Materi 2

E-Commerce Lanjut (CodeIgniter)

3 SKS | Semester 7 | S1 Sistem Informasi | UNIKOM | 2015
Nizar Rabbi Radliya | nizar.radliya@yahoo.com

Nama Mahasiswa	
NIM	
Kelas	

Kompetensi Dasar

Memahami konsep dasar MVC (*Model View Controller*), instalasi, struktur, alur kerja, serta dasar routing pada CodeIgniter.

Pokok Bahasan

Pengenalan Framework CodeIgniter:
1. Konsep dasar MVC (*Model View Controller*)
2. Pengantar CodeIgniter
3. Instalasi CodeIgniter
4. Struktur CodeIgniter
5. Alur kerja CodeIgniter
6. Dasar routing CodeIgniter

I. Konsep Dasar MVC (*Model View Controller*)

Sebelum kita membahas lebih jauh mengenai framework PHP, sebaiknya terlebih dahulu kita memahami konsep dasar MVC (*Model View Controller*). Pada awalnya konsep MVC digunakan untuk bahasa pemrograman Smalltalk. Seiring dengan perkembangan teknologi, konsep MVC mulai diterapkan dalam framework pemrogaman baik untuk aplikasi berbasis desktop, web, maupun mobile.

Konsep dasar MVC adalah pemisahan antara logika aplikasi dengan tampilan. MVC merupakan konsep sistem aplikasi yang dibangun oleh tiga komponen utama yang terpisah, dimana ketiga komponen tersebut adalah manipulasi data (model), antarmuka pengguna (view), dan kontrol aplikasi (controller). Meskipun ketiganya dikembangkan/dibangun secara terpisah, akan tetapi ketiga komponen tersebut memiliki keterkaitan pada setiap prosesnya. Berikut adalah penjelasan dari 3 komponen MVC:

1. Model, merupakan bagian yang berhubungan langsung dengan database untuk memanipulasi data (*insert, update, delete, select*), menangani validasi dari bagian controller, namun tidak dapat berhubungan langsung dengan bagian view.
2. View, merupakan bagian yang menangani *presentation logic* (antarmuka pengguna). Pada suatu aplikasi web bagian ini biasanya berupa file template HTML, yang diatur

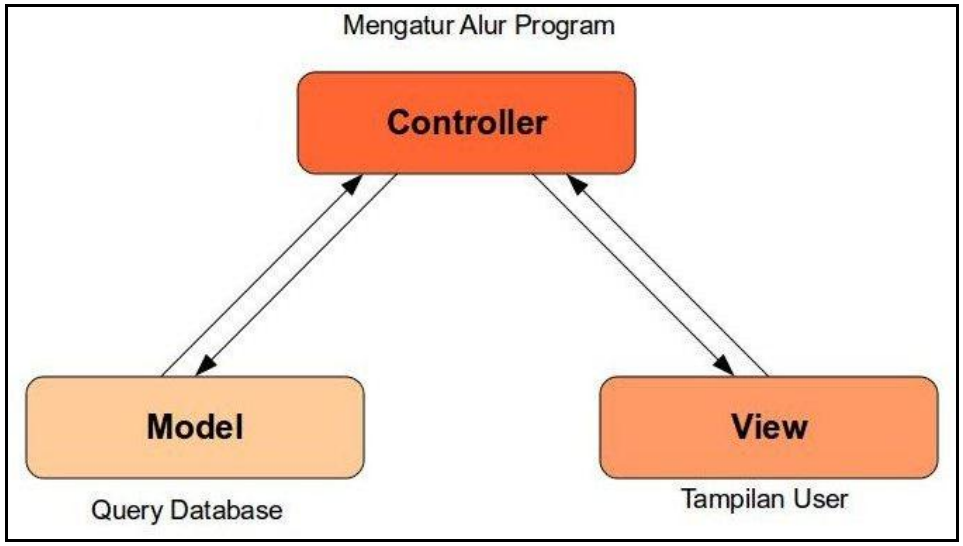
oleh controller. View berfungsi untuk menerima dan merepresentasikan data kepada user. Bagian ini tidak memiliki akses langsung terhadap bagian model.

3. Controller, merupakan bagian yang mengatur hubungan antara bagian model dan bagian view, controller berfungsi untuk menerima request dan data dari user kemudian menentukan apa yang akan diproses oleh aplikasi.

