# Code Lab 1: Laravel Web Framework

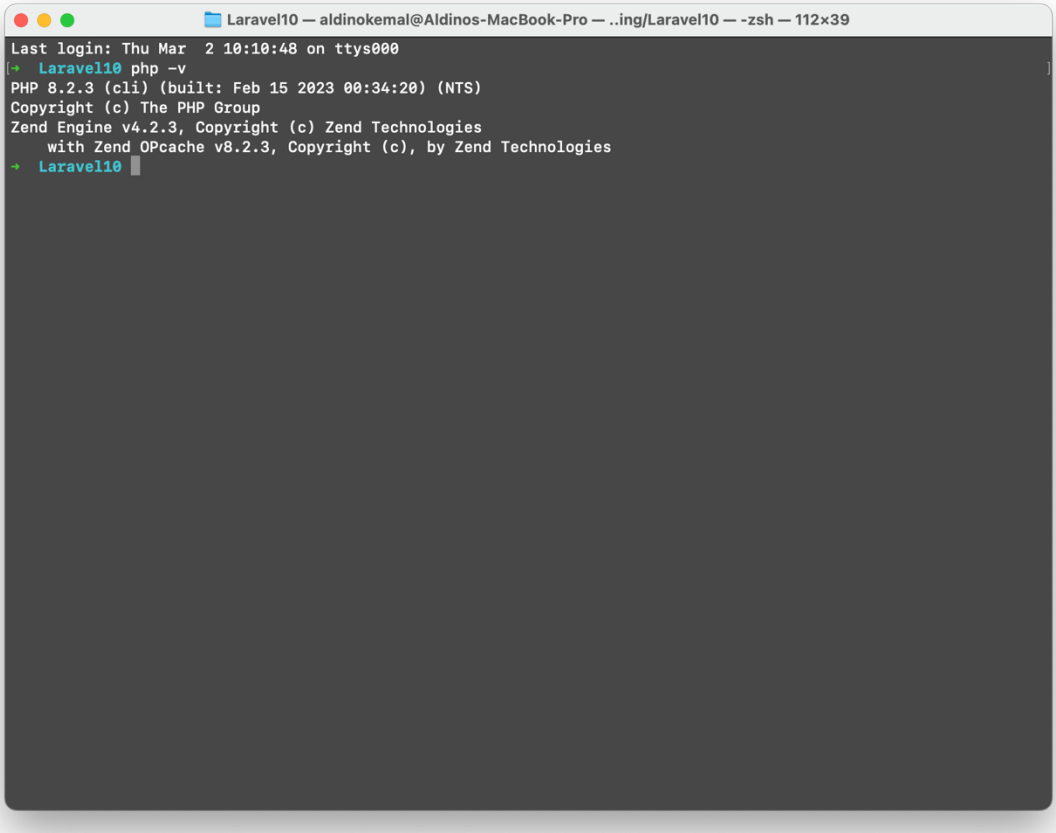
## Daftar Isi

## Requirements

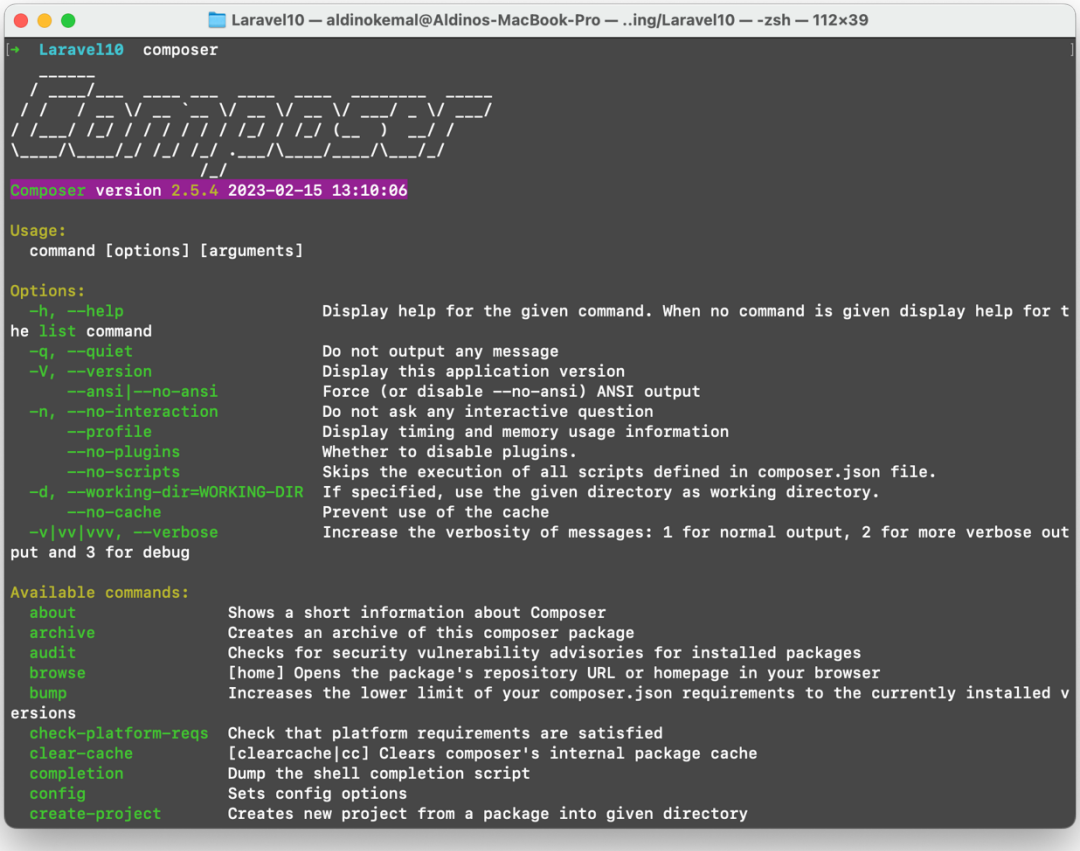
1. PHP >= 8.2 (recommend 8.3)
2. Database MySQL
3. Text Editor Canggih (recommend VsCode / SublimeText / PhpStorm)
