Web Programming Advanced Job Sheet 2

From:

AL AZHAR RIZQI RIFA’I FIRDAUS

Class:

2 I

Absence:

01

Student Number Identity:

2241720263

Department:

Information Technology

Study Program:

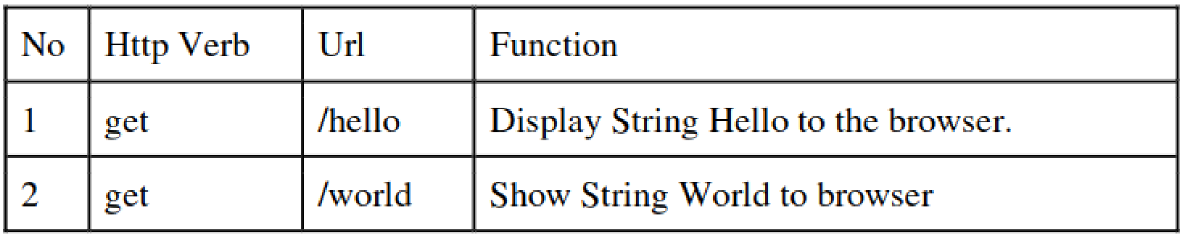
Informatics Engineering

Routing

Basic Routing

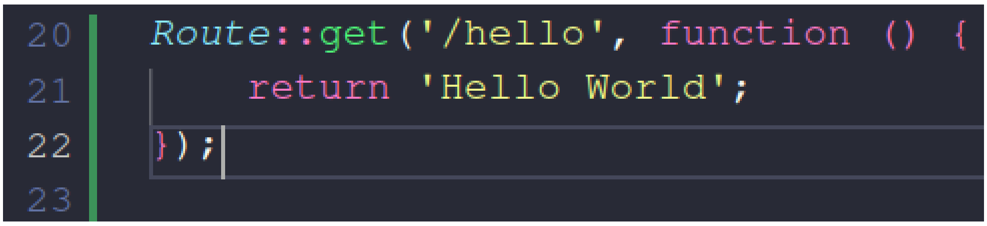
Practicum Steps:

a. In this section, we will create two routes with the following conditions.



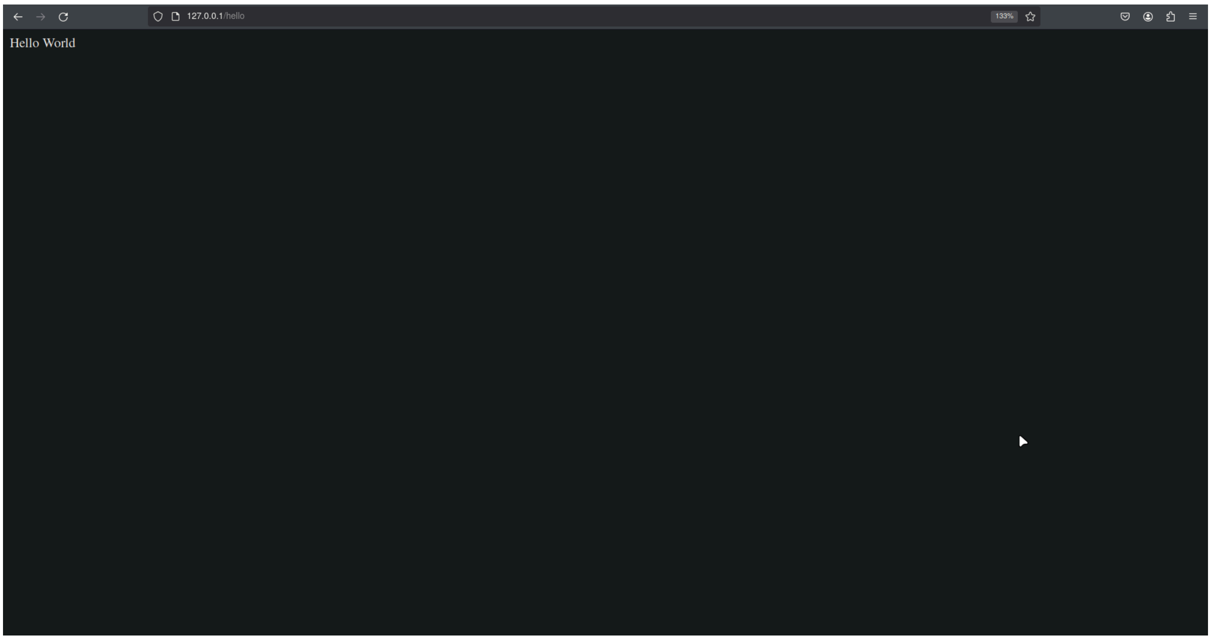
No	Http Verb	Url	Function
1	get	/hello	Display String Hello to the browser.
2	get	/world	Show String World to browser

b. Open the routes/web.php file. Add a route for number 1 as below:



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

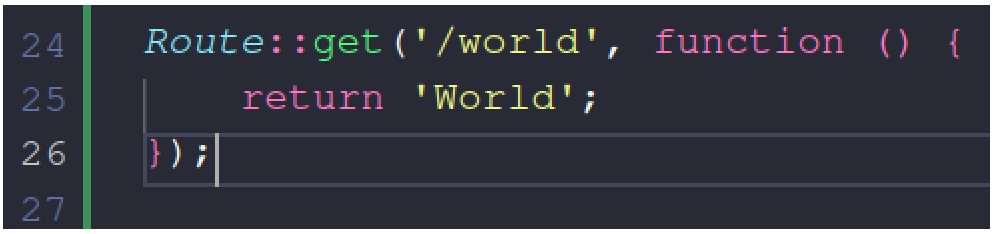
c. Open a browser, write the URL to call the route: localhost/PWL_2024/public/hello. Pay attention to the page that appears to see if it is appropriate and describe your observations.



```
Hello World
```

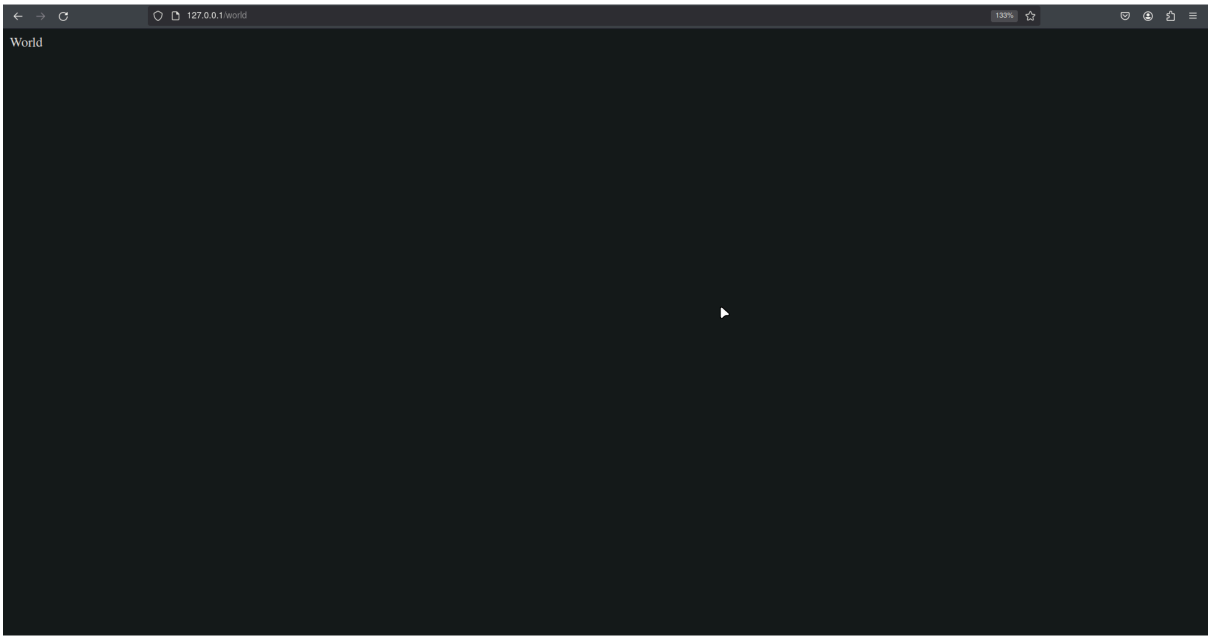
- When we write route with /hello path and get method, then we write function with return hello world message. When we type localhost:80/hello or 127.0.0.1:80/hello, it will be return hellow world message as same with that function return.

d. To create a second route, add the /world route as below:



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

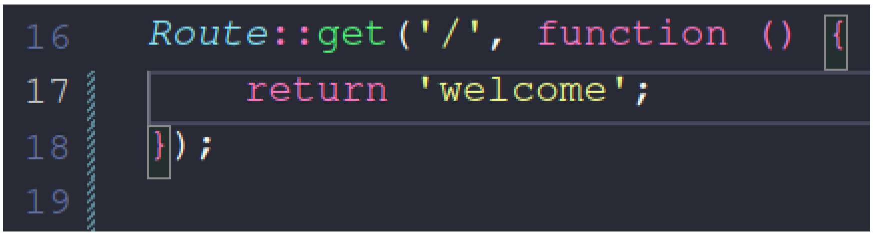
e. Open your browser, write the URL to call the route: localhost/PWL_2024/public/world. Pay attention to the page that appears to see if it is appropriate and describe your observations.



World

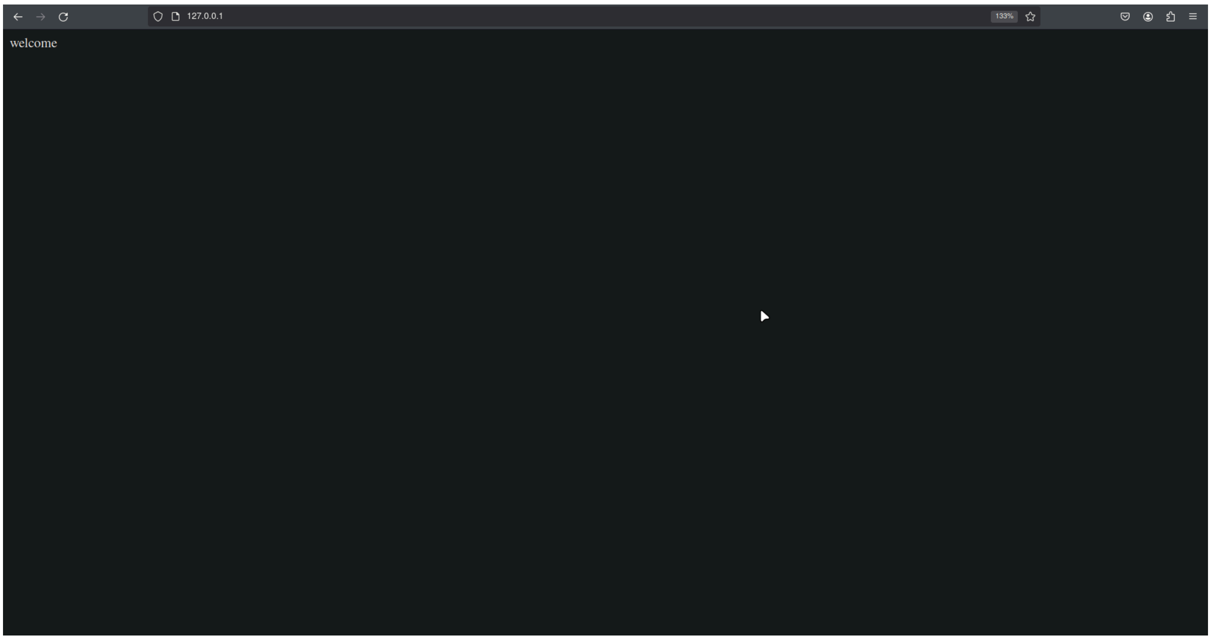
- Same with previous steps, when we change the URL path from /hello to /world, the message will be changed because both functions return a different message.

f. Code :



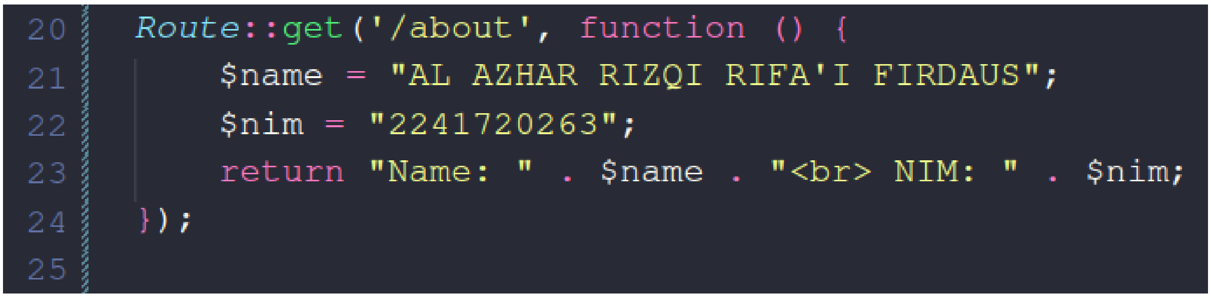
```
Route::get('/', function () {
    return 'welcome';
});
```

Result :



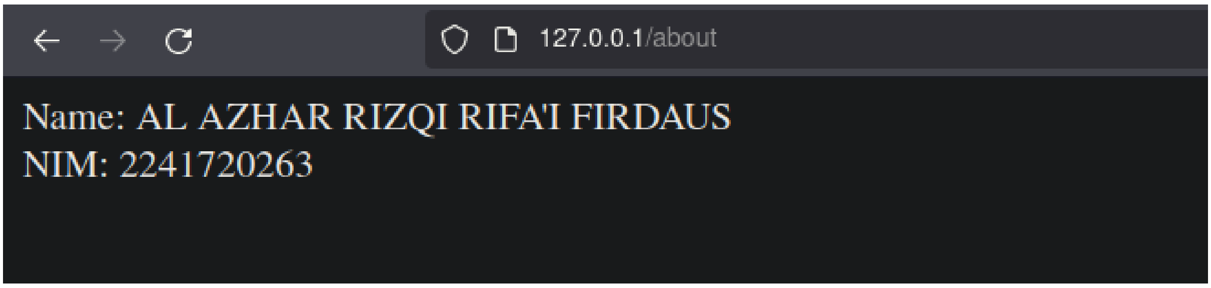
welcome

g. Code :



```
Route::get('/about', function () {
    $name = "AL AZHAR RIZQI RIFA'I FIRDAUS";
    $nim = "2241720263";
    return "Name: " . $name . "<br> NIM: " . $nim;
});
```

Result :

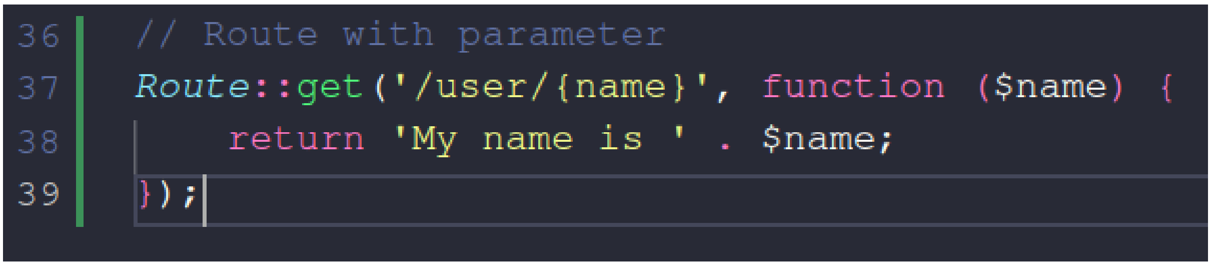


```
Name: AL AZHAR RIZQI RIFA'I FIRDAUS
NIM: 2241720263
```

Route Parameters

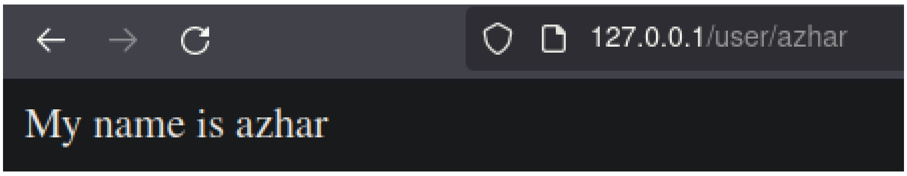
Practicum Steps:

a. We will call route /user/{name} as well as send parameters in the form of user names $name as the code below.



```php
// Route with parameter
Route::get('/user/{name}', function ($name) {
    return 'My name is ' . $name;
});
```

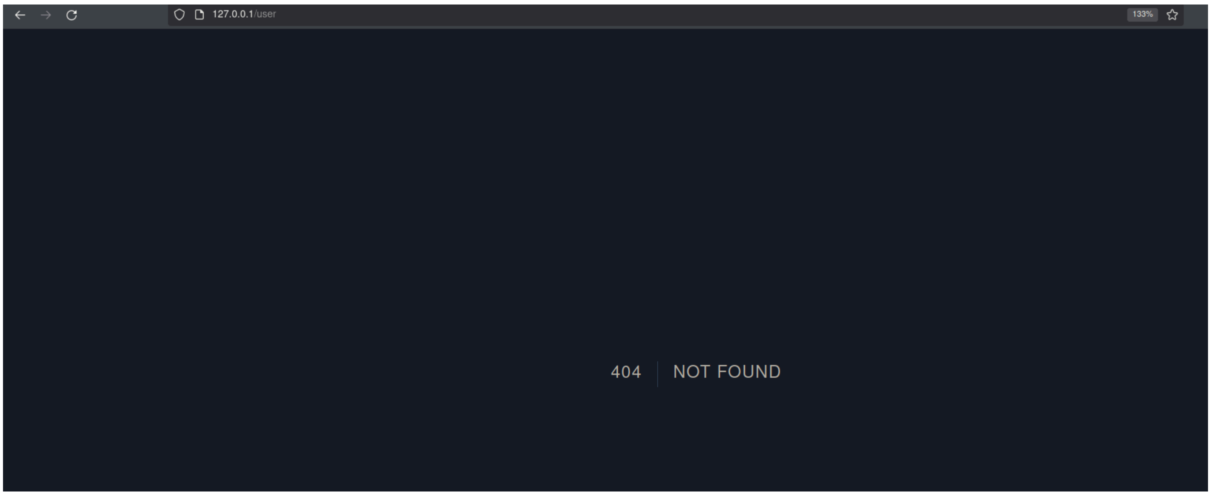
b. Run the code by writing the URL to call the route: localhost/PWL_2024/public/user/yourname. Pay attention to the page that appears and describe your observations.



