Praktikum Pemrograman Web Minggu 6 – PHP Advanced

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

***PHP File Include***

Statement PHP Include (atau require) berfungsi untuk mengikutsertakan seluruh text/code/markup yang ada pada file yang dispesifikan lalu meng-copy nya ke file yang dibubuhi statement *include.* Dengan meng-*include* suatu file akan dapat memudahkan programmer untuk menerapkan baris kode PHP ke banyak page yang ada.

* PHP include dan require Statements

Menyematkan kode atau konten dari satu file PHP ke file yang lain sangat memungkinkan dilakukan menggunakan statemen include atau require. Perbedaan mendasar antara statemen include dan require adalah pada saat kode gagal dieksekusi atau failure. Pada statemen require akan menghasilkan fatal error lalu skrip akan di stop, statemen include hanya akan menghasilkan warning dan skrip akan tetap dijalankan

* Penggunaan PHP include

Dikondisikan jika programmer memiliki file PHP untuk pembuatan footer sepeerti yang ada pada *Box Code 1*

<?php

**echo** "<p>Copyright &copy; 1999-" . date("Y") . " W3Schools.com</p>";

?>

*Box Code 1 – footer.php*

Untuk menyisipkannya, cukup menggunakan statement include seperti pada *Box Code 2,* penyisipan seperti yang ada pada box tersebut akan menyisipkan seluruh kode yang ada pada file *footer.php* setelah line ‘*<p>Some more ...</p>’* sehingga hasilnya 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">

HTML FOOTER INCLUDE

</**div**>

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

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

<**h3**>Selamat Datang Di Home Page Saya</**h3**>

<**p**>Ini text,</**p**>

<**p**>Some more text.</**p**>

<?php **include** 'footer.php';?>

</**blockquote**>

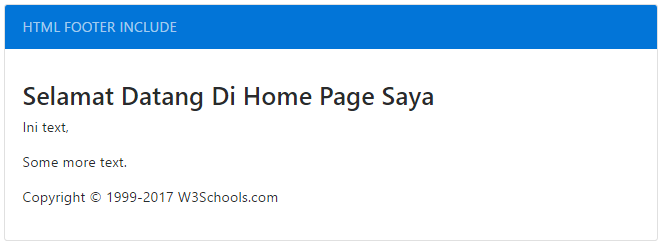
</**div**>

</**div**>

<**br**>

</**div**>

*Box Code 2 – Penerapan footer.php pada file php lain*

**

*Gambar 1.0. Output box code 2*

Contoh penggunaan lainnya adalah jika terdapat file PHP *menu.php* yang fungsinya untuk membuat bilah menu yang dapat dilihat dalam *Box Code 3*, dan ingin disematkan ke beberapa pages yang harus sama sama menggunakan menu tersebut.

<?php

**echo** '<a href="/default.asp">Home</a> -

<a href="/html/default.asp">HTML Tutorial</a> -

<a href="/css/default.asp">CSS Tutorial</a> -

<a href="/js/default.asp">JavaScript Tutorial</a> -

<a href="default.asp">PHP Tutorial</a>';

?>

*Box Code 3 – menu.php*

Penerapannya sama dan tidak berbeda dengan yang sebelumnya, cukup untuk menggunakan statement *include*. Penerapannya dapat dilihat dalam *Box Code 4*, dan hasilnya ada pada *gambar 1.1*.

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

<**div** class="menu">

<?php **include** 'menu.php';?>

</**div**>

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

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

HTML MENU INCLUDE

</**div**>

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

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

<**h3**>Selamat Datang Di Home Page Saya</**h3**>

<**p**>Ini text,</**p**>

<**p**>Some more text.</**p**>

</**blockquote**>

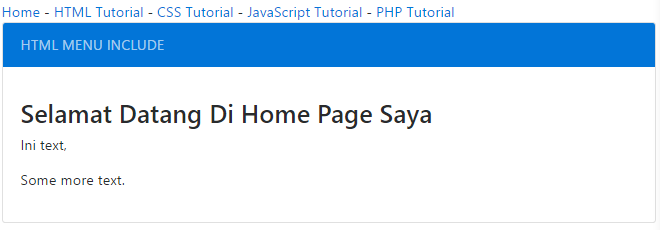
</**div**>

</**div**>

<**br**>

</**div**>

*Box Code 4 – Penerapan include file menu.php*

**

*Gambar 1.1 Output box code 4*

Penerapan include juga termasuk definisi variabel dalam sebuah file php, artinya seluruh variabel yang telah didefinisikan pada suatu file yang di include akan terbawa juga. Contohnya adalah jika ada file *vars.php* dengan code dapat dilihat pada *Box Code 5*

