Praktikum Pemrograman Web Minggu 6 – PHP Forms

Dimas Rizky H.P. – 2110141011 – 3 D4 IT A

***PHP FORM***

Form pada PHP memungkinkan browser untuk mengirimkan suatu data atau informasi, ke sebuah fungsi PHP pada saat aplikasi web tersebut dalam posisi runtime. Seperti yang diketahui, skrip PHP hanya bisa dijalankan di sisi server, sehingga untuk menjalankan suatu fungsi dibutuhkan protokol/metode khusus untuk menjalankannya pada saat aplikasi runtime. Form banyak digunakan untuk membuat web yang dinamis dan memiliki interaksi yang imersif antara client dan servernya.

Data yang dikirim oleh form akan disimpan pada variabel superglobal $\_GET atau $\_POST tergantung metode mana yang akan digunakan untuk menyimpan dan mengirimkan data/informasi ke sebuah fungsi.

Pada metode GET, pengiriman data menggunakan URI, sehingga data yang terkirim dapat terlihat pada URL bar, sedangkan metode POST menyimpan dan mengirim data menggunakan method HTTP POST, yang mana dikirim secara secure dan tidak akan tampak ke di user interface.

Ruang lingkup dari PHP form juga ada pada sisi validasi inputnya. Validasi input pada PHP form secara default sudah tersedia beberapa validasi, contohnya adalah input email, password, website, comment, number, string dan lainnya. Form validation sangat diperlukan agar data yang dikirimkan memiliki format yang sama dengan parameter yang diterima oleh fungsi PHP yang dituju.

Form seringkali menjadi salah satu kelemahan yang ada pada suatu web server. PHP menyediakan beberapa method untuk melindungi input dari serangan XSS (Cross Site Scripting) ataupun SQL Injection.

***PHP Form Handling***

* Contoh HTML Form

Form pada HTML merupakan interface antara user dan fungsi PHP yang dapat melakukan sesuatu pada data yang diberikan oleh user. Tag HTML yang digunakan adalah *<form></form>* dengan atribut *action* berisi file skrip/method PHP yang akan dijalankan, selain itu atribut *method* juga perlu dispesifikan untuk mendefinisikan metode apa yang akan digunakan untuk mengirimkan sebuah data

<div class="container-fluid">

<div class="card form-satu">

<div class="card-header card-inverse card-primary mb-3">

HTML FORM

</div>

<div class="card-block">

<blockquote class="card-blockquote">

<form action="welcome.php" method="post">

<div class="form-group">

<label **for**="inputName">Nama</label>

<input type="text" name="nama" class="form-control" id="inputName">

</div>

<div class="form-group">

<label **for**="inputEmail">Email</label>

<input type="email" name="email" class="form-control" id="inputEmail">

</div>

<button type="submit" class="btn btn-primary">Submit</button>

</form>

</blockquote>

</div>

</div>

<br>

</div>

<BOX CODE 1 Html Form>

Ketika user melakukan klik pada button submit, data yang ada pada form diproses menggunakan file php yang sudah ditentukan pada atribut action, dan data tersebut dikirimkan menggunakan POST method.

File PHP yang meng-handle data tersebut adalah seperti yang ada pada *Box Code 2*, yang mana akan melakukan cetak nama dan email sesuai dengan format yang ada. Outputnya dapat dilihat pada *gambar 1.0*

<**div** class="container-fluid">

<**div** class="card form-satu">

<**div** class="card-header card-inverse card-primary mb-3">

WELCOME PHP SCRIPT

</**div**>

<**div** class="card-block">

<**blockquote** class="card-blockquote">

<**h3**>Selamat Datang, <?php **echo** $\_POST["nama"]; ?></**h3**>

<**h4**>Alamat email kamu adalah : <?php **echo** $\_POST["email"]; ?></**h4**>

</**blockquote**>

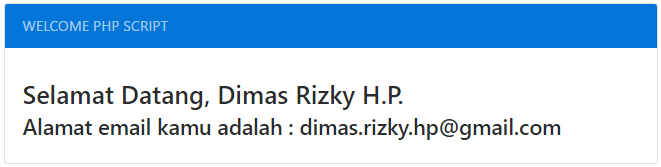
</**div**>

</**div**>

<**br**>

</**div**>

*<BOX CODE 2 PHP Code POST>*

**

*Gambar 1.0. Output form PHP*

Form tersebut juga dapat dikirimkan menggunakan method GET sebagai metode pengiriman datanya,

