

Laporan Praktikum Pemrograman Web Lanjut
Jobsheet 2 : Routing, Controller, dan View



Oleh :
Danica Nasywa Putrinier (2341760122)
Kelas SIB 2B / 05

PROGRAM STUDI D-IV SISTEM INFORMASI BISNIS JURUSAN
TEKNOLOGI INFORMASI
POLITEKNIK NEGERI MALANG

Jl. Soekarno Hatta No.9, Jatimulyo, Kec. Lowokwaru, Kota Malang, Jawa Timur

65141

I. MVC pada Laravel

II. Routing

1. Basic Routing

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

b. Buka file routes/web.php. Tambahkan sebuah route nomor 1 seperti dibawah ini

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

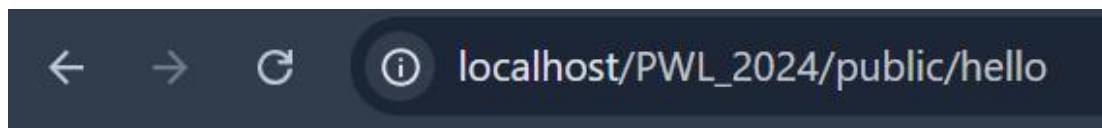
Route::get('/hello', function () {
    return 'Hello World';
});
```

Code :

```
16
17 Route::get('/world', function () { //mendefinisikan rute untuk halaman utam
18 |   return 'Hello World'; //mengembalika tampilan
19 | });
20
```

c. Buka browser, tuliskan URL untuk memanggil route tersebut: localhost/PWL_2024/public/hello. Perhatikan halaman yang muncul apakah sudah sesuai dan jelaskan pengamatan Anda.

Hasil :



Hello World

Menurut pengamatan saya code

- Route::get digunakan untuk mengambil data atau menampilkan halaman.
- '/hello' digunakan untuk menentukan URI (Uniform Resource Identifier) yang akan di panggil di browser
- return 'Hello World'; digunakan untuk mengembalikan output dengan yang ditampilkan adalah Hello World

d. Untuk membuat route kedua, tambahkan route /world seperti dibawah ini

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

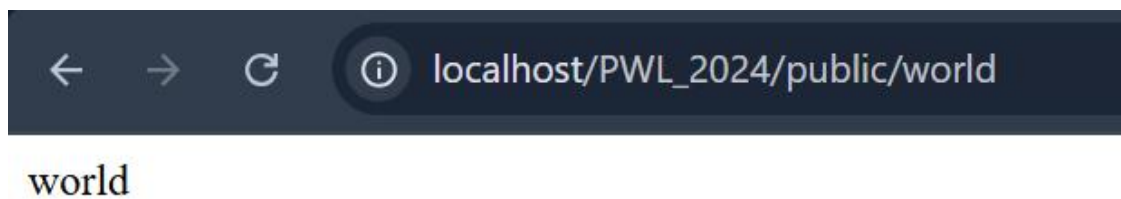
Route::get('/world', function () {
    return 'World';
});
```

Code :

```
Route::get('/world', function () { //mendefinisikan rute untuk halaman utama
    return 'world'; //mengembalikan tampilan
});
```

e. Bukalah pada browser, tuliskan URL untuk memanggil route tersebut localhost/PWL_2024/public/world. Perhatikan laman yang muncul apakah sesuai dan jelaskan pengamatan Anda.

Hasil :



Penjelasan :

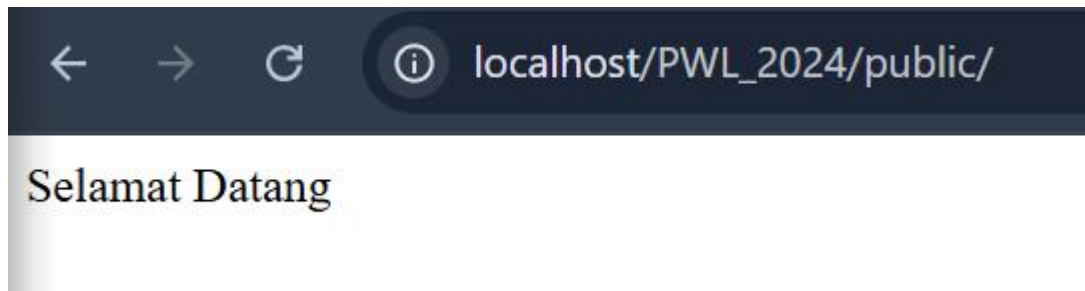
Sama seperti jawaban pada point c dengan menggunakan logic yang sama fungsi get akan mengambil data, dan return digunakan untuk menampilkan.

f. Selanjutnya cobalah membuat route '/' yang menampilkan pesan 'Selamat Datang'.

Code :

```
Route::get('/', function () { //mendefinisikan rute untuk halaman utama
    return 'Selamat Datang'; //mengembalikan tampilan
});
```

Hasil :



- g. Kemudian buatlah route '/about' yang akan menampilkan NIM dan nama Anda.