4. Composer

## Instalasi Laravel

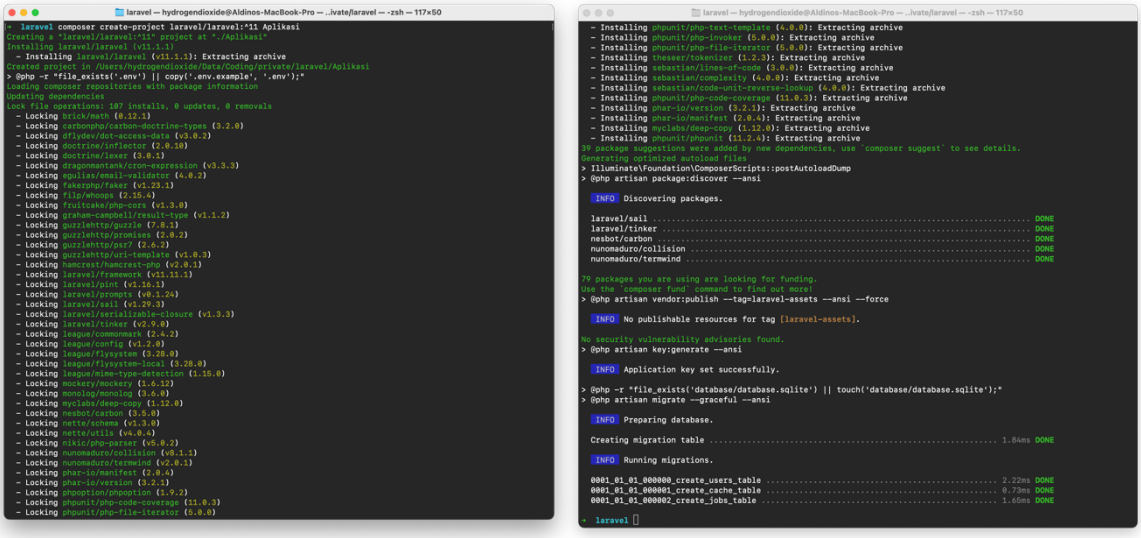
1. Buat folder bernama Training (lokasi sesuka anda)
2. Buka CMD/Terminal kemudian arahkan ke folder Training
3. Cek apakah php sudah bisa digunakan dengan syntax berikut di CMD/Terminal. `php -v`



