

# **Code Lab 1: Laravel Web Framework**

## **Daftar Isi**

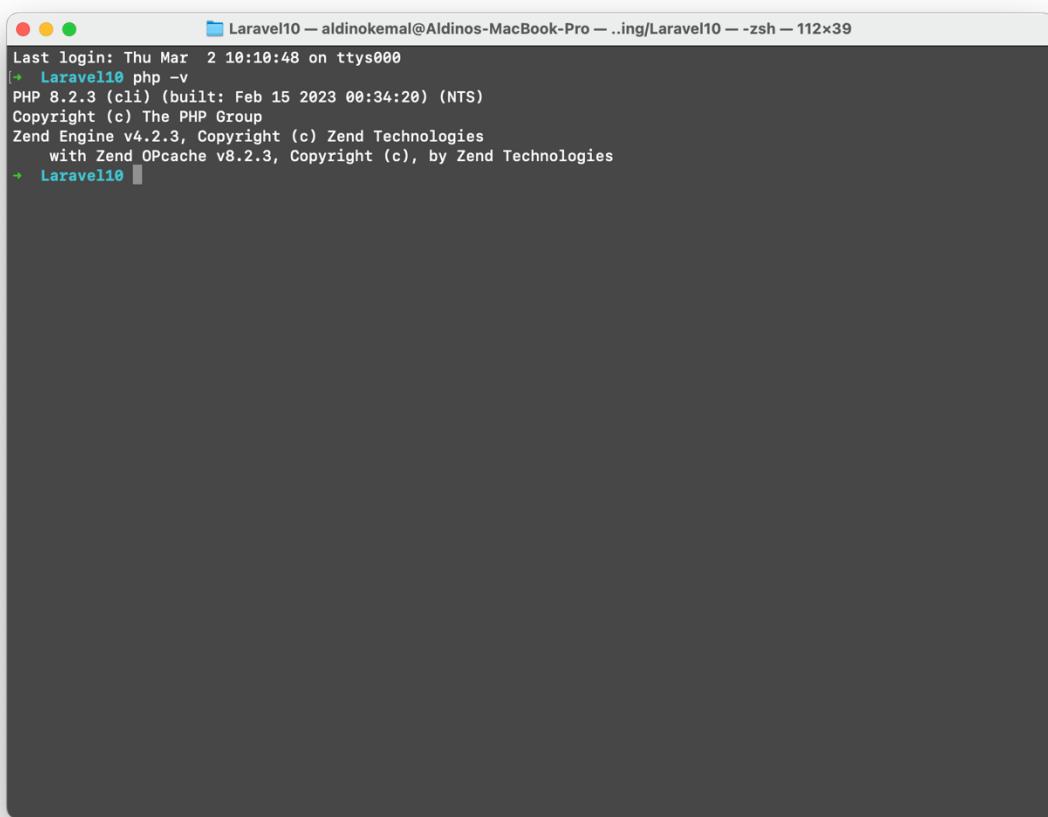
<b>Requirements.....</b>	<b>2</b>
<b>Instalasi Laravel.....</b>	<b>2</b>
<b>Mengenal Flow Laravel.....</b>	<b>5</b>
<b>Laravel Introduction .....</b>	<b>6</b>
1. <i>Laravel Routing.....</i>	<i>6</i>
2. <i>Laravel Routing + Controller + View .....</i>	<i>7</i>
3. <i>Blade Layout.....</i>	<i>9</i>
4. <i>Database Migration dan Seeder.....</i>	<i>13</i>
5. <i>Build your own Migration and Seeder.....</i>	<i>15</i>
6. <i>Laravel Middleware.....</i>	<i>19</i>

## Requirements

1. [PHP >= 8.2 \(recommend 8.3\)](#)
2. Database MySQL
3. Text Editor Canggih (recommend [VsCode](#) / [SublimeText](#) / [PhpStorm](#))
4. [Composer](#)

## Instalasi Laravel

1. Buat folder bernama Training (lokasi sesuka anda)
2. Buka CMD/Terminal kemudian arahkan ke folder Training
3. Cek apakah php sudah bisa digunakan dengan syntax berikut di CMD/Terminal. `php -v`



```
Last login: Thu Mar  2 10:10:48 on ttys000
[+ Laravel10 php -v
PHP 8.2.3 (cli) (built: Feb 15 2023 00:34:20) (NTS)
Copyright (c) The PHP Group
Zend Engine v4.2.3, Copyright (c) Zend Technologies
    with Zend OPcache v8.2.3, Copyright (c), by Zend Technologies
+ Laravel10 ]
```