My name is azhar

- Different with previous steps, we write url path with /user/{name}, it means “name” is parameter that fetch from user input. Then on callback function, there is name parameter, it receive from url parameter, then it will be return on callback function it self as message.

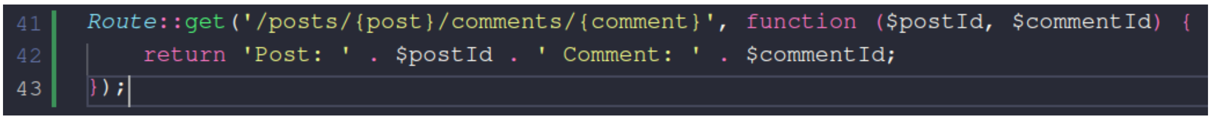
c. Next, try writing the URL: localhost/PWL_2024/public/user/. Pay attention to the page that appears and describe your observations.



404 | NOT FOUND

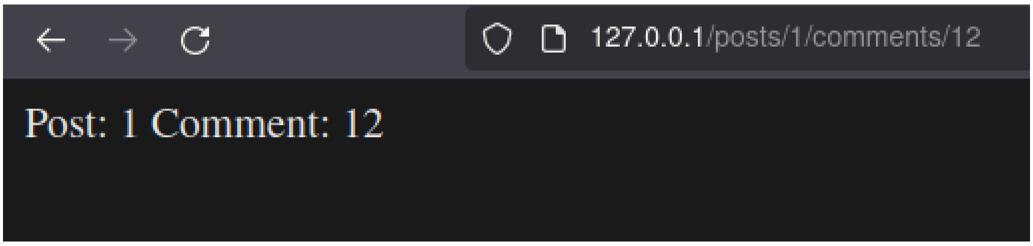
- The result is 404 not found. It means url path that client request which is /user is doesn’t exist on the server because we don’t already write the /user route.

d. A route, can also accept more than 1 parameter as the following code. The route accepts $postId parameters and also $comment.



```php
Route::get('/posts/{post}/comments/{comment}', function ($postId, $commentId) {
    return 'Post: ' . $postId . ' Comment: ' . $commentId;
});
```

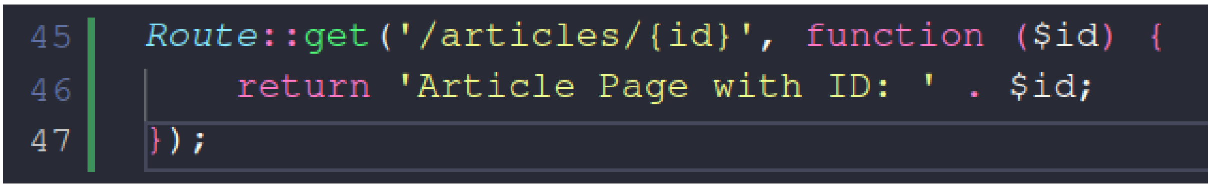
e. Run the code by writing the URL to call the route: localhost/PWL_2024/public/posts/1/comments/5. Pay attention to the page that appears and describe your observations.



```
Post: 1 Comment: 12
```

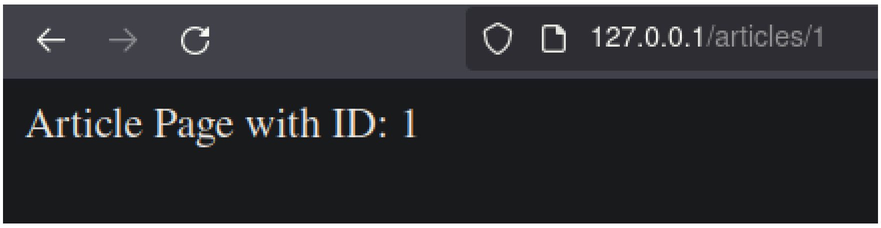
- Same with before, we add one more parameter on route.

f. Code :



```php
Route::get('/articles/{id}', function ($id) {
    return 'Article Page with ID: ' . $id;
});
```

Result :

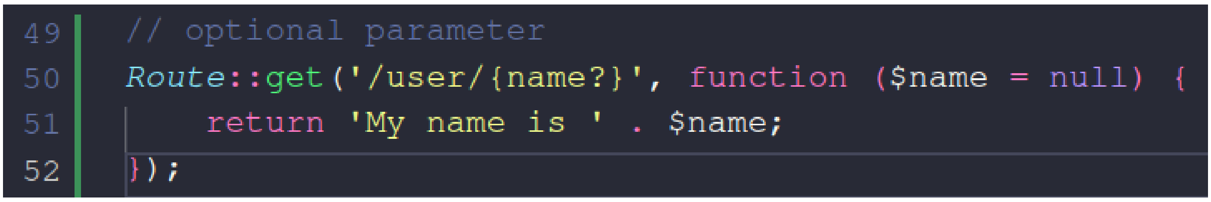


```
Article Page with ID: 1
```

Optional Parameters

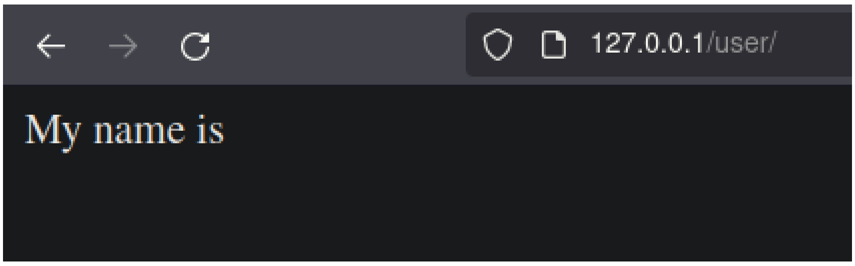
Practicum Steps:

a. We will call route /user as well as send a parameter in the form of a username $name where the parameter is optional.



```php
// optional parameter
Route::get('/user/{name?}', function ($name = null) {
    return 'My name is ' . $name;
});
```

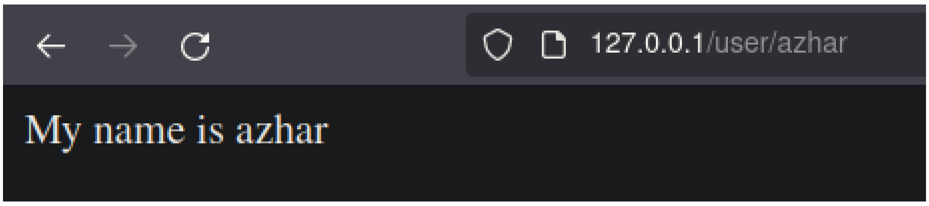
b. Run the code by typing the URL: localhost/PWL_2024/public/user/. Pay attention to the page that appears and describe your observations.



```
My name is
```

- Different with before, when we type url path /user is not found, now there is message result because we set on name parameter the default is null on callback function.

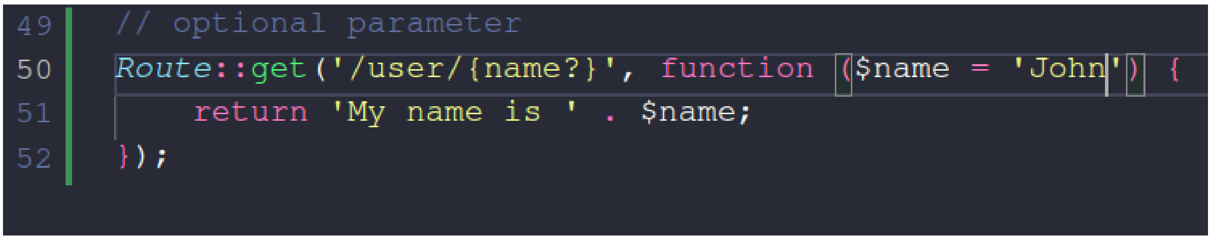
c. Next write the URL: localhost/PWL_2024/public/user/yourname. Pay attention to the page that appears and describe your observations.



My name is azhar

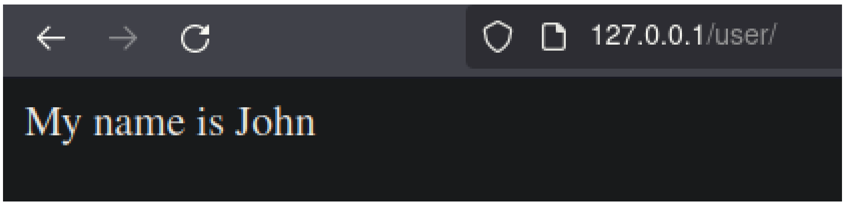
- The result still appear because the default value from name parameter is null. when we type anything we want, the result will be the same as we type.

d. Change the code in the /user route to as below.



```
// optional parameter
Route::get('/user/{name?}', function ($name = 'John') {
    return 'My name is ' . $name;
});
```

e. Run the code by typing the URL: localhost/PWL_2024/public/user/. Pay attention to the page that appears and describe your observations.



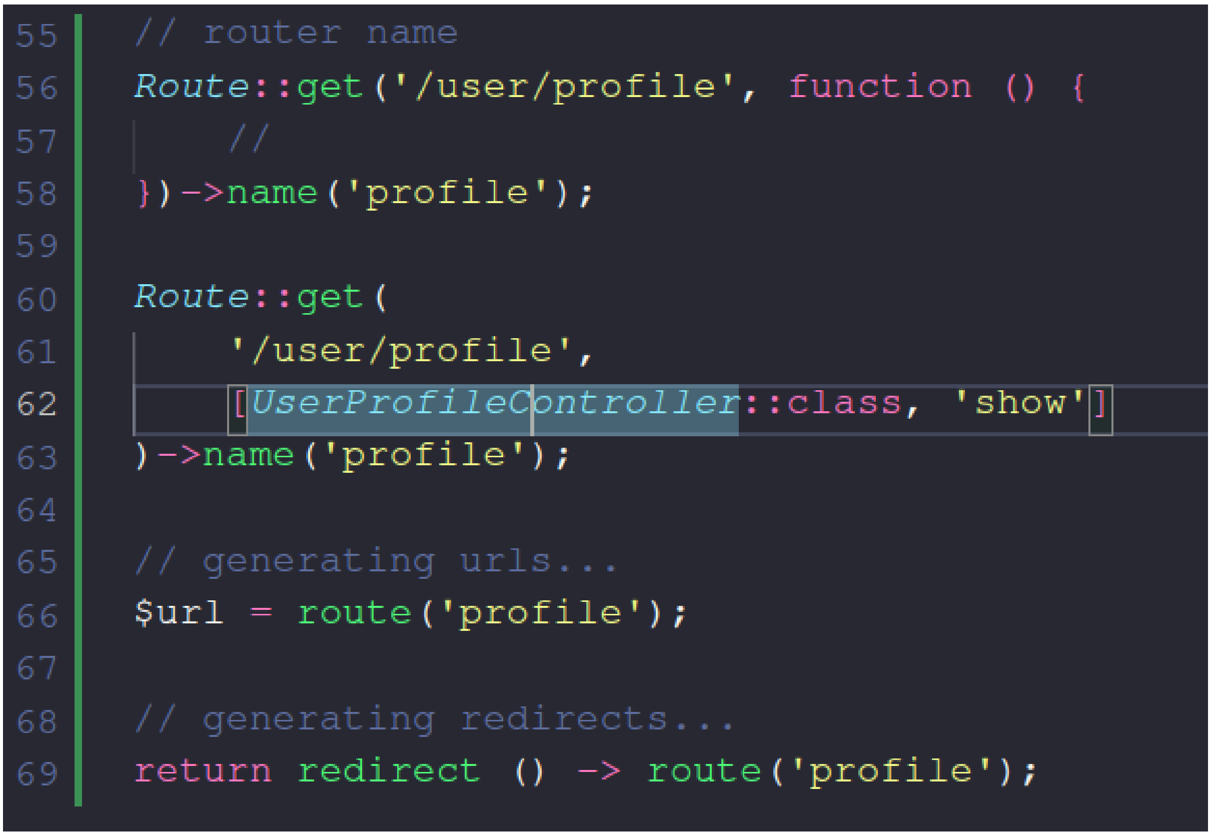
My name is John

- The result is John, because we set the default value on name parameter is john.

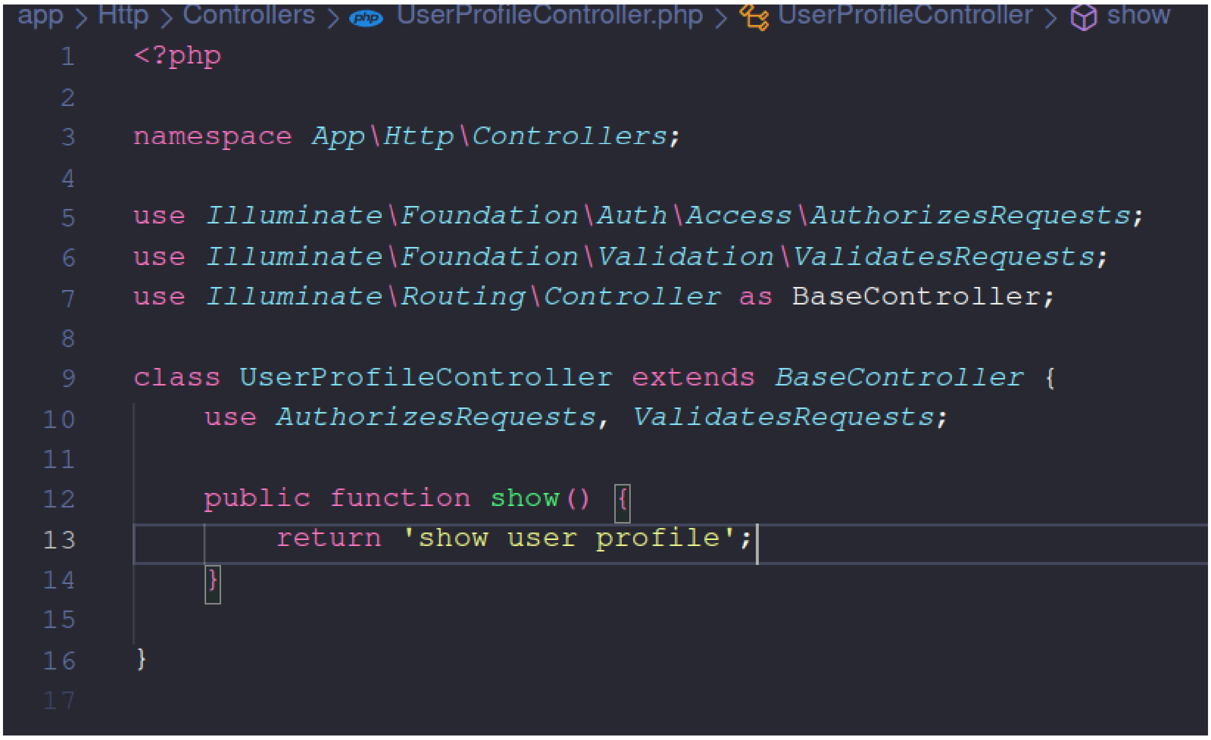
Router Name

Route names are usually used to make it easier for us to call routes when building applications. We simply call the name of the route.