<?php

$color='Merah';

$car='BMW';

?>

*Box Code 5 – vars.php*

Penerapannya sama dengan contoh sebelumnya, menggunakan statement include pada file php yang dikehendaki. Code nya dapat dilihat pada *Box Code 6* dan outputnya dapat dilihat pada *gambar 1.2.*

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

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

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

HTML VARIABLE INCLUDE

</**div**>

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

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

<**h3**>Selamat Datang Di Home Page Saya</**h3**>

<?php **include** 'vars.php';

**echo** "Saya mempunyai sebuah $car $color.";?>

</**blockquote**>

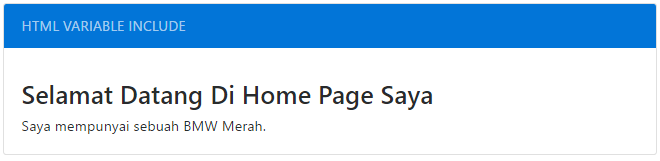
</**div**>

</**div**>

<**br**>

</**div**>

*Box Code 6 – Include vars.php pada sebuah page*

**

*Gambar 1.2 output dari box code 6*

* PHP include vs. Require

Statement require juga digunakan untuk melakukan include pada sebuah skrip PHP, namun penggunaannya akan efektif jika dalam beberapa kondisi. Hal ini dikarenakan karena adanya perbedaan antara statement *Include* dan *Requirement* itu sendiri.

Ketika suatu file di-*include* menggunakan statement *include*, jika PHP tidak dapat menemukan file tersebut, script akan tetap berjalan/dieksekusi. Contoh dapat dilihat pada *Box Code 7*, dan hasilnya dapat dilihat pada *gambar 1.3.*

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

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

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

PHP INCLUDE

</**div**>

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

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

<**h3**>Selamat Datang Di Home Page Saya</**h3**>

<?php **include** 'filenyatidakaada.php';

**echo** "Saya mempunyai sebuah $car $color.";?>

</**blockquote**>

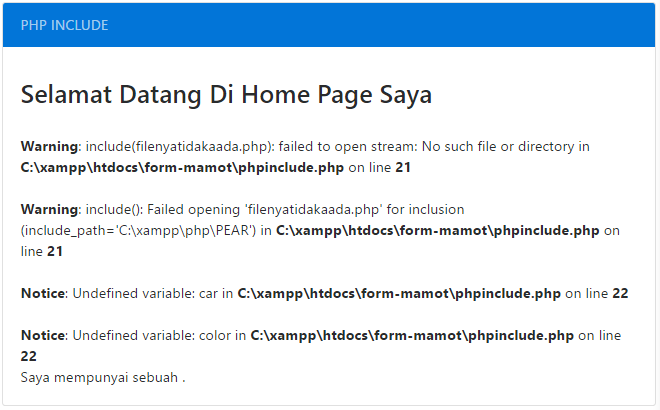
</**div**>

</**div**>

<**br**>

</**div**>

*Box Code 7 – Statement include*

**

*Gambar 1.3 output jika menggunakan statement include*

Pada statement *require* jika PHP tidak menemukan file yang dispesifikan, script akan berakhir dan selesai dieksekusi disertai dengan fatal error. Contoh dapat dilihat pada *Box Code 8* dan hasilnya dapat dilihat pada *gambar 1.4*.