Singkat kata Model untuk mengatur alur database, View untuk menampilkan antarmuka web, sedangkan Controller untuk mengatur alur kerja antara Model dan View. Sebagai contoh cara kerja MVC adalah ketika kita akan membuat akun e-mail pada sebuah situs, dimana prosesnya adalah:

1. Pertama kita akan melihat tampilan sign-up, yang dimana itu adalah bagian View.
2. Kemudian kita mamasukan data pada form seperti username, password, dan data lainnya, lalu kita klik tombol sign-up maka di sini View memanggil Controller.
3. Kemudian Controller memanggil Model, sedangkan Model mengecek apakah data yang kita masukan sudah sesuai dengan kriteria pendaftaran seperti sudah mengisi username, password dan lain sebagainya.
4. Kemudian Model menggembalikan (*callback*) ke Controller dan Controller mengembalikan ke View, dan kita akan melihat status berhasil atau tidak proses sign-up yang kita lakukan.

Untuk ilustrasi dari konsep MVC dapat dilihat pada gambar 1 di bawah ini.



Gambar 1. Ilustrasi Konsep MVC

II. Pengantar CodeIgniter

CodeIgniter adalah framework dengan model MVC untuk membangun website dinamis dengan menggunakan bahasa pemrograman PHP. CodeIgniter memudahkan

developer untuk membuat aplikasi web dengan cepat dan mudah dibandingkan dengan membuatnya dari awal. Framework secara sederhana dapat diartikan kumpulan dari fungsi-fungsi/prosedur-prosedur dan class-class untuk tujuan tertentu yang sudah siap digunakan sehingga bisa lebih mempermudah dan mempercepat pekerjaan seorang programer, tanpa harus membuat fungsi atau class dari awal.

Ada beberapa alasan mengapa menggunakan Framework:

1. Mempercepat dan mempermudah pembangunan sebuah aplikasi web.
2. Relatif memudahkan dalam proses *maintenance* karena sudah ada pola tertentu dalam sebuah framework (dengan syarat programmer mengikuti pola standar yang ada).
3. Umumnya framework menyediakan fasilitas-fasilitas yang umum dipakai sehingga kita tidak perlu membangun dari awal (misalnya validasi, ORM, pagination, multiple database, scaffolding, pengaturan session, error handling, dll).
4. Lebih bebas dalam pengembangan jika dibandingkan dengan CMS (*Content Management System*).

Berikut adalah kelebihan yang dimiliki oleh framework CodeIgniter:

1. Performa sangat cepat. Salah satu alasan menggunakan CodeIgniter adalah karena eksekusinya yang sangat cepat bahkan mungkin bisa dibilang CodeIgniter merupakan framework yang paling cepat dibanding framework yang lain.
2. CodeIgniter adalah sebuah framework PHP yang sangat ringan (ukuran file yang kecil dibanding dengan framework PHP lainnya).
3. Konfigurasi yang sangat minim. Tentu saja untuk menyesuaikan dengan database dan keleluasaan routing tetap diizinkan melakukan konfigurasi dengan mengubah beberapa file konfigurasi seperti database.php atau autoload.php, namun untuk menggunakan CodeIgniter dengan setting standard, anda hanya perlu mengubah sedikit saja file pada folder config.
4. Banyak komunitas. Dimana dengan banyaknya komunitas CodeIgniter ini, memudahkan kita untuk berinteraksi dengan yang lain, baik itu bertanya atau melihat teknologi terbaru.
5. Dokumentasi yang sangat lengkap. Setiap paket instalasi CodeIgniter sudah disertai *user guide* yang sangat bagus dan lengkap untuk dijadikan permulaan, bahasanya pun mudah dipahami.

III. Instalasi CodeIgniter

Instalasi framework CodeIgniter (CI) cukup mudah, dengan catatan kita telah melakukan instalasi paket aplikasi web server lokal (contohnya Xampp). Pertama kita download dulu paket CI di situs resminya (URL: https://www.CodeIgniter.com/download). Saat ini, versi terbaru adalah versi CodeIgniter 3.0.1.

Paket CI yang kita download adalah dalam bentuk *.zip, lalu file tersebut kita ekstrak terlebih dahulu dan kemudian kita simpan di dalam direktori xampp/htdocs/ (pada penggunaan Xampp). Nama folder CI dapat kita ubah dengan nama aplikasi web yang akan kita kembangkan.

IV. Struktur CodeIgniter

Pada folder paket CodeIgniter (CI) terdapat 3 folder/direktori yang diantaranya adalah:

1. application

   Direktori ini adalah tempat kita menyimpan atau membuat aplikasi web kita. Di dalam direktori inilah yang nantinya web kita ditampilkan di browser.

