

Pemrograman Web Lanjut
Jobsheet 2 - Routing, Controller, dan View

Dosen Pengampu : Dimas Wahyu Wibowo, S.T., M.T.



Nama : Athallah Ayudya Paramesti
NIM : 2341760061
Program Studi : D-IV Sistem Informasi Bisnis

Jurusan Teknologi Informasi
Politeknik Negeri Malang
2025

1. MVC pada Laravel

MVC merupakan singkatan dari Model View Controller. Laravel menggunakan model MVC, oleh karena itu ada tiga bagian inti dari framework yang bekerja bersama: model, view, dan controller. Controller adalah bagian utama di mana sebagian besar pekerjaan dilakukan. Controller terhubung ke Model untuk mendapatkan, membuat, atau memperbarui data dan menampilkan hasilnya pada View, yang berisi struktur HTML aktual dari aplikasi.

a. Model

Dalam Laravel, kelas Model berisi semua metode dan atribut yang diperlukan untuk berinteraksi dengan skema database yang ditentukan.

b. View

View mewakili bagaimana informasi ditampilkan, digunakan untuk semua logika antarmuka pengguna perangkat lunak. View mewakili Antarmuka Pengguna (Frontend) dari halaman web.

c. Controller

Controller berperan sebagai perantara antara Model dan View, memproses semua masukan yang dikirim oleh pengguna dari View. Controller memproses semua logika bisnis, memanipulasi data menggunakan komponen Model, dan berinteraksi dengan View untuk merender output akhir.

2. Routing

Pada Laravel terdapat fitur yang bernama route. Route ini digunakan sebagai penghubung antara user dengan aplikasi. Dengan kata lain, URL yang kita tulis di dalam browser akan melewati route. Dan pada route tersebut akan ditentukan kemana selanjutnya, bisa ke Controller atau ke View. Routing sendiri adalah proses pengiriman data maupun informasi ke pengguna melalui sebuah permintaan yang dilakukan kepada alamat yang sudah terdaftar, lalu alamat tersebut akan memproses dari permintaan kita tadi. Setelah proses selesai maka akan mengembalikan sebuah output atau hasil dari proses tersebut.

Untuk membuat route digunakan Facade Route diikuti dengan verb yang merupakan HTTP verb, umumnya terdiri dari get, post, put, delete, options, patch. Selain itu dibutuhkan path yang berupa URL setelah nama domain aplikasi yang diakses oleh pengguna. Dan pada bagian akhir terdapat callback yang dapat berupa callback function atau controller action yang menjalankan logika ketika path diakses oleh pengguna.

- Basic Routing

Pada dasarnya Routing di Laravel membutuhkan informasi mengenai http verb kemudian input berupa url dan apa yang harus dilakukan ketika menerima url tersebut. Untuk membuat sebuah route anda dapat menggunakan callback function atau menggunakan sebuah controller.

Langkah-langkah Praktikum:

- a. Membuat dua buah route dengan ketentuan sebagai berikut.

No	Http Verb	Url	Fungsi
1	get	/hello	Tampilkan String Hello ke browser.
2	get	/world	Tampilkan String World ke browser

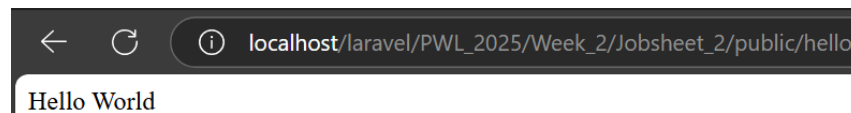
gunakan project minggu sebelumnya yaitu PWL_2025.

- b. Buka file routes/web.php. Tambahkan sebuah route untuk nomor 1 seperti di bawah ini:

```
use Illuminate\Support\Facades\Route;

Route::get('/hello', function () {
    return 'Hello World';
});
```

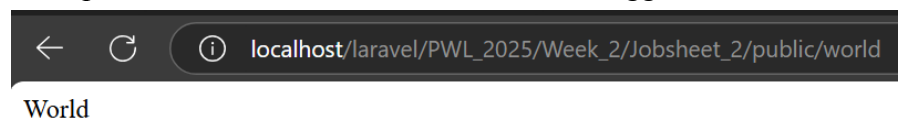
- c. Buka browser, tuliskan URL untuk memanggil route tersebut:



- d. Untuk membuat route kedua, tambahkan route /world seperti di bawah ini:

```
Route::get('/world', function () {
    return 'World';
});
```