- Code :



```
// router name
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');
```

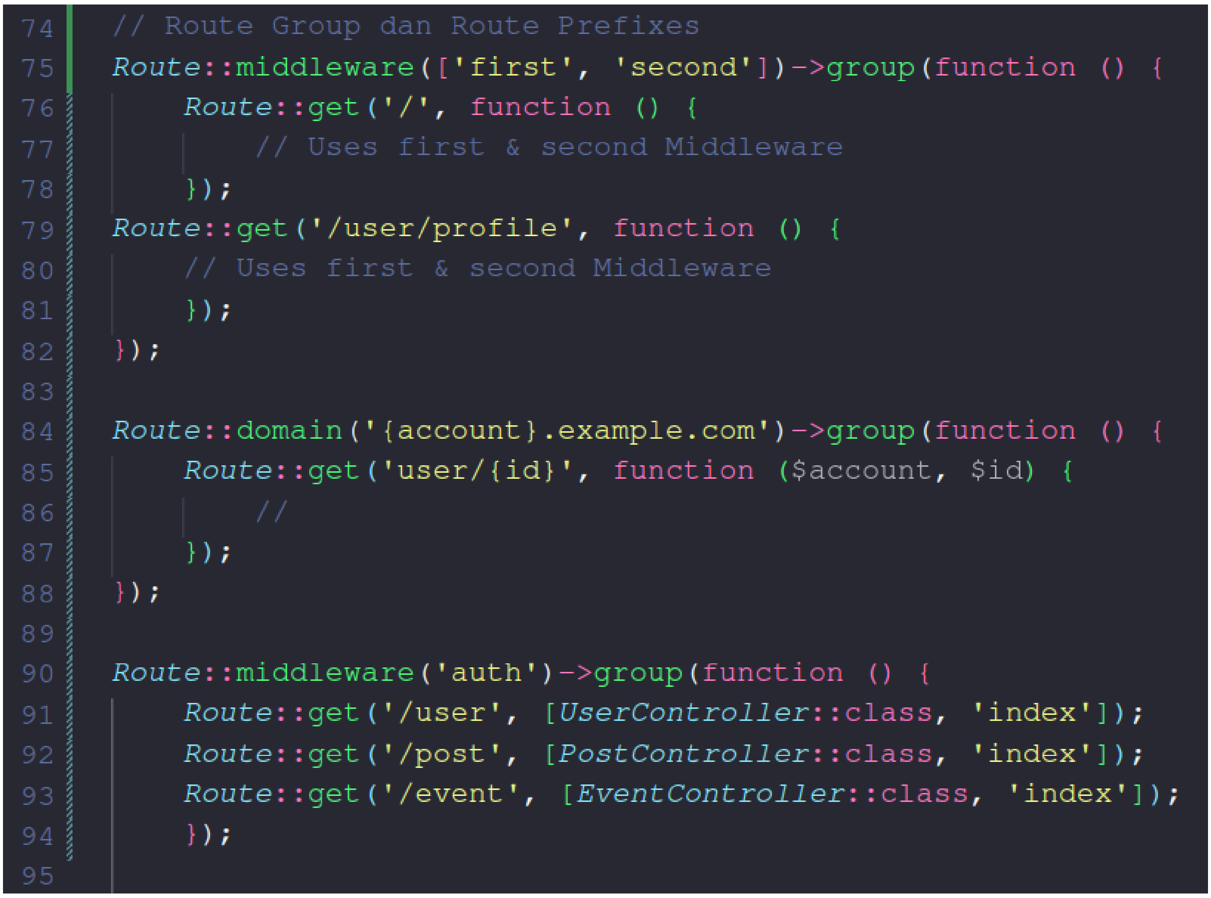


app > Http > Controllers > UserProfileController.php > UserProfileController > show

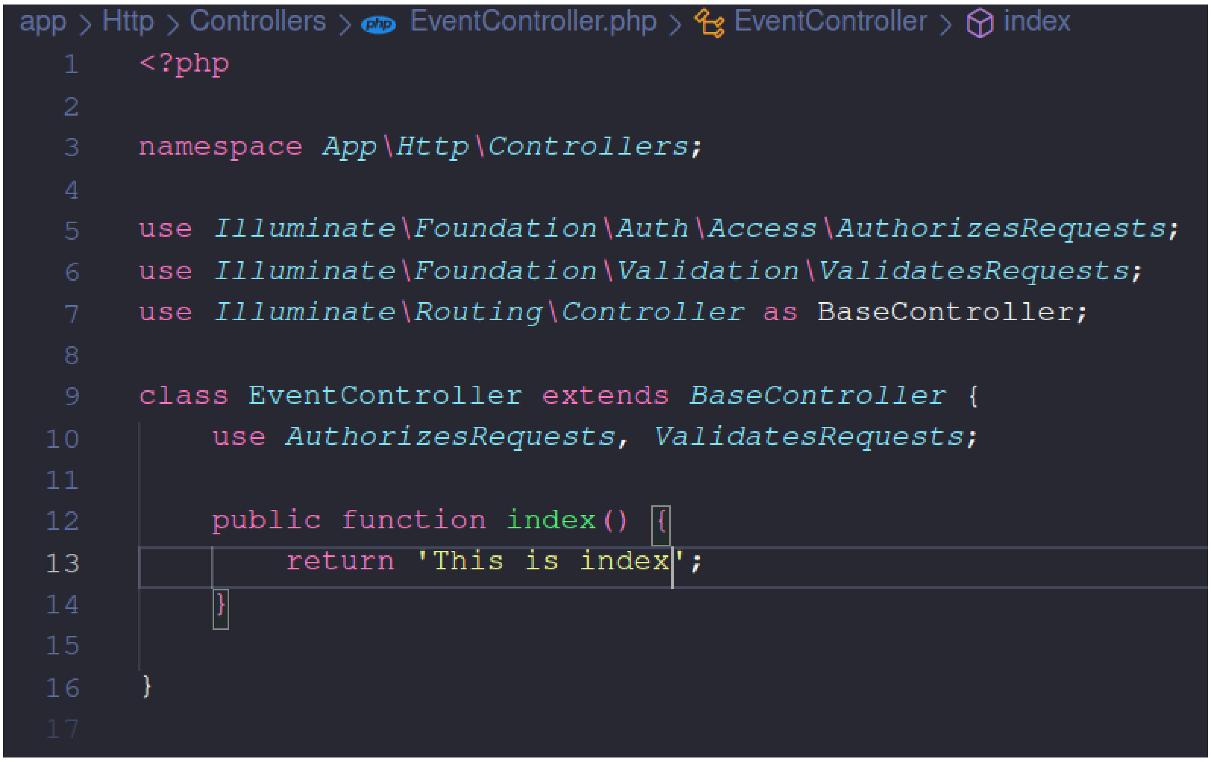
```
<?php

namespace App\Http\Controllers;

use Illuminate\Foundation\Auth\Access\AuthorizesRequests;
use Illuminate\Foundation\Validation\ValidatesRequests;
use Illuminate\Routing\Controller as BaseController;

class UserProfileController extends BaseController {
    use AuthorizesRequests, ValidatesRequests;

    public function show() {
        return 'show user profile';
    }

}
```

Route Group dan Route Prefixes

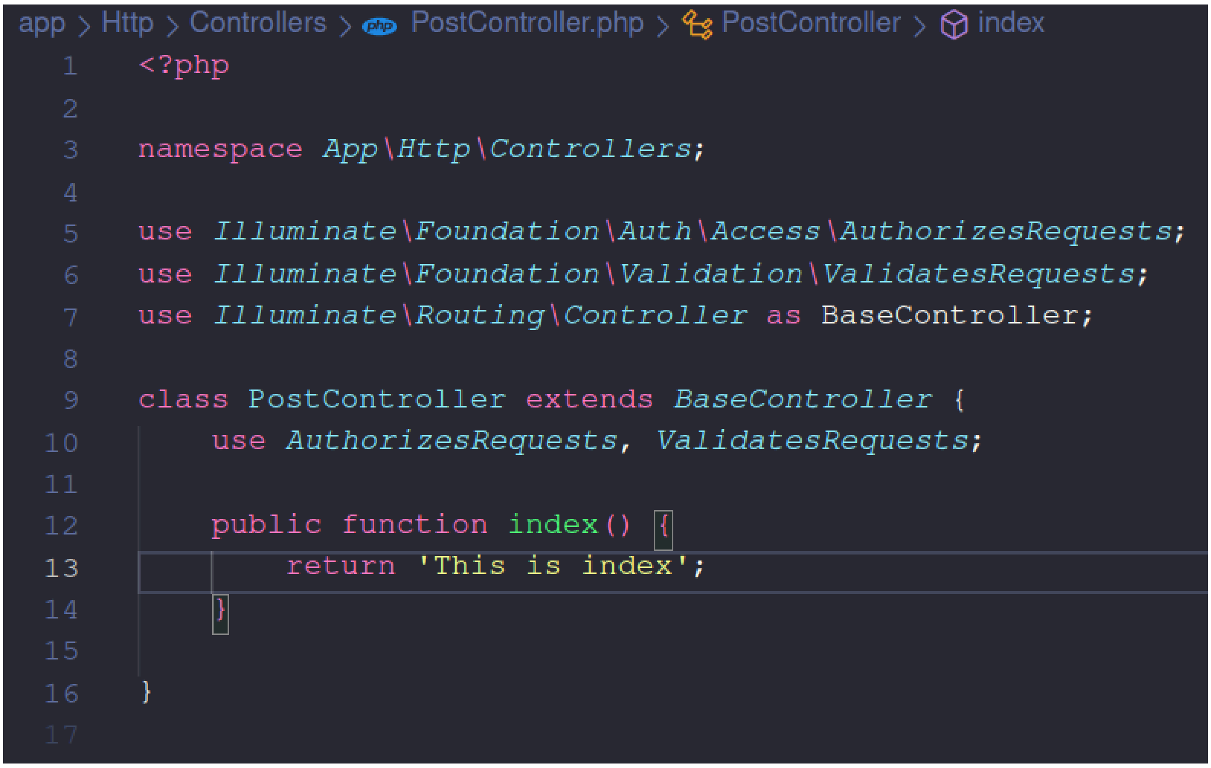
Several routes that have the same attributes as the same middleware can be grouped into one group to make it easier to write routes, in addition to being used for middleware, there is still the use of route groups for routes that are under one subdomain. Examples of using route groups are as follows:



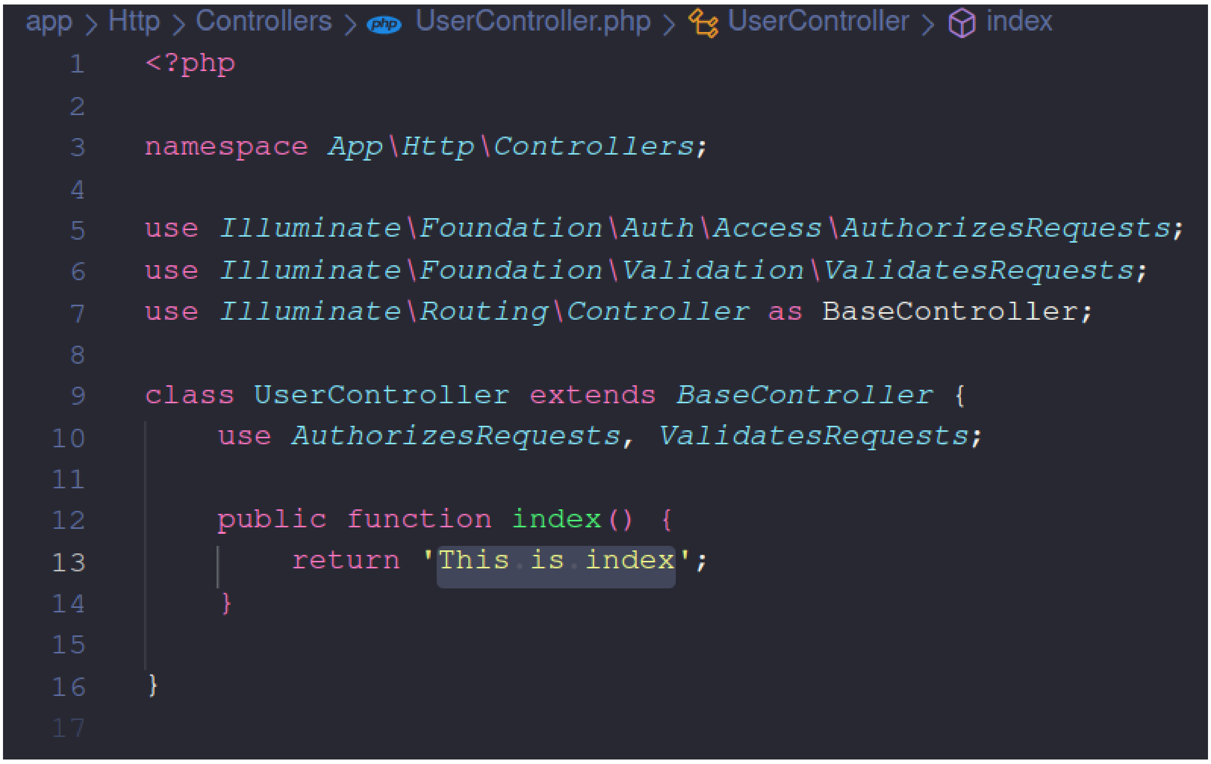
```
// Route Group dan Route Prefixes
Route::middleware(['first', 'second'])->group(function () {
    Route::get('/', function () {
        // Uses first & second Middleware
    });
Route::get('/user/profile', function () {
    // Uses first & second Middleware
    });
});

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']);
    });

```


app > Http > Controllers > EventController.php > EventController > index

```php
<?php

namespace App\Http\Controllers;

use Illuminate\Foundation\Auth\Access\AuthorizesRequests;
use Illuminate\Foundation\Validation\ValidatesRequests;
use Illuminate\Routing\Controller as BaseController;

class EventController extends BaseController {
    use AuthorizesRequests, ValidatesRequests;

    public function index() {
        return 'This is index';
    }

}
```


app > Http > Controllers > PostController.php > PostController > index

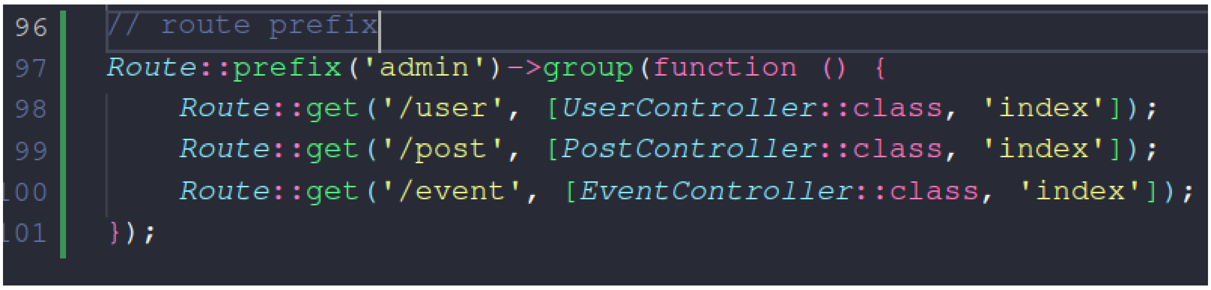
```php
<?php

namespace App\Http\Controllers;

use Illuminate\Foundation\Auth\Access\AuthorizesRequests;
use Illuminate\Foundation\Validation\ValidatesRequests;
use Illuminate\Routing\Controller as BaseController;

class PostController extends BaseController {
    use AuthorizesRequests, ValidatesRequests;

    public function index() {
        return 'This is index';
    }

}
```


app > Http > Controllers > UserController.php > UserController > index

```php
<?php

namespace App\Http\Controllers;

use Illuminate\Foundation\Auth\Access\AuthorizesRequests;
use Illuminate\Foundation\Validation\ValidatesRequests;
use Illuminate\Routing\Controller as BaseController;

class UserController extends BaseController {
    use AuthorizesRequests, ValidatesRequests;

    public function index() {
        return 'This is index';
    }

}
```

Route Prefix

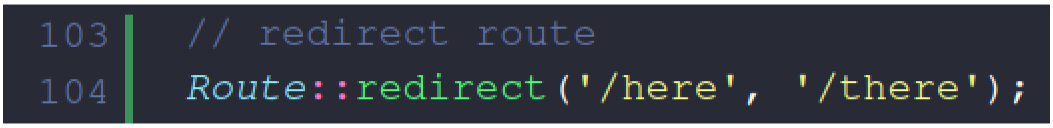
Route grouping can also be done for routes that have the same prefix. For route creation with prefix can be seen the code as below.



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

Redirect Routes

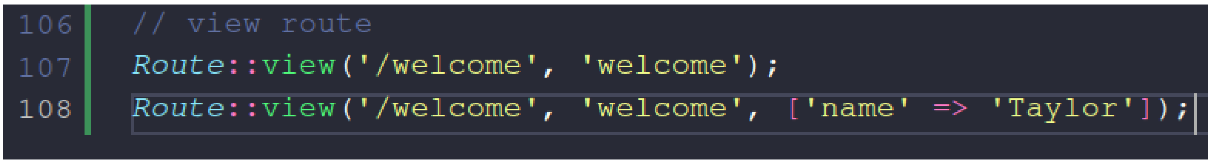
To redirect Laravel can be done using Route::redirect how to use it can be seen in the program code below.