Dengan menggunakan method GET, data disimpan dalam sebuah array superglobal $\_GET, lalu dikirimkan melalui URI, sehingga tampak dalam URL Bar seperti pada *gambar 1.1*. Output dari kedua method tetap sama hanya metode pengirimannya yang berbeda.



Gambar 1.1 URL pada method GET

Untuk mencetak data yang dikirimkan melalui metode GET, digunakan baris code untuk mengakses array $\_GET.

<**div** class="container-fluid">

<**div** class="card form-satu">

<**div** class="card-header card-inverse card-primary mb-3">

WELCOME PHP SCRIPT

</**div**>

<**div** class="card-block">

<**blockquote** class="card-blockquote">

<**h3**>Selamat Datang, <?php **echo** $\_GET["nama"]; ?></**h3**>

<**h4**>Alamat email kamu adalah : <?php **echo** $\_GET["email"]; ?></**h4**>

</**blockquote**>

</**div**>

</**div**>

<**br**>

</**div**>

*<BOX CODE 2 PHP Code GET>*

* GET vs. POST

GET dan POST keduanya akan membuat key-value array. Array ini berisi pasangan key dan value, dimana key sam dengan atribut *name* pada form HTML, sedangkan value merupakan input dari user.

GET dan POST akan menyimpan array itu pada variabel $\_GET dan $\_POST. Variabel ini merupakan variabel superglobal, yang berarti dapat diakses kapanpun dan dimanapun tidak bergantung pada scope. Perbedaannya ada pada bagaimana data tersebut di pass atau dikirimkan.

Value pada $\_GET akan dikrimkan ke skrip melalui URL parameter, sedangkan value pada $\_POST akan diikirimkan ke skrip melalui method HTTP POST

* GET

Penggunaan GET digunakan ketika informasi yang dikirimkan oleh form bukan merupakan informasi sensitif(*password, nomor kartu kredit, dll)*. Hal ini disebabkan karena apapun tipe data dan informasinya akan selalu ditampilkan pada URL page.



*Gambar 1.2 Pengiriman password menggunakan method Get*

GET juga memili batasan seberapa panjang atau banyak informasi yang dikrimkan. Batasannya adalah sekitar 2000 karakter. Namun GET memiliki kelebihan, karena datanya tersemat pada URL, page tersebut dapat di-*bookmark* tanpa harus mengisi form lagi.

* POST

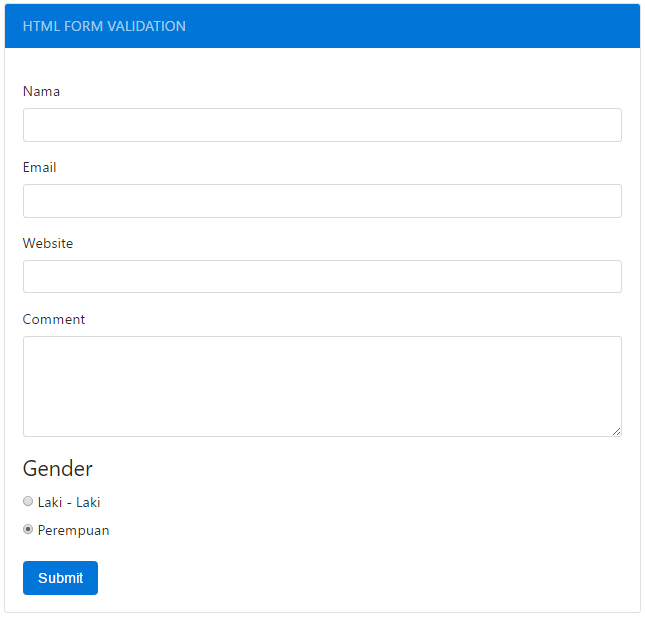
Data yang dikirim melalui method POST tidak dapat dilihat oleh user. Hal ini dapat terjadi karena data disematkan dalam HTTP request body, dan tidak ada batasan seberapa besar atau banyak informasi atau data yang dikirimkan.

