MODUL LARAVEL VERSI 4.2

Step by Step to Laravel

(route-controller-view)

Part 1

Nur Chamdani

2 September 2016

Jumat

DAFTAR ISI

1. Pengenalan laravel 4.2
   1. Pengembang pertama kali
2. Persiapan laravel 4.2
   1. Persiapan peralatan
   2. Persiapan pemasangan
      1. Pasang peralatan pembangun
      2. Pasang pusat data lokal
3. Kupas tuntas laravel 4.2
   1. Struktur laravel 4.2
   2. Raw Query
   3. Modeling Table
   4. Routing No Controller
   5. Routing Controller
      1. Routing alias and use method Controller
   6. Controller
      1. pasang controller via CMD
      2. fungsi tanpa parameter
      3. fungsi dengan parameter
   7. View
      1. pengunaan view tanpa blade
      2. pengunaan view dengan blade
4. Project 1 Simple Native Website dan Simple Native Crud
   1. Simple Native Website
      1. Homepage
      2. Setting migrasi, routing, controller, view
      3. Hasil
   2. Simple native crud
      1. Instruksi
      2. Setting migrasi, routing, controller, view
      3. Setting crud
         1. Insert
         2. Update
         3. delete
      4. Hasil
5. Project 5 Shared hosting native website
   1. Instruksi
   2. setting migrasi, routing, controller, view
   3. setting crud
      1. insert
      2. update
      3. delete
   4. Testing result

SELAMAT MENCOBA !!

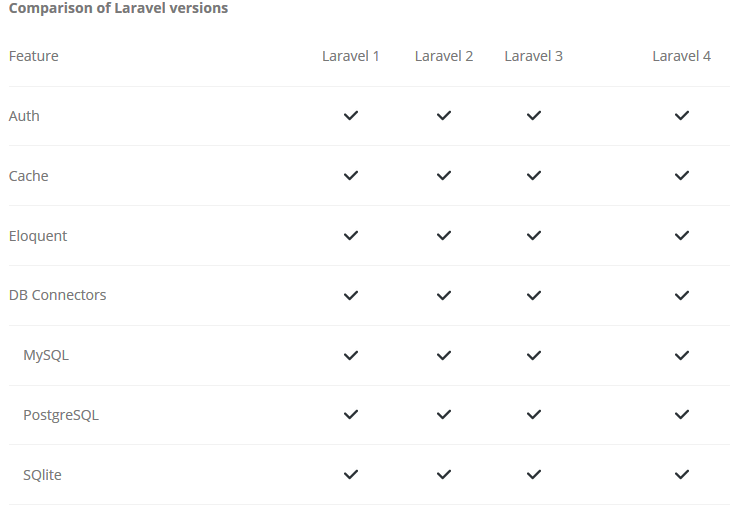
Salam Coding !!!!

