

Tutorial PHP dan MySQL : Membuat Aplikasi CRUD + Foto

PHP dan MySQL, dua hal yang sangat familiar di kalangan Programmer Web. Sebenarnya jika berurusan dengan database, tidak hanya MySQL, PHP pun dapat di kombinasikan dengan Oracle, PostgreSQL, SQLite3, MongoDB, MS SQL Server dan masih banyak lagi.

Kali ini kita akan belajar cara membuat aplikasi sederhana berbasis CRUD namun akan kita beri fitur upload foto agar lebih menarik..

Bagaimana PHP dan MySQL dapat terhubung ?

PHP dan MySQL dapat saling terhubung karena adanya API (Application Programming Interface) , sebuah perantara yang dapat digunakan untuk berkomunikasi antar dua Program Aplikasi. Terdapat 3 API yang dapat digunakan untuk menghubungkan MySQL dengan PHP :

1. mysql API (kadaluarsa , telah dihapus di PHP versi 7)
2. mysqli API
3. PDO (PHP Data Object)

Kali ini kita akan belajar cara membuat aplikasi sederhana berbasis CRUD menggunakan API mysqli API secara Object Oriented.

Database dan tabel

Silahkan buat database dengan nama 'latihan' dan buat tabel didalamnya dengan sintak SQL berikut :

```
CREATE TABLE `teman` (  
  `id` int(5) NOT NULL AUTO_INCREMENT,
```

```
`nama` varchar(250) NOT NULL,  
`alamat` text NOT NULL,  
`foto` varchar(250) NOT NULL,  
PRIMARY KEY (`id`)  
) ENGINE=InnoDB DEFAULT CHARSET=latin1
```

Buat Proyek Baru

Silahkan buat satu folder baru di dalam direktori utama web server (xampp/htdocs) dengan nama 'crud' lalu untuk buat satu lagi folder khusus untuk menampung gambar dengan nama 'file'

lalu buat beberapa file seperti gambar berikut :

