

        <p class="warn"> <?php
                                if($passw != null){
                                    if(cekPass($passw)){
                                        echo
                                " <p
class=\"apr\">Input password berhasil </p>;

                                    }

                                }

        ?> </p>

    </div>

</div>

    <div class="tengah">

        <input    class="rad"    name="Submit"
type="submit" value="Submit">

    </div>

</form>

    <div class="tengah">

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

            <input class="rad" name="Submit" type="submit"
value="Refresh">

        </form>

    </div>

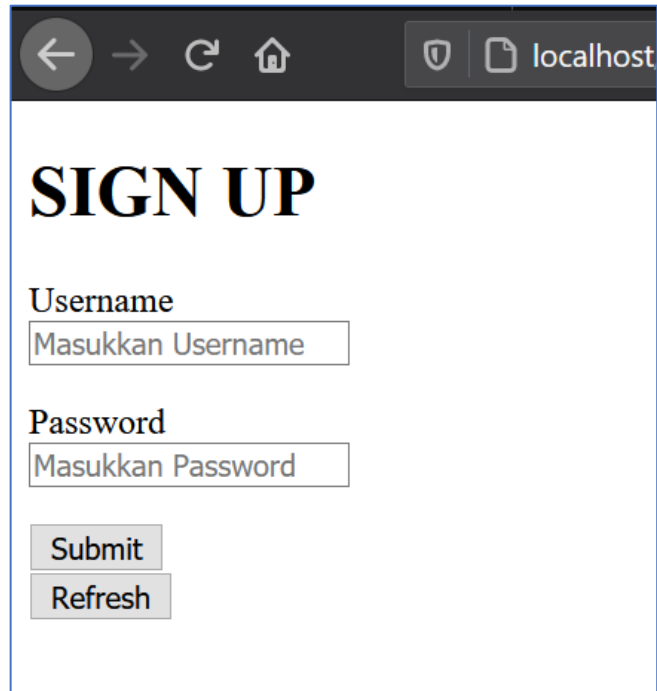
</div>

</body>

```

```
</html>
```

Berikut ini merupakan tampilan website dari file sintaks ini (jika tidak menggunakan CSS).



The image shows a web browser window with a dark address bar displaying 'localhost'. The main content area has a white background. At the top, the text 'SIGN UP' is displayed in a large, bold, black serif font. Below this, there are two labels: 'Username' and 'Password', both in a black serif font. Under the 'Username' label is a text input field with a light gray border and the placeholder text 'Masukkan Username'. Under the 'Password' label is a similar text input field with the placeholder text 'Masukkan Password'. At the bottom of the form are two buttons: 'Submit' and 'Refresh', both with a light gray background and black text.

**Gambar 2. 1** File PHP yang dijalankan di browser (tanpa menggunakan CSS).

Untuk menambah dan mengatur tampilan dari web yang dibuat, maka diperlukan CSS. Berikut ini merupakan sintaks dari CSS yang mana file-nya terpisah dari file saat ini.

```
body{  
    height: 420px;  
    background-color: lightblue;  
}  
  
div.login{  
    padding : 30px 20px 60px 20px;  
    margin: 5% 20% 10% 20%;  
    border: 1px solid white;  
    border-radius: 20px;  
    background-color: white;
```

```
        box-shadow: 3px 3px 10px #000000;  
    }
```

```
div.tengah{  
    margin-top: 6px;  
    margin-bottom: 6px;  
    text-align: center;  
    font-weight: bold;  
}
```

```
h1.tengah{  
    text-align: center;  
}
```

```
.warn{  
    color: #ff0000;  
    font-style: italic;  
    font-weight: lighter;  
}
```

```
.apr{  
    color: green;  
    font-style: italic;  
    font-weight: lighter;  
}
```