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

View Routes

Laravel also provides a special route that makes it easy to create a route without using controllers or callback functions. These routes directly accept input in the form of urls and return views. Here's how to create view routes.



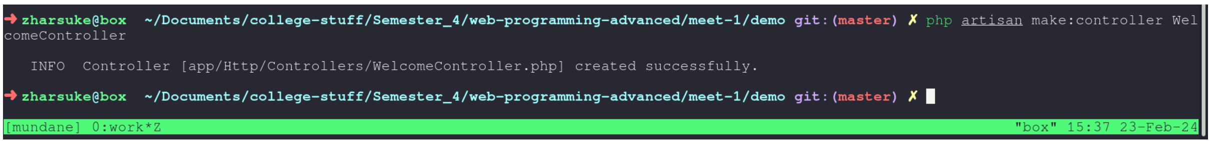
```
// view route
Route::view('/welcome', 'welcome');
Route::view('/welcome', 'welcome', ['name' => 'Taylor']);
```

Controller

Creating a Controller

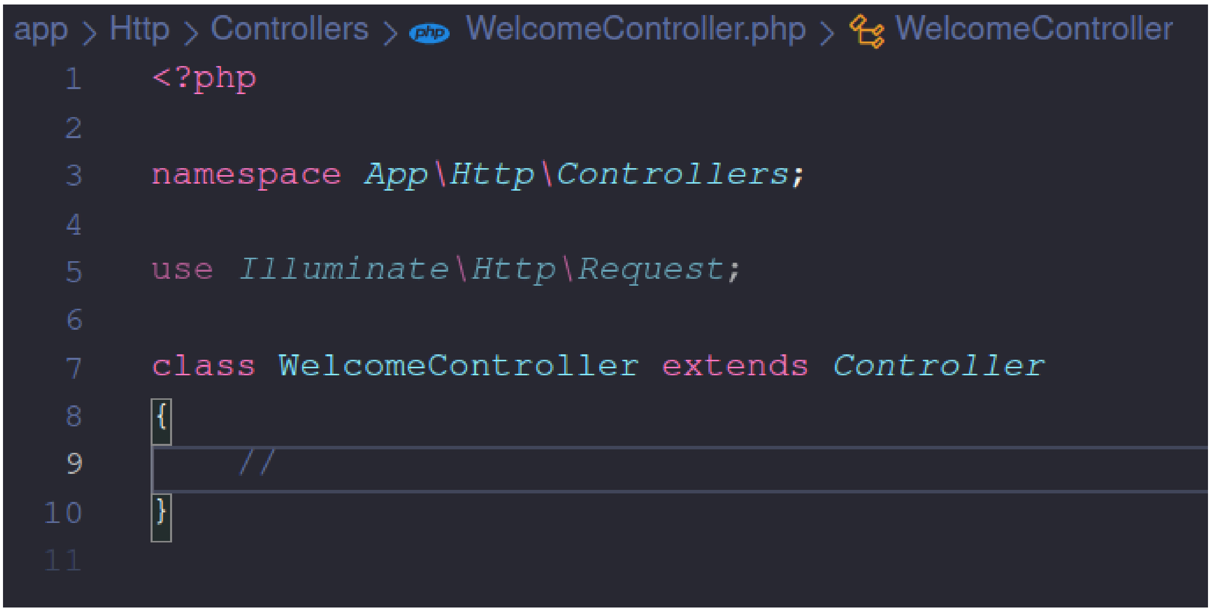
Practicum Steps:

a. To create a controller in Laravel a command is provided to generate its basic structure. We can use the artisan command followed by the definition of the name of the controller to be created.



```
➜ zharsuke@box  ~/Documents/college-stuff/Semester_4/web-programming-advanced/meet-1/demo git:(master) ✗ php artisan make:controller Wel
comeController

   INFO  Controller [app/Http/Controllers/WelcomeController.php] created successfully.

➜ zharsuke@box  ~/Documents/college-stuff/Semester_4/web-programming-advanced/meet-1/demo git:(master) ✗ █
[mundane] 0:work*Z                                                                                    "box" 15:37 23-Feb-24
```

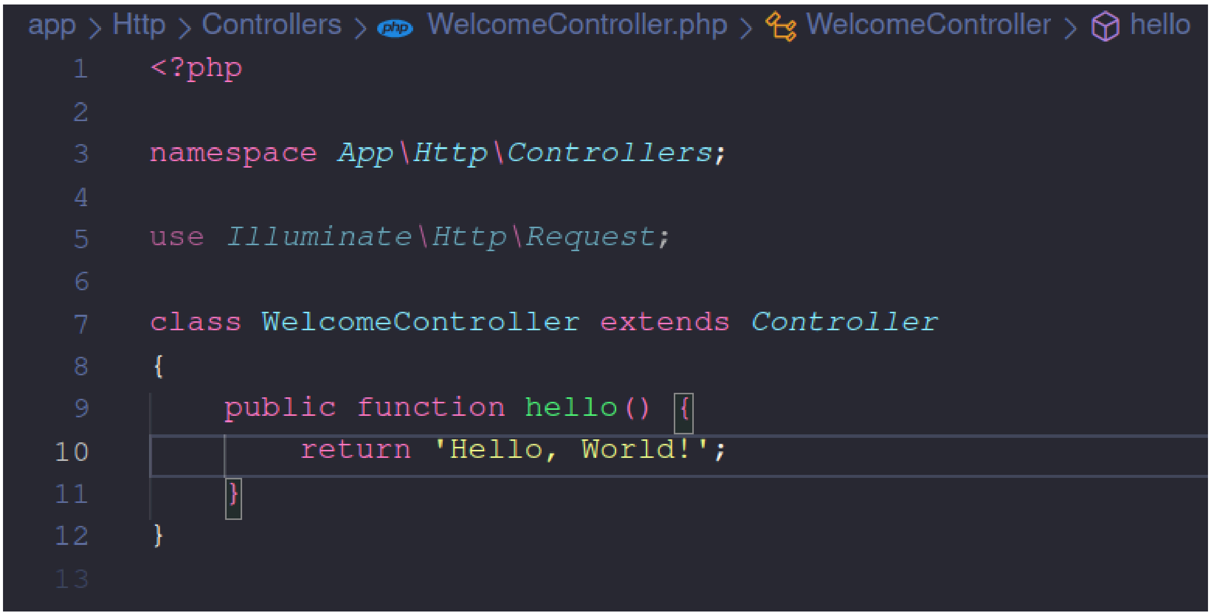
b. Open the file in app/Http/Controllers/WelcomeController.php. The structure on the controller can be described as follows:



app > Http > Controllers > WelcomeController.php > WelcomeController

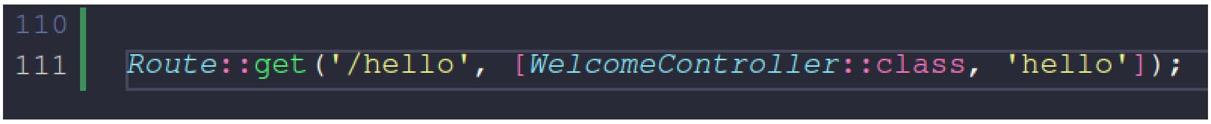
```php
<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;

class WelcomeController extends Controller
{
    //
}

```

c. To define an action, please add a function with public access. So that the controller above becomes as follows:


app > Http > Controllers > WelcomeController.php > WelcomeController > hello

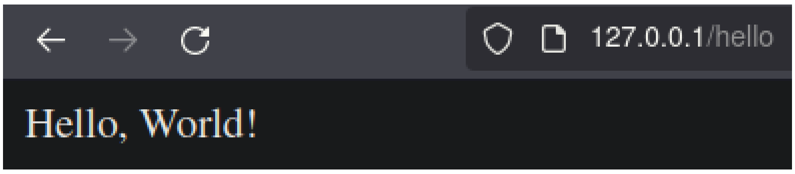
```php
<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;

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

d. Once a controller has defined an action, we can add that controller to the route. Change the /hello route to something like this:



```php

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

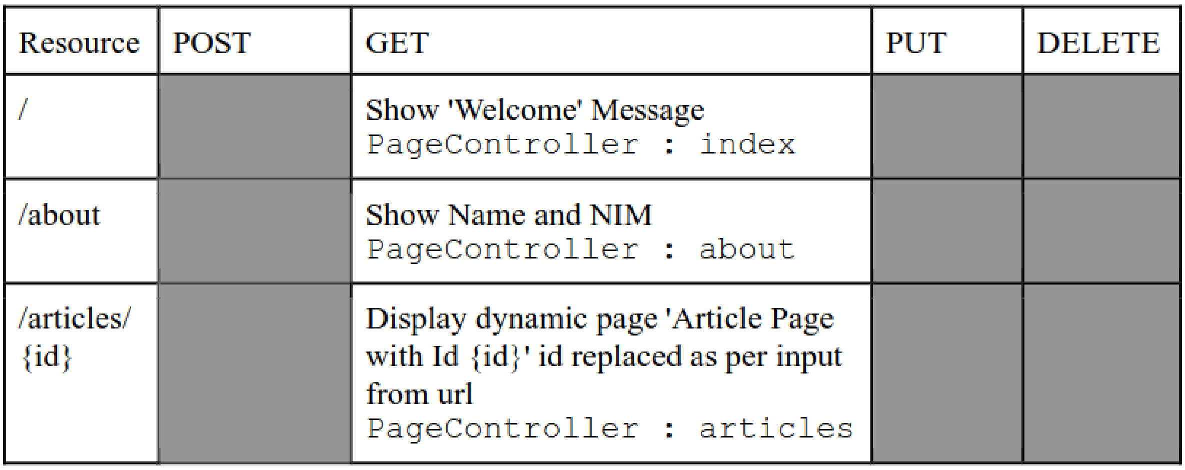
e. Open a browser, write the URL to call the route: localhost/PWL_2024/public/hello. Pay attention to the page that appears and describe your observations.



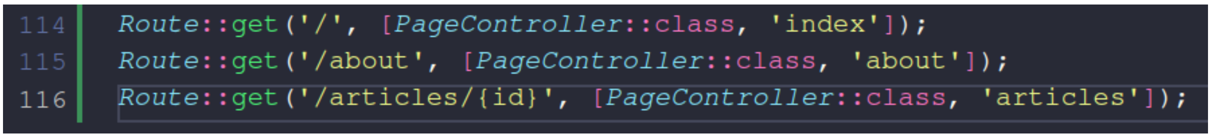
Hello, World!

- Different with before, we write route with callback function. But now we separated it, store the function inside controller.

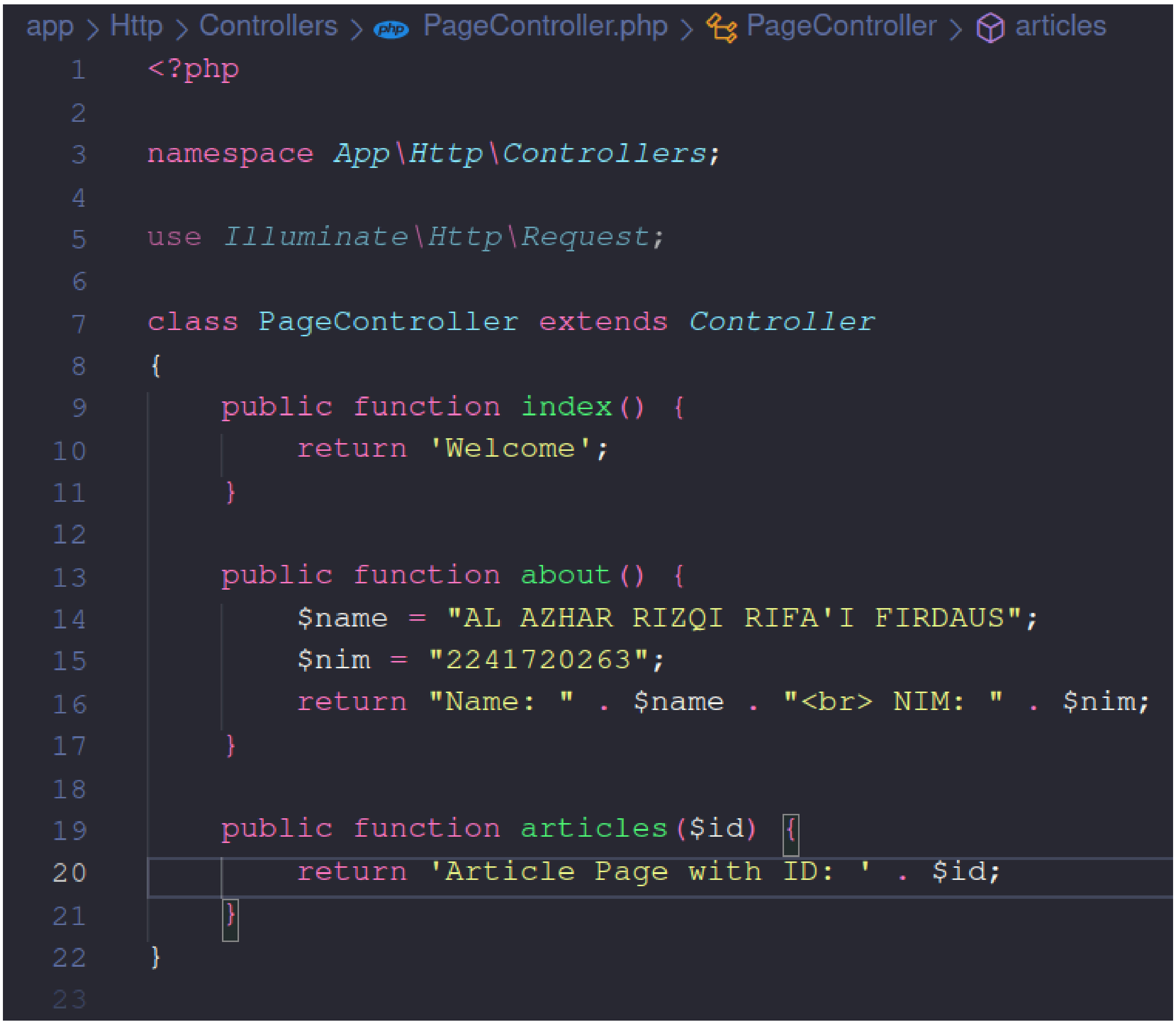
f. Modification of results in practicum point 2 (Routing) with the concept of controller. Move the execution logic into the controller with the name PageController.



Resource	POST	GET	PUT	DELETE
/		Show 'Welcome' Message `PageController : index`		
/about		Show Name and NIM `PageController : about`		
/articles/{id}		Display dynamic page 'Article Page with Id {id}' id replaced as per input from url `PageController : articles`		

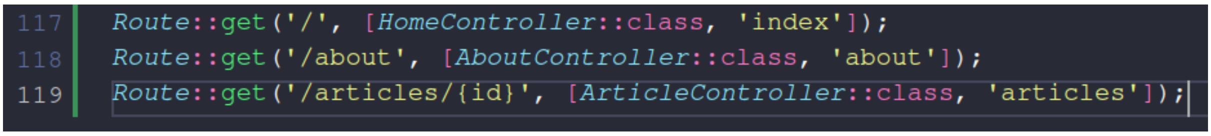


```php
Route::get('/', [PageController::class, 'index']);
Route::get('/about', [PageController::class, 'about']);
Route::get('/articles/{id}', [PageController::class, 'articles']);
```

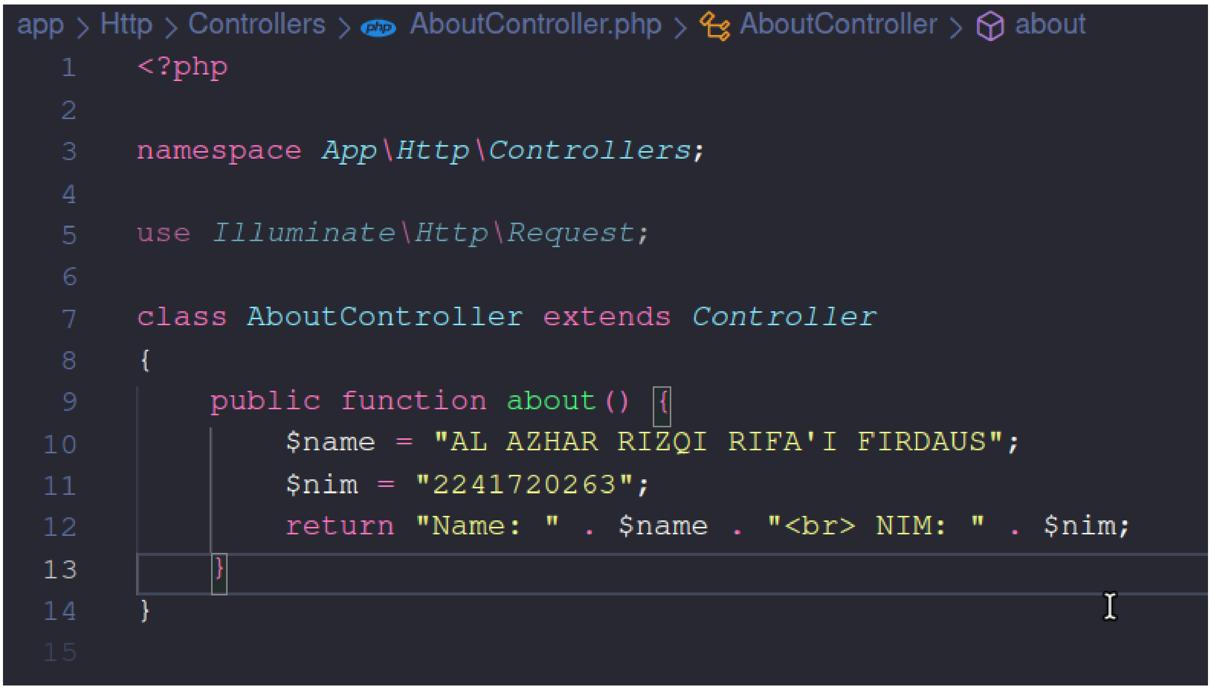

app > Http > Controllers > PageController.php > PageController > articles

```php
<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;

class PageController extends Controller
{
    public function index() {
        return 'Welcome';
    }

    public function about() {
        $name = "AL AZHAR RIZQI RIFA'I FIRDAUS";
        $nim = "2241720263";
        return "Name: " . $name . "<br> NIM: " . $nim;
    }

    public function articles($id) {
        return 'Article Page with ID: ' . $id;
    }
}
```

g. Modify the previous implementation with the concept of Single Action Controller. So for the final result obtained there will be HomeController, AboutController and ArticleController. Modifications are also the routes used.

