

CodeIgniter Framework



IT507 - Rekayasa Web

CodeIgniter (CI) adalah suatu framework pengembangan aplikasi web berbasis PHP. CI menyediakan banyak library sehingga memungkinkan mengembangkan aplikasi dengan lebih cepat. Salah satu keunggulan CI dibanding framework lain adalah kesederhanaan penggunaannya dan kecepatan eksekusinya. Pada modul 1 ini akan mengenalkan pembaca pada CI serta dasar pengembangan aplikasi yang menggunakan framework CI.





CodeIgniter Framework

Modul 1

| | |
|----------------------|---|
| Materi | Pengenalan CI, Alur proses data, MVC di CI, Konfigurasi awal, Latihan membuat form input data menggunakan helper. |
| Target | Mengerti struktur CI, Mengerti alur proses data di CI mulai dari request sampai response, Form input data. |
| Pre requisite | Mengerti Object Oriented Programming di PHP. |



A. Berkenalan dengan CodeIgniter

1. Apa itu CodeIgniter?

CodeIgniter (CI) adalah sebuah kerangka (framework) pembangunan aplikasi atau mudahnya disebut *toolkit*, untuk developer yang akan membuat aplikasi web dengan PHP. Tujuan CI adalah supaya pembangunan aplikasi lebih cepat dibanding menulis *source code* dari awal, karena CI telah menyediakan banyak *library* untuk proses-proses yang sering digunakan pada suatu aplikasi, dan juga dengan kemudahan dalam menggunakan *library* tersebut serta kesederhaan penggunaannya.

CodeIgniter ditulis (dibuat) oleh [Rick Ellis](#), seorang musisi rock yang menjadi programmer. Ia membangun perusahaan bernama [Ellis Lab](#), yang mengembangkan beberapa produk unggulan salah satunya CodeIgniter.

CodeIgniter cocok untuk developer yang:

- Menginginkan framework yang sederhana.
- Mebutuhkan kinerja yang luar biasa.
- Mebutuhkan kompatibilitas yang luas dengan berbagai web hosting.
- Menginginkan framework yang hampir tidak ada konfigurasi.
- Menginginkan framework yang tidak menggunakan *command line*.
- Menginginkan framework yang tidak mengharuskan mematuhi aturan penulisan *source code*.
- Tidak ingin dipaksa harus mempelajari *templating language*.
- Tidak menyukai kompleksitas, lebih menyukai solusi yang sederhana.
- Mebutuhkan dokumentasi yang baik.

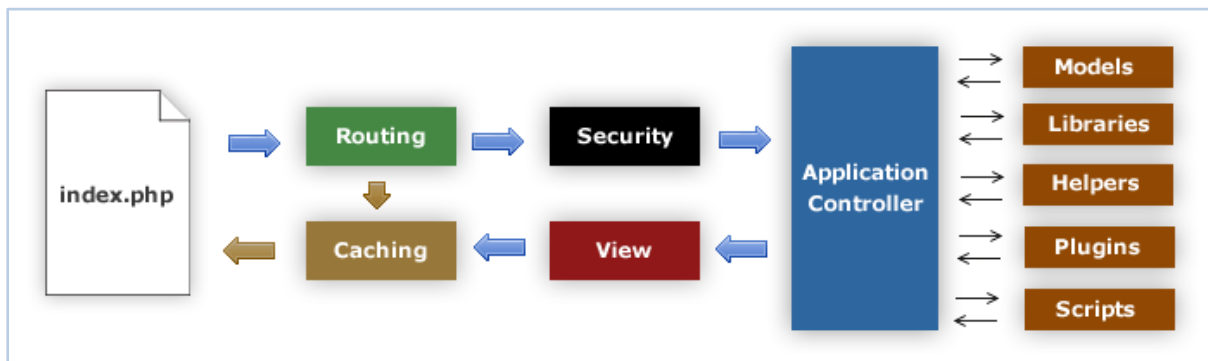
Keunggulan CodeIgniter:

- Gratis.
- Ringan.
Inti sistem CI hanya membutuhkan sangat sedikit *library*, hal ini merupakan perbedaan kontras dengan framework lain. *Library* lainnya dapat digunakan dinamis berdasarkan kebutuhan.
- Cepat.
Sampai saat ini CI masih diakui sebagai framework yang paling cepat.
- Menggunakan Konsep MVC.
- Clean URL.
URL yang digunakan CI bersifat *search-engine friendly*. Menggunakan pendekatan *segment-based*.