Code :

```
Route::get('/about', function () { //mendefinisika  
    return '2341760122 - Danica Nasywa Putrinia  
});
```

Hasil :



2. Route Parameters

- a. Kita akan memanggil route `/user/{name}` sekaligus mengirimkan parameter berupa nama user `$name` seperti kode di bawah ini

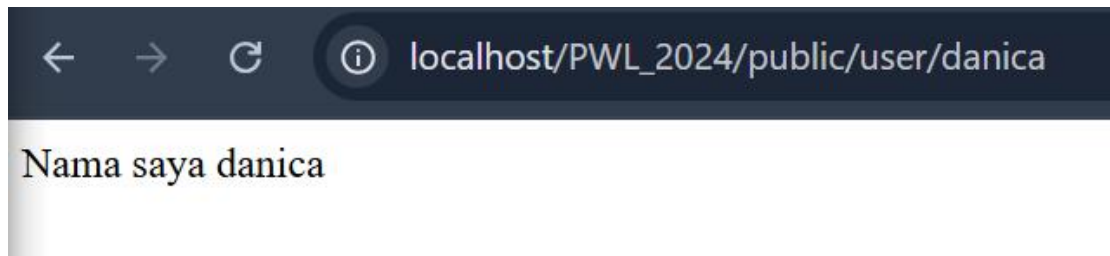
```
Route::get('/user/{name}', function ($name) {  
    return 'Nama saya '.$name;  
});
```

- b. Jalankan kode dengan menuliskan URL untuk memanggil route tersebut: **localhost/PWL_2024/public/user>NamaAnda**. Perhatikan yang muncul dan jelaskan pengamatan Anda.

Code :

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

Hasil :

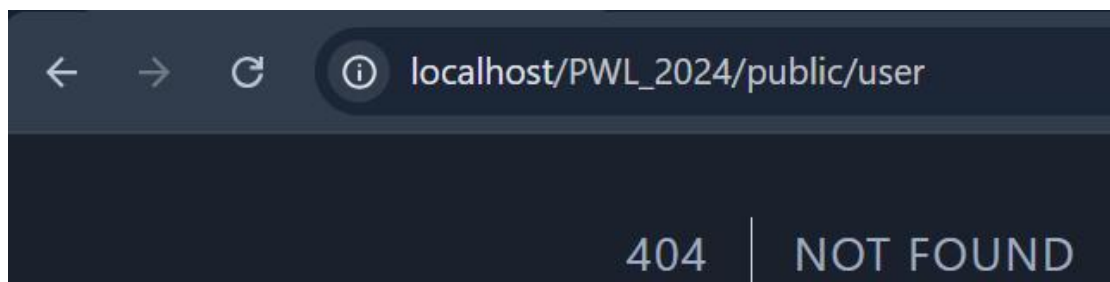


Penjelasan :

Untuk penggunaan logic kurang lebih sama dengan logic sebelumnya, terdapat perbedaan dalam penambahan parameter dinamis {name} yang menunjukkan bahwa bagian tersebut merupakan parameter dinamis. Nilai dari {name} akan ditangkap sebagai parameter \$name dan ditampilkan sebagai “Nama saya {name}” saat diakses. Contohnya, mengunjungi /user/danica akan menampilkan “Nama saya danica”.

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

Hasil :



Penjelasan :

Terjadi error karena inputan yang seharusnya diisi username null.

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

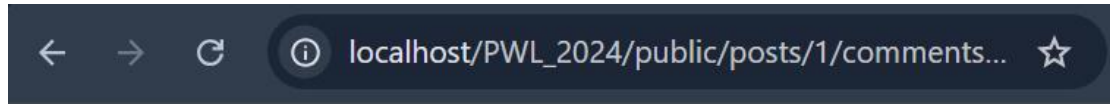
```
Route::get('/posts/{post}/comments/{comment}', function ($postId, $commentId) {  
    return 'Pos ke-' . $postId . " Komentar ke-: " . $commentId;  
});
```

- 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

Code :

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

Hasil :



Pos ke- 1 Komentar ke-: 5

Penjelasan :

Untuk penggunaan logic kurang lebih sama dengan logic sebelumnya, terdapat perbedaan dalam penambahan 2 parameter dinamis {post} dan {comment} yang menunjukkan bahwa bagian tersebut merupakan parameter dinamis. Nilai dari {post} dan {comment} akan ditangkap sebagai parameter \$postId dan \$commentId yang ditampilkan sebagai 'Pos ke- '.\$postId." Komentar ke-: ".\$commentId saat diakses. Contohnya, mengunjungi /posts/1/comment/5 akan menampilkan “Pos ke- 1 Komentar ke-: 5”.

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

Code :

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

Hasil :



Halaman Artikel dengan ID 2001D

Penjelasan :

Terjadi error karena inputan yang seharusnya diisi username null.

3. Optimal Parameters

- a. Kita akan memanggil route `/user` sekaligus mengirimkan parameter berupa

nama user \$name dimana parameternya bersifat opsional.

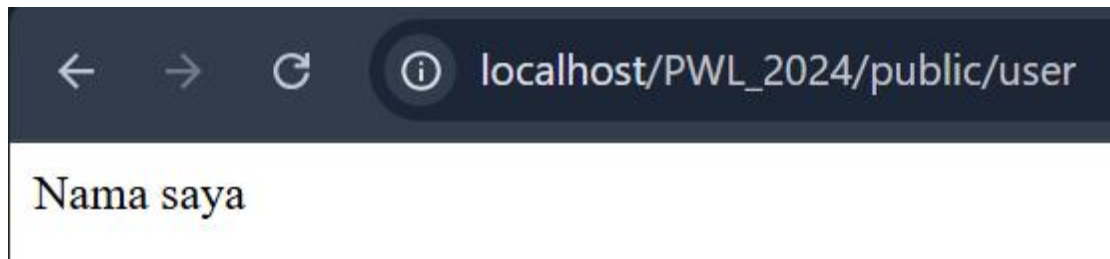
```
Route::get('/user/{name?}', function ($name=null) {  
    return 'Nama saya '.$name;  
});
```