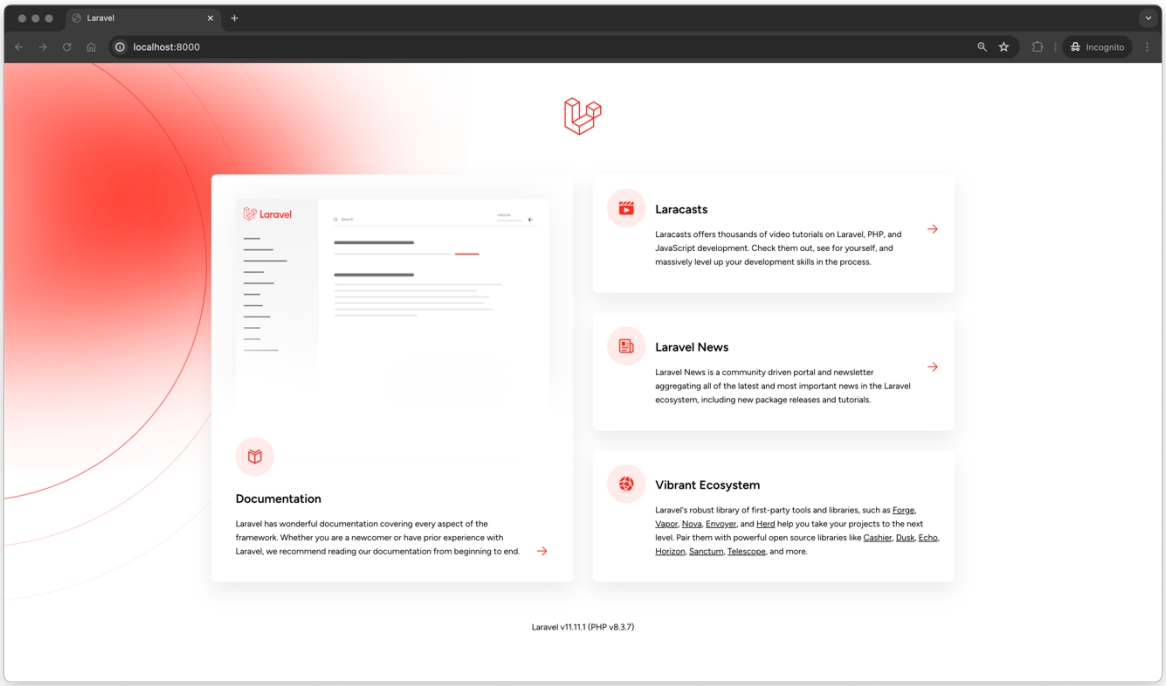

4. Cek apakah git sudah terinstall pada computer kita dengan menuliskan syntax `composer`



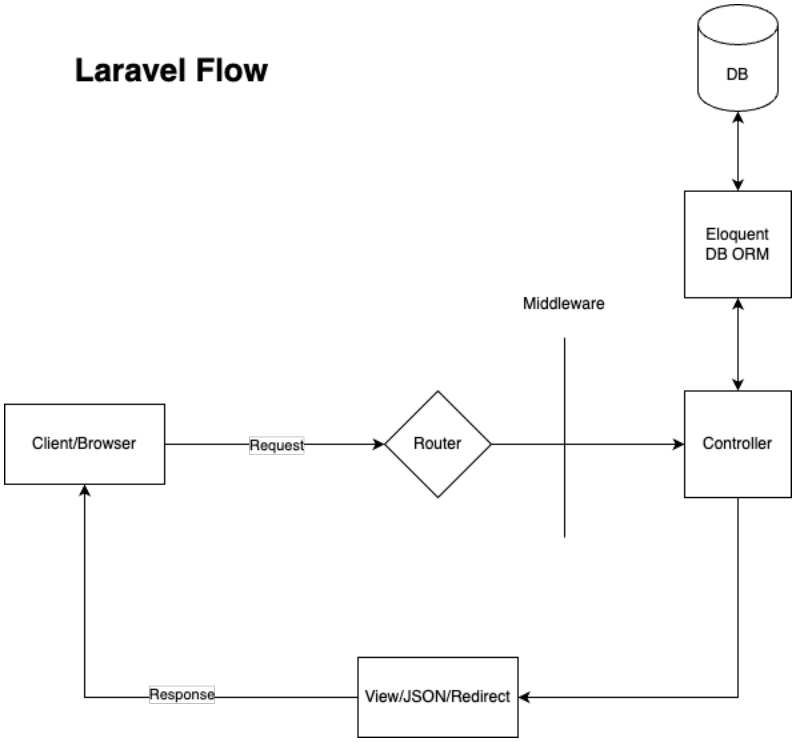

5. Untuk menginstall Laravel 11 tuliskan script berikut
`composer create-project laravel/laravel:^11 Aplikasi`



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


Laravel
localhost:8000
Incognito
Laravel
Documentation
Laravel has wonderful documentation covering every aspect of the framework. Whether you are a newcomer or have prior experience with Laravel, we recommend reading our documentation from beginning to end.
Laracasts
Laracasts offers thousands of video tutorials on Laravel, PHP, and JavaScript development. Check them out, see for yourself, and massively level up your development skills in the process.
Laravel News
Laravel News is a community driven portal and newsletter aggregating all of the latest and most important news in the Laravel ecosystem, including new package releases and tutorials.
