LAPORAN PRAKTIKUM

PEMROGRAMAN WEB & MOBILE

NAMA : **AHMAD FALDHI YUDIANTO**

NIM : **E1E118064**

KELAS : **C**

MODUL : **II**

JURUSAN TEKNIK INFORMATIKA

FAKULTAS TEKNIK

UNIVERSITAS PALANGKA RAYA

2021

BAB I
LANDASAN TEORI

1.1 Tujuan Praktikum

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>
```

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

```
<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 Randi dan email diinputkan dengan abil@mail.com maka output yang akan tampil adalah sebagai berikut : Welcome Budi Your email address is abil@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 1.4 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 formcontrol 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 sebuaharray dari variabel yang dikirimkan ke skrip melalui parameter URL sedangkan $_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.

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!

Gambar 1.5 Validasi

Form HTML yang akan kita gunakan pada modul ini, mengandung bermacammacam field input, misalnya text field yang harus diisi dan textfield 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 Rule Validasi

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">
Email: <input type="text"name="email">
Website: <input type="text" name="website">
Comment: <textarea name="comment" rows="7" cols="50"><
/textarea>
```

Radio Button

Field jenis kelamin adalah radio button yaitu sebagai berikut:

```
Gender:
<input type="radio" name="gender" value="female">Femal
e
<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 echohtmlspecialchars
($_SERVER["PHP_SELF"]);? >">
```

Ketika form disubmit, data pada form dikirim dengan method “post”. 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 < dan >. 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 “form.php”, danform 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//form.php/%22%3E%3Cscript%3Ealert('hacked')%3C/scr
ipt %3E

yang jika ditranslasikan akan menjadi :

```
<form method="post" action="test_form.php/"><script>al
ert('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”.

Menghindari penyalahgunaan $_SERVER[“PHP_SELF”] dengan cara 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;&l
t;script&gt;alert('hacked')&lt;/script&gt;">
```

dengan cara ini, percobaan penyalahgunaan akan gagal.Memvalidasi data Form dengan PHP

Hal pertama yang akan kita lakukan adalah memasukkan semua variabel melalui fungsi htmlspecialchars(). Kemudian ada juga dua hal ketika user melakukan submit form :

1. Membuang karakter-karakter yang tidak dibutuhkan (seperti spasi extra, tab extra, dan baris baru yang ektra) dari data input user (dengan fungsi trim()).
2. Membuang backslash (\) tatu garis miring dari data input user (dengan fungsi stripslashed()).

Langkah berikutnya adalah membuat fungsi yang akan melakukan pemeriksaan kebenaran data yang diinputkan oleh user. Contohnya adalah sebagai berikut :

```php
<?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 PHP

Ingat bahwa pada permulaan skrip, adalah pemeriksaan apakahform 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 disimpandalam variabel error yang berbeda, dan jika tidak kosong, ia akan mengirim data input user melalui fungsi test_input() :

```php
<?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 1.8 PHP

```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 1.9 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 1.10 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 1.11 PHP

Validasi Nama

Kode berikut menunjukkan cara sederhana untuk memeriksa apakah field nama hanya mengandung huruf dan spasi. Jika nilai dari namatidak 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 1.12 Validasi Nama

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

Gambar 1.14 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>.

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 1.14 Validasi URL

BAB II
PEMBAHASAN

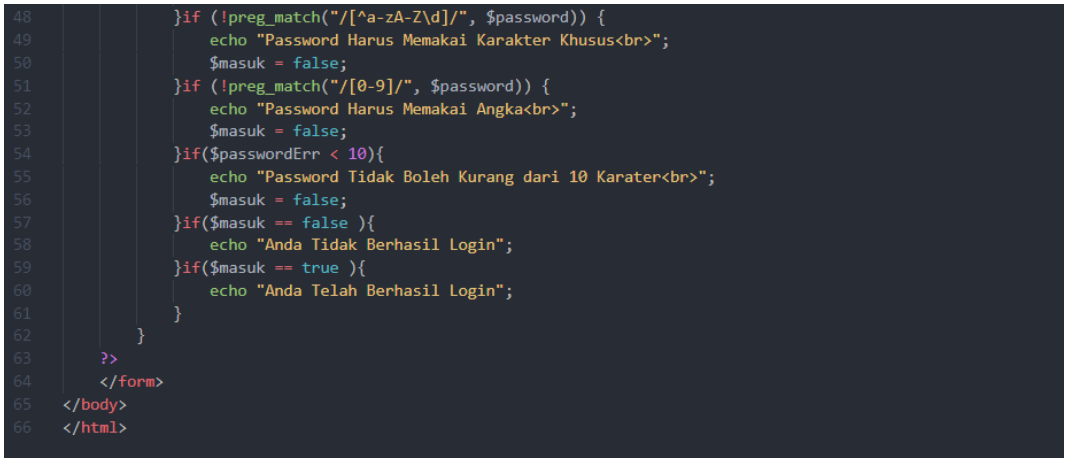
Buatlah program web untuk mengolah nama-nama anggota keluarga anda dengan kriteria sebagai berikut :