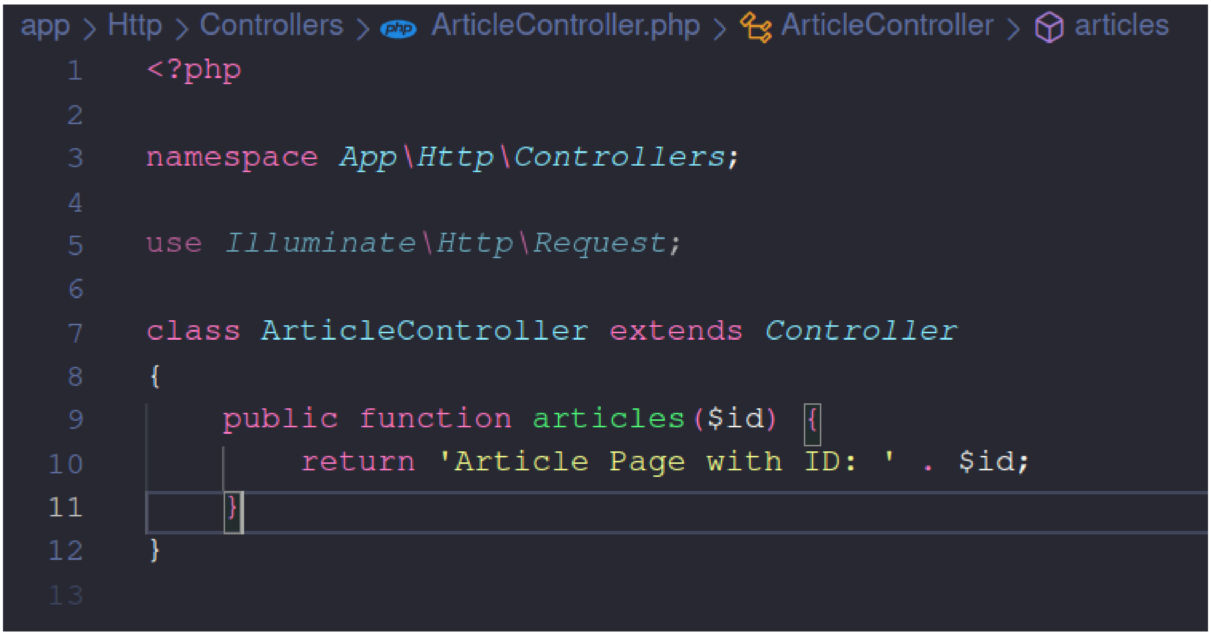


```php
Route::get('/', [HomeController::class, 'index']);
Route::get('/about', [AboutController::class, 'about']);
Route::get('/articles/{id}', [ArticleController::class, 'articles']);
```



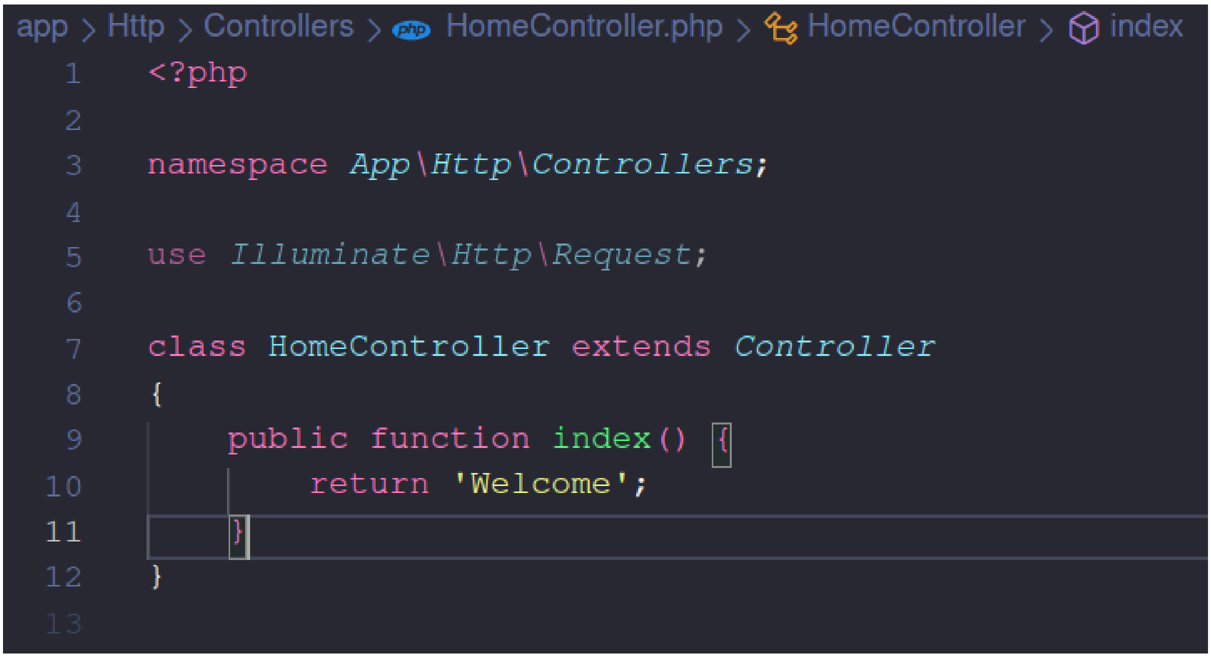
app > Http > Controllers > AboutController.php > AboutController > about

```php
<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;

class AboutController extends Controller
{
    public function about() {
        $name = "AL AZHAR RIZQI RIFA'I FIRDAUS";
        $nim = "2241720263";
        return "Name: " . $name . "<br> NIM: " . $nim;
    }
}
```



app > Http > Controllers > ArticleController.php > ArticleController > articles

```php
<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;

class ArticleController extends Controller
{
    public function articles($id) {
        return 'Article Page with ID: ' . $id;
    }
}
```



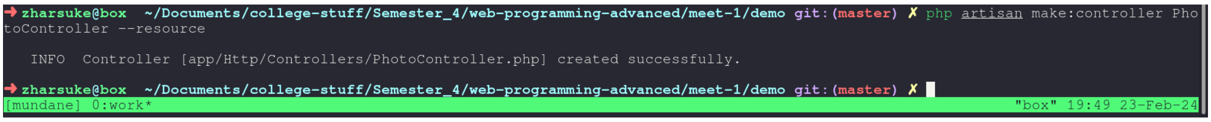
app > Http > Controllers > HomeController.php > HomeController > index

```php
<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;

class HomeController extends Controller
{
    public function index() {
        return 'Welcome';
    }
}
```

Resource Controller

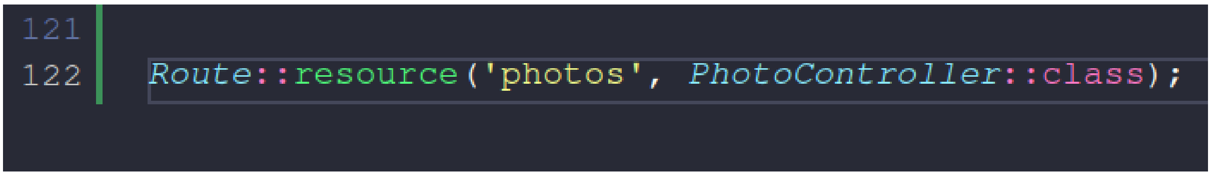
Practicum Steps:

a. To create it is done by running the following command in the terminal.



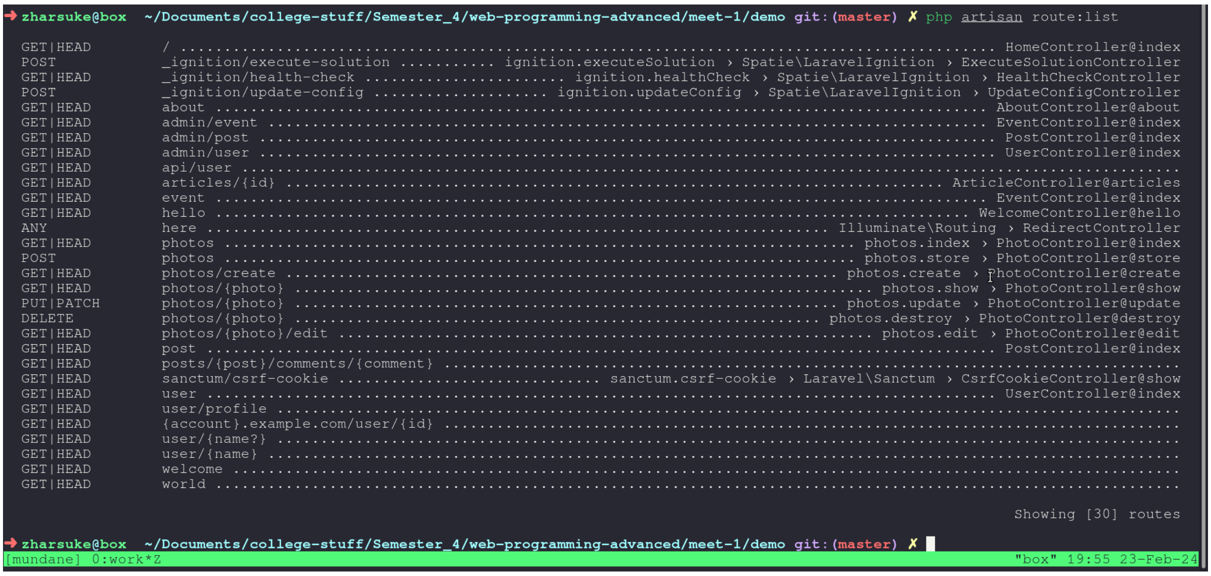
```
➜ zharsuke@box  ~/Documents/college-stuff/Semester_4/web-programming-advanced/meet-1/demo git:(master) ✗ php artisan make:controller Pho
toController --resource

   INFO  Controller [app/Http/Controllers/PhotoController.php] created successfully.

➜ zharsuke@box  ~/Documents/college-stuff/Semester_4/web-programming-advanced/meet-1/demo git:(master) ✗
[mundane] 0:work*                                                                                            "box" 19:49 23-Feb-24
```

b. After the controller is successfully degenerated, a route must be created so that it can connect to the frontend. Add the following program code to the web.php file.



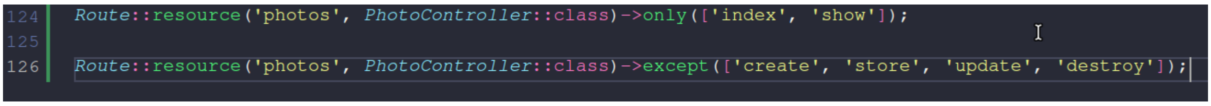
```php

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

c. Run the list route check (php artisan route:list) the following route will be generated.



```
➜ zharsuke@box  ~/Documents/college-stuff/Semester_4/web-programming-advanced/meet-1/demo git:(master) ✗ php artisan route:list

  GET|HEAD        / ........................................................................................ HomeController@index
  POST            _ignition/execute-solution ........... ignition.executeSolution › Spatie\LaravelIgnition › ExecuteSolutionController
  GET|HEAD        _ignition/health-check ....................... ignition.healthCheck › Spatie\LaravelIgnition › HealthCheckController
  POST            _ignition/update-config .................... ignition.updateConfig › Spatie\LaravelIgnition › UpdateConfigController
  GET|HEAD        about ................................................................................... AboutController@about
  GET|HEAD        admin/event ............................................................................. EventController@index
  GET|HEAD        admin/post ............................................................................... PostController@index
  GET|HEAD        admin/user ............................................................................... UserController@index
  GET|HEAD        api/user ......................................................................................................
  GET|HEAD        articles/{id} ................................................................... ArticleController@articles
  GET|HEAD        event ................................................................................... EventController@index
  GET|HEAD        hello ................................................................................. WelcomeController@hello
  ANY             here ............................................................... Illuminate\Routing › RedirectController
  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        post ..................................................................................... PostController@index
  GET|HEAD        posts/{post}/comments/{comment} ...............................................................................
  GET|HEAD        sanctum/csrf-cookie .............................. sanctum.csrf-cookie › Laravel\Sanctum › CsrfCookieController@show
  GET|HEAD        user ..................................................................................... UserController@index
  GET|HEAD        user/profile ..................................................................................................
  GET|HEAD        {account}.example.com/user/{id} ...............................................................................
  GET|HEAD        user/{name?} ..................................................................................................
  GET|HEAD        user/{name} ...................................................................................................
  GET|HEAD        welcome .......................................................................................................
  GET|HEAD        world .........................................................................................................

                                                                                                               Showing [30] routes

➜ zharsuke@box  ~/Documents/college-stuff/Semester_4/web-programming-advanced/meet-1/demo git:(master) ✗ 
[mundane] 0:work*Z                                                                                          "box" 19:55 23-Feb-24
```

d. In the route list all routes related to crud photos have been generated by laravel. If not all routes in the resource controller are needed, it can be reduced by updating the routes in theweb.php to resemble the following.



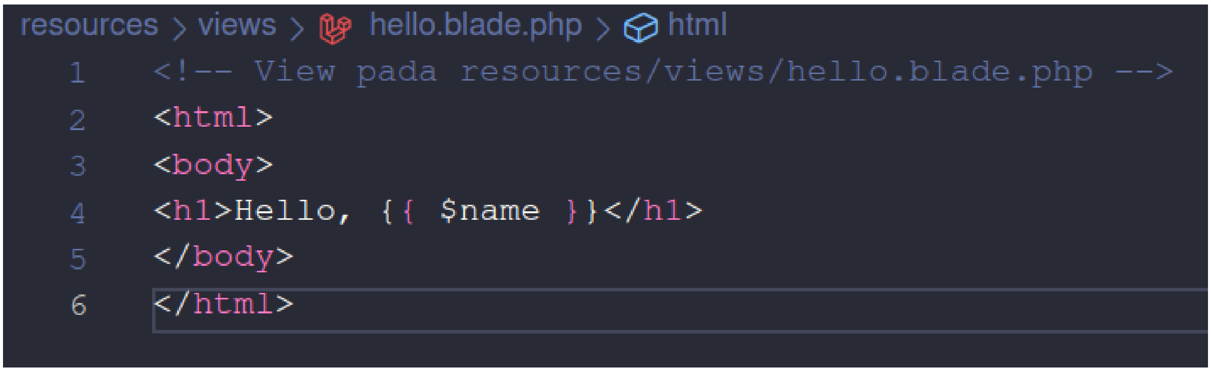
```php
Route::resource('photos', PhotoController::class)->only(['index', 'show']);

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

View

Creating a View

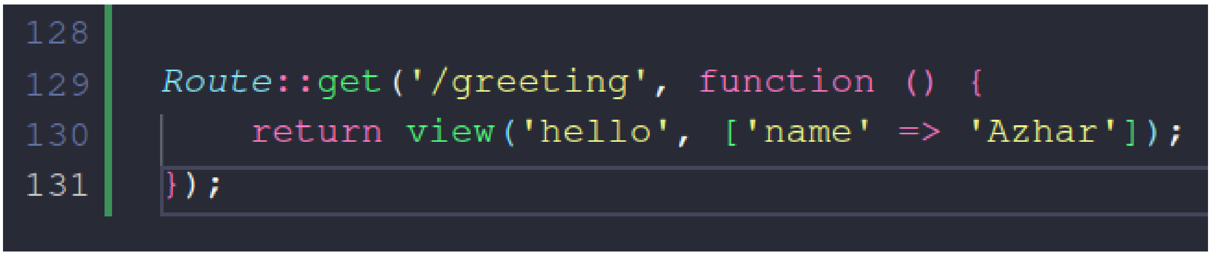
Practicum Steps:

a. In the app/resources/views directory, create a hello.blade.php file.