[example.com/news/article/345](#)
- Library* yang lengkap.
- Dapat diperluas.
Developer dapat dengan mudah mengembangkan (*extend*) *library*, *helper* atau bahkan perluasan class inti CI.
- Dukungan teknis yang lengkap di forum CI.

2. Alur Proses Aplikasi

Gambar berikut ini mengilustrasikan alur proses data pada CI:



- Index.php** berfungsi sebagai pengendali awal, menginisialisasi sumber daya utama yang dibutuhkan CodeIgniter.
- Router** memeriksa paket HTTP *request* untuk menentukan aksi apa yang harus dilakukan oleh sistem.
- Jika **cache** tersedia, maka halaman langsung dikirim ke browser, eksekusi sistem yang normal akan dilewati.
- Security**. Sebelum *Application Controller* dieksekusi, paket HTTP *request* dan semua data yang dikirimkan pengguna akan disaring terlebih dahulu oleh *Security Class*.
- Application Controller** menginisialisasi *model*, *library* utama, *helpers* dan semua sumberdaya yang dibutuhkan untuk setiap *request*.
- Antarmuka aplikasi (*view*) yang sudah disiapkan dikirimkan ke browser. Jika *caching* diaktifkan, maka *view* akan disimpan sementara untuk request yang sama berikutnya.

3. Model, View, Controller, Libraries, Helper

Seperti framework PHP pada umumnya, CodeIgniter menggunakan konsep MVC serta menyediakan banyak *library* dan *helper* untuk digunakan. Berikut penjelasan mengenai model, view, controller, library dan helper.

- Model**, merepresentasikan struktur data. Biasanya class model akan berisi fungsi-fungsi untuk mengambil data, insert data, dan update data ke database. Pada CI, model tidak harus digunakan, tapi hal ini akan menghilangkan konsep MVC itu sendiri.
- View**, adalah informasi / halaman yang ditampilkan ke pengguna. Sebuah *view* biasanya adalah sebuah *web page*, tapi di CodeIgniter *view* juga dapat berupa bagian-bagian halaman web, seperti *header* dan *footer*. Bahkan *view* juga dapat berupa halaman RSS.
- Controller**, berfungsi sebagai penghubung antara Model, View dan dengan sumber daya lain yang digunakan untuk memproses HTTP *request*. Controller juga biasanya berfungsi sebagai inti pemrosesan logik aplikasi.
- Libraries**, adalah macam-macam class yang masing-masing mempunyai fungsi khusus yang dapat digunakan untuk mengembangkan aplikasi. Contoh library database, email, validasi form, dan lain-lain.

-
- **Helper**, seperti namanya berfungsi menolong untuk melakukan tugas-tugas tertentu. Setiap file *helper* terdiri dari kumpulan fungsi (function). Contoh URL Helper yang berfungsi untuk membuat link, Form helper untuk membuat elemen-elemen form. Tidak seperti *library*, helper tidak menggunakan format Object Oriented, sehingga dapat digunakan dimanapun, baik itu di model, view, controller dan library.

4. CodeIgniter URL

Secara *default*, URL pada CodeIgniter didesain agar *search-engine* dan *human-friendly*, menggunakan pendekatan **segment-based**.

```
example.com/index.php/class/function/parameter1/parameter2
```

- a. **Index.php** merupakan segment ke-0.
- b. Segment pertama merepresentasikan **class controller** yang diakses.
- c. Segment kedua merepresentasikan **nama method** yang dipanggil pada class tersebut.
- d. Segment ketiga dan seterusnya bersifat *optional*, merepresentasikan **parameter masukan** untuk fungsi yang dipanggil tersebut.

Index.php dapat dihilangkan dengan .htaccess sederhana. Jika ingin menghilangkan index.php, sebaiknya lakukan diawal sebelum *source-code* aplikasi dibuat, karena jika aplikasi sudah jadi dan index.php dihilangkan, dapat merubah struktur *link* seluruh aplikasi, contohnya menu, dan *paging*.