2. system

   Direktori ini adalah tempat dimana core atau inti dari CI tersebut yang memuat tentang database, library, helper dan lain-lain. Pada direktori ini kita tidak disarankan merubah code yang ada di dalamnya, kecuali jika kita sudah benar-benar paham tentang CI.

3. user_guide

   Direktori ini adalah tempat dokumentasi CI yang bisa kita akses secara offline melalui web browser.

Selanjutnya kita masuk ke direktori application/ dimana di dalamnya terdapat beberapa folder/direktori diantaranya:

1. cache

   Direktori ini adalah tempat untuk men-caching web yang kita bangun yang berfungsi meningkatkan performa web, dan berimbas pada kecepatan load web.

2. config

   Direktori ini adalah tempat untuk mengatur konfigurasi web, dari mulai konfigurasi database, routing, mime, hook, config, autoload dan lain sebagainya.

3. controllers

   Direktori ini adalah tempat untuk menyimpan controller web untuk mengontrol antara model dan views (sesuai dengan konsep MVC yang diadopsi oleh CI).

4. core

   Direktori ini adalah tempat untuk menyimpan elemen core buatan sendiri yang merupakan perluasan (*extend*) dari core bawaan CI.

5. helpers

   Direktori ini adalah tempat untuk menyimpan elemen helper buatan sendiri maupun perluasan (*extend*) dari bawaan CI. Helper tersebut nantinya berguna baik untuk controller, model dan view.

6. hooks

   Direktori ini adalah tempat di mana kita bisa modifikasi mekanisme kerja dasar sistem CI.

7. language

   Direktori ini adalah tempat untuk penyimpanan berkas-berkas bahasa apabila web yang kita bangun tersedia multibahasa.

8. libraries

   Direktori ini adalah tempat untuk menyimpan elemen library buatan sendiri maupun perluasan (*extend*) dari bawaan CI.

9. logs

   Direktori ini adalah tempat pencatatan log.

10. model

    Direktori ini adalah tempat untuk menyimpan model yang akan dihubungkan dengan controller dan akan diteruskan ke view (sesuai dengan konsep MVC yang diadopsi oleh CI).

11. third_party

    Direktori ini adalah tempat pembuatan pihak ketiga.

12. views

    Direktori ini adalah tempat untuk menyimpan view yang akan ditampilkan di browser sebagai antarmuka web (sesuai dengan konsep MVC yang diadopsi oleh CI).

Selanjutnya kita akan membahas mengenai file index.php yang ada di dalam folder/direktori paket CI (sejajar dengan direktori application, system dan user_guide).

Ada beberapa hal penting yang harus kita pahami pada file index.php tersebut:

1. Application Environment

   Hal ini berkaitan dengan kode:

```
define('ENVIRONMENT', isset($_SERVER['CI_ENV']) ? $_SERVER['CI_ENV']
: 'development');
```

   Pada pengembangan web menggunakan framework CI, terdapat 3 tahapan environment, yaitu:

   a. development

   Kondisi dimana kita mengaktifkan fungsi pengecekan *error* di *apache* (*error_reporting*). Jadi pada tahap *development*, apabila masih terdapat *error* pada kode program yang kita buat, seperti masalah *variable* yang tak terdefinisi, kode yang telah *depricated*, maka akan ditampilkan status kesalahan pada browser.

   b. testing

   Kondisi ini disesuaikan dengan pengaturan default apache. Jika pengaturan *default apache* (*error_reporting*) di aktifkan maka status *error* akan ditampilkan di browser, jika tidak, maka tidak akan ditampilkan.

   c. production

   Kondisi ini adalah sesi dimana jika kode program kita sudah siap diproduksi dan beberapa pengaturan yang menampilkan *error* dinonaktifkan. Biasanya jika ada *error* dalam kode program maka browser hanya akan menampilkan gambar putih tanpa keterangan (*white screen*). Ini berfungsi untuk menutupi beberapa *error* seperti *variable* yang tak terdefinisi, kode yang telah *depricated* dan lain-lain supaya tidak tampil di browser Anda.

