

CONTROLLER

1.1 Kompetensi Utama

- Diharapkan mahasiswa dapat memahami controller pada laravel.

1.2 Kompetensi Penunjang

- Mengenalkan kepada mahasiswa tentang controller
- Membuat controller pada laravel

1.3 Dasar Teori

1. Pengertian Controller

Secara sederhana, **controller** adalah bagian kode program yang mengatur logika serta lalu lintas data. Controller berada di pusat MVC. Pada saat user mengetik sebuah alamat di web browser, route akan membaca alamat tersebut dan (idealnya) memanggil controller yang sesuai. Di dalam controller inilah logika program kita tulis. Apabila butuh mengambil data dari database, controller akan mengakses model. Hasil dari model dikembalikan ke controller yang bisa diolah lebih lanjut untuk kemudian dikirim ke view. Sesampainya di view, data tinggal ditampilkan ke web browser.

2. Cara Mengakses Controller

Dalam Laravel, route adalah pintu masuk ke setiap alamat yang kita ketik di web browser. Route-lah yang menentukan proses mana yang akan menangani URL tersebut, apakah langsung ke View, atau ke Controller.

untuk memanggil Controller ditulis dengan format berikut:

```
Route::get('<url>', [App\Http\Controllers\Nama_Controller::class, 'nama_method']);
```

Sebagai contoh, untuk memanggil method **index()** di controller bernama **MahasiswaController**, penulisan route-nya adalah:

```
Route::get('/',  
[App\Http\Controllers\MahasiswaController::class, 'index']);
```

Dengan penulisan ini, maka ketika alamat `http://localhost:8000` di akses, route akan menjalankan method `index()` milik `MahasiswaController`.

Contoh lain, apabila kita ingin alamat URL `http://localhost:8000/mahasiswa` akan menjalankan method `show()` milik `MahasiswaController`, penulisan routenya adalah:

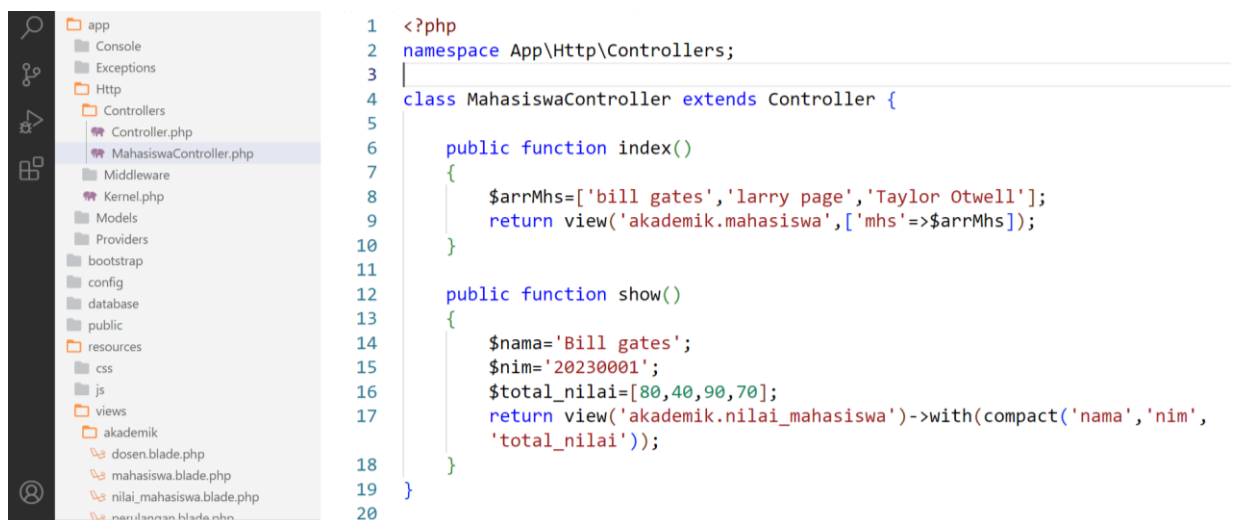
```
Route::get('/mahasiswa',  
[App\Http\Controllers\MahasiswaController::class, 'show']);
```

Dalam laravel, controller berbentuk sebuah file yang berisi 1 object dengan berbagai method. S file controller berada di folder `app\Http\Controllers\`.