4. Cek apakah git sudah terinstall pada computer kita dengan menuliskan syntax `composer`

```
[+ Laravel10 composer
-----
Composer version 2.5.4 2023-02-15 13:10:06

Usage:
  command [options] [arguments]

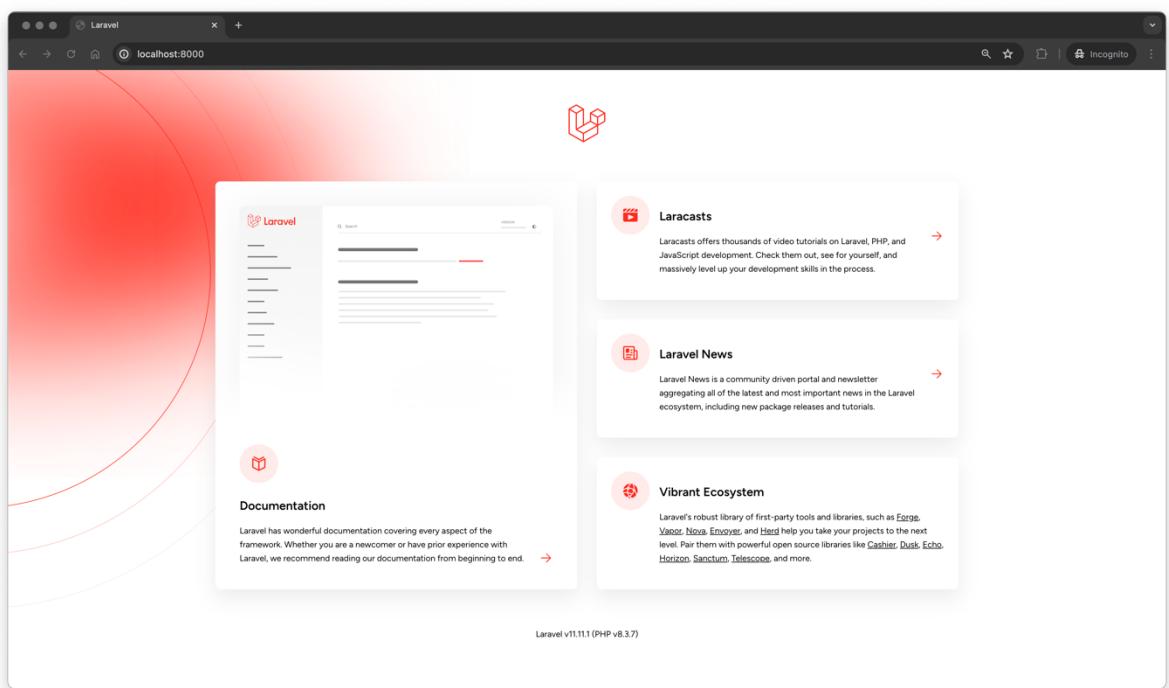
Options:
  -h, --help          Display help for the given command. When no command is given display help for t
he list command
  -q, --quiet         Do not output any message
  -V, --version       Display this application version
  --ansi|--no-ansi   Force (or disable --no-ansi) ANSI output
  -n, --no-interaction Do not ask any interactive question
  --profile          Display timing and memory usage information
  --no-plugins       Whether to disable plugins.
  --no-scripts        Skips the execution of all scripts defined in composer.json file.
  -d, --working-dir=WORKING-DIR If specified, use the given directory as working directory.
  --no-cache          Prevent use of the cache
  -v|vv|vvv|--verbose Increase the verbosity of messages: 1 for normal output, 2 for more verbose out
put and 3 for debug

Available commands:
  about              Shows a short information about Composer
  archive            Creates an archive of this composer package
  audit               Checks for security vulnerability advisories for installed packages
  browse              [homel] Opens the package's repository URL or homepage in your browser
  bump                Increases the lower limit of your composer.json requirements to the currently installed v
ersions
  check-platform-reqs Check that platform requirements are satisfied
  clear-cache        [clearcache|cc] Clears composer's internal package cache
  completion          Dump the shell completion script
  config              Sets config options
  create-project      Creates new project from a package into given directory
```