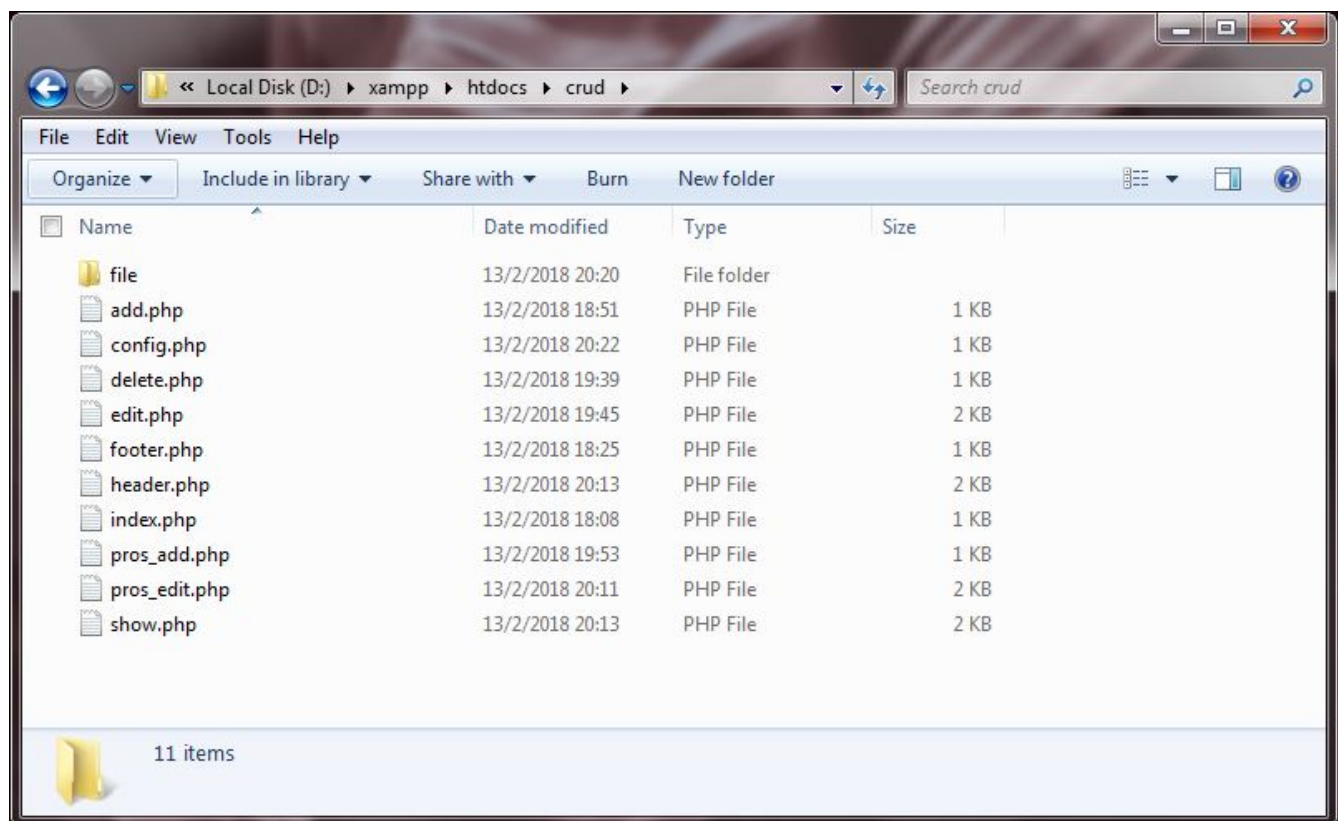


Illustration 1: stuktur file dan folder

add.php – halaman untuk tambah data

config.php – file konfigurasi database dan yang lainnya

delete.php – file untuk menangani action hapus data

edit.php – halaman untuk edit data

footer.php , header.php – template untuk navigasi dan menyamakan tampilan

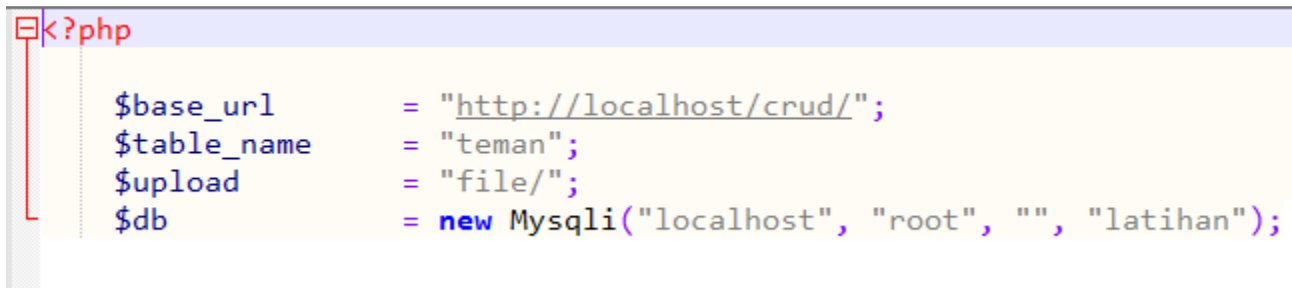
index.php – sebagai file pertama yang dijalankan, untuk mengarahkan ke halaman show.php

pros_add.php, pros_edit.php – untuk menangani proses tambah dan edit data.

show.php – untuk menampilkan data.

Membuat koneksi ke database MySQL

Buka file **config.php** , lalu isikan kode berikut :



```
<?php

$base_url      = "http://localhost/crud/";
$table_name     = "teman";
$upload        = "file/";
$db             = new mysqli("localhost", "root", "", "latihan");
```

Illustration 2: isi config.php

berikut penjelasan masing-masing kode :

\$base_url : menyimpan URL yang kita ketikkan saat akan membuka Aplikasi (diakhiri slash)

\$table_name : berisi nama tabel yang digunakan pada database.

\$upload : berisi nama folder tempat foto diupload (diakhiri slash)

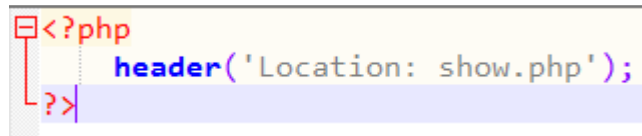
\$db : berisi Objek mysqli untuk memulai koneksi ke database MySQL

pada **\$db** sebenarnya kita membuat objek baru mysqli dimana beberapa parameter dari fungsi konstruktor objek mysqli adalah **nama host, nama user, password, nama database**.

Mengkonfigurasi file index.php (auto redirect)

Karena index.php adalah script utama yang dijalankan saat sebuah halaman web dimuat, maka kita dapat memanfaatkan sifat alami ini untuk mengalihkan langsung ke halaman penampilan data.

Buka file **index.php** dan isikan kode berikut :

A screenshot of a code editor showing PHP code. The first line is <?php in red. The second line is header('Location: show.php'); in blue. The third line is ?> in red. The code is highlighted with a light blue background.

```
<?php
header('Location: show.php');
?>
```

Illustration 3: redirect otomatis - index.php

Setelah itu , jika kita membuka alamat <http://localhost/crud> maka kita akan langsung dialihkan ke halaman **show.php** (halaman penampilan data)

Mengedit halaman template (header.php dan footer.php)

Kita akan meletakkan 2 hal yang harus ada disetiap halaman. Untuk efisiensi kode, maka ada baiknya untuk membuat file khusus lalu jika ingin menggunakannya, tinggal kita sertakan (dengan perintah **require**)

menambahkan 2 Link untuk navigasi pada file

buka file **header.php** dan ketikkan kode berikut:

```

<!DOCTYPE html>
<html lang="en">
<head>
  <style>
    *{ font-family: sans-serif; }
    .pesan{text-align: center;padding: 10px 0;font-weight: bold;}
    table.data{border-collapse: collapse;width: 88%;margin: 0 auto;}
    table.data th{padding: 6px;background-color: #eee;}
    table.data td{padding: 2px;}
    img{height: 60px;width: 60px;}
    footer{text-align: center;}
  </style>
  <title>Crud APP</title>
</head>
<body>
  <nav>
    <a href="<?php echo $base_url.'add.php' ?>">Tambah data</a>
    <a href="<?php echo $base_url ?>">Lihat data</a>
  </nav>

```

Illustration 4: template header.php

darimana **\$base_url** berasal ? Dari file config.php yang sudah kita buat tadi (silahkan dicek kembali) , sehingga setiap kali kita membutuhkan file **header.php** maka kita **WAJIB** menyertakan file **config.php**. (sebelum menyertakan header.php) Misal nya seperti ini :

```

<?php require_once('config.php') ?>
<?php require('header.php') ?>

```

Illustration 5: cara memanggil template

Lalu sekarang buka file **footer.php** dan ketikkan kode berikut :

```

<footer>
  <p> &copy; APP Crud <?php echo date("Y") ?></p>
</footer>
</body>
</html>

```

Illustration 6: template footer.php

fungsi date("Y") adalah fungsi PHP untuk menampilkan Tahun saat ini.