- e. Buka pada browser, tuliskan URL untuk memanggil route tersebut:



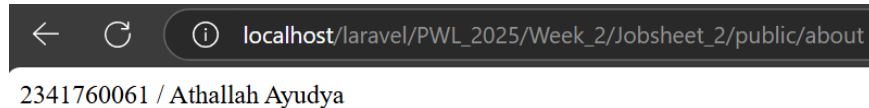
- f. Berikut merupakan route '/' yang menampilkan pesan 'Selamat Datang'.

```
Route::get('/', function () {
    return 'Selamat Datang';
});
```



- g. Berikut merupakan route '/about' yang menampilkan NIM dan nama.

```
Route::get('/about', function () {  
    return '2341760061 / Athallah Ayudya';  
});
```



- Route Parameters

Terkadang saat membuat sebuah URL, kita perlu mengambil sebuah parameter yang merupakan bagian dari segmen URL dalam route kita. Misalnya, kita membutuhkan nama user yang dikirim melalui sebuah URL.

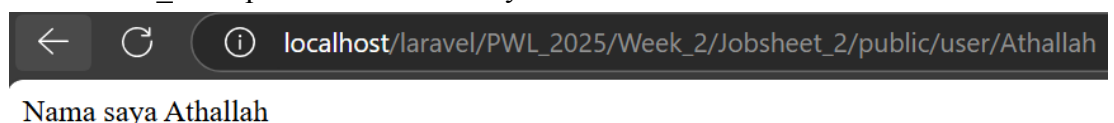
Langkah-langkah Praktikum:

Untuk membuat routing dengan parameter dapat dilakukan dengan cara berikut ini.

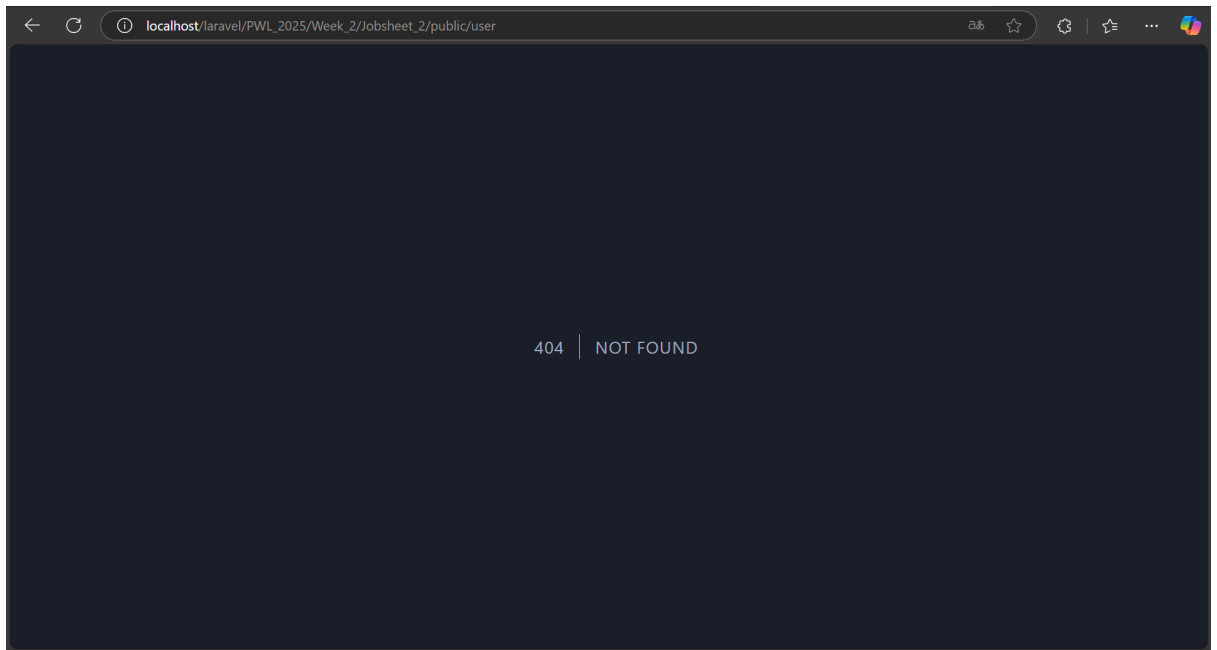
- a. Memanggil route /user/{name} sekaligus mengirimkan parameter berupa nama user \$name seperti kode di bawah ini.

```
Route::get('/user/{name}', function ($name) {  
    return 'Nama saya '.$name;  
});
```

- b. Jalankan kode dengan menuliskan URL untuk memanggil route tersebut: localhost/PWL_2025/public/user>NamaSaya.