1. Username yang dibutuhkan tidak boleh lebih dari tujuh karakter.
2. Password yang diinputkan harus terdiri dari huruf kapital, huruf kecil, angka dan karakter khusus.
3. Jumlah karakter password tidak boleh kurang dari sepuluh karakter.

Source code untuk melakukan perintah di atas, terdapat dibawah ini :

```php
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Tugas Modul 2</title>
</head>
<body>
    <h1>Halaman Login</h1>
    <form method="POST" action="<?php echo htmlspecialchars($_SERVER["PHP_SELF"])?>">
        <table>
            <tr>
                <td>Username</td>
                <td>:</td>
                <td><input type="text" name="username" id="username" placeholder="Masukkan Username"></td>
            </tr>
            <tr>
                <td>Password</td>
                <td>:</td>
                <td><input type="pass" name="password" id="password" placeholder="Masukkan Password"></td>
            </tr>
            <tr>
                <td><button name="btnSubmit" type="submit">Submit</button></td>
            </tr>
        </table>
    <?php
        if($_SERVER["REQUEST_METHOD"]=="POST"){
            $username = $_REQUEST["username"];
            $usernameErr = strlen($_REQUEST["username"]);
            $password = $_REQUEST["password"];
            $passwordErr = strlen($_REQUEST["password"]);
            $masuk = true;
            if ($usernameErr > 7){
                echo "Username Tidak Diperkenankan Lebih dari 7 Karakter<br>";
                $masuk = false;
            }if ($usernameErr == 0){
                echo "Username Tidak Boleh Kosong<br>";
                $masuk = false;
            }if ($passwordErr == 0){
                echo "Password Tidak Boleh Kosong<br>";
                $masuk = false;
            }if (!preg_match("/[A-Z]/", $password) ) {
                echo "Password Harus Memakai Huruf Kapital<br>";
                $masuk = false;
            }if (!preg_match("/[a-z]/", $password)) {
                echo "Password Harus Memakai Huruf Kecil<br>";
                $masuk = false;
```

Gambar 2.1 Source Code Bagian 1



Gambar 2.2 Source Code Bagian 2

Pada bagian pertama sampai akhir yang dimana perintah tersebut meminta menginput username dan password dengan kriteria tertentu. Pertama buat deklarasi variabel untuk menangkap data dari form dengan $username dan $password dengan perintah $_REQUEST[“username”] dan $_REQUEST[“password”] yang menangkap nilai berdasarkan id dari form. Selanjutnya buat variabel untuk menghitung jumlah huruf dari username dan password dengan kode berikut:

```
$username_count = strlen($username);
$password_count = strlen($password);
```

Setelah itu buat kriteria dengan logika if untuk mengecek sesuai kriteria yang sudah diperintahkan, syntax nya sebagai berikut :

```
if ($usernameErr > 7){
      echo "Username Tidak Diperkenankan Lebih dari 7 Karakter<br>
";
      $masuk = false;
}if ($usernameErr == 0){
      echo "Username Tidak Boleh Kosong<br>";
      $masuk = false;
}if ($passwordErr == 0){
      echo "Password Tidak Boleh Kosong<br>";
      $masuk = false;
}if (!preg_match("/[A-Z]/", $password) ) {
      echo "Password Harus Memakai Huruf Kapital<br>";
      $masuk = false;
}if (!preg_match("/[a-z]/", $password)) {
      echo "Password Harus Memakai Huruf Kecil<br>";
      $masuk = false;
```

```
}if (!preg_match("/[^a-zA-Z\d]/", $password)) {
      echo "Password Harus Memakai Karakter Khusus<br>";
      $masuk = false;
}if (!preg_match("/[0-9]/", $password)) {
      echo "Password Harus Memakai Angka<br>";
      $masuk = false;
}if($passwordErr < 10){
      echo "Password Tidak Boleh Kurang dari 10 Karater<br>";
      $masuk = false;
}
```