BAB I

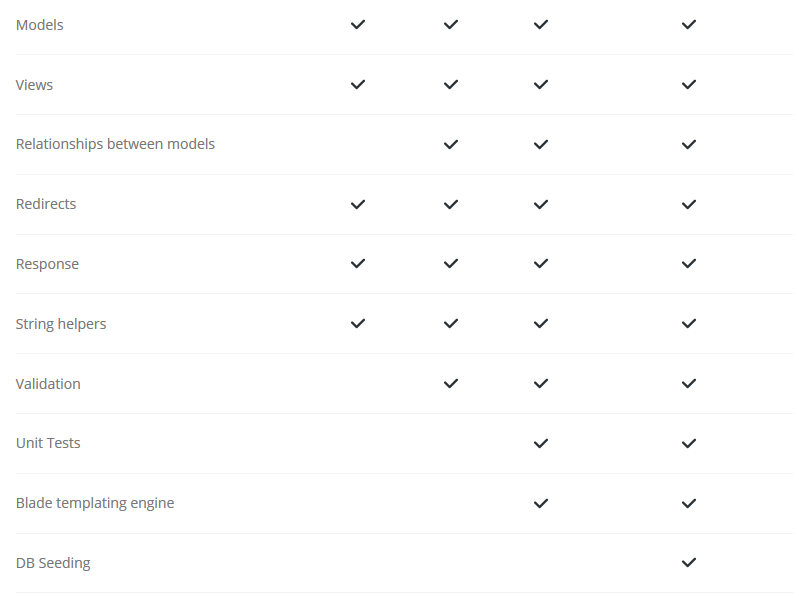
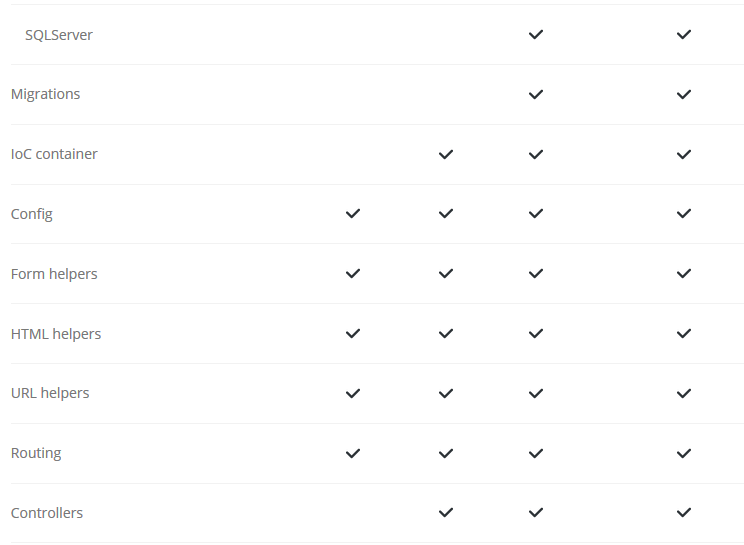
PENGENALAN LARAVEL VERSI 4.2

1. Pengenalan Laravel

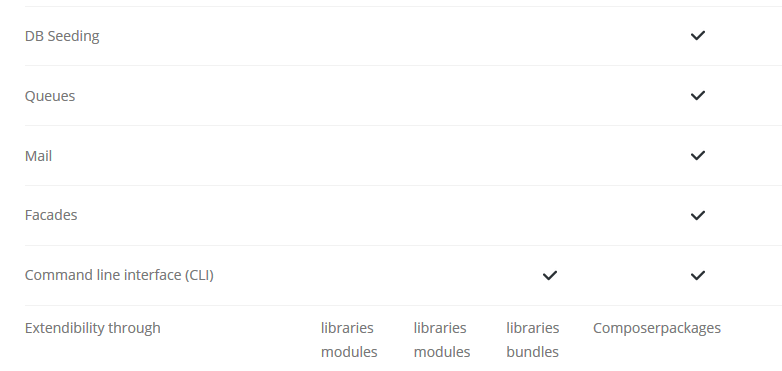
Laravel merupakan bahasa pemprograman berorientasi objek PHP. Laravel dikembangkan pertama kali oleh Taylor Outwell 1 Juli 2014, yang pada awalnya Taylor Outwell mengunakan Codeigniter. Jadi, Laravel merupakan generasi ke 2 setelah CodeIgniter. Berikut manifaktur yang bersumber dari <https://maxoffsky.com/code-blog/history-of-laravel-php-framework-eloquence-emerging/> :



Gambar 1.1 Manufaktur Semua Versi Laravel Part 1



Gambar 1.2 Manufaktur Semua Versi Laravel Part 2



Gambar 1.3 Manufaktur Semua Versi Laravel Part 3

* 1. **pengembangan laravel**

Perilisan laravel 1.0 diawali dengan fitur autentifikasi, Eloquent ORM untuk operasi lokasi database, model dan relasi, mekanisme router, pencacahan, sesi, tampilan, modul dan perpustakan yang terpercaya, lembar input dan masih banyak lagi. Pengembangan MVC tidak dilakukan pada fitur ini bagi pengembang. Pengembangan masih bertahap pada paginasi, metode yang dibenarkan dan pemasang berkas dengan perintah baris.