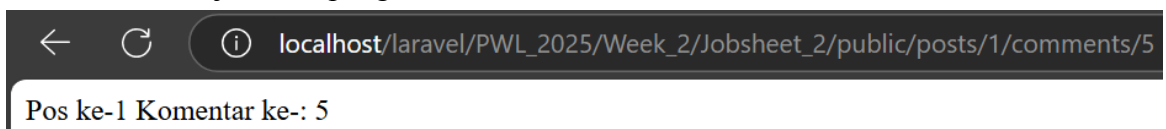
- c. Selanjutnya, coba tuliskan URL: localhost/PWL_2024/public/user/. Perhatikan halaman yang muncul dan jelaskan pengamatan Anda



- d. Suatu route, juga bisa menerima lebih dari 1 parameter seperti kode berikut ini. Route menerima parameter \$postId dan juga \$comment.

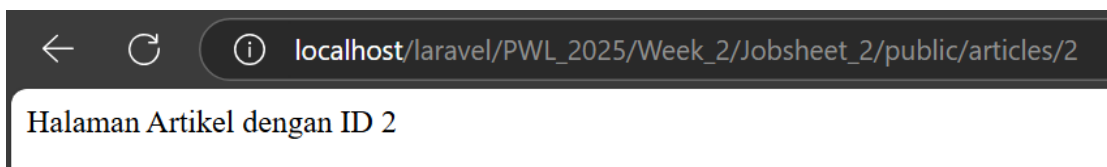
```
Route::get('/posts/{post}/comments/{comment}', function ($postId, $commentId) {  
    return 'Pos ke-' . $postId. " Komentar ke-: " . $commentId;  
});
```

- e. Jalankan kode dengan menuliskan URL untuk memanggil route tersebut: localhost/PWL_2024/public/posts/1/comments/5. Perhatikan halaman yang muncul dan jelaskan pengamatan Anda.



- f. Kemudian buatlah route /articles/{id} yang akan menampilkan output “Halaman Artikel dengan ID {id}”, ganti id sesuai dengan input dari url.

```
Route::get('/articles/{id}', function ($id) {  
    return 'Halaman Artikel dengan ID ' . $id;  
});
```



- Optional Parameter

Kita dapat menentukan nilai parameter route, tetapi menjadikan nilai parameter route tersebut opsional. Pastikan untuk memberikan variabel yang sesuai pada route sebagai nilai default. Parameter opsional diberikan tanda '?'.

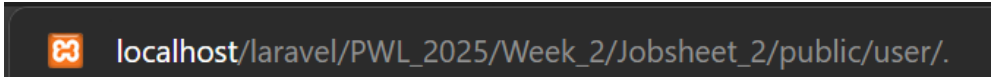
Langkah-langkah Praktikum:

Untuk membuat routing dengan optional parameter dapat dilakukan dengan cara berikut ini.

- a. Kita akan memanggil route /user sekaligus mengirimkan parameter berupa nama user \$name dimana parameternya bersifat opsional.

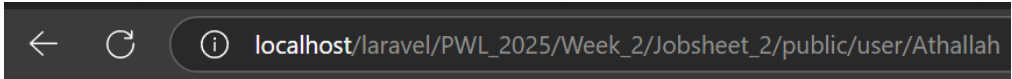
```
Route::get('/user/{name?}', function ($name=null){  
    return 'Nama saya '.$name;  
});
```

- b. Jalankan kode dengan menuliskan URL: localhost/PWL_2024/public/user/. Perhatikan halaman yang muncul dan jelaskan pengamatan Anda.



localhost/laravel/PWL_2025/Week_2/Jobsheet_2/public/user/.

- c. Selanjutnya tuliskan URL: localhost/PWL_2024/public/user>NamaAnda. Perhatikan halaman yang muncul dan jelaskan pengamatan Anda




localhost/laravel/PWL_2025/Week_2/Jobsheet_2/public/user/Athallah

Nama saya Athallah

- d. Ubah kode pada route /user menjadi seperti di bawah ini.

```
Route::get('/user/{name?}', function ($name='John') {  
    return 'Nama saya '.$name;  
});
```

- e. Jalankan kode dengan menuliskan URL: localhost/PWL_2025/public/user/. Perhatikan halaman yang muncul dan jelaskan pengamatan Anda.



localhost/laravel/PWL_2025/Week_2/Jobsheet_2/public/user

Nama saya John

- Route Name

Route name biasanya digunakan untuk mempermudah kita dalam pemanggilan route saat membangun aplikasi. Kita cukup memanggil name dari route tersebut.

