

Tutorial Codeigniter Untuk Pemula

by M Fikri Setiadi

Saya mengerti kegelisahan Anda dalam memulai belajar Codeigniter.

Bingung, apa saja yang perlu diketahui, dan tidak tahu harus mulai dari mana.

Kabar baiknya,

Pada tutorial kali ini, saya akan sharing bagaimana memulai belajar codeigniter dari awal.

Secara step by step.

Jika Anda pemula, anda akan menyukai tutorial ini.

Let's dive right in.

1. Pengenalan Codeigniter

Codeigniter merupakan suatu Web Application Framework (WAF) yang dirancang khusus untuk mempermudah developer web dalam mengembangkan aplikasi berbasis web.

Codeigniter berisi kumpulan kode berupa pustaka (library) dan alat (tools) yang dipadukan sedemikian rupa menjadi suatu kerangka kerja (framework).

Codeigniter adalah framework web untuk bahasa pemrograman PHP yang dirancang oleh Rick Ellis pada tahun 2006, penemu dan pendiri EllisLab (www.ellislabs.com).

Ellislab adalah tim kerja yang berdiri pada tahun 2002 dan bergerak di bidang pembuatan software dan tool untuk para pengembang web.

Sejak tahun 2014 sampai sekarang, EllisLab telah menyerahkan hak kepemilikan codeigniter ke British Columbia Institute of Technology (BCIT) untuk proses pengembangan lebih lanjut.

Codeigniter memiliki banyak fitur (fasilitas) yang membantu para pengembang PHP untuk dapat membuat aplikasi web secara mudah dan cepat.

Codeigniter memiliki desain yang lebih sederhana dan bersifat fleksibel (tidak kaku).

Codeigniter mengizinkan para pengembang web untuk menggunakan framework secara parsial atau secara keseluruhan.

Ini berarti bahwa codeigniter masih memberikan kebebasan kepada pengembang untuk menulis bagian-bagian kode tertentu di dalam aplikasi menggunakan cara konvensional (tanpa framework).

Codeigniter menganut pola desain atau arsitektur **Model-View-Controller (MVC)** yang memisahkan bagian kode untuk penanganan proses bisnis dengan bagian kode untuk keperluan presentasi (tampilan).

Dengan menggunakan pola desain ini, memungkinkan para pengembangan web untuk mengerjakan aplikasi berbasis web secara bersama (*teamwork*).

Dengan begitu para pengembang web lebih berfokus pada bagiannya masing-masing tanpa mengganggu bagian yang lain.

Sehingga aplikasi yang dibangun akan selesai lebih cepat.

2. Keunggulan Codeigniter

Codeigniter merupakan sebuah toolkit yang ditujukan untuk Anda yang ingin membangun aplikasi berbasis web dalam bahasa pemrograman PHP.

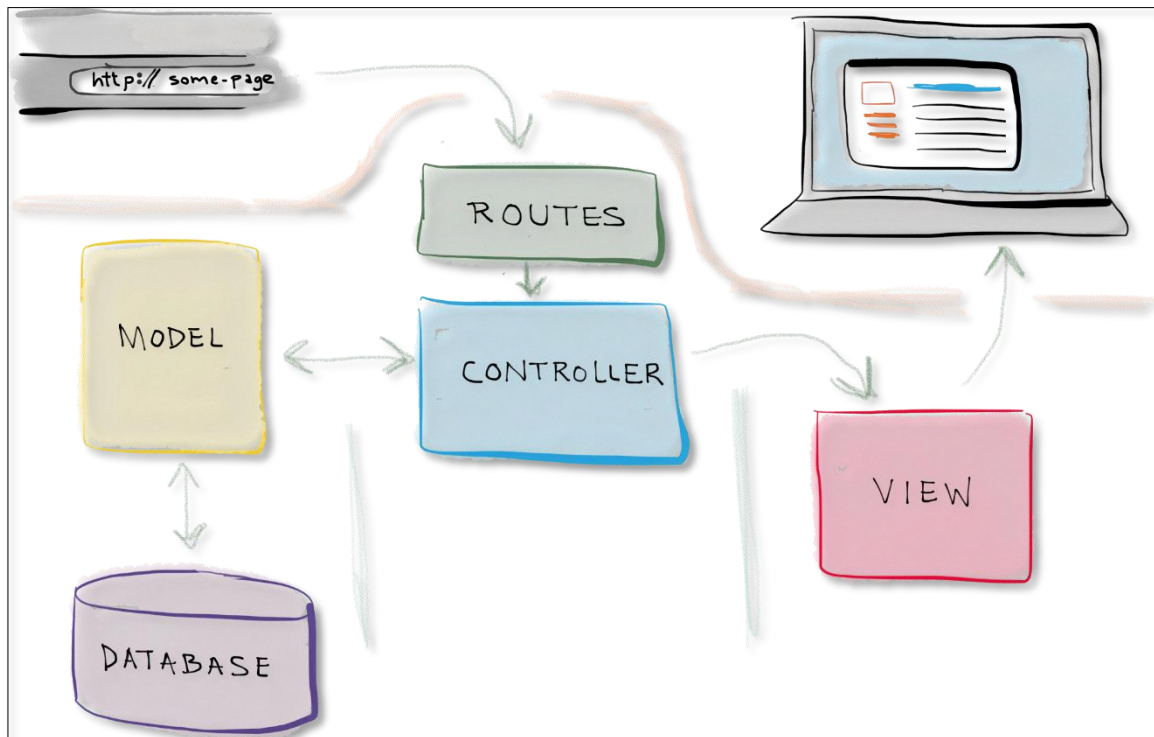
Adapun beberapa keunggulan yang ditawarkan oleh codeigniter adalah sebagai berikut:

1. Codeigniter adalah framework PHP yang bersifat open-source.
2. Codeigniter memiliki ukuran yang kecil dibandingkan dengan framework lain. Setelah proses instalasi, framework Codeigniter hanya berukuran kurang lebih 2 MB (tanpa dokumentasi atau jika user_guide dihapus).
3. Aplikasi yang dibuat menggunakan codeigniter bisa berjalan cepat.
4. Codeigniter menggunakan pola desain Model-View-Controller (MVC) sehingga satu file tidak terlalu berisi banyak kode. Hal ini menjadikan kode lebih mudah dibaca, dipahami, dan dipelihara di kemudian hari.
5. Codeigniter dapat diperluas sesuai dengan kebutuhan.
6. Codeigniter terdokumentasi dengan baik informasi tentang pustaka (*Library*) dan fungsi yang disediakan oleh codeigniter dapat diperoleh melalui dokumentasi yang disertakan di dalam paket distribusinya.
7. Codeigniter memiliki library dan helper yang lengkap.
8. Codeigniter memiliki security yang handal seperti xss filtering, session encryption, dan lain-lain.
9. Codeigniter mengizinkan pengembang web menggunakan library atau helper yang tidak disediakan oleh codeigniter seperti: Google Map API, Facebook API, fpdf, dan lain sebagainya.

10. Codeigniter bersifat tidak kaku. sehingga memberikan kebebasan kepada developer web untuk mengembangkan aplikasi berbasis web bahkan tanpa framework.
11. Codeigniter memiliki komunitas yang besar dan tersebar di seluruh dunia, sehingga memudahkan para pengembang web untuk memecahkan permasalahan (*problem solving*) yang dihadapi para pengembang web di saat mengembangkan aplikasi berbasis web.
12. Codeigniter mendukung banyak RDBMS (*Relational Database Management System*) seperti MySQL, MySQLi, SQL Server, Oracle, Maria DB, PostgreSQL, SQLite, dan lain sebagainya.
13. Codeigniter pada dasarnya menganut Clean URL dan mendukung SEO (*Search Engine Optimazation*).

Sehingga setiap aplikasi yang dibangun menggunakan codeigniter lebih mudah di index oleh search engine populer seperti google, yahoo, msn, dan lain sebagainya.

3. Model-View-Controller (MVC)



Seperti yang saya singgung sebelumnya bahwa codeigniter menganut arsitektur Model-View-Controller (MVC).

Jadi, sangatlah penting bagi Anda untuk mengetahui konsep dari desain MVC ini.

Apa itu Model-View-Controller (MVC)?

MVC adalah sebuah pendekatan yang ditempuh untuk memisahkan aplikasi menjadi tiga bagian, yaitu Model, View, dan Controller.

MVC memberikan struktur kepada aplikasi, sehingga dapat di capai "code reusability".

Berikut penjabaran dari komponen-komponen MVC:

1. Model

Model merepresentasikan data yang digunakan aplikasi, seperti database, RSS, atau data yang diperoleh dari pemanggilan API, dan aksi yang melibatkan operasi Create, Read, Update, dan Delete (CRUD) data.

2. View

View adalah informasi yang ditampilkan kepada user melalui browser. Biasanya berupa file HTML atau kode PHP yang menyusun template untuk sebuah website.

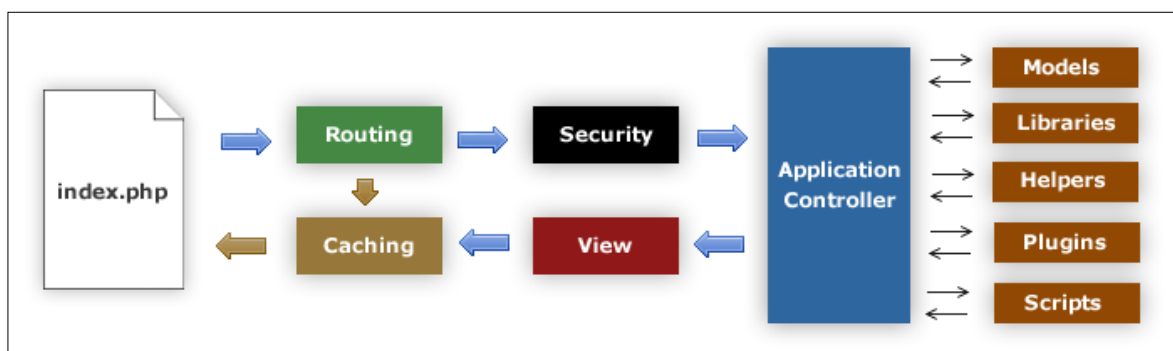
Pada codeigniter, view dapat berupa bagian-bagian sebuah halaman, template, atau jenis lain dari halaman atau template.

3. Controller

Controller adalah "business logic" yang bertugas sebagai jembatan antara model dan view.

Controller akan merespon HTTP request yang datang dari user (melalui browser), dari request ini controller akan menentukan apa yang harus dilakukan.

Didalam codeigniter, secara detail MVC digambarkan sebagai berikut:



Pada gambar diatas, file "index.php" berperan sebagai controller utama yang memanggil fungsi-fungsi dasar yang digunakan untuk menjalankan controller.

Router memeriksa HTTP request kemudian memutuskan Controller mana yang akan digunakan untuk menangani request tersebut.

Apabila file cache tersedia, alur aplikasi akan dilewati dan file cache tersebut yang akan dikirimkan ke browser pengguna.

Sebelum controller dipanggil, HTTP request dan data yang dikirimkan pengguna akan di sortir terlebih dahulu untuk alasan keamanan.

Controller memanggil file Model, Library, Helper, dan file lain yang dibutuhkan untuk menangani HTTP request.

Hasil akhirnya akan ditampilkan ole file View kemudian dikirimkan ke browser pengguna untuk ditampilkan.

Jika modus caching diaktifkan, hasil view akan di-cache terlebih dahulu.

Sehingga jika nanti ada request yang sama, bisa langsung digunakan.

4. Instalasi Codeigniter

Codeigniter merupakan framework PHP yang cara installnya paling sederhana dibandingkan framework PHP lainnya.

Anda hanya perlu mengextractnya ke web server Anda dan langsung terinstall.

Mudah bukan?

Jika Anda ingin menginstall codeigniter di server lokal atau localhost, ada beberapa software yang bisa Anda gunakan.

Diantaranya yang paling populer yaitu WAMPSERVER, MAMP, atau XAMPP.

Anda boleh pilih salah satunya.

Disini saya menggunakan WAMPSERVER, jika Anda juga menggunakan wampserver Anda akan menyukai artikel ini.

Tetapi, jika Anda menggunakan XAMPP.

No Problem,

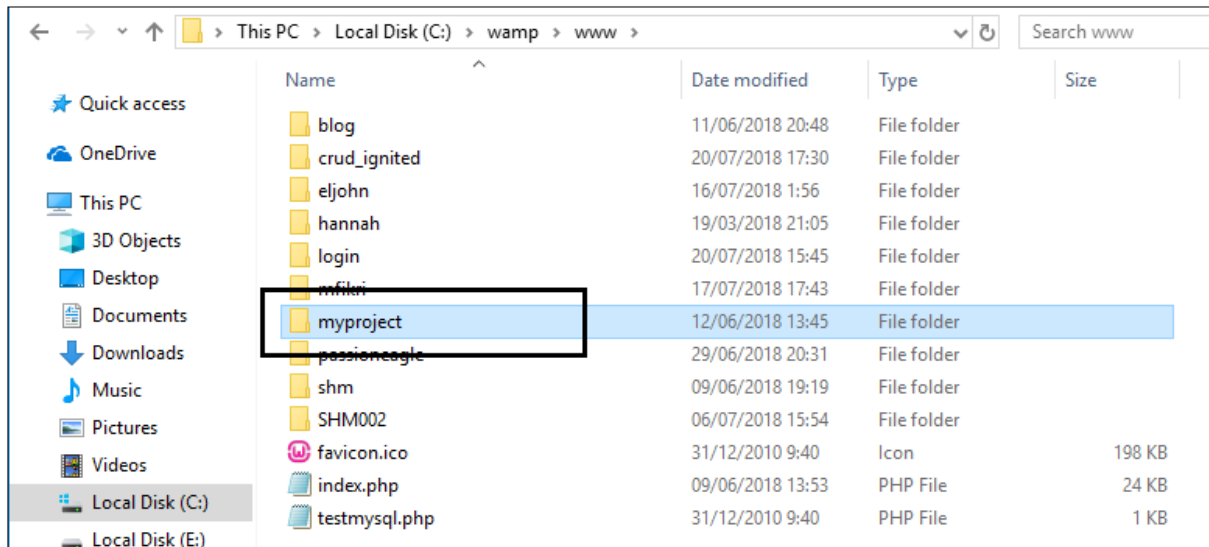
Karena saya akan menunjukan cara install pada WAMP dan XAMPP.

