OPTIMASI MODULARITAS DENGAN HIERARCHY MODEL VIEW CONTROLLER (HMVC) PADA FRAMEWORK CODE IGNITER

Ibnu Akil

Program Studi Manajemen Administrasi

ASM BSI JAKARTA

Ibnu.ial@bsi.ac.id

Abstract

The idea of model-view-controller (MVC) is to separate between logic, presentation and entity layer of application. This separation can increase maintainability of a system. But, MVC is still has limitation that is a scalability. To overcome this problem the MVC model has been modified and enhances to cope with scalability, using Hierarchy Model-View-Controller (HMVC). The point of this paper is to show the implementation of HMVC in CodeIgniter as PHP Framework.

Keyword: MVC, HMVC, modularity.

Abstrak:

Ide dari Model-View-Controller (MVC) adalah untuk memisahkan antara lapisan logika, presentasi dan data dari suatu aplikasi. Pemisahan ini dapat meningkatkan kemudahan pengelolaan dari suatu system. Akan tetapi, MVC masih memiliki keterbatasan, yaitu skalabilitas. Untuk mengatasi masalah ini model MVC dimodifikasi dan ditingkatkan agar mampu memenuhi skalabilitas, dengan menggunakan Hierarchy Model-View-Controller (HMVC). Tujuan dari artikel ini adalah untuk menunjukkan implementasi dari HMVC pada framework PHP CodeIgniter.

Kata kunci: MVC, HMVC, modularitas.

I. Pendahuluan

Pengembangan system yang berskala besar dan dikembangkan secara team work, bisa berakhir dengan kekacauan dan kegagalan, jika tidak ada mekanisme manajemen modul yang baik. Kebutuhan untuk memisahkan sisi presentasi dengan sisi logika menjadi sangat signifikan, apalagi belum termasuk memperhitungkan persistensi dari objek atau model.

Pengembangan system berbasis framework tampaknya menjadi solusi yang cukup menjanjikan ditengah kompetisi analisa dan perancangan model perangkat lunak yang saling mengklaim bahwa model-nyalah yang paling baik. Diantara model-model itu adalah Extreme Programming, SCRUM, Agile dan Model Driven Architecture (MDA). Lalu apakah framework itu? Dalam pengembangan system aplikasi web tentunya anda pernah mendengar berbagai macam framework seperti Laravel, Cake PHP, CodeIgniter, Struts, Spring dan lain-lain.

Framework bukanlah tools untuk analisa ataupun pemodelan. Framework adalah satu set komponen-komponen yang terintegrasi yang berkolaborasi untuk menyediakan arsitektur yang reusable untuk aplikasi yang relatif (Schmidt, 2013). Diantara framework pengembangan aplikasi web yang terkenal adalah Code Igniter untuk bahasa PHP, dimana Code Igniter ini telah menerapkan konsep Model View Controller (MVC) yang memisahkan lapisan presentasi dengan lapisan logis pemrograman dan lapisan model. Selain itu Code Igniter dapat dikembangkan lagi dengan penambahan konsep HMVC untuk meningkatkan modularitas.

II. Metode Penelitian

A. Pattern dan Framework

Sebuah pattern adalah solusi berulang untuk masalah yang sama atau standar (Schmidt, Software Patterns, 1996). Sedangkan framework adalah satu set komponen-komponen yang terintegrasi yang berkolaborasi untuk menyediakan arsitektur yang reusable untuk aplikasi yang relatif (Schmidt, 2013).

B. MVC

Menurut Deacon konsep MVC dikenalkan oleh penemu bahasa SmalTalk yaitu Trygve Reenskaug untuk membungkus data dengan pemrosesannya (model) dan mengisolasinya dari proses manipulasi (controller) dan presentasi (view) untuk ditampilkan di user interface ('Uyun & Ma'arif, 2010).

Model

Model adalah sebuah nama yang diberikan kepada penyimpanan yang permanen dari data (table) yang digunakan dalam desain keseluruhan. Model harus mengijinkan akses terhadap data untuk ditampilkan, atau dikumpulkan dan ditulis (Hopkins, 2013).