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

Code :

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

Hasil :

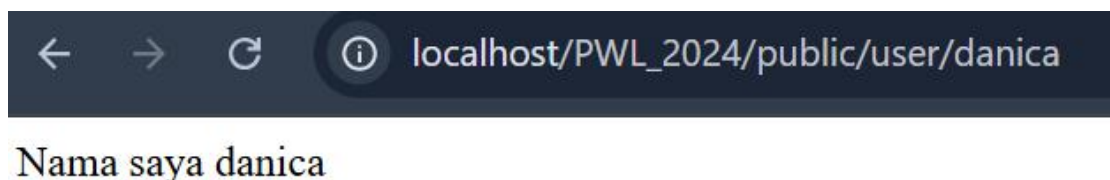


Penjelasan :

Pada *endpoint* yang digunakan adalah `/user/{name?}`, memiliki tanda tanya (?) yang menandakan bahwa parameter tersebut bersifat opsional. `{name?}` menunjukkan bahwa parameter name tidak wajib diisi. Maka, pada saat fungsi dijalankan tidak terjadi *error* karena apabila *name* masih null fungsi masih bisa dijalankan.

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

Hasil :



Penjelasan :

Seperti penjelasan sebelumnya karena parameternya bersifat opsional maka saat diisi nilai “danica” akan dikirim ke fungsi sebagai *\$name*.

d. Ubah kode para route /user menjadi seperti dibawah ini

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

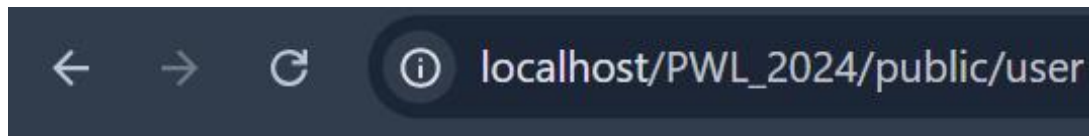
e. Jalankan kode dengan URL:localhost/PWL_2024/public/user/.

Perhatikan halaman yang muncul dan jelaskan pengamatan Anda

Code :

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

Hasil :



Nama saya John

Penjelasan :

Seperti penjelasan sebelumnya karena parameternya bersifat opsional maka saat diisi nilai “danica” akan dikirim ke fungsi sebagai *\$name*.

4. Route Name

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

```
Route::get('/user/profile', function () {  
    //  
})->name('profile');  
  
Route::get(  
    '/user/profile',  
    [UserProfileController::class, 'show']  
)->name('profile');  
  
// Generating URLs...  
$url = route('profile');  
  
// Generating Redirects...  
return redirect()->route('profile');
```

5. Route Group dan Route Prefixes

Route name biasanya digunakan untuk mempermudah kita dalam pemanggilan route saat membangun aplikasi. Kita cukup memanggil name dari route tersebut. digunakan untuk middleware masih ada lagi penggunaan route group untuk route yang berada dibawah satu subdomain. Contoh penggunaan route group adalah sebagai berikut:

```
Route::middleware(['first', 'second'])->group(function () {
    Route::get('/', function () {
        // Uses first & second middleware...
    });

    Route::get('/user/profile', function () {
        // Uses first & second middleware...
    });
});

Route::domain('{account}.example.com')->group(function () {
    Route::get('user/{id}', function ($account, $id) {
        //
    });
});

Route::middleware('auth')->group(function () {
    Route::get('/user', [UserController::class, 'index']);
    Route::get('/post', [PostController::class, 'index']);
    Route::get('/event', [EventController::class, 'index']);
});
```

Route Prefixes

Pengelompokan route juga dapat dilakukan untuk route yang memiliki prefix (awalan) yang sama. Untuk pembuatan route dengan prefix dapat dilihat kode seperti di bawah ini

```
Route::prefix('admin')->group(function () {
    Route::get('/user', [UserController::class, 'index']);
    Route::get('/post', [PostController::class, 'index']);
    Route::get('/event', [EventController::class, 'index']);
});
```

6. Redirect Routes

Untuk melakukan redirect pada laravel dapat dilakukan dengan menggunakan Route::redirect cara penggunaannya dapat dilihat pada kode program dibawah ini.

```
Route::redirect('/here', '/there');
```

Redirect ini akan sering digunakan pada kasus kasus CRUD atau kasus lain yang membutuhkan redirect.

7. 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.

```
Route::view('/welcome', 'welcome');  
Route::view('/welcome', 'welcome', ['name' => 'Taylor']);
```

Pada view routes diatas /welcome akan menampilkan view welcome dan pada route kedua /welcome akan menampilkan view welcome dengan tambahan data berupa variabel name.