Ok, untuk melakukan instalasi pada codeigniter ikuti langkah berikut:

1. Pastikan Web Server telah terinstall dan berjalan (*running*) di komputer Anda.
2. Download file codeigniter di situs resminya: www.codeigniter.com
3. Extract file **Codeigniter.zip** ke direktori **C:/wamp/www/** (jika Anda menggunakan wampserver). Tetapi, jika Anda menggunakan XAMPP. Extract file **Codeigniter.zip** ke direktori **C:/xampp/htdocs/**.

4. Pergi ke folder `c:/wamp/www/` (jika Anda menggunakan WAMP) dan **rename** (ganti nama) file folder codeigniter yang baru di extraxt tadi menjadi nama project Anda.

Misalnya, disini saya ganti menjadi **"myproject"**. Sehingga terlihat seperti gambar berikut:

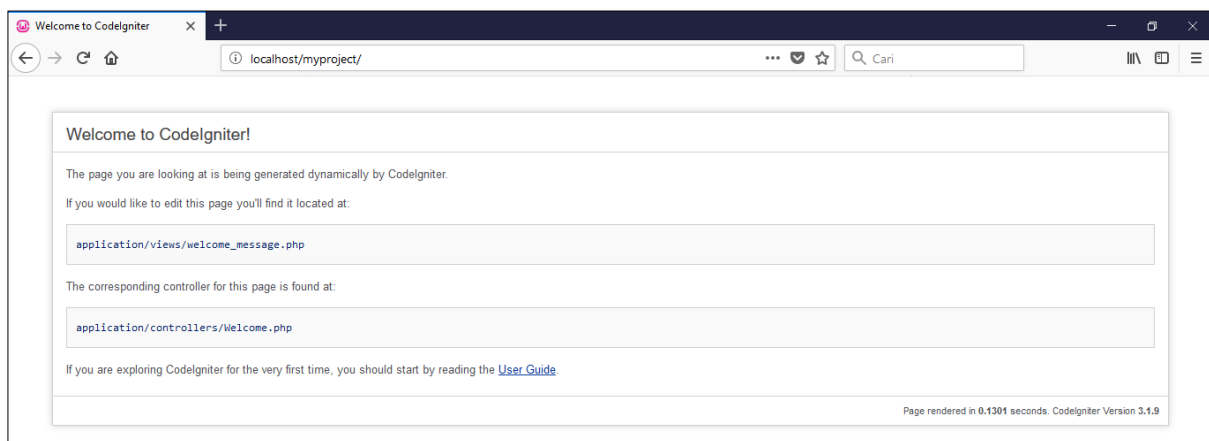


5. Selanjutnya, buka browser Anda. disini saya menggunakan Mozilla Firefox.

Kemudian kunjungi URL berikut:

`http://localhost/myproject/`

Jika instalasi berhasil maka, akan terlihat seperti gambar berikut:

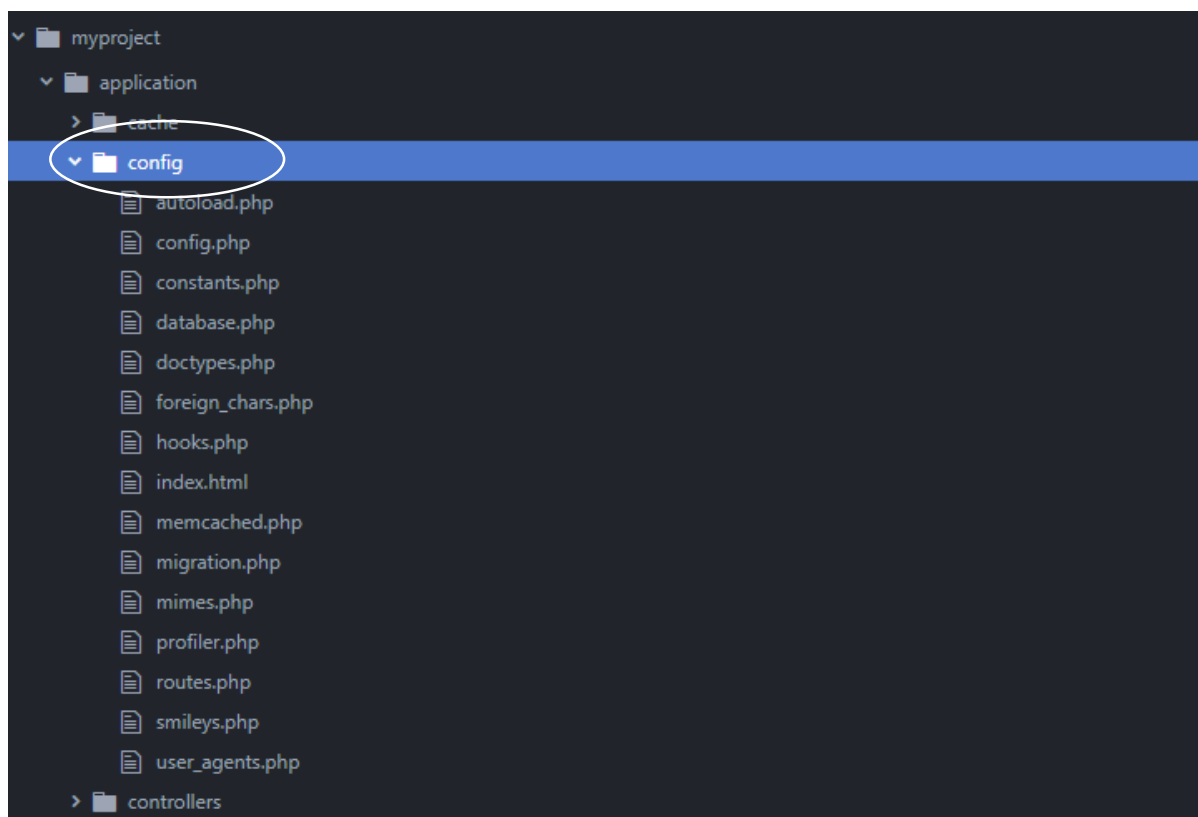


5. Konfigurasi dasar Codeigniter

Dalam memulai codeigniter, ada beberapa konfigurasi dasar yang perlu Anda ketahui.

Yaitu autoload.php, config.php, dan database.php.

Semua konfigurasi pada codeigniter, terletak pada satu tempat yaitu di dalam folder **application/config**.



Bagaimana dan apa saja yang perlu di konfigurasi pada file autoload.php, config.php, dan database.php?

Berikut penjelasannya.

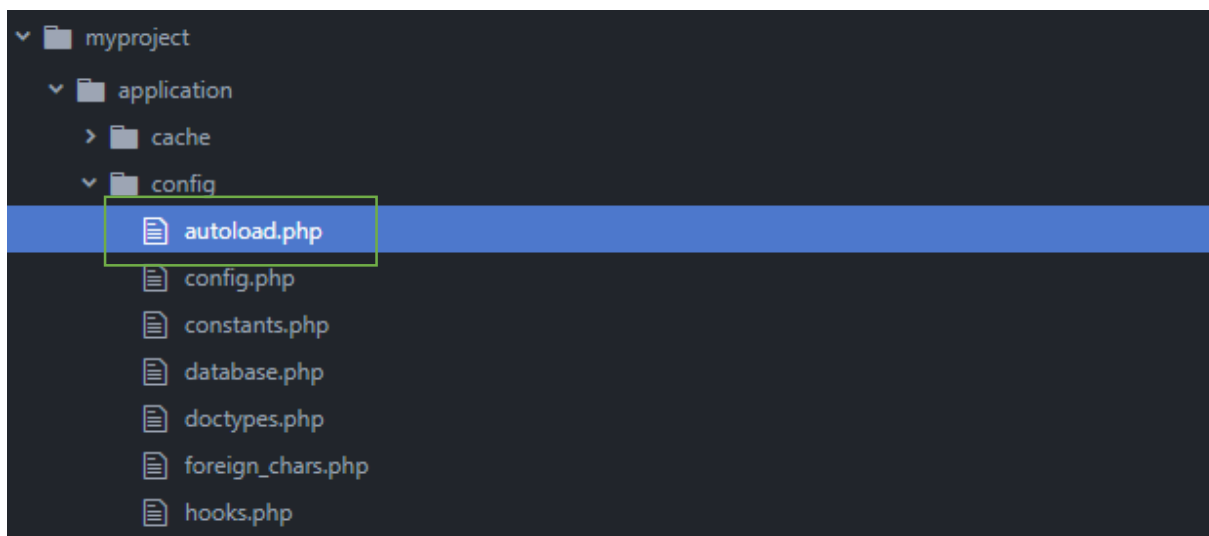
1. Autoload.php

Autoload.php, file ini digunakan untuk mengatur fungsi-fungsi yang akan dimuat otomatis di awal ketika program dijalankan.

Untuk melakukan konfigurasi pada file autoload.php, silahkan buka folder:

application/config/autoload.php

seperti berikut gambar berikut:



Ada beberapa hal yang bisa diloat secara otomatis diantaranya: packages, libraries, drivers, helper files, custom config files, language files, dan models.

Untuk konfigurasi dasar yang perlu Anda ketahui adalah **libraries** dan **helper files**.

Hal ini bertujuan agar beberapa library dan helper tertentu berjalan secara otomatis.

Untuk melakukan konfigurasi pada libraries, buka file autoload.php dengan text editor seperti notepad++, sublime text, atau lainnya.

kemudian temukan kode berikut:

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

Atur menjadi seperti berikut:

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

Pada kode diatas, artinya kita meload library “database” secara otomatis.

Dengan demikian Anda dapat menggunakan fungsi-fungsi database pada codeigniter.

Seperti fungsi: **Query Builder Class**

Selanjutnya, untuk melakukan konfigurasi pada helper files, buka file autoload.php dengan text editor.

kemudian temukan kode berikut:

```
$autoload['helper'] = array();
```

Atur menjadi seperti berikut:

```
$autoload['helper'] = array('url');
```

Pada kode diatas, artinya kita meload helper “url” secara otomatis.

Dengan demikian Anda dapat menggunakan fungsi-fungsi url pada codeigniter.

Seperti fungsi: **base_url()**, **site_url()**, **URI Segment**, dan sebagainya.

2. Config.php

Pada file ini terdapat beberapa konfigurasi yang secara standar sudah terkonfigurasi.

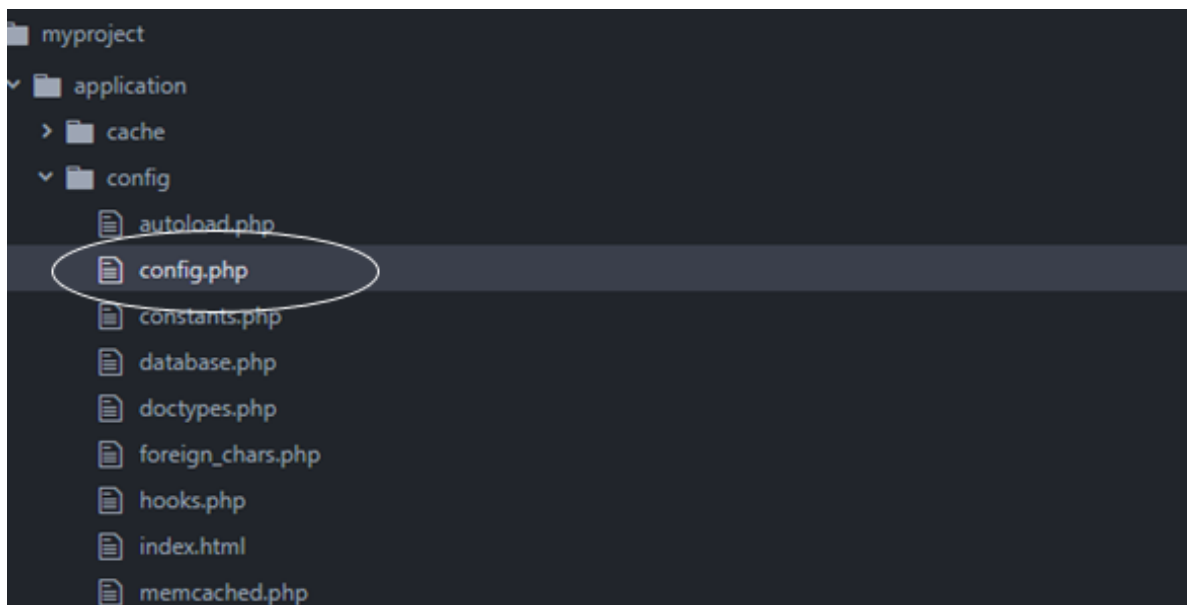
Namun terdapat beberapa konfigurasi yang perlu diperhatikan yaitu:

```
$config['base_url']  
$config['index_page']  
$config['encryption_key']
```

Untuk konfigurasi dasar, Anda cukup mengetahui konfigurasi **base_url**.

Base_url merupakan url dasar dari project Anda.

Untuk mengkonfigurasi base_url, buka file config.php dengan text editor.



kemudian temukan kode berikut:

```
$config['base_url'] = '';
```

Atur menjadi seperti berikut:

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

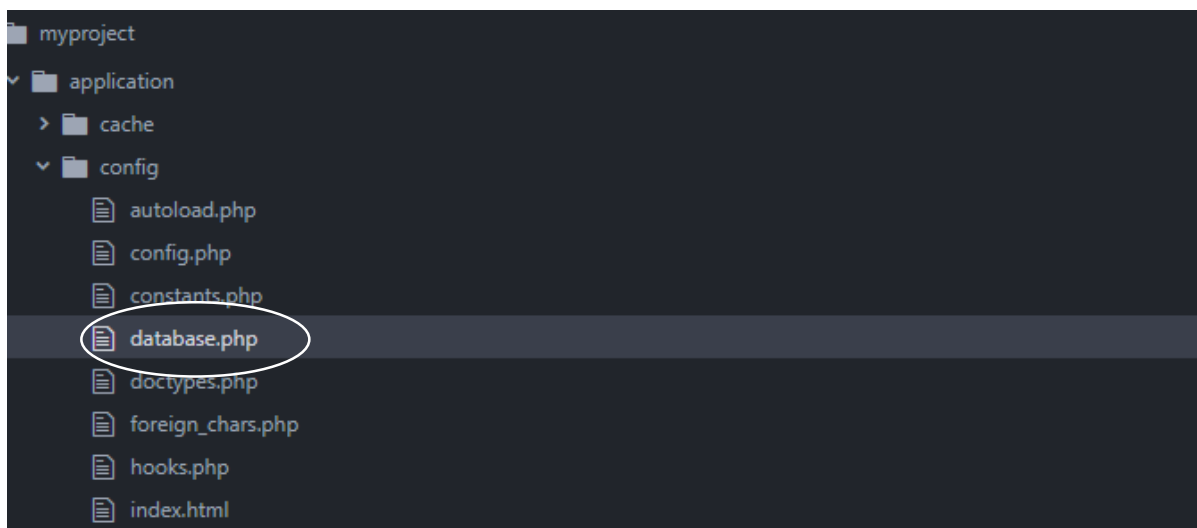
3. Database.php

Dilihat dari nama filenya maka Anda sudah dapat menangkap apa fungsi dari file ini.