Halaman Penampilan data

Buka file **show.php** dan ketikkan kode berikut :

```
<?php require_once('config.php') ?>
<?php require('header.php') ?>

<h1>Daftar Teman-temanku</h1>

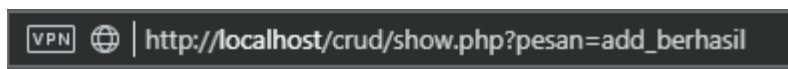
<?php
    if(isset($_GET['pesan'])) :
        echo "<p class='pesan'>".$_GET['pesan']. "</p>";
    endif;
?>

<table class="data" border='1'>
    <tr>
        <th>NO</th>
        <th>NAMA</th>
        <th>ALAMAT</th>
        <th>FOTO</th>
        <th>OPERASI</th>
    </tr>
<?php require("footer.php") ?>
```

Illustration 7: menampilkan data - show.php

petama kita menyertakan file konfigurasi utama kita (**config.php**) kemudian kita sertakan juga template **header.php** dan bagian akhir, **footer.php** . Lalu kita cek apakah terdapat data `$_GET` pada URL jika ada maka tampilkan.

Berikut contoh URL yang mengandung data `$_GET`



The image shows a browser address bar with a dark background. On the left, there is a 'VPN' icon and a globe icon. To the right of these icons, the URL 'http://localhost/crud/show.php?pesan=add_berhasil' is displayed in white text.

Illustration 8: contoh url

artinya , ada data `$_GET` dengan index 'pesan' dengan nilai 'add_berhasil'

Jika kita buka browser sekarang dengan alamat <http://localhost/crud> maka akan seperti ini :

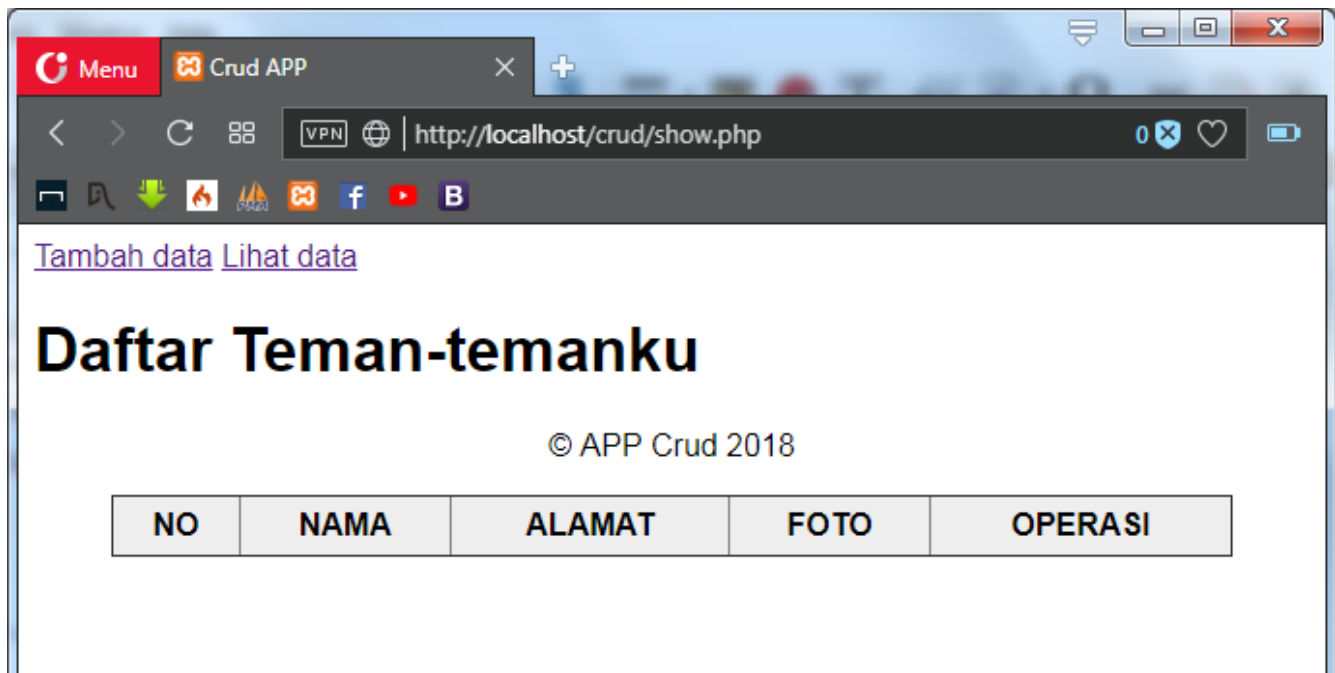


Illustration 9: masih berantakan ???

Tampilan nya mungkin masih berantakan, namun kita akan memperbaikinya nanti..