III. Controller

1. Membuat Controller

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

Contoh :

```
php artisan make:controller WelcomeController
```

Hasil :

```
Donny@DESKTOP-TTAQTTR MINGW64 /e/Aplik/laragon/www/PwL_2024 (main)  
$ php artisan make:controller WelcomeController  
  
[INFO] Controller [E:\Aplik\laragon\www\PwL_2024\app\Http\Controllers\WelcomeC  
ontroller.php] created successfully.
```

- b. Buka file pada app/Htto/Controllers/WElcomeController.php. Struktur pada controller dapat digambrkn sebagai berikut

```
<?php  
  
namespace App\Http\Controllers;  
  
use Illuminate\Http\Request;  
  
Class WelcomeController extedns Controller  
{  
    //  
}
```

Hasil :

```
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 tmbhkan function dengan access public. Sehingga controller di atas mejadi sebagai berikut:

```
<?php
namespace App\Http\Controllers;
use Illuminate\Http\Request;
class WelcomeController extends Controller
{
    public function hello() {
        return 'Hello World';
    }
}
```

Hasil :

```
<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;

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

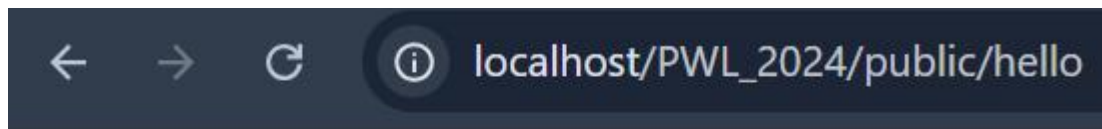
- d. Setelah sebuah controller telah didefinisikan action, kita dapat menambahkan controller tersebut pada route. Ubah route/hello menjadi seperti berikut:

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

Hasil :

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

- e. Buka browser, tuliskan URL untuk memanggil route tersebut localhost/PWL_2024/public/hello. Perhatikan halaman yang muncul dan jelaskan pengamatan Anda.



Hello World

Penjelasan :

Pada point pertama membuat controller baru yang akan digunakan untuk menangani logika aplikasi. Setelah dibuat akan otomatis terisi template dengan menunjukkan Controller berada dalam namespace App\Http\Controllers. Lalu menambahkan method Hello() dengan menampilkan teks “Hello World”. Setelah controller memiliki method, kita menyambungkan ke route dengan definisi route /hello di file routes/web.php. [WelcomeController::class, 'hello'] menunjukkan bahwa WelcomeController akan menangani request ini. Saat diakses di browser tampilan akan muncul Hello World.

- f. Modifikasi hasil apda praktikum poin 2 (Routing) dengan konsep controller. Pindahkan logika eksekusi ke dalam controller dengan nama PageController.

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		

Code :

```

1  <?php
2
3  namespace App\Http\Controllers;
4
5  use Illuminate\Http\Request;
6
7  class PagesController extends Controller
8  {
9      public function index() {
10         return 'Selamat Datang';
11     }
12
13     public function about() {
14         return 'Nama : Danica Nasywa Putrinia || NIM : 2341760122';
15     }
16
17     public function articles($Id) {
18         return 'Halaman artikel dengan ID : '.$Id;
19     }
20 }
21

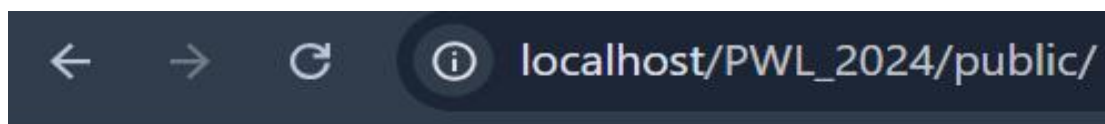
```

```

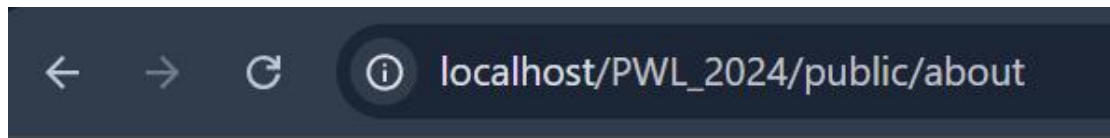
Route::get('/', [PagesController::class, 'index']);
Route::get('/about', [PagesController::class, 'about']);
Route::get('/articles/{id}', [PagesController::class, 'articles']);

```

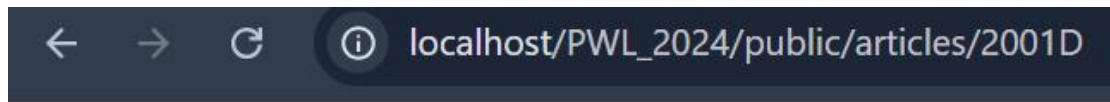
Hasil :



Selamat Datang



Nama : Danica Nasywa Putrinier || NIM : 2341760122



Halaman artikel dengan ID : 2001D

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 akan digunakan

Terminal Controller :

```
Donny@DESKTOP-TTAQTTR MINGW64 /e/Aplik/laragon/www/PWL_2024 (main)
$ php artisan make:controller HomeController

[INFO] Controller [E:\Aplik\laragon\www\PWL_2024\app\Http\Controllers\HomeController.php] created successfully.

Donny@DESKTOP-TTAQTTR MINGW64 /e/Aplik/laragon/www/PWL_2024 (main)
$ php artisan make:controller AboutController

[INFO] Controller [E:\Aplik\laragon\www\PWL_2024\app\Http\Controllers\AboutController.php] created successfully.

Donny@DESKTOP-TTAQTTR MINGW64 /e/Aplik/laragon/www/PWL_2024 (main)
$ php artisan make:controller ArticleController

[INFO] Controller [E:\Aplik\laragon\www\PWL_2024\app\Http\Controllers\ArticleController.php] created successfully.
```

Single Action Controller :

```
app > Http > Controllers > AboutController.php > AboutController > about
1  <?php
2
3  namespace App\Http\Controllers;
4
5  use Illuminate\Http\Request;
6
7  class AboutController extends Controller
8  {
9      public function about() {
10         return 'Nama : Danica Nasywa Putrinier || NIM : 2341760122';
11     }
12 }
```



```

app > Http > Controllers > ArticleController.php > ArticleController > articles
1  <?php
2
3  namespace App\Http\Controllers;
4
5  use Illuminate\Http\Request;
6
7  class ArticleController extends Controller
8  {
9      public function articles($Id) {
10         return 'Halaman artikel dengan ID : '.$Id;
11     }
12 }