Vibrant Ecosystem
Laravel's robust library of first-party tools and libraries, such as Forge, Vapor, Nova, Envoyer, and Herd help you take your projects to the next level. Pair them with powerful open source libraries like Cashier, Dusk, Echo, Horizon, Sanctum, Telescope, and more.
Laravel v11.11.1 (PHP v8.3.7)

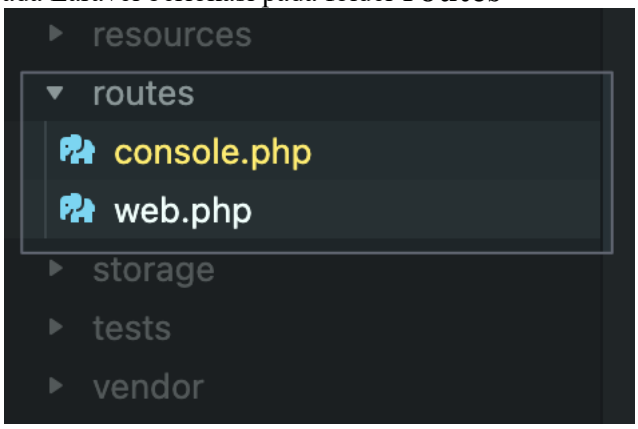
## Mengenal Flow Laravel



# Laravel Introduction

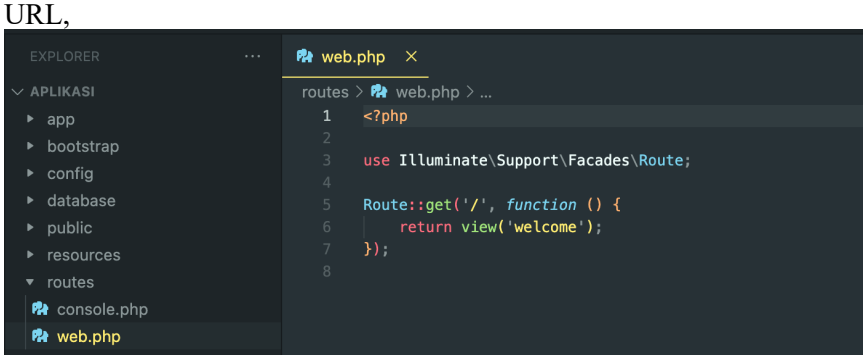
## 1. Laravel Routing

a. Route pada Laravel berlokasi pada folder **routes**




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


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




d. 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

e. Grouping Route: kita dapat membuat penulisan coding menjadi sederhana dengan group route pada Laravel, sebagai contoh buat 3 routing untuk authentikasi:

   i. /auth/login

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

```php
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";
});
```