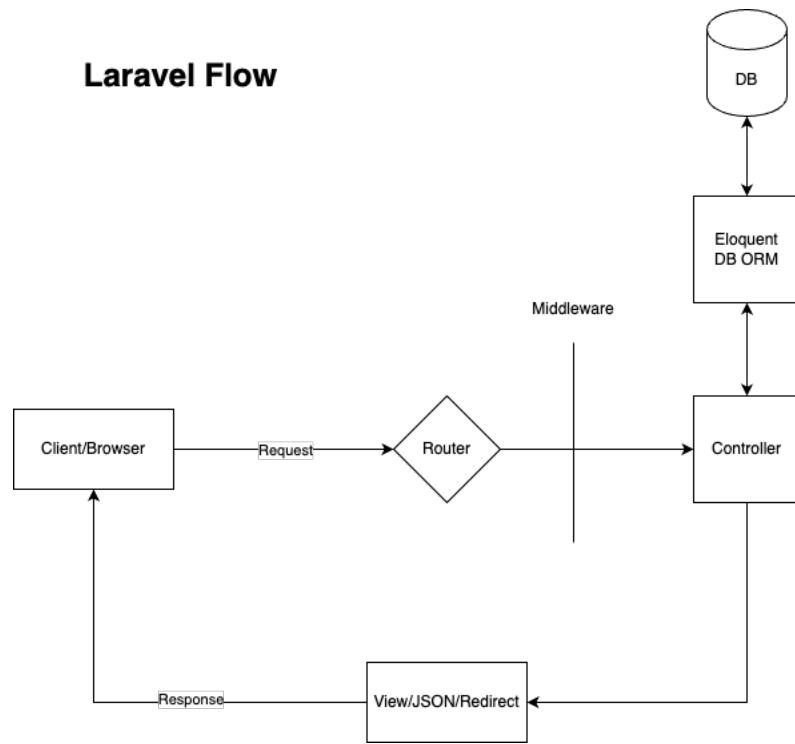
5. Untuk menginstall Laravel 11 tuliskan script berikut

```
Untuk menginstal Laravel 11 tancuh script berikut:  
composer create-project laravel/laravel:^11 Aplikasi
```

Jika sudah selesai tampilan akan seperti berikut, dan terbuat folder bernama Aplikasi. Masuk ke dalam folder aplikasi `cd Aplikasi` dan jalankan `php artisan serve`



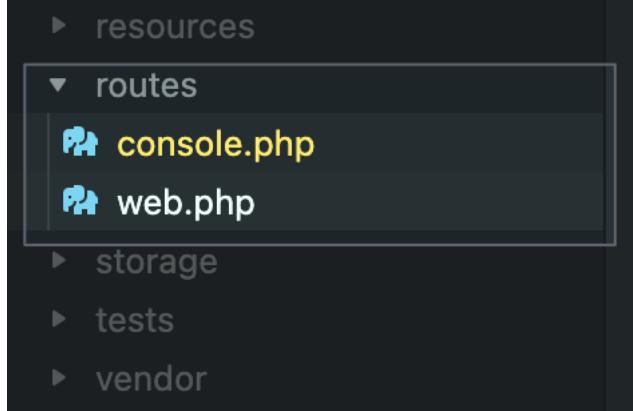
## Mengenal Flow Laravel



# Laravel Introduction

## 1. Laravel Routing

- Route pada Laravel berlokasi pada folder **routes**



- routes/**console.php** ini adalah salah satu fitur Laravel yang digunakan untuk mendaftarkan Laravel command

```
EXPLORER ... routes > console.php <?php
1 <?php
2
3 use Illuminate\Foundation\Inspiring;
4 use Illuminate\Support\Facades\Artisan;
5
6 Artisan::command('inspire', function () {
7     $this->comment(Inspiring::quote());
8 })->purpose('Display an inspiring quote')->hourly();
```

A screenshot of a code editor showing the 'console.php' file. The file contains PHP code that registers a command named 'inspire' which prints an inspiring quote. The code uses the Artisan facade to handle the command.

- routes/**web.php** merupakan routing pada Laravel yang bertujuan untuk mengatur URL,

```
EXPLORER ... routes > web.php <?php
1 <?php
2
3 use Illuminate\Support\Facades\Route;
4
5 Route::get('/', function () {
6     return view('welcome');
7 });
```

A screenshot of a code editor showing the 'web.php' file. It defines a simple route for the root URL ('/') that returns the 'welcome' view.