Namun, dikarenakan data tersebut tidak tersemat pada URL seperti method GET, utnuk mendapatkan hasil output fungsi tertentu sebuah form, user harus mengisi lagi form yang ada untuk melihat hasil dari fungsi tersebut (mustahil untuk dilakukan *bookmark*)

***PHP Form Validation***

* Form HTML

Form validation sangat diperlukan untuk menghindari adanya celah keamanan yang serisu dalam pembuatan form. Untuk memahami lebih dalam tentang form validation, akan dibuat form seperti *gambar 2.0*.



*Gambar 2.0 Contoh Form HTML 2*

* Form Element

Pada *gambar 2.0,* dapat dilihat ada beberapa element terkait, dengan menggunakan fitur form HTML, dapat dibuat input field dengan type yang berbeda-beda. Pada form tersebut terdapat field, name, email, website comment, dan gender serta dilengkapi dengan button submit

<div class="form-group">

<label **for**="inputName">Nama</label>

<input type="text" name="nama" class="form-control" id="inputName">

</div>

<div class="form-group">

<label **for**="inputEmail">Email</label>

<input type="email" name="email" class="form-control" id="inputEmail">

</div>

<div class="form-group">

<label **for**="inputWebsite">Website</label>

<input type="text" name="website" class="form-control" id="inputWebsite">

</div>

<div class="form-group">

<label **for**="inputComment">Comment</label>

<textarea class="form-control" id="inputComment" rows="5"></textarea>

</div>

*<BOX CODE 3 element text field>*

<fieldset class="form-group">

<legend>Gender</legend>

<div class="form-check">

<label class="form-check-label">

<input type="radio" name="optionsRadios" class="form-check-input" id="optionsRadios1" value="Laki-Laki" checked> Laki - Laki

</label>

</div>

<div class="form-check">

<label class="form-check-label">

<input type="radio" name="optionsRadios" class="form-check-input" id="optionsRadios2" value="Perempuan" checked> Perempuan

</label>

</div>

</fieldset>

*<BOX CODE 4 element radio button>*

<form action="<?php echo htmlspecialchars($\_SERVER['PHP\_SELF']);?>" method="post">

*<BOX CODE 5 elemetn form>*

Dapat diperhatikan pada *Box Code 5*, terdapat skrip php singkat pada atribut *action* nya. Dapat diperhatikan juga adanya fungsi htmlspecialchars() dan superglobal *PHP\_SELF*. Superglobal *PHP\_SELF* berisi nama file php itu sendiri, sehingga maksud dari skrip ini adalah, data tersebut di kirimkan ke skrip pada halaman itu juga.

Untuk method htmlspecialchars() digunakan untuk mengkonversi special character ke HTML entity. Artinya, jika ada data yang berbentuk salah satu karakter essensial pada HTML, seperti *‘<>’, ‘&lt’, ‘&gt’* dan lainnya akan direplace menjadi sebuah entitas HTML. Hal ini dapat mencegah penyerangan melalui XSS atau Cross Site Scripting.

* PHP Form Security

Seperti yang dijelaskan di atas, superglobal *PHP\_SELF* berisi nama file yang sedang diakses. Hal ini dapat sangat berbahaya dan dapat di eksploitasi untuk menjalankan XSS.