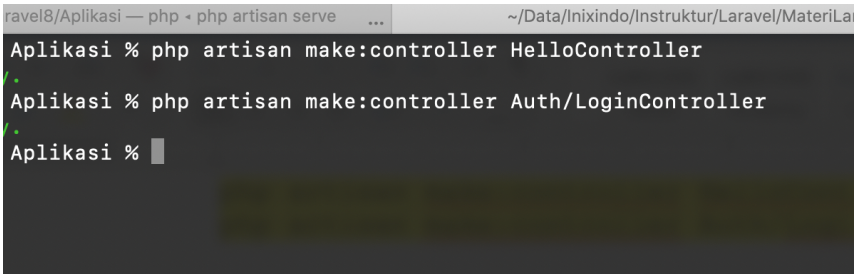
Kita dapat mengubah penulisannya lebih ringkas menggunakan group seperti berikut

```php
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
```




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;
```

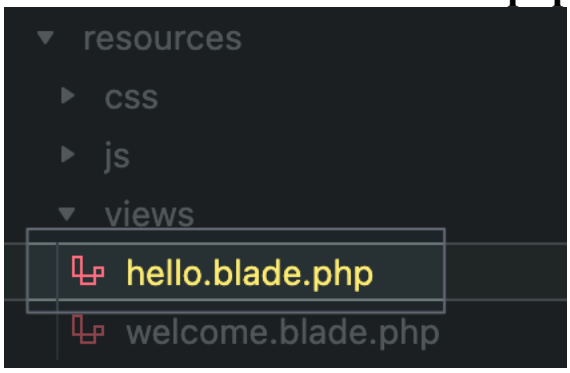
tuliskan script berikut dipaling bawah

```
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/

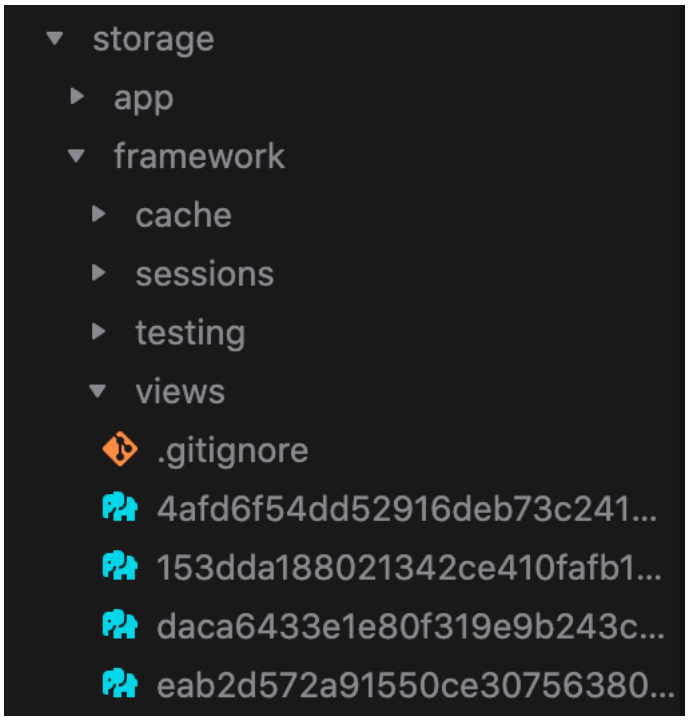



Tuliskan script seperti berikut

```
<!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="container" style="padding-top:50px">
    <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>

@endsection
```