3. Membuat Controller Manual

Terdapat 2 cara pembuatan controller, yakni secara manual dari text editor, atau menggunakan perintah `php artisan`.

Contoh membuat controller secara manual, pada editor buat file baru dengan nama `MahasiswaController.php`, lalu simpan di folder `app\Http\Controllers`. Sehingga alamat file ada di: `app\Http\Controllers\MahasiswaController.php`. Isi file ini dengan kode berikut:



Di baris 2 terdapat pendeklarasian namespace `App\Http\Controllers`. Ini sama seperti yang ada di file `Controller.php`. Karena memiliki namespace yang sama, kita bisa langsung mengakses semua kode yang ada di `Controller.php`.

Struktur namespace ini juga bersesuaian dengan alamat **path**, yakni folder `app\Http\Controllers\`. Jika nantinya file controller di pindah ke folder lain, penulisan namespace ini juga harus disesuaikan.

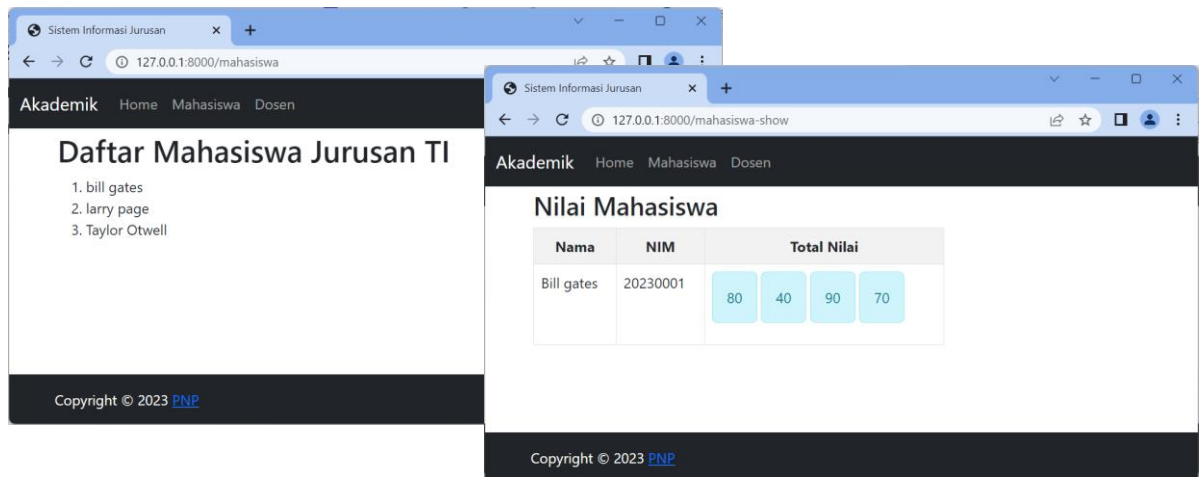
Selanjutnya di baris 4 kita membuat class `MahasiswaController` yang meng-extends `Controller`. Artinya, class `MahasiswaController` ini adalah **turunan** atau *child class* dari class `Controller`. Di dalam Laravel, setiap controller **wajib** meng-extends class `Controller`.