```

```

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         return 'Selamat Datang';
11     }
12 }
13

```

Route Modification :

```

7  use App\Http\Controllers\AboutController;
8  use App\Http\Controllers\ArticleController;
9  use App\Http\Controllers\HomeController;

28 Route::get('/', [HomeController::class, 'index']);
29 Route::get('/about', [AboutController::class, 'about']);
30 Route::get('/articles/{id}', [ArticleController::class, 'articles']);
31

```

2. Resource Controller

- Untuk membuatnya dilakukan dengan menjalankan perintah berikut ini di terminal.

```
php artisan make:controller PhotoController --resource
```

Hasil :

```
Donny@DESKTOP-TTAQTTR MINGW64 /e/Aplik/laragon/www/PwL_2024 (main)
$ php artisan make:controller PhotoController --resource

INFO Controller [E:\Aplik\laragon\www\PwL_2024\app\Http\Controllers\PhotoController.php] created successfully.
```

Perintah ini akan men generate sebuah controller dengan nama PhotoController yang berisi method standar ntuk proses CRUD.

- b. Setelah controller berhasil di generate, selanjutnya harus dibuatkan route agar dapat terhubung dengan frontend. Tambahkan kode program berikut pada file web.php.

```
use App\Http\Controllers\PhotoController;

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

Hasil :

```
use App\Http\Controllers\HomeController;
use App\Http\Controllers\PhotoController;

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

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

Domain	Method	URI	Name	Action	Middleware
	GET HEAD	/		Closure	web
	GET HEAD	api/user		Closure	api auth:api
	GET HEAD	photos	photos.index	App\Http\Controllers\PhotoController@index	web
	POST	photos	photos.store	App\Http\Controllers\PhotoController@store	web
	GET HEAD	photos/create	photos.create	App\Http\Controllers\PhotoController@create	web
	GET HEAD	photos/{photo}	photos.show	App\Http\Controllers\PhotoController@show	web
	PUT PATCH	photos/{photo}	photos.update	App\Http\Controllers\PhotoController@update	web
	DELETE	photos/{photo}	photos.destroy	App\Http\Controllers\PhotoController@destroy	web
	GET HEAD	photos/{photo}/edit	photos.edit	App\Http\Controllers\PhotoController@edit	web
	GET HEAD POST PUT PATCH DELETE OPTIONS	specialMahasiswa		Closure	web
	GET POST HEAD	specialUrl		Closure	web

Hasil :

```
Donny@DESKTOP-TTAQTTR MINGW64 /e/Aplik/laragon/www/PwL_2024 (main)
$ php artisan route:list

GET|HEAD / ..... HomeController@index
POST _ignition/execute-solution ignition.executeSolution > Spatie\Ignition\HealthCheck
GET|HEAD _ignition/health-check ignition.healthCheck > Spatie\Ignition\HealthCheck
POST _ignition/update-config ignition.updateConfig > Spatie\Ignition\HealthCheck
GET|HEAD about ..... AboutController@about
GET|HEAD api/user .....
GET|HEAD articles/{id} ..... ArticleController@articles
GET|HEAD items ..... items.index > ItemController@index
POST items ..... items.store > ItemController@store
GET|HEAD items/create ..... items.create > ItemController@create
GET|HEAD items/{item} ..... items.show > ItemController@show
PUT|PATCH items/{item} ..... items.update > ItemController@update
DELETE items/{item} ..... items.destroy > ItemController@destroy
GET|HEAD items/{item}/edit ..... items.edit > ItemController@edit
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 sanctum/csrf-cookie sanctum.csrf-cookie > Laravel\Sanctum\Sanctum

Showing [22] routes
```

- d. 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'
]);
```

IV. View

1. Membuat View

- a. Pada direktori resources/views, buatlah file hello.blade.php

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

Hasil :

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

- b. View tersebut dapat dijalankan melalui Routinf, dimana *route* akan memanggil View sesuai dengan nama *file* tanpa 'blade.php'. (Catatan : Gantilah Andi dengan nama Anda)

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

Hasil :

```

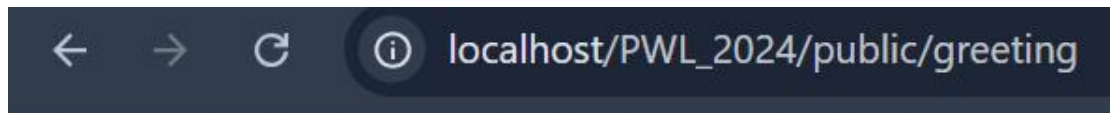
41 Route::get('/greeting', function () {
42     return view('hello', ['name' => 'Danica']);
43 });

```

c. Jalankan code dengan membuka url localhost/PWL_2024/public/greeting.

Perhatikan halaman yang muncul dan jelaskan pengamatan Anda.

Hasil :



Hello, Danica!

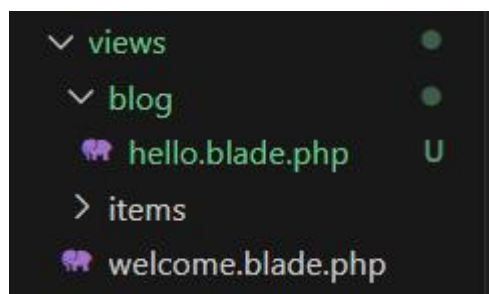
Penjelasan :

Laravel mencocokkan permintaan (GET/greeting) dan apabila cocok, route tersebut akan mengembalikan tampilan (hello.blade.php) dengan variabel *name* yang berisi “Danica”. Dalam file resource/views/hello.blade.php ekspresi {{ \$name }} akan digantikan dengan nilai “Danica”.

2. View Dalam Direktori

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

Hasil :



c. Selanjutnya lakukan perubahan pada route.