- Route Group dan Route Prefixes

Beberapa route yang memiliki atribut yang sama seperti middleware yang sama dapat dikelompokkan menjadi satu kelompok untuk mempermudah penulisan route selain digunakan untuk middleware masih ada lagi penggunaan route group untuk route yang berada dibawah satu subdomain.

Route Prefixes

Pengelompokan route juga dapat dilakukan untuk route yang memiliki prefix (awalan) yang sama.

- Redirect Routes

Untuk melakukan redirect pada laravel dapat dilakukan dengan menggunakan `Route::redirect`. Redirect ini akan sering digunakan pada kasus kasus CRUD atau kasus lain yang membutuhkan redirect.

- View Routes

Laravel juga menyediakan sebuah route khusus yang memudahkan dalam membuat sebuah routes tanpa menggunakan controller atau callback function. Routes ini langsung menerima input berupa url dan mengembalikan view / tampilan. Berikut ini cara membuat view routes.

3. Controller

Controller digunakan untuk mengorganisasi logika aplikasi menjadi lebih terstruktur. Logika action aplikasi yang masih ada kaitan dapat dikumpulkan dalam satu kelas Controller. Atau sebuah Controller dapat juga hanya berisi satu buah action. Controller pada Laravel disimpan dalam folder `app/Http/Controllers`.

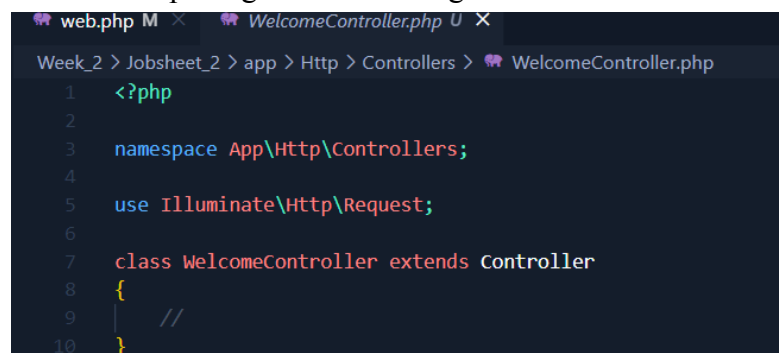
- Membuat Controller

Langkah-langkah Praktikum:

- a. Untuk membuat controller pada Laravel telah disediakan perintah untuk menggenerate struktur dasarnya. Kita dapat menggunakan perintah `artisan` diikuti dengan definisi nama controller yang akan dibuat.

```
INFO Controller [C:\laragon\www\laravel\PWL_2025\Week_2\Jobsheet_2\app\Http\Controllers>WelcomeController.php] created successfully.
```

- b. Buka file pada `app/Http/Controllers/WelcomeController.php`. Struktur pada controller dapat digambarkan sebagai berikut:



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

- c. Untuk mendefinisikan action, silahkan tambahkan function dengan access public. Sehingga controller di atas menjadi sebagai berikut:

```
<?php

namespace App\Http\Controllers;

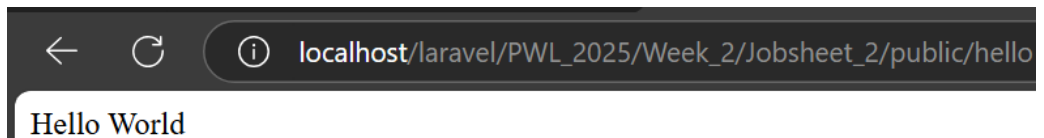
use Illuminate\Http\Request;

class WelcomeController extends Controller
{
    public function hello() {
        return 'Hello World';
    }
}
```

- d. Setelah sebuah controller telah didefinisikan action, kita dapat menambahkan controller tersebut pada route. Ubah route /hello menjadi seperti berikut:

```
Route::get('/hello', [WelcomeController::class, 'hello']);
```

- e. Buka browser, tuliskan URL untuk memanggil route tersebut:



- f. Hasil modifikasi praktikum poin 2 (Routing) dengan konsep Controller.