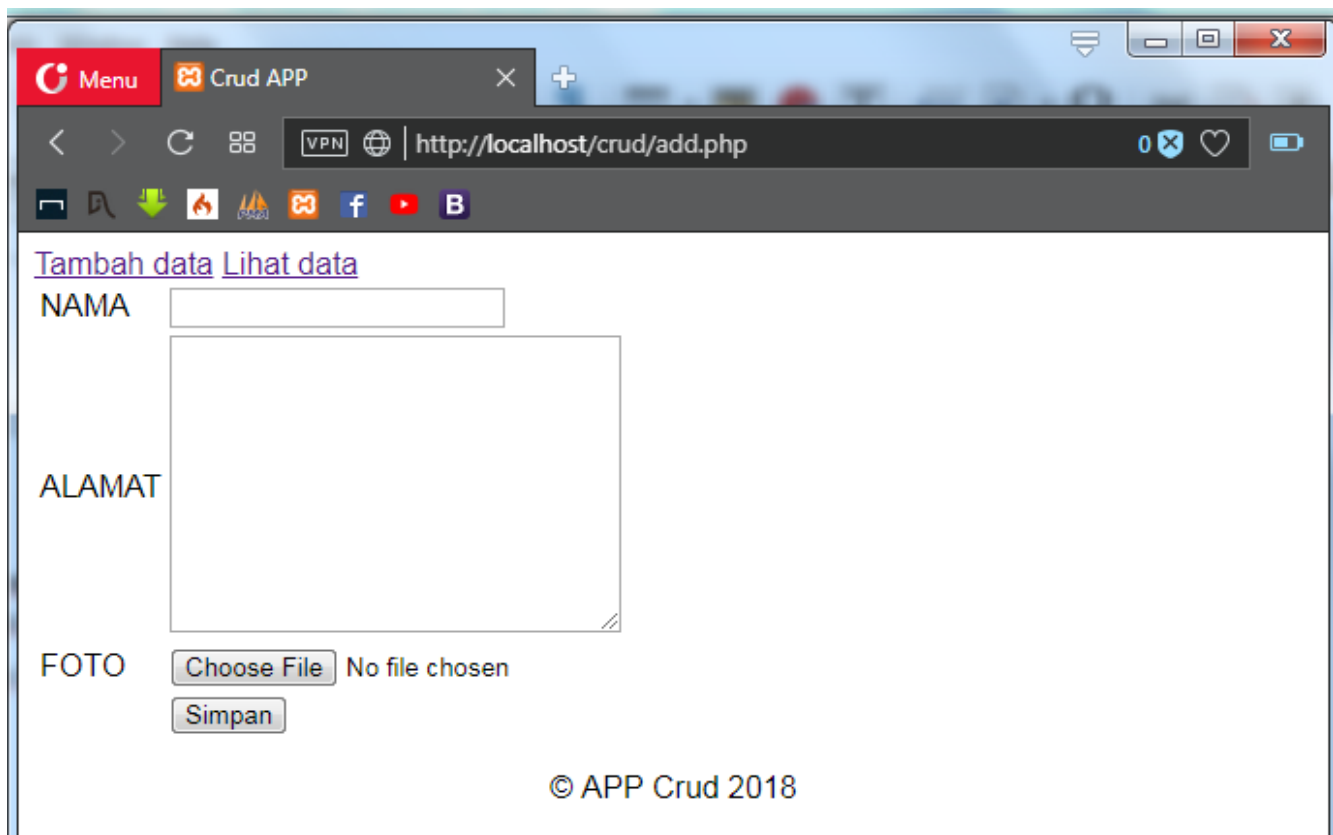
Membuat halaman Tambah data

Buka file **add.php** , ketikkan kode ini :

```
<?php require_once('config.php') ?>
<?php require('header.php') ?>
<form action="pros_add.php" method="post" enctype="multipart/form-data">
    <table>
        <tr>
            <td>NAMA</td>
            <td><input type="text" name="nama"></td>
        </tr>
        <tr>
            <td>ALAMAT</td>
            <td><textarea name="alamat" cols="30" rows="10"></textarea></td>
        </tr>
        <tr>
            <td>FOTO</td>
            <td><input type="file" name="foto"></td>
        </tr>
        <tr>
            <td></td>
            <td><input type="submit" value="Simpan" name="submit"></td>
        </tr>
    </table>
</form>
<?php require("footer.php") ?>
```

Illustration 10: script add.php

Atribut **multipart/form-data** wajib disertakan karena kita tidak hanya mengirim data teks, melainkan foto juga. Terdapat atribut HTML Tabel , untuk merapikan tampilan form
Silahkan buka lagi browser dan klik menu Tambah Data, maka akan tampil seperti ini :



The screenshot shows a web browser window with the title 'Crud APP'. The address bar displays 'http://localhost/crud/add.php'. The page content includes two links at the top: 'Tambah data' and 'Lihat data'. Below these are three form fields: 'NAMA' with a single-line text input, 'ALAMAT' with a large text area, and 'FOTO' with a 'Choose File' button and a 'No file chosen' status. At the bottom right of the form area is a 'Simpan' button. The footer of the page reads '© APP Crud 2018'.

Illustration 11: tampilan Halaman tambah data

Memproses penambahan data

Buka file **pros_add.php** dan ketikkan kode ini :


```

1 <?php
2     if ( isset($_POST['submit']) ) :
3         require_once('config.php');
4         $nama      = $_POST['nama'];
5         $alamat    = $_POST['alamat'];
6         $temporary = $_FILES['foto']['tmp_name'];
7         $nama_foto  = $_FILES['foto']['name'];
8         $encoded_name = date('Y-m-d_H-i-s').urlencode($nama_foto);
9         if ( move_uploaded_file($temporary, $upload.$encoded_name) ) :
10             $sql = "INSERT INTO $table_name (nama, alamat, foto) VALUES ('$nama', '$alamat', '$encoded_name'
11             ')" ;
12             $query = $db->query($sql);
13             if ($query) :
14                 header('Location: show.php?pesan=add_berhasil');
15             else :
16                 header('Location: show.php?pesan=add_gagal');
17             endif;
18         endif;
19     endif;

```

Illustration 12: pros_add.php

Hal pertama yang perlu dilakukan adalah cek apakah telah terdapat data yang dimasukkan user lewat form halaman tambah Data lewat fungsi **isset()** .