Lebih jelasnya seperti berikut, sebuah web memiliki form dan skrip dengan nama est\_form.php” yang berisi seperti ini

<form action="<?php echo $\_SERVER['PHP\_SELF'];?>" method="post">

*Gambar 2.1 Atribut action pada tag <form>*

Jika seseorang membuka URL *‘http://* [*www.contoh.com/test\_form.php*](http://www.contoh.com/test_form.php)*’* maka atribut action pada tag <form> akan direpresentasikan menjadi seperti yang ada pada *gambar 2.2.*

<form action="test\_form.php" method="post">

*Gambar 2.2, Akan direpresentasikan seperti ini*

Dan jika sesorang membuka URL custom semacam seperti yang ada pada *gambar 2.3,*

localhost/test\_form.php/%22%3E%3Cscript%3Ealert('hacked')%3C

*Gambar 2.3, URL custom dijalankan*

Representasi pada action formnya adalah seperti pada *gambar 2.4* yang mana dapat dilihat, bahwa tag <script> dijalankan sebagai syntax HTML biasa.

<form method="post" action="test\_form.php/"><script>alert(***'hacked'***)

</script>

*Gambar 2.4, Akan direpresentasikan sperti ini*

Hal seperti ini akan sangat berbahaya jika script yang dijalankan dapat merusak tampilan website atau bahkan digunakan untuk mencuri data pada database web application. Untuk mengatasi hal ini, PHP memiliki beberapa pendekatan dalam coding nya. Salah satunya adalah dengan menggunakan fungsi htmlspecialchars().

Jika fungsi htmlspecialchars() dijalankan dan diaplikasikan pada action sebuah form, sperti pada *gambar 2.5,*

<form action="<?php echo htmlspecialchars($\_SERVER['PHP\_SELF']);?>" method="post">

*Gambar 2.5, pengaplikasian fungsi htmlspecialchars()*

Maka fungsi tersebut akan melakukan konversi special character ke entitas HTML. Jika seseorang mencoba menjalankan atau membuka URL seperti pada *Gambar 2.3* di atas, maka representasinya pada action adalah sebagai yang ditunjukan pada *gambar 2.6*

<form method="post" action="test\_form.php/&quot;&gt;&lt;script***&***

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

*Gambar 2.6, representasi dengan fungsi htmlspecialchars()*

* Form Validation menggunakan PHP

Ketika form dilengkapi dengan fungsi htmlspecialchars() form tidak hanya terlindung dari percobaan XSS dengan memodifikasi URL, namun juga terlindung dari percobaan XSS yang dilakukan dengan cara melakukan input script pada input field sebuah form.

Pada saat dijalankan fungsi htmlspecialchars, fungsi tersebut akan mengubah special character manjadi HTML escape character, sehingga special character tersebut tidak dianggap sebagai sebuah syntax.

Selain itu, ada beberapa langkah pencegahan yang sudah harus dilakukan yaitu menghilangkan ekstara spasi, newline atau tab dari input field dan menghilangkan *backslahses* dari sebuah input data.

<?php

    $name = $email = $gender = $comment = $website = "";

**if**($\_SERVER["REQUEST\_METHOD"] == "POST"){

      $name = test\_input($\_POST["nama"]);

      $email = test\_input($\_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;

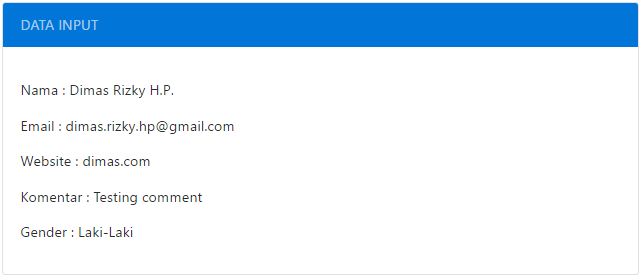
    }

?>

*<BOX CODE 6 FORM VALIDATION>*

Pada *Box Code 6,* dapat diperhatikan adanya tiga fungsi validasi yang dijalankan, diantaranya adalah fungsi trim() untuk menghilangkan ekstra spasi, fungsi stripslashes() untuk menghilangkan backslahses, lalu fungsi htmlspecialchars() untuk mengkonversi special character menjadi escape character HTML.

Hasil dari form tersebut dapat dilihat pada *gambar 2.7*



*Gambar 2.7 output test input*

***PHP Form Required***

* Required Property

Penggunaan form erat kaitannya dengan operasi query data melalui database ataupun eksekusi sebuha baris kode melalui fungsi yang memerlukan data dari form yang dikirimkan. Akan sangat berbahaya jika user melakukan kesalahan dalam memasukan data pada form, contohnya adalah dengan secara tidak sengaja user lupa mengisi salah satu input form.

