

# Cara Mudah Membuat Blog Dengan Codeigniter 1.7.2

By : Puji Rahmadiyanto

CD :

CodeIgniter 1.7.2

Xampp 1.7.0

NetBeans IDE 6.8

Contoh blog

Template Web

Tinymce 3.3.4

Jquery PI



## Profil

Nama : Puji Rahmadiyanto

Alamat : Jatisarono , Nanggulan, Kulon Progo  
 , Yogyakarta

Tempat/tgl lahir : Kulon Progo /29 Mei 1990

Sekolah : SD N Jatisarono , SMP N 1 Nanggulan  
 SMA N 1 Sentolo

Kuliah : Universitas Ahmad Dahlan Fakultas Teknologi Industri  
 Program Studi Teknik Informatika 2008

About me :



Aku Puji Rahmadiyanto sering di panggil Puji, kuliah di Universitas Swasta di Kota Jogja. Pada awal perkuliahan sempat stress dengan matakuliah yang banyak itung-itungannya tapi I like it hehe :D . tidak mengenal bahasa pemrograman apapun

Dulu ada mata kuliah pemrograman C++ , ku pikir C++ adalah resep obat .. e ternyata bahasa pemrograman dari C (apa lagi tu gak ngerti pada saat itu ). Lama-kelamaan C++ akrab dengan ku, ternyata menarik juga. Nah dari situ sampai sekarang aku gemar coding

Pada awal semester 3 aku coba-coba untuk mendaftar menjadi asisten praktikum C++ dan di terima. Ternyata dari situ aku mendapatkan banyak ilmu.

Sebenarnya Di perkuliahan aku tidak menadapatkan ilmu apa-apa (jujur) ilmu-ilmu itu aku peroleh dari kakak tingkat teman asisten. dari situ pula aku kenal dengan PHP,CMS lanjut Frame Work

Dan sekarang “Tiada hari tanpa coding” semboyan hidupku, mungkin cita-cita jadi programmer ini dulu gak terbersit di dalam benakku

Cita-citaku ini bertolak belakang dengan target S1 TI yaitu jadi seorang analis( kata dosenku) tapi aku malah pengen jadi programmer. Tapi gak papa lah hidup adalah sebuah pilihan jika kita tekun menjalaninya maka kesuksesanlah yang kita dapat

Mengapa aku memilih Codeigniter ? hmm ... codeigniter menurutku banyak tutorialnya. Mudah , bisa di integrasikan dengan ajax , sudah dilengkapi dengan user guide, open source ...

Pokoknya Hidup Codeigniter ... hehe :D

## PHP

### Kisah Sang PHP

Web pada kisah awalnya sangat menjemukan bagi orang-orang yang dinamis. Bagaimana tidak, pemakainya hanya dicekoki oleh isi (*content*) halaman web yang meskipun bersifat saling terhubung dengan halaman web yang lain (*hyperlink*) tetap saja tidak memberikan saluran bagi pengguna yang ingin mengemukakan pendapatnya. Tidak ada demokrasi, karena pengguna hanya bersifat pasif dan tidak bisa berinteraksi secara aktif dalam web.

Ketika akhirnya ditemukan *tag* <FORM> barulah kejemuan dan kebuntuan yang ada menjadi sirna. Pengguna menjadi bisa secara aktif berinteraksi dengan halaman web, dan mulailah era aplikasi berbasis web yang dinamis. Secara tradisi, bahasa *script* Perl menjadi bahasa utama yang digunakan oleh *programmer* web untuk menangani pemrosesan form dalam berinteraksi dengan pengguna web. Tidak diragukan lagi kedigjayaan dari Perl dalam menangani urusan ini, hal ini juga didukung dengan begitu dominannya bahasa ini digunakan di situs-situs web yang ada.

Perl bisa menjadi alat bantu yang sangat hebat di tangan ahlinya, namun akan berubah menjadi mimpi paling buruk bagi seorang *programmer* web pemula yang dikejar waktu dan bosnya untuk segera merilis halaman webnya. Tidak mudah memang, mempelajari bahasa Perl, dan seringkali dibutuhkan langkah panjang dan rumit untuk sebuah maksud yang sederhana saja. Pendek kata, dibutuhkan suatu bahasa yang lebih praktis dan mudah dipelajari serta adidaya untuk memudahkan dalam membangun sebuah aplikasi yang berbasis web.

Di rimba belantara web, tersebutlah dua bahasa yang paling kondang yang mampu menggantikan tugas-tugas Perl namun dengan tingkat kesulitan belajar yang rendah, ASP (*Active Server Page*) dan PHP (*PHP: Hypertext Preprocessor*). ASP yang dijagokan oleh Pak Bill Gates tentu saja berjalan di lingkungan sistem operasi Windows dan sampai saat ini belum terlihat akan di-*porting* ke platform yang lain. Padahal dunia web saat ini masih didominasi oleh platform UNIX dan *variant*-nya termasuk sistem operasi *like UNIX* seperti Linux. Selain itu, untuk dapat menggunakan ASP yang resmi, kita juga harus merelakan sebagian uang kita untuk menambah isi kantong Pak Bill Gates.

PHP sebagai alternatif lain memberikan solusi sangat murah (karena gratis digunakan) dan dapat berjalan di berbagai jenis platform. Awalnya memang PHP berjalan di sistem UNIX dan *variant*-nya, namun kini dapat berjalan dengan mulus di lingkungan sistem operasi Windows. Suatu nilai tambah yang luar biasa karena proses *development* program berbasis web dapat dilakukan lintas sistem operasi. Pak Fulan, misalnya, bisa mencuri waktu memprogram aplikasi untuk usaha pribadinya di kantor yang menggunakan sistem operasi Windows dan meneruskannya di rumahnya dengan komputer yang menggunakan sistem operasi Linux.

Dengan luasnya cakupan sistem operasi yang mampu menjalankan PHP dan ditambah begitu lengkapnya fungsi-fungsi program (tersedia lebih dari 400 fungsi di



PHP yang sangat berguna) tidak heran jika PHP ini semakin menjadi *trend* di kalangan *programmer* web. Konon, saat ini lebih dari satu juta situs web menggunakan PHP sebagai *script* pemrogramannya.

Pak Rasmus Lerdorf adalah bapak penemu awal bahasa PHP ini, yang bermula dari keinginan sederhana Pak Lerdorf untuk mempunyai alat bantu (*tools*) dalam memonitor pengunjung yang melihat situs web pribadinya. Inilah sebabnya pada awal pengembangannya, PHP merupakan singkatan dari *Personal Home Page tools*, sebelum akhirnya dipaksakan menjadi singkatan rekursif dari *PHP: Hypertext Preprocessor*. Pertengahan tahun 1995 dirilis PHP/FI (FI adalah singkatan dari *Form Interpreter*) yang memiliki kemampuan dasar membangun aplikasi web, memproses form, dan mendukung database mSQL.

Antusias komunitas internet terhadap bahasa PHP ini begitu besar, sehingga Pak Rasmus Lerdorf akhirnya menyerahkan pengembangan PHP ini kepada sebuah team pemrograman dalam kerangka gerakan *open source*. Team ini membangun kembali PHP dari awal dengan menulis ulang program *parser* PHP Hasilnya adalah PHP 3.0 yang memiliki dukungan lebih luas lagi terhadap database yang ada termasuk MySQL dan Oracle. PHP 4.0 sebagai versi lanjutan dari PHP 3.0 dirilis setelah itu dengan menggunakan mesin *scripting* Zend (akronim dari pengembangnya, Zeev Suraski dan Andi Gutmans) untuk memberikan kinerja yang lebih cepat dan lebih baik Versi terakhir ini mampu mendukung server web selain Apache dan secara *built-in* telah mampu menangani manajemen *session*.

Singkat kata, PHP kita pilih sebagai bahasa untuk pengembangan web yang akan kita pelajari di bagian selanjutnya. Sebelum memulainya, ada baiknya Anda mengetahui kebutuhan-kebutuhan dasar yang akan membantu Anda memahami tulisan ini. Anda diasumsikan telah memiliki sebuah sistem yang telah terinstalasi dan terkonfigurasi dengan baik Apache Web Server, PHP 4, dan database MySQL. Ketiganya adalah program *open source* yang tersedia secara gratis di Internet dan dapat berjalan di berbagai platform (Windows maupun UNIX/Linux).

Seperti halnya Codeigniter adalah framework PHP jadi fungsi-fungsi PHP murninya sudah di masukan kedalam Codeigniter

## FrameWork

Anda sebagai seorang programmer PHP pasti pernah mengalami kejenuhan dalam membuat sebuah aplikasi web, kadang sebuah ritme yang monoton pasti sering anda alami dalam membuat sebuah web dinamis entah itu untuk website klien atau web untuk anda sendiri. Sebetulnya dari sebuah ritme tersebut ada beberapa hal yang dapat dibuat permanen dari struktur dasar kode program anda.

Mempelajari *Object Oriented Programming* (OOP) dalam PHP sangat penting Tujuan pemrograman menggunakan class (OOP) disini adalah *code reuse*, yang berarti kode program anda dapat di gunakan kembali untuk project-project web yang lainnya. Karena pada intinya dalam sebuah program yang berhubungan dengan

database, variable yang selalu berubah-ubah adalah struktur database dan user interface. Dimana struktur database dan tampilan user interface ini selalu mengikuti alur business process dari pengguna program anda.

*Framework* adalah kerangka kerja siap bangun yang sudah dilengkapi library dan fungsi-fungsi. beda *framework* dengan CMS (*content management system*) jika framework kita benar-benar membangun web tetapi sudah disediakan library dan fungsi-fungsi yang mendukung sedangkan CMS adalah web siap guna sudah lengkap dan tidak repot-repot coding

Saat ini banyak sekali framework PHP yang ditawarkan di internet seperti Code Igniter, Cake PHP, Zend Framework, VCL Delphi PHP, Blue Shoes, dan masih banyak lagi

## Sejarah Singkat Codeigniter



CodeIgniter awalnya dikembangkan oleh [Rick Ellis](#) (CEO [Ellislab, Inc.](#)). Framework ini ditulis untuk kinerja di dunia nyata, dengan banyak class libraries, helpers, dan sub-systems yang disediakan dari kode-dasar [ExpressionEngine](#).

Saat ini CI dikembangkan dan dikelola oleh Tim Pengembangan ExpressionEngine.

Gambar 1.1

## Pemahaman Dasar mengenai Codeiniter(CI)

Apa itu codeigniter(CI) ?