Lalu jika user telah memasukkan data, maka sertakan file **config.php** .

Ambil semua yang di masukkan user (**\$_POST**) dan simpan ke variabel. Sementara untuk data file upload , gunakan **\$_FILES** .

Untuk menghindari kesamaan nama file, maka kita tambahkan waktu hari ini dengan fungsi **date()** . Fungsi **urlencode** digunakan untuk merubah karakter tertentu menjadi URL friendly.

Misal nya user mengupload foto dengan nama *"img10.jpg"* maka kode tadi akan mengubah nya menjadi :

2018-02-13_14-19-30img10.jpg

lalu kita pindahkan file hasil upload dari user tadi ke direktori **"file/"** dimana telah diwakilkan dengan variabel **\$upload** yang telah kita konfigurasi pada file **config.php**. Operator **"."**(titik) digunakan untuk menggabungkan dua string menjadi satu.

Pemindahan ini dilakukan dengan fungsi **move_uploaded_file(file temporary, destinasinya)**, dimana fungsi ini akan mengembalikan nilai **TRUE** jika pemindahan berhasil , dan **FALSE** jika gagal. Atas dasar inilah , kita tempatkan penulisannya sebagai syarat blok **IF / percabangan**.

Jika pemindahan file berhasil, maka Jalankan query SQL untuk memasukkan data . Lalu eksekusi dengan method “-> query(perintah SQL)”.

Jika ukuran file melebihi aturan max_filesize maka penambahan data otomatis gagal. Anda dapat mengatur besar maksimal fie upload dengan mengubah file xampp/php/php.ini :

```
; Maximum allowed size for uploaded files.  
; http://php.net/upload-max-filesize  
upload_max_filesize=128M
```

Illustration 13: konfigurasi pada php.ini

Dan jika eksekusi berhasil , alihkan ke halaman **show.php** dengan data **\$_GET** berindex **'pesan'** dengan nilai **add_berhasil**.

Sekarang cobalah untuk tambah data baru dan klik tombol simpan maka akan seperti ini :

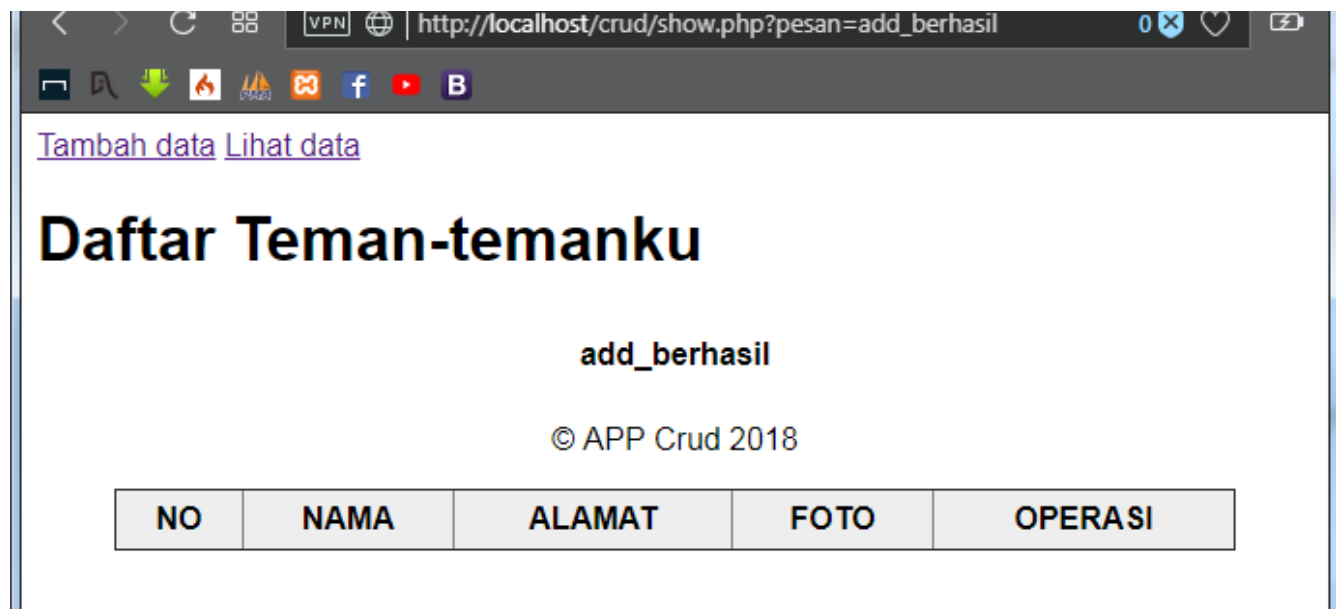


Illustration 14: mana datanya ???