Secara lugas dapat dikatakan bahwa model adalah sekumpulan kelas atau objek yang merepresentasikan table-tabel didalam database. Kelas ini memiliki fungsionalitas untuk memanipulasi (insert, update, delete, dan select) data pada table yang terkait dengan kelas tersebut.

View

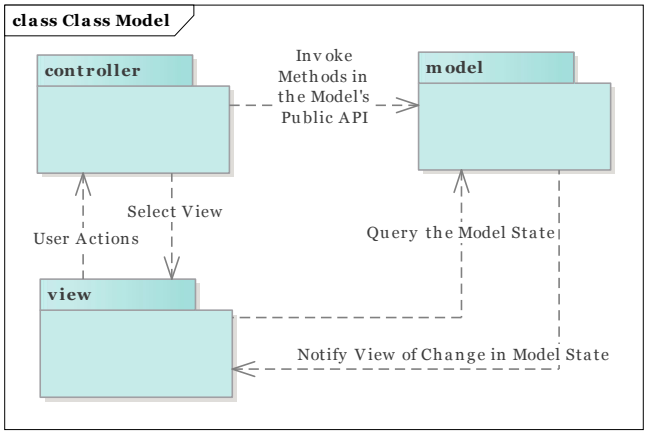
View adalah bagian dari system dimana HTML dihasilkan dan ditampilkan. View juga memicu reaksi dari pengguna, yang kemudian berinteraksi dengan controller (Hopkins, 2013).

Pada umumnya dalam konsep MVC view tidak berinteraksi dengan model, view hanya berurusan dengan tampilan dan template, namun pada prakteknya pembatasan ini mengurangi fleksibilitas logika pada aplikasi yang kompleks sehingga seringkali model juga dapat berinteraksi dengan view secara langsung.

Controller

Tugas Controller adalah menangani input yang dikirim oleh user kemudian menugaskan model dan view untuk melaksanakan tindakan berdasarkan input tersebut. Controller pada dasarnya menjembatani dan mengatur interaksi antara model dengan view.

Menurut Gulzar hubungan antara model – view – controller dapat digambarkan sebagai berikut:

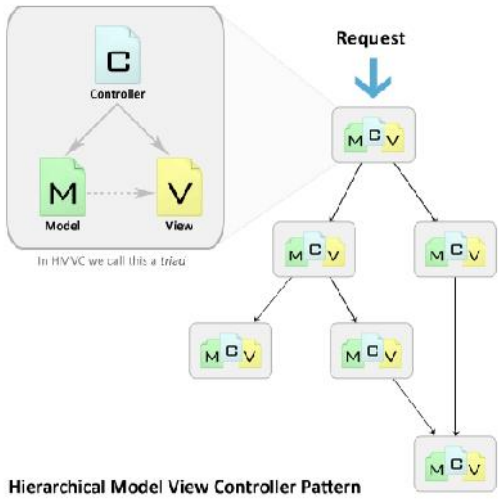


Gambar II.1 – Diagram MVC

HMVC

Model – view – controller adalah pattern yang cukup membantu untuk menyelesaikan masalah pengembangan system yang sederhana. Namun untuk pengembangan arsitektur aplikasi yang melibatkan banyak modul atau component dan banyak pengembang, dependensi akan menjadi isu yang perlu diperhatikan. Begitu juga scalability.

Pada dasarnya HMVC dibangun berdasarkan MVC namun untuk menambahkan modularitas dibuatkan level hirarki yang masing-masing berisi MVC yang mandiri. Perhatikan struktur HMVC berikut:



Gambar II.2 – Struktur HMVC (Cogan, 2010)

Request dari user pertama akan ditangani oleh MVC level atas, kemudian akan diteruskan kepada level dibawahnya berdasarkan path layanan (controller) yang diminta oleh user. Masing-masing MVC tersebut dibungkus dalam modul-modul. Modul-modul tersebut bisa saling berinteraksi dan bertukar layanan. Misalkan modul A dapat memanggil layanan dari model yang ada di modul B, begitu juga sebaliknya.