File database.php digunakan untuk melakukan konfigurasi yang berkaitan dengan konfigurasi database dari *website* yang akan dibuat.

Adapun konfigurasi yang perlu diperhatikan yaitu: hostname, username, password, dan database.

Untuk melakukan konfigurasi pada database.php. Buka file database.php dengan text editor.



Kemudian temukan kode berikut:

```
$active_group = 'default';
$query_builder = TRUE;

$db['default'] = array(
    'dsn' => '',
    'hostname' => 'localhost',
    'username' => '',
    'password' => '',
```

```
'database' => '',
'dbdriver' => 'mysqli',
'dbprefix' => '',
'pconnect' => FALSE,
'db_debug' => (ENVIRONMENT !== 'production'),
'cache_on' => FALSE,
'cachedir' => '',
'char_set' => 'utf8',
'dbcollat' => 'utf8_general_ci',
'swap_pre' => '',
'encrypt' => FALSE,
'compress' => FALSE,
'stricton' => FALSE,
'failover' => array(),
'save_queries' => TRUE
);
```

Atur menjadi seperti berikut:

```
$active_group = 'default';
$query_builder = TRUE;

$db['default'] = array(
    'dsn' => '',
    'hostname' => 'localhost', // Hostname
    'username' => 'root',      // Username
    'password' => '',         // password
    'database' => 'database_name', //database name
    'dbdriver' => 'mysqli',
    'dbprefix' => '',
    'pconnect' => FALSE,
    'db_debug' => (ENVIRONMENT !== 'production'),
    'cache_on' => FALSE,
```



```
'cachedir' => '',  
'char_set' => 'utf8',  
'dbcollat' => 'utf8_general_ci',  
'swap_pre' => '',  
'encrypt' => FALSE,  
'compress' => FALSE,  
'stricton' => FALSE,  
'failover' => array(),  
'save_queries' => TRUE  
);
```

6. Hello World Codeigniter

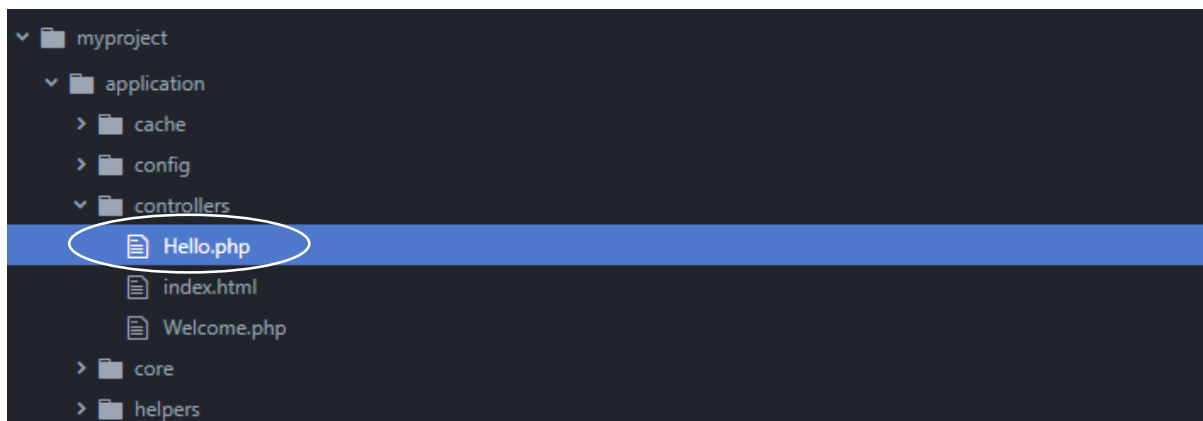
Jika serius dengan codeigniter, Anda harus mengerti bagaimana sebuah controller bekerja.

Untuk lebih jelasnya, saya akan sharing kasus sederhana agar Anda dapat memahami bagaimana controller bekerja.

Disini saya mengangkat kasus yaitu bagaimana menampilkan text "Hello World" pada browser menggunakan controller.

Let's dive right in.

Buat sebuah controller dengan nama **Hello.php** seperti gambar berikut:



Kemudian ketikkan kode berikut:

```
<?php
class Hello extends CI_Controller{

    function index(){
        echo "Hello World";
    }

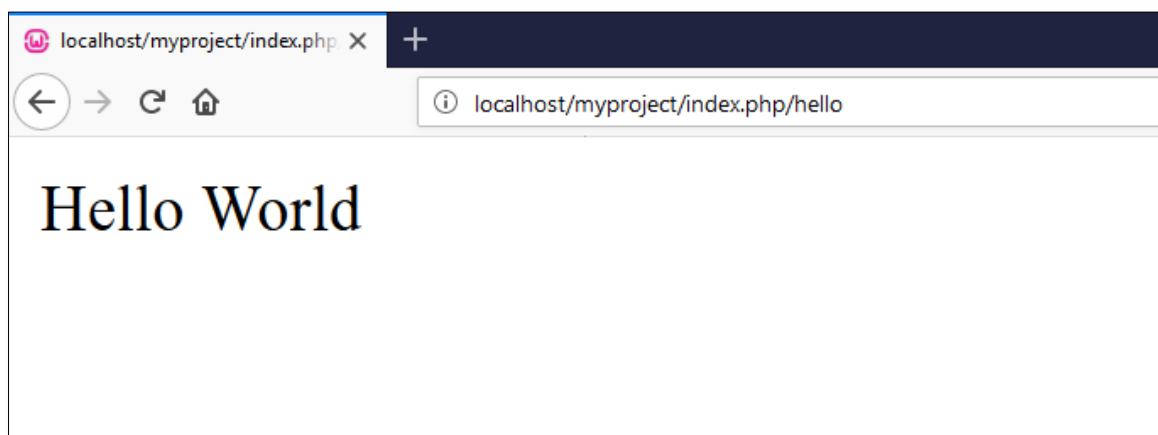
}
```

NB: Setiap penulisan nama file dan nama class selalu di dahului dengan huruf *Capital*.

Setelah itu save dan buka browser Anda, lalu kunjungi url berikut:

`http://localhost/myproject/index.php/hello`

Maka akan akan terlihat text "Hello World" pada browser Anda seperti berikut:



Jika Anda perhatikan dengan seksama, pada dasarnya url pada codeigniter terlihat seperti gambar berikut:



Dimana, terdapat protocol, primary domain, index.php, class name, dan function name.

Mungkin terdegar rumit, tapi sebenarnya tidak.

Untuk lebih jelasnya silahkan tambahkan satu function lagi pada Controller Hello.php. disini saya beri nama "**show**".

Sehingga controller **Hello.php** menjadi seperti berikut:

```
<?php
class Hello extends CI_Controller{

    function index(){
        echo "Hello World";
    }

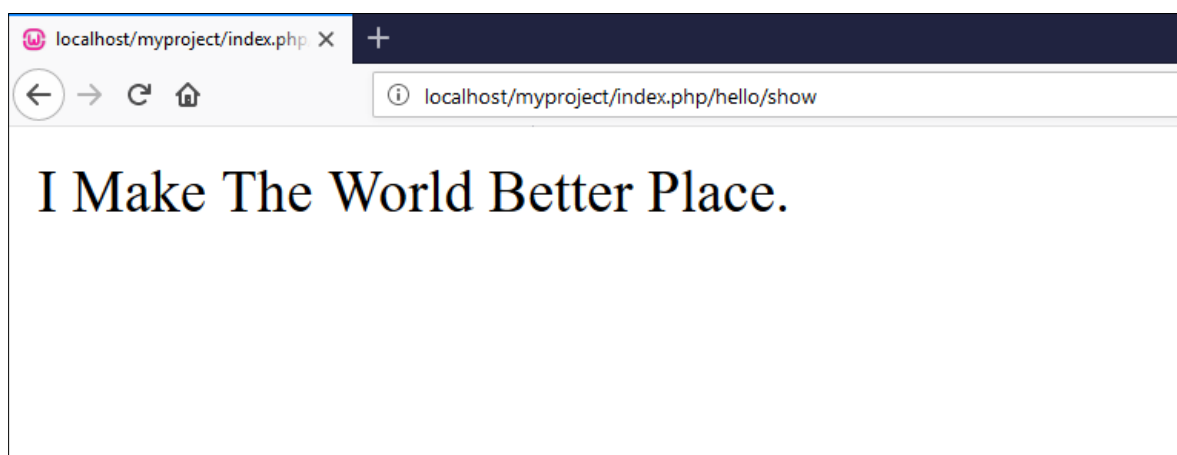
    function show(){
        echo "I Make The World Better Place.";
    }

}
```

Jika Anda jalankan dengan mengunjungi URL berikut:

<http://localhost/myproject/index.php/hello/show>

Maka, akan tampil hasilnya seperti berikut:



7. Menghilangkan index.php pada URL

Codeigniter merupakan framework php yang mendukung clean URL.

Dengan demikian Anda dapat membuat URL yang mudah dibaca dan sekaligus SEO Friendly.

Pada URL aplikasi "Hello World" diatas, dapat dilihat bahwa adanya **index.php** pada url yang terlihat mengganggu.

Adakah cara untuk menghilangkan index.php dari URL?

Tentu saja, Anda dapat menggunakan file **.htaccess** untuk menghilangkannya.

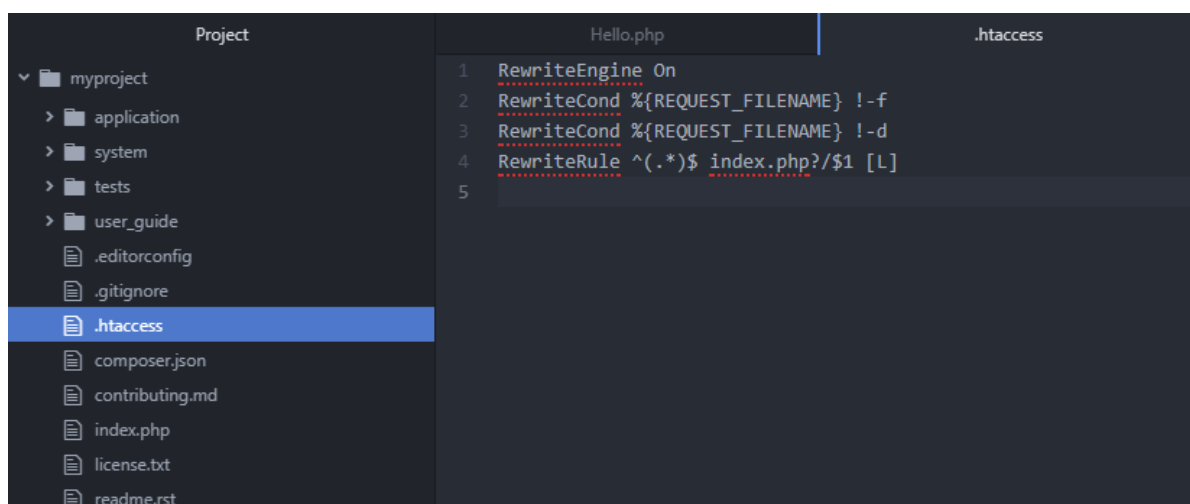
Bagaimana membuat file **.htaccess**?

Mari kita mulai.

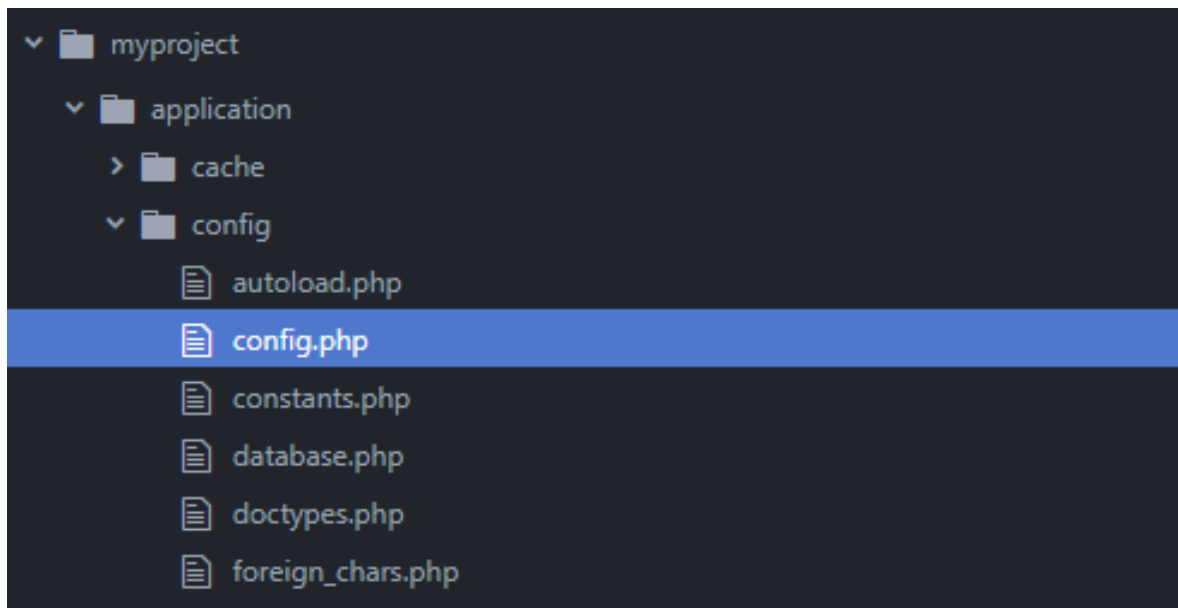
Buat sebuah file dengan nama **.htaccess** pada web root Anda dan ketikkan kode berikut:

```
RewriteEngine On
RewriteCond %{REQUEST_FILENAME} !-f
RewriteCond %{REQUEST_FILENAME} !-d
RewriteRule ^(.*)$ index.php?/$1 [L]
```

Seperti gambar berikut:



Kemudian buka folder **application/config/config.php** dengan text editor.