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

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

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

PHP INCLUDE

</**div**>

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

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

<**h3**>Selamat Datang Di Home Page Saya</**h3**>

<?php **require** 'filenyatidakaada.php';

**echo** "Saya mempunyai sebuah $car $color.";?>

</**blockquote**>

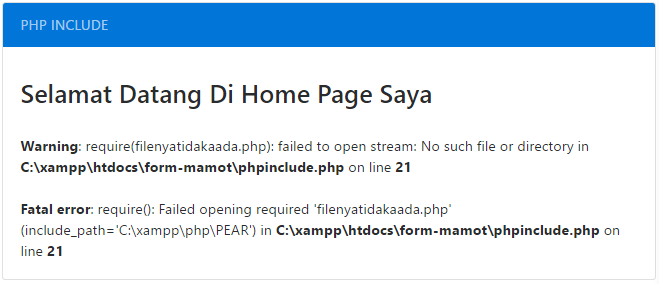
</**div**>

</**div**>

<**br**>

</**div**>

*Box Code 8 – Statement require*

**

*Gambar 1.3 output jika menggunakan statement require*

Sehingga ada kondisi optimal dimana statement require dan include bisa digunakan, yaitu gunakan *require* ketika file tersebut sangat dibutuhkan oleh aplikasi, dan statement *include* jika file tidak terlalu dibutuhkan oleh aplikasi dan jika skrip harus terus berjalan meskipun file tidak dapat ditemukan.

***PHP File Handling***

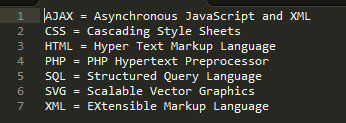
File handling merupakan bagian penting dari sebuah web aplikasi. Programmer seringkali membutuhkan sebuah file untuk dieksekusi sebagai bagian dari program atau data yang akan diproses.

* Memanipulasi File dengan PHP

PHP memiliki beberapa fungsi yang khusus untuk membuat, membaca, mengunggah, dan mengedit file. Perlu diingat ketika memanipulasi sebuah file, karena dengan penanganan file yang salah, dapat terjadi error pada file, error pada hard-drive karena penuh dengan data sampah, membuat file menjadi corrupt dan lainnya.

* Fungsi PHP readfile()

Fungsi *readifile()* digunakan untuk membaca dan menulis pada sebuah file. Contohnya dapat dilihat pada *Box Code 9,* dimana perintah *readfile()* dipanggil dengan parameter *“kamusweb.txt”*. *“kamusweb.txt”* merupakana file .txt yang isinya dapat dilihat pada  *gambar 2.0*. Hasil dari eksekusi *Box Code 9* dapat dilihat pada *gambar 2.1*

**

*Gambar 2.0 File kamusweb.txt*

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

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

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

PHP readfile()

</**div**>

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

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

<?php

**echo** readfile("kamusweb.txt");

          ?>

</**blockquote**>

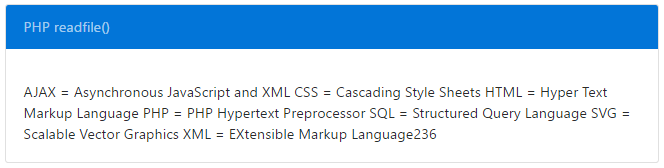
</**div**>

</**div**>

<**br**>

</**div**>

*Box Code 9 – penerapan fungsi readfile()*

**

*Gambar 2.1 Output*

***PHP File Open/Read***

* PHP Open file – fopen()

Selain fungsi *readfile()* terdapat juga fungsi yang memberikan kontrol yang lebih pada saat membuka file yaitu function *fopen(). Fopen()* menawarkan mode pembacaan yang berbeda beda sehingga kontrol dalm file juga lebih baik didapatkan menggunakan fungsi *fopen()*.

Fungsi *fopen()* memiliki beberapa parameter, parameter pertamanya adalah nama file yang akan dibuka, parameter selanjutnya adalah mode pembacaan file yang akan digunakan lalu dapat ditambahkan message handling jika file tersebut gagal terbuka.

Mode file yang tersedia pada fungsi ini adalah sebagai berikut :

* + *r, membuka file dengan mode read only, pointer file berada pada awal file*
  + *w, membuka file dengan mode write only, menghapus konten dari file atau membuat file baru jika filenya belum ada.*
  + *a, membuka file dengan mode write only, data yang sudah ada tidak akan dihapus, namun akan membuat file baru ketika file yang dibuka tidak ada*
  + *x, membuat file baru untuk mode write only, akan menghasilkan nilai return FALSE dan sebuah error statement jika file sudah ada*
  + *r+, membuka sebuah file dengan mode read/write*
  + *w+, membuka file dengan mode read/write, menghapus seluruh konten yang ada pada file dan akan membuat file baru jika file tidak ditemukan*
  + *a+, membuka file dengan mode read/write, data yang sudah ada sebelumnya tidak akan dihapus, dan akan membuat file baru jika file yang dibuka tidak ditemukan*
  + *x+, membuat file baru untuk mode read/write, memiliki return value FALSE jika file sudah ada*