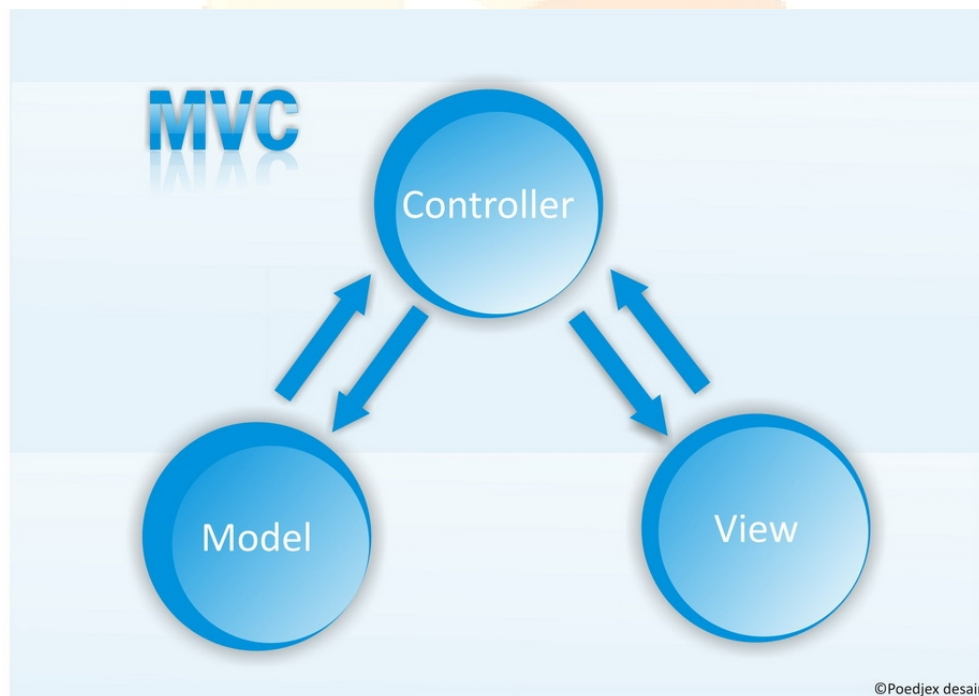
CodeIgniter adalah sebuah Application Development Framework - a toolkit - bagi orang-orang yang membangun situs web menggunakan PHP. Tujuannya adalah untuk memungkinkan untuk mengembangkan proyek-proyek yang jauh lebih cepat daripada menulis kode dari awal atau sering menyebutkan PHP murni, dengan menyediakan library-library yang dibutuhkan untuk tugas-tugas, serta antarmuka

yang sederhana dan struktur logika untuk mengakses libraries tersebut. CodeIgniter meminimalkan jumlah sintak untuk tugas tertentu.

#### Pertimbangan-Pertimbangan

- Anda menginginkan kerangka kerja dengan proses cepat.
- Anda membutuhkan kinerja luar biasa.
- Anda perlu luas kompatibilitas dengan account hosting standar yang menjalankan berbagai versi dan konfigurasi PHP.
- Anda menginginkan sebuah kerangka kerja yang memerlukan konfigurasi hampir nol.
- Anda menginginkan sebuah kerangka kerja yang tidak mengharuskan Anda untuk menggunakan baris perintah.
- Anda menginginkan sebuah kerangka kerja yang tidak mengharuskan Anda untuk mematuhi aturan pengkodean ketat.
- Anda tidak tertarik pada skala besar seperti PEAR libraries monolitik.
- Anda tidak mau dipaksa untuk belajar bahasa template (meskipun template parser adalah opsional tersedia jika Anda inginkan).
- Anda menghindari kerumitan, menyukai solusi sederhana.
- Anda perlu jelas, teliti dokumentasi.

#### Model-View-Controller



Gambar 1.2

CodeIgniter didasarkan pada Model-View-Controller pola pengembangan. MVC adalah sebuah pendekatan perangkat lunak aplikasi yang memisahkan logika dari antarmuka. Dalam praktiknya, hal itu memungkinkan halaman web Kamu berisi minimal scripting sejak antarmuka terpisah dari PHP scripting.

- Model mengkomunikasikan dengan struktur data. Biasanya kelas-kelas model akan berisi fungsi yang membantu Kamu mengambil, memasukkan, dan mengupdate informasi dalam database.
- View adalah informasi yang sedang disajikan kepada user. Yang bisa dilihat di halaman web, tetapi dalam CodeIgniter, view juga dapat dipisah-pisah menjadi halaman seperti header atau footer
- Controller berfungsi sebagai *perantara* antara Model, View, dan sumber daya lainnya yang dibutuhkan untuk memproses permintaan HTTP dan menghasilkan suatu halaman web.

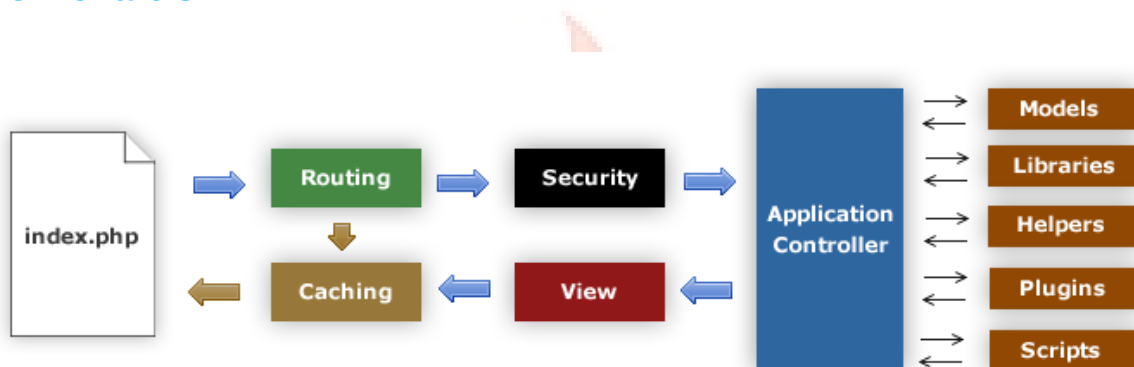
CodeIgniter memiliki pendekatan yang cukup longgar untuk MVC Model karena tidak diperlukan. Jika anda tidak memerlukan Model, dan menemukan bahwa mempertahankan kompleksitas model, anda dapat mengabaikan model dan membangun aplikasi seminimal mungkin dengan menggunakan Controller dan Views saja. CodeIgniter juga memungkinkan kamu untuk memasukkan script buatan anda sendiri, atau bahkan mengembangkan libraries inti untuk sistem

#### Fitur-fitur di dalam CI

- Model-View-Controller Based System
- PHP 4 Compatible
- Extremely Light Weight
- Full Featured database classes with support for several platforms.
- Active Record Database Support
- Form and Data Validation
- Security and XSS Filtering
- Session Management
- Email Sending Class. Supports Attachments, HTML/Text email, multiple protocols (sendmail, SMTP, and Mail) and more.
- Image Manipulation Library (cropping, resizing, rotating, etc.). Supports GD, ImageMagick, and NetPBM
- File Uploading Class
- FTP Class
- Localization
- Pagination
- Data Encryption
- Benchmarking
- Full Page Caching
- Error Logging
- Application Profiling
- Scaffolding
- Calendaring Class
- User Agent Class

- Zip Encoding Class
- Template Engine Class
- Trackback Class
- XML-RPC Library
- Unit Testing Class
- Search-engine Friendly URLs
- Flexible URI Routing
- Support for Hooks, Class Extensions, and Plugins
- Large library of "helper" functions

## Flow chart CI



Gambar 1.3

Dapat dilihat di flowchart di atas Controller bisa di ibaratkan sebagai polisi lalulintas

index.php berfungsi sebagai pengendali depan, menginisialisasi basis sumber daya yang dibutuhkan untuk menjalankan CodeIgniter.

Router memeriksa HTTP request untuk menentukan apa yang harus dilakukan dengan hal itu.

Jika file cache ada, maka langsung dikirim ke browser, melewati sistem normal eksekusi.

## Codeigniter Mempermudah Anda

Perbedaan sintak codeigniter bertujuan untuk mempermudah dan mempercepat coding

Jika dengan penulisan link dengan tag html :

```
<a href="contoh.com">contoh</a>
```

Di codeigniter



```
<?php =anchor('contoh.com','contoh');?>
```

Kelihatannya emang agak sedikit rumit tapi jika ditulis dengan PHP akan panjang

Sintak php echo bisa menggunakan =

```
<?='hello word'?> sama halnya dengan <?php echo 'hello word';?>
```

## Gaya Coding Di Codeigniter

Codeigniter menggunakan class-class dan function

### Penulisan Controller

```
Class namacontroller extends Controller{

    function namacontroller(){

        parent::Controller();

    }

    function namafungsitambahan(){

        //penjabaran fungsi

    }

}
```

### Penulisan Model

```
Class namamodel extends Model{

    function namamodel(){

        parent::Model();

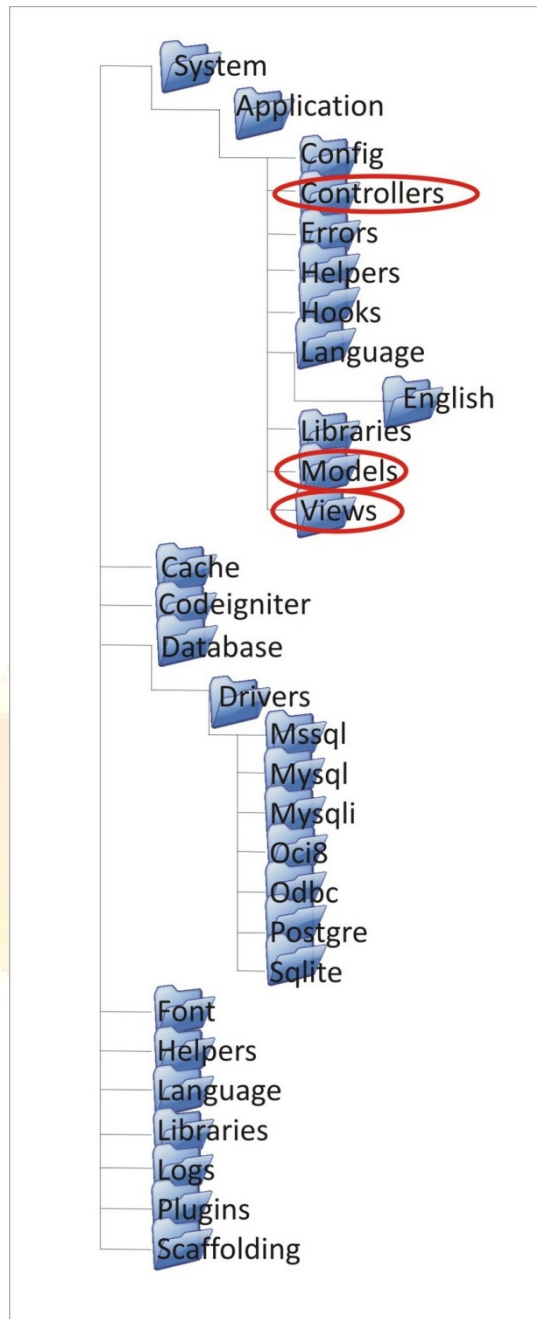
    }

    function tambahan(){

    }

}
```

## Struktur folder paket CI



Gambar 1.4

## Controller

Apa itu Controller ?