Kemudian temukan kode berikut:

```
$config['index_page'] = 'index.php';
```

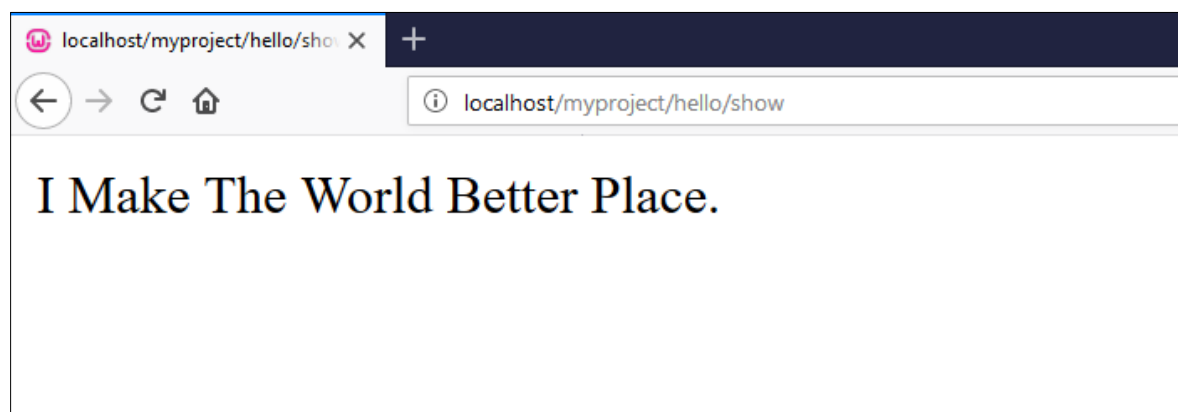
Atur menjadi seperti berikut:

```
$config['index_page'] = '';
```

Sekarang silahkan kunjungi url berikut untuk uji coba:

<http://localhost/myproject/hello/show>

Maka akan terlihat hasilnya seperti berikut:



Pada gambar diatas, dapat dilihat bahwa URL menjadi lebih rapi dan SEO friendly dengan menghilangkan **index.php** pada URL.

8. Controller dan View

Pada kasus sebelumnya, Anda telah mengetahui bagaimana menampilkan text "Hello World" langsung dari controller.

Namun, hal tersebut sebaiknya dilakukan di view.

Sekarang saya akan menunjukkan bagaimana menampilkan view melalui controller.

Mari kita mulai.

Pertama, buat sebuah file pada **application/controller** dengan nama **Blog.php**.

Kemudian ketikkan kode berikut:

```
<?php
class Blog extends CI_Controller
{
    function __construct()
    {
        parent::__construct();
    }

    function index(){
        $this->load->view('blog_view');
    }
}
```


Kedua, buat sebuah file di **application/views** dengan nama **blog_view.php**.

Kemudian ketikkan kode berikut:

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="utf-8">
    <title>My Blog</title>
  </head>
  <body>
    <h1>Welcome To My Blog.</h1>
  </body>
</html>
```

Kemudian, buka browser Anda dan akses controller **blog**. Maka akan terlihat hasilnya seperti berikut:



Anda juga dapat mengirimkan parameter ke view melalui controller.

Sebagai contoh, silahkan ubah controller **Blog.php** menjadi seperti berikut:

```
<?php
class Blog extends CI_Controller
{
```

```

function __construct()
{
    parent::__construct();
}

function index(){
    $data['title']    = "This Is Title";
    $data['content'] = "This Is The Contents";
    $this->load->view('blog_view',$data);
}
}

```

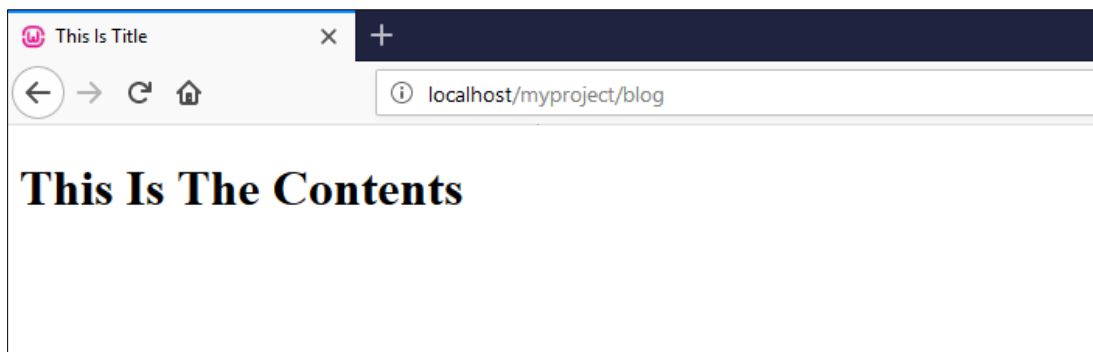
Kemudian ubah view **blog_view.php** menjadi seperti berikut:

```

<!DOCTYPE html>
<html lang="en">
    <head>
        <meta charset="utf-8">
        <title><?php echo $title;?></title>
    </head>
    <body>
        <h1><?php echo $content;?></h1>
    </body>
</html>

```

Kemudian, buka browser Anda dan akses kembali controller **blog**. Maka akan terlihat hasilnya seperti berikut:



Saya harap Anda dapat memahami perbedaannya.

9. Codeigniter dan Bootstrap

Pada kasus sebelumnya, Anda telah memahami bagaimana memanggil view melalui controller.

Sekarang, ada hal yang sangat penting untuk Anda ketahui, yaitu mengkombinasikan codeigniter dengan bootstrap.

Apa itu **BOOTSTRAP**?

Bootstrap adalah toolkit open source untuk dikembangkan dengan HTML, CSS, dan JS.

Dengan kata lain, Bootstrap merupakan framework untuk mempercantik user interface (UI).

Bootstrap bersifat responsive.

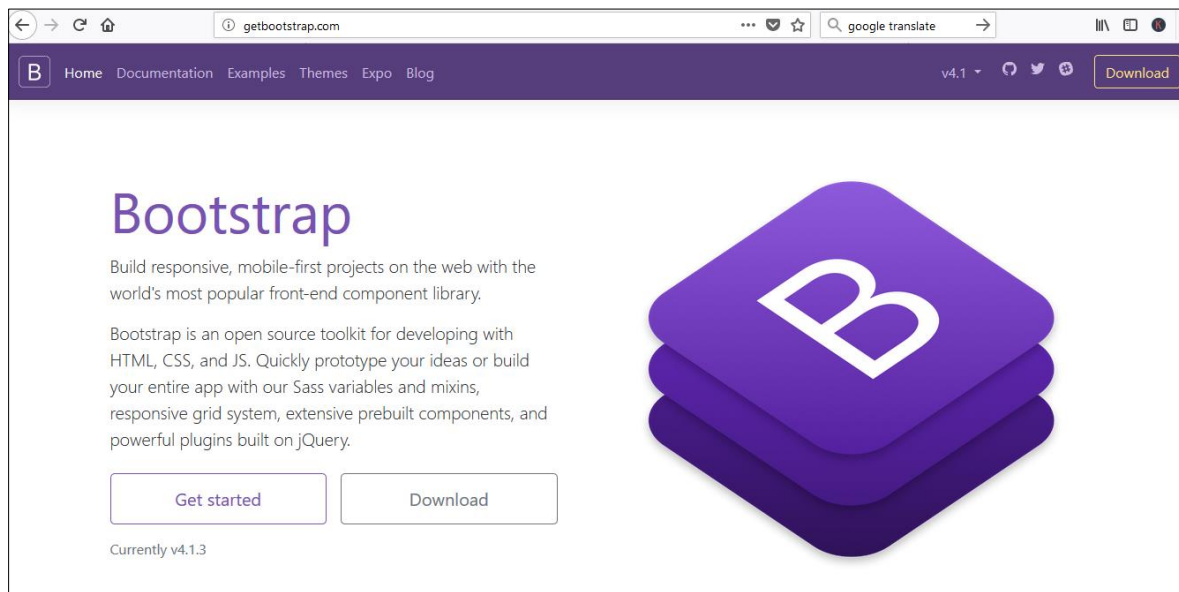
Dengan kata lain, merender dengan baik di berbagai macam perangkat (*platform*) seperti tablet maupun mobile phone.

Keren bukan?

Bagaimana mengkombinasikan codeigniter dan bootstrap?

Let's begin.

Pertama-tama, silahkan download bootstrap di situs resminya **getbootstrap.com**.

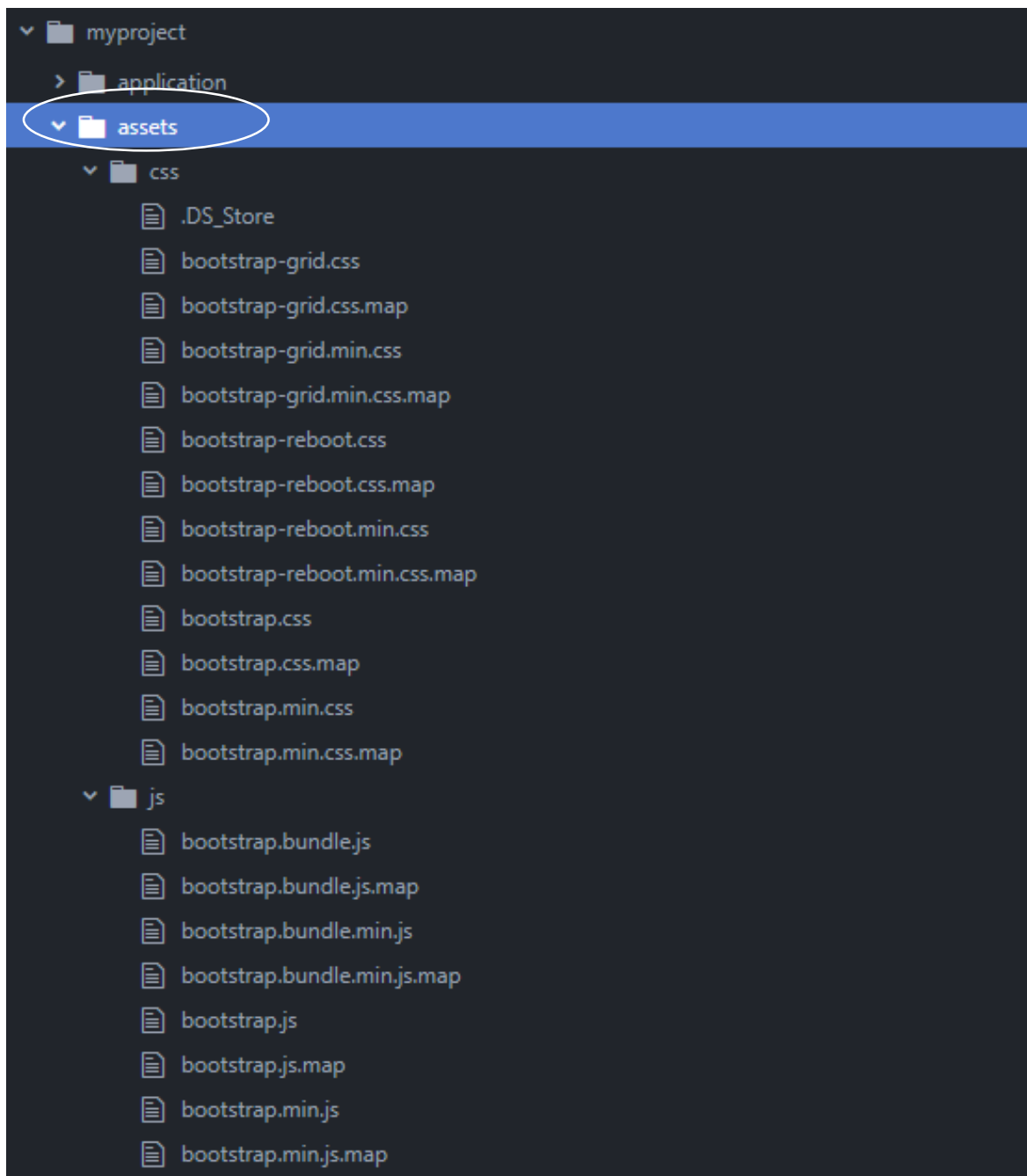


Kedua, buat sebuah folder baru pada project (*webroot*) Anda.

Disini saya beri nama folder "**assets**".

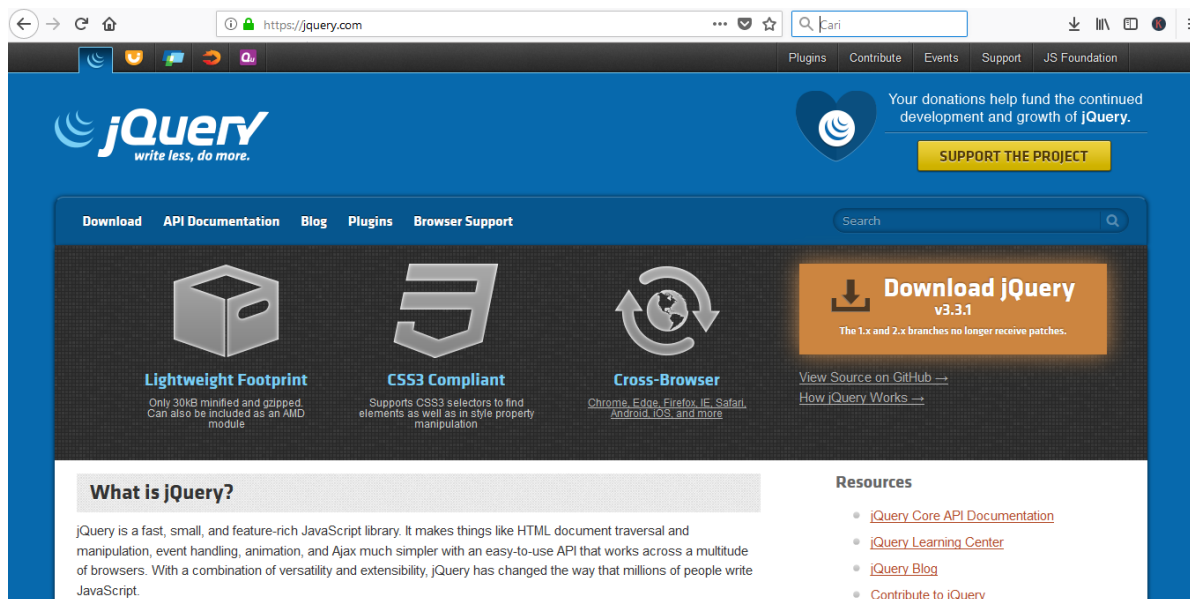
Kemudian extract file bootstrap yang telah di download tadi kedalam folder **assets**.

Seperti berikut:

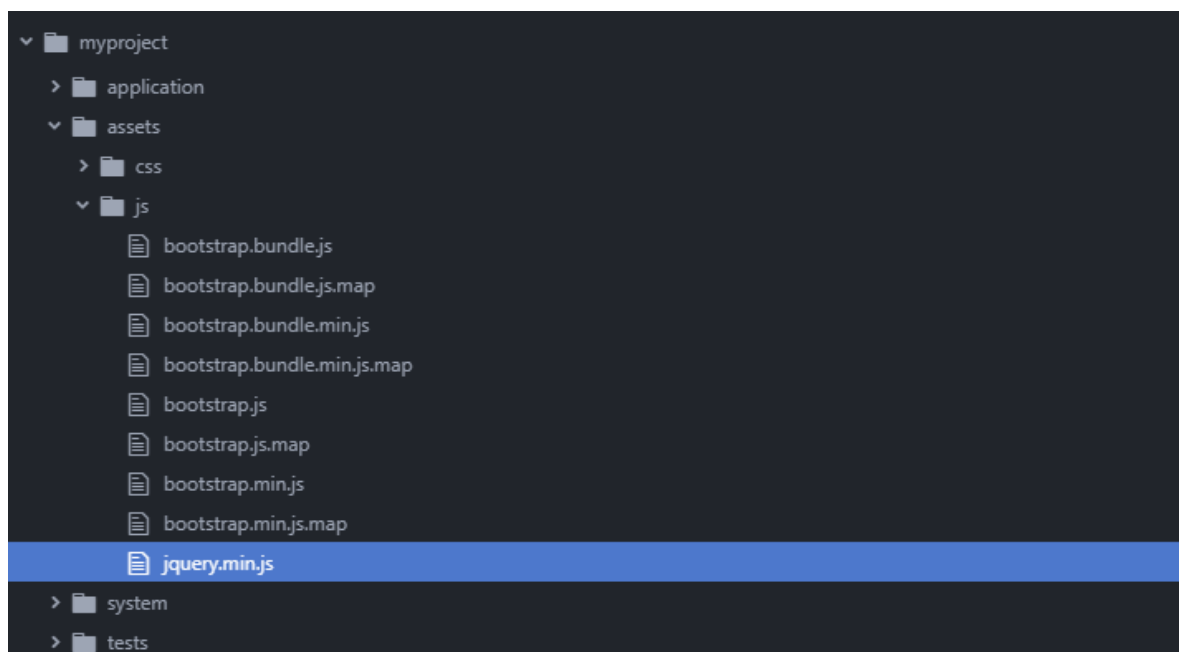


Selain bootstrap, kita juga membutuhkan **jquery** agar javascript pada bootstrap berjalan dengan optimal.

Untuk mendownload JQuery, silahkan download di situs resminya **jquery.com**.



Kemudian, letakkan file jquery pada folder **assets/js/** seperti gambar berikut:



Mungkin terlihat sedikit rumit, tapi sebenarnya tidak.

Agar Anda dapat memahami seperti apa bootstrap, silahkan edit file view

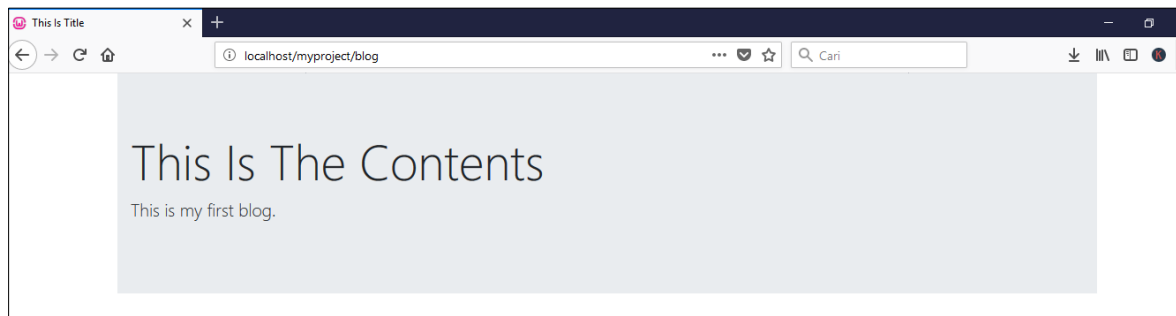
blog_view.php menjadi seperti berikut:

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="utf-8">
    <title><?php echo $title;?></title>
    <!-- load bootstrap css file -->
    <link href="<?php echo base_url('assets/css/bootstrap.min.css');?>"
rel="stylesheet">
  </head>
  <body>

    <div class="container">
      <div class="jumbotron jumbotron-fluid">
        <div class="container">
          <h1 class="display-4"><?php echo $content;?></h1>
          <p class="lead">This is my first blog.</p>
        </div>
      </div>
    </div>

    <!-- load jquery js file -->
    <script src="<?php echo
base_url('assets/js/jquery.min.js');?>"></script>
    <!-- load bootstrap js file -->
    <script src="<?php echo
base_url('assets/js/bootstrap.min.js');?>"></script>
  </body>
</html>
```