Penerapannya dapat dilihat pada *Box Code 10* dan hasilnya dapat dilihat pada *gambar 3.0.* File yang akan dibuka masih menggunakan file *”kamusweb.txt”* yang isinya dapat dilihat pada *gambar 2.0.*

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

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

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

PHP fopen()

</**div**>

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

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

<?php

            $myfile = fopen("kamusweb.txt","r")or **die**("Tidak dapat membuka file");

**echo** fread($myfile,filesize("kamusweb.txt"));

            fclose($myfile);

          ?>

</**blockquote**>

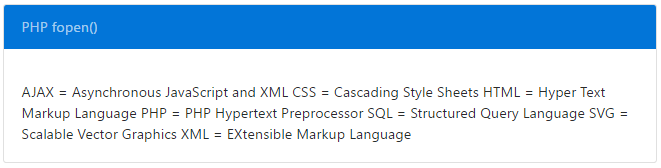
</**div**>

</**div**>

<**br**>

</**div**>

*Box Code 10 – Penerapan fopen()*

**

*Gambar 3.0 Output dari box code 10*

* PHP Read file – fread()

Fungsi *fread()* digunakan untuk membaca file yang sudah terbuka. Terdapat beberapa parameter pada fungsi *fread()*. Paramater pertamanya adalah nama file yang akan dibaca dan parameter keduanya merupakan nilai maksimum byte yang akan dibaca. Contoh penerapan *fread()* dapat dilihat pada *Box Code 11*.

<?php

fread($myfile,filesize("webdictionary.txt"));

?>

*Box Code 11 – Penggunaan fread()*

* PHP Close file – fclose()

Fungsi *fclose()* digunakan untuk menutup file yang terlah terbuka. *Fclose()* sangat penting dijalankan agar file yang sudah dibutuhkan agar ditutup dan tidak membebani kerja memori dari server. *Fopen()* memiliki beberapa parameter salah satunya adalah nama file yang akan ditutup. Contoh penerapannya ada pada *Box Code 12.*

<?php

$myfile = fopen("webdictionary.txt", "r");

fclose($myfile);

?>

*Box Code 12 – Penggunaan fclose()*

* PHP Read Baris – fgets()

Fungsi *fgets()* digunakan untuk membaca suatu baris pada sebuah file satu persatu. Pada *Box Code 13,* ditunjukan cetak untuk isi baris pertama dari file *“kamusweb.txt”* yang output nya dapat ditunjukan pada *gambar 3.1.* Setelah fungsi fgets dipanggil, file pointer akan berpindah ke line selanjutnya.

<?php

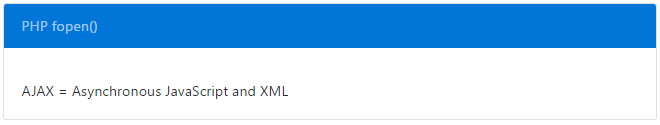
$myfile = fopen("kamusweb.txt","r")or **die**("Tidak dapat membuka file");

**echo** fgets($myfile);

fclose($myfile);

?>

*Box Code 13 – Penggunaan fgets()*

**

*Gambar 3.1 Output*

* PHP Cek End-Of-File – feof()

Fungsi *feof()* akan melakukan cek apakah EOF telah dicapai atau file pointer telah mencapai akhir dari file. Fungsi *feof()* sangat berguna untuk mendefinisikan kondisi looping dengan data yang panjangnya tidak diketahui. Contohnya dapat dilihat pada *Box Code 14* yang mana akan membaca text pada file *“kamusweb.txt”* sampai dengan end-of-file nya tercapai. Hasilnya dapat dilihat pada *gambar 3.2*.