Controller adalah sebuah file class yang dapat dikaitkan dengan URI.

Controller juga bertindak sebagai polisi lalu lintas antara Model dan View



Gambar 1.5

## Penulisan

Class namacontroller extends Controller{

function namacontroller(){

parent::Controller();

}

function namafunksitambahan(){

//penjabaran fungsi

}

}

Constructor harus dibuat di awal dengan membuat nama function sama dengan nama class atau kalau dengan PHP 5 **\_\_construct()**

Parent::Controller();

Menunjukkan bahwa controller yang telah kita buat adalah class turunan dari class controller

Contoh :

controller bernama contohcontroller.php

```
<?php

class Contohcontroller extends Controller{

    function Contohcontroller(){

        parent::Controller();

    }

    function index(){

        echo "controller contohcontroller sedang berjalan";

    }

    function fungsi(){

        echo "function fungsi pada contohcontroller sedang berjalan ";

    }

}
```

Pemanggilan di urlnya <http://localhost/contohci/index.php/contohcontroller/>

Output nya adalah yang berada di function index()

**controller contohcontroller sedang berjalan**

Sedangkan untuk memanggil function fungsi() cukup tambahkan nama function tersebut di belakang sendiri

<http://localhost/contohci/index.php/contohcontroller/fungsi>

output nya

**function fungsi pada contohcontroller sedang berjalan**

## VIEW

View adalah Sebuah tampilan pada halaman web, atau halaman fragmen, seperti sebuah header, footer, sidebar, dll

View tidak pernah panggil secara langsung, view harus dimuat pada sebuah [Controller](#). Ingat bahwa dalam kerangka MVC, Controller bertindak sebagai polisi lalu lintas, sehingga bertanggung jawab untuk mengambil view tertentu

## Membuat View

Pada view sebenarnya cukup mudah karena bermain dengan HTML ataupun PHP



Misalkan contohview.php

```
<html>

<head>

<title>Contoh CI</title>

</head>

<body>

    <h1>Selamat datang</h1>

</body>

</html>
```

View diatas tidak mungkin bisa dipanggil langsung di browser kalau tidak ada controllernya

Misalnya

```
<?php

class Contohcontroller extends Controller{

    function Contohcontroller(){

        parent::Controller();

    }

    function index(){

        $this->load->view("contohview");

    }

}

?>
```

`$this->load->view("contohview");` bertindak mengload file view bernama contohview.php Dalam penulisannya tidak menggunakan .php untuk mencobanya  
<http://localhost/contohci/index.php/contohcontroller>

View dengan beban data

Misalnya di controller

```
function index(){

    $data['title']="hello word";

    $this->load->view('contohview',$data);

}
```

Controller memberikan beban data bernama **title** kedalam view

Di view ,**title** dianggap sebagai variable menjadi **\$title**

```
<html>

<head>

<title>Contoh CI</title>

</head>

<body>

    <h1><?php echo $title;?> </h1>

</body>

</html>
```

## Model

Model adalah class yang bertugas sebagai pengolah database

```
<?php

class Contohmodel extends Model{

    function Contohmodel(){

        parent::Model();

    }

    function ambildata(){

        $query =$this->db->get('artikel');

        return $query;

    }

    function insertdata($judul,$isi){

        $data=array('judul'=>$judul,

            'isiartikel'=>$isi
```

```
);

$this->db->insert('artikel',$data);

}

function updatedata($id,$judul,$isi){

    $this->db->where('id',$id);

    $this->db->update('artikel',array('judul'=>$judul,'isiartikel'=>$isi));

}

}

?>
```

## Pemanggilan nama model di controller

```
<?php

class Contohcontroller extends Controller{

    function Contohcontroller(){

        parent::Controller();

        $this->load->model('contohmodel'); //memanggil model bernama contohmodel.php

    }

    function index(){

        $data['title']="hello word";

        $data['content']=$this->contohmodel->ambildata(); //ambil data merupakan nama fungsi yang terdapat di contohmodel

        $this->load->view('contohview',$data);

    }

}

?>
```

## Konfigurasi database terletak di

Autoload.php –( system/application/config/autoload.php)

```
$autoload['libraries'] = array('database');
```

database.php –( system/application/config/database.php)

```
$db['default']['hostname'] = "localhost";
```

```
$db['default']['username'] = "root";
```

```
$db['default']['password'] = "";
```

```
$db['default']['database'] = ""; //nama database
```

```
$db['default']['dbdriver'] = "mysql";
```

## DATABASE

### 1. Query standard dengan hasil multiple (object version)

```
$query="select nama,alamat,telp from mahasiswa";
```

```
$hasil=$this->db->query("$query");
```

```
foreach ($hasil as $row)
```

```
{
```

```
    echo $row->nama;
```

```
    echo $row->alamat;
```

```
    echo $row->telp;
```

```
}
```

```
echo "Hasil Nilai :". $hasil->num_rows();
```

### 2. Query standard dengan hasil multiple (Array version)

```
$query="select nama,alamat,telp from mahasiswa";
```

```
$hasil=$this->db->query("$query");
```

```
foreach ($hasil as $row)
```

```
{
```

```
    echo $row['nama'];
```

```
    echo $row['alamat'];
```

```
    echo $row['telp'];
```

```
}
```

```
echo "Hasil Nilai :". $hasil->num_rows();
```



### 3. Query dengan hasil tunggal

Untuk menampilkan sebuah hasil query, kita dapat menggunakan syntax sbb :

```
$query->row()
```

Contoh :

```
$query="select nama from mahasiswa limit 1";  
  
$hasil=$this->db->query("$query");  
  
$hasilnya=$hasil->row();  
  
echo $hasilnya->nama;
```

### 4. Query Bindings

Yaitu membuat query dengan nilai yang dapat berubah-ubah.

```
$sql="select * from mahasiswa where nama=? And kota=?";
```

```
$this->db->query($sql,array('andi','jakarta');
```

### 5. Active Record Class

CI menggunakan teknik active record, di samping juga dapat memproses perintah sql. Cara

ini adalah cara lain yang di gunakan di CI untuk memproses query.

Perintah-perintah di dalam Active record class adalah sbb :

#### **a. Selecting Data**

- `$this->db->get()`

Fungsi : Untuk menampilkan semua isi tabel mahasiswa.

Contoh :

- `$this->db->get('mahasiswa');`

// Sama dengan : `select * from mahasiswa.`

Parameter pertama dan kedua memuat limit dan offset, yaitu :

- `$query=$this->db->get('mytable',10,20)`

Menghasilkan : `select * from mytable limit 10,20`

- `$this->db->get_where();`

Fungsi : Untuk menampilkan semua isi tabel dengan di tambah kondisi where

Contoh :

```
$this->db-> get_where ('mytable',array('id'=>$id),$limit,$offset);
```

// Sama dengan : "select \* from mytable where id='\$id' limit 20, 10";

- **\$this->db->select();**

Fungsi : Untuk memilih tabel yang akan di proses dengan perintah select.

Contoh :

```
$this->db->select('nama,nilai,alamat');
```

```
$query=$this->db->get('mahasiswa');
```

// Sama dengan : select nama,nilai,alamat from mahasiswa;

- **\$this->db->from();**

Fungsi : Untuk memilih tabel.

Contoh :

```
$this->db->select('nama,nilai,alamat');
```

```
$this->db->from('mahasiswa');
```

```
$query=$this->db->get();
```

```
// Sama dengan : select nama,nilai,alamat from mahasiswa;
```

- **`$this->db->join();`**

Fungsi : Untuk melakukan perintah join terhadap 2 atau lebih tabel.

Contoh :

```
$this->db->select("*");
```

```
$this->db->from("nilai");
```

```
$this->db->join("mahasiswa","mahasiswa.nim=nilai.nim");
```

```
$query=$this->db->get();
```

```
// Sama dengan : "select * from nilai join mahasiswa on mahasiswa.nim=nilai.nim";
```

- **`$this->db->where();`**

Fungsi : Untuk menerapkan kondisi where suatu syntax query.

Contoh :

```
$this->db->where('nama','opan');
```

```
$query= $this->db->get('mahasiswa');
```

```
// sama dengan : "select * from mahasiswa where nama='opan';
```



- `$this->db->like();`

Fungsi : Menyatakan syntax like ke dalam query.

Contoh :

```
$this->db->like('nama','sofwan');
```

```
$query=$this->db->get("mahasiswa");
```

// sama dengan : "select \* from mahasiwa where nama like '%sofwan%'";

- `$this->db->group_by()`

Fungsi : Menambahkan perintah group by pada query.

Contoh :

```
$this->db->group_by("kota");
```

```
$query=$this->db->get("mahasiswa");
```

// Sama dengan : select \* from mahasiswa group by kota";

## **b.Inserting Data**

- `$this->db->insert();`

Fungsi : Untuk menginsert data ke dalam sebuah tabel.

Kita dapat menggunakan data yang akan di insert berupa array atau object.

Contoh, menggunakan array :

```
$data=array(

    'nim'=>'0811500292',

    'nama'=>'ali',

    'kota'=>'jakarta');

$this->db->insert('mahasiswa',$data);
```

Contoh menggunakan Object :

```
class kelasku

{ var $nim="0811500292",

    var $nama="ali",

    var $kota="jakarta"}

$obj=new kelasku;

$this->db->insert("mahasiswa",$obj);
```

Kedua contoh di atas sama dengan perintah :

```
// insert into mahasiswa (nim,nama,kota) values('0811500292','ali','jakarta');
```

- `$this->db->set();`

Fungsi : Fungsi ini mengambil data untuk di lakukan perintah insert dan update.