Jika Anda panggil lagi controller **blog** pada browser, maka akan terlihat hasilnya seperti berikut:



Pada gambar diatas, Anda dapat melihat bahwa kita tidak perlu membuat kode **css** untuk memberikan style pada suatu halaman website.

Demikian pula, jika Anda membutuhkan table yang cantik. Anda juga tidak perlu mengetikan kode **css** untuk memberikan style pada table tersebut.

Melainkan, anda dapat langsung memiliki table yang cantik secara instan.

Contoh, silahkan edit lagi file view **blog_view.php** menjadi seperti berikut:

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="utf-8">
    <title><?php echo $title;?></title>
    <!-- load bootstrap css file -->
    <link href="<?php echo base_url('assets/css/bootstrap.min.css');?>"
rel="stylesheet">
  </head>
  <body>

    <div class="container">
      <h1><?php echo $content;?></h1>
```

```
<table class="table table-striped">
  <thead>
    <tr>
      <th scope="col">#</th>
      <th scope="col">First</th>
      <th scope="col">Last</th>
      <th scope="col">Handle</th>
    </tr>
  </thead>
  <tbody>
    <tr>
      <th scope="row">1</th>
      <td>Mark</td>
      <td>Otto</td>
      <td>@mdo</td>
    </tr>
    <tr>
      <th scope="row">2</th>
      <td>Jacob</td>
      <td>Thornton</td>
      <td>@fat</td>
    </tr>
    <tr>
      <th scope="row">3</th>
      <td>Larry</td>
      <td>the Bird</td>
      <td>@twitter</td>
    </tr>
  </tbody>
</table>
```

```
</div>
```

```
<!-- load jquery js file -->

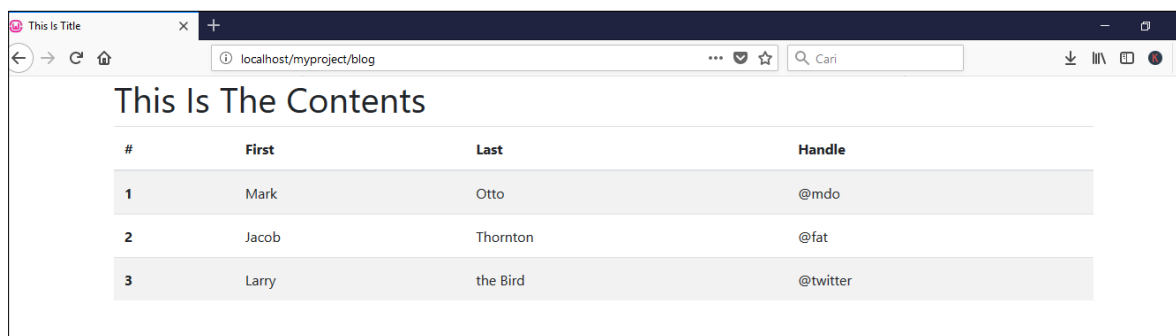
<script src="<?php echo
base_url('assets/js/jquery.min.js');?>"></script>

<!-- load bootstrap js file -->

<script src="<?php echo
base_url('assets/js/bootstrap.min.js');?>"></script>

</body>
</html>
```

Jika Anda jalankan kembali controller **Blog** pada browser, maka Anda akan dapatkan hasilnya seperti berikut:



The screenshot shows a web browser window with the address bar set to localhost/myproject/blog. The page title is "This Is Title". The main content of the page is a table with the heading "This Is The Contents". The table has four columns: "#", "First", "Last", and "Handle". It contains three rows of data.

#	First	Last	Handle
1	Mark	Otto	@mdo
2	Jacob	Thornton	@fat
3	Larry	the Bird	@twitter

Keren bukan?

Semoga dapat dipahami dengan baik.

10. Bekerja dengan Database

Pada segment kali ini, Anda akan belajar semua hal yang Anda butuhkan untuk tahu bagaimana berinteraksi dengan database menggunakan codeigniter.

Mulai dari Create, Read, Update, dan Delete.

Mari kita mulai.

1. Persiapan database.

Pertama-tama buat sebuah database. Disini saya membuat sebuah database dengan nama "**pos_db**".

Jika Anda membuat database dengan nama yang sama, itu lebih baik.

Untuk membuat database, anda dapat mengeksekusi query berikut:

```
CREATE DATABASE pos_db;
```

Query diatas akan membuat sebuah database dengan nama "**pos_db**".

Kemudian, buat table "**product**" dengan struktur sebagai berikut:

Field	Type	Null	Key	Default	Extra
product_id	int(11)	NO	PRI	NULL	auto_increment
product_name	varchar(100)	YES		NULL	
product_price	int(11)	YES		NULL	

Anda dapat mengeksekusi query berikut untuk menghasilkan table dengan struktur seperti diatas.

```
CREATE TABLE product(  
product_id INT PRIMARY KEY AUTO_INCREMENT,
```

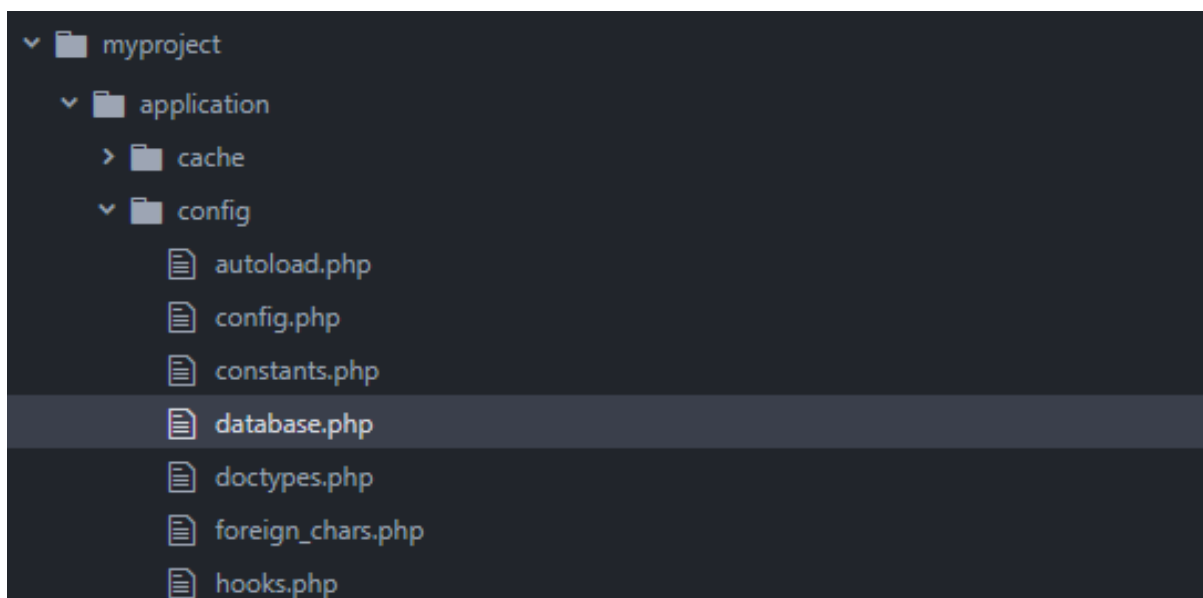
```
product_name VARCHAR(100),  
product_price INT  
);
```

Kemudian masukan beberapa data kedalam table "**product**" dengan mengeksekusi query berikut:

```
INSERT INTO product(product_name,product_price) VALUES  
( 'Coca Cola', '5000'),  
( 'Teh Botol', '3700'),  
( 'You C 1000', '6300'),  
( 'Ponds Men', '18000'),  
( 'Rexona Men', '13000');
```

Langkah selanjutnya adalah mengkoneksikan codeigniter dengan database.

Untuk mengkoneksikan codeigniter dengan database sangatlah sederhana, silahkan buka file **application/config/database.php**



Buka file database.php dengan text editor dan temukan kode berikut:

```
$active_group = 'default';
```

```
$query_builder = TRUE;

$db['default'] = array(
    'dsn' => '',
    'hostname' => 'localhost',
    'username' => '',
    'password' => '',
    'database' => '',
    'dbdriver' => 'mysqli',
    'dbprefix' => '',
    'pconnect' => FALSE,
    'db_debug' => (ENVIRONMENT !== 'production'),
    'cache_on' => FALSE,
    'cachedir' => '',
    'char_set' => 'utf8',
    'dbcollat' => 'utf8_general_ci',
    'swap_pre' => '',
    'encrypt' => FALSE,
    'compress' => FALSE,
    'stricton' => FALSE,
    'failover' => array(),
    'save_queries' => TRUE
);
```

Lalu setting menjadi seperti berikut:

```
$active_group = 'default';
$query_builder = TRUE;

$db['default'] = array(
    'dsn' => '',
    'hostname' => 'localhost',
    'username' => 'root',
```