Resource	POST	GET	PUT	DELETE
/		Tampilkan Pesan 'Selamat Datang' PageController : index		
/about		Tampilkan Nama dan NIM PageController : about		
/articles/ {id}		Tampilkan halaman dinamis 'Halaman Artikel dengan Id {id}' id diganti sesuai input dari url PageController : articles		

Resource “ / ”

web.php

```
Route::get('/', [WelcomeController::class, 'Selamat Datang']);
```

WelcomeController.php

```
public function index() {
    return 'Selamat Datang';
}
```

output



Resource “ /about ”

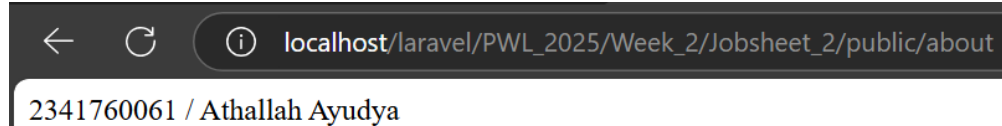
web.php

```
Route::get('/about', [WelcomeController::class, 'about']);
```

WelcomeController.php

```
public function about() {  
    return '2341760061 / Athallah Ayudya';  
}
```

output



Resource “ /articles/{id} ”

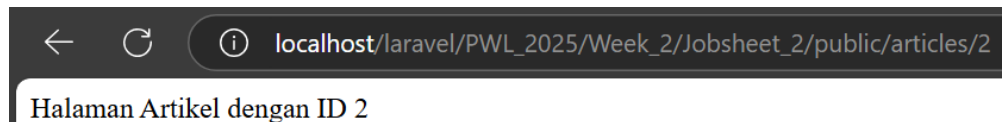
web.php

```
Route::get('/articles/{id}', [WelcomeController::class, 'articles']);
```

WelcomeController.php

```
public function articles($id) {  
    return 'Halaman Artikel dengan ID '.$id;  
}
```

output



- g. Modifikasi kembali implementasi sebelumnya dengan konsep Single Action Controller. Sehingga untuk hasil akhir yang didapatkan akan ada HomeController, AboutController dan ArticleController. Modifikasi juga route yang digunakan.

```
Week_2 > Jobsheet_2 > app > Http > Controllers > HomeController.php  
1  <?php  
2  
3  namespace App\Http\Controllers;  
4  
5  use Illuminate\Http\Request;  
6  
7  class HomeController extends Controller{  
8      public function __invoke() {  
9          return 'Selamat Datang';  
10     }  
11 }
```

```

Week_2 > Jobsheet_2 > app > Http > Controllers > 🐞 AboutController.php
1  <?php
2
3  namespace App\Http\Controllers;
4
5  use Illuminate\Http\Request;
6
7  class AboutController extends Controller{
8      public function __invoke() {
9          return '2341760061 / Athallah Ayudya';
10     }
11 }

```

```

Week_2 > Jobsheet_2 > app > Http > Controllers > 🐞 ArticleController.php
1  <?php
2
3  namespace App\Http\Controllers;
4
5  use Illuminate\Http\Request;
6
7  class ArticleController extends Controller{
8      public function __invoke($id) {
9          return 'Halaman Artikel dengan ID ' . $id;
10     }
11 }

```

web.php

```

<?php

use App\Http\Controllers\HomeController;
use App\Http\Controllers\AboutController;
use App\Http\Controllers\ArticleController;
use Illuminate\Support\Facades\Route;

Route::get('/', HomeController::class);
Route::get('/about', AboutController::class);
Route::get('/articles/{id}', ArticleController::class);

```

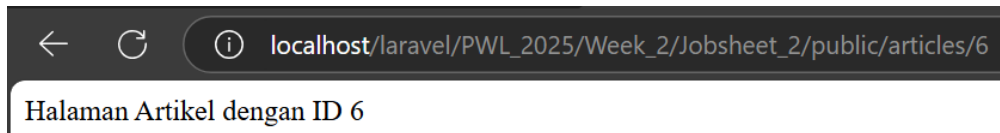
output

```

localhost/laravel/PWL_2025/Week_2/Jobsheet_2/public/
Selamat Datang

localhost/laravel/PWL_2025/Week_2/Jobsheet_2/public/about
2341760061 / Athallah Ayudya

```



- Resource Controller

Khusus untuk controller yang terhubung dengan Eloquent model dan dapat dilakukan operasi CRUD terhadap model Eloquent tersebut, kita dapat membuat sebuah controller yang bertipe Resource Controller. Dengan membuat sebuah resource controller, maka controller tersebut telah dilengkapi dengan method-method yang mendukung proses CRUD, serta terdapat sebuah route resource yang menampung route untuk controller tersebut.