Contoh : `$this->db->set('nama',$nama);`

```
$this->db->insert('mahasiswa');
```

// Sama dengan : `insert into mahasiswa (nama) values ('{$nama}');`

### c.Updating Data

- `$this->db->update();`

Fungsi : Untuk update data

```
$data=array(
```

```
    nim=>'$nim',
```

```
    'nama'=>'$nama',
```

```
'kota'=>'$kota');
```

```
$this->db->where('id',$id);
```

```
$this->db->update('mahasiswa',$data);
```

// Sama dengan : update mahasiswa set nim="\$nim",nama="\$nama",kota="\$kota"

```
where id="$id";
```

Atau dapat juga dengan menggunakan object.

```
Class kelasku {
```

```
var $nim="$vnim",
```

```
var $nama="$vnama",
```

```
var $kota="$vkota";
```

```
}
```

```
$object = new kelasku;
```

```
$this->db->where ('id',$id);
```

```
$this->db->update ("mahasiswa",$object);
```



#### d.Deleting Data

- `$this->db->delete()`

Fungsi : Menghapus data di dalam query

contoh :

```
$this->db->delete("mahasiswa",array('nim'->$nim));
```

// Sama dengan : delete mahasiswa where nim="\$nim"



# BUILD BLOG WITH CI

## A. Rancangan dan Study kasus

Disini Kita akan membuat Webblog dengan assumasi

- ➔ Berita
- ➔ Kategori
- ➔ Komentar
- ➔ User

Dengan mengacu 4 hal diatas dapat dibuat 4 table

Berita

Id	Judul	Isi	Tanggal	Penulis	Status	Id_kategori
----	-------	-----	---------	---------	--------	-------------

Kategori

Id	Nama_kategori
----	---------------

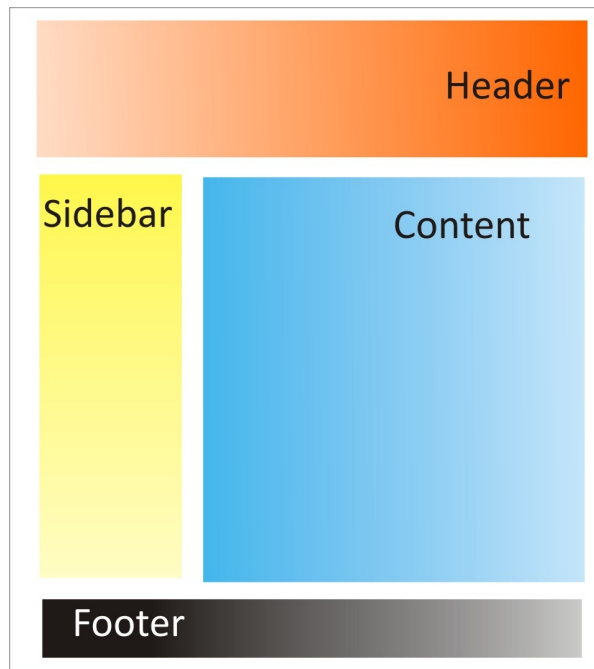
Komentar

id	Id_berita	Nama	Email	Website	Tanggal	Isi_komentar
----	-----------	------	-------	---------	---------	--------------

User

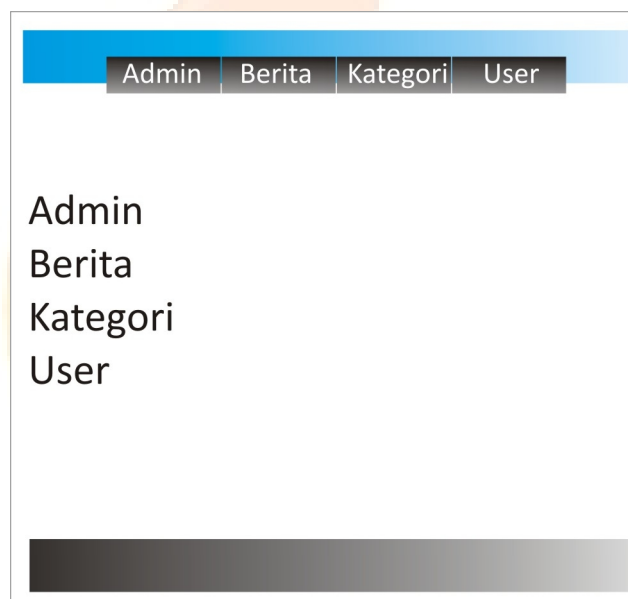
Id	Nama_lengkap	Username	Password	status
----	--------------	----------	----------	--------

Tampilan Home



Gambar 2.1

Tampilan adminpanel



Gambar 2.2

## B. Apa yang perlu disiapkan ?

Download codeiniter versi terbaru di <http://codeigniter.com>

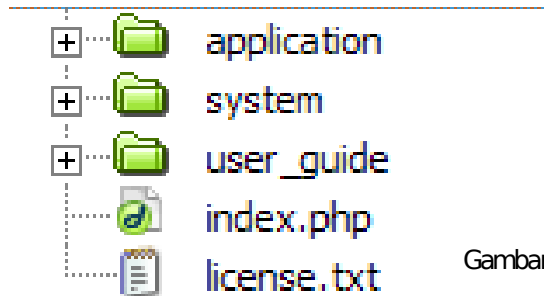
Server localhost (terserah anda memakai apa tapi disini aku pake xampp 1.7.0)

Editor (bisa menggunakan notepad, Dreamweaver, Netbeans atau yang lain)

Extrak file zip codeiniter di folder root C:\xampp\htdocs

Rename folder codeiniter dengan nama blog atau yang lain terserah

Dan pindahkan folder application yang berada di dalam system menjadi diluar system



Gambar 2.3

Sebenarnya folder application tidak di pindah juga tidak apa-apa hanya disini sesuai dengan keinginan programmer (saya ☺)

Buat database di <http://localhost/phpmyadmin> dengan nama blogdb

### C. Setingan awal

config.php - (application/config/config.php )

Ganti pada:

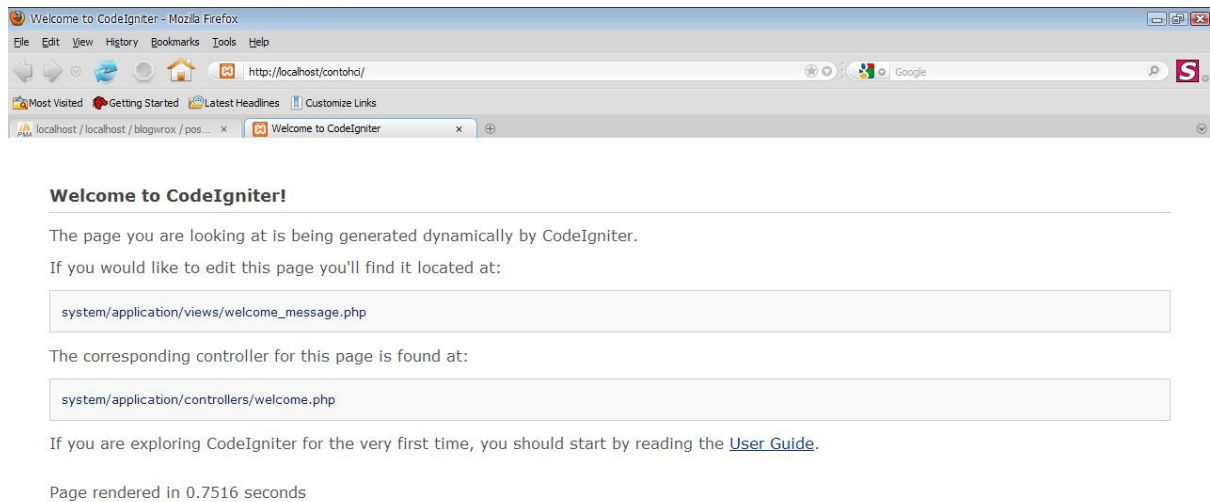
```
$config['base_url'] = "http://example.com/";
```

menjadi

```
$config['base_url'] = "http://localhost/blog/";
```

Sesui dengan nama folder

Buka browser anda Localhost/blog



Gambar 2.4

Sebenarnya tutorial lengkap ada di user guide silahkan masuk ke [http://localhost/blog/user\\_guide](http://localhost/blog/user_guide)

Lanjut konfigurasinya

Autoload.php –(application/config/autoload.php)

```
$autoload['libraries'] = array('database','session');
```

```
$autoload['helper'] = array('url','form','text','html');
```

database.php –(application/config/database.php)

```
$db['default']['hostname'] = "localhost";
```

```
$db['default']['username'] = "root";
```

```
$db['default']['password'] = "";
```

```
$db['default']['database'] = "blogdb"; //yang telah dibuat tadi
```

```
$db['default']['dbdriver'] = "mysql";
```

routes.php –(application/config/routes.php)

```
$route['default_controller'] = "home"; //default controller
```

Persiapan controller

## D. Halaman Depan

Buat file php di controller dengan nama home.php (merupakan controller default yang telah kita seting di routes)

```
<?php

class Home extends Controller{

    function Home(){
        parent::Controller();
    }

    function index(){
        $data['title']='contoh codeiniter'; //$title di view
        $data['content']='content'; //akan kita buat file content.php di view
        $data['berita']='ini adalah berita pertama'; //$berita di view
        $data['footer']='poedjex.wordpress.com'; //$footer di view
        $this->load->view('template',$data); //load ke view template.php dengan beban $data
    }
}

?>
```

### Persiapan View

Buat file php dengan nama template.php

```
<html>

<head>

    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">

    <title><?php echo $title;?></title>

</head>

<body>

    <?php $this->load->view($content); ?>

    <div id="footer"><?php echo $footer; ?></div>

</body>

</html>
```



Buat juga file content.php di view tersebut

```
<?php
echo '<h3>'.$berita.'</h3>';

?>
```

Dicek <http://localhost/blog>

---

**ini adalah berita pertama**

[poedjex.wordpress.com](http://poedjex.wordpress.com)

**Buatlah table di database bernama berita**

```
CREATE TABLE `berita` (
  `id` INT( 5 ) NOT NULL AUTO_INCREMENT PRIMARY KEY ,
  `judul` VARCHAR( 255 ) NOT NULL ,
  `isi` TEXT NOT NULL ,
  `tanggal` DATE NOT NULL ,
  `penulis` INT( 5 ) NOT NULL ,
  `status` INT( 5 ) NOT NULL DEFAULT '1',
  `id_kategori` INT( 5 ) NOT NULL DEFAULT '1'
) ENGINE = MYISAM ;

INSERT INTO `berita` (`id`, `judul`, `isi`, `tanggal`, `penulis`, `status`, `id_kategori`) VALUES
(1, 'Berita Pertama', ' Apa itu codeigniter(CI) ?
```