```
'password' => '',
'database' => 'pos_db',
'dbdriver' => 'mysqli',
'dbprefix' => '',
'pconnect' => FALSE,
'db_debug' => (ENVIRONMENT !== 'production'),
'cache_on' => FALSE,
'cachedir' => '',
'char_set' => 'utf8',
'dbcollat' => 'utf8_general_ci',
'swap_pre' => '',
'encrypt' => FALSE,
'compress' => FALSE,
'stricton' => FALSE,
'failover' => array(),
'save_queries' => TRUE
);
```

Silahkan jalankan lagi project Anda pada browser, jika tidak ada error berarti koneksi ke database berhasil.

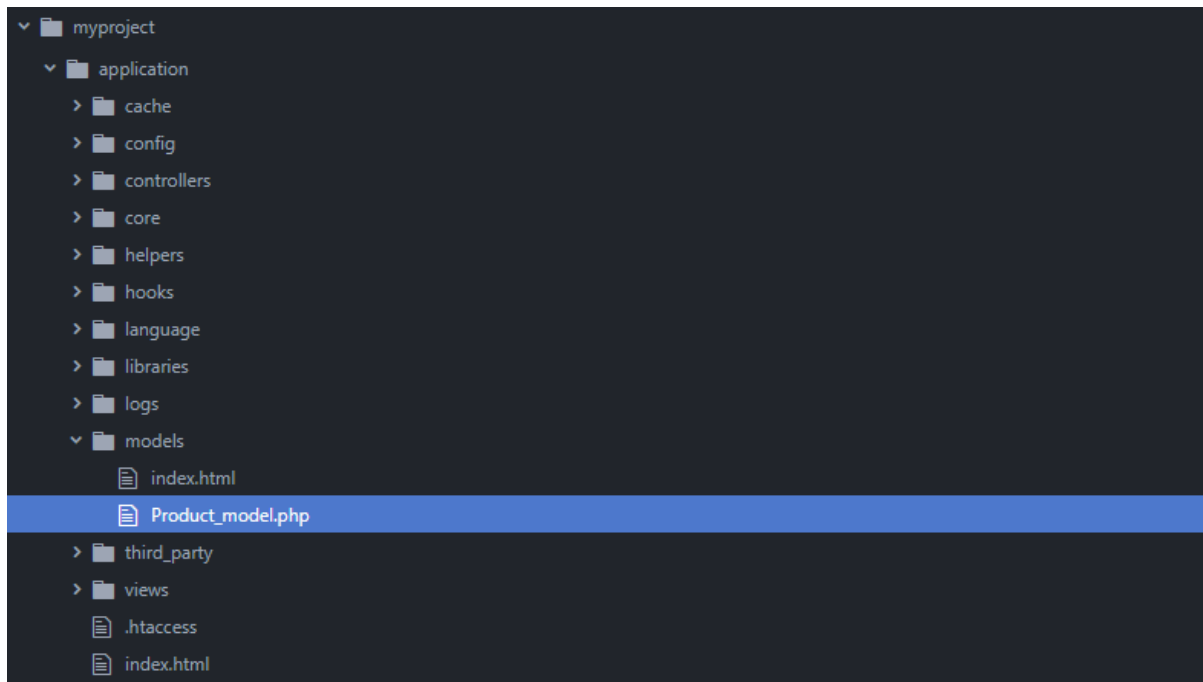
2. Menampilkan dari database ke view (Read).

Pada segment kali ini, saya akan menunjukkan kepada Anda bagaimana menampilkan data dari database ke view.

Mari kita mulai.

#1. Buat sebuah file didalam **application/models** dengan nama **"Product_model.php"**.

Seperti gambar berikut:



Buka file “Product_model.php” dengan text editor. Kemudian ketikan kode berikut:

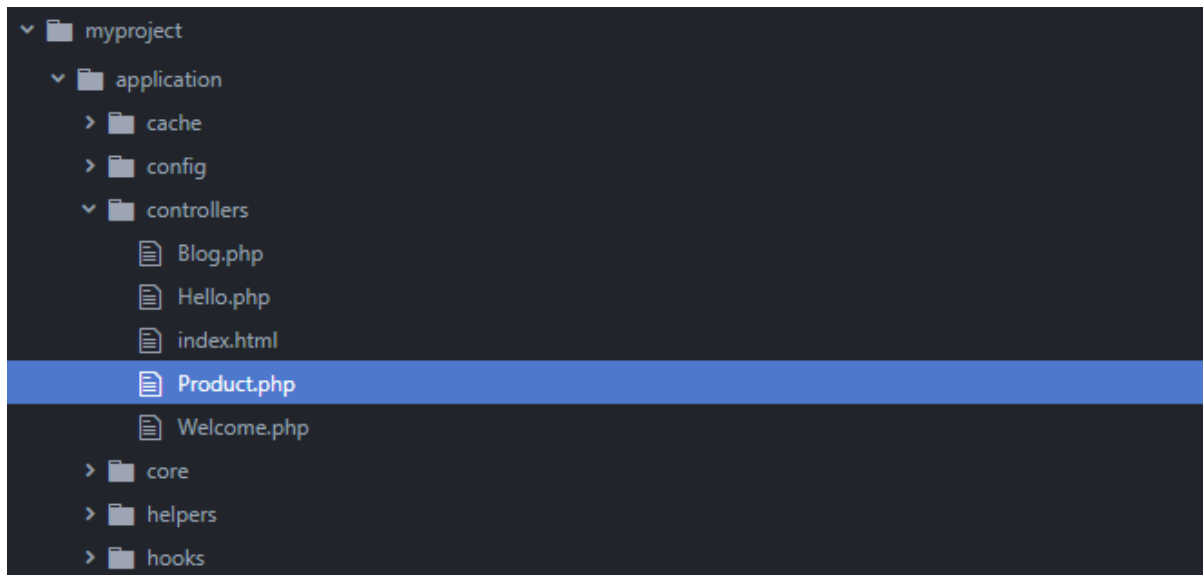
```
<?php
class Product_model extends CI_Model{

    function get_product(){
        $result = $this->db->get('product');
        return $result;
    }

}
```

#2. Buat sebuah file didalam **application/controllers** dengan nama “**Product.php**”.

Seperti gambar berikut:

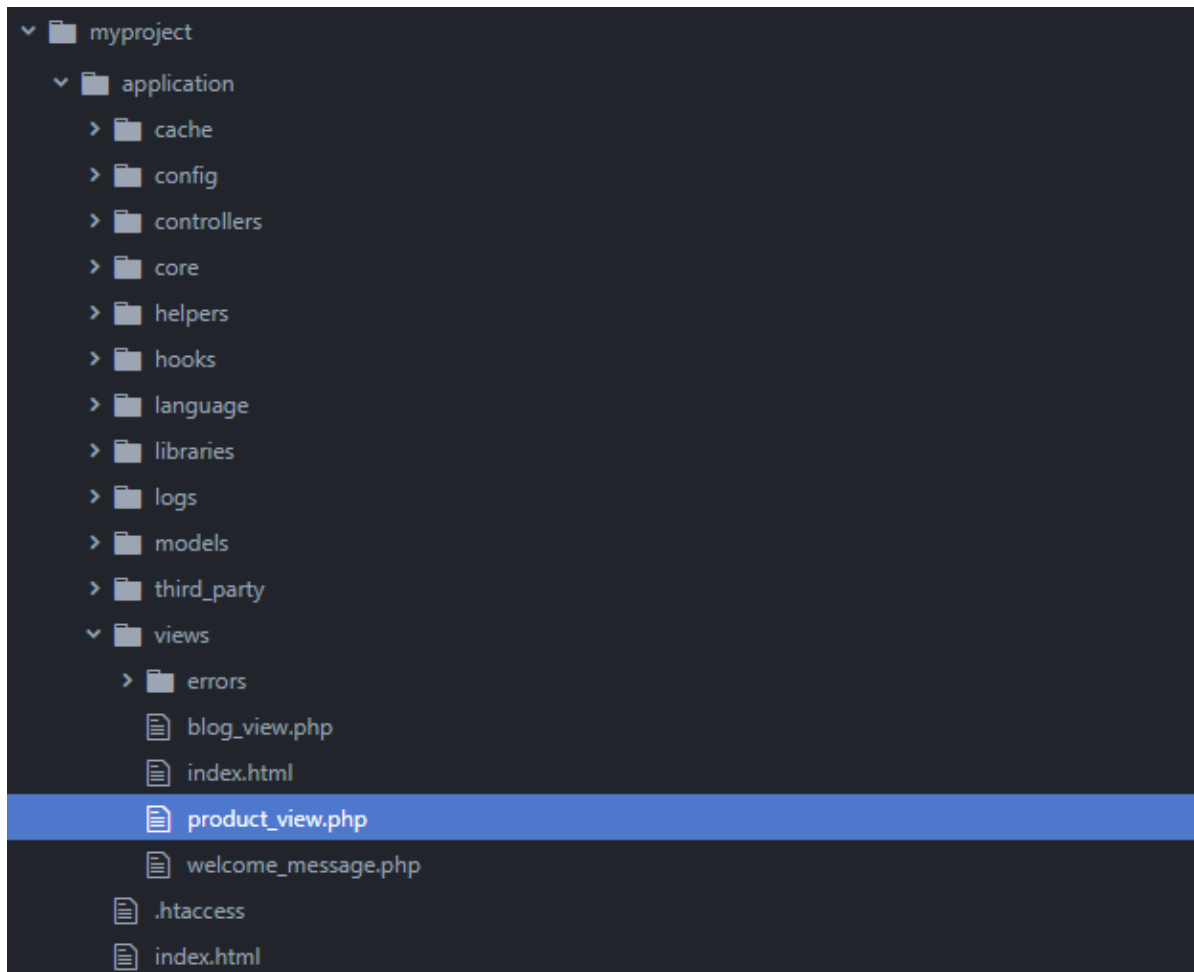


Buka file controller "Product.php" dengan text editor. Kemudian ketikkan kode berikut:

```
<?php
class Product extends CI_Controller{
    function __construct(){
        parent::__construct();
        $this->load->model('product_model');
    }
    function index(){
        $data['product'] = $this->product_model->get_product();
        $this->load->view('product_view',$data);
    }
}
```

#3. Buat sebuah file view dengan nama "**product_view.php**".

Seperti gambar berikut:



kemudian ketikkan kode berikut:

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="utf-8">
    <title>Product List</title>
    <!-- load bootstrap css file -->
    <link href="<?php echo base_url('assets/css/bootstrap.min.css');?>"
rel="stylesheet">
  </head>
  <body>

    <div class="container">
      <h1><center>Product List</center></h1>
      <table class="table table-striped">
```

```

        <thead>
            <tr>
                <th scope="col">#</th>
                <th scope="col">Product Name</th>
                <th scope="col">Price</th>
            </tr>
        </thead>
        <?php
            $count = 0;
            foreach ($product->result() as $row) :
                $count++;
            ?>
            <tr>
                <th scope="row"><?php echo $count;?></th>
                <td><?php echo $row->product_name;?></td>
                <td><?php echo number_format($row->product_price);?></td>
            </tr>
        <?php endforeach;?>
    </tbody>
</table>

</div>

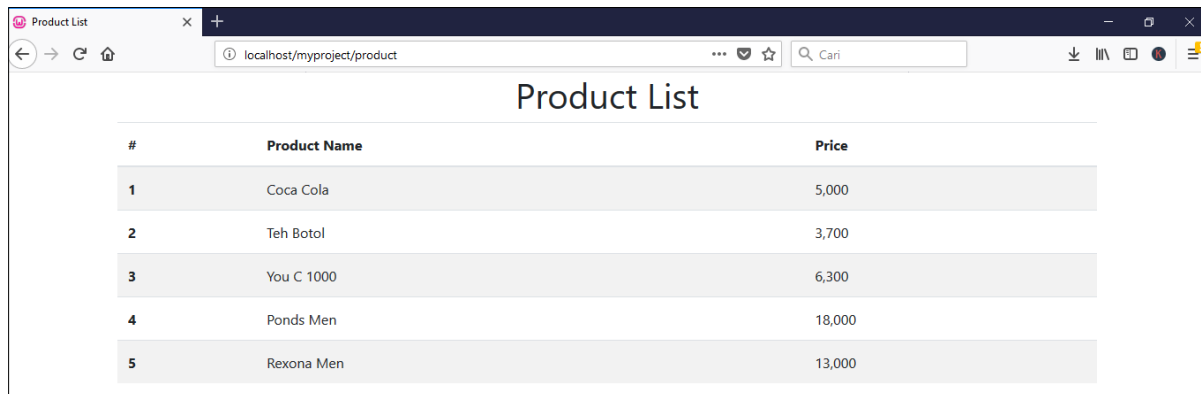
<!-- load jquery js file -->
<script src="<?php echo
base_url('assets/js/jquery.min.js');?>"></script>
<!-- load bootstrap js file -->
<script src="<?php echo
base_url('assets/js/bootstrap.min.js');?>"></script>
</body>
</html>

```

Kemudian, jalankan controller “**Product**” melalui browser Anda, dengan mengunjungi url berikut:

<http://localhost/myproject/product>

Maka akan terlihat hasilnya seperti berikut:



#	Product Name	Price
1	Coca Cola	5,000
2	Teh Botol	3,700
3	You C 1000	6,300
4	Ponds Men	18,000
5	Rexona Men	13,000

3. Insert data ke database (Create).

Pada segment kali ini, saya akan menunjukkan kepada Anda bagaimana insert data ke database.

Mari kita mulai.

#1. Buka file model "**Product_model.php**". kemudian tambahkan satu function lagi seperti berikut:

```
<?php
class Product_model extends CI_Model{

    function get_product(){
        $result = $this->db->get('product');
        return $result;
    }

    function save($product_name,$product_price){
        $data = array(
            'product_name' => $product_name,
            'product_price' => $product_price
```

```

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

```

#2. Buka file controller "**Product.php**". kemudian tambahkan beberapa function lagi seperti berikut:

```

<?php
class Product extends CI_Controller{
    function __construct(){
        parent::__construct();
        $this->load->model('product_model');
    }
    function index(){
        $data['product'] = $this->product_model->get_product();
        $this->load->view('product_view',$data);
    }
    function add_new(){
        $this->load->view('add_product_view');
    }
    function save(){
        $product_name = $this->input->post('product_name');
        $product_price = $this->input->post('product_price');
        $this->product_model->save($product_name,$product_price);
        redirect('product');
    }
}

```

#3. Buat sebuah file view lagi dengan nama "**add_product_view.php**". dengan kode seperti berikut:

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="utf-8">
    <title>Add New Product</title>
    <!-- load bootstrap css file -->
    <link href="<?php echo base_url('assets/css/bootstrap.min.css');?>"
rel="stylesheet">
  </head>
  <body>

    <div class="container">
      <h1><center>Add New Product</center></h1>
      <div class="col-md-6 offset-md-3">
        <form action="<?php echo site_url('product/save');?>"
method="post">
          <div class="form-group">
            <label>Product Name</label>
            <input type="text" class="form-control" name="product_name"
placeholder="Product Name">
          </div>
          <div class="form-group">
            <label>Price</label>
            <input type="text" class="form-control" name="product_price"
placeholder="Price">
          </div>
          <button type="submit" class="btn btn-primary">Submit</button>
        </form>
      </div>
    </div>