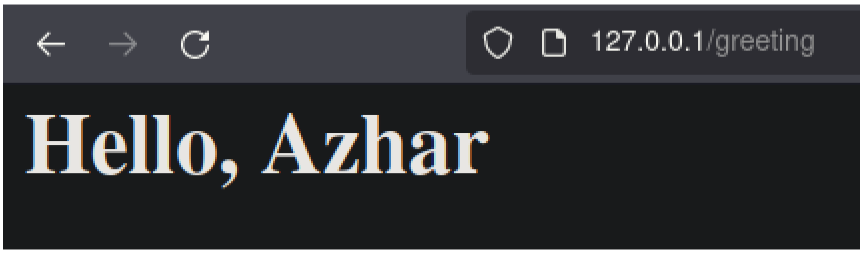
```html
<!-- View pada resources/views/hello.blade.php -->
<html>
<body>
<h1>Hello, {{ $name }}</h1>
</body>
</html>
```

b. The View can be run via Routing, where the route will call the View according to the file name without the 'blade.php'. (Note: Replace Andi with your name)



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

c. Run the code by opening the url localhost/PWL_2024/public/greeting. Pay attention to the page that appears and describe your observations.



127.0.0.1/greeting

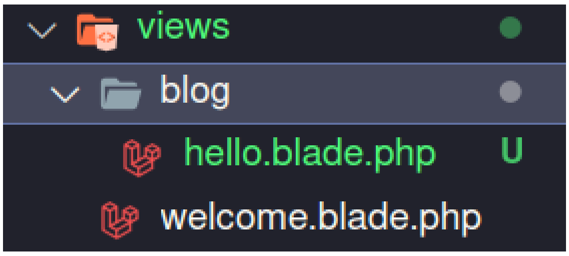
Hello, Azhar

- We create hello views to display web interface. Then in route, we write callback function, then return hello view and fill name parameter with Azhar.

View in Directory

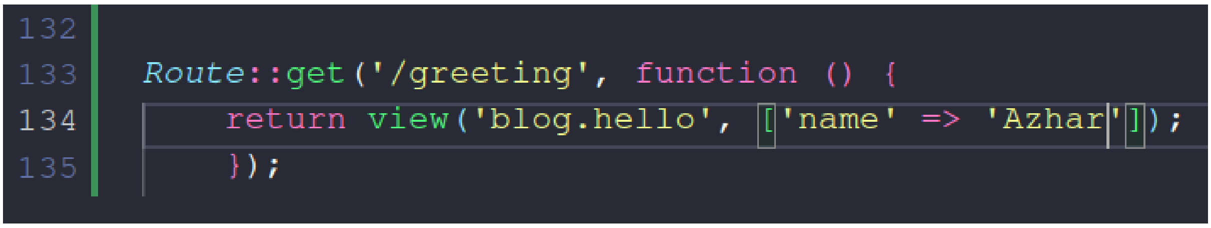
Practicum Steps:

a. Create a blog directory within the views directory.
b. Move the hello.blade.php file into the blog directory.



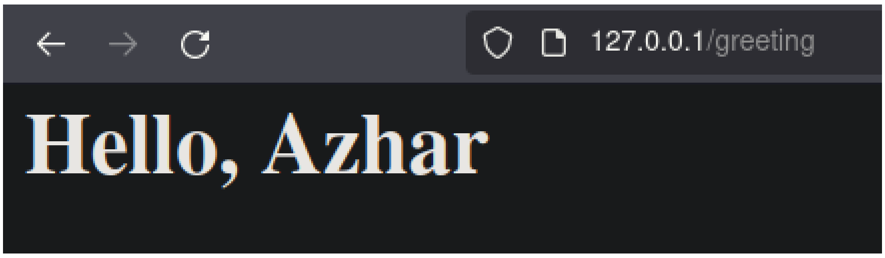
```
views
  blog
    hello.blade.php   U
  welcome.blade.php
```

c. Next, make changes to the route.



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

d. Run the code by opening the url localhost/PWL_2024/public/greeting. Pay attention to the page that appears and describe your observations.



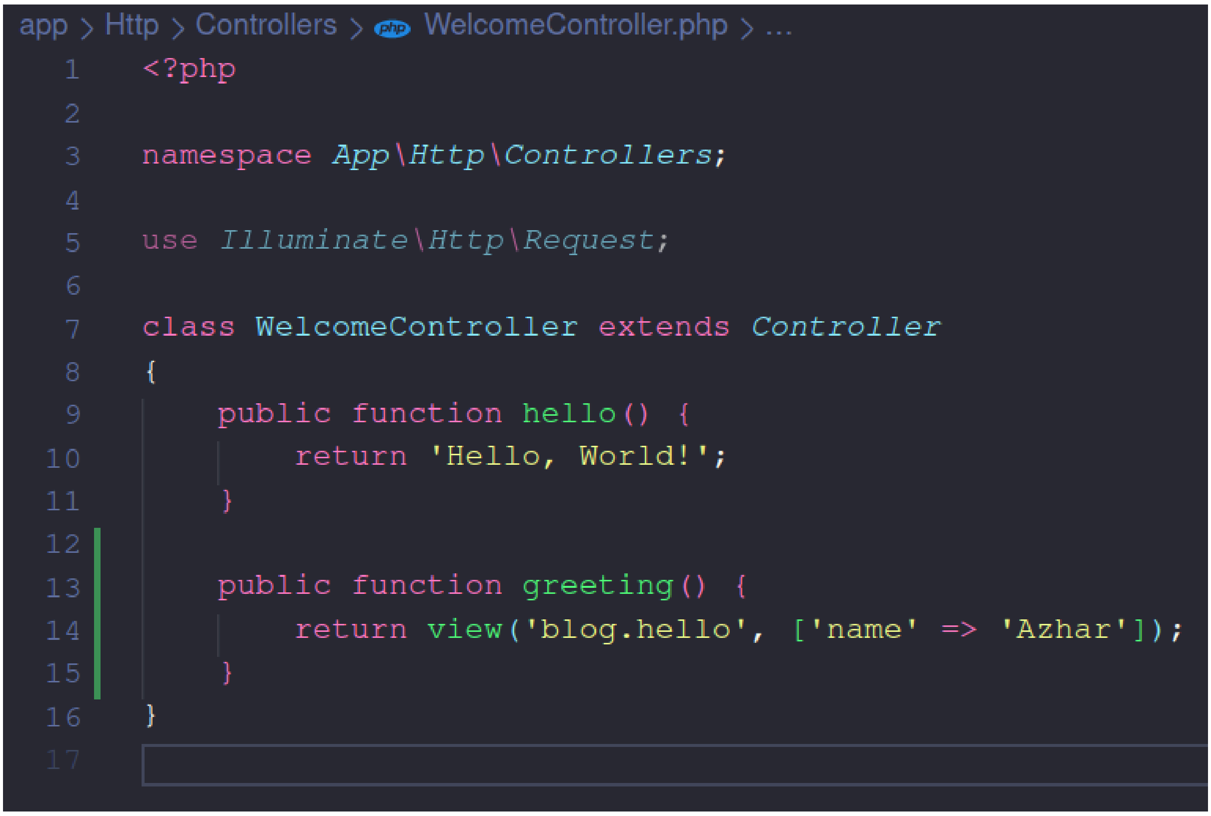
Hello, Azhar

- The result and url path still same, but the difference is hello blade that we create, it moves to inside blog directory. Then we add blog.hello on router view.

Displaying a View from a Controller

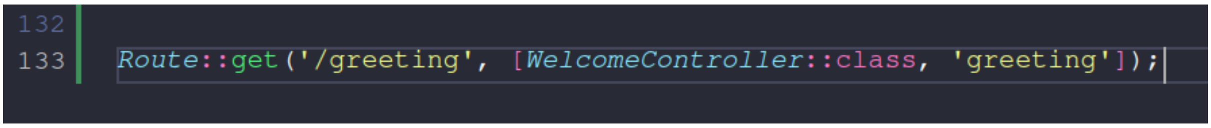
Practicum Steps:

a. Open WelcomeController.php and add a new function, greeting.



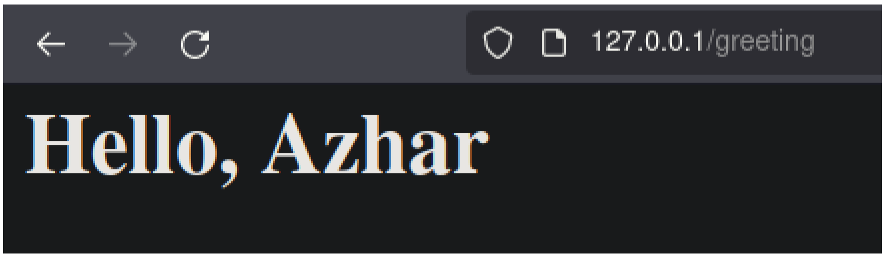
```php
<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;

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

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

b. Change the /greeting route and navigate to the WelcomeController in the greeting function.



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

c. Run the code by opening the url localhost/PWL_2024/public/greeting. Pay attention to the page that appears and describe your observations.



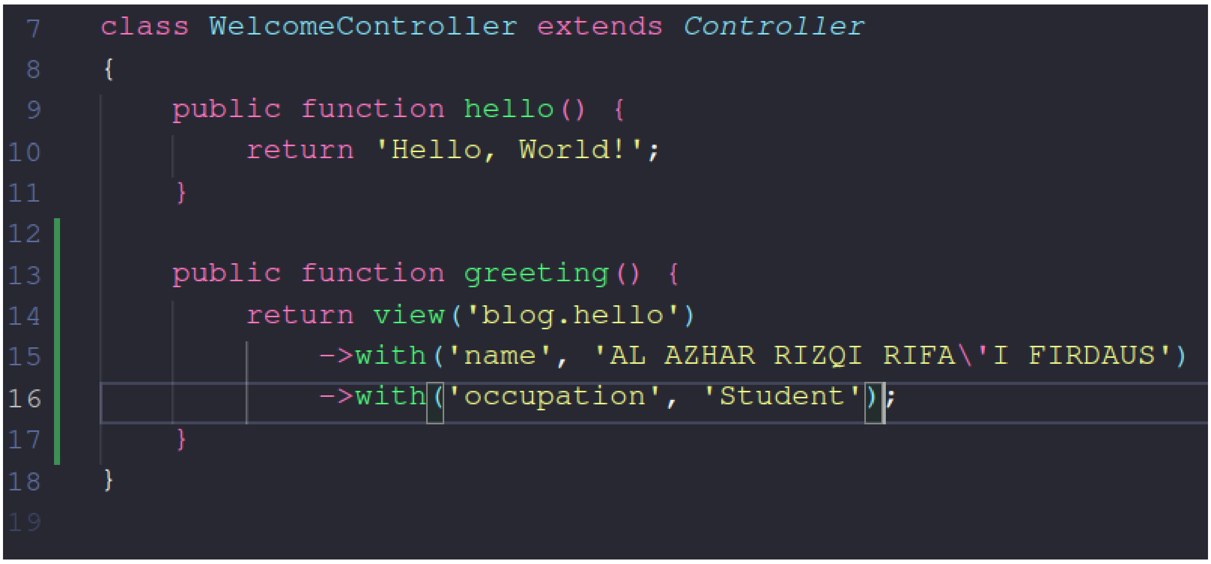
Hello, Azhar

- The result and url path still same with previous. But now, we call view inside controller, then we call controller inside routes.

Forward data to view

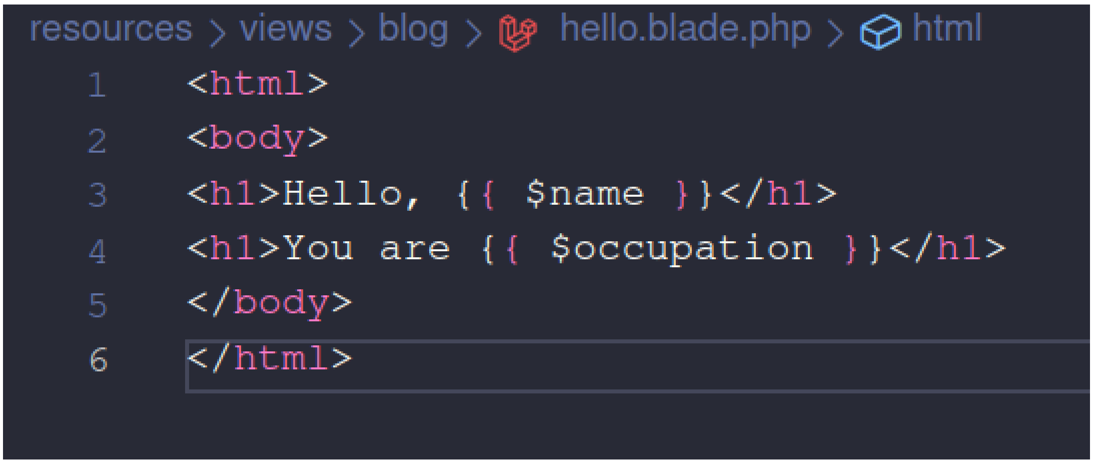
Practicum Steps:

a. Open WelcomeController.php and add change greeting function.



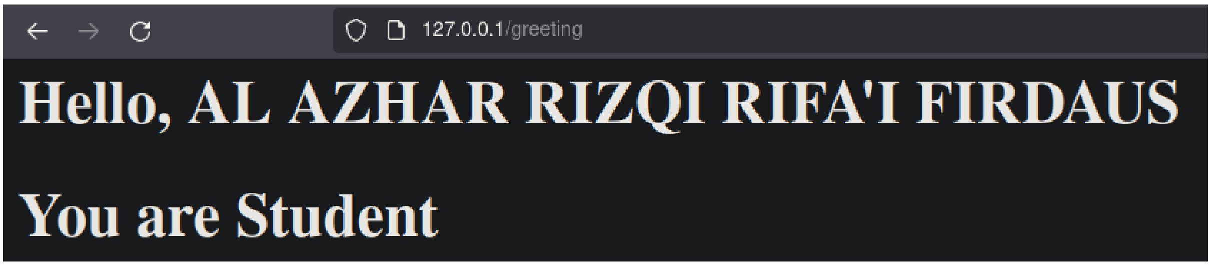
```
class WelcomeController extends Controller
{
    public function hello() {
        return 'Hello, World!';
    }