```

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

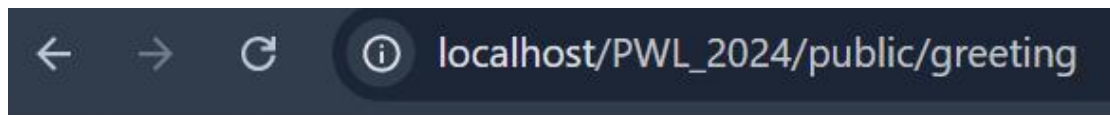
```

Hasil :


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

d. Jalankan code dengan membuka URL `localhost/PWL_2024/public/greeting`. Perhatikan halaman yang muncul dan jelaskan pengamatan Anda.

Hasil :



Hello, Danica!

Penjelasan :

Laravel mencocokkan permintaan (GET/greeting) dan apabila cocok, route tersebut akan mengembalikan tampilan (hello.blade.php) yang berada di folder blog, dengan variabel *name* yang berisi “Danica” . Dalam file resource/views/blog/hello.blade.php ekspresi {{ \$name }} akan digantikan dengan nilai “Danica”.

3. Menampilkan View dari Controller

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

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

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

Hasil :

```

app > Http > Controllers > WelcomeController.php > WelcomeController > greeting
1  <?php
2
3  namespace App\Http\Controllers;
4
5  use Illuminate\Http\Request;
6
7  class WelcomeController extends Controller
8  {
9      public function hello(){
10         return 'Hello World';
11     }
12
13     public function greeting(){
14         return view('blog.hello', ['name' => 'Danica']);
15     }
16 }
17

```

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

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

Hasil :

```

40
41 Route::get('/greeting', [WelcomeController::class, 'greeting']);
42

```

- c. Jalankan code dengan membuka URL localhost/PWL_2024/public/greeting. Perhatikan halaman yang muncul dan jelaskan pengamatan Anda.

Hasil :



Penjelasan :

Laravel mencocokkan permintaan (GET/greeting) dan apabila cocok, route tersebut akan mengembalikan tampilan (hello.blade.php) yang berada di folder blog, dengan variabel *name* yang berisi “Danica” . Dalam file resource/views/blog/hello.blade.php ekspresi {{ \$name }}

akan digantikan dengan nilai “Danica”.

4. Meneruskan data ke view

Pada contoh sebelumnya, kita dapat meneruskan data array ke view agar data tersebut tersedia untuk view:

```
return view('blog.hello', ['name' => 'Andi']);
```

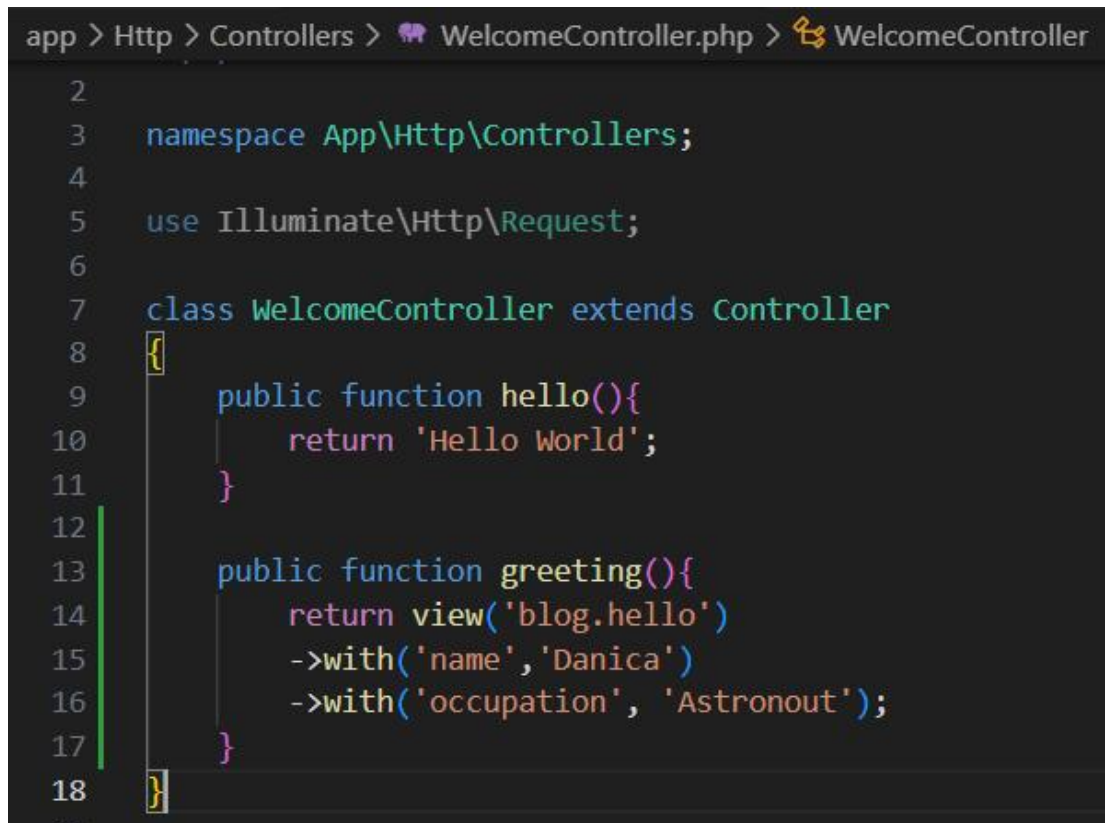
a. Buka WelcomeController.php dan tambahkan ubah fungsi greeting

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

    public function greeting(){
        return view('blog.hello')
```

```
        ->with('name', 'Andi')
        ->with('occupation', 'Astronaut');
    }
}
```