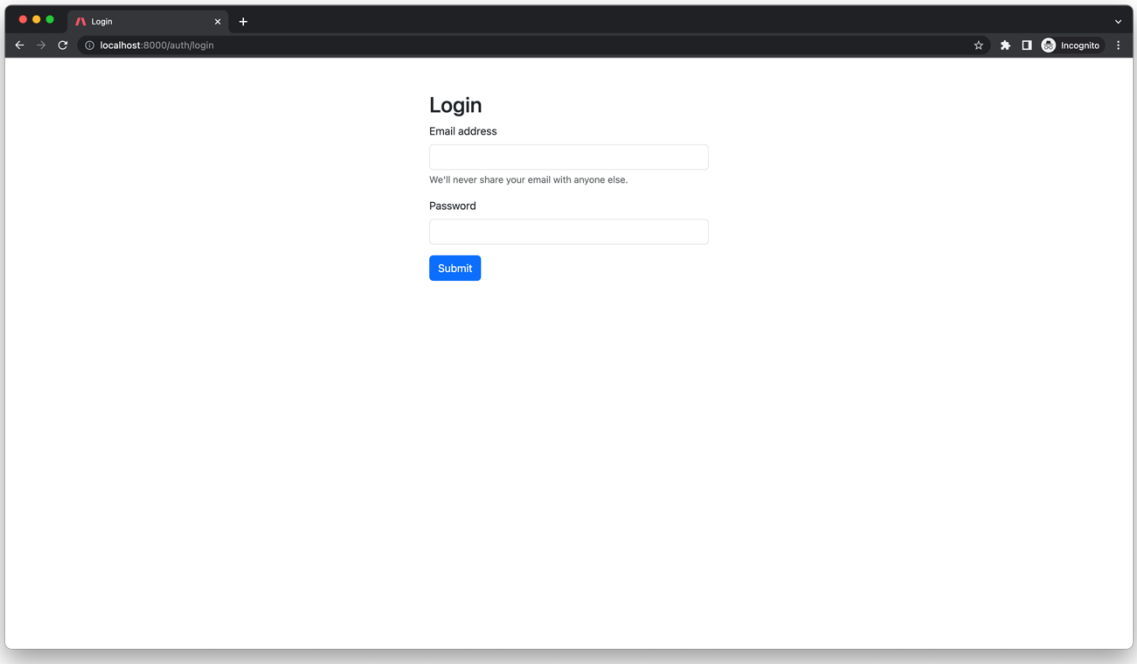
3. Sambungkan antara views dan controller, tambahkan fungsi berikut pada halaman Auth/**LoginController.php**

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

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

routes > web.php > Closure

```php
<?php

use App\Http\Controllers\Auth\LoginController;
use App\Http\Controllers\HelloController;
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')->group(function () {
    Route::get('/login', [LoginController::class, 'index']);

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

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

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

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

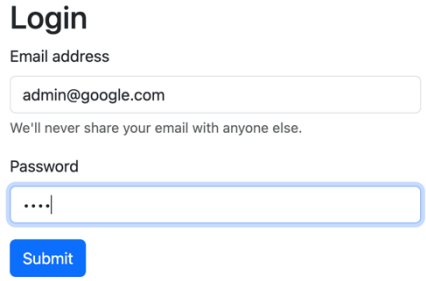
6. Buat fungsi bernama processLogin pada LoginController

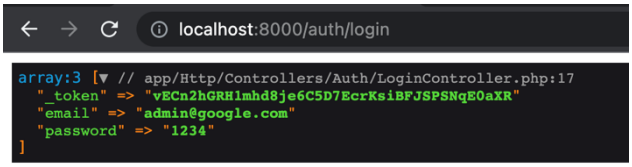
```php
public function processLogin(Request $request)
    {
        dd($request->all());
    }
```

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

resources > views > auth > login.blade.php > ...

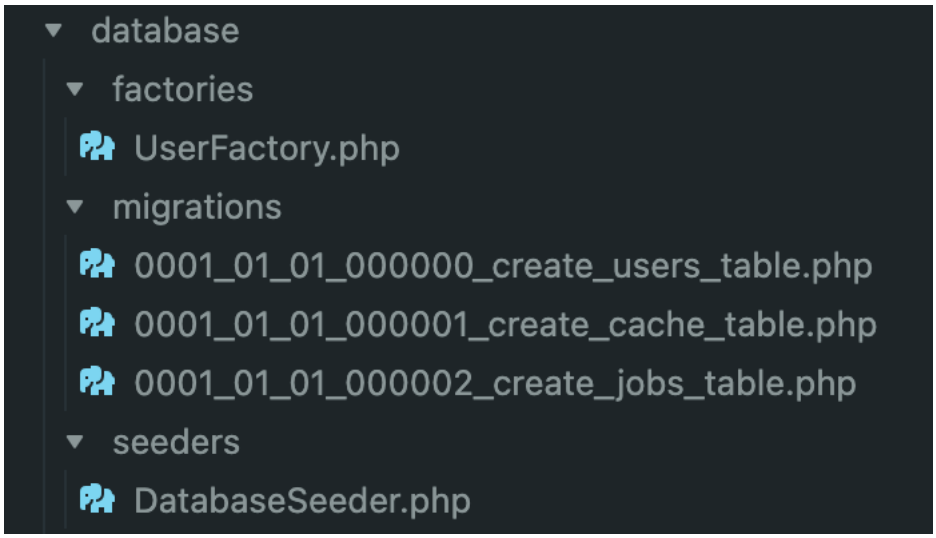
```blade
@extends("layout_auth")