Data belum tampil di halaman show.php namun sebenarnya data telah berhasil masuk ke database, dan gambar juga sudah berhasil terupload :

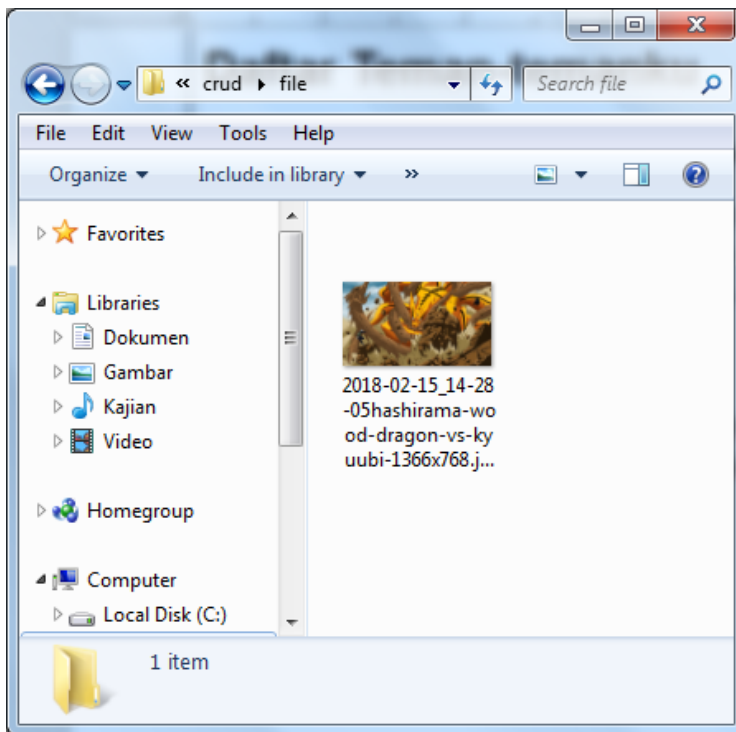


Illustration 15: foto terupload

	id	nama	alamat	foto
	10	Sebuah Nama	Sebuah Tempat	2018-02-15_14-28-05hashirama-wood-dragon-vs-kyuubi...

Illustration 16: data masuk ke database

Kita mau **show.php** dapat menampilkan data yang sudah dimasukkan.

Maka perlu mengeksekusi query SQL untuk menampilkan data Tabel ("SELECT * FROM tabel_name") dan mengeksekusinya lewat method "->query()". Lalu bagaimana menampilkan data nya sehingga dapat terbaca oleh PHP ? Konversi ! Ya! Kita mengkonversi data dari MySQL menjadi data Objek.

Maka tugas kita adalah melakukan perubahan pada file **show.php** :

```

<th>FOTO</th>
<th>OPERASI</th>
</tr>
<?php
    $sql = "SELECT * FROM $table_name";
    $query = $db->query($sql);
    $no = 1;
    while( $data = $query->fetch_object() ) :
        ?>
        <tr>
            <td><?php echo $no ?>.</td>
            <td><?php echo $data->nama ?></td>
            <td><?php echo $data->alamat ?></td>
            <td>
                
            </td>
            <td>
                <a href="<?php echo $base_url.'edit.php?id='.$data->id ?>">Edit</a>
                <a onclick="return confirm('Yakin ingin hapus?')" href="<?php echo $base_url.
                'delete.php?id='.$data->id ?>">Delete</a>
            </td>
        </tr>
    <?php
        $no++;
    endwhile;
    ?>
</table>
<?php require("footer.php") ?>

```

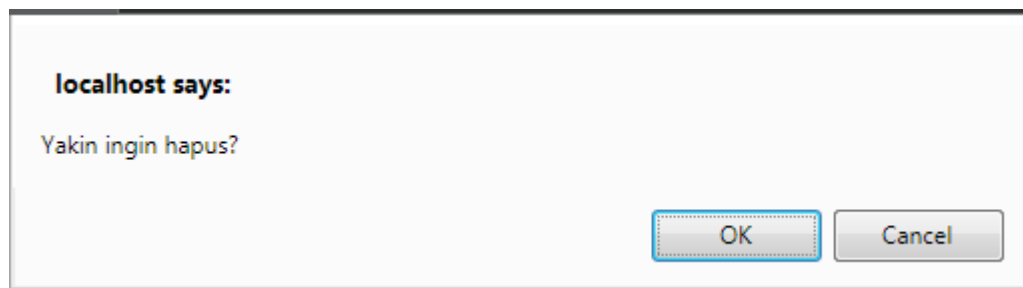
Illustration 17: perubahan file show.php

mengapa kita menggunakan perulangan ? Mengapa juga perulangan jenis **while** ?

Karena kita tidak tau berapa jumlah pasti seluruh data yang ada pada database. Jadi selama data dari database masih ada, maka konversi kan ke object lalu tampilkan.

Kita juga menambahkan link untuk menghapus data dan mengeditnya.

Keyword "return confirm" merupakan perintah Javascript untuk menampilkan konfirmasi semacam ini :



Silahkan lihat kembali halaman **show.php** / klik menu Lihat Data, dan



Data yang kita masukkan tadi berhasil ditampilkan . Jika tidak tampil , periksalah kembali penulisan kode **show.php** pastikan anda telah menyertakan file **config.php** karena kita perlu objek Mysqli untuk operasi ini. Cobalah untuk memasukkan lebih banyak data lagi.

Membuat action 'delete'

Terdapat dua link pada setiap data yang ditampilkan, yaitu **Edit** dan **Delete** . Coba anda klik link **Delete** pada salah satu data , muncul konfirmasi ("Yakin ingin hapus") pilih OK, maka anda akan diarahkan ke halaman kosong dengan URL :

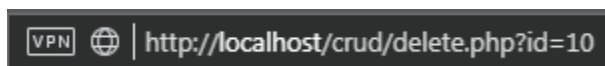


Illustration 18: url untuk delete

terdapat data **\$_GET** berindeks **id** dengan nilai 10 . Bagaimana ini bisa terjadi ?? Jika anda klik link **Delete** pada data yang lain , maka nilai **\$_GET** berindeks **id** ini akan berubah **sesuai dengan id dari data yang bersangkutan** . Kode pada **show.php** inilah yang membuatnya jadi seperti itu:

```
'>" href="<?php echo $base_url.'delete.php?id='.$data->id ?>">
```

Illustration 19: potongan script show.php

Kita tinggal gunakan data **\$_GET** berindeks **id** ini untuk keperluan hapus data, karena setiap data pasti **memiliki id yang berbeda (primary key)** .