Perilisan laravel 2 membutuhkan waktu 6 bulan berakhir tanggal 24 November 2011. Fitur yang ditambahkan adalah fungsi scss, less, dan pengubahan fungsi html menjadi blade dengan tanda kurawal ({}) . hal ini diakukan agar pengunaan MVC bisa lebih maksimal karena menghapus beberapa modul pendukung yang berpengaruh pada kinerja pada router maupun controller

Pada 22 Februari 2012 merupakan perilisan laravel 3 sebagai framework website yang lebih ringkas dan fitur – fitur yang memadai. Versi ini lebih cepat pencacahannya dari Codeigniter maupun Kohana. Beberapa pengguna jejaring social yang berdedikasi selama 5 bulan mengembangkan framework yang ada dengan tambahan “illuminate”. Fitur selain penambahan “illuminate” adalah dependency manager, dimana semua versi laravel mulai dari 1.0 sampai 5.3 bisa diunduh sesuai kebutuhan.

Pengembangan laravel 4 merupakan penyelesaian dalam mengurangi kredibilitas sebuah framework itu sendiri. Laravel 4 dapat menulis ulang dari koleksi component diintegrasi membentuk framework yang dibutuhkan. Laravel 4 yang rilis pada 28 Mei 2013.

**BAB II**

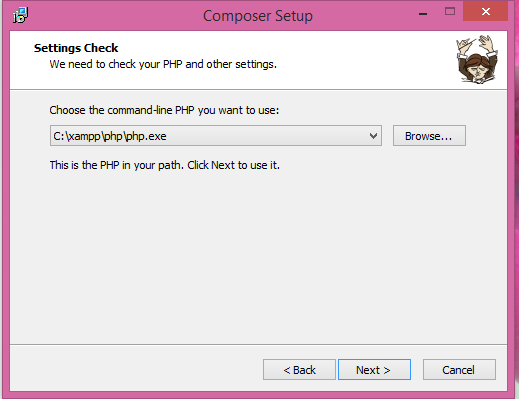
**PERSIAPAN**

**PEMASANGAN**

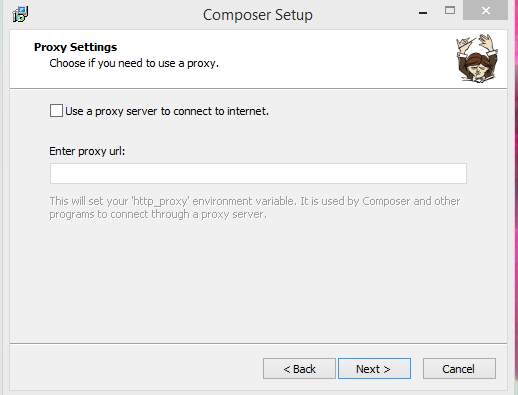
1. **peralatan dan pemasangan**

Pemasangan awal pada laravel adalah XAMPP dan MySQL untuk control pusat data local dan kontrol database. Langkah pertama unduh XAMPP sesuai *platform* (Sistem Operasi) yang anda miliki, dan unduh MySQL pada website [www.4shared.com](http://www.4shared.com) kemudian pasang, perlu diingat bahwa pemasangan diluar dari local disk C (C:\\) berpengaruh pada integrasi php pada pemeriksaan pengaturan otomatis. apabila peralatan pendukung dipasang selanjutnya ke pemasangan *composer*. Kebutuhan koneksi internet adalah hal mutlak pada proses pemasangan laravel melalui *command prompt*. Pengambilan berkas melalui *Github* bisa dilakukan tapi setiap laravel memiliki *key* project, fungsi *key* itu sendiri adalah sebagai kunci untuk melakukan akses keseluruh file pada setiap user pemasang.

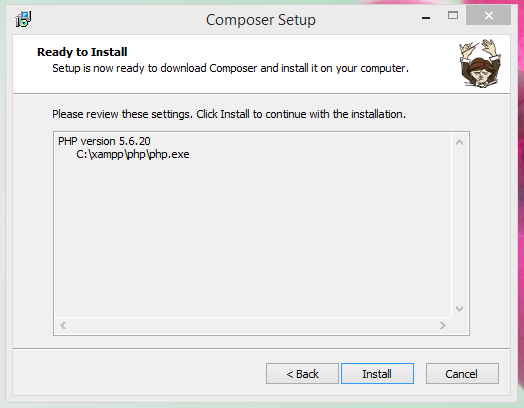
Pemasangan via internet disarankan dengan kecepatan stabil agar memudahkan pemasangan dan penggunaan laravel itu sendiri. Sebelum memasangkan laravel pada *device* mengharuskan pemasangan Composer sebagai aplikasi manager kebutuhan pada framework Laravel. Pemasangan Composer meliputi:

1. masukkan situs [www.composer.com](http://www.composer.com) pada mesin pencari situs.
2. Dari cari Composer-setup.exe pada [www.composer.com](http://www.composer.com), klik, simpan dan jalankan.
3. Setelah dijalankan klik next, setting check adalah pemeriksaan pengaturan php otomatis pada local disk C.

**Gambar 1.4 Setting Check PHP**

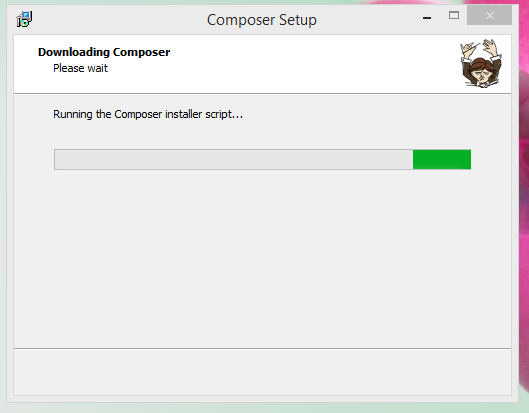
1. Apabila next di klik, akan ada check koneksi dan composer akan menampilkan permintaan koneksi via proxy atau tidak. klik next jika tidak dan jika menggunakan masukkan IP proxy anda.

**Gambar 1.5 setting proxy**

1. Klik next untuk melanjutkan, composer menampilkan hasil *review version* php yang anda pasang

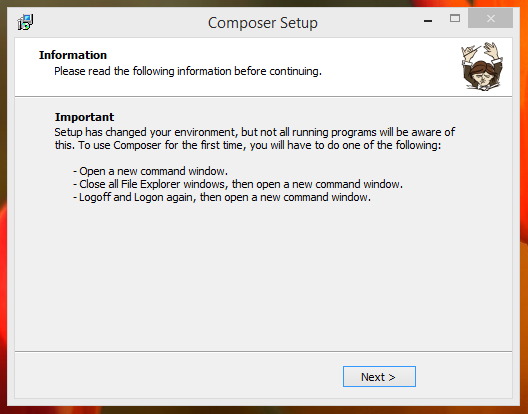
**Gambar 1.6 Konfigurasi composer pada versi php**

1. Selanjutnya proses pengunduhan dan pemasangan file composer.phar

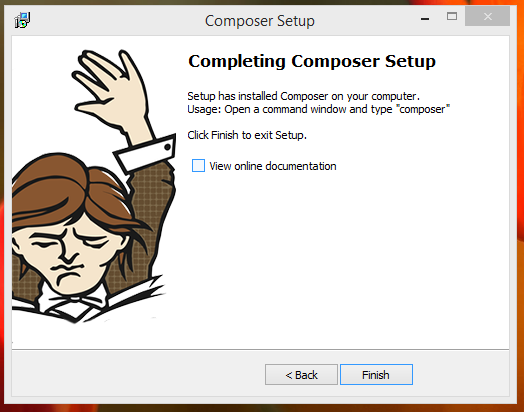


**Gambar 1.7 pemasangan file composer.phar**

1. Informasi sukses pemasangan dari composer dan laravel siap dijalankan.



**Gambar 1.8 Informasi pemasangan suksees**

1. Informasi pelengkap composer berisi penawaran menampilkan dokumen composer pada user untuk memilih (centang) atau tidak.