```
.rad{  
    padding : 5px 5px 5px 5px;  
    border-radius: 5px;  
    text-align: center;  
    border: 2px solid black;  
}
```

Berikut ini merupakan tampilan website dari file sintaks ini (jika menggunakan CSS).



**SIGN UP**

**Username**  
Masukkan Username

**Password**  
Masukkan Password

Submit  
Refresh

**Gambar 2. 2** Tampilan Web dengan CSS.



**SIGN UP**

**Username**  
Masukkan Username

**Password**  
Masukkan Password

Submit  
Refresh

**Gambar 2. 3** Tampilan Web, jika melakukan klik 'Submit' tanpa mengisi *Username* dan *Password*.

Pada **Gambar 2.3** terlihat bahwa field tersebut pada bagian pingirnya berwarna merah setelah klik submit namun dengan data yang masih kosong. Hal ini karena disematkannya atribut *required* yang ada pada sintaks **Bagian 5**, yang mana berfungsi untuk ‘mengharuskan’ pengguna untuk mengisi field tersebut sebelum dilakukan submit.

The screenshot shows a web form titled "SIGN UP" with a light blue background and a white form container. The form has two input fields: "Username" with the value "jikamaka" and "Password" with the value "jikamakaaku". Below the username field, a red error message reads "Panjang username lebih dari 7 karakter!". Below the password field, a red error message reads "password tidak memenuhi kriteria!" followed by three bullet points: "- Anda perlu menambahkan huruf kapital", "- Anda perlu menambahkan karakter angka", and "- Anda perlu menambahkan karakter simbol". At the bottom of the form are "Submit" and "Refresh" buttons.

**Gambar 2. 4** Tampilan Web setelah melakukan submit dengan nilai *Username* dan *Password* yang tidak sesuai kriteria.

Pada **Gambar 2.4** akan terlihat penggunaan 6 blok field <div> yang telah dituliskan. Tulisan dan field tersebut tidak saling berdempetan atau bertabrakan karena telah diatur masing-masing di dalam <div> dan juga ditambah pengaturan CSS.

The screenshot shows the same "SIGN UP" web form. The "Username" field now contains "jikamax" and has a green success message "Input username berhasil" below it. The "Password" field still contains "jikamakaaku" and has the same red error message as in the previous image: "password tidak memenuhi kriteria!" followed by the three bullet points. The "Submit" and "Refresh" buttons remain at the bottom.

**Gambar 2. 5** Tampilan Web setelah melakukan submit dengan nilai *Username* yang benar namun dengan *Password* yang tidak sesuai kriteria.

The screenshot shows a web form titled "SIGN UP" on a light blue background. The form is a white rounded rectangle. It contains two input fields: "Username" with the value "jikamaka" and "Password" with the value "jikaMaka12@". Below the Username field, a red error message reads "Panjang username lebih dari 7 karakter!". Below the Password field, a green success message reads "Input password berhasil". At the bottom of the form are two buttons: "Submit" and "Refresh".

**Gambar 2. 6** Tampilan Web setelah melakukan submit dengan nilai *Username* yang tidak sesuai kriteria namun dengan *Password* yang sesuai kriteria.

This screenshot shows the same "SIGN UP" form as in Gambar 2. 6, but with updated validation. The Username field now contains "jikamax" and has a green success message "Input username berhasil" below it. The Password field remains "jikaMaka12@" with the same green success message "Input password berhasil". The "Submit" and "Refresh" buttons are still present at the bottom.

**Gambar 2. 7** Tampilan Web setelah melakukan submit dengan nilai *Username* dan *Password* yang sesuai kriteria.

The screenshot shows the "SIGN UP" form in its initial state. The Username field has the placeholder text "Masukkan Username" and the Password field has "Masukkan Password". The "Submit" and "Refresh" buttons are located at the bottom of the form.

**Gambar 2. 8** Tampilan Web yang kembali seperti semula saat dilakukan klik *Refresh*.

Pada **Gambar 2.8** akan terlihat tampilan web yang kembali seperti saat file pertama dibuka atau dijalankan di browser. Karena pada saat melakukan klik tombol *Refresh*, maka akan mengirim informasi bahwa *method* yang digunakan berjenis *get* sehingga sintaks akan mengeksekusi percabangan *if* di bagian awal sintaks yang memenuhi kondisi tersebut, sehingga memberi nilai pada variabel *\$username* dan *\$passw* menjadi *null*.



### **BAB III**

### **KESIMPULAN**

Berdasarkan hasil dari tugas praktikum yang dikerjakan, hal yang dapat disimpulkan dari praktikum modul 2 ini adalah form handling digunakan untuk mengambil data yang telah diinput, dan nantinya diproses agar bisa menentukan langkah berikutnya yang akan dieksekusi atau dijalankan. Form handling biasanya menggunakan *method GET* dan *POST*, yang masing-masing memiliki kegunaan. Method ini selalu digunakan bersamaan dengan tag `<form>` HTML yang berisi tag `<input>` dengan tipe berbeda untuk mengumpulkan data secara presisi.