Class `MahasiswaController` memiliki 2 buah method: `index()` dan `tampil()`. Isi kedua method menampilkan view mahasiswa dan nilai_mahasiswa yang ada di folder akademik. Sekarang mari kita coba akses. Tambah dua routes berikut ke dalam file `routes/web.php`:

```
routes > web.php > ...
6 | -----
7 | Web Routes
8 | -----
9 |
10 | Here is where you can register web routes for your application. These
11 | routes are loaded by the RouteServiceProvider and all of them will
12 | be assigned to the "web" middleware group. Make something great!
13 |
14 | */
15 |
16 | Route::get('/', function () {
17 |     return view('welcome');
18 | });
19 |
20 | Route::get('/mahasiswa', [\App\Http\Controllers\MahasiswaController::class, 'index']);
21 | Route::get('/mahasiswa-show', [\App\Http\Controllers\MahasiswaController::class, 'show']);
22 |
```

Sebagai pengulangan, route di baris 20 artinya jika halaman `http://localhost:8000/mahasiswa` diakses, eksekusi method `index()` yang ada di `MahasiswaController`.

Sedangkan route di baris 21 berarti jika halaman `http://localhost:8000/mahasiswa-show` diakses, eksekusi method `show()` yang ada di `MahasiswaController`. Berikut hasilnya:



Alternatif penulisan lain, namespace "App\Http\Controllers\MahasiswaController" bisa di pindah ke bagian atas file routes/web.php. Sebagai contoh, dalam praktek sebelumnya file routes/web.php sudah berisi 2 route berikut:

```

15
16 Route::get('/', function () {
17     return view('welcome');
18 });
19
20 Route::get('/mahasiswa', [\App\Http\Controllers\MahasiswaController::class, 'index']);
21 Route::get('/mahasiswa-show', [\App\Http\Controllers\MahasiswaController::class, 'show']);
22

```

Dengan memindahkan namespace "App\Http\Controllers\MahasiswaController" ke bagian atas, file route bisa ditulis seperti ini:

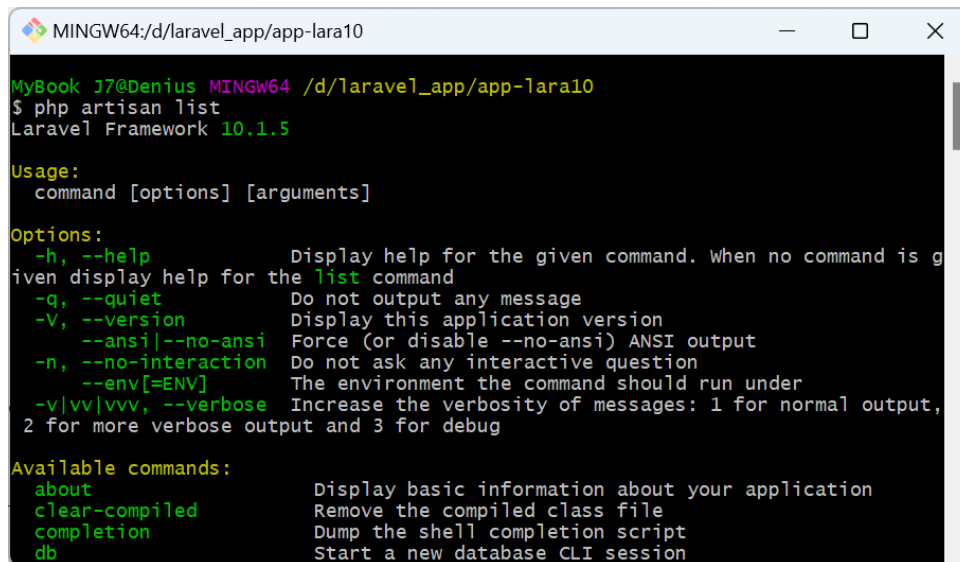
```

routes > web.php > ...
1 use Illuminate\Support\Facades\Route;
2
3 use \App\Http\Controllers\MahasiswaController;
4
5 /*
6  |-----
7  | Web Routes
8  |-----
9  |
10 | Here is where you can register web routes for your application. These
11 | routes are loaded by the RouteServiceProvider and all of them will
12 | be assigned to the "web" middleware group. Make something great!
13 |
14 |
15 */
16
17 Route::get('/', function () {
18     return view('welcome');
19 });
20
21 Route::get('/mahasiswa', [MahasiswaController::class, 'index']);
22 Route::get('/mahasiswa-show', [MahasiswaController::class, 'show']);
23