**Gambar 1.9 informasi pelengkap berisi menampilkan**

**dokumentasi atau tidak**

1. Mulai membuat aplikasi.

**BAB III**

**KUPAS LARAVEL VERSI 4.2**

1. **perbedaan laravel dari framework menurut beberapa pendapat**
   1. **struktur folder laravel 4.2**

--------------------------

-->app -- berisi folder dan fille

-->bootstrap --||--

-->public --||--

-->vendor --||--

->artisan.php

->composer.json

->composer.lock

->phpunit.xml

->server.php

---------------------------

* 1. **composer php artisan**

composer php artisan berfungsi sebagai script untuk membuat controller, model dan yang lainnya. Sebelum melakukan penggunaan php artisan pastikan lokasi *command prompt* anda berada di folder htdocs dimana project anda berada seperti:



**Gambar 1.10 Lokasi file project**

Selanjutnya Contoh membuat Controller dengan php artisan:



**Gambar 1.11 buat controller dari php artisan CLI**

Selanjutnya anda bisa mengubah kebutuan apa yang anda buat hanya dengan mengubah **controller:make** menjadi **model:make** dan seterusnya.

* 1. **Query Builder**

Query merupakan bahasa pemrograman yang dieksekusi oleh sebuah computer dengan keluaran sesuai yang diinginkan pengguna. Query Builder merupakan macam – macam bentuk Query dari bahasa pemrograman SQL. Query Builder tergantung pada penggunaan setiap framework karena tidak semua Query bisa dimengerti framework. Framework laravel menggunakan 3 type Query seperti Raw Query, Eloquent ORM dan Query Builder

Raw Query adalh query identik penamaan kelas adalah **DB::(**jenis perintah**)(**isi perintah sesuai jenis**)**. jadi, pada Raw Query kita bisa memasukkan sebuah perintah SQL seperti pada perintah SQL manual ke bentuk Raw Query tanpa membuat object kelas.

Query Builder merupakan Query yang sama persis dengan query Codeigniter dan query ini bisa dijalankan dalam berbagai versi laravel. Script Query Builder seperti berikut:

$posts = DB::table('post')->get();

Contoh Eloquent ORM, ORM (Object Relational Mapper) dimana kita bisa membuat sebuah kelas table sendiri sesuai yang kita inginkan. Seperti:

<?php

class Post extends Eloquent {

protected $table = 'post';

} ?>

Untuk tutorial kali ini kita menggunakan Raw Query.

* 1. **Routing No Controller**

Penggunaan routing tanpa menggunakan Controller pada laravel bisa diakukan selama memasukkan nama alamat referensi yang diinginkan. Contoh:

Route::get(‘/’, function(){

Return View::make(‘hello’);

});

* 1. **Routing dan Controller**
     1. **Routing alias dan pengunaan metode Controller**

Penggunaan alias adalah penamaan dari route untuk metode papda kelas Controller. Dengan penulisan:

Route::post(‘input’, array(‘as’ => ‘masuk’, ‘uses’ => ‘HomeController@input\_data’));

Cara membaca route diatas : nama alias route adalah masuk menggunakan fungsi input\_data dengan kelas HomeController dengan alamat route input

Penggunaan lain dengan tujuan untuk mengubah data adalah menambahkan tanda kurung kurawal pada nama alamat route. Dengan penulisan:

Route::get(‘ubah/{id}’, array(‘as’ => ‘masuk’, ‘uses’ => ‘HomeController@input\_data’));

Cara membaca route diatas : nama alias route adalah masuk menggunakan fungsi input\_data dengan kelas HomeController dengan alamat route ubah 123

Perlu diingat untuk melakukan perubahan pada data gunakan

stdClass *get* karena stdClass post tidak menerima data dari address

* 1. **Controller**

Controller merupakan pusat kendali fungsi dari panggilan route. Kendati demikian pemanfaatan Controller juga tergantung kebutuhan dari penguna sendri (pembangun). Keluaran dari Controller bisa sebuah pemanggilan View maupun array atau json.

* + 1. **Pasang Controller dengan CMD**

laravel menyediakan pemasangan Controller melalui CMD. Namun kita masih bisa menggunakan teori “copas” hanya saja penulis tidak menyarankan. Perlu diingat bahwa untuk pemasangan Controller maupun Migrasi tidak menggunakan koneksi internet karena sudah terinstal composer sebagai compiler dari “php artisan”. Berikut cara pemasangan melalui CMD:



**Gambar 1.12 Membuat Controller dari CMD**

* + 1. **fungsi tanpa parameter**

pembuatan fungsi tanpa parameter pada kelas Controller sebagai berikut:

public function input\_data(){

Return View::make(‘hello’);

}

Fungsi tersebut bernama input\_data dengan output dari fungsi itu adalah menampilkan file helo dengan stdClass adalah make, dimana make berfungsi menampilkan file. Fungsi ini hanya digunakan untuk menampilkan dan tidak digunakan untuk mengubah data.