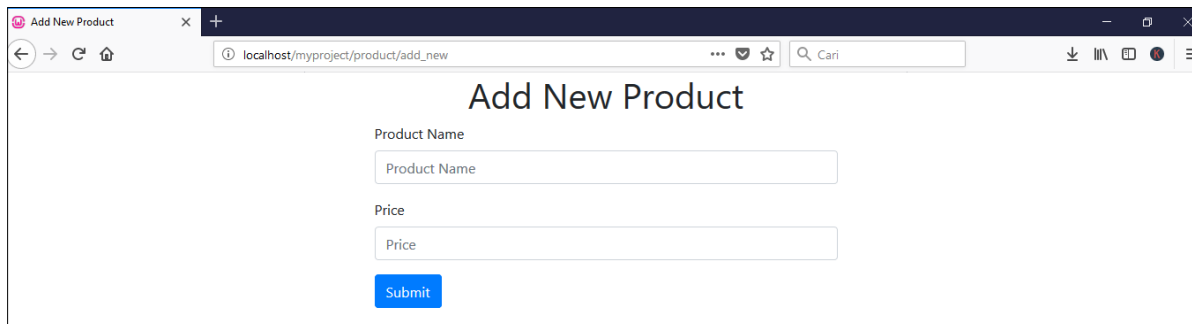
    <!-- load jquery js file -->
    <script src="<?php echo
base_url('assets/js/jquery.min.js');?>"></script>
    <!-- load bootstrap js file -->
```

```
<script src="<?php echo  
base_url('assets/js/bootstrap.min.js');?>"></script>  
  
</body>  
</html>
```

Kemudian, kembali ke browser dan ketikkan url berikut pada browser Anda:

http://localhost/myproject/product/add_new

Maka akan terlihat hasilnya seperti berikut:

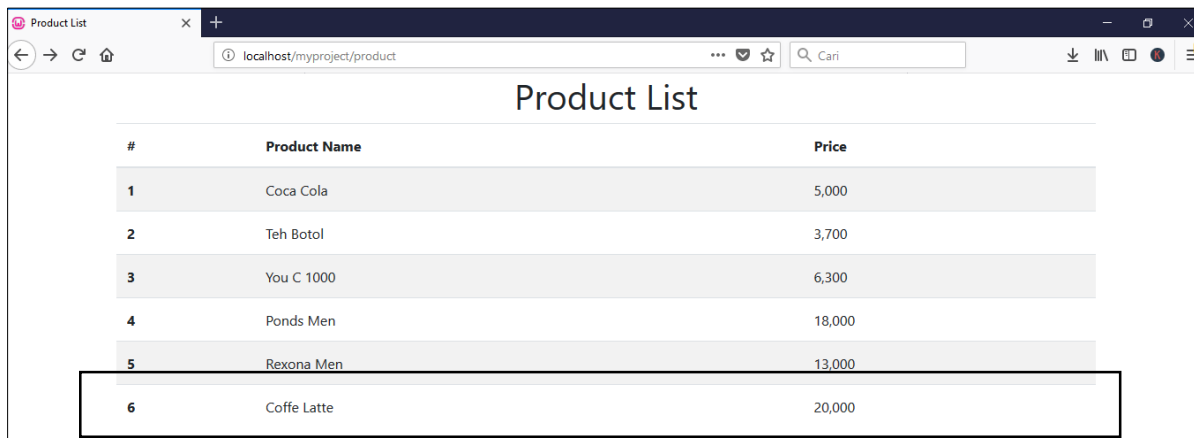


The screenshot shows a web browser window with the title 'Add New Product'. The address bar displays 'localhost/myproject/product/add_new'. The main content area contains a form with the following elements:

- Product Name:** A text input field with the placeholder text 'Product Name'.
- Price:** A text input field with the placeholder text 'Price'.
- Submit:** A blue button labeled 'Submit'.

Masukkan **product name** dan **price** pada textbox, kemudian klik tombol submit.

Maka akan terlihat datanya pada product list seperti berikut:



The screenshot shows a web browser window with the title 'Product List'. The address bar displays 'localhost/myproject/product'. The main content area contains a table with the following data:

#	Product Name	Price
1	Coca Cola	5,000
2	Teh Botol	3,700
3	You C 1000	6,300
4	Ponds Men	18,000
5	Rexona Men	13,000
6	Coffe Latte	20,000

4. Delete data ke database (Delete).

Pada segment kali ini, saya akan menunjukkan kepada Anda bagaimana menghapus (*delete*) data ke database.

Mari kita mulai.

#1. Buka file view "**product_view.php**". kemudian ubah menjadi seperti berikut:

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="utf-8">
    <title>Product List</title>
    <!-- load bootstrap css file -->
    <link href="<?php echo base_url('assets/css/bootstrap.min.css');?>"
rel="stylesheet">
  </head>
  <body>

    <div class="container">
      <h1><center>Product List</center></h1>
      <table class="table table-striped">
        <thead>
          <tr>
            <th scope="col">#</th>
            <th scope="col">Product Name</th>
            <th scope="col">Price</th>
            <th width="200">Action</th>
          </tr>
        </thead>
        <?php
          $count = 0;
          foreach ($product->result() as $row) :
            $count++;
```



```

        ?>
        <tr>
            <th scope="row"><?php echo $count;?></th>
            <td><?php echo $row->product_name;?></td>
            <td><?php echo number_format($row->product_price);?></td>
            <td>
                <a href="<?php echo
site_url('product/delete/' . $row->product_id);?>" class="btn btn-sm btn-
danger">Delete</a>
            </td>
        </tr>
    <?php endforeach;?>
</tbody>
</table>

</div>

<!-- load jquery js file -->
<script src="<?php echo
base_url('assets/js/jquery.min.js');?>"></script>
<!-- load bootstrap js file -->
<script src="<?php echo
base_url('assets/js/bootstrap.min.js');?>"></script>
</body>
</html>

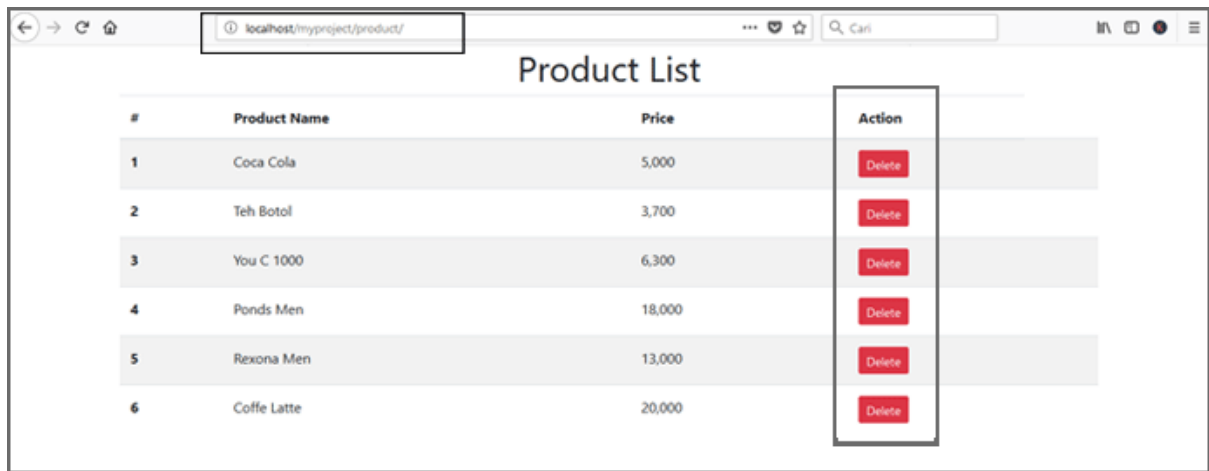
```

Pada file “**product_view.php**” diatas kita menambahkan satu kolom lagi pada table **product list**. Yaitu kolom **action**.

Pada kolom action terdapat tombol **delete**. Sehingga jika Anda jalankan Controller **product**, dengan mengunjungi url berikut:

http://localhost/myproject/product/

Maka akan terlihat hasilnya seperti berikut:



#	Product Name	Price	Action
1	Coca Cola	5,000	Delete
2	Teh Botol	3,700	Delete
3	You C 1000	6,300	Delete
4	Ponds Men	18,000	Delete
5	Rexona Men	13,000	Delete
6	Coffe Latte	20,000	Delete

#2. Tambahkan sebuah **function delete** pada controller **Product.php**.

Adapun kodenya sebagai berikut:

```
function delete(){
    $product_id = $this->uri->segment(3);
    $this->product_model->delete($product_id);
    redirect('product');
}
```

Sehingga terlihat kode lengkap dari controller **Product.php** seperti berikut:

```
<?php
class Product extends CI_Controller{
    function __construct(){
        parent::__construct();
        $this->load->model('product_model');
    }
    function index(){
        $data['product'] = $this->product_model->get_product();
        $this->load->view('product_view',$data);
    }
    function add_new(){
        $this->load->view('add_product_view');
```

```

}
function save(){
    $product_name = $this->input->post('product_name');
    $product_price = $this->input->post('product_price');
    $this->product_model->save($product_name,$product_price);
    redirect('product');
}
function delete(){
    $product_id = $this->uri->segment(3);
    $this->product_model->delete($product_id);
    redirect('product');
}
}

```

#3. Tambahkan sebuah **function delete** pada mode **Product_model.php**.

Adapun kodenya sebagai berikut:

```

function delete($product_id){
    $this->db->where('product_id', $product_id);
    $this->db->delete('product');
}

```

Sehingga terlihat kode lengkap dari model **Product_model.php** seperti berikut:

```

<?php
class Product_model extends CI_Model{
    function get_product(){
        $result = $this->db->get('product');
        return $result;
    }
    function save($product_name,$product_price){
        $data = array(

```

```

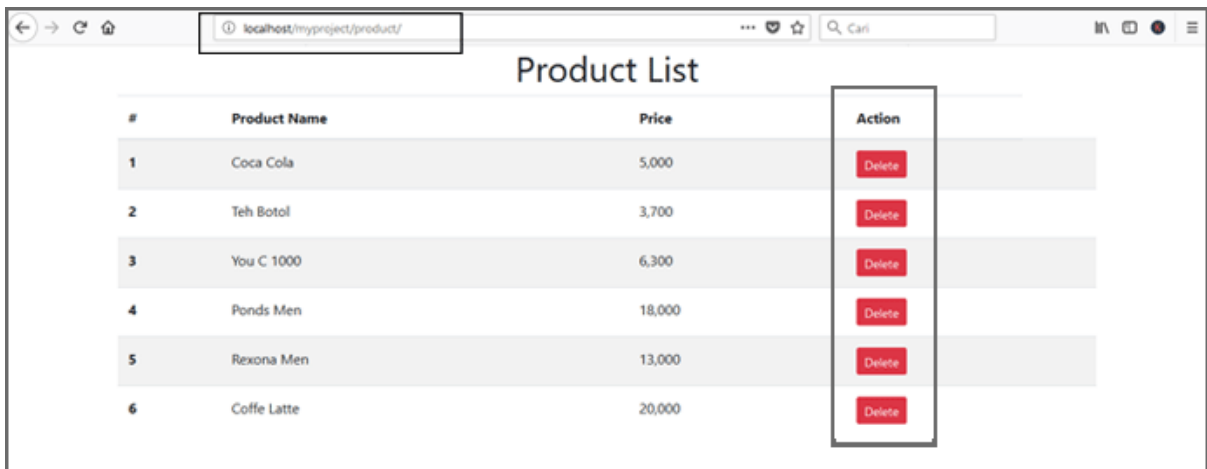
        'product_name' => $product_name,
        'product_price' => $product_price
    );
    $this->db->insert('product',$data);
}
function delete($product_id){
    $this->db->where('product_id', $product_id);
    $this->db->delete('product');
}
}

```

Sekarang kembali ke browser dan kunjungi url berikut:

<http://localhost/myproject/product>

Maka akan tampil product list seperti gambar berikut:



#	Product Name	Price	Action
1	Coca Cola	5,000	Delete
2	Teh Botol	3,700	Delete
3	You C 1000	6,300	Delete
4	Ponds Men	18,000	Delete
5	Rexona Men	13,000	Delete
6	Coffie Latte	20,000	Delete

Silahkan klik satu dari tombol **delete** pada kolom **action** untuk menghapus record.

Selesai.

5. Update data ke database (Update).

Anda telah mengetahui bagaimana menampilkan data (READ) dari database ke view, Meng-insert data ke database (CREATE), dan menghapus data ke database (DELETE).

Sekarang waktunya untuk mengetahui bagaimana mengubah data ke database (UPDATE).

Mari kita mulai.

#1. Buka file view “**product_view.php**”. kemudian ubah menjadi seperti berikut:

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="utf-8">
    <title>Product List</title>
    <!-- load bootstrap css file -->
    <link href="<?php echo base_url('assets/css/bootstrap.min.css');?>"
rel="stylesheet">
  </head>
  <body>

    <div class="container">
      <h1><center>Product List</center></h1>
      <table class="table table-striped">
        <thead>
          <tr>
            <th scope="col">#</th>
            <th scope="col">Product Name</th>
            <th scope="col">Price</th>
            <th width="200">Action</th>
          </tr>
```

```

</thead>

<?php
    $count = 0;
    foreach ($product->result() as $row) :
        $count++;
    ?>

    <tr>
        <th scope="row"><?php echo $count;?></th>
        <td><?php echo $row->product_name;?></td>
        <td><?php echo number_format($row->product_price);?></td>
        <td>
            <a href="<?php echo site_url('product/get_edit/' . $row->product_id);?>" class="btn btn-sm btn-info">Update</a>
            <a href="<?php echo site_url('product/delete/' . $row->product_id);?>" class="btn btn-sm btn-danger">Delete</a>
        <td>
    </tr>
<?php endforeach;?>
</tbody>
</table>

</div>

<!-- load jquery js file -->
<script src="<?php echo
base_url('assets/js/jquery.min.js');?>"></script>
<!-- load bootstrap js file -->
<script src="<?php echo
base_url('assets/js/bootstrap.min.js');?>"></script>
</body>
</html>