```

4. Membuat Controller menggunakan PHP Artisan

Untuk melihat seluruh daftar perintah yang tersedia, silahkan buka cmd, masuk ke folder laravel dan ketik perintah `php artisan list`:



```
MINGW64:/d/laravel_app/app-lara10
MyBook J7@Denius MINGW64 /d/laravel_app/app-lara10
$ php artisan list
Laravel Framework 10.1.5

Usage:
  command [options] [arguments]

Options:
  -h, --help            Display help for the given command. When no command is g
                        iven display help for the list command
  -q, --quiet           Do not output any message
  -V, --version         Display this application version
  --ansi|--no-ansi     Force (or disable --no-ansi) ANSI output
  -n, --no-interaction  Do not ask any interactive question
  --env[=ENV]          The environment the command should run under
  -v|vv|vvv, --verbose Increase the verbosity of messages: 1 for normal output,
                        2 for more verbose output and 3 for debug

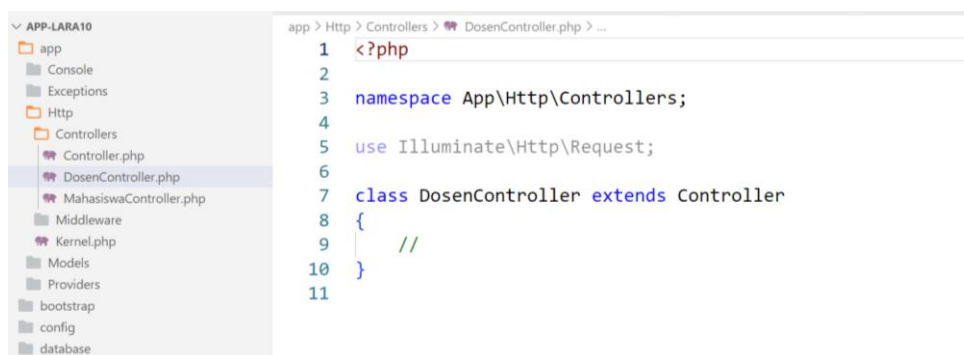
Available commands:
  about                Display basic information about your application
  clear-compiled       Remove the compiled class file
  completion           Dump the shell completion script
  db                  Start a new database CLI session
```

perintah untuk membuat controller adalah:

`php artisan make:controller <namaController>`

Sebagai contoh, untuk membuat controller DosenController. Perintahnya adalah:

`php artisan make:controller DosenController`



```
APP-LARA10
├── app
│   ├── Console
│   ├── Exceptions
│   └── Http
│       ├── Controllers
│       │   ├── Controller.php
│       │   ├── DosenController.php
│       │   └── MahasiswaController.php
│       ├── Middleware
│       └── Kernel.php
├── Models
├── Providers
├── bootstrap
├── config
└── database

app > Http > Controllers > DosenController.php > ...
1  <?php
2
3  namespace App\Http\Controllers;
4
5  use Illuminate\Http\Request;
6
7  class DosenController extends Controller
8  {
9      //
10 }
11
```

Inilah struktur dasar controller yang dibuat menggunakan perintah `php artisan make:controller`. Terlihat kerangka controller sudah lengkap, yang terdiri dari penulisan *namespace* di baris 3 serta pembuatan class `DosenController` yang sudah langsung *mengextends* class `Controller` di baris 7.

Selain itu terdapat 1 perintah tambahan, yakni use Illuminate\Http\Request di baris 5. Jika diperhatikan, ini adalah perintah untuk proses import class bernama **Request**. Sesuai dengan namanya, class Request berisi informasi seputar *HTTP request* yang dikirim dari web browser. Class Request ini tidak wajib (opsional) dan baru terpakai jika kita ingin memproses form. Yang harus ada di sebuah controller adalah penulisan *namespace*, serta pembuatan sebuah class controller yang meng-*extends* class Controller.