Kesalahan lain yang rentan adalah jika user salah memasukan type data sebuah atribut, contohnya pada form email, user salah memasukan nomor teleponnya ataupun tanggal ulang tahunnya, tentunya hal ini akan menenggarai error pada fungsi yang akan berjalan atau query database yang akan dilakukan. Untuk menghindari hal ini, terdapat fitur pada PHP yaitu required fields.

<?php

    $name = $email = $gender = $comment = $website = "";

    $nameErr = $emailErr = $genderErr = $websiteErr = "";

**if**($\_SERVER["REQUEST\_METHOD"] == "POST"){

**if**(**empty**($\_POST["nama"])){

        $nameErr = "Nama harus diisi";

      } **else** {

        $name = test\_input($\_POST["nama"]);

      }

**if**(**empty**($\_POST["email"])){

        $emailErr = "Email harus diisi";

      } **else** {

        $email = test\_input($\_POST["email"]);

      }

**if**(**empty**($\_POST["website"])){

        $websiteErr = "";

      } **else** {

        $website = test\_input($\_POST["website"]);

      }

**if**(**empty**($\_POST["gender"])){

        $genderErr = "Gender harus diisi";

      } **else** {

        $gender = test\_input($\_POST["gender"]);

      }

      $comment = test\_input($\_POST["comment"]);

    }

**function** test\_input($data){

      $data = trim($data);

      $data = stripslashes($data);

      $data = htmlspecialchars($data);

**return** $data;

    }

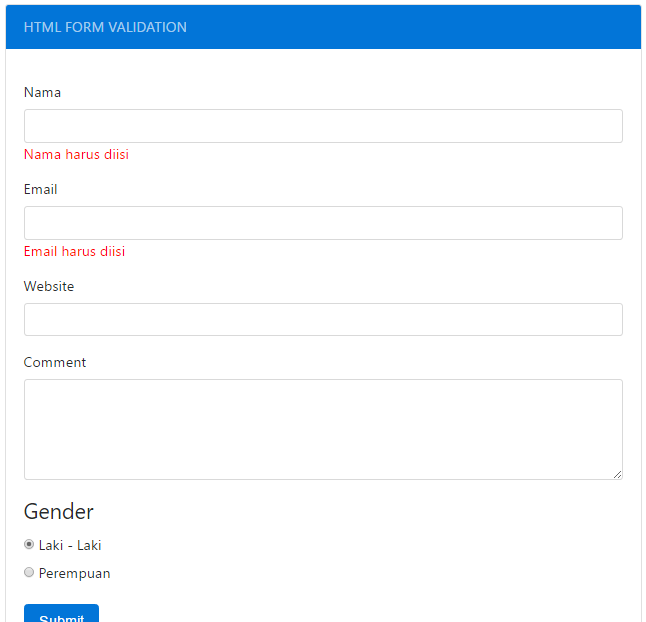
?>

\

*<BOX CODE 7 PHP REQUIRED>*

* Require Error Message

Untuk lebih memberikan kemudahan pada user, tentunya harus diberitahukan letak kesalahan yang dilakukan oleh user. Baiknya, pada saat terjadi error input, pesan error dimunculkan untuk memberitahu user letak kesalahannya, sehingga user dapat melakukan input pada form dengan benar.