CodeIgniter

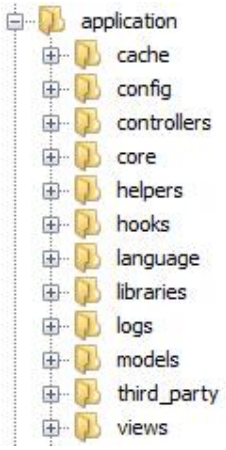
CodeIgniter adalah framework PHP yang dikembangkan oleh EllisLab yang sekarang menjadi proyek dari British Columbia Institute of Technology. CodeIgniter memiliki penggunaan memori yang sangat efisien dibandingkan dengan framework yang lain. Proses instalasinya pun mudah. CodeIgniter sudah menyediakan librari-librari yang standar untuk pengembangan aplikasi web. Dan juga berbasis MVC. Banyak para contributor yang mengembangkan plugin untuk CodeIgniter termasuk salah satunya plugin HMVC.

III. Hasil dan Pembahasan

Untuk mendapatkan gambaran bagaimana HMVC bekerja berikut akan penulis jelaskan rancangan dan desain serta implementasi HMVC dalam suatu studi kasus. Disini CodeIgniter yang digunakan adalah versi 3.1.2 dan untuk plugin HMVC dapat diunduh di https://bitbucket.org/wiredesignz/codeigniter-modular-extensions-hmvc.

Perbedaan MVC dan HMVC

Struktur dasar framework CodeIgniter dapat dilihat di gambar II. 3 berikut:

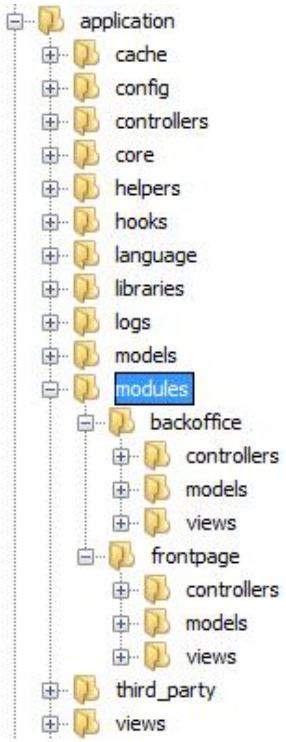


Gambar III.1 Struktur Dasar MVC Framework CodeIgniter

Yang perlu diperhatikan adalah tiga folder yaitu: models, views, controllers. Ketiga folder tersebut berisi kelas-kelas yang relevan, folder controllers berisi kelas-kelas controller, folder models, berisi kelas-kelas model, sedangkan folder views berisi file-file yang menghasilkan HTML.

Untuk system berskala kecil, struktur MVC dasar tersebut sudah mencukupi, namun untuk system berskala besar dimana banyak penambahan modul-modul baru untuk pengembangan system, model MVC tidak bisa mengakomodir pengembangan modul.

Dengan mengembangkan model MVC menjadi HMVC, struktur framework CodeIgniter menjadi seperti gambar II.4 berikut:



Gambar III.2 Struktur HMVC pada Framework CodeIgniter

Disitu kita lihat ada penambahan folder "modules" yang berisi sub modul; "backoffice" dan "frontpage", sub modul ini masing-masing berstruktur MVC. Sedangkan folder models, views dan controllers yang berada pada sub folder application menjadi top level dari MVC. Dan model ini disebut HMVC.

Dengan struktur HMVC ini kita dapat mengembangkan system dengan lebih mudah, tinggal menambah modul baru saja tanpa harus mengganggu modul yang lain. Salah satu kelebihan dari HMVC ini adalah reusable dimana masing-masing modul dapat saling berbagi data atau operasi.

Konfigurasi Framework dengan Model HMVC

Yang pertama dilakukan adalah mengkonfigurasi file config.php yang ada pada path /application/config/. Tambahkan cuplikan kode berikut:

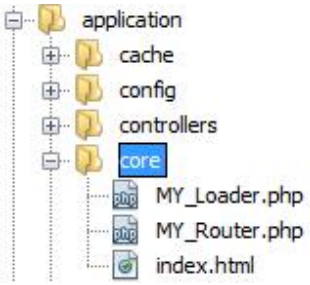
```
$config['modules_locations'] = array(

      APPPATH.'modules/' => '../modules/',

  );
```

Kemudian edit file "routes.php" seperti baris kode berikut:

```
$route['default_controller'] = 'frontpage';
```

Baris tersebut menjelaskan bahwa yang menjadi default dari controller adalah modul "frontpage" dimana ketika system diakses maka user akan diarahakan ke modul "frontpage".