@section("title", "Login")

@section("content")

<div class="container" style="padding-top:50px">
    <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>

@endsection
```



## 4. Database Migration dan Seeder

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



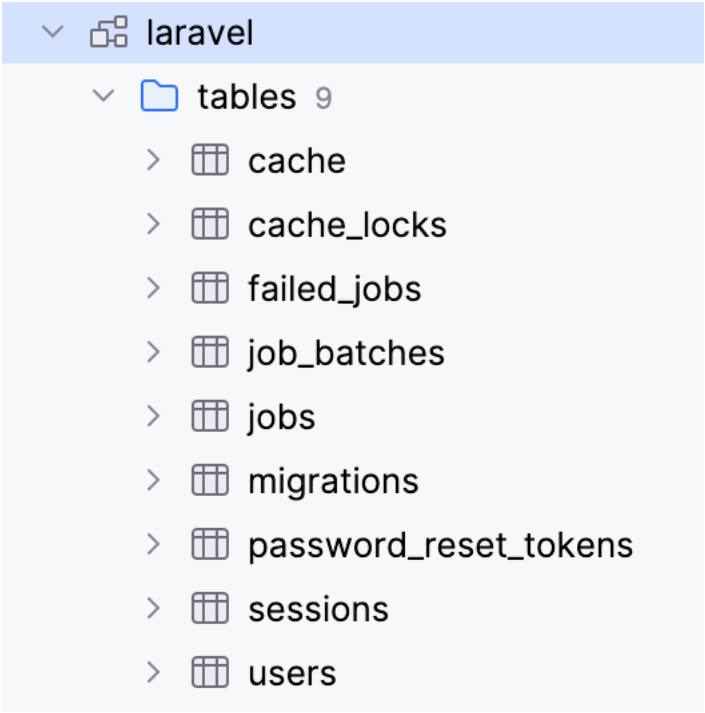

1. Nyalakan MySQL pada local anda
2. Buat database bernama example
3. Setup connection ke database pada file **.env** ubah sesuai setting pada computer anda

```
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 > ...
<?php

namespace Database\Seeders;

use App\Models\User;
// use Illuminate\Database\Console\Seeds\WithoutModelEvents;
use Illuminate\Database\Seeder;

class DatabaseSeeder extends Seeder
{
    /**
     * Seed the application's database.
     */
    public function run(): void
    {
        User::factory(10)->create();

        User::factory()->create([
            'name' => 'Test User',
            'email' => 'test@example.com',
        ]);
    }
}
```

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

```
→  Aplikasi php artisan migrate:refresh --seed

  INFO  Rolling back migrations.

  0001_01_01_000002_create_jobs_table ..... 50.86ms DONE
  0001_01_01_000001_create_cache_table ..... 17.83ms DONE
  0001_01_01_000000_create_users_table ..... 25.62ms DONE


  INFO  Running migrations.