<?php

$myfile = fopen("kamusweb.txt","r")or **die**("Tidak dapat membuka file");

**while**(!feof($myfile)){

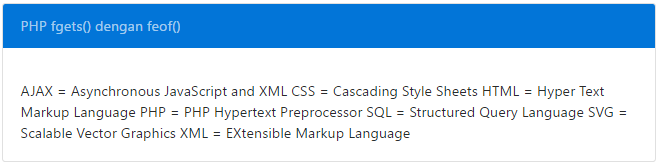
**echo** fgets($myfile);

}

fclose($myfile);

?>

*Box Code 14 – Penggunaan feof() pada kondisi looping*

**

*Gambar 3.2 Output file*

* PHP Read Karakter – fgetc()

Fungsi *fgetc()* digunakan untuk membaca file per karakter sampai dengan end-of file tercapai. Contohnya dapat dilihat pada *Box Code 15* yang mana akan membaca dan mencetak isi dari file *“kamusweb.txt”* per karakter. Hasilnya dapat dilihat pada *gambar 3.3*.

<?php

$myfile = fopen("kamusweb.txt","r")or **die**("Tidak dapat membuka file");

**while**(!feof($myfile)){

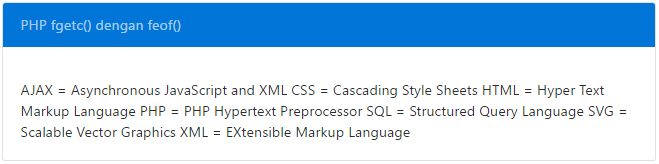
**echo** fgetc($myfile);

}

fclose($myfile);

?>

*Box Code 15 – penggunaan fgetc()*

**

*Gambar 3.3 Output pembacaan file menggunakan fgetc()*

***PHP File Create/Write***

* PHP Create File menggunakaan fopen()

Fungsi *fopen()* juga dapat digunakan untuk membuat sebuah file. Jika programmer menggunakan fungsi *fopen()* dan dengan parameter nama file nya tidak dapat ditemukan oleh sistem, maka sistem akan secara otomatis membuat file tersebut. Hal ini juga bergantung dengan parameternya yaitu dibuka dengan mode *writing (w)* atau *appending (a).* Penggunaan dalam kode dapat dilihat pada *Box Code 16.*

* PHP File permission

File permission menjadi hal yang penting jika sedang berurusan dengan file handling. Hal ini dikarenakan karena seringkali terjadi beberapa error penggunaan fungsi untuk membuka file karena permissionnya tidak cukup tinggi untuk membuka/menuliskan sebuah file. Sehingga harus dipastikan terlebih dahulu permission yang dimiliki.

* PHP Write File menggunakan write()

Fungsi *fwrite()* digunakan untuk menuliskan sebuah konten kedalam sebuah file. Parameter pertama dari fungsi *fwrite()* adalah nama file yang akan ditulis. Parameter keduanya adalah string yang akan dituliskan pada file tersebut. Contohnya dapat dilihat pada *Box Code 17* dan hasilnya dapat dilihat pada *gambar 4.0.*

<?php

$myfile = fopen("newfile.txt","w")or **die**("Tidak dapat membuka file");

$txt = "John Doe\n";

fwrite($myfile, $txt);

$txt = "John Doe\n";

fwrite($myfile, $txt);

fclose($myfile);

?>

*Box Code 17 – Penggunaan fungsi fwrite()*

**

*Gambar 4.0 isi dari filebaru.txt*

Dapat dilihat pada *Box Code 17* dilakukan penulisan ke file *“filebaru.txt”* sebanyak dua kali. Setiap kali dilakukan penulisan pada sebuah file, akan dikirimkan string $txt yang berisi string “John Doe” pada definisi variabel pertama lalu yang kedua juga berisi “John Doe”. Setelah selesai menuliskan variabel tersebut dipanggil fungsi *fclose()* untuk menutup file tersebut.

* PHP Overwriting

Untuk melakukan overwrite pada sebuah file dapat digunakan fungsi *fopen()* dengan mode *w*. Mode ini akan menghapus seluruh konten file yang sudah ada, sehingga file tersebut menjadi sebuah file yang baru dan dapat dilakukan overwrite terhadap file tersebut. Contohnya dapat dilihat pada *Box Code 18* dan hasilnya berupa file *“filebaru.txt”* yang telah di-overwrite pada *gambar 4.1*.