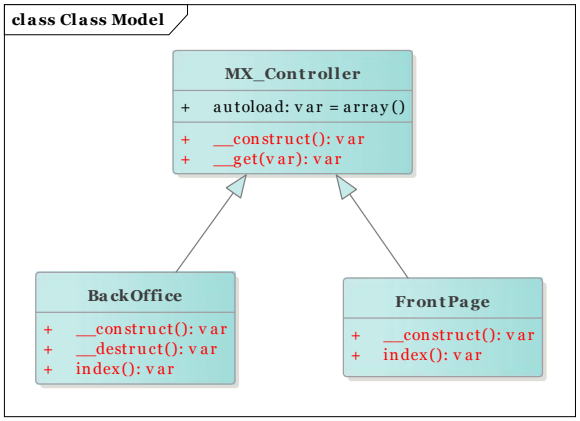
Kemudian copy-kan file yang ada pada folder core dari plugin HMVC, ke folder /application/core pada framework.



Gambar III.3 Core HMVC

Kemudian yang terakhir copy-kan folder "MX" dari plugin HMVC, ke folder /application/third_party.

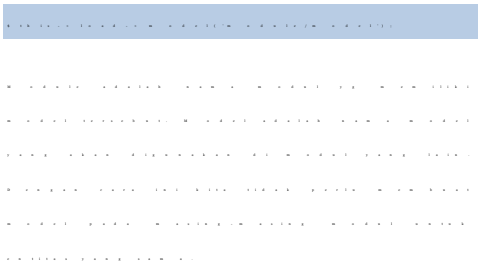
Setiap controller yang berada di dalam masing-masing modul haruslah mewarisi dari kelas MX_Controller yang ada pada folder “MX”. Perhatikan rancangan class diagram berikut:



Gambar III.4 Class Diagram Controller

Reusable Model

Kelebihan dari HMVC salah satunya adalah reusability. Model sebagai representasi dari entitas atau table dapat dipakai oleh modul yang berbeda-beda. Untuk menggunakan model pada modul yang berbeda penggunaannya adalah sebagai berikut:



IV. Kesimpulan

Model HMVC memiliki beberapa kelebihan dibanding MVC diantaranya:

- Modularization: mengurangi ketergantungan antara bagian-bagian aplikasi yang terpisah.
- Organization: memiliki folder-folder (modul) dari MVC yang terkait membuat load kerja yang lebih ringan.
- Reusability: secara alami berdasarkan desain, sangat mudah menggunakan kembali setiap bagian dari kode.
- Extendibility: membuat aplikasi lebih mudah dikembangkan tanpa harus mengorbankan maintenance.

Secara umum performance antara MVC dan HMVC adalah sama, namun untuk scalability HMVC lebih baik. Tinggal penggunaannya saja tergantun dari system yang akan dikembangkan.

Daftar Pustaka

Cogan, B. (2010, Mei 18). *How to Tutorials: Envatotuts+*. Retrieved Maret 4, 2017, from Envatotuts+: https://code.tutsplus.com/tutorials/hmvc-an-introduction-and-application--net-11850

Hopkins, C. (2013, Maret 4). *Web: Sitepoint*. Retrieved Maret 4, 2017, from Sitepoint: https://www.sitepoint.com/the-mvc-pattern-and-php-1/

Schmidt, D. C. (1996). Software Patterns. *Communication of the ACM*.

Schmidt, D. C. (2013, Oktober 31). *http://www.cs.wustl.edu/~schmidt*. Retrieved Maret 4, 2017, from http://www.cs.wustl.edu/: http://www.cs.wustl.edu/~schmidt/PDF/patterns-intro4.pdf

'Uyun, S., & Ma'arif, M. R. (2010). Implementation of Model View Controller (MVC). *Seminar Nasional Aplikasi Teknologi Informasi*. Yogyakarta.

BIODATA PENULIS

Ibnu Akil. Jakarta 15 Januari 1980. Magister Ilmu Komputer Program Pasca Sarjana Nusamandiri. Bekerja sebagai Dosen di AMIK BSI dan Konsultan IT.