Hasil :



```
app > Http > Controllers > WelcomeController.php > WelcomeController
2
3 namespace App\Http\Controllers;
4
5 use Illuminate\Http\Request;
6
7 class WelcomeController extends Controller
8 {
9     public function hello(){
10         return 'Hello World';
11     }
12
13     public function greeting(){
14         return view('blog.hello')
15             ->with('name', 'Danica')
16             ->with('occupation', 'Astronaut');
17     }
18 }
```

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

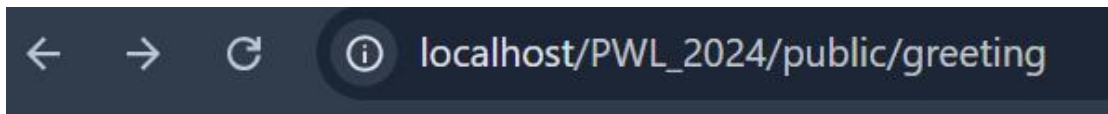
```
<html>
  <body>
    <h1>Hello, {{ $name }}</h1>
    <h1>You are {{ $occupation }}</h1>
  </body>
</html>
```

Hasil :

```
resources > views > blog > hello.blade.php > html
1  <!-- View pada resource/view/hello.blade.php -->
2  <html>
3    <body>
4      <h1>Hello, {{ $name }}!</h1>
5      <h1>You are {{ $occupation }}</h1>
6    </body>
7  </html>
```

c. Jalankan code dengan membuka URL localhost/PWL_2024/public/greeting.
Perhatikan halaman yang muncul dan jelaskan pengamatan Anda.

Hasil :



Hello, Danica!

You are Astronout

Penjelasan :

Pada view menampilkan \$name dan \$occupation yang akan dikirim dari controller. Metode greeting() mengembalikan view dengan dua variabel. Saat dijalankan akan menampilkan hasil dari 2 parameter yang terdapat dalam code.

SOAL PRAKTIKUM

1. Jalankan Langkah-langkah Praktikum pada jobsheet di atas.
Lakukan sinkronisasi perubahan pada project PWL_2024 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.

Jawaban

1. Jalankan composer create-project Larvel/Laravel="10.3" POS pada terminal

```
Donny@DESKTOP-TTAQTTR MINGW64 /e/Aplik/laragon/www
$ composer create-project Laravel/Laravel="10.3" POS
Creating a "Laravel/Laravel=10.3" project at "./POS"
Installing laravel/laravel (v10.3.0)
  - Downloading laravel/laravel (v10.3.0)
  - Installing laravel/laravel (v10.3.0): Extracting archive
Created project in E:\Aplik\laragon\www\POS
> @php -r "file_exists('.env') || copy('.env.example', '.env');"
Loading composer repositories with package information
Updating dependencies
Lock file operations: 111 installs, 0 updates, 0 removals
  - Locking brick/math (0.12.1)
  - Locking carbonphp/carbon-doctrine-types (2.1.0)
  - Locking dflydev/dot-access-data (v3.0.3)
```

2. Selanjutnya jalankan cd POS dan php artisan serve

```
Donny@DESKTOP-TTAQTTR MINGW64 /e/Aplik/laragon/www
$ cd POS

Donny@DESKTOP-TTAQTTR MINGW64 /e/Aplik/laragon/www/POS
$ php artisan serve

INFO Server running on [http://127.0.0.1:8000].
```

3. Jalankan php artisan make:controller HomeController, php artisan make:controller ProductController, php artisan make:controller UserController, php artisan make:controller SalesController.

```
Donny@DESKTOP-TTAQTTR MINGW64 /e/Aplik/laragon/www/POS
$ php artisan make:controller HomeController

INFO Controller [E:\Aplik\laragon\www\POS\app\Http\Controllers\HomeController.php] created successfully.

Donny@DESKTOP-TTAQTTR MINGW64 /e/Aplik/laragon/www/POS
$ php artisan make:controller ProductController

INFO Controller [E:\Aplik\laragon\www\POS\app\Http\Controllers\ProductController.php] created successfully.

Donny@DESKTOP-TTAQTTR MINGW64 /e/Aplik/laragon/www/POS
$ php artisan make:controller UserController

INFO Controller [E:\Aplik\laragon\www\POS\app\Http\Controllers\UserController.php] created successfully.

Donny@DESKTOP-TTAQTTR MINGW64 /e/Aplik/laragon/www/POS
$ php artisan make:controller SalesController

INFO Controller [E:\Aplik\laragon\www\POS\app\Http\Controllers\SalesController.php] created successfully.
```