<?php

$myfile = fopen("newfile.txt","w")or **die**("Tidak dapat membuka file");

$txt = "Mickey Mouse\n";

fwrite($myfile, $txt);

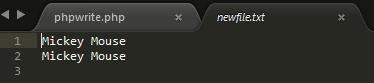
$txt = "Mickey Mouse\n";

fwrite($myfile, $txt);

fclose($myfile);

?>

*Box Code 18 – Overwrte file filebaru.txt*

**

*Gambar 4.1 filebaru.txt yang telah di-overwrite*

***PHP File Upload***

* Konfigurasi file “php.ini”

Pengunggahan file dalam aplikasi web bukan hal yang jarang saat ini, dengan PHP pengunggahan file dapat menjadi sangat mudah namun juga sangat berbahaya bagi keamanan aplikasi web itu sendiri. Oleh karena itu, tim developer PHP membuat file konfigurasi tersendiri untuk mengatur wewenang upload file ke server. Konfigurasi ini ada pada file *“php.ini”.* Dengan merubah directive file\_uploads menjadi on cukup untuk mengkonfigurasikan php untuk menerima file melalui pengunggahan.

* Membuat Form Upload di HTML

Setelah itu, langkah yang diperlukan adalah dengan membuat form untuk melakukan pengunggahan file ke server. Kode file dapat dilihat pada *Box Code 19*, dengan tampilannya dapat dilihat pada *gambar 5.0.*

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

    <div **class**="card form-satu">

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

            PHP fgetc() dengan feof()

        </div>

        <div **class**="card-block">

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

                <form action="upload.php" method="post" enctype="multipart/form-data">

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

                        <label **for**="uploadFile">File Upload</label>

                        <input type="file" name="file" **class**="form-control-file" id="uploadFile" aria-describedby="filehelp">

                        <small id="filehelp" **class**="form-text text-muted">Pilih Gambar Yang Akan Di-upload.</small>

                    </div>

                </form>

            </blockquote>

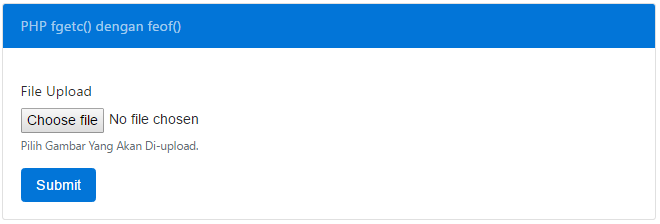
        </div>

    </div>

    <br>

</div>

*Box Code 19 – Kode Upload Form HTML*

**

*Gambar 5.0 output tampilan form HTML yang dibuat*

Beberapa hal yang harus diperhatikan, method yang digunakan pada form adalah POST dan form juga harus memiliki atribut enctype=”multipart/form-data”. Tanpa kedua spesifikasi tersebut proses upload akan gagal dilakukan.

* Buat Script PHP File Upload

Pada *Box Code 19* dapat dilihat bahwa data file tersebut akan dikirimkan ke script php bernama *upload.php.* Script yang akan dibuat dapat dilihat pada *Box Code 20*.

<?php

$target\_dir = "uploads/";

$target\_file = $target\_dir . basename($\_FILES["fileToUpload"]["name"]);

$uploadOK = 1;

$imageFileType = pathinfo($target\_file,PATHINFO\_EXTENSION);

**if**(**isset**($\_POST["submit"])){

$check = getimagesize($\_FILES["fileToUpload"]["tmp\_name"]);

**if**($check !== false){

**echo** "File merupakan file gambar - " . $check["mime"].".";

$uploadOK = 1;

} **else** {

**echo** "File bukan gambar.";

$uploadOK = 0;

}

}

?>

*Box Code 20 – Script upload.php*

Pada *Box Code 20* dapat dilihat terdapat beberapa variabel diantaranya adalah variabel berikut ini,

* + *$target\_dir = “uploads/”* – *menspesifikan direktori tempat dimana file hasil unggahan akan ditempatkan*
  + *$target\_file – menspesifikan file path dari file yang akan diunggah*
  + *$uploadOK=1*
  + *imageFileType – merupakan ekstensi dari file yang diunggah*

Selanjutnya adalah melakukan cek, apakah file tersebut sudah ada pada direktori /upload ataukah belum ada. Jika sudah, maka error message akan ditampilkan, dan variabel *$uploadOK* di set bernilai 0. Script nya dapat dilihat pada *Box Code 21*.