Sepertinya cukup basa-basinya, mari kita mulai mainkan CodeIgniter...

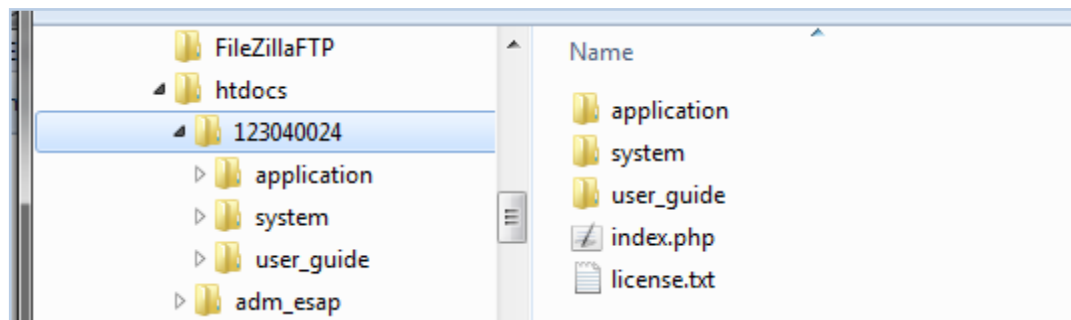
B. Memulai Menggunakan CodeIgniter

1. Download CI

CodeIgniter dapat di-*download* gratis di website resminya yaitu www.codeigniter.com. Pada kuliah kali ini silahkan *download* di modul online.

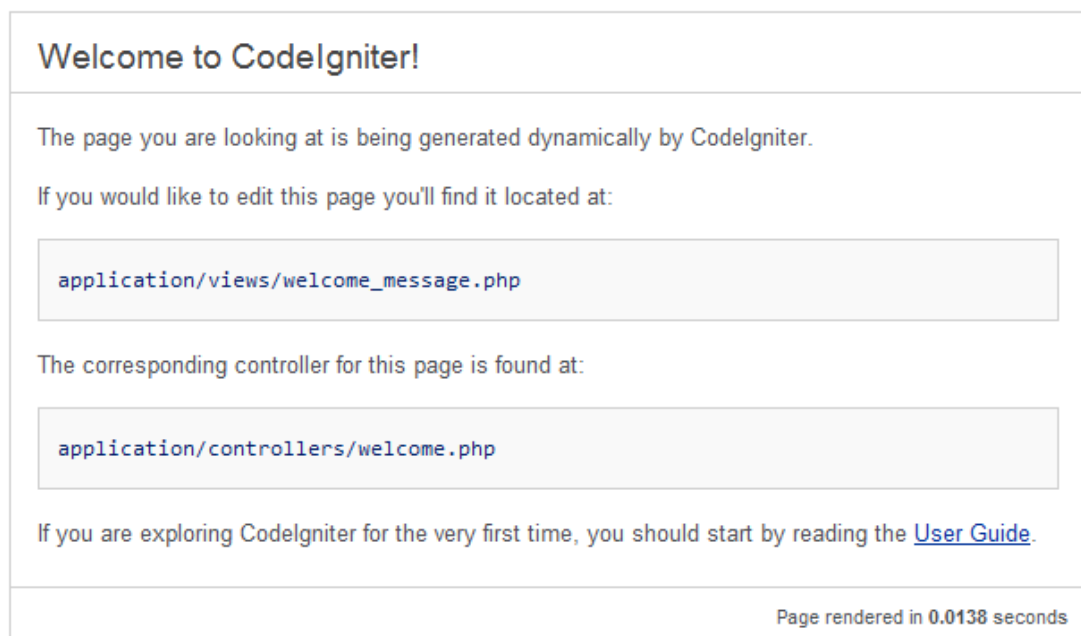
2. Instalasi

Extract file CodeIgniter yang sudah di-*download* sebelumnya ke direktori htdocs, kemudian ganti nama folder **CodeIgniter_2.x.x** tersebut dengan **NRP Anda** yang selanjutnya akan menjadi direktori kerja Anda, sehingga seperti ini:



Gambar B.2.1. Direktori Instalasi.