4. Membuat Route dan Controller pada routes/web.php

```

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\SalesController;
7  use App\Http\Controllers\UserController;
8  /*
9  |-----
10 | Web Routes
11 |-----
12 |
13 | Here is where you can register web routes for your application. These
14 | routes are loaded by the RouteServiceProvider and all of them will
15 | be assigned to the "web" middleware group. Make something great!
16 |
17 */
18 // Route Home
19 Route::get('/', [HomeController::class, 'index'])->name('home');
20
21 // Route Product dengan Prefix
22 Route::prefix('product')->group(function () {
23     Route::get('/food-beverage', [ProductController::class, 'foodBeverages'])->name('category.food-beverage');
24     Route::get('/beauty-health', [ProductController::class, 'beautyHealth'])->name('category.beauty-health');
25     Route::get('/home-care', [ProductController::class, 'homeCare'])->name('category.home-care');
26     Route::get('/baby-kid', [ProductController::class, 'babyKid'])->name('category.baby-kid');
27 });
28
29 // Route User dengan Parameter
30 Route::get('/user/{id}/name/{name}', [UserController::class, 'showProfile'])->name('user.profile');
31
32 // Route Penjualan
33 Route::get('/sales', [SalesController::class, 'index'])->name('sales');

```

5. Mengisi controller

a. HomeController.php

```

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         return view('home');
11     }
12 }
13

```

b. ProductController.php


```
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         return view('products.food-baverages');
11     }
12     public function beautyHealth(){
13         return view('products.beauty-health');
14     }
15     public function homeCare(){
16         return view('products.home-care');
17     }
18     public function babyKid(){
19         return view('products.baby-kid');
20     }
21 }
```

c. UserController.php

```
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 showProfile($id, $name){
10         return view('user.profile', compact('id', 'name'));
11     }
12 }
13
```

d. SalesController.php


```

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

```

6. Membuat view

Membuat file Blade untuk setiap halaman di resource/views/.

a. resource/views/home.blade.php

```

resources > views > home.blade.php > html > body > h2
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4      <title>Home</title>
5  </head>
6  <body>
7      <h1>Welcome to POS</h1>
8
9      <h2>Product Categories</h2>
10 </body>
11 </html>
12

```

b. resource/views/products/food-baverages.blade.php

```

resources > views > products > food-baverages.blade.php > ...
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4      <title>Food & Beverage</title>
5  </head>
6  <body>
7      <h1>Food & Beverage Products</h1>
8  </body>
9  </html>
10

```

c. resource/views/products/beauty-health.blade.php

```
resources > views > products > 🧠 beauty-health.blade.php > ...
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4  |   <title>Beauty & Health</title>
5  </head>
6  <body>
7  |   <h1>Beauty & Health Products</h1>
8  </body>
9  </html>
10 |
```

d. resource/views/products/baby-kid.blade.php

```
resources > views > products > 🧠 baby-kid.blade.php > ...
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4  |   <title>Beauty & Health</title>
5  </head>
6  <body>
7  |   <h1>Beauty & Health Products</h1>
8  </body>
9  </html>
```

e. resource/views/products/home-care.blade.php

```
resources > views > products > 🧠 home-care.blade.php > ...
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4  |   <title>Home Care</title>
5  </head>
6  <body>
7  |   <h1>Home Care Products</h1>
8  </body>
9  </html>
10 |
```

f. resource/views/sales/index.blade.php

```

1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4  |   <title>Sales</title>
5  </head>
6  <body>
7  |   <h1>Point of Sales (POS)</h1>
8  </body>
9  </html>
10

```

g. resource/views/user/profil.blade.php

```

resources > views > user > 🐾 profile.blade.php > ...
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4  |   <title>User Profile</title>
5  </head>
6  <body>
7  |   <h1>User Profile</h1>
8  |   <p>ID: {{ $id }}</p>
9  |   <p>Name: {{ $name }}</p>
10 </body>
11 </html>
12

```

7. Hasil

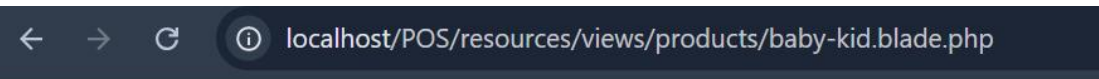
Welcome to POS

Product Categories

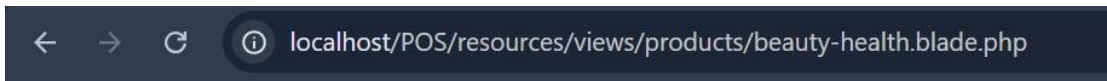
- [Food & Beverages](#)
- [Beauty & Health](#)
- [Home Care](#)
- [Baby & Kids](#)

Index of /POS/resources/views/products

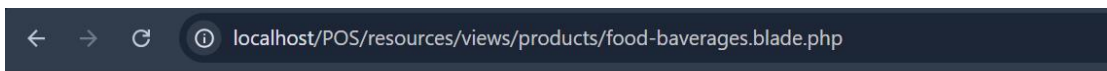
- [Parent Directory](#)
- [baby-kid.blade.php](#)
- [beauty-health.blade.php](#)
- [food-baverages.blade.php](#)
- [home-care.blade.php](#)



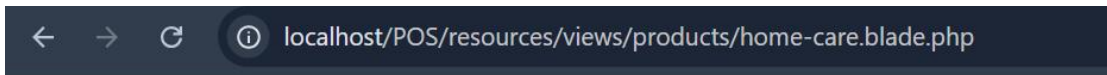
Beauty & Health Products



Beauty & Health Products



Food & Beverage Products



Home Care Products

Github : <https://github.com/snowvann/POS>