Langkah-langkah Praktikum:

- Membuat sebuah file controller dengan nama PhotoController yang berisi method method standar untuk proses CRUD.

INFO Controller [C:\laragon\www\laravel\PWL_2025\Week_2\Jobsheet_2\app\Http\Controllers\PhotoController.php] created successfully.

- Buat routenya untuk menghubungkan dengan frontend pada web.php

```
use App\Http\Controllers\PhotoController;
use Illuminate\Support\Facades\Route;

Route::resource('photos', PhotoController::class);
```

- Jalankan cek list route (php artisan route:list) akan dihasilkan route berikut ini.

```
GET|HEAD / ..... WelcomeController@index
POST _ignition/execute-solution ignition.executeSolution > Spatie\LaravelIgnition > Execut...
GET|HEAD _ignition/health-check ignition.healthCheck > Spatie\LaravelIgnition > HealthCheckCon...
POST _ignition/update-config ignition.updateConfig > Spatie\LaravelIgnition > UpdateConfig...
GET|HEAD about ..... WelcomeController@about
GET|HEAD api/user .....
GET|HEAD articles/{id} ..... WelcomeController@articles
GET|HEAD hello ..... WelcomeController@hello
GET|HEAD photos ..... photos.index > PhotoController@index
POST photos ..... photos.store > PhotoController@store
GET|HEAD photos/create ..... photos.create > PhotoController@create
GET|HEAD photos/{photo} ..... photos.show > PhotoController@show
PUT|PATCH photos/{photo} ..... photos.update > PhotoController@update
DELETE photos/{photo} ..... photos.destroy > PhotoController@destroy
GET|HEAD photos/{photo}/edit ..... photos.edit > PhotoController@edit
GET|HEAD posts/{post}/comments/{comment} .....
GET|HEAD sanctum/csrf-cookie sanctum.csrf-cookie > Laravel\Sanctum > CsrfCookieController@show
GET|HEAD user/{name?} .....
GET|HEAD user/{name} .....
GET|HEAD world .....
```

- Pada route list semua route yang berhubungan untuk crud photo sudah di generate oleh laravel. Jika tidak semua route pada resource controller dibutuhkan dapat dikurangi dengan mengupdate route pada web.php menjadi seperti berikut ini.

```
Route::resource('photos', PhotoController::class)->only([
    'index', 'show'
]);

Route::resource('photos', PhotoController::class)->except([
    'create', 'store', 'update', 'destroy'
]);
```

4. View

Dalam kerangka kerja Laravel, View merujuk pada bagian dari aplikasi web yang bertanggung jawab untuk menampilkan antarmuka pengguna kepada pengguna akhir. View pada dasarnya adalah file template yang digunakan untuk menghasilkan HTML yang akan ditampilkan kepada pengguna.

Blade merupakan templating engine bawaan Laravel. Berguna untuk mempermudah dalam menulis kode tampilan. Dan juga memberikan fitur tambahan untuk memanipulasi data di view yang dilempar dari controller. Blade juga memungkinkan penggunaan plain PHP pada kode View. Karena Laravel menggunakan templating engine bawaan Blade, maka setiap file View diakhiri dengan `.blade.php`. Misal: `index.blade.php`, `home.blade.php`, `product.blade.php`.

- Membuat View

Langkah-langkah Praktikum:

- Membuat file `hello.blade.php` pada direktori `app/resources/views`

```
<!-- View pada resources/views/hello.blade.php -->
<html>
  <body>
    <h1>Hello, {{ $name }}</h1>
  </body>
</html>
```

- View tersebut dapat dijalankan melalui Routing, dimana route akan memanggil View sesuai dengan nama file tanpa `'blade.php'`.

```
Route::get('/greeting', function () {
    return view('hello', ['name' => 'Athallah']);
});
```