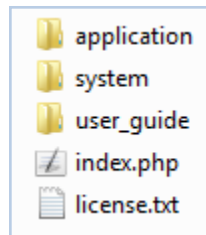
Untuk menguji apakah instalasi sudah benar dan membuktikan salah satu keunggulan CI (hampir tidak ada konfigurasi awal), silahkan akses direktori kerja Anda pada browser. Contoh: <http://localhost/123040024>. Jika sukses maka akan tampil halaman pembuka CodeIgniter seperti di bawah ini:



Gambar B.2.2. Halaman Pembuka CodeIgniter.

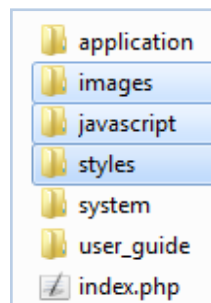
3. Struktur Folder

Berikut akan dibahas fungsi masing-masing folder dan file default dari CodeIgniter. Dimulai dari root direktori kerja Anda.

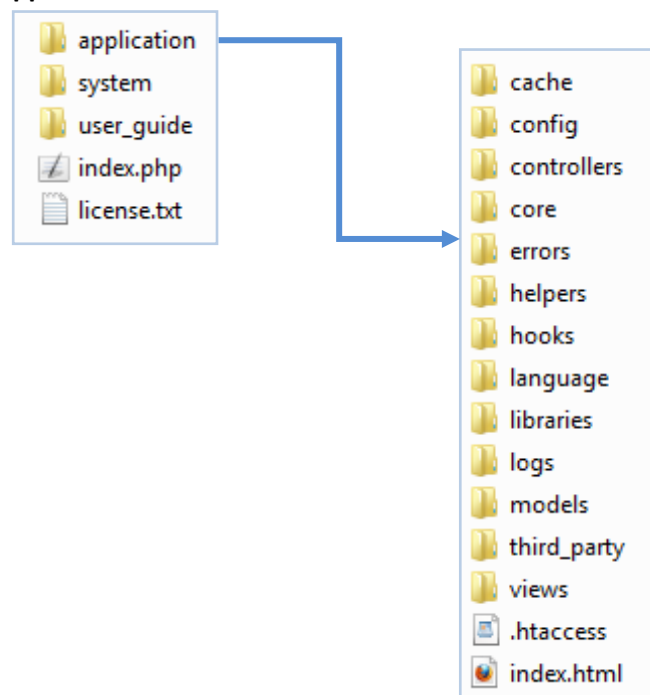


- a. **Application**, berisi folder dan file yang terkait dengan aplikasi yang akan dibuat. Kita akan banyak bekerja pada direktori **application** ini.
- b. **System**, berisi file-file inti framework CodeIgniter. Kita tidak akan mengubah apapun pada direktori **system** ini.
- c. **User_guide**, berisi file-file dokumentasi CI yang dapat diakses melalui browser.
- d. **Index.php**, file konfigurasi awal CodeIgniter.

Aplikasi yang akan Anda bangun tentunya menggunakan JavaScript, gambar dan CSS. Anda dapat buat direktori untuk masing-masing sumberdaya tersebut pada root aplikasi (setingkat dengan file index.php), sehingga seperti contoh berikut:



3.1. Direktori Application



Berikut penjelasan direktori-direktori utama yang akan sering kita gunakan.

- a. **Config**, berisi file-file yang terkait konfigurasi dasar aplikasi.
- b. **Controllers**, berisi file-file controller dari aplikasi yang dibangun. Simpan seluruh file controller aplikasi pada folder ini.
- c. **Core**, jika ingin memperluas class-class inti CodeIgniter, maka simpan file class yang Anda kembangkan tersebut pada direktori ini.
- d. **Helpers**, berisi file-file *helper* yang Anda buat sendiri.
- e. **Libraries**, berisi file-file *library* yang Anda buat sendiri.
- f. **Logs**, jika error log pada config diaktifkan maka file error log akan tersimpan pada direktori ini.
- g. **Models**, berisi file-file model dari aplikasi yang dibangun. Simpan seluruh file model aplikasi pada folder ini.
- h. **Views**, berisi file-file view dari aplikasi yang dibangun. Simpan seluruh file view aplikasi pada folder ini.