- Dynamic URL: Laravel memiliki fitur untuk membuat dynamic URL. Ubah script diatas menjadi seperti berikut

```
Route::get('/hello/{nama}', function($nama) {
    echo "hai $nama selamat datang di web.php";
});
```

Dan lihat hasilnya

- Grouping Route: kita dapat membuat penulisan coding menjadi sederhana dengan group route pada Laravel, sebagai contoh buat 3 routing untuk authentikasi:
  - /auth/login

- ii. /auth/register
- iii. /auth/reset-password

```
Route::get('/auth/login', function() {
    echo "halaman Login";
});
Route::get('/auth/register', function() {
    echo "halaman register";
});
Route::get('/auth/reset-password', function() {
    echo "halaman reset password";
});
```

Kita dapat mengubah penulisannya lebih ringkas menggunakan group seperti berikut

```
Route::prefix('auth')->group(function () {
    Route::get('/login', function() {
        echo "halaman Login";
    });

    Route::get('/register', function() {
        echo "halaman register";
    });

    Route::get('/reset-password', function() {
        echo "halaman reset password";
    });
});
```

## 2. Laravel Routing + Controller + View

- a. Buat controller menggunakan artisan

```
php artisan make:controller HelloController
php artisan make:controller Auth/LoginController
```

```
Laravel8/Aplikasi — php -S php artisan serve ... ~ /Data/Inixindo/Instruktur/Laravel/MateriLar
Applikasi % php artisan make:controller HelloController
/.
Applikasi % php artisan make:controller Auth/LoginController
/.
Applikasi %
```

- b. Lokasi controller berada pada app/Http/**Controllers**

- c. Menghubungkan Route dengan Controller

- a. “Import“ controller yang telah kita buat di web.php **dibawah**  
use Illuminate\Support\Facades\Route;

```
use App\Http\Controllers\HelloController;
```

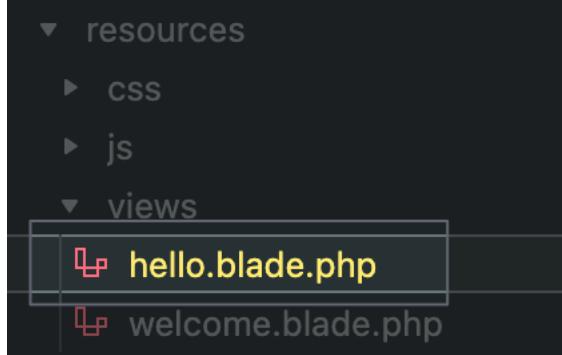
tuliskan script berikut dipaling bawah

```
Route::get("/hello-world", [HelloController::class, 'index']);
```

- b. Buat fungsi bernam index pada HelloController

```
public function index()
{
    // Parsing data ke route
    $parse = ['nama' => "Instruktur"];
    return view("hello")->with($parse);
}
```

- c. Buat file view bernama **hello.blade.php** pada folder resources/views/

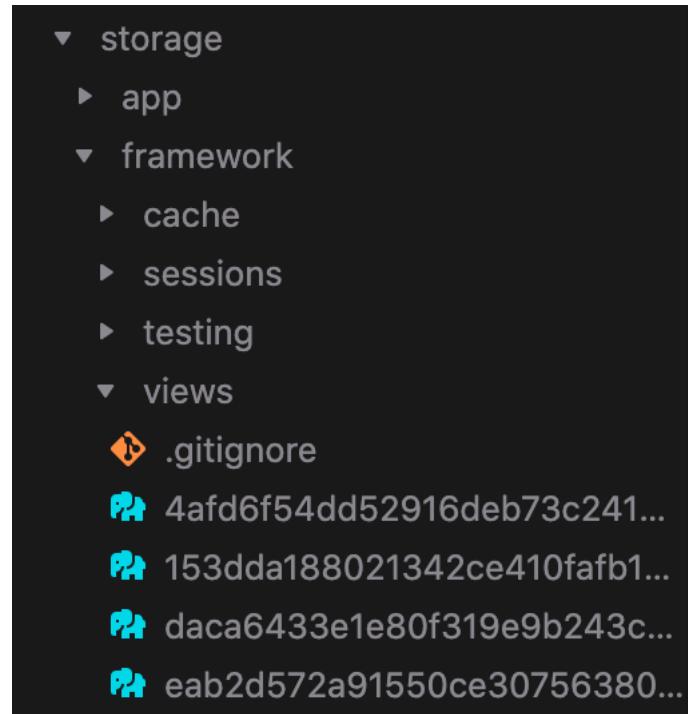


Tuliskan script seperti berikut

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <title>Hello</title>
</head>
<body>
    Halo selamat datang ! {{ $nama }}
</body>
</html>
```

### 3. Blade Layout

Laravel merupakan framework yang sangat lengkap, framework ini memiliki fitur template yang sangat powerfull dengan syarat ekstensi file harus berakhiran **.blade.php** dimana file ini nanti akan di generate oleh framework Laravel, hasil generate dapat dilihat di **storage/framework/views**



Kita sekarang akan mencoba membuat templating di Laravel

1. Buat file bernama resources/views/**layout\_auth.blade.php**, kemudian tuliskan script berikut