* + 1. **fungsi dengan parameter**

pembuatan fungsi parameter pada kelas Controller sebagai berikut:

public function input\_data($id){

Return View::make(‘hello’, array(‘nomer’ => $id));

}

Fungsi tersebut bernama input\_data memiliki parameter $id. $id merupakan variable bernama id. Nilai dari variable id diambil pada nama yang ada pada isi kurawal. Penambahan array pada array merupakan cara menyisipkan sebuah variable untuk dimuat pada halaman yang diinginkan

Route::get(‘ubah/{id}’,………..

Perlu diingat, penggunaan {} bernilai array dan harus berekstra hati – hati dalam setiap variable array, karena PHP sangat spesifik.

* 1. **View**

View pada laravel berfungsi sebagai menampilkan halaman yang diinginkan pada nilai balik dari fungsi yang kita buat.

* + 1. **pengunaan view tanpa blade**

penggunaan ekstensi file .blade pada setiap file php adalah salah satu bahasa php yang cukup simple dalam penulisan namun ketika dijalankan ekstennsi tersebut sama persis dengan penulisan php manual. Cara penulisan tersebut yaitu:

<form action=”<?php route(‘masuk’); ?>” >

</form>

* + 1. **pengunaan view dengan blade**

penggunaan ekstensi file .blade pada file php semisal “test.blade.php” agar penulisan bahasa yang memiliki tanda {{}} kurawal ganda bisa terbaca saat eksekusi file tersebut. Contoh penulisan blade pada sebiah file php.