Dan terakhir ketika semua kondisi t erpenuhi, buat sebuah variabel $masuk dengan nilai false yang ketika semua kondisi benar, maka akan muncul kata berhasil dengan kode berikut :

```
if($masuk == false ){
      echo "Anda Tidak Berhasil Login";
}if($masuk == true ){
      echo "Anda Telah Berhasil Login";
```

Adapun hasil keluaran dari keseluruhan source code yang telah dituliskan adalah sebagai berikut :

Gambar 2.3 Keluaran Source Code

KESIMPULAN

Berbagai element terdapat di PHP untuk membuat sesuatu agar hal tersebut itu dapat melakukan proses seperti yang diinginkan, salah satunya untuk username yang dinputkan tidak boleh lebih dari tujuh karakter menggunakan sebuah metode yang bernama Regex (Reguler Expression) yang mempunyai artian untuk mengenali atau mendeteksi suatu pola tertentu pada suatu string yang telah diinputkan sesorang.

DAFTAR PUSTAKA

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

Huda, Nuru. “PHP: Belajar Reguler Expression”. 13 Maret 2020. https://jagongoding.com/web/php/menengah/regular-expression/ (Di akses 11 April 2020).

LAMPIRAN

```php
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Tugas Modul 2</title>
</head>
<body>
    <h1>Halaman Login</h1>
    <form method="POST" action="<?php echo htmlspecialchars($_SERVER["PHP_SELF"])?>">
        <table>
            <tr>
                <td>Username</td>
                <td>:</td>
                <td><input type="text" name="username" id="username" placeholder="Masukkan Username"></td>
            </tr>
            <tr>
                <td>Password</td>
                <td>:</td>
                <td><input type="pass" name="password" id="password" placeholder="Masukkan Password"></td>
            </tr>
            <tr>
                <td><button name="btnSubmit" type="submit">Submit</button></td>
            </tr>
        </table>
    <?php
        if($_SERVER["REQUEST_METHOD"]=="POST"){
            $username = $_REQUEST["username"];
            $usernameErr = strlen($_REQUEST["username"]);
            $password = $_REQUEST["password"];
            $passwordErr = strlen($_REQUEST["password"]);
            $masuk = true;
            if ($usernameErr > 7){
                echo "Username Tidak Diperkenankan Lebih dari 7 Karakter<br>";
                $masuk = false;
            }if ($usernameErr == 0){
                echo "Username Tidak Boleh Kosong<br>";
                $masuk = false;
            }if ($passwordErr == 0){
                echo "Password Tidak Boleh Kosong<br>";
                $masuk = false;
            }if (!preg_match("/[A-Z]/", $password) ) {
                echo "Password Harus Memakai Huruf Kapital<br>";
                $masuk = false;
            }if (!preg_match("/[a-z]/", $password)) {
                echo "Password Harus Memakai Huruf Kecil<br>";
                $masuk = false;
```

Lampiran 1

```php
            }if (!preg_match("/[^a-zA-Z\d]/", $password)) {
                echo "Password Harus Memakai Karakter Khusus<br>";
                $masuk = false;
            }if (!preg_match("/[0-9]/", $password)) {
                echo "Password Harus Memakai Angka<br>";
                $masuk = false;
            }if($passwordErr < 10){
                echo "Password Tidak Boleh Kurang dari 10 Karater<br>";
                $masuk = false;
            }if($masuk == false ){
                echo "Anda Tidak Berhasil Login";
            }if($masuk == true ){
                echo "Anda Telah Berhasil Login";
            }
        }
    ?>
    </form>
</body>
</html>
```

Lampiran 2


Tugas Modul 2
localhost/webmobile/Tugas2/index.php
Halaman Login
Username : admin
Password : A11aaaaasdsadsa$
Submit
Anda Telah Berhasil Login

Lampiran 3