```
<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <title>@yield("title")</title>

    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-alpha1/dist/css/bootstrap.min.css" rel="stylesheet">

    @stack("header")
</head>

<body>
    @yield("content")

```

```

<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-alpha1/dist/js/bootstrap.bundle.min.js"></script>

    @stack("footer")
</body>

</html>

```

2. Buat file bernama resources/views/auth/**login.blade.php**, kemudian tuliskan script berikut

```

@extends("layout_auth")

@section("title", "Login")

@section("content")



<div class="row">
        <div class="col-md-4"></div>
        <div class="col-md-4">
            <h2>Login</h2>
            <form method="post" action="{{ url('auth/login') }}>
                @csrf
                <div class="mb-3">
                    <label for="exampleInputEmail1" class="form-label">Email
                    address</label>
                    <input type="email" class="form-control"
id="exampleInputEmail1" aria-describedby="emailHelp" name="email">
                    <div id="emailHelp" class="form-text">
                        We'll never share your email with anyone else.
                    </div>
                </div>
                <div class="mb-3">
                    <label for="exampleInputPassword1" class="form-
label">Password</label>
                    <input type="password" class="form-control"
id="exampleInputPassword1" name="password">
                    </div>
                    <button type="submit" class="btn btn-primary">Submit</button>
                </form>
            </div>
        </div>
    </div>


```

3. Sambungkan antara views dan controller, tambahkan fungsi berikut pada halaman Auth/**LoginController.php**

```

public function index()
{
    return view("auth.login");
}

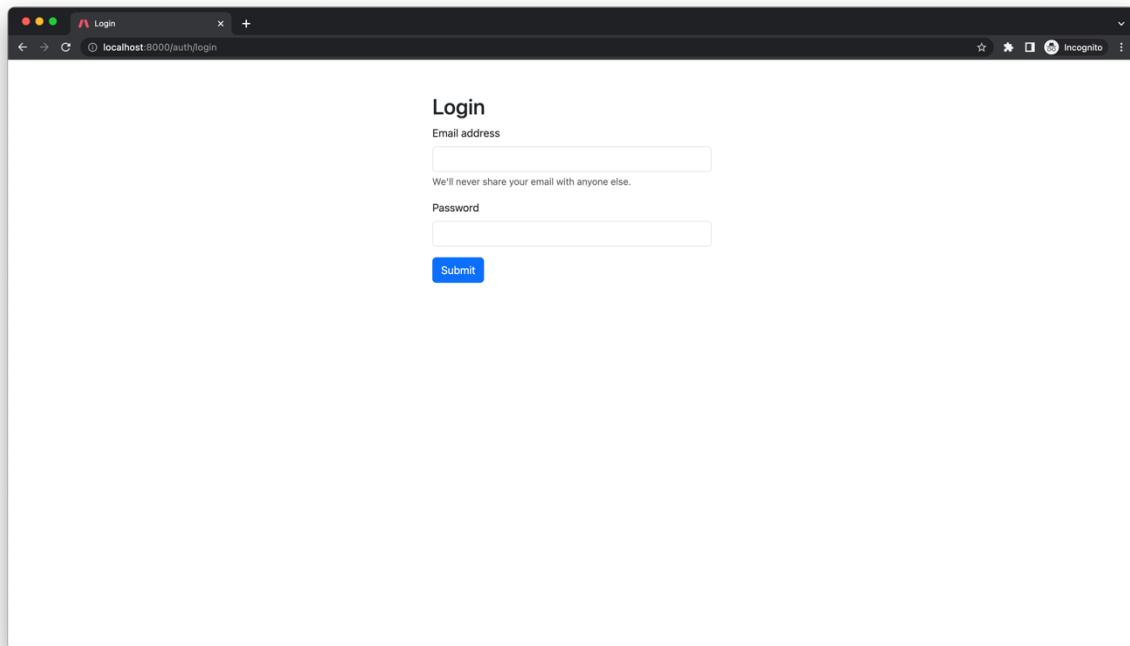
```

4. Ubah routes/**web.php**, menjadi seperti berikut