*Gambar 3.1, Muncul error message*

***PHP Form URL/E-mail***

* Validate Name

Pada praktisnya, form name pada aplikasi web hanya bisa berisi huruf dan spasi, tanpa ada nomor ataupun simbol. Hal tersebut dapat dispesifikan menggunakan fungsi PHP preg\_match(). Fungsi ini akan mencari pola sebuah string, akan menghasilkan return true jika pattern terpenuhi namun, false jika tidak terpenuhi, untuk penerapan pada koding dapat dilihat pada

$name = test\_input($\_POST["nama"]);

**if** (!preg\_match("/^[a-zA-Z ]\*$/",$name)) {

  $nameErr = "Hanya huruf dan spasi yang diperbolehkan";

}

*<BOX CODE 9 VALIDATE NAME>*

* Validate Email

Alamat email secara standar memiliki struktur yang spesifik, yaitu <nama email>@<domain email>, akan menghasilkan error jika user mensuplai informasi email dengan struktur yang salah. Untuk melakukan validasi email, dapat dengan mudah dilakukan dengan fungsi PHP filter\_var(). Penerapan kodenya dapat dilihat pada *Box Code 10*

$email = test\_input($\_POST["email"]);

**if** (!filter\_var($email, FILTER\_VALIDATE\_EMAIL)) {

  $emailErr = "Format emailnya salah";

}

*<BOX CODE 10 VALIDATE EMAIL>*

* Validate URL

URL juga memiliki struktur yang spesifik layaknya email, yaitu www.<namadomain>. Untuk memeriksa kevalidan suatu URL dapat dilakukan dengan fungsi preg\_match() dengan melakukan cek polanya. Penerapan pada coding dapat dilihat pada *Box Code 11.*

$website = test\_input($\_POST["website"]);

**if** (!preg\_match("/\b(**?:**(**?:**https?|ftp):\/\/|www\.)[-a-z0-9+&@#\/%?=~\_|!:,.;]\*[-a-z0-9+&@#\/%=~\_|]/**i**",$website)) {

  $websiteErr = "URL Salah";

}

<BOX CODE 11 VALIDATE URL>

* PHP Complete Validation

Implementasi validation dan pemunculan message error tersebut dapat dilihat bentuk kodingnya pada *Box Code 12,* dan hasilnya pada *gambar 4.1*

<?php

**if**($\_SERVER["REQUEST\_METHOD"] == "POST"){

**if**(**empty**($\_POST["nama"])){

      $nameErr = "Nama harus diisi";

    } **else** {

      $name = test\_input($\_POST["nama"]);

**if**(!preg\_match("/^[a-zA-Z ]\*$/", $nama)){

        $nameErr = "Hanya huruf dan spasi yang diperbolehkan"

      }

    }

**if**(**empty**($\_POST["email"])){

      $emailErr = "Email harus diisi";

    } **else** {

      $email = test\_input($\_POST["email"]);

**if** (!filter\_var($email, FILTER\_VALIDATE\_EMAIL)) {

        $emailErr = "Format emailnya salah";

      }

    }

**if**(**empty**($\_POST["website"])){

      $websiteErr = "";

    } **else** {

      $website = test\_input($\_POST["website"]);

**if** (!preg\_match("/\b(**?:**(**?:**https?|ftp):\/\/|www\.)[-a-z0-9+&@#\/%?=~\_|!:,.;]\*[-a-z0-9+&@#\/%=~\_|]/**i**",$website)) {

        $websiteErr = "URL Salah";

      }

    }

**if**(**empty**($\_POST["gender"])){

      $genderErr = "Gender harus diisi";

    } **else** {

      $gender = test\_input($\_POST["gender"]);

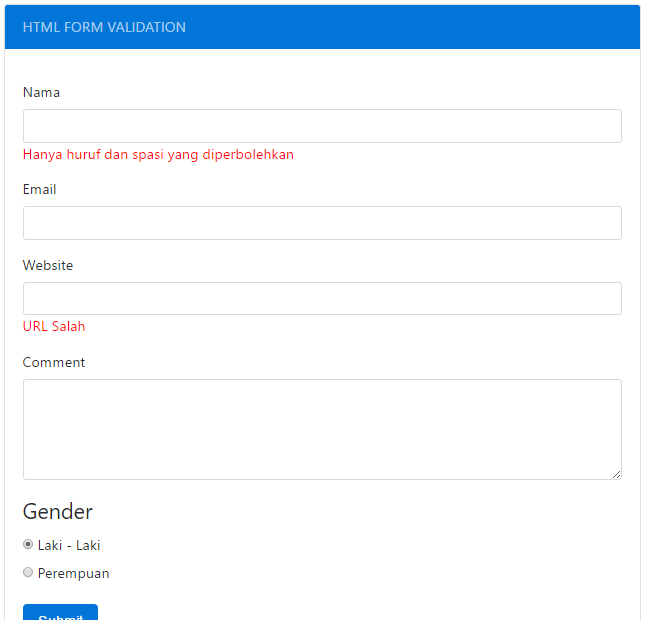
    }

    $comment = test\_input($\_POST["comment"]);