5. Membuat Controller pada Folder yang berbeda

Untuk aplikasi yang besar, bisa jadi jumlah controller kita banyak, maka akan lebih baik jika file controller diatur ke folder terkait agar lebih rapi. Namun kita tidak bisa sekedar memindahkan file controller karena ada beberapa hal yang harus diubah.

Sebagai bahan praktek, silahkan buat folder '**Auth**' di dalam app\Http\Controllers, lalu pindahkan file DosenController.php ke dalam folder ini, sehingga alamat path file berada di app\Http\Controllers\Auth\DosenController.php. Pembuatan folder Auth misalnya dipakai untuk menampung semua controller yang berkaitan dengan otentikasi.

```
1  <?php
2
3  namespace App\Http\Controllers;
4
5  use Illuminate\Http\Request;
6
7  class DosenController extends Controller
8  {
9      public function index()
10     {
11         $arrDosen=['Ronal Hadi','Deni S','Fazrol R','Deddy P','Ervan A','Cipto
12         P'];
13         return view('akademik.dosen',['dosen'=>$arrDosen]);
14     }
15
16     public function show()
17     {
18         # code...
19     }
20 }
```

Karena file ini sudah di pindah, ada beberapa hal yang harus kita ubah. Pertama, penulisan namespace di baris 3 harus disesuaikan dengan alamat path yang baru, yakni menjadi:

```
namespace App\Http\Controllers\Auth;
```

Terdapat tambahan Auth di akhir namespace. Ini sesuai dengan nama folder tempat file DosenController.php berada.

Karena namespace sudah berubah, kita juga tidak bisa langsung mengakses class Controller untuk di *extends* di baris 7. Alasannya karena class Controller tersebut tersimpan di file Controller.php yang berada di namespace App\Http\Controllers, bukan App\Http\Controllers\Auth.

Solusinya, kita harus 'import' file Controller.php dengan menggunakan keyword use sebagai berikut: `use App\Http\Controllers\Controller;`

Dengan tambahan perintah ini, maka class Controller sudah kembali bisa di *extends*.

Berikut kode program lengkap dari DosenController.php setelah penambahan:



```
1 <?php
2
3 namespace App\Http\Controllers\auth;
4
5 use \App\Http\Controllers\Controller;
6 use Illuminate\Http\Request;
7
8 class DosenController extends Controller
9 {
10     public function index()
11     {
12         $arrDosen=['Ronald Hadi','Deni S','Fazrol R','Deddy P','Ervan A','Cipto P'];
13         return view('akademik.dosen',['dosen'=>$arrDosen]);
14     }
15
16     public function show()
17     {
18         # code...
19     }
20 }
21 }
```

Kemudian, bagaimana cara mengakses file ini dari route? Cukup tambah nama folder "Auth" ke dalam penulisan namespace:

APP-LARA10

views
 akademik
 layouts
 footer.blade.php
 header.blade.php
 main.blade.php
 welcome.blade.php
routes
 api.php
 channels.php
 console.php
 web.php
storage
tests
vendor
 .editorconfig
 .env
 .env.example
 .gitattributes
 .gitignore
 artisan
 composer.json
 composer.lock

OUTLINE
TIMELINE

routes > web.php > ...
5 use \App\Http\Controllers\MahasiswaController;
6 use \App\Http\Controllers\auth\DosenController;
7 /*
8 |-----
9 | Web Routes
10 |-----
11 |
12 | Here is where you can register web routes for your application. These
13 | routes are loaded by the RouteServiceProvider and all of them will
14 | be assigned to the "web" middleware group. Make something great!
15 |
16 */
17
18 Route::get('/', function () {
19 | return view('welcome');
20 |});
21
22 Route::get('/mahasiswa', [MahasiswaController::class, 'index']);
23 Route::get('/mahasiswa-show', [MahasiswaController::class, 'show']);
24
25 Route::get('/dosen', [DosenController::class, 'index']);
26