```

routes >  web.php >  Closure
1  <?php
2
3  use App\Http\Controllers\Auth\LoginController;
4  use App\Http\Controllers\HelloController;
5  use Illuminate\Support\Facades\Route;
6
7  Route::get('/', function () {
8      return view('welcome');
9  });
10
11
12 Route::get('/hello/{nama?}', function ($nama = 'Joni') {
13     return "hai $nama selamat datang di web.php";
14 });
15
16 Route::prefix('auth')->group(function () {
17     Route::get('/login', [LoginController::class, 'index']);
18
19     Route::get('/register', function() {
20         echo "halaman register";
21     });
22
23     Route::get('/reset-password', function() {
24         echo "halaman reset password";
25     });
26 });
27
28 Route::get('/hello-world', [HelloController::class, 'index']);

```



5. Menerima parameter post form

Ubah routes/**web.php**, seperti berikut:

Tambahkan route dengan method **::post** untuk memproses form data dari login

```
Route::get("/login", [LoginController::class, 'index']);
Route::post("/login", [LoginController::class, 'processLogin']);
```

6. Buat fungsi bernama processLogin pada LoginController

```
public function processLogin(Request $request)
{
    dd($request->all());
}
```

7. Ubah tag form pada login.blade.php menjadi seperti berikut

```
resources > views > auth > login.blade.php > ...
1  @extends("layout_auth")
2
3  @section("title", "Login")
4
5  @section("content")
6
7  <div class="container" style="padding-top:50px">
8  <div class="row">
9      <div class="col-md-4"></div>
10     <div class="col-md-4">
11         <h2>Login</h2>
12         <form method="post" action="{{ url('auth/login') }}>
13             @csrf
14             <div class="mb-3">
15                 <label for="exampleInputEmail1" class="form-label">Email address</label>
16                 <input type="email" class="form-control" id="exampleInputEmail1" aria-describedby="emailHelp" name="email">
17                 <div id="emailHelp" class="form-text">
18                     We'll never share your email with anyone else.
19                 </div>
20             </div>
21             <div class="mb-3">
22                 <label for="exampleInputPassword1" class="form-label">Password</label>
23                 <input type="password" class="form-control" id="exampleInputPassword1" name="password">
24             </div>
25             <button type="submit" class="btn btn-primary">Submit</button>
26         </form>
27     </div>
28 </div>
29 </div>
30
31 @endsection
```

```
array:3 [▼ // app/Http/Controllers/Auth/LoginController.php:17
  "_token" => "vECn2hGRHlhd8je6C5D7EcrKsIBFJSPSNqEOaXR"
  "email" => "admin@google.com"
  "password" => "1234"
]
```

#### 4. Database Migration dan Seeder

Secara default Laravel telah memberikan contoh penggunaan migration, seeder dalam folder database

```
▼ database
  ▼ factories
    ↗ UserFactory.php
  ▼ migrations
    ↗ 0001_01_01_000000_create_users_table.php
    ↗ 0001_01_01_000001_create_cache_table.php
    ↗ 0001_01_01_000002_create_jobs_table.php
  ▼ seeders
    ↗ DatabaseSeeder.php
```

1. Nyalakan MySQL pada local anda
2. Buat database bernama example
3. Setup connection ke database pada file `.env` ubah sesuai setting pada computer anda

```
DB_CONNECTION=mysql
DB_HOST=127.0.0.1
DB_PORT=3306
DB_DATABASE=laravel
DB_USERNAME=root
DB_PASSWORD=
```

4. Jalankan `php artisan migrate` melalui CMD/Terminal

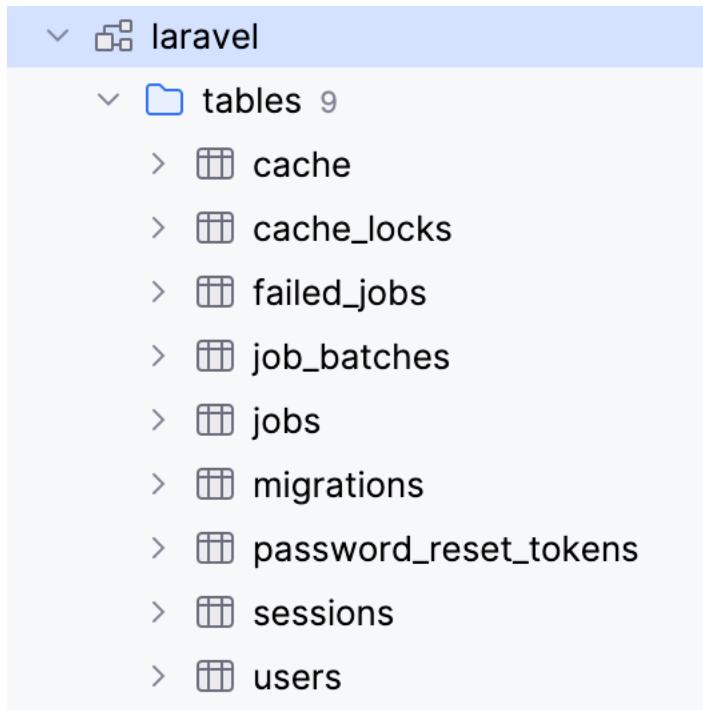
```
→ Aplikasi php artisan migrate
[INFO] Preparing database.