CodeIgniter adalah sebuah Application Development Framework - a toolkit - bagi orang-orang yang membangun situs web menggunakan PHP. Tujuannya adalah untuk memungkinkan untuk mengembangkan proyek-proyek yang jauh lebih cepat daripada menulis kode dari awal atau sering menyebutkan PHP murni , dengan menyediakan library-library yang dibutuhkan untuk tugas-tugas, serta antarmuka yang sederhana dan struktur logika untuk mengakses libraries tersebut. CodeIgniter meminimalkan jumlah sintak untuk tugas tertentu.

```
, '2010-07-09', 1, 1, 1);
```

**Buatlah file mberita.php di application/model**

```
<?php

class Mberita extends Model

{

    function Mberita()

    {

        parent::Model();

    }

    function getberita(){

        $this->db->order_by('id','desc');

        $this->db->where('status',1);
```

```

        $q=$this->db->get('berita');

        return $q;

    }

    function selengkapnya($id){

        $this->db->where('id',$id);

        $this->db->where('status',1);

        $q=$this->db->get('berita');

        return $q;

    }

}

?>

```

## Ubah controller home.php menjadi

```

<?php

class Home extends Controller{

    function Home(){

        parent::Controller();

        $this->load->model('mberita');

    }

    function index(){

        $data['title']='contoh codeiniter'; //$title di view

        $data['content']='content'; //akan kita buat file content.php di view

        $data['berita']=$this->mberita->getberita();

        $data['footer']='poedjex.wordpress.com'; //$footer di view

        $this->load->view('template',$data); //load ke view template.php dengan beban $data

    }

    function selengkapnya(){

        $id=$this->uri->segment(3);

        $data['title']='contoh codeiniter';

        $data['content']='selengkapnya';

        $data['berita']=$this->mberita->selengkapnya($id);

        $data['footer']='poedjex.wordpress.com';

        $this->load->view('template',$data);

    }

}

```

```
}
?>
```

## Ubah view content.php menjadi

```
<div class="content">

<?php foreach($berita->result() as $row):?>

    <div id="berita">

        <h2><?php echo $row->judul?></h2>

        <p><?php echo word_limiter($row->isi,20);?></p> <!--membatasi 20 kata -->

        <p><?php echo anchor('home/selengkapnya/'.$row->id, 'selengkapnya');?></p>

    <hr>

    </div>

<?php endforeach;?>

</div>
```

## Buatlah view bernama selengkapnya.php

```
<div class="content">

<?php foreach($berita->result() as $row):?>

    <div id="berita">

        <h2><?php echo $row->judul?></h2>

        <p><?php echo $row->isi;?></p>

        <p><?php echo anchor('home','Kembali');?></p>

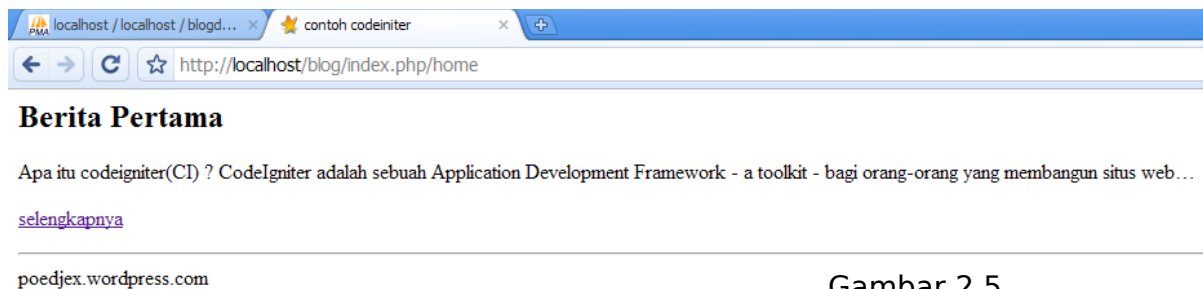
        <hr>

    </div>

<?php endforeach;?>

</div>
```

## Coba di cek localhost/home



Gambar 2.5

Dalam sekenario ketika link selengkapnya di klick maka akan tampil seluruh postingan dan ada bagian komentar di bawahnya

## Untuk komentar kita siapkan table komentar terlebih dahulu

```
CREATE TABLE `komentar` (
  `id` INT( 5 ) NOT NULL AUTO_INCREMENT PRIMARY KEY ,
  `id_berita` INT( 5 ) NOT NULL ,
  `nama` VARCHAR( 255 ) NOT NULL ,
  `email` VARCHAR( 255 ) NOT NULL ,
  `website` VARCHAR( 255 ) NOT NULL ,
  `tanggal` DATE NOT NULL ,
  `isi_komentar` TEXT NOT NULL
) ENGINE = MYISAM ;
```

## Buatlah model bernama mkomentar.php

```
<?php
```

```
class Mkomentar extends Model
```

```
{

    function Mkomentar()

    {

        parent::Model();

    }

    function addkomentar($id_berita,$nama,$email,$website,$koment)

    {

        $data=array(

            'id_berita'=>$id_berita,

            'nama'=>$nama,

            'email'=>$email,

            'website'=>$website,

            'isi_komentar'=>$koment,

            'tanggal'=>date('Y-m-d')

        );

        $this->db->insert('komentar',$data);

    }

    function listkomentar($id){

        $this->db->where('id_berita',$id);

        $this->db->order_by('id','desc');

        $q=$this->db->get('komentar');

        return $q;

    }

}
```

```
}
```

```
?>
```

## Edit controller home.php menjadi

```
<?php
```

```
class Home extends Controller{
```

```
function Home(){
```

```
    parent::Controller();
```

```
        $this->load->model(array('mberita','mkomentar'));
```

```
        $this->load->scaffolding('berita');
```

```
}
```

```
function index(){
```

```
    $data['title']='contoh codeiniter'; //$title di view
```

```
    $data['content']='content'; //akan kita buat file content.php di view
```

```
    $data['berita']=$this->mberita->getberita();
```

```
    $data['footer']='poedjex.wordpress.com'; //$footer di view
```

```
    $this->load->view('template',$data); //load ke view template.php dengan beban $data
```

```
}
```

```
function selengkapnya(){
```

```
    $id=$this->uri->segment(3);
```

```
    $data['title']='contoh codeiniter';
```

```
    $data['content']='selengkapnya';
```

```
    $data['berita']=$this->mberita->selengkapnya($id);
```

```
    $data['komentar']=$this->mkomentar->liskoment($id);
```

```
    $data['footer']='poedjex.wordpress.com';
```

```
    $this->load->view('template',$data);
```

```
}
```

```
function addkomentar(){
```

```
    $id_berita=$this->input->post('id',TRUE);
```

```
    $nama=$this->input->post('nama',TRUE);
```

```
    $email=$this->input->post('email',TRUE);
```

```
    $website=$this->input->post('website',TRUE);
```

```

        $komentar=$this->input->post('komentar',TRUE);

        $this->mkomentar->addkomentar($id_berita,$nama,$email,$website,$komentar);

        redirect('home/selengkapnya/'.$id_berita,'refresh');

    }

}

?>

```

## Ubah view selengkapnya.php

```

<div class="content">

<?php foreach($berita->result() as $row):?>

    <div id="berita">

        <h2><?php echo $row->judul?></h2>

        <p><?php echo $row->isi;?></p>

        <p><?php echo anchor('home','Kembali');?></p>

        <hr>

    </div>

<?php endforeach;?>

    <div id="komentar">

        <div id="listkomentar">

            <ul>

                <?php foreach($komentar->result() as $kmt):?>

                    <li>

                        <h4><?php echo $kmt->nama; ?></h4>

                        <p><?php echo $kmt->isi_komentar;?></p>

                        <hr>

                    </li>

                <?php endforeach;?>

            </ul>

        </div>

        <?php echo form_open('home/addkomentar');?>

        <?php echo form_hidden('id',$row->id);?>

        <table width="200" border="0" cellspacing="2" cellpadding="1">

            <tr>

                <td>Nama</td><td>&nbsp;</td><td><input type="text" name="nama" size="50"/></td>

            </tr>

```

```

</tr>

<tr>

    <td>Email</td><td>&nbsp;</td><td><input type="text" name="email" size="50"/></td>

</tr>

<tr>

    <td>Website</td><td>&nbsp;</td><td><input type="text" name="website" size="50"/></td>

</tr>

<tr>

    <td>&nbsp;</td><td>&nbsp;</td><td><textarea name="komentar" cols="40" rows="10"></textarea></td>

</tr>

<tr>

    <td>&nbsp;</td><td>&nbsp;</td><td><input type="submit" value="Submit" /></td>

</tr>

</table>

    <?php echo form_close();?>

</div>

</div>

```

Agar tampilan lebih menarik Kita buat folder template/css di dekat folder system (masih di dalam folder blog)

Buatlah file style.css

body{	color:#0099FF;	background:#FF6600;
background:#FFFFFF;	}	color:#FFFFFF;
}	.main,.header,.footer{	}
a {	width:960px;	.header #nav{
color:#0000FF;	margin:0 auto;	padding:35px 0 20px;
text-decoration:none;	}	margin-left:auto;
}	.header{	}
a:hover{	height:100px;	.header #title{

```

position:absolute;
margin-left:400px;
font-size:16px;
}
#nav ul{
list-style:none;
}
#nav ul li{
margin:0px 2px;
float:left;
}
#nav ul li a{
padding:10px;
background:#000000;
color:#FFFFFF;
text-decoration:none;
}
#nav ul li a:hover{
background:#00FF00;
}

}

.main .footer{
color:#FFFFFF;
background:#000000;
height:50px;
}
.main .content_resize{
width:960px;
height:auto;
background:#CCCCCC;
}
.content_resize .right{
background:#D6FEFD;
width:75%;
height:auto;
margin-left:auto;
}
.content_resize .sidebar{
width:20%;
float:left;
background:#FFFF00;
height:auto;
background:#CCCC00;
}
.sidebar #widget{
margin-left:10px;
padding:3px;
width:90%;
background:#CCCC00;
}
.content{
padding:3px;
}
#berita{
padding:10px;
border:1px solid #000;
margin-top:10px;
}

```

Rubah file template.php menjadi

```

<html>

<head>

<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">

<title><?php echo $title;?></title>

<link rel="stylesheet" href="<?php echo base_url();?>template/css/style.css" type="text/css"/>

</head>

<body>

<div class="main">

<div class="header"><br>

<div id="title"><h3><?php echo $title;?></h3></div>

<div id="nav">