- Jalankan code

← ↻ ⓘ localhost/laravel/PWL_2025/Week_2/Jobsheet_2/public/greeting

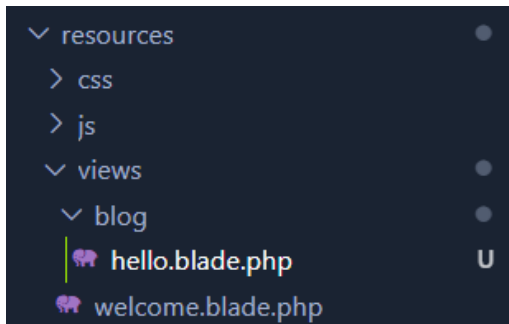
Hello, Athallah

- View dalam direktori

Jika di dalam direktori `resources/views` terdapat direktori lagi untuk menyimpan file view, sebagai contoh `hello.blade.php` ada di dalam direktori `blog`, maka kita bisa menggunakan “dot” notation untuk mereferensikan direktori,

Langkah-langkah Praktikum:

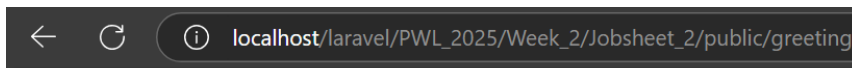
- Buatlah direktori `blog` di dalam direktori `views`.
- Pindahkan file `hello.blade.php` ke dalam direktori `blog`.



- c. Selanjutnya lakukan perubahan pada route.

```
Route::get('/greeting', function () {  
    return view('blog.hello', ['name' => 'Athallah']);  
});
```

- d. Jalankan kode



Hello, Athallah

- Menampilkan View dari Controller

View dapat dipanggil melalui Controller. Sehingga Routing akan memanggil Controller terlebih dahulu, dan Controller akan me-return view yang dimaksud.

Langkah-langkah Praktikum:

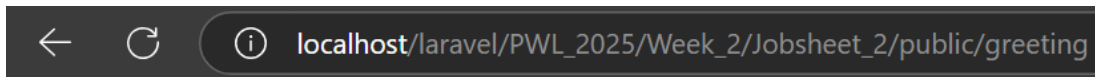
- a. Buka WelcomeController.php dan tambahkan fungsi baru yaitu greeting.

```
public function greeting(){  
    return view('blog.hello', ['name' => 'Athallah']);  
}
```

- b. Ubah route /greeting dan arahkan ke WelcomeController pada fungsi greeting.

```
Route::get('/greeting', [WelcomeController::class, 'greeting']);
```

- c. Jalankan kode



Hello, Athallah

- Meneruskan data ke view

Saat meneruskan informasi dengan cara ini, data harus berupa array dengan pasangan kunci / nilai. Setelah memberikan data ke view, kemudian kita dapat mengakses setiap nilai dalam view menggunakan kunci data seperti: {{ \$name }}. Sebagai alternatif untuk meneruskan array data lengkap ke fungsi view helper, kita dapat menggunakan metode with untuk menambahkan bagian data individual ke view.

Metode with mengembalikan instance view objek sehingga kita dapat melanjutkan rangkaian metode sebelum mengembalikan tampilan notation untuk mereferensikan direktori,

Langkah-langkah Praktikum:

- a. Buka WelcomeController.php dan tambahkan ubah fungsi greeting.

```
public function greeting(){  
    return view('blog.hello')  
        -> with ('name','Athai')  
        -> with ('occupation', 'Damkar');  
}
```

- b. Ubah hello.blade.php agar dapat menampilkan dua parameter.

```
<html>  
    <body>  
        <h1>Hello, {{ $name }}</h1>  
        <h1>You are {{ $occupation }}</h1>  
    </body>  
</html>
```

- c. Jalankan kode



SOAL PRAKTIKUM

1. Jalankan Langkah-langkah Praktikum pada jobsheet di atas. Lakukan sinkronisasi perubahan pada project PWL_2025 ke Github.
2. Buatlah project baru dengan nama POS. Project ini merupakan sebuah aplikasi Point of Sales yang digunakan untuk membantu penjualan.
3. Buatlah beberapa route, controller, dan view sesuai dengan ketentuan sebagai berikut.

1	Halaman Home Menampilkan halaman awal website
2	Halaman Products Menampilkan daftar product (route prefix) /category/food-beverage /category/beauty-health /category/home-care /category/baby-kid
3	Halaman User Menampilkan profil pengguna (route param) /user/{id}/name/{name}
4	Halaman Penjualan Menampilkan halaman transaksi POS

4. Route tersebut menjalankan fungsi pada Controller yang berbeda di setiap halaman.
5. Fungsi pada Controller akan memanggil view sesuai halaman yang akan ditampilkan.
6. Simpan setiap perubahan yang dilakukan pada project POS pada Git, sinkronisasi perubahan ke Github.