    public function greeting() {
        return view('blog.hello')
            ->with('name', 'AL AZHAR RIZQI RIFA\'I FIRDAUS')
            ->with('occupation', 'Student');
    }
}
```

b. Change the hello.blade.php to display two parameters.



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

c. Run the code by opening the url localhost/PWL_2024/public/greeting. Pay attention to the page that appears and describe your observations.



Hello, AL AZHAR RIZQI RIFA'I FIRDAUS

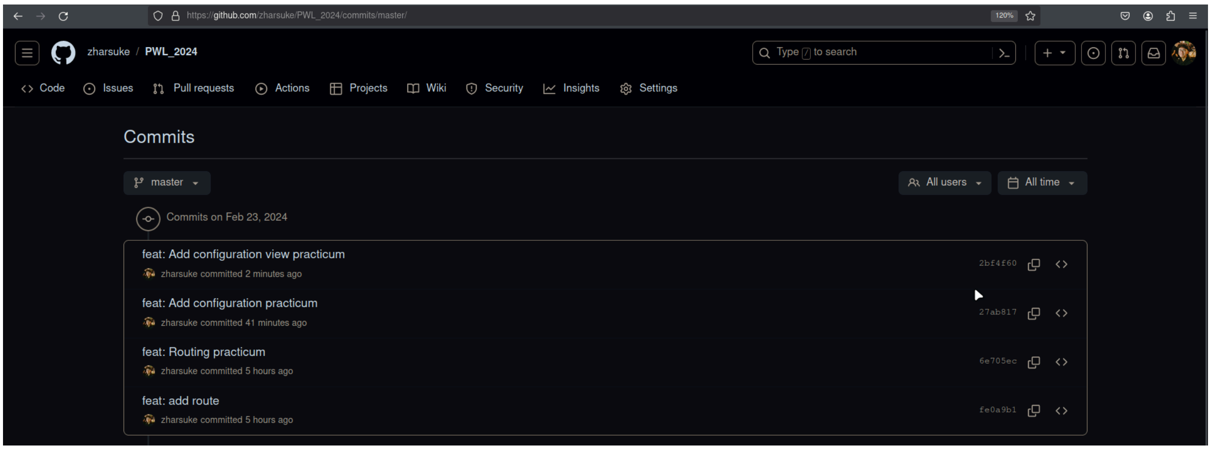
You are Student

- Inside greeting function, we use with() method to make array that contains key value. Then we call both key to view, to display it.

PRACTICUM QUESTIONS

1. Run the Practicum steps on the jobsheet above. Sync changes to PWL_2024 project to Github.

Link repository github: https://github.com/zharsuke/PWL_2024



Commits

Commits on Feb 23, 2024

feat: Add configuration view practicum
zharsuke committed 2 minutes ago
2bf4f60

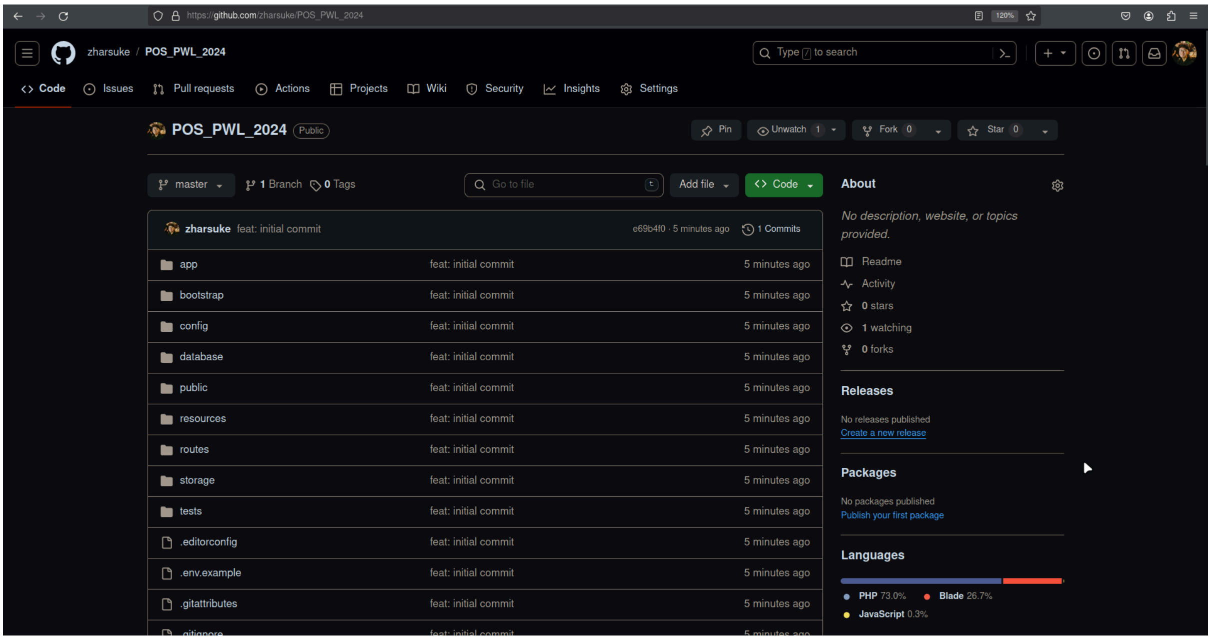
feat: Add configuration practicum
zharsuke committed 41 minutes ago
27ab817

feat: Routing practicum
zharsuke committed 5 hours ago
6e705ec

feat: add route
zharsuke committed 5 hours ago
fe0a9b1

2. Create a new project named POS. This project is a Point of Sales application that is used to help sales.

Link repository github: https://github.com/zharsuke/POS_PWL_2024



zharsuke feat: initial commit e69b4f0 · 5 minutes ago 1 Commits

app	feat: initial commit	5 minutes ago
bootstrap	feat: initial commit	5 minutes ago
config	feat: initial commit	5 minutes ago
database	feat: initial commit	5 minutes ago
public	feat: initial commit	5 minutes ago
resources	feat: initial commit	5 minutes ago
routes	feat: initial commit	5 minutes ago
storage	feat: initial commit	5 minutes ago
tests	feat: initial commit	5 minutes ago
.editorconfig	feat: initial commit	5 minutes ago
.env.example	feat: initial commit	5 minutes ago
.gitattributes	feat: initial commit	5 minutes ago
.gitignore	feat: initial commit	5 minutes ago

About

No description, website, or topics provided.

Readme
Activity
0 stars
1 watching
0 forks

Releases

No releases published
Create a new release

Packages

No packages published
Publish your first package

Languages

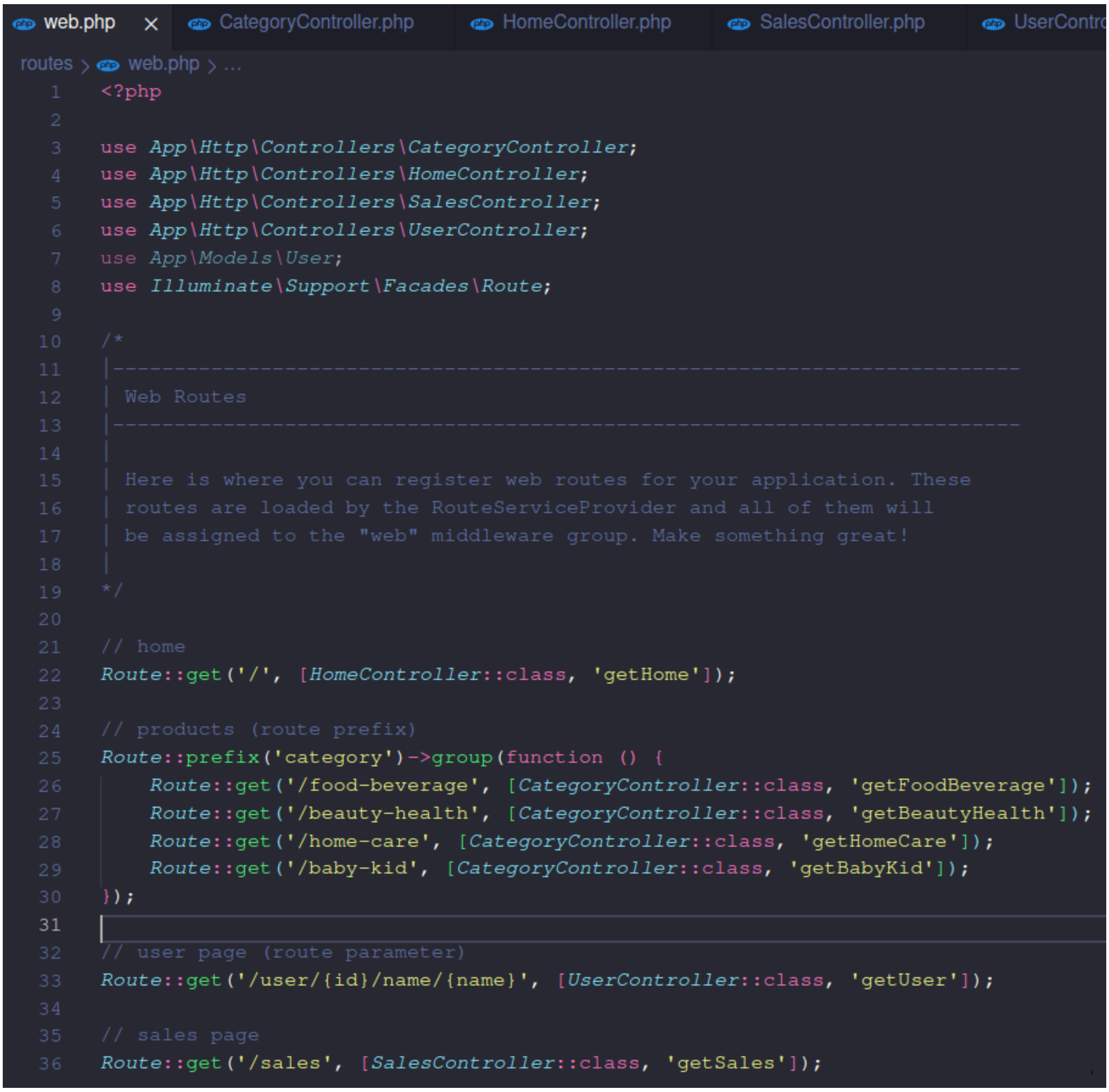
PHP 73.0% Blade 26.7%
JavaScript 0.3%

3. Create several routes, controllers, and views according to the following conditions.

1	Home Page Display the start page of the website
2	Halaman Products Display product list (route prefix) /category/food-beverage /category/beauty-health /category/home-care /category/baby-kid
3	User Page Display user profiles (route param) /user/{id}/name/{name}
4	Sales Page Display the POS transaction page

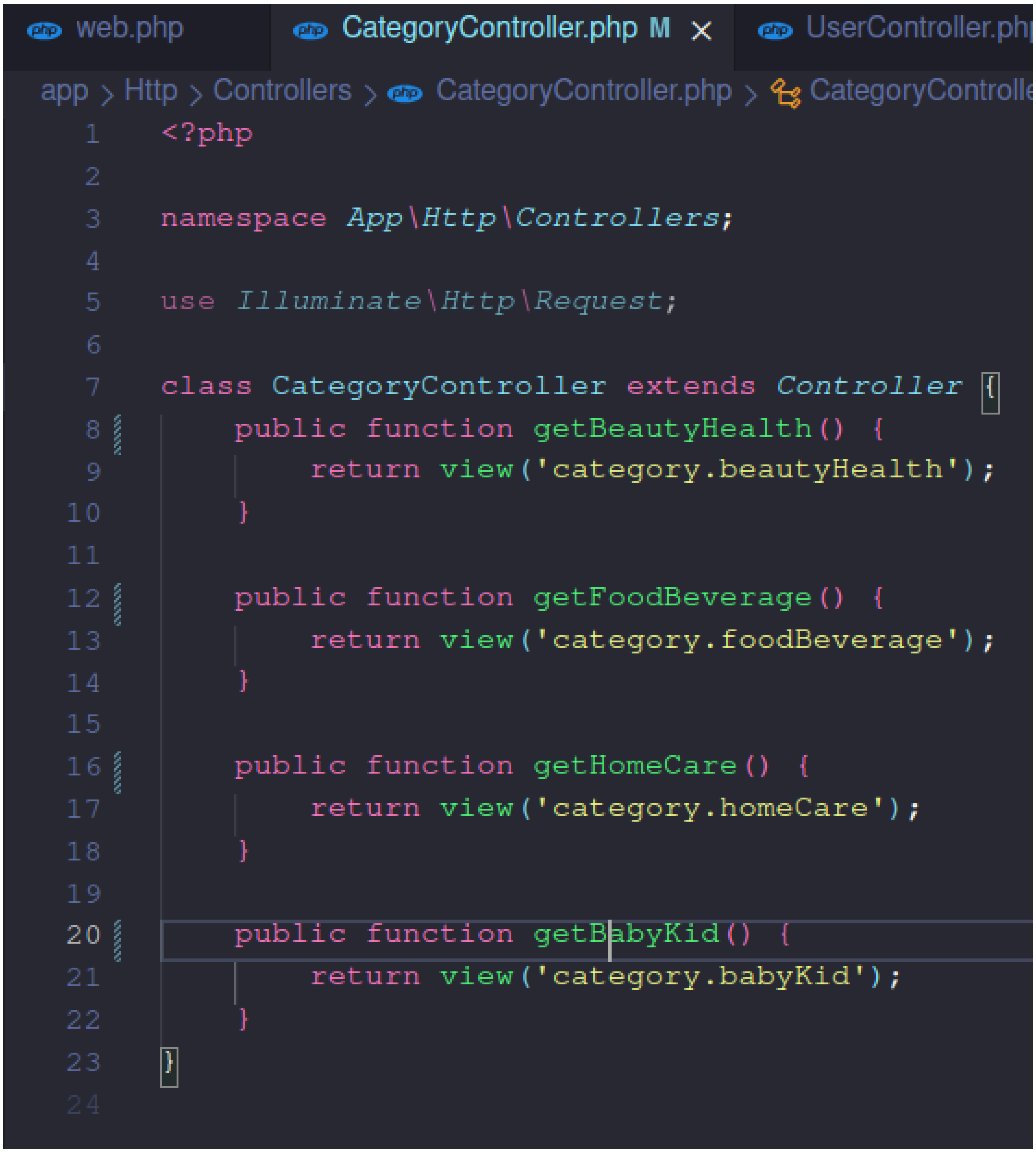
4. The route performs functions on different controllers on each page.

Code :



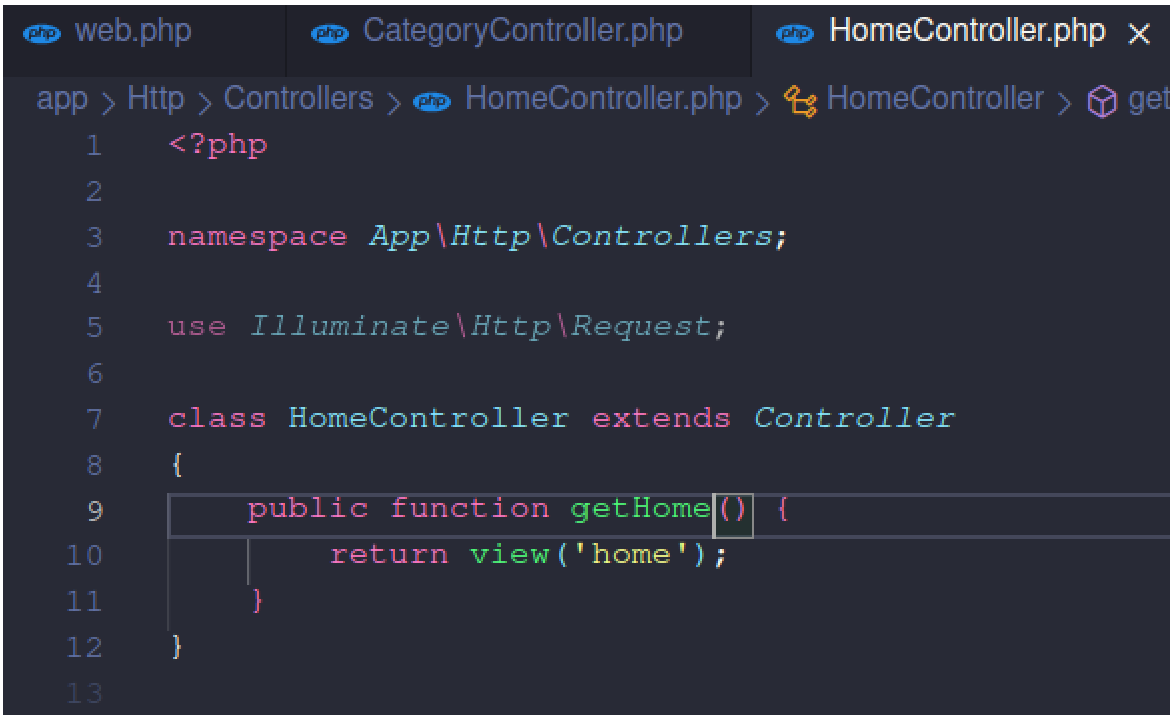
```php
<?php

use App\Http\Controllers\CategoryController;
use App\Http\Controllers\HomeController;
use App\Http\Controllers\SalesController;
use App\Http\Controllers\UserController;
use App\Models\User;
use Illuminate\Support\Facades\Route;

/*
|--------------------------------------------------------------------------
| Web Routes
|--------------------------------------------------------------------------
|
| Here is where you can register web routes for your application. These
| routes are loaded by the RouteServiceProvider and all of them will
| be assigned to the "web" middleware group. Make something great!
|
*/

// home
Route::get('/', [HomeController::class, 'getHome']);

// products (route prefix)
Route::prefix('category')->group(function () {
    Route::get('/food-beverage', [CategoryController::class, 'getFoodBeverage']);
    Route::get('/beauty-health', [CategoryController::class, 'getBeautyHealth']);
    Route::get('/home-care', [CategoryController::class, 'getHomeCare']);
    Route::get('/baby-kid', [CategoryController::class, 'getBabyKid']);
});

// user page (route parameter)
Route::get('/user/{id}/name/{name}', [UserController::class, 'getUser']);