```



```

<ul>

<li><?php echo anchor(base_url(),'Beranda');?></li>

</ul>

</div>

</div>

<div class="content_resize">

<div class="sidebar">

<div id="widget">

<h2>Kategori</h2>

</div>

</div>

<div class="right">

<?php $this->load->view($content); ?>

</div>

</div>

<div class="footer"><?php echo $footer; ?></div>

</div>

</body>

</html>

```

Maka Tampilan sudah berubah menjadi



Gambar 2.6

Sekarang pembuatan Link kategori

Terlebih dahulu buat table bernama kategori

```
CREATE TABLE `kategori` (
  `id` int(11) NOT NULL auto_increment,

  `nama_kategori` varchar(255) NOT NULL,

  PRIMARY KEY (`id`)

) ENGINE=MyISAM DEFAULT CHARSET=latin1 AUTO_INCREMENT=2 ;

INSERT INTO `KATEGORI` (`id`, `nama_kategori`) VALUES
(1, 'Uncategory');
```

Buat model bernama mkategori.php

```
<?php

class Mkategori extends Model{

    function Mkategori()

    {

        parent::Model();

    }

    function getkategori(){

        return $this->db->get('kategori');

    }

}

?>
```

Tambahkan 'mkategori' di controller home.php

```
$this->load->model(array('mberita','mkomentar','mkategori'));
```

Karena kategori di akses disetiap fungsi maka untuk mempermudah kita taruh di view template.php caranya

```
<div id="widget">

    <h2>Kategori</h2>

    <?php $kategori=$this->mkategori->getkategori();?>

    <ul>

        <?php foreach($kategori->result() as $kat){?>

            <li><?php echo anchor('home/perkategori/'.$kat->id,$kat->nama_kategori);?></li>

        <?php }?>

    </ul>
```

</div>

Dan tambahkan function `getberitakategori($id)` untuk mengelinkkan kategori ke berita

```
function getberitakategori($id){
    $this->db->where('id_kategori',$id);
    $this->db->order_by('id','desc');
    $q=$this->db->get('berita');
    return $q;
}
```



Gambar 2.7

## E. Halaman Admin

Untuk membuat halaman admin kita harus berfikir ganda tentang keamanannya.

Dalam halaman admin ini kita Sudah menggunakan DML (Data Manipulation Language) yang kita butuhkan hanya menambah , delete dan update

Oke kita mulai. Hal pertama yang perlu kita siapkan adalah tabel dalam database kita. Beri nama user

```
CREATE TABLE `user` (
  `id` int(11) NOT NULL auto_increment,
  `nama_lengkap` varchar(255) NOT NULL,
  `username` varchar(255) NOT NULL,
  `password` varchar(255) NOT NULL,
  `status` int(11) NOT NULL default '0',
```

PRIMARY KEY (`id`)

) ENGINE=MyISAM DEFAULT CHARSET=latin1 AUTO\_INCREMENT=2 ;

INSERT INTO `user` (`id`, `nama\_lengkap`, `username`, `password`, `status`) VALUES

(1, 'admin', 'admin', '21232f297a57a5a743894a0e4a801fc3', 1);

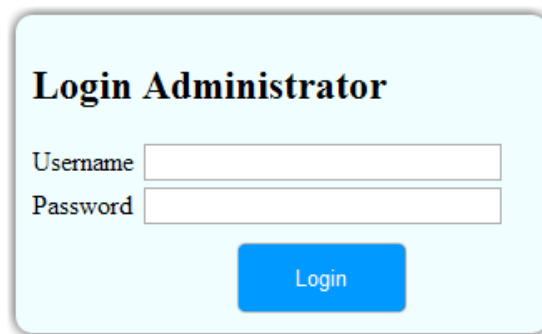
//user : admin

//password: admin

Untuk tampilan kita butuh 2 tampilan

1. Halaman login fungsinya memasukan username dan password dengan form
2. Halaman dashboard fungsinya mengatur halaman web dari mulai berita , kategori

## Membuat halaman login



Pertama kita buat view dengan nama login.php

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
```

```
<html xmlns="http://www.w3.org/1999/xhtml">
```

```
<head>
```

```
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
```

```
<title>Login Administrator</title>
```

```
<style type="text/css">
```

```
body{
```

```
background:#FFFFFF;
```

```
}
```

```
.box{
```

```
background:#F0FEFF;

width:300px;

height:170px;

padding:10px;

margin:auto;

margin-top:100px;

-moz-border-radius:10px;

-webkit-border-radius:10px;

border-radius:10px;

-moz-box-shadow:0 0 10px #333;

-webkit-box-shadow:0 0 10px #333;

-box-shadow:0 0 10px #333;

}

input[type=submit]{

background:#0099FF;

color:#FFFFFF;

width:100px;

height:40px;

border:none;

margin-top:10px;

-moz-border-radius:4px;

-webkit-border-radius:4px;

border-radius:4px;

-moz-box-shadow:0 0 2px #333;

-webkit-box-shadow:0 0 2px #333;

-box-shadow:0 0 2px #333;

}

input[type=submit]:hover{

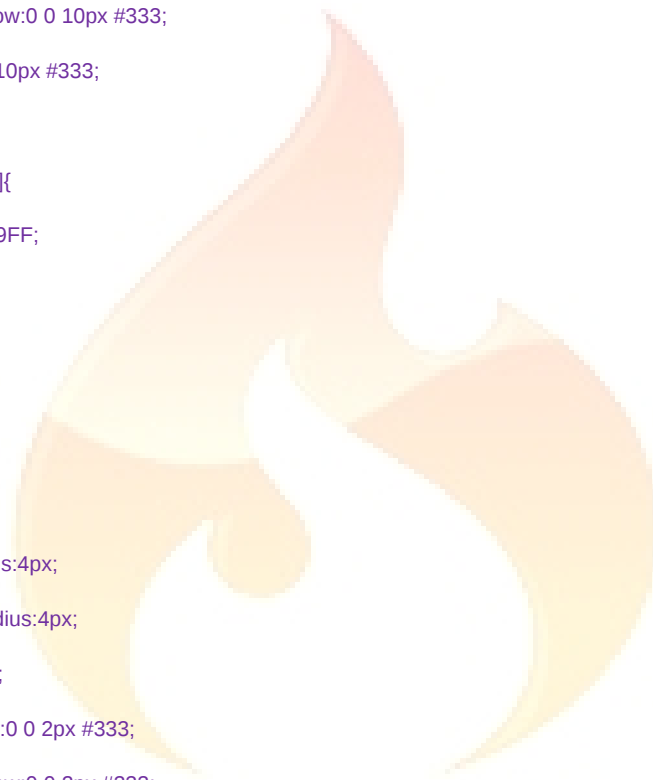
background:#0000FF;

}

</style>

</head>

<body>
```



```
<div class="box">

<h2>Login Administrator</h2>

<?php echo form_open('administrator/login');?>

<table width="200" border="0" cellspacing="0" cellpadding="0">

<tr>

<td>Username</td><td>&nbsp;</td><td><input name="username" type="text" size="30"></td>

</tr>

<tr>

<td>Password</td><td>&nbsp;</td><td><input name="password" type="password" size="30"></td>

</tr>

<tr>

<th></th>

<th>&nbsp;</th>

<th><input type="submit" value="Login"></th>

</tr>

</table>

<?php echo form_close();?>

</div>

</body>

</html>
```

kita buat controller baru bernama administrator.php supaya seperti CMS joomla dengan maka kita hilangkan index.php

sehingga seharusnya <http://localhost/blog/index.php/administrator> menjadi <http://localhost/blog /administrator>

tanpa index.php

kita buat terlebih dahulu file bernama .htaccess di dekat index.php (masih di dalam folder blog)

isikan :

```
RewriteEngine On

RewriteCond $1 !^(index\.php|image|css|public|ajax|tmp|download|javascript|rte|document|
xinha|robots\.txt)
RewriteCond %{REQUEST_FILENAME} !-f
RewriteCond %{REQUEST_FILENAME} !-d
RewriteRule ^(.*)$ index.php/$1 [L]
```

kembali ke controller administrator

kita isikan

```
<?php

class Administrator extends Controller
{
    function Administrator()
    {
        parent::Controller();

        $this->load->model('muser');
    }

    function index(){
        if($this->session->userdata('logged_in'))
            redirect('admin/dashboard','refresh');

        $this->load->view('login');
    }

    function login(){
        $username = $this->input->post('username',TRUE);
        $password = $this->input->post('password',TRUE);

        $this->db->where('username', $username);
        $this->db->where('password', md5($password));

        $query = $this->db->get('user');

        if ($query->num_rows() == 1) {
            foreach($query->result() as $row) {

                $nama = $row->nama_lengkap;

                $id_user=$row->id;
            }
        }

        $user= $this->muser->cek_user($username,$password);

        if($user == TRUE){
            $data = array( 'nama_lengkap' => $nama,
                'username'=> $username,
```

```
'id_user'=>$id_user,

'logged_in' => TRUE );

$this->session->set_userdata($data);

redirect('admin/dashboard','refresh');

}else{

$this->session->set_flashdata('nama',$username);

$this->session->set_flashdata('login_message', '<divclass="error">Username atau Password Anda Tidak
Sesuai</div>');

redirect('administrator');

}

}

function logout(){

$this->session->unset_userdata('nama_lengkap');

$this->session->unset_userdata('username');

$this->session->unset_userdata('id_user');

$this->session->unset_userdata('logged_in');

redirect(site_url()); // sesudah logout di redirect ke halaman utama

}

}

?>
```

Untuk pengecekan dengan table user maka kita buat juga model bernama muser.php

```
<?php

class Muser extends Model {

function Muser() {

parent::Model();

}

function cek_user($username, $password) {

$this->db->where('username',$username);

$this->db->where('password',md5($password));

$query = $this->db->get('user');

if($query->num_rows() > 0) {
```



```

return TRUE;

}

else {

    return FALSE;