4. Apa Saja yang Harus Dikonfigurasi untuk Mulai Membangun Aplikasi?

Pada setiap pembangunan aplikasi tentunya diperlukan beberapa konfigurasi tambahan yang mendukung aplikasi tersebut, misalnya konfigurasi error_reporting, database, security, dan controller default aplikasi. Berikut akan dijelaskan beberapa file konfigurasi minimal untuk mulai membangun aplikasi (silahkan buka masing-masing file yang ada di penjelasan berikut ini):

- a. **Index.php**. Sebenarnya pada tahap development tidak ada konfigurasi yang diubah pada file ini, tapi jika aplikasi akan di-*publish* / *deploy* maka error reporting harus dimatikan, silahkan lihat di baris 21 ada konfigurasi ENVIRONMENT. *Default*-nya terdapat 3 nilai konstanta ENVIRONMENT, yaitu **development**, **testing**, dan **production**. Setiap nilai tersebut berpengaruh pada error_reporting aplikasi. Selain ENVIRONMENT, pada file index.php juga terdapat beberapa konfigurasi lain seperti nama direktori **application** dan **system**, ekstensi file aplikasi yang bisa diubah sesuai kebutuhan.

Selanjutnya buka direktori **application/config/**

- b. **Autoload.php**, pada file ini didefinisikan apa saja yang harus dipanggil otomatis oleh CI tanpa harus dideklarasikan di setiap fitur aplikasi. Yang dapat otomatis dipanggil adalah:
 - 1). Packages
 - 2). Libraries
 - 3). Helper
 - 4). Custom config, jika Anda membuat file konfigurasi diluar file default CodeIgniter.
 - 5). Language
 - 6). Model.

Kita dapat mendefinisikan apa saja yang harus dipanggil otomatis oleh CI sesuai kebutuhan, misalnya library **session**, **database**, **security**, helper **form** dan **url**.

-
- c. **Config.php**, pada file ini didefinisikan macam-macam konfigurasi utama aplikasi. Diantaranya yang biasanya kita ubah pada setiap project adalah:
- 1). **Base url**, berfungsi sebagai URL utama aplikasi yang dibangun. Jika tidak didefinisikan CI akan mendeteksi otomatis URL dan Domain yang dipakai aplikasi. Pada modul 1 ini silahkan isi dengan URL kerja Anda atau biarkan kosong tidak masalah.
 - 2). **Index page**, jika file index diganti namanya maka sesuaikan konfigurasi ini, jika index.php dihilangkan dari URL menggunakan .htaccess maka kosongkan nilai `$config['index_page']`.
Pada modul 1 ini tetap index.php saja.
 - 3). **Url suffix**, jika ingin menambahkan akhiran ekstensi file di URL. Misal `example.com/news/show/231.html`
Pada modul 1 ini Anda silahkan menentukan sendiri url_suffix yang diinginkan.
 - 4). **Language**, bahasa yang akan digunakan pada aplikasi. Kita dapat merubah atau membuat bahasa sendiri untuk sistem CI pada direktori root: `system/language/`.
Pada modul 1 ini tetap 'english' saja.
 - 5). **Encryption key**, jika aplikasi menggunakan **session**, maka `$config['encryption_key']` harus didefinisikan. Pada modul 1 ini kita tidak menggunakan session.
 - 6). **Session**, konfigurasi untuk session:
 - 6).1. `['sess_cookie_name']` : nama cookie aplikasi yang dibangun, bisa diubah.
 - 6).2. `['sess_expiration']` : waktu kadaluarsa session, satuannya detik.
 - 6).3. `['sess_expire_on_close']` : jika bernilai TRUE, maka session *expire* ketika browser ditutup.
 - 6).4. `['sess_encrypt_cookie']` : beri nilai TRUE jika cookie ingin dienkrpsi.
 - 6).5. `['sess_use_database']` : beri nilai TRUE jika ingin menggunakan database untuk menyimpan data session.
 - 6).6. `['sess_table_name']` : nama table penyimpanan data session, bisa diubah.
 - 6).7. `['sess_match_ip']` : beri nilai TRUE jika ingin membanding IP address setiap membaca data session.
 - 6).8. `['sess_match_useragent']` : beri nilai TRUE untuk membandingkan browser pengguna setiap membaca data session.
 - 6).9. `['sess_time_to_update']` : waktu (dalam detik) CI harus memperbaharui data session yang disimpannya.
Pada modul 1 ini tidak ada konfigurasi session yang diubah.
 - 7). **XSS Filtering**, mendefinisikan apakah setiap data yang dikirimkan oleh browser / pengguna (POST, GET, COOKIE) diperiksa dulu dari bahaya *hacking* metode XSS (Cross Site Scripting).
Sebaiknya nilai `$config['global_xss_filtering']` bernilai **TRUE**.