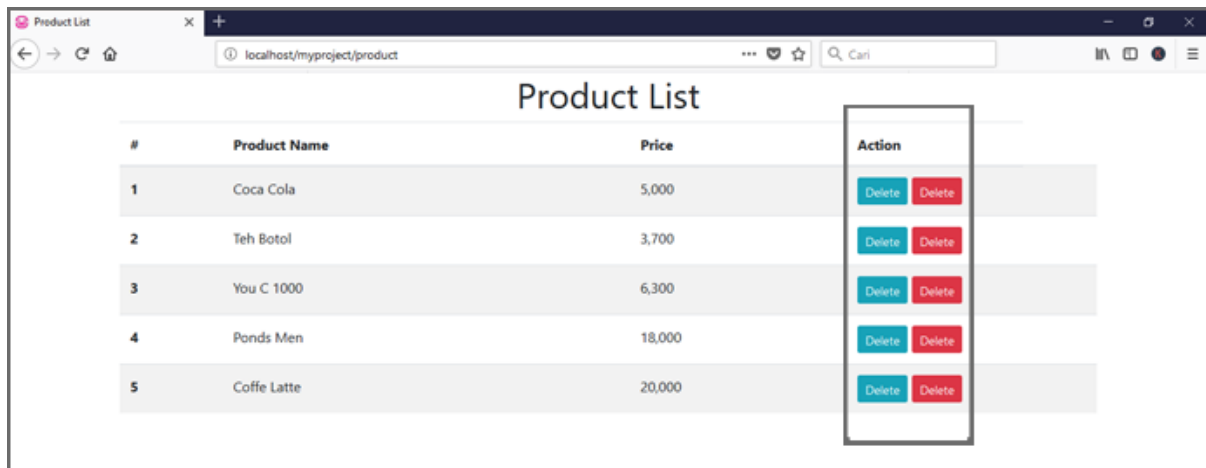
```

Pada file “**product_view.php**” diatas kita menambahkan satu tombol lagi pada kolom **action**. Yaitu tombol **edit**.

Sehingga jika Anda jalankan Controller **product**, dengan mengunjungi url berikut:

<http://localhost/myproject/product/>

Maka akan terlihat hasilnya seperti berikut:



#	Product Name	Price	Action
1	Coca Cola	5,000	<button>Delete</button> <button>Delete</button>
2	Teh Botol	3,700	<button>Delete</button> <button>Delete</button>
3	You C 1000	6,300	<button>Delete</button> <button>Delete</button>
4	Ponds Men	18,000	<button>Delete</button> <button>Delete</button>
5	Coffie Latte	20,000	<button>Delete</button> <button>Delete</button>

#2. Tambahkan sebuah **function get_edit** pada controller **Product.php**.

Adapun kodenya sebagai berikut:

```
function get_edit(){
    $product_id = $this->uri->segment(3);
    $result = $this->product_model->get_product_id($product_id);
    if($result->num_rows() > 0){
        $i = $result->row_array();
        $data = array(
            'product_id'      => $i['product_id'],
            'product_name'    => $i['product_name'],
            'product_price'   => $i['product_price']
        );
        $this->load->view('edit_product_view',$data);
    }else{
        echo "Data Was Not Found";
    }
}
```

#3. Tambahkan sebuah **function get_product_id** pada controller

Product_model.php.

Adapun kodenya sebagai berikut:

```
function get_product_id($product_id){  
    $query = $this->db->get_where('product', array('product_id' =>  
$product_id));  
    return $query;  
}
```

#4. Buat sebuah view lagi dengan nama **edit_product_view.php**. Kemudian ketikkan kode berikut:

```
<!DOCTYPE html>  
<html lang="en">  
    <head>  
        <meta charset="utf-8">  
        <title>Edit Product</title>  
        <!-- load bootstrap css file -->  
        <link href="<?php echo base_url('assets/css/bootstrap.min.css');?>"  
rel="stylesheet">  
    </head>  
    <body>  
  
        <div class="container">  
            <h1><center>Edit Product</center></h1>  
            <div class="col-md-6 offset-md-3">  
                <form action="<?php echo site_url('product/update');?>"  
method="post">  
                    <div class="form-group">
```



```

        <label>Product Name</label>

        <input type="text" class="form-control" name="product_name"
value="<?php echo $product_name;?>" placeholder="Product Name">

    </div>

    <div class="form-group">

        <label>Price</label>

        <input type="text" class="form-control" name="product_price"
value="<?php echo $product_price;?>" placeholder="Price">

    </div>

        <input type="hidden" name="product_id" value="<?php echo
$product_id?>">

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

    </form>

</div>

</div>

<!-- load jquery js file -->
<script src="<?php echo
base_url('assets/js/jquery.min.js');?>"></script>

<!-- load bootstrap js file -->
<script src="<?php echo
base_url('assets/js/bootstrap.min.js');?>"></script>

</body>
</html>

```

#5. Tambahkan sebuah **function update** pada controller **Product.php**.

Adapun kodenya sebagai berikut:

```

function update(){
    $product_id = $this->input->post('product_id');
    $product_name = $this->input->post('product_name');
    $product_price = $this->input->post('product_price');
    $this->product_model-
>update($product_id,$product_name,$product_price);
    redirect('product');
}

```

```
}
```

Sehingga terlihat kode lengkap dari controller **Product.php** seperti berikut:

```
<?php
class Product extends CI_Controller{
    function __construct(){
        parent::__construct();
        $this->load->model('product_model');
    }
    function index(){
        $data['product'] = $this->product_model->get_product();
        $this->load->view('product_view',$data);
    }
    function add_new(){
        $this->load->view('add_product_view');
    }
    function save(){
        $product_name = $this->input->post('product_name');
        $product_price = $this->input->post('product_price');
        $this->product_model->save($product_name,$product_price);
        redirect('product');
    }
    function delete(){
        $product_id = $this->uri->segment(3);
        $this->product_model->delete($product_id);
        redirect('product');
    }
    function get_edit(){
        $product_id = $this->uri->segment(3);
        $result = $this->product_model->get_product_id($product_id);
        if($result->num_rows() > 0){
            $i = $result->row_array();
```

```

        $data = array(
            'product_id'      => $i['product_id'],
            'product_name'    => $i['product_name'],
            'product_price' => $i['product_price']
        );
        $this->load->view('edit_product_view',$data);
    }else{
        echo "Data Was Not Found";
    }
}

function update(){
    $product_id = $this->input->post('product_id');
    $product_name = $this->input->post('product_name');
    $product_price = $this->input->post('product_price');
    $this->product_model-
>update($product_id,$product_name,$product_price);
    redirect('product');
}
}

```

#6. Tambahkan sebuah **function update** pada model **Product_model.php**.

Adapun kodenya sebagai berikut:

```

function update($product_id,$product_name,$product_price){
    $data = array(
        'product_name' => $product_name,
        'product_price' => $product_price
    );
    $this->db->where('product_id', $product_id);
    $this->db->update('product', $data);
}

```

Sehingga terlihat kode lengkap dari model **Product_model.php** seperti berikut:

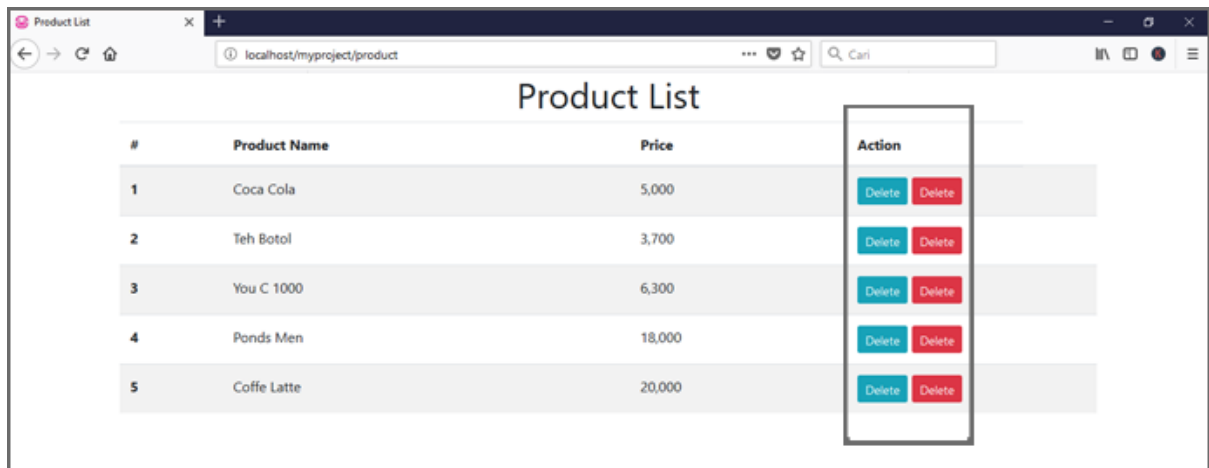
```
<?php
class Product_model extends CI_Model{

    function get_product(){
        $result = $this->db->get('product');
        return $result;
    }
    function save($product_name,$product_price){
        $data = array(
            'product_name' => $product_name,
            'product_price' => $product_price
        );
        $this->db->insert('product',$data);
    }
    function delete($product_id){
        $this->db->where('product_id', $product_id);
        $this->db->delete('product');
    }
    function get_product_id($product_id){
        $query = $this->db->get_where('product', array('product_id' =>
$product_id));
        return $query;
    }
    function update($product_id,$product_name,$product_price){
        $data = array(
            'product_name' => $product_name,
            'product_price' => $product_price
        );
        $this->db->where('product_id', $product_id);
        $this->db->update('product', $data);
    }
}
```

Sekarang kembali ke browser dan kunjungi url berikut:

<http://localhost/myproject/product>

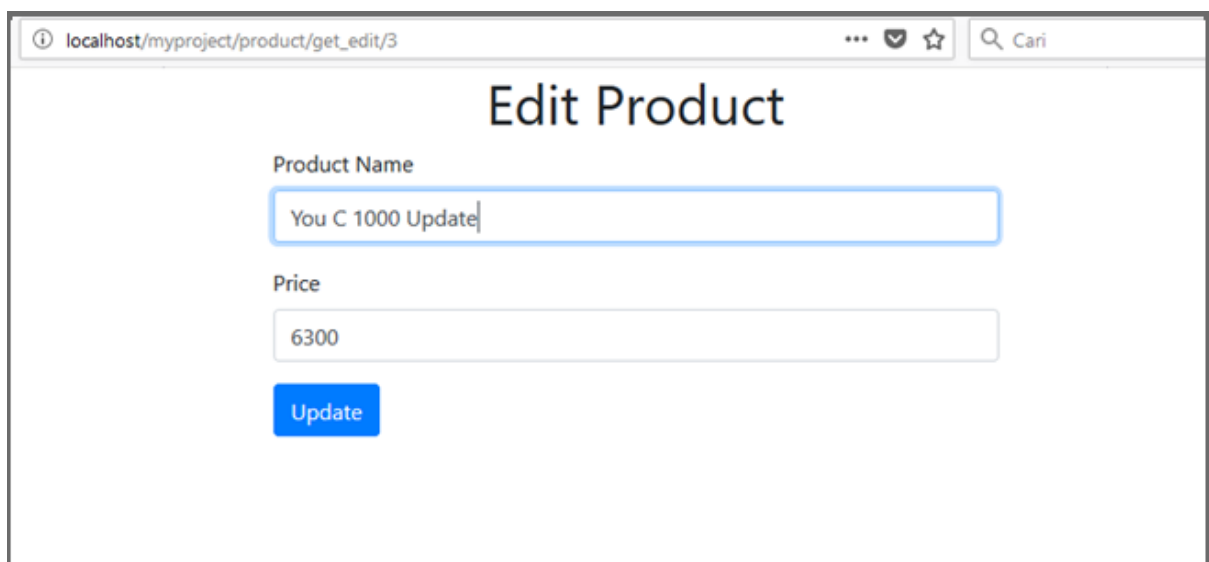
Maka akan tampil product list seperti gambar berikut:



#	Product Name	Price	Action
1	Coca Cola	5,000	Delete Delete
2	Teh Botol	3,700	Delete Delete
3	You C 1000	6,300	Delete Delete
4	Ponds Men	18,000	Delete Delete
5	Coffe Latte	20,000	Delete Delete

Silahkan klik satu dari tombol **edit** pada kolom **action** untuk mengupdate record.

Maka akan muncul form editnya seperti berikut:



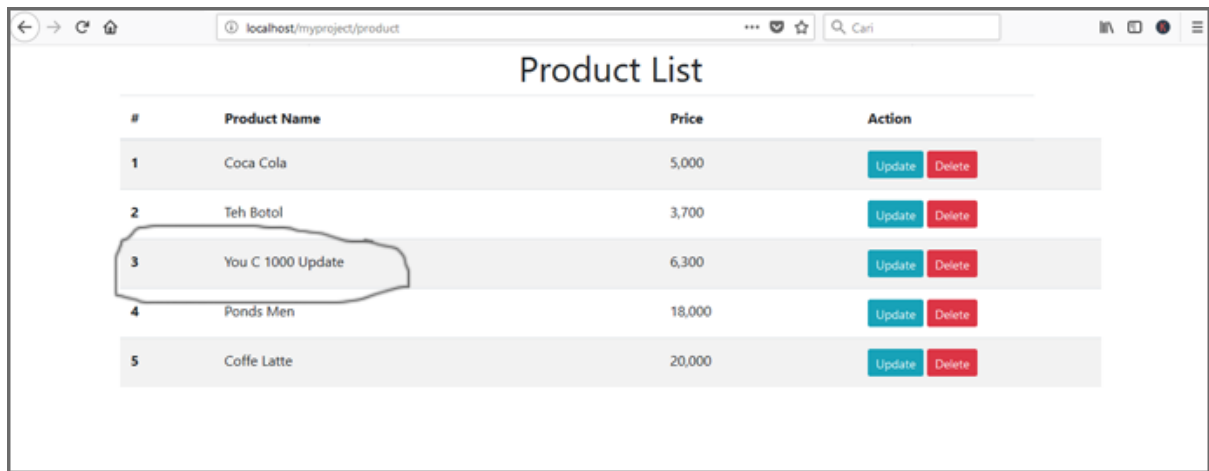
Edit Product

Product Name

Price

Update

Klik tombol update, maka record akan terupdate.



#	Product Name	Price	Action
1	Coca Cola	5,000	Update Delete
2	Teh Botol	3,700	Update Delete
3	You C 1000 Update	6,300	Update Delete
4	Ponds Men	18,000	Update Delete
5	Coffe Latte	20,000	Update Delete

Selesai.

KESIMPULAN

Pembahasan kali ini adalah tentang tutorial lengkap codeigniter untuk pemula.

Mulai dari pengenalan codeigniter, keunggulan codeigniter, konsep MVC, instalasi codeigniter, serta konfigurasi dasar pada codeigniter.

Anda juga telah mempelajari bagaimana sebuah controller bekerja, dengan kasus hello world codeigniter.

Anda juga telah belajar bagaimana menghilangkan index.php pada url, sehingga URL menjadi lebih rapi sekaligus SEO Friendly.

Anda juga telah mempelajari bagaimana menampilkan sebuah view melalui controller.

Anda juga telah mempelajari bagaimana menghubungkan codeigniter dengan bootstrap.

Terakhir, anda telah mempelajari bagaimana bekerja dengan database.

Dan Anda juga telah mempelajari bagaimana membuat aplikasi CRUD (Create Read Update Delete) dengan codeigniter dan bootstrap.

Jadi, tunggu apa lagi...Let's coding.!

Source: <http://mfikri.com/artikel/tutorial-codeigniter>