Atas dasar konsep tersebut, silahkan buka file **delete.php** , isikan dengan :

```
<?php
if (isset($_GET['id'])) :
    require_once('config.php');
    $id = $_GET['id'];
    $ambil_data = "SELECT foto FROM $table_name WHERE id=$id";
    $file = $db->query($ambil_data)->fetch_object();
    $sql = "DELETE FROM $table_name WHERE id=$id";
    if ( $db->query($sql) ) :
        unlink($upload.$file->foto);
        header('Location: show.php?pesan=hapus_berhasil');
    else :
        header('Location: show.php?pesan=hapus_gagal');
    endif;
endif;
```

Illustration 20: file delete.php

Tentu kita tidak mau hanya menghapus data di Database saja. Kita juga mau, file Foto dari data yang bersangkutan dihapus dari direktori upload kita (folder "file") . Maka kita ambil dulu nama foto dari data yang bersangkutan. Setelah itu kita gunakan fungsi **unlink()** yang berfungsi untuk menghapus suatu file. Silahkan kembali ke Halaman penampilan data, lalu cobalah hapus data tertentu dengan klik link **Delete** .

Antar Tampilan

hapus_berhasil



NO	NAMA	ALAMAT	FOTO	OPERASI
1.	Data dummy	Alamat Dummy		Edit Delete
2.	Ya dummy	Alamat dummy		Edit Delete

Illustration 21: hapus berhasil

Lalu cobalah untuk cek gambar dari file yang bersangkutan pada direktori “file” apakah sudah terhapus atau belum ?

Membuat halaman Edit data

Baik mari kita samakan konsep untuk edit data :

1. Setiap data memiliki **id yang berbeda** (sama dengan konsep delete)
2. Setiap data memiliki foto (file gambar)
3. Jika user ingin melakukan edit, ada 2 pilihan yaitu :
 - a) Mengedit data teks nya saja
 - b) Mengedit data gambarnya (mengubah gambar) atau mengubah keduanya (gambar dan teks)
4. Jika user hanya mengedit data teks saja maka, **data gambar / foto tidak berubah**
5. Jika user mengedit data gambar (foto) maka , **data gambar / foto berubah dan file foto yang lama , akan dihapus.**
6. Tentu saja jika foto nya diganti, maka kita akan melakukan perubahan nama file seperti saat proses [Penambahan data](#) (dengan fungsi **urlencode()** dsb)

Baik silahkan pahami beberapa konsep diatas, karena kita akan mengimplementasikan nya kedalam script **pros_edit.php** .

Namun sebelum memproses pengeditan data dari user, kita perlu membuat halaman untuk mengeditnya. Silahkan buka file **edit.php** , isikan dengan kode ini :

```

<?php
    if (isset($_GET['id'])) :
        require_once('config.php');
        $id = $_GET['id'];
        $ambil_data = "SELECT * FROM $table_name WHERE id=$id";
        $data = $db->query($ambil_data)->fetch_object();
    ?>
    <?php require('header.php') ?>
    <form action="pros_edit.php" method="post" enctype="multipart/form-data">
        <input type="hidden" name="id" value="<?php echo $data->id ?>">
        <input type="hidden" name="foto_lama" value="<?php echo $data->foto ?>">
        <table>
            <tr>
                <td>NAMA</td>
                <td><input type="text" name="nama" value="<?php echo $data->nama ?>"></td>
            </tr>
            <tr>
                <td>ALAMAT</td>
                <td><textarea name="alamat" cols="30" rows="10"><?php echo $data->alamat ?></textarea></td>
            </tr>
            <tr>
                <td>FOTO</td>
                <td>
                    
                    <input type="file" name="foto">
                </td>
            </tr>
            <tr>
                <td></td>
                <td><input type="submit" value="Simpan" name="submit"></td>
            </tr>
        </table>
    </form>
    <?php require('footer.php') ?>
    <?php
        endif;

```

Illustration 22: halaman edit.php

Perhatikan baris script ini :

```

<input type="hidden" name="id" value="<?php echo $data->id ?>">
<input type="hidden" name="foto_lama" value="<?php echo $data->foto ?>">

```

Illustration 23: fokusss!

Kita menyimpan id dari data yang akan diedit dalam sebuah elemen HTML , namun kita tidak menampilkannya karena kita memberinya atribut **"type=hidden"** . Begitu juga dengan data nama foto , karena kita akan membutuhkan nya jika user memutuskan untuk mengganti foto.

Silahkan ke halaman penampilan data dan klik link **Edit** pada salah satu data , maka akan tampil seperti ini :

Illustration 24: yuk ngediit

Semua data yang tampil pada atribut HTML Form diatas , didapat dari script :

```
$id = $_GET['id'];
$ambil_data = "SELECT * FROM $table_name WHERE id=$id";
$data = $db->query($ambil_data)->fetch_object();
```

Illustration 25: fokusss

Mengapa kita tidak menggunakan perulangan **while** seperti yang kita lakukan saat [menampilkan data](#) ? Karena kita sudah tau pasti bahwa jumlah data yang akan kita dapat kan dari **query \$sql** diatas adalah satu data saja. Mengapa satu ? Kembali ke fakta **bahwa setiap data mempunyai id yang berbeda dari data lain** . Sehingga tidak mungkin kita mendapatkan lebih dari satu data.