Jawab:

1. Membuat project baru dengan nama POS
2. Membuat Controller. Berikut merupakan Controller yang dibuat:
 - HomeController → Menampilkan halaman utama.

```
λ php artisan make:controller HomeController
```
 - ProductController → Menampilkan daftar kategori produk.

```
λ php artisan make:controller ProductController
```
 - UserController → Menampilkan profil pengguna dengan parameter id dan name.

```
λ php artisan make:controller UserController
```
 - TransactionController → Menampilkan halaman transaksi POS.

```
λ php artisan make:controller TransactionController
```

3. Mendefinisikan Route

```

POS > routes > web.php
1  <?php
2
3  use Illuminate\Support\Facades\Route;
4  use App\Http\Controllers\HomeController;
5  use App\Http\Controllers\ProductController;
6  use App\Http\Controllers\UserController;
7  use App\Http\Controllers\TransactionController;
8
9  // Halaman Home
10 Route::get('/', [HomeController::class, 'index']);
11
12 // Halaman Products (Menggunakan route prefix)
13 Route::prefix('category')->group(function () {
14     Route::get('/food-beverage', [ProductController::class, 'foodBeverage']);
15     Route::get('/beauty-health', [ProductController::class, 'beautyHealth']);
16     Route::get('/home-care', [ProductController::class, 'homeCare']);
17     Route::get('/baby-kid', [ProductController::class, 'babyKid']);
18 });
19
20 // Halaman User (Menggunakan route parameter)
21 Route::get('/user/{id}/name/{name}', [UserController::class, 'show']);
22
23 // Halaman Penjualan (POS)
24 Route::get('/transaction', [TransactionController::class, 'index']);

```

4. Mengimplementasikan Controller

```

POS > app > Http > Controllers > HomeController.php
1  <?php
2
3  namespace App\Http\Controllers;
4
5  use Illuminate\Http\Request;
6
7  class HomeController extends Controller
8  {
9      public function index()
10     {
11         return view('home');
12     }
13 }

```


POS > app > Http > Controllers > ProductController.php

```
1  <?php
2
3  namespace App\Http\Controllers;
4
5  use Illuminate\Http\Request;
6
7  class ProductController extends Controller
8  {
9      public function foodBeverage()
10     {
11         return view('products.food_beverage');
12     }
13
14     public function beautyHealth()
15     {
16         return view('products.beauty_health');
17     }
18
19     public function homeCare()
20     {
21         return view('products.home_care');
22     }
23
24     public function babyKid()
25     {
26         return view('products.baby_kid');
27     }
28 }
```

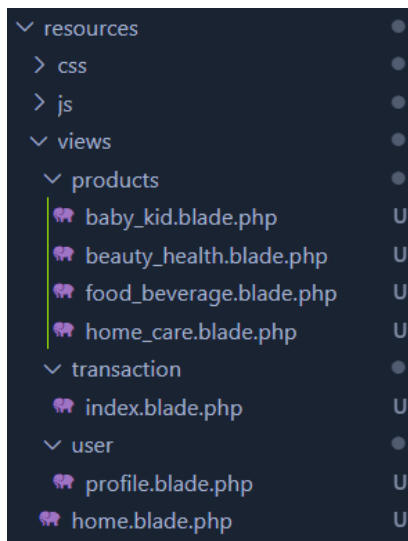
POS > app > Http > Controllers > UserController.php

```
1  <?php
2
3  namespace App\Http\Controllers;
4
5  use Illuminate\Http\Request;
6
7  class UserController extends Controller
8  {
9      public function show($id, $name)
10     {
11         return view('user.profile', compact('id', 'name'));
12     }
13 }
```

POS > app > Http > Controllers > TransactionController.php

```
1  <?php
2
3  namespace App\Http\Controllers;
4
5  use Illuminate\Http\Request;
6
7  class TransactionController extends Controller
8  {
9      public function index()
10     {
11         return view('transaction.index');
12     }
13 }
```

5. Membuat View sesuai dengan Controller



6. Output

Halaman Home



Halaman Product

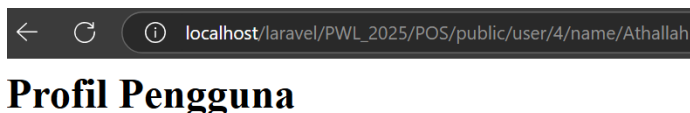


Daftar makanan dan minuman



Produk kecantikan dan kesehatan

Halaman User



ID: 4

Nama: Athallah

Halaman Penjualan



Transaksi POS