2. Folder Name

   Guna menjaga keamanan sistem web, kita dapat mengganti nama **folder/direktori/ application** dan **system**. Jika kita akan mengganti nama direktori application menjadi nama lain, misalnya dengan nama **app_ci**, maka kita ubah nama direktori application dan juga perubahan kode pada bagian:

```
$application_folder = 'application';
```

   Menjadi:

```
$application_folder = 'app_ci';
```

Begitu juga kalau kita mau mengganti nama direktori system menjadi nama lain, misalnya dengan nama **system_ci**, maka kita ubah nama direktori system dan juga perubahan kode pada bagian:

```
$system_path = 'system';
```

Menjadi:

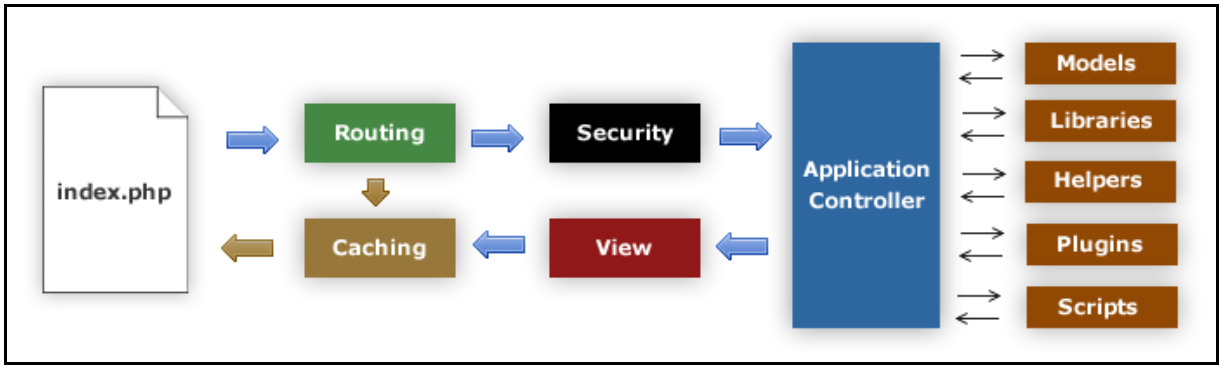
```
$system_path = 'system_ci';
```

Hal tersebut juga dapat berlaku pada folder **application/views/**. Dimana kita lakukan perubahan kode sesuai nama direktori baru untuk direktori **views**.

```
$view_folder = '';
```

V. Alur Kerja CodeIgniter

Untuk melengkapi pemahaman mengenai CodeIgniter, berikut terdapat sebuah diagram yang menjelaskan bagaimana CodeIgniter bekerja.



Gambar 2. Flow Chart CodeIgniter

Sumber: http://localhost/learn_ci/user_guide/overview/appflow.html [04/10/2015]

Berikut adalah penjelasan alur kerja CodeIgniter:

1. index.php bertindak sebagai controller terdepan, dan menginisialisasi resource yang diperlukan untuk menjalankan CodeIgniter
2. Router memeriksa HTTP request untuk menentukan apa yang harus dikerjakan
3. Jika cache file ada, maka akan ditampilkan langsung, dengan melewati eksekusi normal sistem
4. Sebelum memuat controller, HTTP request akan memeriksa apa yang disubmit user dan memfilternya untuk keamanan

5. Controller memuat model, core, libraries, plugin, helper, dan resource lainnya untuk memproses permintaan tertentu
6. View ditampilkan di browser sesuai proses yang dikerjakan controller. Jika caching dijalankan, view akan di-cache terlebih dahulu agar dapat ditampilkan di request selanjutnya.

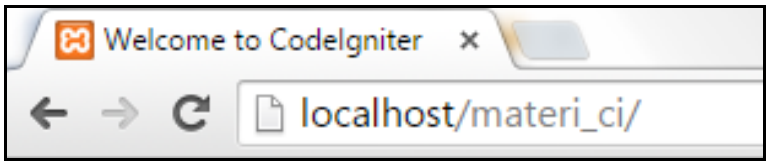
VI. Dasar Routing CodeIgniter

Sekarang kita memasuki tahap routing yang merupakan alur aplikasi CodeIgniter (CI) yang akan nantinya kita bangun. Sebelum memasuki tahap MVC pada CI, kita wajib mengetahui tentang dasar routing. Karena routing inilah tempat dimana kita menentukan default controller dan mengatur URL dari aplikasi kita. Akan tetapi sebelumnya kita simpan dulu paket CI ke dalam direktori **xampp/htdocs/** (pada penggunaan Xampp) dengan nama **materi_ci**.

Untuk direktori routing ini berada pada **application/config/routes.php**. Dari file tersebut, kita akan membuat alur aplikasi CI kita. Secara default file routes.php sebagai berikut:

```
$route['default_controller'] = 'welcome';
$route['404_override'] = '';
$route['translate_uri_dashes'] = FALSE;
```

Pada bagian kode **$route['default_controller'] = 'welcome';** menunjukan tentang default controller atau controller yang pertama kali dibaca oleh CI, adalah controller **welcome**. Jadi apabila kita mengakses URL CI kita yang ada di local web server maka yang akan pertama kali muncul adalah controller **welcome** dengan method **index()**. Lebih jelasnya bisa dilihat pada gambar 3 di bawah ini.