}

}

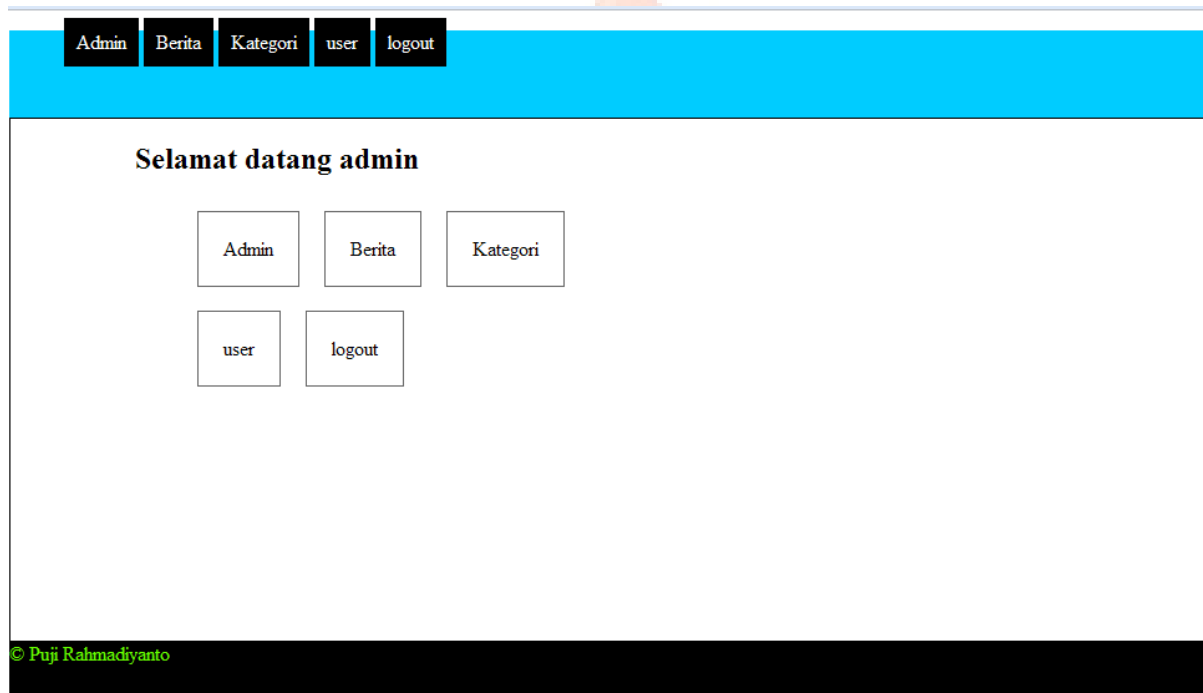
}

?>

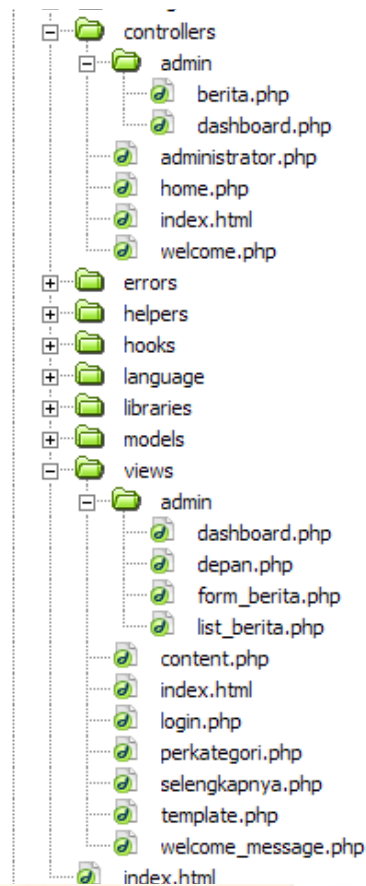
```

Terlihat bahwa jika user dan password dicocokkan akan memberikan nilai pengembali TRUE dan FALSE

## Halaman Dashboard



Dalam dashboard ini kita perlu pisahkan controller maupun view dengan folder admin misalkan



Buatlah controller dashboard.php di dalam controller/admin

```
<?php
class Dashboard extends Controller
{
    function Dashboard()
    {
        parent::Controller();

        if(!$this->session->userdata('logged_in') && !$this->session->userdata('username'))
            redirect(base_url(), 'refresh');
    }

    function index()
    {
        $data['content'] = 'admin/depan';

        $this->load->view('admin/dashboard', $data);
    }
}
?>
```

Buat folder admin dan buat file dashboard.php di dalam folder view/admin

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
```

```
<html xmlns="http://www.w3.org/1999/xhtml">
```

```
<head>
```

```
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
```

```
<title>Administrator</title>
```

```
<style type="text/css">
```

```
body{
```

```
background:#FFFFFF;
```

```
}
```

```
a{
```

```
padding:2px;
```

```
background:#FF99FF;
```

```
text-decoration:none;
```

```
}
```

```
a:hover{
```

```
background:#999933;
```

```
}
```

```
.main{
```

```
width:960px;
```

```
margin:auto;
```

```
}
```

```
.header{
```

```
width:100%;
```

```
height:70px;
```

```
background:#00CCFF;
```

```
}
```

```
.header ul{
```

```
list-style:none;
```

```
}
```

```
.header ul li {
```



```
float:left;

margin-left:4px;

}

.header ul li a {

color:#FFFFFF;

padding:10px;

background:#000000;

text-decoration:none;

}

.header ul li a:hover{

background:#0000FF;

}

.content{

width:100%;

background:#FFFFFF;

border:1px solid #000000;

}

.footer{

background:#000000;

color:#66FF00;

width:100%;

height:45px;

}

.depan{

width:400px;

margin-left:100px;

height:400px;

}

.depan ul{

list-style:none;

}

.depan ul li{
```



```
float:left;

margin-left:10px;

margin:10px;

margin-top:30px;

margin-bottom:30px;

}

.depan ul li a{

padding:20px;

border:1px solid #666666;

text-decoration:none;

color:#000000;

}

.depan ul li a:hover{

background:#CCCCCC;

}

td{

border:1px solid #000000;

}

</style>

</head>

<body>

<div class="main">

<div class="header">

<ul>

<li><?php echo anchor('administrator','Admin');?></li>

<li><?php echo anchor('admin/berita','Berita');?></li>

<li><?php echo anchor('admin/kategori','Kategori');?></li>

<li><?php echo anchor('administrator/logout','logout');?></li>

</ul>
```



```
</div>

<div class="content">

<?php $this->load->view($content);?>

</div>

<div class="footer">

&copy; Puji Rahmadiyanto

</div>

</div>

</body>

</html>
```

Dan buat juga depan.php di dalam folder admin tadi

```
<div class="depan">

<h2>Selamat datang <?php echo $this->session->userdata('nama_lengkap');?></h2>

<ul>

<li><?php echo anchor('administrator','Admin');?></li>

<li><?php echo anchor('admin/berita','Berita');?></li>

<li><?php echo anchor('admin/kategori','Kategori');?></li>

<li><?php echo anchor('administrator/logout','logout');?></li>

</ul>

</div>
```

Kita akan membuat halaman pengaturan berita dan kategori jadi yang di butuhkan di controller adalah controller berita dan kategori dilengkapi

Dengan fungsi-fungsi:

- ➔ Function index
- ➔ Function tambah()
- ➔ Function submit()
- ➔ Function edit()
- ➔ Function edit\_submit()
- ➔ Function delete()

Untuk viewnya

Berita :

- List\_berita.php
- Form\_berita.php
- Form\_edit\_berita.php

Kategori :

- List\_kategori.php
- Form\_kategori.php
- Form\_edit\_berita.php

Kita mulai dengan controller berita di admin/berita.php

<?php

```
class Berita extends Controller{

    function Berita(){

        parent::Controller();

        if(!$this->session->userdata('logged_in')&& !$this->session->userdata('username')) // tambahkan

            redirect(base_url(),'refresh');

    }

    function index(){

        $this->load->model(array('mberita','mkomentar','mkategori'));

        $data['berita']=$this->mberita->getallberita();

        $data['content']='admin/list_berita';

        $this->load->view('admin/dashboard',$data);

    }

    function tambah(){

        $this->load->model(array('mberita','mkomentar','mkategori'));

        $data['content']='admin/form_berita';

        $this->load->view('admin/dashboard',$data);

    }

    function submit(){

        $this->load->model(array('mberita','mkomentar','mkategori'));

        $judul=$this->input->post('judul',TRUE);

        $isi=$this->input->post('isi',TRUE);
```

```

        $status=$this->input->post('status',TRUE);

        $kategori=$this->input->post('kategori',TRUE);

        $this->mberita->tambahberita($judul,$isi,$status,$kategori);

        redirect('admin/berita');

    }

    function edit(){

        $id=$this->uri->segment(4);

        $this->load->model(array('mberita','mkomentar','mkategori'));

        $data['berita']=$this->mberita->ambilberita($id);

        $data['content']='admin/form_edit_berita';

        $this->load->view('admin/dashboard',$data);

    }

    function edit_submit(){

        $this->load->model(array('mberita','mkomentar','mkategori'));

        $id=$this->input->post('id',TRUE);

        $judul=$this->input->post('judul',TRUE);

        $isi=$this->input->post('isi',TRUE);

        $status=$this->input->post('status',TRUE);

        $kategori=$this->input->post('kategori',TRUE);

        $this->mberita->updateberita($id,$judul,$isi,$status,$kategori);

        redirect('admin/berita');

    }

    function delete(){

        $id=$this->uri->segment(4);

        $this->load->model(array('mberita','mkomentar','mkategori'));

        $this->mberita->deleteberita($id);

        redirect('admin/berita');

    }

}

?>

```



## View

### Admin/list\_berita.php

```
<br>

<table width="700" border="0" align="center">

<tr>

<td>No</td>

<td>Berita</td>

<td>Aksi </td>

<td>Status</td>

</tr>

<?php $i=1;?>

<?php foreach($berita->result() as $row):?>

<tr>

<td><?php echo $i;?></td>

<td><?php echo "<b>$row->judul</b><br/><p>".word_limiter($row->isi,10)."</p>"; ?></td>

<td><?php echo anchor('admin/berita/edit/'.$row->id,'edit').'|'.anchor('admin/berita/delete/'.$row->id,'delete',array ('onClick'
=> "return confirm('apakah anda yakin '));?></td>

<td><?php if($row->status==1){echo "Aktif";}else{echo "Nonaktif";}?></td>

</tr>

<?php $i++;?>

<?php endforeach;?>

</table>

<?php echo anchor('admin/berita/tambah','tambah berita');?>
```



## Admin/form\_berita.php

```
<?php echo form_open('admin/berita/submit');?>

<table width="200" border="0" cellspacing="2" cellpadding="1" align="center">

<tr>

<td>Judul<br><input name="judul" type="text" size="50"/></td>

</tr>

<tr>

<td>Isi Berita<br><textarea name="isi" cols="50" rows="20"></textarea></td>

</tr>

<tr>

<td><select name="kategori">

<?php
$kat=$this->mkategori->getkategori();
foreach($kat->result() as $kategori):?>

<option value="<?php echo $kategori->id;?>" ><?php echo $kategori->nama_kategori?></option>

<?php endforeach;?>

</select>

</td>

</tr>

<tr>

<td><select name="status">

<option value="1" selected="selected">Aktif</option>

<option value="0" >Nonaktif</option>

</select>

</td>

</tr>

<tr>

<td><input type="submit" value="Post">|<input type="reset" value="Reset"></td>

</tr>

</table>
```

```
<?php echo form_close();?>
```

http://localhost/blog/index.php/admin/berita/tambah

[Admin](#) [Berita](#) [Kategori](#) [logout](#)