<?php

**if** (file\_exists($target\_file)) {

**echo** "File Sudah Ada.";

    $uploadOk = 0;

}

?>

*Box Code 21 – Cek keberadaan file*

Setelah mengecek keberadaan file pada direktori upload juga harus dispesifikan ukuran maksimal yang dapat diunggah. Pada contoh *Box Code 22,* akan dilakukan pembatasan file dengan batas maksimal 500KB, jika file berukuran lebih dari batas tersebut, pesan error akan ditampilkan dan variabel *$uploadOK* akan bernilai 0.

<?php

**if** ($\_FILES["fileToUpload"]["size"] > 500000) {

**echo** "Ukuran File Terlalu Besar.";

    $uploadOk = 0;

}

?>

*Box Code 22 – Batasi ukuran file*

Selanjutnya adalah dengan membatasi jenis file atau tipe file nya. Pada contoh *Box Code 20* form didesain untuk menerima file gambar, sehingga hanya dapat menerima file dengan ekstensi JPG, JPEG, PNG dan GIF. Jika file yang diunggah selain dari file tersebut, maka error message akan ditampilkan dan variabel *$uploadOK* akan di set menjadi 0. Penerapan kode ini dapat dilihat pada *Box Code 23*.

<?php

**if**($imageFileType != "jpg" && $imageFileType != "png" && $imageFileType != "jpeg"

&& $imageFileType != "gif" ) {

**echo** "Hanya File JPG, JPEG, PNG & GIF Yang Dapat Diunggah.";

    $uploadOk = 0;

}

?>

*Box Code 23 – Batasi jenis file*

* Complete PHP Script

Setelah seluruh dibuat, hasil akhir dari script *upload.php* akan menjadi seperti pada *Box Code 24*. Script ini siap dijalankan dan hasilnya dapat dilihat pada *gambar 5.1* dan *gambar 5.2*.

<?php

$target\_dir = "uploads/";

$target\_file = $target\_dir . basename($\_FILES["fileToUpload"]["name"]);

$uploadOK = 1;

$imageFileType = pathinfo($target\_file,PATHINFO\_EXTENSION);

**if**(**isset**($\_POST["submit"])){

$check = getimagesize($\_FILES["fileToUpload"]["tmp\_name"]);

**if**($check !== false){

**echo** "File merupakan file gambar - " . $check["mime"].".";

$uploadOK = 1;

} **else** {

**echo** "File bukan gambar.";

$uploadOK = 0;

}

}

**if** (file\_exists($target\_file)) {

**echo** "File Sudah Ada.";

$uploadOK = 0;

}

**if** ($\_FILES["fileToUpload"]["size"] > 500000) {

**echo** "Ukuran File Terlalu Besar.";

$uploadOK = 0;

}

**if**($imageFileType != "jpg" && $imageFileType != "png" && $imageFileType != "jpeg"

&& $imageFileType != "gif" ) {

**echo** "Hanya File JPG, JPEG, PNG & GIF Yang Dapat Diunggah.";

$uploadOK = 0;

}

**if**($uploadOK == 0){

**echo** "Terjadi Error, File Gagal Diunggah.";

} **else** {

**if**(move\_uploaded\_file($\_FILES["fileToUpload"]["tmp\_name"], $target\_file)){

**echo** "File ". basename($\_FILES["fileToUpload"]["name"]). " Berhasil Diunggah.";

} **else** {

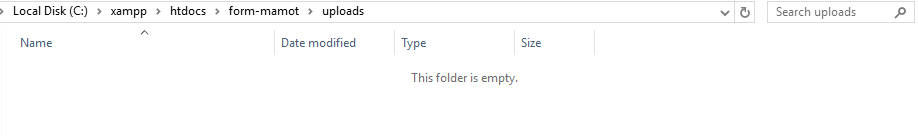
**echo** "Terjadi Error, File Gagal Diunggah";

}

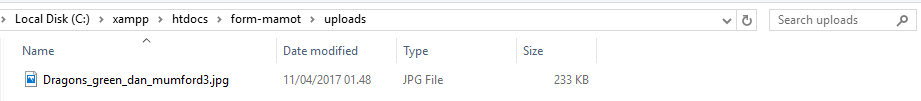
}

?>

*Box Code 24 – Kode script lengkap*

**

*Gambar 5.1 Folder upload sebelum dilakukan pengunggahan file*

**

*Gambar 5.2 Folder upload setelah pengunggahan telah dilakukan*