Gambar 3. URL Direktori CodeIgniter di Local Web Server

Sebetulnya URL lengkapnya adalah **localhost/materi_ci/index.php/welcome/index**. Dimana maksud dari URL tersebut bahwa **localhost/materi_ci/index.php/** merupakan base URL sedangkan **welcome** merupakan Controller dengan file

application/controller/Welcome.php dan **index** merupakan method **index()** dari controller **welcome**. Method **index()** merupakan method yang secara otomatis dipanggil apabila kita tidak mencantumkan nama method setelah nama controller pada URL.

Kemudian untuk kode **$route['404_override'] = '';** berfungsi untuk memberi tahu CI apabila default controller (mengacu pada URL) tidak ditemukan. Sebagai contoh:

```
$route['default_controller'] = 'welcome';
$route['404_override'] = 'kosong';
$route['translate_uri_dashes'] = FALSE;
```

Kemudian kita buat file di **application/controllers/Kosong.php** (membuat controller baru yaitu **kosong**). Penamaaan file controller harus diawali dengan huruf besar (kapital) dan nama class didalamnya sesuai dengan nama file controller serta sama menggunakan kapital di awal. Pada controller **kosong** yang kita buat, isikan kode sebagai berikut:

```
<?php
defined('BASEPATH') OR exit('No direct script access allowed');

class Kosong extends CI_Controller {
    public function index()
    {
        echo "<h1>URL (controller dan/atau method) tidak tersedia
!</h1>";
    }

}
```

Coba kita sengaja akses URL http://localhost/materi_ci/index.php/asalxyz/abcde maka browser kita akan menampilkan halaman seperti pada gambar 4 di bawah ini.

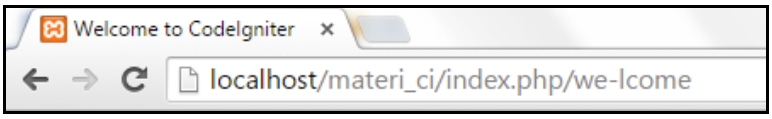
Gambar 4. Controller yang Otomatis Dipanggil Apabila URL Tidak Ditemukan

Selanjutnya untuk kode **`$route['translate_uri_dashes'] = FALSE;`** berfungsi apabila kita mengubah nilainya menjadi **`TRUE`**, maka kita akan diperbolehkan penggunaan tanda dash (-) dalam penulisan controller di URL browser. Sebagai contoh kita akan menjadikan controller **`welcome`** menjadi **`wel-come`**, maka kita ubah pada **`routes.php`** sebagai berikut:

```
$route['default_controller'] = 'wel_come';
$route['404_override'] = 'kosong';
$route['translate_uri_dashes'] = TRUE;
```

Kemudian pada file **`application/controllers/Welcome.php`** Anda rename menjadi **`wel_come.php`**. Begitu juga pada class kita ubah menjadi **`class Wel_com extends CI_Controller`**. Pertanyaannya, mengapa kita harus menggunakan underscore pada nama file dan nama class controller nya? Karena dalam PHP nama class tidak boleh menggunakan tanda dash (-), dan ini juga berpengaruh pada penamaan file, dimana CI membaca class controller berdasarkan nama file.

Jika semua sudah dilakukan kemudian mari kita tes dengan link berikut: **`http://localhost/materi_ci/index.php/wel-come`**, seperti pada gambar 5 di bawah ini.



Gambar 5. Contoh Penggunaan Dash (-) Pada Controller

VII. Daftar Pustaka

7.1. Buku Utama

[1] Nugraha, A.W.P. 2010. CodeIgniter: Cara Mudah Membangun Aplikasi PHP. Jakarta: Mediakita.
[2] Riyanto. 2013. Membangun Mobile Web Store dengan CodeIgniter, MySQL, jQuery Mobile. Yogyakarta: Andi.
[3] Stendy, B.S. 2010. PHP 5 Pemrograman Berorientasi Objek – Konsep & Implementasi. Yogyakarta: Andi.

7.2. Referensi

http://www.w3schools.com/ [September 2015]
http://www.CodeIgniter.com/index.php [September 2015]
http://forum.CodeIgniter.com/portal.php [September 2015]

VIII. Materi Berikutnya

Pokok Bahasan	Interaksi Database dan Form Validasi
Sub Pokok Bahasan	1. Konfigurasi dan koneksi database 2. CRUD (*Create, Read, Update, Delete*) 3. Form validasi