Judul

Isi Berita

Uncategory

Aktif

Post

Reset

© Puji Rahmadiyanto

**Admin/form\_edit\_berita.php**

```
<?php foreach($berita->result() as $brt){

    $judul=$brt->judul;

    $isi=$brt->isi;

    $kategori_id=$brt->id_kategori;

    $status=$brt->status;

    $id=$brt->id;

}??>

<?php echo form_open('admin/berita/edit_submit');?>

<table width="200" border="0" cellspacing="2" cellpadding="1" align="center">

<tr>

    <td>Judul<br><input name="judul" type="text" size="50" value='<?php echo $judul;?>' /></td>

</tr>

<tr>

    <td>Isi Berita<br><textarea name="isi" cols="50" rows="20"><?php echo $isi;?></textarea></td>
```

```

</tr>

<tr>

<td><select name="kategori">

<?php
$kat=$this->mkategori->getkategori();
foreach($kat->result() as $kategori):
if($kategori_id==$kategori->id){
$select="selected='elected'";
}
else {
$select="";
}
?>
<option value="<?php echo $kategori->id;?>" <?php echo $select;?><?php echo $kategori-
>nama_kategori?></option>
<?php endforeach;?>
</select>
</td>
</tr>
<tr>
<td><select name="status">
<?php if ($status==1){?>
<option value="1" selected="selected">Aktif</option>
<option value="0" >Nonaktif</option>
<?php }else{?>
<option value="1" >Aktif</option>
<option value="0" selected="selected">Nonaktif</option>
<?php }?>
</select>
</td>
</tr>
<?php echo form_hidden('id',$id);?>

```

```
<tr>
```

```
<td><input type="submit" value="Update"></td>
```

```
</tr>
```

```
</table>
```

```
<?php echo form_close();?>
```

Pasti masih error karena function untuk untuk update, delete, insertnya masih belum ada maka kembali ke model mberita.php

Tambahkan function berikut

```
function getallberita(){
```

```
    $this->db->order_by('id','desc');
```

```
    $q=$this->db->get('berita');
```

```
    return $q;
```

```
}
```

```
function tambahberita($judul,$isi,$status,$kategori){
```

```
    $data=array('judul'=>$judul,
```

```
                'isi'=>$isi,
```

```
                'tanggal'=>date('Y-m-d'),
```

```
                'penulis'=>$this->session->userdata('id_user'),
```

```

        'status'=>$status,

        'id_kategori'=>$kategori

    );

    $this->db->insert('berita',$data);

}

function ambilberita($id){

    $this->db->where('id',$id);

    $q=$this->db->get('berita');

    return $q;

}

function updateberita($id,$judul,$isi,$status,$kategori){

    $data=array('judul'=>$judul,

                'isi'=>$isi,

                'tanggal'=>date("Y-m-d"),

                'penulis'=>$this->session->userdata('id_user'),

                'status'=>$status,

                'id_kategori'=>$kategori

    );

    $this->db->where('id',$id);

    $this->db->update('berita',$data);

}

function deleteberita($id){

    $this->db->delete('berita',array('id'=>$id));

}

```

## Managemen kategori

Kita buat controller kategori.php di folder admin

```

<?php

class Kategori extends Controller{

    function Kategori(){

        parent::Controller();

        if(!$this->session->userdata('logged_in')&& !$this->session->userdata('username')) // tambahkan

            redirect(base_url(),'refresh');

    }

}

```

```
function index(){

    $this->load->model(array('mberita','mkomentar','mkategori'));

    $data['content']='admin/list_kategori';

    $data['kategori']=$this->mkategori->getkategori();

    $this->load->view('admin/dashboard',$data);

}

function tambah(){

    $data['content']='admin/form_kategori';

    $this->load->view('admin/dashboard',$data);

}

function submit(){

    $this->load->model(array('mberita','mkomentar','mkategori'));

    $kategori=$this->input->post('kategori',TRUE);

    $this->mkategori->tambahkategori($kategori);

    redirect('admin/kategori');

}

function edit(){

    $id=$this->uri->segment(4);

    $this->load->model(array('mberita','mkomentar','mkategori'));

    $data['kategori']=$this->mkategori->ambilkategori($id);

    $data['content']='admin/form_edit_kategori';

    $this->load->view('admin/dashboard',$data);

}

function edit_submit(){

    $this->load->model(array('mberita','mkomentar','mkategori'));

    $id=$this->input->post('id',TRUE);

    $kategori=$this->input->post('kategori',TRUE);

    $this->mkategori->updatekategori($id,$kategori);

    redirect('admin/kategori');

}

function delete(){
```

```

        $id=$this->uri->segment(4);

        $this->load->model(array('mberita','mkomentar','mkategori'));

        $this->mkategori->deletekategori($id);

        redirect('admin/kategori');

    }

}

?>

```

## View

### Admin/list\_kategori.php

```

<?php echo anchor('admin/kategori/tambah','tambah kategori');?><br>

<?php $i=1;?>

<table width="200" border="0" cellpadding="1" cellspacing="2" align="center">

    <tr>

        <td>No</td>

        <td>Kategori</td>

        <td>Aksi</td>

    </tr>

    <?php foreach($kategori->result() as $kat):?>

    <tr>

        <td><?php echo $i;?></td>

        <td><?php echo $kat->nama_kategori;?></td>

        <td><?php echo anchor('admin/kategori/edit/'.$kat->id,'edit').'|'.anchor('admin/kategori/delete/'.$kat->id,'delete',array('onClick' => 'return confirm('apakah anda yakin ')'));?></td>

    </tr>

    <?php $i++; endforeach;?>

</table>

```





## Admin/form\_kategori.php

```
<?php echo form_open('admin/kategori/submit')?>

<table width="200" border="0" cellpadding="2" cellspacing="1" align="center">

    <tr>

        <td><input name="kategori" size="40" type="text" /></td>

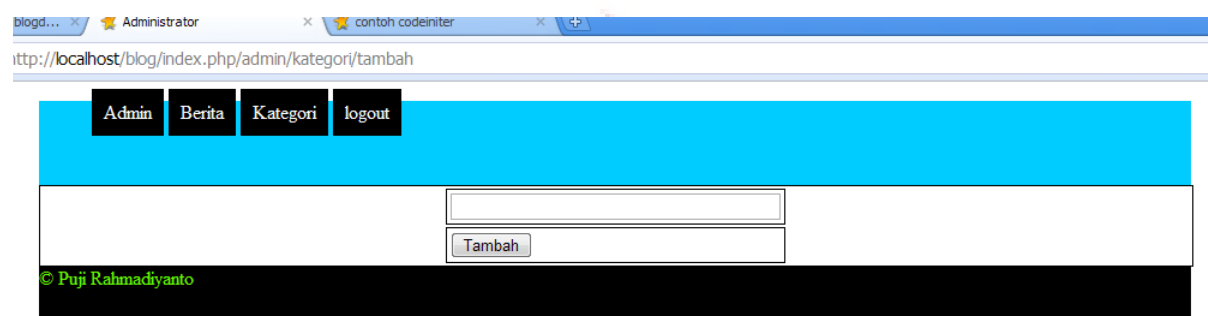
    </tr>

    <tr><td><input type="submit" value="Tambah"/></td>

    </tr>

</table>

<?php echo form_close();?>
```



## Admin/form\_edit\_kategori.php

```
<?php foreach($kategori->result() as $kat){

    $judul=$kat->nama_kategori;

    $id=$kat->id;

}>

<?php echo form_open('admin/kategori/edit_submit')?>

<table width="200" border="0" cellpadding="2" cellspacing="1">

    <tr>

        <td><input name="kategori" size="40" type="text" value='<?php echo $judul;?>' /></td>

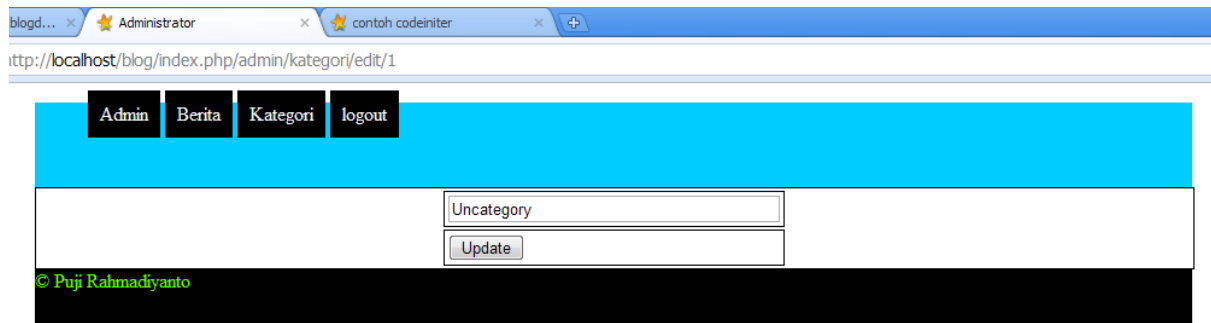
    </tr>

    <tr><td><input type="submit" value="Update"/></td>

    </tr>

</table>

<?php echo form_close();?>
```



Samahalnya dengan management berita diatas form-form tersebut akan error karena model mkategori belum ditambahkan function untuk insert ,update dan delete

Tambahkan function di model mkategori.php

```
function tambahkategori($kategori){
    $this->db->insert('kategori',array('nama_kategori'=>$kategori));
}

function ambilkategori($id){
    $this->db->where('id',$id);
    return $this->db->get('kategori');
}

function updatekategori($id,$kategori){
    $this->db->where('id',$id);
    $this->db->update('kategori',array('nama_kategori'=>$kategori));
}

function deletekategori($id){
    $this->db->delete('kategori',array('id'=>$id));
}
```

Sekarang Blog sederhana buatan anda sudah selesai.

Untuk masuk ke halaman admin username dan passwordnya "admin" tanpa tanda petik

Blog diatas masih banyak bug-bugnya jika ingin menambah ataupun merombak semua script silahkan karena disini saya juga masih belajar

Terima Kasih

## Daftar Pustaka

<http://ilmukomputer.com>

[http://codeigniter.com/user\\_guide](http://codeigniter.com/user_guide)



Web yang pernah dibuat