}

?>

*<BOX CODE 12 COMPLETE VALIDATION>*

**

*Gambar 4.1 Output hasil koding box code 12*

***PHP Form Complete***

* PHP Complete Form Validation

Pengaplikasian keseluruhan element form PHP yang ada dapat dilihat pada *Box Code 13* dan outputnya pada *Gambar 5.1.* Form ini dilengkapi dengan form validation, required attribute, input validation dan action validation. Dengan tambahan framework bootstrap untuk tampilan dari formnya

<!DOCTYPE html>

<**html** lang="en">

<**head**>

<!-- REQUIRED METAS -->

<**meta** charset="utf-8">

<**meta** name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">

<!-- CSS BOOTSTRAP -->

<**link** rel="stylesheet" href="css/bootstrap.min.css">

<**link** rel="stylesheet" href="css/custom.css">

</**head**>

<**body**>

<?php

  $name = $email = $gender = $comment = $website = "";

  $nameErr = $emailErr = $genderErr = $websiteErr = "";

**if**($\_SERVER["REQUEST\_METHOD"] == "POST"){

**if**(**empty**($\_POST["nama"])){

      $nameErr = "Nama harus diisi";

    } **else** {

      $name = test\_input($\_POST["nama"]);

**if**(!preg\_match("/^[a-zA-Z ]\*$/", $name)){

        $nameErr = "Hanya huruf dan spasi yang diperbolehkan";

      }

    }

**if**(**empty**($\_POST["email"])){

      $emailErr = "Email harus diisi";

    } **else** {

      $email = test\_input($\_POST["email"]);

**if** (!filter\_var($email, FILTER\_VALIDATE\_EMAIL)) {

        $emailErr = "Format emailnya salah";

      }

    }

**if**(**empty**($\_POST["website"])){

      $websiteErr = "";

    } **else** {

      $website = test\_input($\_POST["website"]);

**if** (!preg\_match("/\b(**?:**(**?:**https?|ftp):\/\/|www\.)[-a-z0-9+&@#\/%?=~\_|!:,.;]\*[-a-z0-9+&@#\/%=~\_|]/**i**",$website)) {

        $websiteErr = "URL Salah";

      }

    }

**if**(**empty**($\_POST["gender"])){

      $genderErr = "Gender harus diisi";

    } **else** {

      $gender = test\_input($\_POST["gender"]);

    }

    $comment = test\_input($\_POST["comment"]);

  }

**function** test\_input($data){

    $data = trim($data);

    $data = stripslashes($data);

    $data = htmlspecialchars($data);

**return** $data;

  }

  ?>

<**div** class="container-fluid">

<**div** class="card form-satu">

<**div** class="card-header card-inverse card-primary mb-3">

HTML FORM VALIDATION

</**div**>

<**div** class="card-block">

<**blockquote** class="card-blockquote">

<**form** action="<?php **echo** htmlspecialchars($\_SERVER['PHP\_SELF']);?>" method="post">

<**div** class="form-group">

<**label** for ="inputName">Nama</**label**>

<**input** type="text" name="nama" class="form-control" id="inputName"><**span** class="error" style="**color**: red"><?php **echo** $nameErr; ?></**span**>

</**div**>

<**div** class="form-group">

<**label** for="inputEmail">Email</**label**>

<**input** type="email" name="email" class="form-control" id="inputEmail"><**span** class="error" style="**color**: red"><?php **echo** $emailErr; ?></**span**>

</**div**>

<**div** class="form-group">

<**label** for="inputWebsite">Website</**label**>

<**input** type="text" name="website" class="form-control" id="inputWebsite"><**span** class="error" style="**color**: red"><?php **echo** $websiteErr; ?></**span**>