  0001_01_01_000000_create_users_table ..... 59.20ms DONE
  0001_01_01_000001_create_cache_table ..... 19.68ms DONE
  0001_01_01_000002_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/seeders/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

```php
<?php

namespace App\Models;

use Illuminate\Database\Eloquent\Factories\HasFactory;
use Illuminate\Database\Eloquent\Model;

class Perusahaan extends Model
{
    use HasFactory;

    protected $table = "perusahaan";
    protected $primaryKey = "perusahaan_id";
}
```

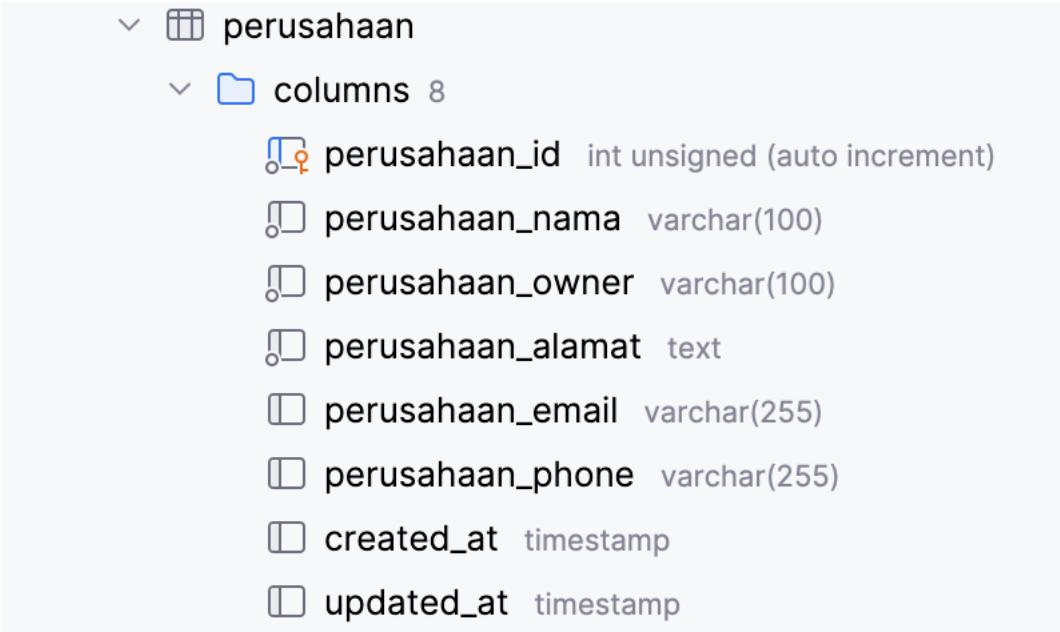
kemudian ubah script migration pada function UP()

```php
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();
    });
}
```

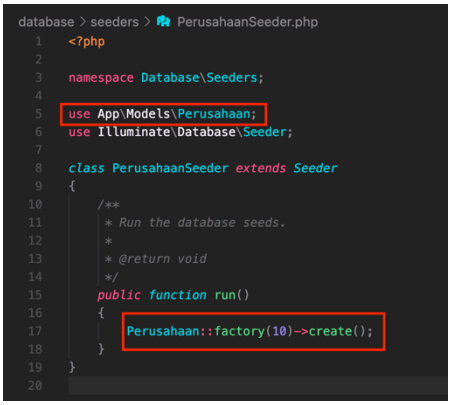
Ubah fungsi down() menjadi seperti berikut

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

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




2) Membuat custom seeder pada perusahaan



3) Membat factory pada perusahaan

```php
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**

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

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

perusahaan [[lokal] Localhost]

10 rows | Tx: Auto | DDL | CSV

WHERE | ORDER BY

| | perusahaan_id | perusahaan_nama | perusahaan_owner | perusahaan_alamat | perusahaan_ |
|---|---|---|---|---|---|
| 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 | Tomasa 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⏎… | 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 gamabar 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

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

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

```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";
    });
});
```