## **DAFTAR PUSTAKA**

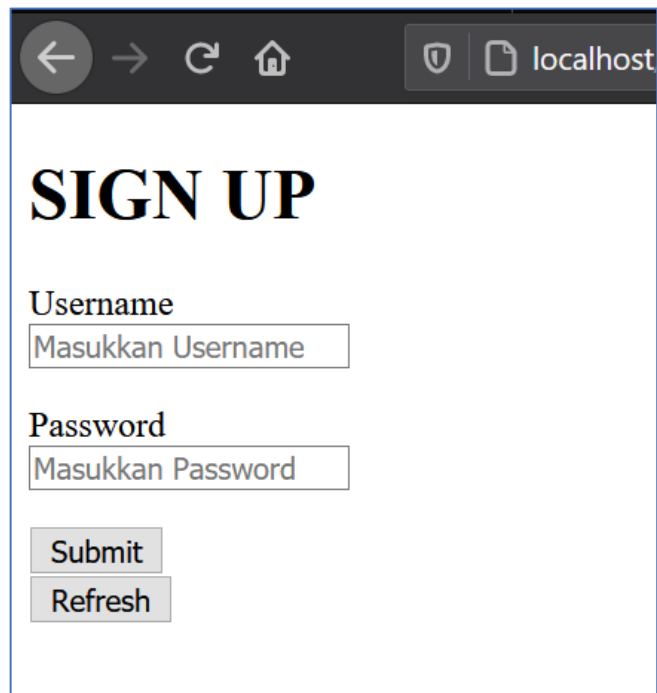
Adhitya, Alfa. 2012. "Lab-Informatika | PHP Form Handling." <https://www.lab-informatika.com/php-form-handling> (April 4, 2021).

Jurusan Teknik Informatika. 2021. "MODUL PRAKTIKUM PEMROGRAMAN WEB I."

Neko, Kuro. 2020. "Membuat Form Di HTML | Kopiding.In." <https://kopiding.in/form-html/> (April 3, 2021).

Prihandaya, Robby. 2016. "Berkenalan Dan Memahami Regular Expression (Regex) Di PHP - PHPMU.Com - Mudahnya Belajar Coding." <https://phpmu.com/berkenalan-dan-memahami-regular-expression-regex-di-php/> (April 3, 2021).

## LAMPIRAN



A screenshot of a web browser window displaying a 'SIGN UP' form. The browser's address bar shows 'localhost'. The form is titled 'SIGN UP' in a large, bold, serif font. Below the title, there are two input fields: 'Username' with the placeholder text 'Masukkan Username' and 'Password' with the placeholder text 'Masukkan Password'. At the bottom of the form, there are two buttons: 'Submit' and 'Refresh'. The form is styled with default browser defaults, lacking any custom CSS.

Gambar 2. 1 File PHP yang dijalankan di browser (tanpa menggunakan CSS).



A screenshot of a web browser window displaying a 'SIGN UP' form with CSS styling. The form is centered on a light blue background. The form itself is a white rounded rectangle with a subtle drop shadow. The title 'SIGN UP' is in a bold, sans-serif font. Below the title, the labels 'Username' and 'Password' are in a smaller, bold, sans-serif font. The input fields have a light gray border and placeholder text 'Masukkan Username' and 'Masukkan Password'. The 'Submit' and 'Refresh' buttons are small, rectangular, and have a light gray border. The overall design is clean and modern.

Gambar 2. 2 Tampilan Web dengan CSS.

A screenshot of a web application's 'SIGN UP' form. The form is centered on a light blue background. It has a white background with a subtle shadow. The title 'SIGN UP' is at the top in bold. Below it are two input fields: 'Username' with a placeholder 'Masukkan Username' and 'Password' with a placeholder 'Masukkan Password'. At the bottom are two buttons: 'Submit' and 'Refresh'.

Gambar 2. 3 Tampilan Web, jika melakukan klik 'Submit' tanpa mengisi Username dan Password.

A screenshot of the 'SIGN UP' form after a submission attempt. The 'Username' field contains 'jikamaka' and has a red error message below it: 'Panjang username lebih dari 7 karakter!'. The 'Password' field contains 'jikamakaaku' and has a red error message below it: 'password tidak memenuhi kriteria!'. Below the password error are three lines of red text: '- Anda perlu menambahkan huruf kapital', '- Anda perlu menambahkan karakter angka', and '- Anda perlu menambahkan karakter simbol'. The 'Submit' and 'Refresh' buttons are still visible at the bottom.

Gambar 2. 4 Tampilan Web setelah melakukan submit dengan nilai Username dan Password yang tidak sesuai kriteria.

A screenshot of the 'SIGN UP' form after a submission attempt. The 'Username' field contains 'jikamax' and has a green success message below it: 'Input username berhasil'. The 'Password' field contains 'jikamakaaku' and has a red error message below it: 'password tidak memenuhi kriteria!'. Below the password error are three lines of red text: '- Anda perlu menambahkan huruf kapital', '- Anda perlu menambahkan karakter angka', and '- Anda perlu menambahkan karakter simbol'. The 'Submit' and 'Refresh' buttons are still visible at the bottom.

Gambar 2. 5 Tampilan Web setelah melakukan submit dengan nilai Username yang benar namun dengan Password yang tidak sesuai kriteria.

The screenshot shows a web form titled "SIGN UP" with two input fields: "Username" and "Password". The "Username" field contains the text "jikamaka" and has a red error message below it: "Panjang username lebih dari 7 karakter!". The "Password" field contains the text "jikaMaka12@" and has a green success message below it: "Input password berhasil". Below the fields are two buttons: "Submit" and "Refresh".

Gambar 2. 6 Tampilan Web setelah melakukan submit dengan nilai Username yang tidak sesuai kriteria namun dengan Password yang sesuai kriteria.

The screenshot shows the same "SIGN UP" form. The "Username" field now contains "jikamax" and has a green success message below it: "Input username berhasil". The "Password" field still contains "jikaMaka12@" and has the same green success message: "Input password berhasil". The "Submit" and "Refresh" buttons are still present.

Gambar 2. 7 Tampilan Web setelah melakukan submit dengan nilai Username dan Password yang sesuai kriteria.

The screenshot shows the "SIGN UP" form in its initial state. The "Username" field has the placeholder text "Masukkan Username" and the "Password" field has the placeholder text "Masukkan Password". The "Submit" and "Refresh" buttons are still present.

Gambar 2. 8 Tampilan Web yang kembali seperti semula saat dilakukan klik Refresh.