Creating migration table ..... 40.00ms DONE
[INFO] Running migrations.

0001_01_01_000000_create_users_table ..... 72.40ms DONE
0001_01_01_000001_create_cache_table ..... 20.28ms DONE
0001_01_01_000002_create_jobs_table ..... 46.87ms DONE

→ Aplikasi █
```

5. Buka database example



6. Berkenalan dengan Seeder: buka file bernama database/seeders/**DatabaseSeeder.php**. uncomment script yang terdapat dalam function run()

```
database > seeders > DatabaseSeeder.php > ...
1  <?php
2
3  namespace Database\Seeders;
4
5  use App\Models\User;
6  // use Illuminate\Database\Console\Seeds\WithoutModelEvents;
7  use Illuminate\Database\Seeder;
8
9  class DatabaseSeeder extends Seeder
10 {
11     /**
12      * Seed the application's database.
13      */
14     public function run(): void
15     {
16         User::factory(10)->create();
17
18         User::factory()->create([
19             'name' => 'Test User',
20             'email' => 'test@example.com',
21         ]);
22     }
23 }
24
```

7. Jalankan `php artisan migrate:refresh --seed`. Kemudian cek table user

```
→ Aplikasi php artisan migrate:refresh --seed
[INFO] Rolling back migrations.

0001_01_01_00002_create_jobs_table ..... 50.86ms DONE
0001_01_01_00001_create_cache_table ..... 17.83ms DONE
0001_01_01_00000_create_users_table ..... 25.62ms DONE

[INFO] Running migrations.

0001_01_01_00000_create_users_table ..... 59.20ms DONE
0001_01_01_00001_create_cache_table ..... 19.68ms DONE
0001_01_01_00002_create_jobs_table ..... 58.36ms DONE

[INFO] Seeding database.

→ Aplikasi
```

## 5. Build your own Migration and Seeder

1) Berikut adalah step yang harus kita lakukan

1. Buat Model Perusahaan
2. Buat Migration untuk table perusahaan
3. Buat Seeder untuk table perusahaan
4. Buat Factory (data dummy) untuk table perusahaan

Namun pada Laravel 8 keatas kita dapat langsung membuat ke 4 step tersebut menjadi 1 langkah

```
php artisan make:model Perusahaan -f -s -m
```

```
→ Aplikasi php artisan make:model Perusahaan -f -s -m
[INFO] Model [app/Models/Perusahaan.php] created successfully.
[INFO] Factory [database/factories/PerusahaanFactory.php] created successfully.
[INFO] Migration [database/migrations/2024_06_20_135132_create_perusahaans_table.php] created successfully.
[INFO] Seeder [database/seeds/PerusahaanSeeder.php] created successfully.

→ Aplikasi
```

secara default nama table di database adalah nama **model** + huruf **S**, maka jika kita membuat model perusahaan otomatis Laravel secara default akan membuat migration dengan nama **perusahaan + s = perusahaans**

ubah model Perusahaan sesuai dengan table yang kita buat, tambahkan **\$table** dan **\$primaryKey**

```
app > Models > Perusahaan.php
1  <?php
2
3  namespace App\Models;
4
5  use Illuminate\Database\Eloquent\Factories\HasFactory;
6  use Illuminate\Database\Eloquent\Model;
7
8  class Perusahaan extends Model
9  {
10     use HasFactory;
11
12     protected $table = "perusahaan";
13     protected $primaryKey = "perusahaan_id";
14 }
15
```

kemudian ubah script migration pada function UP()

```
public function up()
{
    Schema::create('perusahaan', function (Blueprint $table) {
        $table->increments("perusahaan_id", 11);
        $table->string("perusahaan_nama", 100);
        $table->string("perusahaan_owner", 100);
        $table->text("perusahaan_alamat");
        $table->string("perusahaan_email")->nullable();
        $table->string("perusahaan_phone")->nullable();
        $table->timestamps();
    });
}
```

Ubah fungsi down() menjadi seperti berikut