// sales page
Route::get('/sales', [SalesController::class, 'getSales']);
```

5. The function on the Controller will call the view according to the page to be displayed.

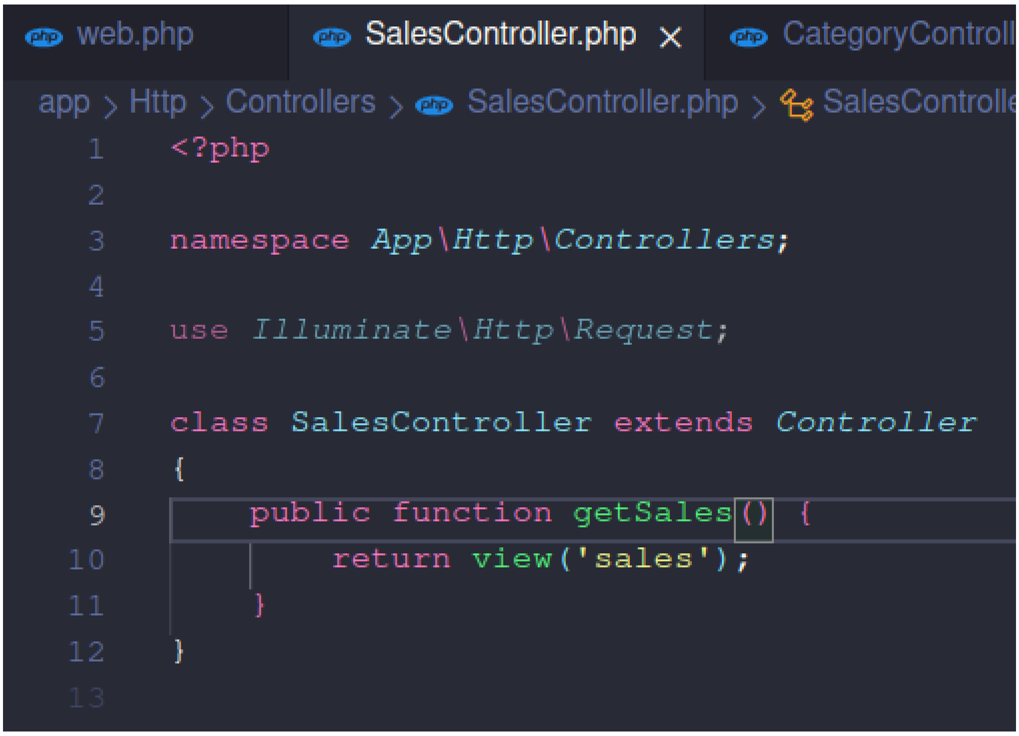


web.php
CategoryController.php M ×
UserController.ph
app > Http > Controllers > CategoryController.php > CategoryControlle

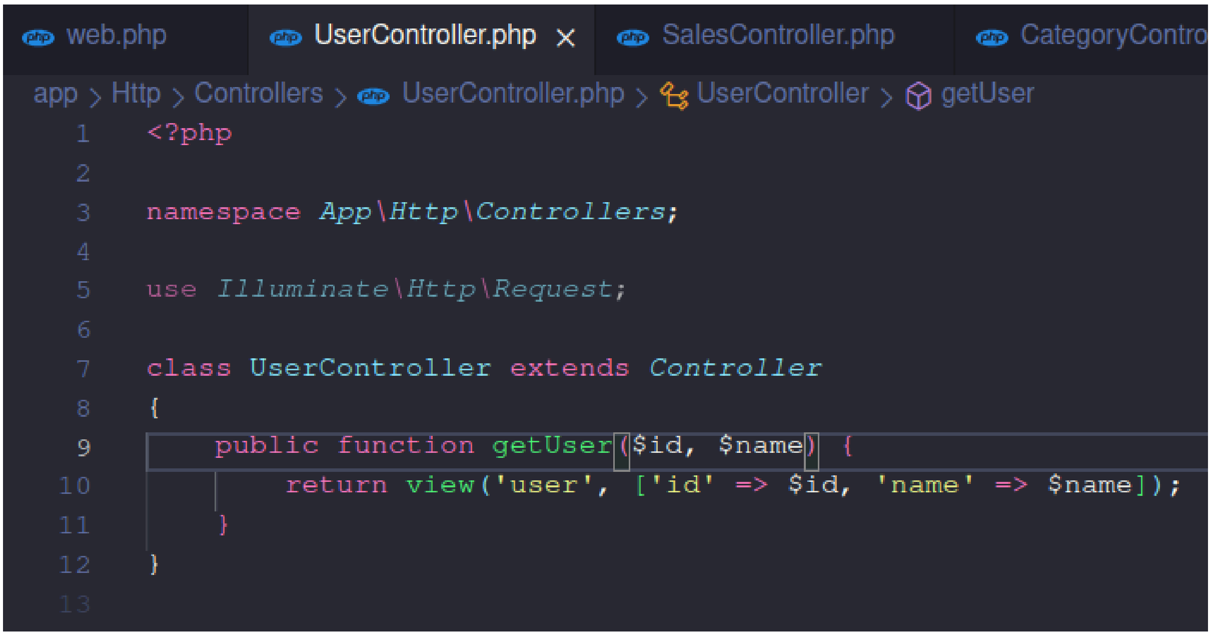
```php
<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;

class CategoryController extends Controller {
    public function getBeautyHealth() {
        return view('category.beautyHealth');
    }

    public function getFoodBeverage() {
        return view('category.foodBeverage');
    }

    public function getHomeCare() {
        return view('category.homeCare');
    }

    public function getBabyKid() {
        return view('category.babyKid');
    }
}
```


web.php
CategoryController.php
HomeController.php
app > Http > Controllers > HomeController.php > HomeController > get

```php
<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;

class HomeController extends Controller
{
    public function getHome() {
        return view('home');
    }
}
```

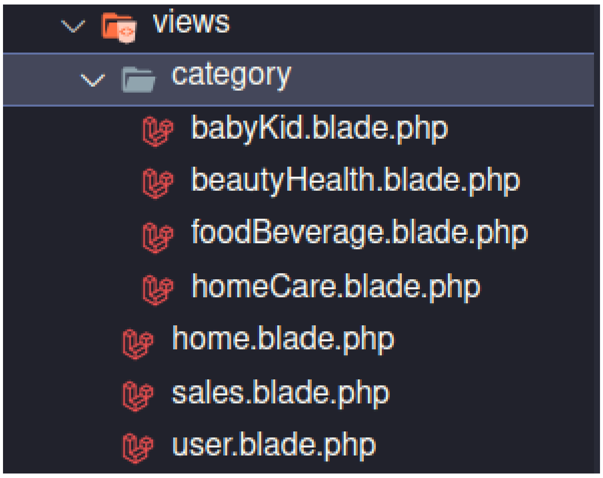

web.php
SalesController.php
CategoryControll
app > Http > Controllers > SalesController.php > SalesControlle

```php
<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;

class SalesController extends Controller
{
    public function getSales() {
        return view('sales');
    }
}
```


web.php
UserController.php
SalesController.php
CategoryContro
app > Http > Controllers > UserController.php > UserController > getUser

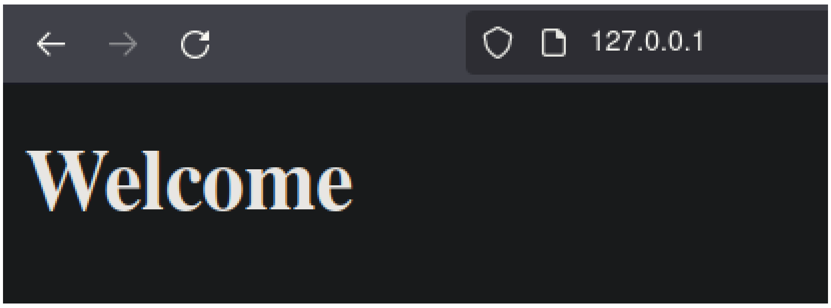
```php
<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;

class UserController extends Controller
{
    public function getUser($id, $name) {
        return view('user', ['id' => $id, 'name' => $name]);
    }
}
```

Views Folder Structure:



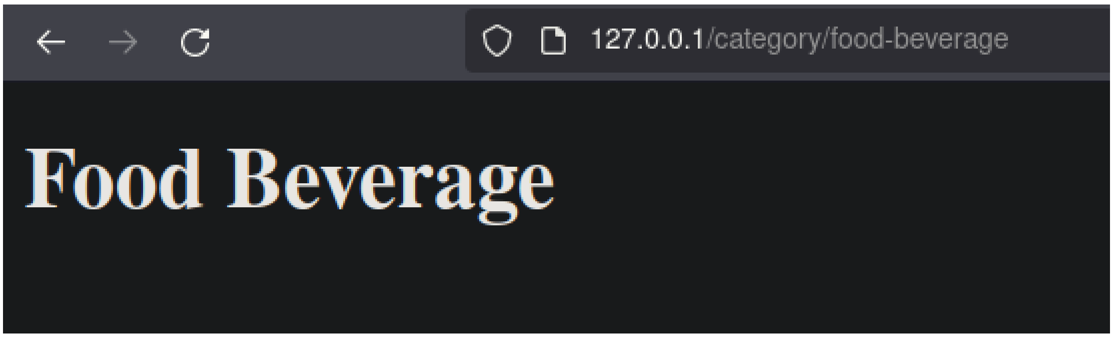
```
views
  category
    babyKid.blade.php
    beautyHealth.blade.php
    foodBeverage.blade.php
    homeCare.blade.php
  home.blade.php
  sales.blade.php
  user.blade.php
```

Result:



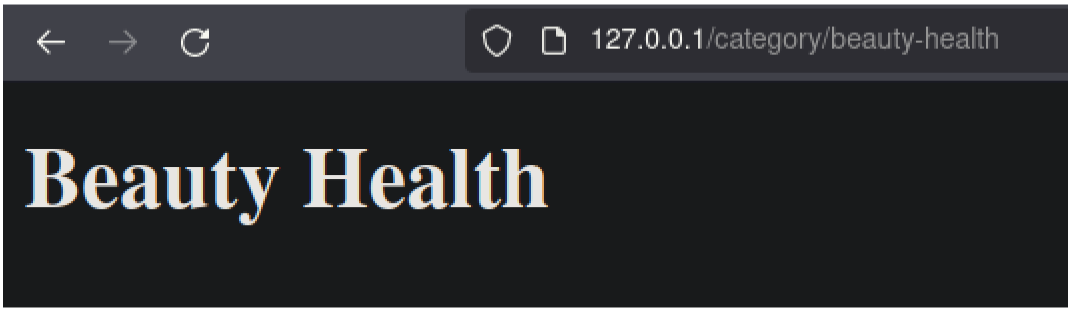
127.0.0.1

Welcome



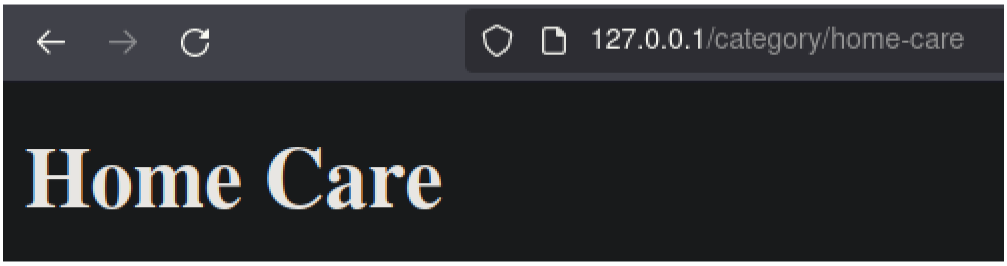
127.0.0.1/category/food-beverage

Food Beverage



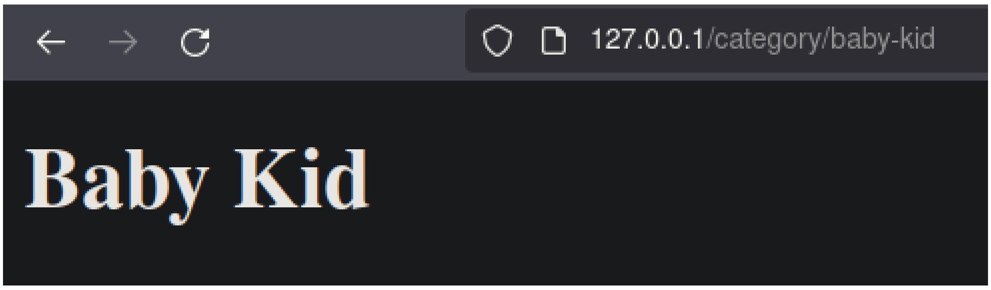
127.0.0.1/category/beauty-health

Beauty Health



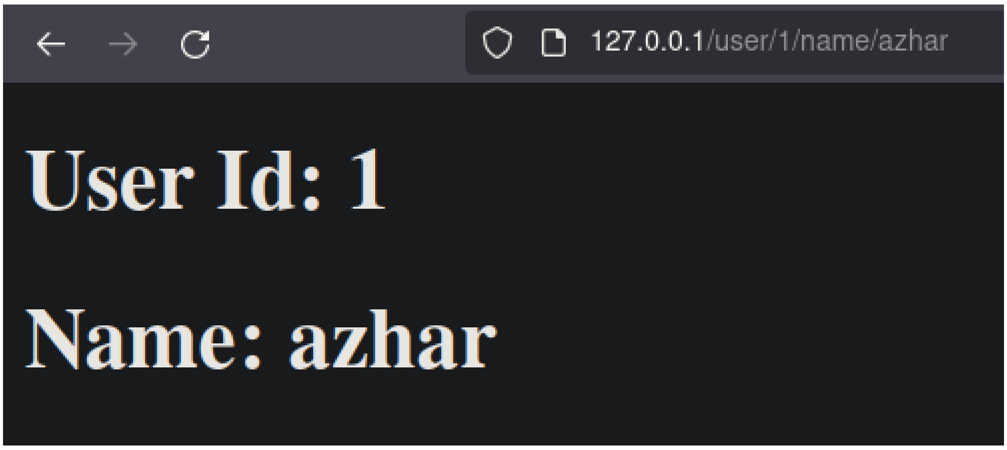
127.0.0.1/category/home-care

Home Care



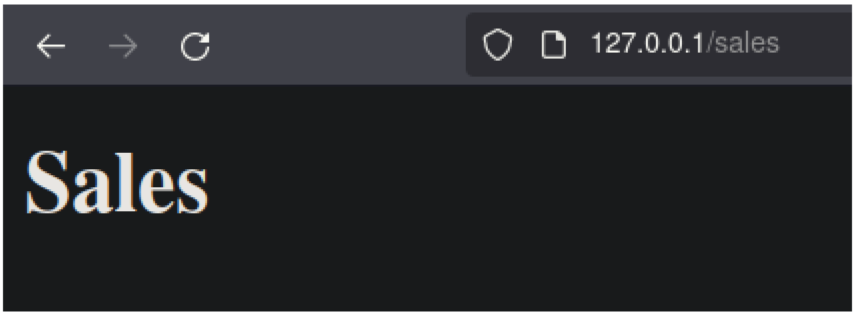
127.0.0.1/category/baby-kid

Baby Kid



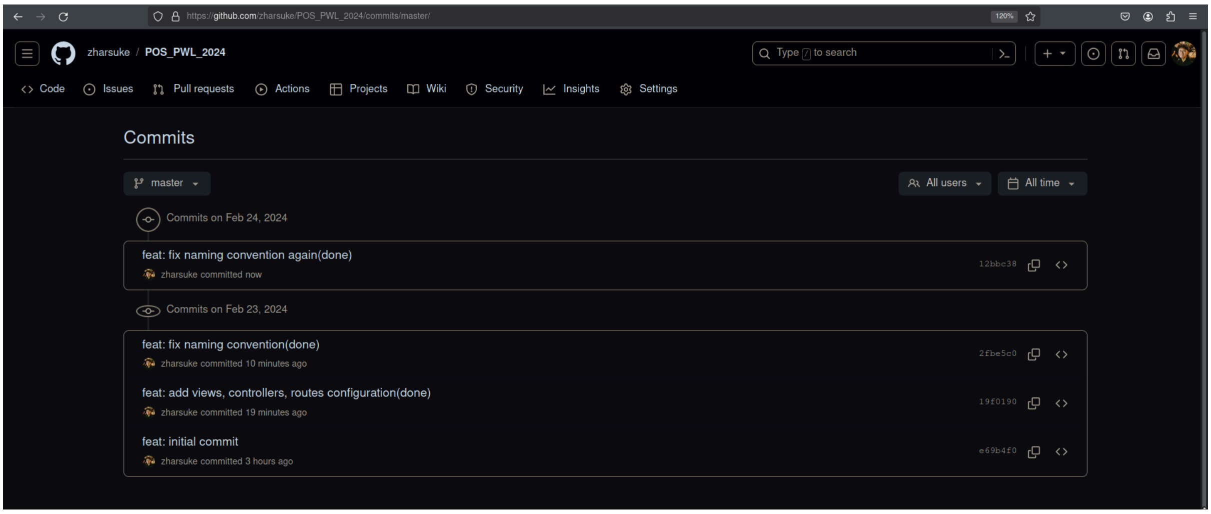
User Id: 1

Name: azhar



Sales

6. Save any changes made to the POS project in Git, sync the changes to Github.



Commits

master

All users

All time

Commits on Feb 24, 2024

feat: fix naming convention again(done)
zharsuke committed now
12bbc38

Commits on Feb 23, 2024

feat: fix naming convention(done)
zharsuke committed 10 minutes ago
2fbe5c0

feat: add views, controllers, routes configuration(done)
zharsuke committed 19 minutes ago
19f0190

feat: initial commit
zharsuke committed 3 hours ago
e69b4f0