{{ Form::open() }}

{{ Form::input(‘nama’, array(‘class’ => ‘test’)) }}

{{ Form::submit(‘close’, ‘close’) }}

{{ Form::close() }}

**BAB IV**

**PROJECT 1**

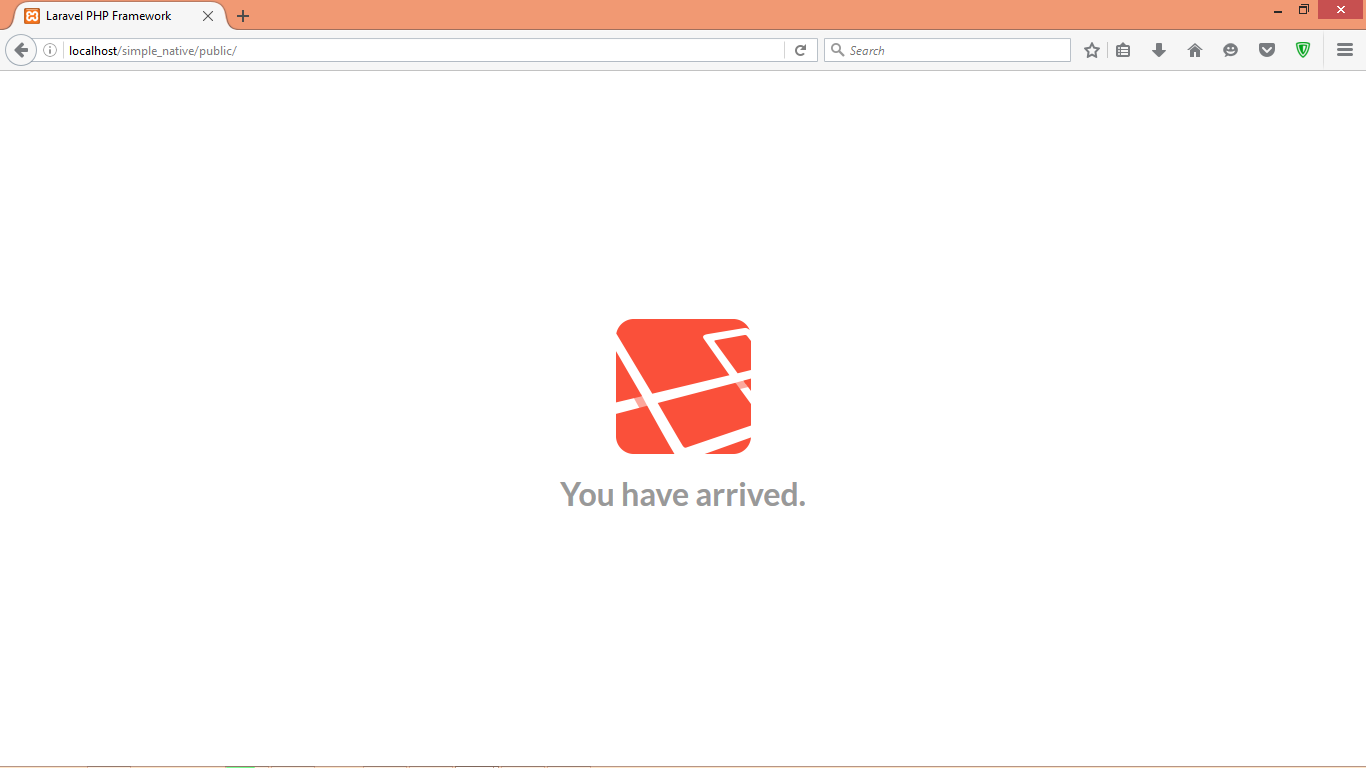
**SIMPLE NATIVE WEBSITE**

**dan**

**CRUD NATIVE WEBSITE**

* 1. **Simple native website**
     1. **Homepage**

Halaman utama pada laravel 4.2 jika berhasil memasangkan.



**Gambar 1.13 Membuat Controller dari CMD**

* + 1. **atur route, controller, view**
       1. **Route awal**

Route::get('/', array(‘as’ => ‘home’, ‘uses’ => ‘TestMakeController@index’));

* + - 1. **Controller awal**

Tambahkan “Return View::make(“hello”) dalam fungsi index.

public function index()

{

//

Return View::make("hello");

}

* + 1. **View awal**

Kembali ke homepage.

* 1. **SIMPLE NATIVE CRUD**
     1. **lekas buat !!!**

kembali ke project 1 simple native. Kemudian ubah project tersebut menjadi native\_crud. Tambahkan:

* + - 1. **atur routing, controller, view**
         1. **routing**

ubah isi file routers.php seperti ini :

Route::get(‘/’, array(‘as’ => ‘simpan’, ‘uses’ => ‘TestMakeController@beranda’);

Route::post(‘insert’, array(‘as’ => ‘simpan’, ‘uses’ => ‘TestMakeController@simpan’);

Route::get(‘lihat/{id}’, array(‘as’ => ‘lihat’, ‘uses’ => ‘TestMakeController@lihat’);

Route::get(‘ubah/{id}’, array(‘as’ => ‘lihat’, ‘uses’ => ‘TestMakeController@detail’);

Route::post(‘simpan/update’, array(‘as’ => ‘simpan’, ‘uses’ => ‘TestMakeController@ubah’);

Route::post(‘hapus/{id}’, array(‘as’ => ‘hapus’, ‘uses’ => ‘TestMakeController@hapus’);

* + - * 1. **controller**
        2. **view**
      1. **atur crud**
         1. **insert**
         2. **update**
         3. **delete**
      2. **uji coba**

BAB VI

PART II

Shared Hosting, AJAX laravel, Phonegap

1. Shared Hosting

Point utama adalah pemasangan versi 4.2 pada service server hosting dan menerapkan project simple native native crud.

1. AJAX Laravel

Point utama adalah penerapan script dari jquery $.post() dan $.get() dengan mennggubah project simple native ajax crud.

1. Phonegap (IntelXDK)

Point utama adalah penerapan laravel pada mobile html5 dengan peralatan pembangun IntelXDK .

1. Upload Google Play Store

Point utama adalah mengunggah fille yang sudah berekstensi apk kedalam akun google play store untuk bisa dinikmati diri sendiri maupun orang lain.