```
public function down()
{
    Schema::dropIfExists('perusahaan');
}
```

Kemudian jalankan `php artisan migrate:refresh --seed`. Lihat table baru

perusahaan

columns 8

- perusahaan\_id int unsigned (auto increment)
- perusahaan\_nama varchar(100)
- perusahaan\_owner varchar(100)
- perusahaan\_alamat text
- perusahaan\_email varchar(255)
- perusahaan\_phone varchar(255)
- created\_at timestamp
- updated\_at timestamp

2) Membuat custom seeder pada perusahaan

```
database > seeders > PerusahaanSeeder.php
1  <?php
2
3  namespace Database\Seeders;
4
5  use App\Models\Perusahaan;
6  use Illuminate\Database\Seeder;
7
8  class PerusahaanSeeder extends Seeder
9  {
10
11     /**
12      * Run the database seeds.
13      *
14      * @return void
15      */
16     public function run()
17     {
18         Perusahaan::factory(10)->create();
19     }
20 }
```

3) Membat factory pada perusahaan

```
public function definition(): array
{
    return [
        "perusahaan_nama" => fake()>company,
        "perusahaan_owner" => fake()>name,
        "perusahaan_alamat" => fake()>address,
        "perusahaan_email" => fake()>email,
        "perusahaan_phone" => fake()>tollFreePhoneNumber,
    ];
}
```

4) Panggil faker dengan menambahkan berikut pada **DatabaseSeeder.php**

```
$this->call(PerusahaanSeeder::class);
```

5) Jalankan **php artisan migrate:refresh --seed**

	perusahaan_id	perusahaan_nama	perusahaan_owner	perusahaan_alamat	perusahaan_email
1	1	Blanda Ltd	Angeline Okuneva	718 Heathcote Curve Apt. 800 So...	kristofer92@h...
2	2	Lubowitz-White	Curtis Homenick	720 Thiel Loaf West Otilialand,...	stroman.shyan...
3	3	Cummerata-Lubowitz	Tomas Dickinson	43493 Bergnaum Square Heidenrei...	cole.tessie@m...
4	4	Hayes Ltd	Prudence Bednar	4263 Gutmann Junctions Suite 94...	nbradtke@yaho...
5	5	Christiansen-Becker	Dr. Holden Hintz	2811 Schumm Lock North Addison,...	jodie.huels@z...
6	6	Wilkinson, Heaney and Kub	Rachael Hyatt	82836 Raphael Branch Suite 303 No...	jimmy.kuphal@...
7	7	Kuvalis LLC	Yolanda Robel	41430 Tillman Hills Apt. 150 No...	jaycee.wilkin...
8	8	Flatley, Kub and Satterfield	Ariane Cremin V	76167 Francisco Villages Lake G...	giuseppe.yund...
9	9	Stanton, Beatty and Gottlieb	Hallie Okuneva	935 Delilah Ridge Flatleymouth,...	pdietrich@sha...
10	10	Nitzsche-Schinner	Brennon Anderson	69937 Herzog Road Apt. 822 Port...	frederique48@...

## 6. Laravel Middleware

Seperti pada gambar diatas pertama, Middleware menyediakan mekanisme yang mudah untuk memfilter permintaan HTTP yang memasuki aplikasi Anda.

Custom middleware pada Laravel terletak pada app/Http/**Middleware.php**. mari kita coba buat custom middleware dengan Langkah berikut

1. Buat middleware melalui CMD/Terminal

```
php artisan make:middleware ExampleMiddleware
```

Maka akan terbuat file pada folder middleware

ubah fungsi handle menjadi seperti berikut

```
public function handle(Request $request, Closure $next): Response
{
    dd("Ini Example Middleware");
    return $next($request);
}
```

2. Pasangkan middleware pada **web.php**

```
use App\Http\Controllers\Auth\LoginController;
use App\Http\Controllers\HelloController;
use App\Http\Middleware\ExampleMiddleware;
use Illuminate\Support\Facades\Route;

Route::get('/', function () {
    return view('welcome');
});

Route::get('/hello/{nama?}', function ($nama = 'Joni') {
    return "hai $nama selamat datang di web.php";
});

Route::prefix('auth')->middleware(ExampleMiddleware::class)->group(function () {
    Route::get('/login', [LoginController::class, 'index']);

    Route::get('/register', function() {
        echo "halaman register";
    });

    Route::get('/reset-password', function() {
        echo "halaman reset password";
    });
});
```