</**div**>

<**div** class="form-group">

<**label** for="inputComment">Comment</**label**>

<**textarea** class="form-control" id="inputComment" rows="5" name="comment"></**textarea**>

</**div**>

<**fieldset** class="form-group">

<**legend**>Gender</**legend**><**span** class="error"><?php **echo** $genderErr; ?></**span**>

<**div** class="form-check">

<**label** class="form-check-label">

<**input** type="radio" name="gender" class="form-check-input" id="optionsRadios1" value="Laki-Laki" checked>

Laki - Laki

</**label**>

</**div**>

<**div** class="form-check">

<**label** class="form-check-label">

<**input** type="radio" name="gender" class="form-check-input" id="optionsRadios2" value="Perempuan">

Perempuan

</**label**>

</**div**>

</**fieldset**>

<**button** type="submit" class="btn btn-primary">Submit</**button**>

</**form**>

</**blockquote**>

</**div**>

</**div**>

<**br**>

<**div** class="card form-satu">

<**div** class="card-header card-inverse card-primary mb-3">

DATA INPUT

</**div**>

<**div** class="card-block">

<**blockquote** class="card-blockquote">

<?php

**echo** "<p>Nama : ".$name."</p>";

**echo** "<p>Email : ".$email."</p>";

**echo** "<p>Website : ".$website."</p>";

**echo** "<p>Komentar : ".$comment."</p>";

**echo** "<p>Gender : ".$gender."</p>";

          ?>

</**blockquote**>

</**div**>

</**div**>

<**br**>

</**div**>

<!-- FILE JQUERY, TETHER, JS BOOTSTRAP -->

<**script** src="js/jquery-3.1.1.min.js"></**script**>

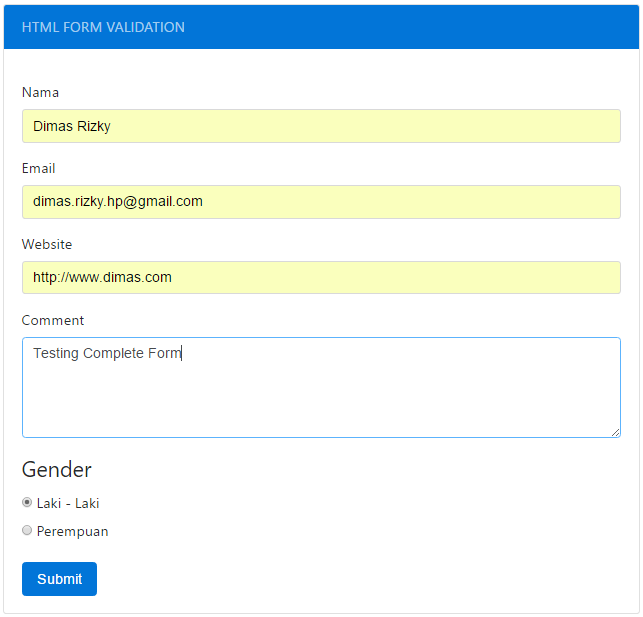
<**script** src="js/tether-1.4.0.min.js"></**script**>

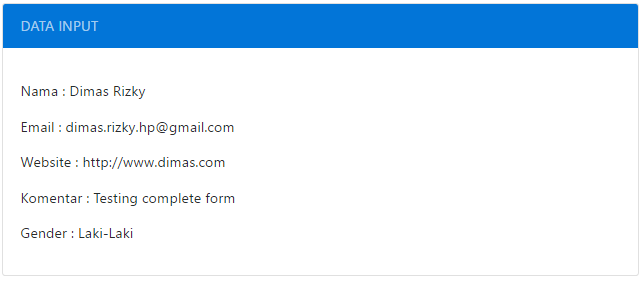
<**script** src="js/bootstrap.min.js"></**script**>

</**body**>

</**html**>

*<BOX CODE 13 COMPLETE FORM>*

**

**

*Gambar 5.1 Output dri complete form example.*