-
- 8). CSRF Protection**, mendefinisikan apakah setiap data yang dikirimkan melalui form masih masuk waktu SESSION atau tidak dan memastikan bahwa data dikirimkan oleh browser yang mengakses halaman web tersebut, jika tidak maka data akan ditolak. Konfigurasi ini untuk menghindari bahaya *hacking* metode Cross Site Request Forgery.
Sebaiknya nilai `$config['csrf_protection']` bernilai **TRUE**.
- d. **Database.php**, pada file ini mendefinisikan parameter-parameter yang dibutuhkan untuk koneksi ke database. CodeIgniter mendukung koneksi ke beberapa produk DBMS, bahkan mendukung koneksi ke beberapa database dan DBMS sekaligus pada satu aplikasi. Berikut penjelasan beberapa konfigurasi:
- 1). Active group**, isikan nama konfigurasi database default yang digunakan.
 - 2). Active record**, jika bernilai TRUE, maka fasilitas Active Records CI dapat digunakan. Active Record membantu membuat query yang akan *support* multi DBMS.
 - 3). Hostname**, isikan alamat IP atau hostname database server.
 - 4). Username**, isikan username yang dipakai untuk login ke database.
 - 5). Password**, isikan password yang dipakai untuk login ke database.
 - 6). Database**, isikan nama database yang dipakai aplikasi.
 - 7). DB Driver**, isikan tipe driver DBMS yang dipakai aplikasi. Driver DBMS yang terkoneksi dengan CI saat ini : mysql, mysqli, postgres, odbc, mssql, sqlite, oci8.
 - 8). P Connect**, isikan nilai TRUE jika ingin koneksi ke database server tetap terhubung untuk beberapa waktu sejak eksekusi query terakhir (Persistent Connection). Cukup membantu agar proses aplikasi lebih cepat, dibandingkan harus menjalin koneksi ke database setiap akan eksekusi query.
 - 9). DB Debug**, isikan nilai TRUE jika ingin error koneksi atau error query ditampilkan. Sangat membantu pada proses development.
Jika aplikasi akan di-*publish* / *deploy* maka nilai `['db_debug']` sebaiknya **FALSE**.
- e. **Routes.php**, pada file ini didefinisikan beberapa URL yang spesifik mengacu ke controller yang spesifik pula sesuai kebutuhan aplikasi. Pada file ini juga dapat didefinisikan default controller yang akan dipanggil jika tidak didefinisikan pada URL. Itulah mengapa tanpa konfigurasi awal halaman pembuka CI bisa tampil seperti di Gambar B.2.2, karena `$route['default_controller']` diisi controller "*welcome*".
Silahkan buka file `welcome.php` di folder `controllers` untuk lebih jelasnya.

5. Akses User guide

Buku pedoman (user guide) CodeIgniter dapat diakses pada alamat:

- http://www.codeigniter.com/user_guide
- Direktori kerja Anda, misalnya http://localhost/123040024/user_guide

Untuk penjelasan lebih detail mengenai seluk-beluk CodeIgniter silahkan akses User Guide-nya yang rapi dan mudah dipahami.

Supaya tangannya gak kesemutan, mendingan kita berdayakan jari-jari kita yuk...