Memproses edit data..

Buka file pros_edit.php dan mari ketikkan kode-kode ini :

```

<?php
if ( isset($_POST['submit']) ) :
    require_once('config.php');
    $nama      = $_POST['nama'];
    $alamat    = $_POST['alamat'];
    $id        = $_POST['id'];
    $foto_lama = $_POST['foto_lama'];
    $edit_foto = FALSE;
    if ( empty($_FILES['foto']['name']) ) :
        $sql = "UPDATE $table_name SET nama='$nama', alamat='$alamat' WHERE id=$id";
    else :
        $edit_foto = TRUE;
        $temporary = $_FILES['foto']['tmp_name'];
        $nama_foto = $_FILES['foto']['name'];
        $encoded_name = date('Y-m-d_H-i-s').urlencode($nama_foto);
        if ( move_uploaded_file($temporary, $upload.$encoded_name) ) :
            $sql = "UPDATE $table_name SET nama='$nama', alamat='$alamat', foto='$encoded_name' WHERE id=$id";
        endif;
    endif;
    if ( $db->query($sql) ) :
        if ($edit_foto) :
            unlink($upload.$foto_lama);
        endif;
        header('Location: show.php?pesan=edit_berhasil');
    else :
        header('Location: show.php?pesan=edit_gagal');
    endif;
endif;

```

Illustration 26: memproses edit

Untuk memeriksa apakah user mengupload gambar baru, maka kita gunakan fungsi **empty(variabel)** dimana fungsi ini akan memeriksa variabel yang dijadikan parameternya. Jika variabel isinya kosong maka fungsi ini akan mengembalikan nilai TRUE dan FALSE jika variabel tsb ada isinya.

Jika ukuran file gambar melebihi [settingan pada php.ini](#) maka proses edit data akan gagal.

Jika anda masih bingung dengan kode diatas silahkan kembali pahami [konsep-konsep yang saya berikan tadi](#).

Silahkan kembali ke halaman penampilan data dan klik link edit pada salah satu data , ubah beberapa data yang ada , ganti fotonya lalu cek apakah foto lama ikut terhapus (lihat di direktori "file") dan cek juga data teks lain apakah ikut terganti.


Berikut contohnya :

[Tambah data](#) [Lihat data](#)

NAMA

ALAMAT

Alamat Dummy ya

FOTO  Forrest_(imag... Bilcliff).jpg

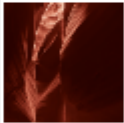

© APP Crud 2018

Illustration 27: edit data teks dan foto nya sekalian

[Tambah data](#) [Lihat data](#)

Daftar Teman-temanku

edit_berhasil

NO	NAMA	ALAMAT	FOTO	OPERASI
1.	Data dummy ya	Alamat Dummy ya		Edit Delete
2.	Ya dummy	Alamat dummy		Edit Delete

© APP Crud 2018

Illustration 28: data berhasil teredit

SELESAI SUDAH...

Akhirnya selesai sudah semua tahapan dasar Aplikasi CRUD sederhana ini. Penulis berharap , pembaca dapat memahami dan mengerti Logika dari CRUD ini karena Logika CRUD ini merupakan dasar dari berbagai aplikasi berbasis Web dan Database.

Pembaca dapat mengembangkan fitur dari aplikasi ini seperti memeriksa file upload dari user , perbaikan tampilan dan tentu saja logika yang lebih baik dari yang diberikan oleh Panduan ini.

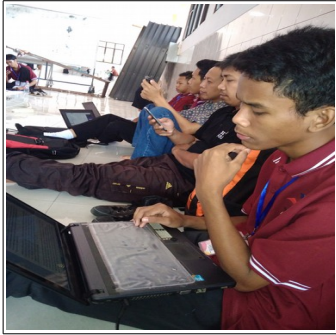
Penulis menyadari , bahwa panduan ini masih banyak kekurangan dan kurang lengkap penjelasannya. Pembaca dapat langsung mengunjungi manual resmi PHP melalui pranala <http://www.php.net> untuk mendapatkan panduan dan referensi dari fungsi dan fitur yang dimiliki PHP.

Jika gambar, kode dan beberapa tampilan dari Ebook ini kurang jelas, Pembaca dapat langsung membuka file “utama.odt” atau membuka folder “images” yang keduanya disertakan dalam paket ZIP bersama Ebook ini.

Seluruh **source code** yang telah saya beri banyak baris komentar dan **file dump** database dari Aplikasi telah saya sertakan bersama dengan paket ZIP Lengkap Ebook ini. Jika pembaca tidak menemukan nya, bisa pembaca dapatkan melalui repositori daring penulis dengan pranala <http://github.com/taufiq33/crud-tutorial>

Seluruh isi dari Panduan dan **source code** dari aplikasi ini bebas untuk disebarluaskan, dimodifikasi, dan digunakan untuk kepentingan apa saja dengan atau tanpa ijin dari penulis.

Tentang Penulis



Taufiq Hidayat, saat sedang menulis panduan ini, aktif sebagai siswa di SMK Negeri 1 Sukoharjo Jurusan Teknik Komputer dan Jaringan. Menyukai pemrograman dan apa saja yang berbau IT. Serius mempelajari *Python* demi mencapai impian untuk bisa membuat aplikasi berbasis *GUI*. Penulis dapat dihubungi melalui alamat surel taufiqhidayat1908@gmail.com. Atau alamat repositori <http://www.github.com/taufiq33>