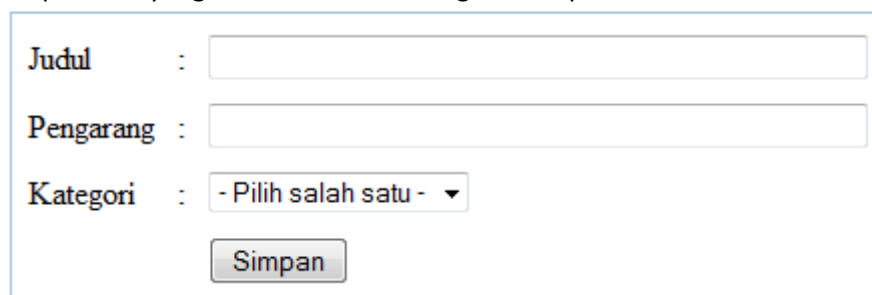
C. Latihan

Studi kasus latihan di modul 1 dan seterusnya adalah mengenai fitur pengelolaan data Buku. Semoga tidak bosan dengan studi kasus ini ya. Diketahui atribut data buku sebagai berikut:

| Field | Tipe Data | Ukuran |
|-----------|-----------|-------------------|
| ID_Buku | Integer | 8, auto increment |
| Judul | Varchar | 100 |
| Pengarang | Varchar | 150 |
| Kategori | Varchar | 10 |

Pada modul 1 ini, kita akan membuat form untuk input data buku sesuai atribut di atas. Pada modul 2 minggu depan, kita akan buat tabel database sesuai atribut di atas lalu koneksikan CodeIgniter dengan database tersebut.

Form input data yang akan kita buat kurang lebih seperti ini :

A screenshot of a web form for adding a new book. It contains three input fields: 'Judul' (Title), 'Pengarang' (Author), and 'Kategori' (Category). The 'Kategori' field is a dropdown menu with the text '- Pilih salah satu -' and a downward arrow. Below the fields is a 'Simpan' (Save) button.

Judul :

Pengarang :

Kategori :

Gambar C.0. Form Input Data Buku.

Selanjutnya ikuti langkah-langkah berikut :

1. Buat controller Buku.

Buatlah sebuah file menggunakan editor PHP, ketikkan script di bawah ini:

```
1 <?php
2
3 class Buku extends CI_Controller
4 {
5     var $data = array();
6
7     function __construct()
8     {
9         parent :: __construct();
10
11         $this->load->helper('form');
12
13         $this->data['opt_kategori'] = array('' => '- Pilih salah satu -',
14                                           'novel' => 'Novel',
15                                           'komik' => 'Komik',
16                                           'kamus' => 'Kamus');
17     }
18
19     function index()
20     {
21         $this->add_new();
22     }
23
24     function add_new()
25     {
26         $this->data['is_update'] = 0;
27
28         $this->load->view('buku_form_v', $this->data);
29     }
30 }
31
32 ?>
```

Gambar C.1. Controller Buku.

Simpan file tersebut di direktori kerja Anda, pada folder: **application/controllers/**

Beri nama file : **buku.php**

Penjelasan:

```
class Buku extends CI_Controller
```

- Nama class harus diawali huruf kapital.
- Semua controller yang kita buat harus turunan dari controller CI (**CI_Controller**)
- Lebih detail mengenai controller : user_guide/general/controllers.html

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

- Membuat konstruktor class Buku yang merupakan overriding dari konstruktor class CI_Controller.
- Maksud dibuat konstruktor agar saat class dipanggil, kita dapat menyisipkan beberapa script yang otomatis dieksekusi.

```
$this->load->helper('form');
```

- Memanggil helper **form**, agar dapat digunakan di view.
- Lebih detail mengenai helper form : user_guide/helpers/form_helper.html

```
$this->data['opt_kategori'] = array('' => '- Pilih salah satu -',
                                   'novel' => 'Novel',
                                   'komik' => 'Komik',
                                   'kamus' => 'Kamus');
```

- Untuk mengirimkan variabel ke view, harus variabel bertipe array.
- **opt_kategori** dimaksudkan untuk nilai *option* pada dropdown di view. Lihat point C.2. berikutnya untuk lebih jelas.

```
function index()
{
    $this->add_new();
}
```

- Jika pada class ada method bernama **index** dan pada URL tidak didefinisikan method apa yang dipanggil, maka method index yang otomatis dipanggil oleh CI.

```
function add_new()
{
    $this->data['is_update'] = 0;
```

- Method **add_new** untuk memanggil form input data buku.
- **\$this->data['is_update'] = 0;**
Dimaksudkan untuk mengirim variabel **is_update** yang bernilai **0** ke view. Berarti aksi yang akan dilakukan bukan update data.

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

- Memanggil view bernama **buku_form_v**, yang merupakan form input data buku.
- Parameter ke-2, berarti mengirimkan variabel **\$this->data** ke view.
- Karena variabel **\$this->data** bertipe array, maka **index** dari array tersebut akan menjadi nama variabel di view.

2. Buat view Form Input Data

Buat file baru, kemudian ketikkan script di bawah ini:

```
1 <?php
2
3     echo form_open('buku/save/'.$sis_update);
4
5     echo form_hidden('id','');
6
7     echo "Judul : ".form_input('judul','', "size='50' maxlength='100'");
8     echo "<br /><br />";
9
10    echo "Pengarang : ".form_input('pengarang','', "size='50' maxlength='150'");
11    echo "<br /><br />";
12
13    echo "Kategori : ".form_dropdown('kategori',$opt_kategori,'');
14    echo "<br /><br />";
15
16    echo form_submit('btn_simpan','Simpan');
17
18    echo form_close();
19
20 ?>
```

Gambar C.2. Script Form Input Data Buku.

Simpan file tersebut di direktori kerja Anda, pada folder: **application/views/**
Beri nama file : **buku_form_v.php**

Penjelasan:

```
echo form_open('buku/save/'.$sis_update);
```

- Membuat elemen form.
- Script tersebut sama dengan:

```
<form action="buku/save/0" method="POST">
```

```
echo form_hidden('id','');
```

- Membuat elemen **hidden** bernama **id**.
- Parameter pertama adalah nama elemen, parameter kedua adalah default value.
- Script tersebut sama dengan:

```
<input type="hidden" name="id" value="" />
```

```
echo "Judul : ".form_input('judul','', "size='50' maxlength='100'");
```

- Membuat elemen input textbox bernama judul.
- Parameter pertama adalah nama elemen, kedua adalah default value, ketiga adalah atribut tambahan.
- Script tersebut sama dengan:

```
Judul : <input type="text" name="judul" size="50" maxlength="100" value="" />
```

```
echo "Kategori : ".form_dropdown('kategori',$opt_kategori,'');
```

- Membuat elemen combobox / dropdown list.
- Parameter pertama adalah nama elemen, kedua adalah opsi yang tersedia untuk combobox tersebut, ketiga adalah default value, keempat adalah atribut tambahan.
- Script tersebut sama dengan:

```
Kategori : <select name="kategori">
    <option selected="selected">- Pilih salah satu -</option>
    <option value="novel">Novel</option>
    <option value="komik">Komik</option>
    <option value="kamus">Kamus</option>
</select>
```

```
echo form_submit('btn_simpan','Simpan');
```

- Membuat button submit bernama btn_simpan dan berteks Simpan.
- Parameter pertama adalah nama elemen, kedua adalah teks button, ketiga adalah atribut tambahan.
- Script tersebut sama dengan:

```
<input type="submit" name="btn_submit" value="Simpan" />
```

```
echo form_close();
```

- Membuat tag tutup form.

3. Akses Form Input Data buku pada browser

Akses form input data buku pada browser dengan URL sesuai direktori kerja Anda.

Contoh :

<http://localhost/123040024/index.php/buku/>

atau

http://localhost/123040024/index.php/buku/add_new

Jika sukses maka akan tampil form seperti Gambar C.0.

4. Rubah konfigurasi Routes

Agar form input data buku otomatis tampil jika pada URL tidak didefinisikan controller apa yang diakses, maka ubah konfigurasi di file **routes.php**.

Lihat point B.4.e. mengenai konfigurasi routes. Ganti default controller ke controller Buku, sehingga seperti ini :

```
$route['default_controller'] = "buku/add_new";
```

Kemudian akses dengan alamat : <http://localhost/123040024/>

Jika sukses maka hasilnya akan sama dengan point C.3 di atas



CodeIgniter Framework
Modul 1
Selesai